

2º DAM

# MEMORIA DE TRABAJO

## PROYECTO INTEGRAL

---



Alumno: Roberto Velázquez Vázquez

Clase: 2º DAM

Asignatura: Proyecto Integral

---

# ÍNDICE

## PROYECTO INTEGRAL

---

Sprint 1.....	3.
- Objetivos.....	3.
- Alcance del proyecto.....	3.
- Identificación de Requisitos del Sistema.....	4.
- Identificación del entorno tecnológico.....	6.
- Definición de casos de uso.....	8.
- Definición del plan de pruebas.....	19.
 Sprint 2.....	22.
- Modelo de datos.....	22.
- Modelo E/R.....	22.
- Paso a tabla.....	23.
- Backend.....	24.
- Diagrama UML de clases.....	24.
- Definición de catálogo de servicios.....	25.
- Codificación y pruebas de los servicios REST.....	33.
 Sprint 4.....	37.
- Integración.....	37.
- Securización.....	37.
- Ejecución del plan de pruebas.....	38.
- Pruebas de Aceptación del Usuario (UAT).....	43.

---

# SPRINT 1 PROYECTO INTEGRAL

## PROYECTO INTEGRAL

---

### 1. Objetivo

El proyecto tiene como objetivo principal desarrollar una página web institucional que facilite la gestión de la conducta del alumnado en un centro educativo.

Especificamente, busca:

- Agilizar el registro y seguimiento de sanciones y expulsiones.
- Permitir la notificación eficiente a los padres o tutores sobre las sanciones.
- Mejorar la coordinación entre la jefatura y el profesorado para la asignación de tareas al alumnado sancionado.
- Garantizar un seguimiento efectivo de las tareas asignadas durante el periodo de expulsión.
- Facilitar la consulta de información relevante sobre sanciones y su historial.

### 2. Alcance del Proyecto.

El sistema estará diseñado para su uso por jefatura, docentes y tutores legales.

Incluye las siguientes funcionalidades:

- Registro de sanciones y expulsiones con detalles clave.
- Consulta y modificación de sanciones registradas.
- Notificación a los padres o tutores sobre las expulsiones.

- 
- Generación y seguimiento de tareas para alumnos sancionados.
  - Revisión de tareas completadas al reincorporarse el alumno.

### **3. Identificación de Requisitos del Sistema.**

- **Identificación de requisitos funcionales.**
  - El sistema deberá almacenar y gestionar info. sanciones.
  - **Registro de sanciones:**
    - Permitir a la administración registrar sanciones con datos del alumno, fecha, tipo de sanción, duración, descripción y tareas asociadas.
  - **Consulta y modificación de sanciones:**
    - Listado de sanciones registradas.
    - Visualización detallada de cada sanción.
    - Edición de sanciones para actualizar información relevante.
  - **Notificaciones:**
    - Notificación a los padres sobre la expulsión del alumno.
    - Registrar el medio utilizado para la notificación.
  - **Tareas para alumnos sancionados:**
    - Registro de tareas con asignatura, descripción, estado y fecha de reincorporación.
    - Consulta de tareas asignadas a alumnos expulsados.
    - Revisar si se han completado las tareas propuestas.
    -

#### **Roles del Sistema**

##### **1. ROL001 - Administrador**

Acceso a todas las funcionalidades, incluyendo la gestión de sanciones.

---

## 2. ROL002 - Docente

Acceso a la consulta del historial de conducta del alumnado. Consulta de sanciones y tareas asignadas. Creación, notificación y revisión de Tareas por expulsión tras el alta de la misma. Notificación de expulsiones tras el alta o modificación de la misma. Consulta de sanciones y tareas asignadas.

### - Identificación de requisitos NO funcionales.

#### - RNF001 - Usabilidad:

- La interfaz deberá ser intuitiva y accesible para los usuarios del sistema.
- Compatible con dispositivos móviles y de escritorio.

#### - RNF002 - Seguridad:

- Acceso restringido mediante autenticación de usuarios.
- Control de permisos según el rol del usuario (jefatura, docente, tutor).
- Cifrado de datos sensibles como información del alumnado.

#### - RNF003 - Rendimiento:

- Disponibilidad 24/7 con tiempos de respuesta óptimos.
- Capacidad para manejar múltiples solicitudes simultáneas.

#### - RNF004 - Mantenibilidad:

- Diseño modular para facilitar futuras mejoras o expansiones.
- Uso de tecnologías actualizadas y documentación adecuada.

## 4. Identificación del entorno tecnológico.

### - Frontend:

- **Angular:** Framework basado en TypeScript para la creación de aplicaciones SPA (Single Page Application).
- **Bootstrap:** Biblioteca de estilos CSS que facilita el diseño responsive y estético.

- 
- **Backend:**
    - **Spring Boot:** Framework de Java para la construcción de aplicaciones web y APIs REST.
    - **Spring MVC:** Arquitectura basada en controladores para manejar las solicitudes HTTP.
    - **Hibernate & JPA:** ORM (Object-Relational Mapping) para la interacción con la base de datos.
    - **Spring Data JPA:** Simplificación del acceso a los datos mediante repositorios.
    - **Spring Security:** Para autenticación y autorización si se requiere.
  - **Base de Datos:**
    - **MySQL:** Base de datos relacional para almacenar los registros de partes disciplinarios, expulsiones, historial, etc.

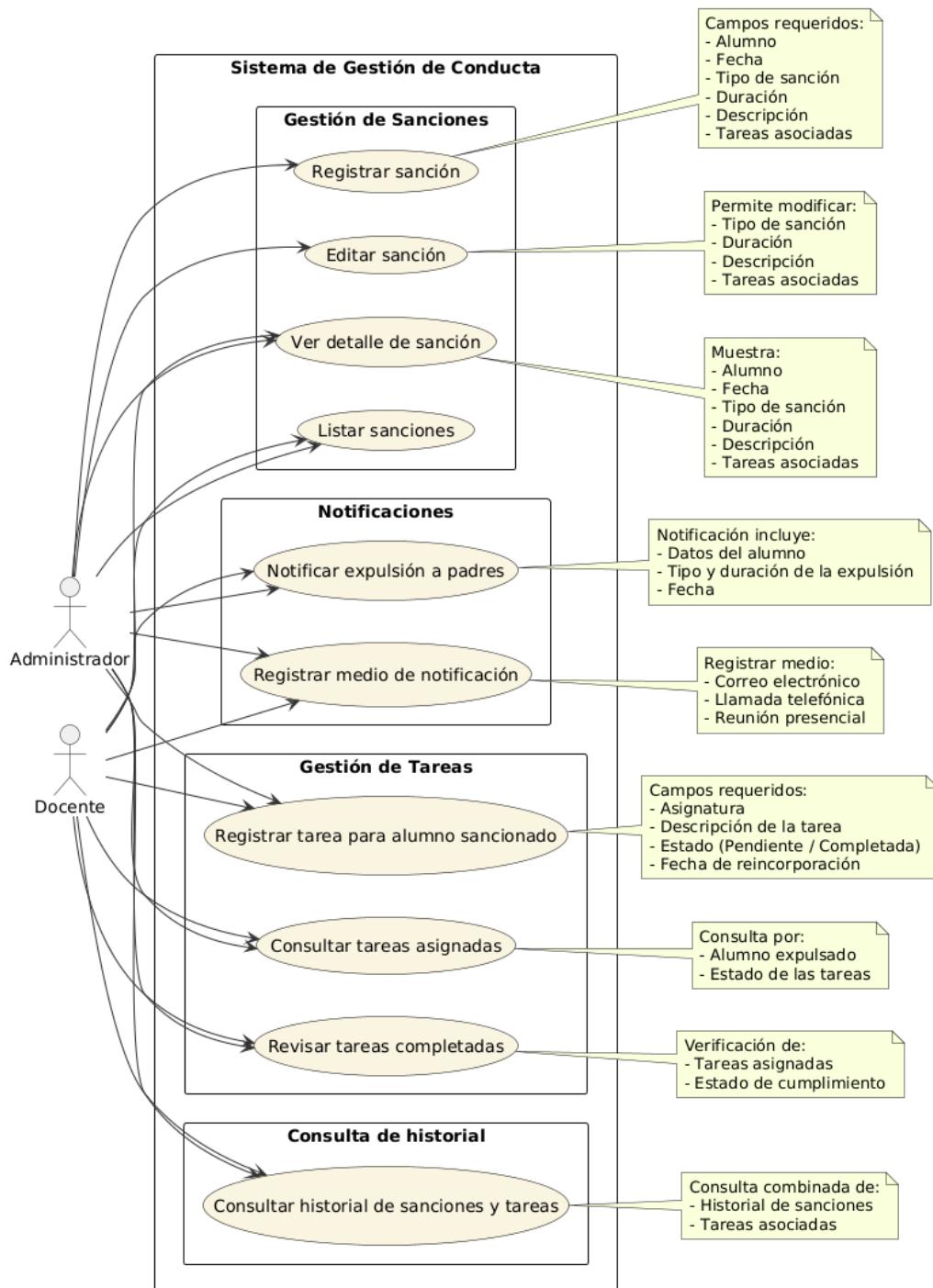
## Infraestructura y Desarrollo

- **Arquitectura de la Aplicación:**
  - Aplicación **cliente-servidor** con un frontend en Angular y un backend en Spring Boot.
  - Comunicación mediante **APIs RESTful**, donde Angular envía peticiones al backend.

