

# UNIVERSIDAD DEL VALLE DE GUATEMALA

## Facultad de Ingeniería



**Arquitectura e implementación de estrategias, prácticas y políticas de ciberseguridad y aseguramiento de calidad para una plataforma móvil de gestión administrativa de formularios de recolección de datos en un ingenio azucarero en Guatemala**

Trabajo de graduación en la modalidad de Trabajo Profesional presentado por Emilio José Solano Orozco para optar al grado académico de Licenciado en Ingeniería en Ciencia de la Computación y Tecnologías de la Información

Guatemala,

2025



# UNIVERSIDAD DEL VALLE DE GUATEMALA

## Facultad de Ingeniería



**Arquitectura e implementación de estrategias, prácticas y políticas de ciberseguridad y aseguramiento de calidad para una plataforma móvil de gestión administrativa de formularios de recolección de datos en un ingenio azucarero en Guatemala**

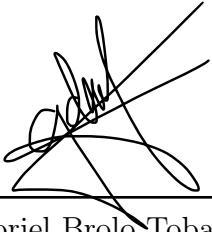
Trabajo de graduación en la modalidad de Trabajo Profesional  
presentado por Emilio José Solano Orozco para optar al grado  
académico de Licenciado en Ingeniería en Ciencia de la Computación y  
Tecnologías de la Información

Guatemala,

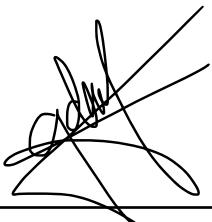
2025



Vo.Bo.:

(f)   
Ing. Gabriel Brolo Tobar

Tribunal Examinador:

(f)   
Ing. Gabriel Brolo Tobar

(f)   
Ing. Marlon Osiris Fuentes Lopez

Fecha de aprobación: Guatemala, \_\_\_\_\_ de \_\_\_\_\_ de 2025.



---

## Prefacio

---

El proyecto desarrollado con el Ingenio Santa Ana surge de la iniciativa institucional de colaborar con estudiantes de la Universidad del Valle de Guatemala para explorar soluciones innovadoras en software y tecnologías de la información. Bajo este marco, se identificó la oportunidad de optimizar la elaboración de formularios en campo y se diseñó el módulo que aquí se presenta.

Este trabajo se circunscribe al diseño e implementación de un módulo de formularios offline-first y a su validación técnica y con usuarios en un entorno controlado; no pretende abarcar despliegues productivos a gran escala ni la integración con todos los sistemas legados del Ingenio. Los resultados discutidos reflejan las condiciones de las pruebas realizadas y el alcance definido para esta fase.

Agradezco al Ingenio Santa Ana por su apertura en instalaciones y transferencia de conocimiento, y a la Universidad del Valle de Guatemala por la formación recibida. Agradezco profundamente a mis padres, Frida Eugenia Orozco Barrios de Solano y Ángel Emilio Solano Ruiz, por su apoyo incondicional; a mi hermana, María Clara Solano Orozco; a la familia Fuentes Orozco; y a mis amistades y compañeros de la universidad que colaboraron con el Ingenio Santa Ana —Diego Hernández, Linda Jiménez, Mario Guerra, Javier Alvarado, Andrea Ramírez y Adrián Flores— por su ayuda cotidiana. Extiendo mi gratitud a Daniel Valdez por su apoyo diario y su amistad; a mi asesor Gabriel Brolo Tobar por su consejo y directriz; a Erick Marroquín por su guianza en este último año; y a amistades entrañables como Diana Fernández, Elías Alvarado, Adrián Fulladolsa, Jennifer Toxcón, David Aragón, José Auyón y Eunice Mata, por su apoyo y compañía. Finalmente, un agradecimiento para Emily Elvia Melissa Pérez Alarcón, por su apoyo desde el cariño, la empatía y la confianza plena.

Dedico este trabajo a la memoria de mis abuelos Celeste Aurora Ruiz Polanco, Telma Eugenia Barrios Schaub de Orozco y Jaime Humberto Orozco Joaquín, presentes siempre en mi pensamiento y mi corazón.

A Dios y a la Virgen Santísima, por su guía y bendición abundante a lo largo de mi vida.



---

## Índice

---

<b>Prefacio</b>	<b>V</b>
<b>Lista de figuras</b>	<b>XI</b>
<b>Lista de cuadros</b>	<b>XIII</b>
<b>Lista de abreviaturas y siglas</b>	<b>XV</b>
<b>Resumen</b>	<b>XIX</b>
<b>Abstract</b>	<b>XXI</b>
<b>1. Introducción</b>	<b>1</b>
<b>2. Antecedentes</b>	<b>3</b>
<b>3. Justificación</b>	<b>5</b>
<b>4. Objetivos</b>	<b>7</b>
4.1. Objetivo general . . . . .	7
4.2. Objetivos específicos . . . . .	7
<b>5. Alcance</b>	<b>9</b>
5.1. Ámbito funcional . . . . .	9
5.2. Ámbito técnico . . . . .	9
5.3. Resultados e indicadores esperados . . . . .	10
5.4. Entregables . . . . .	10
5.5. Actores y contexto . . . . .	10
5.6. Supuestos y restricciones . . . . .	11
5.7. Fuera de alcance . . . . .	11
5.8. Criterios de aceptación . . . . .	11

<b>6. Marco teórico</b>	<b>13</b>
6.1. Arquitectura <i>offline-first</i> y sincronización de datos . . . . .	13
6.2. Dispositivos personales, gestión móvil y modelos de confianza . . . . .	14
6.3. Seguridad por diseño, ciclo de vida de desarrollo y cadena de suministro . . . . .	15
6.4. Modelado de amenazas en aplicaciones de datos . . . . .	16
6.5. Calidad, usabilidad y medición de la experiencia . . . . .	17
6.5.1. Aseguramiento de calidad y automatización de pruebas . . . . .	17
6.5.2. Usabilidad de formularios y escala SUS . . . . .	18
6.6. Privacidad, gobernanza de datos y dependencia de proveedores . . . . .	18
6.7. Datos de formularios y analítica operativa . . . . .	19
<b>7. Metodología</b>	<b>21</b>
7.1. Enfoque y diseño metodológico . . . . .	21
7.2. Área de estudio y población . . . . .	21
7.3. Fases y cronograma . . . . .	22
7.4. Materiales . . . . .	23
7.5. Métodos . . . . .	24
7.5.1. Seguridad en backend ( <i>pre/post</i> ) . . . . .	24
7.5.2. Usabilidad en frontend (pruebas moderadas por tareas) . . . . .	24
7.5.3. QA automatizado (regresión) . . . . .	25
7.5.4. Validación y satisfacción . . . . .	25
7.6. Pruebas de resiliencia y conectividad . . . . .	25
7.7. Métricas, análisis y validez . . . . .	26
<b>8. Resultados</b>	<b>27</b>
8.1. Diagnóstico del estado actual y propuesta . . . . .	27
8.2. Seguridad del backend (DAST con OWASP ZAP) . . . . .	28
8.2.1. Alcance y contexto del escaneo . . . . .	28
8.2.2. Severidad alta: CORS mal configurado . . . . .	28
8.2.3. Severidad media: cabeceras de clickjacking y CSP ausentes; disclosure de proxy . . . . .	29
8.2.4. Severidad baja e informativa: cabeceras endurecedoras y mensajes de error . . . . .	29
8.2.5. Discusión frente a los objetivos . . . . .	30
8.2.6. Validez y limitaciones . . . . .	30
8.3. Usabilidad del frontend . . . . .	30
8.3.1. Alcance y método . . . . .	30
8.3.2. Resultados: efectividad, eficiencia y satisfacción . . . . .	31
8.3.3. Evidencia de calidad a partir de pruebas automatizadas . . . . .	31
8.3.4. Discusión y validez . . . . .	32
8.4. QA automatizado con Appium . . . . .	32
8.4.1. Alcance y método . . . . .	32
8.4.2. Configuración técnica . . . . .	32
8.4.3. Ejecución y evidencia . . . . .	32
8.4.4. Discusión frente a objetivos . . . . .	33
8.5. Análisis estático con Snyk . . . . .	33
8.6. Resultados de la Escala de Usabilidad del Sistema (SUS) . . . . .	35

<b>9. Conclusiones</b>	<b>39</b>
<b>10. Recomendaciones</b>	<b>41</b>
<b>11. Bibliografía</b>	<b>43</b>



---

## Lista de figuras

---

1.	Confirmación de pruebas con Appium exitosas . . . . .	33
2.	Listado de vulnerabilidades en Snyk API/Web . . . . .	34
3.	Resultados de escaneo y focalización de <i>targets</i> en Snyk API/Web . . . . .	34
4.	Uso confrecuencia de la aplicación . . . . .	35
5.	Complejidad innecesaria de la aplicación . . . . .	35
6.	Percepción de facilidad después de la navegación . . . . .	36
7.	Necesidad de asistencias durante navegación . . . . .	36
8.	Correcta integración de diversidad de funciones . . . . .	36
9.	Pensamiento sobre inconsistencia en la aplicación . . . . .	36
10.	Velocidad de aprendizaje hacia la aplicación . . . . .	37
11.	Percepción en cuanto a una aplicación complicada . . . . .	37
12.	Seguridad al utilizar la aplicación . . . . .	37
13.	Requerimientos previos a la navegación por la aplicación . . . . .	37



---

## Lista de cuadros

---

2.	Resumen de severidades detectadas por ZAP . . . . .	28
3.	Alertas detectadas por ZAP en el backend y número de instancias . . . . .	29
4.	Ejemplos de endpoints asociados a <i>Proxy Disclosure</i> . . . . .	30
5.	Cobertura en módulos críticos . . . . .	31



---

## Lista de abreviaturas y siglas

---

<b>Sigla</b>	<b>Significado</b>
ACM	Association for Computing Machinery.
AES	Advanced Encryption Standard.
API	Application Programming Interface (interfaz de programación de aplicaciones).
AST	Application Security Testing (pruebas de seguridad de aplicaciones).
ATS	App Transport Security (política de seguridad de transporte en iOS/iPadOS).
ATT&CK	Adversarial Tactics, Techniques and Common Knowledge (marco de tácticas y técnicas de ataque de MITRE).
BI	Business Intelligence (inteligencia de negocios).
BYOD	Bring Your Own Device (trae tu propio dispositivo).
CD	Continuous Delivery / Continuous Deployment (entrega / despliegue continuo).
CDN	Content Delivery Network (red de entrega de contenidos).
CI	Continuous Integration (integración continua).
CI/CD	Continuous Integration / Continuous Delivery (integración y entrega continua).
CIO	Chief Information Officer (director de tecnologías de información).
CORS	Cross-Origin Resource Sharing (compartición de recursos de origen cruzado).
CSP	Content Security Policy (política de seguridad de contenidos).
DAST	Dynamic Application Security Testing (pruebas dinámicas de seguridad de aplicaciones).
DNS	Domain Name System (sistema de nombres de dominio).
ENISA	European Union Agency for Cybersecurity (agencia de ciberseguridad de la Unión Europea).
EU	European Union (Unión Europea).
GDPR	General Data Protection Regulation (Reglamento General de Protección de Datos de la UE).

GET	Verbo HTTP GET (operación de lectura en APIs REST).
GRC	Governance, Risk and Compliance (gobierno, riesgo y cumplimiento).
HIPAA	Health Insurance Portability and Accountability Act (ley de portabilidad y responsabilidad de seguros de salud, EE. UU.).
HSTS	HTTP Strict Transport Security (seguridad estricta de transporte HTTP).
HTML	HyperText Markup Language (lenguaje de marcado de hipertexto).
HTTP	Hypertext Transfer Protocol (protocolo de transferencia de hipertexto).
HTTPS	Hypertext Transfer Protocol Secure (HTTP sobre TLS).
IBM	International Business Machines Corporation.
IEC	International Electrotechnical Commission (Comisión Electrotécnica Internacional).
IETF	Internet Engineering Task Force (grupo de trabajo de estandarización de Internet).
IP	Internet Protocol (protocolo de Internet).
ISO	International Organization for Standardization (Organización Internacional de Normalización).
IT	Information Technology (tecnologías de la información).
JSON	JavaScript Object Notation (notación de objetos de JavaScript).
JWT	JSON Web Token (token web en formato JSON).
MAC	Media Access Control (dirección de control de acceso al medio).
MAM	Mobile Application Management (gestión de aplicaciones móviles).
MASTG	Mobile Application Security Testing Guide (guía de pruebas de seguridad de aplicaciones móviles de OWASP).
MASVS	Mobile Application Security Verification Standard (estándar de verificación de seguridad de aplicaciones móviles de OWASP).
MDM	Mobile Device Management (gestión de dispositivos móviles).
MDN	Mozilla Developer Network (portal de documentación para desarrolladores).
MITM	Man-in-the-Middle (ataque de intermediario).
MITRE	MITRE Corporation (organización de I+D en seguridad y tecnología).
NCSC	National Cyber Security Centre (centro nacional de ciberseguridad del Reino Unido).
NIST	National Institute of Standards and Technology (Instituto Nacional de Estándares y Tecnología de EE. UU.).
ODK	Open Data Kit (plataforma para formularios y captura de datos en campo).
OIDC	OpenID Connect (protocolo de autenticación basado en OAuth 2.0).
OWASP	Open Worldwide Application Security Project (proyecto abierto de seguridad de aplicaciones).
PFS	Perfect Forward Secrecy (secreto perfecto hacia adelante).
POST	Verbo HTTP POST (operación de creación en APIs REST).

PUT	Verbo HTTP PUT (operación de reemplazo/actualización en APIs REST).
QA	Quality Assurance (aseguramiento de la calidad).
RFC	Request for Comments (documentos de especificación y estandarización de Internet).
RGPD	Reglamento General de Protección de Datos (traducción al español de GDPR).
SAMM	Software Assurance Maturity Model (modelo de madurez de aseguramiento de software de OWASP).
SAST	Static Application Security Testing (pruebas estáticas de seguridad de aplicaciones).
SDL	Security Development Lifecycle (ciclo de vida de desarrollo seguro).
SIEM	Security Information and Event Management (gestión de eventos e información de seguridad).
SLSA	Supply-chain Levels for Software Artifacts (niveles de seguridad para la cadena de suministro de artefactos de software).
SP	Special Publication (serie de publicaciones especiales, por ejemplo NIST SP 800-xx).
SPSS	Statistical Package for the Social Sciences (paquete estadístico para ciencias sociales).
SQL	Structured Query Language (lenguaje de consulta estructurada).
SSDLC	Secure Software Development Life Cycle (ciclo de vida de desarrollo de software seguro).
SSL	Secure Sockets Layer (capa de sockets seguros).
STRIDE	Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege (mnemotecnia de modelado de amenazas).
SUS	System Usability Scale (escala de usabilidad del sistema).
TEE	Trusted Execution Environment (entorno de ejecución confiable).
TLS	Transport Layer Security (seguridad de la capa de transporte).
UI	User Interface (interfaz de usuario).
UK	United Kingdom (Reino Unido).
US	United States (Estados Unidos de América).
W3C	World Wide Web Consortium (consorcio de estandarización de la Web).
XML	eXtensible Markup Language (lenguaje de marcado extensible).
XSS	Cross-Site Scripting (inyección de scripts en sitios cruzados).
ZAP	Zed Attack Proxy (herramienta de OWASP para pruebas DAST).



---

## Resumen

---

El Ingenio Santa Ana depende de datos de campo oportunos para planificar labores agro-nómicas y decisiones operativas. Actualmente usa una plataforma de formularios de terceros, con costos de licenciamiento, exportaciones poco flexibles y poca visibilidad sobre la seguridad del *backend*. Este trabajo diseña y valida un módulo propio de gestión administrativa de formularios, con arquitectura *offline-first*, orientado a mejorar la seguridad del API, la usabilidad de la aplicación móvil y el aseguramiento de calidad automatizado.

