Documento de Arquitectura del Sistema

**SIPRA-VR-RS01**

|  |  |  |  |
| --- | --- | --- | --- |
| Responsable | Diseño | Versión | Fecha de Creación |
| Max Saavedra | Arquitectura | 1.0 | 14/04/2025 |

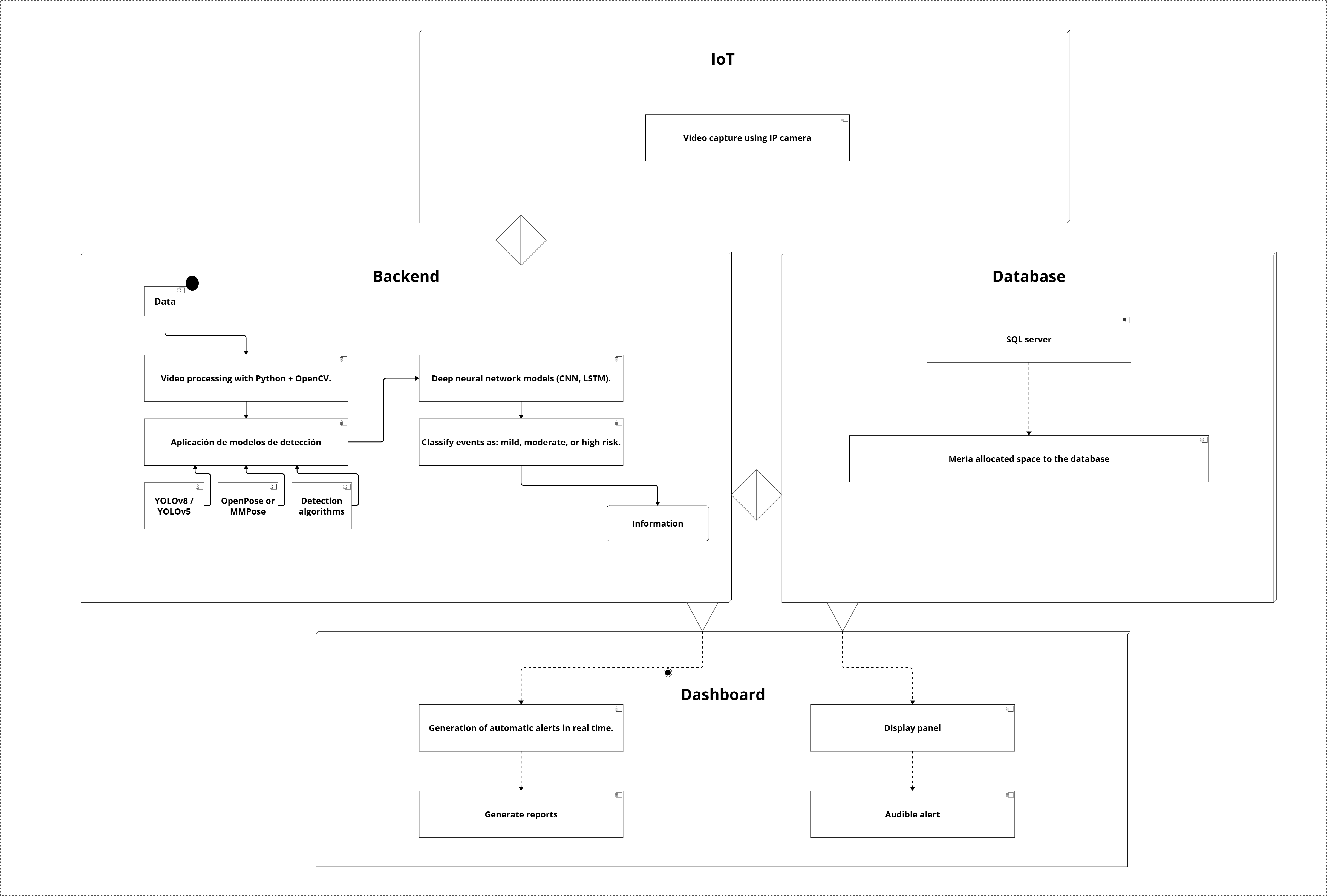
## 1. Introducción

El presente documento describe la arquitectura del sistema propuesto para la detección y prevención de accidentes en espacios abiertos al público. La solución se basa en el uso de tecnologías de visión por computadora, redes neuronales profundas y módulos de comunicación en tiempo real. El objetivo es desarrollar un sistema inteligente capaz de analizar video en directo, detectar situaciones de riesgo, clasificar su nivel de peligrosidad y emitir alertas automatizadas al personal correspondiente, todo dentro de un tiempo de respuesta adecuado y con alta disponibilidad.