- 
- **Base de datos centralizada** en MySQL, gestionada a través de Hibernate y JPA.
- **Desarrollo y Herramientas:**
- **IDE:** Visual Studio Code para Angular y Eclipse para el backend.
  - **Gestión de dependencias:** Maven para Spring Boot y npm para Angular.
  - **Control de versiones:** Git (GitHub).
  - **Entorno de desarrollo:** Local.
- **Infraestructura y Despliegue:**
- **Backend:** se ejecuta en un servidor Tomcat embebido.
  - **Frontend:** de forma local.
  - **Base de datos:** MySQL alojado en un servidor local.
  - **Autenticación y Seguridad:** JWT y spring security para el control de acceso.

## 5. Definición de casos de uso.

### a. Diagrama de casos de uso.



---

b. Especificación de cada caso de uso.

i. Registro de sanciones:

- Identificador Caso de Uso: Caso de Uso 1
- Título del Caso de Uso: Registrar sanciones
- Descripción: Este caso de uso permite a la Administración registrar una sanción a un alumno, incluyendo detalles como la fecha, duración, descripción, partes asociados, tipo de sanción (con o sin expulsión), y tareas asociadas.
- Precondición: El usuario debe estar autenticado como Jefatura y tener acceso al sistema de gestión de sanciones. El alumno debe estar identificado en el sistema.
- Requisitos relacionados: Sistema de gestión de sanciones, acceso a los datos del alumno, validación de la sanción (expulsión, tareas, etc.).
- Secuencia de pasos que ocurren:
  - a. El usuario (Administrador) accede al sistema de gestión de sanciones.
  - b. Se selecciona al alumno filtrándolo por nivel, grupo y nombre y apellido al que se le aplicará la sanción.
  - c. El sistema cargará el listado de partes pendientes de sanción del alumno.
  - d. Se ingresan los detalles de la sanción (fecha, duración, descripción, partes asociados, tipo de sanción y tareas asociadas).
  - e. El usuario seleccionara los partes que se asociaran a la sancion.

- 
- f. El sistema valida los datos e ingresa la sanción en el sistema.
  - Postcondición: La sanción se registra correctamente identificada con un id autogenerado en el sistema.
  - Excepciones:
    - a. Si los datos ingresados no son válidos (por ejemplo, fechas incorrectas o campos vacíos), se muestra un mensaje de error y el usuario debe corregir los datos.
    - b. Si el alumno no tiene partes pendientes de sanción, se muestra un error indicando que no se pudo encontrar al alumno.
- 
- ii. Consulta Lista de Sanciones:
    - Identificador Caso de Uso: Caso de Uso 2.
    - Título del Caso de Uso: Consultar Lista de sanción.
    - Descripción: Este caso de uso permite a los actores (Administración, Profesor) consultar una lista de sanciones registradas, visualizando información sobre el alumno, fecha de sanción, tipo de sanción, etc.
    - Precondición: El usuario debe estar autenticado y tener permisos para acceder a la información de sanciones.
    - Requisitos relacionados: Sistema de gestión de sanciones, permisos de acceso.
    - Secuencia de pasos que ocurren:
      - a. El usuario accede al sistema de gestión de sanciones.

- 
- b. Se solicita la lista de sanciones registradas.
  - c. El sistema muestra la lista de sanciones con los detalles (alumno, fecha, tipo, estado).
- Postcondición: El usuario visualiza la lista de sanciones.
  - Excepciones:
    - a. Si no hay sanciones registradas, se muestra un mensaje indicando que no hay sanciones en el sistema.
    - b. Si el usuario no tiene permisos para acceder a la lista, se muestra un mensaje de acceso denegado.

iii. Consultar detalle de sanción:

- Identificador Caso de Uso: Caso de Uso 3.
- Título del Caso de Uso: Consultar detalle de sanción
- Descripción: Permite a los actores consultar el detalle completo de una sanción específica registrada, incluyendo información del alumno, tipo de sanción, tareas asociadas, etc.
- Precondición: El usuario debe estar autenticado y tener permisos para acceder al detalle de las sanciones.
- Requisitos relacionados: Sistema de gestión de sanciones, permisos de acceso.
- Secuencia de pasos que ocurren:
  - a. El usuario accede al sistema de gestión de sanciones.
  - b. Se selecciona una sanción de la lista de sanciones.
  - c. El sistema muestra el detalle de la sanción seleccionada.

- 
- Postcondición: El usuario visualiza la información completa de la sanción.
  - Excepciones:  
Si la sanción seleccionada no existe, se muestra un mensaje de error indicando que no se encontró.
    - a. Si el usuario no tiene permisos, se muestra un mensaje de acceso denegado.

iv. Modificar detalle de una sanción:

- Identificador Caso de Uso: Caso de Uso 4.
- Título del Caso de Uso: Modificar detalle de una sanción
- Descripción: Permite a Administración modificar los detalles de una sanción registrada (fecha, duración, descripción, etc.).  
Precondición: El usuario debe estar autenticado y tener permisos para modificar sanciones. La sanción debe existir en el sistema.
- Requisitos relacionados: Sistema de gestión de sanciones, permisos de acceso, validación de los cambios.
- Secuencia de pasos que ocurren:
  - a. El usuario (Administración) accede al sistema.
  - b. Se selecciona una sanción para modificar.
  - c. El usuario realiza los cambios necesarios en los detalles de la sanción.
  - d. El sistema valida los datos ingresados y guarda los cambios.

- 
- Postcondición: La sanción es actualizada correctamente en el sistema con los nuevos datos.
  - Excepciones:
    - a. Si los datos modificados son inválidos, se muestra un mensaje de error y se solicita la corrección.
    - b. Si el usuario no tiene permisos para modificar la sanción, se muestra un mensaje de acceso denegado.

v. Notificar a los Padres sobre una Expulsión:

- Identificador Caso de Uso: Caso de Uso 5.
- Título del Caso de Uso: Notificar a los Padres sobre una Expulsión
- Descripción: Permite a Profesor y Administración enviar una notificación a los padres de un alumno cuando se le impone una expulsión.
- Precondición: El alumno debe tener una sanción registrada con expulsión.
- Requisitos relacionados: Sistema de gestión de sanciones, permisos de acceso, sistema de notificación (email).
- Secuencia de pasos que ocurren:
  - a. El sistema identifica que un alumno tiene una expulsión registrada.
  - b. El usuario (Profesor o Administración) accede al sistema de sanciones.

- 
- c. El sistema genera automáticamente la notificación a los padres, incluyendo todos los detalles relevantes.
  - d. La notificación se envía a los padres.
- Postcondición: Los padres reciben la notificación de expulsión.
  - Excepciones:
    - a. Si no se puede enviar la notificación por algún medio (problemas con el email o dirección incorrecta), se muestra un mensaje de error.

vi. Notificar a los Padres tras creación de una tarea por expulsión:

- Identificador Caso de Uso: Caso de Uso 6.
- Título del Caso de Uso: Notificar a los Padres tras creación de tarea por expulsión
- Descripción: Permite a Profesor y Administración notificar a los padres cuando se crea una tarea asociada a una expulsión.
- Precondición: El alumno debe tener una tarea registrada asociada a una expulsión.
- Requisitos relacionados: Sistema de tareas, permisos de acceso, sistema de notificación.
- Secuencia de pasos que ocurren:
  - a. El sistema genera la tarea asociada a la expulsión del alumno.
  - b. El usuario (Profesor o Administración) accede al sistema de tareas.

- 
- c. El sistema genera una notificación con la descripción de la tarea.
  - d. La notificación se envía a los padres.
- Postcondición: Los padres reciben la notificación de la tarea asociada a la expulsión.
  - Excepciones:
    - a. Si no se puede enviar la notificación, se muestra un mensaje de error y el usuario debe intentar de nuevo.

vii. Registrar tareas para el alumnado durante expulsión:

- Identificador Caso de Uso: Caso de Uso 7.
- Título del Caso de Uso: Registrar tareas para el alumnado durante expulsión.
- Descripción: Permite registrar tareas a un profesor para un alumno durante su expulsión, incluyendo asignatura, descripción, estado y fecha de reincorporación.
- Precondición: El alumno debe estar expulsado y el usuario (Profesor) debe estar autenticado.
- Requisitos relacionados: Sistema de tareas, gestión de expulsiones.
- Secuencia de pasos que ocurren:
  - a. El usuario (Profesor) accede al sistema de tareas.
  - b. Se selecciona al profesor y se asigna una tarea relacionada con la expulsión del alumno.

- 
- c. El usuario ingresa los detalles de la tarea (asignatura, descripción, estado, fecha).
  - d. El sistema guarda la tarea en el registro correspondiente.
- Postcondición: La tarea es registrada correctamente en el sistema.
  - Excepciones:
    - a. Si los datos ingresados no son válidos, el sistema muestra un mensaje de error.
    - b. Si el alumno no está registrado o no está expulsado, se muestra un mensaje de error.

