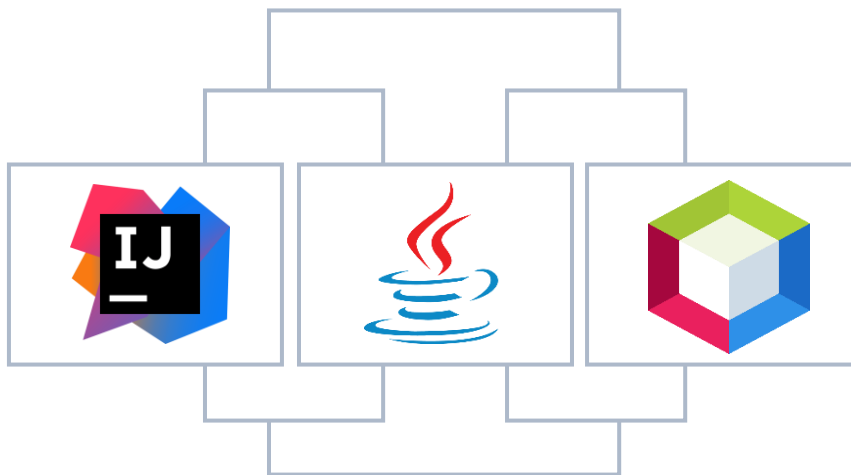




Universidad Peruana Los Andes

Facultad de Ingeniería

Escuela Profesional de Ingeniería de Sistemas y Computación



Arquitectura de **Software**

Mg. Ing. Raúl Fernández Bejarano

2024

CRUD

CREATE

READ

UPDATE

DELETE



CRUD en Java

Ejercicios

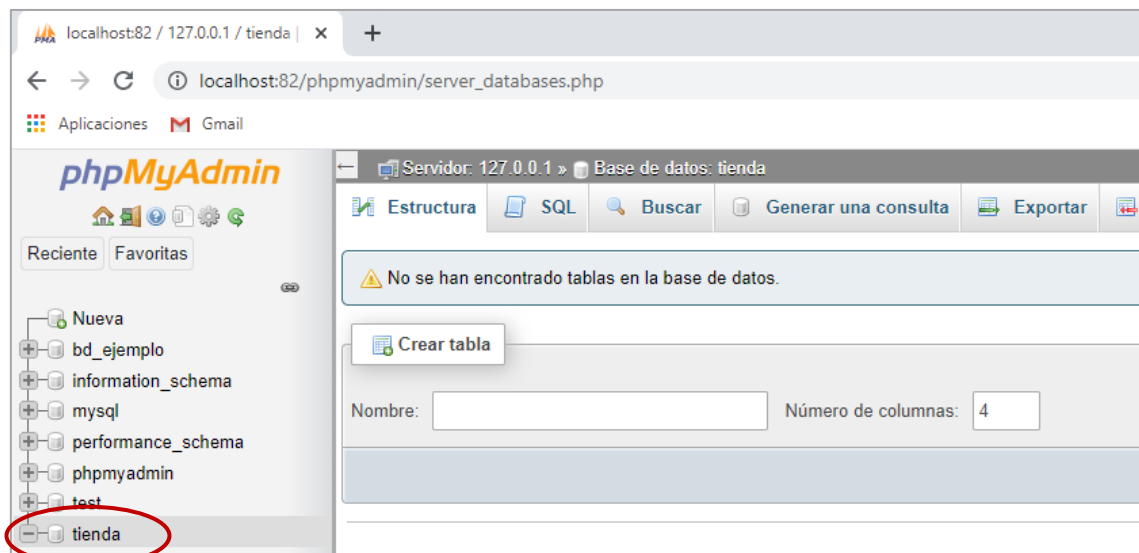
1. *CRUD en Java Escritorio MVC – Listar*
2. *CRUD en Java Escritorio MVC – Agregar*
3. *CRUD en Java Escritorio MVC – Actualizar*
4. *CRUD en Java Escritorio MVC – Eliminar*

Solución

Antes de realizar los ejercicios, haremos una *prueba de conexión* de Java con MySQL, para ello crearemos una base de datos llamada tienda, no crearemos ninguna tabla, la conexión lo haremos sólo con la base de datos creada, posteriormente haremos los cuatro *CRUD* dejados como ejercicios.

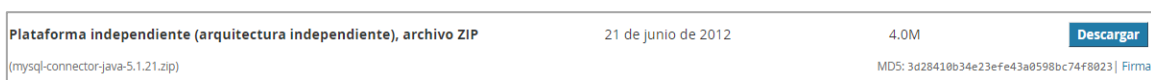
0. Prueba de Conexión

Creemos una base de datos llamado tienda:

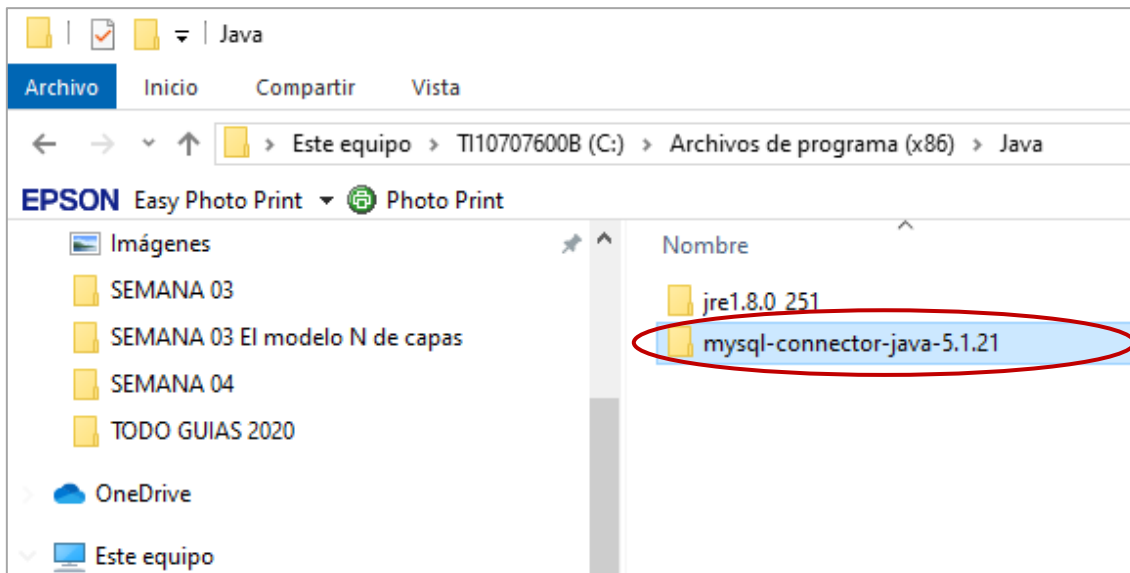


A continuación descargamos y guardaremos el driver o librería necesaria para realizar la prueba de conexión:

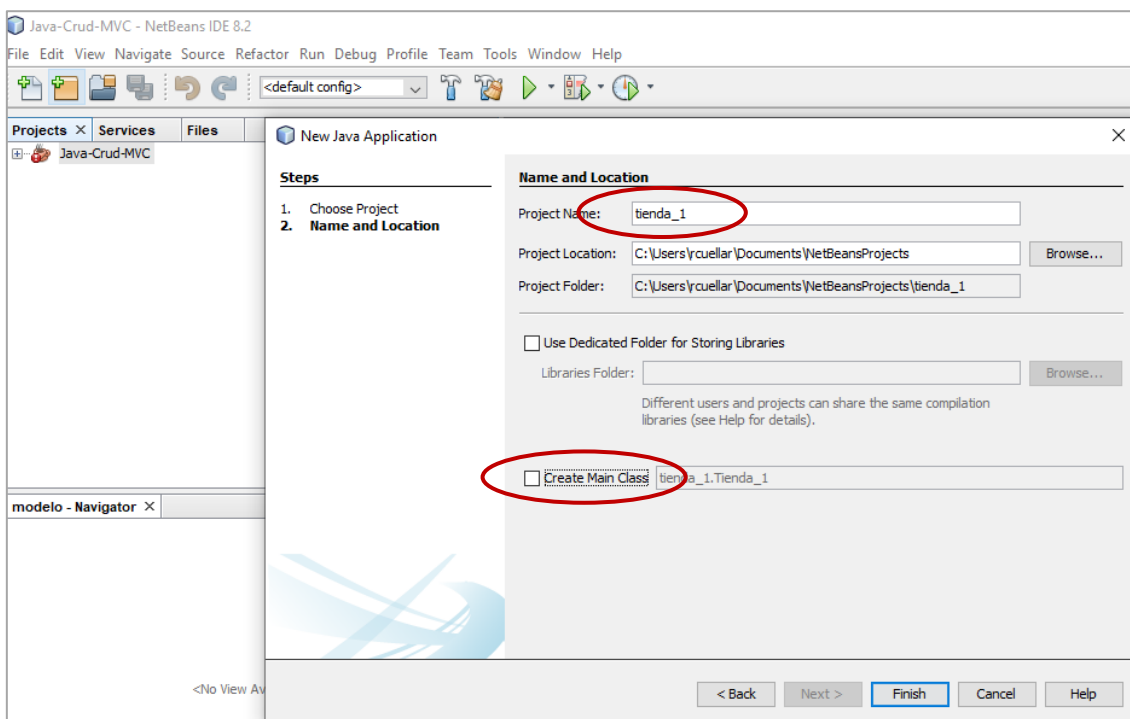
Descargamos el archivo *mysql-connector-java-5.1.21-bin.jar*



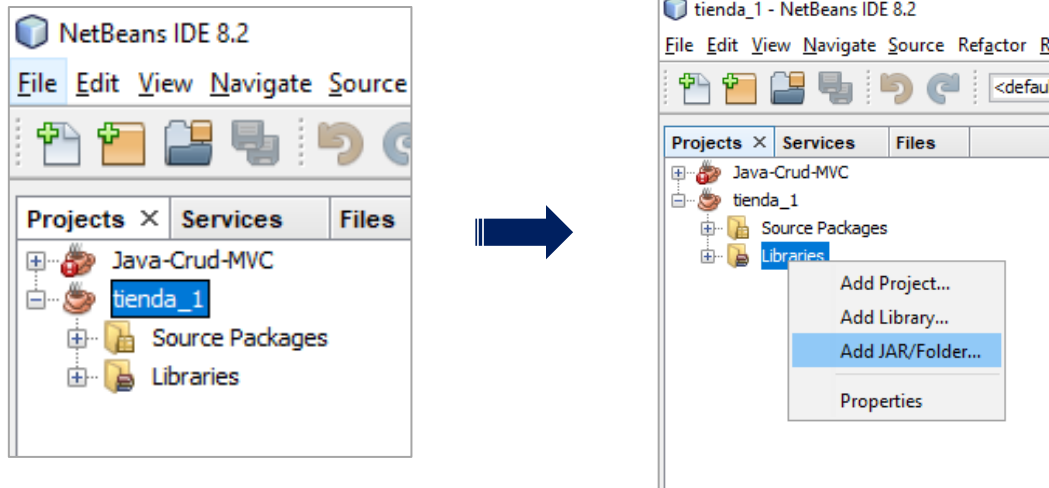
Guardaremos el archivo en una carpeta (mysql-connector-5.1.21)



