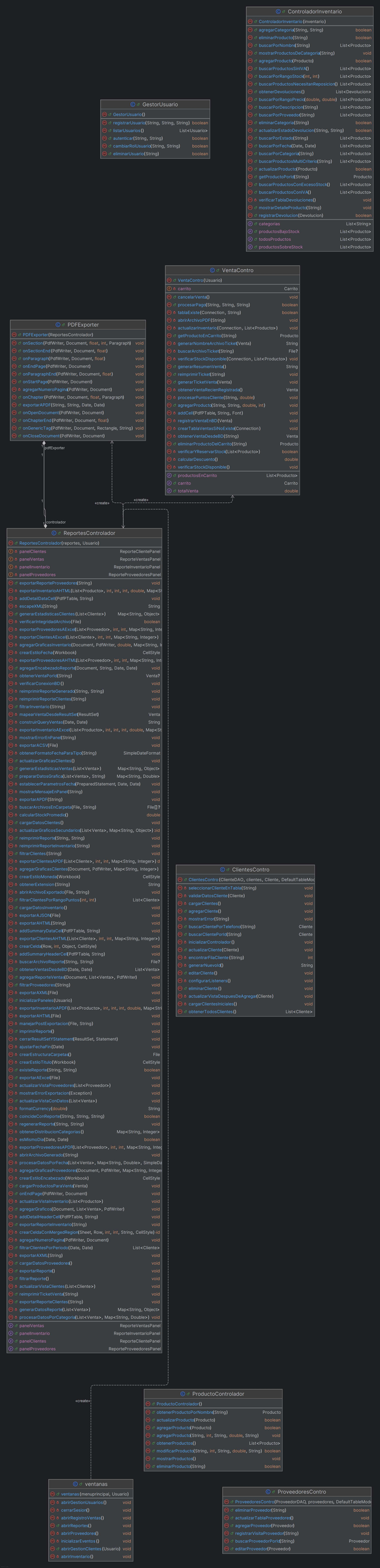
ControladorInventario
ControladorInventario(Inventario)
agregarCategoria(String, String) boolean
eliminarProducto(String) boolean
buscarPorNombre(String) List<Producto>
mostrarProductosDeCategoria(String) void
agregarProducto(Producto) boolean
buscarProductosSinIVA() List<Producto>
buscarPorRangoStock(int, int) List<Producto>
buscarProductosNecesitanReposicion() List<Producto>
obtenerDevoluciones() List<Devolucion>
buscarPorRangoPrecio(double, double) List<Producto>
buscarPorDescripcion(String) List<Producto>
buscarPorProveedor(String) List<Producto>
eliminarCategoria(String) boolean
actualizarEstadoDevolucion(String, String) boolean
buscarPorEstado(String) List<Producto>
buscarPorFecha(Date, Date) List<Producto>
buscarPorCategoria(String) List<Producto>
buscarProductosMultiCriterio(String) List<Producto>
actualizarProducto(Producto) boolean
getProductoPorId(String) Producto
buscarProductosConExcesoStock() List<Producto>
buscarProductosConIVA() List<Producto>
verificarTablaDevoluciones() void
mostrarDetalleProducto(String) void
registrarDevolucion(Devolucion) boolean
categorias List<String>
productosBajoStock List<Producto>
todosProductos List<Producto>
productosSobreStock List<Producto>
GestorUsuario
GestorUsuario()
registrarUsuario(String, String, String) boolean
listarUsuarios() List<Usuario>
autenticar(String, String) boolean
cambiarRolUsuario(String, String) boolean
eliminarUsuario(String) boolean
VentaContro
VentaContro(Usuario)
carrito Carrito
cancelarVenta() void
procesarPago(String, String, String) boolean
tablaExiste(Connection, String) boolean
abrirArchivoPDF(String) void
actualizarInventario(Connection, List<Producto>) void
getProductoEnCarrito(String) Producto
generarNombreArchivoTicket(Venta) String
buscarArchivoTicket(String) File?
