TRABAJO FINAL

DESARROLLO BASADO EN COMPONENTES DISTRIBUIDOS Y SERVICIOS

Máster de TWCAM, ETSE-UV
Pablo Gómez Bolós y Diego Ruiz Sierra

Índice

1. Introducción..... 3
2. Diagrama del sistema..... 4
3. Diseño y justificación de decisiones..... 6
3.1. Microservicios.....6
3.1.1. Separación en microservicios..... 6
3.1.2. Estructura interna de cada API..... 6
3.2. Configuración y conexión de microservicios..... 9
3.2.1. config-server.....9
3.2.2. Peticiones HTTP..... 10
3.3. Seguridad en los microservicios.....11
3.3.1. Configuración de Keycloak embebido..... 11
3.3.2. Generación de tokens JWT.....14
3.3.3. Protección de endpoints según el rol..... 14
3.3.4. Uso de endpoints desde el ayuntamiento..... 15
4. Documentación de las APIs..... 17
5. Resultados..... 18
5.1. Ejecución del proyecto..... 18
5.2. Ejemplo de ejecución..... 18
5.3. Conclusión.....20
A. Anexo I: JSON resultante de OpenAPI para las bicicletas..... 21
B. Anexo II: JSON resultante de OpenAPI para la polución..... 32
C. Anexo III: JSON resultante de OpenAPI para el ayuntamiento..... 43

1. Introducción

En esta memoria se explica el proyecto final de la asignatura "Desarrollo Basado en Componentes Distribuidos y Servicios", realizado junto con otras dos asignaturas del máster de TWCAM. En este proyecto se ha desarrollado un conjunto de microservicios utilizando el *framework* Spring Boot con el objetivo de cubrir las necesidades de un ayuntamiento ficticio que necesita gestionar el estado de los aparcamientos de bicicletas en la ciudad, medir la polución ambiental próxima a estos aparcamientos y, posteriormente, agregar todos estos datos recogidos en unas bases de datos. Cada uno de estos microservicios está diseñado de forma independiente respetando la arquitectura basada en microservicios y se comunican entre sí a través de APIs REST y peticiones HTTP.

El objetivo de esta memoria es describir la arquitectura general de los microservicios y justificar las decisiones tomadas durante su diseño. Así, el sistema incluye:

- Una API de bicicletas, que gestiona la disponibilidad de los aparcamientos y los eventos relacionados.
- Una API de polución, que recopila y gestiona datos de calidad del aire.
- Una API de ayuntamiento, que combina la información de las dos anteriores y proporciona datos agregados.
- Servicios adicionales para:
    - Capas de acceso de datos para cada una de las APIs
    - Autenticación y seguridad, implementado utilizando tokens JWT integrados con un servidor Keycloak.
    - Configuración centralizada, utilizando Spring Cloud Config Server.

De esta forma, todas las APIs cuentan con documentación generada automáticamente mediante OpenAPI, lo que facilita su prueba utilizando Swagger.

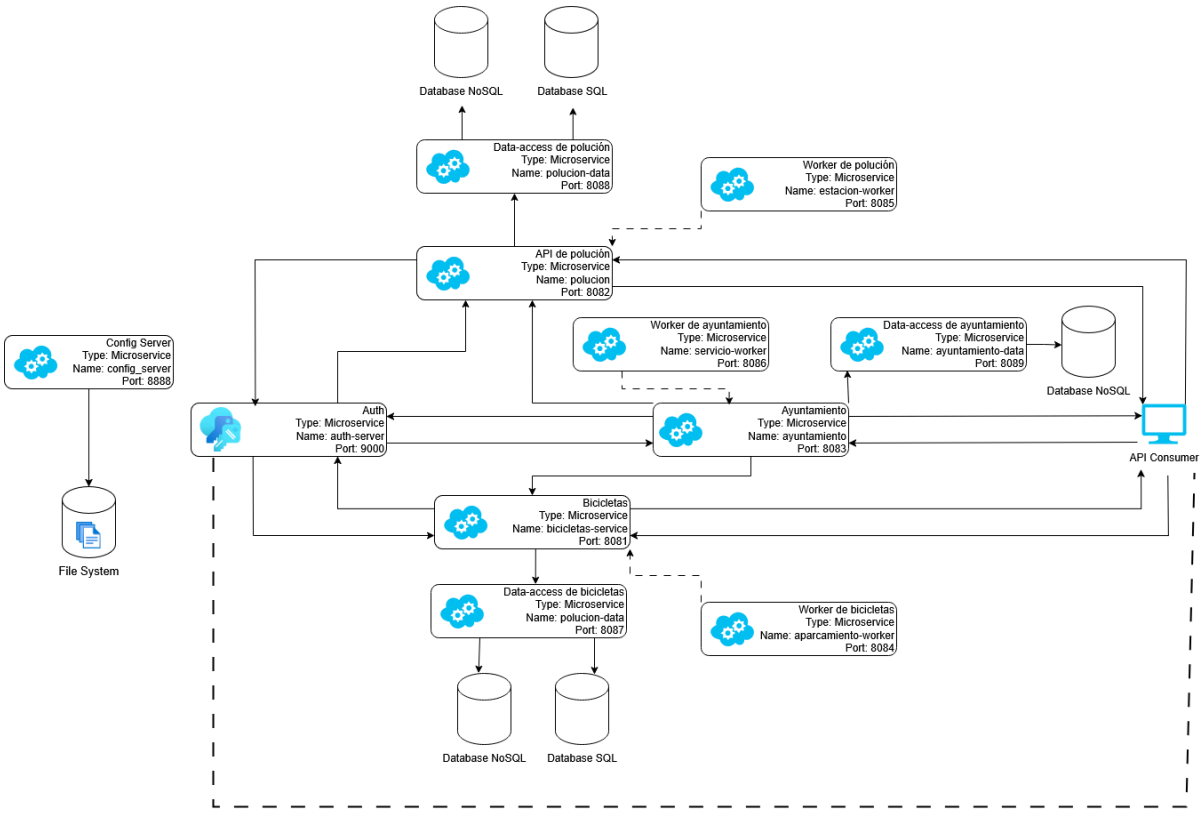
La memoria no aborda aspectos relacionados con:

- La persistencia de datos, detallada en la memoria de "Persistencia relacional y no relacional de datos"
- El balanceo de carga o escalado, detallada en la memoria de "Computación en la Nube". Debido a que se pedía para esta asignatura, se explica que estos aspectos se gestionan mediante el despliegue en Kubernetes, donde los pods de cada servicio pueden escalarse y balancearse mediante réplicas junto con un Ingress Controller encargado de gestionar el tráfico y un ClusterIP. Tampoco se ha visto necesario el uso de perfiles ya que la aplicación en Spring ya está lista para su ejecución en standalone y no es necesario cambiar configuraciones de distintos entornos, cosa que ya se hace en Cloud.

Finalmente, la memoria incluye también instrucciones para poder realizar un despliegue correcto de la aplicación junto con una prueba del uso de las APIs mediante un ejemplo de ejecución real que pone en práctica todas las funcionalidades básicas del sistema.

2. Diagrama del sistema

Se crea un diagrama del sistema para explicar su funcionamiento:



En el diagrama se puede observar que el sistema está formado por:

- Un consumidor del API que:
  - Se conecta una vez con el servidor de autenticación del *auth-service* localizado en el puerto 9000 para obtener los tokens necesarios para entrar a los diferentes endpoints que necesitan roles, que son “admin”, “estación”, “aparcamiento” y “servicio”.
  - Se comunica con el servicio del ayuntamiento (puerto 8083) para utilizar sus endpoints. También desde el ayuntamiento se conecta a los endpoints de bicicletas (puerto 8081) y estación (puerto 8082) para poder utilizarlos.
  - En los endpoints que se necesita verificación los servicios se conectan al *auth-service* para comprobar la identificación del usuario mediante el token utilizado.
- Bases de datos a las que se conectan cada uno de los servicios de las API, de la siguiente manera:
  - Todas las APIs se conectan a una capa de acceso de datos que separa las operaciones relacionadas con las BD de los endpoints de la aplicación. Estas capas se conectan directamente a las BD para obtener los datos que el microservicio principal necesita

    - La API de bicicletas y el de estación se conectan cada uno a una base de datos SQL y a una NoSQL.
    - La API del ayuntamiento se conecta únicamente a una base de datos NoSQL, que es donde guardará los datos agregados.
- A cada microservicio de las APIs se le asocia un *worker* que realizará peticiones cada pocos segundos a los endpoints que guardan datos en las bases de datos NoSQL.
- Un servicio *config-server* que centraliza toda la configuración de los servicios de las APIs.

