Appendix 2

Code For AddRecord2.java

```
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;
import java.sql.Connection;
import java.sql.Statement;

import javax.swing.BorderFactory;
import javax.swing.ButtonGroup;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JComboBox;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JTextField;
import javax.swing.border.Border;

public class AddRecord_2 extends JPanel {

	private JLabel labelNo_of_forms;
	JTextField txtNo_of_forms;
	private JLabel printStyle;
	JComboBox comboPrintStyle;
	private JLabel paperUsed;
	JComboBox combopaperUsed;
	private JLabel no_of_col;
	JTextField txtCol;
	private JLabel print_qt;
	JTextField txtPrintQt;
	private JLabel plate;
	JComboBox comboPlate;
	private JLabel paper_Print;
	JTextField txtPaper_Print;
	private JLabel book_size;
	JTextField txtBookSize;
	private JLabel no_of_pages;
	JTextField txtNo_of_pages;
	JRadioButton plateNEW;
	JRadioButton plateOLD;
	private JLabel paperRequird;
	JTextField txtPaperReq;
	private JLabel paper_supp;
	JTextField txtPapersupp;
	private JLabel paper_in_sheets;
	JTextField txtPaperSheets;
	private JLabel finish_Quantity;
	JTextField txtfinishQt;
	private JLabel deliveryDate;
	JTextField txtdeliverydate;
	private JLabel status;
	JComboBox comboStatus;
	private JLabel machine;
	JComboBox comboMachine;
	private ButtonGroup bg;
```

```
public AddRecord_2(Connection conn, Statement st) {

Dimension dim = getPreferredSize();
dim.width = 750;
dim.height = 600;
setPreferredSize(dim);

labelNo_of_forms = new JLabel("No. of Forms");
labelNo_of_forms.setFont(new Font("Times New Roman", Font.BOLD, 14));
txtNo_of_forms = new JTextField(6);
printStyle = new JLabel("Print Style");
printStyle.setFont(new Font("Times New Roman", Font.BOLD, 14));
comboPrintStyle = new JComboBox();
comboPrintStyle.setFont(new Font("Helvetica", Font.ITALIC, 14));
paperUsed = new JLabel("Paper Used");
paperUsed.setFont(new Font("Times New Roman", Font.BOLD, 14));
combopaperUsed = new JComboBox();
combopaperUsed.setFont(new Font("Helvetica", Font.ITALIC, 14));
no_of_col = new JLabel("No. of Colors");
no_of_col.setFont(new Font("Times New Roman", Font.BOLD, 14));
txtCol = new JTextField(6);
print_qt = new JLabel("Paper Quantity");
print_qt.setFont(new Font("Times New Roman", Font.BOLD, 14));
txtPrintQt = new JTextField(6);
plate = new JLabel("Plate");
plate.setFont(new Font("Times New Roman", Font.BOLD, 14));
comboPlate = new JComboBox();
comboPlate.setFont(new Font("Times", Font.ITALIC, 13));
paper_Print = new JLabel("Paper Print");
paper_Print.setFont(new Font("Times New Roman", Font.BOLD, 14));
txtPaper_Print = new JTextField(6);
book_size = new JLabel("Book Size");
book_size.setFont(new Font("Times New Roman", Font.BOLD, 14));
txtBookSize = new JTextField(6);
no_of_pages = new JLabel("No. of Pages");
no_of_pages.setFont(new Font("Times New Roman", Font.BOLD, 14));
txtNo_of_pages = new JTextField(6);
plateNEW = new JRadioButton("New Plate");
plateNEW.setFont(new Font("Times", Font.BOLD | Font.ITALIC, 14));
plateOLD = new JRadioButton("Old Plate");
plateOLD.setFont(new Font("Times", Font.BOLD | Font.ITALIC, 14));
bg = new ButtonGroup();
bg.add(plateNEW);
bg.add(plateOLD);

paperRequird = new JLabel("Paper Required");
paperRequird.setFont(new Font("Times New Roman", Font.BOLD, 14));
txtPaperReq = new JTextField(6);
paper_supp = new JLabel("Paper Supplied");
paper_supp.setFont(new Font("Times New Roman", Font.BOLD, 14));
txtPapersupp = new JTextField(6);
paper_in_sheets = new JLabel("Paper in Sheets");
paper_in_sheets.setFont(new Font("Times New Roman", Font.BOLD, 14));
txtPaperSheets = new JTextField(6);
finish_Quantity = new JLabel("Finish Quantity");
finish_Quantity.setFont(new Font("Times New Roman", Font.BOLD, 14));
```

```
txtfinishQt = new JTextField(6);
deliveryDate = new JLabel("Delivery Date");
deliveryDate.setFont(new Font("Times New Roman", Font.BOLD, 14));
txtdeliverydate = new JTextField(6);
txtdeliverydate.setText("YYYY/MM/DD");
//labelJobName.setFont(new Font("Times New Roman", Font.BOLD, 14));
txtdeliverydate.setCaretColor(Color.GRAY);
status = new JLabel("Status");
status.setFont(new Font("Times New Roman", Font.BOLD, 14));
comboStatus = new JComboBox();
comboStatus.setFont(new Font("Helvetica", Font.ITALIC, 14));
machine = new JLabel("Machine");
//machine.setFont(new Font("Times", Font.BOLD , 14));
machine.setFont(new Font("Times New Roman", Font.BOLD, 14));
comboMachine = new JComboBox();
comboMachine.setFont(new Font("Helvetica", Font.ITALIC, 14));

Border inner = BorderFactory.createTitledBorder("Job Details");
Border outer = BorderFactory.createEmptyBorder(3, 3, 3, 3);
setBorder(BorderFactory.createCompoundBorder(outer, inner));

DefaultComboBoxModel comboModel1 = new DefaultComboBoxModel();
comboModel1.addElement("W/TUMBLW");
comboModel1.addElement("W/TURN");
comboPrintStyle.setModel(comboModel1);
comboPrintStyle.setEditable(true);

DefaultComboBoxModel comboModel2 = new DefaultComboBoxModel();
comboModel2.addElement("Art Card");
comboModel2.addElement("Matt Card");
comboModel2.addElement("Maplitho");
comboModel2.addElement("Newsprint");
comboModel2.addElement("Natural");
combopaperUsed.setModel(comboModel2);
combopaperUsed.setEditable(true);

DefaultComboBoxModel comboModel3 = new DefaultComboBoxModel();
comboModel3.addElement("CTP");
comboModel3.addElement("YIPON");
comboModel3.addElement("PS +/-");
comboPlate.setModel(comboModel3);
comboPlate.setEditable(true);

DefaultComboBoxModel comboModel4 = new DefaultComboBoxModel();
comboModel4.addElement("Pending");
comboModel4.addElement("Ready");
//comboModel4.addElement("PS +/-");
comboStatus.setModel(comboModel4);
comboStatus.setEditable(true);

DefaultComboBoxModel comboModel5 = new DefaultComboBoxModel();
comboModel5.addElement("H1");
comboModel5.addElement("H2");
comboModel5.addElement("H4");
comboModel5.addElement("ZP");
comboModel5.addElement("Single Color 28 x 40");
```

```
comboModel5.addElement("2 Color 25 x 36");
comboModel5.addElement("WEB-1");
comboModel5.addElement("WEB-2");
comboModel5.addElement("WEB-3");
comboModel5.addElement("PressLine");
comboMachine.setModel(comboModel5);
comboMachine.setEditable(true);

/////////FIRST ROW//////////

setLayout(new GridBagLayout());
GridBagConstraints gc = new GridBagConstraints();
gc.weightx = 1;
gc.weighty = 0.1;

//gc.gridwidth=1;
gc.gridx = 0;
gc.gridy = 0;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(labelNo_of_forms, gc);

gc.gridx = 1;
gc.gridy = 0;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(txtNo_of_forms, gc);

gc.gridx = 3;
gc.gridy = 0;
gc.weightx = 1;
gc.weighty = 0.1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(printStyle, gc);

gc.gridx = 4;
gc.gridy = 0;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(comboPrintStyle, gc);

gc.gridx = 6;
gc.gridy = 0;
gc.weightx = 1;
gc.weighty = 0.1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(paperUsed, gc);

gc.gridx = 7;
gc.gridy = 0;
//gc.fill=GridBagConstraints.WEST;
```

```
gc.anchor = GridBagConstraints.LINE_START;
add(combopaperUsed, gc);

///////SECOND ROW////////

gc.weightx = 1;
gc.weighty = 0.1;

//gc.gridwidth=1;
gc.gridx = 0;
gc.gridy = 1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(no_of_col, gc);

gc.gridx = 1;
gc.gridy = 1;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(txtCol, gc);

gc.gridx = 3;
gc.gridy = 1;
gc.weightx = 1;
gc.weighty = 0.1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(print_qt, gc);

gc.gridx = 4;
gc.gridy = 1;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(txtPrintQt, gc);

gc.gridx = 6;
gc.gridy = 1;
gc.weightx = 1;
gc.weighty = 0.1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(plate, gc);

gc.gridx = 7;
gc.gridy = 1;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(comboPlate, gc);

///////THIRD ROW////////

gc.weightx = 1;
gc.weighty = 0.1;
```

```
//gc.gridwidth=1;
gc.gridx = 0;
gc.gridy = 2;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(paper_Print, gc);

gc.gridx = 1;
gc.gridy = 2;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(txtPaper_Print, gc);

gc.gridx = 3;
gc.gridy = 2;
gc.weightx = 1;
gc.weighty = 0.1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(book_size, gc);

gc.gridx = 4;
gc.gridy = 2;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(txtBookSize, gc);

gc.gridx = 6;
gc.gridy = 2;
gc.weightx = 1;
gc.weighty = 0.1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(no_of_pages, gc);

gc.gridx = 7;
gc.gridy = 2;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(txtNo_of_pages, gc);
///////FOURTH ROW////////

gc.weightx = 1;
gc.weighty = 0.1;

//gc.gridwidth=1;
gc.gridx = 0;
gc.gridy = 3;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(plateNEW, gc);
```

```
gc.gridx = 1;
gc.gridy = 3;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(plateOLD, gc);

gc.gridx = 3;
gc.gridy = 3;
gc.weightx = 1;
gc.weighty = 0.1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(paperRequird, gc);

gc.gridx = 4;
gc.gridy = 3;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(txtPaperReq, gc);

gc.gridx = 6;
gc.gridy = 3;
gc.weightx = 1;
gc.weighty = 0.1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(paper_supp, gc);

gc.gridx = 7;
gc.gridy = 3;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(txtPapersupp, gc);

///////FIFTH ROW////////

gc.weightx = 1;
gc.weighty = 0.1;

//gc.gridwidth=1;
gc.gridx = 0;
gc.gridy = 4;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(paper_in_sheets, gc);

gc.gridx = 1;
gc.gridy = 4;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(txtPaperSheets, gc);
```

```
gc.gridx = 3;
gc.gridy = 4;
gc.weightx = 1;
gc.weighty = 0.1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(finish_Quantity, gc);

gc.gridx = 4;
gc.gridy = 4;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(txtfinishQt, gc);

gc.gridx = 6;
gc.gridy = 4;
gc.weightx = 1;
gc.weighty = 0.1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(deliveryDate, gc);

gc.gridx = 7;
gc.gridy = 4;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(txtdeliverydate, gc);

///////SIXTH ROW////////

gc.weightx = 1;
gc.weighty = 0.1;

//gc.gridwidth=1;
gc.gridx = 3;
gc.gridy = 5;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
add(status, gc);

gc.gridx = 4;
gc.gridy = 5;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(comboStatus, gc);

gc.gridx = 6;
gc.gridy = 5;
gc.weightx = 1;
gc.weighty = 0.1;
//gc.fill=GridBagConstraints.EAST;
gc.anchor = GridBagConstraints.LINE_END;
gc.insets = new Insets(0, 0, 0, 5);
```

```
add(machine, gc);

gc.gridx = 7;
gc.gridy = 5;
//gc.fill=GridBagConstraints.WEST;
gc.anchor = GridBagConstraints.LINE_START;
add(comboMachine, gc);

}
public String getPlate() {
return (String) comboPlate.getSelectedItem();

}

public String Psheets() {
return txtPaperSheets.getText();
}

public String no_ofForms() {
return txtNo_of_forms.getText();
}

public String ComboPrintStyle() {
return (String) comboPrintStyle.getSelectedItem();
}

public String CombopaperUsed() {
return (String) combopaperUsed.getSelectedItem();
}

public String TxtCol() {
return txtCol.getText();
}

public String TxtPrintQt() {
return txtPrintQt.getText();
}

public String ComboPlate() {
return (String) comboPlate.getSelectedItem();
}

public String TxtPaper_Print() {
return txtPaper_Print.getText();
}

public String TxtBookSize() {
return txtBookSize.getText();
}

public String TxtNo_of_pages() {
return txtNo_of_pages.getText();
}

public String TxtPaperReq() {
return txtPaperReq.getText();
```

```
}

public String TxtPapersupp() {
return txtPapersupp.getText();
}

public String TxtfinishQt() {
return txtfinishQt.getText();
}

public String Txtdeliverydate() {
String r = "";
if (txtdeliverydate.getText().matches(".*\\d+.*")) {
r = txtdeliverydate.getText();
} else {
JOptionPane.showMessageDialog(txtdeliverydate, "InCorrect Date Format");
}
return r;
}

public String ComboStatus() {
return (String) comboStatus.getSelectedItem();
}

public String ComboMachine() {
return (String) comboMachine.getSelectedItem();
}

public String PlateOld() {
String r;
if (plateNEW.isSelected()) {
r = plateNEW.getText();
} else {
r = plateOLD.getText();
}
return r;
}

}
```

Code For AddRecord_3.java

```
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.EventQueue;
import java.sql.Connection;
import java.sql.Statement;
import java.util.Arrays;

import javax.swing.BorderFactory;