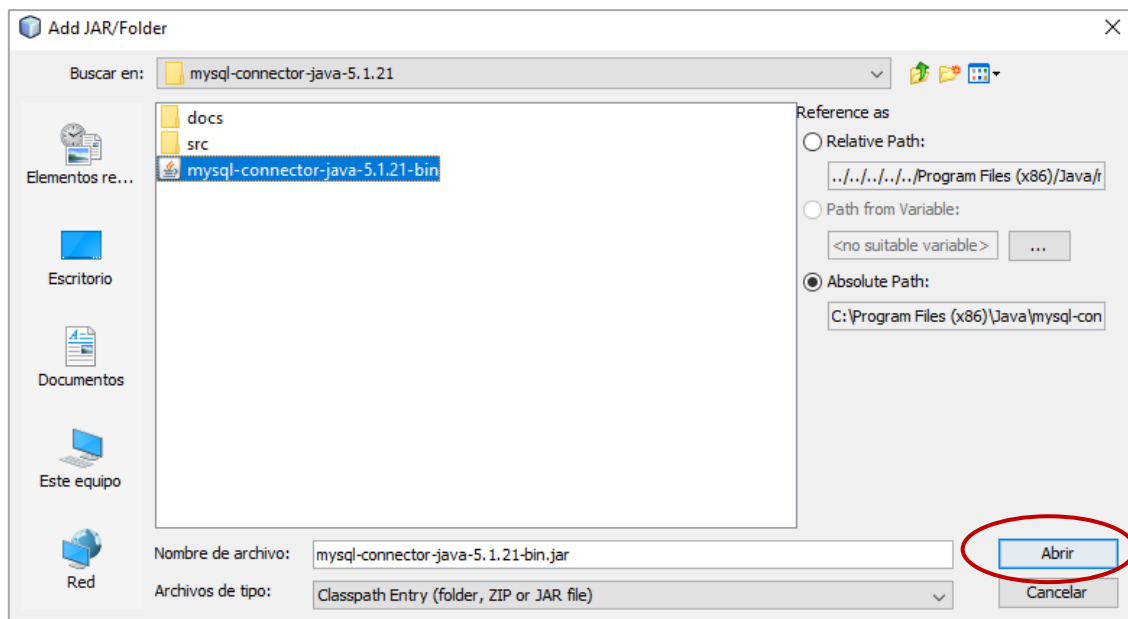
A Continuación en Netbeans creamos un nuevo proyecto llamado tienda_1, (desactivamos el Create main class):



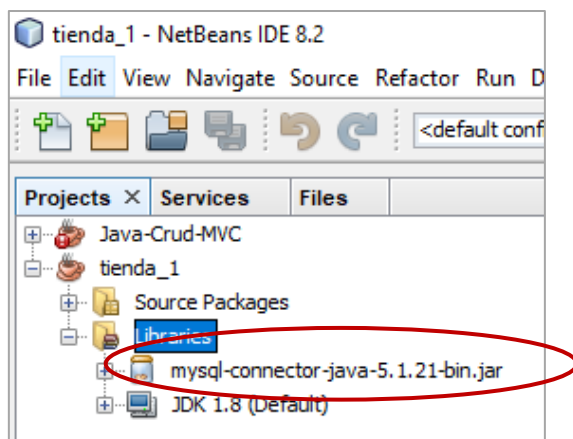
Una vez creada el proyecto tienda_1, importaremos el driver o librería, descargado anteriormente), damos clic derecho en libraries y Add JAR/Folder:



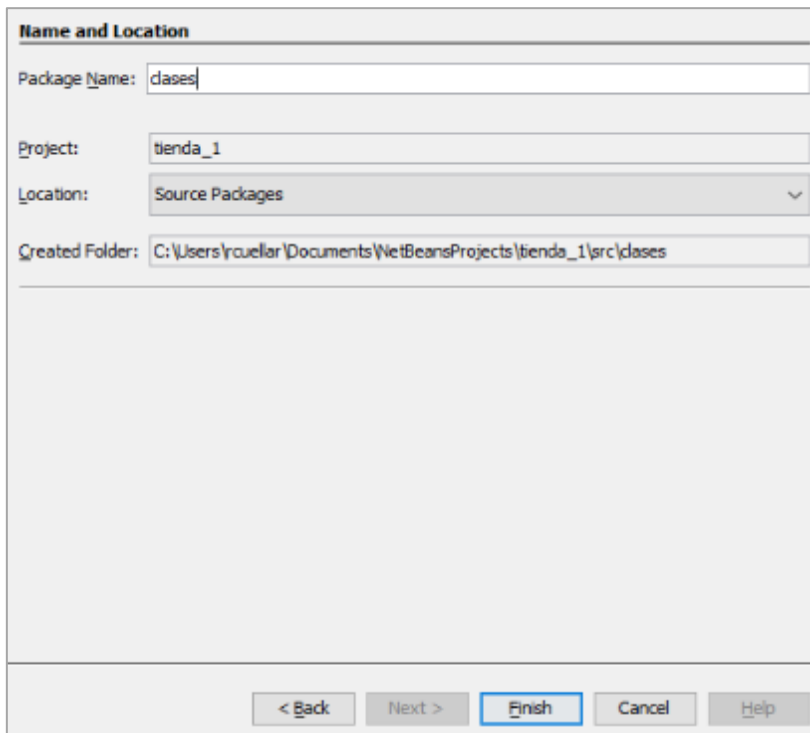
Abrimos la carpeta y el archivo *mysql-connector-java-5.1.21-bin.jar*



Y ya tenemos el archivo importado para el presente ejemplo:

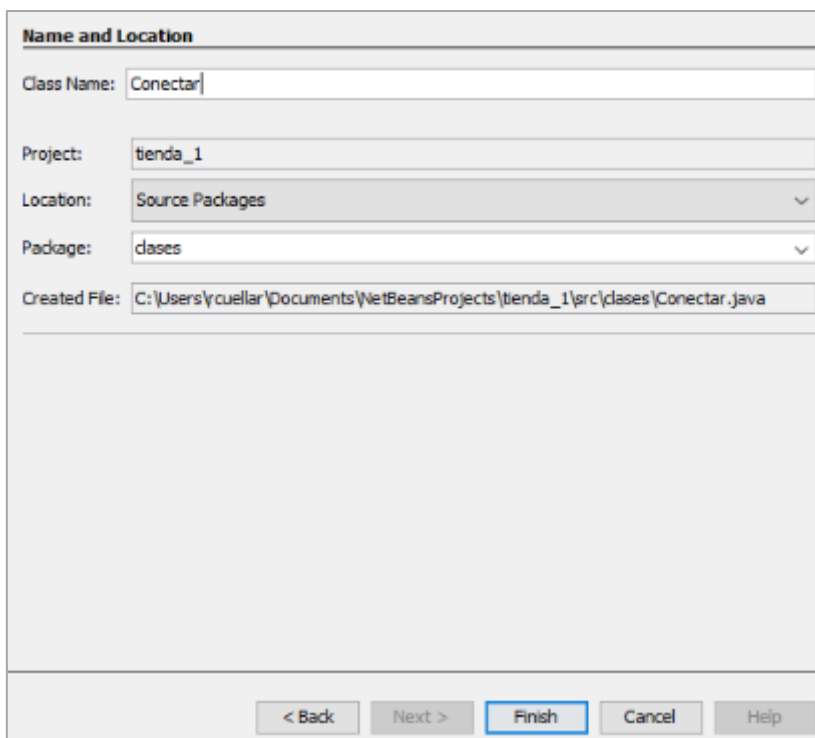


En el proyecto tienda_1 creamos un nuevo paquete llamado clases:



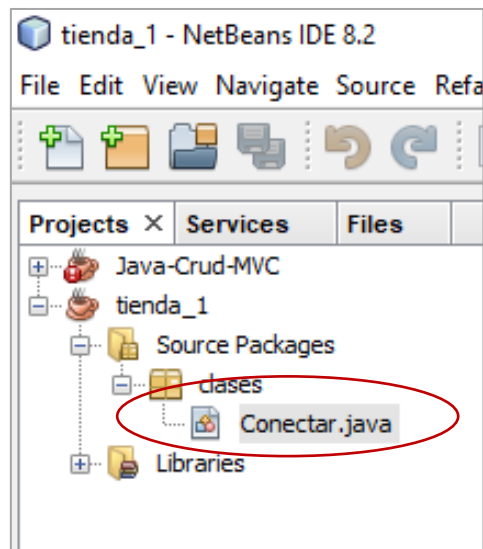
The 'Name and Location' dialog in NetBeans is shown. The 'Package Name' field contains 'clases'. The 'Project' field shows 'tienda_1'. The 'Location' dropdown is set to 'Source Packages'. The 'Created Folder' field displays the path 'C:\Users\rcuellar\Documents\NetBeansProjects\tienda_1\src\clases'. At the bottom, there are buttons for '< Back', 'Next >', 'Finish' (highlighted with a blue border), 'Cancel', and 'Help'.

Dentro del paquete clases crearemos una nueva clase llamada conectar:



The 'Name and Location' dialog in NetBeans is shown. The 'Class Name' field contains 'Conectar'. The 'Project' field shows 'tienda_1'. The 'Location' dropdown is set to 'Source Packages'. The 'Package' dropdown is set to 'clases'. The 'Created File' field displays the path 'C:\Users\rcuellar\Documents\NetBeansProjects\tienda_1\src\clases\Conectar.java'. At the bottom, there are buttons for '< Back', 'Next >', 'Finish' (highlighted with a blue border), 'Cancel', and 'Help'.

A continuación tenemos la clase conectar, donde escribiremos los códigos necesarios para hacer la prueba de conexión:



Escribimos el código en clases:

```
package clases;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class Conectar {

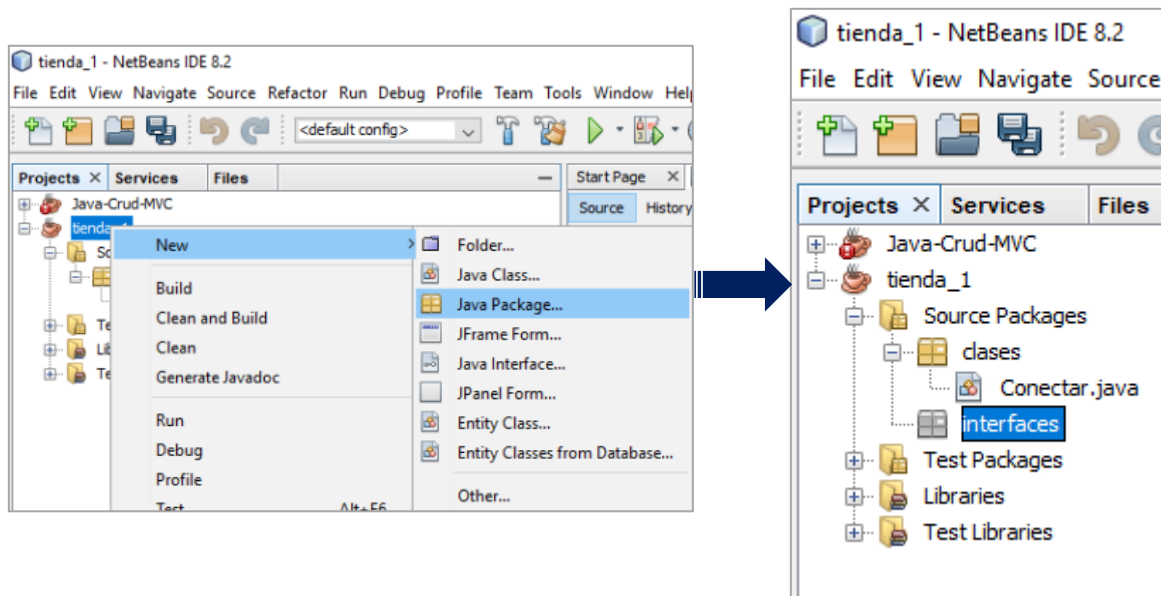
    private static Connection conn;
    private static final String driver = "com.mysql.jdbc.Driver";
    private static final String user = "root";
    private static final String password = "";
    private static final String url = "jdbc:mysql://localhost:3306/tienda";

    public Conectar() {
        conn=null;
        try{
            Class.forName(driver);
            conn=DriverManager.getConnection(url, user, password);
            if(conn !=null){
                System.out.println("Conexion establecida");
            }
        } catch (ClassNotFoundException | SQLException e){
            System.out.println("error al conectar" +e);
        }
    }

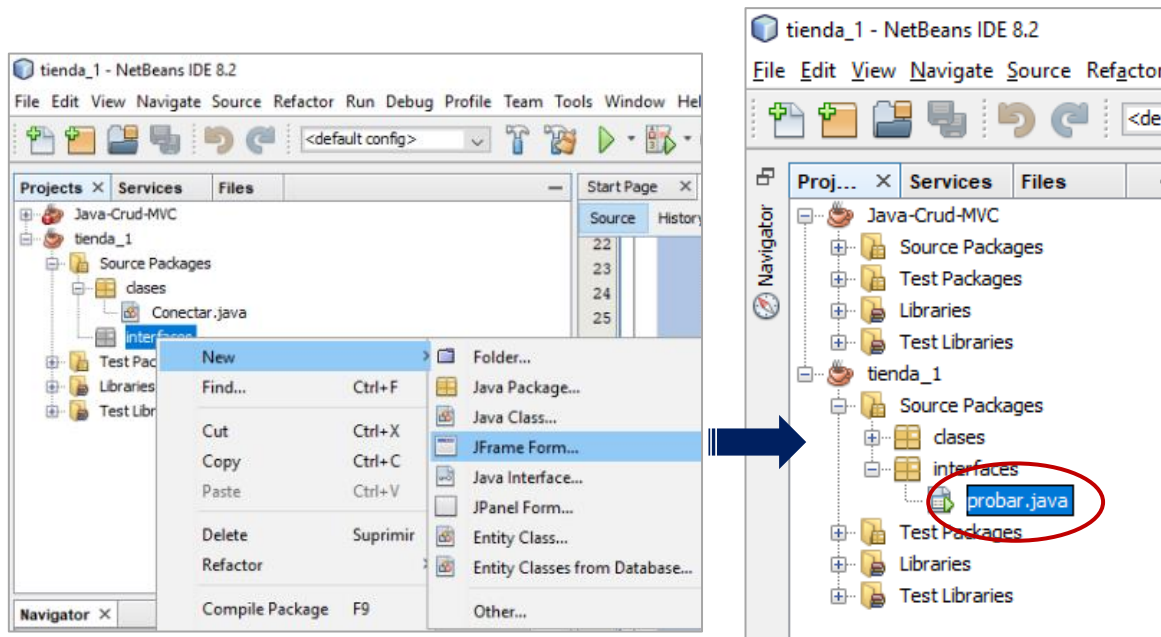
    // este metodo nos retorna la conexion
    public Connection getConnection(){
        return conn;
    }

    // con este metodo nos desconectamos de la base de datos
    public void desconectar(){
        conn=null;
        if(conn==null){
            System.out.println("conexion terminada..");
        }
    }
}
```

