

INFORME – Sistema de Gestión de Reservas de Salas de Estudio

Base de Datos 1 – Universidad Católica del Uruguay (UCU)

Segundo Semestre 2025

Trabajo Obligatorio – Gestión de Salas de Estudio

Integrantes: Esteban Durán, Sebastián Martony y Joaquín Viola

1. Introducción

El presente informe documenta el desarrollo completo del sistema solicitado por la consigna del trabajo obligatorio “Sistema para Gestión de Reserva de Salas de Estudio” para la UCU. Este proyecto abarcó:

- Diseño y construcción de la base de datos MySQL.
- Implementación de un backend en Python (FastAPI) sin ORM.
- Creación de un frontend en React + Vite + Tailwind.
- Dockerización de la aplicación.
- Implementación de todas las reglas de negocio descritas en la consigna.
- Validaciones y controles de seguridad.

El objetivo fue desarrollar un sistema que permita administrar salas de estudio de manera centralizada, registrando reservas, participantes, asistencia y sanciones, y mantener trazabilidad completa del uso de las salas.

2. Descripción General del Sistema

El sistema implementa:

Gestión completa (ABM) de:

- Participantes
- Salas
- Turnos
- Reservas

- Sanciones

Procesos automáticos:

- Registro de asistencia
- Generación automática de sanciones cuando una reserva finaliza sin asistencia
- Validación de restricciones (horas permitidas, tipos de salas, límite de horas diarias, límite semanal, etc.)

Seguridad:

- Contraseñas hasheadas con bcrypt
- Validación de datos en las distintas capas
- Restricciones en la base de datos
- Dockerización evitando exponer puertos innecesarios

Consultas para reportes (BI):

- Salas más reservadas
- Turnos más usados
- Asistencias
- Reservas por facultad/programa
- Entre otras

Estas funcionalidades abarcan todos los puntos obligatorios definidos en la consigna del curso.

3. Arquitectura General

La solución se estructuró en **tres capas**:

Frontend (React + Vite + Tailwind)

|
v

Backend (FastAPI – Python – REST)

|

3.1. Tecnologías utilizadas

Frontend

- React 18
- Vite
- TailwindCSS
- Lucide Icons
- TypeScript
- Next (para compatibilidad de estructura)
- Vercel Analytics (removable)

Se incluyeron componentes reutilizables, formularios y validaciones complementarias antes de enviar los datos al backend.

Backend

- Python 3.12
- FastAPI
- bcrypt
- mysql-connector-python
- Modularización por routers, models y services
- Validadores propios para todas las reglas de negocio
- Responses estandarizadas (200, 400, 404, 422)

Base de Datos

- MySQL 8.0
- Estructura exacta a la solicitada en la letra:
 - participante
 - login
 - sala
 - reserva

- turno
- sancion_participante
- etc.

Infraestructura

- Docker
- Docker Compose (servicios: backend, base de datos)

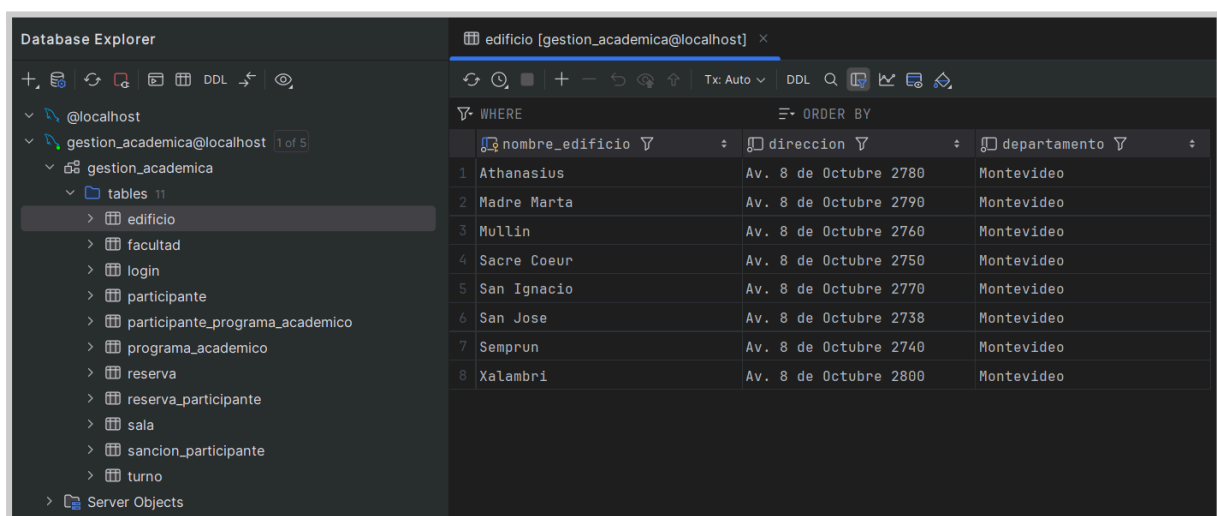
4. Base de Datos (MySQL)