import javax.swing.GroupLayout.Alignment;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.border.Border;
import javax.swing.border.EmptyBorder;
import java.awt.GridBagLayout;
```

```
import javax.swing.JTextArea;
import java.awt.GridBagConstraints;
import java.awt.Insets;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.Font;

public class AddRecord_3 extends JPanel {

private JPanel contentPane;
JButton btnA;
JButton btnB;
JButton btnC;
JButton btnD;
private String a1;
private String a2;
public AddRecord_3(Connection conn, Statement st) {

Border inner = BorderFactory.createTitledBorder("Choose Space");
Border outer = BorderFactory.createEmptyBorder(3, 3, 3, 3);
setBorder(BorderFactory.createCompoundBorder(outer, inner));

//setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setBounds(100, 100, 610, 392);
GridBagLayout gridBagLayout = new GridBagLayout();
gridBagLayout.columnWidths = new int[] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
gridBagLayout.rowHeights = new int[] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
gridBagLayout.columnWeights = new double[] { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
Double.MIN_VALUE };
gridBagLayout.rowWeights = new double[] { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, Double.MIN_VALUE };
setLayout(gridBagLayout);

JLabel lblNewLabel = new JLabel("");
lblNewLabel.setIcon(new ImageIcon("/Users/DGair/Desktop/New Java Programs /SoftwareSIS/AREA.png"));
GridBagConstraints gbc_lblNewLabel = new GridBagConstraints();
gbc_lblNewLabel.insets = new Insets(0, 0, 5, 5);
gbc_lblNewLabel.gridx = 4;
gbc_lblNewLabel.gridy = 4;
add(lblNewLabel, gbc_lblNewLabel);

btnA = new JButton("A");
btnA.setFont(new Font("Lucida Grande", Font.PLAIN, 13));
btnA.setForeground(new Color(0, 0, 0));
btnA.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
btnA.setForeground(Color.red);
btnA.setFont(new Font("Lucida Grande", Font.BOLD, 18));
returnButtonText(btnA);
}
});
GridBagConstraints gbc_btnA = new GridBagConstraints();
gbc_btnA.insets = new Insets(0, 0, 5, 5);
gbc_btnA.gridx = 3;
gbc_btnA.gridy = 5;
add(btnA, gbc_btnA);
btnB = new JButton("B");
GridBagConstraints gbc_btnB = new GridBagConstraints();
gbc_btnB.insets = new Insets(0, 0, 5, 5);
gbc_btnB.gridx = 4;
gbc_btnB.gridy = 5;
add(btnB, gbc_btnB);
btnB.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
btnB.setForeground(Color.red);
btnB.setFont(new Font("Lucida Grande", Font.BOLD, 18));
returnButtonText_1(btnB);
}
```

```
});

btnC = new JButton("C");
GridBagConstraints gbc_btnC = new GridBagConstraints();
gbc_btnC.insets = new Insets(0, 0, 5, 5);
gbc_btnC.gridx = 5;
gbc_btnC.gridy = 5;
add(btnC, gbc_btnC);
btnC.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
btnC.setForeground(Color.red);
btnC.setFont(new Font("Lucida Grande", Font.BOLD, 18));
returnButtonText(btnC);
}
});

btnD = new JButton("D");
GridBagConstraints gbc_btnD = new GridBagConstraints();
gbc_btnD.insets = new Insets(0, 0, 5, 5);
gbc_btnD.gridx = 4;
gbc_btnD.gridy = 6;
add(btnD, gbc_btnD);
btnD.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
btnD.setForeground(Color.red);
btnD.setFont(new Font("Lucida Grande", Font.BOLD, 18));
returnButtonText_1(btnD);
}
});
contentPane = new JPanel();
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
//setContentPane(contentPane);
GridBagLayout gbl_contentPane = new GridBagLayout();
gbl_contentPane.columnWidths = new int[] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
gbl_contentPane.rowHeights = new int[] { 0, 0, 0, 0 };
gbl_contentPane.columnWeights = new double[] { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0,
Double.MIN_VALUE };
gbl_contentPane.rowWeights = new double[] { 0.0, 0.0, 1.0, Double.MIN_VALUE };
contentPane.setLayout(gbl_contentPane);
}
public void reset(JButton A, JButton B) {
A.setForeground(Color.BLACK);
A.setFont(new Font("Lucida Grande", Font.PLAIN, 13));
B.setForeground(Color.BLACK);
B.setFont(new Font("Lucida Grande", Font.PLAIN, 13));

}

public void returnButtonText(JButton A) {
this.setA1(A.getText());
}

public void returnButtonText_1(JButton B) {
this.setA2(B.getText());
}

public String getA1() {
return a1;
}

public void setA1(String a1) {
this.a1 = a1;
}

public String getA2() {
return a2;
}
```

```
public void setA2(String a2) {
this.a2 = a2;
}

}
```

Method Code Generatexls from AdminForm.java

```
public void generateXls(Connection con, Statement st, String filename)
throws SQLException, FileNotFoundException, IOException {
try {
System.out.println("kjwdn 0");
HSSFWorkbook hwb = new HSSFWorkbook();
HSSFSheet sheet = hwb.createSheet("Admin sheet");
HSSFRow rowhead = sheet.createRow((short) 0);
rowhead.createCell((short) 0).setCellValue("Job No.");
rowhead.createCell((short) 1).setCellValue("Job Date");
rowhead.createCell((short) 2).setCellValue("Client Name");
rowhead.createCell((short) 3).setCellValue("Job Name");
rowhead.createCell((short) 4).setCellValue("No. Of Forms ");
rowhead.createCell((short) 5).setCellValue("Print Style");
rowhead.createCell((short) 6).setCellValue("Paper Used");
rowhead.createCell((short) 7).setCellValue("No. of Colors");
rowhead.createCell((short) 8).setCellValue("Paper Qt.");
rowhead.createCell((short) 9).setCellValue("Plate");
rowhead.createCell((short) 10).setCellValue("Paper Print");
rowhead.createCell((short) 11).setCellValue("Book Size");
rowhead.createCell((short) 12).setCellValue("No. Of Pages");
rowhead.createCell((short) 13).setCellValue("Plate new/old");
rowhead.createCell((short) 14).setCellValue("Plate Required");
rowhead.createCell((short) 15).setCellValue("Paper Supplied");
rowhead.createCell((short) 16).setCellValue("Paper in Sheets");
rowhead.createCell((short) 17).setCellValue("Finish Qt.");
rowhead.createCell((short) 18).setCellValue("Delivery Date");
rowhead.createCell((short) 19).setCellValue("Status");
rowhead.createCell((short) 20).setCellValue("Machine");
rowhead.createCell((short) 21).setCellValue("Team Leader Name");
String query = "Select * from SisClient";
Statement statement = con.createStatement();
ResultSet rs = statement.executeQuery(query);
//System.out.println("kjwdn 1" +rs.getString(4));
int i = 1;
while (rs.next()) {
System.out.println("kjwdn 2");
HSSFRow row = sheet.createRow((short) i);
row.createCell((short) 0).setCellValue(Integer.toString(rs.getInt("Job_no")));
row.createCell((short) 1).setCellValue(rs.getString("Job_date"));
row.createCell((short) 2).setCellValue(rs.getString("Client_name"));
row.createCell((short) 3).setCellValue(rs.getString("Job_name"));
row.createCell((short) 4).setCellValue(rs.getString("No_of_forms"));
row.createCell((short) 5).setCellValue(rs.getString("Print_style"));
row.createCell((short) 6).setCellValue(rs.getString("Paper_used"));
row.createCell((short) 7).setCellValue(rs.getString("No_of_col"));
row.createCell((short) 8).setCellValue(rs.getString("Paper_qt"));
row.createCell((short) 9).setCellValue(rs.getString("Plate"));
row.createCell((short) 10).setCellValue(rs.getString("Paper_print"));
row.createCell((short) 11).setCellValue(rs.getString("Book_size"));
row.createCell((short) 12).setCellValue(rs.getString("No_of_pages"));
row.createCell((short) 13).setCellValue(rs.getString("Plate_new/old"));
row.createCell((short) 14).setCellValue(rs.getString("Paper_Required"));
row.createCell((short) 15).setCellValue(rs.getString("Paper_supp"));
row.createCell((short) 16).setCellValue(rs.getString("Paper_in_sheets"));
row.createCell((short) 17).setCellValue(rs.getString("Finish_quantity"));
row.createCell((short) 18).setCellValue(rs.getString("Delivery_date"));
row.createCell((short) 19).setCellValue(rs.getString("Status"));
row.createCell((short) 20).setCellValue(rs.getString("Machine"));
```

```
row.createCell((short) 21).setCellValue(rs.getString("Team_Leader"));
i++;
}
FileOutputStream fileOut = new FileOutputStream(filename);
hwb.write(fileOut);
fileOut.close();
JOptionPane.showMessageDialog(btnGenerateExcelFile, filename + ".xls file generated");

} catch (Exception ex) {
System.out.println(ex);

}
}


public void generateXls(Connection con, Statement st, String filename, String query)
throws SQLException, FileNotFoundException, IOException {
try {

HSSFWorkbook hwb = new HSSFWorkbook();
HSSFSheet sheet = hwb.createSheet("Admin sheet");

HSSFRow rowhead = sheet.createRow((short) 0);
rowhead.createCell((short) 0).setCellValue("Job No.");
rowhead.createCell((short) 1).setCellValue("Job Date");
rowhead.createCell((short) 2).setCellValue("Client Name");
rowhead.createCell((short) 3).setCellValue("Job Name");
rowhead.createCell((short) 4).setCellValue("No. Of Forms ");
rowhead.createCell((short) 5).setCellValue("Print Style");
rowhead.createCell((short) 6).setCellValue("Paper Used");
rowhead.createCell((short) 7).setCellValue("No. of Colors");
rowhead.createCell((short) 8).setCellValue("Paper Qt.");
rowhead.createCell((short) 9).setCellValue("Plate");
rowhead.createCell((short) 10).setCellValue("Paper Print");
rowhead.createCell((short) 11).setCellValue("Book Size");
rowhead.createCell((short) 12).setCellValue("No. Of Pages");
rowhead.createCell((short) 13).setCellValue("Plate new/old");
rowhead.createCell((short) 14).setCellValue("Plate Required");
rowhead.createCell((short) 15).setCellValue("Paper Supplied");
rowhead.createCell((short) 16).setCellValue("Paper in Sheets");
rowhead.createCell((short) 17).setCellValue("Finish Qt.");
rowhead.createCell((short) 18).setCellValue("Delivery Date");
rowhead.createCell((short) 19).setCellValue("Status");
rowhead.createCell((short) 20).setCellValue("Machine");
rowhead.createCell((short) 21).setCellValue("Team Leader");

st = con.createStatement();
ResultSet rs = st.executeQuery(query);
int i = 1;
while (rs.next()) {
HSSFRow row = sheet.createRow((short) i);
row.createCell((short) 0).setCellValue(Integer.toString(rs.getInt("Job_no")));
row.createCell((short) 1).setCellValue(rs.getString("Job_date"));
row.createCell((short) 2).setCellValue(rs.getString("Client_name"));
row.createCell((short) 3).setCellValue(rs.getString("Job_name"));
row.createCell((short) 4).setCellValue(rs.getString("No_of_forms"));
row.createCell((short) 5).setCellValue(rs.getString("Print_style"));
row.createCell((short) 6).setCellValue(rs.getString("Paper_used"));
row.createCell((short) 7).setCellValue(rs.getString("No_of_col"));
row.createCell((short) 8).setCellValue(rs.getString("Paper_qt"));
row.createCell((short) 9).setCellValue(rs.getString("Plate"));
row.createCell((short) 10).setCellValue(rs.getString("Paper_print"));
row.createCell((short) 11).setCellValue(rs.getString("Book_size"));
row.createCell((short) 12).setCellValue(rs.getString("No_of_pages"));
row.createCell((short) 13).setCellValue(rs.getString("Plate_new/old"));
row.createCell((short) 14).setCellValue(rs.getString("Paper_Required"));
row.createCell((short) 15).setCellValue(rs.getString("Paper_supp"));
row.createCell((short) 16).setCellValue(rs.getString("Paper_in_sheets"));
row.createCell((short) 17).setCellValue(rs.getString("Finish_quantity"));
```

```
row.createCell((short) 18).setCellValue(rs.getString("Delivery_date"));
row.createCell((short) 19).setCellValue(rs.getString("Status"));
row.createCell((short) 20).setCellValue(rs.getString("Machine"));
row.createCell((short) 21).setCellValue(rs.getString("Team_Leader"));
i++;
}
FileOutputStream fileOut = new FileOutputStream(filename);
hwb.write(fileOut);
fileOut.close();
JOptionPane.showMessageDialog(btnGenerateExcelFile, filename + ".xls file generated");
} catch (Exception ex) {
System.out.println(ex);

}

        }
```

Code for AddRecords1.java

```
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.sql.Connection;
import java.sql.Statement;
import java.text.SimpleDateFormat;
import java.util.Date;

import javax.swing.BorderFactory;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JComboBox;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;
import javax.swing.border.Border;

public class AddRecord_1 extends JPanel {

        private Connection conn;
        private Statement st;
        private JLabel labelJobNo;
        private JLabel labelDate;
        private JLabel labelPartyName;
        private JLabel labelJobName;
        private JLabel labelTeamName;

        JTextField txtJobName;
        JTextField txtJobNo;
        private JTextField txtDate;
        private JComboBox comboPartyName;
        private JComboBox comboLeaderName;

