

10/6/2019



Manual Técnico Programa de Gestión.

Programación



Andrés Ceballos Rodríguez

Manual Técnico Programa de Gestión.

Programación

Índice

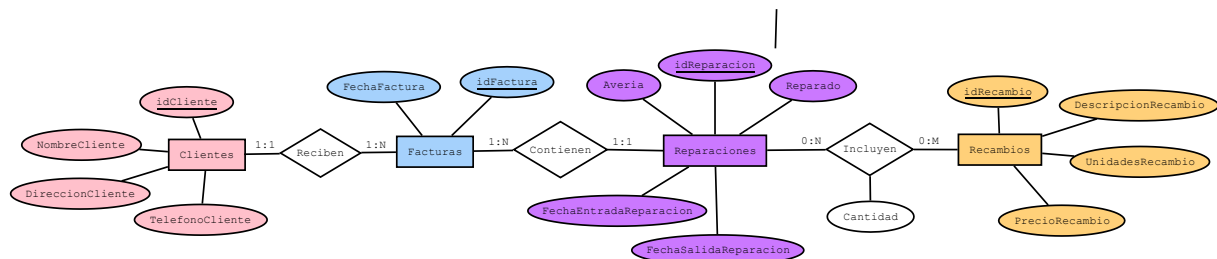
1. Introducción	2
2. Base de Datos	2
3. Librerías.....	5
4. Requisitos e Instalación.....	5
5. Clases Java:	6
7. Bibliografía.....	137

1. Introducción

En el siguiente manual, vamos a exponer los detalles técnicos del Programa de Gestión, así como los detalles de su base de datos (diagrama e-r, esquema relacional, sentencias sql...), como de las librerías externas que hemos usado para la realización del programa, los requisitos para poder utilizar el programa y la explicación de las clases Java que hemos desarrollado para la realización del mismo. También, describiremos el proceso de la creación del ejecutable de nuestro programa.

2. Base de Datos

Modelo ERD



Modelo E/R

CLIENTES (idCliente, NombreCliente, DireccionCliente, TelefonoCliente)

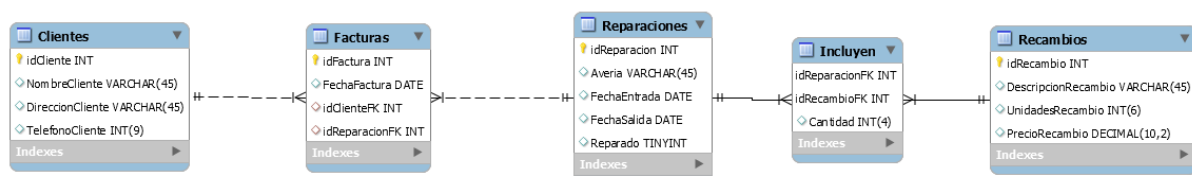
REPARACIONES (idReparacion, Averia, FechaEntradaReparacion, FechaSalidaReparacion, Reparado)

FACTURAS (idFactura, FechaFactura, idClienteFK, idReparacionFK)

RECAMBIOS (idRecambio, DescripcionRecambio, UnidadesRecambio, PrecioRecambio)

INCLUYEN (idReparacionFK, idRecambioFK, Cantidad)

Modelo Workbench



Sentencias SQL

Creación de la Base de datos:

```
CREATE DATABASE IF NOT EXISTS `tallerjava` CHARACTER SET utf8mb4 COLLATE utf8mb4_spanish2_ci;
```

Creación de las Tablas:

Clientes:

```
CREATE TABLE `clientes` (  
  `idCliente` int(11) NOT NULL AUTO_INCREMENT,  
  `NombreCliente` varchar(100) COLLATE utf8mb4_spanish2_ci DEFAULT NULL,  
  `DireccionCliente` varchar(100) COLLATE utf8mb4_spanish2_ci DEFAULT NULL,  
  `TelefonoCliente` int(9) DEFAULT NULL,  
  PRIMARY KEY (`idCliente`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_spanish2_ci;
```

Recambios:

```
CREATE TABLE `recambios` (  
  `idRecambio` int(11) NOT NULL AUTO_INCREMENT,  
  `DescripcionRecambio` varchar(45) COLLATE utf8mb4_spanish2_ci DEFAULT NULL,  
  `UnidadesRecambio` int(6) DEFAULT NULL,  
  `PrecioRecambio` decimal(10,2) DEFAULT NULL,  
  PRIMARY KEY (`idRecambio`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_spanish2_ci;  
/*!40101 SET character_set_client = @saved_cs_client */;
```

Reparaciones:

```
CREATE TABLE `reparaciones` (  
  `idReparacion` int(11) NOT NULL AUTO_INCREMENT,  
  `Averia` varchar(45) COLLATE utf8mb4_spanish2_ci DEFAULT NULL,  
  `FechaEntrada` date DEFAULT NULL,  
  `FechaSalida` date DEFAULT NULL,  
  `Reparado` tinyint(4) DEFAULT NULL,  
  PRIMARY KEY (`idReparacion`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_spanish2_ci;
```

Facturas:

```
CREATE TABLE `facturas` (  
  `idFactura` int(11) NOT NULL AUTO_INCREMENT,  
  `FechaFactura` date DEFAULT NULL,  
  `idClienteFK` int(11) DEFAULT NULL,  
  `idReparacionFK` int(11) DEFAULT NULL,  
  PRIMARY KEY (`idFactura`),  
  KEY `idClienteFK_idx` (`idClienteFK`),  
  KEY `idReparacionFK_idx` (`idReparacionFK`),  
  CONSTRAINT `idClienteFK` FOREIGN KEY (`idClienteFK`) REFERENCES `clientes` (`idCliente`),  
  CONSTRAINT `idReparacionFK` FOREIGN KEY (`idReparacionFK`) REFERENCES `reparaciones` (`idReparacion`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_spanish2_ci;
```

Incluyen:

```
CREATE TABLE `incluyen` (  
  `idIncluyen` int(11) NOT NULL AUTO_INCREMENT,  
  `idReparacionFK` int(11) DEFAULT NULL,  
  `idRecambioFK` int(11) DEFAULT NULL,  
  `cantidad` int(4) DEFAULT NULL,  
  PRIMARY KEY (`idIncluyen`),  
  KEY `idReparacionFK` (`idReparacionFK`),  
  KEY `idRecambioFK` (`idRecambioFK`),  
  CONSTRAINT `incluyen_ibfk_1` FOREIGN KEY (`idReparacionFK`) REFERENCES `reparaciones` (`idReparacion`),  
  CONSTRAINT `incluyen_ibfk_2` FOREIGN KEY (`idRecambioFK`) REFERENCES `recambios` (`idRecambio`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_spanish2_ci;
```

Usuarios:

```
CREATE TABLE `usuarios` (  
  `idUsuario` int(11) NOT NULL AUTO_INCREMENT,  
  `correoUsuario` varchar(45) COLLATE utf8mb4_spanish2_ci NOT NULL,  
  `claveUsuario` varchar(45) COLLATE utf8mb4_spanish2_ci NOT NULL,  
  PRIMARY KEY (`idUsuario`)  
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_spanish2_ci;
```

3. Librerías

Para desarrollar el programa, hemos usado una serie de librerías externas que son las siguientes:

MYSQL Connector 5.1.46 Utilizada para proporcionarnos las utilidades para conectarnos a la Base de Datos.

Itex 5.0.1: Utilizada para exportar las diferentes consultas en un archivo pdf.

4. Requisitos e Instalación








Requisitos

Los requisitos para poder correr el programa son los siguientes:

- MySQL Server
- Java Versión 8
- Archivos de instalación de nuestro programa.

Instalación del programa

Para la instalación del programa, deberemos tener la carpeta de nuestro programa con los siguientes archivos:

Nombre	Fecha de modifica...	Tipo	Tamaño
 imagenes	10/06/2019 13:39	Carpeta de archivos	
 AyudaGestion.chm	10/06/2019 17:34	Archivo de Ayuda ...	737 KB
 instalador.bat	05/06/2019 12:13	Archivo por lotes ...	2 KB
 Manual Tecnico Programa de Gestion.pdf	10/06/2019 9:44	Chrome HTML Do...	387 KB
 Manual Usuario Programa Gestion.pdf	10/06/2019 13:38	Chrome HTML Do...	707 KB
 TallerDeRecambios.jar	10/06/2019 8:44	Executable Jar File	2.551 KB
 TallerJava.sql	10/06/2019 8:49	SQL Text File	8 KB

En esta carpeta, hemos guardado todo lo necesario para el correcto funcionamiento del programa, también hemos creado un instalador.bat en el que tenemos creadas las líneas de código para poder instalar el programa.

```
@echo off
REM Habilitar tildes y ñ
chcp 65001
echo Bienvenido al programa de instalación
REM Crear Ubicaciones
mkdir "C:\Program Files\TallerDeRecambios"
mkdir "C:\Program Files\TallerDeRecambios\imagenes"
mkdir "C:\Program Files\TallerDeRecambios\AyudaPGestion"
REM Mover los ficheros
copy "C:\Users\Andres Ceballos\Desktop\TallerDeRecambios\*.*)" "C:\Program Files\TallerDeRecambios"
copy "C:\Users\Andres Ceballos\Desktop\TallerDeRecambios\imagenes\*.*)" "C:\Program Files\TallerDeRecambios\imagenes"
copy "C:\Users\Andres Ceballos\Desktop\TallerDeRecambios\AyudaPGestion\*.*)" "C:\Program Files\TallerDeRecambios\AyudaPGestion"

cd "C:\Program Files\TallerDeRecambios"
REM Montar la BD
echo Montando la base de datos: Introduzca clave de root
mysql -u root -p -e "CREATE DATABASE TallerJava CHARACTER SET utf8 COLLATE utf8_spanish2_ci;"
echo Importando la base de datos: Introduzca clave de root
mysql -u root -p TallerJava<TallerJava.sql
echo Creando el usuario: Introduzca clave de root
mysql -u root -p -e "CREATE USER 'usuarioTaller'@'localhost' IDENTIFIED BY 'Studium2018;'"
echo Dando permisos al usuario: Introduzca clave de root
mysql -u root -p -e "GRANT ALL ON TallerJava.* TO 'usuarioTaller'@'localhost'"
REM Iniciar
echo Iniciando el programa
TallerDeRecambios.jar
```

Para instalar el programa deberemos pulsar sobre el instalador.bat y a continuación seguiremos los pasos que nos va indicando el instalador.

Cuando tengamos creado el instalador, nos dirigiremos a la carpeta donde hayamos instalado el programa y lo ejecutaremos.

5. Clases Java:

Lo siguiente que vamos a exponer es una breve explicación de cada clase Java que hemos desarrollado que componen nuestro programa.

AddCli.java: Esta clase nos sirve para añadir un cliente.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.TextEvent;
import java.awt.event.TextListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
```

```
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Calendar;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class AddCli implements WindowListener, ActionListener,
TextListener
{

    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/TallerJava?autoReconnect=true&useSSL=f
alse";
    String login = "usuarioTaller";
    String password = "Studium2018;";
    Connection connection = null;
    Statement statement = null;
    ResultSet rs = null;

    String user="";

    JFrame ventanaAddCli = new JFrame ("Añadir Cliente");
    JLabel lblNombreCli = new JLabel ("Nombre:");
    JLabel lblDireccionCli = new JLabel ("Dirección:");
    JLabel lblTelefonoCli = new JLabel ("Teléfono:");

    JTextField txtNombreCli = new JTextField(15);
    JTextField txtDireccionCli = new JTextField(15);
    JTextField txtTelefonoCli = new JTextField(15);

    JButton btnCrear = new JButton("Crear Cliente");
    JButton btnLimpiar = new JButton("Limpiar");

    JPanel pnlPanel = new JPanel();
    JPanel pnlPanel3 = new JPanel();
    JPanel pnlPanel4 = new JPanel();
    JPanel pnlPanel5 = new JPanel();

    public AddCli(String usuario) {
        user = usuario;
```



```
ventanaAddCli.setLayout(new GridLayout(4,1));
ventanaAddCli.setLocationRelativeTo(null);
ventanaAddCli.setSize(400,300);

pnlPanel.setLayout(new FlowLayout());
pnlPanel3.setLayout(new FlowLayout());
pnlPanel4.setLayout(new FlowLayout());
pnlPanel5.setLayout(new FlowLayout());

pnlPanel.add(lblNombreCli);
pnlPanel.add(txtNombreCli);
ventanaAddCli.add(pnlPanel);

pnlPanel3.add(lblDireccionCli);
pnlPanel3.add(txtDireccionCli);
ventanaAddCli.add(pnlPanel3);

pnlPanel4.add(lblTelefonoCli);
pnlPanel4.add(txtTelefonoCli);
ventanaAddCli.add(pnlPanel4);

pnlPanel5.add(btnCrear);
btnCrear.addActionListener(this);
pnlPanel5.add(btnLimpiar);
btnLimpiar.addActionListener(this);
ventanaAddCli.add(pnlPanel5);

ventanaAddCli.addWindowListener(this);
ventanaAddCli.setVisible(true);
}

@Override
public void textValueChanged(TextEvent e) {}

@Override
public void actionPerformed(ActionEvent ae)
{
    if (btnCrear.equals(ae.getSource())) {
        txtNombreCli.selectAll();
        String Nombre = txtNombreCli.getText();
        txtDireccionCli.selectAll();
        String Direccion = txtDireccionCli.getText();
        txtTelefonoCli.selectAll();
        String Telefono = txtTelefonoCli.getText();
```

```

        try
        {
            Class.forName(driver);
            String sentencia = "INSERT INTO clientes
VALUES (null,'" + Nombre + "','', '" + Direccion + "','', '" + Telefono + "','');";
            connection = DriverManager.getConnection(url,
login,password);

            statement = connection.createStatement();
            statement.executeUpdate(sentencia);
            JOptionPane.showMessageDialog(null, "Cliente
creado", "Cliente Creado", JOptionPane.INFORMATION_MESSAGE);
            Calendar horaFecha = Calendar.getInstance();
            int hora, minutos, dia, mes, anyo;
            hora = horaFecha.get(Calendar.HOUR_OF_DAY);
            minutos = horaFecha.get(Calendar.MINUTE);
            dia = horaFecha.get(Calendar.DAY_OF_MONTH);
            mes = horaFecha.get(Calendar.MONTH)+1;
            anyo = horaFecha.get(Calendar.YEAR);
            try {
                FileWriter fw = new
FileWriter("movimientos.log", true);
                BufferedWriter bw = new
BufferedWriter(fw);

                PrintWriter outPut = new PrintWriter(bw);

                outPut.print("[ "+dia+"/ "+mes+"/ "+anyo+" ] [ "+hora+": "+minutos+" ]
"+"["+user+"]"+"["+sentencia+"]"+"\\n");
                outPut.close();
                bw.close();
                fw.close();
            } catch (IOException ioe) {
                System.out.print("Error");
            }
        }
        catch (ClassNotFoundException cnfe)
        {

            JOptionPane.showMessageDialog(null, "Error", cnfe.getMessage(),
JOptionPane.ERROR_MESSAGE);
        }
        catch (SQLException sqle)
        {

            JOptionPane.showMessageDialog(null, "Error", sqle.getMessage(),
JOptionPane.ERROR_MESSAGE);
        }
    }
}

```

```
        finally
        {
            try
            {
                if(connection!=null)
                {
                    connection.close();
                }
            }
            catch (SQLException e)
            {

                JOptionPane.showMessageDialog(null,"Error",e.getMessage(),
                JOptionPane.ERROR_MESSAGE);
            }
        }

    } else if (btnLimpiar.equals(ae.getSource())) {
        txtNombreCli.selectAll();
        txtNombreCli.setText("");
        txtDireccionCli.selectAll();
        txtDireccionCli.setText("");
        txtTelefonoCli.selectAll();
        txtTelefonoCli.setText("");
    }

}

@Override
public void windowActivated(WindowEvent arg0) {}

@Override
public void windowClosed(WindowEvent arg0) {}

@Override
public void windowClosing(WindowEvent arg0) {

    if(ventanaAddCli.isActive()) {
        ventanaAddCli.setVisible(false);
    }
}

@Override
public void windowDeactivated(WindowEvent arg0) {}
```

```
@Override
public void windowDeiconified(WindowEvent arg0) {}

@Override
public void windowIconified(WindowEvent arg0) {}

@Override
public void windowOpened(WindowEvent arg0) {}

}
```

AddFac.java: Esta clase la utilizamos para añadir una factura.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.Choice;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Calendar;

import javax.swing.*.*;

public class AddFac extends JFrame implements WindowListener,
ActionListener{

    /**
     *
     */
    private static final long serialVersionUID = 1L;

    String user;
```

```

JLabel lblFecha = new JLabel ("Fecha Factura :");
JLabel lblClientes = new JLabel ("Cliente:");
JLabel lblReparaciones = new JLabel ("Reparación:");

Choice clientes = new Choice();
Choice reparaciones = new Choice();
JTextField txtFecha = new JTextField(10);

JButton btnCrear = new JButton("Crear Factura");
JButton btnLimpiar = new JButton("Limpiar");

JPanel pnlPanel = new JPanel();
JPanel pnlPanel2 = new JPanel();
JPanel pnlPanel3 = new JPanel();
JPanel pnlPanel4 = new JPanel();

Calendar horaFecha = Calendar.getInstance();
int hora,minutos,dia,mes,anyo;

public AddFac(String usuario)
{
    user = usuario;
    this.setTitle("Añadir Factura");
    this.setLayout(new GridLayout(5,2));
    this.setLocationRelativeTo(null);
    this.setSize(400,300);

    hora = horaFecha.get(Calendar.HOUR_OF_DAY);
    minutos = horaFecha.get(Calendar.MINUTE);
    dia = horaFecha.get(Calendar.DAY_OF_MONTH);
    mes = horaFecha.get(Calendar.MONTH)+1;
    anyo = horaFecha.get(Calendar.YEAR);

    ResultSet selectClientes = ejecutarSelect("SELECT * FROM
clientes",conectar("TallerJava","usuarioTaller","Studium2018;"));
    try {
        while(selectClientes.next())
        {
            String
cli=Integer.toString(selectClientes.getInt("idCliente"));
            cli = cli + ".-"+"
"+selectClientes.getString("nombreCliente");
            clientes.add(cli);
        }
    } catch (SQLException e) {

```

```
JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }

    desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

    ResultSet selectReparaciones = ejecutarSelect("SELECT *
FROM
reparaciones",conectar("TallerJava","usuarioTaller","Studium2018;"))
;
    try {
        while(selectReparaciones.next())
        {
            String
rep=Integer.toString(selectReparaciones.getInt("idReparacion"));
            rep = rep + "-"+
selectReparaciones.getString("Averia");
            reparaciones.add(rep);
        }
    } catch (SQLException e) {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }

    desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

    pnlPanel.setLayout(new FlowLayout());
    pnlPanel2.setLayout(new FlowLayout());
    pnlPanel3.setLayout(new FlowLayout());
    pnlPanel4.setLayout(new FlowLayout());

    pnlPanel.add(lblFecha);
    txtFecha.setText(dia+"/"+mes+"/"+anyo);
    pnlPanel.add(txtFecha);
    this.add(pnlPanel);

    pnlPanel2.add(lblClientes);
    pnlPanel2.add(clientes);
    this.add(pnlPanel2);

    pnlPanel3.add(lblReparaciones);
    pnlPanel3.add(reparaciones);
    this.add(pnlPanel3);
```

```

        pnlPanel4.add(btnCrear);
        btnCrear.addActionListener(this);
        pnlPanel4.add(btnLimpiar);
        btnLimpiar.addActionListener(this);
        this.add(pnlPanel4);

        this.addWindowListener(this);
        this.setVisible(true);
    }
    @Override
    public void actionPerformed(ActionEvent ae) {

        if (btnCrear.equals(ae.getSource()))
        {
            String[] arrayClientes =
clientes.getSelectedItem().toString().split("-");
            int idCliente = Integer.parseInt(arrayClientes[0]);
            String[] arrayReparaciones=
reparaciones.getSelectedItem().toString().split("-");
            int idReparacion =
Integer.parseInt(arrayReparaciones[0]);

            String Fecha = txtFecha.getText();
            String[] arrayFecha = Fecha.split("/");
            Fecha = arrayFecha[2]+"-"+arrayFecha[1]+"-
"+arrayFecha[0];

            String sentencia = "INSERT INTO facturas VALUES
(null,'"+Fecha+"','"+idCliente+"','"+idReparacion+"");

            ejecutarIDA(sentencia,conectar("TallerJava","usuarioTaller","St
udium2018;"));
            ResultSet rsCodFac = ejecutarSelect("select * from
facturas order by 1 desc;",
conectar("TallerJava","usuarioTaller","Stadium2018;"));
            try {
                rsCodFac.next();
                new LineaRepRec(idReparacion,
rsCodFac.getInt("idFactura"));
                this.setVisible(false);
            } catch (SQLException e) {

                JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);

