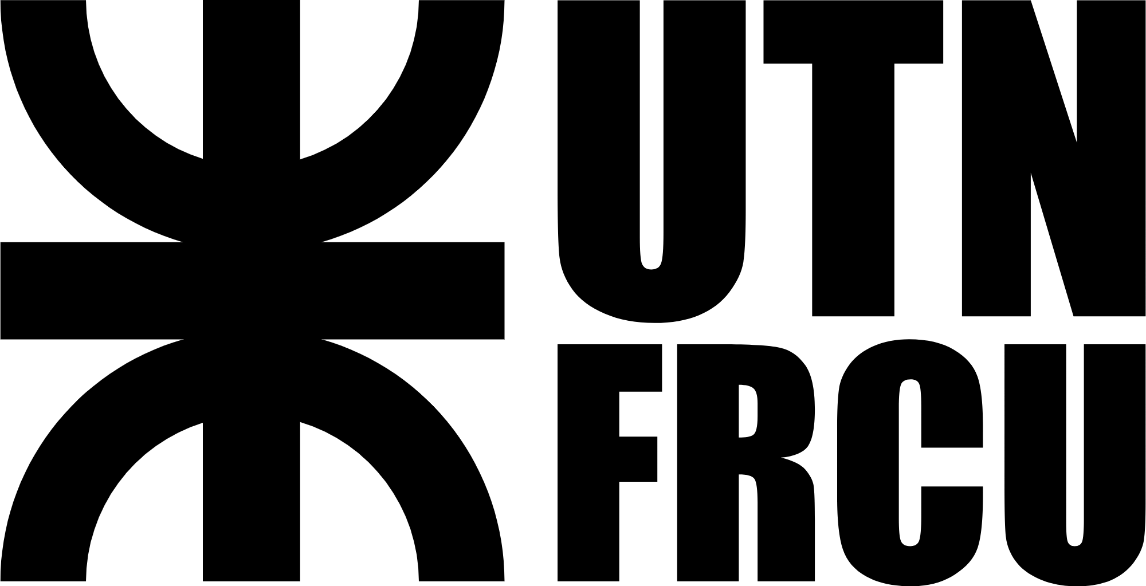
~~~~

**Universidad Tecnológica Nacional**

**Facultad Regional Concepción del Uruguay**

| **Sistema de gestión avícola** |
| --- |

**Ingeniería en Sistemas de Información**

**Docentes:**

Luis Enrique Rodriguez Prudenza

Jorgelina Cecilia Nadal

Dario Alejandro Dartuqui

**Tutor:**

Ing. Joaquin Rene Suarez

**Grupo 3:**

Francisco Dutra

Manuel Callegari

Marcos Retamero

Gonzalo Silveyra

Resumen

Abstract

**Índice**

[**Capítulo I - Descripción de la empresa 7**](#_9syubqr5uiuz)

[I.1 - Justificación 7](#_wvxsycm0fzra)

[**Capítulo II - Objetivos, alcance y requerimientos 9**](#_shp3fk5v0f39)

[II.1 - Objetivos 9](#_9x6fqvp0d4d3)

[II.1.1 - Objetivo general 9](#_188zy2h4tiic)

[II.1.2 - Objetivos específicos 9](#_qqp0eskmprp0)

[II.1.3 - KPI 9](#_9gos4w8pqmkw)

[II.2 - Alcance 10](#_5axw5s7itngf)

[II.3 - Requerimientos 12](#_fwxj80a6a529)

[II.3.1 - Requerimientos funcionales 12](#_vcq8hteclsvj)

[II.3.2 - Requerimientos no funcionales 13](#_k52k6zq4gb68)

[II.3.3 - Requerimientos de experiencia de usuarios (UX) 13](#_ahzamgc5y32m)

[II.4 - Resultados esperados 15](#_hrs0xxrtwbai)

[**Capítulo III - Factibilidad del proyecto 17**](#_xz9fosvvp7uh)

[III.1 - Factibilidad técnica 17](#_qp7im8dxzdov)

[III.1.1 - Recursos tecnológicos 17](#_x0p1nfq3fek9)

[III.1.2 - Recursos humanos 17](#_z2o22n3wuwi4)

[III.1.3 - Infraestructura 18](#_4mcuj9nx29g1)

[III.1.4 - Viabilidad de la Implementación 18](#_q3r6ztvvn13h)

[III.2 - Factibilidad económica 18](#_wwltinext1nz)

[III.2.1 - Beneficios esperados del proyecto 19](#_if90ptn6fntq)

[III.3 - Factibilidad financiera 19](#_ob37rrwpqnqn)

[III.3.1 - Inversión inicial 19](#_wjd4w4ohf2re)

[III.3.2 - Flujo de caja y financiamiento 21](#_t2ah5hwelknc)

[III.3.3 - Retorno de inversión (ROI) 21](#_x00seqgox7xd)

[III.3.4 - Plan de pagos 21](#_mardcimws7tz)

[III.4 - Conclusión del análisis de factibilidad 23](#_rcbf444nda7g)

[**Capítulo IV - Marco Legal y Normativa Ambiental 24**](#_69ll58grrqqi)

[IV.1 - Ley General del Ambiente (Ley Nacional 25.675) 24](#_hw8xs5do7344)

[IV.2 - Clasificación del Impacto Ambiental 24](#_288xmpkmxys)

[IV.3 - Normativa Provincial en Entre Ríos: Decreto 4977/09 25](#_trnn38cf2l6t)

[IV.4 - Clasificación del Proyecto 25](#_6fz1pfh4d2pd)

[**Capítulo V – Marco Teórico 26**](#_aa80aqgmvbx1)

[V.1 - Industria Avícola Argentina: Relevancia Regional y Nacional 26](#_z2j3uo3d6ohj)

[V.2 - Factores Críticos del Bienestar Animal en la Producción Avícola 26](#_iuckiq9zpcmo)

[V.3 - Transformación Digital e IoT en el Manejo Avícola 27](#_jitsjmttsf04)

[V.4 - Análisis de Datos y Sistemas de Alerta Temprana 27](#_ht0siiq7w8w3)

[V.5 - Ingeniería de Sistemas: Enfoque Sistémico para una Producción Inteligente 28](#_k0nim9c47ayn)

[V.6 - Avicultura de Precisión y Tendencias en Agricultura 4.0 28](#_buxboso42i8y)

[**Capítulo VI - Metodología de resolución 30**](#_22tjo8h1mved)

[VI.1 - Introducción a la metodología 30](#_uoooxzemq0uo)

[VI.2 - Estructura general 32](#_mj8goctnqxt1)

[Duración de cada sprint 32](#_ndpj872d7jrj)

[Equipo 32](#_dd0o8gy4xsxi)

[Modelo de gobernanza y toma de decisiones 32](#_fkj662ohux1r)

[Plan de comunicación y seguimiento 33](#_48rryizdtzjy)

[VI.3 - Dinámica Scrum + Kanban 34](#_rr2hivazela)

[**Capítulo VII - Planificación 35**](#_etskx28t9q59)

[VII.1 - Métricas de rendimiento y flujo de trabajo 35](#_78wma7y65zeh)

[VII.2 - Actividades a realizar 36](#_kx0ma1a8mjwd)

[VII.3 - Sprints 38](#_xhx5ldgmk08)

[Sprint 1: Inicio y Organización 38](#_slngg59enhfx)

[Sprint 2: Manejo de Usuarios y Seguridad Básica 39](#_bhhet7rjx067)

[Sprint 3: Estructura de Datos y Configuración de Galpones 40](#_cbmnpt9vu1hq)

[Sprint 4: Gestión de Camadas y Datos Productivos (Parte 1) 41](#_f7gd8w18x5oj)

[Sprint 5: Gestión de Camadas y Datos Productivos (Parte 2) 42](#_qqsxz3mk22xb)

[Sprint 6: Integración IoT Básica (Lectura de Sensores) 43](#_5b2remxbw4bb)

[Sprint 7: Alertas y Tablero Ambiental 44](#_7t9zpuscpoj3)

[Sprint 8: Control Remoto de Dispositivos 44](#_ny7cj7zb7d13)

[Sprint 9: Reportes Históricos y Auditoría 45](#_e9s89xtoy0q8)

[Sprint 10: Pruebas de Integración (Parte 1) 46](#_8skswganywya)

[Sprint 11: Pruebas de Integración (Parte 2) y Ajustes Finales 47](#_9e94wdf98fno)

[Sprint 12: Cierre, Documentación y Puesta en Producción 48](#_jwk4y2kjto3)

[Diagrama de Gantt - Cronograma de Sprints 49](#_226fu1n4hq3a)

[VII.4 - Revisiones 49](#_t52wvi1txarv)

[Revisiones técnicas de ingeniería de sistemas 49](#_11kndr6xmbh6)

[**Capítulo VIII - Análisis de riesgos 50**](#_l8daqynzejk5)

[VIII.1 - Plan de Gestión de Riesgos 50](#_mf677k8ntjl1)

[VIII.1.1 - Enfoque a Utilizar 50](#_3nvhwd4nw0tm)

[VIII.1.2 - Categorización 51](#_ubo5btp6syq3)

[VIII.1.3 - Formatos de Documentos 52](#_3dmfm3rjkzlz)

[VIII.1.4 - Escala de Medición para Probabilidades 52](#_yd11p34efnmd)

[VIII.1.5 - Escala de Medición para Impactos 53](#_ycohd3pq7pyt)

[VIII.2 - Identificación de los Riesgos 53](#_ivp2xyuymoue)

[VIII.2.1 - Actividades para la Identificación de Riesgos 53](#_7m75w9sayyai)

[VIII.2.2 - Análisis de Bowtie 54](#_ekb0ckszjm4n)

[VIII.2.3 - Análisis de Supuestos 54](#_vgw6xov5s6yo)

[VIII.2.4 - Registro Preliminar de Riesgos 55](#_6pyj9s6lv80l)

[VIII.3 - Análisis Cualitativo del Riesgo 61](#_brquqna1rikp)

[VIII.4 - Análisis de Bowtie 65](#_gah6vxv9l7yf)

[VIII.4.1 - Justificación de la metodología 65](#_8yqwgvj85ppc)

[VIII.4.1 - Aplicación al sistema 66](#_wsmyqkqowhlf)

[VIII.5 - Análisis Cuantitativo del Riesgo 67](#_2kz8gfgewn0f)

[VIII.6 - Planificación de Respuestas y Plan de Contingencia de los Riesgos 67](#_32yao4gx5s7g)

[VIII.7 - Plan de Monitoreo y Control de Riesgos 70](#_9jl3via3k80u)

[Objetivos del monitoreo 70](#_267110inkr3m)

[Herramientas y mecanismos utilizados 70](#_mos8w7bto7eq)

[Participación y responsabilidades 71](#_1t244r642kh8)

[VIII.8 - Conclusión 71](#_sknl0ygky327)

[**Capítulo IX: Diseño 73**](#_jt1bf7vx727w)

[IX.1 - Definición de Reglas de Negocio 73](#_xt2gldaj5ags)

[IX.2 - Casos de uso 76](#_mfkbmuptru1q)

[CU1 – Iniciar Sesión 77](#_ehuhe6o7f8o9)

[CU2 – Visualizar Tablero Principal 79](#_upv5tchdkhez)

[CU3 – Configurar y Ajustar Rango de Alertas 80](#_9sqm1ubtfpsd)

[CU4 – Control Remoto de Dispositivos Climatizadores 82](#_xmgouuuoixun)

[CU5 – Registrar Nueva Camada 83](#_xnn4xg4w6tv2)

[CU6 – Registrar Muertes de Aves 85](#_9ikeus79nfyu)

[CU8 – Registrar Pesos de Aves 86](#_rjwck8rfm2l1)

[CU8 – Generar Informes y Visualizar Datos Históricos 88](#_s8bvcdk1pfy)

[CU9 – Gestionar Alertas y Notificaciones 89](#_xlizdr4zwtzv)

[CU10 – Registrar Acciones de Usuario en Bitácora 90](#_hjzh582dqn24)

[CU11 – Manejo de Roles y Usuarios 92](#_rz8xsk2sehdp)

[CU12 – Cerrar Sesión (Logout) 93](#_xbutkku9ilkt)

[CU13 – Envío de Resumen Diario a Proveedor 94](#_wioiaqy93ca)

[CU14 – Cerrar Camada al Finalizar la Cría 96](#_ldws7wky403k)

[CU15 – Consultar Reportes Históricos Recibidos por el Proveedor 98](#_255aldj7bs0c)

[CU16 – Consultar la Bitácora del Sistema 99](#_wp1vboa16v5m)

[CU17 – Registrar Visitas Veterinarias y Observaciones Sanitarias 101](#_weid6jofcgcr)

[CU18 – Consultar Información de Camadas Activas o Finalizadas 102](#_cb8ngc9h4ad)

[CU19 – Visualizar Dashboard de Galpón 103](#_n6ed55uhuc3i)

[IX.3 - Diagramas de secuencia 106](#_tgo5fgz2frw4)

[**Capítulo X – Enfoque de Arquitectura Modular: Monolito Modular 118**](#_qguv37d0at9f)

[X.1 - Introducción al Enfoque Modular 118](#_se13lnk8rfoh)

[X.1.2 - ¿Qué es un Monolito Modular? 118](#_mtthw45i2yqu)

[X.1.3. Aplicación del Enfoque Modular al Sistema Actual 119](#_c5pt15ypg6ls)

[X.1.4. Fundamentos Teóricos 120](#_m71q6qb35g7u)

[X.1.5. Conclusión 120](#_bo1er2x4zr50)

[X.2.1 Implementación de AWS 122](#_vcpbfex8bk5x)

[X.2.2. Servicios de AWS utilizados 122](#_lkchr72e8waa)

[a) AWS IoT Core 123](#_z3mcx6t7s3fh)

[b) AWS Lambda 123](#_r2i80i1qqd6f)

[c) Amazon API Gateway 123](#_m0hh0msb3f04)

[d) Amazon RDS (Relational Database Service) 124](#_pgayqj2nmqw7)

[e) Amazon S3 + CloudFront 124](#_4n82dxlc4yqh)

[f) AWS IAM (Identity and Access Management) 124](#_iul6ok93mx4c)

[g) AWS Certificate Manager (ACM) 124](#_5qame459te25)

[X.3.1 Diseño de la Base de Datos 125](#_41w04awhztau)

[X.4.1. Introducción al Backend del Sistema 126](#_frh1fuo7mdr)

[X.5.1. Introducción al Frontend del Sistema 127](#_mj2c52qg1e9u)

[X.5.2. Tecnologías Utilizadas en el Frontend 128](#_gtf5u0133cwn)

[X.5.3 Comunicación con el Backend 128](#_z4cnxqfcgevz)

[X.5.4Autenticación y envío de tokens 129](#_n1jjhvb0t788)

[X.5.5 Manejo de errores y feedback 129](#_smxoq8u8qdej)

[**Capítulo XI – Diseño del sistema 131**](#_frn85eucxpny)

[XI.1. Arquitectura General 131](#_atqi6dq61jne)

[XI.2. Sistema AWS IoT y la Integración con MQTT 132](#_tn9b3326t3lr)

[XI.3. Capitulo de conexión 134](#_b3p5rxks8bbt)

[XI.4. Sensado Ambiental: Monitoreo de Temperatura y Humedad en Galpones 134](#_8voxysqkkqk9)

[XI.5. Gestión de Malacates Eléctricos con Control Proporcional 136](#_whnscubriktn)

[XI.6. Control de Iluminación: Gestión Remota de Luces por Galpón 140](#_m1rbigudx19y)

[XI.7. Sistema de Aspersores: Activación Seccional de Válvulas para Enfriamiento Ambiental 142](#_kghuhcmaaqz4)

[XI.8. Sistema de Ventilación: Control de Ventiladores Trifásicos por Sección 144](#_y7ttepzeuvfu)

[XI.9. Estructura de Comunicación MQTT: Organización de Topics por Funcionalidad y Galpón 147](#_fsbkbss969qj)

[XI.10. Seguridad en el Sistema IoT: Comunicación, Autenticación 148](#_qxs39lx5ge42)

[XI.11. Diccionario 148](#_rsohz266h9yd)

[TLS 148](#_704d6regb7nd)

[GPIO 149](#_bjd9xt6t7sxy)

[Certificado X.509 149](#_vdezkwl8x1sr)

[Conclusión del capítulo 149](#_jzkry2k7r2gq)

[Bibliografía 150](#_gnx71n9en89q)

# Capítulo I - Descripción de la empresa

La empresa elegida para la realización de este proyecto es una empresa familiar ubicada sobre ruta 20, en el paraje Estación Escriña, departamento Gualeguaychú, Entre Ríos. Nuestro comitente en este caso es el dueño y gerente de la empresa, y posee un total de 3 empleados a disposición.

La empresa se dedica a la cría y engorde de pollos desde 2004, cuenta con cuatro galpones en donde pueden alojar hasta 70000 pollos en conjunto. Su proveedor es desde entonces Las Camelias, empresa que les provee las aves recién nacidas y que las retira cuando la crianza finaliza. Además, esta empresa les provee semanalmente el alimento necesario para los silos y el gas natural que utilizan para calentar los galpones, así como también, facilitan un veterinario para el control de salud de los pollos.

## I.1 - Justificación

Por medio de reuniones con el comitente de la empresa y del relevamiento realizado en la infraestructura, pudimos detectar lo siguiente: La empresa sufre la pérdida de una cantidad considerable de aves, actualmente alrededor del 10%, y no cuenta con las herramientas necesarias para analizar las causas y encontrar formas de mitigarlas. Esto repercute principalmente de manera económica, pero también afecta a la confianza con el proveedor, quien ante esta problemática podría buscar otras opciones. Además, ante la ineficiencia en la cría y la entrega de pocas aves al finalizar la crianza, esta falta de stock podría trasladarse a los precios en las góndolas, afectando finalmente al consumidor final.

En este proyecto se busca diseñar un sistema que proporcione al comitente y a sus empleados herramientas e información para mejorar la toma de decisiones, reducir el tiempo de respuesta a problemas y rediseñar su proceso de negocio para lograr una producción mayor y más eficiente.

La importancia de este proyecto radica, en parte, en lo relevante que es la industria avícola en la provincia. Según el gobierno de Entre Ríos, la provincia concentra más del 50% del total de las áreas de producción avícola del país, seguida por Buenos Aires con el 35,5%. Esto posiciona a la región como la productora y exportadora de carne aviar más importante y de mayor impacto de Argentina. La aplicación de este proyecto podría influenciar granjas similares en la región y ser un disparador de su implementación y modernización en ellas, con modificaciones mínimas al ser escalable.

La realización e implementación del proyecto beneficia a todos los involucrados en la cadena productiva, así como a agentes externos al negocio. El productor percibirá mayores ingresos al mejorar el proceso de cría y reducir el fallecimiento de aves, lo cual se traduce en una mayor competitividad en el mercado local e internacional, permitiendo a los productores expandirse y generar más empleo en la cadena de producción. Los empleados serán capacitados para el uso de la tecnología y adquirirán nuevos conocimientos, además de experimentar mejoras en sus condiciones de trabajo. El frigorífico proveedor se beneficiará de un mayor stock, lo que puede derivar en nuevos potenciales clientes y mayores ingresos, otorgándole una ventaja competitiva. El consumidor final también podría verse beneficiado con el aumento de stock de carne aviar, lo cual podría influir positivamente en los precios finales de dicho producto.

## 

## 

# Capítulo II - Objetivos, alcance y requerimientos

## II.1 - Objetivos

### II.1.1 - **Objetivo general**

Mejorar la gestión del proceso de crianza avícola mediante la optimización de la captura, monitoreo y control de variables ambientales clave, reduciendo en un 2% la mortalidad de los pollos y permitiendo a la empresa contar con procesos más eficientes que incrementen su productividad y competitividad en el mercado.

### II.1.2 - Objetivos específicos

* Captar y almacenar en tiempo real los datos de temperatura y humedad de los galpones de cría con una precisión mínima del 90% desde el inicio de la implementación del sistema.
* Mejorar la visualización de información relevante y contextualizada sobre la situación actual de cada galpón, asegurando una precisión del 95% y un retraso máximo de 15 segundos entre la medición y el muestreo de datos.
* Optimizar el registro de información sobre las camadas de aves entrantes y salientes, asegurando una digitalización precisa y estructurada desde el primer mes de funcionamiento.
* Agilizar el control sobre los artefactos climatizadores de los galpones, permitiendo su manipulación remota en un tiempo máximo de 15 segundos desde la ejecución de la orden.
* Mejorar la alerta ante situaciones climáticas fuera de los rangos definidos, garantizando notificaciones en un tiempo máximo de 15 segundos desde la detección del evento.

### II.1.3 - KPI

* **Tasa de precisión de los datos capturados por los sensores**, medida como el porcentaje de coincidencia con valores obtenidos por un sensor de referencia calibrado.
* **Tiempo promedio de actualización de datos en la interfaz**, desde la toma del dato por los sensores hasta su visualización en el sistema.
* **Precisión y completitud de los registros de camadas**, medida por la cantidad de datos ingresados correctamente vs. inconsistencias detectadas.
* **Tiempo promedio de respuesta del sistema para accionar dispositivos**, desde que el usuario envía una acción hasta su ejecución en el dispositivo.
* **Precisión del sistema de alertas**, medida por la tasa de falsos positivos y falsos negativos en la detección de eventos críticos.

## II.2 - Alcance

El alcance de este proyecto queda estrictamente limitado al contexto de una empresa avícola familiar ubicada en el paraje Estación Escriña, departamento Gualeguaychú, provincia de Entre Ríos. La organización cuenta con tres galpones de crianza de pollos, y un equipo reducido de trabajo conformado por su dueño-gerente y tres empleados. Si bien el desarrollo se orienta a esta empresa en particular, el sistema ha sido concebido con una arquitectura que permita su replicabilidad en otras granjas avícolas de características similares, con mínimas adaptaciones.

Este proyecto tiene como objetivo principal brindar una solución tecnológica a los desafíos actuales en la gestión productiva y ambiental del proceso de crianza avícola, cubriendo todo el ciclo desde la recepción de pollitos hasta la entrega de los pollos engordados. Se busca sistematizar el registro de datos críticos, facilitar la toma de decisiones y garantizar una trazabilidad completa de los eventos que afectan a cada camada.

Entre los procesos de negocio que serán abordados se encuentra la gestión integral de camadas, incluyendo su creación, subdivisión por galpón, seguimiento diario de mortalidad, control de peso por muestra y cierre de ciclo. El sistema permitirá además visualizar datos en tiempo real provenientes de sensores físicos instalados en cada galpón (temperatura, humedad, presión, luz, entre otros), sin posibilidad de ingreso manual de variables ambientales, garantizando así la confiabilidad de los registros.

Se contempla el uso de una API externa para obtener pronósticos climáticos, los cuales no se almacenarán, sino que se utilizarán exclusivamente para anticipar condiciones en tiempo real. El sistema generará alertas automáticas en base a condiciones ambientales críticas o umbrales definidos, pudiendo configurarse estas alertas de manera general o específicas por galpón. También se registrarán visitas veterinarias, causas de mortalidad seleccionadas de una lista predefinida, logs de eventos y dispositivos, y datos relevantes del entorno externo.

El proveedor o frigorífico, identificado con un nombre hipotético, podrá acceder al sistema con permisos de solo lectura, visualizando información general de las camadas, reportes, gráficos y el tablero principal.

El sistema será exclusivamente web, accesible desde múltiples dispositivos con conexión a internet, y no contempla funcionamiento offline. Todas las funcionalidades estarán organizadas bajo distintos perfiles de usuario (Operario, Gerente, Administrador, Proveedor), cada uno con permisos diferenciados según sus responsabilidades. No se contempla la automatización de dispositivos ni la generación de recomendaciones por parte del sistema, aunque sí el envío de notificaciones ante situaciones relevantes.

La visualización de datos en el sistema será en línea, permitiendo exportar reportes a pedido del usuario. No se generarán reportes automáticos ni se realizarán acciones sin intervención humana.

Este proyecto no contempla el desarrollo completo del sistema, limitándose a las etapas de análisis, diseño y desarrollo de un MVP funcional que permita validar los requerimientos definidos. Como entregables se incluirán los 20 casos de uso completamente desarrollados y el modelo de base de datos relacional, además de la documentación necesaria para futuras fases de implementación.

## 

## II.3 - Requerimientos

### II.3.1 - Requerimientos funcionales

El sistema permitirá:

1. Registrar nuevas camadas indicando fecha de ingreso, proveedor y distribución inicial de aves por galpón. La suma total de aves ingresadas deberá coincidir con el total general de la camada.
2. Registrar y consultar información detallada de cada camada, incluyendo fecha de ingreso, fecha de egreso, cantidad inicial de aves, total de fallecidos, pesos promedios registrados, y estado actual (activa o finalizada).
3. Registrar muertes diarias por galpón, seleccionando el motivo desde una lista predefinida y actualizando automáticamente la cantidad de aves vivas por galpón y por camada.
4. Registrar pesos promedios de muestras de aves por galpón, permitiendo el ingreso de valores decimales. El sistema validará que el peso ingresado no supere 10 kg y no sea inferior al último peso registrado para la misma camada.
5. Monitorear en tiempo real la temperatura y humedad interna de cada galpón, desagregado por secciones. Los datos serán actualizados al menos una vez por hora.
6. Monitorear variables ambientales externas (temperatura, humedad, presión, viento, lluvia) mediante estación meteorológica y API externa.
7. Visualizar un tablero principal con información consolidada de todas las camadas, galpones, condiciones ambientales y alertas activas.
8. Visualizar dashboards individuales por galpón, incluyendo variables ambientales, datos de la camada, estado de dispositivos y alertas.
9. Controlar manualmente ventiladores, aspersores y cortinas por sección, desde una interfaz web accesible solo a roles autorizados. El sistema registrará la acción con fecha, hora y usuario.
10. Configurar umbrales de alerta por galpón o de forma general, especificando valores críticos de temperatura y humedad. El sistema generará alertas automáticas si se exceden dichos valores.
11. Enviar alertas automáticamente mediante notificaciones push, correo electrónico o WhatsApp, según preferencia del usuario configurada en su perfil.
12. Consultar y generar reportes históricos por camada o galpón, incluyendo gráficos de evolución de peso y series temporales de temperatura, humedad y mortalidad.
13. Registrar visitas veterinarias, con fecha, responsable, observaciones y si se administró o no medicación.
14. Registrar incidencias operativas (ej. cortes de energía, fallas técnicas o eventos extraordinarios) indicando fecha, tipo y descripción.
15. Generar y enviar automáticamente un informe diario al proveedor con resumen de cada galpón y camada, incluyendo mortalidad, peso promedio, estado de dispositivos y alertas. Si faltan datos, se indicará su ausencia. El envío se realiza por WhatsApp y correo electrónico.
16. Permitir a los usuarios con rol Proveedor consultar informes anteriores, generar gráficos propios, visualizar el estado actual del sistema y exportar información en PDF.
17. Registrar en bitácora todas las acciones relevantes del sistema, incluyendo manipulación de dispositivos, ingreso de datos, generación de reportes, cambios de configuración, y accesos de usuarios.

### II.3.2 - Requerimientos no funcionales

El sistema deberá:

1. Garantizar un tiempo máximo de respuesta de 15 segundos en operaciones de monitoreo, consulta de datos y control de dispositivos.
2. Tener una disponibilidad 24/7 con un uptime mínimo del 99.9%.
3. Encriptar los datos sensibles almacenados en la base de datos, incluyendo credenciales, registros de usuarios y configuraciones del sistema.
4. Cerrar automáticamente la sesión de cualquier usuario luego de 15 minutos de inactividad.
5. Integrarse con sensores físicos mediante protocolos estándar como ModBus RTU o MQTT.
6. Integrarse con una API de clima externo para enriquecer los registros con variables ambientales externas.
7. Ser compatible con navegadores modernos: Chrome (versión 90+), Firefox (versión 88+) y Edge (versión 91+).
8. Escalar horizontalmente para soportar la incorporación de nuevos galpones, sensores y usuarios sin afectar la estabilidad del sistema.
9. Realizar copias de seguridad diarias automáticas de la base de datos para prevenir pérdida de información.

### II.3.3 - Requerimientos de experiencia de usuarios (UX)

La interfaz deberá:

1. Validar y restringir valores ingresados en los formularios, mostrando errores claros ante inconsistencias (por ejemplo, pesos excesivos o negativos).
2. Ofrecer diseño responsive adaptable a pantallas de celulares, tablets o monitores, asegurando buena visualización en campo y oficina.
3. Emplear íconos descriptivos, indicadores visuales y tooltips para facilitar el aprendizaje del sistema, especialmente en las funciones más técnicas.
4. Permitir distinguir visualmente el estado de camadas, alertas y dispositivos mediante etiquetas e íconos de color fácilmente interpretables.
5. Clasificar las alertas según tipo (ambiental, técnica, informativa) mediante distintos estilos de notificación.
6. Facilitar la navegación a usuarios con experiencia limitada mediante diseño limpio, jerarquías claras de información y ayuda contextual.
7. Permitir que cualquier función principal (monitoreo, control, registro de datos) se realice en no más de 3 clics desde el menú principal.
8. Mostrar los datos clave (por ejemplo, temperatura actual o número de muertes diarias) sin necesidad de hacer scroll ni abrir submenús.
9. La criticidad de las alertas deberá ser distinguible por color.

## II.4 - Resultados esperados

Esta sección describe los resultados que se esperan en relación a los objetivos del proyecto.

**Mejora en la Toma de Decisiones Basadas en Datos en Tiempo Real**

Se espera que, desde la implementación del sistema, el 100% de las decisiones críticas relacionadas con control ambiental y gestión de camadas se basen en datos monitoreados en tiempo real. Esto permitirá una reducción del tiempo de reacción ante anomalías a menos de 15 segundos y una notoria mejora en la eficiencia operativa.

**Reducción de Mortalidad en las Camadas**

El sistema contribuirá a una disminución mínima del 2% en la tasa de mortalidad promedio de las aves dentro de los primeros cinco ciclos de crianza desde su implementación, comparado con los valores históricos registrados antes del sistema.

**Registro Eficiente y Digitalizado de Información Productiva**

Durante los primeros meses de uso, se espera alcanzar un 70% de completitud en los registros diarios de mortalidad y pesajes por parte de los operarios, considerando una curva de adopción progresiva del sistema. A partir del sexto mes de implementación, se proyecta llegar a un 95% de registros completos, lo cual permitirá garantizar una trazabilidad plena por camada, galpón y sección. Esta digitalización estructurada será clave para generar reportes confiables, facilitar auditorías internas y optimizar la toma de decisiones.

**Alerta Temprana y Gestión de Riesgos Ambientales**

El sistema emitirá el 100% de las alertas correspondientes a condiciones ambientales fuera de los rangos definidos, en un tiempo menor a 15 segundos desde su detección. El objetivo es reducir al mínimo las incidencias derivadas del estrés térmico en las aves, aumentando la tasa de acciones correctivas oportunas por parte de los operarios.

**Registro y Control Total de los Dispositivos Climatizadores**

Desde la puesta en marcha del sistema, se espera registrar el 100% de las acciones realizadas sobre dispositivos (encendido, apagado, ajuste), asociadas a usuario, fecha y hora. Esto permitirá correlacionar acciones manuales con resultados ambientales e implementar futuras mejoras en eficiencia energética y operativa.

**Consulta Histórica y Reportes Dinámicos**

Dentro de los primeros tres meses de uso, se espera que el 100% de los usuarios con permisos adecuados (como el Gerente y el Proveedor) utilicen las herramientas del sistema para consultar datos históricos y generar sus propios reportes sin intervención técnica. Esto promoverá una cultura de análisis autónomo y permitirá identificar tendencias productivas de forma más eficiente.

**Seguridad, Auditoría y Control de Accesos**

Desde su implementación se garantice la trazabilidad completa de todas las acciones críticas realizadas por los usuarios, asegurando que el 100% de las operaciones clave queden registradas en la bitácora. Esto permitirá auditorías internas completas, detección rápida de errores o acciones no autorizadas, y mejorará la seguridad general del sistema.

# Capítulo III - Factibilidad del proyecto

El objetivo de este análisis es evaluar si el proyecto es viable en términos técnicos, económicos y financieros. Cada uno de estos aspectos será evaluado por separado, y para que el proyecto sea considerado factible, se debe demostrar la viabilidad en todas las áreas.

## III.1 - Factibilidad técnica

La factibilidad técnica evalúa si el proyecto puede implementarse con los recursos técnicos disponibles, como la infraestructura tecnológica, herramientas, recursos humanos y conocimientos necesarios. En este proyecto, la factibilidad técnica está determinada por la capacidad de implementar el sistema de monitoreo, control y gestión de datos para los galpones de la granja avícola.

### III.1.1 - Recursos tecnológicos

El proyecto requiere de la implementación de sensores IoT para monitorear las condiciones ambientales (temperatura y humedad) de los galpones, un sistema centralizado de control para los dispositivos de climatización, y un sistema de registro y análisis de datos.

* **Hardware necesario:**
  + **Sensores de temperatura y humedad** para cada uno de los galpones, que estarán conectados a un sistema de monitoreo centralizado.
  + **Raspberry Pi** o dispositivos similares para procesar localmente los datos de los sensores y actuar como intermediarios entre los sensores y el servidor central.
  + **Servidores** para almacenar los datos y procesar la información en tiempo real, con capacidad suficiente para gestionar la cantidad de información generada por los galpones.
  + **Dispositivos de control remoto** para operar ventiladores, aspersores, calefacción y cortinas en cada galpón, integrados con el sistema de automatización.
  + **Cables y fuentes de alimentación** necesarios para conectar los sensores, Raspberry Pi y otros dispositivos de control, garantizando una instalación segura y eficiente.
  + **Redes inalámbricas** (Wi-Fi o Zigbee) para comunicar los dispositivos en cada galpón con el servidor central, permitiendo una comunicación rápida y estable entre los equipos.
* **Software necesario:**
  + Sistema de control que pueda gestionar y actuar sobre las condiciones ambientales en tiempo real.
  + Base de datos para almacenar registros históricos de temperatura, humedad y condiciones de las aves.
  + Software de visualización de datos para permitir el acceso a los operadores y gerentes.

Los equipos necesarios se encuentran disponibles en el mercado, y la tecnología comercial actual es adecuada para cumplir con los objetivos del proyecto. Esto incluye la capacidad de captar datos en tiempo real con una precisión del 95% y emitir alertas en menos de 15 segundos.

