



Universidad de las Fuerzas Armadas - ESPE

Departamento de Ciencias de la Computación

Carrera: Ingeniería de Software

Asignatura: Pruebas de Software– **NRC22431**

Tema: Prueba parcial 2

Estudiante

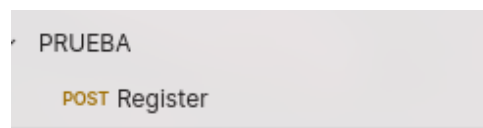
Bonilla Hidalgo Jairo Smith

Fecha: 10/07/2025

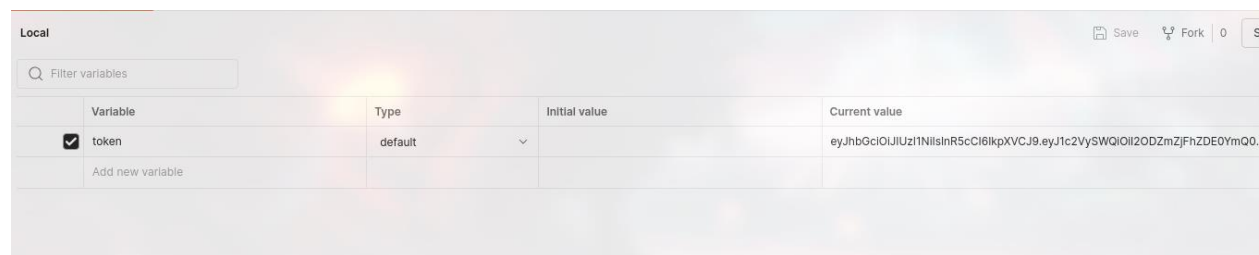
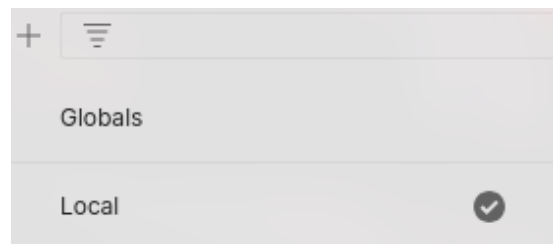
- Registro y consulta de estudiantes (/api/students)

Primero nos autenticamos, ocupamos un método post con la siguiente endpoint

<http://localhost:3000/api/auth/login>



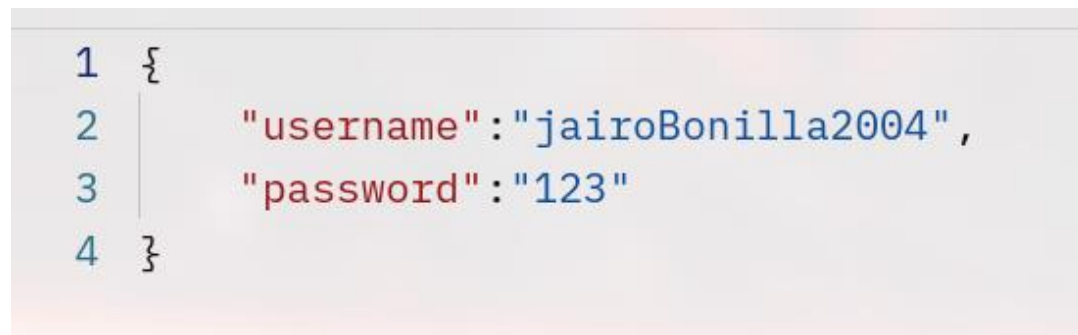
Paso 2 creamos un Enviroment con el nombre de local para guardar el token

