Universidad Espíritu Santo

Modalidad en Línea

Ingeniería en Ciencias de la Computación

PLATAFORMA DE REPORTE DE PROBLEMAS URBANOS

Asignatura	: Lenguajes de Programación
Docente	: MGS. David Gurumendi Párraga
Fecha	: Quito, 14 de abril del 2025
Versión	: 7.0
Creado por	: Alex Mendoza Morante
	Carlos Plúa
	David Tello

Especificación de requisitos de software

Historial de Versiones			
Versión	**Autor**	**Descripción**	**Fecha**
1.0	David Tello	Creación del Documento de especificación	14/04/2025
2.0	David Tello	Detalle de secciones del documento	15/04/2025
3.0	Carlos Plúa	Declaración de requisitos funcionales	15/04/2025
4.0	Carlos Plúa	Declaración de interfaz interna	16/04/2025
5.0	Alex Mendoza	Requisitos no funcionales	16/04/2025
6.0	Alex Mendoza	Definiciones y Acrónimos	16/04/2025
7.0	Alex Mendoza	Corrección de errores ortográficos	17/04/2025

Historial de Revisiones			
Versión Revisada	**Revisor**	**Descripción**	**Fecha**
1.0	Alex Mendoza	Revisión del documento y sus secciones	14/04/2025
4.0	Carlos Plúa	Revisión de alcance	16/04/2025
6.0	Alex Mendoza	Revisión de requisitos y alcance del proyecto	17/04/2025

Historial de Aprobaciones			
Versión Aprobada	**Aprobador**	**Descripción**	**Fecha**
7.0	Alex Mendoza	Revisión del documento y sus secciones	17/04/2025

Especificación de requisitos de software

INTRODUCCIÓN

El presente proyecto tiene como finalidad el desarrollo de una plataforma web que permita a los ciudadanos reportar, de manera sencilla y georreferenciada, diversos problemas o incidencias que afectan su entorno urbano. A través de un sistema intuitivo y accesible, los usuarios podrán identificar en un mapa la ubicación exacta de problemas como baches, luminarias apagadas, acumulación de basura, veredas rotas, entre otros, acompañando el reporte con una breve descripción y una fotografía que respalde la denuncia. Esta herramienta busca fortalecer el vínculo entre la comunidad y las autoridades locales, facilitando el seguimiento y la priorización de las tareas de mantenimiento urbano.

En muchas ciudades, la comunicación entre la ciudadanía y los entes responsables del mantenimiento del espacio público es limitada, lo que genera demoras innecesarias en la solución de problemas que afectan la calidad de vida de los habitantes. “VozUrbana” busca cubrir esa brecha brindando una plataforma transparente y participativa, en la que los reportes sean visibles para todos y su evolución pueda ser monitoreada en tiempo real. De esta forma, no solo se promueve una gestión más eficiente por parte de las autoridades, sino también una ciudadanía más activa y comprometida con su entorno.

Además del registro de problemas, el sistema incorporará un mecanismo de votación que permitirá a los ciudadanos apoyar los reportes que consideren prioritarios. Esto otorgará a las autoridades una herramienta adicional para identificar las demandas más urgentes y tomar decisiones basadas en datos generados directamente por la comunidad. También se contempla la posibilidad de implementar un panel de administración para moderadores o funcionarios municipales, quienes podrán actualizar el estado de los reportes, agregar comentarios sobre las acciones tomadas y generar estadísticas por zona o tipo de incidencia.

En resumen, “VozUrbana” se plantea como una solución tecnológica orientada a mejorar la gobernanza urbana y la calidad del espacio público mediante la colaboración entre ciudadanía y gobierno. Se espera que esta plataforma, de fácil acceso y uso, fomente una cultura de participación ciudadana, transparencia en la gestión y corresponsabilidad en el cuidado de las ciudades. Al centrarse en problemáticas cotidianas pero importantes, el proyecto aspira a generar un impacto tangible en las comunidades, contribuyendo a construir entornos más seguros, limpios y funcionales para todos.