```

```

    }

    desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

    Calendar horaFecha = Calendar.getInstance();
    int hora,minutos,dia,mes,anyo;
    hora = horaFecha.get(Calendar.HOUR_OF_DAY);
    minutos = horaFecha.get(Calendar.MINUTE);
    dia = horaFecha.get(Calendar.DAY_OF_MONTH);
    mes = horaFecha.get(Calendar.MONTH)+1;
    anyo = horaFecha.get(Calendar.YEAR);
    try {
        FileWriter fw = new
FileWriter("movimientos.log", true);
        BufferedWriter bw = new BufferedWriter(fw);
        PrintWriter outPut = new PrintWriter(bw);

        outPut.print "["+dia+"/"+mes+"/"+anyo+"["+hora+": "+minutos+"]
"+"["+user+"]"+"["+sentencia+"]"+"\\n");
        outPut.close();
        bw.close();
        fw.close();
    } catch (IOException ioe) {
        System.out.print("Error");
    }

}

else if (btnLimpiar.equals(ae.getSource()))
{
    txtFecha.selectAll();
    txtFecha.setText("");
}

}

@Override
public void windowActivated(WindowEvent arg0) {
    // TODO Auto-generated method stub

}

@Override
public void windowClosed(WindowEvent arg0) {

```



```
// TODO Auto-generated method stub

}

@Override
public void windowClosing(WindowEvent arg0)
{
    this.setVisible(false);
}

@Override
public void windowDeactivated(WindowEvent arg0) {
    // TODO Auto-generated method stub
}

@Override
public void windowDeiconified(WindowEvent arg0) {
    // TODO Auto-generated method stub
}

@Override
public void windowIconified(WindowEvent arg0) {
    // TODO Auto-generated method stub
}

@Override
public void windowOpened(WindowEvent arg0) {
    // TODO Auto-generated method stub
}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;
    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,
```

```
login,password);
    }
    catch (ClassNotFoundException cnfe)
    {

        JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    catch (SQLException sqle)
    {

        JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    return connection;
}

public void desconectar(Connection c)
{
    try
    {
        if(c!=null)
        {
            c.close();
        }
    }
    catch (SQLException e)
    {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
}

public ResultSet ejecutarSelect(String sentencia, Connection c)
{
    try
    {
        Statement statement = c.createStatement();
        ResultSet rs= statement.executeQuery(sentencia);
        return rs;
    }
    catch(SQLException e)
    {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
```

```
JOptionPane.ERROR_MESSAGE);  
        return null;  
    }  
  
    }  
  
    public void ejecutarIDA(String sentencia, Connection c)  
    {  
        try  
        {  
            Statement statement = c.createStatement();  
            statement.executeUpdate(sentencia);  
            JOptionPane.showMessageDialog(null, "Factura  
creada", "Factura creada con éxito",  
JOptionPane.INFORMATION_MESSAGE);  
            this.setVisible(false);  
  
        }  
        catch(SQLException e)  
        {  
  
            JOptionPane.showMessageDialog(null, e.getMessage(), "Error",  
JOptionPane.ERROR_MESSAGE);  
        }  
  
    }  
  
}
```

AddRec.java: Esta clase nos sirve para poder añadir un recambio.

```
package es.studium.PracticaSegundoTrimestre;  
  
import java.awt.FlowLayout;  
import java.awt.GridLayout;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.awt.event.WindowEvent;  
import java.awt.event.WindowListener;  
import java.io.BufferedWriter;  
import java.io.FileWriter;  
import java.io.IOException;
```

```
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Calendar;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class AddRec implements WindowListener, ActionListener{

    String driver = "com.mysql.jdbc.Driver";
    String url
    ="jdbc:mysql://localhost:3306/TallerJava?autoReconnect=true&useSSL=f
    else";
    String login = "usuarioTaller";
    String password = "Stodium2018;";
    Connection connection = null;
    Statement statement = null;
    ResultSet rs = null;

    String user;

    JFrame ventanaAddRec = new JFrame ("Añadir Recambio");
    JLabel lblDescripcionRec = new JLabel ("Descripción:");
    JLabel lblUnidadesRec = new JLabel ("Unidades:");
    JLabel lblPrecioRec = new JLabel ("Precio:");

    JTextField txtDescripcionRec = new JTextField(10);
    JTextField txtUnidadesRec = new JTextField(10);
    JTextField txtPrecioRec = new JTextField(10);

    JButton btnCrear = new JButton("Crear Recambio");
    JButton btnLimpiar = new JButton("Limpiar");

    JPanel pnlPanel = new JPanel();
    JPanel pnlPanel2 = new JPanel();
    JPanel pnlPanel3 = new JPanel();
    JPanel pnlPanel4 = new JPanel();
```

```
public AddRec(String usuario) {
    user = usuario;
    ventanaAddRec.setLayout(new GridLayout(4,2));
    ventanaAddRec.setLocationRelativeTo(null);
    ventanaAddRec.setSize(400,300);

    pnlPanel1.setLayout(new FlowLayout());
    pnlPanel2.setLayout(new FlowLayout());
    pnlPanel3.setLayout(new FlowLayout());
    pnlPanel4.setLayout(new FlowLayout());

    pnlPanel1.add(lblDescripcionRec);
    pnlPanel1.add(txtDescripcionRec);
    ventanaAddRec.add(pnlPanel1);

    pnlPanel2.add(lblUnidadesRec);
    pnlPanel2.add(txtUnidadesRec);
    ventanaAddRec.add(pnlPanel2);

    pnlPanel3.add(lblPrecioRec);
    pnlPanel3.add(txtPrecioRec);
    ventanaAddRec.add(pnlPanel3);

    pnlPanel4.add(btnCrear);
    btnCrear.addActionListener(this);
    pnlPanel4.add(btnLimpiar);
    btnLimpiar.addActionListener(this);
    ventanaAddRec.add(pnlPanel4);
    ventanaAddRec.addWindowListener(this);
    ventanaAddRec.setVisible(true);
}

@Override
public void actionPerformed(ActionEvent ae)
{
    if (btnCrear.equals(ae.getSource())) {
        txtDescripcionRec.selectAll();
        String Descripcion = txtDescripcionRec.getText();
        txtUnidadesRec.selectAll();
        String Unidades = txtUnidadesRec.getText();
        txtPrecioRec.selectAll();
        String Precio = txtPrecioRec.getText();
    }
}
```

```

        try
        {
            Class.forName(driver);
            String sentencia = "INSERT INTO recambios
VALUES (null,'" + Descripcion + "', '"+Unidades+"', '"+Precio+"');";
            connection = DriverManager.getConnection(url,
login,password);

            statement = connection.createStatement();
            statement.executeUpdate(sentencia);
            JOptionPane.showMessageDialog(null,"Recambio
creado","Recambio Creado con éxito",
JOptionPane.INFORMATION_MESSAGE);
            Calendar horaFecha = Calendar.getInstance();
            int hora,minutos,dia,mes,anyo;
            hora = horaFecha.get(Calendar.HOUR_OF_DAY);
            minutos = horaFecha.get(Calendar.MINUTE);
            dia = horaFecha.get(Calendar.DAY_OF_MONTH);
            mes = horaFecha.get(Calendar.MONTH)+1;
            anyo = horaFecha.get(Calendar.YEAR);
            try {
                FileWriter fw = new
FileWriter("movimientos.log", true);
                BufferedWriter bw = new
BufferedWriter(fw);

                PrintWriter outPut = new PrintWriter(bw);

                outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ]["+hora+": "+minutos+" ]
"+"["+user+"]"+"["+sentencia+"]"+"\\n");
                outPut.close();
                bw.close();
                fw.close();
            } catch (IOException ioe) {
                System.out.print("Error");
            }
        }
        catch (ClassNotFoundException cnfe)
        {
            JOptionPane.showMessageDialog(null,"Error",cnfe.getMessage(),
JOptionPane.ERROR_MESSAGE);
        }
        catch (SQLException sqle)
        {
            JOptionPane.showMessageDialog(null,"Error",sqle.getMessage(),
JOptionPane.ERROR_MESSAGE);
        }
    }
}

```

```
        }  
        finally  
        {  
            try  
            {  
                if(connection!=null)  
                {  
                    connection.close();  
                }  
            }  
            catch (SQLException e)  
            {  
                JOptionPane.showMessageDialog(null,"Error",e.getMessage(),  
JOptionPane.ERROR_MESSAGE);  
            }  
        }  
  
        } else if (btnLimpiar.equals(ae.getSource())) {  
            txtDescripcionRec.selectAll();  
            txtDescripcionRec.setText("");  
            txtUnidadesRec.selectAll();  
            txtUnidadesRec.setText("");  
            txtPrecioRec.selectAll();  
            txtPrecioRec.setText("");  
        }  
    }  
}  
  
@Override  
public void windowActivated(WindowEvent arg0) {}  
  
@Override  
public void windowClosed(WindowEvent arg0) {}  
  
@Override  
public void windowClosing(WindowEvent arg0) {  
    if(ventanaAddRec.isActive()) {  
        ventanaAddRec.setVisible(false);  
    }else {  
        //System.exit(0);  
    }  
}  
}
```

```
@Override
public void windowDeactivated(WindowEvent arg0) {}

@Override
public void windowDeiconified(WindowEvent arg0) {}

@Override
public void windowIconified(WindowEvent arg0) {}

@Override
public void windowOpened(WindowEvent arg0) {}

}
```

AddRep.java: Esta clase la utilizamos para añadir una reparación.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Calendar;

import javax.swing.*;

public class AddRep extends JFrame implements WindowListener,
ActionListener{

    /**
     *
     */
    private static final long serialVersionUID = 1L;
```



```
String user;

JLabel lblAveriaRep = new JLabel ("Avería:");
JLabel lblFechaEntradaRep = new JLabel ("Fecha de Entrada:");
JLabel lblFechaSalidaRep = new JLabel ("Fecha de Salida:");
JLabel lblReparadoRep = new JLabel ("Reparado:");

JTextField txtAveriaRep = new JTextField(10);
JTextField txtFechaEntradaRep = new JTextField(10);
JTextField txtFechaSalidaRep = new JTextField(10);
ButtonGroup chkReparadoRep = new ButtonGroup ();
JRadioButton chkSiRep = new JRadioButton ("Sí", false);
JRadioButton chkNoRep = new JRadioButton ("No", true);

JButton btnCrear = new JButton("Crear Reparación");
JButton btnLimpiar = new JButton("Limpiar");

JPanel pnlPanel = new JPanel();
JPanel pnlPanel2 = new JPanel();
JPanel pnlPanel3 = new JPanel();
JPanel pnlPanel4 = new JPanel();
JPanel pnlPanel5 = new JPanel();

Calendar horaFecha = Calendar.getInstance();
int hora,minutos,dia,mes,anyo;

public AddRep(String usuario)
{
    user = usuario;
    this.setTitle("Añadir Reparación");
    this.setLayout(new GridLayout(5,2));
    this.setLocationRelativeTo(null);
    this.setSize(400,300);

    hora = horaFecha.get(Calendar.HOUR_OF_DAY);
    minutos = horaFecha.get(Calendar.MINUTE);
    dia = horaFecha.get(Calendar.DAY_OF_MONTH);
    mes = horaFecha.get(Calendar.MONTH)+1;
    anyo = horaFecha.get(Calendar.YEAR);

    pnlPanel.setLayout(new FlowLayout());
    pnlPanel2.setLayout(new FlowLayout());
    pnlPanel3.setLayout(new FlowLayout());
    pnlPanel4.setLayout(new FlowLayout());
```

```
pnlPanel5.setLayout(new FlowLayout());

pnlPanel.add(lblAveriaRep);
pnlPanel.add(txtAveriaRep);
this.add(pnlPanel);

pnlPanel2.add(lblFechaEntradaRep);
txtFechaEntradaRep.setText(dia+"/"+mes+"/"+anyo);
pnlPanel2.add(txtFechaEntradaRep);
this.add(pnlPanel2);

pnlPanel3.add(lblFechaSalidaRep);
txtFechaSalidaRep.setText(dia+"/"+mes+"/"+anyo);
pnlPanel3.add(txtFechaSalidaRep);
this.add(pnlPanel3);

pnlPanel4.add(lblReparadoRep);
chkReparadoRep.add(chkSiRep);
chkReparadoRep.add(chkNoRep);
pnlPanel4.add(chkSiRep);
pnlPanel4.add(chkNoRep);
this.add(pnlPanel4);

pnlPanel5.add(btnCrear);
btnCrear.addActionListener(this);
pnlPanel5.add(btnLimpiar);
btnLimpiar.addActionListener(this);
this.add(pnlPanel5);

this.addWindowListener(this);
this.setVisible(true);
}
@Override
public void actionPerformed(ActionEvent ae)
{
    if (btnCrear.equals(ae.getSource()))
    {
        if(chkSiRep.isSelected())
        {
            String FechaEntrada =
txtFechaEntradaRep.getText();
            String[] arrayFechaEntrada =
FechaEntrada.split("/");
            FechaEntrada = arrayFechaEntrada[2]+"-
"+arrayFechaEntrada[1]+"-"+arrayFechaEntrada[0];
```

```

        String FechaSalida =
txtFechaSalidaRep.getText();
        String[] arrayFechaSalida =
FechaSalida.split("/");
        FechaSalida = arrayFechaSalida[2]+"-
"+arrayFechaSalida[1]+"-"+arrayFechaSalida[0];

        String sentencia1 = "INSERT INTO reparaciones
VALUES (null, '"+txtAveriaRep.getText()+"',
 '"+FechaEntrada+"', '"+FechaSalida+"', '1');"
        ejecutarIDA(sentencia1,
conectar("TallerJava","usuarioTaller","Studium2018;"));

        desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

        try {
            FileWriter fw = new
FileWriter("movimientos.log", true);
            BufferedWriter bw = new
BufferedWriter(fw);

            PrintWriter outPut = new PrintWriter(bw);

            outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ] [ "+hora+": "+minutos+" ]
"+" [ "+user+" ] "+" [ "+sentencia1+" ] "+" \n");
            outPut.close();
            bw.close();
            fw.close();
        } catch (IOException ioe) {
            System.out.print("Error");
        }

    } else if (chkNoRep.isSelected())
    {
        String FechaEntrada =
txtFechaEntradaRep.getText();
        String[] arrayFechaEntrada =
FechaEntrada.split("/");
        FechaEntrada = arrayFechaEntrada[2]+"-
"+arrayFechaEntrada[1]+"-"+arrayFechaEntrada[0];

        String FechaSalida =
txtFechaSalidaRep.getText();
        String[] arrayFechaSalida =
FechaSalida.split("/");
        FechaSalida = arrayFechaSalida[2]+"-
"+arrayFechaSalida[1]+"-"+arrayFechaSalida[0];

```

```

        String sentencia2 = "INSERT INTO reparaciones
VALUES (null, '"+txtAveriaRep.getText()+"',
 '"+FechaEntrada+"', '"+FechaSalida+"', '0');"
        ejecutarIDA(sentencia2,
conectar("TallerJava","usuarioTaller","Studium2018;"));

        desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

        try {
            FileWriter fw = new
FileWriter("movimientos.log", true);
            BufferedWriter bw = new
BufferedWriter(fw);

            PrintWriter outPut = new PrintWriter(bw);

            outPut.print("[ "+dia+"/ "+mes+"/ "+anyo+" ] [ "+hora+": "+minutos+" ]
"+ " [ "+user+" ] "+ " [ "+sentencia2+" ] "+ "\n");
            outPut.close();
            bw.close();
            fw.close();
        } catch (IOException ioe) {
            System.out.print("Error");
        }
    }

    } else if (btnLimpiar.equals(ae.getSource())) {
        txtAveriaRep.selectAll();
        txtAveriaRep.setText("");
        txtFechaEntradaRep.selectAll();
        txtFechaEntradaRep.setText("");
        txtFechaSalidaRep.selectAll();
        txtFechaSalidaRep.setText("");
    }
}

@Override
public void windowActivated(WindowEvent arg0) {}

@Override
public void windowClosed(WindowEvent arg0) {}

@Override
public void windowClosing(WindowEvent arg0)
{

```

```
        this.setVisible(false);
    }

    @Override
    public void windowDeactivated(WindowEvent arg0) {}

    @Override
    public void windowDeiconified(WindowEvent arg0) {}

    @Override
    public void windowIconified(WindowEvent arg0) {}

    @Override
    public void windowOpened(WindowEvent arg0) {}

    public Connection conectar(String baseDatos, String usuario,
String clave)
    {
        String driver = "com.mysql.jdbc.Driver";
        String url
="jdbc:mysql://localhost:3306/" + baseDatos + "?autoReconnect=true&useSS
L=false";
        String login = usuario;
        String password = clave;
        Connection connection = null;
        try
        {
            Class.forName(driver);
            connection = DriverManager.getConnection(url,
login,password);
        }
        catch (ClassNotFoundException cnfe)
        {
            JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error 1",
JOptionPane.ERROR_MESSAGE);
        }
        catch (SQLException sqle)
        {
            JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error 2",
JOptionPane.ERROR_MESSAGE);
        }
        return connection;
    }

    public void desconectar(Connection c)
```

```
{
    try
    {
        if(c!=null)
        {
            c.close();
        }
    }
    catch (SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error 3",
        JOptionPane.ERROR_MESSAGE);
    }
}

public void ejecutarIDA(String sentencia, Connection c)
{
    try
    {
        Statement statement = c.createStatement();
        statement.executeUpdate(sentencia);
        JOptionPane.showMessageDialog(null,"Reparación
añadida","Reparación añadida con éxito",
        JOptionPane.INFORMATION_MESSAGE);
    }
    catch(SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error 4",
        JOptionPane.ERROR_MESSAGE);
    }
}
}
```

Ayuda.java: En esta clase, ejecutamos el ejecutable de la ayuda.

```
package es.studium.PracticaSegundoTrimestre;

import java.io.IOException;

public class Ayuda {
```

```
public Ayuda()  
{  
    try  
    {  
        Runtime.getRuntime().exec("hh.exe  
AyudaPGestion/AyudaGestion.chm");  
    }  
    catch (IOException e)  
    {  
        e.printStackTrace();  
    }  
}  
}
```

ConCliList.java: En esta clase, encontramos un listado de los clientes que tenemos en la base de datos con posibilidad de imprimirlo en PDF.