### III.1.2 - Recursos humanos

El equipo humano es un factor clave en la implementación de un proyecto de esta magnitud. En este caso, se necesitan los siguientes roles técnicos:

* **Project Manager:** Responsable de coordinar y supervisar todo el proyecto.
* **Desarrolladores:** Encargados de la programación del sistema de control, base de datos y frontend para la interfaz de usuario.
* **Ingenieros de IoT:** Responsables de la instalación, calibración y monitoreo de los sensores.
* **Tester QA:** Encargado de asegurar la calidad y el buen funcionamiento del sistema.

Ademas, deberemos considerar las capacitaciones necesarias para los empleados de la granja:

* **Operarios:** Serán capacitados para interpretar los datos que se muestran en la aplicación y operar el sistema de control remoto de manera eficiente.
* **Gerentes:** Necesitará formación sobre la toma de decisiones basadas en los datos recopilados y analizados por el sistema.

El equipo humano necesario puede ser conformado internamente o contratando consultores externos especializados en desarrollo de sistemas IoT y bases de datos. La capacitación de los empleados es factible y está planeada para minimizar la resistencia al cambio​.

### III.1.3 - Infraestructura

El proyecto requiere de una infraestructura adecuada para operar los sensores y el sistema de control. La granja debe contar con lo siguiente:

* **Conectividad a internet estable:** esto se requiere para la transmisión de datos desde los sensores a los servidores y para la gestión remota del sistema.
* **Energía eléctrica confiable:** esto es para poder alimentar los dispositivos y servidores.

La granja ya dispone de conexión a internet estable (StarLink) y suministro eléctrico confiable, no habrá impedimentos para la instalación de los sensores y el sistema de control.

### III.1.4 - Viabilidad de la Implementación

Los módulos del sistema (monitoreo ambiental, control manual de dispositivos, recopilación de datos, proyección y análisis, alertas) son técnicamente viables y pueden integrarse sin problemas. La infraestructura de red de la granja soporta la transmisión en tiempo real, y el sistema de visualización garantizará que la información esté disponible con un retraso máximo de 30 segundos.

En conclusión, el proyecto es **técnicamente factible**, ya que cuenta con los recursos tecnológicos y humanos necesarios para su implementación exitosa.

## III.2 - Factibilidad económica

Para llevar a cabo el estudio de factibilidad económica, es posible utilizar un enfoque de **Costos-Beneficios** o **Costos-Costos**. El enfoque **Costos-Beneficios** permite determinar si la inversión realizada en el proyecto está justificada en función de las ganancias que generará, mientras que el enfoque **Costos-Costos** evalúa el ahorro que producirá la implementación del nuevo sistema, en comparación con los costos actuales. En este análisis se empleará el enfoque **Costos-Beneficios** para evaluar la viabilidad del proyecto.

A partir de los costos detallados en el estudio de **factibilidad financiera**, se han identificado tanto los **costos en equipamiento** como los **costos en recursos humanos**, los cuales se encuentran desglosados en la **Tabla 2.1**. Adicionalmente, se presenta la distribución mensual de estos costos durante la ejecución del proyecto en las **Tablas 2.2**.

La **Figura 2.1** muestra la gráfica de costos distribuidos por cada mes, junto con su correspondiente curva de costos. Esta curva permite visualizar cómo se realizarán los desembolsos a lo largo del proyecto y facilita el control de la inversión.

### III.2.1 - Beneficios esperados del proyecto

Entre los beneficios que se espera obtener con la implementación del sistema de gestión para la granja avícola, destacan los siguientes:

1. **Reducción de la mortalidad avícola**: El proyecto permitirá reducir la mortalidad de las aves en al menos un 2%, lo cual se traduce en un beneficio económico directo al evitar pérdidas de producción.
2. **Mejora de la productividad**: La optimización de los procesos de cría, junto con el acceso a datos en tiempo real, permitirá una mayor eficiencia operativa, lo que se refleja en un aumento de la productividad de la granja.
3. **Toma de decisiones basada en datos**: La recopilación y análisis de datos históricos proporcionará una mejor planificación para la gestión de las camadas de aves, el control ambiental y la salud avícola, lo que impactará positivamente en la rentabilidad del negocio.

Estos beneficios esperados superan los costos estimados, lo que hace que el proyecto sea económicamente viable. La inversión inicial se recuperará a través de la reducción de la mortalidad y la mejora de la eficiencia operativa​

## III.3 - Factibilidad financiera

La factibilidad financiera determina si la empresa tiene la capacidad de afrontar los costos del proyecto y gestionar el financiamiento de manera eficiente.

### III.3.1 - Inversión inicial

Para evaluar la viabilidad financiera del proyecto, es fundamental realizar una estimación detallada de la inversión inicial requerida. Esta sección desglosa los costos asociados a los **recursos humanos** y al **equipamiento** necesarios para la implementación del sistema de gestión de la granja avícola. La inversión inicial considera tanto los costos relacionados con el desarrollo del software, la integración de los sensores y dispositivos, como el entrenamiento del personal involucrado.

Las siguientes tablas presentan un resumen de los recursos requeridos, su cantidad, las horas de trabajo estimadas, los costos por hora y el costo total de cada recurso. Además, se incluye el desglose del equipamiento necesario, que abarca desde los sensores ambientales hasta los dispositivos de control. Estos valores permitirán estimar de manera precisa la inversión necesaria para la puesta en marcha del proyecto.

| Rol | Cantidad | Cantidad de horas totales | Costo unitario por hora (USD) | Costo Total (USD) |
| --- | --- | --- | --- | --- |
| Project Manager | 1 | 1000 | 6 | $ 6000 |
| Arquitecto de software | 1 | 72 | 5 | $ 360 |
| Analista Funcional | 1 | 32 | 4 | $ 128 |
| Administrador de Bases de Datos | 1 | 40 | 4 | $ 160 |
| Administrador de Redes | 1 | 40 | 4 | $ 160 |
| Desarrollador Backend & Frontend | 2 | 1000 | 5 | $ 10.000 |
| Tester QA | 1 | 384 | 4 | $ 1.532 |
| Soporte Técnico | 1 | 112 | 4 | $ 448 |
| **Total:** |  |  |  | **$ 18.788** |

**Tabla 2.1**

| Equipo | Cantidad | Costo Unitario (USD) | Costo Total (USD) |
| --- | --- | --- | --- |
| Sensores de temperatura y humedad | 17 | 40$ | $680 |
| Servidores | 2 | $1000 | $2000 |
| Malacate electrico | 8 | $252 | $2016 |
| Switch de red | 1 | $70 | $70 |
| Raspberry Pi (para control) | 4 | $75 | $300 |
| Tablet | 1 | $300 | $300 |
| Cables | 1 lote | $100 | $100 |
| Relé | 68 | $50 | $3400 |
| Converter fibra óptica - ethernet | 5 | $50 | $250 |
| Switch fibra óptica | 1 | $150 | $150 |
| **Total** |  |  | **$** |

**Tabla 2.2**

El **total del proyecto**, considerando los gastos de recursos humanos y los gastos en equipamiento, es de **$25.094**.

### III.3.2 - Flujo de caja y financiamiento

Se proyecta un flujo de caja positivo a mediano plazo, debido a los ahorros derivados de la reducción de mortalidad y la optimización de procesos. El financiamiento se realizará mediante pagos periódicos alineados con el cronograma de implementación. La empresa puede afrontar los costos sin necesidad de recurrir a financiamiento externo, ya que se espera que los beneficios del proyecto comiencen a generarse poco después de su implementación.

### III.3.3 - Retorno de inversión (ROI)

**Cálculo del ROI:** Actualmente, los 4 galpones pueden albergar hasta 70.000 pollos. La mortalidad actual es del 10%, lo que genera un ingreso promedio de 15.750.000 ARS por camada (con un precio de venta de 250 ARS por pollo). Con una reducción de la mortalidad al 8%, el ingreso aumentaría a 16.100.000 ARS por camada.

* **Beneficio por reducción de mortalidad:** 16.100.000 - 15.750.000 = 350.000 ARS por camada.
* **Beneficio anual:** 350.000 ARS x 8 crianzas al año = 2.800.000 ARS (equivalente a 2.333 USD).

**Conclusión del ROI:** Se proyecta un ROI en 11 años considerando únicamente la reducción en la mortalidad de pollos, que aumentaría los ingresos anuales en $2,333 USD. Este período podría reducirse si se incluyen ganancias derivadas de la optimización de procesos y mayor productividad.

### III.3.4 - Plan de pagos

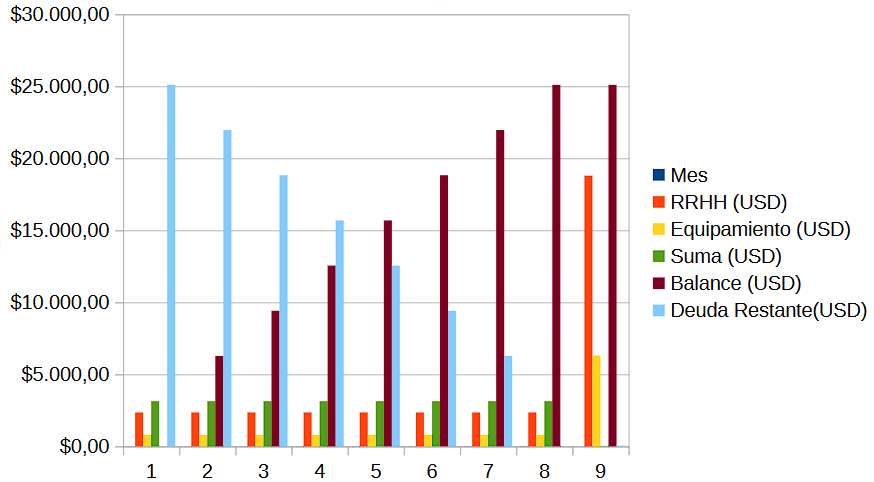
El plan de pagos está diseñado para asegurar que los recursos estén disponibles en el momento adecuado, alineando los desembolsos con el avance del proyecto. Cada mes se registran los costos de los recursos humanos, que incluyen salarios y honorarios, así como los costos de equipamiento, que abarcan la adquisición e instalación de sensores, dispositivos de control y otros componentes necesarios para el sistema de gestión.

La tabla proporcionada desglosa los costos mensuales en tres columnas: **RRHH**, que refleja los gastos en personal; **Equipamiento**, que muestra los costos relacionados con los dispositivos y herramientas; y **Suma**, que calcula el total de gastos por mes. Adicionalmente, se incluyen las **Cuotas**, que representan los pagos programados, y el **Balance**, que indica el saldo acumulado a lo largo del proyecto.

Este plan permite no solo una mejor planificación y control de la inversión, sino que también facilita la identificación de posibles desviaciones en el presupuesto y ayuda a garantizar la viabilidad económica del proyecto en su conjunto.

| **Mes** | **RRHH (USD)** | **Equipamiento (USD)** | **Suma (USD)** | **Balance (USD)** | **Deuda Restante(USD)** |
| --- | --- | --- | --- | --- | --- |
| 1 | $2.348,5 | $788,25 | $3.136,75 | $3.136.75 | $25.094 |
| 2 | $2.348,5 | $788,25 | $3.136,75 | $6.273,5 | $21.957,25 |
| 3 | $2.348,5 | $788,25 | $3.136,75 | $9.410,25 | $18.820,5 |
| 4 | $2.348,5 | $788,25 | $3.136,75 | $12.547 | $15.683,75 |
| 5 | $2.348,5 | $788,25 | $3.136,75 | $15.683,75 | $12.547 |
| 6 | $2.348,5 | $788,25 | $3.136,75 | $18.820,5 | $9.410,25 |
| 7 | $2.348,5 | $788,25 | $3.136,75 | $21.957,25 | $6.273,5 |
| 8 | $2.348,5 | $788,25 | $3.136,75 | $25.094 | $3.136.75 |
| Total | $18788 | $6306 | $25,094 | $25.094 | $0 |

**Tabla 2.3**



**Gráfica 7.1**

## III.4 - Conclusión del análisis de factibilidad

Después de realizar los análisis de factibilidad técnica, económica y financiera, se concluye que el proyecto es viable en los tres aspectos. El sistema puede implementarse con los recursos disponibles, los beneficios económicos superan los costos y la empresa tiene la capacidad financiera para financiar el proyecto de manera eficiente. Por lo tanto, se puede afirmar que el proyecto es **totalmente factible**.

# Capítulo IV - Marco Legal y Normativa Ambiental

Este capítulo aborda las normativas legales aplicables y el impacto ambiental asociado al proyecto de implementación del sistema de gestión en la granja avícola. Se detallan las leyes y regulaciones relevantes en Argentina, especialmente las disposiciones a nivel nacional y provincial, así como la clasificación del impacto ambiental que puede generar el proyecto.

## IV.1 - Ley General del Ambiente (Ley Nacional 25.675)

La **Ley General del Ambiente (Ley 25.675)**, promulgada en el año 2002, es la legislación marco que establece los principios rectores para la preservación, conservación y mejora del medio ambiente en Argentina. Esta ley regula el desarrollo sustentable en todo el país, promoviendo el uso racional de los recursos naturales y la protección de los ecosistemas.

**Principios Rectores de la Ley General del Ambiente**

La ley se basa en varios principios rectores que deben guiar cualquier acción o política que involucre actividades con impacto ambiental. Entre estos, se destacan:

1. **Principio de Prevención**: Las actividades deben planificarse y ejecutarse minimizando cualquier posible daño al ambiente. Ante la posibilidad de un impacto negativo, deben tomarse medidas preventivas.
2. **Principio de Precaución**: Cuando haya peligro de daño ambiental grave o irreversible, la falta de certeza científica no debe usarse como razón para posponer medidas de prevención.
3. **Principio de Responsabilidad**: Quien genere un daño ambiental será responsable de repararlo o compensarlo.
4. **Principio de Sustentabilidad**: El uso de los recursos naturales debe ser compatible con las necesidades presentes y futuras de la sociedad.
5. **Principio de Equidad Intergeneracional**: Las decisiones ambientales deben asegurar que las futuras generaciones puedan acceder a un ambiente saludable.

## IV.2 - Clasificación del Impacto Ambiental

En relación con la Ley General del Ambiente y las disposiciones provinciales, se identifican tres categorías principales de impacto ambiental para los proyectos:

1. **Categoría 1: Bajo Impacto Ambiental**
   * Proyectos que generan impactos mínimos o nulos sobre el medio ambiente. Los efectos negativos, si los hubiera, son insignificantes y fácilmente reversibles. Estos proyectos no requieren estudios ambientales exhaustivos.
2. **Categoría 2: Mediano Impacto Ambiental**
   * Proyectos con impacto moderado sobre el ambiente, cuyas consecuencias son más significativas pero pueden ser mitigadas mediante medidas correctivas conocidas. Es necesario realizar un estudio de impacto ambiental, aunque de menor envergadura.
3. **Categoría 3: Alto Impacto Ambiental**
   * Proyectos que presentan un impacto ambiental considerable, con posibles daños irreversibles o de gran magnitud. Estos proyectos deben someterse a una Evaluación de Impacto Ambiental (EsIA) exhaustiva y requieren medidas específicas para prevenir o mitigar los efectos.

## IV.3 - Normativa Provincial en Entre Ríos: Decreto 4977/09

A nivel provincial, el **Decreto 4977/09** de Entre Ríos regula los procedimientos y requisitos que deben cumplir los emprendimientos o actividades que puedan afectar el ambiente. Este decreto se complementa con la Ley Nacional del Ambiente y establece que, dependiendo del nivel de impacto que se espere del proyecto, los emprendedores deberán presentar informes específicos.

El decreto también se basa en la clasificación de tres categorías de impacto ambiental:

* **Categoría 1**: Bajo impacto, que no requiere estudios ambientales exhaustivos.
* **Categoría 2**: Mediano impacto, que requiere medidas correctivas.
* **Categoría 3**: Alto impacto, que exige una evaluación completa del impacto ambiental.

## IV.4 - Clasificación del Proyecto

El proyecto de implementación del sistema de gestión en la granja avícola se clasifica dentro de la Categoría 1 de impacto ambiental, según el Decreto 4977/09 de Entre Ríos. Este proyecto tiene un impacto ambiental bajo debido al uso de tecnologías digitales y sensores, que reducen la necesidad de materiales físicos (como papel) y optimizan el consumo energético. No se prevén riesgos significativos para el medio ambiente, ya que los dispositivos y equipos utilizados están diseñados para minimizar la emisión de desechos y no tienen un impacto negativo importante sobre el entorno natural.

# Capítulo V – Marco Teórico

## V.1 - Industria Avícola Argentina: Relevancia Regional y Nacional

La industria avícola argentina representa uno de los pilares del sistema agroalimentario nacional, con un impacto significativo en la economía, el empleo rural y el abastecimiento interno. Según el Ministerio de Economía y la Secretaría de Agricultura, Ganadería y Pesca, Argentina ocupa el octavo lugar a nivel mundial en producción de carne aviar, y figura también entre los principales países exportadores del rubro.

En 2020, el país alcanzó un consumo per cápita de 57 kg de carne de pollo, ocupando el primer puesto mundial. Esta preferencia del consumidor argentino responde a factores como su bajo costo relativo, la versatilidad del producto y los avances tecnológicos que han permitido una oferta constante y segura. Tal como sostiene el IERAL de la Fundación Mediterránea (2024), *“Argentina cambia sus patrones de consumo, pero no las cantidades”*, consolidando el consumo aviar como una constante en la dieta nacional.

Entre Ríos se posiciona como el principal núcleo productivo del país, concentrando alrededor del 50% de la producción nacional. Su rol es clave dentro de un sistema de integración vertical donde cooperativas, granjas, frigoríficos y proveedores articulan una cadena eficiente. Esta estructura ha favorecido la tecnificación de procesos, el acceso a genética mejorada, y el cumplimiento de estándares sanitarios mediante fiscalización del Senasa.

La competitividad del sector también ha sido reforzada por la incorporación de tecnologías digitales, la automatización de procesos y la profesionalización del manejo productivo. A pesar de esto, muchas granjas pequeñas aún enfrentan desafíos relacionados con el control ambiental, la eficiencia operativa y el acceso a sistemas de información, lo que justifica la necesidad de soluciones tecnológicas accesibles que optimicen la toma de decisiones y minimicen pérdidas por mortalidad.

## V.2 - Factores Críticos del Bienestar Animal en la Producción Avícola

El bienestar animal ha dejado de ser un concepto vinculado exclusivamente a lo ético para convertirse en una variable crítica de calidad, rentabilidad y sostenibilidad. Según la Organización Mundial de Sanidad Animal (OMSA, 2023), *“el bienestar animal es el estado físico y mental del animal en relación con las condiciones en las que vive y muere”*. Su incumplimiento repercute directamente sobre indicadores productivos como la tasa de conversión alimenticia, el crecimiento, la mortalidad y la calidad de la carne.

Diversos estudios confirman que el estrés térmico es una de las principales amenazas en granjas intensivas. Como señala Rodríguez (2023) en AviNews, *“el estrés térmico o calórico es el conjunto de cambios fisiológicos y comportamentales que se desencadenan en los animales cuando son sometidos a condiciones ambientales que superan su zona termoneutral y son incapaces de regular su temperatura interna”*. Este fenómeno afecta el comportamiento alimentario, reduce el crecimiento, y aumenta la tasa de mortalidad.

En pollitos recién nacidos, el desafío es mayor: presentan comportamiento poiquilotérmico, es decir, dependen completamente del ambiente para mantener su temperatura corporal. Por ello, se recomienda mantener la temperatura del galpón entre 30 y 32 °C durante la primera semana. A medida que las aves crecen, esta temperatura puede disminuir hasta 20–23 °C. Asimismo, la humedad relativa óptima debe mantenerse entre 50% y 70% para evitar deshidratación o proliferación de patógenos en la cama húmeda.

Factores adicionales como la sobrepoblación, el manejo brusco, la ventilación deficiente o la iluminación inadecuada pueden provocar estrés crónico, afectando la salud intestinal, inmunidad y calidad de la canal. La normativa del Senasa establece controles periódicos de bienestar, infraestructura y registros productivos, en cumplimiento del Programa Nacional de Sanidad Avícola.

## V.3 - Transformación Digital e IoT en el Manejo Avícola

La adopción del Internet de las Cosas (IoT) en el sector avícola permite transformar las granjas en sistemas inteligentes capaces de monitorear en tiempo real las condiciones ambientales, reduciendo la necesidad de intervención humana. Dispositivos como sensores, microcontroladores y plataformas de visualización se integran para registrar variables como temperatura, humedad, amoníaco, consumo de agua o alimentación, y comportamiento animal.

Como detalla Gutiérrez en AviNews (2023), *“los parámetros de temperatura, humedad, gas amoniaco y nivel de agua son monitoreados y controlados con ayuda del microcontrolador PIC. Los valores se cargan en la web, permitiendo conocer el entorno interno de las granjas desde una computadora o dispositivo móvil.”*

Entre las tecnologías disponibles se encuentran sensores como el DHT22 (temperatura y humedad), MQ135 (gases), BH1750 (intensidad lumínica), y controladores como Raspberry Pi o ESP32. La transmisión se realiza mediante protocolos como ModBus RTU o MQTT, eficientes para entornos rurales por su bajo consumo y robustez.

El uso de ordenadores de placa única (SBC) permite integrar múltiples sensores y automatizar el encendido de dispositivos como extractores o aspersores. Esta infraestructura, accesible y escalable, permite al productor pequeño o mediano acceder a herramientas que antes estaban reservadas a sistemas industriales complejos.

## V.4 - Análisis de Datos y Sistemas de Alerta Temprana

El análisis de datos en la avicultura permite identificar patrones, anticipar problemas y tomar decisiones informadas en tiempo real. Como explica David Speller, director de Optifarm, el uso de datos permite monitorear el desempeño, predecir resultados y reducir la dependencia de antibióticos.

En el proceso de engorde, se pueden analizar variables como:

* Temperatura y humedad ambiente.
* Consumo diario de alimento y agua.
* Mortalidad acumulada y diaria.
* Actividad del lote (medida por sensores o cámaras).

Por ejemplo, un descenso del consumo de agua puede predecir caídas de rendimiento y activar decisiones como la revisión de bebederos. De igual forma, patrones inusuales de temperatura o humedad pueden correlacionarse con aumentos en la mortalidad y disparar alarmas automatizadas.

El sistema propuesto incorpora una lógica de alertas que evalúa condiciones anómalas mediante umbrales configurables, permitiendo su notificación por email, WhatsApp o notificación visual. Estas alertas actúan como mecanismos de intervención temprana ante situaciones de riesgo, alineándose con prácticas recomendadas por el PMBOK (2021) en cuanto a medición del desempeño y mitigación de incertidumbre.

## V.5 - Ingeniería de Sistemas: Enfoque Sistémico para una Producción Inteligente

La ingeniería de sistemas aporta una visión estructurada y multidisciplinaria para abordar problemas complejos como el manejo ambiental en galpones avícolas. Según SEBoK (2023), los sistemas productivos deben abordarse como **sistemas socio-técnicos**, en los que interactúan dispositivos físicos, software, personas, procesos y regulaciones.

El diseño del sistema propuesto contempla:

* Subsistemas físicos: sensores, actuadores, galpones.
* Subsistemas informáticos: backend, dashboards, protocolos de comunicación.
* Actores humanos: operarios, gerentes, veterinarios, proveedores.
* Reglas de negocio y objetivos de eficiencia, bienestar y rentabilidad.

Este enfoque permite modelar flujos de datos, puntos de falla, dependencias funcionales y protocolos de actuación. Además, facilita la escalabilidad modular del sistema, la trazabilidad de decisiones y la estandarización de reportes. La metodología empleada considera fases iterativas de diseño, integración, prueba y validación técnica, alineadas con prácticas de ingeniería de sistemas y gestión de proyectos modernas.

## V.6 - Avicultura de Precisión y Tendencias en Agricultura 4.0

El concepto de Avicultura 4.0 surge como parte del paradigma de la Agricultura 4.0, integrando sensores, automatización, inteligencia artificial y big data en sistemas productivos orientados a la eficiencia, sostenibilidad y bienestar. Esta evolución promueve la digitalización de cada aspecto de la producción avícola, permitiendo intervenciones basadas en datos y no en la experiencia empírica únicamente.

Según FONTAGRO (2020), *“la agricultura de precisión transforma cada unidad productiva en un sistema inteligente, permitiendo intervenciones específicas basadas en evidencia y no en suposiciones.”* En el caso del proyecto desarrollado, el sistema actúa como base para futuras integraciones como:

* Algoritmos de predicción de mortalidad por machine learning.
* Automatización completa del microclima por controladores adaptativos.

El avance hacia estos modelos representa una oportunidad concreta para productores de Entre Ríos y del país, permitiendo una mejora continua, sostenible y científicamente informada de la producción aviar.

# 

# Capítulo VI - Metodología de resolución

## VI.1 - Introducción a la metodología

En el presente proyecto, orientado a la gestión y control de galpones avícolas mediante la integración de tecnologías IoT y dispositivos de climatización remota, se plantean requisitos cambiantes y una alta demanda de entregas rápidas que aporten valor de forma constante. Desde la perspectiva de la Ingeniería en Sistemas de la Información, resulta esencial adoptar un enfoque que satisfaga tanto la **eficiencia** y **flexibilidad** operacional como la **rapidez** y **seguridad** en la construcción del producto. Bajo este panorama, se ha optado por la adopción de una **metodología ágil** que cumpla con los siguientes principios:

1. **Enfoque iterativo e incremental**Dado el elevado nivel de complejidad técnica —sensores IoT para temperatura y humedad y dispositivos de climatización con control en tiempo real— los requisitos están sujetos a cambios frecuentes (p. ej., nuevas normas de bioseguridad o ajustes para disminuir la mortalidad aviar).  
   Al desarrollar en ciclos cortos, se entrega de forma incremental un producto que puede ser evaluado en etapas tempranas y ajustado de inmediato según el feedback recibido, garantizando la calidad en cada iteración.
2. **Colaboración estrecha entre los miembros del equipo**El éxito de un proyecto no solo depende de un óptimo diseño técnico, sino también de la **coordinación** y **sinergia** de todos los integrantes: desarrolladores, analistas y los propios usuarios finales (gerente de la granja, operarios).  
   Las metodologías ágiles fomentan la comunicación y la transparencia, facilitando la identificación temprana de riesgos o bloqueos y promoviendo la resolución colaborativa de problemas.
3. **Capacidad de adaptarse a cambios sin comprometer la calidad**El entorno avícola puede requerir reacciones rápidas ante situaciones inesperadas (como urgencias sanitarias o modificaciones en la gestión de los pollos).  
   Una metodología ágil posibilita reelaborar el plan ante cualquier contingencia y **re-priorizar** funcionalidades, a la vez que asegura la robustez del producto final gracias a prácticas de mejora continua y validación constante.

En particular, se decidió combinar **Scrum** y **Kanban**, aprovechando los beneficios específicos que cada marco de trabajo ofrece:

1. **Scrum  
   Estructura organizada en ciclos de trabajo (sprints):** Define intervalos regulares (por lo general, de 2 a 4 semanas) en los que se planifican objetivos concretos para el equipo, se construyen incrementos de producto y se inspeccionan los resultados al cierre de cada ciclo.  
   **Roles y ceremonias claras:** El Product Owner establece prioridades de negocio, el Scrum Master facilita las dinámicas y remueve impedimentos, mientras que el Equipo de Desarrollo lleva a cabo las tareas técnicas. Ceremonias como la Sprint Review y la Retrospective aseguran la retroalimentación y la optimización continua de procesos.  
   **Priorización y entrega de valor:** Cada sprint se orienta a proveer algo funcional, ya sea un prototipo para la captura de datos de sensores IoT o un módulo básico para la gestión de camadas en los galpones. Esto genera confianza en las partes interesadas y confirma la utilidad del sistema antes de su fase final.
2. **Kanban  
   Herramienta visual para el seguimiento de tareas:** A través de tableros con columnas (por ejemplo, “Por hacer”, “En curso”, “Hecho”), se obtiene un panorama claro y compartido del estado actual de las actividades.  
   **Límites en trabajo en proceso (WIP limits):** Para evitar cuellos de botella, se restringe la cantidad de tareas que pueden permanecer en la columna “En curso”. Esta práctica fomenta la finalización de tareas antes de comenzar nuevas, mejorando la fluidez y evitando acumulaciones de pendientes.  
   **Respuesta ágil ante imprevistos:** En un proyecto donde pueden surgir incidentes de manera inesperada (p. ej., falla repentina en un sensor o necesidad de ajustar los umbrales de calor), Kanban facilita la incorporación inmediata de la incidencia al tablero y su priorización según el impacto, sin tener que esperar al siguiente sprint formal.

La **sinergia** entre **Scrum** y **Kanban** cobra especial relevancia en la implementación de un sistema complejo:

* **Gestión flexible de actividades diarias**: Mientras Scrum propone objetivos de sprint y ceremonias de revisión, Kanban permite un control más fino del flujo de trabajo diario. Por ejemplo, en mitad de un sprint, si aparece una urgencia ligada a la calibración de sensores, Kanban brinda la inmediatez de visualizarlo y tomar acción prioritaria sin esperar la conclusión de la iteración.
* **Priorización y visibilidad constantes**: El Product Owner puede decidir en qué momento incluir nuevos requerimientos en el backlog; al mismo tiempo, el tablero Kanban deja en evidencia cualquier cuello de botella que obstaculice el progreso, ya sea en programación, pruebas de hardware o configuración de software.
* **Reducción de riesgos y mayor calidad**: La integración de sensores IoT o módulos de control remoto introduce riesgos técnicos considerables. Los ciclos cortos de Scrum (sprints) posibilitan validar porciones del sistema y obtener retroalimentación temprana; simultáneamente, Kanban transparenta el avance y destaca tareas cruciales que pudieran complicar la calidad final del proyecto (por ejemplo, asegurar la conexión y estabilidad de datos en tiempo real).

De esta manera, la propuesta responde a las exigencias de este **proyecto**, combinando rigor metodológico (Scrum) con la flexibilidad y la inmediatez de Kanban, dando como resultado:

* **Entrega temprana y recurrente de incrementos**, validando hipótesis y funcionalidades cruciales (control de temperatura/humedad, registro de mortalidad, alertas automáticas, etc.).
* **Gestión efectiva de cambios**, enfocada en la mejora continua y la satisfacción de las necesidades reales de la granja avícola.
* **Transparencia y colaboración** entre todos los involucrados, esencial para proyectos en los que convergen aspectos tecnológicos, operativos y de investigación académica.

En suma, la elección de un enfoque ágil basado en **Scrum + Kanban** promueve un desarrollo ordenado y una implementación exitosa del sistema de gestión de galpones avícolas, asegurando que el producto final cumpla con los estándares de calidad y con los objetivos de eficiencia, flexibilidad y rápida entrega de valor, que son centrales para este proyecto de final de carrera.

## VI.2 - Estructura general

### Duración de cada sprint

Se sugiere 2 semanas por sprint.

### **Equipo**

* **Scrum Master** / Líder de proyecto: Facilita ceremonias, remueve impedimentos.
* **Product Owner (PO):** Define prioridades de negocio y cambios de alcance.
* **Equipo de desarrollo:** Encargado del diseño, programación, pruebas.
* **Stakeholders clave:** Dueño/gerente de la granja, operarios, y asesores externos (veterinario, etc.).

### Modelo de gobernanza y toma de decisiones

En línea con las recomendaciones del PMBOK (2021), se estableció un modelo de toma de decisiones que permite mantener claridad en la asignación de autoridad durante el desarrollo del proyecto. Este modelo garantiza un control eficiente del alcance, la calidad de los entregables y la priorización de funcionalidades relevantes para el negocio.

| **Nivel de decisión** | **Responsable asignado** | **Rol específico en el proyecto** |
| --- | --- | --- |
| **Aprobación de entregables clave** | Dueño de la granja (Sponsor del proyecto) | Valida entregas funcionales y autoriza el cierre de sprints. |
| **Priorización de funcionalidades** | Product Owner (PO) designado | Define el backlog, aprueba cambios de alcance, evalúa valor entregado. |
| **Validación técnica de componentes** | Scrum Master + Equipo de desarrollo | Evalúan calidad técnica, viabilidad y cumplimiento de Definition of Done. |
| **Gestión de incidencias y cambios operativos** | Scrum Master + PO + Stakeholders | Deciden colectivamente sobre imprevistos que afecten planificación. |

Este modelo se refuerza mediante ceremonias ágiles (Review y Retrospective), donde todas las partes tienen visibilidad sobre el progreso, pueden emitir observaciones y aprobar formalmente cada incremento entregado.

### Plan de comunicación y seguimiento

Para asegurar la alineación entre los miembros del equipo, el comitente y los usuarios clave, se definió un plan de comunicación regular que permite visualizar avances, resolver bloqueos y documentar decisiones. Este plan combina instancias formales e informales, aprovechando herramientas digitales accesibles para todos los actores del proyecto.

| **Canal / Herramienta** | **Frecuencia** | **Participantes** | **Objetivo principal** |
| --- | --- | --- | --- |
| **Daily Scrum (virtual / presencial)** | Diaria (15 min) | Scrum Master, equipo de desarrollo | Seguimiento diario, visualización de bloqueos, coordinación de tareas. |
| **Sprint Planning** | Quincenal | Todo el equipo, PO, SM | Definir alcance del sprint, estimar tareas, alinear prioridades. |
| **Sprint Review** | Quincenal | PO, stakeholders, equipo | Validar entregables, obtener feedback de usuario. |
| **Sprint Retrospective** | Quincenal | Equipo completo | Identificar mejoras de proceso, resolver conflictos. |
| **Reunión de revisión con el Sponsor** | Mensual o por entregable clave | Gerente de la granja, PO, SM | Validar avances críticos, revisar posibles cambios de alcance. |
| **WhatsApp / Telegram** | Permanente | PO, SM, desarrolladores | Comunicación informal y ágil para resolver dudas urgentes. |
| **GitHub / Trello / Jira** | Permanente | Todo el equipo | Seguimiento técnico de tareas, issues, bugs, commits y documentación. |
| **Google Drive / Notion** | Actualización continua | Todo el equipo | Repositorio de documentación, entregables, decisiones, avances. |

Además, se lleva un registro de decisiones clave mediante bitácoras compartidas, y se documentan los cambios de alcance o backlog con trazabilidad clara en el historial del proyecto.

## VI.3 - Dinámica Scrum + Kanban

Cada sprint inicia con una **Sprint Planning**, donde se seleccionan las historias de usuario o casos de uso priorizados para abordar en las próximas 2 semanas.  
Durante el sprint, el equipo utiliza un **tablero Kanban** (con columnas como “Pendiente / En curso / Hecho”) para visualizar y gestionar las tareas diarias. Se establecen límites de WIP (Work In Progress) para evitar la sobrecarga de trabajo simultáneo.  
Cada día se realiza un **Daily Scrum** breve (15 min aprox.) para sincronizar esfuerzos y detectar bloqueos.  
