

PROYECTO SISTEMA DE GESTIÓN DE EVENTOS SIGEU  
BASE DE DATOS DOCUMENTAL

RODRIGO ANDRÉS GÓMEZ LÓPEZ. Código: 2247014

DAVID HERNÁNDEZ PAZ. Código: 2247003

DANIEL ALEXANDER BRAND GARCÍA. Código:2246133

SEBASTIÁN MANRIQUE MEJIA. 2246988

MICHAEL MACOWLI CARDONA RODRIGUEZ. Código: 2246268

ALMACENAMIENTO DE DATOS

Presentado a:

PHD. JHON EDER MASSO DAZA



UNIVERSIDAD AUTÓNOMA DE OCCIDENTE  
INGENIERÍA DE DATOS E INTELIGENCIA ARTIFICIAL  
SANTIAGO DE CALI

2025

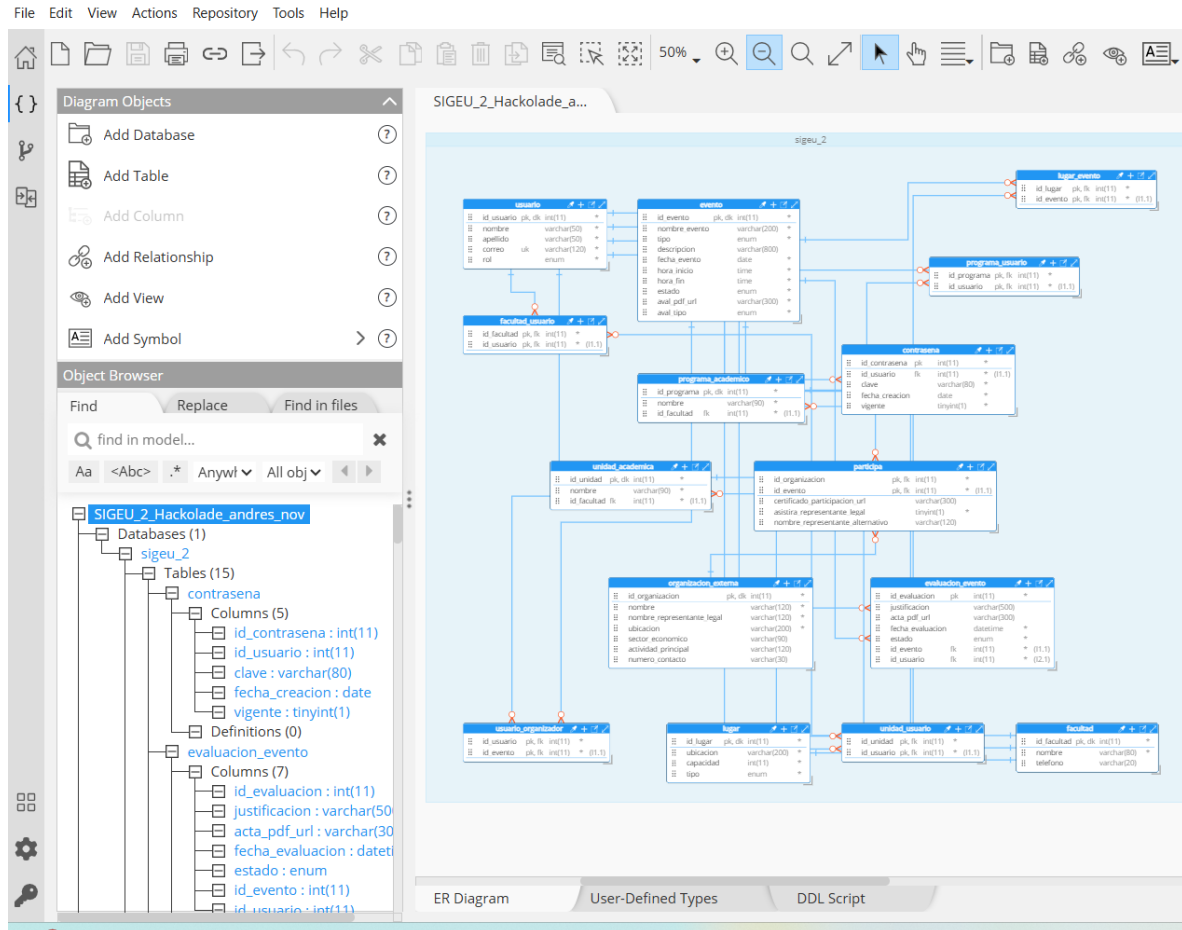
## CONTENIDO

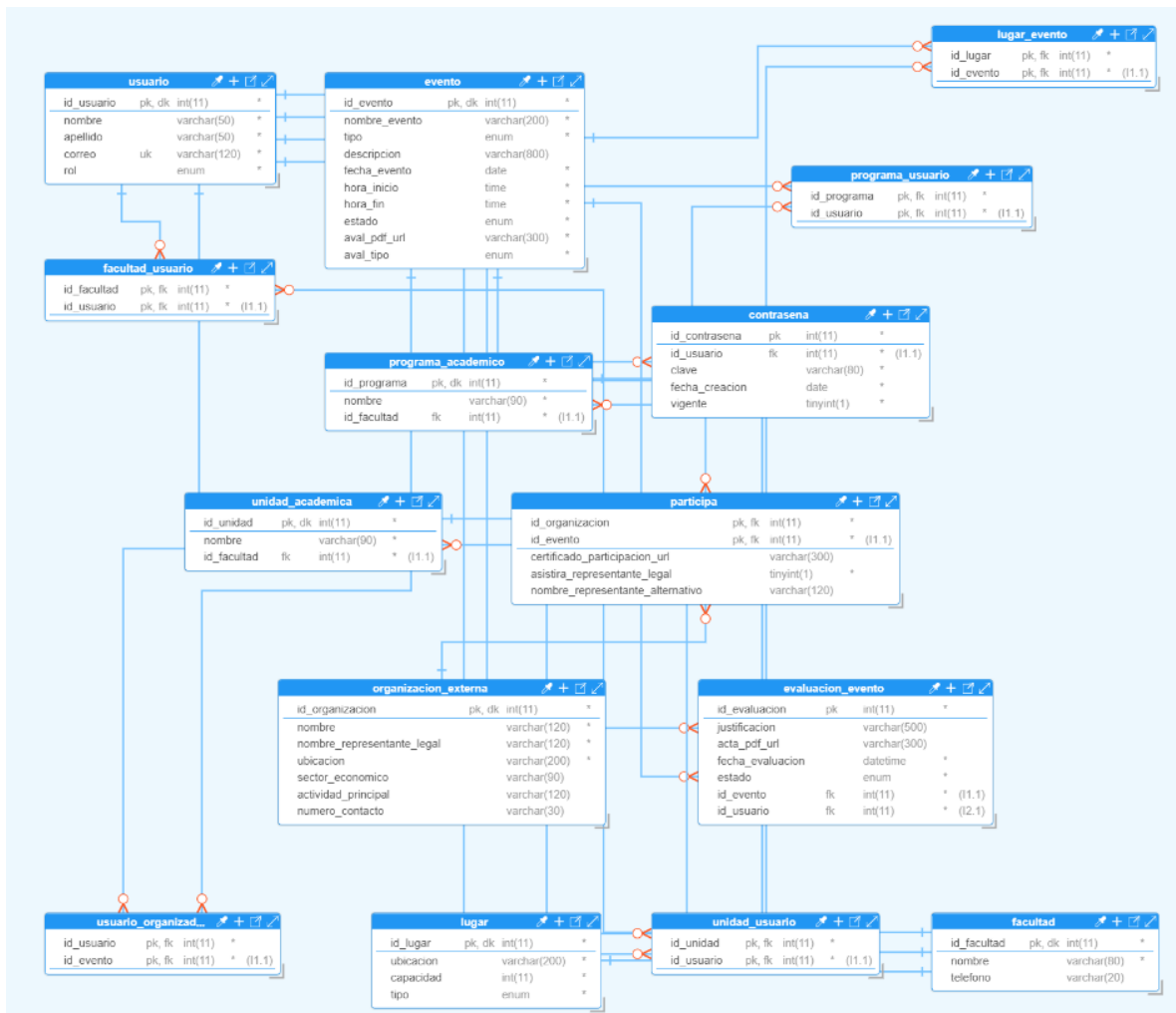
1. VERIFICAR EL MODELADO DE DATOS EN HACKOLADE. ....	3
2. CREAR BASE DATOS SEGÚN MODELADO. ....	5
3. EMULAR LOS TRIGER EN MONGO DB.....	6
4. INGRESO DE DATOS DE PRUEBA .....	10
5. API REST BACKEND.....	16
5.1 ESTRUCTURA DEL BACKEND .....	16
5.1.1 Carpeta docs/ – Documentos, scripts y recursos del proyecto.....	17
5.1.2. Carpeta src/ – Backend FastAPI (API REST).....	19
5.1.3 Archivos Adicionales .....	21
5.2 EJECUCION ENTORNO VIRTUAL .....	21
5.3 INSTALAR DEPENDENCIAS DEL BACKEND .....	21
5.4 REVISAR EL ARCHIVO .env .....	22
5.5 VERIFICAR LA CONEXIÓN EN src/db/client.py .....	22
5.6 LEVANTAR EL BACKEND CON UVICORN .....	22
5.7 ABRIR SWAGGER UI Y VERIFICAR ENDPOINTS .....	23
5.7.1 PRUEBA DE EVENTOS .....	24
5.7.2 PRUEBA DE USUARIOS .....	37
5.7.3 PRUEBA DE FACULTADES .....	37
ANEXO 2. ACCESO GITHUB – BACKEND .....	42
ANEXO 3. ACCESO PRUEBAS POSTMAN .....	42

## 1. VERIFICAR EL MODELADO DE DATOS EN HACKOLADE.

Esto se hace ejecutando desde Hackolade, el archivo adjunto de la carpeta Docs:

SIGEU\_2\_Hackolade\_andres\_nov.hck.json





NOTA: VER DETALLE DE MODELADO EN HACKOLADE EN EL ARCHIVO ADJUNTO:

SIGEU\_2\_DOCUMENTACION\_MODELADO\_Hackolade.pdf

## 2. CREAR BASE DATOS SEGÚN MODELADO.

Esto se hace desde Mongo DB compas, ejecutando el contenido del archivo:

sigeu\_2\_PARA\_MONGO.txt

Se copia todo su contenido a la Shell y se presiona enter.



### 3. EMULAR LOS TRIGER EN MONGO DB.

MongoDB no maneja triggers en el motor como los RDBMS tradicionales (MySQL, PostgreSQL). Pero sí permite replicar la lógica de negocio mediante funciones JavaScript ejecutadas en el servidor.

