

```
1 module com.msebastiao.sap {  
2     requires javafx.controls;  
3     requires javafx.fxml;  
4     requires java.sql;  
5  
6     opens com.msebastiao.sap to javafx.fxml;  
7     opens com.msebastiao.sap.controller to javafx.fxml;  
8     opens com.msebastiao.sap.model to javafx.base;  
9  
10    exports com.msebastiao.sap;  
11    exports com.msebastiao.sap.controller;  
12    exports com.msebastiao.sap.model;  
13    exports com.msebastiao.sap.dao;  
14 }  
15
```



```
65     MenuItem gestionarConvenios = new MenuItem("Gestionar Convenios");
66     gestionarConvenios.setDisable(true);
67     MenuItem configurarSistema = new MenuItem("Configurar Sistema");
68     configurarSistema.setDisable(true);
69     MenuItem administrarUsuarios = new MenuItem("Administrar Usuarios");
70     administrarUsuarios.setDisable(true);
71     MenuItem generarInformeItem = new MenuItem("Emitir Informes Mensuales");
72     administracionMenu.getItems().addAll(gestionarConvenios, configurarSistema
73 , administrarUsuarios);
73     generarInformeItem.setOnAction(e -> openView("Administración - Emitir
74     Informes Mensuales", "GenerarInformeMensualView"));
74     administracionMenu.getItems().add(generarInformeItem);
75
76     // Agregando menús a la barra de menús
77     menuBar.getMenus().addAll(inicioMenu, clientesMenu, recepcionMenu,
78     tallerMenu, operacionesMenu, administracionMenu);
78
79     root.setTop(menuBar);
80     root.setCenter(tabPane);
81
82     Scene scene = new Scene(root, 630, 320);
83     primaryStage.setScene(scene);
84     primaryStage.show();
85 }
86
87 private void openView(String title, String viewFXML) {
88     Pane view = (Pane) ViewFactory.createView(viewFXML);
89     OpenViewCommand command = new OpenViewCommand(tabPane, title, view);
90     command.execute();
91 }
92
93 public static void main(String[] args) {
94     launch(args);
95 }
96 }
97 }
```

```
1 package com.msebastiao.sap.dao;
2
3 import java.util.List;
4
5 /**
6  * Interface que define las operaciones básicas del patrón Data Access Object (DAO).
7  *
8  * @param <T> clase del objeto que se va a manipular en la base de datos (ej. Turno
9  , Mecanico, etc.)
10 */
11 public interface DAO<T> {
12     void insert(T t) throws Exception;
13     T getById(int id) throws Exception;
14     List<T> getAll() throws Exception;
15     void update(T t) throws Exception;
16     void delete(int id) throws Exception;
17 }
18
19
20 }
21
```

```

1 package com.msebastiao.sap.dao;
2
3 import com.msebastiao.sap.database.DatabaseConnection;
4 import com.msebastiao.sap.model.TitularVehiculo;
5 import com.msebastiao.sap.model.Turno;
6
7 import java.sql.*;
8 import java.time.LocalDate;
9 import java.time.LocalTime;
10 import java.util.ArrayList;
11 import java.util.List;
12
13 public class TurnoDAO implements DAO<Turno> {
14
15     private final Connection connection;
16
17     public TurnoDAO() throws SQLException {
18         this.connection = DatabaseConnection.getInstance().getConnection();
19     }
20
21     @Override
22     public void insert(Turno turno) throws SQLException {
23         String query = "INSERT INTO turnos (fecha, hora_inicio, hora_fin, estado, "
24         titular_vehiculo_id) VALUES (?, ?, ?, ?, ?)";
25         try (PreparedStatement stmt = connection.prepareStatement(query, Statement.
26 RETURN_GENERATED_KEYS)) {
27             stmt.setDate(1, Date.valueOf(turno.getFecha()));
28             stmt.setTime(2, Time.valueOf(turno.getHoraInicio()));
29             stmt.setTime(3, Time.valueOf(turno.getHoraFin()));
30             stmt.setString(4, turno.getEstado());
31             if (turno.getTitularVehiculo() != null) {
32                 stmt.setInt(5, turno.getTitularVehiculo().getId());
33             } else {
34                 stmt.setNull(5, Types.INTEGER);
35             }
36             stmt.executeUpdate();
37
38             try (ResultSet generatedKeys = stmt.getGeneratedKeys()) {
39                 if (generatedKeys.next()) {
40                     turno.setId(generatedKeys.getInt(1));
41                 }
42             }
43
44     }
45     @Override
46     public Turno getById(int id) throws SQLException {
47         String query = "SELECT * FROM turnos WHERE id = ?";
48         try (PreparedStatement stmt = connection.prepareStatement(query)) {
49             stmt.setInt(1, id);
50             try (ResultSet rs = stmt.executeQuery()) {
51                 if (rs.next()) {
52                     int agendaId = rs.getInt("agenda_id");
53                     LocalDate fecha = rs.getDate("fecha").toLocalDate();
54                     LocalTime horaInicio = rs.getTime("hora_inicio").toLocalTime();
55                     LocalTime horaFin = rs.getTime("hora_fin").toLocalTime();
56                     String estado = rs.getString("estado");
57                     TitularVehiculoDAO titularVehiculoDAO = new TitularVehiculoDAO();
58                     TitularVehiculo titularVehiculo = titularVehiculoDAO.getById(rs.
59                     getInt("titular_vehiculo_id"));
60                     return new Turno(id, fecha, horaInicio, horaFin, estado,
61                     titularVehiculo);
62                 }
63             }
64         }
65     }
66     @Override

```

```
66     public List<Turno> getAll() throws SQLException {
67         List<Turno> turnos = new ArrayList<>();
68         String query = "SELECT * FROM turnos";
69         try (Statement stmt = connection.createStatement(); ResultSet rs = stmt.
70             executeQuery(query)) {
71             while (rs.next()) {
72                 int id = rs.getInt("id");
73                 LocalDate fecha = rs.getDate("fecha").toLocalDate();
74                 LocalTime horaInicio = rs.getTime("hora_inicio").toLocalTime();
75                 LocalTime horaFin = rs.getTime("hora_fin").toLocalTime();
76                 String estado = rs.getString("estado");
77                 TitularVehiculoDAO titularVehiculoDAO = new TitularVehiculoDAO();
78                 TitularVehiculo titularVehiculo = titularVehiculoDAO.getById(rs.
79                     getInt("titular_vehiculo_id"));
80                 turnos.add(new Turno(id, fecha, horaInicio, horaFin, estado,
81                     titularVehiculo));
82             }
83         }
84
85     @Override
86     public void update(Turno turno) throws SQLException {
87         String query = "UPDATE turnos SET fecha = ?, hora_inicio = ?, hora_fin
88         = ?, estado = ?, titular_vehiculo_id = ? WHERE id = ?";
89         try (PreparedStatement stmt = connection.prepareStatement(query)) {
90             stmt.setDate(1, Date.valueOf(turno.getFecha()));
91             stmt.setTime(2, Time.valueOf(turno.getHoraInicio()));
92             stmt.setTime(3, Time.valueOf(turno.getHoraFin()));
93             stmt.setString(4, turno.getEstado());
94             if (turno.getTitularVehiculo() != null) {
95                 stmt.setInt(5, turno.getTitularVehiculo().getId());
96             } else {
97                 stmt.setNull(5, Types.INTEGER);
98             }
99             stmt.setInt(6, turno.getId());
100            stmt.executeUpdate();
101        }
102    }
103    @Override
104    public void delete(int id) throws SQLException {
105        String query = "DELETE FROM turnos WHERE id = ?";
106        try (PreparedStatement stmt = connection.prepareStatement(query)) {
107            stmt.setInt(1, id);
108            stmt.executeUpdate();
109        }
110    }
111 }
112 }
```

```

1 package com.msebastiao.sap.dao;
2
3 import com.msebastiao.sap.database.DatabaseConnection;
4 import com.msebastiao.sap.model.Informe;
5
6 import java.sql.*;
7 import java.time.LocalDate;
8 import java.util.ArrayList;
9 import java.util.List;
10
11 public class InformeDAO implements DAO<Informe> {
12
13     private final Connection connection;
14
15     public InformeDAO() throws SQLException {
16         this.connection = DatabaseConnection.getInstance().getConnection();
17     }
18
19     @Override
20     public void insert(Informe informe) throws Exception {
21         String query = "INSERT INTO informes (fecha, contenido) VALUES (?, ?)";
22         try (PreparedStatement stmt = connection.prepareStatement(query)) {
23             stmt.setDate(1, Date.valueOf(informe.getFecha()));
24             stmt.setString(2, informe.getContenido());
25             stmt.executeUpdate();
26         }
27     }
28
29     @Override
30     public Informe getById(int id) throws Exception {
31         String query = "SELECT * FROM informes WHERE id = ?";
32         try (PreparedStatement stmt = connection.prepareStatement(query)) {
33             stmt.setInt(1, id);
34             try (ResultSet rs = stmt.executeQuery()) {
35                 if (rs.next()) {
36                     LocalDate fecha = rs.getDate("fecha").toLocalDate();
37                     String contenido = rs.getString("contenido");
38                     return new Informe(fecha, contenido);
39                 }
40             }
41         }
42         return null;
43     }
44
45     @Override
46     public List<Informe> getAll() throws Exception {
47         List<Informe> informes = new ArrayList<>();
48         String query = "SELECT * FROM informes";
49         try (Statement stmt = connection.createStatement();
50              ResultSet rs = stmt.executeQuery(query)) {
51             while (rs.next()) {
52                 LocalDate fecha = rs.getDate("fecha").toLocalDate();
53                 String contenido = rs.getString("contenido");
54                 informes.add(new Informe(fecha, contenido));
55             }
56         }
57         return informes;
58     }
59
60     @Override
61     public void update(Informe informe) throws Exception {
62         String query = "UPDATE informes SET fecha = ?, contenido = ? WHERE id = ?";
63         try (PreparedStatement stmt = connection.prepareStatement(query)) {
64             stmt.setDate(1, Date.valueOf(informe.getFecha()));
65             stmt.setString(2, informe.getContenido());
66             stmt.setInt(3, informe.getId());
67             stmt.executeUpdate();
68         }
69     }
70 }
```

```
71     @Override
72     public void delete(int id) throws Exception {
73         String query = "DELETE FROM informes WHERE id = ?";
74         try (PreparedStatement stmt = connection.prepareStatement(query)) {
75             stmt.setInt(1, id);
76             stmt.executeUpdate();