Alcance del Producto

VozUrbana tiene como alcance principal el desarrollo de una aplicación web que permita a los ciudadanos reportar problemáticas urbanas de forma georreferenciada y con un sistema de seguimiento de estado. El sistema incluirá funcionalidades para el registro de usuarios, la creación de reportes por categoría, la carga de imágenes ilustrativas y la visualización de dichos reportes en un mapa interactivo.

Además, se incluirá un panel administrativo básico para gestionar el estado de los reportes y permitir el seguimiento de su resolución. La plataforma será accesible desde navegadores web y desarrollada en Python, utilizando el framework Django y una base de datos relacional MySQL.

Esta primera versión no contempla la integración con sistemas gubernamentales reales ni el desarrollo de una aplicación móvil nativa. Tampoco se implementarán sistemas de notificaciones por correo o SMS, ni análisis avanzados de datos. No obstante, estas funcionalidades se consideran posibles líneas de evolución del proyecto en futuras etapas.

El sistema será construido como un prototipo funcional, enfocado en la validación de la idea, la experiencia de usuario, la estructura técnica y la lógica de los procesos, priorizando la claridad, la usabilidad y la robustez de las funcionalidades esenciales.

Valor del Producto

En 'VozUrbana', los usuarios encontrarán una plataforma web intuitiva y de código abierto diseñada para transformar su participación en el mejoramiento de sus ciudades. Con esta herramienta, podrán reportar fácilmente problemas urbanos como calles dañadas, mala iluminación o acumulación de residuos mediante un sistema accesible desde cualquier dispositivo con internet. La plataforma no solo agiliza el proceso de denuncia, sino que también georreferencia y organiza los reportes en un mapa interactivo, permitiendo visualizar problemas recurrentes y priorizar soluciones. Además, fomenta la transparencia al hacer visible el seguimiento de cada caso, empoderando a los ciudadanos para colaborar activamente con las autoridades en la construcción de entornos más seguros, saludables y sostenibles.

Público Objetivo

VozUrbana está destinada a ciudadanos comprometidos con su comunidad, vecinos organizados, líderes barriales y cualquier persona que desee reportar problemas urbanos de manera eficiente y transparente. También está dirigida a municipalidades, gobiernos locales y organizaciones civiles que buscan mejorar la gestión urbana mediante la participación ciudadana y datos georreferenciados. La plataforma es ideal para quienes valoran la corresponsabilidad en el espacio público, desde jóvenes con acceso a tecnología hasta asociaciones de vecinos que requieren herramientas sencillas para incidir en soluciones concretas. Además, al ser de código abierto, resulta atractiva para desarrolladores e instituciones que quieran adaptarla a necesidades específicas de sus ciudades

Uso Previsto

VozUrbana es una plataforma web que permite a ciudadanos reportar problemas urbanos como baches, basura o alumbrado público de forma rápida, con fotos y geolocalización. Los vecinos pueden seguir el estado de sus reportes en tiempo real, mientras las municipalidades usan los datos para priorizar reparaciones con transparencia. Organizaciones y líderes barriales generan informes para impulsar cambios, y los medios verifican soluciones. Al ser de código abierto, desarrolladores pueden adaptarla a nuevas ciudades. Conectando a ciudadanos, gobiernos y sociedad civil, convierte la denuncia pasiva en acción colectiva organizada.

Descripción General

VozUrbana es una plataforma web diseñada para facilitar la participación ciudadana en la identificación y reporte de problemas urbanos que afectan la calidad de vida en las comunidades. A través de una interfaz intuitiva, los usuarios pueden registrar incidencias como baches, luminarias apagadas, acumulación de basura y otras afectaciones, geolocalizándolas mediante un mapa interactivo. Este sistema permite adjuntar descripciones y fotografías para documentar mejor la situación, generando reportes que pueden ser visualizados públicamente por otros ciudadanos y autoridades competentes.

