

**UNIVERSIDAD PRIVADA DOMINGO SAVIO**  
**FACULTAD DE INGENIERIA**  
**INGENIERIA DE SISTEMAS**



**Proyecto investigativo: “Sistema de juegos interactivos para estudiantes de Primaria de Primer a Sexto curso”**

**Autores:**

Aron Jairo Arana Valdivia  
Diego Villarroel Sandy  
Joel Benítez Paco  
José Vargas Camacho  
Oscar Flores Herrera

**Docente:** Ing. Alejandra Gómez Mendoza

**Fecha de entrega:** 21/05/2025

## INDICE

1. Introducción .....	1
1.1. Antecedentes .....	1
1.2. Planteamiento del Problema.....	1
1.3. Objetivos .....	2
1.3.1. Objetivo general .....	2
1.3.2. Objetivos específicos .....	2
2. MARCO CONCEPTUAL.....	3
2.1. Metodología Scrum.....	3
2.2. Trello .....	3
2.3. Python .....	3
2.4. Flask.....	3
2.5. PostgreSQL .....	4
2.6. Git y GitHub .....	4
3. MARCO PRACTICO.....	4
3.1. Plan de Desarrollo de software .....	4
3.2. Preparación del entorno de trabajo .....	4
3.2.1. Herramientas de desarrollo utilizadas .....	4
3.2.2. Estructura del repositorio en GitHub .....	5
3.2.3. Configuración inicial .....	5
3.3. Desarrollo del Sistema .....	6
3.3.1. Sprint 0 .....	6
3.3.2. Sprint 1 .....	14
3.3.3 Sprint 2: .....	17
3.3.4 Sprint 3: .....	19
Bibliografía .....	21

## **1. Introducción**

En el presente documento se desarrollará el proyecto formativo de desarrollo de una aplicación con juegos interactivos para estudiantes de 1° a 6° de primaria. Esto responde a una necesidad planteada por una ONG que necesita un software de control de desempeño en su alumnado frente a diferentes juegos interactivos.

### **1.1. Antecedentes**

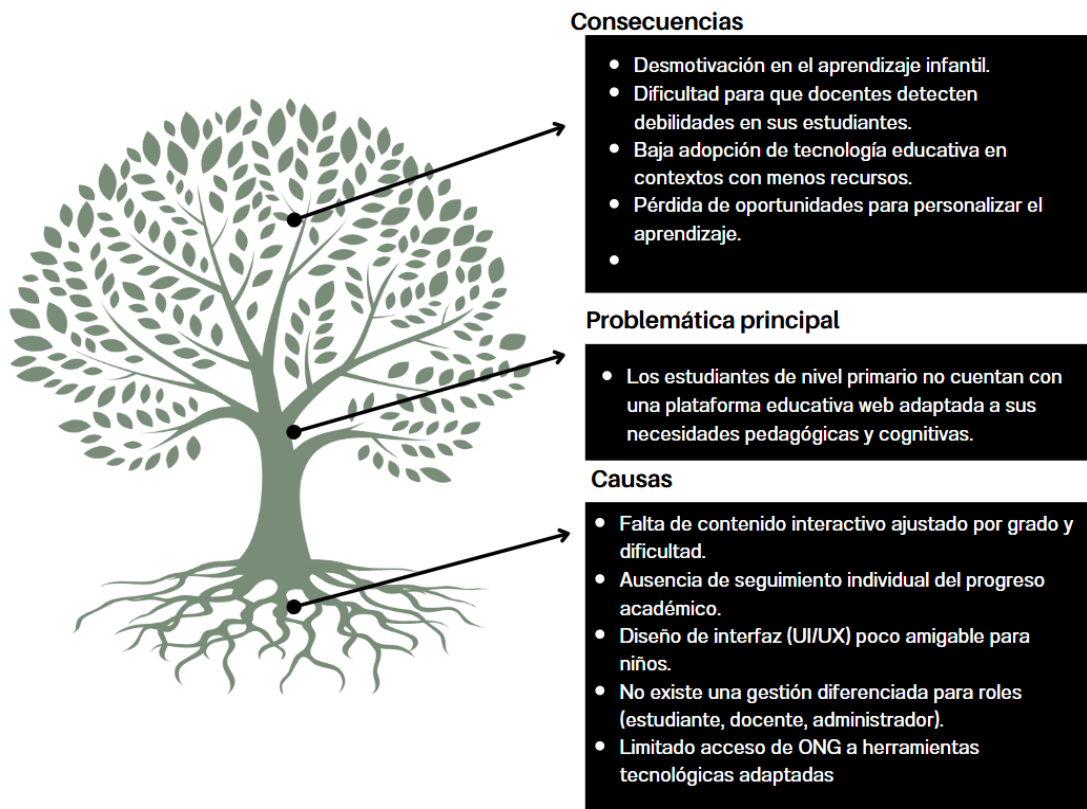
La ONG “Educo” es una organización dedicada a mejorar las condiciones educativas de niños en más de 18 países. Desde hace 30 años concentran esfuerzos en niños niñas y adolescentes vulnerables que no poseen las mismas oportunidades.

De acuerdo a su sitio oficial, esta Organización trabaja en 285 escuelas en los departamentos de La Paz, Chuquisaca, Tarija, Santa Cruz, Cochabamba, Pando y Beni. Su oficina central se encuentra en la ciudad de La Paz, específicamente en la Avenida Los Álamos N.º 52.

Dicha Organización enfrenta un problema de falta de integración de la educación que ofrece en las diferentes escuelas con el constante avance de la tecnología, siendo incapaz de centralizar los datos del desempeño de los estudiantes en juegos interactivos para su posterior análisis y/o generación de reportes.

### **1.2. Planteamiento del Problema**

En la actualidad, muchos niños y niñas de primaria no pueden disfrutar de una plataforma educativa en la red que sea a la vez fácil de usar y adecuada para su edad. Las plataformas que existen, no siempre tienen actividades con las que se puedan aprender mejor, no permiten a los profesores y las profesoras seguir cómo avanza cada uno de los estudiantes de forma clara, y muchas veces no están pensadas para que los niños y las niñas las entiendan. Esto hace que aprender sea más difícil y menos estimulante para los niños y las niñas. Por eso, este proyecto intenta crear una plataforma educativa que está orientada especialmente a niños y niñas de 1er a 6to de primaria.



### 1.3. Objetivos

#### 1.3.1. Objetivo general

Diseñar y desarrollar una plataforma educativa web como solución tecnológica para la educación primaria, orientada a mejorar el aprendizaje de los estudiantes de 1.º a 6.º grado mediante actividades interactivas y un sistema de seguimiento personalizado, brindando a los docentes herramientas efectivas para apoyar su labor educativa.

#### 1.3.2. Objetivos específicos

1. Desarrollar un sistema de autenticación con acceso diferenciado para estudiantes y docentes, incluyendo la gestión de roles y permisos.
2. Diseñar e implementar actividades educativas interactivas organizadas por grado escolar, que fomenten el aprendizaje dinámico en los estudiantes de 1.º a 6.º de primaria.
3. Incorporar un sistema de seguimiento académico que permita a los docentes monitorear el progreso y las calificaciones de sus estudiantes.