Se siguió un enfoque mixto. En seguridad se ejecutó OWASP ZAP sobre doce *endpoints* del *backend*. No se detectaron vulnerabilidades críticas ni evidencias de inyección SQL o XSS persistente. Sí se encontró una alerta alta por configuración CORS y tres alertas medias por cabeceras de protección ausentes y divulgación de información del servidor. En paralelo, Snyk identificó dos hallazgos de severidad alta por secretos codificados en el repositorio y varias vulnerabilidades medias o bajas en dependencias, lo que permitió definir un plan de actualización y gestión de secretos.

En usabilidad, diez participantes completaron tareas críticas y la escala SUS, con un puntaje promedio de 83 puntos, superior al umbral de 68. En aseguramiento de calidad, las pruebas unitarias y tres flujos Appium (inicio de sesión, captura y sincronización diferida) lograron 100 % de éxito en el *pipeline* de integración continua. La solución resultante incrementa el control sobre el ciclo de vida del dato y establece indicadores cuantitativos para la mejora continua en seguridad, usabilidad y calidad dentro del ecosistema del ingenio.



---

## Abstract

---

Ingenio Santa Ana relies on timely field data to plan agronomic activities and operational decisions. At present, data capture uses a third-party forms platform, which entails licensing costs, rigid export formats and limited visibility into backend security. This work designs and validates an in-house administrative form management module with an *offline-first* architecture, aimed at improving API security, mobile application usability and automated quality assurance.

A mixed-methods approach was used. For security, OWASP ZAP was executed against twelve backend *endpoints*. No critical vulnerabilities or evidence of SQL injection or persistent XSS were found. One high-risk alert related to CORS configuration and three medium-risk alerts related to missing protection headers and server information disclosure were identified. In parallel, Snyk reported two high-severity findings due to hard-coded secrets in the repository and several medium- or low-severity vulnerabilities in dependencies, which informed an update and secret management plan.

For usability, ten participants completed critical tasks in the mobile application and the SUS scale, obtaining an average score of 83 points, above the threshold of 68. For quality assurance, available unit tests and three Appium flows (login, data capture and deferred synchronization) achieved 100 % success in the continuous integration *pipeline*. Overall, the solution increases control over the data life cycle and establishes quantitative indicators for continuous improvement in security, usability and quality within the mill's ecosystem.



# CAPÍTULO 1

---

## Introducción

---

Las operaciones del Ingenio Santa Ana dependen de datos de campo confiables para planificar labores agronómicas, gestionar recursos y evaluar el desempeño de la zafra. Actualmente, la captura se apoya en plataformas de terceros para formularios móviles, lo que ha permitido abandonar el papel pero también ha introducido una serie de problemas concretos: costos de licenciamiento y personalización, estructuras de exportación fuertemente anidadas que dificultan el análisis tabular, dependencia tecnológica de proveedores externos y visibilidad limitada sobre cómo se protegen y procesan los datos críticos del ingenio. Estas fricciones impactan los tiempos de análisis, aumentan el riesgo de errores en la transformación de datos y restan flexibilidad para adaptar los formularios al ritmo de la operación.

El problema que aborda este trabajo es, por tanto, doble. En primer lugar, el ingenio carece de una plataforma propia que combine operación offline-first con un modelo de datos alineado a sus necesidades analíticas, lo que limita la capacidad de explotar la información de campo en forma oportuna. En segundo lugar, los procesos actuales de desarrollo y pruebas no incorporan de manera sistemática controles de ciberseguridad ni automatización de QA, de modo que resulta difícil medir y mejorar la robustez del backend, la experiencia de uso en la aplicación móvil y la calidad de los formularios desplegados. El proyecto surge como respuesta a estas limitaciones y busca dotar al ingenio de una base técnica y metodológica sobre la cual pueda evolucionar su ecosistema de formularios.

En este contexto, la finalidad del trabajo es diseñar e implementar un módulo de gestión administrativa de formularios, con arquitectura offline-first, e integrar estrategias de seguridad y control de calidad que permitan fortalecer el backend, mejorar la usabilidad del frontend móvil y preservar la gobernanza del dato dentro de la organización. Para ello se delimita el alcance a un prototipo funcional y a su evaluación en un entorno controlado, a través de pruebas dinámicas de seguridad (DAST con OWASP ZAP y análisis estático con Snyk sobre el API), pruebas de usabilidad moderadas apoyadas en la Escala de Usabilidad del Sistema (SUS) y *pipelines* de integración continua que ejecutan pruebas unitarias y flujos automatizados de Appium sobre recorridos críticos.

La hipótesis central de este trabajo establece que la implementación de un módulo pro-

pio de gestión administrativa de formularios con arquitectura offline-first, acompañado de controles sistemáticos de ciberseguridad y automatización de pruebas de calidad, permite alcanzar en un entorno controlado de pruebas tres metas cuantificables. Primero, ausencia de vulnerabilidades críticas conocidas en el API. Segundo, una usabilidad percibida al menos aceptable (puntaje SUS  $\geq 68$ ). Tercero, una tasa de éxito  $\geq 95\%$  en pruebas automatizadas sobre flujos funcionales críticos.

Metodológicamente se adopta un enfoque mixto que combina métricas cuantitativas (conteo y severidad de hallazgos de seguridad, puntaje SUS, tasas de éxito y *pass rate* de pruebas automatizadas) con evidencias cualitativas (observaciones moderadas y retroalimentación de usuarios) para contextualizar las decisiones de diseño. De manera sintética, los resultados muestran que la API escaneada no presenta vulnerabilidades críticas conocidas, que la interfaz móvil alcanza un puntaje SUS de 83 puntos —indicativo de buena usabilidad— y que los *pipelines* de pruebas ofrecen ejecuciones estables en flujos esenciales. Estos hallazgos aportan evidencia para valorar la viabilidad de la propuesta y abren una línea de trabajo para que el ingenio consolide prácticas de seguridad, usabilidad y QA orientadas a la mejora continua.

## CAPÍTULO 2

---

### Antecedentes

---

La digitalización de formularios se ha consolidado como una pieza clave de la transformación digital en organizaciones que requieren captura estructurada de datos en campo y trazabilidad de procesos. Su adopción reemplaza formatos en papel por interfaces electrónicas con validaciones, control de calidad y sincronización, lo que mejora la eficiencia operativa y la precisión de los registros (KoboToolbox, [2025]; “Welcome to ODK’s Docs”, [2025]; “XLSForm”, [2025]). En particular, ecosistemas como ODK y KoboToolbox habilitan trabajo *offline-first* (aplicación móvil y formularios web con almacenamiento local) y despliegue de formularios complejos mediante el estándar XLSForm, ampliamente documentado en la literatura técnica (“Form Design Introduction”, [2025]; “Overview on Data Collection Tools”, [2025]; “XLSForm tutorial: Your first form”, [2025]).

Este avance, sin embargo, introduce responsabilidades técnicas y organizativas en materia de privacidad y seguridad: principios como licitud, transparencia, minimización de datos, integridad/confidencialidad y limitación de conservación orientan el diseño de procesos y controles de acceso cuando se tratan datos personales o sensibles (académicos, laborales o de salud) (“Art. 5 GDPR — Principles relating to processing of personal data”, [2025]; Hoofnagle et al., [2019]; “Regulation (EU) 2016/679 (General Data Protection Regulation)”, [2016]). En consecuencia, un proyecto de formularios no solo es un *builder*, sino también una arquitectura segura, validada y evaluada con el usuario final.

El mercado ofrece alternativas con alcances diversos. *Microsoft Forms* integra recolección básica con exportación directa a Excel, útil cuando ya se opera en Microsoft 365 (“How do I export Forms results to Excel online?”, [2024]; “Microsoft Forms to Excel”, [2024]). *Typeform* provee una API de respuestas en JSON para consumo programático y analítica (“Responses API & JSON response explanation”, [2025]; “Typeform Developers — APIs”, [2025]). En el frente de cumplimiento, *Jotform* ofrece cuentas con características orientadas a HIPAA para flujos del sector salud (“How to enable HIPAA compliance”, [2025]; “Jotform HIPAA Compliance”, [2025]). Para operaciones de campo con necesidades empresariales (flujos offline, lógica dinámica, orquestación y tableros), plataformas como *FORM.com* complementan la captura con automatización y BI (“FORM OpX — Google Play (características y modo

offline)", [2025]; "Mobile Forms App — Features", [2025]).

Actualmente, el ingenio utiliza *Digiforms* mediante licencias de suscripción. Las fichas oficiales en App Store y Google Play confirman que la aplicación requiere una cuenta de suscripción y se apoya en un constructor web para desplegar formularios en dispositivos móviles ("Digiforms — App Store (Interlink Software, SA)", [2025]; "DigiForms — Google Play", [2025]). A nivel operativo, el equipo del ingenio ha identificado como dolor principal que, al exportar datos, la estructura llega *muy anidada*, lo cual dificulta el análisis tabular y la integración con reportes/BI.

Ese problema es común cuando las plataformas devuelven respuestas en JSON jerárquico—por ejemplo, las cargas útiles de la Responses API de Typeform—y exige una etapa de normalización previa ("flattening"). La documentación de `pandas.json_normalize` y guías técnicas reconocidas describen precisamente cómo aplanar datos semiestructurados hacia tablas analíticas ("All pandas json\_normalize() you should know for flattening JSON", [2025]; "pandas.json\_normalize — pandas documentation", [2025]; "Python Pandas — Flatten nested JSON", [2025]).

## CAPÍTULO 3

---

### Justificación

---

La toma de decisiones en un ingenio azucarero depende de datos de campo oportunos, íntegros y analizables. En la práctica actual, el uso de plataformas de terceros para formularios móviles (p. ej., Digiforms) introduce puntos de dolor claros: (i) **costos**, asociados a licenciamiento por usuario y a desarrollos específicos que encarecen la incorporación de nuevos formularios; (ii) **tiempos**, debido a exportaciones en JSON fuertemente anidado que requieren pasos manuales de normalización antes de poder explotar la información; (iii) **errores**, ya que cada transformación adicional abre espacio a fallos de mapeo, omisiones de campos o duplicidades; y (iv) **riesgos**, tanto por la dependencia tecnológica del proveedor (*vendor lock-in*) como por la limitada visibilidad sobre cómo se gestionan la seguridad y la privacidad de los datos críticos del ingenio. En este contexto, desarrollar una solución propia no es sólo una alternativa tecnológica, sino una respuesta directa a estos costos, demoras, errores operativos y riesgos de gobernanza del dato.

En el sector agroindustrial, la digitalización de procesos de captura en campo ha mostrado beneficios sistemáticos en eficiencia, trazabilidad y calidad de datos, habilitando decisiones más rápidas y precisas en cadenas de valor agrícolas (Trendov et al., 2019; World Bank, 2017). Para un ingenio con cuadrillas distribuidas y conectividad intermitente, el principio *offline-first* resulta esencial: permite operar sin red, conservar el rendimiento de la aplicación y sincronizar de forma confiable cuando hay señal (Android Developers, 2025b; Archibald et al., 2025; MDN Web Docs, 2025). La solución propuesta aborda los puntos de dolor anteriores mediante: (i) una aplicación móvil *offline-first* con validaciones en origen para reducir errores de captura; (ii) un backend propio, con cifrado extremo a extremo y controles de autenticación/autorización, que disminuye riesgos de exposición y dependencia externa (National Institute of Standards and Technology, 2020; Open Data Kit, 2025; OWASP Foundation, 2024); y (iii) un modelo de datos alineado a las necesidades de *analytics* del negocio, que simplifica la transformación hacia vistas tabulares o esquemas en estrella, reduciendo tiempo y errores en el *flattening* de estructuras anidadas (GeeksforGeeks, 2025; Google Cloud, 2020).

Si bien existen alternativas consolidadas para formularios (p. ej., ODK/KoBo, Fulcrum

o FastField) con capacidades robustas y recolección sin conexión (FastField, [2023]; Fulcrum, [2025]; KoboToolbox, [2025]), la realidad del Ingenio Santa Ana demanda control total del ciclo de vida del dato, integración estrecha con sistemas internos y eliminación del riesgo de *vendor lock-in*. La literatura y los marcos de referencia en nube advierten que la portabilidad e interoperabilidad deben planificarse explícitamente para mitigar dicho riesgo; construir una solución propia con estándares abiertos posiciona mejor al ingenio frente a esa amenaza y permite ajustar los formularios al ritmo operativo sin incurrir en tiempos ni costos adicionales de terceros (European Network and Information Security Agency [ENISA], [2009]; National Institute of Standards and Technology, [2013]; Opara-Martins et al., [2016]).

Finalmente, una plataforma desarrollada y operada por el ingenio facilita institucionalizar prácticas de aseguramiento de calidad (QA) y pruebas automatizadas en *frontend* y *backend*, algo que las soluciones empaquetadas rara vez ofrecen a la medida del contexto local. La evidencia empírica vincula la automatización de pruebas y las capacidades de entrega continua con mejoras en desempeño de equipos, menor densidad de defectos y ciclos de entrega más cortos (Forsgren et al., [2018a], [2018b]; Rafi et al., [2012]). En este proyecto, dichas prácticas se traducen en indicadores concretos para medir la mejora: en seguridad, el porcentaje de reducción de hallazgos críticos/altos en escaneos DAST y de dependencias; en usabilidad, la tasa de éxito y tiempos medios de tarea, así como el puntaje SUS; en QA, el *pass rate* de regresión y las causas de falla; y en satisfacción, el porcentaje de usuarios que reportan una experiencia positiva en campo. Contar con estos indicadores y con una arquitectura propia fortalece la gobernanza del dato, reduce la dependencia de terceros y proporciona una base sólida para la mejora continua de los formularios y de la operación digital del ingenio.

# CAPÍTULO 4

---

## Objetivos

---

### 4.1. Objetivo general

Implementar estrategias de seguridad y control de calidad de una plataforma administrativa móvil para optimizar la gestión de formularios en un ingenio azucarero, asegurando la integridad y accesibilidad de la información.

### 4.2. Objetivos específicos

- Desarrollar e integrar, entre los meses de mayo y junio, al menos tres medidas de seguridad en el backend del sistema que reduzcan la posibilidad de accesos no autorizados en al menos un **80 %** en pruebas de seguridad automatizadas realizadas en julio.
- Diseñar e implementar, entre los meses de junio y julio, pruebas de usabilidad en la versión móvil del frontend, evaluando aspectos como navegación, tiempos de respuesta y comprensión visual, con el objetivo de alcanzar una **tasa de éxito del 90 %** en tareas comunes, las cuales son definidas durante las pruebas realizadas en el mes de agosto.
- Configurar e integrar pruebas automatizadas entre junio y julio que evalúen el rendimiento y la seguridad con resultados obtenidos y documentados en agosto, demostrando al menos un **95 % de cumplimiento** de los criterios definidos en los protocolos para QA establecidos.
- Corroborar la efectividad de las estrategias implementadas por medio de sesiones de validación con usuarios reales durante agosto y septiembre para medir la experiencia de uso y la percepción de seguridad, recopilando retroalimentación por medio de encuestas estructuradas y entrevistas, con una meta de por lo menos **80 % de satisfacción general** en los resultados.



# CAPÍTULO 5

---

Alcance

---

## 5.1. Ámbito funcional

- Fortalecimiento del **backend** con, al menos, tres controles de seguridad (autenticación robusta, control de acceso por roles, cifrado en tránsito y reposo, entre otros) con evidencia de efectividad.
- Intervenciones en el **frontend móvil** (encuestadores/encuestados) centradas en **usabilidad, validaciones en origen** y operación *offline-first* (almacenamiento local, sincronización y resolución de conflictos).
- **Pruebas automatizadas** (regresión de flujos críticos, rendimiento básico y chequeos de seguridad no intrusivos) integradas al pipeline de CI.
- **Validación con usuarios** mediante pruebas por tareas, encuestas de satisfacción y entrevistas semiestructuradas.

## 5.2. Ámbito técnico

- Arquitectura *offline-first*: almacenamiento local cifrado, sincronización diferida, manejo de conectividad intermitente y políticas anti-*root/jailbreak*.
- Backend con **endpoints** protegidos, validación de entradas, registro/auditoría de eventos y endurecimiento de superficie de ataque.
- **Automatización de pruebas**: suite móvil (p.ej., Appium) y escaneo de API (p.ej., DAST) contra *staging*, con reportería reproducible.
- **Modelado de datos para analítica**: normalización de exportaciones para reducir anidamiento y facilitar consumo tabular/indicadores.

