

UNIVERSIDAD MARIANO GÁLVEZ DE GUATEMALA
CENTRO UNIVERSITARIO EL NARANJO
FACULTAD DE INGENIERIA DE SISTEMAS DE
INFORMACIÓN

Programacion 3

ING. Alan G. Ucelo Morán



Tarea #7

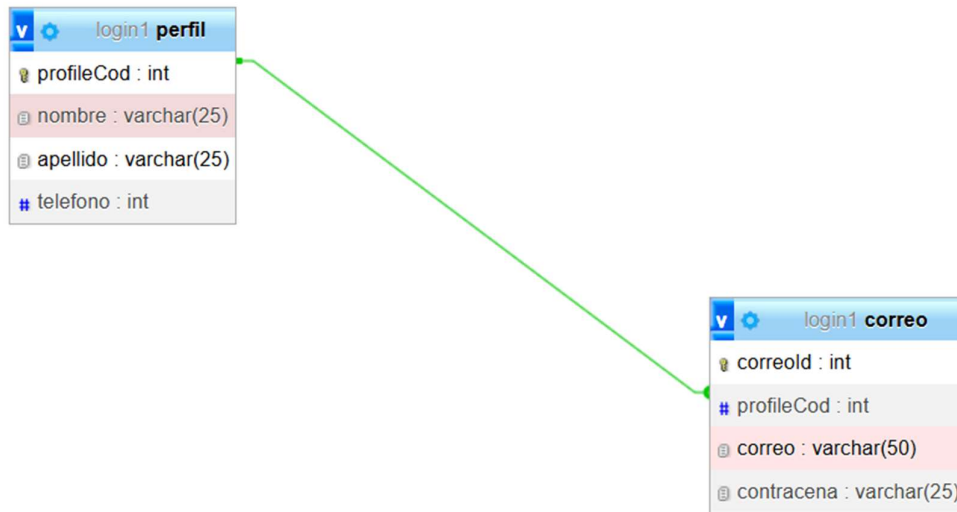
KEVIN ALEXANDER MAZARIEGOS PINEDA

CARNÉ: 9390-22-5048

5 de Junio del 2024

Base de datos:

- Modelo :



- Código:

```
create database login1;
use login1;

DROP TABLE IF EXISTS `correo`;
CREATE TABLE IF NOT EXISTS correo (
  correoId int NOT NULL,
  profileCod int DEFAULT NULL,
  correo varchar(50) DEFAULT NULL,
  contracena varchar(25) DEFAULT NULL,
  PRIMARY KEY (correoId),
  CONSTRAINT FkProfile FOREIGN KEY (profileCod) REFERENCES perfil (profileCod)
) ENGINE=InnoDB ;

DROP TABLE IF EXISTS perfil;
CREATE TABLE IF NOT EXISTS perfil (
  profileCod int NOT NULL,
  nombre varchar(25) DEFAULT NULL,
  apellido varchar(25) DEFAULT NULL,
  telefono int DEFAULT NULL,
  PRIMARY KEY (profileCod)
) ENGINE=InnoDB
```

```
Create database login1;
use login1;
```

```
DROP TABLE IF EXISTS `correo`;
CREATE TABLE IF NOT EXISTS correo (
  correold int NOT NULL,
  profileCod int DEFAULT NULL,
  correo varchar(50) DEFAULT NULL,
  contracena varchar(25) DEFAULT NULL,
  PRIMARY KEY (correold),
  CONSTRAINT FkProfile FOREIGN KEY (profileCod) REFERENCES perfil (profileCod)
) ENGINE=InnoDB ;
```

```
DROP TABLE IF EXISTS perfil;
CREATE TABLE IF NOT EXISTS perfil (
  profileCod int NOT NULL,
  nombre varchar(25) DEFAULT NULL,
  apellido varchar(25) DEFAULT NULL,
  telefono int DEFAULT NULL,
  PRIMARY KEY (profileCod)
) ENGINE=InnoDB
```