#### viii. Consultar tareas alumnado expulsado:

- Identificador Caso de Uso: Caso de Uso 8.
- Título del Caso de Uso: Consultar tareas alumnado expulsado
- Descripción: Este caso de uso permite a los actores (Administración, Profesor) consultar las tareas asociadas a un alumno que se encuentra en situación de expulsión.
- Precondición: El usuario debe estar autenticado y tener permisos para acceder a la información sobre las tareas del alumnado expulsado, además el alumno debe estar expulsado.
- Requisitos relacionados: Sistema de gestión de tareas, permisos de acceso.
- Secuencia de pasos que ocurren:

- 
- a. El usuario (Administración, Profesor) accede al sistema de tareas.
  - b. Se solicita la lista de tareas asociadas a un alumno expulsado.
  - c. El sistema muestra todas las tareas asignadas al alumno, con los detalles correspondientes (asignatura, estado, fecha de reincorporación, etc.).
- Postcondición: El usuario visualiza la lista de tareas asociadas al alumno expulsado.
  - Excepciones:
    - a. Si no existen tareas registradas para el alumno expulsado, se muestra un mensaje indicando que no hay tareas pendientes.
    - b. Si el usuario no tiene permisos para acceder a las tareas, se muestra un mensaje de acceso denegado.

ix. Revisión de tareas de alumnado expulsado en la incorporación:

- Identificador Caso de Uso: Caso de Uso 9.
- Título del Caso de Uso: Revisión de tareas de alumnado expulsado en la incorporación
- Descripción: Este caso de uso permite a los profesores que son tutores y a la Administración revisar las tareas entregadas por un alumno al reincorporarse después de una expulsión, comparando las tareas entregadas con las previstas.

- 
- Precondición: El alumno debe haberse reincorporado al centro después de la expulsión y las tareas deben haber sido previamente registradas.
  - Requisitos relacionados: Sistema de gestión de tareas, registro de tareas entregadas.
  - Secuencia de pasos que ocurren:
    - a. El usuario (Profesor o Administración) accede al sistema de tareas.
    - b. Se selecciona al alumno que se reincorpora.
    - c. El sistema muestra las tareas asignadas al alumno durante la expulsión.
    - d. El usuario revisa las tareas entregadas por el alumno y las compara con las tareas previstas.
    - e. Si todas las tareas se han entregado correctamente, se marca la tarea como completada.
    - f. El usuario guarda el estado actualizado de las tareas.
  - Postcondición: Las tareas del alumno se revisan y se marca la entrega como completada si se cumple todo.
  - Excepciones:
    - a. Si no se han entregado todas las tareas, se muestra un mensaje indicando que algunas tareas no han sido entregadas.
    - b. Si el alumno no está identificado o no tiene tareas asignadas, se muestra un mensaje de error.

---

## **6. Definición de plan de pruebas.**

### **1. Introducción**

Este documento presenta el plan de pruebas para la página web institucional destinada a gestionar la conducta del alumnado en un centro educativo. Se detallan la estructura, cobertura, estrategias, cronograma, documentación, validación, riesgos y presentación del proceso de pruebas.

### **2. Objetivos y Alcance**

El objetivo principal del sistema es agilizar y mejorar el seguimiento de las sanciones del alumnado, notificar a padres/tutores, coordinar la asignación de tareas durante expulsiones y garantizar un seguimiento adecuado. El sistema será utilizado por la jefatura, docentes y tutores legales, y permitirá: registrar y modificar sanciones, notificar expulsiones, gestionar tareas para alumnos sancionados y revisar el historial de conducta.

### **3. Cobertura de Pruebas**

Las pruebas incluirán:

- Pruebas funcionales: validación de cada funcionalidad descrita en los requisitos.
- Pruebas no funcionales: usabilidad, rendimiento, seguridad y mantenibilidad.
- Pruebas de rendimiento: tiempos de respuesta y concurrencia.

---

#### **4. Identificación de Casos de Prueba**

Cada caso de prueba incluirá:

- Entrada esperada (datos de usuario, acciones).
- Salida esperada (resultados del sistema).
- Criterios de éxito (condiciones para considerar la prueba como superada).

#### **5. Cobertura de Requisitos**

El plan asegura que todos los requisitos funcionales y no funcionales tienen al menos un caso de prueba asociado, garantizando una trazabilidad total.

#### **6. Estrategia y Herramientas de Pruebas**

- Pruebas unitarias: Junit, Mockito (backend).
- Pruebas de integración: Cypress.
- Pruebas de sistema: Postman.

#### **7. Cronograma y Planificación**

El cronograma contempla las siguientes fases:

- Preparación de pruebas: 1 semana.
- Ejecución de pruebas unitarias e integración: 2 semanas.
- Pruebas de sistema y aceptación: 2 semanas.
- Documentación y corrección de errores: 1 semana.

---

## **8. Documentación de Resultados**

Los resultados se documentarán en informes por módulo, incluyendo casos ejecutados, resultados obtenidos, errores detectados, su gravedad y acciones tomadas. Se usará un formato estándar con tablas de seguimiento.

## **9. Validación y Aceptación**

Los entregables del software se validarán mediante pruebas de aceptación ejecutadas por usuarios clave. Los criterios de aceptación incluyen: cumplimiento funcional, estabilidad, rendimiento y seguridad esperada.

## **10. Identificación de Riesgos**

- Fallos en notificaciones → Mitigación: pruebas específicas de comunicación.
- Errores de acceso o permisos → Mitigación: pruebas con distintos roles.
- Lenta respuesta bajo carga → Mitigación: pruebas de carga y estrés.