La plataforma se estructura como una herramienta colaborativa que busca cerrar la brecha entre la ciudadanía y las entidades municipales. En lugar de depender de trámites engorrosos o canales poco eficaces, **VozUrbana** ofrece una vía digital abierta, transparente y en tiempo real para que los reportes sean gestionados de manera más eficiente. Además, incluye un sistema de votación donde los vecinos pueden apoyar reportes existentes, permitiendo así establecer una priorización democrática de los problemas más urgentes en cada sector.

Desde el punto de vista técnico, el sistema está desarrollado en Django (Python), con MySQL como base de datos y funcionalidades integradas con Google Maps API para la gestión espacial de los datos. Cuenta con módulos específicos para autenticación de usuarios, creación y visualización de reportes, filtros por categoría y estado, así como un panel básico de administración para el control y seguimiento de las incidencias. Todo esto se implementa bajo una arquitectura modular que facilita la escalabilidad y futuras integraciones con servicios municipales o aplicativos móviles.

El propósito de este proyecto no solo es tecnológico, sino también social. **VozUrbana** representa una oportunidad para fomentar una ciudadanía más activa, empoderada y corresponsable en el cuidado del espacio público. Al proporcionar una solución digital accesible y participativa, se espera contribuir a la mejora del entorno urbano y a fortalecer los lazos entre vecinos y autoridades. Este enfoque permite que las decisiones sobre mantenimiento urbano se tomen con base en evidencia real y participación comunitaria, generando así un impacto directo en la calidad de vida urbana.

REQUISITOS FUNCIONALES

- **Registro de usuarios ciudadanos**, El sistema debe permitir a nuevos usuarios registrarse mediante un formulario con datos básicos (nombre, correo, contraseña, etc.).
- **Inicio de sesión y cierre de sesión**, Los usuarios deben poder iniciar y cerrar sesión con credenciales válidas.
- **Recuperación de contraseña**, El sistema debe permitir recuperar la contraseña mediante un enlace enviado al correo electrónico.

- **Creación de reportes ciudadanos**, Los usuarios deben poder crear reportes de problemas urbanos ingresando título, categoría, descripción, imagen y ubicación en el mapa.
- **Geolocalización de incidentes**, El sistema debe capturar coordenadas GPS mediante un mapa interactivo para ubicar el problema reportado.
- **Visualización pública de reportes,** Todos los usuarios deben poder ver los reportes activos en un mapa, con opción de filtrarlos por categoría, estado y zona.
- **Sistema de votación ciudadana**, Los usuarios registrados deben poder votar por reportes de otros ciudadanos para priorizar los más urgentes (un voto por usuario por reporte).
- **Edición de perfil de usuario,** Cada usuario debe poder actualizar sus datos personales desde su perfil.
- **Panel de administración**, Los administradores deben poder ver la lista completa de reportes y actualizar su estado (nuevo, en proceso, resuelto).
- **Cambio de estado de los reportes**, El sistema debe permitir modificar el estado de un reporte por parte de usuarios con permisos administrativos.
- **Visualización de reportes por estado**, Los usuarios deben poder filtrar los reportes según su estado de avance.
- **Control de acceso por rol,** El sistema debe restringir funciones administrativas a usuarios con rol de moderador o administrador.
- **Prevención de acciones no autorizadas**, Los usuarios no deben poder modificar ni eliminar reportes que no les pertenecen, salvo administradores.
- **Validación de entradas en formularios**, El sistema debe validar que todos los campos requeridos estén correctamente ingresados antes de guardar información.
- **Interfaz amigable y accesible**, La aplicación debe mostrar botones, formularios y mapas de forma clara y responsiva para facilitar el uso por ciudadanos de distintos perfiles.

