**UNIVERSIDAD NACIONAL DE INGENIERÍA**

**FACULTAD DE INGENIERÍA INDUSTRIAL Y DE SISTEMAS**



**ESCUELA DE INGENIERÍA DE SISTEMAS**

**Curso:**

**PROGRAMACIÓN ORIENTADA A OBJETOS**

**Proyecto:**

**Sistema de de control de pedidos en un taller de manufactura**

**Autores:**

Chicmana Arias, Piero Alejandro ([orcid.org/0009-0009-7963-6901](https://orcid.org/0009-0009-7963-6901))

**Docente:**

Mag. Coronel Castillo, Eric Gustavo ([orcid.org/0000-0003-0494-5629](http://orcid.org/0000-0003-0494-5629))

**LIMA – PERÚ**

**2025**

# DEDICATORIA

A nuestro querido profesor,

Este proyecto, más allá de ser un simple conjunto de líneas de código, es el reflejo de todo lo que hemos aprendido bajo su guía. Cada decisión tomada, cada algoritmo implementado y cada validación diseñada ha sido posible gracias a la base sólida que nos ha proporcionado. En un mundo que avanza a la velocidad de la inteligencia artificial, donde la tecnología redefine constantemente lo posible, usted ha sido nuestro mentor, enseñándonos no solo el "cómo", sino también el "por qué" de la programación.  
“DIVIDE Y VENCERÁS”

# ÍNDICE GENERAL

Pág.

[**DEDICATORIA 3**](#_62hdyorlex4y)

[**ÍNDICE GENERAL 4**](#_nvute62x0glr)

[**RESUMEN 6**](#_noj72wjnlow2)

[**ABSTRACT 7**](#_gztgf4ddh5qj)

[**INTRODUCCIÓN 8**](#_qkclk9f0ounn)

[**OBJETIVOS 9**](#_fr09bydjg5di)

[OBJETIVO GENERAL 9](#_mpb9g2epbduc)

[OBJETIVOS ESPECÍFICOS 9](#_ubmp8h7ex6ql)

[**REQUERIMIENTOS 10**](#_85erlkxo41g1)

[REQUERIMIENTOS FUNCIONALES 10](#_ikhrmq5b5561)

[REQUERIMIENTOS NO FUNCIONALES 11](#_wbc6msbu2zu0)

[ALCANCES 11](#_7o8bd5lb83fq)

[LÍMITES 13](#_wa38pd2427m1)

[**ANÁLISIS DE REQUERIMIENTOS 15**](#_63766m9oc0m2)

[IDENTIFICCIÓN DE ACTORES 15](#_k1xe1j9vtle)

[CASOS DE USOS 15](#_7zi0bz709gux)

[DIAGRAMA DE CASOS DE USO 16](#_8kgq9knq105k)

[**CU01 – MATRÍCULA 17**](#_e65k0nljhpr)

[ESPECIFICACIÓN DEL CASO DE USO 17](#_n650vzml65xn)

[ENFOQUE DE LA CAJA NEGRA 18](#_ld8ynnawgub)

[DIAGRAMA DE SECUENCIA 19](#_w6ggr9p0bfka)

[MODELO DE BASE DE DATOS 20](#_ja1aokbuhd1y)

[CRITERIOS DE ACEPTACIÓN 20](#_36ruqav52n02)

[**CU02 – REGISTRAR PAGO DE CUOTA 23**](#_jfv4a6p107r2)

[ESPECIFICACIÓN DEL CASO DE USO 23](#_np78gg3nf0ke)

[CAJA NEGRA 24](#_szxxn7g47xsw)

[DIAGRAMA DE SECUENCIA 25](#_pfwvd240w5ox)

[MODELO DE BASE DE DATOS 26](#_cyddj6d6qs6s)

[CRITERIOS DE ACEPTACIÓN 26](#_wdgrtl5jtesr)

[**CU03 – CONSULTAS 29**](#_g0y892dc7xcn)

[ESPECIFICACIÓN DEL CASO DE USO 29](#_1jhbbsrwctsu)

[CAJA NEGRA 31](#_j05wu6jw805b)

[MODELO DE BASE DE DATOS 31](#_t6y77xvzseq5)

[CRITERIOS DE ACEPTACIÓN 32](#_iyw6hr6up60w)

