Desarrollo de pruebas unitarias para el componente front-end del software AcademiAPP para la empresa Webcloster S.A.S

Pasantía en Modalidad de Convenio

Samuel Ignacio Gómez Moreno

Universidad del Valle
Facultad de ingeniería
Escuela de ingeniería de sistemas y computación
Tuluá
2025

Desarrollo de pruebas unitarias para el componente front-end del software AcademiAPP para la empresa Webcloster S.A.S

Pasantía en Modalidad de Convenio

Samuel Ignacio Gómez Moreno
Código 201958829
gomez.samuel@correounivalle.edu.co

Director
Royer David Estrada Esponda. Msc
royer.estrada@correounivalle.edu.co

Universidad del Valle
Facultad de ingeniería
Escuela de ingeniería de sistemas y computación
Tuluá
2025

Trabajo de grado presentado por
Samuel Ignacio Gómez Moreno
Como requisito parcial para la obtención del título de Ingeniero de Sistemas

Royer David Estrada Esponda	Mauricio López Benitez
Director	Jurado

En primer lugar, agradezco a Dios por permitirme avanzar en mi formación con entusiasmo, determinación y pasión.

A mi familia, por su apoyo incondicional, su cariño y la confianza que me han dado para llegar a este logro.

A mis docentes por ser consejeros y guías en mi crecimiento personal y profesional.

A mis compañeros de estudio, por convertir este viaje académico en una experiencia llena de aprendizajes y alegrías compartidas.

Tabla de Contenido

Lista de Tablas

Lista de Figuras

Resumen.

Este proyecto se enfoca en el desarrollo de pruebas unitarias y la mejora de módulos dentro de AcademiApp, una aplicación en línea creada por Webcloster SAS para la gestión académica y administrativa en instituciones educativas. AcademiApp permite administrar datos, consultar horarios de aulas, evaluar desempeños y optimizar diversos procesos dentro del entorno académico.

Con el objetivo de mejorar la estabilidad y fiabilidad del sistema, se implementaron pruebas unitarias en múltiples módulos de la aplicación. Estas pruebas permiten detectar y corregir errores de manera temprana, evitando que afecten otras partes del sistema. Además, el proyecto incluye la adición de nuevas funcionalidades y la incorporación de requerimientos específicos en módulos ya implementados, asegurando una evolución constante de la plataforma según las necesidades de los usuarios.

Para llevar a cabo el desarrollo, se aplicó una metodología de investigación aplicada, complementada con el marco ágil Scrum. Esto facilitó un proceso iterativo y organizado, permitiendo una integración eficiente de mejoras y asegurando que los cambios realizados contribuyan al correcto funcionamiento de la aplicación.

En última instancia, este trabajo busca fortalecer a AcademiApp como una solución confiable y robusta para la gestión académica, respondiendo a la creciente necesidad de las instituciones de contar con herramientas tecnológicas eficientes para administrar sus procesos de manera óptima.

Introducción.

En los últimos años, el software ha adquirido un papel fundamental en varias áreas [1], entre ellas, la gestión académica y administrativa de las universidades, facilitando la automatización de procesos y mejorando la eficiencia en la administración de datos. La digitalización de estos procedimientos permite optimizar la asignación de recursos, reducir la carga operativa del personal administrativo y mejorar la experiencia educativa de estudiantes y docentes. En un entorno académico en constante evolución, donde la demanda de acceso rápido y eficiente a la información es cada vez mayor, contar con herramientas tecnológicas que centralicen y optimicen la gestión académica se ha vuelto una necesidad. Además, la integración de soluciones basadas en software permite garantizar la trazabilidad de la información y asegurar una toma de decisiones más informada dentro de las instituciones educativas. En este contexto, AcademiApp, una aplicación desarrollada por Webcloster SAS, se presenta como una solución tecnológica que permite optimizar la gestión de información en entornos académicos, ofreciendo herramientas para la administración de aulas, evaluación de desempeño y otros procesos clave [2].

En este contexto, las instituciones de educación superior enfrentan el desafío de adoptar tecnologías que les permitan gestionar eficazmente sus operaciones administrativas y académicas [3]. La implementación de plataformas digitales no solo optimiza la administración interna, sino que también mejora la comunicación entre docentes, estudiantes y personal administrativo [4]. Sistemas de gestión académica como AcademiApp facilitan la organización de actividades clave, como la asignación de horarios, el control del rendimiento estudiantil y la administración de recursos educativos. Sin embargo, para que estas soluciones sean realmente efectivas, es fundamental garantizar su estabilidad, confiabilidad y escalabilidad a través de procesos rigurosos de prueba y mantenimiento continuo.

Con base en lo anterior, se desarrolló este trabajo en el marco del programa de pasantías de Webcloster SAS en colaboración con la Universidad del Valle, con el objetivo de fortalecer la estabilidad y funcionalidad de AcademiApp. Durante el desarrollo del proyecto, se aplicaron pruebas unitarias para mejorar la confiabilidad de varios puntos claves del sistema, mejorando su calidad y asegurando el correcto funcionamiento de los módulos evaluados. Además, se incorporaron nuevas funcionalidades y se realizaron ajustes en módulos existentes, optimizando así la experiencia de los usuarios y garantizando una gestión académica más eficiente y precisa.

Capítulo 1

Contexto y objetivos.

1.1. Formulación del problema.

¿Cómo pueden los conceptos y técnicas de ingeniería de sistemas, como el desarrollo de pruebas unitarias y la mejora de módulos en la aplicación académica en línea AcademiApp, contribuir a la estabilidad y fiabilidad de los sistemas de gestión educativa, permitiendo la detección y corrección temprana de errores, la optimización de procesos administrativos y la evolución continua de la plataforma según las necesidades de los usuarios, con el objetivo de fortalecer la eficiencia y confiabilidad de las herramientas tecnológicas en el mercado?

1.2. Descripción del problema

Webcloster S.A.S. identificó la necesidad latente de una gestión de información más detallada en las universidades en muchas aspectos de los procesos que se llevan a cabo, abarcando tanto operaciones internas como servicios para la comunidad académica. Estos incluyen la matrícula de estudiantes, la gestión de dificultades académicas, la administración de proyectos de grado, reasignaciones académicas y administración de almuerzos.

Para responder a estos requerimientos, la aplicación AcademiApp ofrece una solución eficiente que optimiza la ejecución de los procesos académicos. Sin embargo, Webcloster S.A.S. también detectó una completa carencia de pruebas en el sistema, lo que ponía en riesgo su estabilidad y fiabilidad. Ante esta situación, se propuso la implementación de pruebas unitarias de manera organizada, asegurando un desarrollo estructurado que permita mejorar la calidad del sistema al momento de desplegar e integrar nuevas funcionalidades sin descuidar las actualizaciones a las ya existentes, siendo estas necesarias para la adaptación a la demanda del mercado.

1.3. Objetivos

1.3.1. Objetivo general

Implementar pruebas unitarias enfocadas en el Front-End de la aplicación AcademiApp para la empresa Webcloster SAS.

1.3.2. Objetivos específicos

1. Analizar los módulos de la aplicación general con ausencia de pruebas.
2. Diseñar pruebas unitarias para los módulos sin documentación ni sistematización suficiente.
3. Desarrollar los casos de prueba priorizados por la empresa.
4. Generar informes de pruebas estáticas y de cobertura.

1.4. Antecedentes

Diversas instituciones académicas incorporan herramientas tecnológicas como sistemas informáticos para la gestión de sus procesos, como lo son Uninorte.co, la cual es una aplicación móvil diseñada para estudiantes, que ofrece acceso desde sus dispositivos a diversas funcionalidades destinadas a hacer la experiencia universitaria más efectiva, eficiente y entretenida, incluyendo horarios, calificaciones, notificaciones, disponibilidad de salas, opiniones, calendario, redes sociales y más.

Por otro lado, UPB Móvil en su primera etapa se enfocó en ofrecer una agenda para uso de los estudiantes, con el fin de programar sus trabajos de clase, además de proporcionar una manera más directa y centralizada de acceder a los sistemas de información en línea de la universidad, permitiéndoles revisar su estado académico actual, así como sus calificaciones.

Asimismo, la aplicación Unisabana, se centra en facilitar la indagación académica y administrativa, dirigida a la comunidad relevante de la universidad y una parte de ella, a externos que deseen conocer las ofertas educativas de la misma.

Y, por último, se tiene a App Central, una aplicación que ofrece acceso a noticias sobre el calendario actual de varios departamentos de la universidad,entre ellos, el de Estudios Musicales, Arte Dramático, Comunicación Social y Periodismo, entre otros. Además, permite solicitar garantías de procesos actuales que esté realizando un usuario y realizar diligenciamientos de preinscripción con la ventaja de poder llevarlos a cabo de manera virtual, removiendo completamente el requerimiento físico de estar en las oficinas de la universidad, además ofrece la posibilidad de tener acceso a una lista de docentes para consultar sus áreas de interés y disponibilidad para tutorías de una manera más directa y centralizada.

A continuación, se presenta un resumen, en el que se exponen los hallazgos clave de la investigación realizada sobre sistemas más representativos similares a AcademiAPP, capturando las ventajas y desventajas que se caracterizaron en cada uno.

Aplicación	**Ventajas**	**Desventajas**
Uninorte.co	Plataforma con foro de discusión estudiantil, programación eficiente de reuniones y gestión de contactos académicos.	Presenta inestabilidad bajo alta demanda, eliminación de herramientas clave como la calculadora promedio y disponibilidad exclusiva en dispositivos móviles, limitando su accesibilidad.
UPB Móvil	Incluye sección de noticias institucionales, agenda académica integrada, biblioteca digital y una interfaz recientemente optimizada.	Problemas de rendimiento ante alta concurrencia, falta de notificaciones automáticas para actualizaciones importantes y restricción a dispositivos móviles.
App Unisabana	Facilita la programación de reuniones, permite identificación mediante código QR y mejora la interacción con el cuerpo docente.	Errores frecuentes en el servidor que impiden el acceso, fallos en la entrega y recepción de mensajes con los profesores y disponibilidad exclusiva en dispositivos móviles.
App Central	Proporciona acceso a calificaciones, horarios y aulas; permite a los docentes gestionar cursos y consultar ubicaciones, y ofrece a los egresados acceso a títulos electrónicos.	Incompatibilidad con dispositivos de gama baja y limitación a plataformas móviles.

Tabla 1.1: Comparación de aplicaciones similares

Los sistemas académicos evaluados presentan diversas funcionalidades orientadas a mejorar la experiencia de los estudiantes y docentes en la gestión de información universitaria. Aplicaciones como Uninorte.co y UPB Móvil destacan por ofrecer herramientas de comunicación, como foros y noticias institucionales, mientras que otras como App Unisabana y App Central se enfocan en la identificación estudiantil y la gestión académica. Sin embargo, un patrón común entre estos sistemas es la falta de disponibilidad multiplataforma, ya que la mayoría solo funcionan en dispositivos móviles, limitando su accesibilidad.

Además, se evidencia un problema recurrente relacionado con la estabilidad del sistema, ya que varias aplicaciones experimentan fallos cuando hay un alto flujo de usuarios, lo que sugiere deficiencias en la escalabilidad y en el manejo de concurrencia. Otro aspecto importante es la falta de ciertas notificaciones clave, como la actualización de notas o eventos académicos de interés general, lo que afecta la experiencia del usuario y la eficiencia de estos sistemas. Esto demuestra la necesidad de mejorar la optimización del código, la infraestructura de servidores y la usabilidad de estas aplicaciones para garantizar un mejor desempeño y satisfacer las necesidades de los usuarios de manera más efectiva.

En conclusión, teniendo en cuenta que un gran porcentaje de sistemas académicos de este tipo suelen presentar problemas de estabilidad y rendimiento, la implementación de pruebas unitarias en AcademiAPP resulta fundamental para mejorar la calidad del software y evitar errores comunes antes de su despliegue. Las pruebas unitarias permitirían verificar individualmente cada componente del sistema, asegurando que funciones críticas operen correctamente antes de ser integradas en el entorno de producción. Además, al aplicar técnicas como pruebas de integración y análisis de cobertura de código, se podría reducir la probabilidad de fallos inesperados y controlar casos esenciales, mejorando la escalabilidad y estabilidad del sistema.

Otro beneficio clave es la detección temprana de errores en el código, lo que facilita un desarrollo más ágil y una reducción en los costos de sostenimiento, por estas razones, se opta por la ejecución de pruebas unitarias en AcademiAPP para garantizar una mejor experiencia para estudiantes y docentes, sin comprometer el mantenimiento ni la incorporación de nuevas funcionalidades a prioridad de la empresa.

1.5. Información sobre los capítulos

En la siguiente tabla se muestra una corta descripción de cada capítulo del documento.

Capítulo	Descripción
Capítulo 2: Marco de referencia	Se explican los conceptos necesarios para comprender el problema y su solución.
Capítulo 3: Alcance del proyecto	En este capítulo se define el alcance del proyecto.
Capítulo 4: Metodologías	En este capítulo se presentan las metodologías empleadas en la investigación, pruebas unitarias, desarrollo de módulos y gestión del proyecto.
Capítulo 5: Desarrollo del proyecto	En este capítulo se caracterizan aspectos de la implementación de las pruebas unitarias y de los módulos desarrollados.
Capítulo 6: Conclusiones y proyecciones del proyecto	En este capítulo se resumen los logros del proyecto, su impacto y posibles mejoras o ampliaciones futuras.

Tabla 1.2: Estructura de capítulos

Capítulo 2

Marco de Referencia.

En este capítulo se presentan los conceptos fundamentales para comprender la aplicación AcademiAPP, su implementación de pruebas unitarias, el desarrollo de funcionalidades y la gestión general del proyecto.

Se recomienda que el lector de este documento cuente con conocimientos previos en:

- **Arquitectura desacoplada en aplicaciones web**, considerando que AcademiAPP utiliza Angular en el front-end y Laravel(PHP) en el back-end, ambos en repositorios independientes.
- **Desarrollo de API REST**, dado que la aplicación gestiona la comunicación entre frontend y backend mediante este enfoque..
- **Herramientas para el desarrollo web**, incluyendo Trello para la gestión de tareas, Scrum como metodología ágil, y patrones como MVC.

A continuación, se definen los términos clave que sustentan el desarrollo del proyecto, abordando conceptos de pruebas unitarias, pruebas de cobertura y enfoques utilizados en el desarrollo del proyecto de grado.

2.1. Marco Teórico

2.1.1. Metodología

¿Qué es una metodología? De acuerdo con Roberto Sampieri [5], una metodología se considera al conjunto de métodos utilizados en una investigación científica con un objetivo en especificado, exteriorizando los métodos utilizados para la obtención de datos relativos al fin. Por otro lado, según Carlos Collado [5], un método se describe como "el conjunto de pasos ordenados y sistemáticos que se siguen para alcanzar un objetivo específico en una investigación científica".

2.1.2. Aplicación Web

Una aplicación web es un software alojado en un servidor y accesible mediante un navegador, sin necesidad de instalación en el dispositivo del usuario. Su desarrollo se basa en tecnologías como HTML, CSS y JavaScript, permitiendo desde sitios informativos hasta plataformas interactivas avanzadas.

Características de una aplicación web: Las aplicaciones web poseen diversas características que las hacen versátiles y beneficiosas. Algunas de ellas son:

- **Accesibilidad**: Se puede acceder a ellas desde cualquier dispositivo con conexión a internet.
- **Escalabilidad**: Permiten aumentar sus capacidades sin afectar su rendimiento.
- **Simplicidad para el usuario**: Suelen ser intuitivas y fáciles de usar.
- **Automatización y seguridad**: Implementan mecanismos para garantizar la integridad y protección de los datos.
- **Almacenamiento eficiente**: Pueden gestionar grandes volúmenes de información optimizando el uso de recursos.

Funcionamiento de una aplicación web: Las aplicaciones web se basan en una arquitectura cliente-servidor, lo que significa que existe una comunicación constante entre el navegador del usuario (cliente) y un servidor remoto que procesa las solicitudes y entrega las respuestas. Esta arquitectura permite que las aplicaciones puedan ser ejecutadas sin necesidad de instalación local, facilitando el acceso desde múltiples dispositivos con conexión a internet.

El código de una aplicación web se divide en dos partes fundamentales: los scripts del lado del cliente y los scripts del lado del servidor. El **lado del cliente**, también conocido como *frontend*, es responsable de la interfaz gráfica con la que interactúa el usuario. Esta parte está compuesta principalmente por tecnologías como HTML, CSS y JavaScript, y permite que la aplicación responda a eventos, visualice datos y genere una experiencia interactiva. En aplicaciones modernas, frameworks como Angular permiten desarrollar interfaces dinámicas, desacopladas del servidor y con una lógica de presentación robusta.

Por otro lado, el **lado del servidor** o *backend*, es el encargado de la lógica de negocio, la conexión a bases de datos y el manejo de las peticiones enviadas por el cliente. Tecnologías como Laravel permiten construir APIs RESTful eficientes, seguras y bien estructuradas, facilitando la interacción entre cliente y servidor. En este contexto, cuando el usuario realiza una acción en la interfaz, esta genera una solicitud HTTP que es enviada al servidor. Este procesa la solicitud, realiza las operaciones necesarias (como consultar o modificar datos) y devuelve una respuesta al cliente, que posteriormente se muestra en la interfaz.

Además, este tipo de arquitectura permite adoptar diferentes estrategias de renderizado. Una de ellas es el *renderizado del lado del servidor* (SSR), donde el servidor genera el contenido HTML y lo envía completo al cliente. Otra opción es el *renderizado del lado del cliente* (CSR), donde el cliente interpreta los datos y genera dinámicamente la interfaz, descargando solo la estructura base de la aplicación desde el servidor.

> "El **script del lado del cliente** se encarga de la funcionalidad de la interfaz de usuario, como los botones y los menús. Cuando el usuario final hace clic en el enlace de la aplicación web, el navegador web carga el script y renderiza los elementos gráficos y el texto para la interacción del usuario. Las acciones se dirigen al servidor como una solicitud del cliente. El **script del lado del servidor** se encarga de procesar los datos y las solicitudes del cliente que pueden ser: recibir más datos, editarlos o guardar nuevos. En algunos casos, el servidor completa la solicitud de datos y envía la página HTML completa al cliente. Esto se llama ≪renderizado del lado del servidor≫" [6].

2.1.3. Datos e información

Los datos pueden definirse como representaciones simbólicas de hechos, acciones o eventos. Por sí solos, no aportan un significado concreto hasta que son procesados. La información, en cambio, se refiere al resultado del procesamiento, estructuración o interpretación de esos datos, de manera que adquieran sentido y relevancia para el receptor. En este sentido, los datos actúan como materia prima para la generación de información útil y significativa [7].

Para que la información pueda considerarse valiosa dentro de un sistema, debe cumplir con ciertas características fundamentales:

- **Significativa:** Debe tener sentido para el usuario que la recibe.
- **Conveniente:** Debe responder a una necesidad concreta.
- **Entera:** No debe estar incompleta ni fragmentada.
- **Apropiada:** Debe estar relacionada con el propósito específico del usuario.
- **Oportuna:** Debe entregarse en el momento adecuado.
- **Detallada:** Debe tener el nivel de profundidad necesario.
- **Inteligible:** Debe ser comprensible y clara para quien la recibe.

2.1.4. Desarrollo de software

En el ámbito de las tecnologías de la información, el desarrollo de software constituye una disciplina compleja que abarca múltiples fases interrelacionadas, orientadas a la concepción, construcción, despliegue y sostenimiento de soluciones informáticas. Estas soluciones, comúnmente denominadas programas o aplicaciones, representan el componente

lógico indispensable para la operación de los sistemas de cómputo, dado que contienen conjuntos estructurados de instrucciones capaces de gestionar el comportamiento de los dispositivos ante situaciones diversas. Es importante resaltar que, a diferencia del hardware, el software posee la capacidad de ejecutarse en distintas plataformas tecnológicas, lo cual le otorga un alto grado de versatilidad y portabilidad entre sistemas con configuraciones heterogéneas.

Podemos clasificar el software en diferentes distinciones, siendo las más comunes:

1. **Software del sistema**: Este tipo de software cumple una función esencial al proporcionar las bases necesarias para la operatividad y usabilidad de los sistemas de cómputo. Dentro de esta categoría se encuentran los sistemas operativos, tales como Windows, macOS y Linux, los cuales gestionan los recursos del sistema y permiten la interacción entre el usuario y el hardware. Asimismo, se incluyen en esta clasificación diversas herramientas orientadas a la administración de componentes físicos, unidades de almacenamiento y servicios fundamentales, cuya adecuada gestión garantiza el correcto funcionamiento del entorno informático.

2. **Software de programación**: El software de desarrollo o de programación está orientado a facilitar la labor de los programadores en la creación, modificación y optimización de programas informáticos. Este conjunto de herramientas especializadas incluye editores de texto, compiladores, enlazadores, depuradores, y entornos de desarrollo integrados (IDE), los cuales permiten escribir código, identificar errores, realizar pruebas y mejorar el rendimiento de las aplicaciones. Su propósito principal es proporcionar un entorno eficiente y funcional que favorezca la productividad y calidad del software desarrollado.

3. **Software de aplicación**: El software de aplicación comprende un conjunto de programas diseñados para asistir al usuario en la ejecución de tareas específicas en computadores y dispositivos móviles. Este tipo de software abarca herramientas de productividad, como las suites ofimáticas (por ejemplo, Microsoft Office), aplicaciones para la gestión y análisis de datos, reproductores multimedia, soluciones de seguridad como los antivirus, así como una amplia variedad de aplicaciones web y móviles orientadas a la comunicación, el entretenimiento, las redes sociales y el comercio electrónico. Su función principal es brindar funcionalidades concretas que responden a necesidades particulares de los usuarios finales.

Además de las categorías tradicionales de software, se destaca el software embebido o integrado, el cual está diseñado para gestionar y controlar dispositivos electrónicos que no son considerados computadores en el sentido convencional. Este tipo de software se encuentra incorporado en sistemas como automóviles, electrodomésticos inteligentes, maquinaria industrial y dispositivos del Internet de las Cosas (IoT). Su funcionamiento está estrechamente vinculado al hardware específico del dispositivo, y cumple tareas precisas con altos niveles de eficiencia y confiabilidad, operando muchas veces de forma autónoma o en tiempo real.

En el proceso de desarrollo de software, se pueden identificar diversos roles fundamentales que intervienen de manera activa en cada etapa del ciclo de vida del proyecto. A continuación, se destacan los perfiles más comunes dentro de este ámbito:

1. **Ingenieros de software**: Son profesionales especializados en la aplicación de principios y enfoques propios de la ingeniería para el diseño, desarrollo y mantenimiento de soluciones informáticas que atienden a problemas concretos o necesidades específicas dentro de un entorno determinado. Su labor implica una profunda comprensión de los requisitos funcionales y no funcionales del sistema, así como la capacidad de traducir dichas necesidades en arquitecturas lógicas y técnicas eficientes.

2. **Programadores**: Son los profesionales responsables de la implementación técnica de los sistemas informáticos mediante la escritura del código fuente que permite a las aplicaciones ejecutar funciones específicas. Su labor consiste en traducir los requerimientos del proyecto en instrucciones lógicas comprensibles para las máquinas, utilizando diversos lenguajes de programación como C++, Java, Python, entre otros.

3. **Desarrolladores de software**: Desempeñan un papel fundamental y transversal en el ciclo de vida del desarrollo de sistemas informáticos, asumiendo responsabilidades que van más allá de la simple codificación. Su labor comprende la implementación de funcionalidades mediante la escritura de código, pero también incluye la coordinación de procesos técnicos, la ejecución de pruebas de validación, la resolución de incidencias y el mantenimiento continuo del software.

Es importante resaltar que el desarrollo de software no constituye una actividad exclusiva de programadores o empresas dedicadas al ámbito tecnológico. En la actualidad, profesionales provenientes de diversas disciplinas, tales como las ciencias naturales, la ingeniería, la medicina, e incluso la manufactura de dispositivos y hardware, también participan activamente en la creación de soluciones informáticas adaptadas a sus contextos específicos.

2.1.5. Teoría de la transformación digital

Hace referencia a un proceso estratégico y estructural mediante el cual las organizaciones redefinen sus modelos operativos, estructuras internas, metodologías de trabajo y enfoques de negocio con el propósito de adaptarse a un entorno altamente digitalizado y competitivo. Este proceso implica no solo la incorporación de tecnologías emergentes —como la inteligencia artificial, el análisis de datos, la automatización, la computación en la nube o el internet de las cosas (IoT)—, sino también una transformación profunda de la cultura organizacional y de la mentalidad de los líderes y colaboradores.

Tal como lo expone Sonia Duro Limia [8], la transformación digital permite a las empresas generar ventajas competitivas y obtener beneficios sostenibles a través de la integración estratégica de herramientas tecnológicas en sus procesos clave. Esta transformación, por tanto, no debe entenderse como una simple modernización tecnológica, sino como una reconceptualización integral de cómo la organización opera, interactúa con sus clientes,

responde al mercado y genera valor.

En este sentido, la transformación digital se convierte en un elemento crucial para la supervivencia y el crecimiento en la era digital, exigiendo una visión a largo plazo, liderazgo sólido, capacitación continua del talento humano y una constante apertura al cambio e innovación. Cuando se habla de emociones, se debe estar listo para abordar varias teorías, y es que este tema es algo muy abstracto y gracias a estas teorías se puede obtener una estructura conceptual que permita comprender un poco mejor la vida emocional humana, en este caso se verán las comparaciones realizadas a cuatro teorías principales:

2.1.6. Pruebas Unitarias

Las pruebas unitarias constituyen fragmentos automatizados de código diseñados para verificar el comportamiento individual de unidades específicas dentro de un sistema de software, tales como funciones o métodos. Estas pruebas se centran en validar que cada unidad funcional opere de forma correcta cuando se encuentra aislada del resto del sistema, especialmente aquellas que ejecutan procesos críticos o complejos. Su principal objetivo es asegurar que cada componente genere los resultados esperados de forma independiente, facilitando así la detección temprana de errores y permitiendo su corrección oportuna durante las primeras fases del ciclo de desarrollo.

La implementación de pruebas unitarias no solo contribuye a la calidad del software, sino que además mejora la mantenibilidad del código, ya que fomenta el desarrollo de módulos más desacoplados y coherentes. En contraste, la omisión de estas prácticas puede derivar en arquitecturas frágiles y con alto acoplamiento, dificultando significativamente su modificación o ampliación. Esta situación se agrava cuando el desarrollador encargado de realizar un mantenimiento o una extensión del módulo no es el mismo que lo diseñó originalmente, lo que puede generar riesgos adicionales y ralentizar el proceso.

En suma, las pruebas unitarias representan una herramienta esencial en el aseguramiento de la calidad del software, ya que permiten no solo validar funcionalidades aisladas, sino también construir una base sólida para el desarrollo ágil, continuo y confiable de aplicaciones complejas [9].

2.1.7. Pruebas de cobertura

Las **pruebas de cobertura**, también conocidas como *coverage testing*, constituyen una técnica fundamental en la ingeniería de software, orientada a evaluar la efectividad y exhaustividad de las pruebas realizadas sobre el código fuente de una aplicación. Este procedimiento permite cuantificar el grado de ejecución del código durante el proceso de prueba, es decir, identifica qué segmentos han sido efectivamente evaluados y cuáles han quedado sin comprobar, lo cual es crucial para garantizar la fiabilidad del software [10].

La cobertura del código puede analizarse desde distintos niveles o perspectivas:

- **Cobertura de líneas**: Determina el porcentaje de líneas de código ejecutadas du-

rante la ejecución de los tests. A mayor cobertura de líneas, mayor es la seguridad de que el flujo básico del código ha sido revisado.

- **Cobertura de ramas**: Evalúa qué proporción de las posibles rutas de decisión —como condicionales `if`, `else`, `switch`, entre otros— ha sido recorrida durante las pruebas. Este enfoque es vital para asegurar que el sistema se comporta correctamente ante diferentes escenarios lógicos.
- **Cobertura de condiciones**: Verifica si todas las condiciones booleanas dentro de las expresiones lógicas han sido evaluadas tanto con un resultado verdadero como falso. Esta cobertura proporciona un análisis más detallado de las decisiones internas del software.

Es fundamental aclarar que alcanzar un alto porcentaje de cobertura no garantiza la completa ausencia de errores. Sin embargo, sí representa un indicio sólido de la calidad de las pruebas ejecutadas, y por ende, del grado de confianza que se puede tener sobre la solidez y robustez del sistema. En este sentido, las pruebas de cobertura son un recurso estratégico para los equipos de desarrollo, no solo para detectar omisiones en el proceso de verificación, sino también para optimizar la estructura del código y priorizar tareas de refactorización o pruebas adicionales.

