

**DEPARTAMENTO DE ELECTROTECNIA E INFORMÁTICA
INGENIERÍA EN INFORMÁTICA**

**Desarrollo de una Aplicación Móvil para la Gestión y
Monitoreo en tiempo real de Trabajos de corta duración
con Estudiantes universitarios**

Alonso Vicente Sánchez Encina

alonso.s.encia@gmail.com

Catherine Gómez
Profesora Guía
David Larrondo
Profesor correferente

Resumen: SurModel gestiona trabajos de corta duración con estudiantes mediante flujos manuales, lo que deriva en baja trazabilidad del ciclo publicación–asignación–ejecución–cierre, validaciones débiles de presencia/cumplimiento y fricción operativa, afectando la supervisión y el pago oportuno. Como solución, se propone GoodJobs!, una aplicación móvil en Flutter apoyada por servicios serverless en Firebase, que automatiza el flujo de extremo a extremo e incorpora verificación geolocalizada mediante geocercas para check-in/check-out, registro de evidencias fotográficas en tiempo real. El objetivo es gestionar y monitorear estos trabajos, aumentando la rapidez y la confiabilidad del proceso. La validación se aborda de forma iterativa con Scrum, realizando demostraciones en Sprint Review y recogiendo retroalimentación, además de definir criterios de aceptación y una estrategia de pruebas por sprint. Como resultados, se implementó un prototipo con autenticación y verificación, exploración de ofertas en lista/mapa, postulación/confirmación, gestión administrativa y seguimiento con medición de tiempo y evidencias. Se espera reducir la carga operativa y mejorar la escalabilidad y trazabilidad del proceso.

Palabras Clave: Flutter; Firebase; geocercas; trazabilidad; trabajos de corta duración.

1 Introducción

1.1 Contexto y relevancia

SurModel es una empresa que gestiona la búsqueda y coordinación de personas principalmente jóvenes y universitarios para trabajos de corta duración (usualmente de 1 día). Sus procesos actuales incluyen: recepción de currículum vía una Landing Page de contacto, formación de grupos en WhatsApp para publicar ofertas, filtrado y asignación manual de candidatos, seguimiento del cumplimiento en terreno y pago a los colaboradores. Este esquema ha permitido operar, pero presenta limitaciones de trazabilidad, escalabilidad, automatización y control de calidad de evidencias cuando aumenta el volumen o la dispersión geográfica de las tareas.

Para SurModel tras un periodo prolongado de operación con procesos manuales este proyecto constituye el primer esfuerzo formal de transición tecnológica. La meta es migrar gradualmente hacia flujos trazables sin elevar demasiado el costo de adopción, aprovechando toda la experiencia acumulada con los años



CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título ☐ Tesis de Postgrado

Título del trabajo: Desarrollo de una Aplicación Móvil para la Gestión y Monitoreo en tiempo real de Trabajos de corta duración con Estudiantes universitarios

Nombre del candidato(a): Alonso Vicente Sánchez Encina

Carrera / Grado: Ingeniería Informática

Campus: Sede Viña del Mar **Departamento:** Electrotecnia e Informática

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Catherine Gómez Barrera, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)

☒ El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (**embargo**) por (**marcar una opción**):

☐ 6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 22-07-2026

Firma: Catherine BC

Estudiante o Candidato(a):

Fecha: 24-07-2026

Firma: Alonso :3

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

1.2 Definición del problema.

La coordinación manual de trabajos de corta duración genera:

- Baja trazabilidad del ciclo publicación–asignación–ejecución–cierre.
- Validaciones débiles de presencia y cumplimiento en terreno.
- Lentitud y fricción en postulación, asignación y comunicación operativa.
- Sobrecarga operativa para supervisar múltiples tareas en paralelo.
- Riesgos en pago oportuno y calidad entregable por falta de evidencias verificables.

Se requiere una solución que automatice y estandarice estos flujos, incorpore verificación geolocalizada y notificaciones en tiempo real.

1.3 Breve descripción sobre la propuesta de solución

La propuesta es una solución móvil integral que automatiza de extremo a extremo el ciclo de trabajos de corta duración, sustituyendo flujos manuales por procesos orquestados y trazables.

- Aplicación móvil (Flutter) [1, 2]: permite a los colaboradores descubrir, postularse y ejecutar tareas. Durante la ejecución, la aplicación habilita registro de inicio (check-in) y término (check-out) dentro de geocercas [3, 4, 5] y captura evidencias fotográficas con coordenadas y marca de tiempo.
- Backend sin servidores (Firebase) [6]: provee Autenticación, Cloud Firestore (base de datos), Cloud Storage (almacenamiento de evidencias), Cloud Functions y Firebase Cloud Messaging (notificaciones de asignación, inicio, recordatorios y cierre).
- Supervisión y observabilidad: un panel mínimo de seguimiento y métricas para medir tiempos de ciclo, validez de evidencias y disponibilidad, sin necesidad de desplegar servidores físicos.

Desde la perspectiva del proceso, el flujo objetivo abarca: publicación de la tarea, postulación con filtros de ubicación y disponibilidad, asignación con confirmación, ejecución en terreno y validación/cierre, combinando reglas automáticas y revisión selectiva; tras el cierre, el sistema deja preparado el resumen operativo que respalda el pago.

Para poder desarrollar la solución, el equipo se dividió en múltiples roles complementarios. El estudiante asumió los roles de Scrum Máster y Desarrollador Full Stack, participando más activamente en el desarrollo de funcionalidades clave y coordinando la priorización del conjunto de tareas del proyecto (backlog)

El alcance del trabajo comprende una aplicación móvil para colaboradores; servicios backend; reglas de seguridad; flujo de publicación–postulación–asignación–ejecución–validación–cierre; panel de consulta. Quedan fuera de alcance la pasarela de pago productiva, la analítica/Bi avanzada y un portal web empresarial completo.

Estado actual del producto (alcance implementado):

- Inicio seguro y verificación de rol: enrutamiento inicial por estado de sesión y rol (usuario/administrador) tras inicializar Firebase.

- Onboarding y bienvenida: secuencia de diapositivas con progreso y persistencia de "ya visto" en preferencias.
- Autenticación: inicio de sesión, registro, recuperación de contraseña y verificación de cuenta.
- Navegación inferior para usuarios: cuatro secciones (Trabajos, Mis trabajos, Mis pagos, Configuración) con conservación de pestaña al navegar. Cabe señalar que la sección "Mis pagos" funciona como un registro administrativo del estado de la remuneración. La aplicación no procesa la transacción bancaria (pasarela de pago productiva), sino que permite al administrador marcar el pago como "realizado" y adjuntar el comprobante de transferencia externo, permitiendo al usuario visualizar que su remuneración ha sido gestionada.
- Detalle de trabajo y postulaciones: cuenta regresiva al cierre; postular/cancelar/confirmar asignación.
- Gestión administrativa: creación/edición de trabajos con formularios por pasos, selección de ubicación, carga de imágenes y plantillas guardadas.
- Panel de administrador y actividad: listas en vivo (trabajos, pagos, estado).
- Seguimiento de trabajo: medición de tiempo trabajado durante la ejecución y almacenamiento de evidencias fotográficas asociadas al trabajo.

1.4 Objetivos Generales.

El objetivo principal es diseñar, implementar y validar una solución móvil con servicios en la nube que gestione y monitoree en tiempo real trabajos de corta duración, incorporando trazabilidad, verificación geolocalizada y notificaciones, para transformar un proceso manual no medible en un flujo digital trazable, eliminando la latencia de comunicación asíncrona (WhatsApp) y asegurando la integridad de los datos de asistencia.

Como objetivos específicos:

1. Definir la arquitectura técnica y el modelo de datos en Firebase.
2. Implementar componentes críticos: geolocalización, validación de evidencias y reglas de seguridad en Cloud Firestore y Cloud Functions.
3. Mejorar la experiencia de uso para colaboradores y administradores.
4. Reducir de forma consistente el tiempo entre publicación y asignación de trabajos.

1.5 Metodología

Se utilizará Scrum, un marco de trabajo basado en la transparencia, la inspección y la adaptación. El equipo se organiza en Product Owner, Scrum Máster y Desarrolladores. Para el desarrollo del producto, se planteó un programa de tres Sprints, alineado con la Feria de Software.

Se eligió por su agilidad y adaptabilidad: permite entregas frecuentes de incrementos funcionales, promueve la mejora continua mediante ciclos cortos de inspección y adaptación, y se ajusta a un contexto de primera transición tecnológica para SurModel. El enfoque iterativo reduce riesgo y facilita incorporar retroalimentación temprana junto al patrocinador

1.5.1 Artefactos Scrum

Product Backlog: Listado completo de historias de usuario con criterios de aceptación. Se actualiza continuamente con retroalimentación del comité de profesores a cargo y de SurModel.

Sprint Backlog: Conjunto de historias seleccionadas para el sprint con objetivo del sprint, tareas técnicas descompuestas, estimaciones y responsables.

Incremento: resultado potencialmente entregable al final de cada sprint. Para este proyecto, implica versión funcional de aplicación móvil y servicios en la nube.

1.5.2 Eventos Scrum

Sprint Planning: Se revisa la visión del producto y el Product Backlog; se seleccionan las historias que aportan más valor; se define el Objetivo del Sprint; se descomponen historias en tareas técnicas; se acuerdan criterios de aceptación, riesgos y estrategia de pruebas.

Al finalizar el evento se espera tener: El objetivo del Sprint claro, Sprint Backlog con historias y tareas estimadas con criterios de aceptación explícitos.

Reunión diaria (Daily Scrum): Aquí es donde se sincroniza el equipo; revisión del tablero de trabajo; identificación de bloqueos y dependencias; ajuste del plan de los próximos días y, si es necesario, redistribución de tareas.

Sprint Review: el equipo presenta el avance (Incremento). Se realiza una demostración funcional de lo construido, se explican decisiones y se recoge retroalimentación como comentarios, preguntas y sugerencias.

Retrospectiva del Sprint: Se identifican fortalezas, oportunidades de mejora y experimentos para el siguiente sprint, con acciones concretas.

1.5.3 Cronograma de Sprints

Sprints	Inicio	Final	Duracion
Sprint 1: Fundación del producto	10 de Agosto 2025	31 de Agosto 2025	21 días
Sprint 2: Ejecución y verificación	3 de Septiembre 2025	26 de Septiembre	23 días
Sprint 3: Supervisión y cierre	3 de Octubre	24 de Octubre	21 días

Figura 1. Tabla resumen de cronograma Sprints.

Fuente: elaboración propia

1.5.4 Herramienta colaborativa

Se utiliza JIRA como herramienta colaborativa para:

- Gestionar el Product Backlog y Sprint Backlog (épicas, historias, tareas y errores), con estados típicos: To Do, In Progress, In Review y Done.
- Tableros Scrum para seguimiento visual del flujo de trabajo.
- Trazabilidad: vínculo de historias con épicas, tareas técnicas, adjuntos y criterios de aceptación.
- Comentarios y notificaciones para coordinar al equipo.

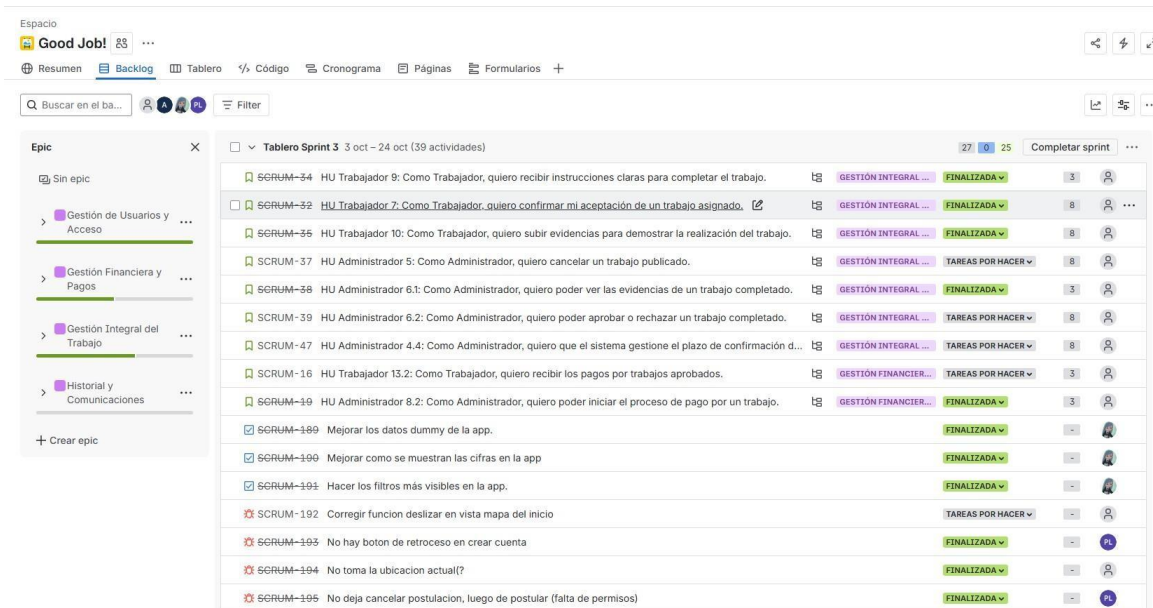


Figura 2. Captura de pantalla herramienta colaborativa: Backlog
Fuente: atlassian.net

2 Marco Teórico

2.1 Flutter como marco de desarrollo móvil multiplataforma:

Flutter ofrece una interfaz de usuario nativa multiplataforma sustentada en un motor de renderizado propio y un árbol de componentes “widgets” declarativo [1, 2]. Esto permite consistencia visual, alto rendimiento y menor costo de mantenimiento al compartir código entre iOS y Android. En este trabajo se aprovecha su enfoque declarativo para construir flujos guiados de postulación, ejecución haciendo check-in y check-out y carga de evidencias, con atención al manejo de permisos (cámara, ubicación, almacenamiento).