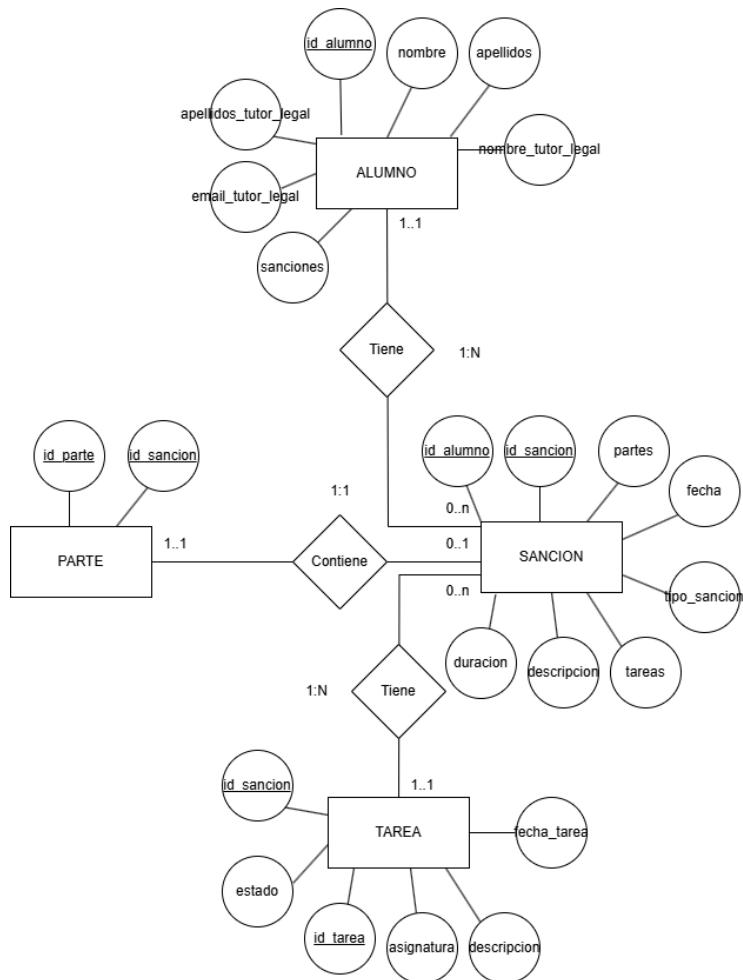
# SPRINT 2 PROYECTO INTEGRAL

## PROYECTO INTEGRAL

---

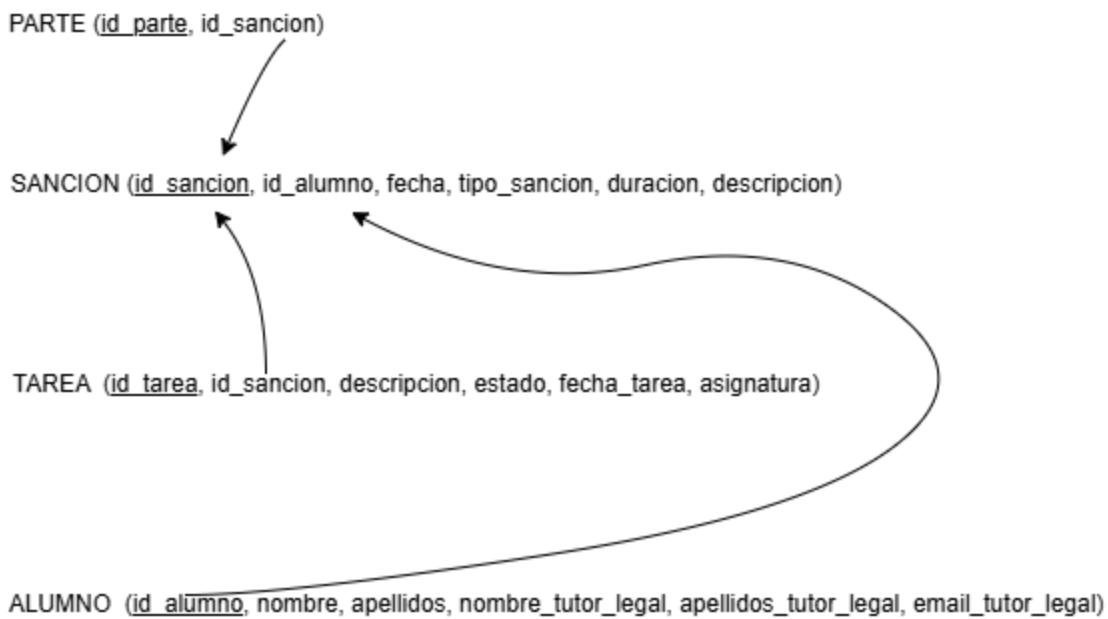
### 1. Modelo de datos

#### 1.1. Modelo E/R.



---

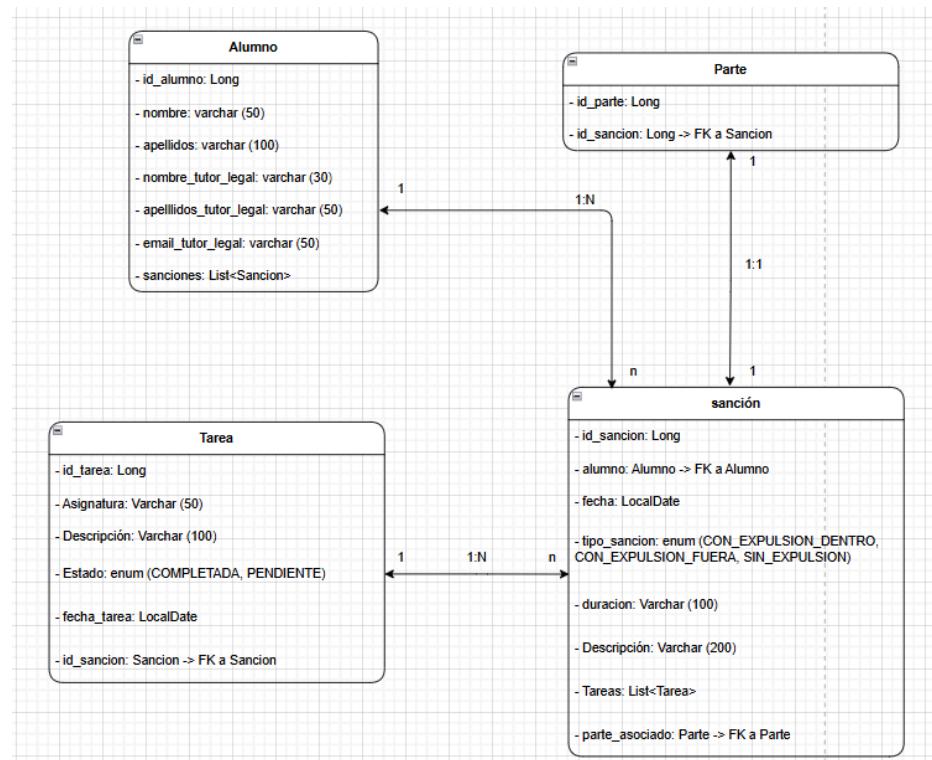
## 1.2. Paso a Tabla.



## 1.3. Creación Base de Datos.

## 2. Backend

### 2.1. Diagrama UML de clases.



---

## **2.2. Definición de catálogo de servicios.**

<b>Nombre del Servicio</b>	<b>Crear Sanción</b>
Ruta URL	localhost:8084/guzpasen/conducta/sancion
Método	POST
Entrada: valores y tipo	Body (JSON con objeto Sanción)
Modo de Acceso	Postman
Formato Respuesta	JSON (Objeto Sanción)
Código Respuesta	201 CREATED
Descripción	Crea una nueva sanción en el sistema.
Dependencias:	Clientes que necesiten registrar sanciones.

```
{  
  
    "tipoSancion": "CON_EXPULSION_FUERA",  
  
    "duracion": "3 días",  
  
    "alumno": {  
  
        "dni": "123456780"  
  
    },  
  
    "parte": {  
  
        "id": 1  
  
    },  
  
}
```

---

<b>Nombre del Servicio</b>	Obtener Sanción por ID
Ruta URL	localhost:8084/guzpasen/conducta/sancion/{id}
Método	GET
Entrada: valores y tipo	Path Variable (Long id)
Modo de Acceso	Navegador, Postman
Formato Respuesta	JSON (Objeto Sanción)
Código Respuesta	200 OK
Descripción	Retorna los detalles de una sanción específica según su ID.
Dependencias:	Clientes que necesiten consultar una sanción específica.

---

<b>Nombre del Servicio</b>	<b>Obtener Todas las Sanciones</b>
Ruta URL	localhost:8084/guzpasen/conducta/sancion
Método	GET
Entrada: valores y tipo	Ninguna
Modo de Acceso	Navegador, Postman
Formato Respuesta	JSON (Lista de objetos Sanción)
Código Respuesta	200 OK
Descripción	Retorna una lista con todas las sanciones registradas en el sistema.
Dependencias:	Clientes que necesiten listar todas las sanciones.

---

<b>Nombre del Servicio</b>	<b>Actualizar Sanción</b>
Ruta URL	localhost:8084/guzpasen/conducta/sancion/{id}
Método	PUT
Entrada: valores y tipo	Path Variable (Long id), Body (JSON con objeto Sanción actualizado)
Modo de Acceso	Postman
Formato Respuesta	JSON (Objeto Sanción actualizado)
Código Respuesta	200 OK
Descripción	Actualiza los datos de una sanción existente según su ID.
Dependencias:	Clients que necesiten modificar sanciones existentes.

```
{
    "tipoSancion": "SIN_EXPULSION",
    "duracion": "1 mes"
}
```

---

<b>Nombre del Servicio</b>	<b>Eliminar Sanción</b>
Ruta URL	localhost:8084/guzpasen/conducta/sancion/{id}
Método	DELETE
Entrada: valores y tipo	Path Variable (Long id)
Modo de Acceso	Postman
Formato Respuesta	Ninguno (Void)
Código Respuesta	200 OK
Descripción	Elimina una sanción del sistema según su ID.
Dependencias:	Clientes que necesiten eliminar sanciones.

---

<b>Nombre del Servicio</b>	<b>Obtener Tareas de Alumno Expulsado</b>
Ruta URL	localhost:8084/guzpasen/conducta/sancion/tareas-alumno-expulsado/{dni}
Método	GET
Entrada: valores y tipo	Path Variable (String dni)
Modo de Acceso	Navegador, Postman
Formato Respuesta	JSON (Lista de objetos Tarea)
Código Respuesta	200 OK
Descripción	Retorna las tareas asignadas a un alumno expulsado según su DNI.
Dependencias:	Clientes que necesiten consultar tareas de alumnos expulsados.

---

### 2.3. Definición de catálogo de clientes que usan servicios externos.

Nombre del Servicio	Crear Tarea
Ruta URL	localhost:8084/guzpasen/conducta/tarea
Método	POST
Entrada: valores y tipo	Body (JSON con objeto Tarea)
Modo de Acceso	Postman
Formato Respuesta	JSON (Objeto Tarea creado)
Código Respuesta	201 CREATED
Descripción	Crea una nueva tarea en el sistema con los datos proporcionados en el body.
Dependencias	Clientes que necesiten registrar tareas (ej: frontend de profesores).

```
{  
    "titulo": "Realizar tarea de Matemáticas",  
    "descripcion": "Tarea relacionada con la implicación del alumno en la materia.",  
    "estado": "PENDIENTE",
```

```
"sancion": {  
    "id": 1  
}
```

## 2.4. Arquitectura y codificación del modelo.

## 2.5. Mapeo ORM.

## 2.6. Codificación y pruebas de los servicios REST.

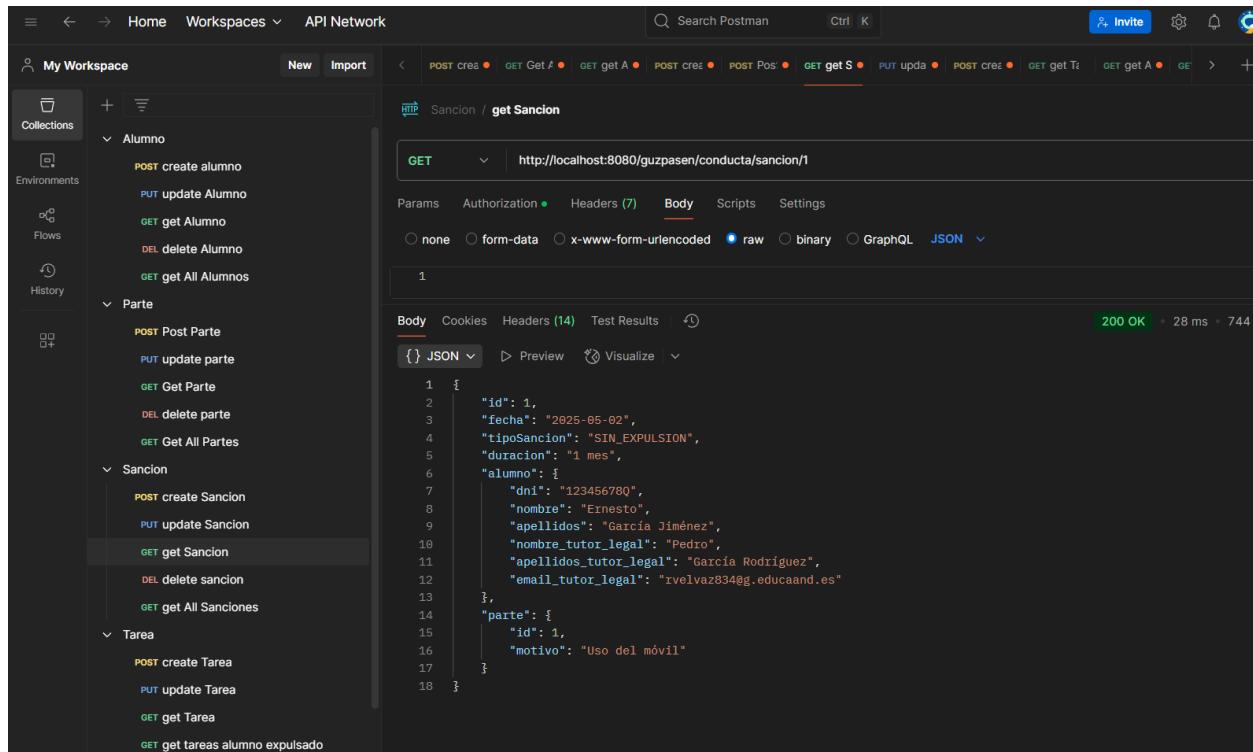
### 1. Crear Sancion:

The screenshot shows the Postman application interface. On the left, there's a sidebar with collections like 'Alumno', 'Parte', 'Sancion', and 'Tarea'. The 'Sancion' collection is expanded, showing methods: 'POST create Sancion', 'PUT update Sancion', 'GET get Sancion', 'DEL delete Sancion', and 'GET get All Sanciones'. The main area shows a POST request to 'http://localhost:8080/guzpasesn/conducta/sancion'. The 'Body' tab is selected, showing a JSON payload:

```
{  
  "id": 1,  
  "fecha": "2025-05-02",  
  "tipoSancion": "CON_EXPULSION_FUERA",  
  "duracion": "3 días",  
  "alumno": {  
    "dni": "12345678Q"  
  },  
  "parte": {  
    "id": 1  
  }  
}
```

The 'Test Results' section shows a successful response: '201 Created' with a response time of '1.43 s' and a size of '757 B'. There are tabs for Body, Cookies, Headers, and Test Results.

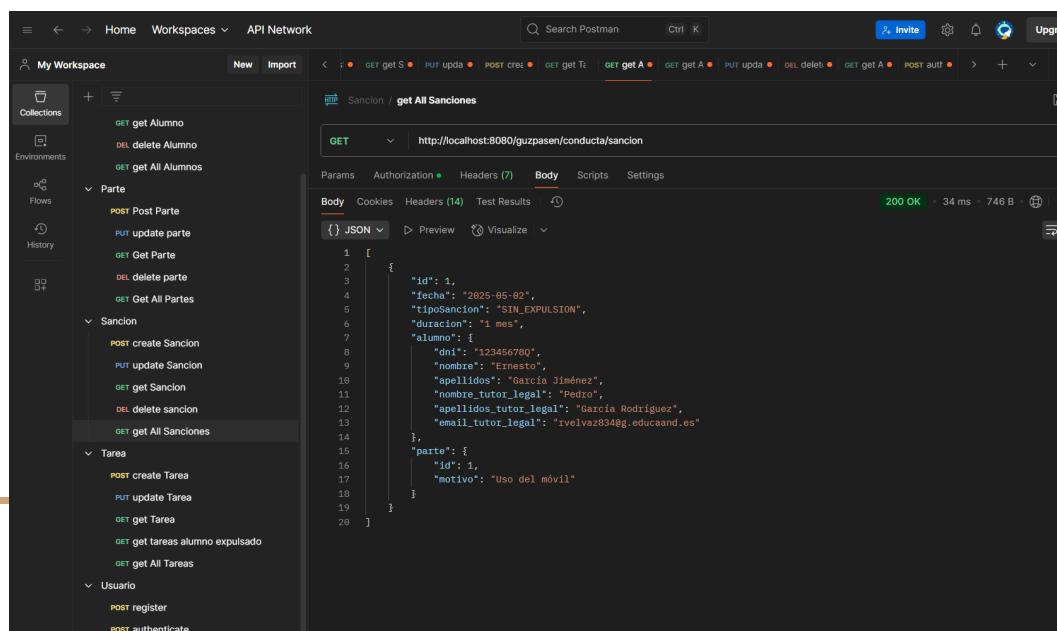
## 2. Obtener detalle sanción:



The screenshot shows the Postman interface with a collection named "My Workspace". The "Sancion" folder contains a "get Sancion" endpoint. A GET request is being made to `http://localhost:8080/guzpisen/conducta/sancion/1`. The response status is 200 OK, and the response body is a JSON object representing a single sanction record.

```
1 {  
2     "id": 1,  
3     "fecha": "2025-05-02",  
4     "tipoSancion": "SIN_EXPULSION",  
5     "duracion": "1 mes",  
6     "alumno": {  
7         "dni": "12345678Q",  
8         "nombre": "Ernesto",  
9         "apellidos": "García Jiménez",  
10        "nombre_tutor_legal": "Pedro",  
11        "apellidos_tutor_legal": "García Rodriguez",  
12        "email_tutor_legal": "rvelvaz834@educaand.es"  
13    },  
14    "parte": {  
15        "id": 1,  
16        "motivo": "Uso del móvil"  
17    }  
18 }
```

## 3. Obtener todas las sanciones:



The screenshot shows the Postman interface with a collection named "My Workspace". The "Sancion" folder contains a "get All Sanciones" endpoint. A GET request is being made to `http://localhost:8080/guzpisen/conducta/sancion`. The response status is 200 OK, and the response body is a JSON array containing multiple sanction records.

```
1 [  
2     {  
3         "id": 1,  
4         "fecha": "2025-05-02",  
5         "tipoSancion": "SIN_EXPULSION",  
6         "duracion": "1 mes",  
7         "alumno": {  
8             "dni": "12345678Q",  
9             "nombre": "Ernesto",  
10            "apellidos": "García Jiménez",  
11            "nombre_tutor_legal": "Pedro",  
12            "apellidos_tutor_legal": "García Rodriguez",  
13            "email_tutor_legal": "rvelvaz834@educaand.es"  
14        },  
15        "parte": {  
16            "id": 1,  
17            "motivo": "Uso del móvil"  
18        }  
19    }  
20 ]
```

#### 4. Actualizar sanción:

The screenshot shows the Postman interface with the following details:

- Collection:** My Workspace
- Request:** PUT /Sancion / update Sancion
- Method:** PUT
- URL:** http://localhost:8080/guzpasen/conducta/sancion/1
- Body:** JSON (raw)
- Body Content:**

```
1 {
2     "tipoSancion": "SIN_EXPULSION",
3     "duracion": "1 mes"
4 }
```
- Response:** 200 OK (24 ms, 744 B)

#### 5. Eliminar sanción:

The screenshot shows the Postman interface with the following details:

- Collection:** My Workspace
- Request:** DELETE /Sancion / delete sancion
- Method:** DELETE
- URL:** http://localhost:8080/guzpasen/conducta/sancion/1
- Body:** Raw (empty)
- Response:** 200 OK (33 ms, 382 B)

---

**2.7. Codificación y pruebas de los clientes REST.**

**2.8. JavaDoc proyecto.**

# SPRINT 4 PROYECTO INTEGRAL

## PROYECTO INTEGRAL

---

### 1. Integración

#### 1.1. Integración Frontend-Backend

Entrega realizada en “**Entrega: Integración Frontend-Backend (Entrega AMPLIADA hasta 30-5)**”

### 2. Securización

Securización implementado en el proyecto adjunto en la tarea mencionada.

---

## 3. Ejecución del plan de pruebas.

### 3.1. Pruebas Unitarias.

(No he implementado Jacoco ya que en IntelliJ no hace falta herramientas externas para ver la cobertura del código)

The screenshot shows the IntelliJ IDEA interface with the following details:

- Project View:** Shows the project structure with modules like "Script Seguimiento Alumnado SQL.sql" and "Sprint 2.pdf".
- Coverage View:** Shows code coverage statistics for the "GUZPASEN" class. The table includes columns for Element, Class, %, Method, %, Line, %, and Branch, %. A red box highlights the first row: "com.app.GUZPASEN" with values 100% (27/27), 98% (93/94), 99% (292/292), and 72% (71/98).
- Test Results View:** Shows the results of the "GUZPASEN" test suite. It displays 103 tests passed in 7 seconds. A red box highlights this message: "Tests passed: 103 of 103 tests - 7 sec 261ms". Below this, the terminal output shows Java agent loading and Spring Boot test context bootstrapping logs.

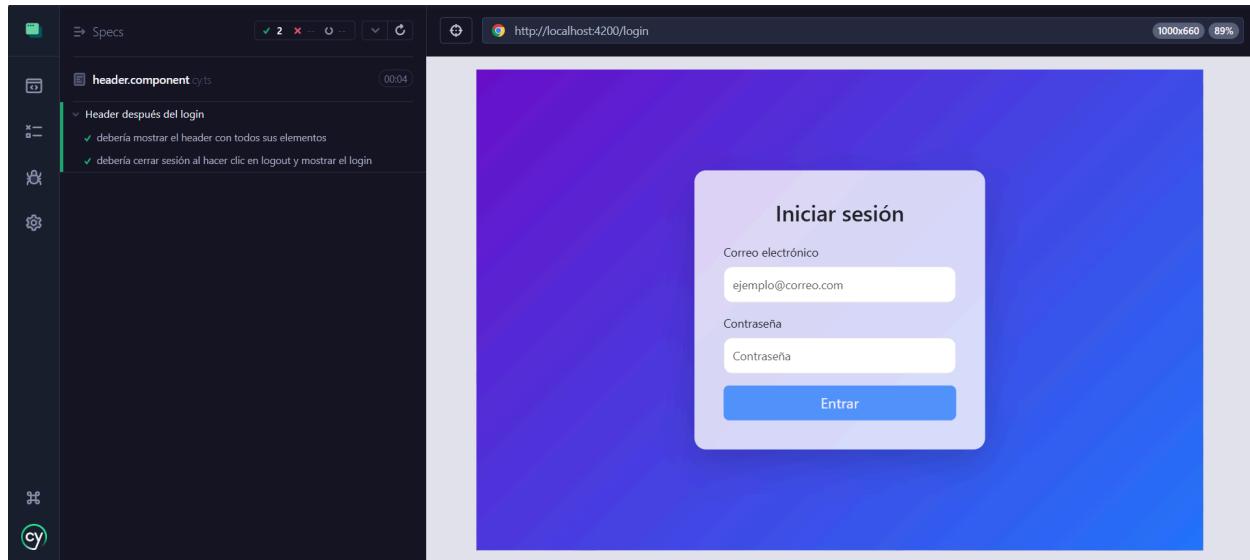
### 3.2. Pruebas de Integración.

