

Universidad ORT Uruguay

Facultad de Ingeniería

Bernard Wand Polak

Diseño de aplicaciones 2

Obligatorio 1

Diseño y especificación de la API

https://github.com/IngSoft-DA2-2023-2/268505_303433_303804

Rodrigo Conze - 268505

Sebastián Borjas – 303433

Agustina Martínez – 303804

Docente: Pablo Geymonat

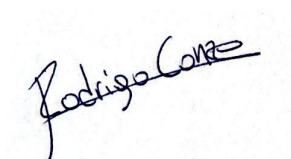
Abril 2024

Declaración de Autoría

Montevideo, Uruguay, 2 de Mayo de 2024.

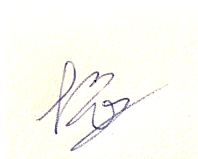
Nosotros, Rodrigo Conze, Sebastián Borjas y Agustina Martínez, declaramos que el trabajo que se presenta en esta obra es de nuestra propia mano. Podemos asegurar que:

- La obra fue producida en su totalidad mientras realizábamos el primer obligatorio de la asignatura Diseño de Aplicaciones II;
- Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad;
- Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra;
- En la obra, hemos acusado recibo de las ayudas recibidas;
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, hemos explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros;
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes



Rodrigo Conze

02/05/2024



Sebastián Borjas

02/05/2024



Agustina Martínez

02/05/2024

Abstract

Este documento describe el diseño y la especificación de la API, centrándose en los criterios REST, el mecanismo de autenticación, los códigos de estado, y los recursos de la API.

Palabras Claves

API REST, criterios REST, mecanismo de autenticación, códigos de estado, recursos de la API, SwaggerHub.

Índice

Declaración de Autoría.....	2
Abstract.....	3
Palabras Claves.....	4
Índice.....	5
Introducción:.....	6
Criterios Seguidos.....	6
Mecanismo de autenticación de requests.....	7
Códigos de error.....	7
Resources de la API.....	8

Introducción:

El diseño y la especificación de una API son elementos fundamentales para su éxito y adopción por parte de los desarrolladores y usuarios. En este documento, se presenta una descripción detallada de la API, siguiendo las mejores prácticas de la industria y asegurando que cumple con los estándares de REST. Se abordan aspectos como el mecanismo de autenticación, los códigos de error, y la estructura de los recursos de la API, con el objetivo de proporcionar una guía clara y completa para su implementación y uso.

Criterios Seguidos

Discusión de los criterios seguidos para asegurar que la API cumple con los criterios REST.

Nuestra API sigue los principios fundamentales de REST para garantizar un diseño flexible, escalable y fácilmente mantenible. Algunos de los criterios seguidos incluyen:

Arquitectura Cliente-Servidor: La API sigue un enfoque cliente-servidor, donde el cliente y el servidor son independientes entre sí, lo que permite una mayor escalabilidad y flexibilidad en el desarrollo y mantenimiento del sistema.

Estado de las Solicitudes: Las solicitudes son stateless, lo que significa que cada solicitud HTTP contiene toda la información necesaria para comprenderla y procesarla. Esto simplifica la implementación del servidor y permite una fácil escalabilidad.

Interfaz Uniforme: La API expone recursos a través de URLs únicas y utiliza métodos HTTP estándar (GET, POST, PUT, DELETE) para realizar operaciones en estos recursos. Esto proporciona una interfaz uniforme y coherente para interactuar con la API.

Manipulación de Recursos mediante Representaciones: Los recursos de la API son manipulados mediante representaciones, como JSON, que son fáciles de entender y procesar tanto para el cliente como para el servidor.

Sin Estado (Stateless): La API no almacena ningún estado de sesión en el servidor. Cada solicitud HTTP contiene toda la información necesaria para que el servidor la procese de manera independiente.

Mecanismo de autenticación de requests

El mecanismo de autenticación de las solicitudes se realiza mediante el uso de tokens generados con GUID. Los clientes deben incluir un token GUID válido en el encabezado de autorización (Authorization header) en cada solicitud HTTP. El servidor verifica la validez del token para autenticar al cliente y autorizar las acciones solicitadas.

Códigos de error

Los códigos de estado HTTP utilizados en la API y su significado son los siguientes:

200 (Success):

- La solicitud se completó con éxito.
- En algunos casos, se devuelve un body response.
- En otros casos, la solicitud se completa correctamente, pero no se devuelve nada.

400 (Client Error):

- Error de solicitud incorrecta.
- Indica que la solicitud enviada al servidor es incorrecta.

404 (Client Error):

- La solicitud con el ID especificado no existe.
- Indica que el recurso solicitado no se encuentra en el servidor.

401 (Client Error):

- Unauthorized.
- Indica que la solicitud no ha sido aplicada porque carece de credenciales de autenticación válidas para el recurso solicitado.

500 (Server Error):

- Errores internos del servidor.

```
public class ExceptionFilter: IExceptionHandler
{
    0 references
    public void OnException(ExceptionContext context)
    {
        try
        {
            throw context.Exception;
        }
        catch (KeyNotFoundException exception)
        {
            context.Result = new JsonResult(exception.Message) { StatusCode = 404 };
        }
        catch (ArgumentNullException exception)
        {
            context.Result = new JsonResult(exception.Message) { StatusCode = 400 };
        }
        catch (ArgumentOutOfRangeException exception)
        {
            context.Result = new JsonResult(exception.Message) { StatusCode = 400 };
        }
        catch (InvalidDataException exception)
        {
            context.Result = new JsonResult(exception.Message) { StatusCode = 400 };
        }
        catch (ArgumentException exception)
        {
            context.Result = new JsonResult(exception.Message) { StatusCode = 400 };
        }
        catch (InvalidOperationException exception)
        {
            context.Result = new JsonResult(exception.Message) { StatusCode = 400 };
        }
        catch (WrongEmailFormatException exception)
        {
            context.Result = new JsonResult(exception.Message) { StatusCode = 400 };
        }
        catch (PasswordNotFollowPolicy exception)
        {
            context.Result = new JsonResult(exception.Message) { StatusCode = 400 };
        }
    }
}
```

Resources de la API

La API proporciona acceso a varios recursos que permiten la gestión de edificios, categorías, invitaciones, informes, solicitudes de tickets y usuarios. Cada recurso tiene sus propios endpoints para realizar operaciones específicas.

En el archivo adjunto con nombre: Diseño WebApi, que se adjunta en la carpeta documentación, se detalla la información de los recursos de la API ordenados por controlador. Este archivo proporciona una descripción detallada de cada recurso, incluyendo

el verbo HTTP utilizado, la ruta de acceso, la descripción de la función, el cuerpo de la solicitud, el cuerpo de respuesta, los códigos de estado HTTP y una justificación para su inclusión.

Por favor, consulta el archivo adjunto para obtener más detalles sobre los recursos de la API.