



UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

DEPARTAMENTO DE ELECTROTECNIA E INFORMÁTICA INGENIERÍA EN INFORMÁTICA

Gestión de proyecto para sistema de comunicación efectiva ante catástrofes - Alertify

Alumna: Francisca Isidora Capelli Zúñiga

Correo: francisca.capelli@usm.cl

Profesor Guía: Ricardo Cahe

Resumen: En febrero de 2024, un devastador mega incendio en Valparaíso afectó más de 8,000 hectáreas y destruyó 15,000 viviendas, dejando varios fallecidos y damnificados. El desafío de mi equipo es mejorar la comunicación de los ciudadanos durante catástrofes. En el reciente mega incendio, la comunidad no sabía cómo resguardarse y las vías de evacuación estaban obstruidas, lo que impidió que las instituciones de asistencia llegaran a tiempo. En esta tesina se presenta como se gestionará este proyecto bajo la metodología Scrum.

Conceptos clave: Gestión de proyecto, Metodología Scrum.



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: Gestión de proyecto para sistema de comunicación efectiva ante catástrofes - Alertify

Nombre del candidato(a): Francisca Isidora Capelli Zúñiga

Carrera / Grado: Ingeniería en informática

Campus: Viña del Mar ; **Departamento:** ELINF

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Ricardo Cahe Cabach, 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

☒ 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:

☐ 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: 22-07-2025

; Firma:

Estudiante o Candidato(a):

Fecha: 22-07-2025

; Firma:

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.

Índice

1.	Introducción	3
1.1	<i>Planteamiento del problema</i>	3
1.2	<i>Objetivo general</i>	3
1.3	<i>Objetivos específicos</i>	3
1.4	<i>Relevancia de las metodologías ágiles en proyectos de software.</i>	4
2.	Marco Teórico	5
2.1	<i>Principios y valores del Manifiesto Ágil.</i>	5
2.2	<i>Descripción de metodologías ágiles.</i>	6
3.	Metodología y Plan de trabajo	7
3.1	<i>Enfoque para la implementación de metodologías ágiles en el proyecto.</i>	7
3.2	<i>Herramientas de gestión empleadas.</i>	7
4.	Planificación de Sprints	7
4.1	<i>Definición de sprints y ciclos de desarrollo</i>	7
4.2	<i>Estimación de tareas y asignación de recursos.</i>	8
4.3	<i>Historias de usuario</i>	8
4.4	<i>Desarrollo real de proyecto- Carta Gantt</i>	10
4.5	<i>Requerimientos de la aplicación</i>	12
5.	Ejecución de Reuniones Ágiles	13
5.1	<i>Descripción de reuniones diarias, de planificación, revisiones y retrospectivas.</i>	13
5.2	<i>Importancia de la comunicación y colaboración en el equipo.</i>	13
6.	Gestión de Cambios y Mejoras Continuas	14
7.	Evaluación del Desempeño Ágil	16
	<i>Resultados de Sprints</i>	16
	<i>Métricas de eficacia y análisis de resultados obtenidos</i>	20
8.	Desafíos y Soluciones	21
	<i>Obstáculos identificados</i>	21
	<i>Estrategias para superarlos</i>	21
10.	Conclusiones	22
	Referencias	24
	Bibliografía	24

Tabla de ilustraciones

Ilustración 1- Carta Gantt con Historias de Usuarios "Alerta discriminatoria" y "Detección de distancia"	10
Ilustración 2- Carta Gantt con Historias de Usuarios "Cálculo de vía de escape" y "Accesibilidad de la alerta"	10
Ilustración 3- Carta Gantt con Historias de Usuarios "Personalización del sonido de la alerta " e "Informativo ante catástrofes"	11
Ilustración 4 - Maqueta, Vista de alerta en dispositivo móvil	12
Ilustración 5 - Maqueta de notificación tipo alerta	12
Ilustración 6 - Vista de inicio	16
Ilustración 7 - Vista de sección Informativo ante catástrofes	16
Ilustración 8 - Vista de sección Personalización de la alerta	17
Ilustración 9 - Vista de cargador de archivos funcional	17
Ilustración 10 - Vista de nuevo inicio	18
Ilustración 11 - Vista de nueva sección Historial de desastres	18
Ilustración 12 - Maqueta inicial de sección Informativo ante catástrofes	19
Ilustración 13 - Interfaz final de sección Informativo ante catástrofes	19
Ilustración 14 - Vista de informativo de Terremoto	19
Ilustración 15 - Vista de informativo de Incendio	19

Tabla de Tablas

Tabla 1 - Distribución de Historias de usuario por Sprints	9
Tabla 2 - Simbología de Carta Gantt en Microsoft Project	11
Tabla 3 - Gestión de cambios	14
Tabla 4 - Gestión de Riesgos	15
Tabla 5 - Estimación de costos	21

1. Introducción

1.1 Planteamiento del problema

En febrero del 2024, en la región de Valparaíso sucedió uno de los más catastróficos *“incendios que se han registrado en la historia de la región, más de 8 mil hectáreas entre las ciudades de Valparaíso, Viña del Mar y Quilpué fueron afectadas por uno de los mega incendios más catastróficos registrados en nuestro país dejando, en dos días, un considerable saldo de fallecidos y damnificados, consumiendo además alrededor de 15 mil viviendas (1)”*.

Bajo este contexto, a inicio de este año lectivo, para el proyecto de software se presentaron diferentes empresas con diferentes desafíos, con un enfoque común de mejorar la calidad de vida en la quinta región, haciendo hincapié en cómo podemos ayudarnos mutuamente en comunidad.

El desafío que tuvo que afrontar mi equipo de trabajo es acerca de mejorar la comunicación de los ciudadanos en el momento en que ocurra una catástrofe, ya que al momento del mega incendio uno de los mayores problemas que se presentaron fue que la comunidad afectada no sabía cómo poder resguardarse y no había vías de evacuación realmente despejadas, que hizo que no llegaran las instituciones de asistencia a tiempo.

1.2 Objetivo general

Implementar un enfoque de gestión del proyecto informático basado en Scrum buscando maximizar la agilidad del equipo y la alineación con las necesidades del cliente para el proyecto *Alertify*, sistema de comunicación efectiva ante catástrofes.

1.3 Objetivos específicos

- I. Identificar y analizar las principales funciones y responsabilidades del Scrum Master y del Product Owner en el contexto del proyecto Alertify, y cómo su colaboración impacta en el éxito del proyecto.
- II. Diseñar un conjunto de prácticas y herramientas que faciliten la comunicación y la gestión de expectativas entre el Scrum Master, el Product Owner y el equipo de desarrollo, con el fin de mejorar la efectividad del proceso Scrum.
- III. Evaluar el impacto del marco propuesto en la calidad de los entregables y la satisfacción del cliente a través de métricas de rendimiento, antes y después de su implementación.

