

Universidad San Carlos de Guatemala

Centro Universitario de Occidente

División de Ciencias de la Ingeniería

Laboratorio de Introducción a la Programación y
Computación 2

Ingeniero Daniel González



Proyecto 1

Luis David Pérez Gómez

202231411

Quetzaltenango 12 de Marzo del 2025

Manual Técnico

El siguiente programa fue hecho en la IDE de Apache Netbeans en el lenguaje de programación java, con ayuda de JSP como aplicación web y con base de datos por parte de MySQL, también con la conexión tomcat para la ejecución.

Para el inicio del programa se ubica primero el index.jsp es lo primero que aparecerá en la página web como el inicio de sesión.

```
1  <!--
2  Document:  index
3  Created on: 08-mar-2023, 18:19:37
4  Author:    luis
5  -->
6
7  <%PAGE contentType="text/html" pageEncoding="UTF-8"%>
8  <!DOCTYPE html>
9  <html>
10     <head>
11         <meta charset="UTF-8" http-equiv="X-UA-Compatible" content="IE=edge">
12         <title>Login</title>
13         <script src="https://cdn.tailwindcss.com" ></script>
14     </head>
15     <body class="flex items-center justify-center min-h-screen bg-blue-100">
16         <div class="w-full max-w-md p-6 bg-white shadow-lg rounded-lg">
17             <h1 class="text-2xl font-bold text-center text-gray-800 mb-6">Iniciar Sesión</h1>
18
19             <form class="space-y-4">
20                 <div class="mb-4">
21                     <label class="block text-gray-700 font-medium mb-2">Usuario</label>
22                     <input type="text" class="w-full px-4 py-2 border rounded-lg focus:outline-none focus:ring-2 focus:ring-purple-500">
23                 </div>
24
25                 <div class="mb-4">
26                     <label class="block text-gray-700 font-medium mb-2">Contraseña</label>
27                     <input type="password" class="w-full px-4 py-2 border rounded-lg focus:outline-none focus:ring-2 focus:ring-purple-500">
28                 </div>
29             </form>
30         </div>
```

Luego de pedirle al usuario que inicie sesión con sus datos, el botón de ingresar re direcciona a la siguiente página, que es el panel de administración.

```

37 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
38 <title>Administración de Usuarios</title>
39 <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
40 <script src="https://code.jquery.com/jquery-ui-1.12.1.min.js"></script>
41 </head>
42 <body class="flex h-screen bg-gray-100">
43 <aside class="w-64 bg-gray-800 text-white flex flex-col">
44 <div class="p-5 text-xl font-bold border-b border-gray-700">
45 Panel de Acciones
46 </div>
47 <nav class="flex-1 p-4">
48 <a href="#" class="block px-4 py-2 rounded-lg hover:bg-gray-700">Áreas de Trabajo</a>
49 <a href="#" class="block px-4 py-2 rounded-lg hover:bg-gray-700">Usuarios</a>
50 <a href="#" class="block px-4 py-2 rounded-lg hover:bg-gray-700">Lista de Usuarios</a>
51 <a href="#" class="block px-4 py-2 rounded-lg hover:bg-gray-700">Piezas</a>
52 <a href="#" class="block px-4 py-2 rounded-lg hover:bg-gray-700">Computadoras</a>
53 </nav>
54 <div class="p-4 border-t border-gray-700">
55 <a href="#" class="block px-4 py-2 text-red-500 hover:bg-red-600 hover:text-white rounded-lg">
56 Cerrar sesión
57 </a>
58 </div>
59 </aside>
60 <div class="flex-1 flex flex-col">
61
62 <header class="bg-white shadow-md p-4 flex justify-between items-center">
63 <h1 class="text-xl font-bold">Administración de Usuarios</h1>
64 <div class="flex items-center space-x-4">
65 

