City Lens: Documento de Requerimientos del Proyecto

Fecha	@17 de febrero de 2025
Clases y Cursos	Ingeniería de Software
Tipo de Evaluación	Tarea
Estado	En curso
Dias Restantes	1 días

Introducción

Propósito

Este documento especifica los requerimientos funcionales y no funcionales de City Lens, un Sistema de Gestión de Incidentes Urbanos (SGIU), la cual permitirá a la población el registrar, modificar y actualizar reportes sobre una gama de incidentes urbanos del día a día, como baches, luminarias descompuestas, obstáculos en la vía pública, entre otros.

Alcance

El SGUI permitirá a la población notificar a la misma y ser notificada sobre los incidentes urbanos que puedan existir en un día común. También, permitirá el incluir multimedia sobre el incidente para, con la información necesaria, pueda ser reportado a las autoridades pertinentes para ser solucionado.

Definiciones y Abreviaciones

- **SGUI:** Sistema de Gestión de Incidentes Urbanos
- **Usuario**: Cual persona que use City Lens será considerada un usuario. Esto sin importar si solo recibe información sobre los incidentes o si crea los reportes.
- **Incidente:** Se considerará como incidente a alguna de las situaciones que se pueden reportar usando el SGUI, como lo son baches, fallos en iluminaria, etc.
- **Reporte: F**uncionalidad básica de City Lens. Un reporte será la descripción del incidente junto a una fotografía que demuestre la veracidad del mismo.

Requerimientos del Sistema

Requisitos Funcionales

Gestión de Usuarios

- El sistema permitirá el registro, edición y eliminación de usuarios.
- El sistema también permitirá visualizar los reportes sin tener una cuenta de usuario, pero con acceso limitado a visualización.

Gestión de Reportes

Autenticación de Usuarios

- El sistema deberá permitir el acceso mediante credenciales únicas (usuario y contraseña).
- Deberán existir distintos niveles de acceso: usuario, moderador, administrador.

Interacción de la Interfaz

- Un usuario registrado tendrá la habilidad de gestionar un reporte (crear, editar, borrar, marcar como resuelto, etc) de alguna de las incidencias marcadas en el sistema.
- El reporte debe de incluir título, descripción, posición espacial, tipo y al menos una imagen que acompañe la información. Por default, el reporte inicia en un estado de “Reportado”, pero puede cambiar a “En Proceso” o “Resuelto”.
- Los reportes resueltos se eliminan del mapa 24 horas después de ser resueltos.
- Se debe de usar una base de datos para almacenar información de manera persistente.
- Las interacciones del Frontend serán acciones realizadas mediante una REST API
- El Backend igualmente deberá poder procesar peticiones de una REST API y, a su vez, deberá poder realizar llamadas a la base de datos de los reportes.
- La GUI incluirá un mapa interactivo en el que se puedan mostrar incidentes categorizados por tipo y estado.

Requisitos No Funcionales

Seguridad

- Uso del protocolo HTTPS en toda la pagina.
- El registro debe pedir un mail de verificación (que se usara como login)
- Conteo de Intentos Fallidos que pueda llevar a un bloqueo de cuenta
- Las contraseñas no se deben de mostrar al momento del login por privacidad.
- Asegurarse del cifrado de contraseñas en el Backend y en el DB.
- Login obligatorio para editar los atributos de los incidentes.
- Los usuarios tienen un tiempo de espera para poder generar y/o actualizar un reporte tras el último realizado, esto para evitar alguna situación de que un usuario intente atacar a la BD de los registros.

Usabilidad

- Interfaz web intuitiva, estética y accesible para personas con alguna discapacidad visual.
- El sistema debe de tener moderación con el contenido subido a la plataforma (imágenes, palabras altisonantes, etc).

Rendimiento

- El sistema debe de poder actualizarse en menos de 3 segundos (sin contar latencia).
- Comprimir imágenes para evitar llenar el espacio disponible en la BD.
- El sistema deberá soportar al menos 100 usuarios simultáneos.

Disponibilidad

- El sistema deberá estar disponible al menos el 99% del tiempo.