Al final de cada sprint, hay una **Sprint Review** (donde se presenta lo logrado al PO y stakeholders) y una **Retrospective** (para mejorar procesos y comunicación en el equipo).

# 

# Capítulo VII - Planificación

En el proceso de realizar la planificación de cómo se llevará a cabo nuestra metodología, se han establecido diez sprints, cada uno con objetivos y actividades bien delimitadas. Esta segmentación en ciclos cortos posibilita entregar progresivamente resultados concretos, recibiendo retroalimentación continua y garantizando la capacidad de adaptación a cambios o mejoras que pudiesen surgir durante el desarrollo.

Los sprints se conciben con una duración de dos semanas cada uno, lo que ofrece un balance adecuado entre la rapidez de las entregas y el tiempo necesario para realizar la planificación, el desarrollo y las pruebas correspondientes. De esta manera, se obtiene un proyecto con una duración total aproximada de 20 semanas (alrededor de cinco meses). En cada sprint se implementan partes específicas del sistema —como la autenticación de usuarios, la gestión de camadas, la integración de sensores o la emisión de reportes históricos—, permitiendo evaluar avances de manera continua y asegurar la calidad del producto final.

A lo largo de este período, se incorporan los principios y ceremonias de Scrum —Planning, Daily, Review y Retrospective— combinados con la visualización de tareas y control de flujo propio de Kanban. De este modo, el equipo mantiene una visión clara de sus responsabilidades y el estado de cada tarea, y puede reaccionar con agilidad ante imprevistos, sin que ello comprometa el objetivo global del proyecto. Al término de los diez sprints, se espera un sistema completamente funcional, probado en escenarios reales y documentado de acuerdo con las exigencias de un proyecto final de la carrera de Ingeniería en Sistemas de la Información.

## VII.1 - Métricas de rendimiento y flujo de trabajo

Durante el desarrollo del sistema, se aplicaran métricas clave provenientes de enfoques ágiles (Scrum y Kanban) para evaluar la eficiencia del equipo, la estabilidad del flujo de trabajo y la calidad de la entrega incremental.

Estas métricas permiten realizar un seguimiento objetivo del rendimiento y aplicar mejoras basadas en datos, en línea con los dominios de **Medición** (PMBOK) y **Technical Management & Quality** (SEBoK).

| **Métrica** | **Definición práctica en el proyecto** | **Herramienta utilizada** |
| --- | --- | --- |
| **Lead Time** | Tiempo desde que una tarea entra al backlog hasta que se completa. | Jira / Trello |
| **Cycle Time** | Tiempo desde que se comienza a trabajar en una tarea hasta su entrega. | Jira / Trello |
| **WIP (Work In Progress)** | Cantidad de tareas simultáneamente activas. Se estableció un límite de 3 por persona. | Tablero Kanban |
| **% de cumplimiento por sprint** | Tareas completadas vs. tareas comprometidas en Sprint Planning. | Jira / Spreadsheet compartido |
| **Tasa de bugs detectados** | Incidencias abiertas durante QA sobre el total de historias entregadas. | GitHub Issues / QA logs |
| **Velocidad del equipo** | Promedio de puntos de historia completados por sprint (solo para estimaciones internas). | Jira / Google Sheets |
| **Burndown chart** | Gráfico de evolución de tareas pendientes en el sprint. | Jira / gráficos manuales por sprint |

Estas métricas serán monitoreadas semanalmente y utilizadas en las **retrospectivas** para identificar cuellos de botella, sobrecarga de tareas o desvíos frente a lo planificado, favoreciendo así una mejora continua del proceso y mayor previsibilidad en las entregas.

## VII.2 - Actividades a realizar

| **Actividad** | **Descripción** | **Horas** | **Recursos Humanos** | **Resultados Esperados** |
| --- | --- | --- | --- | --- |
| **1. Sprint 1: Inicio y Organización** | Configuración inicial (repositorio, tableros Kanban, entorno), revisión de alcance y requerimientos, definición de la arquitectura preliminar (BD, frameworks, IoT). Convertir casos de uso en historias de usuario y organizar el primer plan de tareas. | 80 | Project Manager, Analista Funcional, Dev Team, QA | - Product Backlog priorizado y documentado  - Entorno de desarrollo configurado  - Documento con esquema preliminar de arquitectura |
| **2. Sprint 2: Manejo de Usuarios y Seguridad** | Implementar autenticación (login/logout), cifrado de credenciales, creación de usuarios con roles (Operario, Gerente, Admin). Validaciones de sesión y seguridad (tiempos de expiración, encriptación). | 80 | Project Manager, Dev Team, QA | - Módulo de autenticación completo  - Interfaz de creación/edición de usuarios y roles  - Documentación del flujo de login y asignación de roles |
| **3. Sprint 3: Estructura de Datos y Configuración de Galpones** | Refinar la BD para Galpones, Camadas, Sensores, Alertas. Implementar CRUD de galpones con validaciones de capacidad y ubicación. Preparar asociaciones iniciales con sensores si el alcance lo contempla. | 70 | Dev Team, Administrador de BD, QA | - BD actualizada con entidades clave  - Pantalla/endpoint de alta/edición de galpones  - Modelo de datos validado |
| **4. Sprint 4: Gestión de Camadas y Datos (Parte 1)** | Iniciar la lógica de camadas: caso de uso “Registrar Nueva Camada” (fecha de ingreso, proveedor), validaciones iniciales (capacidad de galpones). Se diseña la parte del frontend necesaria para crear camadas y almacenarlas en la BD. | 80 | Dev Team, QA | - CRUD parcial de camadas (alta, edición básica)  - BD con primeros datos de camadas  - Formularios (frontend/backend) para ingresar camadas |
| **5. Sprint 5: Gestión de Camadas y Datos (Parte 2)** | Completar el registro de mortalidad (“Registrar Muertes de Aves”) y pesaje (“Registrar Pesos”). Validar rangos de peso, no sobrepasar la capacidad real de cada galpón, etc. Finalizar CRUD de camadas, abarcando la edición y cierre de ciclo. | 80 | Dev Team, QA | - CRUD completo de camadas (mortalidad, pesaje, cierre de camada)  - BD reflejando datos productivos en tiempo real |
| **6. Sprint 6: Integración IoT Básica (Lectura de Sensores)** | Lectura real o simulada de los sensores de temperatura/humedad. Escoger protocolo (MQTT, ModBus), programar el servicio de recolección periódica y almacenar las mediciones (timestamps) en la base de datos. | 80 | Dev Team (IoT), QA | - Servicio que lee datos de los sensores  - BD con logs/timestamps asociados a cada galpón  - Pruebas iniciales con hardware real o simulación |
| **7. Sprint 7: Alertas y Tablero Ambiental** | Definir umbrales por galpón (caso de uso “Configurar Alertas”). Al superar los valores, generar notificaciones (correo, push). Diseñar un tablero/dashboards que muestre variables ambientales en tiempo real e indique alertas con íconos o código de color. | 80 | Dev Team, QA | - Sistema de alertas funcional  - Tablero que visualiza temperatura/humedad de cada galpón  - Notificaciones automáticas a usuarios habilitados |
| **8. Sprint 8: Control Remoto de Dispositivos** | Caso de uso “Control Remoto de Dispositivos”: encender/apagar/regular ventiladores, aspersores, cortinas. Mostrar estado actual (encendido, potencia). Opcionalmente, vincular alertas con acciones automáticas (si temp > X, encender ventilador). | 80 | Dev Team (IoT), QA | - Interfaz para manipular dispositivos remotamente  - Bitácora que registra la acción (fecha, usuario, dispositivo)  - Integración con capa IoT de salida |
| **9. Sprint 9: Reportes Históricos y Auditoría** | Generar informes sobre mortalidad, pesaje y condiciones ambientales en determinados rangos de fechas o galpones. Habilitar consulta de bitácora para revisar acciones importantes (creación de camadas, control remoto, cambios de umbrales). Opción de exportar PDF/CSV. | 70 | Dev Team, QA | - Módulo de reportes con filtros (galpón, fechas, etc.)  - Registro detallado de acciones (auditoría)  - Exportación de datos (si así se definió) |
| **10. Sprint 10: Pruebas de Integración (Parte 1)** | Realizar la primera tanda de pruebas exhaustivas en todos los módulos: usuarios, camadas, sensores, control de dispositivos y reportes. Se corrigen bugs críticos y se identifican áreas de optimización de base de datos o refactor de código. | 80 | Tester QA, Dev Team | - Versión parcial del sistema con bugs críticos resueltos  - Evidencias de QA y conclusiones iniciales sobre rendimiento |
| **11. Sprint 11: Pruebas de Integración (Parte 2) y Ajustes Finales** | Segunda ronda de pruebas integrales y optimizaciones, cubriendo escenarios complejos de seguridad o rendimiento. Se corrigen fallos restantes, se pule la documentación técnica para alinear con la implementación. | 80 | Tester QA, Dev Team | - Sistema estable y libre de errores críticos  - Informe final de QA (escenarios y resultados)  - Base de datos y lógica optimizadas |
| **12. Sprint 12: Cierre, Documentación y Puesta en Producción** | Implementación final en ambiente de producción (servidores, sensores reales, accesos), pruebas de aceptación con usuarios (UAT), capacitación y entrega de documentación integral (manuales de usuario, manual técnico, memorias de cálculo). | 80 | Project Manager, Soporte Técnico, QA, Dev Team | - Sistema en pleno funcionamiento real  - Usuarios capacitados  - Documentación final (técnica, funcional y académica)  - Plan de sostenibilidad y mejoras futuras |

### 

## VII.3 - Sprints

### Sprint 1: Inicio y Organización

**Objetivo principal**: Sentar las bases del proyecto, preparar el entorno de trabajo y refinar el Product Backlog.

**Fecha inicio:** 01 de Agosto 2024

1. **Actividades**
   * **Configuración inicial** Crear repositorio de código, definir la estructura de carpetas y pipeline de integración.  
      Seleccionar herramientas para el tablero Kanban (Jira, Trello, GitHub Projects, etc.).
   * **Revisión de Alcance** Analizar en detalle los requerimientos y casos de uso (CU) contemplados en el informe.  
      Convertirlos en historias de usuario, priorizarlas junto al Product Owner.
   * **Arquitectura preliminar** Identificar la tecnología de base de datos, frameworks para frontend y backend, y consideraciones de IoT.
   * **Planificación del Sprint** Seleccionar tareas introductorias factibles en 2 semanas (maquetado de la primera pantalla, set-up de microservicio, etc.).
2. **Uso de Kanban/Scrum en este sprint**
   * **Tablero Kanban**: Se crean las tarjetas iniciales ("Definir backlog", "Configurar repositorio", etc.).
   * **Daily Scrum**: Primera reunión diaria para alinear expectativas y resolver impedimentos.
3. **Entregables**
   * Product Backlog priorizado y detallado (casos de uso transformados en historias de usuario).
   * Entorno de desarrollo configurado.
   * Documento breve de arquitectura base (tecnologías principales, esquema preliminar de BD).
4. **Ceremonias de cierre**
   * **Sprint Review**: Presentación de la organización lograda, el backlog y la configuración del entorno.
   * **Sprint Retrospective**: Ajustar cualquier mejora para la colaboración y la comunicación inicial.

### Sprint 2: Manejo de Usuarios y Seguridad Básica

**Objetivo principal**: Implementar la autenticación y la estructura de roles, dejando lista la base para la interacción de diferentes tipos de usuario (Operario, Gerente, Administrador).

**Fecha inicio:** 15 de Agosto 2024

1. **Actividades**
   * **Caso de uso “Iniciar Sesión” (Autenticación)** Implementar registro y login, con cifrado de contraseñas, validaciones de roles y control de sesión.
   * **Caso de uso “Manejo de Usuarios”** Alta, baja, modificación de usuarios; asignar roles (Operario, Gerente, Admin).
   * **Diseño DB inicial** Creación de tablas Usuarios, Roles, y configuración de la conexión.
   * **Seguridad y validaciones** Encriptar credenciales y definir tiempo de expiración de sesión, si corresponde.
2. **Uso de Kanban/Scrum en este sprint**
   * **Sprint Planning**: Se eligen historias de usuario relacionadas a seguridad y creación de roles.
   * **Tablero Kanban**: Tarjetas como “Crear tabla Usuarios”, “Programar Endpoint de Login”, “Probar validaciones de contraseñas”.
3. **Entregables**
   * Módulo de autenticación completo.
   * Gestión de roles y permisos, interfaz mínima para crear/editar usuarios.
   * Documentación o diagrama de secuencia que muestre el flujo de login y asignación de roles.
4. **Ceremonias de cierre**
   * **Sprint Review**: Presentar al PO cómo se crea un usuario y se valida la sesión.
   * **Retrospective**: Ajustes en la coordinación del equipo, definición de tareas, etc.

### Sprint 3: Estructura de Datos y Configuración de Galpones

**Objetivo principal**: Definir el modelo de datos detallado para las entidades principales (Galpones, Camadas, etc.) e implementar la gestión básica de galpones.

**Fecha inicio:** 29 de Agosto 2024

1. **Actividades**
   * **Modelo de datos** Refinar la base de datos con tablas Galpones, Camadas, Sensores, Alertas, etc.
   * **Caso de uso “Configurar Galpones”** Dar de alta, baja y modificar la descripción, capacidad y ubicación de cada galpón.
   * **Casos de uso auxiliares** Asociar sensores a un galpón si el informe lo contempla.
2. **Uso de Kanban/Scrum en este sprint**
   * En el **Planning**, se añaden las historias de “Registrar galpones” y la creación del esquema principal de BD.
   * Durante el sprint, las tareas se visualizan en el tablero Kanban (“Diseñar entidad Galpones”, “CRUD de Galpones”, “Probar migraciones DB”).
3. **Entregables**
   * BD actualizada con las entidades clave.
   * Pantalla/endpoint de alta y edición de galpones (con validaciones de capacidad).
4. **Ceremonias de cierre**
   * **Sprint Review**: Mostrar la estructura de la BD y cómo se gestionan los galpones en la interfaz.
   * **Retrospective**: Discutir la coordinación del equipo y la claridad del backlog.

### Sprint 4: Gestión de Camadas y Datos Productivos (Parte 1)

**Objetivo principal**: Iniciar la implementación de los casos de uso clave sobre la crianza de aves, con énfasis en la lógica de camadas (creación y datos básicos).

**Fecha inicio:** 12 de Septiembre 2024

1. **Actividades**
   * **Caso de uso “Registrar Nueva Camada”** Datos de fecha de ingreso, cantidad de aves, proveedor, asignación a galpones.
   * **Validaciones iniciales** Comprobar capacidad máxima de un galpón antes de asignar aves, verificación de datos obligatorios.
   * **Diseño parcial de formularios** Frontend para registrar camadas, guardarlas en la BD.
2. **Uso de Kanban/Scrum**
   * **Sprint Planning**: Se priorizan sub-historias de “Camadas” que no excedan las 80 horas planificadas.
   * **Tablero Kanban**: Tareas como “Backend – Alta de camada”, “Frontend – Formulario de Nueva Camada”, “Validaciones de capacidad”.
3. **Entregables**
   * CRUD parcial de camadas (alta, edición inicial).
   * BD actualizada con primeros datos de camadas.
4. **Ceremonias de cierre**
   * **Sprint Review**: Demostración de cómo se crea una camada, con la BD actualizada.
   * **Retrospective**: Ajustar la estimación del próximo sprint, validando si el alcance fue demasiado amplio o escaso.

### Sprint 5: Gestión de Camadas y Datos Productivos (Parte 2)

**Objetivo principal**: Completar la lógica de camadas con el registro de mortalidad, pesos y demás validaciones de negocio.

**Fecha inicio:** 26 de Septiembre 2024

1. **Actividades**
   * **Caso de uso “Registrar Muertes de Aves”** Seleccionar galpón y causa de fallecimiento, descontar del total de la camada, validaciones de rangos.
   * **Caso de uso “Registrar Pesos”** Tomar el peso promedio de muestras, asociar al galpón y la camada, verificación de valores lógicos.
   * **Finalizar CRUD de Camadas** Contemplando el ciclo de vida de una camada (apertura, edición, cierre).
2. **Uso de Kanban/Scrum**
   * **Sprint Planning**: Se incorporan los pendientes de mortalidad y peso que no cupieron en el sprint anterior.
   * **Tablero Kanban**: “Backend – Mortalidad”, “Frontend – Formulario de Pesos”, “Rango de peso plausible”.
3. **Entregables**
   * Funcionalidades CRUD completas para camadas (mortalidad, pesaje, fin de camada).
   * Pantallas donde el operario registra datos en tiempo real.
4. **Ceremonias de cierre**
   * **Sprint Review**: Demostración de la carga de mortalidad y pesos, verificación de reglas de negocio.
   * **Retrospective**: Ajustar flujo de trabajo y asignación de tareas en caso de saturación.

### Sprint 6: Integración IoT Básica (Lectura de Sensores)

**Objetivo principal**: Comenzar con la lectura real de sensores de temperatura y humedad, o un simulador si todavía no se dispone del hardware.

**Fecha inicio:** 10 de Octubre 2024

1. **Actividades**
   * **Prototipo de lectura de datos** Escoger protocolo (ModBus, MQTT, etc.), conectar un servicio que lea la información y guarde en la BD.
   * **Asociar datos de sensores** Incluir campo “galpónId” o similar para relacionar la lectura con su ubicación física.
   * **Pruebas y validaciones** Verificar frecuencia de lectura, control de errores (sensor desconectado).
2. **Uso de Kanban/Scrum**
   * **Daily Scrum**: Se presta especial atención a la configuración de hardware y la integración con el backend.
   * **Tablero Kanban**: Tareas de “Programar servicio de recolección”, “Probar con sensor real/simulado”, “Registrar lecturas en la BD”.
3. **Entregables**
   * Código que se ejecute periódicamente para leer datos de los sensores.
   * Registro automático de temperatura/humedad en la base de datos, con timestamps y referencia al galpón.
4. **Ceremonias de cierre**
   * **Sprint Review**: Demostrar al PO cómo el sistema recibe lecturas.
   * **Retrospective**: Evaluar la integración con hardware y resolver cuellos de botella.

### Sprint 7: Alertas y Tablero Ambiental

**Objetivo principal**: Agregar lógica de alertas al detectar umbrales superados, y presentar un tablero o dashboards con el estado actual de cada galpón.

**Fecha inicio:** 24 de Octubre 2024

1. **Actividades**
   * **Caso de uso “Configurar Alertas”** Definir umbrales de temperatura/humedad por galpón o por rol (Gerente).
   * **Generación de alertas** Al exceder los valores, disparar notificaciones (correo, notificación web, etc.).
   * **Tablero ambiental** Pantalla con la lectura actual de cada galpón, indicando color o icono en caso de alerta.
2. **Uso de Kanban/Scrum**
   * **Sprint Planning**: El PO prioriza la creación de un panel visual central.
   * **Tablero Kanban**: “Configurar umbrales”, “Diseño UI del tablero”, “Envío de alerta por email”.
3. **Entregables**
   * Sistema que registra la última lectura de cada galpón y la compara con los umbrales configurados.
   * Notificaciones (p. ej. “Temperatura Alta – Galpón 2: 34°C”) a usuarios habilitados.
   * Tablero en la interfaz que muestre información consolidada de varios galpones.
4. **Ceremonias de cierre**
   * **Sprint Review**: Presentación de la demo, simulando un valor elevado.
   * **Retrospective**: Ajustar reglas (ej. humedad vs. ventilador) y UX del tablero.

### Sprint 8: Control Remoto de Dispositivos

**Objetivo principal**: Implementar la funcionalidad de activar, desactivar o regular dispositivos de climatización (ventiladores, aspersores, cortinas).

**Fecha inicio:** 7 de Noviembre 2024

1. **Actividades**
   * **Caso de uso “Control Remoto de Dispositivos”** Integrar las acciones con la plataforma IoT, permitiendo encender, apagar o regular velocidad.
   * **Estado del dispositivo** El tablero muestra si el dispositivo está encendido/apagado, potencia, etc.
   * **Reglas automáticas (opcional)** Vincular alertas con acciones (si temp > X, encender ventilador).
2. **Uso de Kanban/Scrum**
   * **En el Planning**, se incluyen historias de “Accionar ventilador”, “Visualizar estado del dispositivo”.
   * **Tablero Kanban**: “Tarea en curso”, “QA”, “Hecho”.
3. **Entregables**
   * Botón o interfaz donde (Operario/Gerente) enciende/apaga remotamente.
   * Registro de bitácora de cada acción (quién y cuándo).
4. **Ceremonias de cierre**
   * **Sprint Review**: Demostración real/simulada de dispositivos.
   * **Retrospective**: Ajustes en la priorización de tareas según feedback.

### Sprint 9: Reportes Históricos y Auditoría

**Objetivo principal**: Permitir la generación de informes y la revisión de acciones importantes (bitácora).

**Fecha inicio:** 21 de Noviembre 2024

1. **Actividades**
   * **Caso de uso “Generar Informes”** Reportes de mortalidad, pesos, comparativas de temperatura, etc.
   * **Bitácora de acciones (auditoría)** Registrar toda acción relevante y consultarla con filtros (fecha, usuario, tipo de acción).
   * **Exportación** Posibilidad de exportar informes a PDF/CSV, si se definió en el proyecto.
2. **Uso de Kanban/Scrum**
   * Se crean historias como “Reporte Mortalidad por galpón”, “Exportar PDF”, “Consultar Bitácora”.
   * Se definen en el Sprint Planning y se gestionan en el tablero.
3. **Entregables**
   * Pantallas donde el usuario filtra datos (rango de fechas, galpón) y ve gráficos/tablas.
   * Módulo de bitácora (registro de acciones) con búsqueda y filtrado.
   * Exportación de datos, si procede.
4. **Ceremonias de cierre**
   * **Sprint Review**: El PO verifica la utilidad de los reportes para la toma de decisiones.
   * **Retrospective**: Ajustar problemas de rendimiento, usabilidad, etc.

### Sprint 10: Pruebas de Integración (Parte 1)

**Objetivo principal**: Realizar la primera tanda de pruebas exhaustivas, abarcando todos los módulos en conjunto, corrigiendo errores críticos y evaluando la cohesión del sistema.

**Fecha inicio:** 5 de Diciembre 2024

1. **Actividades**
   * **Pruebas integrales** Revisar manejo de usuarios, camadas, sensores, control de dispositivos, reportes.
   * **Optimización inicial** Analizar rendimiento de BD, refactorizar partes del código si se hallan cuellos de botella.
   * **Corrección de bugs prioritarios** Registrar incidencias en el tablero, priorizarlas y resolverlas.
2. **Uso de Kanban/Scrum**
   * **Daily Scrum**: Orientado a detección de problemas y priorización de correcciones.
   * **Tablero Kanban**: Las tarjetas se llenan con incidencias encontradas en pruebas de integración.
3. **Entregables**
   * Versión mejorada del sistema, con parte de los bugs críticos resueltos.
   * Evidencias de pruebas (QA) y conclusiones iniciales.
4. **Ceremonias de cierre**
   * **Sprint Review**: Presentar las mejoras de rendimiento y la reducción de errores.
   * **Retrospective**: Ajustar planeamiento para la siguiente ronda de pruebas y optimizaciones.

### Sprint 11: Pruebas de Integración (Parte 2) y Ajustes Finales

**Objetivo principal**: Completar la segunda ronda de pruebas integrales y optimizaciones, preparando el sistema para su puesta en producción sin errores importantes.

**Fecha inicio:** 19 de Diciembre 2024

1. **Actividades**
   * **Segunda fase de pruebas** Retomar incidencias pendientes, probar escenarios más complejos, validaciones de seguridad.
   * **Corrección final de errores** Depurar fallos que queden e implementar mejoras de rendimiento.
   * **Revisión general** Validar la coherencia de la documentación técnica con lo implementado.
2. **Uso de Kanban/Scrum**
   * **Sprint Planning**: Se arrastran las incidencias remanentes y se planifica la optimización final.
   * **Tablero Kanban**: Se mantiene la priorización de bugs, dividiendo las tareas entre el equipo.
3. **Entregables**
   * Sistema estable y libre de errores críticos.
   * Informe final de QA con escenarios y resultados.
   * Base de datos y lógica optimizadas para el despliegue.
4. **Ceremonias de cierre**
   * **Sprint Review**: Mostrar la versión final “pre-producción”.
   * **Retrospective**: Ajustar últimos detalles de cara al despliegue y la documentación.

### Sprint 12: Cierre, Documentación y Puesta en Producción

**Objetivo principal**: Realizar la implementación final en el entorno definitivo, pruebas de aceptación con usuarios, capacitación y entrega de la documentación integral del proyecto.

**Fecha inicio:** 02 de Enero 2025

1. **Actividades**
   * **Pruebas de aceptación con usuarios (UAT)** Operarios y gerente usan el sistema en un entorno casi productivo, detectando mejoras finales.
   * **Despliegue en producción** Configurar servidores, sensores definitivos, accesos y pruebas en vivo.
   * **Capacitación** Manuales de usuario (operarios, gerente) y capacitaciones prácticas.
   * **Documentación final** Manual técnico, diagramas de arquitectura y base de datos, especificación de casos de uso, memorias de cálculo.
2. **Uso de Kanban/Scrum**
   * En este sprint, se reduce el desarrollo y aumentan las tareas de **documentación, pruebas finales y entrenamiento**.
   * Las tareas de despliegue son prioridad absoluta en el tablero Kanban.
3. **Entregables**
   * Sistema completo y funcionando en producción, cumpliendo todos los requerimientos.
   * Documentación final (funcional, técnica y académica), esencial para la presentación del proyecto.
   * Plan de sostenibilidad y mejora, con proyecciones a futuro.
4. **Ceremonias de cierre**
   * **Sprint Review Final**: El Product Owner y stakeholders confirman que el producto cumple los objetivos.
   * **Retrospective Final**: Se extraen lecciones aprendidas y se sientan bases para posibles evoluciones.

### Diagrama de Gantt - Cronograma de Sprints

## VII.4 - Revisiones

### Revisiones técnicas de ingeniería de sistemas

En cumplimiento de las buenas prácticas recomendadas por el **SEBoK (v2.8)**, se incorporaron tres revisiones técnicas clave a lo largo del ciclo de vida del proyecto:

| **Revisión técnica** | **Sprint** | **Objetivo principal** |
| --- | --- | --- |
| **PDR – Preliminary Design Review** | Entre Sprint 2 y 3 | Validar que los requisitos funcionales y la arquitectura preliminar satisfacen las necesidades de los interesados. Incluye revisión del modelo de datos, APIs y diagramas de arquitectura. |
| **CDR – Critical Design Review** | Entre Sprint 5 y 6 | Confirmar que el diseño detallado del sistema (backend, frontend, IoT, controladores) está completo y preparado para ser implementado sin riesgos críticos. |
| **TRR – Test Readiness Review** | Antes del Sprint 10 | Evaluar si el sistema está listo para pruebas integradas. Se revisan planes de prueba, cobertura, herramientas de QA y definición de criterios de aceptación para entorno real. |

Estas revisiones serán documentadas internamente como checkpoints obligatorios, asegurando que el equipo avance solo cuando se cumplan los criterios técnicos y funcionales predefinidos.

# Capítulo VIII - Análisis de riesgos

La gestión de riesgos constituye un pilar clave en la planificación y ejecución del sistema de gestión avícola, ya que permite anticipar, evaluar y mitigar los eventos que puedan afectar negativamente el cumplimiento de los objetivos del proyecto. Dado el entorno rural, la dependencia tecnológica y los múltiples actores involucrados (operarios, veterinarios, proveedores, sistemas físicos y digitales), se vuelve esencial establecer un marco de trabajo riguroso para la identificación, análisis, seguimiento y control de riesgos.

Este capítulo presenta el plan de gestión de riesgos, la metodología aplicada, las herramientas utilizadas, el análisis de riesgos cualitativo y la planificación de respuestas, incluyendo contingencias concretas para los eventos más críticos.

## VIII.1 - Plan de Gestión de Riesgos

La gestión de riesgos en este proyecto se estructura según las prácticas recomendadas por el **PMBOK 7.ª edición** y el enfoque de ingeniería de sistemas del **SEBoK**, estableciendo un proceso iterativo que incluye:

* Identificación sistemática de riesgos.
* Evaluación cualitativa mediante escalas de probabilidad e impacto.
* Desarrollo de estrategias de respuesta (evitar, mitigar, transferir, aceptar).
* Planes de contingencia.
* Monitoreo continuo y adaptación durante la ejecución del sistema.

### VIII.1.1 - Enfoque a Utilizar

El enfoque se compone de las siguientes fases:

1. **Identificación de riesgos:** Se detectan posibles amenazas mediante revisión de documentación técnica, entrevistas con stakeholders, análisis de supuestos, checklists, y diagramas de causa-efecto (Ishikawa).
2. **Análisis cualitativo:** Se asigna a cada riesgo una probabilidad e impacto estimado, clasificándolo y priorizándolo.
3. **Planificación de respuestas:** Se definen estrategias y responsables para abordar cada riesgo.
4. **Control de riesgos:** Se revisa periódicamente la evolución de los riesgos durante los sprints, incorporando nuevas amenazas o actualizando las respuestas planificadas.

### VIII.1.2 - Categorización

Los riesgos identificados se agrupan en cinco grandes categorías:

* **Técnicos:** Relacionados con sensores, dispositivos IoT, pérdida de conectividad, errores de medición, etc.
* **Operacionales:** Vinculados a procesos de registro manual, errores humanos, carga de datos erróneos.
* **Del proyecto:** Incluye riesgos en la planificación, falta de recursos, desvinculación de roles clave, y fallas de integración.
* **Externos y ambientales:** Fenómenos meteorológicos extremos, fallas de energía, falta de conectividad rural (ej. caída de Starlink).
* **Sanitarios:** Brotes de enfermedades en aves o restricciones sanitarias externas que afecten el funcionamiento de la granja.

A continuación, se presentan las plantillas así como la matriz de riesgos:

**Plantilla de Registro de Riesgos**

| Código del Riesgo | |
| --- | --- |
| Nombre del Riesgo | |
| Descripción del Riesgo |  |
| Causa Raíz |  |
| Entregables Afectados |  |
| Objetivo Afectado |  |
| Estimación de Probabilidad |  |
| Estimación de Impacto |  |
| Nivel de riesgo (Probabilidad x Impacto) |  |
| Tipo de Riesgo (cualitativo) |  |

**Plantilla de Respuestas y Plan de Contingencias**

| Código del Riesgo | |
| --- | --- |
| Trigger |  |
| Respuestas Planificadas |  |
| Tipo de Respuestas |  |
| Plan de Contingencias |  |

**Matriz de Probabilidad x Impacto**

| Amenazas | | | | | |
| --- | --- | --- | --- | --- | --- |
| Impacto /Probabilidad | Muy bajo | Bajo | Medio | Alto | Muy alto |
| Muy alta | .05 | .09 | .18 | .36 | .72 |
| Alta | .04 | .07 | .14 | .28 | .56 |
| Mediana | .03 | .05 | .1 | .2 | .4 |
| Baja | .02 | .03 | .06 | .12 | .24 |
| Muy baja | .01 | .01 | .02 | .04 | .08 |

### VIII.1.3 - Formatos de Documentos

Se utilizarán las siguientes plantillas:

* **Registro de riesgos:** Con código, descripción, causa raíz, entregables y objetivos afectados, probabilidad, impacto, y tipo de riesgo.
* **Plan de respuesta y contingencia:** Incluye trigger, responsable, respuesta prevista, tipo de estrategia, y plan de contingencia.
* **Matriz de probabilidad e impacto:** Herramienta visual para facilitar la priorización.

### VIII.1.4 - Escala de Medición para Probabilidades

Se utilizará una escala de 5 niveles (Muy baja a Muy alta), con valores asignados de 0.1 a 0.9.

### VIII.1.5 - Escala de Medición para Impactos

La escala de impacto considera las dimensiones de **costo, tiempo, alcance y calidad**. Se establecen cinco niveles (Muy bajo a Muy alto), y se aplican valores que reflejan el potencial perjuicio para el proyecto.

Escala de medición para probabilidades e impactos

| Probabilidad | Valor Numérico | Impacto | Valor Numérico |
| --- | --- | --- | --- |
| Muy improbable | 0.1 | Muy bajo | 0.05 |
| Relativamente probable | 0.3 | Bajo | 0.1 |
| Probable | 0.5 | Moderado | 0.2 |
| Muy probable | 0.7 | Alto | 0.4 |
| Casi Certeza | 0.9 | Muy alto | 0.8 |

Tabla 6.1: Lista de probabilidades e impactos que se utilizaran para el análisis de los riesgos en esta sección

## VIII.2 - Identificación de los Riesgos

La identificación de riesgos constituye un proceso sistemático y proactivo para reconocer los eventos que podrían afectar negativamente o positivamente el desarrollo del sistema de gestión avícola. Dada la complejidad inherente a los entornos de producción intensiva, así como las múltiples dependencias tecnológicas y ambientales involucradas, esta fase es crítica para asegurar una gestión de riesgos eficiente durante todo el ciclo de vida del proyecto.

La identificación de riesgos se realiza de manera iterativa, considerando diversas fuentes de información como la planificación del proyecto, especificaciones técnicas, experiencia de expertos, análisis de supuestos y revisiones periódicas de contexto.

La información obtenida se formaliza en el Registro de Riesgos, que servirá de base para los análisis posteriores.

### VIII.2.1 - Actividades para la Identificación de Riesgos

Para lograr una identificación exhaustiva y precisa, se implementaron las siguientes actividades:

* Revisión de la documentación técnica y de planificación.
* Realización de entrevistas y sesiones de brainstorming con stakeholders clave (gerente, operarios, veterinarios, proveedor).
* Análisis de supuestos críticos del proyecto.
* Aplicación de un checklist de verificación para asegurar cobertura completa.
* Elaboración de Análisis Bowtie para eventos críticos.

### VIII.2.2 - Análisis de Bowtie

Se optó por la metodología **Bowtie Analysis**, la cual resulta más adecuada para el contexto del proyecto. Esta herramienta permite representar de manera gráfica:

* Las **causas** que pueden conducir a un evento crítico.
* Las **consecuencias** derivadas si dicho evento se materializa.
* Los **controles preventivos** diseñados para evitar que el evento ocurra.
* Los **controles de mitigación** para reducir el impacto si el evento se concreta.

Para cada riesgo crítico se elaborará un Bowtie, representando las amenazas principales al proyecto, entre las cuales se destacan:

* Fallo de conectividad rural prolongado (ej. caída de la red Starlink o falla de red mesh).
* Fallo de sensores de temperatura y humedad (lecturas erróneas o inactividad).
* Corte de energía prolongado en los galpones.
* Errores en la carga manual de mortalidad y pesajes.
* Brotes de enfermedad en aves y limitaciones sanitarias externas.

Cada uno de estos eventos será analizado identificando causas raíces y consecuencias potenciales, permitiendo el diseño de estrategias de respuesta proactivas.

### VIII.2.3 - Análisis de Supuestos

Se realizó una evaluación crítica de los principales supuestos en los que se apoya el proyecto. Entre los supuestos analizados se incluyen:

**Supuesto 1:** Estabilidad del suministro eléctrico en la zona rural

Riesgo asociado: Cortes de energía que comprometan el funcionamiento de sensores, actuadores o nodos de red, afectando todo el sistema.

Acciones sugeridas: Estudio de factibilidad para instalar UPS en puntos críticos, priorización de tableros eléctricos bien protegidos.

**Supuesto 2:** Condiciones climáticas dentro de parámetros normales

Riesgo asociado: Eventos extremos (vientos fuertes, granizo, lluvias prolongadas) que dañen la infraestructura de red o los galpones.

Acciones sugeridas: Refuerzo de instalaciones exteriores, configuración de alertas meteorológicas preventivas y documentación de protocolos de respuesta ante emergencias climáticas.

Cada supuesto fue evaluado para identificar posibles desviaciones que pudieran materializar riesgos relevantes, estableciendo planes de contingencia cuando fuera necesario.

### VIII.2.4 - Registro Preliminar de Riesgos

Como resultado de las actividades de identificación, se elaboró un **Registro Preliminar de Riesgos** que documenta cada amenaza reconocida, indicando:

* Descripción breve.
* Causa raíz.
* Entregables afectados.
* Objetivo del proyecto impactado.
* Estimaciones preliminares de probabilidad e impacto.

Este registro será utilizado como insumo en la fase de Análisis Cualitativo del Riesgo.

| Código del Riesgo R01 | |
| --- | --- |
| Fallo de conectividad en la granja | |
| Descripción del Riesgo | Pérdida de conexión a Internet en la granja, afectando la transmisión de datos de sensores y el control remoto de dispositivos. |
| Causa Raíz | Caída del servicio Starlink, problemas en la red interna, cortes de energía prolongados. |
| Entregables Afectados | Sistema de monitoreo en tiempo real, dashboards de visualización de datos, alertas automáticas. |
| Objetivo Afectado | Disponibilidad y continuidad operativa del sistema. |

| Código del Riesgo R02 | |
| --- | --- |
| Fallo en sensores ambientales | |
| Descripción del Riesgo | Lectura incorrecta o pérdida total de mediciones de temperatura y humedad en galpones. |
| Causa Raíz | Mal funcionamiento de sensores, desconexión de dispositivos IoT, interferencias eléctricas o climáticas. |
| Entregables Afectados | Sistema de control ambiental, generación de alertas, informes diarios. |
| Objetivo Afectado | Fiabilidad del monitoreo ambiental y calidad de las condiciones de crianza. |

| Código del Riesgo R03 | |
| --- | --- |
| Corte de energía eléctrica en galpones | |
| Descripción del Riesgo | Interrupción prolongada del suministro eléctrico que afecta dispositivos y nodos de red. |
| Causa Raíz | Fallas en la red eléctrica rural, tormentas severas, falta de backup de energía local. |
| Entregables Afectados | Funcionamiento de sensores, dispositivos de control (aspersores, ventiladores, malacates). |
| Objetivo Afectado | Continuidad de operación y protección del bienestar animal. |

| Código del Riesgo R04 | |
| --- | --- |
| Error humano en carga de datos críticos | |
| Descripción del Riesgo | Registro incorrecto de mortalidad diaria, pesos promedio u otras variables relevantes por parte de los operarios. |
| Causa Raíz | Capacitación insuficiente, errores de interpretación, falta de validaciones en el sistema. |
| Entregables Afectados | Base de datos histórica, informes de desempeño, alertas automatizadas. |
| Objetivo Afectado | Precisión de la trazabilidad y calidad de la información de gestión. |

| Código del Riesgo R05 | |
| --- | --- |
| Brote de enfermedad aviar. | |
| Descripción del Riesgo | Aparición de enfermedades infecciosas en las aves que afecten la producción o generen cierres sanitarios. |
| Causa Raíz | Ingreso de patógenos externos, deficiencias en bioseguridad, transmisión aérea o vectorial. |
| Entregables Afectados | Camada de aves en producción, reportes sanitarios, indicadores de desempeño. |
| Objetivo Afectado | Continuidad productiva y cumplimiento de normas sanitarias. |

| Código del Riesgo R06 | |
| --- | --- |
| Fallo en dispositivos de control de ambiente | |
| Descripción del Riesgo | Mal funcionamiento de malacates eléctricos, aspersores, ventiladores o iluminación, afectando el control ambiental automático. |
| Causa Raíz | Fallos mecánicos, desconexiones, sobrecarga eléctrica o errores de configuración. |
| Entregables Afectados | Sistema de climatización automatizado, bienestar de las aves, reportes de estado de dispositivos. |
| Objetivo Afectado | Calidad ambiental interna y eficiencia operativa. |

| Código del Riesgo R07 | |
| --- | --- |
| Retrasos en la disponibilidad de repuestos o servicios técnicos | |
| Descripción del Riesgo | Demoras en la reposición de equipos críticos o en la asistencia técnica por problemas logísticos o falta de stock. |
| Causa Raíz | Ubicación rural remota, proveedores de servicios tercerizados, escasez de componentes electrónicos. |
| Entregables Afectados | Mantenimiento correctivo de dispositivos, tiempos de recuperación ante fallos. |
| Objetivo Afectado | Continuidad del servicio y tiempo de respuesta ante incidentes. |

| Código del Riesgo R08 | |
| --- | --- |
| Error de sincronización entre datos locales y nube | |
| Descripción del Riesgo | Pérdida o inconsistencia de datos entre la base local de las Raspberry Pi y el almacenamiento en la nube. |
| Causa Raíz | Fallos en la programación de sincronización, cortes intermitentes de conexión, problemas de buffer de datos. |
| Entregables Afectados | Base de datos consolidada, dashboards históricos, trazabilidad de eventos. |
| Objetivo Afectado | Integridad de los registros históricos y calidad de la información analítica. |

| Código del Riesgo R09 | |
| --- | --- |
| Fallos de seguridad informática | |
| Descripción del Riesgo | Vulnerabilidades de ciberseguridad que permitan accesos no autorizados o modificaciones de datos críticos. |
| Causa Raíz | Fallas en la configuración de AWS IoT, gestión inadecuada de certificados digitales, accesos poco restringidos. |
| Entregables Afectados | Integridad de la plataforma, confidencialidad de datos, cumplimiento de políticas de seguridad. |
| Objetivo Afectado | Seguridad del sistema y confianza de los stakeholders. |

| Código del Riesgo R10 | |
| --- | --- |
| Eventos climáticos extremos | |
| Descripción del Riesgo | Impacto de tormentas severas, granizo o vientos intensos sobre las instalaciones físicas de la granja. |
| Causa Raíz | Fenómenos meteorológicos fuera de rango normal, aumento de eventos climáticos extremos. |
| Entregables Afectados | Infraestructura de galpones, antenas de red, dispositivos exteriores. |
| Objetivo Afectado | Continuidad física de la operación y necesidad de inversiones de recuperación. |

| Código del Riesgo R11 | |
| --- | --- |
| Desvinculación de un desarrollador clave del equipo | |
| Descripción del Riesgo | Salida imprevista de un miembro responsable del backend o frontend durante etapas críticas del proyecto. |
| Causa Raíz | Cambios laborales, problemas personales, desmotivación, falta de incentivos o sobrecarga. |
| Entregables Afectados | Módulos de software incompletos, retraso en integración de funcionalidades. |
| Objetivo Afectado | Cumplimiento del cronograma y continuidad del desarrollo. |

| Código del Riesgo R12 | |
| --- | --- |
| Subestimación de tiempos de desarrollo | |
| Descripción del Riesgo | Planificación demasiado optimista en cuanto a la duración de tareas técnicas complejas (como integración con sensores o interfaz de usuario). |
| Causa Raíz | Falta de experiencia previa en proyectos similares, presión por cumplir fechas o falta de márgenes de contingencia. |
| Entregables Afectados | Calendario general del proyecto, fechas de entrega intermedias. |
| Objetivo Afectado | Cumplimiento del cronograma y reputación del equipo. |

| Código del Riesgo R13 | |
| --- | --- |
| Desalineación de objetivos entre stakeholders y equipo técnico | |
| Descripción del Riesgo | Diferencias en las expectativas de lo que el sistema debe entregar entre el equipo desarrollador y el cliente. |
| Causa Raíz | Comunicación insuficiente, falta de validación de requisitos o cambios no documentados. |
| Entregables Afectados | Funcionalidades del sistema, diseño de interfaz, reportes. |
| Objetivo Afectado | Satisfacción del cliente y cumplimiento del alcance funcional. |

| Código del Riesgo R14 | |
| --- | --- |
| Falta de visibilidad del avance del proyecto | |
| Descripción del Riesgo | Ausencia de reportes periódicos o seguimiento riguroso del estado del proyecto, lo que impide detectar desviaciones a tiempo. |
| Causa Raíz | No utilización de herramientas de gestión (Jira, tableros ágiles), falta de reuniones de revisión o roles poco definidos. |
| Entregables Afectados | Supervisión del cumplimiento de hitos y entregables. |
| Objetivo Afectado | Gestión efectiva y monitoreo del progreso. |

## VIII.3 - Análisis Cualitativo del Riesgo

En este apartado se evalúan los riesgos asociados al proyecto, tanto en términos de amenazas como de oportunidades, con el fin de garantizar su éxito. Para ello, se emplea la matriz de riesgos presentada en el plan de gestión de riesgos, la cual asigna probabilidades y niveles de impacto a cada escenario posible.

Esta herramienta permite medir y priorizar los riesgos de manera sistemática, facilitando la toma de decisiones para mitigar posibles amenazas y aprovechar las oportunidades de mejora. 121 Para realizar el análisis cualitativo de los riesgos se utiliza la plantilla de registro de riesgos presentada en el plan de gestión de riesgos, la cual contiene secciones referidas tanto a la probabilidad de ocurrencia como al impacto y se clasifica a los riesgos en base al producto de estos valores.

| Riesgo R01 – Pérdida de conectividad prolongada | |
| --- | --- |
| Estimación de Probabilidad | alta (0.7) |
| Estimación de Impacto | muy alto (0.8) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.56 |
| Tipo de Riesgo (cualitativo) | Alto |

## 

| Riesgo R02 – Fallo en sensores ambientales | |
| --- | --- |
| Estimación de Probabilidad | media (0.5) |
| Estimación de Impacto | alto (0.4) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.20 |
| Tipo de Riesgo (cualitativo) | Moderado |

## 

| Riesgo R03 – Corte de energía eléctrica en galpones | |
| --- | --- |
| Estimación de Probabilidad | media (0.5) |
| Estimación de Impacto | muy alto (0.8) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.40 |
| Tipo de Riesgo (cualitativo) | Alto |

## 

| Riesgo R04 – Error humano en carga de datos | |
| --- | --- |
| Estimación de Probabilidad | media (0.5) |
| Estimación de Impacto | alto (0.7) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.35 |
| Tipo de Riesgo (cualitativo) | Alto |

## 

| Riesgo R05 – Brote de enfermedad aviar | |
| --- | --- |
| Estimación de Probabilidad | baja (0.3) |
| Estimación de Impacto | muy alto (0.8) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.24 |
| Tipo de Riesgo (cualitativo) | Moderado |

## 

| Riesgo R06 – Fallo en dispositivos de control ambiental | |
| --- | --- |
| Estimación de Probabilidad | media (0.5) |
| Estimación de Impacto | alto (0.4) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.20 |
| Tipo de Riesgo (cualitativo) | Moderado |

## 

| Riesgo R07 – Retrasos en repuestos o soporte técnico | |
| --- | --- |
| Estimación de Probabilidad | media (0.5) |
| Estimación de Impacto | medio (0.2) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.10 |
| Tipo de Riesgo (cualitativo) | Bajo |

## 

| Riesgo R08 – Inconsistencias entre datos locales y nube | |
| --- | --- |
| Estimación de Probabilidad | baja (0.3) |
| Estimación de Impacto | medio (0.2) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.06 |
| Tipo de Riesgo (cualitativo) | Bajo |

## 

| Riesgo R09 – Fallos de seguridad informática | |
| --- | --- |
| Estimación de Probabilidad | baja (0.3) |
| Estimación de Impacto | alto (0.4) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.12 |
| Tipo de Riesgo (cualitativo) | Bajo |

## 

| Riesgo R10 – Evento climático extremo | |
| --- | --- |
| Estimación de Probabilidad | baja (0.3) |
| Estimación de Impacto | muy alto (0.8) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.24 |
| Tipo de Riesgo (cualitativo) | Moderado |

## 

| Riesgo R11 – Desvinculación de un desarrollador clave | |
| --- | --- |
| Estimación de Probabilidad | media (0.5) |
| Estimación de Impacto | medio (0.2) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.10 |
| Tipo de Riesgo (cualitativo) | Bajo |

## 

| Riesgo R12 – Subestimación de tiempos de desarrollo | |
| --- | --- |
| Estimación de Probabilidad | media (0.5) |
| Estimación de Impacto | alto (0.4) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.20 |
| Tipo de Riesgo (cualitativo) | Moderado |

## 

| Riesgo R13 – Desalineación de expectativas con stakeholders | |
| --- | --- |
| Estimación de Probabilidad | media (0.5) |
| Estimación de Impacto | medio (0.2) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.10 |
| Tipo de Riesgo (cualitativo) | Bajo |

## 

| Riesgo R14 – Falta de visibilidad del avance del proyecto | |
| --- | --- |
| Estimación de Probabilidad | media (0.5) |
| Estimación de Impacto | medio (0.2) |
| Nivel de riesgo (Probabilidad x Impacto) | 0.10 |
| Tipo de Riesgo (cualitativo) | Bajo |

Matriz de Probabilidad x Impacto con los riesgos identificados

| Amenazas | | | | | |
| --- | --- | --- | --- | --- | --- |
| Impacto /Probabilidad | Muy bajo | Bajo | Medio | Alto | Muy alto |
| Muy alta | – | – | – | – | – |
| Alta | – | – | – | – | R01 |
| Mediana | – | – | R07, R11, R13, R14 | R02, R06, R12, R04 | R03 |
| Baja | – | – | R08 | R09 | R05, R10 |
| Muy baja | – | – | – | – | – |

## VIII.4 - Análisis de Bowtie

El análisis de riesgos adoptado en este proyecto se propone no solo identificar eventos potencialmente perjudiciales, sino también comprender sus causas, anticipar sus consecuencias y establecer mecanismos eficaces de control. Para tal fin, se empleó la técnica de **Bowtie Analysis**, reconocida por su capacidad de representar de manera visual y estructurada las relaciones entre amenazas, eventos críticos y consecuencias dentro de sistemas complejos.

### VIII.4.1 - Justificación de la metodología

La elección del enfoque Bowtie se fundamenta en su adecuación para proyectos donde coexisten múltiples factores técnicos, operativos y organizacionales que pueden derivar en fallos sistémicos. Este tipo de análisis permite:

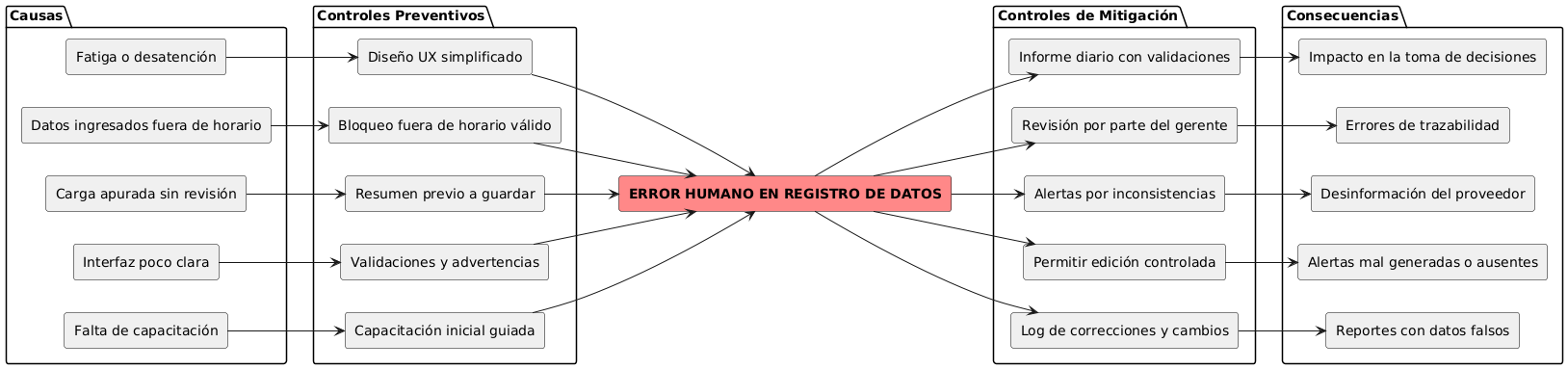
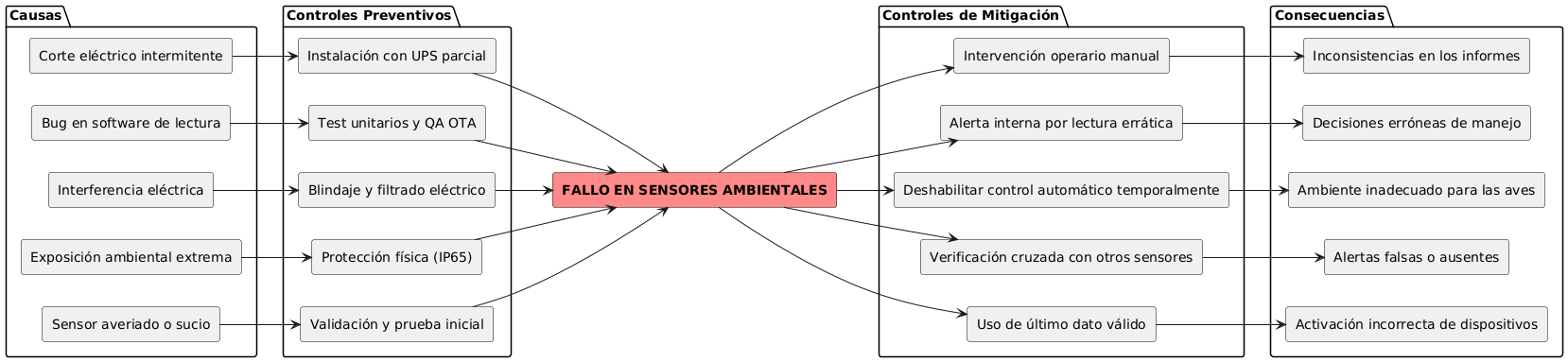
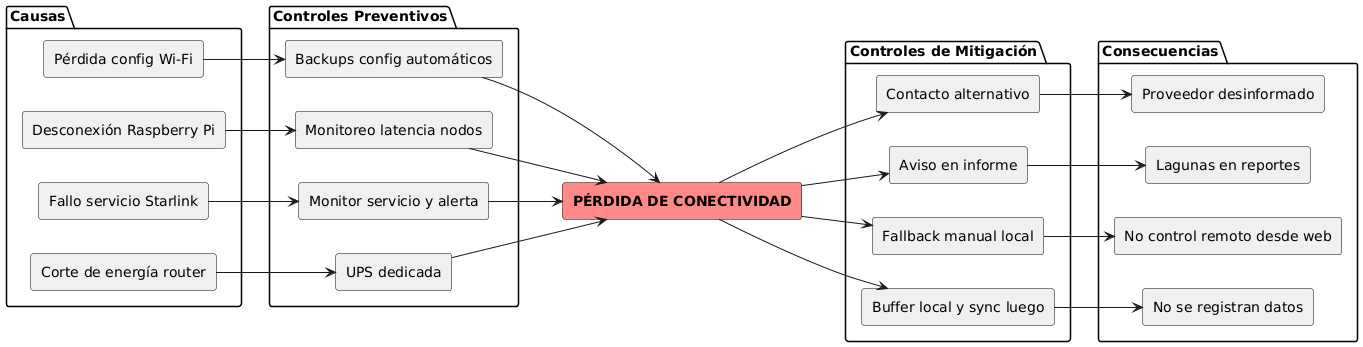
* Establecer una **conexión clara entre las causas raíz de un evento y sus posibles impactos**, facilitando la comprensión global del riesgo.
* Asociar a cada causa **controles preventivos específicos**, y a cada consecuencia **acciones de mitigación concretas**, promoviendo una gestión activa y anticipatoria.
* Integrar el análisis en entornos de documentación técnica, facilitando su actualización iterativa durante el ciclo de vida del proyecto.
* Servir como herramienta de comunicación efectiva entre actores técnicos y no técnicos, favoreciendo la toma de decisiones consensuada y fundamentada.

### VIII.4.1 - Aplicación al sistema

Dado que el sistema propuesto para la gestión avícola combina infraestructura tecnológica (sensores, dispositivos, enlaces de comunicación), procesos manuales (registro de datos por operarios) y un entorno ambiental altamente variable, ciertos riesgos fueron identificados como especialmente críticos por su impacto potencial o su probabilidad de ocurrencia.

Para estos casos, se construyeron diagramas Bowtie centrados en **eventos clave** cuya ocurrencia comprometería el funcionamiento esperado del sistema. Cada diagrama representa:

* En el **lado izquierdo**, las amenazas o causas del evento, junto con los **controles preventivos** establecidos para evitarlo.
* En el **lado derecho**, las consecuencias directas del evento si este se produce, junto con los **controles de mitigación** diseñados para minimizar su impacto.



## **VIII.5 -** Análisis Cuantitativo del Riesgo

El análisis cuantitativo del riesgo tiene como objetivo realizar una evaluación más detallada y numérica del impacto potencial de los riesgos identificados. Este tipo de análisis suele requerir herramientas estadísticas, simulaciones por Monte Carlo, modelos probabilísticos o análisis de sensibilidad, y se aplica principalmente en proyectos de gran escala, con alta complejidad y disponibilidad de datos históricos.

En el caso del presente proyecto, se optó por no realizar un análisis cuantitativo formal, y esta decisión se encuentra fundamentada en los siguientes criterios:

a) Relación costo-beneficio

La implementación de un análisis cuantitativo implica una inversión significativa de tiempo y recursos técnicos (modelado, estimaciones probabilísticas precisas, recopilación de datos históricos), los cuales no resultan proporcionales al alcance ni a la naturaleza del proyecto. Dado que se trata de una solución de gestión avícola desarrollada en un entorno controlado, con riesgos moderados y acotados, los recursos disponibles se consideraron mejor aplicados a otras áreas críticas del desarrollo.

b) Suficiencia del análisis cualitativo

El análisis cualitativo realizado permite identificar con claridad los eventos críticos, sus causas, consecuencias y niveles de severidad. A través de este enfoque se logró una clasificación efectiva de los riesgos en categorías de bajo, moderado y alto impacto, suficiente para priorizar acciones de respuesta y diseñar controles tanto preventivos como de mitigación. La matriz de probabilidad e impacto proporciona una herramienta visual clara y adecuada para la toma de decisiones.

c) Naturaleza del proyecto y grado de incertidumbre

El sistema propuesto no depende de variables aleatorias de alta volatilidad, ni requiere previsiones financieras o estratégicas donde el riesgo cuantificable determine inversiones millonarias. En cambio, se trata de un proyecto técnico-operativo con variables bien definidas, donde los principales riesgos pueden abordarse mediante controles de diseño, buenas prácticas de implementación y mecanismos de supervisión continua.

d) Recomendaciones de estándares internacionales

Tanto la Guía del PMBOK (7.ª edición) como el SEBoK establecen que el análisis cuantitativo de riesgos debe ser aplicado solo cuando aporte valor tangible al proceso de toma de decisiones, y no como requisito obligatorio. En línea con estas recomendaciones, y tras una evaluación técnica del contexto, se concluye que su aplicación no sería proporcional ni necesaria.

## VIII.6 - Planificación de Respuestas y Plan de Contingencia de los Riesgos

Una vez evaluada la severidad de cada riesgo mediante el análisis cualitativo, se procedió a definir estrategias de respuesta proporcionales a su nivel de criticidad. Esta planificación busca minimizar la probabilidad de ocurrencia o el impacto de los riesgos, y en caso de materialización, disminuir sus consecuencias negativas sobre el proyecto y su operación.

La respuesta ante riesgos se organiza según cuatro enfoques establecidos por el estándar PMBOK:

* Evitar: modificar aspectos del proyecto para eliminar completamente la amenaza.
* Mitigar: reducir la probabilidad de ocurrencia o el impacto del riesgo.
* Transferir: delegar parcial o totalmente el riesgo a un tercero (por ejemplo, mediante servicios externos).
* Aceptar: asumir el riesgo sin implementar medidas extraordinarias, con o sin plan de contingencia.

**Riesgos con planificación activa**

Los riesgos clasificados como altos o moderados-altos fueron priorizados en la planificación. A continuación, se describen las estrategias aplicadas a los casos más críticos:

| R01 – Pérdida de conectividad prolongada | |
| --- | --- |
| Trigger | El sistema detecta pérdida de conexión con el broker MQTT durante más de 10 minutos. |
| Respuesta planificada | Instalar UPS en router, validar enlaces de red y activar almacenamiento local en Raspberry Pi. |
| Tipo de respuesta | Mitigar |
| Plan de contingencia | Activar modo local en cada galpón, permitir operación manual de dispositivos y sincronizar datos diferidos una vez restablecida la conexión. |

| R02 – Fallo en sensores de temperatura o humedad | |
| --- | --- |
| Trigger | Sensor entrega el mismo valor durante más de 60 minutos o valores fuera de rango técnico posible. |
| Respuesta planificada | Validación cruzada con otros sensores, uso del último valor válido, lógica de verificación de consistencia. |
| Tipo de respuesta | Mitigar |
| Plan de contingencia | Emitir alerta de sensor fallado, permitir monitoreo manual por parte del operario y registrar manualmente las condiciones ambientales. |

| R03 – Corte de energía eléctrica en galpones | |
| --- | --- |
| Trigger | Lectura nula o desconexión simultánea de todos los sensores/dispositivos en un galpón. |
| Respuesta planificada | Segmentar y proteger tableros eléctricos, estudiar UPS críticos, instalar alertas de caída de voltaje. |
| Tipo de respuesta | Mitigar |
| Plan de contingencia | Enviar operario al sitio, verificar condiciones ambientales y accionar manualmente los dispositivos prioritarios (aspersores, ventiladores). |

| R04 – Error humano en carga de datos | |
| --- | --- |
| Trigger | Se ingresa un dato fuera de rango esperado (ej. peso > 10 kg o mortalidad negativa). |
| Respuesta planificada | Validaciones automáticas en la interfaz, advertencias previas y registro de cambios. |
| Tipo de respuesta | Mitigar |
| Plan de contingencia | Permitir corrección controlada dentro de una ventana de tiempo, informar al gerente, generar reporte de inconsistencias y mantener un log de auditoría. |

| R05 – Brote de enfermedad aviar | |
| --- | --- |
| Trigger | Mortalidad diaria superior al umbral configurado (por ejemplo, 0.5% del total de aves del galpón). |
| Respuesta planificada | Alertas automáticas al proveedor y gerente, revisión de visitas sanitarias y activación de protocolo de revisión. |
| Tipo de respuesta | Aceptar con contingencia |
| Plan de contingencia | Aislar el galpón, registrar manualmente síntomas o causas, enviar visita veterinaria urgente, suspender egresos y notificar al SENASA si aplica. |

| R10 – Evento climático extremo | |
| --- | --- |
| Trigger | Se recibe alerta meteorológica oficial o se observan daños físicos en antenas/dispositivos. |
| Respuesta planificada | Fortalecer instalaciones expuestas, monitorear previsiones del tiempo y resguardar antenas. |
| Tipo de respuesta | Mitigar |
| Plan de contingencia | Evaluar daños, reubicar temporalmente sensores/antenas, cambiar a modo manual de operación y registrar el evento en el sistema. |

## VIII.7 - Plan de Monitoreo y Control de Riesgos

El monitoreo y control de riesgos es una actividad continua que permite asegurar que las medidas definidas en la planificación sean efectivamente implementadas, detectar la aparición de nuevos riesgos y adaptar las estrategias existentes ante cambios en el contexto operativo o tecnológico del proyecto.

Dado que el sistema se implementa en un entorno dinámico —una granja avícola tecnificada— con múltiples variables externas (ambientales, técnicas y humanas), el proceso de control de riesgos fue diseñado para integrarse con las actividades rutinarias del proyecto, tanto durante la etapa de desarrollo como en la fase operativa.

### Objetivos del monitoreo

* Verificar la eficacia de los controles implementados.
* Detectar tempranamente desviaciones o eventos activadores de riesgo (triggers).
* Identificar la aparición de nuevos riesgos o la modificación de los existentes.
* Actualizar el registro de riesgos y adaptar las respuestas planificadas cuando sea necesario.
* Informar a las partes interesadas sobre la evolución del perfil de riesgo del sistema.

### Herramientas y mecanismos utilizados

Para llevar a cabo el monitoreo y control de riesgos, se aplican las siguientes herramientas y prácticas:

* **Revisiones técnicas periódicas:** Se realizan instancias de revisión del estado de sensores, dispositivos, alertas y registros, alineadas con los sprints de implementación del sistema o con el ciclo productivo avícola.
* **Indicadores operativos clave:** Se monitorean variables críticas que actúan como indicadores indirectos de riesgo (por ejemplo, ausencia de datos diarios, tasa de mortalidad atípica, lecturas ambientales congeladas o desconexiones de red prolongadas).
* **Auditorías internas de riesgo:** En puntos estratégicos del cronograma se efectúan auditorías de riesgos para validar si los controles planificados se ejecutan, si son suficientes, y si se requiere redefinir acciones preventivas o contingencias.
* **Actualización del registro de riesgos:** Cualquier evento nuevo o modificación de los riesgos existentes se documenta en el registro vivo del sistema, con sus respectivas valoraciones y responsables asignados.
* **Alertas automáticas del sistema:** El propio sistema genera alertas cuando ciertos parámetros preconfigurados exceden sus umbrales normales, funcionando como mecanismo temprano de detección de incidentes.
* **Reportes periódicos:** Al finalizar cada ciclo operativo (por ejemplo, al cierre de una camada), se genera un informe de revisión de riesgos que permite evaluar su evolución y efectividad del plan de respuesta.

### Participación y responsabilidades

* El **gerente del sistema** es el responsable de revisar periódicamente el estado de los riesgos, con apoyo del equipo técnico en cuestiones operativas.
* Los **operarios** participan reportando incidencias desde el sistema, especialmente aquellas relacionadas con sensores, conectividad o carga manual de datos.
* El **proveedor o veterinario** puede recibir alertas sanitarias o informes que también pueden dar lugar a ajustes en el registro de riesgos.

## **VIII.8 -** Conclusión

El análisis de riesgos realizado permitió identificar, clasificar y abordar de forma sistemática los eventos que podrían afectar negativamente la implementación y operación del sistema de gestión avícola. A partir de un enfoque basado en buenas prácticas del PMBOK y del SEBoK, se construyó un modelo de gestión de riesgos alineado a las características del proyecto: su base tecnológica, el entorno rural de operación, la interacción humano-máquina y la sensibilidad del proceso productivo.

Se definieron escalas de probabilidad e impacto, se aplicó una matriz de priorización para clasificar los riesgos según su nivel de criticidad, y se documentaron planes de respuesta proporcionales. Los riesgos más relevantes —como la pérdida de conectividad, los fallos eléctricos o los errores humanos en el ingreso de datos— fueron tratados mediante estrategias preventivas, controles de mitigación y planes de contingencia específicos.

La utilización de diagramas Bowtie permitió representar gráficamente la estructura causal y las consecuencias de los eventos críticos, reforzando la comprensión integral del sistema y guiando el diseño de mecanismos de control robustos.

Asimismo, se estableció un plan de monitoreo continuo de los riesgos, integrado a los procesos operativos del sistema, con participación activa del gerente, los operarios y otras partes interesadas. Este plan contempla auditorías internas, análisis de indicadores clave y actualización periódica del registro de riesgos, asegurando un enfoque dinámico y adaptativo.

En función del alcance y características del proyecto, se consideró técnicamente justificada la no realización de un análisis cuantitativo, dado que el análisis cualitativo realizado resultó suficiente para apoyar la toma de decisiones, asignar prioridades y diseñar medidas efectivas de gestión.

En conclusión, el proceso de análisis y gestión de riesgos desarrollado constituye una herramienta fundamental para garantizar la confiabilidad, continuidad y adaptabilidad del sistema, y contribuye de manera directa al éxito del proyecto en su conjunto.

# 

# Capítulo IX: Diseño

## IX.1 - Definición de Reglas de Negocio

En este apartado se describen las reglas de negocio que regirán el funcionamiento del sistema de gestión avícola. Estas reglas establecen criterios y comportamientos obligatorios que el sistema debe cumplir para adecuarse a las buenas prácticas de cría, mantener procesos internos coherentes y asegurar seguridad y trazabilidad de la información.

**Capacidad máxima de aves por galpón** Cada galpón posee un límite máximo de aves definido en función de su capacidad. El sistema no permitirá registrar una nueva camada si la suma de las aves de esa camada con las ya existentes en el galpón excede dicha capacidad. En caso de requerir más espacio, se deberá dividir la camada o alojarla en otro galpón.

**Datos obligatorios para registrar una nueva camada** Al crear una nueva camada, se exige obligatoriamente la fecha de ingreso, la cantidad de aves, el proveedor o procedencia, y el galpón asignado. Si alguno de estos campos falta, el sistema rechazará la operación hasta que se suministre toda la información requerida.

**Rango de temperatura y humedad por etapa de crecimiento** Para cada etapa de crecimiento de los pollos, existen umbrales recomendados de temperatura y humedad. El sistema permitirá configurar tales rangos, pero no aceptará valores ilógicos (por ejemplo, humedad mayor al 100%). Al sobrepasar los límites establecidos, se generará automáticamente una alerta para que el personal pueda tomar medidas correctivas.

**Mortalidad y registros diarios** El operario debe registrar a diario la mortalidad de aves de cada camada, especificando la causa si es conocida (enfermedad, calor, etc.). El sistema descontará la cantidad del total de la camada y actualizará la tasa acumulada de mortalidad. No se podrán ingresar muertes en camadas finalizadas o galpones inactivos.

**Pesaje de aves** El pesaje se realiza sobre una muestra representativa, consignando el peso promedio de las aves. El sistema almacenará cada registro con la fecha y el galpón, asociándolo a la camada. No se aceptarán valores fuera de rango, negativos o no numéricos. En caso de que se registre un peso inusual, el sistema solicitará confirmación para descartar un posible error de ingreso.

**Control de dispositivos climatizadores** Solo el operario y el gerente pueden activar, desactivar o ajustar manualmente los dispositivos (ventiladores, aspersores, cortinas). Cada acción se registra en una bitácora, junto con la fecha, hora y el estado final del dispositivo. Si existe un modo automático activo, las acciones manuales podrían anularse o requerir confirmación adicional cuando contravengan los setpoints configurados.

**Alerta ante variables fuera de rango** Cuando la temperatura o la humedad superan los umbrales permitidos, se emite automáticamente una alerta visible en el tablero principal y/o enviada al dispositivo móvil correspondiente. La alerta permanece abierta hasta que se realice la acción correctiva y el usuario la marque como resuelta. Cada alerta registra la fecha, la hora y la condición que la originó.

**Registro de auditoría (bitácora)** Todas las operaciones críticas (dar de alta una camada, manipular dispositivos, modificar configuraciones) se registran en una bitácora con fecha, hora y usuario. Únicamente el administrador puede acceder a la totalidad de la bitácora. No se permite la eliminación de registros; en caso de un error, se marcan como anulados, pero se conserva el rastro de la acción.

**Roles y permisos de usuarios** Se contemplan cuatro roles: operario, gerente, proveedor y administrador.

**Operario**: registra mortalidad, pesos, crea camadas y controla dispositivos.

**Gerente**: suma a las facultades del operario la posibilidad de configurar umbrales de alerta, generar informes y aprobar acciones críticas.

**Administrador**: maneja usuarios y roles, consulta la bitácora completa y administra la configuración global del sistema. No interviene directamente en las operaciones de la granja salvo que se le asigne permiso expreso.

**Proveedor:** puede consultar el histórico de camadas y reportes de las mismas.

**Cierre de camadas** Una camada se marca como finalizada cuando todas las aves han sido retiradas o la mortalidad es total. Una camada finalizada no admite más registros de mortalidad, peso ni traslados. Si se desea ingresar más aves, deberá crearse una nueva camada.

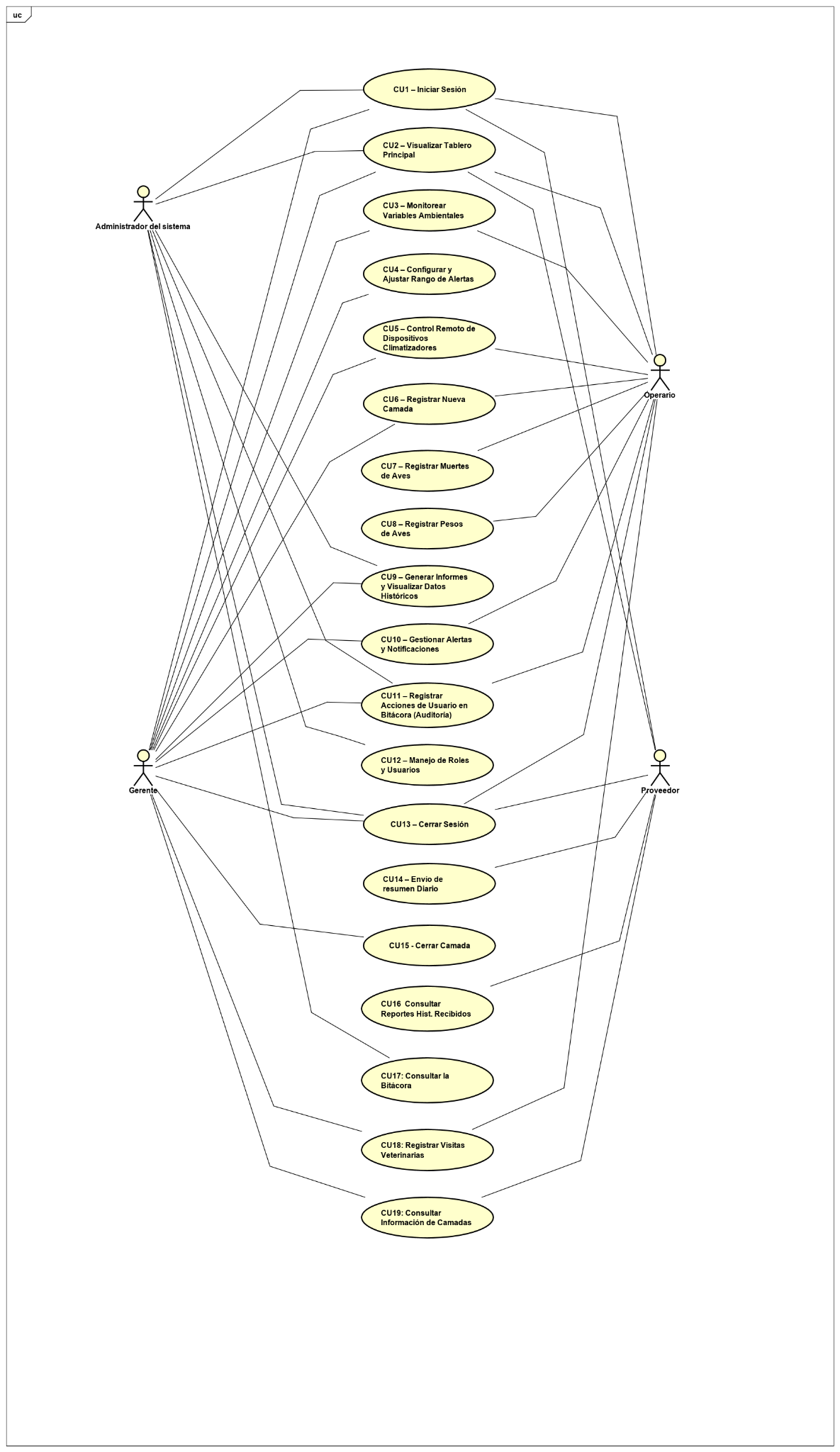
**Generación de informes y estadísticas** El gerente y el administrador pueden generar reportes de mortalidad, evolución de pesos y tendencias de temperatura/humedad en distintos rangos de fechas, camadas o galpones. Los informes estarán disponibles en formatos PDF y CSV para su impresión o posterior análisis externo.

**Integración con el veterinario y control sanitario (opcional)** De desearse, el sistema podrá registrar la visita del veterinario, sus diagnósticos o medicaciones en la ficha de cada camada, con acceso restringido al gerente y al operario responsable.

**Validación de datos en tiempo real** Al registrarse datos de mortalidad o de peso, se verifica automáticamente si existen alertas activas o inconsistencias (por ejemplo, un pico de mortalidad inesperado en un solo día). El sistema puede así disparar alertas internas para revisión.

**Periodo de inactividad y cierre automático de sesión** Para salvaguardar la seguridad, si un usuario permanece inactivo más de un tiempo predeterminado, la sesión se cierra automáticamente. Se deja constancia del cierre en la bitácora y el usuario tendrá que autenticarse de nuevo para continuar trabajando.

## IX.2 - Casos de uso



**CU1:** Iniciar Sesión

**CU2:** Visualizar Tablero Principal

**CU3:** Configurar Rangos y Canales de Alerta

**CU4:** Control Remoto de Dispositivos Climatizadores *(integrado en CU19)*

**CU5:** Registrar Nueva Camada

**CU6:** Registrar Muertes de Aves

**CU7:** Registrar Pesos de Aves

**CU8:** Generar Informes y Visualizar Datos Históricos

**CU9:** Gestionar Alertas Activas

**CU10:** Registrar Acciones de Usuario en Bitácora *(actor: Sistema)*

**CU11:** Manejo de Roles y Usuarios

**CU12:** Envío Automático de Resumen Diario al Proveedor *(actor: Sistema)*

**CU13:** Consultar Reportes Históricos Recibidos por el Proveedor *(actor: Proveedor, Gerente)*

**CU14:** Cerrar Camada al Finalizar la Cría

**CU15:** Consultar la Bitácora del Sistema *(actor: Administrador)*

**CU16:** Cerrar Sesión

**CU17:** Registrar Visitas Veterinarias y Observaciones Sanitarias

**CU18:** Consultar Información de Camadas

**CU19:** Visualizar Dashboard de Galpón

| **CU1 – Iniciar Sesión** |
| --- |
| **Objetivo:** |
| Permitir que el usuario acceda al sistema utilizando sus credenciales, de modo que pueda realizar las funciones correspondientes a su rol. |
| **Actores:** |
| Operario, Gerente, Administrador, Proveedor |
| **Pre-Condiciones** |
| * El usuario debe poseer un nombre de usuario y contraseña válidos, previamente registrados en el sistema. * Debe existir conexión con la base de datos para validar credenciales. |
| **Post-Condiciones:** |
| * El usuario queda autenticado. * El sistema registra en la bitácora (auditoría) la fecha/hora y el usuario que inició sesión. |
| **Disparador/Trigger** |
| El usuario accede a la pantalla de login o intenta realizar una acción restringida que requiere autenticación. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El usuario ingresa al portal del sistema y selecciona la opción “Iniciar Sesión”. 2. El sistema solicita al usuario el **nombre de usuario** y la **contraseña**. 3. El usuario ingresa sus credenciales y confirma. 4. El sistema verifica las credenciales contra la base de datos. 5. El sistema identifica el **rol** del usuario (Operario, Gerente, Administrador). 6. El sistema redirige al usuario segun su rol. |
| **Referencia Flujos alternativos** |
| 1. Si el rol fuese *Administrador* se lo redirige a la pantalla de administración del sistema. 2. Si el rol fuese *Proveedor/Gerente/Operario* se lo redirige al Dashboard Principal |
| **Referencia Flujos de Excepción o de Validación** |
| **Credenciales inválidas**   * El usuario ingresa un nombre de usuario o contraseña incorrectos. * El sistema muestra un mensaje de error e invita a reingresar credenciales.   **Usuario bloqueado o inactivo**   * El sistema detecta que la cuenta está bloqueada o inactiva. * El sistema notifica “Cuenta bloqueada, comuníquese con el administrador”. |

| **CU2 – Visualizar Tablero Principal** |
| --- |
| **Objetivo:** |
| Permitir al usuario visualizar en una única interfaz el estado general de los galpones, la camada activa, alertas recientes, variables ambientales resumidas y estado de los dispositivos. Permitir navegar a dashboards detallados por galpón (CU19), a la sección de visualización de camadas (CU18) y a la sección de generación de informes (CU08). |
| **Actores:** |
| Operario, Gerente, Proveedor(*sólo lectura)* |
| **Pre-Condiciones** |
| * El actor debe haber iniciado sesión (CU1 completado exitosamente). |
| **Post-Condiciones:** |
| * El usuario accede al tablero principal del sistema, donde obtiene una visión general consolidada con indicadores clave, tales como temperatura promedio, peso promedio de las aves y mortalidad acumulada. Desde este panel puede además navegar fácilmente hacia todas las demás secciones y funcionalidades de la aplicación. |
| **Disparador/Trigger** |
| El usuario, ya autenticado, selecciona la opción “Tablero Principal” o es redirigido automáticamente tras el login exitoso. |
| **Definición de Datos** |

**Flujo Principal**

| **Flujo Principal** |
| --- |
| 1. El usuario es redirigido a “Dashboard Principal” luego de pasar por el login (CU01). 2. El sistema consulta la información de los galpones (temperatura, humedad, estado de dispositivos, etc.). 3. El sistema muestra un resumen consolidado de:    1. Card de cada galpón (datos ambientales promedio, aves vivas y fallecidas)    2. Camada activa (cantidad total de aves, edad en días, mortalidad acumulada)    3. Alertas recientes    4. Condiciones meteorológicas externas actuales y pronóstico a corto plazo 4. El usuario puede:    1. Ir al dashboard de un galpón (CU19)    2. Ir a la sección de de visualización de camadas (CU18)    3. Ir a la sección de generación de informes (CU08) |
| **Referencia Flujos alternativos** |
| El usuario aplica algún filtro de galpón o de rango de datos, y el sistema refresca la vista. |
| **Referencia Flujos de Excepción o de Validación** |
| El sistema no puede obtener datos de uno o varios galpones.  Muestra valor “N/A” o mensaje “Error al recibir datos” y genera alerta interna.  Si existen galpones sin información de humedad/temperatura, se muestran campos en blanco o con indicador “No disponible”. |

| **CU3 – Configurar y Ajustar Rango de Alertas** |
| --- |
| **Objetivo:** |
| Proporcionar una vista general de los galpones en tiempo real: temperatura, humedad, estado de dispositivos, datos de camadas y alertas activas. |
| **Actores:** |
| Gerente, Administrador |
| **Pre-Condiciones** |
| * El usuario posee rol con permisos para modificar alertas (Gerente o Admin). |
| **Post-Condiciones:** |
| * El sistema actualiza los umbrales y usará estos valores para generar alertas (CU10). |
| **Disparador/Trigger** |
| El usuario entra a la sección de configuración de alertas. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El usuario selecciona “Configuración de Alertas”. 2. El sistema muestra la lista de galpones y los umbrales actuales. 3. El usuario ingresa nuevos valores (temp/humedad mín/máx), canales de notificación, etc. 4. El sistema valida que los valores sean lógicos. 5. El sistema guarda la nueva configuración en la base de datos. 6. El sistema informa “Configuración guardada con éxito”. |
| **Referencia Flujos alternativos** |
| El usuario repite la operación para múltiples galpones. |
| **Referencia Flujos de Excepción o de Validación** |
| Valores fuera de rango: El usuario ingresa valores no válidos. Por ejemplo, alertar de una humedad relativa de 200% |

| **CU4 – Control Remoto de Dispositivos Climatizadores** |
| --- |
| **Objetivo:** |
| Permitir la activación, desactivación o ajuste manual de ventiladores, aspersores, luces y cortinas en cada galpón. |
| **Actores:** |
| Gerente, Operario |
| **Pre-Condiciones** |
| * El usuario está autenticado y tiene los permisos adecuados. * El hardware y la conexión IoT están funcionando correctamente. |
| **Post-Condiciones:** |
| * El dispositivo queda en el estado elegido. * Se registra la acción en la bitácora (usuario, fecha, dispositivo, orden). |
| **Disparador/Trigger** |
| El usuario elige enciende o apaga un dispositivo de un galpón o sección (Cortinas, ventiladores, luces o aspersores) |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El usuario accede a “Control de Dispositivos”. 2. Selecciona el galpón y el dispositivo (ventilador/aspersor/cortina). 3. Elige la acción a realizar (encender, apagar, regular). 4. El sistema envía la orden al controlador. 5. El sistema muestra el estado final y confirma “Dispositivo activado / desactivado / ajustado”. |
| **Referencia Flujos alternativos** |
| - |
| **Referencia Flujos de Excepción o de Validación** |
| **Fallo de comunicación**   * El sistema no recibe confirmación del dispositivo. * Muestra un aviso de error “No se pudo completar la orden” e invita a reintentar.   **Permisos insuficientes**   * El usuario no tiene derechos para manipular el dispositivo (por política interna). * El sistema no ejecuta la acción y muestra un mensaje de “Acceso Denegado”. |

| **CU5 – Registrar Nueva Camada** |
| --- |
| **Objetivo:** |
| Dar de alta una nueva camada de pollos, indicando los datos de ingreso, proveedor y cantidad destinada a cada galpón. |
| **Actores:** |
| Gerente, Operario |
| **Pre-Condiciones** |
| * El usuario tiene una **sesión activa** con rol que permita registrar camadas. * No hay una camada activa, es decir, se registró el egreso de la camada anterior. |
| **Post-Condiciones:** |
| * La camada queda registrada, y se crean las Camadas Galpón. * Se habilitan las opciones de registrar pesos y fallecidos por galpón. |
| **Disparador/Trigger** |
| El usuario desea dar de alta una nueva camada de pollos ante un nuevo envío del proveedor. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El usuario elige la opción “Registrar Camada”. 2. El sistema solicita datos: fecha de ingreso, cantidad de aves totales, proveedor. 3. El sistema solicita datos por cada galpón: cantidad de aves. 4. El usuario ingresa la información y confirma. 5. El sistema valida las capacidades máximas de los galpones según las cantidades de aves destinadas a cada uno. 6. El sistema registra la camada y asigna un ID único. 7. El sistema muestra un mensaje de confirmación: “Camada creada exitosamente”. |
| **Referencia Flujos alternativos** |
| - |
| **Referencia Flujos de Excepción o de Validación** |
| **Capacidad excedida**   * El usuario ingresa una cantidad mayor a la capacidad del galpón. * El sistema alerta “Capacidad máxima superada” y pide corregir datos o usar otro galpón.   **Datos incompletos**   * El usuario deja campos obligatorios vacíos. * El sistema muestra error y solicita completar toda la información requerida. |

| **CU6 – Registrar Muertes de Aves** |
| --- |
| **Objetivo:** |
| Mantener actualizado el recuento de aves vivas/fallecidas en cada camada, registrando la causa, la fecha y el galpón correspondiente. |
| **Actores:** |
| Operario, gerente |
| **Pre-Condiciones** |
| * El usuario está autenticado y tiene permisos para registrar. * Hay una camada activa actualmente. |
| **Post-Condiciones:** |
| * Se ajusta el conteo de aves para la camada. * El sistema almacena el evento en el historial de mortalidad. |
| **Disparador/Trigger** |
| El Operario realiza el recuento diario o detecta una mortalidad y decide ingresarla en el sistema. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El usuario selecciona “Registrar Muertes”. 2. El sistema solicita: fecha, número de aves, galpón, causa de fallecimiento. 3. El usuario ingresa los datos y confirma. 4. El sistema descuenta la cantidad de aves muertas del total de la camada. 5. El sistema muestra un mensaje de confirmación. |
| **Referencia Flujos alternativos** |
| - |
| **Referencia Flujos de Excepción o de Validación** |
| **Número negativo o no válido**   * El usuario ingresa un valor negativo o no numérico. * El sistema muestra “Valor inválido” y no permite confirmar.   **Camada no existe / cerrada**   * El sistema alerta “Camada finalizada, no se pueden registrar más datos”. |

| **CU8 – Registrar Pesos de Aves** |
| --- |
| **Objetivo:** |
| Registrar el peso promedio de una muestra de aves de un determinado galpón para monitorear su desarrollo. |
| **Actores:** |
| Operario |
| **Pre-Condiciones** |
| * Sesión iniciada con rol de Operario (o superior). * Camada activa * Cantidad de aves en el galpón mayor a cero. |
| **Post-Condiciones:** |
| * El sistema registra el nuevo peso en el historial de la camada. |
| **Disparador/Trigger** |
| El Operario realiza un pesaje de un muestreo de aves y desea registrar el peso promedio. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El usuario selecciona “Registrar Pesos”. 2. El sistema solicita galpón, fecha, peso promedio y cantidad de aves tomadas para la muestra. 3. El usuario ingresa los datos. 4. El sistema valida que el peso sea un valor positivo en el rango esperado. 5. El sistema guarda el registro y actualiza los datos estadísticos. |
| **Referencia Flujos alternativos** |
| * El usuario ingresa un peso menor al de la última medición de la misma camada * El sistema da aviso de esto y pregunta si desea continuar * El usuario puede continuar por flujo principal (5) o cancelar el registro. |
| **Referencia Flujos de Excepción o de Validación** |
| **Peso fuera de rango**   * El usuario ingresa un peso exagerado, negativo o cero. * El sistema no permite ingreso del dato.   **Camada inexistente**   * Idem caso del CU7. |

| **CU8 – Generar Informes y Visualizar Datos Históricos** |
| --- |
| **Objetivo:** |
| Permitir la consulta de información histórica de temperatura, humedad, mortalidad y pesos de aves para identificar tendencias. |
| **Actores:** |
| Gerente, Administrador |
| **Pre-Condiciones** |
| * El usuario tiene privilegios de **Gerente** o superior. * Existen registros de datos históricos almacenados. |
| **Post-Condiciones:** |
| * El usuario dispone del **reporte histórico** para análisis o toma de decisiones. |
| **Disparador/Trigger** |
| El usuario (Gerente/Admin) decide consultar estadísticas pasadas (temperatura, mortalidad, peso, etc.) o generar un reporte. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El usuario elige la opción “Informes / Datos Históricos”. 2. Selecciona tipo de reporte: (a) Temperatura/Humedad, (b) Mortalidad, (c) Pesos. 3. Define filtros (rango de fechas, galpón, camada). 4. El sistema procesa la información y genera un informe con **gráficas y estadísticas**. 5. El usuario visualiza el informe y puede exportarlo (PDF, CSV). |
| **Referencia Flujos alternativos** |
| - |
| **Referencia Flujos de Excepción o de Validación** |
| **No hay datos en el rango solicitado**   * El sistema muestra “No se encontraron datos para esos filtros”. * El usuario puede cambiar criterios y reintentar. |

| **CU9 – Gestionar Alertas y Notificaciones** |
| --- |
| **Objetivo:** |
| Revisar alertas (ej. temperatura fuera de rango) y tomar acción correctiva, dejando un registro de resolución o comentarios. |
| **Actores:** |
| Gerente, Operario |
| **Pre-Condiciones** |
| * El sistema previamente generó alertas (por CU4). * El usuario tiene sesión iniciada. |
| **Post-Condiciones:** |
| * Cada alerta queda en un estado final (Resuelta, Falsa Alarma, Pendiente). * El historial de alertas reflejará las acciones y comentarios. |
| **Disparador/Trigger** |
| El usuario accede a la sección de alertas para ver y gestionar las que estén activas (disparadas por fuera de rango ambiental, fallas, etc.). |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El usuario va a “Alertas Activas”. 2. El sistema lista las alertas con la descripción (p.ej. “Galpón 2 supera 30°C”). 3. El usuario selecciona una alerta y ejecuta acción (p.ej. encender ventilador). 4. El sistema registra la acción y marca la alerta como “En proceso” o “Resuelta”. 5. El usuario puede añadir comentarios. 6. La alerta pasa a histórico al considerarse “Resuelta”. |
| **Referencia Flujos alternativos** |
| - |
| **Referencia Flujos de Excepción o de Validación** |
| **No se puede resolver**   * El usuario no tiene permisos para accionar dispositivos. * El sistema le indica que derive la alerta al Gerente. |

| **CU10 – Registrar Acciones de Usuario en Bitácora** |
| --- |
| **Objetivo:** |
| Mantener un **historial** de las acciones críticas que realizan los usuarios para fines de auditoría y seguridad. |
| **Actores:** |
| Sistema |
| **Pre-Condiciones** |
| * El usuario realiza una acción “importante” (ej. ajustes de dispositivos, altas de camada, etc.). |
| **Post-Condiciones:** |
| * El evento queda registrado en la bitácora. |
| **Disparador/Trigger** |
| Cualquier acción “importante” realizada por un usuario o por el sistema (envío de correo) activa el registro en bitácora. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. Cuando un usuario realiza una acción relevante, el sistema registra:    1. Fecha y hora.    2. Usuario y rol.    3. Acción ejecutada (e.g. “Encender Aspersor en Galpón 2”). 2. Estos datos se guardan automáticamente en la base de datos. 3. El Administrador puede acceder a “Ver Bitácora” y filtrar por fecha, usuario, acción. |
| **Referencia Flujos alternativos** |
| - |
| **Referencia Flujos de Excepción o de Validación** |
| **Fallo en la base de auditoría**   * Si no se puede registrar la acción, el sistema genera una alerta de error y lo notifica al Administrador. |

| **CU11 – Manejo de Roles y Usuarios** |
| --- |
| **Objetivo:** |
| Crear, modificar o eliminar usuarios y asignar roles (Operario, Gerente, Administrador). |
| **Actores:** |
| Administrador |
| **Pre-Condiciones** |
| * El Administrador tiene una sesión iniciada y privilegios para la gestión de usuarios. * Existe un listado de roles definidos. |
| **Post-Condiciones:** |
| * El usuario queda creado, modificado o eliminado en la base de datos. * Cualquier cambio de rol se refleja en los permisos del usuario. |
| **Disparador/Trigger** |
| El Administrador necesita crear, editar o eliminar cuentas de usuarios, o asignar roles (Operario, Gerente, Admin). |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El Administrador ingresa a “Administración de Usuarios”. 2. Selecciona una acción: *Crear usuario*, *Editar usuario* o *Eliminar usuario*. 3. En caso de creación, el sistema solicita datos (nombre, usuario, contraseña, rol). 4. El Administrador confirma; el sistema guarda o actualiza registros. 5. El sistema muestra un mensaje de confirmación y notifica al usuario creado (si procede). |
| **Referencia Flujos alternativos** |
| - |
| **Referencia Flujos de Excepción o de Validación** |
| **Usuario ya existe**   * El sistema detecta duplicación de nombre de usuario. * Muestra el error “Nombre de usuario en uso”.   **Eliminación restringida**   * No se puede eliminar la cuenta de un usuario que esté realizando una tarea crítica. |

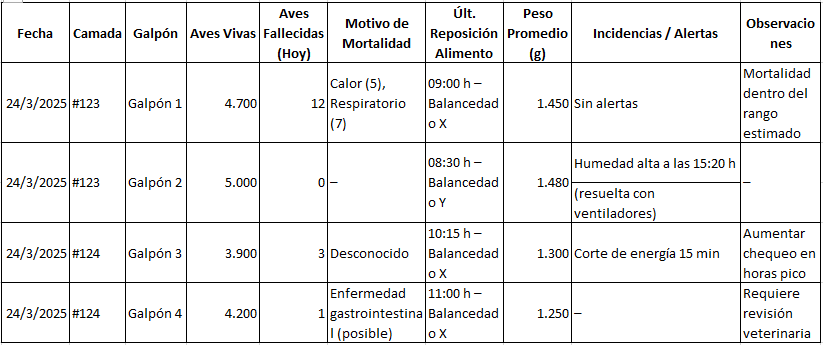
| **CU12 – Cerrar Sesión (Logout)** |
| --- |
| **Objetivo:** |
| Permitir que el usuario finalice su sesión de forma segura en el sistema. |
| **Actores:** |
| Operario, Gerente, Administrador, Proveedor |
| **Pre-Condiciones** |
| * El usuario se encuentra **logueado** en el sistema. |
| **Post-Condiciones:** |
| * El usuario queda desautenticado. * El sistema registra en bitácora la hora de cierre de sesión. |
| **Disparador/Trigger** |
| El usuario hace clic en “Cerrar Sesión” o se excede el tiempo de inactividad configurado. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El usuario selecciona la opción “Cerrar Sesión”. 2. El sistema confirma si desea salir. 3. El usuario confirma. 4. El sistema **invalida la sesión** y redirige a la pantalla de inicio de sesión. |
| **Referencia Flujos alternativos** |
| **Tiempo de inactividad**   * Si el usuario está inactivo por más de X minutos (ej. 15 min), el sistema cierra sesión automáticamente. * Muestra mensaje “Su sesión ha expirado”. |
| **Referencia Flujos de Excepción o de Validación** |
| **-** |

| **CU13 – Envío de Resumen Diario a Proveedor** |
| --- |
| **Objetivo:** |
| El objetivo de este caso de uso es que el operador genere en el sistema, al finalizar cada jornada, un **informe diario por camada y galpón** con datos clave del proceso de crianza, y lo **envíe** al proveedor para su seguimiento y control.  Este informe actúa como un resumen operativo del estado actual de la camada, útil para coordinar futuras extracciones y monitorear la evolución. |
| **Actores:** |
| Operario |
| **Pre-Condiciones** |
| * Debe existir al menos una camada activa registrada. * La camada debe tener **galpones asociados**. * Debe existir al menos un usuario con rol **Proveedor** y email configurado. |
| **Post-Condiciones:** |
| * Se ha generado y enviado un informe por camada activa. * El sistema tiene un log de cada envío realizado (o intento fallido). |
| **Disparador/Trigger** |
| Al final de la jornada el Operario decide manualmente enviar el reporte. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. Todos los días a una hora programada, el operador ingresa al panel de gestión de bitácora.. 2. El operador ingresa los siguientes datos del día actual:    1. Cantidad de aves fallecidas por galpón (y motivo si se ingresó).    2. Fecha y tipo de última reposición de alimento por galpón.    3. Último peso promedio registrado (si hay datos).    4. Incidencias reportadas durante el día (si existen). 3. El sistema genera un informe tipo resumen (ej. PDF o XLS) por camada. 4. El sistema envía el informe por correo electrónico a los usuarios con rol **Proveedor**. 5. El sistema registra en la bitácora (logSistema) que el envío fue realizado. |
| **Referencia Flujos alternativos** |
| **No hay datos del día**   * Si no se registró ningún dato (mortalidad, alimento, peso) para una camada ese día:   + El sistema aún puede generar el informe, pero indicará “Sin datos registrados hoy”. |
| **Referencia Flujos de Excepción o de Validación** |
| **Error en el envío (fallo de red o configuración)**   * El sistema detecta que no se pudo enviar el correo. * Se registra el error en la bitácora con mensaje detallado. * Se puede configurar un reintento automático o notificación al administrador. |

Ejemplo del informe enviado al proveedor



| **CU14 – Cerrar Camada al Finalizar la Cría** |
| --- |
| **Objetivo:** |
| Permitir al Gerente marcar una camada como finalizada cuando las aves han sido retiradas o la mortalidad es total, de modo que ya no se registren más eventos (muertes, pesos, etc.) y se consoliden los datos de esa crianza. |
| **Actores:** |
| Gerente |
| **Pre-Condiciones** |
| * El usuario (Gerente) debe haber iniciado sesión. * Debe existir al menos una camada activa. * La camada tiene fecha de inicio registrada (CU6) y registros de mortalidad y peso al día. |
| **Post-Condiciones:** |
| * La camada queda en estado “Finalizada” y no se permiten más registros de peso o mortalidad para la misma. * Sus datos sí pueden consultarse para informes (CU9, CU19) o históricos. |
| **Disparador/Trigger** |
| El Gerente considera que la camada ya finalizó (aves retiradas o mortalidad total) y decide cerrarla. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El Gerente selecciona la opción “Cerrar Camada”. 2. El sistema muestra la lista de camadas activas. 3. El Gerente elige la camada a cerrar. 4. El sistema solicita confirmación (p.ej. “¿Está seguro de cerrar la camada X?”). 5. El Gerente confirma. 6. El sistema cambia el estado de la camada a “Cerrada” o “Finalizada” y ya no permite su modificación. 7. El sistema registra el evento en la bitácora (fecha, hora, usuario, camada). 8. El sistema notifica “Camada cerrada exitosamente”. |
| **Referencia Flujos alternativos** |
| **-** |
| **Referencia Flujos de Excepción o de Validación** |
| **Camada no apta para cerrar**   * El sistema detecta que la camada ya está cerrada o no existe. * Muestra un mensaje “Esta camada no está activa o ya fue cerrada”. |

| **CU15 – Consultar Reportes Históricos Recibidos por el Proveedor** |
| --- |
| **Objetivo:** |
| Permitir al Proveedor ver los reportes antiguos (p.ej. informes diarios enviados anteriormente), de modo que pueda revisar datos de días pasados, compararlos y asegurar la trazabilidad de la información. |
| **Actores:** |
| Proveedor |
| **Pre-Condiciones** |
| * El Proveedor tiene credenciales y rol apropiado para acceder a la plataforma. * Existen reportes previos enviados (por CU14 u otras vías). * El proveedor ha iniciado sesión (CU1). |
| **Post-Condiciones:** |
| * El Proveedor puede ver o descargar los reportes históricos que se han generado. * Se registra la acción en la bitácora para auditoría. |
| **Disparador/Trigger** |
| El Proveedor desea ver reportes de días pasados que el sistema (o Operario) le ha enviado. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El Proveedor se autentica y selecciona la opción “Reportes Históricos”. 2. El sistema lista la cronología de reportes con fecha, tipo de informe, etc. 3. El Proveedor selecciona un reporte. 4. El sistema muestra el contenido (PDF, HTML, etc.) o un enlace para descargarlo. 5. Opcionalmente, el Proveedor puede filtrar reportes por rango de fechas, tipo de informe o estado. 6. El sistema registra en bitácora que el Proveedor accedió a “N” reporte(s), con fecha y hora. |
| **Referencia Flujos alternativos** |
| **-** |
| **Referencia Flujos de Excepción o de Validación** |
| **No hay reportes disponibles**   * El sistema muestra “No se encontraron reportes” si la búsqueda no retorna resultados. |

| **CU16 – Consultar la Bitácora del Sistema** |
| --- |
| **Objetivo:** |
| El Administrador puede revisar (auditar) los registros de acciones y eventos que fueron registrados (por CU11) en la bitácora: acciones de usuarios (creación de camadas, control de dispositivos, configuración de alertas) o eventos técnicos (errores, notificaciones del sistema). |
| **Actores:** |
| Administrador |
| **Pre-Condiciones** |
| * El Administrador ha iniciado sesión. * Existen eventos en la bitácora (acciones de usuarios, fallos, alertas generadas, etc.). |
| **Post-Condiciones:** |
| * El Administrador ha obtenido la información histórica de la bitácora. * Queda constancia de la consulta en la propia bitácora (opcional). |
| **Disparador/Trigger** |
| El Administrador requiere auditar eventos o acciones pasadas registrados en la bitácora. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El Administrador accede a “Ver Bitácora”. 2. El sistema muestra un formulario de filtrado opcional (fecha inicial, fecha final, tipo de acción, usuario). 3. El Administrador ingresa los filtros deseados. 4. El sistema consulta la base de datos de auditoría y retorna los registros que cumplan los criterios. 5. El Administrador visualiza la lista de eventos, con detalles (fecha, usuario, acción, resultado, etc.). 6. Opcionalmente, el Administrador puede exportar esa lista (CSV, PDF) o ver más detalle de un evento específico. |
| **Referencia Flujos alternativos** |
| **-** |
| **Referencia Flujos de Excepción o de Validación** |
| **Sin resultados**   * El sistema muestra “No hay registros para ese filtro”. |

| **CU17 – Registrar Visitas Veterinarias y Observaciones Sanitarias** |
| --- |
| **Objetivo:** |
| Permitir el registro de visitas de un veterinario a la granja, anotando diagnósticos, medicaciones y recomendaciones sanitarias, relacionándolos con las camadas afectadas. |
| **Actores:** |
| Operario, Gerente |
| **Pre-Condiciones** |
| * El Operario o Gerente está autenticado en el sistema. * Existen camadas activas que pueden necesitar la intervención del veterinario. |
| **Post-Condiciones:** |
| * Queda registrada la visita del veterinario, junto con las observaciones sanitarias en esa camada/galpón. * Esta información se podrá usar para informes sanitarios o futuros seguimientos. |
| **Disparador/Trigger** |
| Hay una visita del veterinario, o se realizan observaciones sanitarias, y se desea documentarlo en el sistema. |
| **Definición de Datos** |

| **Flujo Principal** |
| --- |
| 1. El usuario (Operario/Gerente) accede a “Visitas Veterinarias”. 2. El sistema solicita: fecha de visita, nombre o ID del veterinario (opcional), galpón o camada, diagnóstico, medicación, recomendaciones. 3. El usuario ingresa los datos. 4. El sistema valida la camada y galpón. 5. El sistema guarda el registro sanitario en la base de datos. 6. El sistema confirma “Visita registrada correctamente” y registra la acción en la bitácora. |
| **Referencia Flujos alternativos** |
| **-** |
| **Referencia Flujos de Excepción o de Validación** |
| **Camada inexistente o cerrada**   * El sistema no permite registrar una observación en una camada finalizada. * Muestra mensaje “La camada X está cerrada; no se pueden anexar más datos de salud”. |

| **CU18 – Consultar Información de Camadas Activas o Finalizadas** |
| --- |
| **Objetivo:** |
| Permitir a Gerente (y opcionalmente al Proveedor, con permisos restringidos) consultar un listado de camadas, ya sean activas o finalizadas, con detalles como fechas, número de aves inicial, total de aves muertas, fecha de cierre, etc. |
| **Actores:** |
| Proveedor, Gerente |
| **Pre-Condiciones** |
| * El actor (Gerente / Proveedor) está autenticado. * Existen camadas registradas (al menos una). |
| **Post-Condiciones:** |
| * El usuario ve la información resumida o detallada de las camadas, su estado y datos relevantes. * Se registra la consulta en la bitácora (opcional). |
| **Disparador/Trigger** |
| El Gerente o el Proveedor quieren revisar el estado o histórico de una o varias camadas, sea que estén activas o ya finalizadas (CU15). |
| **Definición de Datos** |

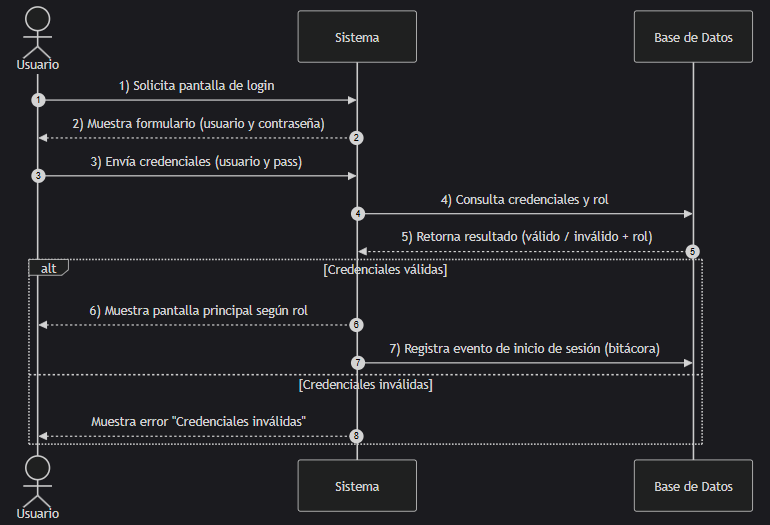
| **Flujo Principal** |
| --- |
| 1. El usuario (Gerente/Proveedor) ingresa a “Consultar Camadas”. 2. El sistema muestra filtros (estado: activo/finalizado, rango de fechas, etc.). 3. El usuario aplica filtros o ve todas las camadas. 4. El sistema recupera los datos de la base: ID de camada, fecha de inicio, cantidad de aves inicial, total muertes, si está cerrada o no, fecha de cierre, etc. 5. El sistema presenta la lista, permitiendo al usuario ver detalles más profundos (p.ej. pesajes, mortalidad diaria, etc.). 6. Opcionalmente, puede exportar la información o generar un informe. |
| **Referencia Flujos alternativos** |
| **-** |
| **Referencia Flujos de Excepción o de Validación** |
| **No hay camadas en el estado solicitado**   * El sistema notifica: “No se encontraron camadas con esos criterios”. |

| **CU19 – Visualizar Dashboard de Galpón** |
| --- |
| **Objetivo:** |
| Permitir al usuario acceder al panel específico de un galpón para consultar en tiempo real las variables ambientales, el estado de los dispositivos, visualizar gráficas históricas y realizar |
| **Actores:** |
| Operario, Gerente |
| **Pre-Condiciones** |
| * El usuario debe haber iniciado sesión (CU1) con rol de Operario o Gerente. * El sistema debe tener al menos un galpón registrado y sensores/dispositivos vinculados. * Deben existir datos disponibles provenientes de sensores IoT. |
| **Post-Condiciones:** |
| * El usuario ha visualizado el estado detallado del galpón. * Si el rol lo permite, puede haber accionado dispositivos o consultado históricos. * Todas las acciones se registran en la bitácora del sistema (CU11). |
| **Disparador/Trigger** |
| El Gerente o el Operario quieren visualizar información sobre un galpón en especifico. |
| **Definición de Datos** |

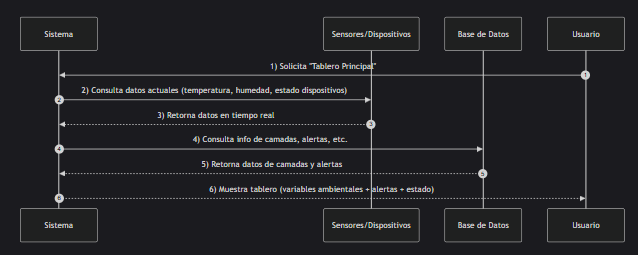
| **Flujo Principal** |
| --- |
| 1. El usuario accede al DashboardPrincipal (CU2). 2. Desde allí, selecciona un galpón específico. 3. El sistema despliega una vista detallada del galpón seleccionado. 4. Se muestran las variables ambientales actuales: temperatura, humedad, y otras disponibles. 5. Se muestra el estado actual de los dispositivos: ventiladores, aspersores, luces, cortinas. 6. El usuario puede consultar el historial de variables (últimas 24h, semana, etc.) en formato gráfico. 7. El sistema permite, según el rol, ejecutar acciones sobre dispositivos (CU5) o consultar solo en modo lectura. 8. Si hay alertas activas en el galpón, se listan y permiten navegar a su gestión (CU10). |
| **Referencia Flujos alternativos** |
| **-** |
| **Referencia Flujos de Excepción o de Validación** |
| **Galpón sin datos registrados**   * El sistema muestra un mensaje: “No hay datos disponibles aún para este galpón.”   **Error en la comunicación con sensores o dispositivos**   * Se muestran los campos como “N/D” y se emite una advertencia no bloqueante. * Se registra en la bitácora (CU11) el fallo de lectura correspondiente.   **Usuario sin permisos de acción**   * El sistema desactiva los botones de control y solo permite la visualización. |

## IX.3 - Diagramas de secuencia

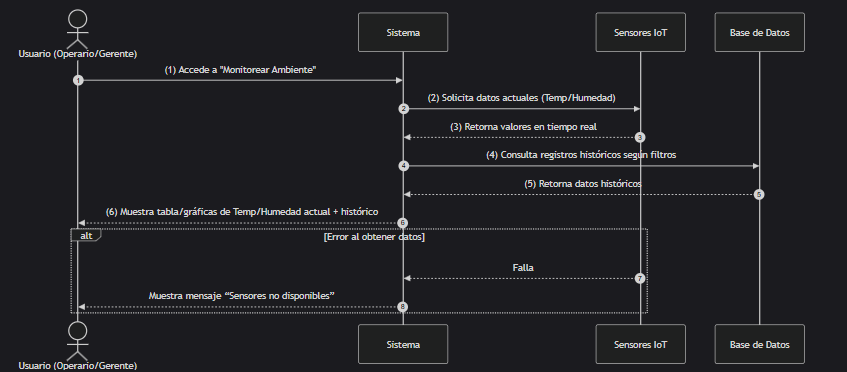
CU1 – Iniciar Sesión



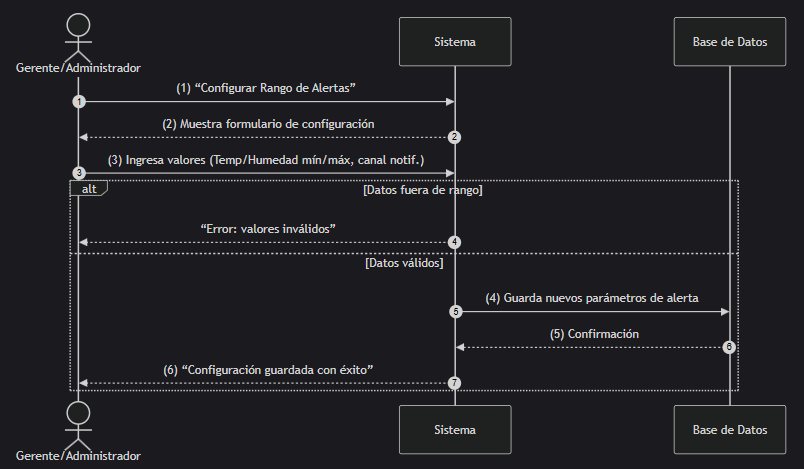
CU2 – Visualizar Dashboard Principal



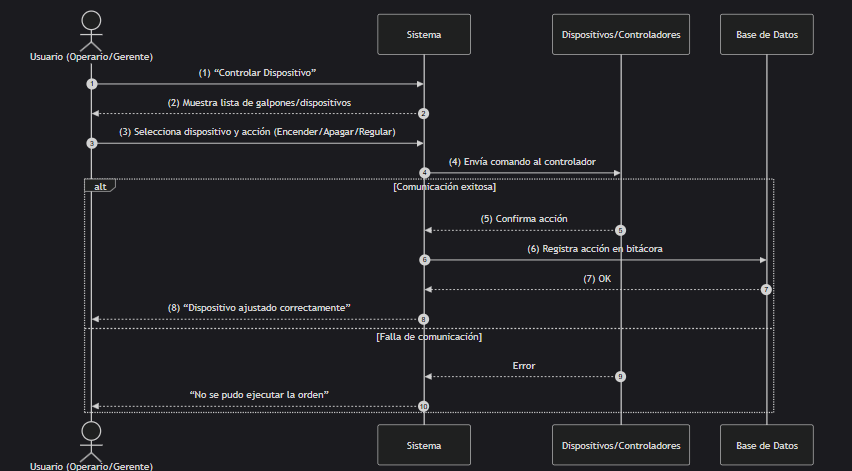
CU3 – Monitorear Variables Ambientales



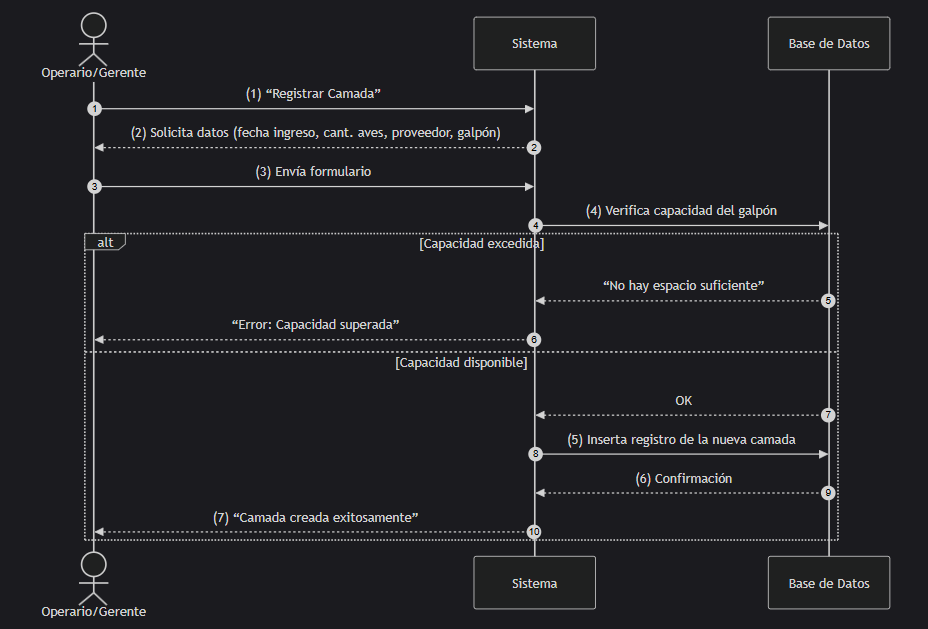
CU4 – Configurar y Ajustar Rango de Alertas



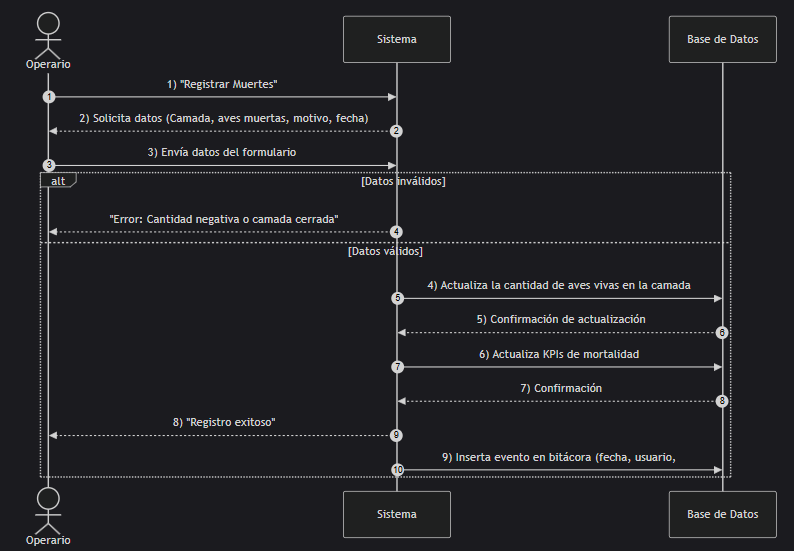
CU5 – Control Remoto de Dispositivos Climatizadores



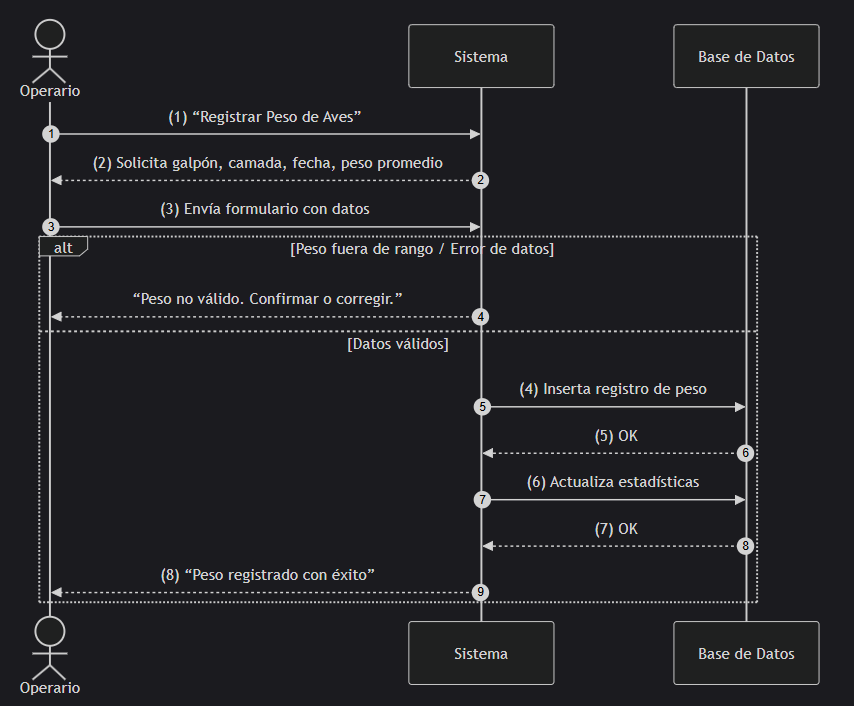
CU6 – Registrar Nueva Camada



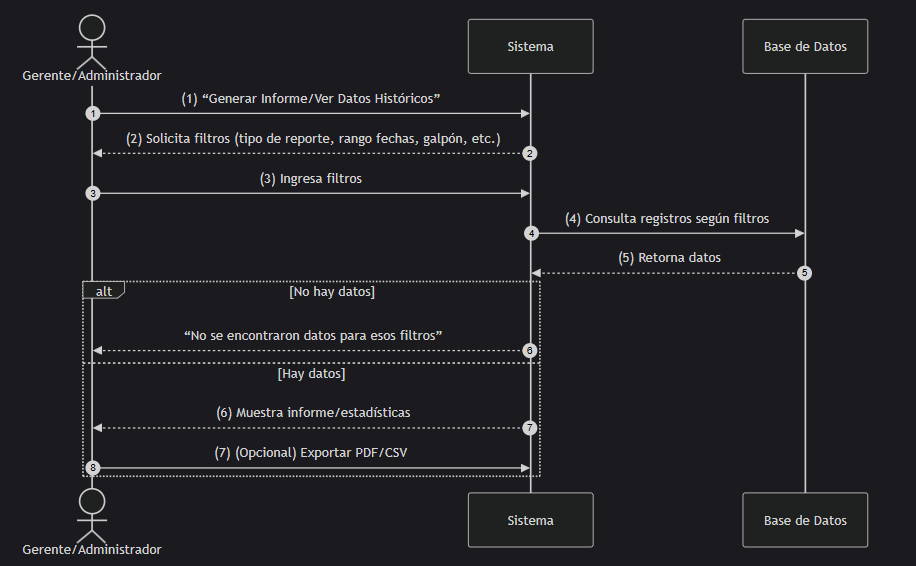
CU7 – Registrar Muertes de Aves



CU8 – Registrar Pesos de Aves

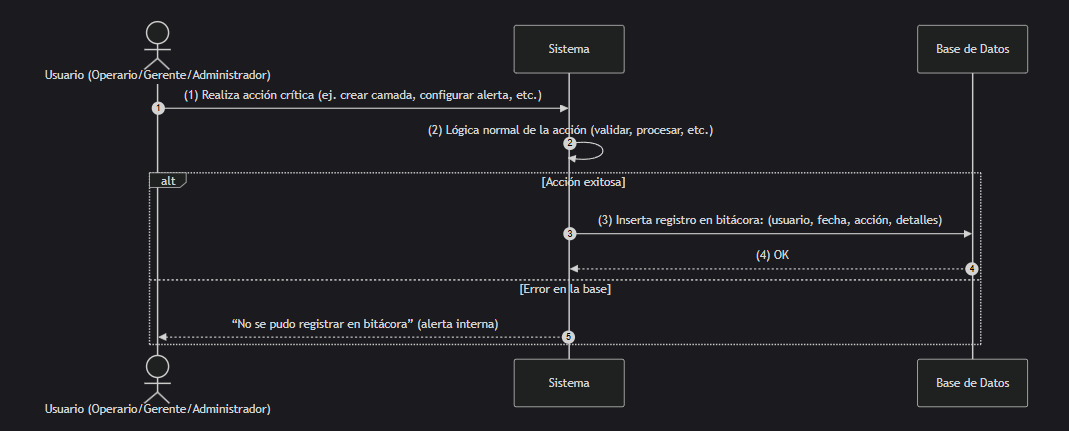


CU9 – Generar Informes y Visualizar Datos Históricos

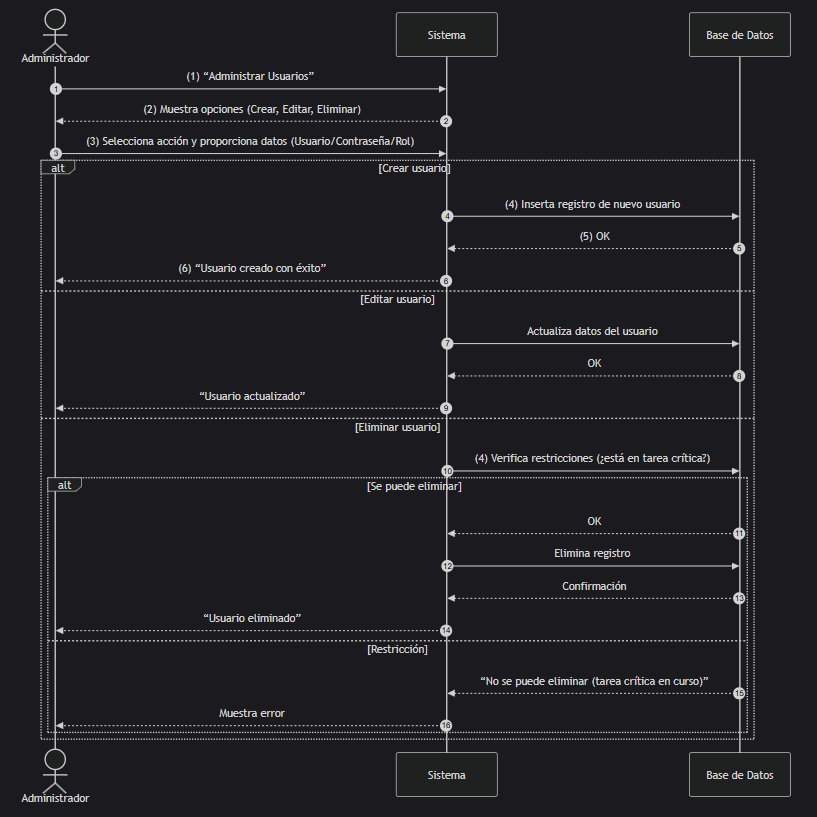


CU10 – Gestionar Alertas y Notificaciones

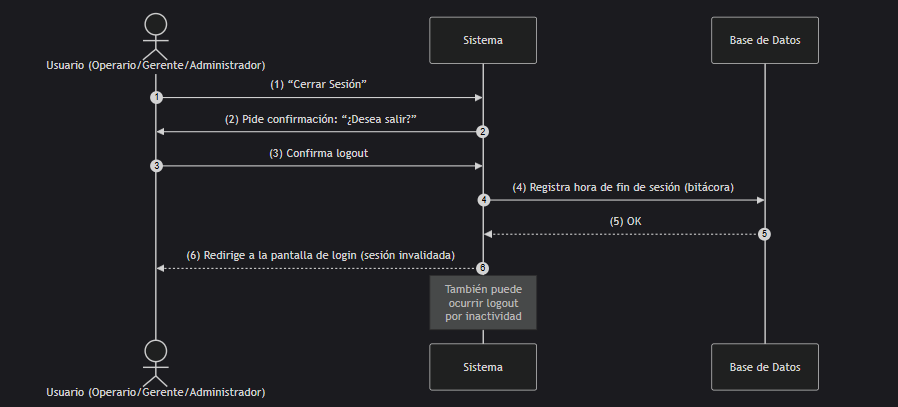
CU11 – Registrar Acciones de Usuario en Bitácora (Auditoría)



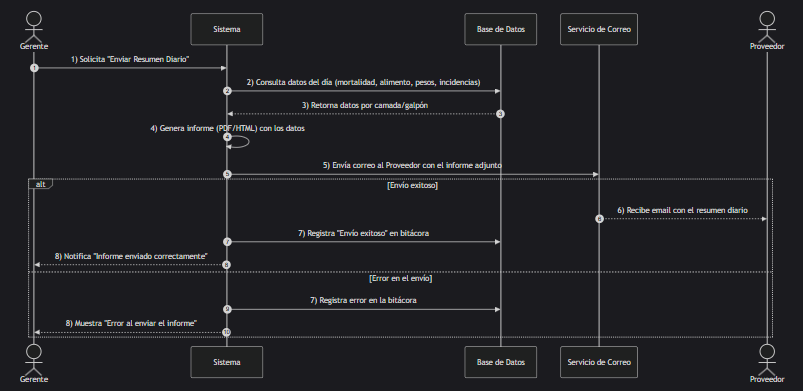
CU12 – Manejo de Roles y Usuarios



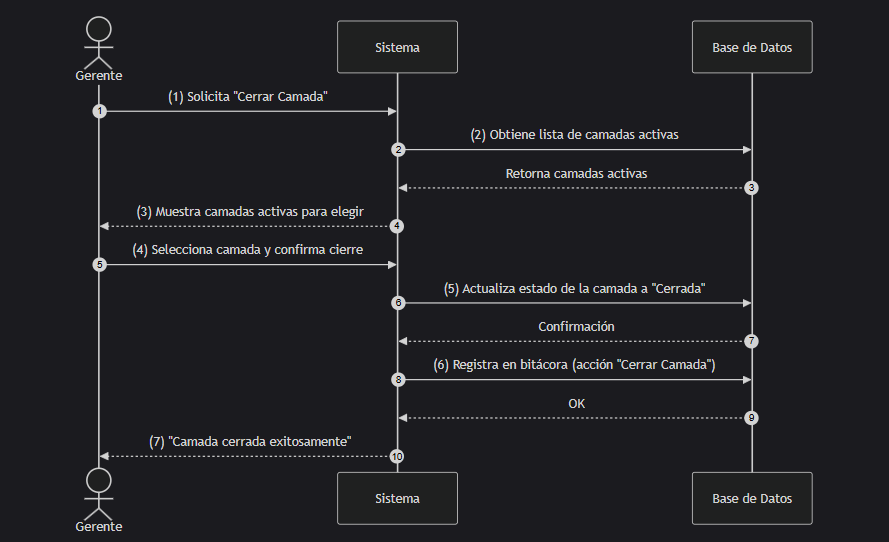
CU13 – Cerrar Sesión



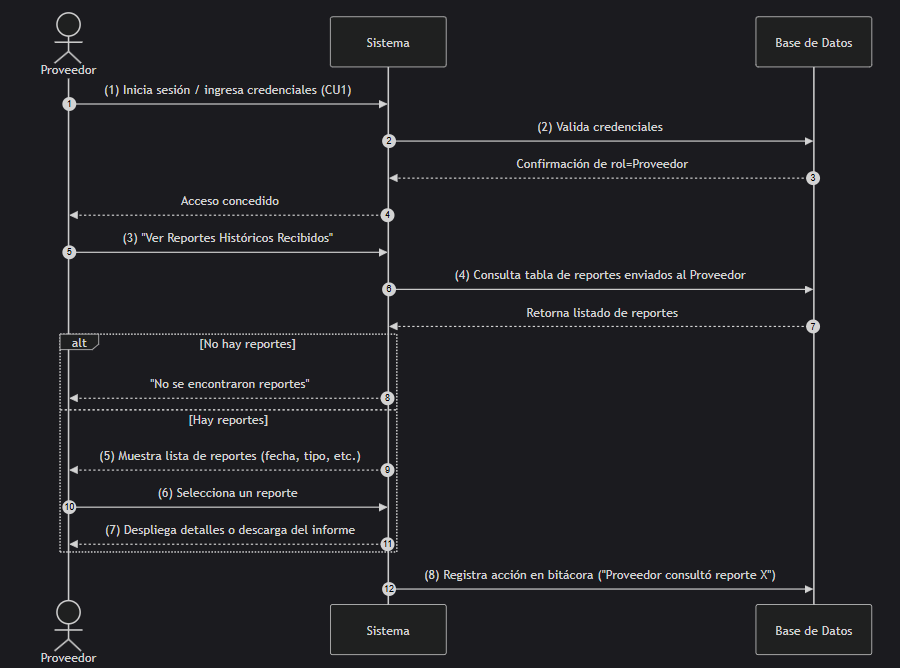
CU14 – Envío de Resumen Diario a Proveedor



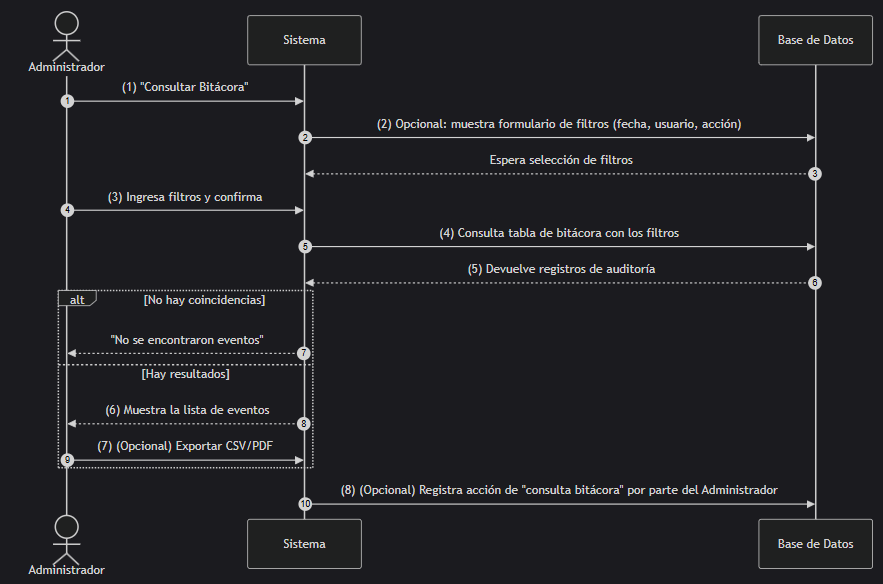
CU15 – Cerrar Camada al Finalizar la Cría



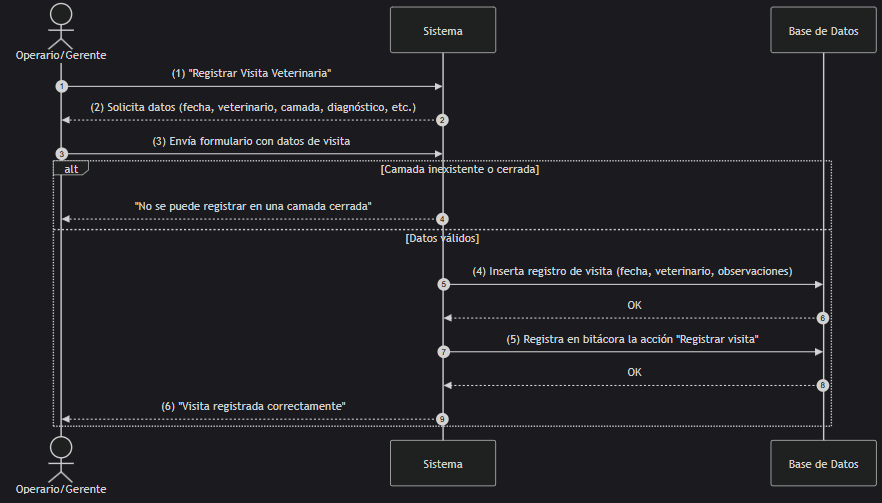
CU16 – Consultar Reportes Históricos Recibidos por el Proveedor



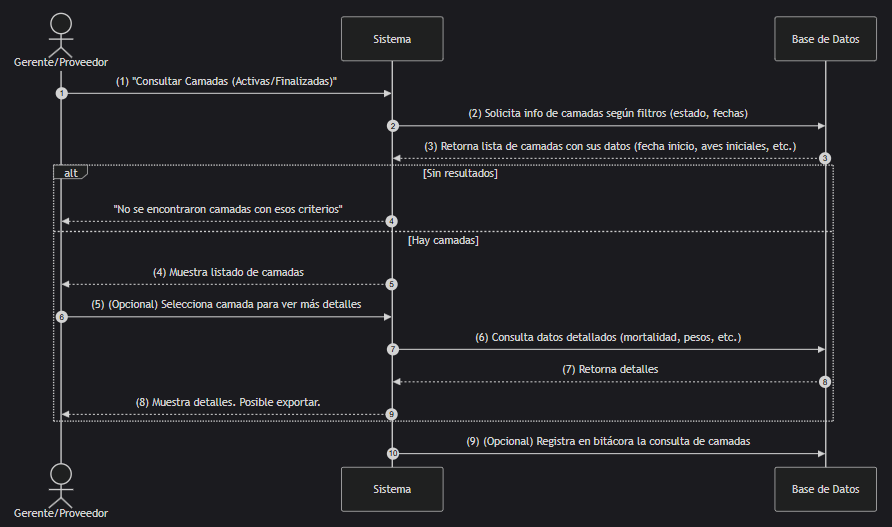
CU17 – Consultar la Bitácora del Sistema



CU18 – Registrar Visitas Veterinarias y Observaciones Sanitarias



CU19 – Consultar Información de Camadas Activas o Finalizadas



# Capítulo X – Enfoque de Arquitectura Modular: Monolito Modular

## X.1 - Introducción al Enfoque Modular

El diseño arquitectónico de un sistema de software cumple un rol fundamental en su mantenibilidad, escalabilidad y capacidad de evolución a largo plazo. En proyectos reales, especialmente aquellos con múltiples áreas funcionales y lógica de negocio diversa, el enfoque tradicional de un *monolito rígido* tiende a generar acoplamientos innecesarios, dificultades en el testing, y problemas de crecimiento estructural a medida que el sistema evoluciona.

Para evitar estas limitaciones, en este proyecto se optó por un enfoque intermedio entre el monolito clásico y una arquitectura de microservicios: el monolito modular. Esta decisión permite organizar la lógica del sistema en módulos funcionales bien definidos, con responsabilidades claras y separadas, manteniendo al mismo tiempo la simplicidad de despliegue de una única aplicación.

Modularizar no implica necesariamente dividir el sistema en múltiples servicios desplegables, sino establecer fronteras internas claras que reduzcan el acoplamiento y mejoren la cohesión. Estas fronteras permiten que cada módulo se desarrolle, pruebe y mantenga en forma relativamente independiente, utilizando principios derivados del enfoque de Domain-Driven Design (DDD), como los *bounded contexts*.

El modelo modular adoptado en este sistema, inspirado en autores como Eric Evans y Vladik Khononov, brinda una solución técnica y conceptual adecuada para sistemas medianos que requieren organización por dominio funcional, sin asumir la complejidad operativa de una arquitectura de microservicios completa desde el inicio.

### X.1.2 - ¿Qué es un Monolito Modular?

El término *monolito modular* hace referencia a un enfoque arquitectónico en el que una aplicación se desarrolla y despliega como una única unidad, pero su estructura interna está claramente **organizada en módulos funcionales independientes**, cada uno con su propia lógica de negocio, modelo de datos y responsabilidades específicas.

A diferencia del **monolito tradicional**, donde todas las funcionalidades suelen estar entrelazadas en una misma base de código sin una separación clara, el monolito modular propone una **compartimentación del dominio** que mejora la organización interna del sistema, favorece el mantenimiento y permite trabajar de forma aislada en cada módulo.

Este enfoque toma como base los principios del **Domain-Driven Design (DDD)** propuestos por *Eric Evans (2004)*, en particular el concepto de *bounded contexts*, donde cada módulo representa una porción autónoma del dominio que puede evolucionar de manera controlada. Además, adopta los lineamientos modernos presentados por *Vladik Khononov (2021)* en su obra *“Learning Domain-Driven Design”*, donde se plantea al monolito modular como una **alternativa viable y realista a los microservicios**, especialmente en fases iniciales o en equipos pequeños.

En un monolito modular:

* Cada módulo contiene su propia lógica de negocio y modelos (entidades, DTOs, servicios).
* No existe acoplamiento directo entre módulos: la comunicación se realiza mediante interfaces bien definidas.
* Se favorece el bajo acoplamiento y la alta cohesión dentro de cada área funcional.
* La estructura del código refleja fielmente la estructura del dominio de negocio.

Este enfoque permite obtener muchas de las **ventajas de los microservicios** (separación de responsabilidades, escalabilidad organizacional, autonomía de desarrollo) sin los inconvenientes de una infraestructura distribuida compleja, como la necesidad de orquestación, monitoreo y gestión de múltiples servicios y bases de datos.

### X.1.3. Aplicación del Enfoque Modular al Sistema Actual

El sistema desarrollado para la gestión avícola fue estructurado siguiendo los principios del enfoque monolito modular, separando claramente cada área funcional del dominio en módulos independientes, tanto a nivel lógico como organizativo. Esta decisión permitió abordar la complejidad del proyecto dividiéndolo en unidades coherentes, fácilmente testeables y mantenibles, sin incurrir en la sobrecarga de un entorno distribuido.

Cada módulo encapsula una parte específica del negocio, incluyendo sus controladores REST, servicios, modelos de datos y lógica de validación. La comunicación entre módulos, cuando es necesaria, se realiza a través de interfaces claramente definidas, sin compartir clases internas ni dependencias cruzadas directas.

Entre los módulos definidos se encuentran:

* Camadas y galpones: alta y cierre de camadas, distribución de aves, subdivisión por galpón.
* Alertas: configuración de umbrales, generación automática y gestión de alertas activas.
* Monitoreo ambiental: recepción y consulta de datos de sensores, variables internas y externas.
* Control de dispositivos: encendido/apagado de luces, ventiladores, cortinas y aspersores mediante comandos MQTT.
* Visitas sanitarias: registro de visitas veterinarias, observaciones y estado sanitario.
* Usuarios y roles: gestión de cuentas, autenticación, roles y permisos.
* Reportes: generación de informes históricos y reportes diarios para el proveedor.
* Bitácora y auditoría: registro de acciones del sistema, eventos críticos y trazabilidad.

Entre las ventajas observadas se encuentran la posibilidad de desarrollar por partes sin interferencia entre módulos, el mantenimiento encapsulado de la lógica de negocio, la reducción de efectos colaterales al modificar una funcionalidad puntual, y el hecho de que este diseño sienta las bases para una futura migración a microservicios si el sistema lo requiere.

### X.1.4. Fundamentos Teóricos

El diseño adoptado en este proyecto se fundamenta en principios ampliamente reconocidos dentro de la ingeniería de software, en particular aquellos derivados del Domain-Driven Design (DDD) y de las propuestas modernas sobre arquitectura modular.

Uno de los pilares conceptuales de esta organización es el trabajo de Eric Evans, quien en su libro *Domain-Driven Design: Tackling Complexity in the Heart of Software* (2004) introdujo el concepto de *bounded context*. Este principio sugiere que cada módulo o subdominio dentro de un sistema debe tener límites claros y una semántica consistente, lo cual permite reducir la complejidad, evitar acoplamientos innecesarios y facilitar la evolución independiente de las distintas partes del sistema.

Sobre este mismo enfoque se apoya Vladik Khononov, quien en su obra *Learning Domain-Driven Design* (O’Reilly, 2021) propone explícitamente el uso de un monolito modular como una solución práctica y escalable para sistemas de tamaño medio o en etapas tempranas. Según Khononov, un monolito bien modularizado permite obtener muchas de las ventajas de los microservicios (separación de responsabilidades, claridad arquitectónica, posibilidad de escalado futuro), sin los costos operativos que conlleva una infraestructura distribuida.

El enfoque modular utilizado en este sistema busca mantener estos límites conceptuales entre dominios funcionales, sin caer en la fragmentación excesiva ni en la sobreingeniería. De esta forma, se logra un balance entre organización interna, simplicidad operativa y preparación para el crecimiento futuro del sistema.

### X.1.5. Conclusión

La elección de un enfoque modular para el diseño del sistema permitió construir una arquitectura sólida, mantenible y preparada para escalar. El modelo de **monolito modular** adoptado combina lo mejor de dos mundos: por un lado, la simplicidad de una aplicación unificada en términos de despliegue y gestión; y por otro, la organización interna clara y desacoplada entre los distintos dominios funcionales.

Gracias a esta separación, el equipo pudo desarrollar cada módulo de forma independiente, facilitar las tareas de prueba y mantenimiento, y preparar el terreno para una futura migración parcial o total hacia una arquitectura de microservicios, si así lo requiere la evolución del sistema.

Apoyado en fundamentos teóricos sólidos como los *bounded contexts* del Domain-Driven Design de Evans, y en propuestas prácticas modernas como las de Khononov, este enfoque se muestra como una estrategia ideal para proyectos como el presente: con múltiples áreas funcionales, requerimientos técnicos exigentes y una visión clara de escalabilidad progresiva.

## 

## X.2.1 Implementación de AWS

Para garantizar una solución moderna, escalable y segura, se decidió implementar toda la infraestructura lógica del sistema en **Amazon Web Services (AWS)**. Esta plataforma en la nube permite desplegar sistemas complejos sin necesidad de mantener servidores físicos, reduciendo costos de infraestructura, simplificando el mantenimiento y asegurando alta disponibilidad.

La naturaleza del proyecto —un sistema de monitoreo y control IoT distribuido en entornos rurales— requiere servicios que permitan manejar múltiples dispositivos, autenticar conexiones seguras, procesar datos en tiempo real y ofrecer una interfaz web accesible desde cualquier dispositivo. AWS ofrece todos estos servicios de manera integrada y con herramientas pensadas específicamente para aplicaciones IoT.

A lo largo de este capítulo se describen los principales servicios utilizados y cómo se interconectan para dar soporte al sistema.

Diagrama de despliegue del sistema con AWS:



### X.2.2. Servicios de AWS utilizados

A continuación se detallan los principales servicios de Amazon Web Services que componen la infraestructura lógica del sistema. Cada uno cumple una función específica dentro del ecosistema, y todos están integrados para trabajar de forma conjunta de manera segura, escalable y sin servidores físicos propios.

#### **a) AWS IoT Core**

Este servicio actúa como el centro de comunicaciones MQTT del sistema. Su principal función es recibir y distribuir mensajes entre los dispositivos físicos (Raspberry Pi) y los servicios de backend.

* Opera como broker MQTT con soporte para TLS (cifrado).
* Cada dispositivo se registra y autentica usando certificados X.509 únicos, lo que permite controlar quién puede publicar o suscribirse a qué topics.
* Permite definir reglas y políticas para limitar el acceso por dispositivo y topic, brindando seguridad granular.

#### **b) AWS Lambda**

Lambda permite ejecutar funciones de backend sin necesidad de servidores. Cada vez que el usuario realiza una acción desde el frontend (por ejemplo, encender un ventilador), se invoca una función Lambda que:

* Valida la acción
* Publica un mensaje MQTT en el topic correspondiente (por ejemplo, galpon2 /ventiladores/commands)
* Opcionalmente registra el evento en la base de datos

Esto permite tener una lógica de control escalable, ligera y de bajo costo.

#### c) Amazon API Gateway

Es el servicio que expone los endpoints HTTP/HTTPS que utiliza la interfaz web. Actúa como puerta de entrada al sistema, recibiendo solicitudes del usuario y redirigiéndolas a las funciones Lambda correspondientes.

* Aplica validaciones básicas (headers, métodos, etc.)
* Controla el tráfico y protege de accesos no autorizados
* Puede aplicar limitaciones por seguridad (throttling)

#### d) Amazon RDS (Relational Database Service)

Es el servicio utilizado para almacenar la información estructurada del sistema: registros de sensores, estados de actuadores, logs de acciones, datos de camadas, visitas veterinarias, alertas, etc.

* Puede utilizar motores como PostgreSQL o MySQL
* Soporta backups automáticos y alta disponibilidad
* Es accedido únicamente desde las funciones Lambda

#### e) Amazon S3 + CloudFront

S3 (Simple Storage Service) se utiliza para almacenar los siguientes recursos:

* Archivos PDF y reportes generados (por ejemplo, los informes diarios)
* Archivos estáticos de la interfaz web

CloudFront funciona como una red de distribución (CDN) para mejorar el tiempo de carga del frontend y reducir latencia, sobre todo en conexiones rurales.

#### f) AWS IAM (Identity and Access Management)

Este servicio gestiona los permisos y accesos a los recursos de AWS.

* Se definen roles para Lambda, IoT Core y API Gateway
* Cada dispositivo IoT tiene una política personalizada que define qué topics puede usar
* Permite asegurar que cada parte del sistema sólo acceda a lo que necesita (principio de menor privilegio)

#### g) AWS Certificate Manager (ACM)

Se encarga de emitir y gestionar los certificados digitales que utilizan los dispositivos para conectarse de forma segura a AWS IoT Core.

* Certificados X.509 para autenticar Raspberry Pi
* Permite revocar o renovar certificados según sea necesario
* Integración nativa con AWS IoT

## X.3.1 Diseño de la Base de Datos

La base de datos del sistema se diseñó utilizando un modelo relacional, en concordancia con la naturaleza estructurada de los datos manejados en el sistema: camadas, sensores, dispositivos, eventos, alertas, usuarios y reportes. Se optó por PostgreSQL como motor de base de datos por su robustez, compatibilidad con JPA/Hibernate y facilidad de despliegue en la nube o servidores autogestionados.

El diseño de la base de datos refleja fielmente los módulos funcionales definidos en el backend. Cada área cuenta con su propio conjunto de tablas, las cuales se relacionan entre sí mediante claves foráneas para garantizar la integridad referencial.

El modelo se encuentra normalizado, evita la duplicación innecesaria de datos, y fue pensado para facilitar consultas eficientes, tanto en tiempo real (dashboard) como para generación de informes. Además, se adoptó la convención de nombres en camelCase, alineado con el estilo del backend y el frontend del sistema.

El diseño completo del modelo puede visualizarse en el Diagrama X.3.1, donde se representan las entidades, relaciones y claves principales. El modelo fue validado en base a los requerimientos funcionales y casos de uso priorizados durante el desarrollo.

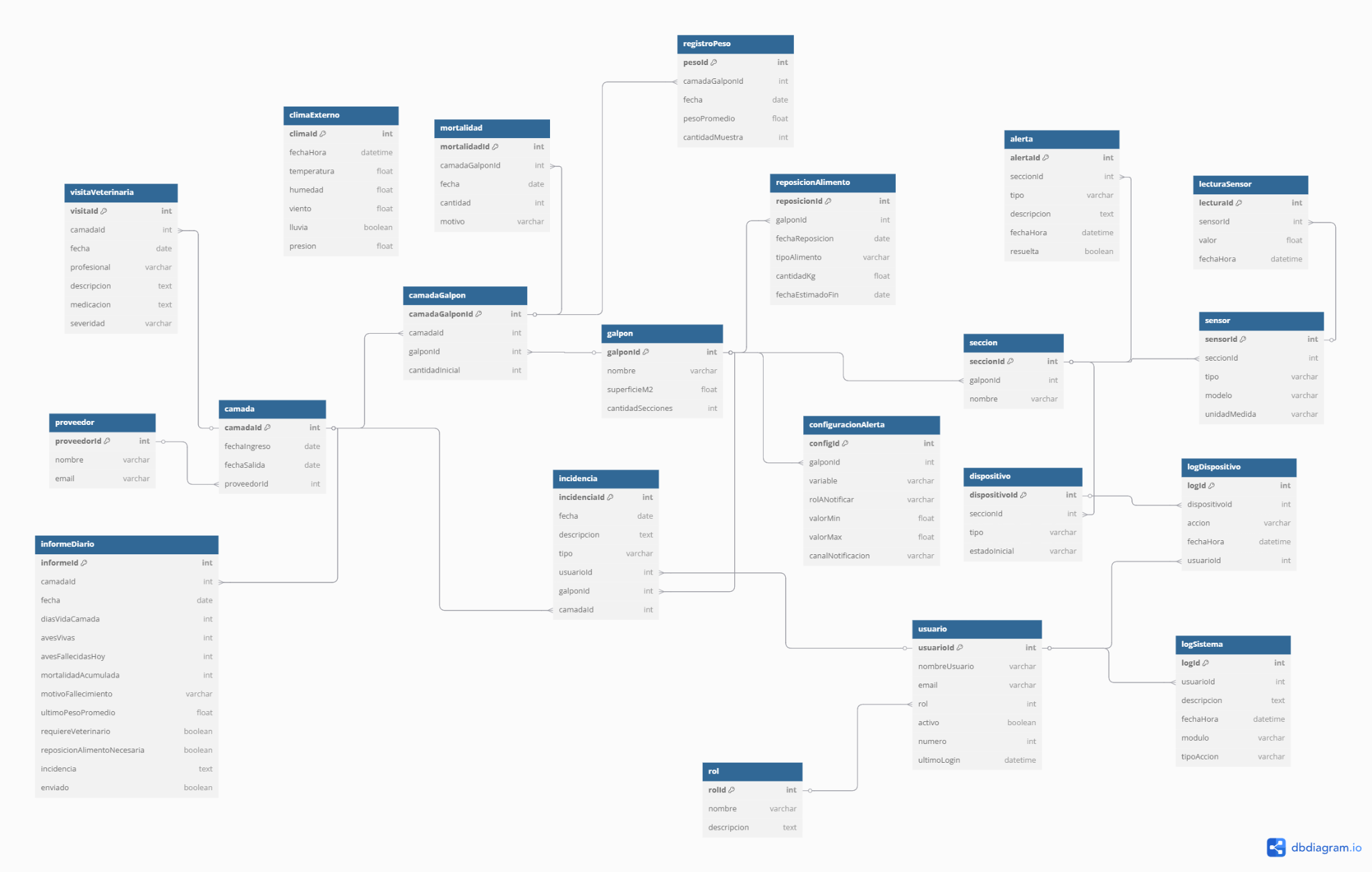


Diagrama X.3.1

## X.4.1. Introducción al Backend del Sistema

El backend del sistema de gestión avícola cumple un rol fundamental como núcleo lógico de la aplicación, actuando como intermediario entre la interfaz de usuario, los servicios en la nube (AWS) y los dispositivos físicos desplegados en la granja. Es responsable de orquestar los procesos centrales del sistema: manejo de camadas, control de dispositivos, monitoreo de sensores, gestión de alertas, usuarios, reportes, entre otros.

Para su diseño e implementación se adoptó un enfoque de monolito modular, que permite organizar el código fuente en módulos funcionales independientes basados en el dominio del sistema. Cada módulo encapsula su propia lógica, datos y responsabilidades, lo que facilita la mantenibilidad, escalabilidad interna y el desarrollo paralelo sin conflictos entre áreas.

El backend se desarrolla utilizando Spring Boot como framework principal, junto con otras tecnologías complementarias:

* **Spring Web**:
  + Módulo utilizado para exponer controladores HTTP/RESTful.
  + Gestiona rutas, verbos HTTP, parámetros y respuesta en formato JSON.
  + Permite interacción con el frontend desplegado en Vercel.
* **Spring Data JPA**:
  + Abstracción sobre JPA (Hibernate) para acceso a base de datos relacional.
  + Simplifica el trabajo con entidades, consultas, paginación y persistencia.
  + Facilita la conexión con PostgreSQL o MySQL mediante anotaciones.
* **Spring Security** + **Keycloak**:
  + Seguridad basada en tokens JWT y OpenID Connect.
  + Control de acceso según roles: Operario, Gerente, Administrador, Proveedor.
  + Integración con Keycloak para autenticación centralizada y autorización granular.

* **MQTT (cliente Eclipse Paho)**:
  + Librería MQTT embebida que permite la publicación y suscripción a topics en AWS IoT Core.
  + Utilizada para enviar comandos desde el backend y recibir confirmaciones.
  + Comunicación segura mediante certificados X.509 y protocolo MQTT con TLS.

* **Swagger**:
  + Generación automática de documentación interactiva de la API REST.
  + Permite probar endpoints desde el navegador y documentar contratos entre backend y frontend.
* **PostgreSQL**:
  + Base de datos relacional para almacenar información estructurada: camadas, sensores, alertas, usuarios, etc.
  + Modelo optimizado con claves foráneas, relaciones normalizadas y convenciones en camelCase.

En conjunto, esta arquitectura permite una solución robusta, extensible y preparada para evolucionar con nuevas funcionalidades, integraciones o incluso transformarse en una arquitectura distribuida en el futuro si las necesidades del sistema lo requieren.

## X.5.1. Introducción al Frontend del Sistema

El frontend del sistema de gestión avícola constituye la interfaz principal de interacción para los usuarios, permitiendo consultar información clave, ejecutar acciones operativas, controlar dispositivos, visualizar alertas, registrar eventos y acceder a reportes. Diseñado con un enfoque centrado en la usabilidad, el sistema busca ofrecer una experiencia ágil, clara y segura desde cualquier dispositivo con acceso a internet.

El sistema contempla diferentes roles de usuario (Operario, Gerente, Administrador, Proveedor), cada uno con accesos y funcionalidades específicas. Por ejemplo, el operario puede registrar mortalidad o encender un ventilador, el gerente puede crear una camada y configurar alertas, y el proveedor puede visualizar informes diarios de evolución.

Para lograr una interfaz moderna, mantenible y escalable, se eligió un framework de frontend robusto, con componentes reutilizables, rutas protegidas por rol, validación de formularios y diseño responsive adaptado a distintos tamaños de pantalla (PC, tablet, móvil).

El frontend se comunica con el backend mediante API REST, y respeta el sistema de autenticación basado en tokens JWT emitidos por el proveedor de identidad (Keycloak). También permite a los usuarios visualizar los datos provenientes de AWS IoT en tiempo real, con posibilidad de ejecutar comandos que terminan actuando sobre dispositivos físicos mediante el backend.

Gracias a su despliegue en la nube a través de Vercel, la interfaz puede ser accedida desde cualquier lugar, sin necesidad de infraestructura local, lo que es ideal para una granja familiar con múltiples responsables en campo y en oficina.

### X.5.2. Tecnologías Utilizadas en el Frontend

La interfaz web del sistema fue desarrollada utilizando tecnologías modernas del ecosistema JavaScript, seleccionadas por su madurez, extensibilidad y compatibilidad con entornos cloud. Estas herramientas permiten construir una aplicación robusta, modular, visualmente atractiva y fácil de mantener.

A continuación se detallan las principales tecnologías utilizadas:

* **React.js**: Framework principal para el desarrollo del frontend. React permite construir interfaces de usuario basadas en componentes reutilizables, con una gestión eficiente del estado y actualizaciones dinámicas de la vista.
* **Tailwind CSS + Material UI**: Se combinan dos estrategias de diseño visual. Tailwind CSS es un framework de utilidades CSS que permite construir estilos directamente en los componentes. Material UI (MUI) es una librería de componentes visuales que respeta las guías de diseño de Google, utilizada para elementos como botones, tarjetas, menús, inputs, tablas e íconos.
* **React Router**: Permite la navegación entre distintas vistas de la aplicación sin recargar la página. Se utiliza para secciones como login, dashboard, gestión de camadas, reportes, etc. Las rutas están protegidas mediante verificación de rol.
* **Axios**: Librería utilizada para realizar solicitudes HTTP al backend. Permite consumir los endpoints de la API REST, por ejemplo, para obtener alertas activas, enviar datos, actualizar estados de dispositivos, entre otros.
* **JWT / Keycloak**: La autenticación se basa en tokens JWT emitidos por Keycloak. Al iniciar sesión, el usuario recibe un token que se almacena y se utiliza en cada solicitud para verificar identidad y permisos. Se implementaron login, logout, protección de rutas y control de visibilidad según rol.
* **Vercel**: Plataforma utilizada para desplegar la aplicación web. Permite builds automáticos, integración con Git y acceso global mediante dominio propio. Es ideal para este tipo de proyecto por su facilidad de uso y rendimiento.

### X.5.3 Comunicación con el Backend

La comunicación entre la interfaz web y el backend se realiza mediante API RESTful, utilizando el protocolo HTTP/HTTPS. Las solicitudes se gestionan con la librería Axios, y cada endpoint del backend responde en formato JSON, cumpliendo con los contratos definidos en Swagger/OpenAPI.

**Consumo de endpoints**

Cada módulo del sistema cuenta con su propio servicio en el frontend, organizado por funcionalidad, por ejemplo:

* /services/camadasService.js para crear o cerrar camadas
* /services/alertasService.js para obtener alertas activas y resolverlas
* /services/dispositivosService.js para enviar comandos a luces, ventiladores, etc.

Las funciones dentro de estos servicios realizan solicitudes HTTP usando Axios, por ejemplo:

* GET /api/camadas/activas
* POST /api/ventiladores/encender

### X.5.4Autenticación y envío de tokens

La autenticación en el sistema se basa en JWT (JSON Web Tokens) emitidos por Keycloak. Cuando el usuario inicia sesión correctamente, recibe un token que se guarda en memoria (generalmente en localStorage o contexto React) y se incluye automáticamente en el header de cada solicitud:

Authorization: Bearer {token}

Este token permite:

* Identificar al usuario
* Determinar su rol (Operario, Gerente, etc.)
* Proteger rutas específicas en el frontend
* Validar acceso en los endpoints del backend

### X.5.5 Manejo de errores y feedback

El frontend maneja los errores devueltos por el backend mediante interceptores de Axios o validaciones específicas. Algunos ejemplos:

* Si un token es inválido o expiró → redirección automática al login
* Si un usuario intenta acceder a una función sin permisos → mensaje “Acceso denegado”
* Si ocurre un error de red o el backend no responde → se muestra un mensaje claro para el usuario

Cada operación (como registrar un peso o encender un ventilador) proporciona una respuesta visual inmediata, con feedback positivo (mensaje de éxito) o negativo (alerta con detalle del error).

# 

# Capítulo XI – Diseño del sistema

## XI.1. Arquitectura General

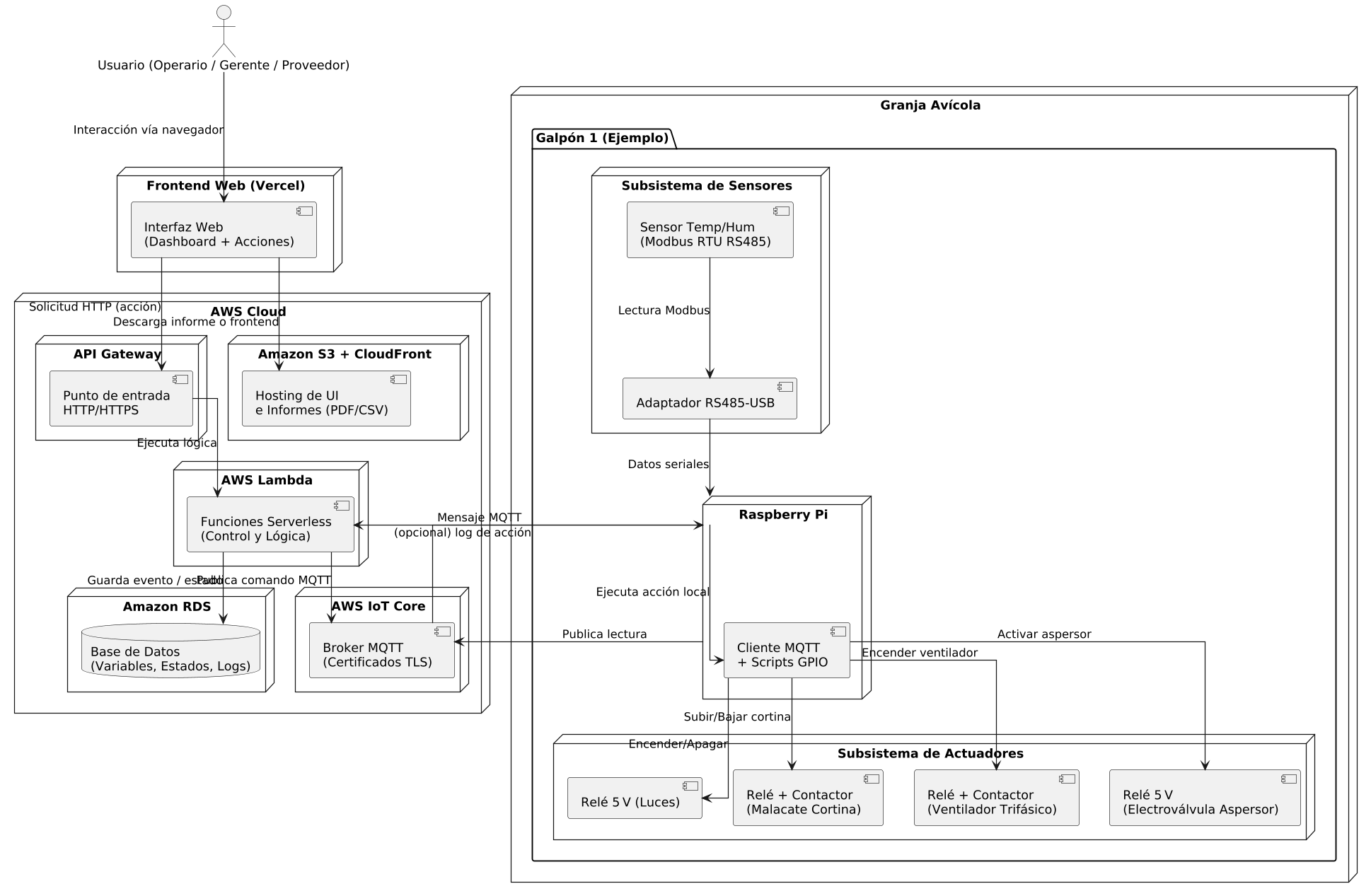
El sistema de gestión avícola desarrollado se basa en una arquitectura distribuida que combina componentes físicos locales (sensores, actuadores y controladores en los galpones) con servicios en la nube para el procesamiento, almacenamiento y visualización de datos. Esta arquitectura permite brindar una solución integral que cubre las necesidades de monitoreo y control remoto en tiempo real, asegurando escalabilidad, modularidad y facilidad de mantenimiento.

Cada galpón cuenta con una Raspberry Pi que actúa como controlador local, encargada de interactuar con los sensores y dispositivos eléctricos (luces, ventiladores, aspersores y malacates). Estas unidades se comunican con la nube a través de internet, utilizando el protocolo MQTT para el envío y recepción de mensajes. Por su parte, el sistema web (frontend) permite que los distintos usuarios (Operario, Gerente, Proveedor, etc.) visualicen el estado del sistema y ejecuten acciones desde cualquier dispositivo conectado.

En la nube, los servicios de AWS (Amazon Web Services) proporcionan la infraestructura necesaria para gestionar esta comunicación de manera segura y eficiente. Se utilizan AWS IoT Core como broker MQTT, Lambda para procesar las peticiones desde el frontend y API Gateway para exponer las funciones al cliente. La base de datos y los recursos estáticos (informes, interfaz web) también están alojados en la nube.

Esta separación entre la capa física (en campo) y la capa lógica (en la nube) permite una arquitectura altamente modular, donde cada parte del sistema puede evolucionar o escalar de manera independiente.

A continuación se presenta el diagrama de despliegue de esta arquitectura:



## XI.2. Sistema AWS IoT y la Integración con MQTT

Para la comunicación entre los distintos componentes del sistema, se optó por utilizar el protocolo MQTT (Message Queuing Telemetry Transport), un protocolo de mensajería liviano, ideal para sistemas embebidos y redes con limitaciones de ancho de banda. MQTT permite establecer una arquitectura pub/sub (publicación y suscripción), donde los dispositivos publican datos o comandos en determinados canales llamados “topics”, y otros dispositivos o servicios (como la Raspberry Pi o el backend) se suscriben a esos topics para actuar en consecuencia.

El servicio AWS IoT Core cumple el rol de broker MQTT, recibiendo los mensajes enviados desde el backend o desde los dispositivos y redirigiéndolos hacia su destino. Este broker opera bajo el protocolo TLS, utilizando certificados X.509 únicos por dispositivo, lo que garantiza una comunicación segura y autenticada en ambas direcciones.

Cuando un usuario interactúa con la interfaz web (por ejemplo, para encender un ventilador o consultar los sensores), el frontend envía una solicitud a API Gateway, que luego invoca una función AWS Lambda. Esta función genera un mensaje MQTT en el topic correspondiente (por ejemplo, galpon1/ventiladores/commands) y lo publica a través de AWS IoT Core. La Raspberry Pi suscripta a ese topic recibe el mensaje, interpreta la orden y actúa en consecuencia sobre el GPIO necesario.

Del mismo modo, las Raspberry Pi también pueden publicar mensajes hacia la nube, como lecturas de sensores, estados de actuadores o confirmaciones de comandos recibidos, utilizando topics como galpon2/sensores/lecturas o galpon3/luces/status.

Este modelo desacoplado, basado en mensajería, permite una comunicación asíncrona, segura y escalable, ideal para entornos productivos como el avícola, donde se requiere confiabilidad y capacidad de respuesta en tiempo real.

## XI.3. Capitulo de conexión



## XI.4. Sensado Ambiental: Monitoreo de Temperatura y Humedad en Galpones

¿Cómo se van a comunicar los sensores con nuestro sistema?

1. Sensores Modbus RTU

* Dispositivo: El sensor de temperatura y humedad con interfaz RS485/Modbus (por ejemplo, el que compartiste en MercadoLibre).
* Cantidad: Uno o varios, según cuántos puntos de medición quieras.
* Direcciones Modbus: Cada sensor se configura con una dirección única (p. ej. 1, 2, 3…) en el mismo bus físico.

2. Cableado RS485

* Cable: Un cable twisted pair (par trenzado) para RS485, preferentemente de categoría industrial (ej. Belden 3105A o cable similar).
* Distancias: RS485 soporta largas distancias (varios cientos de metros), pero la calidad del cable y la velocidad de baudios influyen.
* Topología: Generalmente bus lineal (daisy chain), con terminadores al principio y fin del bus (resistencia de 120 Ω).

3. Dispositivo Maestro (Gateway / Microcontrolador)

* Función: Actuará como “maestro Modbus”, interrogando a los sensores esclavos por su dirección Modbus y leyendo los registros de temperatura/humedad.
* Ejemplos:
  + Raspberry Pi + Adaptador RS485 (USB-RS485 o HAT RS485).
* Requerimientos:
  + Debe tener soporte físico para RS485.
  + Un software/librería que haga de maestro Modbus (p.ej. pymodbus en Python, ModbusMaster en Arduino, etc.).

4. Conexión a Internet o Red Local

* Si usas una Raspberry Pi u otro gateway con Linux, típicamente se conectará vía Ethernet o WiFi a la red de la granja.
* Esto permitirá que, una vez que lea las variables por Modbus, publique los datos (temperatura/humedad) en el broker MQTT (en la nube o en un servidor local).

5. Broker MQTT

* Servidor que recibe los mensajes publicados por el gateway y los pone a disposición de los suscriptores (por ejemplo, tu backend).
* Si se encuentra “en la nube”, deberás configurar la Raspberry Pi (o maestro) para conectarse por Internet/TLS al broker externo.

6. Backend / Servidor de Aplicaciones

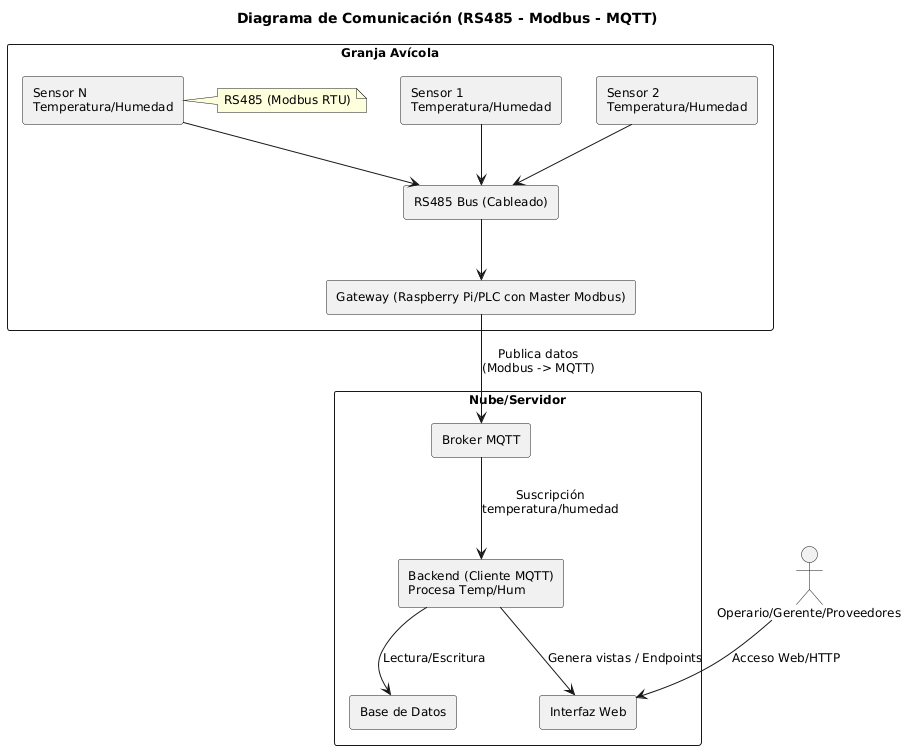
* Cliente MQTT que se suscribe a los topics donde el gateway publica.
* Al recibir un mensaje (JSON con la lectura de temperatura/humedad), lo guarda en la base de datos y dispara alertas o acciones si es necesario.

Granja Avícola: Contiene sensores (S1, S2, S3) conectados por bus RS485 (RSB) a un *gateway* (G).

Nube/Servidor: Incluye el Broker MQTT (B), el Backend (APP) y la Base de Datos (DB).

Flujo:

* El Gateway lee los sensores vía Modbus RTU/RS485 y publica al Broker MQTT.
* El Backend está suscrito a esos topics, procesa los datos y los guarda en la BD.
* Opcionalmente, el Backend (o un usuario) puede ordenar acciones (encender ventiladores, etc.), publicando otro mensaje al Broker, que el Gateway recibe para actuar localmente.



## XI.5. Gestión de Malacates Eléctricos con Control Proporcional

El sistema permite que el Operario o Gerente pueda subir o bajar de forma remota las cortinas laterales de los galpones, las cuales están accionadas mediante malacates eléctricos. El control se realiza desde la interfaz web del sistema, mediante el envío de comandos que se traducen en señales para activar los motores de los malacates a través de relés y contactores, utilizando la infraestructura de comunicación MQTT sobre AWS IoT Core.

Cada galpón cuenta con dos cortinas, una por cada lateral, y cada una puede ser controlada de forma independiente. Las cortinas pueden ser operadas con acciones simples como "subir" o "bajar", o con comandos más precisos que indican un porcentaje de apertura o cierre, por ejemplo, “bajar al 50%”.

Para implementar esta funcionalidad, se utiliza una lógica basada en control por tiempo. Se determina previamente cuánto tiempo tarda cada malacate en realizar el recorrido completo de subida o bajada (por ejemplo, 15 segundos). A partir de esa medición, se calcula el tiempo proporcional necesario para alcanzar un porcentaje deseado. Por ejemplo, si se desea bajar la cortina al 50%, el motor se acciona durante la mitad del tiempo total.

Este enfoque permite un control más flexible de la ventilación natural en los galpones, adaptando la apertura de las cortinas a las condiciones ambientales.

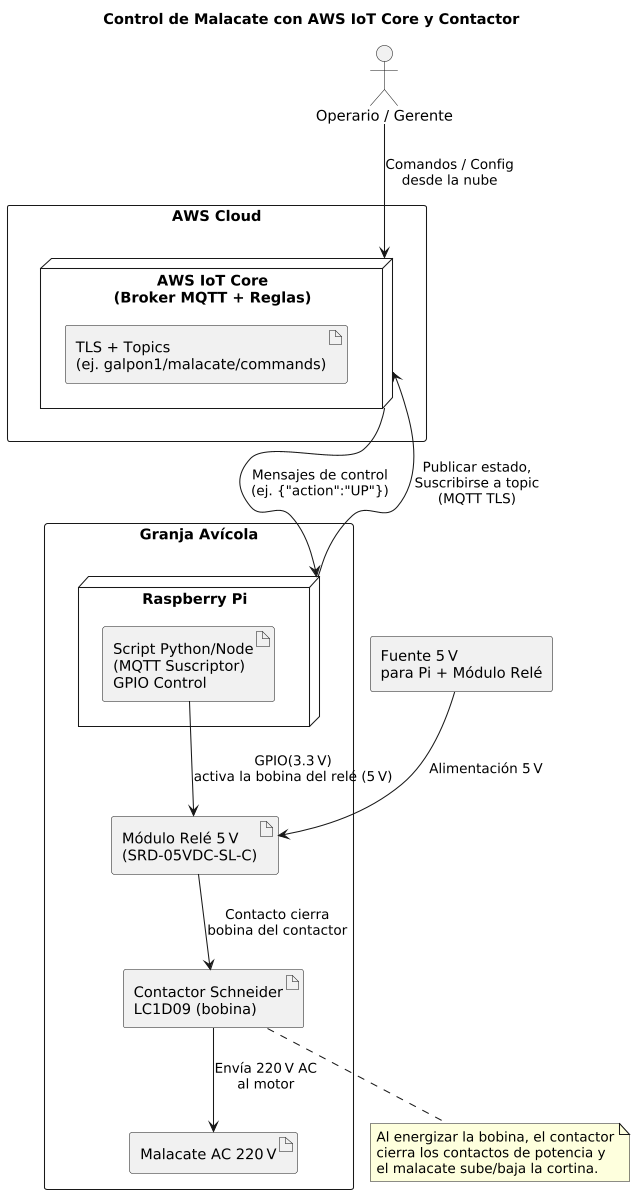
Diagrama de la arquitectura de comunicación Raspberry - Malacate

Comunicación entre el Operario/Gerente y la Raspberry Pi a través de AWS IoT Core

En el primer diagrama, titulado:  
 “Control de Malacate con AWS IoT Core y Contactor”  
 (se ve el Operario/Gerente, AWS IoT Core y la Raspberry Pi en la granja), se muestra el flujo de mensajes para que el malacate pueda accionarse de manera remota:

1. Operario/Gerente (parte superior del diagrama)
   * Realiza una acción en la interfaz web o en alguna herramienta que se comunique con la nube (por ejemplo, a través del Backend o directamente con AWS IoT).
   * Envía un comando para subir o bajar la cortina (por ejemplo, {"action": "UP"}).
2. AWS IoT Core
   * Actúa como Broker MQTT. Recibe los mensajes publicados por el sistema y los distribuye a los clientes suscritos.
   * Define “topics” (por ejemplo, galpon1/malacate/commands) donde se publican las órdenes de control.
   * Utiliza TLS y certificados para asegurar la comunicación con la Raspberry Pi.
3. Raspberry Pi
   * En la Granja Avícola, la Pi corre un script (Python, Node.js, etc.) que se suscribe al topic de órdenes (galpon1/malacate/commands).
   * Cuando llega un mensaje del tipo {"action": "UP"}, la Pi interpreta que debe energizar el contactor para subir la cortina.
   * Asimismo, la Pi puede publicar en otro topic (p. ej. galpon1/malacate/status) para informar estados (si está subiendo, si finalizó, etc.).

En este primer diagrama, se ve cómo el Operario/Gerente, a través de AWS IoT Core, puede enviar comandos (UP, DOWN, STOP) y la Raspberry Pi recibe dichos comandos mediante MQTT. Es la comunicación a alto nivel entre el “mundo nube” y el “mundo físico” de la granja.



Esquema Simplificado: Raspberry Pi + Contactor (Schneider LC1D09) + Módulo de Relé (SRD-05VDC-SL-C)

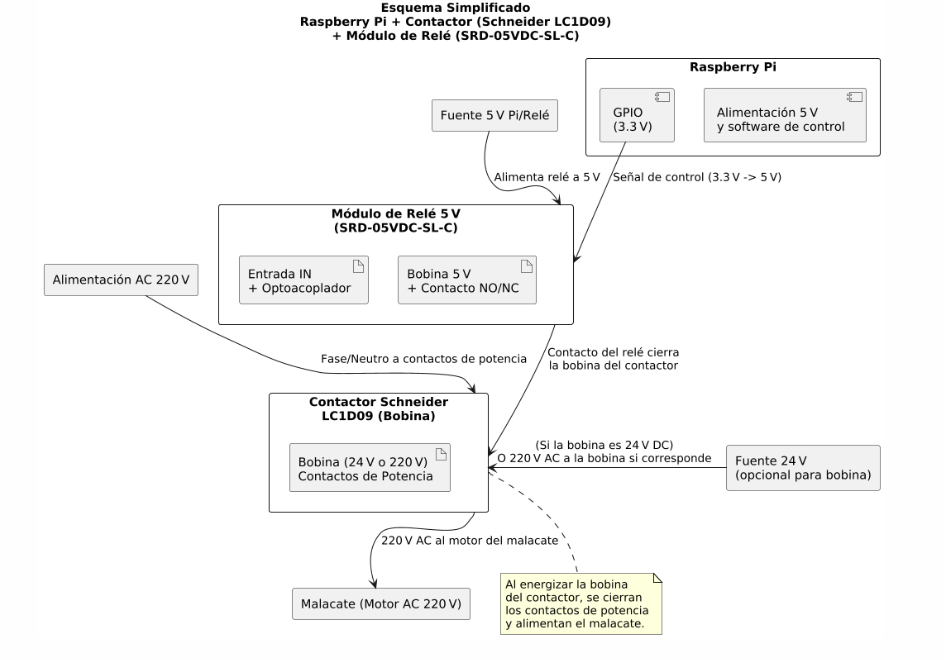
Detalle de los Componentes Físicos: Raspberry Pi, Módulo de Relé, Contactor y Malacate

En el diagrama se muestra la parte local, es decir, cómo la Raspberry Pi energiza realmente el motor del malacate:

1. Raspberry Pi
   * Tiene un GPIO (3,3 V) que envía una señal de control al módulo de relé.
   * Se alimenta con 5 V (por ejemplo, la fuente oficial de la Pi), y el script local (Python/Node) gestiona las órdenes recibidas por MQTT.
2. Módulo de Relé 5 V (SRD-05VDC-SL-C)
   * Este módulo amplifica la señal de 3,3 V de la Pi para cerrar un contacto que alimenta la bobina del contactor.
   * Se alimenta a 5 V y tiene un optoacoplador y una bobina interna de 5 V que, al ser excitada, cierra un contacto NO/NC.
   * El contacto que se cierra no es para el motor, sino para la bobina del contactor.
3. Contactor Schneider LC1D09
   * Es el elemento de potencia capaz de soportar la corriente del motor (220 V AC).
   * Tiene una bobina (que puede ser de 24 V DC, 220 V AC, etc.). Al energizarla, cierra los contactos de potencia y deja pasar la línea de 220 V al motor del malacate.
   * El diagrama deja claro que, si la bobina es de 24 V DC, se requerirá una fuente de 24 V que se conectará al contacto del relé. Si la bobina es de 220 V, ese contacto conmuta directamente la fase de 220 V hacia la bobina.
4. Malacate (Motor AC 220 V)
   * Una vez el contactor está energizado, la fase/neutro alimentan el motor monofásico, y el malacate comienza a subir o bajar (dependiendo de si es un sistema reversible o si se maneja con un segundo contactor para inversión de giro).

En este segundo diagrama se aprecian conexiones más concretas:

* GPIO Pi (3,3 V) → Módulo de Relé (5 V).
* Contacto de relé cierra la alimentación de la bobina del contactor.
* Contactor envía 220 V al motor.



Ejemplo Práctico: Paso por Todos los Componentes

Supongamos que el Operario en la web hace clic en “Subir Cortina Galpón 1”:

1. Front-end
   * Llama a una API (o Lambda) en la nube que internamente publica en galpon1/malacate/commands el JSON: {"action":"UP"}.
2. AWS IoT Core
   * Recibe ese mensaje y, al ver que la Raspberry Pi del galpón 1 está suscrita, se lo envía mediante MQTT con TLS.
3. Raspberry Pi
   * El script local (Python) revisa si el action es “UP” o “DOWN”. Ve que es “UP”.
   * Setea su GPIO #17 (por ejemplo) a HIGH, energizando el Módulo de Relé.
4. Módulo de Relé (SRD-05VDC-SL-C)
   * Al recibir el nivel HIGH y la alimentación 5 V, cierra su contacto interno.
   * Ese contacto lleva la tensión (24 V DC o 220 V AC) a la bobina del Schneider LC1D09.
5. Contactor LC1D09
   * Energizado, cierra los contactos de potencia, dando 220 V AC al motor del malacate.
   * El malacate comienza a subir la cortina.
6. Fin de Carrera/Temporizador (opcional)
   * La Pi, tras 10 segundos o al detectar un sensor, manda STOP. Pone GPIO #17 a LOW, el relé se abre, el contactor se desenergiza y el motor se detiene.
7. La Pi publica Estado
   * Puede publicar: {"status":"UP\_COMPLETE"} o algo similar en galpon1/malacate/status.
   * El Operario ve en la interfaz que el proceso finalizó.

## XI.6. Control de Iluminación: Gestión Remota de Luces por Galpón

1. Objetivo

El sistema busca permitir que el Operario o Gerente pueda encender o apagar las luces de cada galpón (o de cada sección si se implementa así) desde la interfaz web, utilizando tecnologías IoT seguras y escalables.

Esto se logra a través de la integración de:

* AWS IoT Core como broker MQTT.
* Raspberry Pi en cada galpón como controlador local.
* Módulo de relé para accionar luces alimentadas con 220 V AC.
* Mensajería MQTT segura mediante TLS.

2. Flujo general de funcionamiento

El proceso completo de encendido/apagado de luces sigue el siguiente flujo lógico:

1. El usuario accede al sistema web desde cualquier dispositivo.
2. Desde la interfaz principal o el dashboard de un galpón, solicita encender o apagar las luces.
3. El sistema envía un mensaje MQTT con el comando correspondiente ({"action": "ON"} o {"action": "OFF"}) a través de AWS IoT Core, en un topic específico del galpón (por ejemplo, galpon1/luces/commands).
4. La Raspberry Pi del galpón correspondiente está suscrita a ese topic. Cuando recibe el mensaje:
   * Interpreta el contenido JSON.
   * Activa o desactiva el GPIO correspondiente según la acción.
5. El GPIO controla un módulo de relé de 5 V, el cual:
   * Al recibir señal, cierra su contacto y permite el paso de 220 V AC hacia el circuito de luces.
   * Si la acción es "OFF", abre el contacto y apaga la alimentación de las luces.
6. Opcionalmente, la Raspberry Pi puede publicar un mensaje de estado en otro topic (galpon1/luces/status) para que el sistema informe si la acción se ejecutó correctamente.

3. Arquitectura de comunicación

3.1 MQTT y AWS IoT Core

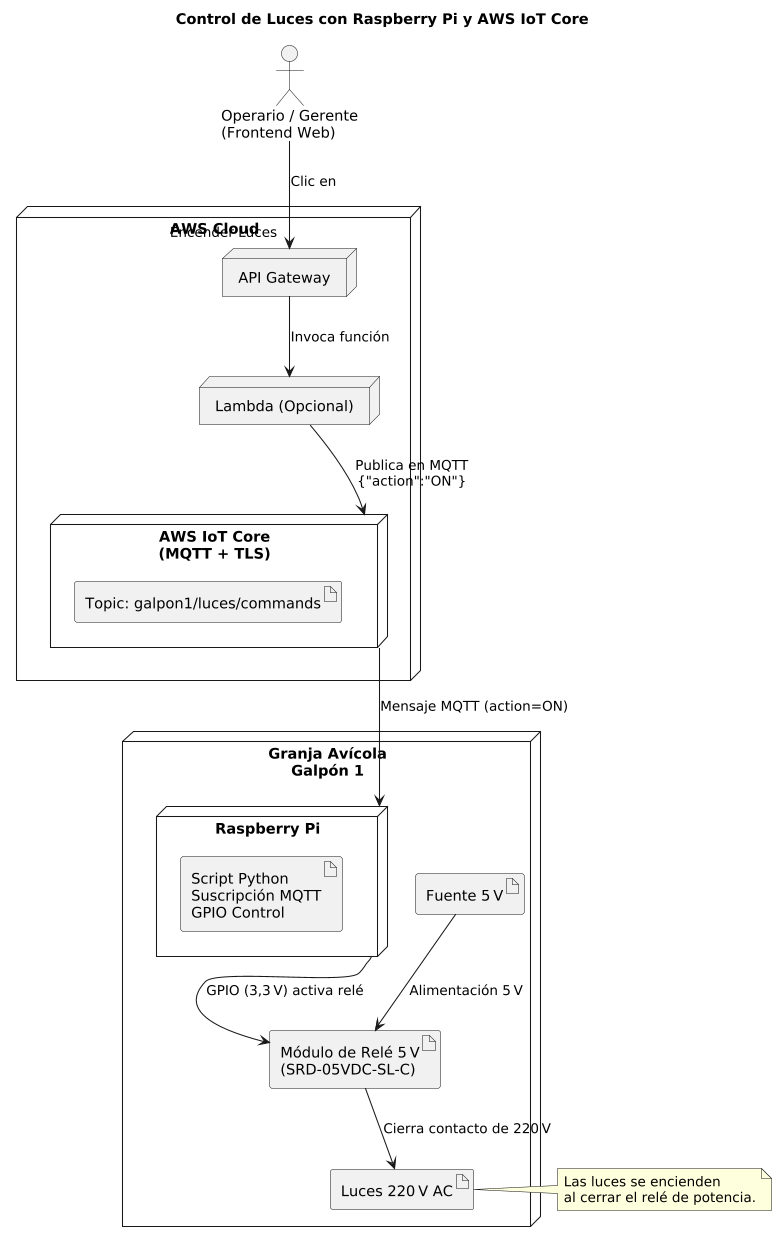
Se utiliza MQTT como protocolo de mensajería ligera, ideal para dispositivos embebidos con bajo consumo. Los componentes involucrados son:

* Broker: AWS IoT Core, que garantiza:
  + Seguridad mediante TLS + certificados X.509.
  + Escalabilidad y enrutamiento por topics.
* Topic ejemplo: galpon1/luces/commands

La Raspberry Pi se conecta como cliente MQTT usando un certificado propio y se suscribe a su topic correspondiente.

3.2 Raspberry Pi y Módulo de Relé

* La Raspberry Pi ejecuta un script en Python (u otro lenguaje) que:
  + Escucha permanentemente el topic galponX/luces/commands.
  + Al recibir {"action":"ON"}, activa un pin GPIO (por ejemplo, GPIO17).
  + Al recibir {"action":"OFF"}, desactiva ese mismo pin.
* El GPIO se conecta a un módulo de relé de 5 V (modelo SRD-05VDC-SL-C).  
   Este relé:
  + Tiene una bobina alimentada a 5 V (desde la Pi o una fuente externa).
  + Cuando recibe señal, cierra un contacto seco que activa el circuito de 220 V AC para las luces.



## XI.7. Sistema de Aspersores: Activación Seccional de Válvulas para Enfriamiento Ambiental

El sistema está diseñado para que el Operario o Gerente pueda activar o desactivar los aspersores de cada galpón (por sección), de manera remota, segura y eficiente. El objetivo principal es refrescar el ambiente interno en los días de calor mediante un sistema de microaspersores que humidifican el aire sin mojar a los animales ni el piso.

Cada galpón está dividido en secciones, y cada sección cuenta con un microaspersor, el cual se activa por medio de una electroválvula de baja tensión (12 V o 24 V DC). Estas electroválvulas son controladas por relés, que a su vez son accionados por una Raspberry Pi ubicada en el mismo galpón.

Las órdenes de riego se envían desde el sistema web (interfaz del usuario) a través del protocolo MQTT, y se distribuyen por medio de AWS IoT Core. La Raspberry Pi, al recibir la orden, activa el GPIO correspondiente que controla el relé y permite la apertura de la válvula.

| Componente | Función |
| --- | --- |
| Microaspersores | Dispersan agua en forma de niebla para refrescar. |
| Electroválvulas 12/24 V | Controlan el paso de agua hacia cada aspersor. |
| Raspberry Pi | Recibe órdenes vía MQTT y activa GPIOs. |
| Módulo de relé 5 V | Permite controlar las válvulas desde la Raspberry Pi. |
| Fuente 12/24 V DC | Alimenta todas las electroválvulas. |
| AWS IoT Core (MQTT) | Comunicación segura nube–dispositivo. |
| Frontend Web | Interfaz donde el usuario acciona aspersores. |



Flujo de funcionamiento

1. El Operario accede a la interfaz web y selecciona una sección de un galpón.
2. Al presionar el botón de “Activar riego”, el sistema genera un mensaje MQTT con un JSON como este: {"seccion": 3, "accion": "ON"}
3. El mensaje se publica en el topic MQTT del galpón:  
    galpon1/aspersores/commands
4. La Raspberry Pi del galpón 1 está suscrita a este topic.  
    Al recibir el mensaje, ejecuta un script que:
   * Verifica la sección especificada.
   * Activa el GPIO correspondiente a esa sección.
5. El GPIO activa un canal del módulo de relé, el cual cierra el circuito de 12/24 V hacia la electroválvula.
6. La electroválvula se abre, y el microaspersor se activa, rociando agua por un tiempo determinado.
7. La Pi puede publicar en otro topic (ej. galpon1/aspersores/status) un mensaje confirmando la ejecución de la acción.

## XI.8. Sistema de Ventilación: Control de Ventiladores Trifásicos por Sección

Este subsistema permite que el Operario o Gerente pueda activar o desactivar los ventiladores industriales de cada sección del galpón, mediante una interfaz web, como parte de la estrategia de control ambiental. Dado que se utilizan ventiladores trifásicos de alta potencia, el control debe realizarse de forma segura, usando contactores.

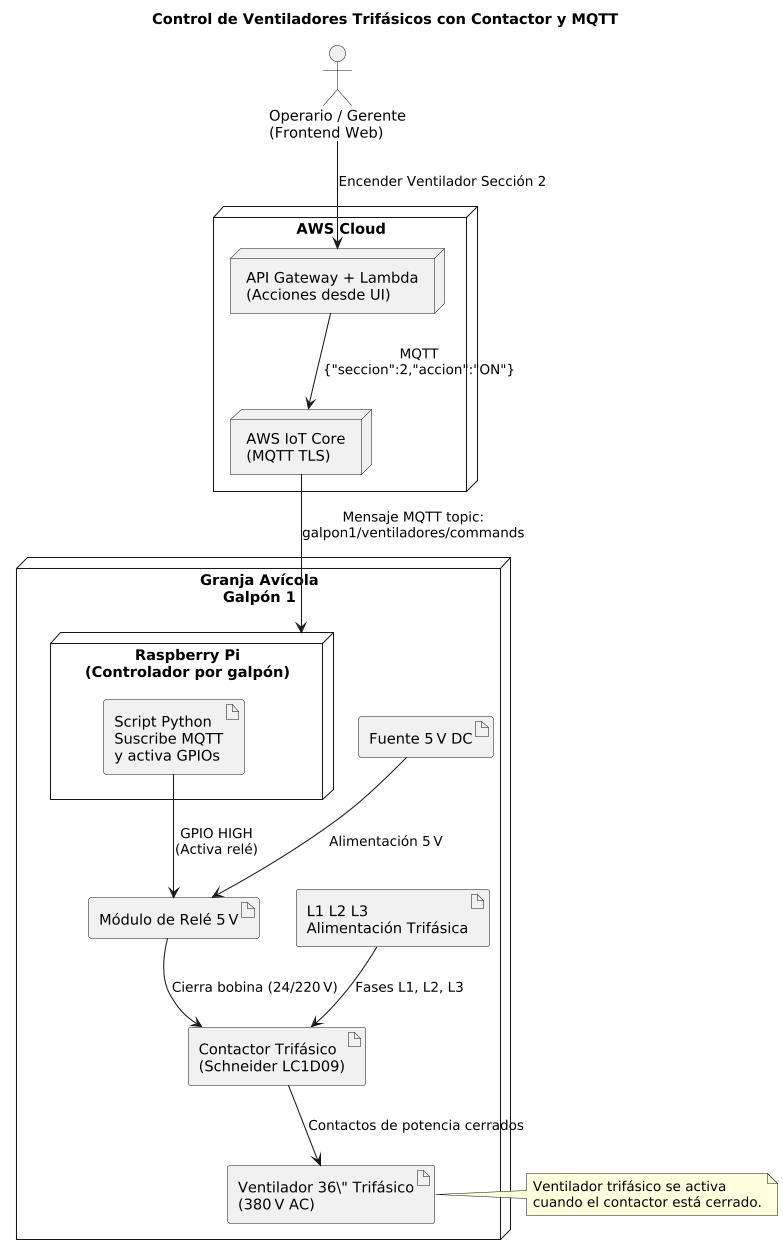
Cada galpón está dividido en varias secciones, y cada sección está equipada con un ventilador extractor de 36" trifásico, modelo:

Ventilador Avícola Porcino 36" Trifásico Aspa Plástica

* Motor trifásico (380 V AC)
* Gran caudal de aire
* Uso intensivo para climatización en ambientes de cría intensiva

| Componente | Función |
| --- | --- |
| Ventilador trifásico 36" (380 V) | Ventilación mecánica potente en cada sección del galpón |
| Contactor trifásico (ej. LC1D09) | Permite cortar o alimentar las tres fases del ventilador |
| Módulo de relé 5 V | Controla la bobina del contactor mediante un GPIO de la Pi |
| Raspberry Pi | Dispositivo controlador por galpón, recibe órdenes MQTT |
| Fuente 5 V DC | Alimenta el relé (y opcionalmente la Pi) |
| AWS IoT Core | Broker MQTT con seguridad TLS |
| API Gateway + Lambda | Genera y publica comandos desde la interfaz web |
| Interfaz Web | Punto de control para el Operario o Gerente |

Debido a su potencia y requerimientos eléctricos, cada ventilador se activa mediante un contactor trifásico, el cual es controlado desde la Raspberry Pi mediante un módulo de relé de 5 V. Toda la comunicación entre el sistema web y los galpones se realiza mediante MQTT sobre AWS IoT Core, con comunicación segura basada en certificados.



Flujo de funcionamiento

1. El Operario accede a la interfaz web y selecciona un galpón y sección.
2. Presiona el botón “Encender ventilador sección 2”.
3. El sistema publica un mensaje MQTT con el siguiente contenido: {"seccion": 2, "accion": "ON"} en el topic: galpon1/ventiladores/commands.
4. La Raspberry Pi del galpón 1 está suscrita a ese topic y ejecuta un script que:
   * Identifica el número de sección.
   * Activa el GPIO correspondiente.
5. El GPIO energiza el módulo de relé de 5 V, que cierra el circuito de la bobina del contactor trifásico.
6. El contactor se cierra, permitiendo el paso de las tres fases (L1, L2, L3) al motor del ventilador.
7. El ventilador se enciende.
8. Para apagar, se publica el mensaje {"accion": "OFF"} y el proceso se revierte.
9. La Pi puede publicar una confirmación del estado en galpon1/ventiladores/status.

## XI.9. Estructura de Comunicación MQTT: Organización de Topics por Funcionalidad y Galpón

La arquitectura del sistema está basada en el protocolo MQTT (Message Queuing Telemetry Transport), ideal para aplicaciones IoT por su ligereza, eficiencia y modelo de comunicación desacoplado. En este modelo, los mensajes son enviados a través de topics o “canales”, a los cuales los dispositivos pueden suscribirse o publicar datos.

Para mantener el sistema organizado, escalable y seguro, se ha definido una estructura de topics jerárquica, con nombres que identifican claramente el galpón, el dispositivo o subsistema, y el tipo de operación.

| Topic | Descripción | Tipo de mensaje |
| --- | --- | --- |
| galponX/sensores/lecturas | Lecturas de temperatura y humedad por sección | Publicación |
| galponX/sensores/status | Confirmación del funcionamiento del sensor | Publicación |
| galponX/luces/commands | Comandos para encender o apagar luces | Suscripción (acción) |
| galponX/luces/status | Estado actual de las luces (encendidas/apagadas) | Publicación |
| galponX/aspersores/commands | Activación de aspersores por sección | Suscripción (acción) |
| galponX/aspersores/status | Estado de electroválvulas (activadas/desactivadas) | Publicación |
| galponX/cortinas/commands | Comando para subir, bajar o mover parcialmente las cortinas | Suscripción (acción) |
| galponX/cortinas/status | Confirmación de posición estimada de cortina | Publicación |
| galponX/ventiladores/commands | Comando para encender/apagar ventiladores por sección | Suscripción (acción) |
| galponX/ventiladores/status | Confirmación del estado del ventilador | Publicación |
| galponX/reportes/diario | Envío automático del resumen diario del galpón | Publicación |
| galponX/alertas/detalles | Notificación de condiciones ambientales fuera de rango | Publicación |
| galponX/alertas/comandos | Respuesta del usuario a una alerta (resolver, justificar) | Suscripción |

## XI.10. Seguridad en el Sistema IoT: Comunicación, Autenticación

La comunicación entre los distintos componentes del sistema se realiza sobre el protocolo MQTT, operando sobre conexiones cifradas mediante TLS (Transport Layer Security). Este protocolo garantiza que toda la información enviada desde y hacia las Raspberry Pi esté protegida contra intercepciones o modificaciones durante el tránsito.

Además, cada Raspberry Pi cuenta con un certificado digital X.509 único, emitido por AWS IoT Core, que le permite autenticarse ante el broker. Esto significa que:

* Solo dispositivos previamente registrados pueden conectarse al sistema.
* Cada dispositivo tiene acceso restringido a un conjunto específico de topics (por ejemplo, solo los de su galpón).
* Si un dispositivo se pierde o se compromete, se puede revocar su certificado de forma inmediata.

Este modelo de autenticación robusta y control de permisos por topic permite una operación segura incluso en contextos con múltiples dispositivos distribuidos.

## XI.11. Diccionario

### TLS

TLS (Transport Layer Security) es un protocolo de cifrado y autenticación que protege la comunicación entre dos extremos en una red. Es el sucesor de SSL (Secure Sockets Layer) y se emplea habitualmente en conexiones HTTP seguras (HTTPS), correo electrónico, mensajería, y en este caso, MQTT seguro.

1. Cifrado: TLS cifra los datos mientras viajan por la red, de modo que si alguien intercepta los paquetes, no podrá leer la información sin la clave de cifrado.
2. Autenticación: Mediante certificados digitales, cada extremo (por ejemplo, la Raspberry Pi y el broker MQTT) puede verificar la identidad del otro y asegurarse de que no se comunica con un impostor.
3. Integridad de los datos: TLS incluye sumas de verificación (MAC o HMAC) para garantizar que los mensajes no sean alterados de forma malintencionada en tránsito.

En el contexto de AWS IoT Core (o cualquier broker MQTT seguro), TLS se usa para:

* Cifrar el tráfico MQTT (en lugar de usar puertos no seguros como 1883, se usa 8883 con TLS).
* Verificar con certificados X.509 que la Raspberry Pi está autorizada a comunicarse con el broker, y viceversa.

Así, TLS protege tanto la confidencialidad (datos encriptados) como la integridad y autenticidad (certificados) de la comunicación.

### GPIO

Un GPIO es básicamente un interruptor electrónico controlado por software. Desde tu código (por ejemplo, en Python), podés decir:

GPIO.output(17, GPIO.HIGH) # Activa el pin 17 (3.3V)

Y eso genera una señal de voltaje (3.3 V) en ese pin, que podés usar para encender un relé, mover un servo, o hacer que algo suceda.

También se pueden leer entradas:

if GPIO.input(27) == GPIO.HIGH:

print("Botón presionado")

### Certificado X.509

Un certificado X.509 es un archivo digital que sirve para identificar y autenticar de forma segura a un dispositivo o usuario en una red. Es el mismo tipo de certificado que usan los sitios web con HTTPS (candadito verde), pero en este caso lo usamos para dispositivos como tu Raspberry Pi.

Cuando una Raspberry Pi quiere conectarse al AWS IoT Core para enviar o recibir mensajes MQTT, debe probar su identidad. Para eso:

* Se le asigna un certificado digital X.509 único, generado desde AWS IoT.
* Este certificado contiene una clave pública y privada, y está firmado por AWS, lo que garantiza que es válido.
* Al conectarse, la Pi usa ese certificado para que AWS IoT verifique que es un dispositivo autorizado.

## Conclusión del capítulo

La solución desarrollada representa una base sólida para la transformación digital de granjas avícolas familiares, con un enfoque escalable y adaptable a nuevas tecnologías. Las decisiones tomadas permiten lograr un sistema funcional desde el primer día, sin sacrificar posibilidades de evolución hacia una plataforma más inteligente, automatizada y resiliente en el futuro.

# Bibliografía

<https://www.argentina.gob.ar/senasa/programas-sanitarios/cadenaanimal/aves>

<https://www.infocampo.com.ar/no-hay-crisis-que-lo-impida-argentina-sigue-al-tope-en-el-ranking-mundial-de-consumo-de-carnes/>

<https://www.argentina.gob.ar/noticias/entre-rios-fiscalizacion-de-granjas-avicolas-para-fortalecer-la-sanidad-y-el-bienestar>

<https://avinews.com/estres-termico-y-su-efecto-sobre-la-salud-hepatica-en-pollos/>

<https://certifiedhumanelatino.org/bienestar-animal-y-calidad-de-la-carne-sepa-como-el-manejo-interfiere-en-la-produccion/>

<https://www.argentina.gob.ar/senasa/programas-sanitarios/bienestar-animal/que-es>

Gallard, E.A., Menichelli, M., Fernández, R.J., Sanz, S.P., Di Masso, R.J., & Revidatti, F.A. 2018. Efectos de la densidad de alojamiento y la zona del galpón sobre indicadores de bienestar en pollos parrilleros criados en lotes mixtos. Recuperado de [Efectos de la densidad de alojamiento y la zona del galpón sobre indicadores de bienestar en pollos parrilleros criados en lotes mixtos](https://www.fcv.unl.edu.ar/investigacion/wp-content/uploads/sites/7/2018/11/PA_GALLARD_EFECTOS.pdf)

<https://www.argentina.gob.ar/noticias/recomendaciones-para-la-prevencion-del-estres-termico-por-calor-en-aves-de-granja>

<https://www.phosphea.com/es/estres-termico-en-la-produccion-avicola-riesgos-y-soluciones-tecnicas/>

<https://aws.amazon.com/es/what-is/iot/>

<https://avinews.com/como-se-vincula-una-granja-avicola-y-la-internet-de-las-cosas/#:~:text=La%20Internet%20de%20las%20Cosas%20(IOT)%20es%20una%20tecnolog%C3%ADa%20innovadora,y%20el%20crecimiento%20del%20pollo>.

<https://internetdelascosas.xyz/articulo.php?id=597&titulo=IoT-en-la-agricultura-inteligente-modernizar-la-avicultura-con-dispositivos-conectados>

<https://www.portalveterinaria.com/avicultura/articulos/33426/manejo-inteligente-en-avicultura-sensores-big-data-e-internet-de-las-cosas.html>

<https://blog.climatefieldview.com.ar/big-data-en-la-agricultura>

<https://avinews.com/analisis-de-datos-en-una-granja-avicola-con-la-central-de-regulacion-copilot-control-webisense/>

<https://www.catedraavicola.com.ar/perspectivas-avicolas-sobre-bioestadistica-y-big-data/>

🧱 Domain-Driven Design / Arquitectura Modular

* Evans, E. (2004). *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Addison-Wesley.
* Khononov, V. (2021). *Learning Domain-Driven Design: Aligning Software Architecture and Business Strategy*. O’Reilly Media.
* Hombergs, T. (2020). *Modularization in Spring Boot Applications*. Recuperado de https://reflectoring.io/spring-boot-modules

☁️ Cloud y Seguridad

* Amazon Web Services. (s.f.). *AWS IoT Core Developer Guide*. Recuperado de<https://docs.aws.amazon.com/iot/latest/developerguide/>
* Pahl, C., & Jamshidi, P. (2016). *Microservices: A Systematic Mapping Study*. In *Proceedings of the 6th International Conference on Cloud Computing and Services Science (CLOSER)*, 137–146.

⚠️ Análisis de riesgos

* Evans, R., & Munier, N. (2011). *Gestión de riesgos en proyectos: Métodos y herramientas*. Ediciones Díaz de Santos.
* Aviagen. (2009). *Manejo del Ambiente en el Galpón de Pollo de Engorde*. [https://es.aviagen.com](https://es.aviagen.com/)
* AviNews. (2023). *Mecanismos de enfriamiento no evaporativo*. [https://avinews.com](https://avinews.com/)
* Daimob. (2023). *Monitoreo IoT de ambientes en avicultura*. [https://daimob.co](https://daimob.co/)
* FAO. (2013). *Revisión del desarrollo avícola mundial*. [https://www.fao.org](https://www.fao.org/)
* Fundación Mediterránea. (2024). *Informe IERAL sobre consumo de carnes en Argentina*. [https://www.infocampo.com.ar](https://www.infocampo.com.ar/)
* OMSA. (2023). *Bienestar animal en la producción avícola*. [https://www.woah.org](https://www.woah.org/)
* Phosphea. (2020). *Estrés térmico: riesgos y soluciones técnicas*. [https://www.phosphea.com](https://www.phosphea.com/)
* Project Management Institute. (2021). *PMBOK Guide* (7ma ed.). Project Management Institute.
* Snoderly, J., & Faisandier, A. (2023). *Guide to the Systems Engineering Body of Knowledge (SEBoK), Version 2.8*. [www.sebokwiki.org](http://www.sebokwiki.org/)
* FONTAGRO. (2020). *Memoria AgTech: Agricultura 4.0*. [https://www.fontagro.org](https://www.fontagro.org/)
* Speller, D. (2023). *Data-Driven Poultry Farming*. Applied Poultry Group.
* Senasa. (2023). *Programa Nacional de Sanidad Aviar*.<https://www.argentina.gob.ar/senasa>