3. Diseño y justificación de decisiones

En este apartado se desarrolla la arquitectura del proyecto. En todos los casos se justifican las decisiones tomadas.

3.1. Microservicios

3.1.1. Separación en microservicios

Como se ha podido ver en el punto anterior, el sistema está dividido en diferentes microservicios. Esta decisión, a parte de ser una exigencia del enunciado, responde a la necesidad de crear una arquitectura modular que permita el desarrollo independiente de cada uno de los componentes del sistema, simulando, por ejemplo, que cada API está diseñada por una empresa diferente e independiente de las demás. Así, cada microservicio está diseñado para cumplir con una responsabilidad específica:

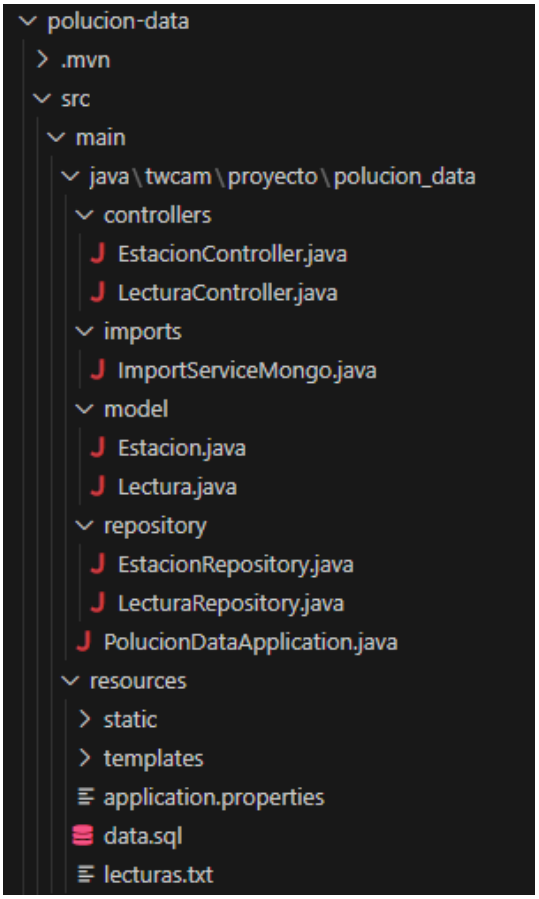
- El microservicio de bicicletas gestiona los datos relacionados con los aparcamientos de bicicletas y los eventos de uso asociados a estos aparcamientos.
- El microservicio de estaciones gestiona las estaciones de medición de la calidad del aire y sus lecturas.
- El microservicio de ayuntamiento actúa como un agregador de la información obtenida de los otros dos microservicios, proporcionando endpoints que permiten obtener datos integrados y ofreciendo funcionalidades adicionales de consulta. Desde esta API, además, se accederá a algunos endpoints de bicicletas y estaciones, ya que no podrán acceder por ellos mismos.
- Los microservicios de acceso de datos se encargan de separar los datos de las operaciones de las APIs.
- El servicio de autenticación (*auth-server*) se encarga de gestionar la seguridad y autenticación de los usuarios a través de JWT y Keycloak.
- El config-server centraliza la configuración de todos los servicios, asegurando su consistencia y facilitando la gestión de los entornos.

Esta separación garantiza la independencia de cada microservicio y que los cambios o nuevas funcionalidades en uno de ellos no afecten directamente a los demás. Además, facilita la escalabilidad: cada servicio puede crecer de forma independiente según la demanda, sin necesidad de redimensionar toda la aplicación (aunque como ya se ha dicho, esto se potenciará con Cloud).

3.1.2. Estructura interna de cada API

Cada uno de los microservicios implementados para las APIs (bicicletas, estaciones, ayuntamiento) está formado por tres capas bien diferenciadas, que a su vez están divididas en carpetas. Esta estructura modular permite separar claramente las responsabilidades de cada parte del sistema, organizándose de la siguiente forma para cada microservicio:

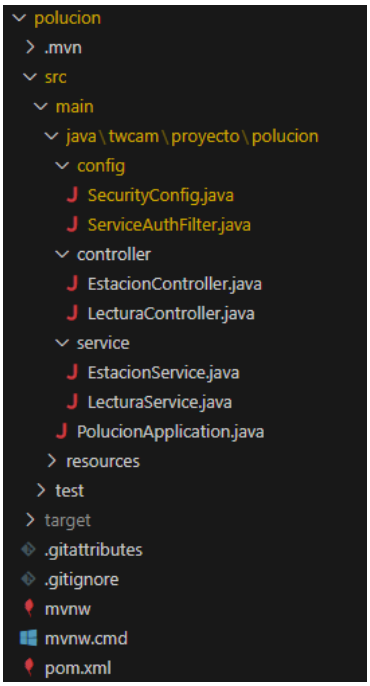
- La capa de acceso de datos, que consiste en proyectos Spring utilizados para leer e interpretar la información de la base de datos de forma que sea usable por el servicio principal. Está formada por los repositorios (uno para cada tabla) y por las clases (modelos) que forman la BD. También se encuentran las clases que se utilizan para importar los datos de Mongo, llamadas *ImportServiceMongo*. Esta capa se explica en detalle en la memoria de la asignatura "Persistencia relacional y no relacional de datos". Un ejemplo de la capa de acceso a datos en la API de polución:



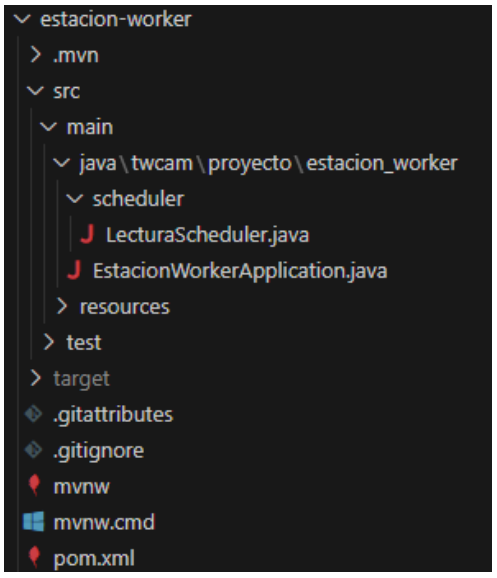
Sí que es relevante para esta asignatura el uso de los *controllers*, que generan endpoints a los que el microservicio principal se puede conectar para así poder acceder a los datos de la BD. Para llamar a estos endpoints se utiliza la tecnología Feign en el microservicio principal, que crea clientes que ejecutan estas operaciones para obtener los datos desde los servicios de la aplicación.

- El microservicio principal, formado por:
  - Servicios, que contienen la lógica de negocio y actúan como intermediarios entre los controladores y la capa de acceso a datos.
  - Controladores, que gestionan las peticiones HTTP y exponen los endpoints públicos.
  - Ficheros de configuración de seguridad. Detallados en el punto "*3.3. JWT y Keycloak*" junto con el *auth-server*.

Un ejemplo del microservicio principal de polución:



- Un servicio *worker* desarrollado con *@Scheduled* que se encarga de llamar automáticamente cada cierto tiempo a los endpoints que registran los eventos, lecturas y datos agregados en las bases de datos NoSQL, introduciendo datos aleatorios. Además de que son necesarios para que la aplicación tenga sentido, estos servicios se encuentran separados del microservicio principal para permitir que las llamadas que realizan se ejecuten de forma desacoplada de los endpoints que exponen las APIs. Se evita así que estos procesos puedan interferir en el rendimiento de los endpoints o bloqueen la atención a peticiones externas, pudiendo ejecutarse a una frecuencia diferente. Un ejemplo del *worker* de polución:



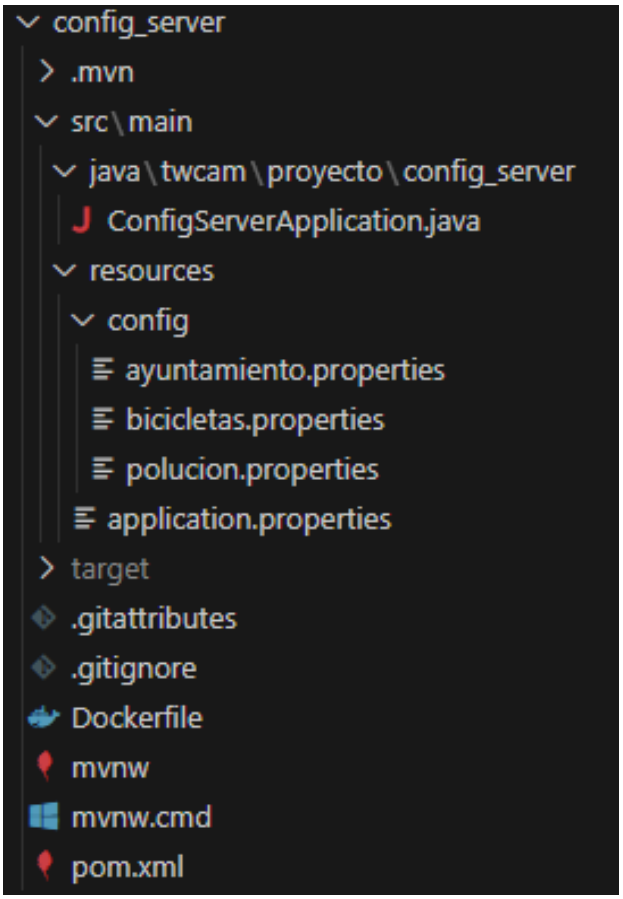
3.2. Configuración y conexión de microservicios

3.2.1. config-server

La configuración y la conexión de los distintos microservicios se ha gestionado utilizando Spring Cloud Config Server como mecanismo de configuración centralizada en todos los microservicios Este servicio se encarga de proporcionar a cada microservicio su archivo de configuración (llamado, dependiendo de la API, **.properties*), evitando configuraciones duplicadas. Se define así en la clase principal *ConfigServerApplication.java* con la anotación *EnableConfigServer*.

De esta forma, estos archivos de configuración (ubicados en la carpeta *resources/config* del *config-server*) permiten definir parámetros como los puertos, las credenciales de la base de datos o las URLs de servicios externos pudiendo así modificar las configuraciones más importantes en uno lugar y propagarse automáticamente a todos los microservicios en el momento de su arranque. Esto simplifica la gestión de la configuración y permite adaptar al programa rápidamente a cambios sin necesidad de reconfigurar manualmente cada componente.

Esta es la estructura del *config-server* en el proyecto:



En los *properties* del servidor de configuración se define que escucha en el puerto 8888, así:

```
spring.profiles.active=native
spring.application.name=config_server
server.port=8888
spring.cloud.config.server.native.search-locations=classpath:/config
```

De esta forma, en cada microservicio sólo se debe crear un fichero *properties* que le indique al servicio que debe escuchar el puerto 8888 para saber su configuración junto con su nombre de Spring y su puerto.

3.2.2. Peticiones HTTP

Por otro lado, la comunicación entre microservicios se realiza a través de peticiones HTTP REST mediante el servicio de ayuntamiento como orquestador, que consulta las APIs de bicicletas y estaciones para agregar la información y exponerla de manera integrada. Esta conexión se implementa utilizando el cliente RestTemplate de Spring, que permite enviar solicitudes HTTP a los otros microservicios y procesar sus respuestas de forma sencilla.

La configuración de este RestTemplate (ubicado en la carpeta *config* del ayuntamiento) se detalla en el punto "3.3. JWT y Keycloak" ya que se ha establecido que sólo se puede acceder a algunos endpoints de las bicicletas y estaciones si vienen del ayuntamiento, por lo que en este RestTemplate se añade unas cabeceras a las peticiones indicando que vienen del ayuntamiento, junto con un token de autenticación interno.

3.3. Seguridad en los microservicios

Se explica ahora el uso del *auth-server* para garantizar la seguridad de los microservicios, la generación de tokens y el correcto uso de los roles y permisos, que se ha basado en la utilización de Keycloak como servidor de autenticación y autorización: en lugar de desplegar Keycloak como un servicio independiente, se ha optado por embederlo directamente en la aplicación utilizando Keycloak Server. De este modo, el propio proyecto contiene su servidor de autorización y no se necesita un servicio externo como Docker. Podría decirse entonces que se ha creado un "mini Keycloak" que se despliega al desplegar la aplicación y que se usa para emitir y validar tokens JWT.

Se ha utilizado esta tecnología porque se ha considerado una herramienta poderosa que permitía ahorrar mucho trabajo, además de que aporta claridad al código. También hace bastante sencilla la gestión de usuarios y tokens mediante los realms, por lo que se consideró que era más sencillo que crear un sistema de autenticación desde cero.

3.3.1. Configuración de Keycloak embebido

Este servidor de Keycloak está configurado para ejecutarse en el puerto 9000, exponiendo sus endpoints en la ruta */auth*. Esta ruta y el resto de la configuración se define principalmente en el archivo *application.yml*, donde se especifica el puerto, la base de datos (en este caso, una base de datos H2 en memoria) y las credenciales del administrador:

```yaml
server:
  port: 9000

spring:
  datasource:
    username: sa
    url: jdbc:h2:mem:testdb

keycloak:
  server:
    contextPath: /auth
    adminUser:
      username: admin
      password: password
```

Sabiendo esto, se resume en una tabla las funciones de todas las clases que se utilizan en el *auth-server* para configurar Keycloak (página 11).

Utilizando todas estas clases se consigue que, cuando la aplicación se inicia, se creen automáticamente el realm y el usuario administrador, garantizando que Keycloak esté listo para emitir y validar los tokens JWT necesarios para autorizar las peticiones a los distintos microservicios. De esta manera, Keycloak embebido actúa como el punto central de autenticación de todo el sistema.

Clase	**Función principal**	**Otras funciones**
AuthorizationServerApp	Clase principal que arranca la aplicación en SpringBoot	- Importa la configuración del KeycloakServerProperties - Imprime por consola la URL del servidor si el despliegue ha sido correcto
EmbeddedKeycloakApplication	Configura Keycloak durante la inicialización	- Crea el usuario administrador con sus credenciales - Carga la configuración del realm mediante un archivo JSON
EmbeddedKeycloakConfig	Registra el servlet principal y los filtros necesarios para el funcionamiento interno de Keycloak	Crea un entorno ficticio para que Keycloak pueda inyectar el datasource de H2
EmbeddedKeycloakRequestFilter	Filtro que gestiona las sesiones y la codificación de las peticiones en la capa REST de Keycloak a */auth/***.	
KeycloakServerProperties	Clase de configuración que define las propiedades del servidor de Keycloak	Define el *context path*, las credenciales y el archivo del realm. Es de las clases más importantes
Resteasy3Provider, SimplePlatformProvider y RegularJsonConfigProviderFactory	Clases de integración que permiten que Keycloak funcione correctamente como parte de la aplicación de Spring Boot.	

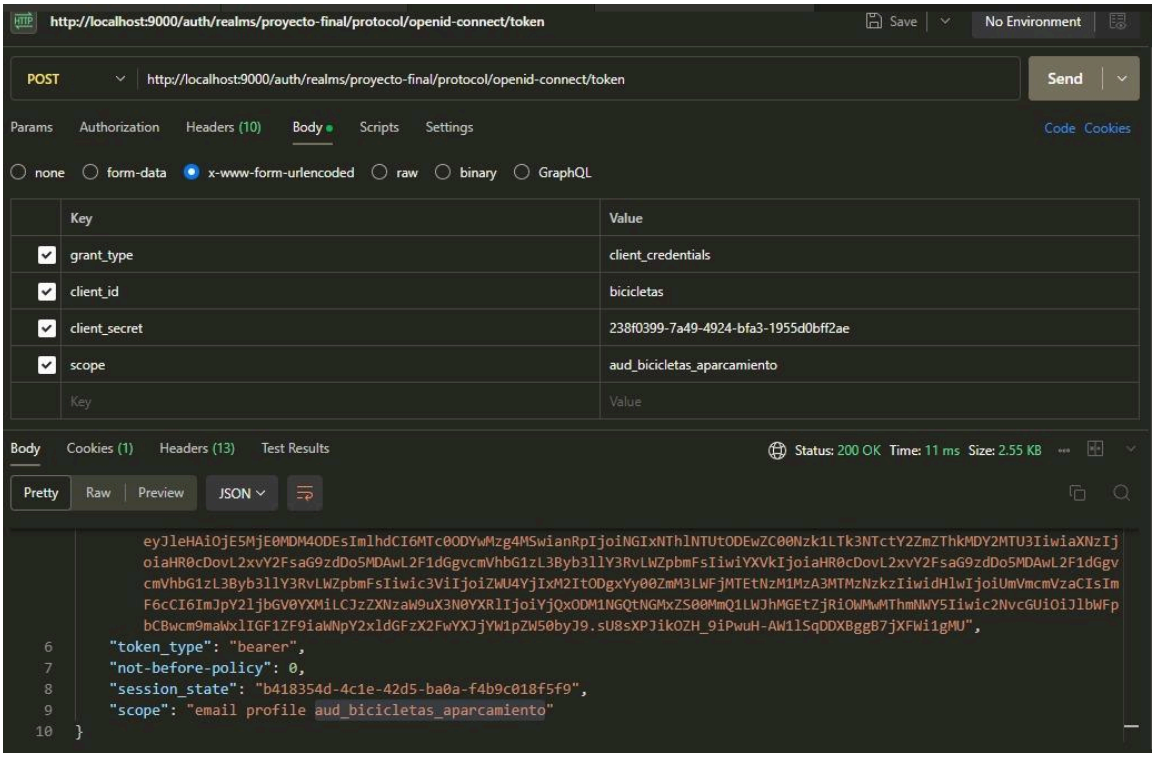
3.3.2. Generación de tokens JWT

Una vez desplegado el servidor de Keycloak embebido, se puede proceder a la generación de tokens JWT que permitirán acceder a los endpoints protegidos de las APIs mediante el endpoint POST */auth/realms/proyecto-final/protocol/openid-connect/token*

