

UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE

SEGUNDO SEMESTRE

PRIMER PARCIAL

PROGRAMACIÓN ORIENTADA A OBJETOS

SISTEMA PARA GESTIONAR UN PARQUEADERO

INTEGRANTES:

- Camacho Bravo Pablo Guber
- Clavijo Alban Joan Sebastian
- Licuy Mecías Estalyn Danie
- Loachamin Tipan Nayeli Scarleth
- Quillupangui Tupe Deisy Abigail

NRC:

1323

DOCENTE:

LUIS ENRIQUE JARAMILLO MONTAÑO

SANGOLQUÍ - ECUADOR

1. INTRODUCCIÓN

Un sistema de parqueo en Programación Orientada a Objetos (POO) es un software diseñado para gestionar y controlar el acceso a un estacionamiento o parqueo. Este tipo de sistema se puede utilizar para administrar diferentes aspectos de un parqueo, como la disponibilidad de espacios, el registro de vehículos, la asignación de espacios, el control de pagos y la gestión de tiempos de estacionamiento. En POO, el sistema se modela mediante clases y objetos donde cada componente del parqueo, como los vehículos, los espacios de estacionamiento o el sistema de pagos, se representa como una clase.

2. OBJETIVOS

2.1 OBJETIVO GENERAL

Desarrollar un sistema que simplifique el proceso de cobro por parqueadero, así como la distribución de los vehículos dentro de este.

2.2 OBJETIVOS ESPECÍFICOS

- Optimizar el flujo de vehículos en el parqueadero, permitiendo el ahorro del tiempo por parte del usuario.
- Facilitar la búsqueda de un cupo disponible en el parqueadero.
- Garantizar al usuario que sus autos no sufrirán algún daño.

3. MARCO TEÓRICO

3.1 ANÁLISIS DEL PROCESO

Se identificaron las funciones necesarias para realizar el sistema, dividiéndolas en tareas específicas asignadas a todos los integrantes del grupo.

3.2 ANÁLISIS DE REQUISITOS

El sistema debe cumplir con las siguientes funcionalidades:

- **Registro de Vehículos:** Permite registrar datos como la placa, el tipo de vehículo, marca y modelo
- **Consulta de Vehículos:** Brindar información sobre los vehículos registrados, ya sea en tiempo real o mediante filtros.

3.3 DIAGRAMA UML:

El diseño del UML para el sistema de gestión de un parqueadero incluye las funcionalidades mencionadas: registro, consulta y actualización de vehículos. El diagrama incluye las clases principales, sus atributos, métodos y las relaciones entre ellas.

Clases principales del sistema

Vehículo

- Atributos: placa, marca, modelo, color, tipo
- Métodos: registrarVehiculo(), actualizarVehiculo(), consultarVehiculo()

EspacioParqueo

- Atributos: idEspacio, disponible, tipoEspacio
- Métodos: asignarEspacio(), liberarEspacio()

GestorParking

- Atributos: listaVehiculos, listaEspacios
- Métodos: registrarVehiculo(), consultarVehiculo(), actualizarVehiculo(), asignarEspacio(), generarReporte()

BaseDatos

- Atributos: conexion, nombreTabla
- Métodos: guardarDatos(), leerDatos(), actualizarDatos(), eliminarDatos()

InterfazGráfica

- Métodos: mostrarFormularioRegistro(), mostrarConsulta(), mostrarActualizacion()

Relaciones:

- **GestorParking** tiene una relación de agregación con **Vehículo** y **EspacioParqueo**.
- **GestorParking** interactúa con **BaseDatos** para las operaciones CRUD.
- **InterfazGrafica** se comunica con **GestorParking** para realizar acciones solicitadas por el usuario.

3.4 CÓDIGO IMPLEMENTADO

Descripción del Código

Conexión a la Base de Datos

Se implementó la clase ConexionBD, que establece una conexión con la base de datos gestion_parqueadero en MySQL.

Los parámetros de conexión (URL, usuario y contraseña) se configuran en esta clase.

Gestión de Vehículos

La clase VehiculoDAO incluye dos métodos principales:

registrarVehiculo: Inserta un nuevo vehículo en la tabla Vehiculos.