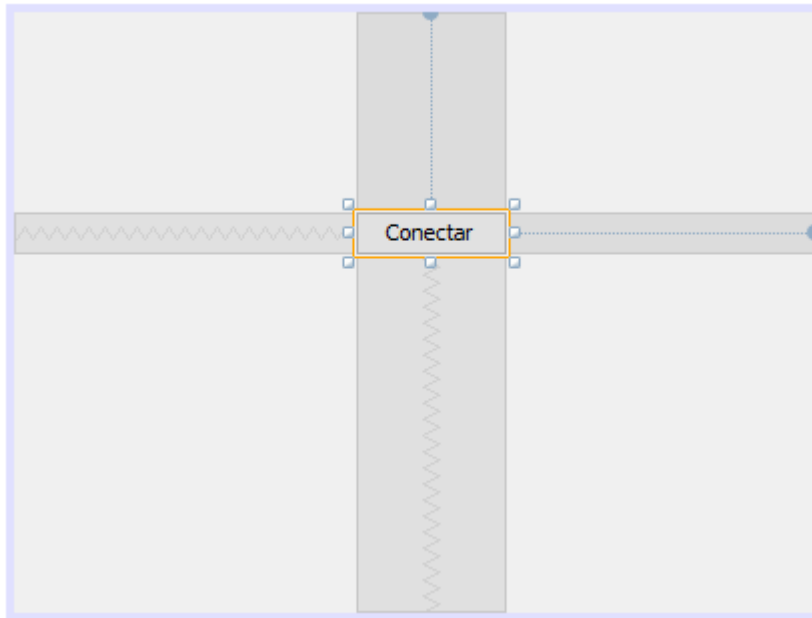

En tienda crearemos un nuevo paquete llamado interfaces (ahí estarán los Formularios, JFrame,)



En interfaces creamos un JFrame Form... llamada *probar*



Agregamos un JButton 1 llamado *Conectar*



Ahora le damos un evento al botón *Conectar*:

```

-  /*
   * To change this license header, choose License Headers in Project Properties.
   * To change this template file, choose Tools | Templates
   * and open the template in the editor.
   */
package interfaces;

-  import clases.Conectar;
   import java.sql.Connection;

-  /**
   *
   * @author rcuellar
   */
   public class probar extends javax.swing.JFrame {

       Conectar con;

-  /**
   * Creates new form probar
   */
-  public probar() {
       initComponents();
   }

-  /**
   * This method is called from within the constructor to initialize the form.
   * WARNING: Do NOT modify this code. The content of this method is always
   * regenerated by the Form Editor.

```

```

@SuppressWarnings("unchecked")
Generated Code

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    con= new Conectar();
    Connection reg=con.getConnection();
}

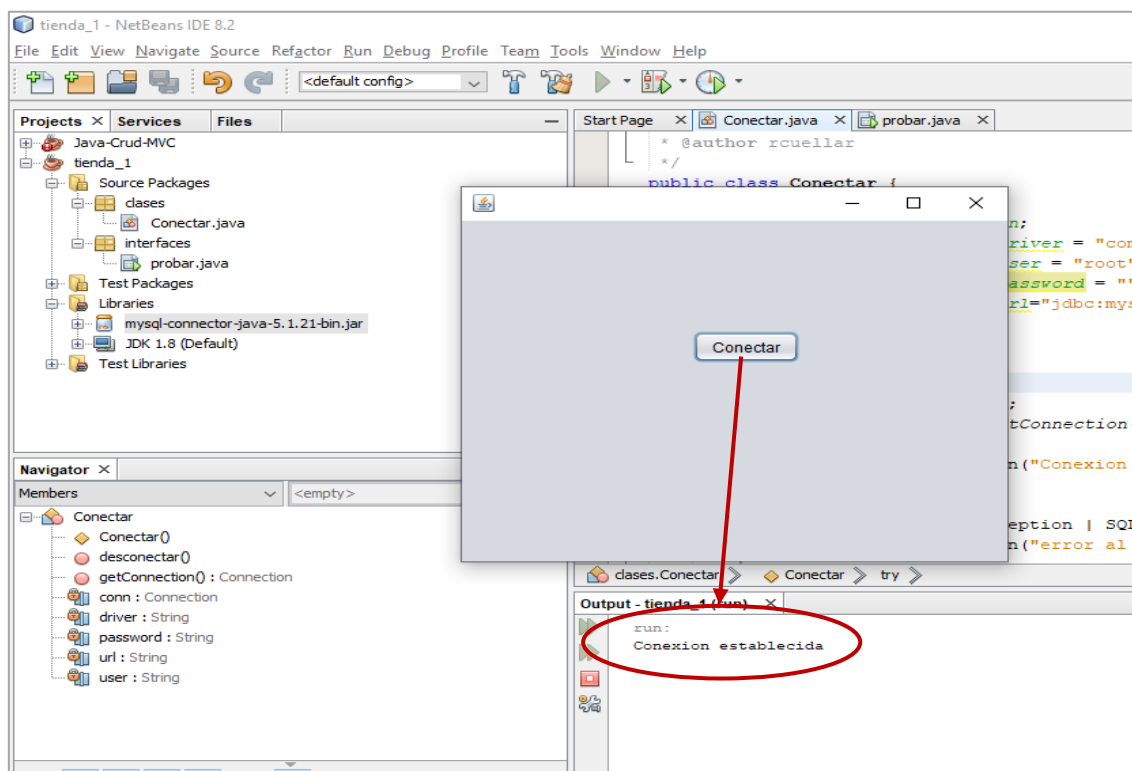
/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new probar().setVisible(true);
        }
    });
}

// Variables declaration - do not modify
private javax.swing.JButton jButton1;
// End of variables declaration
}

```

Probamos la conexión y finalmente observamos que se ha *establecido la conexión*.



A continuación realizaremos los cuatro ejercicios sobre CRUD:

1. CRUD en Java Escritorio MVC – Listar

Para realizar el ejercicio, crearemos primero una base de datos una tabla y registros y luego el *CRUD en java escritorio MVC-Listar*:

Base de datos, tabla y registros

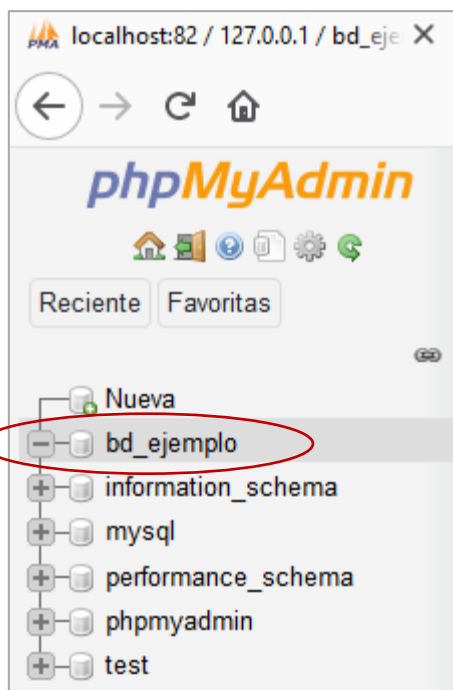
- a. Base datos: *bd_ejemplo*
- b. Tabla: *persona*
- c. Campos: *Id, nombres, correo, teléfono*

La tabla así como la tabla y los registros quedarán como se muestra:

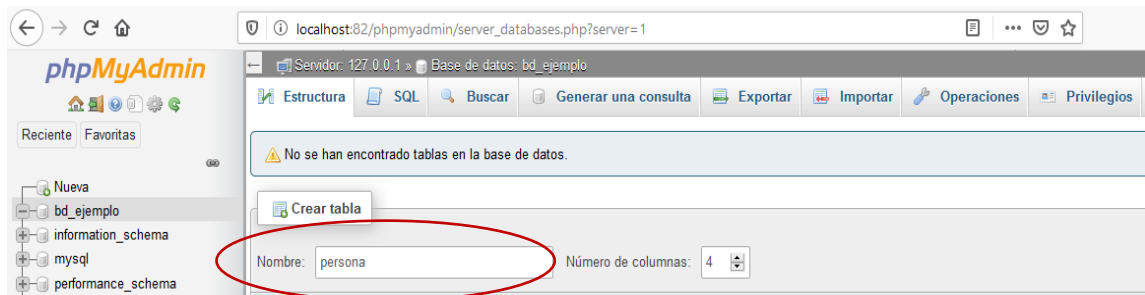
<div><div><div>←</div><div>→</div></div><div></div></div>						<div>Id</div>	<div>Nombres</div>	<div>Correo</div>	<div>Telefono</div>	
<div><div><div></div></div></div>	<div><div><div></div></div></div>	<div>Editar</div>	<div><div><div></div></div></div>	<div>Copiar</div>	<div><div><div></div></div></div>	<div>Borrar</div>	<div>1</div>	<div>Torres Peredo Perez</div>	<div>peredo@gmail.com</div>	<div>983252459</div>
<div><div><div></div></div></div>	<div><div><div></div></div></div>	<div>Editar</div>	<div><div><div></div></div></div>	<div>Copiar</div>	<div><div><div></div></div></div>	<div>Borrar</div>	<div>2</div>	<div>Torres Vargas Torres</div>	<div>torres@gmail.com</div>	<div>900567347</div>
<div><div><div></div></div></div>	<div><div><div></div></div></div>	<div>Editar</div>	<div><div><div></div></div></div>	<div>Copiar</div>	<div><div><div></div></div></div>	<div>Borrar</div>	<div>3</div>	<div>Mary Luz Milagros Vargas</div>	<div>maryluz@gmail.com</div>	<div>3456874</div>
<div><div><div></div></div></div>	<div><div><div></div></div></div>	<div>Editar</div>	<div><div><div></div></div></div>	<div>Copiar</div>	<div><div><div></div></div></div>	<div>Borrar</div>	<div>4</div>	<div>Lourdes Peña Linares</div>	<div>lourdes@gmail.com</div>	<div>973636622</div>

