**ADOO — Sistema de Gestión de Ciber (Videojuegos El Profe 3.0)**

*Arquitectura WPF + MVVM • .NET 8 • EF Core + SQLite • Iterativo-Ágil*

# 1. Visión y objetivos

Visión: Administrar tiempos de consola, ventas de snacks/impresiones y cobros en un ciber café de forma rápida, confiable y con reportes diarios.

Objetivos: (1) Reducir tiempos de atención, (2) Evitar fugas de dinero, (3) Llevar historial por cliente, (4) Generar cortes diarios/semana.

# 2. Arquitectura WPF + MVVM (WPF/.NET 8)

Estilo: Arquitectura por capas con MVVM. Separación estricta de UI (View), lógica de presentación (ViewModel) y dominio/servicios (Model/Application).

## 2.1 Capas y proyectos

|  |  |  |
| --- | --- | --- |
| Capa/Proyecto | Responsabilidad | Tecnologías |
| AppJB.Ciber.Presentation (WPF) | Views XAML, estilos, recursos, navegación (Frame/Region), validaciones visuales. | WPF, DataBinding, ICommand, ValidationRules |
| AppJB.Ciber.Application | Servicios de aplicación, orquestación UC, pricing, timers, inventario, recibos. | C# .NET 8, DTOs, mapeo, AutoMapper (opcional) |
| AppJB.Ciber.Domain | Entidades, reglas de negocio, repositorios (interfaces), eventos dominio. | POCOs, Value Objects |
| AppJB.Ciber.Infrastructure | Persistencia y adapters a servicios externos (impresora). | EF Core (SQLite), Repos impl., Logging |
| AppJB.Ciber.Tests | Pruebas unitarias y de integración ligeras. | xUnit/NUnit, FluentAssertions |

## 2.2 Principales decisiones

• MVVM con ICommand (RelayCommand/CommunityToolkit.Mvvm) y mensajería ligera (WeakReferenceMessenger).

• Persistencia local con EF Core + SQLite; migraciones para versionar el esquema; patrón Repository + Unit of Work.

• DI con Microsoft.Extensions.DependencyInjection; configuración en App.xaml.cs.

• Navegación por regiones (Shell) o Frame + NavigationService; VM-First (IoC crea ViewModel y resuelve la View).

• Impresión de tickets: servicio de recibos (texto plano 58/80mm) y/o PDF.

## 2.3 Mapa de Views ↔ ViewModels

|  |  |
| --- | --- |
| View (XAML) | ViewModel (Responsabilidad) |
| Shell/MainWindow | MainViewModel: navegación, sesión del usuario, shell de módulos. |
| LoginView | LoginViewModel: autenticación local del operador. |
| DashboardView | DashboardViewModel: KPIs del día, botones rápidos. |
| ConsolasView | ConsolasViewModel: estado de consolas, iniciar/pausar/finalizar sesión, tarifas. |
| ClientesView | ClientesViewModel: CRUD clientes, historial. |
| SesionesView | SesionViewModel: detalle de sesión actual, extender, aplicar descuentos. |
| VentasView | VentasViewModel: venta rápida de snacks/impresiones, combos. |
| InventarioView | InventarioViewModel: existencias, mermas, alertas de bajo stock. |
| ReportesView | ReportesViewModel: corte diario/semana, exportación CSV/PDF. |
| ConfigView | ConfigViewModel: tarifas, categorías, impresora, IVA/propinas. |

## 2.4 Servicios de aplicación (interfaz)

|  |  |  |
| --- | --- | --- |
| Servicio | Funciones clave | Notas |
| ISessionService | Iniciar/Extender/Finalizar sesión; cálculo de costo por tiempo; pausa. | Timers de alta precisión, tolerancia a cierres inesperados. |
| IPricingService | Cálculo por minuto/paquetes; promociones/HH; redondeos. | Reglas configurables por consola/horario. |
| IClientService | CRUD cliente, ranking de frecuencia. | Búsqueda por teléfono; duplicados. |
| IInventoryService | Kardex simple; disminuir stock en ventas. | Alertas bajo stock. |
| ISalesService | Venta rápida; combos; totales y métodos de pago. | Efectivo/mixto; descuento % o monto. |
| IReceiptService | Imprimir ticket; nota simple; exportar PDF. | Plantillas texto; ancho 58/80mm. |
| IReportingService | Corte diario/semanal; CSV/PDF. | Totales por rubro y consola. |

