

[Home](#)[About](#)[Content](#)[Others](#)

PROYECTO: *GESTIÓN DE VENTAS* *EN UNA PASTELERÍA*

Curso: Programación avanzada

Docente: MSc. Ing. Hugo Manuel Barraza Vizcarra

Integrantes:

Diesdanderson Dudu Vela Laurente	2025-119060
Elvis Juan Yana Cayo	2025-119048
Wilfredo Valdivia Vera	2025-119007



Antecedentes

En el Perú, muchos emprendimientos de repostería, principalmente liderados por jóvenes, gestionan sus procesos de forma manual, lo que genera errores y dificultades en la administración. Por ello, es necesario implementar sistemas informáticos simples que optimicen la gestión de inventarios y ventas y fortalezcan la sostenibilidad del negocio.



Planteamiento del problema

PROBLEMA

Los emprendimientos de repostería en el Perú, especialmente los liderados por jóvenes, presentan deficiencias en la gestión de inventarios y ventas debido al uso de métodos manuales. Esta situación genera errores en el control de stock, pérdida de información y dificultades en la planificación de la producción.

JUSTIFICACIÓN

La implementación de un sistema informático de gestión se justifica por la necesidad de automatizar y organizar los procesos internos del emprendimiento. Un sistema permitirá reducir errores humanos, mejorar el control de insumos y facilitar la toma de decisiones.

ALCANCE

El sistema permitirá controlar el stock actual y mínimo, registrar la producción mediante recetas, administrar insumos en el almacén y registrar las ventas, incluyendo la generación y almacenamiento de boletas.



OBJETIVOS

GENERAL

Desarrollar un sistema basado en programación orientada a objetos que automatice la gestión de inventario, producción y ventas de un emprendimiento de repostería, mejorando la organización y la toma de decisiones.

ESPECÍFICOS

El proyecto consiste en aplicar principios de Programación Orientada a Objetos, diseñar el sistema mediante diagramas UML e implementar la gestión de productos, ventas y control de stock, incluyendo la generación y almacenamiento de boletas de venta.



```
"container">
class="row">
v class="col-md-6 col-lg-8"> <!-- _____
<nav id="nav" role="navigation">
  <ul>
    <li><a href="index.html">Home</a></li>
    <li><a href="home-events.html">Home
    <li><a href="multi-col-menu.html">Mu
    <li class="has-children"> <a href="#"
      <ul>
        <li><a href="tall-button-hea
        <li><a href="image-logo.html
        <li class="active"><a href="#"
      </ul>
    </li>
    <li class="has-children"> <a href="#"
      <ul>
        <li><a href="variable-width-s
```


Receta
<ul style="list-style-type: none"> - IdReceta: int - nombreProducto: string - insumos : vector<itemReceta>
<ul style="list-style-type: none"> + Receta(string) + cargarDefinicion(string): bool + producir(Almacen): bool + agregarInsumo(string) :void + crearReceta() : void

Almacen
<ul style="list-style-type: none"> - insumos: vector<Insumo>
<ul style="list-style-type: none"> + cargarBase() : void + guardarBase() : void + mostrarInsumos(): void + reponerInsumos() : void + hayStock(string, float): bool + descontar(string, float):bool

itemReceta
<ul style="list-style-type: none"> + nombreInsumo: string + cantidadUtil: float + unidad : string

Insumo
<ul style="list-style-type: none"> - nombre : string - stock: float - unidad: string
<ul style="list-style-type: none"> + Insumo(string, float, string) + getNombre(): string + getStock(): float + getUnidad(): string + consumir(float): bool + reponer(float): void



Inventario
<ul style="list-style-type: none"> - productos : vector<Producto*> - contadores : map<char,int> - formatearId() : string
<ul style="list-style-type: none"> + agregarProducto(Producto*): void + actualizarStock(): void + producir + buscarPorId(iint) : Producto* + venderProducto (int,int) : void + mostrarProductos() : void + cargarProductos() : void + guardarProductos() : void + mostrarBajoMinimo() : void

Producto
<ul style="list-style-type: none"> - idProducto: string - nombre: string - stockActual : int - stockMinimo : int - precio : float
<ul style="list-style-type: none"> + Producto() + Producto(string, int , int) + virtual ~Producto() = default + getStockActual(): int + getStockMinimo(): int + getPrecio(): float + getIdProducto(): string + getNombre(): string + setId(string): void + {virtual} actualizarStock(int) = 0: void + {virtual} prefijold() = 0; char + {virtual} mostrarDatos() : void + {virtual} ssdatos() : string + {virtual} cargarDesdeStream(stringstream&)= 0: void + producir(int cant, Almacen&) : int
<ul style="list-style-type: none"> + estaBajoMinimo() : bool + mostrarDatos() : void

