Escuela Superior Politécnica del Litoral

Facultad de Ingeniería en Electricidad y Computación

Automatización de pruebas unitarias y de integración del sistema AGATA

Previo la obtención del Título de:

Ingeniero en Ciencias de la Computación

Presentado por:

Richard David Cevallos Apolo

Ricardo Aurelio Rivera Guerra

Guayaquil - Ecuador

Año: 2023

Dedicatoria

El presente proyecto lo dedico a mis padres, que me han dado mucho más de lo que podría pedir en esta vida. Y lo más importante su apoyo y amor.

Ricardo Aurelio Rivera Guerra.

El presente proyecto se lo dedico a mis padres, cuyo amor y sacrificio me han llevado hasta aquí; a mi novia, quien ha estado a mi lado en cada etapa de este viaje, y a mi amado hijo, quien me da la razón para esforzarme todos los días.

Richard David Cevallos Apolo.

Agradecimientos

Mi más sincero agradecimiento a mi familia y compañeros que me han acompañado en todo este proceso de desarrollo como la persona que soy ahora. Y también quiero agradecer a todos los profesores que me han preparado profesionalmente para mi futuro como Ingeniero de la ESPOL.

Ricardo Aurelio Rivera Guerra

Quiero expresar mi más sincero agradecimiento a mi familia que siempre me ha apoyado a lo largo de mi carrera y a mis profesores cuyas enseñanzas y orientación fueron esenciales para mi desarrollo académico.

Richard David Cevallos Apolo

Declaración Expresa

“Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; Richard Cevallos y Ricardo Rivera damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

Richard David Cevallos Apolo

Ricardo Aurelio Rivera Guerra

Evaluadores

Firmado electrónicamente por:
ERICK VICENTE LAVID CEDENO

Erick Vicente Lavid Cedeño

Profesor de Materia

Firmado electrónicamente por:
DAVID ALONSO JURADO MOSQUERA

David Alonso Jurado Mosquera

Tutor de proyecto

Resumen

La ejecución de pruebas es de vital importancia en la actualidad para asegurar la calidad de un sistema o producto. Se requirió la elaboración de un plan de pruebas que tiene como objetivo principal evaluar y mejorar el sistema AGATA, diseñado para la gestión de archivos de titulación de los estudiantes de la FIEC. La justificación radica en la necesidad de pruebas exhaustivas para asegurar la funcionalidad y calidad del sistema y poder ser lanzada al ambiente de producción. En la etapa de desarrollo, se utilizaron técnicas de pruebas unitarias y de integración para verificar y validar diferentes módulos del sistema AGATA, además que se actualizaron pruebas de las iteraciones previas del sistema completando así el resto del proceso. Los resultados obtenidos demostraron que las pruebas permitieron identificar errores y mejorar la función general del sistema. Se resaltan la optimización de procesos, la interacción efectiva entre componentes y la adaptabilidad del sistema a futuras mejoras. En conclusión, el análisis final permitió destacar la relevancia de las pruebas en el desarrollo de sistemas web y cómo la implementación rigurosa garantiza productos funcionales.

Palabras Clave: Verificación, Evaluación, Pruebas, Funcionalidad., Integración, Unitarias

Abstract

The execution of tests is of vital importance Abstract today to ensure the quality of a system or product. The development of a testing plan was required, with the main objective of evaluating and enhancing the AGATA system, designed for managing students' degree documents at FIEC. The justification lies in the need for comprehensive testing to ensure the system's functionality and quality before its deployment in the production environment. During the development phase, unit testing and integration techniques were used to verify and validate various modules of the AGATA system. Additionally, tests from previous iterations were updated to complete the testing process. The obtained results demonstrated that testing identified errors and improved the overall system function. Highlights include process optimization, effective interaction between components, and system adaptability for future enhancements. In conclusion, the final analysis underscored the importance of testing in web systems development and how rigorous implementation ensures functional products.

Keywords: *Verification, Evaluation, Testing, Functionality, Integration, Unit.*

Índice general

Abreviaturas

ESPOL	Escuela Superior Politécnica del Litoral
FIEC	Facultad de Ingeniera en Electricidad y Computación
AGATA	Aplicación web para la gestión de Archivos de titulación
SAAC	Sistema de Administración Académica
STA	Secretaría Técnica Académica
API	Interfaz de Programación de Aplicaciones

Índice de figuras

Índice de tablas

Capítulo 1

1.1 Introducción

La complejidad inherente de los sistemas web modernos plantea la necesidad de realizar pruebas adecuadas. Estos sistemas están compuestos por múltiples componentes, servicios y API's que deben interactuar de manera fluida para garantizar un correcto funcionamiento en su conjunto. Los errores de integración pueden tener un impacto significativo en la funcionalidad, rendimiento, seguridad y experiencia del usuario final. Por lo tanto, las pruebas de integración se convierten en una herramienta esencial para identificar y prevenir estos problemas antes de que lleguen a producción. Según Chiu [7] las pruebas de software son una inversión que tiene como objetivos principales detectar errores, mejorar la calidad del producto, reducir riesgos y cumplir con compromisos y requisitos establecidos. Además, al realizar pruebas efectivas, se logra un software más confiable, seguro y apto para su uso, ya que se encargan de buscar todos o la mayoría de los errores que pueden impedir el correcto funcionamiento del software.

1.2 Descripción del problema

La Facultad de Ingeniería en Electricidad y Computación (FIEC) de ESPOL está trabajando en la etapa final de desarrollo de un sistema denominado AGATA (Aplicación web para la gestión de Archivos de titulación). El sistema consta de dos etapas que tienen como objetivo brindar orientación y apoyo al estudiante durante el proceso de titulación, con la participación de otros usuarios como tutores, secretarias y docentes. Estos usuarios tienen acceso al sistema, lo que les permite visualizar los documentos presentados por el estudiante y proporcionar retroalimentación. Mediante notificaciones, todos los involucrados en el proceso reciben alertas sobre el estado actual.

La segunda etapa, se enfoca en brindar a la secretaría un control y acceso a los servicios que gestionan la información no solo del estudiante, sino también del tutor y del profesor, a través de una API. Este proceso comienza después de que el documento del proyecto ha sido verificado y aceptado durante la primera etapa. Los estudiantes proceden a registrar sus documentos personales, mientras que la secretaría revisa que estén en orden y toma el acta de evaluación utilizando el SAAC (Sistema de Administración Académica). A continuación, se envía el acta para su firma al subdecanato y, finalmente, se entrega al STA (Secretaría Técnica Académica) para que realice el trámite correspondiente para emitir el título de grado del estudiante.

El sistema de gestión académica AGATA empezó a desarrollarse desde el PAO1 – 2021 y ha pasado por varias actualizaciones para cumplir con los nuevos requerimientos que se han presentado. Se han realizado pruebas unitarias del sistema en los módulos de secretaría, profesor y tutor, pero no se ha probado el sistema en su totalidad como en los módulos de estudiante y subdecanato y las nuevas funcionalidades que se han ido implementando. Este sistema presenta una carencia crítica de pruebas unitarias y pruebas de integración. Estas pruebas son esenciales para asegurar la calidad y el correcto funcionamiento del software.

Como explica Pressman [2], la falta de pruebas unitarias y de pruebas de integración tiene consecuencias en el desarrollo de software. Las pruebas unitarias evalúan los componentes individuales de un sistema de manera aislada, lo que permite identificar y corregir errores específicos en cada uno de ellos. Si estas pruebas no se realizan adecuadamente o se omiten por completo, existe un riesgo de que los errores no se detecten y se propaguen en el sistema, lo que puede resultar en un software defectuoso.

Por otro lado, las pruebas de integración son esenciales para verificar la correcta interacción entre los diferentes componentes de un sistema, como explica Beltrán [8] Sin estas pruebas, es difícil garantizar que los componentes se comuniquen de manera efectiva, intercambien datos correctamente y coordinen sus acciones de manera adecuada. Esto puede conducir a problemas de compatibilidad, donde los componentes no funcionan bien juntos debido a diferencias en formatos de datos, protocolos de comunicación u otros aspectos de la interfaz.

1.3 Justificación del problema

Para que el sistema finalmente salga producción es necesario realizar las pruebas adecuadas dado que en su estado actual puede generar problemas de integración entre los diferentes módulos del sistema. Por lo tanto, las pruebas de integración y unitarias se vuelven fundamentales para asegurar que los diferentes componentes del sistema funcionen correctamente y se integren de manera adecuada. Estas pruebas permiten identificar y corregir errores tempranos, validar la interoperabilidad entre los diversos módulos y verificar que las funcionalidades individuales cumplan con los requisitos establecidos, lo que permitiría finalmente uso del sistema al cuerpo estudiantil y directivo de la FIEC que por consecuente generaría retroalimentación de parte del usuario que ayudaría a identificar áreas donde puede haber mejora, corregir errores y realizar actualizaciones que impulsen el crecimiento y la evolución continua del programa.

1.4 Objetivos

1.4.1 Objetivo general

Evaluar el correcto funcionamiento de todos los módulos del sistema AGATA aplicando un plan de pruebas de software integral.

1.4.2 Objetivos específicos

1. Diseñar un plan de pruebas integral para la evaluación completa del sistema.
2. Automatizar la verificación y ejecución del plan pruebas diseñado del sistema.
3. Corregir los errores de mayor impacto en el sistema.

1.5 Marco teórico

1.5.1 Pruebas unitarias o funcionales

Gonzales [1], afirma que las pruebas funcionales, también conocidas como "Functional Testing", tienen como finalidad identificar inconsistencias, asegurar requisitos funcionales, reducir costos de no conformidades, evitar reprocesos, mejorar la productividad y aumentar la satisfacción del cliente. Estas pruebas se centran en probar que los sistemas desarrollados cumplan con las funciones específicas para las que han sido creados. Se suelen realizar en la última etapa de pruebas y, al dar conformidad, el siguiente paso es el pase a producción. Estas pruebas también se conocen como pruebas de comportamiento o pruebas de caja negra, ya que no se enfocan en cómo se generan las respuestas del sistema, sino en el análisis de los datos de entrada y salida, lo cual se define en los casos de prueba preparados antes del inicio de las pruebas.

1.5.2 Pruebas de integración

Según Pressman [2], las pruebas de integración son esenciales para asegurar el funcionamiento efectivo de los componentes de software cuando se combinan. Durante la integración, se abordan problemas como pérdida de datos, efectos adversos entre componentes y subfunciones que no producen los resultados esperados. Estas pruebas siguen un enfoque sistemático que involucra construir la arquitectura del software y detectar errores

relacionados con la interfaz. El objetivo es combinar componentes probados individualmente y crear una estructura de programa coherente con el diseño previsto.

Beltrán [8] resalta la importancia de las pruebas de integración dado que juegan un papel central al garantizar que las diferentes partes del sistema se unan de manera efectiva y sin problemas, a pesar de la automatización y la integración continua. Al enfocarse en la comunicación entre el cliente y el servidor, las pruebas de integración se convierten en la línea de defensa esencial para identificar posibles problemas y conflictos en esta interacción crucial.

Al igual que como demuestra Arango y Javier [9] en su trabajo con la Plataforma Educativa ZERA 2.0, con las pruebas de integración se aseguraron de que los elementos dentro del esquema de ejecución operen de manera adecuada al ser integrados para llevar a cabo una función en conjunto. Estos procedimientos de evaluación identifican fallos en las definiciones de las interconexiones de los módulos.

1.5.3 Herramientas de Pruebas

1.5.3.1 Jest

Según Burnham [3], Jest es un corredor de pruebas, una biblioteca de afirmaciones y una biblioteca de simulación. Jest también proporciona una herramienta integrada de cobertura de pruebas y una función llamada test con instantánea, el cual es un tipo de prueba que captura el estado actual de una sección de código y lo compara con una versión previamente guardada, lo que permite verificar cambios visuales o estructurales en interfaz del usuario. A diferencia de algunos de sus predecesores como Jasmine, Jest no requiere un entorno de navegador real para ejecutarse. En cambio, se ejecuta dentro de un proceso de Node.js donde las API del navegador se emulan con Jsdom. Un entorno de navegador simulado acelera la ejecución de pruebas y permite ejecutar pruebas en diferentes sistemas

con los mismos resultados. Las mismas pruebas se pueden ejecutar tanto en la máquina de un desarrollador como en un servidor de entrega continua.

1.5.3.2 Selenium

Según Gundecha [4], Selenium es una herramienta de prueba de automatización de código abierto que permite realizar pruebas funcionales en aplicaciones web. Se puede utilizar Selenium para simular la interacción del usuario con la aplicación web y verificar que los diferentes componentes se integren correctamente.

1.5.3.3 Jest Mock Functions

En su sitio web describe a Mock Functions como [6]: una característica de Jest, un *framework* de pruebas utilizado en el ecosistema de JavaScript, que permite crear y utilizar mocks (simulaciones) de funciones de manera sencilla. Estos mocks pueden ser utilizados para simular el comportamiento de funciones reales en pruebas unitarias y facilitar la creación de entornos controlados para las pruebas.

Con las Mock Functions, se pueden crear objetos simulados que imitan el comportamiento de objetos reales, pero de una manera controlada y predefinida. Esto es útil para eliminar dependencias externas en las pruebas unitarias y garantizar que las pruebas se centren en el código específico que se está probando. Al finalizar la prueba se debe eliminar la información generada en la prueba realizada.