Conexión con Java:

```
package login;

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

public class conexion {
    private static final String DRIVER = "com.mysql.cj.jdbc.Driver";
    private static final String URL = "jdbc:mysql://localhost:3306/login1"; // Asegúrate de que el puerto es correcto (usualmente 3306).
    private static final String USUARIO = "root";
    private static final String PASS = "";

    public static Connection obtenerConexion() {
        try {
            System.out.println("conectada papa");
            Class.forName("com.mysql.cj.jdbc.Driver");
            return DriverManager.getConnection(url: URL, user: USUARIO, password: PASS);
        } catch (ClassNotFoundException e) {
            System.out.println("Error al registrar el driver de MySQL: " + e);
        } catch (SQLException e) {
            System.out.println("Error al conectar con la base de datos: " + e);
        }
        return null; // Retornamos null si la conexión falló
    }
}
```

Creamos los métodos get, set y constructor para referenciar la base de datos

```
package model;

public class correo {
    int correoID;
    int perfilID;
    String correo;
    String contracena;

    public correo(int correoID, int perfilID, String correo, String contracena) {
        this.correoID = correoID;
        this.perfilID = perfilID;
        this.correo = correo;
        this.contracena = contracena;
    }

    public correo() {
    }

    public int getCorreoID() {
        return correoID;
    }

    public void setCorreoID(int correoID) {
        this.correoID = correoID;
    }

    public int getPerfilID() {
        return perfilID;
    }

    public void setPerfilID(int perfilID) {
        this.perfilID = perfilID;
    }

    public String getCorreo() {
        return correo;
    }
}
```

Luego creamos el constructor donde estará la lógica del CRUD

```
public boolean insertarCorreo(correo correo) {
    sql = "INSERT INTO correo(correoId, profileCod, correo, contracena) VALUES (?, ?, ?, ?)";

    boolean inserted = false;
    try {
        conn = conexion.obtenerConexion();
        pstmt = conn.prepareStatement(string:sql);
        pstmt.setInt(1, correo.getCorreoID());
        pstmt.setInt(2, correo.getPerfilID());
        pstmt.setString(3, correo.getCorreo());
        pstmt.setString(4, correo.getContracena());

        int success = pstmt.executeUpdate();
        if (success > 0) {
            inserted = true;
        }
    } catch (SQLException e) {
        System.out.println("Error al insertar Correo: " + e.getMessage());
    } finally {
        try {
            if (pstmt != null) pstmt.close();
            if (conn != null) conn.close();
        } catch (SQLException ex) {
            System.out.println("Error al cerrar conexiones: " + ex.getMessage());
        }
    }
    return inserted;
}
```

```
public boolean eliminarCorreo(int idCorreo) {
    sql = "DELETE FROM correo WHERE correoId=?";
    boolean deleted = false;

    try {
        conn = conexion.obtenerConexion();
        pstmt = conn.prepareStatement(string:sql);
        pstmt.setInt(1, idCorreo);

        int success = pstmt.executeUpdate();
        if (success > 0) {
            deleted = true;
        }
    } catch (SQLException e) {
        System.out.println("Error al eliminar correo: " + e.getMessage());
    } finally {
        try {
            if (pstmt != null) pstmt.close();
            if (conn != null) conn.close();
        } catch (SQLException ex) {
            System.out.println("Error al cerrar conexiones: " + ex.getMessage());
        }
    }
    return deleted;
}
```

```

public boolean modificarCorreo(correo correo) {
    sql = "UPDATE correo SET correo=?, contracena=? WHERE correoId=?";
    boolean updated = false;
    try {
        conn = conexion.obtenerConexion();
        pstmt = conn.prepareStatement(string:sql);
        pstmt.setString(i: 1, string:correo.getCorreo());
        pstmt.setString(i: 2, string:correo.getContracena());
        pstmt.setInt(i: 3, i1: correo.getCorreoID());
        int success = pstmt.executeUpdate();
        if (success > 0) {
            updated = true;
        }
    } catch (SQLException e) {
        System.out.println("Error al modificar correo: " + e.getMessage());
    } finally {
        try {
            if (pstmt != null) pstmt.close();
            if (conn != null) conn.close();
        } catch (SQLException ex) {
            System.out.println("Error al cerrar conexiones: " + ex.getMessage());
        }
    }
    return updated;
}