## 2. Diagrama de Componentes

El sistema propuesto se organiza en torno a tres componentes principales: **Backend**, **Base de Datos** y **Dashboard**, con la opción de considerar un cuarto componente para dispositivos **IoT** en caso de incluir hardware de captura inteligente (edge computing). A continuación, se describe detalladamente el funcionamiento interno de cada componente y su interacción con los demás.

**Imagen 1:** Diagrama de Componentes



*Nota: Interacción entre los componentes de IoT, Backend, Database y Dashboard*

### 1.1 Componente Backend

Este componente constituye el núcleo del sistema y está encargado de recibir, procesar y analizar los flujos de datos provenientes de las cámaras de vigilancia.

Flujo de procesamiento:

* **Recepción de datos en formato de video (streaming)**

El sistema recibe transmisiones de video en tiempo real a través de cámaras IP utilizando protocolos como **RTSP** o **HLS**. El formato de entrada es comúnmente **MP4**, **AVI** o secuencias de frames en **JPEG**/**PNG** si se realiza procesamiento por cuadros.

* **Procesamiento de video con Python y OpenCV**

Utilizando **OpenCV**, el sistema preprocesa los videos para reducir ruido, extraer regiones de interés y preparar los datos para el análisis con modelos de inteligencia artificial.

* **Aplicación de modelos de detección**

Se emplean algoritmos de visión por computadora y modelos preentrenados como:

* + **YOLOv8 / YOLOv5** para la detección de personas y objetos.
  + **OpenPose o MMPose** para análisis postural y detección de caídas.
  + **Modelos basados en CNN** para detección de humo, fuego o comportamiento inusual.
* **Redes neuronales profundas**

Una vez identificadas las entidades en la escena, se emplean **modelos de redes neuronales** como **CNN** (para reconocimiento de patrones visuales) y **LSTM** (para análisis de secuencias temporales) que permiten comprender el contexto del evento.

* **Clasificación de eventos**

Los eventos detectados se clasifican en tres niveles de riesgo:

* + Leve
  + Moderado
  + Alto: Esta clasificación se basa en parámetros como la duración del evento, la cantidad de personas involucradas y el tipo de amenaza visual identificada.
* **Salida**

La información procesada y clasificada se envía simultáneamente al componente **Dashboard** para su visualización y al componente **Base de Datos** para su registro permanente.

### 1.2 Componente Base de Datos

Este componente se encarga de almacenar toda la información relevante del sistema de forma estructurada, segura y consultable.

* **Gestor de base de datos**

Se utiliza **SQL Server** como sistema de gestión de base de datos relacional, lo cual garantiza robustez y escalabilidad.

* **Estructura de almacenamiento**

La base de datos cuenta con tablas para almacenar:

* + Detalles de eventos detectados
  + Tiempos de detección y clasificación
  + Niveles de riesgo asignados
  + Historial de alertas
  + Usuarios y configuraciones del sistema
* **Manejo de espacio**

El sistema asigna un área específica de almacenamiento para registros, clips de video asociados, y logs de actividad del sistema, asegurando integridad y disponibilidad de la información.

### 1.3 Componente Dashboard

Este módulo representa la interfaz visual del sistema, diseñado para que operadores o personal autorizado puedan interactuar con la información procesada en tiempo real.

**Funcionalidades principales:**

* **Generación automática de alertas en tiempo real**

Las alertas se despliegan visual y auditivamente ante la ocurrencia de eventos de riesgo, permitiendo la intervención inmediata.

* **Generación de reports**

Se permite la exportación y visualización de reportes históricos basados en fecha, tipo de evento o ubicación, facilitando el análisis posterior.