1.4 Relevancia de las metodologías ágiles en proyectos de software.

Las metodologías ágiles son fundamentales en el desarrollo de proyectos de software por varias razones:

- **Flexibilidad y Adaptación:** Permiten adaptarse rápidamente a los cambios en los requisitos. Esto es crucial en un entorno tecnológico en constante evolución, donde las necesidades del cliente pueden cambiar.
- **Colaboración Continua:** Fomentan la comunicación constante entre equipos de desarrollo y partes interesadas. Esto asegura que todos estén alineados y que el producto final cumpla con las expectativas.
- **Entrega Incremental:** Promueven la entrega de incrementos funcionales del software en ciclos cortos (Sprint), lo que permite a los clientes recibir valor desde etapas tempranas del proyecto y proporcionar feedback que mejora el producto.
- **Mejora Continua:** Las retrospectivas al final de cada ciclo permiten al equipo reflexionar sobre su proceso y hacer ajustes, lo que lleva a un aprendizaje constante y mejora en la calidad del trabajo.
- **Enfoque en el Usuario:** Las metodologías ágiles suelen centrarse en la satisfacción del cliente, priorizando las características que más aportan valor al usuario final.
- **Reducción de Riesgos:** Al permitir pruebas y ajustes frecuentes, se identifican y mitigan riesgos desde etapas tempranas del proyecto, lo que mejora la probabilidad de éxito.
- **Motivación del Equipo:** Fomentan un ambiente de trabajo colaborativo y empoderan a los miembros del equipo, lo que puede resultar en mayor motivación y productividad.

Las metodologías ágiles no solo mejoran la eficiencia y efectividad en el desarrollo de software, sino que también aseguran una mejor alineación con las necesidades del cliente, lo que resulta en productos de mayor calidad y relevancia en el mercado.

2. Marco Teórico

2.1 Principios y valores del Manifiesto Ágil.

El Manifiesto Ágil, creado en 2001 por un grupo de desarrolladores de software, establece cuatro valores y doce principios fundamentales que guían las prácticas ágiles.

Valores del Manifiesto Ágil

- **Individuos e interacciones sobre procesos y herramientas:** Se prioriza la comunicación y colaboración entre los miembros del equipo en lugar de seguir estrictamente procesos o depender de herramientas.
- **Software funcionando sobre documentación extensiva:** El enfoque está en entregar software que funcione, en lugar de crear una documentación exhaustiva que no contribuya directamente al valor del producto.
- **Colaboración con el cliente sobre negociación de contratos:** Se fomenta una relación colaborativa con el cliente, buscando su participación activa y ajustando los requisitos según sus necesidades cambiantes.
- **Respuesta ante el cambio sobre seguir un plan:** Se valora la capacidad de adaptarse a cambios en los requisitos, incluso en etapas avanzadas del desarrollo, en lugar de seguir ciegamente un plan predeterminado.

Principios del Manifiesto Ágil

- **Satisfacción del cliente:** La máxima prioridad es satisfacer al cliente mediante la entrega temprana y continua de software valioso.
- **Cambio bienvenido:** Se aceptan los cambios en los requisitos, incluso en etapas tardías del desarrollo.
- **Entrega frecuente:** Se entregan versiones funcionales del software de manera frecuente, desde unas pocas semanas hasta unos pocos meses.
- **Colaboración diaria:** Los desarrolladores y los usuarios deben trabajar juntos de manera diaria durante todo el proyecto.
- **Proyectos construidos en torno a individuos motivados:** Se debe confiar en los miembros del equipo y proporcionarles el entorno y el apoyo que necesitan.
- **Comunicación cara a cara:** La forma más eficiente y efectiva de comunicar información es a través de la conversación cara a cara.
- **Software funcionando como medida principal de progreso:** Se mide el progreso principalmente por la entrega de software que funcione.
- **Sostenibilidad:** Los desarrolladores deben trabajar de manera sostenible, manteniendo un ritmo constante y evitando el agotamiento.
- **Excelencia técnica y buen diseño:** Se fomenta la atención continua a la calidad técnica y al buen diseño.
- **Simplicidad:** La simplicidad es esencial; se debe maximizar la cantidad de trabajo no realizado.
- **Autoorganización:** Los equipos autoorganizados producen los mejores resultados.
- **Reflexión y ajuste:** En intervalos regulares, el equipo reflexiona sobre cómo convertirse en más efectivo y ajusta su comportamiento en consecuencia.

Con estos valores y principios es posible ayudar al equipo *Alertify* a adaptarse y a enfocarse en lo que realmente importa en el desarrollo de software.

2.2 Descripción de metodologías ágiles.

I. Scrum

Scrum es un marco de trabajo ágil que se centra en la gestión y entrega de proyectos a través de iteraciones cortas llamadas "Sprints", que suelen durar de dos a cuatro semanas.

Roles:

- Scrum Master: Facilita el proceso y elimina obstáculos.
- Product Owner: Define y prioriza el backlog del producto.
- Equipo de Desarrollo: Realiza el trabajo de desarrollo.

Eventos:

- Sprint: Ciclo de trabajo.
- Planificación del Sprint: Se establecen las tareas a realizar.
- Daily Scrum: Reunión diaria para coordinar actividades.
- Revisión del Sprint: Presentación de lo completado.
- Retrospectiva del Sprint: Reflexión sobre el proceso para mejorar.

Artefactos:

- Backlog del Producto: Lista priorizada de características.
- Backlog del Sprint: Tareas a completar en el sprint.
- Incremento: Resultado del trabajo completado.

II. Kanban

Kanban es un método visual para gestionar el trabajo a medida que avanza a través de un proceso. Se basa en la visualización del flujo de trabajo a través de un formato tipo tablero y la limitación del trabajo en curso (WIP).

Elementos Clave:

- Tablero Kanban: Visualiza las tareas en diferentes etapas (por ejemplo, "Por Hacer", "En Progreso", "Hecho").
- Tarjetas: Representan tareas individuales.
- Límites WIP: Restricciones sobre cuántas tareas pueden estar en progreso al mismo tiempo.

III. Extreme Programming (XP)

XP es una metodología ágil centrada en la mejora de la calidad del software y en la capacidad de adaptarse a los cambios. Se basa en prácticas de desarrollo de software específicas.