```
package es.studium.PracticaSegundoTrimestre;  
  
import java.awt.BorderLayout;  
import java.awt.FileDialog;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.awt.event.WindowEvent;  
import java.awt.event.WindowListener;  
import java.io.FileNotFoundException;  
import java.io.FileOutputStream;  
import java.sql.Connection;  
import java.sql.DriverManager;  
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 javax.swing.JPanel;  
import javax.swing.JScrollPane;  
import javax.swing.JTable;  
import javax.swing.table.DefaultTableModel;
```

```

import com.itextpdf.text.pdf.PdfPTable;
import com.itextpdf.text.pdf.PdfWriter;
import com.itextpdf.text.Document;
import com.itextpdf.text.DocumentException;
import com.itextpdf.text.Font;
import com.itextpdf.text.Paragraph;

public class ConCliList extends JFrame implements WindowListener,
ActionListener
{
    /**
     *
     */
    private static final long serialVersionUID = 1L;

    DefaultTableModel modelo = new DefaultTableModel();
    JTable tablaClientes= new JTable(modelo);
    JButton btnAceptar = new JButton("Aceptar");
    JButton btnPDF = new JButton("Exportar a PDF");
    JPanel pnl2 = new JPanel();
    ResultSet rs;

    Document documento = new Document();

    public ConCliList() {
        this.setLayout(new BorderLayout());
        this.setLocationRelativeTo(null);
        this.setSize(600,200);
        this.setTitle("Consulta de Clientes");

        this.add(new
JScrollPane(tablaClientes),BorderLayout.CENTER);
        pnl2.add(btnAceptar);
        pnl2.add(btnPDF);
        this.add(pnl2, BorderLayout.SOUTH);
        btnAceptar.addActionListener(this);
        btnPDF.addActionListener(this);

        modelo.addColumn("Nº Cliente");
        modelo.addColumn("Nombre");
        modelo.addColumn("Dirección");
        modelo.addColumn("Teléfono");

        rs = ejecutarSelect("SELECT * FROM clientes",
conectar("TallerJava","usuarioTaller","Studium2018;"));
        try {

```



```
        while (rs.next())
        {
            Object [] fila = new Object[4];

            for (int i=0;i<4;i++) {
                fila[i] = rs.getObject(i+1);
            }
            modelo.addRow(fila);
        }
    } catch (SQLException e) {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
    }
    desconectar(conectar("TallerJava","usuarioTaller"
    ,"Stodium2018;"));
    tablaClientes.setEnabled(false);

    this.addWindowListener(this);
    this.setVisible(true);
}
@Override
public void actionPerformed(ActionEvent ae)
{

    if(btnAceptar.equals(ae.getSource()))
    {
        this.setVisible(false);
    }
    else if (btnPDF.equals(ae.getSource()))
    {
        FileOutputStream ficheroPdf;
        try {
            FileDialog fd = new FileDialog(this,
            "Seleccionar archivo", FileDialog.SAVE);
            fd.setFile("*.pdf");
            fd.setVisible(true);
            String filename =
            fd.getDirectory()+fd.getFile();
            ficheroPdf = new FileOutputStream(filename);

            PdfWriter.getInstance(documento,ficheroPdf).setInitialLeading(2
            0);

            documento.open();
            PdfPTable tabla = new PdfPTable(4);
```

```
Paragraph idCliente = new Paragraph("Nº  
Cliente");  
  
idCliente.getFont().setStyle(Font.BOLD);  
idCliente.getFont().setSize(15);  
tabla.addCell(idCliente);  
  
Paragraph nombreCliente = new  
Paragraph("Nombre");  
  
nombreCliente.getFont().setStyle(Font.BOLD);  
nombreCliente.getFont().setSize(15);  
tabla.addCell(nombreCliente);  
  
Paragraph direccionCliente = new  
Paragraph("Dirección");  
  
direccionCliente.getFont().setStyle(Font.BOLD);  
direccionCliente.getFont().setSize(15);  
tabla.addCell(direccionCliente);  
  
Paragraph telefono = new  
Paragraph("Teléfono");  
  
telefono.getFont().setStyle(Font.BOLD);  
telefono.getFont().setSize(15);  
tabla.addCell(telefono);  
  
ResultSet Co = ejecutarSelect("SELECT * FROM  
clientes;", conectar("TallerJava", "usuarioTaller", "Studium2018;"));  
try {  
    while (Co.next())  
    {  
        for (int i=0; i<4; i++) {  
            if(i==0) {  
  
tabla.addCell(Co.getString("idCliente"));  
                } else if(i==1) {  
  
tabla.addCell(Co.getString("nombreCliente"));  
                } else if(i==2) {  
  
tabla.addCell(Co.getString("direccionCliente"));  
                } else if(i==3) {  
  
tabla.addCell(Co.getString("telefonoCliente"));  
                }  
            }  
        }  
    }  
}
```

```

        documento.add(tabla);
        documento.close();

        JOptionPane.showMessageDialog(null,"Documento pdf creado
correctamente.", "Documento creado",
JOptionPane.INFORMATION_MESSAGE);
    } catch (SQLException e) {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    } catch (FileNotFoundException e1) {
        // TODO Auto-generated catch block

        JOptionPane.showMessageDialog(null,e1.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    } catch (DocumentException e2) {
        // TODO Auto-generated catch block

        JOptionPane.showMessageDialog(null,e2.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }

    }
}
@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}
@Override
public void windowClosing(WindowEvent arg0)
{
    this.setVisible(false);
}

@Override
public void windowDeactivated(WindowEvent arg0) {

}
@Override
public void windowDeiconified(WindowEvent arg0) {

}

@Override

```

```
public void windowIconified(WindowEvent arg0) {

}

@Override
public void windowOpened(WindowEvent arg0) {

}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;
    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,
login,password);
    }
    catch (ClassNotFoundException cnfe)
    {
        JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    catch (SQLException sqle)
    {
        JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    return connection;
}

public void desconectar(Connection c)
{
    try
    {
        if(c!=null)
        {
            c.close();
        }
    }
}
```

```
        catch (SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
            JOptionPane.ERROR_MESSAGE);
        }
    }
    public ResultSet ejecutarSelect(String sentencia, Connection c)
    {
        try
        {
            Statement statement = c.createStatement();
            ResultSet rs= statement.executeQuery(sentencia);
            return rs;
        }
        catch(SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
            JOptionPane.ERROR_MESSAGE);
            return null;
        }
    }
}
```

ConFacList.java: En esta clase, encontramos un choice en el que se encuentran todas las facturas que aparecen en la Base de Datos y la utilizamos para seleccionarla para poder ver su contenido.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.BorderLayout;
import java.awt.Choice;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.Connection;
import java.sql.DriverManager;
```

```
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

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

public class ConFacList extends JFrame implements WindowListener,
ActionListener {

    /**
     *
     */
    private static final long serialVersionUID = 1L;
    Choice facturas = new Choice();
    JLabel lblConsultaFac = new JLabel("Seleccione Factura a
consultar:");
    JButton btnSeleccionar = new JButton("Seleccionar");
    JPanel pnl2 = new JPanel();
    int idReparacion;
    Font negrita = new Font("Arial", Font.BOLD, 14);
    public ConFacList()
    {
        this.setLayout(new FlowLayout());
        this.setLocationRelativeTo(null);
        this.setSize(500,150);
        this.setTitle("Consulta de Facturas");
        this.add(lblConsultaFac);
        this.add(facturas);
        pnl2.add(btnSeleccionar);
        this.add(pnl2, BorderLayout.SOUTH);
        btnSeleccionar.addActionListener(this);

        ResultSet selectFacturas = ejecutarSelect("SELECT
idFactura, DATE_FORMAT(FechaFactura, '%d/%m/%Y'), nombreCliente,
idReparacionFK FROM facturas, clientes where idClienteFK =
idCliente;", conectar("TallerJava","usuarioTaller","Studium2018;"));

        try {

            while (selectFacturas.next())
            {
```

```

        String
        fac=Integer.toString(selectFacturas.getInt("idFactura"));
        fac = fac + "-" +
        "
        "+selectFacturas.getString("DATE_FORMAT(FechaFactura, '%d/%m/%Y')")+
        ", Cliente:
        "+selectFacturas.getString("nombreCliente")+
        "
        Reparación:"+selectFacturas.getInt("idReparacionFK");
        facturas.add(fac);
    }
} catch (SQLException e) {

    JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
    JOptionPane.ERROR_MESSAGE);
}
desconectar(conectar("TallerJava","usuarioTaller"
,"Studium2018;"));

    this.addWindowListener(this);
    this.setVisible(true);
}

@Override
public void actionPerformed(ActionEvent ae) {

    if(btnSeleccionar.equals(ae.getSource())) {

        String[] array=
        facturas.getSelectedItem().toString().split("-");
        int idFactura = Integer.parseInt(array[0]);
        String[] array2=
        facturas.getSelectedItem().toString().split(", Reparación:");
        int idReparacion = Integer.parseInt(array2[1]);
        new ConLineaRepRec(idReparacion, idFactura);
        this.setVisible(false);
    }
}

@Override
public void windowActivated(WindowEvent e) {
    // TODO Auto-generated method stub
}

@Override

```

```
public void windowClosed(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowClosing(WindowEvent e) {
    this.setVisible(false);

}

@Override
public void windowDeactivated(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowDeiconified(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowIconified(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowOpened(WindowEvent e) {
    // TODO Auto-generated method stub

}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;

    try
```



```
        {
            Class.forName(driver);
            connection = DriverManager.getConnection(url,
login,password);
        }
        catch (ClassNotFoundException cnfe)
        {

            JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        }
        catch (SQLException sqle)
        {

            JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        }
        return connection;
    }
    public void desconectar(Connection c)
    {
        try
        {
            if(c!=null)
            {
                c.close();
            }
        }
        catch (SQLException e)
        {

            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        }
    }
    public ResultSet ejecutarSelect(String sentencia, Connection c)
    {
        try
        {
            Statement statement = c.createStatement();
            ResultSet rs= statement.executeQuery(sentencia);
            return rs;
        }
        catch(SQLException e)
        {

            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
```

```
JOptionPane.ERROR_MESSAGE);  
        return null;  
    }  
}  
}
```

ConLineaRepRec.java: En esta clase, encontramos los recambios que se han utilizado en una reparación a través de la factura que hemos seleccionado en la clase ConFacList.java.

```
package es.studium.PracticaSegundoTrimestre;  
  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.awt.event.WindowEvent;  
import java.awt.event.WindowListener;  
import java.io.FileNotFoundException;  
import java.io.FileOutputStream;  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.sql.Statement;  
import javax.swing.*;  
  
import com.itextpdf.text.Document;  
import com.itextpdf.text.DocumentException;  
import com.itextpdf.text.Font;  
import com.itextpdf.text.Paragraph;  
import com.itextpdf.text.pdf.PdfWriter;  
  
import java.awt.GridBagLayout;  
import java.awt.FileDialog;  
import java.awt.GridBagConstraints;  
  
public class ConLineaRepRec extends JFrame implements  
WindowListener, ActionListener  
{  
  
    /**  
     *    */  
}
```

```

    */
    private static final long serialVersionUID = 1L;
    JLabel lblTotal = new JLabel("Total");

    JTextField txtTotal = new JTextField(6);

    JPanel pnl1 = new JPanel();
    JPanel pnl2 = new JPanel();
    int FactSelec;
    JTextArea txtRecambiosFac = new JTextArea(6,35);

    double Total = 0;
    String recambios;
    double subTotal=0;
    double precio = 0;
    int idReparacion;
    double total = 0;
    String title;
    private final JPanel pnl3 = new JPanel();
    private final JButton btnAceptar = new JButton("Aceptar");
    private final JButton btnImprimirFactura = new
JButton("Imprimir Factura");

    Document documento = new Document();

    public ConLineaRepRec(int idRep,int idFac)
    {
        recambios="";
        idReparacion = idRep;
        FactSelec = idFac;
        title = "Factura nº: "+Integer.toString(FactSelec);
        this.setLocationRelativeTo(null);
        this.setSize(500, 332);
        this.setTitle(title);
        //Rellenando el TextArea

        ResultSet rsRec = ejecutarSelect("SELECT * FROM
incluyen,recambios where idRecambio = idRecambioFK and
idReparacionfk
="+idReparacion+";",conectar("TallerJava","usuarioTaller","Studium20
18;"));
        try {
            while (rsRec.next()) {
                recambios =recambios+"-
"+rsRec.getString("descripcionRecambio")+", Precio:
"+rsRec.getDouble("precioRecambio")+", Cantidad:
"+rsRec.getInt("cantidad")+", Subtotal:

```

```

"+(rsRec.getDouble("precioRecambio")*rsRec.getInt("cantidad"))+"\n";
        total =
total+(rsRec.getDouble("precioRecambio")*rsRec.getInt("cantidad"));
    }
    } catch (SQLException e) {
        // TODO Auto-generated catch block

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    this.setLayout(new GridBagLayout());

    pnl1.setBorder(BorderFactory.createTitledBorder(BorderFactory.c
reateEtchedBorder(), "Desglose"));
    txtRecambiosFac.setText(recambios);
    txtRecambiosFac.setEditable(false);
    pnl1.add(txtRecambiosFac);
    GridBagConstraints gbc_pnl1 = new GridBagConstraints();
    gbc_pnl1.gridx = 0;
    gbc_pnl1.gridy = 0;
    this.add(pnl1, gbc_pnl1);
    txtTotal.setEditable(false);

    pnl2.setBorder(BorderFactory.createTitledBorder(BorderFactory.c
reateEtchedBorder(), "Precio Total"));
    pnl2.add(lblTotal);
    txtTotal.setText(Double.toString(total));
    pnl2.add(txtTotal);
    GridBagConstraints gbc_pnl2 = new GridBagConstraints();
    gbc_pnl2.gridx = 0;
    gbc_pnl2.gridy = 1;
    this.add(pnl2, gbc_pnl2);

    GridBagConstraints gbc_pnl3 = new GridBagConstraints();
    gbc_pnl3.gridx = 0;
    gbc_pnl3.gridy = 2;
    this.add(pnl3, gbc_pnl3);

    pnl3.add(btnAceptar);
    btnAceptar.addActionListener(this);
    pnl3.add(btnImprimirFactura);
    btnImprimirFactura.addActionListener(this);
    this.setVisible(true);

}
@Override
public void actionPerformed(ActionEvent ae)

```

```

{
    // TODO Auto-generated method stub
    if(btnAceptar.equals(ae.getSource()))
    {
        this.setVisible(false);
    } else if(btnImprimirFactura.equals(ae.getSource()))
    {
        FileOutputStream ficheroPdf;
        try {
            FileDialog fd = new FileDialog(this,
"Seleccionar archivo", FileDialog.SAVE);
            fd.setFile("*.pdf");
            fd.setVisible(true);
            String filename =
fd.getDirectory()+fd.getFile();
            ficheroPdf = new FileOutputStream(filename);

            PdfWriter.getInstance(documento,ficheroPdf).setInitialLeading(2
0);

            documento.open();
            Paragraph titulo = new Paragraph("Factura
nº"+FactSelec);

            titulo.getFont().setStyle(Font.BOLD);
            titulo.getFont().setSize(15);
            documento.add(titulo);
            Paragraph contenido = new
Paragraph(txtRecambiosFac.getText());
            documento.add(contenido);
            Paragraph total = new Paragraph ("TOTAL:
"+txtTotal.getText());
            total.getFont().setStyle(Font.BOLD);
            documento.add(total);
            documento.close();
            JOptionPane.showMessageDialog(null,"Documento
pdf creado correctamente.", "Documento creado",
JOptionPane.INFORMATION_MESSAGE);

        } catch (FileNotFoundException e1) {
            // TODO Auto-generated catch block

            JOptionPane.showMessageDialog(null,e1.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        } catch (DocumentException e2) {
            // TODO Auto-generated catch block

```

```
JOptionPane.showMessageDialog(null,e2.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
}

@Override
public void windowActivated(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowClosed(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowClosing(WindowEvent e) {
    // TODO Auto-generated method stub
    this.setVisible(false);
}

@Override
public void windowDeactivated(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowDeiconified(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowIconified(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowOpened(WindowEvent e) {
    // TODO Auto-generated method stub

}
```

```
public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;

    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,
login,password);
    }
    catch (ClassNotFoundException cnfe)
    {
        JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    catch (SQLException sqle)
    {
        JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    return connection;
}

public void desconectar(Connection c)
{
    try
    {
        if(c!=null)
        {
            c.close();
        }
    }
    catch (SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
}
```

```
    }

    public ResultSet ejecutarSelect(String sentencia, Connection c)
    {

        try
        {
            Statement statement = c.createStatement();
            ResultSet rs= statement.executeQuery(sentencia);
            return rs;
        }
        catch(SQLException e)
        {

            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
            JOptionPane.ERROR_MESSAGE);
            return null;
        }

    }

}
```

ConRecList.java: En esta clase, encontramos un listado de los recambios que encontramos en nuestra base de datos, así como la posibilidad de exportarlos a PDF.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.BorderLayout;
import java.awt.FileDialog;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.sql.Connection;
import java.sql.DriverManager;
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 javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import com.itextpdf.text.Document;
import com.itextpdf.text.DocumentException;
import com.itextpdf.text.Font;
import com.itextpdf.text.Paragraph;
import com.itextpdf.text.pdf.PdfPTable;
import com.itextpdf.text.pdf.PdfWriter;

public class ConRecList extends JFrame implements WindowListener,
ActionListener {

    /**
     *
     */
    private static final long serialVersionUID = 1L;
    DefaultTableModel modelo = new DefaultTableModel();
    JTable tablaRecambios= new JTable(modelo);
    JButton btnAceptar = new JButton("Aceptar");
    JButton btnPDF = new JButton("Crear Pdf");
    JPanel pnl1 = new JPanel();
    ResultSet rs;

    Document documento = new Document();

    public ConRecList()
    {
        this.setLayout(new BorderLayout());
        this.setLocationRelativeTo(null);
        this.setSize(600,200);
        this.add(new
JScrollPane(tablaRecambios),BorderLayout.CENTER);

        modelo.addColumn("Nº Recambio");
        modelo.addColumn("Descripción");
        modelo.addColumn("Unidades");
        modelo.addColumn("Precio");

        rs = ejecutarSelect("SELECT * FROM recambios",
conectar("TallerJava","usuarioTaller","Studium2018;"));
        try {

```

```

        while (rs.next())
        {
            Object [] fila = new Object[4];

            for (int i=0;i<4;i++)
                fila[i] = rs.getObject(i+1);
            modelo.addRow(fila);
        }
    } catch (SQLException e) {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
    }
    desconectar(conectar("TallerJava","usuarioTaller"
    ,"Studium2018;"));
    tablaRecambios.setEnabled(false);
    pnl1.add(btnAceptar);
    pnl1.add(btnPDF);
    btnAceptar.addActionListener(this);
    btnPDF.addActionListener(this);
    this.add(pnl1, BorderLayout.SOUTH);
    this.addWindowListener(this);
    this.setVisible(true);
}

@Override
public void actionPerformed(ActionEvent ae)
{
    if(btnAceptar.equals(ae.getSource()))
    {
        this.setVisible(false);
    }
    else if (btnPDF.equals(ae.getSource()))
    {
        FileOutputStream ficheroPdf;
        try {
            FileDialog fd = new FileDialog(this,
            "Seleccionar archivo", FileDialog.SAVE);
            fd.setFile("*.pdf");
            fd.setVisible(true);
            String filename =
            fd.getDirectory()+fd.getFile();
            ficheroPdf = new FileOutputStream(filename);

            PdfWriter.getInstance(documento,ficheroPdf).setInitialLeading(2
            0);

```

```

        documento.open();
        PdfPTable tabla = new PdfPTable(4);

        Paragraph idRecambio = new Paragraph("Nº
Recambio");

        idRecambio.getFont().setStyle(Font.BOLD);
        idRecambio.getFont().setSize(15);
        tabla.addCell(idRecambio);

        Paragraph descripcionRecambio = new
Paragraph("Descripción del Recambio");

        descripcionRecambio.getFont().setStyle(Font.BOLD);
        descripcionRecambio.getFont().setSize(15);
        tabla.addCell(descripcionRecambio);

        Paragraph unidadesRecambio = new
Paragraph("Unidades del Recambio");

        unidadesRecambio.getFont().setStyle(Font.BOLD);
        unidadesRecambio.getFont().setSize(15);
        tabla.addCell(unidadesRecambio);

        Paragraph precioRecambio = new
Paragraph("Precio del Recambio");

        precioRecambio.getFont().setStyle(Font.BOLD);
        precioRecambio.getFont().setSize(15);
        tabla.addCell(precioRecambio);

        ResultSet Co = ejecutarSelect("SELECT * FROM
recambios;", conectar("TallerJava", "usuarioTaller"
, "Studium2018;"));
        try {
            while (Co.next())
            {
                for (int i=0; i<4; i++) {
                    if(i==0) {

                        tabla.addCell(Co.getString("idRecambio"));
                    } else if(i==1) {

                        tabla.addCell(Co.getString("descripcionRecambio"));
                    } else if(i==2) {