* **Panel de visualización**

Incluye mapas, íconos de alerta, tablas de incidentes y gráficos analíticos que reflejan la situación actual del entorno supervisado.

* **Alerta audible integrada**

El sistema puede emitir sonidos configurables para llamar la atención ante alertas críticas.

### 1.4 Componente IoT

El componente IoT está compuesto por dispositivos inteligentes capaces de capturar y preprocesar información localmente antes de enviarla al backend para su análisis completo. Este componente es fundamental en entornos donde se requiere minimizar la latencia o donde las condiciones de conectividad pueden ser variables.

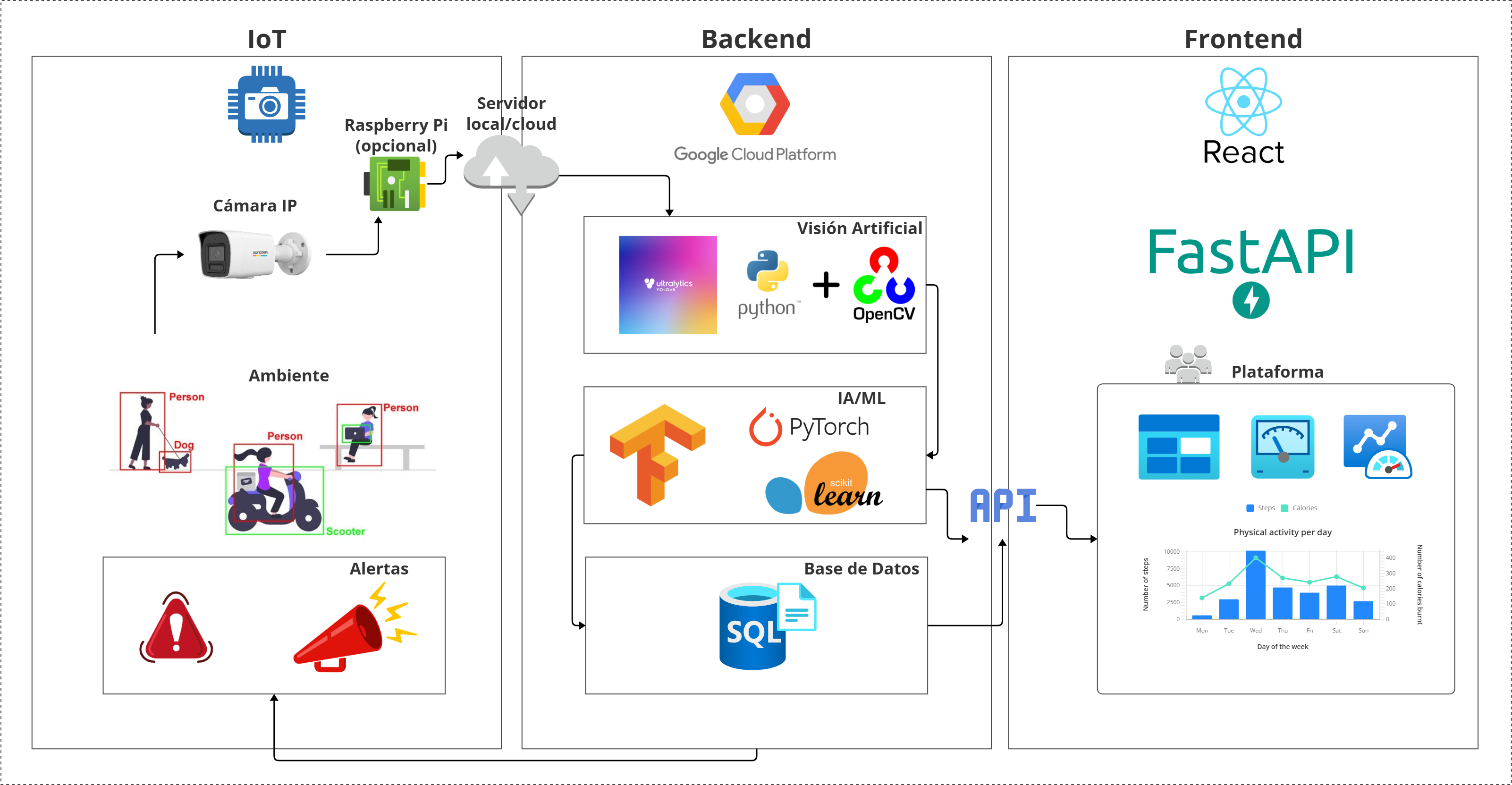
Entre los dispositivos utilizados se encuentran el microordenador Raspberry Pi y cámaras inteligentes con capacidades de procesamiento en el borde. Estos dispositivos ejecutan versiones optimizadas de modelos de visión por computadora, como YOLOv8 o SSD MobileNet, lo que les permite identificar personas, movimientos anómalos u objetos sospechosos sin depender completamente de un servidor central.