REQUISITOS DE LA INTERFAZ EXTERNA

Interfaz de Usuarios

- La interfaz debe ser intuitiva y responsiva, permitiendo a cualquier ciudadano utilizarla desde un navegador web sin conocimientos técnicos.
- Los elementos visuales deben ser accesibles y claros: botones grandes, colores contrastantes, formularios bien organizados.
- El sistema debe incluir:

  - Formularios de registro, inicio de sesión y recuperación de contraseña.
  - Formulario para la creación de reportes (con campos para texto, selección de categoría, subida de imagen y mapa interactivo).
  - Mapa público con filtros para navegar entre reportes.
  - Panel de usuario para ver el historial de reportes propios.
  - Panel administrativo para gestionar reportes y estados.
- La plataforma debe ser compatible con navegadores modernos (Chrome, Firefox, Edge, Safari) y adaptarse a diferentes tamaños de pantalla (PC, tablet, móvil).
- Debe cumplir principios de accesibilidad básica (uso por personas mayores o con baja visión).

Interfaz de Hardware

- El sistema será accedido desde dispositivos del usuario final como:
  - Computadoras de escritorio o portátiles.
  - Teléfonos inteligentes.
  - Tablets.
- No se requieren periféricos especiales; el único requerimiento es un dispositivo con acceso a internet y un navegador moderno.
- Para el entorno de desarrollo y pruebas:
  - Se recomienda un equipo con mínimo 8 GB de RAM y procesador i5 o superior para correr servidores locales, base de datos y entorno de desarrollo (Django + MySQL).

Interfaz de Software

- La plataforma utilizará:
  - **Django** como framework backend.
  - **MySQL** como sistema de gestión de bases de datos.
  - **HTML5, CSS3 y JavaScript** para el frontend, con soporte opcional de Bootstrap.
  - **Google Maps API** o alternativa para geolocalización de reportes.
- El servidor local deberá contar con:
  - Python 3.x
  - Librerías necesarias (django, mysql, etc.)

- El sistema será instalado y ejecutado en entorno de desarrollo local (Windows, Linux o MacOS) y podrá ser desplegado en servidores web compatibles (como Heroku, PythonAnywhere o VPS).

Interfaz de Comunicación

- El sistema debe estar habilitado para:
    - Enviar correos electrónicos automatizados mediante una interfaz SMTP (para recuperación de contraseña y futuras notificaciones).
    - Comunicarse con la API de Google Maps para la integración del mapa y captura de coordenadas.
- La arquitectura debe permitir, en futuras versiones, integraciones mediante API REST para:
    - Recibir datos desde apps móviles o sistemas externos.
    - Compartir reportes con plataformas municipales o dashboards externos.
- Todas las comunicaciones deben manejarse bajo protocolos seguros (HTTPS) en una versión en producción

REQUISITOS NO FUNCIONALES

Seguridad

- El sistema debe proteger las contraseñas de los usuarios mediante algoritmos de hashing (como pbkdf2_sha256 por defecto en Django).
- Debe implementar validación de formularios para evitar inyecciones y accesos indebidos (por ejemplo, ataques XSS o CSRF).
- Las sesiones de usuario deben ser gestionadas de forma segura, con vencimiento por inactividad.
- El acceso a funciones administrativas debe estar restringido mediante control de roles y autenticación obligatoria.
- Todas las operaciones sensibles deberán protegerse usando protocolos seguros (HTTPS en versión productiva).

Capacidad

- El sistema debe ser capaz de gestionar al menos 1,000 usuarios activos y almacenar hasta 10,000 reportes de incidentes sin comprometer la funcionalidad.

- Las imágenes adjuntas en los reportes deben estar limitadas a un peso máximo de 2MB cada una, para optimizar el almacenamiento y la velocidad de carga.

Compatibilidad

- La aplicación debe ser compatible con los navegadores más utilizados (Chrome, Firefox, Edge, Safari) en sus versiones recientes.
- Debe ejecutarse sin inconvenientes en sistemas operativos de desarrollo como Windows, Linux o macOS.
- La base de datos debe ser fácilmente migrable a otros gestores relacionales compatibles con Django, como PostgreSQL.

Confiabilidad

- El sistema debe registrar correctamente cada reporte sin pérdida de datos.
- Debe garantizar la persistencia de la información aún si ocurre una caída del servidor durante una transacción.
- El sistema debe permitir su recuperación en caso de fallo, mediante respaldo periódico de la base de datos.