4. Garantizar la usabilidad y accesibilidad de la plataforma mediante la realización de pruebas funcionales y de experiencia de usuario., incluyendo diagramas UML, historias de usuario, product backlog y sprints

## **2. MARCO CONCEPTUAL**

### **2.1. Metodología Scrum**

Scrum es un marco de trabajo ágil para la gestión de proyectos complejos, centrado en la entrega iterativa e incremental de productos.

*"Scrum se basa en principios de transparencia, inspección y adaptación, con roles definidos como Scrum Master, Product Owner y el equipo de desarrollo".* (Sutherland, 2017).

Esta metodología se adapta especialmente al desarrollo de software, permitiendo flexibilidad y mejora continua.

### **2.2. Trello**

Trello es una plataforma visual basada en el sistema *Kanban*, que facilita la organización de tareas mediante tableros, listas y tarjetas. Su enfoque colaborativo lo hace ideal para equipos ágiles. *"Trello es una plataforma pensada para la gestión de tareas, tanto individuales como desarrolladas en equipo. Trello se puede utilizar tanto en la nube como a través de Apps que permiten editar y gestionar los recursos offline, sincronizándose posteriormente en todos los dispositivos en los que esté activada la misma cuenta."* (Cuartas, 2019)

### **2.3. Python**

Python es un lenguaje de programación interpretado, multiparadigma y de alto nivel, ampliamente utilizado en desarrollo web, análisis de datos e inteligencia artificial. (Rossum, 2025) destaca que *"Python prioriza la legibilidad del código y su sintaxis minimalista, lo que acelera el proceso de desarrollo"*.

### **2.4. Flask**

Flask es un *microframework* de Python diseñado para construir aplicaciones web ligeras y modulares. Su flexibilidad y escalabilidad lo hacen adecuado para proyectos pequeños y medianos.

*"Flask ofrece las herramientas esenciales sin imponer estructuras rígidas, permitiendo integraciones personalizadas con bases de datos o APIs".* (Grinberg, 2018)

## 2.5. PostgreSQL

*“PostgreSQL es un sistema de gestión de bases de datos objeto-relacional (ORDBMS) basado en POSTGRES, versión 4.2, desarrollado en el Departamento de Informática de la Universidad de California en Berkeley.”* (Group, 2025)

PostgreSQL es un sistema de gestión de bases de datos relacionales (SGBDR). Esta tecnología se caracteriza por ser open source y por su facilidad de implementación tanto en un proyecto local como al momento de hacer el deployment.

## 2.6. Git y GitHub

Git es un sistema de control de versiones distribuido que permite rastrear cambios en el código fuente. GitHub, basado en Git, es una plataforma colaborativa para alojar proyectos. Chacon y Straub (2014) señalan que *“Git facilita la gestión de ramas (branches) y la resolución de conflictos, esenciales en equipos de desarrollo ágiles”*.

# 3. MARCO PRACTICO

## 3.1. Plan de Desarrollo de software

Para el presente proyecto se desarrollo un plan de desarrollo con el fin de determinar tanto la metodología como las herramientas necesarias para su implementación.

De igual forma se define el calendario de trabajo describiendo las fechas para cada sprint. Para observar dicho plan de desarrollo referirse al Anexo 1.

## 3.2. Preparación del entorno de trabajo

Una vez determinado el plan de desarrollo se procede a la implementación del proyecto, para lo cual es necesario detallar lo necesario para la preparación del entorno de trabajo.

### 3.2.1. Herramientas de desarrollo utilizadas

En este apartado se detallará las herramientas utilizadas para el desarrollo del proyecto. En la siguiente tabla (Ver Tabla N°1) se lista las herramientas y su uso dentro de la implementación.

**Tabla N°1: Herramientas de desarrollo utilizada**

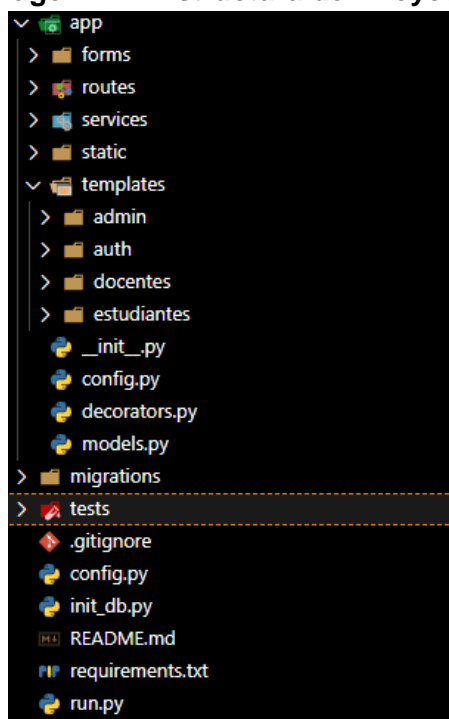
Herramienta	Propósito
Visual Studio Code	Editor de código principal
PostgreSQL	Motor de base de datos (Interfaz Opcional)
Git + Github	Control de versiones y colaboración de trabajo remoto.
Trello	Gestión de tareas y seguimiento Scrum

*Fuente: Elaboración Propia (2025)*

### 3.2.2. Estructura del repositorio en GitHub

Posterior definición de las herramientas y tecnologías a utilizar se define la estructura bajo la cual se trabajará. En este y, de acuerdo a la necesidad del Framework elegido, se trabaja con rutas y templates donde en el primero se define la configuración de las rutas y sus respectivas acciones, y en los templates se desarrollan las vistas. Junto a lo previo se muestra a detalle la estructura en la siguiente imagen (Ver Imagen N°1):

**Imagen N°1: Estructura del Proyecto**



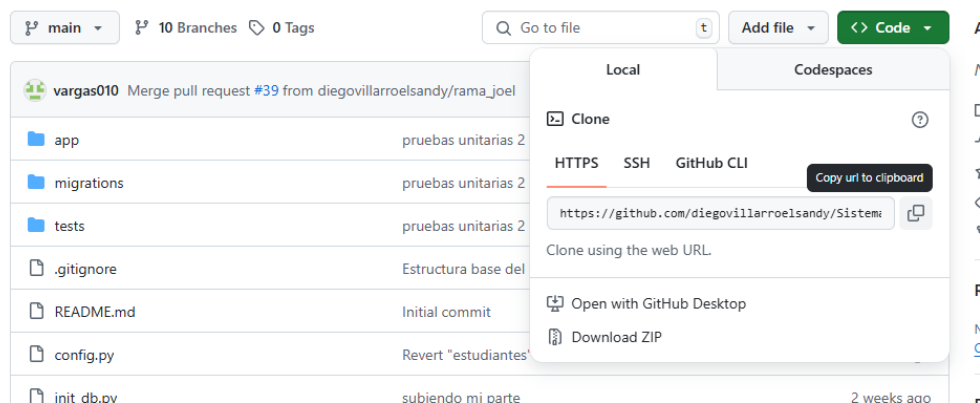
*Fuente: Elaboración Propia*

### 3.2.3. Configuración inicial

Para el trabajo colaborativo en remoto es necesario que cada integrante del equipo cuente con el proyecto en sus máquinas, para lo cual es necesario listar los pasos para poder correr el proyecto en el ambiente local.

