RescueNet: Tecnología para rescatar animales

Cuando un animal en situación de calle necesita ayuda, muchas personas quieren intervenir, pero la coordinación es caótica: mensajes dispersos, voluntarios desorganizados y falta de seguimiento.

RescueNet plantea una pregunta tecnológica: ¿podemos construir un sistema que coordine rescates de forma eficiente y convierta la empatía de la comunidad en acción organizada?

1. Definición del Proyecto

Descripción del Problema

En muchas ciudades, los animales en situación de calle enfrentan abandono, enfermedades o accidentes. Cuando una persona detecta un animal que necesita ayuda, muchas veces quiere intervenir, pero se encuentra con una dificultad inmediata: **no existe una forma sencilla de coordinar a las personas y recursos necesarios para resolver el caso.**

Actualmente, muchos rescates se organizan de manera informal a través de redes sociales o grupos de mensajería. Aunque estos canales permiten difundir información, presentan varias limitaciones:

- Los reportes se pierden fácilmente entre publicaciones.
- Es difícil coordinar voluntarios, transporte o atención veterinaria.
- No existe seguimiento claro del estado de cada rescate.
- La información suele duplicarse o fragmentarse.

El resultado es que **muchos casos que podrían resolverse rápidamente terminan sin atención o con respuestas desorganizadas.**

El reto consiste en diseñar y construir **una plataforma digital que permita coordinar rescates de animales de forma estructurada y colaborativa**, conectando a personas que reportan casos con voluntarios y patrocinadores que pueden aportar tiempo o recursos.

Objetivo General

Desarrollar un **MVP funcional** de una plataforma web que permita:

- Reportar animales que necesitan ayuda,
- Visualizar los reportes en un **mapa geográfico**,
- Coordinar voluntarios y recursos disponibles,
- Dar seguimiento al progreso de cada rescate.

Durante el desarrollo, el sistema deberá estar disponible en línea en un entorno **closed-beta**, que permita validar el funcionamiento del sistema con usuarios controlados.

El objetivo es demostrar cómo el software puede **convertir la empatía de las personas en acción coordinada.**

Requerimientos Funcionales
Must Have:

- Crear reportes con ubicación, fotografías y descripción.
- Visualizar reportes activos en un mapa interactivo.
- Registrar usuarios con distintos roles (reportante, voluntario, patrocinador).
- Permitir que voluntarios participen en rescates.
- Registrar recursos ofrecidos por patrocinadores.
- Mostrar el estado de cada rescate.
- Mantener historial básico de acciones en cada caso.

Should Have:

- Detectar reportes duplicados cercanos.
- Priorizar casos según urgencia.
- Notificar a voluntarios cercanos.
- Mostrar estadísticas básicas del sistema.

Could Have:

- Sistema de reputación de usuarios.
- Visualización del impacto comunitario.

Bonus (Impacto Real – No obligatorio): Se valorará especialmente si el equipo logra validar su solución con **usuarios reales en el entorno Beta,** demostrando interés no solo en la ingeniería del sistema sino también en su adopción e impacto en la comunidad.

2. **Restricciones y Entorno Técnico**

Los equipos tienen libertad para elegir su stack tecnológico. Se recomienda utilizar tecnologías modernas de desarrollo web.

Ejemplos de tecnologías sugeridas:

- **Frontend:** React, Vue o Angular
- **Backend:** Node.js, Python
- **Base de datos:** PostgreSQL, MySQL o MongoDB
- **Mapas:** Mapbox o Google Maps API
- **Infraestructura:** Amazon Web Services, Google Cloud o Microsoft Azure

Entornos de Desarrollo
Cada equipo deberá trabajar con **dos entornos separados**:

1. **Staging** - Entorno de desarrollo y pruebas donde se validan nuevas funcionalidades.
2. **Producción** - Entorno estable para demostraciones. Solo se deben desplegar versiones previamente validadas.

El sistema deberá estar **disponible en línea desde etapas tempranas del proyecto**, iniciando con una versión básica e incorporando mejoras en cada iteración.

Entregables Técnicos

Cada equipo deberá entregar:

- Código fuente en repositorio Git (GitHub/GitLab)
- Plataforma funcional desplegada en línea
- Documentación técnica básica
- Manual breve de uso del sistema

3. **Cronograma de Desarrollo**

El proyecto se desarrollará durante **4 meses** (16 semanas). Se recomienda trabajar mediante **iteraciones de dos semanas**, incorporando mejoras progresivas al sistema.

Primer despliegue
Se espera que el primer deployment ocurra en la semana 2, aunque sea con funcionalidades mínimas.

Evaluaciones
Se realizarán evaluaciones cada **4 semanas** para revisar el avance del sistema.

Milestones sugeridos
Mes 1: Fundamentos
Primer despliegue y estructura básica del sistema.

Mes 2: Coordinación de rescates
Gestión de casos y visualización en mapa.

Mes 3: Flujo completo
Integración de roles y funcionalidades principales.

Mes 4: MVP final
Estabilización del sistema y preparación para la demo final.

4. Mentoría y Seguimiento

El proyecto contará con acompañamiento continuo por parte de mentores.

Mentoría continua - Comunicación con el mentor mediante **Microsoft Teams** para resolver dudas técnicas.

Reunión semanal presencial - Cada equipo tendrá una reunión de **1 hora por semana en las oficinas de la empresa**, para revisar avances y discutir decisiones técnicas.

Evaluación mensual - Cada 4 semanas los equipos presentarán una demo del sistema a evaluadores, quienes registrarán el progreso del proyecto.

5. Criterios de Evaluación

Los equipos serán evaluados mensualmente considerando:

1. **Demo funcional** - Calidad, estabilidad y funcionalidad de la versión desplegada.
2. **Incrementos de valor** - Evolución del sistema entre evaluaciones.
3. **Ingeniería de software** - Buenas prácticas de código, arquitectura y despliegue.
4. **Organización y metodología** - Planificación del trabajo y gestión del equipo.
5. **Calidad de la presentación** - Claridad al explicar avances y decisiones del proyecto.

Escala de evaluación

- **Excelente:** supera ampliamente la expectativa
- **Bueno:** cumple adecuadamente
- **Deficiente:** cumplimiento mínimo
- **Insuficiente:** sin avance claro