1.5.3.4 Django Testing Framework

Se define en el sitio web como [5]: Un conjunto de herramientas y utilidades proporcionadas por Django, el popular *framework* de desarrollo web en Python, para facilitar la realización de pruebas automatizadas en aplicaciones Django. Se puede aprovechar este "framework" para realizar pruebas de integración específicas de Django, como probar las vistas, modelos y otros componentes de la aplicación Django. Las herramientas de pruebas de

Django están diseñadas para ayudar a los desarrolladores a escribir pruebas unitarias, pruebas de integración y pruebas funcionales de manera eficiente.

Capítulo 2

2.1 Metodología.

Para cumplir los objetivos planteados optamos por la metodología de un plan de pruebas que es un documento que establece el enfoque, la estrategia y los detalles de las actividades de prueba que se llevarán a cabo en un proyecto o sistema. Decidimos diseñar un plan de pruebas porque nos permite organizar mejor los objetivos. Realizamos un plan de pruebas que cubra todos los aspectos funcionales del sistema a partir de pruebas funcionales para los módulos que no han sido probados aun y pruebas de integración para validar que todo el sistema funcione en conjunto.

2.2 Análisis.

Después de tener las entrevistas pertinentes con el cliente, se identificaron los requisitos necesarios para abordar de manera específica las necesidades planteadas por el cliente con relación al sistema AGATA.

2.2.1 Requerimientos

- Realizar las pruebas unitarias y de integración de los módulos del sistema que aún no han sido probados.
- Probar la integración con servicios externos.
- Corregir los errores que se encuentren en las pruebas.

2.2.3 Alcance y limitaciones de la solución

2.2.3.1 Alcance

Se llevarán a cabo pruebas en los módulos del sistema que hasta el momento no han sido realizadas. Estos son los módulos del estudiante, notificaciones y recordatorios. Asimismo, se probarán nuevas funcionalidades que se han añadido a los módulos de secretaría y subdecanato.

2.2.3.2 Limitaciones

No se proporcionará solución a los errores derivados del uso de servicios externos, pero si se reportarán dichos errores y de igual forma el sistema debe ser tolerante a estos fallos mediante el uso de excepciones que eviten que se caiga en el caso que suceda.

2.2.4 Riesgos y beneficios de la solución

2.2.4.1 Beneficios

- Al realizar pruebas unitarias y de integración de manera regular, se detectan problemas y errores de forma más rápida y eficiente.
- Con las pruebas se garantiza que los diferentes componentes y módulos de un sistema funcionen correctamente de manera individual y en conjunto.

2.2.4.2 Riesgos

- Que las pruebas de software ejecutadas no sean capaces de identificar fallos significativos en el código, lo cual puede tener un impacto negativo en la funcionalidad del sistema.
- Que no exista tiempo suficiente de lo planificado, para solucionar todos los errores encontrados, por lo tanto, solo se solucionarían los errores de mayor impacto.

2.2.5 Usuarios de la solución

- **Personal administrativo:** tendrá la capacidad de utilizar el sistema en los módulos que abarquen la gestión de documentación personal y la firma de documentos, tanto por parte de la secretaría como del subdecanato.
- **Estudiante:** utilizará el sistema para poder validar su tesis con las autoridades de la universidad. Podrá subir la información de su trabajo de titulación para que sea revisado por las autoridades, podrá validar el nombre que debe aparecer en su título y subir su hoja de vida.

- **FIEC:** utilizará el sistema con el propósito de mejorar y automatizar su proceso de titulación dentro de la facultad.

2.3 Plan de pruebas

2.3.1. Introducción

El documento presenta el Plan de Pruebas del sistema AGATA, el cual es una plataforma diseñada para facilitar y optimizar el proceso de gestión de la documentación relacionada con las tesis de los estudiantes, brindando un entorno seguro, eficiente y confiable.

2.3.2 Objetivo

El objetivo principal de este plan de pruebas es garantizar que AGATA funcione correctamente, cumpliendo con los requisitos funcionales y no funcionales establecidos, y que proporcione una experiencia fluida y satisfactoria para los usuarios. Además, se busca identificar y corregir los defectos o errores de mayor impacto a la integridad del sistema según lo acordado con el cliente, con el fin de ofrecer un sistema robusto y confiable.

2.3.3 Alcance

El alcance de las pruebas abarca todos los módulos del sistema AGATA. Se realizarán pruebas exhaustivas y detalladas, cubriendo tanto las funcionalidades básicas como las características avanzadas del sistema.

2.3.4 Requerimientos

- El usuario con rol secretaría pueda crear períodos académicos con sus respectivas etapas del proceso de titulación.
- El usuario con rol estudiante pueda verificar que los datos del proyecto integrador estén presentes.
- El usuario con rol estudiante puede subir un archivo de titulación y verificar que el título de este coincida con el título registrado en el sistema de proyectos.

- El usuario con rol estudiante puede verificar que se suba correctamente el resumen/abstract, las palabras clave, el ODS y los entregables.
- El usuario con rol tutor o profesor puede tanto aprobar el proyecto del estudiante como devolverlo. Asimismo, verificar que el documento pase de estado “No Iniciado” a “Aprobado” o “Devuelto” según sea el caso.
- Una vez que el tutor y el profesor aprueben el proyecto del estudiante, verificar que el documento pase de estado “No Iniciado” a “Aprobado”.
- En el usuario con rol secretaría se debe verificar que el título del proyecto dentro del documento coincide con el registrado en el sistema.
- Verificar que cuando el proyecto sea devuelto el estudiante pueda agregar un nuevo entregable.
- Verificar que cuando el proyecto sea devuelto el estudiante pueda eliminar un entregable.
- Verificar que el usuario con rol secretaría pueda descargar el acta de evaluación de SAAC.
- Verificar que el usuario con rol secretaría pueda subir a GTSI la información del proyecto de titulación.
- Verificar que el usuario con rol secretaría suba correctamente el acta de evaluación al SAAC.
- Verificar que el subdecano pueda firmar las actas.
- El usuario con rol secretaría verificar que cuando se descargue el acta de evaluación se debe agregar el nombre del estudiante en una hoja adicional adjunta al mismo documento en formato pdf.
- Verificar que las notificaciones de proyecto subido al profesor, tutor y secretaria, y que el al estudiante se le aprobó el proyecto se envíen correctamente.

2.3.5 Diseño de las pruebas

Dentro del Plan de Pruebas del sistema AGATA, se incluirán tanto las pruebas unitarias como las pruebas de integración en todos los módulos del sistema. Estas pruebas son esenciales para asegurar la calidad y el correcto funcionamiento del sistema en su conjunto. A continuación, se detallan las estrategias específicas para cada tipo de prueba:

Pruebas Unitarias:

- Se realizarán pruebas unitarias en cada uno de los módulos individuales que componen el sistema AGATA.
- Se diseñarán casos de prueba específicos para validar las funciones y componentes a nivel de unidad.
- Se verificará el comportamiento de cada módulo de forma aislada, asegurando su correcta funcionalidad y cumplimiento de los requisitos establecidos.
- Se comprobará el flujo de datos y la lógica interna de cada módulo, identificando posibles errores o comportamientos inesperados.

Pruebas de Integración:

- Se realizarán pruebas de integración para verificar la correcta interacción y comunicación entre los módulos del sistema AGATA.
- Se diseñarán casos de prueba que abarquen diferentes escenarios de integración, asegurando la coherencia y consistencia en el intercambio de datos.
- Se validarán los flujos de información entre los distintos módulos, comprobando que los datos se transmitan correctamente y que se mantenga la integridad de la información.
- Se evaluará la compatibilidad y la interoperabilidad entre los módulos, identificando posibles conflictos o incompatibilidades.

Estas pruebas se llevarán a cabo siguiendo un enfoque sistemático y exhaustivo. Se diseñarán casos de prueba relevantes y se asignará a los responsables para su ejecución y

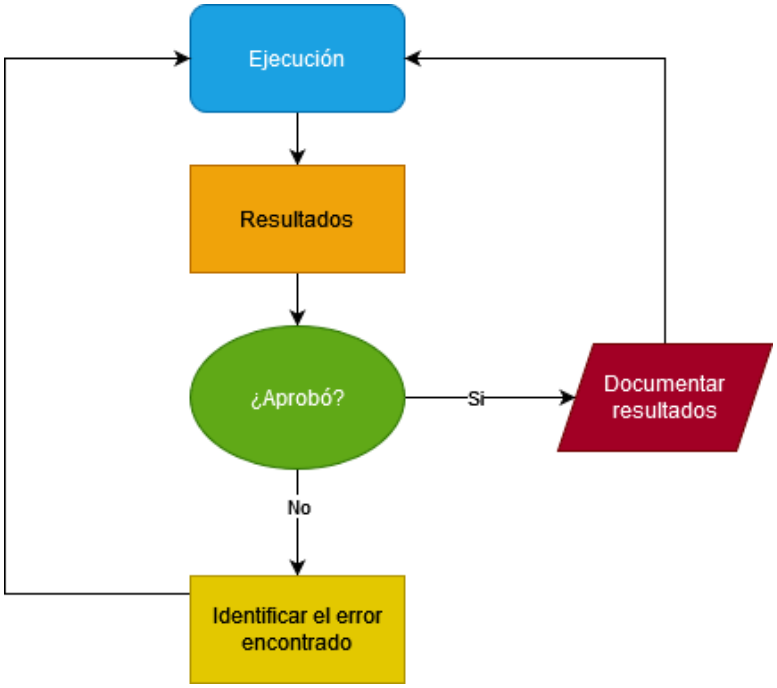
seguimiento. Los resultados de las pruebas unitarias y de integración se registrarán detalladamente.

El objetivo de estas pruebas es asegurar que todos los módulos del sistema AGATA funcionen correctamente de forma individual e integrados entre sí, garantizando un sistema robusto, coherente y confiable en su conjunto.

2.3.6 Ejecución de las pruebas

Al momento de empezar la ejecución de las pruebas, se las realizar de acuerdo con el diagrama mostrado a continuación en las pruebas unitarias y de integración.

Figura 1
Diagrama de ejecución de pruebas



Se deben probar todos los módulos requeridos para cumplir los requerimientos y las pruebas que no dan un resultado serán corregidas según su grado de impacto y los requerimientos del cliente.

2.3.7 Cierre de pruebas

Para el cierre de las pruebas se presentará el informe de pruebas donde se documentará el resultado de cada una de las diferentes pruebas ejecutadas.

2.3.8 Entregable de pruebas

El entregable de las pruebas unitarias y de integración en el contexto del Plan de Pruebas del sistema AGATA sería un conjunto de artefactos y documentación que refleje los resultados obtenidos durante la ejecución de las pruebas. Estos entregables son listados a continuación:

Tabla 1
Entregables respectivos a cada una de las pruebas.

Tipo de pruebas	**Entregables**
Pruebas unitarias	• Informe de pruebas unitarias • Evidencia de prueba
Pruebas de integración	• Informe de pruebas de integración • Evidencia de prueba

2.4 Herramientas seleccionadas para la solución

Para las pruebas unitarias en el *FrontEnd* se utilizará Jest, que permite crear y ejecutar pruebas unitarias para los módulos y funciones del *FrontEnd*, asegurando que su comportamiento sea el esperado. Además de Jest, se empleará una herramienta de simulación, Mock de Jest, en el contexto de las pruebas unitarias, para simular varios objetos y así completar la ejecución de procesos de forma satisfactoria e independiente. Para las pruebas de integración se utilizará Selenium, donde se podrá simular interacciones y escenarios reales de los usuarios, permitiendo verificar la integración correcta entre los componentes del *FrontEnd* y el *BackEnd*.

2.5 Cronograma

Tabla 2
Cronograma de actividades planificado del proyecto integrador.

Fecha de entrega	**Actividad**
1 – 14 Mayo	Definir requerimientos
8 – 21 Mayo	Definir plan de pruebas
22 Mayo – 30 Junio	Ejecución de pruebas
14 – 30 Junio	Clasificar errores encontrados
26 – 30 Junio	Seguimiento con el cliente
3 – 14 Julio	Corrección de errores de mayor impacto
17 Julio – 4 Agosto	Ejecución de pruebas finales
8 – 15 Agosto	Entrega del producto

Capítulo 3

3.1 Resultados y análisis

Los resultados obtenidos tras la implementación y ejecución del plan de pruebas demostraron el cumplimiento exitoso de los objetivos planteados en cada módulo. La secretaría pudo iniciar y gestionar períodos académicos, los estudiantes accedieron y completaron sus proyectos integradores, los tutores y profesores revisaron y aprobaron documentos, y se manejaron notificaciones y recordatorios de manera eficiente.