                        tabla.addCell(Co.getString("unidadesRecambio"));
                    } else if(i==3) {

```

```

        tabla.addCell(Co.getString("precioRecambio"));
    }
}
documento.add(tabla);
documento.close();

JOptionPane.showMessageDialog(null,"Documento pdf creado
correctamente.", "Documento creado",
JOptionPane.INFORMATION_MESSAGE);
} catch (SQLException e) {

    JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
}
} catch (FileNotFoundException e1) {
    // TODO Auto-generated catch block

    JOptionPane.showMessageDialog(null,e1.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
} catch (DocumentException e2) {
    // TODO Auto-generated catch block

    JOptionPane.showMessageDialog(null,e2.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
}

desconectar(conectar("TallerJava","usuarioTaller"
,"Studium2018;"));
}

@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}
@Override
public void windowClosing(WindowEvent arg0)
{
    this.setVisible(false);
}

@Override
public void windowDeactivated(WindowEvent arg0) {

```

```
    }  
    @Override  
    public void windowDeiconified(WindowEvent arg0) {  
  
    }  
    @Override  
    public void windowIconified(WindowEvent arg0) {  
  
    }  
    @Override  
    public void windowOpened(WindowEvent arg0) {  
  
    }  
  
    public Connection conectar(String baseDatos, String usuario,  
String clave)  
    {  
        String driver = "com.mysql.jdbc.Driver";  
        String url  
= "jdbc:mysql://localhost:3306/" + baseDatos + "?autoReconnect=true&useSSL=false";  
        String login = usuario;  
        String password = clave;  
        Connection connection = null;  
  
        try  
        {  
            Class.forName(driver);  
            connection = DriverManager.getConnection(url,  
login,password);  
        }  
        catch (ClassNotFoundException cnfe)  
        {  
  
            JOptionPane.showMessageDialog(null, cnfe.getMessage(), "Error",  
JOptionPane.ERROR_MESSAGE);  
        }  
        catch (SQLException sqle)  
        {  
  
            JOptionPane.showMessageDialog(null, sqle.getMessage(), "Error",  
JOptionPane.ERROR_MESSAGE);  
        }  
    }  
}
```

```
        return connection;
    }
    public void desconectar(Connection c)
    {
        try
        {
            if(c!=null)
            {
                c.close();
            }
        }
        catch (SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        }
    }

    public ResultSet ejecutarSelect(String sentencia, Connection c)
    {
        try
        {
            Statement statement = c.createStatement();
            ResultSet rs= statement.executeQuery(sentencia);
            return rs;
        }
        catch(SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
            return null;
        }
    }
}
```

ConRepList.java: En esta clase, encontramos un listado de todas las reparaciones que tenemos en nuestra base de datos. También tenemos la posibilidad de imprimirlas en un PDF.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.BorderLayout;
import java.awt.FileDialog;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import com.itextpdf.text.Document;
import com.itextpdf.text.DocumentException;
import com.itextpdf.text.Font;
import com.itextpdf.text.Paragraph;
import com.itextpdf.text.pdf.PdfPTable;
import com.itextpdf.text.pdf.PdfWriter;

public class ConRepList extends JFrame implements WindowListener,
ActionListener {
    /**
     *
     */
    private static final long serialVersionUID = 1L;
    DefaultTableModel modelo = new DefaultTableModel();
    JTable tablaReparaciones= new JTable(modelo);
    JButton btnAceptar = new JButton("Aceptar");
    JButton btnPDF = new JButton("Crear Pdf");
    JPanel pnl2 = new JPanel();
    ResultSet rs;
    int reparado;

    Document documento = new Document();

    public ConRepList()
    {
```

```

        this.setLayout(new BorderLayout());
        this.setLocationRelativeTo(null);
        this.setSize(600,200);
        this.setTitle("Consulta de Reparaciones");

        this.add(new
JScrollPane(tablaReparaciones),BorderLayout.CENTER);
        pnl2.add(btnAceptar);
        pnl2.add(btnPDF);
        this.add(pnl2, BorderLayout.SOUTH);
        btnAceptar.addActionListener(this);
        btnPDF.addActionListener(this);

        modelo.addColumn("Nº Reparacion");
        modelo.addColumn("Avería");
        modelo.addColumn("Fecha Entrada");
        modelo.addColumn("Fecha de Salida");
        modelo.addColumn("Reparado");

        rs = ejecutarSelect("select idReparacion, averia,
date_format(fechaEntrada, '%d/%m/%y'), date_format(fechaSalida,
'%d/%m/%y'), Reparado from reparaciones;",
conectar("TallerJava","usuarioTaller","Studium2018;"));
        try {

            while (rs.next())
            {
                Object [] fila = new Object[5];

                for (int i=0;i<5;i++)
                    if(i==0) {
                        fila[i]=
rs.getString("idReparacion");

                    }else if (i==1)
                    {
                        fila[i] = rs.getString("averia");

                    }else if (i==2)
                    {
                        fila[i] =
rs.getString("date_format(fechaEntrada, '%d/%m/%y')");

                    }else if (i==3)
                    {
                        fila[i] =
rs.getString("date_format(fechaSalida, '%d/%m/%y')");

```



```

        } else if(i==4)
        {
            reparado =
Integer.parseInt(rs.getString("reparado"));
            if(reparado==1) {
                fila[i] = "Reparado";
            } else {
                fila[i] = "No reparado";
            }
        }

        modelo.addRow(fila);
    }
} catch (SQLException e) {

    JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    desconectar(conectar("TallerJava","usuarioTaller"
,"Stodium2018;"));
    tablaReparaciones.setEnabled(false);

    this.addWindowListener(this);
    this.setVisible(true);
}
@Override
public void actionPerformed(ActionEvent ae)
{
    if(btnAceptar.equals(ae.getSource())) {
        this.setVisible(false);
    } else if (btnPDF.equals(ae.getSource())) {
        FileOutputStream ficheroPdf;
        try {
            FileDialog fd = new FileDialog(this,
"Seleccionar archivo", FileDialog.SAVE);
            fd.setFile("*.pdf");
            fd.setVisible(true);
            String filename =
fd.getDirectory()+fd.getFile();
            ficheroPdf = new FileOutputStream(filename);

            PdfWriter.getInstance(documento,ficheroPdf).setInitialLeading(2
0);

            documento.open();

```

```

PdfPTable tabla = new PdfPTable(5);

Paragraph idReparacion = new Paragraph("Nº
Reparación");

idReparacion.getFont().setStyle(Font.BOLD);
idReparacion.getFont().setSize(15);
tabla.addCell(idReparacion);

Paragraph averia = new Paragraph("Avería");
averia.getFont().setStyle(Font.BOLD);
averia.getFont().setSize(15);
tabla.addCell(averia);

Paragraph fechaEntrada = new Paragraph("Fecha
de Entrada");

fechaEntrada.getFont().setStyle(Font.BOLD);
fechaEntrada.getFont().setSize(15);
tabla.addCell(fechaEntrada);

Paragraph fechaSalida = new Paragraph("Fecha
de Salida");

fechaSalida.getFont().setStyle(Font.BOLD);
fechaSalida.getFont().setSize(15);
tabla.addCell(fechaSalida);

Paragraph reparado = new
Paragraph("Reparado");
reparado.getFont().setStyle(Font.BOLD);
reparado.getFont().setSize(15);
tabla.addCell(reparado);

ResultSet Co = ejecutarSelect("SELECT * FROM
reparaciones;", conectar("TallerJava", "usuarioTaller"
, "Studium2018;"));

try {
    while (Co.next())
    {
        for (int i=0; i<5; i++) {
            if(i==0) {

tabla.addCell(Co.getString("idReparacion"));
            } else if(i==1) {

tabla.addCell(Co.getString("averia"));
            } else if(i==2) {

tabla.addCell(Co.getString("fechaEntrada"));

```

```

        } else if(i==3) {

            tabla.addCell(Co.getString("fechaSalida"));
            }else if(i==4) {

                if(Co.getString("reparado").equals("0")) {
                    tabla.addCell("No
reparado");
                } else if
(Co.getString("reparado").equals("1"))
                {

                    tabla.addCell("Reparado");
                }
            }
        }
        documento.add(tabla);
        documento.close();

        JOptionPane.showMessageDialog(null,"Documento pdf creado
correctamente.", "Documento creado",
JOptionPane.INFORMATION_MESSAGE);
    } catch (SQLException e) {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    } catch (FileNotFoundException e1) {

        JOptionPane.showMessageDialog(null,e1.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    } catch (DocumentException e2) {

        JOptionPane.showMessageDialog(null,e2.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }

    desconectar(conectar("TallerJava","usuarioTaller"
,"Stodium2018;"));
    }
}
@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}
@Override

```

```
public void windowClosing(WindowEvent arg0)
{
    this.setVisible(false);
}
@Override
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;

    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,
login,password);
    }
    catch (ClassNotFoundException cnfe)
    {
        JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    catch (SQLException sqle)
    {
        JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    return connection;
}
public void desconectar(Connection c)
{

```

```
        try
        {
            if(c!=null)
            {
                c.close();
            }
        }
        catch (SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        }
    }
    public ResultSet ejecutarSelect(String sentencia, Connection c)
    {
        try
        {
            Statement statement = c.createStatement();
            ResultSet rs= statement.executeQuery(sentencia);
            return rs;
        }
        catch(SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
            return null;
        }
    }
}
```

EICliList.java: En esta clase, encontramos un choice en el que elegiremos el cliente que queramos y tendremos la posibilidad de eliminarlo de la base de datos.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.*;
import java.awt.event.*;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
```

```
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Calendar;

import javax.swing.*.*;

public class ElCliList extends JFrame implements WindowListener,
ActionListener{
    /**
     *
     */
    private static final long serialVersionUID = 1L;

    Choice ListaCli = new Choice();

    JButton btnBorrar = new JButton("Eliminar Cliente");
    int idClienteBorrar;

    JPanel pnl1 = new JPanel();
    JPanel pnl2 = new JPanel();

    int idCliBorrar;
    ResultSet con;

    String user = "";

    public ElCliList(String usuario) {
        user = usuario;
        this.setTitle("Eliminar clientes");
        this.setLayout(new GridLayout(2,1));
        this.setLocationRelativeTo(null);
        this.setSize(400,200);
        ListaCli.add("Seleccionar cliente a eliminar");
        con = ejecutarSelect("SELECT * FROM clientes",
conectar("TallerJava", "usuarioTaller", "Stodium2018;"));
        try {
            while(con.next())
            {
                String
clientes=Integer.toString(con.getInt("idCliente"));
                clientes = clientes+".-
"+con.getString("nombreCliente");
                ListaCli.add(clientes);
            }
        } catch (SQLException e) {
```

```

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    desconectar(conectar("TallerJava","usuarioTaller"
,"Stodium2018;"));

    pnl1.add(ListaCli);
    pnl2.add(btnBorrar);

    this.add(pnl1);
    this.add(pnl2);
    this.addWindowListener(this);
    btnBorrar.addActionListener(this);
    this.setVisible(true);
}

@Override
public void actionPerformed(ActionEvent ae) {
    // TODO Auto-generated method stub
    if(btnBorrar.equals(ae.getSource())) {
        int seleccion = JOptionPane.showOptionDialog(
null,"¿Desea eliminar cliente?", "Eliminar
cliente",JOptionPane.YES_NO_CANCEL_OPTION,JOptionPane.QUESTION_MESSA
GE,null,new Object[] { "Eliminar", "Cancelar"}, "Cancelar");
        if (seleccion == 0){
            try {
                String[] array=
ListaCli.getSelectedItem().toString().split("-");
                idCliBorrar = Integer.parseInt(array[0]);
            } catch (NumberFormatException Nf) {

                JOptionPane.showMessageDialog(null,"Introduzca cliente
válido","Error de cliente", JOptionPane.ERROR_MESSAGE);
            }
            String sentencia="DELETE FROM clientes where
idCliente =" +idCliBorrar+";";
            ejecutarIDA(sentencia, conectar("TallerJava",
"usuarioTaller", "Stodium2018;"));
            JOptionPane.showMessageDialog(null,"El cliente
"+idCliBorrar+" ha sido eliminado","Cliente eliminado",
JOptionPane.INFORMATION_MESSAGE);
            Calendar horaFecha = Calendar.getInstance();
            int hora,minutos,dia,mes,anyo;
            hora = horaFecha.get(Calendar.HOURLY_OF_DAY);
            minutos = horaFecha.get(Calendar.MINUTE);
            dia = horaFecha.get(Calendar.DAY_OF_MONTH);

```

```
        mes = horaFecha.get(Calendar.MONTH)+1;
        anyo = horaFecha.get(Calendar.YEAR);
        try {
            FileWriter fw = new
FileWriter("movimientos.log", true);
            BufferedWriter bw = new
BufferedWriter(fw);

            PrintWriter outPut = new PrintWriter(bw);

            outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ]["+hora+": "+minutos+" ]
"+"["+user+"]"+"["+sentencia+"]"+"\\n");
            outPut.close();
            bw.close();
            fw.close();
        } catch(IOException ioe) {
            System.out.print("Error");
        }

        this.setVisible(false);
    } else if(seleccion == 1) {

    }

}

@Override
public void windowActivated(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowClosed(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowClosing(WindowEvent e) {
    // TODO Auto-generated method stub
    this.setVisible(false);
}
```



```
@Override
public void windowDeactivated(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowDeiconified(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowIconified(WindowEvent e) {
    // TODO Auto-generated method stub

}

@Override
public void windowOpened(WindowEvent e) {
    // TODO Auto-generated method stub

}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;

    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,
login,password);
    }
}
```

```
        catch (ClassNotFoundException cnfe)
        {
            JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        }
        catch (SQLException sqle)
        {
            JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        }
        return connection;
    }
    public void desconectar(Connection c)
    {
        try
        {
            if(c!=null)
            {
                c.close();
            }
        }
        catch (SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        }
    }
    public ResultSet ejecutarSelect(String sentencia, Connection c)
    {
        try
        {
            Statement statement = c.createStatement();
            ResultSet rs= statement.executeQuery(sentencia);
            return rs;
        }
        catch(SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
            return null;
        }
    }
}
```

```

    }
    public void ejecutarIDA(String sentencia, Connection c)
    {

        try
        {
            Statement statement = c.createStatement();
            statement.executeUpdate(sentencia);
        }
        catch(SQLException e)
        {

            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
            JOptionPane.ERROR_MESSAGE);
        }

    }
}

```

ElRecList.java: En esta clase, encontramos un choice con todos los recambios y nos dará la posibilidad de eliminar el recambio que queramos.

```

package es.studium.PracticaSegundoTrimestre;

import java.awt.*;
import java.awt.event.*;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Calendar;

import javax.swing.*;

public class ElRecList extends JFrame implements WindowListener,
ActionListener{
    /**
     *
     */
    private static final long serialVersionUID = 1L;

```

```

Choice ListaRec = new Choice();

JButton btnBorrar = new JButton("Eliminar recambio");
int idRecambioBorrar;

JPanel pnl1 = new JPanel();
JPanel pnl2 = new JPanel();

int idRecBorrar;
ResultSet con;
String user="";

public ElRecList(String usuario)
{
    user = usuario;
    this.setTitle("Eliminar recambios");
    this.setLayout(new GridLayout(2,1));
    this.setLocationRelativeTo(null);
    this.setSize(400,200);
    ListaRec.add("Seleccionar recambio a eliminar");
    con = ejecutarSelect("SELECT * FROM recambios",
conectar("TallerJava", "usuarioTaller", "Stodium2018;"));
    try {
        while(con.next())
        {
            String
recambios=Integer.toString(con.getInt("idRecambio"));
            recambios = recambios+".-
"+con.getString("descripcionRecambio");
            ListaRec.add(recambios);
        }
    } catch (SQLException e) {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    desconectar(conectar("TallerJava","usuarioTaller"
,"Stodium2018;"));

    pnl1.add(ListaRec);
    pnl2.add(btnBorrar);

    this.add(pnl1);
    this.add(pnl2);
    this.addWindowListener(this);
    btnBorrar.addActionListener(this);

```

```

        this.setVisible(true);
    }

    @Override
    public void actionPerformed(ActionEvent ae)
    {
        // TODO Auto-generated method stub
        if(btnBorrar.equals(ae.getSource())) {
            int seleccion = JOptionPane.showOptionDialog(
null,"¿Desea eliminar recambio?", "Eliminar
recambio",JOptionPane.YES_NO_CANCEL_OPTION,JOptionPane.QUESTION_MESS
AGE,null,new Object[] { "Eliminar", "Cancelar"}, "Cancelar");
            if (seleccion == 0){
                try {
                    String[] array=
ListaRec.getSelectedItem().toString().split(".-");
                    idRecBorrar = Integer.parseInt(array[0]);
                } catch (NumberFormatException Nf) {

                    JOptionPane.showMessageDialog(null,"Introduzca recambio
válido","Error de recambio", JOptionPane.ERROR_MESSAGE);
                }
                String sentencia = "DELETE FROM recambios
where idRecambio =" + idRecBorrar + ";";
                ejecutarIDA(sentencia, conectar("TallerJava",
"usuarioTaller", "Studium2018;"));
                JOptionPane.showMessageDialog(null,"El
recambio " + idRecBorrar + " ha sido eliminado","Recambio eliminado",
JOptionPane.INFORMATION_MESSAGE);
                Calendar horaFecha = Calendar.getInstance();
                int hora,minutos,dia,mes,anyo;
                hora = horaFecha.get(Calendar.HOUR_OF_DAY);
                minutos = horaFecha.get(Calendar.MINUTE);
                dia = horaFecha.get(Calendar.DAY_OF_MONTH);
                mes = horaFecha.get(Calendar.MONTH)+1;
                anyo = horaFecha.get(Calendar.YEAR);
                try {
                    FileWriter fw = new
FileWriter("movimientos.log", true);
                    BufferedWriter bw = new
BufferedWriter(fw);

                    PrintWriter outPut = new PrintWriter(bw);

                    outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ]["+hora+": "+minutos+" ]
"+"["+user+"]"+"["+sentencia+"]"+"\\n");
                    outPut.close();
                    bw.close();
                }
            }
        }
    }

```

```

        fw.close();
    } catch(IOException ioe) {
        System.out.print("Error");
    }
    this.setVisible(false);
} else if(seleccion == 1) {

}

}

@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}
@Override
public void windowClosing(WindowEvent arg0)
{
    this.setVisible(false);
}

@Override
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;

    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,
login,password);

```

```
    }
    catch (ClassNotFoundException cnfe)
    {

        JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
    }
    catch (SQLException sqle)
    {

        JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
    }
    return connection;
}
public void desconectar(Connection c)
{
    try
    {
        if(c!=null)
        {
            c.close();
        }
    }
    catch (SQLException e)
    {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
    }
}
public ResultSet ejecutarSelect(String sentencia, Connection c)
{
    try
    {
        Statement statement = c.createStatement();
        ResultSet rs= statement.executeQuery(sentencia);
        return rs;
    }
    catch(SQLException e)
    {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
        return null;
    }
}
```

```
    }  
    public void ejecutarIDA(String sentencia, Connection c)  
    {  
        try  
        {  
            Statement statement = c.createStatement();  
            statement.executeUpdate(sentencia);  
        }  
        catch(SQLException e)  
        {  
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",  
JOptionPane.ERROR_MESSAGE);  
        }  
    }  
}
```

ElRepList.java: En esta clase, encontramos un choice con todas las reparaciones y nos dará la posibilidad de eliminar la reparación que queramos.

