

**UNIVERSIDAD EUROPEA DEL ATLÁNTICO**

**GRADO EN**

**Ingeniería Informática**



**Sistemas Distribuidos y Programación en Paralelo**

Proyecto Final:

**Sistema de Gestión de Eventos**

Trabajo realizado por:

**Brenda Lopes Ventura de Souza**

**Manuela Grizoni Smelan**

# Sistema de Gestión de Evento

## Requisitos básicos para el funcionamiento

Este proyecto consiste en desarrollar una aplicación para la gestión de eventos. Se utiliza un backend en C# usando el framework .NET que expone una API REST, que se comunica con JSON, para manejar eventos, inscripciones y recursos, usando una base de datos en MySQL Workbench y Postman para probar los endpoints.

Se utiliza un frontend en HTML para la interacción del usuario, que tiene un formulario para registrar eventos, inscripción de usuarios y visualización de eventos disponibles.

## Gestión de Eventos

Se puede crear, editar y listar eventos. Cada evento tiene informaciones como nombre, fecha, ubicación y capacidad máxima. lista de participantes

## Inscripción de Participantes

Los usuarios pueden registrarse en un evento mediante un formulario, y los datos registrados se guardan en una base de datos.

## Base de Datos

Todas las informaciones se quedan almacenadas en una base de datos, esto permite gestionar y consultar la información de forma eficiente

## Interfaz de Usuario

### Creación de la Interfaz de Usuario para el Proyecto

Se diseñó una interfaz de usuario sencilla utilizando **HTML**, **CSS** y **JavaScript puro** para interactuar con la API REST desarrollada en .NET Core. El objetivo fue crear un frontend funcional que permitiera realizar las operaciones básicas de gestión de eventos.

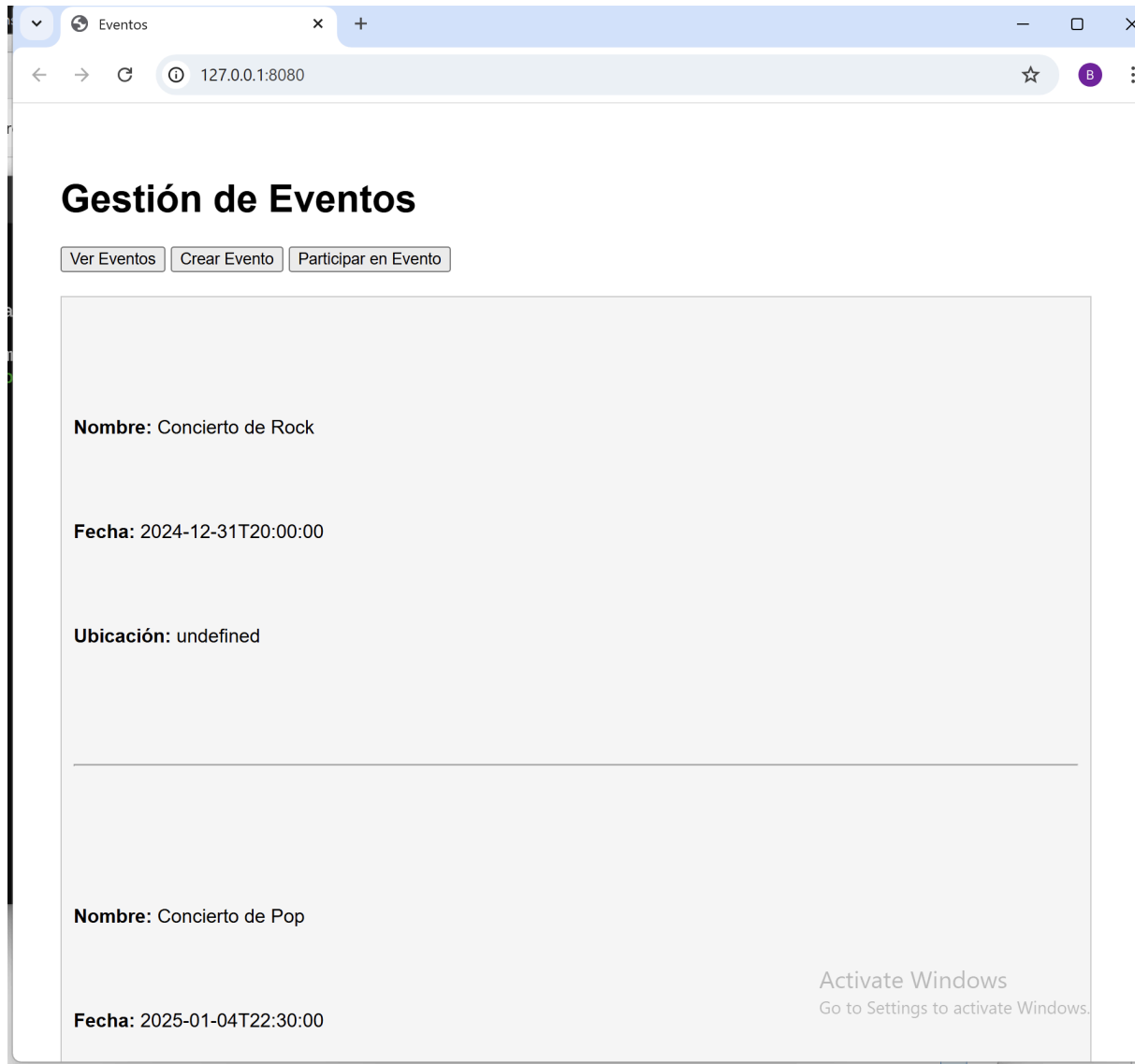
#### 1. Herramientas Utilizadas:

- **HTML/CSS:** Para estructurar y estilizar la interfaz.
- **JavaScript:** Para realizar solicitudes HTTP (**fetch**) a la API.
- **live-server:** Para servir el frontend localmente.

#### 2. Funcionalidades Implementadas:

- **Ver Eventos:** Un botón que realiza una solicitud **GET** a la API y lista los eventos en formato **Nombre**, **Fecha** y **Ubicación**.

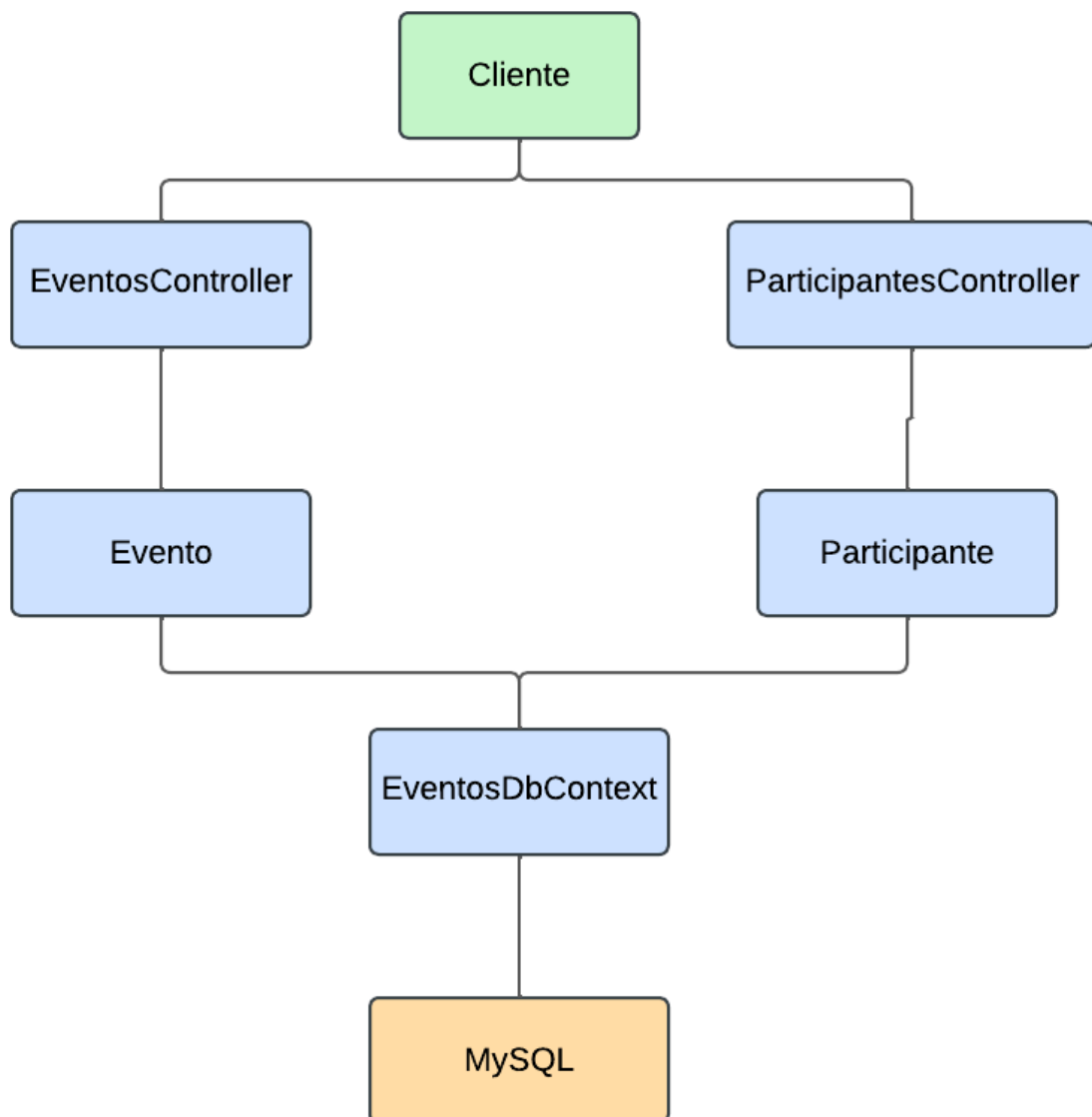
- **Crear Evento:** Un formulario que envía los datos del nuevo evento mediante una solicitud **POST** a la API.
- **Participar en Evento:** Un formulario para registrar un participante en un evento específico utilizando una solicitud **POST**.

