Guía Informe final de proyecto de Desarrollo de SIAs

# Integrantes:

1. Heidy Calizaya
2. Rodrigo Díaz
3. Nicole García
4. Ricardo Zepeda

# Introducción

En el presente informe se desarrolla la propuesta de un sistema de gestión para la biblioteca de la Universidad Católica del Norte, enfocado en mejorar los procesos de reserva de cubículos de estudio.. El proyecto considera la implementación de una página web diseñada en HTML y JavaScript, conectada a una base de datos en SQL Workbench, lo que asegura un entorno confiable, eficiente y accesible para estudiantes y personal de la biblioteca.  
El sistema GRCL no solo se centra en las funciones de gestión y reserva de cubículos, sino también en una base tecnológica sólida. Para su desarrollo se emplea SQL Workbench, encargado de almacenar y administrar los datos de usuarios, cubículos y reservas; HTML, para estructurar la interfaz web que utilizan estudiantes y funcionarios y JavaScript, que aportará dinamismo y validación en tiempo real, mejorando la experiencia del usuario. Con esto se integra una capa de presentación (HTML + JavaScript) y una capa de datos (SQL Workbench), asegurando un sistema accesible y eficiente para los estudiantes.

# Etapa 1: Visión y Alcance del Proyecto

## 1.1 Visión (máx de 3 frases)

Construir un sistema de reservas de cubículo que permita a los estudiantes y funcionarios consultar, reservar y gestionar espacios en tiempo real, garantizando un acceso equitativo y reduciendo la saturación del sistema actual.

## 1.2 Problema y contexto

El sistema de reservas que actualmente utilizan en la biblioteca de la UCN genera falta de equidad en el acceso a cubículos de estudio. Este problema se ha evidenciado directamente en la experiencia diaria de los estudiantes, quienes deben desplazarse y hacer filas y subir hasta cada meson de cada piso para preguntar por la disponibilidad, provocando descontento en los estudiantes. Es por esto que surge la necesidad de implementar una solución tecnológica que automatice las reservas, registre entradas y salidas en tiempo real y reduzca la carga administrativa del personal de biblioteca.

## 1.3 Usuarios y stakeholders

* **Usuarios directos:**
  + **Estudiantes:** requieren un sistema rápido, claro y confiable para reservar cubículos para optimizar el tiempo de estudio.
  + **Personal de biblioteca:** necesita una herramienta que facilite la administración de horarios, disponibilidad y control de reservas.
* **Stakeholders:**
  + **Estudiantes de la UCN**: Son los principales beneficiarios , cuyo rendimiento académico puede mejorar al contar con espacios de estudio gestionados de manera más justa y eficiente.

## 1.4 Propuesta de valor

El sistema ofrecerá múltiples beneficios:

* **Comodidad:** acceso a reservas en línea sin necesidad de asistir presencialmente a la biblioteca.
* **Accesibilidad rápida y simple:** interfaz web intuitiva que funciona en distintos navegadores, disponible en todo momento.
* **Eficiencia operativa:** automatización de reservas que reduce la carga administrativa del personal y evita errores humanos.
* **Uso equitativo de recursos:** control de tiempos y disponibilidad en tiempo real, garantizando mayor acceso para todos los estudiantes.
* **Transparencia y confiabilidad:** reportes de uso que permitirán medir la demanda y mejorar continuamente la gestión de la biblioteca.

# Etapa 2: Organización y Planificación del Proyecto

## 2.1 Definición de Roles y Responsabilidades.

* **Jefe de proyecto:** Rodrigo Diaz

Se encargará de planificar, coordinar y monitorear el proyecto para que se ejecute de manera eficiente.

* **Analista de negocios:** Ricardo Zepeda

Será encargado de analizar el negocio para que el proyecto esté acorde a las necesidades de los estudiantes.

* **Arquitecto:** Heidy Calizaya

Se encargará de planificar el proyecto y construir el código para que funcione sin problemas

* **Equipo de desarrollo:** Todos

Se encargará de construir el proyecto, con el código en html y con sus funcionalidades.

* **Tester:** Ricardo Zepeda

Se encargará de probar el proyecto para verificar que funcione.

* **Administrador Base de datos:** Nicole Garcia

Estará a cargo de la base de datos, en donde se tendrá información relevante.