#### 3.2.1. Login.

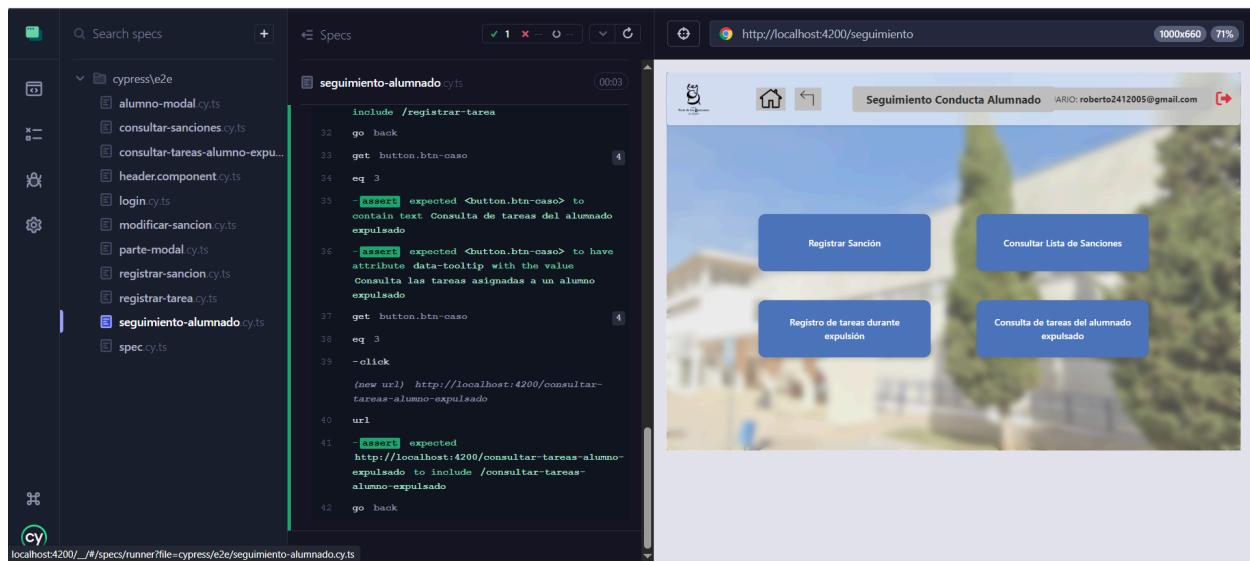
The screenshot shows the Cypress Test Runner interface and a browser preview side-by-side:

- Cypress Test Runner:** On the left, it shows a test spec named "login.cyts" with a single test case: "debería iniciar sesión y redirigir al seguimiento". The status of this test is green, indicating it has passed.
- Browser Preview:** On the right, a browser window displays the "Seguimiento Conducta Alumnado" application. The page has a header with a logo, a search bar, and a user dropdown. Below the header are four blue buttons: "Registrar Sanción", "Consultar Lista de Sanciones", "Registro de tareas durante expulsión", and "Consulta de tareas del alumnado expulsado".

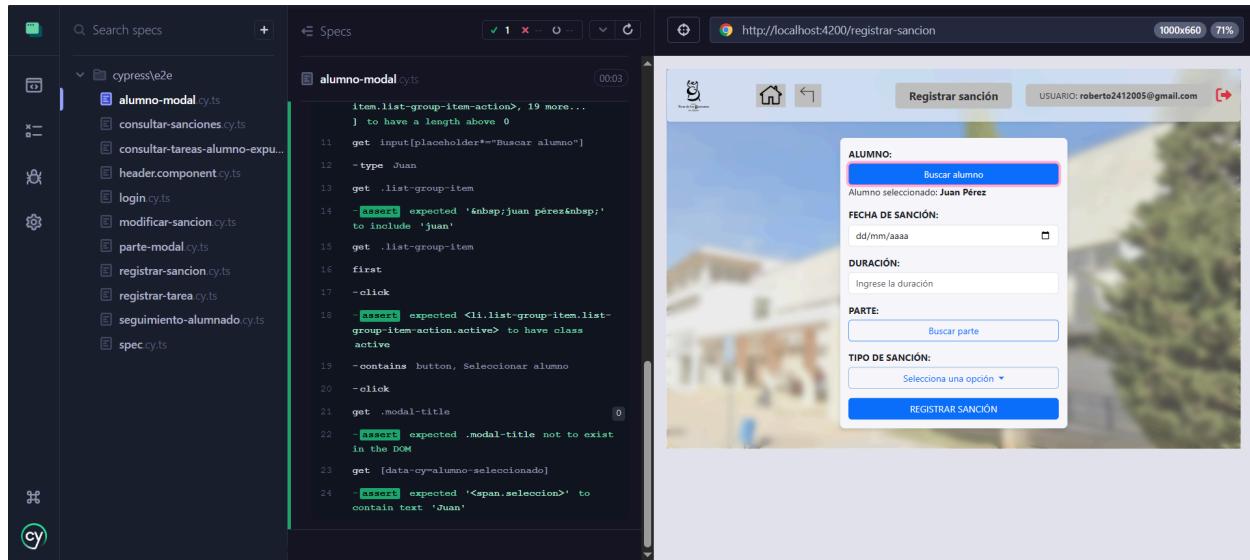
### 3.2.2. Header.



### 3.2.3. Página de Inicio (Seguimiento del alumnado).



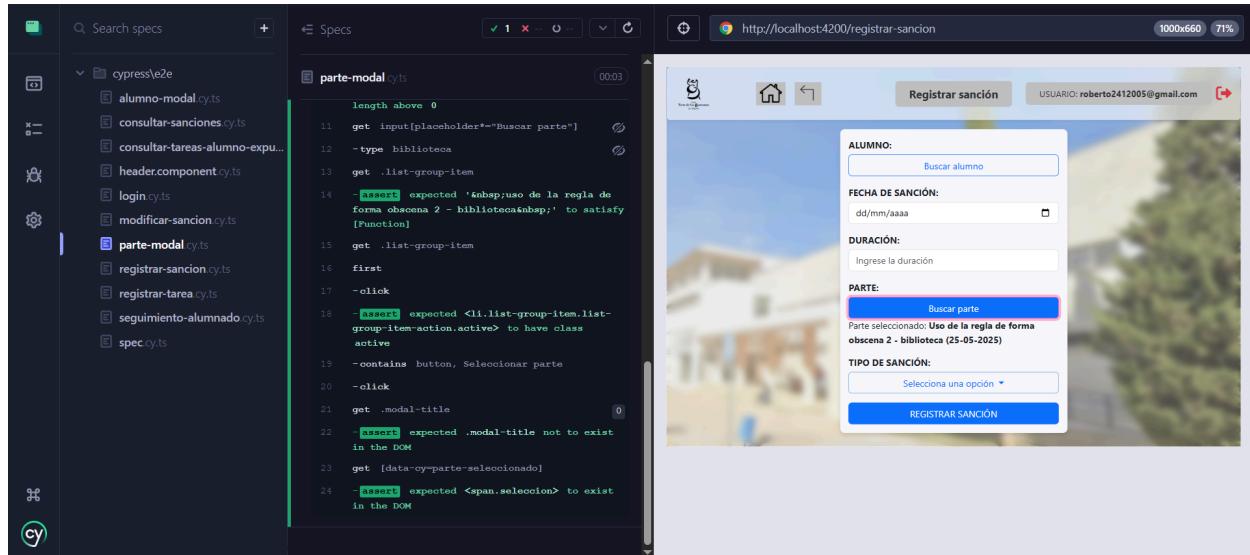
### 3.2.4. Modal de Alumno



The screenshot shows the Cypress Test Runner interface. On the left, a sidebar lists various test files under the 'cypress/e2e' directory. In the center, a code editor displays a file named 'alumno-modal.cy.ts'. The code contains a single test for the 'alumno-modal' component, which checks if a student can be selected from a dropdown list. The right side of the interface shows a browser window with the URL 'http://localhost:4200/registrar-sancion'. The browser displays a modal dialog titled 'ALUMNO:' with a search bar and a dropdown menu showing 'Alumno seleccionado: Juan Pérez'. Below the modal, the main page has fields for 'FECHA DE SANCIÓN:', 'DURACIÓN:', 'PARTE:', and 'TIPO DE SANCIÓN:'.

```
item.list-group-item-action>, 19 more...
] to have a length above 0
11 get input[placeholder="Buscar alumno"]
12 - type Juan
13 get .list-group-item
14 - assert expected 'nbsp;juan p  eznbsp;' to include 'juan'
15 get .list-group-item
16 first
17 - click
18 - assert expected <li.list-group-item.list-group-item-action.active> to have class active
19 - contains button, Seleccionar alumno
20 - click
21 get .modal-title
22 - assert expected .modal-title not to exist in the DOM
23 get [data-cy=alumno-seleccionado]
24 - assert expected <span.seleccion>' to contain text 'Juan'
```