[**MODELO DE BASE DE DATOS 35**](#_64drk7ogh0uu)

[MODELO RELACIONAL 35](#_zc8imojitt9)

[DICCIONARIO DE DATOS 35](#_fuoy9vgvhpa2)

[**IMPLEMENTACIÓN DE LA BASE DE DATOS 40**](#_rrpsmxejxfvt)

[CREACIÓN DE LA BASE DE DATOS 40](#_7r6e1zhkpd7b)

[CREACIÓN DE LOS OBJETOS DE LA BASE DE DATOS 40](#_ajd3hss47j1)

[CARGAR DATOS DE PRUEBA 45](#_wq3semlx4ng)

[**CÓDIGO FUENTE 61**](#_f19cwvsxpqp4)

[SERVICIO DE MATRICULA\_CU01 61](#_kcxwrvbzgdoq)

[SERVICIO DE PAGOS\_CU02 69](#_cv1bajk3gdj1)

[SERVICIO DE CONSULTAS\_CU03 75](#_2lk6zuxrenuu)

[**CONCLUSIONES 81**](#_cyfyvzy3d6rg)

[**LECCIONES APRENDIDAS 83**](#_9m755c3743af)

[**BIBLIOGRAFÍA 84**](#_vn80n13a6erp)

[**ANEXOS 85**](#_bs8dvma026ng)

[Anexo 1: Titulo 86](#_bs8dvma026ng)

[Anexo 2: Titulo 87](#_bs8dvma026ng)

# RESUMEN

Este documento describe el desarrollo del backend para un sistema de gestión escolar dirigido a colegios de primaria. El sistema, implementado utilizando Spring Boot y SQL Server, está diseñado para gestionar tres áreas clave: la matrícula estudiantil, los pagos de los estudiantes y la asignación de profesores, con funcionalidades específicas como la visualización de horarios de los docentes. A través de la integración de servicios backend, se garantiza una gestión eficiente y precisa de los registros, validaciones y operaciones asociadas a cada uno de estos procesos. Además, se implementaron controles y validaciones a nivel de base de datos para asegurar la integridad y coherencia de la información, incluyendo la verificación de pagos, matrículas y asignación de profesores a los cursos. El sistema permite a los administradores gestionar los pagos de matrícula y visualizar horarios de los profesores de manera sencilla y confiable.

**Palabras clave:** Spring Boot, SQL Server, Desarrollo Backend, Gestión Escolar Primaria, Matrícula Estudiantil, Procesamiento de Pagos, Asignación de Profesores, Visualización de Horarios, Validación de Base de Datos, Sistema Educativo.

# ABSTRACT

This document describes the development of the backend for a school management system aimed at primary schools. The system, implemented using Spring Boot and SQL Server, is designed to manage three key areas: student enrollment, payments, and teacher assignments, with specific functionalities such as visualization of teacher schedules. Through the integration of backend services, efficient and accurate management of records, validations, and operations associated with each of these processes is ensured. Additionally, database-level controls and validations were implemented to ensure the integrity and consistency of the information, including payment, enrollment, and teacher course assignment verifications. The system allows administrators to manage tuition payments and view teacher schedules in a simple and reliable way.

**Keywords:** Spring Boot, SQL Server, Backend Development, Primary School Management, Student Enrollment, Payment Processing, Teacher Assignment, Schedule Visualization, Database Validation, Educational System.

# INTRODUCCIÓN

El proyecto desarrollado tiene como objetivo la creación de un sistema…

La implementación del sistema se realizó utilizando Spring Boot como framework principal para el desarrollo de los servicios backend y SQL Server como sistema de gestión de bases de datos. La arquitectura del sistema permite una interacción fluida con el usuario a través de diversas funcionalidades que garantizan la correcta gestión de datos y la eficiencia en los procesos de pedidos de manufactura.

…

Este documento detalla las principales características del backend desarrollado, los requerimientos tanto funcionales como no funcionales, y la implementación de cada uno de los casos de uso más relevantes. Además, se incluye una descripción de la base de datos utilizada, la creación y gestión de los objetos relacionados con el sistema, y los criterios de aceptación que garantizan la correcta implementación del mismo.

# OBJETIVOS

## OBJETIVO GENERAL

Desarrollar un sistema de control de pedidos que permita digitalizar el registro, seguimiento y gestión de órdenes de trabajo en el taller de manufactura “Mecafab”, con el fin de mejorar la planificación, reducir errores por registros manuales y optimizar el control del estado de cada pedido desde su solicitud hasta su entrega final.