77         }
78     }
79 }
80
```

```

1 package com.msebastiao.sap.dao;
2
3 import com.msebastiao.sap.database.DatabaseConnection;
4 import com.msebastiao.sap.model.Mecanico;
5
6 import java.sql.*;
7 import java.util.ArrayList;
8 import java.util.List;
9
10 public class MecanicoDAO implements DAO<Mecanico> {
11
12     private final Connection connection;
13
14     public MecanicoDAO() throws SQLException {
15         this.connection = DatabaseConnection.getInstance().getConnection();
16     }
17
18     @Override
19     public void insert(Mecanico mecanico) throws Exception {
20         String query = "INSERT INTO mecanicos (nombre, apellido) VALUES (?, ?)";
21         try (PreparedStatement stmt = connection.prepareStatement(query, Statement.RETURN_GENERATED_KEYS)) {
22             stmt.setString(1, mecanico.getNombre());
23             stmt.setString(2, mecanico.getApellido());
24             stmt.executeUpdate();
25
26             try (ResultSet generatedKeys = stmt.getGeneratedKeys()) {
27                 if (generatedKeys.next()) {
28                     mecanico.setId(generatedKeys.getInt(1));
29                 }
30             }
31         }
32     }
33
34     @Override
35     public Mecanico getById(int id) throws Exception {
36         String query = "SELECT * FROM mecanicos WHERE id = ?";
37         try (PreparedStatement stmt = connection.prepareStatement(query)) {
38             stmt.setInt(1, id);
39             try (ResultSet rs = stmt.executeQuery()) {
40                 if (rs.next()) {
41                     String nombre = rs.getString("nombre");
42                     String apellido = rs.getString("apellido");
43                     return new Mecanico(id, nombre, apellido);
44                 }
45             }
46         }
47         return null;
48     }
49
50     @Override
51     public List<Mecanico> getAll() throws Exception {
52         List<Mecanico> mecanicos = new ArrayList<>();
53         String query = "SELECT * FROM mecanicos";
54         try (Statement stmt = connection.createStatement();
55              ResultSet rs = stmt.executeQuery(query)) {
56             while (rs.next()) {
57                 int id = rs.getInt("id");
58                 String nombre = rs.getString("nombre");
59                 String apellido = rs.getString("apellido");
60                 mecanicos.add(new Mecanico(id, nombre, apellido));
61             }
62         }
63         return mecanicos;
64     }
65
66     @Override
67     public void update(Mecanico mecanico) throws Exception {
68         String query = "UPDATE mecanicos SET nombre = ?, apellido = ? WHERE id = ?";
69     }

```

```
69     try (PreparedStatement stmt = connection.prepareStatement(query)) {
70         stmt.setString(1, mecanico.getNombre());
71         stmt.setString(2, mecanico.getApellido());
72         stmt.setInt(3, mecanico.getId());
73         stmt.executeUpdate();
74     }
75 }
76
77 @Override
78 public void delete(int id) throws Exception {
79     String query = "DELETE FROM mechanicos WHERE id = ?";
80     try (PreparedStatement stmt = connection.prepareStatement(query)) {
81         stmt.setInt(1, id);
82         stmt.executeUpdate();
83     }
84 }
85 }
```



```
67         while (rs.next()) {
68             int id = rs.getInt("id");
69             String marca = rs.getString("marca");
70             String modelo = rs.getString("modelo");
71             int anio = rs.getInt("anio");
72             int titularId = rs.getInt("titular_id");
73             TitularVehiculoDAO titularVehiculoDAO = new TitularVehiculoDAO();
74             TitularVehiculo titularVehiculo = titularVehiculoDAO.getById(
75                 titularId);
76             vehiculos.add(new Vehiculo(id, marca, modelo, anio,
77                 titularVehiculo));
78         }
79     }
80
81     @Override
82     public void update(Vehiculo vehiculo) throws SQLException {
83         String query = "UPDATE vehiculos SET marca = ?, modelo = ?, anio = ?,
84         titular_vehiculo_id = ? WHERE id = ?";
85         try (PreparedStatement stmt = connection.prepareStatement(query)) {
86             stmt.setString(1, vehiculo.getMarca());
87             stmt.setString(2, vehiculo.getModelo());
88             stmt.setInt(3, vehiculo.getAnio());
89             if (vehiculo.getTitularVehiculo() == null) {
90                 stmt.setNull(4, Types.INTEGER);
91             } else {
92                 stmt.setInt(4, vehiculo.getTitularVehiculo().getId());
93             }
94             stmt.setInt(5, vehiculo.getId());
95             stmt.executeUpdate();
96         }
97     }
98     @Override
99     public void delete(int id) throws SQLException {
100        String query = "DELETE FROM vehiculos WHERE id = ?";
101        try (PreparedStatement stmt = connection.prepareStatement(query)) {
102            stmt.setInt(1, id);
103            stmt.executeUpdate();
104        }
105    }
106 }
```

```
1 package com.msebastiao.sap.dao;
2
3 import com.msebastiao.sap.database.DatabaseConnection;
4 import com.msebastiao.sap.model.Actividad;
5
6 import java.sql.*;
7 import java.util.ArrayList;
8 import java.util.List;
9
10 public class ActividadDAO implements DAO<Actividad> {
11
12     private final Connection connection;
13
14     public ActividadDAO() throws SQLException {
15         this.connection = DatabaseConnection.getInstance().getConnection();
16     }
17
18     @Override
19     public void insert(Actividad actividad) throws Exception {
20         String query = "INSERT INTO actividades (descripcion, tiempo_empleado,
21         estado, ficha_mecanica_id) VALUES (?, ?, ?, ?)";
22         try (PreparedStatement stmt = connection.prepareStatement(query, Statement.
23         RETURN_GENERATED_KEYS)) {
24             stmt.setString(1, actividad.getDescripcion());
25             stmt.setInt(2, actividad.getTiempoEmpleado());
26             stmt.setString(3, actividad.getEstado());
27             stmt.setInt(4, actividad.getFichaMecanicaId());
28             stmt.executeUpdate();
29
30             try (ResultSet generatedKeys = stmt.getGeneratedKeys()) {
31                 if (generatedKeys.next()) {
32                     actividad.setId(generatedKeys.getInt(1));
33                 }
34             }
35         }
36
37     @Override
38     public Actividad getById(int id) throws Exception {
39         String query = "SELECT * FROM actividades WHERE id = ?";
40         try (PreparedStatement stmt = connection.prepareStatement(query)) {
41             stmt.setInt(1, id);
42             try (ResultSet rs = stmt.executeQuery()) {
43                 if (rs.next()) {
44                     String descripcion = rs.getString("descripcion");
45                     int tiempoEmpleado = rs.getInt("tiempo_empleado");
46                     String estado = rs.getString("estado");
47                     int fichaMecanicaId = rs.getInt("ficha_mecanica_id");
48                     return new Actividad(id, descripcion, tiempoEmpleado, estado,
49                     fichaMecanicaId);
50                 }
51             }
52         }
53
54     @Override
55     public List<Actividad> getAll() throws Exception {
56         List<Actividad> actividades = new ArrayList<>();
57         String query = "SELECT * FROM Actividades";
58         try (Statement stmt = connection.createStatement();
59             ResultSet rs = stmt.executeQuery(query)) {
60             while (rs.next()) {
61                 int id = rs.getInt("id");
62                 String descripcion = rs.getString("descripcion");
63                 int tiempoEmpleado = rs.getInt("tiempo_empleado");
64                 String estado = rs.getString("estado");
65                 int fichaMecanicaId = rs.getInt("ficha_mecanica_id");
66                 actividades.add(new Actividad(id, descripcion, tiempoEmpleado,
67                     estado, fichaMecanicaId));
68             }
69         }
70     }
71 }
```

```
67        }
68    }
69    return actividades;
70  }
71
72  @Override
73  public void update(Actividad actividad) throws Exception {
74    String query = "UPDATE actividades SET descripcion = ?, tiempo_empleado
= ?, estado = ?, ficha_mecanica_id = ? WHERE id = ?";
75    try (PreparedStatement stmt = connection.prepareStatement(query)) {
76      stmt.setString(1, actividad.getDescripcion());
77      stmt.setInt(2, actividad.getTiempoEmpleado());
78      stmt.setString(3, actividad.getEstado());
79      stmt.setInt(4, actividad.getFichaMecanicaId());
80      stmt.setInt(5, actividad.getId());
81      stmt.executeUpdate();
82    }
83  }
84
85  @Override
86  public void delete(int id) throws Exception {
87    String query = "DELETE FROM actividades WHERE id = ?";
88    try (PreparedStatement stmt = connection.prepareStatement(query)) {
89      stmt.setInt(1, id);
90      stmt.executeUpdate();
91    }
92  }
93
94  public List<Actividad> getByFichaMecanicaId(int fichaMecanicaId) throws
Exception {
95    List<Actividad> actividades = new ArrayList<>();
96    String query = "SELECT * FROM actividades WHERE fichaMecanica_id = ?";
97    try (PreparedStatement stmt = connection.prepareStatement(query)) {
98      stmt.setInt(1, fichaMecanicaId);
99      try (ResultSet rs = stmt.executeQuery()) {
100        while (rs.next()) {
101          int id = rs.getInt("id");
102          String descripcion = rs.getString("descripcion");
103          int tiempoEmpleado = rs.getInt("tiempo_empleado");
104          String estado = rs.getString("estado");
105          actividades.add(new Actividad(id, descripcion, tiempoEmpleado
, estado, fichaMecanicaId));
106        }
107      }
108    }
109    return actividades;
110  }
111 }
```

```

1 package com.msebastiao.sap.dao;
2
3 import com.msebastiao.sap.database.DatabaseConnection;
4 import com.msebastiao.sap.model.Repuestos;
5
6 import java.sql.*;
7 import java.util.ArrayList;
8 import java.util.List;
9
10 public class RepuestosDAO implements DAO<Repuestos> {
11
12     private final Connection connection;
13
14     public RepuestosDAO() throws SQLException {
15         this.connection = DatabaseConnection.getInstance().getConnection();
16     }
17
18     @Override
19     public void insert(Repuestos repuesto) throws Exception {
20         String query = "INSERT INTO repuestos (nombre, cantidad, ficha_mecanica_id"
21         ) VALUES (?, ?, ?)";
22         try (PreparedStatement stmt = connection.prepareStatement(query, Statement.