### 5.3. Resultados e indicadores esperados

- **Seguridad (backend)**: reducción  $\geq 80\%$  de accesos no autorizados/hallazgos críticos en pruebas automatizadas respecto a la línea base.
- **Usabilidad (frontend móvil)**: **90 %** de tasa de éxito en tareas comunes; **25 %** de reducción en tiempo promedio de llenado frente a la herramienta actual.
- **Calidad de captura**: **50 %** menos errores (incompletos, mal ingresados o duplicados) por validaciones en origen.
- **QA automatizado**: **95 %** de *pass rate* en la suite de regresión.
- **Resiliencia offline**: **100 %** de sincronizaciones efectivas en escenarios controlados con conectividad intermitente.
- **Satisfacción de usuarios**:  $\geq 80\%$  de respuestas 4–5 (escala 1–5) tras las pruebas de campo.

### 5.4. Entregables

- **Plan de Seguridad y Evidencia Técnica**: controles implementados, parámetros, bitácora de cambios y resultados *pre/post*.
- **Plan de QA y Suite Automatizada**: casos, datos de prueba, guiones de ejecución y reportes.
- **Guía de Operación offline-first**: criterios de almacenamiento, políticas de sincronización y manejo de conflictos.
- **Informe de Usabilidad y Satisfacción**: diseño de pruebas, instrumentos, métricas, análisis y recomendaciones.
- **Anexo de Modelo de Datos**: esquema normalizado de exportaciones y mapeos a tablas/indicadores.

### 5.5. Actores y contexto

- **Usuarios de campo**: personal que captura información en entornos con conectividad limitada.
- **Stakeholders internos**: personal administrativo/gerencial que consume reportes y define requerimientos.
- **Equipo de desarrollo/QA**: responsable de controles, automatización y generación de evidencia.

## 5.6. Supuestos y restricciones

- Disponibilidad de **entorno de staging** con datos sintéticos y servicios equivalentes a producción.
- **Matriz de dispositivos** (Android/iOS) acotada a los modelos prevalentes en el ingenio.
- Acceso a **línea base** de tiempos de llenado y estructura actual de exportaciones para comparativos.
- Ventana temporal de ejecución técnica y validaciones según cronograma institucional.

## 5.7. Fuerza de alcance

- Sustitución total y despliegue *productivo* a toda la organización (limitado a prototipo y piloto).
- Implementación de **MDM** corporativo; el enfoque se limita a controles a nivel de aplicación.
- Construcción de tableros **BI** avanzados más allá de datasets listos para análisis.
- Integraciones con sistemas legados fuera de los endpoints definidos para pruebas.

## 5.8. Criterios de aceptación

- Cumplimiento de las **métricas objetivo** (seguridad, usabilidad, QA, satisfacción y sincronización).
- Disponibilidad de **evidencia reproducible**: scripts, configuraciones, registros y reportes en anexos.
- **Aprobación** de stakeholders del ingenio sobre la factibilidad técnica y operativa del prototipo en contexto BYOD y conectividad limitada.



# CAPÍTULO 6

---

## Marco teórico

---

### 6.1. Arquitectura *offline-first* y sincronización de datos

Una aplicación *offline-first* parte de la idea de que la *fuente de verdad* vive de forma primaria en el dispositivo del usuario y no en la nube. Esto se relaciona con el concepto más amplio de *local-first software*, donde el almacenamiento y la edición ocurren localmente, y los servicios remotos actúan como mecanismos de replicación, colaboración y respaldo. Este enfoque reduce la latencia percibida, mantiene la disponibilidad ante fallos de red y otorga mayor control al usuario sobre la información que manipula (Kleppmann et al., 2019).

Desde una perspectiva de datos, las arquitecturas *offline-first* suelen apoyarse en almacenes transaccionales locales y en un registro de cambios (*journal append-only*) que captura operaciones de creación, edición y eliminación, junto con metadatos de orden lógico. La idea de desacoplar la experiencia del usuario del transporte se refleja en patrones como la *Transactional Outbox*, donde los cambios confirmados se publican hacia el servidor cuando existe conectividad y los eventos entrantes se aplican de manera ordenada al estado local (Microsoft Learn, 2021; Richardson, 2025). La prioridad es garantizar que el usuario pueda seguir trabajando aunque el canal remoto falle, y que la sincronización se recupere de forma confiable cuando sea posible.

En términos de teoría de sistemas distribuidos, este tipo de aplicaciones privilegia disponibilidad y tolerancia a particiones, adoptando modelos de consistencia eventual y de convergencia posterior a la reconexión. El teorema CAP formaliza que no es posible garantizar, al mismo tiempo, consistencia fuerte, disponibilidad y tolerancia a particiones; en presencia de desconexiones, la arquitectura *offline-first* prefiere mantener la disponibilidad y restablecer la consistencia cuando la red se normaliza (Gilbert & Lynch, 2002). Para razonar sobre concurrencia y orden de eventos en estos entornos, se emplean relaciones de precedencia (*happens-before*) y relojes lógicos que permiten derivar órdenes totales compatibles con la causalidad (Lamport, 1978).

La resolución de conflictos entre réplicas constituye un aspecto conceptual clave. Las es-

trategias van desde políticas simples basadas en marcas temporales (*last-write-wins*) hasta fusiones definidas por dominio a nivel de campo o registro, pasando por estructuras formales como los *Conflict-free Replicated Data Types* (CRDT). Estos últimos ofrecen *Strong Eventual Consistency* mediante reglas algebraicas que garantizan convergencia sin coordinación centralizada, incluso bajo latencias y fallos arbitrarios (Gomes et al., 2017; Shapiro et al., 2011). De manera complementaria, ciertos sistemas emplean control de versiones multiversión (MVCC) y árboles de revisiones, en los que las ramas divergentes se detectan explícitamente y pueden listarse y reconciliarse, como ilustra el protocolo de replicación de CouchDB (Apache Software Foundation, 2024a, 2024b). En todos los casos, el objetivo teórico es que las réplicas converjan hacia un estado consistente sin sacrificar la capacidad de trabajar desconectado.

Las topologías de replicación juegan también un papel conceptual. La sincronización puede ser unidireccional (dispositivo → servidor), bidireccional o continua, según las necesidades del dominio. Experiencias documentadas en ecosistemas de bases de datos orientadas a documentos muestran cómo la replicación incremental, los filtros y las estrategias de reintento permiten adaptar estos patrones a contextos móviles y de borde, manteniendo un equilibrio entre frescura de datos y uso de recursos (PouchDB Project, 2025). Sobre este fundamento teórico, conceptos como idempotencia, reintentos seguros y deduplicación completan el cuadro: la semántica de métodos HTTP y propuestas como el encabezado **Idempotency-Key** formalizan cómo diseñar operaciones remotas que toleren reenvíos, *timeouts* y duplicados sin introducir efectos inconsistentes (Fielding et al., 2022; Michel et al., 2025).

En síntesis, la arquitectura *offline-first* se apoya en principios de consistencia eventual, orden lógico de eventos, resolución explícita de conflictos y patrones de sincronización robustos. Estos elementos proporcionan el marco conceptual necesario para diseñar motores de sincronización que mantengan una experiencia fluida aun en presencia de conectividad intermitente (Apache Software Foundation, 2024a; Kleppmann et al., 2019; Shapiro et al., 2011).

## 6.2. Dispositivos personales, gestión móvil y modelos de confianza

En organizaciones que permiten el uso de equipos personales para acceder a recursos corporativos, el modelo *Bring Your Own Device* (BYOD) ha dado lugar a esquemas de gestión que desplazan el foco de control desde el dispositivo completo hacia las aplicaciones y los datos. Las guías de organismos como el National Cyber Security Centre (NCSC) sintetizan este enfoque en dos principios: limitar el acceso a la información que la organización está dispuesta a exponer en equipos personales y aplicar controles técnicos que separen y protejan los datos corporativos frente al ámbito privado del usuario (UK National Cyber Security Centre, 2025).

Dentro de este contexto han surgido dos grandes familias conceptuales de gestión: *Mobile Device Management* (MDM) y *Mobile Application Management* (MAM). MDM se orienta a administrar el dispositivo como activo completo (políticas de seguridad, inventario, configuración), mientras que MAM se centra en las aplicaciones y sus datos, permitiendo aplicar

cifrado lógico, borrado selectivo o restricciones de intercambio de información sin asumir control total sobre el equipo personal (Microsoft Learn, [2025b], [2025d]). Plataformas como Android Enterprise y los programas de inscripción para dispositivos de propiedad del empleado en ecosistemas móviles modernos ilustran cómo se materializa esta separación lógica mediante perfiles de trabajo o espacios gestionados (Apple, [2024]; Google Developers, [2025a], [2025b]; Microsoft Learn, [2025c]).

Estos modelos se alinean con el paradigma de *Zero Trust*, que plantea abandonar la confianza implícita en la red interna o en la propiedad del dispositivo y, en su lugar, evaluar la confianza de forma continua a partir de la identidad, el contexto y la postura del activo. Documentos de referencia como el marco de NIST sobre arquitecturas *Zero Trust* y los modelos de madurez de CISA enfatizan que la protección de datos y servicios debe estar desacoplada de la ubicación física y orientada a políticas explícitas (Cybersecurity and Infrastructure Security Agency, [2023]; Rose et al., [2020]). En escenarios BYOD, este enfoque proporciona el trasfondo conceptual para pensar en aplicaciones de formularios que tratan datos sensibles en dispositivos que no son propiedad de la organización.

La literatura en seguridad móvil también ha prestado atención a la detección de entornos comprometidos (por ejemplo, dispositivos *rooteados* o *jailbroken*) y a la limitación de la confianza que puede derivarse de estas señales. Los lineamientos de OWASP orientados a pruebas de aplicaciones móviles recogen estas preocupaciones y las integran dentro de catálogos de controles recomendados para proteger datos en reposo, en tránsito y en uso (OWASP Mobile Application Security Testing Guide, [2025a], [2025b], [2025c]). Finalmente, las guías específicas sobre autenticación para aplicaciones nativas recomiendan separar el flujo de autorización (realizado en un agente de usuario de confianza, típicamente el navegador del sistema) del código de la aplicación, aprovechando estándares como OAuth 2.0 y sus perfiles para aplicaciones móviles (Denniss & Bradley, [2017]; OAuth.net, [2025]; RFC Editor, [2017]). En conjunto, estos marcos conceptuales proporcionan criterios para analizar, en el plano teórico, cómo delimitar responsabilidades entre dispositivo, aplicación y servicios de identidad.

### 6.3. Seguridad por diseño, ciclo de vida de desarrollo y cadena de suministro

La noción de *seguridad por diseño* propone incorporar objetivos y controles de seguridad desde las etapas iniciales del ciclo de vida del software, en lugar de abordarlos únicamente como una fase tardía de revisión. Modelos como el *Secure Software Development Life Cycle* (SSDLC) formalizan este enfoque, describiendo actividades de seguridad asociadas a las fases de requisitos, diseño, implementación, verificación y operación. Dentro de este marco, se incluyen prácticas como la definición de requisitos de seguridad y privacidad, el modelado de amenazas, el uso de reglas de codificación segura, la gestión de dependencias y la realización de análisis estáticos y dinámicos (Microsoft Security Engineering, [2025a], [2025b]).

El *Software Assurance Maturity Model* (OWASP SAMM) complementa esta perspectiva proponiendo un modelo de madurez que agrupa prácticas de seguridad en dominios como gobierno, diseño, implementación, verificación y operaciones. Cada dominio se descompone

en actividades medibles, lo que permite a las organizaciones evaluar su situación actual y trazar una hoja de ruta incremental hacia niveles superiores de aseguramiento (OWASP Foundation, 2025b). En paralelo, iniciativas de organismos públicos, como las referencias de DevSecOps del Departamento de Defensa de Estados Unidos, describen cómo integrar estas prácticas en *pipelines* de integración y entrega continua, de manera que las verificaciones de seguridad se conviertan en parte natural del proceso de desarrollo (*DoD Enterprise DevSecOps Reference Design (Cloud Native/SaaS)*, 2022; *DoD Enterprise DevSecOps Reference Design (v1.0)*, 2021).

En arquitecturas basadas en servicios distribuidos, la atención a la *cadena de suministro de software* ha cobrado relevancia. Marcos como SLSA (*Supply-chain Levels for Software Artifacts*) modelan niveles de garantía sobre la procedencia y la integridad de los artefactos de software, proponiendo prácticas como la generación de metadatos de proveniencia firmados, *builds* reproducibles y controles de revisión (SLSA Framework, 2025a, 2025b). Documentos técnicos de NIST amplían estas ideas para contextos específicos de DevSecOps, sugiriendo mecanismos de verificación de dependencias, artefactos y políticas de promoción entre ambientes (Chandramouli, 2022, 2024). Este corpus teórico proporciona el trasfondo para analizar cómo se relacionan los procesos de desarrollo, las herramientas de automatización y los mecanismos de aseguramiento de la cadena de suministro en proyectos de software crítico.

## 6.4. Modelado de amenazas en aplicaciones de datos

El *modelado de amenazas* es una técnica sistemática para identificar, estructurar y razonar sobre posibles ataques contra un sistema. Entre los enfoques más difundidos se encuentra STRIDE, que clasifica las amenazas en seis categorías: Suplantación (*Spoofing*), Manipulación (*Tampering*), Repudio, Divulgación de información, Denegación de servicio y Elevación de privilegios. Esta taxonomía permite recorrer elementos de una arquitectura y preguntarse, de forma ordenada, qué vectores pueden materializarse en cada categoría (Microsoft Learn, 2022; Shostack, 2014).

Otro ángulo conceptual relevante es el *modelado centrado en datos*. En lugar de partir únicamente de componentes técnicos, este enfoque propone comenzar por las clases de datos que se desean proteger, sus ubicaciones, los flujos a través de los cuales se mueven y los actores que los procesan. La literatura de NIST sobre modelos de seguridad centrados en datos enfatiza la importancia de mapear activos de información, ubicaciones de almacenamiento, canales de transmisión y procesos de transformación como base para derivar controles y mecanismos de validación (Souppaya, Scarfone et al., 2016). Este punto de vista resulta especialmente pertinente cuando se trabaja con formularios que capturan información operativa y potencialmente sensible.

De forma complementaria, se han desarrollado catálogos y taxonomías que organizan técnicas de ataque y debilidades frecuentes en aplicaciones móviles. Entre ellos se encuentran matrices que agrupan tácticas y técnicas observadas en la práctica, así como conjuntos de riesgos prioritarios específicos para aplicaciones móviles, que incluyen problemas de almacenamiento inseguro, autenticación débil, exposición de datos en tránsito y endurecimiento insuficiente (MITRE ATT&CK, 2025a, 2025b; OWASP Foundation, 2025a). Estos recursos

no se enfocan en una implementación particular, sino que proporcionan un lenguaje común para describir amenazas y facilitan que los modelos conceptuales se conecten con escenarios concretos de riesgo.

En este marco, las amenazas relevantes para aplicaciones que manejan datos de formularios suelen agruparse en torno a la confidencialidad e integridad de los datos en reposo y en tránsito, la autenticidad de los actores que acceden a ellos, la trazabilidad de las acciones realizadas y la resiliencia frente a fallos o abusos. Los mecanismos de protección de datos en reposo y de gestión de claves en plataformas móviles modernas, documentados en las guías de seguridad de Android y Apple, constituyen ejemplos de cómo las capacidades de la plataforma pueden utilizarse como bloques conceptuales para implementar las decisiones de protección derivadas del modelado de amenazas (Android Developers, 2025a; Apple Support, 2024). Finalmente, propuestas como las de OWASP sobre modelado de amenazas ofrecen rutas prácticas para incorporar estas técnicas al ciclo de vida de desarrollo, manteniendo la conexión entre análisis conceptual y evidencia de mitigación (OWASP Foundation, 2025d).

