

| **1. Resumen avance Proyecto APT** |
| --- |
| Durante la fase de desarrollo implementada, se han completado los módulos principales del sistema incluyendo autenticación por roles, gestión administrativa de usuarios y proyectos, formularios de recepción y posventa, y un sistema integral de gestión de incidencias. La metodología XP (Extreme Programming) ha permitido avanzar de forma positiva mediante iteraciones, con integración continua y retroalimentación entre el equipo de trabajo. Las tecnologías empleadas incluyen React.js para el frontend, Node.js/Express para el backend, y Supabase como base de datos, cumpliendo con estándares industriales modernos. El sistema actualmente permite a beneficiarios visualizar el estado de sus viviendas y reportar incidencias, a técnicos gestionar y dar seguimiento a los casos, y a administradores controlar usuarios y proyectos habitacionales, generando reportes en PDF según exige la normativa DS49. |

| Resumen de avance proyecto APT | Actividades Realizadas y Objetivos Cumplidos Durante esta fase del proyecto hemos logrado implementar un \*\*MVP (Producto Mínimo Viable) completamente funcional\*\* que cumple con los requerimientos principales establecidos por TECHO-Chile. Los avances específicos incluyen:  Arquitectura del Sistema Implementada:   * Backend modular con Node.js y Express siguiendo patrones de arquitectura profesional (React, s.f.; Node.js, s.f.; Supabase, s.f.) * Base de datos PostgreSQL gestionada en Supabase con esquema completo y optimizado * Frontend en React.js con sistema de rutas protegidas por roles * Integración completa entre todos los componentes del sistema   Funcionalidades Desarrolladas:  1. Sistema de Autenticación y Roles:   * Registro e inicio de sesión seguro con JWT * Tres tipos de usuario: Administrador, Técnico y Beneficiario * Rutas protegidas según permisos de rol * Recuperación de contraseñas funcional   2. Módulo Administrativo:   * Gestión completa de usuarios (crear, editar, listar) * Administración de proyectos habitacionales * Gestión de viviendas con campos detallados (metros cuadrados, habitaciones, baños) * Sistema de templates de posventa configurable por tipo de vivienda * Creación automática de formularios al entregar viviendas   3. Módulo de Beneficiario:   * Visualización del estado de la vivienda asignada * Formularios de recepción de vivienda digitalizados * Sistema de reporte de incidencias post-entrega * Seguimiento del estado de incidencias reportadas   4. Módulo Técnico:   * Dashboard con estadísticas de incidencias * Gestión completa del flujo de incidencias (asignación, seguimiento, resolución) * Revisión de formularios de posventa * Generación automática de incidencias desde problemas detectados * Creación y descarga de reportes PDF   5. Características Técnicas Avanzadas:   * Geocodificación automática de direcciones * Sistema de archivos multimedia para evidencias * Generación de PDFs con Puppeteer * Almacenamiento en la nube con Supabase Storage |
| --- | --- |
| Objetivos | Tenemos como objetivo principal:  Desarrollar una plataforma web que permita a TECHO-Chile a administrar y gestionar el proceso de entrega y postventa de sus proyectos de entrega de viviendas sociales bajo la normativa DS49 (Ministerio de Vivienda y Urbanismo [MINVU], 2019), con el fin de facilitar el seguimiento del estado de viviendas y el registro de incidencias por parte de los beneficiarios, en busca de tener transparencia en los procesos.  Como objetivos de avance:   1. Crear un sistema de registro de incidencias: Implementado completamente con flujo desde reporte hasta resolución y satisfacción del beneficiario. 2. Visualización de estado de vivienda: Los beneficiarios pueden ver el estado y fecha estimada de entrega 3. Panel administrativo: Funcional para gestión de proyectos, viviendas y usuarios 4. Gestión de incidencias por técnicos: Sistema completo de asignación, seguimiento y resolución 5. Reportes PDF: Generación automática de documentos siguiendo normativa DS49 6. Seguridad y manejo de datos: Autenticación JWT, encriptación de contraseñas, control de acceso por roles 7. Usabilidad: Interfaz intuitiva diseñada para usuarios con diferentes niveles técnicos. |
| Metodología | Para el proyecto se utilizara Extreme Programming (XP), a continuación se explicara que es y como la implementamos  **¿Qué es Extreme Programming (XP)?