## **Planificación y Gestión de Proyectos**

### **a. Plan de Proyecto**

**Objetivo general:** Desarrollar un sistema de gestión de inventarios para pequeñas y medianas empresas, que integre principios de aseguramiento de la calidad del software en todas sus etapas, garantizando robustez funcional, facilidad de uso y estándares altos de seguridad.

**Alcance:**

* Módulos funcionales: Gestión de productos, control de stock, interfaz amigable, reportes, y API de integración.
* Usuarios: Administrador, Empleado, Invitado.
* Entornos: Desarrollo, Pruebas, Producción (con Docker y CI/CD).
* Soporte multiplataforma: Navegadores modernos (Pruebas con Playwright).

**Entregables:**

1. Documento de requisitos funcionales y no funcionales.
2. Documentación técnica (arquitectura, instalación, mantenimiento).
3. Sistema desarrollado y desplegado.
4. Pruebas automatizadas (seguridad, regresión, aceptación, etc.).
5. Informe final de calidad y revisión post-mortem.

**Cronograma Tentativo (ejemplo de 6 semanas):**

| Semana |  |  |
| --- | --- | --- |
| 1 | - Recolección y análisis de requerimientos  - Diseño de arquitectura inicial | - Documento de requisitos (funcionales y no)  - Boceto inicial de arquitectura |
| 2 | - Diseño de base de datos  - Diagramas de clases  - Configuración del entorno (repositorio, CI/CD, Trello) | - Diagrama de clases- Estructura de base de datos  - Trello configurado con tareas iniciales |
| 3 | - Desarrollo del módulo de Gestión de Productos (CRUD)  - Pruebas unitarias del módulo | - Endpoints de productos funcionales  - Pruebas unitarias (JUnit/TestNG) |
| 4 | - Desarrollo del módulo de Control de Stock Alertas por stock mínimo  - Pruebas funcionales del módulo anterior | - Funcionalidad de stock implementada  - Pruebas funcionales documentadas |
| 5 | - Desarrollo de la interfaz de usuario (productos y stock)  - Documentación técnica inicial | - Pantallas funcionales de productos y stock  - Documentación técnica v1 |
| 6 | - Implementación de roles (Admin, Empleado, Invitado)  - API de integración  - Pruebas de seguridad y regresión | - Gestión de usuarios  - Integración básica por API  - Pruebas de seguridad (OWASP ZAP / manual) |
| 7 | - Pruebas finales:  - Aceptación (Cucumber)  - Compatibilidad (Playwright)  - Pruebas de estrés (JMeter)  - Observabilidad (OpenTelemetry) | - Resultados de pruebas documentados  - Monitoreo básico implementado |
| 8 | - Despliegue en entorno simulado y producción  - Revisión post-mortem  - Preparación de presentación final | - Sistema desplegado  - Informe post-mortem  - Demo lista para evaluación |

**Gestión de Riesgos**

| Riesgo | Descripción | Probabilidad | Impacto | Estrategia de mitigación |
| --- | --- | --- | --- | --- |
| Falla de base de datos | Pérdida de conexión o corrupción de datos en PostgreSQL. | Media | Alta | Uso de backups automáticos, monitorización activa y restauración rápida. |
| Errores de seguridad | Accesos no autorizados, fuga de datos o ataques (por ejemplo, inyecciones). | Alta | Alta | Implementar JWT, control de roles, validaciones y pruebas de seguridad constantes. |
| Pérdida de información por bugs | Fallos en la lógica que provoquen errores en inventario. | Media | Alta | Pruebas unitarias y de integración, revisiones de código, pruebas de aceptación (Cucumber). |
| Sobrecarga del servidor | Caídas del sistema ante alta demanda o mal dimensionamiento. | Baja | Media | Monitorear uso de recursos, pruebas de estrés, escalabilidad horizontal. |
| Incompatibilidad entre módulos | Conflictos al integrar nuevas funcionalidades o dependencias. | Media | Media | Documentación técnica clara, control de versiones y pruebas continuas. |
| Falta de capacitación del personal | Usuarios finales no entienden el uso del sistema. | Baja | Alta | Crear manuales de usuario, capacitación inicial y soporte técnico. |
| Problemas de compatibilidad con navegadores | Problemas en la interfaz web en ciertos dispositivos. | Media | Media | Pruebas cross-browser (opcional con herramientas como Playwright o Cypress). |