Guía Informe final de proyecto de Desarrollo de SIAs

# Introducción

Este documento guía está diseñado para ayudarte a través de todas las fases del ciclo de vida del desarrollo de software utilizando principios de ingeniería de software. Completa cada sección según corresponda al proyecto que te haya tocado. Recuerda que la calidad y la claridad en cada una de estas fases son cruciales para el éxito de tu proyecto.

# Etapa 1: Visión y Alcance del Proyecto

En esta fase se debe establecer con claridadel **para qué**, **para quién**, **qué incluirá** y **qué no** incluirá el sistema a construir. Esta sección guía todas las decisiones posteriores (modelo de desarrollo, requerimientos, diseño, pruebas y operación).

1.1 Visión (máx de 3 frases)

Nuestro sistema convierte las órdenes de venta en facturación simple, económica y confiable, ayudando a las empresas a ahorrar tiempo, cumplir con el SII y controlar sus finanzas para crecer con confianza.

1.2 Problema y contexto

El problema que buscamos solucionar es poder con el software llegar a las pequeñas empresas con un sistema de facturación simple, accesible y seguro. Las grandes empresas usan el sistema de SAP que por su alto valor las pequeñas empresas no pueden acceder a este y necesitan una alternativa que les permita mantener el orden de sus datos de ventas.

1.3 Usuarios y stakeholders

Usuarios:

* Administrador: Tiene el control total del sistema, de los inventarios y de los pagos.
* Facturador: Registrar ventas rápido, emitir facturas, consultar stock.
* Contador: Revisar cuentas, generar reportes financieros, cumplir con normativa tributaria.

1.4 Propuesta de valor

* Ahorro de tiempo y costos administrativos
* Cumplimiento normativo con el SII
* Simplicidad y facilidad de uso
* Mejor control financiero
* Profesionalización del negocio

# Etapa 2: Organización y Planificación del Proyecto

En esta fase, se debe realizar toda la organización administrativa del proyecto. Esto incluye la planificación de tareas, asignación de roles, creación de una carta Gantt, y selección de herramientas de gestión de proyectos. Este paso es crucial para asegurar que el proyecto avance de manera ordenada y eficiente.

2.1 Definición de Roles y Responsabilidades.

Persona 1: Valentina Aróstica Ávalos

1. Jefe de Proyecto / Project Manager • Responsable de planificar, coordinar y controlar el proyecto.
2. Administrador de Base de Datos (DBA) • Diseña, mantiene y asegura la integridad de la BD
3. Encargado de Documentación / Librería Técnica • Centraliza los manuales, diagramas, trazabilidad de requisitos y políticas.

Persona 2: Luis Molina Rivera.

1. Equipo de Desarrollo (Programadores) • Codifican el sistema según los diseños y requisitos
2. Soporte / Mesa de ayuda • Atiende incidencias de usuarios cuando el sistema entra en operación.
3. Tester / QA (Control de Calidad) • Diseña y ejecuta casos de prueba.

Persona 3: Mayra Gutierrez Paredes

1. Arquitecto de Software / Diseñador • Define la arquitectura del sistema: componentes, bases de datos, integraciones, patrones de diseño.
2. Analista de Negocio / Analista de Requerimientos • Traduce las necesidades del cliente en requisitos funcionales y no funcionales.
3. Administrador de Sistemas / Infraestructura • Configura y mantiene los servidores donde corre el software.

2.2 Planificación de Tareas (Carta Gantt/ Kanban.).

[Diagrama de Gantt simple.xlsx](https://alumnosucncl-my.sharepoint.com/:x:/g/personal/valentina_arostica_alumnos_ucn_cl/EU8TnpPxy8xChQ6ONehYUaABu9EDdJvy1PPa1zQTQ4iG5g?e=lLlTLa)

