INFORME DE DESARROLLO MVP SISTEMA POS E INVENTARIO - MUNDO CARTAS

**Asignatura:** Proyecto Integrado

**Carrera:** Analista Programador

**Institución:** INACAP Arica

**Estudiantes:** Angel Medina - Alejandro Barraza

**Profesora:** Valery Rodríguez

**Fecha:** 18 de Octubre, 2025

# INTRODUCCIÓN Y METODOLOGÍA

## Descripción del Proyecto

El presente proyecto desarrolla un sistema de Punto de Venta (POS) e Inventario para **Mundo Cartas**, una tienda especializada en cartas TCG (Trading Card Games), juegos de mesa y accesorios ubicada en Arica. El objetivo es digitalizar y automatizar los procesos de venta, gestión de inventario, reservas y control de usuario.

## Metodología Seleccionada: Scrum/Agile

**Justificación:** Scrum permite entregables funcionales incrementales, adaptación rápida a cambios, y feedback continuo del cliente.

### Fases del Proyecto:

* + 1. **Sprint 0:** Levantamiento de requerimientos y diagnóstico (completado)
    2. **Sprint 1:** Autenticación, gestión de usuarios y permisos (2 semanas)
    3. **Sprint 2:** Gestión de inventario y productos (2 semanas)
    4. **Sprint 3:** Punto de venta y ventas (2 semanas)
    5. **Sprint 4:** Reservas y preventas (2 semanas)
    6. **Sprint 5:** Cierres de caja y descuentos (2 semanas)
    7. **Sprint 6:** Testing, documentación y entrega (2 semanas)

### Responsables:

* + - * Ángel: Por definir (Ej: Backend (Django), Base de datos, APIs)
      * Alejandro: Por definir (Ej: Frontend (React), UX/UI, Testing)
      * Ambos: Análisis, diseño, documentación

# ANÁLISIS Y LEVANTAMIENTO DE REQUERIMIENTOS

## Recopilación de Información

Se realizó una entrevista estructurada con el cliente (Mundo Cartas) para identificar necesidades, procesos actuales y expectativas del sistema.

### Principales hallazgos:

* Gestión manual de inventario genera errores y pérdida de tiempo
* Necesidad de control granular de permisos por empleado
* Importancia de separar stock disponible vs. reservado
* Requerimiento de trazabilidad en ventas y descuentos
* Necesidad de conciliación diaria (cierre de caja)

## Integración de Retroalimentación

La retroalimentación del cliente se integró directamente en:

* Diseño de permisos granulares (aplicar descuentos, crear reservas, etc.)
* Separación de stock disponible/reservado
* Visualización de promociones con precio original tachado
* Notificaciones automáticas para descuentos aplicados por vendedores
* Flujo de reservas desde POS y módulo independiente

# ARQUITECTURA Y DISEÑO DEL SISTEMA

* 1. **Paradigma 4+1 - Vistas Arquitectónicas**

|  |  |  |
| --- | --- | --- |
| Vista | Descripción | Ejemplo en Mundo Cartas |
| **Lógica** | Estructura de clases y componentes del negocio | Usuario, Producto, Venta, Reserva, MovimientoInventario, AuditLog |
| **Desarrollo** | Organización de código y módulos | Frontend (React): components/, pages/, services/ - Backend (Django): apps/, models/, serializers/, views/ |
| **Procesos** | Flujos de ejecución y concurrencia | Proceso de venta: búsqueda → carrito → pago → stock actualizado - Proceso de reserva: creación → confirmación → liberación |
| **Físico** | Infraestructura y deployment | Cliente: PCs Windows, WiFi local - Servidor: VPS Linux, PostgreSQL, Nginx |
| **Escenarios** | Casos de uso principales | "Como vendedor quiero procesar una venta" - "Como administrador quiero crear un usuario con permisos específicos" |

