

UNIVERSIDAD AUTÓNOMA GABRIEL RENÉ MORENO



SOFTWARE WEB Y MÓVIL PARA LA GESTIÓN DE UN AULA VIRTUAL

GRUPO DE EXAMEN : 9

INTEGRANTES :

Cuéllar Flores Victor Hugo	222008180
Méndez López Sebastián	222055987

MATERIA : Sistemas de la Información II

GRUPO DE LA MATERIA : SA

DOCENTE : MSc. Ing. Angélica Garzón Cuéllar

GESTIÓN : 1/2025

Índice

1. Fundamentos teóricos	7
1.1 Introducción	7
1.2 Objetivo General.....	8
1.3 Objetivos Específicos	8
1.4 Alcance	8
1.4.1 Módulo de Gestión de Usuarios y Roles	8
1.4.2 Gestión de Cursos y Materias	9
1.4.3 Registro Académico y de Comportamiento	9
1.4.4 Motor de Predicción del Rendimiento Académico	9
• Predicciones numéricas	9
• Clasificación categórica:.....	9
1.4.5 Dashboard Interactivo y Visualización.....	9
1.4.6 Módulo de Administración y Configuración	10
1.4.7 Experiencia de Usuario y Diseño Adaptativo.....	10
1.4.8 Integración Tecnológica Full Stack	10
• Backend:	10
• Frontend:	10
• Base de datos:	10
• Mobile:.....	10
1.4.9 Gestión de Proyecto con SCRUM.....	10
1.4.10 Despliegue, Documentación y Repositorio	11
2. Marco teórico referencial y metodológico	11
2.1 Marco Referencial.....	11
2.1.1 Fundamentos de la Inteligencia Artificial en Educación	11
2.1.2 Conceptos Educativos Clave	11
• Rendimiento académico:	11
• Asistencia:	11
• Participación activa:	12
• Evaluación formativa:	12
• Intervención temprana:.....	12
2.1.3 Generalidades y Características del Sistema de Aula Inteligente	12
2.1.4 Conceptos Específicos del Proyecto Aula Inteligente	12

• Predicción del rendimiento académico:	12
• Variables predictoras:	12
• Dashboard académico:	12
• Machine Learning Supervisado:	12
• Desarrollo full stack:	13
• SCRUM:	13
2.2 Marco de Trabajo Ágil SCRUM.....	13
2.2.1 Definición de SCRUM	13
2.2.2 Roles en SCRUM.....	13
2.2.3 Eventos y Artefactos de SCRUM	13
2.2.4 Aplicación de SCRUM en el Proyecto	14
3. Herramientas tecnológicas para el desarrollo	15
3.1 Lenguaje de Programación.....	15
3.1.1 Desarrollo Web	15
3.1.2 Desarrollo Móvil	15
3.1.3 Desarrollo Backend.....	15
3.2 Frameworks y Entornos de Ejecución	15
3.2.1 Frameworks Web	15
3.2.2 Frameworks Móviles	15
3.2.3 Framework Backend	15
3.2.4 Entornos de Ejecución	16
3.3 Sistema de Gestión de Base de Datos	16
3.4 Lenguaje de Modelado de Software (Modelo C4).....	16
3.5 Herramienta de Diseño y Modelado	16
3.6 Entorno de Desarrollo.....	17
3.7 Infraestructura de Software (IaaS).....	17
3.8 Servicio en la Nube (Cloud Computing).....	17
3.9 SaaS (Software as a Service).....	17
3.10 Herramientas Colaborativas para Seguimiento de Proyectos	18
3.11 Sistema de Control de Versiones de Código.....	18
3.12 Herramientas de Gestión de Código en la Nube	18
4. Requerimientos	18
4.1 Propósito	18

4.2 Ámbito del Sistema	19
4.3 Equipo SCRUM	19
4.4 Definiciones, Acrónimos y Abreviaturas	20
4.5 Funciones del Producto	21
4.6 PRODUCT BACKLOG.....	22
4.7 Requisitos Funcionales.....	24
4.8 Requisitos No Funcionales	25
4.9 Lista de Casos de Uso (Web y Móvil).....	26
4.10 Paquetes y casos de Uso	28
4.10.1 Paquete de Gestión de usuarios	28
4.10.2 Paquete de Gestión Académica	28
4.10.3 Paquete de Analíticas Académicas	29
4.10.4 Paquete de Visualización y Reportes.....	30
4.10.5 Paquete de Autenticación y Seguridad	31
4.11 Planificación Sprint (Diagrama de Gantt).....	31
4.11.1 Sprint 1 13/05 – 20/05	31
4.11.2 Sprint 2 21/05 – 27/05	32
4.11.3 Sprint 3 28/05 – 03/06	33
5. PROCESO DE DESARROLLO DE SOFTWARE	34
Sprint 1	34
1. Sprint Planning	34
1.1. Objetivos del Sprint	34
1.2. Historias de usuario (tarjetas 3C, Planning Póker, prototipos).....	34
1.3. Contexto del sistema	38
1.4. Sprint backlog – Sprint 1.....	39
1.5. Equipo SCRUM	40
2. Proceso/patrón de desarrollo por Historia de Usuario.....	41
2.1 Diseño	41
2.1.1 Diseñar de la arquitectura (Modelo C4)	41
2.1.1.1 Diagrama de contexto	41
2.1.1.2 Diagrama de contenedores	41
2.1.1.3 Diagrama de componentes	42
2.1.1.4 Diagrama de código	42

2.1.2 Diseño de datos	43
a) Diagrama de clase.....	43
b) Script	43
c) Mapeo	46
2.1.3 Diseño de la lógica de negocio	47
2.1.3.1 Diagramas de secuencia	48
2.1.3.1.1 CU01: Registro de usuarios	48
2.1.3.1.2 CU03: Edición de información de usuario.....	48
2.1.3.1.3 CU04: Cambio de contraseña	49
2.1.3.2 Diagramas de estado	49
2.1.3.2.1 CU02: Gestión de roles y permisos	49
2.1.3.3 Diagrama de tiempo.....	50
2.1.3.3.1 CU05: Consulta de ficha individual del estudiante	50
2.1.3.4 Diagrama de navegación	50
2.1.3.4.1 Pagina principal.....	50
2.1.3.4.2 Por subsistemas.....	51
2.2 Implementación	53
2.2.1. Componentes y artefactos generados	53
2.3 Pruebas	55
2.3.1. Plan de pruebas (criterios de aceptación).....	55
2.3.2 Reporte de prueba	57
3. Daily Scrum (o Scrum diario)	61
Daily Scrum - Sprint 1.....	61
Daily Scrum 13/05/2025.....	61
Daily Scrum 14/05/2025.....	61
Daily Scrum 15/05/2025.....	62
Daily Scrum 16/05/2025.....	62
Daily Scrum 17/05/2025.....	62
Daily Scrum 18/05/2025.....	62
4. Sprint Review	63
5. Sprint Retrospective	64
6. Burndown y BurnUp(Grafica de tareas y Datos de tareas)	65
7. Grafica de esfuerzo y Datos de esfuerzo	65

7.1. Gráfico de esfuerzo	65
7.2. Gráfico de datos de esfuerzo.....	66
8. Scrum TaskBoard (Backlog, to do, doing, done)	67
LINKS & QR – Sprint 1	67

1. Fundamentos teóricos

1.1 Introducción

En la actualidad, los sistemas educativos enfrentan múltiples desafíos relacionados con la gestión eficiente de la información académica, la identificación oportuna de estudiantes con bajo rendimiento y la necesidad de adaptar los procesos pedagógicos a las capacidades individuales de los alumnos. A medida que las tecnologías de la información evolucionan, se abre un camino prometedor hacia la automatización inteligente de estas tareas, ofreciendo a instituciones educativas herramientas más potentes para la toma de decisiones fundamentadas.

El presente proyecto, titulado *Aula Inteligente*, propone el desarrollo de una aplicación web y móvil centrada en la gestión académica y la predicción del rendimiento estudiantil utilizando técnicas de Inteligencia Artificial (IA). Esta solución busca integrarse de forma natural al ecosistema educativo, permitiendo a docentes, administradores y alumnos interactuar con el sistema de forma eficiente, segura y accesible. A través de la recolección y procesamiento de datos como notas, asistencia y participación, el sistema será capaz de proyectar el rendimiento académico futuro de cada estudiante mediante modelos de aprendizaje automático supervisado, brindando alertas y sugerencias preventivas ante posibles escenarios de riesgo académico.

La propuesta contempla la implementación de una arquitectura full stack moderna que abarque tanto el backend como el frontend, el almacenamiento estructurado de datos en una base de datos relacional, y la integración de un modelo predictivo basado en algoritmos como Random Forest o Regresión Lineal. Además, se diseñará un dashboard visual e interactivo que facilite la interpretación de datos e impulse una experiencia de usuario agradable y funcional, adaptada a distintos tipos de dispositivos.

El sistema estará orientado a tres tipos de usuarios principales: administradores, docentes y alumnos, cada uno con funcionalidades específicas dentro del entorno. El rol de administrador podrá gestionar usuarios, cursos, materias y configuración general del sistema; los docentes podrán registrar calificaciones, controlar la asistencia, monitorear la participación y acceder a predicciones personalizadas; mientras que los alumnos podrán consultar su historial académico y visualizar sus proyecciones de rendimiento.

El desarrollo de *Aula Inteligente* se realizará aplicando la metodología ágil **SCRUM**, dividiendo el proyecto en tres sprints claramente definidos, con entregables incrementales, pruebas funcionales, documentación técnica y seguimiento mediante herramientas de gestión como Trello o Jira. También se hará uso de herramientas de modelado modernas como **el Modelo C4** para representar la arquitectura del sistema y **UML 2.5+** para documentar los procesos, clases y casos de uso.

Con esta solución, se busca no solo automatizar tareas académicas tradicionales, sino también innovar en la forma en que se analiza y utiliza la información estudiantil, fortaleciendo el vínculo entre tecnología e inteligencia pedagógica. La implementación de *Aula Inteligente* representa un paso concreto hacia una educación más proactiva, adaptativa y centrada en el estudiante.

1.2 Objetivo General

Desarrollar una aplicación web/móvil denominada *Aula Inteligente*, que permita registrar y gestionar información académica de los estudiantes, e integre un modelo de inteligencia artificial supervisado capaz de predecir el rendimiento académico futuro, brindando herramientas visuales útiles tanto para el alumno como para el profesor.

1.3 Objetivos Específicos

- Diseñar e implementar un sistema de gestión académica que registre datos de alumnos, notas, asistencia y participación.
- Entrenar y aplicar un modelo de machine learning supervisado (como Random Forest o Regresión Lineal) para predecir el rendimiento académico.
- Visualizar de forma clara el desempeño histórico y proyectado de cada estudiante mediante dashboards e interfaces gráficas.
- Aplicar metodologías ágiles (SCRUM) y herramientas de modelado (Modelo C4 y UML 2.5+) durante el proceso de desarrollo.
- Documentar el sistema y el modelo de IA, garantizando su comprensión, reproducibilidad y mantenimiento.

1.4 Alcance

1.4.1 Módulo de Gestión de Usuarios y Roles

Se implementará un sistema de autenticación y autorización para gestionar distintos tipos de usuarios: administradores, docentes y alumnos. Cada perfil tendrá permisos específicos en función de sus responsabilidades dentro del sistema.

- **Administrador:** acceso total al sistema, configuración inicial, gestión de usuarios, materias y cursos.
- **Docente:** registro y consulta de notas, asistencias, participaciones, acceso a predicciones de sus alumnos.

- **Alumno:** consulta de su historial académico y de predicciones sobre su rendimiento.
El sistema garantizará una experiencia segura y diferenciada para cada tipo de usuario, siguiendo buenas prácticas de gestión de identidad.

1.4.2 Gestión de Cursos y Materias

Se desarrollará un módulo que permita a los administradores definir cursos y asignaturas asociadas, con la posibilidad de establecer ciclos, niveles, paralelos y asignaciones docentes. Cada alumno podrá estar vinculado a uno o varios cursos y materias, lo que permitirá organizar eficientemente la carga académica y el seguimiento individualizado.

1.4.3 Registro Académico y de Comportamiento

El sistema contará con formularios intuitivos para el registro periódico de:

- **Notas académicas**, ingresadas por materia y por periodo
- **Asistencia**, almacenando porcentajes de presencia por curso
- **Participación en clase**, medida por frecuencia de intervenciones u otras métricas cualitativas/ponderadas
Toda esta información se almacenará en una base de datos estructurada (PostgreSQL), asegurando trazabilidad y calidad de datos para alimentar el modelo de predicción.

1.4.4 Motor de Predicción del Rendimiento Académico

Se integrará un modelo de machine learning supervisado, con opciones como Random Forest o Regresión Lineal, capaz de procesar las variables registradas (notas, asistencia, participación) para predecir el rendimiento académico futuro de cada estudiante.

El modelo ofrecerá:

- *Predicciones numéricas* (ejemplo: 62.7)
- *Clasificación categórica*: bajo / medio / alto rendimiento
Estas predicciones se mostrarán en tiempo real y podrán ser actualizadas conforme se registren nuevos datos, ayudando a detectar a tiempo posibles riesgos académicos.

1.4.5 Dashboard Interactivo y Visualización

Se desarrollará una interfaz gráfica moderna y funcional que incluya:

- Total de alumnos registrados
- Promedio general de calificaciones por curso y materia
- Comparación entre rendimiento real vs rendimiento predicho (gráficos)

- Ficha individual por estudiante, con historial académico, evolución y predicción
El dashboard ofrecerá una visualización clara para apoyar a docentes y alumnos en la toma de decisiones, implementando gráficos interactivos y visualizaciones sencillas adaptadas a distintos dispositivos.

1.4.6 Módulo de Administración y Configuración

Se incluirá un panel de administración desde el cual el usuario con rol de administrador podrá:

- Crear usuarios y asignar roles
- Gestionar cursos, paralelos y asignaturas
- Activar/desactivar funciones del sistema
- Configurar parámetros del modelo predictivo
Este módulo centralizará la configuración del sistema y facilitará su mantenimiento y escalabilidad.

1.4.7 Experiencia de Usuario y Diseño Adaptativo

El sistema ofrecerá una experiencia de usuario fluida, amigable y responsive. El diseño se adaptará a distintos dispositivos (PC, tablet, móvil) para garantizar accesibilidad. Se implementarán formularios validados, navegación intuitiva y tiempos de respuesta optimizados, con énfasis en la facilidad de uso para docentes y estudiantes sin conocimientos técnicos.

1.4.8 Integración Tecnológica Full Stack

Se adoptará una arquitectura moderna y escalable utilizando las siguientes tecnologías:

- *Backend*: Python con Flask o Django
- *Frontend*: Angular o React
- *Base de datos*: PostgreSQL
- *Mobile*: Flutter
Estas tecnologías estarán conectadas mediante APIs RESTful. Se asegurará la integración entre todos los módulos, así como la posibilidad futura de extender el sistema (por ejemplo, a módulos de pagos, mensajería, o notificaciones).

1.4.9 Gestión de Proyecto con SCRUM

Todo el proceso de desarrollo se estructurará en tres sprints utilizando el marco de trabajo ágil SCRUM. Se definirá un Sprint Planning, se gestionarán tareas mediante tableros (Trello o Jira), y se generarán entregables incrementales. Se realizarán

Daily Scrums, Sprint Reviews y Retrospectivas, y se documentarán los avances mediante gráficos Burndown/Burnup y tableros de tareas (to do, doing, done).

1.4.10 Despliegue, Documentación y Repositorio

El sistema será desplegado en un entorno de producción en la nube (Google Cloud, AWS o Azure). Además, se entregará:

- Repositorio en GitHub con código organizado y comentado
- Documentación técnica del sistema y del modelo de IA
- Dataset utilizado (real o simulado)
- Código QR con acceso al sistema o al repositorio
- Manual de usuario básico

2. Marco teórico referencial y metodológico

2.1 Marco Referencial

2.1.1 Fundamentos de la Inteligencia Artificial en Educación

La Inteligencia Artificial (IA) ha transformado numerosos sectores, y la educación no es la excepción. En el ámbito educativo, la IA permite desarrollar sistemas capaces de analizar información académica y comportamental para ofrecer diagnósticos, alertas tempranas y predicciones sobre el rendimiento estudiantil. Estas tecnologías favorecen una enseñanza personalizada, la automatización de tareas repetitivas y la mejora en la toma de decisiones pedagógicas, tanto a nivel individual como institucional.

Particularmente, los modelos de aprendizaje automático supervisado como **Random Forest** y **Regresión Lineal** permiten predecir comportamientos futuros con base en variables históricas. En educación, esto se traduce en la posibilidad de anticipar el desempeño académico de un estudiante analizando datos como calificaciones previas, asistencia y participación.