```
package es.studium.PracticaSegundoTrimestre;  
  
import java.awt.*;  
import java.awt.event.*;  
import java.io.BufferedWriter;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.sql.Statement;  
import java.util.Calendar;  
  
import javax.swing.*;  
  
public class ElRepList extends JFrame implements WindowListener,  
ActionListener  
{  
    /**
```



```

    *
    */
    private static final long serialVersionUID = 1L;

    Choice ListaRep = new Choice();

    JButton btnBorrar = new JButton("Eliminar Reparación");
    int idReparacionBorrar;

    JPanel pnl1 = new JPanel();
    JPanel pnl2 = new JPanel();

    int idRepBorrar;
    ResultSet con;
    String user="";

    public ElRepList(String usuario)
    {
        user = usuario;
        this.setTitle("Eliminar reparaciones");
        this.setLayout(new GridLayout(2,1));
        this.setLocationRelativeTo(null);
        this.setSize(500,200);
        ListaRep.add("Seleccionar reparación a eliminar");
        con = ejecutarSelect("SELECT * FROM reparaciones",
conectar("TallerJava", "usuarioTaller", "Stodium2018;"));
        try {
            while(con.next())
            {
                String
reparaciones=Integer.toString(con.getInt("idReparacion"));
                reparaciones = reparaciones+".-
"+con.getString("Averia")+", Fecha Entrada:
"+con.getString("FechaEntrada")+", Fecha Salida:
"+con.getString("FechaSalida");
                ListaRep.add(reparaciones);
            }
        } catch (SQLException e) {

            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        }
        desconectar(conectar("TallerJava", "usuarioTaller"
,"Stodium2018;"));

        pnl1.add(ListaRep);
        pnl2.add(btnBorrar);
    }

```

```

        this.add(pnl1);
        this.add(pnl2);
        this.addWindowListener(this);
        btnBorrar.addActionListener(this);
        this.setVisible(true);
    }
    @Override
    public void actionPerformed(ActionEvent ae)
    {
        // TODO Auto-generated method stub
        if(btnBorrar.equals(ae.getSource())) {
            int seleccion = JOptionPane.showOptionDialog(
null,"¿Desea eliminar reparación?", "Eliminar
reparación", JOptionPane.YES_NO_CANCEL_OPTION, JOptionPane.QUESTION_ME
SSAGE, null, new Object[] { "Eliminar", "Cancelar"}, "Cancelar");
            if (seleccion == 0){
                try {
                    String[] array=
ListaRep.getSelectedItem().toString().split("-");
                    idRepBorrar = Integer.parseInt(array[0]);
                } catch (NumberFormatException Nf) {

                    JOptionPane.showMessageDialog(null, "Introduzca reparación
válida", "Error de reparación", JOptionPane.ERROR_MESSAGE);
                }
                String sentencia = "DELETE FROM reparaciones
where idReparacion =" + idRepBorrar + ";";
                ejecutarIDA(sentencia, conectar("TallerJava",
"usuarioTaller", "Stodium2018;"));
                JOptionPane.showMessageDialog(null, "La
reparación " + idRepBorrar + " ha sido eliminada", "Reparación
eliminada", JOptionPane.INFORMATION_MESSAGE);
                Calendar horaFecha = Calendar.getInstance();
                int hora, minutos, dia, mes, anyo;
                hora = horaFecha.get(Calendar.HOUR_OF_DAY);
                minutos = horaFecha.get(Calendar.MINUTE);
                dia = horaFecha.get(Calendar.DAY_OF_MONTH);
                mes = horaFecha.get(Calendar.MONTH)+1;
                anyo = horaFecha.get(Calendar.YEAR);
                try {
                    FileWriter fw = new
FileWriter("movimientos.log", true);
                    BufferedWriter bw = new
BufferedWriter(fw);

                    PrintWriter outPut = new PrintWriter(bw);

```

```

        outPut.print("[dia+"/"+mes+"/"+anyo+"["+hora+": "+minutos+"]
"+"["+user+"]"+"["+sentencia+"]"+"\\n");
        outPut.close();
        bw.close();
        fw.close();
    } catch(IOException ioe) {
        System.out.print("Error");
    }
    this.setVisible(false);
} else if(seleccion == 1) {

}

}

}

@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}
@Override
public void windowClosing(WindowEvent arg0)
{
    this.setVisible(false);
}
@Override
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSS
L=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;

    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,

```

```
login,password);
    }
    catch (ClassNotFoundException cnfe)
    {

        JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    catch (SQLException sqle)
    {

        JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    return connection;
}
public void desconectar(Connection c)
{
    try
    {
        if(c!=null)
        {
            c.close();
        }
    }
    catch (SQLException e)
    {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
}
public ResultSet ejecutarSelect(String sentencia, Connection c)
{
    try
    {
        Statement statement = c.createStatement();
        ResultSet rs= statement.executeQuery(sentencia);
        return rs;
    }
    catch(SQLException e)
    {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        return null;
    }
}
```

```

    }

    }

    public void ejecutarIDA(String sentencia, Connection c)
    {

        try
        {
            Statement statement = c.createStatement();
            statement.executeUpdate(sentencia);
        }
        catch(SQLException e)
        {

            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
            JOptionPane.ERROR_MESSAGE);
        }

    }
}

```

LineaRepRec.java: En esta clase, nos permitirá añadir los recambios utilizados en un recambio. A medida que vayamos metiendo recambios se nos irá aumentando el total del precio de la reparación.

```

package es.studium.PracticaSegundoTrimestre;

import java.awt.Choice;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.*;

public class LineaRepRec extends JFrame implements WindowListener,
ActionListener
{

    /**

```

```

    *
    */
    private static final long serialVersionUID = 1L;
    JLabel lblTituloFactura = new JLabel("Factura nº ");
    JLabel lblFactura = new JLabel("");
    JLabel lblRecambio = new JLabel("Recambio:");
    JLabel lblCantidad = new JLabel("Cantidad: ");
    JLabel lblPrecio = new JLabel("Precio");
    JLabel lblTotal = new JLabel("Total");

    Choice recambios = new Choice();
    JTextField txtCantidad = new JTextField(3);
    JTextField txtTotal = new JTextField(6);
    JButton btnAgregar = new JButton("Agregar");
    JButton btnCancelar = new JButton("Cancelar");
    JButton btnAceptar = new JButton("Aceptar");

    JPanel pnl1 = new JPanel();
    JPanel pnl2 = new JPanel();
    JPanel pnl3 = new JPanel();
    JPanel pnl4 = new JPanel();
    JPanel pnl5 = new JPanel();
    int FactSelec;
    JTextArea txtRecambiosFac = new JTextArea(6,35);

    double total = 0;
    String recambioSeleccionado = "";
    double subTotal=0;
    double precio = 0;
    int idReparacion;

    public LineaRepRec(int idRep, int idFac) {
        idReparacion = idRep;
        FactSelec = idFac;
        lblFactura.setText(Integer.toString(FactSelec));
        this.setLayout(new GridLayout(5,1));
        this.setLocationRelativeTo(null);
        this.setSize(500, 400);
        pnl1.add(lblTituloFactura);
        pnl1.add(lblFactura);
        this.add(pnl1);

        pnl2.setBorder(BorderFactory.createTitledBorder(BorderFactory.createEtchedBorder(), "Detalles"));
        pnl2.add(lblRecambio);
        txtTotal.setEditable(false);
        recambios.add("Elige un articulo...");
    }

```

```

//Selec de recambios para choice

ResultSet rsRec = ejecutarSelect("SELECT * FROM
recambios;", conectar("TallerJava", "usuarioTaller", "Studium2018;"));
try {
    while(rsRec.next())
    {
        String rec =
Integer.toString(rsRec.getInt("idRecambio"));
        rec = rec + "-" +
rsRec.getString("descripcionRecambio");
        recambios.add(rec);
    }
} catch (SQLException e) {

    JOptionPane.showMessageDialog(null, e.getMessage(), "Error",
JOptionPane.ERROR_MESSAGE);
}
desconectar(conectar("TallerJava", "usuarioTaller"
, "Studium2018;"));

pn12.add(recambios);
pn12.add(lblCantidad);
pn12.add(txtCantidad);
pn12.add(btnAgregar);
this.add(pn12);
pn13.add(txtRecambiosFac);
this.add(pn13);

pn14.setBorder(BorderFactory.createTitledBorder(BorderFactory.c
reateEtchedBorder(), "Precio Total"));
pn14.add(lblTotal);
pn14.add(txtTotal);
pn15.add(btnAceptar);
pn15.add(btnCancelar);
btnAgregar.addActionListener(this);
btnAceptar.addActionListener(this);
btnCancelar.addActionListener(this);

this.add(pn14);
this.add(pn15);
this.setVisible(true);
}

@Override
public void actionPerformed(ActionEvent ae)

```

```

{

    //Botón Agregar
    if(btnAgregar.equals(ae.getSource()))
    {
        try
        {
            int Cantidad =
Integer.parseInt(txtCantidad.getText());
            String[] arrayCod=
recambios.getSelectedItem().toString().split("-");
            ResultSet recSelect = ejecutarSelect("SELECT
precioRecambio from recambios where idRecambio
="+arrayCod[0]+";",conectar("TallerJava", "usuarioTaller",
"Stodium2018;"));
            try {
                recSelect.next();
                precio =
recSelect.getDouble("precioRecambio");
                subTotal = precio*Cantidad;
                total = total +
recSelect.getDouble("precioRecambio")*Cantidad;
                recambioSeleccionado =
recambioSeleccionado+" "+recambios.getSelectedItem().toString()+",
Precio: "+Double.toString(precio)+", Cantidad:"+Cantidad+",
Subtotal:"+Double.toString(subTotal)+"\n";

                txtRecambiosFac.setText(recambioSeleccionado);
                txtTotal.setText(Double.toString(total));
                desconectar(conectar("TallerJava",
"usuarioTaller", "Stodium2018;"));
                ejecutarIDA("INSERT INTO incluyen valores
(null,"+idReparacion+", "+arrayCod[0]+", "+Cantidad+");",
conectar("TallerJava", "usuarioTaller", "Stodium2018;"));

            } catch (SQLException e){

                JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
            }

            desconectar(conectar("TallerJava","usuarioTaller"
,"Stodium2018;"));
        } catch (NumberFormatException nf) {
            JOptionPane.showMessageDialog(null,"Introduzca
cantidad o artículo válidos","Error", JOptionPane.ERROR_MESSAGE);
        }
    }
}

```



```

    }
    //Botón Aceptar
    if(btnAceptar.equals(ae.getSource()))
    {
        JOptionPane.showMessageDialog(null,"Factura creada
correctamente con un total de "+txtTotal.getText(),"Factura creada",
JOptionPane.INFORMATION_MESSAGE);
    }
    //Botón Cancelar
    else if(btnCancelar.equals(ae.getSource()))
    {
        int seleccion = JOptionPane.showOptionDialog(
null,"¿Desea salir de la línea de
factura?", "Salir",JOptionPane.YES_NO_CANCEL_OPTION,JOptionPane.QUEST
ION_MESSAGE,null,new Object[] { "Salir", "Cancelar"},"Cancelar");
        if (seleccion == 0){
            this.setVisible(false);
        }
    }
}
@Override
public void windowActivated(WindowEvent arg0) {
    // TODO Auto-generated method stub

}
@Override
public void windowClosed(WindowEvent e) {
    // TODO Auto-generated method stub

}
@Override
public void windowClosing(WindowEvent e) {
    // TODO Auto-generated method stub
    int seleccion = JOptionPane.showOptionDialog(
null,"¿Desea salir de la línea de
factura?", "Salir",JOptionPane.YES_NO_CANCEL_OPTION,JOptionPane.QUEST
ION_MESSAGE,null,new Object[] { "Salir", "Cancelar"},"Cancelar");
    if (seleccion == 0){
        this.setVisible(false);
    }
}
@Override
public void windowDeactivated(WindowEvent e) {
    // TODO Auto-generated method stub

}
@Override

```

```
public void windowDeiconified(WindowEvent e) {
    // TODO Auto-generated method stub

}
@Override
public void windowIconified(WindowEvent e) {
    // TODO Auto-generated method stub

}
@Override
public void windowOpened(WindowEvent e) {
    // TODO Auto-generated method stub

}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSS
L=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;

    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,
login,password);
    }
    catch (ClassNotFoundException cnfe)
    {
        JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    catch (SQLException sqle)
    {
        JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    return connection;
}
```

```
public void desconectar(Connection c)
{
    try
    {
        if(c!=null)
        {
            c.close();
        }
    }
    catch (SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
}

public ResultSet ejecutarSelect(String sentencia, Connection c)
{
    try
    {
        Statement statement = c.createStatement();
        ResultSet rs= statement.executeQuery(sentencia);
        return rs;
    }
    catch(SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        return null;
    }
}

public void ejecutarIDA(String sentencia, Connection c)
{
    try
    {
        Statement statement = c.createStatement();
        statement.executeUpdate(sentencia);
    }
    catch(SQLException e)
    {

```

```
JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }

}

}
```

Login.java: Esta es la clase que se encarga de ejecutar el programa y es un login en el que nos pedirá que iniciemos sesión con un usuario del programa.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Calendar;
import javax.swing.*;

public class Login implements WindowListener, ActionListener {

    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/TallerJava?autoReconnect=true&useSSL=f
alse";
    String login = "usuarioTaller";
    String password = "Studium2018;";
    Connection connection = null;
    Statement statement = null;
    ResultSet rs = null;

    JFrame ventanaLogin = new JFrame ("Iniciar Sesión");
```

```
JLabel lblIniciar = new JLabel("Iniciar Sesión");

JLabel lblCorreo = new JLabel("Correo Electrónico: ");
JLabel lblPass = new JLabel("Contraseña: ");
JLabel lblCorreoOlvidado = new JLabel("Introduzca su Correo Electrónico: ");
JLabel lblCorreoOlvidadoOK = new JLabel("Compruebe su correo electrónico");

JPasswordField txtPass = new JPasswordField(20);
JTextField txtCorreo = new JTextField(20);
JTextField txtCorreoOlvidado = new JTextField(20);

JButton btnIniciar = new JButton ("Iniciar Sesión");
JButton btnLimpiar = new JButton ("Limpiar");
JButton btnOlvidePass = new JButton ("Olvidé la contraseña");
JButton btnOlvidePassSiguiente = new JButton ("Siguiente");

JDialog dlgOlvidada = new JDialog(ventanaLogin, "Contraseña olvidada");
JDialog dlgOlvidadaOK = new JDialog(ventanaLogin, "Contraseña olvidada ");

JPanel pnlPanel = new JPanel();
JPanel pnlPanel2 = new JPanel();
JPanel pnlPanel3 = new JPanel();
JPanel pnlPanel4 = new JPanel();
JPanel pnlPanel5 = new JPanel();

public Login()
{
    ventanaLogin.setLocationRelativeTo(null);
    ventanaLogin.setSize(400,200);
    ventanaLogin.setLayout(new GridLayout(5,1));

    dlgOlvidada.setLocationRelativeTo(null);
    dlgOlvidada.setSize(300, 150);
    dlgOlvidada.setLayout(new FlowLayout());
    dlgOlvidada.setVisible(false);

    dlgOlvidada.add(lblCorreoOlvidado);
    dlgOlvidada.add(txtCorreoOlvidado);
    dlgOlvidada.add(btnOlvidePassSiguiente);

    dlgOlvidadaOK.setLocationRelativeTo(null);
    dlgOlvidadaOK.setSize(300, 150);
    dlgOlvidadaOK.setLayout(new FlowLayout());
```

```
        dlgOlvidadaOK.setVisible(false);

        dlgOlvidadaOK.add(lblCorreoOlvidadoOK);

        pnlPanel1.setLayout(new FlowLayout());
        pnlPanel2.setLayout(new FlowLayout());
        pnlPanel3.setLayout(new FlowLayout());
        pnlPanel4.setLayout(new FlowLayout());
        pnlPanel5.setLayout(new FlowLayout());

        pnlPanel1.add(lblIniciar);
        ventanaLogin.add(pnlPanel1);

        pnlPanel2.add(lblCorreo);
        pnlPanel2.add(txtCorreo);
        ventanaLogin.add(pnlPanel2);

        pnlPanel3.add(lblPass);
        pnlPanel3.add(txtPass);
        ventanaLogin.add(pnlPanel3);

        pnlPanel4.add(btnIniciar);
        pnlPanel4.add(btnLimpiar);
        ventanaLogin.add(pnlPanel4);

        pnlPanel5.add(btnOlvidePass);
        ventanaLogin.add(pnlPanel5);

        btnIniciar.addActionListener(this);
        btnLimpiar.addActionListener(this);
        btnOlvidePass.addActionListener(this);
        btnOlvidePassSiguiente.addActionListener(this);

        ventanaLogin.addWindowListener(this);
        dlgOlvidada.addWindowListener(this);
        dlgOlvidadaOK.addWindowListener(this);
        ventanaLogin.setVisible(true);
    }

    public static void main(String[] args)
    {
        new Login();
    }

    @Override
    public void actionPerformed(ActionEvent ae)
    {
```

```

        if(btnIniciar.equals(ae.getSource())) {
            String usuario = txtCorreo.getText();

            if(txtCorreo.getText().equals("administrador@studium.es"))
            {
                try
                {
                    Class.forName(driver);
                    String sentencia = "SELECT * FROM
usuarios where correoUsuario ='"+txtCorreo.getText()+"' AND
claveUsuario = MD5('"+txtPass.getText()+"')";
                    connection =
DriverManager.getConnection(url, login,password);
                    statement =connection.createStatement();
                    rs=statement.executeQuery(sentencia);
                    if(rs.next())
                    {

                        JOptionPane.showMessageDialog(null,"Login Correcto","Login
correcto", JOptionPane.INFORMATION_MESSAGE);
                        Calendar horaFecha =
Calendar.getInstance();

                        int hora,minutos,dia,mes,anyo;
                        hora =
horaFecha.get(Calendar.HOUR_OF_DAY);
                        minutos =
horaFecha.get(Calendar.MINUTE);
                        dia =
horaFecha.get(Calendar.DAY_OF_MONTH);
                        mes =
horaFecha.get(Calendar.MONTH)+1;
                        anyo = horaFecha.get(Calendar.YEAR);
                        try {
                            FileWriter fw = new
FileWriter("movimientos.log", true);
                            BufferedWriter bw = new
BufferedWriter(fw);
                            PrintWriter outPut = new
PrintWriter(bw);

                            outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ]["+hora+": "+minutos+" ]
"+"["+usuario+"]"+"\\n");

                            outPut.close();
                            bw.close();
                            fw.close();
                        } catch(IOException ioe) {

```

```
        System.out.print("Error");
    }
    ventanaLogin.setVisible(false);

    new MenuPrincipal(usuario);

    } else {
        JOptionPane.showMessageDialog(null,
"Usuario o contraseña erroneos", "Error al iniciar sesión",
JOptionPane.ERROR_MESSAGE);
    }
}
catch (ClassNotFoundException cnfe)
{
    JOptionPane.showMessageDialog(null, cnfe.getMessage(), "Error",
JOptionPane.ERROR_MESSAGE);

}
catch (SQLException sqle)
{
    JOptionPane.showMessageDialog(null,
sqle.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
}
finally
{
    try
    {
        if(connection!=null)
        {
            connection.close();
        }
    }
    catch (SQLException e)
    {
        JOptionPane.showMessageDialog(null,
e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
    }
}

}

else {

    try
    {
        Class.forName(driver);
        String sentencia = "SELECT * FROM
```



```

usuarios where correoUsuario = '"+txtCorreo.getText()+"' AND
claveUsuario = MD5('"+txtPass.getText()+"');";
        connection =
DriverManager.getConnection(url, login,password);
        statement =connection.createStatement();
        rs=statement.executeQuery(sentencia);
        if(rs.next())
        {

            JOptionPane.showMessageDialog(null,"Login Correcto","Login
correcto", JOptionPane.INFORMATION_MESSAGE);
            Calendar horaFecha =
Calendar.getInstance();
            int hora,minutos,dia,mes,anyo;
            hora =
horaFecha.get(Calendar.HOUR_OF_DAY);
            minutos =
horaFecha.get(Calendar.MINUTE);
            dia =
horaFecha.get(Calendar.DAY_OF_MONTH);
            mes =
horaFecha.get(Calendar.MONTH)+1;
            anyo = horaFecha.get(Calendar.YEAR);
            try {
                FileWriter fw = new
FileWriter("movimientos.log", true);
                BufferedWriter bw = new
BufferedWriter(fw);
                PrintWriter outPut = new
PrintWriter(bw);

                outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ]["+hora+": "+minutos+" ]
"+"["+usuario+"]"+"\\n");

                outPut.close();
                bw.close();
                fw.close();
            } catch(IOException ioe) {
                System.out.print("Error");
            }
            ventanaLogin.setVisible(false);
            new MenuPrincipalUsuario(usuario);

        } else {
            JOptionPane.showMessageDialog(null,
"Usuario o contraseña erroneos","Error al iniciar sesión",
JOptionPane.ERROR_MESSAGE);
        }
    }

