ESTRUCTURA DE DATOS Y PROGRAMACIÓN ORIENTADA A OBJETOS

Tema Nº11:Proyecto: Implementación de clases del paquete Vista

Indicador de logro Nº11:Implementa las clases del proyecto utilizando las propiedades y métodos de clases contenidas en la librería PyQt5.

**TEMA 01 Teoría de los**

Imagen que contiene Icono

Descripción generada automáticamente

**TEMA Nº11:**

Proyecto: Implementación de clases del paquete Vista

**Subtema 11.1:**

Implementación de clases del paquete Vista

En esta parte del proyecto se implementarán las clases:

* login.py
* ventanaClientes.py
* ventanaDetalleVentas.py
* ventanaFacturas.py
* ventanaPrincipal.py
* ventanaProductos.py
* ventanaVentas.py

Se implementará también la clase main.py.

**login.py**

from PyQt5 import QtWidgets, uic

from PyQt5 import QtGui

from Vista.ventanaPrincipal import VentanaPrincipal

from Controlador.arregloProductos import \*

from Controlador.arregloClientes import \*

from Controlador.arregloDetalleVenta import \*

from Controlador.arregloFactura import \*

qtCreatorFile = "UI/login.ui" # Nombre del archivo aquí!!!

Ui\_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)

class Login(QtWidgets.QMainWindow):

    def \_\_init\_\_(self,parent = None):

        super(Login,self).\_\_init\_\_(parent)

        self.setWindowIcon(QtGui.QIcon("UI/imagenes/venta.png"))

        uic.loadUi("UI/login.ui",self)

        self.btnIniciar.clicked.connect(self.iniciarSesion)

        self.show()

    # Aquí van las nuevas funciones

    def iniciarSesion(self):

        usuario = self.txtUsuario.text().lower()

        contraseña = self.txtPassword.text()

        if usuario == "kenny" and contraseña == "123456":

            self.close()

            vprincipal = VentanaPrincipal(self)

            vprincipal.show()

        else:

            mensaje = QtWidgets.QMessageBox()

            mensaje.setWindowTitle("Punto de Venta")

            mensaje.setText("Los datos ingresados son incorrectos...!!!")

            mensaje.setIcon(QtWidgets.QMessageBox.Information)

**ventanaClientes.py**

from PyQt5 import QtWidgets, uic

from PyQt5 import QtGui

from Controlador.arregloProductos import \*

from Controlador.arregloClientes import \*

from Controlador.arregloDetalleVenta import \*

from Controlador.arregloFactura import \*

aCli = ArregloClientes()

class VentanaClientes(QtWidgets.QMainWindow):

    def \_\_init\_\_(self,parent = None):

        super(VentanaClientes,self).\_\_init\_\_(parent)

        self.setWindowIcon(QtGui.QIcon("UI/imagenes/venta.png"))

        uic.loadUi("UI/ventanaClientes.ui", self)

        self.btnRegistrar.clicked.connect(self.registrar)

        self.btnConsultar.clicked.connect(self.consultar)

        self.btnListar.clicked.connect(self.listar)

        self.btnEliminar.clicked.connect(self.eliminar)

        self.btnModificar.clicked.connect(self.modificar)

        self.btnGrabar.clicked.connect(self.grabar)

        self.listar()

        self.show()

    def obtenerDni(self):

        return self.txtDni.text()

    def obtenerNombres(self):

        return self.txtNombres.text()

    def obtenerApellidoPaterno(self):

        return self.txtApellidoPaterno.text()

    def obtenerApellidoMaterno(self):

        return self.txtApellidoMaterno.text()

    def obtenerDireccion(self):

        return self.txtDireccion.text()

    def obtenerTelefono(self):

        return self.txtTelefono.text()

    def limpiarTabla(self):

        self.tblClientes.clearContents()

        self.tblClientes.setRowCount(0)

    def valida(self):

        if self.txtDni.text() == "":

            self.txtDni.setFocus()

            return "DNI del cliente...!!!"

        elif self.txtNombres.text() == "":

            self.txtNombres.setFocus()

            return "Nombre del cliente...!!!"

        elif self.txtApellidoPaterno.text() == "":

            self.txtApellidoPaterno.setFocus()

            return "Apellido Paterno del cliente...!!!"

        elif self.txtApellidoMaterno.text() == "":

            self.txtApellidoMaterno.setFocus()

            return "Apellido Materno del cliente...!!!"

        elif self.txtDireccion.text() == "":

            self.txtDireccion.setFocus()

            return "Dirección del cliente...!!!"

        elif self.txtTelefono.text() == "":

            self.txtTelefono.setFocus()

            return "Teléfono del cliente...!!!"