2.1.2 Conceptos Educativos Clave

El sistema Aula Inteligente se construye sobre pilares fundamentales de la educación contemporánea, tales como:

- **Rendimiento académico:** Medida del logro de objetivos de aprendizaje por parte de un estudiante, generalmente expresado mediante calificaciones o promedios.
- **Asistencia:** Presencia física o virtual del estudiante en el aula. Está estrechamente vinculada al compromiso y al rendimiento.

- *Participación activa*: Interacción del estudiante durante el proceso de enseñanza, mediante preguntas, respuestas, tareas u otras actividades. Es un indicador clave de involucramiento.
- *Evaluación formativa*: Proceso continuo de recolección de información sobre el aprendizaje, que permite hacer ajustes pedagógicos oportunos.
- *Intervención temprana*: Estrategia pedagógica que busca detectar y actuar sobre estudiantes en riesgo antes de que se produzcan fracasos académicos.

Estos conceptos orientan el diseño funcional del sistema y determinan los indicadores a utilizar para la predicción de resultados.

2.1.3 Generalidades y Características del Sistema de Aula Inteligente

El sistema Aula Inteligente se concibe como una **plataforma digital multiplataforma (web/móvil)** que integra funciones administrativas, académicas y analíticas. Entre sus características principales se destacan:

- Registro de alumnos, cursos y materias por niveles educativos.
- Registro periódico de calificaciones, asistencia y participación en clase.
- Uso de un modelo predictivo de IA que genera una proyección de rendimiento futuro por estudiante.
- Visualización de datos mediante dashboards interactivos, tanto a nivel general como individual.
- Gestión de usuarios por roles: administrador, docente y estudiante.
- Experiencia de usuario intuitiva, con diseño responsive y adaptable a distintos dispositivos.
- Arquitectura modular, escalable e integrable con futuras funcionalidades como mensajería, pagos o gamificación.

2.1.4 Conceptos Específicos del Proyecto Aula Inteligente

- *Predicción del rendimiento académico*: Proceso de estimar el desempeño futuro de un alumno mediante modelos de IA entrenados con datos históricos.
- *Variables predictoras*: Indicadores cuantificables como notas, asistencia y participación, que alimentan el modelo de machine learning.
- *Dashboard académico*: Panel visual con métricas clave que permite a docentes y estudiantes interpretar fácilmente la información académica y las proyecciones.
- *Machine Learning Supervisado*: Rama de la IA que entrena algoritmos con datos etiquetados (inputs con resultados conocidos) para generar predicciones sobre nuevos datos.

- *Desarrollo full stack*: Implementación del sistema utilizando tecnologías para frontend, backend, base de datos y despliegue en la nube.
- *SCRUM*: Marco ágil de desarrollo por sprints que permite entregas incrementales y mejora continua del producto.

2.2 Marco de Trabajo Ágil SCRUM

2.2.1 Definición de SCRUM

SCRUM es un marco de trabajo ágil que se utiliza para gestionar y ejecutar proyectos complejos, en especial aquellos relacionados con el desarrollo de software. Se caracteriza por su enfoque iterativo e incremental, permitiendo que el trabajo se divida en ciclos cortos llamados Sprints. Esta metodología promueve la revisión continua del progreso, la adaptación a cambios y la entrega de incrementos funcionales del producto en cada ciclo, lo que facilita la colaboración, la transparencia y la mejora constante durante el desarrollo del proyecto.

2.2.2 Roles en SCRUM

En SCRUM se definen tres roles fundamentales que aseguran la correcta ejecución y coordinación del proyecto:

- **Product Owner**: Responsable de definir y priorizar las funcionalidades del producto en el Product Backlog. Representa la voz del cliente y se asegura de que el equipo trabaje en tareas que aporten valor real.
- **Scrum Master**: Facilita la aplicación de SCRUM, eliminando impedimentos y guiando al equipo para que se autoorganice y mejore continuamente. No dirige el equipo, sino que actúa como un coach para garantizar que se sigan las prácticas ágiles.
- **Development Team**: Conformado por profesionales multidisciplinarios encargados de desarrollar el producto. Este equipo es autónomo y tiene la capacidad de completar todas las tareas necesarias para entregar un incremento funcional al final de cada Sprint.

2.2.3 Eventos y Artefactos de SCRUM

SCRUM organiza el trabajo mediante eventos y artefactos que permiten una gestión eficiente del proyecto:

- **Sprint Planning**: Reunión inicial de cada Sprint en la que se planifican las tareas a realizar y se define el objetivo del Sprint (Sprint Goal).
- **Daily Standup (Daily Scrum)**: Encuentro diario de corta duración (15 minutos) donde el equipo sincroniza actividades, identifica impedimentos y ajusta el plan de trabajo para el día.

- **Sprint Review:** Al final del Sprint, el equipo presenta el incremento de producto completado, recibiendo feedback del Product Owner y stakeholders para validar avances y ajustar futuras prioridades.
- **Sprint Retrospective:** Sesión de reflexión post-Sprint en la que se analizan los procesos, se identifican áreas de mejora y se proponen acciones para optimizar la colaboración y eficiencia en los próximos ciclos.

Los artefactos clave incluyen:

- **Product Backlog:** Lista dinámica y priorizada de requerimientos y funcionalidades que deben ser desarrolladas.
- **Sprint Backlog:** Conjunto de tareas seleccionadas del Product Backlog que se comprometen a completar en el Sprint.
- **Incremento:** Resultado tangible y funcional del trabajo realizado durante el Sprint, que debe cumplir con la definición de "Backlog" y estar listo para ser entregado.

2.2.4 Aplicación de SCRUM en el Proyecto

En el contexto del proyecto "Sistema web y móvil Aula Inteligente para la gestión académica y administración de usuarios," SCRUM se aplicará para gestionar el desarrollo de manera eficiente y adaptable.

- **Estructuración de Sprints:** El proyecto se dividirá en Sprints de duración determinada, en los que se planificarán y ejecutarán funcionalidades específicas, tales como el registro y edición de usuarios, gestión de roles y permisos, cambio de contraseñas y consulta de fichas estudiantiles.
- **Gestión del Product Backlog:** Se elaborará y mantendrá un Product Backlog que incluya todas las funcionalidades y mejoras requeridas, priorizadas según el valor que aporten al entorno educativo y a los diferentes perfiles de usuarios (administradores, docentes y estudiantes).
- **Roles Claramente Definidos:** El equipo asignará roles específicos para el Product Owner, el Scrum Master y los miembros del Development Team, asegurando que cada integrante se enfoque en sus responsabilidades para alcanzar los objetivos del Sprint.
- **Eventos SCRUM Regulares:** Se realizarán reuniones diarias (Daily Scrum) para mantener la comunicación fluida, reuniones de planificación al inicio de cada Sprint y sesiones de revisión y retrospectiva al final de cada ciclo, facilitando la identificación de mejoras y la rápida adaptación a cambios en los requerimientos educativos.
- **Entrega de Incrementos Funcionales:** Cada Sprint culminará con la entrega de un incremento funcional del producto, lo que permitirá recibir retroalimentación temprana de los usuarios educativos y ajustar el desarrollo según las necesidades específicas de la institución educativa.

3. Herramientas tecnológicas para el desarrollo

3.1 Lenguaje de Programación

Para el desarrollo del software web y móvil se emplearán los siguientes lenguajes de programación:

3.1.1 Desarrollo Web

El frontend web estará desarrollado utilizando TypeScript, un superconjunto de JavaScript que mejora la seguridad y escalabilidad del código mediante tipado estático y herramientas avanzadas de desarrollo. TypeScript se utilizará dentro del framework Angular, el cual proporciona una arquitectura basada en componentes y un sistema modular altamente eficiente.

3.1.2 Desarrollo Móvil

Para la aplicación móvil, se utilizará Dart, un lenguaje optimizado para interfaces de usuario rápidas y eficientes. Dart se empleará con Flutter, un framework que permite el desarrollo de aplicaciones móviles nativas para iOS y Android con un solo código base, garantizando un rendimiento óptimo y reduciendo costos de desarrollo.

3.1.3 Desarrollo Backend

El backend del sistema se construirá utilizando Python, debido a su versatilidad y amplio ecosistema de librerías. Se empleará el framework Django, el cual ofrece alto rendimiento.

3.2 Frameworks y Entornos de Ejecución

3.2.1 Frameworks Web

- **Angular:** Un framework de desarrollo frontend basado en TypeScript que permite construir aplicaciones escalables y modulares con una interfaz de usuario interactiva y altamente optimizada.
- **Tailwind CSS:** Para el diseño de la interfaz web, se utilizarán estas tecnologías que proporcionan componentes predefinidos y estilos personalizables.

3.2.2 Frameworks Móviles

- **Flutter:** Framework basado en Dart que permite el desarrollo de aplicaciones móviles nativas con una única base de código, optimizando tiempos de desarrollo y mantenimiento.

3.2.3 Framework Backend

- **Django:** Es un framework de desarrollo web de alto nivel basado en Python que permite construir aplicaciones robustas, seguras y escalables de forma

rápida. Django sigue el patrón arquitectónico MTV (Model-Template-View), e incluye un sistema de administración automática, herramientas integradas para manejo de formularios, autenticación, ORM para bases de datos relacionales como PostgreSQL, y seguridad incorporada contra ataques comunes como CSRF, XSS y SQL Injection. Gracias a su enfoque en la reutilización de componentes y el principio de "no te repitas" (*Don't Repeat Yourself*), Django permite desarrollar APIs REST a través de integraciones como **Django REST Framework**, facilitando la conexión entre el backend y el frontend de manera eficiente.

3.2.4 Entornos de Ejecución

- **Node.js:** Se utilizará para la ejecución de procesos en el frontend de Angular y la gestión del entorno de desarrollo.
- **Python 3.11:** La versión más reciente de Python será el entorno de ejecución del backend con Django.

3.3 Sistema de Gestión de Base de Datos

Se utilizará PostgreSQL como sistema de gestión de bases de datos (DBMS) debido a su estabilidad, rendimiento y capacidad para manejar grandes volúmenes de datos. PostgreSQL es una base de datos relacional que ofrece soporte para transacciones ACID, consultas SQL optimizadas y una estructura escalable. Además, para mejorar el rendimiento y la disponibilidad del sistema, se considerará la implementación de caché con Redis, lo que optimizará la velocidad de respuesta en consultas frecuentes.

3.4 Lenguaje de Modelado de Software (Modelo C4)

Para la representación de la arquitectura del sistema, se utilizará el Modelo C4, el cual facilita la visualización de la estructura del software en diferentes niveles de abstracción. Se implementarán los siguientes diagramas:

- **Diagrama de Contexto:** Representa el sistema en su totalidad y sus interacciones con usuarios, proveedores y otros sistemas externos.
- **Diagrama de Contenedores:** Muestra la arquitectura de alto nivel, incluyendo la aplicación web (Angular), el backend (FastAPI), la base de datos (PostgreSQL) y el almacenamiento en la nube (AWS S3).
- **Diagrama de Componentes:** Desglosa los módulos principales, como la gestión de catálogo, pagos, envíos y feedback de clientes.
- **Diagrama de Código:** Representa la estructura del código fuente, con detalles sobre las clases, controladores y servicios.

3.5 Herramienta de Diseño y Modelado

Para la diagramación y modelado del sistema se utilizará Draw.io, una herramienta profesional para la creación de diagramas UML y modelos C4. Draw.io facilitará la

documentación de la arquitectura, flujos de trabajo y relaciones entre los componentes del sistema.

3.6 Entorno de Desarrollo

El desarrollo del software se llevará a cabo en Visual Studio Code (VS Code), un editor de código ampliamente utilizado que ofrece:

- Extensiones para Angular, Django y PostgreSQL.
- Integración con GitHub para control de versiones.
- Debugging y herramientas de linting para optimización del código.

3.7 Infraestructura de Software (IaaS)

El proyecto se desplegará utilizando Infraestructura como Servicio (IaaS) a través de Amazon Web Services (AWS). Los servicios utilizados serán:

- Amazon EC2: Para el hosting de los servicios backend en instancias escalables.
- Amazon S3: Para el almacenamiento de imágenes, modelos 3D y archivos estáticos.
- AWS RDS (PostgreSQL): Para gestionar la base de datos con alta disponibilidad.
- AWS CloudFront: Para la distribución de contenido y mejora del rendimiento del frontend.

3.8 Servicio en la Nube (Cloud Computing)

El sistema operará sobre una infraestructura en la nube para garantizar escalabilidad y disponibilidad. Se utilizarán:

- **Amazon Web Services (AWS)**: Como plataforma principal de computación en la nube.
- **Amazon S3**: Para la optimización y entrega de imágenes en la aplicación web y móvil.

3.9 SaaS (Software as a Service)

El modelo SaaS (Software as a Service) permitirá que los usuarios accedan al sistema sin necesidad de instalar software adicional.

Una plataforma de alojamiento para el control de versiones utilizando Git. GitHub facilita la colaboración en proyectos de software mediante la administración del código fuente, el seguimiento de cambios y la integración continua. Es ampliamente utilizado por desarrolladores para gestionar y compartir código.

3.10 Herramientas Colaborativas para Seguimiento de Proyectos

Para la gestión y planificación del desarrollo del proyecto, se utilizarán herramientas colaborativas:

- Jira Software: Para la gestión ágil del desarrollo y la planificación de sprints SCRUM. Al igual que para la organización de tareas y monitoreo de avances.
- Discord: Para la comunicación en equipo y notificaciones sobre cambios en el código.
- Draw.io: Para la documentación del proyecto y almacenamiento de diagramas.

3.11 Sistema de Control de Versiones de Código

Se empleará Git como sistema de control de versiones, lo que permitirá:

- Control y seguimiento de cambios en el código.
- Creación de ramas para desarrollo paralelo.
- Implementación de revisiones de código y gestión de conflictos.

3.12 Herramientas de Gestión de Código en la Nube

Para alojar y gestionar el código fuente del sistema, se utilizará GitHub, que proporciona:

- Repositorios privados y seguros para almacenar el código.
- CI/CD (Integración y Despliegue Continuo) con pipelines automatizados.

4. Requerimientos

4.1 Propósito

El presente documento de requerimientos tiene como finalidad definir, estructurar y documentar las funcionalidades, características técnicas y condiciones operativas del sistema web y móvil *Aula Inteligente*. Este sistema busca mejorar la gestión académica en instituciones educativas mediante la digitalización de registros escolares y la aplicación de modelos de inteligencia artificial para predecir el rendimiento académico de los estudiantes.

El propósito principal es ofrecer una solución que centralice la información estudiantil, facilite su análisis y permita tomar decisiones preventivas ante posibles bajos desempeños. Además, se busca ofrecer una experiencia fluida tanto para docentes como para estudiantes, con una interfaz clara, segura y accesible desde múltiples dispositivos.

4.2 Ámbito del Sistema

El sistema *Aula Inteligente* se enfocará en el entorno escolar de nivel secundario y superior, permitiendo registrar y analizar información relevante de alumnos, materias, calificaciones, asistencia y participación. Su principal función diferenciadora es la implementación de un motor de predicción basado en técnicas de machine learning supervisado (Random Forest o Regresión Lineal), que generará proyecciones sobre el desempeño académico de cada estudiante.

Este sistema incluye dos plataformas:

- Una **plataforma web** dirigida principalmente a docentes y administradores, para gestión académica, configuración del sistema, y visualización de reportes predictivos.
- Una **aplicación móvil** orientada a los estudiantes, donde podrán consultar sus datos, recibir notificaciones y visualizar sus predicciones de rendimiento.

El sistema operará con tres tipos de usuarios:

- **Administrador:** gestiona usuarios, materias, cursos, configuración del modelo de IA.
- **Docente:** registra notas, asistencia, participación y accede al dashboard predictivo.
- **Estudiante:** consulta su historial académico, predicciones y alertas.

4.3 Equipo SCRUM

Con el objetivo de desarrollar este sistema de forma ordenada, incremental y centrada en el valor, se aplicará el marco de trabajo **ágil SCRUM**, el cual ha sido adaptado a la dinámica del equipo y las necesidades del proyecto.

- **Product Owner:**

El Product Owner es el responsable de definir y mantener actualizado el Product Backlog, priorizando las funcionalidades en función del valor que aportan al negocio. Su rol es ser el nexo entre los stakeholders y el equipo de desarrollo, asegurando que el producto evolucione en la dirección correcta.

- **Scrum Máster:**

El Scrum Máster tiene la misión de velar por el cumplimiento de los principios y prácticas ágiles dentro del equipo. Actúa como un facilitador, ayudando a resolver impedimentos, coordinando los eventos SCRUM (Daily Standup, Sprint Planning, Sprint Review y Sprint Retrospective), y promoviendo la mejora continua.

