Instituto Tecnologico de Culiacan



Modelado Orientado a Objetos

Dr. Ricardo Rafael Quintero Meza

Patrón de Diseño Strategy (Mi Tiendita)

Elaborado por: Oscar Omar Sanchez Alvarado

[**Requisitos-minipunto de venta**](#_3zdrgp31vmt1) **3**

[**Modelo de Diseño**](#_ncn4y6rthgow) **4**

[Vista Estructural](#_hjqhhvogx5kw) 4

[Vista Dinámica](#_awt2pcubvxu) 5

[**Codigo**](#_xxlr3xm5oxa6) **6**

[app/MainMiTiendita.java](#_ymvtiofvnw64) 6

[app/daysOfSale/MondaySale.java](#_a0h7qubl6eou) 8

[app/daysOfSale/TuesdaySale.java](#_kywpzad8s3d) 8

[app/daysOfSale/WednesdaySale.java](#_fh86vvf75fg9) 8

[app/daysOfSale/ThursdaySale.java](#_3xb3uyjap7ht) 8

[app/daysOfSale/FridaySale.java](#_ru32y9cuni99) 9

[app/daysOfSale/SaturdaySale.java](#_pv0sovejbtaf) 9

[app/daysOfSale/SundaySale.java](#_n1ng8ti0oc1) 9

[app/domain/ProductCategory.java](#_qkkqg4ckq38l) 9

[app/domain/entities/Product.java](#_7qpno5mmyy60) 10

[framework/Sale.java](#_op2lru9zhhic) 11

[framework/behaviors/discountBehaviors/DiscountDayBehavior.java](#_r8tcuf6j4fqt) 11

[framework/behaviors/discountBehaviors/DayWithoutDiscount.java](#_cwgsh1zhewwc) 12

[framework/behaviors/discountBehaviors/FruitDiscountDay.java](#_2stc65ecmswm) 12

[framework/behaviors/discountBehaviors/MenonitaDiscountDay.java](#_mszykxml7e8i) 13

[framework/behaviors/discountBehaviors/OlderAdultDiscountDay.java](#_hzms1y7rovlv) 13

[framework/behaviors/discountBehaviors/SausageAndDairyDiscountDay.java](#_x6vufeush6m) 14

[**Pruebas**](#_m9c2vvv8x4il) **15**

[Caso de prueba](#_j4gndhblqhva) 15

[SetUp](#_j1nr9he4xdxn) 15

[Prueba 1: Lunes (Día del adulto mayor), 5% en el total de la venta para clientes de la 3ª. Edad](#_l92zlzpjl6b0) 17

[Prueba 2: Martes no hay descuento](#_51nk81g0x6qm) 17

[Prueba 3: Miércoles (Día menonita), 10% en lácteos](#_g3o73sodroic) 17

[Prueba 4: Jueves (Día de la fruta), 15 % en frutas](#_f1moacspqapt) 18

[Prueba 5: Viernes (Día de embutidos y lácteos), 5% en embutidos y 15% en lácteos](#_b0eto2zhyjm) 18

[Prueba 6: Sábado No hay descuentos](#_amxg9af0qmcq) 18

[Prueba 7: Domingo (Día del adulto mayor), 5% en el total de la venta para clientes de la 3ª. Edad](#_hzexcrqcxamq) 18

[**Resultados**](#_y0c8xjbr8zme) **19**

[**Repositorio GIT**](#_88lcs9lpxkp0) **19**

## Requisitos-minipunto de venta

“Mi tiendita” ofrece el siguiente catálogo de productos de abarrote a sus clientes:

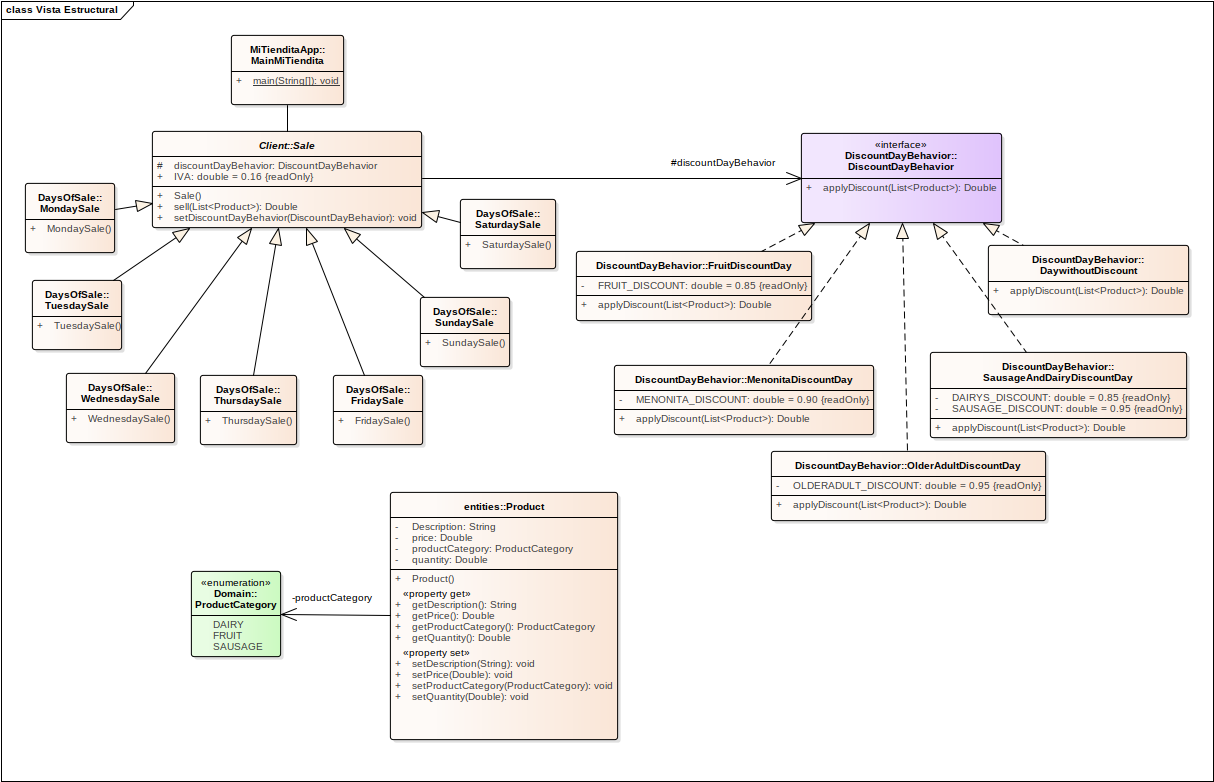
|  |  |
| --- | --- |
| **PRODUCTO** | **PRECIO UNITARIO** |
| Leche | 15.50 lt |
| Crema | 10.00 Kg |
| Mantequilla | 4.50 Kg |
| Jamón | 35.00 Kg |
| Queso | 100 Kg |
| Plátano | 14.00 Kg |
| Manzana | 13.00 Kg |

Se desea crear una aplicación que calcule el total de cualquier venta en base a la siguiente política de cobro:

**Total Venta = Suma (Precio Unitario \* Unidades) + Impuesto (IVA 16%) – Descuento.**

## Modelo de Diseño

#### **Vista Estructural**



#### 

#### 

#### 

#### 

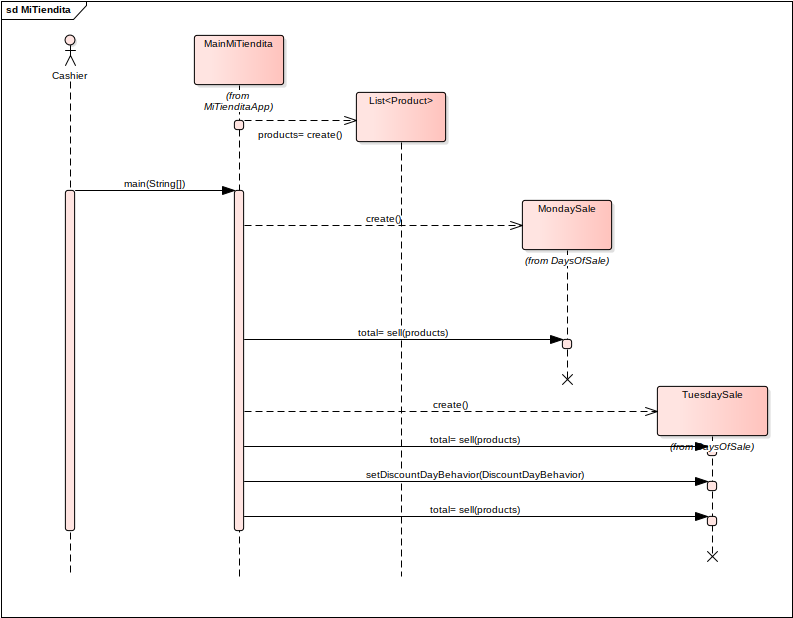
#### 

#### 

#### 

#### 

#### **Vista Dinámica**



## 

## Codigo

#### **app/MainMiTiendita.java**

public class MainMiTiendita {

public static void main(String[] args) {

Product leche = new Product();

leche.setDescription("Leche");

leche.setQuantity(10.0);

leche.setPrice(15.50);

leche.setProductCategory(ProductCategory.DAIRY);

Product crema = new Product();

crema.setDescription("Crema");

crema.setQuantity(0.5);

crema.setPrice(10.0);

crema.setProductCategory(ProductCategory.DAIRY);

Product mantequilla = new Product();

mantequilla.setDescription("Mantequilla");

mantequilla.setQuantity(0.250);

mantequilla.setPrice(4.50);

mantequilla.setProductCategory(ProductCategory.DAIRY);

Product queso = new Product();

queso.setDescription("Queso");

queso.setQuantity(0.500);

queso.setPrice(100.0);

queso.setProductCategory(ProductCategory.DAIRY);

Product platano = new Product();

platano.setDescription("Platano");

platano.setQuantity(1.0);

platano.setPrice(14.0);

platano.setProductCategory(ProductCategory.FRUIT);

Product manzana = new Product();

manzana.setDescription("Manzana");

manzana.setQuantity(2.0);

manzana.setPrice(13.0);

manzana.setProductCategory(ProductCategory.FRUIT);

Product jamon = new Product();

jamon.setDescription("Jamon");

jamon.setQuantity(1.0);

jamon.setPrice(35.0);

jamon.setProductCategory(ProductCategory.SAUSAGE);

List<Product> products = new ArrayList<>();

products.add(leche);

products.add(crema);

products.add(mantequilla);

products.add(queso);

products.add(platano);

products.add(manzana);

products.add(jamon);

Sale mondaySale = new MondaySale();

System.out.println("Total venta del lunes:" + mondaySale.sell(products));

Sale tuesdaySale = new TuesdaySale();

System.out.println("Total venta del martes:" + tuesdaySale.sell(products));

Sale wednesdaySale = new WednesdaySale();

System.out.println("Total venta del miercoles:" + wednesdaySale.sell(products));

Sale thursdaySale = new ThursdaySale();

System.out.println("Total venta del jueves:" + thursdaySale.sell(products));

Sale fridaySale = new FridaySale();

System.out.println("Total venta del viernes:" + fridaySale.sell(products));

Sale saturdaySale = new SaturdaySale();

System.out.println("Total venta del sabado:" + saturdaySale.sell(products));

Sale sundaySale = new SundaySale();

