**PROYECTO FINAL**

**DRINKERS**

**Materia:** Programación I – Ing. Jose Jesus Cabrera

Base de datos I - Ing. Carmen Rosa Marin S.

**Alumnos:** Erwin Alejandro Ojeda Justiniano

Leandro Emiliano Miranda Roman

Santa Cruz – Bolivia

2024

**Introducción**

El presente proyecto consiste en un sistema para la gestión de una licorería. Este software tiene como finalidad facilitar la administración de las operaciones del negocio, permitiendo registrar y manejar información sobre clientes, productos, empleados y ventas.

**Objetivos**

* Automatizar la gestión del inventario de la licorería.
* Optimizar el registro y seguimiento de las ventas.
* Centralizar la información de clientes y empleados.

**Organización del Proyecto**

La estructura del proyecto está diseñada para mantener un flujo de trabajo organizado y eficiente.

**Estructura del proyecto**

1. **Carpeta “cerrar”**
   * Clases.py:
   * Close.py: Muestra mensaje, para saber si conecto o no las base de datos
   * Post.py:
2. **Carpeta “cerrarpo”**

* Closepo.py: Muestra mensaje acerca de la conexión a la base de datos en postgres

1. **Carpeta “clase”**
   * conexion.py: Configuración y gestión de la conexión a la base de datos PostgreSQL.
2. **Carpeta “funciones”**
   * agre.py: Incluye métodos para agregar registros.
   * dele.py: Métodos para eliminar registros específicos.
   * mod.py: Funciones para modificar datos existentes.
   * show.py: Encargado de la visualización de datos registrados.
3. **Carpeta “funpo”**
   * showpo.py: Muestra lo contenido en la base de datos de postgres
4. **Carpeta “tablas”**
   * Categoría.py: Visualiza e interactua en la tabla categoria
   * cliente.py: Modelo de la tabla para gestionar datos de clientes.
   * Detalle.py: Contiene todo para insertar, ver o eliminar un detalle de venta
   * producto.py: Modelo para los productos de la licorería.
   * empleado.py: Gestiona los registros de los empleados.
   * Subcategoria.py: Para visualizar todo lo contenido en la tabla subcategoria
   * ventas.py: Estructura de datos para el registro de ventas.
5. **Carpeta “tablaspo”**

Estas funciones son para la base de datos de postgres

* + categoriapo.py: Visualizar los registros en la tabla categoria
  + clientepo.py: Métodos relacionados con las operaciones de clientes.
  + detallepo.py: Contiene lo necesario para insertar, ver o eliminar registros de detalle de venta
  + empleadopo.py: Es para agregar modificar o eliminar datos de empleados
  + productospo.py: Funciones enfocadas en los productos, como ser agregar o eliminar productos, asi mismo modificar la cantidad.
  + subcategoriapo.py:
  + ventaspo.py: Manejo de las operaciones de las ventas.

1. **Otros archivos importantes**
   * faz.py: Punto de entrada principal del sistema.
   * migracion.py: Responsable de las migraciones de la base de datos.

**Requisitos del Sistema**

**Software Necesario**

* Python 3.10 o superior.
* PostgreSQL.
* Librerías adicionales:
  + psycopg2
  + tkinter

**Instalación y Configuración**

1. **Clonar el repositorio**

git clone [URL del repositorio]

1. **Instalar dependencias**

pip install -r requirements.txt

1. **Configurar la base de datos**
   * Crear una base de datos en PostgreSQL.
   * Ejecutar el script migracion.py para crear las tablas necesarias.
2. **Archivo de configuración**
   * Añadir las credenciales de conexión al archivo.