A continuación empezaremos creando la Base de datos *bd_ejemplo*:

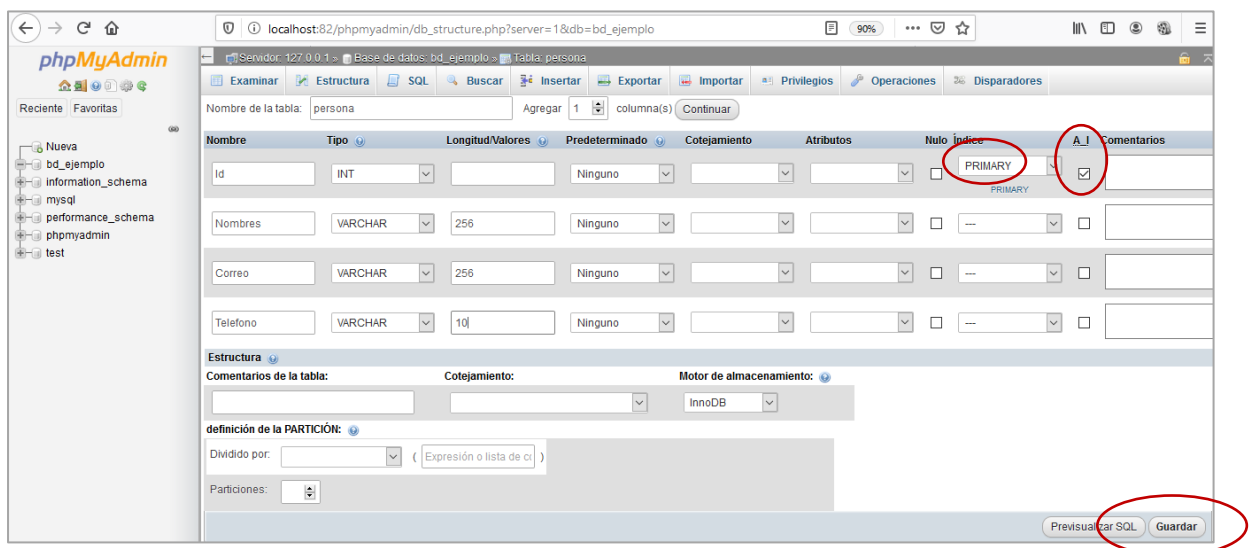
Crear base de datos



Enseguida creamos la tabla llamada persona:



Ingresamos los valores correspondientes a la tabla. En primary aparece una ventana, simplemente le damos continuar y finalmente guardamos la tabla creada. El A_I es autoincremento para que se llene sólo.



A continuación insertamos los registros a la tabla (esto es mediante un formulario), damos clic en Insertar y llenamos los formularios. En Id, lo dejamos en blanco, pues anteriormente le dimos autocompletar, entonces se incrementará automáticamente, observamos que está por defecto seleccionado Ignorar, es decir sólo se considera sólo el primer registro que se llenará y luego continuar:

Columna	Tipo	Función	Nulo	Valor
Id	int(10)		<input type="checkbox"/>	
Nombres	varchar(50)		<input type="checkbox"/>	Torres Peredo Perez
Correo	varchar(50)		<input type="checkbox"/>	peredo@gmail.com
Telefono	varchar(10)		<input type="checkbox"/>	983252459

☒ Ignorar

A continuación se muestra el registro creado:

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Disparadores

✓ 1 fila insertada.
La Id de la fila insertada es: 1

```
INSERT INTO `persona` (`Id`, `Nombres`, `Correo`, `Telefono`) VALUES (NULL, 'Torres Peredo Perez', 'peredo@gmail.com', '983252459');
```

Ejecutar la(s) consulta(s) SQL en la tabla bd_ejemplo.persona:

```
1 INSERT INTO `persona` (`Id`, `Nombres`, `Correo`, `Telefono`) VALUES (NULL, 'Torres Peredo Perez', 'peredo@gmail.com', '983252459');
```

Columnas

Id
Nombres
Correo
Telefono

Si le damos clic en la tabla persona, observamos el registro creado:

← → ↺ 🏠 localhost:82/phpmyadmin/sql.php?server=1&db=bd_ejemplo&table=persona&pos=0

Servidor: 127.0.0.1 » Base de datos: bd_ejemplo » Tabla: persona

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios

✓ Mostrando filas 0 - 0 (total de 1, La consulta tardó 0,0333 segundos.)

```
SELECT * FROM `persona`
```

☐ Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla

+ Opciones

	Id	Nombres	Correo	Telefono
<input type="checkbox"/> Editor <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	1	Torres Peredo Perez	peredo@gmail.com	983252459

☐ Seleccionar todo Para los elementos que están marcados: ☐ Editor ☐ Copiar ☐ Borrar ☐ Exportar

☐ Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla

Operaciones sobre los resultados de la consulta

Como vamos a crear 4 registros, a continuación damos clic en Insertar y en *continuar inserción* colocaremos 3 filas y luego continuamos con el ingreso de registros que nos en el formulario, teniendo en cuenta que ahora está desactivado la opción ignorar y finalmente continuar:

Sección 127.0.0.1 - Base de datos: bd_Ejemplo - Tabla: persona

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones

Columna	Tipo	Función	Nulo	Valor
Id	int(10)			
Nombres	varchar(50)			Torres Vargas Torres
Correo	varchar(50)			torres@gmail.com
Telefono	varchar(10)			900567347

☐ Ignorar

Columna	Tipo	Función	Nulo	Valor
Id	int(10)			
Nombres	varchar(50)			Mary Luz Milagros Vargas
Correo	varchar(50)			maryluz@gmail.com
Telefono	varchar(10)			3456874

☐ Ignorar

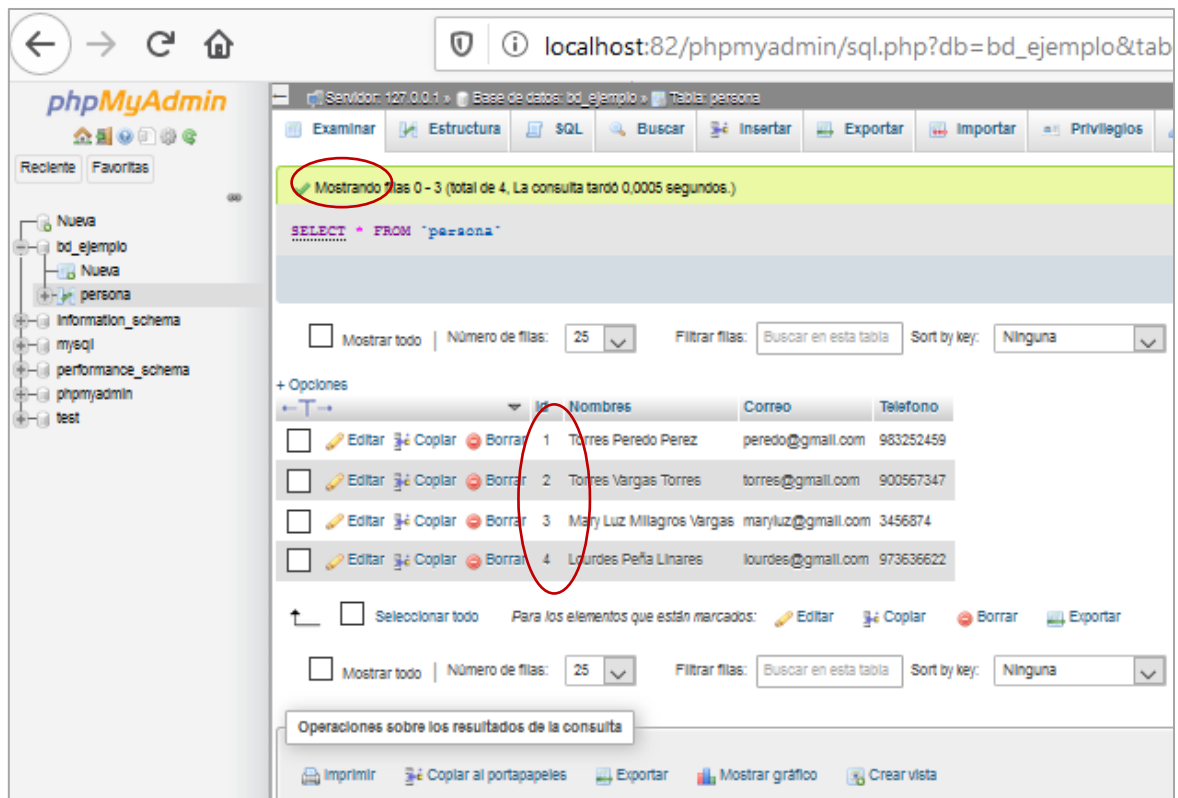
Columna	Tipo	Función	Nulo	Valor
Id	int(10)			
Nombres	varchar(50)			Lourdes Peña Linares
Correo	varchar(50)			lourdes@gmail.com
Telefono	varchar(10)			973636622

Insertar como una nueva fila y luego Volver

Previsualizar SQL Reiniciar Continuar

Continuar inserción con 3 filas

En examinar podemos ver la tabla y los registros creados, además el Id (autoincremento) se incrementa de uno en uno, obteniendo finalmente nuestra base de datos, la tabla y los registros:

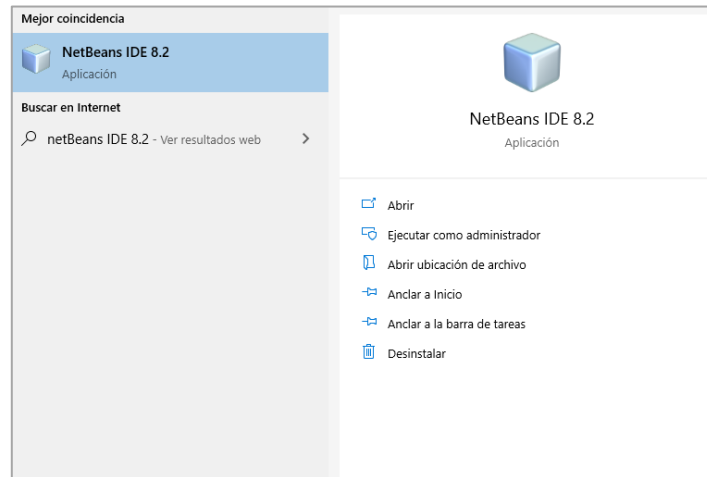


Finalmente hemos creado la base de datos, con la tabla y los registros, tal y como se había establecido:

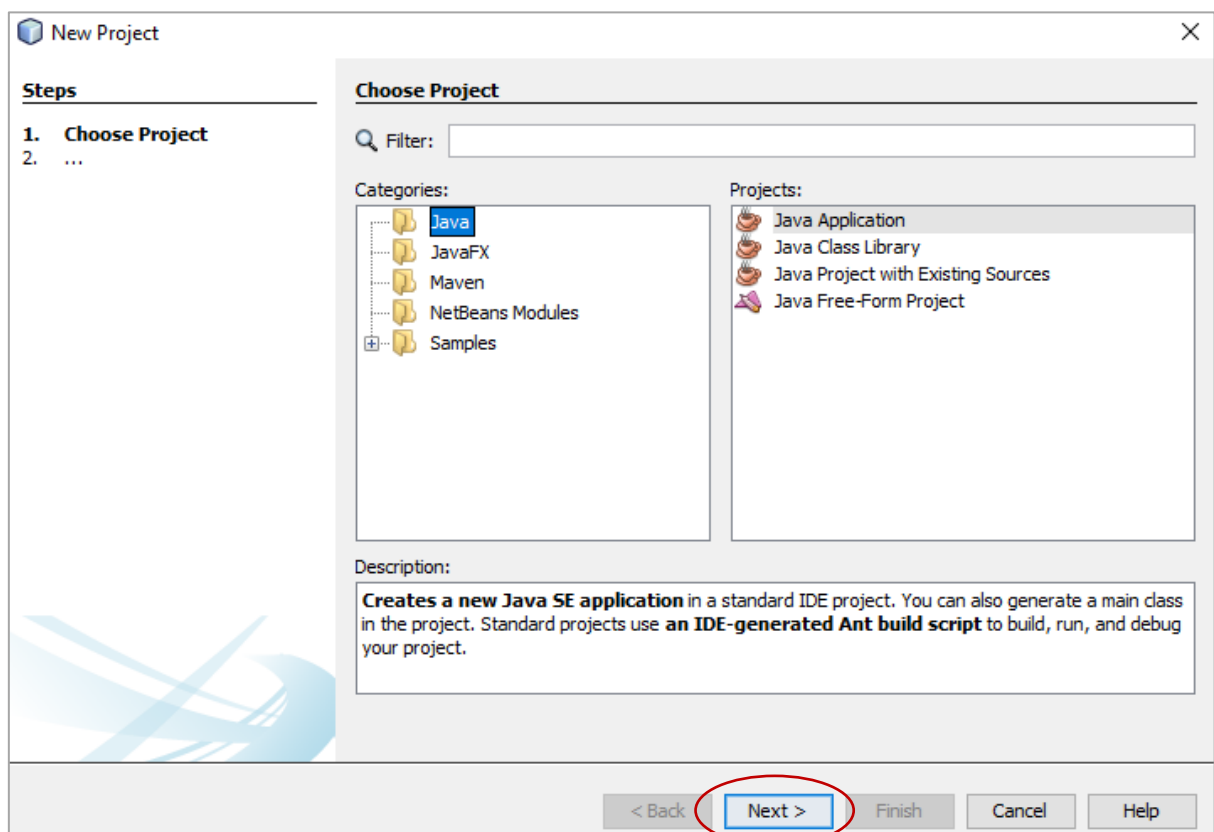
+ Opciones				
<input type="checkbox"/>	Editar	Copiar	Borrar	
Id	Nombres	Correo	Telefono	
1	Torres Peredo Perez	peredo@gmail.com	983252459	
2	Torres Vargas Torres	torres@gmail.com	900567347	
3	Mary Luz Milagros Vargas	maryluz@gmail.com	3456874	
4	Lourdes Peña Linares	lourdes@gmail.com	973636622	
<input type="checkbox"/> Seleccionar todo Para los elementos que están marcados: <input type="checkbox"/> Editar <input type="checkbox"/> Copiar				

CRUD en Java Escritorio MVC – Listar

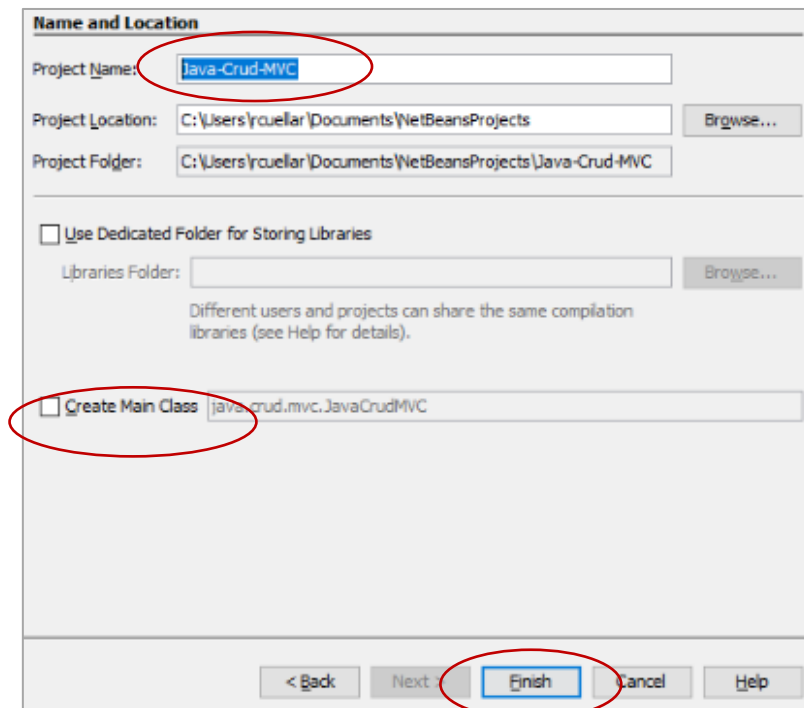
A continuación empezamos a crear el *CRUD en Java Escritorio MVC – Listar*, abrimos NetBeans IDE 8.2 (con el JDK) instalado previamente:



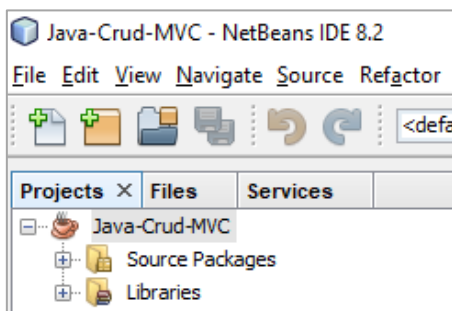
Crearemos un nuevo proyecto File/New Project y le damos Next:



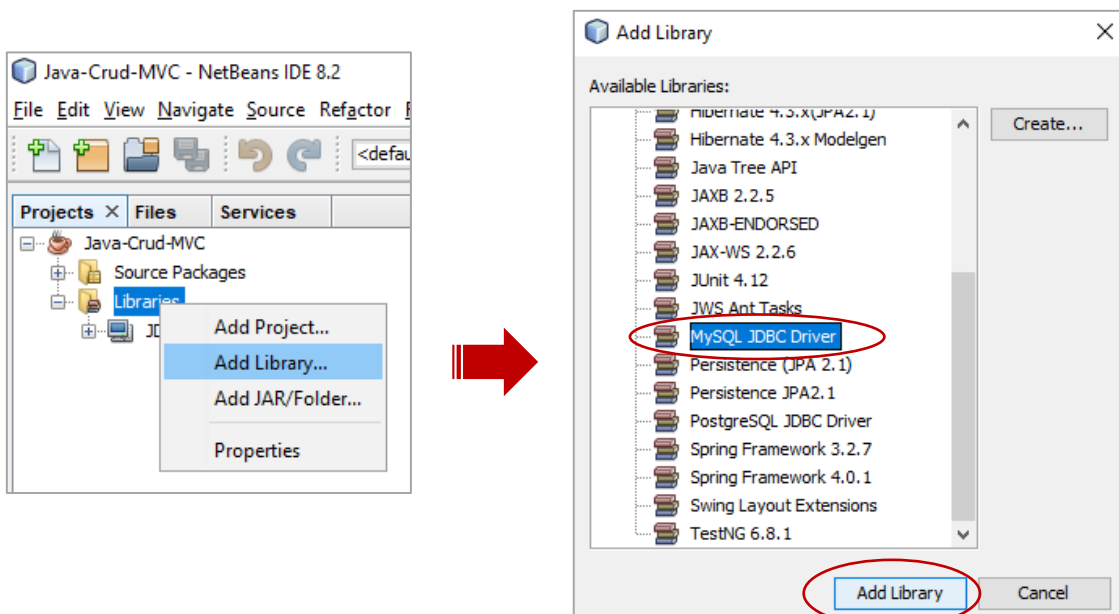
Escribimos el nombre del proyecto: Java-Crud-MVC, no vamos a crear la clase principal (desactivamos Create Main Class), luego finalizar:

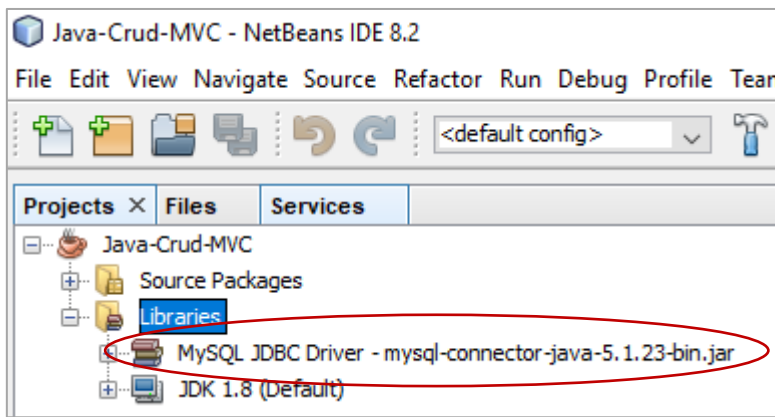


A continuación nos muestra el proyecto creado:

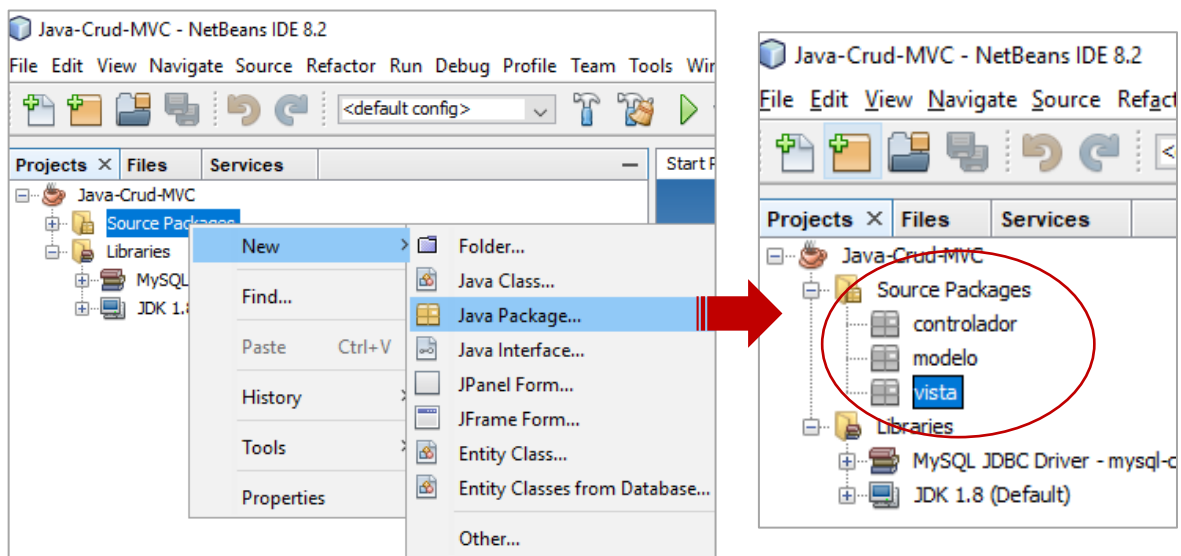


Agregaremos la librería correspondiente para conectar a nuestra base de datos el **MySQL JDBC Driver**, dando clic derecho en Libraries y seleccionando Add library:

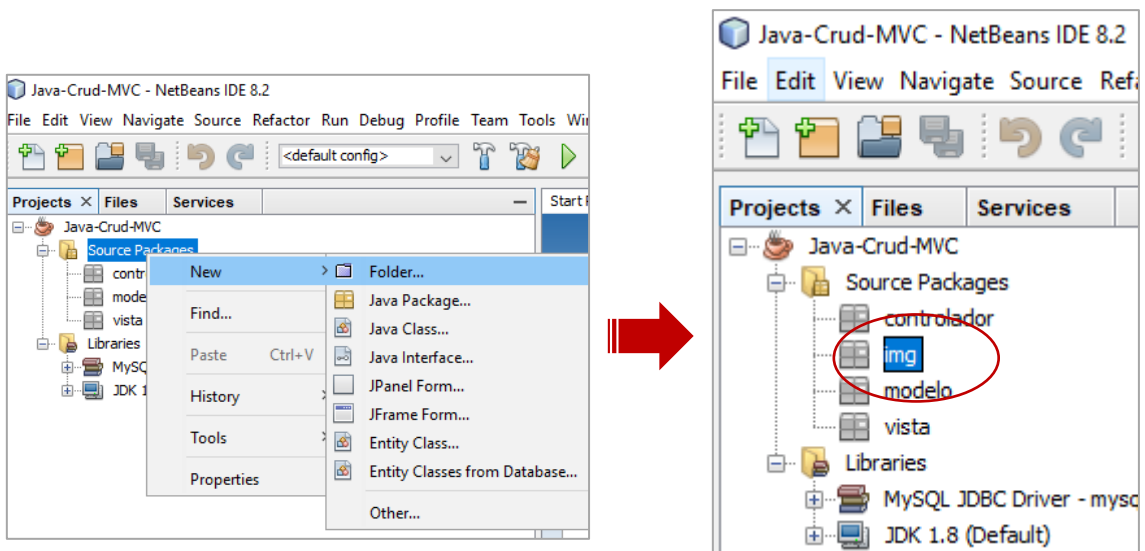




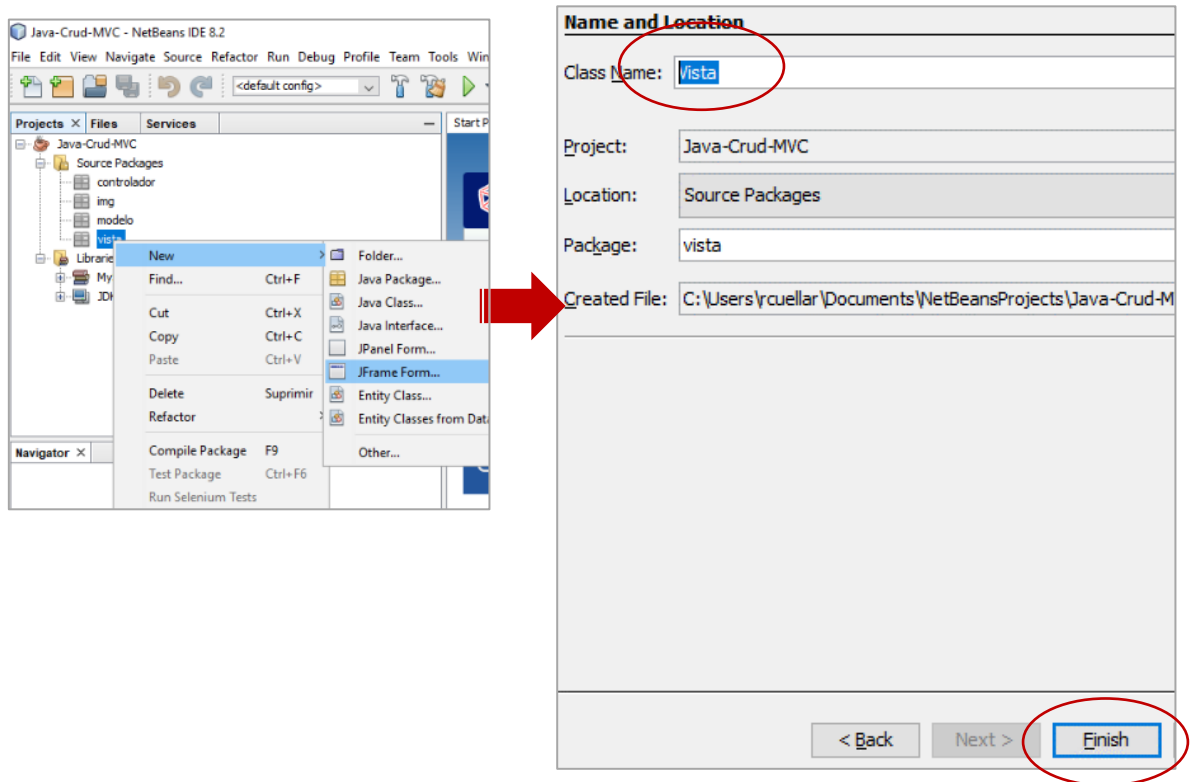
A continuación creamos la estructura de nuestro proyecto, damos clic derecho en Source packages/New/Java Package y crearemos tres paquetes: **modelo**, **controlador**, **vista**.



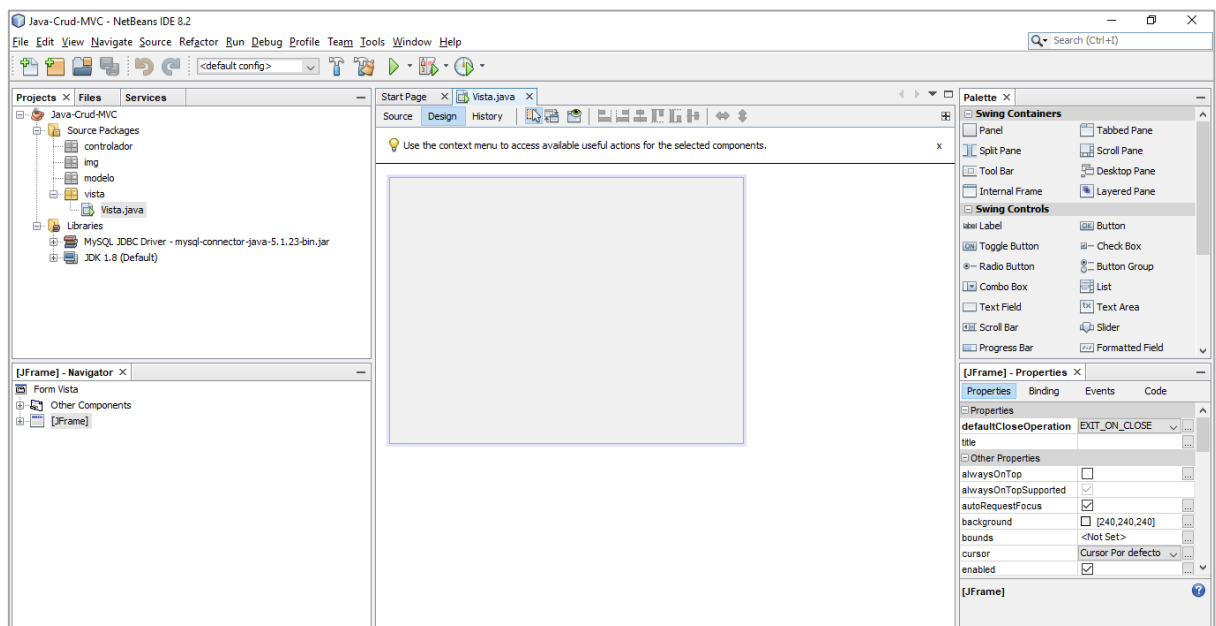
Seguidamente creamos un folder: **img**



En la parte de vista agregamos un nuevo JFrame Form... llamada **Vista** y luego finalizar:



A continuación tenemos el formulario:



Diseñamos el formulario, según lo indicado:

Datos

ID: Guardar

Nombres: Listar

Correo: Editar Ok

Teléfono: Eliminar

Detalles

ID	NOMBRES	CORREO	TELEFONO
----	---------	--------	----------

Donde los nombres de las variables de las cajas de texto:

txtId

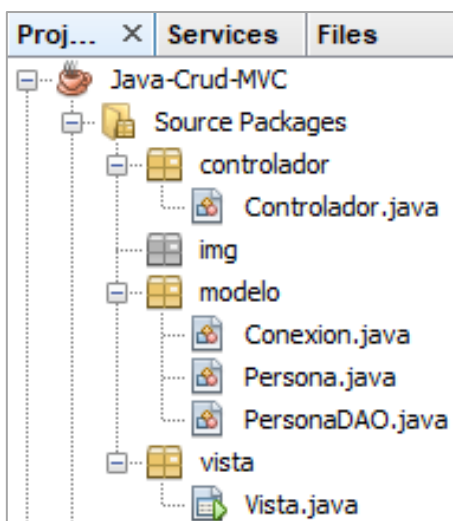
txtNombres (en el video está con txtNom, el cual no lo hemos considerado)

txtCorreo

txtTelefono

btnActualizar

A continuación, ingresamos los códigos:



Controlador.java:

```
package controlador;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.List;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import modelo.Persona;
import modelo.PersonaDAO;
import vista.Vista;

public class Controlador implements ActionListener{

    PersonaDAO dao=new PersonaDAO();
    Persona p=new Persona();
    Vista vista=new Vista();
    DefaultTableModel modelo=new DefaultTableModel();

    public Controlador(Vista v){
        this.vista=v;
        this.vista.btnListar.addActionListener(this);
    }

    @Override
    public void actionPerformed(ActionEvent e){
        if(e.getSource()==vista.btnListar){
            listar(vista.tabla);
        }
    }

    public void listar(JTable tabla){
        modelo=(DefaultTableModel)tabla.getModel();
        List<Persona>lista=dao.listar();
        Object[]object=new Object[4];
        for (int i=0; i < lista.size(); i++){
            object[0]=lista.get(i).getId();
            object[1]=lista.get(i).getNom();
            object[2]=lista.get(i).getCorreo();
            object[3]=lista.get(i).getTel();
            modelo.addRow(object);
        }
        vista.tabla.setModel(modelo);
    }
}
```

Conexion.java:

```
package modelo;

import java.sql.Connection;
import java.sql.DriverManager;

public class Conexion {
    Connection con;
    public Connection getConnection () {
        String url="jdbc:mysql://localhost:3306/bd_ejemplo";
        String user="root";
        String pass="";
        try {
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection(url,user,pass);
        } catch (Exception e){
        }
        return con;
    }
}
```

Persona.java:

```
package modelo;

public class Persona {
    int id;
    String nom;
    String correo;
    String tel;

    public Persona(){
    }

    public Persona(int id, String nom, String correo, String tel){
        this.id=id;
        this.nom=nom;
        this.correo=correo;
        this.tel=tel;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getNom() {
        return nom;
    }

    public void setNom(String nom) {
        this.nom = nom;
    }
}
```

```

[-] public String getCorreo() {
    return correo;
}

[-] public void setCorreo(String correo) {
    this.correo = correo;
}

[-] public String getTel() {
    return tel;
}

[-] public void setTel(String tel) {
    this.tel = tel;
}
}

```

```

public void setId(int id) {
    this.id = id;
}

```

```

public String getNom() {
    return nom;
}

```

```

public void setNom(String nom) {

```

PersonaDAO.java:

```

package modelo;

```

```

[-] import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.List;

public class PersonaDAO {
    Conexion conectar=new Conexion();
    Connection con;
    PreparedStatement ps;
    ResultSet rs;

    public List listar(){
        List<Persona>datos=new ArrayList<>();
        String sql="select*from persona";
        try {
            con=conectar.getConnection();
            ps=con.prepareStatement(sql);
            rs=ps.executeQuery();
            while (rs.next()){
                Persona p=new Persona();
                p.setId(rs.getInt(1));
                p.setNom(rs.getString(2));
                p.setCorreo(rs.getString(3));
                p.setTel(rs.getString(4));
                datos.add(p);
            }
        } catch (Exception e){
        }
        return datos;
    }
}

```


Vista.java:

```
package vista;

import controlador.Controlador;

public class Vista extends javax.swing.JFrame {

    public Vista() {
        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code

    private void txtIdActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnListarActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void txtNombresActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        Vista v=new Vista();
        Controlador c=new Controlador(v);
        v.setVisible(true);
        v.setLocationRelativeTo(null);
    }

    // Variables declaration - do not modify
    public javax.swing.JButton btnEditar;
    public javax.swing.JButton btnEliminar;
    public javax.swing.JButton btnGuardar;
    public javax.swing.JButton btnListar;
    public javax.swing.JButton btnOk;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JTable jTable1;
    public javax.swing.JTable tabla;
    public javax.swing.JTextField txtCorreo;
    public javax.swing.JTextField txtId;
    public javax.swing.JTextField txtNombres;
    public javax.swing.JTextField txtTelefono;
    // End of variables declaration

    private void setLocationRelativeTo(Object object) {
        throw new UnsupportedOperationException("Not supported yet."); //To change
    }
}
```

A continuación ejecutamos y presionamos el botón **Listar**:

The screenshot shows a web application window with a title bar. Inside, there's a section titled "Datos" with four input fields: "ID:", "Nombres:", "Correo:", and "Teléfono:". To the right of these fields are five buttons: "Guardar", "Listar", "Editar", "Ok", and "Eliminar". The "Listar" button is circled in red. Below the "Datos" section is a section titled "Detalles" which contains a table with four columns: "ID", "NOMBRES", "CORREO", and "TELEFONO". The table is currently empty.

