**Arquitectura de Microservicios para Sistema de Cámaras ESP32**

Crea una práctica de programación que se ejecute en local, Para programarla implementando una arquitectura de microservicios. Los microservicios se publicarán utilizando Minikube, y se empleará Kafka para establecer colas de mensajes entre ellos.

**Ejercicio:**

Se cuenta con varias cámaras **ESP32-CAM**, las cuales permiten visualizar una imagen JPEG o transmitir en streaming mediante el acceso a su IP y a un endpoint a través del navegador.

Se debe desarrollar un sistema basado en una arquitectura de microservicios que permita verificar si cada cámara está encendida o no.

El sistema deberá incluir un frontend desde el cual los usuarios puedan iniciar sesión. Una vez autenticados, se les mostrará un listado de todas las cámaras disponibles, junto con su estado en tiempo real (encendida o apagada). Dado que las cámaras pueden estar encendiéndose o apagándose, el sistema deberá reflejar su estado actualizado en tiempo real.

Si un usuario desea acceder al streaming de una cámara, podrá suscribirse mediante un botón disponible en la interfaz.

Cada vez que un usuario se suscriba a un streaming, se deberá registrar en una base de datos tanto la suscripción como la duración de la conexión.

**Microservicios Necesarios**

1. **Auth Service**: Gestiona autenticación y autorización de usuarios.
   * Registro, login y gestión de sesiones
   * Emisión de tokens JWT para acceso a otros servicios
   * Autorización basada en roles
2. **Camera Discovery Service**: Descubre y monitorea el estado de las cámaras.
   * Escaneo periódico de cámaras en la red
   * Verificación del estado (encendida/apagándose/apagada/encendiéndose)
   * Publicación de cambios de estado en Kafka
3. **Camera Streaming Service**: Gestiona las conexiones de streaming.
   * Manejo de suscripciones a streams de cámaras
   * Proxy para el streaming de video desde las cámaras ESP32
   * Publicación de eventos de suscripción en Kafka
4. **User Activity Service**: Registra la actividad del usuario.
   * Procesamiento de eventos de suscripción desde Kafka
   * Registro de tiempo de visualización
   * Almacenamiento en base de datos
5. **API Gateway**: Punto de entrada único para clientes.
   * Enrutamiento de solicitudes a microservicios apropiados
   * Validación de tokens JWT
   * Manejo de rate limiting
6. **Frontend Service**: Interfaz de usuario.
   * Panel de control para visualizar cámaras
   * Indicadores de estado en tiempo real
   * Interfaz para suscripción a streams
7. **Notification Service**: Notificaciones en tiempo real.
   * Suscripción a eventos de Kafka
   * Envío de actualizaciones por WebSockets al frontend
   * Notificaciones sobre cambios de estado de cámaras
8. **Database Service**: Gestión de datos persistentes.
   * Almacenamiento de usuarios, cámaras y actividad
   * Servicios de consulta y análisis
   * Respaldo y recuperación de datos

**Diagrama de Arquitectura y Flujo de Datos**