1. Clonar el repositorio en el ambiente local. – El integrante deberá clonar el proyecto desde el repositorio en Github usando el comando git clone y la url obtenida desde el proyecto.

**Imagen N°2: Clonado de Proyecto**



*Fuente: Elaboración Propia*

2. Instalación de paquetes. – Una vez clonado el proyecto se abre una terminal en la ubicación del proyecto y se instala los paquetes detallados en el archivo “requirements.py” con el comando “pip install -r requirements.txt”.
3. Base de datos. – Cada integrante debera crear de forma local una base de datos con el nombre “colegio” usando el motor de PostgreSQL.
4. Cadena de conexión. – El integrante deberá cambiar la cadena de conexión en el archivo “config.py” ingresando su contraseña para su servidor de base de datos.

**Imagen N°3: Cadena de conexión**

```
import os

SQLALCHEMY_DATABASE_URI = 'postgresql://postgres:p%2E210404@localhost/colegio'

SQLALCHEMY_TRACK_MODIFICATIONS = False
SECRET_KEY = os.urandom(24)
```

*Fuente: Elaboración Propia*

5. Script inicial. – Una vez conectado el proyecto a la base de datos se corre el comando “run init\_db.py” para crear las tablas necesarias y el usuario administrador inicial.

### 3.3. Desarrollo del Sistema

#### 3.3.1. Sprint 0



Este sprint se definió tanto el plan de desarrollo del software como el product backlog, el cual se muestra a continuación (Ver Tabla N°2)

**Tabla N°2: Product backlog**

ID	NOMBRE	HISTORIAS DE USUARIO	PRIORIDAD
1	Modelado del sistema	Creación de diagramas UML para una mejor comprensión de proyecto	Alta
2	Registro de Usuario Estudiante	Como estudiante, quiero registrarme con ayuda de mi padre o docente para acceder a la plataforma.	Alta
3	Gestión de Usuarios Docentes y Estudiantes	Como docente, quiero crear, modificar o eliminar cuentas de estudiantes para que accedan fácilmente a sus actividades.	Alta
4	Login Seguro con Roles	Como usuario (docente, estudiante, administrador), quiero iniciar sesión con mis credenciales para acceder a mis funcionalidades.	Alta
5	Dashboard de usuario administrador	Como docente, quiero ver un panel de desempeño de los estudiantes.	Alta
6	Carga y Validación de Contenidos	Como docente, quiero subir y validar contenidos por curso y materia para que los estudiantes practiquen con materiales adecuados.	Alta
7	Ejercicios Interactivos por nivel de grado	Como estudiante, quiero resolver ejercicios con animaciones e interactividad, adaptados a mi nivel, para reforzar lo aprendido.	Alta
8	Evaluación Automática y Retroalimentación	Como estudiante, quiero recibir retroalimentación inmediata tras resolver ejercicios para saber si estoy en lo correcto.	Alta
9	Gestión de Roles y Permisos	Como administrador, quiero asignar roles (docente, estudiante, administrador) para controlar el acceso al sistema.	Alta

10	Diseño UI/UX Infantil	Como estudiante de primaria, se quiere usar una plataforma visualmente atractiva y amigable para niños de mi edad.	Baja
11	Gestión Grados desde usuario administrador	Como administrador se quiere crear grados para asignarle Docentes	Alta
12	Implementación de pruebas unitarias para login	Se quiere implementar mínimo una prueba unitaria para el módulo de login	Baja
13	Implementación de pruebas unitarias para funcionalidades de estudiante	Se quiere implementar mínimo una prueba unitaria para las funcionalidades del estudiante	Baja
14	Implementación de pruebas unitarias para funcionalidades de docente	Se quiere implementar mínimo una prueba unitaria para las funcionalidades del docente	Baja
15	Implementación de pruebas unitarias para funcionalidades de administrador	Se quiere implementar mínimo una prueba unitaria para las funcionalidades del administrador	Baja
16	Desarrollo de proyecto base	Dentro del repositorio se requiere la implementación de la estructura básica bajo la cual se trabajará	Alta
17	Implementación de juegos interactivos estáticos	Se quiere tener los juegos pensados para el proyecto de forma estática en el proyecto	Alta
18	Implementar vista de juegos para usuario administrador	Como administrador se quiere tener una vista de los juegos para su asignación	Alta
19	Implementar vista de juegos para usuario estudiante	Como estudiante quiero ver los juegos que se asignaron a mi grado	Alta
20	Agregar lógica para puntuación de los juegos	Como usuario del sistema quiero contar con lógica de puntuación para los juegos	Alta
21	Agregar vista de puntuaciones para usuario estudiante	Como estudiante quiero ver la puntuación de los juegos que complete.	Alta

22	Agregar vista de puntuaciones para usuario docente	Como docente quiero ver la puntuación de los juegos para cada estudiante de mi grado asignado	Alta
----	--	---	------

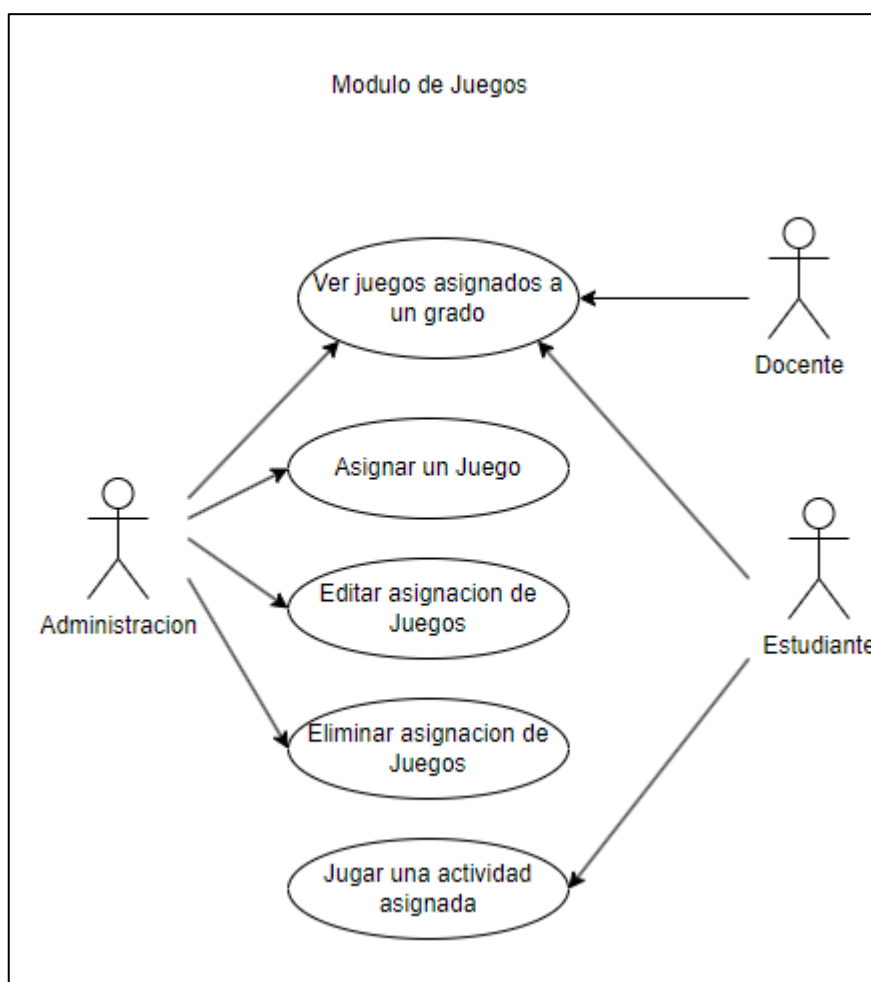
*Fuente: Elaboración Propia*

De igual forma se elaboraron los diagramas de casos de uso para demarcar lo deseado por parte del sistema desde el punto de vista de los diferentes actores