Por tanto para este ejercicio, creamos 4 funciones:

Trigger 1 – crearUsuario()

- Valida dominio @uao.edu.co
- Valida roles permitidos
- Inserta usuario limpio

Emula un “BEFORE INSERT” en SQL.

Trigger 2 – asignarUsuarioAPrograma()

- Solo estudiantes pueden pertenecer a un programa

Emula una restricción de integridad.

Trigger 3 – registrarOrganizadorEvento()

- Solo docentes o estudiantes pueden organizar eventos

Emula regla de rol.

Trigger 4 – evaluarEvento()

- Solo secretarios pueden evaluar
- Solo secretarios de LA MISMA FACULTAD del evento
- Inserta evaluación

- Actualiza estado del evento

Emula un “AFTER INSERT” y un “BEFORE UPDATE” en SQL.

Como lo comentamos anteriormente, se emulan las reglas de negocio para que el sistema sea más funcional; esto se logra a través de la creación de funciones que son llamadas desde Mongo. Esto se realiza después de crear la BD, y desde la Shell de Mongo DB se copia todo el contenido del archivo:

sigeu\_2\_triggers.js

```
>_MONGOSH

// 3. Validar que el secretario pertenece a la facultad del evento
const rel = db.facultad_usuario.findOne({
  facultad_id: evento.facultad_id,
  usuario_id: usuarioEvaluadorId
});

if (!rel) {
  throw new Error('El secretario no está asociado a la facultad del evento, no puede evaluarlo');
}

// 4. Insertar registro de evaluación (como si fuera AFTER INSERT)
db.evaluacion_evento.insertOne({
  justificacion,
  acta_pdf_url: actaPdfUrl,
  fecha_evaluacion: new Date(),
  estado,
  evento_id: eventoId,
  usuario_id: usuarioEvaluadorId
});

// 5. Actualizar estado del evento
db.evento.updateOne(
  { _id: eventoId },
  { $set: { estado } }
);
}
< [Function: evaluarEvento]
sigeu_2>
```

La otra opción es desde mongosh:

```
sigeu_2> use('sigeu_2');  
... load("D:\\SIGEU\\SIGEU_2_MONGO\\docs\\sigeu_2_triggers.js");  
...  
true  
sigeu_2> |
```

para validar que son funcionales, ejecutamos la siguiente orden desde la Shell:

typeof evaluarEvento

```
sigeu_2> typeof evaluarEvento
```

En este caso verificamos la función evaluar evento.