El procesamiento local permite detectar eventos críticos con mayor rapidez, enviar únicamente los eventos relevantes al backend, y continuar funcionando incluso ante una caída temporal de la red. Además, los dispositivos IoT pueden actuar como nodos autónomos dentro del sistema, aportando resiliencia, eficiencia energética y escalabilidad al diseño general.

## 3. Arquitectura General del Sistema

La arquitectura general de esta solución tecnológica está organizada en tres partes clave: **IoT**, **Backend** y **Frontend (Dashboard Web)**. Cada uno cumple una función específica en la prevención de accidentes en espacios públicos a través de tecnologías de video, inteligencia artificial y sistemas de alerta automatizados. Esta distribución modular permite una implementación escalable, eficiente y adaptable a distintos contextos urbanos.

**Imagen 2:** Diagrama General



*Nota: Diagrama general de las herramientas*

### 3.1 IoT

La sección **IoT** (Internet de las Cosas) corresponde al entorno físico donde suceden los hechos. Aquí se ubican:

* **Cámaras inteligentes o cámaras IP**, que vigilan constantemente espacios como parques, estaciones o campus universitarios.
* De manera opcional, estas cámaras pueden estar conectadas a pequeños dispositivos como un **Raspberry Pi**, que permiten realizar un primer análisis del video en el mismo lugar donde ocurre la acción, sin necesidad de enviarlo todo a la nube. Esto reduce el tiempo de reacción.
* Además, el sistema puede incluir **dispositivos de alerta local**, como luces o sonidos que se activan automáticamente cuando se detecta una situación de riesgo.

### 3.2 Backend

El **backend** es el corazón del sistema. Se encuentra en la nube, en una máquina virtual de alta disponibilidad (por ejemplo, en Google Cloud Platform) y tiene las siguientes funciones principales:

* **Analiza los videos recibidos** con técnicas de visión artificial (por ejemplo, YOLOv8 y OpenCV) para detectar personas, caídas o comportamientos inusuales.
* Una vez detectado el evento, se aplica inteligencia artificial (con herramientas como TensorFlow o PyTorch) para **clasificar el nivel de peligro** del evento: leve, moderado o alto.
* La información se **almacena en una base de datos segura y estructurada**, permitiendo revisar el historial de eventos, generar reportes y respaldar decisiones futuras.
* Desde aquí también se pueden **activar de forma remota** los sistemas de alerta del entorno (luces o sirenas), generando una respuesta inmediata y automática.

### 3.3 Frontend

El **frontend** es la parte visible del sistema para los usuarios autorizados, como el personal de seguridad o los operadores. Consiste en un **panel de monitoreo web** que se puede abrir desde cualquier computadora o dispositivo móvil.

Este panel permite:

* **Visualizar alertas en tiempo real**, saber qué pasó, cuándo y dónde.
* Consultar **informes detallados** y estadísticas de seguridad por zonas, horarios o tipos de eventos.
* **Controlar configuraciones del sistema**, como zonas de riesgo o criterios para activar alarmas.
* Actuar rápidamente ante emergencias, gracias a una interfaz intuitiva desarrollada con tecnologías modernas como React y FastAPI.