        else:

            return ""

    def listar(self):

        self.tblClientes.setRowCount(aCli.tamañoArregloCliente())

        self.tblClientes.setColumnCount(6)

        self.tblClientes.verticalHeader().setVisible(False)

        for i in range(0, aCli.tamañoArregloCliente()):

            self.tblClientes.setItem(i, 0, QtWidgets.QTableWidgetItem(aCli.devolverCliente(i).getDniCliente()))

             self.tblClientes.setItem(i, 1, QtWidgets.QTableWidgetItem(aCli.devolverCliente(i).getNombresCliente()))

             self.tblClientes.setItem(i, 2, QtWidgets.QTableWidgetItem(aCli.devolverCliente(i).getApellidoPaternoCliente()))

             self.tblClientes.setItem(i, 3, QtWidgets.QTableWidgetItem(aCli.devolverCliente(i).getApellidoMaternoCliente()))

             self.tblClientes.setItem(i, 4, QtWidgets.QTableWidgetItem(aCli.devolverCliente(i).getDireccionCliente()))

             self.tblClientes.setItem(i, 5, QtWidgets.QTableWidgetItem(aCli.devolverCliente(i).getTelefonoCliente()))

    def limpiarControles(self):

        self.txtDni.clear()

        self.txtNombres.clear()

        self.txtApellidoPaterno.clear()

        self.txtApellidoMaterno.clear()

        self.txtDireccion.clear()

        self.txtTelefono.clear()

    def registrar(self):

        if self.valida() == "":

            objCli = Cliente(self.obtenerDni(), self.obtenerNombres(),

self.obtenerApellidoPaterno(),

self.obtenerApellidoMaterno(),

self.obtenerDireccion(), self.obtenerTelefono())

            dni = self.obtenerDni()

            if aCli.buscarCliente(dni) == -1:

                aCli.adicionaCliente(objCli)

                aCli.grabar()

                self.limpiarControles()

                self.listar()

            else:

                QtWidgets.QMessageBox.information(self, "Registrar Cliente", "El DNI ingesado ya  existe...!!!", QtWidgets.QMessageBox.Ok)

        else:

            QtWidgets.QMessageBox.information(self, "Registrar Cliente", "Error en " +

self.valida(), QtWidgets.QMessageBox.Ok)

    def consultar(self):

        self.limpiarTabla()

        if aCli.tamañoArregloCliente() == 0:

            QtWidgets.QMessageBox.information(self, "Consultar Cliente", "No existen clientes a

consultar...!!!", QtWidgets.QMessageBox.Ok)

        else:

            dni, \_ = QtWidgets.QInputDialog.getText(self, "Consultar Cliente", "Ingrese

 el DNI a consultar")

            pos = aCli.buscarCliente(dni)

            if pos == -1:

                QtWidgets.QMessageBox.information(self, "Consultar Cliente", "El DNI ingresado no existe...!!! ",

QtWidgets.QMessageBox.Ok)

            else:

                self.tblClientes.setRowCount(1)

                 self.tblClientes.setItem(0, 0, QtWidgets.QTableWidgetItem(aCli.devolverCliente(pos).getDniCliente()))

                 self.tblClientes.setItem(0, 1, QtWidgets.QTableWidgetItem(aCli.devolverCliente(pos).getNombresCliente()))

                 self.tblClientes.setItem(0, 2, QtWidgets.QTableWidgetItem(aCli.devolverCliente(pos).getApellidoPaternoCliente()))

                 self.tblClientes.setItem(0, 3, QtWidgets.QTableWidgetItem(aCli.devolverCliente(pos).getApellidoMaternoCliente()))

                 self.tblClientes.setItem(0, 4, QtWidgets.QTableWidgetItem(aCli.devolverCliente(pos).getDireccionCliente()))

                 self.tblClientes.setItem(0, 5, QtWidgets.QTableWidgetItem(aCli.devolverCliente(pos).getTelefonoCliente()))

    def eliminar(self):

        if aCli.tamañoArregloCliente() == 0:

            QtWidgets.QMessageBox.information(self, "Eliminar Cliente", "No existen clientes a eliminar...!!!",

QtWidgets.QMessageBox.Ok)

        else:

            fila = self.tblClientes.selectedItems()

            if fila:

                indiceFila = fila[0].row()

                dni = self.tblClientes.item(indiceFila, 0).text()

                pos = aCli.buscarCliente(dni)

                aCli.eliminarCliente(pos)

                aCli.grabar()

                self.limpiarTabla()

                self.listar()

            else:

                QtWidgets.QMessageBox.information(self, "Eliminar Cliente", "Debe seleccionar una fila...!!!",

QtWidgets.QMessageBox.Ok)

    def modificar(self):

        if aCli.tamañoArregloCliente() == 0:

            QtWidgets.QMessageBox.information(self, "Modificar Cliente", "No existen clientes a modificar...!!!",  QtWidgets.QMessageBox.Ok)

        else:

            dni, \_ = QtWidgets.QInputDialog.getText(self, "Modificar Cliente", "Ingrese el DNI a

 modificar")

            pos = aCli.buscarCliente(dni)

            if pos != -1:

                objCliente = aCli.devolverCliente(pos)

                 self.btnModificar.setVisible(False)

                 self.btnGrabar.setVisible(True)

                  self.txtDni.setText(objCliente.getDniCliente())

                 self.txtDni.setVisible(False)

                 self.lblDni.setVisible(False)

                 self.txtNombres.setText(objCliente.getNombresCliente())

                  self.txtApellidoPaterno.setText(objCliente.getApellidoPaternoCliente())

                 self.txtApellidoMaterno.setText(objCliente.getApellidoMaternoCliente())

                 self.txtDireccion.setText(objCliente.getDireccionCliente())

                 self.txtTelefono.setText(objCliente.getTelefonoCliente())

    def grabar(self):

        try:

            pos = aCli.buscarCliente(self.obtenerDni())

            objCliente = aCli.devolverCliente(pos)

            objCliente.setNombresCliente(self.obtenerNombres())

            objCliente.setApellidoPaternoCliente(self.obtenerApellidoPaterno())

            objCliente.setApellidoMaternoCliente(self.obtenerApellidoMaterno())

            objCliente.setDireccionCliente(self.obtenerDireccion())

            objCliente.setTelefonoCliente(self.obtenerTelefono())

            self.btnModificar.setVisible(True)

            self.btnGrabar.setVisible(False)

            aCli.grabar()

            self.limpiarTabla()

            self.limpiarControles()

            self.listar()

            self.txtDni.setVisible(True)

            self.lblDni.setVisible(True)

        except:

            QtWidgets.QMessageBox.information(self, "Modificar Cliente", "Error al modificar cliente...!!!",

QtWidgets.QMessageBox.Ok)

**ventanaDetalleVentas.py**

from PyQt5 import QtWidgets, uic

from PyQt5 import QtGui

from Controlador.arregloProductos import \*

from Controlador.arregloClientes import \*

from Controlador.arregloDetalleVenta import \*

from Controlador.arregloFactura import \*

from Vista.ventanaVentas import \*

class VentanaDetalleVentas(QtWidgets.QMainWindow):

    def \_\_init\_\_(self,parent = None):

        super(VentanaDetalleVentas,self).\_\_init\_\_(parent)

        self.setWindowIcon(QtGui.QIcon("UI/imagenes/venta.png"))

        uic.loadUi("UI/ventanaDetalleVentas.ui", self)

        self.btnMostrar.clicked.connect(self.listar)

    def listar(self):

        aDetVent = ArregloDetalleVenta()

        self.tblDetalleVenta.setRowCount(aDetVent.tamañoDetalleVenta())

        self.tblDetalleVenta.setColumnCount(5)

        self.tblDetalleVenta.verticalHeader().setVisible(False)

        for i in range(0, aDetVent.tamañoDetalleVenta()):

            self.tblDetalleVenta.setItem(i, 0, QtWidgets.QTableWidgetItem(aDetVent.devolverDetalleVenta(i).getNDocumentoVenta()))

             self.tblDetalleVenta.setItem(i, 1, QtWidgets.QTableWidgetItem(str(aDetVent.devolverDetalleVenta(i).getNItem())))

             self.tblDetalleVenta.setItem(i, 2, QtWidgets.QTableWidgetItem(aDetVent.devolverDetalleVenta(i).getCodigoProducto()))

             self.tblDetalleVenta.setItem(i, 3, QtWidgets.QTableWidgetItem(str(aDetVent.devolverDetalleVenta(i).getPrecioVenta())))

             self.tblDetalleVenta.setItem(i, 4, QtWidgets.QTableWidgetItem(str(aDetVent.devolverDetalleVenta(i).getCantidad())))

**ventanaFacturas.py**

from PyQt5 import QtWidgets, uic

from PyQt5 import QtGui

from Controlador.arregloProductos import \*

from Controlador.arregloClientes import \*

from Controlador.arregloDetalleVenta import \*

from Controlador.arregloFactura import \*

from Vista.ventanaVentas import \*

class VentanaFacturas(QtWidgets.QMainWindow):

    def \_\_init\_\_(self,parent = None):

        super(VentanaFacturas,self).\_\_init\_\_(parent)

        self.setWindowIcon(QtGui.QIcon("UI/imagenes/venta.png"))

        uic.loadUi("UI/ventanaFactura.ui", self)

        self.btnMostrar.clicked.connect(self.listar)

    def listar(self):

        aFac = ArregloFactura()

        self.tblFactura.setRowCount(aFac.tamañoFactura())

        self.tblFactura.setColumnCount(6)

        self.tblFactura.verticalHeader().setVisible(False)

        for i in range(0, aFac.tamañoFactura()):

            self.tblFactura.setItem(i, 0, QtWidgets.QTableWidgetItem(aFac.devolverFactura(i).getNDocumentoVenta()))

             self.tblFactura.setItem(i, 1, QtWidgets.QTableWidgetItem(aFac.devolverFactura(i).getFecha()))

             self.tblFactura.setItem(i, 2, QtWidgets.QTableWidgetItem(aFac.devolverFactura(i).getIgv()))

             self.tblFactura.setItem(i, 3, QtWidgets.QTableWidgetItem(aFac.devolverFactura(i).getTotalPrecioVenta()))

             self.tblFactura.setItem(i, 4, QtWidgets.QTableWidgetItem(aFac.devolverFactura(i).getRuc()))

             self.tblFactura.setItem(i, 5, QtWidgets.QTableWidgetItem(aFac.devolverFactura(i).getEstado()))