Ahora, si queremos validar todas las funciones, se hace a través del siguiente comando:

Sigue\_2>typeof crearUsuario

typeof asignarUsuarioAPrograma

typeof registrarOrganizadorEvento

```
sigeu_2> |typeof crearUsuario  
typeof asignarUsuarioAPrograma  
typeof registrarOrganizadorEvento
```

Devuelve “function” indicando que se ejecutaron bien.



```
> typeof crearUsuario  
typeof asignarUsuarioAPrograma  
typeof registrarOrganizadorEvento  
  
< function  
sigeu_2>
```

#### 4. INGRESO DE DATOS DE PRUEBA

Para terminar de verificar las funciones y validación del dominio, el sistema nos muestra algunas advertencias como las siguientes, cuando creamos datos:

“Error: El correo debe ser del dominio @uao.edu.co”

Para crear los datos ejecutamos en la Shell, el script:

sigeu\_2\_datos\_prueba.js

```
// 15. Evaluaciones de ejemplo (usa evaluarEvento)
// -----
evaluarEvento({
  eventoId: evSeminarioIAId,
  usuarioEvaluadorId: uSec1, // secretario Facultad Ingeniería
  estado: 'aprobado',
  justificacion: 'Cumple todos los requisitos académicos',
  actaPdfUrl: 'https://uao.edu.co/actas/acta_seminario_ia.pdf'
});

evaluarEvento({
  eventoId: evFeriaBienestarId,
  usuarioEvaluadorId: uSec3, // secretario Facultad Ciencias Sociales
  estado: 'aprobado',
  justificacion: 'Actividad alineada con el plan de bienestar universitario',
  actaPdfUrl: 'https://uao.edu.co/actas/acta_feria_bienestar.pdf'
});

evaluarEvento({
  eventoId: evJornadaLecturaId,
  usuarioEvaluadorId: uSec3,
  estado: 'rechazado',
  justificacion: 'Falta claridad en el objetivo de aprendizaje',
  actaPdfUrl: null
});

print('Datos de prueba SIGEU_2 cargados correctamente.');
```

✖ Error: El correo debe ser del dominio @uao.edu.co

sigeu\_2>|

Para ver que datos se ingresaron ejecutamos:

```
[
  'facultad',
  'unidad_academica',
  'programa_academico',
  'usuario',
  'lugar',
  'evento',
  'organizacion_externa',
  'programa_usuario',
  'unidad_usuario',
  'facultad_usuario',
  'usuario_organizador',
  'lugar_evento',
  'participa',
  'evaluacion_evento'
].forEach(c => {
  print(c, ':', db.getCollection(c).countDocuments());
});
```

En este caso (practica), los resultados fueron los siguientes:

```

> [
  'facultad',
  'unidad_academica',
  'programa_academico',
  'usuario',
  'lugar',
  'evento',
  'organizacion_externa',
  'programa_usuario',
  'unidad_usuario',
  'facultad_usuario',
  'usuario_organizador',
  'lugar_evento',
  'participa',
  'evaluacion_evento'
].forEach(c => {
  print(c, ': ', db.getCollection(c).countDocuments());
});
< facultad
< :
< 2
< unidad_academica
< :
< 3
< programa_academico
< :
< 4
< usuario
< :
< 0
< lugar
< :
< 0
< evento
< :
< 0
< organizacion_externa
< :
< 0
< programa_usuario
< :
< 0
< unidad_usuario
< :
< 0
< facultad_usuario
< :
< 0
< usuario_organizador
< :
< 0
< lugar_evento
< :
< 0
< participa
< :
< 0
< evaluacion_evento
< :
< 0
sigeu_2>

```

Solo se crearon los siguientes registros:

```

facultad : 2
unidad_academica : 3
programa_academico : 4

```

Por tanto como practica, limpiamos las colecciones:

```

use('sigeu_2');

db.usuario.deleteMany({});
db.contrasena.deleteMany({});
db.programa_usuario.deleteMany({});
db.unidad_usuario.deleteMany({});
db.facultad_usuario.deleteMany({});

```

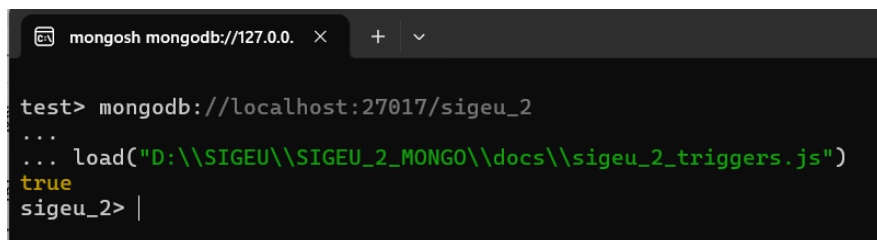
```
db.lugar.deleteMany({});
db.evento.deleteMany({});
db.usuario_organizador.deleteMany({});
db.lugar_evento.deleteMany({});
db.organizacion_externa.deleteMany({});
db.participa.deleteMany({});
db.evaluacion_evento.deleteMany({});
```

```
> use('sigeu_2');

db.usuario.deleteMany({});
db.contrasena.deleteMany({});
db.programa_usuario.deleteMany({});
db.unidad_usuario.deleteMany({});
db.facultad_usuario.deleteMany({});
db.lugar.deleteMany({});
db.evento.deleteMany({});
db.usuario_organizador.deleteMany({});
db.lugar_evento.deleteMany({});
db.organizacion_externa.deleteMany({});
db.participa.deleteMany({});
db.evaluacion_evento.deleteMany({});
< {
  acknowledged: true,
  deletedCount: 0
}
sigeu_2>
```

Luego volveremos a cargar los trigger-

1. Coentamos a la base de datos desde mongo sh y ejecutamos la emulacionde triggers



```
mongosh mongodb://127.0.0.1:27017/sigeu_2
test> mongod --localhost:27017/sigeu_2
... load("D:\\SIGEU\\SIGEU_2_MONGO\\docs\\sigeu_2_triggers.js")
true
sigeu_2> |
```

- a.
2. Probamos que se ejecutaron bien las fuciones
  - a.

```
sigeu_2> typeof crearUsuario
... typeof asignarUsuarioAPrograma
... typeof registrarOrganizadorEvento
... typeof evaluarEvento
...
function
sigeu_2> |
```

b.

Otra opción desde mongosh:

```
sigeu_2> use('sigeu_2')
... load("D:\\SIGEU\\SIGEU_2_MONGO\\docs\\sigeu_2_triggers.js")
... load("D:\\SIGEU\\SIGEU_2_MONGO\\docs\\sigeu_2_datos_prueba.js")
...
TRIGGERS SIGEU_2 cargados correctamente.
DATOS DE PRUEBA SIGEU_2 CARGADOS CORRECTAMENTE
true
sigeu_2> |
```

Validacion de registros creados:

```
sigeu_2> [  
...   'facultad',  
...   'unidad_academica',  
...   'programa_academico',  
...   'usuario',  
...   'lugar',  
...   'evento',  
...   'organizacion_externa',  
...   'programa_usuario',  
...   'unidad_usuario',  
...   'facultad_usuario',  
...   'usuario_organizador',  
...   'lugar_evento',  
...   'participa',  
...   'evaluacion_evento'  
... ].forEach(c => {  
...   print(c, ': ', db.getCollection(c).countDocuments());  
... });
```

```
facultad : 2  
unidad_academica : 3  
programa_academico : 4  
usuario : 19  
lugar : 5  
evento : 5  
organizacion_externa : 5  
programa_usuario : 10  
unidad_usuario : 9  
facultad_usuario : 4  
usuario_organizador : 5  
lugar_evento : 5  
participa : 5  
evaluacion_evento : 2  
  
sigeu_2> |
```