23             RETURN_GENERATED_KEYS)) {
24             stmt.setString(1, repuesto.getNombre());
25             stmt.setInt(2, repuesto.getCantidad());
26             stmt.setInt(3, repuesto.getFichaMecanicaId());
27             stmt.executeUpdate();
28
29             try (ResultSet generatedKeys = stmt.getGeneratedKeys()) {
30                 if (generatedKeys.next()) {
31                     repuesto.setId(generatedKeys.getInt(1));
32                 }
33             }
34         }
35
36     @Override
37     public Repuestos getById(int id) throws Exception {
38         String query = "SELECT * FROM repuestos WHERE id = ?";
39         try (PreparedStatement stmt = connection.prepareStatement(query)) {
40             stmt.setInt(1, id);
41             try (ResultSet rs = stmt.executeQuery()) {
42                 if (rs.next()) {
43                     String nombre = rs.getString("nombre");
44                     int cantidad = rs.getInt("cantidad");
45                     int fichaMecanicaId = rs.getInt("ficha_mecanica_id");
46                     return new Repuestos(id, nombre, cantidad, fichaMecanicaId);
47                 }
48             }
49             return null;
50         }
51
52     @Override
53     public List<Repuestos> getAll() throws Exception {
54         List<Repuestos> repuestos = new ArrayList<>();
55         String query = "SELECT * FROM repuestos";
56         try (Statement stmt = connection.createStatement();
57             ResultSet rs = stmt.executeQuery(query)) {
58             while (rs.next()) {
59                 int id = rs.getInt("id");
60                 String nombre = rs.getString("nombre");
61                 int cantidad = rs.getInt("cantidad");
62                 int fichaMecanicaId = rs.getInt("ficha_mecanica_id");
63                 repuestos.add(new Repuestos(id, nombre, cantidad, fichaMecanicaId
64             ));
65             }
66         }
67     }
68 }
```

```
68
69     @Override
70     public void update(Repuestos repuesto) throws Exception {
71         String query = "UPDATE repuestos SET nombre = ?, cantidad = ?,
72                                     ficha_mecanica_id = ? WHERE id = ?";
73         try (PreparedStatement stmt = connection.prepareStatement(query)) {
74             stmt.setString(1, repuesto.getNombre());
75             stmt.setInt(2, repuesto.getCantidad());
76             stmt.setInt(3, repuesto.getFichaMecanicaId());
77             stmt.setInt(4, repuesto.getId());
78             stmt.executeUpdate();
79         }
80     }
81     @Override
82     public void delete(int id) throws Exception {
83         String query = "DELETE FROM repuestos WHERE id = ?";
84         try (PreparedStatement stmt = connection.prepareStatement(query)) {
85             stmt.setInt(1, id);
86             stmt.executeUpdate();
87         }
88     }
89
90     public List<Repuestos> getByFichaMecanicaId(int fichaMecanicaId) throws
Exception {
91         List<Repuestos> repuestos = new ArrayList<>();
92         String query = "SELECT * FROM repuestos WHERE ficha_mecanica_id = ?";
93         try (PreparedStatement stmt = connection.prepareStatement(query)) {
94             stmt.setInt(1, fichaMecanicaId);
95             try (ResultSet rs = stmt.executeQuery()) {
96                 while (rs.next()) {
97                     int id = rs.getInt("id");
98                     String nombre = rs.getString("nombre");
99                     int cantidad = rs.getInt("cantidad");
100                    repuestos.add(new Repuestos(id, nombre, cantidad,
fichaMecanicaId));
101                }
102            }
103        }
104        return repuestos;
105    }
106 }
107
```



```

66             TitularVehiculo titularVehiculo = titularVehiculoDAO.getById(
67                 titularVehiculoId);
68
69             VehiculoDAO vehiculoDAO = new VehiculoDAO();
70             Vehiculo vehiculo = vehiculoDAO.getById(vehiculoId);
71
72             ActividadDAO actividadDAO = new ActividadDAO();
73             List<Actividad> actividades = actividadDAO.
74                 getByFichaMecanicaId(id);
75
76             RepuestosDAO repuestosDAO = new RepuestosDAO();
77             List<Repuestos> repuestos = repuestosDAO.getByFichaMecanicaId(
78                 id);
79
80             return new FichaMecanica(id, mecanico, titularVehiculo,
81                 vehiculo, fechaInicio, fechaFin, estado, actividades, repuestos);
82         }
83
84     @Override
85     public List<FichaMecanica> getAll() throws Exception {
86         List<FichaMecanica> fichas = new ArrayList<>();
87         String query = "SELECT * FROM fichas_mecanicas";
88         try (Statement stmt = connection.createStatement());
89             ResultSet rs = stmt.executeQuery(query)) {
90             while (rs.next()) {
91                 int id = rs.getInt("id");
92                 int mecanicoId = rs.getInt("mecanico_id");
93                 int titularVehiculoId = rs.getInt("titular_vehiculo_id");
94                 int vehiculoId = rs.getInt("vehiculo_id");
95                 LocalDate fechaInicio = rs.getDate("fecha_inicio").toLocalDate();
96                 LocalDate fechaFin = rs.getDate("fecha_fin").toLocalDate();
97                 String estado = rs.getString("estado");
98
99                 MecanicoDAO mecanicoDAO = new MecanicoDAO();
100                Mecanico mecanico = mecanicoDAO.getById(mecanicoId);
101
102                TitularVehiculoDAO titularVehiculoDAO = new TitularVehiculoDAO();
103                TitularVehiculo titularVehiculo = titularVehiculoDAO.getById(
104                    titularVehiculoId);
105
106                VehiculoDAO vehiculoDAO = new VehiculoDAO();
107                Vehiculo vehiculo = vehiculoDAO.getById(vehiculoId);
108
109                ActividadDAO actividadDAO = new ActividadDAO();
110                List<Actividad> actividades = actividadDAO.getByFichaMecanicaId(id
111            );
112
113                RepuestosDAO repuestosDAO = new RepuestosDAO();
114                List<Repuestos> repuestos = repuestosDAO.getByFichaMecanicaId(id);
115
116                fichas.add(new FichaMecanica(id, mecanico, titularVehiculo,
117                    vehiculo, fechaInicio, fechaFin, estado, actividades, repuestos));
118            }
119        }
120    @Override
121    public void update(FichaMecanica ficha) throws Exception {
122        String query = "UPDATE fichas_mecanicas SET mecanico_id = ?,"
123        "titular_vehiculo_id = ?, vehiculo_id = ?, fecha_inicio = ?, fecha_fin = ?,"
124        "estado = ? WHERE id = ?";
125        try (PreparedStatement stmt = connection.prepareStatement(query)) {
126            stmt.setInt(1, ficha.getMecanico().getId());
127            stmt.setInt(2, ficha.getTitularVehiculo().getId());
128            stmt.setInt(3, ficha.getVehiculo().getId());

```

```
127         stmt.setDate(4, Date.valueOf(ficha.getFechaInicio()));
128         stmt.setDate(5, Date.valueOf(ficha.getFechaFin()));
129         stmt.setString(6, ficha.getEstado());
130         stmt.setInt(7, ficha.getId());
131         stmt.executeUpdate();
132
133     ActividadDAO actividadDAO = new ActividadDAO();
134     for (Actividad actividad : ficha.getActividadesRealizadas()) {
135         if (actividad.getId() == 0) {
136             actividadDAO.insert(actividad);
137         } else {
138             actividadDAO.update(actividad);
139         }
140     }
141
142     RepuestosDAO repuestosDAO = new RepuestosDAO();
143     for (Repuestos repuesto : ficha.getRepuestosEmpleados()) {
144         if (repuesto.getId() == 0) {
145             repuestosDAO.insert(repuesto);
146         } else {
147             repuestosDAO.update(repuesto);
148         }
149     }
150 }
151
152
153 @Override
154 public void delete(int id) throws Exception {
155     ActividadDAO actividadDAO = new ActividadDAO();
156     List<Actividad> actividades = actividadDAO.getByFichaMecanicaId(id);
157     for (Actividad actividad : actividades) {
158         actividadDAO.delete(actividad.getId());
159     }
160
161     RepuestosDAO repuestosDAO = new RepuestosDAO();
162     List<Repuestos> repuestos = repuestosDAO.getByFichaMecanicaId(id);
163     for (Repuestos repuesto : repuestos) {
164         repuestosDAO.delete(repuesto.getId());
165     }
166
167     String query = "DELETE FROM fichas_mecanicas WHERE id = ?";
168     try (PreparedStatement stmt = connection.prepareStatement(query)) {
169         stmt.setInt(1, id);
170         stmt.executeUpdate();
171     }
172 }
173 }
```



```

66
67         }
68     }
69     return titulares;
70 }
71
72 @Override
73 public void update(TitularVehiculo titularVehiculo) throws SQLException {
74     String query = "UPDATE titular_vehiculo SET dni = ?, nombre = ?, apellido = ? WHERE id = ?";
75     try (PreparedStatement stmt = connection.prepareStatement(query)) {
76         stmt.setString(1, titularVehiculo.getDni());
77         stmt.setString(2, titularVehiculo.getNombre());
78         stmt.setString(3, titularVehiculo.getApellido());
79         stmt.setInt(4, titularVehiculo.getId());
80         stmt.executeUpdate();
81     }
82 }
83
84 @Override
85 public void delete(int id) throws SQLException {
86     String query = "DELETE FROM titular_vehiculo WHERE id = ?";
87     try (PreparedStatement stmt = connection.prepareStatement(query)) {
88         stmt.setInt(1, id);
89         stmt.executeUpdate();
90     }
91 }
92
93 public TitularVehiculo getByDni(String dni) throws SQLException {
94     String query = "SELECT * FROM titular_vehiculo WHERE dni = ?";
95     try (PreparedStatement stmt = connection.prepareStatement(query)) {
96         stmt.setString(1, dni);
97         try (ResultSet rs = stmt.executeQuery()) {
98             if (rs.next()) {
99                 int id = rs.getInt("id");
100                String nombre = rs.getString("nombre");
101                String apellido = rs.getString("apellido");
102                TitularVehiculo titularVehiculo = new TitularVehiculo(id, dni,
103 , nombre, apellido);
104                titularVehiculo.setVehiculos(getVehiculosByTitularId(