- **Development Team:**

Conformado por desarrolladores con conocimientos multidisciplinarios, el equipo de desarrollo es el encargado de construir los incrementos funcionales del sistema. Entre sus responsabilidades están: el desarrollo frontend y backend, pruebas, integración de sistemas externos y despliegue en la nube. El equipo es autoorganizado y colaborativo.

- **Stakeholders:**

Incluyen a las partes interesadas del proyecto, como los clientes finales, gerentes de la tienda, personal de soporte y representantes de los fabricantes. Aportan requerimientos, validan funcionalidades y retroalimentan al Product Owner durante las revisiones de Sprint.

Tabla del Equipo SCRUM

Rol	Nombre completo	Registro
Product Owner	Cuéllar Flores Victor Hugo	222008180
Scrum Máster	Méndez López Sebastián	222055987
Development Team		

4.4 Definiciones, Acrónimos y Abreviaturas

Abreviatura	Significado
IA	Inteligencia artificial
ML	Machine Learning
SPA	<i>Single Page Application</i> : Aplicación web que se ejecuta en una sola página HTML dinámica.
PO	<i>Product Owner</i> : Dueño del producto en SCRUM.
SM	<i>Scrum Máster</i> : Facilitador de la metodología SCRUM.
DB	<i>Database (Base de Datos)</i> : Sistema de almacenamiento estructurado de datos.
RF	<i>Requisito Funcional</i> .
RNF	<i>Requisito No Funcional</i> .
CU	<i>Casos de Uso</i> .
HU	<i>Historias de Usuario</i> .

Abreviatura	Significado
SP	<i>Sprint.</i>
Flutter	Framework de Google para desarrollo de apps móviles con un solo código base.
Angular	Framework frontend mantenido por Google, orientado a SPAs modernas.
UML	Lenguaje de modelado unificado
C4	Modelo de arquitectura de software en 4 niveles

4.5 Funciones del Producto

Módulo	Funcionalidad
Gestión de Usuarios	Registro, edición y control de acceso por roles: administrador, docente, estudiante.
Gestión Académica	Registro y edición de materias, cursos, paralelos y asignaciones docentes.
Registro de Alumnos	Alta, modificación y seguimiento de alumnos por curso.
Registro de Evaluaciones	Registro de notas por periodo académico y por materia.
Registro de Asistencia y Participación	Registro porcentual por materia, por periodo y por estudiante.
Dashboard General	Visualización de estadísticas agregadas del sistema (promedios, rendimiento general, etc.).
Ficha del Estudiante	Vista individual con datos históricos y predicción académica personalizada.
Modelo Predictivo de Rendimiento	IA entrenada con datos académicos para proyectar el rendimiento futuro del estudiante.
Panel Administrativo	Gestión de configuraciones globales del sistema, parámetros de predicción, control de acceso.
Plataforma Web	Accesible desde navegador, enfocada en administración, registro docente y visualización de IA.

Módulo	Funcionalidad
Aplicación Móvil	Accesible desde dispositivos móviles para visualización estudiantil y notificaciones.

4.6 PRODUCT BACKLOG

Código	Rol Responsable	Característica / Funcionalidad
RF1	Product Owner	Gestión de Usuarios: Permitir el registro, edición y eliminación de usuarios, así como la asignación de roles (administrador, docente, alumno) y permisos asociados.
RF2	SCRUM Master	Control de Accesos: Implementar autenticación y autorización segura para cada tipo de usuario.
RF3	Product Owner	Administración Académica: Registrar materias, cursos, niveles y paralelos, y asignar docentes a cursos específicos.
RF4	SCRUM Master	Gestión de Alumnos: Crear, modificar y vincular estudiantes a cursos y materias dentro del sistema.
RF5	Product Owner	Registro de Evaluaciones: Permitir a los docentes ingresar calificaciones por materia y por periodo.
RF6	SCRUM Master	Registro de Asistencia y Participación: Registrar y visualizar asistencia y participación de los estudiantes en clase.
RF7	Product Owner	Ficha Académica del Estudiante: Mostrar historial de notas, asistencia, participación y predicción académica individual.
RF8	SCRUM Master	Implementación del Modelo de Predicción: Entrenar e integrar un modelo de machine learning para estimar el rendimiento académico futuro (Random Forest o Regresión Lineal).
RF9	Product Owner	Visualización Global: Implementar un dashboard con métricas generales: promedios, participación, predicciones agregadas.

Código	Rol Responsable	Característica / Funcionalidad
RF10	SCRUM Master	Comparación Real vs Predicho: Mostrar visualmente el rendimiento actual frente al proyectado por IA, usando gráficos interactivos.
RF11	Product Owner	Notificaciones y Alertas: Enviar alertas a docentes y estudiantes sobre bajo rendimiento o cambios en sus proyecciones.
RF12	SCRUM Master	Desarrollo de Aplicación Móvil: Crear app en Flutter para acceso del alumno a calificaciones, predicciones y alertas.
RF13	Product Owner	Configuración del Sistema: Permitir la definición de periodos académicos, parámetros de predicción y niveles de alerta.
RF14	SCRUM Master	Integración Backend-Frontend: Crear y documentar APIs RESTful para la comunicación segura entre backend Django y frontend Angular o React.
RF15	Product Owner	Validación de Reglas del Negocio: Asegurar que el flujo del sistema respete condiciones académicas reales (mínimos de nota, asistencia, etc.).
RF16	SCRUM Master	Pruebas y Control de Calidad: Diseñar e implementar pruebas unitarias, de integración y aceptación para validar funcionalidades.
RF17	Product Owner	Gestión de Parámetros de IA: Permitir el ajuste fino de las variables predictoras y su peso en el modelo.
RF18	SCRUM Master	Seguridad Técnica: Implementar cifrado, validaciones y protección ante ataques como inyecciones SQL, CSRF y XSS.
RF19	Product Owner	Manual de Usuario: Crear una guía básica de uso del sistema para docentes y estudiantes.
RF20	SCRUM Master	Control de Versiones y Documentación Técnica: Gestionar el código en GitHub y documentar la arquitectura, modelo y endpoints del sistema.

4.7 Requisitos Funcionales

Plataforma Web

CÓDIGO	DESCRIPCIÓN
RF1	Gestión de Usuarios: Registro, edición, eliminación y asignación de roles diferenciados (administrador, docente, alumno).
RF2	Administración de Usuarios del Sistema: Supervisión de cuentas activas, control de accesos y permisos por parte del administrador.
RF3	Gestión Académica: Registro y asignación de cursos, paralelos y materias; vinculación entre docentes y materias.
RF4	Registro Académico: Ingreso de calificaciones por materia y periodo, visualización de historial académico por estudiante.
RF5	Control de Asistencia y Participación: Registro por parte del docente, con cálculo automático de porcentajes por materia.
RF6	Visualización General: Dashboard con métricas agregadas del sistema como promedio de notas, porcentaje de asistencia, predicciones generales.
RF7	Ficha Individual del Alumno: Visualización del historial académico del alumno y su predicción generada por IA.
RF8	Configuración del Sistema: Ajuste de parámetros globales como periodos, pesos de variables, niveles de alerta del modelo predictivo.
RF9	Alertas y Notificaciones: Sistema de alertas para informar al docente sobre posibles riesgos académicos detectados.
RF10	Control de Accesos: Implementación de inicio de sesión seguro y control de permisos por rol.
RF11	Administración del Modelo de IA: Carga del dataset, entrenamiento, ajuste de hiperparámetros y actualización de predicciones.

Aplicación Móvil

CÓDIGO	DESCRIPCIÓN
RF1	Gestión de Usuario: Acceso del estudiante a su perfil con datos personales y académicos.
RF2	Visualización Académica: Consulta de calificaciones actuales, historial de materias y estados de asistencia.
RF3	Gestión de Autenticación y Autorización: Inicio de sesión seguro mediante validación de credenciales y tokens.
RF4	Visualización de Predicción Académica: Consulta de la predicción generada por el sistema sobre su rendimiento futuro.
RF5	Notificaciones: Recepción de alertas sobre cambios en calificaciones, predicciones o mensajes del docente.
RF6	Diseño Adaptativo: Interfaz responsive y amigable, adaptada a distintos tamaños de pantalla.

4.8 Requisitos No Funcionales

CÓDIGO	DESCRIPCIÓN
RNF1	Cambio de contraseña: Los usuarios podrán modificar su contraseña de forma segura dentro del sistema mediante formularios validados.
RNF2	Encriptación de contraseña: Todas las contraseñas serán almacenadas en la base de datos utilizando algoritmos de cifrado seguros como bcrypt o Argon2.
RNF3	Tiempo de respuesta: El sistema deberá responder a cualquier acción del usuario en un tiempo menor a 2 segundos.
RNF4	Compatibilidad multiplataforma: La plataforma web será compatible con los navegadores modernos (Chrome, Firefox, Edge), y la app móvil con Android e iOS.
RNF5	Disponibilidad: El sistema deberá estar disponible en un 99% del tiempo durante horas laborales (07:00 - 22:00).
RNF6	Escalabilidad: La arquitectura permitirá incorporar nuevas funcionalidades, usuarios y módulos sin comprometer el rendimiento.

CÓDIGO	DESCRIPCIÓN
RNF7	Seguridad de la información: Se implementarán políticas de control de acceso, cifrado de datos sensibles y protección contra inyecciones SQL, XSS y CSRF.
RNF8	Backup automático: La base de datos se respaldará automáticamente cada 24 horas para garantizar la integridad de los datos.

4.9 Lista de Casos de Uso (Web y Móvil)

Casos de Uso – Plataforma Web

NRO	DESCRIPCIÓN	ESTADO
1	Registrar nuevos usuarios (administrador, docente, alumno)	Terminado
2	Editar información de usuarios registrados	Terminado
3	Asignar roles y permisos a los usuarios	Terminado
4	Crear materias, cursos y paralelos	Terminado
5	Asignar docentes a materias y cursos	Terminado
6	Registrar calificaciones por estudiante y por periodo	Terminado
7	Registrar asistencia de los alumnos por clase	Terminado
8	Registrar nivel de participación de los estudiantes	Terminado
9	Visualizar estadísticas académicas generales (dashboard)	Terminado
10	Consultar ficha individual del estudiante	Terminado
11	Entrenar modelo de IA con dataset cargado	Terminado
12	Generar predicción de rendimiento por estudiante	Terminado
13	Visualizar comparativo rendimiento real vs predicho	Terminado
14	Configurar parámetros del sistema (periodos, alertas, etc.)	Terminado
15	Generar alertas para docentes por bajo rendimiento detectado	Terminado
16	Gestionar cambios de contraseña para usuarios	Terminado

NRO	DESCRIPCIÓN	ESTADO
17	Realizar búsquedas y filtrado de alumnos por criterios	Terminado
18	Exportar reportes académicos en formato PDF o Excel	Terminado

Casos de Uso – Aplicación Móvil

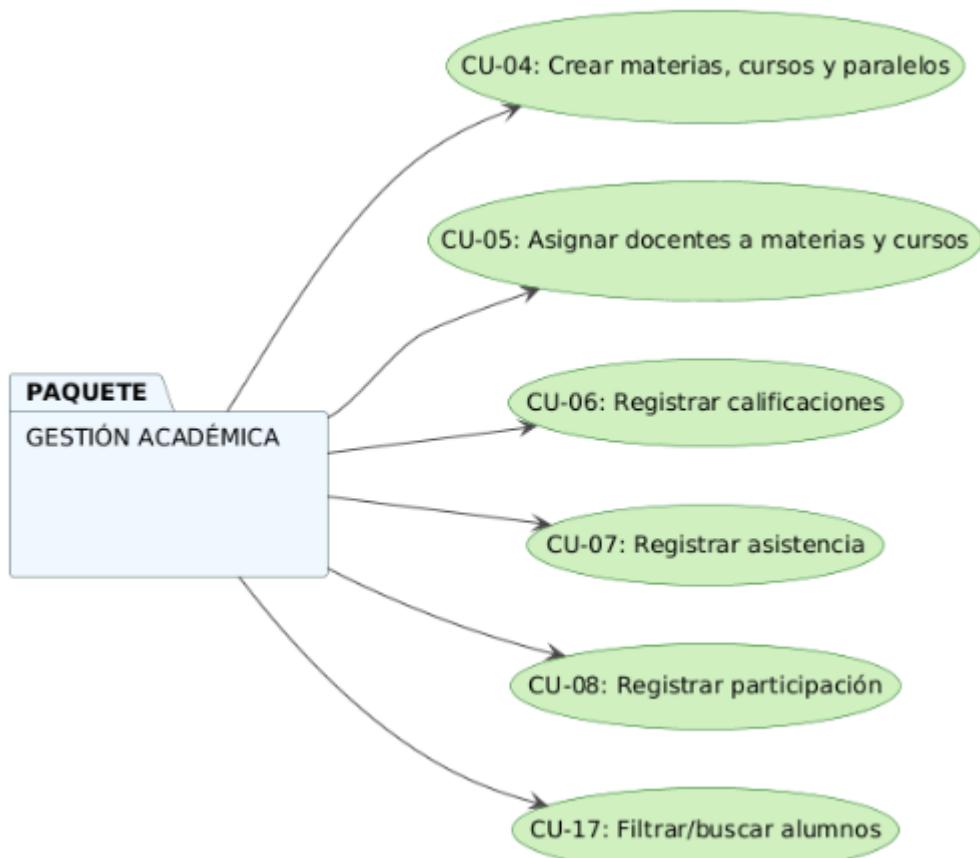
NRO	DESCRIPCIÓN	ESTADO
1	Iniciar sesión según rol (con validación segura)	Terminado
2	Visualizar perfil académico del estudiante	Terminado
3	Consultar calificaciones por materia y periodo	Terminado
4	Visualizar asistencia y participación acumulada	Terminado
5	Consultar predicción de rendimiento académico	Terminado
6	Cambiar contraseña desde la aplicación	Terminado
7	Cerrar sesión	Terminado