Así, para generar un token se envía una petición con los siguientes parámetros:

- *grant_type*: el tipo de concesión, en este caso *client_credentials* debido a que se utiliza autenticación basada en clientes.
- *client_id*: el identificador del cliente configurado en el realm de Keycloak (por ejemplo, bicicletas, polucion o ayuntamiento).
- *client_secret*: la clave secreta generada por Keycloak para cada cliente.
- *scope*: los permisos específicos para el token, por ejemplo, *aud_bicicletas_aparcamiento* para el cliente de bicicletas.

En la siguiente imagen se muestra un ejemplo de creación de un token para la API de bicicletas y el rol "aparcamiento". Se puede observar que se pasan los parámetros necesarios y cómo se devuelve el JWT junto con otros elementos en la respuesta:



3.3.3. Protección de endpoints según el rol

Una vez generados los tokens JWT es necesario garantizar que cada microservicio permita el acceso únicamente a los endpoints según el rol que tenga el cliente que hace la petición. Para ello, cada uno de los microservicios incluye una configuración de seguridad propia basada en Spring Security.

Esta configuración se define en la clase *SecurityConfig* de cada microservicio, como por ejemplo en la API de polución:

```java
@Configuration
@EnableMethodSecurity
public class SecurityConfig {

    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
        http
                .authorizeHttpRequests(auth -> auth
                        .requestMatchers("/api/v1/api-spec/**").permitAll()
                        .requestMatchers("/api/v1/swagger-ui/**").permitAll()

                        .requestMatchers("/estaciones").permitAll()
                        .requestMatchers("/estacion/*/status**").permitAll()

                        .requestMatchers(HttpMethod.POST, "/estacion/*").hasAuthority("SCOPE_aud_polucion_estacion")

                        .requestMatchers("/estacion").hasAuthority("SCOPE_aud_ayuntamiento_admin")
                        .requestMatchers("/estacion/*").hasAuthority("SCOPE_aud_ayuntamiento_admin")

                        .anyRequest().authenticated())
                .oauth2ResourceServer(oauth2 -> oauth2.jwt());

        return http.build();
    }
}
```

En este ejemplo (replicable en las demás APIs) se puede ver que se establecen unas reglas utilizando el método *authorizeHttpRequests*. Las reglas son las siguientes:

- Endpoints públicos: algunos endpoints, como los de la documentación (*/api/v1/api-spec/*** y */api/v1/swagger-ui/***) o ciertos GET como */estaciones*, se exponen de forma pública sin requerir autenticación.
- Endpoints protegidos por rol: para otros endpoints es necesario que el cliente tenga un token JWT válido con un scope específico. Por ejemplo:
    - "SCOPE_aud_polucion_estacion" controla el rol “estacion” para el POST del endpoint */estacion/**.
    - "SCOPE_aud_ayuntamiento_admin" controla el rol “admin” para las operaciones que debe poder realizar únicamente el ayuntamiento.

Al final del método se puede ver que la configuración se completa con la integración de Spring Security como un resource server OAuth2, indicando que se validarán los tokens JWT emitidos por el servidor de Keycloak.

3.3.4. Uso de endpoints desde el ayuntamiento

El sistema implementa un mecanismo de autenticación interna entre microservicios debido a que algunos endpoints de las APIs de bicicletas y polución están diseñados para ser accedidos exclusivamente mediante el rol “admin” mientras se utiliza el microservicio del ayuntamiento, que actúa como un orquestador.

Para implementar esta autenticación el microservicio de ayuntamiento utiliza un cliente *RestTemplate*, configurado en la clase *RestTemplateConfig* dentro de la API del ayuntamiento como se puede ver en la imagen:

```java
@Configuration
public class RestTemplateConfig {

    @Value("${app.service.secret}")
    private String serviceSecret;

    @Bean
    public RestTemplate restTemplate() {
        RestTemplate restTemplate = new RestTemplate();
        restTemplate.setInterceptors(List.of(new ServiceAuthInterceptor()));
        return restTemplate;
    }

    public class ServiceAuthInterceptor implements ClientHttpRequestInterceptor {

        @Override
        public ClientHttpResponse intercept(
                HttpRequest request,
                byte[] body,
                ClientHttpRequestExecution execution) throws IOException {

            request.getHeaders().add("X-Service-Auth", serviceSecret);
            request.getHeaders().add("X-Service-Name", "ayuntamiento");

            return execution.execute(request, body);
        }
    }
}
```

En esta configuración se añade un *ServiceAuthInterceptor* que inserta dos cabeceras personalizadas en cada petición:

- *X-Service-Auth*: un token compartido que permite identificar y autenticar al microservicio emisor (que siempre será el ayuntamiento). Este se declara en los properties de cada API.
- *X-Service-Name*: un identificador del nombre del servicio emisor, en este caso "ayuntamiento".

Por su parte, las APIs de bicicletas y polución incluyen un filtro específico (ServiceAuthFilter) que intercepta las peticiones entrantes y valida el contenido de la cabecera X-Service-Auth. En el ejemplo del punto anterior era el “SCOPE_aud_ayuntamiento_admin“. Si el valor recibido no coincide con el secreto que tienen configurado, la petición se rechaza con un error 401 Unauthorized.

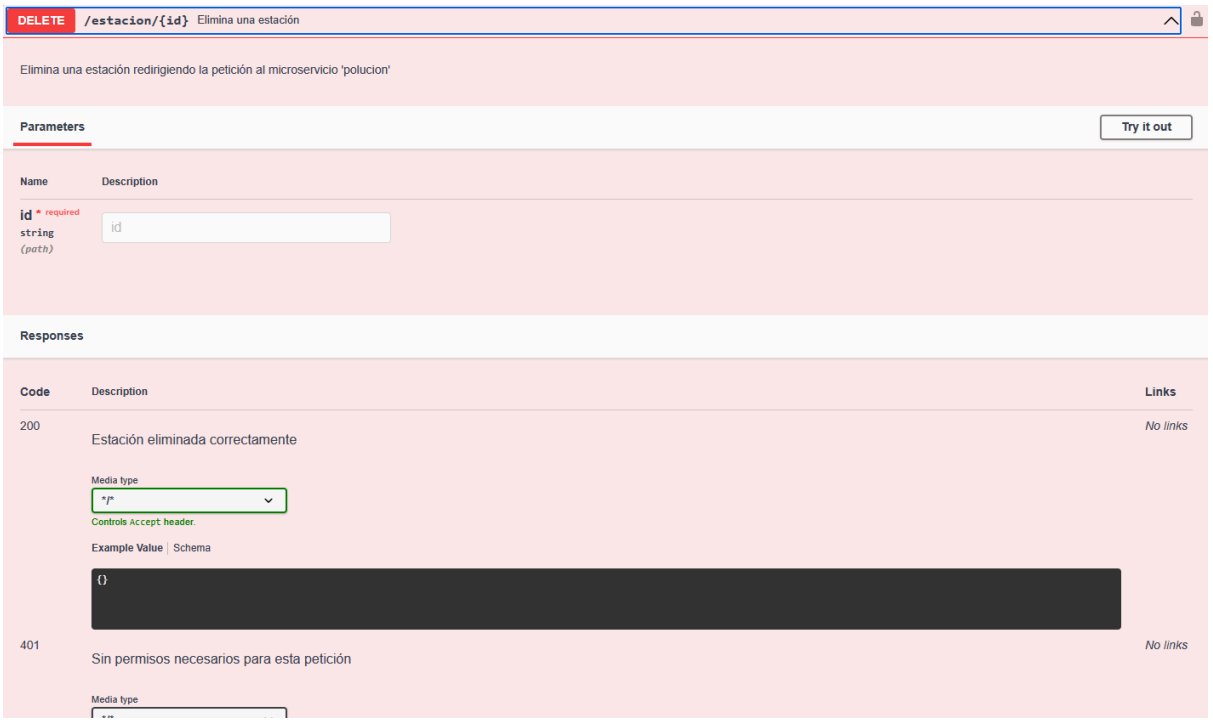
4. Documentación de las APIs

Para facilitar la comprensión de las APIs desarrolladas se ha integrado la herramienta OpenAPI. Gracias a esta integración, la especificación de cada API se genera automáticamente y se expone en formato JSON en un endpoint accesible públicamente en la ruta */api/v1/api-spec* (los JSON generados para las APIs de este proyecto se encuentran en los Anexos I, II y III y en la carpeta README del proyecto).