System.out.println("Total venta del domingo:" + sundaySale.sell(products));

}

}

#### **app/daysOfSale/MondaySale.java**

public class MondaySale extends Sale {

public MondaySale() {

discountDayBehavior = new OlderAdultDiscountDay();

}

}

#### **app/daysOfSale/TuesdaySale.java**

public class TuesdaySale extends Sale {

public TuesdaySale() {

discountDayBehavior = new DaywithoutDiscount();

}

}

#### **app/daysOfSale/WednesdaySale.java**

public class WednesdaySale extends Sale {

public WednesdaySale() {

discountDayBehavior = new MenonitaDiscountDay();

}

}

#### **app/daysOfSale/ThursdaySale.java**

public class ThursdaySale extends Sale {

public ThursdaySale() {

discountDayBehavior = new FruitDiscountDay();

}

}

#### 

#### **app/daysOfSale/FridaySale.java**

public class FridaySale extends Sale {

public FridaySale() {

discountDayBehavior = new SausageAndDairyDiscountDay();

}

}

#### **app/daysOfSale/SaturdaySale.java**

public class SaturdaySale extends Sale {

public SaturdaySale() {

discountDayBehavior = new DaywithoutDiscount();

}

}

#### **app/daysOfSale/SundaySale.java**

public class SundaySale extends Sale {

public SundaySale() {

discountDayBehavior = new OlderAdultDiscountDay();

}

}

#### **app/domain/ProductCategory.java**

public enum ProductCategory {

DAIRY,

FRUIT,

SAUSAGE

}

#### **app/domain/entities/Product.java**

public class Product {

public Product() {}

private String Description;

private Double quantity;

private Double price;

private ProductCategory productCategory;

public String getDescription() {

return Description;

}

public void setDescription(String description) {

Description = description;

}

public Double getQuantity() {

return quantity;

}

public void setQuantity(Double quantity) {

this.quantity = quantity;

}

public Double getPrice() {

return price;

}

public void setPrice(Double price) {

this.price = price;

}

public ProductCategory getProductCategory() {

return productCategory;

}

public void setProductCategory(ProductCategory productCategory) {

this.productCategory = productCategory;

}

}

#### **framework/Sale.java**

public abstract class Sale {

public final Double IVA = 0.16;

protected DiscountDayBehavior discountDayBehavior;

public Sale() {}

public Double sell(List<Product> products) {

Double total = 0.0;

if (products != null && !products.isEmpty()) {

total = discountDayBehavior.applyDiscount(products);

if (total != null && total > 0) {

total += (total \* 0.16);

}

}

return total;

}

public void setDiscountDayBehavior(DiscountDayBehavior discountDayBehavior) {

this.discountDayBehavior = discountDayBehavior;

}

}

#### **framework/behaviors/discountBehaviors/DiscountDayBehavior.java**

public interface DiscountDayBehavior {

public Double applyDiscount(List<Product> products);

}

#### **framework/behaviors/discountBehaviors/DayWithoutDiscount.java**

public class DaywithoutDiscount implements DiscountDayBehavior {

@Override

public Double applyDiscount(List<Product> products) {

Double total = 0.0;

if (products != null && !products.isEmpty()) {

for (Product item : products) {

total += item.getPrice() \* item.getQuantity();

}

}

return total;

}

}

#### **framework/behaviors/discountBehaviors/FruitDiscountDay.java**

public class FruitDiscountDay implements DiscountDayBehavior {

//15 percent

private final Double FRUIT\_DISCOUNT = 0.85;

@Override

public Double applyDiscount(List<Product> products) {

Double total = 0.0;

if (products != null && !products.isEmpty()) {

for (Product item : products) {

if (ProductCategory.FRUIT.equals(item.getProductCategory())) {

total += (item.getPrice() \* item.getQuantity()) \* FRUIT\_DISCOUNT;