4.10 Paquetes y casos de Uso

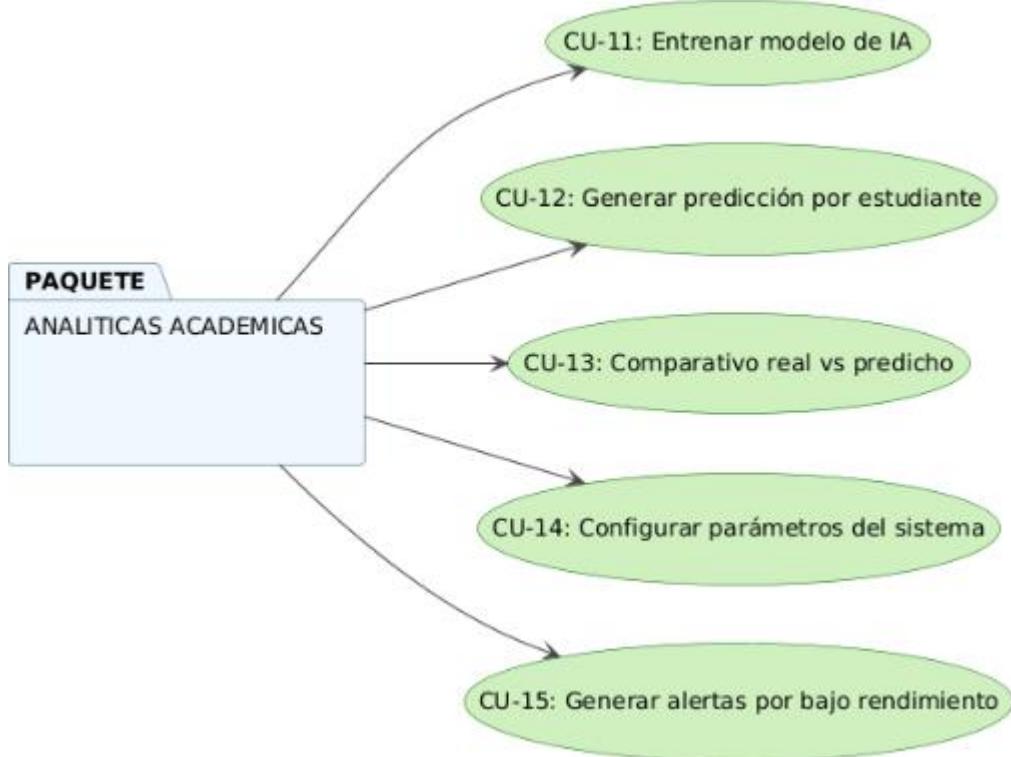
4.10.1 Paquete de Gestión de usuarios



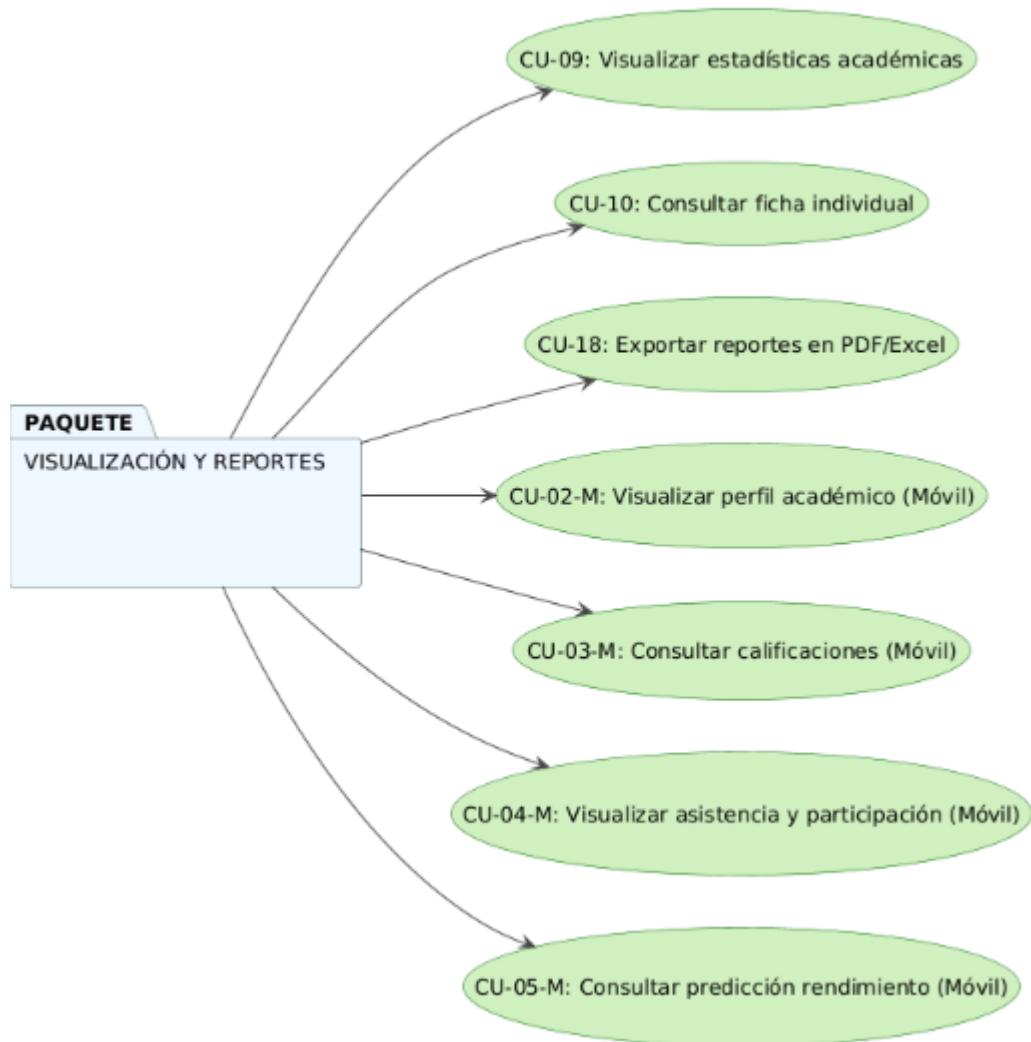
4.10.2 Paquete de Gestión Académica



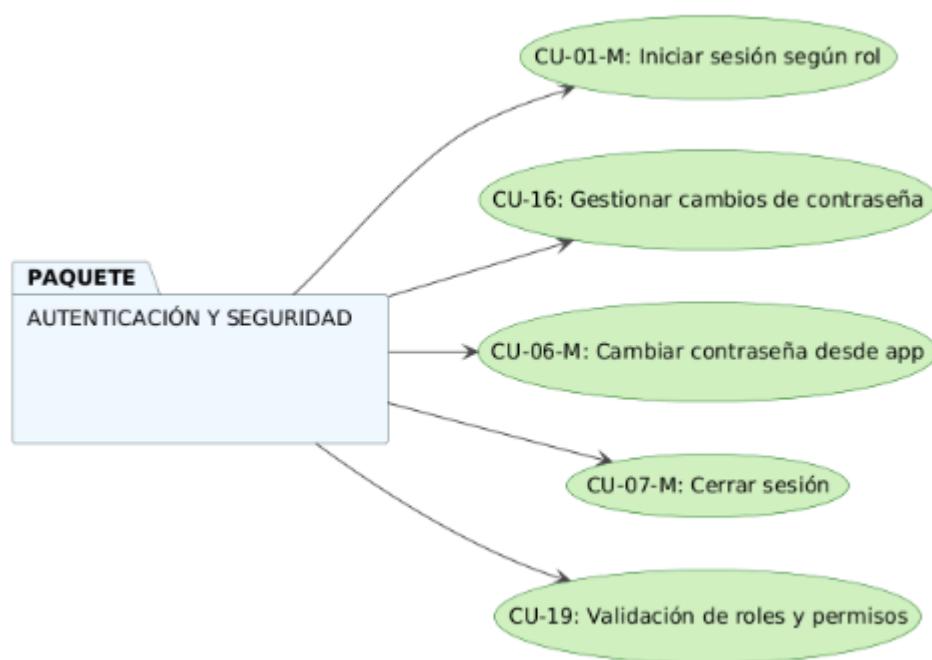
4.10.3 Paquete de Analíticas Académicas



4.10.4 Paquete de Visualización y Reportes



4.10.5 Paquete de Autenticación y Seguridad



4.11 Planificación Sprint (Diagrama de Gantt)

4.11.1 Sprint 1 13/05 – 20/05

Código	Descripción	13/05	14/05	15/05	16/05	17/05	18/05	19/05	20/05
SP1-1	Documentación y revisión general	●							
SP1-2	Organización de paquetes y casos	●							
SP1-3	Diseño y registro de nuevos usuarios (web)		●	●					
SP1-4	Edición de usuarios registrados			●	●				
SP1-5	Asignación de roles y permisos				●	●			
SP1-6	Consulta ficha individual estudiante					●	●		

Código	Descripción	13/05	14/05	15/05	16/05	17/05	18/05	19/05	20/05
SP1-7	Gestión de cambios de contraseña						●	●	
SP1-8	Revisión y pruebas integradas							●	●

4.11.2 Sprint 2 21/05 – 27/05

CÓDIGO	DESCRIPCIÓN	21/05	22/05	23/05	24/05	25/05	26/05	27/05
SP2-1	Creación de materias, cursos y paralelos	●						
SP2-2	Asignación de docentes		●					
SP2-3	Registro de calificaciones		●					
SP2-4	Registro de asistencia y participación			●				
SP2-5	Filtrado y búsqueda de alumnos			●				
SP2-6	Dashboard académico (estadísticas)				●			
SP2-7	Ficha individual del estudiante (detalle)				●			
SP2-8	Reportes exportables (PDF/Excel)					●		
SP2-9	Pruebas de funcionalidades web					●	●	
SP2-10	Retrospectiva Sprint 2							●

4.11.3 Sprint 3 28/05 – 03/06

CÓDIGO	DESCRIPCIÓN	28/05	29/05	30/05	31/05	01/06	02/06	03/06
SP3-1	Entrenamiento de modelo IA con dataset	●						
SP3-2	Generación de predicción por estudiante		●					
SP3-3	Comparativo rendimiento real vs predicho			●				
SP3-4	Configuración del sistema (parámetros, alertas)				●			
SP3-5	Generación de alertas para docentes				●			
SP3-6	Visualización académica en app móvil					●		
SP3-7	Predicción y ficha en app móvil						●	
SP3-8	Pruebas completas (web + móvil + IA)					●	●	
SP3-9	Revisión general y presentación							●
SP3-10	Retrospectiva Sprint 3							●

5. PROCESO DE DESARROLLO DE SOFTWARE

Sprint 1

Código	Nombre del Caso de Uso
CU01	Registro de usuarios
CU02	Gestión de roles y permisos
CU03	Edición de información de usuarios
CU04	Cambio de contraseña
CU05	Consulta de ficha individual del estudiante

1. Sprint Planning

1.1. Objetivos del Sprint

Durante este primer sprint se busca construir las funcionalidades fundamentales de autenticación, gestión de usuarios y permisos, necesarias para asegurar el acceso controlado al sistema. Además, se desarrollarán componentes básicos de la ficha estudiantil, necesarios para el posterior análisis del rendimiento académico.

- Desarrollar el módulo de autenticación: login, logout y recuperación/cambio de contraseña.
- Implementar la funcionalidad de registro y edición de usuarios.
- Implementar la gestión de roles y permisos para el acceso a funcionalidades diferenciadas por perfil.
- Construir el módulo de consulta de la ficha individual del estudiante.
- Sentar las bases de diseño de la arquitectura del sistema (Modelo C4).
- Realizar pruebas de funcionamiento básico (unitarias y de integración).

1.2. Historias de usuario (tarjetas 3C, Planning Póker, prototipos)

CU 01: Registro de Usuarios

Registro de Usuarios	
ID: 1	DESCRIPCION: Permitir el registro de nuevos usuarios al sistema (administrador, docente y alumno)
PRIORIDAD: Alta	ESTIMACION: 5hr

CRITERIOS DE ACEPTACION: El sistema debe permitir el registro mediante un formulario con campos obligatorios según el tipo de usuario. Al registrarse, el usuario debe quedar asociado a un rol.

DESARROLLADOR A CARGO: Cuéllar Flores Victor Hugo

Registro de Usuarios

Tipo de Usuario

Nombre de Usuario

Correo Electrónico

Nombres

Apellidos

Contraseña

Confirmar Contraseña

CU 02: Gestión de Roles

Gestión de Roles	
ID: 2	DESCRIPCION: Gestionar roles de los usuarios del sistema
PRIORIDAD: Alta	ESTIMACION: 5hr
CRITERIOS DE ACEPTACION: Los administradores deben poder crear un nuevo rol asignando un nombre único. Los administradores pueden modificar permisos asociados a un rol.	
DESARROLLADOR A CARGO: Méndez López Sebastián	

Gestión de Roles

Crear Nuevo Rol

Nombre del Rol

Descripción

Permisos

Crear Usuarios Editar Usuarios Eliminar Usuarios Asignar Roles Ver Estudiantes Editar Estudiantes

Crear Rol

Nombre	Descripción	Permisos	Acciones
Administrador	Control total del sistema	Crear Usuarios Editar Usuarios Eliminar Usuarios Asignar Roles Ver Estudiantes Editar Estudiantes	Editar Eliminar
Profesor	Gestión de materias y estudiantes	Ver Estudiantes Editar Estudiantes	Editar Eliminar
Estudiante	Acceso a contenido educativo	Ver Contenido	Editar Eliminar

CU 03: Edición de Usuarios

Edición de Usuarios	
ID: 3	DESCRIPCION: Permitir modificar la información registrada de los usuarios
PRIORIDAD: Media	ESTIMACION: 4hr
CRITERIOS DE ACEPTACION: El sistema debe permitir al administrador editar datos personales del usuario y cambiar el rol asignado.	
DESARROLLADOR A CARGO: Cuéllar Flores Victor Hugo	

Edición de Usuarios					
Buscar usuario por nombre o correo <input type="button" value="Buscar"/>					
Nombre	Usuario	Correo	Rol	Acciones	
Juan Pérez	jperez	juan.perez@correo.com	Administrador	Editar	
María López	mlopez	maria.lopez@correo.com	Profesor	Editar	
Carlos González	cgonzalez	carlos.gonzalez@correo.com	Estudiante	Editar	

CU 04: Gestión de Contraseña

Gestión de Contraseña	
ID: 4	DESCRIPCION: Permitir a los usuarios cambiar su contraseña
PRIORIDAD: Media	ESTIMACION: 3hr

CRITERIOS DE ACEPTACION: El usuario debe poder ingresar su contraseña actual, y una nueva. El sistema debe validar que las contraseñas coincidan y cumplan con criterios mínimos de seguridad.

DESARROLLADOR A CARGO: Cuéllar Flores Victor Hugo

Gestión de Contraseña

Cambiar Contraseña

Contraseña Actual

Ingrese contraseña actual

Nueva Contraseña

Ingrese nueva contraseña

Confirmar Nueva Contraseña

Confirme nueva contraseña

Fortaleza de la Contraseña

Cambiar Contraseña

Recuperación de Contraseña

Correo Electrónico

Ingrese su correo electrónico

Enviar Enlace de Recuperación

CU 05: Consulta de Ficha Estudiantil

Registro de Usuarios

ID: 5	DESCRIPCION: Permitir la visualización de la ficha individual de un estudiante
PRIORIDAD: Media	ESTIMACION: 4hr
CRITERIOS DE ACEPTACION: El sistema debe mostrar información del estudiante como nombre, carrera, materias inscritas y rendimiento académico básico. Solo docentes y administradores tienen acceso.	
DESARROLLADOR A CARGO: Méndez López Sebastián	

Ficha Estudiantil

Buscar estudiante por nombre o código

Buscar



Carlos González
Código: EST-2023-0103
Correo: carlos.gonzalez@correo.com

Información Personal	Rendimiento Académico
Fecha de Nacimiento: 15/07/2001	Promedio General: 8.5/10
Dirección: Av. Principal 123	Porcentaje de Asistencia: 92%
Teléfono: 555-1234	Participaciones: Alta
Grado: Segundo año	Predicción IA: Rendimiento Alto

Materias Inscritas	Promedio	Asistencia
Matemáticas	8.7	95%
Física	7.9	88%
Programación	9.5	98%
Literatura	8.2	89%

1.3. Contexto del sistema

El sistema “Aula Inteligente” es una plataforma web y móvil dirigida a la gestión académica, que permite registrar usuarios (alumnos, docentes, administradores), administrar su información y utilizar IA para predecir el rendimiento académico.

El sistema está compuesto por distintos módulos que permiten:

- Registro y autenticación de usuarios.
- Asignación de roles y permisos.
- Acceso a fichas individuales de estudiantes.
- Cambio de contraseñas.
- Visualización de datos académicos.
- Predicción del rendimiento estudiantil mediante algoritmos de IA (futuro sprint).

La arquitectura se basa en una estructura cliente-servidor, donde los clientes pueden ser navegadores o aplicaciones móviles, y el servidor expone una API que maneja la lógica de negocio, conexión con la base de datos y con los servicios de predicción.