## 5. API REST BACKEND

### 5.1 ESTRUCTURA DEL BACKEND

Ruta raíz: D:\SIGEU\SIGEU\_2\_MONGO

SIGEU\_2\_MONGO/

|

└─ docs

├─ requirements.txt

├─ .env

└─ src/

├─ main.py

├─ db/

├─ models/

├─ crud/

└─ routes/



### 5.1.1 Carpeta docs/ – Documentos, scripts y recursos del proyecto

Carpeta / Archivo	Función dentro del proyecto
docs/	Carpeta principal del proyecto MongoDB; contiene el modelado, scripts de creación, triggers, datos de prueba, CRUD y agregaciones.
1. MODELADO/	Carpeta que contiene el MER/MR generado en Hackolade y su documentación.
SIGEU_2_DOCUMENTACION_MODELADO_Hackolade.pdf	Documento formal del modelado.
SIGEU_2_Hackolade_andres_nov.hck.json	Archivo del modelo en Hackolade; permite abrir y editar el MER/MR con las entidades y relaciones.
2. CREAR_BD/	Scripts usados para crear la base en MongoDB con validaciones.
sigeu_2_PARA_MONGO.txt	Script que contiene todos los \$jsonSchema para crear colecciones e índices en MongoDB.
3. TRIGGERS_EMULADOS/	Carpeta que contiene las funciones que simulan los triggers en MongoDB.
sigeu_2_triggers.js	Script que define las funciones tipo trigger: crearUsuario(), asignarUsuarioAPrograma(), registrarOrganizadorEvento(), evaluarEvento().
4. DATOS_PRUEBA/	Carpeta con el script que llena la BD con datos reales.

sigeu_2_datos_prueba.js	Crea facultades, usuarios, eventos, organizaciones, relaciones y evaluaciones. Asegura 20 usuarios, 5 eventos, etc.
5. CRUD/	Carpeta con código de consulta en formato .txt, usado como demostración de CRUD manual desde MongoDB.
CODIGO_CRUD_...txt	18 archivos de CRUD manual: crear usuario, leer todos, filtrar por rol, actualizar, eliminar, CRUD de eventos, organizaciones, relaciones, etc. Son consultas de ejemplo, no código de FastAPI.
6. AGREGACIONES/	Carpeta con pipelines MongoDB para reportes y análisis.
AGREGACION-EVENTOS_POR_FACULTAD.txt	Pipeline que agrupa eventos por facultad.
AGREGACION-TOTAL_USUARIOS_POR_ROL.txt	Cuenta usuarios por rol.
AGREGACION-ORG_EXTERNAS_ENV_REPRESENTANTE.txt	Reporta organizaciones por representante.
AGREGACION-EVENTOS_EVALUADOS_SECRETARIO.txt	Lista eventos evaluados por un secretario.
7. CONSULTAS_VERIFICACION/	Consultas rápidas para confirmar integridad de la BD.
CONSULTA_VERIFICACION-ESTUDIANTES_ING.txt	Verifica usuarios por programa.
CONSULTA_VERIFICACION-EVENTOS_EN_REVISION.txt	Busca eventos cuyo estado es "en_revision".

CONSULTA_VERIFICACION-FACULTAD_CON_MAS_EVENTOS.txt	Muestra agrupación para ver qué facultad genera más eventos.
PASO_A_PASO_EJECUCION_PROYECTO_FINAL_MONGO_DB.docx	Documento narrativo para sustentación: paso a paso del proyecto en MongoDB.

### 5.1.2. Carpeta src/ – Backend FastAPI (API REST)

Carpeta / Archivo	Función
src/	Carpeta raíz del backend FastAPI. Aquí está todo el API REST.
src/db/	Contiene la conexión a MongoDB.
client.py	Configura motor.AsyncIOMotorClient y la conexión a la BD sigeu_2. Es el archivo encargado de conectar FastAPI con tu base de datos real.
src/models/	Definición de modelos Pydantic para request/response.
objectid.py	Clase PyObjectId para manejar _id de MongoDB en FastAPI. Convierte ObjectId : String.
usuario.py	Modelo de Usuario (rol, correo, nombre, etc.). Incluye reglas como formato de correo.
facultad.py	Modelo para facultad (nombre, teléfono).
unidad_academica.py	Modelo de unidad académica (nombre, facultad_id).
programa_academico.py	Modelo de programa académico (nombre, facultad_id).
lugar.py	Modelo de lugar del evento (ubicación, tipo, capacidad).
evento.py	Modelo de evento (nombre, fechas, estado, aval, lugar, facultad_id).

organizacion_externa.py	Modelo de organización externa (representante, sector, contacto, etc.).
src/crud/	Contiene la lógica CRUD real de FastAPI.
usuario_crud.py	Crear, leer, actualizar y eliminar usuarios en MongoDB usando Motor.
facultad_crud.py	CRUD de facultad.
unidad_academica_crud.py	CRUD de unidad académica.
programa_academico_crud.py	CRUD de programas académicos.
lugar_crud.py	CRUD de lugares.
evento_crud.py	CRUD completo de eventos.
organizacion_externa_crud.py	CRUD de organizaciones externas.
src/routes/	Endpoints REST de FastAPI.
usuario_router.py	Expone /usuarios: GET, POST, PUT, DELETE.
facultad_router.py	Expone /facultad.
unidad_academica_router.py	Expone /unidad-academica.
programa_academico_router.py	Expone /programa-academico.
lugar_router.py	Expone /lugar.
evento_router.py	Expone /eventos.
organizacion_externa_router.py	Expone /organizacion-externa.
main.py	Punto de entrada de FastAPI. Registra todos los routers y define la raíz /.

### 5.1.3 Archivos Adicionales

Archivo	Función
.env	Variables de entorno (MONGO_URI, MONGO_DB).
requirements.txt	Lista de dependencias: FastAPI, Motor, Uvicorn, Pydantic, python-dotenv.