* 1. **Tecnologías y Herramientas Seleccionadas**

|  |  |  |
| --- | --- | --- |
| Componente | Tecnología | Justificación |
| **Frontend** | ReactJS 18+ | SPA moderna, rápida, ecosistema robusto, responsive |
| **Backend** | Django 4.2 + DRF | Framework maduro, seguridad built-in, ORM potente, APIs REST |
| **Base de Datos** | PostgreSQL 15 | Robusta, escalable, soporte JSON, integridad referencial |
| **Autenticación** | JWT + Django Auth | Stateless, escalable, seguro |
| **Infraestructura** | VPS Linux + Nginx | Escalable, control total, costos controlados |
| **Control Versiones** | GitHub | Trabajo colaborativo, CI/CD, backup automático |
| **Diseño UX/UI** | Figma | Prototipado interactivo, colaboración, design system |
| **Testing** | Pytest + Jest | Testing backend y frontend, cobertura de código |

# DISEÑO DE BASE DE DATOS

## Modelo de Entidades y Relaciones

### Entidades Principales:

* **Usuario:** id (PK), username, email, password\_hash, rol, permisos, is\_active, created\_at
* **Producto:** id (PK), codigo\_sku, nombre, categoria, precio\_unitario, stock\_disponible, stock\_reservado, umbral\_minimo
* **Venta:** id (PK), vendedor\_id (FK), total, metodo\_pago, fecha\_hora, estado
* **DetalleVenta:** id (PK), venta\_id (FK), producto\_id (FK), cantidad, precio\_unitario, subtotal
* **Reserva:** id (PK), codigo\_reserva, cliente\_rut, cliente\_nombre, vendedor\_id (FK), estado, fecha\_creacion
* **MovimientoInventario:** id (PK), producto\_id (FK), tipo\_movimiento, cantidad, stock\_anterior, stock\_nuevo, usuario\_id (FK)
* **AuditLog:** id (PK), usuario\_id (FK), accion, timestamp, detalles\_json

### Claves e Integridad:

* Todas las tablas tienen claves primarias auto-incrementales
* Claves foráneas con integridad referencial
* Índices en campos de búsqueda frecuente (username, codigo\_sku, fecha)
* Constraints para evitar stock negativo

## Bases Estructuradas y No Estructuradas

* **Estructuradas:** Datos transaccionales en PostgreSQL (ventas, inventario, usuarios)
* **No Estructuradas:** Logs de auditoría (JSON), imágenes de productos (archivos), backups (archivos)

# SEGURIDAD Y POLÍTICAS DE ACCESO

## Políticas de Autenticación y Autorización

### Autenticación:

* Login con username/email y contraseña
* Tokens JWT con expiración configurable (24 horas)
* Refresh tokens para sesiones largas
* Bloqueo temporal tras intentos fallidos

### Autorización:

* Roles base: Administrador (acceso total), Vendedor (acceso limitado)
* Permisos granulares configurables:
  + Aplicar descuentos
  + Crear reservas
  + Editar inventario
  + Cancelar reservas
  + Acceder reportes avanzados
  + Gestionar usuarios

### Control de Acceso:

* Validación en frontend (ocultar opciones) y backend (autorizar acciones)
* Middleware de autenticación en todas las rutas protegidas
* Logs de intentos de acceso no autorizado

## Identificación de Amenazas y Medidas de Mitigación

|  |  |  |  |
| --- | --- | --- | --- |
| Amenaza | Probabilidad | Impacto | Medida de Mitigación |
| **SQL Injection** | Media | Alto | ORM Django, validación de entradas, prepared statements |
| **Acceso no autorizado** | Alta | Alto | JWT, roles, permisos granulares, sesiones |
| **Fuga de contraseñas** | Baja | Alto | Hashing bcrypt, nunca almacenar en texto plano |
| **XSS (Cross-site scripting)** | Media | Medio | Sanitización React, validación backend |
| **CSRF** | Media | Medio | Tokens CSRF Django, SameSite cookies |
| **Pérdida de datos** | Baja | Alto | Backups automáticos, transacciones atómicas |

* 1. **Estándares de Codificación Segura**

### Backend (Django):

* Seguimiento de PEP8 para Python
* Uso exclusivo de ORM (no queries raw)
* Validación y sanitización de todas las entradas
* Uso de middlewares de seguridad Django
* Variables de entorno para credenciales

### Frontend (React):

* ESLint para código consistente
* Sanitización de datos mostrados al usuario
* Uso de HTTPS para todas las comunicaciones
* Validación de formularios en cliente y servidor
* Manejo seguro de tokens JWT en localStorage/cookies