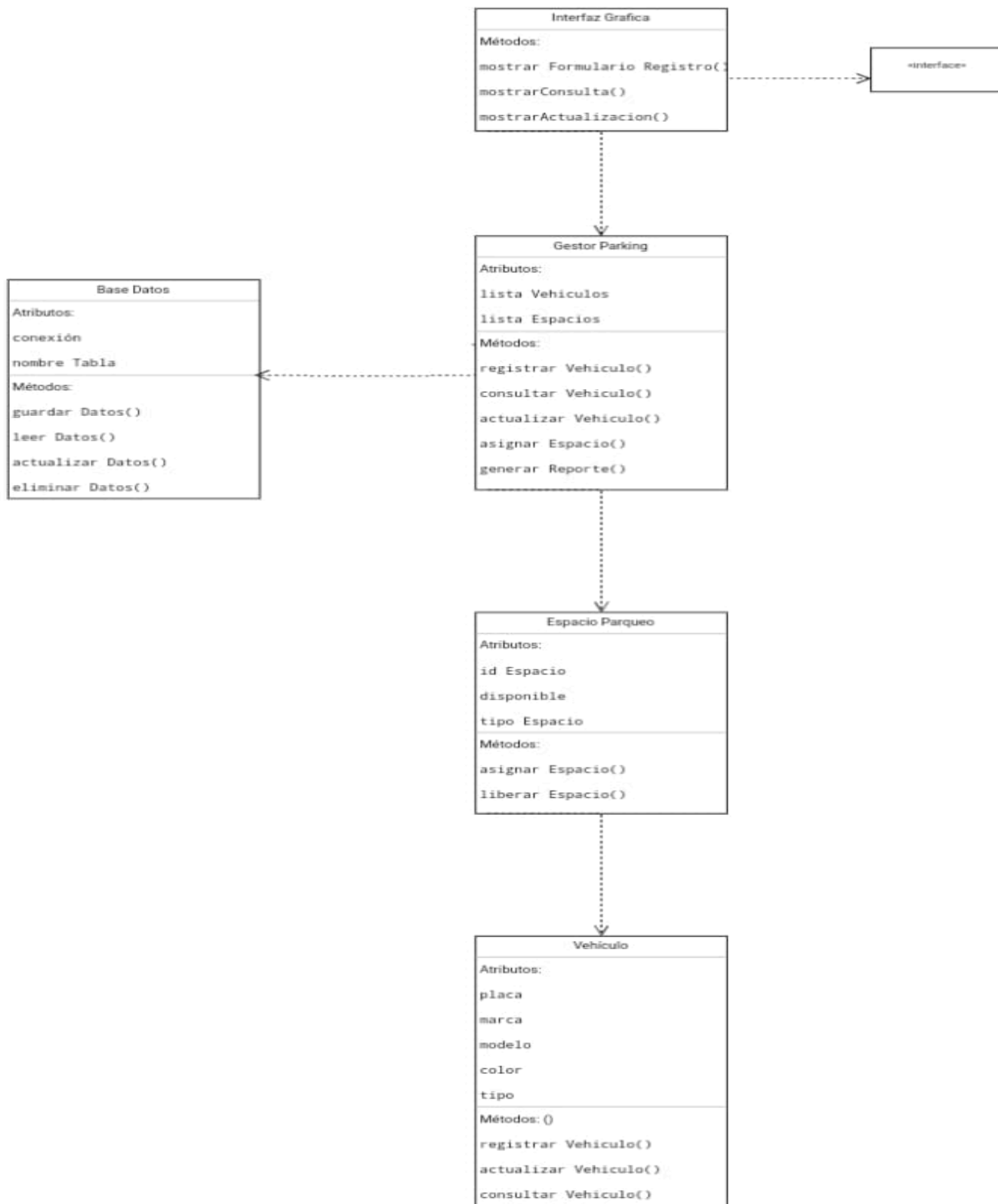
consultarVehiculos: Recupera y muestra la información de todos los vehículos registrados.

