Documento Técnico

28 de junio del 2024

Sistema de Gestión de Eventos

Introducción

Este documento técnico describe el diseño, desarrollo e implementación de una API RESTful para la gestión de eventos. La API permitirá a los usuarios crear, consultar, actualizar y eliminar eventos a través de solicitudes HTTP. Además, el uso de una API para la obtención de información relacionado con los eventos.

Se mencionará sobre la Metodología de Desarrollo de Software para la creación del programa, el tipo de arquitectura, los recursos, lista de endpoints definidos en la API, el manejo de errores y el modo de utilizar este desarrollo.

Al momento de realizar el programa se consideró el uso de principios SOLID y de buenas practicas para el desarrollo de aplicaciones web.

1. Metodología de Desarrollo de Software.

El desarrollo de la API RESTFul se basó en la siguiente metodología:

a) **Análisis de Requisitos:** Se identificaron los requisitos funcionales y no funcionales de la API, incluyendo las operaciones CRUD (Crear, Leer, Actualizar y Eliminar) sobre eventos, el manejo de errores y la documentación. No se consideró el uso de autenticación y autorización en esta primera versión, por considerarse una versión de fase probatoria.
b) **Diseño**: Se diseñó la arquitectura de la API, incluyendo los recursos, métodos HTTP, código de estado HTTP y formato de datos (JSON).
c) **Implementación**: Se implementó la API utilizando el Framework ASP.NET Core Web API. Se hizo uso de buenas prácticas de desarrollo, pruebas unitarias y de integración para garantizar el funcionamiento y la calidad del código.
d) **Pruebas**: Se realizaron pruebas detalladas a la API para verificar su correcto funcionamiento, incluyendo prueba de carga, prueba de rendimiento y prueba de seguridad.
e) Despliegue: aunque en este punto no se ha desplegado en un servidor web accesible a los usuarios, se hizo uso de software Git y Github para el manejo de versiones.

2. Arquitectura de la API.

La API RESTFul está basada en una arquitectura de recursos, donde cada recurso representa una entidad del sistema (por ejemplo, la entidad eventos). Los recursos se identifican por URIs (Uniform Resources Identifiers) y se pueden acceder a traves de metodos HTTP (GET, POST, PUT, DELETE).

3. Recursos de la API.

La API define los siguientes recursos:

- **Eventos**: Este recurso permite crear, consultar, actualizar y eliminar eventos. Los eventos tienen los siguientes atributos:

- eventId: Identificador unico del evento.
- eventName: Nombre del evento.
- eventDescription: Descripción del evento.
- startDate: Fecha de inicio del evento.
- endDate: Fecha de culminacion del evento.
- eventStatus: Estatus del evento.

Cada campo es requerido al momento de su creación y/o edicion, además los campos eventName y eventDescription tienen validaciones personalizadas en relación a la cantidad de caracteres permitidas.

4. **Endpoints de la API.**

   La API define los siguientes endpoints:

   Eventos:

   - GET => /api/Evento
   - GET => /api/Evento/id:int
   - POST => /api/Evento
     - Parametros: { "eventId": 0, "eventName": "string", “eventDescription": "string", "startDate": "string", "endDate": "string", "eventStatus": "string"}
   - PUT => /api/Evento/{id}
   - DELETE => /api/Evento/{id}

5. **Manejo de Errores:**

   La API utiliza códigos de estado HTTP estándar para indicar el resultado de cada solicitud. En caso de error, la API devolverá un

mensaje de error bajo la implementación de Logger para su posterior revisión y análisis.

6. **Documentación.**

   La API está documentada utilizando la especificación OpenAPI. La documentación está disponible en formato Swagger y se puede acceder a ella en la siguiente URL: https://localhost:44391/swagger/index.html

7. **Estructura.**

   El proyecto se denomina GestorEvento, la cual contiene dos proyectos:

   - GestorEventos.Server.

     Este proyecto fue desarrollado en la plantilla ASP.NET Core Web API, bajo el framework .NET 8.0, en el lenguaje de programación C#.

     Se divide en:

     - Clase EventoController, en donde se definen los endpoints y los métodos que cada uno de ellos requiere para su funcionamiento.
     - Clase EventoDto para la transferencia de datos.
     - Clase StoredEvents la cual permite extraer y listar los datos del archivo JSON que contiene los datos para probar la API RESTFul y poder utilizarlo en el CRUD de la API.
     - El archivo event_list.json que es utilizado para almacenar los datos y para manipular la información desde el CRUD de la API.

- GestorEventos.Cliente

  Este proyecto fue creado bajo la aplicación web de ASP.NET Core (MVC), la misma también posee herramientas para el desarrollo de servicios RESTFul HTTP. Se utilizó el framework NET 8.0 y el lenguaje de programación C#.

  Se divide en:

  - La carpeta raíz wwwroot, contenedora de hojas de estilo CSS, imágenes, JavaScript, librerías e imagen.ico.
  - La clase EventoController en donde se encuentra los métodos para el consumo y manipulación de la API.
  - La clase Eventos para la definición y transferencia de los datos de los eventos, además de definir sus validaciones personalizadas.
  - Las paginas Razor {Create, Details, Edit, Index}
  - Se manipulo el _Layout.cshtml para modificar aspectos del front de la pagina.
  - En la clase program se agrega el servicio AddHttpClient() para realizar solicitudes Http y poder conectar a la web api.

8. **Ruta GitHub:**

   Para descarga y revisión del proyecto, presento el enlace de la última versión: **https://github.com/EL-DesaFull/Gestor_Eventos**

9. **Conclusión.**

   Se ha desarrollado una API RESTful para la gestión de eventos que cumple con los requisitos funcionales y no funcionales establecidos. La API está diseñada de forma modular, escalable y segura. Se ha

utilizado una metodología de desarrollo de software rigurosa para garantizar la calidad del código.