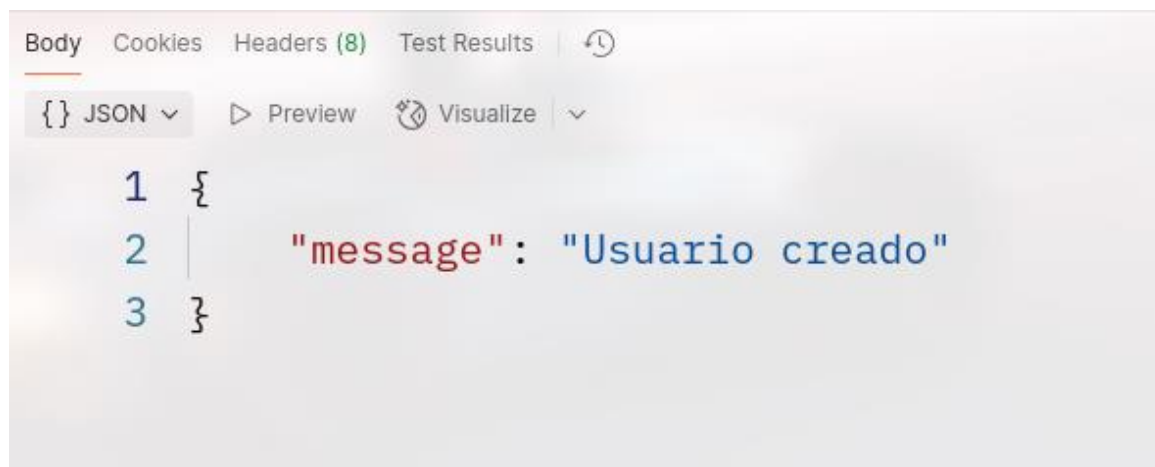


En Scripts ponemos lo siguiente:

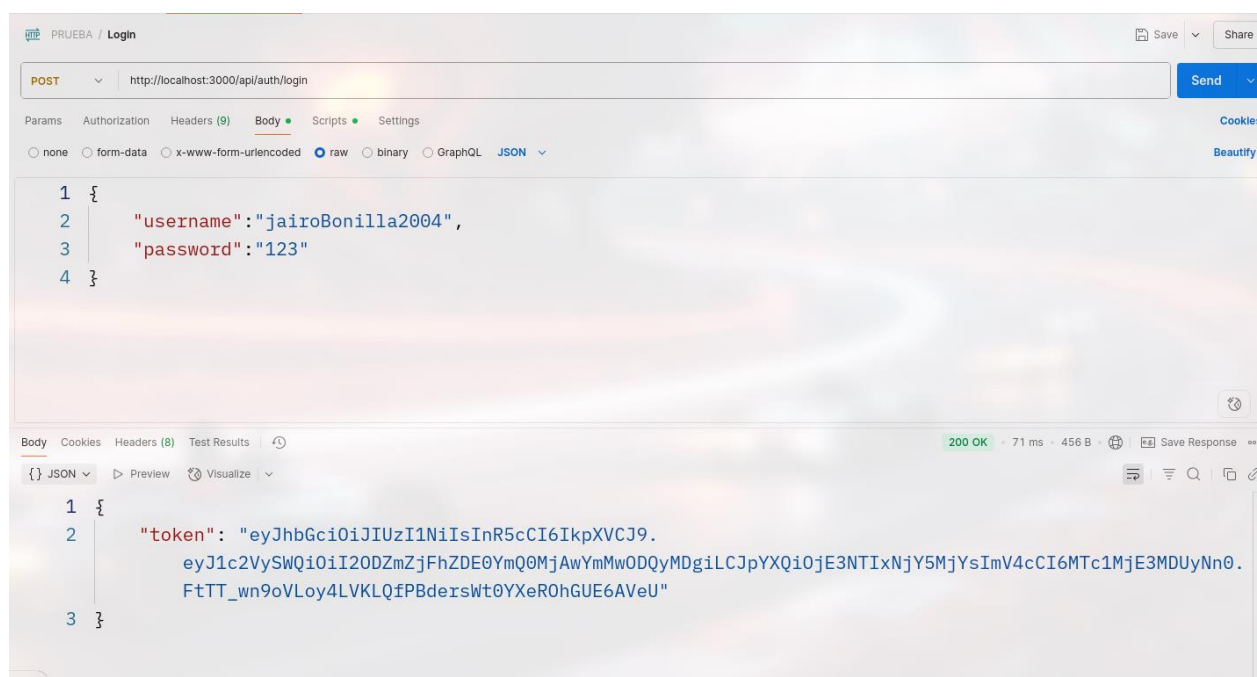


Y mandamos como cuerpo de solicitud el siguiente JSON





PASO 2 NOS LOGEAMOS, CON EL USUARIO CREADO



```

users
academico> db.users.find()
[
  {
    _id: ObjectId('686ff1ad14bd4200bc084208'),
    username: 'jairoBonilla',
    password: '$2a$10$.efJCYq05/oovIdQPmI7B0rZMFaf4AzJBBrnhEEEnzEB50ki4a1JHE0',
    __v: 0
  },
  {
    _id: ObjectId('686ff4a314bd4200bc08420e'),
    username: 'jairoBonilla2004',
    password: '$2a$10$JL6a4WCiarwHra1hij75TuRUN5SvY.5Dt12tw7g9i9nj3lBsTCPS',
    __v: 0
  }
]