Aplicación Principal

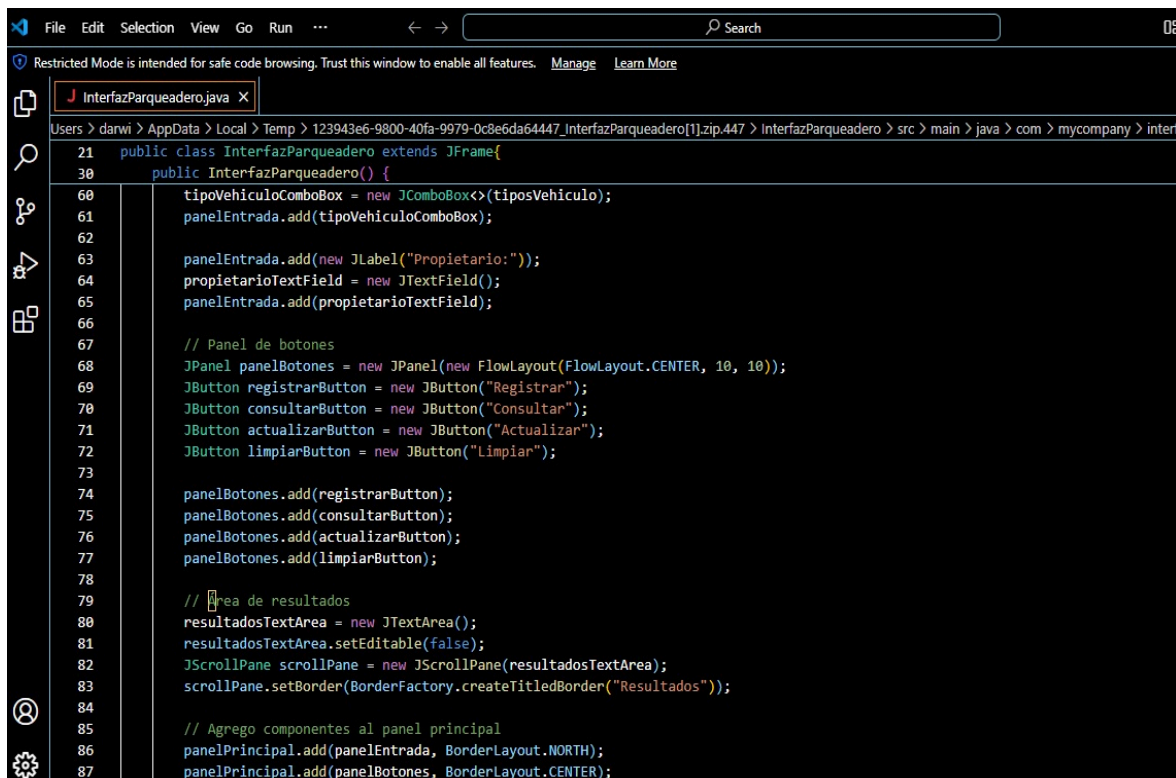
La clase ParqueaderoApp contiene el método main que realiza pruebas iniciales: registra dos vehículos y consulta los datos en la base de datos.