Diagramas iniciales. –

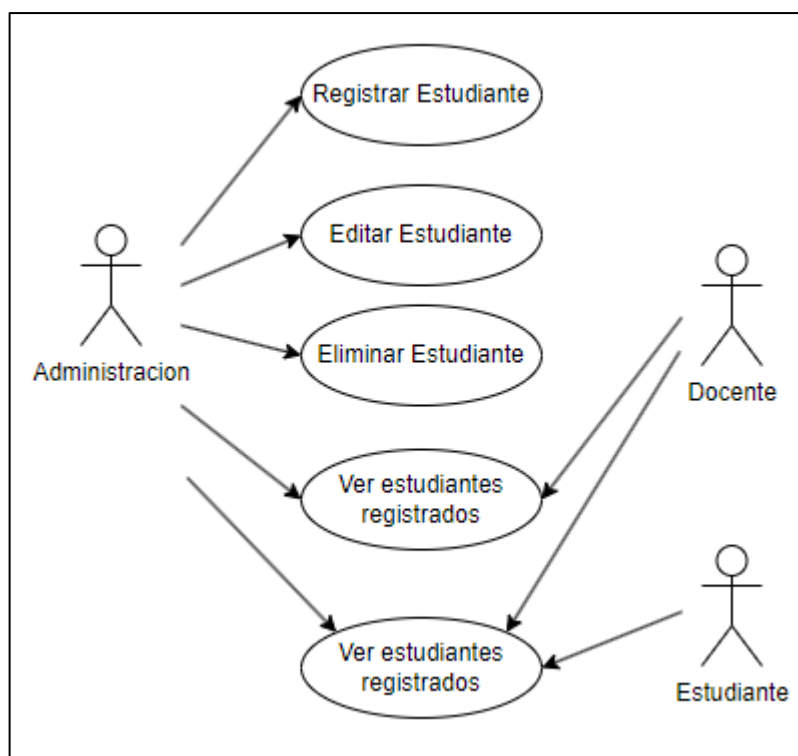
- Diagramas de casos de uso

**Imagen N°4: Casos de uso Modulo de Juegos**



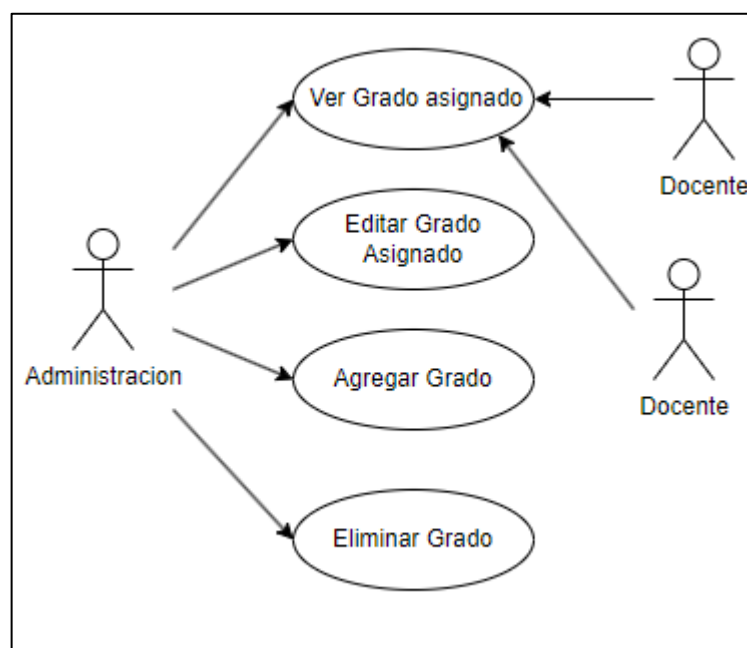
*Fuente: Elaboración propia*

**Imagen N°5: Casos de uso Modulo de estudiantes**



*Fuente: Elaboración propia*

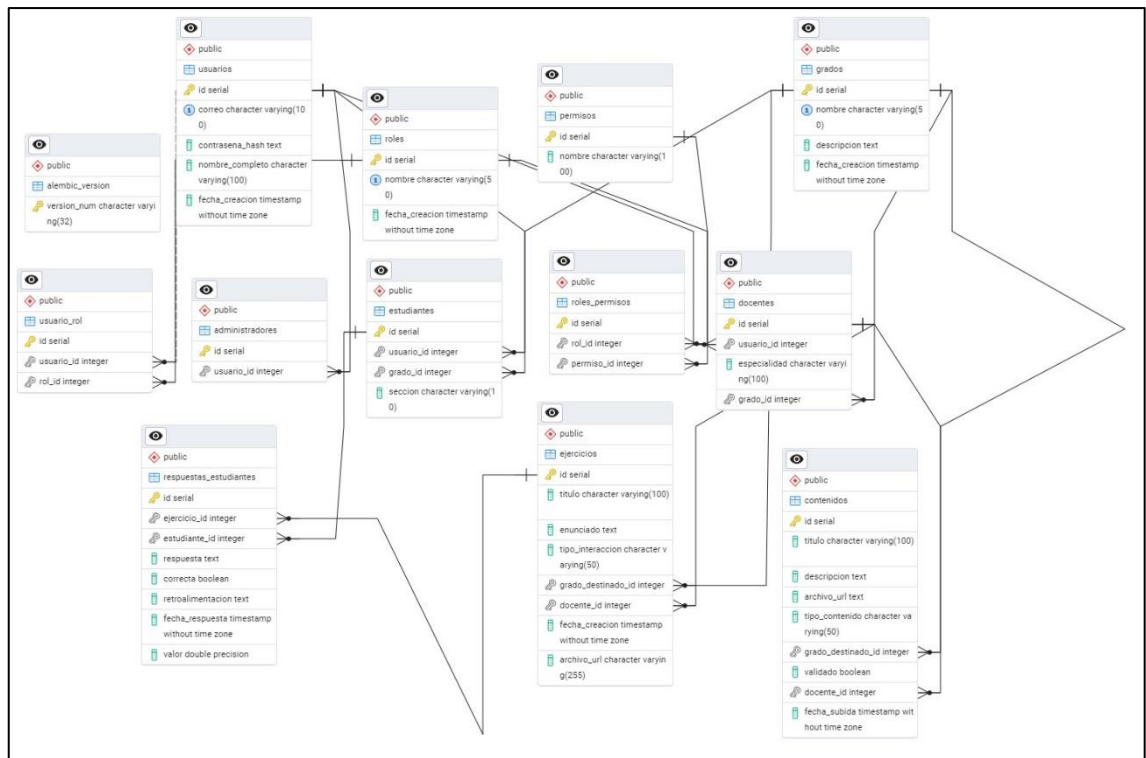
**Imagen N°6: Casos de uso Modulo de Grados**