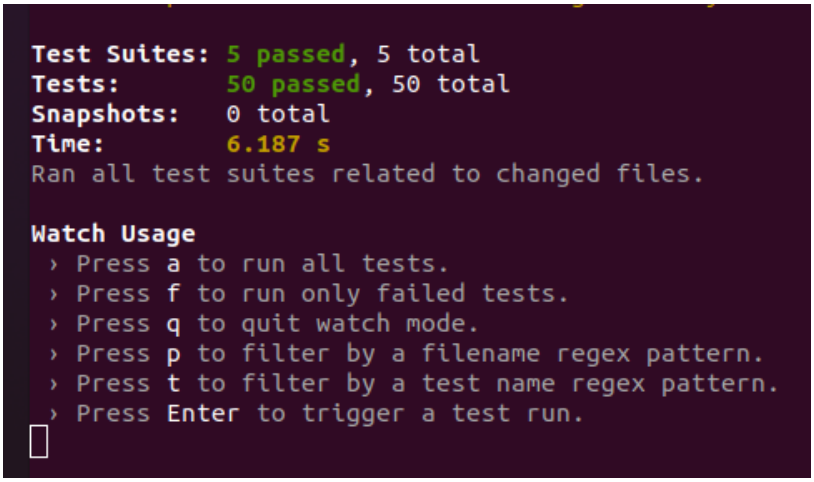
Las pruebas de integración con Selenium y pruebas unitarias con Jest aseguraron que las funcionalidades en cada módulo interactuaran adecuadamente y produjeran los resultados esperados. Finalizamos la etapa de pruebas de forma exitosa logrando probar todos los módulos requeridos de forma satisfactoria y demostrando la funcionalidad completa del sistema.

Durante el desarrollo del plan de pruebas nos encontramos con algunos inconvenientes que retrasaron la entrega de la solución, uno de esos inconvenientes fue la implementación errónea de las pruebas, al funcionar de forma secuencial en lugar de forma independiente, lo que ocasiono retrabajo por nuestra parte y retraso en general la planificación original.

3.1.1 Desarrollo de Pruebas

Por el lado de las pruebas unitarias desarrolladas con Jest, como primera tarea se tuvo que comprobar que las pruebas anteriores funcionaran sin mayor inconveniente y en el caso que no den un resultado satisfactorio, buscar una solución. Luego se procedió con la creación de las pruebas para las nuevas funcionalidades añadidas desde la última iteración del sistema AGATA, se hizo uso de jest mock para poder hacer una simulación de los componentes necesarios para las pruebas de forma que puedan ejecutarse de forma independiente al éxito de las demás pruebas.

Figura 2
Captura de los resultados de las pruebas unitarias en Jest



Nota. El resto de las capturas de las pruebas unitarias se las puede encontrar en el Apéndice A.

Durante el desarrollo de las pruebas se tuvo que realizar ajustes a las pruebas anteriores de implementaciones previas del sistema dado que ya estaban desactualizadas y no daba un resultado satisfactorio a causa de los cambios de parámetros y funcionalidades del sistema.

Tabla 3
Tabla de pruebas unitarias que fueron ajustadas.

Prueba	**Resultado insatisfactorio**	**Solución**
Prueba unitaria el profesor: No puede aprobar proyecto que no ha sido subido	Se devolvía un mensaje erróneo.	Se ajustaron los parámetros y el mensaje.
Prueba unitaria el profesor: No puede aprobar proyecto devuelto por el tutor.	Se devolvía un mensaje erróneo.	Se ajustaron los parámetros y el mensaje.
Pruebas relacionadas a hoja de vida	La función de subir hoja de vida fue eliminada.	Se borraron las pruebas.
Pruebas unitarias tutor: Mostrar el proyecto en diferentes escenarios.	Valores erróneos por parámetros desactualizados	Se actualizaron los parámetros.

Con respecto a las pruebas de integración desarrolladas con Selenium, se tuvo que crear los datos necesarios para el funcionamiento de cada prueba, y asimismo eliminar los datos de prueba anteriores al iniciar su ejecución, debido a que Selenium prueba el funcionamiento de la aplicación en el navegador, por lo tanto, se deben crear registros de prueba en la base de datos. Para esto se crearon *EndPoints* para la creación de datos de pruebas y asimismo un *EndPoint* para la eliminación, ambos ubicados en el *BackEnd*. Con la funcionalidad *before* de selenium se eliminan los datos creados en las anteriores pruebas y al iniciar la prueba se crean los datos llamando al *EndPoint* de creación. En la Figura 3 se muestra la estructura de una prueba de integración en la que se puede observar que se utiliza en repetidas ocasiones el método *sleep* el cual sirvió para que la prueba se termine de ejecutar sin ningún tipo de error como por ejemplo que no se encuentre un elemento del html en el navegador ya que las acciones que se realizan en la aplicación pueden demorar.

Figura 3
Captura de la prueba integral de revisión grupal de la secretaria

```
describe('revisionSecretariaGrupal', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('revisionSecretariaGrupal', async function() {
    await driver.sleep(3000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/revisionSecretaria');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).sendKeys("secre-fiec1")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(3500)
```

Nota. El resto de las capturas de las pruebas de integración se las puede encontrar en el Apéndice B.

Las pruebas funcionan de forma individual pero también se pueden ejecutar todos de forma secuencial para visualizar el flujo de funcionalidades del sistema. Asimismo, se agregó un mensaje al momento de que la secretaría intente extraer el acta de evaluación de un estudiante que en el api de GTSI aún no esté registrada. Por lo demás los resultados de las pruebas fueron positivos en general.

3.1.2 Corrección de errores

Durante la revisión individual realizada por la secretaria, se detectó un inconveniente en el sistema. Concretamente, se observó que el sistema permitía avanzar al siguiente paso incluso si el paso actual no había sido completado. Para abordar esta situación, fue necesario realizar ajustes en el código del *Backend*. A continuación, se presenta el flujo corregido en el que se puede apreciar la sincronización adecuada de los botones, habilitándolos o deshabilitándolos según las expectativas del sistema. En la Figura 4 y Figura 5 se puede observar parte del flujo.

Figura 4

Captura del step previo a revisión información del estudianteSolución de error en pasos de la revisión individual de la secretaria parte 1

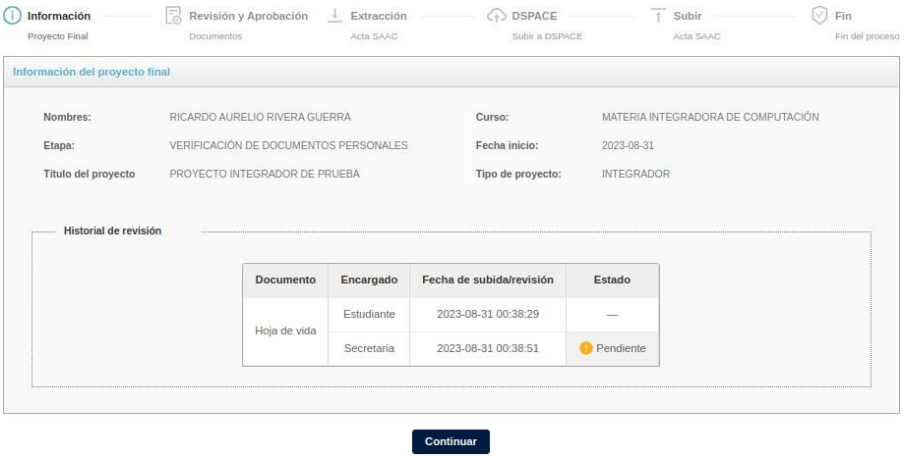
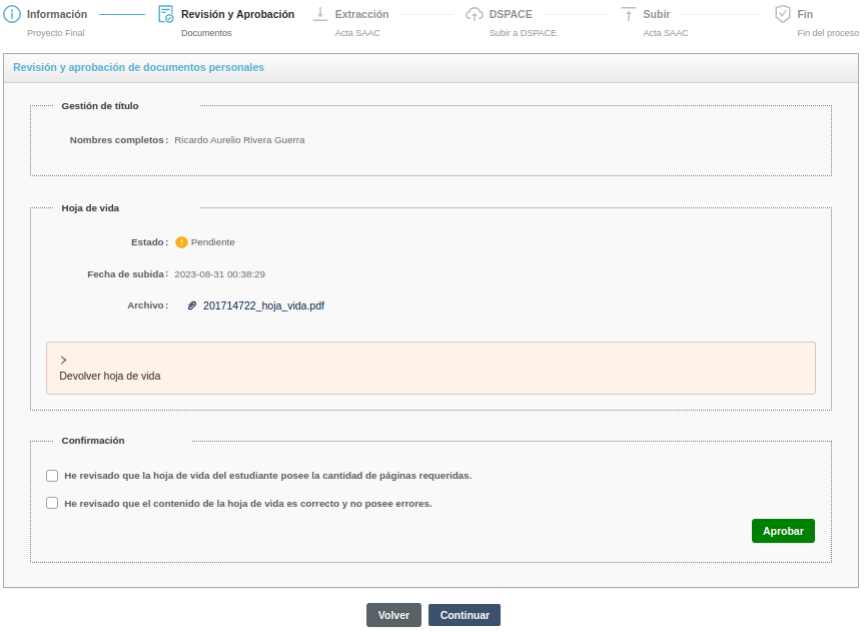


Figura 5

Captura del step de revisión y aprobación

Solución de error en pasos de la revisión individual de la secretaria parte 2

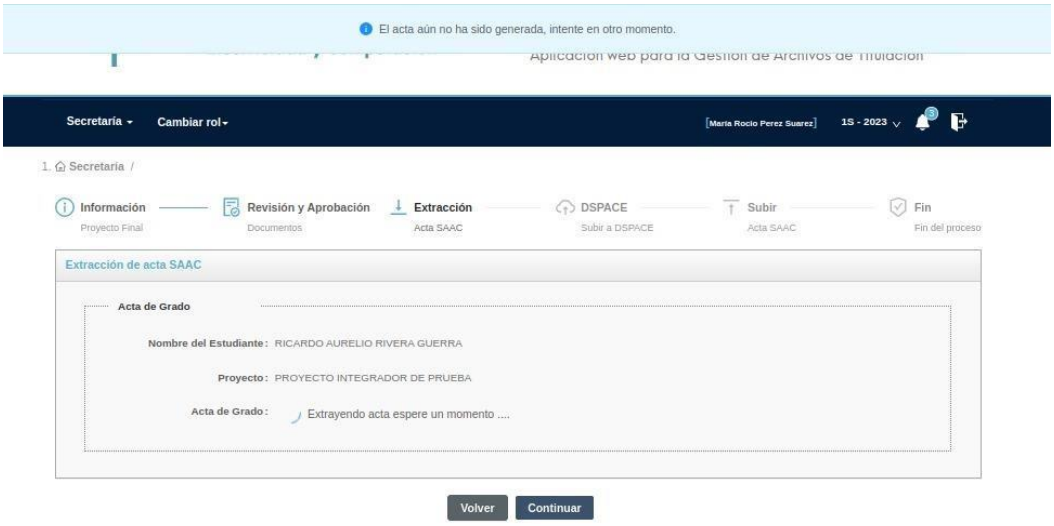


La funcionalidad de extracción de actas se integra con un API de GTSI que recupera la información de las actas de los estudiantes. Sin embargo, cuando un estudiante no cuenta aún con un acta, el API responde con un código 404. Esta situación generaba incertidumbre en la aplicación, ya que al seleccionar "extraer acta", se quedaba en un estado de espera sin proporcionar detalles claros.

Para resolver este problema, se implementó un manejo de errores. Ahora, cuando se presiona el botón de extracción de acta y el sistema detecta que no hay un acta disponible, se muestra un mensaje específico para la secretaria. Este mensaje notifica que el estudiante no tiene aún un acta. En la Figura 6 se visualiza la corrección.

Figura 6

Captura de la corrección del error en el rol de secretaria Captura de solución de error de extraer acta de GTSI



Nota. El resto de las capturas de las evidencias de correcciones se las puede encontrar en el Apéndice C.

3.1.3 Resultados

A continuación, se puede observar la tabla con las pruebas realizadas y sus resultados después de su ejecución:

Tabla 4

Tabla de pruebas realizadas en el proyecto AGATA.