2.1.8. Pruebas Estáticas

Las pruebas estáticas constituyen un mecanismo de verificación dentro del ciclo de vida del desarrollo de software, cuyo propósito es identificar errores, inconsistencias y defectos sin necesidad de ejecutar el código del programa. A diferencia de las pruebas dinámicas, este tipo de validación se realiza mediante el análisis y revisión de los artefactos generados durante el desarrollo, tales como los documentos de requisitos, diseños arquitectónicos y el propio código fuente.

Entre las técnicas comúnmente empleadas en las pruebas estáticas se encuentran las revisiones por pares, las inspecciones formales y el análisis estático automatizado mediante herramientas especializadas. Estos enfoques permiten detectar, en etapas tempranas, aspectos como el incumplimiento de estándares de codificación, errores lógicos, redundancias, posibles vulnerabilidades de seguridad, o interpretaciones incorrectas de los requerimientos funcionales.

El objetivo principal de este tipo de pruebas es asegurar la calidad estructural del software desde las primeras fases del proceso, promoviendo buenas prácticas de codificación y reduciendo significativamente los costos asociados a la corrección de errores cuando estos son identificados en etapas más avanzadas del desarrollo. Además, contribuyen a mejorar la mantenibilidad, legibilidad y comprensión del código, factores clave en entornos colaborativos y en proyectos de largo plazo donde es frecuente que diferentes desarrolladores intervengan sobre el mismo módulo a lo largo del tiempo [11].

2.1.9. Pruebas Dinámicas

Las pruebas dinámicas constituyen una técnica fundamental dentro del proceso de aseguramiento de calidad del software, la cual se basa en la ejecución del programa o aplicación con el fin de identificar defectos que solo pueden evidenciarse durante su funcionamiento en tiempo de ejecución. A diferencia de las pruebas estáticas, que se enfocan en el análisis estructural del código sin ejecutarlo, las pruebas dinámicas requieren un entorno controlado donde se simulen escenarios de uso reales y extremos para observar el comportamiento del sistema bajo diversas condiciones.

Este tipo de pruebas permite validar no solo la funcionalidad del software, sino también aspectos como la estabilidad, eficiencia, robustez y experiencia de usuario. Al ejecutarse casos de prueba diseñados tanto para los flujos esperados como para situaciones excepcionales o no previstas, se logra una evaluación más integral del desempeño de la aplicación. Es común que los desarrolladores construyan el software bajo ciertos supuestos o estándares que, si bien pueden ser técnicamente correctos, no siempre se alinean con las necesidades, expectativas o hábitos reales del usuario final.

Por tanto, las pruebas dinámicas resultan esenciales para garantizar que el producto final cumpla con criterios de calidad tanto funcionales como no funcionales, minimizando la presencia de errores en producción y asegurando que la entrega sea confiable, eficiente y adecuada al contexto de uso previsto [12].

2.2. Marco Conceptual

2.2.1. Trello y la metodología Kanban

Trello constituye una herramienta de gestión de proyectos basada en los principios de la metodología Kanban, la cual se fundamenta en un sistema visual que regula de forma sincrónica tanto los procesos de producción como el desarrollo estructurado de proyectos. Este enfoque se materializa mediante el uso de tarjetas, que actúan como unidades informativas representando las diferentes actividades necesarias para alcanzar un objetivo específico, reflejando de manera dinámica el avance en el flujo de trabajo.

Dentro de este entorno colaborativo, Trello proporciona funcionalidades orientadas a facilitar la interacción entre los miembros del equipo. Estas incluyen la asignación detallada de tareas, la incorporación de descripciones y recursos necesarios, así como el seguimiento del rendimiento asociado a las actividades completadas [13].

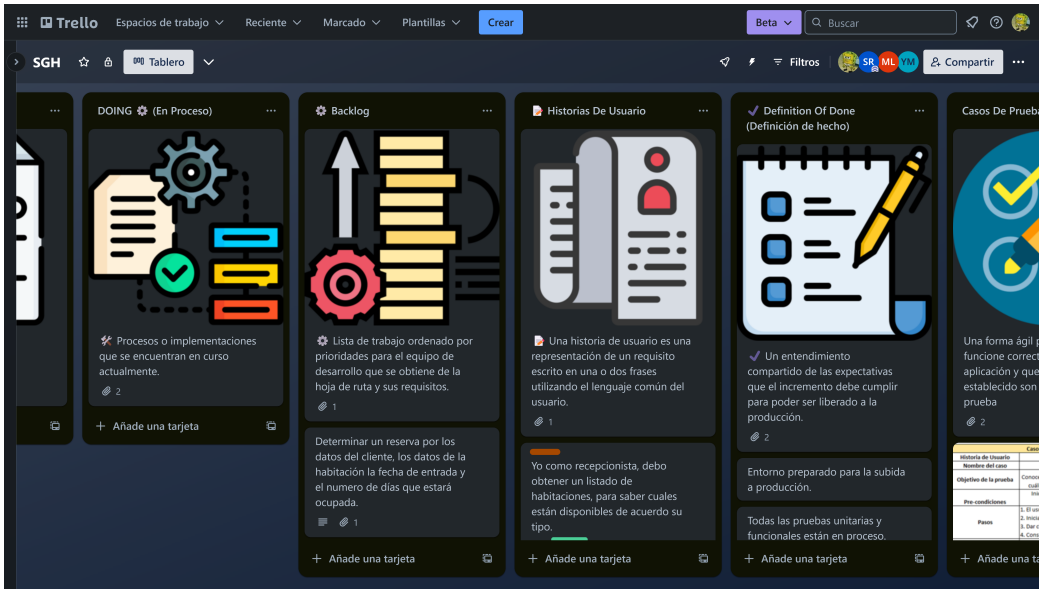


Figura 2.1: Tablero de organización de un proyecto en Trello.
Fuente: Elaboración propia

En Trello, los proyectos son representados mediante tableros, los cuales se dividen en listas que agrupan diversas tarjetas, cada una correspondiente a una tarea o subtarea específica. Estas listas permiten estructurar y visualizar el estado actual del proyecto, como se observa en la Figura 2.1, donde se ejemplifica un tablero compuesto por cinco listas principales relacionadas con diferentes fases del flujo de trabajo.

Cada tarjeta puede ser desplazada entre listas para reflejar cambios en el estado de la tarea, desde su planificación inicial hasta su ejecución o finalización. Esta capacidad de reubicación simboliza la evolución continua del trabajo dentro del proyecto. Además, las tarjetas almacenan múltiples atributos que enriquecen la gestión individual de cada asignación [13], como se presenta en la Figura 2.2.

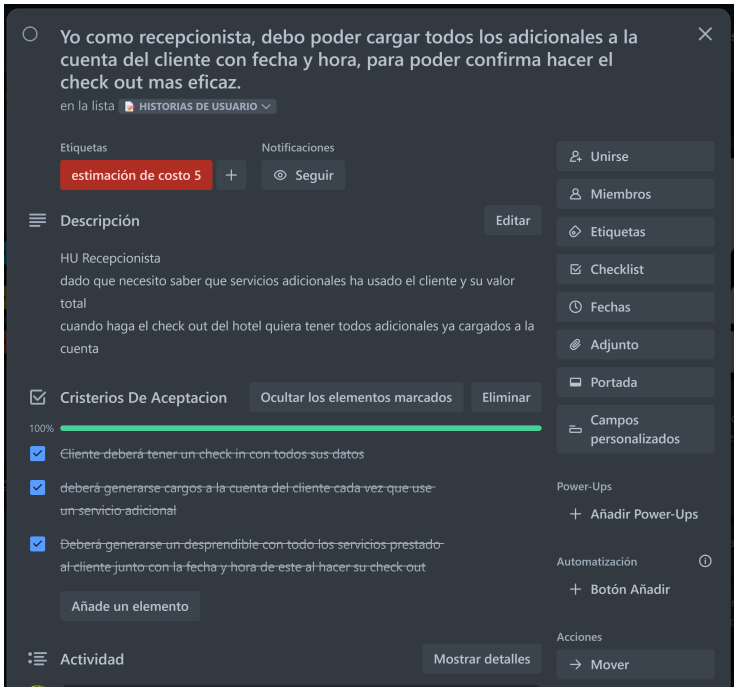


Figura 2.2: Panel de una tarea en Trello.
Fuente: Elaboración propia

Cada tarjeta en Trello incorpora una serie de propiedades que permiten una administración detallada de cada tarea:

- **Título de la tarea:** Funciona como identificador único de la asignación.
- **Estado de la tarea:** Refleja la etapa actual en la que se encuentra la tarea, categorizándola como pendiente", en progreso" o "completada".
- **Descripción:** Proporciona información detallada sobre la tarea, incluyendo objetivos y criterios de éxito.
- **Miembros asignados:** Indica los colaboradores responsables de ejecutar la tarea.
- **Fecha de vencimiento:** Define el plazo límite para la finalización de la actividad, con posibilidad de sincronización con herramientas como Google Calendar.
- **Lista de verificación:** Descompone la tarea en pasos o actividades más pequeñas, permitiendo monitorear el progreso individual.
- **Etiquetas:** Permite clasificar visualmente las tareas mediante hasta seis etiquetas codificadas por color, personalizables según las necesidades del equipo.
- **Archivos adjuntos:** Ofrece la posibilidad de incorporar documentos relevantes desde plataformas de almacenamiento en la nube como Dropbox o Google Drive.
- **Comentarios:** Espacio de comunicación donde los miembros asignados pueden compartir actualizaciones, sugerencias o resolver inquietudes relacionadas con la tarea.

2.2.2. Scrum

Scrum es un marco de trabajo ágil que se fundamenta en un enfoque iterativo e incremental, diseñado para facilitar la gestión eficiente de proyectos, productos y aplicaciones en entornos complejos. Esta metodología organiza el desarrollo en ciclos llamados *Sprints*, los cuales tienen una duración fija, comúnmente entre una y cuatro semanas, y se suceden de manera continua y planificada.

Cada Sprint inicia con una reunión de planificación en la que el equipo identifica los elementos del producto que representan mayor prioridad, seleccionándolos desde una lista denominada *Product Backlog*. Estos elementos se trasladan al *Sprint Backlog*, y el equipo se compromete a completarlos dentro del periodo establecido. Es importante destacar que, una vez iniciado el Sprint, no se permite modificar los elementos seleccionados, promoviendo así el enfoque y la estabilidad del trabajo en curso.

Durante la ejecución del Sprint, el equipo realiza encuentros diarios conocidos como *Daily Scrum*, cuyo objetivo es comunicar los avances, identificar obstáculos y actualizar el estado de las tareas a través de herramientas visuales como gráficos de quemado (*burndown charts*). Al finalizar el Sprint, se lleva a cabo una revisión del incremento desarrollado junto a los interesados, permitiendo la recolección de retroalimentación para futuros ciclos. Posteriormente, se realiza una *retrospectiva* con el equipo, con el fin de reflexionar sobre el proceso y establecer mejoras continuas.

Una de las características esenciales de Scrum es su orientación hacia la entrega continua de productos funcionales y potencialmente entregables al final de cada Sprint. Esto requiere que cada componente desarrollado esté debidamente probado, validado y en condiciones de ser implementado, lo que favorece una mayor adaptabilidad a los cambios y necesidades del cliente.

El principio de *inspección y adaptación* constituye el núcleo de esta metodología. Dado que el desarrollo de productos implica incertidumbre, aprendizaje y creatividad, Scrum propone avanzar mediante pequeños pasos, evaluando de forma constante tanto el resultado del trabajo como la eficacia del proceso aplicado. Esto permite realizar ajustes oportunos que optimicen la calidad del producto y la eficiencia del equipo.

Los eventos fundamentales que componen el marco de Scrum se ilustran en la Figura 2.3, la cual proporciona una visión general de su estructura y funcionamiento [14].

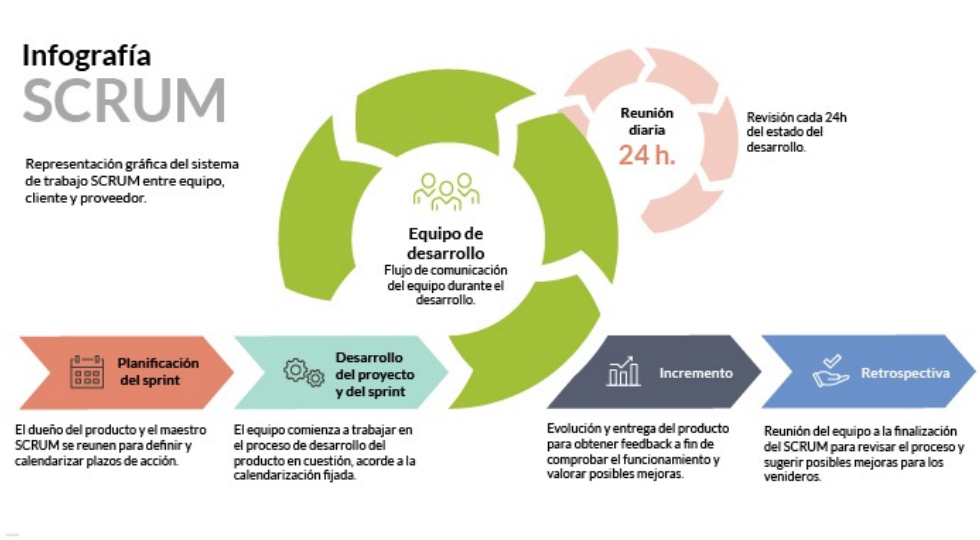


Figura 2.3: Esquema general de Scrum.
Fuente: [15]

2.2.3. MVC (Modelo-Vista-Controlador)

El patrón arquitectónico Modelo-Vista-Controlador (MVC) se ha consolidado como una estrategia eficaz para la estructuración de sistemas que requieren la interacción entre múltiples componentes y el uso compartido de datos. Su propósito fundamental es facilitar el desarrollo de aplicaciones mediante la separación de responsabilidades, promoviendo así la modularidad, el mantenimiento y la escalabilidad del software. En esta arquitectura, cada componente cumple un rol específico:

- **Modelo:** Representa la lógica de negocio y la gestión de los datos del sistema.
- **Vista:** Se encarga de la representación visual de la información contenida en el Modelo.
- **Controlador:** Actúa como intermediario entre la Vista y el Modelo, gestionando la entrada del usuario y actualizando los otros componentes en consecuencia.

Una de las características más destacadas de esta arquitectura es que cualquier cambio efectuado en el Modelo se refleja automáticamente en las distintas Vistas que lo referencian, garantizando una sincronización en tiempo real. Esta propiedad es especialmente valiosa en aplicaciones de visualización de datos gráficos, en las que diferentes secciones del contenido pueden representarse en ventanas independientes con distintos niveles de zoom o detalle [16].

Entre los beneficios que ofrece el enfoque MVC, se pueden destacar los siguientes:

- **Separación de responsabilidades:** Los componentes del sistema se implementan de manera aislada, lo que facilita su desarrollo, mantenimiento y prueba.

- **Definición clara de interfaces:** El esquema proporciona una Interfaz de Programación de Aplicaciones (API) bien estructurada, permitiendo la sustitución de cualquiera de los componentes (Modelo, Vista o Controlador) sin afectar a los demás.
- **Vinculación dinámica:** La relación entre el Modelo y sus Vistas se establece durante la ejecución, lo que dota al sistema de una alta capacidad de adaptación a diferentes contextos o requisitos [16].

La implementación del patrón MVC permite construir los módulos del programa de manera independiente y posteriormente integrarlos en tiempo de ejecución. Esta modularidad implica que, en caso de fallo o necesidad de actualización de un componente, este puede ser reemplazado sin repercutir en el funcionamiento de los demás. Esta capacidad contrasta con los enfoques monolíticos tradicionales, comunes en aplicaciones de baja o media complejidad, en los cuales los componentes se integran en un solo bloque o contenedor (Frame), dificultando su modificación sin comprometer el sistema en su totalidad, tal como se referencia en la figura 2.4.

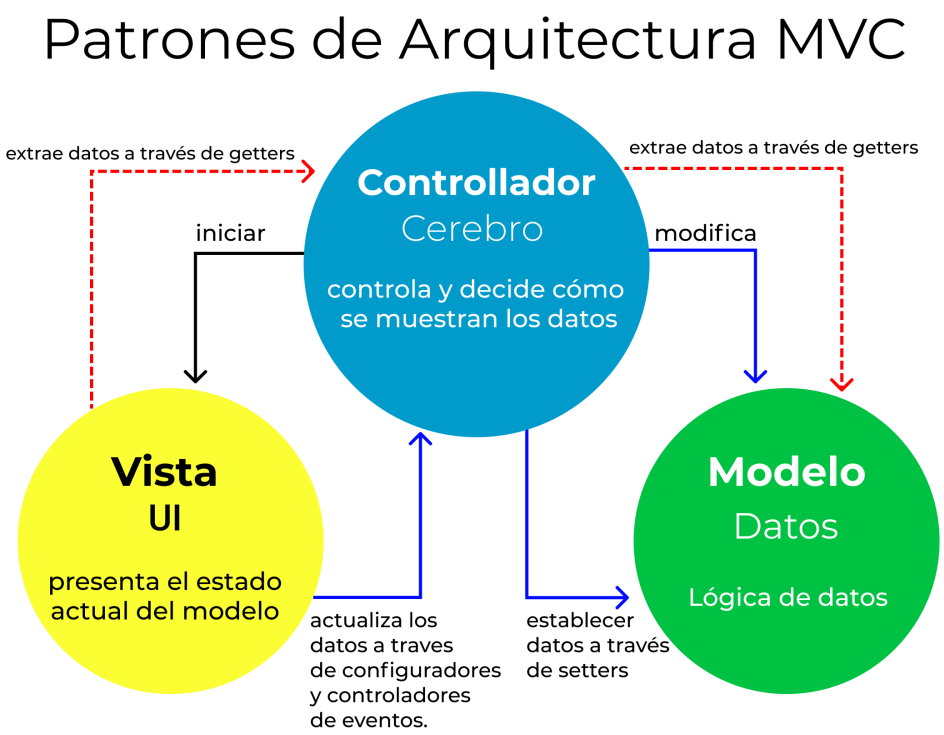


Figura 2.4: Gráfica de patrones de arquitectura MVC.
Fuente: [17]

2.2.4. API (Interfaz de Programación de Aplicaciones)

Una **API** (*Application Programming Interface*, por sus siglas en inglés) constituye un conjunto de definiciones y protocolos que facilitan la comunicación entre distintos compo-

nentes de software. Se trata de una interfaz o punto de contacto provisto por bibliotecas o paquetes, representada en la figura 2.5, diseñada para que otros programas puedan interactuar con ellos, acceder a sus funcionalidades y ejecutarlas de manera eficiente [18].

En términos generales, una API actúa como un puente que permite la integración entre sistemas, posibilitando la reutilización de código y promoviendo la interoperabilidad entre diferentes plataformas o tecnologías.

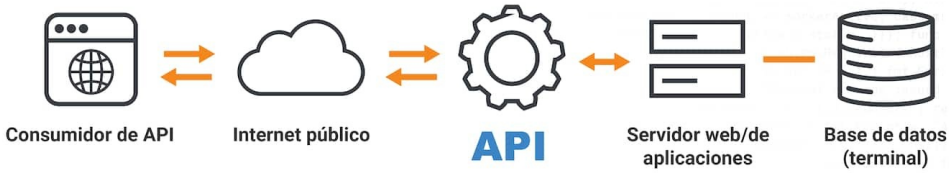


Figura 2.5: Representación gráfica del funcionamiento general de una API.
Fuente: [19]

2.2.5. REST (Representational State Transfer)

REST, acrónimo de *Representational State Transfer*, es un estilo de arquitectura de software que define un conjunto de restricciones y principios para el diseño de servicios web. Su propósito principal es permitir la interoperabilidad entre sistemas en red utilizando los protocolos y estándares existentes de la web, siendo HTTP el más comúnmente utilizado [20].

Este modelo arquitectónico fue propuesto por Roy Fielding en el año 2000 en su tesis doctoral, y desde entonces se ha convertido en una de las metodologías más adoptadas para la creación de servicios web ligeros, también conocidos como *Web APIs* o *RESTful APIs*.

REST emplea métodos estándar de HTTP, como `GET`, `POST`, `PUT`, `DELETE`, entre otros, para realizar operaciones sobre recursos representados habitualmente en formatos como JSON o XML, tal como en la figura 2.6. Cada recurso puede ser identificado mediante una URI (Identificador Uniforme de Recursos), y la comunicación entre el cliente y el servidor se mantiene sin estado, es decir, cada solicitud HTTP contiene toda la información necesaria para ser procesada, sin requerir conocimiento del contexto previo.

Entre las principales características de REST se destacan:

- **Ligereza:** En comparación con otros protocolos como SOAP (*Simple Object Access Protocol*), REST es más simple y eficiente, lo que lo hace ideal para aplicaciones web modernas, móviles y servicios en la nube.
- **Escalabilidad:** Gracias a su arquitectura sin estado, REST permite una mayor escalabilidad del sistema, facilitando la distribución de las solicitudes entre distintos servidores.
- **Flexibilidad en los formatos:** Aunque JSON es el formato más comúnmente utilizado, REST permite el uso de múltiples tipos de representaciones para los recursos, como XML, YAML o incluso HTML.
- **Separación cliente-servidor:** REST fomenta una clara separación entre las responsabilidades del cliente (interfaz de usuario) y el servidor (gestión de datos), lo cual permite desarrollar cada parte de manera independiente.

REST se ha consolidado como una alternativa ampliamente adoptada frente a otras arquitecturas más complejas. Su simplicidad, junto con el uso de estándares ya existentes en la web, ha promovido su implementación en una amplia gama de aplicaciones, desde servicios web hasta sistemas distribuidos de gran escala.

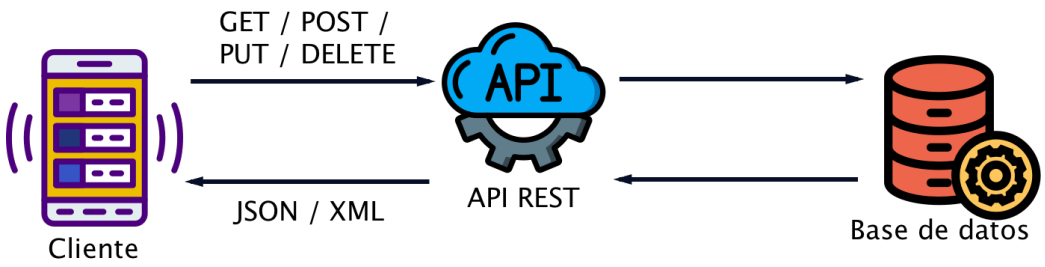


Figura 2.6: Interacción entre cliente, API REST y base de datos.
Fuente: [21]

Capítulo 3

Alcances del Proyecto

3.1. Declaración del Alcance

Durante esta pasantía, se trabajó en el fortalecimiento de la funcionalidad y estabilidad de la aplicación AcademiApp, desarrollada por la empresa Webcloster S.A.S, a través del desarrollo e implementación de pruebas unitarias en el front-end de diversos módulos. Estos módulos incluyen funcionalidades clave como almuerzos para estudiantes, trabajos de grado, carga de registros de huella, votaciones y registro de dificultades académicas.

La aplicación AcademiApp es una herramienta web diseñada para apoyar tanto los procesos académicos como los administrativos de las instituciones educativas. La ejecución de pruebas se llevará a cabo utilizando el framework Angular, haciendo uso de Karma como herramienta principal para la generación y creación de pruebas unitarias e integración.

Además de las pruebas, se realizó desarrollos funcionales como la reserva de almuerzos por bloques de días a la semana, gestión de actividades y evaluación de trabajos de grado, subida y revisión de documentos para la habilitación del servicio de almuerzos, búsqueda de similitudes en títulos de trabajos de grado, optimización de múltiples filtros y mejora en los servicios de generación de informes. Asimismo, se llevará a cabo el mantenimiento de los módulos existentes, según las prioridades establecidas por la empresa y las necesidades de los clientes.

Todas las tareas asignadas se desarrollaron siguiendo los lineamientos técnicos y estándares de calidad definidos por Webcloster S.A.S, ajustándose a las condiciones del sistema, a los requerimientos urgentes de los clientes, y a la disponibilidad de los equipos de desarrollo involucrados.

3.1.1. Restricciones

Una de las principales limitaciones del proyecto fue el **tiempo**, ya que se contó con un período de **ocho meses** para alcanzar los objetivos propuestos. Este plazo exigió una adecuada planificación y adaptación por parte del responsable del trabajo, quien se integró

a un proyecto en curso y debió ajustarse rápidamente al entorno tecnológico ya establecido por la empresa.

Durante el desarrollo de las actividades, se hizo uso de diversas tecnologías y herramientas que facilitaron el trabajo colaborativo y la gestión eficiente del proyecto:

- **Lenguajes de programación:**
  - *TypeScript* (superset de *JavaScript*) con el framework *Angular* para el desarrollo del front-end.
- **Gestor de actividades:** Se utilizó una herramienta interna de la empresa llamada *Webdesk*, empleada como radiador de información. Esta plataforma permite registrar y organizar tareas en un tablero Kanban, con una interfaz y dinámica similar a Trello, como se muestra en la figura 2.1. Cabe mencionar que, antes de la implementación de Webdesk, también se utilizaron los *issues* de GitLab para la asignación y seguimiento de tareas.
- **Metodología de desarrollo de software:** Se empleó la metodología ágil *SCRUM*, permitiendo un desarrollo iterativo e incremental mediante reuniones periódicas y revisiones constantes del producto.
- **Controlador de versiones:** Se utilizó *GitLab* para el control de versiones, la gestión de ramas, la integración continua y el seguimiento de incidencias.
- **Arquitectura de software:** La aplicación cuenta con una arquitectura *desacoplada*, que permite el desarrollo y mantenimiento de módulos de forma independiente, facilitando su escalabilidad y reutilización.

3.1.2. Supuestos

- Se contó con el apoyo constante del director de trabajo de grado durante el desarrollo del proyecto, abarcando las asignaturas de *Trabajo de Grado I* y *Trabajo de Grado II*.
- Se asumió el compromiso por parte del responsable del proyecto para el cumplimiento de los objetivos planteados dentro del periodo estipulado de ocho meses.
- Se dispuso de los recursos tecnológicos necesarios, tanto a nivel de hardware como de software, para el adecuado desarrollo del proyecto.

3.1.3. Entregables

A continuación, en la tabla 3.2, se especifican los resultados esperados para cada objetivo específico planteado en el proyecto:

Objetivo General	**Objetivo Específico**	**Metodología**	**Entregables**
Implementar pruebas unitarias enfocadas en el Front-End de la aplicación AcademiApp para la empresa Webcloster S.A.S.	Analizar los módulos de la aplicación general con ausencia de pruebas.	1. Revisión estática del código fuente. 2. Revisión manual de funcionalidades.	1. Lista de hallazgos y posibles riesgos por ausencia de pruebas. 2. Recomendaciones para implementación de pruebas.
	Diseñar pruebas unitarias para los módulos sin documentación ni sistematización suficiente.	1. Diseño de pruebas basado en casos límite y flujo lógico del código. 2. Revisión colaborativa con el equipo de desarrollo (cuando sea posible) para validar supuestos. 3. Exploración funcional y mapeo manual de entradas/salidas esperadas. 4. Priorización de funciones críticas o con mayor complejidad para ser probadas.	1. Código fuente de las pruebas unitarias con comentarios aclaratorios. 2. Conjunto de pruebas unitarias implementadas.
	Desarrollar los casos de prueba priorizados por la empresa. (incluyendo nuevos módulos que sean requeridos con urgencia)	1. SCRUM 2. Diseño de casos de prueba basados en criterios funcionales y de negocio.	1. Módulos construidos con base en los requerimientos funcionales. 2. Código fuente y reportes de cobertura de pruebas unitarias desarrolladas. 3. Interfaces de usuario diseñadas bajo principios de UX. 4. Código fuente de los módulos desarrollados.
	Generar informes de pruebas estáticas y de cobertura.	1. Pruebas funcionales. 2. Pruebas de aceptación. 3. Pruebas de estabilidad.	1. Documento con los casos de prueba documentados. 2. Pruebas funcionales ejecutadas. 3. Reportes de cobertura en los módulos probados.