```

```

        }
        catch (ClassNotFoundException cnfe)
        {
            JOptionPane.showMessageDialog(null,
"Error",cnfe.getMessage(), JOptionPane.ERROR_MESSAGE);

        }
        catch (SQLException sqle)
        {
            JOptionPane.showMessageDialog(null,
"Error",sqle.getMessage(), JOptionPane.ERROR_MESSAGE);
        }
        finally
        {
            try
            {
                if(connection!=null)
                {
                    connection.close();
                }
            }
            catch (SQLException e)
            {
                JOptionPane.showMessageDialog(null,
"Error",e.getMessage(), JOptionPane.ERROR_MESSAGE);
            }
        }
    }

} else if (btnLimpiar.equals(ae.getSource())) {
    txtCorreo.selectAll();
    txtCorreo.setText("");
    txtPass.selectAll();
    txtPass.setText("");

} else if (btnOlvidePass.equals(ae.getSource())) {
    dlgOlvidada.setVisible(true);
}
if(btnOlvidePassSiguiente.equals(ae.getSource())) {
    dlgOlvidadaOK.setVisible(true);
}
}
@Override
public void windowActivated(WindowEvent we) {}

@Override
public void windowClosed(WindowEvent arg0) {}

```

```
@Override
public void windowClosing(WindowEvent arg0)
{
    if(ventanaLogin.isActive()) {
        ventanaLogin.setVisible(false);
    }else {
        System.exit(0);
    }
    if(dlgOlvidada.isActive()) {
        dlgOlvidada.setVisible(false);
    }
    if(dlgOlvidadaOK.isActive()) {
        dlgOlvidada.setVisible(false);
        dlgOlvidadaOK.setVisible(false);
    }
}
@Override
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}
}
```

MenuPrincipal.java: En esta clase, encontramos el menú principal del usuario administrador.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.CardLayout;
import java.awt.List;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Calendar;

import javax.swing.JButton;
```

```
import javax.swing.JFrame;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JPanel;
import java.awt.GridBagLayout;
import java.awt.GridBagConstraints;
import javax.swing.JLabel;
import javax.swing.ImageIcon;
import javax.swing.border.TitledBorder;

public class MenuPrincipal implements WindowListener,
ActionListener{

    String user = new String("");

    JFrame ventana = new JFrame ("Taller de Recambios");
    List lista = new List();
    JPanel pnlLista = new JPanel();
    JPanel pnlCard = new JPanel();
    JPanel pnlClientes = new JPanel();
    JPanel pnlRecambios = new JPanel();
    JPanel pnlReparaciones = new JPanel();
    JPanel pnlFacturas = new JPanel();

    JMenuBar barraMenu = new JMenuBar();
    JMenu menuOtros = new JMenu("Opciones");
    JMenu menuAyuda = new JMenu("Ayuda");
    JMenuItem mniOtrosAyuda = new JMenuItem("Ayuda");
    JMenuItem mniOtrosSalir = new JMenuItem("Cerrar Sesión");

    final static String Clientes = "Clientes";
    final static String Recambios = "Recambios";
    final static String Reparaciones = "Reparaciones";
    final static String Facturas = "Facturas";

    JButton btnAddCli = new JButton("Añadir Clientes");
    JButton btnModCli = new JButton("Modificar Clientes");
    JButton btnElCli = new JButton("Eliminar Clientes");
    JButton btnConCli = new JButton("Consultar Clientes");

    JButton btnAddRec = new JButton("Añadir Recambios");
    JButton btnModRec = new JButton("Modificar Recambios");
    JButton btnElRec = new JButton("Eliminar Recambios");
    JButton btnConRec = new JButton("Consultar Recambios");

    JButton btnAddRep = new JButton("Añadir Reparaciones");
```

```

JButton btnModRep = new JButton("Modificar Reparaciones");
JButton btnElRep = new JButton("Eliminar Reparaciones");
JButton btnConRep = new JButton("Consultar Reparaciones");

JButton btnAddFac = new JButton("Añadir Facturas");
JButton btnConFac = new JButton("Consultar Facturas");
private final JPanel pnlImg = new JPanel();
private final JLabel label = new JLabel("");

public MenuPrincipal(String usuario) {
    user = usuario;

    ventana.setSize(870,350);
    ventana.setLocationRelativeTo(null);
    ventana.setJMenuBar(barraMenu);
    menuAyuda.add(mniOtrosAyuda);
    mniOtrosAyuda.addActionListener(this);
    // Añadimos un separador
    menuOtros.addSeparator();
    menuOtros.add(mniOtrosSalir);
    mniOtrosSalir.addActionListener(this);
    barraMenu.add(menuOtros);
    barraMenu.add(menuAyuda);
    GridBagLayout gridBagLayout = new GridBagLayout();
    ventana.setLayout(gridBagLayout);
    pnlCard.setBorder(new TitledBorder(null, "Opciones",
TitledBorder.LEADING, TitledBorder.TOP, null, null));
    pnlCard.setLayout(new CardLayout() );
    pnlClientes.setBorder(null);

    pnlClientes.add(btnAddCli);
    pnlClientes.add(btnModCli);
    pnlClientes.add(btnElCli);
    pnlClientes.add(btnConCli);

    btnAddCli.addActionListener(this);
    btnModCli.addActionListener(this);
    btnElCli.addActionListener(this);
    btnConCli.addActionListener(this);

    pnlRecambios.add(btnAddRec);
    pnlRecambios.add(btnModRec);
    pnlRecambios.add(btnElRec);
    pnlRecambios.add(btnConRec);

    btnAddRec.addActionListener(this);
    btnModRec.addActionListener(this);

```

```

        btnElRec.addActionListener(this);
        btnConRec.addActionListener(this);

        pnlReparaciones.add(btnAddRep);
        pnlReparaciones.add(btnModRep);
        pnlReparaciones.add(btnElRep);
        pnlReparaciones.add(btnConRep);

        btnAddRep.addActionListener(this);
        btnModRep.addActionListener(this);
        btnElRep.addActionListener(this);
        btnConRep.addActionListener(this);

        pnlFacturas.add(btnAddFac);
        pnlFacturas.add(btnConFac);

        btnAddFac.addActionListener(this);
        btnConFac.addActionListener(this);
        GridBagConstraints gbc_pnlLista = new
GridBagConstraints();
        gbc_pnlLista.gridheight = 2;
        gbc_pnlLista.gridx = 0;
        gbc_pnlLista.gridy = 0;
        ventana.add(pnlLista, gbc_pnlLista);
        GridBagConstraints gbc_Lista = new GridBagConstraints();
        gbc_Lista.gridx = 1;
        gbc_Lista.gridy = 0;
        ventana.add(Lista, gbc_Lista);

        Lista.add(Clientes);
        Lista.add(Recambios);
        Lista.add(Reparaciones);
        Lista.add(Facturas);
        Lista.addActionListener(this);

        pnlCard.add(Clientes , pnlClientes);
        pnlCard.add(Recambios , pnlRecambios);
        pnlCard.add(Reparaciones , pnlReparaciones);
        pnlCard.add(Facturas , pnlFacturas);
        GridBagConstraints gbc_pnlCard = new
GridBagConstraints();
        gbc_pnlCard.gridx = 2;
        gbc_pnlCard.gridy = 0;
        ventana.add(pnlCard, gbc_pnlCard);

        GridBagConstraints gbc_pnlImg = new GridBagConstraints();
        gbc_pnlImg.gridx = 2;

```

```

gbc_pnlImg.gridy = 1;
ventana.add(pnlImg, gbc_pnlImg);
label.setIcon(new ImageIcon("imagenes/logo.png"));

pnlImg.add(label);
ventana.addWindowListener(this);
ventana.setVisible(true);
}

@Override
public void actionPerformed(ActionEvent ae) {

    //OPCIONES LISTA
    if(Clientes.equals(Lista.getSelectedItem()))
    {
        pnlCard.add(Clientes , pnlClientes);
        pnlCard.add(Recambios , pnlRecambios);
        pnlCard.add(Reparaciones , pnlReparaciones);
        pnlCard.add(Facturas , pnlFacturas);
    } else if(Recambios.equals(Lista.getSelectedItem()))
    {
        pnlCard.add(Recambios , pnlRecambios);
        pnlCard.add(Clientes , pnlClientes);
        pnlCard.add(Reparaciones , pnlReparaciones);
        pnlCard.add(Facturas , pnlFacturas);
    } else if(Reparaciones.equals(Lista.getSelectedItem()))
    {
        pnlCard.add(Reparaciones , pnlReparaciones);
        pnlCard.add(Clientes , pnlClientes);
        pnlCard.add(Recambios , pnlRecambios);
        pnlCard.add(Facturas , pnlFacturas);
    } else if(Facturas.equals(Lista.getSelectedItem()))
    {
        pnlCard.add(Facturas , pnlFacturas);
        pnlCard.add(Reparaciones , pnlReparaciones);
        pnlCard.add(Clientes , pnlClientes);
        pnlCard.add(Recambios , pnlRecambios);
    }

    //BOTONES DE CLIENTES
    if(btnAddCli.equals(ae.getSource())) {
        new AddCli(user);
    }
    else if(btnModCli.equals(ae.getSource())) {
        new ModCliList(user);
    }
    else if(btnElCli.equals(ae.getSource())) {

```

```

        new ElCliList(user);
    }
    else if (btnConCli.equals(ae.getSource())) {
        new ConCliList();
        Calendar horaFecha = Calendar.getInstance();
        int hora,minutos,dia,mes,anyo;
        hora = horaFecha.get(Calendar.HOUR_OF_DAY);
        minutos = horaFecha.get(Calendar.MINUTE);
        dia = horaFecha.get(Calendar.DAY_OF_MONTH);
        mes = horaFecha.get(Calendar.MONTH)+1;
        anyo = horaFecha.get(Calendar.YEAR);
        try {
            FileWriter fw = new
FileWriter("movimientos.log", true);
            BufferedWriter bw = new BufferedWriter(fw);
            PrintWriter outPut = new PrintWriter(bw);

            outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ]["+hora+": "+minutos+" ]
"+"["+user+"]"+"["+ "SELECT * FROM CLIENTES"+" ]"+"\\n");
            outPut.close();
            bw.close();
            fw.close();
        } catch (IOException ioe) {
            System.out.print("Error");
        }
    }
    //BOTONES DE RECAMBIOS
    if(btnAddRec.equals(ae.getSource())) {
        new AddRec(user);
    } else if(btnModRec.equals(ae.getSource())) {
        new ModRecList(user);
    }
    else if(btnElRec.equals(ae.getSource())) {
        new ElRecList(user);
    }
    else if (btnConRec.equals(ae.getSource())) {
        new ConRecList();
        Calendar horaFecha = Calendar.getInstance();
        int hora,minutos,dia,mes,anyo;
        hora = horaFecha.get(Calendar.HOUR_OF_DAY);
        minutos = horaFecha.get(Calendar.MINUTE);
        dia = horaFecha.get(Calendar.DAY_OF_MONTH);
        mes = horaFecha.get(Calendar.MONTH)+1;
        anyo = horaFecha.get(Calendar.YEAR);
        try {
            FileWriter fw = new
FileWriter("movimientos.log", true);

```



```

        BufferedWriter bw = new BufferedWriter(fw);
        PrintWriter outPut = new PrintWriter(bw);

        outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ][ "+hora+": "+minutos+" ]"
        +"[ "+user+" ]"+"[ "+"SELECT * FROM RECAMBIOS"+" ]"+"\\n");
        outPut.close();
        bw.close();
        fw.close();
    } catch(IOException ioe) {
        System.out.print("Error");
    }
}
//BOTONES DE REPARACIONES
if(btnAddRep.equals(ae.getSource())) {
    new AddRep(user);
} else if(btnModRep.equals(ae.getSource())) {
    new ModRepList(user);
} else if(btnElRep.equals(ae.getSource())) {
    new ElRepList(user);
} else if (btnConRep.equals(ae.getSource())) {
    new ConRepList();
    Calendar horaFecha = Calendar.getInstance();
    int hora,minutos,dia,mes,anyo;
    hora = horaFecha.get(Calendar.HOUR_OF_DAY);
    minutos = horaFecha.get(Calendar.MINUTE);
    dia = horaFecha.get(Calendar.DAY_OF_MONTH);
    mes = horaFecha.get(Calendar.MONTH)+1;
    anyo = horaFecha.get(Calendar.YEAR);
    try {
        FileWriter fw = new
FileWriter("movimientos.log", true);
        BufferedWriter bw = new BufferedWriter(fw);
        PrintWriter outPut = new PrintWriter(bw);

        outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ][ "+hora+": "+minutos+" ]"
        +"[ "+user+" ]"+"[ "+"SELECT * FROM REPARACIONES"+" ]"+"\\n");
        outPut.close();
        bw.close();
        fw.close();
    } catch(IOException ioe) {
        System.out.print("Error");
    }
}
//BOTONES DE REPARACIONES
if(btnAddFac.equals(ae.getSource())) {
    new AddFac(user);
} else if (btnConFac.equals(ae.getSource())) {

```

```

        new ConFacList();
        Calendar horaFecha = Calendar.getInstance();
        int hora,minutos,dia,mes,anyo;
        hora = horaFecha.get(Calendar.HOUR_OF_DAY);
        minutos = horaFecha.get(Calendar.MINUTE);
        dia = horaFecha.get(Calendar.DAY_OF_MONTH);
        mes = horaFecha.get(Calendar.MONTH)+1;
        anyo = horaFecha.get(Calendar.YEAR);
        try {
            FileWriter fw = new
FileWriter("movimientos.log", true);
            BufferedWriter bw = new BufferedWriter(fw);
            PrintWriter outPut = new PrintWriter(bw);

            outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ] [ "+hora+": "+minutos+" ]
"+" [ "+user+" ] "+" [ "+"SELECT * FROM FACTURAS"+" ] "+" \n");
            outPut.close();
            bw.close();
            fw.close();
        } catch (IOException ioe) {
            System.out.print("Error");
        }
    }

    if(mniOtrosSalir.equals(ae.getSource())) {
        ventana.setVisible(false);
        new Login();
    }
    if(mniOtrosAyuda.equals(ae.getSource())) {
        new Ayuda();
    } else if(mniOtrosSalir.equals(ae.getSource())) {
        ventana.setVisible(false);
    }
}

@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}
@Override
public void windowClosing(WindowEvent arg0)
{
    System.exit(0);
}
@Override
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}

```

```
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}

}
```

MenuPrincipalUsuario.java: En esta clase, encontramos el menú principal del usuario básico.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.CardLayout;
import java.awt.List;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Calendar;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JPanel;
import java.awt.GridBagLayout;
import java.awt.GridBagConstraints;
import javax.swing.JLabel;
import javax.swing.ImageIcon;
import javax.swing.border.TitledBorder;

public class MenuPrincipalUsuario implements WindowListener,
ActionListener{

    String user = new String("");

    JFrame ventana = new JFrame ("Taller de Recambios");
    List lista = new List();
    JPanel pnlLista = new JPanel();
    JPanel pnlCard = new JPanel();
```

```
JPanel pnlClientes = new JPanel();
JPanel pnlRecambios = new JPanel();
JPanel pnlReparaciones = new JPanel();
JPanel pnlFacturas = new JPanel();

JMenuBar barraMenu = new JMenuBar();
JMenu menuOtros = new JMenu("Opciones");
JMenu menuAyuda = new JMenu("Ayuda");
JMenuItem mniOtrosAyuda = new JMenuItem("Ayuda");
JMenuItem mniOtrosSalir = new JMenuItem("Cerrar Sesión");

final static String Clientes = "Clientes";
final static String Recambios = "Recambios";
final static String Reparaciones = "Reparaciones";
final static String Facturas = "Facturas";

JButton btnAddCli = new JButton("Añadir Clientes");
JButton btnConCli = new JButton("Consultar Clientes");

JButton btnAddRec = new JButton("Añadir Recambios");
JButton btnConRec = new JButton("Consultar Recambios");

JButton btnAddRep = new JButton("Añadir Reparaciones");
JButton btnConRep = new JButton("Consultar Reparaciones");

JButton btnAddFac = new JButton("Añadir Facturas");
JButton btnConFac = new JButton("Consultar Facturas");
private final JPanel pnlImg = new JPanel();
private final JLabel label = new JLabel("");

public MenuPrincipalUsuario(String usuario) {
    user = usuario;

    ventana.setSize(500,350);
    ventana.setLocationRelativeTo(null);
    ventana.setJMenuBar(barraMenu);
    menuAyuda.add(mniOtrosAyuda);
    mniOtrosAyuda.addActionListener(this);
    // Añadimos un separador
    menuOtros.addSeparator();
    menuOtros.add(mniOtrosSalir);
    mniOtrosSalir.addActionListener(this);
    barraMenu.add(menuOtros);
    barraMenu.add(menuAyuda);
    GridBagLayout gridBagLayout = new GridBagLayout();
    ventana.setLayout(gridBagLayout);
    pnlCard.setBorder(new TitledBorder(null, "Opciones",
```

```

TitledBorder.LEADING, TitledBorder.TOP, null, null));
    pnlCard.setLayout(new CardLayout());
    pnlClientes.setBorder(null);

    pnlClientes.add(btnAddCli);
    pnlClientes.add(btnConCli);

    btnAddCli.addActionListener(this);
    btnConCli.addActionListener(this);

    pnlRecambios.add(btnAddRec);
    pnlRecambios.add(btnConRec);

    btnAddRec.addActionListener(this);
    btnConRec.addActionListener(this);

    pnlReparaciones.add(btnAddRep);
    pnlReparaciones.add(btnConRep);

    btnAddRep.addActionListener(this);
    btnConRep.addActionListener(this);

    pnlFacturas.add(btnAddFac);
    pnlFacturas.add(btnConFac);

    btnAddFac.addActionListener(this);
    btnConFac.addActionListener(this);
    GridBagConstraints gbc_pnlLista = new
GridBagConstraints();
    gbc_pnlLista.gridheight = 2;
    gbc_pnlLista.gridx = 0;
    gbc_pnlLista.gridy = 0;
    ventana.add(pnlLista, gbc_pnlLista);
    GridBagConstraints gbc_Lista = new GridBagConstraints();
    gbc_Lista.gridx = 1;
    gbc_Lista.gridy = 0;
    ventana.add(Lista, gbc_Lista);
    Lista.add(Clientes);
    Lista.add(Recambios);
    Lista.add(Reparaciones);
    Lista.add(Facturas);
    Lista.addActionListener(this);

    pnlCard.add(Clientes , pnlClientes);
    pnlCard.add(Recambios , pnlRecambios);
    pnlCard.add(Reparaciones , pnlReparaciones);
    pnlCard.add(Facturas , pnlFacturas);

