

UNIVERSIDAD DE LAS FUERZAS ARMADAS-ESPE SEDE SANTO DOMINGO
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN - DCCO-SS
CARRERA DE INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

PERIODO : Abril 2023 – Octubre 2023

ASIGNATURA : POO

TEMA : Evaluación 3

NOMBRES : Kelvin Quezada

NIVEL-PARALELO : Segundo “A”

DOCENTE : Ing. Cevallos Farias Javier Moyota

FECHA DE ENTREGA : 14/08/2023



SANTO DOMINGO - ECUADOR
2023

Tema:

Paso Para crear nuestra Interfaz de registro de Estudiante

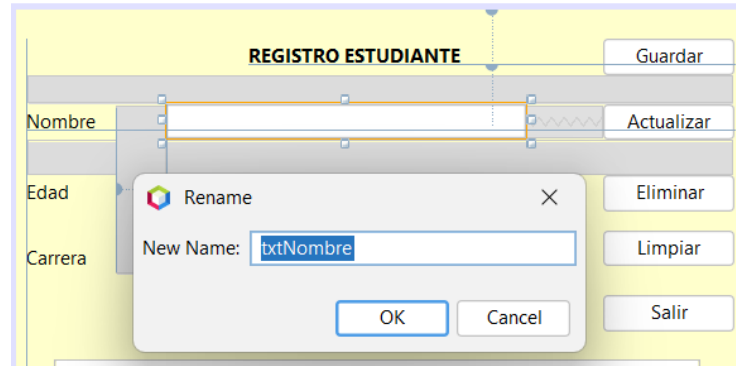
Paso Uno

Creo una Clase llamada RegistroProfesores con el JFrame para crear un panel donde colocaremos tres Label y creamos cinco botones, y una tabla para conocer nuestros resultados.



Paso Segundo

Le daremos a dar nombres específicos a la caja de textos, botones y a la tabla como esta imagen ya que la caja de textos como los botones y la tabla nos ayuda mucho a la configuración de nuestro programa y pueda funcionar correctamente.



Paso Tercer

Nos dirigimos al soucer y comendamos a configuras nuestras variables que es DB db para ser referencia a la base de datos creamos un boolea para almacenar el estado disponible de la base de datos, creo una conexión de mongo usando el método que vamos a crear que es crearConexion, utilizamos un BasicDBObject backup1 para declarar un respaldo.

```
public class RegistroEstudiante extends javax.swing.JFrame {
    // Variable para la conexión a la base de datos
    DB db;
    // Inicialización de la conexión MongoDB
    MongoClient mongo=crearConexion();
    // Variable para indicar disponibilidad de la base de datos
    boolean bdavilable;
    // Objeto para respaldos (no utilizado en este fragmento)
    BasicDBObject backup;
```

Paso Cuatro

Creamos nuestro constructor RegistroEstudiante para llamar nuestros datos utilizamos un setTitle para crear un título a nuestro panel vamos a utilizar una condicional para verificar si la conexión del mongo fue exitoso, y creamos una base de datos llamado Prueba del servidor de MongoDB, utilizamos el initComponents para inicializar nuestro campo y llamaremos el método de mostrar Estudiante para conocer la lista de los Estudiante.

```
// Verificar si la conexión MongoDB se estableció
public RegistroEstudiante() {
    if(mongo!=null){
        // Asignar la base de datos "Prueba" a la variable db
        db=mongo.getDB("Prueba");
    }
    // Inicializar los componentes de la interfaz gráfica
    initComponents();
    // Mostrar los datos de estudiantes en la interfaz
    mostrar();
}
// Método estático para crear una conexión MongoDB
```

Paso Quinto

Creamos nuestro método llamado conexión para conectar al servidor de mongodb con el localhost 27017 y le diremos que retorne a la instancia de mongo del cliente MongoDB.

```
}
// Método estático para crear una conexión MongoDB
public static MongoClient crearConexion(){
    // Crear una instancia de MongoClient conectando a localhost en el puerto 27017
    MongoClient mongo =new MongoClient("localhost",27017);
    // Devolver la instancia de MongoClient
    return mongo;
}
```