2.3 Herramientas de Organización y Gestión del Proyecto.

Sistema Control de versiones: Git, GitHub

Herramienta de Gestión de proyectos: Notion

Herramienta de Diseño: Figma

2.4 Matriz de riesgos inicial (seguridad, datos, continuidad).

| **Riesgo identificado** | **Categoría (Seguridad/Datos/Continuidad)** | **Probabilidad**  **(Alta/Media/Baja)** | **Impacto (Alto/Medio/Bajo)** | **Mitigación inicial** | **Responsable** |
| --- | --- | --- | --- | --- | --- |
| Acceso no autorizado a la BD | Seguridad | Media | Alto | Implementar login con roles y contraseñas seguras | Persona 2 (Backend) |
| Pérdida de información crítica | Datos | Media | Alto | Respaldos semanales de la BD (dump MySQL) | Persona 1 (BD) |
| Caída del servidor en la demo | Continuidad | Alta | Medio | Tener copia local del sistema y BD para contingencia | Persona 2 (Backend) |
| Fuga de datos sensibles de clientes | Seguridad | Baja | Alto | Validar accesos, no mostrar datos privados en frontend | Persona 3 (Frontend) |
| Error humano al borrar registros | Datos | Media | Medio | Implementar “soft delete” o confirmación antes de eliminar | Persona 1 (BD) |
| Retraso en el desarrollo por tiempo | Continuidad | Alta | Medio | Planificar con tareas semanales y reuniones cortas de avance | Todos |
| Inyección SQL en endpoints | Seguridad | Baja | Alto | Usar consultas parametrizadas en Flask y sanitizar entradas | Persona 2 (Backend) |
| Falta de integración Front-Back | Continuidad | Media | Medio | Pruebas semanales con Postman y React conectado | Persona 3 (Frontend) |

2.5 Checklist de decisión rápida (producto vs SaaS, nube vs on-premise, etc.).

## 1. Producto instalado vs SaaS (Software como servicio)

* Producto instalado (On-Premise)
* ~~SaaS (Servicio en la nube)~~

**Justificación**

El modelo SaaS es mucho más indicado para este proyecto ya que, al querer trabajar con pequeñas empresas, a estas se les vuelve mucho más eficiente el arranque y manejo del software, puesto que, podrán acceder a él en cualquier momento y en el dispositivo que gusten sin la necesidad de costear la instalación de este.

## 2. Infraestructura: Nube vs On-Premise

* On-Premise**.**
* ~~Nube (IaaS/PaaS/DBaaS)~~

**Justificación**

La Nube por lo general es mucho más accesible a nivel de costos (un punto importante si el proyecto va dirigido a pequeñas y medianas empresas). Por escalabilidad también es mejor la nube, puesto que si en algún futuro la empresa tiene un crecimiento de personal o de nivel en general, con la nube es más factible aumentar los recursos sin necesidad de cambiar la infraestructura.

La nube ofrece cifrado, copias de seguridad y una alta disponibilidad por lo que a nivel de seguridad es lo mejor a ofrecer a una empresa que no tiene los suficientes recursos.

## 3. Arquitectura: Monolito vs Microservicios/Servicios

* ~~Monolito modular~~
* Microservicios / SOA

**Justificación**

* El proyecto sólo contiene un backend
* El equipo es pequeño por lo que con un monolito es más fácil de desarrollar en comparación a múltiples micros servicios

## 4. Proceso de desarrollo: Plan-Dirigido vs Ágil/DevOps

* ~~Plan-dirigido (Cascada / V-Model)~~
* Ágil / DevOps (Scrum, Kanban, XP)

**Justificación**

Escogimos un plan dirigido con un desarrollo iterativo e incremental, puesto que, iniciamos el proyecto con un alcance concreto (solo facturas de ventas) pero no cerramos la posibilidad de mejorar y ampliar los servicios en fases posteriores. Con el plan dirigido generamos claridad en los entregables de cada fase, tenemos un mejor control a los cambios, y construimos una base sólida para futuras ampliaciones

# Etapa 3: Selección del Modelo de Desarrollo

En esta fase, deberás seleccionar el modelo de desarrollo de software que utilizarás para guiar el proyecto. Justifica tu elección basándote en las características del proyecto y las características de cada modelo.

3.1 Descripción del Proyecto

Facturación para pequeñas empresas

### Resumen Ejecutivo

El proyecto consiste en **diseñar, desarrollar e implementar un sistema integral de facturación digital**, orientado a empresas que buscan **automatizar, controlar y optimizar** su proceso de emisión y gestión de facturas. El sistema permitirá gestionar transacciones de manera **eficiente, confiable y segura**, garantizando cumplimiento normativo y ofreciendo herramientas analíticas para la toma de decisiones financieras estratégicas.

### Objetivos del Proyecto

**Objetivo General:** Implementar un sistema de facturación digital que centralice y automatice la emisión, almacenamiento y gestión de facturas, garantizando exactitud, trazabilidad y disponibilidad de la información financiera.

**Objetivos Específicos:**

1. Permitir lacreación y emisión de facturas electrónicas cumpliendo con la normativa tributaria vigente.
2. Integrar la gestión de cuentas por cobrar y por pagar para facilitar el seguimiento de transacciones pendientes.
3. Generar reportes financieros y contables en tiempo real para la planificación y control gerencial.
4. Implementar un módulo de inventario vinculado con las ventas y facturación.
5. Garantizar interoperabilidad con otros sistemas corporativos, como ERP o bancos.
6. Asegurar la seguridad y confidencialidad de la información mediante control de acceso y almacenamiento seguro.
7. Facilitar la usabilidad y accesibilidad, permitiendo un aprendizaje rápido y uso intuitivo para los usuarios.