        public AddRecord_1(Connection conn, Statement st)
        {
                this.conn=conn;
                this.st=st;

                Dimension dim = getPreferredSize();
                dim.width=600;
                dim.height=90;
                setPreferredSize(dim);
                //getRootPane().setBackground(Color.WHITE);

                labelJobName=new JLabel("Job Name And Team Leader");
                labelJobName.setFont(new Font("Times New Roman", Font.BOLD, 14));
                txtJobName=new JTextField(10);
                labelJobNo = new JLabel("Job Number");
                labelJobNo.setFont(new Font("Times New Roman", Font.BOLD, 14));
```

```
        labelDate = new JLabel("Date");
        labelDate.setFont(new Font("Times New Roman", Font.BOLD, 14));
        labelPartyName = new JLabel("CLient Name");
        labelPartyName.setFont(new Font("Times New Roman", Font.BOLD, 14));
        txtJobNo = new JTextField(7);
        txtJobNo.setEditable(false);
        txtDate= new JTextField(7);
        txtDate.setEditable(false);
        comboPartyName=new JComboBox();
        comboLeaderName = new JComboBox();
        comboLeaderName.setFont(new Font("Helvetica", Font.ITALIC, 13));
        comboLeaderName.setEditable(false);
        comboPartyName.setFont(new Font("Helvetica", Font.ITALIC, 13));
        comboPartyName.setEditable(true);

        Border inner = BorderFactory.createTitledBorder("Primary Details");
        Border outer = BorderFactory.createEmptyBorder(3, 3, 3, 3);
        setBorder(BorderFactory.createCompoundBorder(outer, inner));

        Date date= new Date();
        SimpleDateFormat ft = new SimpleDateFormat ("yyyy.MM.dd");
        String fdate=ft.format(date);
        txtDate.setText(fdate);

        DefaultComboBoxModel comboModel=new DefaultComboBoxModel();
        comboModel.addElement("Kiran Prakashan");
        comboModel.addElement("Fab Files");
        comboModel.addElement("Deep Enterprises");
        comboModel.addElement("Manohar");
        comboModel.addElement("Gyan Ganga");
        comboPartyName.setModel(comboModel);
        comboPartyName.setEditable(true);


        DefaultComboBoxModel comboModel1=new DefaultComboBoxModel();
        comboModel1.addElement("Sharad");
        comboModel1.addElement("Dinesh");
        comboModel1.addElement("Ashutosh");
        comboModel1.addElement("Varun");
        comboModel1.addElement("Karan");
        comboLeaderName.setModel(comboModel1);
        comboLeaderName.setEditable(true);

        setLayout(new GridBagLayout());
        GridBagConstraints gc= new GridBagConstraints();
        gc.gridwidth=1;
        gc.gridx=0;
        gc.gridy=0;
        gc.weightx=0.01;
        gc.weighty=1;
        gc.fill=GridBagConstraints.EAST;
        gc.anchor=GridBagConstraints.FIRST_LINE_START;
        //gc.insets=new Insets(0,0,0,1);
        add(labelJobNo, gc);

        gc.gridx=1;
        gc.gridy=0;
        gc.weightx=0.05;
        gc.fill=GridBagConstraints.WEST;
//      gc.anchor=GridBagConstraints.FIRST_LINE_START;
        add(txtJobNo, gc);

        gc.gridx=3;
        gc.gridy=0;
        gc.weightx=0.01;
        gc.fill=GridBagConstraints.EAST;
        gc.anchor=GridBagConstraints.PAGE_START;
        //gc.insets=new Insets(0,0,0,1);
```

```
		add(labelDate, gc);

		gc.gridx=4;
		gc.gridy=0;
		gc.fill=GridBagConstraints.WEST;
		//gc.anchor=GridBagConstraints.PAGE_START;
		add(txtDate, gc);


		gc.gridx=6;
		gc.gridy=0;
		gc.weightx=0.05;
		gc.fill=GridBagConstraints.EAST;
		gc.anchor=GridBagConstraints.FIRST_LINE_END;
		//gc.insets=new Insets(0,0,0,1);
		add(labelPartyName, gc);

		gc.gridx=7;
		gc.gridy=0;
		gc.fill=GridBagConstraints.WEST;
		gc.anchor=GridBagConstraints.FIRST_LINE_START;
		add(comboPartyName, gc);

/*
		gc.gridx=2;
		gc.gridy=1;
		gc.weightx=0.05;
		gc.fill=GridBagConstraints.EAST;
		gc.anchor=GridBagConstraints.FIRST_LINE_END;
		//gc.insets=new Insets(0,0,0,1);
		add(labelTeamName, gc);*/

		gc.gridx=3;
		gc.gridy=1;
		gc.fill=GridBagConstraints.WEST;
		gc.anchor=GridBagConstraints.FIRST_LINE_START;
		add(comboLeaderName, gc);

		gc.gridx=0;
		gc.gridy=1;
		gc.weightx=0.01;
		gc.weighty=1;
		gc.fill=GridBagConstraints.EAST;
		gc.anchor=GridBagConstraints.FIRST_LINE_START;
		//gc.insets=new Insets(0,0,0,1);
		add(labelJobName, gc);

		gc.gridx=1;
		gc.gridy=1;
		gc.weightx=0.05;
		gc.fill=GridBagConstraints.WEST;
//		gc.anchor=GridBagConstraints.FIRST_LINE_START;
		add(txtJobName, gc);
		//setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

	}
	public String getJobName()
	{
		String job_no=txtJobName.getText();
		return job_no;
	}

	public String getLeaderName()
	{
		String leader_name=(String)comboLeaderName.getSelectedItem();
		return leader_name;
	}
```

```
	public String getJobNo()
	{
		String job_no=txtJobNo.getText();
		return job_no;
	}
	public String getDate()
	{
		String date=txtDate.getText();
		return date;
	}
	public String getClient_name()
	{
		String client_name=(String)comboPartyName.getSelectedItem();
		return client_name;
	}


}
```

Code for AddRedord.java

```
import java.awt.Color;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JOptionPane;

import org.apache.poi.hssf.usermodel.HSSFRow;
import org.apache.poi.hssf.usermodel.HSSFSheet;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;

public class AddRecord extends JFrame {

	private AddRecord_1 record1;
	private AddRecord_2 record2;
	private AddRecord_3 record3;
	private JButton btnPun;
	private JButton btnBack;
	private Connection conn;
	private PreparedStatement st;
	String job_no;
	String job_date;
	String client;
	String leader;
	String job_name;
	String forms;
	String style;
	// String style;
	String pUsed;
	String col;
	String printqt;
	String book;
	String pages;
	String pl;
	String plate;
```

```
String pPrint;
String paperReq;
String paperSup;
String sheets;
String finish;
String Ddate;
String sta;
String mach;
boolean nullVal;
String area;
String are;
String finAr;
int jNo;
int c;
private JButton btnFile;

public AddRecord(Connection conn, PreparedStatement st, int countJobN) {
	setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
	this.conn = conn;
	this.st = st;
	record1 = new AddRecord_1(conn, st);
	record2 = new AddRecord_2(conn, st);
	record3 = new AddRecord_3(conn, st);
	btnPun = new JButton("PUNCH RECORD");
	btnPun.setSize(20, 20);
	// getContentPane().setBackground(Color.WHITE);
	this.c = countJobN;
	jNo = c;
	jNo++;
	String jno = Integer.toString(jNo);
	record1.txtJobNo.setText(jno);
	btnPun.addActionListener(new ActionListener() {
		PreparedStatement state;

		public void actionPerformed(ActionEvent e) {

			boolean val = fieldCheck(); // field check is a function to
										// check if all the fields are
										// filled

			if (val == false) {
				JOptionPane.showMessageDialog(btnPun, "Query not Added");
			} else {
				PreparedStatement state;
				int countJobN = 0;
				String qu1 = "Select max(Job_no) from SisClient";
				try {
					state = conn.prepareStatement(qu1);
					ResultSet rst = state.executeQuery();
					while (rst.next()) {
						countJobN = rst.getInt(1);
						System.out.println(countJobN);
						countJobN = countJobN + 1;
					}
				} catch (SQLException e2) {
					e2.printStackTrace();
				}

				String jno = Integer.toString(countJobN);
				record1.txtJobNo.setText(jno);
				// Method used to insert query into DB
				punchMethod(conn, st);
				record1.txtJobName.setText("");
				record2.txtNo_of_forms.setText("");
				record2.txtCol.setText("");
				record2.txtPrintQt.setText("");
				record2.txtPaper_Print.setText("");
				record2.txtBookSize.setText("");
```

```
                                record2.txtNo_of_pages.setText("");
                                record2.plateNEW.setSelected(false);
                                record2.plateOLD.setSelected(false);
                                record2.txtPaperReq.setText("");
                                record2.txtPapersupp.setText("");
                                record2.txtPaperSheets.setText("");
                                record2.txtfinishQt.setText("");
                                record2.txtdeliverydate.setText("");
                                record3.reset(record3.btnA, record3.btnB);
                                record3.reset(record3.btnC, record3.btnD);

                                int countJobN1 = 0;
                                // Query to get the last Job No
                                String qu11 = "Select max(Job_no) from SisClient";
                                try {
                                        state = conn.prepareStatement(qu11);
                                        ResultSet rst = state.executeQuery();
                                        while (rst.next()) {
                                                countJobN1 = rst.getInt(1);
                                                System.out.println(countJobN1);
                                                countJobN1 = countJobN1 + 1;
                                        }

                                } catch (SQLException e2) {
                                        // TODO Auto-generated catch block
                                        e2.printStackTrace();
                                }

                                String jno1 = Integer.toString(countJobN1);
                                record1.txtJobNo.setText(jno1);

                        }

                }

});

btnFile = new JButton("Generate Excel File");
btnFile.addActionListener(new ActionListener() {
        Statement st1;

        public void actionPerformed(ActionEvent e) {
                try {
                        generateXls_4(conn, st1, "Records");
                } catch (SQLException | IOException e1) {
                        // TODO Auto-generated catch block
                        e1.printStackTrace();
                }
        }

});

btnBack = new JButton("Back");
btnBack.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
                dispose();
                new StaffForm(conn, st);
        }

});

setLayout(new GridBagLayout());
GridBagConstraints gc = new GridBagConstraints();
gc.gridwidth = 4;
gc.gridx = 0;
gc.gridy = 0;
gc.weightx = 0.01;
gc.weighty = 1;
```

```
        gc.fill = GridBagConstraints.HORIZONTAL;
        gc.anchor = GridBagConstraints.FIRST_LINE_START;
        // gc.insets=new Insets(0,0,0,1);
        add(record1, gc);

        gc.gridx = 0;
        gc.gridy = 1;
        gc.gridwidth = 4;
        gc.weightx = 0.01;
        gc.ipady = 40;
        gc.fill = GridBagConstraints.HORIZONTAL;
        gc.anchor = GridBagConstraints.PAGE_START;
        // gc.insets=new Insets(0,0,0,1);
        add(record2, gc);

        gc.gridx = 0;
        gc.gridy = 2;
        gc.weightx = 0;
        gc.weighty = 1;
        gc.gridwidth = 1;
        gc.gridheight = 4;
        gc.ipady = 0;
        gc.fill = GridBagConstraints.PAGE_END;
        gc.anchor = GridBagConstraints.CENTER;
        gc.insets = new Insets(0, 0, 0, 1);
        add(record3, gc);

        gc.gridx = 1;
        gc.gridy = 2;
        gc.weightx = 0;
        gc.weighty = 0.1;
        gc.gridwidth = 1;
        gc.gridheight = 1;
        gc.fill = GridBagConstraints.EAST;
        gc.anchor = GridBagConstraints.CENTER;
        // gc.insets=new Insets(0,0,0,1);
        add(btnPun, gc);

        gc.gridx = 2;
        gc.gridy = 2;
        gc.weightx = 0;
        gc.weighty = 0.1;
        gc.gridwidth = GridBagConstraints.RELATIVE;
        gc.gridheight = 1;
        gc.fill = GridBagConstraints.WEST;
        gc.anchor = GridBagConstraints.WEST;
        // gc.insets=new Insets(0,0,0,1);
        add(btnBack, gc);

        gc.gridx = 3;
        gc.gridy = 2;
        gc.weightx = 0;
        gc.weighty = 0.1;
        gc.gridwidth = GridBagConstraints.REMAINDER;
        gc.gridheight = 1;
        gc.fill = GridBagConstraints.WEST;
        gc.anchor = GridBagConstraints.WEST;
        // gc.insets=new Insets(0,0,0,1);
        add(btnFile, gc);

        // add(record1, BoxLayout.PAGE_AXIS);
        // add(record2, BorderLayout.CENTER);
        // add(record3, BorderLayout.SOUTH);
        // add(btnPun, BoxLayout.X_AXIS);
        setSize(750, 650);
        setVisible(true);
}
```

```
	// Method ensures that all field values have been filled by the user.