2.2 Servicios de backend administrados y arquitectura Serverless:

Los servicios de backend administrados como Firebase proveen Autenticación, Cloud Firestore (base de datos), Cloud Storage (almacenamiento), Cloud Functions (lógica) y Firebase Cloud Messaging (mensajería) [6]. En este proyecto se utilizan estas capacidades para orquestar reglas de seguridad, validaciones automáticas y notificaciones de los hitos operativos.

2.3 Geolocalización, geocercas y evidencias fotográficas controlados por la aplicación:

La geolocalización con geocercas permite registrar de forma controlada el inicio y término de una tarea (check-in/check-out). La evidencia fotográfica se basa en captura y guardado de los archivos JPG. La implementación se apoya en APIs de geofencing de Android y Apple. [3, 4, 5]

2.4 Soluciones existentes y diferenciación

En el mercado global existen plataformas de como Wonolo o Uber Works que gestionan trabajos bajo demanda. Sin embargo, estas soluciones presentan barreras significativas para el contexto local y de nicho de SurModel: están diseñadas para mercados masivos principalmente EE. UU, no integran validación de documentos locales (RUT chileno, antecedentes), y sus modelos de negocio suelen ser rígidos y costosos para PyMEs locales.

Adicionalmente, se debe considerar como competencia directa el flujo manual actual utilizado por empresas como SurModel, basado en herramientas no integradas como WhatsApp (comunicación) y hojas de cálculo (gestión). Si bien estas herramientas tienen una barrera de entrada nula, carecen de validación transaccional: WhatsApp no permite georreferenciar la asistencia ni estructurar la entrega de evidencias, mientras que las planillas manuales sufren de baja integridad de datos y no permiten actualizaciones en tiempo real. La propuesta actual busca cubrir esta brecha tecnológica, ofreciendo la inmediatez de la mensajería pero con la estructura de un sistema ERP simplificado.

Este producto se diferencia al enfocarse específicamente en el segmento de estudiantes universitarios y trabajos de eventos/promociones en Chile, incorporando lógica de negocio adaptada a la realidad local (validación de RUT, gestión de transferencias manuales comunes en el medio) y ofreciendo un control de asistencia por geocercas más estricto que las herramientas genéricas de mensajería (WhatsApp), pero sin la complejidad y costo de un ERP corporativo.

2.5 Seguridad y privacidad:

La solución aplica autenticación y autorización por roles, reglas a nivel de documento/colección en Cloud Firestore, cifrado en tránsito y minimización de datos.

Considerando la Ley N° 19.628 sobre la Protección de la Vida Privada, la aplicación adopta el principio de minimización de datos. Información considerada sensible (como datos bancarios o RUT) se almacena con reglas de seguridad restrictivas en Firestore, accesibles únicamente por el administrador para fines de pago y por el propio titular.

Las reglas restringen lectura/escritura por rol y propiedad del documento; las Cloud Functions ejecutan validaciones y auditoría con logs para trazabilidad. [6]

2.6 Fundamentos conceptuales de Scrum:

Scrum es un marco basado en transparencia, inspección y adaptación con roles, artefactos y eventos definidos. Sustenta ciclos cortos con Incrementos potencialmente entregables y promueve la mejora continua. En el proyecto guía la priorización por valor, la validación temprana y la adaptación de alcance en Sprints.

3 Propuesta de solución

3.1 Visión general del sistema

La solución consiste en una aplicación móvil para colaboradores y un conjunto de servicios en la nube para administración, validación y seguimiento.

3.2 Arquitectura Técnica y Modelo de Datos

La solución utiliza una arquitectura Serverless basada en Firebase. La aplicación móvil se comunica directamente con los servicios gestionados sin un servidor intermedio, utilizando el patrón MVVM (Model-View-ViewModel) dentro de Flutter para separar la lógica de interfaz de la lógica de negocio.

Modelo de Datos en Cloud Firestore (NoSQL): La persistencia se maneja en Cloud Firestore utilizando una estructura documental optimizada para lectura rápida. La entidad central es el "Trabajo", cuyo esquema (implementado en la clase Trabajo) define la integridad de la oferta y su ejecución. Las colecciones principales son:

- Users: Almacena perfiles, roles (admin/trabajador), datos de contacto y estado de verificación.
- Jobs (Trabajos): Contiene la definición operativa de la oferta. Cada documento incluye:
 - Identificación: título, descripción, empresa, imagenPrincipal.
 - Logística y Control: ubicación (coordenadas para geocerca), fechaInicioTrabajo, fechaFinTrabajo.
 - Estado: Campo estado controlado por un enumerador estricto (activo, porConfirmar, enCurso, porRevisar, finalizado, cancelado) que gobierna las transiciones del flujo.
 - Asignación: trabajadorAsignadoId (referencia al usuario seleccionado).
 - Finanzas: precio, pagado
 - Applications: Subcolección o colección vinculada que registra la relación postulante-trabajo.
- Evidencias: Subcolección dentro de trabajos que almacena referencias a fotos en Cloud Storage.

Reglas de Seguridad (Firestore Security Rules): La seguridad se implementó mediante reglas declarativas en el servidor que validan la identidad y autorización antes de cualquier operación de lectura o escritura, sin depender de la validación del cliente.

- Funciones de Ayuda: Se definieron funciones reutilizables como isAdmin(), isTrabajadorAsignado y esPostulanteDelTrabajo.
- Políticas de Acceso:
 1. Colección trabajos: Lectura pública, pero escritura exclusiva para administradores. La actualización está restringida: los administradores tienen control total, mientras que el trabajador asignado solo puede modificar campos específicos de ejecución como inicioTrabajoReal o evidenciasEnviadas, preveniendo la alteración de montos o fechas contractuales.
 2. Colección usuarios: Un usuario solo puede leer y editar su propio perfil, protegiendo datos sensibles como RUT o cuenta bancaria frente a otros usuarios.

3. Colección evidencias: Solo el trabajador asignado o el administrador pueden subir archivos; la eliminación de evidencias está restringida para evitar la pérdida de trazabilidad.

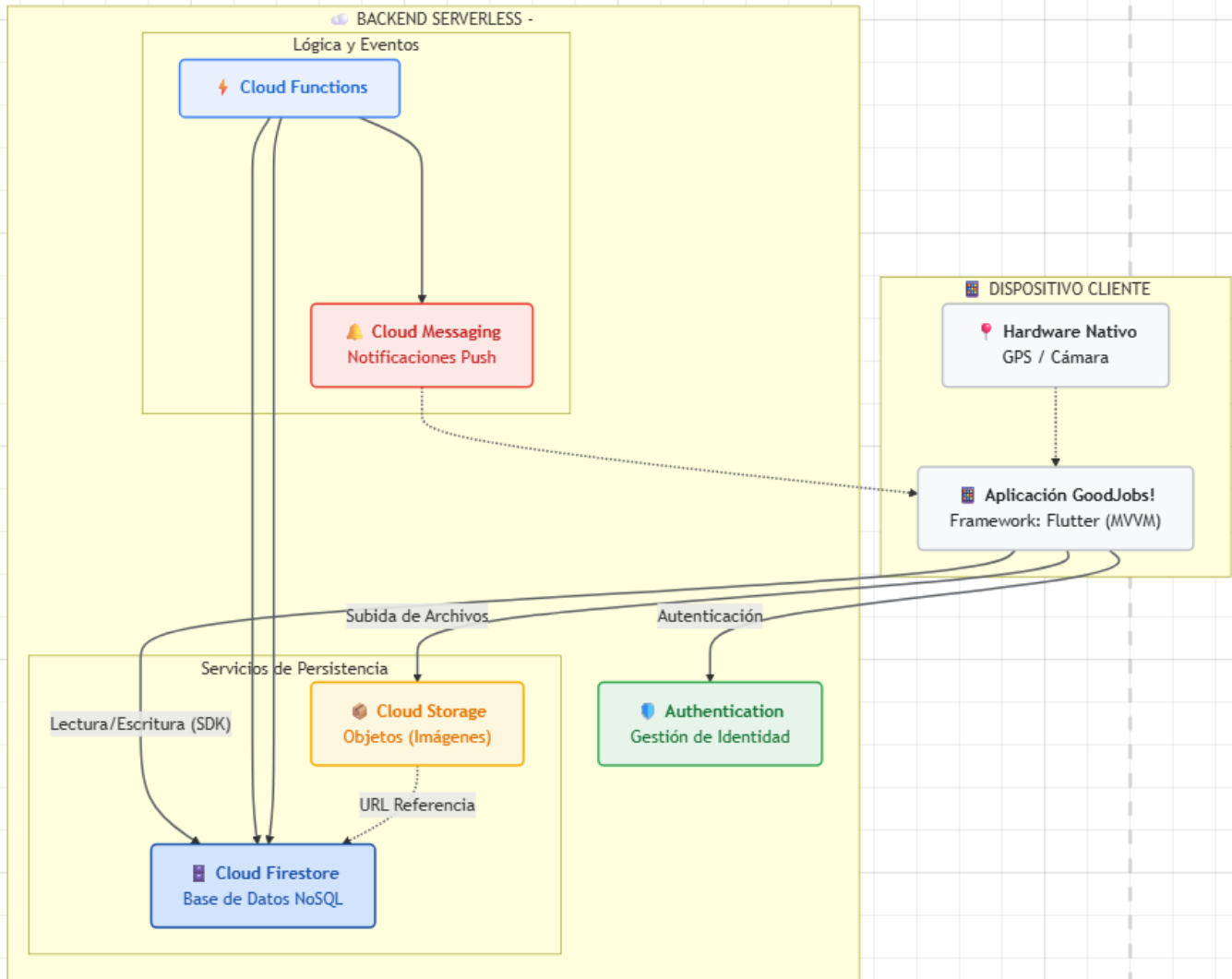


Figura 3. Diagrama de Arquitectura
Fuente: Elaboración propia

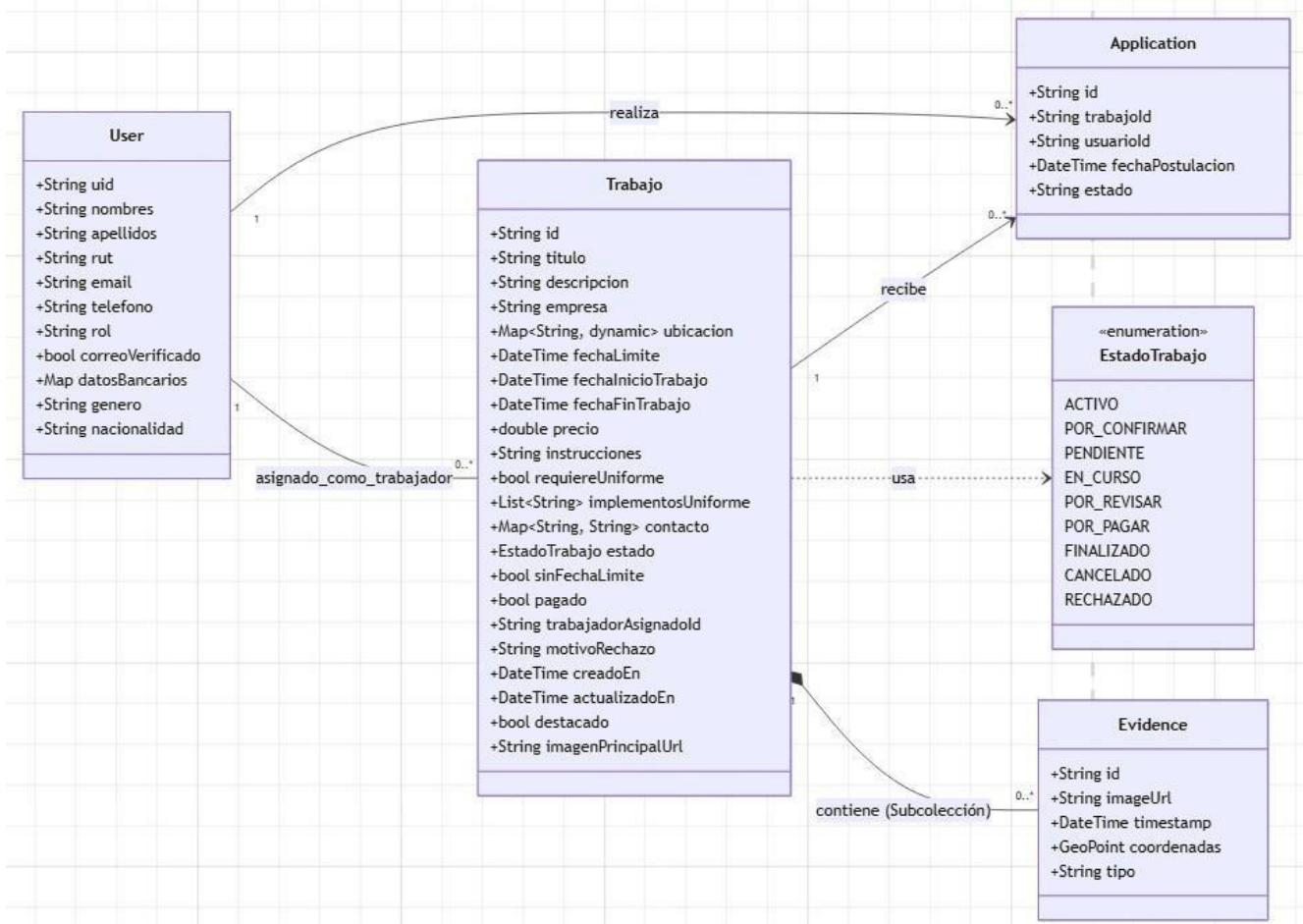


Figura 4. Diagrama de modelo de datos.