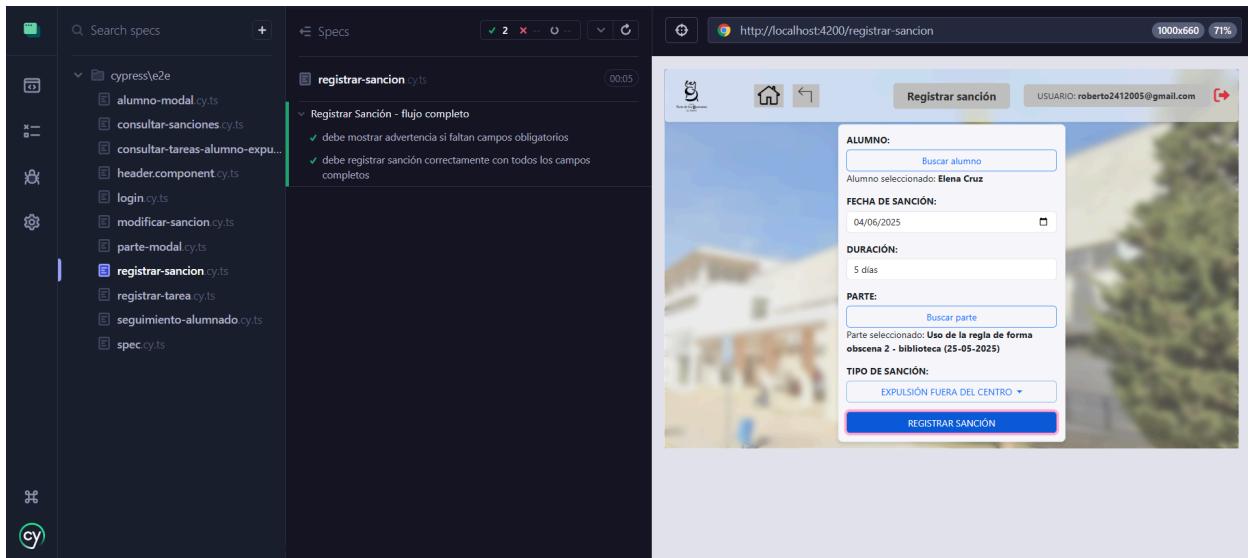
### 3.2.5. Modal de Parte.



The screenshot shows the Cypress Test Runner interface. On the left, a sidebar lists various test files under the 'cypress/e2e' directory. In the center, a code editor displays a file named 'parte-modal.cy.ts'. The code contains a single test for the 'parte-modal' component, which checks if a part can be selected from a dropdown list. The right side of the interface shows a browser window with the URL 'http://localhost:4200/registrar-sancion'. The browser displays a modal dialog titled 'ALUMNO:' with a search bar and a dropdown menu. Below the modal, the main page shows a message indicating that 'Parte seleccionado: Uso de la regla de forma obscena 2 - biblioteca (25-05-2025)' has been selected. The other fields in the modal and main page remain the same as in the previous screenshot.

```
length above 0
11 get input[placeholder="Buscar parte"]
12 - type biblioteca
13 get .list-group-item
14 - assert expected 'nbsp;uso de la regla de forma obscena 2 - bibliotecanbsp;' to satisfy [Function]
15 get .list-group-item
16 first
17 - click
18 - assert expected <li.list-group-item.list-group-item-action.active> to have class active
19 - contains button, Seleccionar parte
20 - click
21 get .modal-title
22 - assert expected .modal-title not to exist in the DOM
23 get [data-cy=parte-seleccionado]
24 - assert expected <span.seleccion> to exist in the DOM
```

### 3.2.6. Registrar sanción.

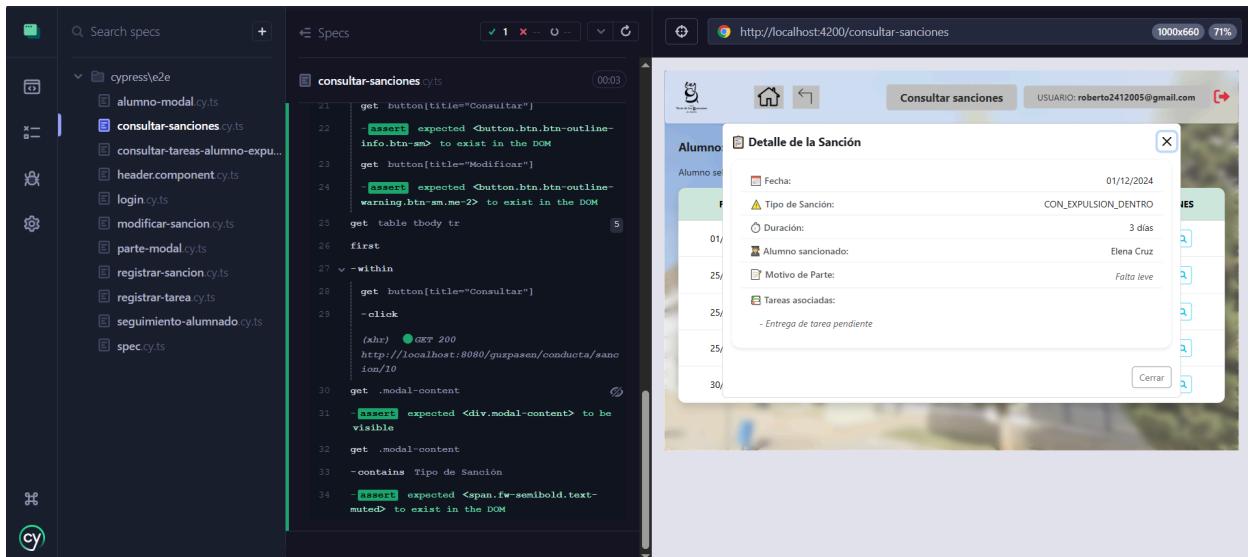


The screenshot shows the Cypress Test Runner interface. On the left, there's a sidebar with icons for file operations and a search bar labeled "Search specs". The main area displays a list of test files under "cypress/e2e". One file, "registrar-sancion.cy.ts", is expanded, showing its contents:

```
describe('registrar-sancion cyts', () => {
  it('Registrar Sanción - flujo completo', () => {
    // Test logic here
  })
})
```

On the right, a browser window is open at the URL <http://localhost:4200/registrar-sancion>. The page has a header "Registrar sanción" and a user "roberto2412005@gmail.com". It contains fields for "ALUMNO:" (with a dropdown for "Elena Cruz"), "FECHA DE SANCIÓN:" (set to "04/06/2024"), "DURACIÓN:" (set to "5 días"), "PARTE:" (with a dropdown for "Uso de la regla de forma obscena 2 - biblioteca (25-05-2025)"), and "TIPO DE SANCIÓN:" (set to "EXPULSIÓN FUERA DEL CENTRO"). A large blue button at the bottom right says "REGISTRAR SANCIÓN".

### 3.2.7. Consultar sanciones y detalle de sanción.



The screenshot shows the Cypress Test Runner interface. On the left, there's a sidebar with icons for file operations and a search bar labeled "Search specs". The main area displays a list of test files under "cypress/e2e". One file, "consultar-sanciones.cy.ts", is expanded, showing its contents:

```
describe('consultar-sanciones cyts', () => {
  it('Consultar sanciones', () => {
    // Test logic here
  })
})
```

On the right, a browser window is open at the URL <http://localhost:4200/consultar-sanciones>. The page has a header "Consultar sanciones" and a user "roberto2412005@gmail.com". It shows a table of punishment details. A modal window titled "Detalle de la Sanción" is open over the table, displaying specific information for a punishment:

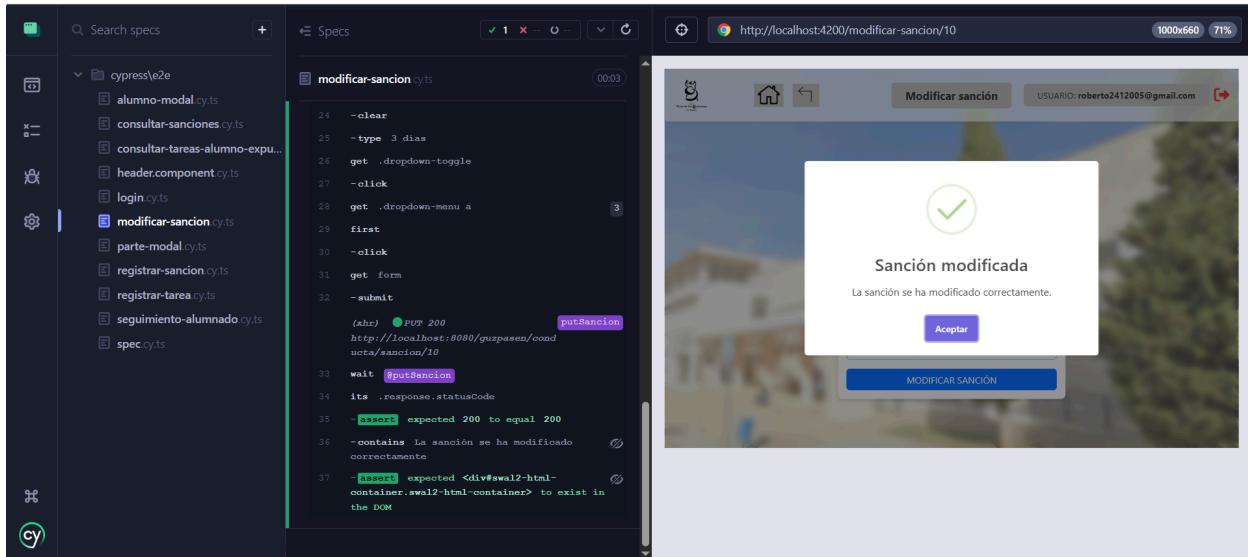
Alumno	Fecha	Tipo de Sanción	Duración
Elena Cruz	01/12/2024	CON_EXPULSION_DENTRO	3 días