	public boolean fieldCheck() {
		outer: {
			if (record1.getJobNo().isEmpty()) {
				nullVal = false;
				JOptionPane.showMessageDialog(btnPun, "Field Empty." + "Add Job No. Other Fields may/may not be
empty");
				break outer;
			} else {
				nullVal = true;
				job_no = record1.getJobNo();
			}
			if (record1.getDate().isEmpty()) {
				nullVal = false;
				JOptionPane.showMessageDialog(btnPun, "Field Empty. Add Job Date. Other Fields may/may not be
empty");
				break outer;
			} else {
				nullVal = true;
				job_date = record1.getDate();
			}
			if (record1.getClient_name().isEmpty()) {
				nullVal = false;
				JOptionPane.showMessageDialog(btnPun,
						"Field Empty. Add Client Name. Other Fields may/may not be empty");
				break outer;
			} else {
				nullVal = true;
				client = record1.getClient_name();
			}
			if (record1.getLeaderName().isEmpty()) {
				nullVal = false;
				JOptionPane.showMessageDialog(btnPun,
						"Field Empty. Add Client Name. Other Fields may/may not be empty");
				break outer;
			} else {
				nullVal = true;
				leader = record1.getLeaderName();
			}
			if (record1.getJobName().isEmpty()) {
				nullVal = false;
				JOptionPane.showMessageDialog(btnPun, "Field Empty. Add Job Name. Other Fields may/may not be
empty");
				break outer;
			} else {
				nullVal = true;
				job_name = record1.getJobName();
			}
			if (record2.no_ofForms().isEmpty()) {
				nullVal = false;
				JOptionPane.showMessageDialog(btnPun,
						"Field Empty. Add No of Forms. Other Fields may/may not be empty");
				break outer;
			} else {
				nullVal = true;
				forms = record2.no_ofForms();
			}
			if (record2.ComboPrintStyle().isEmpty()) {
				nullVal = false;
				JOptionPane.showMessageDialog(btnPun,
						"Field Empty. Add Print Style. Other Fields may/may not be empty");
				break outer;
			} else {
				nullVal = true;
				style = record2.ComboPrintStyle();
			}
			if (record2.CombopaperUsed().isEmpty()) {
				nullVal = false;
```

```
                    JOptionPane.showMessageDialog(btnPun, "Field Empty. Add Paper Used. Other Fields may/may not be
empty");
                    break outer;
                } else {
                    nullVal = true;
                    pUsed = record2.CombopaperUsed();
                }
                if (record2.TxtCol().isEmpty()) {
                    nullVal = false;
                    JOptionPane.showMessageDialog(btnPun,
                            "Field Empty. Add Number of Colors. Other Fields may/may not be empty");
                    break outer;
                } else {
                    nullVal = true;
                    col = record2.TxtCol();
                }
                if (record2.TxtPrintQt().isEmpty()) {
                    nullVal = false;
                    JOptionPane.showMessageDialog(btnPun,
                            "Field Empty. Add Paper Quantity. Other Fields may/may not be empty");
                    break outer;
                } else {
                    nullVal = true;
                    printqt = record2.TxtPrintQt();
                    int printqtity = Integer.parseInt(printqt);
                }
                if (record2.TxtBookSize().isEmpty()) {
                    nullVal = false;
                    JOptionPane.showMessageDialog(btnPun, "Field Empty. Add Book Size Other. Fields may/may not be
empty");
                    break outer;
                } else {
                    nullVal = true;
                    book = record2.TxtBookSize();
                }
                if (record2.TxtNo_of_pages().isEmpty()) {
                    nullVal = false;
                    JOptionPane.showMessageDialog(btnPun,
                            "Field Empty. Add Number of Pages. Other Fields may/may not be empty");
                    break outer;
                } else {
                    nullVal = true;
                    pages = record2.TxtNo_of_pages();
                }
                if (record2.PlateOld().isEmpty()) {
                    nullVal = false;
                    JOptionPane.showMessageDialog(btnPun,
                            "Field Empty. Choose Plate (Old/New). Other Fields may/may not be empty");
                    break outer;
                } else {
                    nullVal = true;
                    pl = record2.PlateOld();
                }
                if (record2.getPlate().isEmpty()) {
                    nullVal = false;
                    JOptionPane.showMessageDialog(btnPun,
                            "Field Empty. Add values into Plate Field. Other Fields may/may not be empty");
                    break outer;
                } else {
                    nullVal = true;
                    plate = record2.getPlate();
                }
                if (record2.TxtPaper_Print().isEmpty()) {
                    nullVal = false;
                    JOptionPane.showMessageDialog(btnPun,
                            "Field Empty. Add Paper Print. Other Fields may/may not be empty");
                    break outer;
                } else {
```

```
                                nullVal = true;
                                pPrint = record2.TxtPaper_Print();
                        }
                        if (record2.TxtPaperReq().isEmpty()) {
                                nullVal = false;
                                JOptionPane.showMessageDialog(btnPun,
                                                "Field Empty. Add Paper Required. Other Fields may/may not be empty");
                                break outer;
                        } else {
                                nullVal = true;
                                paperReq = record2.TxtPaperReq();
                        }
                        if (record2.TxtPapersupp().isEmpty()) {
                                nullVal = false;
                                JOptionPane.showMessageDialog(btnPun, "Field Empty. Add Paper Used. Other Fields may/may not be
empty");
                                break outer;
                        } else {
                                nullVal = true;
                                paperSup = record2.TxtPapersupp();
                        }
                        if (record2.Psheets().isEmpty()) {
                                nullVal = false;
                                JOptionPane.showMessageDialog(btnPun,
                                                "Field Empty. Add No of Sheets. Other Fields may/may not be empty");
                                break outer;
                        } else {
                                nullVal = true;
                                sheets = record2.Psheets();
                        }
                        if (record2.ComboStatus().isEmpty()) {
                                nullVal = false;
                                JOptionPane.showMessageDialog(btnPun, "Field Empty. Add Status. Other Fields may/may not be
empty");
                                break outer;
                        } else {
                                nullVal = true;
                                sta = record2.ComboStatus();
                        }
                        if (record2.ComboMachine().isEmpty()) {
                                nullVal = false;
                                JOptionPane.showMessageDialog(btnPun, "Field Empty. Add Machine. Other Fields may/may not be
empty");
                                break outer;
                        } else {
                                nullVal = true;
                                mach = record2.ComboMachine();
                        }
                        if (record2.Txtdeliverydate().isEmpty()) {
                                nullVal = false;
                                JOptionPane.showMessageDialog(btnPun,
                                                "Field Empty. Add Delivery Date. Other Fields may/may not be empty");
                                break outer;
                        } else {
                                nullVal = true;
                                Ddate = record2.Txtdeliverydate();
                        }
                        if (record2.TxtfinishQt().isEmpty()) {
                                nullVal = false;
                                JOptionPane.showMessageDialog(btnPun,
                                                "Field Empty. Add Finish Quantity. Other Fields may/may not be empty");
                                break outer;
                        } else {
                                nullVal = true;
                                finish = record2.TxtfinishQt();
                        }
                }
                return nullVal;
```

```
}

public void punchMethod(Connection conn, PreparedStatement state) {
		String job_no = record1.getJobNo();
		String job_date = record1.getDate();
		String client = record1.getClient_name();
		String leader = record1.getLeaderName();
		String job_name = record1.getJobName();
		String forms = record2.no_ofForms();
		String style = record2.ComboPrintStyle();
		String pUsed = record2.CombopaperUsed();
		String col = record2.TxtCol();
		String printqt = record2.TxtPrintQt();
		int printqtity = Integer.parseInt(printqt);
		String book = record2.TxtBookSize();
		String pages = record2.TxtNo_of_pages();
		String pl = record2.PlateOld();
		String plate = record2.getPlate();
		String pPrint = record2.TxtPaper_Print();
		String paperReq = record2.TxtPaperReq();
		String paperSup = record2.TxtPapersupp();
		String sheets = record2.Psheets();
		String finish = record2.TxtfinishQt();
		String Ddate = record2.Txtdeliverydate();
		String sta = record2.ComboStatus();
		String mach = record2.ComboMachine();
		area = record3.getA1();
		are = record3.getA2();
		finAr = area + " " + are;

		String qu1 = "insert into SisClient values (" + "'" + job_no + "'" + "," + "'" + job_date + "'" + "," + "'"
				+ client + "'" + "," + "'" + job_name + "'" + "," + "'" + forms + "'" + "," + "'" + style + "'" + ","
				+ "'" + pUsed + "'" + "," + "'" + col + "'" + "," + printqtity + "," + "'" + plate + "'" + "," + "'"
				+ pPrint + "'" + "," + "'" + book + "'" + "," + "'" + pages + "'" + "," + "'" + pl + "'" + "," + "'"
				+ paperReq + "'" + "," + "'" + paperSup + "'" + "," + "'" + sheets + "'" + "," + "'" + finish + "'"
				+ "," + "'" + Ddate + "'" + "," + "'" + sta + "'" + "," + "'" + mach + "'" + "," + "'" + finAr + "'"
				+ "," + "'" + leader + "'" + ")";

		Statement state1;
		try {
			state1 = conn.createStatement();
			state1.executeUpdate(qu1);
			JOptionPane.showMessageDialog(btnPun, "RECORD ADDED SUCCESFULLY");
		} catch (SQLException e1) {
			// TODO Auto-generated catch block
			e1.printStackTrace();
		}

}

public void generateXls_4(Connection con, Statement st, String filename)
			throws SQLException, FileNotFoundException, IOException {
		try {
			String filename1 = "Records.xls";
			HSSFWorkbook hwb = new HSSFWorkbook();
			HSSFSheet sheet = hwb.createSheet("Staff sheet");

			HSSFRow rowhead = sheet.createRow((short) 0);
			rowhead.createCell((short) 0).setCellValue("Job No.");
			rowhead.createCell((short) 1).setCellValue("Job Date");
			rowhead.createCell((short) 2).setCellValue("Client Name");
			rowhead.createCell((short) 3).setCellValue("Job Name");
			rowhead.createCell((short) 4).setCellValue("No. Of Forms ");
			rowhead.createCell((short) 5).setCellValue("Print Style");
			rowhead.createCell((short) 6).setCellValue("Paper Used");
			rowhead.createCell((short) 7).setCellValue("No. of Colors");
			rowhead.createCell((short) 8).setCellValue("Paper Qt.");
			rowhead.createCell((short) 9).setCellValue("Plate");
```

```
                        rowhead.createCell((short) 10).setCellValue("Paper Print");
                        rowhead.createCell((short) 11).setCellValue("Book Size");
                        rowhead.createCell((short) 12).setCellValue("No. Of Pages");
                        rowhead.createCell((short) 13).setCellValue("Plate new/old");
                        rowhead.createCell((short) 14).setCellValue("Plate Required");
                        rowhead.createCell((short) 15).setCellValue("Paper Supplied");
                        rowhead.createCell((short) 16).setCellValue("Paper in Sheets");
                        rowhead.createCell((short) 17).setCellValue("Finish Qt.");
                        rowhead.createCell((short) 18).setCellValue("Delivery Date");
                        rowhead.createCell((short) 19).setCellValue("Status");
                        rowhead.createCell((short) 20).setCellValue("Machine");
                        rowhead.createCell((short) 21).setCellValue("Team Leader");
                        // Team_Leader

                        // Class.forName("com.mysql.jdbc.Driver");
                        // Connection con =
                        // DriverManager.getConnection("jdbc:mysql://localhost:3306/test",
                        // "root", "root");
                        st = con.createStatement();
                        ResultSet rs = st.executeQuery("Select * from SisClient");
                        int i = 1;
                        while (rs.next()) {
                                HSSFRow row = sheet.createRow((short) i);
                                row.createCell((short) 0).setCellValue(Integer.toString(rs.getInt("Job_no")));
                                row.createCell((short) 1).setCellValue(rs.getString("Job_date"));
                                row.createCell((short) 2).setCellValue(rs.getString("Client_name"));
                                row.createCell((short) 3).setCellValue(rs.getString("Job_name"));
                                row.createCell((short) 4).setCellValue(rs.getString("No_of_forms"));
                                row.createCell((short) 5).setCellValue(rs.getString("Print_style"));
                                row.createCell((short) 6).setCellValue(rs.getString("Paper_used"));
                                row.createCell((short) 7).setCellValue(rs.getString("No_of_col"));
                                row.createCell((short) 8).setCellValue(rs.getString("Paper_qt"));
                                row.createCell((short) 9).setCellValue(rs.getString("Plate"));
                                row.createCell((short) 10).setCellValue(rs.getString("Paper_print"));
                                row.createCell((short) 11).setCellValue(rs.getString("Book_size"));
                                row.createCell((short) 12).setCellValue(rs.getString("No_of_pages"));
                                row.createCell((short) 13).setCellValue(rs.getString("Plate_new/old"));
                                row.createCell((short) 14).setCellValue(rs.getString("Paper_Required"));
                                row.createCell((short) 15).setCellValue(rs.getString("Paper_supp"));
                                row.createCell((short) 16).setCellValue(rs.getString("Paper_in_sheets"));
                                row.createCell((short) 17).setCellValue(rs.getString("Finish_quantity"));
                                row.createCell((short) 18).setCellValue(rs.getString("Delivery_date"));
                                row.createCell((short) 19).setCellValue(rs.getString("Status"));
                                row.createCell((short) 20).setCellValue(rs.getString("Machine"));
                                row.createCell((short) 21).setCellValue(rs.getString("Team_Leader"));
                                // Team_Leader
                                i++;
                        }
                        FileOutputStream fileOut = new FileOutputStream(filename1);
                        hwb.write(fileOut);
                        fileOut.close();
                        JOptionPane.showMessageDialog(btnFile, filename1 + "file generated");

                } catch (Exception ex) {
                        System.out.println(ex);

                }

        }
}
```

Code For AdminForm.java

```
import java.awt.Color;
import java.awt.Font;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
```

```
import java.awt.Insets;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.PriorityQueue;
import java.util.Stack;

import javax.swing.BorderFactory;
import javax.swing.ButtonGroup;
import javax.swing.DefaultComboBoxModel;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JSeparator;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.UIManager;
import javax.swing.border.Border;
import javax.swing.border.EmptyBorder;

import org.apache.poi.hssf.usermodel.HSSFRow;
import org.apache.poi.hssf.usermodel.HSSFSheet;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;
import java.awt.event.MouseMotionAdapter;
import java.awt.event.MouseEvent;

public class AdminForm extends JFrame {

	private JPanel contentPane;
	private JTextField textField;
	private JTextField txtDdmmyy;
	private JTextField txtDdmmyy_1;
	private JTextField textField_2;
	private JLabel lblClientName;
	private JLabel lblNewLabel;
	private JLabel lblDeliveryDate;
	private JLabel lblMachine;
	private JLabel lblStatus;
	JButton btnGenerateExcelFile;
	private Connection conn;
```

```
private Statement st;
private JSeparator separator_2;
private JTextField txtSpecifyFileName;
private JLabel lblExcelFileName;
private JLabel lblNewLabel_1;
private JComboBox comboBox;
private JRadioButton rdbtnTeamAndClient;
private JButton btnNewButton;
private JButton btnNewButton_1;
private Stack<String> stac;
private String lastStringVal;
private static final String FileName = "/Users/DGair/Desktop/New Java Programs /SoftwareSIS/Comments.txt";
private JTextArea txtrJhb;

/**
 * Create the frame.
