



Ingeniería en Software

**INFORME
PROGRAMACIÓN ORIENTADA A OBJETOS**

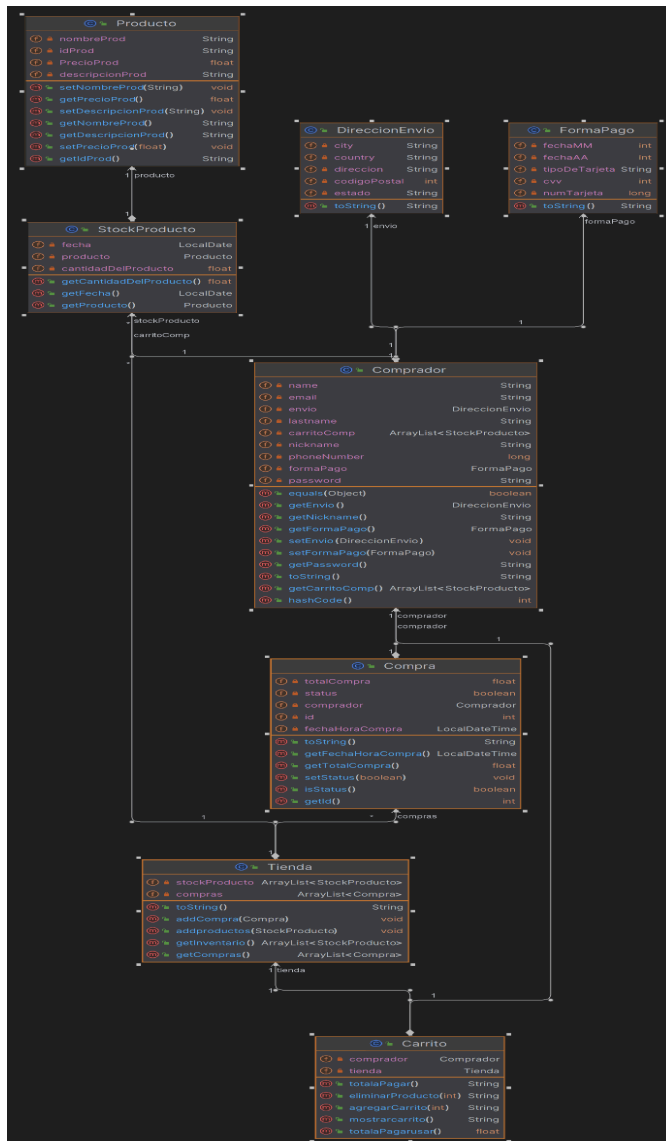
<Carrito de compras>

<Marcos Daniel Brindis Esquinca>

Suchiapa, 2024

Fase de análisis

Diagrama de clases

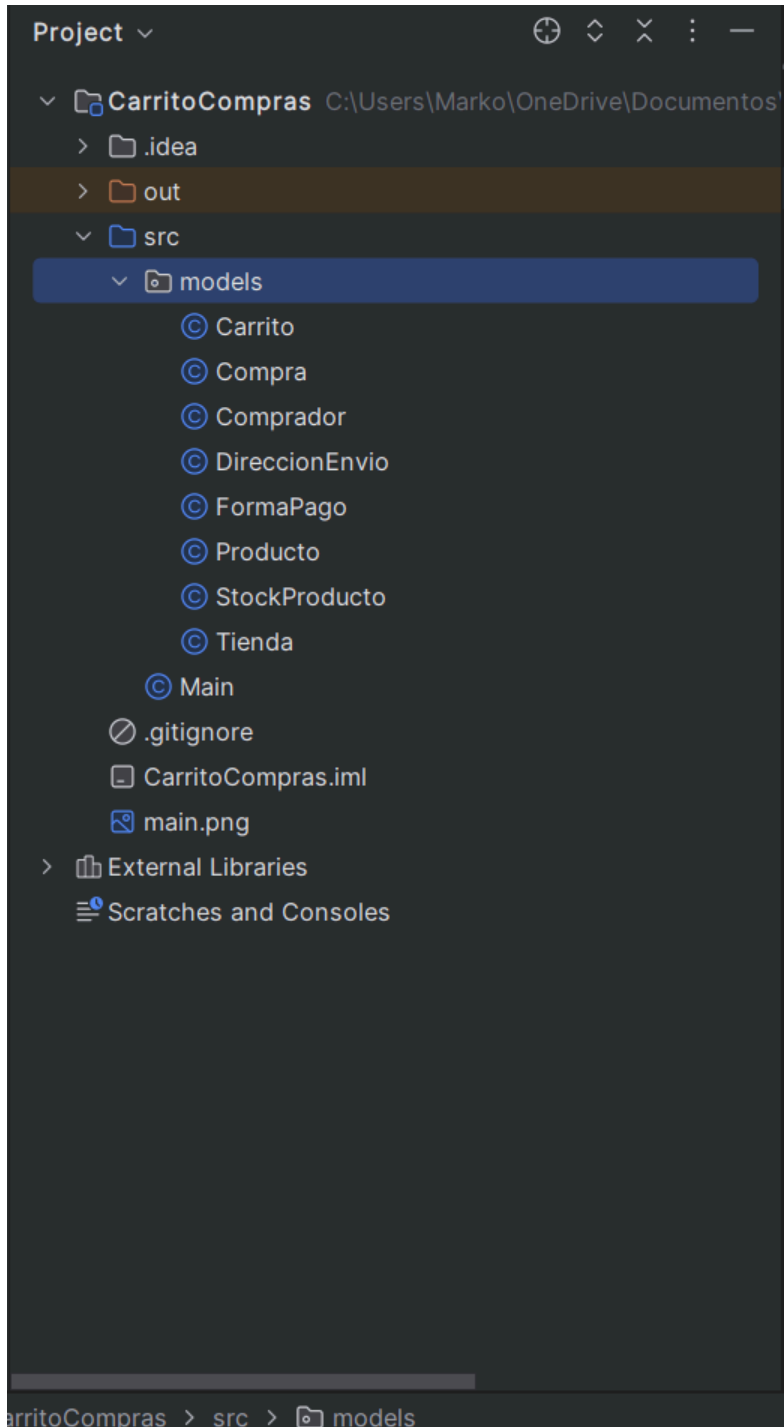


Repositorio del proyecto

Repositorio del proyecto de github

Fase de desarrollo

Estructura de carpetas



Código

```
package models;

import java.time.LocalDateTime;

public class Compra {
    private boolean status;
    private int id;
    private LocalDateTime fechaHoraCompra;
    private Comprador comprador;
    private float totalCompra;

    public Compra(boolean status, int id, LocalDateTime fechaHoraCompra, Comprador comprador, float totalCompra) {
        this.status = status;
        this.id = id;
        this.fechaHoraCompra = fechaHoraCompra;
        this.comprador = comprador;
        this.totalCompra = totalCompra;
    }

    @Override
    public String toString() {
        return "Compra{" +
            "status=" + status +
            ", id=" + id +
            ", fechaHoraCompra=" + fechaHoraCompra +
            ", comprador=" + comprador +
            ", totalCompra=" + totalCompra +
            '}';
    }

    public boolean isStatus() {
        return status;
    }

    public float getTotalCompra() {
        return totalCompra;
    }

    public int getId() {
        return id;
    }

    public void setStatus(boolean status) {
        this.status = status;
    }

    public LocalDateTime getFechaHoraCompra() {
        return fechaHoraCompra;
    }
}
```

```

package models;

import java.util.ArrayList;
import java.util.Objects;

public class Comprador {
    private String nickname;
    private String name;
    private String lastname;
    private String email;
    private long phoneNumber;
    private String password;
    private FormaPago formaPago;
    private DireccionEnvio envio;
