PROGRAMACION ORIENTADA A OBJETOS 1-SIN-3ª

Fabián Paredes

Contenido

Introducción

El siguiente documento describe la planeación par el desarrollo de un software para la gestión de un e-commerce en una aplicación desarrollada en Golang. En el siguiente documento se incluirán, el diagrama de clases, la lista de paquetes necesarios para el funcionamiento y las características del software.

Planeación

Diagrama de clases:

En el diagrama de clases del software seleccionado “Sistema de Gestión de e-commerce” pude identificar 4 clases principales:

- Clase Cliente:
  - Está compuesta por los datos:
    - idCliente: Identificador único del usuario de tipo Int.
    - nombre: Variable de tipo String que contiene el nombre del cliente.
    - email: Variable de tipo String que contiene el correo del cliente.
    - direccion: Variable de tipo String que contiene la dirección del cliente.
  - Funciones:
    - agregarCliente(): Función que permite agregar un nuevo cliente, encargada de generar un Id único para el cliente que se está agregando y gestionar la ingesta de los datos necesarios para cada cliente.
    - borrarCliente(): Función que permite realizar el borrado lógico del cliente.
    - actualizarDireccion(): Función que permite actualizar la dirección de un cliente.
- Clase Carrito:
  - Está compuesta por los datos:
    - idPedido: Identificador del pedido.

        - idCliente: Identificador del cliente al que pertenece el pedido
        - fecha: Fecha de creación del carrito.
        - productos[productos]: Array de productos con sus atributos.
    - Funciones:
        - agregarProductoCarrito(producto): Función responsable de agregar productos al carrito.
        - quitarProductoCarrito(producto): Función responsable de remover productos al carrito.
        - calcularTotal(): Función que calcula el total del carrito.
        - cancelarCarrito(): Función que cancela el carrito actual para un cliente determinado.

- Clase Pedidos:
    - Está compuesta por los datos:
        - idPedido: Identificador del pedido.
        - idCliente: Identificador del cliente al que pertenece el pedido
        - fecha: Fecha de creación del carrito.
        - productos[productos]: Array de productos con sus atributos.
        - estado: Es el estado del pedido: creado,pagado,entregado,cancelado
        - precio: Es el valor total del pedido.
    - Funciones:
        - agregarPedido(idCliente,productos[productos]): Función responsable de crear el pedido.
        - pagarPedido(): Función responsable de cambiar el estado a pagado.
        - entregarPedido(): Función responsable de cambiar el estado del pedido a entregado.

- cancelarPedido(): Función responsable de cambiar el estado del pedido a cancelado.

- Clase Productos:
  - Está compuesta por los datos:
    - codigo: Identificador del producto.
    - nombre: Nombre del producto.
    - precio: Precio del producto.
    - cantidad: Cantidad de productos en stock.
    - categorías: Array de strings con la lista de las categorías del producto.
  - Funciones:
    - agregar(nombre,precio,cantidad): Función encargada de agregar un nuevo producto.
    - quitar(codigo); Función responsable de hacer la baja lógica del producto.
    - revisarStock(codigo); Función que retorna true si hay productos en stock.
      - listarProductos(): Función responsable de listar todos los productos.

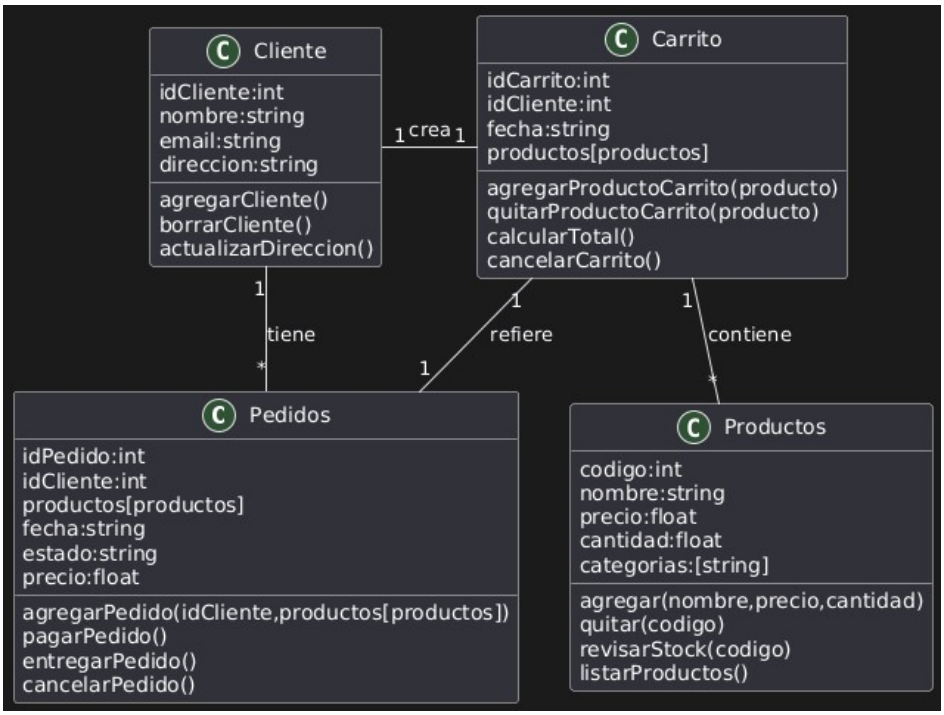

Diagrama de clases para un Sistema de Gestión de e-commerce.

Investigación de paquetes

Para poder determinar que paquetes eran necesarios para que este código funcione, primero me puse a pensar en como iba a consolidar estos datos en disco para mantener los datos.

Para este proyecto decidí utilizar archivos CSV por su facilidad de lectura y escritura. Al realizar una investigación llegué a la siguiente página Reading and Writing CSV Files in Golang - Golang Docs la cual contiene no solo ejemplos, sino que también las librerías necesarias para un proyecto como el mío:

```
package main
 import (
    "encoding/csv"
    "fmt"
    "os"
)
```