Fuente: Elaboración propia

3.3 Justificación de stack tecnológico

Esta sección justifica cada decisión técnica principal del proyecto. Los criterios dominantes del contexto son: un equipo de desarrollo reducido, un plazo acotado de tres Sprints, presupuesto operativo inicial cercano a cero para el patrocinador y la necesidad de sincronización en tiempo real entre los roles trabajador y administrador.

3.3.1 Flutter frente a las alternativas de desarrollo móvil

Se evaluaron tres opciones: desarrollo nativo dual; Kotlin para Android y Swift para iOS, React Native y Flutter. El desarrollo nativo dual implica mantener dos bases de código independientes, lo que en la práctica duplica el esfuerzo de implementación y de pruebas de cada funcionalidad; con un solo desarrollador y tres Sprints, esta opción quedaba fuera del presupuesto de tiempo. Entre las opciones multiplataforma, Flutter se seleccionó porque su motor de renderizado propio garantiza un comportamiento visual idéntico en ambas plataformas y evita la capa de puente hacia componentes nativos que React Native requiere, la cual introduce puntos adicionales de fallo y de degradación de

rendimiento en vistas con actualización frecuente, como el mapa con posición en vivo y los contadores regresivos de la aplicación. El resultado cuantificable de la decisión es una sola base de código para dos plataformas objetivo, es decir, una reducción aproximada del 50% del esfuerzo de construcción y mantención frente al desarrollo nativo dual.

3.3.2 Backend serverless (Firebase) frente a un backend autoadministrado

La alternativa evaluada fue una API REST propia (por ejemplo Node.js/Express con una base de datos relacional) desplegada en un servidor virtual. Esta opción exige diseñar e implementar la API, el sistema de autenticación, la infraestructura de notificaciones push, el aprovisionamiento del servidor y su monitoreo; trabajo que no aporta valor directo al proceso de negocio y que habría consumido una fracción relevante de los tres Sprints disponibles. Firebase entrega esas mismas capacidades como servicios administrados (Authentication, Cloud Firestore, Cloud Storage, Cloud Functions y Cloud Messaging) [6], con dos propiedades decisivas para este contexto: En primer lugar la sincronización en tiempo real es nativa del SDK mediante listeners de Firestore, sin necesidad de construir infraestructura de websockets, y es la base técnica de los paneles en vivo del administrador y del seguimiento de estados del trabajador; y el modelo de costos es de pago por uso con una cuota gratuita diaria [6], frente al costo fijo mensual de un servidor autoadministrado que se paga aun sin tráfico. El análisis de costos del capítulo de Impacto proyecta que, a la escala objetivo de SurModel, el costo operativo de la alternativa elegida es cercano a cero.

3.3.3 Modelo de datos NoSQL documental frente a un modelo relacional

El patrón de acceso del sistema está dominado por lecturas de listados (ofertas, postulaciones, pagos) y de documentos individuales, con escrituras poco frecuentes y acotadas al ciclo de vida de cada trabajo. Ese perfil favorece un modelo documental desnormalizado: la postulación se duplica en la subcolección del trabajo y en la del usuario, de modo que cada vista lee directamente los documentos que necesita sin operaciones de unión, con costo de lectura proporcional solo a los elementos mostrados. Un modelo relacional habría simplificado consultas analíticas complejas, pero éstas no forman parte del alcance, mientras que la lectura rápida de listados sincronizados en tiempo real sí es un requisito central.

3.3.4 Cartografía con flutter_map y OpenStreetMap frente al SDK de Google Maps

Para los mapas de exploración, selección de ubicación y check-in se utilizó la librería flutter_map con fragmentos de OpenStreetMap, en lugar del SDK de Google Maps. La justificación es principalmente cuantitativa: la opción elegida no requiere clave de API ni habilitar una cuenta de facturación, mientras que el SDK de Google Maps queda sujeto a tarifas por carga de mapa una vez superado el crédito mensual del proveedor y obliga a habilitar facturación desde el primer día. Dado que la funcionalidad requerida se limita a mostrar marcadores y el círculo de la geocerca, ambas opciones son funcionalmente equivalentes para el alcance del proyecto.

3.4 Roles principales

Colaborador (estudiante/trabajador): descubre trabajos, se postula, confirma asignaciones, ejecuta tareas con check-in/check-out dentro de geocercas y sube evidencias fotográficas.

Administrador: publica y edita trabajos, revisa postulaciones, asigna/ cancela, monitorea ejecución, valida cierres y gestiona el proceso de pago.

3.5 Flujo de extremo a extremo

- 1) Publicación del trabajo (administrador): crea el aviso con título, descripción, requisitos, ubicación, ventana temporal y cupos; puede usar plantillas y subir imágenes.
- 2) Descubrimiento (colaborador): explora en lista o mapa (centrado en su ubicación) y ordena por próximas/recientes.
- 3) Postulación: desde el detalle, se postula; puede cancelar su postulación antes del cierre.
- 4) Asignación y confirmación: el administrador asigna; el colaborador recibe notificación y confirma su aceptación.
- 5) Ejecución en terreno: al llegar a la geocerca, realiza check-in; la aplicación inicia un temporizador de trabajo. Durante la ejecución, sube evidencias fotográficas asociadas al trabajo.
- 6) Cierre: realiza check-out y envía el resumen del trabajo.
- 7) Validación y pago (administrador): revisa reglas automáticas y evidencias; aprueba/rechaza y marca para pago.

3.6 Recorrido por la aplicación Vista Usuario/Trabajador

Este apartado describe el recorrido principal dentro de la aplicación desde la perspectiva del trabajador, abarcando la creación de cuenta, el cumplimiento de requisitos mínimos para operar, la exploración de ofertas, la postulación, el seguimiento del estado y la ejecución en terreno hasta el cierre con pago. El objetivo es documentar el flujo como un proceso operativo digitalizado, explicando cómo la interfaz guía decisiones y cómo las validaciones de sistema previenen errores comunes, entregan transparencia de estado y fortalecen trazabilidad, entendida como la posibilidad de reconstruir qué ocurrió, cuándo ocurrió y en qué condiciones, durante el ciclo completo.

3.6.1 Registro Screen

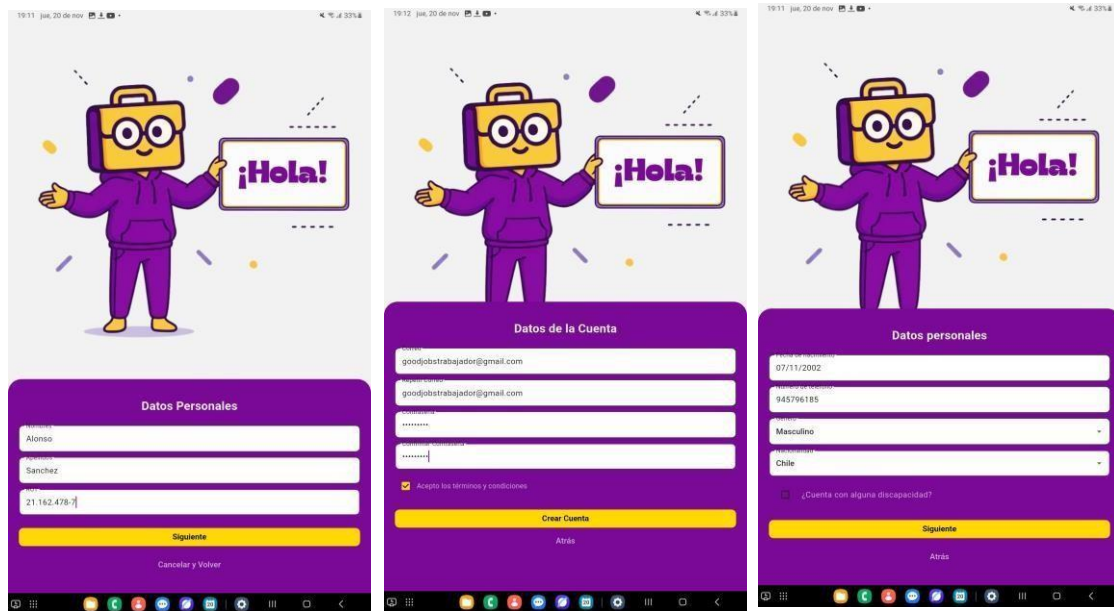


Figura 5-6-7. Captura de pantalla aplicación GoodJobs!

Fuente: Elaboración propia.

El proceso de creación de usuario se realiza en pasos para facilitar el llenado y mejorar la calidad de datos. Se recogen solo los mínimos necesarios para operar y proteger a las personas involucradas.

Paso 1. Datos personales (básicos). Nombres y Apellidos identifican al postulante en comunicaciones y comprobantes. RUT actúa como identificador único, ayuda a evitar cuentas duplicadas y habilita procesos posteriores. Se validan formato y dígito verificador; ante error, se informa y se permite corregir.

Paso 2. Datos personales (perfil y contacto). Fecha de nacimiento para verificar mayoría de edad y cumplir requisitos legales del contratante. Teléfono para contacto operativo y coordinación ante cambios o imprevistos. Género y Nacionalidad se solicitan con fines estadísticos y para requisitos logísticos que algunas campañas establecen (por ejemplo, tallas de vestuario o documentación específica); su tratamiento respeta principios de no discriminación y privacidad. La casilla "¿Cuenta con alguna discapacidad?" permite informar necesidades de apoyo o adecuaciones, de modo que el administrador asigne tareas compatibles y seguras.

Paso 3. Datos de la cuenta. Correo (debe ser real porque se utiliza para verificar la cuenta y recuperar acceso) y Repetir correo (para evitar errores). Contraseña y Confirmar contraseña. Al aceptar términos y condiciones y pulsar Crear Cuenta, se crea el perfil y se envía el correo de verificación.

3.6.2 Inicio de sesión



Figura 8. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

El inicio de sesión opera como punto de control para autenticar credenciales, recuperar acceso y encauzar al usuario hacia una experiencia consistente según su rol. Se privilegia un formulario compacto con campos mínimos y ayudas como mostrar u ocultar contraseña, lo que reduce errores y mejora usabilidad en contexto móvil donde la digitación es un factor de fricción frecuente, además se incorpora recuperación de contraseña para disminuir bloqueos y dependencia de soporte manual. Tras autenticar, la aplicación consulta el perfil y aplica un enrutamiento por rol que separa trabajador y administrador, este comportamiento es relevante porque transforma el control de acceso en un componente visible de la experiencia y ayuda a prevenir errores de operación, por ejemplo, evitando que un trabajador intente acceder a funciones administrativas o que un administrador utilice vistas incompletas para su propósito. A nivel de diseño de sistema, este enrutamiento refuerza el principio de mínimo privilegio, reduce superficie de error en navegación, ya que cada rol puede evolucionar con componentes y permisos propios sin comprometer consistencia del flujo.