Pruebas	**Descripción**	**Datos de entrada**	**Resultado**
[1] Prueba integral crear datos en el módul admin	Se crea el semestre, el proyecto, los usuarios estudiantes, secretaría, profesor, tutor y subdecano desde el administrador	Ninguno	Exitoso
[2] Prueba integral del módulo secretaría	Crear las etapas del proceso.	Periodo académico, usuario secretaria	Exitoso
[3] Prueba integral del	Se sube la información	Periodo académico,	Exitoso

módulo estudiante Título coincide	de su proyecto integrador (Palabras clave, ODS, Resumen, Archivo Final y Entregables) y se verifica que el título del archivo coincida con el del sistema.	usuarios secretaria, estudiante, y proyecto creado en el admin.	
[4] Prueba integral del módulo estudiante Título No coincide	El estudiante sube el archivo del proyecto que no contenga el título que está registrado en el sistema.	Periodo académico, usuarios secretaria, estudiante, y proyecto creado en el admin.	Exitoso
[5] Prueba integral del módulo tutor:	Revisar y aprobar el archivo que subió el estudiante.	Proyecto estudiante en estado entregado	Exitoso
[6] Prueba integral del módulo profesor:	Revisar y aprobar el archivo que subió el estudiante.	Proyecto estudiante en estado entregado y aprobado por tutor	Exitoso
[7] Prueba integral del módulo secretaria:	Revisar y aprobar el archivo que subió el estudiante.	Proyecto estudiante en estado entregado y aprobado por tutor y profesor	Exitoso

[8] Prueba integral del módulo tutor: Devolver trabajo final.	Revisar archivo del estudiante y devolverlo con un comentario.	Proyecto estudiante en estado entregado	Exitoso
[9] Prueba integral del módulo profesor: Devolver trabajo final.	Revisar archivo del estudiante y devolverlo con un comentario.	Proyecto estudiante en estado entregado y aprobado por tutor	Exitoso
[10] Prueba integral del módulo secretaria: Devolver trabajo final.	Revisar archivo del estudiante y devolverlo con un comentario.	Proyecto estudiante en estado entregado y aprobado por tutor y profesor	Exitoso
[11] Prueba integral de la secretaría: Notificación a estudiante	Notificar al estudiante por correo electrónico recordatorios de fechas del proceso.	Proyecto del estudiante subido.	Exitoso
[12] Prueba integral del módulo estudiante: Subir información personal	Subir nombres completos y hoja de vida.	Proyecto aprobado por tutor, profesor y secretaria.	Exitoso
[13] Prueba integral de la secretaría: Revisión y aprobación de documentos personales.	Revisar y aprobar nombres completos y hoja de vida del estudiante	Documentos personales subidos por el estudiante.	Exitoso
[14] Prueba integral del módulo secretaría:	Se extrae el acta de evaluación del	Documentos personales subidos por	Exitoso

Extracción del Acta SAAC.	estudiante del api del GTSI	el estudiante y aprobados por la secretaria.	
[15] Prueba integral del módulo subdecano: Firmar actas	Descargar acta del estudiante y proceder a firmarlas en el sistema.	Documentos personales subidos por el estudiante y aprobados por la secretaria, extracción de acta.	Exitoso
[16] Prueba integral del módulo secretaría: Subir trabajo final al Dspace	Subir trabajo final al Dspace	Documentos personales subidos por el estudiante y aprobados por la secretaria, extracción de acta.	Exitoso
[17] Prueba integral del módulo secretaría: Subir acta al SAAC	Subir acta al SAAC con hoja agregada con los nombres del estudiante.	Documentos personales subidos por el estudiante y aprobados por la secretaria, extracción de acta, trabajo subido al Dspace deuda no valor eliminada	Exitoso
[18] Pruebas unitarias	Verificar que pruebas	Ninguno	Exitoso

anteriores corregidas	unitarias realizadas anteriormente de módulos profesor, tutor y secretaría funcionen correctamente.		
[19] Pruebas unitarias módulo estudiante: Título coincide	El título del archivo coincide con el de la base de datos.	Periodo académico, estudiante, información del proyecto.	Exitoso
[20] Pruebas unitarias módulo estudiante: Título no coincide	El título del archivo no coincide con el de la base de datos.	Periodo académico, estudiante, información del proyecto.	Exitoso
[21] Pruebas unitarias módulo estudiante: Subir archivo integrador	Se sube correctamente el archivo con palabras clave, ODS, y Resumen.	Abstract, chips, ods, csrfToken.	Exitoso
[22] Pruebas unitarias módulo estudiante: Renderización de información	Se renderiza correctamente la información del proyecto integrador.	Información del estudiante e información del proyecto.	Exitoso
[23] Pruebas unitarias notificaciones:	Se cargan correctamente las	Id, descripción, fecha de creación, estado de	Exitoso

Notificaciones	notificaciones que son enviadas.	lectura.

3.2 Análisis de Costos:

Si este proyecto fuera desarrollado fuera del contexto de la materia integradora de los estudiantes de la FIEC, se podría realizar un análisis de costo estimado basado en el trabajo que hemos realizado en el transcurso del semestre. Tomando en cuenta que dos estudiantes fuimos los involucrados en el desarrollo de la solución, se investigó el sueldo promedio de un QA Tester que es alrededor de $900.00 y el tiempo estimado que se tardaría en completar la solución que es de dos meses aproximadamente, podríamos calcular cual sería el costo aproximado del proyecto.

Como realizamos las pruebas con herramientas de licencia libre ese costo es 0, pero dado el caso si es requerido usar licencias para programas, deberá ser tomado en cuenta para el cálculo final.

Costos de Personal:

- 2 desarrolladores x $900.00 dólares/mes = $1800.00 dólares/mes
- Duración del proyecto: 2 meses
- Costo total de personal = $3600.00 dólares

Herramientas y Licencias:

- Licencias de herramientas de pruebas: Gratis
- Costo total de herramientas y licencias = $0 dólares

Contingencias:

- Asumamos un 10% del costo total como contingencia.
- Contingencia = 10% x ($3600.00 dólares) = $360.00 dólares

Costo Total Estimado:

- Costo total de personal: $3600.00 dólares
- Contingencia: $360.00 dólares
- Costo total estimado: $3960.00 dólares

3.3 Viabilidad Económica y Tecnológica

El análisis de los costos, como se detalló previamente, demuestra que la inversión en recursos humanos y la utilización de herramientas tecnológicas están optimizadas para evitar gastos innecesarios, manteniendo un costo enfocado en lo esencial que es el personal.

Es importante destacar que la viabilidad económica no se limita simplemente a la comparación de cifras. Más allá de la evaluación de los gastos directos, este análisis considera la relación entre los costos y los beneficios que el sistema AGATA puede ofrecer a la comunidad estudiantil y administrativa.

Las herramientas y tecnologías escogidas para el desarrollo de la solución se han basado en su sostenibilidad y en la existencia de comunidades activas que respaldan su desarrollo y evolución continúa dado que son herramientas que siguen en una evolución constante y recibe mantenimiento continuo. Esto asegura que la solución no solo esté en consonancia con las tendencias tecnológicas actuales, sino que también pueda adaptarse a futuros cambios en el sistema si se da el caso de que agreguen nuevas funcionalidades o requerimientos.

3.4 Cierre de Proyecto

Para cerrar el proyecto, nos reunimos con el cliente para obtener sus conclusiones y observaciones finales, y realizamos una prueba final con usuarios reales en los servidores de la ESPOL y con un resultado satisfactorio. Como entregables de la solución tenemos:

- El plan de pruebas aceptado, detallando el análisis y metodología de ejecución de pruebas.
- Informe de pruebas unitarias e integrales, detallando inconvenientes y resultados obtenidos durante la realización de las pruebas.

Capítulo 4

4.1 Conclusiones y recomendaciones

La ejecución exitosa del Plan de Pruebas diseñado para el sistema AGATA ha proporcionado una comprensión profunda de la calidad y funcionalidad del sistema desarrollado. Mediante el uso de pruebas unitarias implementadas con Jest y pruebas de integración realizadas con Selenium, se ha llevado a cabo una evaluación exhaustiva de cada módulo y su interacción.

Además de hacernos comprender la importancia sobre la realización y ejecución de un plan de pruebas conciso y completo para garantizar la correcta funcionalidad del producto o sistema en cuestión, siendo este una necesidad para cualquier equipo de desarrollo que desee brindar un producto funcional y sólido.

4.1.1 Conclusiones

- A través de las pruebas exitosas, se ha confirmado la capacidad del sistema AGATA para cumplir con los objetivos establecidos. Desde la administración de períodos académicos hasta la revisión y aprobación de proyectos, el sistema opera de manera eficiente y de acuerdo con las expectativas del cliente.
- Por medio del diseño de un plan de pruebas hemos podido llevar a cabo con éxito la verificación del estado del sistema, permitiendo organizar mejor que funcionalidades se necesitaban evaluar.
- Las pruebas han permitido la identificación de áreas que requerían mejoras en el sistema, al igual que unos errores que necesitaban ser corregidos. Algunas funcionalidades necesitaban ajustes para alinearse con las expectativas y requisitos de los usuarios. Estas optimizaciones han contribuido a mejorar la usabilidad global del sistema.
- La corrección de errores fue clave para el desarrollo de la solución, al ser un plan de pruebas existen requerimientos que se tenían que cumplir y solo por medio de la detección y corrección de errores pudimos concretar nuestro objetivo.

4.1.2 Recomendaciones

Con la conclusión de las pruebas y la entrega de sus resultados, surgen recomendaciones para orientar futuros desarrollos del sistema. A pesar de que las pruebas confirman la preparación del sistema para su implementación, es importante considerar ciertos aspectos para su crecimiento y mejora continua.

- Se recomienda adaptar y expandir las pruebas existentes en el caso que se implementen nuevas funcionalidades o se realice un cambio en el funcionamiento del sistema. Las pruebas unitarias y de integración deben adaptarse para garantizar que las modificaciones no tengan un impacto negativo en las funcionalidades ya existentes.
- Algo para tomar en cuenta de parte del grupo de desarrollo próximo es el uso de metodologías agiles para organizar mejor las tareas a realizar y planificar reuniones recurrentes con el cliente de forma que la comunicación y retroalimentación es constante y se evita atrasos o contratiempos.

En definitiva, la etapa de pruebas y la ejecución del plan de pruebas ha sido esencial para garantizar la calidad y el rendimiento del sistema AGATA. Siguiendo estas recomendaciones y adoptando un enfoque de mejora continua, el sistema podrá cumplir sus objetivos en el proceso de para la comunidad de la FIEC.

Referencias

[1] L. González, «Método para generar casos de prueba funcional en el desarrollo de software,» *Revista Ingenierías Universidad de Medellín*, vol. 8, nº 15, pp. 29-36, 2009.

[2] R. S. Pressman, Software engineering: A practiocioner's approach, New York: Mc Grawhill, 2020.

[3] T. Burnham, Test-Driven React, Pragamtic Bookslhelf, 2019.

[4] U. Gundecha, Selenium Testing Tools Cookbook, Birmingham: Packt Publishing, 2015.

[5] «Django Official Website,» Django Software Foundation, [En línea]. Available: http://www.djangoproject.com. [Último acceso: 11 Junio 2023].

[6] S. Faber, «mockito,» [En línea]. Available: https://site.mockito.org/. [Último acceso: 12 Junio 2023].

[7] C. C. Chiu, «Las pruebas en el desarrollo de software,» Universidad Nacional Autonoma de Mexico, Mexico, 2015.

[8] N. BeltránGalvis, C. EngaritaSanguino y C. GómezLlanez, «Método de pruebas de integración en arquitecturas orientadas a servicio,» *Revista INGENIO UFPSO*, vol. 12, 2016.

[9] J. A. Arango López, «Componente para la integración de la Plataforma Educativa ZERA 2.0 con el Sistema de Gestión Universitaria,» Universidad de las Ciencias Informáticas. Facultad 4., 2017.

Apéndices

Apéndice A

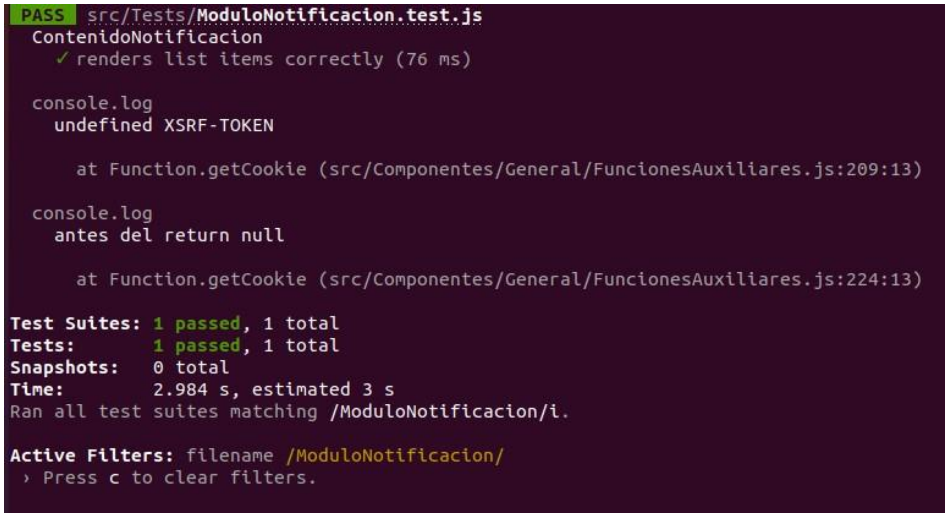
Pruebas Unitarias

Módulo Notificaciones [23]

Descripción	Datos de entrada	Resultados esperados
Las notificaciones generadas por cualquier modulo se renderice correctamente.	Id, descripción, fecha de creación, estado de lectura.	Que la información insertada se muestre renderice correctamente.

```
describe('ContenidoNotificacion', () => {
  test('renders list items correctly', () => {
    const datos = [
      { id: 1, descripcion: 'Notification 1', fechaCreacion: '2021-01-01', leido: false },
      { id: 2, descripcion: 'Notification 2', fechaCreacion: '2021-01-02', leido: true },
    ];

    const { getByText, getByTestId } = render(<ContenidoNotificacion datos={datos} />);

    expect(getByText('Notification 1')).toBeInTheDocument();
    expect(getByText('2021-01-01')).toBeInTheDocument();
    expect(getByText('Notification 2')).toBeInTheDocument();
    expect(getByText('2021-01-02')).toBeInTheDocument();

    expect(getByTestId('badge-notificacion')).toBeInTheDocument();
  });
});
```