## 6.5. Calidad, usabilidad y medición de la experiencia

### 6.5.1. Aseguramiento de calidad y automatización de pruebas

El aseguramiento de calidad (QA) en ingeniería de software se refiere al conjunto de actividades orientadas a proporcionar confianza en que un producto cumple con requisitos funcionales y no funcionales. La literatura enfatiza que la automatización de pruebas contribuye a acortar ciclos de retroalimentación, reducir regresiones y mejorar la estabilidad de sistemas complejos (Berihun et al., 2023; Kong et al., 2019). Un modelo conceptual ampliamente citado es la *pirámide de pruebas*, que sugiere organizar el esfuerzo de verificación en capas: muchas pruebas de unidad de bajo costo y alta señal, un conjunto más reducido de pruebas de servicio o integración, y un número aún menor de pruebas de interfaz de usuario o extremo a extremo (Cohn & Fowler, 2018).

En el ámbito móvil, estos principios se adaptan a la existencia de múltiples dispositivos, sistemas operativos y condiciones de ejecución. Las plataformas de desarrollo proporcionan marcos de pruebas específicas para validar componentes lógicos y de interfaz, así como infraestructuras que permiten ejecutar suites de pruebas sobre matrices de dispositivos físicos y virtuales (Android Developers, 2025c, 2025d; Apple Developer Documentation, 2025a, 2025b, 2025c; Firebase, 2024, 2025a, 2025b). A su vez, marcos de pruebas extremo a extremo multiplataforma ofrecen una base conceptual para pensar en recorridos de usuario que atraviesan el sistema de manera completa (Appium Project, 2025b, 2025c; Wix, 2025a, 2025b).

Un desafío teórico y práctico asociado es el de las *pruebas inestables* o *flaky tests*, que fallan de manera no determinista debido a factores externos, asincronía o dependencias frágiles. Estudios empíricos publicados por grandes proveedores de servicios en la nube analizan las causas y proponen estrategias de mitigación, subrayando que la confiabilidad de las señales de prueba es tan importante como su cobertura (Google Testing Blog, 2016, 2020). Esto refuerza la idea de que la calidad no depende únicamente de la existencia de pruebas, sino de su diseño, estabilidad y capacidad para informar decisiones sobre el estado del sistema.

### **6.5.2. Usabilidad de formularios y escala SUS**

La usabilidad se define, de acuerdo con la norma ISO 9241-11, como el grado en que un sistema puede ser utilizado por usuarios específicos para lograr objetivos concretos con eficacia, eficiencia y satisfacción en un contexto de uso determinado (International Organization for Standardization, 2018). En el caso particular de formularios, la investigación aplicada ha mostrado que factores como la cantidad y complejidad de los campos, la claridad del etiquetado, la disposición visual y la retroalimentación de errores influyen de manera directa en la carga cognitiva y en la probabilidad de abandono. Estudios y guías de organizaciones especializadas en experiencia de usuario recomiendan, entre otros aspectos, reducir campos opcionales, agrupar la información de manera lógica, alinear etiquetas, ofrecer ayuda contextual y aplicar validación en línea (Nielsen Norman Group, 2016). Investigaciones sobre formularios en comercio electrónico han encontrado que el número de campos visibles y su organización afectan más la usabilidad que la elección entre un flujo de una página o multipaso, destacando la importancia de eliminar campos innecesarios y emplear *progressive disclosure* (Baymard Institute, 2011, 2024).

Para medir la usabilidad percibida de sistemas interactivos de manera estandarizada, la *System Usability Scale* (SUS) se ha consolidado como uno de los instrumentos más difundidos. Este cuestionario de diez ítems, propuesto originalmente por Brooke, ofrece una medida global de usabilidad en una escala de 0 a 100 a partir de respuestas en formato Likert. A pesar de su sencillez, múltiples estudios han documentado su validez, confiabilidad y utilidad para comparar sistemas o versiones de un mismo producto (Bangor et al., 2009; Brooke, 1996; Lewis & Sauro, 2009). Desde un punto de vista teórico, SUS aporta una métrica sintética de usabilidad percibida que puede combinarse con indicadores de tarea (tasas de éxito, tiempos, errores) y con observaciones cualitativas para caracterizar la experiencia de uso de una aplicación de formularios en contextos de campo.

## **6.6. Privacidad, gobernanza de datos y dependencia de proveedores**

El tratamiento de datos personales y operativos se encuentra enmarcado por normas jurídicas y estándares de gestión que, aunque varían entre jurisdicciones, comparten ciertos principios comunes. En el contexto guatemalteco, la ausencia de una ley general de protección de datos personales de alcance amplio convive con disposiciones constitucionales y con regulaciones específicas, como la Ley de Acceso a la Información Pública, que incorporan conceptos vinculados a datos personales, datos sensibles y *habeas data* en el ámbito de entidades públicas o sujetas a obligaciones de transparencia.<sup>1</sup> Esto genera un escenario en el que, aunque no existe un régimen integral equivalente al de otras jurisdicciones, persisten expectativas de protección de la vida privada y de diligencia en el manejo de información por parte de organizaciones privadas.

En paralelo, marcos como el Reglamento General de Protección de Datos (RGPD) de la Unión Europea han influido en la discusión global sobre protección de datos, aun cuando

---

<sup>1</sup>Véanse, por ejemplo, análisis sobre el marco guatemalteco en materia de datos personales y la Ley de Acceso a la Información Pública.

su aplicabilidad directa se limite a supuestos específicos de establecimiento o de oferta de bienes y servicios hacia residentes en la Unión (European Data Protection Board, 2020). Más allá de su carácter obligatorio o no para una organización concreta, los principios de minimización, limitación de finalidad, transparencia y derechos de los titulares se han convertido en referencias frecuentes para el diseño de prácticas de tratamiento de datos.

Estándares como ISO/IEC 27001 e ISO/IEC 27701 proporcionan, desde la gestión de riesgos, un marco sistemático para organizar la seguridad de la información y la privacidad. ISO/IEC 27001 define requisitos para establecer y mejorar un Sistema de Gestión de Seguridad de la Información (SGSI), mientras que ISO/IEC 27701 extiende este esquema a un Sistema de Gestión de la Información de Privacidad (PIMS), precisando roles y controles para responsables y encargados de datos personales (International Organization for Standardization, 2019; “ISO/IEC 27001:2022 Information security, cybersecurity and privacy protection — Information security management systems — Requirements”, 2022). Aunque su adopción es voluntaria, estos estándares ofrecen un lenguaje común para articular políticas, controles y evidencias de cumplimiento frente a terceros.

Dentro de este marco conceptual, la *minimización de datos* aparece como principio central: recolectar y conservar sólo la información necesaria para fines legítimos y definidos. Guías de autoridades de protección de datos, como la Information Commissioner’s Office (ICO) del Reino Unido, destacan este principio tanto en contextos generales como en situaciones que involucran poblaciones vulnerables (Information Commissioner’s Office, 2025a, 2025b). Asimismo, documentos de organismos europeos sobre portabilidad y medidas complementarias para transferencias internacionales subrayan la importancia de que los datos sean exportables en formatos estructurados y de que se preserven niveles adecuados de protección al mover información entre proveedores o jurisdicciones (European Data Protection Board, 2020, 2025).

La noción de *vendor lock-in* o dependencia de proveedores se relaciona con estas preocupaciones desde la perspectiva de gobernanza de datos. La literatura sobre arquitectura de sistemas y gestión de información plantea que el uso de formatos propietarios, interfaces cerradas o condiciones contractuales restrictivas puede limitar la capacidad de una organización para migrar sus datos o cambiar de proveedor sin costos elevados. Como respuesta, se proponen principios como el uso de modelos de datos documentados y exportables, la adopción de estándares abiertos para autenticación e intercambio de información, y la inclusión de cláusulas de portabilidad razonable en acuerdos con terceros (Microsoft Learn, 2024). Estos elementos, considerados de manera teórica, ayudan a enmarcar decisiones arquitectónicas orientadas a preservar la soberanía del dato y a reducir riesgos asociados a la concentración de capacidades en plataformas externas.

## 6.7. Datos de formularios y analítica operativa

Desde la perspectiva de analítica de datos, la utilidad de la información capturada mediante formularios depende no sólo de su contenido, sino también de su estructura. Aunque los motores de bases de datos modernos permiten almacenar y consultar datos semiestructurados (por ejemplo, en formatos JSON), las prácticas de modelado orientadas a inteligencia de negocios recomiendan generar vistas analíticas con esquemas claros y relativamente sim-

ples. Un enfoque consolidado es el de los esquemas en estrella (*star schema*), donde una tabla de hechos recoge eventos o mediciones y se relaciona con tablas de dimensiones que describen entidades relevantes, facilitando la construcción e interpretación de indicadores (Microsoft Learn, [2024]).

Herramientas analíticas en la nube documentan funciones específicas para transformar estructuras anidadas en tablas relacionales, como la expansión de arreglos y objetos mediante operaciones de *flattening* y uniones laterales (Snowflake Inc., [2025a], [2025b]). Más allá de la implementación concreta, la idea teórica subyacente es que la complejidad estructural puede trasladarse de las fuentes de captura a las capas de transformación, produciendo conjuntos de datos “analizables” que reduzcan ambigüedades y costos recurrentes de preparación. En aplicaciones que recogen información operativa para apoyar la toma de decisiones, esta perspectiva refuerza la importancia de diseñar desde el origen formularios y modelos de datos que faciliten la construcción de indicadores y el uso de técnicas de análisis exploratorio y supervisado.

# CAPÍTULO 7

---

## Metodología

---

### 7.1. Enfoque y diseño metodológico

La investigación adopta un **enfoque mixto**, combinando mediciones *cuantitativas* (p. ej., tiempos de llenado, *pass rate*, reducción de vulnerabilidades) y *cualitativas* (p. ej., entrevistas semiestructuradas, observaciones de uso). Este diseño es idóneo para evaluar simultáneamente desempeño técnico y experiencia de usuario en una aplicación de formularios *offline-first*, articulando el ciclo de vida de seguridad (SSDLC/DevSecOps) con evaluación de usabilidad y QA automatizado (Chatterjee, 2024; Holl, 2019; Rezaee et al., 2023).

### 7.2. Área de estudio y población

El estudio se realiza **en y para el Ingenio Santa Ana** (Guatemala), contemplando:

- **Hábitat físico-operativo:** cuadrillas en campo con conectividad intermitente o nula.
- **Hábitat digital:** aplicación móvil (encuestadores/encuestados) y backend.

La población se compone de: (i) **usuarios de campo** con distintos niveles de alfabetización digital, e (ii) **stakeholders** administrativos/gerenciales. Para validación, se emplea *muestreo por conveniencia* cubriendo diversidad de roles y antigüedad. Como contexto sectorial, se consideran informes públicos sobre el *sector azucarero* en Guatemala (Compañía Agrícola Industrial Santa Ana, S. A., 2019; Superintendencia de Bancos de Guatemala, 2016).

### 7.3. Fases y cronograma

La ejecución del proyecto se desarrolló entre **mayo y octubre de 2025**, organizada en cuatro fases que combinan trabajo técnico, coordinación con actores del ingenio y actividades de validación. Aunque se presentan de forma secuencial, varias actividades se solaparon parcialmente para aprovechar iteraciones rápidas entre diseño, implementación y evaluación.

1. **Análisis del estado actual y planificación (mayo 2025).** En esta fase se realizó la revisión bibliográfica inicial sobre *offline-first*, BYOD, seguridad móvil, SSDLC y usabilidad de formularios, así como el levantamiento de información sobre la herramienta de captura vigente en el Ingenio Santa Ana. Se efectuaron observaciones de campo y sesiones exploratorias con usuarios clave para entender flujos de captura, condiciones de conectividad y restricciones operativas. A partir de esta información se definieron los *casos de uso* prioritarios, los flujos críticos a monitorear y una línea base preliminar de usabilidad y seguridad (situación actual), además de afinar la hipótesis y los criterios de éxito del proyecto.
2. **Diseño de la solución y arquitectura *offline-first* (junio–julio 2025).** Se definieron los controles de seguridad a implementar tomando como referencia lineamientos de MASVS y el marco de seguridad por diseño, y se diseñó la estrategia de QA automatizado (tipos de pruebas, alcance de la suite, criterios de aceptación). Esta fase concluyó con la planificación detallada de los escaneos de seguridad, las sesiones de usabilidad y las métricas a recolectar.
3. **Implementación y *hardening* de la solución (julio–agosto 2025).** En esta fase se desarrollaron los flujos de autenticación y los formularios objetivo. Se configuraron los *pipelines* de integración continua, incluyendo la ejecución de pruebas automatizadas y escaneos de seguridad sobre el API. Asimismo, se aplicaron los controles de *hardening* definidos (validaciones de entrada, manejo de sesiones, almacenamiento local seguro) y se preparó el entorno controlado donde se realizarían las evaluaciones técnicas y con usuarios.
4. **Validación, análisis y ajustes (septiembre–octubre 2025).** Finalmente, se ejecutaron los escaneos de seguridad *pre/post*, las suites de pruebas automatizadas (móvil y API) y las sesiones de usabilidad moderadas con usuarios de campo. Se recopilaron las métricas de seguridad, usabilidad, QA y satisfacción, y se procesaron en las herramientas de análisis planificadas. Con base en estos resultados se realizaron ajustes puntuales al prototipo (mensajes, validaciones, configuración de seguridad) y se documentaron los hallazgos, limitaciones y recomendaciones para el ingenio, así como la evidencia necesaria para evaluar la hipótesis planteada.

## 7.4. Materiales

### Software de seguridad (DAST/SAST)

**OWASP ZAP.** Se seleccionó OWASP ZAP como herramienta principal de *Dynamic Application Security Testing* (DAST) por tratarse de un proyecto abierto y mantenido por la comunidad OWASP, con soporte específico para el escaneo de aplicaciones web y APIs HTTP. Su capacidad de realizar *spiders*, escaneos activos, autenticación contra el backend y generación de reportes estructurados permite establecer una línea base de vulnerabilidades y comparar de forma reproducible el estado *pre/post* de la API del ingenio (OWASP Foundation, [2025c]).

**Snyk SAST.** Para el análisis estático de código y dependencias se eligió Snyk, que ofrece integración nativa con repositorios y *pipelines* de CI/CD, así como reglas actualizadas para detectar vulnerabilidades conocidas e incluso secretos expuestos en el código. Su integración con *Pull Requests* y con *Github Actions* permite incorporar *checks* automáticos en el flujo de desarrollo y cuantificar la reducción de hallazgos de severidad alta y media después de los cambios de *hardening* (Snyk Project, [2024]). Esta elección responde a la necesidad de medir el impacto de la implementación sobre la superficie de riesgo del backend.

### Pruebas automatizadas (móvil)

**Appium.** Para la automatización de pruebas extremo a extremo en la aplicación móvil se seleccionó Appium, un marco ampliamente utilizado que permite controlar aplicaciones nativas sobre Android e iOS mediante la misma API de automatización. Esto facilita la definición de flujos de prueba que simulan el recorrido real del usuario (inicio de sesión, captura *offline*, sincronización diferida) y su integración en la suite de regresión del proyecto. El uso de Appium permite, además, instrumentar la tasa de éxito y los fallos observados en escenarios representativos, alineado con la necesidad de contar con QA automatizado sobre los flujos críticos de formularios (Appium Project, [2025a]).

### Recolección de datos

**Google Forms/Sheets y MS Excel.** Para la recolección y organización de datos de evaluación (respuestas de cuestionarios, tiempos de tarea, resultados de pruebas) se emplearon Google Forms y Google Sheets, que facilitan la distribución de encuestas, la captura remota y la tabulación automática de respuestas (Google Help, [2025]; Google Workspace Learning Center, [2025]). Microsoft Excel se utilizó como herramienta complementaria para depurar, consolidar y transformar las tablas de seguimiento, dado su uso extendido en el entorno del ingenio y su compatibilidad con formatos de intercambio habituales (Microsoft Learn, [2025a]). La elección de estas herramientas responde a la necesidad de minimizar la fricción en la captura de datos y permitir un procesamiento posterior flexible.