 *
 * @param st
 * @param conn
 */

public String[] parse() {
	ArrayList<String> list1 = new ArrayList<String>();

	try (BufferedReader br = new BufferedReader(new FileReader(FileName))) {

		String sCurrentLine;

		while ((sCurrentLine = br.readLine()) != null) {
			list1.add(sCurrentLine);
		}

	} catch (IOException e) {
		e.printStackTrace();
	}

	String[] ar = new String[list1.size()];
	list1.toArray(ar);

	return ar;

}

public AdminForm(Connection conn, Statement st) {
	this.conn = conn;
	this.st = st;
	stac = new Stack<String>();
	// textField_1.setText(lastStringVal);
	setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
	setBounds(250, 250, 488, 541);
	contentPane = new JPanel();
	contentPane.setBackground(Color.WHITE);
	contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
	setContentPane(contentPane);
	Border innerborder = BorderFactory.createTitledBorder("Admin Query");
	contentPane.setBorder(innerborder);
	GridBagLayout gbl_contentPane = new GridBagLayout();
	gbl_contentPane.columnWidths = new int[] { 0, 0, 0, 0, 0, 0 };
```

```
		gbl_contentPane.rowHeights = new int[] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
		gbl_contentPane.columnWeights = new double[] { 0.0, 1.0, 1.0, 1.0, 0.0, Double.MIN_VALUE };
		gbl_contentPane.rowWeights = new double[] { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0,
				0.0, 0.0, 0.0, Double.MIN_VALUE };
		contentPane.setLayout(gbl_contentPane);

		JRadioButton rdbtnQueryResultsBased = new JRadioButton("Query Data");
		rdbtnQueryResultsBased.setFont(new Font("Times", Font.BOLD | Font.ITALIC, 14));
		rdbtnQueryResultsBased.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent e) {
				if (rdbtnQueryResultsBased.isSelected()) {
					// query data from database using one of the filters
					textField.setEnabled(true);
					txtDdmmyy_1.setEnabled(true);
					txtDdmmyy.setEnabled(true);
					comboBox.setEnabled(true);
					textField_2.setEnabled(true);
					lblClientName.setEnabled(true);
					lblNewLabel.setEnabled(true);
					lblDeliveryDate.setEnabled(true);
					lblMachine.setEnabled(true);
					lblStatus.setEnabled(true);
					btnGenerateExcelFile.setEnabled(true);
					lblExcelFileName.setEnabled(true);
					txtSpecifyFileName.setEnabled(true);
					JOptionPane.showMessageDialog(rdbtnQueryResultsBased, "Query Table Using
Single Filter Only");
				}
			}
		});

		GridBagConstraints gbc_rdbtnQueryResultsBased = new GridBagConstraints();
		gbc_rdbtnQueryResultsBased.anchor = GridBagConstraints.WEST;
		gbc_rdbtnQueryResultsBased.insets = new Insets(0, 0, 5, 5);
		gbc_rdbtnQueryResultsBased.gridx = 1;
		gbc_rdbtnQueryResultsBased.gridy = 0;
		contentPane.add(rdbtnQueryResultsBased, gbc_rdbtnQueryResultsBased);

		lblNewLabel_1 = new JLabel("");
		lblNewLabel_1.setIcon(new ImageIcon("/Users/DGair/Desktop/New Java Programs
/SoftwareSIS/pic.png"));
		GridBagConstraints gbc_lblNewLabel_1 = new GridBagConstraints();
		gbc_lblNewLabel_1.gridheight = 2;
		gbc_lblNewLabel_1.insets = new Insets(0, 0, 5, 5);
		gbc_lblNewLabel_1.gridx = 2;
		gbc_lblNewLabel_1.gridy = 0;
		contentPane.add(lblNewLabel_1, gbc_lblNewLabel_1);

		JRadioButton rdbtnGenerateExcelFile = new JRadioButton("Generate Excel File");
		rdbtnGenerateExcelFile.setToolTipText("CLICK TO CREATE EXCEL FILE FOR ALL ENTRIES");
		rdbtnGenerateExcelFile.setFont(new Font("Times", Font.BOLD | Font.ITALIC, 14));
		// rdbtnQueryResultsBased.setEnabled(false);
		GridBagConstraints gbc_rdbtnGenerateExcelFile = new GridBagConstraints();
		gbc_rdbtnGenerateExcelFile.anchor = GridBagConstraints.WEST;
		gbc_rdbtnGenerateExcelFile.insets = new Insets(0, 0, 5, 5);
		gbc_rdbtnGenerateExcelFile.gridx = 1;
		gbc_rdbtnGenerateExcelFile.gridy = 1;
		contentPane.add(rdbtnGenerateExcelFile, gbc_rdbtnGenerateExcelFile);
```

```
rdbtnTeamAndClient = new JRadioButton("Team And Client Relationship");
rdbtnTeamAndClient.addActionListener(new ActionListener() {
	public void actionPerformed(ActionEvent e) {

		dispose();
		try {
			new TeamAndClient(conn, st);
		} catch (SQLException e1) {
			// TODO Auto-generated catch block
			e1.printStackTrace();
		}
	}
});
rdbtnTeamAndClient.setFont(new Font("Times", Font.BOLD | Font.ITALIC, 14));
GridBagConstraints gbc_rdbtnTeamAndClient = new GridBagConstraints();
gbc_rdbtnTeamAndClient.anchor = GridBagConstraints.WEST;
gbc_rdbtnTeamAndClient.insets = new Insets(0, 0, 5, 5);
gbc_rdbtnTeamAndClient.gridx = 1;
gbc_rdbtnTeamAndClient.gridy = 2;
contentPane.add(rdbtnTeamAndClient, gbc_rdbtnTeamAndClient);



lblClientName = new JLabel("Client Name");
lblClientName.setFont(new Font("Times New Roman", Font.BOLD, 14));
lblClientName.setEnabled(false);
GridBagConstraints gbc_lblClientName = new GridBagConstraints();
gbc_lblClientName.insets = new Insets(0, 0, 5, 5);
gbc_lblClientName.anchor = GridBagConstraints.EAST;
gbc_lblClientName.gridx = 1;
gbc_lblClientName.gridy = 4;
contentPane.add(lblClientName, gbc_lblClientName);

ButtonGroup bg = new ButtonGroup();
bg.add(rdbtnGenerateExcelFile);
bg.add(rdbtnQueryResultsBased);
bg.add(rdbtnTeamAndClient);

textField = new JTextField();
textField.setEnabled(false);
GridBagConstraints gbc_textField = new GridBagConstraints();
gbc_textField.insets = new Insets(0, 0, 5, 5);
gbc_textField.anchor = GridBagConstraints.WEST;
gbc_textField.gridx = 2;
gbc_textField.gridy = 4;
contentPane.add(textField, gbc_textField);
textField.setColumns(10);

lblNewLabel = new JLabel("Date Order Recieved");
lblNewLabel.setFont(new Font("Times New Roman", Font.BOLD, 14));
lblNewLabel.setEnabled(false);
GridBagConstraints gbc_lblNewLabel = new GridBagConstraints();
gbc_lblNewLabel.anchor = GridBagConstraints.EAST;
gbc_lblNewLabel.insets = new Insets(0, 0, 5, 5);
gbc_lblNewLabel.gridx = 1;
gbc_lblNewLabel.gridy = 5;
```

```
contentPane.add(lblNewLabel, gbc_lblNewLabel);

txtDdmmyy = new JTextField();
txtDdmmyy.setEnabled(false);
GridBagConstraints gbc_txtDdmmyy = new GridBagConstraints();
gbc_txtDdmmyy.insets = new Insets(0, 0, 5, 5);
gbc_txtDdmmyy.anchor = GridBagConstraints.WEST;
gbc_txtDdmmyy.gridx = 2;
gbc_txtDdmmyy.gridy = 5;
contentPane.add(txtDdmmyy, gbc_txtDdmmyy);
txtDdmmyy.setColumns(10);

lblDeliveryDate = new JLabel("Delivery Date");
lblDeliveryDate.setFont(new Font("Times New Roman", Font.BOLD, 14));
lblDeliveryDate.setEnabled(false);
GridBagConstraints gbc_lblDeliveryDate = new GridBagConstraints();
gbc_lblDeliveryDate.anchor = GridBagConstraints.EAST;
gbc_lblDeliveryDate.insets = new Insets(0, 0, 5, 5);
gbc_lblDeliveryDate.gridx = 1;
gbc_lblDeliveryDate.gridy = 6;
contentPane.add(lblDeliveryDate, gbc_lblDeliveryDate);

txtDdmmyy_1 = new JTextField();
txtDdmmyy_1.setEnabled(false);
GridBagConstraints gbc_txtDdmmyy_1 = new GridBagConstraints();
gbc_txtDdmmyy_1.insets = new Insets(0, 0, 5, 5);
gbc_txtDdmmyy_1.anchor = GridBagConstraints.WEST;
gbc_txtDdmmyy_1.gridx = 2;
gbc_txtDdmmyy_1.gridy = 6;
contentPane.add(txtDdmmyy_1, gbc_txtDdmmyy_1);
txtDdmmyy_1.setColumns(10);

lblMachine = new JLabel("Machine");
lblMachine.setFont(new Font("Times New Roman", Font.BOLD, 14));
lblMachine.setEnabled(false);
GridBagConstraints gbc_lblMachine = new GridBagConstraints();
gbc_lblMachine.anchor = GridBagConstraints.EAST;
gbc_lblMachine.insets = new Insets(0, 0, 5, 5);
gbc_lblMachine.gridx = 1;
gbc_lblMachine.gridy = 7;
contentPane.add(lblMachine, gbc_lblMachine);

comboBox = new JComboBox();
comboBox.setFont(new Font("Helvetica", Font.ITALIC, 13));
comboBox.setMaximumRowCount(6);
GridBagConstraints gbc_comboBox = new GridBagConstraints();
gbc_comboBox.fill = GridBagConstraints.BOTH;
gbc_comboBox.insets = new Insets(0, 0, 5, 5);
gbc_comboBox.gridx = 2;
gbc_comboBox.gridy = 7;
contentPane.add(comboBox, gbc_comboBox);
DefaultComboBoxModel comboModel5 = new DefaultComboBoxModel();
comboModel5.addElement("");
comboModel5.addElement("H1");
comboModel5.addElement("H2");
comboModel5.addElement("H4");
comboModel5.addElement("ZP");
comboModel5.addElement("Single Color 28 x 40");
```

```
comboModel5.addElement("2 Color 25 x 36");
comboModel5.addElement("WEB-1");
comboModel5.addElement("WEB-2");
comboModel5.addElement("WEB-3");
comboModel5.addElement("PressLine");
comboBox.setModel(comboModel5);
comboBox.setEditable(true);
comboBox.setEnabled(false);

lblStatus = new JLabel("Status");
lblStatus.setFont(new Font("Times New Roman", Font.BOLD, 14));
lblStatus.setEnabled(false);
GridBagConstraints gbc_lblStatus = new GridBagConstraints();
gbc_lblStatus.anchor = GridBagConstraints.EAST;
gbc_lblStatus.insets = new Insets(0, 0, 5, 5);
gbc_lblStatus.gridx = 1;
gbc_lblStatus.gridy = 8;
contentPane.add(lblStatus, gbc_lblStatus);

textField_2 = new JTextField();
textField_2.setEnabled(false);
GridBagConstraints gbc_textField_2 = new GridBagConstraints();
gbc_textField_2.insets = new Insets(0, 0, 5, 5);
gbc_textField_2.anchor = GridBagConstraints.WEST;
gbc_textField_2.gridx = 2;
gbc_textField_2.gridy = 8;
contentPane.add(textField_2, gbc_textField_2);
textField_2.setColumns(10);

lblExcelFileName = new JLabel("Excel File Name");
lblExcelFileName.setFont(new Font("Times New Roman", Font.BOLD, 14));
lblExcelFileName.setEnabled(false);
GridBagConstraints gbc_lblExcelFileName = new GridBagConstraints();
gbc_lblExcelFileName.insets = new Insets(0, 0, 5, 5);
gbc_lblExcelFileName.anchor = GridBagConstraints.EAST;
gbc_lblExcelFileName.gridx = 1;
gbc_lblExcelFileName.gridy = 9;
contentPane.add(lblExcelFileName, gbc_lblExcelFileName);

txtSpecifyFileName = new JTextField();
txtSpecifyFileName.setEnabled(false);
GridBagConstraints gbc_txtSpecifyFileName = new GridBagConstraints();
gbc_txtSpecifyFileName.anchor = GridBagConstraints.WEST;
gbc_txtSpecifyFileName.insets = new Insets(0, 0, 5, 5);
gbc_txtSpecifyFileName.gridx = 2;
gbc_txtSpecifyFileName.gridy = 9;
contentPane.add(txtSpecifyFileName, gbc_txtSpecifyFileName);
txtSpecifyFileName.setColumns(10);

btnGenerateExcelFile = new JButton("Generate Excel File ");
btnGenerateExcelFile.setFont(new Font("Times", Font.BOLD | Font.ITALIC, 14));
btnGenerateExcelFile.setEnabled(false);
GridBagConstraints gbc_btnGenerateExcelFile = new GridBagConstraints();
gbc_btnGenerateExcelFile.insets = new Insets(0, 0, 5, 5);
gbc_btnGenerateExcelFile.gridx = 2;
gbc_btnGenerateExcelFile.gridy = 10;
contentPane.add(btnGenerateExcelFile, gbc_btnGenerateExcelFile);
btnGenerateExcelFile.addActionListener(new ActionListener() {
```

```
                public void actionPerformed(ActionEvent e) {

                        // generating excel file using name
                        if (txtDdmmyy.getText().equals("") && txtDdmmyy_1.getText().equals("")
                                && comboBox.getSelectedItem().equals("") &&
textField_2.getText().equals("")) {
                                try {
                                        String filename_1 = txtSpecifyFileName.getText();
                                        String filename = filename_1;
                                        String name = textField.getText();
                                        String query = "Select * from SisClient where Client_name=" + "'" +
name + "'";
                                        generateXls(conn, st, filename, query);
                                } catch (SQLException | IOException e1) {
                                        e1.printStackTrace();
                                }
                        }
                        // generating excel file using date
                        else if (textField.getText().equals("") && txtDdmmyy_1.getText().equals("")
                                && comboBox.getSelectedItem().equals("") &&
textField_2.getText().equals("")) {
                                try {
                                        String filename_1 = txtSpecifyFileName.getText();
                                        String filename = filename_1;
                                        String name = txtDdmmyy.getText();
                                        String query = "Select * from SisClient where Job_date=" + "'" + name +
"'";
                                        generateXls(conn, st, filename, query);
                                } catch (SQLException | IOException e1) {
                                        e1.printStackTrace();
                                }
                        }
                        // generating excel file using Delivery Date
                        else if (textField.getText().equals("") && txtDdmmyy.getText().equals("")
                                && comboBox.getSelectedItem().equals("") &&
textField_2.getText().equals("")) {
                                try {
                                        String filename_1 = txtSpecifyFileName.getText();
                                        String filename = filename_1;
                                        String name = txtDdmmyy_1.getText();
                                        String query = "Select * from SisClient where Delivery_date=" + "'" +
name + "'";
                                        generateXls(conn, st, filename, query);
                                } catch (SQLException | IOException e1) {
                                        e1.printStackTrace();
                                }
                        }
                        // generating excel file using Machine
                        else if (textField.getText().equals("") && txtDdmmyy.getText().equals("")
                                && txtDdmmyy_1.getText().equals("") &&
textField_2.getText().equals("")) {
                                try {
                                        String filename_1 = txtSpecifyFileName.getText();
                                        String filename = filename_1;
                                        String name = (String) comboBox.getSelectedItem();
                                        String query = "Select * from SisClient where Machine=" + "'" + name +
"'";
                                        generateXls(conn, st, filename, query);
```

```
                                        } catch (SQLException | IOException e1) {
                                                e1.printStackTrace();
                                        }
                                }
                                // generating excel file using status
                                else if (textField.getText().equals("") && txtDdmmyy.getText().equals("")
                                                && txtDdmmyy_1.getText().equals("") &&
comboBox.getSelectedItem().equals("")) {
                                        try {
                                                String filename_1 = txtSpecifyFileName.getText();
                                                String filename = filename_1;
                                                String name = textField_2.getText();