La estructura respeta estrictamente los lineamientos:

- Tablas y claves primarias
- Relaciones con claves foráneas y ON DELETE CASCADE cuando corresponde
- Validaciones con CHECK (por ejemplo: duración del turno = 1 hora exacta)
- Integridad referencial total

Incluye:

- Script completo SQL, que se ejecuta automáticamente al levantar el contenedor, en /database/docker-entry
- Datos iniciales (salas, facultades, programas académicos, turnos, participantes)



The screenshot shows a MySQL Database Explorer interface. On the left, the 'Database Explorer' pane shows a tree view of the database structure. The 'gestion_academica' database is selected, and its tables are listed: edificio, facultad, login, participante, participante_programa_academico, programa_academico, reserva, reserva_participante, sala, sancion_participante, and turno. The 'edificio' table is highlighted. On the right, the 'Query Results' pane shows the data for the 'edificio' table. The table has three columns: 'nombre_edificio', 'direccion', and 'departamento'. The results are as follows:

	nombre_edificio	direccion	departamento
1	Athanasius	Av. 8 de Octubre 2780	Montevideo
2	Madre Marta	Av. 8 de Octubre 2790	Montevideo
3	Mullin	Av. 8 de Octubre 2760	Montevideo
4	Sacre Coeur	Av. 8 de Octubre 2750	Montevideo
5	San Ignacio	Av. 8 de Octubre 2770	Montevideo
6	San Jose	Av. 8 de Octubre 2738	Montevideo
7	Semprun	Av. 8 de Octubre 2740	Montevideo
8	Xalambri	Av. 8 de Octubre 2800	Montevideo

Database Explorer

facultad [gestion_academica@localhost] x

WHERE ORDER BY

	id_facultad	nombre
1		1 Facultad de Ciencias Empresariales
2		2 Facultad de Ingenieria y Tecnologias
3		3 Facultad de Psicologia
4		4 Facultad de Derecho
5		5 Facultad de Ciencias Humanas

Database Explorer

login [gestion_academica@localhost] x

WHERE ORDER BY

	correo	contrasenia
	ana.silva@ucu.edu.uy	\$2b\$12\$R5j0fCDAW004T2rhqF41deLGByr0q5BxynTQ6LHjz0xTUUQJ32Rmu
	ana@example.com	\$2b\$12\$eNDhmR0MPJu4KJLity8wB0hDfr0fWPSuK9desdb3H/JRZ1C75UNnT
	carlos.lopez@ucu.edu.uy	\$2b\$12\$0YxHuFRzKsoF5n4EfvgJHe6inLLycQqk5AhDS44m5C7.LoXjvz7V6
	juan.perez@ucu.edu.uy	\$2b\$12\$EK4h6JUSqxuk8r7IzWzZ3uAhsj003Vv3Qn4wNsrr8u2JkpU4rB24e
	lucia.suarez@ucu.edu.uy	\$2b\$12\$FZsZyW0v5egxL4Qh2bTQFeWcM4YhmxqcaJYt6ZcPggqxJCj0K3lpLK
	maria.garcia@ucu.edu.uy	\$2b\$12\$6Z.5fFqCqyVJ07kIsPu3q06A8MyxSSB7kLcmxE5tV3m6AtYJQ1QM
	matipere@example.com	\$2b\$12\$2TS6p8vRvJtmz6JST3S0P0XqaD0sVBj0IBAieSS2.ExwXZeJRrVBy
	rodrigo.fernandez@ucu.edu.uy	\$2b\$12\$0fnuqWqRzXKQFbdE6K0LN.paVR8qSSVimeoYndZakb.h8u5K5s2lm
9	sofia.mendez@ucu.edu.uy	\$2b\$12\$mSST9jPjFT2hJAiI8fVphu7bZar9Cy0S1l1QWgFvRbkrnf466TqeJe