```

PRUEBAS

- Registro y consulta de estudiantes (/api/students)

The screenshot shows a REST client interface with a POST request to `http://localhost:3000/api/students`. The request body is a JSON object with the following fields:

```

1 {
2   "firstName": "Jairo",
3   "lastName": "Bonilla",
4   "email": "jairo.bonilla@example.com"
5 }
6

```

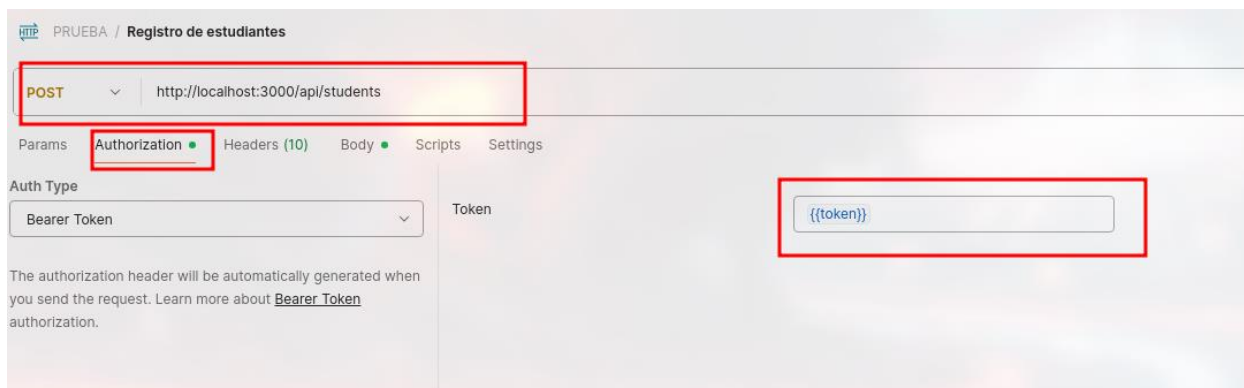
The response body is also a JSON object with the following fields:

```

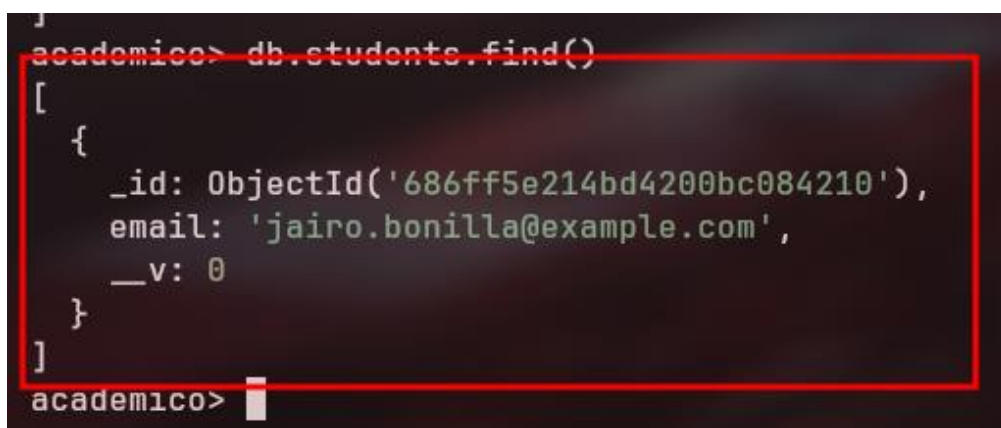
1 {
2   "email": "jairo.bonilla@example.com",
3   "_id": "686ff5e214bd4200bc084210",
4   "__v": 0
5 }