```

```

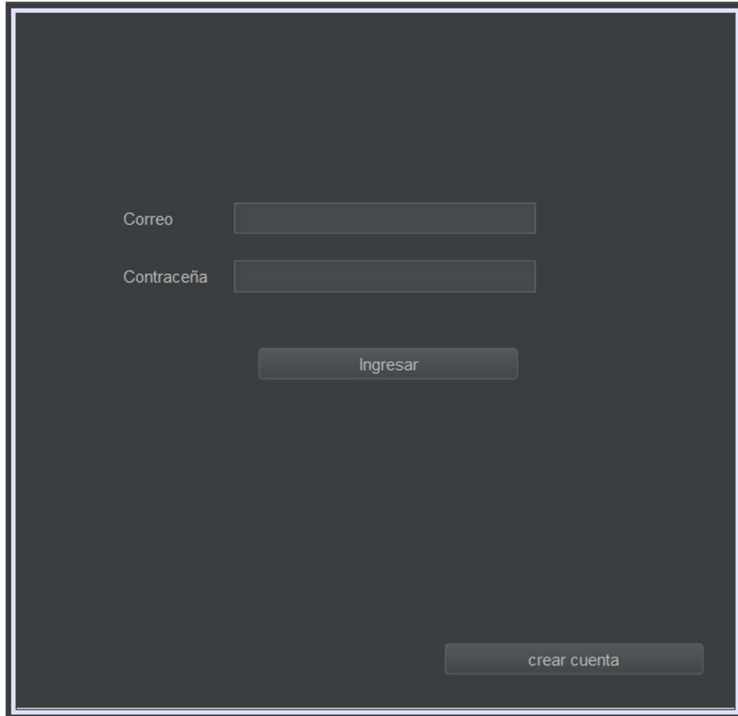
public correo buscarCorreo(String idCorreo) {
    sql = "SELECT * FROM correo WHERE correo=?";
    correo correoEncontrado = null;

    try {
        conn = conexion.obtenerConexion();
        pstmt = conn.prepareStatement(string:sql);
        pstmt.setString(i: 1, string:idCorreo);
        rs = pstmt.executeQuery();

        if (rs.next()) {
            correoEncontrado = new correo();
            correoEncontrado.setCorreoID(correoID: rs.getInt(string:"correoId"));
            correoEncontrado.setPerfilID(perfilID: rs.getInt(string:"profileCod"));
            correoEncontrado.setCorreo(correo:rs.getString(string:"correo"));
            correoEncontrado.setContracena(contracena: rs.getString(string:"contracena"));
        }
    } catch (SQLException e) {
        System.out.println("Error al buscar correo: " + e.getMessage());
    } finally {
        try {
            if (rs != null) rs.close();
            if (pstmt != null) pstmt.close();
            if (conn != null) conn.close();
        } catch (SQLException ex) {
            System.out.println("Error al cerrar conexiones: " + ex.getMessage());
        }
    }
    return correoEncontrado;
}

```

Ya con el constructor podemos crear la parte grafica y las llamadas



The image shows a dark-themed login window. It contains two text input fields: the first is labeled 'Correo' and the second is labeled 'Contraseña'. Below these fields is a button labeled 'Ingresar'. At the bottom right of the window is another button labeled 'crear cuenta'.

El botón de crear cuenta nos direcciona a otro formato donde se crea el perfil y el correo a la vez

```
private void btnCrearActionPerformed(java.awt.event.ActionEvent evt) {  
    System.out.println(x: "entro a crear cuenta");  
    Crear crear = new Crear();  
    crear.setVisible(b: true);  
    this.dispose();  
}
```

El ingresar usa un select from para validar y enviar al usuario a la pagina principal

```
private void btnIngresarActionPerformed(java.awt.event.ActionEvent evt) {  
  
    String correo = txtcorreo.getText().trim();  
    String contrasena = txtContracena.getText().trim();  
  
    if (this.correo.validarCredenciales(correo, contrasena)) {  
        System.out.println(x: "Ingreso exitoso");  
        pagina1 pagina = new pagina1(email: correo);  
        pagina.setVisible(b: true);  
        this.dispose(); // Cierra la ventana de login  
    } else {  
        System.out.println(x: "Credenciales incorrectas");  
    }  
}
```