Database Explorer

participante [gestion_academica@localhost] x

WHERE ORDER BY

	ci	nombre	apellido	email
1	11111111	Ana	López	ana@example.com
2	40298377	Sofia	Mendez	sofia.mendez@ucu.edu.uy
3	43782910	Ana	Silva	ana.silva@ucu.edu.uy
4	45111223	Carlos	Lopez	carlos.lopez@ucu.edu.uy
5	46293847	Lucia	Suarez	lucia.suarez@ucu.edu.uy
6	48923123	Juan	Perez	juan.perez@ucu.edu.uy
7	49011892	Rodrigo	Fernandez	rodrigo.fernandez@ucu.edu.uy
8	52201984	Maria	Garcia	maria.garcia@ucu.edu.uy
9	45134134	Matias	Pérez	matipere@example.com

Database Explorer

participa...academico [gestion_academica@localhost]

WHERE

ORDER BY

	id_alumno_programa	id_participante	nombre_programa	rol
1		1	48923123 Ingenieria en Informatica	alumno
2		2	52201984 Administracion de Empresas	alumno
3		3	43782910 Maestria en Psicologia Organizacional	alumno
4		4	49011892 Ingenieria en Informatica	docente
5		5	40298377 Licenciatura en Educacion	docente
6		6	45111223 Derecho	alumno
7		7	46293847 Contador Publico	alumno

Database Explorer

programa_academico [gestion_academica@localhost]

WHERE

ORDER BY

	nombre_programa	id_facultad	tipo
1	Administracion de Empresas		1 grado
2	Contador Publico		1 grado
3	Derecho		4 grado
4	Diploma en Gestion Educativa		5 posgrado
5	Ingenieria en Informatica		2 grado
6	Licenciatura en Educacion		5 grado
7	Maestria en Psicologia Organizacional		3 posgrado

Database Explorer

reserva [gestion_academica@localhost]

WHERE

ORDER BY

	id_reserva	nombre_sala	edificio	fecha	id_turno	estado
1		1 Laboratorio 1	San Jose	2025-05-10		1 finalizada
2		2 Aula Magna	Semprun	2025-05-11		2 finalizada
3		3 Aula 101	Sacre Coeur	2025-05-12		3 activa
4		4 Sala Docentes 1	Mullin	2025-05-13		4 activa
5		5 Auditorio Central	San Ignacio	2025-05-14		5 activa
6		6 Sala de Reuniones	Athanasius	2025-05-14		2 cancelada
7		7 Aula 101	Sacre Coeur	2025-11-24		1 activa
8		8 Aula 101	Sacre Coeur	2025-11-25		1 activa
9		9 Auditorio Central	San Ignacio	2025-12-01		8 activa

Database Explorer

reserva_participante [gestion_academica@localhost]

ci_participante id_reserva fecha_solicitud_reserva asistencia

1	40298377	4	2025-05-12 10:15:00	1
2	43782910	3	2025-05-11 09:45:00	0
3	46293847	5	2025-05-12 11:00:00	1
4	48923123	1	2025-05-09 10:00:00	1
5	49011892	4	2025-05-12 08:00:00	1
6	52201984	2	2025-05-10 12:30:00	1

Database Explorer

sala [gestion_academica@localhost]

nombre_sala edificio capacidad tipo_sala

1	Auditorio Central	San Ignacio	200 libre
2	Aula 101	Sacre Coeur	40 libre
3	Aula 202	Sacre Coeur	35 libre
4	Aula Magna	Semprun	120 libre
5	Laboratorio 1	San Jose	30 posgrado
6	Laboratorio 2	San Jose	25 posgrado
7	Sala de Reuniones	Athanasius	10 docente
8	Sala Docentes 1	Mullin	15 docente