**  XP es una metodología de desarrollo ágil que enfatiza la adaptabilidad ante cambios frecuentes de requerimientos, la calidad del software y la satisfacción del cliente. Fue creada por Kent Beck y se centra en valores fundamentales que guían todo el proceso de desarrollo. Extreme Programming (XP) enfatiza la adaptabilidad ante cambios frecuentes de requerimientos, la calidad del software y la satisfacción del cliente” (Godaddy, 2025, p. 3)  **Valores de XP Implementados**    A continuación se explica cómo aplicamos estos valores  **1. Comunicación (Communication)**   * Implementación: Reuniones semanales con profesora de portafolio para validar avances y reuniones diarias del equipo * Evidencia: Retroalimentación académica documentada y seguimiento continuo del progreso * Impacto: Permitió detectar tempranamente desviaciones del plan y ajustar el desarrollo   **2. Simplicidad (Simplicity)**   * Implementación: Utilizamos la típica frase de "Hacer la cosa más simple que funcione", priorizando la funcionalidad, pero sin dejar de lado lo visual. * Evidencia: Código limpio y refactorizado constantemente, arquitectura modular. * Impacto: Aceleró el desarrollo y facilitó el mantenimiento del código   **3. Retroalimentación (Feedback)**   * Implementación: Ciclos cortos de desarrollo (2-3 semanas) con entregas funcionales * Evidencia: Validación académica semanal y testing interno del equipo * Impacto: se aseguró que el producto final cumpliera las expectativas reales del cliente   **4. Coraje (Courage)**   * Implementación: Decisiones técnicas arriesgadas como migrar de Firebase a Supabase. * Evidencia: Refactorización major del backend sin romper funcionalidades * Impacto: Mejoró significativamente la robustez y escalabilidad del sistema   **Prácticas XP Implementadas**  **1. Planificación del Juego (Planning Game)**   * Descripción: Proceso colaborativo donde se definen prioridades y el equipo técnico estima esfuerzos * Implementación: Sesiones de planificación al inicio de cada ciclo de desarrollo con seguimiento académico * Herramientas: GitHub Projects para tracking, reuniones semanales con profesora de portafolio * Resultados: Desarrollo enfocado en valor alineado con los requerimientos documentados   2**. Pequeñas Entregas (Small Releases)**   * Descripción: Entrega frecuente de software funcional en períodos cortos * Implementación: Desarrollo iterativo con versiones funcionales internas cada 2-3 semanas * Evidencia: MVP funcional completado para esta primera entrega académica * Beneficios: Validación temprana y continua con el cliente real   **3. Programación en Pares (Pair Programming)**   * Descripción: Dos desarrolladores trabajando en el mismo código simultáneamente * Implementación: Sesiones de 2 a 4 horas para módulos críticos (autenticación, PDF generation, sistema de incidencias) * Rotación: Alternancia entre "driver" (quien escribe) y "navigator" (quien revisa) * Impacto: Reducción de bugs, mejora en calidad del código, transferencia de conocimiento   **4. Refactoring Continuo**   * Descripción: Mejora constante de la estructura del código sin alterar funcionalidad * Implementación: Refactoring en cada sesión de desarrollo * Ejemplos:   + Consolidación de esquema de base de datos   + Modularización del backend en controllers, models, routes, services   + Optimización de componentes React para reutilización   **5. Integración Continua (Continuous Integration)**   * Descripción: Integración frecuente de cambios en el repositorio principal * Implementación: Commits múltiples diarios con mensajes descriptivos * Proceso:   + Desarrollo local con testing manual   + Commit con mensaje descriptivo   + Push a rama principal   + Validación de integración * Herramientas: Git, GitHub para control de versiones y colaboración   **Herramientas de Apoyo a la Metodología**   * **GitHub:** Control de versiones, issues, projects para planificación * **Discord/Teams:** Comunicación diaria y reuniones con cliente * **Postman:** Testing de APIs durante desarrollo * **Chrome DevTools:** Debugging de frontend   **Métricas de Efectividad XP**   * **Commits promedio:** 2-4 por día por desarrollador * **Sesiones de pair programming:** Mínimo 4 veces a la semana * **Refactorings mayores:** 2 refactorings significativos, incluyendo base de datos y arquitectura modular. * **Cambios de requerimiento:** 3 cambios mayores incorporados exitosamente, como: Implementación de fechas críticas para garantías, admin modo técnico, mapa con visualización de direccion.   Como parte de la implementación de XP, se utilizó el artefacto de Historias de Usuario para definir las funcionalidades del sistema desde la perspectiva del usuario final, incluyendo criterios de aceptación claros para cada historia (Godaddy, 2025). El documento completo de historias de usuario se encuentra disponible en el siguiente enlace: [Historias de usuario](https://docs.google.com/spreadsheets/d/1f7pe4szQZmFj33VzGVw6hoNgfKHcgNthAincODd7Dgc/edit?usp=sharing) |
| Evidencias de avance | Como evidencia del progreso alcanzado durante esta fase de desarrollo, se presentan capturas de pantalla de las principales vistas funcionales del sistema, correspondientes a los distintos tipos de usuarios: Administrador, Técnico y Beneficiario). Luego un video demostrando el funcionamiento actual del sistema.  