Paso Sexto

Creamos otro método para mostrar los datos del estudiante donde vamos a llamar el título de nuestro encabezado, utilizaremos arreglos y le damos una dimensión, especificamos el encabezado de nuestra columna, vamos a obtener la colección de usuario de la base de datos, vamos a iterar los documentos de nuestra colección donde vamos a tener los datos de registro y especificamos el llamado de nuestra TablaRegistro ya que establece el DefaultTableModel donde se obtendrá los datos registrados en nuestra tabla.

```

    }
    // Método para mostrar los datos de estudiantes en la interfaz gráfica
    public void mostrar(){
        // Títulos de las columnas de la tabla
        String[] titulos={"ID","Nombre","Edad","Carrera"};
        // Arreglo para almacenar los valores de cada fila
        String [] registros=new String [4];
        // Crear un modelo de tabla con los títulos especificados
        DefaultTableModel mod=new DefaultTableModel(null,titulos);
        // Obtener la colección "usuarios" de la base de datos
        DBCollection colec =db.getCollection("usuarios");
        // Crear un cursor para iterar sobre los documentos en la colección
        DBCursor cursor=colec.find();
        while(cursor.hasNext()){
            // Obtener el valor del campo cada 0 c0ampo
            registros[0]=""+cursor.next().get("_id")+"";
            registros[1]=""+cursor.curr().get("Nombre")+"";
            registros[2]=""+cursor.curr().get("Edad")+"";
            registros[3]=""+cursor.curr().get("Carrera")+"";
            // Agregar la fila al modelo de la tabla
            mod.addRow(registros);
        }
        // Establecer el modelo de la tabla en la interfaz
        TablaRegistro.setModel(mod);
        // Configurar la tabla para que no se puedan editar las celdas
        TablaRegistro.setDefaultEditor(Object.class, null);
    }
}

```

Paso Séptimo

Creamos un método guardar ya que nos ayuda aguardar los datos que emos ingresados donde daremos a conocer una referencia de la colección para usuario que se almacena en la base de datos, creamos un nuevo objeto de BasicDBObject para representar el documento, llamaos el objeto de documento donde llamaremos todos nuestros txt que emos ingresados y le diremos que aguarde los datos utilizamos una colección para llamar a nuestra colección de nuestra base de datos.

```

// Método para guardar un nuevo registro de estudiante en la base de datos
public void Guardar() {
    // Obtener la colección "usuarios" de la base de datos
    DBCollection cole=db.getCollection("usuarios");
    // Crear un nuevo objeto para almacenar los campos del nuevo documento
    BasicDBObject documento=new BasicDBObject();
    // Establecer los valores del nuevo documento a partir de los datos de la interfaz gráfica
    documento.put("Nombre", txtNombre.getText());
    documento.put("Nombre", txtNombre.getText());
    documento.put("Edad", Integer.parseInt(txtEdad.getText()));
    documento.put("Carrera", txtCarrera.getText());
    // Insertar el nuevo documento en la colección
    cole.insert(documento);
}

```

Paso Octavo

Creamos un método actualizar ya que nos ayuda editar los datos que queríamos cambiar si hemos ingresado algún dato mal aguarado, daremos a conocer una referencia de la colección para

profesores que se almacena en la base de datos, creamos un nuevo objeto de BasicDBObject para representar el actualizarAlum, llamaos el objeto de actualizarAlum donde llamaremos todos nuestros txt y le diremos que aguarde los datos utilizamos una colección para llamar a nuestra colección de nuestra base de datos.

```
// Método para actualizar un registro de estudiante en la base de datos
public void actualizar () {
    // Obtener la colección "usuarios" de la base de datos
    DBCollection colec=db.getCollection("usuarios");
    // Crear un objeto para almacenar los campos a actualizar
    BasicDBObject actualizarAlumni=new BasicDBObject();
    // Establecer los valores actualizados a partir de los datos de la interfaz gráfica
    actualizarAlumni.put("Nombre", txtNombre.getText());
    actualizarAlumni.put("Edad", Integer.parseInt(txtEdad.getText()));
    actualizarAlumni.put("Carrera", txtCarrera.getText());
    // Buscar y modificar el registro utilizando el objeto de actualización
    colec.findAndModify(backup, actualizarAlumni);
}
```

Paso Nueve

Creamos un método limpiar para limpiar la caja de textos de los datos que hemos ingresados.

```
// Método para limpiar los campos de entrada de la interfaz gráfica
public void limpiar() {
    txtNombre.setText("");
    txtEdad.setText("");
    txtCarrera.setText("");
}
```