Esta documentación describe de forma estructurada todos los endpoints, los parámetros de entrada, las respuestas esperadas, los posibles códigos de estado HTTP que pueden devolver y la posible autentificación en caso de que sea necesaria. Para ello se usan varios parámetros dentro de anotaciones como *@Operation* o *@ApiResponse*, como se puede ver en este ejemplo:

```java
@PostMapping("/estacion")
@Operation(summary = "Crea una estación", description = "Crea una estación redirigiendo la petición al microservicio 'polucion'", tags = {
        "Operaciones que necesitan el rol 'admin'" }, security = @SecurityRequirement(name = "bearerAuth"))
@ApiResponse(responseCode = "201", description = "Estación creada correctamente")
@ApiResponse(responseCode = "400", description = "Faltan campos obligatorios como 'id' o 'dirección'")
@ApiResponse(responseCode = "401", description = "Sin permisos necesarios para esta petición")
@ApiResponse(responseCode = "409", description = "Ya existe una estación con el id indicado")
public ResponseEntity<?> crearEstacion(@RequestBody Estacion estacion,
        @Parameter(hidden = true) @RequestHeader("Authorization") String authHeader) {
    return ayuntamientoService.crearEstacion(estacion, authHeader);
}
```

Además, se ha habilitado la interfaz de usuario de Swagger UI en la ruta */api/v1/swagger-ui*, que ofrece una representación visual e interactiva de los endpoints. El ejemplo anterior se ve así:



5. Resultados

Después de todo el desarrollo, se muestra el proyecto funcional y en ejecución.

5.1. Ejecución del proyecto

Para ejecutar el proyecto se debe tener activo MySQL Workbench y MongoDB con Docker para después ejecutar un *docker-compose* que creará las bases de datos correspondientes. Así, se despliegan los microservicios de la siguiente manera y en el siguiente orden:

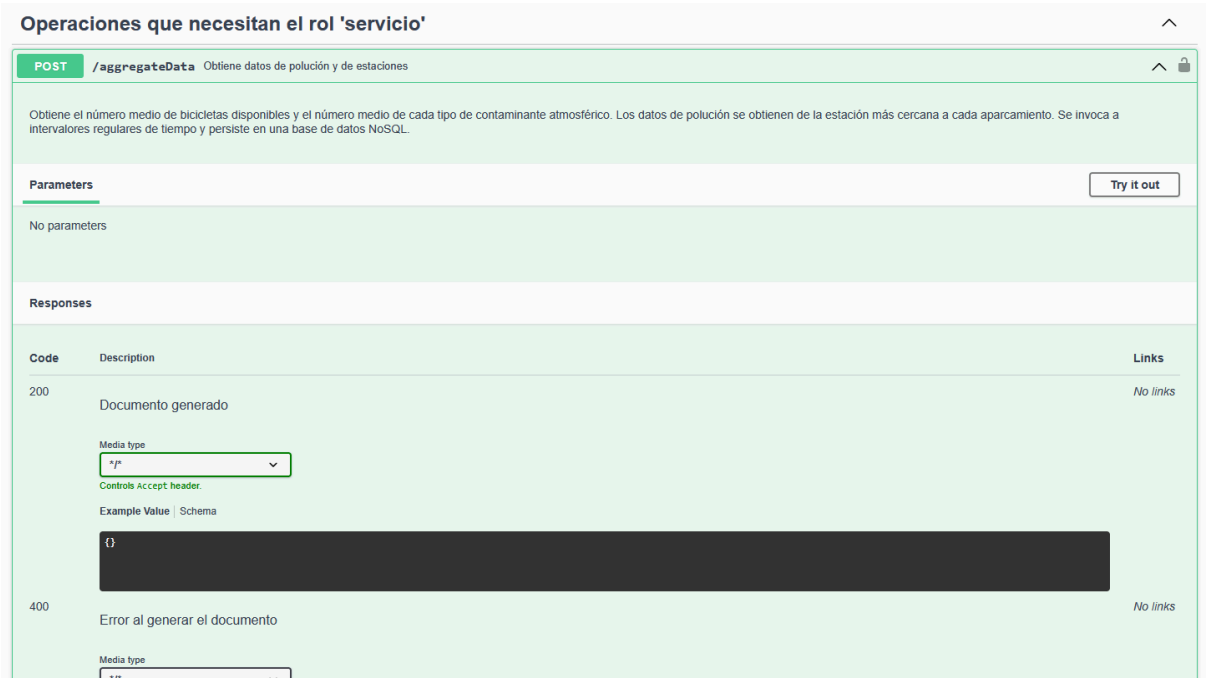
- Para desplegar las bases de datos se puede utilizar el *docker-compose* ubicado en la carpeta *cloud* del proyecto, con la siguiente orden*:*

```
docker-compose -f docker-compose-mysql-mongo.yml up -d
```

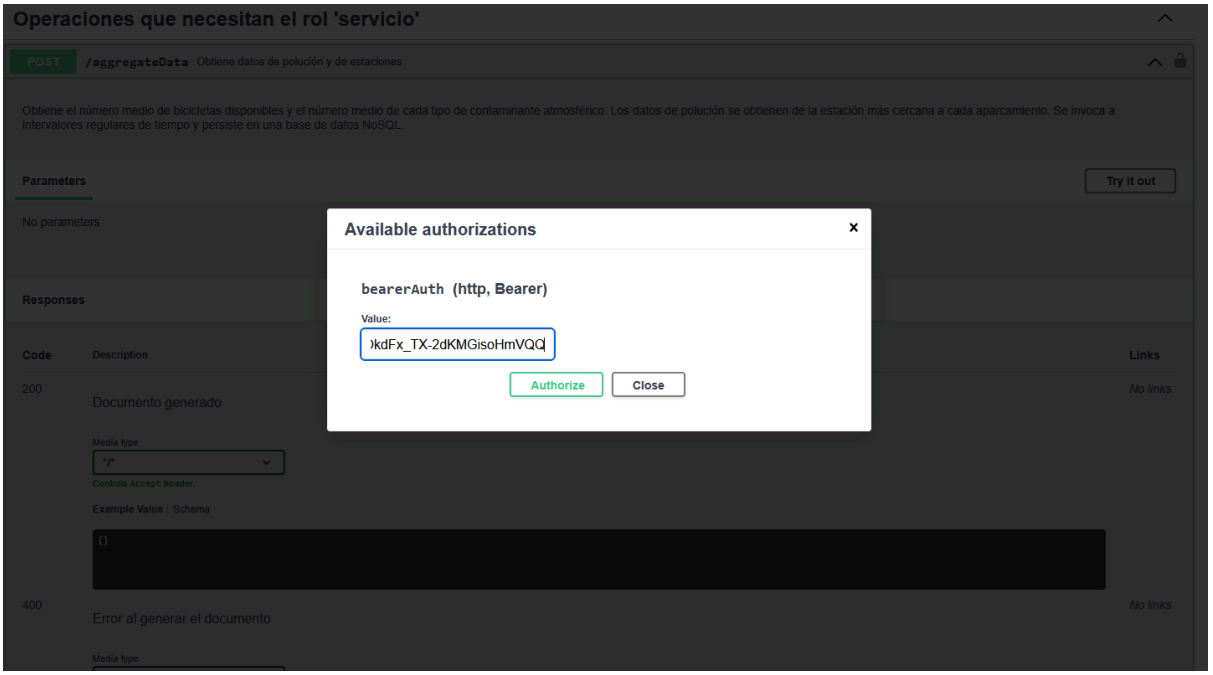
- Ahora se despliegan en Spring, primero *authorization-server* y *config_server*.
- Luego, *bicicletas*, *bicicletas-data*, *polucion* y *polucion-data*.
- Después, *aparcamiento-worker, estacion-worker, ayuntamiento* y *ayuntamiento-data*.
- Y por último, *servicio-worker*.