    private ArrayList<StockProducto> carritoComp = new ArrayList<>();

    @Override
    public String toString() {
        return "Comprador{" +
            "nickname='" + nickname + '\'' +
            ", name='" + name + '\'' +
            ", lastname='" + lastname + '\'' +
            ", email='" + email + '\'' +
            ", phoneNumber=" + phoneNumber +
            ", \nforma de pago:" + (formaPago != null ? formaPago.toString() : "sin rellenar") +
            ", \ndatos envio " + (envio != null ? envio.toString() : "sin rellenar") +
            '}';
    }

    public ArrayList<StockProducto> getCarritoComp() {
        return carritoComp;
    }

    public String getNickname() {
        return nickname;
    }

    public String getPassword() {
        return password;
    }

    public void setFormaPago(FormaPago formaPago) {
        this.formaPago = formaPago;
    }

    public void setEnvio(DireccionEnvio envio) {
        this.envio = envio;
    }

    public FormaPago getFormaPago() {
        return formaPago;
    }

    public DireccionEnvio getEnvio() {
        return envio;
    }

    public Comprador() {
    }

    public Comprador(String nickname, String name, String lastname, String email, long phoneNumber,
        String password, FormaPago formaPago, DireccionEnvio envio) {
        this.nickname = nickname;
        this.name = name;
        this.lastname = lastname;
        this.email = email;
        this.phoneNumber = phoneNumber;
        this.password = password;
        this.formaPago = formaPago;
        this.envio = envio;
    }

    public Comprador(String nickname) {
        this.nickname = nickname;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Comprador comprador = (Comprador) o;
        return Objects.equals(nickname, comprador.nickname);
    }

    @Override
    public int hashCode() {
        return Objects.hash(nickname);
    }
}