3.5 EVIDENCIAS

GRÁFICO UML



CÓDIGO INTERFAZ



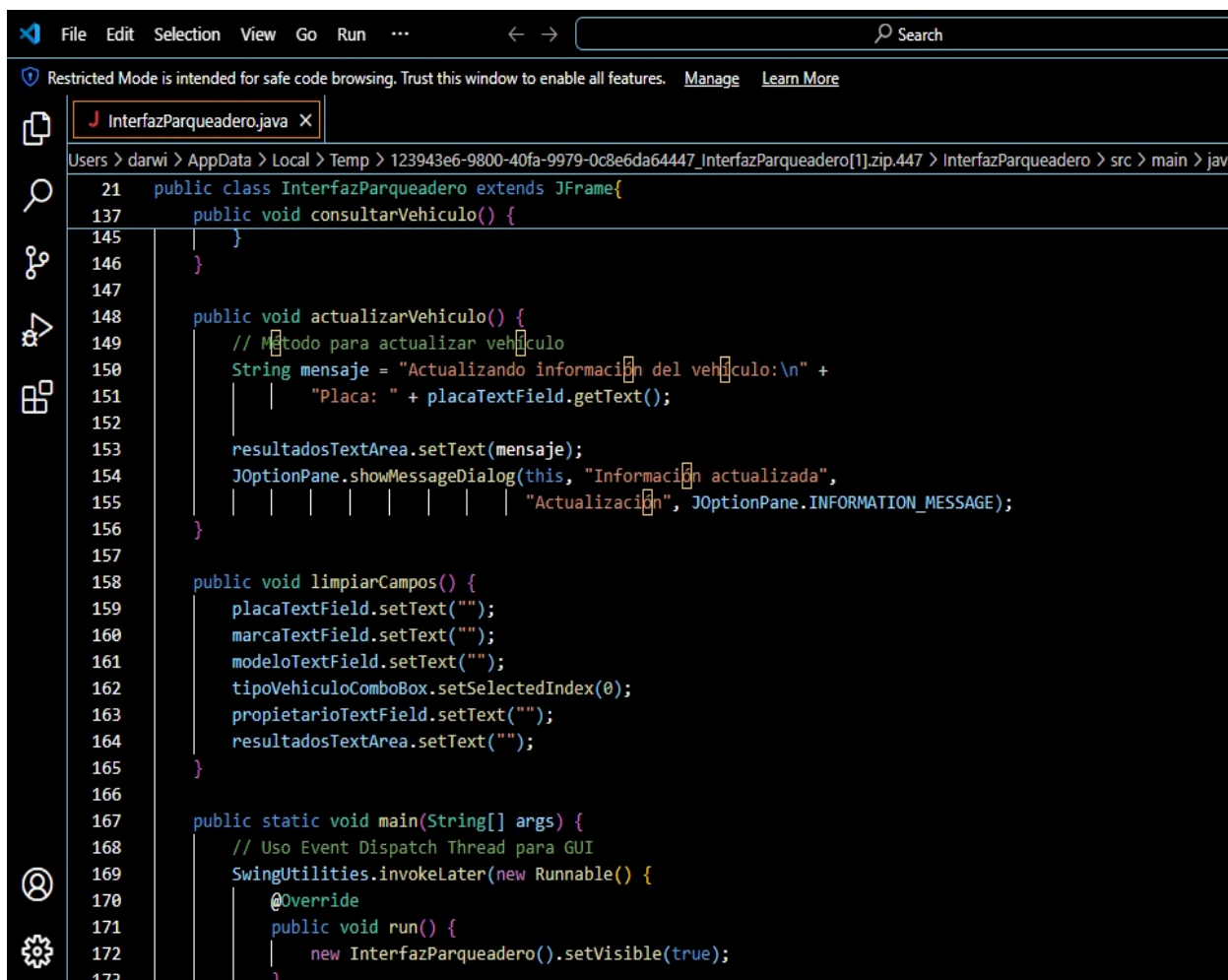
```
File Edit Selection View Go Run ... Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
J InterfazParqueadero.java X
Users > darwi > AppData > Local > Temp > 123943e6-9800-40fa-9979-0c8e6da64447_InterfazParqueadero[1].zip.447 > InterfazParqueadero > src > main > java > com > mycompany > interfaz
21 public class InterfazParqueadero extends JFrame{
30     public InterfazParqueadero() {
60         tipoVehiculoComboBox = new JComboBox<>(tiposVehiculo);
61         panelEntrada.add(tipoVehiculoComboBox);
62
63         panelEntrada.add(new JLabel("Propietario:"));
64         propietarioTextField = new JTextField();
65         panelEntrada.add(proprietarioTextField);
66
67         // Panel de botones
68         JPanel panelBotones = new JPanel(new FlowLayout(FlowLayout.CENTER, 10, 10));
69         JButton registrarButton = new JButton("Registrar");
70         JButton consultarButton = new JButton("Consultar");
71         JButton actualizarButton = new JButton("Actualizar");
72         JButton limpiarButton = new JButton("Limpiar");
73
74         panelBotones.add(registrarButton);
75         panelBotones.add(consultarButton);
76         panelBotones.add(actualizarButton);
77         panelBotones.add(limpiarButton);
78
79         // Área de resultados
80         resultadosTextArea = new JTextArea();
81         resultadosTextArea.setEditable(false);
82         JScrollPane scrollPane = new JScrollPane(resultadosTextArea);
83         scrollPane.setBorder(BorderFactory.createTitledBorder("Resultados"));
84
85         // Agrego componentes al panel principal
86         panelPrincipal.add(panelEntrada, BorderLayout.NORTH);
87         panelPrincipal.add(panelBotones, BorderLayout.CENTER);
```

```
File Edit Selection View Go Run ... Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
J InterfazParqueadero.java X
Users > darwi > AppData > Local > Temp > 123943e6-9800-40fa-9979-0c8e6da64447_InterfazParqueadero[1].zip.447 > InterfazParqueadero > src > main > java > com > mycompany > interfazparqueadero

21 public class InterfazParqueadero extends JFrame{
30     public InterfazParqueadero() {
88         panelPrincipal.add(scrollPane, BorderLayout.SOUTH);
89
90         // Agrego listeners de eventos
91         registrarButton.addActionListener(new ActionListener() {
92             @Override
93             public void actionPerformed(ActionEvent e) {
94                 registrarVehiculo();
95             }
96         });
97
98         consultarButton.addActionListener(new ActionListener() {
99             @Override
100             public void actionPerformed(ActionEvent e) {
101                 consultarVehiculo();
102             }
103         });
104
105         actualizarButton.addActionListener(new ActionListener() {
106             @Override
107             public void actionPerformed(ActionEvent e) {
108                 actualizarVehiculo();
109             }
110         });
111
112         limpiarButton.addActionListener(new ActionListener() {
113             @Override
114             public void actionPerformed(ActionEvent e) {
115                 limpiarCampos();
116         });
117     }
118 }
```



```
File Edit Selection View Go Run ... Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
InterfazParqueadero.java x
Users > darwi > AppData > Local > Temp > 123943e6-9800-40fa-9979-0c8e6da64447_InterfazParqueadero[1].zip.447 > InterfazParqueadero > src > main > java > com > mycompa
21 public class InterfazParqueadero extends JFrame{
30 public InterfazParqueadero() {
114 // Agrego panel principal a la ventana
117 add(panelPrincipal);
118 }
119
120 public void registrarVehiculo() {
121 // Metodo para registrar vehiculo
122 String mensaje = "Registrando vehiculo:\n" +
123     "Placa: " + placaTextField.getText() + "\n" +
124     "Marca: " + marcaTextField.getText() + "\n" +
125     "Modelo: " + modeloTextField.getText() + "\n" +
126     "Tipo: " + tipoVehiculoComboBox.getSelectedItem() + "\n" +
127     "Propietario: " + propietarioTextField.getText();
128 resultadosTextArea.setText(mensaje);
129 JOptionPane.showMessageDialog(this, "Vehiculo registrado exitosamente",
130     "Registro Exitoso", JOptionPane.INFORMATION_MESSAGE);
131 }
132
133 public void consultarVehiculo() {
134 // Metodo para consultar vehiculo
135 String placa = placaTextField.getText();
136 if (!placa.isEmpty()) {
137     resultadosTextArea.setText("Consultando vehiculo con placa: " + placa);
138 } else {
139     JOptionPane.showMessageDialog(this, "Ingrese una placa para consultar",
140     "Error", JOptionPane.ERROR_MESSAGE);
141 }
```



```
File Edit Selection View Go Run ... Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

J InterfazParquadero.java X
Users > darwi > AppData > Local > Temp > 123943e6-9800-40fa-9979-0c8e6da64447_InterfazParquadero[1].zip.447 > InterfazParquadero > src > main > java

21 public class InterfazParquadero extends JFrame{
137     public void consultarVehiculo() {
145     }
146 }
147
148     public void actualizarVehiculo() {
149         // Metodo para actualizar vehiculo
150         String mensaje = "Actualizando informaci3n del vehiculo:\n" +
151             | "Placa: " + placaTextField.getText();
152
153         resultadosTextArea.setText(mensaje);
154         JOptionPane.showMessageDialog(this, "Informaci3n actualizada",
155             | | | | | | | "Actualizaci3n", JOptionPane.INFORMATION_MESSAGE);
156     }
157
158     public void limpiarCampos() {
159         placaTextField.setText("");
160         marcaTextField.setText("");
161         modeloTextField.setText("");
162         tipoVehiculoComboBox.setSelectedIndex(0);
163         propietarioTextField.setText("");
164         resultadosTextArea.setText("");
165     }
166
167     public static void main(String[] args) {
168         // Uso Event Dispatch Thread para GUI
169         SwingUtilities.invokeLater(new Runnable() {
170             @Override
171             public void run() {
172                 new InterfazParquadero().setVisible(true);
173             }
174         });
175     }
176 }
```