} else {

total += item.getPrice() \* item.getQuantity();

}

}

}

return total;

}

}

#### **framework/behaviors/discountBehaviors/MenonitaDiscountDay.java**

public class MenonitaDiscountDay implements DiscountDayBehavior {

// 10 percent

private final Double MENONITA\_DISCOUNT = 0.90;

@Override

public Double applyDiscount(List<Product> products) {

Double total = 0.0;

if (products != null && !products.isEmpty()) {

for (Product item : products) {

if (ProductCategory.DAIRY.equals(item.getProductCategory())) {

total += (item.getPrice() \* item.getQuantity()) \* MENONITA\_DISCOUNT;

} else {

total += item.getPrice() \* item.getQuantity();

}

}

}

return total;

}

}

#### **framework/behaviors/discountBehaviors/OlderAdultDiscountDay.java**

public class OlderAdultDiscountDay implements DiscountDayBehavior {

// 10 percent

private final Double OLDERADULT\_DISCOUNT = 0.95;

@Override

public Double applyDiscount(List<Product> products) {

Double total = 0.0;

if (products != null && !products.isEmpty()) {

for (Product item : products) {

total += (item.getPrice() \* item.getQuantity()) \* OLDERADULT\_DISCOUNT;

}

}

return total;

}

}

#### **framework/behaviors/discountBehaviors/SausageAndDairyDiscountDay.java**

public class SausageAndDairyDiscountDay implements DiscountDayBehavior {

// 5 percent

private final Double SAUSAGE\_DISCOUNT = 0.95;

// 15 percent

private final Double DAIRYS\_DISCOUNT = 0.85;

@Override

public Double applyDiscount(List<Product> products) {

Double total = 0.0;

if (products != null && !products.isEmpty()) {

for (Product item : products) {

if (item.getProductCategory() != null) {

switch (item.getProductCategory()) {

case DAIRY:

total += (item.getPrice() \* item.getQuantity()) \* DAIRYS\_DISCOUNT;

break;

case SAUSAGE:

total += (item.getPrice() \* item.getQuantity()) \* SAUSAGE\_DISCOUNT;

break;

default:

total += item.getPrice() \* item.getQuantity();

break;

}

}

}

}

return total;

}

}

## Pruebas

#### **Caso de prueba**

Considerando el siguiente mandado:

10 lts. De leche = $155

0.5 Kg. De Crema = $5

250 gr. De Mantequilla = $1.125

500 gr. De Queso = $50

**Total lacteos = $211.125**

1 Kg. De plátanos = $14

2 Kg. De Manzana = $26

**Total frutas = $40**

1 kg de jamon - 35 = $35

**Total embutidos = $35**

------------------------------------------------------

**Total general = $286.125**

Demuestre el funcionamiento de su programa para cada día de la semana, mostrando el total de la venta así como una explicación de los descuentos aplicados.

#### **SetUp**

@Before

public void setUp() {

Product leche = new Product();

leche.setDescription("Leche");

leche.setQuantity(10.0);

leche.setPrice(15.50);

leche.setProductCategory(ProductCategory.DAIRY);

Product crema = new Product();

crema.setDescription("Crema");

crema.setQuantity(0.5);

crema.setPrice(10.0);

crema.setProductCategory(ProductCategory.DAIRY);

Product mantequilla = new Product();

mantequilla.setDescription("Mantequilla");

mantequilla.setQuantity(0.250);

mantequilla.setPrice(4.50);

mantequilla.setProductCategory(ProductCategory.DAIRY);

Product queso = new Product();

queso.setDescription("Queso");

queso.setQuantity(0.500);

queso.setPrice(100.0);

queso.setProductCategory(ProductCategory.DAIRY);

Product platano = new Product();

platano.setDescription("Platano");

platano.setQuantity(1.0);

platano.setPrice(14.0);

platano.setProductCategory(ProductCategory.FRUIT);