3.6.3 Solicitud de permiso al usuario

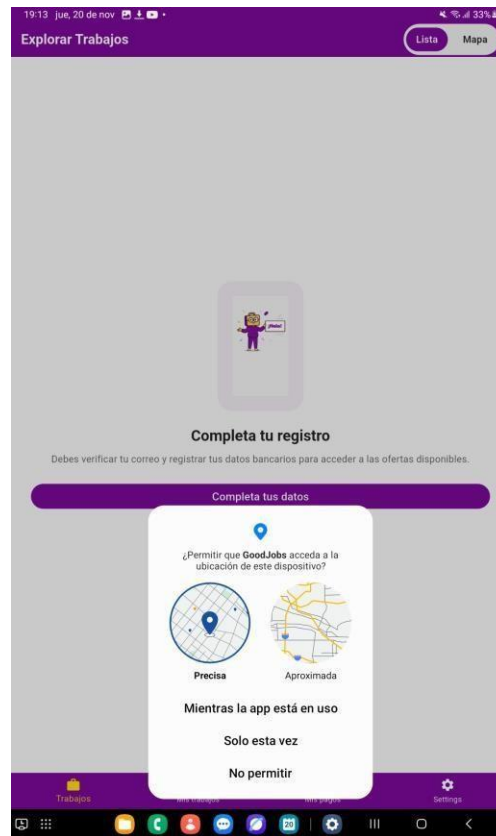


Figura 9. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

En el primer acceso se solicita permiso de ubicación debido a que habilita funcionalidades centrales que influyen directamente en la propuesta de valor y en la trazabilidad operativa. Por una parte, la ubicación permite estimar cercanía y ordenar oportunidades con un criterio espacial comprensible, contribuyendo a que el usuario seleccione ofertas viables en tiempo y distancia, lo que reduce postulaciones impulsivas que luego terminan en cancelación. Por otra parte, el permiso habilita mecanismos de control en terreno cuando existe validación por geocerca, es decir cuando ciertas acciones se habilitan solo si el usuario se encuentra dentro de un perímetro asociado al lugar de ejecución, lo que fortalece evidencia de presencia y disminuye validaciones manuales. Para alinear esta solicitud con principios de privacidad, se promueve el permiso durante el uso y se contextualiza la necesidad del permiso en el momento correcto, evitando solicitarlo de forma anticipada sin propósito claro, lo que tiende a aumentar rechazo y afecta la continuidad del flujo. La aplicación debe contemplar escenarios de permiso denegado, ofreciendo degradación funcional, por ejemplo, permitir explorar sin mapa centrado o solicitar permiso nuevamente cuando el usuario intente una acción que lo requiera, manteniendo claridad sobre qué se pierde y cómo recuperarlo.

3.6.4 Inicio por primera vez sin verificar

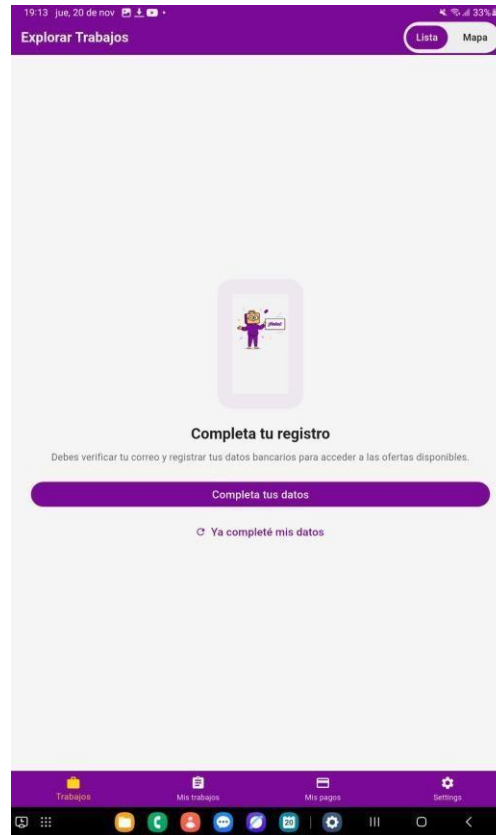


Figura 10. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

Cuando el usuario inicia por primera vez, la aplicación presenta un estado de cuenta incompleta que actúa como control preventivo para sostener la integridad del proceso. El propósito de esta barrera no es restringir arbitrariamente, sino evitar que el usuario avance a etapas donde el sistema no podrá completar el ciclo, por ejemplo, registrar una postulación que luego no podrá confirmarse o cerrar con pago por falta de información bancaria o por ausencia de verificación, lo que generaría incertidumbre, retrabajo administrativo y una percepción de falla del sistema. En términos de experiencia, este estado se comunica como un paso guiado, entregando una acción directa para resolver pendientes y una explicación breve de por qué es necesario, con ello se reduce frustración y se evita que el usuario interprete el bloqueo como un error. Esta etapa también aporta a trazabilidad porque fija un umbral de condiciones mínimas, de modo que la postulación representa una intención válida de participación y no un registro incompleto que después deba descartarse.

3.6.5 Verificación de cuenta

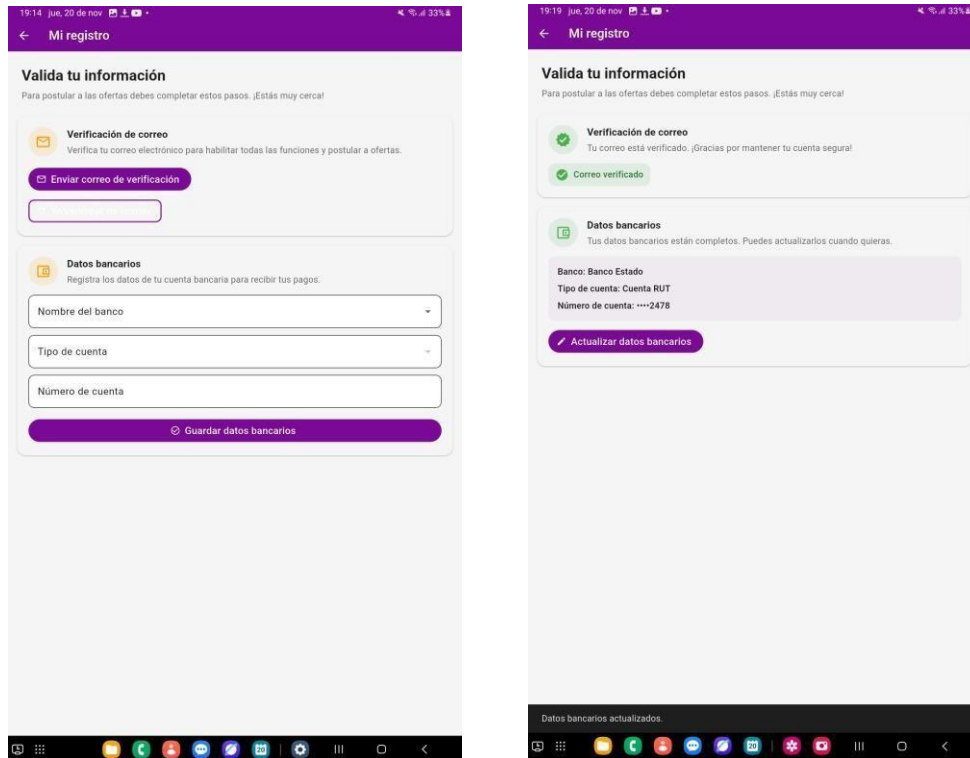


Figura 11-12. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

El flujo de habilitación concentra requisitos operativos en un punto único con el fin de preparar el proceso completo antes de permitir postulaciones. Primero se gestiona la verificación de correo como mecanismo de confirmación de identidad de contacto y de canal de comunicación confiable, lo que es clave para notificar decisiones administrativas, cambios logísticos o incidencias, se contempla reenvío de verificación para resolver casos comunes como correos no recibidos o enlaces expirados, y se incorpora una acción de confirmación dentro de la aplicación que sincroniza el estado real con el servidor, evitando inconsistencias donde el usuario cree haber verificado pero Luego se solicita información bancaria que soporta el cierre y pago, estructurando campos como banco, tipo de cuenta y número de cuenta, esta captura anticipada reduce el riesgo de retrasos en el pago y disminuye dependencia de coordinación por canales externos. Desde la perspectiva de calidad de datos, este punto debe validar formatos, evitar caracteres inválidos y permitir actualización posterior desde configuración, de modo que el dato sea mantenible y no se convierta en una fuente de errores en el tiempo. Desde una perspectiva de seguridad, esta información se trata como dato sensible por lo que el diseño de interfaz busca minimizar exposición, solicitar solo lo necesario y mostrarlo de forma responsable, además la lógica de sistema debe considerar control de acceso y registro de cambios para poder auditar modificaciones ante discrepancias.

3.6.6 Trabajos Screen

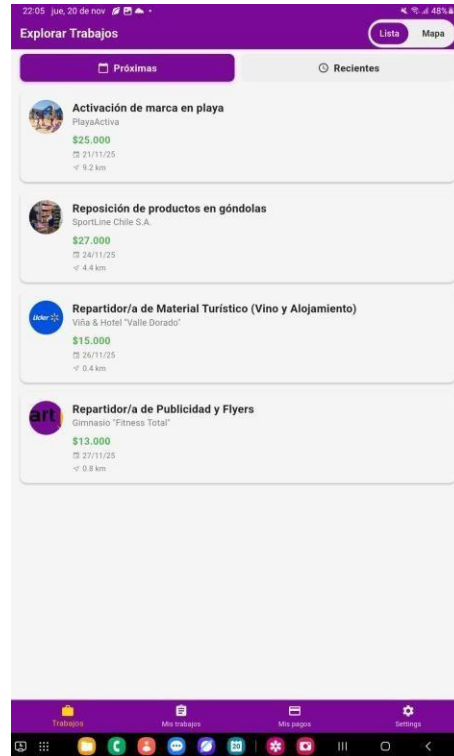


Figura 13. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

Una vez que el usuario verifica su correo y registra sus datos bancarios, accede por primera vez a Explorar Trabajos. Esta pantalla lista todas las oportunidades disponibles y actúa como punto de partida del flujo de postulación.

En la parte superior se ofrecen dos criterios de orden: Próximas, que prioriza las ofertas cuya fecha de ejecución está más cercana a la fecha actual y recientes, que prioriza las últimas ofertas publicadas.

En la esquina superior derecha se ubica el conmutador Lista / Mapa. En esta captura se muestra Lista; el usuario puede cambiar a Mapa para visualizar las ofertas georreferenciadas.

Cada oferta presenta (en orden):

- 1) Imagen de portada o logo representativo del trabajo.
- 2) Título de la oferta y empresa que publica.
- 3) Remuneración para el trabajo.
- 4) Fecha programada de ejecución.
- 5) Distancia estimada respecto de la ubicación actual del usuario.

3.6.7 Detalles de trabajo

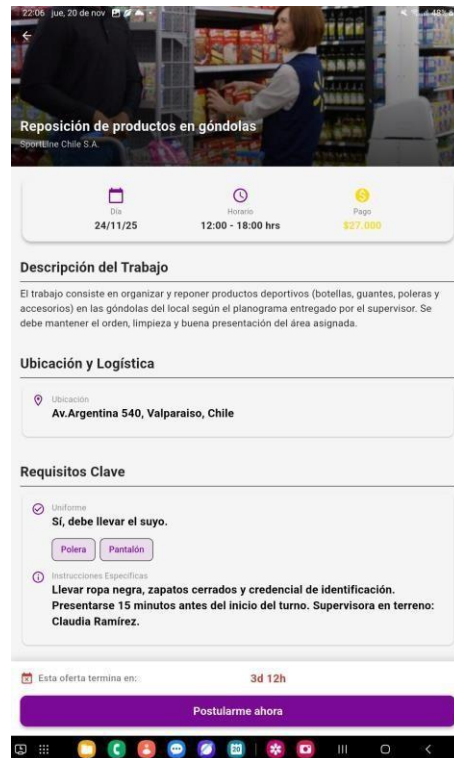


Figura 14. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

Al seleccionar una oferta se presenta el detalle completo con una jerarquía diseñada para permitir una decisión informada antes de postular. Se prioriza identificación del trabajo y empresa, luego se resumen variables operativas como día, horario y pago, y después se entrega la descripción de tareas, ubicación, logística y requisitos, de modo que el usuario pueda confirmar viabilidad real, por ejemplo, verificar si el lugar es alcanzable y si cumple con expectativas de la tarea. Esta estructura actúa como una forma de contrato operativo previo, ya que establece condiciones y reduce interpretaciones erróneas que suelen producir cancelaciones o incumplimientos. El detalle mantiene un indicador de tiempo restante para postular, lo que aporta claridad sobre la ventana de disponibilidad, reduce incertidumbre y evita que el usuario asuma que la oferta estará siempre vigente. La acción principal de postulación se mantiene visible y consistente para reducir ambigüedad, además se espera que el sistema considere mensajes claros cuando el usuario no puede postular, por ejemplo, por cupos agotados o por requisitos incompletos, de modo que el error sea entendible y no parezca un fallo del sistema.