CÓDIGO BASE DE DATOS

```

File Edit Selection View Go Run ... Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
J base_datos[1].java X
C: > Users > darwi > AppData > Local > Microsoft > Windows > INetCache > IE > ZZCMXCKV > J base_datos[1].java

1 import java.sql.*;
2
3 class ConexionBD {
4     static final String URL = "jdbc:mysql://localhost:3306/gestion_parqueadero";
5     static final String USUARIO = "root"; // Cambiar según configuración
6     static final String CONTRASENA = ""; // Cambiar según configuración
7
8     static Connection obtenerConexion() throws SQLException {
9         return DriverManager.getConnection(URL, USUARIO, CONTRASENA);
10    }
11 }
12
13 class VehiculoDAO {
14     static void registrarVehiculo(String placa, String tipo, String marca, String modelo) {
15         String sql = "INSERT INTO Vehiculos (placa, tipo, marca, modelo) VALUES (?, ?, ?, ?)";
16         try (Connection conn = ConexionBD.obtenerConexion();
17             PreparedStatement stmt = conn.prepareStatement(sql)) {
18             stmt.setString(1, placa);
19             stmt.setString(2, tipo);
20             stmt.setString(3, marca);
21             stmt.setString(4, modelo);
22             stmt.executeUpdate();
23         } catch (SQLException e) {
24             e.printStackTrace();
25         }
26     }
27
28     static void consultarVehiculos() {
29         String sql = "SELECT * FROM Vehiculos";
30         try (Connection conn = ConexionBD.obtenerConexion();