```

```

package models;

public class Carrito {
    private Comprador comprador;
    private Tienda tienda;

    public Carrito(Comprador comprador, Tienda tienda) {
        this.comprador = comprador;
        this.tienda = tienda;
    }

    public String agregarCarrito(int selector) {
        String imprimir;
        if (selector > 0 && selector <= tienda.getInventario().size()) {
            StockProducto stockProducto = tienda.getInventario().get(selector - 1);
            comprador.getCarritoComp().add(stockProducto);
            imprimir = "Producto agregado al carrito.";
        } else {
            imprimir = "Número de producto no válido.";
        }
        return imprimir;
    }

    public String mostrarcarrito() {
        String imprimir = "";
        if (comprador.getCarritoComp().isEmpty()) {
            imprimir = "El carrito está vacío.";
        } else {
            int i = 0;
            for (StockProducto stockProducto : comprador.getCarritoComp()) {
                i++;
                Producto producto = stockProducto.getProducto();
                imprimir += i + " " + producto.getNombreProd() + "\n$" + "precio: " +
                producto.getPrecioProd();
                imprimir += "\n-----\n";
            }
            return imprimir;
        }
    }

    public String eliminarProducto(int seleccionarElim) {
        String imprimir;
        if (comprador.getCarritoComp().isEmpty()) {
            imprimir = "El carrito está vacío. No hay productos para eliminar.";
        } else if (seleccionarElim > 0 && seleccionarElim <= comprador.getCarritoComp().size()) {
            StockProducto productoAEliminar = comprador.getCarritoComp().get(seleccionarElim - 1);
            comprador.getCarritoComp().remove(productoAEliminar);
            imprimir = "Producto eliminado del carrito.";
        } else if (seleccionarElim == 0) {
            imprimir = "Operación cancelada.";
        } else {
            imprimir = "Número de producto no válido.";
        }
        return imprimir;
    }

    public String totalaPagar() {
        String imprimir = "";
        float total = 0;
        for (StockProducto stockProducto : comprador.getCarritoComp()) {
            Producto producto = stockProducto.getProducto();
            total += producto.getPrecioProd();
        }
        imprimir = "total: $" + total;
        return imprimir;
    }

    public float totalaPagarusar() {
        float total = 0;
        for (StockProducto stockProducto : comprador.getCarritoComp()) {
            Producto producto = stockProducto.getProducto();
            total += producto.getPrecioProd();
        }
        return total;
    }
}