1.4. Sprint backlog – Sprint 1

ID	Tarea	Tipo	Estimación (hr)	Responsable	Estado
1	Diseño de base de datos para gestión de usuarios y roles	Diseño	3	Cuéllar Flores Victor Hugo	Terminado
2	Backend: Registro y edición de usuarios	Desarrollo	4	Cuéllar Flores Victor Hugo	Terminado
3	Frontend Web: Formulario de registro y edición de usuarios	Desarrollo	3	Cuéllar Flores Victor Hugo	Terminado
4	Frontend Móvil: Formulario de registro y edición de usuarios	Desarrollo	3	Cuéllar Flores Victor Hugo	Terminado
5	Backend: Asignación de roles y permisos	Desarrollo	4	Méndez López Sebastián	Terminado
6	Frontend Web: Gestión de roles y permisos	Desarrollo	3	Méndez López Sebastián	Terminado
7	Frontend Móvil: Visualización de roles (solo lectura)	Desarrollo	2	Méndez López Sebastián	Terminado
8	Backend: Consulta de ficha individual del estudiante	Desarrollo	3	Méndez López Sebastián	Terminado
9	Frontend Web: Vista de ficha individual del estudiante	Desarrollo	3	Méndez López Sebastián	Terminado
10	Frontend Móvil: Vista de ficha individual del estudiante	Desarrollo	2	Méndez López Sebastián	Terminado
11	Backend: Gestión de cambio de contraseña	Desarrollo	2	Cuéllar Flores Victor Hugo	Terminado

ID	Tarea	Tipo	Estimación (hr)	Responsable	Estado
12	Frontend Web: Formulario de cambio de contraseña	Desarrollo	2	Cuéllar Flores Victor Hugo	Terminado
13	Frontend Móvil: Formulario de cambio de contraseña	Desarrollo	2	Cuéllar Flores Victor Hugo	Terminado
14	Pruebas unitarias y de integración (CU01– CU03)	Pruebas	3	Cuéllar Flores Victor Hugo	Terminado
15	Pruebas unitarias y de integración (CU10– CU16)	Pruebas	3	Méndez López Sebastián	Terminado
16	Revisión general y documentación Sprint 1	Documentación	2	Ambos	Terminado
17	Retrospectiva Sprint 1	Reunión	1	Ambos	Terminado

1.5. Equipo SCRUM

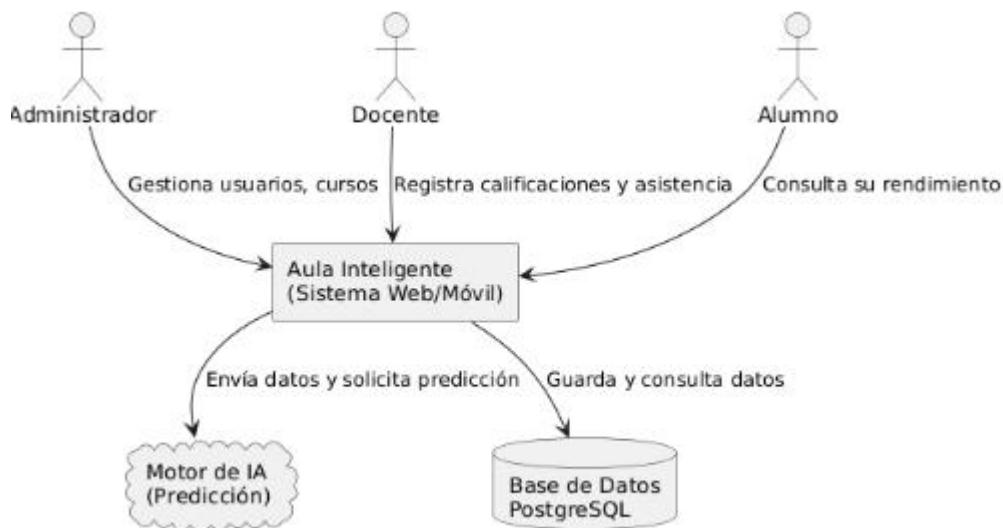
Rol	Nombre
Product Owner	Cuellar Flores Victor Hugo
Scrum Master	Méndez López Sebastián
Development Team	

2. Proceso/patrón de desarrollo por Historia de Usuario

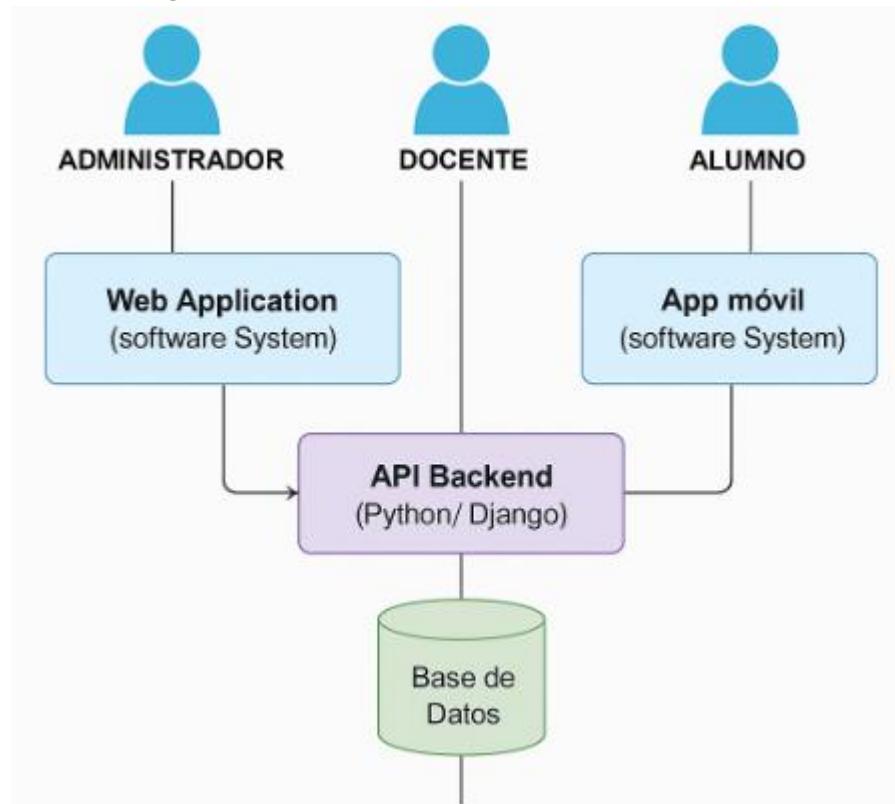
2.1 Diseño

2.1.1 Diseñar de la arquitectura (Modelo C4)

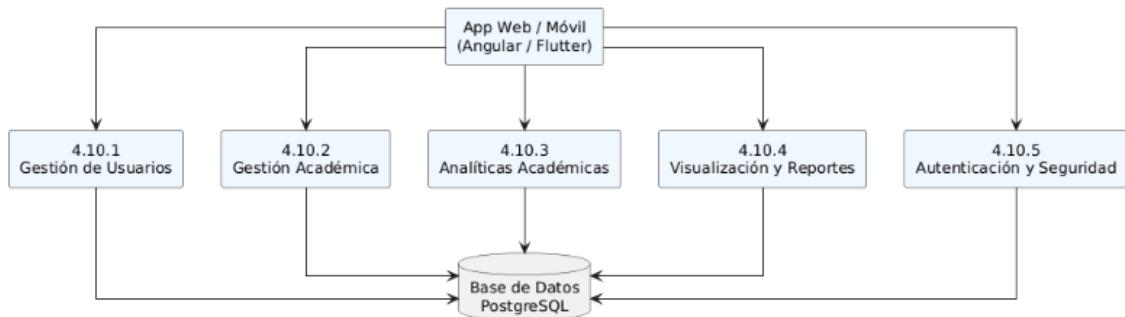
2.1.1.1 Diagrama de contexto



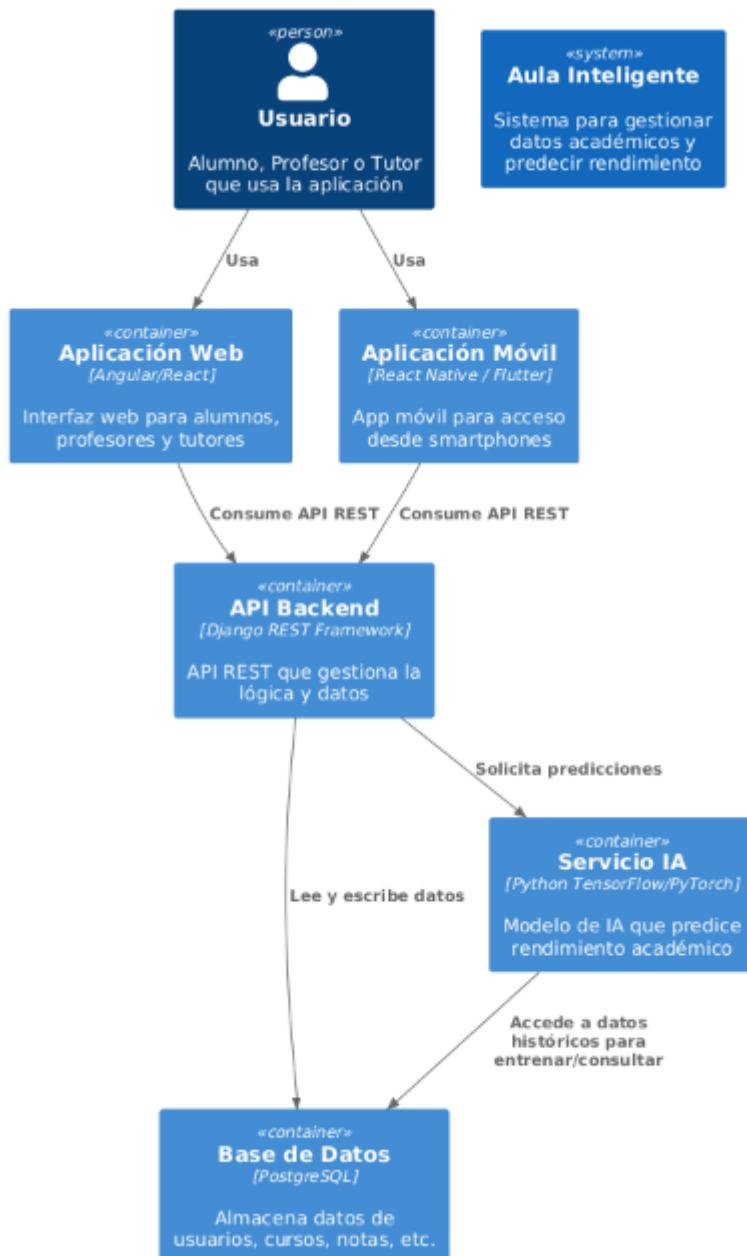
2.1.1.2 Diagrama de contenedores



2.1.1.3 Diagrama de componentes

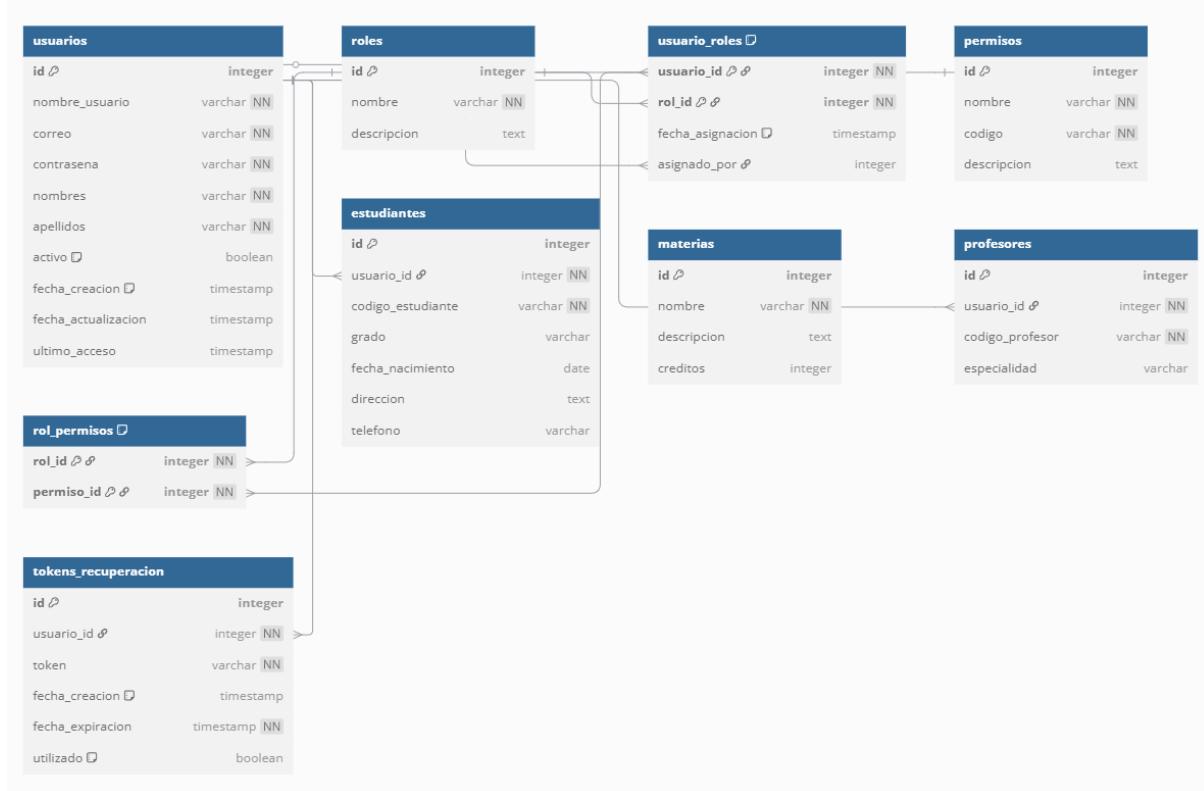


2.1.1.4 Diagrama de código



2.1.2 Diseño de datos

a) Diagrama de clase



b) Script

```

DROP TABLE IF EXISTS tokens_recuperacion;
DROP TABLE IF EXISTS rol_permisos;
DROP TABLE IF EXISTS permisos;
DROP TABLE IF EXISTS usuario_roles;
DROP TABLE IF EXISTS roles;
DROP TABLE IF EXISTS profesores;
DROP TABLE IF EXISTS estudiantes;
DROP TABLE IF EXISTS materias;
DROP TABLE IF EXISTS usuarios;

CREATE TABLE usuarios (
    id SERIAL PRIMARY KEY,
    nombre_usuario VARCHAR(50) UNIQUE NOT NULL,
    correo VARCHAR(100) UNIQUE NOT NULL,
    contrasena VARCHAR(255) NOT NULL,
    nombres VARCHAR(100) NOT NULL,
    apellidos VARCHAR(100) NOT NULL,
    activo BOOLEAN DEFAULT TRUE,
    fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    fecha_actualizacion TIMESTAMP,
    ultimo_acceso TIMESTAMP
);

CREATE TABLE roles (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(50) NOT NULL,
    descripcion TEXT
);

CREATE TABLE permisos (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(50) NOT NULL,
    codigo VARCHAR(50) NOT NULL,
    descripcion TEXT
);

CREATE TABLE estudiantes (
    id SERIAL PRIMARY KEY,
    usuario_id INTEGER NOT NULL,
    codigo_estudiante VARCHAR(50) NOT NULL,
    grado VARCHAR(50),
    fecha_nacimiento DATE,
    direccion TEXT,
    telefono VARCHAR(50)
);

CREATE TABLE materias (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(50) NOT NULL,
    descripcion TEXT,
    creditos INTEGER
);

CREATE TABLE profesores (
    id SERIAL PRIMARY KEY,
    usuario_id INTEGER NOT NULL,
    codigo_profesor VARCHAR(50),
    especialidad VARCHAR(50)
);

CREATE TABLE usuario_roles (
    usuario_id INTEGER NOT NULL,
    rol_id INTEGER NOT NULL,
    fecha_asignacion TIMESTAMP,
    asignado_por INTEGER
);

CREATE TABLE rol_permisos (
    rol_id INTEGER NOT NULL,
    permiso_id INTEGER NOT NULL
);

CREATE TABLE tokens_recuperacion (
    id SERIAL PRIMARY KEY,
    usuario_id INTEGER NOT NULL,
    token VARCHAR(255) NOT NULL,
    fecha_creacion TIMESTAMP,
    fecha_expiracion TIMESTAMP,
    utilizado BOOLEAN
);

```

```
        fecha_actualizacion TIMESTAMP,
        ultimo_acceso TIMESTAMP
    );

CREATE TABLE roles (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(50) UNIQUE NOT NULL,
    descripcion TEXT
);

CREATE TABLE permisos (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(100) UNIQUE NOT NULL,
    codigo VARCHAR(50) UNIQUE NOT NULL,
    descripcion TEXT
);

CREATE TABLE usuario_roles (
    usuario_id INTEGER NOT NULL,
    rol_id INTEGER NOT NULL,
    fecha_asignacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    asignado_por INTEGER,
    PRIMARY KEY (usuario_id, rol_id),
    CONSTRAINT fk_usuario_roles_usuario FOREIGN KEY (usuario_id)
REFERENCES usuarios(id) ON DELETE CASCADE,
    CONSTRAINT fk_usuario_roles_rol FOREIGN KEY (rol_id) REFERENCES
roles(id) ON DELETE CASCADE,
    CONSTRAINT fk_usuario_roles_asignador FOREIGN KEY (asignado_por)
REFERENCES usuarios(id)
);

CREATE TABLE rol_permisos (
    rol_id INTEGER NOT NULL,
    permiso_id INTEGER NOT NULL,
    PRIMARY KEY (rol_id, permiso_id),
    CONSTRAINT fk_rol_permisos_rol FOREIGN KEY (rol_id) REFERENCES
roles(id) ON DELETE CASCADE,
    CONSTRAINT fk_rol_permisos_permiso FOREIGN KEY (permiso_id)
REFERENCES permisos(id) ON DELETE CASCADE
);

CREATE TABLE materias (
    id SERIAL PRIMARY KEY,
```

```
        nombre VARCHAR(100) NOT NULL,
        descripcion TEXT,
        creditos INTEGER
    );

CREATE TABLE estudiantes (
    id SERIAL PRIMARY KEY,
    usuario_id INTEGER UNIQUE NOT NULL,
    codigo_estudiante VARCHAR(20) UNIQUE NOT NULL,
    grado VARCHAR(20),
    fecha_nacimiento DATE,
    direccion TEXT,
    telefono VARCHAR(20),
    CONSTRAINT fk_estudiante_usuario FOREIGN KEY (usuario_id)
    REFERENCES usuarios(id) ON DELETE CASCADE
);

CREATE TABLE profesores (
    id SERIAL PRIMARY KEY,
    usuario_id INTEGER UNIQUE NOT NULL,
    codigo_profesor VARCHAR(20) UNIQUE NOT NULL,
    especialidad VARCHAR(100),
    CONSTRAINT fk_profesor_usuario FOREIGN KEY (usuario_id)
    REFERENCES usuarios(id) ON DELETE CASCADE
);

CREATE TABLE tokens_recuperacion (
    id SERIAL PRIMARY KEY,
    usuario_id INTEGER NOT NULL,
    token VARCHAR(255) NOT NULL,
    fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    fecha_expiracion TIMESTAMP NOT NULL,
    utilizado BOOLEAN DEFAULT FALSE,
    CONSTRAINT fk_token_usuario FOREIGN KEY (usuario_id) REFERENCES
    usuarios(id) ON DELETE CASCADE
);

INSERT INTO roles (nombre, descripcion) VALUES
('Administrador', 'Control total del sistema'),
('Profesor', 'Gestión de materias y estudiantes'),
('Estudiante', 'Acceso a contenido educativo');

INSERT INTO permisos (nombre, codigo, descripcion) VALUES
```

```

('Crear Usuario', 'CREAR_USUARIO', 'Permite crear nuevos usuarios'),
('Editar Usuario', 'EDITAR_USUARIO', 'Permite modificar usuarios existentes'),
('Eliminar Usuario', 'ELIMINAR_USUARIO', 'Permite eliminar usuarios'),
('Asignar Roles', 'ASIGNAR_ROLES', 'Permite asignar roles a usuarios'),
('Ver Estudiantes', 'VER_ESTUDIANTES', 'Permite ver fichas de estudiantes'),
('Editar Estudiantes', 'EDITAR_ESTUDIANTES', 'Permite modificar datos de estudiantes');

INSERT INTO rol_permisos (rol_id, permiso_id) VALUES
(1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6),
(2, 5), (2, 6);

INSERT INTO usuarios (nombre_usuario, correo, contrasena, nombres,
apellidos, activo) VALUES
('admin', 'admin@aulainteligente.edu',
'$2a$12$1InE3SxJYCsoS7RD8crJv0GPZp5kQgIxKUGJz.W.YgNAhGMHXkGgW',
'Administrador', 'Sistema', TRUE);

INSERT INTO usuario_roles (usuario_id, rol_id) VALUES
(1, 1);

CREATE INDEX idx_usuarios_correo ON usuarios(correo);
CREATE INDEX idx_estudiantes_usuario_id ON estudiantes(usuario_id);
CREATE INDEX idx_profesores_usuario_id ON profesores(usuario_id);