Database Explorer

sancion_participante [gestion_academica@localhost]

ci_participante fecha_inicio fecha_fin

1	43782910	2025-05-13	2025-05-20
2	46293847	2025-05-15	2025-05-18

	id_turno	hora_inicio	hora_fin
1	1	08:00:00	09:00:00
2	2	09:00:00	10:00:00
3	3	10:00:00	11:00:00
4	4	11:00:00	12:00:00
5	5	12:00:00	13:00:00
6	6	13:00:00	14:00:00
7	7	14:00:00	15:00:00
8	8	15:00:00	16:00:00
9	9	16:00:00	17:00:00
10	10	17:00:00	18:00:00
11	11	18:00:00	19:00:00
12	12	19:00:00	20:00:00
13	13	20:00:00	21:00:00
14	14	21:00:00	22:00:00
15	15	22:00:00	23:00:00

5. Backend (FastAPI)

La API se implementó modularmente con:

- routes/
- models/
- services/
- utils/helpers.py
- utils/validators.py

5.1. Endpoints incluidos

Sistema de Gestión de Salas UCU 1.0.0 OAS 3.1

/openapi.json

Backend del obligatorio de Bases de Datos 1 - UCU

Participantes

GET	/participantes/	Listar Todos	⌵
POST	/participantes/	Crear	⌵
GET	/participantes/{ci}	Obtener	⌵
PATCH	/participantes/{ci}	Actualizar	⌵
DELETE	/participantes/{ci}	Borrar	⌵

Login			^
POST	/login/	Crear Login Endpoint	▼
GET	/login/{correo}	Obtener Login Endpoint	📄 ▼
PATCH	/login/{correo}	Actualizar Login Endpoint	▼
DELETE	/login/{correo}	Eliminar Login Endpoint	▼
POST	/login/authenticate	Autenticar Login Endpoint	▼
Salas			^
GET	/salas/	Obtener Salas	▼
POST	/salas/	Agregar Sala	▼
DELETE	/salas/{nombre_sala}/{edificio}	Eliminar Sala	▼
PUT	/salas/{nombre_sala}/{edificio}	Actualizar Sala	▼
Reservas			^
GET	/reservas/	Obtener Reservas	▼
POST	/reservas/	Crear Nueva Reserva	📄 ▼
GET	/reservas/{id_reserva}	Obtener Reserva	▼
DELETE	/reservas/{id_reserva}	Eliminar Reserva	▼
PATCH	/reservas/{id_reserva}/cancelar	Cancelar Reserva Endpoint	▼
PATCH	/reservas/{id_reserva}/finalizar	Finalizar Reserva Endpoint	▼
POST	/reservas/{id_reserva}/participantes	Agregar Participante	▼
GET	/reservas/{id_reserva}/participantes	Obtener Participantes Reserva	▼
PATCH	/reservas/{id_reserva}/asistencia	Actualizar Asistencia Reserva	▼
Sanciones			^
GET	/sanciones/	Listar Todas	▼
POST	/sanciones/	Crear	▼
GET	/sanciones/participante/{ci}	Listar Por Ci	▼
PATCH	/sanciones/{id_sancion}	Actualizar	▼
DELETE	/sanciones/{id_sancion}	Eliminar	▼

5.2. Reglas de negocio implementadas

Se implementaron restricciones como:

- *Reservas solo en bloques de 1 hora*
- *Máximo 2 horas diarias (para grado)*
- *Máximo 3 reservas activas por semana (para grado)*
- *Salas posgrado disponibles solo para posgrado/docentes*

- Salas docentes exclusivas para docentes
- Capacidad de sala no superada
- Asistencia obligatoria
- Sanción automática de 2 meses si nadie asiste

Estas reglas están implementadas tanto en forma de *triggers* en la DB MySQL, como en *validators* en los servicios del backend Python.