Tabla 3.2: Entregables del proyecto

Capítulo 4

Metodologías

4.1. Metodología de Investigación

La presente investigación se enmarca dentro del enfoque aplicado, ya que busca generar conocimientos útiles mediante la intervención directa en problemáticas reales del campo de estudio. En este sentido, se establece una sólida relación entre la teoría y la práctica, fundamentada en los hallazgos obtenidos a través de la investigación tecnológica. Esta aproximación metodológica se respalda en los niveles de maduración tecnológica definidos por el Ministerio de Ciencia, Tecnología e Innovación de Colombia [22] mostrados en la figura 4.1, los cuales permiten clasificar los proyectos según su grado de desarrollo. Dado que el sistema objeto de estudio ya se encuentra desplegado y en funcionamiento, se determina que corresponde al nivel TRL 9.

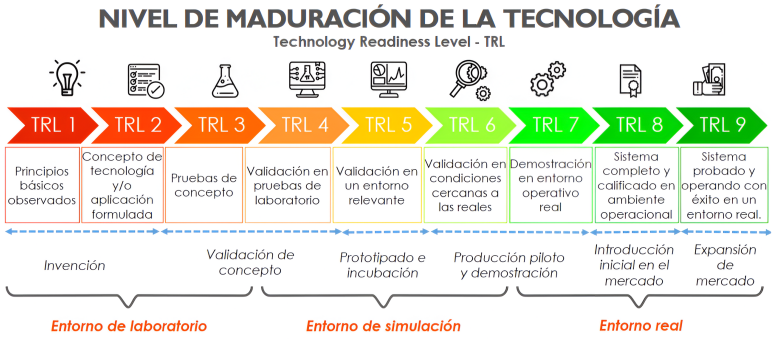


Figura 4.1: Niveles de Maduración Tecnológica.
Fuente: [23]

4.2. Selección de la Metodología de Desarrollo

Para el desarrollo de los módulos del proyecto, se llevó a cabo una evaluación comparativa entre tres metodologías ágiles: *XP*, *Scrum* y *RAD*. La valoración se realizó a través de un análisis multicriterio, en el cual se definieron una serie de aspectos clave para el con-

texto del proyecto. Cada criterio fue puntuado en una escala de 1 a 5, donde 1 representa una condición poco favorable y 5 una condición altamente favorable.

Los criterios considerados incluyeron: presupuesto, tamaño del proyecto, tiempos límite de entrega, nivel de documentación, disponibilidad del personal, adaptabilidad a cambios, claridad en los roles y responsabilidades, y presencia previa de la metodología en la empresa. La Tabla 4.2 presenta el resultado de esta evaluación.

Criterios	**XP**	**Scrum**	**RAD**
Presupuesto	4	5	3
Tamaño del proyecto	3	4	4
Tiempos límite de entrega	4	5	4
Documentación	5	5	4
Personal del proyecto	3	5	3
Adaptabilidad a cambios	4	5	3
Roles y responsabilidades	3	5	3
Presencia en la empresa	0	5	0
Total	**26**	**39**	**23**

Tabla 4.2: Análisis cualitativo de metodologías de desarrollo.

A partir del análisis, la metodología *Scrum* obtuvo el mayor puntaje, destacándose especialmente por su adaptabilidad, claridad en los roles y responsabilidades, y por estar ya implementada en la empresa Webcloster S.A.S. Esta última característica representa una ventaja significativa, pues facilita la integración del proceso de pasantía al flujo de trabajo existente.

Aunque *Scrum* no se considera una metodología de desarrollo en sentido estricto, su aplicación como marco de trabajo ágil ha demostrado eficiencia y efectividad en el desarrollo de software. Por estas razones, se eligió *Scrum* como la metodología principal para la ejecución del proyecto, asegurando una planificación iterativa, seguimiento constante mediante reuniones semanales (*weekly stand-ups*), revisiones de sprint (*sprint reviews*) y retrospectivas (*retrospectives*), así como el cumplimiento oportuno de los entregables acordados.

4.3. Metodología de Gestión de Actividades

En respuesta a la creciente demanda y a la constante evolución del entorno tecnológico, se ha evidenciado la necesidad de adoptar estrategias que permitan optimizar tanto los costos como los tiempos de desarrollo, manteniendo la capacidad de adaptación frente a los cambios continuos en los requerimientos del cliente. En este sentido, se ha decidido implementar el enfoque Kanban dentro del ciclo de ingeniería de software, enmarcado dentro de las denominadas metodologías ágiles.

Kanban, originalmente aplicado con éxito en la industria manufacturera, ha demostrado ser una alternativa eficaz en el ámbito del desarrollo de software. Sus principales fortalezas

radican en la capacidad de adaptación, la gestión eficiente de cambios en los requisitos y la visualización clara y constante del estado del flujo de trabajo, lo cual permite una toma de decisiones más ágil y fundamentada [24].

Cabe resaltar que la empresa Webcloster S.A.S. hace uso de la plataforma de gestión de actividades Trello, una herramienta basada en los principios del modelo Kanban. En consecuencia, durante el desarrollo de esta pasantía, se empleará dicha metodología de gestión, aprovechando no solo su eficacia comprobada, sino también la familiaridad del equipo de trabajo con esta herramienta, lo cual facilitará su integración en los procesos ya establecidos.

Capítulo 5

Desarrollo del Proyecto

5.1. Sobre AcademiApp

La aplicación web *AcademiApp*, desarrollada por la empresa Webcloster S.A.S. constituye una solución tecnológica orientada a la gestión de procesos académicos y administrativos en instituciones educativas. Su propósito es optimizar la administración de la información relacionada con estudiantes, docentes y personal directivo, mediante una interfaz organizada y funcional.

La estructura de navegación de AcademiApp se basa en una jerarquía compuesta por tres niveles: módulos, submódulos y opciones, distribuidos estratégicamente para facilitar el acceso a las funcionalidades del sistema.

- **Módulos:** Constituyen los contenedores principales del sistema. Cada módulo está diseñado con base en un propósito específico, una temática o un tipo de usuario. Dentro de ellos se agrupan los submódulos correspondientes.
- **Submódulos:** Funcionan como categorías intermedias que agrupan las diferentes funcionalidades del sistema, permitiendo una mejor organización y accesibilidad.
- **Opciones:** Representan las funcionalidades específicas disponibles para el usuario. Estas se despliegan desde los submódulos y, al ser el último nivel jerárquico, están enlazadas mediante Angular a componentes que direccionan a las respectivas interfaces de usuario.

Actualmente, AcademiApp cuenta con un total de 20 módulos. A continuación, se describen algunos de los más representativos:

- **Módulo de estudiantes:** Este módulo proporciona herramientas para que los estudiantes gestionen su información académica. Incluye funcionalidades como el uso de unidades gamificadas en las asignaturas, acceso a simuladores de pruebas y registro de trabajos de grado.
- **Módulo de profesores:** Permite a los docentes organizar su horario de clases, registrar asistencia, gestionar su disponibilidad y definir unidades gamificadas para las materias a su cargo.

- **Módulo de gerencia académica:** Contiene submódulos destinados al área administrativa, ofreciendo funcionalidades como la gestión de trabajos de grado, el registro y consulta de monitorías, y la administración de la disponibilidad docente.
- **Módulo de usuarios y perfiles:** Este módulo permite registrar y consultar usuarios y perfiles, así como definir los permisos y funcionalidades asociadas a cada rol dentro del sistema.

5.2. Gestión del proyecto

Durante la ejecución del proyecto, se adoptaron diversas herramientas y estrategias que permitieron una gestión eficiente de las actividades, el seguimiento del avance y la coordinación entre los miembros del equipo.

- **GitLab Issues:** Inicialmente, la gestión de tareas del proyecto se realizó mediante el sistema de **Issues** de **GitLab**, herramienta que facilitó el registro, clasificación y seguimiento de requerimientos o tareas específicas. Cada issue incluía descripciones detalladas, etiquetas de categorización, asignaciones y fechas objetivo. Tal como se muestra en la figura 5.1, esta funcionalidad permitió centralizar el control de los avances en el ciclo de desarrollo.

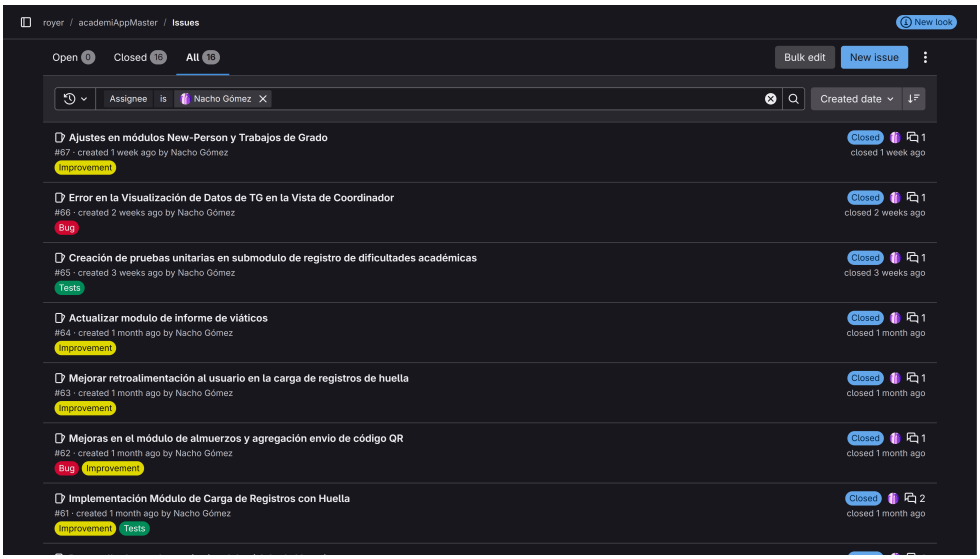


Figura 5.1: Seguimiento de tareas con Issues en GitLab.
Fuente: Elaboración propia

- **Webdesk:** Posteriormente, se incorporó **Webdesk**, una herramienta interna de la empresa con funcionalidades similares a Trello, la cual es mostrada en la figura 5.2, orientada al seguimiento de tareas y procesos de despliegue. Webdesk permite llevar un control visual del avance de cada actividad, identificando claramente su estado (por hacer, en progreso, en revisión, finalizado, etc.), así como los responsables

y observaciones asociadas. Esta plataforma ha optimizado la gestión interna del proyecto, integrando tanto los aspectos de desarrollo como los de despliegue.

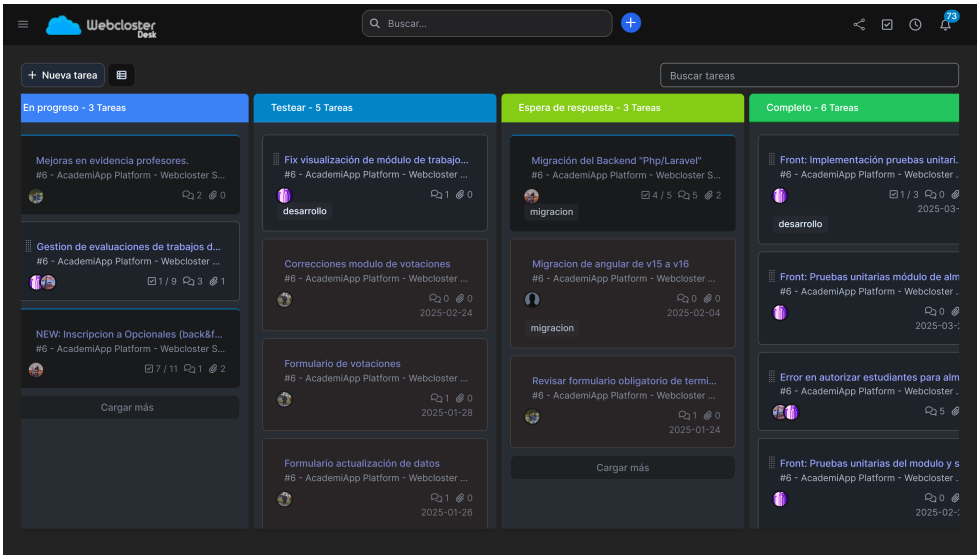


Figura 5.2: Tablero de tareas en Webdesk.
Fuente: Elaboración propia

- **Reunión semanal:** Se definió una reunión semanal como espacio fundamental para el seguimiento del proyecto. En estos encuentros, se discutieron los avances, se identificaron y resolvieron impedimentos, y se planificaron nuevas tareas una vez completadas las anteriores. Esta práctica garantizó una comunicación constante y una colaboración efectiva entre los integrantes del equipo.

- **GitLab (control de versiones):** El control de versiones fue gestionado a través de **GitLab**, herramienta esencial para el desarrollo colaborativo del código fuente. GitLab permitió mantener un repositorio organizado, facilitando la integración de nuevas funcionalidades y el control de cambios realizados durante el ciclo de desarrollo.

- **Pull Request:** Entre las funcionalidades de GitLab, se destaca el uso de *pull requests*, mecanismo que permitió solicitar revisiones de código antes de integrarlo en la versión de producción. Esta práctica mostrada en la figura 5.3 aseguró que cada implementación pasara por un entorno de pruebas, donde se validó el cumplimiento de los requerimientos antes de ser fusionada con la rama principal.

Figura 5.3: Proceso de Pull Request en GitLab.
Fuente: Elaboración propia

- **Notion:** Como complemento a Webdesk y Telegram, se utilizó **Notion** para el control de observaciones, anotaciones y correcciones derivadas de las reuniones. Esta herramienta facilitó la categorización de la información y permitió un registro claro del progreso de las tareas asignadas, tal como se muestra en la figura 5.4.

Figura 5.4: Uso de Notion como apoyo en la gestión de actividades.
Fuente: Elaboración Propia

- **Telegram:** Para la comunicación interna del equipo, se implementó **Telegram**, aprovechando la creación de un grupo donde se organizaron diferentes chats temáticos según los propósitos del proyecto. Esta estructura permitió mantener una comunicación clara y segmentada, facilitando el seguimiento de avances diarios, la actualización de repositorios, la gestión documental y la coordinación general, promoviendo así un entorno colaborativo eficiente, representado en la figura 5.5.

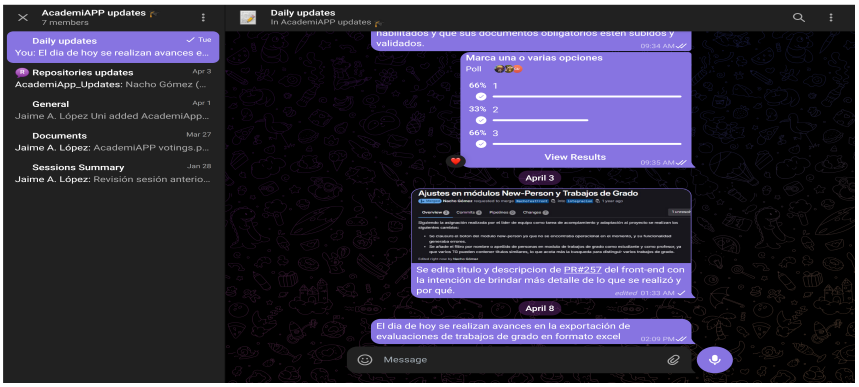


Figura 5.5: Canal de grupo con chats temáticos organizados en Telegram.
Fuente: Elaboración propia

5.3. Arquitectura del Sistema

La arquitectura de software constituye un elemento fundamental que incide directamente en el desempeño del sistema y en su capacidad para satisfacer criterios de calidad esenciales. Esta comprende tanto aspectos técnicos —como la eficiencia en los tiempos de respuesta ante solicitudes específicas de información— como consideraciones de carácter visual y funcional, tales como la usabilidad percibida por los usuarios finales.

En el caso de *Academipp*, se adoptó una arquitectura basada en el modelo cliente-servidor. Este enfoque facilita la gestión centralizada de los servicios, al concentrar la lógica y los recursos críticos en el servidor, los cuales son accedidos por los clientes de forma remota. Esta configuración no solo mejora la escalabilidad del sistema, permitiendo ampliar o modificar funcionalidades sin comprometer el funcionamiento general, sino que también simplifica el mantenimiento. Al posibilitar actualizaciones o adiciones de manera modular, se reduce el impacto sobre la experiencia del usuario durante los cambios del lado del servidor.

La Figura 5.6 ilustra la interacción entre los componentes del sistema bajo este modelo, destacando los roles del servidor como proveedor de servicios y del cliente como consumidor.

Figura 5.6: Modelo cliente-servidor.
Fuente: [25]

5.4. Módulo de Almuerzos

Este módulo corresponde a un componente funcional dentro de *AcademiApp*, diseñado para gestionar de manera integral el proceso de asignación y uso del subsidio de alimentación ofrecido por la universidad. Está compuesto por tres partes principales:

- **Interfaz para estudiantes:** Desde donde pueden visualizar los documentos requeridos y cargar los correspondientes archivos para acceder al beneficio.
- **Panel administrativo:** Permite al personal encargado habilitar estudiantes para ser beneficiarios del subsidio en un periodo y en unos días en específico.
- **Sección del distribuidor:** Enfocada en la gestión del control de entrega de los almuerzos, con posibilidad de generar reportes en formato Excel para facilitar el seguimiento.

5.4.1. Pruebas Unitarias del Módulo de Almuerzos

Anteriormente, este módulo en todo su flujo solo contaba con pruebas unitarias a nivel de servicio. Como parte del proceso de mejora, se han implementado pruebas unitarias en todas sus partes, abarcando tanto los componentes visuales como los flujos de interacción, con el fin de garantizar la calidad, funcionalidad y mantenibilidad del sistema, obteniendo una cobertura total aceptable, la cual se establece con un mínimo del 80 % del total de *statements* o declaraciones del componente cubiertas, teniendo en cuenta que una misma linea de codigo puede tener una o más declaraciones, tal como lo muestra el reporte de cobertura de la figura 5.7, esta condición de aceptación aplicó para todos los reportes de prueba generados en el desarrollo de este proyecto.

Cabe mencionar que todos los cambios realizados, tanto en los módulos ya existentes como en los nuevos módulos creados que se describen a continuación, fueron incluidos dentro del proceso de pruebas unitarias, y su respectiva cobertura está reflejada en el reporte mostrado en la figura 5.7. Esto asegura que cada funcionalidad implementada fue verificada mediante pruebas unitarias.

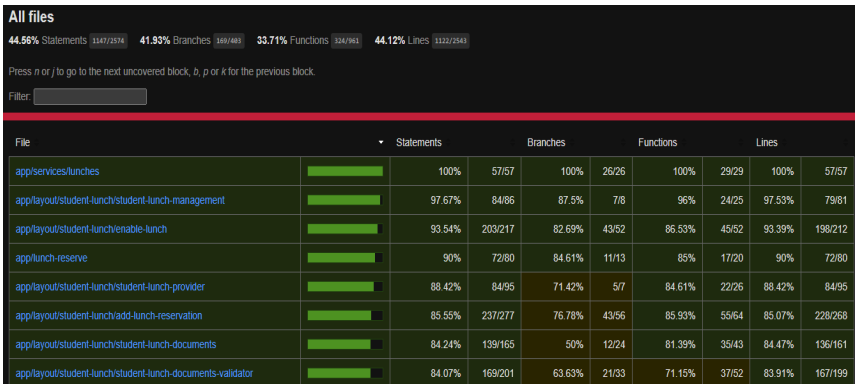


Figura 5.7: Reporte de cobertura de pruebas unitarias en módulo de almuerzos.
Fuente: Elaboración propia

5.4.1.1. Servicio de comunicación con el Back-End

Las pruebas unitarias desarrolladas para este servicio validan principalmente los siguientes aspectos:

- La correcta inicialización del servicio.
- La respuesta esperada de cada llamado a la API que espera el Back-End.
- La validación de los mensajes, códigos de error y cantidad de llamadas recibidas desde la API.
- El comportamiento de la llamada cuando se realiza una consulta o petición vacía.

En la figura 5.8 se muestra un fragmento del código de las pruebas minimizadas del servicio con el fin de abarcar una mayor cantidad en la evidencia.

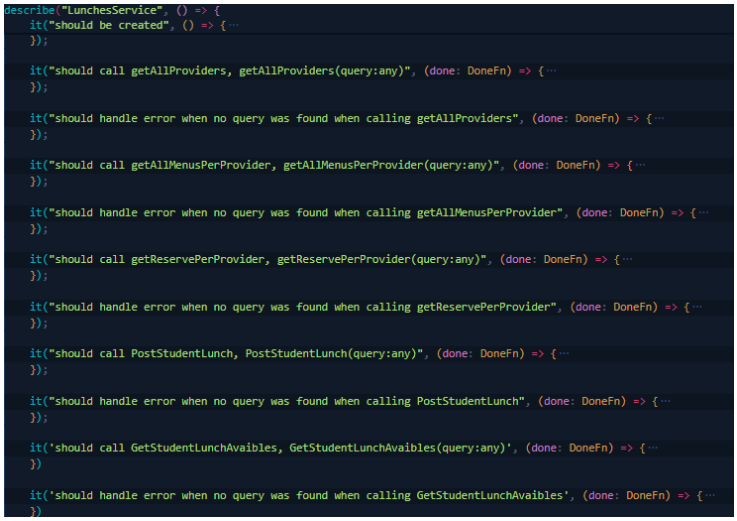


Figura 5.8: Fragmento de código de pruebas del servicio de almuerzos.
Fuente: Elaboración propia

5.4.1.2. Módulo de subida de documentos como estudiante

A continuación, se detallan los principales casos de prueba implementados en este componente, los cuales a su vez, se subdividieron en muchos casos en más pruebas para cada caso de uso por el usuario:

- La correcta inicialización del componente y de los formularios principales, asegurando su estructura y valores por defecto.
- La actualización dinámica del sistema al seleccionar una sede, obteniendo los periodos académicos disponibles y manejando respuestas vacías del servicio.
- La carga, visualización y eliminación de documentos, con retroalimentación visual adecuada al usuario tras cada acción.

- El correcto funcionamiento de la descarga de archivos almacenados en el sistema.
- La adaptación y actualización dinámica de la grilla de documentos, incluyendo el ajuste automático de sus columnas.
- El manejo apropiado de errores durante operaciones de carga, descarga y eliminación, manteniendo la estabilidad del sistema y proporcionando mensajes al usuario como retroalimentación.

Estas pruebas permitieron garantizar que el módulo responda adecuadamente ante entradas válidas e inválidas, errores del servidor, y múltiples interacciones del usuario. Además, se integraron mocks para servicios como `LunchesService`, el cual se comunica con la API para generar las actualizaciones, inserciones y entre otros, el `ModalService` el cual dispara los accionadores para mostrar las ventanas modal y el `TranslateService` el cual se usa para traducir mensajes en pantalla al idioma seleccionado por el usuario desde la interfaz, estos mock facilitaron el aislamiento de cada funcionalidad, esta misma estructura se realizó en cada componente en el que se procedió con la implementación de las pruebas.

En la figura 5.9 se muestra un fragmento del código de las pruebas del componente.

```
describe("StudentLunchDocumentsComponent", () => {
  it("should validate mod form as invalid when file is missing", () => {
    const modForm = component.mod;
    modForm.get("file").setValue(null);
    expect(modForm.invalid).toBeTruthy();
  });

  it("should call getPeriodPerHeadquarter when headquarter is selected", () => {
    const headquarter = { id_headquarters: 1 };
    spyOn(headquartersService, "getPeriodPerHeadquarter").and.returnValue(
      of({ university_periods: [] })
    );
    component.onSelectHeadQuarter(headquarter);
    expect(headquartersService.getPeriodPerHeadquarter).toHaveBeenCalledWith(1);
    expect(component.id_headquarter).toEqual(1);
  });

  it("should handle empty response when no mandatory documents are returned", () => {
    lunchesServiceMock.GetAllMandatoryDocuments.and.returnValue(
      of({ documentsHeadquarters: [] })
    );
    component.onSelectPeriod({ id_university_period: 1 });
    expect(component.dataGridEvidences.length).toBe(0); // No deberia haber documentos
  });

  it("should handle error when GetAllMandatoryDocuments fails", () => {
    lunchesServiceMock.GetAllMandatoryDocuments.and.returnValue(throwError({ message: 'Error' }));

    component.onSelectPeriod({ id_university_period: 1 });

    expect(component['modalService'].showRequestError).toHaveBeenCalledWith('GET', 'Error');
  });
```

Figura 5.9: Fragmento de código de pruebas del componente de subida de documentos.
Fuente: Elaboración propia

5.4.1.3. Módulo para reservar almuerzo como estudiante

Las pruebas unitarias desarrolladas para este componente validan principalmente los siguientes aspectos:

- La correcta inicialización del componente y sus formularios.

- La actualización del estado del sistema cuando el usuario selecciona una sede o periodo académico.
- La validación de los campos del formulario de búsqueda, asegurando que se muestren mensajes adecuados cuando hay campos incompletos o inválidos.
- El comportamiento del sistema cuando se realiza una búsqueda exitosa, incluyendo la visualización y gestión de los resultados (almuerzos) obtenidos en días habilitados para el estudiante.
- La capacidad de enviar el recordatorio de QR cuando existen datos disponibles, así como la notificación adecuada cuando no los hay.
- La limpieza del formulario y el estado del componente cuando el usuario decide reiniciar la búsqueda.
- El manejo apropiado de errores provenientes de los servicios asociados, sin afectar la estabilidad del sistema.

En la figura 5.10 se muestra un fragmento del código de las pruebas del componente.

Figura 5.10: Fragmento de código de pruebas del servicio de almuerzos.
Fuente: Elaboración propia

5.4.1.4. Módulo para habilitar estudiantes a recibir almuerzos

Las pruebas unitarias desarrolladas para este módulo validan principalmente los siguientes aspectos:

- La correcta inicialización del formulario vacío, del componente y de la grilla de estudiantes.
- La respuesta esperada de una correcta e incorrecta utilización del formulario y respuestas esperadas ante fallos al recibir o no recibir datos.

- La validación de mostrar al usuario la información correcta recibida desde le servicio de almuerzos con la información de cada estudiante que coincida con la búsqueda.
- La persistencia de las actualizaciones realizadas por el usuario al modificar los días habilitados de un estudiante o deshabilitarlo completamente y posteriormente consultarlo nuevamente.

En la figura 5.8 se muestra un fragmento del código de las pruebas minimizadas del componente con el fin de abarcar una mayor cantidad en la evidencia.

Figura 5.11: Fragmento de código de pruebas del componente de habilitación de estudiantes para almuerzos.

Fuente: Elaboración propia

5.4.1.5. Módulo para validar documentos subidos por estudiantes para recibir almuerzos

Las pruebas unitarias desarrolladas para este módulo validan principalmente los siguientes aspectos:

- La correcta inicialización del formulario vacío, del componente y de la grilla de documentos subidos por los estudiantes.
- La respuesta esperada de una correcta e incorrecta utilización del formulario y respuestas esperadas ante fallos al recibir o no recibir datos de estudiantes.
- La validación de mostrar al usuario la información correcta recibida desde le servicio de almuerzos con la información de cada documento que coincida con la búsqueda y haya sido subido a la base de datos.
- La persistencia de las actualizaciones realizadas por el usuario al modificar los documentos validados de un estudiante y posteriormente consultarlo nuevamente.
- La correcta descarga para consulta de los documentos subidos por los estudiantes.
- El correcto funcionamiento de la funcionalidad de generar una reclamación sobre algún documento subido por algún estudiante.

En la figura 5.12 se muestra un fragmento del código de las pruebas minimizadas del componente con el fin de abarcar una mayor cantidad en la evidencia.

Figura 5.12: Fragmento de código de pruebas del componente de validación de documentos subidos estudiantes.

Fuente: Elaboración propia

En total el número de pruebas en correcta ejecución e implementación para el flujo completo de almuerzos fue de **96**, a continuación en la figura 5.13 se muestran las pruebas ejecutadas y aprobadas.

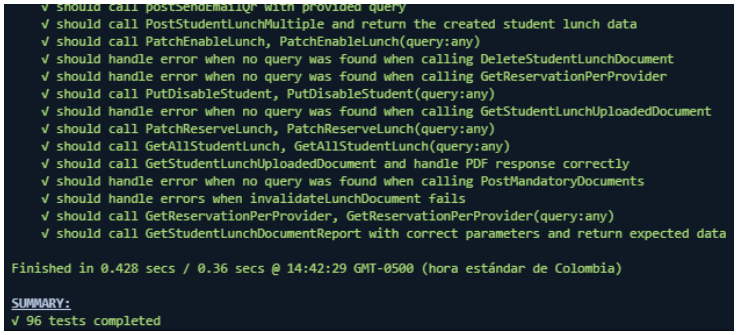


Figura 5.13: Resumen de ejecución de pruebas del flujo de almuerzos.
Fuente: Elaboración propia

5.4.2. Módulo de Estudiantes

El módulo de estudiantes de *AcademiApp*, preexistente al desarrollo actual, agrupa un conjunto de funcionalidades orientadas a facilitar la gestión de diversos procesos académicos desde el perfil del estudiante. Dentro de este módulo se encuentra el submódulo **Almuerzo**, el cual incluye la opción de **Reservar Almuerzo**, diseñada para permitir a los estudiantes registrar sus solicitudes de alimentación de manera individual por día.