## 5.2 EJECUCION ENTORNO VIRTUAL

- Instalar virtualenv
  - pip install virtualenv
- Crea el entorno virtual
  - py -3.11 -m venv sigeu2\_venv
- Activa el entorno
  - sigeu2\_venv\Scripts\activate

```
Directorio: D:\SIGEU\SIGEU_2_MONGO

Mode                LastWriteTime         Length Name
----                -
d-----          17/11/2025   7:23 p. m.         docs
d-----          17/11/2025   6:59 p. m.         src
-a-----          17/11/2025   6:58 p. m.           53 .env
-a-----          6/10/2025   9:20 p. m.    2224946 ARCHIVO_PASO_A_PASO_EJECUCION_SIGEU.docx
-a-----          17/11/2025   6:58 p. m.           45 requirements.txt

● PS D:\SIGEU\SIGEU_2_MONGO> pip install virtualenv
>>
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: virtualenv in c:\users\data\appdata\roaming\python\python313\site-packages (20.34.0)
Requirement already satisfied: distlib<1,>=0.3.7 in c:\users\data\appdata\roaming\python\python313\site-packages (from virtualenv) (0.4.0)
Requirement already satisfied: filelock<4,>=3.12.2 in c:\users\data\appdata\roaming\python\python313\site-packages (from virtualenv) (3.19.1)
Requirement already satisfied: platformdirs<5,>=3.9.1 in c:\users\data\appdata\roaming\python\python313\site-packages (from virtualenv) (4.3.8)
● PS D:\SIGEU\SIGEU_2_MONGO> py -3.11 -m venv sigeu2_venv
>>
● PS D:\SIGEU\SIGEU_2_MONGO> sigeu2_venv\Scripts\activate
>>
○ (sigeu2_venv) PS D:\SIGEU\SIGEU_2_MONGO> |
```

## 5.3 INSTALAR DEPENDENCIAS DEL BACKEND

- Actualizar pip
  - python -m pip install --upgrade pip
- Instalar todo desde requirements.txt
  - pip install -r requirements.txt

- pip install email-validator

#### 5.4 REVISAR EL ARCHIVO .env

```
MONGO_URI=mongodb://localhost:27017  
MONGO_DB=sigeu_2
```

#### 5.5 VERIFICAR LA CONEXIÓN EN src/db/client.py

```
from motor.motor_asyncio import AsyncIOMotorClient  
from dotenv import load_dotenv  
import os  
  
load_dotenv()  
  
MONGO_URI = os.getenv("MONGO_URI", "mongodb://localhost:27017")  
MONGO_DB = os.getenv("MONGO_DB", "sigeu_2")  
  
client = AsyncIOMotorClient(MONGO_URI)  
db = client[MONGO_DB]
```

#### 5.6 LEVANTAR EL BACKEND CON UVICORN

- uvicorn src.main:app --reload

```
>> geu2_venv) PS D:\SIGEU\SIGEU_2_MONGO>
INFO: Will watch for changes in these directories: ['D:\\SIGEU\\SIGEU_2_MONGO']
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO: Started reloader process [3104] using StatReload
INFO: Started server process [22520]
INFO: Waiting for application startup.
INFO: Application startup complete.
```

## 5.7 ABRIR SWAGGER UI Y VERIFICAR ENDPOINTS

- <http://127.0.0.1:8000/docs>

```
(sigeu2_venv) PS D:\SIGEU\SIGEU_2_MONGO> uvicorn src.main:app --reload
INFO: Will watch for changes in these directories: ['D:\\SIGEU\\SIGEU_2_MONGO']
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO: Started reloader process [23540] using StatReload
INFO: Started server process [21928]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: 127.0.0.1:64780 - "GET / HTTP/1.1" 307 Temporary Redirect
INFO: 127.0.0.1:64780 - "GET /docs HTTP/1.1" 200 OK
INFO: 127.0.0.1:64780 - "GET /openapi.json HTTP/1.1" 200 OK
```

← → ↺ 🏠  🔍 ☆ ⬇️ ⌂

### SIGEU\_2 API – Sistema de Gestión de Eventos UAO 1.0.0 OAS 3.1

/openapi.json

PROYECTO SISTEMA DE GESTIÓN DE EVENTOS SIGEU  
BASE DE DATOS DOCUMENTAL

Integrantes del equipo

- RODRIGO ANDRÉS GÓMEZ LÓPEZ. Código: 2247014
- DAVID HERNÁNDEZ PAZ. Código: 2247003
- DANIEL ALEXANDER BRAND GARCÍA. Código: 2246133
- SEBASTIÁN MANRIQUE MEJÍA. Código: 2246988
- MICHAEL MACOWLI CARDONA RODRIGUEZ. Código: 2246268

UNIVERSIDAD AUTÓNOMA DE OCCIDENTE  
Ingeniería de Datos e Inteligencia Artificial  
Santiago de Cali – 2025

#### Eventos

GET	/eventos/	List Eventos Endpoint	⌵
POST	/eventos/	Create Evento Endpoint	⌵
GET	/eventos/{item_id}	Get Evento Endpoint	⌵
PUT	/eventos/{item_id}	Update Evento Endpoint	⌵
DELETE	/eventos/{item_id}	Delete Evento Endpoint	⌵

## 5.7.1 PRUEBA DE EVENTOS

Eventos		^
GET	/eventos/ List Eventos Endpoint	▼
POST	/eventos/ Create Evento Endpoint	▼
GET	/eventos/{item_id} Get Evento Endpoint	▼
PUT	/eventos/{item_id} Update Evento Endpoint	▼
DELETE	/eventos/{item_id} Delete Evento Endpoint	▼

### 5.7.1.1 Listar todos los eventos (GET /eventos/)

GET /eventos/ List Eventos Endpoint

Parameters

Cancel

No parameters

ExecuteClear

Responses

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:8000/eventos/' \
  -H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:8000/eventos/
```

Server response

CodeDetails

200

Response body

```
{
  "id": "691ba800c3779517a973527b",
  "nombre": null,
  "descripcion": "Aplicaciones de IA",
  "fecha_inicio": null,
  "fecha_fin": null,
  "lugar_id": "None",
  "facultad_id": "691ba800c3779517a9735241",
  "unidad_academica_id": "None",
  "programa_academico_id": "None",
  "organizacion_externa_id": null
}
```

### 5.7.1.2 Crear Evento Endpoint

- Antes de crear: conseguir IDs válidos, para ello necesitamos usar ObjectId reales de las colecciones relacionadas; para lo anterior ejecutamos este código en mongosh:



```
use('sigeu_2');

// Lugares
db.lugar.find({}, { _id:1, nombre:1 }).pretty();

// Facultades
db.facultad.find({}, { _id:1, nombre:1 }).pretty();

// Unidades académicas
db.unidad_academica.find({}, { _id:1, nombre:1 }).pretty();

// Programas académicos
db.programa_academico.find({}, { _id:1, nombre:1 }).pretty();

// Organizaciones externas (opcional)
db.organizacion_externa.find({}, { _id:1, nombre:1 }).pretty();
```

Y esta es la salida:

```
[
  { _id: ObjectId('691ba800c3779517a973528a'), nombre: 'TechEdu SAS'
},
  {
    _id: ObjectId('691ba800c3779517a973528b'),
    nombre: 'DataLab Consulting'
  },
  {
    _id: ObjectId('691ba800c3779517a973528c'),
    nombre: 'Emprendimientos Futuro'
  },
  { _id: ObjectId('691ba800c3779517a973528d'), nombre: 'Deportes UAO'
},
]
```