Capturas para insertar aquí

- Swagger mostrando todos los endpoints
- Ejemplos reales de POST /reservas
- Ejemplos reales de PATCH asistencia
- Mensajes de error de validación

6. Frontend (React + Vite + Tailwind)

El frontend implementa:

- Pantalla de login
- Pantalla de reservas actuales

CI	Nombre	Apellido	Email	Tipo	Programa	Acciones
40298377	Sofía	Méndez	sofia.mendez@ucu.edu.uy			✎ Editar ✖ Eliminar
43782910	Ara	Silva	ara.silva@ucu.edu.uy			✎ Editar ✖ Eliminar
45111223	Carlos	López	carlos.lopez@ucu.edu.uy			✎ Editar ✖ Eliminar
46293847	Luca	Suárez	luca.suarez@ucu.edu.uy			✎ Editar ✖ Eliminar
48923123	Juan	Pérez	juan.perez@ucu.edu.uy			✎ Editar ✖ Eliminar
49011892	Rodrigo	Fernández	rodrigo.fernandez@ucu.edu.uy			✎ Editar ✖ Eliminar
52201984	Maria	García	maria.garcia@ucu.edu.uy			✎ Editar ✖ Eliminar

- Formulario de creación de reservas
- Gestión de salas

Nombre	Edificio	Capacidad	Tipo	Ubicación	Acciones
	San Ignacio	200			Editar Eliminar
	Sacre Coeur	40			Editar Eliminar
	Sacre Coeur	35			Editar Eliminar
	Sempron	120			Editar Eliminar
	San Jose	30			Editar Eliminar
	San Jose	25			Editar Eliminar
	Alfonso	10			Editar Eliminar
	Müller	15			Editar Eliminar

- Gestión de participantes
- Vista de sanciones

El objetivo fue entregar una interfaz clara que facilite a los funcionarios de UCU el uso del sistema.

Dependencias principales (package.json)

- Vista inicial
- ABM participantes
- ABM salas
- Reservas
- Sanciones
- Errores de validación del front

7. Validaciones y Seguridad

Seguridad

- Contraseñas hasheadas con bcrypt en el backend
- No se guardan contraseñas en texto plano
- No se usa ORM → consultas SQL controladas manualmente
- Sanitización de inputs
- Validaciones Pydantic en todos los modelos
- Validaciones adicionales en base de datos con CHECK + FK

Validaciones de reglas del negocio

Ejecutadas en utils/validators.py:

- Bloques estrictos de 1 hora
- No superposición de horarios
- Restricción de 2 horas por día
- Restricción de 3 reservas por semana
- Restricciones según tipo de sala
- Capacidad de sala
- Sanciones activas impiden reservar

8. Bitácora del Trabajo

Resumen realista del proceso:

1. Creación del script SQL en MySQL y pruebas en Docker.
2. Diseño inicial del modelo relacional según la letra.
3. Implementación del backend base (routers, services, db connection).
4. Implementación progresiva de reglas de negocio.
5. Creación de endpoints adicionales (asistencia, sanciones, reportes).
6. Construcción del frontend con Vite + React + Tailwind.
7. Instalación de entornos, Docker y estructura del proyecto
8. Conexión front-back y pruebas de integración.
9. Corrección de errores detectados en swagger.
10. Generación del informe, instructivo y capturas de pantalla.

9. Decisiones de Diseño

- Elegimos FastAPI por su velocidad, claridad, tipado y compatibilidad con Pydantic.
- Se evitó el uso de ORM para respetar la letra del curso.
- Se modularizó por servicios para mantener un código limpio y testeable.

- Todas las reglas de negocio se centralizaron en un módulo de validadores para evitar duplicación y mejorar el mantenimiento.
- Se optó por React + Vite por eficiencia y facilidad para construir SPAs modernas.
- Se dockerizó todo el entorno para tener una reproducibilidad total del sistema mucho más fácil.

10. Bibliografía

- Documentación oficial de FastAPI
- Documentación de MySQL 8
- Pydantic – documentación oficial
- Docker & Docker Compose docs
- TailwindCSS docs
- bcrypt documentation
- Consigna oficial del curso (UCU, Base de Datos 1)

11. Conclusión

Se intentó realizar un sistema robusto y consistente, que use las tecnologías más apropiadas y que funcione acorde a lo pedido. Tanto el uso de Swagger, MySQL y el frontend fueron fundamentales, y las múltiples verificaciones en las distintas capas fueron capaces de generar seguridad y un sistema más sólido .

Las capturas a ser agregadas demostrarán:

- Creación correcta de la base
- Uso real de todos los endpoints
- Validaciones funcionando
- Sanciones generadas automáticamente
- Frontend operativo