3.6.8 Botón de postulación

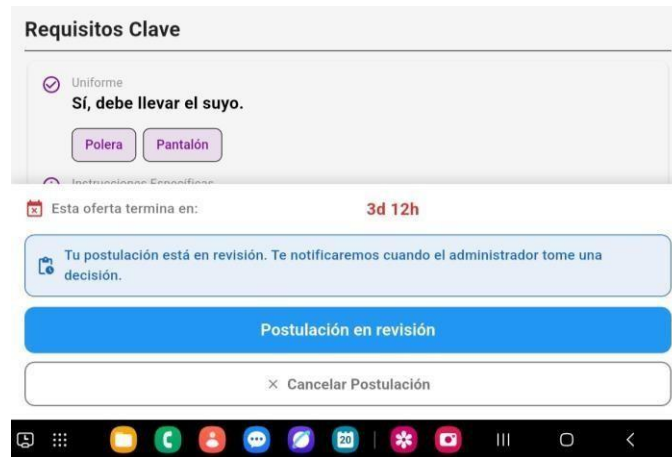


Figura 15. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

Al ejecutar la acción de postular, la aplicación realiza validaciones previas, verificando condiciones como cuenta verificada, datos bancarios presentes y disponibilidad de cupos, con ello se evita registrar solicitudes que no podrán procesarse. Una vez registrada la postulación, la interfaz transita a un estado de revisión que comunica de inmediato que la acción tuvo efecto, este feedback es esencial para confianza, ya que evita que el usuario intente repetir la acción por falta de confirmación. El control de duplicidad se implementa deshabilitando el botón principal y mostrando un texto de estado, además se ofrece la posibilidad de cancelar mientras el proceso no se haya resuelto, lo que entrega control y reduce soporte manual. Mantener visible la referencia temporal de cierre aporta coherencia entre reglas del sistema y percepción del usuario, ayudando a explicar por qué una oferta puede expirar incluso si la postulación existe, y reforzando que la revisión se encuentra sujeta a una ventana de tiempo definida. En términos de trazabilidad, esta etapa es crítica porque marca el punto donde el usuario pasa de explorar a comprometerse con una solicitud formal, por lo que el sistema debe registrar fecha y estado de la postulación y debe poder reconstruir cambios, por ejemplo, cuándo se canceló o cuándo cambió a aceptada o rechazada desde la perspectiva administrativa.

3.6.9 Seguimiento de postulación



Figura 16. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

El seguimiento consolida el estado de la postulación en una sección dedicada que permite al usuario comprender su situación sin depender de canales externos. Esta vista reduce incertidumbre y disminuye consultas manuales porque el usuario puede ver en qué etapa se encuentra y qué ocurrirá después, además debe mantener consistencia con lo observado en el detalle del trabajo y con el estado de postulación en revisión. Desde un enfoque de diseño de proceso, el seguimiento debe expresar estados de manera inequívoca y asociar acciones coherentes cuando corresponda, por ejemplo, permitir cancelar si aún es posible o mostrar mensajes informativos cuando una decisión ya fue tomada. La relevancia de este componente representa la materialización de la trazabilidad percibida, es decir la capacidad del usuario de seguir el progreso del proceso, lo que impacta directamente en la confianza hacia la plataforma.

3.6.10 Trabajo en curso

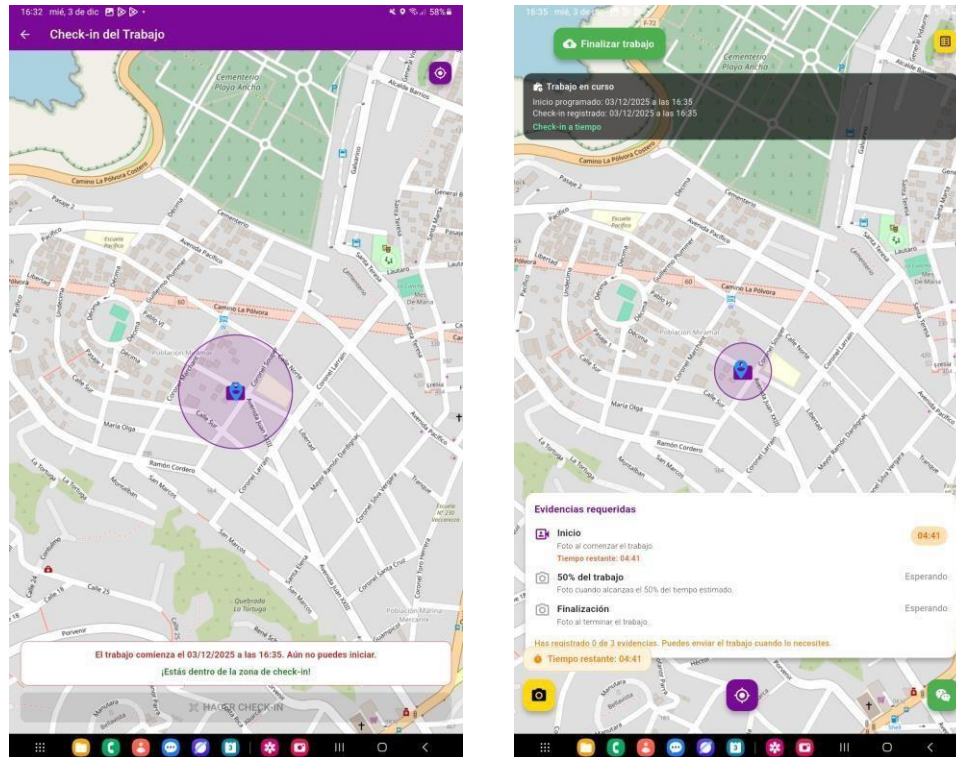


Figura 17-18. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

Esta etapa corresponde a la ejecución en terreno desde la perspectiva del trabajador y comienza cuando el usuario intenta realizar el check in en el lugar definido para el trabajo. La aplicación utiliza un control combinado de ubicación y tiempo, de manera que el inicio solo puede registrarse si el usuario se encuentra dentro del perímetro autorizado y además se cumple la ventana horaria establecida. En la primera captura se observa un escenario frecuente en operaciones reales, el trabajador ya llegó al punto correcto y la interfaz confirma que está dentro de la zona de check in mediante la visualización del área delimitada en el mapa, sin embargo, la acción principal se mantiene inhabilitada porque aún no se alcanza la hora de inicio programada. Este comportamiento busca evitar registros prematuros que distorsionen la medición de asistencia y también reduce ambigüedad al explicar explícitamente por qué no se permite continuar, entregando al usuario una señal clara de que su ubicación es correcta y que solo falta el requisito temporal.

Una vez alcanzada la hora definida, el usuario puede iniciar el trabajo y el sistema transita a un estado de "trabajo en curso", tal como se aprecia en la segunda captura. En este momento la aplicación registra el inicio efectivo de la jornada para su verificación posterior y muestra un resumen operativo con la hora programada y el momento de check in, además se comunica si el ingreso fue a tiempo o con retraso, lo que permite transparencia inmediata respecto del cumplimiento horario. En paralelo se despliega un panel de control de ejecución que incluye un temporizador o indicador de tiempo restante estimado, junto con un conjunto de evidencias fotográficas requeridas distribuidas a lo largo de la jornada. En particular se solicitan tres evidencias asociadas a hitos, una al inicio, una al alcanzar aproximadamente el 50 por ciento del tiempo estimado y una al

finalizar, con el objetivo de construir un rastro mínimo verificable de presencia y continuidad sin depender de coordinación externa.

El diseño de esta pantalla convierte la ejecución en un flujo guiado donde el trabajador sabe qué acciones debe completar y en qué momento, reduciendo omisiones y mejorando consistencia del registro. Cada evidencia se presenta como una tarea con estado, lo que permite al usuario identificar si está pendiente o completada y facilita la planificación durante la jornada, mientras que la acción de finalización queda disponible como control de cierre cuando corresponde. En términos de trazabilidad, esta etapa es crítica porque integra en un mismo punto la verificación de ubicación, el registro temporal de inicio y el seguimiento de evidencias, generando información estructurada que posteriormente puede ser revisada por el administrador para validar cumplimiento y respaldar el proceso de cierre y pago

3.6.11 Pago a trabajador

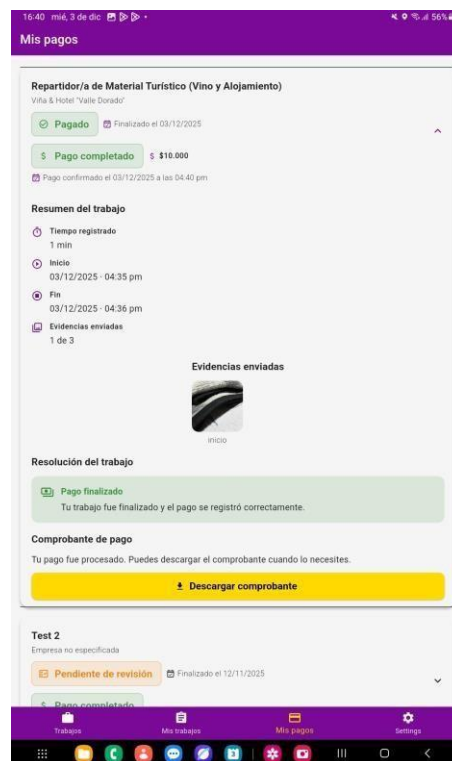


Figura 19. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

Una vez concluida la ejecución y realizada la validación correspondiente, la aplicación refleja el cierre financiero del proceso en la sección "Mis pagos". En esta pantalla el trabajador dispone de un historial de trabajos asociados a pagos, organizado en tarjetas que permiten revisar el estado de cada caso. Tal como se mencionó en el alcance, esta sección actúa como un visor de estado administrativo; el administrador registra manualmente que la transferencia fue realizada y adjunta el comprobante, el cual queda disponible para descarga por parte del trabajador, cerrando el ciclo de información sin integrar una pasarela bancaria en tiempo real.

Al desplegar el detalle de un pago, la aplicación presenta un resumen estructurado del trabajo, incluyendo el tiempo registrado y las marcas temporales de inicio y término. Esta información cumple un rol de trazabilidad, ya que permite al trabajador verificar que el registro de asistencia coincide con lo ocurrido en terreno y, al mismo tiempo, actúa como respaldo en caso de discrepancias. En el mismo detalle se informa la cantidad de evidencias enviadas a administración y se muestran las evidencias registradas, lo que vincula el pago con el cumplimiento operacional y deja explícita la relación entre ejecución, verificación y cierre.

La sección incluye además un bloque de resolución que confirma que el trabajo fue finalizado y que el pago se registró correctamente, reforzando la comunicación de estado final del proceso. Finalmente, se incorpora la opción de descargar un comprobante de pago, lo que entrega un documento verificable para el trabajador y consolida el cierre administrativo. En conjunto, esta pantalla integra en un único punto el resultado económico y el respaldo operativo asociado, fortaleciendo la percepción de control del usuario y aportando evidencia estructurada que facilita auditoría y seguimiento.

3.7 Recorrido por la aplicación Vista Administrador

Este apartado describe el recorrido principal dentro de la aplicación desde la perspectiva del administrador, responsable de publicar ofertas, supervisar el avance de los trabajos y gestionar la transición de estados a lo largo del ciclo, desde la publicación hasta el cierre y pago. A diferencia del rol trabajador, cuyo foco es la postulación y la ejecución, el rol administrador se orienta a la gestión operativa, la revisión de evidencia y la toma de decisiones que impactan directamente en la trazabilidad del proceso.

3.7.1 Vista principal de trabajos

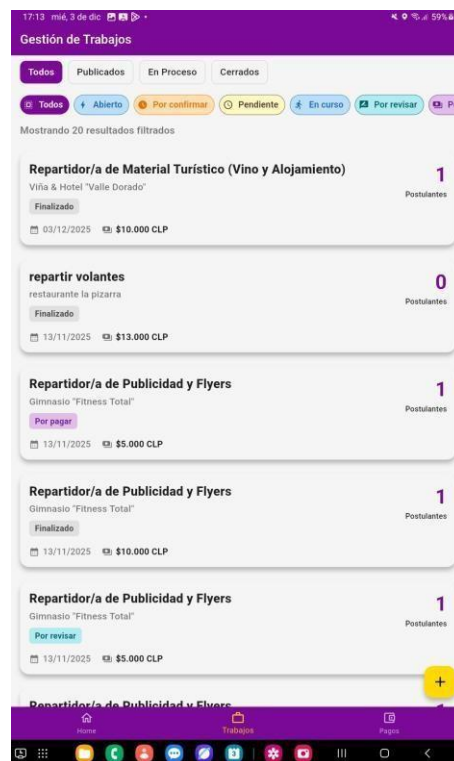


Figura 20. Captura de pantalla aplicación GoodJobs!