```

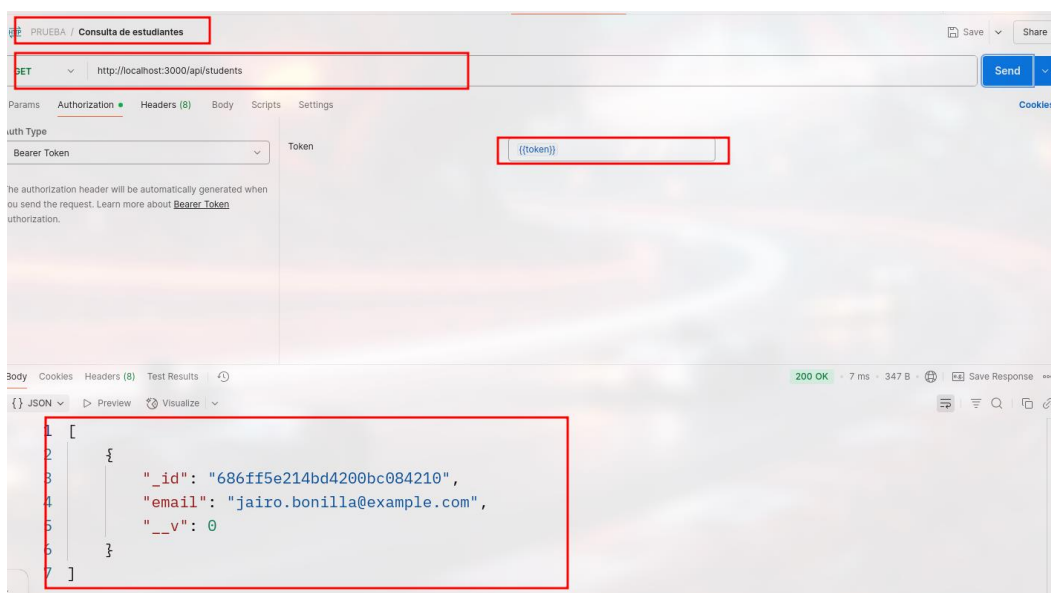
Mandamos el token de autorizacion



Verificamos en el MONGODB que se creo este usuario



CONSULTA DE ESTUDIANTES



REGISTRO DE CURSOS

Solicitud POST

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:3000/api/courses`
- Method:** `POST`
- Body (JSON):**

```
1 {
2   "name": "Introducción a Node.js",
3   "description": "Curso básico de desarrollo backend"
4 }
```
- Response (JSON):**

```
1 {
2   "description": "Curso básico de desarrollo backend",
3   "_id": "686ff7a814bd4200bc084213",
4   "__v": 0
5 }
```
- Status:** `201 Created`
- Time:** 7 ms
- Size:** 366 B

Consulta de cursos, solicitud tipo GET

The screenshot shows a REST client interface with the following details:

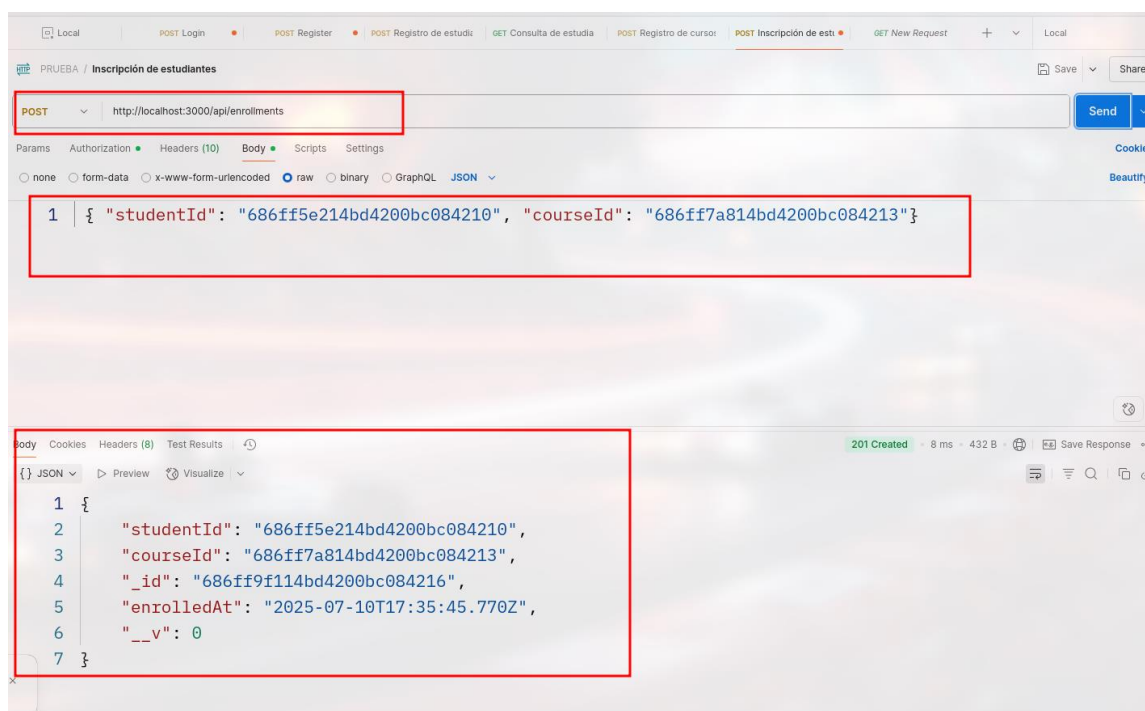
- URL:** `http://localhost:3000/api/courses`
- Method:** `GET`
- Authorization:** Bearer Token
- Token:** `{{token}}`
- Body (JSON):**

```
1 [
2   {
3     "_id": "686ff7a814bd4200bc084213",
4     "description": "Curso básico de desarrollo backend",
5     "__v": 0
6   }
7 ]
```
- Status:** `200 OK`
- Time:** 6 ms
- Size:** 363 B

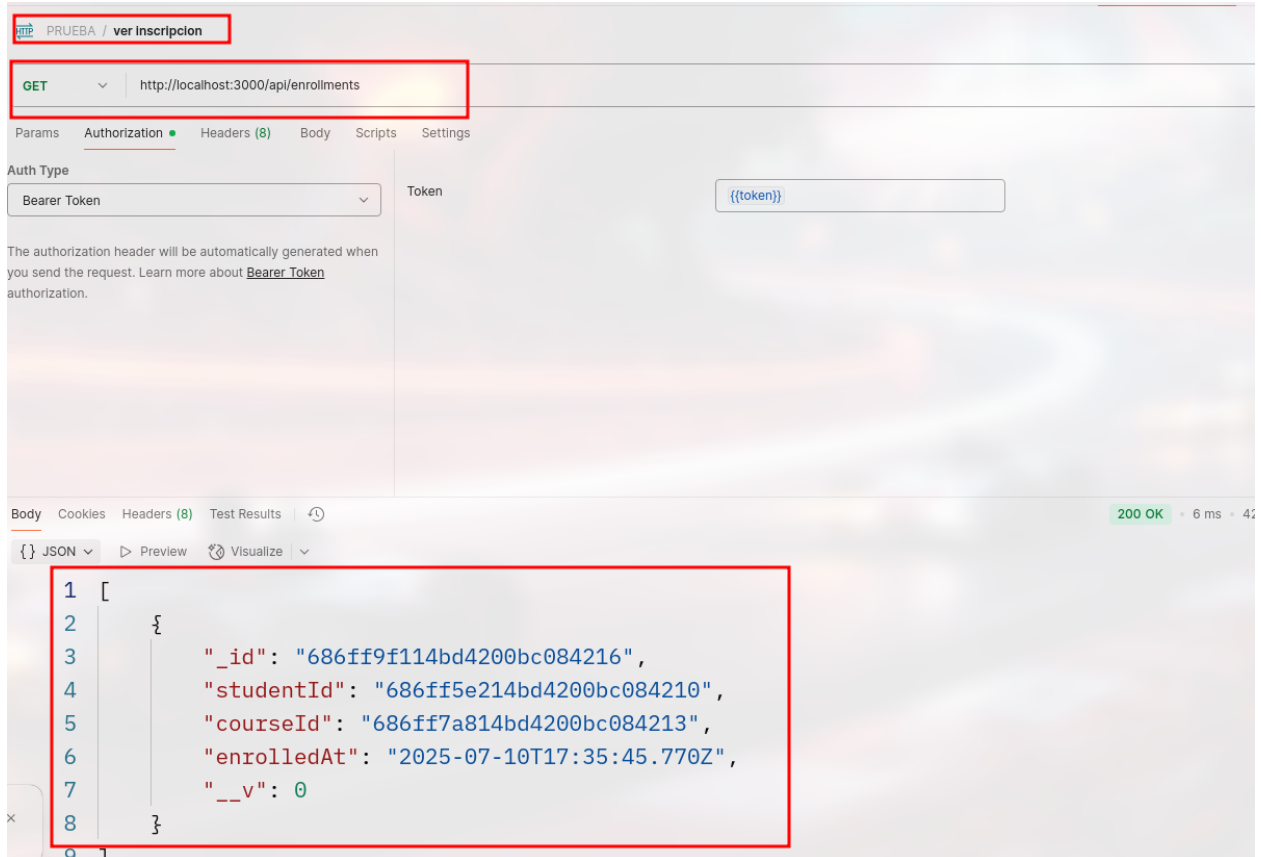
VERIFICAMOS QUE SE GUARDE EN EL MONGO

```
academico> db.courses.find()
[
  {
    _id: ObjectId('686ff7a814bd4200bc084213'),
    description: 'Curso básico de desarrollo backend',
    __v: 0
  }
]
academico>
```