## OBJETIVOS ESPECÍFICOS

* **Implementar un módulo de registro de pedidos**, que permita ingresar especificaciones detalladas de cada orden, generando un identificador único y asegurando su almacenamiento correcto en el sistema.
* **Diseñar la funcionalidad para asignar pedidos** a operarios o equipos de trabajo, permitiendo registrar al responsable y actualizar el estado del pedido a “En producción”.
* **Permitir la actualización del estado de los pedidos**, reflejando los distintos avances del proceso productivo, incluyendo estados como “En proceso”, “En espera de material” o “Completado”.
* **Desarrollar un sistema de alertas automáticas**, que notifique al jefe de taller sobre pedidos próximos a vencer o ya retrasados, para facilitar la toma de decisiones oportunas.
* **Incorporar un módulo de cierre y entrega del pedido**, que registre la entrega final al cliente y genere documentación resumida del trabajo realizado, archivando correctamente el pedido como finalizado.

# REQUERIMIENTOS

## REQUERIMIENTOS FUNCIONALES

**Registrar Pedidos**

* —
* —

**Asignar Pedidos**

* —
* —
* —

**Actualizar Estado del Pedido**

* —
* —

**Alertas de Vencimientos o Retrasos**

* —
* —
* —

**Registrar Entrega Final del Pedido y Generar Boleta**

* **—**

## REQUERIMIENTOS NO FUNCIONALES

**Usabilidad**

El sistema debe contar con una **interfaz amigable y accesible** para usuarios que no son expertos en informática, como operarios o jefes de taller. Las pantallas deben ser simples e intuitivas: listas de pedidos con botones visibles para cambiar estados y formularios claros para registrar nuevos pedidos, reduciendo al mínimo la posibilidad de error humano.

**Eficiencia**

Dado que el taller maneja un volumen moderado de pedidos (decenas simultáneamente), el sistema debe **responder sin demoras perceptibles**. Las consultas, filtros y actualizaciones deben ejecutarse en tiempo real para apoyar la toma de decisiones del jefe de taller.

**Confiabilidad**

La aplicación debe garantizar la **consistencia y persistencia de los datos**, incluso ante cierres inesperados. Los pedidos no deben perderse ni corromperse. Se deben implementar **copias de respaldo automáticas** —por ejemplo, cada noche— para asegurar la recuperación ante fallos graves.

**Seguridad**

El sistema debe contar con **mecanismos de autenticación** y control de roles. Solo los supervisores podrán realizar acciones críticas como eliminar pedidos o marcar entregas finales, mientras que los operarios solo podrán modificar estados intermedios. Esto previene accesos no autorizados y mantiene la integridad de la información.

**Mantenibilidad**

El sistema debe ser fácil de modificar y escalar. Agregar nuevos estados de pedido o ajustar el flujo de trabajo no debe requerir reescribir el sistema desde cero. Se recomienda una arquitectura modular y una buena documentación del código para facilitar futuras mejoras.

## LÍMITES

**Limite01**

* —

**Limite02**

* —

**Restricciones 01**

* —
* —

# ANÁLISIS DE REQUERIMIENTOS

## IDENTIFICCIÓN DE ACTORES

| **CÓDIGO** | **NOMBRE** | **DESCRIPCIÓN** |
| --- | --- | --- |
| **ACT01** | Jefe de Taller | Puede asignar el pedido a un operario o equipo específico para su fabricación; el sistema registra el responsable asignado. |
| **ACT02** | Empleado | Pueden marcar un pedido con estados intermedios relevantes (por ejemplo, “En proceso”, “En espera de material”, “En acabado”, “Completado”). |

## CASOS DE USOS

| **CÓDIGO** | **ACTOR** | **NOMBRE** | **DESCRIPCIÓN** |
| --- | --- | --- | --- |
| **CU01** | Jefe de Taller | Registrar Pedido | Permite asignar un pedido a un empleado. |
| **CU02** | Empleado | Cambiar Estado del Pedido | Puede cambiar el estado del pedido. |