Paso Diez

Creamos un método para eliminar donde nos ayuda a eliminar un registro que no querido ingresar.

```
// Método para eliminar un registro de estudiante de la base de datos
public void eliminar() {
    // Obtener la colección "usuarios" de la base de datos
    DBCollection colec=db.getCollection("usuarios");
    // Buscar y eliminar el registro utilizando el objeto de respaldo
    colec.findAndRemove(backup);
}
```

Paso Once

El método `docbackupProfesor` nos ayuda mucho a tener un respaldo del documentó que hemos ingresado.

```
// Método para crear un respaldo del documento actual mostrado en la interfaz
public void docbackup() {
    // Crear un nuevo objeto para almacenar los campos de respaldo
    backup=new BasicDBObject();
    // Establecer los valores actuales en el objeto de respaldo a partir de los campos de entrada de la interfaz
    backup.put("Nombre", txtNombre.getText());
    backup.put("Edad", Integer.parseInt(txtEdad.getText()));
    backup.put("Carrera", txtCarrera.getText());
}
```

Paso Doce

Damos una configuración en la Tabla del registro Profesor donde por medio de un clic podemos editar o eliminar nuestro registro y también llamaremos nuestro respaldos de nuestros datos ingresados.

```
private void TablaRegistroMouseClicked(java.awt.event.MouseEvent evt) {
    // Obtener la fila en la que se hizo clic
    int fila=TablaRegistro.rowAtPoint(evt.getPoint());
    // Establecer los valores de la fila seleccionada en los campos de entrada de la interfaz
    // Establecer los campo
    txtNombre.setText(TablaRegistro.getValueAt(fila,1).toString());
    txtEdad.setText(TablaRegistro.getValueAt(fila,2).toString());
    txtCarrera.setText(TablaRegistro.getValueAt(fila,3).toString());
    // Realizar un respaldo del documento seleccionado
    docbackup();
}
```

Paso Trece

Llamaremos todos los métodos en nuestros botones correctamente para que el programa funciones correctamente.

```

private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
    Guardar();
    mostrar();
    limpiar();
}

private void btnActualizarActionPerformed(java.awt.event.ActionEvent evt) {
    actualizar();
    mostrar();
    limpiar();
}

private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
    eliminar();
    mostrar();
    limpiar();
}

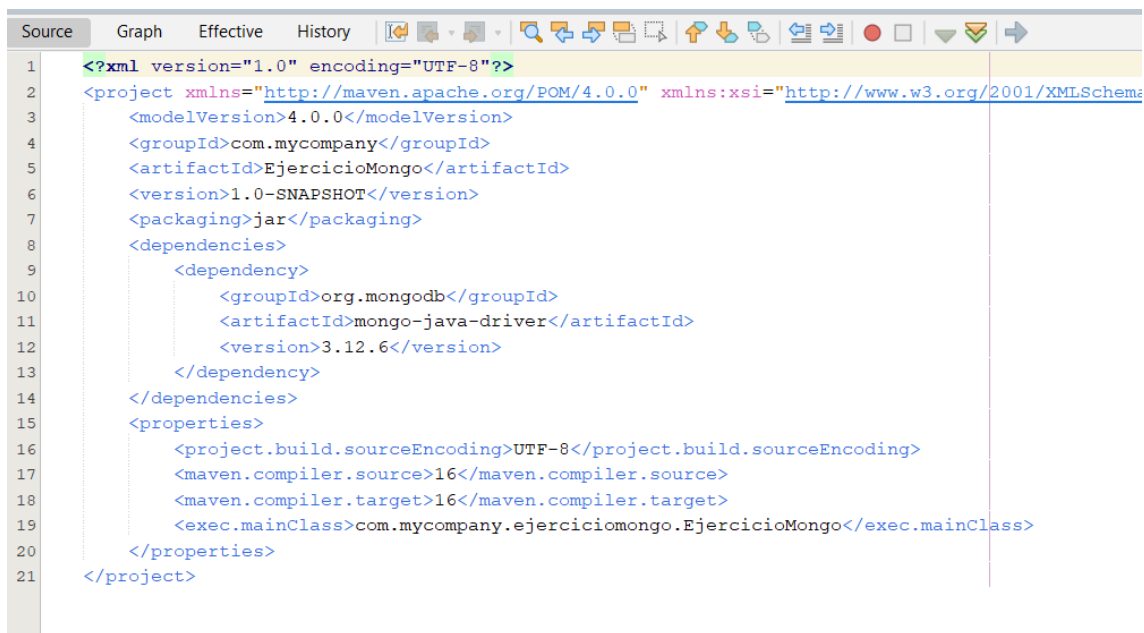
private void btnSalirActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}

private void btnLimpiarActionPerformed(java.awt.event.ActionEvent evt) {
    limpiar();
}