- **Inscripción de estudiantes**
(/api/enrollments)



Verificamos con un método GET



Verificamos que se registro en MONGODB

```

academico> db.enrollments.find()
\[
  {
    _id: ObjectId('686ff9f114bd4200bc084216'),
    studentId: '686ff5e214bd4200bc084210',
    courseId: '686ff7a814bd4200bc084213',
    enrolledAt: ISODate('2025-07-10T17:35:45.770Z'),
    __v: 0
  }
]
academico> \

```

2. Pruebas de regresión - CAMBIO LÓGICO

- Aplica este cambio en `enrollmentController.js`:

// Antes (correcto)

```
backend_academico_completo_V2 > controllers > enrollmentController.js > create > create
1  const Enrollment = require('../models/Enrollment');
2  exports.getAll = async (_, res) => res.json(await Enrollment.find());
3  exports.create = async (req, res) => {
4      const { studentId, courseId } = req.body;
5
6      // LÍNEA CLAVE PARA REGRESIÓN
7      if (!studentId || !courseId) return res.status(400).json({ message: "Campos requeridos" });
8
9      res.status(201).json(await new Enrollment({ studentId, courseId }).save());
10 };
11 exports.delete = async (req, res) => { await Enrollment.findByIdAndDelete(req.params.id); res.json({ m
```