```

```

File Edit Selection View Go Run ... Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
J base_datos[1].java X
C: > Users > darwi > AppData > Local > Microsoft > Windows > INetCache > IE > ZZCMXCKV > J base_datos[1].java

13 class VehiculoDAO {
28     static void consultarVehiculos() {
29         try (Connection conn = ConexionBD.obtenerConexion();
30             Statement stmt = conn.createStatement();
31             ResultSet rs = stmt.executeQuery(sql)) {
32             while (rs.next()) {
33                 System.out.println("Placa: " + rs.getString("placa") +
34                                     ", Tipo: " + rs.getString("tipo") +
35                                     ", Marca: " + rs.getString("marca") +
36                                     ", Modelo: " + rs.getString("modelo"));
37             }
38         } catch (SQLException e) {
39             e.printStackTrace();
40         }
41     }
42 }
43
44
45 class ParqueaderoApp {
46     static void main(String[] args) {
47         VehiculoDAO.registrarVehiculo("ABC123", "Automovil", "Toyota", "Corolla");
48         VehiculoDAO.registrarVehiculo("XYZ789", "Moto", "Honda", "CBR500R");
49
50         System.out.println("Listado de vehiculos registrados:");
51         VehiculoDAO.consultarVehiculos();
52     }
53 }
54

```

4. *MATERIALES*

Vamos a necesitar Visual Studio Code

5. *CONCLUSIONES*

- El desarrollo de un sistema de parqueadero utilizando POO facilita el diseño del software en la que el software complejo se divide en módulos más pequeños , lo que permite la extensión y el mantenimiento del software en el futuro.
- La integración de una base de datos asegura el manejo eficiente y seguro de la información de los vehículos y la disponibilidad de espacios.
- La implementación de un sistema automatizado optimiza el tiempo y reduce los errores humanos en la gestión del parqueadero.

6. *RECOMENDACIONES*

- Realizar pruebas minuciosas del sistema para garantizar su correcto funcionamiento antes de su implementación en un entorno real.
- Diseñar la interfaz gráfica de manera intuitiva, priorizando la simplicidad y facilidad de navegación.
- Considerar la posibilidad de escalar el sistema en el futuro, incorporando nuevas funcionalidades como la reserva de espacios y el pago en línea.

7. BIBLIOGRAFÍAS

Caballero, G. (s. f.). Proyecto_POO. Scribd.

<https://es.scribd.com/document/535524706/Proyecto-POO>

(N.d.-c). Udemy.com. Retrieved December 13, 2024, from

<https://www.udemy.com/course/sistema-de-control-de-parqueo-en-java-mysql-poo-con-codigo/?srsltid=AfmBOoqw96tGy-oIL5daWloOHMswNg2RCyZZ4t9l2MTsMk493KepaRgQ&couponCode=ST19MT121224>

Prezi, E. C. O. (s. f.). *Proyecto POO-Sistema Gestión Parquadero*. prezi.com.

https://prezi.com/p/x_ellxgqxyml/proyecto-poo-sistema-gestion-parquadero/