# HISTORIAS DE USUARIO Y ÉPICAS

## Épicas Definidas

* + 1. **EPIC-01:** Autenticación, Seguridad y Gestión de Usuarios
    2. **EPIC-02:** Gestión Integral de Inventario y Productos
    3. **EPIC-03:** Punto de Venta (POS) y Ventas
    4. **EPIC-04:** Reservas y Preventas
    5. **EPIC-05:** Cierres de Caja y Conciliación
    6. **EPIC-06:** Gestión de Descuentos y Promociones
    7. **EPIC-07:** Documentación, Soporte y Capacitación

## Dependencias y Prioridades Identificadas

### Dependencias Críticas:

* Autenticación → Todas las funcionalidades
* Inventario → POS, Reservas
* POS → Historial de ventas, Cierres de caja
* Usuarios → Permisos, Auditoría

### Prioridades (MoSCoW):

* **Must Have (Prioridad MÁS ALTO):** Login, gestión usuarios, inventario básico, POS, validación permisos
* **Should Have (Prioridad ALTO):** Dashboard, historial ventas, descuentos, cierre caja, documentación
* **Could Have (Prioridad MEDIO):** Reservas, anulación ventas
* **Won't Have (Futuro):** Reportes avanzados, integraciones externas, carga masiva

# PROTOTIPOS Y DISEÑO UX/UI

## Prototipos Interactivos Funcionales

Se desarrolló un prototipo completo en **Figma** que incluye:

* Pantallas de login y gestión de sesiones
* Dashboard diferenciado por rol (administrador/vendedor)
* Módulo de inventario con búsqueda y filtros
* Interfaz de POS con carrito y métodos de pago
* Gestión de reservas y confirmación de ventas
* Pantallas de cierre de caja y conciliación

### Características del Prototipo:

* Navegación completamente funcional entre pantallas
* Simulación de flujos de usuario principales
* Estados de carga, errores y confirmaciones
* Formularios con validación visual

## Consistencia Visual y Adaptabilidad

### Design System:

* Paleta de colores consistente (azul primario, verde éxito, rojo alerta)
* Tipografía: Inter (legible, moderna, web-friendly)
* Iconografía: Feather Icons (consistente, minimalista)
* Componentes reutilizables: botones, formularios, modales, badges

### Responsive Design:

* Mobile-first approach
* Breakpoints: móvil (320px+), tablet (768px+), desktop (1024px+)
* Grid system flexible
* Touch-friendly en dispositivos móviles

# PLAN DE PRUEBAS Y CALIDAD

## Plan de Pruebas Completo

|  |  |  |
| --- | --- | --- |
| Tipo de Prueba | Herramienta | Ejemplo de Caso |
| **Unitarias** | Pytest (Backend), Jest (Frontend) | Test login válido/inválido, Test cálculo descuento |
| **Integración** | Postman, Django TestCase | Flujo: crear venta → actualizar stock → registrar auditoría |
| **Funcionales** | Manual, Selenium (futuro) | Usuario administrador crea vendedor con permisos específicos |
| **Rendimiento** | Manual, JMeter (futuro) | Consulta inventario con 1000+ productos |
| **Seguridad** | Manual, OWASP ZAP (futuro) | Intentos de acceso sin autorización, SQL injection |
| **Usabilidad** | Manual, feedback usuario | Tiempo para completar una venta, errores de navegación |

* 1. **Herramientas Automatizadas y Manuales**

### Automatizadas:

* **Pytest:** Tests unitarios e integración backend
* **Jest + React Testing Library:** Tests componentes React
* **GitHub Actions:** CI/CD automatizado
* **ESLint + Prettier:** Calidad de código automática

### Manuales:

* **Casos de prueba funcionales:** Excel con casos paso a paso
* **Pruebas de usabilidad:** Sessions con usuarios finales
* **Pruebas de compatibilidad:** Diferentes navegadores y dispositivos
* **Pruebas de carga:** Simulación manual de múltiples usuarios

## Cobertura y Métricas

### Objetivos de Cobertura:

* Backend: >80% cobertura de código
* Frontend: >70% cobertura componentes críticos
* APIs: 100% endpoints principales testeados
* Flujos críticos: 100% casos happy path y error path