En el apartado de crear solicita los datos a ingresar en la base de datos

nombre

apellido

telefono

correo

contraseña

confirmacion

registrar

cancelar

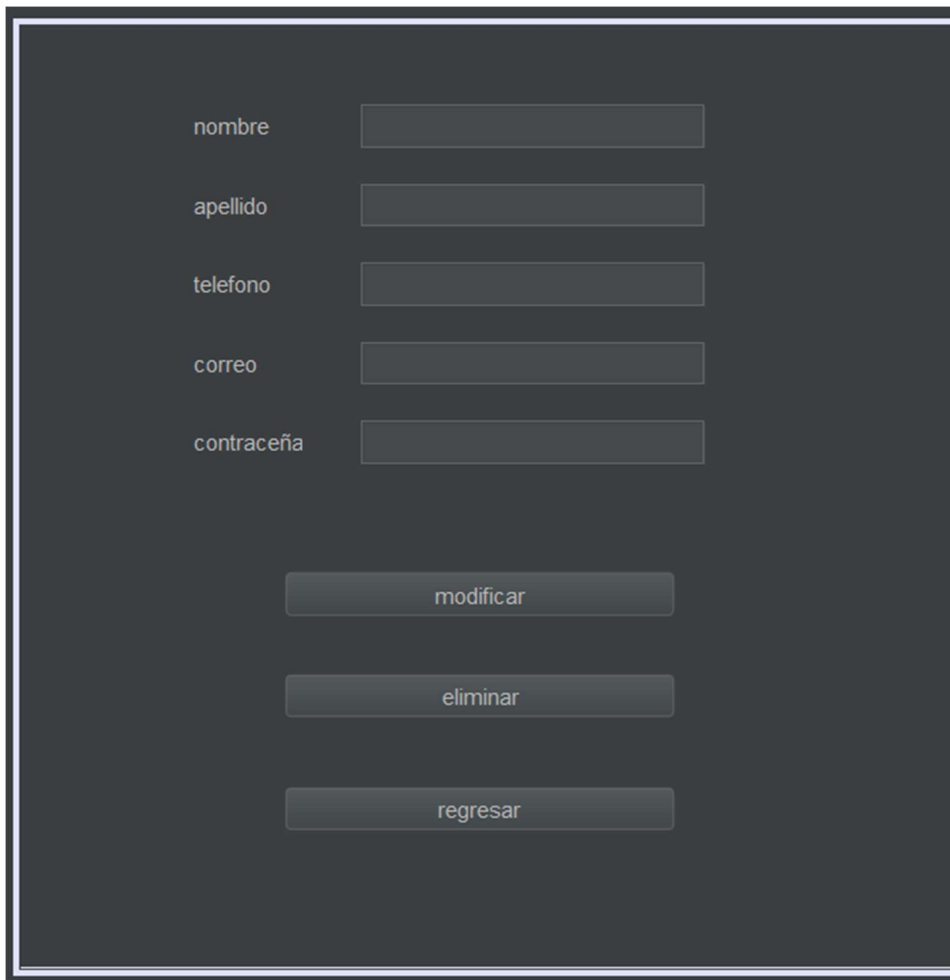
Con el botón insertar se ingresan los datos a la base de datos y se ingresa a la pagina principal

```
private void txtInsertarActionPerformed(java.awt.event.ActionEvent evt) {  
    String nombre = txtNombre.getText().trim();  
    String apellido = txtApellido.getText().trim();  
    int telefono = Integer.parseInt(txtTelefono.getText().trim());  
    String credencial = txtCorreo.getText().trim();  
    String pass = txtPass.getText().trim();  
    String confirmacion = txtConfirmar.getText().trim();  
  
    if (pass.equals(confirmacion)) {  
        int codigoPerfil = generarCodigo();  
        perfil perfil = new perfil(perfilCod:codigoPerfil, nombre, apellido, telefono);  
        correo correo = new correo(correoID: generarCodigo(), perfilID: codigoPerfil, correo:credencial, contraseña:pass);  
        if (perfilC.insertarPerfil(perfil)) {  
            if (correoC.insertarCorreo(correo)) {  
                System.out.println(x: "Usuario registrado exitosamente.");  
  
                if (this.correoC.validarCredenciales(correo:credencial, contraseña:pass)) {  
                    System.out.println(x: "Ingreso exitoso");  
                    paginal pagina = new paginal(email: credencial);  
                    pagina.setVisible(b: true);  
                    this.dispose(); // Cierra la ventana de login  
                } else {  
                    System.out.println(x: "Credenciales incorrectas");  
                }  
  
            } else {  
                System.out.println(x: "Error al insertar correo.");  
            }  
        } else {  
            System.out.println(x: "Error al insertar perfil.");  
        }  
    } else {  
        System.out.println(x: "Las contraseñas no coinciden.");  
    }  
}
```