Casos de Uso

Caso #1: Creación del Reporte de Incidente

Actores: Usuario

- El usuario inicia sesión.
- Selecciona el tipo de incidencia (bache, iluminación, basura, etc.).
- Adjunta fotos y una descripción.
- Elige la ubicación en el mapa o permite que la app la detecte automáticamente.
- **Flujo Alternativo:** Si el sistema no logra detectar la ubicación de manera correcta, el usuario puede ingresarla manualmente.

Caso #2: Gestión del Reporte

Actores: Usuario

- El usuario, tras hacer login, revisa su historial de reportes.
- Visualiza el estado de la incidencia (en caso de que haya sido modificado por otro usuario)
- El usuario puede modificar su reporte para marcarlo como resuelto o incluso, eliminarlo.

Caso #3: Moderación del Contenido en el Sistema

Actores: Moderador

- Un moderador inicia sesión.
- El moderador puede visualizar todos los reportes creados en las últimas 24 horas.
- El moderador debe revisar si un reporte dado incluye contenido prohibido (lenguaje, imágenes, etc).
- El moderador borra el reporte de ser necesario.

Caso #4: Filtrado de Incidentes en el Mapa de Visualización

Actores: Usuario

- El usuario entra en la aplicación (puede o no iniciar sesión)
- El usuario se dirige a la visualización de incidentes reportados más recientes.
- El usuario ingresa sus criterios de búsqueda (tipo y estado)

- La aplicación muestra la información de los reportes que coincidan con los parámetros de búsqueda así como el mapa con indicadores de los lugares donde se realizó el reporte.
- **Flujo Alternativo:** Si el filtrado no regresa resultados, la aplicación muestra la leyenda “Sin Resultados” y una versión general del mapa sin indicadores (ie: muestra un mapa mínimo de la ciudad).

Caso #5: Notificación de Incidentes Cercanos

Actores: Usuario

- El usuario inicia sesión (opcional)
- El sistema detecta la ubicación actual del usuario
- El usuario activa la opción de recibir notificaciones sobre incidentes cercanos
- El sistema envía alertas en tiempo real cuando se registra un nuevo incidente en un radio definido.
- El usuario puede acceder a la información del incidente desde la notificación.
- **Flujo Alternativo**: Si el usuario no permite la geolocalización, se le pedirá ingresar una ubicación manualmente.

Requisitos de Hardware y Software

Hardware

Esto en caso de ser alojado en algún sitio y deje de estar en localhost, podría estar alojado en algún servidor físico como una VM. Por ejemplo, una instancia en AWS o Azure. Si tuviera que ser un servidor propio, bastaría con tener un equipo con un SSD de al menos 125 GB con 2 núcleos y 4 GB de RAM.

Software (Librerías, Lenguajes, IDE, SQL...)

- Se seleccionó **Spring Boot y Kotlin** como librerías y lenguajes para el Backend ya que el equipo tiene conocimientos en Java, y por consiguiente, de la JVM. A esto se suma el soporte por parte de los ayudantes de laboratorio de la materia sobre los mismos.
- Se seleccionó **Figma y React** para realizar el Frontend por su versatilidad y relativa facilidad para hacer un frontend funcional. (Figma tiene funcionalidades que permiten exportar el proyecto a código de React).
- En caso de requerir ser alojado en un servidor remoto, como control de dominio de la página se seleccionaron **Cloudflare y Hostinger**.

  Se seleccionó por ser los servicios de control de dominio mas usados.

- **PostgreSQL** para la base de datos. Se seleccionó debido a los requerimientos del proyecto debido a que es la base de datos mas útil para estos fines.
- **Postman** se seleccionó como sistema de tests del Backend sin Interfaz.
- Se eligió a **IntelliJ** como IDE por ser el IDE de la JVM por experiencia, y también por su capacidad de actualización dinámica de clases, permitiéndonos probar cosas sin recompilar el código.
- Se seleccionó **Notion** para gestionar el proyecto por su versatilidad para hacer todo tipo de documentos, para tener un mejor control del proyecto.
- Se usarán las especificaciones de **Conventional Commits** para, como su nombre lo dice, los commits que se harán en la creación del software.

Consideraciones Finales

Este documento establece los requisitos iniciales del SGUI. Cualquier modificación, ampliación o eliminación de requisitos será documentada y posteriormente aprobada por los interesados en el desarrollo del mismo.