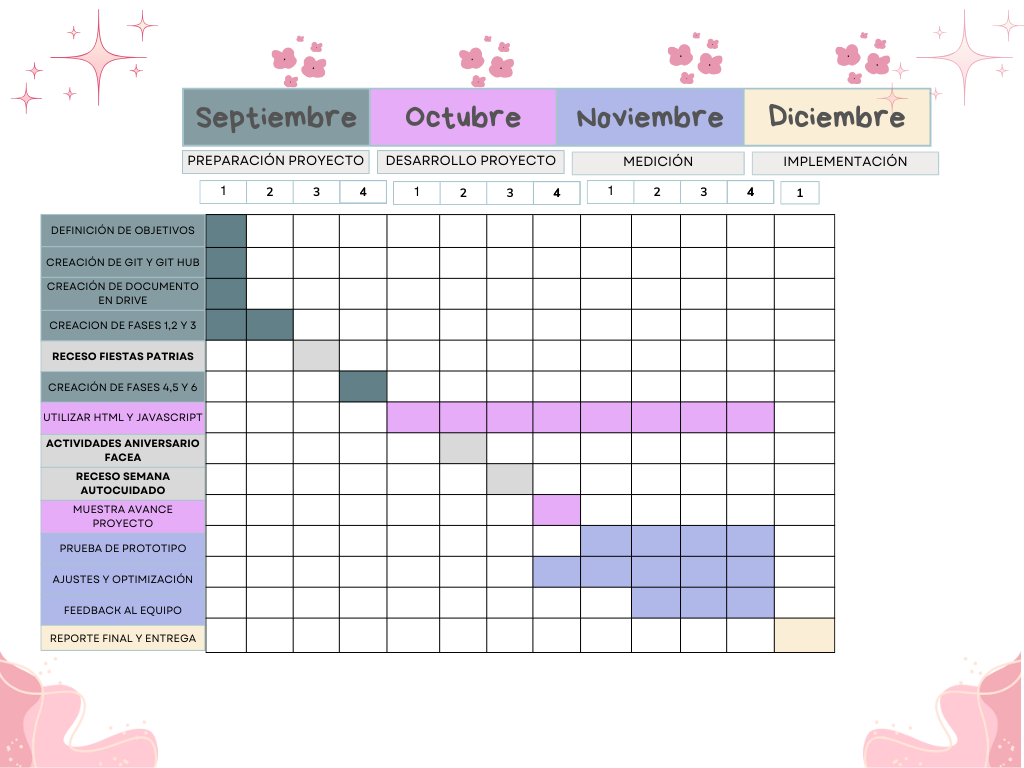
* **Encargado de documentación:** Rodrigo Diaz

Se encargará de organizar la documentación del proyecto.

* **Soporte:** Todos

Se encargará de planificar y controlar el proyecto, utilizar herramientas de gestión y ayudar a prevenir riesgos.

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



## 2.3 Herramientas de Organización y Gestión del Proyecto.

En este proyecto se utilizan las siguientes herramientas que permiten un control eficiente y visual del progreso del proyecto:

* **Carta Gantt:** es adecuada porque facilita la planificación y seguimiento de las fases del proyecto de manera cronológica, permitiendo visualizar la duración de cada actividad, sus dependencias y los hitos importantes como la semana de receso o la semana de la carrera. Esto asegura que nuestro equipo mantenga un control del tiempo y que se cumpla con los plazos establecidos.
* **Kanban:** Nos permite gestionar las tareas de forma dinámica y colaborativa como equipo, mostrando el estado de cada una (pendiente, en proceso, finalizada). Su uso promueve la organización, la priorización de actividades para la creación del HTML, códigos, etc. y la comunicación efectiva dentro del equipo, evitando retrasos y mejorando la coordinación.

## 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* | *Políticas de acceso y control de permisos* | *Administrador Base de Datos* | | *Pérdida de información crítica* | *Continuidad* | *Baja* | *Alto* | *Implementar respaldos automáticos diarios en la nube y pruebas mensuales de recuperación de datos* | *Tester* | | *Fallas del computador* | *Continuidad* | *Baja* | *Media* | *Tener otros computadores para que no se produzcan problemas con el proyecto* | *Soporte* | | *Fallo en registro de clientes* | *Continuidad y datos* | *Media* | *Alto* | *Revisar el código* | *Arquitecto* | | *Fallas en guardar datos en la base de datos* | *Datos* | *Media* | *Media* | *Verificar los datos guardados* | *Soporte* | | *Duplicación de datos* | *Datos* | *Media* | *Alto* | *Verificar el código de la base de datos* | *Administrador de base de datos* | | *Interfaz confusa para usuarios* | *Datos* | *Media* | *Media* | *Capacitación de uso de la interfaz* | *Equipo de desarrollo* | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