```

```
package models;

import java.util.ArrayList;

public class Tienda {
    private ArrayList<StockProducto> stockProducto = new ArrayList<>
(); private ArrayList<Compra> compras = new ArrayList<>();

    public ArrayList<Compra> getCompras() {
        return compras;
    }

    @Override
    public String toString() {
        return "Tienda{" +
            "compras=" + compras +
            '}';
    }

    public void addproductos(StockProducto inventario) {
        stockProducto.add(inventario);
    }

    public ArrayList<StockProducto> getInventario() {
        return stockProducto;
    }

    public void addCompra(Compra nuevaCompra) {
        compras.add(nuevaCompra);
    }
}
```



```
package models;

public class FormaPago {
    private String tipoDeTarjeta;
    private long numTarjeta;
    private int fechaMM;
    private int fechaAA;
    private int cvv;

    public FormaPago(String tipoDeTarjeta, long numTarjeta, int fechaMM, int fechaAA, int cvv)
    {
        this.tipoDeTarjeta = tipoDeTarjeta;
        this.numTarjeta = numTarjeta;
        this.fechaMM = fechaMM;
        this.fechaAA = fechaAA;
        this.cvv = cvv;
    }

    @Override
    public String toString() {
        return
            "tipoDeTarjeta=" + tipoDeTarjeta + '\'' +
            ", numTarjeta=" + numTarjeta +
            ", fechaMM=" + fechaMM +
            ", fechaAA=" + fechaAA +
            ", cvv=" + cvv
        ;
    }
}
```

```
package models;

public class DireccionEnvio {
    private String country;
    private String estado;
    private String city;
    private String direccion;
    private int codigoPostal;

    public DireccionEnvio(String country, String estado, String city, String direccion, int
codigoPostal) {
        this.country = country;
        this.estado = estado;
        this.city = city;
        this.direccion = direccion;
        this.codigoPostal = codigoPostal;
    }

    @Override
    public String toString() {
        return
            "pais='" + country + '\'' +
            ", estado='" + estado + '\'' +
            ", ciudad='" + city + '\'' +
            ", direccion='" + direccion + '\'' +
            ", codigo Postal=" + codigoPostal
        ;
    }
}
```



```
package models;

import java.util.UUID;

public class Producto {
    private String nombreProd;
    private String descripcionProd;
    private String idProd;
    private float PrecioProd;

    public Producto() {
        this.idProd = UUID.randomUUID().toString();
    }

    public String getNombreProd() {
        return nombreProd;
    }

    public void setNombreProd(String nombreProd) {
        this.nombreProd = nombreProd;
    }

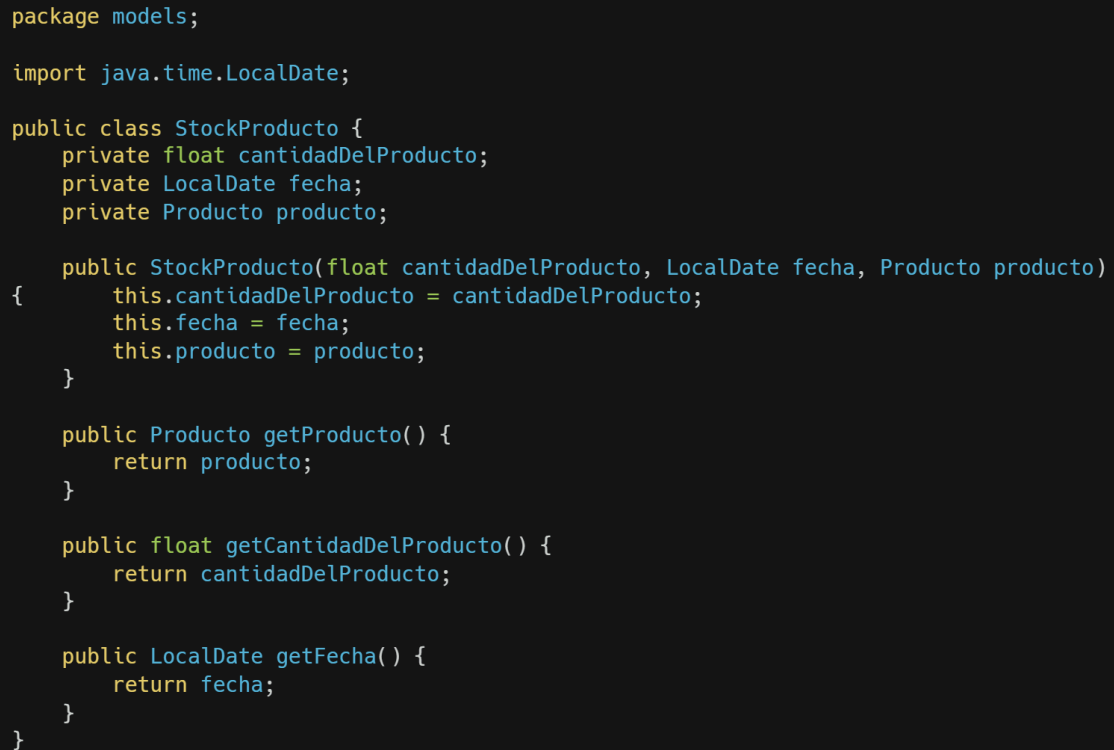
    public String getDescripcionProd() {
        return descripcionProd;
    }

    public void setDescripcionProd(String descripcionProd)
    {
        this.descripcionProd = descripcionProd;
    }

    public String getIdProd() {
        return idProd;
    }

    public float getPrecioProd() {
        return PrecioProd;
    }

    public void setPrecioProd(float precioProd) {
        PrecioProd = precioProd;
    }
}
```



```
package models;

import java.time.LocalDate;

public class StockProducto {
    private float cantidadDelProducto;
    private LocalDate fecha;
    private Producto producto;

    public StockProducto(float cantidadDelProducto, LocalDate fecha, Producto producto)
    {
        this.cantidadDelProducto = cantidadDelProducto;
        this.fecha = fecha;
        this.producto = producto;
    }

    public Producto getProducto() {
        return producto;
    }

    public float getCantidadDelProducto() {
        return cantidadDelProducto;
    }

    public LocalDate getFecha() {
        return fecha;
    }
}
```