Con el botón cancelar se regresa al primer formulario sin necesidad de llenar las casillas y también tenemos un método que nos ayuda a insertar el código de manera aleatoria

```
private void btnCancelarActionPerformed(java.awt.event.ActionEvent evt) {  
    login login = new login();  
    login.setVisible(b: true);  
    this.dispose();  
}  
  
private int generarCodigo() {  
    Random random = new Random();  
    return random.nextInt(bound: 999999); // Genera un número aleatorio entre 0 y 999999  
}
```

La pagina principal solo nos muestra los datos que el agente inserto atraves de la base de datos y tiene la opción de editar los valores o eliminar el usuario



A screenshot of a user management interface. It features five text input fields arranged vertically, each preceded by a label: 'nombre', 'apellido', 'telefono', 'correo', and 'contraseña'. Below these fields are three buttons stacked vertically: 'modificar', 'eliminar', and 'regresar'. The entire form is set against a dark gray background with a thin white border.

Con este método imprimimos los valores de la base de datos aquí en la pagina principal

```
private void cargarDatosUsuario(String email) {
    correo user = correoC.buscarCorreo(idCorreo: email);
    if (user != null) {
        perfil usuario = perfilC.buscarPerfil(idPerfil: user.getPerfilID());
        if (usuario != null) {
            perfilId = usuario.getPerfilCod();
            txtNombre.setText(t: usuario.getNombre());
            txtApellido.setText(t: usuario.getApellido());
            txtTefono.setText(t: String.valueOf(1: usuario.getTelefono()));
            txtCorreo.setText(t: user.getCorreo());
            txtPass.setText(t: user.getContracena());

            // Asignar el correoID al objeto co
            co.setCorreoID(correoID: user.getCorreoID());
        } else {
            System.out.println(x: "No se encontró el perfil del usuario");
        }
    } else {
        System.out.println(x: "No se encontró el usuario");
    }
}
```

En esta parte está el código de modificar que se ara cargo de modificar tanto el perfil como la contraseña

```
private void txtModificarActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        String nombre = txtNombre.getText().trim();
        String apellido = txtApellido.getText().trim();
        int telefono = Integer.parseInt(s: txtTefono.getText().trim());
        String correo = txtCorreo.getText().trim();
        String pass = txtPass.getText().trim();

        co.setCorreo(correo);
        co.setContracena(contracena: pass);

        if (correoC.modificarCorreo(correo:co)) {
            System.out.println(x: "Correo modificado con éxito");
            perfil perfilModificado = new perfil();
            perfilModificado.setPerfilCod(perfilCod:perfilId); // Set this according to your logic
            perfilModificado.setNombre(nombre);
            perfilModificado.setApellido(apellido);
            perfilModificado.setTelefono(telefono);

            if (perfilC.modificarPerfil(perfil:perfilModificado)) {
                System.out.println(x: "Perfil modificado con éxito");
            } else {
                System.out.println(x: "Error al modificar el perfil");
            }
        } else {
            System.out.println(x: "Error al modificar el correo electrónico");
        }
    } catch (NumberFormatException e) {
        System.out.println("Error en el formato del teléfono: " + e.getMessage());
    }
}
```

Ahora tenemos el botón de eliminar que eliminara el registro de la base de datos y otro de salir

```
private void txtEliminarActionPerformed(java.awt.event.ActionEvent evt) {  
    if(correoC.eliminarCorreo(idCorreo: co.getCorreoID())) {  
        System.out.println(x: co.getCorreoID());  
        if (perfilC.eliminarPerfil(idPerfil: perfilId)) {  
            System.out.println(x: "credencial eliminado con éxito");  
        } else {  
            System.out.println(x: "Error al eliminar la credencial");  
        }  
    } else {  
        System.out.println(x: "error en eliminar el correo");  
        System.out.println(x: co.getCorreoID());  
    }  
}  
  
private void btnCancelActionPerformed(java.awt.event.ActionEvent evt) {  
    login log = new login();  
    log.setVisible(b: true);  
    this.dispose();  
}
```