```

Esta parte se encarga de mostrar con html al usuario sus opciones en este panel, se verifican la informacion del servlet y también de las cualidades del objeto user.

```

import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;

/**
 *
 * @author Inury
 */
public class UserQuery extends CrudModel {

    @Override
    public void insert(UserModel entity) throws SQLException {
        String sql = "INSERT INTO users (name, password, type) VALUES (?, ?, ?)";
        try {
            Connection conn = DBConnection.getConnection();
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setString(1, entity.getName());
            stmt.setString(2, entity.getPassword());
            stmt.setInt(3, entity.getType());

            int affectedRows = stmt.executeUpdate();
            if (affectedRows > 0) {
                entity.setName(entity.getName());
            }
        }
    }

    @Override
    public void update(UserModel entity) throws SQLException {
        String sql = "UPDATE users SET name=?, password=?, type=? WHERE id=?";
        try {
            Connection conn = DBConnection.getConnection();
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setString(1, entity.getName());
            stmt.setString(2, entity.getPassword());
            stmt.setInt(3, entity.getType());
            stmt.setInt(4, entity.getId());

            int affectedRows = stmt.executeUpdate();
            if (affectedRows > 0) {
                entity.setName(entity.getName());
            }
        }
    }

    @Override
    public void delete(UserModel entity) throws SQLException {
        String sql = "DELETE FROM users WHERE id=?";
        try {
            Connection conn = DBConnection.getConnection();
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setInt(1, entity.getId());

            int affectedRows = stmt.executeUpdate();
            if (affectedRows > 0) {
                entity.setName(entity.getName());
            }
        }
    }

    @Override
    public List<UserModel> findAll() throws SQLException {
        String sql = "SELECT * FROM users";
        try {
            Connection conn = DBConnection.getConnection();
            Statement stmt = conn.createStatement();
            ResultSet rs = stmt.executeQuery(sql);
            List<UserModel> users = new ArrayList<>();
            while (rs.next()) {
                UserModel user = new UserModel();
                user.setName(rs.getString("name"));
                user.setPassword(rs.getString("password"));
                user.setType(rs.getInt("type"));
                user.setId(rs.getInt("id"));
                users.add(user);
            }
            return users;
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    @Override
    public UserModel findById(int id) throws SQLException {
        String sql = "SELECT * FROM users WHERE id=?";
        try {
            Connection conn = DBConnection.getConnection();
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setInt(1, id);
            ResultSet rs = stmt.executeQuery();
            if (rs.next()) {
                UserModel user = new UserModel();
                user.setName(rs.getString("name"));
                user.setPassword(rs.getString("password"));
                user.setType(rs.getInt("type"));
                user.setId(rs.getInt("id"));
                return user;
            }
            return null;
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

Luego está la conexión hacia el MySQL donde la clase BDConnection se encarga de llamar al MySQL.

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

/**
 *
 * @author luisg
 */
public class BDConnection {

    private static final String URL = "jdbc:mysql://localhost:3306/db_ComputadoraFelix";
    private static final String USER = "root";
    private static final String PASSWORD = "root";

    private static Connection connection;

    static {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            throw new RuntimeException("Error al cargar el Driver JDBC", e);
        }
    }

    private BDConnection() {}

    public static Connection getConnection() throws SQLException {
        if (connection == null || connection.isClosed()) {
            synchronized (BDConnection.class) {

```

Y por último están todos los constructores del programa situados en el package de models.

```
package Models;

/**
 *
 * @author luisg
 */
public class ComputerModel {

    private String name;
    private float price;

    public ComputerModel() {
    }

    public ComputerModel(String name, float price) {
        this.name = name;
        this.price = price;
    }

    @Override
    public String toString() {
        return "UserModel{" + "name = " + name + ", price = " + price + '}';
    }

    public String getName() {
        return name;
    }
}
```

```

package Models;

/**
 *
 * @author luisg
 */
public class PieceModel {

    private String type;
    private Integer cost;

    public PieceModel(){
    }

    public PieceModel(String type, Integer cost) {
        this.type = type;
        this.cost = cost;
    }

    @Override
    public String toString(){
        return "PieceModel{" + "type = " + type + ", costo = " + cost + '}';
    }

    public String getType() {
        return type;
    }

    public void setType(String type) {
        this.type = type;
    }
}

```

```

package Models;

/**
 *
 * @author luisg
 */
public class UserModel {

    private String name;
    private String password;
    private int type;

    public UserModel(){
    }

    public UserModel(String name, String password, int type) {
        this.name = name;
        this.password = password;
        this.type = type;
    }

    @Override
    public String toString(){
        return "UserModel{" + "name = " + name + ", password = " + password + ", type = " + type + '}';
    }

    public String getName() {
        return name;
    }
}

```

Diagrama de Clases

Diagrama E/R

[https://drive.google.com/file/d/1BSGEm4mE8F8q3aJ21-gwnpWtQ3JAONxb/view?usp=drive link](https://drive.google.com/file/d/1BSGEm4mE8F8q3aJ21-gwnpWtQ3JAONxb/view?usp=drive_link)

Diagrama y Mapeo Físico de la DB