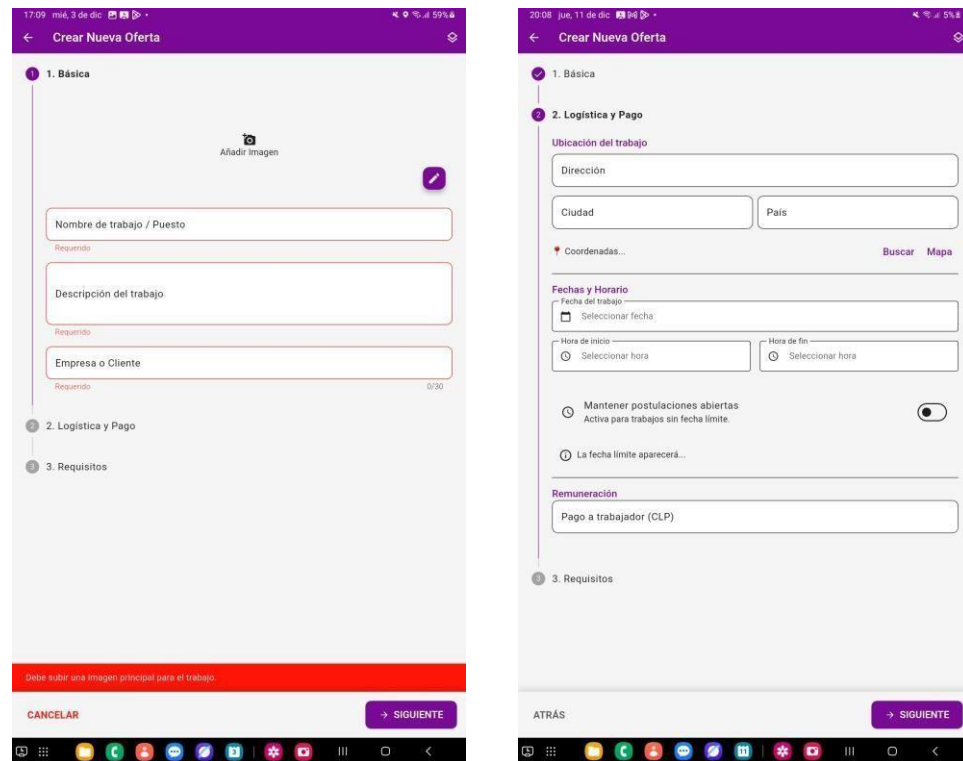
Fuente: Elaboración propia.

La vista principal del administrador corresponde a la sección de gestión de trabajos, donde se presenta el listado de ofertas creadas y su estado actual. La pantalla se estructura para permitir control rápido del volumen de trabajos y para priorizar acciones mediante filtrado, integrando una barra de categorías superiores y un conjunto de filtros por estado.

En la parte superior se dispone un conjunto de pestañas que separa el universo de trabajos en grupos de navegación, por ejemplo, una vista general y vistas por etapa de ciclo como publicados, en proceso y cerrados. Complementariamente, se incluyen filtros por estado que permiten refinar el listado a condiciones específicas de operación, tales como abierto, por confirmar, pendiente, en curso, por revisar, por pagar, pagado y pendiente de confirmación. Este diseño permite al administrador identificar rápidamente qué trabajos requieren intervención inmediata, por ejemplo, aquellos que esperan confirmación, aquellos que están en curso y requieren monitoreo o aquellos que quedaron listos para revisión o pago.

Cada tarjeta del listado presenta información mínima para toma de decisiones sin necesidad de entrar al detalle, incluyendo el nombre del trabajo, la entidad o lugar asociado, la etiqueta de estado, la fecha y el monto definido, además de un contador de postulantes visible en el lateral derecho. Esta combinación permite al administrador evaluar demanda y priorizar revisión, por ejemplo, detectando ofertas con alta postulación o trabajos que avanzaron a estados de cierre. Como elemento de acción, la pantalla incluye un botón flotante de creación que permite publicar un nuevo trabajo desde cualquier punto del listado, reduciendo fricción para la operación y favoreciendo continuidad de gestión.

3.7.2 Creación de trabajo



The figure displays two sequential screenshots of the 'Crear Nueva Oferta' (Create New Offer) process in the GoodJobs! mobile application. The first screenshot, labeled '1. Básica' (Basic), shows a form with three main sections: 'Nombre de trabajo / Puesto' (Job Name / Position), 'Descripción del trabajo' (Job Description), and 'Empresa o Cliente' (Company or Client). Each section has a 'Requerido' (Required) label. A red banner at the bottom states 'Debe subir una imagen principal para el trabajo' (You must upload a main image for the job). The second screenshot, labeled '2. Logística y Pago' (Logistics and Payment), shows a form with several sections: 'Ubicación del trabajo' (Job Location) with fields for 'Dirección' (Address), 'Ciudad' (City), and 'País' (Country); 'Coordenadas' (Coordinates) with a 'Buscar' (Search) button and a 'Mapa' (Map) link; 'Fechas y Horario' (Dates and Schedule) with fields for 'Fecha del trabajo' (Job Date), 'Hora de inicio' (Start Time), and 'Hora de fin' (End Time); a toggle for 'Mantener postulaciones abiertas' (Keep applications open) with the text 'Activa para trabajos sin fecha límite' (Active for jobs without a deadline); and 'Remuneración' (Remuneration) with a field for 'Pago a trabajador (CLP)' (Payment to worker (CLP)). Both screenshots show a progress indicator at the top and a 'SIGUIENTE' (Next) button at the bottom right.

Figura 21-22. Captura de pantalla aplicación GoodJobs!

Fuente: Elaboración propia.

La creación de un nuevo trabajo se realiza mediante un formulario asistido por pasos, diseñado para asegurar completitud de la información antes de publicar una oferta. Esta estructura secuencial reduce errores típicos de publicación, ya que obliga a que cada etapa cumpla validaciones mínimas antes de habilitar el avance, evitando ofertas con datos incompletos que podrían generar confusión en postulantes o dificultades en la ejecución. En las capturas se observan los pasos principales del flujo, donde el sistema marca los campos requeridos y restringe la navegación si estos no han sido completados. En el primer paso, orientado a la información básica, se solicita una imagen principal que funciona como banner de la oferta y se utiliza posteriormente en la vista de detalle, reforzando la identificación visual del trabajo.

En esta etapa también se captura el nombre del puesto o trabajo, una descripción clara de la actividad a realizar y la empresa o cliente asociado. Exigir estos elementos desde el inicio permite que la oferta sea comprensible para el trabajador y disminuye el riesgo de postulaciones basadas en información insuficiente.

En el segundo paso, correspondiente a logística y pago, el administrador define la ubicación y los parámetros temporales. Para la ubicación se incorporan campos de dirección y referencias geográficas, junto con un componente de coordenadas que por defecto utiliza la ubicación actual para facilitar el ingreso, aunque permite edición manual cuando el lugar de ejecución no coincide con la posición del administrador. Esta decisión de diseño mejora eficiencia sin sacrificar control y contribuye a la precisión del punto de trabajo, lo cual es relevante si posteriormente se aplican validaciones por geocerca. En el mismo paso se registran fechas y horario, incluyendo el día del trabajo y las horas de inicio y término, y se incorpora la definición de la fecha límite de postulaciones. En

condiciones normales, el sistema calcula automáticamente la fecha límite a partir de la configuración temporal del trabajo, lo que estandariza el comportamiento y reduce errores de parametrización, sin embargo, el administrador puede activar la opción de mantener postulaciones abiertas cuando el trabajo no requiere una fecha límite explícita, permitiendo mayor flexibilidad para ofertas recurrentes o de disponibilidad prolongada. Finalmente se define la remuneración, expresada como pago al trabajador, completando la información mínima necesaria para que la oferta sea evaluable y comparable.

En conjunto, este flujo de creación no solo publica una oferta, sino que establece la base de trazabilidad operacional, ya que fija de manera estructurada los elementos que luego se utilizan para explorar, postular, ejecutar y validar el trabajo, incluyendo identidad visual, ubicación precisa, ventanas horarias y condiciones económicas.

3.7.3 Aceptación de postulante

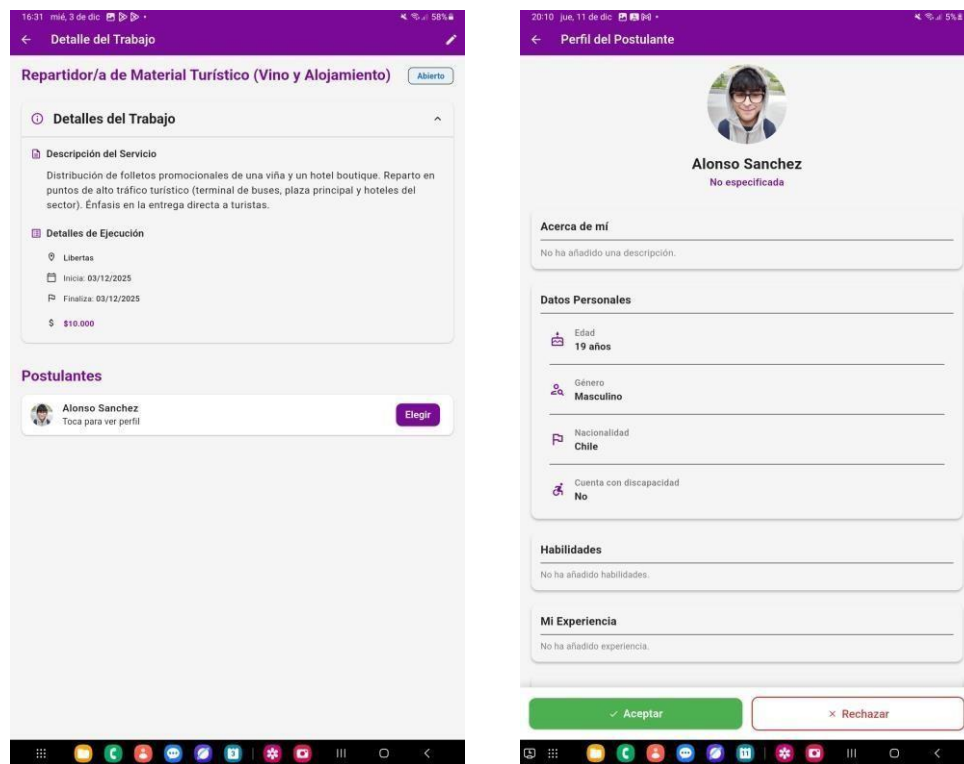


Figura 23-24. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

Una vez publicada la oferta y recibidas postulaciones, el administrador gestiona la selección desde la vista de detalle del trabajo. En esta pantalla se presenta un resumen de la oferta con su estado actual y los elementos operativos necesarios para contextualizar la decisión, incluyendo la descripción del servicio y los detalles de ejecución, como lugar asociado, fechas y remuneración. Este resumen permite que la evaluación del postulante se realice en función de los requisitos y condiciones reales del trabajo, evitando decisiones desconectadas del contexto operativo.

En la sección de postulantes se listan los candidatos asociados a la oferta y el administrador puede acceder al perfil de cada uno para revisar su información antes de tomar una decisión. Al ingresar al perfil del postulante, la aplicación centraliza datos

personales relevantes para la asignación, tales como edad, género, nacionalidad y declaración de discapacidad, además de secciones destinadas a descripción personal, habilidades y experiencia. La presencia de estos campos responde a la necesidad de apoyar decisiones informadas y reducir riesgos de asignación, por ejemplo, asegurando compatibilidad con requisitos del trabajo o considerando condiciones que pudieran requerir adecuaciones.

La interfaz proporciona acciones explícitas para resolver la postulación, permitiendo aceptar o rechazar al candidato. Esta resolución formaliza el cambio de estado del postulante dentro del flujo y habilita las etapas posteriores, como confirmaciones, ejecución y seguimiento, consolidando trazabilidad del proceso al registrar una decisión administrativa asociada a una oferta específica. En términos de control, el diseño busca equilibrar rapidez operativa con respaldo informativo, ya que presenta en un mismo recorrido tanto el contexto del trabajo como la información del postulante, disminuyendo la probabilidad de selección errónea y facilitando la gestión de múltiples ofertas en paralelo.

3.7.4 Pago de trabajo

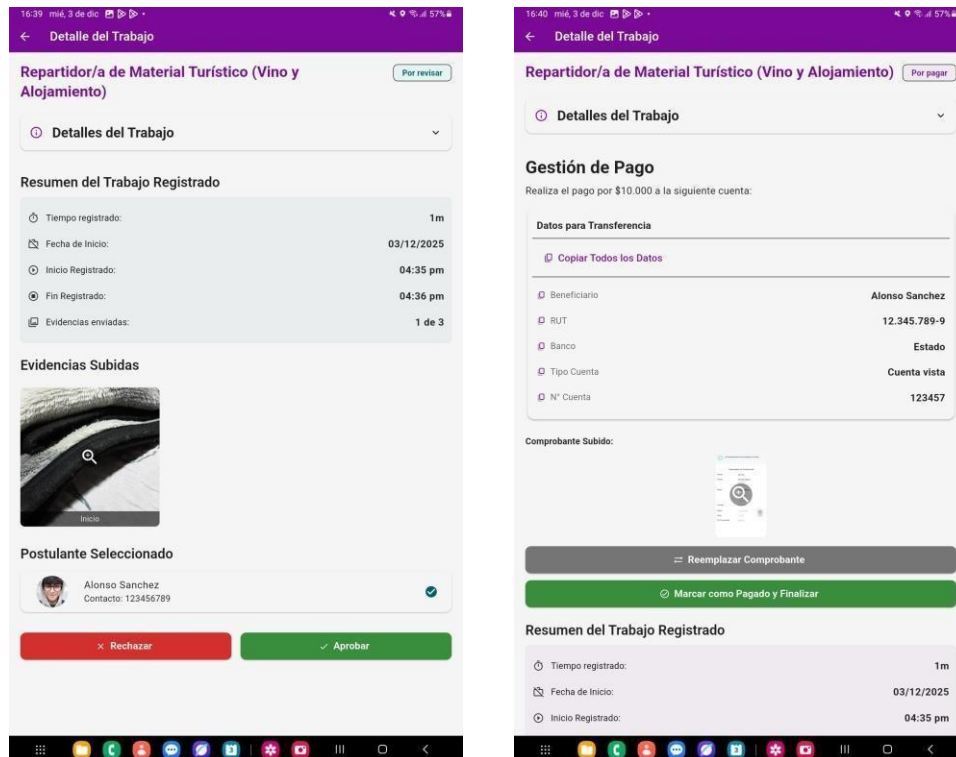


Figura 25-26. Captura de pantalla aplicación GoodJobs!
Fuente: Elaboración propia.