## Análisis estadístico

**IBM SPSS Statistics (o equivalente).** Para el análisis cuantitativo de los datos se contempló el uso de IBM SPSS Statistics (o herramienta funcionalmente equivalente), orientado a la obtención de estadísticas descriptivas (frecuencias, porcentajes, medidas de tendencia central) y al contraste de tiempos de tarea entre la herramienta actual y el prototipo (IBM, [2025]). Este tipo de software es adecuado para manejar muestras pequeñas o piloto y ofrece procedimientos estandarizados que facilitan la trazabilidad del análisis y la documentación de resultados en el contexto académico del trabajo de graduación.

## 7.5. Métodos

### 7.5.1. Seguridad en backend (*pre/post*)

Se realiza un **pre-test** con OWASP ZAP contra los endpoints de API para obtener una línea base de vulnerabilidades (clasificadas por severidad). Tras implementar controles (validación de entradas, endurecimiento, autenticación/gestión de sesiones), se repite el **post-test**. Indicador principal: **reducción porcentual** de hallazgos *Altos/Medios*, con meta  $\geq 80\%$  (Chatterjee, [2024]; OWASP Foundation, [2025c]).

### 7.5.2. Usabilidad en frontend (pruebas moderadas por tareas)

Se definen al menos **tres tareas críticas** alineadas con el flujo operativo de los usuarios de campo: (i) iniciar sesión y localizar el formulario correspondiente, (ii) capturar y guardar información en modo *offline*, y (iii) sincronizar de forma diferida los registros almacenados localmente. Cada participante realiza estas tareas en un entorno moderado, donde se observa su interacción, se registran tiempos, errores y necesidad de ayuda, y se documentan comentarios espontáneos.

Para cada tarea se registra un indicador de **tasa de éxito**, entendido como la proporción de participantes que logran completarla correctamente dentro de los criterios definidos (sin errores graves y sin intervención del moderador). A nivel conceptual, esta tasa operacionaliza la dimensión de eficacia de la usabilidad: si los usuarios pueden completar las tareas que el sistema debe soportar. La tasa de éxito por tarea se calcula como el cociente entre el número de ejecuciones completadas con éxito y el número total de intentos, expresado en porcentaje; el promedio de estas tasas sobre las tareas críticas ofrece una medida sintética de qué tan bien soporta la aplicación los flujos clave. Esta métrica cuantitativa se complementa con la observación cualitativa de dónde se producen bloqueos, errores o dudas frecuentes, lo que permite vincular los resultados con decisiones concretas de diseño de formularios, mensajes y validaciones.

Tras completar las tareas, se aplica el cuestionario *System Usability Scale* (SUS) de diez ítems propuesto por Brooke, que resume la usabilidad percibida en un único puntaje entre 0 y 100 (Bangor et al., [2009]; Brooke, [1996]). En su forma estándar, cada afirmación se responde en una escala Likert de cinco puntos (de “totalmente en desacuerdo” a “totalmente de acuerdo”),

alternando ítems positivos y negativos; las respuestas se transforman y suman para obtener la puntuación global SUS. En este proyecto se trabaja con un *grupo piloto reducido* de 10 usuarios de campo (muestreo por conveniencia), lo cual es adecuado para pruebas formativas y para obtener una primera señal cuantitativa de usabilidad, reconociendo que no permite inferencias estadísticas fuertes.

Por simplicidad y familiaridad con los encuestados, la SUS se administró en formato dicotómico (*Sí/No*) en lugar de la escala de cinco puntos. Las respuestas se recodifican a una métrica de 0 a 100 calculando el porcentaje de ítems respondidos en la dirección deseada (acuerdo en los ítems positivos y desacuerdo en los negativos) y multiplicándolo por 100; este valor se interpreta como un *indicador aproximado* de usabilidad percibida y se utiliza para comparar versiones dentro del proyecto, dejando explícito que no es directamente comparable con baremos internacionales de SUS basados en escalas Likert.

El criterio cuantitativo de aceptación combina ambos niveles de análisis: (i) al menos un 90 % de éxito promedio en las tareas críticas ( $SER \geq 90\%$ ), como expresión de la eficacia del sistema para soportar los flujos fundamentales, y (ii) una puntuación SUS aproximada alineada con la categoría de buena usabilidad (por encima del umbral interno definido en función de la muestra), complementados con el análisis cualitativo de mensajes, validaciones en origen y aspectos de accesibilidad para iterar el diseño (Holl, 2019).

### 7.5.3. QA automatizado (regresión)

Se orquesta una **suite de regresión** en Appium que cubra: *happy paths* de captura/sincronización, validaciones obligatorias, manejo de reconexión y consistencia de datos *post-sync*. Indicador: **pass rate** ( $PR = \frac{p}{t} \cdot 100$ ) con meta  $\geq 95\%$ . Se registran causas de fallo y *flakiness* para hardenizar *locators*/esperas (Appium Project, 2025a; Chatterjee, 2024).

### 7.5.4. Validación y satisfacción

Se aplica encuesta **CSAT** ( $S = \frac{u+e}{t} \cdot 100$ , con  $u = \#4$  y  $e = \#5$ ) tras pruebas de campo, y entrevistas semiestructuradas para profundizar en razones de éxito/fallo. Meta:  $\geq 80\%$  de satisfacción. Los resultados cualitativos informan cambios de interfaz y mensajes.

## 7.6. Pruebas de resiliencia y conectividad

Se simulan escenarios de **conectividad adversa** (pérdida de señal, latencia elevada, *timeouts*) y se instrumenta el *sync engine* para telemetría (*retries*, colas, latencias, tasa de sincronizaciones exitosas). Criterio: **100 %** de sincronizaciones efectivas en escenarios controlados, con *backoff* y reintentos idempotentes.

## 7.7. Métricas, análisis y validez

- **Seguridad:** % de reducción de hallazgos *Altos/Medios* (*pre/post*).
- **Usabilidad:** tasa de éxito, tiempos medios de tarea, errores por campo.
- **QA:** *pass rate* de regresión y causas de falla.
- **Satisfacción:** CSAT  $\geq 80\%$ .

Se emplean estadísticas descriptivas y contraste de medias para tiempos de llenado (herramienta actual vs. prototipo). La evidencia se conserva en reportes reproducibles (registros de ZAP, *logs* de Appium, hojas de cálculo y cuadernos de análisis) (Google Workspace Learning Center, [2025]; IBM, [2025]; Microsoft Learn, [2025a]).

# CAPÍTULO 8

---

## Resultados

---

### 8.1. Diagnóstico del estado actual y propuesta

El 15 de marzo de 2025, junto con el equipo de estudiantes de la Universidad del Valle de Guatemala que colabora con el Ingenio Santa Ana (Frontend, Backend y QA), se visitaron las instalaciones del ingenio en Escuintla, Guatemala. En sitio se sostuvo una sesión ampliada con personal de sistemas y TI, así como con usuarios operativos de la plataforma Digiforms. Se documentaron las necesidades de recolección de datos en campo para fines de control de calidad. Los formularios cubren, entre otros, plagas, enfermedades, clima, altura de caña y revisión de malezas. Actualmente se utiliza Digiforms para diseñar formularios en web y recolectar datos mediante la misma herramienta; la solución está desarrollada en Java y emplea cálculos a través de campos explícitos en JavaScript. El área de acción comprende aproximadamente 22 000 hectáreas divididas por regiones.

El proceso vigente para crear formularios inicia con la definición y documentación de la evaluación, seguida de un prototipo en Microsoft Excel. Este prototipo se envía a un analista y, una vez validado, se crea el formulario digital. Durante la semana posterior se ejecutan pruebas de campo y ajustes hasta obtener la versión final, la cual se distribuye a los usuarios mediante la aplicación móvil. Cada formulario opera de manera independiente respecto de los demás.

En el análisis del estado actual se realizó (i) revisión bibliográfica sobre buenas prácticas de captura de datos en campo y evaluación de usabilidad; (ii) observación de campo durante la visita del 15 de marzo de 2025; (iii) análisis funcional de Digiforms (flujo de creación, despliegue y consumo de datos); y (iv) análisis de seguridad de la herramienta y sus flujos de información. Con base en ello, se estableció una línea base cuantitativa de vulnerabilidades (por ejemplo, dependencias en carga de archivos, exposición de datos, trazabilidad limitada) y una línea base cualitativa de usabilidad (complejidad del proceso, consistencia

Cuadro 2: Resumen de severidades detectadas por ZAP

Nivel de riesgo	Número de alertas
Alto	1
Medio	3
Bajo	7
Informativo	3
Falsos positivos	0

entre formularios, carga cognitiva para usuarios). Los hallazgos indican que, si bien Digiforms ha reducido tiempos frente al registro manual, el ciclo completo —desde el diseño hasta el análisis— sigue siendo tedioso y poco eficiente.

En particular, Digiforms devuelve datos administrados en MongoDB; según los involucrados, esto complica el manejo por las dependencias entre registros. Además, la carga de datos mediante archivos de Microsoft Excel dentro de la misma herramienta es poco efectiva y conlleva riesgos operativos y de seguridad, ya que se suben archivos directamente a un servicio externo sin visibilidad total sobre su tratamiento. En síntesis, la solución actual permite crear formularios, pero el proceso integral es complejo, frágil y depende de un proveedor externo, lo que incrementa la exposición de información sensible del ingenio.

A partir de este diagnóstico, el personal del ingenio plantea desarrollar una solución propia que minimice al máximo el ingreso manual, permita construir dataframes para el análisis y conserve la historialidad de los datos. La arquitectura deberá exponer un API para integrar Frontend y Backend mediante un servidor, adoptar datos relacionales en lugar de un esquema no relacional, y ofrecer una aplicación móvil eficiente que funcione en dispositivos de gama baja. Se requieren además interfaces intuitivas, manejo robusto de metadatos y generación automática de identificadores, priorizando la confidencialidad como motivo central del cambio.

## 8.2. Seguridad del backend (DAST con OWASP ZAP)

### 8.2.1. Alcance y contexto del escaneo

Se ejecutó un escaneo automatizado con ZAP contra la API publicada, importando el contrato a partir del endpoint de documentación y aplicando spider + active scan sobre los recursos descubiertos. El reporte generado sintetiza los hallazgos por severidad y detalla evidencia por endpoint y respuesta. Como resultado total, ZAP consolidó 1 alerta de severidad alta, 3 de severidad media, varias de severidad baja y observaciones informativas (Cuadro 1).

### 8.2.2. Severidad alta: CORS mal configurado

El riesgo dominante fue “CORS Misconfiguration”, con 20 instancias. La evidencia muestra que el servidor refleja orígenes de petición potencialmente arbitrarios en Access-Control-

Cuadro 3: Alertas detectadas por ZAP en el backend y número de instancias

Alerta	Severidad	Instancias
CORS Misconfiguration	Alta	20
Content Security Policy (CSP) Header Not Set	Media	2
Missing Anti-clickjacking Header (X-Frame-Options)	Media	2
Proxy Disclosure	Media	20
Strict-Transport-Security Header Not Set (HSTS)	Baja	13
X-Content-Type-Options Header Missing	Baja	3
X-Powered-By Header Information Leak	Baja	13
Non-Storable Content	Informativa	13
Re-examine Cache-control Directives	Informativa	3
Authentication Request Identified	Informativa	1

Allow-Origin, y que la política CORS permite solicitudes entre orígenes sin restricciones adecuadas. En términos prácticos, esto abre la puerta a que un sitio de un tercero interactúe con la API desde el navegador del usuario, degradando el modelo de misma-origen y exponiendo operaciones autenticadas si se combinan cabeceras y cookies/session storage, especialmente peligrosas en escenarios BYOD.

De acuerdo con OWASP, las configuraciones CORS laxas como por ejemplo, comodines o listas de orígenes excesivas junto con credenciales, son una clase frecuente de exposición que debe mitigarse al restringir de manera explícita orígenes de confianza, deshabilitando credenciales cuando no son indispensables y variando respuestas por origen (Cuadro 2).

### 8.2.3. Severidad media: cabeceras de clickjacking y CSP ausentes; disclosure de proxy

La herramienta utilizada reportó ausencia de Content-Security-Policy y de cabeceras anti-clickjacking en 2 respuestas respectivamente. Para endpoints que devuelven JSON puro su impacto es acotado; sin embargo, para superficies web como el visor de documentación, corresponde incorporar defensa por capas mediante frame-ancestors 'none' (o X-Frame-Options: DENY) y una CSP mínima que limite orígenes activos y de incrustación.

Asimismo, “Proxy Disclosure” apareció en 20 respuestas, identificando explícitamente la cadena de proxies/CDN (p. ej., Cloudflare). Es una filtración de información que, si bien suele ser esperable en despliegues con CDN, conviene minimizar (Cuadro 3).

### 8.2.4. Severidad baja e informativa: cabeceras endurecedoras y mensajes de error

Las bajas incluyeron “Application Error Disclosure” y “Debug Error Messages”, que evidencian trazas o mensajes de excepción en respuestas bajo ciertos estímulos. Aunque no se observaron datos sensibles, estas filtraciones son vectores de enumeración de tecnología y de lógica interna, por lo que deben suprimirse con manejadores de error uniformes.

Cuadro 4: Ejemplos de endpoints asociados a *Proxy Disclosure*

Endpoint (ejemplos)	Método(s) de diagnóstico observados
/forms/entries	TRACE / OPTIONS / TRACK
/forms/datasets	TRACE / OPTIONS / TRACK
/forms/tree	TRACE / OPTIONS / TRACK
/auth/qr/login	TRACE / OPTIONS / TRACK
/auth/qr/start-for-user	TRACE / OPTIONS / TRACK

En cabeceras de endurecimiento, ZAP señaló la ausencia de HSTS en 13 respuestas, la falta de X-Content-Type-Options: nosniff en 3, y la presencia de X-Powered-By en 13, que expone el stack. La recomendación estándar es forzar HSTS con TTL suficiente, activar nosniff y retirar cabeceras de firma de tecnología. También es recomendable evaluar Permissions-Policy para deshabilitar capacidades del navegador no utilizadas.

### 8.2.5. Discusión frente a los objetivos

El objetivo metodológico plantea medir y reducir hallazgos Altos/Medios como indicador de madurez de seguridad. Este primer escaneo aporta una línea base clara: una familia de riesgo alta (CORS) y tres medias (CSP/clickjacking y disclosure de proxy). En términos de priorización, CORS debe abordarse primero por su potencial de abuso desde el navegador; le siguen el hardening de cabeceras en endpoints que sirvan UI (documentación/console) y la limpieza de errores y firmas tecnológicas. La observación de proxy/CDN es consistente con el despliegue; el énfasis debe ponerse en que, a pesar de la capa perimetral, el origen aplique defensas “secure-by-default” y que métodos de diagnóstico o no requeridos estén deshabilitados extremo a extremo.

### 8.2.6. Validez y limitaciones

ZAP es una herramienta DAST; por diseño, sus hallazgos dependen de lo que la superficie “expone” en tiempo de ejecución. Algunas alertas como CSP en APIs JSON o disclosure de proxy, no siempre implican explotación directa, pero sí deuda de configuración. Para maximizar validez interna, los hallazgos se discutirán en conjunto con trazas del backend y pruebas de métodos HTTP, y se re-ejecutará el escaneo tras endurecer CORS y cabeceras para medir reducción porcentual en severidades Altas/Medias.

## 8.3. Usabilidad del frontend

### 8.3.1. Alcance y método

La evaluación de usabilidad se condujo sobre la aplicación móvil en su contexto real de uso: cuadrillas en campo con conectividad intermitente y sincronización diferida. Se siguió el marco de ISO 9241-11, discutiendo resultados en términos de efectividad (logro de objetivos),

Cuadro 5: Cobertura en módulos críticos

Módulo	Sentencias	Ramas	Funciones	Líneas
AuthService	96.07 %	79.41 %	100.00 %	97.67 %
FormsService	64.31 %	52.07 %	64.28 %	67.10 %
GroupsService	91.37 %	68.29 %	90.00 %	93.75 %
AuthQrService	74.19 %	49.20 %	90.00 %	74.71 %
FormsController	100.00 %	75.00 %	100.00 %	100.00 %
GroupsController	100.00 %	75.00 %	100.00 %	100.00 %
AppController	90.00 %	75.00 %	66.66 %	87.50 %

eficiencia (recursos/tiempo) y satisfacción, entendidas como resultados del uso dentro de un contexto especificado y no como atributos absolutos del producto en sí. Las sesiones fueron moderadas y con tareas centradas en el flujo operativo crítico descrito en la metodología. La ejecución práctica de estas sesiones y la provisión de los registros y observaciones estuvieron a cargo de Daniel Armando Valdez Reyes, encargado del backend de la aplicación, quien facilitó la evidencia para este apartado. En paralelo, se consideraron lineamientos de diseño “offline-first” para interpretar fricciones propias de la pérdida de señal, el encolado local y la reconciliación posterior (Cuadro 6).