                                                String query = "Select * from SisClient where Status=" + "'" + name +
"'";
                                                generateXls(conn, st, filename, query);

                                        } catch (SQLException | IOException e1) {
                                                e1.printStackTrace();
                                        }
                                }
                        }
                });

                separator_2 = new JSeparator();
                GridBagConstraints gbc_separator_2 = new GridBagConstraints();
                gbc_separator_2.insets = new Insets(0, 0, 5, 0);
                gbc_separator_2.gridx = 4;
                gbc_separator_2.gridy = 10;
                contentPane.add(separator_2, gbc_separator_2);

                btnNewButton = new JButton("Insert Comment");
                btnNewButton.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent e) {
                                String mess = txtrJhb.getText();
                                if (mess.equals("")) {
                                        JOptionPane.showMessageDialog(btnNewButton, "Please Enter A Comment");
                                } else {

                                        try {
                                                FileWriter fw = new FileWriter(FileName, true);
                                                fw.write("\n" + mess + "\n");

                                                fw.close();
                                                JOptionPane.showMessageDialog(btnNewButton, "Comment Added");
                                        } catch (IOException e1) {
                                                e1.printStackTrace();
                                        }
                                }
                        }
                });

                txtrJhb = new JTextArea();
                txtrJhb.setBackground(Color.LIGHT_GRAY);
                GridBagConstraints gbc_txtrJhb = new GridBagConstraints();
                gbc_txtrJhb.gridheight = 2;
                gbc_txtrJhb.insets = new Insets(0, 0, 5, 5);
                gbc_txtrJhb.fill = GridBagConstraints.BOTH;
                gbc_txtrJhb.gridx = 1;
```

```
gbc_txtrJhb.gridy = 12;
contentPane.add(txtrJhb, gbc_txtrJhb);

btnNewButton.setFont(new Font("Lucida Grande", Font.BOLD | Font.ITALIC, 13));
GridBagConstraints gbc_btnNewButton = new GridBagConstraints();
gbc_btnNewButton.insets = new Insets(0, 0, 5, 5);
gbc_btnNewButton.gridx = 2;
gbc_btnNewButton.gridy = 12;
contentPane.add(btnNewButton, gbc_btnNewButton);

btnNewButton_1 = new JButton("Show All Comments");
btnNewButton_1.addActionListener(new ActionListener() {
	public void actionPerformed(ActionEvent e) {

		String[] artoStack = parse();

		PriorityQueue<String> pq = new PriorityQueue<String>();

		for (String s : artoStack) {
			pq.offer(s);
		}

		Iterator<String> itr = pq.iterator();
		for (int i = 0; i < pq.size(); i++) {
			if (!pq.isEmpty()) {
				txtrJhb.append(itr.next() + "\n");
			}
		}
	}
});
btnNewButton_1.setFont(new Font("Lucida Grande", Font.BOLD | Font.ITALIC, 13));
GridBagConstraints gbc_btnNewButton_1 = new GridBagConstraints();
gbc_btnNewButton_1.insets = new Insets(0, 0, 5, 5);
gbc_btnNewButton_1.gridx = 2;
gbc_btnNewButton_1.gridy = 13;
contentPane.add(btnNewButton_1, gbc_btnNewButton_1);

JSeparator separator = new JSeparator();
GridBagConstraints gbc_separator = new GridBagConstraints();
gbc_separator.fill = GridBagConstraints.VERTICAL;
gbc_separator.insets = new Insets(0, 0, 5, 5);
gbc_separator.gridx = 1;
gbc_separator.gridy = 14;
contentPane.add(separator, gbc_separator);
setVisible(true);

rdbtnGenerateExcelFile.addActionListener(new ActionListener() {

	public void actionPerformed(ActionEvent e) {
		{
			try {
				System.out.println("kjdwn");
				// String query="Select * from SisClient";
				System.out.println("kjwdn 0");
				generateXls(conn, st, "AdminRecords");
			} catch (SQLException | IOException e1) {
				// TODO Auto-generated catch block
				e1.printStackTrace();
```

```
                                        }

                                }

                        }

                });

}

@SuppressWarnings("deprecation")
public void generateXls(Connection con, Statement st, String filename)
                throws SQLException, FileNotFoundException, IOException {
        try {
                System.out.println("kjwdn 0");
                HSSFWorkbook hwb = new HSSFWorkbook();
                HSSFSheet sheet = hwb.createSheet("Admin sheet");
                HSSFRow rowhead = sheet.createRow((short) 0);
                rowhead.createCell((short) 0).setCellValue("Job No.");
                rowhead.createCell((short) 1).setCellValue("Job Date");
                rowhead.createCell((short) 2).setCellValue("Client Name");
                rowhead.createCell((short) 3).setCellValue("Job Name");
                rowhead.createCell((short) 4).setCellValue("No. Of Forms ");
                rowhead.createCell((short) 5).setCellValue("Print Style");
                rowhead.createCell((short) 6).setCellValue("Paper Used");
                rowhead.createCell((short) 7).setCellValue("No. of Colors");
                rowhead.createCell((short) 8).setCellValue("Paper Qt.");
                rowhead.createCell((short) 9).setCellValue("Plate");
                rowhead.createCell((short) 10).setCellValue("Paper Print");
                rowhead.createCell((short) 11).setCellValue("Book Size");
                rowhead.createCell((short) 12).setCellValue("No. Of Pages");
                rowhead.createCell((short) 13).setCellValue("Plate new/old");
                rowhead.createCell((short) 14).setCellValue("Plate Required");
                rowhead.createCell((short) 15).setCellValue("Paper Supplied");
                rowhead.createCell((short) 16).setCellValue("Paper in Sheets");
                rowhead.createCell((short) 17).setCellValue("Finish Qt.");
                rowhead.createCell((short) 18).setCellValue("Delivery Date");
                rowhead.createCell((short) 19).setCellValue("Status");
                rowhead.createCell((short) 20).setCellValue("Machine");
                rowhead.createCell((short) 21).setCellValue("Team Leader Name");
                String query = "Select * from SisClient";
                Statement statement = con.createStatement();
                ResultSet rs = statement.executeQuery(query);
                // System.out.println("kjwdn 1" +rs.getString(4));
                int i = 1;
                while (rs.next()) {
                        System.out.println("kjwdn 2");
                        HSSFRow row = sheet.createRow((short) i);
                        row.createCell((short) 0).setCellValue(Integer.toString(rs.getInt("Job_no")));
                        row.createCell((short) 1).setCellValue(rs.getString("Job_date"));
                        row.createCell((short) 2).setCellValue(rs.getString("Client_name"));
                        row.createCell((short) 3).setCellValue(rs.getString("Job_name"));
                        row.createCell((short) 4).setCellValue(rs.getString("No_of_forms"));
                        row.createCell((short) 5).setCellValue(rs.getString("Print_style"));
                        row.createCell((short) 6).setCellValue(rs.getString("Paper_used"));
                        row.createCell((short) 7).setCellValue(rs.getString("No_of_col"));
                        row.createCell((short) 8).setCellValue(rs.getString("Paper_qt"));
```

```
                    row.createCell((short) 9).setCellValue(rs.getString("Plate"));
                    row.createCell((short) 10).setCellValue(rs.getString("Paper_print"));
                    row.createCell((short) 11).setCellValue(rs.getString("Book_size"));
                    row.createCell((short) 12).setCellValue(rs.getString("No_of_pages"));
                    row.createCell((short) 13).setCellValue(rs.getString("Plate_new/old"));
                    row.createCell((short) 14).setCellValue(rs.getString("Paper_Required"));
                    row.createCell((short) 15).setCellValue(rs.getString("Paper_supp"));
                    row.createCell((short) 16).setCellValue(rs.getString("Paper_in_sheets"));
                    row.createCell((short) 17).setCellValue(rs.getString("Finish_quantity"));
                    row.createCell((short) 18).setCellValue(rs.getString("Delivery_date"));
                    row.createCell((short) 19).setCellValue(rs.getString("Status"));
                    row.createCell((short) 20).setCellValue(rs.getString("Machine"));
                    row.createCell((short) 21).setCellValue(rs.getString("Team_Leader"));
                    i++;
                }
                FileOutputStream fileOut = new FileOutputStream(filename);
                hwb.write(fileOut);
                fileOut.close();
                JOptionPane.showMessageDialog(btnGenerateExcelFile, filename + ".xls file generated");

        } catch (Exception ex) {
                System.out.println(ex);

        }
}

public void generateXls(Connection con, Statement st, String filename, String query)
                throws SQLException, FileNotFoundException, IOException {
        try {

                HSSFWorkbook hwb = new HSSFWorkbook();
                HSSFSheet sheet = hwb.createSheet("Admin sheet");

                HSSFRow rowhead = sheet.createRow((short) 0);
                rowhead.createCell((short) 0).setCellValue("Job No.");
                rowhead.createCell((short) 1).setCellValue("Job Date");
                rowhead.createCell((short) 2).setCellValue("Client Name");
                rowhead.createCell((short) 3).setCellValue("Job Name");
                rowhead.createCell((short) 4).setCellValue("No. Of Forms ");
                rowhead.createCell((short) 5).setCellValue("Print Style");
                rowhead.createCell((short) 6).setCellValue("Paper Used");
                rowhead.createCell((short) 7).setCellValue("No. of Colors");
                rowhead.createCell((short) 8).setCellValue("Paper Qt.");
                rowhead.createCell((short) 9).setCellValue("Plate");
                rowhead.createCell((short) 10).setCellValue("Paper Print");
                rowhead.createCell((short) 11).setCellValue("Book Size");
                rowhead.createCell((short) 12).setCellValue("No. Of Pages");
                rowhead.createCell((short) 13).setCellValue("Plate new/old");
                rowhead.createCell((short) 14).setCellValue("Plate Required");
                rowhead.createCell((short) 15).setCellValue("Paper Supplied");
                rowhead.createCell((short) 16).setCellValue("Paper in Sheets");
                rowhead.createCell((short) 17).setCellValue("Finish Qt.");
                rowhead.createCell((short) 18).setCellValue("Delivery Date");
                rowhead.createCell((short) 19).setCellValue("Status");
                rowhead.createCell((short) 20).setCellValue("Machine");
                rowhead.createCell((short) 21).setCellValue("Team Leader");

                // Class.forName("com.mysql.jdbc.Driver");
```

```
                        // Connection con =
                        // DriverManager.getConnection("jdbc:mysql://localhost:3306/test",
                        // "root", "root");
                        st = con.createStatement();
                        ResultSet rs = st.executeQuery(query);
                        int i = 1;
                        while (rs.next()) {
                                HSSFRow row = sheet.createRow((short) i);
                                row.createCell((short) 0).setCellValue(Integer.toString(rs.getInt("Job_no")));
                                row.createCell((short) 1).setCellValue(rs.getString("Job_date"));
                                row.createCell((short) 2).setCellValue(rs.getString("Client_name"));
                                row.createCell((short) 3).setCellValue(rs.getString("Job_name"));
                                row.createCell((short) 4).setCellValue(rs.getString("No_of_forms"));
                                row.createCell((short) 5).setCellValue(rs.getString("Print_style"));
                                row.createCell((short) 6).setCellValue(rs.getString("Paper_used"));
                                row.createCell((short) 7).setCellValue(rs.getString("No_of_col"));
                                row.createCell((short) 8).setCellValue(rs.getString("Paper_qt"));
                                row.createCell((short) 9).setCellValue(rs.getString("Plate"));
                                row.createCell((short) 10).setCellValue(rs.getString("Paper_print"));
                                row.createCell((short) 11).setCellValue(rs.getString("Book_size"));
                                row.createCell((short) 12).setCellValue(rs.getString("No_of_pages"));
                                row.createCell((short) 13).setCellValue(rs.getString("Plate_new/old"));
                                row.createCell((short) 14).setCellValue(rs.getString("Paper_Required"));
                                row.createCell((short) 15).setCellValue(rs.getString("Paper_supp"));
                                row.createCell((short) 16).setCellValue(rs.getString("Paper_in_sheets"));
                                row.createCell((short) 17).setCellValue(rs.getString("Finish_quantity"));
                                row.createCell((short) 18).setCellValue(rs.getString("Delivery_date"));
                                row.createCell((short) 19).setCellValue(rs.getString("Status"));
                                row.createCell((short) 20).setCellValue(rs.getString("Machine"));
                                row.createCell((short) 21).setCellValue(rs.getString("Team_Leader"));

                                i++;
                        }
                        FileOutputStream fileOut = new FileOutputStream(filename);
                        hwb.write(fileOut);
                        fileOut.close();
                        JOptionPane.showMessageDialog(btnGenerateExcelFile, filename + ".xls file generated");

                } catch (Exception ex) {
                        System.out.println(ex);

                }

        }
}
```

Code For DeleteStaff.java

```
import java.awt.Color;
import java.awt.Font;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
```

```
import java.sql.Statement;

import javax.swing.BorderFactory;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;
import javax.swing.border.EmptyBorder;

public class DeleteSatff extends JFrame {

	private JPanel contentPane;
	private JTextField textField;
	private JButton btnDelete;
	private JButton btnBack;

	/**
	 * Create the frame.