Tras la finalización de la jornada por parte del trabajador, el flujo administrativo se concentra en una etapa de revisión y pago, cuyo propósito es validar el cumplimiento antes de registrar el cierre financiero. En esta vista el trabajo se presenta en estado "por revisar" y se despliega un resumen del registro asociado, incluyendo tiempo registrado, fecha, hora de inicio y término, junto con la cantidad de evidencias enviadas. Este resumen permite una primera verificación de consistencia temporal y actúa como referencia para contrastar el desarrollo de la actividad con lo esperado.

El elemento principal para la toma de decisión corresponde a las evidencias subidas, las cuales se muestran como recursos visuales consultables. Con esta información el administrador determina si el trabajo cumple con los criterios definidos y puede resolver la instancia mediante acciones explícitas de rechazo o aprobación. La resolución constituye un punto crítico de trazabilidad, ya que registra una decisión administrativa vinculada a un conjunto de evidencias y a un registro temporal concreto.

Cuando el trabajo es aprobado, la aplicación habilita el estado "por pagar" y presenta el módulo de gestión de pago con los datos bancarios del trabajador necesarios para ejecutar la transferencia, incluyendo beneficiario, RUT, banco, tipo de cuenta y número de cuenta. Para disminuir errores de digitación y acelerar la operación se incorpora una función para copiar el conjunto de datos de transferencia. Posteriormente el administrador puede adjuntar o reemplazar el comprobante de pago, dejando respaldo verificable dentro del sistema y fortaleciendo la capacidad de auditoría. Finalmente, al marcar como pagado y finalizar, el trabajo queda cerrado en un estado coherente con el historial de pagos visible para el trabajador, consolidando en un flujo continuo la evidencia operacional, la decisión de validación y el respaldo financiero.

4 Pruebas y validación

Este capítulo describe el enfoque de validación aplicado durante el desarrollo de la aplicación y resume las verificaciones realizadas para asegurar que los flujos críticos funcionen de forma consistente. A diferencia de un proceso de pruebas formal con casos cerrados y anexos extensos, la validación se ejecutó principalmente mediante pruebas constantes de flujo basadas en prueba y error, es decir, ciclos iterativos donde se ejecutaba el recorrido completo de una funcionalidad.

Para formalizar la validación, se utilizaron criterios de aceptación explícitos definidos en las Historias de Usuario. A continuación, se presenta una selección de los criterios más críticos validados durante los Sprints, los cuales garantizan la integridad operativa y financiera del sistema:

1. Verificación de Evidencias y Trazabilidad

- Dado que estoy subiendo una foto de evidencia, el sistema captura y adjunta automáticamente las coordenadas GPS y la fecha/hora actual a la imagen, sin permitir edición manual.
- Dado que he subido las evidencias requeridas para el trabajo, entonces el estado del trabajo cambia automáticamente a "Pendiente de Revisión" en la consola del administrador.
- Resultado: Se comprobó que el sistema rechaza envíos sin coordenadas y que el cambio de estado es inmediato, asegurando que el administrador reciba datos veraces.

2. Gestión de Tiempos y Confirmación

- Dado que una postulación ha sido aceptada por el administrador, el sistema inicia un temporizador interno para el plazo de confirmación.
- Si el trabajador no confirma la asignación dentro de ese plazo, entonces el trabajo vuelve automáticamente al estado "Abierta para Postulaciones" y se libera el cupo.
- Resultado: Se validó la liberación automática de cupos, impidiendo que trabajos queden bloqueados por usuarios inactivos.

3. Validación Geográfica para Check-in (Criterio de Seguridad Operativa)

- Dado que intento iniciar un trabajo asignado, si mi ubicación GPS está fuera del radio permitido respecto al punto del trabajo, entonces el botón de "Iniciar Trabajo" permanece deshabilitado.
 - Resultado: La prueba confirmó que el sistema impide iniciar el contador de horas si el dispositivo no reporta estar físicamente en el lugar. Se validó técnicamente el uso de un radio de tolerancia de 100 metros para la geocerca; esta distancia fue seleccionada para equilibrar la precisión del GPS civil (que puede variar en zonas urbanas densas) con la usabilidad, evitando falsos negativos donde el usuario, estando en el sitio correcto, no pudiera realizar el check-in debido a fluctuaciones menores de la señal satelital.
4. Integridad del Proceso de Pago
- Dado que el administrador marca un trabajo como "Pagado", entonces el sistema actualiza el estado visible para el trabajador y bloquea futuras modificaciones en el monto o fecha del trabajo.
 - Resultado: Se asegura que una vez cerrado el ciclo financiero, los datos operativos se vuelven inmutables para garantizar consistencia histórica.

4.1 Objetivo y alcance

El objetivo general de las pruebas es comprobar que la aplicación soporta el ciclo de trabajo de extremo a extremo, desde la publicación y selección hasta la ejecución con registro verificable y el cierre con pago. En términos específicos, se validó que el sistema prevenga acciones inválidas mediante restricciones y mensajes claros, que la información quede persistida de manera consistente y que los estados se reflejen correctamente en las vistas correspondientes para trabajador y administrador.

El alcance se concentró en los flujos críticos asociados a trazabilidad. En el rol trabajador, registro y habilitación, exploración y postulación, seguimiento, ejecución con control de ubicación y tiempo, captura de evidencias y consulta de pagos. En el rol administrador, creación de ofertas, gestión por estados, selección de postulantes, revisión de ejecución y registro del pago con comprobante.

4.2 Validación de flujos del rol trabajador

4.2.1 Registro, autenticación y consistencia del perfil

Se ejecutó el registro por etapas verificando que los campos obligatorios bloqueen el avance cuando están incompletos y que los errores de formato se comunican en el punto de ingreso. Se verificó que el inicio de sesión permita acceso con credenciales válidas, que rechace credenciales inválidas sin generar estados intermedios y que el perfil resultante quede disponible para el resto del flujo.

4.2.2 Habilitación por requisitos mínimos y prevención de postulaciones inválidas

Se validó el comportamiento de cuenta incompleta comprobando que la aplicación impida postular cuando no se cumplen requisitos operativos, específicamente verificación de correo y registro de datos bancarios. Se verificó que la interfaz guíe al usuario hacia la resolución y que, una vez completados los pasos, el estado de habilitación permita avanzar de forma consistente. El resultado esperado es que ninguna postulación quede registrada si el usuario no está habilitado, evitando retrabajo posterior y retrasos en el pago.

4.2.3 Exploración, detalle y claridad de información previa a postulación

Se verificó que la exploración muestre ofertas con información mínima comparable y que el detalle entregue los elementos necesarios para decidir, incluyendo remuneración, fecha, horario y ubicación. Se validó que los cambios de visualización y ordenamiento mantengan consistencia del listado y que al entrar y salir del detalle no se pierda el contexto. El resultado esperado es reducir postulaciones basadas en información incompleta y mejorar la decisión del usuario.

4.2.4 Inicio de trabajo con control temporal y geográfico

Se probó el comportamiento de check in bajo dos condiciones principales. En la primera, el usuario se encuentra dentro de la zona definida, pero intenta iniciar antes del horario programado, caso en el cual la acción debe permanecer inhabilitada y el sistema debe comunicar que la ubicación es correcta pero el tiempo aún no corresponde. En la segunda, el usuario intenta iniciar dentro de la ventana válida y dentro del perímetro, caso en el cual el sistema debe permitir el inicio, registrar el tiempo de entrada y mostrar el trabajo en curso con su resumen operativo.

4.2.5 Evidencias fotográficas durante la ejecución y cierre

Se verificó el flujo de evidencias requerido durante un trabajo en curso, comprobando que la aplicación solicite evidencias según hitos definidos, que el estado de cada evidencia cambie de pendiente a registrada y que el usuario pueda completar el recorrido sin pérdida de información. Se validó que el resumen del trabajo registre evidencias enviadas y que el cierre refleje correctamente el término de la jornada. El resultado esperado es que el trabajador cuente con un flujo guiado que reduzca omisiones y que el administrador disponga de evidencia estructurada para la revisión.

4.2.6 Consulta de pagos y respaldo del cierre

Se validó que el trabajador pueda acceder a un historial de pagos y abrir el detalle de un pago completado, verificando que se muestre el tiempo registrado, marcas de inicio y fin, evidencias asociadas y disponibilidad del comprobante. El resultado esperado es transparencia del cierre y disponibilidad de respaldo en un solo punto, lo que reduce incertidumbre y consultas externas.

4.3 Validación de flujos del rol administrador

4.3.1 Gestión por estados y priorización operativa

Se verificó que la vista principal de administración permita identificar trabajos por estado y filtrar correctamente entre categorías operativas, de modo que el administrador pueda priorizar tareas, por ejemplo, revisar trabajos pendientes o gestionar pagos.

4.3.2 Creación de oferta por etapas con validación de campos

Se validó el formulario de creación por pasos comprobando que no se permita avanzar si faltan campos requeridos y que la oferta final incluya información mínima consistente. Se verificó el ingreso de imagen principal, descripción, empresa, ubicación con coordenadas utilizando ubicación actual como referencia y edición manual, fecha y horario, remuneración y configuración de fecha límite.

4.3.3 Selección de postulante con apoyo de información del perfil

Se ejecutó el flujo de revisión de postulantes desde el detalle del trabajo, verificando que el administrador pueda acceder al perfil del candidato, revisar datos personales disponibles y resolver mediante aceptar o rechazar. El resultado esperado es que la decisión quede registrada y se refleje en el estado del trabajo, habilitando el paso siguiente sin generar inconsistencias.

4.3.4 Revisión de ejecución basada en evidencias y registro de pago

Se validó la etapa posterior a la ejecución comprobando que el sistema muestre el resumen de trabajo registrado y las evidencias subidas para sustentar la decisión de aprobar o rechazar. Al aprobar, se verificó la transición a estado por pagar y la presentación estructurada de datos bancarios del trabajador. Se validó la función de copiado de datos para disminuir errores de transferencia, la carga o reemplazo del comprobante y el cierre marcando el trabajo como pagado. El resultado esperado es que el pago quede respaldado y que el estado final sea consistente con lo que observa el trabajador en su historial.

4.4 Síntesis de resultados de validación

A partir de las pruebas iterativas por flujo se estabilizaron los recorridos críticos del sistema, logrando consistencia de estados entre trabajador y administrador. La validación por prueba y error permitió detectar fallas de integración y fricciones de comprensión en el momento en que ocurren, corrigiéndolas en ciclos cortos hasta obtener un comportamiento predecible. Como resultado, la aplicación soporta el ciclo completo de un trabajo, desde su creación y selección hasta su ejecución con controles y su cierre con pago, manteniendo información estructurada que facilita revisión, auditoría y consulta posterior.

5 Impacto

Este capítulo describe el impacto esperado de la implementación, considerando efectos operativos y experiencia de usuario.

Si bien no se cuenta con registros históricos estructurados del proceso manual para realizar una comparación estadística exacta, es posible establecer una estimación de impacto basada en la eficiencia del flujo. La implementación de la solución digital transforma actividades asíncronas y manuales en procesos transaccionales inmediatos.

A continuación, se presenta una proyección de la mejora:

Indicador Operativo	Situación Actual	Situación Propuesta	Impacto Estimado
Tiempo de Asignación	2 - 4 horas. Requiere negociación individual y confirmación manual en chat.	< 15 minutos. Notificación Push masiva y confirmación en un clic.	Reducción del 90% en latencia de coordinación.

Validación de Asistencia	Confianza / Visual. No existe prueba digital irrefutable de la hora de llegada.	Geocerca + Timestamp. Validación automática por GPS bloqueante.	Eliminación de reportes de asistencia inexactos.
Consolidación de Pagos	Días. Recopilación manual de datos bancarios y cruce con planillas.	Inmediata. Datos estructurados y listos para procesar tras el cierre.	Optimización total del tiempo administrativo previo al pago.

Figura 27. Tabla de proyección de mejora.

Fuente: Elaboración propia.

Reducción de Latencia: La asignación evoluciona de un modelo de negociación asíncrona a uno de confirmación digital inmediata.

Integridad de Datos: Se mitiga el riesgo de error humano en la transcripción de datos bancarios y horas trabajadas, asegurando que el pago corresponda exactamente a lo registrado en el sistema.

5.1 Impacto en el trabajador

Para el trabajador, la aplicación aporta transparencia y autonomía. Al centralizar ofertas, estados de postulación, ejecución y pagos, se reduce la necesidad de coordinar por canales externos y se disminuye incertidumbre respecto de asignación y cierre. Impactó en la gestión operativa.