5.2. Ejemplo de ejecución

Se muestra ahora un ejemplo de ejecución de un endpoint utilizando Swagger para demostrar el correcto funcionamiento de la aplicación. Se va a ejecutar el endpoint POST */aggregateData* ya que se considera el más importante porque utiliza endpoints de las bicicletas, de polución y del propio ayuntamiento. Así se ve el endpoint en Swagger:



Este endpoint necesita la verificación del rol "servicio", por lo que se introduce el JWT correspondiente en el dialog que se abre al pulsar el candado que se ve en la anterior foto, en la esquina superior derecha.



Como se puede ver en la imagen, se ejecuta la petición y se devuelve un código 200 OK junto con los documentos que se han subido a la base de datos de Mongo:

Todas las APIs se pueden probar en Swagger: la de bicicletas, la de polución y la del ayuntamiento.

5.3. Conclusión

Se ha conseguido realizar el proyecto de forma satisfactoria gracias a las diferentes tecnologías utilizadas, que incluyen la arquitectura modular de Spring Boot, la autenticación centralizada con Keycloak embebido y la configuración externa mediante Spring Cloud Config, entre otras. Así el proyecto ofrece una solución escalable y segura integrada con bases de datos SQL y NoSQL, documentada con OpenAPI.

A. Anexo I: JSON resultante de OpenAPI para las bicicletas


```
{
    "openapi": "3.0.1",
    "info": {
        "title": "API de bicicletas",
        "description": "API que contiene todas las operaciones necesarias para
la gestión de las bicicletas",
        "contact": {
            "name": "Pablo Gómez/Diego Ruiz",
            "email": "pagobo2@alumni.uv.es/dieruiz4@alumni.uv.es"
        },
        "license": {
            "name": "Apache 2.0",
            "url": "https://www.apache.org/licenses/LICENSE-2.0"
        },
        "version": "v1"
    },
    "servers": [
        {
            "url": "/",
            "description": "Production"
        }
    ],
    "paths": {
        "/aparcamiento/{id}": {
            "put": {
                "tags": [
                    "Operaciones accesibles desde la API del Ayuntamiento"
                ],
                "summary": "Update parking",
                "description": "Modifica un parking",
                "operationId": "update",
                "parameters": [
                    {
                        "name": "id",
                        "in": "path",
                        "required": true,
                        "schema": {
                            "type": "string"
```

```
                        }
                    }
                ],
                "requestBody": {
                    "content": {
                        "application/json": {
                            "schema": {
                                "$ref": "#/components/schemas/Parking"
                            }
                        }
                    },
                    "required": true
                },
                "responses": {
                    "404": {
                        "description": "No existe una parking con el id
indicado",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "400": {
                        "description": "Faltan campos obligatorios en la
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "401": {
                        "description": "Sin permisos necesarios para esta
petición",
                        "content": {
                            "*/*": {
                                "schema": {
```

```
                                "type": "object"
                            }
                        }
                    }
                },
                "200": {
                    "description": "Parking actualizado correctamente",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                }
            }
        },
        "delete": {
            "tags": [
                "Operaciones accesibles desde la API del Ayuntamiento"
            ],
            "summary": "Delete parking",
            "description": "Elimina un parking pasado un id",
            "operationId": "delete",
            "parameters": [
                {
                    "name": "id",
                    "in": "path",
                    "required": true,
                    "schema": {
                        "type": "string"
                    }
                }
            ],
            "responses": {
                "401": {
                    "description": "Sin permisos necesarios para esta
petición",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
```

```
                            }
                        }
                    }
                },
                "404": {
                    "description": "No se encontró el parking con ese id",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "200": {
                    "description": "Parking eliminado correctamente",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                }
            }
        },
        "/evento/{id}": {
            "post": {
                "tags": [
                    "Operaciones que necesitan el rol 'aparcamiento'"
                ],
                "summary": "Add evento",
                "description": "Añade un nuevo evento a un aparcamiento
determinado",
                "operationId": "crearEvento",
                "parameters": [
                    {
                        "name": "id",
                        "in": "path",
                        "required": true,
                        "schema": {
```

```
                            "type": "string"
                        }
                    }
                ],
                "requestBody": {
                    "content": {
                        "application/json": {
                            "schema": {
                                "$ref": "#/components/schemas/OperacionDTO"
                            }
                        }
                    },
                    "required": true
                },
                "responses": {
                    "404": {
                        "description": "No se encontró el aparcamiento",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "400": {
                        "description": "Operación no válida",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "401": {
                        "description": "Sin permisos necesarios para esta
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
```

```
                        }
                    }
                }
            },
            "201": {
                "description": "Evento creado correctamente",
                "content": {
                    "*/*": {
                        "schema": {
                            "type": "object"
                        }
                    }
                }
            }
        },
        "security": [
            {
                "bearerAuth": []
            }
        ]
    }
},
"/aparcamiento": {
    "post": {
        "tags": [
            "Operaciones accesibles desde la API del Ayuntamiento"
        ],
        "summary": "Add parking",
        "description": "Añade un nuevo aparcamiento",
        "operationId": "add",
        "requestBody": {
            "content": {
                "application/json": {
                    "schema": {
                        "$ref": "#/components/schemas/Parking"
                    }
                }
            },
            "required": true
        },
        "responses": {
            "401": {
```

```
                    "description": "Sin permisos necesarios para esta
petición",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "201": {
                    "description": "Aparcamiento creado correctamente",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "400": {
                    "description": "Faltan campos obligatorios como 'id' o
'dirección'",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "409": {
                    "description": "Ya existe un aparcamiento con el id
indicado",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                }
```

```
                }
            }
        },
        "/aparcamientos": {
            "get": {
                "tags": [
                    "Operaciones públicas"
                ],
                "summary": "Get parkings",
                "description": "Muestra todos los aparcamientos",
                "operationId": "listAll",
                "responses": {
                    "200": {
                        "description": "OK",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    }
                }
            }
        },
        "/aparcamiento/{id}/status": {
            "get": {
                "tags": [
                    "Operaciones públicas"
                ],
                "summary": "Get events by dates",
                "description": "Muestra los cambios de estado de una parada en
un cierto espacio de tiempo",
                "operationId": "statusParking_1",
                "parameters": [
                    {
                        "name": "id",
                        "in": "path",
                        "required": true,
                        "schema": {
                            "type": "string"
                        }
```

```
                    },
                    {
                        "name": "from",
                        "in": "query",
                        "required": true,
                        "schema": {
                            "type": "string",
                            "format": "date-time"
                        }
                    },
                    {
                        "name": "to",
                        "in": "query",
                        "required": true,
                        "schema": {
                            "type": "string",
                            "format": "date-time"
                        }
                    }
                ],
                "responses": {
                    "200": {
                        "description": "OK",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    }
                }
            }
        },
        "/aparcamiento/available": {
            "get": {
                "tags": [
                    "Operaciones públicas"
                ],
                "summary": "Top 10 parkings with free bikes",
                "description": "Consulta los 10 parkings con más bicis
disponibles en este momento",
```

```
            "operationId": "top10Ahora",
            "responses": {
                "200": {
                    "description": "OK",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                }
            }
        }
    },
    "components": {
        "schemas": {
            "Parking": {
                "type": "object",
                "properties": {
                    "idparking": {
                        "type": "string"
                    },
                    "direction": {
                        "type": "string"
                    },
                    "bikesCapacity": {
                        "type": "integer",
                        "format": "int32"
                    },
                    "latitude": {
                        "type": "number",
                        "format": "float"
                    },
                    "longitude": {
                        "type": "number",
                        "format": "float"
                    }
                }
            },
            "OperacionDTO": {
```

```
                "type": "object",
                "properties": {
                    "operation": {
                        "type": "string"
                    }
                }
            }
        },
        "securitySchemes": {
            "bearerAuth": {
                "type": "http",
                "scheme": "bearer",
                "bearerFormat": "JWT"
            }
        }
    }
}
```