*Fuente: Elaboración propia*

- Diagrama de Base de datos físico

**Imagen N°7: Modelo físico de base de datos**



*Fuente: Elaboración propia*

## Historial de usuario:

### Administrador

ID	Acción	Descripción Detallada	Restricciones / Condiciones	Relación con Entidades
A1	Crear Docente	El administrador puede registrar nuevos docentes con sus datos personales, correo y rol asignado.	El correo debe ser único. Rol asignado: "Docente".	Usuarios, Roles
A2	Editar Docente	Puede modificar información del docente: nombre, correo, contraseña, etc.	No puede cambiar el rol a "Administrador".	Usuarios
A3	Eliminar Docente	Puede eliminar cuentas de docentes registrados.	Solo si no está asignado activamente a grados.	Usuarios, Grados
A4	Crear Estudiante	Registra estudiantes manualmente o en	Se debe validar el grado antes de la asignación.	Usuarios, Grados

ID	Acción	Descripción Detallada	Restricciones / Condiciones	Relación con Entidades
		lote, asignándolos a un grado.		
A5	Editar Estudiante	Modifica datos de los estudiantes (nombre, grado, contraseña, etc.).	No puede eliminar si tiene puntuaciones activas.	Usuarios, Grados, Juegos
A6	Eliminar Estudiante	Elimina estudiantes del sistema.	Solo si no tiene progreso registrado.	Usuarios, Resultados
A7	Crear Grado	Crea grados académicos y los relaciona con docentes.	No debe existir un duplicado.	Grados, Usuarios
A8	Editar Grado	Puede cambiar nombre del grado y modificar docente asignado.	El docente debe estar disponible.	Grados, Usuarios
A9	Eliminar Grado	Elimina grados completos.	Solo si no hay estudiantes asignados.	Grados, Usuarios
A10	Asignar Roles	Asigna permisos según el tipo de usuario: estudiante, docente, administrador.	No puede autoasignarse como "Estudiante".	Usuarios, Roles
A11	Cargar Juegos/Ejercicios	Sube archivos HTML/CSS/JS de los juegos educativos.	Deben ser validados antes de su publicación.	Juegos
A12	Asignar Juegos a Grados	Relaciona juegos cargados con grados específicos.	Múltiples juegos por grado permitidos.	Juegos, Grados
A13	Editar o Eliminar Juegos	Puede modificar detalles o eliminar juegos cargados anteriormente.	Solo si no están asignados a estudiantes activos.	Juegos
A14	Ver Panel General (Dashboard)	Accede a métricas globales del sistema: usuarios, puntuaciones, avances.	Solo visible para el rol "Administrador".	Resultados, Usuarios, Juegos
A15	Ver Resultados por Estudiante	Consulta el historial de juego y rendimiento de cualquier estudiante.	Acceso total a todos los registros.	Resultados, Juegos, Usuarios

### Docente

ID	Acción	Descripción Detallada	Restricciones / Condiciones	Relación con Entidades
D1	Crear Estudiante	Puede registrar estudiantes y asignarlos a los grados que tiene a cargo.	Solo puede registrar en grados que coordina.	Usuarios, Grados
D2	Editar Estudiante	Modifica información básica de los estudiantes que él creó.	No puede modificar estudiantes creados por admin.	Usuarios, Grados
D3	Eliminar Estudiante	Puede eliminar estudiantes registrados por él mismo.	Solo si no tienen resultados registrados.	Usuarios, Resultados
D4	Visualizar Lista de Estudiantes	Accede a listado de estudiantes por grado asignado.	Solo sus grados asignados.	Usuarios, Grados
D5	Subir Ejercicios Interactivos	Puede cargar nuevos juegos o ejercicios si el sistema lo permite.	Pendiente de validación por admin.	Juegos
D6	Asignar Juegos a Grados	Asocia juegos existentes con los grados que tiene asignados.	Juegos deben estar previamente validados.	Juegos, Grados
D7	Ver Resultados por Estudiante	Visualiza el progreso, puntajes y retroalimentación de estudiantes de sus grados.	Solo estudiantes que coordina.	Resultados, Juegos
D8	Exportar Resultados	Puede exportar las estadísticas a formatos como CSV para análisis offline.	Exportación limitada a sus grados.	Resultados
D9	Acceder a su Dashboard	Visualiza resumen de estudiantes, juegos asignados, rendimiento general.	Personalizado según grados asignados.	Resultados, Juegos

### Estudiante

ID	Acción	Descripción Detallada	Restricciones / Condiciones	Relación con Entidades
E1	Ver Juegos Asignados	Solo puede ver los juegos asignados a su grado por el docente o administrador.	No puede acceder a juegos de otros grados.	Juegos, Grados

ID	Acción	Descripción Detallada	Restricciones / Condiciones	Relación con Entidades
E2	Interactuar con Juegos	Puede jugar ejercicios interactivos HTML/CSS/JS desde el navegador.	Juegos deben estar activos.	Juegos, Resultados
E3	Obtener Retroalimentación	Tras finalizar un juego, recibe calificación o mensajes inmediatos con aciertos y errores.	Retroalimentación automática o personalizada.	Resultados
E4	Ver Historial de Resultados	Accede a listado personal de juegos completados, puntajes, fecha y progreso general.	Solo sus propios resultados.	Resultados
E5	Continuar Juegos Incompletos	Puede retomar juegos que no haya finalizado si la lógica del sistema lo permite.	Debe estar permitido por configuración del juego.	Juegos, Resultados

Por lo que es correcto decir que el objetivo de este sprint es el de establecer lo necesario para el desarrollo del programa.

### 3.3.2. Sprint 1

#### Objetivo General del Sprint 1:

Establecer los fundamentos técnicos y funcionales del proyecto mediante la implementación de la estructura base, la configuración inicial de la base de datos, y el desarrollo de las primeras vistas esenciales para el usuario administrador, garantizando un sistema escalable, organizado y preparado para iteraciones futuras.

#### Objetivos Específicos

Configuración del Entorno y Estructura del Proyecto:

Crear el repositorio del proyecto y definir su estructura de carpetas.

Desarrollar un plan de proyecto detallado con hitos, tareas y asignaciones.

Configurar la cadena de conexión a la base de datos en config.py para permitir la comunicación con el sistema de gestión de bases de datos.

Definición de Modelos y Base de Datos:



Diseñar y documentar los modelos de datos en models.py para representar entidades clave como usuarios, roles, estudiantes y docentes.

Implementar scripts SQL o migraciones iniciales para la creación de tablas y relaciones básicas.

Implementación de Lógica Base:

Establecer el sistema de logging utilizando el patrón Singleton para garantizar un registro centralizado y eficiente de eventos.