105 titularVehiculo));
106                return titularVehiculo;
107            }
108        }
109    }
110
111     private List<Vehiculo> getVehiculosByTitularId(TitularVehiculo titularVehiculo
112 ) throws SQLException {
113     List<Vehiculo> vehiculos = new ArrayList<>();
114     String query = "SELECT * FROM vehiculos WHERE titular_vehiculo_id = ?";
115     try (PreparedStatement stmt = connection.prepareStatement(query)) {
116         stmt.setInt(1, titularVehiculo.getId());
117         try (ResultSet rs = stmt.executeQuery()) {
118             while (rs.next()) {
119                 int id = rs.getInt("id");
120                 String marca = rs.getString("marca");
121                 String modelo = rs.getString("modelo");
122                 int anio = rs.getInt("anio");
123                 vehiculos.add(new Vehiculo(id, marca, modelo, anio,
124 titularVehiculo));
125             }
126         }
127     }
128 }
129 }
```

```
1 package com.msebastiao.sap.view;
2
3 import javafx.scene.Parent;
4
5 /**
6  * El patrón Factory Method define una interfaz para crear un objeto, pero deja que
7  * las subclases decidan qué clase
8  * instanciar.
9  * Se usa para crear las vistas basadas en FXML.
10 */
11
12 /**
13  * Crea una vista dada su nombre y la devuelve en un objeto de tipo Parent para
14  * ser renderizado en la interfaz
15  * de usuario.
16  *
17  * @param viewName nombre de la vista
18  * @return objeto de tipo {@link Parent}
19  */
20 public static Parent createView(String viewName) {
21     return ViewManager.getInstance().getView(viewName);
22 }
23
```

```
1 package com.msebastiao.sap.view;
2
3 import javafx.fxml.FXMLLoader;
4 import javafx.scene.Parent;
5
6 import java.io.IOException;
7 import java.net.URL;
8 import java.util.ResourceBundle;
9
10 public class ViewManager {
11
12     private static ViewManager instance;
13
14     private ViewManager() {
15
16     }
17
18     /**
19      * Devuelve la instancia de la clase ViewManager aplicando el patrón Singleton
20     */
21     public static ViewManager getInstance() {
22         if (instance == null) {
23             instance = new ViewManager();
24         }
25         return instance;
26     }
27
28     public Parent getView(String viewName) {
29         try {
30             FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource(viewName
+ ".fxml"));
31             return fxmlLoader.load();
32         } catch (IOException e) {
33             e.printStackTrace();
34             return null;
35         }
36     }
37 }
```

```
1 package com.msebastiao.sap.model;
2
3 import java.time.LocalDate;
4 import java.time.LocalTime;
5
6 public class Turno {
7     private int id;
8     private LocalDate fecha;
9     private LocalTime horaInicio;
10    private LocalTime horaFin;
11    private String estado;
12    private TitularVehiculo titularVehiculo;
13
14    public Turno(LocalDate fecha, LocalTime horaInicio, LocalTime horaFin, String
15        estado, TitularVehiculo titularVehiculo) {
16        this.fecha = fecha;
17        this.horaInicio = horaInicio;
18        this.horaFin = horaFin;
19        this.estado = estado;
20        this.titularVehiculo = titularVehiculo;
21    }
22
23    public Turno(int id, LocalDate fecha, LocalTime horaInicio, LocalTime horaFin,
24        String estado, TitularVehiculo titularVehiculo) {
25        this.id = id;
26        this.fecha = fecha;
27        this.horaInicio = horaInicio;
28        this.horaFin = horaFin;
29        this.estado = estado;
30        this.titularVehiculo = titularVehiculo;
31    }
32
33    public int getId() {
34        return id;
35    }
36
37    public void setId(int id) {
38        this.id = id;
39    }
40
41    public LocalDate getFecha() {
42        return fecha;
43    }
44
45    public void setFecha(LocalDate fecha) {
46        this.fecha = fecha;
47    }
48
49    public LocalTime getHoraInicio() {
50        return horaInicio;
51    }
52
53    public void setHoraInicio(LocalTime horaInicio) {
54        this.horaInicio = horaInicio;
55    }
56
57    public LocalTime getHoraFin() {
58        return horaFin;
59    }
60
61    public void setHoraFin(LocalTime horaFin) {
62        this.horaFin = horaFin;
63    }
64
65    public String getEstado() {
66        return estado;
67    }
68
69    public void setEstado(String estado) {
70        this.estado = estado;
71    }
72}
```

```
69     }
70
71     public TitularVehiculo getTitularVehiculo() {
72         return titularVehiculo;
73     }
74
75     public void setTitularVehiculo(TitularVehiculo titularVehiculo) {
76         this.titularVehiculo = titularVehiculo;
77     }
78
79 }
80
```

```
1 package com.msebastiao.sap.model;
2
3 import java.time.LocalDate;
4 import java.util.List;
5
6 public class Agenda {
7     private int id;
8     private LocalDate fecha;
9     private List<Turno> turnos;
10
11    public Agenda(int id, LocalDate fecha, List<Turno> turnos) {
12        this.id = id;
13        this.fecha = fecha;
14        this.turnos = turnos;
15    }
16
17    public int getId() {
18        return id;
19    }
20
21    public void setId(int id) {
22        this.id = id;
23    }
24
25    public LocalDate getFecha() {
26        return fecha;
27    }
28
29    public void setFecha(LocalDate fecha) {
30        this.fecha = fecha;
31    }
32
33    public List<Turno> getTurnos() {
34        return turnos;
35    }
36
37    public void setTurnos(List<Turno> turnos) {
38        this.turnos = turnos;
39    }
40
41    public void agregarTurno(Turno turno) {
42        turnos.add(turno);
43    }
44
45    public void eliminarTurno(Turno turno) {
46        turnos.remove(turno);
47    }
48 }
49
```

```
1 package com.msebastiao.sap.model;
2
3 import java.time.LocalDate;
4
5 public class Informe {
6     private int id;
7     private LocalDate fecha;
8     private String contenido;
9
10    public Informe(int id, LocalDate fecha, String contenido) {
11        this.id = id;
12        this.fecha = fecha;
13        this.contenido = contenido;
14    }
15
16    public Informe(LocalDate fecha, String contenido) {
17        this.fecha = fecha;
18        this.contenido = contenido;
19    }
20
21    public int getId() {
22        return id;
23    }
24
25    public void setId(int id) {
26        this.id = id;
27    }
28
29    public LocalDate getFecha() {
30        return fecha;
31    }
32
33    public void setFecha(LocalDate fecha) {
34        this.fecha = fecha;
35    }
36
37    public String getContenido() {
38        return contenido;
39    }
40
41    public void setContenido(String contenido) {
42        this.contenido = contenido;
43    }
44 }
45 }
```

```
1 package com.msebastiao.sap.model;
2
3 public class Mecanico {
4     private int id;
5     private String nombre;
6     private String apellido;
7
8     public Mecanico(int id, String nombre, String apellido) {
9         this.id = id;
10        this.nombre = nombre;
11        this.apellido = apellido;
12    }
13
14    public Mecanico(String nombre, String apellido) {
15        this.nombre = nombre;
16        this.apellido = apellido;
17    }
18
19    public int getId() {
20        return id;
21    }
22
23    public void setId(int id) {
24        this.id = id;
25    }
26
27    public String getNombre() {
28        return nombre;
29    }
30
31    public void setNombre(String nombre) {
32        this.nombre = nombre;
33    }
34
35    public String getApellido() {
36        return apellido;
37    }
38
39    public void setApellido(String apellido) {
40        this.apellido = apellido;
41    }
42 }
```

```
1 package com.msebastiao.sap.model;
2
3 public class Vehiculo {
4     private int id;
5     private String marca;
6     private String modelo;
7     private int anio;
8     private TitularVehiculo titularVehiculo;
9
10    public Vehiculo(int id, String marca, String modelo, int anio, TitularVehiculo
titularVehiculo) {
11        this.id = id;
12        this.marca = marca;
13        this.modelo = modelo;
14        this.anio = anio;
15        this.titularVehiculo = titularVehiculo;
16    }
17
18    public Vehiculo(String marca, String modelo, int anio, TitularVehiculo
titularVehiculo) {
19        this.marca = marca;
20        this.modelo = modelo;
21        this.anio = anio;
22        this.titularVehiculo = titularVehiculo;
23    }
24
25    public int getId() {
26        return id;
27    }
28
29    public void setId(int id) {
30        this.id = id;
31    }
32
33    public String getMarca() {
34        return marca;
35    }
36
37    public void setMarca(String marca) {
38        this.marca = marca;
39    }
40
41    public String getModelo() {
42        return modelo;
43    }
44
45    public void setModelo(String modelo) {
46        this.modelo = modelo;
47    }
48
49    public int getAnio() {
50        return anio;
51    }
52
53    public void setAnio(int anio) {
54        this.anio = anio;
55    }
56
57    public TitularVehiculo getTitularVehiculo() {
58        return titularVehiculo;
59    }
60
61    public void setTitularVehiculo(TitularVehiculo titularVehiculo) {
62        this.titularVehiculo = titularVehiculo;
63    }
64 }
65
```

```
1 package com.msebastiao.sap.model;
2
3 public class Actividad {
4     private int id;
5     private String descripcion;
6     private int tiempoEmpleado;
7     private String estado;
8     private int fichaMecanicaId;
9
10    public Actividad(int id, String descripcion, int tiempoEmpleado, String estado
11 , int fichaMecanicaId) {
12         this.id = id;
13         this.descripcion = descripcion;
14         this.tiempoEmpleado = tiempoEmpleado;
15         this.estado = estado;
16         this.fichaMecanicaId = fichaMecanicaId;
17     }
18
19    public Actividad(String descripcion, int tiempoEmpleado, String estado, int
20 fichaMecanicaId) {
21         this.descripcion = descripcion;
22         this.tiempoEmpleado = tiempoEmpleado;
23         this.estado = estado;
24         this.fichaMecanicaId = fichaMecanicaId;
25     }
26
27    public int getId() {
28         return id;
29     }
30
31    public void setId(int id) {
32         this.id = id;
33     }
34
35    public String getDescripcion() {