B. Anexo II: JSON resultante de OpenAPI para la polución


```
{
    "openapi": "3.0.1",
    "info": {
        "title": "API de polución",
        "description": "API que contiene todas las operaciones necesarias para
la gestión de las estaciones de medición de polución",
        "contact": {
            "name": "Pablo Gómez/Diego Ruiz",
            "email": "pagobo2@alumni.uv.es/dieruiz4@alumni.uv.es"
        },
        "license": {
            "name": "Apache 2.0",
            "url": "https://www.apache.org/licenses/LICENSE-2.0"
        },
        "version": "v1"
    },
    "servers": [
        {
            "url": "/",
            "description": "Production"
        }
    ],
    "paths": {
        "/estacion/{id}": {
            "put": {
                "tags": [
                    "Operaciones accesibles desde la API del Ayuntamiento"
                ],
                "summary": "Modifica una estación existente",
                "description": "Actualiza los datos (dirección, latitud y
longitud) de una estación de medición identificada por su id",
                "operationId": "actualizarEstacion",
                "parameters": [
                    {
                        "name": "id",
                        "in": "path",
                        "required": true,
                        "schema": {
```

```
                            "type": "string"
                        }
                    }
                ],
                "requestBody": {
                    "content": {
                        "application/json": {
                            "schema": {
                                "$ref": "#/components/schemas/Estacion"
                            }
                        }
                    },
                    "required": true
                },
                "responses": {
                    "400": {
                        "description": "Faltan campos obligatorios en la
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "401": {
                        "description": "Sin permisos necesarios para esta
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "404": {
                        "description": "No existe una estación con el id
indicado",
                        "content": {
                            "*/*": {
```

```
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "200": {
                    "description": "Estación actualizada correctamente",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                }
            }
        },
        "post": {
            "tags": [
                "Operaciones que necesitan el rol 'estacion'"
            ],
            "summary": "Registra una nueva lectura de sensores",
            "description": "Guarda una lectura de calidad del aire enviada
por una estación de medición",
            "operationId": "registrarLectura",
            "parameters": [
                {
                    "name": "id",
                    "in": "path",
                    "required": true,
                    "schema": {
                        "type": "string"
                    }
                }
            ],
            "requestBody": {
                "content": {
                    "application/json": {
                        "schema": {
                            "$ref": "#/components/schemas/Lectura"
                        }
```

```
                        }
                    },
                    "required": true
                },
                "responses": {
                    "201": {
                        "description": "Lectura registrada correctamente",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "401": {
                        "description": "Sin permisos necesarios para esta
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "400": {
                        "description": "Petición mal escrita",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "404": {
                        "description": "La estación con ese id no existe",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
```

```
                            }
                        }
                    }
                }
            },
            "security": [
                {
                    "bearerAuth": []
                }
            ]
        },
        "delete": {
            "tags": [
                "Operaciones accesibles desde la API del Ayuntamiento"
            ],
            "summary": "Elimina una estación de medición",
            "description": "Elimina una estación de medición de la base de
datos según su id",
            "operationId": "eliminarEstacion",
            "parameters": [
                {
                    "name": "id",
                    "in": "path",
                    "required": true,
                    "schema": {
                        "type": "string"
                    }
                }
            ],
            "responses": {
                "401": {
                    "description": "Sin permisos necesarios para esta
petición",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "200": {
```

```
                    "description": "Estación eliminada correctamente",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "404": {
                    "description": "No se encontró la estación con ese
id",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                }
            }
        },
        "/estacion": {
            "post": {
                "tags": [
                    "Operaciones accesibles desde la API del Ayuntamiento"
                ],
                "summary": "Añade una nueva estación de medición",
                "description": "Permite al administrador registrar una nueva
estación de medición con su id, dirección, latitud y longitud",
                "operationId": "crearEstacion",
                "requestBody": {
                    "content": {
                        "application/json": {
                            "schema": {
                                "$ref": "#/components/schemas/Estacion"
                            }
                        }
                    },
                    "required": true
                },
```

```
                "responses": {
                    "409": {
                        "description": "Ya existe una estación con el id
indicado",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "201": {
                        "description": "Estación creada correctamente",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "401": {
                        "description": "Sin permisos necesarios para esta
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "400": {
                        "description": "Faltan campos obligatorios como 'id' o
'dirección'",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
```

```
                        }
                    }
                }
            }
        },
        "/estaciones": {
            "get": {
                "tags": [
                    "Operaciones públicas"
                ],
                "summary": "Obtiene todas las estaciones",
                "description": "Obtiene un listado con todas las estaciones de
medición de la aplicación junto con sus datos",
                "operationId": "getAllEstaciones",
                "responses": {
                    "200": {
                        "description": "Devuelve el listado de estaciones",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "array",
                                    "items": {
                                        "$ref":
"#/components/schemas/Estacion"
                                    }
                                }
                            }
                        }
                    }
                }
            }
        },
        "/estacion/{id}/status": {
            "get": {
                "tags": [
                    "Operaciones públicas"
                ],
                "summary": "Obtener la última lectura o las lecturas por
intervalo",
                "description": "Devuelve la última lectura de una estación o
todas las lecturas en un intervalo si se pasan parámetros 'from' y 'to'",
                "operationId": "obtenerLecturas",
```

```
            "parameters": [
                {
                    "name": "id",
                    "in": "path",
                    "required": true,
                    "schema": {
                        "type": "integer",
                        "format": "int32"
                    }
                },
                {
                    "name": "from",
                    "in": "query",
                    "required": false,
                    "schema": {
                        "type": "string"
                    }
                },
                {
                    "name": "to",
                    "in": "query",
                    "required": false,
                    "schema": {
                        "type": "string"
                    }
                }
            ],
            "responses": {
                "400": {
                    "description": "Fechas mal escritas",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "200": {
                    "description": "Lecturas devueltas correctamente",
                    "content": {
                        "*/*": {
```

```
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "404": {
                    "description": "No se encontraron lecturas",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                }
            }
        }
    },
    "components": {
        "schemas": {
            "Estacion": {
                "type": "object",
                "properties": {
                    "id": {
                        "type": "string"
                    },
                    "direccion": {
                        "type": "string"
                    },
                    "latitud": {
                        "type": "number",
                        "format": "float"
                    },
                    "longitud": {
                        "type": "number",
                        "format": "float"
                    }
                }
            },
            "Lectura": {
```

```
                "type": "object",
                "properties": {
                    "id": {
                        "type": "integer",
                        "format": "int32"
                    },
                    "timeStamp": {
                        "type": "string",
                        "format": "date-time"
                    },
                    "nitricOxides": {
                        "type": "number",
                        "format": "float"
                    },
                    "nitrogenDioxides": {
                        "type": "number",
                        "format": "float"
                    },
                    "VOCs_NMHC": {
                        "type": "number",
                        "format": "float"
                    },
                    "PM2_5": {
                        "type": "number",
                        "format": "float"
                    }
                }
            }
        },
        "securitySchemes": {
            "bearerAuth": {
                "type": "http",
                "scheme": "bearer",
                "bearerFormat": "JWT"
            }
        }
    }
}
```