verificarStockDisponible(Connection, List<Producto>) void
generarResumenVenta() String
reimprimirTicket(String) void
generarTicketVenta(Venta) void
obtenerVentaRecienRegistrada() Venta
procesarPuntosCliente(String, double) void
agregarProducto(String, String, double, int) void
addCell(PdfPTable, String, Font) void
registrarVentaEnBD(Venta) void
crearTablaVentasSiNoExiste(Connection) void
obtenerVentaDesdeBD(String) Venta
eliminarProductoDelCarrito(String) Producto
verificarYReservarStock(List<Producto>) boolean
calcularDescuento() double
verificarStockDisponible() void
productosEnCarrito List<Producto>
carrito Carrito
totalVenta double
PDFExporter
PDFExporter(ReportesControlador)
onSection(PdfWriter, Document, float, int, Paragraph) void
onSectionEnd(PdfWriter, Document, float) void
onParagraph(PdfWriter, Document, float) void
onEndPage(PdfWriter, Document) void
onParagraphEnd(PdfWriter, Document, float) void
onStartPage(PdfWriter, Document) void
agregarNumeroPagina(PdfWriter, Document) void
onChapter(PdfWriter, Document, float, Paragraph) void
exportarAPDF(String, String, Date, Date) void
onOpenDocument(PdfWriter, Document) void
onChapterEnd(PdfWriter, Document, float) void
onGenericTag(PdfWriter, Document, Rectangle, String) void
onCloseDocument(PdfWriter, Document) void
pdfExporter
controlador
«create»
«create»
ReportesControlador
ReportesControlador(reportes, Usuario)
panelClientes ReporteClientePanel
panelVentas ReporteVentasPanel
panelInventario ReporteInventarioPanel
panelProveedores ReporteProveedoresPanel
exportarReporteProveedores(String) void
exportarInventarioAHTML(List<Producto>, int, int, int, double, Map<St
addDetailDataCell(PdfPTable, String) void
escapeXML(String) String
generarEstadisticasClientes(List<Cliente>) Map<String, Object>
verificarIntegridadArchivo(File) boolean
exportarProveedoresAExcel(List<Proveedor>, int, int, Map<String, Inte
exportarClientesAExcel(List<Cliente>, int, int, Map<String, Integer>)
agregarGraficasInventario(Document, PdfWriter, double, Map<String, I
crearEstiloFecha(Workbook) CellStyle
exportarProveedoresAHTML(List<Proveedor>, int, int, Map<String, Inte
agregarEncabezadoReporte(Document, String, Date, Date) void
obtenerVentaPorId(String) Venta?
verificarConexionBD() void
reimprimirReporteGenerado(String, String) void
reimprimirReporteClientes(String) void
filtrarInventario(String) void
mapearVentaDesdeResultSet(ResultSet) Venta
construirQueryVentas(Date, Date) String
exportarInventarioAExcel(List<Producto>, int, int, int, double, Map<St
mostrarErrorEnPanel(String) void
exportarACSV(File) void
obtenerFormatoFechaParaTipo(String) SimpleDateFormat
actualizarGraficasClientes() void
generarEstadisticasVentas(List<Venta>) Map<String, Object>
prepararDatosGrafica(List<Venta>, String) Map<String, Double>
establecerParametrosFecha(PreparedStatement, Date, Date) void
mostrarMensajeEnPanel(String) void
exportarAPDF(String) void
buscarArchivosEnCarpeta(File, String) File[]?
calcularStockPromedio() double
cargarDatosClientes() void
actualizarGraficosSecundarios(List<Venta>, Map<String, Object>) void
reimprimirReporte(String, String) void
reimprimirReporteInventario(String) void
filtrarClientes(String) void
exportarClientesAPDF(List<Cliente>, int, int, Map<String, Integer>) d
agregarGraficasClientes(Document, PdfWriter, Map<String, Integer>)
crearEstiloMoneda(Workbook) CellStyle
obtenerExtension(String) String
abrirArchivoExportado(File, String) void
filtrarClientesPorRangoPuntos(int, int) List<Cliente>
cargarDatosInventario() void
exportarAJSON(File) void
exportarAHTML(String) void
addSummaryDataCell(PdfPTable, String) void
exportarClientesAHTML(List<Cliente>, int, int, Map<String, Integer>)
crearCelda(Row, int, Object, CellStyle) void
addSummaryHeaderCell(PdfPTable, String) void
buscarArchivoReporte(String, String, String) File?