A continuación se presenta la *lista*:

Detalles

ID	NOMBRES	CORREO	TELEFONO
1	Torres Peredo Pe...	peredo@gmail.com	983252459
2	Torres Vargas Tor...	torres@gmail.com	900567347
3	Mary Luz Milagros ...	maryluz@gmail.co...	3456874
4	Lourdes Peña Lin...	lourdes@gmail.co...	973636622

+ Opciones

			Id	Nombres	Correo	Telefono	
<input type="checkbox"/>	Editar	Copiar	Borrar	1	Torres Peredo Perez	peredo@gmail.com	983252459
<input type="checkbox"/>	Editar	Copiar	Borrar	2	Torres Vargas Torres	torres@gmail.com	900567347
<input type="checkbox"/>	Editar	Copiar	Borrar	3	Mary Luz Milagros Vargas	maryluz@gmail.com	3456874
<input type="checkbox"/>	Editar	Copiar	Borrar	4	Lourdes Peña Linares	lourdes@gmail.com	973636622

☐ Seleccionar todo

Para los elementos que están marcados:

Editar

Copiar

2. CRUD en Java Escritorio MVC – Agregar

Continuamos con el *método agregar* de la clase *PersonaDAO.java*:

```
        p.setNom(rs.getString(2));
        p.setCorreo(rs.getString(3));
        p.setTel(rs.getString(4));
        datos.add(p);
    }
} catch (Exception e) {
}
return datos;
}

public int agregar(Persona p) {
    String sql="insert into persona (nombres,Correo,Telefono) values(?,?,?)";
    try {
        con=conectar.getConnection();
        ps=con.prepareStatement(sql);
        ps.setString(1,p.getNom());
        ps.setString(2,p.getCorreo());
        ps.setString(3,p.getTel());
        ps.executeUpdate();
    } catch (Exception e) {
    }
    return 1;
}
```

Controlador.java:

Método guardar

```
package controlador;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.List;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import modelo.Persona;
import modelo.PersonaDAO;
import vista.Vista;

public class Controlador implements ActionListener{

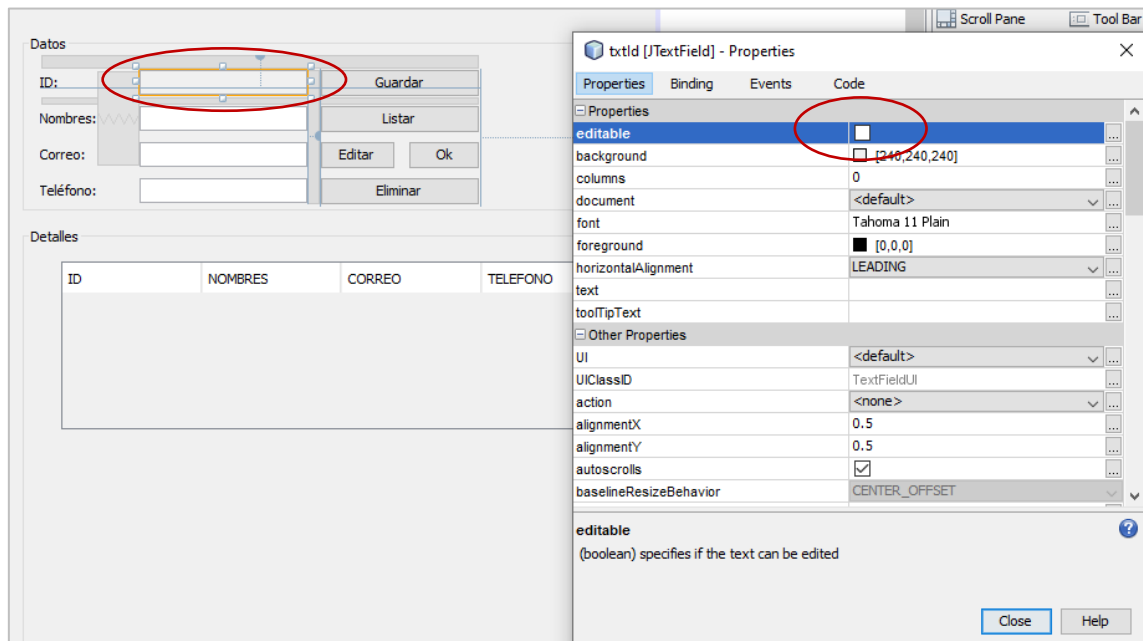
    PersonaDAO dao=new PersonaDAO();
    Persona p=new Persona();
    Vista vista=new Vista();
    DefaultTableModel modelo=new DefaultTableModel();

    public Controlador(Vista v){
        this.vista=v;
        this.vista.btnListar.addActionListener(this);
        this.vista.btnGuardar.addActionListener(this);
    }

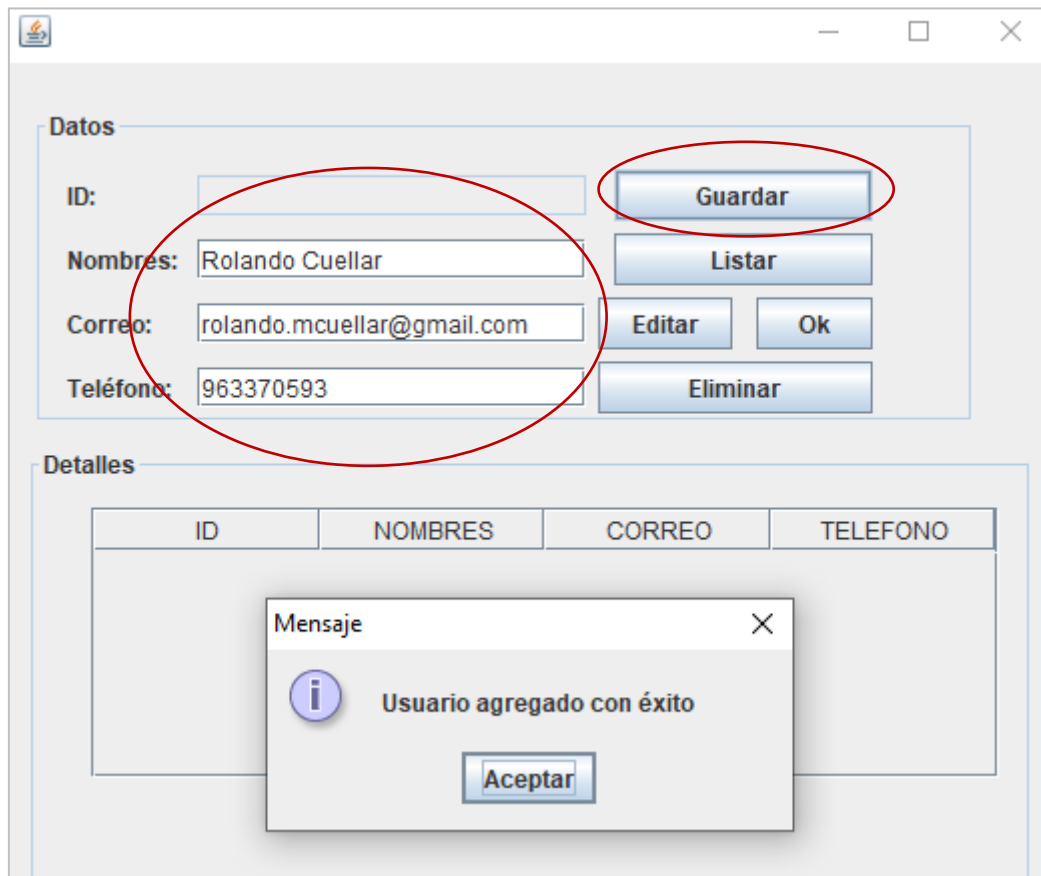
    @Override
    public void actionPerformed(ActionEvent e){
        if(e.getSource()==vista.btnListar){
            listar(vista.tabla);
        }
        if (e.getSource()==vista.btnGuardar){
            agregar();
        }
    }

    public void agregar () {
        String nom=vista.txtNombres.getText();
        String correo=vista.txtCorreo.getText();
        String tel=vista.txtTelefono.getText();
        p.setNom(nom);
        p.setCorreo(correo);
        p.setTel(tel);
        int r=dao.agregar(p);
        if (r==1){
            JOptionPane.showMessageDialog(vista,"Usuario agregado con éxito");
        }else{
            JOptionPane.showMessageDialog(vista,"Error");
        }
    }
}
```

Ahora el campo Id no debe ser editable, para ello, vamos Vista, en la parte de diseño, le damos clic derecho, propiedades y desactivamos la opción editable:



Ejecutamos, y agregamos un nuevo usuario, y luego guardar:



A continuación presionamos en el botón listar, y observamos que se ha agregado sin problemas al nuevo usuario:

The screenshot shows a web application window with a title bar. Inside, there are two main sections: 'Datos' and 'Detalles'.

Datos Section:

- ID:** An empty text input field.
- Nombres:** A text input field containing 'Rolando Cuellar'.
- Correo:** A text input field containing 'rolando.mcuellar@gmail.com'.
- Teléfono:** A text input field containing '963370593'.
- Buttons:** 'Guardar' (circled in red), 'Listar' (circled in red), 'Editar', 'Ok', and 'Eliminar'.

Detalles Section:

A table with 4 columns: ID, NOMBRES, CORREO, and TELEFONO. It contains 5 rows of data. The 5th row is circled in red.