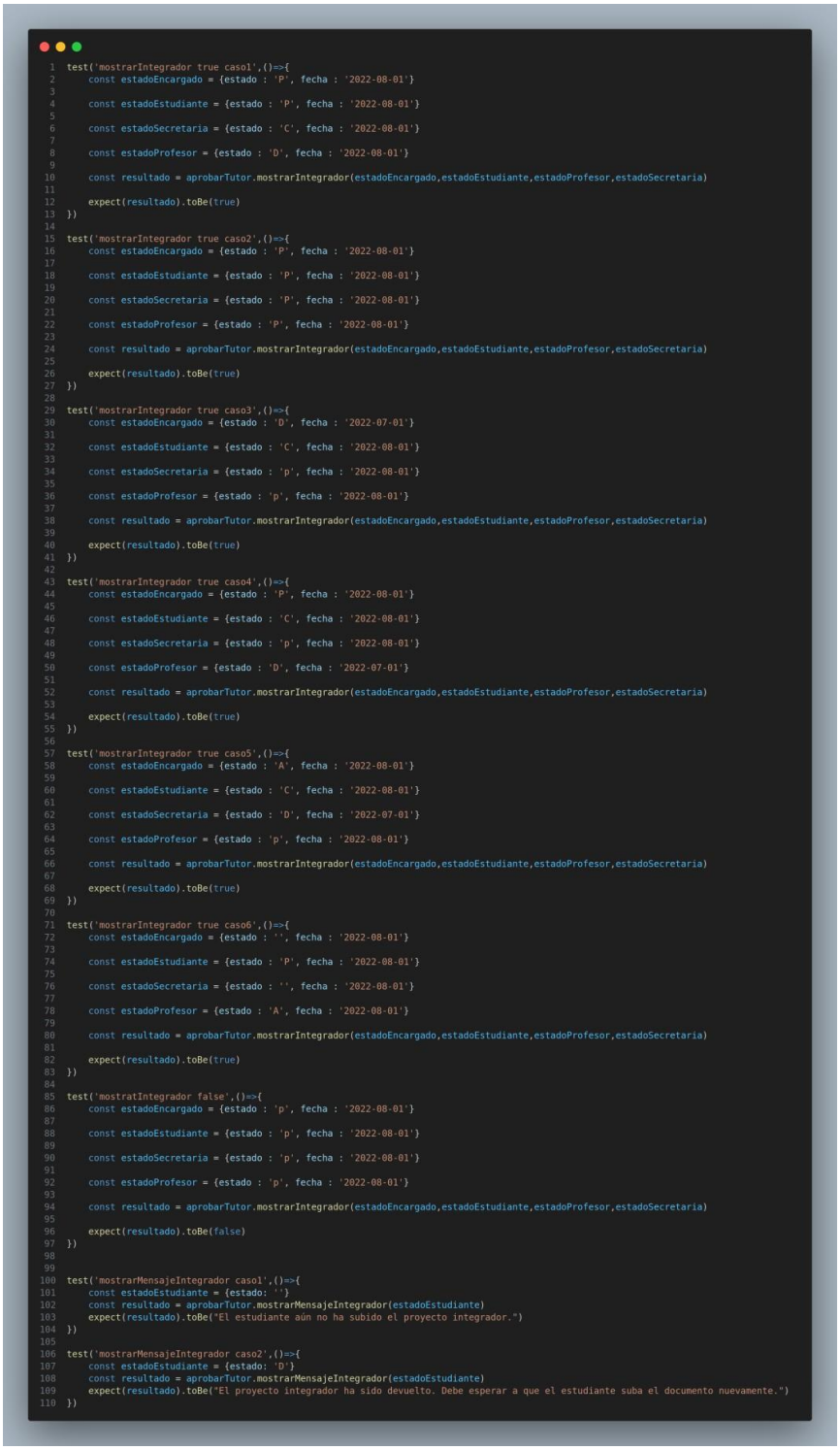
Resultado test notificaciones



```
PASS src/Tests/ModuloNotificacion.test.js
  ContenidoNotificacion
    ✓ renders list items correctly (76 ms)

  console.log
    undefined XSRF-TOKEN

      at Function.getCookie (src/Componentes/General/FuncionesAuxiliares.js:209:13)

  console.log
    antes del return null

      at Function.getCookie (src/Componentes/General/FuncionesAuxiliares.js:224:13)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        2.984 s, estimated 3 s
Ran all test suites matching /ModuloNotificacion/i.

Active Filters: filename /ModuloNotificacion/
  › Press c to clear filters.
```

Modulo Tutor [18]

Descripción	Datos de entrada	Resultados esperados
Verificar que pruebas unitarias realizadas anteriormente de módulos profesor, tutor y secretaría funcionen correctamente.	Ninguno	Que sea completadas exitosamente después de ser reajustadas.



```
test('mostrarIntegrador true caso1',()=>{
    const estadoEncargado = {estado : 'P', fecha : '2022-08-01'}

    const estadoEstudiante = {estado : 'P', fecha : '2022-08-01'}

    const estadoSecretaria = {estado : 'C', fecha : '2022-08-01'}

    const estadoProfesor = {estado : 'D', fecha : '2022-08-01'}

    const resultado = aprobarTutor.mostrarIntegrador(estadoEncargado,estadoEstudiante,estadoProfesor,estadoSecretaria)

    expect(resultado).toBe(true)
})

test('mostrarIntegrador true caso2',()=>{
    const estadoEncargado = {estado : 'P', fecha : '2022-08-01'}

    const estadoEstudiante = {estado : 'P', fecha : '2022-08-01'}

    const estadoSecretaria = {estado : 'P', fecha : '2022-08-01'}

    const estadoProfesor = {estado : 'P', fecha : '2022-08-01'}

    const resultado = aprobarTutor.mostrarIntegrador(estadoEncargado,estadoEstudiante,estadoProfesor,estadoSecretaria)

    expect(resultado).toBe(true)
})

test('mostrarIntegrador true caso3',()=>{
    const estadoEncargado = {estado : 'D', fecha : '2022-07-01'}

    const estadoEstudiante = {estado : 'C', fecha : '2022-08-01'}

    const estadoSecretaria = {estado : 'p', fecha : '2022-08-01'}

    const estadoProfesor = {estado : 'p', fecha : '2022-08-01'}

    const resultado = aprobarTutor.mostrarIntegrador(estadoEncargado,estadoEstudiante,estadoProfesor,estadoSecretaria)

    expect(resultado).toBe(true)
})

test('mostrarIntegrador true caso4',()=>{
    const estadoEncargado = {estado : 'P', fecha : '2022-08-01'}

    const estadoEstudiante = {estado : 'C', fecha : '2022-08-01'}

    const estadoSecretaria = {estado : 'p', fecha : '2022-08-01'}

    const estadoProfesor = {estado : 'D', fecha : '2022-07-01'}

    const resultado = aprobarTutor.mostrarIntegrador(estadoEncargado,estadoEstudiante,estadoProfesor,estadoSecretaria)

    expect(resultado).toBe(true)
})

test('mostrarIntegrador true caso5',()=>{
    const estadoEncargado = {estado : 'A', fecha : '2022-08-01'}

    const estadoEstudiante = {estado : 'C', fecha : '2022-08-01'}

    const estadoSecretaria = {estado : 'D', fecha : '2022-07-01'}

    const estadoProfesor = {estado : 'p', fecha : '2022-08-01'}

    const resultado = aprobarTutor.mostrarIntegrador(estadoEncargado,estadoEstudiante,estadoProfesor,estadoSecretaria)

    expect(resultado).toBe(true)
})

test('mostrarIntegrador true caso6',()=>{
    const estadoEncargado = {estado : '', fecha : '2022-08-01'}

    const estadoEstudiante = {estado : 'P', fecha : '2022-08-01'}

    const estadoSecretaria = {estado : '', fecha : '2022-08-01'}

    const estadoProfesor = {estado : 'A', fecha : '2022-08-01'}

    const resultado = aprobarTutor.mostrarIntegrador(estadoEncargado,estadoEstudiante,estadoProfesor,estadoSecretaria)

    expect(resultado).toBe(true)
})

test('mostratIntegrador false',()=>{
    const estadoEncargado = {estado : 'p', fecha : '2022-08-01'}

    const estadoEstudiante = {estado : 'p', fecha : '2022-08-01'}

    const estadoSecretaria = {estado : 'p', fecha : '2022-08-01'}

    const estadoProfesor = {estado : 'p', fecha : '2022-08-01'}

    const resultado = aprobarTutor.mostrarIntegrador(estadoEncargado,estadoEstudiante,estadoProfesor,estadoSecretaria)

    expect(resultado).toBe(false)
})


test('mostrarMensajeIntegrador caso1',()=>{
    const estadoEstudiante = {estado: ''}
    const resultado = aprobarTutor.mostrarMensajeIntegrador(estadoEstudiante)
    expect(resultado).toBe("El estudiante aún no ha subido el proyecto integrador.")
})

test('mostrarMensajeIntegrador caso2',()=>{
    const estadoEstudiante = {estado: 'D'}
    const resultado = aprobarTutor.mostrarMensajeIntegrador(estadoEstudiante)
    expect(resultado).toBe("El proyecto integrador ha sido devuelto. Debe esperar a que el estudiante suba el documento nuevamente.")
})
```

```
afterEach(cleanup)

test ('Probar si se renderiza bien la información del titulo del proyecto',() =>{
    render(<Router> <InformacionProyecto proyecto = {proyecto} /></Router>)
    expect(screen.getByText(/PROYECTO INTEGRADOR DE PRUEBA/)).toBeInTheDocument();

  })

test ('Probar si se renderiza bien el año del proyecto y si este se carga',() =>{
    render(<Router> <InformacionProyecto proyecto = {proyecto} /></Router>)
    expect(screen.queryAllByText(/2022/)).toBeTruthy();
  })

  test ('Probar si es verdad que no aparece la data como debe ser cargada en el tipo de  tesis',() =>{
    render(<Router> <InformacionProyecto proyecto = {proyecto} /></Router>)
    expect(screen.queryAllByText(/INTEGRADOR/)).toBeDefined()

  })
```

Resultados test tutor

```
PASS src/Tests/ModuloTutor.test.js
  ✓ mostrarIntegrador true caso1 (2 ms)
  ✓ mostrarIntegrador true caso2 (3 ms)
  ✓ mostrarIntegrador true caso3 (1 ms)
  ✓ mostrarIntegrador true caso4 (1 ms)
  ✓ mostrarIntegrador true caso5 (1 ms)
  ✓ mostrarIntegrador true caso6 (1 ms)
  ✓ mostratIntegrador false (1 ms)
  ✓ mostrarMensajeIntegrador caso1 (1 ms)
  ✓ mostrarMensajeIntegrador caso2 (1 ms)
  ✓ Probar si se renderiza bien la información del titulo del proyecto (44 ms)
  ✓ Probar si se renderiza bien el año del proyecto y si este se carga (23 ms)
  ✓ Probar si es verdad que no aparece la data como debe ser cargada en el tipo de  tesis (20 ms)

A worker process has failed to exit gracefully and has been force exited. This is likely caused by
Test Suites: 1 passed, 1 total
Tests:       12 passed, 12 total
Snapshots:   0 total
Time:        3.744 s, estimated 4 s
Ran all test suites matching /ModuloTutor/i.

Active Filters: filename /ModuloTutor/
 › Press c to clear filters.
```

Modulo Profesor [18]

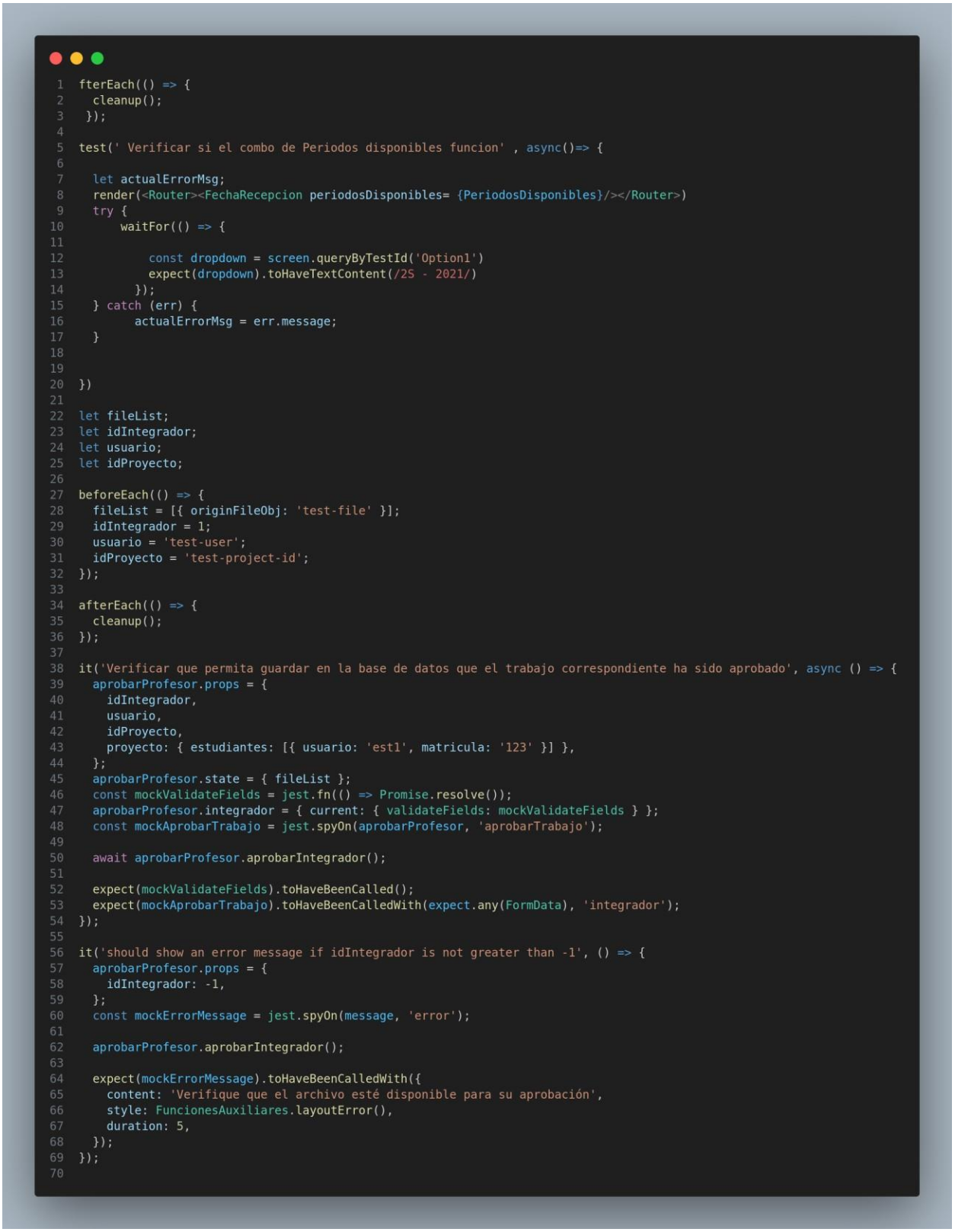
Descripción	Datos de entrada	Resultados esperados
Verificar que pruebas unitarias realizadas anteriormente de módulos profesor, tutor y secretaría funcionen correctamente.	Ninguno	Que sea completadas exitosamente después de ser reajustadas.

```
test('mostrarMensajeIntegrador caso1',()=>{
    const estadoEstudiante = {estado : '', fecha : ''}
    const resultado = aprobarProfesor.mostrarMensajeIntegrador(estadoEstudiante)
    expect(resultado).toBe("El estudiante aún no ha subido el trabajo integrador.")
})

test('mostrarMensajeIntegrador caso2',()=>{
    const estadoEstudiante = {estado : 'P', fecha : ''}
    const resultado = aprobarProfesor.mostrarMensajeIntegrador(estadoEstudiante)
    expect(resultado).toBe("El trabajo final ha sido devuelto debe esperar a que el tutor lo apruebe.")
})
```