Como parte del presente trabajo, se realizaron modificaciones significativas al flujo de esta funcionalidad, permitiendo ahora la reserva de almuerzos por bloques de días dentro de una misma semana. Este ajuste mejora la eficiencia del proceso y la experiencia de usuario, reduciendo la necesidad de múltiples interacciones para registrar reservas consecutivas.

Adicionalmente, se desarrolló un nuevo submódulo denominado **Subir Documentos**, el cuál será detallado y evidenciado más a profundidad en la sección **5.4.2.4**, integrado dentro del módulo de **Almuerzo**. Este componente permite a los estudiantes cargar los documentos requeridos para acceder al subsidio de alimentación, los cuales son definidos previamente como obligatorios para un periodo académico y sede específicos. La implementación de este submódulo automatiza y centraliza la validación documental, facilitando tanto la gestión administrativa como la habilitación del beneficio para los estudiantes.

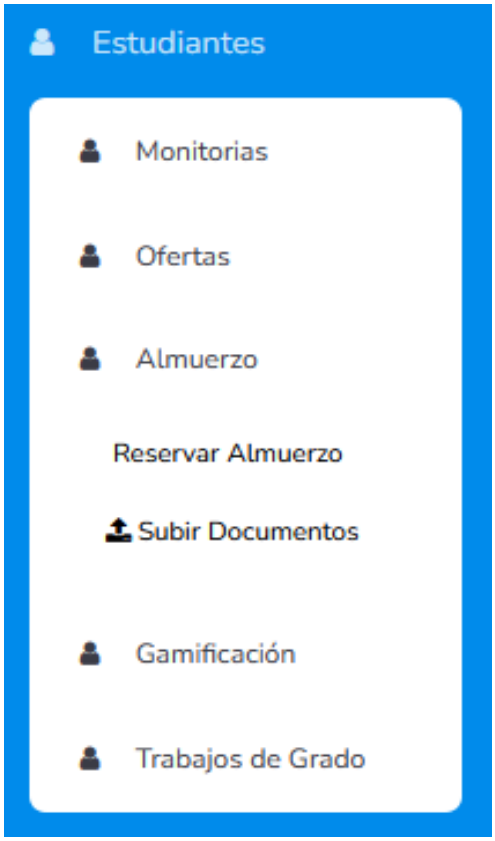


Figura 5.14: Vista del submódulo de Almuerzo en el menú del módulo de Estudiantes.
Fuente: Elaboración propia

5.4.2.1. Visualización de Días Disponibles para Reservas

Previo al desarrollo de este trabajo, el sistema no brindaba al estudiante información respecto a los días en los que se encontraba habilitado para realizar reservas de almuerzo, lo cual podía derivar en intentos fallidos o confusos al utilizar la funcionalidad. Como mejora, se incorporó una lógica de validación que, al momento de diligenciar el formulario de reserva, verifica la sede y el periodo académico seleccionados por el usuario. En caso de que el día actual no esté habilitado para la reserva, el sistema presenta un mensaje informativo junto con un listado de los días autorizados en dicha sede y periodo, y bloquea el botón de *Reservar almuerzo para hoy*.

No obstante, el estudiante aún conserva la posibilidad de realizar su solicitud para fechas futuras de la misma semana en los días en los que se encuentre habilitado, a través del botón *Reservar almuerzo para varios días*, correspondiente a la funcionalidad de reserva por bloques implementada en este proyecto.

La figura 5.15 ilustra el aspecto general del submódulo de reserve de almuerzos como estudiante, la cuál se visualizará en una primera carga del componente.

Figura 5.15: Vista del componente de reserva de almuerzos como estudiante.
Fuente: Elaboración propia

En caso de que el día actual no esté habilitado para que el estudiante pueda generar una reserva de almuerzo en una configuración específica de sede y periodo, el sistema despliega un mensaje informativo mediante un modal, el cual se ilustra a continuación en la figura 5.16, en caso contrario, el estudiante podrá continuar con el flujo de la reserva normalmente.

Figura 5.16: Modal con mensaje de días habilitados para realizar reservas.
Fuente: Elaboración propia

5.4.2.2. Reserva de Almuerzos por Bloques

Al rellenar el formulario por completo, y presionar el botón *Reservar almuerzo para varios días*, se despliega el modal mostrado en la figura 5.17, en el cual se listan los días de la semana actual que se encuentran habilitados para realizar reservas, y en los cuales el estudiante aún no ha efectuado ninguna solicitud. A través de esta interfaz, el usuario puede seleccionar uno o varios días conforme a su necesidad.

Cabe resaltar que dicho botón permanecerá deshabilitado en caso de que el estudiante no cuente con ningún día habilitado para realizar reservas, de acuerdo con la sede y el periodo académico seleccionados. Esto asegura la validación previa de las condiciones necesarias, bloqueando por completo ambos métodos de reserva —tanto diaria como por bloques— cuando no exista disponibilidad para el estudiante dentro del sistema.

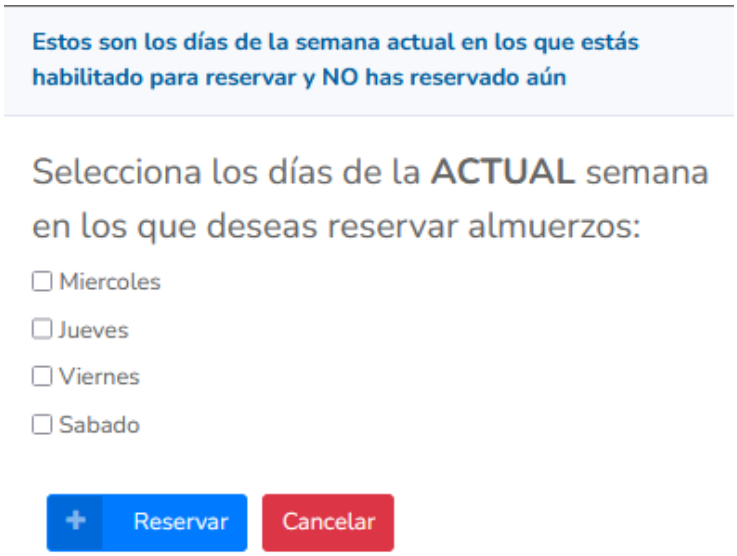


Figura 5.17: Modal con listado de días habilitados para realizar reservas por bloque.
Fuente: Elaboración propia

5.4.2.3. Recordatorio de Reserva de Almuerzo

Adicionalmente, en esta misma interfaz se habilitó un botón dentro de las acciones asociadas a cada reserva, específicamente junto al botón de cancelación con el texto *enviar*, que permite enviar un recordatorio al estudiante sobre la reserva seleccionada. Al utilizar esta opción, el sistema reenvía al correo electrónico registrado del estudiante toda la información relacionada con la reserva, incluyendo el código QR correspondiente, facilitando así el acceso y la verificación del servicio reservado.

En la figura 5.18 se muestra la apariencia de una reserva en la lista de reservas de la tabla 5.15, junto con sus acciones disponibles, incluyendo el botón implementado *enviar*.

Figura 5.18: Acciones disponibles a una reserva.
Fuente: Elaboración propia

El correo que se envía como recordatorio corresponde a la misma información que se envió al estudiante al momento de crear la reservación, la cual responde al formato de la figura 5.19.

Figura 5.19: Correo electrónico enviado como recordatorio al estudiante.
Fuente: Elaboración propia

5.4.2.4. Submódulo de Subir Documentos

El propósito de éste submódulo es permitir al usuario cargar los documentos requeridos para acceder al subsidio de alimentación universitario.

Para ello, el estudiante debe seleccionar primero la **sede** y el **periodo académico** correspondiente en el formulario. Con base en esta combinación, el sistema mostrará automáticamente el listado de documentos definidos como obligatorios para dicho contexto, tal como se muestra en la figura 5.20.

El estudiante podrá entonces adjuntar archivos desde su dispositivo local para cada uno de

los documentos listados. No es obligatorio completar la carga de todos los documentos en una sola sesión; el sistema permite que el usuario suba los archivos de manera progresiva, conforme los tenga disponibles. Sin embargo, para poder ser considerado candidato a la habilitación de días de almuerzo en ese periodo y sede, deberá completar la carga de todos los documentos solicitados.

Adicionalmente, el estudiante podrá consultar y descargar los archivos que ya haya subido para visualizarlos, con el fin de verificar que corresponden a los documentos correctos, especialmente en caso de haber sido notificado por el gestor de una inconsistencia o ante cualquier duda personal.

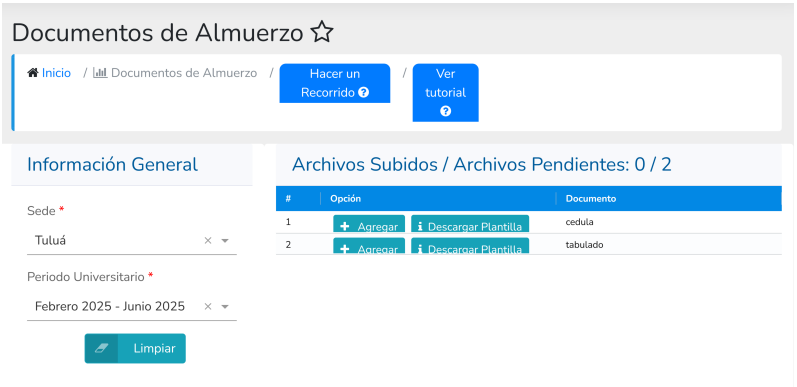


Figura 5.20: Vista del submódulo para subir documentos para subsidio de almuerzo.
Fuente: Elaboración propia

Una vez el estudiante ha cargado correctamente cada uno de los documentos obligatorios, el sistema le notifica mediante el mensaje de confirmación mostrado en la figura 5.21. A partir de ese momento, las opciones disponibles para cada archivo se actualizan a *ver*, que permite la descarga del documento previamente cargado, y *eliminar*, que ofrece la posibilidad de retirar el archivo del sistema. Esta última opción resulta útil en caso de que el estudiante deba reemplazar el archivo por otro, por ejemplo, ante una reclamación por haber subido un documento incorrecto.

Figura 5.21: Modal de confirmación de archivo subido con éxito.
Fuente: Elaboración propia

5.4.3. Módulo de Proveedores

El módulo para proveedores del servicio de almuerzo, nombrado dentro de la plataforma como **Almuerzos para Estudiantes** el cual también se encontraba previamente desarrollado en la plataforma, está diseñado para ser utilizado por los restaurantes o entidades encargadas de entregar los almuerzos subsidiados a los estudiantes beneficiarios del programa. Este módulo proporciona acceso a funcionalidades específicas que permiten llevar un control preciso y ordenado del proceso de entrega de los alimentos, asegurando trazabilidad y transparencia en la operación diaria.

Dentro de este módulo se encuentra el submódulo de **Despachar Almuerzos**, desde el cual los operadores pueden visualizar todas las reservas registradas por los estudiantes. La interfaz permite, tal como se evidencia en la figura 5.22, consultar información detallada de cada reserva, aplicar filtros por fecha y realizar acciones como:

- Modificar el estado de una reserva a **Despachada**, una vez el almuerzo ha sido entregado al estudiante.
- **Eliminar una reserva** cuando por motivos administrativos sea necesario anular una solicitud antes de ser atendida.

Estas funcionalidades están orientadas a brindar eficiencia y claridad en el proceso de control de entregas, facilitando la labor de los proveedores y previniendo inconsistencias en el sistema.

Figura 5.22: Vista general del módulo de Proveedores del Servicio de Almuerzo
Fuente: Elaboración propia

5.4.3.1. Restricción sobre la anulación de cancelaciones de reserva

Como parte de las mejoras de seguridad e integridad del sistema, se realizaron modificaciones en el módulo de **Despachar Almuerzos** utilizado por los restaurantes encargados de la entrega del beneficio alimentario.

En versiones anteriores del sistema, existía la posibilidad de **anular una cancelación de reserva**, lo cual permitía revertir el estado de una reserva previamente cancelada. Esta funcionalidad representaba un riesgo, ya que abría la posibilidad de que, de manera intencionada o maliciosa, se modificara el estado de una reserva cancelada con el fin de permitir que un estudiante recibiera más de un almuerzo en un mismo día.

Con el fin de evitar este tipo de inconsistencias, se eliminó la opción de anular cancelaciones desde el módulo de despacho. Una vez una reserva ha sido cancelada por el estudiante o por el proveedor, esta queda en un estado definitivo e irreversible, impidiendo su reutilización para recibir el beneficio para ese día y mostrando el mensaje de la figura 5.23 al estudiante al intentar reservar nuevamente. Esta medida fortalece el control sobre la entrega de almuerzos y reduce la posibilidad de fraudes por parte de operadores del sistema.

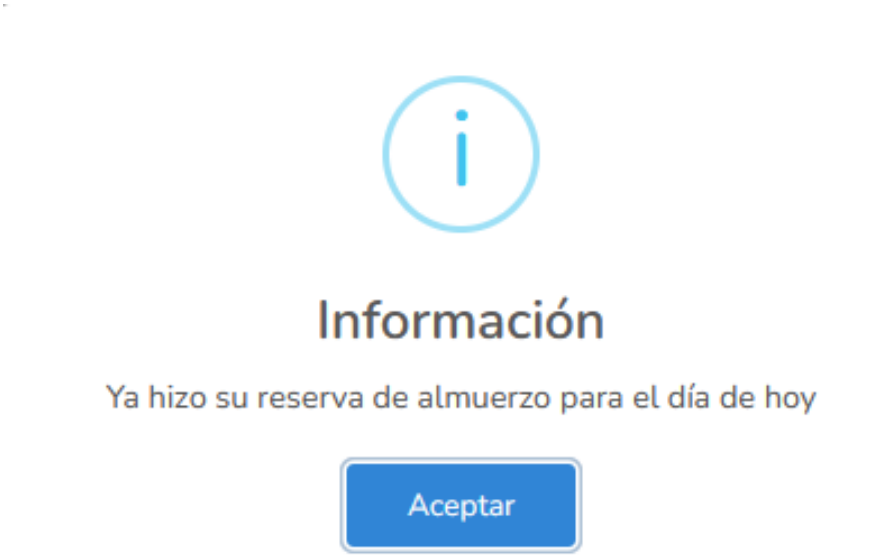


Figura 5.23: Modal mostrado al estudiante al intentar reservar posterior a una cancelación de su reserva.

Fuente: Elaboración propia

5.4.4. Módulo de Bienestar Estudiantil

Este módulo está orientado a la administración de servicios que optimicen la experiencia universitaria del estudiante desde el ámbito institucional. Dentro de este módulo se encuentran los submódulos *Gestión de Almuerzos de Estudiantes* y *Habilitar Estudiantes para Almuerzos*, tal como se muestra en la figura 5.24, los cuales ya se encontraban implementados previamente. Cada uno de estos submódulos dispone de una interfaz correspondiente que permite llevar a cabo una gestión controlada de los estudiantes autorizados para acceder al beneficio alimentario, así como de sus respectivas reservas.

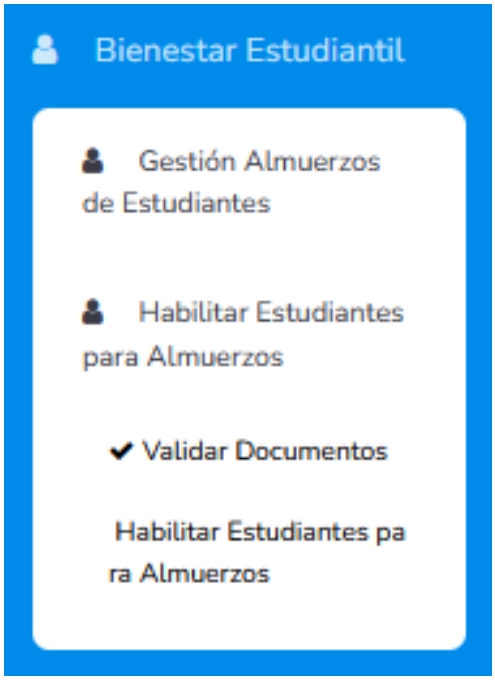


Figura 5.24: Menú principal del módulo de Bienestar Estudiantil.
Fuente: Elaboración propia

Como parte del proceso de mejora funcional del sistema, se incorporó un nuevo submódulo denominado *Validar Documentos*, a través del cual el gestor de bienestar puede marcar los documentos previamente cargados por los estudiantes como válidos o inválidos, o puede incluso generar un mensaje de reclamación al estudiante en cuestión para que verifique el archivo que subió y arregle cualquier discrepancia lo más pronto posible. Esta validación de los documentos constituye un paso previo necesario para que el estudiante pueda ser habilitado en el submódulo de *Habilitar Estudiantes para Almuerzos*, permitiéndole acceder al subsidio alimentario en los días correspondientes. Cabe destacar que este último submódulo ya se encontraba operativo desde versiones anteriores del sistema pero también se imeplementaron algunos cambios operacionales y mejoras las cuales se mostrarán a continuación.

5.4.4.1. Validar Documentos Subidos para Subsidio de Almuerzo

El submódulo *Validar Documentos* permite al gestor del área de Bienestar Estudiantil realizar el control y validación de los archivos cargados por los estudiantes como requisito para acceder al beneficio de alimentación.

La interfaz proporciona filtros por sede y por período académico, permitiendo al gestor delimitar la búsqueda de documentos. Al ejecutar la acción de búsqueda, se presenta una tabla con los documentos correspondientes a la combinación seleccionada, facilitando su revisión.

Cada fila de la tabla contiene información relevante sobre el documento cargado, y ofrece al gestor las siguientes acciones:

- **Ver:** descarga el archivo cargado para su verificación.

- **Reclamar Inconsistencia:** permite emitir una reclamación al estudiante en caso de que el archivo presente errores o no cumpla con los requisitos establecidos. Esta acción genera un correo automático al estudiante, notificando la inconsistencia detectada.
- **Validar / Invalidar:** habilita la posibilidad de marcar un documento como válido o inválido según corresponda.

Cabe resaltar que todos los documentos definidos como obligatorios para una sede y período determinados deben encontrarse en estado *validado* para que el estudiante sea considerado como candidato elegible y se refleje su postulación en el submódulo de *Habilitar Estudiantes para Almuerzos*.

Adicionalmente, la tabla puede ordenarse dinámicamente por fecha de carga del documento, nombre del estudiante o nombre del archivo obligatorio tal como se muestra en la figura 5.25, facilitando así una gestión más eficiente y personalizada del proceso de validación.

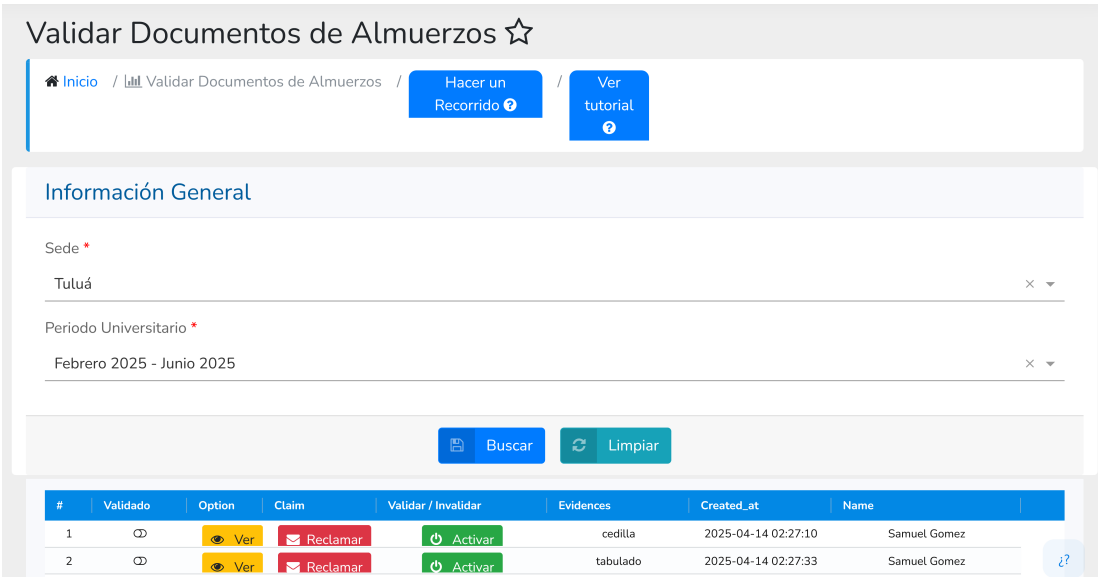


Figura 5.25: Vista del submódulo de validación de documentos para subsidio de almuerzo.
Fuente: Elaboración propia

Al presionar el botón de *Reclamar*, se le muestra el mensaje de confirmación de la figura 5.26 al usuario, cuestionando si realmente quisiera enviar el correo electrónico de reclamación al estudiante.

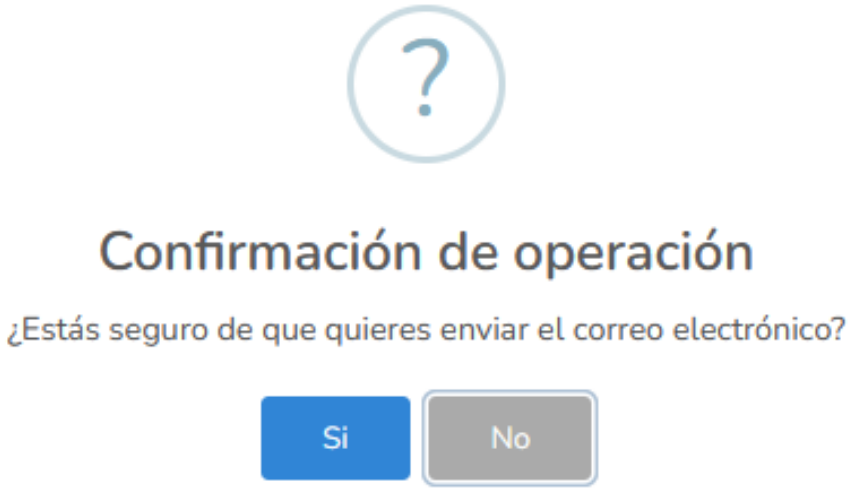


Figura 5.26: Modal de confirmación de envío de correo de reclamación.
Fuente: Elaboración propia

El correo recibido por el estudiante el cual es dueño del archivo subido el cual fue reclamado como inconsistente tendría el aspecto que se ve en la figura 5.27, el cual contiene la información pertinente para ayudar al estudiante a identificar de manera rápida qué documento cuenta con una irregularidad, además de expresarle que debe ser corregido para poder ser activado como beneficiario.

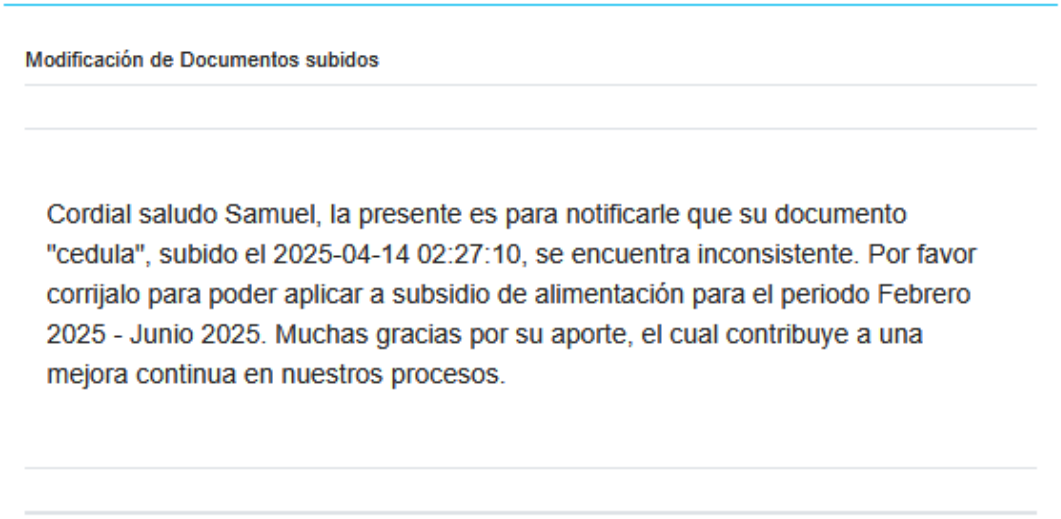


Figura 5.27: Correo recibido por el estudiante como reclamación.
Fuente: Elaboración propia

5.4.4.2. Habilitar Estudiantes para Almuerzo

El submódulo *Habilitar Estudiantes para Almuerzo* permite al gestor de Bienestar Estudiantil gestionar qué estudiantes serán beneficiarios del subsidio alimentario, habilitándoles para realizar reservas de almuerzos en días específicos.

Aunque este submódulo ya se encontraba implementado y operativo previamente, fue sometido a una modificación clave en su lógica de funcionamiento: ahora, únicamente se listan como candidatos para habilitación aquellos estudiantes que hayan completado satisfactoriamente la carga de todos los documentos obligatorios correspondientes a la sede y período académico seleccionados en el formulario de consulta. Anteriormente, el sistema mostraba a todos los estudiantes registrados, al haber sido implementada la funcionalidad de subir y validar documentos, se implementó también esta característica en la búsqueda. Esta mejora permite garantizar que sólo los estudiantes que cumplen con los requisitos documentales puedan acceder al subsidio.

El resto del proceso dentro del módulo se mantiene conforme a la lógica original: al presionar el botón *Agregar* en la fila correspondiente a un estudiante, se despliega un modal en el cual el gestor puede seleccionar los días de la semana en que el estudiante estará habilitado para reservar almuerzos. Una vez seleccionados, estos días quedan registrados en el sistema.

En la figura 5.28 se muestra la apariencia general de módulo la cual no fue modificada.

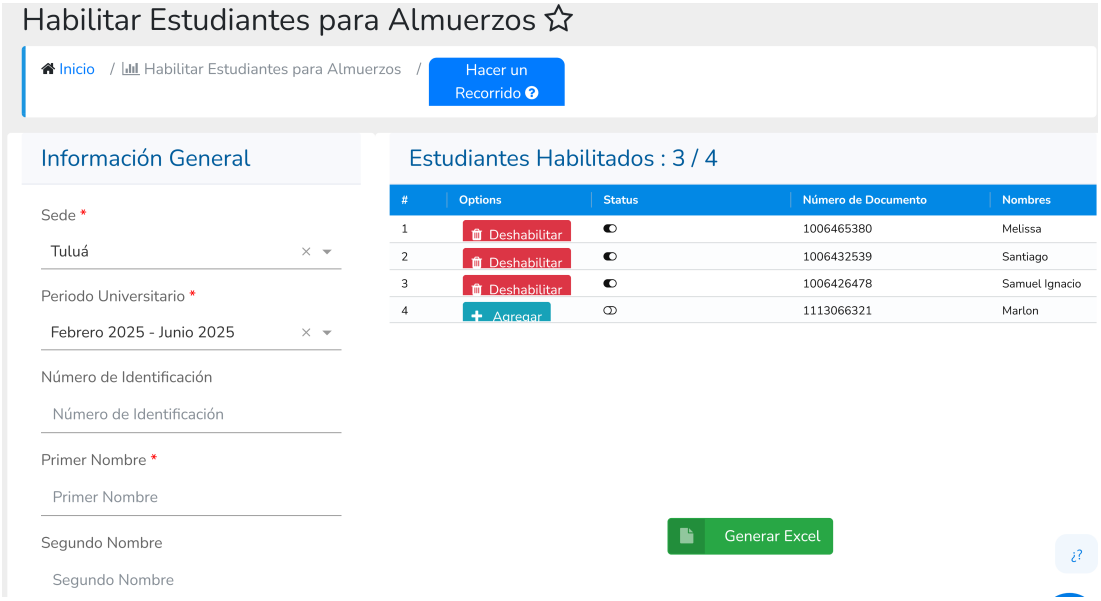


Figura 5.28: Vista del submódulo de habilitación de estudiantes para almuerzo.
Fuente: Elaboración propia

Adicionalmente a esta validación mencionada, se mejoró también la retroalimentación para el usuario gestor de bienestar, al implementar el mensaje de confirmación de la figura 5.29 al habilitar, y el de la figura 5.30 al deshabilitar correctamente a un estudiante para

recibir subsidio alimentario.