ID	NOMBRES	CORREO	TELEFONO
1	Torres Peredo Pe...	peredo@gmail.com	983252459
2	Torres Vargas Tor...	torres@gmail.com	900567347
3	Mary Luz Milagros ...	maryluz@gmail.co...	3456874
4	Lourdes Peña Lin...	lourdes@gmail.co...	973636622
5	Rolando Cuellar	rolando.mcuellar...	963370593

3. CRUD en Java Escritorio MVC – Actualizar

Persona.DAO.java

Método Editar y Actualizar

```
    } catch (Exception e) {  
    }  
    return 1;  
}  
  
public int Actualizar (Persona p) {  
    int r=0;  
    String sql="update persona set Nombres=?, correo=?, Telefono=? Where Id=?";  
    try{  
        con=conectar.getConnection();  
        ps=con.prepareStatement(sql);  
        ps.setString(1,p.getNom());  
        ps.setString(2,p.getCorreo());  
        ps.setString(3,p.getTel());  
        ps.setInt(4,p.getId());  
        r=ps.executeUpdate();  
        if (r==1){  
            return 1;  
        }else{  
            return 0;  
        }  
    } catch (Exception e) {  
    }  
    return r;  
}
```

Controlador.java

Mg. Ing. Raúl Fernández Bejarano

```

package controlador;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.List;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import modelo.Persona;
import modelo.PersonaDAO;
import vista.Vista;

public class Controlador implements ActionListener{

    PersonaDAO dao=new PersonaDAO();
    Persona p=new Persona();
    Vista vista=new Vista();
    DefaultTableModel modelo=new DefaultTableModel();

    public Controlador(Vista v){
        this.vista=v;
        this.vista.btnListar.addActionListener(this);
        this.vista.btnGuardar.addActionListener(this);
        this.vista.btnEditar.addActionListener(this);
        this.vista.btnActualizar.addActionListener(this);
    }

    @Override
    public void actionPerformed(ActionEvent e){
        if(e.getSource()==vista.btnListar){
            listar(vista.tabla);
        }
        if (e.getSource()==vista.btnGuardar){
            agregar();
        }

        if (e.getSource()==vista.btnEditar){
            int fila=vista.tabla.getSelectedRow();
            if (fila==-1){
                JOptionPane.showMessageDialog(vista, "Debe seleccionar una fila");
            }
            else{
                int id=Integer.parseInt((String)vista.tabla.getValueAt(fila,0).toString());
                String nom=(String)vista.tabla.getValueAt(fila,1);
                String correo=(String)vista.tabla.getValueAt(fila,2);
                String tel=(String)vista.tabla.getValueAt(fila,3);
                vista.txtId.setText("" + id);
                vista.txtNombres.setText(nom);
                vista.txtCorreo.setText(correo);
                vista.txtTelefono.setText(tel);
            }
        }
        if (e.getSource()==vista.btnActualizar){
            Actualizar();
        }
    }

    public void Actualizar(){

```



```

int id=Integer.parseInt(vista.txtId.getText());
String nom=vista.txtNombres.getText();
String correo=vista.txtCorreo.getText();
String tel=vista.txtTelefono.getText();
p.setId(id);
p.setNom(nom);
p.setCorreo(correo);
p.setTel(tel);
int r=dao.Actualizar(p);
if (r==1){
    JOptionPane.showMessageDialog(vista,"Usuario actualizado con éxito");
}else{
    JOptionPane.showMessageDialog(vista,"Error");
}
}

```

```

public void agregar () {
    String nom=vista.txtNombres.getText();
    String correo=vista.txtCorreo.getText();
    String tel=vista.txtTelefono.getText();
    p.setNom(nom);
    p.setCorreo(correo);
    p.setTel(tel);
    int r=dao.agregar(p);
    if (r==1){

```

Datos

ID:

Nombres:

Correo:

Teléfono:

Detalles

ID	NOMBRES	CORREO	TELEFONO
1	Torres Peredo Pe...	peredo@gmail.com	983252459
2	Torres Vargas Tor...	torres@gmail.com	990567347
3			56874
4			73636622
5			53370593

Mensaje

Debe seleccionar una fila

Editamos:

Mg. Ing. Raúl Fernández Bejarano

Datos

ID:

Nombres:

Correo:

Teléfono:

Detalles

ID	NOMBRES	CORREO	TELEFONO
1	Torres Peredo Pe...	peredo@gmail.com	983252459
2	Torres Vargas Tor...	torres@gmail.com	900567347
3	Mary Luz Milagros ...	maryluz@gmail.co...	3456874
4	Lourdes Peña Lin...	lourdes@gmail.co...	973636622
5	Rolando Cuellar	rolando.mcuellar...	963370593

Ahora actualizamos dando clic en el botón ok:

Datos

ID:

Nombres:

Correo:

Teléfono:

Detalles

ID	NOMBRES	CORREO	TELEFONO
1	Torres Peredo Pe...	peredo@gmail.c...	983252459
2	Torres Vargas To...	torres@gmail.com	900567347
3			6874
4			636622
5			370593
6			636622
7			636622

Mensaje

Usuario actualizado con éxito

Listamos nuevamente y observamos que se ya se ha actualizó:

Detalles

ID	NOMBRES	CORREO	TELEFONO
1	Torres Peredo Pe...	peredo@gmail.c...	983252459
2	Torres Vargas To...	torres@gmail.com	900567347
3	Mary Luz Milagros...	maryluz@gmail.c...	3456874
4	Lourdes Peña Li...	lourdes@gmail.c...	973636622
5	Cuellar	rolando.mcuellar...	963370593
6	Peña Linares	lourdes@gmail.c...	973636622
7	Linares	lourdes@gmail.c...	973636622

Método LimpiarTabla

```

        vista.txtNombres.setText(nom);
        vista.txtCorreo.setText(correo);
        vista.txtTelefono.setText(tel);
    }
}
if (e.getSource()==vista.btnActualizar) {
    Actualizar();
}
}

void limpiarTabla() {
    for (int i=0; i<vista.tabla.getRowCount(); i++) {
        modelo.removeRow(i);
        i=i-1;
    }
}

public void Actualizar() {
    int id=Integer.parseInt(vista.txtId.getText());
    String nom=vista.txtNombres.getText();
    String correo=vista.txtCorreo.getText();
    String tel=vista.txtTelefono.getText();
    p.setId(id);
    p.setNom(nom);
}

```

```

if (e.getSource() == vista.btnListar) {
    limpiarTabla();
    listar(vista.tabla);
}
if (e.getSource() == vista.btnGuardar) {
    agregar();
    limpiarTabla();
    listar(vista.tabla);
}
if (e.getSource() == vista.btnEditar) {
    int fila = vista.tabla.getSelectedRow();
    if (fila == -1) {
        JOptionPane.showMessageDialog(vista, "Debe seleccionar una fila");
    }
    else {
        int id = Integer.parseInt((String) vista.tabla.getValueAt(fila, 0).toString());
        String nom = (String) vista.tabla.getValueAt(fila, 1);
        String correo = (String) vista.tabla.getValueAt(fila, 2);
        String tel = (String) vista.tabla.getValueAt(fila, 3);
        vista.txtId.setText("" + id);
        vista.txtNombres.setText(nom);
        vista.txtCorreo.setText(correo);
        vista.txtTelefono.setText(tel);
    }
}
if (e.getSource() == vista.btnActualizar) {
    actualizar();
    limpiarTabla();
    listar(vista.tabla);
}
}

```

Por último, ejecutamos:

Primero listamos, luego editamos:

Datos

ID: 5

Nombres: Rolando Marcos Cuellar

Correo: rolando.mcuellar@gmail.com

Teléfono: 963370593

Guardar

Listar

Editar

Ok

Eliminar

Detalles

ID	NOMBRES	CORREO	TELEFONO
1	Torres Peredo Pe...	peredo@gmail.com	983252459
2	Torres Vargas Tor...	torres@gmail.com	900567347
3	Mary Luz Milagros ...	maryluz@gmail.co...	3456874
4	Lourdes Peña Lin...	lourdes@gmail.co...	973636622
5	Cuellar	rolando.mcuellar...	963370593
6	Peña Linares	lourdes@gmail.co...	973636622
7	Linares	lourdes@gmail.co...	973636622

Y finalmente actualizamos (botón ok):



Datos

ID:

Nombres:

Correo:

Teléfono:

Detalles

ID	NOMBRES	CORREO	TELEFONO
1	Torres Peredo Pe...	peredo@gmail.com	983252459
2	Torres Vargas Tor...	torres@gmail.com	900567347
3	Mary Luz Milagros ...	maryluz@gmail.co...	3456874
4	Lourdes Peña Lin...	lourdes@gmail.co...	973636622
5	Rolando Marcos ...	rolando.mcuellar...	963370593
6	Peña Linares	lourdes@gmail.co...	973636622
7	Linares	lourdes@gmail.co...	973636622

4. CRUD en Java Escritorio MVC – Eliminar

El método Eliminar

Clase personaDAO.java

```
        }else{  
            return 0;  
        }  
    } catch (Exception e) {  
    }  
    return r;  
}  
  
public void delete (int id) {  
    String sql="delete from persona where Id="+id;  
    try {  
        con=conectar.getConnection();  
        ps=con.prepareStatement(sql);  
        ps.executeUpdate();  
    } catch (Exception e) {  
    }  
}
```

Controlador.java

```
package controlador;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.List;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import modelo.Persona;
import modelo.PersonaDAO;
import vista.Vista;

public class Controlador implements ActionListener{

    PersonaDAO dao=new PersonaDAO();
    Persona p=new Persona();
    Vista vista=new Vista();
    DefaultTableModel modelo=new DefaultTableModel();

    public Controlador(Vista v){
        this.vista=v;
        this.vista.btnListar.addActionListener(this);
        this.vista.btnGuardar.addActionListener(this);
        this.vista.btnEditar.addActionListener(this);
        this.vista.btnActualizar.addActionListener(this);
        this.vista.btnEliminar.addActionListener(this);
        listar(vista.tabla);
    }

    if (e.getSource()== vista.btnActualizar){
        Actualizar();
        limpiarTabla();
        listar(vista.tabla);
    }

    if(e.getSource()== vista.btnEliminar){
        delete();
        limpiarTabla();
        listar(vista.tabla);
    }

    public void delete(){
        int fila=vista.tabla.getSelectedRow();
        if (fila == -1){
            JOptionPane.showMessageDialog(vista,"Debe seleccionar un usuario");
        }else {
            int id=Integer.parseInt((String)vista.tabla.getValueAt(fila,0).toString());
            dao.delete(id);
            JOptionPane.showMessageDialog(vista,"Usuario eliminado");
        }
    }

    void limpiarTabla(){
        for (int i=0; i < vista.tabla.getRowCount();i++){
            modelo.removeRow(i);
        }
    }
}
```

A continuación ejecutamos, seleccionamos (El usuario de la fila 5) y finalmente eliminamos:

Datos

ID: **Guardar**

Nombres: **Listar**

Correo: **Editar** **Ok**

Teléfono: **Eliminar**

Detalles

ID	NOMBRES	CORREO	TELEFONO
1	Torres Peredo Pe...	peredo@gmail.com	983252459
2	Torres Vargas Tor...	torres@gmail.com	900567347
3	Mary Luz Milagros ...	maryluz@gmail.co...	3456874
4	Lourdes Peña Lin...	lourdes@gmail.co...	973636622
5	Rolando Marcos ...	rolando.mcuellar...	963370593
6	Peña Linares	lourdes@gmail.co...	973636622

Aparece el mensaje Usuario eliminado:

Datos

ID: **Guardar**

Nombres: **Listar**

Correo: **Editar** **Ok**

Teléfono: **Eliminar**

Detalles

ID	NOMBRES	CORREO	TELEFONO
1	Torres Peredo Pe...	peredo@gmail.com	983252459
2	Torres Vargas Tor...	torres@gmail.com	900567347
3	Mary Luz Milagros ...	maryluz@gmail.co...	3456874
4	Lourdes Peña Lin...	lourdes@gmail.co...	973636622
5	Rolando Marcos ...	rolando.mcuellar...	963370593
6	Peña Linares	lourdes@gmail.co...	973636622

Mensaje

Usuario eliminado

Aceptar

Le damos aceptar y se actualiza con el usuario eliminado (fila 5):

Detalles

ID	NOMBRES	CORREO	TELEFONO
1	Torres Peredo Pe...	peredo@gmail.com	983252459
2	Torres Vargas Tor...	torres@gmail.com	900567347
3	Mary Luz Milagros ...	maryluz@gmail.co...	3456874
4	Lourdes Peña Lin...	lourdes@gmail.co...	973636622
6	Peña Linares	lourdes@gmail.co...	973636622