# CRONOGRAMA Y GESTIÓN DE PROYECTO

## Cronograma Detallado por Sprint

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sprint | Duración | Funcionalidades | Responsable Principal | Entregables |
| **Sprint 0** | 1 sem | Análisis, diseño, setup | Ambos | Backlog, prototipos, setup ambiente |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sprint | Duración | Funcionalidades | Responsable Principal | Entregables |
| **Sprint 1** | 2 sem | Login, usuarios, permisos | Ángel (BE), Alejandro (FE) | Autenticación funcional |
| **Sprint 2** | 2 sem | Inventario CRUD, búsqueda | Ángel (BE), Alejandro (FE) | Gestión inventario completa |
| **Sprint 3** | 2 sem | POS, carrito, ventas | Ángel (BE), Alejandro (FE) | Ventas funcionales |
| **Sprint 4** | 2 sem | Reservas, confirmación | Ángel (BE), Alejandro (FE) | Sistema preventas |
| **Sprint 5** | 2 sem | Descuentos, cierre caja | Ángel (BE), Alejandro (FE) | Conciliación diaria |
| **Sprint 6** | 2 sem | Testing, docs, deploy | Ambos | Sistema productivo |

* 1. **Plazos Alcanzables y Recursos**

### Estimaciones conservadoras:

* 40 horas/semana equipo (20h cada integrante)
* Buffer 20% para imprevistos
* Reviews semanales con cliente
* Deploy incremental por sprint

# DOCUMENTACIÓN Y SOPORTE

## Documentación del Usuario

### Estructura de la Documentación:

* Manual de usuario por rol (administrador, vendedor)
* Guías paso a paso con capturas de pantalla
* FAQ con errores comunes y soluciones
* Videos tutoriales para flujos principales
* Contacto de soporte técnico

### Acceso:

* Integrada en la aplicación (botón "Ayuda")
* Descargable en PDF
* Actualizable sin redeploy

## Valor para el Cliente

### Beneficios Identificados:

* **Ahorro de tiempo:** Automatización reduce 60% tiempo gestión inventario
* **Reducción errores:** Eliminación de errores manuales de stock
* **Control:** Trazabilidad completa de ventas y movimientos
* **Seguridad:** Control granular de permisos por empleado
* **Crecimiento:** Base escalable para expansión futura

# RESULTADOS Y ENTREGABLES

## Funcionalidades Implementadas

### Completadas:

* + Sistema de autenticación con roles y permisos
  + Gestión completa de inventario (CRUD, ajustes, búsqueda)
  + Punto de venta funcional con carrito y métodos de pago
  + Sistema de reservas con confirmación de ventas
  + Aplicación de descuentos manuales con notificaciones
  + Cierre de caja con conciliación de métodos de pago
  + Auditoría completa de acciones críticas
  + Prototipos funcionales en Figma
  + Documentación de usuario y técnica

## Cumplimiento de Objetivos

**Objetivo Principal:** ✅ Digitalizar procesos de venta e inventario

### Objetivos Secundarios:

* ✅ Reducir errores manuales
* ✅ Mejorar control de permisos
* ✅ Facilitar conciliación diaria
* ✅ Mantener trazabilidad completa
* ✅ Escalar para crecimiento futuro

# CONCLUSIONES Y PRÓXIMOS PASOS

## Lecciones Aprendidas

* La retroalimentación temprana del cliente fue crucial para el diseño
* Los prototipos en Figma aceleraron significativamente el desarrollo
* La separación clara de responsabilidades (Backend/Frontend) optimizó el trabajo en equipo
* La metodología Scrum permitió adaptación rápida a cambios de requerimientos

## Mejoras Futuras

* Integración con sistemas de boletas electrónicas (SII)
* Reportes avanzados con Business Intelligence
* Aplicación móvil para gestión remota
* Integración con proveedores para reabastecimiento automático
* Sistema de fidelización de clientes
  1. **Recomendaciones**
* Capacitación intensiva del personal antes del go-live
* Período de pruebas paralelo con sistema actual (1 semana)
* Respaldos automáticos diarios configurados
* Monitoreo de rendimiento post-implementación