el main no se subió captura debido a que el código era demasiado extenso.

Resultados

Interfaz visual

¿Con qué rol quieres iniciar?

1) administrador

2) comprador

3) finalizar

1) Agregar producto

2) mostrar información del producto

3) validar ventas

4) regresar

```

1
Ingrese nombre del producto:
asda
Ingrese una descripción del producto:
dsf
Ingrese el precio:
455
¿Cuántos de estos productos estás agregando?
46
¿Desea agregar otro? (s/n)
n

```

- 1) crear usuario
- 2) ingresar usuario
- 3) ver tienda
- 4) regresar

```

*****
****bienvenido*****
*****

```

- 1) comprar productos
- 2) abrir carrito de compras
- 3) cerrar sesión

```

*****bienvenido*****

```

Los productos son:

1) Nombre: asda

Descripción: dsf

Precio: \$455.0

Cantidad maxima que puede comprar: 46.0

coloque el numero del articulo que desea agregar al carrito

0) regresar

El carrito está vacío.

total: \$0.0

1) eliminar producto

2) generar peticion de compra

3) regresar

Comprador{nickname='mark', name='marcos', lastname='brindis', email='markos_bri', phoneNumber=456,

forma de pago:sin rellenar,

datos envio sin rellenar}

1) modificar datos

2) solicitar pago

3) status

4) regresar

1) modificar método de pago

2) modificar direccion de envio

3) regresar

Conclusiones

- se utilizó una metodología deductiva partiendo de un modelo general, para poder desarrollar los aspectos particulares de cada caso, se partió a partir de las historias de usuarios donde se describe que hay dos usuarios para el programa un administrador quien será el que tenga el control de las ventas y un comprador el cual es el que tendrá una mayor interacción con el programa.
- Los aspectos que solicitaban en las historias de usuario eran que el administrador pudiera agregar productos así como también el validará las compras realizadas por otros usuarios; el comprador puede agregar cosas al carrito de compras, y verificar el estado de este.
- Dentro de las técnicas que se utilizaron está realizar un modelo de dominio el cual comienza a clasificar y modelar la funcionalidades.
- se prosiguió a hacer los diagramas de clases, para especificar los atributos y métodos.
- Una vez realizados los diagramas se diseñaron las interfaces de forma sencilla, ya que la salida es por consola, mas sin embargo se necesita que el programa sea fluido y fácil de manejar.
- durante la realización del programa llegué a comprender diversos aspectos que desconocía de la programación orientada a objetos y por ende tuve que realizar diversas variaciones que a mi parecer eran necesarias, para el correcto funcionamiento.

- Dentro de las distintas clases del programa entiendo que hay clases que pueden sobrar (stock).