Prácticas Clave:

- Programación en Parejas: Dos desarrolladores trabajan juntos en una misma tarea.
- Desarrollo Guiado por Pruebas (TDD): Se escriben pruebas antes de desarrollar el código.
- Integración Continua: El código se integra y se prueba frecuentemente.
- Refactorización: Mejora continua del código existente.

IV. Lean Software Development

Inspirada en los principios del lean manufacturing, esta metodología se centra en la eliminación de desperdicios, la mejora continua y la maximización del valor para el cliente.

Principios Clave:

- Eliminar desperdicios: Cualquier actividad que no agrega valor se debe eliminar.
- Amplificar el aprendizaje: Fomentar la experimentación y el aprendizaje continuo.
- Decidir tarde: Mantener la flexibilidad y la capacidad de respuesta hasta el último momento.
- Entregar lo más rápido posible: Acelerar la entrega de valor al cliente.

3. Metodología y Plan de trabajo

3.1 Enfoque para la implementación de metodologías ágiles en el proyecto.

En este proyecto se utilizará la metodología de trabajo Scrum debido a la flexibilidad que aportará al momento de desarrollar el presente. Los artefactos scrum aportarán transparencia y confianza tanto en el equipo como con el cliente visibilizando el estado actual del proyecto y el avance de éste.

3.2 Herramientas de gestión empleadas.

Las herramientas utilizadas para el monitoreo y comunicación con el equipo son Taiga y Discord.

Se ha decidido el uso de Taiga ya que tienen herramientas de visualización de diferentes artefactos como el backlog, tareas y un indicador de burndown del equipo, con esto se podrá tomar mejores decisiones con respecto a los avances estipulados en cada sprint.

Discord es una aplicación de comunicación remota donde se pueden crear espacios para diferentes propósitos, en el caso del proyecto, se utiliza para hacer los eventos Scrum y dejar documentación que es necesaria para llevar a cabo el proyecto.

4. Planificación de Sprints

4.1 Definición de Sprints y ciclos de desarrollo.

La definición de Sprints están basados en los hitos que se realizarán en el ramo Proyecto de software siendo éstos 3:

- Sprint 1 - 11 de Septiembre
Este ciclo de trabajo tiene por objetivo poner a prueba todo el trabajo teórico que se hizo con respecto al proyecto Alertify.
- Sprint 2 - 18 de Octubre.
Este ciclo de trabajo tiene por objetivo demostrar el avance funcional del proyecto, idealmente mostrando a las partes interesadas alguna funcionalidad que sea interactuable con un usuario.
- Sprint 3 - 15 de Noviembre (Final).
Este último ciclo tiene como finalidad demostrar un producto completamente funcional e interactuable con usuarios.

4.2 Estimación de tareas y asignación de recursos.

Bajo la metodología Scrum se han definido 6 funcionalidades (historias de usuario), las cuales las primeras 3 son primordiales para que el proyecto cumpla sus objetivos.

La definición de los puntajes de dificultad está definida por la complejidad de las tareas que componen cada historia de usuario a través de 4 pilares de evaluación (experiencia de usuario, diseño, frontend y backend), las cuales siguen la secuencia Fibonacci, tal como lo establece la metodología Scrum. Estos puntajes de señalarán a un costado del nombre de las historias de usuario bajo el siguiente formato: (1).

La prioridad de las historias de usuarios está definida por el valor que aportan al proyecto, por lo que el orden en el que se tienen que cumplir es independiente al puntaje de dificultad que se definió. A continuación, la prioridad de las historias de usuario está definida por una numeración basada en letras desde la **a** (siendo esta la historia de usuario con mayor prioridad) hasta la **f** (siendo la historia de usuario con menor prioridad).

4.3 Historias de usuario

a) Alerta discriminatoria (11)

Como usuario me gustaría recibir la alerta de la aplicación sólo si me encuentro cercano a una catástrofe para poder accionar consecuentemente al suceso.

- Horas de trabajo: 377 hrs.
- Tareas:
 - Crear interfaz de la alerta.
 - Crear existencia de diferentes tipos de catástrofes.
 - Integrar ubicación de catástrofe y de usuario.
 - Calcular distancia entre catástrofe y usuario.

b) Detección a distancia (1)

Como usuario me gustaría recibir la distancia entre una catástrofe y el lugar donde me ubico para saber si estoy a salvo.

- Horas de trabajo: 256 hrs.
- Tareas:
 - Recuperación de la distancia de la catástrofe y usuario.
 - Mostrar al usuario y catástrofe en un mapa.

c) Cálculo de vía de escape (13)

Como usuario me gustaría recibir una sugerencia de ruta de escape para actuar si estoy en peligro donde me ubico.

- Horas de trabajo: 344 hrs
- Tareas:
 - Cálculo de vía de evacuación.
 - Tipificar la alerta con respecto a la catástrofe.

d) Accesibilidad a alerta (9)

Como usuario con discapacidad visual me gustaría que la aplicación tuviese la opción de lectura en voz alta de la alerta para poder recibir la información de forma efectiva.

- Horas de trabajo: 0 hrs.
- Tareas:

- Implementar lectura de alerta en texto a voz.
- Testear usabilidad con gente con discapacidad visual.

e) Personalización de sonido de la alerta (8)

Como usuario me gustaría tener la posibilidad de poder cambiar el sonido de la alerta para poder diferenciarla de mis otros sonidos de mi celular.

- Horas de trabajo: 72 hrs.
- Tareas:
 - Crear interfaz.
 - Seleccionar sonidos demo.
 - Implementar seleccionador de tonos
 - Implementar cargador de archivos a la aplicación.
 - Reproducir tono al detectar una alerta.

f) Informativo ante catástrofes (5.5)

Como usuario me gustaría que haya en la aplicación una explicación de cada catástrofe que puede suceder y cómo actuar cuando suceda para poder estar informado con antelación de cómo actuar.

- Horas de trabajo: 35 hrs
- Tareas:
 - Investigar tipos de catástrofes.
 - Describir información importante para tipo de catástrofe.
 - Crear interfaz informativa.

Las historias de usuario *Alerta discriminativa*, *Detección de distancia* y *Cálculo de vía de escape* son fundamentales que se cumplan en primera instancia, ya que éstas son las que le agregan mayor valor al proyecto, que es darle la información necesaria a la ciudadanía para poder mantenerse a salvo en momentos de catástrofes.

Las historias de usuario *Accesibilidad de la alerta* y *Personalización del sonido de la alerta* están diseñadas para garantizar la inclusión de personas con discapacidad visual, reconociendo que todos tienen el derecho a recibir información clara y mantenerse seguros durante situaciones de catástrofes.