### 3. Alcance del Proyecto

**Incluye (In-scope):**

* Creación, edición, emisión y almacenamiento de facturas.
* Gestión de clientes, productos y servicios vinculados a las facturas.
* Control de cuentas por cobrar.
* Reportes financieros y analíticos automáticos (mensuales, trimestrales, anuales).
* Integración con inventario y actualización automática de stock.
* Acceso seguro mediante roles y permisos diferenciados.
* Exportación de datos a formatos estándar (Excel, PDF, CSV).
* Interoperabilidad con otros sistemas contables o bancarios.

**No incluye (Out-of-scope):**

* Gestión de nómina y recursos humanos.
* Procesos de compras externas y logística de proveedores.
* Funcionalidades de CRM avanzadas (marketing, seguimiento de clientes).
* Integración con sistemas de terceros no autorizados.

### Arquitectura y Componentes del Sistema

**Arquitectura General:**

* Sistema web con acceso seguro desde cualquier dispositivo.
* Base de datos relacional centralizada para asegurar integridad y trazabilidad.
* API de integración para interoperabilidad con sistemas externos.
* Panel de administración para usuarios internos y gerenciales.

**Módulos Principales:**

1. **Gestión de Facturación:** emisión, edición, anulación y almacenamiento de facturas.
2. **Gestión de Clientes y Productos:** registro de datos, historial de transacciones, códigos de producto y stock asociado.
3. **Cuentas por Cobrar y Pagar:** seguimiento de pagos, vencimientos y alertas automáticas.
4. **Reportes y Analítica:** generación de informes financieros y dashboards para toma de decisiones.
5. **Seguridad y Accesos:** control de permisos, auditorías y respaldo de información.
6. **Integración:** conexión con bancos, ERP y otros sistemas contables.

### Requisitos de Calidad y Aseguramiento

**Funcionalidad:**

* Cumplimiento de requerimientos especificados.
* Exactitud de cálculos y emisión de documentos.
* Interoperabilidad con otros sistemas.

**Confiabilidad:**

* Sistema estable y tolerante a fallos.
* Alta disponibilidad y recuperación ante incidentes.
* Registro de auditoría y control de cambios.

**Usabilidad:**

* Interfaz intuitiva y accesible para distintos niveles de usuario.
* Manuales y ayuda contextual integrada.

**Eficiencia:**

* Procesamiento rápido de grandes volúmenes de datos.
* Optimización de consultas y generación de reportes.

**Seguridad:**

* Almacenamiento cifrado de información sensible.
* Autenticación por roles y permisos diferenciados.
* Registro de actividades críticas y respaldos automáticos.

**Mantenibilidad y Escalabilidad:**

* Código modular y documentado.
* Capacidad de agregar nuevos módulos o funcionalidades en el futuro.
* Fácil actualización de la base de datos y el software.

### Beneficios Esperados

* Reducción de errores humanos en la facturación y contabilidad.
* Mayor eficiencia en la gestión de ventas y cobros.
* Información financiera confiable y oportuna para la toma de decisiones.
* Cumplimiento de normas tributarias y contables.
* Optimización de recursos y ahorro en procesos administrativos.
* Capacidad de escalamiento para soportar el crecimiento de la empresa.

### 

### 

### 

### Riesgos y Mitigaciones

| **Riesgo** | **Probabilidad** | **Mitigación** |
| --- | --- | --- |
| Fallos en el sistema durante operación crítica | Media | Implementar redundancia y backup automático |
| Pérdida de datos por fallas de seguridad | Baja | Cifrado, control de accesos y respaldo diario |
| Resistencia al cambio por parte de usuarios | Media | Capacitación y manuales de uso |
| Integración con sistemas externos fallida | Baja | Pruebas piloto y APIs estandarizadas |

3.2 Modelos de Desarrollo Considerados (cascada, incremental, iterativo, ágil)

* Modelo Iterativo
* Modelo Incremental

3.3 Justificación del Modelo Seleccionado (Cómo este modelo se adapta al ciclo de vida del proyecto elegido)

Elegimos el Modelo iterativo-Incremental ya que podemos dividir nuestro proyecto en iteraciones demostrables y funcionales que nos permitan tener facilidad de acoger los feedbacks sin necesidad que afecten y se tenga que cambiar todo el proyecto y empezar de nuevo, al igual que la corrección de errores que se puedan tener.

# Etapa 4: Recolección y Análisis de Requerimientos

En esta fase, deberás identificar y documentar los requisitos del sistema. Estos pueden dividirse en requisitos funcionales (qué debe hacer el sistema) y requisitos no funcionales (cómo debe comportarse el sistema). Identificar y justificar la técnica que ocuparon para identificar los requisitos (Como documento Anexo deben entregar el [DER](about:blank))

4.1 Identificación de la o las técnicas para la recolección de los requerimientos.

4.2 Requisitos Funcionales y No funcionales.

4.3 Requisitos de seguridad y privacidad (ejemplo: cifrado, roles, logs).

4.4 Otros Requisitos (de actores, funcionales, interacción).

4.5 Priorización de Requerimientos.

# Etapa 5: Diseño del Sistema

En esta fase, debes diseñar la arquitectura general del sistema. Define los componentes principales y sus interacciones, así como la base de datos que soportará al sistema.

