

INSTITUTIO SUPERIOR SUDAMERICANO



Autores:

Jose Gutierrez

ASIGNATURA:

**Despliegue de
diagramas**

CURSO:

Tercer ciclo de software

DOCENTE ENCARGADO:

Ing. Vacacela Jhostin

2025-2026

Introducción

La gestión adecuada de la información es fundamental en un sistema universitario, especialmente cuando se maneja datos críticos de estudiantes, asignaturas y procesos de inscripción. Los principios ACID garantizan que las operaciones con esta información se ejecuten de manera confiable y sin comprometer la integridad de los datos.

Este proyecto implementa los principios ACID específicamente en el módulo de inscripción estudiantil, asegurando transacciones seguras y consistentes.

Atomicidad

Este principio garantiza que una transacción se complete totalmente o no se ejecute en absoluto, evitando estados intermedios o inconsistentes.

Durante el proceso de inscripción, el sistema ejecuta múltiples operaciones:

- Validación del estado activo del estudiante
- Verificación de disponibilidad de cupos en la asignatura
- Creación del registro de matrícula
- Actualización del contador de cupos disponibles

Si cualquiera de estas operaciones falla, el sistema revierte todas las acciones realizadas, previniendo inscripciones parciales o desajustes en el conteo de cupos.

Consistencia

La consistencia mantiene la base de datos en estados válidos antes y después de cada transacción, aplicando reglas de negocio estrictas.

El sistema implementa las siguientes validaciones:

- Los estudiantes deben tener estado activo para poder inscribirse
- Las asignaturas no pueden registrar cupos en valores negativos
- Se valida la existencia de referencias entre entidades relacionadas

Estas restricciones aseguran que cada transacción mantenga la integridad de los datos del sistema.

Aislamiento

El aislamiento previene conflictos cuando múltiples transacciones se ejecutan

simultáneamente sobre los mismos datos.

En escenarios de alta concurrencia, como cuando varios estudiantes se inscriben simultáneamente en la misma asignatura, el sistema:

- Gestiona los cupos mediante control de concurrencia
- Previene condiciones de carrera en la asignación de cupos
- Serializa las operaciones para mantener la coherencia de datos

Esta característica es esencial para garantizar el funcionamiento correcto del sistema con múltiples usuarios concurrentes.

Durabilidad

La durabilidad asegura la persistencia permanente de las transacciones confirmadas, incluso ante fallos del sistema.

Una vez completada exitosamente una inscripción:

- Los datos se almacenan permanentemente en la base de datos
- La información persiste ante interrupciones del servicio
- El registro académico mantiene su integridad a largo plazo

Este principio es crucial para la confiabilidad del historial académico institucional.

Conclusión

La implementación de los principios ACID proporciona al sistema universitario un nivel robusto de confiabilidad e integridad de datos. El proceso de inscripción se beneficia directamente de estas garantías, minimizando errores, preservando la coherencia de la información y protegiendo la integridad transaccional.

La combinación de NestJS, Prisma y PostgreSQL ofrece un entorno óptimo para la aplicación de estos principios, resultando en una arquitectura sólida y escalable para la gestión académica.

```
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/teacher-subjects/:id, GET} route +3ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/teacher-subjects/:id, PATCH} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/teacher-subjects/:id, DELETE} route +0ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RoutesResolver] StudentSubjectController {/student-subjects}: +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/student-subjects, POST} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/student-subjects, GET} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/student-subjects/student/:studentId, GET} route +2ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/student-subjects/subject/:subjectId, GET} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/student-subjects/:id, GET} route +0ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/student-subjects/:id, PATCH} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/student-subjects/:id, DELETE} route +0ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RoutesResolver] EnrollmentController {/enrollments}: +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/enrollments, POST} route +0ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/enrollments/report, GET} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/enrollments/student/:studentId/period/:periodId, GET} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/enrollments, GET} route +2ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/enrollments/:id, GET} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/enrollments/:id, DELETE} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RoutesResolver] AcademicPeriodController {/academic-periods}: +0ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/academic-periods, POST} route +0ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/academic-periods/active, GET} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/academic-periods, GET} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/academic-periods/:id, GET} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/academic-periods/:id, PATCH} route +1ms
[Nest] 22264 - 14/01/2026, 9:45:59 LOG [RouterExplorer] Mapped {/academic-periods/:id, DELETE} route +1ms
```

Sistema Universitario - COM... - Run results

Ran on Dec 05, 2025 at 10:46:27 AM · [View all runs](#)

Source	Environment	Iterations	Duration	All tests	Errors	Avg. Resp. Time
Runner	Local API	1	5s 765ms	0	0	48 ms

All Tests Passed (0) Failed (0) Skipped (0) Errors (0) Console Log

[View Summary](#)