36         return descripcion;
37     }
38
39    public void setDescripcion(String descripcion) {
40         this.descripcion = descripcion;
41     }
42
43    public int getTiempoEmpleado() {
44         return tiempoEmpleado;
45     }
46
47    public void setTiempoEmpleado(int tiempoEmpleado) {
48         this.tiempoEmpleado = tiempoEmpleado;
49     }
50
51    public String getEstado() {
52         return estado;
53     }
54
55    public void setEstado(String estado) {
56         this.estado = estado;
57     }
58
59    public int getFichaMecanicaId() {
60         return fichaMecanicaId;
61     }
62
63    public void setFichaMecanicaId(int fichaMecanicaId) {
64         this.fichaMecanicaId = fichaMecanicaId;
65     }
66 }
```

```
1 package com.msebastiao.sap.model;
2
3 public class Repuestos {
4     private int id;
5     private String nombre;
6     private int cantidad;
7     private int fichaMecanicaId;
8
9     public Repuestos(int id, String nombre, int cantidad, int fichaMecanicaId) {
10         this.id = id;
11         this.nombre = nombre;
12         this.cantidad = cantidad;
13         this.fichaMecanicaId = fichaMecanicaId;
14     }
15
16     public Repuestos(String nombre, int cantidad, int fichaMecanicaId) {
17         this.nombre = nombre;
18         this.cantidad = cantidad;
19         this.fichaMecanicaId = fichaMecanicaId;
20     }
21
22     public int getId() {
23         return id;
24     }
25
26     public void setId(int id) {
27         this.id = id;
28     }
29
30     public String getNombre() {
31         return nombre;
32     }
33
34     public void setNombre(String nombre) {
35         this.nombre = nombre;
36     }
37
38     public int getCantidad() {
39         return cantidad;
40     }
41
42     public void setCantidad(int cantidad) {
43         this.cantidad = cantidad;
44     }
45
46     public int getFichaMecanicaId() {
47         return fichaMecanicaId;
48     }
49
50     public void setFichaMecanicaId(int fichaMecanicaId) {
51         this.fichaMecanicaId = fichaMecanicaId;
52     }
53 }
```

```
1 package com.msebastiao.sap.model;
2
3 import java.time.LocalDate;
4 import java.util.List;
5
6 public class FichaMecanica {
7     private int id;
8     private Mecanico mecanico;
9     private TitularVehiculo titularVehiculo;
10    private Vehiculo vehiculo;
11    private LocalDate fechaInicio;
12    private LocalDate fechaFin;
13    private String estado;
14    private List<Actividad> actividadesRealizadas;
15    private List<Repuestos> repuestosEmpleados;
16
17    public FichaMecanica(int id, Mecanico mecanico, TitularVehiculo titularVehiculo
18 , Vehiculo vehiculo,
19                         LocalDate fechaInicio, LocalDate fechaFin, String estado,
20                         List<Actividad> actividadesRealizadas, List<Repuestos>
21                         repuestosEmpleados) {
22         this.id = id;
23         this.mecanico = mecanico;
24         this.titularVehiculo = titularVehiculo;
25         this.vehiculo = vehiculo;
26         this.fechaInicio = fechaInicio;
27         this.fechaFin = fechaFin;
28         this.estado = estado;
29         this.actividadesRealizadas = actividadesRealizadas;
30         this.repuestosEmpleados = repuestosEmpleados;
31     }
32
33     public int getId() {
34         return id;
35     }
36
37     public void setId(int id) {
38         this.id = id;
39     }
40
41     public Mecanico getMecanico() {
42         return mecanico;
43     }
44
45     public void setMecanico(Mecanico mecanico) {
46         this.mecanico = mecanico;
47     }
48
49     public TitularVehiculo getTitularVehiculo() {
50         return titularVehiculo;
51     }
52
53     public void setTitularVehiculo(TitularVehiculo titularVehiculo) {
54         this.titularVehiculo = titularVehiculo;
55     }
56
57     public Vehiculo getVehiculo() {
58         return vehiculo;
59     }
60
61     public void setVehiculo(Vehiculo vehiculo) {
62         this.vehiculo = vehiculo;
63     }
64
65     public LocalDate getFechaInicio() {
66         return fechaInicio;
67     }
68
69     public void setFechaInicio(LocalDate fechaInicio) {
70         this.fechaInicio = fechaInicio;
71     }
72
73     public List<Actividad> getActividadesRealizadas() {
74         return actividadesRealizadas;
75     }
76
77     public void setActividadesRealizadas(List<Actividad> actividadesRealizadas) {
78         this.actividadesRealizadas = actividadesRealizadas;
79     }
80
81     public List<Repuestos> getRepuestosEmpleados() {
82         return repuestosEmpleados;
83     }
84
85     public void setRepuestosEmpleados(List<Repuestos> repuestosEmpleados) {
86         this.repuestosEmpleados = repuestosEmpleados;
87     }
88 }
```

```
69     }
70
71     public LocalDate getFechaFin() {
72         return fechaFin;
73     }
74
75     public void setFechaFin(LocalDate fechaFin) {
76         this.fechaFin = fechaFin;
77     }
78
79     public String getEstado() {
80         return estado;
81     }
82
83     public void setEstado(String estado) {
84         this.estado = estado;
85     }
86
87     public List<Actividad> getActividadesRealizadas() {
88         return actividadesRealizadas;
89     }
90
91     public void setActividadesRealizadas(List<Actividad> actividadesRealizadas) {
92         this.actividadesRealizadas = actividadesRealizadas;
93     }
94
95     public List<Repuestos> getRepuestosEmpleados() {
96         return repuestosEmpleados;
97     }
98
99     public void setRepuestosEmpleados(List<Repuestos> repuestosEmpleados) {
100        this.repuestosEmpleados = repuestosEmpleados;
101    }
102 }
```

```
1 package com.msebastiao.sap.model;
2
3 import java.util.ArrayList;
4 import java.util.List;
5
6 public class TitularVehiculo {
7     private int id;
8     private String nombre;
9     private String apellido;
10    private String dni;
11    private List<Vehiculo> vehiculos;
12
13
14    public TitularVehiculo(int id, String nombre, String apellido, String dni) {
15        this.id = id;
16        this.nombre = nombre;
17        this.apellido = apellido;
18        this.dni = dni;
19        this.vehiculos = new ArrayList<>();
20    }
21
22    public TitularVehiculo(int id, String nombre, String apellido, String dni, List
23 <Vehiculo> vehiculos) {
24        this.id = id;
25        this.nombre = nombre;
26        this.apellido = apellido;
27        this.dni = dni;
28        this.vehiculos = vehiculos;
29    }
30
31    public int getId() {
32        return id;
33    }
34
35    public void setId(int id) {
36        this.id = id;
37    }
38
39    // Getters y Setters
40
41    public String getNombre() {
42        return nombre;
43    }
44
45    public void setNombre(String nombre) {
46        this.nombre = nombre;
47    }
48
49    public String getApellido() {
50        return apellido;
51    }
52
53    public void setApellido(String apellido) {
54        this.apellido = apellido;
55    }
56
57    public String getDni() {
58        return dni;
59    }
60
61    public void setDni(String dni) {
62        this.dni = dni;
63    }
64
65    public List<Vehiculo> getVehiculos() {
66        return vehiculos;
67    }
68
69    public void setVehiculos(List<Vehiculo> vehiculos) {
70        this.vehiculos = vehiculos;
71    }
72
73}
```

```
70      }
71  }
72
```

```
1 package com.msebastiao.sap.model;
2
3 public class EncargadoRepcion {
4     private String nombre;
5     private String apellido;
6
7     public EncargadoRepcion(String nombre, String apellido) {
8         this.nombre = nombre;
9         this.apellido = apellido;
10    }
11
12    public void consultarAgenda() {
13        // Implementar lógica de consulta de agenda
14    }
15
16    public void registrarAsistencia() {
17        // Implementar lógica de registro de asistencia
18    }
19
20    // Getters y Setters
21
22    public String getNombre() {
23        return nombre;
24    }
25
26    public void setNombre(String nombre) {
27        this.nombre = nombre;
28    }
29
30    public String getApellido() {
31        return apellido;
32    }
33
34    public void setApellido(String apellido) {
35        this.apellido = apellido;
36    }
37 }
38
```

```
1 package com.msebastiao.sap.model;
2
3 public class GerenteOperaciones {
4     private String nombre;
5     private String apellido;
6
7     public GerenteOperaciones(String nombre, String apellido) {
8         this.nombre = nombre;
9         this.apellido = apellido;
10    }
11
12    public void controlarTrabajo() {
13        // Implementar lógica de control de trabajo
14    }
15
16    // Getters y Setters
17
18    public String getNombre() {
19        return nombre;
20    }
21
22    public void setNombre(String nombre) {
23        this.nombre = nombre;
24    }
25
26    public String getApellido() {
27        return apellido;
28    }
29
30    public void setApellido(String apellido) {
31        this.apellido = apellido;
32    }
33 }
34
```

```
1 package com.msebastiao.sap.model;
2
3 public class EmpleadoAdministracion {
4     private String nombre;
5     private String apellido;
6
7     public EmpleadoAdministracion(String nombre, String apellido) {
8         this.nombre = nombre;
9         this.apellido = apellido;
10    }
11
12    public void generarInformeMensual() {
13        // Implementar lógica de generación de informe mensual
14    }
15
16    // Getters y Setters
17
18    public String getNombre() {
19        return nombre;
20    }
21
22    public void setNombre(String nombre) {
23        this.nombre = nombre;
24    }
25
26    public String getApellido() {
27        return apellido;
28    }
29
30    public void setApellido(String apellido) {
31        this.apellido = apellido;
32    }
33 }
34
```

```
1 package com.msebastiao.sap.commands;
2
3 import javafx.scene.control.Tab;
4 import javafx.scene.control.TabPane;
5 import javafx.scene.layout.Pane;
6
7 /**
8  * El patrón Command convierte las solicitudes en objetos, permitiendo parametrizar
9  * los métodos con diferentes
10 * solicitudes, retrasar o poner en cola la ejecución de una solicitud y soportar
11 * operaciones que se pueden deshacer.