	 */
	public DeleteSatff(Connection conn, PreparedStatement st) {
		setTitle("Delete Panel");
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		setBounds(100, 100, 358, 169);
		contentPane = new JPanel();
		contentPane.setBackground(Color.WHITE);
		contentPane.setBorder(BorderFactory.createLineBorder(Color.black));
		setContentPane(contentPane);
		GridBagLayout gbl_contentPane = new GridBagLayout();
		gbl_contentPane.columnWidths = new int[] { 0, 0, 66, 0, 0 };
		gbl_contentPane.rowHeights = new int[] { 0, 0, 0, 0, 0 };
		gbl_contentPane.columnWeights = new double[] { 0.0, 0.0, 0.0, 1.0, Double.MIN_VALUE };
		gbl_contentPane.rowWeights = new double[] { 0.0, 0.0, 0.0, 0.0, Double.MIN_VALUE };
		contentPane.setLayout(gbl_contentPane);

		JLabel lblSpecifyJobNumber = new JLabel("Specify Job Number to Delete Record");
		lblSpecifyJobNumber.setFont(new Font("Times", Font.BOLD, 14));
		GridBagConstraints gbc_lblSpecifyJobNumber = new GridBagConstraints();
		gbc_lblSpecifyJobNumber.insets = new Insets(0, 0, 5, 5);
		gbc_lblSpecifyJobNumber.anchor = GridBagConstraints.EAST;
		gbc_lblSpecifyJobNumber.gridx = 1;
		gbc_lblSpecifyJobNumber.gridy = 1;
		contentPane.add(lblSpecifyJobNumber, gbc_lblSpecifyJobNumber);

		textField = new JTextField();
		GridBagConstraints gbc_textField = new GridBagConstraints();
		gbc_textField.insets = new Insets(0, 0, 5, 5);
		gbc_textField.fill = GridBagConstraints.HORIZONTAL;
		gbc_textField.gridx = 2;
		gbc_textField.gridy = 1;
		contentPane.add(textField, gbc_textField);
		textField.setColumns(10);

		btnBack = new JButton("Back ");
		btnBack.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent e) {
```

```
                        dispose();
                        new StaffForm(conn, st);
                }
        });
        btnBack.setFont(new Font("Times New Roman", Font.BOLD | Font.ITALIC, 13));
        GridBagConstraints gbc_btnBack = new GridBagConstraints();
        gbc_btnBack.insets = new Insets(0, 0, 0, 5);
        gbc_btnBack.gridx = 1;
        gbc_btnBack.gridy = 3;
        contentPane.add(btnBack, gbc_btnBack);

        btnDelete = new JButton("Delete");
        btnDelete.setFont(new Font("Times New Roman", Font.BOLD | Font.ITALIC, 13));
        GridBagConstraints gbc_btnDelete = new GridBagConstraints();
        gbc_btnDelete.insets = new Insets(0, 0, 0, 5);
        gbc_btnDelete.gridx = 2;
        gbc_btnDelete.gridy = 3;
        contentPane.add(btnDelete, gbc_btnDelete);
        btnDelete.addActionListener(new ActionListener() {
                PreparedStatement st;
                Statement s;

                public void actionPerformed(ActionEvent e) {

                        if (textField.getText().equals("")) {
                                JOptionPane.showMessageDialog(btnDelete, "Incomplete Fields");
                        } else {

                                String q2 = "select Job_no from SisClient";
                                try {
                                        s = conn.createStatement();
                                        ResultSet rs = s.executeQuery(q2);
                                        while (rs.next()) {
                                                if (rs.getInt(1) == Integer.parseInt(textField.getText())) {

                                                        String qu = "delete from SisClient where Job_no=" + "'"
                                                                        + Integer.parseInt(textField.getText())
+ "'";

                                                        try {
                                                                st = conn.prepareStatement(qu);
                                                                st.executeUpdate(qu);
                                                                JOptionPane.showMessageDialog(btnDelete,
"Record Deleted !");
                                                        } catch (SQLException e1) {
                                                                // TODO Auto-generated catch block
                                                                e1.printStackTrace();
                                                        }

                                                        textField.setText("");

                                                }
                                        }
                                } catch (SQLException e2) {
                                        // TODO Auto-generated catch block
                                        e2.printStackTrace();
                                }
```

```
                                        }
                                }

                        });
                        setVisible(true);
                }

}
```

Code for OpenForm.java

```
import java.awt.Color;
import java.awt.EventQueue;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.BorderFactory;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;
import java.awt.Toolkit;
import javax.swing.ImageIcon;
import java.awt.Font;

public class OpenForm extends JFrame{

        private JPanel contentPane;
        private JPasswordField passwordField;
        private JTextField textField;
        private boolean rst;
        private Statement st;
        private static Connection conn;


        /**
         * Launch the application.
         */
        public static void main(String[] args) {

                EventQueue.invokeLater(new Runnable() {

                        public void run() {
```

```
                                try {

                                        String driver = "com.mysql.jdbc.Driver";
                                        String url = "jdbc:mysql://localhost/Sis";
                                        String user = "root";
                                        String pass = "1234";
                                        Class.forName(driver);

                                        conn = DriverManager.getConnection(url, user, pass);
                                        System.out.println("Connected");

                                        OpenForm frame = new OpenForm();
                                        frame.setVisible(true);
                                } catch (Exception e) {
                                        e.printStackTrace();
                                }
                        }
                });
        }

        /**
         * Create the frame.
         */
        public OpenForm() {
                //gets the salasar image
                setIconImage(Toolkit.getDefaultToolkit().getImage("/Users/DGair/Desktop/Screen Sho"
                                + "t 2017-07-09 at 4.52.26 PM.png"));
                setTitle("Login Panel");
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setBounds(100, 100, 358, 236);
                contentPane = new JPanel();
                contentPane.setBackground(Color.WHITE);
                contentPane.setBorder(BorderFactory.createLineBorder(Color.BLACK));
                setContentPane(contentPane);

                GridBagLayout gbl_contentPane = new GridBagLayout();
                gbl_contentPane.columnWidths = new int[]{0, 143, 0, 0, 0, 0, 0, 0};
                gbl_contentPane.rowHeights = new int[]{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
                gbl_contentPane.columnWeights = new double[]{0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, Double.MIN_VALUE};
                gbl_contentPane.rowWeights = new double[]{0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
Double.MIN_VALUE};
                contentPane.setLayout(gbl_contentPane);

                JLabel lblNewLabel = new JLabel("");
                GridBagConstraints gbc_lblNewLabel = new GridBagConstraints();
                gbc_lblNewLabel.insets = new Insets(0, 0, 5, 5);
                gbc_lblNewLabel.gridx = 3;
                gbc_lblNewLabel.gridy = 0;
                contentPane.add(lblNewLabel, gbc_lblNewLabel);
                DefaultComboBoxModel comboBox_setuser=new DefaultComboBoxModel();
                comboBox_setuser.addElement("Staff");
                comboBox_setuser.addElement("Admin");

                JLabel lblNewLabel_2 = new JLabel("");
                lblNewLabel_2.setIcon(new ImageIcon("/Users/DGair/Desktop/Screen Shot 2017-08-05 at 4.40.48
PM.png"));
                GridBagConstraints gbc_lblNewLabel_2 = new GridBagConstraints();
                gbc_lblNewLabel_2.insets = new Insets(0, 0, 5, 5);
```

```
		gbc_lblNewLabel_2.gridx = 2;
		gbc_lblNewLabel_2.gridy = 1;
		contentPane.add(lblNewLabel_2, gbc_lblNewLabel_2);

		JLabel lblNewLabel_1 = new JLabel("");
		lblNewLabel_1.setIcon(new ImageIcon("/Users/DGair/Desktop/New Java Programs
/SoftwareSIS/pic.png"));
		GridBagConstraints gbc_lblNewLabel_1 = new GridBagConstraints();
		gbc_lblNewLabel_1.insets = new Insets(0, 0, 5, 5);
		gbc_lblNewLabel_1.gridx = 2;
		gbc_lblNewLabel_1.gridy = 3;
		contentPane.add(lblNewLabel_1, gbc_lblNewLabel_1);

		JLabel lblUsername = new JLabel("Username :");
		lblUsername.setFont(new Font("Times New Roman", Font.BOLD | Font.ITALIC, 14));
		GridBagConstraints gbc_lblUsername = new GridBagConstraints();
		gbc_lblUsername.insets = new Insets(0, 0, 5, 5);
		gbc_lblUsername.gridx = 1;
		gbc_lblUsername.gridy = 5;
		contentPane.add(lblUsername, gbc_lblUsername);

		textField = new JTextField();
		textField.setColumns(10);
		GridBagConstraints gbc_textField = new GridBagConstraints();
		gbc_textField.insets = new Insets(0, 0, 5, 5);
		gbc_textField.fill = GridBagConstraints.HORIZONTAL;
		gbc_textField.gridx = 2;
		gbc_textField.gridy = 5;
		contentPane.add(textField, gbc_textField);

		JLabel lblPassword = new JLabel("Password :");
		lblPassword.setFont(new Font("Times", Font.BOLD | Font.ITALIC, 14));
		GridBagConstraints gbc_lblPassword = new GridBagConstraints();
		gbc_lblPassword.insets = new Insets(0, 0, 5, 5);
		gbc_lblPassword.gridx = 1;
		gbc_lblPassword.gridy = 6;
		contentPane.add(lblPassword, gbc_lblPassword);

		passwordField = new JPasswordField();
		passwordField.setColumns(10);
		GridBagConstraints gbc_passwordField = new GridBagConstraints();
		gbc_passwordField.insets = new Insets(0, 0, 5, 5);
		gbc_passwordField.fill = GridBagConstraints.HORIZONTAL;
		gbc_passwordField.gridx = 2;
		gbc_passwordField.gridy = 6;
		contentPane.add(passwordField, gbc_passwordField);

		JComboBox comboBox = new JComboBox();
		comboBox.setFont(new Font("Helvetica", Font.ITALIC, 13));
		comboBox.setModel(comboBox_setuser);
		comboBox.setMaximumRowCount(8);
		GridBagConstraints gbc_comboBox = new GridBagConstraints();
		gbc_comboBox.fill = GridBagConstraints.HORIZONTAL;
		gbc_comboBox.insets = new Insets(0, 0, 5, 5);
		gbc_comboBox.gridx = 2;
		gbc_comboBox.gridy = 7;
		contentPane.add(comboBox, gbc_comboBox);
```

```
		JButton btnLogin = new JButton("Login");
		btnLogin.setFont(new Font("Times", Font.BOLD, 13));
		btnLogin.addActionListener(new ActionListener() {
			private boolean vas;

			public void actionPerformed(ActionEvent e) {

				String username=textField.getText();
				char[] password_char=passwordField.getPassword();
				String usertype= (String) comboBox.getSelectedItem();
				//performCheck(password_char) methods to perform checks according to limit/char/capitalisation of password
				if (performCheck(password_char)) {
					String pass = new String(password_char);
					String qu = "select * from Logger";
					PreparedStatement state;
					try {
						state = conn.prepareStatement(qu);
						ResultSet rst = state.executeQuery();
						// method to check if password and username exist in the
						// database
						if (checkCredentials(rst, username, pass, usertype)) {
							if (usertype.equals("Admin")) {
								dispose();
								// opens the Admin form
								new AdminForm(conn, st);
							} else {
								dispose();
								// opens the staffform
								new StaffForm(conn, st);
							}
						} else {
							textField.setText("");
							passwordField.setText("");
							JOptionPane.showMessageDialog(btnLogin, "InCorrect");
						}

				} catch (SQLException e1) {
					// TODO Auto-generated catch block
					e1.printStackTrace();
				}



			}

			}

			private boolean performCheck(char[] password_char) {

				boolean passw=false;

				if(password_char.length<7)
				{
					JOptionPane.showMessageDialog(btnLogin, "Password must be atleast 8 characters and must Contain Capitals. TRY AGAIN");
					passw=false;
				}
```

```
                        else if (password_char.length>=7)
                        {
                                char[] correctPass = { 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H',
                                                'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S',
                                                'T','U', 'V', 'W', 'X', 'Y', 'Z'};

                                outerloop:
                                for(int i=0; i<password_char.length; i++)
                                {
                                        for(int j=0; j<correctPass.length; j++)
                                        {
                                                if(correctPass[j] == password_char[i])
                                                {
                                                        passw=true;
                                                        break outerloop;
                                                }
                                                else
                                                {
                                                        continue;
                                                }

                                        }
                                }
                                if(passw==false)
                                {

                                        JOptionPane.showMessageDialog(btnLogin, "Password must contain
Capitalization");
                                }


                        }
                        return passw;

                }
        });
        GridBagConstraints gbc_btnLogin = new GridBagConstraints();
        gbc_btnLogin.insets = new Insets(0, 0, 5, 5);
        gbc_btnLogin.gridx = 1;
        gbc_btnLogin.gridy = 8;
        contentPane.add(btnLogin, gbc_btnLogin);

        JButton btnRegister = new JButton("Register");
        btnRegister.setFont(new Font("Times", Font.BOLD, 13));
        btnRegister.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {

                        String username=textField.getText();
                        char[] password_char=passwordField.getPassword();
                        String usertype= (String) comboBox.getSelectedItem();
                        String pass=new String(password_char);
                        if(performCheck(password_char))
                        {
                                String qu_2="insert into Logger values (" +"'" +username +"'" +"," +"'"
                        +pass +"'" +"," +"'" +usertype +"'" +")" ;


                                PreparedStatement state;
```

```
                                        try {
                                                int action=JOptionPane.showConfirmDialog(btnRegister, "Do you want
to register now",
                                                                "Confirmation Panel",
JOptionPane.OK_CANCEL_OPTION);
                                                if(action==JOptionPane.OK_OPTION)
                                                {
                                                        state = conn.prepareStatement(qu_2);
                                                        int rt=state.executeUpdate();
                                                        JOptionPane.showMessageDialog(btnRegister, "You are Added
now try Loggin in");
                                                }

                                        } catch (SQLException e1) {
                                                // TODO Auto-generated catch block
                                                e1.printStackTrace();
                                        }
                                }



                        }

                        private boolean performCheck(char[] password_char) {

                                boolean passw=false;

                                if(password_char.length<7)
                                {
                                        JOptionPane.showMessageDialog(btnLogin, "Password must be atleast 8
characters and must Contain Capitals. TRY AGAIN");
                                        passw=false;
                                }
                                else if (password_char.length>=7)
                                {
                                        char[] correctPass = { 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H',
                                                        'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S',
                                                        'T','U', 'V', 'W', 'X', 'Y', 'Z'};

                                        outerloop:
                                        for(int i=0; i<password_char.length; i++)
                                        {
                                                for(int j=0; j<correctPass.length; j++)
                                                {
                                                        if(correctPass[j] == password_char[i])
                                                        {
                                                                passw=true;
                                                                break outerloop;
                                                        }

                                                }
                                        }
                                        if(passw==false)
                                        {

                                                JOptionPane.showMessageDialog(btnLogin, "Password must contain
Capitalization");
                                        }
```

```
                        }
                        return passw;

                    }
                });
                GridBagConstraints gbc_btnRegister = new GridBagConstraints();
                gbc_btnRegister.insets = new Insets(0, 0, 5, 5);
                gbc_btnRegister.gridx = 2;
                gbc_btnRegister.gridy = 8;
                contentPane.add(btnRegister, gbc_btnRegister);
        }

        public boolean checkCredentials(ResultSet rst2, String username, String pass, String usertype) throws
SQLException {
                // TODO Auto-generated method stub
                boolean f = false;
                while (rst2.next()) {
                        if (rst2.getString(1).equals(username) && rst2.getString(2).equals(pass)
                                        && rst2.getString(3).equals(usertype)) {
                                f = true;
                                break;
                        } else {
                                f = false;
                        }
                }
                return f;
        }

}
```

Code for StaffForm.java

```
import java.awt.Color;
import java.awt.Dimension;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.BorderFactory;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.border.Border;

public class StaffForm extends JFrame {

        private static final long serialVersionUID = 1L;
        private JButton btnAdd;
        private JButton btnUpdate;
        private JButton btnDelete;
```

```
@SuppressWarnings("unused")
private Connection conn;
@SuppressWarnings("unused")
private Statement st;
@SuppressWarnings("unused")
private JLabel lblNewLabel_1;

public StaffForm(Connection conn, Statement st){
	this.conn=conn;
	this.st=st;
	setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
	@SuppressWarnings("unused")
	Border innerborder=BorderFactory.createTitledBorder("Admin Query");
	btnAdd=new JButton("Add");
	btnAdd.setFont(new Font("Times", Font.BOLD | Font.ITALIC, 14));
	btnUpdate=new JButton("Update");
	btnUpdate.setFont(new Font("Times", Font.BOLD | Font.ITALIC, 14));
	btnDelete=new JButton("Delete");
	btnDelete.setFont(new Font("Times", Font.BOLD | Font.ITALIC, 14));

	btnAdd.addActionListener(new ActionListener(){
		@SuppressWarnings("unused")
		PreparedStatement state;
		public void actionPerformed(ActionEvent e) {
			String qu1="Select max(Job_no) from SisClient";
			PreparedStatement state;
			try {
					state = conn.prepareStatement(qu1);
					ResultSet rst = state.executeQuery();
					while(rst.next()){
					int countJobN= rst.getInt(1);
					dispose();
					new AddRecord(conn,state,countJobN);
				}
			} catch (SQLException e2) {
				// TODO Auto-generated catch block
				e2.printStackTrace();
			}
		}

	});


	btnUpdate.addActionListener(new ActionListener(){
		public void actionPerformed(ActionEvent e) {
			dispose();
			new UpdateStaff(conn,(PreparedStatement) st);
		}

	});

	btnDelete.addActionListener(new ActionListener(){
		public void actionPerformed(ActionEvent e) {
			dispose();
			new DeleteSatff(conn,(PreparedStatement) st);
		}

	});
```

```
		JLabel label=new JLabel("Choose an Option");
		btnDelete.setFont(new Font("Times", Font.BOLD , 14));
		setLayout(new FlowLayout());
		//add(lblNewLabel_1);
		getContentPane().setBackground(Color.WHITE);
		add(label);
		add(btnAdd);
		add(btnUpdate);
		add(btnDelete);
		setSize(150,200);
		setVisible(true);
	}

}
```

Code for TACMehtodsE.java

```
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.Map;
import java.util.Map.Entry;
import java.util.Scanner;

public class TACMethodE implements TACMethods {
	private boolean mpf = false;

	@Override
	public boolean maptofile(String file, Map<String, String> tc) throws IOException {
		mpf = false;

		int toprint = tc.size();
		FileWriter fstream;
		BufferedWriter out;

		String fileName = file + ".txt";

		fstream = new FileWriter(fileName);
		out = new BufferedWriter(fstream);

		int count = 0;

		Iterator<Entry<String, String>> it = tc.entrySet().iterator();

		while (it.hasNext() && count < toprint) {

			Entry<String, String> pairs = it.next();
			out.write(pairs.getKey() + ":" + pairs.getValue() + "\n");
			count++;
		}
```

```
		out.close();
		return false;
}

String[] posOfTitle;
private String filenamer;
private String empl;

@Override

public String[] parseFile(String filename, String employee) throws IOException {

		try {
			this.filenamer = filename;
			this.empl = employee;
			filename = "/Users/DGair/Desktop/New Java Programs /SoftwareSIS/" + filename + ".txt";
			File f = new File(filename);

			// Check if File is Exists and is in the Directory specified.

