

# Sistema de Matrículas Universitarias

## Proyecto de Programación Orientada a Objetos

Isidora Santa Cruz   David Reyes   Matías Silva   Josefina Cabeza   Matías Follert

Programación Orientada a Objetos

27 de noviembre de 2025

# Índice

- 1 Descripción de la problemática
- 2 Objetivos del proyecto
- 3 Justificación
- 4 Diseño y diagrama de clases
- 5 Listado de funcionalidades
- 6 Conjunto Tecnológico

# Descripción de la problemática

El sistema actual enfrenta desafíos que afectan la eficiencia:

## Procesos manuales y errores

- Dependencia de formularios físicos y hojas de cálculo sin conexión.
- Alta probabilidad de error humano.
- Pérdida de documentos y conflictos en la administración.

## Gestión de cupos y espacios

- Falta de control en tiempo real de la capacidad de aulas.
- Demasiados cupos o utilización precaria de recursos.

# Descripción de la problemática

## Conflictos de horarios

- Inscripciones accidentales con topes de horario.
- Detección tardía de conflictos.

## Descentralización y Comunicación

- Información fragmentada entre estudiantes y profesores.
- Falta en la comunicación: Listas desactualizadas para profesores y falta de información del docente para alumnos.

# Objetivo general

## Objetivo principal

Desarrollar un sistema integral de gestión académica en Python que unifique y optimice el proceso de matrícula, asignación docente y administración para las operaciones del instituto.

# Objetivos específicos

## ❶ Gestión de estudiantes:

- Registro digital, historial académico y actualización de datos.
- Búsqueda y visualización de carga académica.

## ❷ Administración centralizada:

- Panel de control con gestión de roles y permisos.
- Generación de reportes, estadísticas y respaldo de seguridad.

## ❸ Inscripción y asignación:

- Validación automática de pre-requisitos y topes de horario.
- Gestión de cupos.

# Justificación del proyecto

## **Eficiencia**

- Reducción del tiempo de matrícula.
- Automatización de tareas repetitivas.

## **Confiabilidad de datos**

- Eliminación de discrepancias entre departamentos.
- Minimización de errores.

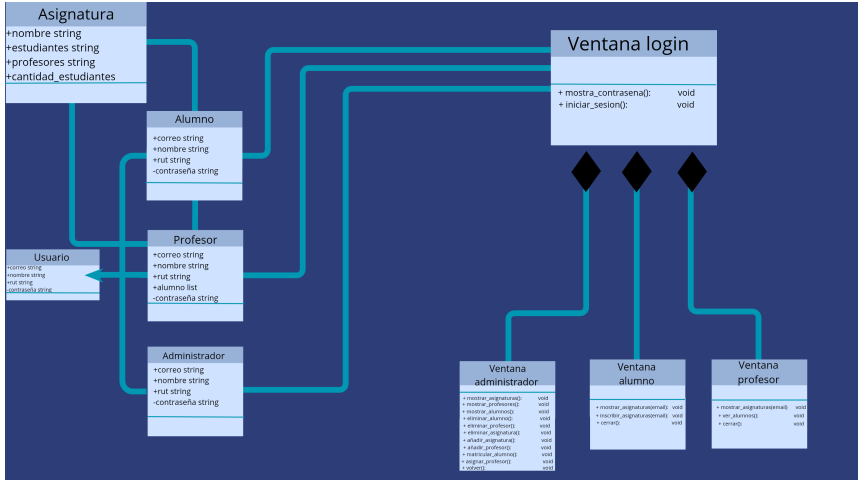
## **Experiencia educativa**

- Procesos gestionables para alumnos.
- Herramientas de gestión efectivas para los docentes.

## **Toma de decisiones**

- Análisis e informes para optimizar los recursos.
- Gestión basada en los datos.

# Diagrama de Clase





El sistema utiliza herencia para gestionar los roles.

## Clase Padre usuario

- Manejo de autenticación.
- Gestión centralizada de credenciales.
- Definición de roles y permisos del sistema.

# Roles Específicos

## Clase estudiante

- Historial académico.
- Asignaturas inscritas.
- Autogestión de matrícula.
- Progreso de carrera.

## Clase profesor

- Información profesional.
- Asignaturas impartidas.
- Seguimiento alumnos.

## Clase administrador

- Privilegios totales.
- Reportes consolidados.
- Integridad de datos.

## Clase asignatura

Representa el catálogo, controla la disponibilidad, los prerrequisitos y las relaciones entre estudiantes y profesores.

# Funcionalidad por perfil

## Funcionalidades generales

- Autenticación con correo institucional.
- Dirección automática según rol de usuario.

## Estudiante

- **Visualización de carga:** Ver horario y cursos actuales.
- **Inscripción:** Selección de materias con validación de cupos.
- **Historial:** Acceso a notas y cursos pasados.

# Funcionalidad por perfil

## Profesor

- **Gestión de cursos:** Ver listado de asignaturas asignadas.
- **Listas de clase:** Visualizar estudiantes inscritos en tiempo real.

## Administrador

- **Usuarios:** Crear, leer, actualizar y borrar usuarios.
- **Gestión académica:** Abrir/cerrar cursos y secciones.
- **Control total:** Resolver conflictos de inscripción manualmente.

# Herramientas y tecnologías

El proyecto fué construido con lo siguiente:

- **Lenguaje:** Python 3.13.7 .
- **Interfaz gráfica:** CustomTkinter
- **Gestión:** Entornos virtuales.