**ventanaPrincipal.py**

from PyQt5 import QtWidgets, uic

from PyQt5 import QtGui

from Controlador.arregloProductos import \*

from Controlador.arregloClientes import \*

from Controlador.arregloDetalleVenta import \*

from Controlador.arregloFactura import \*

from Vista.ventanaClientes import \*

from Vista.ventanaDetalleVentas import \*

from Vista.ventanaFacturas import \*

from Vista.ventanaProductos import \*

from Vista.ventanaVentas import \*

class VentanaPrincipal(QtWidgets.QMainWindow):

    def \_\_init\_\_(self,parent = None):

        super(VentanaPrincipal,self).\_\_init\_\_(parent)

        self.setWindowIcon(QtGui.QIcon("UI/imagenes/venta.png"))

        uic.loadUi("UI/ventanaPrincipal.ui",self)

        self.btnProductos.clicked.connect(self.abrirVentanaProductos)

        self.btnClientes.clicked.connect(self.abrirVentanaClientes)

        self.btnVender.clicked.connect(self.abrirVentanaVentas)

        self.btnDetalleVentas.clicked.connect(self.abrirVentanaDetalleVentas)

        self.btnVentas.clicked.connect(self.abrirVentanaFacturas)

        self.btnSalir.clicked.connect(self.cerrar)

    def abrirVentanaProductos(self):

        vproductos = VentanaProductos(self)

        vproductos.show()

    def abrirVentanaClientes(self):

        vclientes = VentanaClientes(self)

        vclientes.show()

    def abrirVentanaVentas(self):

        vVentas = VentanaVentas(self)

        vVentas.show()

    def abrirVentanaDetalleVentas(self):

        vDetalleVentas = VentanaDetalleVentas(self)

        vDetalleVentas.show()

    def abrirVentanaFacturas(self):

        vFacturas = VentanaFacturas(self)

        vFacturas.show()

    def cerrar(self):

        self.close()

**ventanaProductos.py**

from PyQt5 import QtWidgets, uic

from PyQt5 import QtGui

from Controlador.arregloProductos import \*

from Controlador.arregloClientes import \*

from Controlador.arregloDetalleVenta import \*

from Controlador.arregloFactura import \*

aPro = ArregloProductos()

class VentanaProductos(QtWidgets.QMainWindow):

    def \_\_init\_\_(self,parent = None):

        super(VentanaProductos,self).\_\_init\_\_(parent)

        self.setWindowIcon(QtGui.QIcon("UI/imagenes/venta.png"))

        uic.loadUi("UI/ventanaProductos.ui",self)

        self.btnRegistrar.clicked.connect(self.registrar)

        self.btnConsultar.clicked.connect(self.consultar)

        self.btnListar.clicked.connect(self.listar)

        self.btnEliminar.clicked.connect(self.eliminar)

        self.btnModificar.clicked.connect(self.modificar)

        self.btnGrabar.clicked.connect(self.grabar)

        self.listar()

        self.show()

    def obtenerCodigo(self):

        return self.txtCodigo.text()

    def obtenerNombre(self):

        return self.txtNombre.text()

    def obtenerDescripcion(self):

        return self.txtDescripcion.text()

    def obtenerMinimo(self):

        return self.txtStockMinimo.text()

    def obtenerActual(self):

        return self.txtStockActual.text()

    def obtenerCosto(self):

        return self.txtPrecioCosto.text()

    def obtenerPrecio(self):

        return self.txtPrecioVenta.text()

    def obtenerProveedor(self):

        return self.cboProveedor.currentText()

    def obtenerAlmacen(self):

        return self.cboAlmacen.currentText()

    def limpiarTabla(self):

        self.tblProductos.clearContents()

        self.tblProductos.setRowCount(0)

    def valida(self):

        if self.txtCodigo.text() == "":

            self.txtCodigo.setFocus()

            return "Código del producto...!!!"

        elif self.txtNombre.text() == "":

            self.txtNombre.setFocus()

            return "Nombre del producto...!!!"

        elif self.txtDescripcion.text() == "":

            self.txtDescripcion.setFocus()

            return "Descripción del producto...!!!"

        elif self.txtStockMinimo.text() == "":

            self.txtStockMinimo.setFocus()

            return "Stock mínimo del producto...!!!"

        elif self.txtStockActual.text() == "":

            self.txtStockActual.setFocus()

            return "Stock máximo del producto...!!!"

        elif self.txtPrecioCosto.text() == "":

            self.txtPrecioCosto.setFocus()

            return "Costo del producto...!!!"

        elif self.txtPrecioVenta.text() == "":

            self.txtPrecioVenta.setFocus()

            return "Precio del producto...!!!"

        elif self.cboProveedor.currentText() == "Seleccionar Proveedor":

            self.cboProveedor.setCurrentIndex(0)

            return "Proveedor...!!!"