Log In: log in con redirección por rol a su Home correspondiente    Crear cuenta: Crear cuenta solo está pensado para beneficiarios, ya que el administrador será el encargado de crear las cuentas de los técnicos. El sistema incorpora prácticas de seguridad estándar: las contraseñas se almacenan únicamente como hashes seguros usando bcrypt (coste configurable mediante la variable de entorno BCRYPT\_SALT\_ROUNDS), la autenticación se realiza con tokens JWT con caducidad y el backend aplica control de acceso por roles para proteger las rutas. Además, se validan entradas críticas en el servidor (por ejemplo formato de RUT y fortaleza de contraseñas) y existe un flujo de recuperación de contraseñas mediante códigos temporales enviados por correo con caducidad. Las claves y secretos (JWT secret, credenciales de Supabase, etc.) Se gestionan mediante variables de entorno para evitar su exposición en el repositorio, y el almacenamiento de archivos se realiza en Supabase Storage para separar datos y reducir riesgos.    Páginas pertenecientes a los demás roles (Técnico y Beneficiario), muestran su respectiva interfaz y funcionalidades asignadas tales como creación de reportes e incidencias, trazabilidad de cada proyecto de vivienda y comunicación entre los dos tipos de usuario. A continuación se especifica a mayor detalle los avances en los módulos para cada uno de los roles: Rol: Beneficiario  * **Inicio de sesión y acceso al Home:** El beneficiario puede iniciar sesión con su correo y contraseña, accediendo a su panel principal. Desde allí, puede visualizar la información de su vivienda asignada, incluyendo dirección, estado, metros cuadrados y número de habitaciones. En caso de no tener una vivienda asignada, el sistema muestra un mensaje informativo. * **Formulario de entrega y postventa:** Se habilita un formulario de checklist según el tipo de vivienda, donde el beneficiario puede registrar observaciones, adjuntar fotografías y enviar su evaluación del proceso de entrega o postventa. Este formulario solo se le despliega al beneficiario cuando su casa se encuentra en estado entregada, no antes.      * **Reporte de incidencias:** Permite crear incidencias indicando título, descripción, fotografías y ubicación georreferenciada. Una vez enviada, la incidencia queda registrada en el historial personal del usuario.      * **Historial de incidencias y trazabilidad:** El beneficiario puede revisar todas sus incidencias con sus respectivos estados y comentarios. Al ingresar a una incidencia específica, puede visualizar su evolución, registros de cambios y observaciones del técnico.      * **Evaluación y conformidad:** Tras la resolución de una incidencia, el beneficiario tiene la opción de confirmar la conformidad del cierre o realizar una evaluación final, cambiando el estado de la incidencia a “cerrada” o “confirmada”.      Rol: Técnico  * **Inicio de sesión y dashboard:** El técnico accede mediante login y visualiza un tablero con el resumen de asignaciones, incidencias pendientes y casos resueltos. * **Recepción de formularios y notificaciones:** El sistema notifica automáticamente al técnico sobre nuevos reportes o formularios enviados por beneficiarios, con acceso a los detalles y fotografías adjuntas.      * **Gestión de incidencias:** El técnico puede abrir incidencias, asignarlas, agregar comentarios y actualizar su estado (pendiente, en curso, resuelta). Todos los cambios quedan registrados en el historial del caso.        * **Generación de reportes en PDF:** Desde cada incidencia o formulario, el técnico puede generar un documento en formato PDF (bajo lineamientos DS49) con el resumen de la revisión o intervención técnica.      * **Historial y trazabilidad:** Se dispone de un registro completo de todas las acciones realizadas en una incidencia, incluyendo comentarios, cambios de estado y archivos adjuntos. * **Solicitud de conformidad:** Una vez marcada la incidencia como resuelta, el sistema notifica al beneficiario para que confirme la conformidad antes del cierre definitivo.  Rol: Administrador  * **Inicio de sesión y panel administrativo:** El administrador accede al panel de gestión general del sistema, con menús específicos para la administración de usuarios, proyectos y viviendas. * **Gestión de usuarios:** Permite crear, editar, listar o desactivar técnicos y otros administradores mediante operaciones CRUD básicas.      * **Gestión de proyectos y viviendas:** Posibilita crear o editar proyectos habitacionales y sus viviendas asociadas, registrando atributos como superficie y cantidad de habitaciones.        * **Plantillas de postventa:** Se pueden crear y asociar plantillas de postventa según el tipo de vivienda. Estas plantillas generan automáticamente los formularios que completan los beneficiarios al momento de la entrega.      * **Visualización y exportación de información:** El administrador puede consultar listados generales y descargar reportes o documentos en formato PDF generados desde el sistema. * **Auditoría básica:** Incluye acceso a historiales de actividad, incidencias y cambios de estado realizados por técnicos y beneficiarios. * **Funciones de panel Técnico:**   El administrador cuenta con las funciones necesarias para gestionar incidencias y formularios postventa    Se adjunta link con video demostrando el funcionamiento actual del sistema: [2025-10-15 22-30-44.mkv](https://drive.google.com/file/d/1KagYsl3O57Xwy3NbH5UtDNzaltQy0WHg/view?usp=drive_link) (Video de demostración, 2025) |