La historia de usuario *Informativo ante catástrofes* tiene como objetivo proporcionar al usuario información clara y accesible sobre la naturaleza de cada tipo de catástrofe, junto con recomendaciones generales sobre cómo actuar de manera adecuada en cada caso.

Se ha definido los objetivos a cumplir en cada sprint a través de las historias de usuario descritas anteriormente.

Sprint 1 – 11 de Septiembre	Sprint 2 – 18 de Octubre	Sprint 3 – 15 de Noviembre
Alerta discriminativa	Alerta discriminativa	Cálculo de vía de escape
Detección de distancia	Detección de distancia	Accesibilidad de la alerta
Cálculo de vía de escape	Cálculo de vía de escape	Informativo ante catástrofes
Personalización de sonido de la alerta		

Tabla 1 - Distribución de Historias de usuario por Sprints

4.4 Desarrollo real de proyecto- Carta Gantt

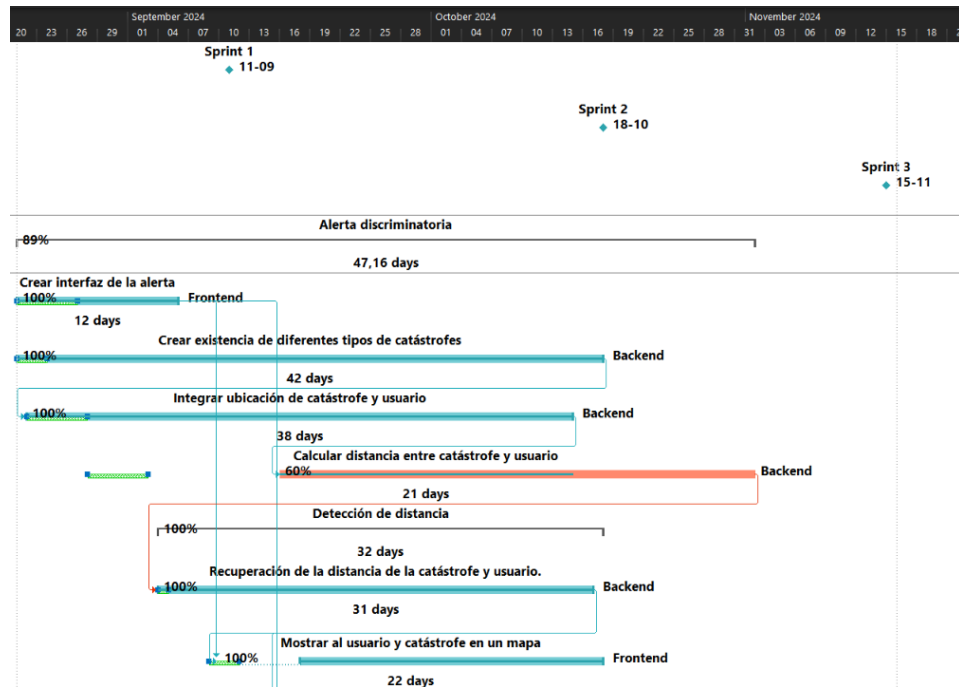


Ilustración 1- Carta Gantt con Historias de Usuarios "Alerta discriminatoria" y "Detección de distancia"

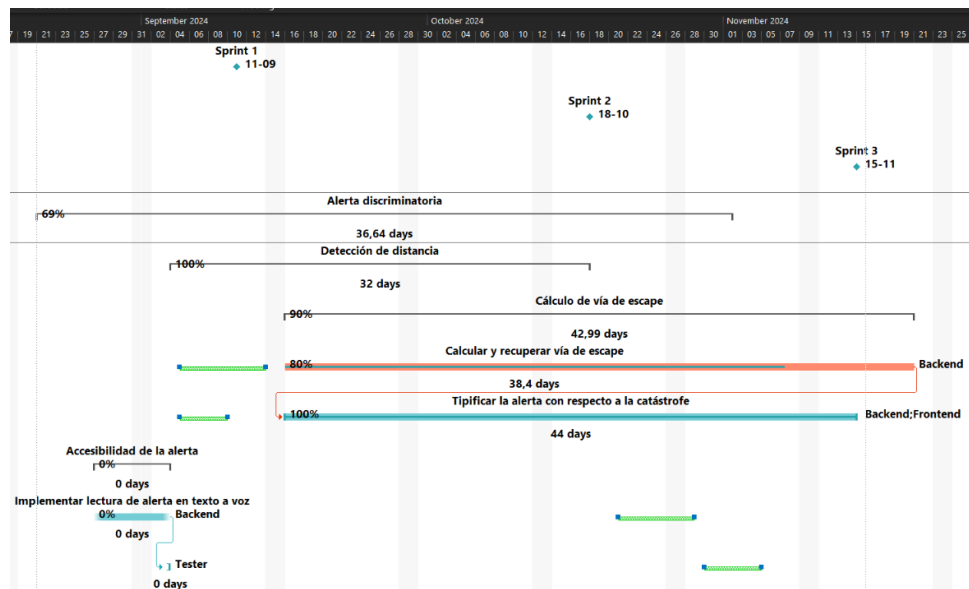


Ilustración 2- Carta Gantt con Historias de Usuarios "Cálculo de vía de escape" y "Accesibilidad de la alerta"