Resultado test profesor

```
PASS src/Tests/ModuloProfesor.test.js
  ✓ mostrarMensajeIntegrador caso1 (2 ms)
  ✓ mostrarMensajeIntegrador caso2
  ✓  Verificar si el combo de Periodos disponibles funcion (202 ms)
  ✓ Verificar que permita guardar en la base de datos que el trabajo correspondiente ha sido aprobado (7 ms)
  ✓ should show an error message if idIntegrador is not greater than -1 (9 ms)

  console.log
    FormData {} integrador

      at AprobarProfesor.aprobarTrabajo (src/ModuloProfesor/AprobarProfesor.js:129:13)

  console.log
    aprobar profe undefined

      at AprobarProfesor.aprobarTrabajo (src/ModuloProfesor/AprobarProfesor.js:130:13)

A worker process has failed to exit gracefully and has been force exited. This is likely caused by tests leaki
Test Suites: 1 passed, 1 total
Tests:       5 passed, 5 total
Snapshots:   0 total
Time:        3.861 s, estimated 4 s
Ran all test suites matching /ModuloProfesor/i.

Active Filters: filename /ModuloProfesor/
 › Press c to clear filters.
```

Parte dos Modulo Profesor

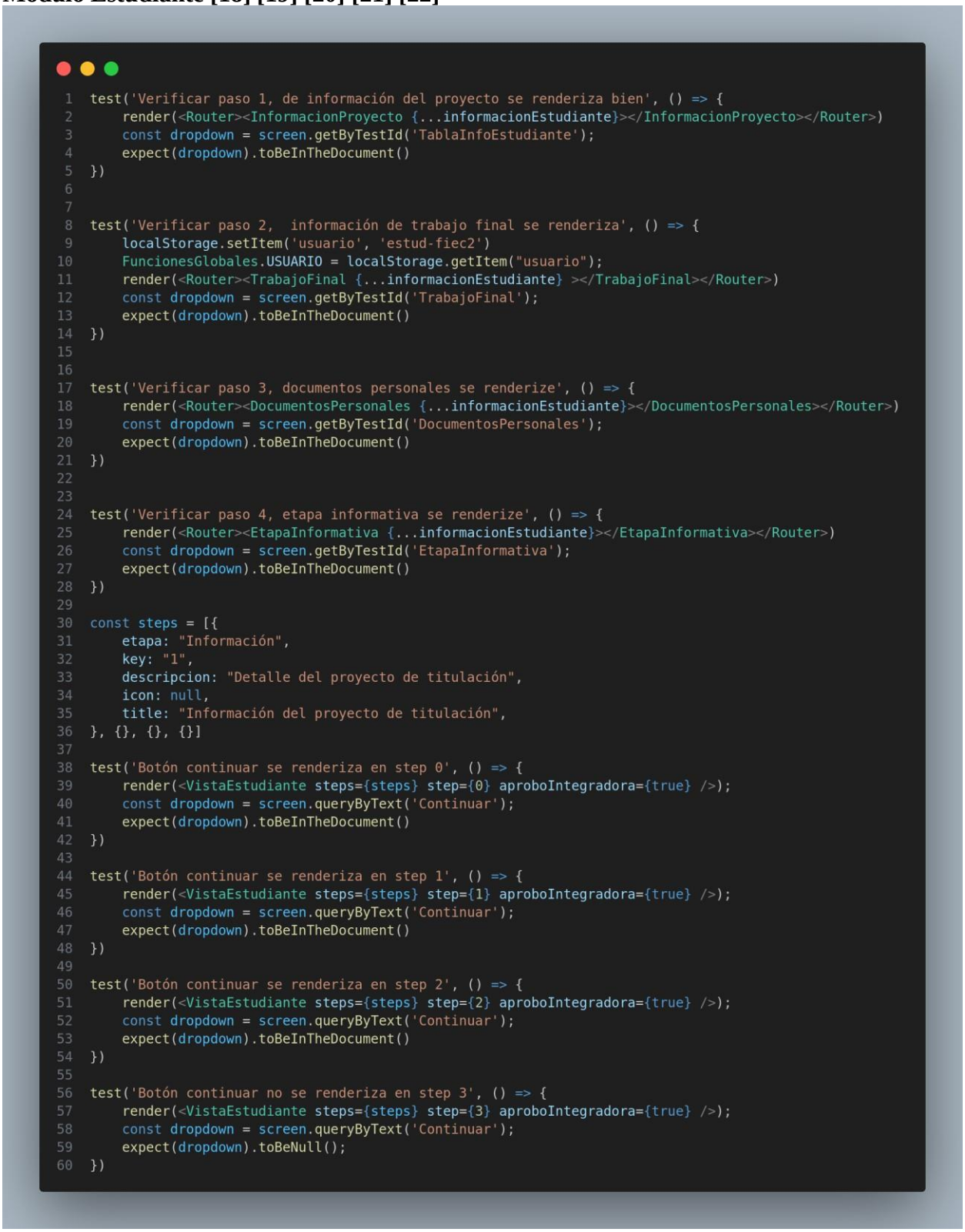


```
fterEach(() => {
  cleanup();
 });

test(' Verificar si el combo de Periodos disponibles funcion' , async()=> {

  let actualErrorMsg;
  render(<Router><FechaRecepcion periodosDisponibles= {PeriodosDisponibles}/></Router>)
  try {
      waitFor(() => {

          const dropdown = screen.queryByTestId('Option1')
          expect(dropdown).toHaveTextContent(/2S - 2021/)
        });
  } catch (err) {
        actualErrorMsg = err.message;
  }


})

let fileList;
let idIntegrador;
let usuario;
let idProyecto;

beforeEach(() => {
  fileList = [{ originFileObj: 'test-file' }];
  idIntegrador = 1;
  usuario = 'test-user';
  idProyecto = 'test-project-id';
});

afterEach(() => {
  cleanup();
});

it('Verificar que permita guardar en la base de datos que el trabajo correspondiente ha sido aprobado', async () => {
  aprobarProfesor.props = {
    idIntegrador,
    usuario,
    idProyecto,
    proyecto: { estudiantes: [{ usuario: 'est1', matricula: '123' }] },
  };
  aprobarProfesor.state = { fileList };
  const mockValidateFields = jest.fn(() => Promise.resolve());
  aprobarProfesor.integrador = { current: { validateFields: mockValidateFields } };
  const mockAprobarTrabajo = jest.spyOn(aprobarProfesor, 'aprobarTrabajo');

  await aprobarProfesor.aprobarIntegrador();

  expect(mockValidateFields).toHaveBeenCalled();
  expect(mockAprobarTrabajo).toHaveBeenCalledWith(expect.any(FormData), 'integrador');
});

it('should show an error message if idIntegrador is not greater than -1', () => {
  aprobarProfesor.props = {
    idIntegrador: -1,
  };
  const mockErrorMessage = jest.spyOn(message, 'error');

  aprobarProfesor.aprobarIntegrador();

  expect(mockErrorMessage).toHaveBeenCalledWith({
    content: 'Verifique que el archivo esté disponible para su aprobación',
    style: FuncionesAuxiliares.layoutError(),
    duration: 5,
  });
});

```

Modulo Estudiante [18] [19] [20] [21] [22]

Descripción	**Datos de entrada**	**Resultados esperados**
Verificar que pruebas unitarias realizadas anteriormente de módulos profesor, tutor y secretaría funcionen correctamente.	Ninguno	Que sea completadas exitosamente después de ser reajustadas.
El título del archivo coincide con el de la base de datos.	Periodo académico, estudiante, información del proyecto.	Que el campo de verificación se False y el campo de coincide titulo sea true.
El título del archivo no coincide con el de la base de datos.	Periodo académico, estudiante, información del proyecto.	Que el campo de verificación se False y el campo de error integrador sea true.
Se sube correctamente el archivo con palabras clave, ODS, y Resumen.	Resumen/Abstract, palabras clave, Ods.	Que se renderice correctamente el resumen, palabras claves y ods al cargar la información del proyecto.
Se renderiza correctamente la información del proyecto integrador.	Información del estudiante e información del proyecto.	Que se renderice correctamente al cargar la información del proyecto.

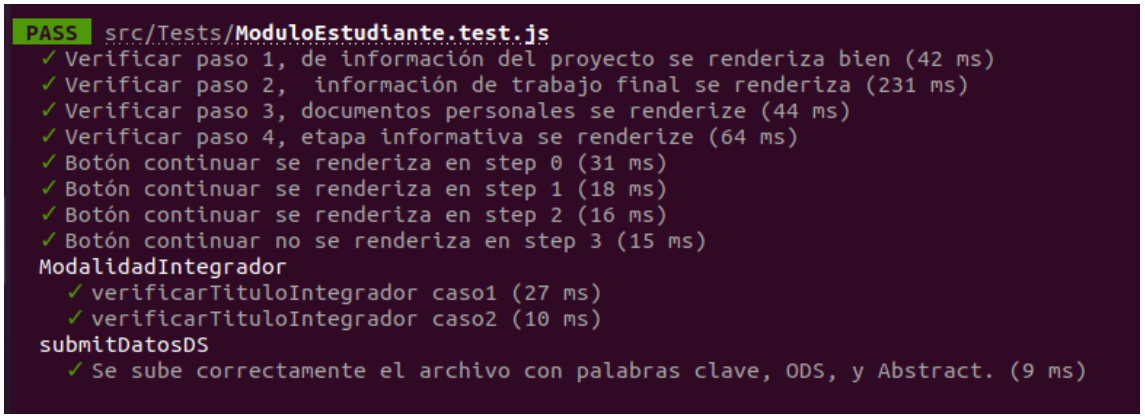
Modulo Estudiante [18] [19] [20] [21] [22]



```
test('Verificar paso 1, de información del proyecto se renderiza bien', () => {
    render(<Router><InformacionProyecto {...informacionEstudiante}></InformacionProyecto></Router>)
    const dropdown = screen.getByTestId('TablaInfoEstudiante');
    expect(dropdown).toBeInTheDocument()
})


test('Verificar paso 2,  información de trabajo final se renderiza', () => {
    localStorage.setItem('usuario', 'estud-fiec2')
    FuncionesGlobales.USUARIO = localStorage.getItem("usuario");
    render(<Router><TrabajoFinal {...informacionEstudiante} ></TrabajoFinal></Router>)
    const dropdown = screen.getByTestId('TrabajoFinal');
    expect(dropdown).toBeInTheDocument()
})


test('Verificar paso 3, documentos personales se renderize', () => {
    render(<Router><DocumentosPersonales {...informacionEstudiante}></DocumentosPersonales></Router>)
    const dropdown = screen.getByTestId('DocumentosPersonales');
    expect(dropdown).toBeInTheDocument()
})


test('Verificar paso 4, etapa informativa se renderize', () => {
    render(<Router><EtapaInformativa {...informacionEstudiante}></EtapaInformativa></Router>)
    const dropdown = screen.getByTestId('EtapaInformativa');
    expect(dropdown).toBeInTheDocument()
})

const steps = [{
    etapa: "Información",
    key: "1",
    descripcion: "Detalle del proyecto de titulación",
    icon: null,
    title: "Información del proyecto de titulación",
}, {}, {}, {}]

test('Botón continuar se renderiza en step 0', () => {
    render(<VistaEstudiante steps={steps} step={0} aproboIntegradora={true} />);
    const dropdown = screen.queryByText('Continuar');
    expect(dropdown).toBeInTheDocument()
})

test('Botón continuar se renderiza en step 1', () => {
    render(<VistaEstudiante steps={steps} step={1} aproboIntegradora={true} />);
    const dropdown = screen.queryByText('Continuar');
    expect(dropdown).toBeInTheDocument()
})

test('Botón continuar se renderiza en step 2', () => {
    render(<VistaEstudiante steps={steps} step={2} aproboIntegradora={true} />);
    const dropdown = screen.queryByText('Continuar');
    expect(dropdown).toBeInTheDocument()
})

test('Botón continuar no se renderiza en step 3', () => {
    render(<VistaEstudiante steps={steps} step={3} aproboIntegradora={true} />);
    const dropdown = screen.queryByText('Continuar');
    expect(dropdown).toBeNull();
})
```