```

```

        GridBagConstraints gbc_pnlCard = new
GridBagConstraints();
        gbc_pnlCard.gridx = 2;
        gbc_pnlCard.gridy = 0;
        ventana.add(pnlCard, gbc_pnlCard);

        GridBagConstraints gbc_pnlImg = new GridBagConstraints();
        gbc_pnlImg.gridx = 2;
        gbc_pnlImg.gridy = 1;
        ventana.add(pnlImg, gbc_pnlImg);
        label.setIcon(new ImageIcon("imagenes/logo.png"));

        pnlImg.add(label);
        ventana.addWindowListener(this);
        ventana.setVisible(true);
    }
    @Override
    public void actionPerformed(ActionEvent ae) {

        //OPCIONES LISTA
        if(Clientes.equals(Lista.getSelectedItem()))
        {
            pnlCard.add(Clientes , pnlClientes);
            pnlCard.add(Recambios , pnlRecambios);
            pnlCard.add(Reparaciones , pnlReparaciones);
            pnlCard.add(Facturas , pnlFacturas);
        } else if(Recambios.equals(Lista.getSelectedItem()))
        {
            pnlCard.add(Recambios , pnlRecambios);
            pnlCard.add(Clientes , pnlClientes);
            pnlCard.add(Reparaciones , pnlReparaciones);
            pnlCard.add(Facturas , pnlFacturas);
        } else if(Reparaciones.equals(Lista.getSelectedItem()))
        {
            pnlCard.add(Reparaciones , pnlReparaciones);
            pnlCard.add(Clientes , pnlClientes);
            pnlCard.add(Recambios , pnlRecambios);
            pnlCard.add(Facturas , pnlFacturas);
        } else if(Facturas.equals(Lista.getSelectedItem()))
        {
            pnlCard.add(Facturas , pnlFacturas);
            pnlCard.add(Reparaciones , pnlReparaciones);
            pnlCard.add(Clientes , pnlClientes);
            pnlCard.add(Recambios , pnlRecambios);
        }

        //BOTONES DE CLIENTES
    }

```

```

        if(btnAddCli.equals(ae.getSource())) {
            new AddCli(user);
        }
        else if (btnConCli.equals(ae.getSource())) {
            new ConCliList();
            Calendar horaFecha = Calendar.getInstance();
            int hora,minutos,dia,mes,anyo;
            hora = horaFecha.get(Calendar.HOUR_OF_DAY);
            minutos = horaFecha.get(Calendar.MINUTE);
            dia = horaFecha.get(Calendar.DAY_OF_MONTH);
            mes = horaFecha.get(Calendar.MONTH)+1;
            anyo = horaFecha.get(Calendar.YEAR);
            try {
                FileWriter fw = new
FileWriter("movimientos.log", true);
                BufferedWriter bw = new BufferedWriter(fw);
                PrintWriter outPut = new PrintWriter(bw);

                outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ]["+hora+": "+minutos+" ]
"+"["+user+"]"+"["+ "SELECT * FROM CLIENTES"+" ]"+"\\n");
                outPut.close();
                bw.close();
                fw.close();
            } catch(IOException ioe) {
                System.out.print("Error");
            }
        }
        //BOTONES DE RECAMBIOS
        if(btnAddRec.equals(ae.getSource())) {
            new AddRec(user);
        }
        else if (btnConRec.equals(ae.getSource())) {
            new ConRecList();
            Calendar horaFecha = Calendar.getInstance();
            int hora,minutos,dia,mes,anyo;
            hora = horaFecha.get(Calendar.HOUR_OF_DAY);
            minutos = horaFecha.get(Calendar.MINUTE);
            dia = horaFecha.get(Calendar.DAY_OF_MONTH);
            mes = horaFecha.get(Calendar.MONTH)+1;
            anyo = horaFecha.get(Calendar.YEAR);
            try {
                FileWriter fw = new
FileWriter("movimientos.log", true);
                BufferedWriter bw = new BufferedWriter(fw);
                PrintWriter outPut = new PrintWriter(bw);

                outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ]["+hora+": "+minutos+" ]

```

```

"+"["+user+"]"+"["+SELECT * FROM RECAMBIOS+"]+"\n");
        outPut.close();
        bw.close();
        fw.close();
    } catch(IOException ioe) {
        System.out.print("Error");
    }
}
//BOTONES DE REPARACIONES
if(btnAddRep.equals(ae.getSource())) {
    new AddRep(user);
} else if (btnConRep.equals(ae.getSource())) {
    new ConRepList();
    Calendar horaFecha = Calendar.getInstance();
    int hora,minutos,dia,mes,anyo;
    hora = horaFecha.get(Calendar.HOUR_OF_DAY);
    minutos = horaFecha.get(Calendar.MINUTE);
    dia = horaFecha.get(Calendar.DAY_OF_MONTH);
    mes = horaFecha.get(Calendar.MONTH)+1;
    anyo = horaFecha.get(Calendar.YEAR);
    try {
        FileWriter fw = new
FileWriter("movimientos.log", true);
        BufferedWriter bw = new BufferedWriter(fw);
        PrintWriter outPut = new PrintWriter(bw);

        outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ]["+hora+": "+minutos+" ]
"+"["+user+"]"+"["+SELECT * FROM REPARACIONES+"]+"\n");
        outPut.close();
        bw.close();
        fw.close();
    } catch(IOException ioe) {
        System.out.print("Error");
    }
}
//BOTONES DE REPARACIONES
if(btnAddFac.equals(ae.getSource())) {
    new AddFac(user);
} else if (btnConFac.equals(ae.getSource())) {
    new ConFacList();
    Calendar horaFecha = Calendar.getInstance();
    int hora,minutos,dia,mes,anyo;
    hora = horaFecha.get(Calendar.HOUR_OF_DAY);
    minutos = horaFecha.get(Calendar.MINUTE);
    dia = horaFecha.get(Calendar.DAY_OF_MONTH);
    mes = horaFecha.get(Calendar.MONTH)+1;
    anyo = horaFecha.get(Calendar.YEAR);

```



```

        try {
            FileWriter fw = new
FileWriter("movimientos.log", true);
            BufferedWriter bw = new BufferedWriter(fw);
            PrintWriter outPut = new PrintWriter(bw);

            outPut.print("[dia+ "/" +mes+ "/" +anyo+ "]" +[hora+ ":" +minutos+ "]"
"+"["+user+"]"+"["+ "SELECT * FROM FACTURAS"+" "]"+"\n");
            outPut.close();
            bw.close();
            fw.close();
        } catch(IOException ioe) {
            System.out.print("Error");
        }
    }

    if(mniOtrosSalir.equals(ae.getSource())) {
        ventana.setVisible(false);
        new Login();
    }
    if(mniOtrosAyuda.equals(ae.getSource())) {
        new Ayuda();
    } else if(mniOtrosSalir.equals(ae.getSource())) {
        ventana.setVisible(false);
    }
}

@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}
@Override
public void windowClosing(WindowEvent arg0)
{
    System.exit(0);
}
@Override
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}
}

```

ModCli.java: En esta clase, podremos modificar a un cliente que hemos elegido anteriormente usando la clase ModCliList.java.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Calendar;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class ModCli implements WindowListener, ActionListener{
    JFrame ventanaModCli = new JFrame ("Modificar cliente:");
    JLabel lblNombreCli = new JLabel ("Nombre:");
    JLabel lblDireccionCli = new JLabel ("Dirección:");
    JLabel lblTelefonoCli = new JLabel ("Teléfono:");

    JTextField txtNombreCli = new JTextField(15);
    JTextField txtDireccionCli = new JTextField(15);
    JTextField txtTelefonoCli = new JTextField(15);

    JButton btnModificar = new JButton("Modificar Cliente");
    JButton btnLimpiar = new JButton("Limpiar");

    JPanel pnlPanel = new JPanel();
    JPanel pnlPanel2 = new JPanel();
    JPanel pnlPanel3 = new JPanel();
    JPanel pnlPanel4 = new JPanel();
    int idCli = 0;
```

```
String user ="";

public ModCli(int id, String usuario)
{
    user = usuario;
    idCli = id;
    ResultSet rs = ejecutarSelect("SELECT * FROM clientes
where idCliente
="+id+";",conectar("TallerJava","usuarioTaller","Studium2018;"));
    try {
        rs.next();
        txtNombreCli.selectAll();
        txtNombreCli.setText(rs.getString("nombreCliente"));
        txtDireccionCli.selectAll();

        txtDireccionCli.setText(rs.getString("DireccionCliente"));
        txtTelefonoCli.selectAll();

        txtTelefonoCli.setText(Integer.toString(rs.getInt("telefonoClien
te"))));
    } catch (SQLException e) {

        JOptionPane.showMessageDialog(null,"Error",e.getMessage(),
JOptionPane.ERROR_MESSAGE);
    }

    desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

    ventanaModCli.setLayout(new GridLayout(4,2));
    ventanaModCli.setLocationRelativeTo(null);
    ventanaModCli.setSize(400,300);

    pnlPanel.setLayout(new FlowLayout());
    pnlPanel2.setLayout(new FlowLayout());
    pnlPanel3.setLayout(new FlowLayout());
    pnlPanel4.setLayout(new FlowLayout());

    pnlPanel.add(lblNombreCli);
    pnlPanel.add(txtNombreCli);
    ventanaModCli.add(pnlPanel);

    pnlPanel2.add(lblDireccionCli);
    pnlPanel2.add(txtDireccionCli);
    ventanaModCli.add(pnlPanel2);

    pnlPanel3.add(lblTelefonoCli);
```

```

        pnlPanel3.add(txtTelefonoCli);
        ventanaModCli.add(pnlPanel3);

        pnlPanel4.add(btnModificar);
        btnModificar.addActionListener(this);
        pnlPanel4.add(btnLimpiar);
        btnLimpiar.addActionListener(this);
        ventanaModCli.add(pnlPanel4);

        ventanaModCli.addWindowListener(this);
        ventanaModCli.setVisible(true);
    }

    @Override
    public void actionPerformed(ActionEvent ae)
    {
        if (btnModificar.equals(ae.getSource())) {
            if(txtNombreCli.getText().equals("")) {
                JOptionPane.showMessageDialog(null,"Error,
Nombre de Cliente Vacío","Nombre vacío", JOptionPane.ERROR_MESSAGE);
            } else {
                String sentencia = "UPDATE clientes SET
nombreCliente = '"+txtNombreCli.getText()+"', direccionCliente =
'"+txtDireccionCli.getText()+"', telefonoCliente
='"+txtTelefonoCli.getText()+" WHERE idCliente ="+idCli+"";

                ejecutarIDA(sentencia,conectar("TallerJava","usuarioTaller","St
udium2018;"));

                desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

                Calendar horaFecha = Calendar.getInstance();
                int hora,minutos,dia,mes,anyo;
                hora = horaFecha.get(Calendar.HOUR_OF_DAY);
                minutos = horaFecha.get(Calendar.MINUTE);
                dia = horaFecha.get(Calendar.DAY_OF_MONTH);
                mes = horaFecha.get(Calendar.MONTH)+1;
                anyo = horaFecha.get(Calendar.YEAR);
                try {
                    FileWriter fw = new
FileWriter("movimientos.log", true);
                    BufferedWriter bw = new
BufferedWriter(fw);

```

```

        PrintWriter outPut = new PrintWriter(bw);

        outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ]["+hora+": "+minutos+" ]"
        "+ "[ "+user+" ]"+"["+sentencia+" ]"+"\\n");
        outPut.close();
        bw.close();
        fw.close();
    } catch (IOException ioe) {
        System.out.print("Error");
    }

    }

    } else if (btnLimpiar.equals(ae.getSource())) {
        txtNombreCli.selectAll();
        txtNombreCli.setText("");
        txtDireccionCli.selectAll();
        txtDireccionCli.setText("");
        txtTelefonoCli.selectAll();
        txtTelefonoCli.setText("");
    }

}

@Override
public void windowActivated(WindowEvent arg0) {}

@Override
public void windowClosed(WindowEvent arg0) {}

@Override
public void windowClosing(WindowEvent arg0) {

    if(ventanaModCli.isActive()) {
        ventanaModCli.setVisible(false);
    }else {
        //System.exit(0);
    }

}

@Override
public void windowDeactivated(WindowEvent arg0) {}

@Override
public void windowDeiconified(WindowEvent arg0) {}

@Override

```

```
public void windowIconified(WindowEvent arg0) {}

@Override
public void windowOpened(WindowEvent arg0) {}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;

    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,
login,password);
    }
    catch (ClassNotFoundException cnfe)
    {
        JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    catch (SQLException sqle)
    {
        JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    return connection;
}

public void desconectar(Connection c)
{
    try
    {
        if(c!=null)
        {
            c.close();
        }
    }
    catch (SQLException e)
```

```
{  
    JOptionPane.showMessageDialog(null,e.getMessage(),"Error",  
JOptionPane.ERROR_MESSAGE);  
}  
  
public ResultSet ejecutarSelect(String sentencia, Connection c)  
{  
    try  
    {  
        Statement statement = c.createStatement();  
        ResultSet rs= statement.executeQuery(sentencia);  
        return rs;  
    }  
    catch(SQLException e)  
    {  
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",  
JOptionPane.ERROR_MESSAGE);  
        return null;  
    }  
}  
  
public void ejecutarIDA(String sentencia, Connection c)  
{  
    try  
    {  
        Statement statement = c.createStatement();  
        statement.executeUpdate(sentencia);  
        JOptionPane.showMessageDialog(null,"Cliente  
modificado","Cliente modificado con éxito",  
JOptionPane.INFORMATION_MESSAGE);  
    }  
    catch(SQLException e)  
    {  
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",  
JOptionPane.ERROR_MESSAGE);  
    }  
}  
}
```

ModCliList.java: En esta clase, encontramos un choice con una lista de clientes en el que seleccionaremos un cliente para que podamos modificarlo a través de la clase ModCli.java.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.Choice;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.*.*;

public class ModCliList implements WindowListener, ActionListener{
    JFrame ventanaModCliList = new JFrame ("Buscar cliente para
    modificar");
    JLabel lblClientes = new JLabel("Selecciona cliente");
    Choice clientes = new Choice();
    JButton btnSeleccionar = new JButton("Seleccionar");

    JPanel pnl1 = new JPanel();
    JPanel pnl2 = new JPanel();
    JPanel pnl3 = new JPanel();

    String user = "";

    public ModCliList(String usuario) {
        ventanaModCliList.setLayout(new GridLayout(3,1));
        ventanaModCliList.setLocationRelativeTo(null);
        ventanaModCliList.setSize(400,300);

        user = usuario;

        ResultSet selectClientes = ejecutarSelect("SELECT * FROM
        clientes",conectar("TallerJava","usuarioTaller","Studium2018;"));
        try {
            while(selectClientes.next())
```



```

        {
            String
cli=Integer.toString(selectClientes.getInt("idCliente"));
            cli = cli + ".-"+
"+selectClientes.getString("nombreCliente");
            clientes.add(cli);
        }
    } catch (SQLException e) {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }

    desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

    pnl1.add(lblClientes);
    pnl2.add(clientes);
    pnl3.add(btnSeleccionar);
    ventanaModCliList.add(pnl1);
    ventanaModCliList.add(pnl2);
    ventanaModCliList.add(pnl3);
    btnSeleccionar.addActionListener(this);
    ventanaModCliList.addWindowListener(this);
    ventanaModCliList.setVisible(true);
}
@Override
public void actionPerformed(ActionEvent ae)
{
    if(btnSeleccionar.equals(ae.getSource())) {

        String[] arrayClientes =
clientes.getSelectedItemAt().toString().split(".-");
        int idCliente = Integer.parseInt(arrayClientes[0]);
        new ModCli(idCliente, user);
        ventanaModCliList.setVisible(false);
    }
}
@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}
@Override
public void windowClosing(WindowEvent arg0)
{
    ventanaModCliList.setVisible(false);
}
}

```

```
@Override
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;

    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,
login,password);
    }
    catch (ClassNotFoundException cnfe)
    {
        JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    catch (SQLException sqle)
    {
        JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    return connection;
}

public void desconectar(Connection c)
{
    try
    {
        if(c!=null)
```

```
        {
            c.close();
        }
    }
    catch (SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
    }
}

public ResultSet ejecutarSelect(String sentencia, Connection c)
{
    try
    {
        Statement statement = c.createStatement();
        ResultSet rs= statement.executeQuery(sentencia);
        return rs;
    }
    catch(SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
        return null;
    }
}
}
```

ModRec.java: En esta clase, podremos modificar a un recambio que hemos elegido anteriormente usando la clase ModRecList.java.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
```

```
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Calendar;

import javax.swing.JButton;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class ModRec implements WindowListener, ActionListener{
    JFrame ventanaModRec = new JFrame ("Modificar recambio:");
    JLabel lblDescripcionRec = new JLabel ("Descripción:");
    JLabel lblUnidadesRec = new JLabel ("Unidades:");
    JLabel lblPrecioRec = new JLabel ("Precio:");

    JTextField txtDescripcionRec = new JTextField(10);
    JTextField txtUnidadesRec = new JTextField(10);
    JTextField txtPrecioRec = new JTextField(10);

    JButton btnModificar = new JButton("Modificar Recambio");
    JButton btnLimpiar = new JButton("Limpiar");

    JPanel pnlPanel = new JPanel();
    JPanel pnlPanel2 = new JPanel();
    JPanel pnlPanel3 = new JPanel();
    JPanel pnlPanel4 = new JPanel();
    int idRec = 0;
    String user = "";
    public ModRec(int id, String usuario)
    {
        user = usuario;
        idRec = id;
        ResultSet rs = ejecutarSelect("SELECT * FROM recambios
where
idRecambio="+id+";",conectar("TallerJava","usuarioTaller","Studium20
18;"));
```

```
        try {
            rs.next();
            txtDescripcionRec.selectAll();

            txtDescripcionRec.setText(rs.getString("descripcionRecambio"));
            txtUnidadesRec.selectAll();

            txtUnidadesRec.setText(rs.getString("unidadesRecambio"));
            txtPrecioRec.selectAll();

            txtPrecioRec.setText(Integer.toString(rs.getInt("precioRecambio"
            ")));
        } catch (SQLException e) {

            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
            JOptionPane.ERROR_MESSAGE);
        }

        desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
        "));

        ventanaModRec.setLayout(new GridLayout(4,1));
        ventanaModRec.setLocationRelativeTo(null);
        ventanaModRec.setSize(400,300);

        pnlPanel1.setLayout(new FlowLayout());
        pnlPanel2.setLayout(new FlowLayout());
        pnlPanel3.setLayout(new FlowLayout());
        pnlPanel4.setLayout(new FlowLayout());

        pnlPanel1.add(lblDescripcionRec);
        pnlPanel1.add(txtDescripcionRec);
        ventanaModRec.add(pnlPanel1);

        pnlPanel2.add(lblUnidadesRec);
        pnlPanel2.add(txtUnidadesRec);
        ventanaModRec.add(pnlPanel2);

        pnlPanel3.add(lblPrecioRec);
        pnlPanel3.add(txtPrecioRec);
        ventanaModRec.add(pnlPanel3);

        pnlPanel4.add(btnModificar);
        btnModificar.addActionListener(this);
        pnlPanel4.add(btnLimpiar);
```

```

        btnLimpiar.addActionListener(this);
        ventanaModRec.add(pnlPanel4);

        ventanaModRec.addWindowListener(this);
        ventanaModRec.setVisible(true);

    }

    @Override
    public void actionPerformed(ActionEvent ae)
    {

        if (btnModificar.equals(ae.getSource())) {
            if(txtDescripcionRec.getText().equals("")) {
                JOptionPane.showMessageDialog(null, "Error,
Nombre de recambio vacío", "Nombre vacío",
JOptionPane.ERROR_MESSAGE);

            } else {
                String sentencia = "UPDATE recambios SET
descripcionRecambio = '"+txtDescripcionRec.getText()+"',
unidadesRecambio = '"+txtUnidadesRec.getText()+"', precioRecambio
='"+txtPrecioRec.getText()+" WHERE idRecambio ="+idRec+"";

                ejecutarIDA(sentencia, conectar("TallerJava", "usuarioTaller", "St
udium2018;"));

                desconectar(conectar("TallerJava", "usuarioTaller", "Stadium2018;
"));

                Calendar horaFecha = Calendar.getInstance();
                int hora, minutos, dia, mes, anyo;
                hora = horaFecha.get(Calendar.HOUR_OF_DAY);
                minutos = horaFecha.get(Calendar.MINUTE);
                dia = horaFecha.get(Calendar.DAY_OF_MONTH);
                mes = horaFecha.get(Calendar.MONTH)+1;
                anyo = horaFecha.get(Calendar.YEAR);
                try {
                    FileWriter fw = new
FileWriter("movimientos.log", true);
                    BufferedWriter bw = new
BufferedWriter(fw);

                    PrintWriter outPut = new PrintWriter(bw);

                    outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ] [ "+hora+": "+minutos+" ]
"+ " [ "+user+" ] "+ " [ "+sentencia+" ] "+ "\n");
                    outPut.close();
                    bw.close();

```