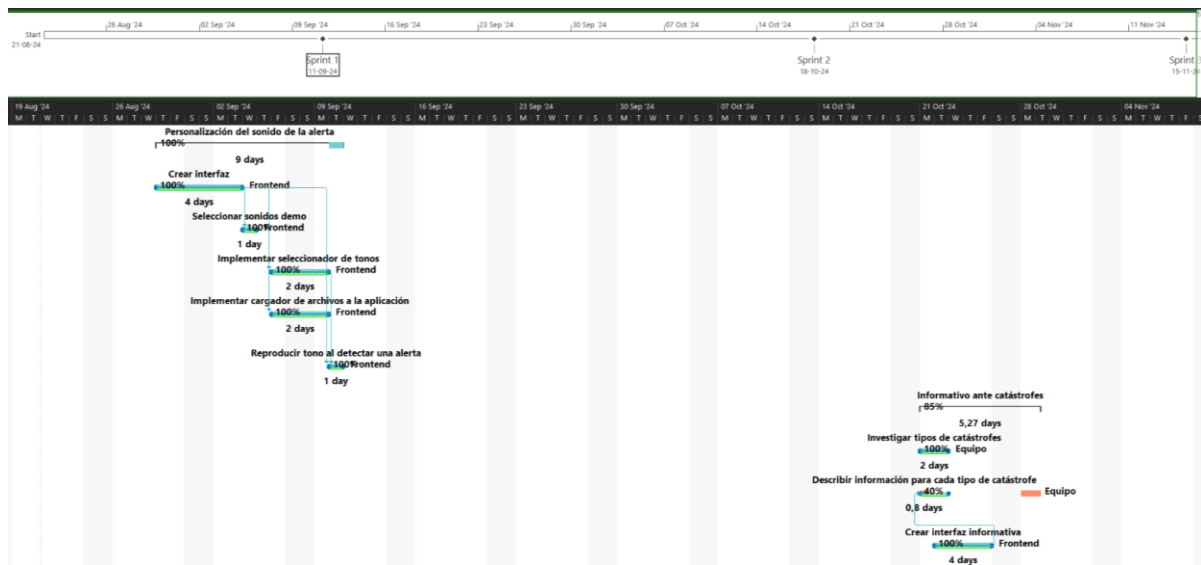


Ilustración 3- Carta Gantt con Historias de Usuarios "Personalización del sonido de la alerta " e "Informativo ante catástrofes"

Simbología de Carta Gantt	
Progreso	
Tarea	
Tarea de resumen	
Hito	
Sólo duración de tarea	
Base de referencia de duración	
Tarea atrasada	

Tabla 2 - Simbología de Carta Gantt en Microsoft Project

4.5 Requerimientos de la aplicación

Aparte de las historias de usuarios presentados anteriormente, el equipo acordó cumplir un producto mínimo viable que se presentaría para el hito Feria de software que sucedería el día 15 de Noviembre del 2024, que también está catalogado como el fin del sprint 3.

- Definición de alerta:

Se acordó que la alerta sería tipo notificación donde se detalle el tipo de catástrofe acompañado de un ícono, fecha, hora, el lugar de la catástrofe, la distancia de la catástrofe con respecto al usuario y vía de escape donde tiene que evacuar el usuario.



Ilustración 4 - Maqueta, Vista de alerta en dispositivo móvil



Ilustración 5 - Maqueta de notificación tipo alerta

Las catástrofes que tienen que estar presentes al menos para la historia de usuario Informativo ante catástrofes son terremoto, tsunami, incendio, inundación y erupción volcánica.

Estas decisiones se tomaron de manera conjunta durante el primer hito de Sprint Planning, considerando las exigencias del ramo Proyecto de Software. Se evaluó cuáles funcionalidades podían desarrollarse con los recursos disponibles del equipo para cumplir con los objetivos de los Sprints y del hito Feria de Software, y cuáles quedarían catalogadas como funcionalidades escalables del proyecto.

5. Ejecución de Reuniones Ágiles

5.1 Descripción de reuniones diarias, de planificación, revisiones y retrospectivas.

Bajo la metodología Scrum se recurren a diferentes ceremonias para poder asegurar el que el desarrollo de un proyecto sea satisfactorio.

Se ha decidido adaptar una de las ceremonias principales de la metodología Scrum, que es la ceremonia Daily scrum, cambiando la frecuencia de las reuniones a día por medio en vez de todos los días, con el motivo de reportar los avances y aclarar dudas de cada integrante.

También se decidió adoptar las ceremonias Sprint planning y Sprint review, siendo la primera reuniones que se realizan al inicio de cada Sprint donde todo el equipo se reúne para planificar las tareas a cumplir para finalizar las funcionalidades propuestas de manera satisfactoria, y la segunda reuniones que se realizan al final cada Sprint para mostrar el incremento de producto completado y obtener feedback de las partes interesadas.

5.2 Importancia de la comunicación y colaboración en el equipo.

La comunicación y la colaboración son fundamentales en un proyecto informático manejado con la metodología ágil Scrum.

- Adaptabilidad y Flexibilidad
 - Reacción Rápida: La comunicación constante permite al equipo adaptarse rápidamente a cambios en los requisitos o en el entorno del proyecto, lo que es esencial en el desarrollo ágil.
- Transparencia
 - Visibilidad del Progreso: Mantener a todos informados sobre el estado del proyecto fomenta la transparencia, lo que ayuda a identificar problemas de forma temprana y a tomar decisiones informadas.
- Trabajo en Equipo
 - Sinergia: La colaboración efectiva potencia las habilidades de cada miembro, lo que resulta en soluciones más innovadoras y eficientes. Los equipos multifuncionales pueden abordar desafíos desde diferentes perspectivas.
- Resolución de Conflictos
 - Manejo de Desacuerdos: Una buena comunicación ayuda a gestionar y resolver conflictos de manera constructiva, minimizando tensiones y manteniendo un ambiente de trabajo positivo.
- Mejora Continua
 - Retrospectivas: En Scrum, las reuniones retrospectivas permiten al equipo reflexionar sobre su desempeño, identificar áreas de mejora y ajustar procesos, fomentando una cultura de aprendizaje continuo.
- Claridad en Roles y Responsabilidades
 - Evitación de Ambigüedades: La comunicación clara asegura que todos entiendan sus roles y responsabilidades, lo que ayuda a evitar malentendidos y redundancias en el trabajo.
- Mejor Gestión del Tiempo
 - Planificación Efectiva: Las reuniones diarias y otras interacciones regulares permiten una mejor gestión del tiempo, asegurando que el equipo se mantenga enfocado y alineado en las prioridades.
- Feedback Constante
 - Mejoras en el Producto: La interacción frecuente con los stakeholders y entre los miembros del equipo permite recibir y aplicar feedback de manera continua, mejorando la calidad del producto final.

6. Gestión de Cambios y Mejoras Continuas

6.1 Adaptación a cambios en los requerimientos

El proyecto adoptará un enfoque flexible basado en los principios ágiles, permitiendo responder rápidamente a cambios en las prioridades del cliente o en los requerimientos técnicos. Esto incluye:

- Backlog dinámico: Revisión constante del backlog del producto para priorizar funcionalidades críticas.
- Feedback iterativo: Incorporar comentarios de los stakeholders al final de cada sprint para ajustar el alcance del proyecto.
- Gestión de riesgos: Identificar riesgos técnicos y organizativos tempranamente, ajustando las tareas y recursos según sea necesario.

6.2 Gestión de mejoras basadas en feedback

- Reuniones retrospectivas: En cada sprint, se analizarán las áreas de mejora en procesos, herramientas y comunicación del equipo.