12 * En este caso se usa para abrir una nueva vista en la interfaz de usuario con el
13 * nombre de la vista dada.
14 */
15 public class OpenViewCommand {
16
17     private final TabPane tabPane;
18     private final String title;
19     private final Pane view;
20
21     public OpenViewCommand(TabPane tabPane, String title, Pane view) {
22         this.tabPane = tabPane;
23         this.title = title;
24         this.view = view;
25     }
26
27     public void execute() {
28         for (Tab tab : tabPane.getTabs()) {
29             if (tab.getText().equals(title)) {
30                 tabPane.getSelectionModel().select(tab);
31                 return;
32             }
33         }
34         Tab tab = new Tab(title);
35         tab.setContent(view);
36         tabPane.getTabs().add(tab);
37         tabPane.getSelectionModel().select(tab);
38     }
39 }
```

```

1 package com.msebastiao.sap.database;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6
7 /**
8  * Clase que gestiona la conexión a la base de datos utilizando el patrón Singleton
9  *
10 * Proporciona un único punto de acceso a la conexión y asegura que solo se cree
11 * una instancia de la conexión.
12 */
13 public class DatabaseConnection {
14     /**
15      * Instancia única de la clase {@link DatabaseConnection}.
16      */
17     private static DatabaseConnection instance;
18     /**
19      * Conexión a la base de datos.
20      */
21     private Connection connection;
22     /**
23      * Constructor privado para prevenir la creación de instancias desde fuera de
24      * la clase.
25      * Establece la conexión a la base de datos al crear la instancia.
26      */
27     private DatabaseConnection() throws SQLException {
28         createConnection();
29     }
30
31     /**
32      * Crea una nueva conexión a la base de datos.
33      *
34      * @throws SQLException si ocurre un error al establecer la conexión.
35      */
36     private void createConnection() throws SQLException {
37         String url = "jdbc:mysql://localhost:3306/supercharger";
38         String usuario = "root";
39         String contrasena = "P4ssw0rd!";
40         connection = DriverManager.getConnection(url, usuario, contrasena);
41     }
42
43     /**
44      * Obtiene la instancia única de la clase {@link DatabaseConnection}.
45      * Si la instancia no existe o la conexión está cerrada, se crea una nueva
46      * instancia.
47      *
48      * @return La instancia única de {@link DatabaseConnection}.
49      * @throws SQLException Si ocurre un error al crear la conexión.
50      */
51     public static DatabaseConnection getInstance() throws SQLException {
52         if (instance == null || instance.connection.isClosed()) {
53             instance = new DatabaseConnection();
54         }
55         return instance;
56     }
57
58     /**
59      * Obtiene la conexión a la base de datos.
60      * Si la conexión no existe o está cerrada, se crea una nueva conexión.
61      *
62      * @return La conexión a la base de datos.
63      * @throws SQLException Si ocurre un error al crear la conexión.
64      */
65     public Connection getConnection() throws SQLException {
66         if (connection == null || connection.isClosed()) {
67             createConnection();
68         }
69         return connection;
70     }
71 }

```

```
67      }
68      return connection;
69  }
70 }
```

```
1 package com.msebastiao.sap.controller;  
2  
3 public class AppController {  
4 }  
5
```

```

1 package com.msebastiao.sap.controller;
2
3 import com.msebastiao.sap.dao.TitularVehiculoDAO;
4 import com.msebastiao.sap.dao.TurnoDAO;
5 import com.msebastiao.sap.model.TitularVehiculo;
6 import com.msebastiao.sap.model.Turno;
7 import javafx.fxml.FXML;
8 import javafx.scene.control.Alert;
9 import javafx.scene.control.TableColumn;
10 import javafx.scene.control.TableView;
11 import javafx.scene.control.TextField;
12 import javafx.scene.control.cell.PropertyValueFactory;
13
14 import java.sql.SQLException;
15 import java.time.LocalDate;
16 import java.time.LocalTime;
17 import java.util.Collections;
18 import java.util.List;
19
20 public class SolicitarTurnoController {
21
22     @FXML
23     private TableView<Turno> turnosTable;
24     @FXML
25     private TableColumn<Turno, LocalDate> fechaColumn;
26     @FXML
27     private TableColumn<Turno, LocalTime> horaInicioColumn;
28     @FXML
29     private TableColumn<Turno, LocalTime> horaFinColumn;
30     @FXML
31     private TableColumn<Turno, String> estadoColumn;
32     @FXML
33     private TextField dniField;
34
35     @FXML
36     public void initialize() {
37         fechaColumn.setCellValueFactory(new PropertyValueFactory<>("fecha"));
38         horaInicioColumn.setCellValueFactory(new PropertyValueFactory<>("horaInicio"));
39         horaFinColumn.setCellValueFactory(new PropertyValueFactory<>("horaFin"));
40         estadoColumn.setCellValueFactory(new PropertyValueFactory<>("estado"));
41
42         cargarTurnos();
43     }
44
45     private void cargarTurnos() {
46
47         List<Turno> turnos = Collections.emptyList();
48         try {
49             TurnoDAO turnoDAO = new TurnoDAO();
50             turnos = turnoDAO.getAll();
51         } catch (SQLException e) {
52             Alert alert = new Alert(Alert.AlertType.WARNING);
53             alert.setTitle("Turnos");
54             alert.setHeaderText(null);
55             alert.setContentText("No se encontraron turnos disponibles");
56             alert.showAndWait();
57         }
58         turnosTable.getItems().setAll(turnos);
59     }
60
61
62     @FXML
63     private void handleSolicitarTurno() {
64         try {
65             Turno selectedTurno = turnosTable.getSelectionModel().getSelectedItem();
66             if (selectedTurno == null) {
67                 Alert alert = new Alert(Alert.AlertType.WARNING);
68                 alert.setTitle("Error");
69             }
70         } catch (Exception e) {
71             Alert alert = new Alert(Alert.AlertType.ERROR);
72             alert.setTitle("Error");
73             alert.setHeaderText("Ha ocurrido un error al solicitar el turno.");
74             alert.setContentText(e.getMessage());
75             alert.showAndWait();
76         }
77     }
78
79 }

```

```
69             alert.setHeaderText(null);
70             alert.setContentText("Por favor, seleccione un turno disponible.");
71         }
72         alert.showAndWait();
73     }
74     String dni = dniField.getText();
75     TitularVehiculoDAO titularVehiculoDAO = new TitularVehiculoDAO();
76     TitularVehiculo titular = titularVehiculoDAO.getByDni(dni);
77
78     if (titular == null) {
79         Alert alert = new Alert(Alert.AlertType.WARNING);
80         alert.setTitle("Error");
81         alert.setHeaderText(null);
82         alert.setContentText("El Titular no existe o no fue encontrado con el DNI ingresado");
83         alert.showAndWait();
84     }
85
86     selectedTurno.setEstado("Solicitado");
87     selectedTurno.setTitularVehiculo(titular);
88     TurnoDAO turnoDAO = new TurnoDAO();
89     turnoDAO.update(selectedTurno);
90
91     cargarTurnos();
92
93     Alert alert = new Alert(Alert.AlertType.INFORMATION);
94     alert.setTitle("Turno Solicitado");
95     alert.setHeaderText(null);
96     alert.setContentText("El turno ha sido solicitado exitosamente.");
97     alert.showAndWait();
98
99 } catch (SQLException e) {
100     Alert alert = new Alert(Alert.AlertType.ERROR);
101     alert.setTitle("Turno");
102     alert.setHeaderText(null);
103     alert.setContentText("No se pudo solicitar el turno por un error inesperado.");
104     alert.showAndWait();
105     e.printStackTrace();
106 }
107 }
108 }
109 }
110 }
```

```

1 package com.msebastiao.sap.controller;
2
3 import com.msebastiao.sap.dao.TurnoDAO;
4 import com.msebastiao.sap.model.Turno;
5 import javafx.collections.FXCollections;
6 import javafx.fxml.FXML;
7 import javafx.scene.control.Alert;
8 import javafx.scene.control.TableColumn;
9 import javafx.scene.control TableView;
10 import javafx.scene.control.TextField;
11 import javafx.scene.control.cell.PropertyValueFactory;
12
13 import java.sql.SQLException;
14 import java.util.List;
15
16 public class ConsultarTurnosController {
17
18     @FXML
19     private TextField dniField;
20     @FXML
21     private TableView<Turno> tablaTurnos;
22     @FXML
23     private TableColumn<Turno, String> fechaColumn;
24     @FXML
25     private TableColumn<Turno, String> horaInicioColumn;
26     @FXML
27     private TableColumn<Turno, String> horaFinColumn;
28     @FXML
29     private TableColumn<Turno, String> estadoColumn;
30
31     private TurnoDAO turnoDAO;
32
33     @FXML
34     private void initialize() throws SQLException {
35         turnoDAO = new TurnoDAO();
36         // Configurar las columnas de la tabla
37         fechaColumn.setCellValueFactory(new PropertyValueFactory<>("fecha"));
38         horaInicioColumn.setCellValueFactory(new PropertyValueFactory<>("horaInicio"));
39         horaFinColumn.setCellValueFactory(new PropertyValueFactory<>("horaFin"));
40         estadoColumn.setCellValueFactory(new PropertyValueFactory<>("estado"));
41     }
42
43     @FXML
44     private void buscarTurnos() {
45         String dni = dniField.getText();
46
47         if (dni.isEmpty()) {
48             mostrarAlerta("Error", "Por favor, ingrese un DNI.");
49             return;