## 

## 

# CU01 – Registrar y Asignar Pedido

## ESPECIFICACIÓN DEL CASO DE USO

| **Código** | **CU01** | |
| --- | --- | --- |
| **Nombre** | Registrar y asignar un pedido | |
| **Descripción** | Permite al usuario registrar un pedido y asignarlo a un empleado. | |
| **Actores** | **Jefe de taller** | |
| **Precondición** | El empleado debe existir | |
| **Secuencia normal** | **Paso** | **Acción** |
| **1** | — |
| **2** | — |
| **3** | — |
| **4** | — |
| **5** | — |
| **6** | — |
| **7** | — |
| **8** | — |
| **Postcondición** | — | |
| **Excepciones** | **Paso** | **Acción** |
| **2** | … |
| **4** | … |
| **5** | … |
| **6** | … |
| **7** | … |
| **Comentarios** | … | |

## 

## MODELO DE BASE DE DATOS

(IMAGEN DE lA PARTE DE DEL PEDIDO DE LA DB DIAGRAMA)

## CRITERIOS DE ACEPTACIÓN

#### 1. Validaciones Previas

* **Validar Pedido:**
  + —.
* **Validar Cliente:**
  + —
* **—**

#### 2. Registro de Pedido

* **—**
  + —

#### 3. Asignación de Pedido a Empleado

* **—**
  + —

# CU02 – Cambiar Estado del Pedido

## ESPECIFICACIÓN DEL CASO DE USO

| **Código** | **CU02** | |
| --- | --- | --- |
| **Nombre** | **Cambiar Estado del Pedido** | |
| **Descripción** | … | |
| **Actores** | **Empleado** | |
| **Requerimiento asociado** | … | |
| **Casos de uso asociados** | **CU01:** Registrar Pedido(Ya que aquí se asigna al empleado). | |
| **Precondición** | * El empleado debe estar autenticado en el sistema. * — | |
| **Secuencia normal** | **Paso** | **Acción** |
| **1** | — |
| **2** | — |
| **3** | — |
| **4** | — |
| **5** | — |
| **6** | — |
| **7** | — |
| **8** | — |
| **9** | — |
| **Excepciones** | **Paso** | **Acción** |
| **1** | — |
| **2** | — |
| **3** | — |
| **4** | — |
| **5** | — |
| **6** | — |
| **Comentarios** | * — * — | |

## 

## MODELO DE BASE DE DATOS

(IMAGEN DE ESA PARTE DE LA DB)

## CRITERIOS DE ACEPTACIÓN

### **Validación del Usuario**

* **Descripción:** El sistema debe verificar que el usuario que realiza la consulta está registrado en la base de datos.
* **Criterios:**
  + El sistema debe mostrar un mensaje de error si el usuario no está registrado: *"Usuario no registrado en el sistema."*.
  + Esta validación aplica tanto para profesores como para alumnos, pero no se hace diferenciación explícita de roles en esta versión del sistema.

### **Consulta de Cronograma de Pagos (Alumno)**

* **Descripción:** El sistema debe permitir consultar el cronograma de pagos de un usuario, considerando que los datos consultados serán únicamente de alumnos.
* **Criterios:**
  + El cronograma debe incluir todas las cuotas pendientes y pagadas.
  + Los datos deben mostrarse en un formato estructurado: *Cuota, Monto a Pagar, Fecha de Pago*.
  + Si el alumno no está matriculado, el sistema debe mostrar el mensaje: *"El alumno no está matriculado."*.

### **Consulta de Horarios (Estudiantes y Profesores)**

* **Descripción:** El sistema debe permitir la consulta de horarios asignados para estudiantes (sección) y profesores (cursos y secciones asignadas).
* **Criterios:**
  + Los horarios deben incluir información sobre días, horas y cursos correspondientes.
  + Si el usuario no tiene horarios asignados, el sistema debe mostrar el mensaje: *"No se han encontrado horarios asignados."*.
  + La información debe consolidarse y mostrarse en una única tabla para estudiantes y profesores, sin separación explícita.

### **Consolidación y Presentación de Datos**

* **Descripción:** La información obtenida en cualquier consulta debe estar consolidada y organizada para una presentación clara al usuario.
* **Criterios:**
  + El sistema debe mostrar los datos en un formato visualmente comprensible (tablas en la interfaz o estructuras JSON para el backend).
  + La información debe estar ordenada cronológicamente en el caso de pagos y horarios.
  + Los encabezados de las tablas deben ser claros y representativos de los datos mostrados.