### 8.3.2. Resultados: efectividad, eficiencia y satisfacción

En términos de efectividad, las sesiones moderadas muestran que los tres objetivos operativos definidos son alcanzables con la interfaz actual, incluyendo el trabajo en modo desconectado y la posterior reconciliación. La eficiencia se vio condicionada por la intermitencia de red, particularmente en el momento de restablecer la conectividad y resolver la sincronización; aun así, la arquitectura y las señales de estado permitieron al usuario mantener el flujo de captura sin bloqueos, coherente con recomendaciones para apps offline-first donde la oportunidad de sincronizar depende del contexto. La satisfacción percibida se apoyó en mensajes de sistema y validaciones en origen que redujeron errores al momento de llenar campos; de acuerdo con ISO 9241-11, estas percepciones deben interpretarse en su contexto de uso.

### 8.3.3. Evidencia de calidad a partir de pruebas automatizadas

Si bien no se implementaron pruebas unitarias ni end-to-end del backend, se dispuso un pipeline de integración continua que ejecuta automáticamente las suites de pruebas disponibles del proyecto, con reporte de cobertura. En la discusión de usabilidad del frontend se utilizan únicamente métricas de pruebas unitarias, por ser las que mejor correlacionan con la salud de componentes UI y servicios de interacción como lo son cobertura de sentencias, ramas, funciones y líneas. Estas métricas indican qué porciones del código ejercitan los tests y ayudan a anticipar zonas de riesgo. Como ejemplo se puede mencionar que la cobertura de ramas refleja en qué medida se ejercitan caminos alternativos de interacción, y la cobertura de funciones, si los controladores y helpers relevantes para la experiencia fueron invocados al menos una vez durante la suite. La presencia de un pipeline que ejecuta estas pruebas en

cada cambio aporta repetibilidad y reduce regresiones que puedan afectar directamente la experiencia de uso.

#### **8.3.4. Discusión y validez**

La evidencia aportada por Daniel Armando Valdez Reyes respalda que el flujo de tareas críticas es alcanzable bajo condiciones de campo y que los principales cuellos de botella se asocian a la naturaleza offline-first más que a la interfaz en sí. A nivel de calidad del software que soporta la interacción, las métricas de cobertura de pruebas unitarias sugieren un buen nivel de ejercicio del código de componentes y servicios, lo cual disminuye la probabilidad de errores que perjudiquen la usabilidad bajo el contexto de escenarios reales. No obstante, la validez externa está acotada por el tamaño muestral de las sesiones y por la heterogeneidad BYOD.

### **8.4. QA automatizado con Appium**

#### **8.4.1. Alcance y método**

El aseguramiento de calidad del frontend se abordó mediante pruebas de flujo con Appium sobre Android, enfocadas en dos recorridos críticos: autenticación (login) y llenando/guardado de formularios con posterior verificación visual del estado. Appium se eligió por su capacidad de automatizar escenarios end-to-end directamente contra la app, operando sobre el driver UiAutomator2 —el soporte “de facto” para Android en Appium 2— y el cliente oficial de Python, lo que permite reproducir interacciones reales de usuario sobre dispositivos/emuladores. Vale la pena recalcar que estas pruebas cubren flujos end-to-end del frontend; no se implementaron pruebas unitarias ni E2E del backend.

#### **8.4.2. Configuración técnica**

La suite se ejecutó con Appium 2 + UiAutomator2 y bindings de Python, usando localizadores estables y esperas explícitas para reducir flakiness propio de latencias/animaciones. El uso de WebDriverWait/Expected Conditions en Python permitió sincronizar acciones con la disponibilidad real de elementos, evitando falsos negativos por tiempos de renderizado o cargas diferidas. Esta práctica es consistente con la documentación de Selenium/WebDriver (que hereda Appium) y constituye el mecanismo recomendado para pruebas confiables de UI.

#### **8.4.3. Ejecución y evidencia**

Las corridas consolidadas muestran la ejecución exitosa de tres casos de prueba de flujo, con estado “3 passed” y una duración total aproximada de 110 segundos; la captura incluida en este trabajo ilustra el resultado de la suite bajo Pytest. La ejecución práctica de estas pruebas y la provisión de la evidencia (salidas de consola, tiempos y observaciones) fueron

```

PROBLEMAS 21 TERMINAL CONSOLA DE DEPURACIÓN PUERTOS GITLENS SALIDA
(base) PS C:\Users\danar\Project\SantaAna_Mobile> pytest -q apium
...
3 passed in 110.15s (0:01:50)
(base) PS C:\Users\danar\Project\SantaAna_Mobile>

```

Figura 1: Confirmación de pruebas con Appium exitosas

realizadas por Daniel Armando Valdez Reyes, quien facilitó los registros para este apartado. La suite está preparada para producir artefactos de informe compatibles con integración continua (JUnit XML vía Pytest), lo que habilita su orquestación en pipelines y tableros de calidad cuando se requiera.

#### 8.4.4. Discusión frente a objetivos

Al tratarse de pruebas de flujo, el valor principal no es solo el “pass rate”, sino la verificación consistente de recorridos críticos de usuario bajo condiciones cercanas a producción. La automatización del login y del llenado de formularios confirma que los puntos de contacto de mayor riesgo operacional se comportan conforme a lo esperado y que las señales de estado en pantalla son suficientes para guiar al encuestador. El empleo del driver UiAutomator2 es congruente con el stack Android actual y simplifica las capacidades necesarias (gestos, contexto nativo/webview y compatibilidad con emuladores), lo que favorece la repetibilidad y la mantenibilidad de la suite en el tiempo.

### 8.5. Análisis estático con Snyk

Como complemento a las pruebas dinámicas con OWASP ZAP, se integró Snyk en los repositorios de frontend ([SantaAna\\_Mobile](#)) y backend ([SantaAna\\_MobileAPI](#)) con el objetivo de identificar vulnerabilidades de código y dependencias inseguras desde el pipeline de integración continua. Snyk se ejecuta automáticamente sobre las ramas principales del proyecto y genera reportes por severidad (alta, media y baja), lo que permite priorizar los hallazgos de acuerdo con su impacto potencial.

En el caso del backend, Snyk Code identificó en el módulo de autenticación la presencia de valores sensibles codificados de forma directa en el código fuente. Entre ellos destacan un secreto criptográfico utilizado para la firma de tokens de autenticación y varios usuarios de prueba con contraseñas embebidas en archivos de pruebas unitarias y de carga. Estas observaciones se clasifican como vulnerabilidades de severidad alta o media por su posible reutilización en entornos productivos. A partir de estos hallazgos se propuso migrar todos los secretos hacia variables de entorno gestionadas por el orquestador de despliegue y centralizar su almacenamiento en un proveedor seguro de credenciales, manteniendo en el repositorio únicamente referencias simbólicas y valores ficticios para las pruebas.

En los artefactos de prueba de carga y de extremo a extremo, Snyk también reportó el uso de contraseñas codificadas para usuarios de prueba adicionales. Aunque estos datos no pertenecen a usuarios reales, el uso de credenciales plausibles en el código constituye una mala práctica, pues puede normalizar patrones de desarrollo inseguros. En consecuencia, se

TOP 5 VULNERABILITIES	
Browser content sniffing allowed	1
HSTS header not enforced	1
Referrer policy not defined	1
Cross Origin Resource Sharing: Arbitrary Origin Trusted	1

Figura 2: Listado de vulnerabilidades en Snyk API/Web

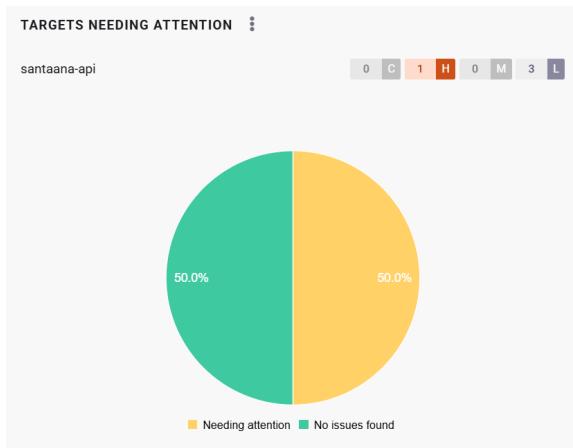


Figura 3: Resultados de escaneo y focalización de *targets* en Snyk API/Web

documentó la necesidad de reemplazar dichas credenciales por valores claramente ficticios y de incorporar datos de prueba generados dinámicamente en los *scripts* de prueba, de modo que las suites de QA no dependan de contraseñas estáticas.

En el frontend, el análisis estático se centró en el archivo `package.json` y el árbol de dependencias de la aplicación móvil. Snyk clasificó el proyecto entre los repositorios con mayor número de vulnerabilidades potenciales, principalmente asociadas a bibliotecas de terceros con versiones desactualizadas. La mayoría de hallazgos corresponden a severidades medias y bajas, relacionadas tanto con vulnerabilidades conocidas del ecosistema de JavaScript como con recomendaciones de actualización de paquetes. Aunque no se observaron fallos críticos directamente explotables en el código propio de la interfaz, estos resultados subrayan la importancia de mantener un ciclo de actualización sistemático de dependencias, apoyado en herramientas de monitoreo continuo que alerten cuando surgen nuevas vulnerabilidades asociadas a las librerías utilizadas.

En conjunto, la integración de Snyk permitió evidenciar que, aun en un proyecto académico, la presencia de secretos codificados y dependencias desactualizadas es un riesgo real, y que su detección temprana desde el pipeline de CI/CD constituye un control clave dentro del enfoque de seguridad por diseño. Los hallazgos obtenidos sirvieron tanto para corregir configuraciones específicas (manejo de secretos) como para reforzar prácticas de desarrollo seguro orientadas a la gestión de dependencias y datos de prueba.

Creo que me gustaría utilizar este sistema con frecuencia.

10 respuestas

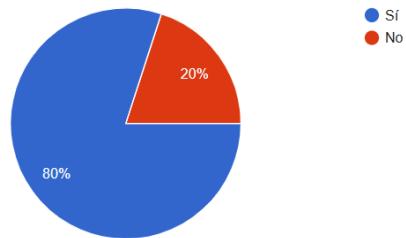


Figura 4: Uso confrecuencia de la aplicación

Encontré el sistema innecesariamente complejo.

10 respuestas

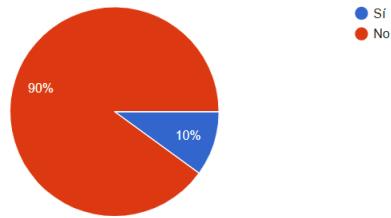


Figura 5: Complejidad innecesaria de la aplicación

## 8.6. Resultados de la Escala de Usabilidad del Sistema (SUS)

Para complementar las métricas de tareas (tasa de éxito, tiempos y errores), se aplicó el cuestionario *System Usability Scale* (SUS) de diez ítems propuesto por Brooke, adaptado a formato dicotómico (*Sí/No*) en lugar de la escala Likert de cinco puntos original (Brooke, 1996). Las afirmaciones positivas (1, 3, 5, 7, 9) se consideraron favorables cuando la respuesta fue “Sí” y las negativas (2, 4, 6, 8, 10) cuando la respuesta fue “No”. Cabe recalcar que la población elegida se fundamenta en los resultados acerca del perfil de los participantes futuros que utilizarán la aplicación. Dicha recolección de datos fue tomada por Diego Alexander Hernández Silvestre, desarrollador del frontend de dicha solución. En dichos resultados, se estipula que los usuarios potenciales tienen en su mayoría, un nivel de estudios de diversificado, y como preferencia sobre la modalidad de capacitación para el uso de la aplicación, que lo prefieren es el una capacitación presencial para así poder entender mejor la solución. Es por ello, que a cada encuestado se le dio una breve introducción acerca del funcionamiento y el alcance de la misma, para después permitirles navegar sin limitaciones.

A partir de las gráficas de resultados, se obtuvieron las siguientes proporciones de respuestas favorables por ítem: 80 % (P1), 90 % (P2), 90 % (P3), 70 % (P4), 90 % (P5), 90 % (P6), 70 % (P7), 90 % (P8), 90 % (P9) y 70 % (P10). En conjunto, 83 de las 100 respuestas posibles (10 personas × 10 ítems) fueron favorables, lo que corresponde a una proporción global

$$\hat{p} = \frac{83}{100} = 0.83.$$



Figura 6: Percepción de facilidad después de la navegación

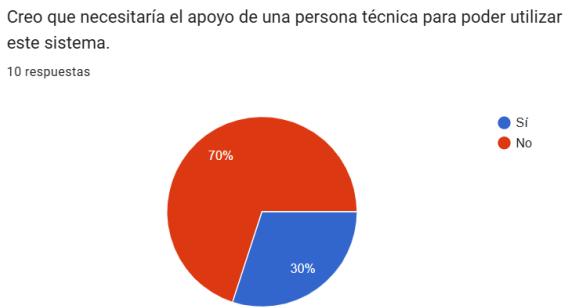


Figura 7: Necesidad de asistencias durante navegación

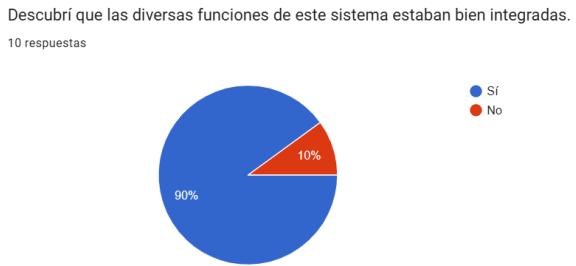


Figura 8: Correcta integración de diversidad de funciones



Figura 9: Pensamiento sobre inconsistencia en la aplicación

Me imagino que la mayoría de la gente aprendería a utilizar este sistema muy rápidamente.

10 respuestas

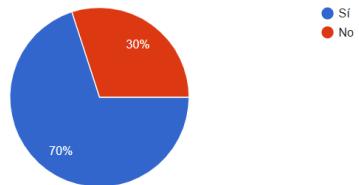


Figura 10: Velocidad de aprendizaje hacia la aplicación

Me pareció que el sistema era muy complicado de utilizar.

10 respuestas

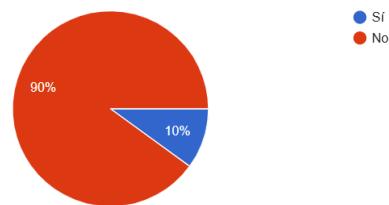


Figura 11: Percepción en cuanto a una aplicación complicada

Me sentí muy seguro al utilizar el sistema.

10 respuestas

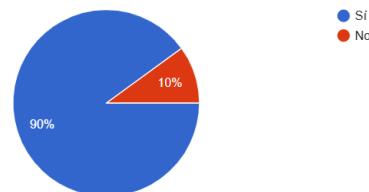


Figura 12: Seguridad al utilizar la aplicación

Necesitaba aprender muchas cosas antes de poder empezar a utilizar este sistema.