50         }
51
52         try {
53             List<Turno> turnos = turnoDAO.getAll().stream().filter(t -> t.
54             getTitularVehiculo() != null
55             && dni.equals(t.getTitularVehiculo().getDni())).toList();
56
57             if (turnos.isEmpty()) {
58                 mostrarAlerta("No se encontraron turnos", "No se encontraron turnos
59 para el DNI ingresado.");
60             } else {
61                 tablaTurnos.setItems(FXCollections.observableArrayList(turnos));
62             }
63         } catch (Exception e) {
64             mostrarAlerta("Error de base de datos", "Ocurrió un error al consultar
65 la base de datos.");
66             e.printStackTrace();
67         }
68     }
69 }

```

```
67     private void mostrarAlerta(String titulo, String mensaje) {  
68         Alert alert = new Alert(Alert.AlertType.ERROR);  
69         alert.setTitle(titulo);  
70         alert.setHeaderText(null);  
71         alert.setContentText(mensaje);  
72         alert.showAndWait();  
73     }  
74 }  
75
```

```
1 package com.msebastiao.sap.controller;  
2  
3 public class InformarClienteController {  
4 }  
5
```

```
1 package com.msebastiao.sap.controller;
2
3 import javafx.fxml.FXML;
4 import javafx.scene.control.Alert;
5 import javafx.scene.control.ListView;
6
7 public class SupervisarTareasController {
8
9     @FXML
10    private ListView<String> tareasList;
11
12    @FXML
13    private void initialize() {
14        tareasList.getItems().addAll(
15            "Cambio de aceite - Juan Pérez",
16            "Revisión de frenos - Ana López",
17            "Alineación - Mario Casas"
18        );
19    }
20
21    @FXML
22    private void handleActualizarLista() {
23        // Lógica para actualizar la lista de tareas
24        Alert alert = new Alert(Alert.AlertType.INFORMATION);
25        alert.setTitle("Lista Actualizada");
26        alert.setHeaderText(null);
27        alert.setContentText("La lista de tareas ha sido actualizada.");
28        alert.showAndWait();
29    }
30
31    @FXML
32    private void handleGenerarReporte() {
33        // Lógica para generar el reporte
34        Alert alert = new Alert(Alert.AlertType.INFORMATION);
35        alert.setTitle("Reporte Generado");
36        alert.setHeaderText(null);
37        alert.setContentText("El reporte ha sido generado exitosamente.");
38        alert.showAndWait();
39    }
40 }
```

```
1 package com.msebastiao.sap.controller;
2
3 import javafx.fxml.FXML;
4 import javafx.scene.control.Alert;
5 import javafx.scene.control.TextField;
6
7 public class RegistrarAsistenciaController {
8
9     @FXML
10    private TextField searchField;
11
12    @FXML
13    private TextField nameField;
14
15    @FXML
16    private TextField dateField;
17
18    @FXML
19    private void handleSearch() {
20        // Aquí deberías implementar la lógica para buscar el turno en la base de
21        // datos
22        nameField.setText("Matias Sebastiao"); // Simulación de respuesta
23        dateField.setText("06/03/2025 10:00 AM"); // Simulación de respuesta
24    }
25
26    @FXML
27    private void handleConfirm() {
28        // Aquí deberías implementar la lógica para confirmar la asistencia en la
29        // base de datos
30        Alert alert = new Alert(Alert.AlertType.INFORMATION);
31        alert.setTitle("Confirmación");
32        alert.setHeaderText(null);
33        alert.setContentText("Asistencia confirmada!");
34    }
35}
```

```
1 package com.msebastiao.sap.controller;
2
3 import javafx.fxml.FXML;
4 import javafx.scene.control.Alert;
5 import javafx.scene.control.ComboBox;
6
7 public class GenerarInformeMensualController {
8
9     @FXML
10    private ComboBox<String> mesComboBox;
11
12    @FXML
13    private ComboBox<String> tipoComboBox;
14
15    @FXML
16    private void initialize() {
17        mesComboBox.setValue("Enero");
18        tipoComboBox.setValue("Servicios realizados");
19    }
20
21    @FXML
22    private void handleGenerarInforme() {
23        // Lógica para generar el informe
24        Alert alert = new Alert(Alert.AlertType.INFORMATION);
25        alert.setTitle("Informe Generado");
26        alert.setHeaderText(null);
27        alert.setContentText("El informe para el mes de " + mesComboBox.getValue()
28            () + " ha sido generado exitosamente.");
29        alert.showAndWait();
30    }
31 }
```

```
1 package com.msebastiao.sap.controller;
2
3 import javafx.fxml.FXML;
4 import javafx.scene.control.Alert;
5 import javafx.scene.control.TextArea;
6 import javafx.scene.control.TextField;
7
8 public class RealizarMantenimientoController {
9
10    @FXML
11    private TextField fichaField;
12
13    @FXML
14    private TextField clienteField;
15
16    @FXML
17    private TextField vehiculoField;
18
19    @FXML
20    private TextArea actividadesArea;
21
22    @FXML
23    private TextArea repuestosArea;
24
25    @FXML
26    private void handleFinalizarMantenimiento() {
27        // Lógica para guardar los datos del mantenimiento
28        Alert alert = new Alert(Alert.AlertType.INFORMATION);
29        alert.setTitle("Mantenimiento Registrado");
30        alert.setHeaderText(null);
31        alert.setContentText("El mantenimiento ha sido registrado exitosamente.");
32        alert.showAndWait();
33    }
34 }
35
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.geometry.Insets?>
4 <?import javafx.scene.control.*?>
5 <?import javafx.scene.layout.*?>
6 <GridPane xmlns="http://javafx.com/javafx"
7         xmlns:fx="http://javafx.com/fxml"
8         fx:controller="com.msebastiao.sap.controller.SolicitarTurnoController"
9         prefHeight="400.0" prefWidth="600.0"
10        hgap="10"
11        vgap="8">
12    <padding>
13        <Insets top="10" right="10" bottom="10" left="10"/>
14    </padding>
15
16    <Label text="Seleccione un turno disponible:" GridPane.rowIndex="0" GridPane.
17 columnIndex="0"/>
17    <TableView fx:id="turnosTable" GridPane.rowIndex="1" GridPane.columnIndex="0"
18 GridPane.columnSpan="2"
18        VBox.vgrow="ALWAYS">
19        <columns>
20            <TableColumn fx:id="fechaColumn" text="Fecha" prefWidth="150"/>
21            <TableColumn fx:id="horaInicioColumn" text="Hora Inicio" prefWidth="150
22 "/>
22            <TableColumn fx:id="horaFinColumn" text="Hora Fin" prefWidth="150"/>
23            <TableColumn fx:id="estadoColumn" text="Estado" prefWidth="150"/>
24        </columns>
25    </TableView>
26
27    <Label text="DNI:" GridPane.rowIndex="2" GridPane.columnIndex="0"/>
28    <TextField fx:id="dniField" GridPane.rowIndex="2" GridPane.columnIndex="1"/>
29
30    <Button text="Solicitar Turno" onAction="#handleSolicitarTurno" GridPane.