## 4. Arquitectura de Software

La arquitectura del software propuesta para el sistema de detección de riesgos en espacios públicos sigue el paradigma clásico de **cliente-servidor**, donde las operaciones críticas de procesamiento y análisis se gestionan desde el backend (servidor), mientras que el frontend (cliente) proporciona la interfaz de interacción y visualización para los usuarios finales.

**Imagen 3:** Diagrama de Arquitectura de Software

Gráfico

El contenido generado por IA puede ser incorrecto.

*Nota: Arquitectura cliente-servidor*

### 4.1 Backend

El backend del sistema se implementa como un servicio persistente que recibe continuamente datos provenientes de cámaras IP u otras fuentes de video. Este componente se despliega en una máquina virtual (VM) alojada en la nube y cumple las siguientes funciones secuenciales:

Una vez que se recibe el video, este es transferido al **Módulo de Procesamiento de Visión**, donde se ejecutan rutinas escritas en Python utilizando la biblioteca OpenCV. En esta etapa, el sistema realiza la preprocesamiento de imágenes, eliminación de ruido, segmentación de objetos y extracción de características relevantes. Esto permite obtener representaciones visuales más limpias y manejables para su posterior análisis.

A continuación, los datos procesados se envían al **Módulo de Aplicación de Modelos de Detección**, donde se utilizan modelos de visión artificial como YOLOv8, además de otros algoritmos especializados que permiten identificar eventos como la presencia humana, objetos sospechosos, caídas, comportamientos anómalos o focos de incendio.

Una vez detectado un evento, se transfiere al **Módulo de Clasificación de Riesgo**, en el cual se aplica inteligencia artificial para determinar la severidad del evento mediante modelos de redes neuronales profundas. Estos modelos (por ejemplo, CNN o LSTM) evalúan variables como el tipo de evento, duración, ubicación y movimiento para asignarle un nivel de criticidad (leve, moderado o alto).

Finalmente, toda la información se organiza y se almacena en una base de datos estructurada, y al mismo tiempo se prepara en formato JSON para ser consumida por el frontend del sistema.

### 4.2 Frontend

El frontend es el componente cliente que se comunica con el backend a través de una **API REST**, recibiendo la información procesada en formato JSON. Esta información contiene los eventos detectados, su clasificación, la hora de ocurrencia y, si corresponde, una alerta generada.

Este cliente, desarrollado con React.js, interpreta los datos recibidos y los presenta en una interfaz gráfica amigable y responsiva. A través de esta plataforma, el usuario puede monitorear alertas en tiempo real, acceder a estadísticas históricas, consultar eventos clasificados y visualizar elementos gráficos como mapas, listas, o indicadores visuales de criticidad.

Además de la visualización, el frontend también permite aplicar filtros, generar reportes, y en algunos casos, configurar parámetros del sistema si se habilita acceso administrativo.

### 4.3 Comunicación y flujo de datos

El flujo de información entre cliente y servidor se gestiona mediante peticiones HTTP sobre una API implementada con FastAPI. El backend envía respuestas en formato estructurado (JSON), lo cual permite una integración rápida y eficiente con la interfaz web, manteniendo la asincronía y escalabilidad del sistema.

Esta arquitectura basada en cliente-servidor garantiza una separación clara de responsabilidades: el backend se encarga del análisis intensivo de datos y la toma de decisiones, mientras que el frontend ofrece una plataforma accesible y dinámica para la supervisión y control por parte del usuario final.

### 4.4 Módulo de Registro y Almacenamiento

Este módulo se encargará de registrar cada evento detectado en una base de datos estructurada, incluyendo tipo de evento, fecha, hora, ubicación y clasificación de riesgo. Se utilizará una base de datos relacional como PostgreSQL o SQL Server, lo que permitirá consultas eficientes y reportes históricos..

