Documentación de la API de Gestión de Productos y Clientes

1. Introducción

Esta documentación proporciona una visión general de la API de Gestión de Productos y Clientes. La API permite la administración de recursos de productos y clientes en una aplicación web. Está diseñada utilizando Flask y SQLAlchemy, y sigue las mejores prácticas de codificación segura.

2. Arquitectura de la API

La API está compuesta por los siguientes componentes principales:

- **Cliente**: Interactúa con la API enviando solicitudes HTTP.

- **Servidor API**: Procesa las solicitudes del cliente y devuelve respuestas. Implementado con Flask.

- **Base de Datos**: Almacena la información de productos y clientes. Utiliza SQLAlchemy para la gestión de la base de datos.

- **Sistema de Registro (Logging)**: Registra eventos y errores para el monitoreo y la depuración.

- **Limitador de Tasa (Rate Limiter)**: Limita la cantidad de solicitudes que un cliente puede hacer a la API en un período de tiempo.

3. Endpoints

3.1. Productos

- **GET /products**: Obtiene una lista de todos los productos.

- **POST /products**: Crea un nuevo producto.

- Cuerpo de la Solicitud: {"name": "Nombre del producto", "price": 100.0, "quantity": 10}

- **GET /products/{id}**: Obtiene los detalles de un producto específico por ID.

- **PUT /products/{id}**: Actualiza un producto existente.

- Cuerpo de la Solicitud: {"name": "Nuevo nombre", "price": 150.0, "quantity": 20}

- **DELETE /products/{id}**: Elimina un producto por ID.

3.2. Clientes

- **GET /clients**: Obtiene una lista de todos los clientes.

- **POST /clients**: Crea un nuevo cliente.

- Cuerpo de la Solicitud: {"name": "Nombre del cliente", "email": "email@ejemplo.com", "phone": "+1234567890"}

- **GET /clients/{id}**: Obtiene los detalles de un cliente específico por ID.

- **PUT /clients/{id}**: Actualiza un cliente existente.

- Cuerpo de la Solicitud: {"name": "Nuevo nombre", "email": "nuevoemail@ejemplo.com", "phone": "+0987654321"}

- **DELETE /clients/{id}**: Elimina un cliente por ID.

4. Seguridad

La API implementa las siguientes medidas de seguridad:

- **Autenticación**: Utiliza tokens JWT para autenticar solicitudes.

- **Control de Acceso**: Define permisos para acceder a diferentes endpoints.

- **Limitación de Tasa**: Restringe el número de solicitudes por cliente para prevenir abusos.

- **Registro**: Mantiene un registro detallado de errores y eventos para su monitoreo y depuración.

5. Pruebas

La API incluye pruebas unitarias para todos los endpoints, garantizando que cada funcionalidad se comporte como se espera. Las pruebas se ejecutan utilizando el marco de pruebas de Python `unittest`.

6. Implementación

Para ejecutar la API, asegúrate de tener las dependencias instaladas y utiliza el siguiente comando:

```sh python app.py ```

Este comando iniciará el servidor de la API en `http://localhost:5000`.
```