Iteration 1

POST 1. AUTH / 1.1 Register User

http://localhost:3000/auth/register

201 • 416 ms • 587 B •

No tests found

POST 1. AUTH / 1.2 Login

http://localhost:3000/auth/login

201 • 190 ms • 587 B •

No tests found

POST 2. SPECIALTIES / 2.1 Create Specialty

http://localhost:3000/specialties

201 • 144 ms • 367 B •

No tests found

GET 2. SPECIALTIES / 2.2 Get All Specialties

http://localhost:3000/specialties?page=1&limit=10

200 • 107 ms • 448 B •

No tests found

GET 2. SPECIALTIES / 2.3 Get Specialty by ID

http://localhost:3000/specialties/1

200 • 9 ms • 375 B •

No tests found

PATCH 2. SPECIALTIES / 2.4 Update Specialty

http://localhost:3000/specialties/1

200 • 29 ms • 374 B •

No tests found

POST 3. CAREERS / 3.1 Create Career http://localhost:3000/careers No tests found	201 • 30 ms • 563 B •
GET 3. CAREERS / 3.2 Get All Careers http://localhost:3000/careers?page=1&limit=10 No tests found	200 • 7 ms • 623 B •
GET 3. CAREERS / 3.3 Get Career by ID http://localhost:3000/careers/1 No tests found	200 • 112 ms • 586 B •
PATCH 3. CAREERS / 3.4 Update Career http://localhost:3000/careers/1 No tests found	200 • 39 ms • 570 B •
POST 4. CYCLES / 4.1 Create Cycle http://localhost:3000/cycles No tests found	201 • 9 ms • 360 B •
GET 4. CYCLES / 4.2 Get All Cycles http://localhost:3000/cycles?page=1&limit=10 No tests found	200 • 10 ms • 434 B •
GET 4. CYCLES / 4.3 Get Cycle by ID http://localhost:3000/cycles/1 No tests found	200 • 6 ms • 369 B •
PATCH 4. CYCLES / 4.4 Update Cycle http://localhost:3000/cycles/1 No tests found	200 • 18 ms • 367 B •
POST 5. SUBJECTS / 5.1 Create Subject http://localhost:3000/subjects No tests found	201 • 32 ms • 722 B •
GET 5. SUBJECTS / 5.2 Get All Subjects http://localhost:3000/subjects?page=1&limit=10 No tests found	200 • 9 ms • 782 B •
GET 5. SUBJECTS / 5.3 Get Subject by ID http://localhost:3000/subjects/1 No tests found	200 • 126 ms • 745 B •
PATCH 5. SUBJECTS / 5.4 Update Subject http://localhost:3000/subjects/1 No tests found	200 • 45 ms • 729 B •

http://localhost:3000/teachers/1 200 • 38 ms • 453 B •

No tests found

POST 7. STUDENTS / 7.1 Create Student

http://localhost:3000/students 201 • 25 ms • 868 B •

No tests found

GET 7. STUDENTS / 7.2 Get All Students

http://localhost:3000/students?page=1&limit=10 200 • 18 ms • 868 B •

No tests found

GET 7. STUDENTS / 7.3 Get Student by ID

http://localhost:3000/students/1 200 • 18 ms • 889 B •

No tests found

PATCH 7. STUDENTS / 7.4 Update Student

http://localhost:3000/students/1 200 • 39 ms • 887 B •

No tests found

POST 8. TEACHER-SUBJECTS / 8.1 Assign Teacher to Subject

http://localhost:3000/teacher-subjects 201 • 38 ms • 1.199 KB •

No tests found

All Tests Passed (0) Failed (0) Skipped (0) Errors (0) Console Log

[View Summary](#)

1

GET 9. STUDENT-SUBJECTS / 9.2 Get All Student-Subjects

http://localhost:3000/student-subjects?page=1&limit=10 200 • 9 ms • 1.378 KB •

No tests found

GET 9. STUDENT-SUBJECTS / 9.3 Get by Student ID

http://localhost:3000/student-subjects/student/1?page=1&limit=10 200 • 18 ms • 947 B •

No tests found

GET 9. STUDENT-SUBJECTS / 9.4 Get by Subject ID

http://localhost:3000/student-subjects/subject/1?page=1&limit=10 200 • 8 ms • 874 B •

No tests found

DELETE 9. STUDENT-SUBJECTS / 9.6 Delete Student-Subject

http://localhost:3000/student-subjects/1 200 • 28 ms • 378 B •

No tests found

GET 10. USERS / 10.1 Get All Users

http://localhost:3000/users 200 • 118 ms • 415 B •

No tests found

GET 10. USERS / 10.2 Get User by ID

http://localhost:3000/users/1 200 • 5 ms • 358 B •

No tests found