C. Anexo III: JSON resultante de OpenAPI para el ayuntamiento


```
{
    "openapi": "3.0.1",
    "info": {
        "title": "API de ayuntamiento",
        "description": "API que contiene todas las operaciones necesarias para
la gestión del ayuntamiento",
        "contact": {
            "name": "Pablo Gómez/Diego Ruiz",
            "email": "pagobo2@alumni.uv.es/dieruiz4@alumni.uv.es"
        },
        "license": {
            "name": "Apache 2.0",
            "url": "https://www.apache.org/licenses/LICENSE-2.0"
        },
        "version": "v1"
    },
    "servers": [
        {
            "url": "/",
            "description": "Production"
        }
    ],
    "paths": {
        "/estacion/{id}": {
            "put": {
                "tags": [
                    "Operaciones que necesitan el rol 'admin'"
                ],
                "summary": "Modifica una estación",
                "description": "Modifica una estación redirigiendo la petición
al microservicio 'polucion'",
                "operationId": "modificarEstacion",
                "parameters": [
                    {
                        "name": "id",
                        "in": "path",
                        "required": true,
                        "schema": {
```

```
                            "type": "string"
                        }
                    }
                ],
                "requestBody": {
                    "content": {
                        "application/json": {
                            "schema": {
                                "$ref": "#/components/schemas/Estacion"
                            }
                        }
                    },
                    "required": true
                },
                "responses": {
                    "400": {
                        "description": "Faltan campos obligatorios en la
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "401": {
                        "description": "Sin permisos necesarios para esta
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "404": {
                        "description": "No existe una estación con el id
indicado",
                        "content": {
                            "*/*": {
```

```
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "200": {
                    "description": "Estación actualizada correctamente",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                }
            },
            "security": [
                {
                    "bearerAuth": []
                }
            ]
        },
        "delete": {
            "tags": [
                "Operaciones que necesitan el rol 'admin'"
            ],
            "summary": "Elimina una estación",
            "description": "Elimina una estación redirigiendo la petición
al microservicio 'polucion'",
            "operationId": "eliminarEstacion",
            "parameters": [
                {
                    "name": "id",
                    "in": "path",
                    "required": true,
                    "schema": {
                        "type": "string"
                    }
                }
            ],
            "responses": {
```

```
                    "401": {
                        "description": "Sin permisos necesarios para esta
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "200": {
                        "description": "Estación eliminada correctamente",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "404": {
                        "description": "No se encontró la estación con ese
id",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    }
                },
                "security": [
                    {
                        "bearerAuth": []
                    }
                ]
            }
        },
        "/aparcamiento/{id}": {
            "put": {
```

```
                "tags": [
                    "Operaciones que necesitan el rol 'admin'"
                ],
                "summary": "Modifica un parking",
                "description": "Modifica una parking redirigiendo la petición
al microservicio 'bicicletas'",
                "operationId": "modificarParking",
                "parameters": [
                    {
                        "name": "id",
                        "in": "path",
                        "required": true,
                        "schema": {
                            "type": "string"
                        }
                    }
                ],
                "requestBody": {
                    "content": {
                        "application/json": {
                            "schema": {
                                "$ref": "#/components/schemas/Parking"
                            }
                        }
                    },
                    "required": true
                },
                "responses": {
                    "404": {
                        "description": "No existe una parking con el id
indicado",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "400": {
                        "description": "Faltan campos obligatorios en la
petición",
```

```
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "401": {
                        "description": "Sin permisos necesarios para esta
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "200": {
                        "description": "Parking actualizado correctamente",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    }
                },
                "security": [
                    {
                        "bearerAuth": []
                    }
                ]
            },
            "delete": {
                "tags": [
                    "Operaciones que necesitan el rol 'admin'"
                ],
                "summary": "Elimina un parking",
```

```
                "description": "Elimina un parking redirigiendo la petición al
microservicio 'bicicletas'",
                "operationId": "eliminarParking",
                "parameters": [
                    {
                        "name": "id",
                        "in": "path",
                        "required": true,
                        "schema": {
                            "type": "string"
                        }
                    }
                ],
                "responses": {
                    "401": {
                        "description": "Sin permisos necesarios para esta
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "404": {
                        "description": "No se encontró el parking con ese id",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "200": {
                        "description": "Parking eliminado correctamente",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
```

```
                        }
                    }
                }
            },
            "security": [
                {
                    "bearerAuth": []
                }
            ]
        }
    },
    "/estacion": {
        "post": {
            "tags": [
                "Operaciones que necesitan el rol 'admin'"
            ],
            "summary": "Crea una estación",
            "description": "Crea una estación redirigiendo la petición al
microservicio 'polucion'",
            "operationId": "crearEstacion",
            "requestBody": {
                "content": {
                    "application/json": {
                        "schema": {
                            "$ref": "#/components/schemas/Estacion"
                        }
                    }
                },
                "required": true
            },
            "responses": {
                "409": {
                    "description": "Ya existe una estación con el id
indicado",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
```

```
                    "201": {
                        "description": "Estación creada correctamente",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "401": {
                        "description": "Sin permisos necesarios para esta
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "400": {
                        "description": "Faltan campos obligatorios como 'id' o
'dirección'",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    }
                },
                "security": [
                    {
                        "bearerAuth": []
                    }
                ]
            }
        },
        "/aparcamiento": {
            "post": {
```

```
                "tags": [
                    "Operaciones que necesitan el rol 'admin'"
                ],
                "summary": "Crea un aparcamiento",
                "description": "Crea un aparcamiento redirigiendo la petición
al microservicio 'bicicletas'",
                "operationId": "crearParking",
                "requestBody": {
                    "content": {
                        "application/json": {
                            "schema": {
                                "$ref": "#/components/schemas/Parking"
                            }
                        }
                    },
                    "required": true
                },
                "responses": {
                    "401": {
                        "description": "Sin permisos necesarios para esta
petición",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "201": {
                        "description": "Aparcamiento creado correctamente",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "400": {
                        "description": "Faltan campos obligatorios como 'id' o
'dirección'",
```

```
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "409": {
                        "description": "Ya existe un aparcamiento con el id
indicado",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    }
                },
                "security": [
                    {
                        "bearerAuth": []
                    }
                ]
            }
        },
        "/aggregateData": {
            "post": {
                "tags": [
                    "Operaciones que necesitan el rol 'servicio'"
                ],
                "summary": "Obtiene datos de polución y de estaciones",
                "description": "Obtiene el número medio de bicicletas
disponibles y el número medio de cada tipo de contaminante atmosférico. Los
datos de polución se obtienen de la estación más cercana a cada aparcamiento.
Se invoca a intervalores regulares de tiempo y persiste en una base de datos
NoSQL.",
                "operationId": "agregar",
                "responses": {
                    "200": {
                        "description": "Documento generado",
```

```
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    },
                    "400": {
                        "description": "Error al generar el documento",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "object"
                                }
                            }
                        }
                    }
                },
                "security": [
                    {
                        "bearerAuth": []
                    }
                ]
            }
        },
        "/estacionCercana": {
            "get": {
                "tags": [
                    "Operaciones públicas"
                ],
                "summary": "Obtiene la estación más cercana",
                "description": "Devuelve la estación más cercana a la posición
indicada (lat/lon)",
                "operationId": "estacionCercana",
                "parameters": [
                    {
                        "name": "lat",
                        "in": "query",
                        "required": true,
                        "schema": {
                            "type": "number",
```

```
                        "format": "float"
                    }
                },
                {
                    "name": "lon",
                    "in": "query",
                    "required": true,
                    "schema": {
                        "type": "number",
                        "format": "float"
                    }
                }
            ],
            "responses": {
                "404": {
                    "description": "No hay estaciones disponibles",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "500": {
                    "description": "Error de comunicación con el servicio
de polución",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "200": {
                    "description": "Se ha encontrado una estación de
medición cercana",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
```

```
                                }
                            }
                        }
                    }
                }
            },
        "/aparcamientoCercano": {
            "get": {
                "tags": [
                    "Operaciones públicas"
                ],
                "summary": "Aparcamiento más cercano con bicis disponibles",
                "description": "Devuelve el más cercano a la posición
indicada",
                "operationId": "aparcamientoCercano",
                "parameters": [
                    {
                        "name": "lat",
                        "in": "query",
                        "required": true,
                        "schema": {
                            "type": "number",
                            "format": "float"
                        }
                    },
                    {
                        "name": "lon",
                        "in": "query",
                        "required": true,
                        "schema": {
                            "type": "number",
                            "format": "float"
                        }
                    }
                ],
                "responses": {
                    "500": {
                        "description": "Error de comunicación con el servicio
de bicicletas",
                        "content": {
                            "*/*": {
```

```
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "200": {
                    "description": "Se ha encontrado un aparcamiento
disponible cercano",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "404": {
                    "description": "No hay aparcamientos disponibles o con
bicis cerca de la ubicación dada",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                }
            }
        },
        "/aggregatedData": {
            "get": {
                "tags": [
                    "Operaciones públicas"
                ],
                "summary": "Obtiene los últimos datos agregados",
                "description": "Devuelve el documento de datos agregados más
reciente",
                "operationId": "obtenerUltimosDatos",
                "responses": {
                    "404": {
```

```
                    "description": "No hay datos disponibles",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                },
                "200": {
                    "description": "Datos encontrados",
                    "content": {
                        "*/*": {
                            "schema": {
                                "type": "object"
                            }
                        }
                    }
                }
            }
        }
    },
    "components": {
        "schemas": {
            "Estacion": {
                "type": "object",
                "properties": {
                    "id": {
                        "type": "string"
                    },
                    "direccion": {
                        "type": "string"
                    },
                    "latitud": {
                        "type": "number",
                        "format": "float"
                    },
                    "longitud": {
                        "type": "number",
                        "format": "float"
                    }
```

```
                }
            },
            "Parking": {
                "type": "object",
                "properties": {
                    "idparking": {
                        "type": "string"
                    },
                    "direction": {
                        "type": "string"
                    },
                    "bikesCapacity": {
                        "type": "integer",
                        "format": "int32"
                    },
                    "latitude": {
                        "type": "number",
                        "format": "float"
                    },
                    "longitude": {
                        "type": "number",
                        "format": "float"
                    }
                }
            }
        },
        "securitySchemes": {
            "bearerAuth": {
                "type": "http",
                "scheme": "bearer",
                "bearerFormat": "JWT"
            }
        }
    }
}
```