5.2 Impacto en el administrador

Para el administrador, el impacto principal se asocia a la reducción de carga manual y a la estandarización del ciclo. La gestión por estados, la creación asistida de ofertas y la revisión basada en evidencias permiten priorizar acciones y resolver trabajos con información estructurada.

5.3 Impacto en trazabilidad y control

El sistema integra registros temporales, controles de ubicación y evidencias fotográficas, lo que fortalece la capacidad de reconstruir eventos en caso de discrepancias. Este impacto es especialmente relevante en trabajos dispersos geográficamente, donde la validación manual es costosa.

5.4 Análisis de rendimiento, escalabilidad y costos

El costo y el rendimiento de una arquitectura sobre Firestore quedan determinados por el número de operaciones de lectura y escritura, por lo que el análisis se construye sobre el patrón de acceso real de la aplicación. La pantalla de exploración mantiene un listener sobre la colección de trabajos ordenada por fecha de creación: el primer snapshot de una sesión cuesta tantas lecturas como ofertas publicadas y, a partir de ahí, una lectura por documento que cambia. Con el volumen operativo esperado, el costo por sesión es de decenas de lecturas; el análisis identifica, sin embargo, que este costo crece linealmente con el catálogo de ofertas, por lo que la paginación de consultas es la optimización prioritaria antes de un crecimiento sostenido del catálogo.

Sobre ese patrón se construye el siguiente modelo de costos. Supuestos declarados: un usuario activo genera del orden de 60 lecturas diarias y 10 escrituras diarias; las tarifas de referencia del proveedor son US\$0,06 por cada 100.000 lecturas y US\$0,18 por cada 100.000 escrituras, con una cuota gratuita diaria de 50.000 lecturas y 20.000 escrituras [6].

Usuarios activos por día	Lecturas por día	Escrituras por día	Costo mensual proyectado
100	6.000 (12% de la cuota gratuita)	1.000 (5% de la cuota)	US\$0
500	30.000 (60% de la cuota gratuita)	5.000 (25% de la cuota)	US\$0
1.000	60.000 (excedente: 10.000/día)	10.000 (50% de la cuota)	≈ US\$0,18

Figura 29. Proyección de costos operativos de Firestore según usuarios activos diarios.

Fuente: Elaboración propia.

El resultado del modelo es que el costo operativo marginal es despreciable hasta el orden de 1.000 usuarios activos diarios, un umbral muy superior a la operación actual de SurModel; esto valida cuantitativamente la elección serverless frente al costo fijo de un servidor autoadministrado, que se paga desde el primer día con independencia del uso. En cuanto a límites de la plataforma, la restricción más relevante de Firestore es la tasa de escritura sostenida sobre un mismo documento (del orden de una escritura por segundo); el documento más contenido del modelo es el propio trabajo, cuyo campo de estado recibe solo unas pocas escrituras a lo largo de todo su ciclo de vida, por lo que el margen respecto del límite es de varios órdenes de magnitud. Finalmente, en materia de latencia percibida, la base de datos reside en la región southamerica-west1, lo que minimiza el tiempo de ida y vuelta de las operaciones interactivas para usuarios en Chile, y las notificaciones push entregan los hitos del ciclo (asignación, recordatorios, cancelación) con latencias del orden de segundos; sobre esta base se sustenta, como estimación analítica y no como medición, la reducción de latencia de coordinación proyectada en la Figura 27.

5.5 Análisis de riesgos

La siguiente matriz sistematiza los riesgos técnicos y operativos identificados durante el desarrollo, clasificados por probabilidad e impacto, junto con su mitigación implementada o propuesta.

Riesgo	Prob.	Impacto	Severidad	Mitigación
R1. Suplantación de ubicación en el check-in: la verificación de geocerca se ejecuta en el cliente.	Media	Alto	Alta	Trasladar la verificación de distancia a Cloud Functions y validar la integridad del dispositivo (propuesta)
R2. Pérdida de conectividad durante check-in o carga de evidencias en terreno.	Media	Medio	Media	Persistencia offline del SDK de Firestore y reintentos de carga; degradación funcional

				comunicada al usuario (parcialmente implementada)
R3. Imprecisión del GPS civil en zonas urbanas densas.	Alta	Bajo	Media	Radio de tolerancia de 100 m definido para absorber la fluctuación de la señal (implementada)
R4. Crecimiento del costo de lecturas por listeners sin paginación al aumentar el catálogo de ofertas.	Media	Medio	Media	Paginación con limit y cursores; consultas filtradas por estado (propuesta, priorizada)
R5. Dependencia del proveedor.	Baja	Alto	Media	Acceso a datos aislado en la capa de servicios de la aplicación, que acota el costo de una migración (implementada como patrón de diseño)
R6. Exposición de datos sensibles como el RUT o datos bancarios.	Baja	Alto	Media	Reglas de acceso por rol y propiedad a nivel de documento, restricción de campos modificables por el trabajador y funciones de cifrado respaldadas en un gestor de secretos (implementada)
R7. Dependencia de los servidores públicos de teselas de OpenStreetMap.	Media	Medio	Media	Migrar a un proveedor de teselas con acuerdo de servicio manteniendo flutter_map (propuesta)
R8. Tokens de notificación inválidos u obsoletos.	Media	Bajo	Baja	Registro de tokens por dispositivo con actualización automática y detección de tokens fallidos en el envío (implementada)

Figura 30. Matriz de riesgos técnicos y operativos con mitigaciones.

Fuente: Elaboración propia

6 Trazabilidad de objetivos y evidencia de cumplimiento

Esta sección contrasta cada objetivo específico del proyecto con un indicador de verificación y con la evidencia disponible al cierre del trabajo, distinguiendo explícitamente entre lo verificado funcionalmente y lo que queda sujeto a medición en operación real.

Objetivo específico	Indicador de verificación	Evidencia disponible	Estado
---------------------	---------------------------	----------------------	--------

Definir la arquitectura técnica y el modelo de datos en Firebase.	Arquitectura y modelo documentados y desplegados en el entorno del proyecto	Arquitectura serverless y modelo de colecciones implementados y en operación (Figuras 3 y 4); comparación formal de arquitecturas (Figura 28)	Cumplido
Implementar componentes críticos: geolocalización, validación de evidencias y reglas de seguridad.	Criterios de aceptación de las historias de usuario críticas verificados	Check-in con geocerca de 100 m y control horario; tres evidencias por hitos con coordenadas y marca de tiempo; reglas con control de acceso por rol y restricción por campo; criterios de aceptación validados en el capítulo de Pruebas y validación	Cumplido
Mejorar la experiencia de uso para colaboradores y administradores.	Puntaje SUS ≥ 68 en evaluación con usuarios finales	Evidencia indirecta: retroalimentación de Sprint Reviews y del patrocinador. Sin medición estandarizada con usuarios finales	Parcial: medición pendiente
Reducir de forma consistente el tiempo entre publicación y asignación de trabajos.	Mediana de (fecha de aceptación – fecha de publicación) inferior a 15 minutos en operación real	Instrumentación completa: el esquema ya registra las marcas temporales necesarias (creación, aceptación, confirmación). Sin datos de operación real ni línea base histórica del proceso manual	Pendiente: plan de medición definido

Figura 31. Matriz de trazabilidad entre objetivos específicos, indicadores y evidencia.

Fuente: Elaboración propia

6.1 Limitaciones de la validación y plan de medición

La principal limitación del trabajo es la ausencia de medición cuantitativa en operación real: no existe una línea base histórica estructurada del proceso manual de SurModel y el sistema no acumuló aún operación productiva suficiente para calcular indicadores. Por

esa razón, las cifras de la Figura 27 deben leerse como proyecciones analíticas fundamentadas en el análisis de este capítulo, y no como resultados medidos.

Esta limitación está acotada por diseño: el modelo de datos ya registra, como parte del flujo normal, todas las marcas temporales que los indicadores requieren, entre ellas la fecha de creación de la oferta, la fecha de aceptación y confirmación de la asignación, el inicio y término reales del trabajo, la duración registrada, el retraso de check-in en minutos y la liberación de asignaciones por expiración. En consecuencia, los indicadores comprometidos son computables directamente sobre los datos existentes, sin desarrollo adicional, en cuanto exista operación real.

El plan de medición propuesto consiste en un piloto acotado con SurModel (del orden de 20 a 30 trabajos reales) que permita calcular: la mediana del tiempo publicación-asignación, contrastada con el umbral de 15 minutos; la tasa de puntualidad de check-in y la distribución del retraso en minutos; la tasa de expiración de confirmaciones y el tiempo de re-publicación de cupos; y la completitud de evidencias por trabajo. Para el objetivo de experiencia de uso, se aplicaría el cuestionario estandarizado SUS a los trabajadores y administradores del piloto, adoptando el umbral convencional de 68 puntos como criterio de aceptación. Con esos datos, la tabla de la Figura 27 podría reemplazar sus estimaciones por valores observados.

7 Conclusiones

Este proyecto permitió llevar a la práctica muchos conocimientos adquiridos a lo largo de la carrera, aplicándolos en un contexto real y con necesidades concretas. Más allá de implementar una solución, el proceso obligó a enfrentar problemas propios de un sistema operativo en escenarios reales, como asegurar que los flujos sean consistentes, que las reglas estén claras y que la información se mantenga ordenada y verificable. En ese sentido, el aprendizaje no fue solo técnico, sino también metodológico, ya que fue necesario planificar, priorizar, iterar y corregir sobre la marcha hasta obtener un resultado estable.

Una conclusión importante es que la calidad de un proyecto de este tipo no depende únicamente de “que funcione”, sino de que el proceso quede bien definido y que el sistema sea capaz de sostenerlo sin generar confusión. Se aprendió que decisiones aparentemente pequeñas, como mantener estados coherentes, bloquear acciones inválidas y comunicar correctamente por qué una acción no se puede realizar, terminan siendo claves para la confianza del usuario y para la trazabilidad del ciclo completo. También se reforzó la importancia de tratar la información sensible con responsabilidad, ya que se manejan datos personales, evidencias y datos bancarios, lo que exige segmentación por rol y exposición controlada de información.

El trabajo en equipo fue fundamental para lograr el resultado. La coordinación permitió dividir responsabilidades, complementar habilidades y mantener continuidad en las tareas, especialmente en etapas donde hubo que integrar componentes y validar recorridos completos. Sin esta colaboración habría sido muy difícil cumplir plazos y mantener consistencia en el desarrollo, ya que un proyecto así requiere muchas decisiones simultáneas, pruebas repetidas y correcciones constantes.

Además, el apoyo de la empresa patrocinadora SurModel fue clave durante todo el proceso. Su disponibilidad para resolver dudas, entregar retroalimentación y validar avances facilitó la toma de decisiones y permitió orientar el trabajo hacia lo que realmente se necesitaba en la operación. Gracias a ese acompañamiento, el desarrollo



fue más claro y eficiente, y se logró cumplir con los objetivos definidos, dejando a la empresa conforme y satisfecha con el resultado.

Finalmente, se identifican aspectos que pueden seguir mejorándose. En escenarios de uso real existen variables externas como conectividad y precisión de ubicación que pueden afectar la experiencia, por lo que es relevante continuar fortaleciendo mecanismos de resiliencia y recuperación. También sería útil ampliar la evaluación con más usuarios y escenarios de prueba, incorporando métricas de usabilidad y desempeño que permitan priorizar mejoras. En síntesis, el proyecto entrega una base sólida, y al mismo tiempo deja aprendizajes claros sobre la importancia de diseñar procesos digitales con reglas comprensibles, estados consistentes y validación iterativa orientada a flujos completos.

8 Referencias

- [Flutter, «Architectural overview,» documentación, 2025. [En línea].
1 Available: <https://docs.flutter.dev/resources/architectural-overview>.
] [Último acceso: 3 Octubre 2025].
- [Flutter, «Widget catalog,» Documentación, 2025. [En línea].
2 Available: <https://docs.flutter.dev/ui/widgets>. [Último acceso: 3
] Octubre 2025].



- [Apple Developer, «Core Location: monitoring the user's proximity to
3 geographic regions,» documentación, 2025. [En línea]. Available:
] [https://developer.apple.com/documentation/corelocation/monitoring](https://developer.apple.com/documentation/corelocation/monitoring-the-user-s-proximity-to-geographic-regions)
-the-user-s-proximity-to-geographic-regions. [Ultimo acceso: 3
Octubre 2025].
- [Google Developers, «GeofencingClient,» documentacion, 2024. [En
4 línea]. Available:
] [https://developers.google.com/android/reference/com/google/android](https://developers.google.com/android/reference/com/google/android/gms/location/GeofencingClient)
d/gms/location/GeofencingClient. [Ultimo acceso: 3 Octubre 2025].
- [Android Developers, «Create and monitor geofences,»
5 documentacion, 2025. [En línea]. Available:
] [https://developer.android.com/develop/sensors-and-](https://developer.android.com/develop/sensors-and-location/location/geofencing)
location/location/geofencing. [Ultimo acceso: 3 Octubre 2025].
- [Google Firebase, «Documentation,» documentación, 2025. [En línea].
6 Available: <https://firebase.google.com/docs>. [Ultimo acceso: 3
] Octubre 2025].