The modal also lists other details:

- Alumno sancionado: Elena Cruz
- Motivo de Parte: Falta leve
- Tareas asociadas:
  - Entrega de tareas pendiente

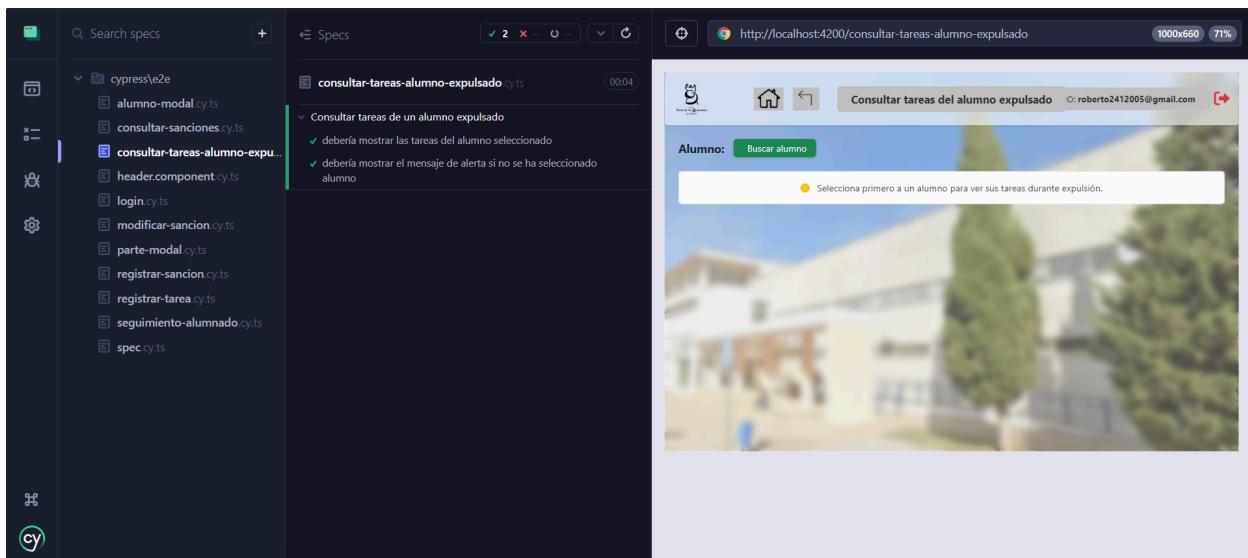
A "Cerrar" button is visible in the bottom right corner of the modal.

### 3.2.8. Modificar sanción.



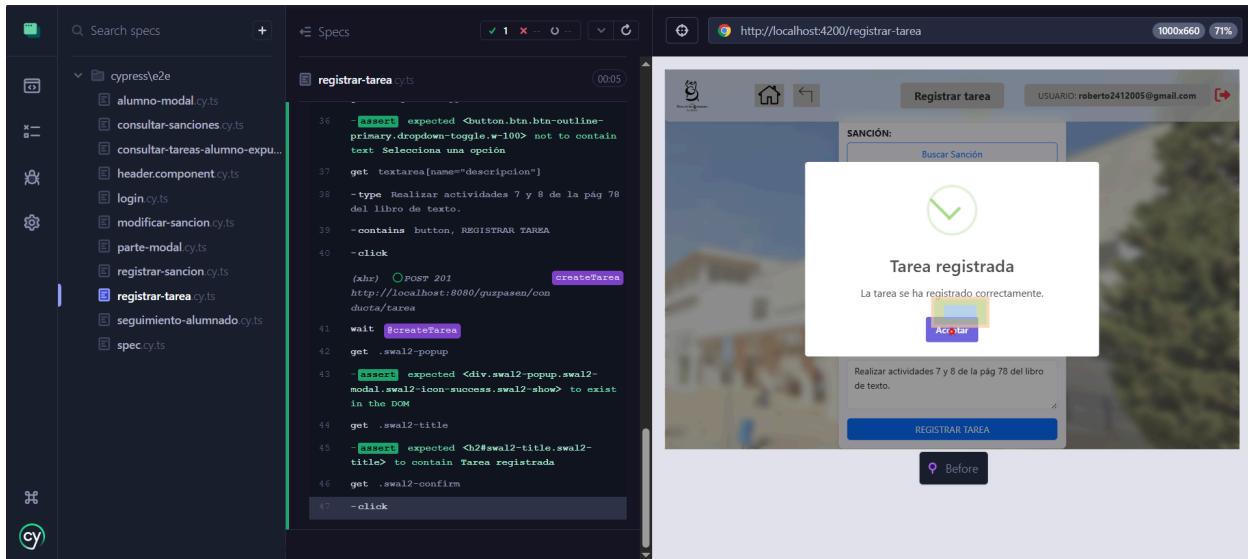
The screenshot shows the Cypress Test Runner interface. On the left, a sidebar lists various test files under the 'cypress/e2e' folder. The main area displays a test spec named 'modificar-sancion.cy.ts'. The code within the spec performs several actions: it clears input fields, types '3 días' into one, and interacts with dropdown menus. It then submits a form via an XMLHttpRequest ('PUT 200') to the URL 'http://localhost:8080/guispasen/consultar-tarea-expulsado/10'. After submission, it waits for a response and asserts that the status code is 200. Finally, it contains a check for the message 'La sanción se ha modificado correctamente' and asserts that a specific DOM element exists. To the right of the code editor is a browser window showing the application's confirmation modal. The modal has a green checkmark icon and the text 'Sanción modificada'. Below it, a message says 'La sanción se ha modificado correctamente.' There are 'Aceptar' and 'MODIFICAR SANCIÓN' buttons at the bottom. The browser title bar shows the URL 'http://localhost:4200/modificar-sancion/10' and the status '1000x660 71%'. The top right corner of the browser window also shows '1000x660 71%'.

### 3.2.9. Consultar todas las tareas durante expulsión de un alumno.



The screenshot shows the Cypress Test Runner interface. The sidebar on the left lists test files. The main area shows a test spec named 'consultar-tareas-alumno-expulsado.cy.ts'. This spec contains a single test case titled 'Consultar tareas de un alumno expulsado'. The test case includes two assertions: 'debería mostrar las tareas del alumno seleccionado' and 'debería mostrar el mensaje de alerta si no se ha seleccionado alumno'. To the right is a browser window displaying the application's search interface. The title bar shows the URL 'http://localhost:4200/consultar-tareas-alumno-expulsado' and the status '1000x660 71%'. The browser title bar also shows the status '1000x660 71%' and the user 'roberto2412005@gmail.com'. The page itself has a header 'Consultar tareas del alumno expulsado'. Below it is a search field labeled 'Alumno:' with a placeholder 'Buscar alumno'. A note below the search field says 'Selecione primero a un alumno para ver sus tareas durante expulsión.' The browser title bar shows the URL 'http://localhost:4200/consultar-tareas-alumno-expulsado' and the status '1000x660 71%'. The top right corner of the browser window also shows '1000x660 71%'.

### 3.2.10. Registrar Tarea.



The image shows the Cypress Test Runner interface. On the left, there's a sidebar with icons for file operations and a search bar labeled 'Search specs'. Below it is a tree view of test files under 'cypress/e2e'. A specific file, 'registrar-tarea.cy.ts', is selected and expanded, showing its code. The code is a Cypress script that performs a POST request to 'http://localhost:8080/guispasen/conducta/tareas' with a payload containing 'createTarea'. It then waits for a swal2-popup to appear and asserts that it contains the text 'Tarea registrada'. On the right, there's a screenshot of a web browser window titled 'Registrar tarea'. The URL is 'http://localhost:4200/registrar-tarea'. The page has a header with a user icon and the text 'SANCIÓN: Buscar Sanción'. Below the header is a modal window with a green checkmark icon and the text 'Tarea registrada'. Underneath the modal, a message says 'La tarea se ha registrado correctamente.' and a blue button labeled 'Aceptar'. At the bottom of the page, there's a blue button labeled 'REGISTRAR TAREA' and a small 'Before' button.

```
36   assert expected <button.btn.btn-outline-primary.dropdown-toggle.w-100> not to contain text 'Selecciona una opción'
37   get textarea[name='descripcion']
38   -type Realizar actividades 7 y 8 de la pág 78 del libro de texto.
39   -contains button, REGISTRAR TAREA
40   -click
41   (xhr) POST 201 createTarea
        http://localhost:8080/guispasen/conducta/tareas
42   wait #createTarea
43   get swal2-popup
44   assert expected <div.swal2-popup.swal2-modal.swal2-icon-success.swal2-show> to exist in the DOM
45   get .swal2-title
46   assert expected <h2#swal2-title.swal2-title> to contain 'Tarea registrada'
47   get swal2-confirm
48   -click
```

### 3.3. Pruebas de Aceptación del Usuario (UAT).

Mismo vídeo realizado para la demostración de la integración backend - frontend, donde se hace una demo de la aplicación a nivel completo.