```

c) Mapeo

USUARIO

i d	nom bre_u suario	co rre o	cont rase na	no mb res	ap elli dos	a ct iv o	fecha_ creaci on	fecha_actu alizacion	ulti mo_ acce so

ROLES

id	nombre	descripcion
-----------	---------------	--------------------

PERMISOS

id	nombre	codigo	descripcion
-----------	---------------	---------------	--------------------

USUARIO_ROLES

usuario_id	rol_id	fecha_asignacion	asignado_por
-------------------	---------------	-------------------------	---------------------

ROL_PERMISOS

rol_id	permiso_id
---------------	-------------------

MATERIAS

id	nombre	descripcion	creditos
-----------	---------------	--------------------	-----------------

ESTUDIANTES

id	usuari o_id	codigo_estud iante	grado	fecha_nacimient o	direccio n	telefon o
-----------	------------------------	-------------------------------	--------------	------------------------------	-----------------------	----------------------

PROFESORES

id	usuario_id	codigo_profesor	especialidad
-----------	-------------------	------------------------	---------------------

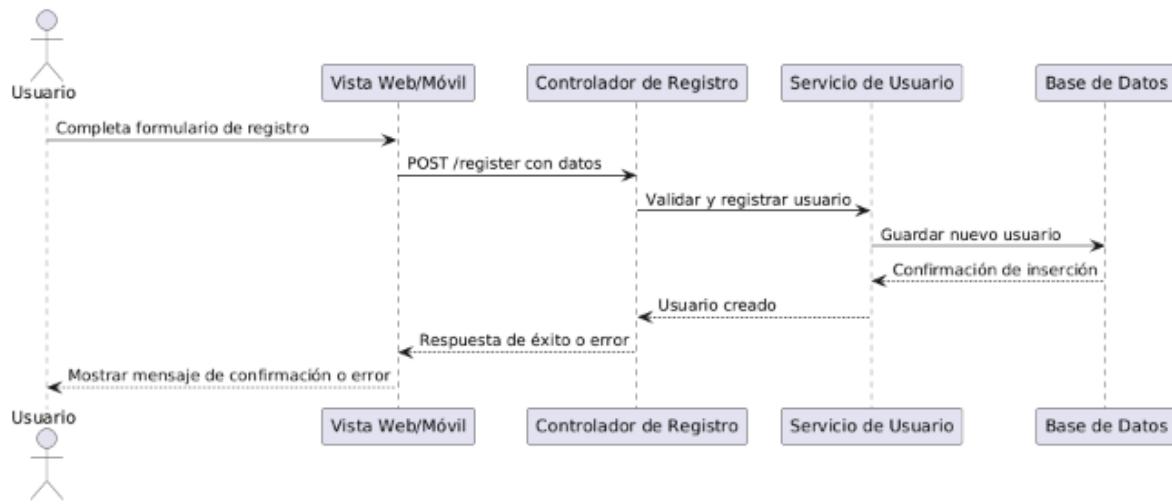
TOKENS_RECUPERACION

id	usuario_ id	toke n	fecha_creatio n	fecha_expiracio n	utilizad o
-----------	------------------------	-------------------	----------------------------	------------------------------	-----------------------