Torta
<ul style="list-style-type: none"> - maxPorciones : int
<ul style="list-style-type: none"> + Torta() + Torta(string, int, int, int, float) + getMaxPorciones(): int + actualizarStock(int) : void + ssdatos() : string + mostrarDatos() : void + cargarDesdeStream() void + producir() : int + prefijold() = "T": char

ItemVenta

- producto : Producto
- cantidad : int
- subTotal : float

- + ItemVenta(Producto, int)
- + getSubtotal(): float
- + getProducto(): Producto
- + getCantidad(): int

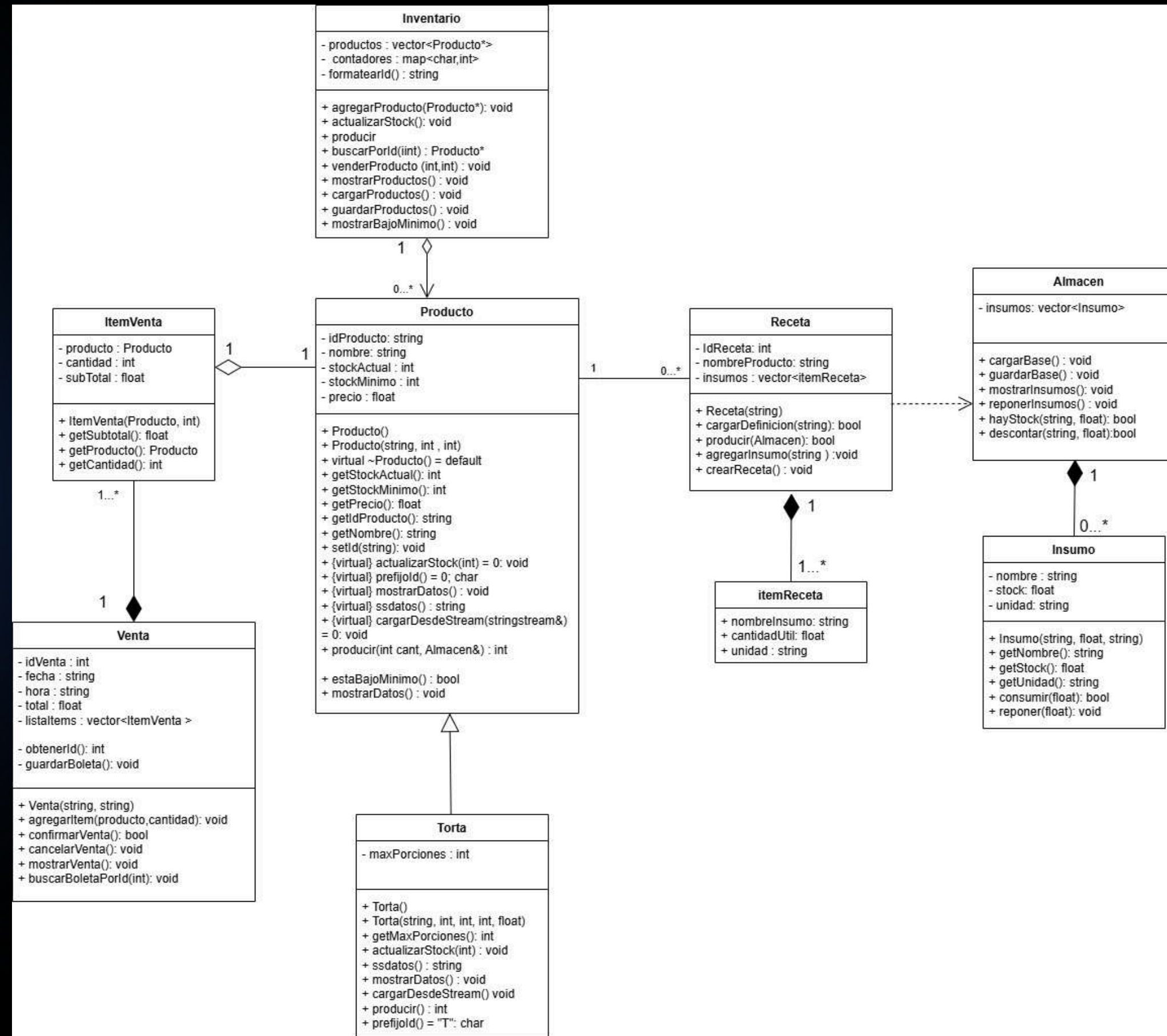
Venta

- idVenta : int
- fecha : string
- hora : string
- total : float
- listItems : vector<ItemVenta >

- obtenerId(): int
- guardarBoleta(): void

- + Venta(string, string)
- + agregarItem(producto,cantidad): void
- + confirmarVenta(): bool
- + cancelarVenta(): void
- + mostrarVenta(): void
- + buscarBoletaPorId(int): void

DIAGRAMA UML



THANK YOU