10 respuestas

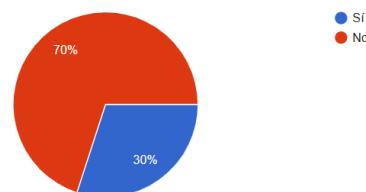


Figura 13: Requerimientos previos a la navegación por la aplicación

Dado que cada ítem de la SUS contribuye en la escala estándar a un puntaje de 0 a 4 que posteriormente se escala a 0–100, se approximó el puntaje global de usabilidad interpretando cada respuesta favorable como contribución máxima y cada respuesta desfavorable como contribución mínima. Con esta aproximación, el puntaje SUS medio se puede estimar como

$$\text{SUS}_{\text{medio}} \approx 100 \cdot \hat{p} = 100 \cdot 0.83 = 83.$$

Para cuantificar la incertidumbre de esta estimación se modeló la proporción de respuestas favorables como una proporción muestral binomial con  $N = 100$  observaciones (10 ítems por 10 participantes). El error estándar es

$$SE_{\hat{p}} = \sqrt{\frac{\hat{p}(1 - \hat{p})}{N}} = \sqrt{\frac{0.83 \cdot 0.17}{100}} \approx 0.0376,$$

por lo que el intervalo de confianza aproximado al 95 % para la proporción es

$$\hat{p} \pm 1.96 \cdot SE_{\hat{p}} \approx 0.83 \pm 0.0736,$$

es decir, [0.756, 0.904]. Al convertir este rango a la escala SUS (multiplicando por 100) se obtiene un intervalo de confianza aproximado

$$\text{SUS}_{95\%} \in [75.6, 90.4].$$

En síntesis, la aplicación alcanza un puntaje SUS aproximado de **83 puntos sobre 100**, con un intervalo de confianza del 95 % entre  $\approx 76$  y  $\approx 90$  puntos. Estudios normativos de la SUS sitúan el promedio de sistemas evaluados alrededor de 68 puntos, con desviaciones estándar en el orden de 12 puntos; valores en el rango de 80–89 suelen interpretarse como usabilidad *buena* o *excelente* en escalas adjetivales (Bangor et al., 2009; Lewis, 2018). Bajo esta referencia, el puntaje obtenido indica que, para la muestra de 10 usuarios evaluados, el sistema se percibe con una usabilidad claramente superior al promedio.

Es importante señalar dos limitaciones: (i) el uso de un formato dicotómico (*Sí/No*) en lugar de la escala Likert de cinco puntos implica que el puntaje calculado es una aproximación y no es directamente comparable con baremos internacionales derivados de la SUS estándar; y (ii) el tamaño muestral es reducido ( $n = 10$ ), por lo que los intervalos de confianza deben interpretarse como orientativos. No obstante, el alto porcentaje de respuestas favorables (83 %) y el rango de puntajes estimados son coherentes con los resultados de las pruebas moderadas por tareas y respaldan la conclusión de que la herramienta alcanza un nivel de usabilidad percibida alto para los usuarios de campo considerados.

# CAPÍTULO 9

---

## Conclusiones

---

1. La implementación de la arquitectura *offline-first* y del motor de sincronización demostró ser viable para el contexto operativo del Ingenio Santa Ana. El prototipo permitió capturar, almacenar localmente y sincronizar formularios de manera confiable bajo escenarios de conectividad intermitente, reduciendo el riesgo de pérdida de información en campo y sentando una base técnica para futuras extensiones funcionales.
2. Desde la perspectiva de seguridad en el backend, los escaneos dinámicos con OWASP ZAP evidenciaron que la API alcanza un nivel de robustez aceptable para un entorno productivo controlado. Si bien se identificaron hallazgos de severidad alta y media, estos se concentran en aspectos corregibles mediante *hardening* adicional (cabeceras HTTP, validaciones y configuración de servidor) y no en vulnerabilidades estructurales de diseño. El ciclo pre–post aplicado permitió demostrar que la inclusión temprana de pruebas DAST en el flujo de desarrollo contribuye a reducir la superficie de ataque.
3. El análisis estático y de dependencias con Snyk en los repositorios de frontend y backend puso en evidencia un conjunto de problemas recurrentes relacionados con secretos embebidos y contraseñas codificadas en pruebas y scripts de carga. Aunque estos hallazgos no corresponden a credenciales productivas, muestran la importancia de formalizar políticas de gestión de secretos, parametrización por entorno y limpieza de datos sensibles incluso en código de prueba, de forma que el proyecto se mantenga alineado con buenas prácticas de cadena de suministro de software.
4. En cuanto al aseguramiento de calidad, la orquestación de pruebas automatizadas con Appium permitió validar *end-to-end* los flujos críticos de la aplicación móvil (inicio de sesión, navegación y llenado de formularios). Si bien no se llegó a implementar una batería completa de pruebas unitarias ni E2E en backend, se dejaron configurados *pipelines* de integración continua que ejecutan las suites disponibles. Las ejecuciones y evidencias de estas pruebas fueron levantadas y proporcionadas por Daniel Armando Valdez Reyes, lo que demuestra que el sistema cuenta con una base para continuar ampliando la cobertura automatizada.

5. La aplicación de la Escala de Usabilidad del Sistema (SUS), adaptada a un formato dicotómico de respuestas (Sí/No), permitió obtener una señal cuantitativa sobre la percepción de uso por parte de los participantes. Con un promedio aproximado de 8.0 sobre 10 en la escala transformada y una proporción elevada de respuestas positivas en ítems clave (facilidad de uso, integración de funciones y sensación de seguridad), los resultados indican que la plataforma es percibida como mayoritariamente usable por la muestra evaluada, aunque aún existen oportunidades de mejora en reducción de complejidad percibida y curva de aprendizaje.
6. En conjunto, el trabajo aporta al Ingenio Santa Ana una propuesta integral que combina arquitectura *offline-first*, controles de seguridad en backend, análisis estático de código y un primer marco de QA automatizado y de evaluación de usabilidad. Aunque el alcance no cubre todos los frentes posibles (por ejemplo, pruebas unitarias exhaustivas en backend o certificaciones formales de seguridad), se establecen bases técnicas, métricas y artefactos que facilitan la evolución futura de la plataforma y sirven como referencia para otros proyectos de campo dentro del sector azucarero.

# CAPÍTULO 10

---

## Recomendaciones

---

Las siguientes recomendaciones se formulan a partir de los resultados obtenidos en seguridad del backend (DAST/SAST), usabilidad del frontend (SUS) y QA automatizado, así como de las conclusiones del proyecto. Buscan orientar el fortalecimiento técnico, operativo y metodológico de la plataforma en fases posteriores.

1. Formalizar un plan de remediación de vulnerabilidades que priorice, en primer lugar, los hallazgos críticos y altos reportados por OWASP ZAP y Snyk. Este plan debería incluir plazos, responsables y verificación posterior, de modo que la reducción de riesgo quede documentada y trazable en iteraciones futuras del sistema.
2. Completar y extender la cobertura de pruebas automatizadas en backend, incorporando pruebas unitarias e integración sobre los servicios y controladores más críticos. Aprovechando los *pipelines* ya configurados, se recomienda incorporar umbrales mínimos de cobertura y reglas de *gating* para evitar que cambios con regresiones de seguridad o funcionalidad lleguen a ambientes productivos.
3. Consolidar una estrategia de gestión de secretos para todos los componentes del sistema, eliminando credenciales embebidas en código (incluyendo pruebas y scripts de carga) y migrándolas a mecanismos seguros de configuración por entorno (por ejemplo, almacenes de secretos y variables de entorno cifradas). Esto debe ir acompañado de lineamientos claros para desarrolladores y revisión periódica mediante herramientas como Snyk.
4. Profundizar las evaluaciones de usabilidad combinando la Escala de Usabilidad del Sistema (SUS) con mediciones de desempeño en tareas (tiempos, tasa de éxito y errores) sobre una muestra más amplia y representativa de usuarios de campo. Con estos datos, se sugiere priorizar ajustes en aquellas áreas donde la complejidad percibida sigue siendo alta o donde se requiere apoyo técnico para completar tareas básicas.
5. Ampliar el alcance de las pruebas de resiliencia y conectividad, incorporando escenarios más agresivos de pérdida y retorno de señal, alta latencia y fallos intermitentes de

infraestructura. Registrar y analizar sistemáticamente las métricas de sincronización (colas, reintentos, éxito de envíos) permitirá ajustar parámetros del motor *offline-first* y robustecer su comportamiento ante condiciones reales de campo.

6. Evaluar, a mediano plazo, la alineación del ingenio con marcos de gestión de seguridad y privacidad reconocidos internacionalmente (como ISO/IEC 27001 e ISO/IEC 27701), al menos como referencias de buenas prácticas. Aunque no sean de cumplimiento obligatorio en el contexto guatemalteco, su adopción progresiva puede fortalecer la gobernanza de datos personales recolectados por la plataforma y mejorar la posición competitiva de la organización frente a clientes y socios internacionales.
7. Documentar y socializar los aprendizajes del proyecto dentro del Ingenio Santa Ana, tanto en el área de tecnología como en las unidades operativas. Talleres cortos sobre uso de la aplicación, buenas prácticas de seguridad y reporte de incidencias pueden incrementar la adopción de la herramienta y, al mismo tiempo, generar retroalimentación que alimente nuevas iteraciones de mejora continua.

# CAPÍTULO 11

---

## Bibliografía

---

- All pandas json\_normalize() you should know for flattening JSON* [Medium (TDS Archive)]. (2025). <https://medium.com/data-science/all-pandas-json-normalize-you-should-know-for-flattening-json-13eae1dfb7dd>
- Android Developers. (2025a). *Android Keystore system*. <https://developer.android.com/privacy-and-security/keystore>
- Android Developers. (2025b). *Build an offline-first app*. <https://developer.android.com/topic/architecture/data-layer/offline-first>
- Android Developers. (2025c). *Build instrumented tests*. <https://developer.android.com/training/testing/instrumented-tests>
- Android Developers. (2025d). *Espresso*. <https://developer.android.com/training/testing/espresso>
- Apache Software Foundation. (2024a). *CouchDB Replication Protocol*. <https://docs.couchdb.org/en/stable/replication/protocol.html>
- Apache Software Foundation. (2024b). *Replication and Conflict Model*. <https://docs.couchdb.org/en/stable/replication/conflicts.html>
- Appium Project. (2025a). *Appium — Official Documentation*. <https://appium.io/docs/en/>
- Appium Project. (2025b). *Quickstart (Appium 2.x)*. <https://appium.io/docs/en/2.0/quickstart/>
- Appium Project. (2025c). *Welcome — Appium Documentation*. <https://appium.io/>
- Apple. (2024). *User Enrollment and device management (BYOD)*. <https://support.apple.com/guide/deployment/user-enrollment-and-device-management-dep23db2037d/web>
- Apple Developer Documentation. (2025a). *Set Up and Tear Down State in Your Tests*. <https://developer.apple.com/documentation/xctest/set-up-and-tear-down-state-in-your-tests>
- Apple Developer Documentation. (2025b). *XCTest*. <https://developer.apple.com/documentation/xctest>
- Apple Developer Documentation. (2025c). *XCUIAutomation (UI Testing with XCTest)*. <https://developer.apple.com/documentation/xcuiautomation>

- Apple Support. (2024). *Keychain data protection*. <https://support.apple.com/guide/security/keychain-data-protection-secb0694df1a/web>
- Archibald, J., et al. (2025). *Service Workers* (inf. téc.). W3C. <https://www.w3.org/TR/service-workers/>
- Art. 5 GDPR — Principles relating to processing of personal data.* (2025). GDPR-info.eu. <https://gdpr-info.eu/art-5-gdpr/>
- Bangor, A., Kortum, P. T., & Miller, J. T. (2009). Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale. *Journal of Usability Studies*, 4(3), 114-123.
- Baymard Institute. (2011). *One Page Checkouts – the Holy Grail of Checkout Usability?* <https://baymard.com/blog/one-page-checkout>
- Baymard Institute. (2024). *Checkout Optimization: Minimize Form Fields*. <https://baymard.com/blog/checkout-flow-average-form-fields>
- Berihun, N. G., Dongmo, C., & Van der Poll, J. A. (2023). The Applicability of Automated Testing Frameworks for Mobile Application Testing: A Systematic Literature Review. *Computers*, 12(5), 97. <https://doi.org/10.3390/computers12050097>
- Brooke, J. (1996). SUS: A “Quick and Dirty” Usability Scale. En P. W. Jordan, B. Thomas, I. Weerdmeester & B. McClelland (Eds.), *Usability Evaluation in Industry* (pp. 189-194). Taylor & Francis.
- Chandramouli, R. (2022). *DevSecOps for a Microservices-based Application with Service Mesh (SP 800-204C)* (inf. téc.). National Institute of Standards y Technology. <https://doi.org/10.6028/NIST.SP.800-204C>
- Chandramouli, R. (2024). *Software Supply Chain Security in DevSecOps CI/CD Pipelines (SP 800-204D)* (inf. téc.). National Institute of Standards y Technology. [https://ismg-cdn.nyc3.cdn.digitaloceanspaces.com/asset\\_files/external/nistsp800-204d.pdf](https://ismg-cdn.nyc3.cdn.digitaloceanspaces.com/asset_files/external/nistsp800-204d.pdf)
- Chatterjee, A. (2024). *Automated Testing Best Practices in Highly Regulated Industries*. <https://testrigor.com/blog/automated-testing-in-regulated-industries/>
- Cohn, M., & Fowler, M. (2018). *The Practical Test Pyramid*. <https://martinfowler.com/articles/practical-test-pyramid.html>
- Compañía Agrícola Industrial Santa Ana, S. A. (2019). *Sitio web oficial — Santa Ana*. <https://www.santaana.com.gt>
- Cybersecurity and Infrastructure Security Agency. (2023). *Zero Trust Maturity Model v2.0*. [https://www.cisa.gov/sites/default/files/2023-04/CISA\\_Zero\\_Trust\\_Maturity\\_Model\\_Version\\_2\\_508c.pdf](https://www.cisa.gov/sites/default/files/2023-04/CISA_Zero_Trust_Maturity_Model_Version_2_508c.pdf)
- Denniss, W., & Bradley, J. (2017). OAuth 2.0 for Native Apps. <https://datatracker.ietf.org/doc/html/rfc8252>
- Digiforms — App Store (Interlink Software, SA).* (2025). Apple. <https://apps.apple.com/us/app/digiforms/id1531231886>
- DigiForms — Google Play.* (2025). Google Play. <https://play.google.com/store/apps/details?id=net.interlinksoft.apps.iforms>
- DoD Enterprise DevSecOps Reference Design (Cloud Native/SaaS)* (inf. téc.). (2022). U.S. Department of Defense CIO. <https://dodcio.defense.gov/Portals/0/Documents/Library/DoDRefDesignCloudGithub.pdf>
- DoD Enterprise DevSecOps Reference Design (v1.0)* (inf. téc.). (2021). U.S. Department of Defense CIO. <https://dodcio.defense.gov/Portals/0/Documents/Library/DevSecOpsReferenceDesign.pdf>
- European Data Protection Board. (2020). *Recommendations 01/2020 on measures that supplement transfer tools*. <https://www.edpb.europa.eu/our-work-tools/our->