Éxito

Estudiante habilitado para almuerzos.

Figura 5.29: Modal de éxito al habilitar un estudiante para almuerzos.
Fuente: Elaboración propia

Éxito

Estudiante deshabilitado para almuerzos.

Figura 5.30: Modal de éxito al deshabilitar un estudiante para almuerzos.
Fuente: Elaboración propia

De igual manera, previamente no existía un patrón de carga en el componente al momento de habilitar o deshabilitar un estudiante, por lo cual, si la petición se tardaba un poco más de lo esperado en realizarse, el usuario veía datos sin actualizar en la tabla de consulta, por lo que se implementó el patrón de carga de la plataforma, en este componente, como se muestra en la figura 5.31.

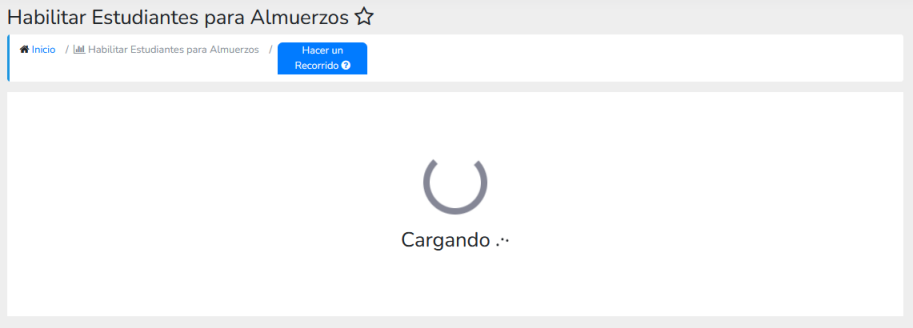


Figura 5.31: Patrón de carga al habilitar o deshabilitar estudiante.
Fuente: Elaboración propia

Otra mejora significativa implementada en este submódulo corresponde al reporte de exportación en formato Excel, generada desde el botón de *Generar Excel* ubicado en la parte inferior de la tabla de resultados de consulta. Previamente, dicho archivo solo indicaba si un estudiante estaba o no habilitado. Con la nueva funcionalidad, ahora también se visualizan los días específicos para los cuales cada estudiante fue habilitado, lo cual mejora la trazabilidad y el control sobre el proceso de asignación de subsidios.

En la figura 5.32 se muestra un ejemplo de un reporte exportado en formato Excel desde el sistema.

Reporte de almuerzos habilitados

Documento	Nombres	Apellidos	Estado	L	M	M	J	V	S
1006465380	Melissa null	Torres Velasquez	Habilitado	X	X	X	X	X	X
1006432539	Santiago	González Rodríguez	Habilitado	X	X	X	X	X	X
1006426478	Samuel Ignacio	Gomez Moreno	Habilitado			X	X	X	X
Informe generado por AcademiApp									

Figura 5.32: Reporte Excel generado con visualización de días.
Fuente: Elaboración propia

5.5. Módulo de Trabajos de Grado

El módulo de trabajos de grado está diseñado para gestionar de forma integral los procesos asociados a los proyectos de grado dentro de la institución. Este módulo centraliza y organiza la información y las acciones necesarias para el correcto seguimiento, validación y acompañamiento académico de estos trabajos. Se encuentra dividido en varias interfaces según el tipo de usuario:

- **Interfaz para docentes:** Permite consultar los trabajos de grado en los que participan como asesores, acceder a documentos relacionados, y realizar seguimiento de las actividades y asesorías asociadas.
- **Interfaz para estudiantes:** Desde esta vista, los estudiantes pueden crear su trabajo de grado, gestionar el envío de documentos para cada etapa del proceso, registrar sus asesorías y hacer seguimiento a las observaciones recibidas.

- **Interfaz para gerencia académica:** Dirigida a los usuarios encargados de la supervisión institucional del proceso de trabajos de grado. Desde aquí es posible consultar el estado detallado de cada trabajo, así como habilitar o deshabilitar trabajos de grado de acuerdo con criterios académicos o administrativos.

Este módulo busca garantizar trazabilidad, transparencia y eficiencia en cada una de las etapas del proceso de grado, facilitando la colaboración entre los distintos actores involucrados.

5.5.1. Pruebas Unitarias del Módulo de Trabajos de Grado

Con el objetivo de fortalecer la calidad del software y garantizar la fiabilidad del sistema, se procedió a la implementación integral de pruebas unitarias sobre el módulo completo de trabajos de grado, abarcando no solo los componentes individuales, sino también el flujo general del servicio que articula su funcionalidad. Cabe resaltar que dicho módulo no contaba previamente con ningún tipo de verificación de calidad, lo cual representaba un riesgo potencial en términos de mantenibilidad y detección temprana de errores. La inclusión de estas pruebas permitió establecer una base sólida de validación que asegura el correcto comportamiento de cada una de las funcionalidades involucradas, facilitando al mismo tiempo su evolución futura y reduciendo significativamente la probabilidad de regresiones en escenarios posteriores de desarrollo.

A continuación en la figura 5.33 se muestra el reporte de cobertura de las pruebas unitarias generado en la ejecución de las mismas, evidenciando el cumplimiento de su criterio de aceptación general.

File		Statements		Branches		Functions		Lines	
environments		100%	1/1	100%	0/0	100%	0/0	100%	1/1
app/layout/degree-works/management-degree-works/view-degree-works		90.28%	251/278	65.21%	60/92	87.83%	65/74	90.37%	244/270
app/layout/degree-works/management-degree-works		86.81%	191/220	56.66%	34/60	85%	51/60	87.32%	186/213
app/layout/degree-works/management-degree-works/search-degree-works		85.14%	86/101	41.17%	7/17	75%	24/32	84.53%	82/97
app/layout/degree-works/management-degree-works/search-all-degree-works		85.02%	142/167	22.72%	10/44	76.08%	35/46	85.97%	141/164
app/layout/degree-works/management-degree-works/edit-degree-works		81.73%	264/323	53.57%	45/84	68.6%	59/86	83.12%	261/314
app/services/degree-works		80.94%	327/404	20%	10/50	76.69%	79/103	81.2%	311/383

Figura 5.33: Reporte de cobertura de pruebas unitarias de componentes y servicio de Trabajos de Grado.

Fuente: Elaboración propia

5.5.1.1. Servicio de comunicación con el Back-End

Las pruebas unitarias desarrolladas para este servicio validan principalmente los siguientes aspectos:

- La correcta inicialización del servicio para su correcto uso en los componente relevantes.
- Se verificó que las respuestas recibidas desde el Back-End coincidieran con los resultados esperados para cada solicitud enviada a la API.

- Se evaluó la validez de los mensajes retornados, los códigos de error asociados y la cantidad de llamadas generadas hacia la API.
- Se analizó el comportamiento del sistema al realizar solicitudes vacías o sin parámetros, observando la gestión de estos casos por parte del servicio.

En la figura 5.34 se muestra un fragmento del código de las pruebas en formato minimizado del servicio con el fin de abarcar una mayor cantidad en la evidencia.

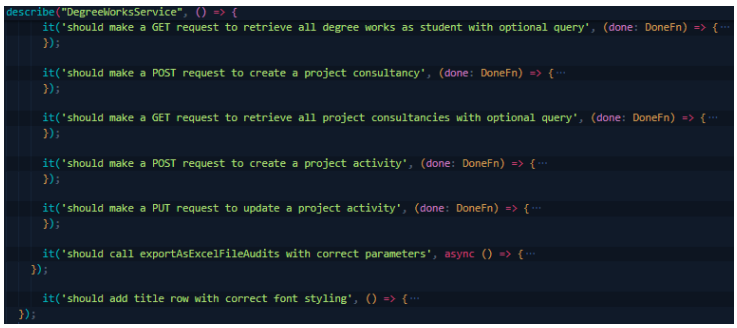


Figura 5.34: Fragmento de código de pruebas del servicio de trabajos de grado.
Fuente: Elaboración propia

5.5.1.2. Módulo de visualización de un trabajo de grado

A continuación, se detallan los principales casos de prueba implementados en este componente, los cuales a su vez, se subdividieron en muchos casos en más pruebas para cada caso de uso por el usuario:

- La correcta inicialización del componente, formularios principales y de los datos de los trabajos de grado, asegurando su estructura y valores por defecto antes de la consulta al trabajo de grado que se está visualizando.
- EL funcionamiento esperado al usuario modificar algún valor del trabajo de grado y guardar los cambios.
- La visualización y reclamación de documentos subidos como actividades de nivel de un trabajo de grado con detalle visual de fecha de subida al usuario.
- La correcta visualización de consultorías realizadas en el trabajo de grado.
- El funcionamiento esperado del botón de desactivación de un trabajo de grado.
- La correcta exportación de un trabajo de grado.

En la figura 5.35 se muestra un fragmento del código de las pruebas del componente.

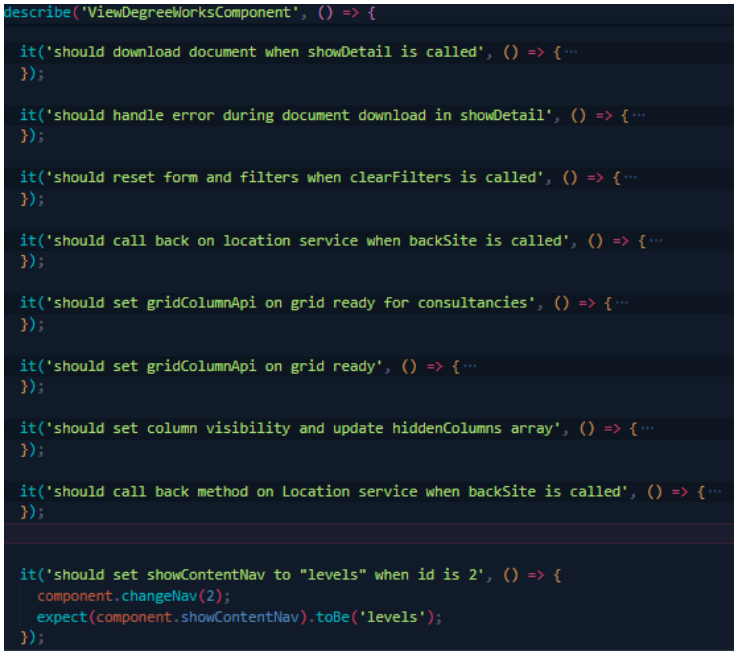


Figura 5.35: Fragmento de código de pruebas del componente de visualización de un trabajo de grado.

Fuente: Elaboración propia

5.5.1.3. Módulo de búsqueda de trabajos de grado como gerente académico

A continuación, se detallan los principales casos de prueba implementados en este componente, los cuales a su vez, se subdividieron en muchos casos en más pruebas para cada caso de uso por el usuario:

- La correcta inicialización del componente, formularios principales y de la grilla de trabajos encontrados, que sus valores por defecto antes de la búsqueda sean nulos.
- EL funcionamiento esperado de los distintos campos de búsqueda posibles.
- La visualización correcta y métodos de ordenamiento de la grilla de trabajos de grados encontrados.

En la figura 5.36 se muestra un fragmento del código de las pruebas del componente.

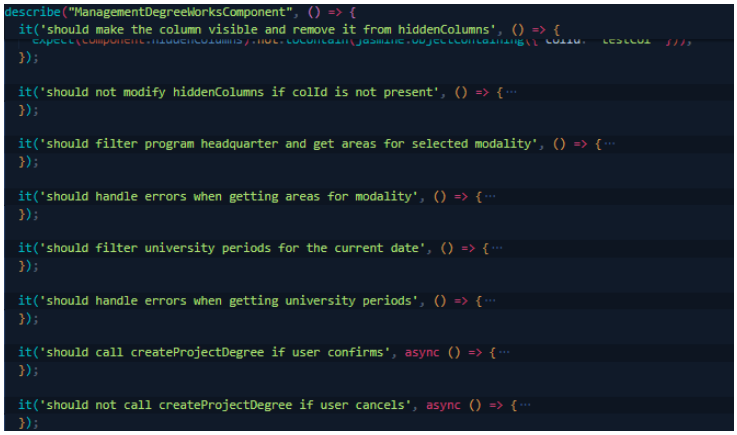


Figura 5.36: Fragmento de código de pruebas del componente de búsqueda de trabajos de grado como gerente académico.

Fuente: Elaboración propia

5.5.1.4. Módulo de búsqueda de trabajos de grado como estudiante

A continuación, se detallan los principales casos de prueba implementados en este componente, los cuales a su vez, se subdividieron en muchos casos en más pruebas para cada caso de uso por el usuario, el objetivo general del módulo es muy similar al anterior, pero con un objetivo de búsqueda distinto:

- Se verificó que el componente, así como sus formularios principales y la grilla de resultados, se inicialicen correctamente, asegurando que los valores por defecto sean nulos antes de ejecutar una búsqueda.
- Se evaluó el comportamiento de los diferentes campos de búsqueda, comprobando que respondan de manera adecuada según lo esperado.
- Se confirmó la correcta visualización de la grilla de trabajos de grado, junto con el funcionamiento adecuado de los métodos de ordenamiento disponibles.

En la figura 5.37 se muestra un fragmento del código de las pruebas del componente. El módulo de búsqueda de trabajos de grado propios del estudiante es en principio con la misma funcionalidad, pero este solo busca entre los trabajos de grado propios del estudiante, en comparación con el módulo mencionado en este punto, el cual busca también entre los de los demás estudiantes.

```
describe("ManagementDegreeWorksComponent", () => {
  it('should clear the table data and reset items at the start', () => {···
  });

  it('should mark form as touched and show an error message if the form is invalid', () => {···
  });

  it('should set "id_modality" and call search with correct filters if the form is valid', () => {···
  });

  it('should correctly configure the title, header, and fileName', () => {···
  });

  it('should remove duplicate periods before exporting', () => {···
  });

  it('should handle an empty data_excel array without errors', () => {···
  });

  it('should assign the event object to objMeeting', () => {···
  });

  it('should call setPQRSHeaders after view init', () => {···
  });

  it('should reset the form and clear the universityPeriods array', () => {···
  });
```

Figura 5.37: Fragmento de código de pruebas del componente de búsqueda de trabajos de grado como estudiante.
Fuente: Elaboración propia

En total el número de pruebas en correcta ejecución e implementación para el flujo completo de trabajos de grado fue de **59**, a continuación en la figura 5.38 se muestran las pruebas ejecutadas y aprobadas.

```
    √ should call getProgramsModalities without query and return expected result
    √ should make a PUT request to update a specific project degree by id
    √ should make a PUT request to deactivate a specific project degree by id
    √ should call postSetupProject
    √ should call getSetupProject without query and return expected result
    √ should call postSetupProject without query and return expected result

Finished in 0.015 secs / 0.076 secs @ 18:00:31 GMT-0500 (hora estándar de Colombia)

SUMMARY:
√ 59 tests completed
```

Figura 5.38: Resumen de ejecución de pruebas del flujo de trabajos de grado.
Fuente: Elaboración propia

5.5.2. Módulo de Profesores

Dentro del módulo de profesores, el docente ya contaba con algunas funcionalidades existentes a su disposición, tal como buscar un trabajo de grado para consultar sus documentos cargados y detallar más información a cerca de los mismos, de los cuales se haga parte.

5.5.2.1. Filtro de Personas

Se ha incorporado el filtro adicional mostrado en la figura 5.39 por nombre o apellido de personas en el módulo de trabajos de grado, tanto para el rol de estudiante como para el de profesor. Esta funcionalidad permite refinar significativamente las búsquedas dentro del sistema, especialmente en aquellos casos en los que múltiples trabajos de grado presentan títulos similares o temáticamente relacionados. Al permitir la búsqueda por

nombre o apellido, se mejora la precisión en la identificación de los trabajos, facilitando la trazabilidad y la gestión de los mismos dentro del entorno académico. Esta mejora responde a la necesidad de optimizar la usabilidad del sistema y garantizar una experiencia de usuario más eficiente, reduciendo ambigüedades y tiempos de consulta.

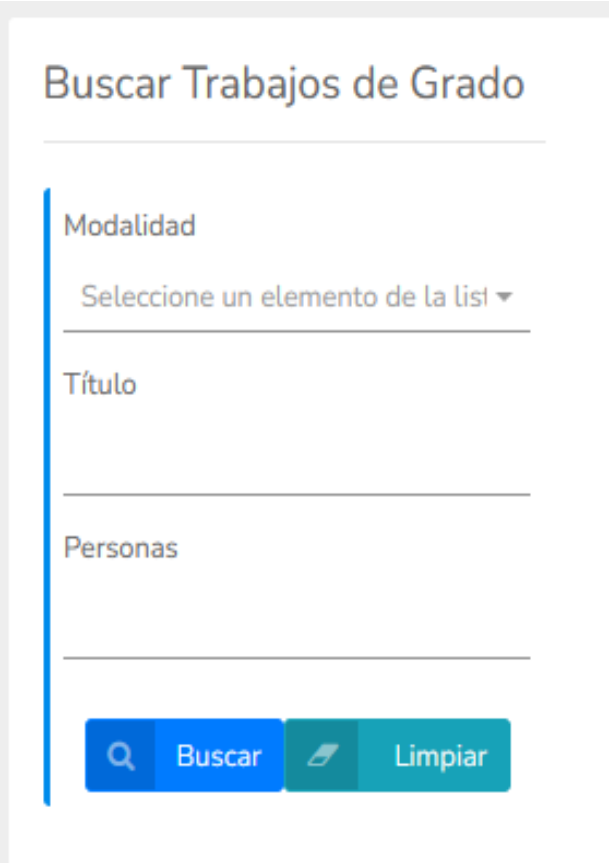


Figura 5.39: Filtros de búsqueda de Trabajos de Grado.
Fuente: Elaboración propia

5.5.2.2. Evaluación de Trabajos de Grado

Se ha incorporado un sistema de evaluación de trabajos de grado que formaliza y organiza el proceso mediante el cual los docentes asignados como evaluadores pueden revisar y calificar los trabajos académicos. Este sistema busca estandarizar las etapas del proceso evaluativo, facilitar la gestión de información y mejorar la trazabilidad de cada evaluación realizada dentro de la plataforma institucional.

El docente evaluador cuenta con una sección dedicada dentro del sistema, accesible desde el menú principal. A través de la ruta **Profesores** > **Trabajos de Grado** > **Evaluar**, el docente puede ingresar a un entorno donde se listan todos los trabajos de grado que le han sido asignados para su análisis.

Desde esta vista, tal como se muestra en la figura 5.40, el evaluador tiene la posibili-

dad de seleccionar el trabajo correspondiente y hacer clic en el botón Evaluar, lo cual desplegará toda la información a cerca del trabajo de grado, y un formulario estructurado que debe ser diligenciado conforme a los criterios establecidos por la coordinación académica.

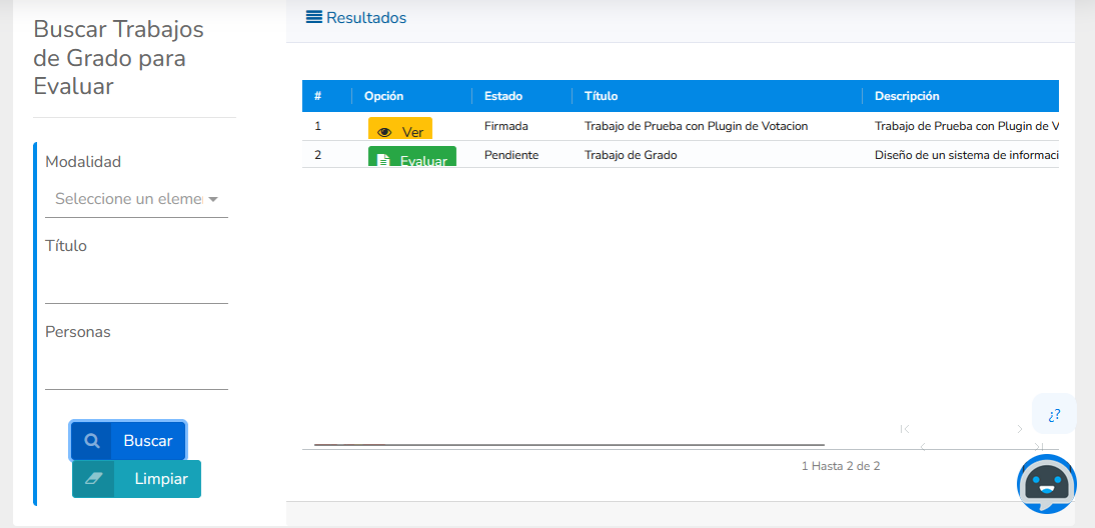


Figura 5.40: Vista de búsqueda de Trabajos de Grado asignados.
Fuente: Elaboración propia

Tal como se evidencia en la figura 5.41, el sistema permite tres acciones principales durante este proceso:

Guardar avances, lo que ofrece la flexibilidad de completar la evaluación en varias sesiones sin pérdida de información.

Finalizar la evaluación y enviarla al coordinador, quien revisará el contenido y podrá, según el caso, retornar el formulario al docente para ajustes o continuar con el flujo de validación correspondiente.

Generar su documento de evaluación en formato Excel, el cual podrá conservar para su reenvío a algún remitente que sea pertinente para el proceso.

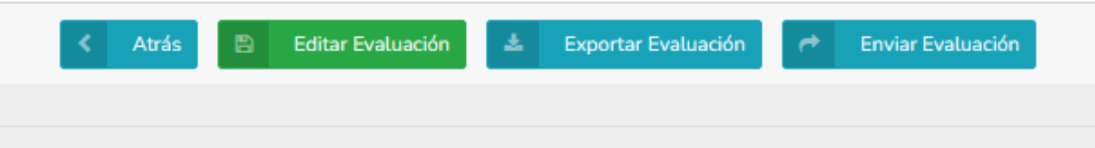


Figura 5.41: Acciones disponibles en evaluación de un Trabajo de Grado.
Fuente: Elaboración propia

Una vez el docente evaluador, finalice la evaluación con el botón **Enviar Evaluación**,

esta ya no será editable, y pasará a solamente poder visualizarla y exportarla en Excel como se muestra en al figura 5.42, momento donde el coordinador pasará a ser el encargado del flujo de evaluación.

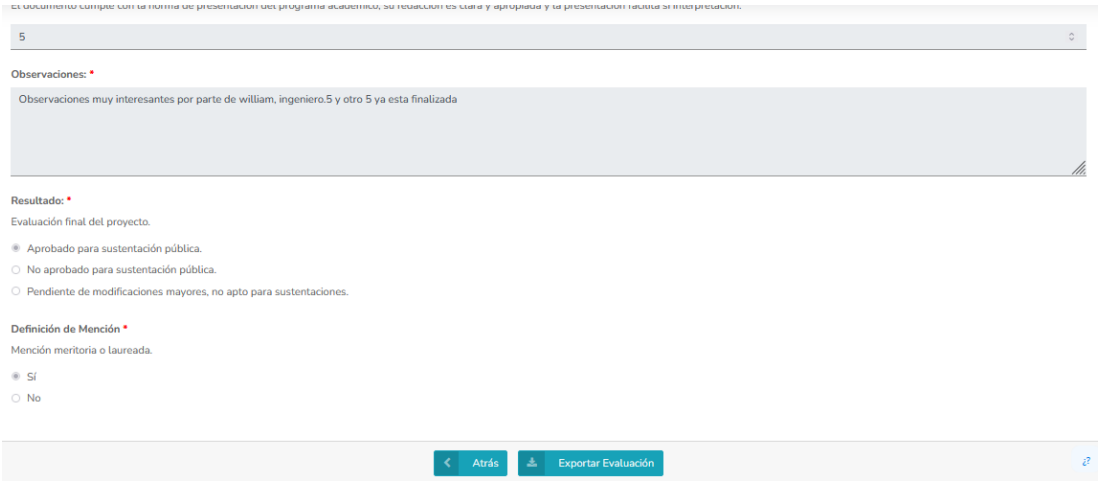


Figura 5.42: Acciones disponibles en evaluación de un Trabajo de Grado ya Evaluado.
Fuente: Elaboración propia

Como parte de la automatización del sistema, una vez que el docente completa y envía la evaluación, el coordinador que realizó la asignación recibe una **notificación automática por correo electrónico**, informándole del avance. Esto permite mantener una comunicación efectiva y asegurar el cumplimiento de los tiempos y etapas definidos en el proceso de evaluación académica.

5.5.3. Módulo de Estudiantes

Dentro de este módulo los estudiantes pueden buscar sus propios trabajos de grado, registrar uno nuevo, y además consultar en la base de datos por los trabajos de grado ya registrados para informarse de posibles coincidencias con el que estén desarrollando en el momento.

5.5.3.1. Búsqueda de Coincidencias en Títulos de Trabajos de Grado

Se ha implementado un mecanismo de validación automática en el proceso de creación de nuevos trabajos de grado por parte de los estudiantes. Esta funcionalidad tiene como objetivo evitar la duplicidad de temas y promover la originalidad de los trabajos registrados en el sistema.

Durante el registro de un nuevo trabajo, el sistema realiza una comparación entre el título ingresado por el estudiante y los títulos ya existentes en la base de datos. Esta comparación se lleva a cabo mediante un algoritmo que calcula el porcentaje de similitud textual entre títulos. En caso de que se detecte una coincidencia igual o superior al 50 %, el sistema genera una alerta modal que se presenta de forma inmediata al usuario.

Esta alerta notifica al estudiante sobre la posible similitud con trabajos previamente registrados, le muestra los títulos que coinciden con el especificado por el estudiante y le solicita tomar una decisión: confirmar si desea continuar con el registro del trabajo tal como está o editar el título antes de proseguir. De esta manera, se ofrece un control preventivo sin restringir completamente la libertad del estudiante para definir su propuesta.

Si no se identifican coincidencias significativas —es decir, inferiores al umbral del 50 %—, el proceso de creación del trabajo de grado continúa de manera normal, permitiendo el registro sin intervención adicional.

Al presionar el botón guardar en el formulario de registro, el estudiante y encontrar una similitud que se encuentre en el porcentaje, el modal que se visualizará será el de la figura 5.43.