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

| 1. **Producto instalado vs SaaS (Software como servicio)**  * ~~SaaS (Servicio en la nube)~~   **Justificación:** El programa será utilizado por los estudiantes, de manera online. 2. **Infraestructura: Nube vs On-Premise**  * ~~Nube (IaaS/PaaS/DBaaS)~~   **Justificación:** La información será guardada en la nube, donde se usará una base de datos 3. **Arquitectura: Monolito vs Microservicios/Servicios**  * ~~Microservicios / SOA~~   **Justificación:**  Optamos por un sistema de Micro Servicios/SOA porque en este caso se va a manejar un sistema de biblioteca en donde tendrán distintas funciones que son independientes entre sí como: gestionar la entrega de libros y entrega de cubículos para estudio, además de reportar el control de usuarios. 4. **Proceso de desarrollo: Plan-Dirigido vs Ágil/DevOps**  * ~~Ágil / DevOps (Scrum, Kanban, XP)~~   **Justificación:** El sistema requiere flexibilidad porque puede que los requisitos del programa cambien con el tiempo. Un enfoque Ágil con Scrum permite entregar los módulos funcionales en iteraciones cortas como: reservas de cubículos, préstamos de libros y de eso se puede sacar reportes. Además el uso de DevOps puede facilitar una rápida actualización del sistema web. |
| --- |

# Etapa 3: Selección del Modelo de Desarrollo

## 3.1 Descripción del Proyecto

Aplicación que se encarga de la administración de una biblioteca online y sistema de reserva de cubículos para la Universidad Católica del Norte. Este programa se encargará de que el estudiante pueda reservar cubículos de manera ordenada y eficiente para aligerar el trabajo de los funcionarios de la biblioteca.

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

En este programa se utiliza el **modelo de desarrollo iterativo** porque el sistema de gestión de biblioteca requiere un proceso de construcción gradual, en el que se puedan ir implementando versiones funcionales del sistema y mejorándolas con retroalimentación constante de los usuarios (alumnos y personal de biblioteca).

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

El modelo iterativo para nuestro proyecto permite:

* **Flexibilidad ante cambios:** los requisitos del sistema, como reservas de cubículos o gestión de libros, pueden modificarse o ampliarse sin necesidad de rehacer el proyecto desde cero.
* **Entrega temprana de resultados:** se pueden liberar módulos básicos en las primeras iteraciones (por ejemplo, registro de usuarios o reservas simples) y posteriormente ir agregando nuevas funciones (reportes, notificaciones, gestión de horarios).
* **Validación continua:** cada iteración se prueba con los usuarios reales, lo que asegura que el sistema responda a sus necesidades y evita errores acumulados.
* **Menor riesgo:** al dividir el desarrollo en ciclos más pequeños, se reducen fallas graves y se controla mejor la calidad del producto final.

# Etapa 4: Recolección y Análisis de Requerimientos

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

La técnica utilizada fue la entrevista y observación directa con los principales stakeholders *(funcionarios de biblioteca y estudiantes)*. La entrevista permitió obtener información detallada sobre las necesidades y problemas actuales en el sistema de reservas, mientras que la observación directa en la biblioteca evidenció la ineficiencia del sistema presencial y la necesidad de modernizar el proceso. Estos datos se complementaron con la elaboración del *DER* (Documento de Especificación de Requerimientos), que se explica de manera formal las funcionalidades y limitaciones del sistema.