| **2. Monitoreo del Plan de Trabajo** |
| --- |
| Durante esta etapa se cumplió con las actividades planificadas dentro del cronograma general, con ajustes menores derivados del orden de integración de componentes. |

**Plan de Trabajo — Fase 2: Desarrollo del Proyecto**

## **Competencia: Realizar pruebas de certificación tanto de los productos como de los procesos utilizando buenas prácticas definidas por la industria**

| Actividad | Recursos | Duración de la actividad | Responsable | Observaciones | Estado de avance | Ajustes |
| --- | --- | --- | --- | --- | --- | --- |
| Pruebas iniciales de levantamiento del proyecto (frontend y backend) | PC, Node.js, VS Code, GitHub, Supabase | 8 horas | Brandon Weisser | Sistema verificado con ejecución local estable; configuración de entorno validada. | Completada | Ajustes wha  menores de dependencias locales. |
| Validación de conexión Supabase y autenticación de usuarios | Supabase, Postman, navegador(React, s.f.; Node.js, s.f.; Supabase, s.f.) | 10 horas | Denihann Maturana | Pruebas de creación e inicio de sesión de usuarios. | Completada | Actualización de claves de entorno `.env`. |
| Pruebas funcionales del flujo de navegación y protección de rutas | React Router DOM, AuthContext, Supabase | 10 horas | Damian Almazabal | Rutas protegidas operativas; roles identificados correctamente. | Completada | Se añadió control de carga para prevenir flicker visual. |
| Pruebas del módulo de incidencias (listado y visualización) | React, Supabase(React, s.f.; Node.js, s.f.; Supabase, s.f.) | 8 horas | Brandon Weisser | Módulo desplegado con datos de prueba. | En curso | Pendiente de validación con base de datos final. |