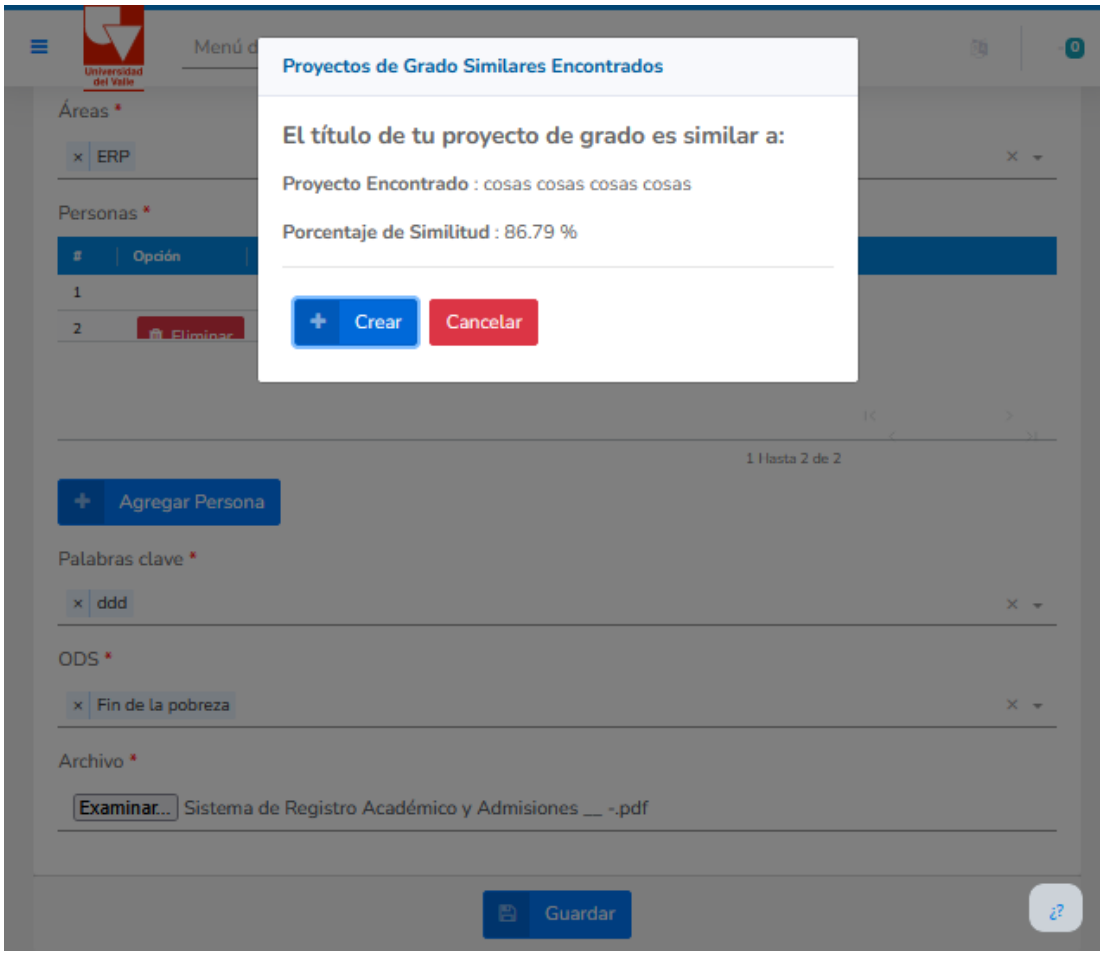


Figura 5.43: Modal de similitudes encontradas al crear Trabajo de Grado.
Fuente: Elaboración propia

Si el estudiante decide continuar con la creación sin importar las similitudes encontradas y presiona el botón **Crear**, recibirá el modal de confirmación de la figura 5.44

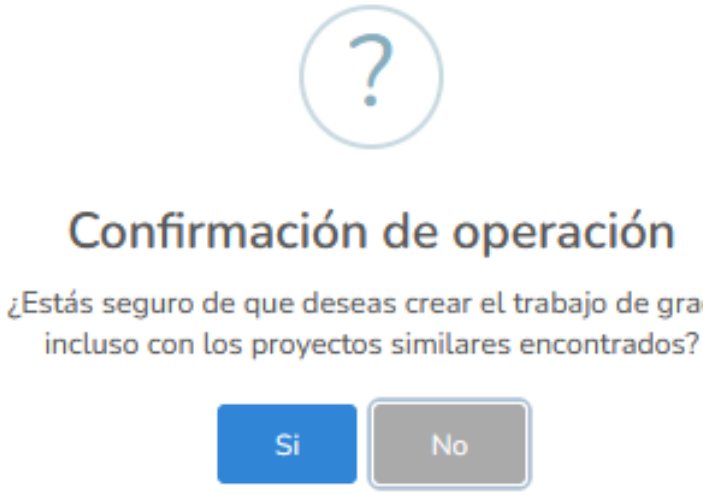


Figura 5.44: Modal de confirmación al crear Trabajo de Grado con similitudes.
Fuente: Elaboración propia

Si el estudiante decide que está seguro de esta acción, recibirá un modal de confirmación de creación como en la figura 5.45, caso contrario, será redirigido nuevamente al modal de información sobre las similitudes encontradas.

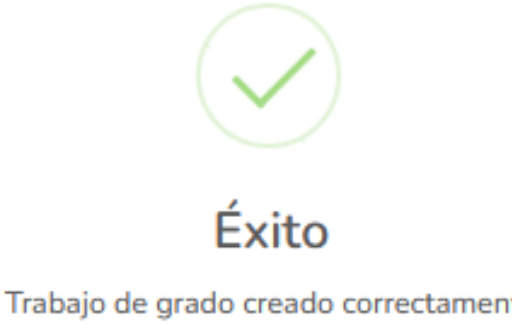


Figura 5.45: Modal de confirmación al de Trabajo de Grado creado con éxito.
Fuente: Elaboración propia

5.5.3.2. Restricción de selección de períodos en la creación de trabajos de grado

Con el objetivo de mantener la coherencia académica y garantizar la correcta asignación temporal de los trabajos de grado, se ha implementado una restricción en el proceso de creación de nuevos trabajos por parte de los estudiantes. A partir de esta mejora, el sistema únicamente mostrará los períodos académicos vigentes al momento de realizar el registro.

Esto significa que, durante la creación de un nuevo trabajo de grado, el estudiante solo podrá seleccionar en el desplegable, aquellos períodos que se encuentren activos y habilitados en el sistema en la fecha de ingreso, tal como se muestra en la figura 5.46, solo deberá mostrarse un periodo por consulta, el cual corresponde al actual. De este modo, se evita la posibilidad de asociar trabajos de grado a períodos pasados por error, lo cual podría generar inconsistencias administrativas y académicas, así como errores en la trazabilidad y seguimiento del proceso.

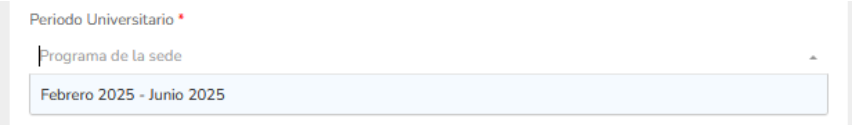


Figura 5.46: Desplegable de períodos vigentes Febrero - Junio 2025.
Fuente: Elaboración propia

5.5.4. Módulo de Gerencia académica

En el módulo de gerente o gestor académico, se cuenta con la posibilidad de buscar trabajos de grado y realizar ajustes y revisiones sobre los mismos, tales como verificar las actividades documentales requeridas por cada nivel evaluativo y verificar los documentos cargados por el o los proyectistas, a este módulo fueron agregadas algunas funcionalidades y mejoradas algunas partes de otras las cuales se expondrán a continuación.

5.5.4.1. Mejoras en el reporte de Excel para coordinadores

Se ha optimizado la funcionalidad de **exportación de datos en formato Excel** disponible para los coordinadores académicos dentro del módulo de trabajos de grado. Esta mejora tiene como objetivo proporcionar un reporte más completo, estructurado y útil para fines de **seguimiento**, **análisis** y **toma de decisiones**.

A partir de esta actualización, el archivo Excel generado incluirá nuevos campos relevantes, tales como la **modalidad del trabajo de grado**, la **fecha de publicación** del mismo y el **programa académico** al que pertenece. Estos datos permiten una visión más detallada de cada trabajo registrado y facilitan la **segmentación de la información** según criterios institucionales clave.

Adicionalmente, se incorpora una mejora en la visualización de los **niveles de evaluación** asociados a las actividades del trabajo de grado. Estos niveles únicamente serán visibles en el reporte cuando se aplique un **filtro por modalidad**, con el fin de evitar sobrecarga de información innecesaria en casos donde no sea relevante. Cuando el filtro está activo, los niveles se mostrarán de forma ordenada según su relevancia, permitiendo al coordinador interpretar rápidamente el estado y el de las evaluaciones asociadas, tal como se muestra en el ejemplo de la figura 5.47.

			Seminario de T.G					Trabajo de grado	Trabajo de grado 2				
Modalidad	Programa	Fecha de Registro	Carta Compromiso Di	Formato P.I.S.	Anteproyecto C	Carta Ev	Car	Formato de Segui	Carta Sol	Carta ent	Formato	Carta a E	Cart
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-23	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-23	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-23	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-23	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-23	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24											
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24	Pendiente Por validar	Pendiente Por val	Pendiente Por va	Pendiente Por validar							
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24	Pendiente Por validar	Pendiente Por validar									
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24	Pendiente Por validar										
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-25											
Proyecto de Ingenieria	Ingeniería de Sistemas	2022-06-24											

Figura 5.47: Columnas añadidas al Excel generado por coordinador.
Fuente: Elaboración propia

5.5.4.2. Corrección de errores críticos en la visualización de trabajos de grado

En el módulo de coordinación se corrigieron diversos errores que afectaban directamente la **funcionalidad** y la **experiencia de usuario** en la plataforma. Estas correcciones han permitido restablecer el comportamiento esperado del sistema, mejorando la estabilidad y facilitando la gestión académica. A continuación, se detallan los principales ajustes aplicados:

Inicialización de tablas: se resolvieron fallos que interrumpían la carga adecuada de las tablas dentro del módulo de trabajos de grado. Este problema provocaba que la información no se mostrara correctamente, afectando la navegación y manipulación de los datos. Con la corrección, las tablas ahora se renderizan de forma estable desde la inicialización.

Recarga del componente y renderización del sidebar: se solucionó un error que impedía la visualización del menú lateral (sidebar) al recargar componentes específicos, tal como se evidencia en la figura 5.48. Esto afectaba el acceso a otras secciones del sistema. La corrección asegura que el sidebar se renderice correctamente después de cada recarga, restaurando la navegación completa, como se evidencia en la figura 5.49.

A pesar de haber ya algunos módulos muy similares en la aplicación, este era el único que contaba con este problema, el cual interrumpía el correcto flujo de uso por parte del usuario, causado inicialmente por la estructura base de la aplicación, la cual no contaba con las suficientes validaciones de nulidad en algunos datos esenciales, lo cual se traducía en cambios inesperados en el comportamiento de la aplicación.

Figura 5.48: Menú lateral al recargar la vista **antes** de las correcciones.
Fuente: Elaboración propia

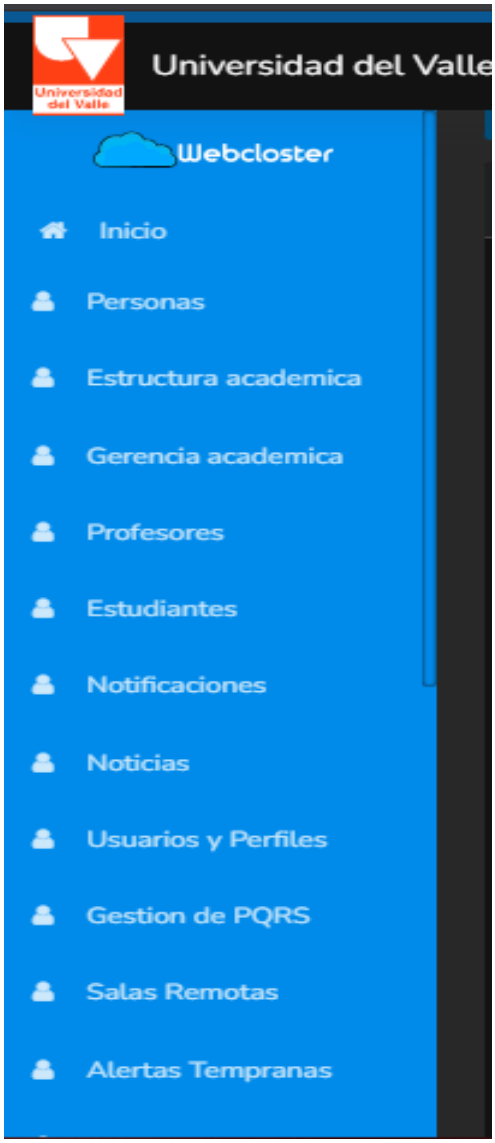


Figura 5.49: Menú lateral al recargar la vista **después** de las correcciones.
Fuente: Elaboración propia

5.5.4.3. Gestión de reclamación de actividades

Se ha incorporado una nueva funcionalidad en el **módulo de trabajos de grado** que permite al **gestor académico** intervenir en el flujo de revisión de las actividades marcadas como **cumplidas** por el estudiante o propietario del trabajo, como se evidencia en la figura 5.50. Esta función está disponible dentro de la pestaña **Niveles**, correspondiente al detalle del trabajo de grado.

Cuando el gestor identifique que una actividad no cumple con los **criterios académicos establecidos** —ya sea por errores en el contenido, incumplimiento de requisitos o inconsistencias en la documentación— podrá revertir su estado de **cumplida** a **rechazada**. Para ello, deberá acceder a la pestaña Niveles y hacer clic en el botón **Rechazar**

correspondiente a la actividad en cuestión.

Una vez ejecutada esta acción, el sistema actualizará automáticamente el estado de la actividad y generará una **notificación por correo electrónico** al estudiante, informándole sobre la observación realizada. El mensaje indicará que debe **corregir y volver a subir la actividad**, conforme a los lineamientos definidos.

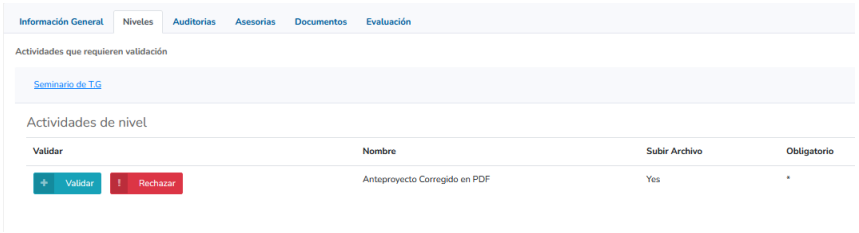


Figura 5.50: Pestaña de actividades completadas.
Fuente: Elaboración propia

El correo que recibirá el estudiante o dueño del trabajo de grado, tendrá el aspecto que se refleja en la figura 5.51.

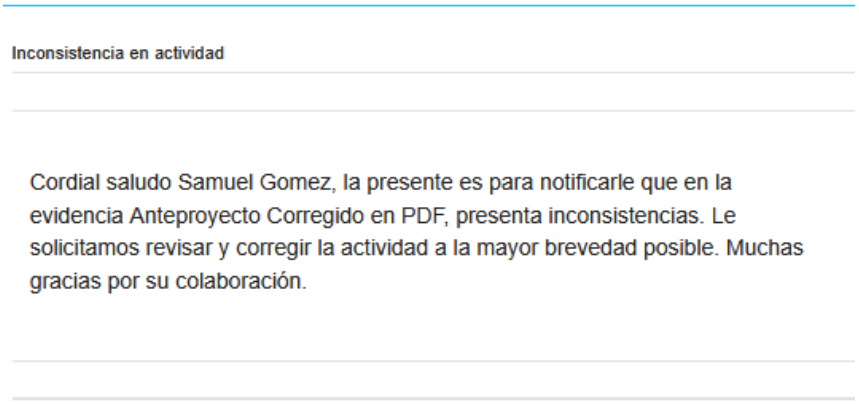


Figura 5.51: Correo electrónico de reclamación de actividad.
Fuente: Elaboración propia

Posterior a esta acción del coordinador y revisor académico, el estudiante se encontrará habilitado para cargar nuevamente la actividad mencionada en la reclamación.

5.5.4.4. Habilitación de control para inactivación de trabajos de grado

Se ha habilitado un nuevo control en la vista individual de cada trabajo de grado dentro del módulo de gestión, específicamente accesible para los usuarios con rol de gestor académico. A través de esta funcionalidad, el gestor podrá inactivar un trabajo de grado de forma manual, en los casos que así lo requiera la administración académica.

Para ello, se ha incorporado un botón de inactivación visible en la interfaz del detalle

de cada trabajo, situado como se muestra en la figura 5.52. Al hacer uso de esta opción, el estado del trabajo de grado cambia a inactivo, lo que impide que continúe su flujo normal dentro del sistema hasta que se reactive manualmente o se realice una acción administrativa posterior.

Con el fin de garantizar la trazabilidad y la transparencia del proceso, cada acción de inactivación será registrada en el sistema mediante un mecanismo de auditoría, que permitirá identificar qué usuario ejecutó el cambio, en qué fecha y bajo qué condiciones. Esta medida proporciona un respaldo institucional frente a decisiones críticas dentro del ciclo académico del trabajo de grado.

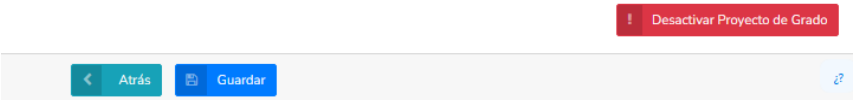


Figura 5.52: Botón de inactivación de trabajo de grado.
Fuente: Elaboración propia

5.5.4.5. Asignación de evaluadores a trabajos de grado

Se ha formalizado e implementado el **flujo de asignación de evaluadores** para trabajos de grado por parte del **coordinador académico**, con el objetivo de garantizar un proceso **estructurado**, **transparente** y **automatizado** en la etapa evaluativa.

El coordinador puede acceder a esta funcionalidad a través de la ruta: **Gerencia Académica** > **Trabajos de grado** > **Buscar** . Una vez localizado el trabajo correspondiente, deberá ingresar a la pestaña **Evaluación**, donde podrá gestionar la asignación de **docentes evaluadores**, además de revisar las evaluaciones ya realizadas por los docentes asignados a tal fin.

Para completar el proceso, el coordinador debe adjuntar al menos un **documento previamente cargado por el estudiante** —por ejemplo, la versión preliminar del trabajo de grado—. Posteriormente, podrá seleccionar uno o varios docentes desde el **desplegable disponible** en la interfaz tal como se muestra en la figura 5.53. El sistema se encarga de **excluir automáticamente al director y codirector** del trabajo de grado, evitando que sean asignados como evaluadores del mismo, en cumplimiento con los principios de **imparcialidad** y **conflicto de interés**.

Una vez realizada la asignación, el sistema envía de forma automática un **correo electrónico de notificación** a cada docente asignado. Este correo incluye los **documentos adjuntados** por el coordinador y las informaciones necesarias para que el evaluador inicie el proceso de revisión.

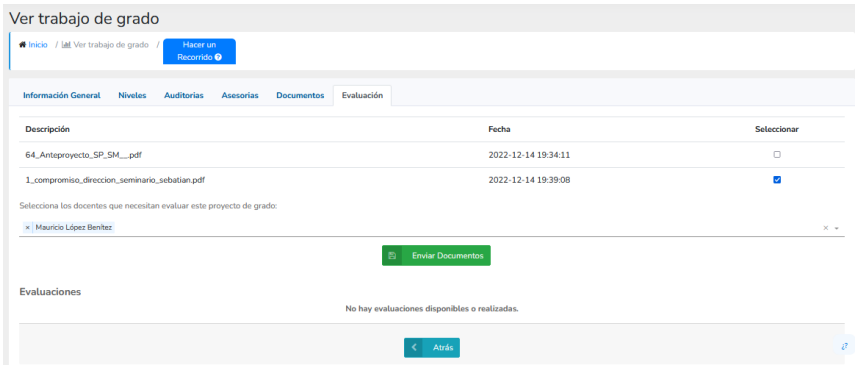


Figura 5.53: Pestaña de asignación de evaluadores.
Fuente: Elaboración propia

En caso de que el coordinador no seleccione ningún documento de los ya cargados, se mostrará el mensaje de información, mostrado en la figura, el cual impedirá que pueda enviar a evaluar el trabajo de grado sin antes seleccionar mínimo un documento. 5.54.

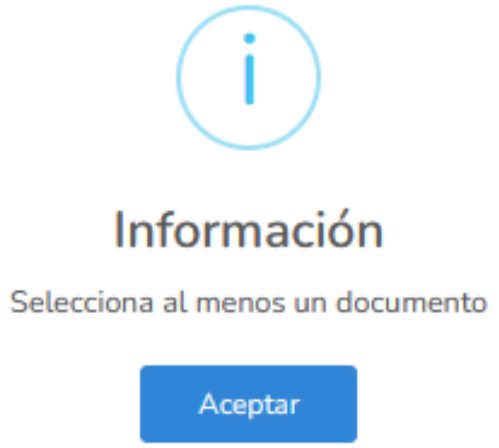


Figura 5.54: Modal de información mostrada al no seleccionar ningún documento.
Fuente: Elaboración propia

De igual manera, en caso de no seleccionar ningún docente de la lista de desplegables, el sistema mostrará una ventana modal informando que debe seleccionar mínimamente un docente para enviar a evaluar el trabajo de grado, como se muestra en la figura 5.55.

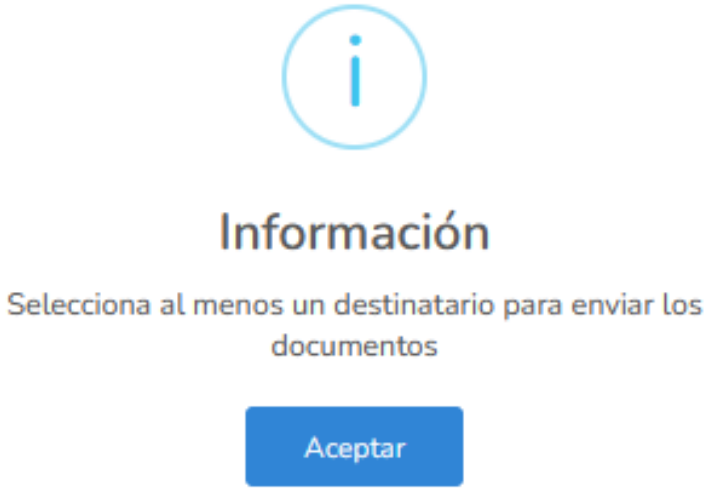


Figura 5.55: Modal de información mostrada al no seleccionar ningún docente.
Fuente: Elaboración propia

Al cumplir con el mínimo de un documento seleccionado y al menos un docente, se mostrará el mensaje modal de confirmación de la figura 5.56, listando los docentes seleccionados por nombre y apellido para que el usuario pueda verificar que está asignando a los evaluadores deseados.

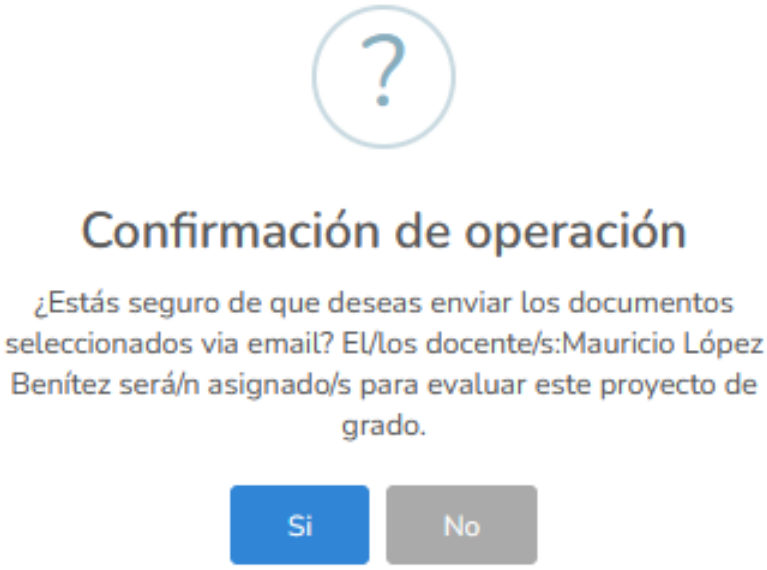


Figura 5.56: Modal de confirmación al enviar a evaluar.
Fuente: Elaboración propia

Al enviar a evaluar correctamente el trabajo de grado a el/los docentes, se mostrará el

mensaje de confirmación de la figura 5.57

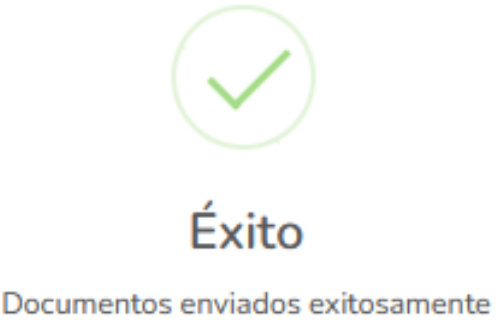


Figura 5.57: Mensaje de confirmación al enviar a evaluar satisfactoriamente.
Fuente: Elaboración propia

El correo que recibirá el/los docentes informándole que debe evaluar el trabajo de grado tendrá el formato de la figura 5.58, además de contar con los archivos adjuntos que se hayan seleccionado por el gestor.

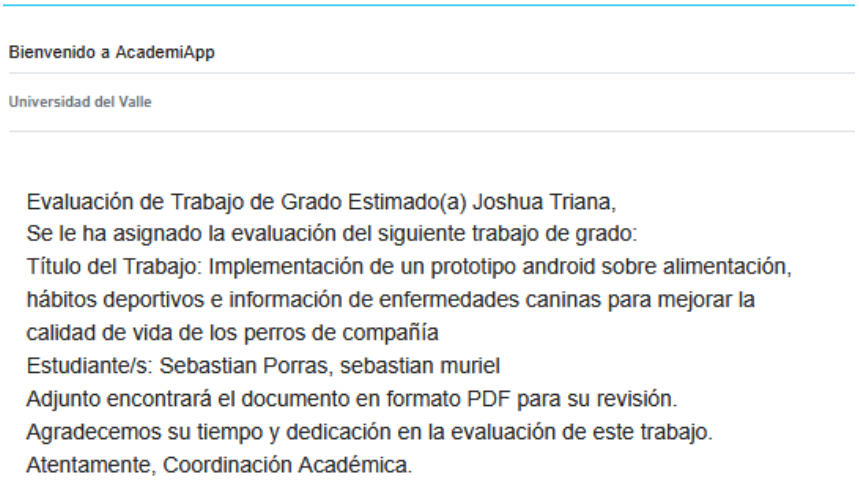


Figura 5.58: Correo informativo recibido por el/los docentes asignados.
Fuente: Elaboración propia

5.5.4.6. Revisión de evaluaciones

Dentro del **flujo de evaluación de trabajos de grado**, el **coordinador académico** cuenta con herramientas específicas para gestionar las evaluaciones recibidas por parte de los **docentes evaluadores**. Todo este proceso se realiza desde la misma pestaña **Evaluación**, ubicada dentro del detalle del trabajo de grado correspondiente.

Una vez los docentes asignados hayan completado sus evaluaciones, el sistema envía automáticamente una **notificación por correo electrónico** al coordinador, informándole

que el trabajo ya ha sido evaluado. Esta notificación permite al coordinador hacer seguimiento oportuno al avance del proceso y proceder a la siguiente etapa de revisión.

Desde la pestaña **Evaluación**, el coordinador podrá, tal como lo demuestra la figura 5.59:

- **Visualizar todas las evaluaciones recibidas**, junto con los documentos adjuntos y formularios diligenciados por cada evaluador.
- **Descargar un archivo Excel** con el contenido de la evaluación con el botón de **Descargar**, lo cual permite revisar y verificar la información antes de continuar con el proceso formal.
- **Retornar la evaluación al docente evaluador** con el botón de **Retornar**, en caso de encontrar observaciones o inconsistencias. Al ejecutar esta acción, el sistema enviará automáticamente un correo electrónico al docente, informándole que debe revisar y corregir su evaluación para posteriormente reenviarla.
- **Enviar la evaluación al estudiante o grupo de estudiantes** involucrados en el trabajo de grado con el botón de **Enviar**. Esta acción se realiza por correo electrónico, incluyendo como adjunto el archivo de evaluación completado. El sistema se encarga de enviar la notificación a todos los proyectistas asociados al trabajo, garantizando que cada estudiante reciba la retroalimentación correspondiente.

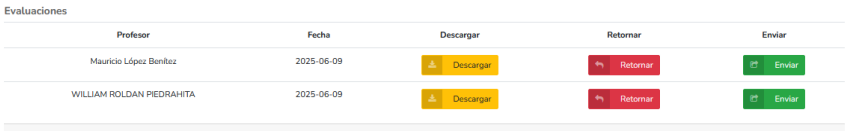


Figura 5.59: Visualización de evaluaciones de docentes.
Fuente: Elaboración propia

Al retornar una evaluación a un docente, y presionar el botón de **Retornar**, el sistema mostrará el mensaje de confirmación de la figura 5.60, el cual menciona el nombre del docente al cual será retornada la evaluación.

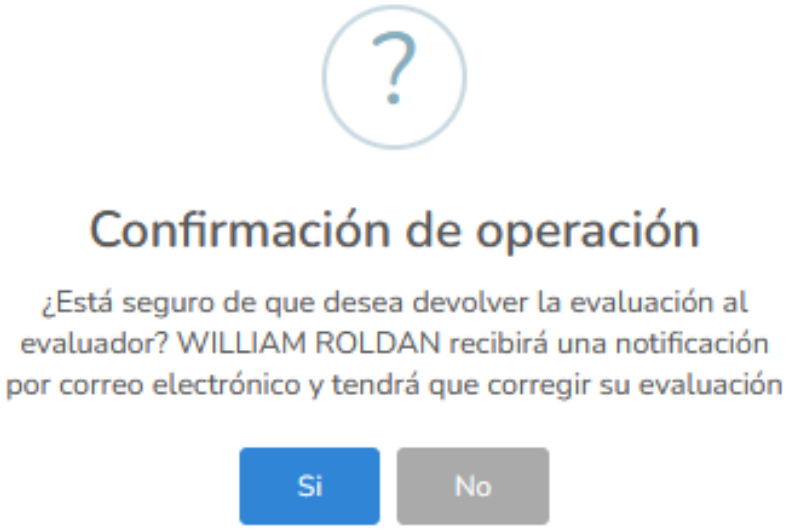


Figura 5.60: Mensaje de confirmación de retorno de evaluación.
Fuente: Elaboración propia

Al confirmar, el sistema mostrará el mensaje de confirmación de la figura 5.61, momento desde el cual, el docente podrá editar nuevamente su evaluación para posteriormente reenviarla al coordinador para una nueva revisión, y si es necesario, un nuevo retorno de la misma hasta cumplir con los requisitos establecidos.