Product manzana = new Product();

manzana.setDescription("Manzana");

manzana.setQuantity(2.0);

manzana.setPrice(13.0);

manzana.setProductCategory(ProductCategory.FRUIT);

Product jamon = new Product();

jamon.setDescription("Jamon");

jamon.setQuantity(1.0);

jamon.setPrice(35.0);

jamon.setProductCategory(ProductCategory.SAUSAGE);

products = new ArrayList<>();

products.add(leche);

products.add(crema);

products.add(mantequilla);

products.add(queso);

products.add(platano);

products.add(manzana);

products.add(jamon);

}

#### **Prueba 1: Lunes (Día del adulto mayor), 5% en el total de la venta para clientes de la 3ª. Edad**

Total general - 5% descuento + IVA = (286.125 - 14.30625) + 43.491 = **$315.30975**

@Test

public void mondayTotalIsEqualToRealValue() {

venta = new MondaySale();

venta.sell(products);

Assert.assertEquals(Double.valueOf(315.30975),venta.sell(products));

}

#### **Prueba 2: Martes no hay descuento**

Total general + IVA = 286.125 + 45.78 = **$331.905**

@Test

public void tuesdayTotalIsEqualToRealValue() {

venta = new TuesdaySale();

venta.sell(products);

Assert.assertEquals(Double.valueOf(331.905),venta.sell(products));

}

#### **Prueba 3: Miércoles (Día menonita), 10% en lácteos**

Total lacteos - 10% descuento = 211.125 - 21.1125 = **$190.0125**

Total frutas = $40

Total embutidos = $35

Total general + IVA = 265.0125 + 42.402 = **$307.4145**

@Test

public void wednesdayTotalIsEqualToRealValue() {

venta = new WednesdaySale();

venta.sell(products);

Assert.assertEquals(Double.valueOf(307.4145),venta.sell(products));

}

#### **Prueba 4: Jueves (Día de la fruta), 15 % en frutas**

Total frutas - 15% descuento = 40 - 6 = **$34**

Total lacteos = $211.125

Total embutidos = $35

Total general + IVA = 280.125 + 44.82 = **$324.945**

#### **Prueba 5: Viernes (Día de embutidos y lácteos), 5% en embutidos y 15% en lácteos**

Total lacteos - 15% de descuento = 211.125 - 31.66875 = **$179.45625**

Total embutidos - 5% de descuento = 35 - 1.75 = **$33.25**

Total frutas - 15% descuento = $40

Total general + IVA = 252.70625 + 40.433 = **$293.13925**

@Test

public void fridayTotalIsEqualToRealValue() {

venta = new FridaySale();

venta.sell(products);

Assert.assertEquals(Double.valueOf(293.13925),venta.sell(products));

}

#### **Prueba 6: Sábado No hay descuentos**

Total general + IVA = 286.125 + 45.78 = **$331.905**

@Test

public void saturdayTotalIsEqualToRealValue() {

venta = new SaturdaySale();

venta.sell(products);

Assert.assertEquals(Double.valueOf(331.905),venta.sell(products));

}

#### 

#### **Prueba 7: Domingo (Día del adulto mayor), 5% en el total de la venta para clientes de la 3ª. Edad**

Total general - 5% descuento + IVA = (286.125 - 14.30625) + 43.491 = **$315.30975**

@Test

public void sundayTotalIsEqualToRealValue() {

venta = new SundaySale();

venta.sell(products);

Assert.assertEquals(Double.valueOf(315.30975),venta.sell(products));

}

## Resultados

Total venta del lunes: **$315.30975**

Total venta del martes: **$331.905**

Total venta del miercoles: **$307.4145**

Total venta del jueves: **$324.945**

Total venta del viernes: **$293.13925**

Total venta del sabado: **$331.905**

Total venta del domingo: **$315.30975**

## Repositorio GIT

https://github.com/oscarsancheza/MiTiendita