## **Competencia: Gestionar proyectos informáticos, ofreciendo alternativas para la toma de decisiones de acuerdo a los requerimientos de la organización**

| Actividad | Recursos | Duración de la actividad | Responsable | Observaciones | Estado de avance | Ajustes |
| --- | --- | --- | --- | --- | --- | --- |
| Coordinación inicial con equipo para definir arquitectura del sistema (frontend–backend) | Reuniones en línea, GitHub, Google Docs | 6 horas | Todos los integrantes del equipo | Se acordó estructura modular y tecnologías base (React, Node, Supabase). | Completada | Decisión de usar Supabase por flexibilidad cloud/on-premise. |
| Distribución de responsabilidades por tipo de usuario | Reuniones, plan de trabajo | 4 horas | Todos los integrantes del equipo | Damian: módulo administrador; otros: técnico, beneficiario y base de datos. | Completada | Se mantuvo la división tras revisión del avance. |
| Revisión de cronograma y planificación de tareas | Cronograma grupal, Trello/Excel | 6 horas | Todos los integrantes del equipo | Se ajustaron tiempos de entrega según progresos reales. | En curso | Reorganización de tareas CRUD. |
| Coordinación de integración entre backend y frontend | Node.js, React, GitHub(React, s.f.; Node.js, s.f.; Supabase, s.f.) | 10 horas | Todos los integrantes del equipo | Definición de endpoints comunes y manejo de rutas. | En curso | Pendiente de pruebas con datos reales. |

## **Competencia: Construir modelos de datos para soportar los requerimientos de la organización de acuerdo a un diseño definido y escalable en el tiempo**

| Actividad | Recursos | Duración de la actividad | Responsable | Observaciones | Estado de avance | Ajustes |
| --- | --- | --- | --- | --- | --- | --- |
| Análisis del modelo de base de datos entregado por TECHO Chile | Supabase, Diagrama E–R | 6 horas | Denihann Maturana | Revisión de tablas y relaciones (usuarios, viviendas, proyectos). | Completada | Documentación agregada al repositorio. |
| Ajuste de relaciones y tipos de datos para compatibilidad con Supabase | Supabase, SQL Editor | 8 horas | Integrante responsable de base de datos | Tipos normalizados (uuid, text, integer). | Completada | Campos ajustados a las vistas requeridas por el frontend. |
| Definición de estructura CRUD para usuarios y viviendas | Node.js, Express, Supabase(React, s.f.; Node.js, s.f.; Supabase, s.f.) | 10 horas | Brandon Weisser | Endpoints y controladores base definidos. | En curso | Se agregó validación y manejo de errores. |
| Implementación de modelo de datos para incidencias | Supabase, Node.js(React, s.f.; Node.js, s.f.; Supabase, s.f.) | 6 horas | Denihann maturana | Diseño preliminar del modelo y dependencias. | Pendiente | Se espera revisión de requerimientos del cliente. |

## **Competencia: Desarrollar una solución de software utilizando técnicas que permitan sistematizar el proceso de desarrollo y mantenimiento**

| Actividad | Recursos | Duración de la actividad | Responsable | Observaciones | Estado de avance | Ajustes |
| --- | --- | --- | --- | --- | --- | --- |
| Implementación del Home centralizado y dinámico por rol | React, AuthContext, Tailwind CSS | 12 horas | Damian Almazabal | Home.jsx funcional, detecta rol y redirige al dashboard correspondiente. | Completada | Se añadió estado de carga visual. |
| Creación del layout de administrador (panel y navegación) | React, Tailwind CSS, Router DOM | 10 horas | Damian Almazabal | Layout funcional con componentes reutilizables. | Completada | Reestructuración en carpetas layouts/ y pages/. |
| Implementación y creación de modulo dashboard para técnico y beneficiario con sus funciones | React, [node.js](http://node.js), Supabase | 140 horas | Brandon Weisser y Denihann Maturana | Estructura de componentes creada, con funciones de prioridad alta desarrollada. | En curso | Se incorporarán funciones con menor prioridad. |
| Desarrollo de CRUD para usuarios y viviendas (Administrador) | Node.js, Express, Supabase | 14 horas | Damian Almazabal | Endpoints listos, conexión con frontend pendiente. | En curso | Se ajusta flujo de validación de datos. |
| Integración de frontend y backend | React, Node.js, Fetch API | 10 horas | Todos los integrantes del equipo | Pruebas de conexión y renderizado dinámico. | Pendiente | Falta conexión con base final. |