# 4.2 Requisitos Funcionales y No funcionales.

**Funcionales:**

| **ID** | **DESCRIPCIÓN** | **PRIORIDAD** | **JUSTIFICACIÓN** |
| --- | --- | --- | --- |
| **RF-001** | El sistema debe permitir la creación de usuarios, ya sean estudiantes o administradores en la página. | Alta | Permite registrar a estudiantes y administradores, asegurando que solo usuarios autorizados puedan acceder al sistema y garantizando la trazabilidad de las acciones realizadas. |
| **RF-002** | Debe permitir que administradores del sistema puedan crear, registrar y activar nuevos cubículos. | Alta | La administración debe poder configurar y actualizar los cubículos disponibles, lo que asegura un control efectivo de los recursos físicos de la biblioteca. |
| **RF-003** | El sistema debe gestionar el acceso a las funcionalidades basado en el rol con que se registró (estudiante o administrador). Es para controlar las acciones que cada usuario puede realizar. | Alta | Diferenciar los permisos según rol (estudiante/administrador) evita acciones indebidas y mantiene la seguridad y orden en el uso del sistema. |
| **RF-004** | El sistema debe permitir que los usuarios realicen reservas de cubículos de forma online con selección de fecha, hora y número de cubículo (identificación). | Alta | Centraliza el proceso de reserva, facilitando a los estudiantes el acceso a los espacios y optimizando el uso de los cubículos. |
| **RF-005** | Debe mostrar un mensaje de confirmación en la pantalla. | Media | Brinda certeza al usuario sobre la acción realizada, mejorando la usabilidad y reduciendo dudas o reservas duplicadas. |
| **RF-006** | El sistema debe permitir a los administradores gestionar las reservas realizadas, incluyendo modificar o cancelar. | Alta | Los administradores requieren control sobre las reservas para resolver errores, cancelar solicitudes indebidas o ajustar la disponibilidad de cubículos. |
| **RF-007** | El sistema debe generar un reporte de uso de cubículos que muestre la cantidad de reservas realizadas por usuario y el tiempo total de uso durante un período de tiempo seleccionado. | Media | Generar reportes permite evaluar la demanda y frecuencia de uso, entregando datos valiosos para la planificación y mejora de los servicios de la biblioteca. |

**No funcionales:**

| **ID** | **DESCRIPCIÓN** | **PRIORIDAD** | **JUSTIFICACIÓN** |
| --- | --- | --- | --- |
| **RNF-001** Rendimiento | El sistema debe responder a las consultas de los usuarios en un máximo de 4 segundos bajo condiciones normales de uso. | Alta | Un tiempo de respuesta inferior a 4 segundos garantiza fluidez y eficiencia, evitando frustración en un entorno de alta demanda. |
| **RNF-002** Compatibilidad | El sistema debe ser compatible con diferentes navegadores (chrome, firefox, entre otros). | Alta | La compatibilidad entre navegadores asegura el acceso uniforme de todos los usuarios, sin fallos de visualización o funcionamiento. |
| **RNF-003** Integridad de datos | El sistema no debe permitir que un mismo usuario reserve dos o más cubículos de la vez y tampoco permitir que un cubículo sea reservado por dos usuarios simultáneamente. | Alta | Evitar reservas duplicadas mantiene la confiabilidad del sistema y asegura un uso justo de los cubículos. |
| **RNF-004** Limitaciones de uso | El sistema debe limitar la duración máxima continua de reservas de cubículo por estudiante a 3 horas (dos bloques). | Alta | Restringir la duración de reservas optimiza la rotación de espacios y promueve la equidad entre los estudiantes. |

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

El sistema garantizará la protección de los datos personales mediante:

* Autenticación con RUT y contraseña alfanumérica encriptada.
* Roles diferenciados (estudiante/administrador) que controlan las acciones permitidas.
* Control de logs y auditoría de accesos y reservas.
* Políticas de acceso a la base de datos para evitar accesos no autorizados.

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

Además de los requisitos técnicos, se consideran aspectos de interacción y actores:

Los funcionarios deben tener una interfaz clara para administrar reservas y generar reportes.

Los estudiantes deben disponer de un sistema rápido y accesible desde cualquier navegador.

La experiencia de usuario debe ser accesible y simple, de esta manera, se reduce la carga operativa de la biblioteca.

# 4.5 Priorización de Requerimientos.

Los requisitos fueron priorizados en función de su impacto en el uso del sistema:

**Alta prioridad:** creación de usuarios, autenticación, gestión de cubículos, reservas, control de roles y disponibilidad en tiempo real.

**Media prioridad:** reportes estadísticos, mensajes de confirmación y notificaciones.

**Baja prioridad:** personalización avanzada de políticas administrativas o de diseño de la página.

# 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 6: 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.

6.1 Lenguajes y Herramientas Utilizados.

6.2 Implementation de Módulos Principales.

6.3 Problemas y Soluciones Durante la Implementación.

6.4 Gestión de versiones (Git/CI/CD).

6.5 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.