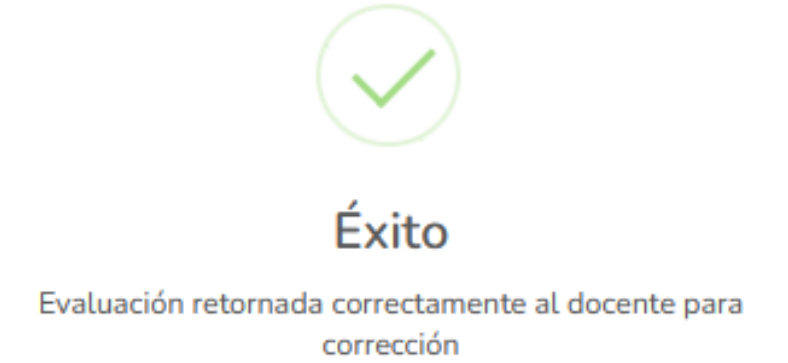


Figura 5.61: Mensaje de éxito de retorno de evaluación.
Fuente: Elaboración propia

El correo de reclamación que recibirá el docente tendrá el formato de la figura 5.62.

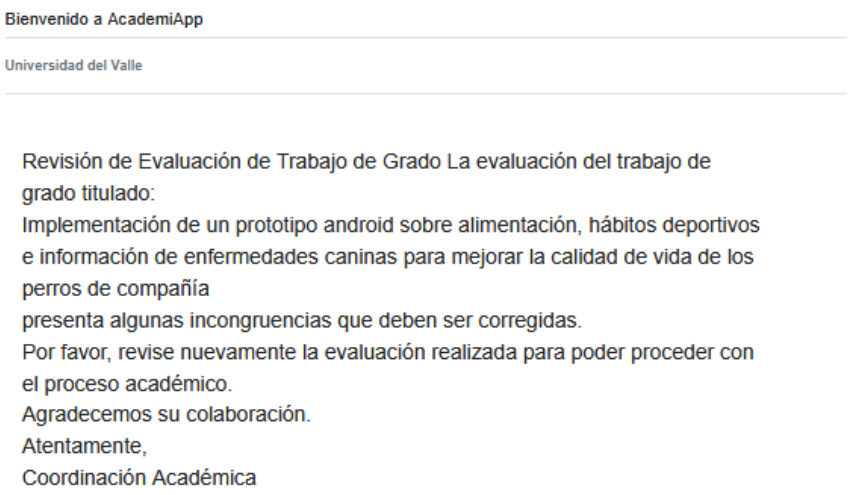


Figura 5.62: Correo recibido por el docente como reclamación de una evaluación.
Fuente: Elaboración propia

Al presionar el botón de **Enviar**, el cual enviará la evaluación de trabajo de grado a los estudiantes o proyectistas en formato Excel, se mostrará el mensaje de confirmación de la figura 5.63 antes el envío.

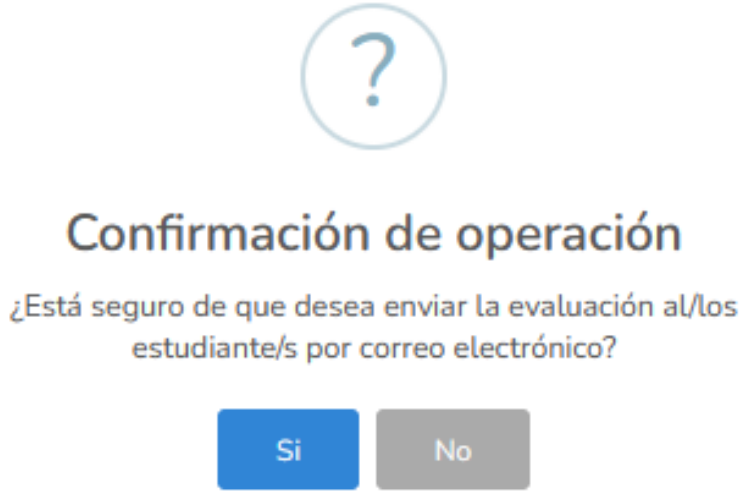


Figura 5.63: Mensaje de confirmación mostrado al presionar **Enviar**.
Fuente: Elaboración propia

El correo electrónico que recibirán los proyectistas del trabajo de grado corresponde al de la figura 5.64, el cual contendrá como adjunta la evaluación en formato Excel del docente en cuestión, ya que el envío de cada evaluador se realiza por individual.

Figura 5.64: Correo de confirmación de evaluación enviado a estudiantes/proyectistas.
Fuente: Elaboración propia

5.6. Módulo de Dificultades Académicas

Este módulo se encuentra integrado dentro del menú de **Asistencia Docente**, tal como se muestra en la figura 5.65, y está orientado a facilitar el seguimiento de los posibles obstáculos o condiciones adversas que puedan afectar el desarrollo normal de las sesiones académicas. Tradicionalmente, los docentes utilizaban esta sección únicamente al registrar la asistencia a las clases impartidas. Sin embargo, a partir de una mejora implementada, se ha incorporado un nuevo submódulo que permite a los docentes registrar de manera explícita cualquier dificultad académica que haya surgido en el contexto de una sesión.

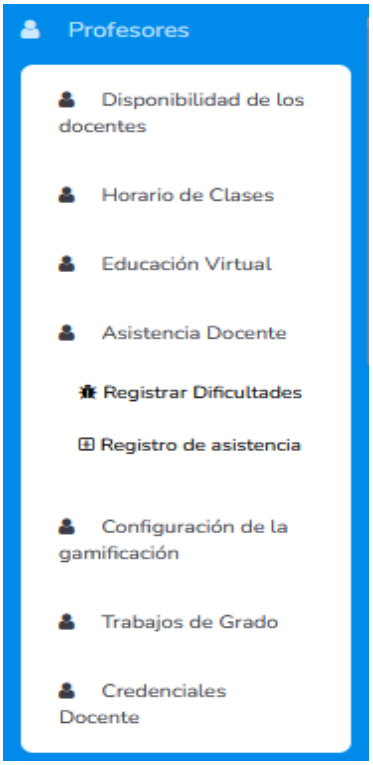


Figura 5.65: Menú de Asistencia Docente con acceso al submódulo de Dificultades Académicas.

Fuente: Elaboración propia

Como parte del rediseño funcional, se modificó completamente el proceso de registro de dificultades académicas, que anteriormente estaba acoplado al formulario de creación de sesiones de asistencia. En dicho formulario, el docente debía responder a una pregunta tipo sí/no sobre si experimentó alguna dificultad académica durante la clase. No obstante, se detectaron casos frecuentes en los que los docentes seleccionaban que sí existió una dificultad, pero ingresaban descripciones del tipo *ninguna* o similares, lo cual dificultaba el tratamiento adecuado de la información y generaba múltiples falsos positivos.

Con el fin de mitigar este tipo de inconsistencias, se decidió separar esta funcionalidad y trasladarla a un submódulo independiente. Ahora, el registro de dificultades se realiza de manera explícita, con campos obligatorios y controles más estrictos para asegurar que la información consignada sea clara, válida y útil para los procesos de seguimiento académico.

La figura 5.66 muestra el diseño general del submódulo, el cual será detallado a continuación.

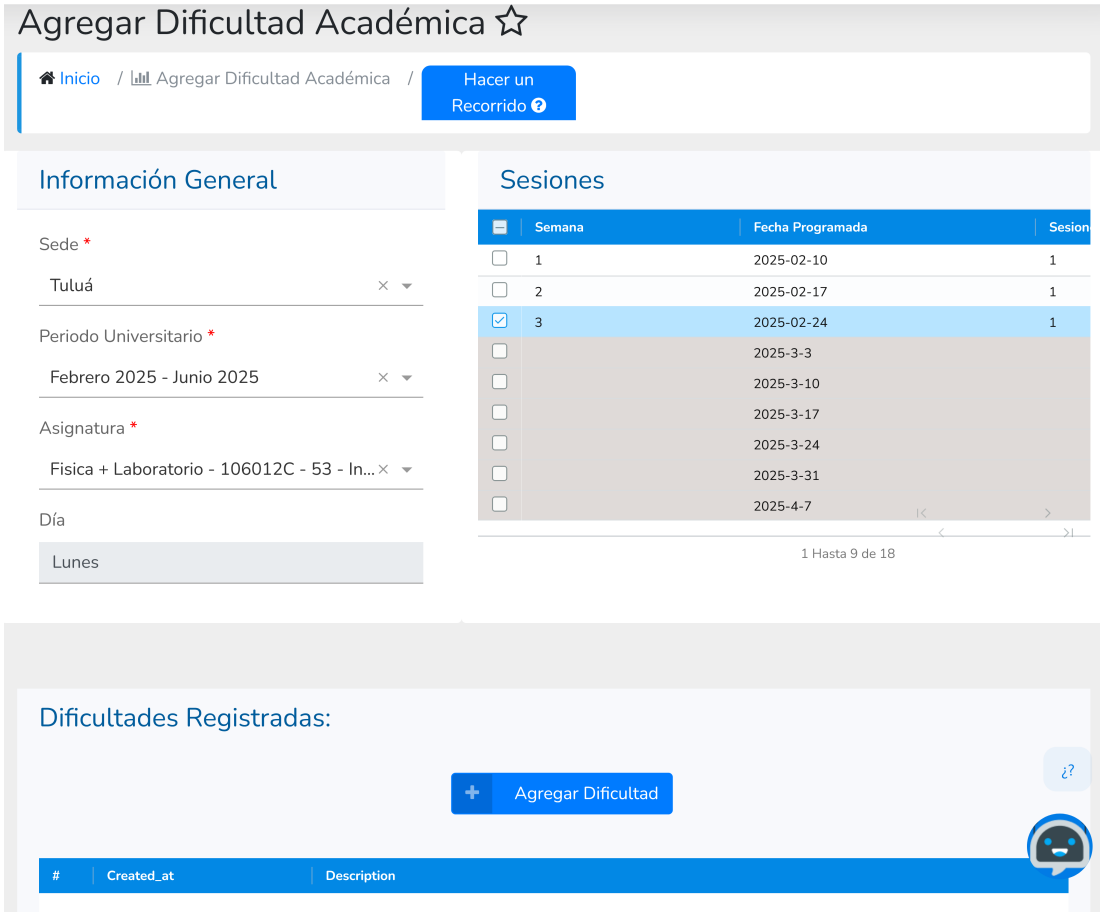


Figura 5.66: Submódulo de Dificultades Académicas.
Fuente: Elaboración propia

Para registrar una nueva dificultad académica, el docente debe completar un formulario inicial en el que se seleccionan los filtros de sede, periodo académico y asignatura. Una vez aplicados estos filtros, se llenará la tabla con las sesiones previamente registradas para dicha combinación. Al seleccionar una sesión específica desde la tabla, se habilita el botón *Agregar Dificultad*, que al ser presionado abre un modal con el formulario de registro de la figura 5.67.

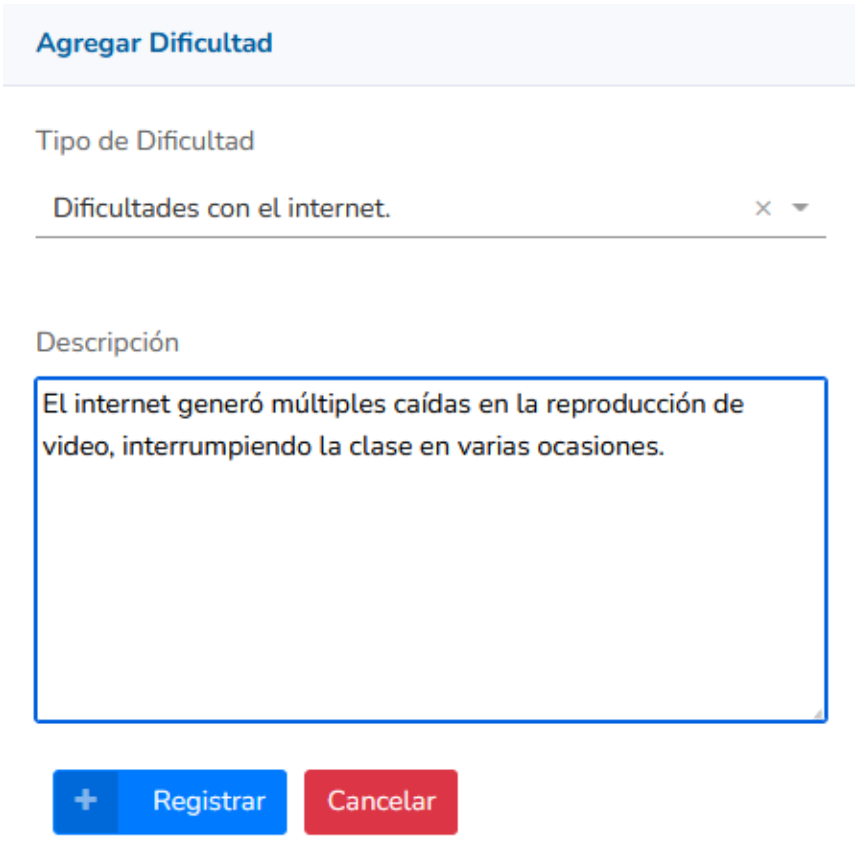


Figura 5.67: Modal de creación de una dificultad académica asociada a una sesión.
Fuente: Elaboración propia

En este modal, el docente deberá especificar el tipo de dificultad encontrada a partir de un conjunto predefinido de tipos desplegados, y consignar una observación o descripción detallada que permita contextualizar adecuadamente la situación. Esta información será utilizada para el análisis posterior de las condiciones que afectan la calidad del proceso formativo, proceso que ya se encuentra con su propio flujo implementado, el cual no fue modificado.

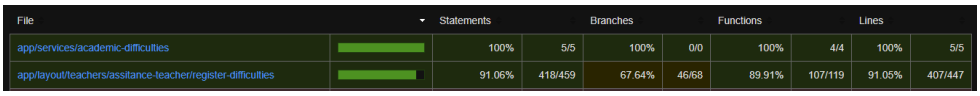
Posteriormente al llenado de este modal de creación de dificultad, el docente recibirá el mensaje de confirmación de la figura 5.68, informándole que su dificultad fue creada con éxito.

Figura 5.68: Modal de confirmación de creación de una dificultad académica.
Fuente: Elaboración propia

5.6.1. Pruebas Unitarias del Módulo de Dificultades Académicas

Como parte del proceso de aseguramiento de calidad, se implementaron pruebas unitarias tanto para el nuevo componente de dificultades académicas como para el servicio responsable de gestionar las peticiones asociadas a este flujo, los cuales no contaban con ningún tipo de cobertura previa. Esto permitió garantizar el correcto funcionamiento de todas las funcionalidades implementadas, así como mejorar la mantenibilidad del código a futuro.

En las figuras 5.69 y 5.72 se evidencian los reportes de cobertura generados en la ejecución de las pruebas implementadas, así como el cumplimiento del criterio de aceptación general.



File	Statements		Branches		Functions		Lines	
app/services/academic-difficulties	100%	5/5	100%	0/0	100%	4/4	100%	5/5
app/layout/teachers/assitance-teacher/register-difficulties	91.06%	418/459	67.64%	46/68	89.91%	107/119	91.05%	407/447

Figura 5.69: Reporte de cobertura de pruebas unitarias del componente y servicio de dificultades académicas.
Fuente: Elaboración propia

5.6.1.1. Servicio de comunicación con el Back-End

A continuación, se detallan los principales casos de prueba implementados en este servicio, los cuales se centran en el manejo de respuestas correctas, mensajes y códigos de error en las peticiones:

- La correcta inicialización del servicio.
- La respuesta esperada de cada llamado a la API que espera el Back-End.
- La validación de los mensajes, códigos de error y cantidad de llamadas recibidas desde la API.

- El comportamiento de la llamada cuando se realiza una consulta o petición vacía.

En la figura 5.70 se muestra un fragmento del código de las pruebas del servicio de forma minimizada para evidenciar un poco más de lo implementado.

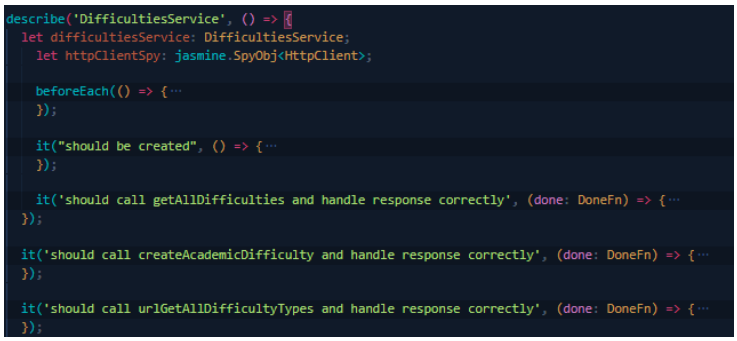


Figura 5.70: Fragmento de código de pruebas del servicio de dificultades académicas.
Fuente: Elaboración propia

5.6.1.2. Módulo de registro de dificultades académicas

A continuación, se detallan los principales casos de prueba implementados en este componente, los cuales a su vez, se subdividieron en muchos casos en más pruebas para cada caso de uso por el usuario:

- La correcta inicialización del componente, formularios principales y de los datos de las dificultades académicas que ya hayan sido creadas por cada sesión, asegurando su estructura y valores por defecto antes de la consulta.
- El funcionamiento esperado al usuario generar una nueva dificultad académica en una sesión específica.
- La correcta consistencia de las dificultades registradas para una sesión y el total de dificultades registradas por período.
- El completo funcionamiento del modal de creación de dificultad el cual se despliega con múltiples tipos de dificultades.

En la figura 5.71 se muestra un fragmento del código de las pruebas del componente.

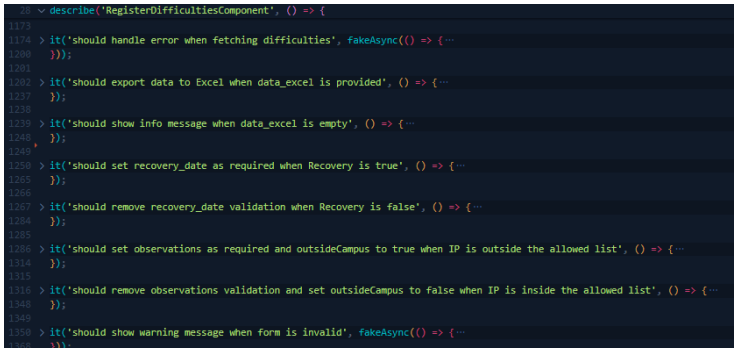


Figura 5.71: Fragmento de código de pruebas del componente de registro de dificultades académicas.

Fuente: Elaboración propia

Adicionalmente, se desarrollaron pruebas unitarias para el servicio encargado de la generación de archivos Excel dentro del flujo de trabajo de dificultades académicas tal como se muestra en la figura 5.72, el cual tampoco disponía de cobertura alguna antes de esta mejora. Gracias a estas pruebas, se validaron correctamente los diferentes escenarios de exportación de datos, asegurando un comportamiento robusto y fiable ante diferentes condiciones de uso.

Figura 5.72: Reporte de cobertura de pruebas unitarias del servicio de generación de Excel para dificultades académicas.

Fuente: Elaboración propia

En total el número de pruebas en correcta ejecución e implementación para el flujo completo de dificultades académicas fue de **80**, a continuación en la figura 5.73 se muestran las pruebas ejecutadas y aprobadas.

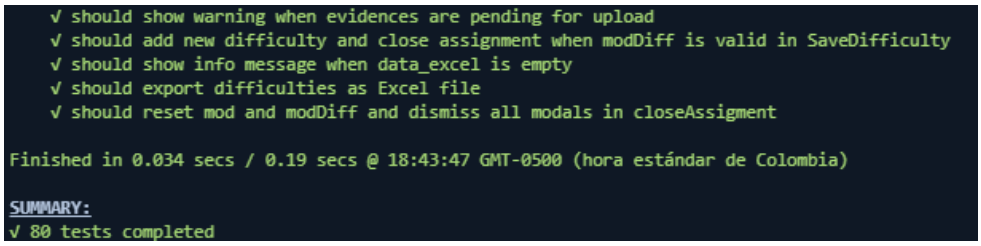


Figura 5.73: Resumen de ejecución de pruebas del flujo de dificultades académicas.

Fuente: Elaboración propia

5.7. Módulo de Reportes

El módulo de reportes, ubicado dentro de la sección de Gerencia Académica, agrupa diversos submódulos que permiten consultar y exportar información relevante relacionada con las actividades académicas realizadas en la plataforma. Como se muestra en la figura

5.74, entre sus funcionalidades actuales se encuentran: el *Reporte de Evidencias de Profesores*, el *Reporte de Asesorías de Trabajos de Grado* y el *Reporte de Asistencia*. Como mejora reciente, se habilitó un nuevo submódulo denominado *Cargar Registros de Huella*, el cual fue desarrollado con el propósito de que el personal encargado pueda cargar los registros de huella dactilar capturados desde los dispositivos de marcación de asistencia en sede.

Figura 5.74: Menú de módulo de reportes.
Fuente: Elaboración propia

5.7.1. Pruebas Unitarias de submódulo de Cargar de Registros de Huella

En este nuevo submódulo habilitado, se desarrollaron e implementaron sus respectivas pruebas unitarias, tanto para el componente que gestiona la interfaz de usuario como para el servicio encargado de manejar la lógica y comunicación con el servidor, tal como lo muestran los siguientes informes de cobertura. Esto permitió garantizar la estabilidad del flujo de trabajo y validar el correcto funcionamiento de las funcionalidades de carga, validación y registro de datos, asegurando una mayor confiabilidad y mantenibilidad del sistema a futuro.

En la figura 5.75 se evidencian los reportes de cobertura generados en la ejecución de las pruebas implementadas, así como el cumplimiento del criterio de aceptación general.

File	Statements		Branches		Functions		Lines	
app/layout/teachers/load-fingerprint-regs	81.52%	75/92	68.42%	13/19	84.21%	16/19	80.68%	71/88
app/services/fingerprints	100%	4/4	100%	0/0	100%	2/2	100%	4/4

Figura 5.75: Reporte de cobertura de pruebas unitarias del componente y servicio de carga de registros de huella.

Fuente: Elaboración propia

5.7.1.1. Servicio de comunicación con el Back-End

Las pruebas unitarias desarrolladas para este servicio, al igual que en los demás, validan principalmente los siguientes aspectos:

- La correcta inicialización del servicio.
- La respuesta esperada de cada llamado a la API que espera el Back-End.
- La validación de los mensajes, códigos de error y cantidad de llamadas recibidas desde la API.
- El comportamiento de la llamada cuando se realiza una consulta o petición vacía.

En la figura 5.76 se muestra un fragmento del código de las pruebas minimizadas del servicio con el fin de abarcar una mayor cantidad en la evidencia.

```
describe('FingerprintsService', () => {
  let httpClientSpy: jasmine.SpyObj<HttpClient>;
  let fingerprintService: FingerprintsService;

  beforeEach(() => {···
  });

  it('should be created', () => {···
  });

  it('should call saveRecords with correct parameters', () => {
    const mockResponse = { id: 1, name: "Fingerprint 1" };
    const mockData = { data: "sampleData" };
    const expectedUrl = AppConfig.postFingerprintRecords; // Asegurar que usamos la URL correcta

    httpClientSpy.post.and.returnValue(of(mockResponse)); // Retorna un Observable
      Nacho Gómez, 5 months ago • Agregar pruebas unitarias para el servicio Fing…
    fingerprintService.saveRecords(mockData).subscribe((result) => {
      expect(result).toEqual(mockResponse);
    });

    expect(httpClientSpy.post).toHaveBeenCalledWith(expectedUrl, mockData);
  });
});
```

Figura 5.76: Fragmento de código de pruebas del servicio de reporte de asistencia con huellas.

Fuente: Elaboración propia

5.7.1.2. Módulo de subida de registros de huellas

A continuación, se detallan los principales casos de prueba implementados en este componente, los cuales a su vez, se subdividieron en muchos casos en más pruebas para cada caso de uso por el usuario:

- La correcta inicialización del componente y del formulario con el archivo a subir asegurando su valor por defecto.
- La retroalimentación al usuario al tener problemas con el archivo que seleccionó.
- La correcta visualización de los mensajes mostrados al usuario sobre la cantidad de registros preprocesados y correctamente subidos a la base de datos.

Estas pruebas permitieron garantizar que el módulo responda adecuadamente ante entradas válidas e inválidas, errores del servidor, y múltiples interacciones del usuario. En la figura 5.77 se muestra un fragmento del código de las pruebas del componente.

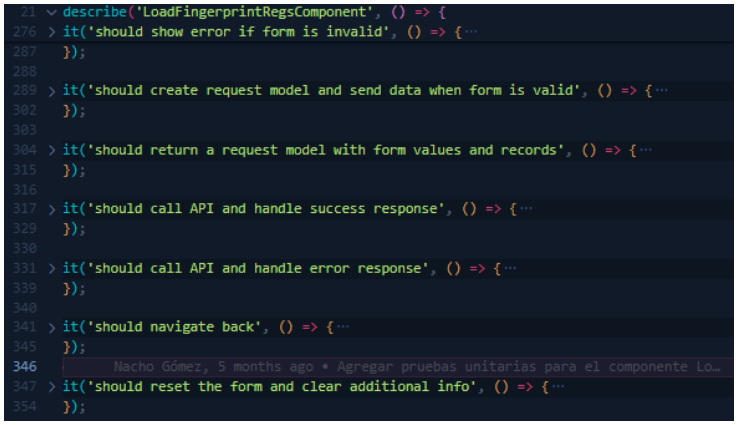


Figura 5.77: Fragmento de código de pruebas del componente de subida de registros de huella.

Fuente: Elaboración propia

En total el número de pruebas en correcta ejecución e implementación para el flujo completo de carga de registros de huella fue de **22**, a continuación en la figura 5.78 se muestran las pruebas ejecutadas y aprobadas.

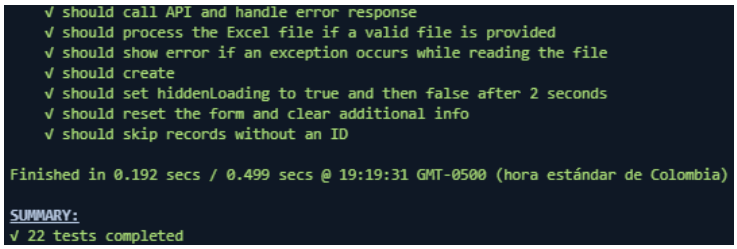


Figura 5.78: Resumen de ejecución de pruebas de subida de registros de huella.
Fuente: Elaboración propia

5.7.2. Submódulo para Cargar Registros de Huella

Este submódulo fue diseñado para permitir al encargado académico cargar registros de asistencia marcados con huella dactilar por parte de los docentes. Para ello, es necesario

contar con un archivo en formato Excel que contenga los datos estructurados conforme a los lineamientos establecidos por la plataforma. Una vez cargado correctamente el archivo, los registros se almacenan en la base de datos, permitiendo que esta información sea posteriormente consultada y cruzada desde el submódulo de *Reporte de Asistencia*, con el fin de validar la presencialidad del docente al momento de impartir sus clases. Esta validación es clave para sustentar informes relacionados con viáticos u otros procesos administrativos asociados a la actividad docente.

En la figura 5.79 se evidencia la apariencia general del submódulo.