6.3 Tabla de Gestión de Cambios

Cambio Solicitado	Motivo	Impacto Estimado	Acciones a Realizar
Cambiar tecnologías al realizar el objetivo de Cálculo de vía de escape	Facilitar el proceso de cumplimiento del objetivo.	Alto: Afecta el backend y los tiempos del sprint.	Realizar investigación y documentación de las posibles tecnologías a usar, ajustar objetivos por sprint y reasignar tareas.
Personalización de rutas de evacuación	Solicitud de stakeholders para priorizar rutas con accesibilidad universal.	Medio: Cambios en lógica del backend y pruebas de usuario.	Actualizar el diseño del algoritmo y realizar pruebas específicas.
Enfocarse en 2 catástrofes para la creación de alerta	Facilitar el cumplimiento de sprint 1 y 2	Medio: Afecta el backend y los tiempos del sprint.	Especializarse en las catástrofes Terremoto e Incendio y sus comportamientos para aconsejar al usuario.

Tabla 3 - Gestión de cambios

6.4 Tabla de Gestión de Riesgos

Riesgo Identificado	Impacto	Probabilidad	Acciones Preventivas	Acciones Correctivas
Falta de tiempo del equipo	Alto	Alta	Crear un cronograma detallado y establecer prioridades claras.	Reasignar tareas, reducir alcance o extender plazos.
Integración fallida con APIs externas	Medio	Alto	Evaluar necesidad de integración de API. Seleccionar APIs probadas y realizar pruebas tempranas.	De ser necesaria, desarrollar API propia.
Comunicación deficiente entre el equipo	Alto	Alta	Implementar reuniones diarias y centralizar la documentación.	Realizar sesiones de retrospectiva para resolver problemas rápidamente. Realizar más sesiones con stakeholder.
Fallas técnicas durante la Feria de Software	Alto	Alta	Realizar pruebas exhaustivas antes del evento.	Tener una copia de seguridad funcional.

Tabla 4 - Gestión de Riesgos

7. Evaluación del Desempeño Ágil

Resultados de Sprints

Sprint 1

- Alerta discriminatoria
 - Crear interfaz de la alerta: Cumplida e implementada.



Ilustración 6 - Vista de inicio



Ilustración 7 - Vista de sección Informativo ante catástrofes

- Crear existencia de diferentes tipos de catástrofes: No terminada
- Obtener ubicación de la catástrofe: No terminada
- Obtener ubicación del usuario: No terminada
- Calcular distancia entre catástrofe y usuario: No iniciada.

- Detección de distancia
 - Recuperación de la distancia de la catástrofe y usuario: No terminada.
 - Mostrar al usuario y catástrofe en un mapa: No terminada.
- Cálculo de vía de escape
 - Recuperación de la distancia de la catástrofe y el usuario: No iniciada
 - Cálculo de vía de evacuación: No iniciada.
 - Tipificar la alerta con respecto a la catástrofe: No iniciada
- Personalización de la alerta
 - Crear interfaz: Cumplida e implementada
 - Seleccionar sonidos demo: Cumplida e implementada
 - Implementar cargador de archivos a la aplicación: Cumplida e implementada.
 - Reproducir tono al detectar una alerta: Cumplida e implementada

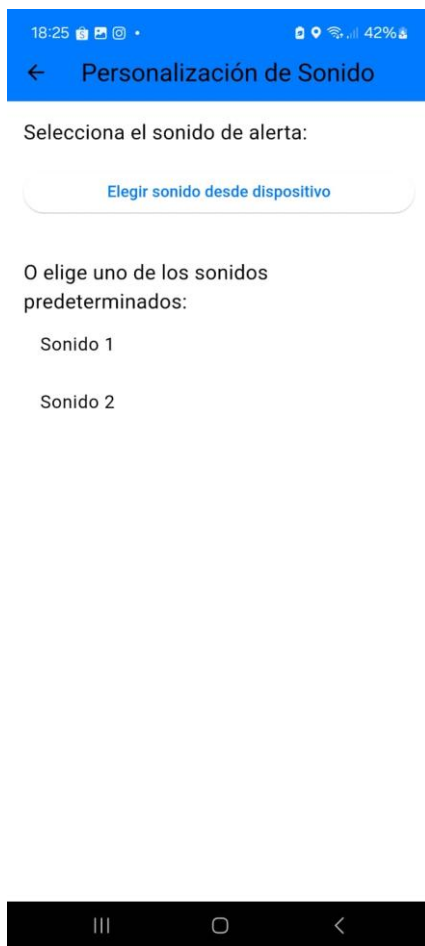


*Ilustración 8 - Vista de sección
Personalización de la alerta*

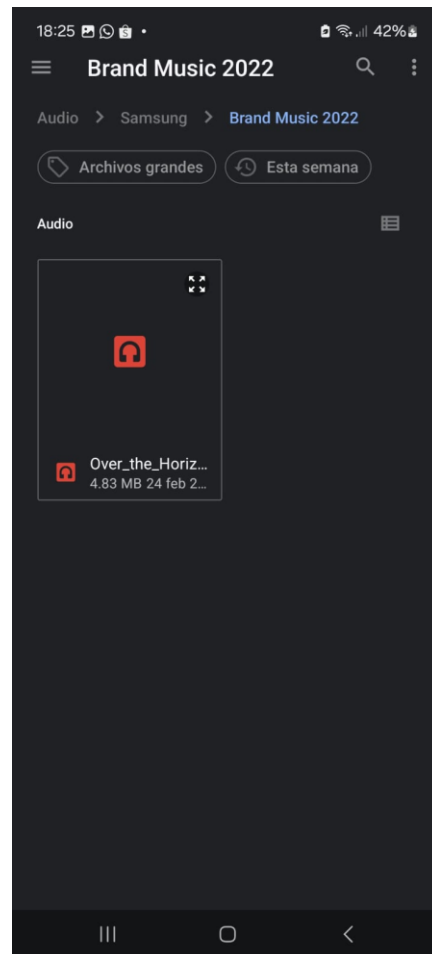


*Ilustración 9 - Vista de cargador de
archivos funcional*

Sprint 2

- Alerta discriminatoria
 - Crear interfaz de la alerta: Actualizaciones de vistas