## 2.5 Repositorios (Dominio/Infra)

|  |  |  |
| --- | --- | --- |
| Repositorio | Entidad | Operaciones |
| IClienteRepository | Cliente | Add/Update/Delete/FindById/FindByTelefono/List |
| IConsolaRepository | Consola | Add/Update/Delete/List/ByEstado |
| ISesionRepository | Sesion | Add/Update/Close/FindOpenByConsola/ListByFecha |
| IProductoRepository | Producto | Add/Update/Delete/List/Search |
| IVentaRepository | Venta/VentaItem | Add/ListByFecha/ByCliente/ByTipo |
| ITarifaRepository | Tarifa | List/ByConsola/SetDefault |

## 2.6 Entidades de dominio (POCO)

|  |  |  |
| --- | --- | --- |
| Entidad | Atributos clave | Notas/Reglas |
| Cliente | Id, Nombre, Telefono, Email?, FechaAlta, Activo | Teléfono único; historial de consumo. |
| Consola | Id, Nombre, Tipo (Xbox/PS/PC), Estado (Libre/Ocupada/Pausa), Notas | No iniciar nueva sesión si Ocupada. |
| Sesion | Id, ConsolaId, ClienteId?, Inicio, Fin?, Pausas[], TarifaId, Total | Costo = PricingService; impedir fin < inicio. |
| Tarifa | Id, Nombre, Modo (por minuto/paquete), PrecioUnidad, MinutosPaquete? | Reglas HH; redondeo al minuto. |
| Producto | Id, Nombre, Categoria, Precio, Stock, MinStock | Stock no negativo. |
| Venta | Id, Fecha, ClienteId?, Items[], Total, MetodoPago | Actualizar stock al confirmar. |
| VentaItem | Id, VentaId, ProductoId, Cantidad, PrecioUnit | Subtotal = Cantidad\*PrecioUnit. |

## 2.7 Navegación y comandos

• NavigationService en MainViewModel; comandos: IniciarSesionCommand, ExtenderSesionCommand, FinalizarSesionCommand, RegistrarVentaCommand, ImprimirTicketCommand, IrA<View>Command.

• Mensajería: publicación de eventos (SesionIniciada, SesionFinalizada, VentaConfirmada) para refrescar Dashboard/Reportes.

## 2.8 Validaciones y estados

• Validations via IDataErrorInfo/ValidationRules. Estados de consola: Libre → Ocupada → Pausa (opcional) → Ocupada → Libre.

# 3. Persistencia (EF Core + SQLite)

Modelo relacional mínimo y migraciones.

|  |  |  |  |
| --- | --- | --- | --- |
| Tabla | Campos | Claves/Índices | Relaciones |
| Clientes | Id PK, Nombre, Telefono UNIQUE, Email, FechaAlta, Activo | UX\_Telefono | - |
| Consolas | Id PK, Nombre, Tipo, Estado, Notas | - | - |
| Tarifas | Id PK, Nombre, Modo, PrecioUnidad, MinutosPaquete | - | - |
| Sesiones | Id PK, ConsolaId FK, ClienteId FK NULL, Inicio, Fin NULL, TarifaId FK, Total | IX\_ConsolaId, IX\_ClienteId | Consolas 1..\* Sesiones; Clientes 0..\* Sesiones |
| Productos | Id PK, Nombre, Categoria, Precio, Stock, MinStock | IX\_Nombre | - |
| Ventas | Id PK, Fecha, ClienteId FK NULL, Total, MetodoPago | IX\_Fecha | Clientes 0..\* Ventas |
| VentaItems | Id PK, VentaId FK, ProductoId FK, Cantidad, PrecioUnit | - | Ventas 1..\* Items; Productos 1..\* VentaItems |

