1. Conexión a la Base de Datos

La clase `PROYECTOFINAL` establece una conexión con la base de datos MySQL utilizando JDBC. La conexión se realiza a través del siguiente método:

```
public static Connection conectar() {
    Connection conexion = null;
    try {
        conexion = DriverManager.getConnection(URL, USER, PASSWORD);
        System.out.println("CONEXION EXITOSA");
    } catch (SQLException e) {
        System.out.println("ERROR AL CONECTAR: " + e.getMessage());
    }
    return conexion;
}
```

- **URL**: La URL de conexión a la base de datos MySQL.
- **USER**: El nombre de usuario de la base de datos.
- **PASSWORD**: La contraseña de la base de datos.

2. Componentes de la Interfaz Gráfica

El sistema está diseñado utilizando la biblioteca Swing para crear una interfaz gráfica de usuario (GUI). La clase `PROYECTOFINAL` extiende `JFrame` para crear la ventana principal del programa, que contiene campos de texto, botones y un área de texto para mostrar los resultados.

- **Campos de Entrada**:
  - `codigoField`: Para ingresar el código del producto.
  - `nombreField`: Para ingresar el nombre del producto.
  - `precioField`: Para ingresar el precio del producto.
  - `cantidadField`: Para ingresar la cantidad del producto.
  - `fechaField`: Para ingresar la fecha de vencimiento del producto.
- **Botones**:
  - `insertButton`: Inserta un nuevo producto en la base de datos.
  - `listButton`: Muestra una lista de todos los productos almacenados.
  - `searchButton`: Permite buscar un producto por código.
  - `updateButton`: Permite modificar un producto existente.
  - `deleteButton`: Elimina un producto de la base de datos.

- **Área de Texto**: Se utiliza un `JTextArea` para mostrar los resultados de las operaciones realizadas, como la inserción, búsqueda, actualización y eliminación de productos.

3. Operaciones de Base de Datos

a) Insertar Producto

Cuando se presiona el botón "Ingresar Producto", el sistema toma los datos ingresados en los campos de texto y los inserta en la base de datos mediante el siguiente código:

```
public void insertarProducto(String codigo, String nombre, double precio, int cantidad, String fecha) {
   String query = "INSERT INTO producto (codigoProducto, nombreProducto, precioUnitario, cantidadProducto, fechaVencimiento) VALUES (?, ?, ?, ?, ?)";
   try (Connection con = conectar(); PreparedStatement pst = con.prepareStatement(query)) {
      pst.setString(1, codigo);
      pst.setString(2, nombre);
      pst.setDouble(3, precio);
      pst.setInt(4, cantidad);
      pst.setDate(5, java.sql.Date.valueOf(fecha));
      int filasAfectadas = pst.executeUpdate();
      displayArea.setText(filasAfectadas > 0 ? "Producto ingresado" : "No se insertó ningún producto");
   } catch (SQLException e) {
      displayArea.setText("Error al insertar producto: " + e.getMessage());
   }
}
```

Este método utiliza `PreparedStatement` para evitar problemas de inyección SQL y asegura que los valores ingresados sean insertados correctamente.

b) Listar Productos

Cuando se hace clic en "Mostrar Lista de Productos", se ejecuta el siguiente método, que obtiene todos los productos de la base de datos y los muestra en el área de texto:

```
public void listarProductos() {
    String query = "SELECT * FROM producto";
    try (Connection con = conectar(); Statement st = con.createStatement(); ResultSet rs =
st.executeQuery(query)) {
        StringBuilder sb = new StringBuilder();
        while (rs.next()) {
            sb.append("Código: ").append(rs.getString("codigoProducto")).append("\n");
            sb.append("Nombre: ").append(rs.getString("nombreProducto")).append("\n");
            sb.append("Precio: ").append(rs.getDouble("precioUnitario")).append("\n");
            sb.append("Cantidad: ").append(rs.getInt("cantidadProducto")).append("\n");
            sb.append("Fecha de Vencimiento:
").append(rs.getDate("fechaVencimiento")).append("\n\n");
        }
        displayArea.setText(sb.toString().isEmpty() ? "No hay productos disponibles." : sb.toString());
    } catch (SQLException e) {
        displayArea.setText("Error al listar productos: " + e.getMessage());
    }
}
```