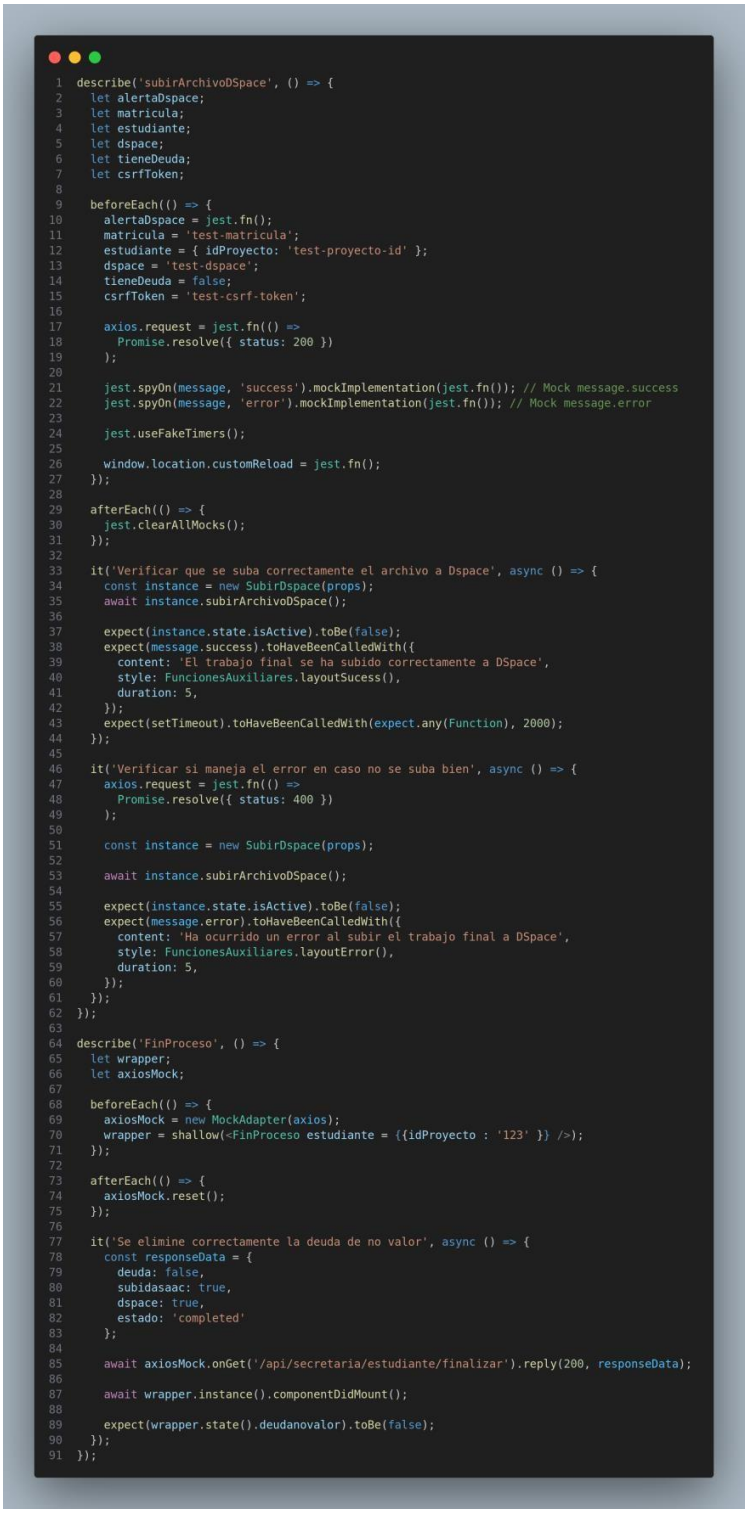
```
describe('ModalidadIntegrador', () => {
    test('verificarTituloIntegrador caso1', () => {
        const successData = {
            lastModified: 1690363986614,
            name: "TesisPrueba.pdf",
            size: 2504135,
            type: "application/pdf",
            uid: "rc-upload-1690837131279-9",
            webkitRelativePath: ""
        };

        const modalidadInt = new ModalidadIntegrador(data);

        const formData = new FormData();
        formData.append("file", successData);
        modalidadInt.verificarTituloIntegrador(formData, 'TC-300');

        expect(modalidadInt.state.verificandoIntegrador).toBe(false);
        expect(modalidadInt.state.coincideTitulo).toBe(true);
    });

    test('verificarTituloIntegrador caso2', () => {
        const failedData = {
            lastModified: 1689903542352,
            name: "Untitled (1).pdf",
            size: 40564,
            type: "application/pdf",
            uid: "rc-upload-1690933188428-3",
            webkitRelativePath: ""
        };

        const modalidadInt = new ModalidadIntegrador(data);
        const formData = new FormData();
        formData.append("file", failedData);

        modalidadInt.verificarTituloIntegrador(formData, 'TC-300');

        expect(modalidadInt.state.verificandoIntegrador).toBe(false);
        expect(modalidadInt.state.errorIntegrador).toBe(true);
    });


});

describe('submitDatosDS', () => {
  let axiosRequestSpy;

  beforeEach(() => {
    axiosRequestSpy = jest.spyOn(axios, 'request');
  });

  afterEach(() => {
    axiosRequestSpy.mockRestore();
  });

  it('Se sube correctamente el archivo con palabras clave, ODS, y Abstract.', () => {

    const modalidadInt = new ModalidadIntegrador(data);

    modalidadInt.state = {
      abstract: 'Sample abstract',
      chips: 'Sample chips',
      ods: 'Sample ods',
      csrfToken: 'SAMPLE_CSRF_TOKEN'
    };

    modalidadInt.submitDatosDS();

    expect(axiosRequestSpy).toHaveBeenCalledTimes(1);
    expect(axiosRequestSpy).toHaveBeenCalledWith(
      expect.objectContaining({
        method: 'POST',
        url: `${FuncionesGlobales.HOST}api/documento/proyecto/TC-300/AUTOMATIZACIÓN DE PRUEBAS UNITARIAS Y DE INTEGRACIÓN DEL SISTEMA AGATA`,
        data: expect.any(FormData),
        headers: {
          'X-CSRFToken': 'SAMPLE_CSRF_TOKEN'
        }
      })
    );
  });
});
```

Resultados test estudiante



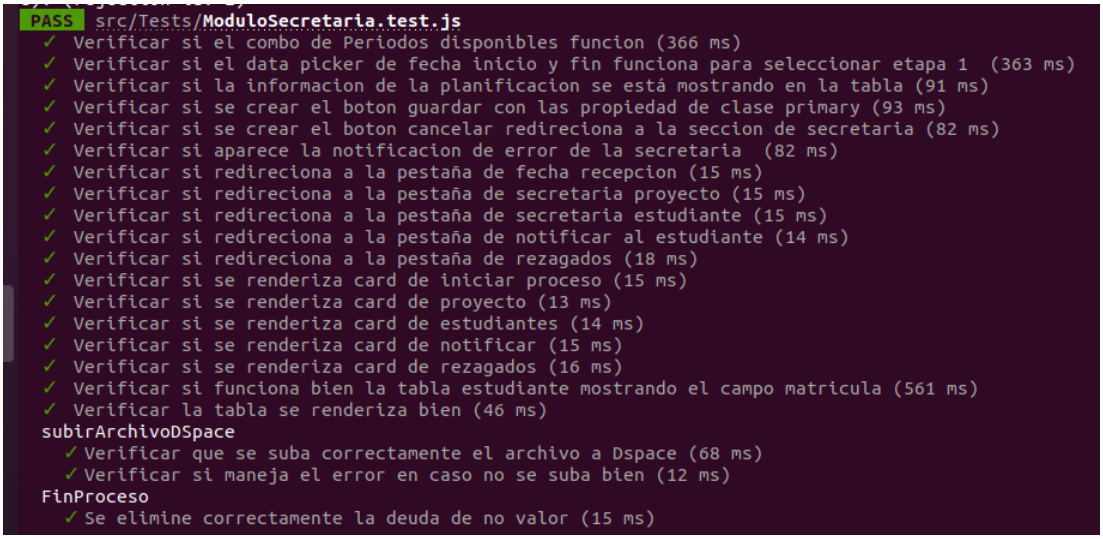
Modulo Secretaría [18]

Descripción	Datos de entrada	Resultados esperados
Verificar que pruebas unitarias realizadas anteriormente de módulos profesor, tutor y secretaría funcionen correctamente.	Ninguno	Que sea completadas exitosamente después de ser reajustadas.



```
describe('subirArchivoDSpace', () => {
  let alertaDspace;
  let matricula;
  let estudiante;
  let dspace;
  let tieneDeuda;
  let csrfToken;

  beforeEach(() => {
    alertaDspace = jest.fn();
    matricula = 'test-matricula';
    estudiante = { idProyecto: 'test-proyecto-id' };
    dspace = 'test-dspace';
    tieneDeuda = false;
    csrfToken = 'test-csrf-token';

    axios.request = jest.fn(() =>
      Promise.resolve({ status: 200 })
    );

    jest.spyOn(message, 'success').mockImplementation(jest.fn()); // Mock message.success
    jest.spyOn(message, 'error').mockImplementation(jest.fn()); // Mock message.error

    jest.useFakeTimers();

    window.location.customReload = jest.fn();
  });

  afterEach(() => {
    jest.clearAllMocks();
  });

  it('Verificar que se suba correctamente el archivo a Dspace', async () => {
    const instance = new SubirDspace(props);
    await instance.subirArchivoDSpace();

    expect(instance.state.isActive).toBe(false);
    expect(message.success).toHaveBeenCalledWith({
      content: 'El trabajo final se ha subido correctamente a DSpace',
      style: FuncionesAuxiliares.layoutSucess(),
      duration: 5,
    });
    expect(setTimeout).toHaveBeenCalledWith(expect.any(Function), 2000);
  });

  it('Verificar si maneja el error en caso no se suba bien', async () => {
    axios.request = jest.fn(() =>
      Promise.resolve({ status: 400 })
    );

    const instance = new SubirDspace(props);

    await instance.subirArchivoDSpace();

    expect(instance.state.isActive).toBe(false);
    expect(message.error).toHaveBeenCalledWith({
      content: 'Ha ocurrido un error al subir el trabajo final a DSpace',
      style: FuncionesAuxiliares.layoutError(),
      duration: 5,
    });
  });
});

describe('FinProceso', () => {
  let wrapper;
  let axiosMock;

  beforeEach(() => {
    axiosMock = new MockAdapter(axios);
    wrapper = shallow(<FinProceso estudiante = {{idProyecto : '123' }} />);
  });

  afterEach(() => {
    axiosMock.reset();
  });

  it('Se elimine correctamente la deuda de no valor', async () => {
    const responseData = {
      deuda: false,
      subidasaac: true,
      dspace: true,
      estado: 'completed'
    };

    await axiosMock.onGet('/api/secretaria/estudiante/finalizar').reply(200, responseData);

    await wrapper.instance().componentDidMount();

    expect(wrapper.state().deudanovalor).toBe(false);
  });
});
```