Ilustración 10 - Vista de nuevo inicio

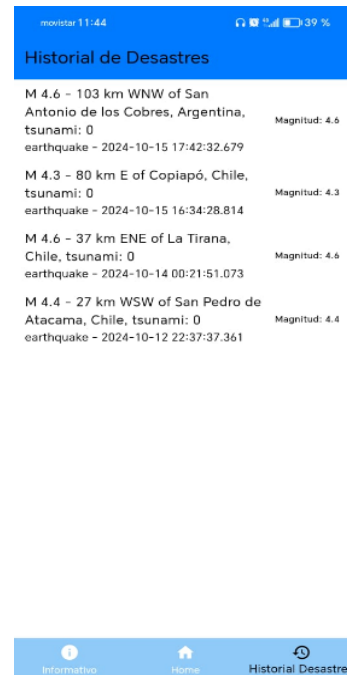


Ilustración 11 - Vista de nueva sección Historial de desastres

- Crear existencia de diferentes tipos de catástrofes: Cumplida e implementada
- Obtener ubicación de la catástrofe: Cumplida e implementada
- Obtener ubicación del usuario: Cumplida e implementada
- Calcular distancia entre catástrofe y usuario: Cumplida e implementada con errores.
- Detección de distancia
 - Recuperación de la distancia de la catástrofe y usuario: Cumplida e implementada
 - Mostrar al usuario y catástrofe en un mapa: Cumplida e implementada
- Cálculo de vía de escape
 - Recuperación de la distancia de la catástrofe y el usuario: Cumplida e implementada
 - Cálculo de vía de evacuación: No terminada
 - Tipificar la alerta con respecto a la catástrofe: No terminada

Sprint 3

- Cálculo de vía de escape
 - Cálculo de vía de evacuación.: Cumplido e implementado con errores
 - Tipificar la alerta con respecto a la catástrofe.: Cumplido e implementado.
- Accesibilidad de la alerta
 - Implementar lectura de alerta en texto a voz: No iniciada.
 - Testear usabilidad con gente con discapacidad visual: No iniciada.
- Informativo ante catástrofes
 - Investigar tipos de catástrofes: Cumplido e implementado (2/5).
 - Describir información importante para tipo de catástrofe: Cumplido e implementado (2/5).
 - Crear interfaz que muestre la información: Se investigó, describió y creó la interfaz de sólo 2 de las 5 catástrofes contempladas (Terremoto e Incendio).



Ilustración 13 - Interfaz final de sección Informativo ante catástrofes



Ilustración 12 - Maqueta inicial de sección Informativo ante catástrofes



Ilustración 14 - Vista de informativo de Terremoto



Ilustración 15 - Vista de informativo de Incendio

Métricas de eficacia y análisis de resultados obtenidos

- Cumplimiento de hitos
 - Sprint 1: Se cumplió 1 historia de usuario de 4 (25%).
 - Sprint 2: Se cumplieron 2 historias de usuario de 3 (66%).
 - Sprint 3: Se cumplieron 2 historias de usuario de 3 (66%).
- Satisfacción del cliente

Las reuniones con *Globant* y el público que asistió al hito de Feria de software demostró y aseguró que los objetivos del proyecto están bien enfocados y satisfacen las necesidades del cliente y del público objetivo.
- Defectos reportados

Al momento de los Sprint 1 y 2 no se demostraron defectos del plan de acción del proyecto, ni bugs de la aplicación. Sin embargo, en el fin del sprint 3 (específicamente durante el hito de Feria de software) se presentó un bug el cuál consistía que el usuario si se encontraba en zona de riesgo y se trasladaba a una zona segura, la aplicación seguía mostrando que estaba en riesgo.
- Cumplimiento de historias de usuario
 - Alerta discriminatoria – Cumplimiento de 89%
 - No se cumplió por completo porque no se detectó de forma efectiva el bug detallado en defectos reportados que sucedió en el hito de Feria de software. Este defecto encontrado, que nacería en el desarrollo de la tarea “Calcular distancia entre catástrofe y usuario”, disminuyó la eficacia de las historias de usuario 1, 2 y 3.
 - Detección de distancia – Cumplimiento de 100%
 - Si bien se completaron las tareas asignadas a esta historia de usuario de manera conforme, estas tareas son dependientes de una historia de usuario predecesor que presenta errores.
 - Cálculo de vía de escape – Cumplimiento de 90%
 - No se cumplió de forma satisfactoria la tarea “Calcular vía de escape”, ya que esta tarea es dependiente de una historia de usuario predecesora que presenta errores.
 - Accesibilidad de la alerta – No cumplida 0%
 - Por dificultades de cumplir el término de otras historias de usuario, no se logró empezar a trabajar con esta HU.
 - Personalización de la alerta – Cumplimiento de 100%
 - Informativo ante catástrofes – Cumplimiento de 85%
 - Sólo se cumplió la inclusión de 2 catástrofes de los 5 que se tenían estipulados.

8. Desafíos y Soluciones

Obstáculos identificados

- **Tiempo limitado:** Los integrantes tienen responsabilidades académicas adicionales.
Como alumnos de la USM, el equipo no puede dedicarse por completo al proyecto, haciendo que cada integrante tenga que organizarse con respecto a sus obligaciones y disponibilidad personal.
- **Desalineación en prioridades:** Algunos desarrolladores priorizan tareas no críticas.
Los desarrolladores aspiran a completar tareas que están destinadas al proceso de escalar el proyecto, sin poder completar satisfactoriamente lo esencial a entregar. Este obstáculo sucede por no seguir el plan de acción que se planeó desde el inicio.
- **Problemas de comunicación:** Falta de claridad en la distribución de tareas.

Estrategias para superarlos

- **Gestión del tiempo:** Crear un calendario compartido para coordinar entregables y disponibilidad.
- **Aclaración de objetivos:** Reunirse con la empresa patrocinadora *Globant* para aclarar dudas del equipo, acreditar los objetivos por cumplir, revisar el plan de acción y redefinir las prioridades.
- **Mejorar la comunicación:** Utilizar Discord para centralizar discusiones y mantener claridad en el progreso.

9. Estimación de costos.

Categoría	Elemento	Descripción	Costo unitario (CLP)	Cantidad	Costo total
Desarrollo de software	Salario de Scrum Master (1)	Tarifa horaria x horas estimadas	10.000	271	2.710.000
	Salarios desarrolladores (3)	Tarifa horaria x horas estimadas	10.000	271	8.130.000
	Licencias de software	Herramientas de desarrollo (Visual Studio, GitHub Premium, etc.)	200.000	1	200.000
	Hosting	Servidor para pruebas y producción	15.000	3	45.000
Otros costos	Electricidad y conectividad	Costo aproximado para 6 meses de trabajo	50.000	3	150.000

Tabla 5 - Estimación de costos

Total estimado: 11.235.000

10. Conclusiones

El proyecto Alertify logró implementar exitosamente la metodología Scrum, cumpliendo con su objetivo general de gestionar el desarrollo de un sistema de comunicación ante catástrofes de manera ágil y adaptativa. A través de tres Sprints (totalizando 1,084 horas de trabajo en un lapso de 3 meses, por un equipo de 4 personas), se priorizaron funcionalidades críticas como la alerta discriminadora y el cálculo de vías de escape, entregando incrementos funcionales en cada iteración.