c) Buscar Producto

El método `buscarProducto` permite buscar un producto por código. El resultado se muestra en el área de texto, indicando si el producto fue encontrado o no.

```
public void buscarProducto(String codigo) {
   String query = "SELECT * FROM producto WHERE codigoProducto=?";
   try (Connection con = conectar(); PreparedStatement pst = con.prepareStatement(query)) {
      pst.setString(1, codigo);
      ResultSet rs = pst.executeQuery();
      if (rs.next()) {
         displayArea.setText("Código: " + rs.getString("codigoProducto") + "\n" +
                  "Nombre: " + rs.getString("nombreProducto") + "\n" +
                  "Precio: " + rs.getDouble("precioUnitario") + "\n" +
                  "Cantidad: " + rs.getInt("cantidadProducto") + "\n" +
                  "Fecha de Vencimiento: " + rs.getDate("fechaVencimiento"));
      } else {
         displayArea.setText("Producto no encontrado.");
      }
   } catch (SQLException e) {
      displayArea.setText("Error al buscar producto: " + e.getMessage());
   }
}
```

d) Actualizar Producto

Este método permite modificar el nombre y el precio de un producto existente en la base de datos. El usuario debe proporcionar el código del producto a modificar.

```
public void actualizarProducto(String nombre, double precio) {
   String codigo = codigoField.getText();
   String query = "UPDATE producto SET nombreProducto = ?, precioUnitario = ? WHERE
codigoProducto = ?";
```

```
    try (Connection con = conectar(); PreparedStatement pst = con.prepareStatement(query)) {
        pst.setString(1, nombre);
        pst.setDouble(2, precio);
        pst.setString(3, codigo);
        int filasAfectadas = pst.executeUpdate();
        displayArea.setText(filasAfectadas > 0 ? "Producto actualizado" : "No se actualizó el
producto");
    } catch (SQLException e) {
        displayArea.setText("Error al actualizar producto: " + e.getMessage());
    }
}
```

e) *Eliminar Producto*

El método `eliminarProducto` elimina un producto de la base de datos según el código proporcionado.

```
public void eliminarProducto(String codigo) {
    String query = "DELETE FROM producto WHERE codigoProducto = ?";
    try (Connection con = conectar(); PreparedStatement pst = con.prepareStatement(query))
{
        pst.setString(1, codigo);
        int filasAfectadas = pst.executeUpdate();
        displayArea.setText(filasAfectadas > 0 ? "Producto eliminado" : "No se eliminó el
producto");
    } catch (SQLException e) {
        displayArea.setText("Error al eliminar producto: " + e.getMessage());
    }
}
```

4. Funcionalidad de Búsqueda y Filtro en la Lista de Productos

El sistema permite buscar productos mediante un campo de búsqueda que filtra los resultados en tiempo real:

```
public void mostrarListaDeProductos() {

    // Crear la ventana para mostrar la lista de productos

    JFrame listaFrame = new JFrame("Lista de Productos");

    listaFrame.setSize(500, 500);

    listaFrame.setLocationRelativeTo(null);


    JPanel panel = new JPanel();

    panel.setLayout(new BorderLayout());


    JTextField searchField = new JTextField();

    searchField.setPreferredSize(new Dimension(400, 30));


    JTextArea areaDeTexto = new JTextArea();

    areaDeTexto.setEditable(false);

    JScrollPane scrollPane = new JScrollPane(areaDeTexto);


    panel.add(searchField, BorderLayout.NORTH);

    panel.add(scrollPane, BorderLayout.CENTER);


    listaFrame.add(panel);

    listaFrame.setVisible(true);
```

```
    searchField.addKeyListener(new KeyAdapter() {
        @Override
        public void keyReleased(KeyEvent e) {
            String filtro = searchField.getText().trim();
            actualizarListaDeProductos(areaDeTexto, filtro);
        }
    });
}
```

- **Filtro de búsqueda**: El usuario puede escribir en el campo de búsqueda y los productos se filtran automáticamente según el texto ingresado.

5. Limpieza de Campos

Después de cada acción, los campos de entrada se limpian utilizando el método `limpiarCampos`:

```
public void limpiarCampos() {
    codigoField.setText("");
    nombreField.setText("");
    precioField.setText("");
    cantidadField.setText("");
    fechaField.setText("");
}
```