```
{
  _id: ObjectId('691ba800c3779517a973528e'),
  nombre: 'EcoCircular ONG'
}
]
```

Organizaciones externas (por ejemplo):

```
TechEdu SAS : 691ba800c3779517a973528a
DataLab Consulting : 691ba800c3779517a973528b
Emprendimientos Futuro : 691ba800c3779517a973528c
Deportes UAO : 691ba800c3779517a973528d
EcoCircular ONG : 691ba800c3779517a973528e
```

Salidas a modo general:

```
sigeu_2> db.lugar.find({}, { _id:1, nombre:1 }).pretty();
[
  { _id: ObjectId('691ba800c3779517a9735276') },
  { _id: ObjectId('691ba800c3779517a9735277') },
  { _id: ObjectId('691ba800c3779517a9735278') },
  { _id: ObjectId('691ba800c3779517a9735279') },
  { _id: ObjectId('691ba800c3779517a973527a') }
]
```

```
sigeu_2> db.facultad.find({}, { _id:1, nombre:1 }).pretty();
[
  {
    _id: ObjectId('691ba800c3779517a9735241'),
    nombre: 'Facultad de Ingeniería'
  },
  {
    _id: ObjectId('691ba800c3779517a9735242'),
```

```
    nombre: 'Facultad de Ciencias Económicas'
  }
]
```

```
sigeu_2> db.unidad_academica.find({}, { _id:1, nombre:1 }).pretty();
[
  {
    _id: ObjectId('691ba800c3779517a9735243'),
    nombre: 'Unidad Académica Ingeniería de Sistemas'
  },
  {
    _id: ObjectId('691ba800c3779517a9735244'),
    nombre: 'Unidad Académica Ingeniería Industrial'
  },
  {
    _id: ObjectId('691ba800c3779517a9735245'),
    nombre: 'Unidad Académica Administración de Empresas'
  }
]
```

```
sigeu_2> db.programa_academico.find({}, { _id:1, nombre:1
}).pretty();
[
  {
    _id: ObjectId('691ba800c3779517a9735246'),
    nombre: 'Ingeniería de Sistemas'
  },
  {
    _id: ObjectId('691ba800c3779517a9735247'),
    nombre: 'Ingeniería Industrial'
  },
]
```

```
{
  _id: ObjectId('691ba800c3779517a9735248'),
  nombre: 'Administración de Empresas'
},
{
  _id: ObjectId('691ba800c3779517a9735249'),
  nombre: 'Contaduría Pública'
}
]
```

```
sigeu_2> db.organizacion_externa.find({}, {_id:1, nombre:1
}).pretty();
[
  { _id: ObjectId('691ba800c3779517a973528a'), nombre: 'TechEdu
SAS' },
  {
    _id: ObjectId('691ba800c3779517a973528b'),
    nombre: 'DataLab Consulting'
  },
  {
    _id: ObjectId('691ba800c3779517a973528c'),
    nombre: 'Emprendimientos Futuro'
  },
  { _id: ObjectId('691ba800c3779517a973528d'), nombre: 'Deportes
UAO' },
  {
    _id: ObjectId('691ba800c3779517a973528e'),
    nombre: 'EcoCircular ONG'
  }
]
```

Con los datos anteriores verificados, se crea un json de referencia para crear un nuevo evento:

```
{
  "nombre": "Foro de Innovación en Ingeniería",
  "descripcion": "Evento de prueba creado desde Swagger para validar el CRUD con MongoDB.",
  "fecha_inicio": "2025-03-20T09:00:00",
  "fecha_fin": "2025-03-20T12:00:00",
  "lugar_id": "691ba800c3779517a9735276",
  "facultad_id": "691ba800c3779517a9735241",
  "unidad_academica_id": "691ba800c3779517a9735243",
  "programa_academico_id": "691ba800c3779517a9735246",
  "organizacion_externa_id": "691ba800c3779517a973528a"
}
```

Request body required

Edit Value | Schema

```
{
  "nombre": "Foro de Innovación en Ingeniería",
  "descripcion": "Evento de prueba creado desde Swagger para validar el CRUD con MongoDB.",
  "fecha_inicio": "2025-03-20T09:00:00",
  "fecha_fin": "2025-03-20T12:00:00",
  "lugar_id": "691ba800c3779517a9735276",
  "facultad_id": "691ba800c3779517a9735241",
  "unidad_academica_id": "691ba800c3779517a9735243",
  "programa_academico_id": "691ba800c3779517a9735246",
  "organizacion_externa_id": "691ba800c3779517a973528a"
}
```

Respuesta correcta: 201 created

Responses

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:8000/eventos/' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "nombre": "Foro de Innovación en Ingeniería",
    "descripcion": "Evento de prueba creado desde Swagger para validar el CRUD con MongoDB.",
    "fecha_inicio": "2025-03-20T09:00:00",
    "fecha_fin": "2025-03-20T12:00:00",
    "lugar_id": "691ba800c3779517a9735276",
    "facultad_id": "691ba800c3779517a9735241",
    "unidad_academica_id": "691ba800c3779517a9735243",
    "programa_academico_id": "691ba800c3779517a9735246",
    "organizacion_externa_id": "691ba800c3779517a973528a"
  }'
```

Request URL

http://127.0.0.1:8000/eventos/

Server response

Code Details

201

Response body

```
{
  "id": "691bf9da4d50c47bf0a18465",
  "nombre": "Foro de Innovación en Ingeniería",
  "descripcion": "Evento de prueba creado desde Swagger para validar el CRUD con MongoDB.",
  "fecha_inicio": "2025-03-20T09:00:00",
  "fecha_fin": "2025-03-20T12:00:00",
  "lugar_id": "691ba800c3779517a9735276",
  "facultad_id": "691ba800c3779517a9735241",
  "unidad_academica_id": "691ba800c3779517a9735243",
  "programa_academico_id": "691ba800c3779517a9735246",
  "organizacion_externa_id": "691ba800c3779517a973528a"
}
```

Response headers

```
access-control-allow-credentials: true
access-control-allow-origin: *
content-length: 470
content-type: application/json
date: Tue, 18 Nov 2025 04:45:14 GMT
server: uvicorn
```

```
INFO: 127.0.0.1:62888 - "GET /docs HTTP/1.1" 200 OK
INFO: 127.0.0.1:62888 - "GET /openapi.json HTTP/1.1" 200 OK
INFO: 127.0.0.1:62888 - "GET /eventos/ HTTP/1.1" 200 OK
INFO: 127.0.0.1:62891 - "POST /eventos/ HTTP/1.1" 201 Created
```

```
{
  "id": "691bf9da4d50c47bf0a18465",
  "nombre": "Foro de Innovación en Ingeniería",
  "descripcion": "Evento de prueba creado desde Swagger para validar el CRUD con MongoDB.",
  "fecha_inicio": "2025-03-20T09:00:00",
  "fecha_fin": "2025-03-20T12:00:00",
  "lugar_id": "691ba800c3779517a9735276",
  "facultad_id": "691ba800c3779517a9735241",
  "unidad_academica_id": "691ba800c3779517a9735243",
  "programa_academico_id": "691ba800c3779517a9735246",
  "organizacion_externa_id": "691ba800c3779517a973528a"
}
```

### 5.7.1.3 Consultar por Id el evento creado

Para este ejemplo usamos el Id del evento creado: 691bf9da4d50c47bf0a18465

GET

/eventos/{item\_id} Get Evento Endpoint

Parameters

Name	Description
<b>item_id</b> * required string (path)	<input type="text" value="691bf9da4d50c47bf0a18465"/>

Execute

GET

/eventos/{item\_id} Get Evento Endpoint

Cancel

Parameters

Name	Description
<b>item_id</b> * required string (path)	<input type="text" value="691bf9da4d50c47bf0a18465"/>

Execute

Clear

Responses

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:8000/eventos/691bf9da4d50c47bf0a18465' \
  -H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:8000/eventos/691bf9da4d50c47bf0a18465
```

Server response

Code	Details
200	<div>Response body</div> <pre>{   "id": "691bf9da4d50c47bf0a18465",   "nombre": "Foro de Innovación en Ingeniería",   "descripcion": "Evento de prueba creado desde Swagger para validar el CRUD con MongoDB.",   "fecha_inicio": "2025-03-20T00:00:00",   "fecha_fin": "2025-03-20T12:00:00",   "lugar_id": "691ba80c3779517a9735276", </pre>

#### 5.7.1.4 Actualizar evento por id.

Para este ejemplo tomamos el evento que creamos anteriormente, con su id: 691bf9da4d50c47bf0a18465. En este caso, cambiamos el nombre, la descripción y las fechas (aunque se podría cambiar solo una parte del evento):

```
{
```

```
"nombre": "Foro de Innovación en Ingeniería - Versión Actualizada",
"descripcion": "Evento actualizado para probar el endpoint PUT. Incluye nuevas ponencias sobre IA y Analítica de Datos.",
"fecha_inicio": "2025-03-21T08:00:00",
"fecha_fin": "2025-03-21T12:30:00",
"lugar_id": "691ba800c3779517a9735276",
"facultad_id": "691ba800c3779517a9735241",
"unidad_academica_id": "691ba800c3779517a9735243",
"programa_academico_id": "691ba800c3779517a9735246",
"organizacion_externa_id": "691ba800c3779517a973528a"
}
```

**PUT** /eventos/{item\_id} Update Evento Endpoint

**Parameters**

Name	Description
<b>item_id</b> <small>★ required</small> string (path)	<input type="text" value="691bf9da4d50c47bf0a18465"/>

**Request body** required

Edit Value | Schema

```
{
  "nombre": "Foro de Innovación en Ingeniería - Versión Actualizada",
  "descripcion": "Evento actualizado para probar el endpoint PUT. Incluye nuevas ponencias sobre IA y Analítica de Datos.",
  "fecha_inicio": "2025-03-21T08:00:00",
  "fecha_fin": "2025-03-21T12:30:00",
  "lugar_id": "691ba800c3779517a9735276",
  "facultad_id": "691ba800c3779517a9735241",
  "unidad_academica_id": "691ba800c3779517a9735243",
  "programa_academico_id": "691ba800c3779517a9735246",
  "organizacion_externa_id": "691ba800c3779517a973528a"
}
```

```
INFO: 127.0.0.1:62905 - "GET /eventos/691bf9da4d50c47bf0a18465 HTTP/1.1" 200 OK
INFO: 127.0.0.1:57641 - "PUT /eventos/691bf9da4d50c47bf0a18465 HTTP/1.1" 200 OK
INFO: 127.0.0.1:57645 - "PUT /eventos/691bf9da4d50c47bf0a18465 HTTP/1.1" 200 OK
```



```
{
  "id": "691bf9da4d50c47bf0a18465",
  "nombre": "Foro de Innovación en Ingeniería - Versión Actualizada",
  "descripcion": "Evento actualizado para probar el endpoint PUT. Incluye nuevas ponencias sobre IA y Analítica de Datos.",
  "fecha_inicio": "2025-03-21T08:00:00",
  "fecha_fin": "2025-03-21T12:30:00",
  "lugar_id": "691ba800c3779517a9735276",
  "facultad_id": "691ba800c3779517a9735241",
  "unidad_academica_id": "691ba800c3779517a9735243",
  "programa_academico_id": "691ba800c3779517a9735246",
  "organizacion_externa_id": "691ba800c3779517a973528a"
}
```

#### Response headers

```
content-length: 1901
content-type: application/json
date: Tue, 18 Nov 2025 05:02:07 GMT
server: uvicorn
```

### 5.7.1.5 Eliminar evento por id.

Usaremos para este punto, el evento que creamos, con el id:

691bf9da4d50c47bf0a18465

**DELETE** `/eventos/{item_id}` Delete Evento Endpoint

**Parameters**

Name	Description
<b>item_id</b> * required	
string	
(path)	

691bf9da4d50c47bf0a18465|

## Responses

### Curl

```
curl -X 'DELETE' \  
  'http://127.0.0.1:8000/eventos/691bf9da4d50c47bf0a18465' \  
  -H 'accept: application/json'
```

### Request URL

```
http://127.0.0.1:8000/eventos/691bf9da4d50c47bf0a18465
```

### Server response

Code	Details
------	---------

200	
-----	--

#### Response body

```
{  
  "message": "Evento eliminado correctamente"  
}
```

#### Response headers

```
access-control-allow-credentials: true  
access-control-allow-origin: *  
content-length: 44  
content-type: application/json  
date: Tue, 18 Nov 2025 05:06:22 GMT  
server: uvicorn
```

```
INFO: 127.0.0.1:57645 - "PUT /eventos/691bf9da4d50c47bf0a18465 HTTP/1.1" 200 OK  
INFO: 127.0.0.1:57646 - "GET /eventos/ HTTP/1.1" 200 OK  
INFO: 127.0.0.1:57660 - "DELETE /eventos/691bf9da4d50c47bf0a18465 HTTP/1.1" 200 OK
```

#### 5.7.1.5 VALIDAR CREACION DE EVENTO POR ROL

```
{
  "creador_id": "691ba800c3779517a9735254",
  "nombre": "Seminario de Análítica Educativa",
  "descripcion": "Evento de prueba creado por docente",
  "fecha_inicio": "2025-11-20T09:00:00",
  "fecha_fin": "2025-11-20T11:00:00",
  "lugar_id": "691ba800c3779517a9735276",
  "facultad_id": "691ba800c3779517a9735241",
  "unidad_academica_id": "691ba800c3779517a9735243",
  "programa_academico_id": "691ba800c3779517a9735246",
  "organizacion_externa_id": "691ba800c3779517a973528a"
}
```

ESTE FUNCIONA BIEN