```
        fw.close();
    } catch (IOException ioe) {
        System.out.print("Error");
    }

    }

    } else if (btnLimpiar.equals(ae.getSource())) {
        txtDescripcionRec.selectAll();
        txtDescripcionRec.setText("");
        txtUnidadesRec.selectAll();
        txtUnidadesRec.setText("");
        txtPrecioRec.selectAll();
        txtPrecioRec.setText("");
    }

}

@Override
public void windowActivated(WindowEvent arg0) {}

@Override
public void windowClosed(WindowEvent arg0) {}

@Override
public void windowClosing(WindowEvent arg0)
{
    ventanaModRec.setVisible(false);
}

@Override
public void windowDeactivated(WindowEvent arg0) {}

@Override
public void windowDeiconified(WindowEvent arg0) {}

@Override
public void windowIconified(WindowEvent arg0) {}

@Override
public void windowOpened(WindowEvent arg0) {}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
```

```
String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
String login = usuario;
String password = clave;
Connection connection = null;

try
{
    Class.forName(driver);
    connection = DriverManager.getConnection(url,
login,password);
}
catch (ClassNotFoundException cnfe)
{
    JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
}
catch (SQLException sqle)
{
    JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
}
return connection;
}

public void desconectar(Connection c)
{
    try
    {
        if(c!=null)
        {
            c.close();
        }
    }
    catch (SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
}

public ResultSet ejecutarSelect(String sentencia, Connection c)
{
```



```
        try
        {
            Statement statement = c.createStatement();
            ResultSet rs= statement.executeQuery(sentencia);
            return rs;
        }
        catch(SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
            JOptionPane.ERROR_MESSAGE);
            return null;
        }
    }

    public void ejecutarIDA(String sentencia, Connection c)
    {
        try
        {
            Statement statement = c.createStatement();
            statement.executeUpdate(sentencia);
            JOptionPane.showMessageDialog(null,"Recambio
            modificado","Recambio modificado con éxito",
            JOptionPane.INFORMATION_MESSAGE);
        }
        catch(SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
            JOptionPane.ERROR_MESSAGE);
        }
    }
}
```

ModRecList.java: En esta clase, encontramos un choice con una lista de recambios en el que seleccionaremos un recambio para que podamos modificarlo a través de la clase ModRec.java

```
package es.studium.PracticaSegundoTrimestre;
```

```

import java.awt.Choice;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;

public class ModRecList extends JFrame implements WindowListener,
ActionListener{
    /**
     *
     */
    private static final long serialVersionUID = 1L;
    JLabel lblRecambios = new JLabel("Selecciona recambio");
    Choice recambios = new Choice();
    JButton btnSeleccionar = new JButton("Seleccionar");

    JPanel pnl1 = new JPanel();
    JPanel pnl2 = new JPanel();
    JPanel pnl3 = new JPanel();

    String user = "";

    public ModRecList(String usuario)
    {
        user = usuario;
        this.setTitle("Buscar recambio para modificar");
        this.setLayout(new GridLayout(3,1));
        this.setLocationRelativeTo(null);
        this.setSize(400,300);

        ResultSet selectRecambios = ejecutarSelect("SELECT * FROM
recambios",conectar("TallerJava","usuarioTaller","Stodium2018;"));
        try {
            while(selectRecambios.next())
            {
                String

```

```

rec=Integer.toString(selectRecambios.getInt("idRecambio"));
        rec = rec + "-" +
selectRecambios.getString("descripcionRecambio");
        recambios.add(rec);
    }
} catch (SQLException e) {

    JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
}

desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

    pnl1.add(lblRecambios);
    pnl2.add(recambios);
    pnl3.add(btnSeleccionar);
    this.add(pnl1);
    this.add(pnl2);
    this.add(pnl3);
    btnSeleccionar.addActionListener(this);
    this.addWindowListener(this);
    this.setVisible(true);
}
public void actionPerformed(ActionEvent ae)
{
    if(btnSeleccionar.equals(ae.getSource())) {

        String[] array=
recambios.getSelectedItem().toString().split("-");
        int idRecambio = Integer.parseInt(array[0]);
        new ModRec(idRecambio, user);
        this.setVisible(false);
    }

}
@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}
@Override
public void windowClosing(WindowEvent arg0)
{
    this.setVisible(false);
}

@Override

```

```
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSS
L=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;

    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,
login,password);
    }
    catch (ClassNotFoundException cnfe)
    {
        JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    catch (SQLException sqle)
    {
        JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    return connection;
}

public void desconectar(Connection c)
{
    try
    {
        if(c!=null)
        {
            c.close();
        }
    }
}
```

```
        }
    }
    catch (SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
    }
}

public ResultSet ejecutarSelect(String sentencia, Connection c)
{
    try
    {
        Statement statement = c.createStatement();
        ResultSet rs= statement.executeQuery(sentencia);
        return rs;
    }
    catch(SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
        return null;
    }
}
}
```

ModRep.java: En esta clase, podremos modificar a una reparación que hemos elegido anteriormente usando la clase ModRepList.java.

```
package es.studium.PracticaSegundoTrimestre;

import java.awt.*;
import java.awt.event.*;
import java.awt.event.ActionListener;
import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
```

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Calendar;

import javax.swing.*;

public class ModRep extends JFrame implements WindowListener,
ActionListener
{

    /**
     *
     */
    private static final long serialVersionUID = 1L;

    JLabel lblAveriaRep = new JLabel ("Avería:");
    JLabel lblFechaEntradaRep = new JLabel ("Fecha de Entrada:");
    JLabel lblFechaSalidaRep = new JLabel ("Fecha de Salida:");
    JLabel lblReparadoRep = new JLabel ("Reparado:");

    JTextField txtAveriaRep = new JTextField(10);
    JTextField txtFechaEntradaRep = new JTextField(10);
    JTextField txtFechaSalidaRep = new JTextField(10);
    ButtonGroup chkReparadoRep = new ButtonGroup ();
    JRadioButton chkSiRep = new JRadioButton ("Sí", false);
    JRadioButton chkNoRep = new JRadioButton ("No", true);

    JButton btnModificar = new JButton("Modificar Reparación");
    JButton btnLimpiar = new JButton("Limpiar");

    JPanel pnlPanel = new JPanel();
    JPanel pnlPanel2 = new JPanel();
    JPanel pnlPanel3 = new JPanel();
    JPanel pnlPanel4 = new JPanel();
    JPanel pnlPanel5 = new JPanel();

    int idRep = 0;
    String user = "";

    public ModRep(int id, String usuario)
    {
        user = usuario;
        idRep = id;
    }
}
```

```

        ResultSet rs = ejecutarSelect("SELECT * FROM reparaciones
where
idReparacion="+id+";",conectar("TallerJava","usuarioTaller","Studium
2018;"));
        try {
            rs.next();
            if(rs.getInt("reparado")==1) {

                txtAveriaRep.selectAll();
                txtAveriaRep.setText(rs.getString("Averia"));
                txtFechaEntradaRep.selectAll();

                txtFechaEntradaRep.setText(rs.getString("DATE_FORMAT(fechaEntra
da, '%d/%m%a'"));
                txtFechaSalidaRep.selectAll();

                txtFechaSalidaRep.setText(rs.getString("DATE_FORMAT(fechaSalida
, '%d/%m%a'"));
                chkSiRep.setSelected(true);
                chkNoRep.setSelected(false);
            } else if(rs.getInt("reparado")==0) {

                txtAveriaRep.selectAll();
                txtAveriaRep.setText(rs.getString("Averia"));
                txtFechaEntradaRep.selectAll();

                txtFechaEntradaRep.setText(rs.getString("DATE_FORMAT(fechaEntra
da, '%d/%m%a'"));
                txtFechaSalidaRep.selectAll();

                txtFechaSalidaRep.setText(rs.getString("DATE_FORMAT(fechaSalida
, '%d/%m%a'"));
                chkSiRep.setSelected(false);
                chkNoRep.setSelected(true);
            }

        } catch (SQLException e) {

            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
        }

        desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

```

```
this.setTitle("Modificar Reparación");
this.setLayout(new GridLayout(5,2));
this.setLocationRelativeTo(null);
this.setSize(400,300);

pnlPanel.setLayout(new FlowLayout());
pnlPanel2.setLayout(new FlowLayout());
pnlPanel3.setLayout(new FlowLayout());
pnlPanel4.setLayout(new FlowLayout());
pnlPanel5.setLayout(new FlowLayout());

pnlPanel.add(lblAveriaRep);
pnlPanel.add(txtAveriaRep);
this.add(pnlPanel);

pnlPanel2.add(lblFechaEntradaRep);
pnlPanel2.add(txtFechaEntradaRep);
this.add(pnlPanel2);

pnlPanel3.add(lblFechaSalidaRep);
pnlPanel3.add(txtFechaSalidaRep);
this.add(pnlPanel3);

pnlPanel4.add(lblReparadoRep);
chkReparadoRep.add(chkSiRep);
chkReparadoRep.add(chkNoRep);
pnlPanel4.add(chkSiRep);
pnlPanel4.add(chkNoRep);
this.add(pnlPanel4);

pnlPanel5.add(btnModificar);
btnModificar.addActionListener(this);
pnlPanel5.add(btnLimpiar);
btnLimpiar.addActionListener(this);
this.add(pnlPanel5);

this.addWindowListener(this);
this.setVisible(true);
}

@Override
public void actionPerformed(ActionEvent ae)
{
    if (btnModificar.equals(ae.getSource())) {
        if(txtAveriaRep.getText().equals("")) {
            JOptionPane.showMessageDialog(null,"Error,
```



```

Nombre de recambio vacío", "Nombre vacío",
JOptionPane.ERROR_MESSAGE);

        } else {
            if(chkSiRep.isSelected())
            {
                String FechaEntrada =
txtFechaEntradaRep.getText();
                String[] arrayFechaEntrada =
FechaEntrada.split("/");
                FechaEntrada = arrayFechaEntrada[2]+"-
"+arrayFechaEntrada[1]+"-"+arrayFechaEntrada[0];

                String FechaSalida =
txtFechaSalidaRep.getText();
                String[] arrayFechaSalida =
FechaSalida.split("/");
                FechaSalida = arrayFechaSalida[2]+"-
"+arrayFechaSalida[1]+"-"+arrayFechaSalida[0];

                String sentencia1 = "UPDATE reparaciones
SET Averia='"+txtAveriaRep.getText()+"',
fechaEntrada='"+FechaEntrada+"', fechaSalida='"+FechaSalida+"',
Reparado = 1 WHERE idReparacion="+idRep+"";

                ejecutarIDA(sentencia1, conectar("TallerJava", "usuarioTaller", "S
tudium2018;"));

                desconectar(conectar("TallerJava", "usuarioTaller", "Studium2018;
"));

                Calendar horaFecha =
Calendar.getInstance();

                int hora, minutos, dia, mes, anyo;
                hora =
horaFecha.get(Calendar.HOUR_OF_DAY);
                minutos = horaFecha.get(Calendar.MINUTE);
                dia =
horaFecha.get(Calendar.DAY_OF_MONTH);
                mes = horaFecha.get(Calendar.MONTH)+1;
                anyo = horaFecha.get(Calendar.YEAR);
                try {
                    FileWriter fw = new
FileWriter("movimientos.log", true);
                    BufferedWriter bw = new
BufferedWriter(fw);

                    PrintWriter outPut = new
PrintWriter(bw);

```

```

        outPut.print("[ "+dia+"/"+mes+"/"+anyo+" ]["+hora+": "+minutos+" ]
"+"["+user+"]"+"["+sentencia1+"]"+"\\n");
        outPut.close();
        bw.close();
        fw.close();
    } catch (IOException ioe) {
        System.out.print("Error");
    }
} else
{
    String FechaEntrada =
txtFechaEntradaRep.getText();
    String[] arrayFechaEntrada =
FechaEntrada.split("/");
    FechaEntrada = arrayFechaEntrada[2]+"-
"+arrayFechaEntrada[1]+"-"+arrayFechaEntrada[0];

    String FechaSalida =
txtFechaSalidaRep.getText();
    String[] arrayFechaSalida =
FechaSalida.split("/");
    FechaSalida = arrayFechaSalida[2]+"-
"+arrayFechaSalida[1]+"-"+arrayFechaSalida[0];

    String sentencia2 = "UPDATE reparaciones
SET Averia ='"+txtAveriaRep.getText()+"',
fechaEntrada='"+FechaEntrada+"', fechaSalida='"+FechaSalida+"',
Reparado = 0 WHERE idReparacion =" +idRep+"";

    ejecutarIDA(sentencia2,conectar("TallerJava","usuarioTaller","S
tudium2018;"));

    desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

    Calendar horaFecha =
Calendar.getInstance();
    int hora,minutos,dia,mes,anyo;
    hora =
horaFecha.get(Calendar.HOUR_OF_DAY);
    minutos = horaFecha.get(Calendar.MINUTE);
    dia =
horaFecha.get(Calendar.DAY_OF_MONTH);
    mes = horaFecha.get(Calendar.MONTH)+1;
    anyo = horaFecha.get(Calendar.YEAR);
    try {
        FileWriter fw = new

```

```

FileWriter("movimientos.log", true);
                                BufferedWriter bw = new
BufferedWriter(fw);
                                PrintWriter outPut = new
PrintWriter(bw);

        outPut.print("[ "+dia+"/"+mes+"/"+anyo+"["+hora+": "+minutos+"]
"+"["+user+"]"+"["+sentencia2+"]"+"\\n");
                                outPut.close();
                                bw.close();
                                fw.close();
                                } catch(IOException ioe) {
                                    System.out.print("Error");
                                }
                            }
                        }

                } else if (btnLimpiar.equals(ae.getSource()))
                {
                    txtAveriaRep.selectAll();
                    txtAveriaRep.setText("");
                    txtFechaEntradaRep.selectAll();
                    txtFechaEntradaRep.setText("");
                    txtFechaSalidaRep.selectAll();
                    txtFechaSalidaRep.setText("");
                }
            }
        }

        @Override
        public void windowActivated(WindowEvent arg0) {}
        @Override
        public void windowClosed(WindowEvent arg0) {}
        @Override
        public void windowClosing(WindowEvent arg0) {
            this.setVisible(false);
        }
        @Override
        public void windowDeactivated(WindowEvent arg0) {}
        @Override
        public void windowDeiconified(WindowEvent arg0) {}
        @Override
        public void windowIconified(WindowEvent arg0) {}
        @Override
        public void windowOpened(WindowEvent arg0) {}

        public Connection conectar(String baseDatos, String usuario,
String clave)
        {

```

```
String driver = "com.mysql.jdbc.Driver";
String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
String login = usuario;
String password = clave;
Connection connection = null;

try
{
    Class.forName(driver);
    connection = DriverManager.getConnection(url,
login,password);
}
catch (ClassNotFoundException cnfe)
{

    JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
}
catch (SQLException sqle)
{

    JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
}
return connection;
}

public void desconectar(Connection c)
{
    try
    {
        if(c!=null)
        {
            c.close();
        }
    }
    catch (SQLException e)
    {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
}

public ResultSet ejecutarSelect(String sentencia, Connection c)
```

```

    {
        try
        {
            Statement statement = c.createStatement();
            ResultSet rs= statement.executeQuery(sentencia);
            return rs;
        }
        catch(SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
            JOptionPane.ERROR_MESSAGE);
            return null;
        }
    }

    public void ejecutarIDA(String sentencia, Connection c)
    {
        try
        {
            Statement statement = c.createStatement();
            statement.executeUpdate(sentencia);
            JOptionPane.showMessageDialog(null,"Reparación
modificada","Reparación modificada con éxito",
JOptionPane.INFORMATION_MESSAGE);

        }
        catch(SQLException e)
        {
            JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
            JOptionPane.ERROR_MESSAGE);
        }
    }
}

```

ModRepList.java: En esta clase, encontramos un choice con una lista de reparaciones en el que seleccionaremos una reparación para que podamos modificarla a través de la clase ModRep.java

```
package es.studium.PracticaSegundoTrimestre;
```

```
import java.awt.Choice;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import java.awt.event.WindowEvent;
import java.awt.event.WindowListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.*.*;

public class ModRepList extends JFrame implements WindowListener,
ActionListener{

    /**
     *
     */
    private static final long serialVersionUID = 1L;

    JLabel lblReparaciones = new JLabel("Selecciona reparación");
    Choice reparaciones = new Choice();
    JButton btnSeleccionar = new JButton("Seleccionar");

    JPanel pnl1 = new JPanel();
    JPanel pnl2 = new JPanel();
    JPanel pnl3 = new JPanel();
    String user = "";
    public ModRepList(String usuario)
    {
        user = usuario;
        this.setTitle("Buscar reparación para modificar");
        this.setLayout(new GridLayout(3,1));
        this.setLocationRelativeTo(null);
        this.setSize(400,300);

        ResultSet selectReparaciones = ejecutarSelect("SELECT *
FROM
reparaciones",conectar("TallerJava","usuarioTaller","Stadium2018;"))
;

        try {
```

```

        while(selectReparaciones.next())
        {
            String
rep=Integer.toString(selectReparaciones.getInt("idReparacion"));
            rep = rep + "-"+
selectReparaciones.getString("Averia");
            reparaciones.add(rep);
        }
    } catch (SQLException e) {

        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }

    desconectar(conectar("TallerJava","usuarioTaller","Studium2018;
"));

    pnl1.add(lblReparaciones);
    pnl2.add(reparaciones);
    pnl3.add(btnSeleccionar);
    this.add(pnl1);
    this.add(pnl2);
    this.add(pnl3);
    btnSeleccionar.addActionListener(this);
    this.addWindowListener(this);
    this.setVisible(true);
}
@Override
public void actionPerformed(ActionEvent ae)
{
    if(btnSeleccionar.equals(ae.getSource())) {

        String[] array=
reparaciones.getSelectedItemAt().toString().split("-");
        int idReparacion = Integer.parseInt(array[0]);
        new ModRep(idReparacion, user);
        this.setVisible(false);
    }
}
@Override
public void windowActivated(WindowEvent arg0) {}
@Override
public void windowClosed(WindowEvent arg0) {}
@Override
public void windowClosing(WindowEvent arg0)
{
    this.setVisible(false);
}

```

```
}
@Override
public void windowDeactivated(WindowEvent arg0) {}
@Override
public void windowDeiconified(WindowEvent arg0) {}
@Override
public void windowIconified(WindowEvent arg0) {}
@Override
public void windowOpened(WindowEvent arg0) {}

public Connection conectar(String baseDatos, String usuario,
String clave)
{
    String driver = "com.mysql.jdbc.Driver";
    String url
="jdbc:mysql://localhost:3306/"+baseDatos+"?autoReconnect=true&useSSL=false";
    String login = usuario;
    String password = clave;
    Connection connection = null;

    try
    {
        Class.forName(driver);
        connection = DriverManager.getConnection(url,
login,password);
    }
    catch (ClassNotFoundException cnfe)
    {
        JOptionPane.showMessageDialog(null,cnfe.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    catch (SQLException sqle)
    {
        JOptionPane.showMessageDialog(null,sqle.getMessage(),"Error",
JOptionPane.ERROR_MESSAGE);
    }
    return connection;
}

public void desconectar(Connection c)
{
    try
    {
```



```
        if(c!=null)
        {
            c.close();
        }
    }
    catch (SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
    }
}

public ResultSet ejecutarSelect(String sentencia, Connection c)
{
    try
    {
        Statement statement = c.createStatement();
        ResultSet rs= statement.executeQuery(sentencia);
        return rs;
    }
    catch(SQLException e)
    {
        JOptionPane.showMessageDialog(null,e.getMessage(),"Error",
        JOptionPane.ERROR_MESSAGE);
        return null;
    }
}
}
```

7. Bibliografía

Temario Tema 9 Programación. (10 de 06 de 2019). Obtenido de Temario Tema 9 Programación:

http://aulastudium.com/pluginfile.php?file=%2F11026%2Fmod_resource%2Fcontent%2F2%2FPR-TEMA9%20Acceso%20a%20Bases%20de%20Datos.pdf