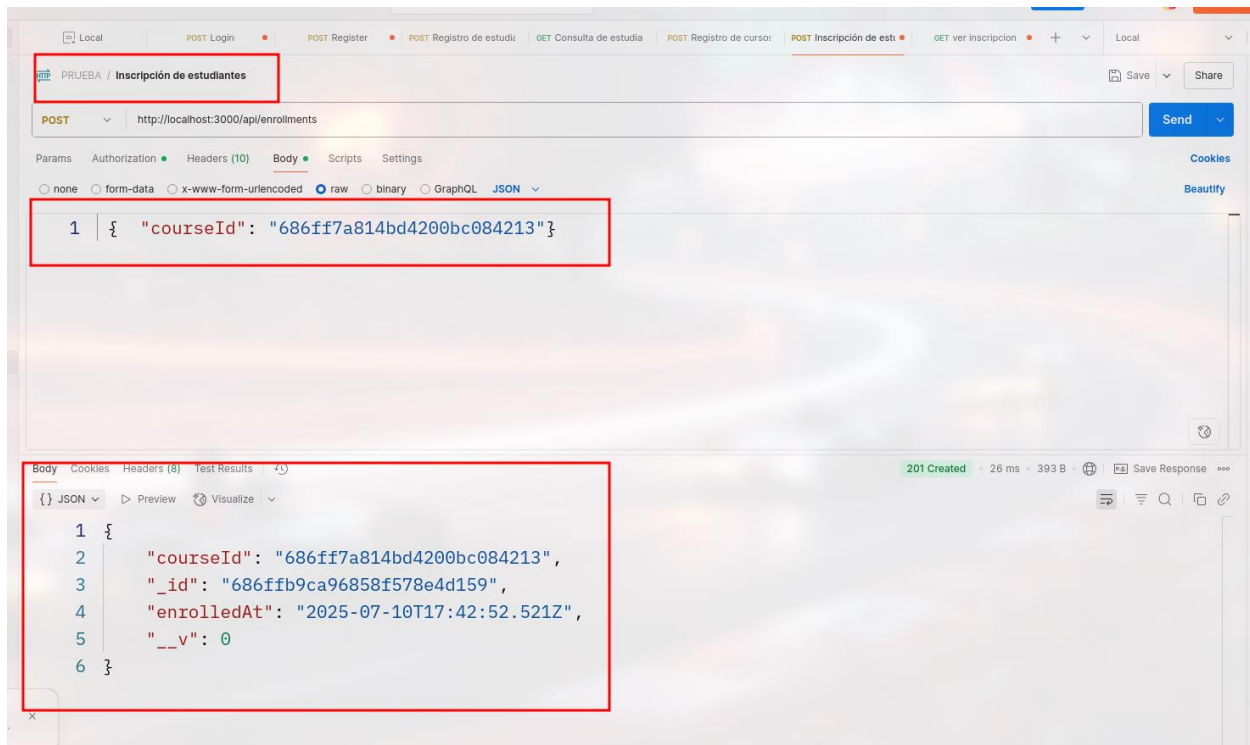
// Después (error intencional)

```
backend_academico_completo_V2 > controllers > enrollmentController.js > create > create
1  const Enrollment = require('../models/Enrollment');
2  exports.getAll = async (_, res) => res.json(await Enrollment.find());
3  exports.create = async (req, res) => {
4      const { studentId, courseId } = req.body;
5
6      // LÍNEA CLAVE PARA REGRESIÓN
7      // if (!studentId || !courseId) return res.status(400).json({ message: "Campos requeridos" });
8
9      res.status(201).json(await new Enrollment({ studentId, courseId }).save());
10 };
11 exports.delete = async (req, res) => { await Enrollment.findByIdAndDelete(req.params.id); res.json({
```

// Se omite la validación

- Ejecuta una prueba de inscripción sin studentId.

Como resultado nos deja registrar pero, registra, solo con los campos enviados en el bpdy en este caso courseId