Organizar la lógica de negocio en la carpeta /Routes, separando responsabilidades.

Desarrollo de Vistas para el Administrador:

Autenticación:

Implementar un sistema de login/logout seguro con manejo de sesiones.

Roles:

Crear vistas CRUD (Crear, Leer, Actualizar, Eliminar) para la gestión de roles de usuarios.

Estudiantes:

Desarrollar al menos una vista funcional para gestionar información básica de estudiantes.

Docentes:

Iniciar el esquema de vistas para docentes.

Sistema de Grados:

Configurar la relación entre estudiantes y grados/académicos en la base de datos.

Implementar una vista preliminar para que el administrador asigne o consulte grados.

**Tabla de sprint backlog**

ID	Tarea	Estado	Notas
16	Desarrollo de proyecto base (estructura de carpetas, config.py, models.py)	Completado	Incluye configuración inicial de BD.
4	Implementar Login Seguro con Roles (backend + frontend básico)	Completado	Integrado con JWT o sesiones.
9	Gestión de Roles y Permisos (CRUD básico para admin)	Completado	Asignación de permisos implementada.
11	Gestión de Grados (Vista inicial para admin)	Completado	Modelo, vista y creación básica completados.

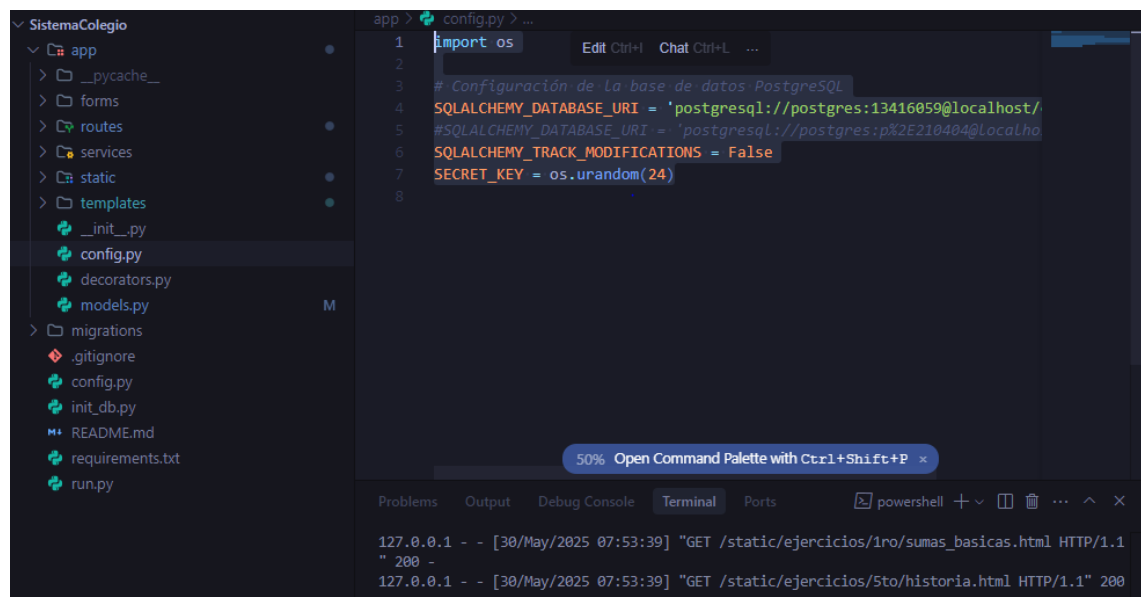
ID	Tarea	Estado	Notas
1	Modelado del sistema (Diagramas UML de clases y flujo de autenticación)	Completado	
5	Dashboard de admin (Vista resumen con estadísticas básicas)	Completado	Finalizado después del login.

### Función principal para la conexión a la base de datos:

```
import os

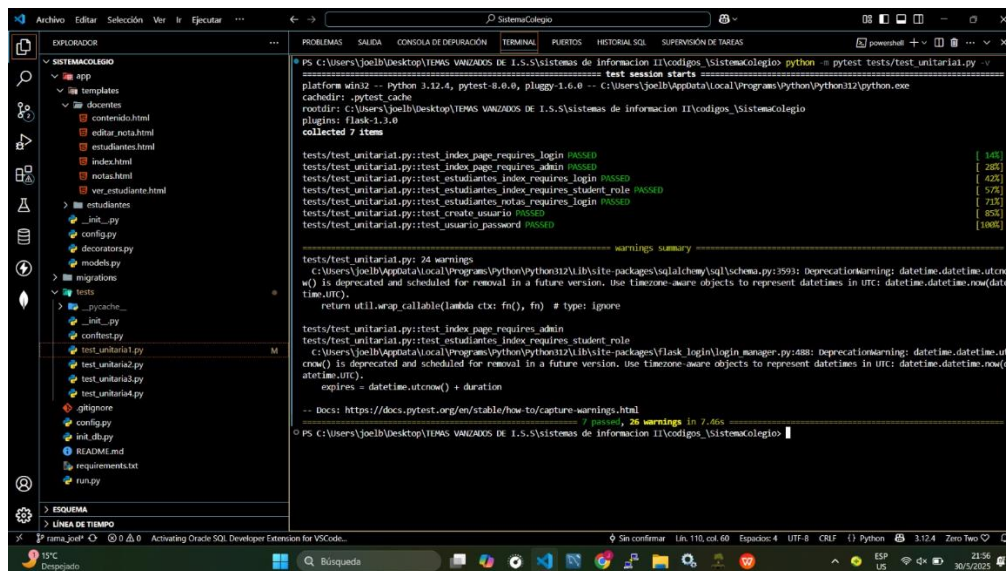
# Configuración de la base de datos PostgreSQL
SQLALCHEMY_DATABASE_URI =
'postgresql://postgres:13416059@localhost/colegio'
#SQLALCHEMY_DATABASE_URI =
'postgresql://postgres:p%2E210404@localhost/colegio'
SQLALCHEMY_TRACK_MODIFICATIONS = False
SECRET_KEY = os.urandom(24)
```

En el código se puede ver los comandos que se tienen que hacer para conectar a la base de datos dentro de nuestro archivo config.py