        elif self.cboAlmacen.currentText() == "Seleccionar Almacén":

            self.cboAlmacen.setCurrentIndex(0)

            return "Almacén...!!!"

        else:

            return ""

    def listar(self):

        self.tblProductos.setRowCount(aPro.tamañoArregloProducto())

        self.tblProductos.setColumnCount(9)

        self.tblProductos.verticalHeader().setVisible(False)

        for i in range(0, aPro.tamañoArregloProducto()):

            self.tblProductos.setItem(i, 0, QtWidgets.QTableWidgetItem(aPro.devolverProducto(i).getCodigo()))

             self.tblProductos.setItem(i, 1, QtWidgets.QTableWidgetItem(aPro.devolverProducto(i).getNombre()))

             self.tblProductos.setItem(i, 2, QtWidgets.QTableWidgetItem(aPro.devolverProducto(i).getDescripcion()))

             self.tblProductos.setItem(i, 3, QtWidgets.QTableWidgetItem(aPro.devolverProducto(i).getStockMinimo()))

             self.tblProductos.setItem(i, 4, QtWidgets.QTableWidgetItem(aPro.devolverProducto(i).getStockActual()))

             self.tblProductos.setItem(i, 5, QtWidgets.QTableWidgetItem(aPro.devolverProducto(i).getPrecioCosto()))

             self.tblProductos.setItem(i, 6, QtWidgets.QTableWidgetItem(aPro.devolverProducto(i).getPrecioVenta()))

             self.tblProductos.setItem(i, 7, QtWidgets.QTableWidgetItem(aPro.devolverProducto(i).getProveedor()))

             self.tblProductos.setItem(i, 8, QtWidgets.QTableWidgetItem(aPro.devolverProducto(i).getAlmacen()))

    def limpiarControles(self):

        self.txtCodigo.clear()

        self.txtNombre.clear()

        self.txtDescripcion.clear()

        self.txtStockMinimo.clear()

        self.txtStockActual.clear()

        self.txtPrecioCosto.clear()

        self.txtPrecioVenta.clear()

        self.cboProveedor.setCurrentIndex(0)

        self.cboAlmacen.setCurrentIndex(0)

    def registrar(self):

        if self.valida() == "":

             objPro = Producto(self.obtenerCodigo(), self.obtenerNombre(), self.obtenerDescripcion(),

self.obtenerMinimo(), self.obtenerActual(), self.obtenerCosto(),

self.obtenerPrecio(), self.obtenerProveedor(), self.obtenerAlmacen())

            codigo = self.obtenerCodigo()

            if aPro.buscarProducto(codigo) == -1:

                aPro.adicionaProducto(objPro)

                aPro.grabar()

                self.limpiarControles()

                self.listar()

            else:

                QtWidgets.QMessageBox.information(self, "Registrar Producto", "El código ingesado ya existe...!!!",  QtWidgets.QMessageBox.Ok)

        else:

            QtWidgets.QMessageBox.information(self, "Registrar Producto", "Error en " +  self.valida(), QtWidgets.QMessageBox.Ok)

    def consultar(self):

        self.limpiarTabla()

        if aPro.tamañoArregloProducto() == 0:

            QtWidgets.QMessageBox.information(self, "Consultar Producto", "No existen productos a consultar...!!!",

QtWidgets.QMessageBox.Ok)

        else:

            codigo, \_ = QtWidgets.QInputDialog.getText(self, "Consultar Producto", "Ingrese

el código a consultar")

            pos = aPro.buscarProducto(codigo)

            if pos == -1:

                QtWidgets.QMessageBox.information(self, "Consultar Producto", "El código ingresado no  existe...!!! ", QtWidgets.QMessageBox.Ok)

            else:

                self.tblProductos.setRowCount(1)

                self.tblProductos.setItem(0, 0, QtWidgets.QTableWidgetItem(aPro.devolverProducto(pos).getCodigo()))

                self.tblProductos.setItem(0, 1, QtWidgets.QTableWidgetItem(aPro.devolverProducto(pos).getNombre()))

                self.tblProductos.setItem(0, 2, QtWidgets.QTableWidgetItem(aPro.devolverProducto(pos).getDescripcion()))

                self.tblProductos.setItem(0, 3, QtWidgets.QTableWidgetItem(aPro.devolverProducto(pos).getStockMinimo()))

                self.tblProductos.setItem(0, 4, QtWidgets.QTableWidgetItem(aPro.devolverProducto(pos).getStockActual()))

                self.tblProductos.setItem(0, 5, QtWidgets.QTableWidgetItem(aPro.devolverProducto(pos).getPrecioCosto()))

                self.tblProductos.setItem(0, 6, QtWidgets.QTableWidgetItem(aPro.devolverProducto(pos).getPrecioVenta()))

                self.tblProductos.setItem(0, 7, QtWidgets.QTableWidgetItem(aPro.devolverProducto(pos).getProveedor()))

                self.tblProductos.setItem(0, 8, QtWidgets.QTableWidgetItem(aPro.devolverProducto(pos).getAlmacen()))

    def eliminar(self):

        if aPro.tamañoArregloProducto() == 0:

            QtWidgets.QMessageBox.information(self, "Eliminar Producto", "No existen productos a  eliminar...!!!", QtWidgets.QMessageBox.Ok)

        else:

            fila = self.tblProductos.selectedItems()

            if fila:

                indiceFila = fila[0].row()

                codigo = self.tblProductos.item(indiceFila, 0).text()

                pos = aPro.buscarProducto(codigo)

                aPro.eliminarProducto(pos)

                aPro.grabar()

                self.limpiarTabla()

                self.listar()

            else:

                QtWidgets.QMessageBox.information(self, "Eliminar Producto", "Debe seleccionar una

fila...!!!", QtWidgets.QMessageBox.Ok)

    def modificar(self):

        if aPro.tamañoArregloProducto() == 0:

            QtWidgets.QMessageBox.information(self, "Modificar Producto", "No existen productos

a modificar...!!!", QtWidgets.QMessageBox.Ok)

        else:

            codigo, \_ = QtWidgets.QInputDialog.getText(self, "Modificar Producto", "Ingrese

el código a modificar")

            pos = aPro.buscarProducto(codigo)

            if pos != -1:

                objProducto = aPro.devolverProducto(pos)

                self.btnModificar.setVisible(False)

                self.btnGrabar.setVisible(True)

                self.txtCodigo.setText(objProducto.getCodigo())

                self.txtCodigo.setVisible(False)

                self.lblCodigo.setVisible(False)

                self.txtNombre.setText(objProducto.getNombre())

                self.txtDescripcion.setText(objProducto.getDescripcion())

                self.txtStockMinimo.setText(objProducto.getStockMinimo())

                self.txtStockActual.setText(objProducto.getStockActual())

                self.txtPrecioCosto.setText(objProducto.getPrecioCosto())

                self.txtPrecioVenta.setText(objProducto.getPrecioVenta())

                self.cboProveedor.setCurrentText(objProducto.getProveedor())

                self.cboAlmacen.setCurrentText(objProducto.getAlmacen())

    def grabar(self):

        try:

            pos = aPro.buscarProducto(self.obtenerCodigo())

            objProducto = aPro.devolverProducto(pos)

            objProducto.setNombre(self.obtenerNombre())

            objProducto.setDescripcion(self.obtenerDescripcion())

            objProducto.setStockMinimo(self.obtenerMinimo())

            objProducto.setStockActual(self.obtenerActual())

            objProducto.setPrecioCosto(self.obtenerCosto())

            objProducto.setPrecioVenta(self.obtenerPrecio())

            objProducto.setProveedor(self.obtenerProveedor())

            objProducto.setAlmacen(self.obtenerAlmacen())

            self.btnModificar.setVisible(True)

            self.btnGrabar.setVisible(False)

            self.limpiarTabla()

            self.limpiarControles()

            aPro.grabar()

            self.listar()

            self.txtCodigo.setVisible(True)

            self.lblCodigo.setVisible(True)

        except:

            QtWidgets.QMessageBox.information(self, "Modificar Producto", "Error al modificar  producto...!!!", QtWidgets.QMessageBox.Ok)

**ventanaVentas.py**

from datetime import date

from PyQt5 import QtWidgets, uic

from PyQt5 import QtGui

from Controlador.arregloProductos import \*

from Controlador.arregloClientes import \*

from Controlador.arregloDetalleVenta import \*

from Controlador.arregloFactura import \*

from Vista.ventanaClientes import \*

from Vista.ventanaProductos import \*

aDetVent = ArregloDetalleVenta()

aFac = ArregloFactura()

lista = []

class VentanaVentas(QtWidgets.QMainWindow):

    def \_\_init\_\_(self,parent = None):

        super(VentanaVentas,self).\_\_init\_\_(parent)

        self.setWindowIcon(QtGui.QIcon("UI/imagenes/venta.png"))

        uic.loadUi("UI/ventanaVentas.ui", self)

        self.btnBuscarCliente.clicked.connect(self.buscarCliente)

        self.btnBuscarProducto.clicked.connect(self.buscarProducto)

        self.btnAgregar.clicked.connect(self.agregar)

        self.btnGenerar.clicked.connect(self.generarVenta)

        self.btnImprimirFactura.clicked.connect(self.imprimirFactura)

        self.btnCerrar.clicked.connect(self.cerrarVentana)

        self.calcularFecha()

        self.generarNumeroDocumento()

        self.item = 0

        self.stock\_actual\_temporal = {}

    def buscarCliente(self):

        codigoCliente = self.txtCodigoCliente.text()

        if self.txtCodigoCliente.text() == "":

            QtWidgets.QMessageBox.information(self, "Código Cliente", "Debe ingresar el código del

cliente...!!!", QtWidgets.QMessageBox.Ok)

        else:

            pos = aCli.buscarCliente(codigoCliente)

            objCliente = aCli.devolverCliente(pos)

            if pos == -1:

                QtWidgets.QMessageBox.information(self, "Código Cliente", "Cliente no registrado...!!!",  QtWidgets.QMessageBox.Ok)

            else:

                self.txtNombres.setText(objCliente.getNombresCliente() + " " + objCliente.getApellidoPaternoCliente()

+ " " + objCliente.getApellidoMaternoCliente())

    def buscarProducto(self):

        codigoProducto = self.txtCodigoProducto.text()

        if self.txtCodigoProducto.text() == "":

            QtWidgets.QMessageBox.information(self, "Código Producto", "Debe ingresar el código del

 producto...!!!", QtWidgets.QMessageBox.Ok)

        else:

            pos = aPro.buscarProducto(codigoProducto)

            objProducto = aPro.devolverProducto(pos)

            if pos == -1:

                QtWidgets.QMessageBox.information(self, "Código Producto", "Producto no registrado...!!!",  QtWidgets.QMessageBox.Ok)

            else:

                stock\_temporal = self.obtenerStockActualTemporal(codigoProducto)

                if stock\_temporal == -1:  # ocurre cuando no existe el producto en el diccionario

                    self.stock\_actual\_temporal[self.txtCodigoProducto.text()] = int(objProducto.getStockActual())

                    stock\_temporal = self.obtenerStockActualTemporal(codigoProducto)

                self.txtDescripcion.setText(objProducto.getDescripcion())

                self.txtStock.setText(str(stock\_temporal))

                self.txtPrecio.setText(objProducto.getPrecioVenta())

    def obtenerStockActualTemporal(self, codigoProducto):

        stockEncontrado = -1 # variable auxiliar que contiene de manera predeterminada -1, indicando que no se encontró el stock

        if len(self.stock\_actual\_temporal) != 0:

            for llave in self.stock\_actual\_temporal:

                if llave == codigoProducto:

                    stockEncontrado = self.stock\_actual\_temporal[llave]

                    return stockEncontrado

        return stockEncontrado

    def actualizarStockActualTemporal(self, codigoProducto, cantidad):

        for llave in self.stock\_actual\_temporal:

            if llave == codigoProducto:

                self.stock\_actual\_temporal[llave]= self.stock\_actual\_temporal[llave] - cantidad

    def obtenerNumeroDocumento(self):

        return self.txtNumeroDocumento.text()

    def obtenerCodigo(self):

        return self.txtCodigoProducto.text()

    def obtenerDescripcion(self):

        return self.txtDescripcion.text()

    def obtenerPrecio(self):

        return float(self.txtPrecio.text())

    def obtenerCantidad(self):

        return int(self.txtCantidad.text())

    def obtenerItem(self):

        self.item = self.item + 1

        return self.item

    def obtenerFecha(self):

        return self.txtFecha.text()

    def calcularFecha(self):

        self.fecha = date.today()

        dia = self.fecha.day

        mes = self.fecha.month

        año = self.fecha.year

        self.txtFecha.setText(str(dia) + "/" + str(mes) + "/" + str(año))

    def imprimirFactura(self):

        try:

            if self.txtNumeroDocumento.text() == "":

                QtWidgets.QMessageBox.information(self, "Imprimir Factura", "No se ha realizado ninguna

venta...!!!", QtWidgets.QMessageBox.Ok)

            else:

                numeroFactura = self.txtNumeroDocumento.text()

                pos = aFac.buscarFactura(numeroFactura)

                objFact = aFac.devolverFactura(pos)

                codigoCliente = self.txtCodigoCliente.text()

                pos = aCli.buscarCliente(codigoCliente)

                objCliente = aCli.devolverCliente(pos)

                objFact.imprimirDocumentoVenta(objCliente)

                aDetVent.imprimirDetalleVenta(numeroFactura, aPro)

                QtWidgets.QMessageBox.information(self, "Imprimir Factura", "La factura se ha impreso correctamente...!!!",

QtWidgets.QMessageBox.Ok)

                self.close()

        except:

            QtWidgets.QMessageBox.information(self, "Imprimir Factura", "No se ha realizado ninguna  venta...!!!", QtWidgets.QMessageBox.Ok)

    def agregar(self):

        try:

            if int(self.txtCantidad.text()) > int(self.txtStock.text()):

                QtWidgets.QMessageBox.information(self, "Venta", "No se puede vender esa cantidad...!!!",

QtWidgets.QMessageBox.Ok)

            elif int(self.txtCantidad.text()) <= 0:

                QtWidgets.QMessageBox.information(self, "Venta", "La cantidad ingresada es incorrecta...!!!",

QtWidgets.QMessageBox.Ok)

            else:

                lista.append((self.obtenerItem(), self.obtenerCodigo(), self.obtenerDescripcion(),  self.obtenerPrecio(), self.obtenerCantidad(),

self.obtenerPrecio()\*self.obtenerCantidad()))

                 self.actualizarStockActualTemporal(self.obtenerCodigo(),self.obtenerCantidad())

                self.mostrarDetalle()

                self.calcularPago()

        except:

            QtWidgets.QMessageBox.information(self, "Agregar Detalle", "No se ha completado los detalles del

producto...!!!", QtWidgets.QMessageBox.Ok)

            self.item = self.item - 1

    def mostrarDetalle(self):

        self.tblDetalle.setRowCount(len(lista))

        self.tblDetalle.setColumnCount(6)

        self.tblDetalle.verticalHeader().setVisible(False)

        for i in range(len(lista)):

            self.tblDetalle.setItem(i, 0, QtWidgets.QTableWidgetItem(str(lista[i][0])))

             self.tblDetalle.setItem(i, 1, QtWidgets.QTableWidgetItem(str(lista[i][1])))

             self.tblDetalle.setItem(i, 2, QtWidgets.QTableWidgetItem(str(lista[i][2])))

             self.tblDetalle.setItem(i, 3, QtWidgets.QTableWidgetItem(str(lista[i][3])))

             self.tblDetalle.setItem(i, 4, QtWidgets.QTableWidgetItem(str(lista[i][4])))

             self.tblDetalle.setItem(i, 5, QtWidgets.QTableWidgetItem(str(lista[i][5])))

        self.limpiarControles()

    def limpiarControles(self):

        self.txtCodigoProducto.clear()

        self.txtDescripcion.clear()

        self.txtStock.clear()

        self.txtPrecio.clear()

        self.txtCantidad.clear()

    def limpiarControlesTotal(self):

        self.txtNumeroDocumento.clear()

        self.txtCodigoCliente.clear()

        self.txtNombres.clear()

        self.txtCodigoProducto.clear()

        self.txtDescripcion.clear()

        self.txtStock.clear()

        self.txtPrecio.clear()

        self.txtCantidad.clear()

        self.txtTotalPagar.clear()

        self.tblDetalle.clearContents()

        self.tblDetalle.setRowCount(0)

    def calcularPago(self):

        self.subtotal\_pagar = 0

        for i in range(len(lista)):

            self.subTotal = lista[i][5]

            self.subtotal\_pagar = self.subtotal\_pagar + float(self.subTotal)

            self.igv = 0.18 \* self.subtotal\_pagar

            self.total\_pagar = self.subtotal\_pagar + self.igv

        self.txtSubTotal.setText("S/. " + str(round(self.subtotal\_pagar, 2)))

        self.txtIgv.setText("S/. " + str(round(self.igv, 2)))

        self.txtTotalPagar.setText("S/. " + str(round(self.total\_pagar, 2)))

    def generarVenta(self):

        if self.tblDetalle.rowCount() == 0:

             QtWidgets.QMessageBox.information(self, "Venta", "No se ha agregado ningún

detalle...!!!", QtWidgets.QMessageBox.Ok)

        else:

            objFact = Factura(self.obtenerNumeroDocumento(), self.obtenerFecha(), self.txtIgv.text(),

self.txtTotalPagar.text(), self.txtCodigoCliente.text())

            numeroFactura = self.obtenerNumeroDocumento()

            if aFac.buscarFactura(numeroFactura) == -1:

                aFac.adicionaFactura(objFact)

                self.actualizarStock()

                self.guardarDetalleVenta()

                aFac.grabar()

                aPro.grabar()

                QtWidgets.QMessageBox.information(self, "Venta", "Se realizó la venta correctamente...!!!",

QtWidgets.QMessageBox.Ok)

                lista.clear()

            else:

                QtWidgets.QMessageBox.information(self, "Venta", "El número de factura ingesado ya

existe...!!!", QtWidgets.QMessageBox.Ok)

    def actualizarStock(self):

        for i in range(self.tblDetalle.rowCount()):

            codigoProducto = self.tblDetalle.item(i, 1).text()

            cantidad = int(self.tblDetalle.item(i, 4).text())

            pos = aPro.buscarProducto(codigoProducto)

            objProducto = aPro.devolverProducto(pos)

            stock\_actual = int(objProducto.getStockActual()) - cantidad

            aPro.actualizarStock(stock\_actual, codigoProducto)

    def guardarDetalleVenta(self):

        for i in range(len(lista)):

            objDetVent = DetalleVenta(self.obtenerNumeroDocumento(), lista[i][0], lista[i][1],

lista[i][3], lista[i][4])

            aDetVent.adicionaDetalleVenta(objDetVent)

            aDetVent.grabar()

    def generarNumeroDocumento(self):

        numeroDocunento = aFac.nroSerie()

        if numeroDocunento == 0:

            self.txtNumeroDocumento.setText("000001")

        else:

            self.incremento = numeroDocunento

            self.incremento = self.incremento + 1

            self.txtNumeroDocumento.setText("00000" + str(self.incremento))

    def cerrarVentana(self):

        lista.clear()

        self.close()

**main.py**

import sys

from PyQt5 import QtWidgets

from Vista.login import Login

if \_\_name\_\_ == '\_\_main\_\_':

    app = QtWidgets.QApplication(sys.argv)

    window = Login()

    app.exec()