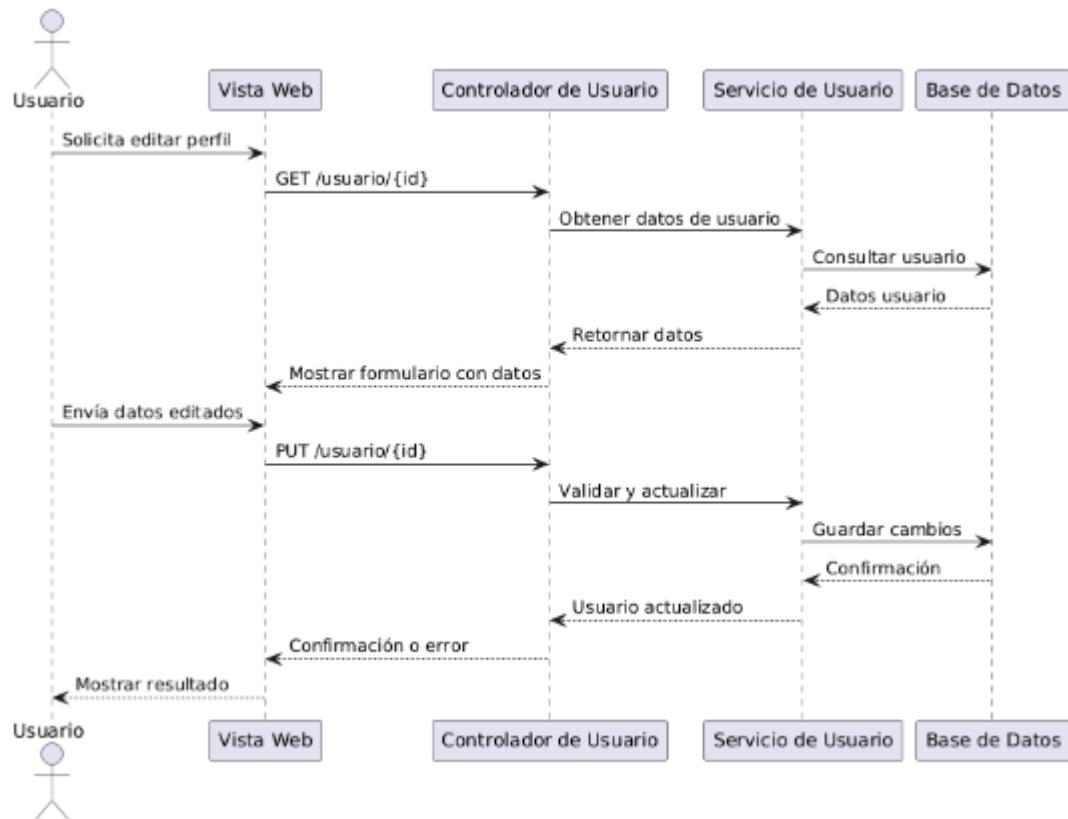
2.1.3 Diseño de la lógica de negocio

2.1.3.1 Diagramas de secuencia

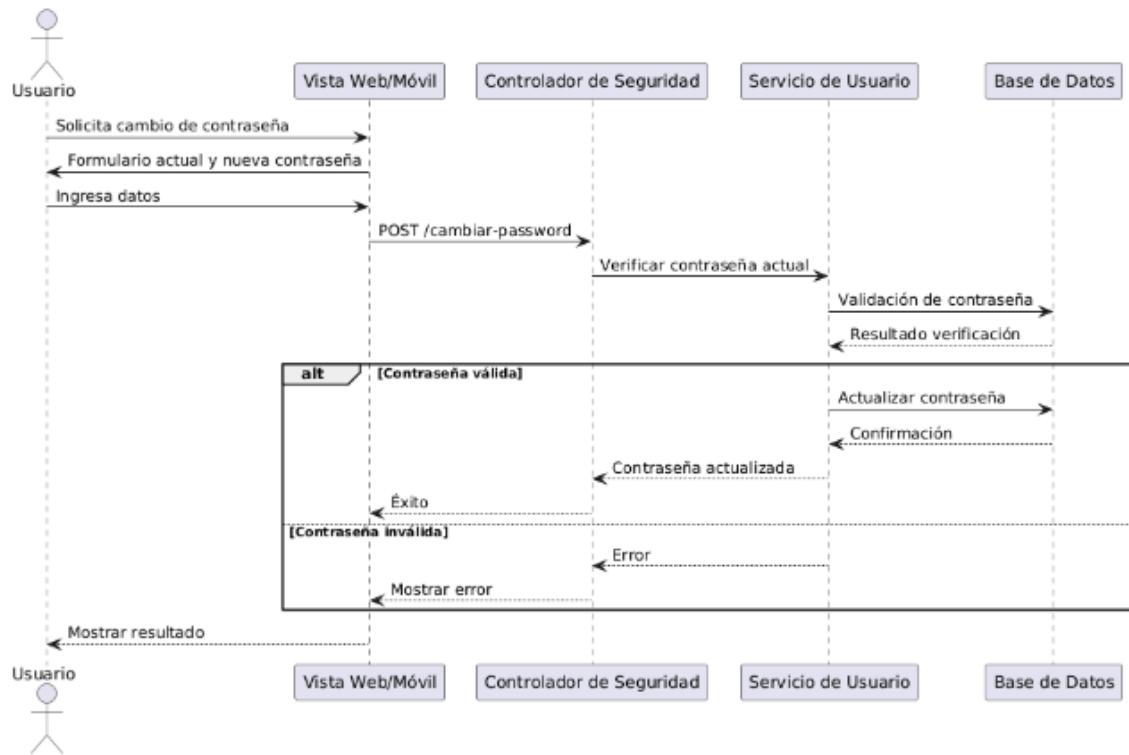
2.1.3.1.1 CU01: Registro de usuarios



2.1.3.1.2 CU03: Edición de información de usuario

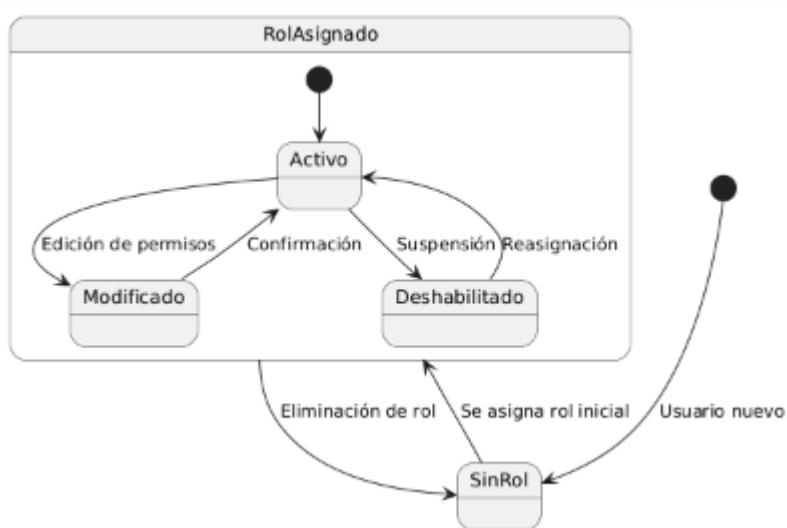


2.1.3.1.3 CU04: Cambio de contraseña



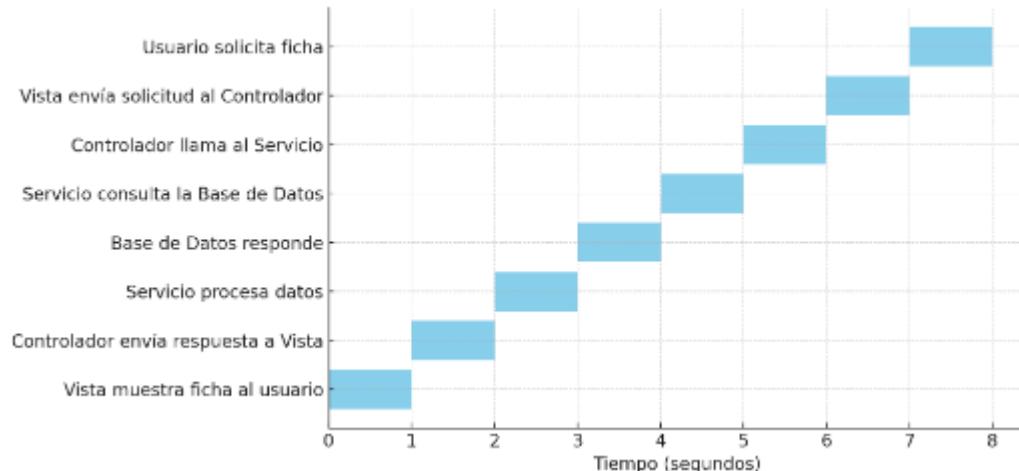
2.1.3.2 Diagramas de estado

2.1.3.2.1 CU02: Gestión de roles y permisos



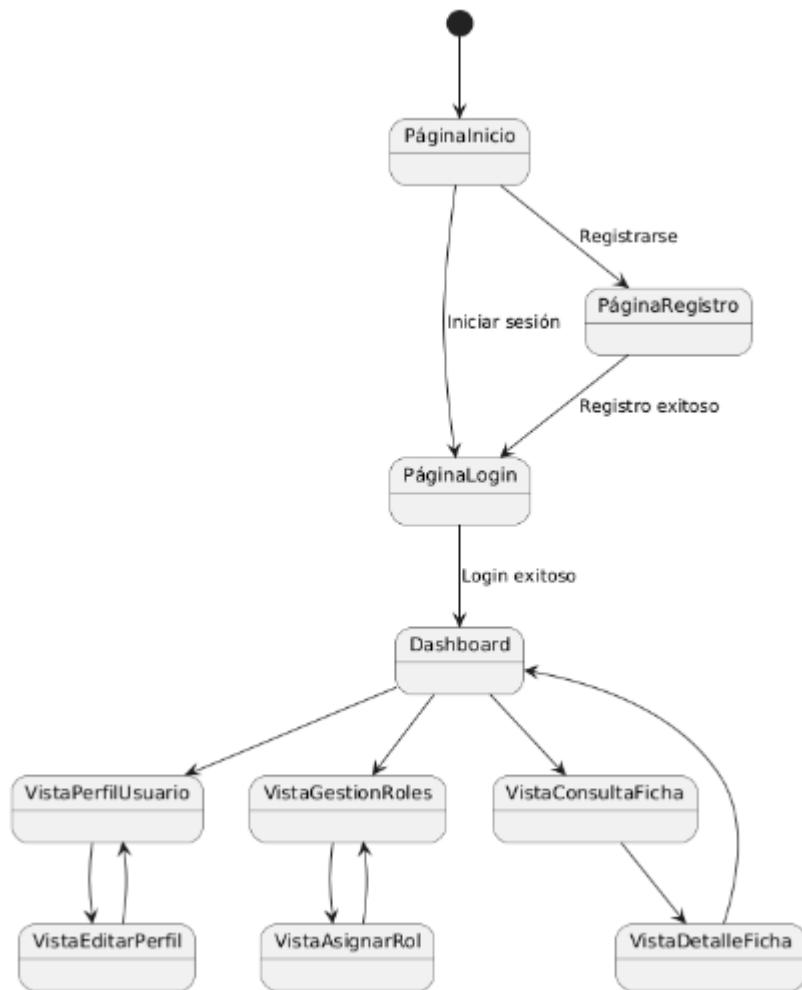
2.1.3.3 Diagrama de tiempo

2.1.3.3.1 CU05: Consulta de ficha individual del estudiante



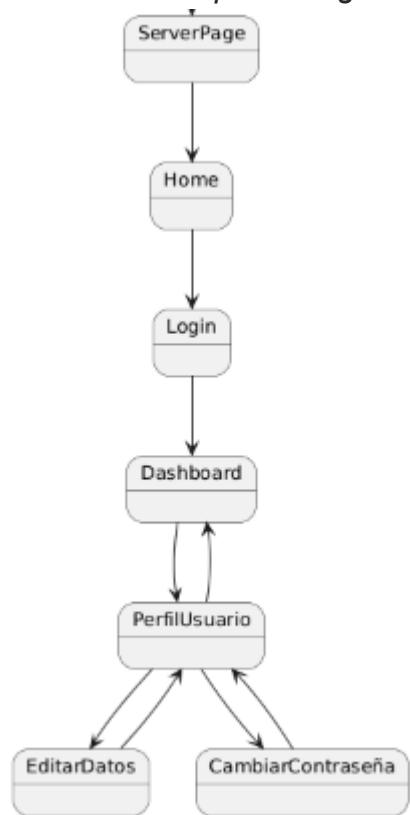
2.1.3.4 Diagrama de navegación

2.1.3.4.1 Pagina principal

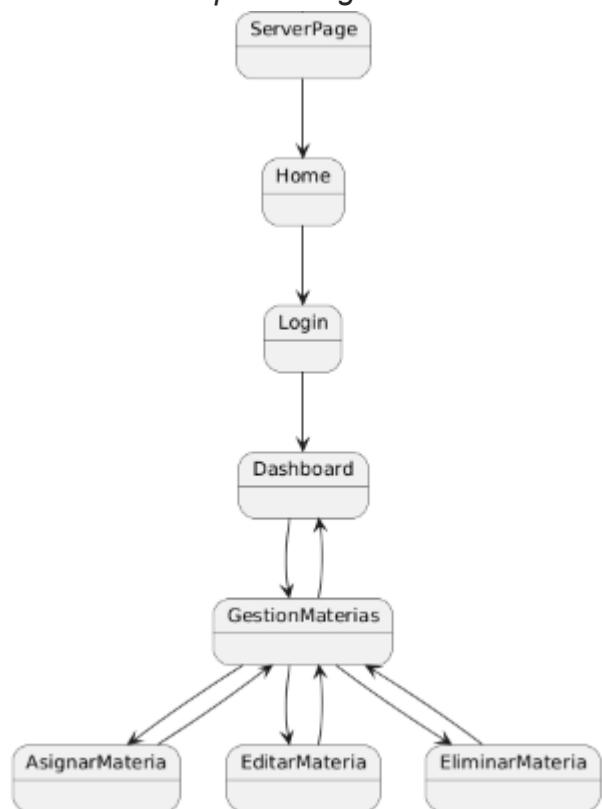


2.1.3.4.2 Por subsistemas

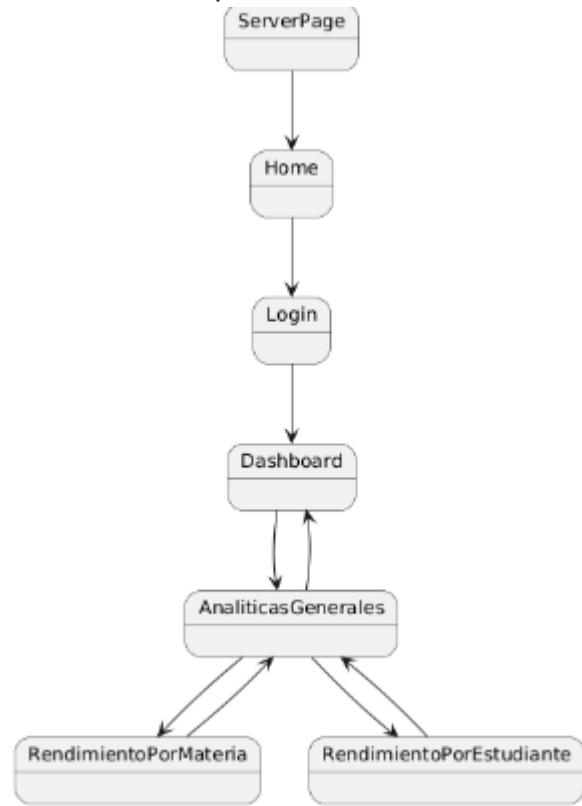
2.1.3.4.2.1 Paquete de gestión de usuarios



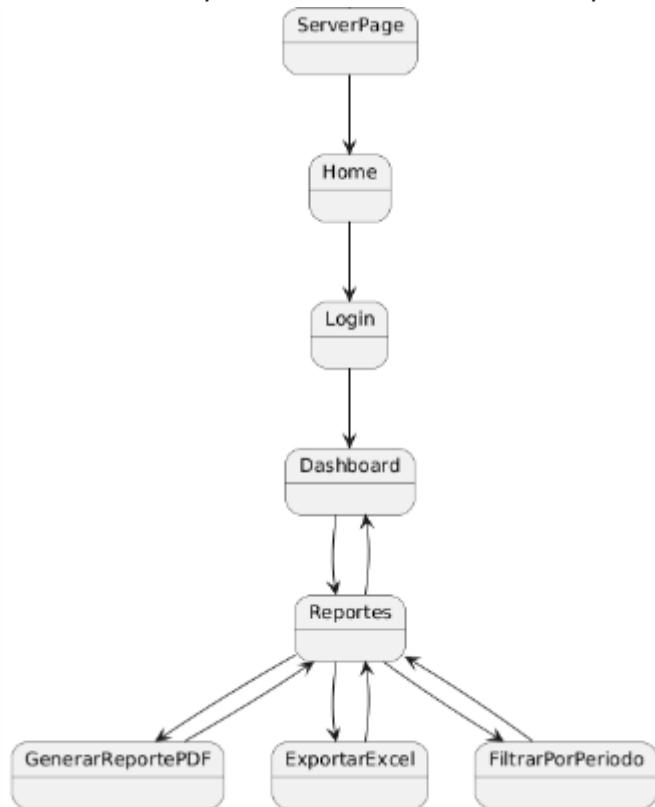
2.1.3.4.2.2 Paquete de gestión académica



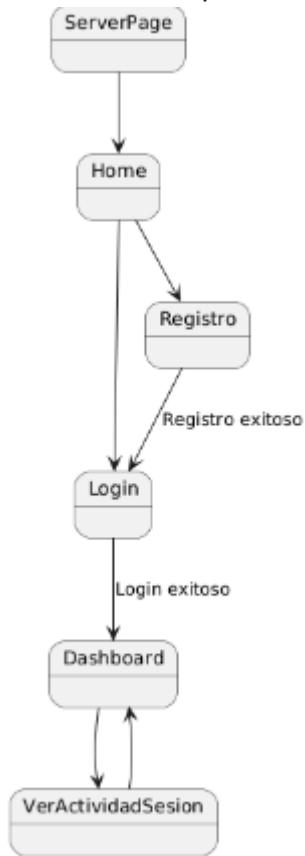
2.1.3.4.2.3 Paquete de analíticas académicas



2.1.3.4.2.4 Paquete de visualización de reportes



2.1.3.4.2.5 Paquete de autenticación y seguridad



2.2 Implementación

2.2.1. Componentes y artefactos generados

Componentes Backend (Python - Django)

Módulos principales desarrollados:

- **usuarios_app**: Módulo central para gestión de usuarios del sistema
 - models.py: Define las entidades Usuario, Rol, Permiso
 - views.py: Controladores para registro, edición y gestión de contraseñas
 - serializers.py: Serialización de datos para API REST
 - urls.py: Rutas de acceso para cada endpoint
- **permisos_app**: Gestión de roles y permisos
 - models.py: Modelos para Rol, Permiso y relaciones
 - views.py: Lógica de asignación y verificación de permisos
 - permissions.py: Decoradores personalizados para control de acceso
- **estudiantes_app**: Gestión de fichas estudiantiles

- models.py: Modelo de Estudiante con sus atributos específicos
- views.py: Controlador para visualización de ficha individual
- serializers.py: Formateo de datos para presentación

Componentes Frontend Web (Angular)

Módulos desarrollados:

- **AuthModule:** Módulo de autenticación
 - login.component.ts: Formulario de inicio de sesión
 - register.component.ts: Formulario de registro de usuarios
 - password-management.component.ts: Gestión y recuperación de contraseña
 - auth.service.ts: Servicio de autenticación y manejo de tokens
- **UserManagementModule:** Gestión de usuarios
 - user-list.component.ts: Lista de usuarios registrados
 - user-edit.component.ts: Formulario de edición de información de usuario
 - role-assignment.component.ts: Asignación de roles a usuarios
 - user.service.ts: Servicio de gestión de usuarios
- **RolePermissionsModule:** Administración de roles
 - role-list.component.ts: Lista de roles existentes
 - role-edit.component.ts: Edición y creación de roles
 - permission-assignment.component.ts: Asignación de permisos a roles
 - role.service.ts: Servicio para gestión de roles y permisos
- **StudentModule:** Ficha estudiantil
 - student-profile.component.ts: Visualización de ficha individual de estudiante
 - student-search.component.ts: Búsqueda de estudiantes por diversos criterios
 - student.service.ts: Servicio para obtener datos de estudiantes

Componentes Frontend Móvil (Flutter)

Módulos desarrollados:

- auth_module: Autenticación móvil y gestión de contraseñas
- user_profile: Visualización y edición de perfil
- student_view: Visualización de ficha estudiantil en móvil

Artefactos generados

1. **Base de datos PostgreSQL** implementada según el esquema definido, con tablas para:
 - usuarios, roles, permisos, usuario_roles, rol_permisos
 - estudiantes, profesores, tokens_recuperacion
2. **API REST** documentada con Swagger:
 - Endpoints de autenticación: /api/auth/login, /api/auth/register
 - Endpoints de gestión de usuarios: /api/usuarios/
 - Endpoints de roles: /api/roles/
 - Endpoints de fichas: /api/estudiantes/{id}
3. **Interfaces de usuario** (mockups implementados):
 - Formularios de registro y login
 - Panel de gestión de usuarios
 - Vista de roles y permisos
 - Ficha estudiantil
4. **Scripts de migración** para la estructura de base de datos
5. **Documentación técnica** generada con Spectacular para Python y Compendio para Angular

2.3 Pruebas

2.3.1. Plan de pruebas (criterios de aceptación)

CU01: Registro de Usuarios

Resultado:

- Registro de usuarios (administrador, docente, alumno): Completado con éxito.
- Validación de campos obligatorios según tipo de usuario: Funcionó correctamente.
- Validación de correo electrónico único: Verificada.
- Asignación automática de rol según tipo: Implementada correctamente.

- Almacenamiento seguro de contraseñas con hash: Verificado. **Estado:** Aprobado.

CU02: Gestión de Roles

Resultado:

- Creación, edición y visualización de roles: Completado con éxito.
- Asignación de permisos a roles: Funcionó correctamente.
- Validación de nombre único para cada rol: Verificada.
- Restricción de acceso solo a administradores: Implementada.
- Auditoría de cambios en roles: Registro implementado. **Estado:** Aprobado.

CU03: Edición de Usuarios

Resultado:

- Modificación de datos personales de usuario: Completado con éxito.
- Validación de formato de correo y campos obligatorios: Funcionó correctamente.
- Actualización de fecha de modificación: Verificada.
- Restricción de edición según permisos: Implementada correctamente.
- Protección contra edición de usuarios administrativos por usuarios sin permisos: Verificada. **Estado:** Aprobado.

CU04: Gestión de Contraseña

Resultado:

- Cambio de contraseña con validación de contraseña actual: Completado con éxito.
- Validación de políticas de seguridad en nueva contraseña: Funciona correctamente.
- Proceso de recuperación por correo electrónico: Verificado.
- Tokens de un solo uso para restablecimiento: Implementados.
- Expiración de tokens de recuperación: Verificada. **Estado:** Aprobado.

CU05: Consulta de Ficha Estudiantil

Resultado:

- Visualización de datos básicos del estudiante: Completado con éxito.

- Restricción de acceso según permisos (solo docentes y administradores): Verificada.
- Presentación de información académica organizada: Implementada.
- Búsqueda de estudiantes por nombre o código: Funcionó correctamente.
- Interfaz responsive para acceso desde diferentes dispositivos: Verificada. **Estado:** Aprobado.

2.3.2 Reporte de prueba

Historia de Usuario: Registro de Nuevos Usuarios

Criterios de Aceptación:

- El sistema debe permitir el registro de diferentes tipos de usuario (administrador, docente, alumno)
- Debe existir validación de campos obligatorios para cada tipo
- El correo electrónico debe ser único en el sistema
- La contraseña debe cumplir con criterios mínimos de seguridad

Datos de Prueba:

- Tipo de usuario: Docente
- Nombre usuario: jmartinez
- Correo: jmartinez@example.com.
- Nombres: Juan Carlos
- Apellidos: Martínez López
- Contraseña: Edu\$2025
- Código Docente: DOC-2025-103

Procedimientos de Prueba:

1. Acceder al formulario de registro desde la página principal
2. Seleccionar tipo de usuario "Docente"
3. Completar todos los campos solicitados
4. Verificar que aparecen los campos específicos para docente
5. Hacer clic en "Registrar Usuario"

Resultado Esperado:

- Mensaje "Usuario registrado exitosamente"
- Correo de confirmación enviado con credenciales
- El usuario puede iniciar sesión y tiene asignado el rol de docente
- Verificado en base de datos la creación correcta del registro

Historia de Usuario: Asignación de Roles y Permisos

Criterios de Aceptación:

- Los administradores deben poder crear y modificar roles
- Se puede asignar múltiples permisos a cada rol
- La interfaz debe ser intuitiva para gestionar permisos
- Los cambios se aplican inmediatamente

Datos de Prueba:

- Rol: Tutor Académico
- Descripción: Profesor con acceso a fichas e informes de rendimiento de grupos específicos
- Permisos: Ver Estudiantes, Ver Calificaciones, Editar Observaciones

Procedimientos de Prueba:

1. Iniciar sesión como administrador
2. Acceder a módulo "Gestión de Roles"
3. Crear nuevo rol "Tutor Académico"
4. Asignar los permisos seleccionados mediante checkboxes
5. Guardar cambios
6. Verificar que el rol aparece en la lista

Resultado Esperado:

- Rol creado correctamente con los permisos asignados
- El rol está disponible para asignar a usuarios
- Al asignar el rol a un profesor, este obtiene los permisos definidos

Historia de Usuario: Edición de Perfiles de Usuario

Criterios de Aceptación:

- Los usuarios deben poder editar su información personal

- Los administradores pueden editar cualquier usuario
- El sistema debe validar los formatos de datos (email, teléfono)
- Se debe registrar fecha de modificación

Datos de Prueba:

- Usuario a editar: cgonzalez (estudiante)
- Nuevos datos:
 - Teléfono: 69558702
 - Dirección: Av. Virgen de Cotoca 345, Santa Cruz
 - Correo alternativo: carlosg@email.com

Procedimientos de Prueba:

1. Iniciar sesión como administrador
2. Buscar usuario "cgonzalez" en lista de usuarios
3. Hacer clic en botón "Editar"
4. Modificar los campos mencionados
5. Guardar cambios

Resultado Esperado:

- Mensaje "Información actualizada correctamente"
- Los nuevos datos se reflejan en la ficha del estudiante
- En el registro de auditoría aparece el cambio con fecha y usuario realizado

Historia de Usuario: Cambio de Contraseña

Criterios de Aceptación:

- El usuario debe ingresar su contraseña actual para validación
- La nueva contraseña debe cumplir requisitos de seguridad
- El sistema debe mostrar indicador de fortaleza de contraseña
- Se debe confirmar la nueva contraseña para evitar errores

Datos de Prueba:

- Usuario: mlopez (profesora)
- Contraseña actual: Inicial#2025
- Nueva contraseña: Pass123

- Confirmar contraseña: Pass123

Procedimientos de Prueba:

1. Iniciar sesión como mlopez
2. Acceder a "Mi Perfil" > "Seguridad"
3. Ingresar los datos de contraseña actual y nueva
4. Verificar el indicador de fortaleza (debe mostrar "Fuerte")
5. Hacer clic en "Cambiar Contraseña"

Resultado Esperado:

- Mensaje de confirmación: "Su contraseña ha sido actualizada"
- Al cerrar sesión, solo se puede iniciar con la nueva contraseña
- Se registra la fecha de cambio de contraseña en el sistema

Historia de Usuario: Consulta de Ficha Estudiantil

Criterios de Aceptación:

- Docentes y administrativos pueden buscar estudiantes
- La ficha muestra datos personales y académicos
- Se debe presentar la información de manera organizada y clara
- La interfaz debe ser responsive para diferentes dispositivos

Datos de Prueba:

- Criterio de búsqueda: "EST-2023-0103" (código de estudiante)
- Estudiante: Carlos González

Procedimientos de Prueba:

1. Iniciar sesión como profesor (mlopez)
2. Ir a sección "Estudiantes"
3. Ingresar el código en el campo de búsqueda
4. Presionar botón "Buscar"
5. Seleccionar al estudiante de los resultados

Resultado Esperado:

- La ficha muestra nombre completo, código, grado y datos de contacto
- Se visualiza el rendimiento académico (calificaciones, asistencia)

- La información se agrupa en secciones claramente identificables
- Todas las pestañas (Información Personal, Académica, etc.) muestran datos correctos

Observación: La función de predicción de rendimiento aún no está implementada en este sprint, pero la estructura de la ficha ya contempla el espacio para esta información.