Objetivos específicos:

- I. Identificar y analizar las principales funciones y responsabilidades del Scrum Master y del Product Owner en el contexto del proyecto Alertify, y cómo su colaboración impacta en el éxito del proyecto.

El rol del Scrum Master fue fundamental para facilitar el proceso ágil, resolviendo problemas como la falta de comunicación y asegurando que las ceremonias se llevaran a cabo de manera efectiva. Por otro lado, el Product Owner cumplió una labor clave al priorizar el backlog, enfocándose en funcionalidades esenciales como la alerta discriminadora y el cálculo de vía de escape. No obstante, la dedicación limitada del equipo impidió una refinación más profunda del backlog, lo que resultó en tareas pendientes, como la implementación completa de la accesibilidad para usuarios con discapacidad visual.

- II. Diseñar un conjunto de prácticas y herramientas que faciliten la comunicación y la gestión de expectativas entre el Scrum Master, el Product Owner y el equipo de desarrollo, con el fin de mejorar la efectividad del proceso Scrum.

Para optimizar la comunicación, se adoptaron herramientas como Taiga para la gestión de tareas y Discord para la coordinación diaria, lo que mejoró significativamente la transparencia y el seguimiento del progreso. Las Daily Scrums se adaptaron a reuniones cada 48 horas, ajustándose a la disponibilidad del equipo, y las retrospectivas permitieron identificar áreas de mejora, como la necesidad de una mayor claridad en la asignación de tareas. Sin embargo, persistieron desafíos, como la desalineación ocasional en prioridades, lo que retrasó algunas integraciones técnicas clave.

- III. Evaluar el impacto del marco propuesto en la calidad de los entregables y la satisfacción del cliente a través de métricas de rendimiento, antes y después de su implementación.

La implementación de Scrum permitió medir el avance del desarrollo del proyecto a través de las historias de usuario identificadas y priorizadas, como el cumplimiento del 66% de las historias de usuario en los Sprints 2 y 3, frente a un 25% en el Sprint inicial. El feedback recogido durante la Feria de Software fue positivo, especialmente en cuanto a la usabilidad de las alertas y el mapa interactivo, validando que el MVP cumplió con los objetivos básicos del proyecto. Sin embargo, la detección tardía de bugs, como el error en la actualización de zonas de riesgo, evidenció la necesidad de incorporar pruebas más exhaustivas en etapas tempranas del desarrollo.

Las limitaciones principales de este proyecto fueron el tiempo acotado para el desarrollo y la dedicación parcial del equipo, ambos factores derivados de las responsabilidades académicas. Esto impactó en la

capacidad para implementar funcionalidades previstas como la historia de usuario “Accesibilidad de la alerta”.

Se logró aplicar la metodología Scrum de manera efectiva, adaptando sus ceremonias (Sprint Planning, Daily Scrums y Sprint Review) a las necesidades del equipo y del proyecto. La metodología permitió identificar y priorizar funcionalidades críticas y entregar incrementos funcionales en cada sprint. Sin embargo, limitaciones de tiempo y recursos afectaron la completitud de algunas historias de usuario.

Recomendaciones para futuras iteraciones:

- Pruebas tempranas: Incluir pruebas de usabilidad y técnicas en etapas iniciales para evitar bugs en fases avanzadas.
- Colaboraciones estratégicas: Establecer acuerdos previos con instituciones como SENAPRED o CONAF para acceder a datos oficiales y mejorar la escalabilidad.
- Integración de IA: Explorar predicción de rutas en tiempo real para optimizar las vías de evacuación.

El equipo de Alertify decidió que no se seguiría trabajando el proyecto Alertify por motivos de tiempo del equipo y lo complicado que sería colaborar con SENAPRED y CONAF para tener información oficial.

Impacto y Aprendizajes

A pesar de las limitaciones, Alertify validó que Scrum es efectivo para proyectos dinámicos, entregando un MVP funcional que responde a necesidades inmediatas de comunicación en emergencias. Los aprendizajes clave incluyen:

- La importancia de una gestión de riesgos proactiva.
- La necesidad de alinear expectativas con stakeholders desde el inicio.
- El valor de adaptar ceremonias ágiles a las realidades del equipo.

En conclusión, Alertify demostró que la metodología Scrum es efectiva para desarrollar soluciones tecnológicas ágiles en contextos de emergencia, entregando un MVP funcional pese a limitaciones de tiempo y recursos. Si bien no se completaron todas las funcionalidades planificadas, el proyecto sentó bases sólidas para futuras iteraciones, destacando la importancia de priorización estratégica, pruebas tempranas y colaboración institucional. Los aprendizajes obtenidos refuerzan el valor de los enfoques ágiles para transformar desafíos complejos en soluciones tangibles, dejando un marco de referencia valioso para iniciativas similares en el futuro.

Referencias

1. Universidad Técnica Federico Santa María. (2024, 29 febrero). *Megaincendios y reconstrucción: el doloroso proceso que espera a Valparaíso - Universidad Técnica Federico Santa María*. <https://usm.cl/noticias/megaincendios-y-reconstruccion-el-doloroso-proceso-que-espera-a-valparaiso/#:~:text=M%C3%A1s%20de%20%20mil%20hect%C3%A1reas,alrededor%20de%2015%20mil%20viviendas>.

Bibliografía

- Agile Alliance. (2001). The Agile manifesto. <https://agilemanifesto.org/>
- Anderson, D. J. (2010). Kanban: Successful evolutionary change for your technology business. Blue Hole Press.
- Beck, K. (2000). Extreme programming explained: Embrace change. Addison-Wesley Professional.
- Cohn, M. (2005). Agile estimating and planning. Prentice Hall.
- Poppendieck, M., & Poppendieck, T. (2003). Lean software development: An agile toolkit. Addison-Wesley Professional.
- Schwaber, K. (2004). Agile project management with Scrum. Microsoft Press.
- Schwaber, K., & Sutherland, J. (2020). The Scrum guide. Scrum.org. <https://scrumguides.org/>
- Sutherland, J. (2014). Scrum: The art of doing twice the work in half the time. Crown Business.
- Verheyen, G. (2013). Scrum: A pocket guide. Van Haren Publishing.
- Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. Information and Software Technology
- Hossain, E., Babar, M. A., & Paik, H. (2009). A survey on the effectiveness of agile methodologies in software development. International Journal of Software Engineering and Its Applications