# 4. Casos de Uso (completamente vestido)

## Caso de Uso: Iniciar sesión de consola

|  |  |
| --- | --- |
| Sección | Contenido |
| Alcance | Sistema de Gestión de Ciber |
| Nivel | Meta de usuario |
| Actor principal | Operador |
| Stakeholders e intereses | Operador (cobro ágil), Cliente (precio justo), Dueño (control), Auditoría (trazabilidad). |
| Precondiciones | Operador autenticado; consola en estado Libre. |
| Éxito garantizado (Postcondiciones) | Sesión creada y temporizador corriendo; estado de consola = Ocupada. |
| Escenario de éxito principal | 1. Selecciona consola Libre en ConsolasView. 2. Elige cliente (opcional) y tarifa vigente. 3. Confirma inicio; sistema crea Sesion e inicia timer. 4. UI actualiza estado a Ocupada y muestra duración/costo estimado. |
| Extensiones | 2a. No hay cliente registrado → opción 'Invitado'. 2b. Cliente nuevo → alta rápida (solo nombre/teléfono). 3a. Error base de datos → reintento; log y alerta. |
| Requisitos especiales | Desempeño: registrar en < 2s; Resiliencia ante cierre inesperado. |
| Tecnología/variaciones | WPF, teclado/mouse; posible lector código barras. |
| Frecuencia de ocurrencia | Alta |
| Misceláneas | - |

## Caso de Uso: Extender tiempo de sesión

|  |  |
| --- | --- |
| Sección | Contenido |
| Alcance | Sistema de Gestión de Ciber |
| Nivel | Meta de usuario |
| Actor principal | Operador |
| Stakeholders e intereses | Operador (cobro ágil), Cliente (precio justo), Dueño (control), Auditoría (trazabilidad). |
| Precondiciones | Operador autenticado; consola en estado Libre. |
| Éxito garantizado (Postcondiciones) | Extensión aplicada; nuevo total estimado actualizado. |
| Escenario de éxito principal | 1. Desde SesionesView, selecciona sesión activa. 2. Elige minutos a agregar o cambia a paquete superior. 3. Sistema recalcula costo (PricingService) y confirma. |
| Extensiones | 2a. Tarifa no compatible → sugerir cerrar y abrir nueva sesión (migración de tarifa). |
| Requisitos especiales | Exactitud de cálculo al minuto; auditoría de extensiones. |
| Tecnología/variaciones | WPF, teclado/mouse; posible lector código barras. |
| Frecuencia de ocurrencia | Alta |
| Misceláneas | - |

## Caso de Uso: Finalizar sesión y cobrar

|  |  |
| --- | --- |
| Sección | Contenido |
| Alcance | Sistema de Gestión de Ciber |
| Nivel | Meta de usuario |
| Actor principal | Operador |
| Stakeholders e intereses | Operador (cobro ágil), Cliente (precio justo), Dueño (control), Auditoría (trazabilidad). |
| Precondiciones | Operador autenticado; consola en estado Libre. |
| Éxito garantizado (Postcondiciones) | Sesión finalizada, cobro registrado, ticket impreso y consola libre. |
| Escenario de éxito principal | 1. Selecciona sesión activa. 2. Presiona Finalizar; sistema calcula total definitivo (tiempo real - pausas). 3. Elige método de pago y registra cobro (puede agregar venta de último minuto). 4. Imprime ticket y libera consola; registra en Reportes. |
| Extensiones | 2a. Cierre forzado (corte de energía) → recuperación al reiniciar y cálculo aproximado desde último checkpoint. 3a. Pago mixto → registrar efectivo + otro método. 4a. Impresora ocupada → cola de impresión o exportar PDF. |
| Requisitos especiales | Disponibilidad: >99% horario laboral; recuperación tras fallo. |
| Tecnología/variaciones | WPF, teclado/mouse; posible lector código barras. |
| Frecuencia de ocurrencia | Alta |
| Misceláneas | - |