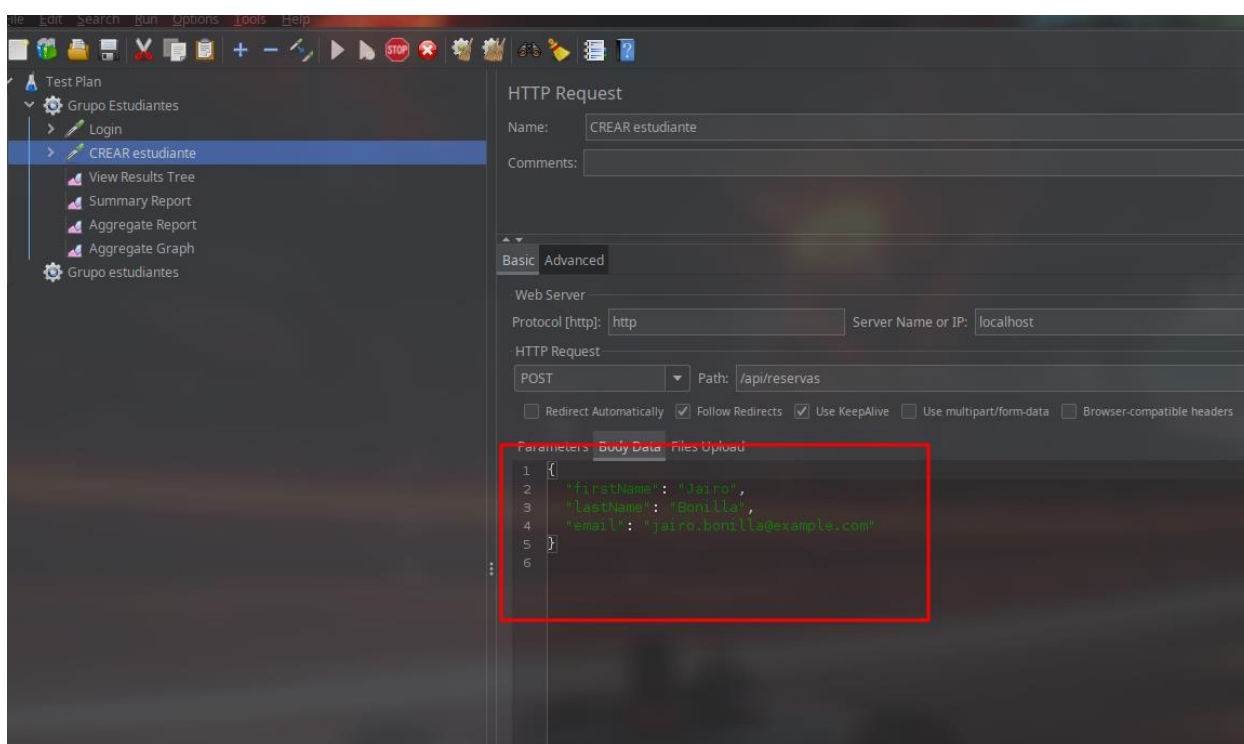
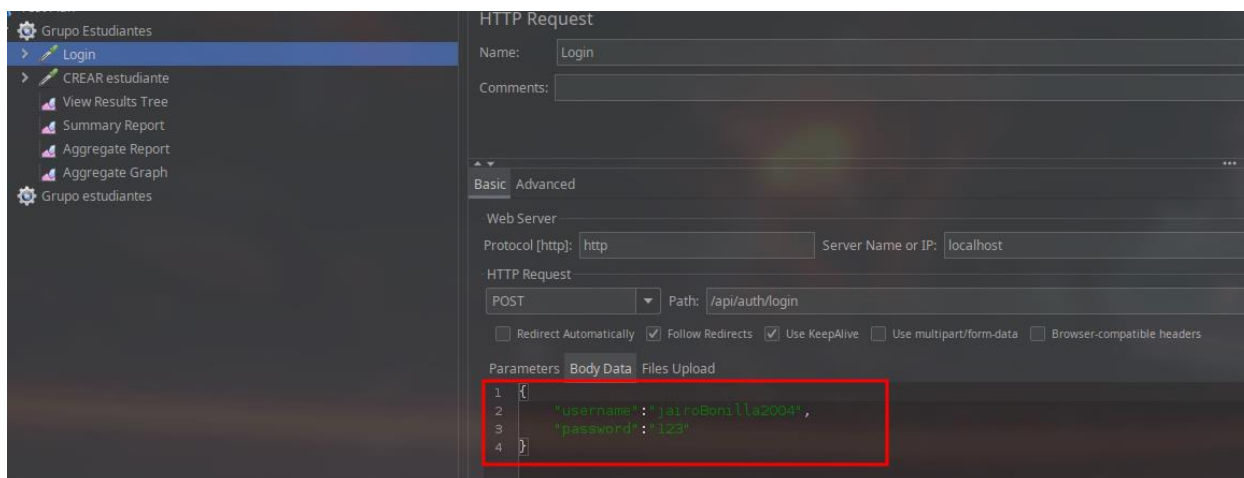


Esta prueba demuestra que un cambio no controlado en el código rompió una funcionalidad que antes estaba bien.

Es decir, la validación de campos requeridos ya no funciona, y esto afecta la integridad de los datos.

3. Pruebas de estrés – JMeter

- Simula 100 estudiantes inscritos a cursos durante 2 minutos.
- Evalúa:
 - Tiempo medio de respuesta
 - Comportamiento del servidor bajo carga
 - Solicitudes fallidas



The screenshot shows the 'Summary Report' in JMeter. The 'Name' field is set to 'Summary Report'. The 'Comments' field is empty. The 'Write results to file / Read from file' section shows 'Filename:'. The 'Log/Display Only' section shows 'Errors', 'Successes', and 'Configure'. The table below shows the test results:

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
Login	250	71	66	183	10.93	0.00%	2.5/sec	1.10	0.59	456.0
CREAR estudiante	259	2	1	17	1.48	38.61%	2.5/sec	0.94	1.15	390.0
TOTAL	519	36	1	183	35.10	19.27%	4.9/sec	2.04	1.74	423.0