### Pruebas unitarias:

Prueba 1 Pruebas de autenticación y acceso básico:



```
PS C:\Users\joelb\Desktop\TEMAS VALUADOS DE I.S.S\Sistemas de Informacion II\codigos\SistemaColegio> python -m pytest tests/test_unitaria1.py -v
platform win32 -- python 3.12.4, pytest-8.6.0, pluggy-1.6.0 -- C:\Users\joelb\AppData\Local\Programs\Python\Python312\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\joelb\Desktop\TEMAS VALUADOS DE I.S.S\Sistemas de Informacion II\codigos\SistemaColegio
plugins: flask-1.3.0
collected 7 items

tests/test_unitaria1.py::test_index_page_requires_login PASSED [ 14%]
tests/test_unitaria1.py::test_index_page_requires_admin PASSED [ 28%]
tests/test_unitaria1.py::test_estudiantes_index_requires_login PASSED [ 42%]
tests/test_unitaria1.py::test_estudiantes_index_requires_student_role PASSED [ 57%]
tests/test_unitaria1.py::test_estudiantes_notas_requires_login PASSED [ 71%]
tests/test_unitaria1.py::test_create_usuario PASSED [ 85%]
tests/test_unitaria1.py::test_usuario_password PASSED [100%]

===== warnings summary =====
tests/test_unitaria1.py: 24 warnings
  C:\Users\joelb\AppData\Local\Programs\Python\Python312\Lib\site-packages\sqlalchemy\sql\schema.py:3993: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objects to represent datetimes in UTC: datetime.datetime.now(datetime.UTC).
    return util.wrap_callable(lambda ctx: fn(), fn) # type: ignore

tests/test_unitaria1.py::test_index_page_requires_admin
tests/test_unitaria1.py::test_estudiantes_index_requires_student_role
  C:\Users\joelb\AppData\Local\Programs\Python\Python312\Lib\site-packages\flask_login\login_manager.py:488: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled for removal in a future version. Use timezone-aware objects to represent datetimes in UTC: datetime.datetime.now(datetime.UTC).
    expires = datetime.utcnow() + duration

-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
= 7 passed, 26 warnings in 7.46s =
PS C:\Users\joelb\Desktop\TEMAS VALUADOS DE I.S.S\Sistemas de Informacion II\codigos\SistemaColegio>
```

### 3.3.3 Sprint 2:

#### Objetivo General:

Implementar la gestión de juegos estáticos, mejorar el panel docente con seguimiento de notas y estudiantes, y garantizar la integración entre las vistas de administrador y estudiante.

#### Objetivos Específicos:

##### 1. Gestión de Juegos y Ejercicios

Como administrador, quiero poder cargar y gestionar juegos estáticos para asignarlos a grados específicos.

Implementar interfaz de administración de juegos.

Añadir lógica para vincular juegos a grados/docentes.

Como estudiante, quiero ver los juegos asignados a mi grado.

Desarrollar vista de listado de juegos para estudiantes.

##### 2. Panel Docente Avanzado

Como docente, quiero acceder a un panel principal con herramientas de seguimiento.

Implementar vista principal del docente con resumen de estudiantes y notas.

##### 3. Integración de Notas y Retroalimentación

Como docente, quiero registrar y visualizar notas de los estudiantes.

Implementar vista de notas.

Como estudiante, quiero ver mis resultados y retroalimentación.

Diseñar vista general de desempeño.

##### 4. Carga de Contenidos Estáticos

Como equipo técnico, necesitamos cargar juegos estáticos en el sistema.

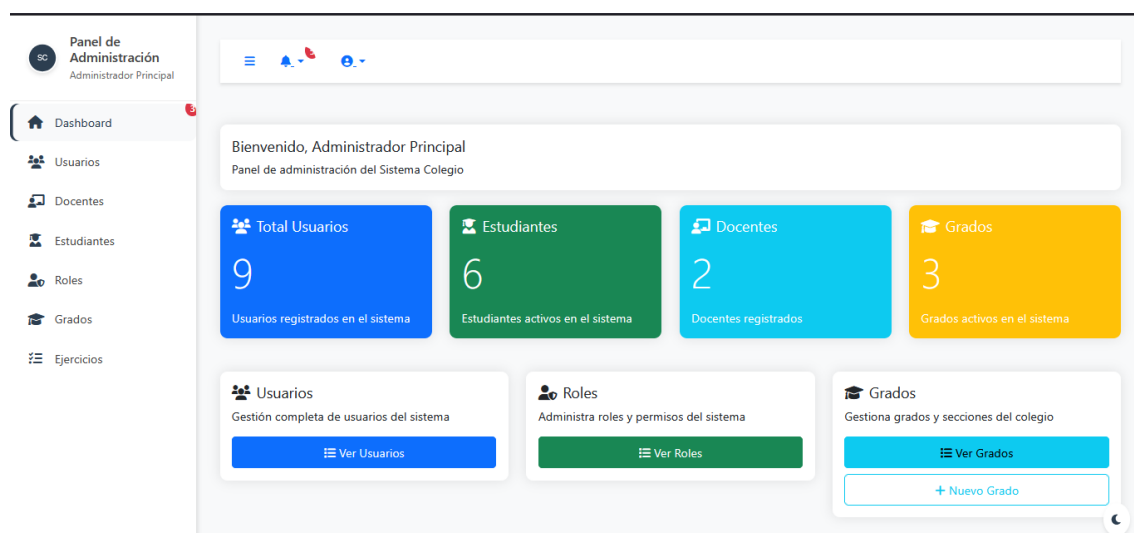
Subir al menos 3 juegos base (ej: crucigramas, sopas de letras) con estructura HTML/CSS/JS.

**Tabla de sprint backlog 2**

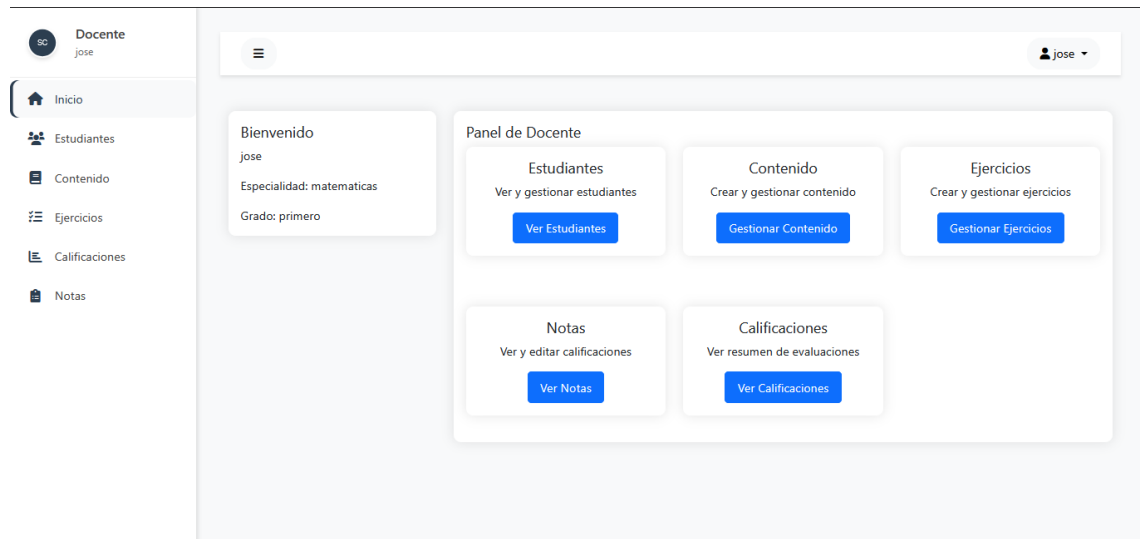
ID	Tarea	Estado	Prioridad	Notas
17	Cargar 3 juegos estáticos al sistema (HTML/CSS/JS)	Completado	Alta	Ubicados en /static/games/.
18	Implementar vista CRUD de juegos para admin (crear/asignar a grados)	Completado	Alta	Integrar con modelo <b>Grado</b> .
19	Vista de listado de juegos asignados para estudiantes	Completado	Alta	Filtrado por <b>grado del estudiante</b> .
3	Panel principal docente (dashboard con resumen de estudiantes/notas)	Completado	Alta	Incluir <b>filtros por grado</b> .
Nueva	Implementar función "Ver detalles del estudiante" (docente)	Completado	Media	Mostrar <b>progreso en juegos y notas</b> .
20	Agregar lógica de puntuación para juegos (backend)	Completado	Alta	Guardar <b>scores</b> en base de datos.
21	Vista de puntuaciones para estudiantes (mis resultados)	Completado	Alta	Tabla con <b>historial de resultados</b> .
22	Vista de puntuaciones para docentes (por estudiante/grado)	Completado	Alta	<b>Exportable a CSV</b> (opcional).
6	Carga inicial de ejercicios interactivos (ejemplo: 2 por materia)	Completado	Alta	—