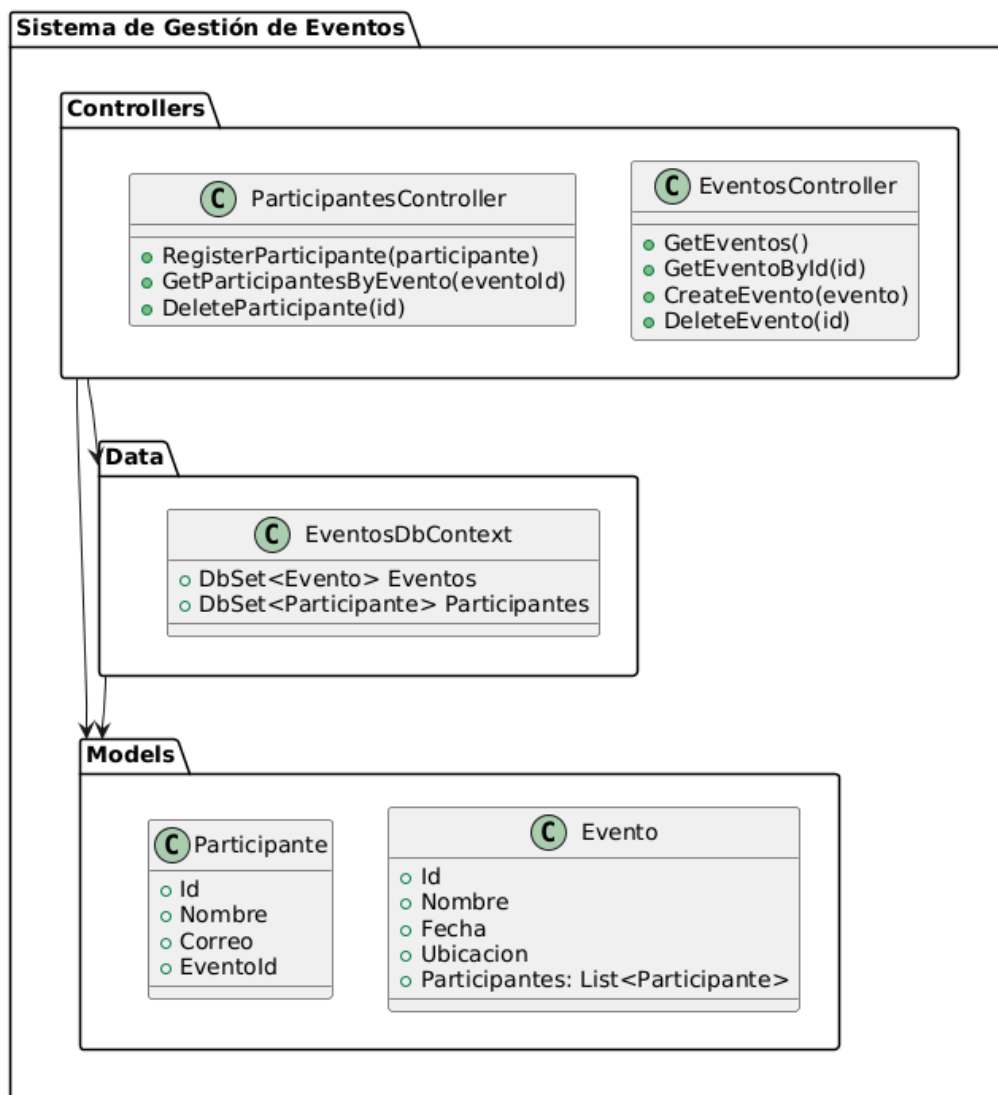


## Programación Asíncrona

En `EventosController` y `ParticipantesController` se usa programación asíncrona, teniendo el recurso compartido la base de datos, que nos asegura que las consultas no bloqueen el hilo principal, permitiendo que otras solicitudes se procesen simultáneamente. Las consultas y modificaciones a la base de datos como `ToListAsync`, `AddAsync`, `SaveChangeAsync` son asíncronas lo que nos asegura un mejor rendimiento y cumplimiento con los requisitos del sistema.