### **Manejo de Errores y Excepciones**

* **Descripción:** El sistema debe manejar errores y excepciones de manera que no afecte la experiencia del usuario ni la integridad del sistema.
* **Criterios:**
  + El sistema debe mostrar mensajes de error amigables al usuario en caso de fallas o datos inexistentes.
  + Ejemplo de mensajes:
    - *"Alumno no existe."*
    - *"No se encontraron horarios asignados."*
    - *"Sección no encontrada."*
  + Los mensajes de error no deben exponer detalles técnicos del sistema.

# MODELO DE BASE DE DATOS

## MODELO RELACIONAL

(IMAGEN DE LA DB COMPLETA)

# IMPLEMENTACIÓN DE LA BASE DE DATOS

## CREACIÓN DE LA BASE DE DATOS

# CONCLUSIONES

**Evaluación General del Proyecto**:

* El proyecto logró establecer una base sólida para la gestión escolar a nivel primario, permitiendo optimizar procesos clave como la matrícula, pagos y asignación de horarios.
* La implementación con Spring Boot y SQL Server garantiza escalabilidad y mantenimiento futuro.

**Limitaciones y Futuro Desarrollo**:

* Aunque se ha cumplido con los objetivos principales, el sistema presenta áreas de mejora, como la inclusión de funciones para el registro directo de estudiantes y el manejo avanzado de reportes.
* Se recomienda priorizar la integración de un frontend y la automatización de procesos más complejos en futuras iteraciones.

**Impacto Tecnológico**:

* Este sistema puede servir como base para expandirse a otras instituciones educativas, integrando niveles superiores y funcionalidades adicionales.

**Importancia de la Validación de Datos**:

* La implementación de validaciones y controles asegura la integridad de los datos, lo que es esencial en entornos educativos.

# LECCIONES APRENDIDAS

**Importancia de los Requerimientos Iniciales**:

* La claridad en la recopilación y análisis de requerimientos es fundamental para el éxito del proyecto.

**Impacto de la Documentación**:

* La documentación técnica detallada ayuda a garantizar la mantenibilidad del sistema y facilita futuras mejoras.

**Gestión de Alcances**:

* Definir un alcance realista desde el inicio evita sobrecarga de trabajo y garantiza entregables de calidad.

**Validaciones y Seguridad**:

* Incorporar validaciones a nivel de base de datos y backend refuerza la seguridad y coherencia del sistema.

**Aprendizaje Colaborativo**:

* El proyecto destaca la importancia del trabajo en equipo y la distribución efectiva de tareas.

# BIBLIOGRAFÍA

Creately. (2022, 21 de octubre). Tutorial del diagrama de secuencia: Guía completa con ejemplos. <https://creately.com/blog/es/diagramas/tutorial-del-diagrama-de-secuencia/>

Daniel. (2023, 30 de octubre). SQL: Todo lo que necesitas saber sobre el lenguaje de programación de bases de datos. *Formación en Ciencia de Datos | DataScientest.com*. <https://datascientest.com/es/sql-todo-lo-que-necesitas-saber-sobre-el-lenguaje-de-programacion-de-bases-de-datos>

GeeksforGeeks. (2024, 1 de enero). How to send different types of requests (GET, POST, PUT, DELETE) in Postman. *GeeksforGeeks*. <https://www.geeksforgeeks.org/how-to-send-different-types-of-requests-get-post-put-delete-in-postman/>

Introducing EDrawMax 10. (n.d.). Edrawsoft. <https://www.edrawsoft.com/es/er-diagram/>

Lucid Software Inc. (2024). *Lucid.app* [Herramienta en línea]. Lucid Software Inc. <https://www.lucid.app>

Microsoft. (2024). *SQL Server Management Studio* (versión 20) [Herramienta de gestión de bases de datos]. Microsoft. <https://learn.microsoft.com/en-us/sql/ssms/>

OpenWebinars. (2019, 28 de noviembre). *CÓMO INSTALAR y USAR LOMBOK- TUTORIAL EN ESPAÑOL* [Video]. YouTube. <https://www.youtube.com/watch?v=EDGTS8SwzPA>

VMware. (2024). *Spring Tools Suite 4* (versión X.X) [Software de desarrollo]. VMware. <https://spring.io/tools>