Escalabilidad

- La arquitectura debe permitir escalar el sistema en caso de ser implementado a nivel municipal o regional.
- Las funciones como visualización de reportes y búsquedas por filtros deben estar preparadas para manejar grandes volúmenes de datos sin pérdida de rendimiento.
- Debe ser posible separar los componentes frontend y backend si se desea evolucionar hacia una arquitectura desacoplada (REST API + SPA).

Mantenibilidad

- El código debe estar estructurado en módulos lógicos, facilitando su comprensión y mantenimiento.
- Se deben seguir buenas prácticas de documentación del código y del uso del sistema.
- Se deben utilizar herramientas de control de versiones (como Git) para facilitar la colaboración entre desarrolladores y el seguimiento de cambios.

Facilidad de uso

- La interfaz debe ser intuitiva, con elementos visuales accesibles para personas con diferentes niveles de alfabetización digital.

- El sistema debe incluir mensajes de validación claros y retroalimentación visual en las acciones del usuario.
- El flujo de creación de reportes debe estar optimizado para que no tome más de 3 minutos en promedio.
- La navegación debe requerir un número mínimo de clics para realizar tareas comunes (registrar, reportar, votar, consultar).

Otros requisitos no funcionales

- **Accesibilidad:** El diseño debe considerar contraste de colores, tamaño de letra y estructura semántica del HTML para facilitar su uso por personas mayores o con discapacidad visual.
- **Desempeño mínimo:** El tiempo de carga de páginas clave no debe superar los 3 segundos en conexiones promedio.
- **Reutilización:** El sistema debe permitir reaprovechar componentes para futuras extensiones como apps móviles o módulos adicionales

DEFINICIONES Y ACRÓNIMOS

Definiciones

- **Ciudadanía activa:** Participación voluntaria de los ciudadanos en los asuntos públicos, promoviendo el bien común más allá de procesos electorales.
- **Problema urbano:** Situación o condición negativa que afecta el funcionamiento, seguridad o calidad del espacio público (ej. baches, basura, alumbrado deficiente).
- **Reporte:** Registro formal realizado por un usuario en la plataforma para notificar un problema urbano específico.
- **Georreferenciación:** Procedimiento mediante el cual se asocia información a una localización geográfica precisa mediante coordenadas.
- **Moderador:** Usuario con privilegios especiales para gestionar el contenido de la plataforma, incluyendo el cambio de estado de los reportes.
- **Plataforma web:** Aplicación alojada en la nube o en entorno local, accesible mediante un navegador de internet.

Acrónimos

Acrónimo	Significado
API	Application Programming Interface (Interfaz de Programación de Aplicaciones)
CSRF	Cross-Site Request Forgery (Falsificación de Petición en Sitios Cruzados)
GPS	Global Positioning System (Sistema de Posicionamiento Global)
HTML	HyperText Markup Language (Lenguaje de Marcado de Hipertexto)
CSS	Cascading Style Sheets (Hojas de Estilo en Cascada)
JS	JavaScript (Lenguaje de Programación para Interfaz Web)
SMTP	Simple Mail Transfer Protocol (Protocolo Simple de Transferencia de Correo)
SMS	Short Message Service (Servicio de Mensajes Cortos)
SPA	Single Page Application (Aplicación de Página Única)
XSS	Cross-Site Scripting (Inyección de Scripts en Sitios Cruzados)
REST	Representational State Transfer (Transferencia de Estado Representacional)
MB	Megabyte (Unidad de almacenamiento)
GB	Gigabyte (Unidad de almacenamiento mayor a MB)
RAM	Random Access Memory (Memoria de Acceso Aleatorio)
Git	Sistema de control de versiones distribuido
Frontend	Parte visual/interactiva de una aplicación que ve el usuario final
Backend	Parte lógica/servidor de una aplicación que gestiona la lógica y datos
Browser	Navegador Web (software que permite acceder a páginas web)