[1] En caso de que el Proyecto APT sea grupal, en esta columna deben indicar el nombre de los responsables de cada tarea o actividad. Esto posteriormente permitirá diferenciar la evaluación por cada integrante.

| **3. Ajustes a partir del monitoreo** |
| --- |
| Durante esta fase se realizaron algunos ajustes en la planificación inicial, principalmente relacionados con la organización del código y la definición de responsabilidades por rol de usuario. |

| Factores que han facilitado y/o dificultado el desarrollo de mi plan de trabajo:  Los principales factores que facilitaron el desarrollo fueron:   * Una arquitectura modular clara, basada en componentes reutilizables en React. * La integración con Supabase, que simplificó el manejo de usuarios y autenticación. * El trabajo colaborativo mediante GitHub, que permitió un control de versiones ordenado.   Entre los factores que dificultaron el avance se destacan:   * Cambios recientes en la estructura de la base de datos y ajustes no documentados. * Curva de aprendizaje del equipo con nuevas tecnologías (Supabase, React Router, integración de roles). * Retrasos menores por incorporación tardía de algunos integrantes al desarrollo.   Como resultado, el sistema ya dispone de una base sólida sobre la cual se continuará construyendo en la siguiente fase, enfocándose en la consolidación de los CRUD de los tipos de usuarios y la implementación completa de las funcionalidades operativas para los demás tipos de usuario. |
| --- |

| Actividades ajustadas o eliminadas:  Durante la fase actual se decidió ajustar la implementación de los CRUD: originalmente se contemplaba conectar el frontend directamente con Supabase, pero se optó por crear endpoints dedicados en el backend (Node.js + Express) para mejorar la seguridad y escalabilidad del sistema.  También se reestructuró la organización de carpetas en el frontend, separando layouts, pages y components, con el fin de mejorar la legibilidad y facilitar la ampliación futura.*.* |
| --- |

| Actividades que no has iniciado o están retrasadas:  Las actividades que presentan retraso o aún no han sido iniciadas corresponden principalmente a:   * Las pruebas unitarias e integración entre componentes, que se realizan una vez finalizada la lógica CRUD. * El módulo de despliegue y pruebas en entorno de producción.   Estos retrasos se deben a la priorización de tareas de infraestructura y autenticación, que eran necesarias para asegurar la estabilidad del sistema antes de avanzar en las operaciones CRUD. |
| --- |

### **Referencias**

* Ministerio de Vivienda y Urbanismo (MINVU). (2019). *Decreto Supremo N°49: Reglamento del Programa Fondo Solidario de Elección de Vivienda*. Gobierno de Chile. [https://www.minvu.gob.cl](https://www.minvu.gob.cl?utm_source=chatgpt.com)
* TECHO-Chile. (s.f.). *Quiénes somos*. Recuperado el 27 de Agosto 2025 de <https://www.techo.org/chile>
* Godaddy. (2025). *Metodología XP (Extreme Programming): Qué es y cómo optimizar el desarrollo ágil con prácticas efectivas*. Recuperado de <https://www.godaddy.com/resources/latam/tecnologia/metodologia-xp-extreme-programming-que-es>
* React. (s.f.). *React – A JavaScript library for building user interfaces*. https://reactjs.org
* Node.js. (s.f.). *Node.js*. https://nodejs.org
* Supabase. (s.f.). *Supabase: The open source Firebase alternative*.<https://supabase.com>
* Video de demostración. (2025, 15 de octubre). *Funcionamiento actual del sistema – Proyecto APT*. Archivo de video interno.