30        rowIndex="3" GridPane.columnIndex="1"/>
31 </GridPane>
32
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.geometry.*?>
4 <?import javafx.scene.control.*?>
5 <?import javafx.scene.layout.*?>
6
7 <GridPane hgap="10" prefHeight="400.0" prefWidth="600.0" vgap="8" xmlns="http://
javafx.com/javafx/17.0.2-ea"
     xmlns:fx="http://javafx.com/fxml/1" fx:controller="com.msebastiao.sap.
controller.ConsultarTurnosController">
8
9     <padding>
10        <Insets bottom="10" left="10" right="10" top="10"/>
11    </padding>
12
13
14     <Label text="DNI:" GridPane.columnIndex="0" GridPane.rowIndex="0"/>
15     <TextField fx:id="dniField" GridPane.columnIndex="0" GridPane.rowIndex="1"/>
16     <Button onAction="#buscarTurnos" text="Buscar" GridPane.columnIndex="1"
GridPane.rowIndex="1"/>
17
18     <TableView fx:id="tablaTurnos" GridPane.columnIndex="0" GridPane.columnSpan="2"
GridPane.rowIndex="3"
19         VBox.vgrow="ALWAYS">
20         <columns>
21             <TableColumn fx:id="fechaColumn" text="Fecha"/>
22             <TableColumn fx:id="horaInicioColumn" text="Hora Inicio"/>
23             <TableColumn fx:id="horaFinColumn" text="Hora Fin"/>
24             <TableColumn fx:id="estadoColumn" prefWidth="81.0" text="Estado"/>
25         </columns>
26     </TableView>
27 </GridPane>
28
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.geometry.Insets?>
4 <?import javafx.scene.control.*?>
5 <?import javafx.scene.layout.*?>
6 <GridPane xmlns="http://javafx.com/javafx"
7           xmlns:fx="http://javafx.com/fxml"
8           fx:controller="com.msebastiao.sap.controller.InformarClienteController"
9           prefHeight="400.0" prefWidth="600.0"
10          hgap="10"
11          vgap="8">
12    <padding>
13      <Insets top="10" right="10" bottom="10" left="10"/>
14    </padding>
15
16    <Label text="ID de la Ficha Mecánica:" GridPane.rowIndex="0" GridPane.
17 columnIndex="0"/>
17    <TextField fx:id="fichaIdField" GridPane.rowIndex="0" GridPane.columnIndex="1"
18 />
18    <Button text="Buscar Ficha" onAction="#handleBuscarFicha" GridPane.rowIndex="1"
19   GridPane.columnIndex="1"/>
19    <TextArea fx:id="informacionArea" editable="false" GridPane.rowIndex="2"
20   GridPane.columnIndex="0"
20      GridPane.columnSpan="2"/>
21    <Button text="Generar Constancia" onAction="#handleGenerarConstancia" GridPane.
21 rowIndex="3"
22      GridPane.columnIndex="1"/>
23 </GridPane>
24
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.geometry.Insets?>
4 <?import javafx.scene.control.Button?>
5 <?import javafx.scene.control.Label?>
6 <?import javafx.scene.control.ListView?>
7 <?import javafx.scene.layout.*?>
8 <GridPane xmlns="http://javafx.com/javafx"
9      xmlns:fx="http://javafx.com/fxml"
10     fx:controller="com.msebastiao.sap.controller.SupervisarTareasController"
11     prefHeight="400.0" prefWidth="600.0"
12     hgap="10"
13     vgap="8">
14     <padding>
15         <Insets top="10" right="10" bottom="10" left="10"/>
16     </padding>
17
18     <Label text="Supervisar Tareas del Taller" GridPane.rowIndex="0" GridPane.
19       columnIndex="0" GridPane.columnSpan="2"/>
20
21     <ListView fx:id="tareasList" prefHeight="100" GridPane.rowIndex="1" GridPane.
22       columnIndex="0"
23           GridPane.columnSpan="2"/>
24
25     <Button text="Actualizar Lista" onAction="#handleActualizarLista" GridPane.
26       rowIndex="2" GridPane.columnIndex="0"/>
27     <Button text="Generar Reporte" onAction="#handleGenerarReporte" GridPane.
28       rowIndex="2" GridPane.columnIndex="1"/>
29 </GridPane>
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.geometry.Insets?>
4 <?import javafx.scene.control.*?>
5 <?import javafx.scene.layout.*?>
6 <GridPane xmlns="http://javafx.com/javafx"
7         xmlns:fx="http://javafx.com/fxml"
8         fx:controller="com.msebastiao.sap.controller.
  RegistrarAsistenciaController"
9         prefHeight="400.0" prefWidth="600.0"
10        hgap="10"
11        vgap="8">
12    <padding>
13        <Insets top="10" right="10" bottom="10" left="10"/>
14    </padding>
15
16    <Label text="Buscar turno:" GridPane.rowIndex="0" GridPane.columnIndex="0"/>
17    <TextField fx:id="searchField" promptText="Introduzca DNI o apellido" GridPane.
  rowIndex="0"
18          GridPane.columnIndex="1"/>
19    <Button text="Buscar" onAction="#handleSearch" GridPane.rowIndex="0" GridPane.
  columnIndex="2"/>
20
21    <Label text="Nombre:" GridPane.rowIndex="1" GridPane.columnIndex="0"/>
22    <TextField fx:id="nameField" editable="false" GridPane.rowIndex="1" GridPane.
  columnIndex="1"/>
23
24    <Label text="Fecha y hora del turno:" GridPane.rowIndex="2" GridPane.
  columnIndex="0"/>
25    <TextField fx:id="dateField" editable="false" GridPane.rowIndex="2" GridPane.
  columnIndex="1"/>
26
27    <Button text="Confirmar Asistencia" onAction="#handleConfirm" GridPane.rowIndex
  ="3" GridPane.columnIndex="1"/>
28 </GridPane>
29
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.collections.FXCollections?>
4 <?import javafx.geometry.Insets?>
5 <?import javafx.scene.control.*?>
6 <?import javafx.scene.layout.*?>
7 <?import java.lang.*?>
8 <GridPane xmlns="http://javafx.com/javafx"
9         xmlns:fx="http://javafx.com/fxml"
10        fx:controller="com.msebastiao.sap.controller.
11          GenerarInformeMensualController"
12          prefHeight="400.0" prefWidth="600.0"
13          hgap="10"
14          vgap="8">
15    <padding>
16      <Insets top="10" right="10" bottom="10" left="10"/>
17    </padding>
18
19    <Label text="Mes:" GridPane.rowIndex="0" GridPane.columnIndex="0"/>
20    <ComboBox fx:id="mesComboBox" GridPane.rowIndex="0" GridPane.columnIndex="1">
21      <items>
22        <FXCollections fx:factory="observableArrayList">
23          <String fx:value="Enero"/>
24          <String fx:value="Febrero"/>
25          <String fx:value="Marzo"/>
26          <String fx:value="Abril"/>
27          <String fx:value="Mayo"/>
28          <String fx:value="Junio"/>
29          <String fx:value="Julio"/>
30          <String fx:value="Agosto"/>
31          <String fx:value="Septiembre"/>
32          <String fx:value="Octubre"/>
33          <String fx:value="Noviembre"/>
34          <String fx:value="Diciembre"/>
35        </FXCollections>
36      </items>
37    </ComboBox>
38
39    <Label text="Tipo de Informe:" GridPane.rowIndex="1" GridPane.columnIndex="0"/>
40    <ComboBox fx:id="tipoComboBox" GridPane.rowIndex="1" GridPane.columnIndex="1">
41      <items>
42        <FXCollections fx:factory="observableArrayList">
43          <String fx:value="Servicios realizados"/>
44          <String fx:value="Ingresos generados"/>
45          <String fx:value="Repuestos utilizados"/>
46        </FXCollections>
47      </items>
48    </ComboBox>
49
50    <Button text="Generar Informe" onAction="#handleGenerarInforme" GridPane.
51      rowIndex="2" GridPane.columnIndex="1"/>
52  </GridPane>
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.geometry.Insets?>
4 <?import javafx.scene.control.*?>
5 <?import javafx.scene.layout.*?>
6 <GridPane xmlns="http://javafx.com/javafx"
7         xmlns:fx="http://javafx.com/fxml"
8         fx:controller="com.msebastiao.sap.controller.
  RealizarMantenimientoController"
9         prefHeight="400.0" prefWidth="600.0"
10        hgap="10"
11        vgap="8">
12    <padding>
13        <Insets top="10" right="10" bottom="10" left="10"/>
14    </padding>
15
16    <Label text="Ficha N°:" GridPane.rowIndex="0" GridPane.columnIndex="0"/>
17    <TextField fx:id="fichaField" promptText="Ingrese el número de ficha" GridPane.
  rowIndex="0"
18                GridPane.columnIndex="1"/>
19
20    <Label text="Cliente:" GridPane.rowIndex="1" GridPane.columnIndex="0"/>
21    <TextField fx:id="clienteField" editable="false" GridPane.rowIndex="1" GridPane
  .columnIndex="1"/>
22
23    <Label text="Vehículo:" GridPane.rowIndex="2" GridPane.columnIndex="0"/>
24    <TextField fx:id="vehiculoField" editable="false" GridPane.rowIndex="2"
  GridPane.columnIndex="1"/>
25
26    <Label text="Actividades realizadas:" GridPane.rowIndex="3" GridPane.
  columnIndex="0"/>
27    <TextArea fx:id="actividadesArea" prefHeight="100" GridPane.rowIndex="3"
  GridPane.columnIndex="1"/>
28
29    <Label text="Repuestos utilizados:" GridPane.rowIndex="4" GridPane.columnIndex
  ="0"/>
30    <TextArea fx:id="repuestosArea" prefHeight="100" GridPane.rowIndex="4" GridPane
  .columnIndex="1"/>
31
32    <Button text="Finalizar Mantenimiento" onAction="#handleFinalizarMantenimiento"
  GridPane.rowIndex="5"
33                GridPane.columnIndex="1"/>
34 </GridPane>
35
```