3.1 Diagrama de Flujo de Datos.

3.2 Modelo de Datos Semánticos.

3.3 Modelo de Dominio.

3.3.1 Diagrama de Clases.

3.4 Modelo de Objetos.

3.4.1 Diagrama de Objetos.

3.4.2 Diagrama de Secuencia.

3.4 Diagrama de Arquitectura del Sistema (monolítico modular / servicio).

3.5 Diseño de la Base de Datos (Modelo Relacional).

3.6 Diseño de la Interfaz de Usuario.

3.7 Esquema de seguridad: autenticación/autorización, cifrado, segregación de datos.

3.8 Política de acceso y roles.

# Etapa 4: Desarrollo e Implementación

En esta fase, el diseño del sistema se convierte en código ejecutable. Documenta las decisiones importantes que tomaste durante la implementación y describe cómo implementaste los principales módulos del sistema.

4.1 Lenguajes y Herramientas Utilizados.

4.2 Implementation de Módulos Principales.

* 1. Problemas y Soluciones Durante la Implementación.

4.4 Gestión de versiones (Git/CI/CD).

4.5 Políticas de backup en desarrollo (ej.: dumps automáticos de BD).

# Etapa 5: Pruebas y corrección de errores

En esta fase, debes probar el sistema para asegurarte de que cumple con los requisitos especificados. Esto incluye pruebas unitarias, pruebas de integración, pruebas del sistema completo y pruebas de aceptación.

5.1 Plan de Pruebas (unitarias, integración, siatema, aceptación)

5.2 Casos de prueba documentados.

5.3 Pruebas de seguridad: inyección SQL, manejo de errores, validación de datos.

5.4 Pruebas de recuperación: restauración de BD desde backup, medición de RTO/RPO.

5.4 Reporte y corrección de errores (Formato para registrar fallos y tiempo de reparación).

| Información General del Proyecto   * Nombre del Proyecto: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ * Nombre del Software: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ * Fecha de Inicio del Registro: //\_\_\_\_ * Fecha de Fin del Registro: //\_\_\_\_ * Nombre del Estudiante/Grupo: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ * Nombre del Responsable: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  | **ID de Fallo** | **Fecha y Hora del Fallo** | **Descripción del Fallo** | **Condiciones Previas** | **Impacto del Fallo** | **Tiempo de Detección (minutos)** | **Tiempo de Reparación (minutos)** | **Estado de la Reparación** | **Comentarios Adicionales** | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 1 | **/**/\_\_\_\_ hh | Breve descripción del fallo | Ejemplo: Al intentar dividir por cero, el software se cierra inesperadamente | Alto | 5 | 15 | Reparado / Pendiente | Detalles adicionales si es necesario | | 2 | **/**/\_\_\_\_ hh | Breve descripción del fallo | Ejemplo: Error al marcar tarea como completada | Medio | 3 | 10 | Reparado / Pendiente | Detalles adicionales si es necesario | | 3 | **/**/\_\_\_\_ hh | Breve descripción del fallo | Ejemplo: Resultados incorrectos en la división de números | Alto | 2 | 5 | Reparado / Pendiente | Detalles adicionales si es necesario | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

# Etapa 6: Seguridad, Respaldo y Recuperación de Datos

En esta etapa se definen, implementan y documentan las medidas que garanticen la integridad, disponibilidad y confidencialidad de la información gestionada por el sistema. Se deben aplicar controles de seguridad como políticas de acceso, gestión de usuarios y cifrado de datos. Asimismo, se establecen y prueban las políticas de respaldo periódico de la base de datos y de los archivos críticos del sistema, definiendo su frecuencia, responsables y herramientas utilizadas. Finalmente, se deben diseñar y ejecutar mecanismos de recuperación que permitan restaurar la información en caso de fallos, incidentes de seguridad o pérdida de datos, midiendo los tiempos de recuperación (RTO) y los puntos de restauración alcanzados (RPO).

6.2 Plan de respaldo: frecuencia, herramientas utilizadas, validación de integridad.

6.3 Plan de recuperación: pasos para restaurar, responsables, tiempos medidos.

6.4 Evidencia de simulacros de restore.

### Conclusiones y Lecciones Aprendidas

* Cómo se cumplieron los RA del curso.
* Valor del sistema desarrollado para la organización.
* Próximos pasos de mejora.