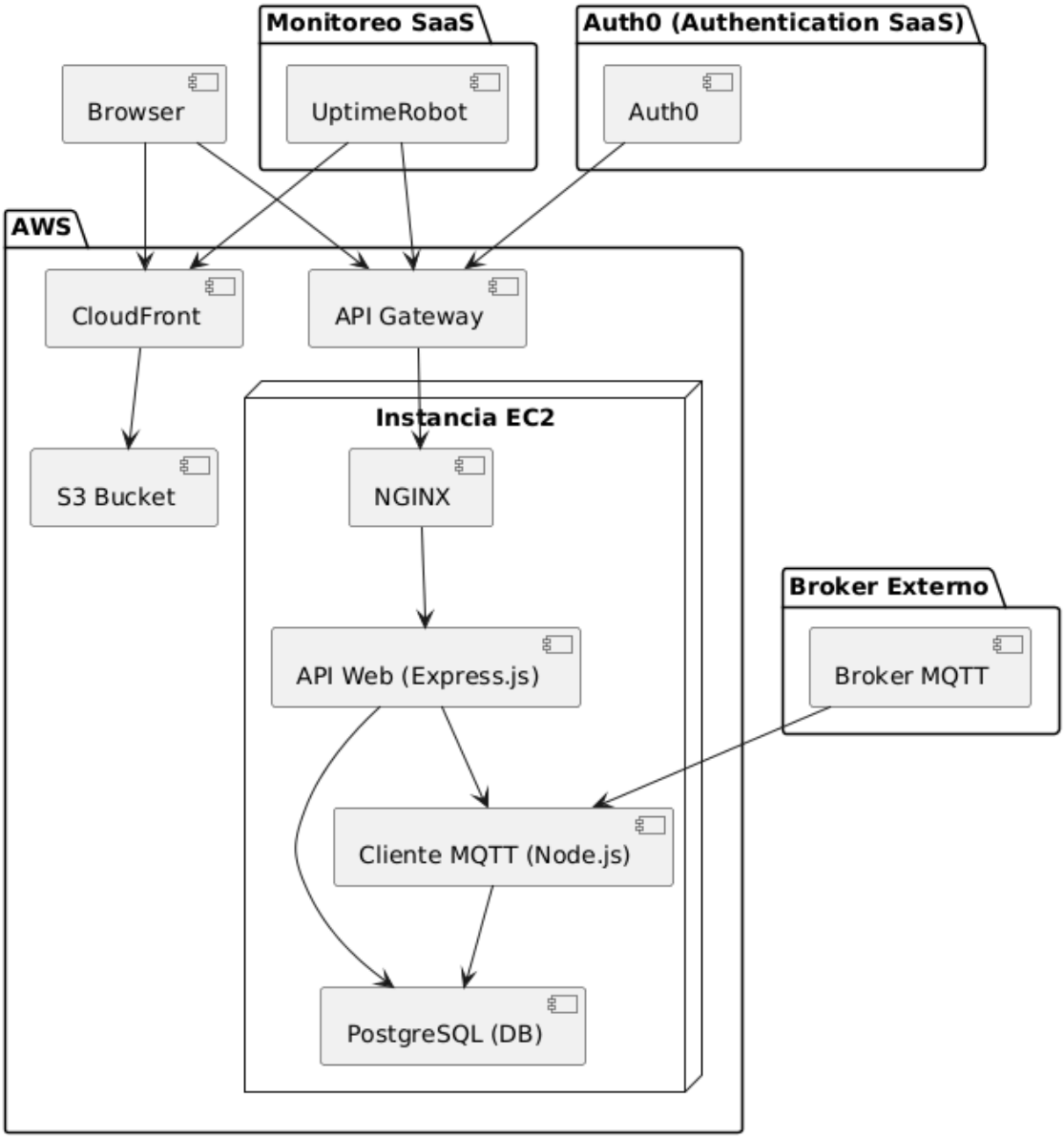
Diagrama UML



Explicación:

1. Introducción general

El sistema desplegado corresponde a una arquitectura basada en servicios independientes, contenerizados mediante Docker y desplegados en una instancia EC2 de AWS. Adicionalmente, se utilizan servicios SaaS externos para distribución de contenido, autenticación y monitoreo.

El propósito general del sistema es recibir, almacenar y permitir el acceso a información de acciones bursátiles (stocks) mediante eventos MQTT y consultas vía API REST.

2. Descripción de los componentes

Dentro de AWS:

- **EC2 Instance**:
  Contiene el núcleo del sistema:
  - **NGINX**: Servidor proxy reverso encargado de enrutar solicitudes desde el API Gateway hacia la API Web.
  - **API Web (Express.js)**: Servicio REST que expone endpoints HTTP para consultar y manipular datos de stocks.
  - **Cliente MQTT (Node.js)**: Servicio que se suscribe a un broker MQTT para recibir eventos de nuevas acciones y almacenarlas en la base de datos.
  - **PostgreSQL**: Base de datos relacional encargada de almacenar los registros de acciones recibidos.
- **S3 Bucket**:
  Repositorio de almacenamiento estático donde se alojan los archivos del frontend (HTML, CSS, JS) generados a partir de una aplicación compilada.
- **CloudFront**:
  Red de distribución de contenido (CDN) que entrega el frontend al usuario final desde el bucket S3 de manera rápida y segura mediante HTTPS.
- **API Gateway**:
  Servicio que actúa como puerta de entrada a la API REST, gestionando las solicitudes externas, delegando autenticación y asegurando el acceso controlado.

Servicios Externos:

- **Broker MQTT**:
  Servicio externo que publica eventos de información de stocks en tiempo real.
  El **Cliente MQTT** en EC2 se suscribe a este broker para recibir los datos y almacenarlos.
- **Auth0**:
  Servicio SaaS de autenticación utilizado para gestionar la identificación de usuarios y sesiones.
  API Gateway se integra con Auth0 para validar los tokens de autenticación antes de permitir el acceso a la API Web.
- **Monitoreo SaaS (UptimeRobot)**:
  Servicio de monitoreo externo que realiza pings periódicos a:
  - El endpoint del API Gateway (monitoreando la disponibilidad del backend).
  - El endpoint de CloudFront (monitoreando la disponibilidad del frontend).

Permite detectar caídas o anomalías en tiempo real.

3. Flujo de funcionamiento

1. El navegador del usuario solicita el frontend a través de CloudFront, que entrega los archivos estáticos almacenados en S3.
2. Cuando el usuario realiza acciones que requieren datos de stocks, el navegador envía solicitudes HTTP a través de API Gateway.
3. API Gateway valida la autenticación de la solicitud mediante Auth0. Si la autenticación es válida, reenvía la solicitud a NGINX.
4. NGINX enruta la solicitud internamente hacia la API Web (Express.js).
5. API Web puede consultar la base de datos PostgreSQL para obtener registros de acciones o puede solicitar información procesada desde el Cliente MQTT, si corresponde a operaciones activas sobre los eventos recibidos.
6. Paralelamente, el Cliente MQTT (Node.js) permanece suscrito al Broker MQTT externo y, cada vez que recibe un evento, lo procesa y guarda en PostgreSQL.
7. El estado del sistema (API y Frontend) es monitoreado continuamente por UptimeRobot, que genera alertas en caso de inactividad o errores.

4. Seguridad y monitoreo

- Todas las comunicaciones de usuarios pasan por API Gateway, lo que permite controlar acceso y proteger la infraestructura backend.
- Auth0 asegura la autenticación de usuarios mediante emisión y validación de tokens.
- UptimeRobot garantiza la supervisión constante de la disponibilidad de servicios.