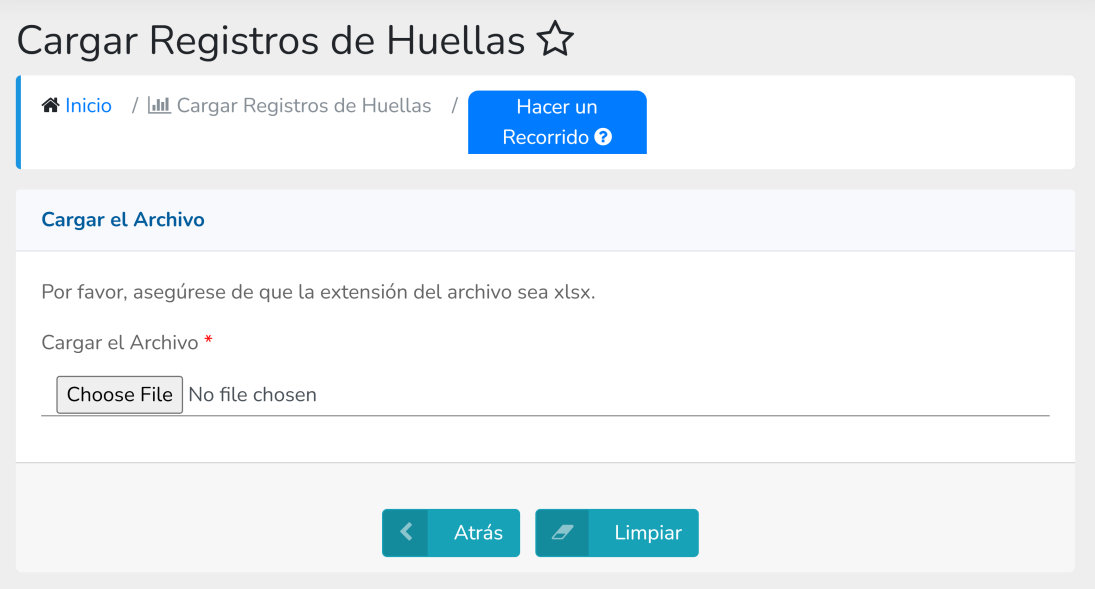
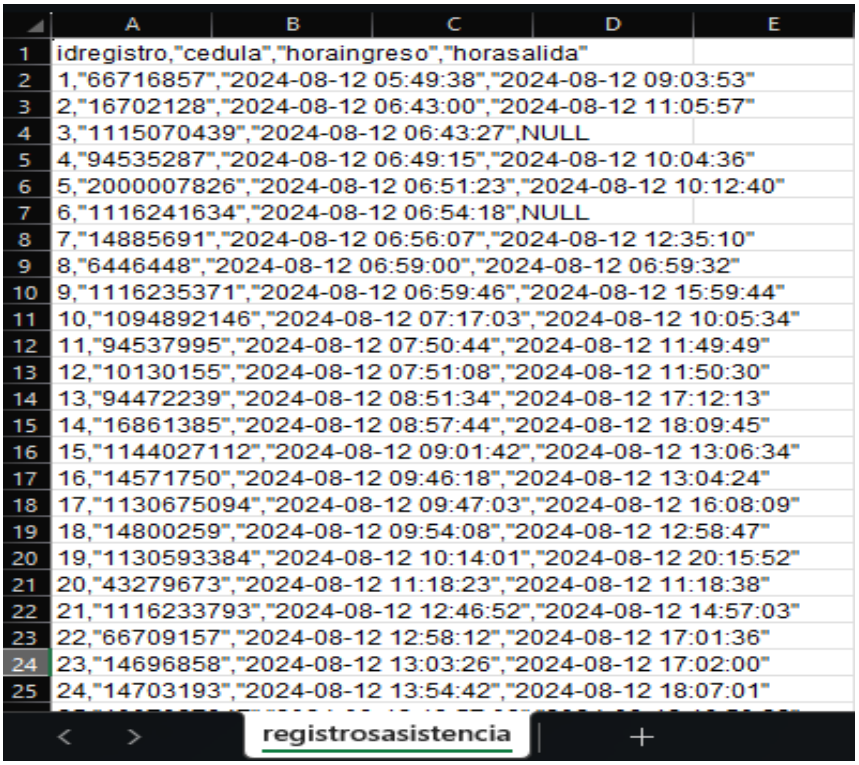


Figura 5.79: Vista del submódulo de Cargar Registros de Huella.
Fuente: Elaboración propia

El formato del archivo Excel que debe ser utilizado para realizar la carga de registros fue definido a partir de un caso real proporcionado por la institución, por lo tanto, el sistema fue desarrollado específicamente para reconocer y procesar dicha estructura, tal como se muestra en la figura 5.80. Es importante que el archivo mantenga ese formato sin alteraciones y se encuentre en extensión `.xlsx` para garantizar una correcta lectura y almacenamiento de los datos. Cualquier variación en la estructura del archivo puede ocasionar errores en la carga o rechazo del mismo por parte del sistema.



	A	B	C	D	E
1	idregistro,"cedula","horaingreso","horasalida"				
2	1,"66716857","2024-08-12 05:49:38","2024-08-12 09:03:53"				
3	2,"16702128","2024-08-12 06:43:00","2024-08-12 11:05:57"				
4	3,"1115070439","2024-08-12 06:43:27",NULL				
5	4,"94535287","2024-08-12 06:49:15","2024-08-12 10:04:36"				
6	5,"2000007826","2024-08-12 06:51:23","2024-08-12 10:12:40"				
7	6,"1116241634","2024-08-12 06:54:18",NULL				
8	7,"14885691","2024-08-12 06:56:07","2024-08-12 12:35:10"				
9	8,"6446448","2024-08-12 06:59:00","2024-08-12 06:59:32"				
10	9,"1116235371","2024-08-12 06:59:46","2024-08-12 15:59:44"				
11	10,"1094892146","2024-08-12 07:17:03","2024-08-12 10:05:34"				
12	11,"94537995","2024-08-12 07:50:44","2024-08-12 11:49:49"				
13	12,"10130155","2024-08-12 07:51:08","2024-08-12 11:50:30"				
14	13,"94472239","2024-08-12 08:51:34","2024-08-12 17:12:13"				
15	14,"16861385","2024-08-12 08:57:44","2024-08-12 18:09:45"				
16	15,"1144027112","2024-08-12 09:01:42","2024-08-12 13:06:34"				
17	16,"14571750","2024-08-12 09:46:18","2024-08-12 13:04:24"				
18	17,"1130675094","2024-08-12 09:47:03","2024-08-12 16:08:09"				
19	18,"14800259","2024-08-12 09:54:08","2024-08-12 12:58:47"				
20	19,"1130593384","2024-08-12 10:14:01","2024-08-12 20:15:52"				
21	20,"43279673","2024-08-12 11:18:23","2024-08-12 11:18:38"				
22	21,"1116233793","2024-08-12 12:46:52","2024-08-12 14:57:03"				
23	22,"66709157","2024-08-12 12:58:12","2024-08-12 17:01:36"				
24	23,"14696858","2024-08-12 13:03:26","2024-08-12 17:02:00"				
25	24,"14703193","2024-08-12 13:54:42","2024-08-12 18:07:01"				

registrosasistencia

Figura 5.80: Formato de Excel proporcionado por la institución.
Fuente: Elaboración propia

Al presionar el botón *Cargar archivo* y seleccionar un archivo en formato `.xlsx`, el sistema ejecuta un proceso de prevalidación del contenido del archivo. Como resultado, se despliega un mensaje informativo que indica la cantidad total de registros leídos correctamente y cuántos de ellos se identificaron como duplicados dentro del mismo archivo, como se muestra en la figura 5.81. Esta verificación previa permite al encargado académico tener una visión general de los datos antes de realizar la carga definitiva, reduciendo así posibles errores o registros innecesarios.

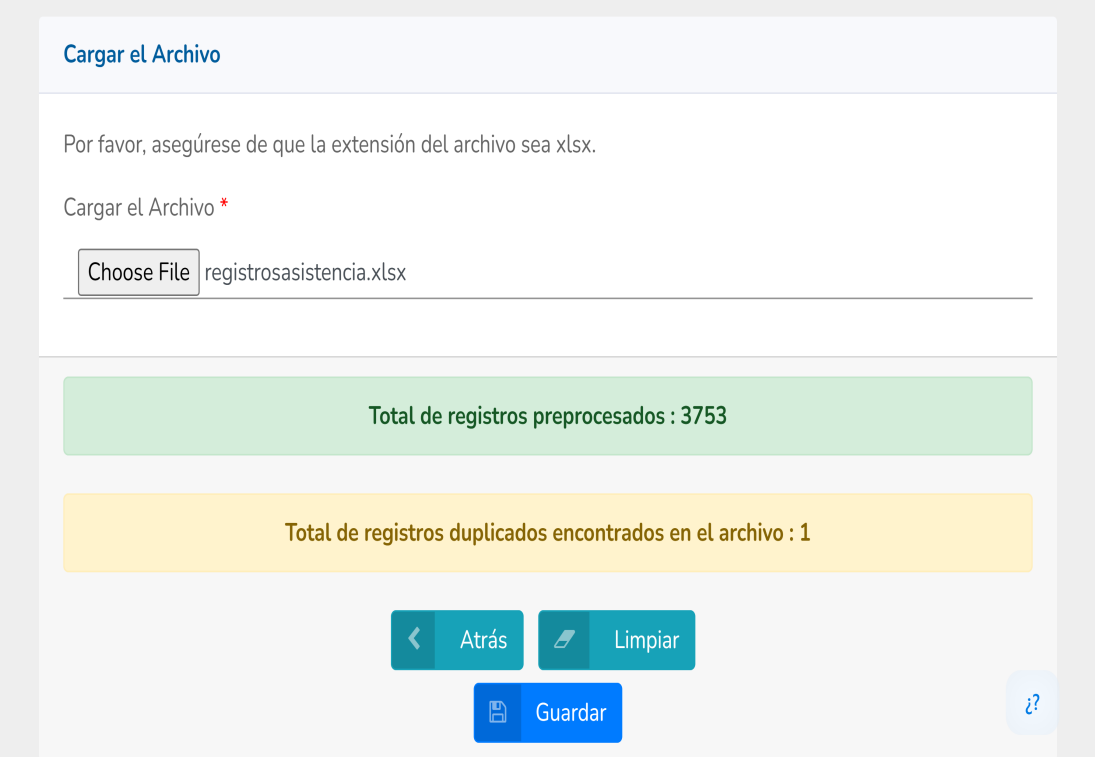


Figura 5.81: Mensaje informativo con registros preprocesados y repetidos.
Fuente: Elaboración propia

Una vez cargado el archivo, al presionar el botón *Guardar*, se ejecuta una validación contra la base de datos para comprobar que cada número de cédula asociado a los registros corresponda a una persona existente en el sistema. Este proceso se realiza sobre todos los registros preprocesados, omitiendo los que fueron detectados como duplicados, ya que estos solo se envían una vez para evitar redundancias. Al finalizar el proceso de inserción, el módulo muestra el mensaje informativo de la figura **??** que indica la cantidad total de registros de huella que fueron exitosamente guardados en la base de datos.

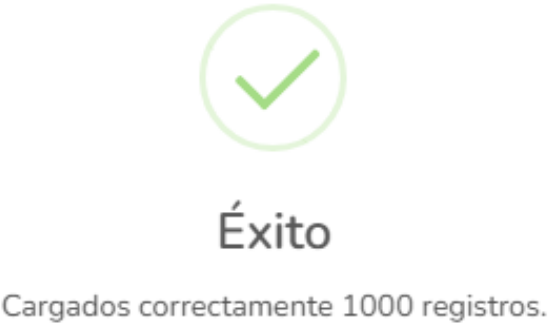


Figura 5.82: Modal informativo con cantidad de registros guardados satisfactoriamente.
Fuente: Elaboración propia

Cuando el archivo seleccionado por el usuario, no es un archivo que corresponda con la extensión admitida, se muestra el mensaje de error de la figura 5.83 indicándole al usuario de revisar la extensión del mismo.

Figura 5.83: Modal de error sobre archivo seleccionado.
Fuente: Elaboración propia

5.7.3. Submódulo de Reporte de Asistencia

En este submódulo se implementó una mejora funcional que permite visualizar con mayor claridad la información relacionada con las asistencias de los docentes. Se añadió una nueva columna en la tabla de resultados, visible una vez se seleccionan los filtros de sede, periodo académico y demás parámetros correspondientes. Esta columna, titulada *Registro de Huella*, indica si el docente registró su asistencia mediante huella biométrica en la fecha correspondiente a cada sesión, tal como se evidencia en la figura 5.84.

Es importante destacar que, si un docente marca su huella en un determinado día, esta marcación aplica automáticamente para todas las sesiones registradas ese mismo día. Esta lógica se implementó teniendo en cuenta que, para efectos del reporte de viáticos, solo se considera un único desplazamiento diario del docente desde su ciudad de origen hasta la sede donde dicta clase. Por lo tanto, una sola marcación de huella valida la presencia del docente en todas sus sesiones académicas del día, optimizando así el proceso de verificación para la asignación de viáticos.

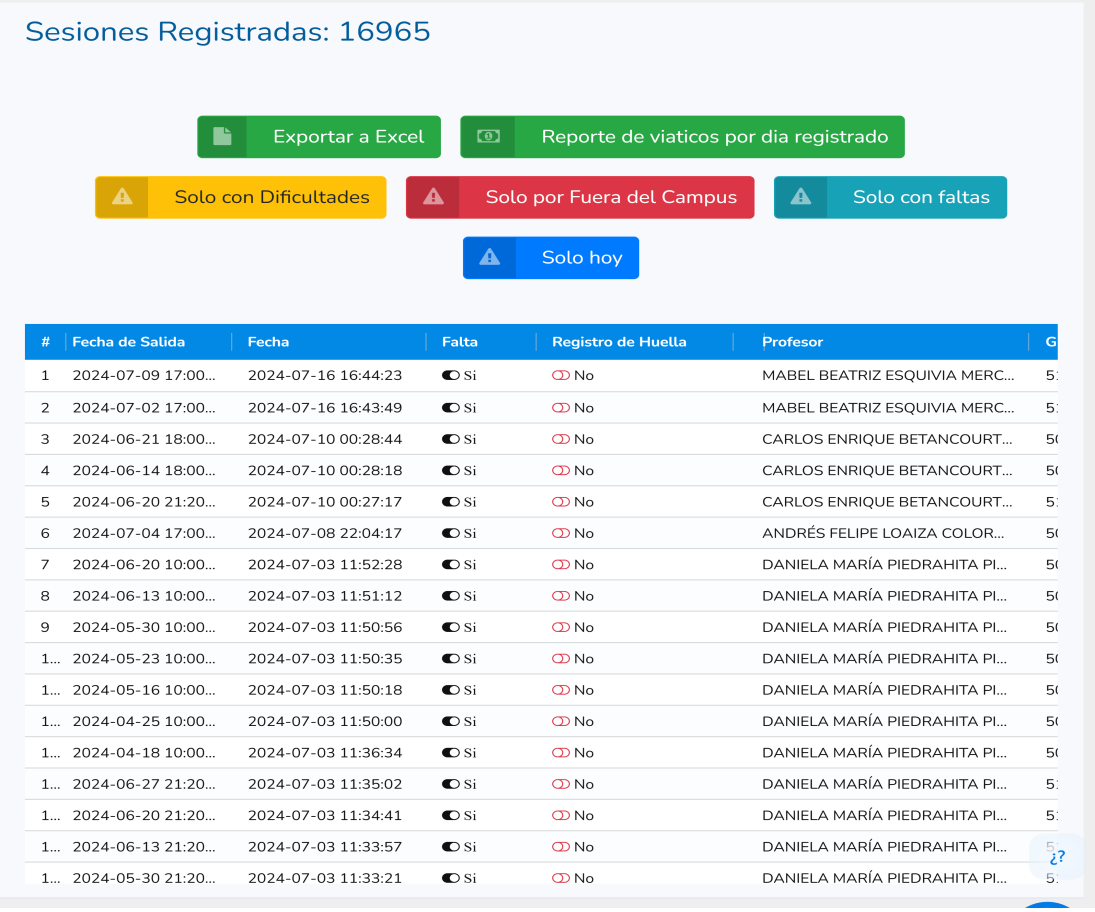


Figura 5.84: Tabla con la nueva columna de *Registro de Huella*.
Fuente: Elaboración propia

A partir de la tabla obtenida mediante los filtros de consulta, este submódulo permite además la generación de un archivo en formato Excel, el cual funciona como reporte de viáticos correspondientes a cada docente, desglosado por cada día en el que dictó clases. En este archivo generado, se incluye ahora una nueva columna denominada *Registro de Huella*, la cual indica si el docente registró o no su asistencia mediante huella biométrica en la fecha de la sesión, como se muestra en la figura 5.85. Esta verificación se realiza cruzando la información de la fecha de la sesión y la cédula del docente con los registros de huella previamente cargados al sistema a través del submódulo *Cargar registros de huella*. Esto permite a la gerencia académica validar de manera más precisa la presencialidad del docente, facilitando así la toma de decisiones en el proceso de asignación de viáticos.

Reporte de viaticos por dia registrado

Sede : Tuluá

Periodo : Febrero 2024 - Junio 2024

Fecha de Registro	N° Documento	Profesor	Correo Electrónico	City	Precio	Registro de Huella
2024-06-29 15:13:03	14898902	JUAN CARLOS RICARDO LADINO	icardito@pruebasgmail.cc	Cali	34000	No
2024-06-26 20:02:08	1017189753	ANDREA PATRICIA GALLO RAMIREZ	llo@pruebascorreounival	Guadalajar	12000	No
2024-06-26 17:44:19	1144053208	SEBASTIAN null CANO PANTOJA	cano@pruebascorreouniv	Cali	34000	No
2024-06-26 14:53:42	14571750	Edison Andrés Carmona Riveros	DNA@pruebasCORREOUN	Cartago	0	No
2024-06-26 14:50:40	14696858	JULIÁN ANDRÉS VILLA CUENCA	s.villa@pruebascorreouni	Palmira	26000	No
2024-06-26 11:22:43	1113669741	Angélica María Panesso Libreros	nesso@pruebascorreouni	Palmira	26000	No
2024-06-26 11:09:21	31792419	Johanna Andrea Ávila Canal	vila@pruebascorreouniva	Cali	34000	No
2024-06-26 10:54:10	1144094073	LUIS FELIPE AZCARATE VILLA	uis@pruebascorreounival	Cali	34000	No
2024-06-26 09:07:28	16930113	HAROLD EDMUNDO MORA Campo	ora@pruebascorreounival	Guadalajar	12000	No

Figura 5.85: Excel generado con nueva columna de Registro de Huella.
Fuente: Elaboración propia

5.8. Entorno y Accesos

El repositorio de Front-End correspondiente se encuentra en [26], donde se pueden verificar en orden y de manera etiquetada, todos y cada uno de los Pull Requests realizados durante el desarrollo de este trabajo, tal como se muestra en la figura 5.3. Adicionalmente, el repositorio del Back-End se encuentra en [27]. Se puede acceder al entorno de pruebas desde [28] con las siguientes credenciales:

- **Usuario:** `UV_admin`
- **Contraseña:** `123456`

Este usuario posee permisos ampliados que permiten interactuar con todos los submódulos relacionados con el sistema de almuerzos, generación de reportes, trabajos de grado, incluyendo los roles de estudiante, proveedor y gerencia académica. Esto facilita la validación integral de las funcionalidades desde distintos perfiles de usuario.

> ***Nota**: Las credenciales indicadas son exclusivamente para uso en entornos de prueba. No deben ser utilizadas en ambientes de producción.*

> ***Importante**: El repositorio del proyecto se encuentra en modalidad privada. Para obtener acceso, es necesario solicitar autorización y permisos de acceso al director del proyecto por medio de correo electrónico royer.estrada@correounivalle.edu.co.*

Capítulo 6

Conclusiones y Proyecciones

6.1. Conclusiones

La parte principal de este aprendizaje fue la incorporación de pruebas unitarias con Jasmine y Karma, herramientas fundamentales en el ecosistema Angular. Se trabajó activamente en la creación de casos de prueba para componentes y servicios que anteriormente no contaban con ningún tipo de validación. Esto aseguró un comportamiento consistente del sistema ante diversos escenarios, minimizando errores y facilitando futuras ampliaciones. La práctica constante de testeo elevó el nivel de profesionalismo técnico, fomentando una cultura de desarrollo más robusta y confiable. Más allá de la parte técnica, el proyecto también incentivó el fortalecimiento de habilidades como la comunicación efectiva, la toma de decisiones bajo presión y la colaboración activa en equipos multidisciplinarios. Estas capacidades complementan el perfil profesional, aportando una mayor preparación para enfrentar desafíos reales en proyectos tecnológicos a gran escala.

Adicionalmente, el desarrollo de los distintos módulos y las funcionalidades implementadas por requerimiento de la empresa, permitió aplicar conocimientos adquiridos previamente en un entorno más cercano al flujo de trabajo profesional. Durante el proceso, se emplearon herramientas clave como Angular, Git y Notion para planificar, desarrollar, documentar y probar funcionalidades reales dentro de una plataforma institucional, logrando no solo cumplir con requerimientos funcionales, sino también mantener estándares de calidad y escalabilidad en el código.

El impacto del trabajo realizado se traduce en una mejora concreta de los procesos académicos internos, reduciendo tareas manuales y ofreciendo nuevas funcionalidades que permiten mayor trazabilidad, control y eficiencia, además de asegurar su solidez mediante la implementación de casos de prueba minimizando su posibilidad de fallar en entornos de uso real y facilitando el desarrollo e implementación de nuevas funcionalidades o una reestructuración dado el caso en que sea necesaria. Cada módulo intervenido y probado no solo soluciona una necesidad puntual, sino que también se integra como parte de una estrategia más amplia para transformar digitalmente los procedimientos de la institución, con una visión enfocada en la sostenibilidad, usabilidad y mejora continua del sistema.

6.2. Proyecciones

Con el objetivo de continuar fortaleciendo el sistema y anticiparse a nuevas necesidades académicas y administrativas, se contemplan diversas proyecciones de mejora que podrían implementarse en futuras fases del proyecto. Estas iniciativas buscan optimizar la trazabilidad, garantizar mayor control institucional, y ofrecer una experiencia más completa y robusta a todos los actores involucrados en el proceso de gestión de trabajos de grado.

En primer lugar, se propone ampliar las validaciones al momento de asignar evaluadores, integrando mecanismos automáticos que detecten posibles conflictos de interés, más allá del director y codirector, como antecedentes compartidos entre docentes y estudiantes o coincidencias en semilleros de investigación. También sería conveniente establecer un límite al número de trabajos que pueden ser evaluados simultáneamente por un mismo docente, con el fin de distribuir la carga académica de forma equitativa y asegurar evaluaciones oportunas.

Respecto a la creación de trabajos de grado, se podrían incorporar validaciones más específicas en el campo del título, incluyendo restricciones de longitud, detección de términos repetitivos y sugerencias automáticas de palabras clave o áreas temáticas. Esto no solo facilitaría la clasificación académica, sino que también contribuiría a análisis posteriores sobre tendencias temáticas dentro de los programas.

En el módulo de evaluación, se proyecta la posibilidad de incluir un historial de versiones para cada evaluación retornada al docente, permitiendo así una trazabilidad detallada de los cambios realizados. Asimismo, podría incorporarse un campo obligatorio para que el coordinador justifique el retorno de la evaluación, mejorando la comunicación académica y la claridad del proceso. También se plantea establecer notificaciones automatizadas que alerten a los coordinadores sobre evaluaciones pendientes de finalización, especialmente si han superado ciertos plazos establecidos.

A nivel de seguimiento, se considera valioso incluir indicadores visuales de estado para cada trabajo de grado, tales como barras de progreso, etiquetas de color o líneas de tiempo con fechas clave del proceso (asignación, evaluación, publicación). Estas representaciones permitirían a los coordinadores tener un panorama claro y ágil del avance de cada trabajo, mejorando la toma de decisiones y el control operativo.

Capítulo 7

Bibliografía

[1] M. Andreessen, "Marc andreessen on why software is eating the world." Disponible en: https://www.wsj.com/articles/SB10001424053111903480904576512250915629460, 2011. Accedido: Jun. 04, 2022.

[2] R. Romero, S. Rico, and J. Barón, "Impacto de un sistema erp en la productividad de las pyme," *Tecnura*, vol. 16, no. 34, pp. 94–103, 2012.

[3] S. Shoham and M. Perry, "Knowledge management as a mechanism for technological and organizational change management in israeli universities," *Higher Education*, vol. 57, no. 2, pp. 227–246, 2009.

[4] E. D. Seeman and M. O'Hara, "Customer relationship management in higher education," *Campus-Wide Information Systems*, vol. 23, pp. 24–34, Jan. 2006.

[5] F. C. B. L. Hernández Sampieri, *Metodología de la investigación.* tercera edición ed., 2014.

[6] "Qué son las aplicaciones web y 8 ejemplos." https://blog.hubspot.es/website/que-es-aplicacion-web.

[7] M. Piattini, *Ingeniería del Software: Un enfoque basado en competencias.* México D.F.: McGraw-Hill, 2010.

[8] S. Durolimia, "¿qué es la transformación digital?." https://soniadurolimia.com/que-es-transformacion-digital/, 2018. Consultado el 8 de mayo de 2023.

[9] Amazon Web Services, "¿qué son las pruebas unitarias?." https://aws.amazon.com/es/what-is/unit-testing/, octubre 2023. Consultado el 8 de marzo de 2025.

[10] P. Ammann and J. Offutt, *Introduction to Software Testing.* New York: Cambridge University Press, 2008.

[11] F. P. C. Valencia, "Pruebas estáticas y dinámicas: La clave para detectar y prevenir errores, defectos y fallos en el desarrollo de software," 2023. Consultado el 1 de abril de 2025.

[12] A. Z. Chernyak, "Pruebas dinámicas en pruebas de software – ¡qué es, tipos, proceso, enfoques, herramientas y más!," 2024. Consultado el 1 de abril de 2025.

[13] A. Delgado, A. Mesquida, and A. Mas, "Uso de trello para seguimiento y coordinación en diseño de prototipos," in *Actas del Simposio XX JENUI*, vol. 1, (Oviedo), pp. 53–58, 2014. Consultado el 1 de abril de 2025.

[14] B. V. Deemer, G. Benefield, and C. Larman, "Básica de scrum (the scrum primer)," *Scrum Training Institute*, vol. 1.1, pp. 1–20, 2009.

[15] J. A. R. Navarro, "Scrum: Trabajando en equipo cliente y proveedor," 2022. Consultado el 1 de abril de 2025.

[16] Y. F. Romero, "Patrón modelo-vista-controlador," *Rev. Telem@tica*, vol. 11, no. 1, p. 11, 2012. [Online].

[17] R. D. Hernandez, "El patrón modelo-vista-controlador: Arquitectura y frameworks explicados," junio 2021. [En línea]. Consultado el 1 de abril de 2025.

[18] Cámara de Comercio de Bogotá, "El mundo conectado por las api," 2019. [En línea]. Consultado el 1 de abril de 2025.

[19] Akamai Technologies, "¿cuáles son los riesgos de seguridad de las api?," 2025. [En línea]. Consultado el 1 de abril de 2025.

[20] BBVA, "Api rest: qué es y cuáles son sus ventajas en el desarrollo de proyectos," 2022. [En línea]. Consultado el 1 de abril de 2025.

[21] S. Gutiérrez, "¿qué es una api rest?," 2024. [En línea]. Consultado el 1 de abril de 2025.

[22] COLCIENCIAS, "Niveles de madurez tecnológica (trl) y de manufactura (mrl)." `https://minciencias.gov.co/sites/default/files/upload/convocatoria/anexo9nivelesdemadureztecnorlymrl.pdf`, Nov. 2021. Consultado 20 de marzo de 2025.

[23] C. T. Isla, "Grados de maduración de la tecnología, los trls." `https://www.reordenate.cl/post/grados-de-maduraci%C3%B3n-de-la-tecnolog%C3%ADa-los-trls`, Dec. 2021. Consultado el 1 de marzo de 2025.

[24] M. O. Ahmad, J. Markkula, and M. Oivo, "Kanban in software development: A systematic literature review," in *2013 39th Euromicro Conference on Software Engineering and Advanced Applications*, pp. 9–16, IEEE, 2013.

[25] Daira Soluciones Empresariales, "Servidores — dell, azure y modelo híbrido." `https://www.dairaps.com/servidores-dell-azure-y-modelo-hibrido/`, 2024. [En línea]. Consultado el 10 de abril de 2025.

[26] "Repositorio frontend." `https://gitlab.com/r3g0r/academiappmaster`. Consultado con permisos de acceso el 20 de abril de 2025.

[27] "Repositorio backend." `https://gitlab.com/andresriascospareja/academiapp-back`. Consultado con permisos de acceso el 20 de abril de 2025.

[28] "Entorno de pruebas." `https://test.academiapp.app/#/login/votings`. Entorno de pruebas del sistema.