3. Daily Scrum (o Scrum diario)

Daily Scrum - Sprint 1

Daily Scrum 13/05/2025

Miembro	¿Qué hice ayer?	¿Qué voy a hacer hoy?	Impedimentos
Cuéllar	Realicé el diseño conceptual de la base de datos	Comenzaré con el backend para el registro de usuarios	Ninguno
Méndez	Revisé el modelo de usuarios y roles	Empezaré a diseñar los endpoints de roles y permisos	Ninguno

Daily Scrum 14/05/2025

Miembro	¿Qué hice ayer?	¿Qué voy a hacer hoy?	Impedimentos
Cuéllar	Avancé en la lógica del registro y edición de usuarios	Diseñar el formulario web para el registro de usuarios	Ninguno
Méndez	Configuré los endpoints para gestión de roles y permisos	Diseñar la interfaz web para gestionar roles	Problemas menores con rutas dinámicas

Daily Scrum 15/05/2025

Miembro	¿Qué hice ayer?	¿Qué voy a hacer hoy?	Impedimentos
Cuéllar	Implementé el formulario web de usuarios	Avanzar con el formulario móvil de usuarios	Requiere revisión en la navegación móvil
Méndez	Maqueté la gestión de roles web	Iniciar backend de ficha individual del estudiante	Ninguno

Daily Scrum 16/05/2025

Miembro	¿Qué hice ayer?	¿Qué voy a hacer hoy?	Impedimentos
Cuéllar	Estructuré el registro de usuarios en versión móvil	Trabajar en la lógica de cambio de contraseña	Ninguno
Méndez	Creé los endpoints de consulta de ficha de estudiante	Crear la interfaz web para la ficha de estudiante	Necesita confirmar campos desde base de datos

Daily Scrum 17/05/2025

Miembro	¿Qué hice ayer?	¿Qué voy a hacer hoy?	Impedimentos
Cuéllar	Configuré la lógica de cambio de contraseña	Probar flujo completo de cambio de contraseña	Coordinación con front móvil
Méndez	Completé vista web de ficha de estudiante	Desarrollar ficha para versión móvil	Ninguno

Daily Scrum 18/05/2025

Miembro	¿Qué hice ayer?	¿Qué voy a hacer hoy?	Impedimentos
Cuéllar	Realicé pruebas unitarias en gestión de usuarios	Preparar entrega del Sprint 1 y documentación	Ninguno
Méndez	Probé flujo completo de roles y ficha de estudiante	Armar presentación y preparar revisión de Sprint	Ninguno

4. Sprint Review

Objetivos del Sprint

Desarrollar e implementar las funcionalidades básicas del sistema *Aula Inteligente*, enfocadas en la gestión de usuarios, roles y la consulta de fichas académicas desde la web y la aplicación móvil. Se buscó garantizar el correcto inicio de sesión, la creación de usuarios con diferentes permisos, el cambio de contraseña y la visualización individual de datos académicos por parte de los estudiantes.

Participantes

Nombre	Rol
Cuéllar Flores Victor Hugo	Product Owner
Méndez López Sebastián	Scrum Master

Presentación del incremento

Función presentada	Retroalimentación
Pruebas y validación de la base de datos, con integración a los endpoints	Funcionalidad terminada
Implementación de migraciones y modelo de datos (usuarios, roles, fichas)	Funcionalidad terminada
Implementación de la lógica de autenticación (backend y frontend)	Funcionalidad terminada
Creación de vistas para el formulario de inicio de sesión y registro (WEB)	Funcionalidad terminada
Creación de vistas para el formulario de inicio de sesión y registro (MÓVIL)	Funcionalidad terminada
Pruebas y validación de autenticación y registro (web y móvil)	Funcionalidad terminada
Implementación del endpoint para gestionar usuarios (listar, crear, editar)	Funcionalidad terminada
Creación de vistas para gestión de usuarios (listar, crear, editar)	Funcionalidad terminada

Función presentada	Retroalimentación
Implementación del endpoint para gestionar roles (listar, crear, editar, eliminar)	Funcionalidad terminada
Creación de vistas para gestión de roles en el frontend (solo WEB)	Funcionalidad terminada
Implementación del cambio de contraseña (endpoint + frontend web y móvil)	Funcionalidad terminada
Pruebas y validación del cambio de contraseña	Funcionalidad terminada
Implementación del endpoint de ficha académica del estudiante	Funcionalidad terminada
Visualización de la ficha del estudiante en la app móvil y web	Funcionalidad terminada

5. Sprint Retrospective

Fecha: 20/05/2025

Facilitador: Cuéllar Flores Victor Hugo

Objetivo: Conclusión del Sprint 1

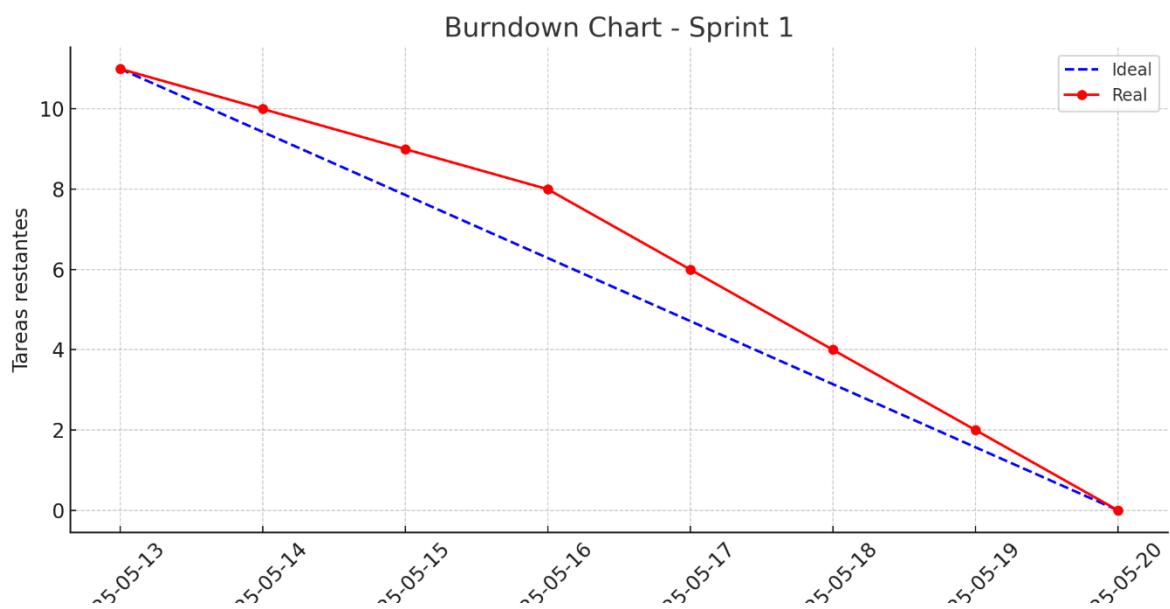
Nombres de asistentes:

- Cuéllar Flores Victor Hugo
- Méndez López Sebastián

Tema	¿Qué salió bien?	¿Qué no salió bien?	¿Qué haremos de manera diferente?
Coordinación del grupo	Buena sinergia en la asignación y entrega de tareas	En algunos días se acumularon tareas para un solo miembro	Planificar con mayor anticipación los tiempos compartidos
Comunicación en el equipo	Comunicación constante vía grupo de trabajo	Falta de reuniones intermedias presenciales	Realizar al menos 2 encuentros durante el Sprint

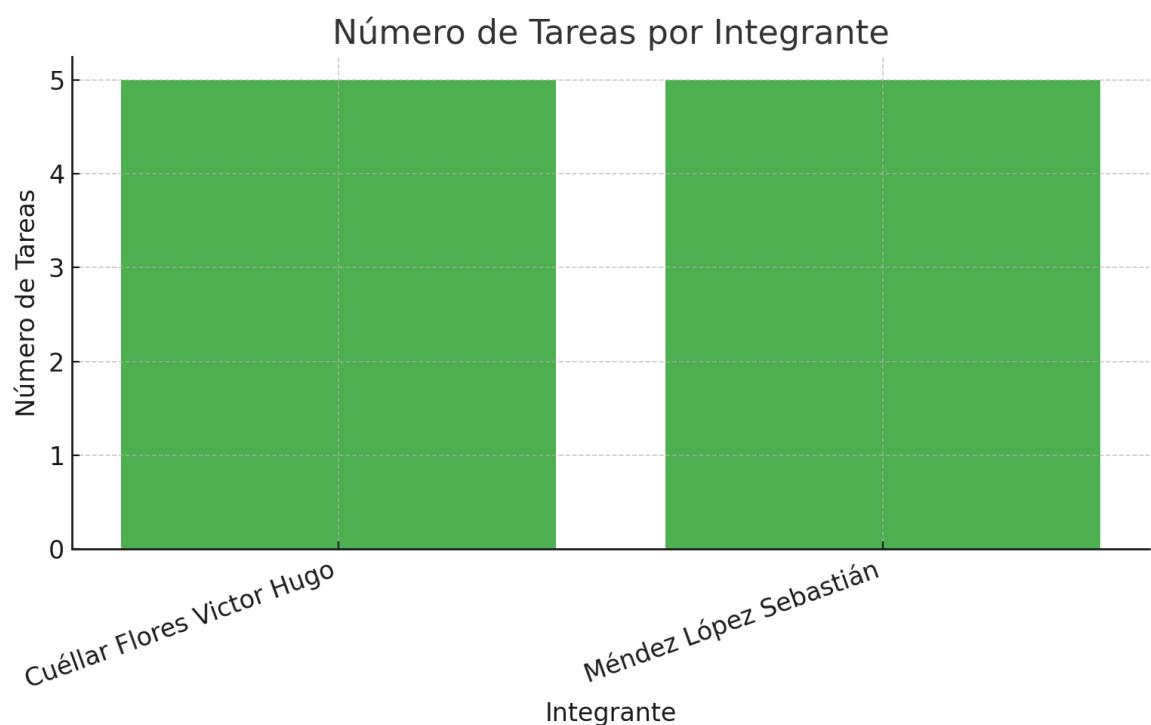
Tema	¿Qué salió bien?	¿Qué no salió bien?	¿Qué haremos de manera diferente?
Asignación de roles	Cada miembro asumió su responsabilidad claramente	No se definieron líderes para testing	Asignar responsable de validaciones por historia de usuario
Gestión del tiempo	La mayoría de entregas fueron a tiempo	Algunos endpoints se integraron el último día	Definir fechas límite internas para cada funcionalidad
Aprendizaje técnico	Se entendió bien la lógica de SCRUM y estructura base	Costó implementar cambios simultáneos en frontend y backend	Dividir mejor tareas entre backend y frontend desde el inicio

6. Burndown y BurnUp(Grafica de tareas y Datos de tareas)

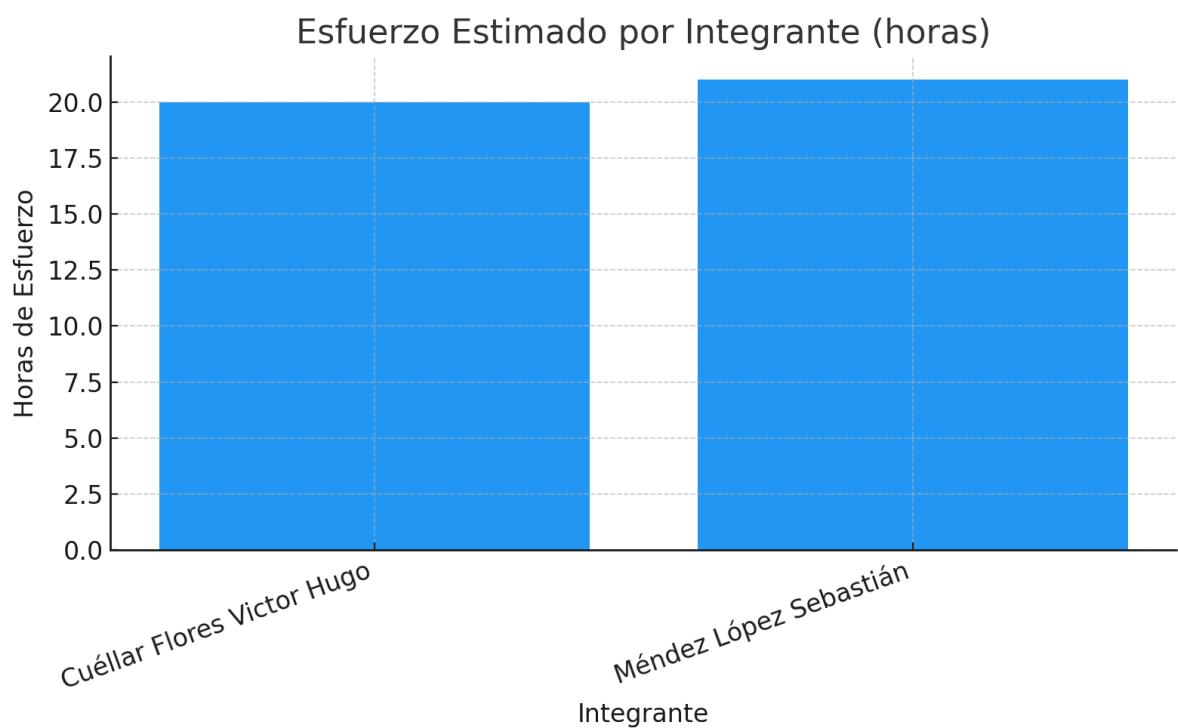


7. Grafica de esfuerzo y Datos de esfuerzo

7.1. Gráfico de esfuerzo



7.2. Gráfico de datos de esfuerzo



8. Scrum TaskBoard (Backlog, to do, doing, done)

	Type	Key	Summary	Status	Assignee
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-1	Diseño de base de datos para gestión de usuarios y roles	DONE	Victor Hugo Cuéllar
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-2	Backend: Registro y edición de usuarios	DONE	Victor Hugo Cuéllar
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-3	Frontend Web: Formulario de registro y edición de usuari...	DONE	Victor Hugo Cuéllar
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-4	Frontend Móvil: Formulario de registro y edición de usuari...	DONE	Victor Hugo Cuéllar
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-5	Backend: Asignación de roles y permisos	DONE	sebamendex11
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-6	Frontend Web: Gestión de roles y permisos	DONE	sebamendex11
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-7	Frontend Móvil: Visualización de roles (solo lectura)	DONE	sebamendex11
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-8	Backend: Consulta de ficha individual del estudiante	DONE	sebamendex11
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-9	Frontend Web: Vista de ficha individual del estudiante	DONE	sebamendex11
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-10	Frontend Móvil: Vista de ficha individual del estudiante	DONE	sebamendex11
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-11	Backend: Gestión de cambio de contraseña	DONE	Victor Hugo Cuéllar
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-12	Frontend Web: Formulario de cambio de contraseña	DONE	Victor Hugo Cuéllar
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-13	Frontend Móvil: Formulario de cambio de contraseña	DONE	Victor Hugo Cuéllar
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-14	Pruebas unitarias y de integración (CU01-CU03)	DONE	Victor Hugo Cuéllar
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-15	Pruebas unitarias y de integración (CU10-CU16)	DONE	sebamendex11
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-16	Revisión general y documentación Sprint 1	DONE	
<input type="checkbox"/>	<input checked="" type="checkbox"/>	KAN-17	Retrospectiva Sprint 1	DONE	

+ Create

LINKS & QR – Sprint 1

Link repo backend: <https://github.com/sebastianmlz/ficct-school-backend.git>



Link repo frontend web: <https://github.com/sebastianmlz/aula-inteligente-Frontend.git>



Link de repo frontend movil: <https://github.com/sebastianmlz/aula-inteligente-Mobile.git>