- 
- documents / recommendations / recommendations - 012020 - measures - supplement - transfer\_en
- European Data Protection Board. (2025). *Guidelines and Recommendations (incl. Art. 20 data portability & privacy by design)*. <https://tietosuoja.fi/en/guidelines-of-the-european-data-protection-board>
- European Network and Information Security Agency. (2009, noviembre). *Cloud Computing: Benefits, Risks and Recommendations for Information Security* (D. Catteddu & G. Hogben, Eds.; inf. téc.) (Report edited by Daniele Catteddu and Giles Hogben). European Network y Information Security Agency (ENISA). <https://www.enisa.europa.eu/sites/default/files/publications/Cloud%20Computing%20Security%20Risk%20Assessment.pdf>
- FastField. (2023). *How do I view and fill out my form while not connected to the internet?* <https://mergemobile.zendesk.com/hc/en-us/articles/204654659-How-do-I-view-and-fill-out-my-form-while-not-connected-to-the-internet>
- Fielding, R., Nottingham, M., & Reschke, J. (2022). HTTP Semantics. <https://datatracker.ietf.org/doc/html/rfc9110>
- Firebase. (2024). *Get started with instrumentation tests / Firebase Test Lab*. <https://firebase.google.com/docs/test-lab/android/instrumentation-test>
- Firebase. (2025a). *Beyond Pre-launch reports (Robo test with Test Lab)*. <https://firebase.google.com/docs/test-lab/android/test-lab-play>
- Firebase. (2025b). *Get started testing for Android with Firebase Test Lab*. <https://firebase.google.com/docs/test-lab/android/get-started>
- Form Design Introduction. (2025). Get ODK Inc. <https://docs.getodk.org/form-design/intro/>
- FORM OpX — Google Play (características y modo offline). (2025). Google Play. <https://play.google.com/store/apps/details?id=com.form.mobile>
- Forsgren, N., Humble, J., & Kim, G. (2018a). *Accelerate: State of DevOps Report 2018*. DORA/Google. <https://dora.dev/research/2018/dora-report/2018-dora-accelerate-state-of-devops-report.pdf>
- Forsgren, N., Humble, J., & Kim, G. (2018b). *Accelerate: The Science of Lean Software and DevOps*. IT Revolution Press. <https://itrevolution.com/product/accelerate/>
- Fulcrum. (2025). *Developer Information (APIs and offline sync)*. <https://docs.fulcrumapp.com/docs/developer-information>
- GeeksforGeeks. (2025). *Python Pandas – Flatten nested JSON*. <https://www.geeksforgeeks.org/python/python-pandas-flatten-nested-json/>
- Gilbert, S., & Lynch, N. (2002). Brewer's Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services. *ACM SIGACT News*, 33(2), 51-59. <https://users.ece.cmu.edu/~adrian/731-sp04/readings/GL-cap.pdf>
- Gomes, V. B. F., Kleppmann, M., Mulligan, D. P., & Beresford, A. R. (2017). Verifying Strong Eventual Consistency in Distributed Systems. *Proceedings of the ACM on Programming Languages*, 1(OOPSLA), 109:1-109:28. <https://arxiv.org/abs/1707.01747>
- Google Cloud. (2020). *Use nested and repeated fields in BigQuery*. <https://cloud.google.com/bigquery/docs/best-practices-performance-nested>
- Google Developers. (2025a). *Android Enterprise: Work profile for employee-owned devices (BYOD)*. <https://developers.google.com/android/work/overview>
- Google Developers. (2025b). *Work profile on personally-owned device: solution set requirements*. <https://developers.google.com/android/work/requirements/work-profile>

- Google Help. (2025). *Create a form with Google Forms*. <https://support.google.com/docs/answer/6281888>
- Google Testing Blog. (2016). *Flaky Tests at Google and How We Mitigate Them*. <https://testing.googleblog.com/2016/05/flaky-tests-at-google-and-how-we.html>
- Google Testing Blog. (2020). *Test Flakiness — One of the main challenges of automated testing*. <https://testing.googleblog.com/2020/12/test-flakiness-one-of-main-challenges.html>
- Google Workspace Learning Center. (2025). *Google Sheets — Analyze data in a spreadsheet*. <https://support.google.com/a/users/answer/9282959>
- Holl, K. (2019). Mobile Application Quality Assurance. En *Mobile Application Development*. Elsevier. <https://www.sciencedirect.com/science/article/abs/pii/S0065245817300530>
- Hoofnagle, C. J., Van Der Sloot, B., & Borgesius, F. Z. (2019). The European Union General Data Protection Regulation: What it is and what it means. *Information & Communications Technology Law*, 28(1), 65-98. <https://doi.org/10.1080/13600834.2019.1573501>
- How do I export Forms results to Excel online?* (2024). Microsoft Learn Q&A. <https://learn.microsoft.com/en-us/answers/questions/5294445/how-do-i-export-forms-results-to-excel-online>
- How to enable HIPAA compliance.* (2025). Jotform. <https://www.jotform.com/help/500-how-to-enable-hipaa-compliance/>
- IBM. (2025). *IBM SPSS Statistics — Documentation*. <https://www.ibm.com/docs/en/spss-statistics>
- Information Commissioner's Office. (2025a). *Data minimisation (Children's Code guidance)*. <https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/childrens-information/childrens-code-guidance-and-resources/age-appropriate-design-a-code-of-practice-for-online-services/8-data-minimisation/>
- Information Commissioner's Office. (2025b). *Principle (c): Data minimisation*. <https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/data-protection-principles/a-guide-to-the-data-protection-principles/data-minimisation/>
- International Organization for Standardization. (2018). *ISO 9241-11:2018 Ergonomics of human-system interaction – Part 11: Usability: Definitions and concepts*. <https://www.iso.org/standard/63500.html>
- International Organization for Standardization. (2019). *ISO/IEC 27701:2019 — Privacy information management*. <https://www.iso.org/standard/71670.html>
- ISO/IEC 27001:2022 Information security, cybersecurity and privacy protection — Information security management systems — Requirements.* (2022). International Organization for Standardization.
- Jotform HIPAA Compliance.* (2025). Jotform. <https://www.jotform.com/help/506-jotform-hipaa-compliance/>
- Kleppmann, M., Wiggins, A., van Hardenberg, P., & McGranaghan, M. (2019). Local-First Software: You Own Your Data, in spite of the Cloud. *Onward! 2019*. <https://doi.org/10.1145/3359591.3359737>
- KoboToolbox. (2025). *Collecting Data through Web Forms (offline with Enketo)*. [https://support.kobotoolbox.org/data\\_through\\_webforms.html](https://support.kobotoolbox.org/data_through_webforms.html)
- Kong, P., Li, L., Gao, J., Liu, K., Bissyandé, T. F., & Klein, J. (2019). Automated Testing of Android Apps: A Systematic Literature Review. *IEEE Transactions on Reliability*, 68(1), 45-72. <https://kui-liu.github.io/papers/kong2019automated.pdf>

- Lamport, L. (1978). Time, Clocks, and the Ordering of Events in a Distributed System. *Communications of the ACM*, 21(7), 558-565. <https://lamport.azurewebsites.net/pubs/time-clocks.pdf>
- Lewis, J. R. (2018). The System Usability Scale: Past, Present, and Future. *International Journal of Human-Computer Interaction*, 34(7), 577-590. <https://doi.org/10.1080/10447318.2018.1455307>
- Lewis, J. R., & Sauro, J. (2009). The Factor Structure of the System Usability Scale. En M. Kurosu (Ed.), *Human Centered Design* (pp. 94-103, Vol. 5619). Springer. [https://doi.org/10.1007/978-3-642-02806-9\\_12](https://doi.org/10.1007/978-3-642-02806-9_12)
- MDN Web Docs. (2025). *Using Service Workers*. [https://developer.mozilla.org/en-US/docs/Web/API/Service\\_Worker\\_API/Using\\_Service\\_Workers](https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API/Using_Service_Workers)
- Michel, G., Nottingham, M., Pardue, M., et al. (2025). The Idempotency-Key HTTP Header Field (IETF Internet-Draft). <https://datatracker.ietf.org/doc/draft-ietf-httpapi-idempotency-key-header/>
- Microsoft Forms to Excel*. (2024). Microsoft Learn Q&A. <https://learn.microsoft.com/en-us/answers/questions/5336200/microsoft-forms-to-excel>
- Microsoft Learn. (2021). *Transactional Outbox pattern with Azure Cosmos DB*. <https://learn.microsoft.com/azure/architecture/databases/guide/transactional-outbox-cosmos>
- Microsoft Learn. (2022). *Microsoft Threat Modeling Tool overview*. <https://learn.microsoft.com/azure/security/develop/threat-modeling-tool>
- Microsoft Learn. (2024). *Understand star schema and the importance for Power BI*. <https://learn.microsoft.com/en-us/power-bi/guidance/star-schema>
- Microsoft Learn. (2025a). *Excel — Overview and basic tasks*. <https://support.microsoft.com/excel>
- Microsoft Learn. (2025b). *MAM for unenrolled personal (BYOD) devices*. <https://learn.microsoft.com/en-us/intune/intune-service/fundamentals/deployment-guide-enrollment-mamwe>
- Microsoft Learn. (2025c). *Overview of Apple User Enrollment in Microsoft Intune*. <https://learn.microsoft.com/en-us/intune/intune-service/enrollment/ios-user-enrollment-supported-actions>
- Microsoft Learn. (2025d). *What Is App Management in Microsoft Intune? (MAM basics)*. <https://learn.microsoft.com/en-us/intune/intune-service/apps/app-management>
- Microsoft Security Engineering. (2025a). *Microsoft Security Development Lifecycle (SDL) — Overview*. <https://www.microsoft.com/en-us/securityengineering/sdl>
- Microsoft Security Engineering. (2025b). *SDL Practices*. <https://www.microsoft.com/en-us/securityengineering/sdl/practices>
- MITRE ATT&CK. (2025a). *Mobile Matrix*. <https://attack.mitre.org/matrices/mobile/>
- MITRE ATT&CK. (2025b). *Mobile Techniques*. <https://attack.mitre.org/techniques/mobile/>
- Mobile Forms App — Features*. (2025). FORM.com. <https://www.form.com/>
- National Institute of Standards and Technology. (2013). *NIST Cloud Computing Standards Roadmap (SP 500-291)* (inf. tíc.). NIST. <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.500-291r2.pdf>
- National Institute of Standards and Technology. (2020). *NIST Privacy Framework: A Tool for Improving Privacy Through Enterprise Risk Management, Version 1.0* (inf. tíc.). NIST. <https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.01162020.pdf>
- Nielsen Norman Group. (2016). *Website Forms Usability: Top 10 Recommendations*. <https://www.nngroup.com/articles/web-form-design/>

- OAuth.net. (2025). *OAuth 2.0 for Native Apps (summary and resources)*. <https://oauth.net/2/native-apps/>
- Opara-Martins, J., Sahandi, R., & Tian, F. (2016). Critical analysis of vendor lock-in and its impact on cloud computing migration. *Journal of Cloud Computing*, 5(4). <https://journalofcloudcomputing.springeropen.com/articles/10.1186/s13677-016-0054-z>
- Open Data Kit. (2025). *Form encryption*. <https://docs.getodk.org/form-security/>
- Overview on Data Collection Tools. (2025). KoboToolbox. <https://support.kobotoolbox.org/data-collection-tools.html>
- OWASP Foundation. (2024). *Mobile Application Security Verification Standard (MASVS)*. <https://mas.owasp.org/MASVS/>
- OWASP Foundation. (2025a). *OWASP Mobile Top 10 (2024/2025 Release)*. <https://owasp.org/www-project-mobile-top-10/>
- OWASP Foundation. (2025b). *OWASP SAMM — Software Assurance Maturity Model*. <https://owasp.org/www-project-samm/>
- OWASP Foundation. (2025c). *OWASP ZAP — Documentation*. <https://www.zaproxy.org/docs/>
- OWASP Foundation. (2025d). *Threat Modeling Cheat Sheet*. [https://cheatsheetseries.owasp.org/cheatsheets/Threat\\_Modeling\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/Threat_Modeling_Cheat_Sheet.html)
- OWASP Mobile Application Security Testing Guide. (2025a). *MASTG-TEST-0045: Testing Root Detection (Android)*. <https://mas.owasp.org/MASTG-TEST-0045/>
- OWASP Mobile Application Security Testing Guide. (2025b). *MASTG-TEST-0088: Testing Jailbreak Detection (iOS)*. <https://mas.owasp.org/MASTG-TEST-0088/>
- OWASP Mobile Application Security Testing Guide. (2025c). *MASWE-0097: Root/Jailbreak Detection Not Implemented*. [https://mas.owasp.org/MASWE-0097/pandas.json\\_normalize — pandas documentation](https://mas.owasp.org/MASWE-0097/pandas.json_normalize — pandas documentation)
- pandas.json\_normalize — pandas documentation. (2025). pandas. [https://pandas.pydata.org/docs/reference/api/pandas.json\\_normalize.html](https://pandas.pydata.org/docs/reference/api/pandas.json_normalize.html)
- PouchDB Project. (2025). *Replication Guide*. <https://pouchdb.com/guides/replication.html>
- Python Pandas — Flatten nested JSON. (2025). GeeksforGeeks. <https://www.geeksforgeeks.org/python/python-pandas-flatten-nested-json/>
- Rafi, D. M., Moses, K. R. K., Petersen, K., & Mäntylä, M. V. (2012). Benefits and Limitations of Automated Software Testing: Systematic Literature Review and Practitioner Survey. *Proceedings of the 7th International Workshop on Automation of Software Test (AST 2012)*. <https://doi.org/10.1109/IWAST.2012.6228988>
- Regulation (EU) 2016/679 (General Data Protection Regulation). (2016). European Union. <https://eur-lex.europa.eu/eli/reg/2016/679/oj/eng>
- Responses API & JSON response explanation. (2025). Typeform. <https://www.typeform.com/developers/responses/JSON-response-explanation/>
- Rezaee, R., et al. (2023). Critical Criteria and Countermeasures for Mobile Health Applications. *Journal of Medical Systems / PMC*. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10020905/>
- RFC Editor. (2017). *Information on RFC 8252*. <https://www.rfc-editor.org/info/rfc8252>
- Richardson, C. (2025). *Pattern: Transactional Outbox*. <https://microservices.io/patterns/data/transactional-outbox.html>
- Rose, S., Borchert, O., Mitchell, S., & Connelly, S. (2020). *Zero Trust Architecture (SP 800-207)* (inf. téc.). National Institute of Standards y Technology. <https://doi.org/10.6028/NIST.SP.800-207>

- Shapiro, M., Preguiça, N., Baquero, C., & Zawirski, M. (2011). *Conflict-free Replicated Data Types* (inf. téc. N.º RR-7687). INRIA. <https://pages.lip6.fr/Marc.Shapiro/papers/RR-7687.pdf>
- Shostack, A. (2014). *Threat Modeling: Designing for Security*. Wiley. <https://www.wiley.com/en-us/Threat+Modeling%3A+Designing+for+Security-p-9781118809990>
- SLSA Framework. (2025a). *About SLSA*. <https://slsa.dev/spec/v1.0/about>
- SLSA Framework. (2025b). *SLSA Specification v1.1*. <https://slsa.dev/spec/v1.1/>
- Snowflake Inc. (2025a). *FLATTEN — Table function to expand semi-structured data*. <https://docs.snowflake.com/en/sql-reference/functions/flatten>
- Snowflake Inc. (2025b). *LATERAL — Using lateral joins with inline views*. <https://docs.snowflake.com/en/sql-reference/constructs/join-lateral>
- Snyk Project. (2024). *Snyk User Docs*. <https://docs.snyk.io>
- Souppaya, M., Scarfone, K., et al. (2016). *Guide to Data-Centric System Threat Modeling (SP 800-154) — Initial Public Draft* (inf. téc.). National Institute of Standards y Technology. <https://csrc.nist.gov/pubs/sp/800/154/isp>
- Superintendencia de Bancos de Guatemala. (2016). *Sector Azucarero — Departamento de Análisis Macroprudencial*. <https://www.sib.gob.gt/web/sib/boletines-economicos>
- Trendov, M. B., Varas, S., & Zeng, M. (2019). *Digital technologies in agriculture and rural areas: Status report*. Food y Agriculture Organization of the United Nations (FAO). <https://openknowledge.fao.org/handle/20.500.14283/ca4887en>
- Typeform Developers — APIs. (2025). Typeform. <https://www.typeform.com/developers/>
- UK National Cyber Security Centre. (2025). *Bring your own device (BYOD)*. <https://www.ncsc.gov.uk/collection/device-security-guidance/bring-your-own-device>
- Welcome to ODK's Docs. (2025). Get ODK Inc. <https://docs.getodk.org/>
- Wix. (2025a). *Detox — Getting Started*. <https://wix.github.io/Detox/docs/introduction/getting-started/>
- Wix. (2025b). *Detox — Gray box end-to-end testing for React Native*. <https://wix.github.io/Detox/>
- World Bank. (2017). *ICT in Agriculture: Connecting Smallholders to Knowledge, Networks, and Institutions (e-Sourcebook, Updated Edition)*. <https://documents.worldbank.org/en/publication/documents-reports/documentdetail/481581506672041837/ict-in-agriculture-connecting-smallholders-to-knowledge-networks-and-institutions>
- XLSForm. (2025). Get ODK Inc. <https://docs.getodk.org/xlsform/>
- XLSForm tutorial: Your first form. (2025). Get ODK Inc. <https://docs.getodk.org/tutorial-first-form/>