## Diagrama de la arquitectura





## Pruebas Endpoint

Se realizaron pruebas GET, POST y DELETE, mediante el Postman para garantizar la calidad y el correcto funcionamiento del sistema:

### 1. POST/api/eventos:

HTTP SistemaEventosAPI / Crear evento

POST https://localhost:7080/api/Eventos

Params Authorization Headers (9) Body Scripts Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "nombre": "Concierto de Pop",
3   "fecha": "2025-01-04T22:30:00",
4   "ubicación": "Virgen del Mar",
5   "capacidad": 2000
6 }
7
```

Body Cookies Headers (5) Test Results 201 Created 945 ms 329 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 3,
3   "nombre": "Concierto de Pop",
4   "fecha": "2025-01-04T22:30:00",
5   "ubicación": "Virgen del Mar",
6   "capacidad": 2000,
7   "participantes": []
8 }
```

## 2. GET/api/eventos:

HTTP SistemaEventosAPI / Obtener todos los eventos

GET https://localhost:7080/api/Eventos

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body Cookies Headers (4) Test Results 200 OK 278 ms 490 B

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 1,
4     "nombre": "Concierto de Rock",
5     "fecha": "2024-12-31T20:00:00",
6     "ubicación": "Auditorio Nacional",
7     "capacidad": 500,
8     "participantes": [
9       {
10        "id": 1,
11        "nombre": "Juan Pérez",
12        "correo": "juan.perez@example.com",
13        "eventoId": 1
14      }
15    ]
16  },
17  {
18    "id": 3,
19    "nombre": "Concierto de Pop",
20    "fecha": "2025-01-04T22:30:00",
21    "ubicación": "Virgen del Mar",
22    "capacidad": 2000,
23    "participantes": []
24  }
25 ]
```

## 3. GET/api/eventos{id}

HTTP SistemaEventosAPI / **Buscar evento** Save Share

GET Send https://localhost:7080/api/Eventos/1

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (4) Test Results 200 OK 63 ms 355 B 🔗 📄 🔍

Pretty Raw Preview Visualize JSON 🔗 📄 🔍

```
1 {
2   "id": 1,
3   "nombre": "Concierto de Rock",
4   "fecha": "2024-12-31T20:00:00",
5   "ubicación": "Auditorio Nacional",
6   "capacidad": 500,
7   "participantes": [
8     {
9       "id": 1,
10      "nombre": "Juan Pérez",
11      "correo": "juan.perez@example.com",
12      "eventoId": 1
13    }
14  ]
15 }
```

#### 4. POST/api/Participantes

HTTP SistemaEventosAPI / **Crear Participante** Save Share

POST Send https://localhost:7080/api/Participantes

Params Authorization Headers (9) Body Scripts Tests Settings Cookies Beautify

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON 🔗 📄 🔍

```
1 {
2   "nombre": "Brenda Lopes",
3   "correo": "brenda.lopes@example.com",
4   "eventoId": 4
5 }
6
```

Body Cookies Headers (4) Test Results 200 OK 29 ms 221 B 🔗 📄 🔍

Pretty Raw Preview Visualize JSON 🔗 📄 🔍

```
1 {
2   "id": 4,
3   "nombre": "Brenda Lopes",
4   "correo": "brenda.lopes@example.com",
5   "eventoId": 4
6 }
```

#### 5. GET/api/participantes/evento/{eventoId}

HTTP SistemaEventosAPI / **Obtener participantes de 1 evento** Save Share

GET Send <https://localhost:7080/api/Participantes/evento/4>

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (4) Test Results 200 OK 27 ms 223 B 🔗 📄 🔍

Pretty Raw Preview Visualize JSON 🔗 📄 🔍

```
1 [
2   {
3     "id": 4,
4     "nombre": "Brenda Lopes",
5     "correo": "brenda.lopes@example.com",
6     "eventoId": 4
7   }
8 ]
```

## 6. DELETE /api/eventos/{id}

HTTP SistemaEventosAPI / **Borrar evento** Save Share

DELETE Send <https://localhost:7080/api/Eventos/3>

Params Authorization Headers (7) Body Scripts Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (2) Test Results 204 No Content 46 ms 81 B 🔗 📄 🔍

Pretty Raw Preview Visualize Text 🔗 📄 🔍

```
1
```

## 7. DELETE /api/participantes/{id}



SistemaEventosAPI / **Borrar participante**

Save

Share

DELETE

https://localhost:7080/api/Participantes/4

Send

Params

Authorization

Headers (7)

Body

Scripts

Tests

Settings

Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (2)

Test Results

204 No Content

27 ms

81 B

Raw

...

Pretty

Raw

Preview

Visualize

Text

1

Copy

Share

Search