### 4.5 Módulo de Visualización

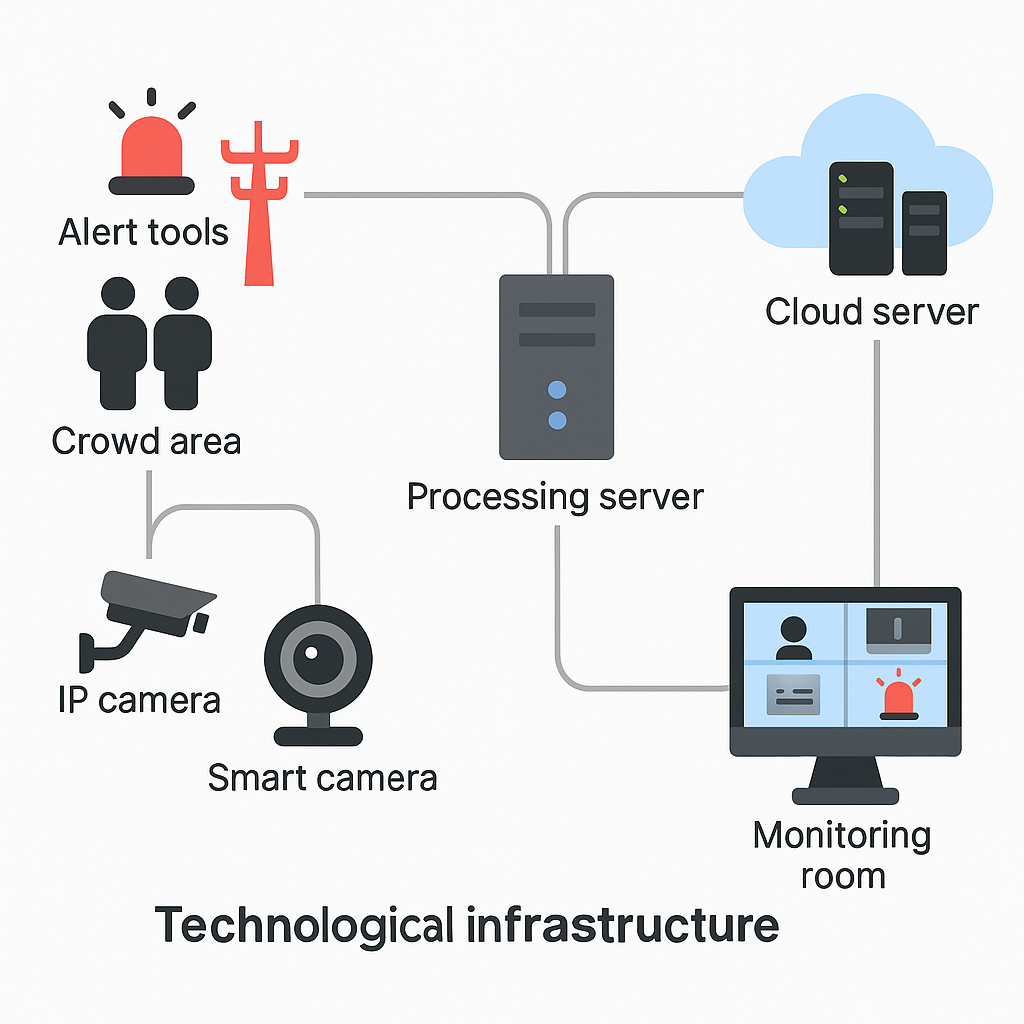
Se desarrollará una interfaz web dinámica y segura utilizando React.js, acompañada por librerías de visualización como Chart.js. Este dashboard permitirá al personal visualizar en tiempo real los eventos detectados, las alertas generadas, estadísticas y mapas de calor. La interfaz será responsive y accesible desde distintos dispositivos.

## 5. Infraestructura Tecnológica

La infraestructura tecnológica propuesta para el sistema de prevención de accidentes en espacios públicos está diseñada para garantizar una comunicación fluida entre los dispositivos físicos, los servidores de procesamiento y las plataformas de monitoreo. El modelo se organiza en capas funcionales que interactúan de manera segura, escalable y eficiente.

El diagrama representa una vista estructural del sistema completo, dividido en cuatro bloques principales: **Captura y Sensado**, **Procesamiento**, **Almacenamiento y Gestión**, y **Monitoreo y Respuesta**. Cada bloque contiene los componentes tecnológicos clave que se integran mediante una red de comunicaciones seguras.