```

Paso Catorce

Colocamos nuestra dependencia de nuestro mongoDB



The screenshot shows an IDE window with a Maven POM file. The file is named 'EjercicioMongo' and is located in the 'com.mycompany' package. It is a snapshot version (1.0-SNAPSHOT) and is packaged as a jar. The dependencies section includes the MongoDB Java driver (org.mongodb:mongo-java-driver:3.12.6). The properties section specifies UTF-8 encoding, Java 16 compiler source and target, and the main class 'com.mycompany.ejerciciomongo.EjercicioMongo'.

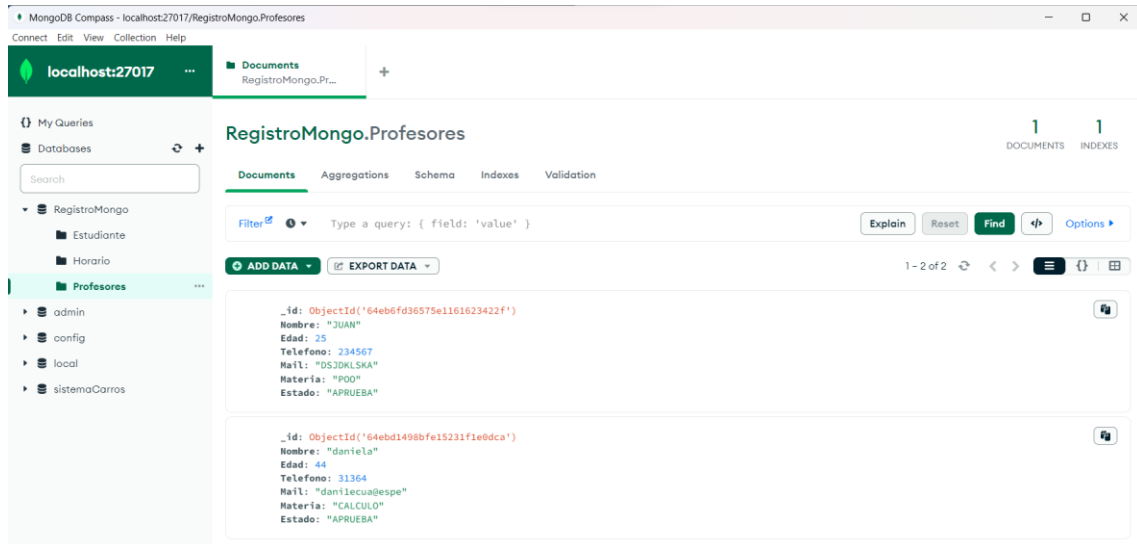
```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema"
3      <modelVersion>4.0.0</modelVersion>
4      <groupId>com.mycompany</groupId>
5      <artifactId>EjercicioMongo</artifactId>
6      <version>1.0-SNAPSHOT</version>
7      <packaging>jar</packaging>
8      <dependencies>
9          <dependency>
10             <groupId>org.mongodb</groupId>
11             <artifactId>mongo-java-driver</artifactId>
12             <version>3.12.6</version>
13          </dependency>
14      </dependencies>
15      <properties>
16          <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
17          <maven.compiler.source>16</maven.compiler.source>
18          <maven.compiler.target>16</maven.compiler.target>
19          <exec.mainClass>com.mycompany.ejerciciomongo.EjercicioMongo</exec.mainClass>
20      </properties>
21  </project>

```


Paso Quince

Conectamos a nuestra base de datos MongoDB donde tendremos la carpeta de nuestra base de datos.



REGISTRO ESTUDIANTE

ID	Nombre	Edad	Carrera
64ebe0d997f5b9...	kelvin	22	itin
64ebe0f197f5b9...	pedro	55	Biotechnologia
64ebe10597f5b9...	Jhon	25	Agropecuaria

Link Del Video

https://www.youtube.com/watch?v=p_H63kVis2E&t=308s

<https://www.youtube.com/watch?v=KuiQYqJz8Jc&t=1s>

<https://www.youtube.com/watch?v=xJ-Tz6MYCUE>

Link de GitHub

https://github.com/KelvinQuezada/Materia_POO.git

Carpeta Recuperacion_Lab1_Quezada