			if (f.exists() && !f.isDirectory()) {
				Scanner sc = new Scanner(f);
				List<String> people = new ArrayList<String>();

				//adding contents of the file to an ArrayList

					while (sc.hasNextLine()) {
						String line = sc.nextLine();
						String[] details = line.split(":");
						String gender = details[0];
						String name = details[1];
						people.add(gender);
						people.add(name);
					}

					//extracting contents from the array list to 2 arrays
					//one array should contain the client names
					//the other array should contain client names
					//Both the array are parallel in nature

					String[] peopleArr = new String[people.size()];
					people.toArray(peopleArr);

					String[] ar1 = new String[peopleArr.length / 2];
					String[] ar2 = new String[peopleArr.length / 2];
					int i1 = 1;
					int i2 = 1;
					ar1[0] = peopleArr[0];
					ar2[0] = peopleArr[1];
					for (int w = 2; w < peopleArr.length; w++)
					{
						if ((w % 2) == 0) {
							ar1[i1] = peopleArr[w];
							i1++;
						} else {
							ar2[i2] = peopleArr[w];
							i2++;
						}
```

```
                }

                boolean cont = false;

                //compare the employee name (by the user)
                // with the name in the array
                // if found, store position in Array List

        ArrayList<Integer> pos = new ArrayList<Integer>();

        for (int i = 0; i < ar2.length; i++) {
                if (ar2[i].equals(empl)) {
                        pos.add(i);
                        cont = true;
                }
        }

                //Transfer the values of the Array List to an Array
                //Use the new Array to find the position of the client name
                //in the parallel Array created before.

        if (cont == true) {
                int[] position = new int[pos.size()];
                for (int i = 0; i < position.length; i++) {
                        position[i] = pos.get(i).intValue();
                }

                posOfTitle = new String[pos.size()];
                for (int i = 0; i < position.length; i++) {
                        int val = position[i];
                        posOfTitle[i] = ar1[val];
                }
        }

                        // If employee name specified by the user does not match
                        // with the name in the Arraylist parsed from the text file
                        // Use the method to return that no values exist.

        else {
                posOfTitle = new String[1];
                posOfTitle[0] = "NO SUCH EMPLOYEE EXISTS";
        }
    }

    // If No file exsit by the name specified by the user
    // Use the method to return that no such file exists

    else {
                posOfTitle = new String[1];
                posOfTitle[0] = "NO SUCH FILE EXISTS";

        }

} catch (FileNotFoundException e) {
        e.printStackTrace();
}
// Return the string array containing
```

```
                // the client name or the failure message

                return posOfTitle;
        }

}
```

Code for TACMethods.java

```
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;

interface TACMethods {

        public abstract boolean maptofile(String file,Map<String, String> tc) throws IOException;

        public abstract String[] parseFile(String filename, String employee) throws FileNotFoundException, IOException;
}
```

Code for TeamsAndClient.java

```
import java.awt.BorderLayout;
import java.awt.EventQueue;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.HashMap;
import java.util.Map;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import java.awt.GridBagLayout;
import javax.swing.JButton;
import java.awt.GridBagConstraints;
import java.awt.Insets;
import javax.swing.JLabel;
import javax.swing.JOptionPane;

import java.awt.event.ActionListener;
import java.io.IOException;
import java.awt.event.ActionEvent;
import javax.swing.JTextField;
import java.awt.Font;
import javax.swing.JTextPane;
import javax.swing.JTextArea;

public class TeamAndClient extends JFrame {

        private JPanel contentPane;
        Connection conn;

        Map<String, String> tc = new HashMap<String, String>();
```

```
private JTextField txtEnterFileName;
private JLabel lblToSearchFor;
private JTextField txtEmployeeName;
private JTextField txtFileName;
private JButton btnSearch;
private JTextArea textArea;
private JButton btnBack;

public TeamAndClient(Connection conn, Statement st) throws SQLException {

	setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
	setBounds(100, 100, 565, 428);
	contentPane = new JPanel();
	contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
	setContentPane(contentPane);
	GridBagLayout gbl_contentPane = new GridBagLayout();
	gbl_contentPane.columnWidths = new int[] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
	gbl_contentPane.rowHeights = new int[] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
	gbl_contentPane.columnWeights = new double[] { 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 0.0, 1.0, 0.0,
			Double.MIN_VALUE };
	gbl_contentPane.rowWeights = new double[] { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0,
			Double.MIN_VALUE };
	contentPane.setLayout(gbl_contentPane);

	JLabel lblTeamsAndClients = new JLabel("Teams And Clients");
	GridBagConstraints gbc_lblTeamsAndClients = new GridBagConstraints();
	gbc_lblTeamsAndClients.insets = new Insets(0, 0, 5, 5);
	gbc_lblTeamsAndClients.gridx = 3;
	gbc_lblTeamsAndClients.gridy = 0;
	contentPane.add(lblTeamsAndClients, gbc_lblTeamsAndClients);

	setVisible(true);

	txtEnterFileName = new JTextField();
	txtEnterFileName.setFont(new Font("Arial Narrow", Font.ITALIC, 11));
	txtEnterFileName.setText("Enter File Name");
	GridBagConstraints gbc_txtEnterFileName = new GridBagConstraints();
	gbc_txtEnterFileName.insets = new Insets(0, 0, 5, 5);
	gbc_txtEnterFileName.fill = GridBagConstraints.HORIZONTAL;
	gbc_txtEnterFileName.gridx = 3;
	gbc_txtEnterFileName.gridy = 2;
	contentPane.add(txtEnterFileName, gbc_txtEnterFileName);
	txtEnterFileName.setColumns(10);

	JButton btnCreateTextFile = new JButton("Create Text File For Team And Client Relationship");
	btnCreateTextFile.addActionListener(new ActionListener() {
		public void actionPerformed(ActionEvent e) {
			TACMethods meth = new TACMethodE();
			try {
				String file = txtEnterFileName.getText();
				meth.maptofile(file, tc);
				JOptionPane.showMessageDialog(btnCreateTextFile, file + " created");

			} catch (IOException e1) {
				// TODO Auto-generated catch block
				e1.printStackTrace();
			}
```

```
		}
});
GridBagConstraints gbc_btnCreateTextFile = new GridBagConstraints();
gbc_btnCreateTextFile.insets = new Insets(0, 0, 5, 5);
gbc_btnCreateTextFile.gridx = 3;
gbc_btnCreateTextFile.gridy = 3;
contentPane.add(btnCreateTextFile, gbc_btnCreateTextFile);

lblToSearchFor = new JLabel("To Search For An Employee And His Clients");
GridBagConstraints gbc_lblToSearchFor = new GridBagConstraints();
gbc_lblToSearchFor.gridwidth = 5;
gbc_lblToSearchFor.insets = new Insets(0, 0, 5, 5);
gbc_lblToSearchFor.gridx = 2;
gbc_lblToSearchFor.gridy = 5;
contentPane.add(lblToSearchFor, gbc_lblToSearchFor);

txtEmployeeName = new JTextField();
txtEmployeeName.setFont(new Font("Andale Mono", Font.ITALIC, 11));
txtEmployeeName.setText("Employee Name");
GridBagConstraints gbc_txtEmployeeName = new GridBagConstraints();
gbc_txtEmployeeName.fill = GridBagConstraints.HORIZONTAL;
gbc_txtEmployeeName.insets = new Insets(0, 0, 5, 5);
gbc_txtEmployeeName.gridx = 3;
gbc_txtEmployeeName.gridy = 6;
contentPane.add(txtEmployeeName, gbc_txtEmployeeName);
txtEmployeeName.setColumns(9);

txtFileName = new JTextField();
txtFileName.setFont(new Font("Andale Mono", Font.ITALIC, 11));
txtFileName.setText("FIle Name");
GridBagConstraints gbc_txtFileName = new GridBagConstraints();
gbc_txtFileName.insets = new Insets(0, 0, 5, 5);
gbc_txtFileName.fill = GridBagConstraints.HORIZONTAL;
gbc_txtFileName.gridx = 3;
gbc_txtFileName.gridy = 7;
contentPane.add(txtFileName, gbc_txtFileName);
txtFileName.setColumns(10);

btnSearch = new JButton("Search");
btnSearch.addActionListener(new ActionListener() {
		public void actionPerformed(ActionEvent e) {

			String empName = txtEmployeeName.getText();
			String FileN = txtFileName.getText();
			TACMethods meth1 = new TACMethodE();
			try {
				String[] titles = meth1.parseFile(FileN, empName);
				for (String titl : titles) {
					textArea.append(titl);
					textArea.append("\n");
				}
			} catch (IOException e1) {
				// TODO Auto-generated catch block
				e1.printStackTrace();
			}
		}
});
```

```
		GridBagConstraints gbc_btnSearch = new GridBagConstraints();
		gbc_btnSearch.insets = new Insets(0, 0, 5, 5);
		gbc_btnSearch.gridx = 3;
		gbc_btnSearch.gridy = 8;
		contentPane.add(btnSearch, gbc_btnSearch);

		textArea = new JTextArea();
		GridBagConstraints gbc_textArea = new GridBagConstraints();
		gbc_textArea.insets = new Insets(0, 0, 0, 5);
		gbc_textArea.fill = GridBagConstraints.BOTH;
		gbc_textArea.gridx = 3;
		gbc_textArea.gridy = 10;
		contentPane.add(textArea, gbc_textArea);

		btnBack = new JButton("Back");
		btnBack.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent e) {
				dispose();
				new AdminForm(conn, st);
			}
		});
		GridBagConstraints gbc_btnBack = new GridBagConstraints();
		gbc_btnBack.insets = new Insets(0, 0, 0, 5);
		gbc_btnBack.gridx = 7;
		gbc_btnBack.gridy = 10;
		contentPane.add(btnBack, gbc_btnBack);

		Statement st1 = conn.createStatement();
		String query = "select Job_name, Team_Leader_Name from SisClient"
				+ " inner join Teams on SisClient.Team_Leader = Teams.Team_Leader_Name ";

		ResultSet rs = st1.executeQuery(query);
		while (rs.next()) {
			String job_name = rs.getString("Job_name");
			String team_leader_name = rs.getString("Team_Leader_Name");
			tc.put(job_name, team_leader_name);
		}

	}

}
```