## Función principal del sprint 2:

Crud de cada usuario (interfaz):



## Docente:



## Estudiante:



### 3.3.4 Sprint 3:

#### Objetivo General:

Mejorar la experiencia docente con vistas de gestión de estudiantes y juegos, implementar pruebas unitarias críticas y refinar el diseño de interfaces existentes.

#### Objetivos Específicos:

##### 1. Gestión de Estudiantes para Docentes

Como docente, quiero ver una lista de estudiantes registrados en mi grado asignado.

- Implementar vista de lista de estudiantes con filtros.
- Permitir acceso rápido a detalles individuales.

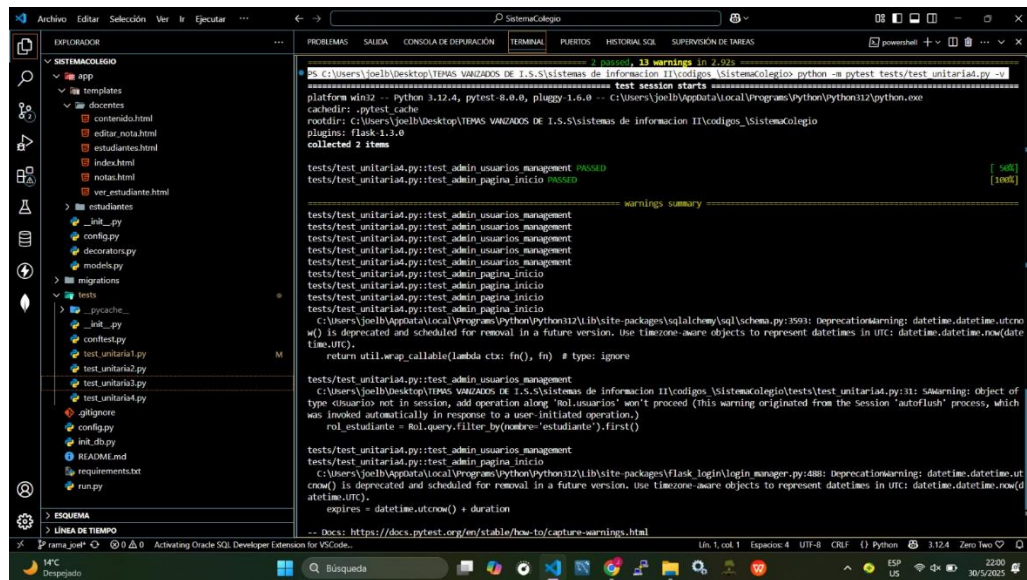
##### 2. Mejoras en la Gestión de Juegos

Como docente, quiero agregar y eliminar juegos desde mi panel.

- Desarrollar vista "Agregar Juego" con formulario intuitivo.



## prueba 4 Pruebas de funcionalidades de administrador



```
PS C:\Users\joelb\Desktop\TPAS VANZADOS DE I.S.S\sisemas de informacion II\codigos \Sistemacolegio python -m pytest tests/test_unitariad.py -v
platform win32 -- python 3.12.4, pytest-8.0.0, pluggy-1.6.0 -- C:\Users\joelb\AppData\Local\Programs\Python\Python312\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\joelb\Desktop\TPAS VANZADOS DE I.S.S\sisemas de informacion II\codigos \Sistemacolegio
plugins: flask-1.3.0
collected 2 items

tests/test_unitariad.py::test_admin_usuarios_management PASSED [ 50%]
tests/test_unitariad.py::test_admin_pagina_inicio PASSED [100%]

===== warnings summary =====

tests/test_unitariad.py::test_admin_usuarios_management
tests/test_unitariad.py::test_admin_usuarios_management
tests/test_unitariad.py::test_admin_usuarios_management
tests/test_unitariad.py::test_admin_usuarios_management
tests/test_unitariad.py::test_admin_usuarios_management
tests/test_unitariad.py::test_admin_pagina_inicio
tests/test_unitariad.py::test_admin_pagina_inicio
tests/test_unitariad.py::test_admin_pagina_inicio
tests/test_unitariad.py::test_admin_pagina_inicio
tests/test_unitariad.py::test_admin_pagina_inicio
C:\Users\joelb\AppData\Local\Programs\Python\Python312\lib\site-packages\sqlalchemy.sql.schema.py:3593: DeprecationWarning: datetime.datetime.utcnow()
is deprecated and scheduled for removal in a future version. Use timezone-aware objects to represent datetimes in UTC: datetime.datetime.now(datetime.UTC).
    return util.wrap_callable(lambda ctx: fn(), fn) # type: ignore

tests/test_unitariad.py::test_admin_usuarios_management
C:\Users\joelb\Desktop\TPAS VANZADOS DE I.S.S\sisemas de informacion II\codigos \Sistemacolegio\tests\test_unitariad.py:31: SWarning: object of
type usuarios not in session, add operation along 'rol.usuarios' won't proceed (this warning originated from the Session 'autoflush' process, which
was invoked automatically in response to a user-initiated operation.)
    rol.estudiante = Rol.query.filter_by(nombre='estudiante').first()

tests/test_unitariad.py::test_admin_usuarios_management
tests/test_unitariad.py::test_admin_pagina_inicio
C:\Users\joelb\AppData\Local\Programs\Python\Python312\lib\site-packages\flask_login\login_manager.py:488: DeprecationWarning: datetime.datetime.utcnow()
is deprecated and scheduled for removal in a future version. Use timezone-aware objects to represent datetimes in UTC: datetime.datetime.now(datetime.UTC).
    expires = datetime.utcnow() + duration

-- docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
```

## Bibliografía

- Cuartas, M. A. (2019). *Trello, Gestion del Tiempo y Organizacion del Trabajo en Equipo*. Madrid.
- Grinberg, M. (2018). *Flask Web Development*. O'Reilly Media.
- Group, P. G. (2025). *What is PostgreSQL?* Obtenido de PostgreSQL Documentation: <http://postgresql.org/docs/current/intro-what-is.html>
- Rossum, G. V. (2025). *Python Documentation Index*. Obtenido de python.org: <https://www.python.org/doc/essays/>
- Scoot Chacon, B. S. (2025). *Pro Git*. California, USA: Creative Commons.
- Sutherland, K. S. (2017). *The Scrum Guide*. Creative Commons.