obtenerVentasDesdeBD(Date, Date) List<Venta>
agregarReporteVenta(Document, List<Venta>, PdfWriter) void
filtrarProveedores(String) void
exportarAXML(File) void
inicializarPaneles(Usuario) void
exportarInventarioAPDF(List<Producto>, int, int, int, double, Map<Stri
exportarAHTML(File) void
manejarPostExportacion(File, String) void
imprimirReporte() void
cerrarResultSetYStatement(ResultSet, Statement) void
ajustarFechaFin(Date) void
crearEstructuraCarpetas() File
crearEstiloTitulo(Workbook) CellStyle
existeReporte(String, String) boolean
exportarAExcel(File) void
actualizarVistaProveedores(List<Proveedor>) void
mostrarErrorExportacion(Exception) void
actualizarVistaConDatos(List<Venta>) void
formatCurrency(double) String
coincideConReporte(String, String, String) boolean
regenerarReporte(String, String) void
obtenerDistribucionCategorias() Map<String, Integer>
esMismoDia(Date, Date) boolean
exportarProveedoresAPDF(List<Proveedor>, int, int, Map<String, Inte
abrirArchivoGenerado(String) void
procesarDatosPorFecha(List<Venta>, Map<String, Double>, SimpleDa
agregarGraficasProveedores(Document, PdfWriter, Map<String, Integer
crearEstiloEncabezado(Workbook) CellStyle
cargarProductosParaVenta(Venta) void
onEndPage(PdfWriter, Document) void
actualizarVistaInventario(List<Producto>) void
agregarGraficos(Document, List<Venta>, PdfWriter) void
addDetailHeaderCell(PdfPTable, String) void
exportarReporteInventario(String) void
crearCeldaConMergedRegion(Sheet, Row, int, int, String, CellStyle) vo
agregarNumeroPagina(PdfWriter, Document) void
filtrarClientesPorPeriodo(Date, Date) List<Cliente>
exportarAXML(String) void
cargarDatosProveedores() void
exportarReporte() void
filtrarReporte() void
actualizarVistaClientes(List<Cliente>) void
reimprimirTicketVenta(String) void
exportarReporteClientes(String) void
generarDatosReporte(List<Venta>) Map<String, Object>
procesarDatosPorCategoria(List<Venta>, Map<String, Double>) void
panelVentas ReporteVentasPanel
panelInventario ReporteInventarioPanel
panelClientes ReporteClientePanel
panelProveedores ReporteProveedoresPanel
ClientesContro
ClientesContro(ClienteDAO, clientes, Cliente, DefaultTableMod
seleccionarClienteEnTabla(String) void
validarDatosCliente(Cliente) void
cargarClientes() void
agregarCliente() void
mostrarError(String) void
buscarClientePorTelefono(String) Cliente
buscarClientePorId(String) Cliente
inicializarControlador() void
actualizarCliente(Cliente) void
encontrarFilaCliente(String) int
generarNuevoId() String
editarCliente() void
configurarListeners() void
eliminarCliente() void
actualizarVistaDespuesDeAgregar(Cliente) void
cargarClientesIniciales() void
obtenerTodosClientes() List<Cliente>
«create»
ProductoControlador
ProductoControlador()
obtenerProductoPorNombre(String) Producto
actualizarProducto(Producto) boolean
agregarProducto(Producto) boolean
agregarProducto(String, int, String, double, String) void
obtenerProductos() List<Producto>
modificarProducto(String, int, String, double, String) boolean
mostrarProductos() void
eliminarProducto(String) boolean
ventanas
ventanas(menuprincipal, Usuario)
abrirGestionUsuarios() void
cerrarSesion() void
abrirRegistroVentas() void
abrirReportes() void
abrirProveedores() void
inicializarEventos() void
abrirGestionClientes(Usuario) void
abrirInventario() void
ProveedoresContro
ProveedoresContro(ProveedorDAO, proveedores, DefaultTableMod
eliminarProveedor(String) boolean
actualizarTablaProveedores() void
agregarProveedor(Proveedor) boolean
registrarVistaProveedor(String) void
buscarProveedorPorId(String) Proveedor
editarProveedor(Proveedor) boolean