## **Manual Técnico**

IDE Utilizado: IntelliJ IDEA

Versión Java Utilizada: 11

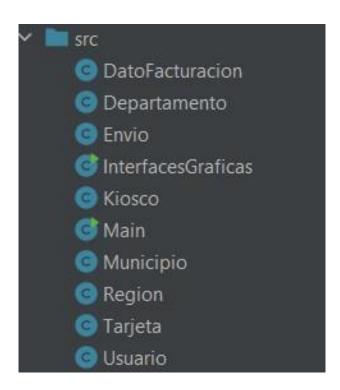
Sistema Operativo: Windows 11

Requerimiento de la aplicación: 20 MB espacio libre y mínimo 2 GB de

ram

A continuación, se mostrarán partes importantes de código y estructura del proyecto para poder comprender como funciona todo.

**Estructura del proyecto:** El proyecto consta de los siguientes archivos, los cuales se desglosarán más adelante.



**Clase Región:** Clase que le da vida a los objetos de tipo región, a continuación, se muestran los atributos de la clase, así como su constructor.

```
public class Region {
    3 usages
    private String codigoRegion;
    3 usages
    private String nombreRegion;
    3 usages
    private double precioNormal;
    3 usages
    private double precioEspecial;
    3 usages
    private int cantidadEnvios;

7 usages
    public Region(String codigoRegion, String nombreRegion, double precioNormal, double precioEspecial, int cantidadEnvios) {
        this.codigoRegion=codigoRegion;
        this.nombreRegion=nombreRegion;
        this.precioNormal=precioNormal;
        this.precioEspecial=precioEspecial;
        this.contidadEnvios=cantidadEnvios;
}
```

Clase Departamento: Clase que le da vida a los objetos de tipo departamento, a continuación, se muestran los atributos de la clase, así como su constructor, se le coloca el "super" ya que hereda de la clase región y los atributos de la clase región deben irse a su padre.

Clase Kiosco: Clase que le da vida a los objetos de tipo Kiosco, a continuación, se muestran los atributos de la clase, así como su constructor.

```
public class Kiosco {
    3 usages
    private String codigoKiosco;
    3 usages
    private String nombreKiosco;
    3 usages
    private String codigoRegion;

1 usage
public Kiosco(String codigoKiosco, String nombreKiosco, String codigoRegion){
    this.codigoKiosco=codigoKiosco;
    this.nombreKiosco=nombreKiosco;
    this.codigoRegion=codigoRegion;
}
```

Clase Usuario: Clase que le da vida a los objetos de tipo Usuario, a continuación, se muestran los atributos de la clase, así como su constructor.

Clase Tarjeta: Clase que le da vida a los objetos de tipo Tarjeta, a continuación, se muestran los atributos de la clase, así como su constructor.

```
public class Tarjeta {
    3 usages
    private String nombreDeLaTarjeta;
    3 usages
    private String tipoTarjeta;
    3 usages
    private String fechaVencimiento;
    3 usages
    private String numeroTarjeta;
    1 usage

public Tarjeta(String nombreDeLaTarjeta, String tipoTarjeta, String fechaVencimiento, String numeroTarjeta){
        this.nombreDeLaTarjeta=nombreDeLaTarjeta;
        this.tipoTarjeta=tipoTarjeta;
        this.fechaVencimiento=fechaVencimiento;
        this.numeroTarjeta=numeroTarjeta;
}
```

Clase DatoFacturacion: Clase que le da vida a los objetos de tipo DatoFacturacion, a continuación, se muestran los atributos de la clase, así como su constructor.

```
public class DatoFacturacion {
    3 usages
    private String nombreCompleto;
    3 usages
    private String direction;
    3 usages
    private String nit;
    1 usage

public DatoFacturacion(String nombreCompleto,String direction, String nit) {
        this.nombreCompleto=nombreCompleto;
        this.direction=direction;
        this.nit=nit;
}
```

**Clase Envio:** Clase que le da vida a los objetos de tipo Envio, a continuación, se muestran los atributos de la clase, así como su constructor.

```
public class Envio (

3 usages
private String codigoPaquete;
3 usages
private String tipoServicio;
3 usages
private String destinatario;
3 usages
private Double totalEnvio;
3 usages
private String tipoPago;
3 usages
private String tipoPago;
3 usages
private ArrayList<String> datosFactura;
3 usages
private ArrayList<String> datosFactura;
3 usages
private ArrayList<String> datosGouia;

2 usages

public Envio(String codigoPaquete, String tipoServicio, String destinatario, Double totalEnvio, String tipoPago, ArrayList<String> datosFactura, this.codigoPaquete=codigoPaquete;
this.tipoServicio=tipoServicio;
this.destinatario=destinatario;
this.tipoPago=tipoPago;
this.tipoPago=tipoPago;
this.datosGouia=datosGouia;

1 }
```

## Librerías utilizadas:

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.File;
import java.io.IOException;
import javax.swing.*;
import java.awt.*;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.util.ArrayList;
import java.util.Objects;
```

Awt y swing son librerías que se importaron para todo lo relacionado con interfaz gráfica, también se usó la librería Java.io.File para la creación de los reportes HTML.

La librería java.util.Objects se utilizó solamente para hacer comparaciones entre Strings, ya que al usar solamente el operador "==", a veces no suele ser muy preciso.

La librería java.util.Arraylist se utilizó para almacenar de forma dinámica todos los objetos e información requerida.