**Imagen 4:** Diagrama de Arquitectura Tecnológica



*Nota: Diagrama de componentes físicos*

### 5.1 Captura y Sensado

Este bloque representa la capa más próxima al entorno físico. En ella se encuentran las **cámaras IP** instaladas en áreas públicas estratégicas, encargadas de captar video en tiempo real. Estas cámaras están conectadas a la red local o inalámbrica, y opcionalmente, pueden estar enlazadas a **dispositivos de borde (edge)** como el **Raspberry Pi**, que permiten preprocesar los datos en el lugar, filtrando el contenido irrelevante y reduciendo la carga de envío.

Además, en esta misma capa se incluyen los **dispositivos de alerta física**, como **luces estroboscópicas** o **sirenas**, que se activan de forma automática desde el backend ante la detección de eventos de alto riesgo.

### 5.2 Procesamiento (Servidor en la Nube)

Los datos capturados son enviados a un **servidor central alojado en la nube**, típicamente en una máquina virtual (VM) proporcionada por servicios como Google Cloud Platform. Este servidor ejecuta el **backend** del sistema, compuesto por varios módulos:

* El **módulo de visión por computadora**, que utiliza Python y OpenCV para el procesamiento de video.
* El **módulo de detección de eventos**, donde se aplican modelos como YOLOv8, OpenPose o algoritmos entrenados con TensorFlow y PyTorch.
* El **módulo de clasificación de riesgos**, que analiza la criticidad de cada evento utilizando redes neuronales profundas.

Este bloque se comunica también con los **módulos de control de alertas**, encargados de activar los sistemas de respuesta en campo cuando sea necesario.

### 5.3 Almacenamiento y Gestión

Toda la información procesada se registra en una **base de datos estructurada**, implementada en **SQL Server**. Esta base de datos almacena eventos, usuarios, niveles de riesgo, alertas y configuraciones del sistema. La conectividad con el backend es directa y permite consultas rápidas, generación de reportes y respaldo automático de datos.

### 5.4 Monitoreo y Respuesta (Frontend)

El bloque final corresponde al **centro de monitoreo**. Aquí se encuentra el **dashboard web**, una plataforma accesible vía navegador desde dispositivos móviles o estaciones de trabajo. Desarrollada con **React.js**, esta interfaz se comunica con el backend a través de **FastAPI**, recibiendo datos en formato JSON.

Desde el dashboard, los usuarios autorizados pueden:

* Visualizar en tiempo real las alertas generadas.
* Supervisar eventos recientes y su ubicación.
* Configurar parámetros del sistema.
* Consultar reportes históricos o exportar datos analíticos.

El centro de monitoreo puede estar instalado en una **sala de control físico** o ser accedido desde múltiples ubicaciones mediante conexión segura.

## 6. Producto Mínimo Viable (MVP)

El MVP consistirá en un sistema que, a partir del video capturado por una cámara, sea capaz de:

* Detectar personas y caídas.
* Clasificar el evento como de riesgo.
* Generar una alerta en menos de 2 segundos.
* Mostrar el evento en un dashboard.
* Registrar el evento en la base de datos.

Este producto será implementado con un solo punto de vigilancia para validar funcionalidad antes de su escalamiento.

## 7. Interacción entre Componentes

1. **Captura**: Las cámaras transmiten video en tiempo real.
2. **Procesamiento**: El módulo de visión analiza el video y detecta eventos.
3. **Clasificación**: El módulo de IA determina el tipo y nivel de riesgo.
4. **Alertas**: Se emite una alerta según la configuración del evento.
5. **Registro**: Toda la información se almacena en la base de datos.
6. **Visualización**: El dashboard muestra los eventos y su ubicación en tiempo real.

Esta interacción entre módulos asegura la automatización y el monitoreo constante del sistema.

## 8. Consideraciones Finales

Esta arquitectura ha sido diseñada para garantizar escalabilidad, disponibilidad, bajo tiempo de respuesta y precisión en la detección de riesgos. Gracias a su enfoque modular, el sistema puede ser adaptado a diversos entornos urbanos y mejorado con la integración de nuevas tecnologías de inteligencia artificial. Su implementación contribuirá a mejorar la seguridad ciudadana mediante la prevención proactiva de accidentes.