# 5. Backlog e Iteraciones

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Historia (Como..., quiero..., para...) | Valor | Riesgo | Esfuerzo | Notas |
| PB01 | Como operador, quiero iniciar sesiones en consolas para controlar el tiempo y costo. | 5 | M | M | Crítico MVP |
| PB02 | Como operador, quiero finalizar sesiones y cobrar con ticket. | 5 | M | M | Crítico MVP |
| PB03 | Como operador, quiero extender tiempo de una sesión. | 4 | B | S |  |
| PB04 | Como dueño, quiero reporte/corte diario en CSV/PDF. | 5 | M | M |  |
| PB05 | Como operador, quiero ventas rápidas de snacks. | 4 | M | M |  |
| PB06 | Como operador, quiero gestionar inventario y alertas. | 3 | M | M |  |
| PB07 | Como operador, quiero alta rápida de clientes. | 3 | B | S |  |
| PB08 | Como dueño, quiero configurar tarifas/reglas HH. | 4 | M | M |  |

## 5.1 Plan de iteraciones (timebox 2 semanas)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Iteración | Objetivos | Ítems | Criterios de aceptación | Fecha fin | Demo |
| I1 (MVP) | Sesiones básicas (iniciar/finalizar) y ticket simple | PB01, PB02 | Iniciar/finalizar en <2s; ticket con total; recuperación tras cierre | \_\_\_\_ | Flujo completo en una consola |
| I2 | Extender sesión + Reporte diario | PB03, PB04 | Reporte CSV con totales; extensión auditable | \_\_\_\_ | Corte del día real |
| I3 | Ventas e Inventario | PB05, PB06 | Stock decrece al vender; alerta bajo stock | \_\_\_\_ | Venta con ticket |
| I4 | Clientes y Tarifas | PB07, PB08 | Alta rápida; reglas HH aplicadas | \_\_\_\_ | Descuentos/HH |

# 6. Diseño de clases (CRC breve)

|  |  |  |
| --- | --- | --- |
| Clase | Responsabilidades | Colaboradores |
| SessionService | Gestionar ciclo de vida de sesiones; timers; cálculo costo | PricingService, SesionRepository, ReceiptService |
| PricingService | Calcular precios por minuto/paquetes/HH | TarifaRepository |
| SalesService | Registrar ventas; actualizar stock | InventoryService, VentaRepository, ProductoRepository |
| InventoryService | Kardex/stock; alertas | ProductoRepository |
| ReceiptService | Render ticket y enviar a impresora/archivo | VentaRepository, SesionRepository |
| ReportingService | Cortes y reportes | Repos varios |
| MainViewModel | Navegación, sesión operador | Servicios |
| ConsolasViewModel | UI estado consolas, comandos sesiones | SessionService, PricingService |

## 6.1 Diagramas de interacción (texto)

UC Finalizar sesión: ConsolasVM.FinalizarSesion() → SessionService.Finalizar(sesionId) → PricingService.CalcularTotal() → SesionRepository.Update() → ReceiptService.Imprimir(ticket) → ConsolasVM.Refrescar().

# 7. UI/UX — Wireframes textuales

Dashboard: métricas del día, accesos rápidos (Nueva Sesión, Venta Rápida, Corte).

Consolas: cuadrícula de tarjetas con estado (Libre/Ocupada/Pausa), tiempo transcurrido, botones ‘Iniciar’, ‘Extender’, ‘Finalizar’.

Ventas: listado de productos con buscador, carrito a la derecha, total, botón Cobrar e Imprimir.

Reportes: filtros por fecha, botones Exportar CSV/PDF.

Config: tarifas, productos, impresora, impuestos.

# 8. Estrategia de pruebas

Unidad: PricingService, SessionService. Integración: repos con SQLite en memoria. Aceptación: flujos UC críticos (iniciar/finalizar sesión y cobro).

# 9. Operaciones y entrega

Distribución: carpeta portable con .NET Runtime incluido. Backups: copia diaria del .db al cierre. Logs: archivos rotativos por día.