```
{
  "creador_id": "691ba800c3779517a973525c",
  "nombre": "Seminario de Análítica Educativa-POR SECRETARIA",
  "descripcion": "Evento de prueba creado por docente",
  "fecha_inicio": "2025-11-20T09:00:00",
  "fecha_fin": "2025-11-20T11:00:00",
  "lugar_id": "691ba800c3779517a9735276",
  "facultad_id": "691ba800c3779517a9735241",
  "unidad_academica_id": "691ba800c3779517a9735243",
  "programa_academico_id": "691ba800c3779517a9735246",
  "organizacion_externa_id": "691ba800c3779517a973528a"
}
```

POST

/eventos/ Create Evento Endpoint

Parameters

No parameters

Request body required

Edit Value | Schema

```
{
  "creador_id": "691ba800c3779517a973525c",
  "nombre": "Seminario de Analítica Educativa-POR SECRETARIA",
  "descripcion": "Evento de prueba creado por docente",
  "fecha_inicio": "2025-11-20T09:00:00",
  "fecha_fin": "2025-11-20T11:00:00",
  "lugar_id": "691ba800c3779517a9735276",
  "facultad_id": "691ba800c3779517a9735241",
  "unidad_academica_id": "691ba800c3779517a9735243",
  "programa_academico_id": "691ba800c3779517a9735246",
  "organizacion_externa_id": "691ba800c3779517a973528a"
}
```

403  
*Undocumented*

Error: Forbidden

Response body

```
{
  "detail": "Solo usuarios con rol 'docente' o 'estudiante' pueden crear eventos"
}
```

Response headers

```
access-control-allow-credentials: true
access-control-allow-origin: *
content-length: 80
content-type: application/json
date: Tue, 18 Nov 2025 07:27:18 GMT
server: uvicorn
```

Responses

Code	Description
------	-------------

ESTE SACA UN ERROR

```
INFO: 127.0.0.1:61908 - "POST /eventos/ HTTP/1.1" 403 Forbidden
```

## 5.7.2 PRUEBA DE USUARIOS

## 5.7.3 PRUEBA DE FACULTADES

### 5.7.3.1 Listar facultades

## Facultades

**GET** **/facultads/** List Facultads Endpoint

**Parameters**

No parameters

Execute

**Responses**

**Curl**

```
curl -X 'GET' \
  'http://127.0.0.1:8000/facultads/' \
  -H 'accept: application/json'
```

**Request URL**

```
http://127.0.0.1:8000/facultads/
```

**Server response**

Code	Details
200	<div><b>Response body</b></div> <pre>[   {     "_id": "691ba800c3779517a9735241",     "nombre": "Facultad de Ingeniería",     "telefono": "6023456789"   },   {     "_id": "691ba800c3779517a9735242",     "nombre": "Facultad de Ciencias Económicas",     "telefono": "6029876543"   } ]</pre>

```
INFO: 127.0.0.1:52587 - "GET /facultads/ HTTP/1.1" 200 OK
```

#### 5.7.3.2 Crear facultades

```
{  
  "nombre": "Facultad de Ciencias Sociales y Humanas",  
  "telefono": "6025551234"  
}
```

```
INFO: 127.0.0.1:59120 - "POST /facultads/ HTTP/1.1" 201 Created
```

#### 5.7.3.3 Consultar facultades

Id de la facultad creada: 691c0e3a4d50c47bf0a18469

**GET****/facultads/{item\_id}** Get Facultad Endpoint

### Parameters

**Name****Description****item\_id** \* required

string

(path)

691c0e3a4d50c47bf0a18469

**Execute**

### Responses

#### Curl

```
curl -X 'GET' \  
  'http://127.0.0.1:8000/facultads/691c0e3a4d50c47bf0a18469' \  
  -H 'accept: application/json'
```

#### Request URL

```
http://127.0.0.1:8000/facultads/691c0e3a4d50c47bf0a18469
```

#### Server response

**Code****Details**

200

**Response body**

```
{  
  "_id": "691c0e3a4d50c47bf0a18469",  
  "nombre": "Facultad de Ciencias Sociales y Humanas",  
  "telefono": "6025551234"  
}
```

#### 5.7.3.4 Actualizar facultades

Id de la facultad creada: 691c0e3a4d50c47bf0a18469

```
{  
  "nombre": "Facultad de Artes, Humanidades y Diseño 3D"  
}
```

**PUT** `/facultads/{item_id}` Update Facultad Endpoint

**Parameters**

Name	Description
<b>item_id</b> * required string (path)	691c0e3a4d50c47bf0a18469

**Request body** required

Edit Value | Schema

```
{  
  "nombre": "Facultad de Artes, Humanidades y Diseño 3D"  
}
```

200

Response body

```
{  
  "_id": "691c0e3a4d50c47bf0a18469",  
  "nombre": "Facultad de Artes, Humanidades y Diseño 3D",  
  "telefono": "6025551234"  
}
```

Response headers

```
access-control-allow-credentials: true  
access-control-allow-origin: *  
content-length: 113  
content-type: application/json  
date: Tue, 18 Nov 2025 06:16:12 GMT  
server: uvicorn
```

**INFO:** 127.0.0.1:59223 - "PUT /facultads/691c0e3a4d50c47bf0a18469 HTTP/1.1" 200 OK



### 5.7.3.5 Eliminar facultades

Id a eliminar: 691c0e3a4d50c47bf0a18469

**DELETE** /facultads/{item\_id} Delete Facultad Endpoint

**Parameters**

Name	Description
<b>item_id</b> * required string (path)	<input type="text" value="691c0e3a4d50c47bf0a18469"/>

Execute

**Request URL**

http://127.0.0.1:8000/facultads/691c0e3a4d50c47bf0a18469

**Server response**

Code	Details
200	<div><b>Response body</b></div> <pre>{   "msg": "Facultad eliminado" }</pre> <div><b>Response headers</b></div> <pre>access-control-allow-credentials: true access-control-allow-origin: * content-length: 28 content-type: application/json date: Tue, 18 Nov 2025 06:18:25 GMT server: uvicorn</pre>

INFO: 127.0.0.1:59278 - "DELETE /facultads/691c0e3a4d50c47bf0a18469 HTTP/1.1" 200 OK

## ANEXO 2. ACCESO GITHUB – BACKEND

[https://github.com/aglsystems-ragl/SIGEU\\_2\\_MONGO.git](https://github.com/aglsystems-ragl/SIGEU_2_MONGO.git)

## ANEXO 3. ACCESO PRUEBAS POSTMAN

<https://andresgomezdoctorado-8529723.postman.co/workspace/andres-gomez's-Workspace~25a1ed28-155c-4f0f-8243-ddf433ed08ec/collection/49033182-aac2ae40-a91f-440d-a8c1-c7774f89d74d?action=share&source=copy-link&creator=49033182>