**Funcionalidades**

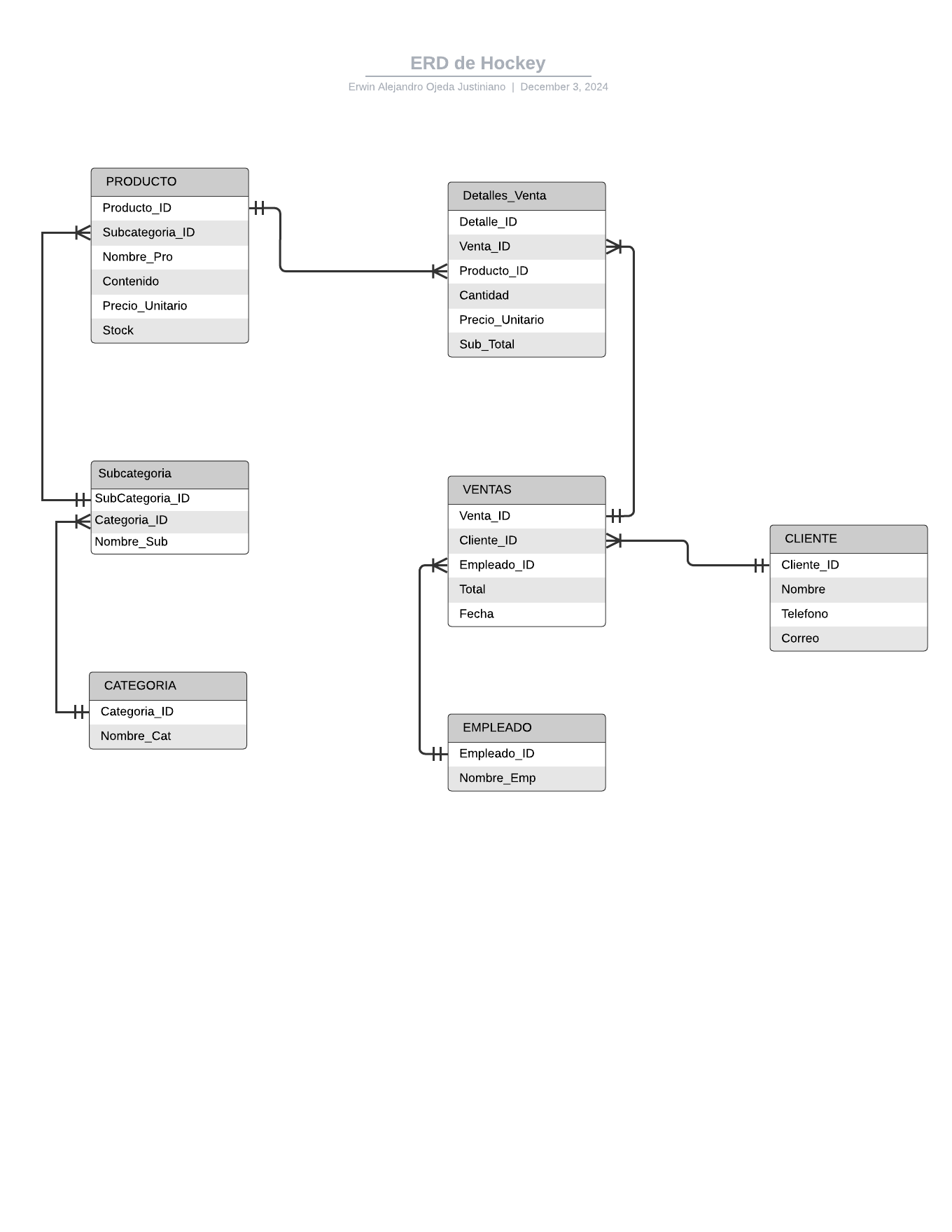
**Funciones principales del sistema**

1. **Gestión de clientes:** Registrar nuevos clientes ,consultar información de clientes existentes.
2. **Manejo de inventarios:** Añadir nuevos productos al catálogo, actualizar precios o stock de productos existentes.
3. **Registro de ventas:** Procesar transacciones de ventas, generar reportes relacionados con las ventas realizadas.
4. **Gestión de empleados:** Registrar y modificar datos de empleados.

**Descripción de la Base de Datos**

**Tablas principales**

1. **Tabla Clientes** (cliente.py):
   * **Campos:** cliente\_id(pk), nombre, email, telefono.
2. **Tabla Ventas** (ventas.py):
   * **Campos:** venta\_id(pk), empleado\_id(fk), cliente\_id(fk), fecha, total.
3. **Tabla Empleado** (empleado.py):
   * **Campos:** empleado\_id(pk), nombre\_emp.
4. **Tabla Detalles\_Venta** (detalle.py):
   * **Campos:** detalle\_id(pk), producto\_id(fk), venta\_id(fk), cantidad, sub\_total, precio\_unitario.
5. **Tabla Productos** (producto.py):
   * **Campos:** producto\_id(pk), subcategoria\_id(fk), nombre\_pro, precio\_unitario, contenido, stock.
6. **Tabla Subcategoria** (subcategoria.py):
   * **Campos:** subcategoria\_id(pk), categoría\_id(fk), nombre\_sub.
7. **Tabla Categoria** (categoria.py):
   * **Campos:** categoría\_id, nombre\_cat.



1. **Clientes**:
   * Relación 1:N con **Ventas**.
2. **Ventas**:
   * Relación 1:N con **Detalles\_Venta**.
   * Relación N:1 con **Empleado**.
3. **Detalles\_Venta**:
   * Relación N:1 con **Productos**.
4. **Productos**:
   * Relación N:1 con **Subcategoria**.
5. **Subcategoria**:
   * Relación N:1 con **Categoria**.

**Ejecución**

Para iniciar el sistema:

1. Ejecutar el archivo principal:

python faz.py

1. Acceder al sistema a través de la interfaz gráfica proporcionada.

Diagrama de Clase:

