



**UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS
FACULTAD DE INGENIERÍA
PROGRAMA ACADÉMICO DE INGENIERÍA DE SOFTWARE
11ASI0732-2520-7491 - DISEÑO DE EXPERIMENTOS DE INGENIERÍA DE SOFTWARE**

**VacApp
TB2**

DOCENTE:

Ivan Robles Fernandez

STARTUP:

Vacow Team

INTEGRANTES DEL EQUIPO:

Saldaña Liberato, Rodrigo

Código: U202215623

Macavilca Quispe, Ian

Código: U202121325

Rojas Velasquez, Maycol

Código: U202219984

Espinoza Cueva, Stephano

Código: U202218590

Paucar Meneses, Jeremy

Código: U201919449

Septiembre 2025

Registro de Versiones del Informe

Versión	Fecha	Autor	Descripción de modificación
TB1	20/09/25	Saldana, Rodrigo Liberato (U202215623), Macavilca Quispe, Ian (U202121325), Rojas Velasquez, Maycol (U202219984), Espinoza Cueva, Stephano (U202218590), Paucar Meneses, Jeremy (U201919449)	Capítulo I: Introducción, Capítulo II: Requirements Elicitation & Analysis, Capítulo III: Requirements Specification, Capítulo IV: Product Design y Capítulo V: Product Implementation, Validation & Deployment
TP	06/10/25	Saldana, Rodrigo Liberato (U202215623), Macavilca Quispe, Ian (U202121325), Rojas Velasquez, Maycol (U202219984), Espinoza Cueva, Stephano (U202218590), Paucar Meneses, Jeremy (U201919449)	Capítulo VI: Product Verification & Validation, Capítulo VI: Product Verification & Validation, Capítulo VI: DevOps Practices

Versión	Fecha	Autor	Descripción de modificación
TB2	12/10/2025	Saldana, Rodrigo Liberato (U202215623), Macavilca Quispe, Ian (U202121325), Rojas Velasquez, Maycol Espinoza Cueva, Stephano (U202218590), Paucar Meneses, Jeremy (U201919449)	Capítulo VI: Product Verification & Validation (6.1 Testing Suites & Validation; 6.2 Static testing & Verification: 6.2.1 Static Code Analysis — 6.2.1.1 Coding standard & Code conventions; 6.2.1.2 Code Quality & Code Security; 6.2.2 Reviews; 6.3 Validation Interviews — 6.3.1 Diseño de entrevistas; 6.3.2 Registro de Entrevistas; 6.3.3 Evaluaciones según heurísticas; 6.4 Auditoría de Experiencias de Usuario — 6.4.1 Auditoría realizada (información del grupo auditado, cronograma, contenido) y 6.4.2 Auditoría recibida (información del grupo auditor, cronograma, contenido, resumen de modificaciones)). Además se incluyen Capítulo VII: DevOps Practices (CI, CD, Continuous Deployment, Continuous Monitoring — herramientas, pipelines y componentes) y Capítulo VIII: Experiment-Driven Development (Experiment Planning, Experiment Design y Experimentation con To-Be User Stories y Product Backlog).

Project Report Collaboration Insights

1. URL del Repositorio en GitHub

Repositorio del Informe en GitHub

<https://github.com/orgs/1ASI0732-Grupo-3/repositories>

2. Actividades de Elaboración del Informe

Actividad	Descripción
Comunicación de avances	Se realizaron breves reuniones donde se comentaban los avances de cada integrante para garantizar un trabajo en equipo sincronizado y estar al tanto de cada modificación.
Delegación de responsabilidades	Se organizaron charlas y reuniones para la delegación de tareas, promoviendo el trabajo en equipo y el avance continuo de manera asíncrona.
Recopilación de datos	Se realizó un estudio del mercado, utilizando los segmentos objetivos para extraer datos significativos mediante entrevistas. Además, se utilizaron referencias bibliográficas para investigar información relevante dentro del desarrollo del informe
Documentación	Se desarrollaron diferentes capítulos del informe, utilizando las técnicas, herramientas y metodologías requeridas.

3. Capturas de Imagen de los Analíticos de Colaboración y Commits en GitHub

Métrica	Descripción
Número de commits por autor	
Historial de cambios	

4. Participación de Todos los Miembros del Equipo

Evidencia	Descripción
Contribuciones en GitHub	Cada integrante del equipo completó sus tareas y subió sus avances al repositorio de GitHub. Ademas, cada uno colaboró revisando el trabajo de sus compañeros.
Discusiones y actividades	Se organizaron encuentros virtuales para compartir avances, resolver dudas y atender los desafíos de cada actividad.
Exposiciones del proyecto	Se llevaron a cabo sesiones para exponer el trabajo realizado antes de la fecha correspondiente y prepararnos adecuadamente para la presentación final.

Tabla de Contenidos

Capítulo I: Introducción

- 1.1. Start-up Profile
 - 1.1.1. Descripción de la Startup
 - 1.1.2. Perfiles de Integrantes del Equipo
 - 1.2. Solution Profile
 - 1.2.1. Antecedentes y Problemática
 - 1.2.2. Lean UX Process
 - 1.2.2.1. Lean UX Problem Statements
 - 1.2.2.2. Lean UX Assumptions
 - 1.2.2.3. Lean UX Hypothesis Statements
 - 1.2.2.4. Lean UX Canvas
 - 1.3. Segmentos Objetivo
-

Capítulo II: Requirements Elicitation & Analysis

- 2.1. Competidores
 - 2.1.1. Análisis Competitivo
 - 2.1.2. Estrategias y Tácticas frente a Competidores
- 2.2. Entrevistas
 - 2.2.1. Diseño de Entrevistas
 - 2.2.2. Registro de Entrevistas
 - 2.2.3. Análisis de Entrevistas
- 2.3. Needfinding
 - 2.3.1. User Personas

- 2.3.2. User Task Matrix
 - 2.3.3. User Journey Mapping
 - 2.3.4. Empathy Mapping
 - 2.3.5. As-is Scenario Mapping
 - 2.4. Ubiquitous Language
-

Capítulo III: Requirements Specification

- 3.1. To-Be Scenario Mapping
 - 3.2. User Stories
 - 3.3. Impact Mapping
 - 3.4. Product Backlog
-

Capítulo IV: Product Design

- 4.1. Style Guidelines
 - 4.1.1. General Style Guidelines
 - 4.1.2. Web Style Guidelines
 - 4.1.3. Mobile Style Guidelines
 - 4.1.3.1. iOS Mobile Style Guidelines
 - 4.1.3.2. Android Mobile Style Guidelines
- 4.2. Information Architecture
 - 4.2.1. Organization Systems
 - 4.2.2. Labeling Systems
 - 4.2.3. SEO Tags and Meta Tags
 - 4.2.4. Searching Systems
 - 4.2.5. Navigation Systems
- 4.3. Landing Page UI Design
 - 4.3.1. Landing Page Wireframe
 - 4.3.2. Landing Page Mock-up
- 4.4. Mobile Applications UX/UI Design
 - 4.4.1. Wireframes
 - 4.4.2. Wireflow Diagrams
 - 4.4.3. Mock-ups
 - 4.4.4. User Flow Diagrams
- 4.5. Mobile Applications Prototyping
 - 4.5.1. Android Prototyping
 - 4.5.2. iOS Prototyping
- 4.6. Web Applications UX/UI Design
 - 4.6.1. Wireframes
 - 4.6.2. Wireflow Diagrams
 - 4.6.3. Mock-ups
 - 4.6.4. User Flow Diagrams
- 4.7. Web Applications Prototyping
- 4.8. Domain-Driven Software Architecture

- 4.8.1. Context Diagram
 - 4.8.2. Container Diagrams
 - 4.8.3. Component Diagrams
 - 4.9. Software Object-Oriented Design
 - 4.9.1. Class Diagrams
 - 4.9.2. Class Dictionary
 - 4.10. Database Design
 - 4.10.1. Relational/Non-Relational Database Diagram
-

Capítulo V: Product Implementation

- 5.1. Software Configuration Management
 - 5.1.1. Development Environment Configuration
 - 5.1.2. Source Code Management
 - 5.1.3. Code Style Guide & Conventions
 - 5.1.4. Deployment Configuration
 - 5.2. Product Implementation & Deployment
 - 5.2.1. Sprint Backlogs
 - 5.2.2. Landing Page Evidence
 - 5.2.3. Frontend Web Application Evidence
 - 5.2.4. Acuerdo de Servicio SaaS
 - 5.2.5. Native Mobile Application Evidence
 - 5.2.6. RESTful API / Serverless Backend Evidence
 - 5.2.7. API Documentation
 - 5.2.8. Team Collaboration Insights
 - 5.3. Video About the Product
-

Capítulo VI: Product Verification & Validation

- 6.1. Testing Suites & Validation
 - 6.1.1. Core Entities Unit Tests
 - 6.1.2. Core Integration Tests
 - 6.1.3. Core Behavior-Driven Development
 - 6.1.4. Core System Tests
- 6.2. Static testing & Verification
 - 6.2.1. Static Code Analysis
 - 6.2.1.1. Coding standard & Code conventions
 - 6.2.1.2. Code Quality & Code Security
 - 6.2.2. Reviews
- 6.3. Validation Interviews
 - 6.3.1. Diseño de entrevistas
 - 6.3.2. Registro de Entrevistas
 - 6.3.3. Evaluaciones Según heurísticas
- 6.4. Auditoría de Experiencias de Usuario
 - 6.4.1. Auditoría realizada

- 6.4.1.1. Información del grupo auditado
- 6.4.1.2. Cronograma de auditoría realizada
- 6.4.1.3. Contenido de auditoría realizada
- 6.4.2. Auditoría recibida
 - 6.4.2.1. Información del grupo auditor
 - 6.4.2.2. Cronograma de auditoría recibida
 - 6.4.2.3. Contenido de auditoría recibida
 - 6.4.2.4. Resumen de modificaciones para subsanar hallazgos

Capítulo VII: DevOps Practices

- 7.1. Continuous Integration
 - 7.1.1. Tools and Practices
 - 7.1.2. Build & Test Suite Pipeline Components
- 7.2. Continuous Delivery
 - 7.2.1. Tools and Practices
 - 7.2.2. Stages Deployment Pipeline Components
- 7.3. Continuous Deployment
 - 7.3.1. Tools and Practices
 - 7.3.2. Production Deployment Pipeline Components
- 7.4. Continuous Monitoring
 - 7.4.1. Tools and Practices
 - 7.4.2. Monitoring Pipeline Components
 - 7.4.3. Alerting Pipeline Components
 - 7.4.4. Notification Pipeline Components

Capítulo VIII: Experiment-Driven Development

- 8.1. Experiment Planning
 - 8.1.1. As-Is Summary
 - 8.1.2. Raw Material: Assumptions, Knowledge Gaps, Ideas, Claims
 - 8.1.3. Experiment-Ready Questions
 - 8.1.4. Question Backlog
 - 8.1.5. Experiment Cards
- 8.2. Experiment Design
 - 8.2.1. Hypotheses
 - 8.2.2. Domain Business Metrics
 - 8.2.3. Measures
 - 8.2.4. Conditions
 - 8.2.5. Scale Calculations and Decisions
 - 8.2.6. Methods Selection
 - 8.2.7. Data Analytics: Goals, KPIs and Metrics Selection
 - 8.2.8. Web and Mobile Tracking Plan
- 8.3. Experimentation

- [8.3.1. To-Be User Stories](#)
- [8.3.2. To-Be Product Backlog](#)
- [Conclusiones](#)
- [Bibliografía](#)
- [Anexos](#)

STUDENT OUTCOME

El curso contribuye al cumplimiento del Student Outcome ABET:

ABET – EAC - Student Outcome 4

Criterio: La capacidad de reconocer responsabilidades éticas y profesionales en situaciones de ingeniería y hacer juicios informados, que deben considerar el impacto de las soluciones de ingeniería en contextos globales, económicos, ambientales y sociales.

Criterio específico	Acciones realizadas	Conclusiones
4.c.1 Reconoce responsabilidad ética y profesional en situaciones de ingeniería de software	<p>1. Rojas Velasquez, Maycol</p> <p>Jhordan</p> <p><i>TB1:</i> Aplicó buenas prácticas en la elaboración de la documentación técnica, asegurando transparencia en los procesos y citando correctamente las fuentes de información utilizadas.</p> <p><i>TP:</i> Aplicó prácticas de ingeniería de software para las pruebas unitarias y de sistema, garantizando trazabilidad y control de calidad en el código.</p> <p><i>TB2:</i> Asumió responsabilidad en la verificación de artefactos UX y auditorías externas e internas (cap. 6.4), liderando la integración ética de resultados dentro del pipeline DevOps (CI / CD / Deployment), asegurando que el producto se valide antes de ser liberado a ambientes superiores.</p> <p>2. Espinoza Cueva, Stephano Jose</p> <p><i>TB1:</i> Me involucré de lleno en definir qué era más urgente y cómo íbamos a entregar el producto, participando</p>	<p>TB1: La ética profesional fue la base en todas las etapas del proyecto, asegurando transparencia en la documentación, honestidad en la gestión del código y respeto por las buenas prácticas de desarrollo.</p> <p>TP: Las pruebas unitarias y de sistema consolidaron la responsabilidad profesional del equipo al validar que cada componente funcionara correctamente bajo principios de calidad y fiabilidad.</p> <p>TB2: La responsabilidad ética se fortaleció al integrar auditorías UX, pipelines CI/CD, análisis estático y diseño experimental; garantizando que las decisiones se fundamenten en evidencia técnica real, mediciones verificables y datos no manipulados.</p>

Criterio específico	Acciones realizadas	Conclusiones
	<p>en la priorización de requisitos y la planificación de entregables.</p> <p><i>TP:</i></p> <p>Colaboré en la implementación de prácticas de despliegue continuo, priorizando la calidad del software y el cumplimiento de estándares éticos en las entregas del producto.</p> <p><i>TB2:</i></p> <p>Fui responsable de asegurar la coherencia ética entre métricas, KPIs y trazabilidad analítica (cap. 8.2.7 / 8.2.8), garantizando que no se manipulen resultados ni se tomen decisiones sin datos reales siendo fieles a validaciones provenientes de usuarios entrevistados y auditores.</p>	
	<p>3. Saldana Liberato, Rodrigo</p> <p><i>TB1:</i></p> <p>Elaboré documentación clara sobre las historias de usuario y realicé la configuración del entorno de desarrollo y despliegue del producto.</p> <p><i>TP:</i></p> <p>Elaboré la documentación y ejecución de las pruebas de integración, asegurando el correcto funcionamiento del sistema en conjunto.</p> <p><i>TB2:</i></p> <p>Asumí responsabilidad ética sobre la transparencia del análisis estático de código y su impacto en seguridad / convenciones (cap. 6.2), asegurando que todos los hallazgos se documenten y que no se oculte ninguno durante el proceso de verificación.</p>	
	<p>4. Macavilca Quispe, Ian</p> <p><i>TB1:</i></p> <p>Elaboré el diseño UI/UX del web</p>	

Criterio específico	Acciones realizadas	Conclusiones
	<p>app, así como el web prototyping.</p> <p><i>TP:</i></p> <p>Desarrollé el flujo de Integración Continua dentro del pipeline de DevOps.</p> <p><i>TB2:</i></p> <p>Diseñé la planificación experimental (cap. 8.1), documentando suposiciones y asegurando que los experimentos no alteren datos reales ni afecten a usuarios sin consentimiento previo.</p> <p>5. Paucar Meneses, Jeremy</p> <p><i>TB1:</i></p> <p>Elaboré la documentación sobre los competidores, así como los user personas.</p> <p><i>TP:</i></p> <p>Implementé BDD.</p> <p><i>TB2:</i></p> <p>Supervisé la formulación de hipótesis (cap. 8.2.1), Domain Business Metrics (cap. 8.2.2) y cálculos de escala experimentales (cap. 8.2.5) garantizando rigurosidad en la toma de decisiones basada en datos reales y no supuestos manipulados.</p>	
4.c.2 Emite juicios informados considerando el impacto de las soluciones de ingeniería de software en contextos globales, económicos, ambientales y sociales	<p>1. Rojas Velasquez, Maycol Jhordan</p> <p><i>TB1:</i></p> <p>Analicé cómo la solución propuesta (VacApp) impacta en la eficiencia de la ganadería, promoviendo prácticas sostenibles que favorecen al sector económico y respetan el medio ambiente.</p> <p><i>TP:</i></p> <p>Aplicué juicios técnicos para las pruebas de sistema, evaluando su</p>	<p>TB1: El desarrollo de VacApp promovió una visión integral que combina innovación tecnológica con sostenibilidad, impulsando prácticas más responsables en la gestión ganadera peruana.</p> <p>TP: La implementación de pruebas, despliegues continuos y evaluaciones automatizadas permitió emitir juicios técnicos más precisos, asegurando que el sistema mantuviera su impacto positivo en el entorno productivo y social.</p> <p>TB2: La consideración de auditorías,</p>

Criterio específico	Acciones realizadas	Conclusiones
	<p>impacto y funcionamiento dentro del contexto productivo real.</p> <p><i>TB2:</i></p> <p>Generé juicios informados a partir de las auditorías UX realizadas y recibidas (cap. 6.4), integrando retroalimentación externa en la toma de decisiones del pipeline CI/CD, para asegurar que cada despliegue considere impacto en usuarios reales, stakeholders y la operación ganadera digital a escala.</p>	<p>CI/CD, métricas analíticas y diseño experimental permitió emitir juicios basados en evidencia concreta, asegurando que cada decisión técnica tenga impacto medible sobre factores económicos, globales y sociales, consolidando una visión profesional madura sobre el rol del software dentro del sector ganadero real.</p>
	<p>2. Espinoza Cueva, Stephano Jose</p> <p><i>TB1:</i></p> <p>Al diseñar las propuestas, mantuve una visión global, considerando cómo nuestro producto afectaría a distintos usuarios y escenarios.</p> <p><i>TP:</i></p> <p>En la implementación de <i>Continuous Deployment</i>, prioricé la estabilidad del entorno productivo y la satisfacción del usuario final.</p> <p><i>TB2:</i></p> <p>Apliqué juicios basados en métricas, KPIs y analítica (cap. 8.2.7 / 8.2.8) asegurando que las decisiones para pivotear o escalar funciones correspondieran a datos reales y no a percepciones subjetivas, considerando el costo real de intervención, adopción y resultado para la industria ganadera peruana.</p>	
	<p>3. Saldana Liberato, Rodrigo</p> <p><i>TB1:</i></p> <p>Documenté las funcionalidades del producto y configuré su despliegue.</p> <p><i>TP:</i></p> <p>Durante la implementación de las pruebas de integración, profundicé en la lógica de negocio.</p> <p><i>TB2:</i></p>	

Criterio específico	Acciones realizadas	Conclusiones
	<p>Utilicé resultados de la verificación estática del código (cap. 6.2) para emitir juicios sobre seguridad, mantenibilidad e impacto futuro del producto; considerando implicancias técnicas, económicas (costos de deuda técnica) y sociales (riesgos de exposición o mal funcionamiento).</p> <p>4. Macavilca Quispe, Ian</p> <p><i>TB1:</i> Consideré la funcionalidad y practicidad.</p> <p><i>TP:</i> Apliqué principios de eficiencia y sostenibilidad.</p> <p><i>TB2:</i> Apliqué juicios informados durante la definición del <i>Experiment Planning</i> (cap. 8.1), sopesando impacto versus beneficio para decidir qué hipótesis merecen experimentación, evitando costo innecesario, sesgo metodológico y riesgo de generar daño a la data o usuarios objetivo.</p>	
	<p>5. Paucar Meneses, Jeremy</p> <p><i>TB1:</i> Realicé los diferentes tipos de <i>Needfinding</i>.</p> <p><i>TP:</i> Analicé los resultados de las pruebas funcionales y de comportamiento.</p> <p><i>TB2:</i> Apliqué análisis crítico al formular las hipótesis, Domain Business Metrics y decisiones de escalabilidad (cap. 8.2.1 – 8.2.6), considerando impacto económico para el productor ganadero, sostenibilidad en el largo plazo y</p>	

Criterio específico	Acciones realizadas	Conclusiones
	valor social de introducir mejoras en un sector con alto nivel de informalidad tecnológica.	

Capítulo I: Introducción

1.1. Start-up Profile

1.1.1. Descripción de la Startup

Nombre del Startup: VacApp

VacApp es una moderna plataforma, disponible para dispositivos móviles y web, creada para ayudar a los ganaderos a gestionar su ganado de forma eficiente y sostenible. La solución cuenta con un conjunto completo de herramientas que les permiten llevar un control detallado de la salud, nutrición y reproducción de sus animales. Esto les facilita tomar mejores decisiones en su granja.

Con VacApp, los ganaderos pueden optimizar sus operaciones y aumentar sus ganancias, mientras fomentan prácticas más responsables y éticas. De esta manera, contribuyen al bienestar de los animales y al crecimiento sostenible de la industria ganadera.

1.1.2. Perfiles de integrantes del equipo

Integrante	Conocimientos técnicos / Habilidades
Ian Macavilca Quispe U202121325	 <p>Programación en C++, Python, JavaScript, HTML y CSS. Aprendo rápido y sé adaptarme a los retos que se presentan a lo largo del desarrollo de proyectos.</p>
Rodrigo Liberato Saldaña U202215623	 <p>Estudiante de Ingeniería de Software con interés en Ciencia de Datos, Ciberseguridad y desarrollo web en .NET, Spring Boot, etc. Me comprometo a apoyar activamente al grupo y asumir el rol de líder para encaminar al equipo hacia el cumplimiento de sus metas.</p>

Integrante	Conocimientos técnicos / Habilidades
Maycol Jhordan Rojas Velasquez U202219984	<p>Elegí la carrera de Ingeniería de Software debido a mi gusto por la innovación y la implementación de la tecnología en cualquier rubro social, de una manera creativa y en todos los aspectos. Me considero una persona creativa, en busca de ideas y estrategias con mente nueva. También me gusta escuchar ideas de mi equipo, dar propuestas de mejora, evaluar ventajas y desventajas.</p>  <p>Tengo conocimientos de programación en C++, HTML, Python, Angular, Backend en Java, y domino Flutter. También cuento con experiencia en LangChain aplicado con LLM y RAG.</p> <p>Tengo un enfoque responsable y dedicado, acompañado de un aprendizaje rápido, lo que me permite ayudar a mis compañeros en sus dudas. Por otro lado, mis hobbies son ver series, jugar, escuchar música, nadar y manejar.</p>
Stephano Espinoza Cueva U202218590	<p>Stephano Espinoza Cueva – Ingeniería de Software - Elegí la carrera de Ingeniería de Software debido a mi gusto por la innovación y la implementación de la tecnología en cualquier rubro social, de una manera creativa y en todos los aspectos. Me considero una persona creativa, en busca de ideas y estrategias con mente nueva. También me gusta escuchar ideas de mi equipo, dar propuestas de mejora, evaluar las ventajas y desventajas. Además, tengo conocimientos de programación en C++, HTML, Python, Angular, Backend en Java, y domino Flutter. También cuento con experiencia en LangChain aplicado con LLM y RAG. Tengo un enfoque responsable y dedicado, acompañado de un aprendizaje rápido, lo que me permite ayudar a mis compañeros en sus dudas. Por otro lado, mis hobbies son ver series, jugar, escuchar música, nadar y manejar.</p> 
Jeremy Paucar Meneses U201919449	<p>Tengo 23 años. En la actualidad estoy estudiando la carrera de ingeniería de software en la Universidad Peruana de Ciencias Aplicadas. Albergo conocimientos intermedios en algunos lenguajes de programación, tales como C++, JavaScript y Python. Me considero una persona responsable con disposición de apoyar al grupo, también me gustan mucho los retos y así mismo poder cumplirlos.</p> 

1.2. Solution Profile

VacApp es una aplicación hecha para ganaderos en Perú. Su propósito es ayudarles a manejar su ganado de manera más fácil y eficiente. La aplicación ofrece varias herramientas para llevar un control de la salud, alimentación y reproducción de los animales. Con esto, VacApp busca aumentar la productividad de los ganaderos, al mismo tiempo que fomenta prácticas ganaderas responsables y sostenibles, promoviendo el bienestar de los animales.

Características Principales:

Nuestra plataforma, disponible para web y móviles, ofrece un conjunto de herramientas esenciales para una gestión integral y eficiente.

- Monitoreo detallado: Permite llevar un seguimiento minucioso de elementos clave, asegurando que se les dé la atención necesaria en el momento justo.
- Gestión de recursos: Facilita la planificación y el control de los recursos para garantizar un uso óptimo y sostenible.
- Seguimiento de procesos: Ayuda a monitorizar ciclos y fases importantes, optimizando los resultados y la calidad.
- Acceso a servicios profesionales: Proporciona un sistema para gestionar citas o consultas, mejorando el acceso a especialistas y evitando contratiempos por falta de atención.
- Automatización de tareas: Simplifica la administración de las operaciones diarias, lo que te ayuda a ahorrar tiempo y esfuerzo en tareas rutinarias.
- Comunidad y colaboración: Además de ser una herramienta de gestión, es un espacio para compartir ideas, buenas prácticas y estar al día con las últimas novedades del sector.

Beneficios Clave:

Usar nuestra solución te ayuda a mejorar la productividad y la sostenibilidad.

- Mejor gestión y resultados: Tendrás acceso a herramientas que te permiten hacer un seguimiento eficiente de tus operaciones, reduciendo pérdidas y mejorando los resultados.
- Optimización de recursos: La plataforma permite una mejor administración de los recursos, lo que se traduce en mayor productividad y rentabilidad.
- Acceso a tecnología sostenible: Promovemos el uso de tecnologías que fomentan prácticas más responsables y respetuosas con el entorno.
- Reducción de costos: La automatización y la mejora en la gestión operativa te ayudan a reducir gastos y aumentar la eficiencia general.
- Educación y apoyo continuo: La plataforma ofrece materiales educativos para que te mantengas siempre actualizado con las mejores prácticas y avances tecnológicos.

Tecnología y Diseño:

Nuestra solución está diseñada para ser accesible desde cualquier lugar, ya sea en el navegador o a través de sus aplicaciones móviles para iOS y Android. Está construida con tecnología avanzada para asegurar un alto rendimiento y una experiencia de uso simple. Se basa en datos para ofrecer recomendaciones y análisis personalizados que mejoran la toma de decisiones. Además, permite la integración con otros sistemas y automatiza diversas tareas administrativas.

1.2.1. Antecedentes y problemática

Gracias a la técnica de las "5W's & 2H's", se ha analizado el origen y la problemática que los ganaderos peruanos enfrentan diariamente, lo que motivó la creación de VacApp.

What? (¿Qué?)

El sector ganadero en Perú tiene dificultades importantes para gestionar su ganado, incluyendo aspectos de salud, nutrición, reproducción y productividad. Esta situación se agrava por la falta de herramientas tecnológicas. Los ganaderos, especialmente los más pequeños, no cuentan con sistemas que les permitan hacer un seguimiento detallado de sus animales ni tomar decisiones bien fundamentadas. Esto afecta su capacidad para optimizar las operaciones y aumentar sus ganancias.

When? (¿Cuándo?)

Este es un problema constante que requiere una solución actual y duradera. La necesidad de una plataforma tecnológica para gestionar la ganadería y mejorar la producción es una demanda permanente. Los ganaderos necesitan acceder a información actualizada y en tiempo real sobre la salud, alimentación, reproducción y comercialización de sus animales.

Where? (¿Dónde?)

La dificultad se concentra en las áreas rurales y productivas de Perú, donde la mayoría de los ganaderos tienen poco acceso a servicios veterinarios y tecnología. Aunque la falta de conectividad a internet en algunas zonas es un obstáculo, el crecimiento de la red móvil permite que VacApp pueda funcionar en cualquier dispositivo móvil, accesible desde cualquier ubicación rural.

Who? (¿Quién?)

Los principales afectados son los ganaderos peruanos, tanto los pequeños productores con recursos limitados como las grandes empresas que gestionan una gran cantidad de ganado y deben cumplir con estándares de bienestar animal. Veterinarios y otros profesionales del sector también se benefician de esta plataforma.

Why? (¿Por qué?)

La falta de herramientas de gestión eficientes provoca una mala nutrición del ganado, enfermedades no tratadas a tiempo, baja productividad y pérdidas económicas. Tomar decisiones con información incompleta o errónea afecta directamente el bienestar animal y la rentabilidad. VacApp busca resolver esto, ofreciendo a los ganaderos una herramienta que les ayude a optimizar sus operaciones, mejorar la trazabilidad y tomar decisiones informadas.

How? (¿Cómo?)

La solución de VacApp es una plataforma web y móvil que integra herramientas de gestión ganadera. La aplicación permite a los usuarios hacer un seguimiento de la salud, alimentación y reproducción de su ganado, generar reportes y recibir notificaciones importantes. Al usar tecnologías web y móviles de vanguardia, VacApp se adapta a las necesidades de los ganaderos de zonas rurales, permitiendo el acceso a datos en tiempo real desde cualquier dispositivo con conexión a internet.

How much? (¿Cuánto?)

Aunque el costo de desarrollar y mantener VacApp depende de la inversión en tecnología y soporte, el retorno económico para los ganaderos es muy favorable. Les permitirá ser más rentables, optimizar recursos y reducir pérdidas. La plataforma también ayudará a disminuir los gastos en atención veterinaria al proporcionar datos que previenen enfermedades y mejoran la productividad.

1.2.2. Lean UX Process

1.2.2.1. Lean UX Problem Statements

Cuando los ganaderos de menor escala aún utilizaban métodos manuales y poco eficientes para gestionar sus rebaños, nuestra plataforma VacApp (tanto la versión web como la aplicación móvil) se presentó como una solución innovadora y accesible, ideal para el trabajo en el campo. Gracias a esto, logramos atraer a los primeros ganaderos interesados en modernizar sus operaciones y aumentar su eficiencia.

Sin embargo, el panorama actual ha cambiado. El mercado ahora está saturado con nuevas empresas que ofrecen soluciones móviles similares, pero con mayores recursos financieros. Esta nueva situación ha provocado que el costo de adquirir nuevos clientes sea más alto, nuestra cuota de mercado se haya estancado y las demandas de soporte técnico hayan aumentado. Como consecuencia, nuestra rentabilidad ha disminuido y no podemos invertir lo suficiente en el desarrollo de nuevas funciones.

1.2.2.2. Lean UX Assumptions

User Assumptions:

Creemos que los ganaderos, sin importar su experiencia con la tecnología, valorarán una plataforma web y móvil que sea sencilla de usar, con una interfaz clara y accesible desde varios dispositivos.

- Los usuarios adoptarán esta tecnología si les ayuda a ahorrar tiempo y a gestionar su ganado de forma más efectiva.
- También suponemos que usarán la aplicación incluso en zonas con poca conexión a internet, siempre que funcione bien sin estar conectados.
- La confianza de los ganaderos se ganará si la plataforma les demuestra beneficios reales, como una mayor productividad y una mejor salud de sus animales.
- Como prefieren soluciones rápidas y prácticas, las funciones más importantes de la plataforma deben ser fáciles de encontrar y usar con solo unos pocos clics.

Bussiness Assumptions:

Partimos de la base de que nuestra solución digital impactará positivamente en el negocio de la ganadería.

- Si los ganaderos pueden llevar sus registros de forma digital, mejorarán la organización y el control de su inventario animal.
- Al ofrecer recordatorios automáticos y herramientas para el cuidado del ganado, los productores podrán reducir los gastos por enfermedades y mejorar el bienestar de sus animales.
- Si facilitamos la planificación de la alimentación, optimizarán sus recursos y evitarán desperdicios.

- Permitir un seguimiento preciso de la reproducción ayudará a los ganaderos a incrementar la productividad.
- Si la experiencia de uso es fluida y confiable, incluso sin conexión a internet, lograremos una mayor adopción en áreas rurales con conectividad limitada.
- Y finalmente, si aseguramos la privacidad y seguridad de sus datos, los usuarios confiarán en nuestra plataforma y se sentirán cómodos al almacenar información crucial en ella.

1.2.2.3. Lean UX Hypothesis Statements

- Creemos que los ganaderos apreciarán una aplicación con una interfaz simple, porque esto les permitirá gestionar su ganado sin necesitar mucha capacitación. Lo sabremos cuando al menos un 70% de los usuarios registre a sus animales al usar la plataforma por primera vez, sin requerir ayuda.
- Creemos que los usuarios adoptarán la plataforma si les ayuda a ahorrar tiempo en sus tareas diarias, porque prefieren dedicarse a actividades más productivas que a la administración. Lo sabremos cuando se logre una reducción del 40% en el tiempo promedio que dedican a los registros manuales.
- Creemos que los ganaderos en zonas rurales usarán la aplicación si funciona sin conexión a internet, porque en sus áreas de trabajo la conectividad es limitada. Lo sabremos cuando el 60% de los usuarios active el modo sin conexión al menos una vez a la semana.
- Creemos que los usuarios confiarán en la plataforma si garantizamos la privacidad de sus datos, porque manejan información crítica sobre su producción. Lo sabremos cuando menos del 5% de los usuarios exprese dudas sobre la seguridad en las encuestas de opinión.
- Creemos que si resaltamos las funciones principales como la salud, alimentación y reproducción, los usuarios las usarán con frecuencia para la gestión de su ganado. Lo sabremos cuando estas funciones sumen al menos el 70% del uso total dentro de la aplicación.
- Creemos que los usuarios notarán beneficios tangibles en la productividad y en la salud de su ganado, porque tendrán herramientas para tomar decisiones más informadas. Lo sabremos cuando el 60% de los usuarios reporte mejoras en el rendimiento de sus animales después de tres meses de uso.

1.2.2.4. Lean UX Canvas

1.3. Segmentos objetivo

La aplicación **VacApp** ha sido diseñada considerando las diversas realidades del sector ganadero peruano, abarcando tanto a pequeños productores independientes como a grandes empresas pecuarias. Cada segmento presenta necesidades, objetivos y desafíos específicos que nuestra plataforma busca atender con soluciones tecnológicas prácticas, accesibles, innovadoras y sostenibles. El análisis de cada grupo nos permite adaptar y mejorar constantemente nuestros servicios para ofrecerles el mayor valor posible.

1.3.1. Productores Ganaderos Independientes

Este segmento se centra en pequeños y medianos ganaderos que poseen animales como vacas, ovejas, corderos, reses y aves de corral. Son productores ubicados principalmente en zonas rurales del Perú, con

recursos limitados y un fuerte compromiso por garantizar el bienestar de sus animales.

Motivaciones:

- Mantener en óptimas condiciones la salud y el cuidado de sus animales.
- Acceder a información práctica sobre nuevas técnicas de crianza y cuidado animal.
- Garantizar productos de calidad con un enfoque ético y sostenible.
- Obtener una compensación justa en el mercado por sus productos.

Problemáticas principales:

- Escasez de recursos económicos para servicios veterinarios.
- Falta de accesibilidad a asesoría técnica y atención médica oportuna.
- La falta de atención veterinaria oportuna genera pérdidas económicas significativas por mortalidad animal.
- Los precios injustos en el mercado reducen hasta un 20 % los ingresos anuales de los productores.

Estas problemáticas limitan la competitividad, el bienestar animal y la sostenibilidad del sector.

1.3.2. Empresas Ganaderas

Este segmento corresponde a corporaciones ganaderas de gran escala que manejan operaciones con altos volúmenes de animales. Se caracterizan por buscar eficiencia en la gestión, trazabilidad de procesos y cumplimiento de estándares de sostenibilidad.

Motivaciones:

- Implementar prácticas éticas y sostenibles de producción.
- Garantizar una alimentación adecuada y atención individualizada para los animales.
- Optimizar la gestión de establos, campañas de vacunación y control sanitario.
- Mejorar su reputación y competitividad en mercados nacionales e internacionales.

Problemáticas principales:

- La ganadería contribuye de manera significativa a la deforestación en zonas amazónicas, representando entre el 49 % y el 80 % de la pérdida de bosques naturales.
- Desde 2001, Perú ha perdido más de **2 millones de hectáreas** de bosques, en parte debido a la ampliación de tierras para uso ganadero.

Estos impactos ambientales comprometen la sostenibilidad del sector y tienen consecuencias negativas en la salud pública y los ecosistemas locales.

Variables del Segmento Objetivo

Geográficas:

- País: Perú
- Ámbito: Zonas rurales y periurbanas

Demográficas:

- Género: Masculino y Femenino

- Ocupación: Productores ganaderos (independientes y corporativos)
- Estado civil: Todos los estados
- Edad: Mayores de 18 años

Psicográficas:

- Nivel socioeconómico: Todos los niveles (NSE bajo, medio y alto)
- Personalidad: Perseverantes, honestos, con fuerte compromiso hacia el trabajo, altruistas y resilientes ante la adversidad.

Capítulo II: Requirements Elicitation & Analysis

2.1. Competidores

2.1.1. Análisis competitivo

Competitive Analysis Landscaspe					
¿Porqué llevar a cabo este análisis?	¿Cómo podemos proporcionar un buen servicio entre los restaurantes y los consumidores de manera que la comunicación entre ambos sea efectiva y agradable?				
	VacApp	Control Ganadero	Agroptima	App Ganadera	
Overview	VacApp es una plataforma móvil accesible que optimiza la gestión ganadera con enfoque en sostenibilidad, bienestar animal y eficiencia, adaptada a pequeños y grandes productores.	Es una aplicación español, británico y brasileño en el sector de la ganadería.	Agroptima es un sitio web multiplataforma cofundado por la unión europea para los países de España, Francia e Inglaterra	Es una empresa fundada en Colombia que cuenta con una para la gestión de ganado.	
Perfil	La plataforma se diferencia al integrar tecnología accesible con enfoque en sostenibilidad y bienestar animal, ofreciendo soluciones prácticas para productores de todos los tamaños.		cuenta con multiplataforma cuenta con algoritmos matemáticos para una mayor gestión del ganado.	Tiene gran variedad de herramientas y gran cantidad de distribuidores	
Perfil de Marketing	Mercado Objetivo	Productores ganaderos,	Para ganaderos españoles,	Para ganaderos españoles,franceses	Para ganaderos

	tanto independientes como empresas, que buscan optimizar el cuidado del ganado y mejorar su rentabilidad.	brasileños e ingleses.	e ingleses.	colombianos
Estrategias de Marketing	Difusión en redes sociales y anuncios pagados	Estrategia de posicionamiento	Estrategia de segmentación	Estrategia
Perfil de Producto	Productos & Servicios	Aplicación móvil con herramientas de gestión de ganado, monitoreo de salud, alimentación y reproducción.		

Web app de gestión de ganado. Web app de gestión de ganado. Web app de gestión de ganado. Precios & Costos Subscripción Bajo costo Subscripción Bajo costo Subscripción Bajo costo Subscripción Bajo costo Canales de distribución (Web y/o Móvil) App App Web y móvil App Análisis SWOT Fortalezas Contamos con lo último en tecnología e implementamos lo nuevo en desarrollo para mejorar la productividad del servicio Es una app netamente para la gestión de vacas cuenta con buena personalización Es famoso por ser bueno en gestión ya que usa algoritmos matemáticos para un mejor cálculo del ganado. Su ecosistema está basado en un país de origen conoce muy bien a sus clientes y se adapta a ellos Debilidades Está en pleno desarrollo puede ser un éxito o fracaso Solo es de móvil eso limita que sea multiplataforma Solo opera en Europa y se basa en reglas ya establecidas por la unión europea Solo es una app y para la zona de Colombia por lo tanto solo está disponible en su país de origen Oportunidades Puede hacer productivos a los ganaderos y empresas de este rubro mejorando sus tomas de decisiones y eficiencia Si planeas ir a Europa es buena idea ya que ese es su público objetivo y contará con más servicios. Si planeas ir a Europa es buena idea ya que ese es su público objetivo y contará con más servicios. Si eres colombiano estarás contento con la app ya que es de uso nacional. Amenazas Competencia de otras aplicaciones móviles y el riesgo de cambios regulatorios en la industria ganadera que puedan afectar las operaciones de la plataforma. No cuenta con muchos clientes la ganadería sigue siendo a la antigua por tanto no hace falta usarla. No cuenta con muchos clientes la ganadería sigue siendo a la antigua por tanto no hace falta usarla. Falta de apoyo económico los gobiernos locales de Colombia no ven viable esta innovación puede llegar a su desaparición

2.1.2. Estrategias y tácticas frente a competidores

Para enfrentarnos a la competencia en el sector ganadero, nuestra estrategia se centrará en ofrecer una plataforma accesible, intuitiva y adaptada a las necesidades locales de los ganaderos peruanos. A diferencia de competidores como Agroptima o Control Ganadero, VacApp destacará por su simplicidad y enfoque en la sostenibilidad y el bienestar animal, facilitando la gestión de la salud, alimentación y reproducción del ganado. Utilizaremos tácticas de marketing digital segmentadas, como campañas en redes sociales y alianzas con organizaciones locales, para aumentar la visibilidad de la aplicación y educar a los usuarios sobre sus beneficios. Además, ofreceremos precios flexibles y modelos de suscripción accesibles, lo que permitirá a pequeños y grandes productores optimizar sus operaciones sin complicaciones, mientras seguimos mejorando continuamente la plataforma para adaptarnos a las necesidades cambiantes del sector ganadero.

2.2. Entrevistas

2.2.1. Diseño de entrevistas

Segmento #1: Productores Ganaderos Independientes

VacApp ha desarrollado preguntas específicas para conocer las necesidades, experiencias y expectativas de los productores ganaderos independientes. Buscamos ayudarlos a gestionar mejor sus operaciones, optimizar el cuidado de sus animales y evaluar su impacto ambiental y social. A través de una plataforma intuitiva, VacApp ofrece herramientas que mejoran la eficiencia, el control de calidad y la conexión con los consumidores, simplificando los procesos diarios del productor.

Datos Generales del Entrevistado:

- Nombre:
- Edad:
- Tiempo de experiencia en la ganadería:
- Preguntas de la Entrevista:
 - ¿Cómo decide la dieta de sus animales y qué factores considera al elegir su alimentación?
 - (¿Sigue asesoría veterinaria, experiencia personal o recomendaciones externas?)
 - ¿Qué medidas toma para garantizar la salud y el bienestar de sus animales?
 - ¿Qué aspectos considera más importantes en la gestión de la salud veterinaria de su ganado?
 - ¿Lleva algún tipo de registro sobre la salud y el crecimiento de sus animales? ¿Cómo lo hace?(¿Utiliza cuadernos, hojas de cálculo, aplicaciones móviles, etc.?)
 - ¿Cuáles son los principales desafíos que enfrenta al administrar su ganadería?
 - ¿Cómo cree que una aplicación podría ayudarle a resolver esos desafíos?
 - Si contara con una aplicación para apoyar su trabajo ganadero, ¿qué funciones le serían más útiles?
 - ¿Qué tipo de información le gustaría tener siempre disponible desde su celular o computadora?
 - ¿Cómo le gustaría registrar la alimentación y consumo de sus animales dentro de la aplicación?

- ¿Qué beneficios espera lograr al implementar una solución como VacApp en su ganadería?

Segmento #2: Empresas Ganaderas

VacApp se orienta a mejorar la eficiencia y sostenibilidad en empresas ganaderas de gran escala. Mediante entrevistas con administradores, identificamos sus necesidades y estrategias clave para una gestión efectiva. Preguntamos qué herramientas consideran esenciales y cómo esperan apoyo para mejorar el bienestar animal y la productividad. Así, VacApp adapta sus soluciones a los retos específicos de la ganadería corporativa.

Datos Generales del Entrevistado:

- Nombre:
- Edad:
- Tiempo de experiencia en la ganadería:
- Preguntas de la Entrevista:
 - ¿Cuántos animales maneja actualmente su empresa y cómo varía esa cantidad durante el año?
 - Si su empresa tuviera acceso a una plataforma digital para gestión ganadera, ¿qué funciones considera imprescindibles para mejorar la eficiencia?
 - ¿Cuáles son los mayores retos que enfrentan en la gestión ganadera a gran escala y cómo los abordan hoy en día?
 - ¿Qué tipo de información o datos son clave para la toma de decisiones en su operación ganadera?
 - ¿Qué funcionalidades le gustaría tener para facilitar la gestión del personal y la planificación de tareas?
 - ¿Qué tipo de informes o análisis considera importantes para evaluar el desempeño de su empresa?
 - ¿Cómo le gustaría interactuar con proveedores y socios comerciales a través de una plataforma como VacApp?
 - ¿Qué tan importante es que una aplicación como VacApp se adapte a los procesos actuales de su empresa?
 - ¿Qué aspectos considera que deberían ser completamente personalizables dentro de la plataforma?
 - ¿Qué mejoras espera obtener al integrar una solución como VacApp en su operación ganadera?

2.2.2. Registro de entrevistas

Ganadores independientes:

Entrevistado: Luis Raimundo

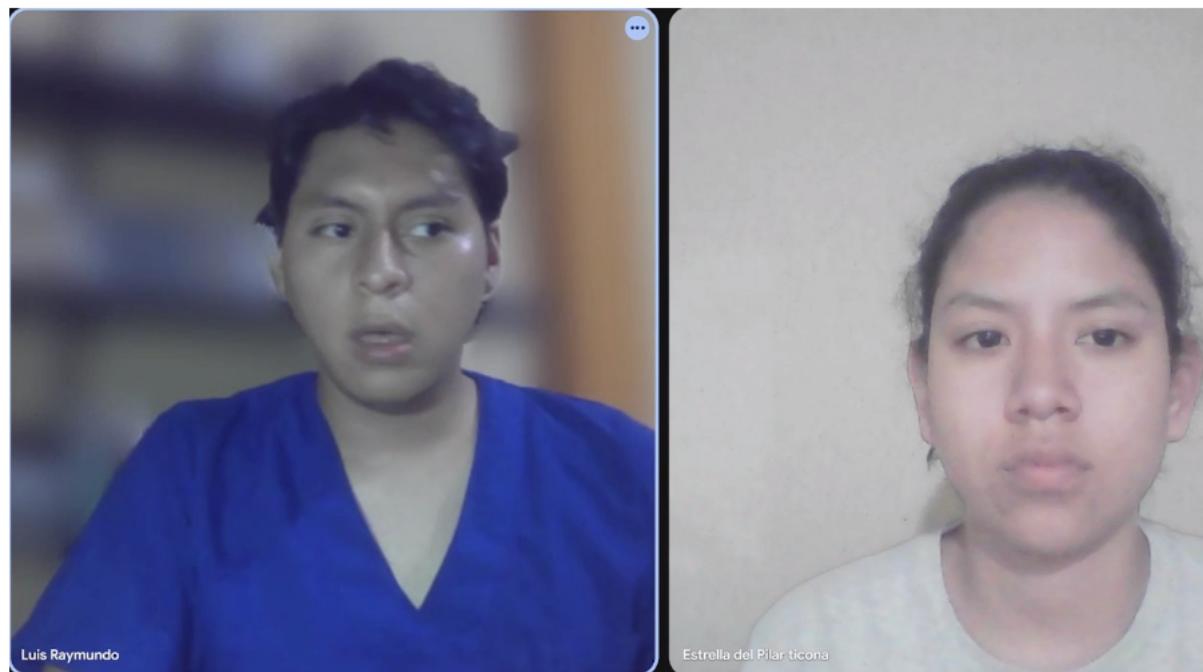
Sexo: Masculino

Edad: 25

Domicilio: Lima, Peru

Inicio de la Entrevista: 07:12

Duración de la Entrevista: 10:30



Resumen de la Entrevista:

Luis Raimundo, estudiante de cuarto año de Medicina Veterinaria con experiencia empírica y académica en la crianza de bovinos, comparte su experiencia en la gestión de una ganadería familiar en Oxapampa. Comenzó en el sector hace seis o siete años, primero de forma práctica con veterinarios y luego con enfoque más técnico por su formación académica. En cuanto al manejo, adapta la alimentación según el propósito del animal (leche o carne), implementa protocolos sanitarios como desinfección, control de parásitos y suplementación vitamínica, y ha evolucionado del registro manual a hojas de cálculo en Excel, aunque la falta de organización sigue siendo un desafío importante. Luis considera valiosa una aplicación especializada para la gestión ganadera que incluya notificaciones sobre la salud animal, registros históricos de enfermedades, seguimiento del ciclo productivo de cada lote y programación de tareas con recordatorios. También destaca la importancia de centralizar la información sobre alimentación con detalles de cantidades y tipos de alimento. Los beneficios principales que espera de una solución como "VacApp" son: mejor organización del fundo, mayor control sanitario para reducir pérdidas por enfermedades y gestión más eficiente del alimento para optimizar gastos, elementos clave para la digitalización del manejo ganadero y el fortalecimiento de la productividad rural.

Entrevistado : Jorge Torres

Edad: 50 años

Experiencia en la ganadería: 30 años

Inicio de la Entrevista: 17:43

Duración de la Entrevista: 06:19



Resumen:

El señor Jorge, ganadero desde hace ya varios años, cuenta como desde muy joven estuvo involucrado en temas ganaderos. Él habla acerca de como realiza la mayor parte de la gestión de su negocio por cuenta propia, desde el cuidado de la alimentación de sus ganados, al igual que actividades de salud o limpieza que garanticen el bienestar de los animales. También habla acerca de como lleva un registro de las actividades que realiza de manera tradicional, empleando un cuaderno de apuntes. Él considera que enfrenta varios desafíos dentro de su negocio, como por ejemplo una falta de organización y eficiencia dentro del control de los ganados, que no cuenta con ayuda profesional, y que ciertos factores externos como el clima no siempre están a su favor. Considera que una aplicación móvil podría ser de gran ayuda, y si contara con funciones que le ayudaran a superar esas dificultades que menciona, podría hacer crecer aún más su negocio y volverlo más rentable con el tiempo.

Entrevistado: Christian Matos

Edad: 25

Tiempo de experiencia en la ganadería: 5 años

Inicio de la Entrevista: 29:40

Duración de la Entrevista: 02:30



Resumen de la Entrevista:

Christian Matos, un ganadero de 25 años con cinco años de experiencia, gestiona su negocio basándose principalmente en conocimientos prácticos adquiridos con el tiempo, consultando ocasionalmente a veterinarios para decisiones sobre alimentación. Su rutina incluye revisiones diarias, limpieza del corral y seguimiento de un calendario de vacunación, demostrando su compromiso con el bienestar animal. Actualmente, registra manualmente en una libreta los nacimientos, enfermedades y tratamientos, un método funcional pero limitado que dificulta el acceso rápido a la información y puede resultar en pérdidas de datos importantes. La falta de tiempo representa su mayor desafío administrativo, impidiéndole completar todas sus tareas y afectando su capacidad organizativa. Christian ve positivamente la implementación de una aplicación móvil que funcione como recordatorio de actividades, facilite consultas rápidas de datos y mejore su eficiencia general. Desea una interfaz sencilla con acceso desde múltiples dispositivos que le permita registrar digitalmente nacimientos, vacunas y alimentación, recibir alertas para revisiones periódicas y consultar información clave como peso y tratamientos individuales de sus animales, lo que optimizaría su tiempo y fortalecería su toma de decisiones.

Segmento 2: Productores Ganadores independientes:

Entrevistado: Edgar Parry

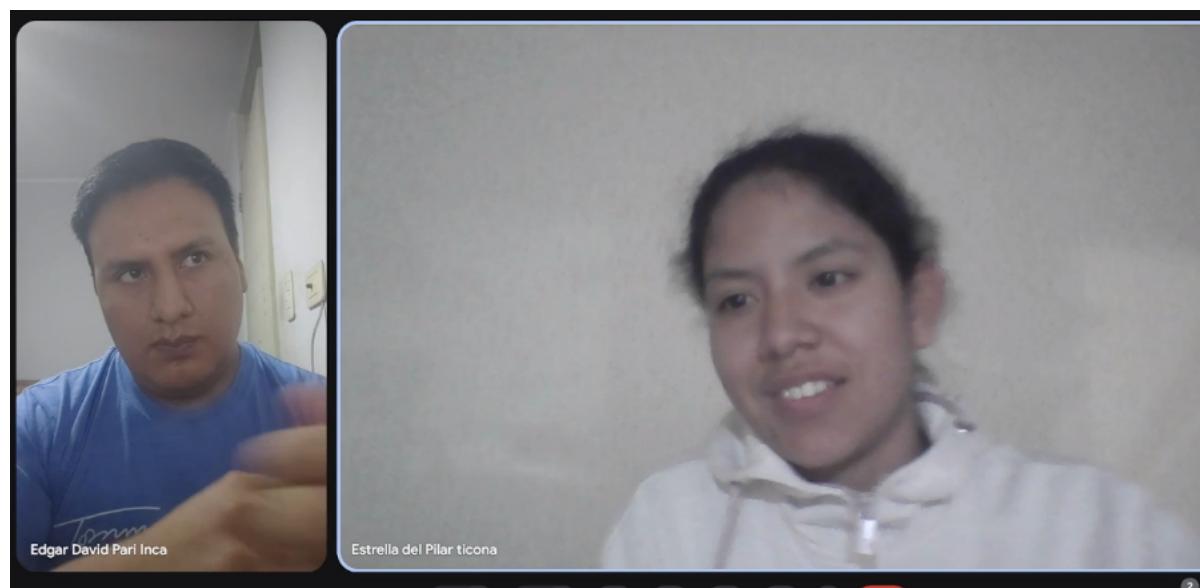
Sexo: Masculino

Edad: 29

Domicilio: Lima, Peru

Inicio de la Entrevista: 24:02

Duración de la Entrevista: 05:27



Resumen de la Entrevista: Edgar Parry, ganadero con operaciones a gran escala que maneja aproximadamente 3,000 cabezas de ganado, tiene experiencia directa, forma parte de una empresa consolidada en el sector. Edgar enfatiza la importancia de monitorear con precisión el rendimiento de cada vaca productora, especialmente en términos de producción láctea, para detectar irregularidades y tomar decisiones correctivas oportunas. Además, señala desafíos estructurales del mercado nacional, como la concentración de compra en una empresa dominante (Gloria), lo que limita las opciones de venta para los productores. Para Edgar, los indicadores productivos, reproductivos, de engorde y particularmente los sanitarios son cruciales para la toma de decisiones y la rentabilidad del negocio, aunque actualmente muchos registros se manejan en formatos físicos poco estructurados. Considera esencial que una plataforma digital como VacApp ofrezca seguimiento individualizado de cada animal (ciclo reproductivo,

producción láctea y estado sanitario), integración de la gestión de alimentos y distribución del producto final, y alta personalización adaptada a sus procesos específicos. La nutrición representa un punto crítico, por lo que valora herramientas que permitan gestionar proveedores de alimentos y monitorear el consumo por lote o animal. En síntesis, ve en la tecnología especializada una oportunidad para transformar la ganadería tradicional y adaptarla a la realidad del productor peruano.

Entrevistada: Camila Sanchez

Sexo: Femenino

Edad: 23

Inicio de la Entrevista: 32:10

Duración de la Entrevista: 05:43



Resumen de la Entrevista:

Camila Sánchez, de 23 años, cuenta con cinco años de experiencia como trabajadora en el sector ganadero. Actualmente participa en la gestión de una empresa con más de 2000 cabezas de ganado, número que puede variar según nacimientos o ventas estacionales. La gestión actual se basa mayormente en registros manuales y hojas de Excel, lo cual genera errores y retrasos.

Camila considera fundamental contar con una plataforma digital que centralice la información, mejore la coordinación del personal y permita una mejor toma de decisiones. Entre los retos más importantes que enfrentan están la gestión de recursos, el bienestar animal y la rentabilidad del negocio.

En cuanto a funcionalidades deseadas, menciona la planificación de tareas, seguimiento del desempeño del personal, generación de reportes, comunicación interna y gestión de incidentes. También valora informes sobre producción, análisis de costos, salud animal y herramientas para identificar áreas de mejora.

Para la interacción con proveedores, ve útil una plataforma que facilite la comunicación, el seguimiento de pedidos, la gestión de inventarios y la negociación. Finalmente, destaca la importancia de que la plataforma

se adapte a sus procesos y sea personalizable en aspectos como roles, alertas, datos del ganado e informes, con el fin de mejorar la eficiencia y reducir errores, siempre priorizando el bienestar animal y la rentabilidad.

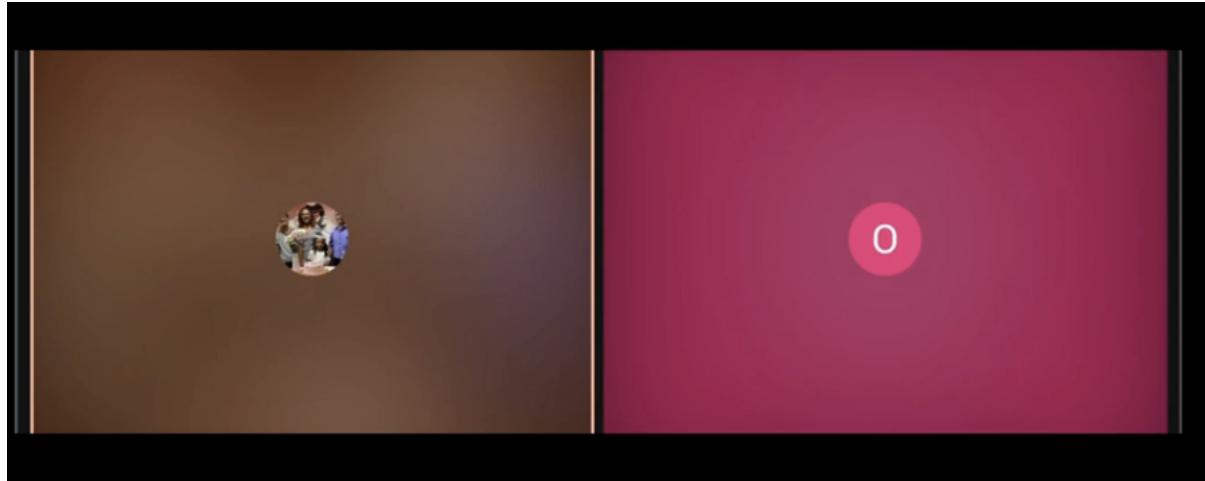
Entrevistada: Mayra Rodríguez

Sexo: Femenino

Edad: 42

Inicio de la Entrevista: 00:00

Duración de la Entrevista: 07:12



Resumen de la Entrevista:

Mayra Rodríguez, de 42 años, cuenta con 18 años de experiencia en el sector ganadero. Toma decisiones alimenticias para sus animales combinando su experiencia con asesorías veterinarias, considerando factores como la etapa de vida, salud, disponibilidad de forraje y costos. A pesar de su experiencia, valora mucho el aprendizaje continuo y las recomendaciones de otros colegas del rubro.

Para el bienestar de los animales, aplica medidas preventivas como vacunas, desparasitaciones, buenas condiciones higiénicas y monitoreo diario. Actualmente usa hojas de Excel para llevar registros, aunque reconoce que una aplicación móvil facilitaría mucho la gestión diaria por su inmediatez.

Entre los principales desafíos que enfrenta está mantener actualizados los registros individuales, algo difícil por la carga de trabajo en el campo. Una solución digital como VacApp le permitiría registrar datos en tiempo real, acceder al historial de cada animal y organizar mejor sus campañas sanitarias, reduciendo errores y ahorrando tiempo.

Le gustaría tener acceso rápido a datos sobre salud, peso, tratamientos, partos y productividad económica. También espera una interfaz sencilla para registrar la alimentación, permitiendo rutinas semanales programadas con alertas. Con VacApp, Mayra espera mejorar la gestión, evitar omisiones, tomar mejores decisiones basadas en datos, y tener todo controlado desde su celular sin depender tanto de su memoria o de registros físicos.

2.2.3. Análisis de entrevistas

Se realizaron entrevistas a dos segmentos clave del sector ganadero: Productores Ganaderos Independientes y Empresas Ganaderas. El objetivo fue identificar necesidades, desafíos, herramientas

utilizadas y expectativas sobre una posible solución digital como VacApp. A continuación, se resumen los principales hallazgos por segmento.

Segmento 1: Productores Ganaderos Independientes

Entrevistados

- Luis Raimundo
- Jorge Torres
- Christian Matos

Principales Hallazgos

Manejo de la Ganadería

- La alimentación del ganado se decide por experiencia personal o con ayuda de veterinarios.
- Hay una fuerte conexión práctica con el cuidado directo de los animales.
- Se aplican prácticas sanitarias básicas como limpieza, vacunación, desparasitación y suplementación.

Registro de Información

- Uso común de métodos manuales (libretas) o rudimentarios (Excel).
- Dificultades para mantener organización y acceso rápido a información.
- Pérdida de datos o desactualización son riesgos frecuentes.

Desafíos Identificados

- Falta de organización y sistematización.
- Limitaciones por tiempo para realizar tareas administrativas.
- Clima y factores externos afectan productividad.
- Falta de personal o asesoría profesional.

Requerimientos para VacApp

- Recordatorios y alertas para actividades críticas (vacunas, alimentación).
- Registro digital de salud, peso, nacimientos y tratamientos.
- Visualización del historial por animal o lote.
- Acceso multiplataforma (celular/computadora) con interfaz sencilla.

Beneficios Esperados

- Mejor organización y control.
- Reducción de pérdidas por enfermedades.
- Gestión eficiente del alimento y reducción de costos.
- Mayor productividad y posibilidad de expansión.

Segmento 2: Empresas Ganaderas

Entrevistados

- Edgar Parry
- Camila Sánchez
- Mayra Rodríguez

Principales Hallazgos

Gestión de Gran Escala

- Número de animales puede llegar a 3000 y fluctúa según la temporada.
- Registro de datos aún en formatos físicos o Excel, con errores frecuentes.
- Necesitan tomar decisiones basadas en datos productivos, sanitarios y reproductivos.

Información Clave

- Indicadores de producción láctea, estado sanitario, engorde y fertilidad.
- Seguimiento individual por animal es esencial.
- Análisis de costos y rendimiento como herramienta estratégica.

Gestión del Personal y Proveedores

- Requieren funcionalidades para:
 - Asignación de tareas.
 - Control de desempeño del personal.
 - Comunicación interna.
 - Registro de incidentes.
- Interacción con proveedores: seguimiento de pedidos, inventarios y negociación.

Personalización y Adaptabilidad

- Alta necesidad de personalizar roles, informes, alertas y datos.
- La plataforma debe adaptarse a los procesos existentes, no al revés.

Beneficios Esperados

- Reducción de errores en registros.
- Mayor eficiencia en toma de decisiones.
- Mejora en rentabilidad, bienestar animal y sostenibilidad.

Conclusiones Generales

Categoría	Independientes	Empresas Ganaderas
-----------	----------------	--------------------

Categoría	Independientes	Empresas Ganaderas
Registro de datos	Manual o Excel, poco estructurado	Similar situación, pero a mayor escala
Desafíos principales	Organización, tiempo, clima	Coordinación, errores, análisis de datos
Interacción con tecnología	Abiertos a soluciones móviles	Necesidad crítica de digitalización
Funciones clave esperadas	Recordatorios, historial, salud, alimentación	Reportes, planificación, gestión de personal
Relevancia de personalización	Media	Alta

2.3. Needfinding

2.3.1. User Personas

Para construir nuestros User Persona, nos basamos en la información obtenida y analizada a partir de las entrevistas realizadas. Se identificó que ambos segmentos debían estar representados por perfiles masculinos. A partir de las respuestas recopiladas, se elaboró un User Persona que refleja los objetivos, motivaciones y frustraciones más comunes entre los participantes. Finalmente, se realizó un análisis que permitió definir los valores y habilidades que resume las características más destacadas de cada uno de los segmentos definidos.

Segmento Ganadero Independiente

PERSONA: Carlos Rodriguez Vega

NAME	MARKET SIZE	TYPE
Carlos Rodriguez Vega	 54 %	Rational
	Goals <ul style="list-style-type: none"> - Correcta gestión de su negocio de ganadería. - Informes precisos que ayuden en la toma de decisiones de sus ganados. -Asegurar el bienestar de sus ganados. 	
Demographic <p>Male 33 years</p> <p>Trujillo, Perú</p> <p>Human Resources Manager</p>	Background <p>Carlos es un adulto al cual le ha interesado el mundo de la ganadería desde muy joven. Inicio como ganadero independiente a los 20 años y se mantiene así desde entonces, siendo un apasionado por el mundo de los animales. Su familia contaba con ganados desde que el era pequeño, por lo que aprendió acerca del negocio y el cuidado de estos animales desde muy temprana edad.</p>	
Brands and influencers <p></p> <p></p> <p></p>	Skills <p>Liderazgo</p>  <p>Creatividad</p>  <p>Comunicación</p> 	Frustrations <ul style="list-style-type: none"> Poca información registrada la cual sea de ayuda en la toma de decisiones. Falta de seguimiento en los tratamientos de sus ganados. Falta de información en la gestión de sus ganados
	Channels <p>Email Messaging Phone</p> <p>Website Mobile app</p>	Motivations <ul style="list-style-type: none"> Pasión por la ganadería. Aumentar su conocimiento. Conseguir una mayor rentabilidad.

UXPRESSIA

This persona was built in upressoia.com

Segmento Empresa Ganadera

PERSONA: José Galindo

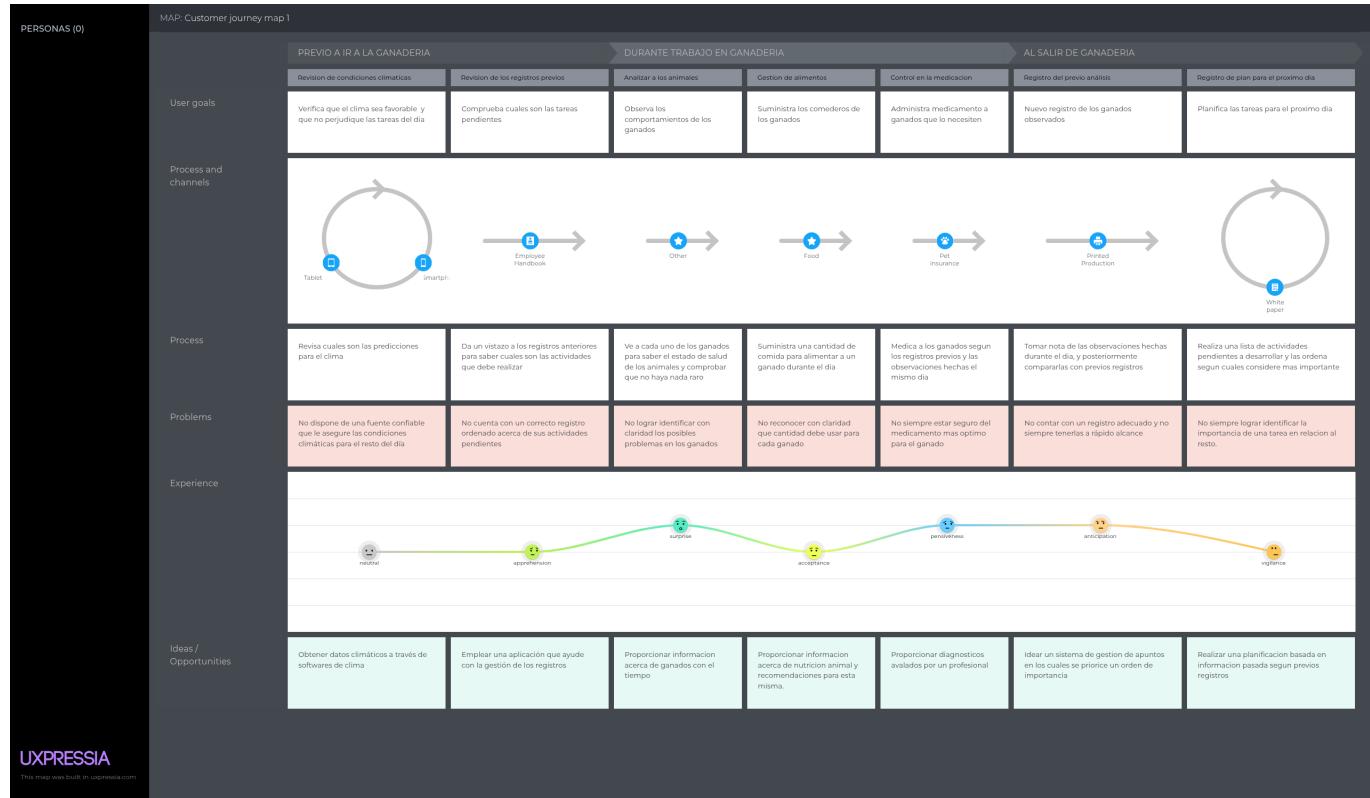
NAME	MARKET SIZE	TYPE
José Galindo	 65 %	Artisan
	Goals <ul style="list-style-type: none"> - Reducir costos en la empresa sin afectar la producción. - Mejorar la gestión de los ganados. - Garantizar la rentabilidad de la empresa en los próximos años. 	
Demographic <p> ♂ Male 48 years 📍 Lima Perú Gerente Ganadero </p>	Background <p>José se graduó como ingeniero industrial y lleva involucrado en el mundo de la gestión de empresas ganaderas desde hace 13 años. Desde pequeño mostró interés por el mundo ganadero, por lo que ese interés fue lo que lo impulsó a relacionarse en las empresas de dicho rubro.</p>	
Skills <p> Negociación:  100 Liderazgo:  100 Gestión:  80 Comunicación:  85 </p>	Motivations <ul style="list-style-type: none"> - Apego hacia el sector agroindustrial. - Pasión por el mundo ganadero. - Amor por su trabajo 	Frustrations <ul style="list-style-type: none"> - Falta de eficiencia al momento de recopilar datos - Dificultad al controlar los gastos en la empresa - Falta de tiempo al realizar actividades
Brands and influencers <div style="display: flex; justify-content: space-around;">    </div>		
Channels <div style="display: flex; justify-content: space-around;">     </div> <p>Mobile app Website Smartphone Messaging</p>		

2.3.2. User Task Matrix

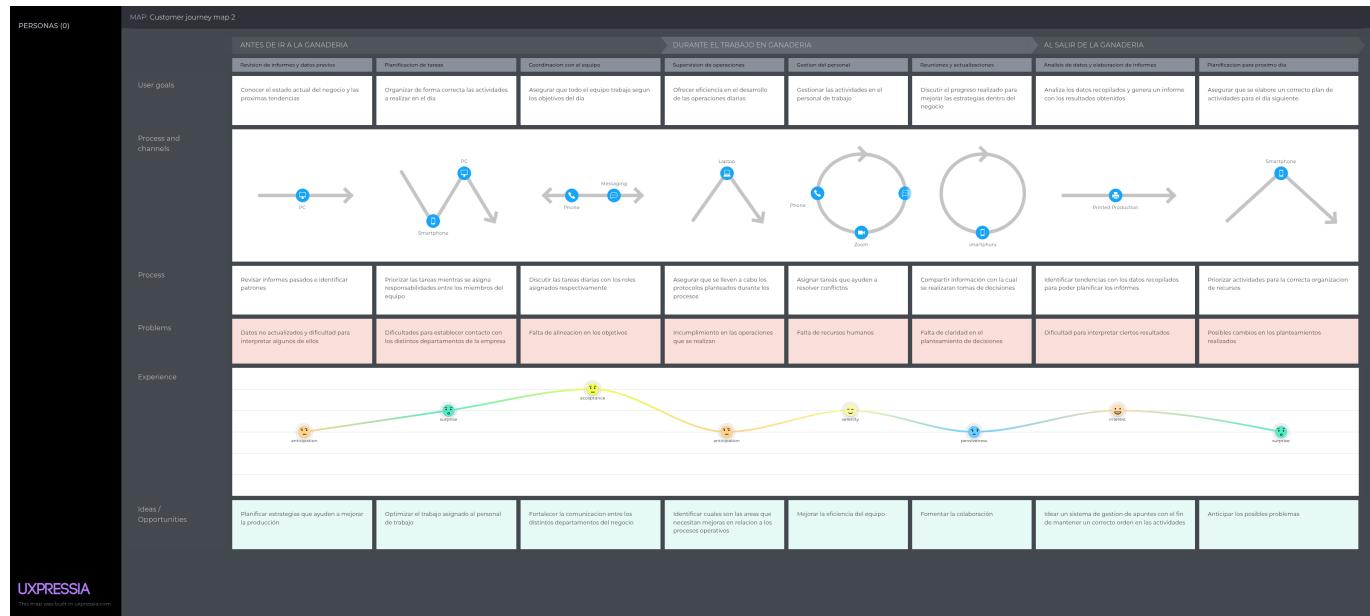
Tarea	Ganadero Independiente	Empresa Ganadera
Registrar nuevos animales	Alta (Manualmente con papel y lápiz)	Alta (Manualmente con registros en cuadernos)
Controlar la alimentación del ganado	Alta (Registro manual en cuadernos)	Alta (Registro manual en registros de alimentación)
Programar citas con el veterinario	Media (Llamadas telefónicas)	Alta (Agenda manual de citas)
Gestionar vacunaciones	Alta (Registro manual en registros de vacunación)	Alta (Registro manual en cuadernos de vacunación)
Realizar seguimiento del estado de salud	Alta (Observación manual del ganado)	Alta (Registro manual de síntomas y signos)
Verificar el pronóstico del tiempo	Media (Consultando en línea o escuchando el pronóstico)	Alta (Consultando en línea o por radio)
Consultar información sobre razas de ganado	Media (Investigación en línea o en libros)	Alta (Investigación en línea o en libros)
Mantenimiento de infraestructura y equipos	Alta (Reparaciones manuales y limpieza)	Alta (Reparaciones manuales y limpieza)
Manejo de la reproducción y cría de ganado	Alta (Observación y gestión manual del ciclo reproductivo)	Alta (Observación y gestión manual del ciclo reproductivo)
Registro de movimientos de ganado	Alta (Registro manual de traslados, compra y venta)	Alta (Registro manual de traslados, compra y venta)

2.3.3. User Journey Mapping

Journey Map for Ganadero Independiente



Journey Map for Empresa Ganadera



2.3.4. Empathy Mapping

Empathy Map for Ganadero Independiente

PERSONA: Empathy map

1.WHO are we empathizing with?

Carlos Rodriguez es un joven ganadero independiente que tiene como objetivo mejorar la gestión de su ganadería

7.What do they THINK and FEEL?

“
"Quisiera hacer uso de un sistema el cual mejore la eficiencia de mi trabajo"
"Me gustaría un sistema que aumente la productividad de mi negocio"
”

2.What do they need to DO?

- Mejorar la eficiencia en la gestión de su ganadería
- Aumentar la productividad
- Toma decisiones previamente consultadas acerca de la salud del ganado

6.What do they HEAR?

"Hay plataformas las cuales me pueden ayudar a mejorar la eficiencia en la gestión de mi negocio"
 Toma en cuenta las sugerencias de sus compañeros ganaderos



3.What do they SEE?

- Observa sistemas de gestión de ganados más complejos
- Observa que sistemas utilizan otros ganaderos en sus negocios
- Visualiza la necesidad de mejora de eficiencia en la gestión de su ganadería

5.What do they DO?

Harlar con distintos compañeros ganaderos para conocer acerca de los sistemas que emplean en la gestión de sus ganaderías.
 Informar acerca de la necesidad de modernizar sus prácticas en la gestión de ganadería.

PAINS

Problemas relacionados al seguimientos de ganados
 Dificultad al realizar registros
 Preocupación por la eficacia

GAINS

Mejora en la eficiencia y productividad del negocio
 Capacidad para tomar mejores decisiones con previa información
 Mejora en la comodidad de las actividades diarias

4.What do they SAY?

“
Quiero funciones básicas en un aplicativo las cuales me permitan mantener un seguimiento en el cuidado de mis ganados
”

UXPRESSIA

This persona was built in uxpressia.com

Empathy Map for Empresa Ganadera

PERSONA: Empathy map

1.WHO are we empathizing with?

Jose Galindo es el gerente general en su empresa de ganadería y busca mejorar los procesos dentro del negocio con el fin de que las operaciones se realicen con mayor eficiencia.

7.What do they THINK and FEEL?

“Motivado con la posibilidad de implementar un sistema que mejore la eficiencia dentro de mi empresa”

2.What do they need to DO?

Mejorar la gestión ganadera en la empresa
Llevar a cabo un correcto seguimiento en el cuidado del ganado
Realizar un correcto seguimiento de la rentabilidad y costos en la producción

6.What do they HEAR?

Escucha recomendaciones acerca de sistemas los cuales podría implementar para mejorar la gestión ganadera

Recibe comentarios acerca de la importancia de llevar a cabo informes detallados con información relevante para una mejor toma de decisiones en la empresa



3.What do they SEE?

Observa la necesidad de implementar una mejora en la eficiencia y rentabilidad de la empresa
Observa como las empresas competidoras emplean sistemas los cuales colaboran en la mejora de los procesos.

5.What do they DO?

Hace uso de métodos tradicionales para el seguimiento del cuidado del ganado

Lleva a cabo un análisis de rentabilidad y costos operativos de forma tradicional

Busca soluciones para mejorar la eficiencia de la empresa

PAINS

Preocupación por la complejidad de los sistemas existentes y por lo que podría tomar acostumbrarse a usarlo con frecuencia.

Falta de herramientas tecnológicas dentro de la empresa

GAINS

Acceso a informes precisos con datos claves para la toma de decisiones en la empresa

Mejora en la eficiencia de operaciones y gestión de recursos

4.What do they SAY?

“Quiero funciones básicas en un aplicativo las cuales me permitan tener un correcto seguimiento en el cuidado de los ganados”

UXPRESSIA

This persona was built in uxpressia.com

2.3.5. As-is Scenario Mapping

Steps	Inspección del ganado en el campo	Registro de datos de salud y alimentación	Alimentación del ganado	Manejo de la reproducción y la cría	Control de enfermedades y parásitos
Doing	<p>Busca signos de posibles enfermedades dentro del ganado. Comunica lo observado al personal responsable</p>	<p>Toma apuntes directamente en un cuaderno. Realiza cálculos de forma manual para determinar el consumo necesario para el ganado. Analiza los próximos gastos en cuanto a medicina para los animales.</p>	<p>Avisa a los trabajadores acerca de las cantidades de alimento que deben ingerir el ganado mientras supervisa el trabajo realizado.</p>	<p>Diseña programa de inseminación artificial en las vacas. Supervisa el proceso de gestación y parto.</p>	<p>Lleva un registro de las vacunas aplicadas al ganado. Implementa sistema de fumigación para el hábitat del ganado.</p>
Thinking	<p>"Sería una buena opción contar con un sistema el cual nos permita mantener un control de las observaciones realizadas con el fin de determinar irregularidades con el tiempo"</p>	<p>"Debemos tener en cuenta cuanto gastamos por cada ganado, esto con el fin de mantener la rentabilidad dentro de la empresa"</p>	<p>"Necesito conocer más acerca del alimento que le proporciono al ganado y si existe otras mejores alternativas a las que usamos"</p>	<p>"Es importante mantener un seguimiento durante el proceso de gestación de las crías, con el fin de evitar problemas al momento del parto"</p>	<p>"Es importante mantener un ambiente limpio para el ganado. Esto ayudaría a evitar enfermedades a futuro"</p>
Feeling	<p>Se siente angustiado por las posibles enfermedades que se puedan encontrar</p>	<p>Estreñido y cansado de tener que calcular por su cuenta propia y tener que usar métodos tradicionales para ello</p>	<p>Se siente abrumado por la responsabilidad de la alimentación de los ganados</p>	<p>Preocupación por la correcta implementación de las medidas al momento del parto del ganado.</p>	<p>Se siente responsable en el correcto cuidado del ganado.</p>

2.4. Ubiquitous Language

Palabra	Descripción
Animal Health (Salud Animal)	Mantener la salud y prevenir enfermedades en el ganado.
Balanced Feeding (Alimentación Equilibrada)	Proporcionar una dieta equilibrada que cumpla con los requisitos nutricionales específicos de los animales.
Artificial Insemination (Inseminación Artificial)	Introducir esperma en el tracto reproductivo de una hembra sin necesidad de apareamiento natural.
Rectal Palpation (Palpación Rectal)	Examen para evaluar la condición reproductiva de una hembra, especialmente en el ganado vacuno.
Deworming (Desparasitación)	Administrar medicamentos antiparasitarios para controlar y prevenir la infestación de parásitos internos y externos en el ganado.
Branding (Marcado)	Identificación de ganado mediante la aplicación de un sello metálico caliente en la piel del animal.
Weaning (Destete)	Separación gradual de los terneros de sus madres para cesar la lactancia y promover la independencia alimentaria.
Grazing Rotation (Rotación de Pastoreo)	Mover el ganado entre diferentes áreas de pastoreo para optimizar el uso del suelo y prevenir el sobrepastoreo.
Dystocia (Distocia)	Dificultades durante el parto que pueden requerir intervención veterinaria.
Dusting (Aplicación de Polvos)	Aplicación de insecticidas en polvo o en aerosol sobre el pelaje del ganado para controlar infestaciones de insectos y parásitos externos.
Campaign (Campaña)	Período durante el cual se llevan a cabo actividades específicas en la gestión de ganadería, con objetivos definidos y metas establecidas.

Palabra	Descripción
Batch (Lote)	Grupo de animales criados o tratados juntos, que se manejan y monitorean como una unidad durante un período específico de tiempo.
Fumigation (Fumigación)	Aplicación de productos químicos o biológicos para eliminar o controlar plagas, insectos, parásitos o enfermedades en el ganado y su entorno.

Capítulo III: Requirements Specification

Con el objetivo de diseñar una solución de software centrada en las necesidades reales de los usuarios, se desarrollaron una serie de herramientas y técnicas que permiten comprender, planificar y proyectar el comportamiento del sistema desde una perspectiva empática y funcional.

3.1. To-Be Scenario Mapping

El To-Be Scenario Mapping permite proyectar el escenario ideal de interacción del usuario con el sistema, describiendo cómo deberían desarrollarse los procesos una vez implementada la solución. A través de esta técnica se identifican los cambios esperados respecto al estado actual, destacando mejoras en la experiencia del usuario, eficiencia operativa y cumplimiento de objetivos del proyecto. Este mapeo resulta clave para alinear el diseño del sistema con las verdaderas necesidades de los usuarios y los objetivos estratégicos de la solución.

Steps	Inspección del Ganado en el Campo	Registro de Datos de Salud y Alimentación	Alimentación del Ganado	Manejo de la Reproducción y la Cria	Control de Enfermedades y Parásitos
Doing	<p>Monitoreo en tiempo real con alertas y ejecución de pasos óptimos para el manejo (alimentación, ajuste ambiental, saneamiento).</p> <p>Reportes y análisis para dirigir el personal y documentación de intervenciones.</p>	<p>Registrar datos de salud, peso y alimentación utilizando la plataforma.</p> <p>Consultar informes detallados de cada animal o lote y filtrar por períodos.</p>	<p>Planificar y programar automáticamente las alimentaciones según cálculos precisos.</p> <p>Supervisar consumo y recibir alertas de desviaciones.</p>	<p>Programación y seguimiento de eventos reproductivos (partos, inseminación, control de gestación) y registro detallado de historiales.</p>	<p>Implementación de medidas preventivas (desparasitación, vacunas, controles) con registro y monitoreo constante.</p>
Thinking	<p>Evaluación de históricos y tendencias para validar las recomendaciones actuales.</p> <p>Ajustar protocolos según condiciones y patrones observados.</p>	<p>Análisis de costos, rentabilidad y complejidad entre alimentación y salud para optimizar recursos.</p>	<p>Evaluación de inventarios, proveedores y análisis histórico para ajustar estrategias y maximizar la eficiencia operativa.</p>	<p>Optimizar la tasa de éxito aplicando recomendaciones basadas en datos predictivos.</p> <p>Planificar a largo plazo según proyecciones de crecimiento.</p>	<p>Evaluación de protocolos y optimización de estrategias sanitarias basadas en análisis históricos y alertas tempranas.</p>
Feeling	<p>Confianza y seguridad al contar con un sistema que reduce el incertidumbre y respalda decisiones.</p>	<p>Seguridad al tener información centralizada y precisa que favorece la toma de decisiones.</p>	<p>Seguridad y respaldo gracias a un sistema que garantiza un suministro controlado y adecuado.</p>	<p>Tranquilidad y seguridad al disponer de un seguimiento detallado que permite anticipar y abordar problemas.</p>	<p>Confianza y control ante la mitigación de riesgos, respaldados por un sistema automatizado.</p>

3.2. User Stories

Las User Stories representan una herramienta fundamental dentro de las metodologías ágiles para capturar los requerimientos funcionales desde la perspectiva del usuario. Cada historia describe una necesidad concreta, quién la necesita y con qué propósito, facilitando la planificación, priorización y desarrollo iterativo del sistema. Esta técnica garantiza que cada funcionalidad responda a una necesidad real, fomentando un desarrollo orientado al valor y alineado con las expectativas del usuario final.

Epic ID	Título	Descripción	Relacionado con (Epic ID)
EP001	Registrar Vacuna	Como ganadero quiero poder registrar mi vacuna para que tenga control sobre la salud de mi bovino	
Story ID	Título	Descripción	Criterios de Aceptación

Epic ID	Título	Descripción
US001	Agregar Vacuna al Registro	<p>- E01: Acceso al formulario para agregar vacuna Dado que un ganadero autenticado. Cuando Accede al módulo/vacunas de la aplicación. Entonces Se muestra el formulario para el registro de una nueva vacuna, permitiéndole ingresar todos los datos necesarios.</p> <p>- E02: Registro exitoso de la vacuna Dado que El ganadero tiene el formulario abierto. Cuando Ingresa todos los datos requeridos correctamente y envía el formulario. Entonces: La vacuna se registra exitosamente y se muestra una confirmación del registro.</p> <p>- E03: Manejo de errores en el formulario Dado que: Un ganadero autenticado. Cuando: Ingresa datos incompletos o erróneos en el formulario. Entonces: El sistema muestra un mensaje de error específico, indicando los campos que deben corregirse.</p>

Epic ID	Título	Descripción
US002	Búsqueda de Vacunas	<p>Como ganadero, quiero buscar vacunas previamente registradas para evitar la duplicación y garantizar que se administre la vacuna correcta a cada bovino.</p> <p>- E01: Búsqueda por fecha de administración Dado que el ganadero posee múltiples vacunas registradas. Cuando: Realiza una búsqueda filtrando por fecha de administración. Entonces: Se despliega una lista de vacunas administradas en la fecha especificada.</p> <p>- E02: Búsqueda por tipo de vacuna Dado que el ganadero tiene varias vacunas en el registro. Cuando filtra la búsqueda por el tipo específico de vacuna. Entonces se muestran únicamente las vacunas que coinciden con el tipo seleccionado.</p> <p>- E03: Sin resultados en la búsqueda Dado que el ganadero tiene vacunas registradas. Cuando busca con criterios que no concuerdan con ningún registro. Entonces se muestra un mensaje informando que no se encontraron coincidencias.</p>

Epic ID	Título	Descripción	
US003	Gestión de Registros de Vacunas	<p>Como ganadero, necesito poder editar o eliminar el registro de una vacuna para garantizar que la información se mantenga precisa y actualizada.</p> <p>- E01: Eliminación exitosa de vacuna Dado que el ganadero selecciona una vacuna del registro. Cuando solicita eliminar dicha vacuna y confirma la acción. Entonces el sistema elimina la vacuna y muestra una confirmación de la eliminación.</p> <p>- E02: Edición exitosa de vacuna Dado que el ganadero visualiza el registro de una vacuna. Cuando modifica los datos y guarda los cambios. Entonces el sistema actualiza la información de la vacuna y muestra un mensaje de confirmación.</p> <p>- E03: Error al eliminar vacuna Dado que el ganadero intenta eliminar una vacuna. Cuando se produce un error interno en el sistema. Entonces no se elimina la vacuna y se muestra un mensaje de error descriptivo.</p> <p>- Escenario 4: Error al editar vacuna Dado que el ganadero intenta modificar una vacuna. Cuando se produce un error durante la actualización (por ejemplo, validación fallida). Entonces el sistema no actualiza los datos y se notifica el error con detalle de los problemas encontrados.</p>	EP001
TS001	Crear	Como desarrollador,	EP001

Epic ID	Vacuna vía API	Descripción
		necesito exponer un endpoint para registrar una vacuna vía API, de modo que los features de la aplicación dispongan de este registro.
		<p>- E01: Registro exitoso Dado que el endpoint /vacunas está disponible. Cuando se envía una solicitud POST con datos válidos. Entonces se recibe respuesta 201 (Created) con el recurso de vacuna recién creado.</p>
		<p>- E02: Error por datos inválidos Dado que el endpoint /vacunas está disponible. Cuando se envía una solicitud POST sin el campo obligatorio "nombre" (u otro dato requerido). Entonces: Se recibe respuesta 400 (Bad Request) con el mensaje: "Falta el nombre de la vacuna".</p>
		<p>- E03: Error por formato incorrecto Dado que El endpoint /vacunas está disponible. Cuando: Se envía una solicitud POST con un formato incorrecto (p.ej., fecha inválida). Entonces: Se recibe respuesta 400 con el mensaje: "Formato de fecha de vacunación no válido".</p>
		<p>- E04: Error por vacuna duplicada Dado que: El endpoint /vacunas está disponible y ya existe una vacuna con el mismo nombre. Cuando: Se envía una solicitud POST con datos para una vacuna existente. Entonces: Se recibe respuesta 409 (Conflict) con el mensaje: "Ya existe una vacuna con el mismo nombre".</p>

Epic ID	Título	Descripción	
TS002	API para Búsqueda de Vacunas	<p>Como desarrollador, quiero implementar un endpoint para buscar vacunas mediante criterios específicos, permitiendo filtrar y obtener registros de manera eficiente.</p> <p>- E01: Búsqueda exitosa Dado que: El endpoint /vacunas está disponible. Cuando: Se realiza una solicitud GET con parámetros de búsqueda válidos. Entonces: Se recibe respuesta 200 (OK) con una lista de vacunas que cumplen los criterios.</p> <p>- E02: Búsqueda sin resultados Dado que: El endpoint /vacunas está disponible. Cuando: Se realiza una solicitud GET con parámetros que no arrojan coincidencias. Entonces: Se recibe respuesta 200 con un mensaje informando que no se encontraron resultados.</p>	EP001

Epic ID	Título	Descripción	
TS003	API para Gestión de Vacunas	<p>Como desarrollador, necesito implementar endpoints para editar y eliminar registros de vacunas, asegurando que se mantenga la integridad y actualización de los datos.</p> <p>- E01: Edición exitosa Dado que: Existe un registro de vacuna y el endpoint /vacunas/{id} está disponible. Cuando: Se envía una solicitud PUT con datos válidos para actualizar. Entonces: Se recibe respuesta 200 (OK) con la vacuna actualizada.</p> <p>- E02: Error al editar vacuna Dado que: El endpoint /vacunas/{id} está disponible. Cuando: Se envía una solicitud PUT con datos inválidos. Entonces: Se recibe respuesta 400 con un mensaje de error descriptivo.</p> <p>E03: Eliminación exitosa Dado que: Existe un registro de vacuna y el endpoint /vacunas/{id} está disponible. Cuando: Se envía una solicitud DELETE para eliminar la vacuna. Entonces: Se recibe respuesta 200 (OK) confirmando la eliminación.</p>	EP001

Epic ID	Título	Descripción		Relacionado con (Epic ID)
EP002	Registrar Ganado	Como ganadero quiero registrar para llevar un control detallado de mi inventario de ganado y su trazabilidad sanitaria		
Story ID	Título	Descripción	Criterios de Aceptación	
US004	Registro de Bovino en Lote	Como usuario autenticado, quiero registrar un bovino en un lote	- E01: Registro completo de un bovino	EP002

Epic ID	Título	Descripción
		<p>lote específico para tener control detallado de la crianza y manejo del animal.</p> <p>Dado que un usuario autenticado.</p> <p>Cuando selecciona la opción de registrar un nuevo bovino e ingresa los datos: procedencia, raza, peso inicial, fecha de registro, fotografía, estado de salud, fecha de nacimiento o edad, y observaciones.</p> <p>Entonces el sistema genera un ID único, asocia el bovino a un lote específico y confirma el registro exitoso.</p> <p>- E02: Registro sin autenticación</p> <p>Dado que un usuario no autenticado.</p> <p>Cuando intenta acceder al módulo de gestión de lotes para registrar un bovino.</p> <p>Entonces se redirige al usuario a la página de inicio de sesión.</p> <p>- E03: Registro incompleto</p> <p>Dado que un usuario autenticado en el registro de un bovino.</p> <p>Cuando envía el formulario sin completar todos los campos obligatorios.</p> <p>Entonces se muestra un mensaje de error solicitando la información faltante.</p> <p>- Escenario 4: Registro con datos inválidos</p> <p>Dado que un usuario autenticado al registrar un bovino.</p> <p>Cuando ingresa datos en un formato incorrecto o no conforme a lo esperado.</p> <p>Entonces el sistema rechaza el registro y muestra un mensaje de error especificando el formato correcto.</p>

Epic ID	Título	Descripción	
US005	Buscar Información de Bovinos	<p>Como usuario, quiero poder buscar animales registrados para acceder de forma rápida y ordenada a la información necesaria.</p>	<p>- E01: Búsqueda exitosa</p> <p>Dado que: El usuario autenticado en la plataforma.</p> <p>Cuando: Realiza una búsqueda (con filtros y criterios) desde el módulo respectivo.</p> <p>Entonces: Se muestra una lista clara y ordenada con los resultados correspondientes.</p>
US006	Actualizar Información de Bovinos	<p>Como usuario, quiero gestionar la información de los animales registrados para mantener la base de datos actualizada y precisa.</p>	<p>- E01: Actualización exitosa</p> <p>Dado que el usuario autenticado y accediendo al módulo de gestión de animales.</p> <p>Cuando actualiza la información requerida y guarda los cambios.</p> <p>Entonces el sistema confirma la actualización mediante un mensaje de éxito.</p>
TS004	API para Registro de Animales	<p>Como desarrollador, quiero implementar un endpoint para registrar un bovino en un lote específico, permitiendo la correcta integración de los datos en la aplicación.</p>	<p>- E01: Registro exitoso</p> <p>Dado que el endpoint /animales está disponible.</p> <p>Cuando se envía una solicitud POST con todos los datos válidos.</p> <p>Entonces se recibe respuesta 201 (Created) con el registro del animal.</p> <p>- E02: Error en el registro</p> <p>Dado que el endpoint /animales está disponible.</p> <p>Cuando se envía una solicitud POST con datos inválidos o incompletos.</p> <p>Entonces se recibe respuesta 400 (Bad Request) con un mensaje de error.</p>

Epic ID	Título	Descripción	
TS005	API para Búsqueda de Animales	<p>Como desarrollador, necesito un endpoint que permita buscar animales registrados usando parámetros de búsqueda, facilitando la localización de registros específicos.</p> <p>- E01: Búsqueda exitosa Dado que el endpoint /animales está disponible. Cuando se realiza una solicitud GET con parámetros válidos. Entonces se recibe respuesta 200 (OK) con la lista de animales que cumplen los criterios.</p> <p>- E02: En caso de parámetros sin resultados, se indica adecuadamente que no se encontraron animales.</p>	EP002
TS006	API para Gestión de Animales	<p>Como desarrollador, quiero implementar funcionalidades para editar y eliminar animales registrados, asegurando la actualización y manejo correcto de la información.</p> <p>- E01: Edición exitosa Dado que el endpoint /animales/{id} está disponible y existe el animal. Cuando se envía una solicitud PUT con información actualizada y válida. Entonces se recibe respuesta 200 (OK) con el animal actualizado.</p> <p>- E02: Eliminación exitosa Dado que el endpoint /animales/{id} está disponible. Cuando se realiza una solicitud DELETE para el animal. Entonces se recibe respuesta 200 con un mensaje de éxito.</p>	EP002

Epic ID	Título	Descripción
EP003	Registrar Campaña	<p>Como usuario de la plataforma, quiero crear y gestionar campañas ganaderas para mejorar el rendimiento y control de las actividades relacionadas con el engorde del ganado.</p>

Epic ID	Título	Descripción		
Story ID	Título	Descripción	Criterios de Aceptación	Relacionado con (Epic ID)
US007	Crear Campaña para Engorde de Ganado	Como usuario de la plataforma, quiero crear una campaña para engordar el ganado asignado, definiendo parámetros como duración, objetivo y selección de animales o establos.	<p>E01: Creación de campaña</p> <p>Dado que el usuario autenticado.</p> <p>Cuando selecciona la opción de crear una campaña y completa los campos (duración, objetivo, selección de animales o establos).</p> <p>Entonces se crea la campaña y se muestra una confirmación</p>	EP003
US008	Asociar Empleados a Campaña	Como empresario ganadero, quiero asignar empleados a una campaña para organizar mejor el personal durante la ejecución de esta.	<p>- E01: Asociación exitosa</p> <p>Dado que: El empresario ganadero autenticado.</p> <p>Cuando: Selecciona una campaña existente y asigna empleados válidos.</p> <p>Entonces: Los empleados quedan asociados a la campaña y se muestra una notificación de éxito.</p>	EP003

Epic ID	Título	Descripción	
US009	Gestión de Campañas	<p>Como usuario, quiero gestionar (editar o modificar) la campaña creada para ajustarla según las necesidades y cambios en la planificación.</p> <p>- E01: Gestión para ganadero independiente Dado que: El usuario autenticado con una campaña creada. Cuando: Realiza cambios (añadir/eliminar tiempo, animales, establos o modificar el objetivo). Entonces: Los cambios se reflejan instantáneamente y se confirma la actualización.</p> <p>- E02: Gestión para empresa ganadera Dado que: La empresa con campaña activa. Cuando: Realiza cambios adicionales como la asignación o eliminación de personal, además de otros ajustes en el objetivo. Entonces: El sistema actualiza y confirma los cambios en la campaña.</p>	EP003
TS007	API para Creación de Campaña	<p>Como desarrollador, necesito crear un endpoint que permita la creación de campañas, de modo que se puedan iniciar campañas de engorde en la aplicación.</p> <p>- E01: Creación exitosa Dado que el endpoint /campanas está disponible. Cuando se envía una solicitud POST con datos válidos para la campaña. Entonces Se recibe respuesta 201 (Created) con el registro de la campaña.</p> <p>- E02: Error en la creación Dado que el endpoint /campanas está disponible. Cuando se envía una solicitud POST con datos incompletos o inválidos. Entonces se recibe respuesta 400 (Bad Request) con un mensaje de error.</p>	EP003

Epic ID	Título	Descripción
TS008	API para Asociar Empleados a Campaña	<p>Como desarrollador, necesito un endpoint para asociar empleados a campañas, permitiendo la asignación de personal a cada campaña desde la aplicación.</p> <p>- E01: Asociación exitosa Dado que el endpoint /campanas/{id}/empleados está disponible. Cuando se envía una solicitud POST con los datos de un empleado válido para la campaña seleccionada. Entonces se recibe respuesta 201 (Created) confirmando la asociación.</p> <p>- E02: Error al asociar Dado que el endpoint /campanas/{id}/empleados está disponible. Cuando se envía una solicitud POST con un empleado inválido. Entonces se recibe respuesta 400 (Bad Request) con el mensaje de error correspondiente.</p>

Epic ID	Título	Descripción
TS009	API para Gestión de Campañas	<p>Como desarrollador, necesito implementar endpoints que permitan editar y eliminar campañas, facilitando su gestión integral desde la plataforma.</p> <p>- E01: Edición exitosa Dado que el endpoint /campanas/{id} está disponible. Cuando se envía una solicitud PUT con información válida para actualizar la campaña. Entonces se recibe respuesta 200 (OK) con la campaña actualizada.</p> <p>- E02 Error al editar campaña Dado que el endpoint /campanas/{id} está disponible. Cuando se envía una solicitud PUT con datos inválidos. Entonces se recibe respuesta 400 (Bad Request) con un mensaje de error.</p> <p>- E03 Eliminación exitosa Dado que el endpoint /campanas/{id} está disponible. Cuando se envía una solicitud DELETE para eliminar la campaña. Entonces se recibe respuesta 200 (OK) con un mensaje confirmando la eliminación.</p>

Epic ID	Título	Descripción	Relacionado con (Epic ID)
EP004	Registrar Personal	Como empresario ganadero, quiero poder registrar y gestionar a mis empleados para organizar eficazmente los recursos humanos de mi empresa,	
Story ID	Título	Descripción	Criterios de Aceptación

Epic ID	Título	Descripción	
US010	Registro de Personal	<p>Como empresario ganadero, quiero poder registrar a mis empleados para organizar y gestionar de forma efectiva los recursos humanos de mi empresa.</p> <p>- E01: Registro completo de empleado Dado que: El empresario autenticado. Cuando: Ingresa correctamente todos los datos requeridos para un empleado. Entonces: El sistema registra al empleado y muestra una confirmación exitosa.</p> <p>- E02: Registro incompleto o incorrecto Dado que: El empresario autenticado. Cuando: Intenta registrar a un empleado sin ingresar toda la información esencial. Entonces: Se muestra un mensaje de error indicando la carencia de información requerida</p>	EP004
US011	Búsqueda de Personal	<p>Como empresario ganadero, quiero buscar entre los empleados registrados para localizar y, de ser necesario, gestionar la información de un empleado en específico.</p> <p>E01: Búsqueda exitosa Dado que: El empresario con sesión iniciada. Cuando: Realiza la búsqueda utilizando filtros (por nombre, cargo, etc.). Entonces: Se muestra el empleado que cumple con los criterios de búsqueda.</p>	EP004

Epic ID	Título	Descripción	
US012	Gestión de Personal	<p>Como empresario ganadero, necesito gestionar la plantilla de empleados, pudiendo agregar o eliminar registros para mantener la base de datos actualizada.</p>	<p>- E01: Agregar y eliminar empleados</p> <p>Dado que: El empresario autenticado y accediendo al módulo de gestión de personal.</p> <p>Cuando: Realiza las operaciones de agregar y/o eliminar empleados.</p> <p>Entonces: El sistema confirma con un mensaje de éxito cada uno de los cambios efectuados.</p>
TS010	API para Registro de Empleados	<p>Como desarrollador, necesito crear un endpoint para registrar empleados, asegurando que la información de cada uno se almacene correctamente y se confirme la creación.</p>	<p>- E01: Registro exitoso</p> <p>Dado que el endpoint /empleados está disponible.</p> <p>Cuando se envía una solicitud POST con datos válidos para el empleado.</p> <p>Entonces se recibe respuesta 201 (Created) con el registro del empleado.</p> <p>- E02: Error al registrar empleado</p> <p>Dado que el endpoint /empleados está disponible.</p> <p>Cuando se envía una solicitud POST con datos incompletos o inválidos.</p> <p>Entonces se recibe respuesta 400 (Bad Request) con el mensaje de error.</p>

Epic ID	Título	Descripción	
TS011	API para Búsqueda de Empleados	<p>Como desarrollador, necesito implementar un endpoint que permita buscar empleados utilizando filtros específicos, para facilitar la administración de los recursos humanos.</p> <p>- E01: Búsqueda exitosa Dado que el endpoint /empleados está disponible. Cuando se realiza una solicitud GET con parámetros válidos. Entonces se recibe respuesta 200 (OK) y se devuelve una lista de empleados que cumplen los criterios.</p> <p>- E02: Búsqueda sin resultados Dado que el endpoint /empleados está disponible. Cuando se realiza una solicitud GET con parámetros que no arrojan resultados. Entonces se recibe respuesta 200 con un mensaje informando que no se encontraron empleados.</p>	EP004

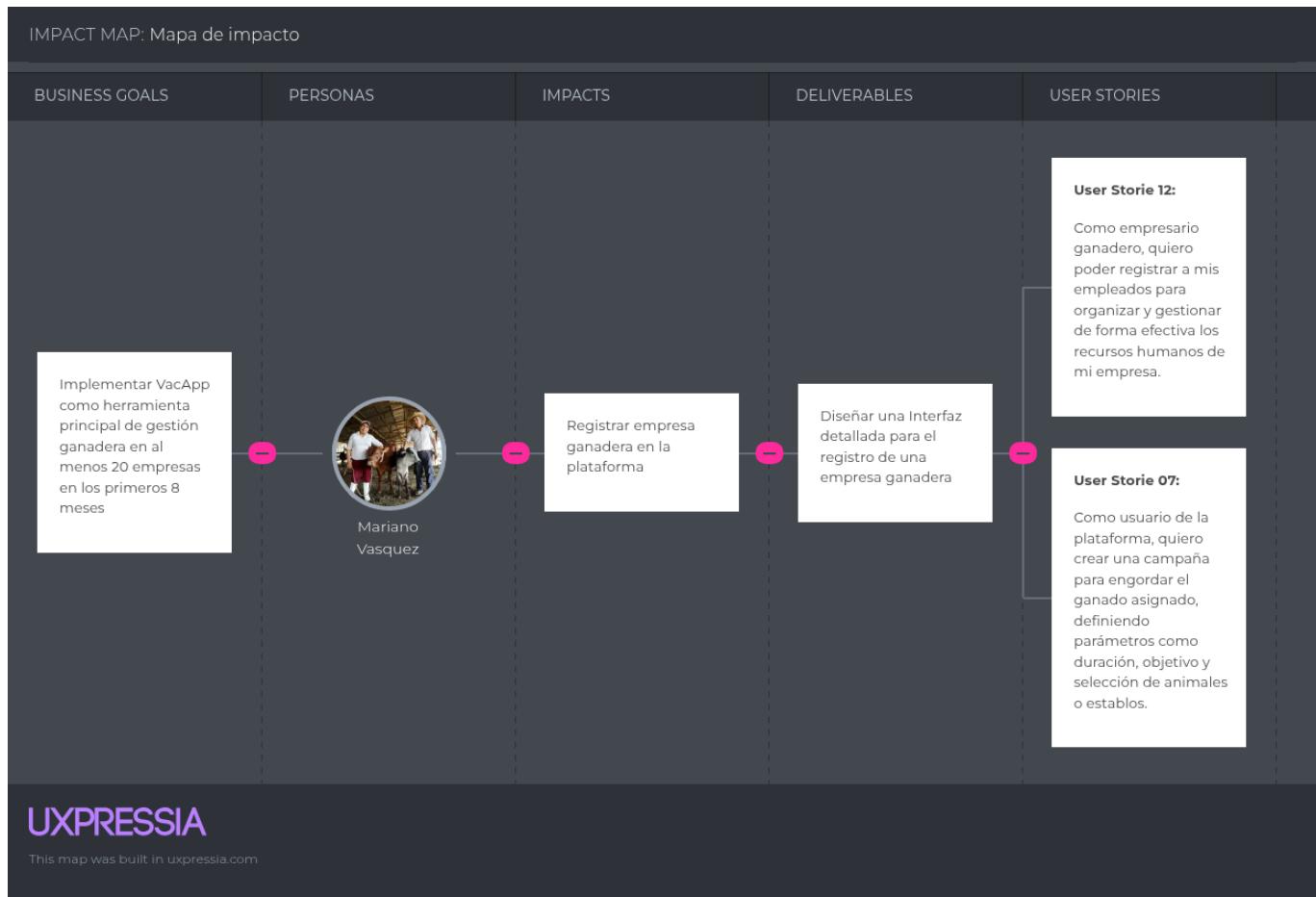
Epic ID	Título	Descripción
TS012	API para Gestión de Empleados	<p>Como desarrollador, necesito implementar funcionalidades para editar y eliminar empleados mediante la API, permitiendo que el empresario administre correctamente su plantilla.</p> <p>- E01: Edición exitosa Dado que el endpoint /empleados/{id} está disponible y existe el empleado. Cuando se envía una solicitud PUT con información válida para actualizar los datos del empleado. Entonces se recibe respuesta 200 (OK) con el registro actualizado.</p> <p>- E02 : Error al editar empleado Dado que el endpoint /empleados/{id} está disponible. Cuando se envía una solicitud PUT con datos inválidos o incompletos. Entonces se recibe respuesta 400 (Bad Request) con el mensaje de error correspondiente.</p> <p>- E03: Eliminación exitosa Dado que el endpoint /empleados/{id} está disponible y existe el empleado a eliminar. Cuando se envía una solicitud DELETE. Entonces se recibe respuesta 200 (OK) confirmando la eliminación con un mensaje de éxito.</p>

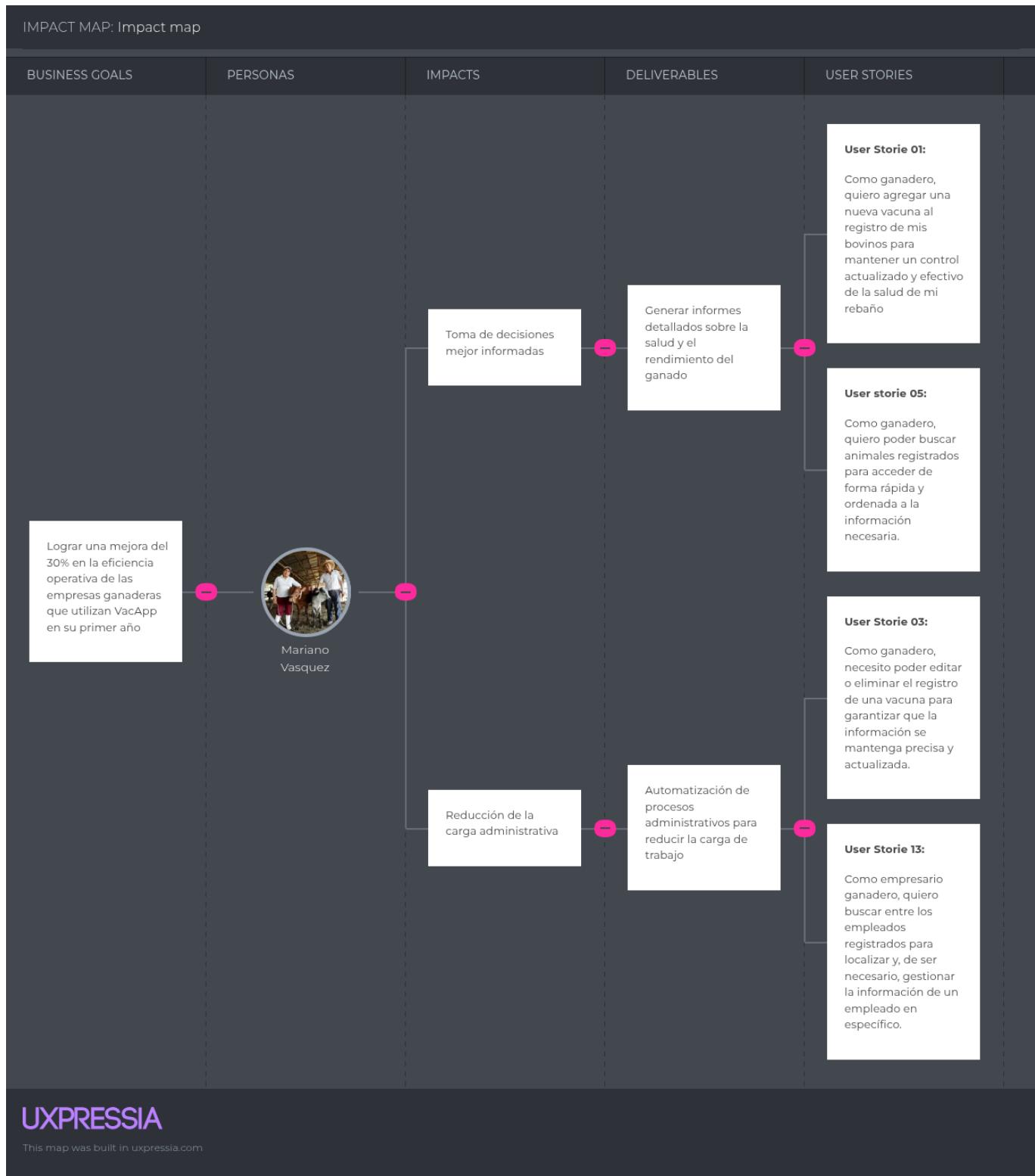
Epic ID	Título	Descripción
EP005	Informarse sobre el Producto	Como visitante, quiero explorar la Landing Page de la aplicación para conocer sus funcionalidades y evaluar si satisface mis necesidades.

Epic ID	Título	Descripción		
Story ID	Título	Descripción	Criterios de Aceptación	Relacionado con (Epic ID)
US015	Explorar la Landing Page	Como visitante, quiero explorar la Landing Page de la aplicación para conocer sus funcionalidades y determinar si satisface mis necesidades.	<p>- E01: Sección "Quiénes Somos"</p> <p>Dado que: El visitante accede a la Landing Page.</p> <p>Cuando: Navega a la sección "Quiénes Somos".</p> <p>Entonces: Encuentra información detallada sobre la misión, visión y valores de la empresa.</p>	EP005
US016	Explorar Secciones Informativas	Como visitante, quiero explorar las secciones informativas (Quiénes Somos, Planes y Funciones) para entender completamente las características y beneficios de la aplicación.	<p>- E01: Navegación en secciones informativas</p> <p>Dado que: El visitante ingresa a la Landing Page.</p> <p>Cuando: Navega por las secciones "Quiénes Somos", "Planes" y "Funciones".</p> <p>Entonces: Se le presenta información detallada y estructurada sobre la empresa, los distintos planes ofrecidos y las funcionalidades de la aplicación.</p>	EP005

3.3. Impact Mapping

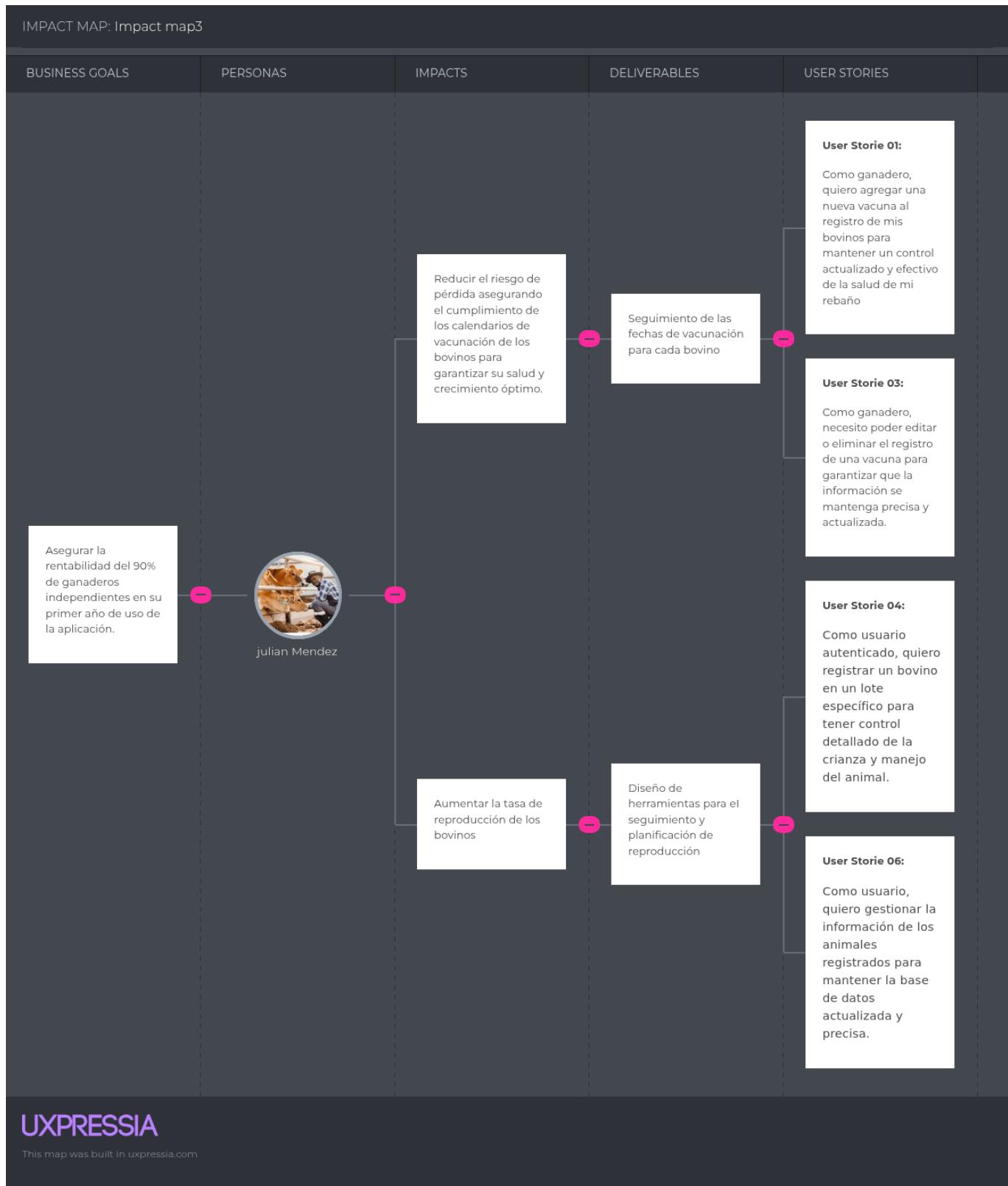
Este es el impact mapping realizado con las entrevistas:





UXPRESSIA

This map was built in upressoia.com



3.4. Product Backlog

Para la gestión ágil de proyectos, se realizó el Product Backlog que pone en alta estima o prioridad las tareas necesarias para el desarrollo completo de la solución. Todas estas realizadas en base a lo anteriormente establecido como las historias de usuario y los mapeos.

Orden	User Story ID	Título	Descripción	Story Points (1/2/3)
-------	---------------	--------	-------------	----------------------

Orden	User Story			Story Points (1/2/3)
	ID	Título	Descripción	
1	US001	Agregar Vacuna al Registro	Como ganadero, quiero agregar una nueva vacuna al registro de mis bovinos para mantener un control actualizado y efectivo de la salud de mi rebaño	2
2	TS001	Crear Vacuna vía API	Como desarrollador, necesito exponer un endpoint para registrar una vacuna vía API, de modo que los features de la aplicación dispongan de este registro.	2
3	TS002	API para Búsqueda de Vacunas	Como desarrollador, quiero implementar un endpoint para buscar vacunas mediante criterios específicos, permitiendo filtrar y obtener registros de manera eficiente.	2
4	US002	Búsqueda de Vacunas	Como ganadero, quiero buscar vacunas previamente registradas para evitar la duplicación y garantizar que se administre la vacuna correcta a cada bovino.	1
5	TS003	API para Gestión de Vacunas	Como desarrollador, necesito implementar endpoints para editar y eliminar registros de vacunas, asegurando que se mantenga la integridad y actualización de los datos.	2
6	TS004	API para Registro de Animales	Como desarrollador, quiero implementar un endpoint para registrar un bovino en un lote específico, permitiendo la correcta integración de los datos en la aplicación.	2
7	US003	Gestión de Registros de Vacunas	Como ganadero, necesito poder editar o eliminar el registro de una vacuna para garantizar que la información se mantenga precisa y actualizada.	2
8	TS005	API para Búsqueda de Animales	Como desarrollador, necesito un endpoint que permita buscar animales registrados usando parámetros de búsqueda, facilitando la localización de registros específicos.	2
9	TS006	API para Gestión de Animales	Como desarrollador, quiero implementar funcionalidades para editar y eliminar animales registrados, asegurando la actualización y manejo correcto de la información.	2
10	US004	Registro de Bovino en Lote	Como usuario autenticado, quiero registrar un bovino en un lote específico para tener control detallado de la crianza y manejo del animal.	3
11	TS007	API para Creación de Campaña	Como desarrollador, necesito crear un endpoint que permita la creación de campañas, de modo que se puedan iniciar campañas de engorde en la aplicación.	2

Orden	User Story	Título	Descripción	Story Points
	ID			(1/2/3)
12	TS008	API para Asociar Empleados a Campaña	Como desarrollador, necesito un endpoint para asociar empleados a campañas, permitiendo la asignación de personal a cada campaña desde la aplicación.	2
13	US005	Buscar Información de Bovinos	Como usuario, quiero poder buscar animales registrados para acceder de forma rápida y ordenada a la información necesaria.	2
14	TS009	API para Gestión de Campañas	Como desarrollador, necesito implementar endpoints que permitan editar y eliminar campañas, facilitando su gestión integral desde la plataforma.	2
15	TS010	API para Registro de Veterinarios	Como desarrollador, necesito exponer un endpoint que permita registrar veterinarios en la aplicación, para que luego puedan ser contactados por los ganaderos.	2
16	US006	Actualizar Información de Bovinos	Como usuario, quiero gestionar la información de los animales registrados para mantener la base de datos actualizada y precisa.	1
17	TS011	API para Búsqueda de Veterinarios	Como desarrollador, necesito un endpoint que permita buscar veterinarios mediante criterios específicos, para brindar opciones de contacto efectivas.	1
18	TS012	API para Registro de Empleados	Como desarrollador, necesito crear un endpoint para registrar empleados, asegurando que la información de cada uno se almacene correctamente y se confirme la creación.	1
19	US07	Crear Campaña para Engorde de Ganado	Como usuario de la plataforma, quiero crear una campaña para engordar el ganado asignado, definiendo parámetros como duración, objetivo y selección de animales o establos.	1
20	TS013	API para Búsqueda de Empleados	Como desarrollador, necesito implementar un endpoint que permita buscar empleados utilizando filtros específicos, para facilitar la administración de los recursos humanos.	1
21	TS014	API para Gestión de Empleados	Como desarrollador, necesito implementar funcionalidades para editar y eliminar empleados mediante la API, permitiendo que el empresario administre correctamente su plantilla.	1

Capítulo IV: Product Design

4.1. Style Guidelines

Las Style Guidelines son esenciales para asegurar una comunicación coherente y profesional en todos los aspectos visuales y de diseño del proyecto, ya sea en medios impresos, digitales o cualquier otra plataforma de difusión. En esta sección, se establecerán las directrices que guiarán al equipo en la creación de VacApp. Estas pautas definirán elementos clave como la elección de colores, tipografía, estructura de documentos y otros aspectos visuales. Para el desarrollo de VacApp, utilizaremos Figma como herramienta principal para diseñar tanto la aplicación móvil como la landing page. En ambos casos, se implementará una paleta de colores basada en tonos verdes y cremas, evocando la naturaleza y transmitiendo la confianza asociada con una gestión responsable y sostenible del cuidado de animales. A continuación, se detalla cada uno de estos aspectos en profundidad.

4.1.1. General Style Guidelines

Branding

El branding de VacApp está diseñado para transmitir confianza, solidez y un compromiso con la sostenibilidad en la ganadería bovina. La identidad visual refleja una conexión directa con la naturaleza y la productividad del campo, utilizando elementos gráficos que representan el cuidado responsable del ganado. Se buscará una imagen fuerte y clara, que sea fácilmente reconocible por los productores y profesionales del sector.

Typography

La tipografía elegida para VacApp es moderna y clara, con un enfoque en la legibilidad, especialmente en pantallas móviles. Se utilizará una fuente Rokkitt para los encabezados, destacando la jerarquía visual de la información, y una tipografía Mulish para el cuerpo de texto, lo que asegura una lectura fácil y cómoda durante el uso prolongado. La tipografía debe reflejar seriedad, sin perder cercanía ni accesibilidad.

Rokkitt

The quick brown fox jumps over the lazy dog
Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm
Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz
1234567890 (.,!/?#\$%&*^@::)

Penultimate

The spirit is willing but the flesh is weak
SCHADENFREUDE
3964 Elm Street and 1370 Rt. 21
<https://fonts-online.ru> info@fonts-online.ru

Mulish

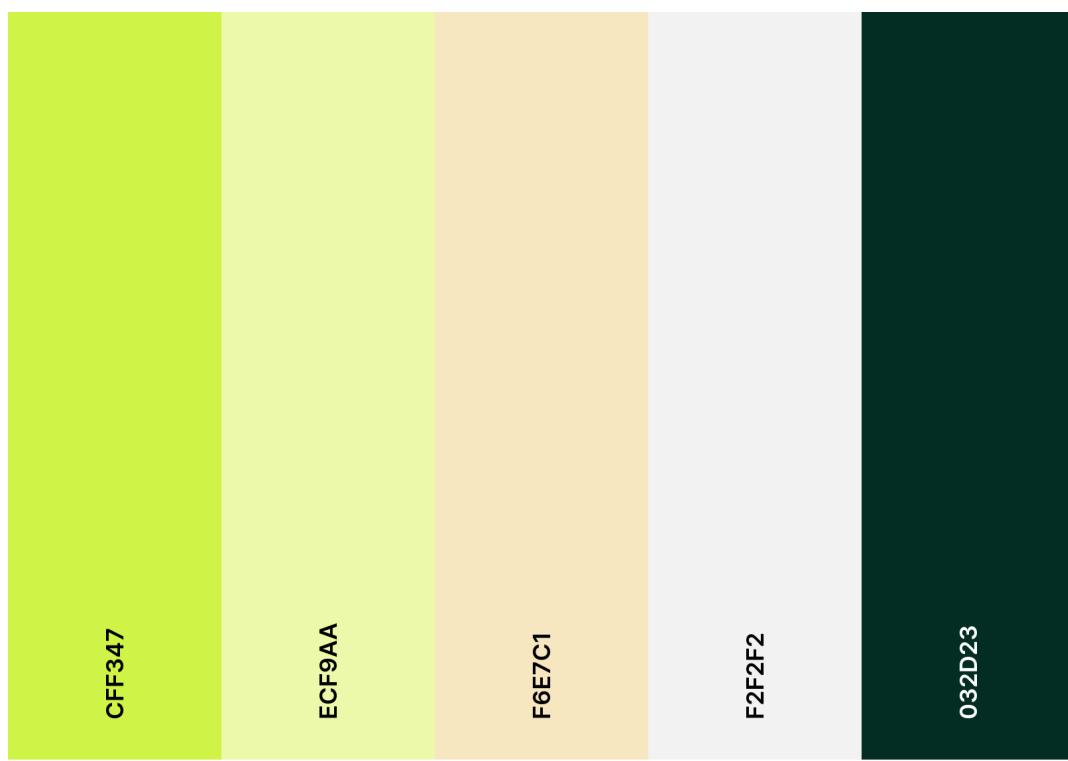
The quick brown fox jumps over the lazy dog
Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm
Nn Oo Pp Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz
1234567890 (.,!/?#\$%&*^@::)

Penultimate

The spirit is willing but the flesh is weak
SCHADENFREUDE
3964 Elm Street and 1370 Rt. 21
<https://fonts-online.ru> info@fonts-online.ru

Colors

La paleta de colores de VacApp está compuesta por tonos verdes y cremas, los cuales se seleccionaron para evocar la naturaleza y la confianza en la gestión sostenible de animales. Los verdes representan frescura, salud y sostenibilidad, mientras que los tonos crema refuerzan la sensación de conexión con la tierra y la tradición del campo. Estos colores se emplearán de manera estratégica para crear una interfaz armónica y accesible en dispositivos móviles.



VacApp

coolors

Spacing

Se aplicará un espaciado adecuado en toda la interfaz para asegurar que los elementos no estén sobrecargados, garantizando una navegación fluida y cómoda. Los márgenes y los espacios entre los componentes estarán cuidadosamente diseñados para ofrecer un diseño equilibrado y organizado, lo que también facilitará la interacción en dispositivos móviles, donde la precisión es crucial.

Tono de Comunicación

El tono de comunicación de VacApp será informativo, respetuoso y cercano, con un enfoque que hable directamente al usuario del sector ganadero bovino. Utilizaremos un lenguaje claro y profesional, pero accesible, para transmitir confianza y conocimiento en temas relacionados con el manejo de ganado. El objetivo es que el usuario se sienta apoyado y bien informado, mientras mantiene la seriedad y la responsabilidad que caracteriza al sector.

4.1.2. Web Style Guidelines

El diseño web de VacApp seguirá principios de **accesibilidad, usabilidad y consistencia visual**, asegurando que la experiencia del usuario sea clara y fluida en distintos dispositivos (desktop, tablet y móvil).

Colores

Se utilizará la misma paleta de colores definida para la identidad de VacApp, garantizando coherencia con la aplicación móvil.

Nombre	Código HEX	Muestra
Primario	#CFF347	
Secundario	#ECF9AA	
Acento	#F6E7C1	
Fondo Claro	#F2F2F2	
Texto Oscuro	#032D23	

Tipografía

- Encabezados: **Rokkitt** (Google Fonts).
- Cuerpo de texto: **Mulish** (Google Fonts).
- Jerarquía visual clara mediante tamaños escalonados (H1-H6).

Componentes UI

- **Navbar fija** en la parte superior con logo y enlaces principales.
- **Botones primarios** con fondo verde (#CFF347) y texto oscuro.
- **Tarjetas (cards)** para mostrar información de animales, campañas y usuarios.
- **Grillas y tablas** responsivas para reportes y datos.

Interacciones

- **Hover effects** en botones y enlaces.
- **Animaciones ligeras** para transiciones de secciones.

- **Diseño responsive** con mobile-first, adaptando las vistas a pantallas pequeñas.

4.1.3. Mobile Style Guidelines

La aplicación móvil de VacApp está diseñada en **Flutter**, lo que asegura consistencia en colores, tipografía y componentes visuales tanto en **iOS** como en **Android**.

La diferencia radica en la **implementación de las guías de diseño nativas** de cada plataforma.

Colores compartidos (iOS y Android)

Nombre	Código HEX	Muestra
Primario	#CFF347	
Secundario	#ECF9AA	
Acento	#F6E7C1	
Fondo Claro	#F2F2F2	
Texto Oscuro	#032D23	

Tipografía (común en Flutter con adaptaciones nativas)

- iOS: SF Pro Display/Text.
- Android: Roboto.
- En Flutter, se define una jerarquía tipográfica coherente que se adapta automáticamente al sistema operativo.

4.1.3.1. iOS Mobile Style Guidelines

El diseño para la versión iOS de VacApp seguirá las guías de **Human Interface Guidelines (HIG)** de Apple.

Colores

Mismos definidos en la paleta general para consistencia de marca.

Tipografía

- Encabezados: **SF Pro Display**.
- Cuerpo de texto: **SF Pro Text**.

Componentes UI

- **Bottom Navigation Bar** con iconografía clara y minimalista.
- Botones redondeados (12–16 px) en línea con la estética iOS.
- **Modal sheets** para formularios y confirmaciones.

Interacciones

- Gestos nativos: swipe back, pull-to-refresh.
- Animaciones con **blur y transparencia**, dando sensación de profundidad.
- Feedback visual inmediato en cada interacción.

4.1.3.2. Android Mobile Style Guidelines

El diseño para la versión Android de VacApp seguirá las guías de **Material Design 3 (Material You)** de Google.

Colores

Mismos definidos en la paleta general.

Tipografía

- Encabezados: **Roboto Bold**.
- Cuerpo de texto: **Roboto Regular**.

Componentes UI

- **BottomNavigationView o NavigationRail** en tablets.
- Botones con esquinas redondeadas según Material Design 3.
- **Floating Action Button (FAB)** para acciones clave como registrar bovinos o vacunas.

Interacciones

- Animaciones dinámicas basadas en **motion design**.
- Soporte completo para **dark mode**.
- Compatibilidad con el sistema de personalización de Material You.

4.2. Information Architecture

La arquitectura de la información constituye un pilar fundamental para el diseño y la usabilidad de un sistema digital. Su correcta aplicación permite que los usuarios encuentren, comprendan y utilicen el contenido de manera sencilla y eficiente.

4.2.1. Organization Systems

Los sistemas de organización definen la estructura y cómo se clasifican los términos dentro de la aplicación.

- **Jerárquica (Visual Hierarchy)**: Se aplica en la Landing Page y en la pantalla principal de la app, destacando las funciones más importantes como registro de ganado y control sanitario. Las acciones frecuentes se ubican en la parte superior o centradas.
- **Secuencial (Step-by-step)**: Utilizada en procesos como el registro de animales, guiando al usuario en pasos definidos.
- **Por Tópicos**: En las secciones de información técnica, como manuales o ayudas, la organización se basa en temas relevantes (salud animal, nutrición, reproducción, etc.).
- **Según Audiencia**: Algunas vistas como el panel administrativo o el perfil del veterinario muestran información personalizada, según el rol del usuario dentro del sistema.

4.2.2. Labeling Systems

Este punto es el etiquetado que proporciona el nombre y presenta los elementos de información.

Este seria el formato establecido:

- Se evita el uso de jergas técnicas excesivas.
- Las acciones se etiquetan con verbos directos como "Registrar", "Consultar", "Programar".
- Las categorías principales usan términos como "Animales", "Citas", "Inventario", "Diagnósticos".
- Se emplean iconos acompañantes para reforzar visualmente el significado de cada etiqueta.

4.2.3. SEO Tags and Meta Tags

Este punto optimiza la visibilidad del sistema en motores de búsqueda y aporta información acerca del contexto.

Landing Page y Web App:

- **Title:** VacApp – Gestión inteligente para la ganadería bovina
- **Meta Description:** Plataforma digital que optimiza el control sanitario, inventario y manejo del ganado bovino.
- **Keywords:** ganadería, salud animal, veterinaria, bovino, control de ganado, app ganadera
- **Author:** VacApp

App Store Optimization (ASO):

- **App Title:** VacApp
- **App Subtitle:** Manejo digital de tu ganado bovino
- **App Keywords:** ganadería, bovinos, recetas, veterinarios, animales
- **App Description:** VacApp es una aplicación móvil diseñada para modernizar y optimizar la gestión ganadera en el Perú. Pensada tanto para ganaderos independientes como para empresas del sector.

4.2.4. Searching Systems

Los sistemas de búsqueda permiten que los usuarios puedan encontrar la información específica dentro del entorno digital y que agilizan el acceso al contenido de este mismo.

- Búsqueda global por nombre, código de animal o categoría.
- Filtros personalizados por estado de salud, tipo de ganado, fechas de registro, entre otros.
- Sugerencias automáticas mientras se escribe (auto-complete).
- Resultados mostrados con etiquetas claras, iconos e información resumida (como nombre, fecha, categoría).

4.2.5. Navigation Systems

Este último punto establece los sistemas de navegación que utilizarán los usuarios para desplazarse dentro de la aplicación web y móvil.

Web

- Barra de navegación superior (Top Navigation): ubicada de forma fija en la parte superior de la interfaz, ofrece accesos rápidos a las secciones principales como Inicio, Animales, Perfil, entre otras. Esto asegura que el usuario pueda moverse con fluidez entre los apartados más relevantes.

- Menú lateral (Sidebar) expandible: en lugar del menú hamburguesa del entorno móvil, la web incorpora un panel lateral con opciones complementarias como Configuración, Soporte y Cerrar sesión. Este diseño aprovecha mejor el espacio disponible en pantalla y permite un acceso más directo a funciones secundarias.
- Enlaces jerárquicos y breadcrumbs: al igual que en la aplicación móvil, se integran rutas de navegación jerárquicas que facilitan al usuario avanzar o retroceder entre secciones sin perder el contexto. Los breadcrumbs se presentan en la parte superior de la interfaz para reforzar la claridad de ubicación dentro del sistema.
- Flujos de usuario optimizados: los procesos complejos, como el registro de un animal, se estructuran en pasos secuenciales y guiados. Esto reduce la carga cognitiva y asegura que el usuario complete cada tarea sin dificultad.
- Indicadores visuales y estados activos: la navegación incluye resaltados en los elementos activos del menú, junto con indicadores visuales que muestran la sección en la que el usuario se encuentra. Esto fomenta la orientación continua y la coherencia en la interacción.

Móvil:

- Navegación inferior (Bottom Navigation) con accesos rápidos a secciones clave: Inicio, Animales, Perfil, etc.
- Menú hamburguesa con opciones complementarias como configuración, soporte y cerrar sesión.
- Enlaces jerárquicos que permiten ir y volver entre secciones sin perder contexto.
- Flujos de usuario optimizados: pasos guiados para tareas como registrar un animal.
- Indicadores visuales (breadcrumbs, estados activos) para mantener claridad sobre la ubicación actual dentro de la app.

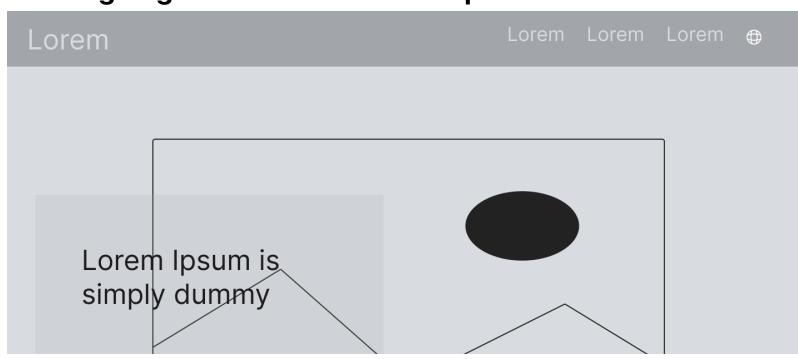
4.3. Landing Page UI Design

El diseño de la **Landing Page** de VacApp tiene como objetivo principal transmitir de manera clara y atractiva la propuesta de valor de la aplicación. Para ello, se han elaborado **wireframes** y **mock-ups** tanto en versión **desktop** como **mobile**, asegurando coherencia visual, accesibilidad y adaptabilidad en diferentes dispositivos.

4.3.1. Landing Page Wireframe

Los **wireframes** representan la estructura inicial de la Landing Page, enfocándose en la disposición de los elementos clave sin incluir detalles visuales avanzados. Sirven como base para validar la jerarquía de la información, la navegación y la organización de los componentes principales.

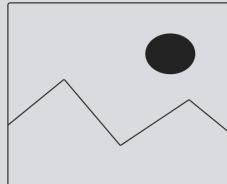
- **Landing Page Wireframe – Desktop**





Lorem Ipsum

Lorem Ipsum is simply dummy text of the printing and typesetting industry.



Lorem Ipsum

Lorem Ipsum



Lorem



Lorem



Lorem



Lorem



Lorem



Lorem

Lorem Ipsum

Lorem

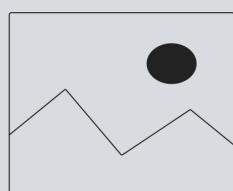
S/0

- lorem
- lorem
- lorem
- lorem
- lorem

Lorem

**Lorum
S/49.90**

- lorem
- lorem
- lorem
- lorem
- lorem



Lorem Ipsum

Lorem Ipsum is simply dummy text of the printing and typesetting industry.



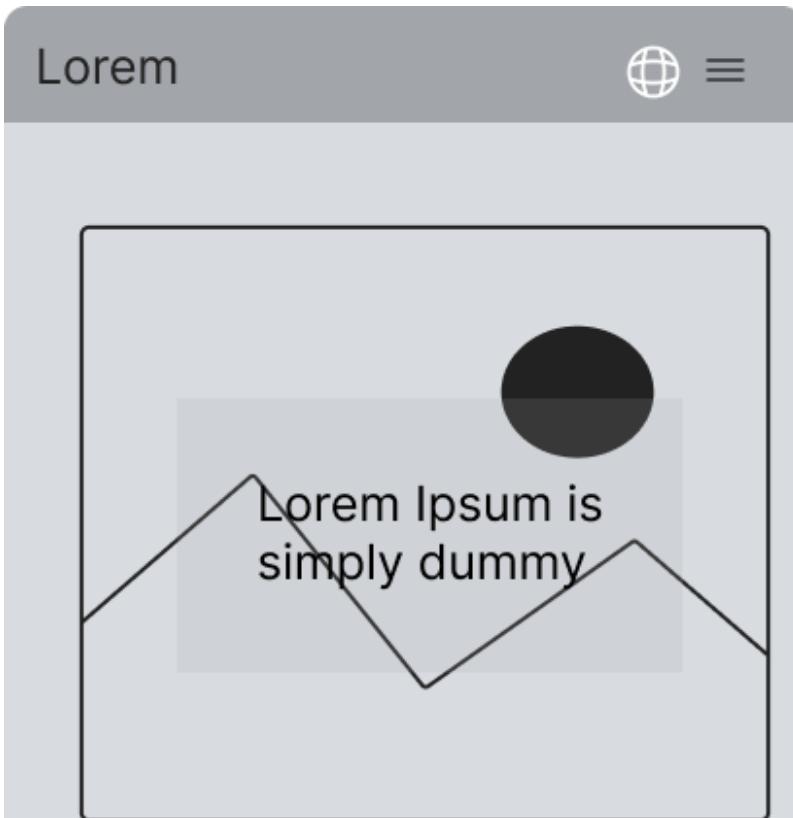
Lorem



Lorem

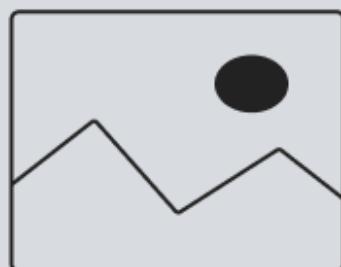
@lorem ipsum

- Landing Page Wireframe – Mobile



Lorem Ipsum

 Lorem Ipsum is simply dummy



 Lorem Ipsum

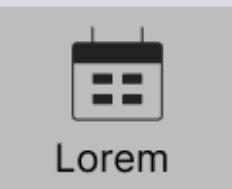
Lorem Ipsum



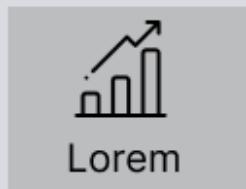
 Lorem



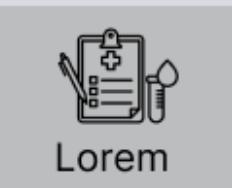
 Lorem



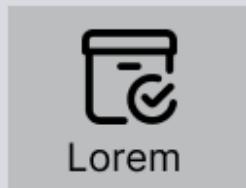
Lorem



Lorem



Lorem



Lorem

Lorerem** Ipsum**

Lorerem****

S/0

- lorem ipsum

Lorerem****

Lorerem****

S/49.90

- lorem ipsum

Lorerem** Ipsum**

Lore**rem** Ipsum is simply dummy

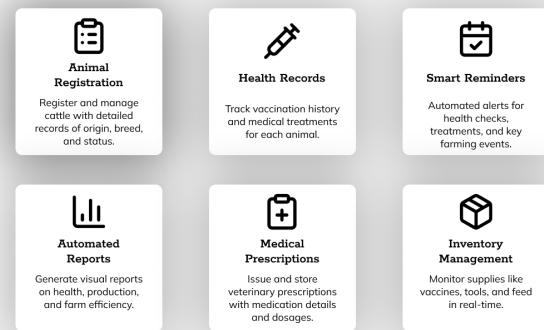


4.3.2. Landing Page Mock-up

Los **mock-ups** representan la versión más cercana al diseño final de la Landing Page. Incorporan la **paleta de colores oficial de VacApp**, la **tipografía definida en las Style Guidelines**, y los elementos gráficos que refuerzan la identidad visual de la aplicación. Estos prototipos permiten visualizar cómo será la experiencia del usuario en el producto final.

- **Landing Page Mock-up – Desktop**

A desktop landing page for VacApp. The header features a dark green bar with the VacApp logo and navigation links for About Us, Features, Prices, Get App, and a user icon. The main visual is a photograph of cows grazing in a field with rolling hills in the background. Overlaid on the image is a dark green callout box containing the text "Revolucionando la agricultura con tecnología". A small "Empezar" button is visible at the bottom left of the image. Below the image, the section title "About Us" is displayed in bold black font. To the left of the title is a paragraph of text describing VacApp's mission and how it aids farmers and veterinarians. To the right is a photograph of a man in a stable, holding a hat, standing next to a horse. A tagline at the bottom states "Conectamos la tradición ganadera con soluciones tecnológicas." The footer features a large "Key Features" section title.



Prices

Free	Premium
S/0	Desde S/49.90
<ul style="list-style-type: none"> • Registro de hasta 10 animales. • Acceso básico al histórico sanitario. • Recordatorios limitados. • Visualización de reportes mensuales en formato simple. • Soporte por correo electrónico. 	<ul style="list-style-type: none"> • Registro ilimitado de animales. • Seguimiento completo de salud y vacunación. • Gestión de inventario y productos. • Soporte prioritario.

Try our app



Discover a smarter way to manage your farm. Try our mobile app and enjoy all features at your fingertips — anytime, anywhere.



© 2025 VacApp. Todos los derechos reservados.

- Landing Page Mock-up – Mobile



VacApp es una aplicación móvil creada para ganaderos y

creada para ganaderos y veterinarios del Perú. Nuestra misión es facilitar la gestión ganadera mediante una plataforma accesible, incluso sin experiencia previa en tecnología. Desde el registro de animales hasta la generación de reportes, VacApp está hecha para ti.



Conectamos la tradición ganadera con soluciones tecnológicas.

Key Features



Animal Registration

Register and manage cattle with detailed records of origin, breed, and status.



Health Records

Track vaccination history and medical treatments for each animal.



Smart Reminders

Automated alerts for health checks, treatments, and key farming events.



Automated Reports

Generate visual reports on health, production, and farm efficiency.



Medical



Inventory

Prescriptions

Issue and store veterinary prescriptions with medication details and dosages.

Management

Monitor supplies like vaccines, tools, and feed in real-time.

Prices

Free**S/0**

- Registro de hasta 10 animales.
- Acceso básico al historial sanitario.
- Recordatorios limitados.
- Visualización de reportes mensuales en formato simple.
- Soporte por correo electrónico.

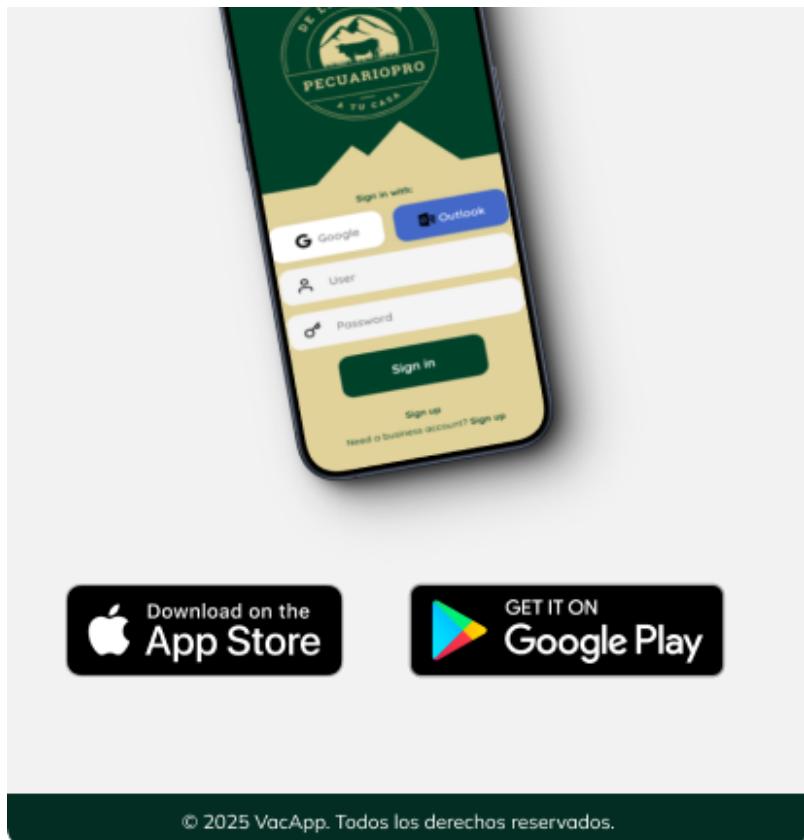
Premium**Desde****S/49.90**

- Registro ilimitado de animales.
- Seguimiento completo de salud y vacunación.
- Gestión de inventario y productos.
- Soporte prioritario.

Try our app

Discover a smarter way to manage your farm. Try our mobile app and enjoy all features at your fingertips — anytime, anywhere.





© 2025 VacApp. Todos los derechos reservados.

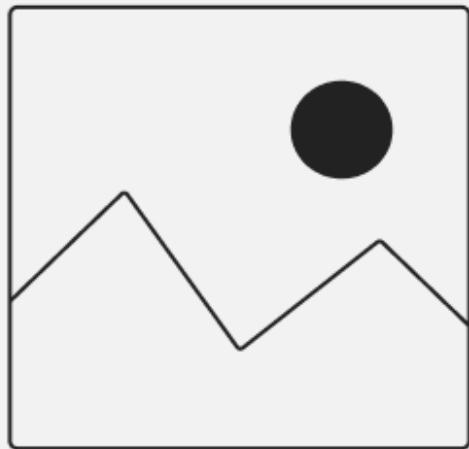
El desarrollo de estos diseños se realizó en **Figma**, lo que garantiza un entorno colaborativo y dinámico para el equipo de diseño y desarrollo.

Figma Project: <https://www.figma.com/design/Ck5RdO3MzAm16SIReLDO15/Sin-t%C3%ADulo?node-id=150-5796&t=hGN3YL7RfASQ5FFk-1>

4.4. Mobile Applications UX/UI Design

4.4.1. Mobile Applications Wireframes

Inicio Sesión



Lorem

G Lorem

O Lorem

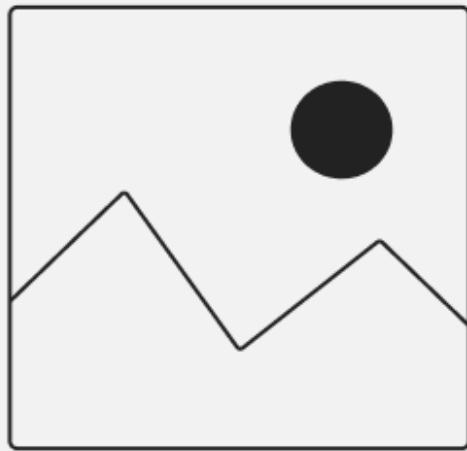
👤 Lorem

⌚ Lorem

Lorem ipsum

>Lorem

Registro



Lorem

G Lorem

O Lorem

👤 Lorem

🗣️ Lorem

✉️ Lorem

Lorem

Lorem ipsum

Planes



Lorem

Lorem

S/0

- lorem ipsum

Lorem

Lorem

Lorem

S/49.90

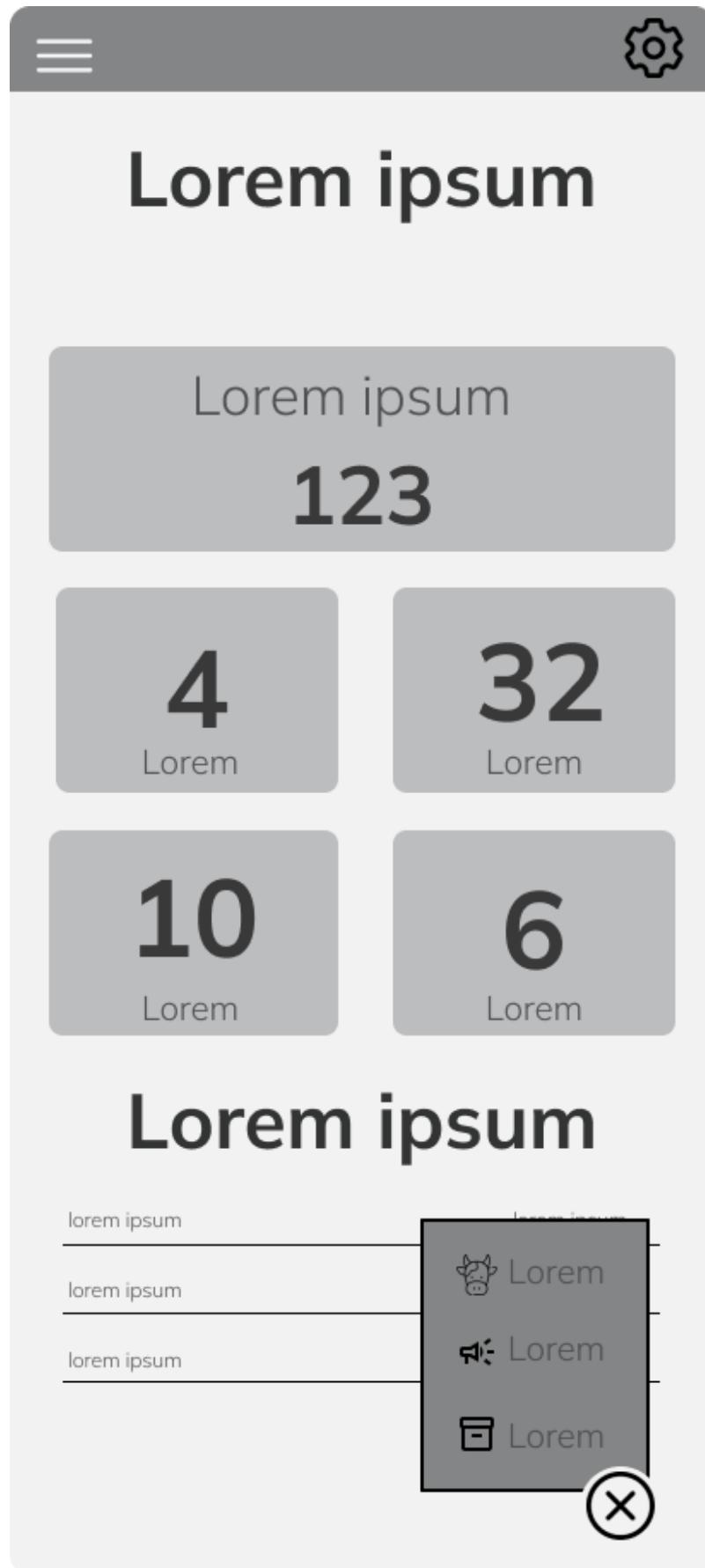
- lorem ipsum

Lorem

Home







Animals

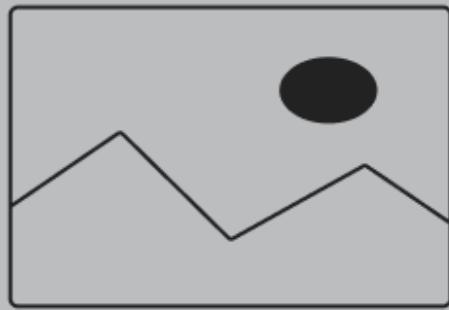
The image shows a mobile application interface with a dark grey header bar at the top containing a menu icon (three horizontal lines) on the left. Below the header are three identical-looking cards, each representing a female user profile. Each card has a light grey background and contains the following elements from top to bottom:

- A title "Lorem ♀" where "Lorem" is in bold.
- A small square placeholder image with a wavy pattern and a central black dot.
- A "More" button represented by the word "Lorem" followed by a downward arrow.
- The text "Lorem ipsum" repeated twice.
- A "Details" button represented by the word "Lorem" followed by a circular icon with an "i".

The image shows a mobile application interface with a dark grey header bar at the top containing a menu icon (three horizontal lines) on the left. Below the header is a single card, similar in structure to the ones above, representing a female user profile. It contains the following elements from top to bottom:

- A title "Lorem ♀" where "Lorem" is in bold.
- A small square placeholder image with a wavy pattern and a central black dot.
- A "More" button represented by the word "Lorem" followed by a downward arrow.
- The text "Lorem ipsum" repeated twice.
- A "Details" button represented by the word "Lorem" followed by a circular icon with an "i".

At the very bottom of the screen is a light grey footer bar containing a back arrow icon on the left and the text "Lorem ♀" in a large, bold font on the right.



Lorem

Lorem ipsum

Lorem

Lorem

Lorem

Lorem

Birdthdate

00/00/0000

Lorem

Lorem
ipsum

Lorem

Lorem

Lorem

Lorem ipsum (00/00/0000)

Lorem ipsum (00/00/0000)

Inventario

The image shows a mobile application interface with a light gray background. At the top left is a menu icon (three horizontal lines). Below it is a back arrow icon.

Card 1: Contains a checkmark icon above a stack of boxes icon, followed by the text "Lorem ipsum". To the right is the number "850" and a small circular info icon.

Card 2: Contains a warning sign icon, followed by the text "Lorem ipsum" and the number "5". Below this are three horizontal rows of text and numbers: "Lorem ipsum 02", "Lorem ipsum 05", and "Lorem ipsum 13".

Card 3: Contains a calendar icon, followed by the text "Lorem ipsum" and the number "3". Below this are three horizontal rows of text and words: "Lorem ipsum Lorem", "Lorem ipsum Lorem", and "Lorem ipsum Lorem".

The image shows a mobile application interface with a light gray background. At the top left is a menu icon (three horizontal lines). Below it is a back arrow icon.

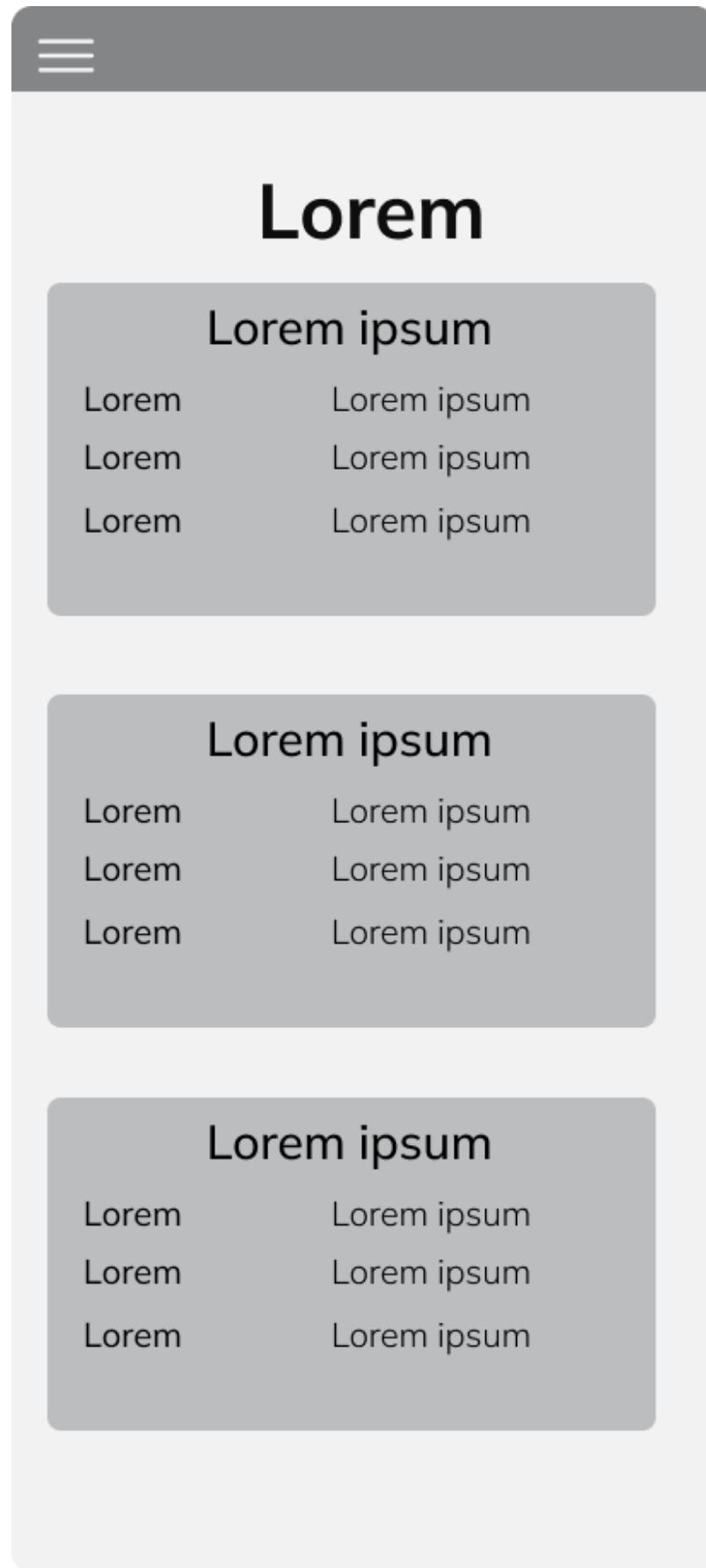
Card: Contains the text "Lorem ipsum" in large bold letters.



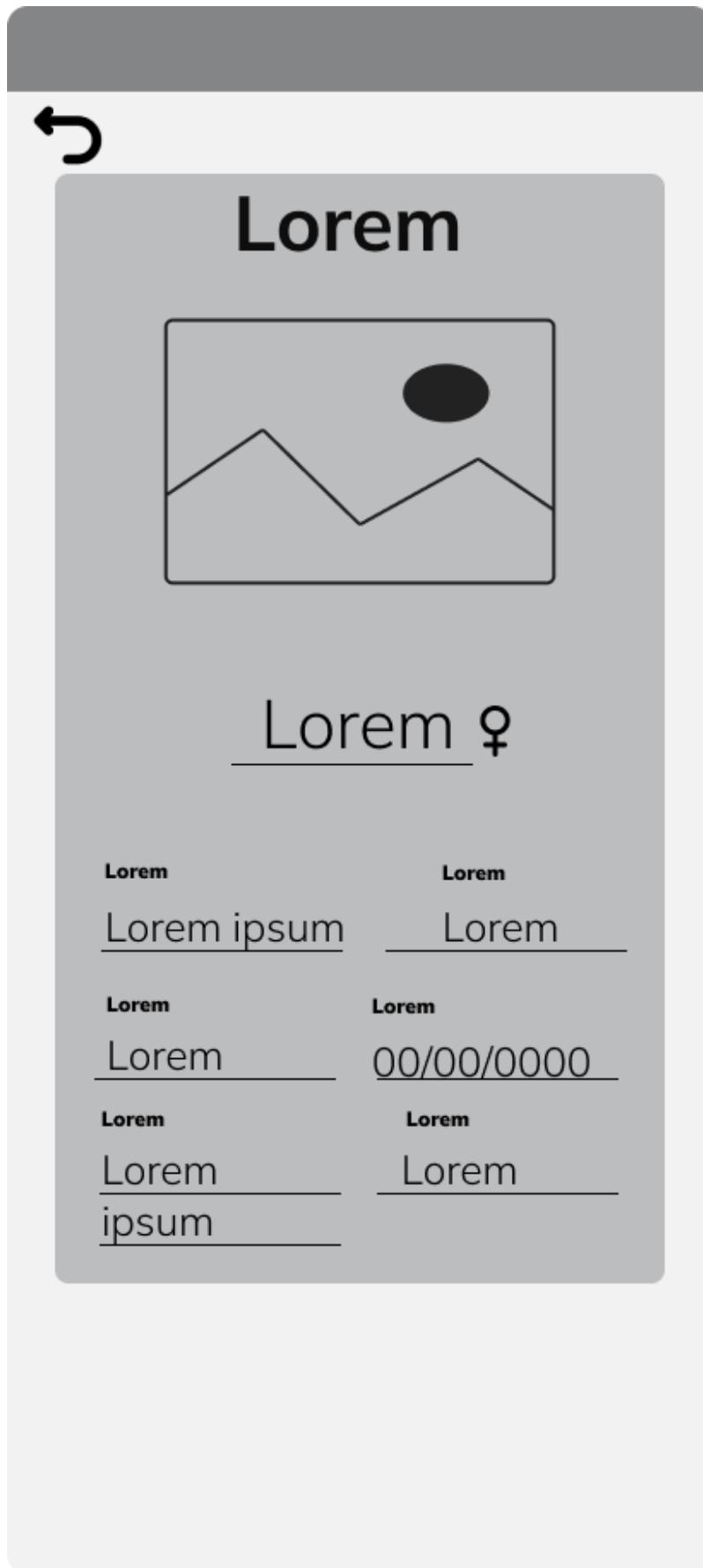
Light Gray Lorem Light Brown Lorem Medium Gray Lorem Dark Gray Lorem

Lorem	Lorem	Lorem
Lorem	Lorem	02
Lorem	Lorem	05
Lorem	Lorem	13
Lorem	Lorem	02
Lorem	Lorem	05
Lorem	Lorem	13
Lorem	Lorem	
Lorem	Lorem	

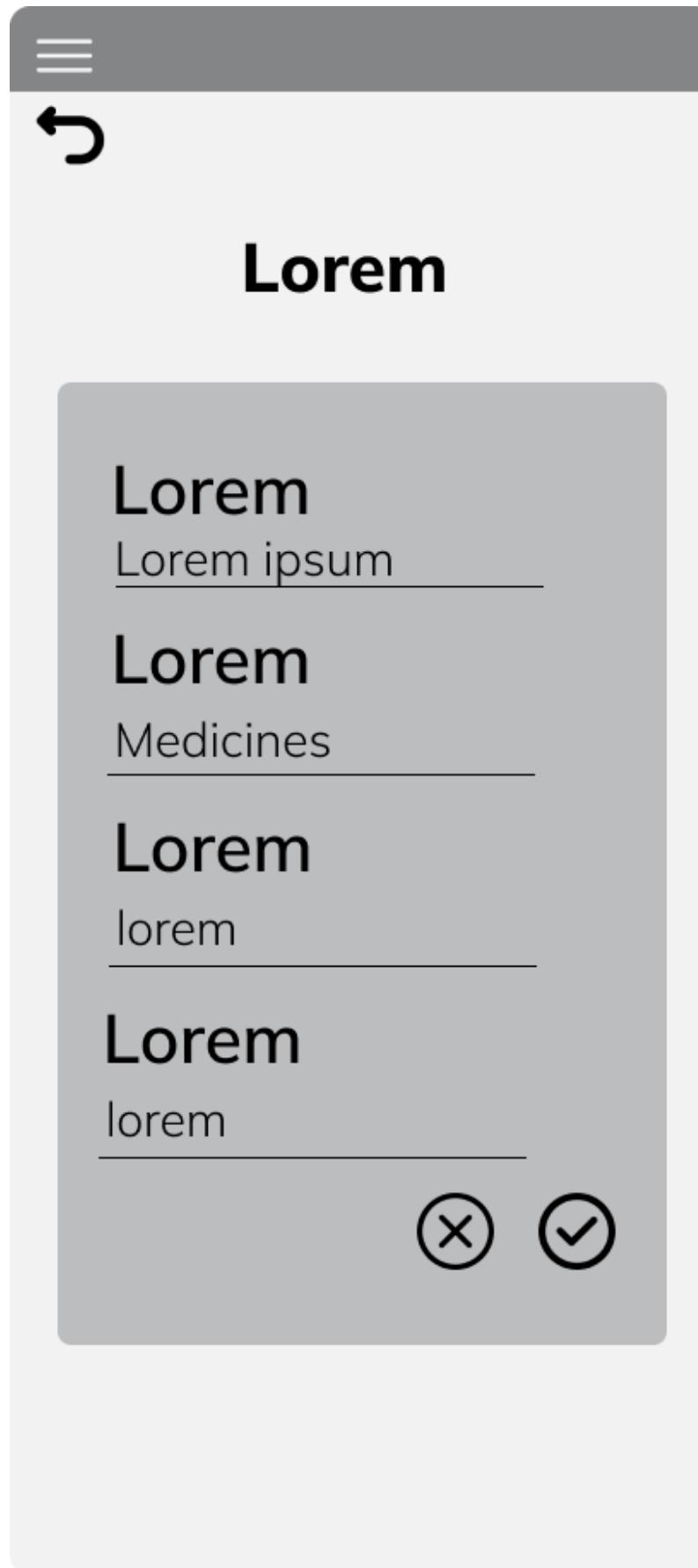
Campaña



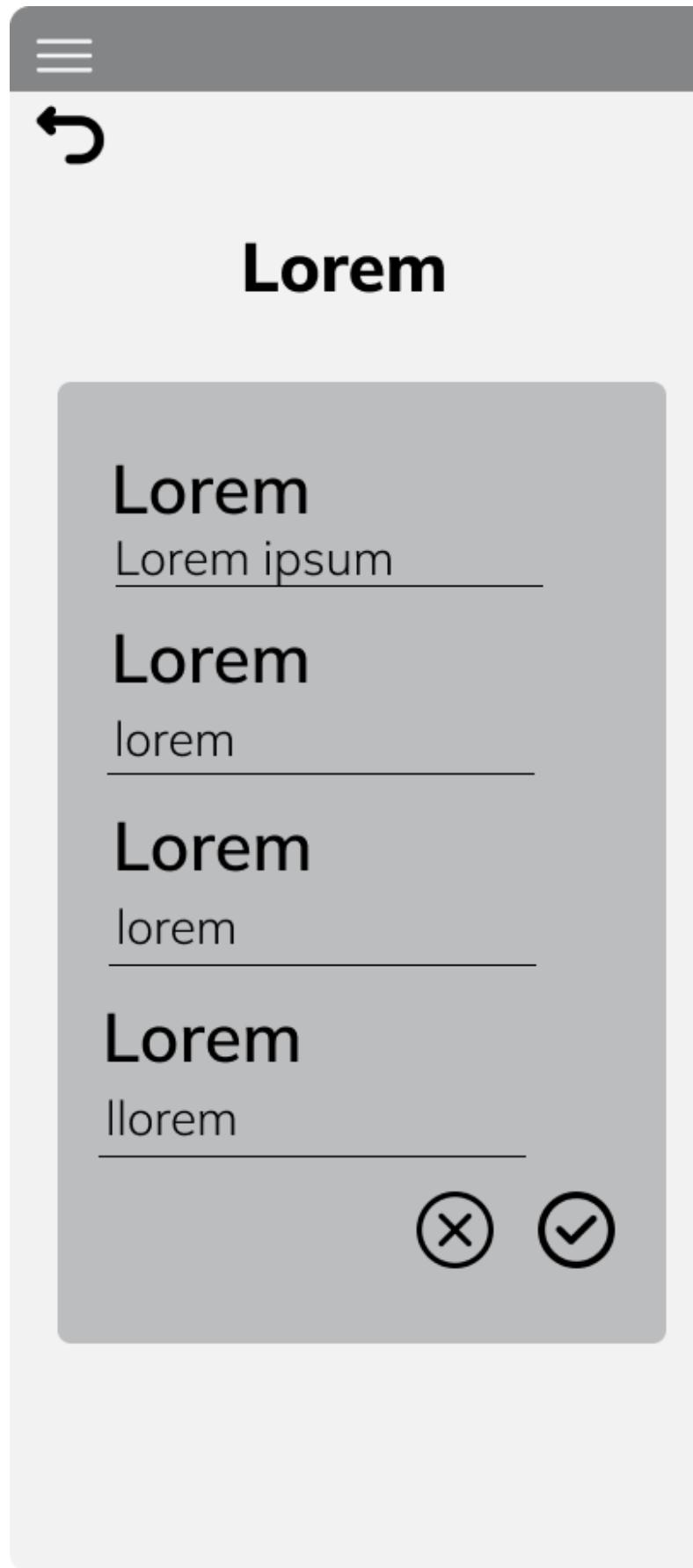
vista de "Agregar Animal"



vista de "Aregar Inventario"



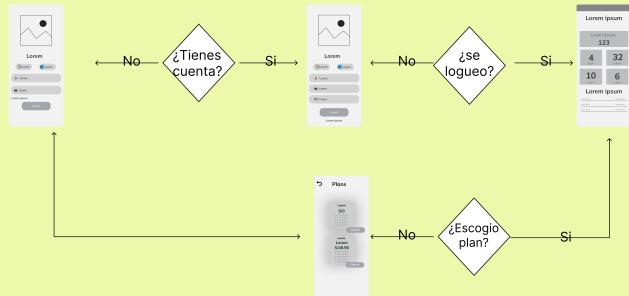
vista de "Agregar Campaña"



4.4.2. Mobile Applications Wireflow Diagrams

User Goal: Iniciar sesión y Registrarse: Este flujo guía al usuario desde la pantalla de bienvenida hacia las opciones de autenticación.

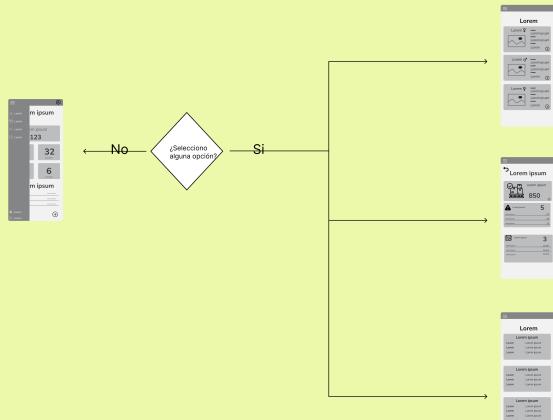
User Goal: Iniciar sesión y Registrarse



User Goal: Home y navegar por las secciones "Animals", "Campaigns" e "Inventory"

Una vez autenticado, el usuario accede al home con accesos rápidos con el sidebar. Este wireflow muestra cómo el usuario puede visualizar las opciones "Animals", "Campaigns e "inventory"

User Goal: navegar por las secciones “Animals”, “Campaigns” e “Inventory”



User Goal: Registro de un nuevo dato ya sea "Animals", "Campaigns" o "Inventory":

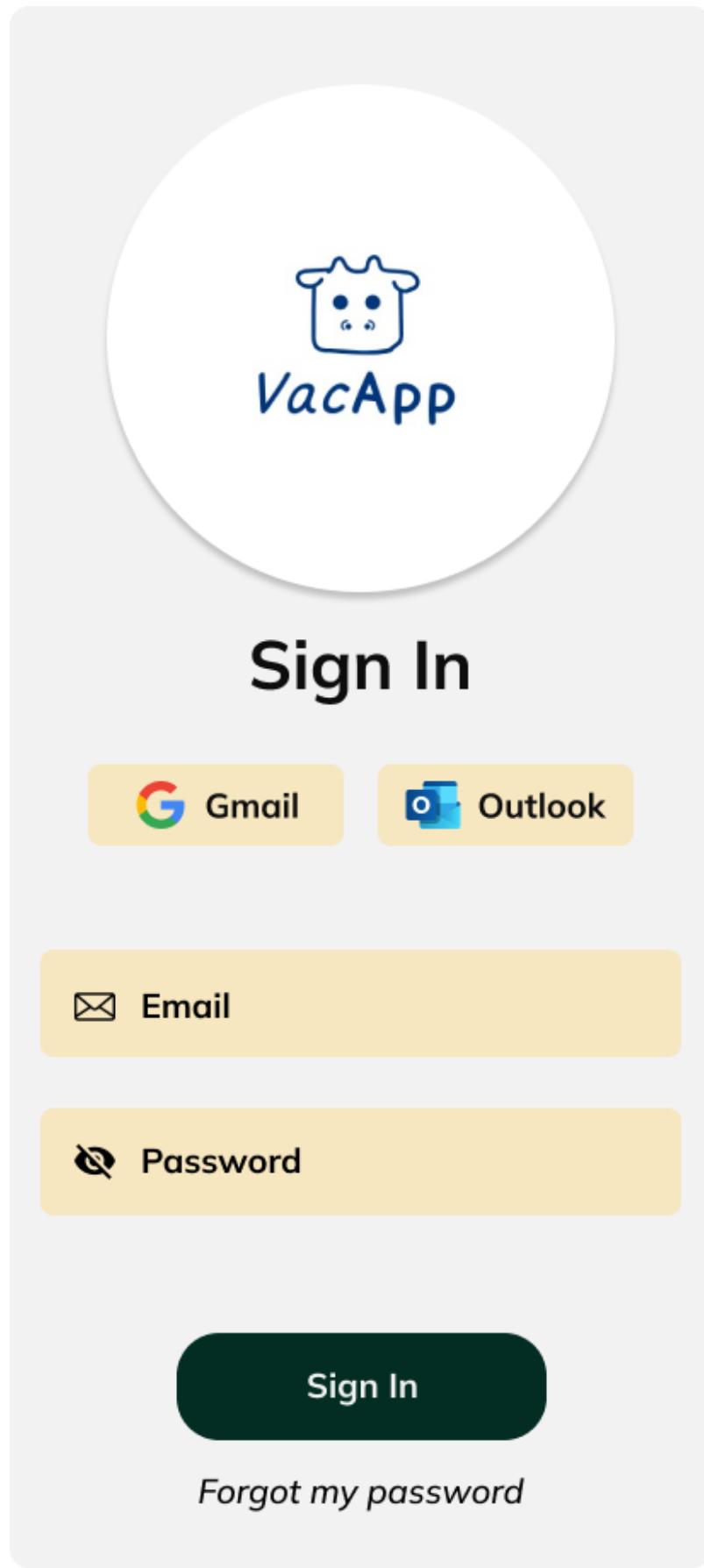
Este flujo está diseñado para facilitar al agregar ya sea un bovino, una campaña o un producto de inventario.

User Goal: Registro de un nuevo dato ya sea “animal”, “campaign” o “inventory”

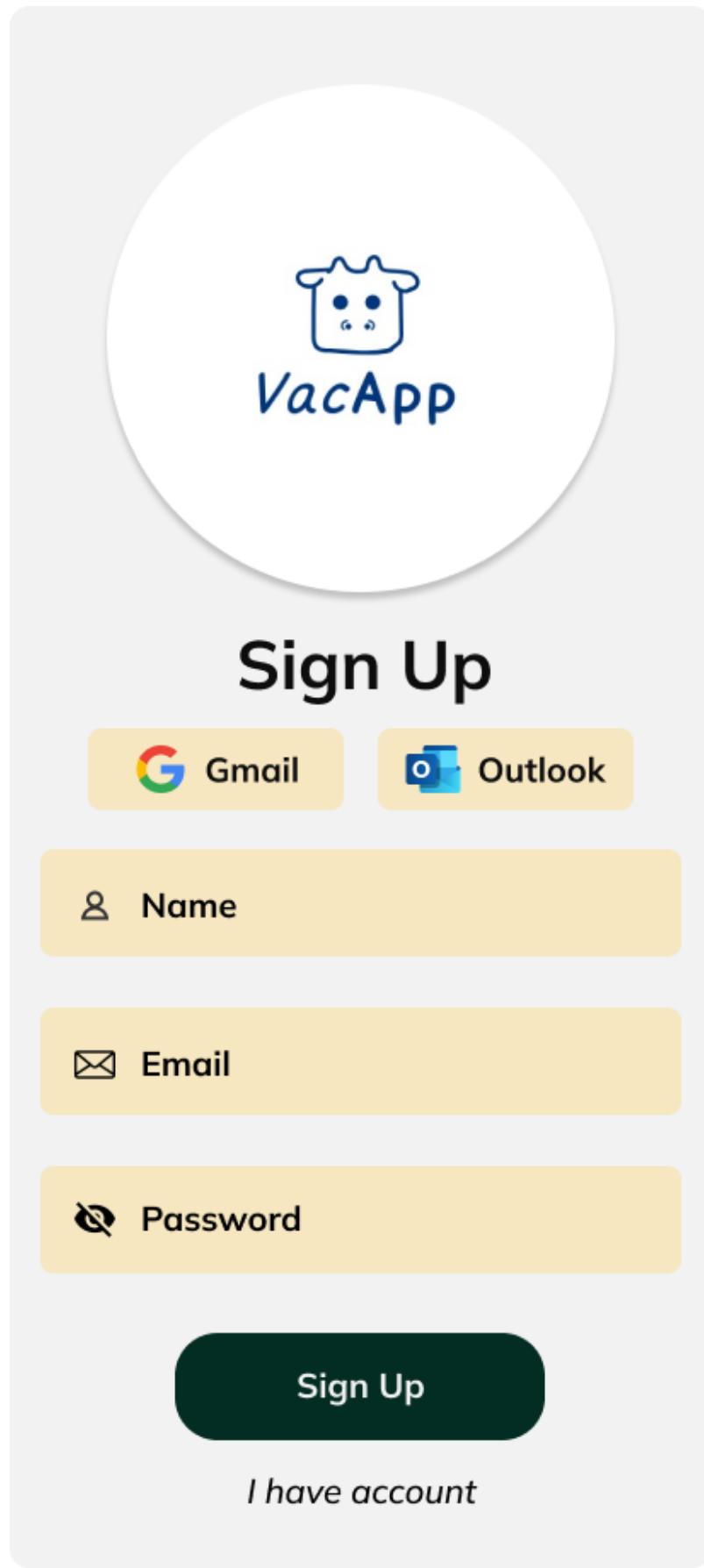


4.4.3. Mobile Applications Mock-ups

Inicio Sesión



[Registro](#)



Planes



Plans

Free

S/0

- Registro de hasta 10 animales.
- Acceso básico al historial sanitario.
- Recordatorios limitados.
- Visualización de reportes mensuales en formato simple.
- Soporte por correo electrónico.

Choose

Premium

Desde

S/49.90

- Registro ilimitado de animales.
- Seguimiento completo de salud y vacunación.
- Gestión de inventario y productos.
- Soporte prioritario.

Choose

Home

The dashboard displays the following key metrics:

- Registered animals: 123
- Campaigns: 4 (Campaigns)
- Employees: 32
- Vaccines about to expire: 10
- Campaigns: 6 (Campaigns)

Upcoming Events

Foot and Mouth Disease Vaccination	10-May
Internal and External Deworming	05-July
Brucellosis Sanitation Campaign	23-August

A large plus sign (+) is located at the bottom right of the main content area.

The footer navigation bar includes:

- Home (indicated by a house icon)
- Log Out (indicated by a person icon)

The word "come" is partially visible on the right side of the footer.

Animals

Campaigns

Inventory

Settings

Log out

San Jose!

Number of animals: 123

Number of employees: 32

Number of campaigns: 6

Upcoming Events

Vaccination	10-May
Training	05-July
Charity Campaign	23-August

+

The dashboard features a dark green header bar with a white three-line menu icon on the left. Below the header, a large title 'Welcome Juan Jose!' is displayed in bold, dark green font. A yellow rounded rectangle contains the text 'Registered animals' above the number '123'. To the left of this box is a yellow box with '4 Campaigns' and 'Campaigns' below it. To the right is another yellow box with '32 Employees' and 'Employees' below it. Below these are two more yellow boxes: one with '10 Vaccines about to expire' and 'Vaccines about to expire' below it, and another with '6 Campaigns' and 'Campaigns' below it. At the bottom, a section titled 'Upcoming Events' lists three items: 'Foot and Mouth Disease Vaccination' (with a date '10-May'), 'Internal and External Deworming' (with a date '10-May'), and 'Brucellosis Sanitation Campaign' (with a date '10-May'). A modal window is overlaid on the dashboard, containing icons for 'Animal' (cow), 'Campaign' (calendar), and 'Inventory' (box), each with a small circular arrow icon. A close button (an 'X') is located at the bottom right of the modal.

Welcome Juan Jose!

Registered animals

123

4 Campaigns

32 Employees

10 Vaccines about to expire

6 Campaigns

Upcoming Events

Foot and Mouth Disease Vaccination (10-May)

Internal and External Deworming (10-May)

Brucellosis Sanitation Campaign (10-May)

Animal

Campaign

Inventory

Animals



Animals

Gloria ♀



Campaign

Vacas locas

Peso

510 kg

Edad

4 años



Motomoto ♂



Campaign

Vacas locas

Peso

510 kg

Edad

4 años



Rebeca ♀



Campaign

Vacas locas

Peso

510 kg

Edad

4 años



Gloria ♀

**Breed**

Gelbvieh

Weight

510 kg

Age

4 age

Birdthdate

12/01/2021

Barn

La Bendición

Location

Chorrillos

Inventario

The image shows a mobile application interface for inventory management. At the top left is a menu icon (three horizontal lines). At the top right is a back arrow icon.

Inventory

Stored products **850** (i)

Products with low stock **5**

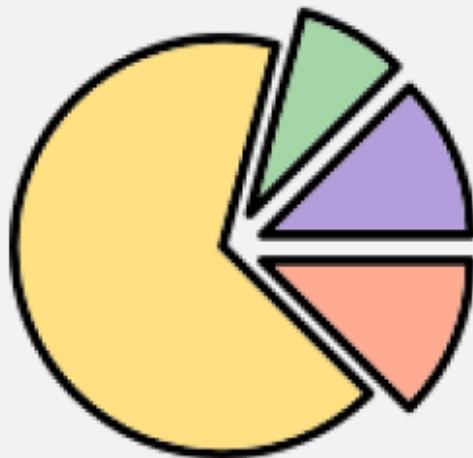
Fiebre Aftosa (Oleosa)	02
Oxitetraciclina	05
Brucelosis (RB51)	13

Products to expire **5**

Ketoprofeno	18-June
Ivermectina 1%	05-July
Clostridiales (8 vías)	23-July

The image shows a mobile application interface for product management. At the top left is a menu icon (three horizontal lines). At the top right is a back arrow icon.

Products



Medicines Vaccines Food Suplements

Name	Category	Stock
Fiebre Aftosa (Oleosa)	Medicines	02
Oxitetraciclina	Vaccines	05
Brucelosis (RB51)	Vaccines	13
Fiebre Aftosa (Oleosa)	Food	02
Oxitetraciclina	Suplements	05
Brucelosis (RB51)	Food	13

Campaña



My Campaigns

Bovine Brucellosis

Description: Cattle vaccination

Date: 09/05/2025

End Date: 20/06/2025

Foot-and-Mouth Disease

Description: Cattle vaccination

Date: 09/05/2025

End Date: 20/06/2025

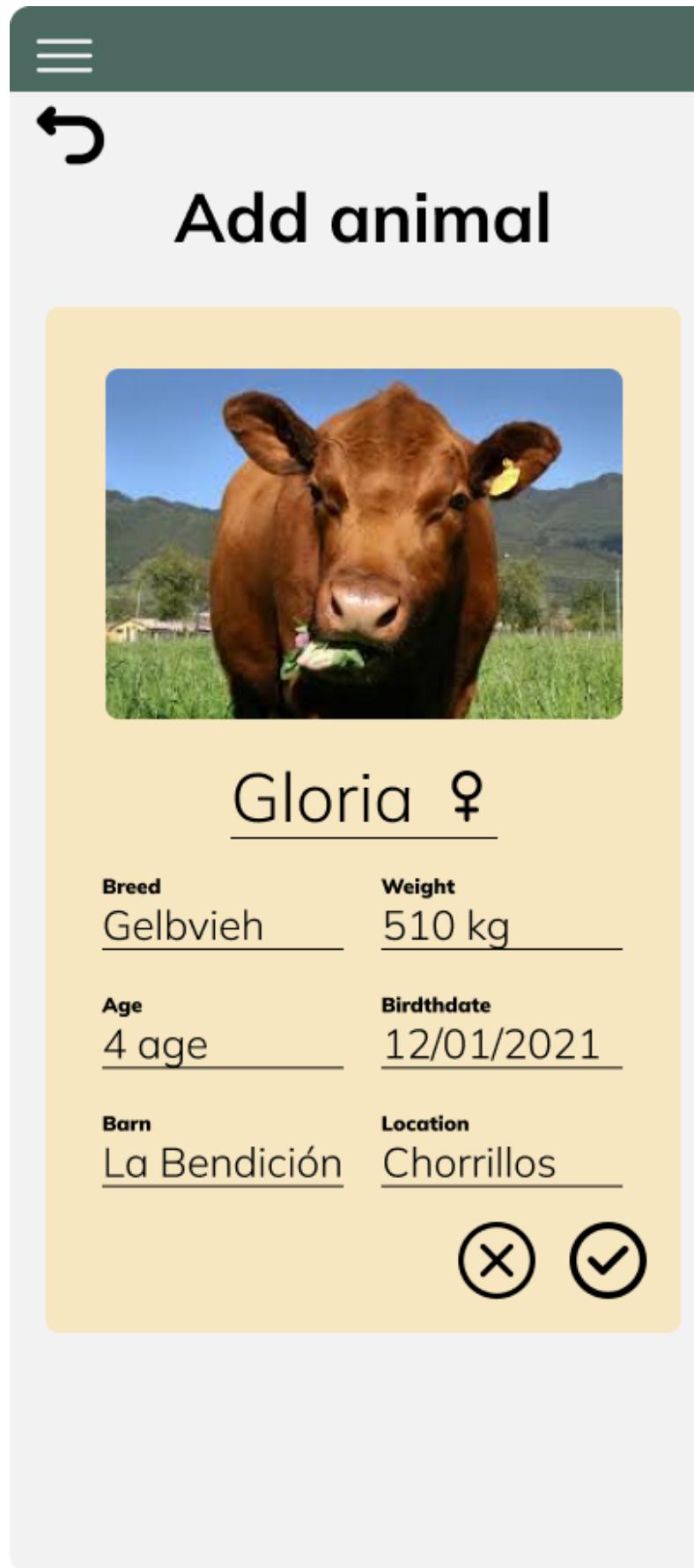
Healthy Herd 2025

Description: Cattle vaccination

Date: 09/05/2025

End Date: 20/06/2025

vista de "Aregar Animal"



vista de "Agregar Inventario"

The screenshot shows a mobile application interface for adding a new product. At the top, there is a dark green header bar with a white back arrow icon on the left and a white three-line menu icon on the right. Below the header, the title "Add product" is displayed in a large, bold, black font. The main content area has a light yellow background and contains four input fields, each with a label and a text input:

- Name**: The input field contains the text "Fiebre Aftosa (Oleosa)".
- Category**: The input field contains the text "Medicines".
- Stock**: The input field contains the text "32".
- Expiration date**: The input field contains the text "15/07/2026".

At the bottom of the yellow section, there are two circular icons: a black one with a white "X" and a black one with a white checkmark.

vista de "Aregar Campaña"

The image shows a smartphone screen displaying a mobile application for managing campaigns. The top navigation bar is dark green with a white back arrow icon and a three-line menu icon. The main title "Add campaign" is centered at the top in a large, bold, black font. Below the title is a yellow rectangular input field containing the campaign details. The first section is labeled "Name" in bold black text, followed by the value "Healthy Herd 2025" in a standard black font. The second section is labeled "Description" in bold black text, followed by the value "Cattle vaccination". The third section is labeled "Start date" in bold black text, followed by the value "09/05/2025". The fourth section is labeled "End date" in bold black text, followed by the value "20/06/2025". At the bottom right of the yellow field are two circular icons: a black one with a white "X" and a white one with a black checkmark.

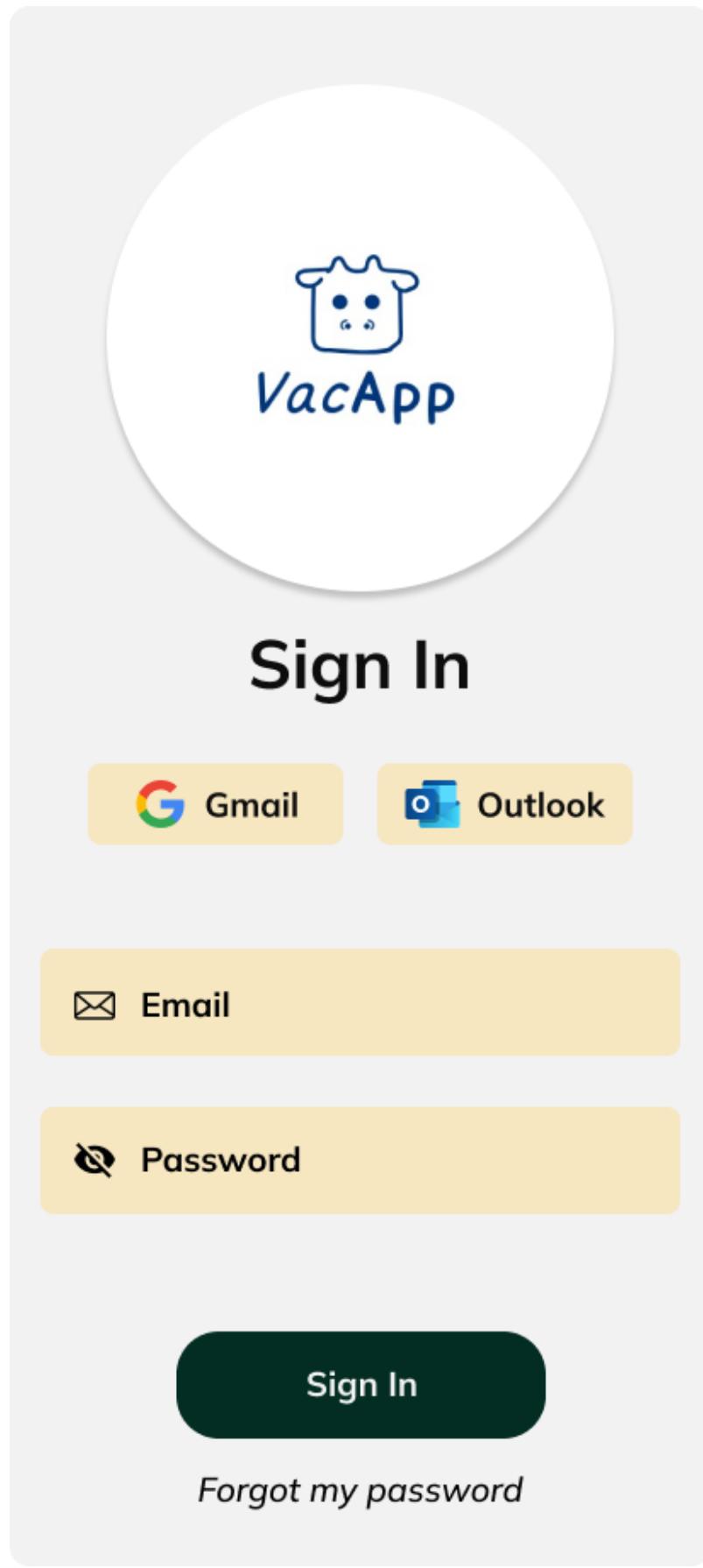
Name
Healthy Herd 2025

Description
Cattle vaccination

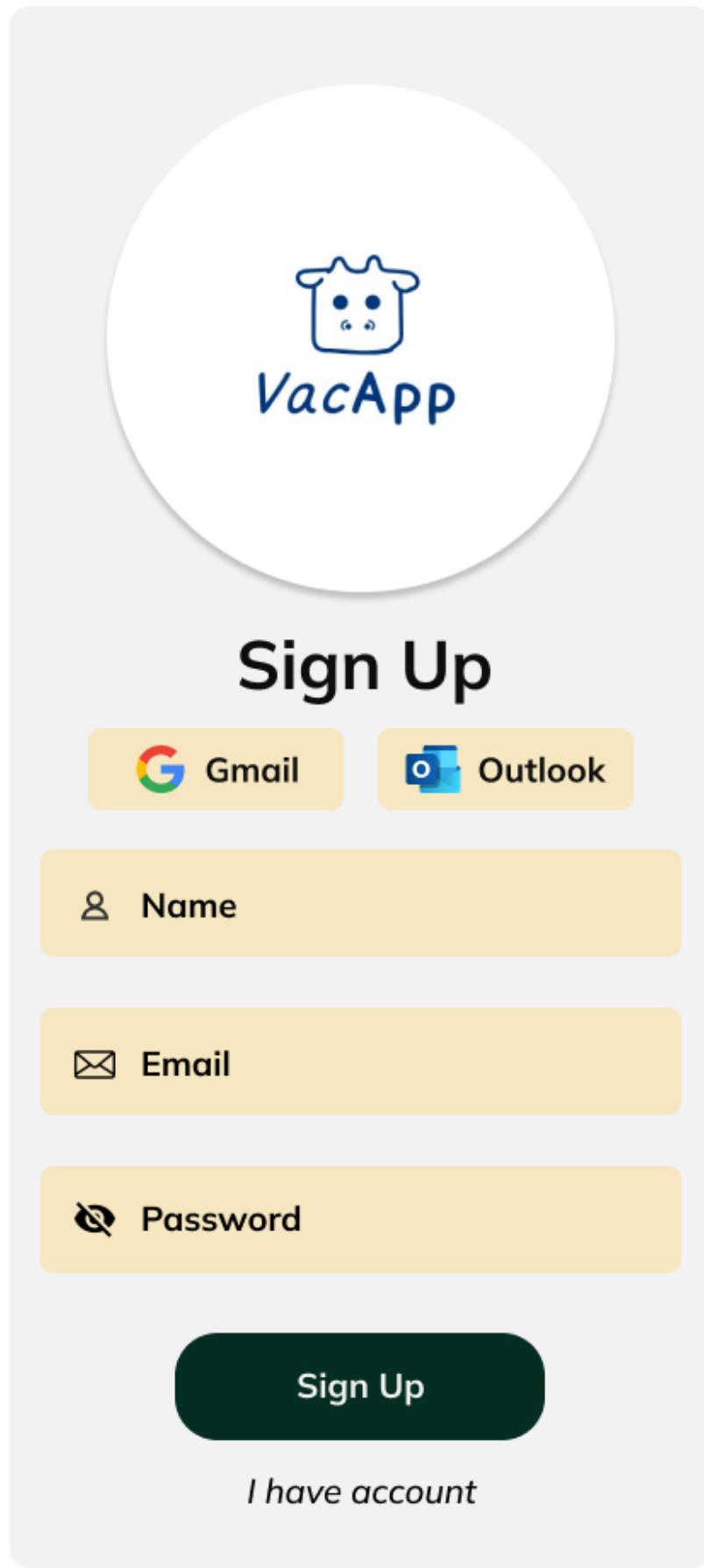
Start date
09/05/2025

End date
20/06/2025

[Inicio Sesión](#)



[Registro](#)



Planes



Plans

Free

S/0

- Registro de hasta 10 animales.
- Acceso básico al historial sanitario.
- Recordatorios limitados.
- Visualización de reportes mensuales en formato simple.
- Soporte por correo electrónico.

Choose

Premium

Desde

S/49.90

- Registro ilimitado de animales.
- Seguimiento completo de salud y vacunación.
- Gestión de inventario y productos.
- Soporte prioritario.

Choose

Home

The dashboard features a dark green header bar with a white three-line menu icon on the left. Below the header is a large, bold, dark green title "Welcome Juan Jose!". A yellow rounded rectangle contains the text "Registered animals" above the number "123". To the left of a light yellow rounded rectangle is the number "4" and the text "Campaigns". To the right of another light yellow rounded rectangle is the number "32" and the text "Employees". Below these are two more light yellow rounded rectangles: one with "10" and "Vaccines about to expire" and another with "6" and "Campaigns". At the bottom center is a large white button with a black plus sign inside a circle. The footer has a dark green bar with a white three-line menu icon on the left, the word "Home" with a house icon in the middle, and the word "come" in large white letters on the right.

Welcome
Juan Jose!

Registered animals

123

4 Campaigns

32 Employees

10 Vaccines about to expire

6 Campaigns

Upcoming Events

Foot and Mouth Disease Vaccination	10-May
Internal and External Deworming	05-July
Brucellosis Sanitation Campaign	23-August

+

The footer consists of a dark green bar with a white three-line menu icon on the left. In the center, there is a white button with a house icon and the word "Home". To its right is a white button with the word "come" in large, bold, dark green letters.

Home

come

Animals

Campaigns

Inventory

Settings

Log out

San Jose!

Number of animals: 123

Number of employees: 32

Number of campaigns: 6

Upcoming Events

Vaccination	10-May
Training	05-July
Charity Campaign	23-August

+

The dashboard features a dark green header bar with a white three-line menu icon on the left. Below the header, a large title 'Welcome Juan Jose!' is displayed in bold, dark green font. A yellow rounded rectangle contains the text 'Registered animals' above the number '123'. To the left of this box is another yellow box with '4 Campaigns' and 'Campaigns' below it. To the right is a yellow box with '32 Employees' and 'Employees' below it. Below these are two more yellow boxes: one with '10 Vaccines about to expire' and 'Vaccines about to expire' below it, and another with '6 Campaigns' and 'Campaigns' below it. At the bottom, a section titled 'Upcoming Events' lists three items: 'Foot and Mouth Disease Vaccination' (with a date '10-May'), 'Internal and External Deworming' (with a date '10-May'), and 'Brucellosis Sanitation Campaign' (with a date '10-May'). A modal window is overlaid on the dashboard, containing icons for 'Animal' (cow), 'Campaign' (calendar), and 'Inventory' (box), each with a small circular arrow icon. A close button (an 'X') is located at the bottom right of the modal.

Welcome Juan Jose!

Registered animals

123

4 Campaigns

32 Employees

10 Vaccines about to expire

6 Campaigns

Upcoming Events

Foot and Mouth Disease Vaccination (10-May)

Internal and External Deworming (10-May)

Brucellosis Sanitation Campaign (10-May)

Animal

Campaign

Inventory

Animals



Animals

Gloria ♀



Campaign

Vacas locas

Peso

510 kg

Edad

4 años



Motomoto ♂



Campaign

Vacas locas

Peso

510 kg

Edad

4 años



Rebeca ♀



Campaign

Vacas locas

Peso

510 kg

Edad

4 años



Gloria ♀

**Breed**

Gelbvieh

Weight

510 kg

Age

4 age

Birdthdate

12/01/2021

Barn

La Bendición

Location

Chorrillos

Inventario

The image shows a mobile application interface for inventory management. At the top left is a menu icon (three horizontal lines) and at the top right is a back arrow icon.

Inventory

Stored products **850** (i)

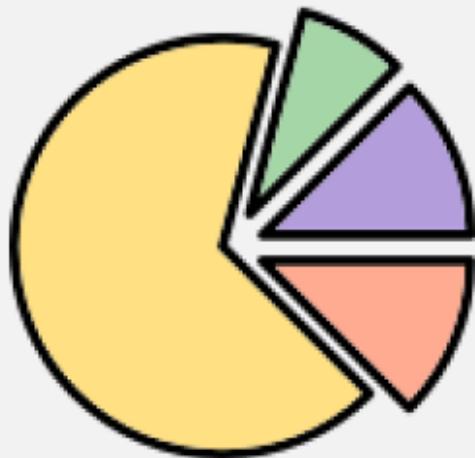
Products with low stock **5**

Fiebre Aftosa (Oleosa)	02
Oxitetraciclina	05
Brucelosis (RB51)	13

Products to expire **5**

Ketoprofeno	18-June
Ivermectina 1%	05-July
Clostridiales (8 vías)	23-July

The image shows the header of a products page. It features a back arrow icon and the word "Products" in large, bold, dark green font.



Medicines Vaccines Food Suplements

Name	Category	Stock
Fiebre Aftosa (Oleosa)	Medicines	02
Oxitetraciclina	Vaccines	05
Brucelosis (RB51)	Vaccines	13
Fiebre Aftosa (Oleosa)	Food	02
Oxitetraciclina	Suplements	05
Brucelosis (RB51)	Food	13

Campaña



My Campaigns

Bovine Brucellosis

Description: Cattle vaccination

Date: 09/05/2025

End Date: 20/06/2025

Foot-and-Mouth Disease

Description: Cattle vaccination

Date: 09/05/2025

End Date: 20/06/2025

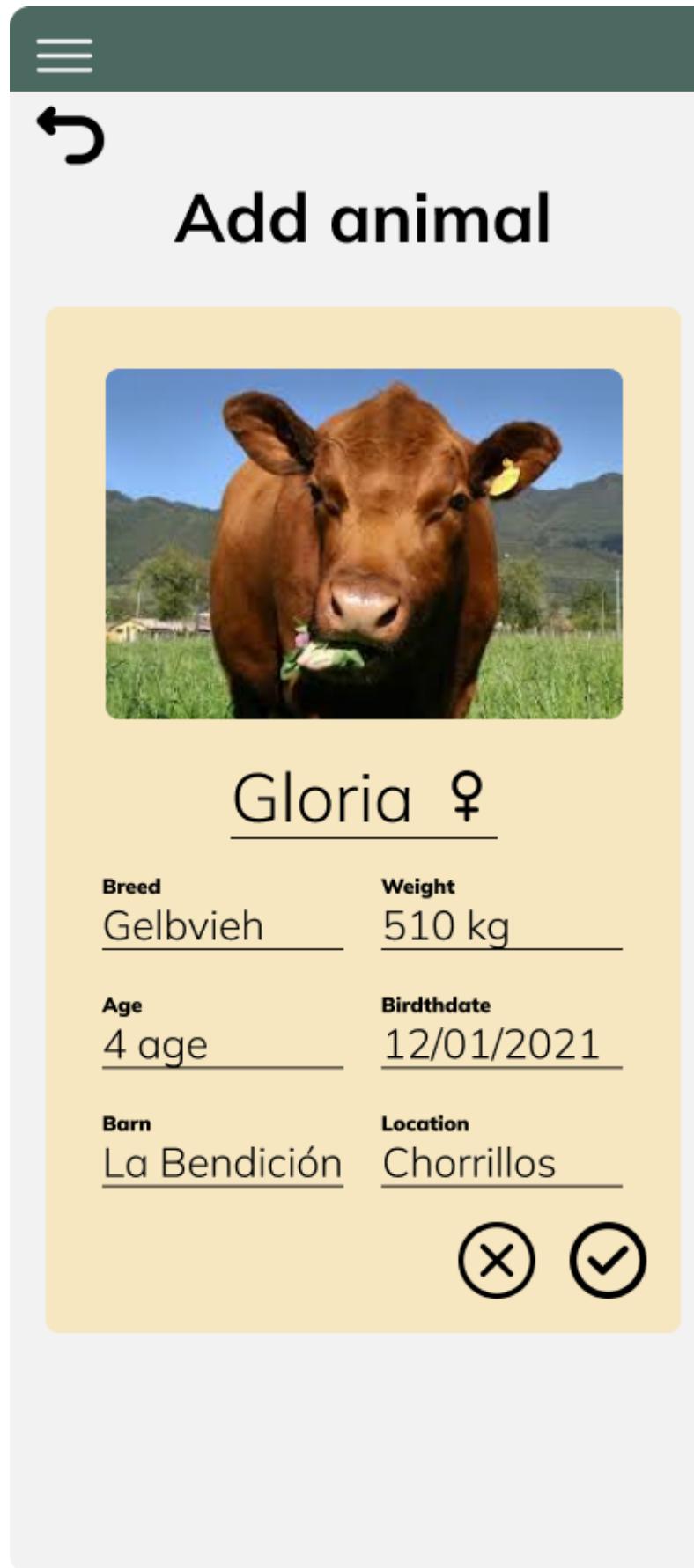
Healthy Herd 2025

Description: Cattle vaccination

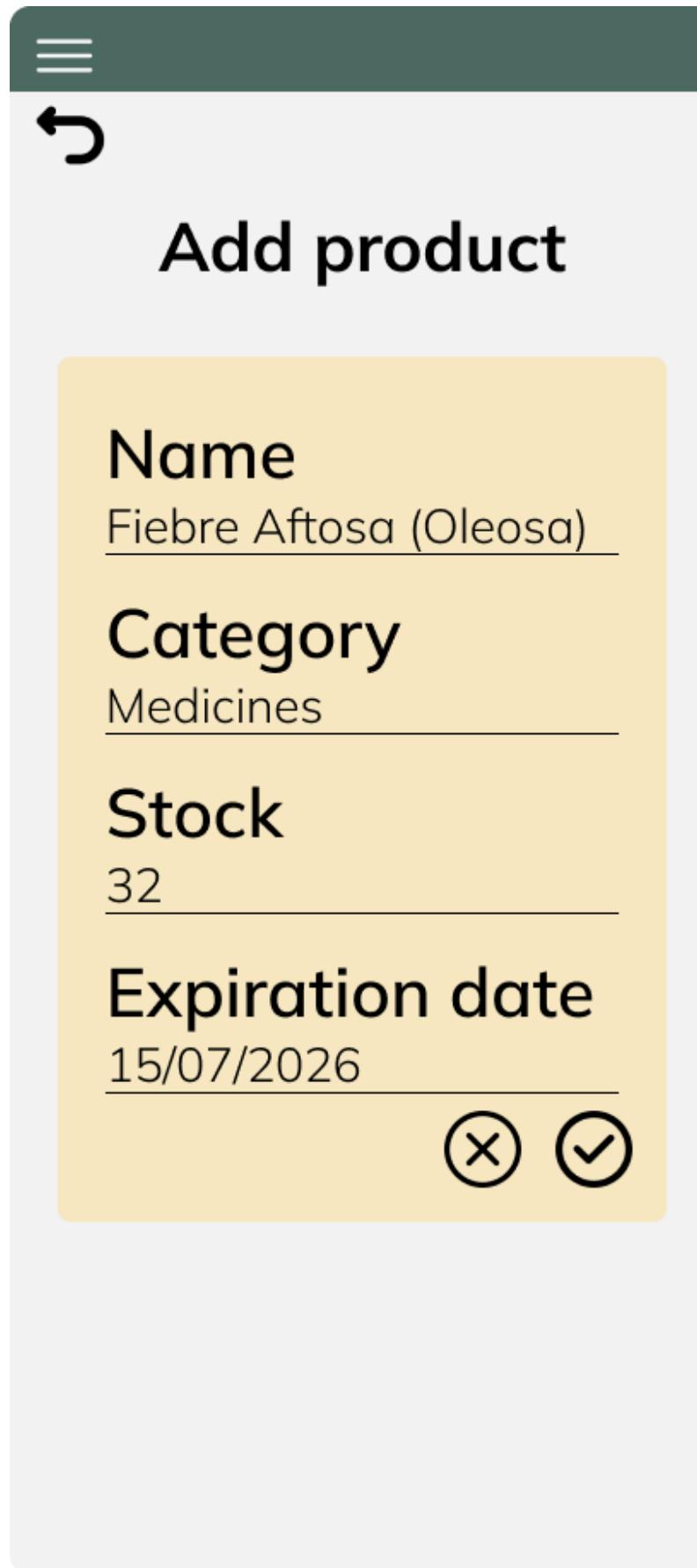
Date: 09/05/2025

End Date: 20/06/2025

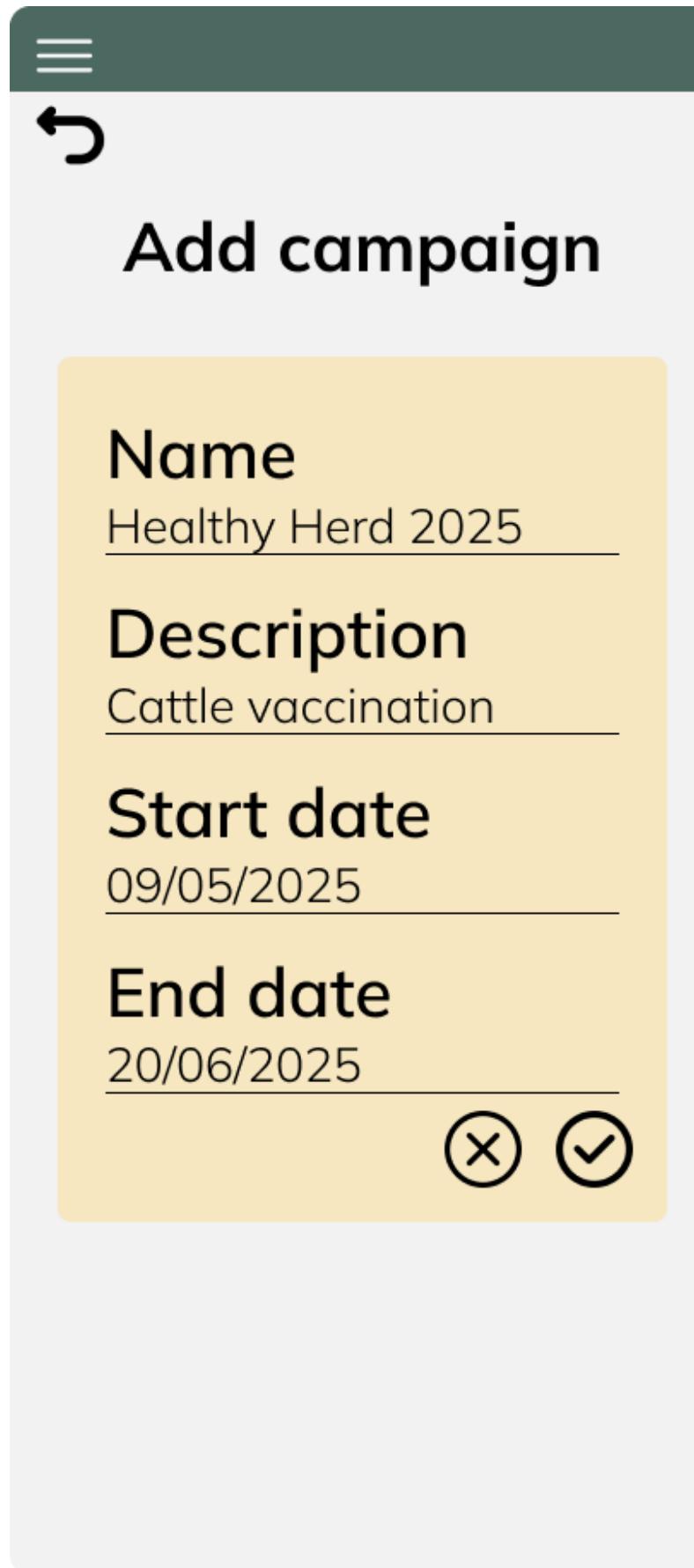
vista de "Aregar Animal"



vista de "Agregar Inventario"



vista de "Aregar Campaña"



4.4.4. Mobile Applications User Flow Diagrams

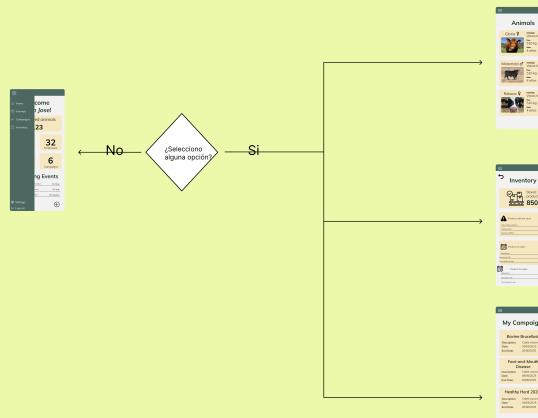
User Goal: Iniciar sesión y Registrarse: Este flujo guía al usuario desde la pantalla de bienvenida hacia las opciones de autenticación.

User Goal: Iniciar sesión y Registrarse



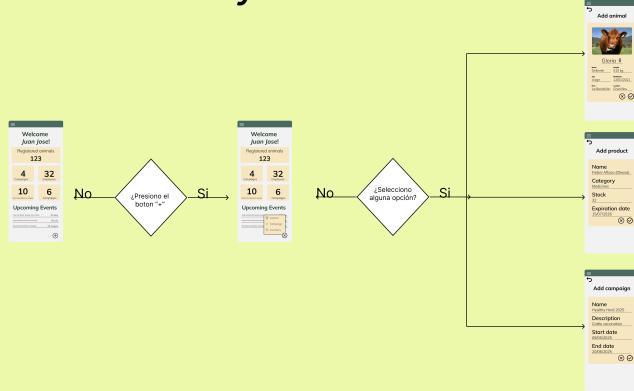
User Goal: Home y navegar por las secciones "Animals", "Campaigns" e "Inventory" Una vez autenticado, el usuario accede al home con accesos rápidos con el sidebar. Este wireflow muestra cómo el usuario puede visualizar las opciones "Animals", "Campaigns" e "inventory"

User Goal: navegar por las secciones "Animals", "Campaigns" e "Inventory"



User Goal: Registro de un nuevo dato ya sea "Animals", "Campaigns" o "Inventory": Este flujo está diseñado para facilitar al agregar ya sea un bovino, una campaña o un producto de inventario.

User Goal: Registro de un nuevo dato ya sea “animal”, “campaign” o “inventory”



4.5. Mobile Applications Prototyping

El proceso de prototipado de las aplicaciones móviles de **VacApp** se llevó a cabo utilizando el framework **Flutter**, lo que permitió generar aplicaciones tanto para **Android** como para **iOS** desde una misma base de código.

De esta manera, los prototipos funcionales se desarrollaron en Flutter y se validaron principalmente en dispositivos Android, asegurando consistencia en la interfaz de usuario y en la experiencia de navegación. Gracias a la naturaleza multiplataforma de Flutter, la misma aplicación está preparada para ejecutarse en iOS sin necesidad de modificaciones significativas, garantizando uniformidad visual y de comportamiento en ambos sistemas operativos.

- **Video de demostración:** <https://youtu.be/aLAsS7FKnE>
- **Diseño en Figma:** <https://www.figma.com/design/Ck5RdO3MzAm16SIReLDO15/Sin-t%C3%A9tulo?node-id=150-5796&t=hGN3YL7RfASQ5FFk-1>

4.5.1. Android Mobile Applications Prototyping

Para Android, los prototipos fueron compilados y probados en un entorno de desarrollo utilizando **Android Studio** y dispositivos físicos con sistema operativo Android.

Este proceso permitió verificar la correcta implementación de los estilos, colores y componentes previamente definidos en las **Style Guidelines**.

Como evidencia, se incluyen capturas de pantalla que muestran el prototipo ejecutándose en Android, lo que asegura que el flujo de navegación y las funcionalidades principales cumplen con lo planificado en el diseño inicial.

4.5.2. iOS Mobile Applications Prototyping

Si bien las pruebas principales se realizaron en Android, el prototipo desarrollado en Flutter está preparado para ejecutarse en **iOS** de forma nativa.

Esto se debe a que Flutter utiliza una única base de código que se adapta automáticamente a ambos entornos móviles.

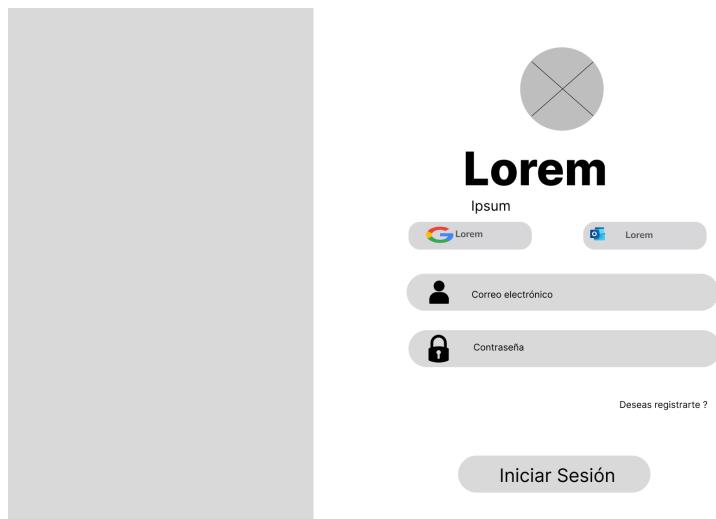
En este sentido, aunque no se disponga de un build completo de iOS para este avance, se documenta que la aplicación mantiene **consistencia visual y funcional** con la versión de Android, cumpliendo los lineamientos definidos en las **iOS Mobile Style Guidelines**.

En futuras iteraciones se contempla la generación de un prototipo directamente en dispositivos iOS o en emuladores de Xcode, lo que permitirá validar también en dicho entorno los aspectos específicos de navegación y experiencia de usuario.

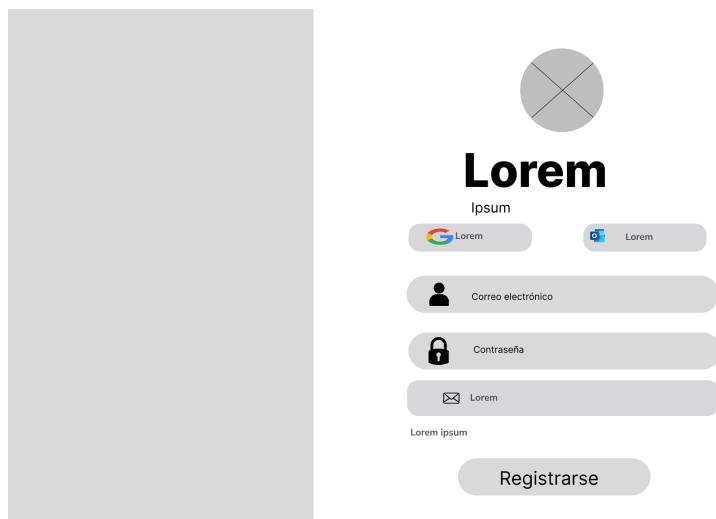
4.6. Web Applications UX/UI Design

4.6.1. Web Applications Wireframes

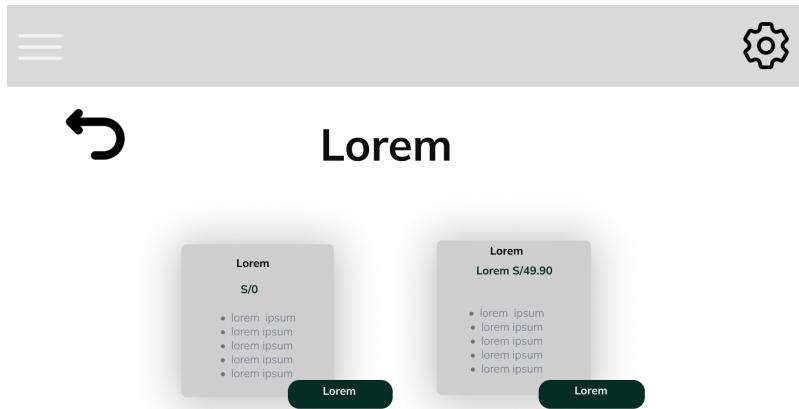
Inicio Sesión



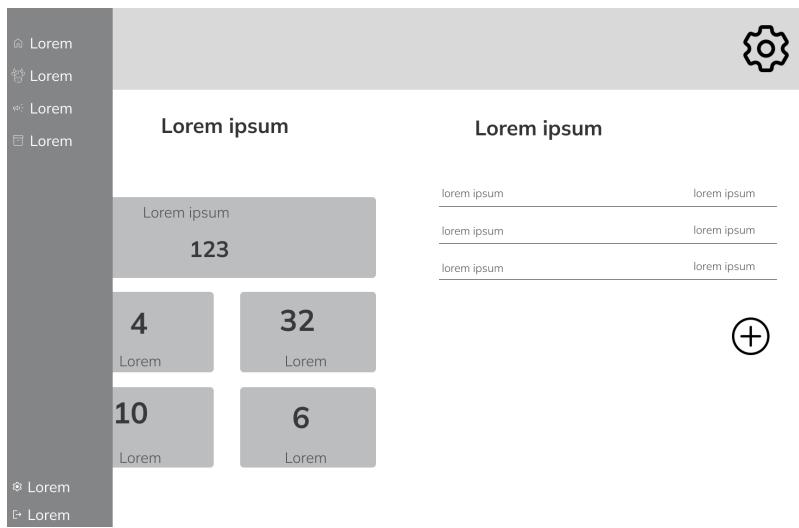
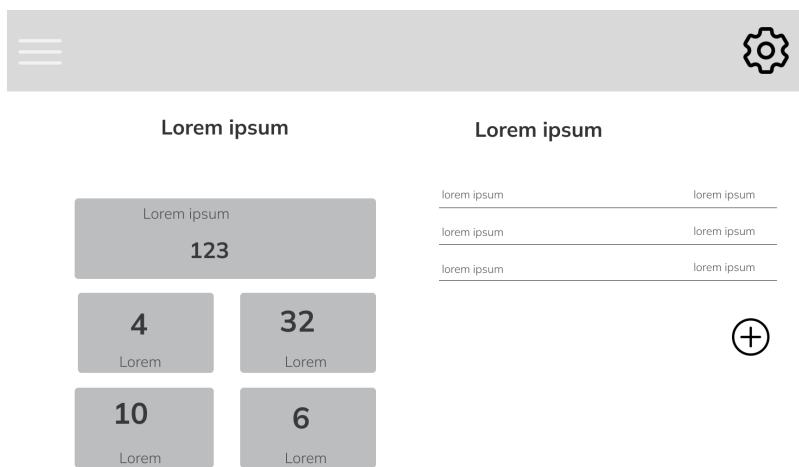
Registro

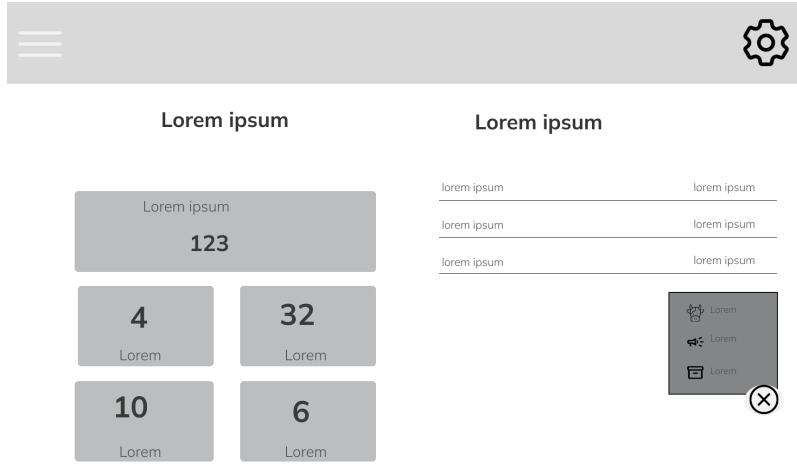


Planes

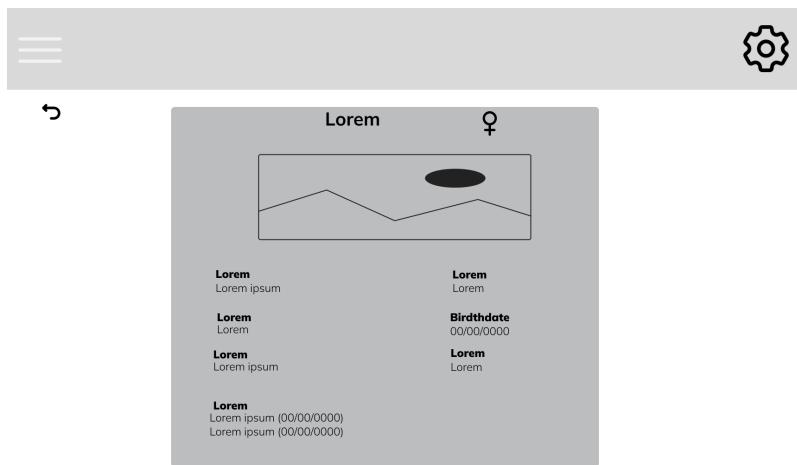
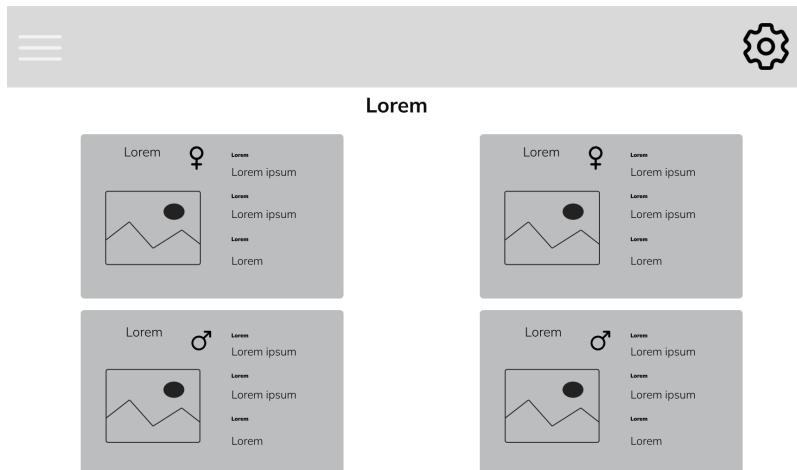


Home

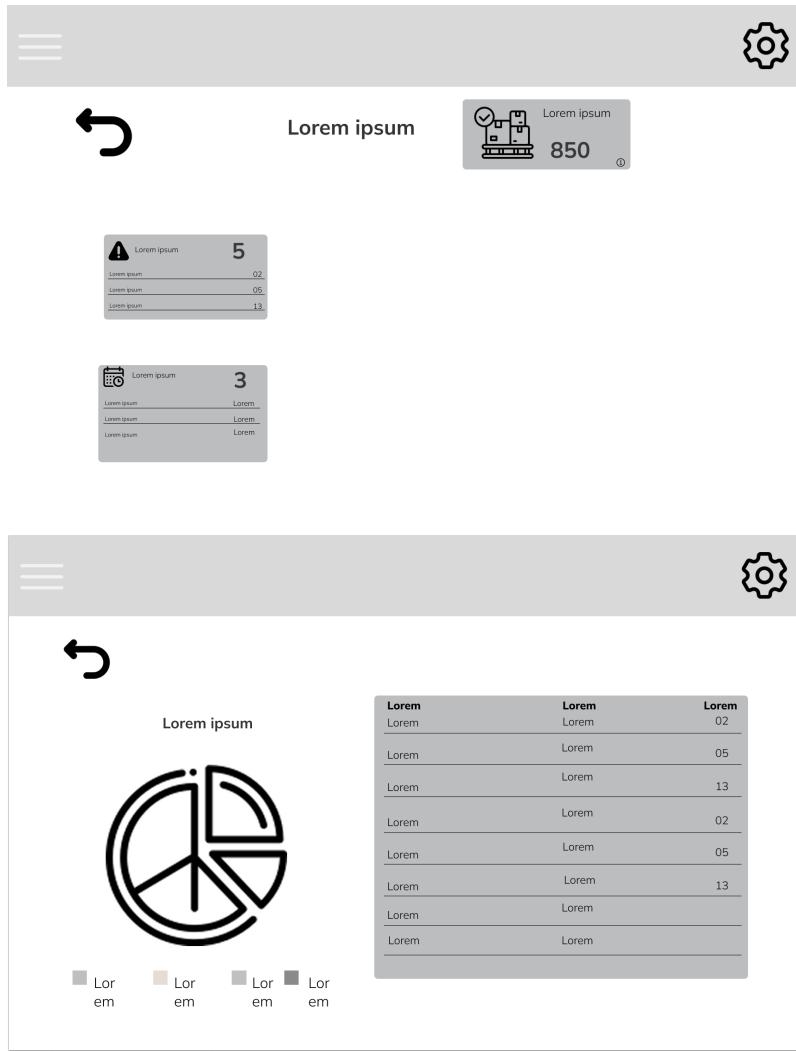




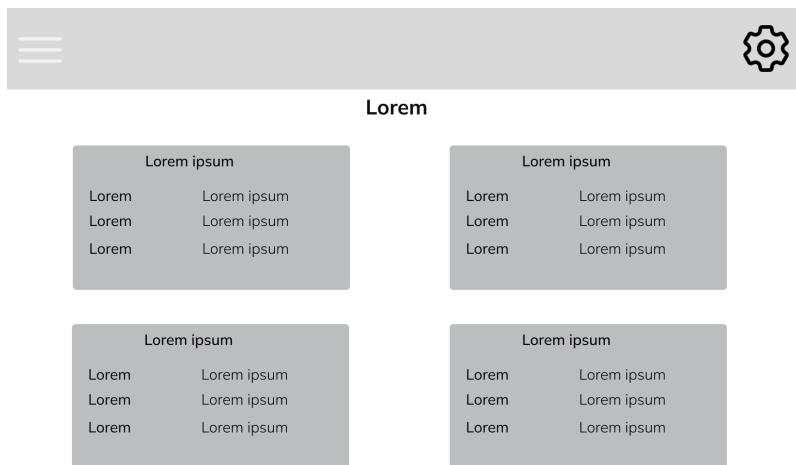
Animals



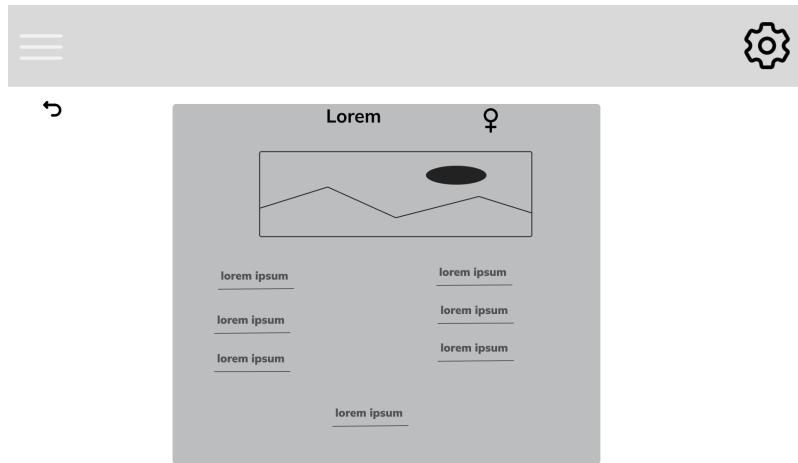
Inventario



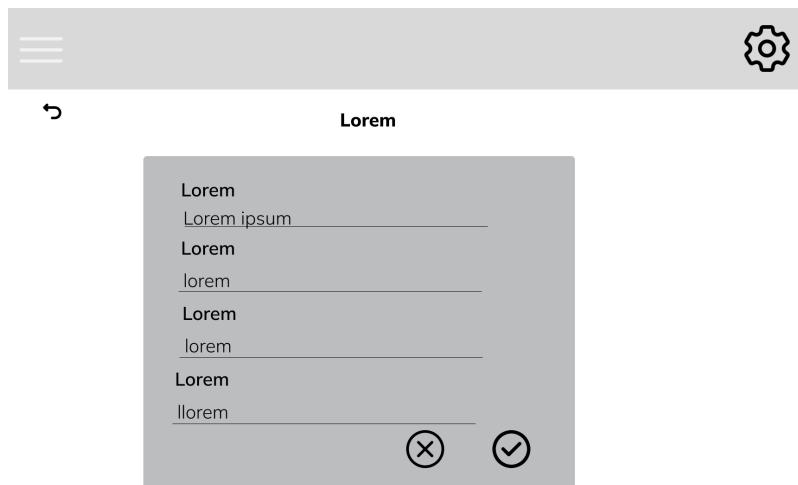
Campaña



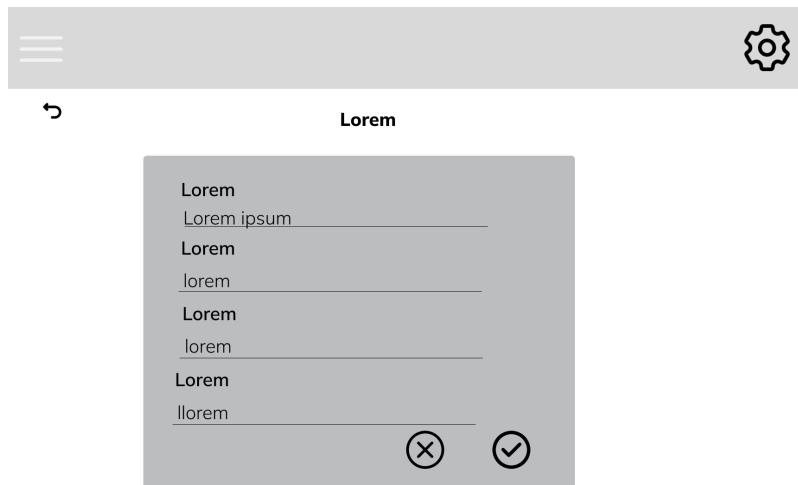
vista de "Agregar Animal"



vista de "Agregar Inventory"



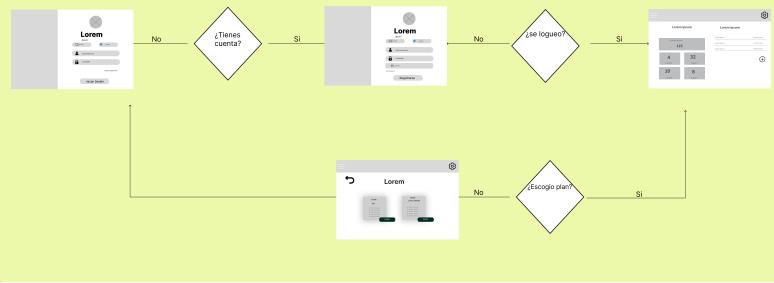
vista de "Agregar Campaña"



4.6.2. Web Applications Wireflow Diagrams

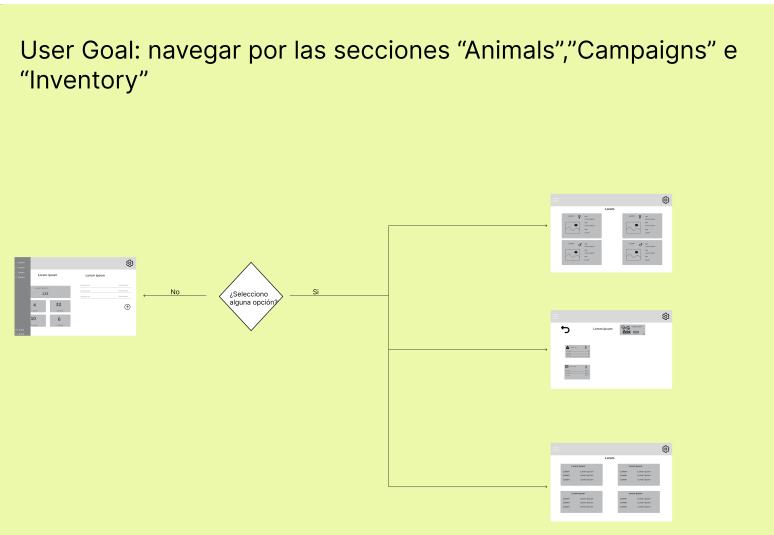
User Goal: Iniciar sesión y Registrarse: Este flujo guía al usuario desde la pantalla de bienvenida hacia las opciones de autenticación.

User Goal: Iniciar sesión y Registrarse



User Goal: Home y navegar por las secciones "Animals", "Campaigns" e "Inventory"

Una vez autenticado, el usuario accede al home con accesos rápidos con el sidebar. Este wireflow muestra cómo el usuario puede visualizar las opciones "Animals", "Campaigns e "inventory"



User Goal: Registro de un nuevo dato ya sea "Animals", "Campaigns" o "Inventory":

Este flujo está diseñado para facilitar al agregar ya sea un bovino, una campaña o un producto de inventario.



4.6.3. Web Applications Mock-ups

Inicio Sesión



VacApp

Sign In

[G Gmail](#) [Outlook](#)

Email

Password

Deseas registrarte?

[Sign In](#)

[Forgot my password](#)

Registro



VacApp

Sign In

[G Gmail](#) [Outlook](#)

Name

Email

Password

Accept términos y condiciones

[Sign Up](#)

[I have account](#)

Planes



Plans

<p>Free S/0</p> <ul style="list-style-type: none"> • Registro de hasta 10 animales. • Acceso básico al historial sanitario. • Recordatorios limitados. • Visualización de reportes mensuales en formato simple. • Soporte por correo electrónico. <p>Choose</p>	<p>Premium Desde S/49.90</p> <ul style="list-style-type: none"> • Registro ilimitado de animales. • Seguimiento completo de salud y vacunación. • Gestión de inventario y productos. • Soporte prioritario. <p>Choose</p>
---	--

Home

Welcome
Juan Jose!

Upcoming Events

Event Type	Date
Foot and Mouth Disease Vaccination	10-May
Internal and External Deworming	05-July
Brucellosis Sanitation Campaign	23-August

Registered animals: 123

Campaigns: 4	Employees: 32
Vaccines about to expire: 10	Campaigns: 6

Upcoming Events

Event Type	Date
Foot and Mouth Disease Vaccination	10-May
Internal and External Deworming	05-July
Brucellosis Sanitation Campaign	23-August

Registered animals: 123

4

Upcoming Events

Event Type	Date
Foot and Mouth Disease Vaccination	10-May
Internal and External Deworming	05-July
Brucellosis Sanitation Campaign	23-August

Welcome
Juan Jose!

Upcoming Events

Event Type	Date
Foot and Mouth Disease Vaccination	10-May
Internal and External Deworming	05-July
Brucellosis Sanitation Campaign	23-August

Registered animals: 123

Campaigns: 4	Employees: 32
Vaccines about to expire: 10	Campaigns: 6

Animal
Campaign
Inventory

Animals

The screenshot shows a mobile application interface with a dark green header bar. On the left is a menu icon (three horizontal lines), and on the right is a gear icon. The title "Animals" is centered above a grid of six animal cards. Each card contains a small image of the animal, its name, breed, weight, age, and a circular icon.

Name	Breed	Weight	Age
Gloria ♀	Vacas locas	510 kg	4 años
Gloria ♀	Vacas locas	510 kg	4 años
Gloria ♀	Vacas locas	510 kg	4 años
Motomoto ♂	Vacas locas	510 kg	4 años
Motomoto ♂	Vacas locas	510 kg	4 años

The screenshot shows a detailed view of a cow named "Gloria ♀". At the top, there is a back arrow icon. The main image is a close-up of Gloria's face, looking directly at the camera. Below the image, her details are listed: Breed (Gelbvieh), Age (4 age), Born (La Bendición), Weight (510 kg), and Birthdate (12/01/2021). The background shows a green field and some hills under a clear blue sky.

Inventario

Inventory

Stored products: 850

⚠ Products with low stock	
Fiebre Aftosa (Oleosa)	02
Oxitetraciclina	05
Brucelosis (RB51)	13

📅 Products to expire	
Ketoprofeno	18-June
Ivermectina 1%	05-July
Clostridiales (8 vias)	23-July

Products

Pie chart distribution:

- Medicines (Yellow)
- Vaccines (Orange)
- Food (Purple)
- Supplements (Green)

Name	Category	Stock
Fiebre Aftosa (Oleosa)	Medicines	02
Oxiteciclina	Vaccines	05
Brucelosis (RB51)	Vaccines	13
Fiebre Aftosa (Oleosa)	Food	02
Oxiteciclina	Supplements	05
Brucelosis (RB51)	Food	13

Campaña

My Campaigns

Bovine Brucellosis	
Description:	Cattle vaccination
Date	09/05/2025
End Date:	20/06/2025

Bovine Brucellosis	
Description:	Cattle vaccination
Date	09/05/2025
End Date:	20/06/2025

Bovine Brucellosis	
Description:	Cattle vaccination
Date	09/05/2025
End Date:	20/06/2025

Bovine Brucellosis	
Description:	Cattle vaccination
Date	09/05/2025
End Date:	20/06/2025

vista de "Aregar Animal"

Add animal

Gloria ♀

Breed	Gelbvieh	Weight	510 kg
Age	4 años	Birthdate	12/01/2021
Born	La Bendición	Location	Chorrillos

✖️ ✅

vista de "Aregar Inventario"

Add product

Name
Fiebre Aftosa (Oleosa)

Category
Medicines

Stock
32

Expiration date
15/07/2026

✖️ ✅

vista de "Aregar Campaña"

Add campaign

Name
Healthy Herd 2025

Description
Cattle vaccination

Start date
09/05/2025

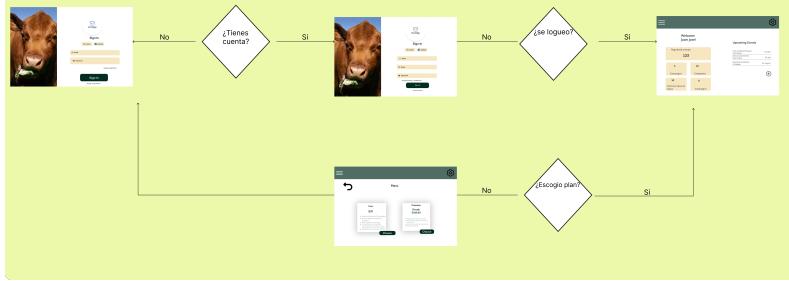
End date
20/06/2025

✖️ ✅

4.6.4. Web Applications User Flow Diagrams

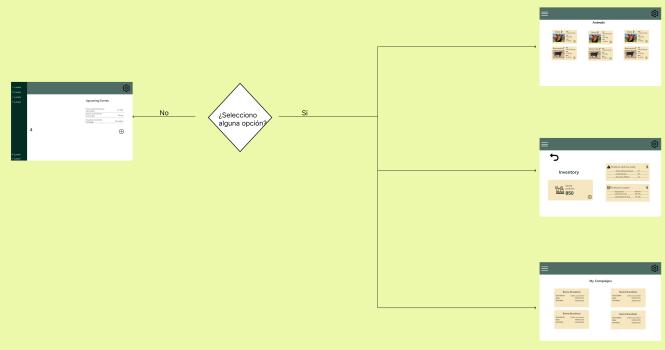
User Goal: Iniciar sesión y Registrarse: Este flujo guía al usuario desde la pantalla de bienvenida hacia las opciones de autenticación.

User Goal: Iniciar sesión y Registrarse



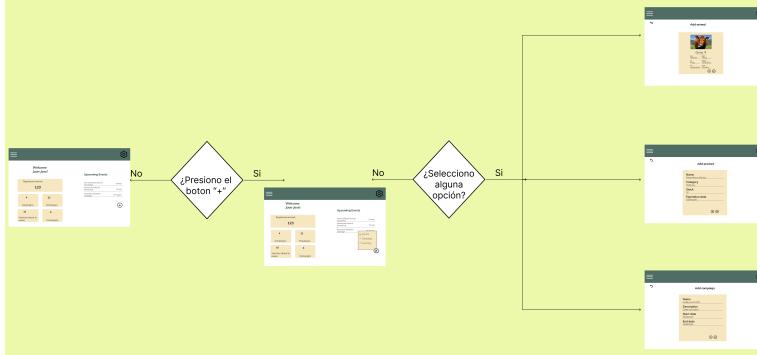
User Goal: Home y navegar por las secciones "Animals", "Campaigns" e "Inventory" Una vez autenticado, el usuario accede al home con accesos rápidos con el sidebar. Este wireflow muestra cómo el usuario puede visualizar las opciones "Animals", "Campaigns" e "inventory"

User Goal: navegar por las secciones "Animals", "Campaigns" e "Inventory"



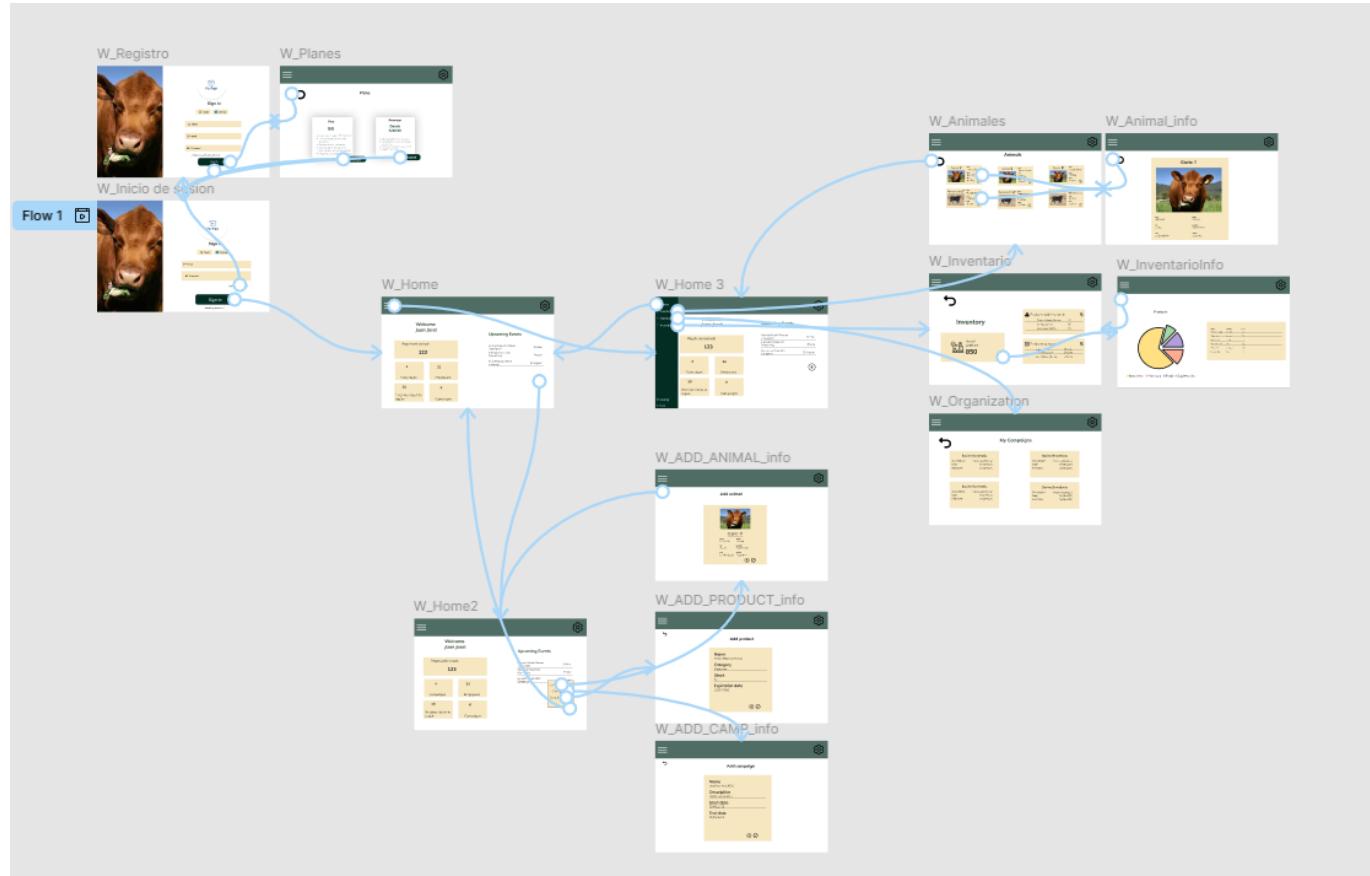
User Goal: Registro de un nuevo dato ya sea "Animals", "Campaigns" o "Inventory": Este flujo está diseñado para facilitar al agregar ya sea un bovino, una campaña o un producto de inventario.

User Goal: Registro de un nuevo dato ya sea "animal", "campaign" o "inventory"



4.7. Web Applications Prototyping

A continuación, se presenta el prototipo que se realizó en base a los mockups que se desarrollaron y documentaron en puntos anteriores. El prototype nos permite evidenciar algunos flujos que se llevarán al desarrollo en código.



<https://www.figma.com/proto/47ngQ2UkWbuuvLy36Fmfnl/Pet-Care---Landing-page--Community--?node-id=2102-445&p=f&t=pbikLYhwwgU7T9wf-1&scaling=scale-down&content-scaling=fixed&page-id=27%3A1&starting-point-node-id=2102%3A37>

4.8. Domain-Driven Software Architecture

En esta sección se presenta la arquitectura de software de **VacApp**, diseñada bajo el enfoque de **Domain-Driven Design (DDD)**.

El objetivo es ofrecer una visión clara de cómo los distintos dominios del sistema (gestión de ganado, campañas de vacunación, establos, usuarios, entre otros) se estructuran y cómo interactúan con los actores externos y los servicios complementarios.

Se emplean diagramas **C4** para ilustrar distintos niveles de detalle, desde el contexto general hasta los componentes principales de la solución.

En esta sección se presenta la arquitectura de software de **VacApp**, diseñada bajo el enfoque de **Domain-Driven Design (DDD)**.

El objetivo es ofrecer una visión clara de cómo los distintos dominios del sistema (gestión de ganado, campañas de vacunación, establos, usuarios, entre otros) se estructuran y cómo interactúan con los actores externos y los servicios complementarios.

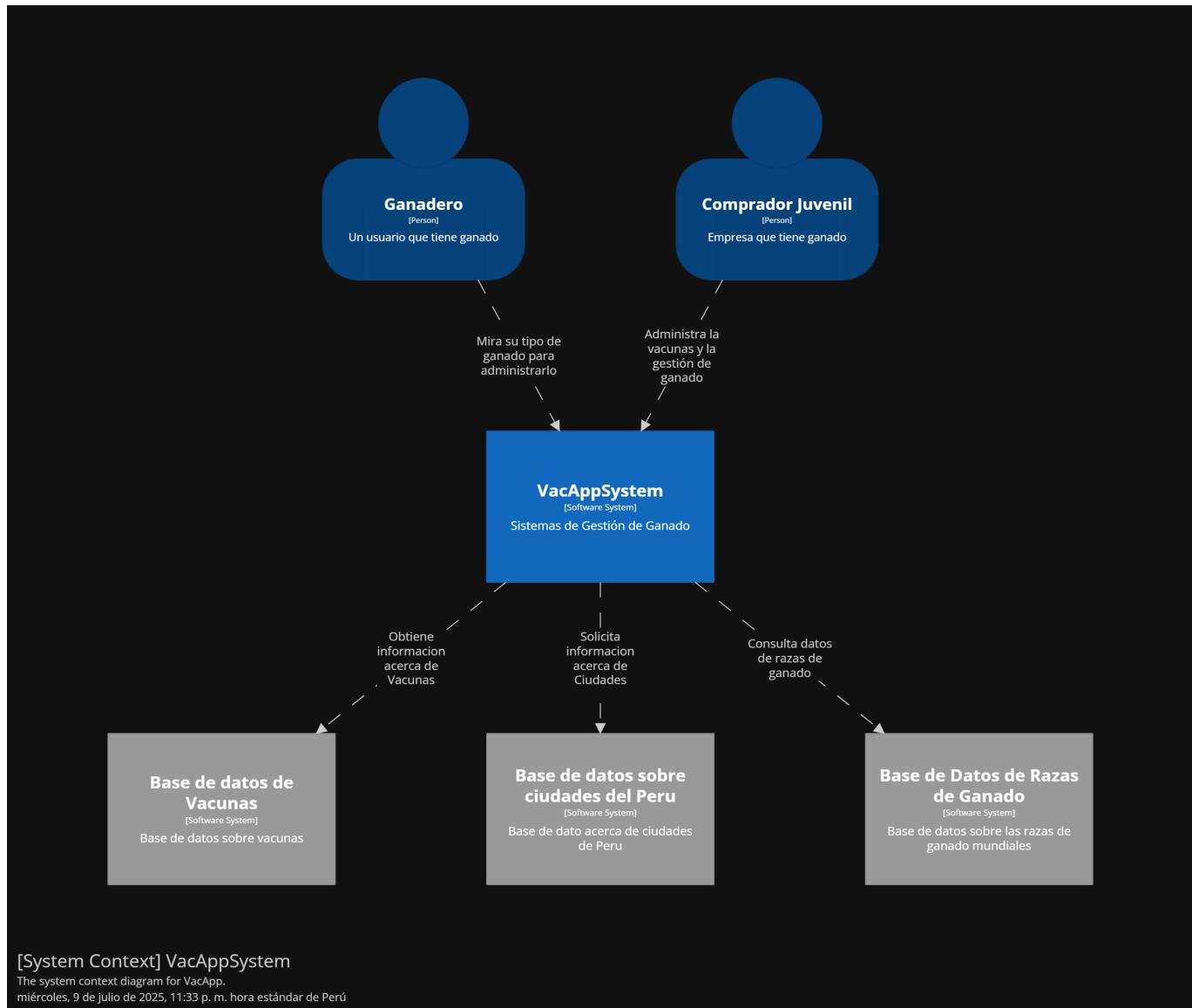
Se emplean diagramas **C4** para ilustrar distintos niveles de detalle, desde el contexto general hasta los componentes principales de la solución.

4.8.1. Software Architecture Context Diagram

El **Diagrama de Contexto** proporciona una visión de alto nivel del sistema, mostrando cómo **VacApp** interactúa con sus principales actores externos:

- **Ganaderos y Empresas** → usuarios finales que gestionan su ganado, vacunas y establos.
- **Servicios Externos** → APIs de pronóstico del clima, bases de datos de razas de ganado y servicios veterinarios.
- **Administradores del Sistema** → responsables del mantenimiento y la supervisión de la aplicación.

Este diagrama permite comprender cómo VacApp se integra en el ecosistema de la gestión ganadera, facilitando la interoperabilidad con otros sistemas.



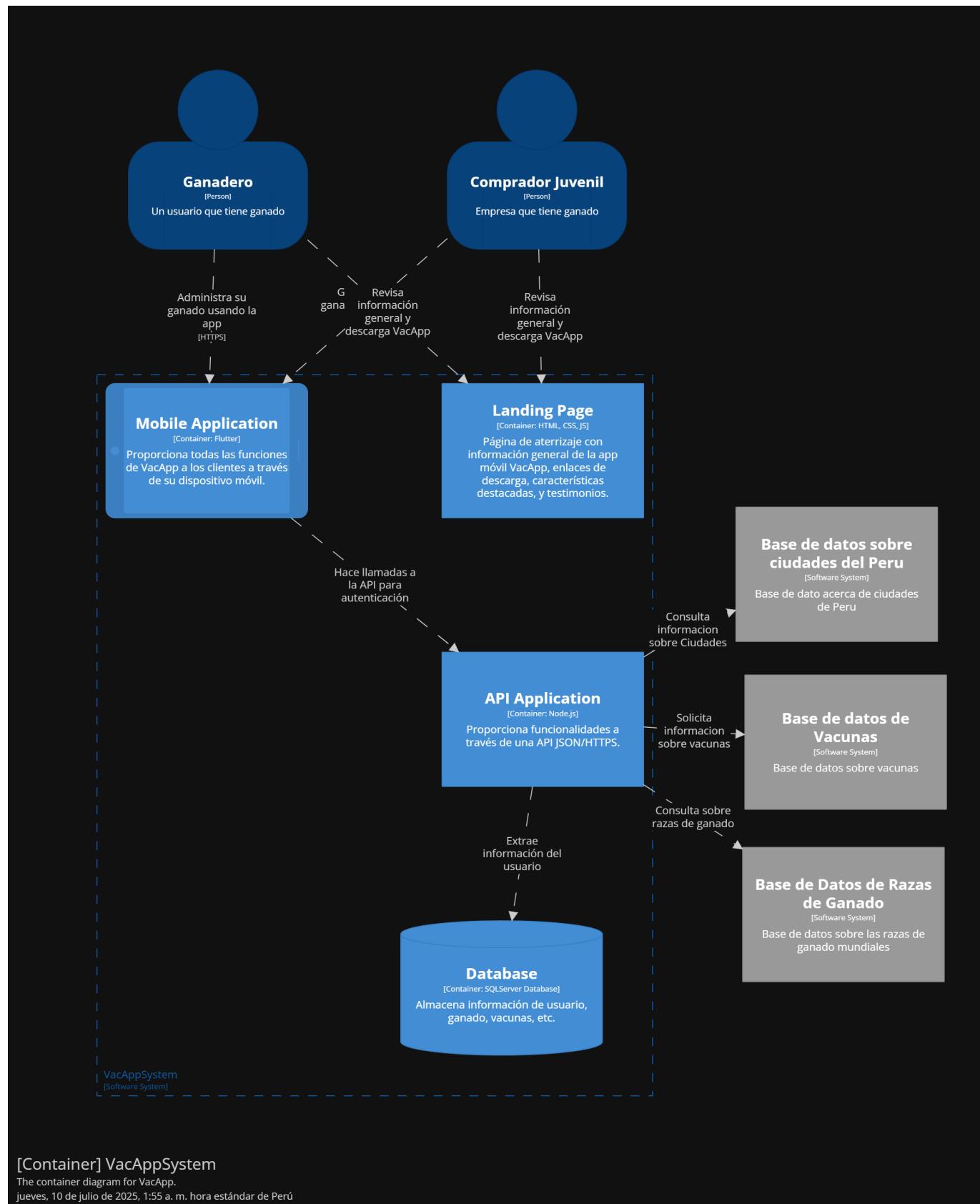
4.8.2. Software Architecture Container Diagrams

El **Diagrama de Contenedores** descompone VacApp en sus principales elementos tecnológicos, ilustrando cómo se organizan y comunican entre sí:

- **Aplicación Web** → interfaz accesible desde navegadores para administradores y empresas.
- **Aplicación Móvil** → interfaz diseñada para ganaderos, desarrollada en **Flutter**, disponible para Android e iOS.

- **API Backend** → expone servicios de negocio a través de un conjunto de endpoints REST, desarrollados en **Java/Spring Boot**.
- **Base de Datos** → repositorio centralizado para almacenar información sobre bovinos, campañas, usuarios, vacunas y establos.

El diagrama refleja cómo cada contenedor coopera para ofrecer una experiencia integral y consistente, garantizando la disponibilidad de la información tanto en la aplicación móvil como en la web.

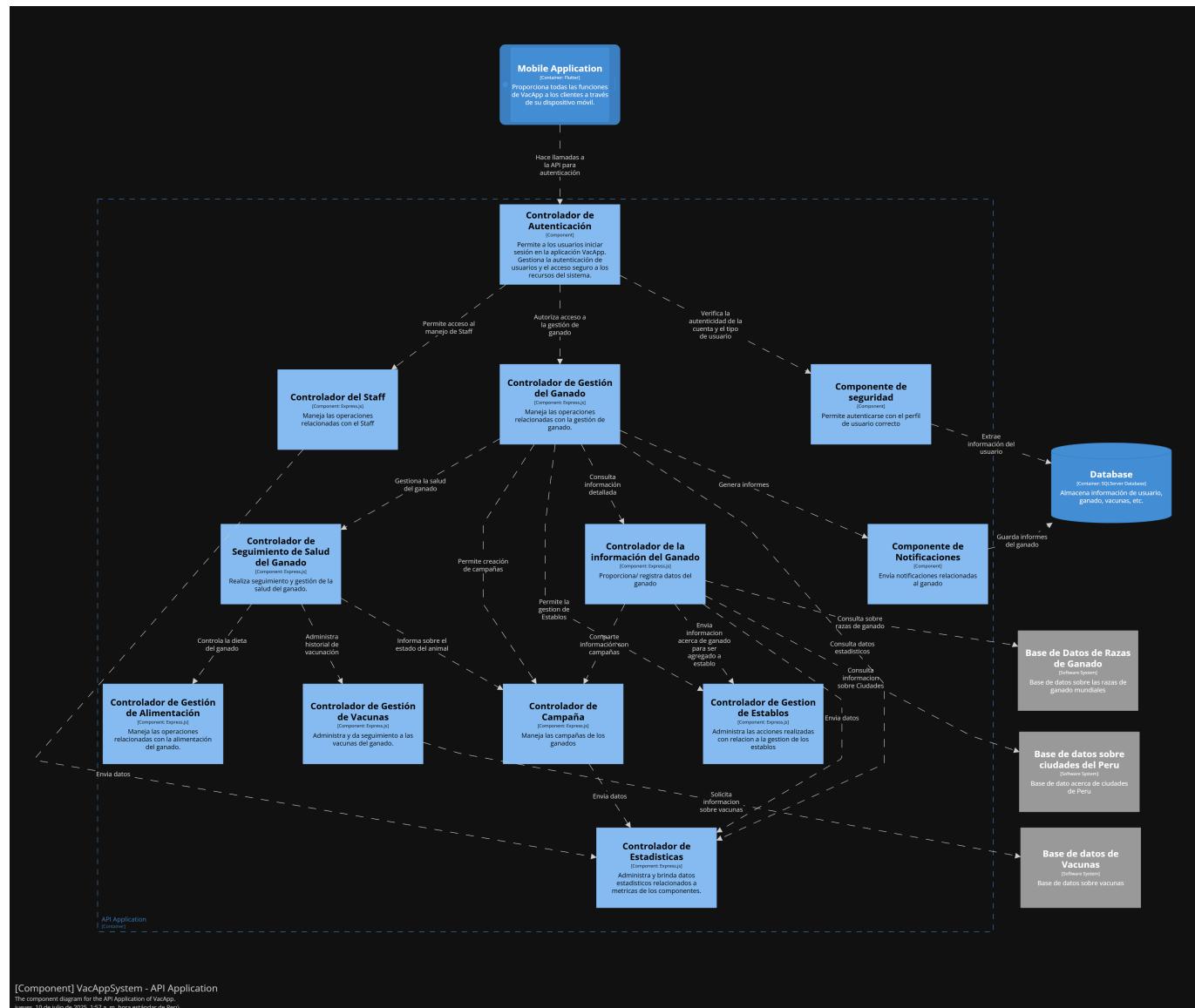


4.8.3. Software Architecture Components Diagrams

El **Diagrama de Componentes** profundiza en los módulos principales dentro del backend y la aplicación móvil. Cada componente está alineado a un **bounded context** del dominio definido por DDD:

- **Gestión de Bovinos** → administración de registros de animales, historial de salud y genealogía.
- **Gestión de Campañas** → planificación y control de campañas de vacunación.
- **Gestión de Establos** → organización de establos y asignación de bovinos.
- **Gestión de Usuarios** → control de accesos, roles y permisos.
- **Notificaciones y Recordatorios** → envío de alertas automáticas relacionadas con vacunas, salud o actividades programadas.

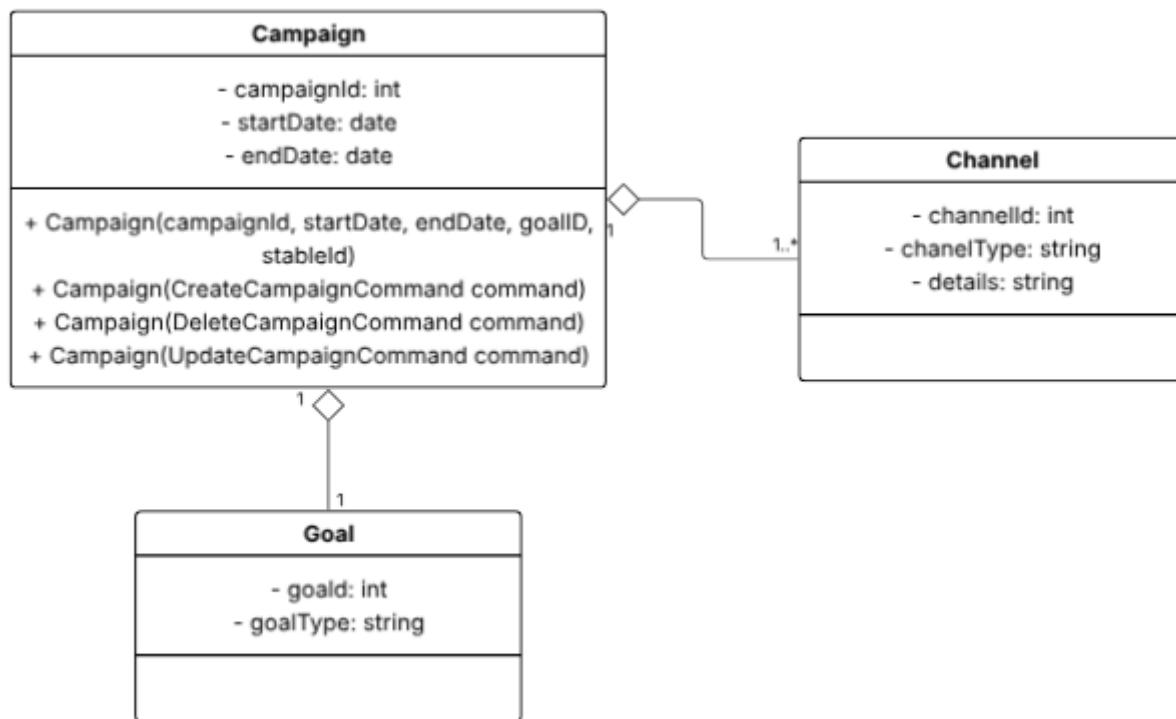
Este nivel de detalle muestra cómo los componentes colaboran entre sí dentro de los contenedores y cómo mantienen la cohesión con el dominio del problema.



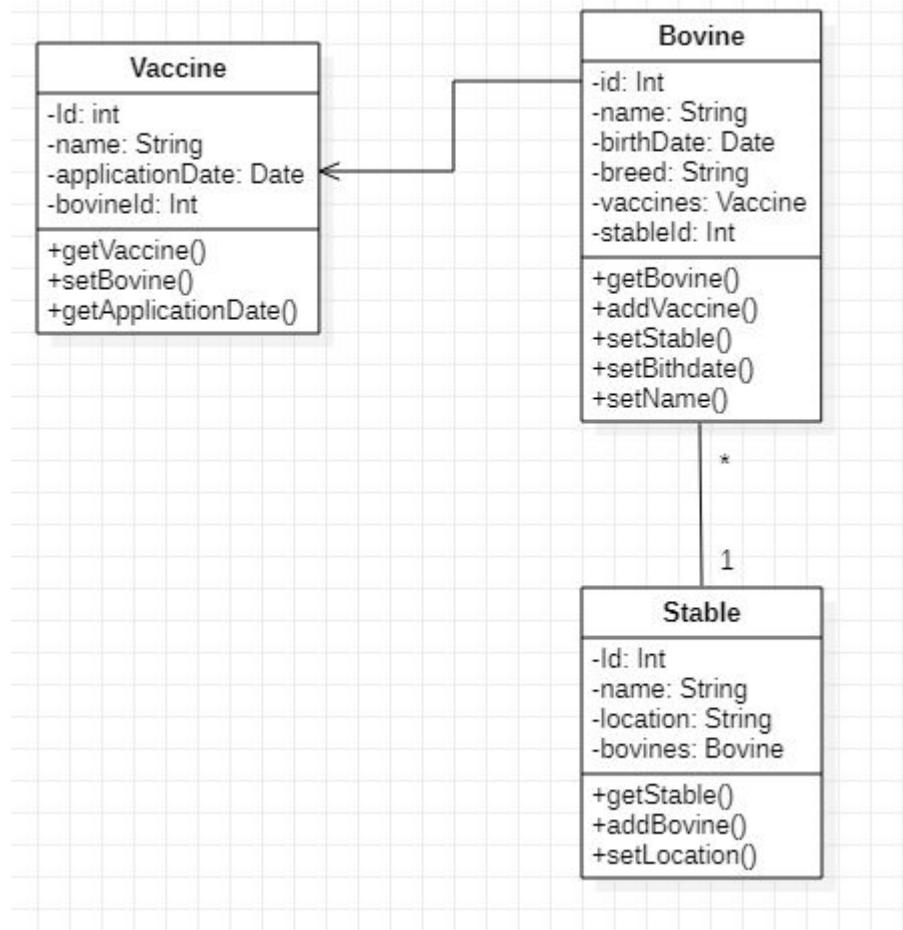
4.9. Software Object-Oriented Design

4.9.1. Class Diagrams

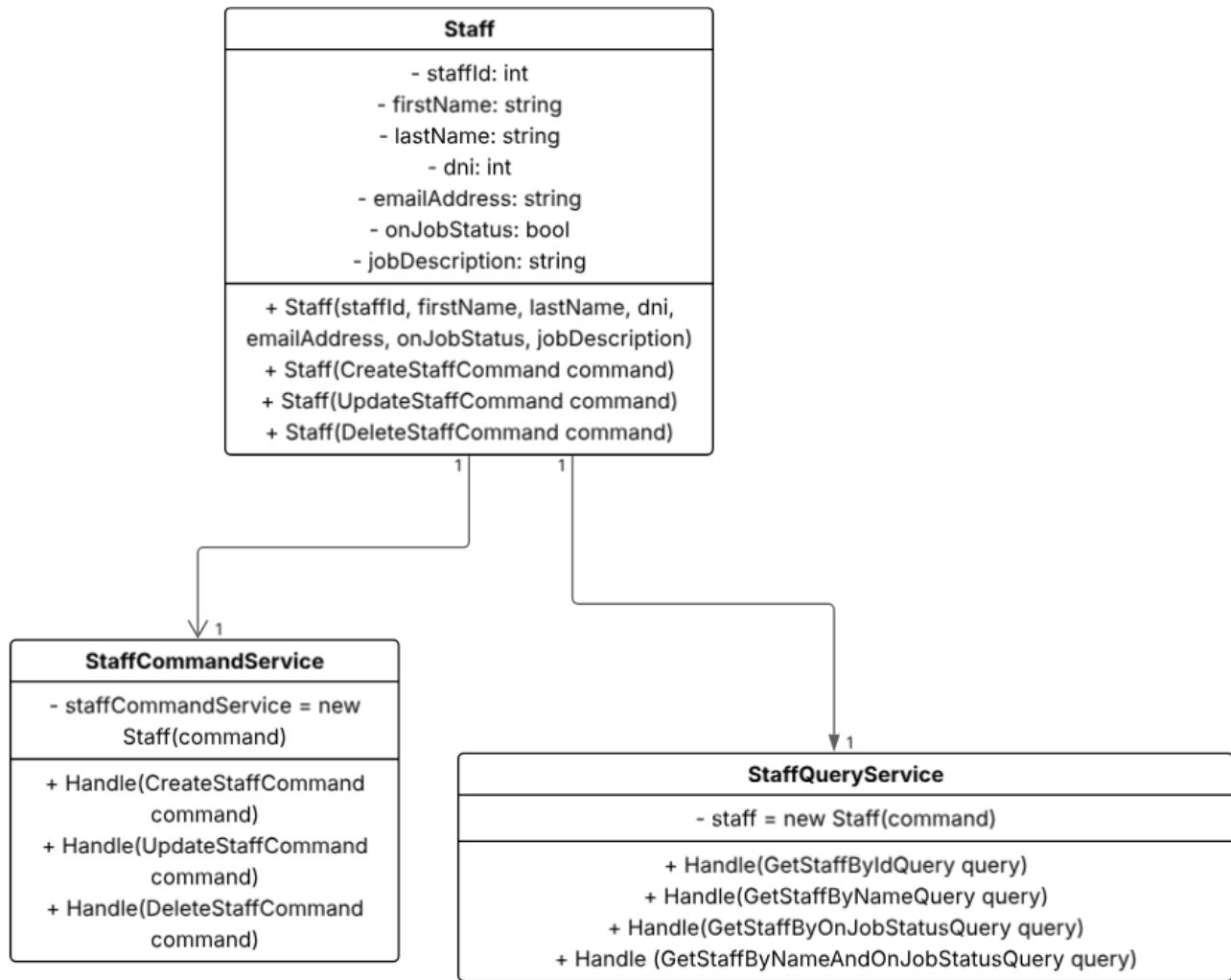
Este diagrama de clases detalla los elementos del Domain Layer para Campaign Management, modelando entidades, agregados, objetos de valor y sus relaciones. A través de esta representación, se puede visualizar cómo se estructuran los conceptos principales del dominio y qué responsabilidades tiene cada clase dentro del modelo de negocio. Es esencial para alinear el diseño técnico con la lógica del dominio.



Incluye entidades como Bovino, Vacuna y Establo, sus atributos, relaciones y métodos, permitiendo visualizar cómo se estructura la lógica de negocio y se representan los objetos reales del sistema productivo. Este diagrama fortalece la alineación entre la realidad ganadera y su implementación en software.



Incluye entidades como StaffMember, objetos de valor como EmployeeStatus y servicios de dominio encargados de las reglas críticas. Este diagrama ayuda a comprender la estructura del dominio y cómo se articulan sus elementos para cumplir los objetivos del sistema.



4.9.2. Class Dictionary

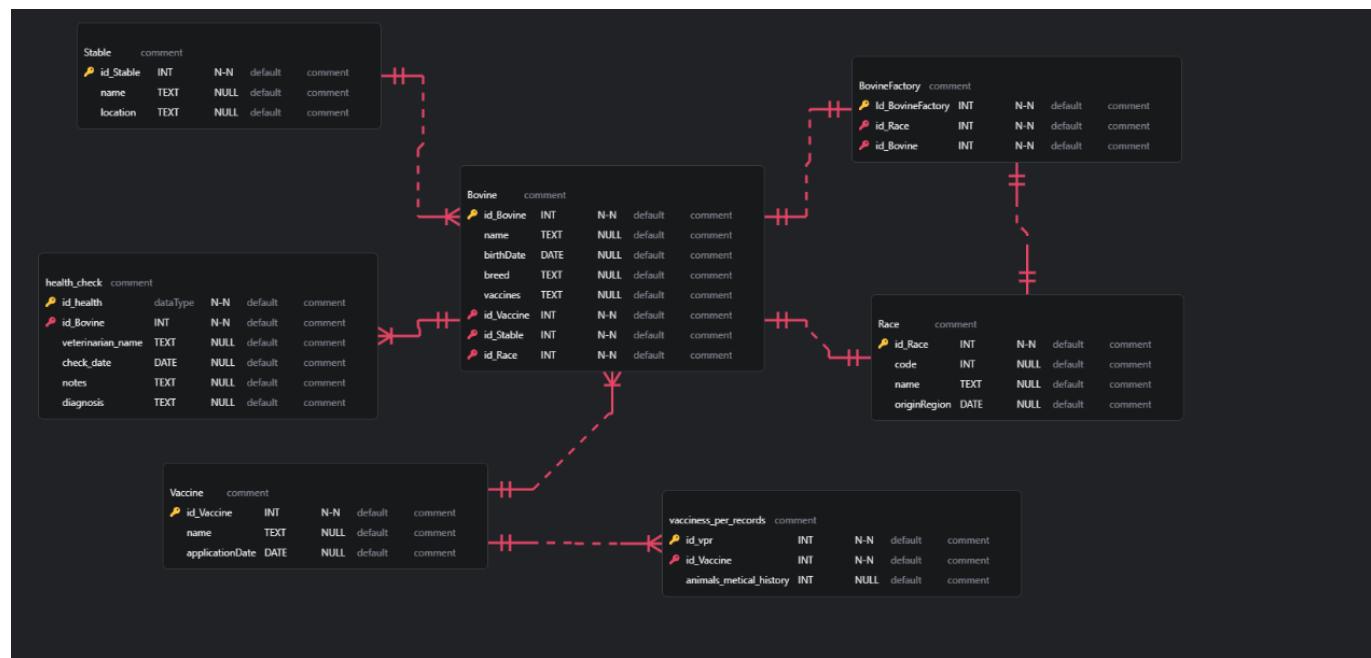
Nombre	Descripción
ID	Identificador único de registro usado como clave primaria
Name	Nombre del animal
Birthday	Fecha de nacimiento del animal
Birth_place	Lugar de Nacimiento del animal
Gender	Genero del animal
Breed	Raza del animal
Location	Lugar donde se ubica el animal
Parent_ID	Identificar de registro de la madre del animal
Expire_date	Fecha de expiracion de la identificacion del bovino
Animal_ID	Identificar de registro de la madre del animal
Nombre	Descripción
id	Identificador único del registro, generalmente una clave primaria.

Nombre	Descripción
first_name	Primer nombre del usuario.
last_name	Apellido del usuario.
job_status	Estado del empleado.
job_description	Descripción del puesto a cargo del empleado
dni	DNI del empleado
email_address	Dirección de correo electrónico del usuario.

4.10. Database Design

4.10.1. Relational Database Diagram

Para esta solución se realizó una Base de datos relacional con las siguientes tablas:



Con esta estructura de la base datos que se basa en el modelo DDD de la arquitectura de la aplicación, se logrará tener un manejo estable de los datos.

Capítulo V: Product Implementation

5.1. Software Configuration Management

5.1.1. Software Development Environment Configuration

- **Android Studio:** Entorno de desarrollo.



- **GitHub:** Repositorio colaborativo en la nube.



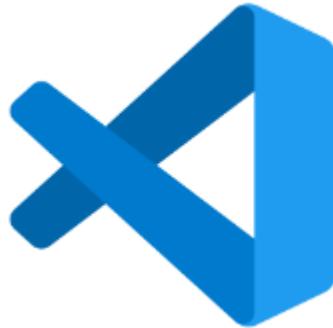
- **Netlify:** Plataforma que facilita implementar despliegues sencillos para nuestras páginas web.



- **Vertabelo:** Plataforma colaborativa para la creación de diagramas de base de datos.



- **Visual Studio Code:** Entorno de desarrollo para diseño de base de datos.



- **Figma:** Herramienta colaborativa que permite elaborar wireframes y mockups.



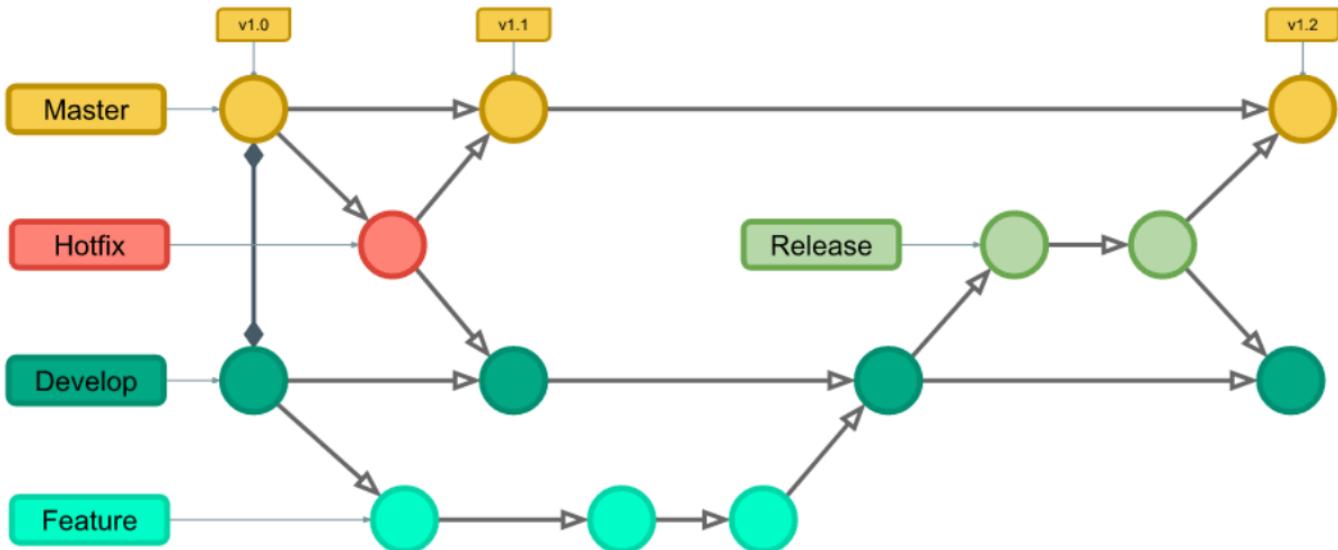
- **Azure:** Herramienta para subir nuestros servicios web en la nube.



5.1.2. Source Code Management

Repositorio de la Landing Page: Implementación de GitFlow: Para nuestra estrategia de gestión de versiones con Git, nos hemos inspirado en el artículo "A successful Git branching model" de Vincent Driessen, adoptando el modelo de ramificación GitFlow. Este enfoque nos permite establecer claramente

las convenciones de ramificación que aplicamos en nuestro proyecto.



- **Rama Principal (Main branch):** Contiene el código en producción y se conoce como la Master branch o Main branch.
 - Notación: main
- **Rama de Desarrollo (Develop branch):** Acumula las últimas actualizaciones y cambios para la próxima versión. Funciona como un entorno de integración y prueba continua.
 - Notación: develop
- **Rama de Lanzamiento (Release branch):** Facilita la preparación de una nueva versión del producto, permitiendo correcciones de errores y recibiendo más actualizaciones de Develop.
 - Debe derivarse de: develop
 - Debe fusionarse con: develop y master/main
 - Notación: release
- **Rama de Características (Feature branch):** Se utiliza para desarrollar nuevas funcionalidades para la siguiente versión o futuras iteraciones.
 - Debe derivarse de: develop
 - Debe fusionarse de vuelta a: develop
 - Notación: feature
- **Rama de Corrección Rápida (Hotfix branch):** Aborda errores críticos en producción, permitiendo la implementación rápida de soluciones.
 - Debe derivarse de: master/main
 - Debe fusionarse con: develop y master/main
 - Notación: hotfix

Conventional Commits: Adoptamos esta metodología para estructurar los mensajes de confirmación de cambios de manera estándar y semántica, lo que facilita la comunicación y la automatización de registros de cambios. **Tipos de Commits Convencionales:**

- feat: Nuevas características o funcionalidades.
- fix: Correcciones de errores.
- docs: Cambios o mejoras en la documentación.
- style: Cambios de formato que no afectan la funcionalidad.
- refactor: Mejoras en la estructura o legibilidad del código.

- test: Adición o modificación de pruebas.
- chore: Cambios en el proceso de construcción o tareas de mantenimiento.
- perf: Mejoras de rendimiento en el código.

5.1.3. Source Code Style Guide & Conventions

HTML

Regla	Ejemplo / Explicación
Etiquetas y atributos en minúsculas	<code><div class="container">, </code>
Atributos ordenados lógicamente	<code>class, id, name, type, value, etc.</code>
Uso de comillas dobles	<code><input type="text" name="username"></code>
Indentación consistente (2 o 4 espacios)	No mezclar espacios con tabs

CSS

Regla	Ejemplo / Explicación
Nombres de clases en kebab-case	<code>.main-header, .user-profile-card</code>
Propiedades en minúsculas y ordenadas	<code>color: #333; font-size: 16px; margin-top: 20px;</code>
Uso de comentarios	<code>/* Sección de estilos para el header */</code>
Indentación consistente	2 o 4 espacios, no usar tabs

JavaScript

Regla	Ejemplo / Explicación
Variables y funciones en camelCase	<code>let userName = "Juan"; function getUserData() {}</code>
Clases en PascalCase	<code>class UserProfile {}</code>
Constantes en UPPER_SNAKE_CASE	<code>const API_URL = "https://api.example.com";</code>
Uso de <code>const</code> y <code>let</code>	Evitar <code>var</code> , usar <code>const</code> por defecto y <code>let</code> si se necesita mutabilidad
Punto y coma al final de líneas	<code>let nombre = "Carlos";</code>
Indentación consistente (2 o 4 espacios)	Mantener el mismo estilo en todo el proyecto

Kotlin

Regla	Ejemplo / Explicación
Variables y funciones en camelCase	<code>val userName = "Juan", fun getUserData() {}</code>
Clases y objetos en PascalCase	<code>class UserProfile, object AppConfig</code>
Constantes en UPPER_SNAKE_CASE	<code>const val MAX_USERS = 100</code>
Archivos nombrados igual que la clase	<code>UserProfile.kt</code>
Indentación con 4 espacios	No usar tabs
Uso de <code>val</code> por defecto, <code>var</code> si mutable	Promueve inmutabilidad
Expresiones lambda con <code>it</code>	<code>users.filter { it.isActive }</code>

5.1.4. Software Deployment Configuration

Deployment Landing Page: En esta sección, detallamos el proceso de implementación de nuestra landing page en la plataforma de GitHub.

1. Se crea un repositorio en GitHub para alojar el código de nuestra landing page.

The screenshot shows a GitHub repository interface. At the top, there's a header with the repository name 'Landing-Page----VacApp'. Below the header, there are navigation links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The main area displays the repository's content, including a file tree and a list of commits. The file tree shows directories like 'main', 'public', 'src', and files like '.gitignore', 'README.md', 'eslint.config.js', 'index.html', 'package-lock.json', 'package.json', 'tsconfig.app.json', 'tsconfig.json', 'tsconfig.node.json', and 'vite.config.ts'. The commit list includes entries from 'FrancoDurand' and others, with details like 'Merge branch 'develop'', 'feat: add dependencies', and dates like '3 days ago' or '5 days ago'. To the right of the code area, there are sections for 'About' (describing it as the landing page for VacApp), 'Releases' (none published), 'Packages' (none published), and 'Contributors' (two listed). A footer at the bottom indicates the repository is built with 'React + TypeScript + Vite'.

2. Agregamos a los participantes:

The screenshot shows the GitHub Insights page for the repository 'Landing-Page---VacApp'. It displays a list of users with access to the repository, categorized by 'Everyone' and 'Outside collaborators'. Each user entry includes a profile picture, name, GitHub handle, and their role ('Admin').

User	Role
Sergio André Gómez Vallejos CB-Sergio-AGV	Admin
Estrella-ticona	Admin
FrancoDurand	Admin
Myck Kmykh	Admin
Aranda Vallejos, Oscar Gabriel OscarGAV	Admin
Piero Miranda PieroMiranda1	Admin

3. Habilitamos Netlify para poder importar nuestro proyecto:

The screenshot shows the Netlify dashboard for the project 'Landing-Page---KingReserve'. It displays the 'Functions directory' set to 'netlify/functions', environment variables, and a deployment status for the 'main' branch.

Functions directory: netlify/functions

Environment variables:
Define environment variables for more control and flexibility over your build.
Add environment variables

Deploy Landing-Page---VacApp

Git repository: Landing-Page---KingReserve
Deploying: main

Docs Pricing Support Blog Changelog Terms
© 2024 Netlify
<https://app.netlify.com/teams/kmykh>

4. Finalmente, se confirma el despliegue de nuestra página web después de completar todo el procedimiento.



Este proceso garantiza el despliegue satisfactorio de nuestra landing page en la plataforma de Netlify, siguiendo las especificaciones y requisitos de nuestro proyecto. **Enlace de la Landing Page:**

<https://vacapp-landing.netlify.app/>

About the product: <https://www.youtube.com/watch?v=JmOW2IkXjeI>

Deployment Frontend: En esta sección, detallamos el proceso del deploy del Frontend-web en la plataforma de Firebase.

1. Al utilizar Firebase instalamos firebase en el proyecto



2. Luego de Instalarlo se inicia sesion y se implementa en Firebase.



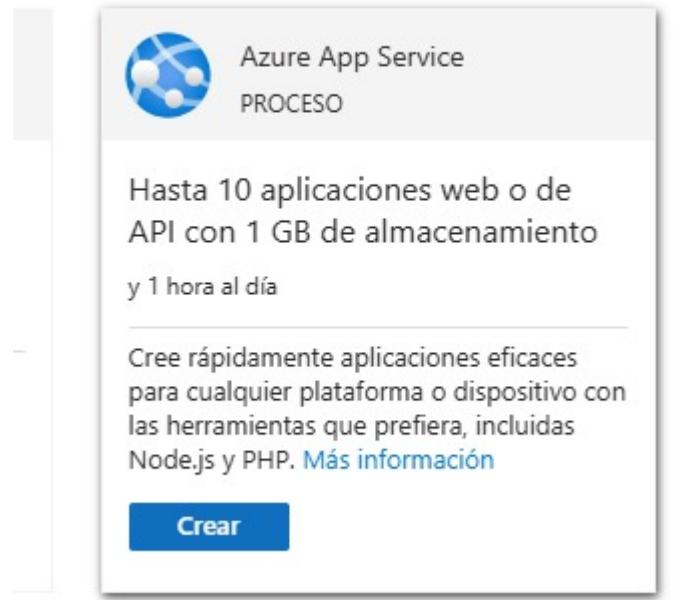
3. Con esto, la aplicacion fue desplegada.



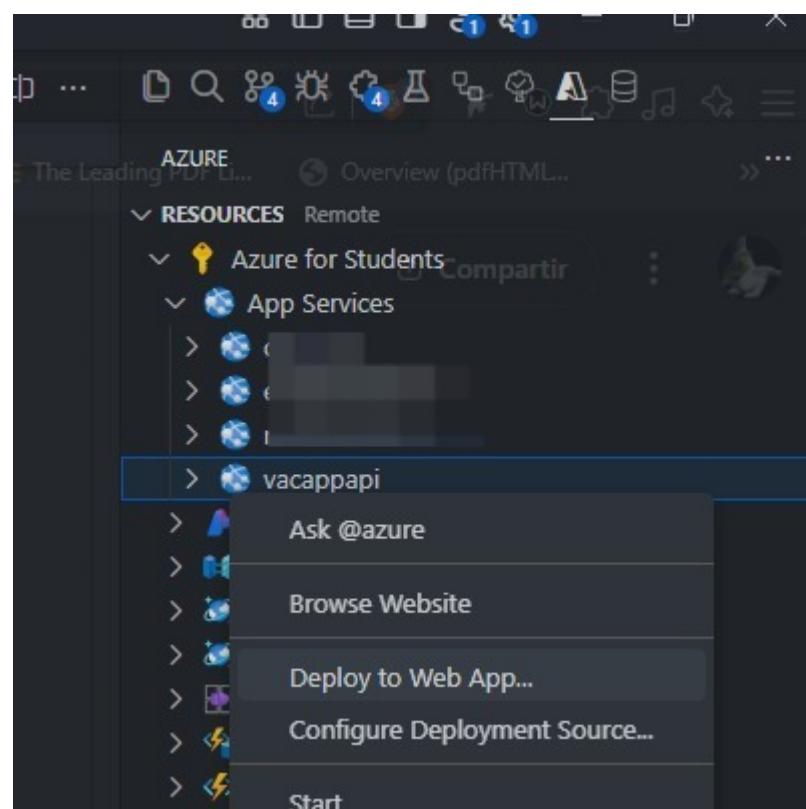
Enlace del Frontend: <https://vacapp-frontend.web.app/auth/login>

Deployment Backend: En esta sección, detallamos el proceso de implementación de nuestro backend en la plataforma de Microsoft Azure.

1. Se crea un servicio web alojado en azure y afiliado a un grupo de recursos determinado



2. Con la herramienta Azure Tool Kit, un plugin disponible en los entornos de desarrollo de jetbrains, podemos habilitar una vista con los recursos y elementos creados en nuestra cuenta de Azure. Esto nos permite poder publicar el backend directamente al servicio en la nube de Azure.



3. Una vez realizado de manera satisfactoria este proceso, resolviendo errores y añadiendo configuraciones adicionales de ser requeridas, podremos confirmar que el enlace muestre correctamente los endpoints y observaremos nuestro backend desplegado en un browser predeterminado.

VacApp-Bovinova-Platform 1.0 OAS 3.0

https://vacappapi.azurewebsites.net/swagger/v1/swagger.json

Authorize

Bovines

- POST** /api/v1/bovines
- GET** /api/v1/bovines Get all bovines
- GET** /api/v1/bovines/{id}
- PUT** /api/v1/bovines/{id}
- DELETE** /api/v1/bovines/{id}
- GET** /api/v1/bovines/stable/{stableId} Get all bovines by stable ID

Enlace del backend: <https://vacappapi.azurewebsites.net/swagger/index.html>

5.2. Product Implementation & Deployment

5.2.1. Sprint Backlogs

Sprint 1

User Story Id	User Story Title	Task Id	Task Title	Estimation	Assigned To	Status
TS015	Acceso a la sección de Home	T01	Implementar acceso a la sección de Home	2	Ticona Panduro, Estrella del Pilar	Done
TS016	Acceso a la sección de Características	T02	Implementar acceso a la sección de Características	1	Durand Vera, Gianfranco Angel	Done
TS017	Registro de Nuevos Usuarios	T03	Implementar registro de nuevos usuarios	3	Ticona Panduro, Estrella del Pilar	Done
TS018	Información de Funcionalidades	T04	Implementar sección de funcionalidades	2	Durand Vera, Gianfranco Angel	Done

User Story Id	User Story Title	Task Id	Task Title	Estimation	Assigned To	Status
TS019	Sector de Planes Disponibles	T05	Implementar sector de planes disponibles	3	Durand Vera, Gianfranco Angel	Done
TS020	Incluir Internacionalización (i18n)	T06	Implementar el cambio de idioma	2	Ticona Panduro, Estrella del Pilar	Done
TS001	Crear Vacuna vía API	T07	Implementar POST para vacunas	2	Aranda Vallejos, Oscar Gabriel	Done
TS002	API para Búsqueda de Vacunas	T08	Implementar GET para vacunas	2	Aranda Vallejos, Oscar Gabriel	Done
TS003	API para Gestión de vacunas	T09	Implementar PUT y DELETE para vacunas	3	Aranda Vallejos, Oscar Gabriel	Done
TS004	API para Registro de animales	T10	Implementar POST para animales	2	Aranda Vallejos, Oscar Gabriel	Done
TS005	API para Búsqueda de animales	T11	Implementar GET para animales	2	Aranda Vallejos, Oscar Gabriel	Done
TS006	API para Gestión de animales	T12	Implementar PUT y DELETE para animales	2	Aranda Vallejos, Oscar Gabriel	Done
TS007	API para Creación de campaña	T13	Implementar POST para campaña	2	Miranda Sinarahua, Piero Stephano	Done

User Story Id	User Story Title	Task Id	Task Title	Estimation	Assigned To	Status
TS009	API para Gestión de campaña	T14	Implementar PUT y DELETE para campaña	2	Miranda Sinarahua, Piero Stephano	Done
TS012	API para Registro de empleados	T15	Implementar POST para empleados	2	Aranda Vallejos, Oscar Gabriel	Done

Sprint 2

User Story Id	User Story Title	Task Id	Task Title	Estimation	Assigned To	Status
TS001	Crear Vacuna vía API	T01	Implementar exponer un endpoint para registrar una vacuna vía API	2	Ticona Panduro, Estrella del Pilar	Done
TS002	API para Búsqueda de Vacunas	T02	Implementación de un endpoint para buscar vacunas mediante criterios específicos	1	Durand Vera, Gianfranco Angel	Done
US002	Búsqueda de Vacunas	T03	Implementar búsqueda de vacunas previamente registradas	3	Ticona Panduro, Estrella del Pilar	Done
TS003	API para Gestión de Vacunas	T04	Implementar endpoint para editar y eliminar registros de vacunas	2	Durand Vera, Gianfranco Angel	Done
TS004	API para Registro de Animales	T05	Implementar endpoint para registrar un bovino en un lote específico	3	Durand Vera, Gianfranco Angel	Done
US003	Gestión de Registros de Vacunas	T06	Implementar editar o eliminar el registro de una vacuna	2	Ticona Panduro, Estrella del Pilar	Done

User Story Id	User Story Title	Task Id	Task Title	Estimation	Assigned To	Status
TS005	API para Búsqueda de Animales	T07	Implementar un endpoint que permita buscar animales registrados usando parámetros de búsqueda	2	Aranda Vallejos, Oscar Gabriel	Done
TS006	API para Gestión de Animales	T08	Implementar funcionalidades para editar y eliminar animales registrados	2	Aranda Vallejos, Oscar Gabriel	Done
US004	Registro de Bovino en Lote	T09	Implementar un registro de un bovino en un lote específico	3	Aranda Vallejos, Oscar Gabriel	Done
TS007	API para Creación de Campaña	T10	Implementar un endpoint que permita la creación de campañas	2	Gómez Vallejos, Sergio André	Done

Sprint 3

User Story Id	User Story Title	Task Id	Task Title	Estimation	Assigned To	Status
US01	Agregar vacuna al registro	T01	Implementar la opción de registro de vacuna	2	Rojas Velasquez, Maycol Jhordan	Done
US03	Gestión de registro de vacunas	T02	Implementación de la administración del registro de vacunas en el app	2	Rojas Velasquez, Maycol Jhordan	Done
US04	Registro de bovino en Lote	T03	Implementación de registro de bovinos dentro de un Lote	2	Rojas Velasquez, Maycol Jhordan	Done
US05	Información de bovino	T04	Implementación de la opción de visualizar información de bovino	2	Durand Vera, Gianfranco Angel	Done

User Story Id	User Story Title	Task Id	Task Title	Estimation	Assigned To	Status
US06	Actualiza información de bovinos	T05	Implementación de la opción de actualizar datos de bovino	2	Durand Vera, Gianfranco Angel	Done
US08	Asocia Empleado a campaña	T06	Implementación de la opción de asociar empleado a una campaña	2	Rojas Velasquez, Maycol Jhordan	Done
TS09	API para Gestión de Campañas	T07	Implementación de API para gestión de campañas	2	Miranda Sinarahua, Piero Stephano	Done
US10	Registro de Personal	T08	Implementación de opción para registrar personal	2	Ticona Panduro, Estrella del Pilar	Done
TS06	API gestión de animales	T09	Implementación de API para gestión de animales	2	Aranda Vallejos, Oscar Gabriel	Done
US12	Gestión de Personal	T10	Implementación de opción para administrar el personal	2	Aranda Vallejos, Oscar Gabriel	Done
TS11	API búsqueda empleados	T11	Implementación del API para búsqueda de personal	2	Aranda Vallejos, Oscar Gabriel	Done
US15	Explorar Landing Page	T12	Implementación secciones restantes para explorar Landing Page	2	Gómez Vallejos, Sergio André	Done

5.2.2. Implemented Landing Page Evidence

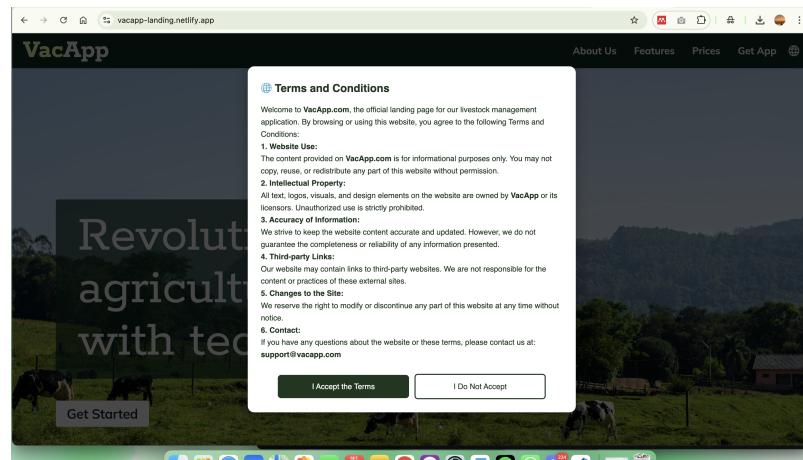
Durante el desarrollo de los Sprints, se completó y desplegó exitosamente la landing page del proyecto VacApp. Esta página presenta el modelo de negocio, integra una barra de navegación funcional, secciones informativas clave y un formulario de contacto operativo que permite a los usuarios dejar su información de manera efectiva.

El sitio fue desarrollado aplicando principios de Responsive Web Design para asegurar una experiencia de usuario óptima en dispositivos móviles, tabletas y computadoras de escritorio. Las pruebas de visualización en múltiples resoluciones confirmaron su correcto funcionamiento.

Asimismo, se implementó la metodología GitFlow, lo que permitió organizar eficientemente el trabajo del equipo mediante ramas específicas para desarrollo, pruebas y producción, asegurando la estabilidad de la rama principal.

La landing page de VacApp se encuentra publicada y accesible en el siguiente enlace: <https://vacapp-landing.netlify.app/>

A continuación, se presentan las imágenes que evidencian los avances logrados durante este Sprint:



Precios

Gratis	Premium
S/0	Desde S/49.90
<ul style="list-style-type: none"> • Registro de hasta 10 animales. • Acceso básico al historial sanitario. • Recordatorios limitados. • Visualización de reportes mensuales en formato simple. • Soporte por correo electrónico. 	<ul style="list-style-type: none"> • Registro ilimitado de animales. • Seguimiento completo de salud y vacunación. • Gestión de inventario y productos. • Soporte prioritario.

Funciones Clave

Registro de Animales Registra y gestiona ganado con información detallada de origen, raza y estado.	Historial de Salud Lleva el control de vacunas y tratamientos médicos para cada animal.	Recordatorios Inteligentes Alertas automáticas para cheques, tratamientos y eventos importantes.
Reportes Automatizados Genera reportes visuales	Recetas Médicas Emite y guarda recetas	Gestión de Inventario Monitorea suministros como

5.2.3. Implemented Frontend-Web Application Evidence

Además del desarrollo de la aplicación móvil de VacApp, se desarrolló paralelamente la **aplicación web** como parte de la estrategia multiplataforma del proyecto. Esta implementación web busca ofrecer una experiencia completa y funcional que permita a los usuarios acceder a todas las funcionalidades desde cualquier dispositivo con navegador.

Características de la Aplicación Web

La aplicación web de VacApp ha sido desarrollada siguiendo los principios de **Responsive Web Design**, garantizando una experiencia óptima tanto en dispositivos de escritorio como en tablets. Las funcionalidades implementadas incluyen:

Gestión Integral del Ganado:

- Registro y actualización de información de bovinos
- Control de salud y seguimiento veterinario
- Gestión de ciclos reproductivos y genealogía
- Trazabilidad completa de cada animal

Administración de Establos:

- Organización y capacidad de instalaciones
- Asignación de animales por establo
- Control de ocupación y distribución

Campañas de Vacunación:

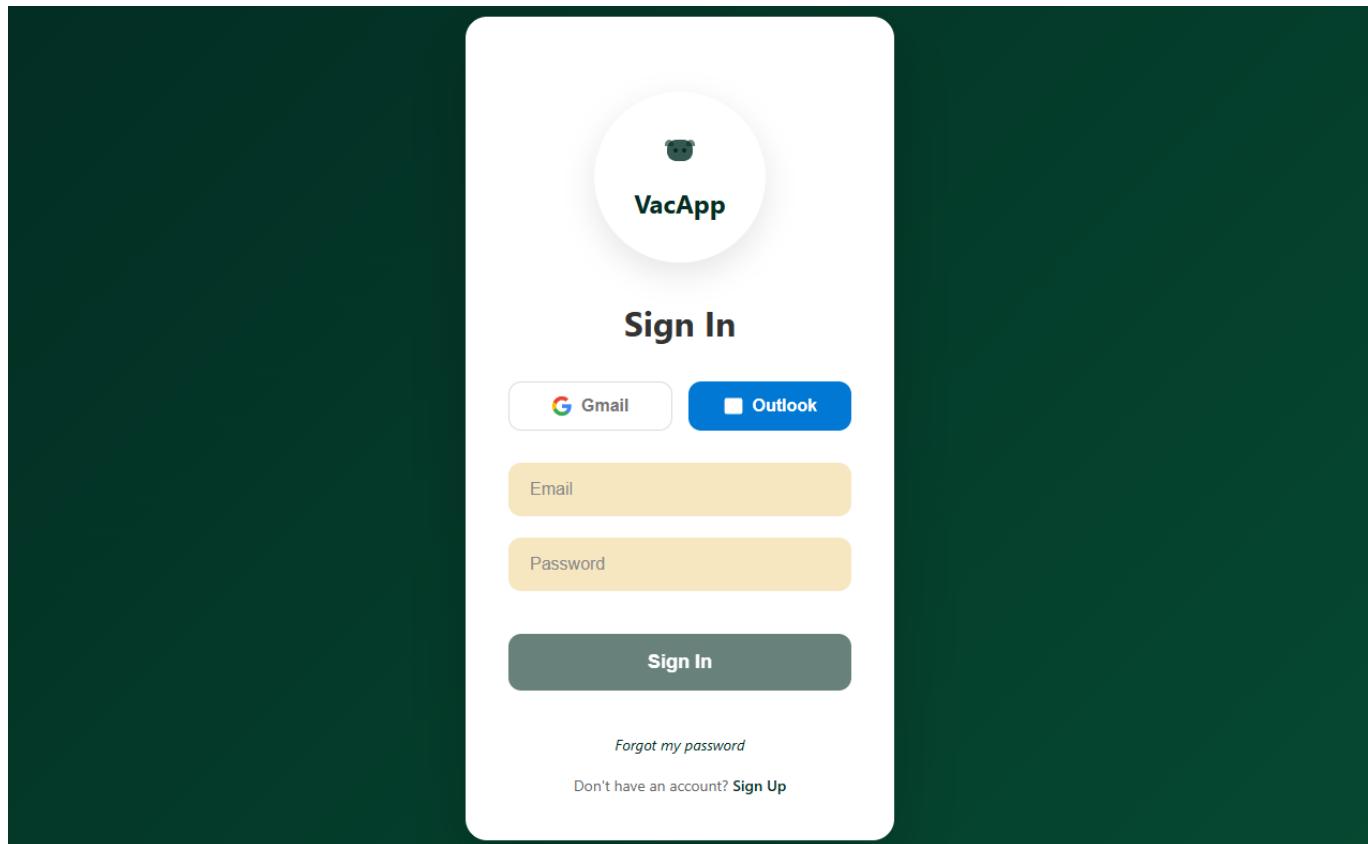
- Planificación de campañas sanitarias
- Programación de vacunas y tratamientos
- Seguimiento del progreso de campañas

Dashboard y Reportes:

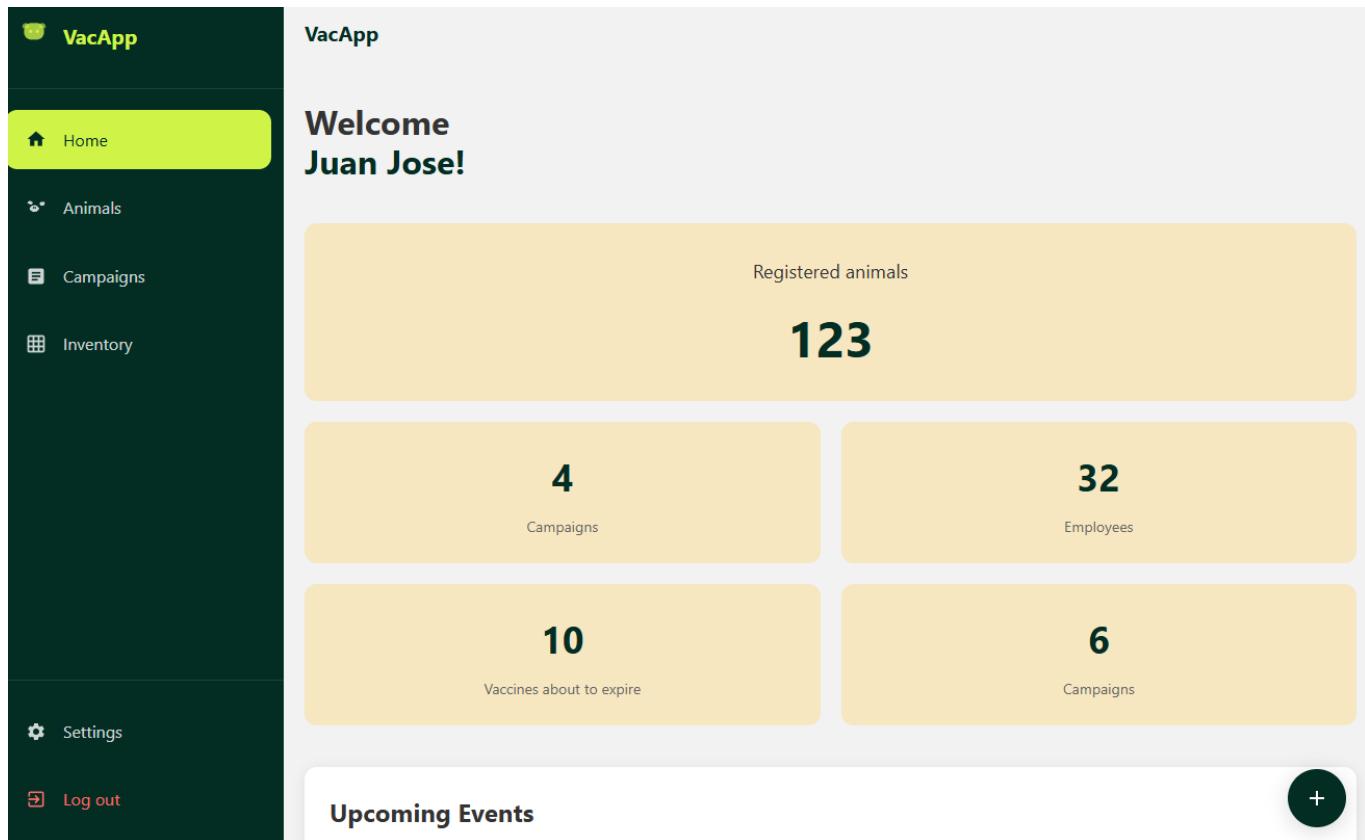
- Panel de control con métricas clave
- Reportes de productividad y salud
- Análisis de tendencias y estadísticas

Evidencias de Implementación

Interfaz de Inicio de Sesión: La pantalla de autenticación implementa las mejores prácticas de seguridad y UX, con validación en tiempo real y diseño responsive.



Dashboard Principal: El panel principal muestra métricas importantes, accesos rápidos a funciones clave y una navegación intuitiva que facilita el trabajo diario del ganadero.



Tecnologías Implementadas

- **Frontend:** React.js con TypeScript
- **UI Framework:** Angular Material / Bootstrap
- **Estado:** Redux Toolkit para gestión de estado
- **Autenticación:** JWT con refresh tokens
- **Responsive:** CSS Grid y Flexbox
- **Build:** Vite para optimización de rendimiento

Acceso y Deployment

La aplicación web está desplegada en **Firebase Hosting**, garantizando alta disponibilidad, CDN global y certificados SSL automáticos.

🔗 **Enlace de acceso:** <https://vacapp-frontend.web.app/auth/login>

Características del deployment:

- Carga rápida con optimización automática
- Acceso global mediante CDN
- HTTPS habilitado por defecto
- Compatible con PWA (Progressive Web App)

Próximas Funcionalidades

En las siguientes iteraciones se planea implementar:

- Módulo avanzado de reportes y analytics
- Sistema de notificaciones push para web

- Captura y gestión de imágenes del ganado
- Integración con mapas para ubicación de establos
- Funcionalidades offline con Service Workers

5.2.4 Acuerdo de Servicio SaaS

Con el propósito de garantizar la transparencia, la protección de los usuarios y el cumplimiento de las normas nacionales e internacionales aplicables al sector agropecuario peruano, se define el presente **Acuerdo de Servicio (SaaS)** de VacApp, una plataforma digital destinada a la **gestión integral de la producción ganadera**.

VacApp ofrece herramientas tecnológicas que facilitan la digitalización del control del ganado, las campañas sanitarias, la gestión de insumos y el monitoreo productivo, fomentando así una **ganadería más moderna, sostenible y eficiente** en el Perú.

Este acuerdo regula los **derechos, responsabilidades y limitaciones** de los usuarios, y establece el marco legal bajo el cual deben interactuar con el servicio. El mismo se encontrará disponible en la sección “*Términos y Condiciones*” de VacApp, siendo obligatoria su aceptación antes de utilizar la aplicación.

Objeto del Servicio

VacApp proporciona un entorno digital accesible vía web y móvil para **productores ganaderos, asociaciones pecuarias y técnicos del sector agropecuario**.

A través de la plataforma, los usuarios pueden:

- Registrar y monitorear la salud, alimentación, peso y producción de sus bovinos.
- Administrar campañas de vacunación y tratamientos veterinarios.
- Gestionar establos, personal y almacenes de insumos pecuarios.
- Trabajar **en modo offline**, con sincronización automática al recuperar conexión.

El propósito principal de VacApp es **optimizar la toma de decisiones ganaderas**, reducir pérdidas y contribuir a la sostenibilidad del sector, alineándose con los lineamientos del **Ministerio de Desarrollo Agrario y Riego (MIDAGRI)** y la **Ley N.º 29733 de Protección de Datos Personales**.

Obligaciones del Usuario

Los usuarios que accedan a VacApp se comprometen a:

1. **Proporcionar información veraz y actualizada** sobre su hato ganadero, campañas y operaciones productivas.
2. **Usar la plataforma de forma ética y responsable**, evitando el uso indebido de datos o acciones que afecten la integridad del sistema.
3. **Cumplir con las normas pecuarias y sanitarias vigentes en el Perú**, incluyendo las disposiciones del SENASA.
4. **Respetar los derechos de propiedad intelectual y privacidad**, absteniéndose de compartir información ajena o confidencial sin autorización.

Obligaciones del Proveedor (VacApp)

VacApp se compromete a:

- **Proveer acceso continuo y seguro** al sistema, salvo interrupciones justificadas por mantenimiento o fuerza mayor.
- **Garantizar la confidencialidad y seguridad de los datos personales y productivos** de los usuarios, conforme a la Ley N.º 29733 y estándares de ciberseguridad (ISO/IEC 27001).
- **Brindar soporte técnico y atención al usuario**, mediante canales digitales oficiales.
- **Informar oportunamente** sobre actualizaciones, cambios en los términos del servicio o nuevas funcionalidades.

Restricciones de Uso

Para asegurar un entorno seguro y justo, los usuarios **no podrán**:

- Utilizar la plataforma con fines comerciales externos al rubro ganadero o para actividades ilícitas.
- Modificar, copiar o redistribuir el software de VacApp sin autorización previa.
- Publicar contenido ofensivo, fraudulento o que infrinja derechos de terceros.
- Manipular datos o usar herramientas automatizadas (bots, scrapers) que afecten la integridad del servicio.

Propiedad Intelectual

Los **datos e información generados por los usuarios** (como registros de animales, vacunas o reportes) seguirán siendo de su propiedad.

Sin embargo, al utilizarlos dentro de la aplicación, los usuarios otorgan a VacApp una **licencia no exclusiva y gratuita** para procesar, analizar y mostrar dichos datos con fines operativos y estadísticos.

La **propiedad intelectual del software, base de datos, interfaz y algoritmos** pertenece exclusivamente a VacApp, quedando protegida por la legislación peruana sobre derechos de autor y propiedad industrial.

Modificaciones del Servicio

VacApp podrá realizar **mejoras o actualizaciones del servicio** que optimicen su funcionamiento o amplíen sus capacidades.

Cualquier cambio relevante será notificado a los usuarios mediante correo electrónico o notificaciones dentro de la aplicación.

Terminación de Cuentas

VacApp se reserva el derecho de **suspender o eliminar cuentas** que incumplan este acuerdo, introduzcan datos falsos o atenten contra la seguridad del sistema.

Los usuarios podrán solicitar la eliminación de sus datos conforme a los principios de **autodeterminación informativa** y derecho al olvido previstos por la legislación peruana.

Marco Normativo de Referencia

- Ley N.º 29733 – *Ley de Protección de Datos Personales (Perú)*.
- Reglamento de la Ley N.º 29733 – D.S. N.º 003-2013-JUS.
- Ley N.º 30494 – *Ley de Promoción y Desarrollo de la Ganadería Sostenible*.
- Normas Técnicas del SENASA sobre sanidad animal y trazabilidad ganadera.
- Estándares internacionales ISO/IEC 27001 y 27017 sobre seguridad y gestión de servicios SaaS.

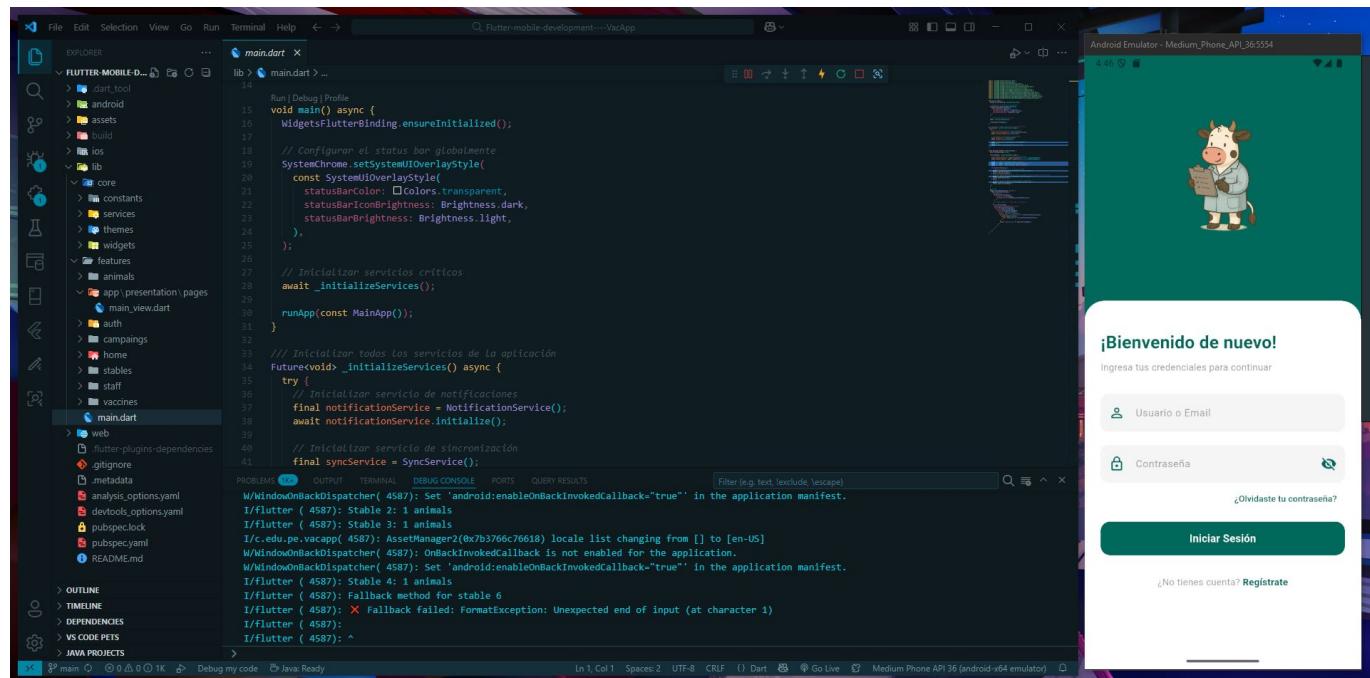
Este **Acuerdo de Servicio SaaS** constituye un componente esencial del ecosistema digital de VacApp y refleja el compromiso con una **ganadería peruana moderna, responsable y tecnológica**, alineada con la sostenibilidad y el bienestar animal.

5.2.5. Implemented Native-Mobile Application Evidence

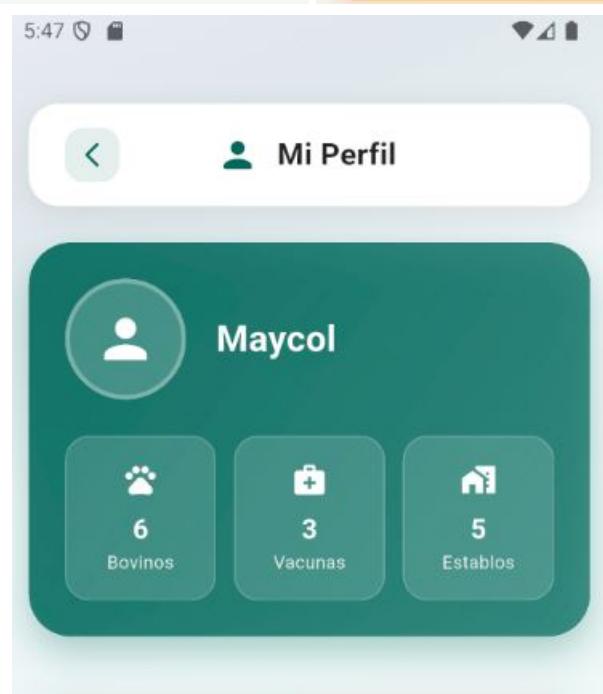
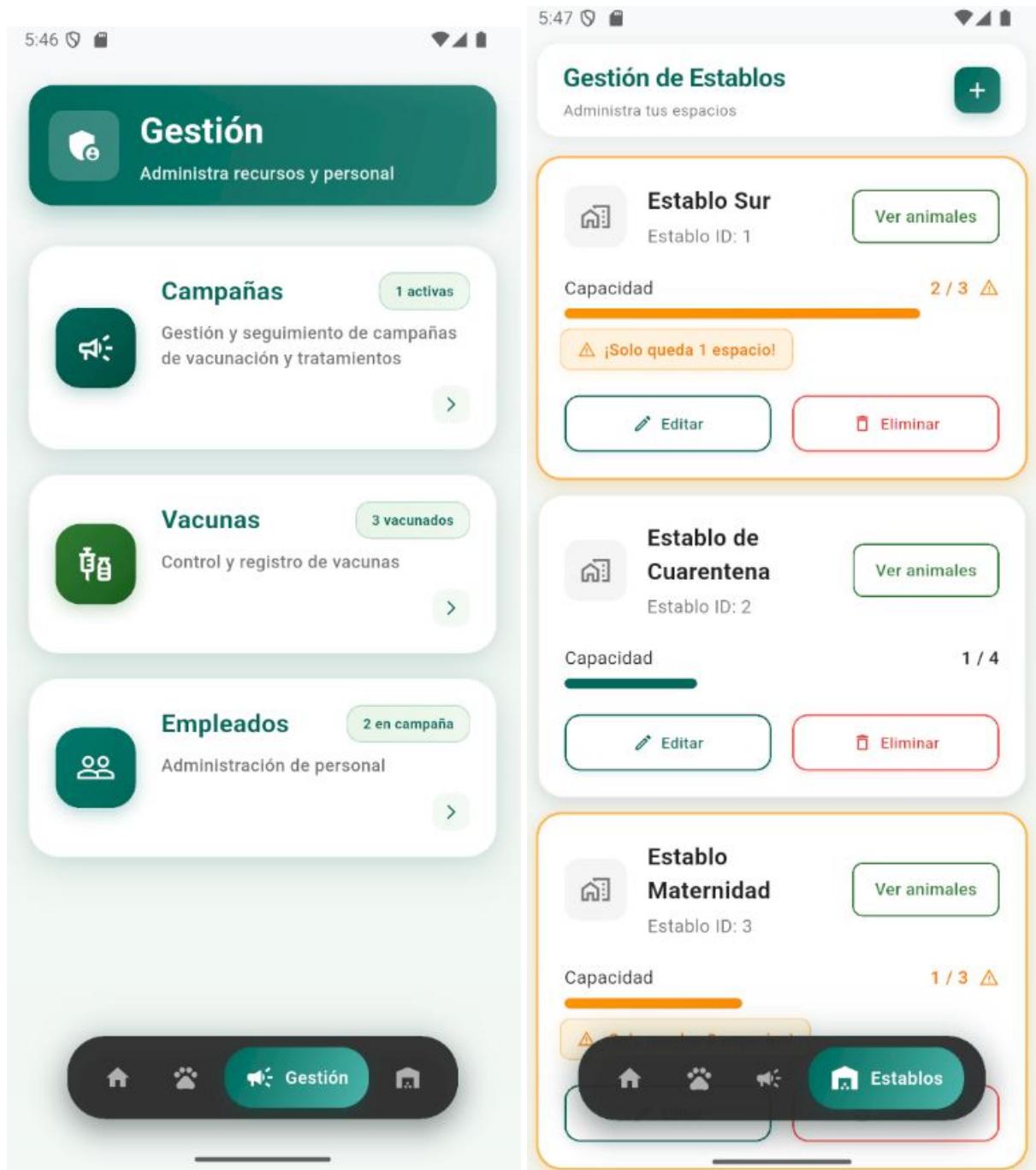
Durante los Sprints del proyecto, se logró desarrollar e implementar por completo la aplicación móvil de VacApp, cumpliendo con las funcionalidades clave definidas para la solución. La app integra de manera efectiva las características relacionadas con la gestión ganadera, permitiendo a los usuarios registrar bovinos, gestionar vacunas, organizar establos, y realizar un seguimiento de la salud y productividad del ganado.

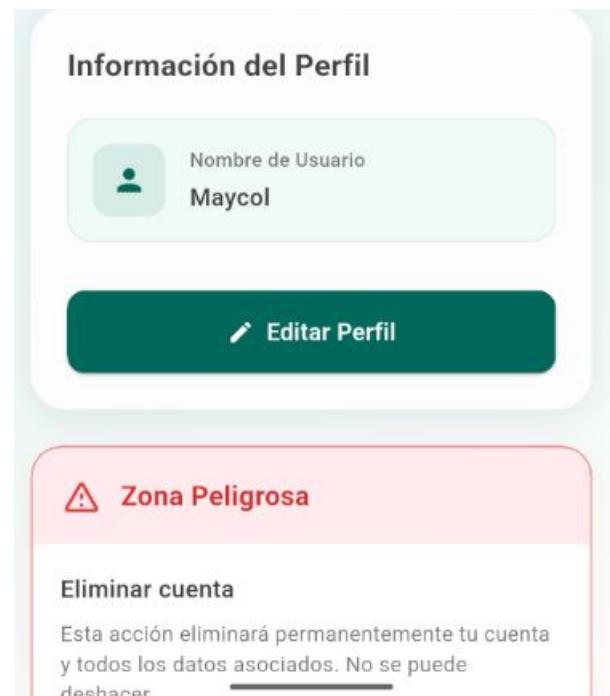
Además, se incluyeron vistas para la gestión de usuarios, control de campañas de vacunación, reportes y estadísticas, asegurando una experiencia de usuario coherente y fluida en dispositivos móviles.

Este desarrollo fue acompañado por pruebas funcionales continuas durante los sprints, garantizando el cumplimiento de los requisitos establecidos y la alineación con los flujos definidos previamente en los prototipos elaborados en Figma. La aplicación móvil de VacApp se encuentra lista para ser validada en un entorno real de usuarios y continuar su evolución en próximos ciclos de mejora.









Personal
Gestión de empleados

Jutmid Vargas

Estado: Trabajando / En Campaña

Campaña Asignada: Campaña para Cuidado Veterinarios

DURACIÓN: 15 días de trabajo

Inicio: 30 Jul 2025 | Fin: 13 Ago 2025

Ver Acceso

Panel de Control

Alexandra Vilcatoma

Estado: Vacaciones

Personal en Vacaciones: Concluirá al volver de sus vacaciones la campaña

+ Agregar

Vacunas
3 vacunas

Buscar vacunas...

Salmonella Dublin
Bacteriana

Mara
Hembra • 14 años | Vacunado

Fecha de Vencimiento: 26 jun 2030

Editar | **Eliminar**

Fiebre Aftosa
Viral

Lola
Hembra • 17 años | Vacunado

Fecha de Vencimiento: 23 jul 2026

+ Nueva Vacuna



5.2.6. Implemented RESTful API and/or Serverless Backend Evidence

Backend

Endpoint para registro e inicio de sesión

User

POST	/api/v1/user/sign-up	
POST	/api/v1/user/sign-in	
GET	/api/v1/user/get-info	

Endpoint de establos**Stables**

POST	/api/v1/stables	
GET	/api/v1/stables Get all stables	
GET	/api/v1/stables/{id}	
PUT	/api/v1/stables/{id}	
DELETE	/api/v1/stables/{id}	

Endpoint de bovinos

https://vacappapi.azurewebsites.net/swagger/index.html

Swagger Supported by SMARTBEAR Select a definition VacApp-Bovinova-Platform v1

VacApp-Bovinova-Platform 1.0 OAS 3.0

https://vacappapi.azurewebsites.net/swagger/v1/swagger.json

Authorize

Bovines		
POST	/api/v1/bovines	
GET	/api/v1/bovines Get all bovines	
GET	/api/v1/bovines/{id}	
PUT	/api/v1/bovines/{id}	
DELETE	/api/v1/bovines/{id}	
GET	/api/v1/bovines/stable/{stableId} Get all bovines by stable ID	

Endpoint de vacunas

Vaccines	
POST	/api/v1/vaccines
GET	/api/v1/vaccines Get all vaccines
GET	/api/v1/vaccines/{id}
PUT	/api/v1/vaccines/{id}
DELETE	/api/v1/vaccines/{id}
GET	/api/v1/vaccines/bovine/{bovineId} Get all vaccines by bovine ID

Endpoint de campañas

Campaign

<code>POST</code>	/api/v1/campaign	
<code>GET</code>	/api/v1/campaign/{id}	
<code>DELETE</code>	/api/v1/campaign/{id}	
<code>GET</code>	/api/v1/campaign/all-campaigns	
<code>PATCH</code>	/api/v1/campaign/{id}/update-status	
<code>PATCH</code>	/api/v1/campaign/{id}/add-goal	
<code>PATCH</code>	/api/v1/campaign/{id}/add-channel	
<code>GET</code>	/api/v1/campaign/{id}/goals	
<code>GET</code>	/api/v1/campaign/{id}/channels	

Endpoint de google auth

UserGoogle

<code>GET</code>	/api/v1/user-google	
<code>GET</code>	/api/v1/user-google/callback	
<code>GET</code>	/api/v1/user-google/sign-in/{userId}	

Endpoint de microsoft auth

UserOutlook

<code>GET</code>	/api/v1/user-outlook	
<code>GET</code>	/api/v1/user-outlook/callback	
<code>GET</code>	/api/v1/user-outlook/sign-in/{userId}	

5.2.7. RESTful API documentation

La documentación de la **API RESTful** del backend del proyecto **VacApp** fue desarrollada siguiendo las mejores prácticas de la industria, utilizando herramientas especializadas como **Swagger (OpenAPI 3)** y **Postman**. Esta documentación integral facilita a los desarrolladores la consulta, comprensión y prueba de los endpoints disponibles, optimizando la integración con el frontend y otros sistemas externos.

Herramientas de Documentación

Swagger (OpenAPI 3)

Swagger constituye la herramienta principal para la generación automática de documentación interactiva de la API. Esta plataforma proporciona una interfaz visual completa que incluye:

Características principales:

- Especificación detallada de endpoints:** Tipo de solicitud (GET, POST, PUT, DELETE)
- Validación de parámetros:** Documentación exhaustiva de parámetros de entrada y sus validaciones
- Ejemplos de respuesta:** Casos de uso en formato JSON para cada endpoint
- Manejo de errores:** Documentación completa de códigos de estado HTTP y mensajes de error

- **Testing integrado:** Capacidad de probar endpoints directamente desde la interfaz sin herramientas externas

Postman Collection

Postman complementa la documentación de Swagger mediante la realización de pruebas exhaustivas y validaciones de escenarios reales:

Funcionalidades implementadas:

- **Pruebas manuales automatizadas** para validar el comportamiento de la API
- **Validación de escenarios complejos** que incluyen casos de éxito y error
- **Testing de integración** para asegurar la correcta comunicación entre servicios
- **Documentación de casos de uso** específicos del dominio ganadero

Estructura de la API

Sistema de Autenticación

VacApp implementa un sistema de autenticación robusto basado en **tokens JWT (JSON Web Tokens)** que garantiza la seguridad y trazabilidad de las operaciones:

Endpoints de autenticación:

- **Registro:** POST /api/v1/authentication/sign-up
- **Inicio de sesión:** POST /api/v1/authentication/sign-in

Configuración de headers:

```
Authorization: Bearer <JWT_Token>
Content-Type: application/json
```

Importante: Todos los endpoints (excepto autenticación) requieren el token JWT en el header de autorización.

Endpoints Principales

Gestión de Bovinos

Obtener bovino específico

```
GET /api/v1/bovines/{id}
```

Recupera información detallada de un bovino específico por su ID.

Registrar nuevo bovino

```
POST /api/v1/bovines
```

Ejemplo de respuesta exitosa:

```
{  
    "id": 12,  
    "name": "Toro Brangus",  
    "birthDate": "2023-07-20",  
    "weight": 380,  
    "breed": "Brangus",  
    "gender": "Macho",  
    "stableId": 3,  
    "healthStatus": "Saludable",  
    "createdAt": "2025-08-10T18:23:01Z",  
    "updatedAt": "2025-08-10T18:23:01Z"  
}
```

Gestión de Establos

Listar todos los establos

```
GET /api/v1/stables
```

Crear nuevo establo

```
POST /api/v1/stables
```

Ejemplo de respuesta:

```
{  
    "id": 3,  
    "name": "Establo Central",  
    "capacity": 50,  
    "currentOccupancy": 32,  
    "location": "Lima, Perú",  
    "status": "Activo",  
    "createdAt": "2025-08-10T18:23:01Z"  
}
```

Gestión de Vacunas

Obtener información de vacuna

```
GET /api/v1/vaccines/{id}
```

Registrar nueva vacuna

```
POST /api/v1/vaccines
```

Ejemplo de respuesta:

```
{  
    "id": 8,  
    "name": "Vacuna Aftosa",  
    "type": "Sanitaria",  
    "description": "Vacuna contra fiebre aftosa",  
    "applicationDate": "2025-08-15",  
    "expirationDate": "2026-08-15",  
    "veterinarian": "Dr. García",  
    "bovineId": 12,  
    "status": "Aplicada"  
}
```

Gestión de Campañas

Consultar detalles de campaña

```
GET /api/v1/campaigns/{id}
```

Registrar nueva campaña

```
POST /api/v1/campaigns
```

Ejemplo de respuesta:

```
{  
    "id": 5,  
    "name": "Campaña Antiparasitaria",  
    "description": "Campaña de desparasitación general",  
    "startDate": "2025-09-01",  
    "endDate": "2025-09-15",  
    "status": "Activa",  
    "responsibleVet": "Dr. Rodríguez",  
    "targetAnimals": 150,
```

```

    "completedAnimals": 45,
    "progress": 30
}

```

Códigos de Estado HTTP

La API implementa un manejo estandarizado de códigos de estado HTTP para facilitar la depuración y el desarrollo:

Código	Descripción	Escenario
200 OK	Operación exitosa	Consultas y actualizaciones correctas
201 Created	Recurso creado exitosamente	Registro de nuevos bovinos, establos, etc.
400 Bad Request	Parámetros incorrectos o datos inválidos	Validaciones fallidas
401 Unauthorized	Token JWT inválido o no proporcionado	Problemas de autenticación
403 Forbidden	Permisos insuficientes	Restricciones de acceso por rol
404 Not Found	El recurso solicitado no existe	Bovino, establo o campaña no encontrado
409 Conflict	Conflictivo de recursos	Nombres duplicados, restricciones de negocio
500 Internal Server Error	Error inesperado en el servidor	Errores no controlados

Validación y Testing

Estrategia de Pruebas

La API ha sido sometida a pruebas exhaustivas utilizando múltiples enfoques:

Pruebas automatizadas en Swagger:

- Validación de sintaxis de requests/responses
- Verificación de códigos de estado HTTP
- Testing de autenticación JWT

Pruebas manuales en Postman:

- Escenarios de uso real del dominio ganadero
- Validación de reglas de negocio específicas
- Testing de restricciones (ej: capacidad máxima de establos)
- Pruebas de rendimiento y carga

Casos de Prueba Validados

- **Autenticación:** Login/logout, expiración de tokens, roles de usuario
- **Gestión de bovinos:** Registro, consulta, actualización, eliminación
- **Campañas de vacunación:** Creación, seguimiento, finalización
- **Restricciones de negocio:** Capacidad de establos, fechas de vacunación

Seguridad y Autenticación

Implementación JWT

La seguridad de VacApp se fundamenta en el uso de **JSON Web Tokens (JWT)** con las siguientes características:

Configuración de seguridad:

- **Algoritmo de encriptación:** HS256
- **Tiempo de expiración:** 24 horas
- **Refresh token:** Implementado para renovación automática
- **Roles y permisos:** Sistema granular de autorización

Ejemplo de solicitud autenticada:

```
GET /api/v1/bovines/12
Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6I
kpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36P0k6yJV_a
dQssw5c
Content-Type: application/json
```

Medidas de Seguridad Adicionales

- **Rate limiting:** Prevención de ataques de fuerza bruta
- **CORS configurado:** Restricción de orígenes permitidos
- **Validación de entrada:** Sanitización de todos los parámetros
- **Logging de auditoría:** Trazabilidad completa de operaciones

5.2.8. Team Collaboration Insights

Durante el desarrollo del Sprint 1, el equipo colaboró activamente en el repositorio del Informe, utilizando herramientas como GitHub, Trello y Discord para coordinar tareas, compartir avances y resolver dudas de forma continua.

Se realizaron reuniones semanales para planificación y revisión, así como sesiones diarias breves (dailys) para mantener sincronizado el trabajo entre los integrantes.

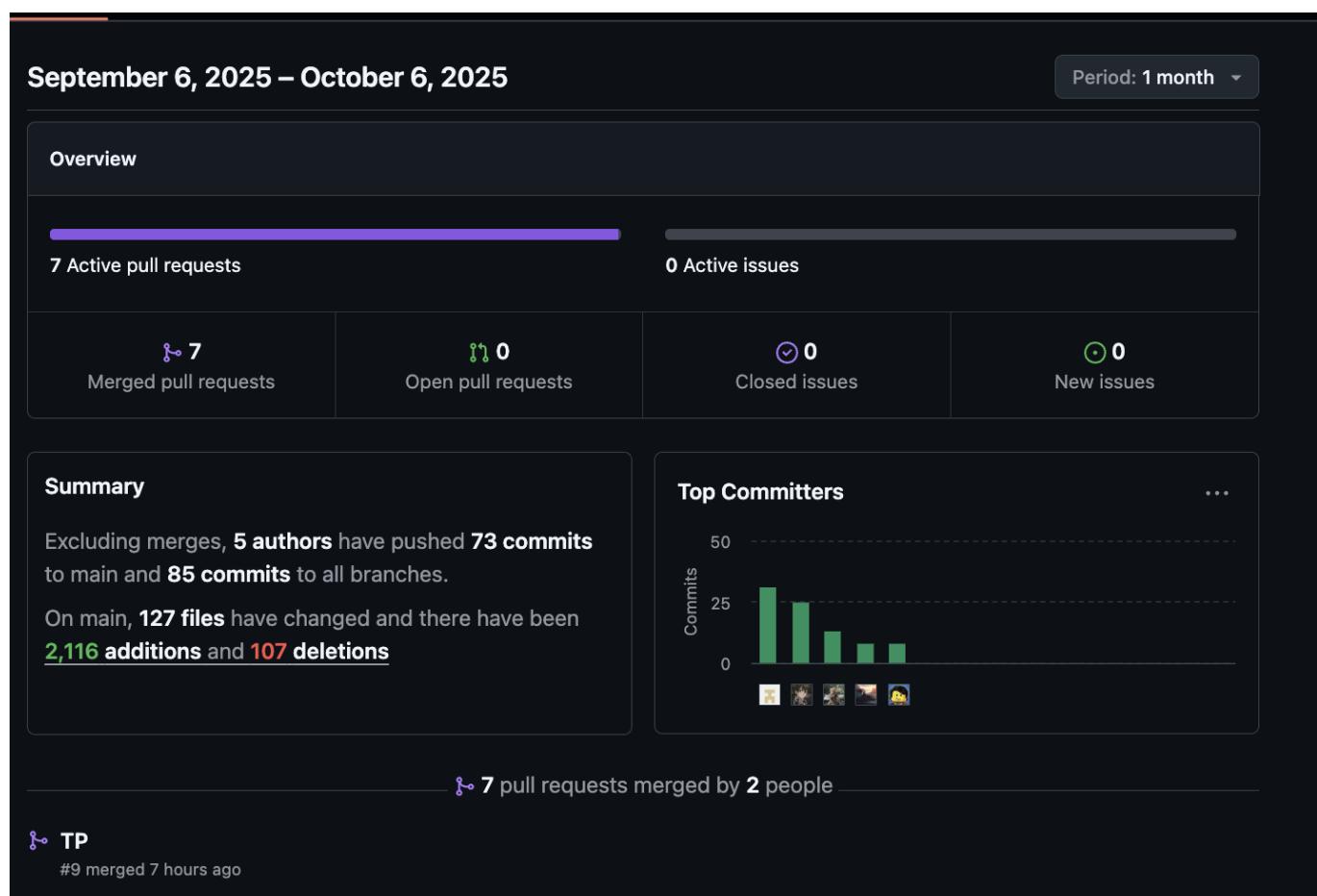
A continuación, gracias a la sección de Insights de GitHub, se presentan gráficas que muestran el nivel de participación de cada miembro del equipo en el repositorio del Informe.

Tabla de identificación del equipo

Username (GitHub)	Nombre completo	Código de estudiante
@Libeman10	Rodrigo Liberato Saldaña	U202215623
@IlanMQ	Ian Macavilca Quispe	U202121325
@Kmykh	Maycol Rojas Velasquez	U202219984
@Stephanoescu	Stephano Espinoza Cueva	U202218590
@Kmykh	Jeremy Paucar Meneses	U201919449

Analíticos de GitHub

Informe



Las gráficas demuestran que todos los integrantes realizaron contribuciones significativas en el repositorio del Informe, reflejando una distribución equilibrada de tareas y un compromiso constante con el avance del proyecto.

5.3. Video About-the-Product

Con el objetivo de complementar la documentación y ofrecer una visión más clara del alcance de VacApp, se ha elaborado un video de presentación denominado About The Product.

Este recurso audiovisual explica de manera concisa las funcionalidades principales de la aplicación, su propuesta de valor y cómo contribuye a optimizar la gestión del ganado bovino. Asimismo, permite a los interesados obtener una experiencia más cercana al producto, facilitando la comprensión de sus características y beneficios en un formato dinámico y accesible.

Simplifica la gestión de tu ganado con



VacApp

The image shows three smartphones side-by-side, each displaying a different screen of the VacApp mobile application. The background is a solid teal color.

- Smartphone 1 (Left): Gestión de Campañas**
 - Header: Gestión de Campañas
 - Buttons: Crear Nueva Campaña (green), Crear Nuevo Establecimiento (blue)
 - Section: Campañas (with a green icon and text: Estráño dásprean)
 - Text: Aquí queremos Cuidar a los animales que están en el ganado Serjío
 - Icons: Precio (00.005) and Establecimiento (Estable de C-Veterinarios)
 - Text: Duración de la campaña: 30/7/2025 - 13/3/2025 (15 días)
 - Text: Def 30/7/2025 al 12/3/2025
 - Text: BS (radio button)
- Smartphone 2 (Middle): Consieión de Campañas**
 - Header: Consieión de Campañas
 - Button: Crear Nueva Campaña (green)
 - Section: Campaña para Cuidado Veterinarios (Active)
 - Text: Aquí queremos cuidar a los animales que están en el ganado Serjío
 - Icons: Precio (00.005) and Establecimiento (Estable de C-Veterinarios)
 - Text: Duración de la campaña: 30/7/2025 - 13/3/2025 (15 días)
 - Text: Def 30/7/2025 al 12/3/2025
- Smartphone 3 (Right): Gestión de Establecimientos**
 - Header: Gestión de Establecimientos
 - Section: Establecimiento (with a green icon and text: Estable de C-Veterinarios)
 - Text: Duración de la campaña: 30/7/2025 - 13/3/2025 (15 días)
 - Text: Def 30/7/2025 al 12/3/2025
 - Text: BS (radio button)

Descárgala ahora

El video se encuentra disponible en el siguiente enlace:

<https://www.youtube.com/watch?v=JmOW2IkXjeI>

Capítulo VI: Product Verification & Validation

6.1. Testing Suites & Validation

En el desarrollo de **VacApp**, las pruebas de software constituyen un componente esencial para garantizar la **calidad, estabilidad y confiabilidad** del sistema. Dado que la aplicación administra información crítica relacionada con la **salud, producción y trazabilidad del ganado**, la validación exhaustiva de los módulos es fundamental para asegurar la precisión de los datos y la continuidad operativa del servicio.

El enfoque de validación de VacApp se basa en la **automatización de pruebas unitarias, de integración y de comportamiento**, verificando el correcto funcionamiento de los servicios principales tanto en la versión web como móvil. Las pruebas se ejecutan dentro de un entorno controlado que replica la infraestructura de despliegue (Dev/Test), lo cual permite detectar errores antes de llegar a producción.

Asimismo, VacApp adopta buenas prácticas de **Continuous Integration (CI) y Behavior-Driven Development (BDD)** para mantener la integridad del modelo de dominio y garantizar la calidad del software a lo largo de su ciclo de vida. Esto asegura que cada iteración del producto preserve la consistencia funcional y la confiabilidad de la información ganadera.

6.1.1. Core Entities Unit Tests

Las *Core Entities Unit Tests* son un componente esencial dentro del proceso de verificación de la plataforma **VacApp**, ya que permiten validar el comportamiento y la integridad lógica de las entidades centrales en cada *bounded context* del dominio. Estas pruebas se desarrollaron utilizando **C#**, el framework de pruebas **xUnit**, y herramientas complementarias como **Moq** para simular dependencias cuando sea necesario.

Identity & Access Management (IAM) Unit Tests

Clase: AdminTests

Las pruebas de la entidad **Admin** validan que el proceso de creación y actualización de administradores cumpla con las políticas internas de VacApp, principalmente el uso obligatorio del dominio corporativo y la validación del formato de correo.

1. Constructor_con_CreateAdminCommand_valido_asigna_propiedades

- **Objetivo:** verificar que, al crear un administrador con un correo válido del dominio **@vacapp.com**, las propiedades se asignen correctamente.
- **Validaciones:**
 - El campo **Email** coincide con el ingresado.
 - **EmailConfirmed** se marca como **true**.
- **Resultado esperado:** el administrador se crea correctamente y el correo queda confirmado de manera automática.

The screenshot shows the Visual Studio IDE interface. The left pane displays the Solution Explorer with project files like Application, Domain, Model, Infrastructure, and Interfaces. The right pane shows two code files: AdminTests.cs and UserTests.cs. AdminTests.cs contains C# code for testing an Admin class, including a constructor test. The Test Explorer pane at the bottom shows a successful run of 15 tests under VacApp.Tests, with one specific test highlighted: 'Constructor_con_CreateAdminCommand_valido_asigna_propiedades' which passed.

```

public class AdminTests
{
    private readonly ITestOutputHelper _output;
    public AdminTests(ITestOutputHelper output) => _output = output;

    [Fact]
    public void Constructor_con_CreateAdminCommand_valido_asigna_propiedades()
    {
        // Arrange
        var cmd = new CreateAdminCommand(Email: "admin@vacapp.com");

        // Act
        var sut = new Admin(cmd);

        // Assert
        Assert.Equal("admin@vacapp.com", sut.Email);
        Assert.True(sut.EmailConfirmed);
        _output.WriteLine("AAA -> Assert: Admin creado correctamente con email @vacapp.com");
    }
}

```

2. Constructor_con_email_no_vacapp_lanza_ArgumentException

- Objetivo:** asegurar que ningún administrador pueda ser creado con un correo ajeno al dominio institucional.
- Escenario:** se intenta crear un Admin con el email `admin@gmail.com`.
- Validación:**
 - Se lanza una excepción `ArgumentException`.
 - El mensaje devuelto comienza con "`Admin email must end with @vacapp.com`".
- Resultado esperado:** se evita la creación de administradores fuera del dominio corporativo.

The screenshot shows the Visual Studio Code interface with the following details:

- Solution Explorer:** Shows the project structure for "VacApp-Bovinova-Platform".
- Code Editor:** Displays the file "AdminTests.cs" containing C# unit test code for an "Admin" entity.
- Test Explorer:** Shows the results of the tests run in "VacApp.Tests". One test, "Constructor_con_email_no_vacapp_lanza_ArgumentException", is highlighted and failed, showing the message "AAA -> Assert: Excepción por no terminar en @vacapp.com".
- Status Bar:** Shows the file path "VacApp-Bovinova-Platform > VacApp.Tests > IAMTests > AdminTests.cs", encoding "UTF-8", and other settings.

```

public class AdminTests
{
    [Fact]
    public void Constructor_con_email_no_vacapp_lanza_ArgumentException()
    {
        // Arrange
        var cmd = new CreateAdminCommand(Email: "admin@gmail.com");

        // Act
        var ex = Assert.Throws<ArgumentException>(() => new Admin(cmd));

        // Assert
        Assert.StartsWith("Admin email must end with @vacapp.com", ex.Message);
        _output.WriteLine("AAA -> Assert: Excepción por no terminar en @vacapp.com");
    }

    [Fact]
    public void Update_con_email_invalido_lanza_ArgumentException()
    {
        // Arrange
    }
}

```

3. Update_con_email_invalido_lanza_ArgumentException

- **Objetivo:** garantizar que las actualizaciones de datos de un administrador validen el formato del correo.
- **Escenario:** se actualiza un **Admin** existente con un correo con formato incorrecto (**bad@vacapp.com**).
- **Validación:**
 - Se lanza una excepción **ArgumentException**.
 - El mensaje comienza con "**Invalid email format.**".
- **Resultado esperado:** la entidad mantiene su integridad y evita registros inválidos durante actualizaciones.

The screenshot shows the Visual Studio Code interface with the following details:

- Solution Explorer:** Shows the project structure for "VacApp-Bovinova-Platform".
- Code Editor:** Displays the `AdminTests.cs` file containing unit tests for the `Admin` class.
- Test Explorer:** Shows the test results for `VacApp.Tests`, specifically for `IAMTests`. A test named `ValidateLogin_siempre_devuelve_true` is highlighted as successful.
- Status Bar:** Shows the path "VacApp-Bovinova-Platform > VacApp.Tests > IAMTests > AdminTests.cs > AdminTests > ValidateLogin_siempre_devuelve_true Success", along with the date and time (46:133, LF, UTF-8).

4. ValidateLogin_siempre_devuelve_true

- Objetivo:** comprobar la respuesta base del método de autenticación.
- Escenario:** se ejecuta el método `ValidateLogin` con cualquier contraseña.
- Validación:**
 - El método retorna `true`.
- Resultado esperado:** confirmación del flujo base del login, que podrá evolucionar con futuras integraciones de seguridad.

The screenshot shows the Visual Studio Code interface with the following details:

- Solution Explorer:** Shows the project structure for "VacApp-Bovinova-Platform".
- Code Editor:** Displays the `AdminTests.cs` file containing the `ValidateLogin_siempre_devuelve_true` test.
- Test Explorer:** Shows the test results for `VacApp.Tests`, specifically for `IAMTests`. A test named `ValidateLogin_siempre_devuelve_true` is highlighted as successful.
- Status Bar:** Shows the path "VacApp-Bovinova-Platform > VacApp.Tests > IAMTests > AdminTests.cs > AdminTests > ValidateLogin_siempre_devuelve_true Success", along with the date and time (57:21, LF, UTF-8).

Clase: UserTests

La entidad **User** representa a los usuarios generales de la plataforma (productores, veterinarios o gestores).

Las pruebas se enfocan en validar la creación, actualización y verificación de los datos básicos del perfil.

1. Constructor_por_defecto_inicializa_campos_vacios_y_email_no_confirmado

- Objetivo:** validar que un usuario nuevo se inicializa con valores vacíos y el email sin confirmar.
- Validaciones:**
 - Username, Password y Email** son `string.Empty`.
 - EmailConfirmed** es `false`.
- Resultado esperado:** el objeto inicia en un estado neutro, listo para asignar datos válidos posteriormente.

The screenshot shows the Visual Studio IDE interface. The left sidebar displays the solution structure, including projects like VoiceCommand, .env, appsettings.json, appsettings.Development.json, Program.cs, VacApp-Bovinova-Platform.http, VacApp.Tests, CampaignTests, IAMTests, AdminTests.cs, and UserTests.cs. The right side shows the code editor for UserTests.cs and the Test Explorer. The UserTests.cs file contains C# code for unit tests. The Test Explorer shows a tree of test results, with one test under IAMTests highlighted as successful: "Constructor_por_defecto_inicializa_campos_vacios_y_email_no_confirmado [4 ms]". The status bar at the bottom indicates the file is 1:1, LF, UTF-8, and has 4 spaces.

```

public class UserTests
{
    [Fact]
    public void Constructor_por_defecto_inicializa_campos_vacios_y_email_no_confirmado()
    {
        // Arrange
        var sut = new User();

        // Act
        Assert.Equal(string.Empty, sut.Username);
        Assert.Equal(string.Empty, sut.Password);
        Assert.Equal(string.Empty, sut.Email);
        Assert.False(sut.EmailConfirmed);

        _output.WriteLine("AAA -> Assert: User por defecto con strings vacios y EmailConfirmed=false");
    }

    [Fact]
    public void Constructor_con_SignUpCommand_valido_asigna_propiedades()
    {
    }
}

```

2. Constructor_con_SignUpCommand_valido_asigna_propiedades

- Objetivo:** comprobar que los datos se asignan correctamente al crear un usuario mediante el comando `SignUpCommand`.
- Validaciones:**
 - Se asignan correctamente **Username, Password** y **Email**.
- Resultado esperado:** el usuario se crea exitosamente con la información registrada en el formulario de inscripción.

The screenshot shows the Visual Studio Code interface with the following details:

- Solution Explorer:** Shows the project structure with files like .env, appsettings.json, Program.cs, and various Test classes (e.g., AdminTests.cs, UserTests.cs).
- Code Editor:** Displays the `UserTests.cs` file containing C# code for testing the `SignUpCommand` constructor.
- Output:** Shows the test results in the bottom right corner, indicating a success message for the test `Constructor_con_SignUpCommand_valido_asigna_propiedades()`.
- Terminal:** Shows the command `dotnet test` being run.

```

public class UserTests
{
    [Fact]
    public void Constructor_con_SignUpCommand_valido_asigna_propiedades()
    {
        // Arrange
        var cmd = new SignUpCommand(
            Username: "jdoe",
            Password: "Secret123!",
            Email: "jdoe@example.com"
        );

        // Act
        var sut = new User(cmd);

        // Assert
        Assert.Equal("jdoe", sut.Username);
        Assert.Equal("Secret123!", sut.Password);
        Assert.Equal("jdoe@example.com", sut.Email);
        _output.WriteLine("AAA -> Assert: User creado correctamente con SignUpCommand válido");
    }
}

```

3. Constructor_con_email_invalido_lanza_ArgumentException

- Objetivo:** validar que el formato del correo electrónico sea correcto durante el registro.
- Escenario:** se intenta crear un usuario con un email sin formato (**sin-formato**).
- Validación:**
 - Se lanza una excepción **ArgumentException**.
 - El mensaje devuelto comienza con "**Invalid email format.**".
- Resultado esperado:** se previene el almacenamiento de correos no válidos en la base de datos.

The screenshot shows the Visual Studio Code interface with the following details:

- Solution Explorer:** Shows the project structure with files like .env, appsettings.json, Program.cs, and various Test classes (e.g., AdminTests.cs, UserTests.cs).
- Code Editor:** Displays the `UserTests.cs` file containing C# code for testing the `SignUpCommand` constructor with an invalid email.
- Output:** Shows the test results in the bottom right corner, indicating a success message for the test `Constructor_con_email_invalido_lanza_ArgumentException()`.
- Terminal:** Shows the command `dotnet test` being run.

```

[Fact]
public void Constructor_con_email_invalido_lanza_ArgumentException()
{
    // Arrange
    var cmd = new SignUpCommand(
        Username: "jdoe",
        Password: "Secret123!",
        Email: "sin-formato"
    );

    // Act
    var ex = Assert.Throws<ArgumentException>(() => new User(cmd));

    // Assert
    Assert.StartsWith("Invalid email format.", ex.Message);
    _output.WriteLine("AAA -> Assert: Excepción por email inválido en constructor de User");
}

```

4. Update_con_email_invalido_lanza_ArgumentException

- **Objetivo:** asegurar que la actualización de un usuario respete las mismas reglas de validación del registro inicial.
- **Escenario:** se actualiza un usuario con un correo inválido ([mal-email](#)).
- **Validación:**
 - Se lanza una excepción [ArgumentException](#).
 - El mensaje devuelto comienza con "[Invalid email format.](#)".
- **Resultado esperado:** las actualizaciones no permiten romper las reglas de integridad del dominio.

The screenshot shows the Visual Studio IDE interface. The top navigation bar includes 'VA VacApp-Bovinova-Platform' and 'main'. The left sidebar shows the project structure under 'Solution'. The main code editor window displays 'UserTests.cs' with the following code:

```

    public class UserTests
    {
        [Fact]
        public void Update_con_email_invalido_lanza_ArgumentException()
        {
            // Arrange
            var sut = new User(new SignUpCommand(
                Username: "jdoe",
                Password: "Secret123!",
                Email: "jdoe@example.com"
            ));
            var update = new UpdateUserCommand(
                Username: "jdoe2",
                Email: "mal-email"
            );

            // Act
            var ex = Assert.Throws<ArgumentException>(() => sut.Update(update));

            // Assert
        }
    }

```

The 'Test Explorer' tab is selected at the bottom, showing a list of tests. One test, 'Update_con_email_invalido_lanza_ArgumentException', is highlighted and failed, with the message: 'AAA -> Assert: Excepción por email inválido en Update de User'.

Ranch Management

El contexto **Ranch Management** de **VacApp** abarca la administración operativa del rancho, incluyendo la gestión de bovinos, establos y vacunas.

Estas pruebas unitarias se desarrollaron en **C# con xUnit**, siguiendo la estructura **AAA (Arrange / Act / Assert)**, con el objetivo de validar las reglas de negocio que garantizan la integridad, coherencia y trazabilidad de los registros ganaderos.

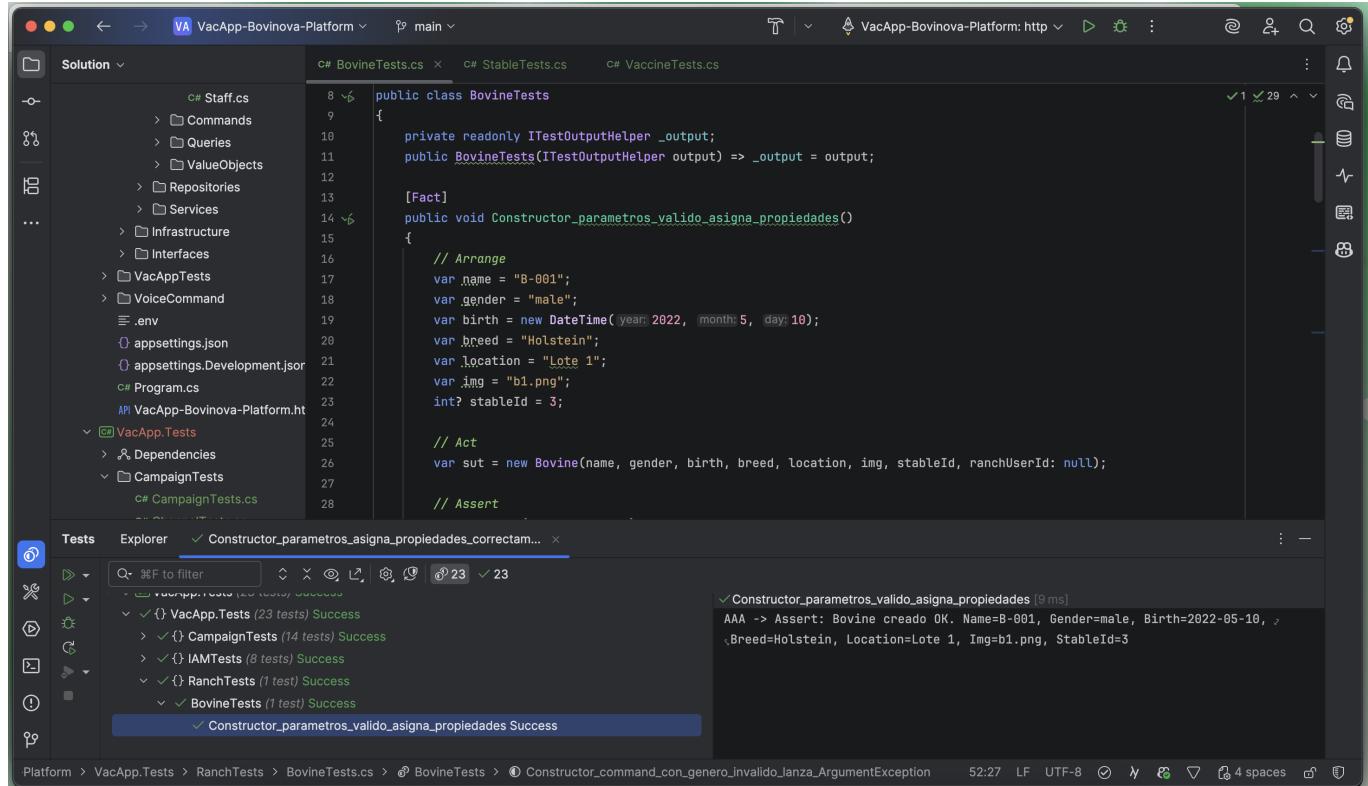
Clase: [BovineTests](#)

La entidad **Bovine** representa al animal dentro del sistema, incluyendo su información sanitaria, ubicación y características biológicas.

Las pruebas unitarias aseguran la correcta construcción y actualización de sus propiedades, así como el cumplimiento de las reglas básicas de validación.

1. [Constructor_parametros_valido_asigna_propiedades](#)

- **Objetivo:** validar que un bovino creado manualmente con parámetros válidos asigne correctamente sus propiedades.
- **Validaciones:**
 - Coincidencia exacta de nombre, género, raza, fecha de nacimiento, ubicación e imagen.
 - Confirmación del **StableId** asignado.
- **Resultado esperado:** el objeto se inicializa correctamente y refleja los valores proporcionados.



```

public class BovineTests
{
    private readonly ITestOutputHelper _output;
    public BovineTests(ITestOutputHelper output) => _output = output;

    [Fact]
    public void Constructor_parametros_valido_asigna_propiedades()
    {
        // Arrange
        var name = "B-001";
        var gender = "male";
        var birth = new DateTime( year: 2022, month: 5, day: 10 );
        var breed = "Holstein";
        var location = "Lote 1";
        var img = "b1.png";
        int? stableId = 3;

        // Act
        var sut = new Bovine(name, gender, birth, breed, location, img, stableId, ranchUserId: null);

        // Assert
        AAA -> Assert: Bovine creado OK. Name=B-001, Gender=Male, Birth=2022-05-10, Breed=Holstein, Location=Lote 1, Img=b1.png, StableId=3
    }
}

```

The screenshot shows the Visual Studio IDE interface. The top navigation bar says "VacApp-Bovinova-Platform" and "main". The left sidebar shows the project structure under "Solution". The main editor window displays the C# code for "BovineTests.cs". The bottom pane shows the "Tests" tab of the "Explorer" tool, listing various test results, with one specific test highlighted: "Constructor_parametros_valido_asigna_propiedades Success". The status bar at the bottom indicates the file is "52:27 LF UTF-8" and has "4 spaces".

2. [Constructor_command_con_genero_invalido_lanza_ArgumentException](#)

- **Objetivo:** garantizar que solo se acepten géneros válidos (**male** o **female**).
- **Escenario:** creación de un bovino con **Gender = "unknown"**.
- **Validación:**
 - Se lanza una excepción **ArgumentException**.
 - El mensaje indica "**Gender must be either 'male' or 'female'**".
- **Resultado esperado:** se evita la creación de registros con valores inconsistentes en el dominio biológico.

```

public class BovineTests
{
    [Fact]
    public void Constructor_command_con_genero_invalido_lanza_ArgumentException()
    {
        // Arrange
        var cmd = new CreateBovineCommand(
            Name: "B-ERR",
            Gender: "unknown",
            BirthDate: new DateTime(2023, 1, 1),
            Breed: "Jersey",
            Location: "Lote X",
            BovineImg: "x.png",
            StableId: 1,
            RanchUserId: new RanchUserId(1),
            FileData: null
        );
        // Act
        var ex = Assert.Throws<ArgumentException>(() => new Bovine(cmd));
        // Assert
    }
}

```

3. Constructor_command_con_userid_nulo_lanza_ArgumentException

- Objetivo:** comprobar que todo bovino esté asociado a un usuario del rancho responsable del registro.
- Escenario:** se omite el **RanchUserId** en la creación.
- Validación:**
 - Se lanza una excepción **ArgumentException**.
 - El mensaje indica "**User Id must be set by the system**".
- Resultado esperado:** se evita la creación de registros huérfanos sin trazabilidad de usuario.

The screenshot shows the Visual Studio IDE interface. The code editor displays the `BovineTests.cs` file, which contains C# code for testing bovine objects. The `Constructor_command_con_userid_nulo_lanza_ArgumentException()` method is highlighted. The Test Explorer window below shows a tree of tests under the `RanchTests` category, with all tests marked as `Success`. A tooltip for one of the tests provides a brief description of the validation.

```

public class BovineTests
{
    [Fact]
    public void Constructor_command_con_userid_nulo_lanza_ArgumentException()
    {
        // Arrange
        var cmd = new CreateBovineCommand(
            Name: "B-NULL",
            Gender: "female",
            BirthDate: new DateTime(2023, month: 2, day: 2),
            Breed: "Angus",
            Location: "Lote 2",
            BovineImg: "b2.png",
            StableId: 2,
            RanchUserId: null,
            FileData: null
        );
        // Act
        var ex = Assert.Throws<ArgumentException>(() => new Bovine(cmd));
        // Assert
    }
}

```

4. Update_con_datos_validos_actualiza_propiedades_y_conserva_imagen

- Objetivo:** asegurar que las actualizaciones modifiquen correctamente los datos básicos del bovino, sin alterar su imagen previa.
- Validaciones:**
 - Los campos `Name`, `Gender`, `BirthDate`, `Breed`, `Location` y `StableId` cambian correctamente.
 - La propiedad `BovineImg` se mantiene inalterada.
- Resultado esperado:** el bovino actualiza sus propiedades sin perder información visual o histórica.

The screenshot shows the Visual Studio IDE interface. The code editor displays the `BovineTests.cs` file, which contains C# unit tests for updating bovine data. The `Test Explorer` window below shows a successful run with 26 tests, including four tests for `RanchTests` and four for `BovineTests`. The output window shows the results of the `Update_con_datos_validos_actualiza_propiedades_y_conserva_imagen` test.

```

public class BovineTests
{
    [Fact]
    public void Update_con_datos_validos_actualiza_propiedades_y_conserva_imagen()
    {
        // Arrange
        var sut = new Bovine("B-Init", "male", new DateTime(2022, 1, 1), "Criollo", "Lote");
        var update = new UpdateBovineCommand(
            Id: sut.Id,
            Name: "B-Updated",
            Gender: "female",
            BirthDate: new DateTime(2021, 12, 31),
            Breed: "Brahman",
            Location: "Lote B",
            StableId: 6
        );

        // Act
        sut.Update(update);

        // Assert
        Assert.Equal("B-Updated", sut.Name);
    }
}

```

Clase: **StableTests**

La entidad **Stable** administra los establos o corrales dentro del rancho, manteniendo control de capacidad y asignación de bovinos.

Las pruebas unitarias validan las restricciones de dominio y las actualizaciones permitidas.

1. [Constructor_con_limit_in valido_lanza_ArgumentException](#)

- **Objetivo:** evitar la creación de establos con límites no válidos.
- **Escenario:** se intenta crear un establo con `Limit = 0`.
- **Validación:**
 - Se lanza una excepción `ArgumentException`.
 - Mensaje: "`Limit must be greater than 0`".
- **Resultado esperado:** el sistema impide crear establos sin capacidad operativa.

The screenshot shows the Visual Studio Code interface with the following details:

- Solution Explorer:** Shows the project structure for "VacApp-Bovinova-Platform".
- Code Editor:** Displays the C# code for `StableTests.cs`. The code includes a constructor that throws an `ArgumentException` if the limit is less than or equal to zero. A specific test case is highlighted: `Constructor_con_limit_invalido_lanza_ArgumentException`.
- Test Explorer:** Shows the test results for the `RanchTests` category. It lists several successful tests, including the one just edited.
- Status Bar:** Shows the file path "VacApp-Bovinova-Platform > VacApp.Tests > RanchTests > C# StableTests.cs" and other standard status bar information.

2. Constructor_con_nombre_vacio_lanza_ArgumentException

- **Objetivo:** garantizar que el nombre del establo no sea vacío.
- **Validación:**
 - Lanza `ArgumentException` con el mensaje "`Name must not be empty`".
- **Resultado esperado:** se preserva la consistencia de identificación de cada establo.

The screenshot shows the Visual Studio Code interface with the following details:

- Solution Explorer:** Shows the project structure for "VacApp-Bovinova-Platform".
- Code Editor:** Displays the C# code for `StableTests.cs`. The code includes a constructor that throws an `ArgumentException` if the name is empty. Another constructor is also partially implemented.
- Test Explorer:** Shows the test results for the `RanchTests` category. It lists several successful tests, including the one just edited.
- Status Bar:** Shows the file path "VacApp-Bovinova-Platform > VacApp.Tests > RanchTests > C# StableTests.cs" and other standard status bar information.

3. Constructor_con_userid_nulo_lanza_ArgumentException

- **Objetivo:** asegurar que todo establo esté vinculado a un usuario registrado del rancho.
- **Validación:**
 - Lanza **ArgumentException** con el mensaje "**RanchUserId must be set by the system**".
- **Resultado esperado:** ningún establo se crea sin responsable asignado.

The screenshot shows the Visual Studio IDE interface. The left pane displays the solution structure for 'VacApp-Bovinova-Platform' with several test projects like 'VacApp.Tests' and 'RanchTests'. The right pane shows the code for 'StableTests.cs' and the 'Test Explorer' results. The code in 'StableTests.cs' contains two test methods: 'Constructor_con_userid_nulo_lanza_ArgumentException()' and 'Constructor_valido_asigna_propiedades()'. The 'Test Explorer' window shows 29 successful tests under 'RanchTests', including three under 'StableTests': 'Constructor_con_limit_invalido_lanza_ArgumentException', 'Constructor_con_nombre_vacio_lanza_ArgumentException', and 'Constructor_con_userid_nulo_lanza_ArgumentException'. The status bar at the bottom indicates '1:1 LF UTF-8' and '4 spaces'.

```

public class StableTests
{
    [Fact]
    public void Constructor_con_userid_nulo_lanza_ArgumentException()
    {
        // Arrange
        var cmd = new CreateStableCommand(Limit: 10, Name: "Establo A", RanchUserId: null);

        // Act
        var ex = Assert.Throws<ArgumentException>(() => new Stable(cmd));

        // Assert
        Assert.Equal("RanchUserId must be set by the system", ex.Message);
        _output.WriteLine("AAA -> Assert: Excepción esperada por RanchUserId nulo al crear Stable.");
    }

    [Fact]
    public void Constructor_valido_asigna_propiedades()
    {
        // Arrange
    }
}

```

4. [Constructor_valido_asigna_propiedades](#)

- **Objetivo:** comprobar la asignación correcta de nombre y capacidad cuando se crean valores válidos.
- **Resultado esperado:** las propiedades **Name** y **Limit** coinciden con el comando recibido.

The screenshot shows the Visual Studio Code interface. The left sidebar displays the project structure for 'VacApp-Bovinova-Platform'. The main area shows the 'StableTests.cs' file with C# code. The code includes several test methods using the [Fact] attribute. The bottom right pane shows the 'Tests' results, indicating successful runs for 'BovineTests' and 'StableTests'.

```

public class StableTests
{
    public void Constructor_con_userid_nulo_lanza_ArgumentException()
    {
        // Arrange
        var cmd = new CreateStableCommand(Limit: 50, Name: "Establo A", RanchUserId: new RanchUserId(1));

        // Act
        var sut = new Stable(cmd);

        // Assert
        Assert.Equal(50, sut.Limit);
        Assert.Equal("Establo A", sut.Name);
        _output.WriteLine($"AAA -> Assert: Stable creado correctamente. Name={sut.Name}, Limit={sut.Limit}");
    }

    [Fact]
    public void Update_con_datos_invalidos_lanza_ArgumentException()
    {
    }
}

```

5. Update_con_datos_invalidos_lanza_ArgumentException

- Objetivo:** evitar actualizaciones con límites negativos o inválidos.
- Validación:**
 - Se lanza **ArgumentException** con el mensaje "**Limit must be greater than 0**".
- Resultado esperado:** la integridad del registro se mantiene estable.

The screenshot shows the Visual Studio Code interface. The left sidebar displays the project structure for 'VacApp-Bovinova-Platform'. The main area shows the 'StableTests.cs' file with C# code. The code includes several test methods using the [Fact] attribute. The bottom right pane shows the 'Tests' results, indicating successful runs for 'BovineTests' and 'StableTests'.

```

public class StableTests
{
    public void Constructor_parametros_asigna_propiedades_correctamente()
    {
    }

    [Fact]
    public void Update_con_datos_invalidos_lanza_ArgumentException()
    {
        // Arrange
        var sut = new Stable(new CreateStableCommand(Limit: 10, Name: "Establo B", RanchUserId: new RanchUserId(1)));
        var update = new UpdateStableCommand(Limit: -1, Name: "Establo B2", Id: sut.Id);

        // Act
        var ex = Assert.Throws<ArgumentException>(() => sut.Update(update));

        // Assert
        Assert.Equal("Limit must be greater than 0", ex.Message);
        _output.WriteLine("AAA -> Assert: Excepción esperada en Update por límite no válido (<= 0).");
    }

    [Fact]
    public void Update_con_datos_validos_actualiza_propiedades()
    {
    }
}

```

6. Update_con_datos_validos_actualiza_propiedades

- Objetivo:** validar que el método `Update()` actualiza correctamente los campos modificables (`Name` y `Limit`).
- Resultado esperado:** los nuevos valores se reflejan correctamente en el estado final del objeto.

The screenshot shows the Visual Studio IDE interface. The code editor displays the `StableTests.cs` file, which contains a test method `Update_con_datos_validos_actualiza_propiedades`. The test usesArrange, Act, and Assert patterns to verify that an update command with valid parameters correctly updates the stable's name and limit. The test passes, as indicated by the green checkmark in the status bar.

The Test Explorer window below shows several other test cases for the `RanchTests` class, all of which have passed (indicated by green checkmarks). The current test being run is `Update_con_datos_validos_actualiza_propiedades`, which completed successfully in 7 ms.

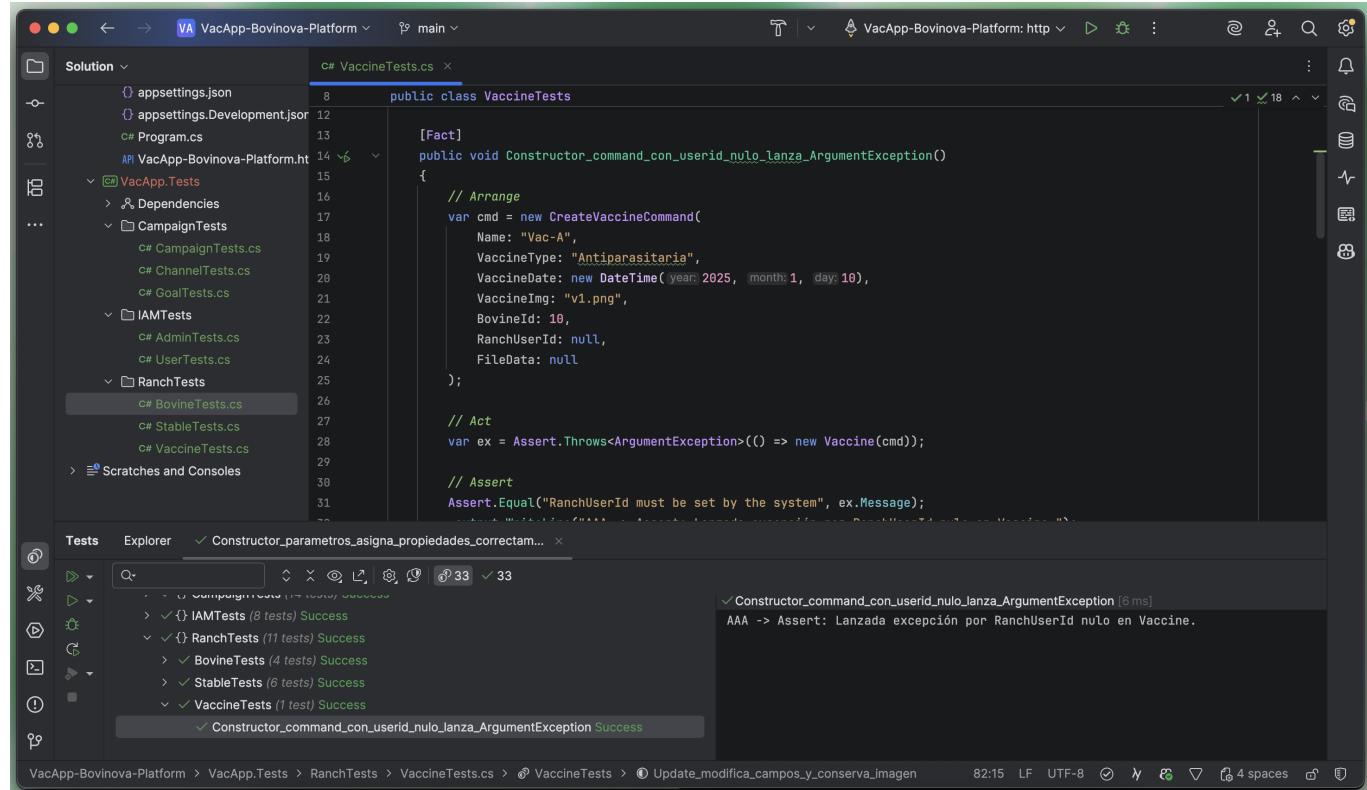
Clase: **VaccineTests**

La entidad **Vaccine** gestiona los registros de vacunación de cada bovino, manteniendo información sobre el tipo, fecha, imagen y usuario responsable.

Estas pruebas garantizan la integridad de los registros sanitarios del ganado.

1. **Constructor_command_con_userid_nulo_lanza_ArgumentException**

- Objetivo:** asegurar que toda vacuna tenga un usuario registrado como responsable.
- Validación:**
 - Lanza `ArgumentException` con el mensaje "`RanchUserId must be set by the system`".
- Resultado esperado:** se impide la creación de registros sin responsable veterinario o administrador.



```

public class VaccineTests
{
    [Fact]
    public void Constructor_command_con_userid_nulo_lanza_ArgumentException()
    {
        // Arrange
        var cmd = new CreateVaccineCommand(
            Name: "Vac-A",
            VaccineType: "Antiparasitaria",
            VaccineDate: new DateTime(2025, 1, 10),
            VaccineImg: "v1.png",
            BovineId: 10,
            RanchUserId: null,
            FileData: null
        );

        // Act
        var ex = Assert.Throws<ArgumentException>(() => new Vaccine(cmd));

        // Assert
        Assert.Equal("RanchUserId must be set by the system", ex.Message);
    }
}

```

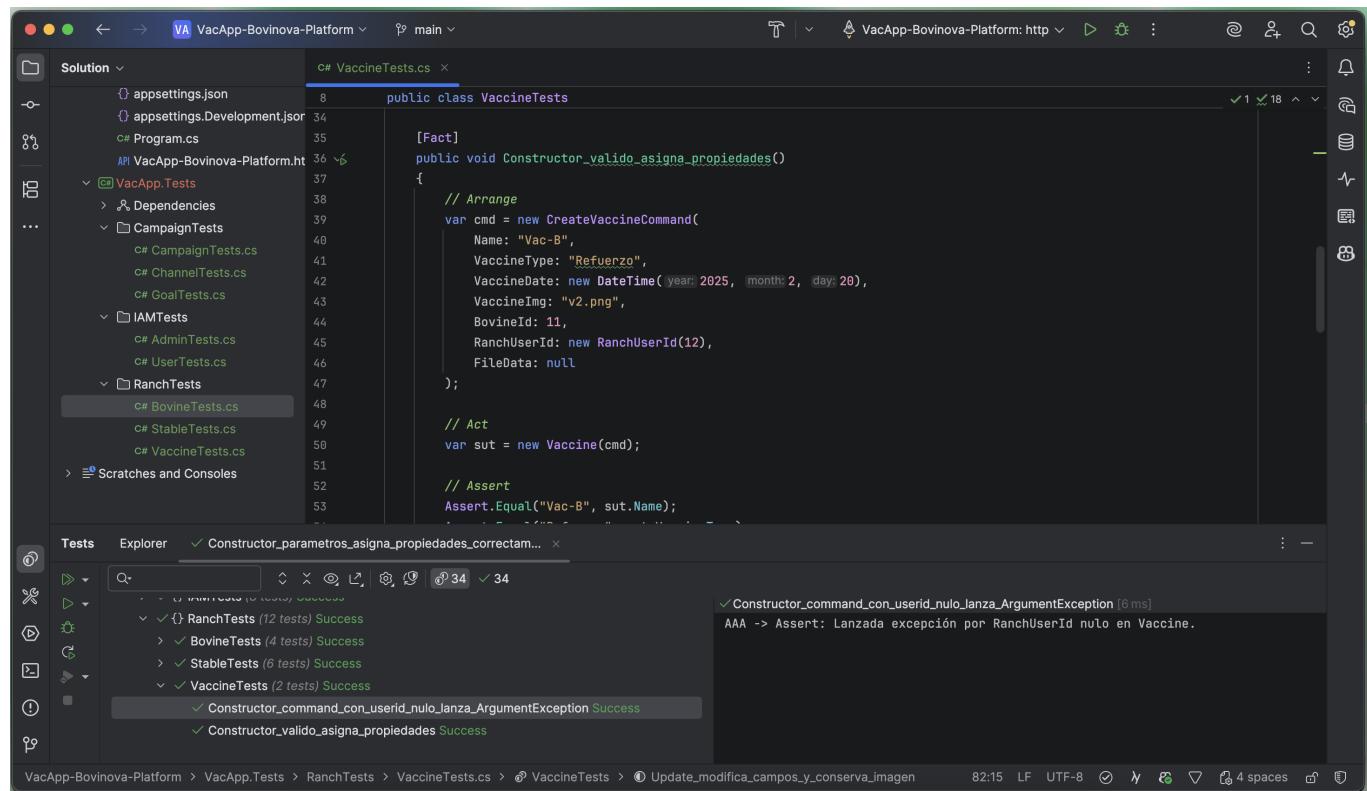
VaccineTests.cs

Tests Explorer

- RanchTests (11 tests) Success
 - BovineTests (4 tests) Success
 - StableTests (6 tests) Success
 - VaccineTests (1 test) Success
 - Constructor_command_con_userid_nulo_lanza_ArgumentException Success

2. Constructor_valido_asigna_propiedades

- Objetivo:** validar la creación correcta de una vacuna con valores válidos.
- Validaciones:**
 - Asignación correcta de nombre, tipo, fecha, imagen y bovino asociado.
- Resultado esperado:** el registro sanitario se crea de manera coherente con los datos de vacunación.



```

public class VaccineTests
{
    [Fact]
    public void Constructor_valido_asigna_propiedades()
    {
        // Arrange
        var cmd = new CreateVaccineCommand(
            Name: "Vac-B",
            VaccineType: "Refuerzo",
            VaccineDate: new DateTime(2025, 2, 20),
            VaccineImg: "v2.png",
            BovineId: 11,
            RanchUserId: new RanchUserId(12),
            FileData: null
        );

        // Act
        var sut = new Vaccine(cmd);

        // Assert
        Assert.Equal("Vac-B", sut.Name);
    }
}

```

VaccineTests.cs

Tests Explorer

- RanchTests (12 tests) Success
 - BovineTests (4 tests) Success
 - StableTests (6 tests) Success
 - VaccineTests (2 tests) Success
 - Constructor_command_con_userid_nulo_lanza_ArgumentException Success
 - Constructor_valido_asigna_propiedades Success

3. Update_modifica_campos_y_conserva_imagen

- **Objetivo:** asegurar que las actualizaciones modifiquen la información esencial sin alterar la imagen del registro anterior.
- **Validaciones:**
 - Cambian correctamente los campos **Name**, **VaccineType**, **VaccineDate**, **BovineId**.
 - La imagen original (**VaccineImg**) se conserva.
- **Resultado esperado:** el registro mantiene trazabilidad visual y consistencia temporal.

```

public class VaccineTests
{
    [Fact]
    public void Update_modifica_celdas_y_conserva_imagen()
    {
        // Arrange
        var created = new Vaccine(new CreateVaccineCommand(
            Name: "Vac-Init",
            VaccineType: "Tipo-1",
            VaccineDate: new DateTime(2025, 3, 1),
            VaccineImg: "init.png",
            BovineId: 5,
            RanchUserId: new RanchUserId(12),
            FileData: null
        ));
        var update = new UpdateVaccineCommand(
            Id: 23,
            Name: "Vac-New",
            VaccineType: "Tipo-2",
            VaccineDate: new DateTime(2025, 3, 15),
            BovineId: 6
        );
    }
}

```

Tests Explorer

- RanchTests (4 tests) Success
- BovineTests (6 tests) Success
- StableTests (6 tests) Success
- VaccineTests (3 tests) Success
 - Constructor_command_con_userid_nulo_lanza_ArgumentException Success
 - Constructor_valido_asigna_propiedades Success
 - Update_modifica_celdas_y_conserva_imagen Success

Campaign Management Unit Tests

El contexto **Campaign Management** de **VacApp** se encarga de la planificación, ejecución y seguimiento de campañas relacionadas con la salud y productividad del ganado.

Incluye las entidades **Campaign**, **Goal** y **Channel**, que representan la estructura principal de las campañas, sus metas operativas y los canales de comunicación utilizados.

Las pruebas unitarias desarrolladas con **C#** y **xUnit** tienen como propósito validar la creación de campañas, la gestión de estados y la relación con sus componentes asociados.

Clase: **CampaignTests**

La clase **CampaignTests** agrupa las pruebas que validan el comportamiento central de la entidad **Campaign**, asegurando que su construcción, actualización y manejo de listas internas cumplan con las reglas de negocio.

1. **Constructor_parametros_asigna_propiedades_correctamente**

- **Objetivo:** validar que la creación de una campaña con parámetros válidos asigne correctamente todas sus propiedades.
- **Validaciones:**

- Se asignan correctamente los valores de **Name**, **Description**, **StartDate**, **EndDate**, **Status**, **StableId**.
- Las listas **Goals** y **Channels** mantienen su referencia original.
- **Resultado esperado:** la campaña se inicializa con datos coherentes y referencias consistentes a sus listas asociadas.

The screenshot shows the Visual Studio IDE interface. The left sidebar displays the project structure for 'VacApp-Bovinova-Platform'. The 'Tests' tab is selected in the bottom-left corner. The 'Test Explorer' window shows a tree view of tests under 'VacApp.Tests'. One test, 'Constructor_parametros_asigna_propiedades_correctamente', is highlighted with a green checkmark and labeled 'Success'. The status bar at the bottom indicates the path 'rm > VacApp.Tests > CampaignTests > CampaignTests.cs > CampaignTests > Constructor_command_con_userid_nulo_lanza_ArgumentException', the time '59:15', and encoding 'UTF-8'.

```

public class CampaignTests
{
    [Fact]
    public void Constructor_parametros_asigna_propiedades_correctamente()
    {
        // Arrange
        var name = "Campaña A";
        var description = "Desc A";
        var start = new DateTime(2025, month: 1, day: 1);
        var end = new DateTime(2025, month: 12, day: 31);
        var status = "Draft";
        var goals = new List<Goal>();
        var channels = new List<Channel>();
        int? stableId = 10;
        CampaignUserId? userId = null;

        // Act
        var sut = new Campaign(name, description, start, end, status, goals, channels, stableId, userId);

        // Assert
        Assert.Equal(name, sut.Name);
        Assert.Equal(description, sut.Description);
        Assert.Equal(start, sut.StartDate);
        Assert.Equal(end, sut.EndDate);
    }
}

```

2. **Constructor_command_con_userid_nulo_lanza_ArgumentException**

- **Objetivo:** asegurar que toda campaña cuente con un usuario asociado del sistema.
- **Escenario:** se crea un **CreateCampaignCommand** con **CampaignUserId = null**.
- **Validación:**
 - Se lanza una excepción **ArgumentException** con el mensaje "**User Id must be set by the system**".
- **Resultado esperado:** se impide crear campañas sin trazabilidad del usuario responsable.

The screenshot shows the Visual Studio IDE interface. The top navigation bar includes tabs for 'main' and 'VacApp-Bovinova-Platform: http'. The left sidebar displays the project structure under 'Solution'. The main editor window shows the code for 'CampaignTests.cs'.

```

public class CampaignTests
{
    [Fact]
    public void Constructor_command_con_userid_nulo_lanza_ArgumentException()
    {
        // Arrange
        var cmd = new CreateCampaignCommand(
            Name: "Campaña B",
            Description: "Desc",
            StartDate: new DateTime(year: 2025, month: 2, day: 1),
            EndDate: new DateTime(year: 2025, month: 3, day: 1),
            Status: "Draft",
            Goals: new List<Goal>(),
            Channel: new List<Channel>(),
            StableId: null,
            CampaignUserId: null
        );

        // Act
        var ex = Assert.Throws<ArgumentException>(() => new Campaign(cmd));

        // Assert
        Assert.Equal("UserId must be set by the system", ex.Message);
    }
}

```

The 'Tests' tab in the bottom-left corner is selected, showing the test results for 'CampaignTests'. The 'CampaignTests' node has two children: 'Constructor_command_con_userid_nulo_lanza_ArgumentException' and 'Constructor_parametros_asigna_propiedades_correctamente', both marked as 'Success'.

3. UpdateStatus_cambia_el_estado_de_la_campania

- Objetivo:** validar la capacidad de cambiar el estado de la campaña entre fases predefinidas.
- Escenario:** el estado inicial es "**Draft**" y se actualiza a "**Active**", "**Paused**" o "**Completed**".
- Validación:**
 - El campo **Status** cambia correctamente al nuevo valor.
- Resultado esperado:** la campaña refleja correctamente su progreso operativo.

The screenshot shows the Visual Studio IDE interface. The top navigation bar includes tabs for 'main' and 'VacApp-Bovinova-Platform: http'. The left sidebar displays the project structure under 'Solution'. The main editor window shows the code for 'CampaignTests.cs'.

```

public class CampaignTests
{
    [Theory]
    [InlineData("Active")]
    [InlineData("Paused")]
    [InlineData("Completed")]
    public void UpdateStatus_cambia_el_estado_de_la_campania(string nuevoEstado)
    {
        // Arrange
        var sut = new Campaign(
            name: "Campaña C",
            description: "Desc",
            startDate: new DateTime(year: 2025, month: 1, day: 1),
            endDate: new DateTime(year: 2025, month: 1, day: 31),
            status: "Draft",
            goals: new List<Goal>(),
            channels: new List<Channel>(),
            stableId: null,
            campaignUserId: null
        );
    }
}

```

The 'Tests' tab in the bottom-left corner is selected, showing the test results for 'CampaignTests'. The 'UpdateStatus_cambia_el_estado_de_la_campania' node has three children: 'Constructor_command_con_userid_nulo_lanza_ArgumentException', 'Constructor_parametros_asigna_propiedades_correctamente', and 'UpdateStatus_cambia_el_estado_de_la_campania', all marked as 'Success'.

4. Constructor_parametros_preserva_referencias_de_listas

- **Objetivo:** garantizar que las listas **Goals** y **Channels** inyectadas mantengan la misma referencia en memoria.
- **Validación:**
 - **Assert.Same(goals, sut.Goals)** y **Assert.Same(channels, sut.Channels)**.
- **Resultado esperado:** la entidad conserva las referencias originales para evitar duplicaciones en la agregación.

The screenshot shows the Visual Studio IDE interface. The left sidebar displays the solution structure for 'VacApp-Bovinova-Platform'. The main editor window shows the code for 'CampaignTests.cs'.

```

public class CampaignTests
{
    public void UpdateStatus_cambia_el_estado_de_la_campaña(string nuevoEstado)
    {
        [Fact]
        public void Constructor_parametros_preserva_referencias_de_listas()
        {
            // Arrange
            var goals = new List<Goal>();
            var channels = new List<Channel>();

            // Act
            var sut = new Campaign(
                name: "Campaña D",
                description: "Desc",
                startDate: new DateTime(year: 2025, month: 4, day: 1),
                endDate: new DateTime(year: 2025, month: 4, day: 30),
                status: "Draft",
                goals: goals,
                channels: channels,
                stableId: null,
                campaignUserId: null
            );
        }
    }
}

```

The 'Test Explorer' window at the bottom shows the results of the test run. It lists several tests under 'CampaignTests' and 'UpdateStatus_cambia_el_estado_de_la_campaña', all marked as 'Success'.

5. AddGoal_agrega_elemento_a_la_colección

- **Objetivo:** comprobar que el método **AddGoal()** agrega correctamente un nuevo elemento a la colección interna.
- **Validación:**
 - El conteo de **Goals** aumenta en uno tras la operación.
- **Resultado esperado:** la campaña puede registrar nuevas metas de forma dinámica.

```

public class CampaignTests
{
    [Fact]
    public void AddGoal_agrega_elemento_a_la_coleccion()
    {
        // Arrange
        var sut = new Campaign(
            name: "Campaña E",
            description: "Desc",
            startDate: new DateTime(year: 2025, month: 5, day: 1),
            endDate: new DateTime(year: 2025, month: 5, day: 31),
            status: "Draft",
            goals: new List<Goal>(),
            channels: new List<Channel>(),
            stableId: null,
            campaignUserId: null
        );
    }
}

```

Tests Explorer `Constructor_parametros_asigna_propiedades_correctamente`

- `AddGoal_agrega_elemento_a_la_colección` Success
- `Constructor_command_con_userid_nulo_lanza_ArgumentException` Success
- `Constructor_parametros_asigna_propiedades_correctamente` Success
- `Constructor_parametros_preserva_referencias_de_listas` Success
- `UpdateStatus_cambia_el_estado_de_la_campaña` (3 tests) Success
 - `UpdateStatus_cambia_el_estado_de_la_campaña(nuevoEstado: "Active")` Success
 - `UpdateStatus_cambia_el_estado_de_la_campaña(nuevoEstado: "Completed")` Success
 - `UpdateStatus_cambia_el_estado_de_la_campaña(nuevoEstado: "Paused")` Success

6. AddChannel_agrega_elemento_a_la_colección

- Objetivo:** verificar que el método `AddChannel()` incremente la lista interna de canales.
- Validación:**
 - El número de elementos en `Channels` aumenta en uno.
- Resultado esperado:** la campaña incorpora correctamente nuevos medios de difusión.

```

public class CampaignTests
{
    [Fact]
    public void AddChannel_agrega_elemento_a_la_colección()
    {
        // Arrange
        var sut = new Campaign(
            name: "Campaña F",
            description: "Desc",
            startDate: new DateTime(year: 2025, month: 6, day: 1),
            endDate: new DateTime(year: 2025, month: 6, day: 30),
            status: "Draft",
            goals: new List<Goal>(),
            channels: new List<Channel>(),
            stableId: null,
            campaignUserId: null
        );
        var countAntes = sut.Channels.Count;
    }
}

```

Tests Explorer `Constructor_parametros_asigna_propiedades_correctamente`

- `AddChannel_agrega_elemento_a_la_colección` Success
- `AddGoal_agrega_elemento_a_la_colección` Success
- `Constructor_command_con_userid_nulo_lanza_ArgumentException` Success
- `Constructor_parametros_asigna_propiedades_correctamente` Success
- `Constructor_parametros_preserva_referencias_de_listas` Success
- `UpdateStatus_cambia_el_estado_de_la_campaña` (3 tests) Success

Clase: `GoalTests`

Las pruebas de la entidad **Goal** se enfocan en la gestión de metas dentro de una campaña, validando su creación, persistencia y referencia compartida con la campaña principal.

1. Goals_inicialmente_vacia_cuando_se_inyecta_lista_vacia

- **Objetivo:** comprobar que la lista de metas inicia vacía cuando se proporciona una lista vacía.
- **Resultado esperado:** el conteo inicial de **Goals** es cero.

The screenshot shows the Visual Studio IDE interface. The top navigation bar displays 'VacApp-Bovinova-Platform' and 'main'. The code editor window shows the 'GoalTests.cs' file, which contains a test method named 'Goals_inicialmente_vacia_cuando_se_inyecta_lista_vacia'. The test arranges a new 'Campaign' object with a name ('Campaña G'), description ('Desc'), start date ('2025-07-01'), end date ('2025-07-31'), status ('Draft'), and an empty goals list. The 'Tests' tab in the bottom navigation bar is selected, showing the 'GoalTests' node expanded, with the specific test case highlighted. The status of the test is 'Success' with a green checkmark icon. The status bar at the bottom right shows the time as 20:43, the encoding as LF, and the file format as UTF-8.

```

public class GoalTests
{
    [Fact]
    public void Goals_inicialmente_vacia_cuando_se_inyecta_lista_vacia()
    {
        // Arrange
        var goals = new List<Goal>();
        var sut = new Campaign(
            name: "Campaña G",
            description: "Desc",
            startDate: new DateTime(2025, month: 7, day: 1),
            endDate: new DateTime( year: 2025, month: 7, day: 31),
            status: "Draft",
            goals: goals,
            channels: new List<Channel>(),
            stableId: null,
            campaignUserId: null
        );
    }
}

```

2. AddGoal_incrementa_conteo_de_goals

- **Objetivo:** verificar que el método **AddGoal()** incremente correctamente el número de metas.
- **Validación:**
 - El conteo aumenta en dos tras dos adiciones.
- **Resultado esperado:** las metas pueden añadirse dinámicamente a la campaña.

```

public class GoalTests
{
    [Fact]
    public void AddGoal_incrementa_conteo_de_goals()
    {
        // Arrange
        var sut = new Campaign(
            name: "Campaña H",
            description: "Desc",
            startDate: new DateTime(year: 2025, month: 8, day: 1),
            endDate: new DateTime(year: 2025, month: 8, day: 31),
            status: "Draft",
            goals: new List<Goal>(),
            channels: new List<Channel>(),
            stableId: null,
            campaignUserId: null
        );
        // Act
        sut.AddGoal(goal: null);
    }
}

```

Tests Explorer: VacApp.Tests (13 tests) Success
 VacApp.Tests.CampaignTests (13 tests) Success
 CampaignTests (8 tests) Success
 ChannelTests (3 tests) Success
 GoalTests (2 tests) Success
 AddGoal_incrementa_conteo_de_goals Success
 Goals_inicialmente_vacia_cuando_se_inyecta_lista_vacia Success

3. Goals_preserva_referencia_compartida

- **Objetivo:** asegurar que las listas de metas mantienen la misma referencia cuando se modifica desde fuera del objeto campaña.
- **Validación:**
 - `Assert.Same(goals, sut.Goals)`
 - `Assert.Equal(goals.Count, sut.Goals.Count)`
- **Resultado esperado:** se garantiza la coherencia entre las referencias internas y externas de la colección de metas.

```

public class GoalTests
{
    [Fact]
    public void Goals_preserva_referencia_compartida()
    {
        // Arrange
        var goals = new List<Goal>();
        var sut = new Campaign(
            name: "Campaña I",
            description: "Desc",
            startDate: new DateTime( year: 2025, month: 9, day: 1),
            endDate: new DateTime( year: 2025, month: 9, day: 30),
            status: "Draft",
            goals: goals,
            channels: new List<Channel>(),
            stableId: null,
            campaignUserId: null
        );
        // Act
        goals.Add(item: null);
    }
}

```

Tests Explorer

- VacApp.Tests (14 tests) Success
 - VacApp.Tests.CampaignTests (14 tests) Success
 - CampaignTests (8 tests) Success
 - Channels_inicialmente_vacia_cuando_se_inyecta_lista_vacia Success
 - Goals_preserva_referencia_compartida Success
 - ChannelTests (3 tests) Success
 - GoalTests (3 tests) Success
 - AddGoal_incremente_conteo_de_goals Success
 - Goals_inicialmente_vacia_cuando_se_inyecta_lista_vacia Success
 - Goals_preserva_referencia_compartida Success

Clase: ChannelTests

Las pruebas de la entidad **Channel** validan el correcto manejo de los canales de comunicación o difusión asociados a cada campaña.

1. Channels_inicialmente_vacia_cuando_se_inyecta_lista_vacia

- Objetivo:** comprobar que la lista de canales inicia vacía al recibir una colección vacía.
- Resultado esperado:** `Channels.Count == 0`.

```

public class ChannelTests
{
    [Fact]
    public void Channels_inicialmente_vacia_cuando_se_inyecta_lista_vacia()
    {
        // Arrange
        var channels = new List<Channel>();
        var sut = new Campaign(
            name: "Campaña J",
            description: "Desc",
            startDate: new DateTime( year: 2025, month: 10, day: 1),
            endDate: new DateTime( year: 2025, month: 10, day: 31),
            status: "Draft",
            goals: new List<Goal>(),
            channels: channels,
            stableId: null,
            campaignUserId: null
        );
        // Act
    }
}

```

Tests Explorer

- VacApp.Tests (9 tests) Success
 - VacApp.Tests.CampaignTests (9 tests) Success
 - CampaignTests (8 tests) Success
 - ChannelTests (1 test) Success
 - Channels_inicialmente_vacia_cuando_se_inyecta_lista_vacia Success

2. AddChannel_increments_channel_count

- **Objetivo:** verificar que el método `AddChannel()` incremente el número de canales dentro de la campaña.
- **Validación:**
 - El conteo final de `Channels` es dos luego de dos llamadas al método.
- **Resultado esperado:** los canales se agregan correctamente al plan de difusión de la campaña.

The screenshot shows the Visual Studio IDE interface. The top navigation bar displays 'VacApp-Bovinova-Platform' and 'main'. The left sidebar shows the project structure under 'Solution'. The main editor window contains the code for `ChannelTests.cs`, specifically the `AddChannel_increments_channel_count()` test. The code sets up a `Campaign` instance with two channels and then adds a third channel, resulting in a total of three channels. The bottom part of the screen shows the 'Test Explorer' window, which lists the test results: 'VacApp.Tests (10 tests) Success' and 'CampaignTests (8 tests) [40 ms]'. The 'ChannelTests (2 tests) Success' section is expanded, showing the two individual test cases: 'AddChannel_increments_channel_count Success' and 'Channels_inicialmente_vacia_cuando_se_inyecta_lista_vacia Success'.

```

public class ChannelTests
{
    [Fact]
    public void AddChannel_increments_channel_count()
    {
        // Arrange
        var sut = new Campaign(
            name: "Campaña K",
            description: "Desc",
            startDate: new DateTime(2025, 11, 1),
            endDate: new DateTime(2025, 11, 30),
            status: "Draft",
            goals: new List<Goal>(),
            channels: new List<Channel>(),
            stableId: null,
            campaignUserId: null
        );

        // Act
        sut.AddChannel(channel: null);
    }
}

```

3. Channels_preserves_shared_reference

- **Objetivo:** garantizar que las listas de canales mantienen la misma referencia en memoria cuando se modifican desde fuera de la campaña.
- **Validación:**
 - `Assert.Same(channels, sut.Channels)`
 - `Assert.Equal(channels.Count, sut.Channels.Count)`
- **Resultado esperado:** las referencias de las listas se conservan, evitando pérdida de sincronización o duplicación de datos.

```

public class ChannelTests
{
    [Fact]
    public void Channels_preserva_referencia_compartida()
    {
        // Arrange
        var channels = new List<Channel>();
        var sut = new Campaign(
            name: "Campaña L",
            description: "Desc",
            startDate: new DateTime( year: 2025, month: 12, day: 1),
            endDate: new DateTime( year: 2025, month: 12, day: 31),
            status: "Draft",
            goals: new List<Goal>(),
            channels: channels,
            stableId: null,
            campaignUserId: null
        );
    }

    // Act
}

```

Tests Explorer pane showing 11 successful tests, including three for ChannelTests.

Staff Administration Unit Tests

El contexto **Staff Administration** en **VacApp** se encarga de gestionar al personal involucrado en las campañas ganaderas, incluyendo veterinarios, operarios y administradores.

Estas pruebas unitarias se implementaron en **C# con xUnit**, aplicando la metodología **AAA (Arrange / Act / Assert)**, con el fin de validar la integridad de la creación, actualización y vinculación del personal con campañas activas dentro del sistema.

Clase: StaffTests

Las pruebas verifican que la entidad **Staff** cumpla con las reglas de negocio asociadas al registro del personal, asegurando que los campos esenciales se asignen correctamente y que no se creen registros incompletos o inconsistentes.

1. Constructor_por_defecto_inicializa_campos_por_defecto

- **Objetivo:** comprobar que, al crear una instancia sin parámetros, el objeto se inicializa en un estado seguro.
- **Validaciones:**
 - **Name = string.Empty.**
 - **EmployeeStatus** no es nulo.
 - **CampaignId** y **StaffUserId** son **null**.
- **Resultado esperado:** la entidad comienza con valores controlados, lista para ser configurada mediante comandos válidos.

The screenshot shows the Visual Studio IDE interface. The left pane displays the Solution Explorer with a tree view of project files, including API VacApp-Bovinova-Platform, VacApp.Tests, and various test files like CampaignTests.cs, IAMTests.cs, RanchTests.cs, and StaffTests.cs. The right pane shows the code editor with the StaffTests.cs file open, containing C# unit test code. Below the code editor is the Test Explorer window, which lists all test cases under the StaffTests category, all of which are marked as 'Success'. A detailed view of one test case, 'Constructor_por_defecto_inicializa_campos_por_defecto', is shown in the bottom right, displaying its execution details and results.

```

public class StaffTests
{
    private static string Dump(Staff s) =>
        $"Staff{{ Id={s.Id}, Name='{s.Name}', EmployeeStatus={s.EmployeeStatus}, CampaignId={(s.CampaignId?.ToString() ?? "null")}}}";

    [Fact]
    public void Constructor_por_defecto_inicializa_campos_por_defecto()
    {
        // Arrange
        _output.WriteLine("AAA -> Arrange: sin parámetros");

        // Act
        var sut = new Staff();
        _output.WriteLine($"AAA -> Act: creado -> {Dump(sut)}");

        // Assert
        Assert.Equal(string.Empty, sut.Name);
        Assert.NotNull(sut.EmployeeStatus);
        Assert.Null(sut.CampaignId);
        Assert.Null(sut.StaffUserId);

        _output.WriteLine("AAA -> Assert: Name='', EmployeeStatus!=null, CampaignId=null, StaffUserId=null");
    }
}

```

2. Constructor_parametros_valido_asigna_propiedades

- **Objetivo:** verificar que al crear un objeto **Staff** con parámetros válidos, estos se asignen correctamente.
- **Validaciones:**
 - Los valores de **Name**, **EmployeeStatus**, **CampaignId** y **StaffUserId** se asignan correctamente.
- **Resultado esperado:** el registro del personal se crea con datos coherentes y trazables dentro del sistema.

```

public class StaffTests
{
    [Fact]
    public void Constructor_parametros_valido_asigna_propiedades()
    {
        // Arrange
        var name = "Alice";
        var employeeStatus = 1;
        int? campaignId = 10;
        var staffUserId = new StaffUserId(1);
        _output.WriteLine($"AAA -> Arrange: name='{name}', employeeStatus={employeeStatus}, campaignId={campaignId}, staffUserId={staffUserId}");
    }

    // Act
    var sut = new Staff(name, employeeStatus, campaignId, staffUserId);
    _output.WriteLine($"AAA -> Act: creado -> {Dump(sut)}");

    // Assert
    Assert.Equal(name, sut.Name);
    Assert.NotNull(sut.EmployeeStatus);
    Assert.Equal(campaignId, sut.CampaignId);
    Assert.Equal(staffUserId, sut.StaffUserId);
    output.WriteLine("AAA -> Assert: propiedades asignadas según Arrange");
}

```

Tests Explorer: 37 successful tests

3. Constructor_con_CreateStaffCommand_valido_asigna_propiedades

- Objetivo:** validar que el constructor basado en `CreateStaffCommand` funcione de manera correcta.
- Validaciones:**
 - El nombre, estado y campaña son los esperados.
 - `StaffUserId` se asigna de forma correcta y única.
- Resultado esperado:** la creación mediante comandos garantiza que las reglas del dominio se cumplan desde el origen.

```

public class StaffTests
{
    [Fact]
    public void Constructor_con_CreateStaffCommand_valido_asigna_propiedades()
    {
        // Arrange
        var name = "Bob";
        var employeeStatus = 2;
        int? campaignId = 7;
        var staffUser = new StaffUserId(42);
        _output.WriteLine($"AAA -> Arrange(cmd): Name='{name}', EmployeeStatus={employeeStatus}, CampaignId={campaignId}, StaffUserId={staffUser}");
        var cmd = new CreateStaffCommand(Name: name, EmployeeStatus: employeeStatus, CampaignId: campaignId, StaffUserId: staffUser);

        // Act
        var sut = new Staff(cmd);
        _output.WriteLine($"AAA -> Act: creado -> {Dump(sut)}");

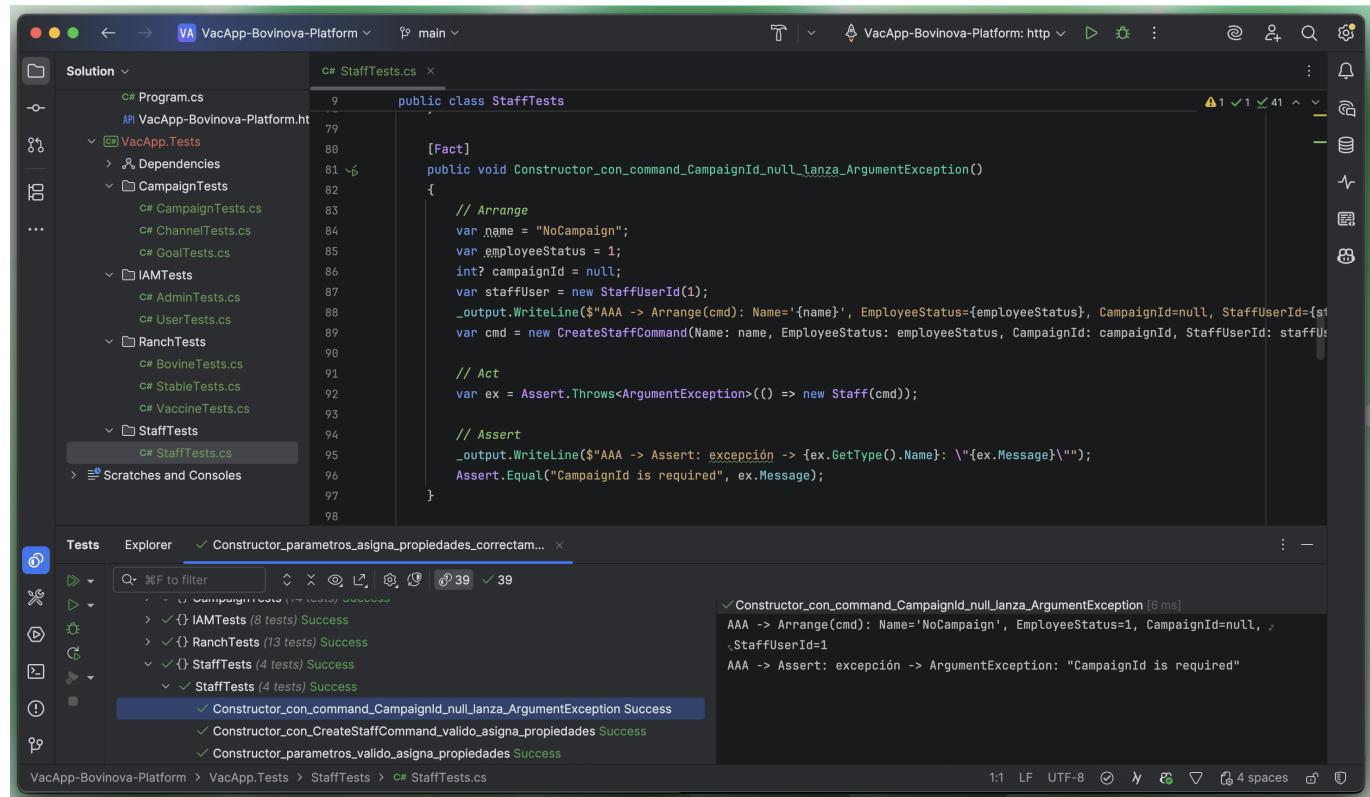
        // Assert
        Assert.Equal(name, sut.Name);
        Assert.NotNull(sut.EmployeeStatus);
        Assert.Equal(campaignId, sut.CampaignId);
        Assert.Equal(staffUser.Identifier, sut.StaffUserId!.Identifier);
    }
}

```

Tests Explorer: 38 successful tests

4. Constructor_con_command_CampaignId_null_lanza_ArgumentException

- Objetivo:** garantizar que cada miembro del personal esté asociado a una campaña.
- Escenario:** se pasa `CampaignId = null` en el comando de creación.
- Validación:**
 - Se lanza una excepción `ArgumentException` con el mensaje "`CampaignId is required`".
- Resultado esperado:** evita la creación de personal sin asignación a campaña, preservando la consistencia del modelo.



```

public class StaffTests
{
    [Fact]
    public void Constructor_con_command_CampaignId_null_lanza_ArgumentException()
    {
        // Arrange
        var name = "NoCampaign";
        var employeeStatus = 1;
        int? campaignId = null;
        var staffUser = new StaffUserId(1);
        _output.WriteLine($"AAA -> Arrange(cmd): Name='{name}', EmployeeStatus={employeeStatus}, CampaignId=null, StaffUserId={staffUser.Id}");
        var cmd = new CreateStaffCommand(Name: name, EmployeeStatus: employeeStatus, CampaignId: campaignId, StaffUserId: staffUser.Id);

        // Act
        var ex = Assert.Throws<ArgumentException>(() => new Staff(cmd));

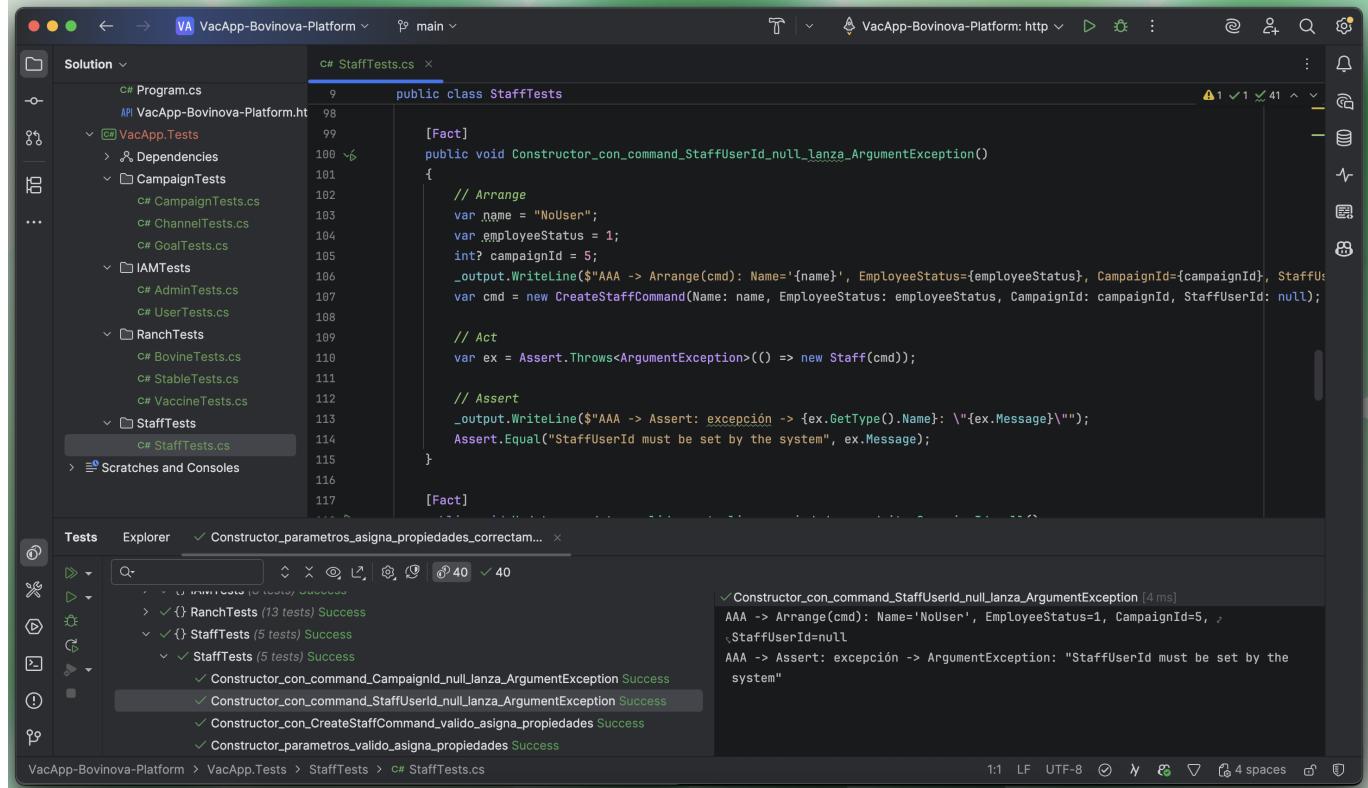
        // Assert
        _output.WriteLine($"AAA -> Assert: excepción -> {ex.GetType().Name}: \"{ex.Message}\"");
        Assert.Equal("CampaignId is required", ex.Message);
    }
}

```

The screenshot shows the Visual Studio IDE interface. The left pane displays the solution structure with projects like Program.cs, VacApp-Bovinova-Platform, and various test files under VacApp.Tests. The right pane shows the code editor for StaffTests.cs with the above test implementation. Below the editor is the Test Explorer window, which lists several test results. One specific test, 'Constructor_con_command_CampaignId_null_lanza_ArgumentException', is highlighted in blue and marked as 'Success'. The status bar at the bottom indicates the file is 'VacApp-Bovinova-Platform' and the path is 'VacApp.Tests > StaffTests > C# StaffTests.cs'.

5. Constructor_con_command_StaffUserId_null_lanza_ArgumentException

- Objetivo:** comprobar que todo miembro del personal tenga un identificador de usuario del sistema.
- Validación:**
 - Se lanza una excepción `ArgumentException` con el mensaje "`StaffUserId must be set by the system`".
- Resultado esperado:** impide la creación de registros anónimos o sin responsable definido.



The screenshot shows the Visual Studio IDE interface. The left pane displays the Solution Explorer with various test projects and files. The right pane shows the code editor for `StaffTests.cs`. The code contains two test methods: `Constructor_con_command_StaffUserId_null_lanza_ArgumentException` and `Constructor_parametros_valido_asigna_propiedades`. The test for `Constructor_parametros_valido_asigna_propiedades` is currently selected. The bottom pane shows the Test Explorer and the Output window, both indicating successful test runs.

```

public class StaffTests
{
    [Fact]
    public void Constructor_con_command_StaffUserId_null_lanza_ArgumentException()
    {
        // Arrange
        var name = "NoUser";
        var employeeStatus = 1;
        int? campaignId = 5;
        _output.WriteLine($"AAA -> Arrange(cmd): Name='{name}', EmployeeStatus={employeeStatus}, CampaignId={campaignId}, StaffUserId=null");
        var cmd = new CreateStaffCommand(Name: name, EmployeeStatus: employeeStatus, CampaignId: campaignId, StaffUserId: null);

        // Act
        var ex = Assert.Throws<ArgumentException>(() => new Staff(cmd));

        // Assert
        _output.WriteLine($"AAA -> Assert: excepción -> {ex.GetType().Name}: \"{ex.Message}\"");
        Assert.Equal("StaffUserId must be set by the system", ex.Message);
    }

    [Fact]
    public void Constructor_parametros_valido_asigna_propiedades()
    {
        // Arrange
        var name = "ValidUser";
        var employeeStatus = 1;
        int? campaignId = 5;
        var staff = new Staff();
        staff.Name = name;
        staff.EmployeeStatus = employeeStatus;
        staff.CampaignId = campaignId;

        // Act
        var ex = Assert.Throws<InvalidOperationException>(() => staff.Update());
    }
}

```

6. Update_con_datos_validos_actualiza_propiedades_y_admite_CampaignId_null

- Objetivo:** validar que el método `Update()` permita modificar las propiedades básicas del personal y opcionalmente dejar `CampaignId` en `null`.
- Validaciones:**
 - Se actualizan `Name` y `EmployeeStatus`.
 - `CampaignId` puede quedar sin asignar.
- Resultado esperado:** permite flexibilidad en la reubicación o desvinculación del personal respecto a una campaña.

```

public class StaffTests
{
    [Fact]
    public void Update_con_datos_validos_actualiza_propiedades_y_admite_CampaignId_null()
    {
        // Arrange
        var sut = new Staff( name: "Init", employeeStatus: 1, campaignId: 10, new StaffUserId(7));
        _output.WriteLine($"AAA -> Arrange: before -> {Dump(sut)}");
        var updateId = 3; // si tu comando requiere Id
        var update = new UpdateStaffCommand(Id: updateId, Name: "Updated", EmployeeStatus: 3, CampaignId: null);
        _output.WriteLine("AAA -> Arrange(update): Id=3, Name='Updated', EmployeeStatus=3, CampaignId=null");

        // Act
        sut.Update(update);
        _output.WriteLine($"AAA -> Act: after -> {Dump(sut)}");

        // Assert
        Assert.Equal("Updated", sut.Name);
        Assert.NotNull(sut.EmployeeStatus);
        Assert.Null(sut.CampaignId);
        _output.WriteLine("AAA -> Assert: nombre y estado actualizados, CampaignId=null");
    }
}

```

Tests Explorer

- Constructor_parametros_asigna_propiedades_correctamente Success
- Constructor_con_Command_CampaignId_null_lanza_ArgumentException Success
- Constructor_con_Command_StaffUserId_null_lanza_ArgumentException Success
- Constructor_con_CreateStaffCommand_valido_asigna_propiedades Success
- Constructor_parametros_valido_asigna_propiedades Success
- Constructor_por_defecto_inicializa_campos_por_defecto Success
- Update_con_datos_validos_actualiza_propiedades_y_admite_CampaignId_null Success

7. Update_con_datos_validos_actualiza_propiedades_con_CampaignId_no_null

- Objetivo:** garantizar que el método `Update()` funcione correctamente cuando se asigna un nuevo `CampaignId`.
- Validaciones:**
 - El nombre y estado cambian correctamente.
 - `CampaignId` se actualiza al nuevo valor proporcionado.
- Resultado esperado:** el sistema permite reasignar personal a nuevas campañas de forma segura.

```

public class StaffTests
{
    [Fact]
    public void Update_con_datos_validos_actualiza_propiedades_con_CampaignId_no_null()
    {
        // Arrange
        var sut = new Staff( name: "Init", employeeStatus: 1, campaignId: 10, new StaffUserId(7));
        _output.WriteLine($"AAA -> Arrange: before -> Staff{ Id=0, Name='Init', EmployeeStatus=EmployeeStatus { Value = 1 }, CampaignId=10, StaffUserId=7 }");
        var updateId = 3; // si tu comando requiere Id
        var update = new UpdateStaffCommand(Id: updateId, Name: "Updated2", EmployeeStatus: 4, CampaignId: 99);
        _output.WriteLine("AAA -> Arrange(update): Id=3, Name='Updated2', EmployeeStatus=4, CampaignId=99");

        // Act
        sut.Update(update);
        _output.WriteLine($"AAA -> Act: after -> {Dump(sut)}");

        // Assert
        Assert.Equal("Updated2", sut.Name);
        Assert.NotNull(sut.EmployeeStatus);
        Assert.Equal(99, sut.CampaignId);
        _output.WriteLine("AAA -> Assert: nombre y estado actualizados, CampaignId=99");
    }
}

```

Tests Explorer

- Constructor_parametros_asigna_propiedades_correctamente Success
- Constructor_con_Command_CampaignId_null_lanza_ArgumentException Success
- Constructor_con_Command_StaffUserId_null_lanza_ArgumentException Success
- Constructor_con_CreateStaffCommand_valido_asigna_propiedades Success
- Constructor_parametros_valido_asigna_propiedades Success
- Constructor_por_defecto_inicializa_campos_por_defecto Success
- Update_con_datos_validos_actualiza_propiedades_con_CampaignId_no_null Success
- Update_con_datos_validos_actualiza_propiedades_y_admite_CampaignId_null Success

Resultados Generales de las Pruebas Unitarias

La siguiente imagen muestra la **ejecución total de las pruebas unitarias** desarrolladas para los diferentes contextos de la plataforma **VacApp**: *IAM, Ranch Management, Campaign Management y Staff Administration*.

Los resultados reflejan que **todas las pruebas unitarias pasaron exitosamente**, evidenciando un correcto comportamiento de las reglas de negocio implementadas y asegurando la estabilidad del dominio de la aplicación.

- Se ejecutaron **todas las pruebas definidas** para cada agregado raíz sin fallos.
- La cobertura abarca **validaciones de constructores, actualizaciones de entidades y manejo de excepciones controladas**.
- El sistema mantiene **coherencia entre las entidades dependientes**, cumpliendo los principios de *Domain-Driven Design (DDD)*.

The screenshot shows the Visual Studio interface with the following details:

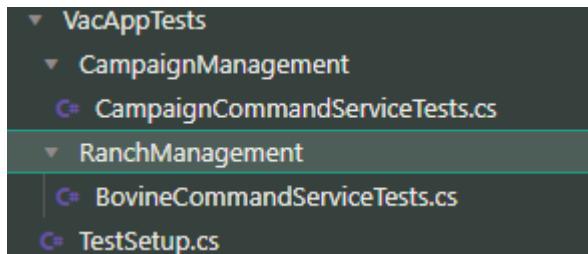
- Solution Explorer:** Shows the project structure under "API VacApp-Bovinova-Platform".
 - VacApp.Tests**: Contains:
 - Dependencies
 - CampaignTests.cs
 - ChannelTests.cs
 - GoalTests.cs
 - IAMTests
 - AdminTests.cs
 - UserTests.cs
 - RanchTests
- Code Editor:** Displays the content of StaffTests.cs.

```
public class StaffTests
{
    [Fact]
    public void Update_con_datos_validos_actualiza()
    {
        // Arrange
        var sut = new Staff(name: "Init", employeeId: 1);
        _output.WriteLine($"AAA -> Arrange: before update");
        var updateId = 3; // si tu comando requiere de un ID
        var update = new UpdateStaffCommand(Id: updateId, name: "Update", employeeId: 1);
        _output.WriteLine("AAA -> Arrange(update)");
    }
}
```
- Tests Explorer:** Shows the test results for VacApp.Tests.
 - VacApp.Tests (42 tests) Success**:
 - {} VacApp.Tests (42 tests) Success**:
 - {} CampaignTests (14 tests) Success**:
 - CampaignTests (8 tests) Success**
 - ChannelTests (3 tests) Success**
 - GoalTests (3 tests) Success**
 - {} IAMTests (8 tests) Success**:
 - AdminTests (4 tests) Success**
 - UserTests (4 tests) Success**
 - {} RanchTests (13 tests) Success**:
 - BovineTests (4 tests) Success**
 - StableTests (6 tests) Success**
 - VaccineTests (3 tests) Success**
 - {} StaffTests (7 tests) Success**:
 - StaffTests (7 tests) Success**
- Status Bar:** Shows the path "VacApp-Bovinova-Platform > VacApp.Tests > StaffTests > C# StaffTests.cs" and a GitHub Des... button.

6.1.2. Core Integration Tests

Luego de haber acabado con los Unit Tests en ciertos puntos del Backend realizado en .NET 8, se realizo los test de Integracion para verificar que los avances realizados son completamente funcionales. Junto a Nunit instalado para los Unit Test anteriores, tambien se insalo la dependencia de Moq y Microsoft Entity Framework InMemory para testear que las aplicaciones funcionen en persistencia dentro de una base de datos ficticia que sigue los parametros establecidos por el DBContext de la solucion.

Para esto se separo en una carpeta externa para los Tests



Luego de esto se creo el TestSetup, que sirve para generar el TestDb que se utilizara para las pruebas de integracion.

```
[TestFixture]
0 references
public abstract class TestSetup
{
    1 reference
    protected DbContextOptions<DbContext>
    DbContextOptions;    Non-nullble field 'DbContextOpti

    [SetUp]
    0 references
    public void BaseSetUp()
    {
        DbContextOptions = new
        DbContextOptionsBuilder<DbContext>()
            .UseInMemoryDatabase($"TestDb_{Guid.NewGuid()
            ()}")
            .Options;
    }
}
```

Asimismo se hieron los test de integracion en dos de los Contextos mas importantes de la aplicacion, como lo vendria a ser CampaignManagment y RanchManagment. Estos dos fueron elegidos debido a la complejidad que tienen dentro de si.

Primero con **CampaignCommandServiceTest** se realizaron dos test, donde se testeia la capacidad de agregar una campana y para validar las restricciones que se diseñaron para este contexto, donde no se puede crear dos campañas con el mismo nombre.

```
[Test]
0 references
public async Task
Handle_CreateCampaignCommand_ShouldAddCampaign()
{
    // Arrange
    var goals = new List<Goal> { new Goal("Increase
    Sales", "Sales", 100, 0, 0) };
    var channels = new List<Channel> { new Channel
    ("Email", "email@example.com", 0) };
    var command = new CreateCampaignCommand(
        "Test Campaign",
        "Description",
        DateTime.Now,
        DateTime.Now.AddDays(10),
        "Active",
        goals,
        channels,
        1,
        new CampaignUserId(1)
    );
    // Act
    var result = await _service.Handle(command);
    // Assert
    Assert.That(result, Is.Not.Null);
    Assert.That(result!.Name, Is.EqualTo("Test
    Campaign"));
    Assert.That(_dbContext.Campaigns.Count(), Is.EqualTo
    (1));
}

```

Test 1

```
[Test]
0 references
public void
Handle_CreateCampaignCommand_ShouldThrowException_WhenDup
licateName()
{
    // Arrange
    var goals = new List<Goal> { new Goal("Increase
        Sales", "Sales", 100, 0, 0) };
    var channels = new List<Channel> { new Channel
        ("Email", "email@example.com", 0) };

    var existingCampaign = new Campaign(
        "NombreDoble",
        "Description",
        DateTime.Now,
        DateTime.Now.AddDays(10),
        "Active",
        goals,
        channels,
        1,
        new CampaignUserId(1)
    );
    _dbContext.Campaigns.Add(existingCampaign);
    _dbContext.SaveChanges();

    var command = new CreateCampaignCommand(
        "NombreDoble",
        "Description",
        DateTime.Now,
        DateTime.Now.AddDays(10),
        "Active",
        goals,
        channels,
        1,
        new CampaignUserId(1)
    );

    // Act & Assert
    Assert.ThrowsAsync<Exception>(() => _service.Handle
        (command));
}
}
```

Test 2

Luego con **BovineCommandServiceTest**, se crearon otras dos, donde se pueden agregar Bovinos y probar si todo es correcto y de igual manera para validar la restriccion de agregar un Bovino a un establo lleno

Test 3

```
[Test]
0 references
public void Handle_CreateBovineCommand_ShouldAddBovine()
{
    // Arrange
    var stable = new Stable(new CreateStableCommand("Stable A", 10, new RanchUserId(1)));
    _dbContext.Stables.Add(stable);
    _dbContext.SaveChanges();
    var command = new CreateBovineCommand(
        "Test Bovine",
        "Male",
        DateTime.Now,
        "Breed",
        "Location",
        null,
        stable.Id,
        new RanchUserId(1),
        null
    );

    // Act
    var result = _service.Handle(command).Result;

    // Assert
    Assert.IsNotNull(result);
    Assert.AreEqual("Test Bovine", result.Name);      Dereference of a possibly null reference.
    Assert.AreEqual(1, _dbContext.Bovines.Count());
}
```

Test 4

```
[Test]
0 references
public void Handle_CreateBovineCommand_ShouldThrowException_WhenStableFull()
{
    // Arrange
    var stable = new Stable(new CreateStableCommand("Stable B", 2, new RanchUserId(1)));
    _dbContext.Stables.Add(stable);
    _dbContext.SaveChanges();

    var bovine1 = new Bovine(new CreateBovineCommand(
        "Bovine 1",
        "Male",
        DateTime.Now,
        "Breed",
        "Location",
        "https://example.com/image1.jpg",
        stable.Id,
        new RanchUserId(1),
        null
    ));
    var bovine2 = new Bovine(new CreateBovineCommand(
        "Bovine 2",
        "Female",
        DateTime.Now,
        "Breed",
        "Location",
        "https://example.com/image2.jpg",
        stable.Id,
        new RanchUserId(1),
        null
    ));

    _dbContext.Bovines.Add(bovine1);
    _dbContext.Bovines.Add(bovine2);
    _dbContext.SaveChanges();

    var command = new CreateBovineCommand(
        "Test Bovine",
        "Male",
        DateTime.Now,
        "Breed",
        "Location",
        "https://example.com/image3.jpg",
        stable.Id,
        new RanchUserId(1),
        null
    );

    // Act & Assert
    Assert.ThrowsAsync<Exception>(() => _service.Handle(command));
}
```

Luego de esto, se corrieron los Test, todos siendo correctos durante este proceso.

```
NUnit Adapter 4.6.0.0: Test execution complete
  VacApp-Bovinova-Platform prueba realizada correctamente (2.7s)

Resumen de pruebas: total: 4; con errores: 0; correcto: 4; omitido: 0; duración: 2.7 s
Compilación correcto con 34 advertencias en 7.2s
```

6.1.3. Core Behavior-Driven Development

Las pruebas **Behavior-Driven Development (BDD)** constituyen una metodología fundamental para validar el comportamiento del sistema **VacApp** desde la perspectiva del usuario final. Esta técnica utiliza un lenguaje natural y estructurado que facilita la comunicación entre desarrolladores, testers, product owners y otros stakeholders del proyecto.

Metodología BDD Implementada

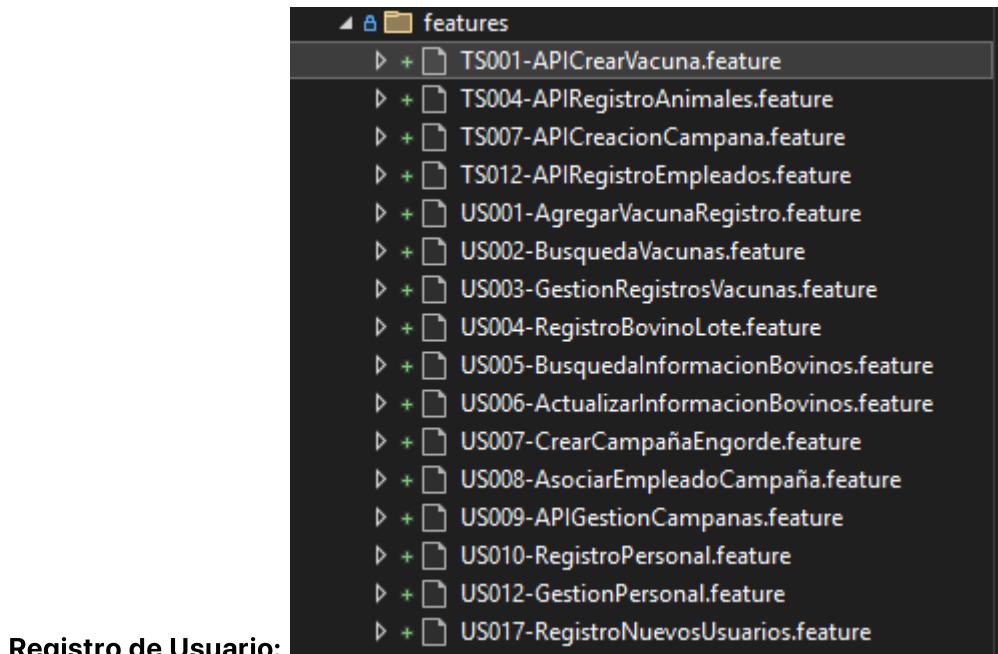
El enfoque BDD en VacApp se basa en la creación de **archivos .feature** escritos en lenguaje **Gherkin**, que describen escenarios específicos de uso utilizando la estructura **Given-When-Then**. Esta metodología permite:

- **Definir comportamientos esperados** en términos comprensibles para todos los miembros del equipo
- **Validar funcionalidades clave** del dominio ganadero de manera sistemática
- **Mantener comunicación clara** entre equipos técnicos y de negocio
- **Asegurar el cumplimiento** de los requisitos funcionales definidos

Escenarios de Prueba Documentados

A continuación se presentan los **archivos .feature** implementados para validar las funcionalidades principales de VacApp:

1. Gestión de Autenticación y Usuarios



Registro de Usuario:

Inicio de Sesión:

```

1  Feature: TS001 - API para Crear Vacuna
2    Como desarrollador, necesito exponer un endpoint para registrar una vacuna vía API para permitir la integración con sistemas externos y aplicaciones móviles
3
4    # Este feature se integra con el backend a través de:
5    # - Controller: RanchManagement/Interfaces/REST/VaccineController.cs (método POST /api/v1/vaccines)
6    # - Command Service: RanchManagement/Application/Internal/CommandServices/VaccineCommandService.cs
7    # - Repository: RanchManagement/Infrastructure/Persistence/EFC/Repositories/VaccineRepository.cs
8    # - Domain Model: RanchManagement/Domain/Model/Aggregates/Vaccine.cs
9    # - API Documentation: Swagger/OpenAPI specification
10
11 Scenario: Creación exitosa de vacuna vía API REST
12 Given que soy un desarrollador consumiendo el endpoint de vacunas
13 When realizo una petición POST al endpoint de creación de vacunas:
14   | Estructura      | <Valor>
15   | método          | POST
16   | endpoint         | /api/v1/vaccines
17   | contentType      | application/json
18   | headers          | Authorization: Bearer {token} |
19 And envío el siguiente JSON en el body:
20   ````json
21   {
22     "name": "Bovilis Bovino",
23     "type": "Antiviral",
24     "description": "Vacuna contra enfermedades virales bovinas",
25     "lot": "B123-2024",
26     "manufacturer": "Zoetis",
27     "manufacturingDate": "2024-01-15",
28     "expirationDate": "2025-12-31",
29     "availableDoses": 50,
30     "costPerDose": 25.50,
31     "storageTemperature": "2-8°C",
32     "administrationRoute": "Intramuscular",
33     "doseInterval": 21,
34     "minimumAge": 3,
35     "maximumAge": 120,
36     "withdrawalPeriod": 21,
37     "certifications": ["GMP", "ISO 9001"],
38     "batchNumber": "BATCH-001",
39     "qualityCertificate": "QC-2024-789"
40   }..
41
42 Then la API debería responder con código HTTP 201 (Created)
43 And el response body debería contener:
44   ````json
45   {
46     "id": 89,
47     "name": "Bovilis Bovino",
48     "type": "Antiviral",
49     "lot": "B123-2024"
50

```

2. Gestión de Bovinos y Ganado

Registro de Bovinos:

```

1  Feature: TS004 - API para Registro de Animales
2    Como desarrollador, quiero implementar un endpoint para registrar un bovino en un lote específico para permitir la integración con sistemas de gestión y aplicaciones móviles
3
4    # Este feature se integra con el backend a través de:
5    # - Controller: RanchManagement/Interfaces/REST/BovineController.cs (método POST /api/v1/bovines)
6    # - Command Service: RanchManagement/Application/Internal/CommandServices/BovineCommandService.cs
7    # - Repository: RanchManagement/Infrastructure/Persistence/EFC/Repositories/BovineRepository.cs
8    # - Domain Model: RanchManagement/Domain/Model/Aggregates/Bovine.cs
9    # - Stable Management: RanchManagement/Application/Internal/CommandServices/StableCommandService.cs
10
11 Scenario: Creación exitosa de bovino vía API REST
12 Given que soy un desarrollador consumiendo el endpoint de bovinos
13 When realizo una petición POST al endpoint de registro de bovinos:
14   | Estructura      | <Valor>
15   | método          | POST
16   | endpoint         | /api/v1/bovines
17   | contentType      | application/json
18   | headers          | Authorization: Bearer {token} |
19 And envío el siguiente JSON en el body:
20   ````json
21   {
22     "code": "A001",
23     "breed": "Angus",
24     "currentWeight": 350.5,
25     "birthDate": "2024-01-15",
26     "gender": "Male",
27     "color": "Black",
28     "motherId": "M001",
29     "fatherId": "P001",
30     "stableId": 1,
31     "healthStatus": "Healthy",
32     "lastVaccination": "2024-09-01",
33     "registrationDate": "2025-10-06",
34     "birthWeight": 35.2,
35     "birthType": "Natural",
36     "identificationType": "Visual Tag + RFID",
37     "tagNumber": "12345",
38     "rfidNumber": "985141001123456",
39     "originFarm": "La Primavera",
40     "notes": "Animal con excelente desarrollo"
41   }..
42
43 Then la API debería responder con código HTTP 201 (Created)
44 And el response body debería contener:
45   ````json
46   {
47     "id": 156,
48     "code": "A001",
49     "breed": "Angus",
50

```

Consulta de Información de Bovinos:

```

1 Feature: TS007 - API para Creación de Campaña
2   Como desarrollador, necesito crear un endpoint que permita la creación de campañas para permitir la integración con sistemas de gestión y aplicaciones móviles
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: CampaignManagement/Interfaces/REST/CampaignController.cs (método POST /api/v1/campaigns)
6   # - Command Service: CampaignManagement/Application/Internal/CommandServices/CampaignCommandService.cs
7   # - Repository: CampaignManagement/Infrastructure.Repositories/CampaignRepository.cs
8   # - Domain Model: CampaignManagement/Domain/Model/Aggregates/Campaign.cs
9   # - Commands: CreateCampaignCommand, AddGoalToCampaignCommand, AddChannelToCampaignCommand
10
11 Scenario: Creación exitosa de campaña completa vía API REST
12   Given que soy un desarrollador consumiendo el endpoint de campañas
13   When realizo una petición POST al endpoint de creación de campañas:
14     | Estructura      | <Valor>
15     | método          | POST
16     | endpoint         | /api/v1/campaigns
17     | contentType      | application/json
18     | headers          | Authorization: Bearer {token} |
19   And envío el siguiente JSON en el body:
20   ```json
21     {
22       "name": "Engorde Verano 2025",
23       "description": "Campaña de engorde intensivo para temporada alta",
24       "durationDays": 90,
25       "targetWeight": 500,
26       "dailyWeightGain": 2.0,
27       "assignedAnimals": 15,
28       "stableId": 1,
29       "budget": 15000.00,
30       "startDate": "2025-11-01",
31       "endDate": "2026-01-29",
32       "feedType": "Alta proteína",
33       "supplements": ["Vitaminas A, D, E", "Minerales Premium"],
34       "targetMarket": "Exportación Premium",
35       "requiredCertifications": ["HACCP", "Organic"],
36       "campaignType": "Fattening",
37       "priority": "High",
38       "responsibleUserId": 15,
39       "goals": [
40         {
41           "description": "Alcanzar peso objetivo",
42           "targetValue": 500,
43           "unitOfMeasure": "kg",
44           "targetDate": "2026-01-29"
45         },
46         {
47           "description": "Mantener conversión alimenticia",
48           "targetValue": 6.5,
49           "unitOfMeasure": "ratio".

```

Actualización de Datos de Bovinos:

```

1 Feature: TS012 - API para Registro de Empleados
2   Como desarrollador, necesito crear un endpoint para registrar empleados para permitir la integración con sistemas de recursos humanos y aplicaciones móviles
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: StaffAdministration/Interfaces/REST/StaffController.cs (método POST /api/v1/staff)
6   # - Command Service: StaffAdministration/Application/Internal/CommandServices/StaffCommandService.cs
7   # - Repository: StaffAdministration/Infrastructure/Persistence/EFC/Repositories/StaffRepository.cs
8   # - Domain Model: StaffAdministration/Domain/Model/Aggregates/Staff.cs
9
10 Scenario: Creación exitosa de empleado vía API REST
11   Given que soy un desarrollador consumiendo el endpoint de personal
12   When realizo una petición POST al endpoint de registro de empleados:
13     | Estructura      | <Valor>
14     | método          | POST
15     | endpoint         | /api/v1/staff
16     | contentType      | application/json
17     | headers          | Authorization: Bearer {token} |
18   And envío el siguiente JSON en el body:
19   ```json
20     {
21       "staffType": "Veterinarian",
22       "firstName": "Carlos Alberto",
23       "lastName": "Martinez Garcia",
24       "email": "carlos.martinez@vacapp.com",
25       "phone": "+51 987 654 321",
26       "dni": "87654321",
27       "professionalLicense": "VC-2024-0156",
28       "specialization": "Medicina Bovina",
29       "university": "Universidad Mayor de San Marcos",
30       "graduationYear": 2019,
31       "experience": 5,
32       "certifications": [
33         "Resolución SENASA-2021-089",
34         "Certificado en Cirugía Bovina"
35       ],
36       "status": "Active",
37       "hireDate": "2025-10-06",
38       "contractType": "FullTime",
39       "baseSalary": 4500.00,
40       "workSchedule": "Monday to Friday, 8:00 - 17:00",
41       "emergencyContact": {
42         "name": "Maria Martinez",
43         "phone": "+51 987 111 222",
44         "relationship": "Spouse"
45       },
46       "skills": [
47         "Surgery",
48         "Nutrition",
49         "Herd Management".

```

3. Administración de Establos

Creación de Establos:

```

1 Feature: US001 - Agregar Vacuna al Registro
2   Como ganadero, quiero agregar una nueva vacuna al registro de mis bovinos para mantener un control actualizado y efectivo de la salud de mi rebaño
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: RanchManagement/Interfaces/REST/VaccineController.cs
6   # - Command Service: RanchManagement/Application/Internal/CommandServices/VaccineCommandService.cs
7   # - Query Service: RanchManagement/Application/Internal/QueryServices/VaccineQueryService.cs
8   # - Repository: RanchManagement/Infrastructure/Persistence/EFC/Repositories/VaccineRepository.cs
9
10  Scenario: Registro exitoso de nueva vacuna antiviral
11    Given que soy un ganadero autenticado en el sistema con email "ganadero@vacapp.com"
12    And tengo acceso completo al módulo de gestión de vacunas
13    When ingreso una nueva vacuna con los siguientes datos:
14      | Estructura | <Ejemplo>
15      | nombre     | Bovilis Bovino
16      | tipo       | Antiviral
17      | descripcion | Vacuna contra enfermedades virales bovinas |
18      | lote        | B123-2024
19      | fabricante | Zoetis
20      | fechaVencimiento | 2025-12-31
21      | dosisDisponibles | 50
22      | costoPorDosis | 25.50
23      | temperaturaAlmacenamiento | 2-8°C
24      | viaAdministracion | Intramuscular
25      | intervaloDosis | 21 días
26    Then la vacuna debería registrarse exitosamente en la base de datos
27    And debería recibir confirmación "Vacuna registrada correctamente"
28    And la vacuna debería aparecer en el listado de vacunas disponibles
29    And el sistema debería generar un ID único para la vacuna
30    And se debería actualizar el inventario total de vacunas
31
32  Scenario: Registro exitoso de nueva vacuna bacteriana con fecha de vencimiento próxima
33    Given que soy un ganadero autenticado en el sistema con email "ganadero@vacapp.com"
34    And cuento con el rol de administrador de rancho
35    When ingreso una nueva vacuna con los siguientes datos:
36      | Estructura | <Ejemplo>
37      | nombre     | Pastobov
38      | tipo       | Bacteriana
39      | descripcion | Protección contra clostridios |
40      | lote        | P987-2024
41      | fabricante | Merck Animal Health
42      | fechaVencimiento | 2025-03-15
43      | dosisDisponibles | 30
44      | costoPorDosis | 18.75
45      | temperaturaAlmacenamiento | 4-8°C
46      | viaAdministracion | Subcutánea
47      | intervaloDosis | 30 días
48    Then la vacuna debería registrarse exitosamente con alerta de vencimiento próximo
49    And debería recibir confirmación "Vacuna registrada correctamente. Alerta: Vence en 90 días"

```

Gestión de Capacidad de Establos:

```

1 Feature: US002 - Búsqueda de vacunas
2   Como ganadero, quiero buscar vacunas previamente registradas para evitar la duplicación y garantizar que se administre la vacuna correcta a cada bovino
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: RanchManagement/Interfaces/REST/VaccineController.cs (métodos GET)
6   # - Query Service: RanchManagement/Application/Internal/QueryServices/VaccineQueryService.cs
7   # - Repository: RanchManagement/Infrastructure/Persistence/EFC/Repositories/VaccineRepository.cs
8   # - Domain Model: RanchManagement/Domain/Model/Aggregates/Vaccine.cs
9
10  Scenario: Búsqueda exitosa de vacuna por nombre exacto
11    Given que soy un ganadero autenticado en el sistema con email "ganadero@vacapp.com"
12    And tengo acceso al módulo de consulta de vacunas
13    And existen las siguientes vacunas registradas en el sistema:
14      | nombre     | tipo       | lote     | dosisDisponibles | fechaVencimiento |
15      | Bovilis Bovino | Antiviral | B123    | 50              | 2025-12-31
16      | Pastobov     | Bacteriana | P987    | 30              | 2025-03-15
17      | Vac-Protect® Plus | Polivalente | VP001   | 15              | 2025-09-15
18    When busco una vacuna por nombre exacto:
19      | Estructura | <Ejemplo>
20      | criterioBusqueda | nombre
21      | valor       | Bovilis Bovino
22    Then el sistema debería mostrar la vacuna encontrada:
23      | Estructura | <Ejemplo>
24      | nombre     | Bovilis Bovino
25      | tipo       | Antiviral
26      | lote        | B123
27      | dosisDisponibles | 50
28      | fechaVencimiento | 2025-12-31
29      | fabricante | Zoetis
30      | estado      | Disponible
31    And debería mostrar el historial de aplicación de esta vacuna
32    And debería mostrar las próximas fechas de aplicación recomendadas
33    And el tiempo de respuesta debería ser menor a 2 segundos
34
35  Scenario: Búsqueda de vacunas por tipo con múltiples resultados
36    Given que soy un usuario autenticado con rol de veterinario
37    And existen múltiples vacunas del mismo tipo registradas:
38      | nombre     | tipo       | lote     | dosisDisponibles | estado
39      | Bovilis Bovino | Antiviral | B123    | 50              | Disponible
40      | Viral-Bov     | Antiviral | V456    | 25              | Disponible
41      | Antivax-Pro    | Antiviral | A789    | 0               | Sin Stock
42      | Vira-Shield    | Antiviral | V012    | 40              | Disponible
43    When busco vacunas por tipo:
44      | Estructura | <Ejemplo>
45      | criterioBusqueda | tipo
46      | valor       | Antiviral
47      | ordenarResultados | por_fecha_vencimiento
48    Then el sistema debería mostrar todas las vacunas antivirales disponibles:
49      | Estructura | <Ejemplo>

```

4. Gestión de Campañas Sanitarias

Creación de Campañas:

```

1 Feature: US003 - Gestión de Registros de Vacunas
2   Como ganadero, necesito poder editar o eliminar el registro de una vacuna para garantizar que la información se mantenga precisa y actualizada
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: RanchManagement/Interfaces/REST/VaccineController.cs (métodos PUT, DELETE)
6   # - Command Service: RanchManagement/Application/Internal/CommandServices/VaccineCommandService.cs
7   # - Query Service: RanchManagement/Application/Internal/QueryServices/VaccineQueryService.cs
8   # - Repository: RanchManagement/Infrastructure/Persistence/EFC/Repositories/VaccineRepository.cs
9   # - Domain Model: RanchManagement/Domain/Model/Aggregates/Vaccine.cs
10
11 Scenario: Edición exitosa de información básica de vacuna
12   Given que soy un administrador de inventario autenticado con email "admin@vacapp.com"
13   And tengo permisos para modificar registros de vacunas
14   And existe la siguiente vacuna registrada en el sistema:
15   | Estructura | <Ejemplo>
16   | id         | 15
17   | nombre     | Bovilis Bovino
18   | tipo       | Antiviral
19   | lote        | B123-2024
20   | fabricante | Zoetis
21   | dosisDisponibles | 50
22   | fechaVencimiento | 2025-12-31
23   | costoPorDosis | 25.50
24   When actualizo la información de la vacuna con los siguientes cambios:
25   | Estructura | <NuevoValor>
26   | nombre     | Bovilis Bovino Plus
27   | costoPorDosis | 28.75
28   | temperaturaAlmacenamiento | 2-8°C
29   | viaAdministracion | Intramuscular profunda
30   | intervaloDosis | 21 días
31   Then la vacuna debería actualizarse exitosamente en la base de datos
32   And debería mantener el mismo ID y lote para trazabilidad
33   And debería generar un registro de auditoria:
34   | Estructura | <Ejemplo>
35   | fechaModificacion | 2025-10-06 14:30:15
36   | usuarioModificacion | admin@vacapp.com
37   | campoModificado | nombre, costo, temperatura, via, intervalo |
38   | valorAnterior | Bovilis Bovino, 25.50, 4°C, IM, 21 días |
39   | valorNuevo | Bovilis Bovino Plus, 28.75, 2-8°C, IM profunda, 21 días |
40   And debería mostrarse confirmación "Vacuna actualizada correctamente"
41   And el historial de cambios debería permanecer accesible
42
43 Scenario: Actualización de stock disponible tras aplicación de vacunas
44   Given que soy un veterinario aplicando vacunas en campaña
45   And tengo una vacuna con stock actual:
46   | Estructura | <Ejemplo>
47   | id         | 18
48   | nombre     | Pastobov
49   | dosisDisponibles | 45

```

Seguimiento de Campañas:

```

1 Feature: US004 - Registro de Bovino en Lote
2   Como usuario autenticado, quiero registrar un bovino en un lote específico para tener control detallado de la crianza y manejo del animal
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: RanchManagement/Interfaces/REST/BovineController.cs
6   # - Command Service: RanchManagement/Application/Internal/CommandServices/BovineCommandService.cs
7   # - Query Service: RanchManagement/Application/Internal/QueryServices/BovineQueryService.cs
8   # - Repository: RanchManagement/Infrastructure/Persistence/EFC/Repositories/BovineRepository.cs
9   # - Stable Service: RanchManagement/Application/Internal/CommandServices/StableCommandService.cs
10
11 Scenario: Registro exitoso de bovino en establo con capacidad disponible
12   Given que soy un usuario autenticado en el sistema con email "administrador@vacapp.com"
13   And tengo el rol de administrador de rancho
14   And tengo el establo "Establo-01" con las siguientes características:
15   | Estructura | <Ejemplo>
16   | nombre     | Establo Principal
17   | capacidadMaxima | 50
18   | animalesActuales | 35
19   | estado      | Activo
20   When registro un nuevo bovino con los siguientes datos:
21   | Estructura | <Ejemplo>
22   | codigo     | A001
23   | raza       | Angus
24   | pesoActual | 350.5
25   | fechaNacimiento | 2024-01-15
26   | sexo       | Macho
27   | color      | Negro
28   | madreId    | M001
29   | padreId    | P001
30   | establoId  | 1
31   | estadioSalud | Saludable
32   | ultimaVacunacion | 2024-09-01
33   Then el bovino debería registrarse exitosamente en la base de datos
34   And debería asociarse automáticamente al establo "Establo-01"
35   And el bovino debería aparecer en el listado de animales del establo
36   And el contador de animales del establo debería actualizarse a 36
37   And debería generarse un historial médico inicial para el bovino
38   And el sistema debería registrar la fecha y hora de registro
39
40 Scenario: Registro exitoso de hembra para reproducción
41   Given que soy un usuario autenticado con rol de administrador
42   And tengo el establo "Establo-Reproducción" disponible
43   When registro una hembra con potencial reproductivo:
44   | Estructura | <Ejemplo>
45   | codigo     | H002
46   | raza       | Hereford
47   | pesoActual | 420.0
48   | fechaNacimiento | 2023-06-20
49   | sexo       | Hembra

```

5. Control de Vacunas y Tratamientos

Registro de Vacunas:

```

1 Feature: US005 - Buscar información de Bovinos
2   Como usuario, quiero poder buscar animales registrados para acceder de forma rápida y ordenada a la información necesaria
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: RanchManagement/Interfaces/REST/BovineController.cs (métodos GET)
6   # - Query Service: RanchManagement/Application/Internal/QueryServices/BovineQueryService.cs
7   # - Repository: RanchManagement/Infrastructure/Persistence/EFC/Repositories/BovineRepository.cs
8   # - Domain Model: RanchManagement/Domain/Model/Aggregates/Bovine.cs
9   # - Stable Management: RanchManagement/Application/Internal/QueryServices/StableQueryService.cs
10
11 Scenario: Búsqueda exitosa de bovino por código exacto
12 Given que soy un usuario autenticado en el sistema con email "administrador@vacapp.com"
13 And tengo acceso al módulo de consulta de ganado
14 And existe el siguiente bovino registrado en el sistema:
15   | Estructura | <Ejemplo>
16   | codigo     | A001
17   | raza       | Angus
18   | pesoActual | 350.5
19   | fechaNacimiento | 2024-01-15
20   | sexo        | Macho
21   | establoActual | Establo-01
22   | estadoSalud | Saludable
23 When busco un bovino por su código identificador:
24   | Estructura | <Ejemplo>
25   | criterioBusqueda | codigo
26   | valor       | A001
27 Then el sistema debería mostrar la información completa del bovino:
28   | Estructura | <Ejemplo>
29   | codigo     | A001
30   | raza       | Angus
31   | pesoActual | 350.5 kg
32   | fechaNacimiento | 2024-01-15 (8 meses)
33   | sexo        | Macho
34   | color      | Negro
35   | madreId    | M001
36   | padreId    | P001
37   | establoActual | Establo-01
38   | estadoSalud | Saludable
39   | ultimaVacunacion | 2024-09-01
40 And debería mostrar su historial médico completo
41 And debería mostrar su árbol genealógico si está disponible
42 And el tiempo de respuesta debería ser menor a 1.5 segundos
43
44 Scenario: Búsqueda de bovinos por raza con múltiples resultados
45 Given que soy un ganadero autenticado con rol de administrador
46 And existen múltiples bovinos de la misma raza registrados:
47   | codigo | raza | peso | establo | estadoSalud |
48   | A001  | Angus | 350.5 | Establo-01 | Saludable
49   | A002  | Angus | 380.2 | Establo-01 | Saludable

```

Programación de Tratamientos:

```

1 Feature: US006 - Actualizar información de Bovinos
2   Como usuario, quiero gestionar la información de los animales registrados para mantener la base de datos actualizada y precisa
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: RanchManagement/Interfaces/REST/BovineController.cs (métodos PUT, DELETE)
6   # - Command Service: RanchManagement/Application/Internal/CommandServices/BovineCommandService.cs
7   # - Query Service: RanchManagement/Application/Internal/QueryServices/BovineQueryService.cs
8   # - Repository: RanchManagement/Infrastructure/Persistence/EFC/Repositories/BovineRepository.cs
9   # - Domain Model: RanchManagement/Domain/Model/Aggregates/Bovine.cs
10  # - Stable Management: RanchManagement/Application/Internal/CommandServices/StableCommandService.cs
11
12 Scenario: Actualización exitosa de peso y estado de salud
13 Given que soy un veterinario autenticado con email "veterinario@vacapp.com"
14 And tengo permisos para actualizar información médica de animales
15 And existe el siguiente bovino registrado en el sistema:
16   | Estructura | <Ejemplo>
17   | codigo     | A001
18   | raza       | Angus
19   | pesoActual | 350.5
20   | fechaNacimiento | 2024-01-15
21   | establoActual | Establo-01
22   | estadoSalud | Saludable
23   | ultimaRevision | 2024-09-01
24 When actualizo la información del bovino después de una revisión:
25   | Estructura | <nuevoValor>
26   | pesoActual | 378.2
27   | estadoSalud | Saludable
28   | condicionCorporal | 3.5 (Excelente)
29   | ultimaRevision | 2025-10-06
30   | responsableRevision | Dr. García
31   | observaciones | Ganancia de peso óptima, buen desarrollo |
32 Then la información del bovino debería actualizarse exitosamente
33 And debería registrarse el histórico de crecimiento:
34   | Estructura | <Ejemplo>
35   | fechaRegistro | 2025-10-06
36   | pesoAnterior | 350.5
37   | pesoNuevo | 378.2
38   | ganancia | 27.7 kg
39   | diasTranscurridos | 35
40   | gananciaDiaria | 0.79 kg/día
41 And debería generarse una alerta si la ganancia está fuera del rango esperado
42 And debería actualizarse la proyección de peso objetivo
43 And debería registrarse la revisión en el histórico médico
44
45 Scenario: Traslado de bovino entre establos
46 Given que soy un administrador de establos autenticado
47 And tengo un bovino que necesita ser trasladado:
48   | Estructura | <Ejemplo>
49   | codigo     | B002

```

6. Administración de Personal

Gestión de Staff:

```

1 Feature: US007 - Crear Campaña para Engorde de Ganado
2   Como usuario de la plataforma, quiero crear una campaña para engordar el ganado asignado, definiendo parámetros como duración, objetivo y selección de animales o establos
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: CampaignManagement/Interfaces/REST/CampaignController.cs
6   # - Command Service: CampaignManagement/Application/Internal/CommandServices/CampaignCommandService.cs
7   # - Query Service: CampaignManagement/Application/Internal/QueryServices/CampaignQueryService.cs
8   # - Repository: CampaignManagement/Infrastructure/Repositories/CampaignRepository.cs
9   # - Goal Management: CampaignManagement/Domain/Model/Aggregates/Goal.cs
10  # - Channel Management: CampaignManagement/Domain/Model/Aggregates/Channel.cs
11
12  Scenario: Creación exitosa de campaña de engorde con metas específicas
13  Given que soy un usuario autenticado con email "administrador@vacapp.com"
14  And tengo el rol de administrador de campañas
15  And tengo animales disponibles con los siguientes datos:
16    | Estructura | <Ejemplo>
17    | totalAnimales | 25
18    | pesoPromedio | 320.5
19    | estadoSalud | Saludable
20    | estableDisponible | Estable-01
21  When creo una nueva campaña con los siguientes parámetros:
22    | Estructura | <Ejemplo>
23    | nombre | Engorde Verano 2025
24    | descripción | Campaña de engorde intensivo para temporada alta |
25    | duracionDias | 90
26    | objetivoPeso | 500
27    | objetivoGananciaDiaria | 2.0
28    | animalesAsignados | 15
29    | estableId | 1
30    | presupuesto | 15000.00
31    | fechaInicio | 2025-11-01
32    | fechaFin | 2026-01-29
33    | tipoAlimentacion | Alta proteína
34    | suplementos | Vitaminas A, D, E
35  Then la campaña debería crearse exitosamente en el sistema
36  And debería tener estado "Activa"
37  And los 15 animales seleccionados deberían asociarse a la campaña
38  And debería crearse un calendario de monitoreo automático
39  And el presupuesto debería registrarse y dividirse por período
40  And deberían establecerse metas semanales de peso
41  And el estable debería marcarse como "Ocupado por campaña"
42  And debería generarse un ID único de campaña
43
44  Scenario: Creación de campaña para temporada específica con objetivos múltiples
45  Given que soy un usuario autenticado con rol de gestor de producción
46  And tenemos la temporada "Navidad 2025" con alta demanda
47  And contamos con 30 animales listos para engorde rápido
48  When creo una campaña con objetivos específicos de temporada:
49    | Estructura | <Ejemplo>

```

Asignación de Roles:

```

1 Feature: US008 - Asociar Empleado a Campaña
2   Como administrador del sistema, quiero asociar empleados a campañas específicas para asegurar la correcta asignación de responsabilidades y seguimiento de las actividades de producción
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: CampaignManagement/Interfaces/REST/CampaignController.cs (AddChannelToCampaign, AddGoalToCampaign)
6   # - Command Service: CampaignManagement/Application/Internal/CommandServices/CampaignCommandService.cs
7   # - Query Service: CampaignManagement/Application/Internal/QueryServices/CampaignQueryService.cs
8   # - Repository: CampaignManagement/Infrastructure/Repositories/CampaignRepository.cs
9   # - Staff Management: StaffAdministration/Application/Internal/CommandServices/StaffCommandService.cs
10  # - Staff Query: StaffAdministration/Application/Internal/QueryServices/StaffQueryService.cs
11
12  Scenario: Asociar veterinario como responsable de campaña de engorde
13  Given que soy un administrador autenticado con email "admin@vacapp.com"
14  And tengo permisos para gestionar campañas y personal
15  And existe una campaña activa con los siguientes datos:
16    | Estructura | <Ejemplo>
17    | nombre | Engorde Verano 2025
18    | estado | Activa
19    | animalesAsignados | 25
20    | fechaInicio | 2025-10-01
21    | duracionDias | 90
22    | estableAsignado | Estable-01
23  And existe un veterinario disponible con los siguientes datos:
24    | Estructura | <Ejemplo>
25    | nombre | Dr. Carlos Martínez
26    | email | carlos.martinez@vacapp.com |
27    | rol | Veterinario
28    | especializacion | Nutrición Bovina
29    | experiencia | 8 años
30    | numeroColegiatura | VC-2024-0156
31    | estado | Disponible
32  When asocio al veterinario "Dr. Carlos Martínez" como responsable médico de la campaña
33  Then el empleado debería asignarse exitosamente a la campaña
34  And debería aparecer en la lista de personal asignado:
35    | Estructura | <Ejemplo>
36    | nombreCompleto | Dr. Carlos Martínez
37    | rolEnCampaña | Veterinario Principal
38    | fechaAsignacion | 2025-10-06
39    | estadoAsignacion | Activo
40    | responsabilidades | Salud y bienestar animal
41  And el veterinario debería recibir notificación por email
42  And debería registrarse la asignación en el histórico de la campaña
43  And el veterinario debería obtener acceso a los informes médicos de la campaña
44  And deberían generarse recordatorios de revisión médica periódica
45
46  Scenario: Asociar supervisor de producción a campaña existente
47  Given que soy un gerente de producción autenticado
48  And existe una campaña "Engorde Premium Navidad" en ejecución:
49    | Estructura | <Ejemplo>

```

7. Reportes y Analytics

Generación de Reportes:

```

1 Feature: US009 - API para Gestión de Campañas
2   Como administrador del sistema, quiero implementar endpoints que permitan editar y eliminar campañas para mantener actualizadas las estrategias de producción
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: CampaignManagement/Interfaces/REST/CampaignController.cs (métodos PUT, DELETE)
6   # - Command Service: CampaignManagement/Application/Internal/CommandServices/CampaignCommandService.cs
7   # - Query Service: CampaignManagement/Application/Internal/QueryServices/CampaignQueryService.cs
8   # - Repository: CampaignManagement/Infrastructure/Repositories/CampaignRepository.cs
9   # - Domain Model: CampaignManagement/Domain/Model/Aggregates/Campaign.cs
10  # - Commands: UpdateCampaignStatusCommand, DeleteCampaignCommand
11
12  Scenario: Actualización exitosa de parámetros de campaña activa
13    Given que soy un administrador de campañas autenticado con email "admin@vacapp.com"
14    And tengo permisos para modificar campañas en ejecución
15    And existe la siguiente campaña activa en el sistema:
16      | Estructura | <Ejemplo>
17      | campaignId | 25
18      | nombre | Engorde Verano 2025
19      | estado | Activa
20      | animalesAsignados | 25
21      | duracionDias | 90
22      | objetivoPeso | 500
23      | presupuesto | 45000.00
24      | fechaInicio | 2025-10-01
25      | fechaFin | 2026-01-29
26    When actualizo los parámetros de la campaña:
27      | Estructura | <NuevoValor>
28      | nombre | Engorde Verano Premium 2025 |
29      | descripción | Campaña optimizada con suplementos premium |
30      | duracionDias | 105
31      | objetivoPeso | 520
32      | presupuesto | 52000.00
33      | suplementos | Vitaminas A,D,E + Minerales Premium |
34      | tipoAlimentacion | Alta calidad Premium
35    Then la campaña debería actualizarse exitosamente
36    And debería generarse un registro de auditoria:
37      | Estructura | <Ejemplo>
38      | fechaModificacion | 2025-10-06 14:30:15
39      | responsableModificacion | admin@vacapp.com
40      | parametrosModificados | nombre, duración, objetivo peso, presupuesto, suplementos |
41      | valorAnteriorDuracion | 90 días
42      | valorNuevoDuracion | 105 días
43      | aumentoPresupuesto | 7000.00
44    And debería recalcularse automáticamente la fecha de fin:
45      | Estructura | <Ejemplo>
46      | fechaFinAnterior | 2026-01-29
47      | nuevaFechaFin | 2026-02-13
48      | diasAdicionales | 15
49    And deberían notificarse todos los empleados asignados

```

Análisis de Productividad:

```

1 Feature: US010 - Registro de Personal
2   Como administrador del sistema, quiero registrar personal calificado para tener disponibles los recursos humanos necesarios en la gestión del rancho
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: StaffAdministration/Interfaces/REST/StaffController.cs (métodos POST)
6   # - Command Service: StaffAdministration/Application/Internal/CommandServices/StaffCommandService.cs
7   # - Query Service: StaffAdministration/Application/Internal/QueryServices/StaffQueryService.cs
8   # - Repository: StaffAdministration/Infrastructure/Persistence/EFC/Repositories/StaffRepository.cs
9   # - Domain Model: StaffAdministration/Domain/Model/Aggregates/Staff.cs
10
11  Scenario: Registro exitoso de veterinario con todas las credenciales
12    Given que soy un administrador de recursos humanos autenticado con email "rh@vacapp.com"
13    And tengo permisos para registrar personal médico
14    When registro un nuevo veterinario con los siguientes datos:
15      | Estructura | <Ejemplo>
16      | tipoPersonal | Veterinario
17      | nombres | Carlos Alberto
18      | apellidos | Martinez Garcia
19      | email | carlos.martinez@vacapp.com
20      | telefono | +51 987 654 321
21      | dni | 87654321
22      | numeroColegiatura | VC-2024-0156
23      | especializacion | Medicina Bovina
24      | universidad | Universidad Mayor de San Marcos
25      | añoGraduacion | 2019
26      | experiencia | 5 años
27      | certificaciones | Resolución SENASA-2021-089
28      | estado | Activo
29      | fechaContratacion | 2025-10-06
30      | tipoContrato | Tiempo completo
31      | sueldoBase | 4500.00
32    Then el veterinario debería registrarse exitosamente en el sistema
33    And debería generarse un código único de empleado:
34      | Estructura | <Ejemplo>
35      | codigoEmpleado | VET-2025-001
36      | estado | Activo
37      | fechaIngreso | 2025-10-06
38    And debería crearse su perfil de usuario con rol "veterinario"
39    And debería enviarse un email de bienvenida con credenciales de acceso
40    And deberían verificarse automáticamente sus credenciales profesionales
41    And debería generarse un contrato digital con sus datos
42    And debería archivarse su documentación de forma segura
43
44  Scenario: Registro de operario de rancho con habilidades específicas
45    Given que soy un administrador de operaciones autenticado
46    And necesito registrar personal para manejo de ganado
47    When registro un operario con el siguiente perfil:
48      | Estructura | <Ejemplo>
49      | tipoPersonal | Operario

```

8. Funcionalidades Avanzadas

Integración de Datos:

```

1 Feature: US012 - Gestión de Personal
2   Como administrador del sistema, quiero administrar el personal registrado para mantener actualizada la información y gestionar su estatus laboral
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: StaffAdministration/Interfaces/REST/StaffController.cs (métodos PUT, DELETE)
6   # - Command Service: StaffAdministration/Application/Internal/CommandServices/StaffCommandService.cs
7   # - Query Service: StaffAdministration/Application/Internal/QueryServices/StaffQueryService.cs
8   # - Repository: StaffAdministration/Infrastructure/Persistence/EFC/Repositories/StaffRepository.cs
9   # - Domain Model: StaffAdministration/Domain/Model/Aggregates/Staff.cs
10
11 Scenario: Actualización exitosa de datos de empleado
12   Given que soy un administrador de RRHH autenticado con email "rh@vacapp.com"
13   And tengo permisos para modificar información de personal
14   And existe el siguiente empleado registrado en el sistema:
15     | Estructura | <Ejemplo>
16     | codigoEmpleado | VET-2025-001
17     | nombres | Carlos Alberto
18     | apellidos | Martinez Garcia
19     | email | carlos.martinez@vacapp.com |
20     | telefono | +51 987 654 321
21     | especializacion | Medicina Bovina
22     | sueldoBase | 4500.00
23     | estado | Activo
24   When actualizo la información del empleado con los siguientes cambios:
25     | Estructura | <NuevoValor>
26     | telefono | +51 987 111 222
27     | emailPersonal | carlos.martinez.personal@email.com |
28     | direccionActualizada | Av. Ganaderos #456, Urbanización El Prado |
29     | especializacion | Medicina Bovina + Cirugía |
30     | sueldoBase | 4800.00
31     | certificacionNueva | Diplomado en Cirugía Bovina (2025) |
32   Then la información del empleado debería actualizarse exitosamente
33   And debería generarse un registro de auditoria:
34     | Estructura | <Ejemplo>
35     | fechaModificacion | 2025-10-06 10:30:15
36     | responsableModificacion | rh@vacapp.com
37     | camposModificados | telefono, emailPersonal, dirección, especialización, sueldo, certificación |
38     | valorAnteriorSueldo | 4500.00
39     | valorNuevoSueldo | 4800.00
40     | motivoCambio | Promoción y especialización adicional |
41   And debería enviarse una notificación al empleado sobre los cambios
42   And el sueldo debería aplicarse desde el próximo periodo de pago
43   And la nueva certificación debería añadirse a su perfil profesional
44
45 Scenario: Promoción de empleado con cambio de rol
46   Given que soy un gerente de operaciones
47   And tengo un operario con excelente desempeño:
48     | Estructura | <Ejemplo>
49     | codigoEmpleado | OP-2025-015
  
```

Validación de Reglas de Negocio:

```

1 Feature: US011 - Registro de Nuevos Usuarios
2   Como productor ganadero, quiero registrarme en la plataforma VacApp para acceder a las herramientas de gestión de mi rancho
3
4   # Este feature se integra con el backend a través de:
5   # - Controller: IAM/Interfaces/REST/UserController.cs (métodos POST)
6   # - Command Service: IAM/Application/CommandServices/UserCommandService.cs
7   # - Query Service: IAM/Application/QueryServices/UserQueryService.cs
8   # - Repository: IAM/Infrastructure/Repositories/UserRepository.cs
9   # - Domain Model: IAM/Domain/Model/Aggregates/User.cs
10  # - Authentication: IAM/Infrastructure/Tokens/JWT/Services/TokenService.cs
11  # - Hashing: IAM/Infrastructure/Hashing/BCrypt/Services/HashingService.cs
12
13 Scenario: Registro exitoso de nuevo usuario productor ganadero
14   Given que soy un nuevo productor ganadero interesado en VacApp
15   And no tengo una cuenta registrada previamente
16   When completo el formulario de registro con mis datos:
17     | Estructura | <Ejemplo>
18     | nombres | Carlos Alberto
19     | apellidos | Mendoza Paredes
20     | email | carlos.mendoza@vacapp.com |
21     | password | Ganado2024*
22     | confirmPassword | Ganado2024*
23     | telefono | +51 987 654 321
24     | dni | 87654321
25     | tipoUsuario | Productor Ganadero
26     | nombreRancho | El Triunfo S.A.
27     | direccionRancho | Km. 15 Carretera Central, Huaral |
28     | departamento | Lima
29     | provincia | Huaral
30     | distrito | Huaral
31     | tamanoRancho | 50 hectáreas
32     | animalesTotal | 150
33     | tipoProduccion | Carne
34     | razaPrincipal | Angus
35     | experiencia | 10 años
36     | comoConocio | Recomendación de colega |
37   Then mi cuenta debería crearse exitosamente en el sistema
38   And debería generarse un perfil de usuario con los siguientes datos:
39     | Estructura | <Ejemplo>
40     | userId | USER-2025-089
41     | username | carlos.mendoza@vacapp.com |
42     | estado | Pendiente de verificación |
43     | fechaRegistro | 2025-10-06 14:30:15 |
44     | rolAsignado | Productor
45   And debería enviarse un email de verificación a mi correo
46   And debería recibir instrucciones para activar mi cuenta
47   And mis datos del rancho deberían registrarse para perfilamiento
48   And debería generarse un historial de actividad inicial
49
  
```

Beneficios de la Implementación BDD

La adopción de BDD en VacApp ha proporcionado los siguientes beneficios:

Comunicación mejorada entre equipos técnicos y de negocio **Documentación viva** que se mantiene actualizada con el código **Validación automática** de comportamientos críticos del sistema **Reducción de defectos** mediante especificaciones claras **Facilita el mantenimiento** y evolución del software **Alineación continua** con las necesidades del dominio ganadero

Herramientas Utilizadas

- **Gherkin:** Lenguaje para escribir especificaciones legibles
- **SpecFlow:** Framework para automatización de pruebas BDD en .NET
- **Cucumber:** Herramienta complementaria para ejecución de escenarios
- **Visual Studio:** IDE para desarrollo e integración de pruebas

6.1.4. Core System Tests

En esta sección se documentan las **pruebas del sistema central (Core System Tests)** realizadas mediante la colección **VacAppTestLocal** en **Postman**, que agrupa todos los módulos críticos del backend de **VacApp**, incluyendo autenticación, gestión ganadera y administración de personal.

Estas pruebas permiten garantizar el correcto funcionamiento de la API, validando las operaciones CRUD principales, los flujos de autenticación JWT y la gestión integral de entidades ganaderas (bovinos, campañas, establos, personal, vacunas y administradores).

Estructura de la Colección

La colección está organizada por módulos, simulando el flujo real de interacción de un usuario dentro del ecosistema VacApp.

Sign-up — Registro exitoso

Prueba el registro de un usuario nuevo en el sistema.

Request: POST {{baseUrl}}/api/v1/User/sign-up

Descripción: Se valida la creación exitosa del usuario y la obtención del **token**.

The screenshot shows the Postman interface with the following details:

- Collection:** My Workspace
- Environment:** VacAppTestLocal
- Request Type:** POST
- URL:** {{baseUrl}}/api/v1/User/sign-up
- Body (JSON):**

```

1 {
2   "username": "{{username_ok}}",
3   "email": "{{email_ok}}",
4   "password": "{{password_ok}}"
5 }
```

- Response Status:** 201 Created
- Response Body:**

```

1 {
2   "token": "eyJhbGciOiJodHRwOi8vd3dLnczLm9yZy8yMDAxLzA0L3htbGRzaWctbW9yZSNobWFjLXNoYTI1NiIsInR5cIi6IkpxVCJ9.
eyJleHAiOiJE3NjAzOTI4OTUsImhoDHA6Ly9zY2h1lWFLnhbHNvYXAb3JnL3dzLzIwMDUvMDUvaR1bnRpdkvY2khaW1zL3npZC16IjU1LCJodHRwO18vc
c2N0ZW1hcyc54bNxzb2FwLm9yZy93cy8yMDA1LzA1L21kZW50aXR5L2NsY1ltcy9uY11IjoiVXNlcjEwNzcvIiwiawFOijoxNzU5Nzg4MDk1LCJuYmYj0jE3
NTk3ODgwTV9.vwxec7yz5CqSi8LTB2kmhmLkgqQihHj9wkt-fRYyZvg",
3   "userName": "User10770",
4   "email": "user10770@vacapp.com"
5 }
```

The screenshot shows the Postman interface with the following details:

- Collection:** VacAppTestLocal
- Environment:** VacAppTestLocal
- Request Type:** POST
- URL:** {{baseUrl}}/api/v1/User/sign-up
- Body (JSON):**

```

1 {
2   "username": "{{username_ok}}",
3   "email": "{{email_ok}}",
4   "password": "{{password_ok}}"
5 }
```

- Response Status:** 201 Created
- Response Body:**

```

1 {
2   "token": "eyJhbGciOiJodHRwOi8vd3dLnczLm9yZy8yMDAxLzA0L3htbGRzaWctbW9yZSNobWFjLXNoYTI1NiIsInR5cIi6IkpxVCJ9.
eyJleHAiOiJE3NjAzOTI4OTUsImhoDHA6Ly9zY2h1lWFLnhbHNvYXAb3JnL3dzLzIwMDUvMDUvaR1bnRpdkvY2khaW1zL3npZC16IjU1LCJodHRwO18vc
c2N0ZW1hcyc54bNxzb2FwLm9yZy93cy8yMDA1LzA1L21kZW50aXR5L2NsY1ltcy9uY11IjoiVXNlcjEwNzcvIiwiawFOijoxNzU5Nzg4MDk1LCJuYmYj0jE3
NTk3ODgwTV9.vwxec7yz5CqSi8LTB2kmhmLkgqQihHj9wkt-fRYyZvg",
3   "userName": "User10770",
4   "email": "user10770@vacapp.com"
5 }
```

Sign-up — Usuario ya existe

Valida que el sistema no permita registrar dos veces el mismo correo.

Request: POST {{baseUrl}}/api/v1/User/sign-up

Descripción: La API retorna un error controlado informando que el usuario ya está registrado.

HTTP VacAppTestLocal / Sign-Up — Usuario ya existe

POST {{baseUrl}}/api/v1/User/sign-up

Body (raw) JSON

```

1 {
2   "username": "{{username_duplicate}}",
3   "email": "{{email_duplicate}}",
4   "password": "{{password_duplicate}}"
5 }
```

Test Results (2/2)

- PASSED | Usuario duplicado detectado correctamente
- PASSED | La excepción es InvalidOperationException

HTTP VacAppTestLocal / Sign-Up — Usuario ya existe

POST {{baseUrl}}/api/v1/User/sign-up

Body (raw) JSON

```

1 System.InvalidOperationException: User already exists
2   at VacApp_Bovinova_Platform.IAM.Application.CommandServices.UserCommandService.Handle(SignUpCommand command) in /Users/maycolrojas/Documents/GitHub/Bacckend-VacApp/VacApp-Bovinova-Platform/IAM/Application/CommandServices/UserCommandService.cs:line 31
3   at VacApp_Bovinova_Platform.IAM.Interfaces.RESTUserController.SignUp(SignUpResource resource) in /Users/maycolrojas/Documents/GitHub/Bacckend-VacApp/VacApp-Bovinova-Platform/IAM/Interfaces/REST/UserController.cs:line 38
4   at Microsoft.AspNetCore.Mvc.Infrastructure.ActionMethodExecutor.TaskOfIAActionResultExecutor.Execute(ActionContext actionContext, IActionResultTypeMapper mapper, ObjectMethodExecutor executor, Object controller, Object[] arguments)
5   at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeActionMethodAsync>g__Logged|12_1(ControllerActionInvoker invoker)
6   at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeNextActionFilterAsync>g__Awaited|10_0(ControllerActionInvoker invoker, Task lastTask, State next, Scope scope, Object state, Boolean isCompleted)
```

Verifica que el backend rechace direcciones de correo con formato incorrecto.

Request: POST {{baseUrl}}/api/v1/User/sign-up

Descripción: Retorna error 400 indicando formato de email inválido.

The screenshot shows the Postman interface with the following details:

- Collection:** My Workspace
- Environment:** VacAppTestLocal
- Request:** POST {{baseUrl}}/api/v1/User/sign-up
- Pre-request Script:**

```
1 pm.environment.set("username_invalid", "UserInvalid");
2 pm.environment.set("email_invalid", "correoSinFormato");
3 pm.environment.set("password_invalid", "P@ssw0rd!");
```
- Test Results:** PASSED - Debe rechazar email inválido con status 500 y mensaje de error de email
- Status:** 500 Internal Server Error

The screenshot shows the Postman interface with the following details:

- Collection:** My Workspace
- Environment:** VacAppTestLocal
- Request:** POST {{baseUrl}}/api/v1/User/sign-up
- Pre-request Script:**

```
1 pm.environment.set("username_invalid", "UserInvalid");
2 pm.environment.set("email_invalid", "correoSinFormato");
3 pm.environment.set("password_invalid", "P@ssw0rd!");
```
- Test Results:** Failed - System.ArgumentException: The provided email format is invalid. Please enter a valid email address. (Parameter 'Email')
- Error Stack Trace:**

```
1 System.ArgumentException: The provided email format is invalid. Please enter a valid email address. (Parameter 'Email')
2   at VacApp_Bovinova_Platform.IAM.Application.CommandServices.UserCommandService.Handle(SignUpCommand command) in /Users/maycolrojas/Documents/GitHub/Bacckend-VacApp/VacApp-Bovinova-Platform/IAM/Application/CommandServices/UserCommandService.cs:line 40
3   at VacApp_Bovinova_Platform.IAM.Interfaces.REST.UserController.SignUp(SignUpResource resource) in /Users/maycolrojas/Documents/GitHub/Bacckend-VacApp/VacApp-Bovinova-Platform/IAM/Interfaces/REST/UserController.cs:line 38
4   at Microsoft.AspNetCore.Mvc.Infrastructure.ActionMethodExecutor.TaskOfActionResultExecutor.Execute(ActionContext actionContext, IActionResultTypeMapper mapper, ObjectMethodExecutor executor, Object controller, Object[] arguments)
5   at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeActionMethodAsync>d__12_1.MoveNext()
6   at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeNextActionFilterAsync>d__10_0.MoveNext()
```

Sign-in — Inicio exitoso

Valida la autenticación correcta y generación del token JWT.

Request: POST {{baseUrl}}/api/v1/User/sign-in

Descripción: Retorna el accessToken y los datos del usuario autenticado.

The screenshot shows the Postman interface with the following details:

- Collection:** My Workspace
- Environment:** VacAppTestLocal
- Request:** POST {{baseUrl}} /api/v1/User/sign-in
- Body:** Raw JSON (selected)
- JSON Body:**

```
1 {
2   "email": "{{email_ok}}",
3   "userName": "",
4   "password": "{{password_ok}}"
5 }
```

- Test Results (2/2):**
 - PASSED: El código de estado HTTP es 200
 - PASSED: La respuesta contiene un token de autenticación válido
- Response Headers:** 200 OK, 152 ms, 607 B

The screenshot shows the Postman interface with the following details:

- Collection:** My Workspace
- Environment:** VacAppTestLocal
- Request:** POST {{baseUrl}} /api/v1/User/sign-in
- Body:** Raw JSON (selected)
- JSON Body:**

```
1 {
2   "email": "{{email_ok}}",
3   "userName": "",
4   "password": "{{password_ok}}"
5 }
```

- Test Results (2/2):**
 - PASSED: El código de estado HTTP es 200
 - PASSED: La respuesta contiene un token de autenticación válido
- Response Headers:** 200 OK, 152 ms, 607 B
- Response Body:** JSON

```
1 {
2   "token": "eyJhbGciOiJodHRw0i8vd3d3LnczLm9yZy8yMDAxLzA0L3htbGRzaWctbW9yZSNobWFjLXNoYTI1NiIsInR5cCI6IkpxVCJ9.eyJleHAiOjE3NjAzOTQ3MjEsImh0dHA6Ly9zY2h1bWFzLnhhbHNvYXaub3JnL3dzLzIwMDUvaWlbnRpdkHkvY2xhawIzL3NpZCI6IjUiLCJodHRwO18vczNoZW1hc54bwxz2FwLm9yZy93cy8yMDA1LzA1L21kZw50aXR5L2NsYw1tcy9uYW1IjoivXNlcjEwNzcwIiwiawF01joXNzU5Nzg50TIxLCJuYmY1ojeNTk3OD0k5MjF9.AR-LUFx7DwREy7CSwMTX1qae1Fs62bF3vPGYczHUNiY",
3   "userName": "User10770",
4   "email": "user10770@vacapp.com"
5 }
```

The screenshot shows the Postman interface with a collection named 'VacAppTestLocal'. A POST request is made to `{{baseUrl}}/api/v1/User/sign-in`. The request body is:

```

1 {
2   "email": "",
3   "userName": "{{username_ok}}",
4   "password": "{{password_ok}}"
5 }

```

The response status is 200 OK, with a duration of 157 ms and a size of 607 B. Test results indicate the HTTP status is 200 and the response contains a valid authentication token.

The screenshot shows the Postman interface with a collection named 'VacAppTestLocal'. A POST request is made to `{{baseUrl}}/api/v1/User/sign-in`. The request body is identical to the successful one above:

```

1 {
2   "email": "",
3   "userName": "{{username_ok}}",
4   "password": "{{password_ok}}"
5 }

```

The response status is 200 OK, with a duration of 157 ms and a size of 607 B. The JSON response is:

```

1 {
2   "token": "eyJhbGciOiJodHRw0i8vd3d3LnczM9yZy8yMDAxLzA0L3htbGRzaWctbW9yZSNobWFjLXNoYTI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3NjAzOTQ4MjUsImh0dHA6Ly92Y2h1bWFzLnhbHNvYXAub3JnL3dzLzIwMDUvaWlRbnRpdkhvY2xhawLzL3NpZC16IjUiLCJodHRwO18vc2NzZW1hcy54bwzxzb2FwLm9yZy93cy8yMDA1LzA1L21kZW50aXR5L2NyW1tcy9uYw11IjoiVXNlcjEwNzcwIiwiwAF0IjoxNzU5NzkwMDI1LCJuYmY1oje3NTk30TawM19v.PRvzz35lnl3nwoZogB21K2tFKhaetTZVhp42iNqqAxKM",
3   "userName": "User10770",
4   "email": "user10770@vacapp.com"
5 }

```

Sign-in — Usuario inexistente

Comprueba el comportamiento ante un nombre o correo no registrado.

Request: POST `{{baseUrl}}/api/v1/User/sign-in`

Descripción: Devuelve un error de autenticación controlado.

The screenshot shows the Postman interface with a collection named 'VacAppTestLocal'. A test case titled 'Sign-in — Invalid username or password' is selected. The request method is POST, the URL is {{baseUrl}}/api/v1/User/sign-in, and the body is raw JSON:

```

1 {
2   "email": "{{email_invalid}}",
3   "userName": "",
4   "password": "{{password_ok}}"
5 }

```

The test results show a 500 Internal Server Error with two passed assertions: 'Status code is 500' and 'Response body contains 'Invalid username or password''. The status bar at the bottom indicates the test was run online.

This screenshot is nearly identical to the one above, showing the same collection and test case. The request body is the same raw JSON:

```

1 {
2   "email": "{{email_invalid}}",
3   "userName": "",
4   "password": "{{password_ok}}"
5 }

```

The test results show a 500 Internal Server Error with two passed assertions: 'Status code is 500' and 'Response body contains 'Invalid username or password''. The status bar at the bottom indicates the test was run online.

Sign-in — Contraseña incorrecta

Evaluá la respuesta del sistema ante credenciales erróneas.

Request: POST {{baseUrl}}/api/v1/User/sign-in**Descripción:** Devuelve un mensaje de error por password incorrecto.

The screenshot shows the Postman interface with the following details:

- Collection:** VacAppTestLocal
- Request:** POST {{baseUrl}}/api/v1/User/sign-in
- Body (JSON):**

```

1 {
2   "email": "",
3   "username": "{{username_ok}}",
4   "password": "{{password_invalid}}"
5 }
```

- Status:** 500 Internal Server Error
- Test Results (2/2):**
 - PASSED Status code is 500
 - PASSED Response body contains 'Invalid username or password'

The screenshot shows the Postman interface with the following details:

- Collection:** VacAppTestLocal
- Request:** POST {{baseUrl}}/api/v1/User/sign-in
- Body (JSON):**

```

1 {
2   "email": "",
3   "username": "{{username_ok}}",
4   "password": "{{password_invalid}}"
5 }
```

- Status:** 500 Internal Server Error
- Test Results (2/2):**
 - Raw Response (Expanded):


```

1 System.UnauthorizedAccessException: Invalid username or password
2   at VacApp_Bovinova_Platform.IAM.Application.CommandServices.UserCommandService.Handle(SignInCommand command) in /Users/maycolrojas/Documents/GitHub/Bacckend-VacApp/VacApp-Bovinova-Platform/IAM/Application/CommandServices/
UserCommandService.cs:line 68
3   at VacApp_Bovinova_Platform.IAM.Interfaces.RESTUserController.SignIn(SignInResource resource) in /Users/maycolrojas/
Documents/GitHub/Bacckend-VacApp/VacApp-Bovinova-Platform/IAM/Interfaces/REST/UserController.cs:line 60
4   at lambda_method102(Closure<Object>)
5   at Microsoft.AspNetCore.Mvc.Infrastructure.ActionMethodExecutor.TaskOfActionResultExecutor.Execute(ActionContext
actionContext, IActionResultTypeMapper mapper, ObjectMethodExecutor executor, Object controller, Object[] arguments)
6   at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeActionMethodAsync>g__Logged|12_1
(ControllerActionInvoker invoker)
7   at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeNextActionFilterAsync>g__Awaited|10_0
```

Bovino — Creación exitosa

Crea un bovino asociado a un establo existente.

Request: POST {{baseUrl}}/api/v1/bovines**Descripción:** La API retorna 201 Created con los datos del bovino.

POST {{baseUrl}}/api/v1/bovines

Key	Value	Description
Name	lola	
Gender	female	
BirthDate	2025-10-06T23:12:22.228Z	
Breed	cmamcac	
Location	chiclayo	
FileData	fondo-de-papel-psicodelico-abstrac...	

Body Cookies Headers (5) Test Results (4/4)

```

1  {
2   "id": 3,
3   "name": "lola",
4   "gender": "female",
5   "birthDate": "2025-10-06T23:12:22.228Z",
6   "breed": "cmamcac",
7   "location": "chiclayo",
8   "bovineImg": "https://res.cloudinary.com/do4y0ivdv/image/upload/v1759794529/gomnjggkbritwoqzzemw.webp",
9   "stableId": 2
10

```

POST {{baseUrl}}/api/v1/bovines

Key	Value	Description
Name	lola	
Gender	female	
BirthDate	2025-10-06T23:12:22.228Z	
Breed	cmamcac	
Location	chiclayo	
FileData	fondo-de-papel-psicodelico-abstrac...	

Body Cookies Headers (5) Test Results (4/4)

PASSED Código 201 (Created)

PASSED El cuerpo contiene las propiedades esperadas

PASSED El campo bovineImg contiene una URL válida

PASSED Guardar ID del bovino en el environment

Bovino — Duplicado

Verifica que no se permitan nombres de bovinos repetidos.

Request: POST {{baseUrl}}/api/v1/bovines

Descripción: Devuelve error controlado indicando duplicidad.

The screenshot shows the Postman interface with a dark theme. On the left, the 'My Workspace' sidebar lists collections, environments, flows, and history. The main area displays a POST request to 'VacAppTestLocal / Bovino / bovinoduplicado'. The request body contains fields: Name (lola), Gender (female), BirthDate (2025-10-06T23:12:22Z), Breed (cmamcac), Location (chiclayo), and FileData (a file named 'fondo-de-papel-psicodelico-abstrac...'). The response status is '500 Internal Server Error' with a message: 'System.Exception: Bovine entity with name 'lola' already exists.' The test results section shows 4/4 tests passed.

This screenshot shows the same Postman session after a fix. The request body is identical to the one above. The response status is now '500 Internal Server Error' with the message: 'Mensaje indica que el bovino ya existe.' The test results section shows 4/4 tests passed, indicating the issue has been resolved.

Stables — Creación exitosa

Registra un nuevo establo con límite de capacidad.

Request: POST {{baseUrl}}/api/v1/stables

Descripción: La API confirma creación con código 201.

The screenshot shows the Postman interface with a dark theme. On the left, the 'My Workspace' sidebar lists collections: 'VacAppTestLocal' (Sign-up, Sign-in, Bovino, Admin, Stables), 'Environments', 'Flows', and 'History'. The main workspace shows a 'POST create stables' request under 'Stables'. The request URL is `http://{{baseUrl}}/api/v1/stables`. The 'Body' tab is selected, showing raw JSON:

```
1 {  
2   "name": "prueba",  
3   "limit": 20  
4 }
```

The 'Test Results' section shows three green 'PASSED' status messages:

- Status code is 201 - Stable successfully created
- Response has valid properties with correct types
- Handle error status codes

At the bottom, the status bar indicates '201 Created' with a response time of 74 ms.

This screenshot is identical to the one above, showing the same workspace and request details. The difference is in the 'Body' tab, where the JSON response is visualized:

```
{ } JSON
```

```
1 {  
2   "id": 1,  
3   "name": "prueba",  
4   "limit": 20  
5 }
```

The rest of the interface, including the test results and status bar, remains the same.

Stables — Repetido

Controla que no se permitan nombres de establos duplicados.

Request: `POST {{baseUrl}}/api/v1/stables`

Descripción: Devuelve error informando que el establo ya existe.

The screenshot shows the Postman interface with a collection named "VacAppTestLocal". A POST request is defined for the endpoint `"/Stables/stable repetido"`. The "Post-response" tab contains a script to validate the response status and content. The "Test Results" section shows four green "PASSED" outcomes: "Código 500 (Bad Request)", "Mensaje indica que el establo ya existe", "No se creó un nuevo establo", and "Respuesta válida del servidor". The status bar at the bottom indicates a 500 Internal Server Error.

This screenshot shows the same Postman setup as the previous one, but the test results are now failing. The "Test Results" section shows a single red "FAILED" outcome: "500 Internal Server Error". The status bar at the bottom also displays a 500 Internal Server Error.

Staff — Creación exitosa

Crea un empleado y lo asocia a una campaña activa.

Request: `POST {{baseUrl}}/api/v1/staff`

Descripción: La API responde con `201 Created` y datos del empleado.

The screenshot shows the Postman interface with a dark theme. On the left, the sidebar displays 'My Workspace' with a collection named 'VacAppTestLocal' containing several API endpoints like 'Sign-up', 'Sign-in', 'Bovino', 'Admin', 'Stables', 'Campaigns', and 'Staff'. The 'Staff' endpoint is selected, and a 'New Request' button is visible at the bottom.

The main workspace shows a 'POST New Request' dialog for the URL `http://VacAppTestLocal / Staff / New Request`. The 'Body' tab is selected, showing raw JSON input:

```

1 {
2   "name": "Juan Torres",
3   "employeeStatus": 1,
4   "campaignId": 2
5 }
6

```

The response section shows a green status bar indicating '201 Created' with a response time of '72 ms' and a size of '264 B'. Below the status bar, the response body is displayed as JSON:

```

1 {
2   "id": 1,
3   "name": "Juan Torres",
4   "employeeStatus": 1,
5   "campaignId": 2
6 }

```

This screenshot shows the same Postman session after running the test. The 'Test Results' tab is selected, displaying three green 'PASSED' status messages:

- PASSED Status code is 201 - Staff created successfully
- PASSED Response contains employee data
- PASSED Employee name matches input

Staff — Duplicado

Evita la creación de un mismo empleado repetidamente.

Request: `POST {{baseUrl}}/api/v1/staff`

Descripción: Retorna error indicando duplicado de registro.

The screenshot shows the Postman interface with a failing API test. The collection is 'My Workspace' and the environment is 'VacAppTestLocal'. The test step is titled 'POST Staff Duplicado'.

Pre-request Script:

```
pm.test("Status code indicates duplicate error", function () {
  pm.expect(pm.response.code).to.eql(500);
});
```

Post-response Script:

```
pm.test("Response contains duplicate message", function () {
  pm.expect(pm.response.text()).to.include("already exists");
});
```

Body Response:

```
System.Exception: Staff entity with name 'Juan Torres' already exists.
at VacApp_Bovinova_Platform.StaffAdministration.Application.Internal.CommandServices.StaffCommandService.Handle
(CreateStaffCommand command) in /Users/maycolrojas/Documents/GitHub/Bacckend-VacApp/VacApp-Bovinova-Platform/
StaffAdministration/Application/Internal/CommandServices/StaffCommandService.cs:line 24
at VacApp_Bovinova_Platform.StaffAdministration.Interfaces.REST.StaffController.CreateStaffs(CreateStaffResource resource)
in /Users/maycolrojas/Documents/GitHub/Bacckend-VacApp/VacApp-Bovinova-Platform/StaffAdministration/Interfaces/REST/
StaffController.cs:line 41
at Microsoft.AspNetCore.Mvc.Infrastructure.ActionMethodExecutor.TaskOfIActionResultExecutor.Execute(ActionResult
actionContext, IActionResultTypeMapper mapper, ObjectMethodExecutor executor, Object controller, Object[] arguments)
at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeActionMethodAsync>g__Logged|12_1
(ControllerActionInvoker invoker)
at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeNextActionFilterAsync>g__Awaited|10_0
(ControllerActionInvoker invoker, Task lastTask, State next, Scope scope, Object state, Boolean isCompleted)
at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.Rethrow(ActionExecutedContextSealed context)
at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.Next(State& next, Scope& scope, Object& state, Boolean&
...)
```

Status: 500 Internal Server Error

The screenshot shows the Postman interface with a passing API test. The collection is 'My Workspace' and the environment is 'VacAppTestLocal'. The test step is titled 'POST Staff Duplicado'.

Body Request:

```
{
  "name": "Juan Torres",
  "employeeStatus": 1,
  "campaignId": 2
}
```

Status: PASSED

Test Results:

- Status code indicates duplicate error
- Response contains duplicate message

Campaign — Creación exitosa

Registra una nueva campaña ganadera.

Request: POST {{baseUrl}}/api/v1/campaigns

Descripción: Devuelve 201 Created con datos de la campaña.

The screenshot shows the Postman interface with a successful API call. The URL is `http://VacAppTestLocal / Campaigns / Crea Campaña`. The response status is `201 Created` with a response time of `13 ms` and a size of `753 B`.

```

1 {
2   "name": "Campaña Reproductiva Primavera 2025",
3   "description": "Iniciativa enfocada en mejorar los indices de fertilidad y reproducción del hato ganadero mediante control hormonal y selección genética.",
4   "startDate": "2025-09-01T00:00:00.000Z",
5   "endDate": "2025-12-15T00:00:00.000Z",
6   "status": "Draft",
7   "goals": [
8     {
9       "id": 6,
10      "title": "Aumentar tasa de natalidad en un 20%",
11      "description": "Aplicación de tratamientos hormonales y seguimiento del ciclo reproductivo en hembras seleccionadas."
12    }
13  ],
14  "channels": [
15

```

The screenshot shows the Postman interface with a successful API call. The URL is `http://VacAppTestLocal / Campaigns / Crea Campaña`. The response status is `201 Created` with a response time of `13 ms` and a size of `753 B`.

```

1 {
2   "id": 7,
3   "name": "Campaña Reproductiva Primavera 2025",
4   "description": "Iniciativa enfocada en mejorar los indices de fertilidad y reproducción del hato ganadero mediante control hormonal y selección genética.",
5   "startDate": "2025-09-01T00:00:00Z",
6   "endDate": "2025-12-15T00:00:00Z",
7   "status": "Draft",
8   "goals": [
9     {
10       "id": 6,

```

The Test Results section shows five green "PASSED" status messages:

- PASSED Código 201 (Created)
- PASSED Estructura de campaña creada correctamente
- PASSED Metas de la campaña registradas correctamente
- PASSED Fechas con formato ISO válido
- PASSED Guardar ID de campaña creada

Campaign — Duplicado

Valida el control de duplicidad en nombres de campañas.

Request: `POST {{baseUrl}}/api/v1/campaigns`

Descripción: Devuelve error indicando campaña existente.

The screenshot shows two instances of the Postman application interface. Both instances are focused on a collection named 'VacAppTestLocal' under the 'Campaigns' folder. A specific POST request named 'POST DuplicadoCampaña' is selected.

Request Details:

- Method: POST
- URL: {{baseUrl}}/api/v1/campaigns
- Pre-request Script (JavaScript):

```

1 // --- Validar código HTTP ---
2 pm.test("Código 500 (Bad Request)", function () {
3     pm.response.to.have.status(500);
4 });
5
6 // --- Verificar mensaje de error específico ---
7 pm.test("Mensaje indica que la campaña ya existe", function () {
8     const body = pm.response.text();
9     pm.expect(body).to.include("Campaign already exists");
10 });
11
12 // --- Verificar que no se devuelva un ID nuevo ---
13 pm.test("No se crea una nueva campaña duplicada", function () {
14     const body = pm.response.text();
15     pm.expect(body).to.not.include("id");
16 });

```
- Body, Cookies, Headers, and Scripts tabs are visible.

Test Results:

The results show a single test failure:

- Test Result: 500 Internal Server Error
- Details: 7 ms, 3.45 KB
- Message: System.Exception: Campaign already exists
- Stack Trace (partial):

```

1 at VacApp_Bovinova_Platform.CampaignManagement.Application.Internal.CommandServices.CampaignCommandService.Handle
2 (CreateCampaignCommand) in /Users/maycolrojas/Documents/GitHub/Bacckend-VacApp/VacApp-Bovinova-Platform/
3 CampaignManagement/Application/Internal/CommandServices/CampaignCommandService.cs:line 16
4 at VacApp_Bovinova_Platform.CampaignManagement.Interfaces.REST.CampaignController.CreateCampaign(CreateCampaignResource
5 resource) in /Users/maycolrojas/Documents/GitHub/Bacckend-VacApp/VacApp-Bovinova-Platform/CampaignManagement/Interfaces/
6 REST/CampaignController.cs:line 33
7 at lambda_method398(Closure, Object)
8 at Microsoft.AspNetCore.Mvc.Infrastructure.ActionMethodExecutor.TaskOfActionResultExecutor.Execute(ActionContext
9 actionContext, IActionResultTypeMapper mapper, ObjectMethodExecutor executor, Object controller, Object[] arguments)
10 at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeActionMethodAsync>g__Logged|12_1
11 (ControllerActionInvoker invoker)

```

Second Instance of Postman:

This instance shows the same setup but with a different outcome. The 'Test Results' tab indicates 4/4 passed tests:

- PASSED Código 500 (Bad Request)
- PASSED Mensaje indica que la campaña ya existe
- PASSED No se crea una nueva campaña duplicada
- PASSED Cabecera Content-Type válida

Admin — Creación válida

Crea un administrador del sistema con credenciales correctas.

Request: POST {{baseUrl}}/api/v1/admin/create

Descripción: Devuelve 201 Created con los datos del nuevo admin.

The screenshot shows the Postman interface with a successful API call. The collection 'VacAppTestLocal' contains several requests under the 'Admin' folder. The 'POST admin-valid' request is selected. The 'Body' tab shows a JSON payload with a single key 'email': '{{email_admin}}'. The response status is '201 Created' with a response time of 53 ms and a body size of 684 B. The response content is a complex JSON token.

```

1 {
2   "token": "eyJhbGciOiJodHRwOi8vd3d3LnczLm9yZy8yMDAxLzA0L3htbGZraWctbw9yZSNobWfjLXNoYTlI1NiIsInR5cCI6IkpXVCJ9.
eyJleHAiOiE3NjAzOTx000sImhdHA6Ly9zY2h1WFZlnhtbNvYXAb3JnL3dzLzIwMDUvMDUvaWR1bnRpdkVvY2xaW1zL3NpZCI6IjEiLCJo
dHlwO18vc2NoZWlhcy54Dwzb2Fwlm9yZy93c8yMDA1LzA1L21kZW50aXRSL2NsYWltcy9uW1l1joidXN1cmFkbWluQHZhY2FwcC5jb20ILCJ1
c2Vyx3R5cgUi010JBZG1pIis1mhdCI6MTC10Tc5MTM4NCwibmJmIjoxNzU5NzkxMzg0fQ.
NoC8pOK09_y7DwOnsN_PwQzE-oUsd9Ls6JyubbgGxkU",
3   "email": "useradmin@vacapp.com"
4 }

```

This screenshot is identical to the one above, showing a successful POST request to '/api/v1/admin/create' with a valid email. Below the main response, there is a detailed 'Test Results' section with five green 'PASSED' status boxes, each detailing a specific validation point: Status Code es 200 o 201, Respuesta contiene 'email', Email pertenece al dominio @vacapp.com, Respuesta contiene 'token', and Token tiene formato JWT (ey...).

Admin — Datos inválidos

Evaluá validaciones de campos requeridos.

Request: POST {{baseUrl}}/api/v1/admin/create

Descripción: Devuelve 400 Bad Request indicando datos inválidos.

The screenshot shows the Postman interface with a collection named 'VacAppTestLocal'. A test case 'POST admin-invalid' is selected. The request URL is `{baseUrl}/api/v1/admin/create`. The request body is set to raw JSON with the value:

```

1 {
2   "email": "{{email_admin_invalid}}"
3 }

```

The response status is 400 Bad Request, with the message: "Admin email must end with @vacapp.com (Parameter 'email')".

The screenshot shows the Postman interface with the same setup as the previous one. The test case 'POST admin-invalid' is selected. The request URL and body are identical. The response status is 400 Bad Request, with the message: "Admin email must end with @vacapp.com (Parameter 'email')".

Below the response, two green status indicators are shown:

- PASSED Status code is 400
- PASSED Response body contains the expected error message

Admin — Email repetido

Valida que no se registren administradores con el mismo email.

Request: `POST {{baseUrl}}/api/v1/admin/create`

Descripción: Devuelve error **409 Conflict** informando duplicado.

The screenshot shows the Postman interface with the following details:

- Collection:** My Workspace
- Environment:** VacAppTestLocal
- Request:**
 - Method: POST
 - URL: {{baseUrl}}/api/v1/admin/create
 - Body (raw JSON):

```
1 {
2   "email": "{{admin_email_existe}}"
3 }
```
- Response:**
 - Status: 400 Bad Request
 - Body (JSON):

```
1   "Admin already exists with this email"
```

The screenshot shows the Postman interface with the following details:

- Collection:** My Workspace
- Environment:** VacAppTestLocal
- Request:**
 - Method: POST
 - URL: {{baseUrl}}/api/v1/admin/create
 - Body (raw JSON):

```
1 {
2   "email": "{{admin_email_existe}}"
3 }
```
- Test Results:**
 - PASSED: Status Code es 400 (Bad Request)
 - PASSED: Detalle del error indica Bad Request
 - PASSED: Mensaje indica que el admin ya existe
 - PASSED: Respuesta no incluye token (error esperado)

Variables de Entorno Utilizadas

- `baseUrl`
- `token`
- `email_ok, username_ok, password_ok`
- `email_duplicate, username_duplicate, password_duplicate`
- `email_invalid, password_invalid`
- `email_admin, email_admin_invalid, admin_email, admin_token, admin_email_existe`
- `bovineId, campaignId`

Ejemplo de Flujo de Pruebas

1. **Registro de usuario:** Ejecutar `Sign-Up – Registro exitoso`.
2. **Login:** Ejecutar `Sign-in – Inicio exitoso` y guardar el token.
3. **Crear estable:** Ejecutar `create stables`.
4. **Registrar bovino:** Ejecutar `Crear bovino`.
5. **Registrar campaña:** Ejecutar `Crea Campaña`.
6. **Registrar staff:** Ejecutar `crear staff`.

Cada request cuenta con scripts automáticos en la pestaña **Tests**, que verifican:

- El **código de estado HTTP** esperado (`200, 201, 400, 409`, etc.).
- Que la **respuesta sea JSON válida**.
- Que contenga **propiedades clave** (`id, name, status`, etc.).
- Que los mensajes de error sean claros y específicos.

Estrategia de Validación

Los tests fueron diseñados bajo criterios de *robustez e integridad*, simulando escenarios reales y de error:

- Duplicidad de registros.
- Violación de integridad referencial.
- Validación de campos requeridos.
- Respuestas HTTP coherentes con los estados del dominio.

6.2. Static testing & Verification

En este punto del testing, se verificarán algunos defectos de la aplicación que tal vez no afecten a la compilación del código, pero que estén lejos de los estándares establecidos por la comunidad.

6.2.1. Static Code Analysis

Realizaremos este análisis con el propósito de eliminar los bugs en este proceso de desarrollo que podría impedirnos el crecimiento del mismo, asimismo nos ayudará a mejorar la productividad y tendremos un mejor manejo con las variables y en términos de calidad, estaremos por encima de los términos establecidos.

6.2.1.1. Coding standard & Code conventions

Para mantener la legibilidad del código se preparó un `.editoconfig`. Esto con el fin de mantener las reglas en el formato del código (variables, métodos, atributos, etc.).

```
root = true

# All files

[*]

charset = utf-8
```

```
indent_style = space

indent_size = 2

end_of_line = lf

trim_trailing_whitespace = true

insert_final_newline = true

# Code files

[*.{cs,csx,vb,vbx}]

indent_size = 4

# XML project files

[*.{csproj,vbproj,vcxproj,vcxproj.filters,proj,projitems,shproj}]

indent_size = 2

# XML config files

[*.{props,tasks,ruleset,config,nuspec,resx,vsixmanifest,vsct}]

indent_size = 2

# JSON files

[*.json]

indent_size = 2

# YAML files

[*.{yml,yaml}]

indent_size = 2

# Shell scripts
```

```
[*.sh]
end_of_line = lf

# C# files

[*.cs]

##### Core EditorConfig Options #####
# Indentation and spacing
indent_size = 4
tab_width = 4

##### .NET Coding Conventions #####
# Organize usings
dotnet_sort_system_directives_first = true
dotnet_separate_import_directive_groups = false

# this. preferences
dotnet_style_qualification_for_field = false:warning
dotnet_style_qualification_for_property = false:warning
dotnet_style_qualification_for_method = false:warning
dotnet_style_qualification_for_event = false:warning

# Language keywords vs BCL types preferences
dotnet_style_predefined_type_for_locals_parameters_members = true:warning
dotnet_style_predefined_type_for_member_access = true:warning
```

```
# Parentheses preferences

dotnet_style_parentheses_in_arithmetic_binary_operators =
always_for_clarity:suggestion

dotnet_style_parentheses_in_relational_binary_operators =
always_for_clarity:suggestion

dotnet_style_parentheses_in_other_binary_operators =
always_for_clarity:suggestion

dotnet_style_parentheses_in_other_operators =
never_if_unnecessary:suggestion


# Modifier preferences

dotnet_style_require_accessibility_modifiers = always:warning

dotnet_style_READONLY_field = true:warning


# Expression-level preferences

dotnet_style_object_initializer = true:suggestion

dotnet_style_collection_initializer = true:suggestion

dotnet_style_explicit_tuple_names = true:warning

dotnet_style_null_propagation = true:suggestion

dotnet_style_coalesce_expression = true:suggestion

dotnet_style_prefer_is_null_check_over_reference_equality_method =
true:warning

dotnet_style_prefer_inferred_tuple_names = true:suggestion

dotnet_style_prefer_inferred_anonymous_type_member_names = true:suggestion

dotnet_style_prefer_auto_properties = true:suggestion

dotnet_style_prefer_conditional_expression_over_assignment = true:silent

dotnet_style_prefer_conditional_expression_over_return = true:silent

dotnet_style_prefer_compound_assignment = true:suggestion

dotnet_style_prefer_simplified_interpolation = true:suggestion
```

```
dotnet_style_prefer_simplified_boolean_expressions = true:suggestion
```

```
# Null-checking preferences
```

```
dotnet_style_coalesce_expression = true:suggestion
```

```
dotnet_style_null_propagation = true:suggestion
```

```
#### C# Coding Conventions ####
```

```
# var preferences
```

```
csharp_style_var_for_builtin_types = true:suggestion
```

```
csharp_style_var_when_type_is_apparent = true:suggestion
```

```
csharp_style_var_elsewhere = true:suggestion
```

```
# Expression-bodied members
```

```
csharp_style_expression_bodied_methods = when_on_single_line:suggestion
```

```
csharp_style_expression_bodied_constructors = false:suggestion
```

```
csharp_style_expression_bodied_operators = when_on_single_line:suggestion
```

```
csharp_style_expression_bodied_properties = when_on_single_line:suggestion
```

```
csharp_style_expression_bodied_indexers = when_on_single_line:suggestion
```

```
csharp_style_expression_bodied_accessors = when_on_single_line:suggestion
```

```
csharp_style_expression_bodied_lambdas = when_on_single_line:suggestion
```

```
csharp_style_expression_bodied_local_functions =
when_on_single_line:suggestion
```

```
# Pattern matching preferences
```

```
csharp_style_pattern_matching_over_is_with_cast_check = true:warning
```

```
csharp_style_pattern_matching_over_as_with_null_check = true:warning
```

```
csharp_style_prefer_switch_expression = true:suggestion
csharp_style_prefer_pattern_matching = true:suggestion
csharp_style_prefer_not_pattern = true:suggestion
csharp_style_prefer_extended_property_pattern = true:suggestion

# Null-checking preferences
csharp_style_throw_expression = true:suggestion
csharp_style_conditional_delegate_call = true:suggestion

# Modifier preferences
csharp_prefer_static_local_function = true:warning
csharp_preferred_modifier_order =
public,private,protected,internal,static,extern,new,virtual,abstract,sealed,override,readonly,unsafe,volatile,async:suggestion

# Code-block preferences
csharp_prefer_braces = true:warning
csharp_prefer_simple_using_statement = true:suggestion
csharp_style_namespace_declarations = file_scoped:warning
csharp_style_prefer_method_group_conversion = true:suggestion
csharp_style_prefer_top_level_statements = true:silent

# Expression-level preferences
csharp_prefer_simple_default_expression = true:suggestion
csharp_style_pattern_local_over_anonymous_function = true:suggestion
csharp_style_inlined_variable_declaration = true:suggestion
csharp_style_prefer_index_operator = true:suggestion
csharp_style_prefer_range_operator = true:suggestion
```

```
csharp_style_implicit_object_creation_when_type_is_apparent =  
true:suggestion  
  
csharp_style_prefer_tuple_swap = true:suggestion  
  
csharp_style_prefer_utf8_string_literals = true:suggestion  
  
csharp_style_deconstructed_variable_declaration = true:suggestion  
  
csharp_style_unused_value_assignment_preference =  
discard_variable:suggestion  
  
csharp_style_unused_value_expression_statement_preference =  
discard_variable:silent  
  
  
# 'using' directive preferences  
  
csharp_using_directive_placement = outside_namespace:warning  
  
  
# New line preferences  
  
csharp_style_allow_embedded_statements_on_same_line_experimental =  
false:warning  
  
csharp_style_allow_blank_lines_between_consecutive_braces_experimental =  
false:warning  
  
csharp_style_allow_blank_line_after_colon_in_constructor_initializer_exper  
imental = false:warning
```

C# Formatting Rules

```
# New line preferences  
  
csharp_new_line_before_open_brace = all  
  
csharp_new_line_before_else = true  
  
csharp_new_line_before_catch = true  
  
csharp_new_line_before_finally = true  
  
csharp_new_line_before_members_in_object_initializers = true  
  
csharp_new_line_before_members_in_anonymous_types = true
```

```
csharp_new_line_between_query_expression_clauses = true

# Indentation preferences

csharp_indent_case_contents = true
csharp_indent_switch_labels = true
csharp_indent_labels = no_change
csharp_indent_block_contents = true
csharp_indent_braces = false
csharp_indent_case_contents_when_block = false

# Space preferences

csharp_space_after_cast = false
csharp_space_after_keywords_in_control_flow_statements = true
csharp_space_between_parentheses = false
csharp_space_before_colon_in_inheritance_clause = true
csharp_space_after_colon_in_inheritance_clause = true
csharp_space_around_binary_operators = before_and_after
csharp_space_between_method_declaration_parameter_list_parentheses = false
csharp_space_between_method_declaration_empty_parameter_list_parentheses = false
csharp_space_between_method_declaration_name_and_open_parenthesis = false
csharp_space_between_method_call_parameter_list_parentheses = false
csharp_space_between_method_call_empty_parameter_list_parentheses = false
csharp_space_between_method_call_name_and_opening_parenthesis = false
csharp_space_after_comma = true
csharp_space_after_dot = false
csharp_space_after_semicolon_in_for_statement = true
csharp_space_before_semicolon_in_for_statement = false
```

```
csharp_space_around_declaration_statements = false  
csharp_space_before_open_square_brackets = false  
csharp_space_between_empty_square_brackets = false  
csharp_space_between_square_brackets = false
```

Wrapping preferences

```
csharp_preserve_single_line_statements = false  
csharp_preserve_single_line_blocks = true
```

Naming conventions

Naming rules

```
dotnet_naming_rule.interface_should_be_begins_with_i.severity = warning  
dotnet_naming_rule.interface_should_be_begins_with_i.symbols = interface  
dotnet_naming_rule.interface_should_be_begins_with_i.style = begins_with_i
```

```
dotnet_naming_rule.types_should_be_pascal_case.severity = warning  
dotnet_naming_rule.types_should_be_pascal_case.symbols = types  
dotnet_naming_rule.types_should_be_pascal_case.style = pascal_case
```

```
dotnet_naming_rule.non_field_members_should_be_pascal_case.severity =  
warning  
dotnet_naming_rule.non_field_members_should_be_pascal_case.symbols =  
non_field_members  
dotnet_naming_rule.non_field_members_should_be_pascal_case.style =  
pascal_case
```

```
dotnet_naming_rule.private_or_internal_field_should_be_begins_with_undersc  
ore.severity = warning
```

```
dotnet_naming_rule.private_or_internal_field_should_be_begins_with_undersc  
ore.symbols = private_or_internal_field
```

```
dotnet_naming_rule.private_or_internal_field_should_be_begins_with_undersc  
ore.style = begins_with_underscore
```

```
dotnet_naming_rule.constant_should_be_pascal_case.severity = warning
```

```
dotnet_naming_rule.constant_should_be_pascal_case.symbols = constant
```

```
dotnet_naming_rule.constant_should_be_pascal_case.style = pascal_case
```

```
dotnet_naming_rule.static_READONLY_should_be_pascal_case.severity =  
warning
```

```
dotnet_naming_rule.static_READONLY_should_be_pascal_case.symbols =  
static_READONLY
```

```
dotnet_naming_rule.static_READONLY_should_be_pascal_case.style =  
pascal_case
```

Symbol specifications

```
dotnet_naming_symbols.interface.applicable_kinds = interface
```

```
dotnet_naming_symbols.interface.applicable_accessibilities = public,  
internal, private, protected, protected_internal, private_protected
```

```
dotnet_naming_symbols.interface.required_modifiers =
```

```
dotnet_naming_symbols.types.applicable_kinds = class, struct, interface,  
enum
```

```
dotnet_naming_symbols.types.applicable_accessibilities = public, internal,  
private, protected, protected_internal, private_protected
```

```
dotnet_naming_symbols.types.required_modifiers =
```

```
dotnet_naming_symbols.non_field_members.applicable_kinds = property,
```

```
event, method

dotnet_naming_symbols.non_field_members.applicable_accessibilities =
public, internal, private, protected, protected_internal,
private_protected

dotnet_naming_symbols.non_field_members.required_modifiers =


dotnet_naming_symbols.private_or_internal_field.applicable_kinds = field

dotnet_naming_symbols.private_or_internal_field.applicable_accessibilities =
internal, private, private_protected

dotnet_naming_symbols.private_or_internal_field.required_modifiers =


dotnet_naming_symbols.constant.applicable_kinds = field

dotnet_naming_symbols.constant.applicable_accessibilities = *

dotnet_naming_symbols.constant.required_modifiers = const


dotnet_naming_symbols.static_readonly.applicable_kinds = field

dotnet_naming_symbols.static_readonly.applicable_accessibilities = *

dotnet_naming_symbols.static_readonly.required_modifiers = static,
readonly


# Naming styles


dotnet_naming_style.pascal_case.required_prefix =

dotnet_naming_style.pascal_case.required_suffix =

dotnet_naming_style.pascal_case.word_separator =

dotnet_naming_style.pascal_case.capitalization = pascal_case


dotnet_naming_style.begins_with_i.required_prefix = I

dotnet_naming_style.begins_with_i.required_suffix =
```

```
dotnet_naming_style.begins_with_i.word_separator =
dotnet_naming_style.begins_with_i.capitalization = pascal_case
dotnet_naming_style.begins_with_underscore.required_prefix = _
dotnet_naming_style.begins_with_underscore.required_suffix =
dotnet_naming_style.begins_with_underscore.word_separator =
dotnet_naming_style.begins_with_underscore.capitalization = camel_case
```

Asimismo se implemento dentro del documento algunas reglas de refactorización utilizando Roslynator, un paquete de [dotnet](#) que facilita esto mismo.

```
# CA1031: Do not catch general exception types
dotnet_diagnostic.CA1031.severity = suggestion

# CA1062: Validate arguments of public methods
dotnet_diagnostic.CA1062.severity = suggestion

# CA1303: Do not pass literals as localized parameters
dotnet_diagnostic.CA1303.severity = none

# CA1707: Identifiers should not contain underscores
dotnet_diagnostic.CA1707.severity = none

# CA1711: Identifiers should not have incorrect suffix
dotnet_diagnostic.CA1711.severity = suggestion

# CA2007: Consider calling ConfigureAwait on the awaited task
dotnet_diagnostic.CA2007.severity = suggestion
```

```
# IDE0005: Using directive is unnecessary  
  
dotnet_diagnostic.IDE0005.severity = warning  
  
  
# IDE0055: Fix formatting  
  
dotnet_diagnostic.IDE0055.severity = warning  
  
  
  
# Remove unnecessary suppressions  
  
dotnet_diagnostic.IDE0079.severity = warning  
  
  
  
# Roslynator Analyzers – Configuración API  
  
  
  
dotnet_analyzer_diagnostic.category-Roslynator.severity = suggestion  
  
  
  
dotnet_diagnostic.RCS1021.severity = warning  
dotnet_diagnostic.RCS1118.severity = warning  
dotnet_diagnostic.RCS1036.severity = warning  
dotnet_diagnostic.RCS1123.severity = warning  
dotnet_diagnostic.RCS1213.severity = warning  
dotnet_diagnostic.RCS1045.severity = suggestion  
dotnet_diagnostic.RCS1158.severity = warning  
  
  
  
dotnet_diagnostic.RCS1002.severity = suggestion  
dotnet_diagnostic.RCS1032.severity = suggestion  
dotnet_diagnostic.RCS1084.severity = suggestion  
dotnet_diagnostic.RCS1215.severity = warning  
dotnet_diagnostic.RCS1046.severity = suggestion  
dotnet_diagnostic.RCS1128.severity = suggestion
```

```
dotnet_diagnostic.RCS1194.severity = suggestion
```

```
dotnet_diagnostic.RCS1077.severity = warning
```

```
dotnet_diagnostic.RCS1201.severity = warning
```

```
dotnet_diagnostic.RCS1100.severity = warning
```

```
dotnet_diagnostic.RCS1170.severity = warning
```

```
dotnet_diagnostic.RCS1188.severity = warning
```

```
dotnet_diagnostic.RCS1217.severity = warning
```

```
dotnet_diagnostic.RCS1018.severity = warning
```

```
dotnet_diagnostic.RCS1024.severity = warning
```

```
dotnet_diagnostic.RCS1072.severity = warning
```

```
dotnet_diagnostic.RCS1146.severity = warning
```

```
dotnet_diagnostic.RCS1210.severity = warning
```

```
dotnet_diagnostic.RCS1211.severity = warning
```

```
dotnet_diagnostic.RCS1196.severity = warning
```

```
dotnet_diagnostic.RCS1003.severity = suggestion
```

```
dotnet_diagnostic.RCS1216.severity = warning
```

```
dotnet_diagnostic.RCS1221.severity = warning
```

```
dotnet_diagnostic.RCS1243.severity = warning
```

Por ultimo la herramienta utilizada para refactorizar todo el formato de la aplicación fue el comando **dotnet format**, con esto realizando con éxito la refactorización y monitoreo de la aplicación en un futuro con nuevas iteraciones.

6.2.1.2. Code Quality & Code Security

Para este punto verificamos la calidad y la seguridad que mantiene el código. Para iniciar, la aplicación se realizó con Entity Framework Core como principal herramienta, esto ya nos da una gran ventaja debido a que la aplicación posee menos riesgo a tener inyecciones SQL.

Sin embargo, para que nuestra aplicación sea mas segura, se instalo otro paquete de **dotnet** para escanear y verificar el código, Security Code Scan. Además que tiene soporte para la Integración Continua que se tiene en nuestro aplicativo gracias a Azure. Este paquete como se dijo anteriormente, escanea el proyecto completo, donde puede detectar diferentes vulnerabilidades de seguridad, como Inyecciones SQL, XSS (Cross-site Scripting), XXE (XML External Entity Injection) y CSRF (Cross-site request Forgery).

Por ultimo, para mantener la calidad del código, se instalo otro paquete, Microsoft Code Analysis. Este paquete ayuda a que verifique si el código puede tener algún defecto dentro de la operabilidad y rendimiento, además de agregar mas reglas al **.editorconfig**.

6.2.2. Reviews

Luego de realizar todos los cambios dichos en los anteriores puntos, se pudo lograr lo querido. La aplicación ahora es mas segura gracias a los cambios y los futuros cambios se tendrán que hacer siguiendo las reglas del nuevo **.editorconfig**.

Esto nos da mas escalabilidad segura para la aplicación, tanto como tener un estándar establecido a comparación de antes de la refactorización. Esto ha sido una gran mejora para la aplicación y fielmente se cree que esto va a ir para mejor.

6.3. Validation Interviews

En esta sección, se registran y explican las actividades que abarcan las entrevistas de validación durante el desarrollo de nuestro proyecto. El objetivo principal de realizar estas entrevistas de validación es obtener retroalimentación, comprender las necesidades y expectativas de los usuarios, así como validar o refutar las hipótesis sobre el producto. Para lograr esto, haremos que nuestros entrevistados de ambos segmentos interactúen con la landing page y la aplicación.

6.3.1. Diseño de entrevistas

Para validar la efectividad y la experiencia de usuario de **VacApp**, se ha diseñado un cuestionario de entrevista dirigido a usuarios activos de la plataforma. El objetivo es recopilar feedback cualitativo y cuantitativo que permita medir la satisfacción, identificar las funcionalidades de mayor valor y descubrir oportunidades de mejora.

Estas preguntas están estructuradas para guiar una conversación fluida, abarcando desde la percepción general del producto hasta detalles específicos sobre su uso diario.

Cuestionario de Validación de Usuario - VacApp

Objetivo: Evaluar la experiencia del usuario, el valor percibido y las áreas de mejora de la aplicación.

Público: Ganaderos independientes y administradores de empresas ganaderas que utilizan VacApp.

Parte 1: Satisfacción General y Lealtad (NPS)

1. En una escala del 1 al 10, ¿qué tan probable es que recomiendes "VacApp" a otro ganadero o colega?

- *Guía para el entrevistador: Registrar la puntuación numérica. Si es 9-10 son promotores, 7-8 pasivos, 0-6 detractores.*

2. ¿Cuál es la razón principal de la calificación que acabas de dar?

- *Guía: Profundizar en los motivos. ¿Es por la facilidad de uso, una función específica, el soporte, o la falta de algo?*
-

Parte 2: Uso y Valor Percibido

3. ¿Cuál es la función o característica de "VacApp" que más utilizas en tu día a día?

- *Guía: Identificar las funcionalidades "ganadoras" que generan más engagement.*

4. ¿Cuál es el principal problema o tarea que "VacApp" te ayuda a resolver o facilitar?

- *Guía: Entender el "job-to-be-done" principal que la app está cumpliendo para el usuario.*

5. ¿Has notado alguna mejora medible desde que usas "VacApp"? (Ej: ahorro de tiempo, mejor control de inventario, reducción de errores en registros, etc.)

- *Guía: Buscar evidencia concreta del impacto positivo de la aplicación en la operación del usuario.*
-

Parte 3: Oportunidades de Mejora y Necesidades no Cubiertas

6. Si tuvieras una "varita mágica", ¿qué nueva función o reporte te gustaría que "VacApp" tuviera?

- *Guía: Fomentar la creatividad para descubrir ideas de nuevas funcionalidades o mejoras significativas.*

7. ¿Qué tarea importante relacionada con la gestión de tu ganado sientes que "VacApp" todavía NO te ayuda a gestionar?

- *Guía: Identificar los "gaps" o vacíos funcionales que la aplicación aún no cubre.*

8. ¿Hay alguna parte de la plataforma que te resulte confusa o difícil de usar?

- *Guía: Detectar puntos de fricción en la experiencia de usuario (UX) que necesiten ser rediseñados.*
-

Parte 4: Contexto y Cierre

9. ¿Qué herramienta o método (otra app, libreta de apuntes, Excel) usabas antes de "VacApp" para gestionar tu ganado?

- *Guía: Comprender el comportamiento previo del usuario y el valor diferencial que VacApp ofrece frente a la competencia o métodos tradicionales.*

10. ¿Hay algo más que te gustaría que supiéramos sobre tu experiencia con "VacApp"?

- *Guía: Pregunta abierta para capturar cualquier otro comentario, sugerencia o crítica que el usuario no haya mencionado.*

6.3.2. Registro de Entrevistas

6.3.2. Registro de Entrevistas

A continuación, se documentan las entrevistas de validación realizadas con usuarios de los segmentos objetivo. Cada registro incluye los datos del entrevistado, un resumen de los hallazgos clave, y las evidencias correspondientes.

Entrevista #1

Detalle	Información
Entrevistado	Sergio Gomez Vallejos
Edad	25
Rol / Experiencia	Ganadero Independiente
Fecha de Entrevista	08/11/2025
Duración	3:46
Tecnologías Usadas	Zoom
Enlace a Grabación	https://drive.google.com/file/d/1qC36xoN-Q9WqCXTWgixpkWKoWFHFszZv/view? usp=sharing

Evidencia de la Entrevista:



Resumen de Hallazgos Clave:

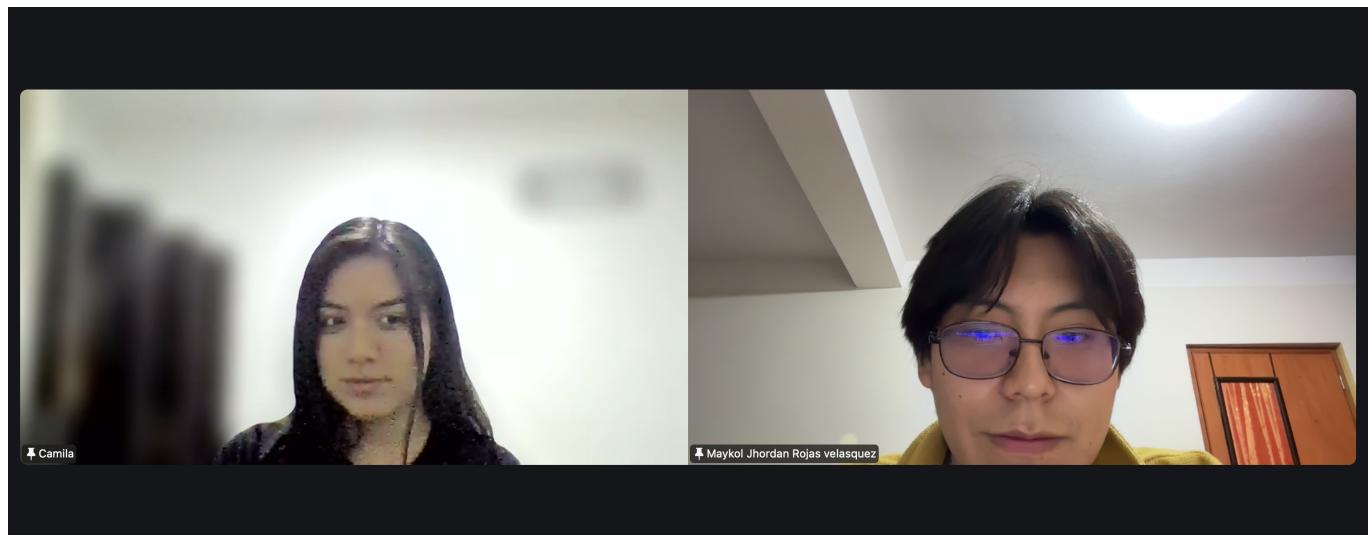
- **Feedback Positivo sobre Funcionalidad:** El entrevistado confirmó que la aplicación funciona correctamente y que pudo navegar por las distintas secciones sin problemas. La funcionalidad general fue validada como intuitiva y completa.
- **Oportunidad de Mejora en la Interfaz (UI):** A pesar de la buena funcionalidad, se señaló que el menú principal se percibe "sobrecargado". Este hallazgo sugiere una oportunidad para simplificar la navegación, posiblemente agrupando opciones o mejorando la jerarquía visual para no abrumar al usuario.
- **Solicitud de Nuevas Funcionalidades (Reportes):** El punto más relevante fue la solicitud de incorporar **reportes de alimentación y peso**. El usuario expresó un fuerte interés en poder visualizar y analizar datos históricos sobre el consumo de alimento y la evolución del peso del ganado, considerándolo una herramienta clave para la toma de decisiones.

Entrevista #2

Detalle	Información
Entrevistado	Camila Fernanda Farias Morales
Edad	21
Rol / Experiencia	Practicante en una empresa Ganadera
Fecha de Entrevista	09/11/2025
Duración	[HH:MM:SS]
Tecnologías Usadas	Zoom

Detalle	Información
Enlace a Grabación	[Pega aquí el enlace al video o audio de la entrevista]

Evidencia de la Entrevista:



Resumen de Hallazgos Clave:

- **Validación de Funcionalidades Clave:** La entrevistada valora positivamente la facilidad de uso y la practicidad de la aplicación, destacando los módulos de **manejo de bovinos y registro de vacunas** como los más utilizados. Confirma que la app resuelve el problema de la desorganización y la dependencia de registros manuales (Excel, cuadernos).
- **Oportunidades de Mejora Críticas:** Se identificaron dos áreas principales para futuras mejoras:
 1. **Reportes Avanzados:** Fuerte interés en la incorporación de **reportes de rendimiento productivo** (ej. litros de leche, ganancia de peso) para facilitar la toma de decisiones.
 2. **Alertas Automáticas:** Solicitud de un sistema de **alertas automáticas** más robusto para vacunas y revisiones veterinarias, considerándolo un diferenciador clave.
- **Funcionalidad Ausente (Gap):** La entrevistada señaló la falta de un módulo para **registrar gastos y costos de producción**, una funcionalidad que considera esencial para un control económico completo de la operación ganadera.
- **Feedback de Usabilidad:** Aunque la interfaz es mayormente intuitiva, hubo una curva de aprendizaje inicial para tareas como **editar registros antiguos o mover animales entre establos**.

Entrevista #3

Detalle	Información
Entrevistado	[Nombre Completo del Entrevistado]
Edad	[Edad]

Detalle	Información
Rol / Experiencia	[Ej: Ganadero con 10 años de experiencia]
Fecha de Entrevista	[DD/MM/AAAA]
Duración	[HH:MM:SS]
Tecnologías Usadas	[Ej: Google Meet, Zoom, Grabadora de voz]
Enlace a Grabación	[Pega aquí el enlace al video o audio de la entrevista]

Evidencia de la Entrevista:

![Evidencia Entrevista 1](assets/entrevista_1.png)

Resumen de Hallazgos Clave:**Entrevista #4**

Detalle	Información
Entrevistado	[Nombre Completo del Entrevistado]
Edad	[Edad]
Rol / Experiencia	[Ej: Ganadero con 10 años de experiencia]
Fecha de Entrevista	[DD/MM/AAAA]
Duración	[HH:MM:SS]
Tecnologías Usadas	[Ej: Google Meet, Zoom, Grabadora de voz]
Enlace a Grabación	[Pega aquí el enlace al video o audio de la entrevista]

Evidencia de la Entrevista:

![Evidencia Entrevista 1](assets/entrevista_1.png)

Resumen de Hallazgos Clave:**Entrevista #5**

Detalle	Información
Entrevistado	[Nombre Completo del Entrevistado]

Detalle	Información
Edad	[Edad]
Rol / Experiencia	[Ej: Ganadero con 10 años de experiencia]
Fecha de Entrevista	[DD/MM/AAAA]
Duración	[HH:MM:SS]
Tecnologías Usadas	[Ej: Google Meet, Zoom, Grabadora de voz]
Enlace a Grabación	[Pega aquí el enlace al video o audio de la entrevista]

Evidencia de la Entrevista:

![Evidencia Entrevista 1](assets/entrevista_1.png)

Resumen de Hallazgos Clave:**Entrevista #6**

Detalle	Información
Entrevistado	[Nombre Completo del Entrevistado]
Edad	[Edad]
Rol / Experiencia	[Ej: Ganadero con 10 años de experiencia]
Fecha de Entrevista	[DD/MM/AAAA]
Duración	[HH:MM:SS]
Tecnologías Usadas	[Ej: Google Meet, Zoom, Grabadora de voz]
Enlace a Grabación	[Pega aquí el enlace al video o audio de la entrevista]

Evidencia de la Entrevista:

![Evidencia Entrevista 1](assets/entrevista_1.png)

Resumen de Hallazgos Clave:**6.3.3. Evaluaciones según heurísticas****6.4. Auditoría de Experiencias de Usuario**

6.4.1. Auditoría realizada

El equipo **Vacow Team**, en su rol de auditores internos UX dentro de la iteración TB2, ejecutó una auditoría enfocada en **Experiencia de Usuario (UX)** y **usabilidad** aplicada sobre el informe, documentación y evidencias técnicas del proyecto *TukunTech*.

El objetivo fue identificar puntos críticos que afectan la lectura del documento, la consistencia visual y la claridad narrativa desde el punto de vista evaluador (comité docente), especialmente al momento de exportación final a PDF.

Esta auditoría se alineó con criterios de evaluación del flujo de información, claridad visual, trazabilidad de evidencias y adecuación de artefactos al estándar de entrega.

Componentes evaluados

Área evaluada	Elementos
Documento raíz	<code>README.md</code> (estructura, orden, legibilidad, jerarquías H1–H4, nivel de limpieza)
Recursos visuales	<code>/assets</code> (nomenclatura, referencias en Markdown, peso y resolución de imágenes)
Evidencias técnicas	Screenshots, endpoints verificados, Swagger, links de despliegue
Secciones técnicas	Testing, CI/CD, despliegue, integración y documentación técnica

Alcance

- Documento principal `README.md`
- Carpeta de recursos `/assets`
- Enlaces de despliegue (frontend y backend)
- Evidencias de pruebas funcionales (pantallas de test, Swagger, endpoints)
- Pipeline CI/CD presentado

Metodología

- **Inspección heurística** (Nielsen Norman)
- **Revisión documental** por consistencia estructural y narrativa
- **Validación de enlaces** y recursos embebidos
- **Pruebas rápidas de renderizado y exportación** a PDF

Herramientas utilizadas

- Google Chrome (pruebas de visualización y renderizado)
- Visual Studio Code (revisión y limpieza de Markdown)
- Figma (comparación con prototipos entregados)
- Swagger UI / Postman (verificación de endpoints)
- Exportador PDF (control de formato A4/Letter y legibilidad final)

Fechas de ejecución

08/11/2025 – 10/11/2025

Equipo auditor – Vacow Team

- **Maycol Jhordan Rojas Velásquez** – U202219984 – Auditor líder UI/UX
 - **Stephano Espinoza Cueva** – U202218590 – Auditor Frontend / validación de prototipos
 - **Rodrigo Liberato Saldaña** – U202215623 – Auditor Backend / validación técnica y evidencias
-

6.4.1.2. Cronograma de ejecución de auditoría

Fecha	Actividad	Responsable (Vacow Auditoría)
08/11/2025	Revisión de carátula, tabla de contenidos y jerarquías del README ; análisis de formato y errores de PDF	Maycol Rojas
09/11/2025	Evaluación heurística de flujos funcionales (login, dashboard) y validación de enlaces activos	Stephano Espinoza
10/11/2025	Consolidación de hallazgos, priorización de correcciones y elaboración del informe de recomendaciones UX-TB2	Rodrigo Liberato

Nota: La auditoría UX realizada por Vacow Team permitió identificar oportunidades de mejora en la estructura del documento y la presentación de evidencias técnicas, garantizando coherencia visual, accesibilidad y trazabilidad entre los artefactos entregados y la experiencia final del evaluador.

6.4.1.3. Contenido de auditoría realizada

SITE / APP evaluado: TukunTech — Aplicación orientada al monitoreo y lectura rápida del estado de salud de un paciente adulto mediante indicadores de signos vitales.

TAREAS EVALUADAS (flujos principales de la APP):

1. Registro de usuario (paciente / familiar)
2. Visualización del estado de salud actual en pantalla principal
3. Interpretación del riesgo mediante colores (verde / azul / rojo)
4. Visualización de histórico de mediciones
5. Visualización de información complementaria (contexto / explicación)
6. Navegación entre vistas básicas de la aplicación

TAREAS NO INCLUIDAS EN ESTA AUDITORÍA:

1. Ajustes avanzados de parámetros médicos
 2. Integración con wearables externos
 3. Gestión multi-paciente
 4. Configuración de notificaciones push
-

ESCALA DE SEVERIDAD USADA:

Nivel	Descripción
1	Observación leve — no afecta la experiencia general
2	Oportunidad de mejora — podría optimizar la claridad
3	Mejora importante — facilita comprensión de usuario

TABLA RESUMEN DE OBSERVACIONES

#	Observación relacionada a la APP	Severidad	Principio relacionado
1	La app se basa en colores para representar el estado del paciente, podría ser útil acompañar este indicador con una breve leyenda textual debajo para reforzar significado.	2	Reconocer antes que recordar
2	Sería beneficioso incluir un “estado previo reciente” al lado del estado actual para facilitar comparación rápida.	2	Contexto de uso
3	El flujo de navegación entre pantallas podría incluir un indicador visual del paso actual (por ejemplo: “inicio / histórico”) para orientar mejor al usuario.	2	Mapeo conceptual
4	En el historial sería útil resaltar valores fuera de rango para priorizar atención visual inmediata.	3	Visibilidad de información crítica
5	La pantalla principal podría incluir un acceso directo al historial de alertas sin tener que pasar por más pantallas.	2	Minimizar esfuerzo del usuario

EJEMPLO DE OBSERVACIÓN DETALLADA

Observación #4 — resaltado de valores fuera de rango

Severidad: 3

Principio: Visibilidad de información crítica

Comentario:

La información del historial es útil pero podría destacar visualmente los datos en riesgo para que el usuario identifique rápidamente cuando hubo episodios relevantes.

Recomendación:

Resaltar con color o ícono los valores fuera del rango normal para acelerar el reconocimiento visual del riesgo.

6.4.2. Auditoría recibida

6.4.2.1. Información del grupo auditor

Se recibió auditoría de Experiencia de Usuario (UX) por parte del **Grupo 4**, quienes realizaron una evaluación orientada a validar la claridad del documento, la correcta jerarquización de secciones, así como la coherencia entre evidencias anexas y el flujo funcional presentado dentro del README y la documentación de entrega.

El alcance del grupo 4 incluyó:

- Revisión de la estructura general del documento principal (README.md)
- Validación del orden lógico y jerarquías H1-H4
- Validación visual de imágenes / capturas funcionales
- Evaluación UX de navegación y lectura narrativa
- Revisión del acceso a evidencias técnicas (endpoints / despliegues)
- Observación general de consistencia entre prototipo UI y documentación final

Fecha de auditoría recibida: 09/11/2025

Integrante	Código	Rol
Palomares Chavez Adriana	u20201f723	Líder
Soto Zorrilla Oscar Eduardo	U201811767	Integrante
Maraza Penemonte Erick	u202213372	Integrante

6.4.2.2. Cronograma de auditoría recibida

Fecha	Actividad ejecutada	Responsable (Grupo 4)	Detalle de lo realizado
08/11/2025	Revisión documental del README y análisis de jerarquía de secciones	Palomares Chavez Adriana	Analizó estructura global del documento y su alineación con los criterios requeridos para la entrega TB2
09/11/2025	Evaluación heurística UX sobre navegación y visualización de evidencias	Soto Zorrilla Oscar Eduardo	Validó el flujo funcional (login → dashboard) y la coherencia entre prototipos y capturas presentadas como evidencia
10/11/2025	Consolidación de observaciones y retroalimentación al equipo Vacow Team	Maraza Penemonte Erick	Sintetizó los hallazgos detectados y formalizó la retroalimentación enviada al equipo para las iteraciones de mejora antes del cierre TB2

6.4.2.3. Contenido de auditoría recibida

6.4.2.4. Resumen de modificaciones para subsanar hallazgos

Capítulo VII: DevOps Practices

7.1. Continuous Integration

La Integración Continua (CI) en este proyecto es la disciplina de desarrollo enfocada en mantener la calidad del código mediante la fusión frecuente y la validación automatizada. La CI abarca las etapas desde que el código es comprometido por el desarrollador hasta el momento en que se genera un artefacto verificable y listo para su distribución.

En nuestro modelo, la CI se activa con cada cambio, desencadenando automáticamente los siguientes pasos para asegurar que el código base esté siempre en un estado funcional:

- Activación: El desarrollador envía el código al Code Repository (repositorio de código).
- Validación de Compilación: El sistema de CI inicia el App Build Process para verificar la capacidad de compilación del código en un entorno estándar e independiente.
- Validación de Calidad: Tras una compilación exitosa, se ejecuta la Test Suite (pruebas unitarias, de integración, etc.) para detectar regresiones o bugs funcionales.

El principal beneficio de este enfoque es la detección y solución inmediata de fallos, manteniendo un historial de integración limpio y reduciendo el riesgo de problemas complejos durante las etapas finales de despliegue.

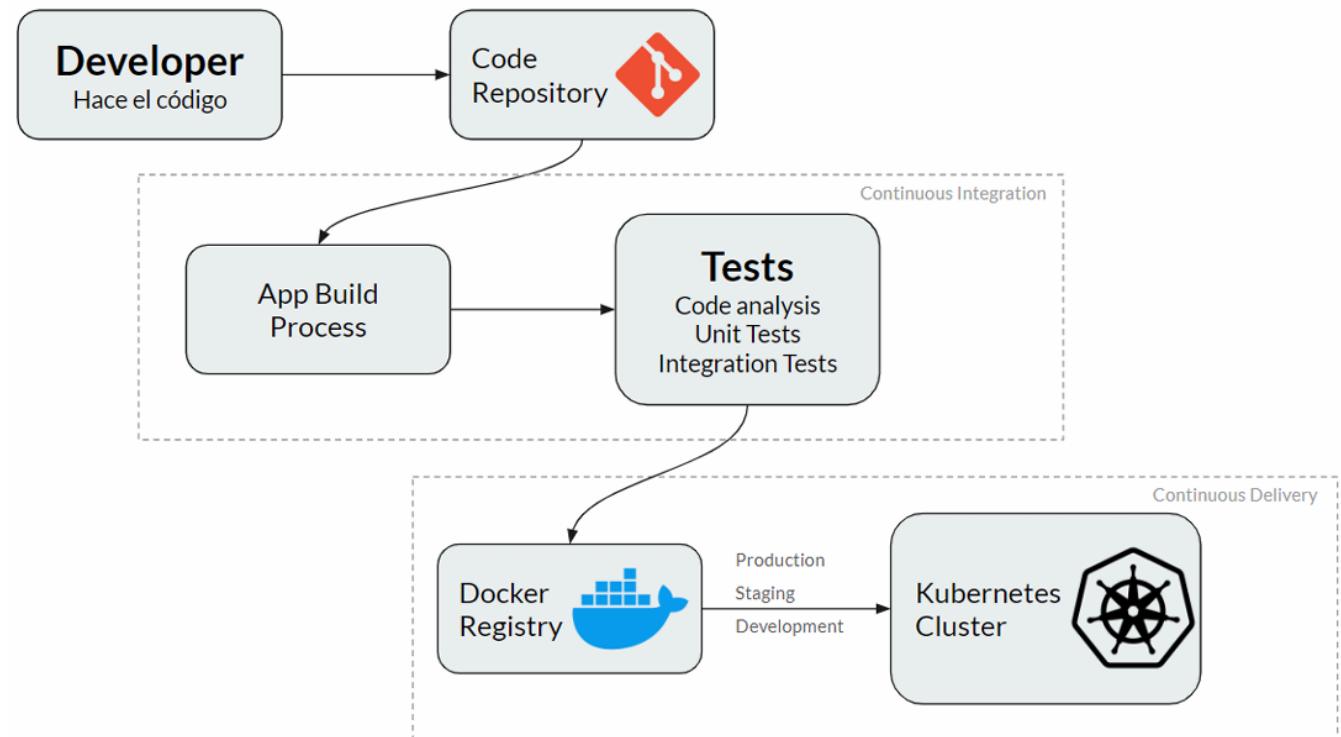
7.1.1. Tools and Practices

Esta sección detalla los componentes técnicos específicos y las prácticas requeridas para ejecutar la fase de Integración Continua del pipeline presentado:

Componente del Pipeline	Práctica Relevante	Función Específica en la Integración Continua (CI)
Code Repository (Git)	Fusión Frecuente	Sirve como la única fuente de verdad (<i>Single Source of Truth</i>) del código. La práctica de fusión diaria es mandatoria para evitar divergencias complejas.
App Build Process	Build Automatizado	Compila el código fuente en un artefacto ejecutable. Si la compilación falla, el <i>pipeline</i> se detiene inmediatamente para forzar la corrección.
Test Suite	Garantía de Calidad	Ejecuta el conjunto de pruebas (unitarias, de integración, etc.). Solo el código que supere el 100% de las pruebas avanza a la siguiente etapa.
Docker Registry	Creación de Artefacto Inmutable	Empaque el ejecutable validado en una imagen Docker estandarizada. Este artefacto inmutable es la salida final de la CI, listo para su distribución.
Developer Commitment	Detener la Línea (Stop the Line)	El desarrollador responsable de un cambio que cause un fallo debe priorizar la corrección del <i>pipeline</i> sobre cualquier otra tarea.

7.1.2. Build & Test Suite Pipeline Components.

Esta sección describe los componentes encargados de compilar el proyecto y ejecutar las pruebas automatizadas, asegurando que el código sea funcional y estable antes de ser desplegado.



```
✓ └─ bin └─ no index
  └─ CampañaTests · no index
    └─ C# CampaignTests.cs · no index
    └─ C# ChannelTests.cs · no index
    └─ C# GoalTests.cs · no index
  └─ IAMTests · no index
    └─ C# AdminTests.cs · no index
    └─ C# UserTests.cs · no index
  > └─ obj · no index
  └─ RanchTests · no index
    └─ C# BovineTests.cs · no index
    └─ C# StableTests.cs · no index
    └─ C# VaccineTests.cs · no index
  └─ StaffTests · no index
    └─ C# StaffTests.cs · no index
  C# VacAppTests.csproj · no index
```

7.2. Continuous Delivery

7.2.1. Tools and Practices.

El Continuous Delivery del proyecto se puso en marcha con un toolchain enfocado en la automatización y la aceleración del pipeline de Continuous Integration (CI) y deployment.

GitHub Actions

Se eligió a GitHub Actions como el Automation Server central para el pipeline CI/CD. Se configuraron workflows que se triggeran automáticamente. El engine arranca con events como pushes o pull requests hacia branches clave.

Estos workflows ejecutan trabajos esenciales: la dependency resolution, el build del proyecto, la test execution y el automated deployment del backend. Esta elección aseguró un time-to-market rápido para el backend, garantizando la product quality con procesos reproducibles.

Version Control y Git Flow

El código fuente se centralizó en GitHub, actuando como el repositorio remoto único para el control de versiones. Se estableció una estrategia de ramas clara:

- **develop**: La integration branch principal; aquí es donde se hace el merge las nuevas adiciones de todos los integrantes.
- **feature/{nombre_del_dev}**: Topic branches dedicadas para que cada contributor desarrolle su parte o realice hotfixes.

Esta estructura de git flow mantuvo un development workflow colaborativo y pulcro, minimizando los conflictos a la hora del merge y optimizando las integraciones continuas.

Quality Assurance con Automated Testing

Durante el ciclo de desarrollo se integraron varios test y tipos de automated testing para asegurar la verificar la calidad del sistema:

- **Unit Tests**: Validan el comportamiento basico de methods y funciones.
- **Integration Tests**: Verifican la interacción correcta entre los componentes del sistema.
- **Development Tests**: Pruebas rápidas que confirman la funcionalidad general del backend durante la etapa de correr el proyecto.
- **System Tests**: Chequean la end-to-end functionality y la interacción entre los core modules.

Para esta capa de pruebas se utilizaron NUnit (para el backend) y Postman (para la validation de API endpoints).

7.2.2. Stages Deployment Pipeline Components.

El pipeline de entrega continua está compuesto por diferentes etapas que garantizan que el código pase por procesos de validación, compilación y despliegue de forma automatizada. A continuación se detalla el flujo de trabajo implementado:

```
graph TD; A[A[Commit en GitHub]] --> B[B[Build]]; B --> C[C[Test]]; C --> D[D[Staging]]; D --> E[E[Approval]]; E --> F[F[Production]]
```

El proyecto sigue un flujo de trabajo bien definido para llevar el código desde el desarrollo hasta la producción.

Etapas del Pipeline:

- Confirmación de Código (Source): El pipeline se activa con la confirmación de cambios; esto ocurre cuando se realiza una subida o una solicitud de integración a una rama del repositorio en GitHub.
- Compilación (Build): En esta fase, la herramienta GitHub Actions ejecuta la compilación del proyecto de backend (.NET 9). Se restauran las dependencias y se generan los artefactos listos para el despliegue.
- Pruebas (Test): Se ejecutan las pruebas automatizadas (unitarias y de integración) para validar que el código cumple con los requisitos funcionales antes de avanzar al siguiente paso.
- Entorno de Prueba (Staging): El sistema se despliega automáticamente en un entorno de pruebas para la validación final y la revisión funcional por parte del equipo.
- Autorización (Approval): Una vez confirmado el correcto funcionamiento en el entorno de prueba, un miembro del equipo da la autorización para la promoción a producción.
- Despliegue Final (Production): El backend se despliega automáticamente en Azure, asegurando la alta disponibilidad y la estabilidad del servicio.

Por otro lado, la interfaz de usuario (frontend) se publica en Netlify, lo que permite actualizaciones continuas con cada nueva versión aprobada del proyecto.

7.3. Continuous deployment

El Continuous Deployment (CD) es la fase en la que cada cambio validado en el código se despliega automáticamente en los entornos de producción, garantizando que el software esté siempre actualizado, estable y disponible para los usuarios finales. Su implementación permite integrar las prácticas DevOps dentro del ciclo de vida del proyecto descrito en el documento, mejorando la entrega continua del producto en cada sprint.

7.3.1. Tools and Practices.

Las herramientas y prácticas utilizadas para implementar Continuous Deployment en el proyecto pueden incluir:

Herramientas

- GitHub Actions / GitLab CI / Jenkins / CircleCI: Automatización de flujos de integración y despliegue.

- Docker & Docker Compose: Empaquetado de los servicios del sistema (backend, frontend, base de datos) para despliegues consistentes.
- Kubernetes / AWS Elastic Beanstalk / Azure DevOps / Vercel / Netlify: Gestión de entornos de despliegue escalables y automatizados.
- Terraform / Ansible: Infraestructura como código (IaC) para configurar entornos reproducibles.
- SonarQube: Análisis continuo de la calidad del código antes de desplegar.
- Postman / Swagger / Newman: Validación automática de endpoints antes del despliegue.

Prácticas

- Automated Testing: Se ejecutan pruebas unitarias, de integración y de aceptación antes del despliegue.
- Blue-Green Deployment: Se mantiene una versión activa (blue) y una de prueba (green) para despliegue seguro.
- Canary Releases: Se libera gradualmente la nueva versión a una porción de usuarios para monitorear su estabilidad.
- Rollback Automático: Reversión inmediata en caso de error de despliegue.
- Monitoring & Logging: Supervisión de rendimiento post-despliegue mediante herramientas como Prometheus, Grafana o ELK Stack.

7.3.2. Production Deployment Pipeline Components

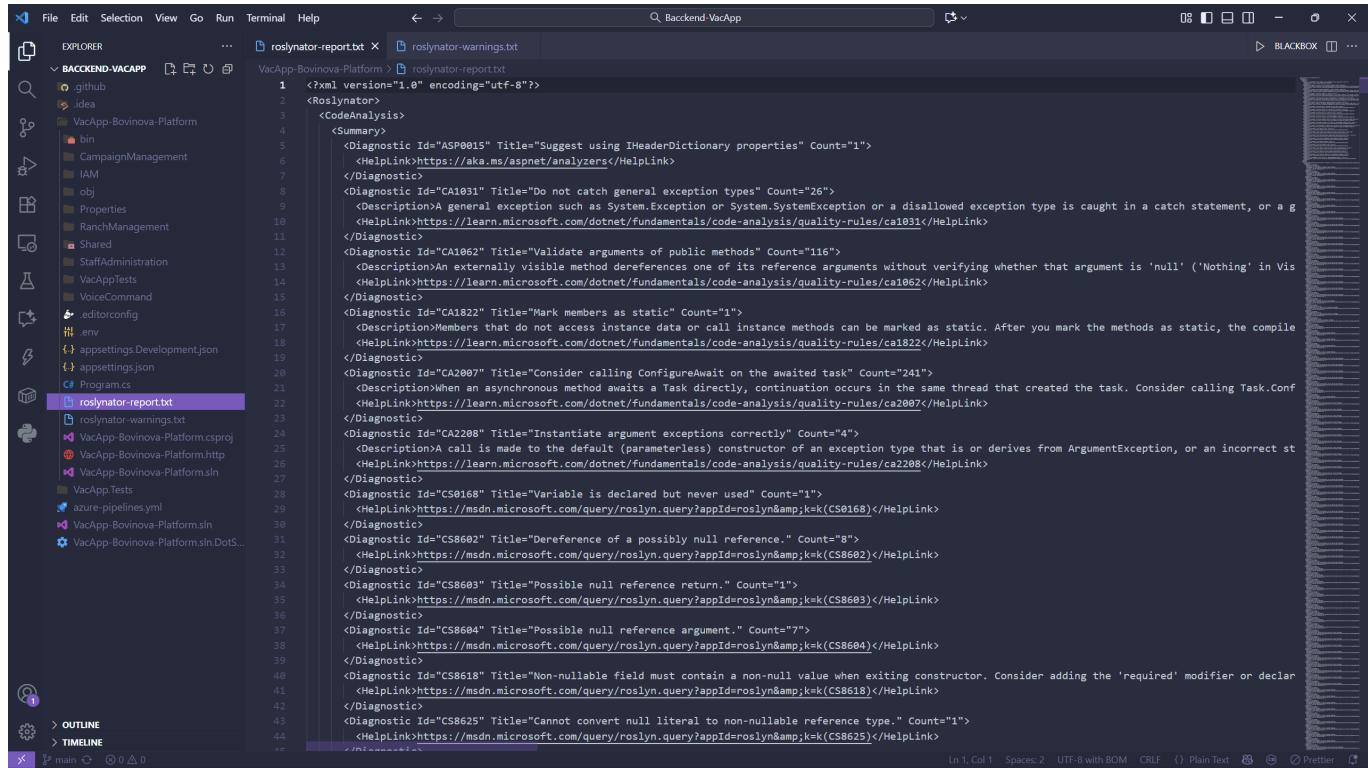
El Deployment Pipeline define las etapas secuenciales del flujo de entrega continua. Para este proyecto, se estructura de la siguiente forma:

Etapa	Componente / Objetivo	Herramientas / Ejemplos
1. Source Stage	Almacenamiento y versionamiento del código fuente.	Git + GitFlow en GitHub/GitLab
2. Build Stage	Compilación, empaquetado y creación de imágenes Docker.	GitHub Actions, Jenkins, Docker
3. Test Stage	Ejecución de pruebas automáticas: unitarias, integración, E2E.	Jest, PyTest, Postman/Newman, Cypress
4. Release Stage	Publicación del artefacto aprobado (imagen o binario) en repositorio.	DockerHub, Nexus, GitHub Packages
5. Deploy Stage	Despliegue automático en entorno de staging o producción.	Kubernetes, AWS EC2, Vercel, Netlify
6. Monitor Stage	Supervisión y retroalimentación sobre rendimiento y errores.	Prometheus, Grafana, ELK Stack, Sentry

7.4. Continuous Monitoring

7.4.1. Tools and Practices

Para el monitoreo de la aplicación se hace uso de un paquete que anteriormente se instaló. Roslynator nos permite tener reportes continuos acerca de cambios que debemos hacer manualmente y posibles mejoras dentro del código.



The screenshot shows a code editor interface with two tabs open: 'roslynator-report.txt' and 'roslynator-warnings.txt'. The left sidebar displays a project structure for 'BACCKEND-VACAPP' containing files like 'Program.cs', '.editorconfig', and various configuration and test files. The right pane shows the content of the 'roslynator-report.txt' file, which is an XML report generated by Roslynator. It lists numerous diagnostic findings with their IDs, titles, descriptions, and help links. Some examples include:

- ASPE0015**: Suggest using IHeaderDictionary properties
- CA1031**: Do not catch general exception types
- CA1062**: Validate arguments of public methods
- CA1822**: Mark members as static
- CA2007**: Consider calling ConfigureAwait on the awaited task
- CA2208**: Instantiate argument exceptions correctly
- CS0168**: Variable is declared but never used
- CS8602**: Dereference of a possibly null reference
- CS8603**: Possible null reference return
- CS8604**: Possible null reference argument
- CS8618**: Non-nullable field must contain a non-null value when exiting constructor
- CS8625**: Cannot convert null literal to non-nullable reference type

Asimismo, Security Code Scan alerta acerca de los warning y permite que la aplicación no se compile con estos errores para que sean resueltos antes de esta acción.

7.4.2. Monitoring Pipeline Components

Dentro del pipeline de Azure, durante la compilación se encuentran algunos puntos importantes que verifican el código. Dentro de estos se utilizan el `.editorconfig` y Roslynator:

```
# verify .editorconfig
  - script: dotnet format --verify-no-changes
    displayName: "Verify code format"

# Roslynator
  - script: |
    dotnet tool install --global roslynator.dotnet.cli --version 4.10.0
    export PATH="$PATH:/home/vsts/.dotnet/tools"
```

```
roslynator analyze . --severity-level warning --output roslynator-report.txt

displayName: "Run Roslynator analysis"
```

7.4.3. Alerting Pipeline Components

Las alertas del pipeline serán visibles una vez se escaneen las dependencias y Analyzer verifique las reglas compilando el código:

```
# Vulnerable dependencies

- script: dotnet list package --vulnerable > vulnerable-packages.txt || true

displayName: "Scan for vulnerable NuGet dependencies"
# Analyzer for security

- script: |

    dotnet build --no-restore --configuration $(buildConfiguration) \
    /p:TreatWarningsAsErrors=true \
    /p:AnalysisMode=AllEnabledByDefault

displayName: "Build with analyzers (Quality & Security)"
```

7.4.4. Notification Pipeline Components

Para el monitoreo de la aplicación gracias al pipeline, se toman las acciones de publicar reportes cada vez que se compila:

```
# publish reports

- task: PublishBuildArtifacts@1

inputs:

PathToPublish: 'roslynator-report.txt'
ArtifactName: 'analysis-reports'
publishLocation: 'Container'

displayName: "Publish Roslynator analysis report"
```

```
- task: PublishBuildArtifacts@1

  inputs:

    PathtoPublish: 'vulnerable-packages.txt'

    ArtifactName: 'dependency-scan'

    publishLocation: 'Container'

  displayName: "Publish dependency vulnerability report"

- task: PublishBuildArtifacts@1

  inputs:

    PathtoPublish: 'gitleaks-report.json'

    ArtifactName: 'security-scan'

    publishLocation: 'Container'

  displayName: "Publish secret scan report"
```

Capítulo VIII: Experiment-Driven Development

8.1. Experiment Planning

Esta sección define el plan operativo para diseñar y priorizar experimentos orientados a validar hipótesis clave del producto. Incluye objetivos concretos, métricas de éxito (KPIs), alcance, criterios de inclusión/exclusión, recursos necesarios y el calendario de ejecución; todo ello para asegurar que cada experimento entregue evidencia accionable y reduzca la incertidumbre sobre decisiones de producto.

8.1.1. As-Is Summary

Actualmente, la gestión del ganado en el Perú se realiza de manera manual o mediante herramientas rudimentarias como cuadernos y hojas de cálculo. La mayoría de ganaderos carece de plataformas digitales que integren el control sanitario, la alimentación, la reproducción y los indicadores productivos de sus animales. Esto genera una administración desorganizada, pérdida de información y baja productividad. Además, las empresas ganaderas no disponen de sistemas centralizados para coordinar personal, gestionar proveedores ni analizar datos en tiempo real, lo que limita la toma de decisiones estratégicas y la sostenibilidad del sector.

Problemas identificados:

- Gestión fragmentada: los procesos de registro y control del ganado se realizan manualmente o en formatos dispersos (cuadernos, Excel), dificultando la trazabilidad y el acceso a información precisa.

- Falta de digitalización: no existen herramientas accesibles que integren funciones clave como control sanitario, alimentación, reproducción y productividad.
- Desorganización operativa: los productores independientes enfrentan sobrecarga administrativa y pérdida de datos por falta de automatización.
- Limitaciones de conectividad: la baja cobertura de internet en zonas rurales impide el uso continuo de plataformas en línea.
- Escasa toma de decisiones basadas en datos: tanto pequeños ganaderos como empresas carecen de métricas centralizadas para evaluar rendimiento, costos o bienestar animal.
- Problemas de sostenibilidad: la ausencia de control y planificación impacta en la eficiencia económica y en la responsabilidad ambiental del sector.

Objetivos de mejora:

- Digitalizar la gestión ganadera: desarrollar una plataforma web y móvil que permita registrar, monitorear y analizar datos sobre salud, alimentación y reproducción del ganado.
- Centralizar la información: unificar registros y reportes en un solo sistema, accesible en todo momento y desde distintos dispositivos.
- Optimizar la productividad: automatizar tareas rutinarias (alertas sanitarias, control de peso, calendario de vacunación) para reducir errores y ahorrar tiempo.
- Facilitar el trabajo en campo: incluir funcionalidades que operen en modo offline para zonas rurales con conectividad limitada.
- Promover la sostenibilidad: ofrecer indicadores y reportes que ayuden a implementar prácticas responsables con el medio ambiente.
- Fortalecer la toma de decisiones: integrar analítica de datos para que los ganaderos y empresas puedan evaluar su rendimiento y planificar estratégicamente.

8.1.2. Raw Material: Assumptions, Knowledge Gaps, Ideas, Claims

Assumptions

- Los ganaderos valorarán una plataforma única que centralice el control de salud, alimentación y reproducción del ganado, eliminando la dependencia de registros manuales o dispersos.
- Una interfaz simple e intuitiva permitirá que usuarios con poca experiencia tecnológica puedan gestionar su ganado sin requerir capacitación extensa.
- Las alertas automáticas sobre vacunas, alimentación o ciclos reproductivos reducirán las omisiones y mejorarán la salud general del ganado.
- Permitir el funcionamiento offline aumentará la adopción en zonas rurales con baja conectividad.
- Los usuarios confiarán más en la aplicación si esta garantiza la privacidad y seguridad de sus datos productivos.

- Mostrar reportes visuales y gráficos de rendimiento facilitará la comprensión del estado sanitario y productivo del hato.
- Incluir recordatorios programados para tareas rutinarias (vacunas, alimentación, control sanitario) aumentará la organización diaria.
- La personalización por tipo de ganado o tamaño de operación (pequeño productor o empresa) mejorará la experiencia y satisfacción del usuario.
- Ofrecer materiales educativos dentro de la app fortalecerá la fidelización y el aprendizaje de buenas prácticas ganaderas.

Knowledge Gaps

- ¿Qué nivel de adopción pueden alcanzar los ganaderos con baja alfabetización digital frente a una interfaz simple?
- ¿Qué frecuencia de alertas (diarias, semanales o mensuales) perciben los usuarios como más útil sin resultar intrusiva?
- ¿Qué tipo de reportes visuales (gráficos, indicadores, tablas) son más comprendidos y usados en la toma de decisiones?
- ¿Cómo afecta la funcionalidad offline a la frecuencia real de uso en zonas rurales?
- ¿Qué nivel de confianza genera el almacenamiento de datos en la nube entre los ganaderos peruanos?
- ¿Qué funciones consideran más prioritarias los diferentes segmentos (pequeños productores vs. empresas)?
- ¿Cuánto valoran los usuarios el acceso a contenido educativo o de asesoría técnica dentro de la app?
- ¿Qué barreras culturales o tecnológicas influyen más en la adopción digital en el sector ganadero?

Ideas

- Realizar pruebas de usabilidad con ganaderos de zonas rurales para validar la facilidad de uso y comprensión de la interfaz.
- Implementar tests A/B con distintos intervalos de notificaciones para identificar la frecuencia óptima.
- Diseñar prototipos de reportes visuales (gráficos, KPI, tablas) y medir cuál facilita mejor la toma de decisiones.
- Evaluar el uso de modo offline con métricas de acceso y sincronización de datos una vez conectados.
- Aplicar encuestas de percepción de seguridad para medir confianza en el almacenamiento de datos.
- Personalizar módulos según el tipo de usuario (independiente o empresa) y analizar su impacto en la retención.

- Integrar un centro de aprendizaje virtual dentro de la app con artículos y videos educativos.
- Medir la reducción del tiempo administrativo mediante la digitalización de registros y tareas automáticas.

Claims

- La digitalización del registro ganadero reducirá en un 40 % el tiempo destinado a tareas administrativas diarias.
- El uso de alertas y recordatorios automáticos disminuirá en al menos 30 % los errores u omisiones sanitarias.
- El funcionamiento offline aumentará la frecuencia de uso en zonas rurales en más del 50 %.
- Garantizar la privacidad y seguridad de datos incrementará la confianza del usuario en un 80 % según encuestas post-uso.
- Los reportes visuales facilitarán la toma de decisiones, mejorando la comprensión operativa en un 70 % de los usuarios.
- La personalización por tipo de usuario incrementará la satisfacción y retención en al menos 25 %.
- La inclusión de material educativo impulsará el uso continuo de la plataforma en un 20 % durante los primeros meses de adopción.

8.1.3. Experiment-Ready Questions

Pregunta	Confianza	Riesgo	Impacto	Interés	Puntaje Total
¿Aumentará la productividad de los ganaderos si se digitaliza el registro de salud, alimentación y reproducción del ganado?	8 – La digitalización ha demostrado mejorar la eficiencia en sectores agrícolas y pecuarios.	3 – Riesgo técnico moderado; depende de una correcta usabilidad.	9 – Reduce errores, tiempos administrativos y pérdidas económicas.	8 – Alta demanda en entrevistas con productores y empresas.	28

Pregunta	Confianza	Riesgo	Impacto	Interés	Puntaje Total
¿Mejorará la gestión diaria si se implementan alertas y recordatorios automáticos en apps de control y salud, vacunación y alimentación?	8 –	2 – Riesgo bajo; se basa en programación de eventos.	8 – Evita omisiones y mejora el bienestar animal.	7 – Interés recurrente en entrevistas con ganaderos.	25
¿Aumentará la adopción en zonas rurales si la app permite funcionar sin conexión (modo offline)?	7 –	4 – Riesgo técnico medio por sincronización de datos.	8 – Garantiza accesibilidad y continuidad operativa.	8 – Muy solicitado por usuarios rurales.	27
¿Fortalecerá la confianza de los usuarios si la aplicación garantiza la seguridad y privacidad de los datos ganaderos?	9 – La seguridad de datos es un factor clave de confianza digital.	3 – Riesgo técnico moderado por manejo de credenciales y cifrado.	8 – Genera fidelización y uso sostenido de la plataforma.	7 – Alta preocupación expresada en entrevistas.	27
¿Facilitará la toma de decisiones incluir reportes visuales con indicadores de salud y producción?	8 – Los dashboards y visualizaciones mejoran la comprensión en la mayoría de sectores.	3 – Riesgo técnico bajo; requiere diseño adecuado.	8 – Mejora el control y análisis productivo.	6 – Interés alto, especialmente en empresas ganaderas.	25

Pregunta	Confianza	Riesgo	Impacto	Interés	Puntaje Total
¿Aumentará la satisfacción del usuario si la interfaz se adapta al tipo de perfil (pequeño productor o empresa)?	7 – La personalización mejora la experiencia en soluciones SaaS.	4 – Riesgo medio; requiere gestión de roles y configuraciones.	7 – Aumenta la eficiencia y retención.	7 – Interés creciente entre ambos segmentos.	25
¿Fomentará el aprendizaje continuo incluir materiales educativos dentro de la aplicación?	6 – Recursos educativos fortalecen la retención y confianza del usuario.	2 – Riesgo bajo; solo requiere curación de contenido.	6 – Mejora la adopción y profesionalización del usuario.	6 – Interés medio según entrevistas.	20
¿Reducirá la carga administrativa integrar un sistema de recordatorios y tareas automáticas?	8 – La automatización reduce tareas repetitivas y errores humanos.	3 – Riesgo bajo; depende de la interfaz de programación.	8 – Ahorra tiempo y mejora la organización.	7 – Interés constante en entrevistas de productores independientes.	26
¿Aumentará la confianza y el uso de la app si ofrece un soporte técnico accesible y rápido?	7 – El soporte eficiente es determinante para usuarios nuevos.	3 – Riesgo organizacional moderado (recursos humanos).	7 – Mejora la satisfacción y evita el abandono.	6 – Interés relevante entre usuarios con baja alfabetización digital.	23

8.1.4. Question Backlog

Prioridad (1, 2, 3, 5, 8)	Pregunta
8	¿Aumentará la productividad de los ganaderos si se digitaliza el registro de salud, alimentación y reproducción del ganado?

Prioridad (1, 2, 3, 5, 8)	Pregunta
8	¿Aumentará la adopción en zonas rurales si la app permite funcionar sin conexión (modo offline)?
5	¿Mejorará la gestión diaria si se implementan alertas y recordatorios automáticos de vacunación y alimentación?
5	¿Fortalecerá la confianza de los usuarios si la aplicación garantiza la seguridad y privacidad de los datos ganaderos?
5	¿Facilitará la toma de decisiones incluir reportes visuales con indicadores de salud y producción?
3	¿Aumentará la satisfacción del usuario si la interfaz se adapta al tipo de perfil (pequeño productor o empresa)?
3	¿Reducirá la carga administrativa integrar un sistema de recordatorios y tareas automáticas?
2	¿Fomentará el aprendizaje continuo incluir materiales educativos dentro de la aplicación?
2	¿Aumentará la confianza y el uso de la app si ofrece un soporte técnico accesible y rápido?

8.1.5. Experiment Cards

Question

¿Aumentará la productividad de los ganaderos si se digitaliza el registro de salud, alimentación y reproducción del ganado? | Why | Actualmente, los ganaderos registran la información de forma manual o dispersa (cuadernos, hojas de cálculo), lo que ocasiona pérdida de datos y baja eficiencia. | What | Desarrollar un módulo centralizado que permita registrar y consultar datos de salud, alimentación y reproducción del ganado desde la app web o móvil. | Hypothesis | Se espera una reducción del 40% en el tiempo de gestión diaria y una mejora del 30% en la precisión de los registros al implementar la digitalización completa.

Question

¿Mejorará la gestión diaria si se implementan alertas y recordatorios automáticos de vacunación y alimentación? | Why | Los ganaderos suelen olvidar fechas de vacunación o alimentación, afectando la salud del ganado y la productividad. | What | Incorporar notificaciones automáticas configurables que alerten sobre próximas vacunas, controles sanitarios y horarios de alimentación. | Hypothesis | Se espera una disminución del 25% en omisiones sanitarias y un aumento del 20% en cumplimiento de cronogramas gracias a las alertas automatizadas.

Question

¿Aumentará la adopción en zonas rurales si la app permite funcionar sin conexión (modo offline)? | Why | Gran parte de los usuarios potenciales se encuentran en zonas con conectividad limitada, lo que dificulta el uso continuo de la aplicación. | What | Desarrollar una funcionalidad offline que permita el registro y consulta local de datos, sincronizándose automáticamente cuando haya conexión. | Hypothesis | Se espera un incremento del 50% en la frecuencia de uso en zonas rurales y una reducción de quejas por conectividad en al menos un 40%.

Question

¿Fortalecerá la confianza de los usuarios si la aplicación garantiza la seguridad y privacidad de los datos ganaderos? | Why | Los productores temen perder o exponer información sensible relacionada con su productividad o inventario. | What | Implementar cifrado de datos, autenticación segura y políticas de privacidad transparentes dentro de la app. | Hypothesis | Se espera que el 80% de los usuarios manifieste mayor confianza y que la retención aumente en un 25% tras reforzar la seguridad.

Question

¿Facilitará la toma de decisiones incluir reportes visuales con indicadores de salud y producción? | Why | Los ganaderos carecen de herramientas visuales para analizar tendencias o comparar rendimientos a lo largo del tiempo. | What | Desarrollar paneles con gráficos e indicadores clave sobre producción, salud y eficiencia del ganado. | Hypothesis | Se espera una mejora del 70% en la comprensión operativa y un 30% de incremento en decisiones basadas en datos entre los usuarios activos.

Question

¿Aumentará la satisfacción del usuario si la interfaz se adapta al tipo de perfil (pequeño productor o empresa)? | Why | Las necesidades de gestión difieren entre productores independientes y empresas ganaderas; una interfaz única puede no cubrir ambos casos. | What | Diseñar perfiles personalizables con módulos y vistas específicas según el tipo de usuario registrado. | Hypothesis | Se estima un incremento del 25% en la satisfacción general y una reducción del 15% en abandonos durante el uso inicial.

Question

¿Fomentará el aprendizaje continuo incluir materiales educativos dentro de la aplicación? | Why | Muchos ganaderos no tienen acceso fácil a capacitación técnica o información actualizada sobre buenas prácticas. | What | Integrar una sección educativa con artículos, videos y tips sobre salud animal, alimentación y sostenibilidad. | Hypothesis | Se espera un 20% de incremento en el uso recurrente de la aplicación y una mayor percepción de valor entre nuevos usuarios.

Question

¿Reducirá la carga administrativa integrar un sistema de recordatorios y tareas automáticas? | Why | Las tareas diarias (pesaje, control sanitario, alimentación) requieren mucho tiempo y son propensas a errores si se gestionan manualmente. | What | Añadir un módulo de tareas programadas con recordatorios automáticos y confirmación de cumplimiento. | Hypothesis | Se espera una reducción del 30% en el tiempo dedicado a tareas rutinarias y una disminución del 25% en registros incompletos.

Question

¿Aumentará la confianza y el uso de la app si ofrece un soporte técnico accesible y rápido? | Why | Muchos usuarios rurales tienen poca experiencia con herramientas digitales y abandonan plataformas por falta de asistencia. | What | Incorporar un canal de soporte en línea con chat o mensajes directos para resolver dudas y brindar acompañamiento. | Hypothesis | Se espera una mejora del 40% en la retención de usuarios nuevos y una reducción del 20% en tickets no resueltos durante el primer mes de uso.

8.2. Experiment Design

8.2.1. Hypotheses

8.2.2. Domain Business Metrics

8.2.3. Measures

8.2.4. Conditions

8.2.5. Scale Calculations and Decisions

8.2.6. Methods Selection

8.2.7. Data Analytics: Goals, KPIs and Metrics Selection

8.2.8. Web and Mobile Tracking Plan

8.3. Experimentation

Aquí se documenta la ejecución práctica de los experimentos: implementación, recogida de datos, monitoreo y análisis de resultados frente a los KPIs definidos. Describe los procedimientos de control (grupos de control, muestreo), las herramientas de medición, el tratamiento de datos y los criterios de decisión para iterar, escalar o abandonar hipótesis según la evidencia obtenida.

8.3.1. To-Be User Stories

User Story ID	Título	Descripción	Criterios de Aceptación	Relación con (Epic ID)

User Story ID	Título	Descripción	Criterios de Aceptación	Relación con (Epic ID)
UA01	Digitalización de registros ganaderos	Como ganadero, quiero registrar digitalmente la información de salud, alimentación y reproducción de mi ganado, para mantener un control centralizado y evitar pérdida de datos.	<p>Escenario 1: Registrar información sanitaria Given tengo un animal registrado en mi cuenta. When ingreso la información de vacunación o tratamiento. Then el sistema guarda el registro y actualiza su historial sanitario.</p> <p>Escenario 2: Consultar historial Given deseo revisar el historial de un animal. When accedo a su ficha. Then puedo ver todos los registros de salud, alimentación y reproducción.</p>	EP01

User Story ID	Título	Descripción	Criterios de Aceptación	Relación con (Epic ID)
UA02	Alertas y recordatorios automáticos	Como ganadero, quiero recibir alertas automáticas sobre vacunas, alimentación y controles sanitarios, para no olvidar fechas importantes y mantener la salud de mi ganado.	<p>Escenario 1: Generar recordatorio automático Given he registrado una fecha de vacunación. When se aproxima la fecha configurada. Then el sistema envía una notificación al usuario con la alerta.</p> <p>Escenario 2: Confirmar cumplimiento Given he recibido una alerta. When realizo la acción (vacunar, alimentar, controlar). Then puedo marcarla como completada y el sistema actualiza el estado.</p>	EP02

User Story ID	Título	Descripción	Criterios de Aceptación	Relación con (Epic ID)
UA03	Modo offline para zonas rurales	Como usuario en una zona con poca conectividad, quiero poder usar la aplicación sin internet, para registrar y consultar datos sin depender de la red.	<p>Escenario 1: Registrar datos sin conexión</p> <p>Given no tengo conexión a internet.</p> <p>When registro nueva información del ganado.</p> <p>Then el sistema la guarda localmente en el dispositivo.</p> <p>Escenario 2: Sincronizar datos</p> <p>Given he recuperado la conexión.</p> <p>When el sistema detecta acceso a la red.</p> <p>Then sincroniza automáticamente la información local con la base de datos.</p>	EP03

User Story ID	Título	Descripción	Criterios de Aceptación	Relación con (Epic ID)
UA04	Seguridad y privacidad de datos	Como usuario, quiero que mi información ganadera esté protegida, para confiar en que solo yo y las personas autorizadas podemos acceder a ella.	<p>Escenario 1: Acceso seguro</p> <p>Given ingreso mis credenciales.</p> <p>When intento acceder al sistema.</p> <p>Then el sistema valida mi identidad mediante autenticación segura.</p> <p>Escenario 2: Encriptación de datos</p> <p>Given el sistema almacena información sensible.</p> <p>When se guarda en la base de datos.</p> <p>Then los datos se encriptan y solo pueden ser leídos por el usuario autorizado.</p>	EP01

User Story ID	Título	Descripción	Criterios de Aceptación	Relación con (Epic ID)
UA05	Reportes visuales de salud y productividad	Como ganadero, quiero ver reportes visuales con indicadores de salud, producción y alimentación, para tomar decisiones más informadas.	<p>Escenario 1: Generar reporte visual</p> <p>Given tengo datos registrados de mi ganado.</p> <p>When accedo al panel de control.</p> <p>Then visualizo gráficos con indicadores de salud, producción y tendencias.</p> <p>Escenario 2: Exportar reporte</p> <p>Given deseo conservar los datos.</p> <p>When selecciono "Exportar reporte".</p> <p>Then obtengo un archivo PDF o Excel con la información visualizada.</p>	EP02

User Story ID	Título	Descripción	Criterios de Aceptación	Relación con (Epic ID)
UA06	Interfaz adaptable por tipo de usuario	Como usuario (pequeño productor o empresa), quiero que la aplicación se adapte a mis necesidades, para visualizar solo las funciones relevantes a mi tipo de perfil.	<p>Escenario 1: Configurar tipo de usuario</p> <p>Given estoy creando mi cuenta. When selecciono "Pequeño productor" o "Empresa". Then el sistema adapta la interfaz y los módulos disponibles.</p> <p>Escenario 2: Cambiar configuración</p> <p>Given ya tengo una cuenta activa. When deseo cambiar mi tipo de usuario. Then puedo hacerlo desde el panel de configuración y el sistema actualiza mis vistas.</p>	EP03

User Story ID	Título	Descripción	Criterios de Aceptación	Relación con (Epic ID)
UA07	Módulo educativo integrado	Como usuario, quiero acceder a material educativo sobre buenas prácticas ganaderas, para mejorar mis conocimientos y la gestión de mi ganado.	<p>Escenario 1: Consultar contenido educativo Given accedo al módulo "Aprende con VacApp". When selecciono una categoría (salud, alimentación, sostenibilidad). Then puedo ver artículos, videos o consejos relacionados.</p> <p>Escenario 2: Guardar contenido Given estoy visualizando un material. When hago clic en "Guardar". Then el contenido se almacena en mi lista personal para consultar luego.</p>	EP04

User Story ID	Título	Descripción	Criterios de Aceptación	Relación con (Epic ID)
UA08	Automatización de tareas y recordatorios	Como usuario, quiero que el sistema programe automáticamente tareas repetitivas, para reducir la carga administrativa y evitar olvidos.	<p>Escenario 1: Crear tarea automática</p> <p>Given realizo frecuentemente una acción (pesaje, vacunación, alimentación).</p> <p>When configuro su frecuencia.</p> <p>Then el sistema genera automáticamente recordatorios para cada evento.</p> <p>Escenario 2: Notificación de tarea pendiente</p> <p>Given tengo tareas programadas.</p> <p>When se aproxima la fecha.</p> <p>Then el sistema me notifica para ejecutarlas.</p>	EP02

User Story ID	Título	Descripción	Criterios de Aceptación	Relación con (Epic ID)
UA09	Soporte técnico accesible	Como usuario con poca experiencia tecnológica, quiero acceder fácilmente a soporte o asistencia, para resolver dudas y continuar usando la aplicación.	<p>Escenario 1: Contactar soporte desde la app</p> <p>Given tengo un problema. When hago clic en "Centro de ayuda". Then puedo iniciar un chat o enviar un mensaje al equipo de soporte.</p> <p>Escenario 2: Seguimiento de incidencia</p> <p>Given he enviado una solicitud. When el equipo responde. Then recibo una notificación y puedo ver el estado de mi caso.</p>	EP04

8.3.2. To-Be Product Backlog

#	User Story ID	Título	Story Points (1/2/3/5/8)
1	UA01	Digitalización de registros ganaderos	8
2	UA02	Alertas y recordatorios automáticos	5
3	UA03	Modo offline para zonas rurales	8
4	UA04	Seguridad y privacidad de datos	5
5	UA05	Reportes visuales de salud y productividad	3
6	UA06	Interfaz adaptable por tipo de usuario	3
7	UA07	Módulo educativo integrado	2
8	UA08	Automatización de tareas y recordatorios	5
9	UA09	Soporte técnico accesible	2

Conclusiones

1. VacApp como ejemplo de arquitectura modular y escalable:

La aplicación VacApp evidencia cómo el uso de **Domain-Driven Design (DDD)** y la definición clara de bounded contexts permiten construir soluciones robustas y escalables para la gestión ganadera. La integración de módulos como *Campaign Management*, *Ranch Management* y *Staff Administration* ha facilitado una arquitectura limpia, adaptable y alineada con las necesidades reales de los usuarios, asegurando la evolución continua del producto.

2. Impacto del enfoque centrado en el usuario y metodologías ágiles:

El desarrollo de VacApp se caracterizó por la constante interacción con los usuarios finales, empleando entrevistas, mapeos de escenarios, user stories y análisis de impacto. El uso de metodologías ágiles como *Scrum* y herramientas como *Lean UX* permitió identificar y priorizar necesidades reales, logrando entregas funcionales frecuentes y una mejora continua en la experiencia del usuario ganadero.

3. Fortalecimiento de competencias técnicas y colaboración efectiva:

El trabajo colaborativo en VacApp no solo resultó en un producto funcional, sino que también impulsó el desarrollo de habilidades clave en el equipo, como el diseño de arquitecturas por capas, modelado de bases de datos, diseño de interfaces y aplicación de patrones estratégicos y tácticos de DDD. La gestión eficiente del proyecto y la comunicación constante consolidaron una visión profesional y una cultura de mejora continua dentro del equipo.

4. Integración continua como eje de calidad y automatización:

La implementación de **pipelines de integración continua (CI)** permitió automatizar la construcción, pruebas y verificación del código en cada commit. Gracias al uso de herramientas como *GitHub Actions* y *xUnit*, el equipo logró detectar errores tempranos, mantener la estabilidad del sistema y asegurar que cada nueva funcionalidad se integre sin comprometer la calidad global del producto.

5. Entrega continua para asegurar despliegues confiables:

El enfoque de **Continuous Delivery (CD)** posibilitó una transición fluida entre los entornos de desarrollo, prueba y producción, garantizando que las nuevas versiones de VacApp pudieran ser liberadas de forma controlada y predecible. La incorporación de pruebas automatizadas dentro del pipeline redujo tiempos de validación y mejoró la trazabilidad de cada versión entregada.

6. Despliegue continuo como práctica de evolución constante:

La aplicación de estrategias de **Continuous Deployment** consolidó la visión DevOps del proyecto, asegurando que cada mejora validada automáticamente se desplegará en producción sin intervención manual. Este enfoque favoreció la agilidad operativa, la reducción de riesgos y la entrega continua de valor hacia los usuarios finales, posicionando a VacApp como una plataforma moderna, mantenible y en evolución permanente.

Bibliografía

- Cohn, M. (2004). *User Stories Applied: For Agile Software Development*. Addison-Wesley.
- Evans, E. (2004). *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Addison-Wesley.
- Evans, E. (2015). *Domain-Driven Design Reference: Definitions and Pattern Summaries*. Domain Language, Inc. <https://www.domainlanguage.com/ddd/reference/>
- Fowler, M. (2003). *Patterns of Enterprise Application Architecture*. Addison-Wesley.
- Gothelf, J. (2013). *Lean UX: Applying Lean Principles to Improve User Experience*. O'Reilly Media.
- ISO/IEC/IEEE 12207:2017 – Systems and software engineering – Software life cycle processes.
- Poppendieck, M., & Poppendieck, T. (2003). *Lean Software Development: An Agile Toolkit*. Addison-Wesley.
- Vernon, V. (2013). *Implementing Domain-Driven Design*. Addison-Wesley.

Anexos

Enlaces de la Aplicación

Aplicación Móvil: <https://appdistribution.firebaseio.dev/i/b5b8b0a89363391d>

Videos del Proyecto

About The Product: <https://youtu.be/JmOW2lkXjeI>

Recursos de Diseño

Mockups en Figma: <https://www.figma.com/design/Ck5RdO3MzAm16SIReLDO15/Sin-t%C3%ADtulo?node-id=150-5796&t=hGN3YL7RfASQ5FFk-1>

Repositorios del Proyecto

Documentación: <https://github.com/1ASI0732-Grupo-3/Documento---VaCowTeam>

Backend: <https://github.com/1ASI0732-Grupo-3/Bacckend-VacApp>

Aplicación Móvil: <https://github.com/1ASI0732-Grupo-3/Mobile--VacApp>

Landing Page: <https://github.com/1ASI0732-Grupo-3/Landing-Page---VacApp>

Enlaces de Implementación (TB1)

Landing Page: <https://vacapp-landing.netlify.app/>