Resultados test secretaría

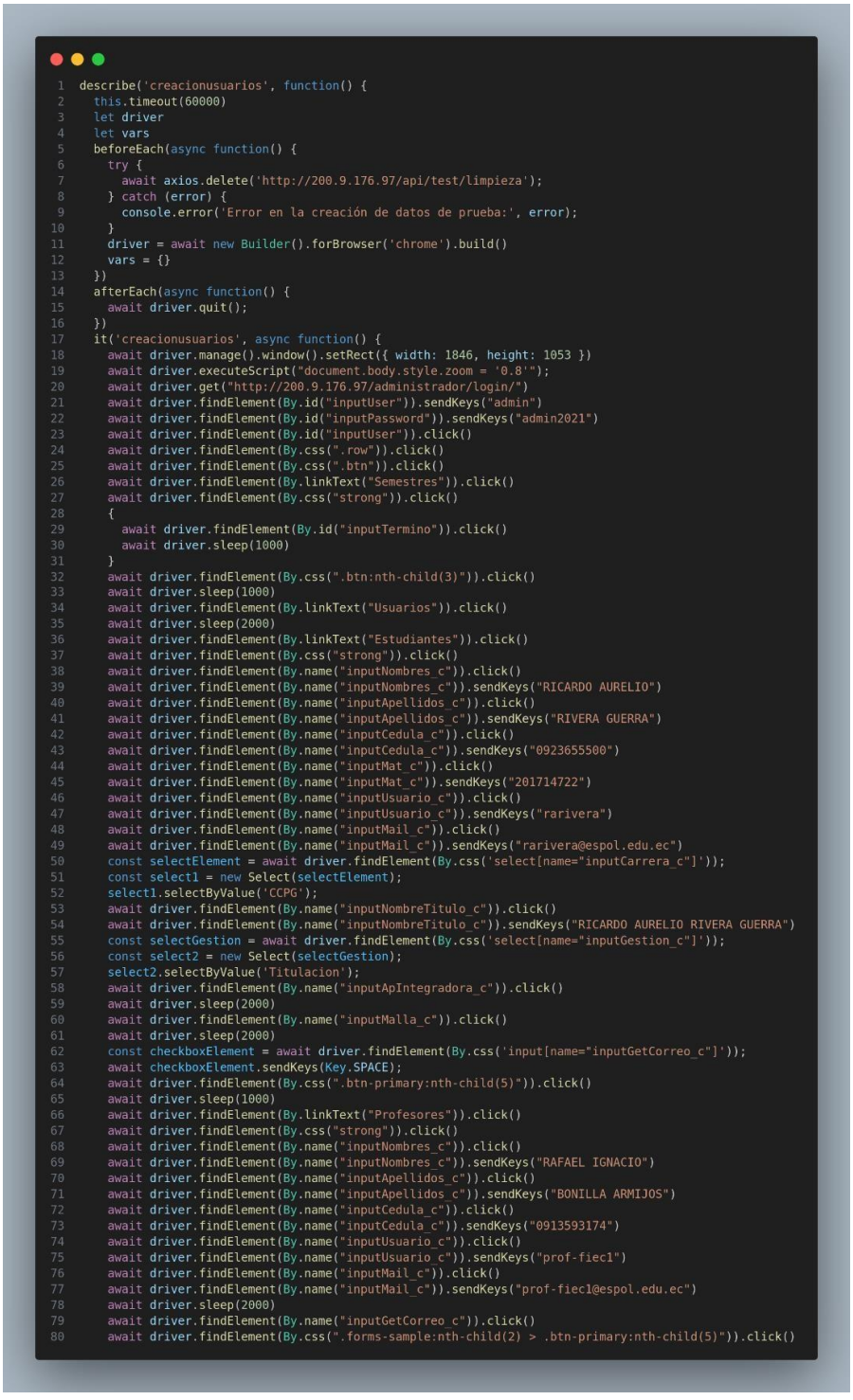


Apéndice B

Pruebas de Integración

Creación de Usuarios (Admin) Parte 1 del Código [1]

Descripción	Datos de entrada	Resultados esperados
Se crea el semestre, el proyecto, los usuarios estudiantes, secretaría, profesor, tutor y subdecanato desde el administrador	Ninguno	Usuarios creados satisfactoriamente



```
describe('creacionusuarios', function() {
  this.timeout(60000)
  let driver
  let vars
  beforeEach(async function() {
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    driver = await new Builder().forBrowser('chrome').build()
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('creacionusuarios', async function() {
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.executeScript("document.body.style.zoom = '0.8'");
    await driver.get("http://200.9.176.97/administrador/login/")
    await driver.findElement(By.id("inputUser")).sendKeys("admin")
    await driver.findElement(By.id("inputPassword")).sendKeys("admin2021")
    await driver.findElement(By.id("inputUser")).click()
    await driver.findElement(By.css(".row")).click()
    await driver.findElement(By.css(".btn")).click()
    await driver.findElement(By.linkText("Semestres")).click()
    await driver.findElement(By.css("strong")).click()
    {
      await driver.findElement(By.id("inputTermino")).click()
      await driver.sleep(1000)
    }
    await driver.findElement(By.css(".btn:nth-child(3)")).click()
    await driver.sleep(1000)
    await driver.findElement(By.linkText("Usuarios")).click()
    await driver.sleep(2000)
    await driver.findElement(By.linkText("Estudiantes")).click()
    await driver.findElement(By.css("strong")).click()
    await driver.findElement(By.name("inputNombres_c")).click()
    await driver.findElement(By.name("inputNombres_c")).sendKeys("RICARDO AURELIO")
    await driver.findElement(By.name("inputApellidos_c")).click()
    await driver.findElement(By.name("inputApellidos_c")).sendKeys("RIVERA GUERRA")
    await driver.findElement(By.name("inputCedula_c")).click()
    await driver.findElement(By.name("inputCedula_c")).sendKeys("0923655500")
    await driver.findElement(By.name("inputMat_c")).click()
    await driver.findElement(By.name("inputMat_c")).sendKeys("201714722")
    await driver.findElement(By.name("inputUsuario_c")).click()
    await driver.findElement(By.name("inputUsuario_c")).sendKeys("rarivera")
    await driver.findElement(By.name("inputMail_c")).click()
    await driver.findElement(By.name("inputMail_c")).sendKeys("rarivera@espol.edu.ec")
    const selectElement = await driver.findElement(By.css('select[name="inputCarrera_c"]'));
    const select1 = new Select(selectElement);
    select1.selectByValue('CCPG');
    await driver.findElement(By.name("inputNombreTitulo_c")).click()
    await driver.findElement(By.name("inputNombreTitulo_c")).sendKeys("RICARDO AURELIO RIVERA GUERRA")
    const selectGestion = await driver.findElement(By.css('select[name="inputGestion_c"]'));
    const select2 = new Select(selectGestion);
    select2.selectByValue('Titulacion');
    await driver.findElement(By.name("inputApIntegradora_c")).click()
    await driver.sleep(2000)
    await driver.findElement(By.name("inputMalla_c")).click()
    await driver.sleep(2000)
    const checkboxElement = await driver.findElement(By.css('input[name="inputGetCorreo_c"]'));
    await checkboxElement.sendKeys(Key.SPACE);
    await driver.findElement(By.css(".btn-primary:nth-child(5)")).click()
    await driver.sleep(1000)
    await driver.findElement(By.linkText("Profesores")).click()
    await driver.findElement(By.css("strong")).click()
    await driver.findElement(By.name("inputNombres_c")).click()
    await driver.findElement(By.name("inputNombres_c")).sendKeys("RAFAEL IGNACIO")
    await driver.findElement(By.name("inputApellidos_c")).click()
    await driver.findElement(By.name("inputApellidos_c")).sendKeys("BONILLA ARMIJOS")
    await driver.findElement(By.name("inputCedula_c")).click()
    await driver.findElement(By.name("inputCedula_c")).sendKeys("0913593174")
    await driver.findElement(By.name("inputUsuario_c")).click()
    await driver.findElement(By.name("inputUsuario_c")).sendKeys("prof-fiec1")
    await driver.findElement(By.name("inputMail_c")).click()
    await driver.findElement(By.name("inputMail_c")).sendKeys("prof-fiec1@espol.edu.ec")
    await driver.sleep(2000)
    await driver.findElement(By.name("inputGetCorreo_c")).click()
    await driver.findElement(By.css(".forms-sample:nth-child(2) > .btn-primary:nth-child(5)")).click()
```

Creación de Usuarios (Admin) Parte 2 del código [1]

```
  await driver.sleep(1000)
     await driver.findElement(By.linkText("Tutores")).click()
     await driver.findElement(By.css("strong")).click()
     await driver.findElement(By.name("inputNombres_c")).click()
     await driver.findElement(By.name("inputNombres_c")).sendKeys("EDUARDO SEGUNDO")
     await driver.findElement(By.name("inputApellidos_c")).click()
     await driver.findElement(By.name("inputApellidos_c")).sendKeys("CRUZ RAMIREZ")
     await driver.findElement(By.name("inputCedula_c")).click()
     await driver.findElement(By.name("inputCedula_c")).sendKeys("0910256081")
     await driver.findElement(By.name("inputUsuario_c")).click()
     await driver.findElement(By.name("inputUsuario_c")).sendKeys("tutor-fiec1")
     await driver.findElement(By.name("inputMail_c")).click()
     await driver.findElement(By.name("inputMail_c")).sendKeys("tutor-fiec1@espol.edu.ec")
     await driver.sleep(2000)
     await driver.findElement(By.name("inputGetCorreo_c")).click()
     await driver.findElement(By.css(".forms-sample:nth-child(2) > .btn-primary:nth-child(5)")).click()
     await driver.sleep(1000)
     await driver.findElement(By.linkText("Secretaria")).click()
     await driver.findElement(By.css("strong")).click()
     await driver.findElement(By.name("inputNombres_c")).click()
     await driver.findElement(By.name("inputNombres_c")).sendKeys("María Rocio")
     await driver.findElement(By.name("inputApellidos_c")).click()
     await driver.findElement(By.name("inputApellidos_c")).sendKeys("Perez Suarez")
     await driver.findElement(By.name("inputCedula_c")).click()
     await driver.findElement(By.name("inputCedula_c")).sendKeys("0910256871")
     await driver.findElement(By.name("inputUsuario_c")).click()
     await driver.findElement(By.name("inputUsuario_c")).sendKeys("secre-fiec1")
     await driver.findElement(By.name("inputMail_c")).click()
     await driver.findElement(By.name("inputMail_c")).sendKeys("secre-fiec1@espol.edu.ec")
     await driver.sleep(2000)
     await driver.findElement(By.name("inputGetCorreo_c")).click()
     await driver.findElement(By.css(".forms-sample:nth-child(2) > .btn-primary:nth-child(5)")).click()
     await driver.sleep(1000)
     await driver.findElement(By.linkText("Subdecanato")).click()
     await driver.findElement(By.css("strong")).click()
     await driver.findElement(By.name("inputNombres_c")).click()
     await driver.findElement(By.name("inputNombres_c")).sendKeys("Juan Carlos ")
     await driver.findElement(By.name("inputApellidos_c")).click()
     await driver.findElement(By.name("inputApellidos_c")).sendKeys("Duarte Criollo")
     await driver.findElement(By.name("inputCedula_c")).click()
     await driver.findElement(By.name("inputCedula_c")).sendKeys("0910277081")
     await driver.findElement(By.name("inputUsuario_c")).click()
     await driver.findElement(By.name("inputUsuario_c")).sendKeys("subde-fiec1")
     await driver.findElement(By.name("inputMail_c")).click()
     await driver.findElement(By.name("inputMail_c")).sendKeys("subde-fiec1@espol.edu.ec")
     await driver.sleep(2000)
     await driver.findElement(By.name("inputGetCorreo_c")).click()
     await driver.findElement(By.css(".forms-sample:nth-child(2) > .btn-primary:nth-child(5)")).click()
     await driver.sleep(2000)
     await driver.findElement(By.css(".nav-item:nth-child(3) .menu-title")).click()
     await driver.sleep(2000)
     await driver.findElement(By.css("strong")).click()
     await driver.findElement(By.name("inputTitulo_c")).click()
     await driver.findElement(By.name("inputTitulo_c")).sendKeys("Proyecto integrador de prueba")
     await driver.findElement(By.name("inputCodigo_c")).click()
     await driver.findElement(By.name("inputCodigo_c")).sendKeys("PG-001")
     const selectTipoProyecto = await driver.findElement(By.css('select[name="inputTipoProy_c"]'));
     const selectTipo = new Select(selectTipoProyecto);
     selectTipo.selectByValue('Integrador');
     const selectEstado = await driver.findElement(By.css('select[name="inputEstado_c"]'));
     const selectE = new Select(selectEstado);
     selectE.selectByValue('No iniciado');
     const selectEstudiante = await driver.findElement(By.css('select[name="inputEstudiantes_c"]'));
     const selectEstud = new Select(selectEstudiante);
     await selectEstud.selectByVisibleText('RICARDO AURELIO RIVERA GUERRA');
     const selectTutor = await driver.findElement(By.css('select[name="inputTutores_c"]'));
     const selectTut = new Select(selectTutor);
     await selectTut.selectByVisibleText('EDUARDO SEGUNDO CRUZ RAMIREZ');
     await driver.findElement(By.name("inputDateIni_c")).click()
     await driver.findElement(By.name("inputDateIni_c")).sendKeys("17/07/2023")
     await driver.findElement(By.name("inputCoinTitulo_c")).click()
     await driver.findElement(By.css(".btn-primary:nth-child(5)")).click()
     await driver.sleep(2000)
   })
})
```

```
creacionusuarios
  ✓ creacionusuarios (35353ms)


1 passing (37s)
```

Creación de período académico (Secretaría) [2]

Descripción	Datos de entrada	Resultados esperados
Crear las etapas del proceso.	Periodo académico, usuario secretaria	Proceso de titulación creado exitasomante

```
describe('crearPeriodoAcademico', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('crearPeriodoAcademico', async function() {
    await driver.sleep(2000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/fechas');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).sendKeys("secre-fiec1")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(5000);
    await driver.findElement(By.css('#idestudiante')).click()
    await driver.findElement(By.linkText("Inicio")).click()
    await driver.findElement(By.css(".anticon-calendar")).click()
    await driver.findElement(By.css(".ant-select-in-form-item > .ant-select-selector")).click()
    await driver.sleep(1000)
    await driver.findElement(By.css(".ant-select-item-option-active > .ant-select-item-option-content")).click()
    await driver.findElement(By.id("inicioProceso")).click()
    await driver.findElement(By.className("ant-picker-cell-today")).click();
    await driver.sleep(2000)
    await driver.findElement(By.id("inicioRecepcion")).click()
    await driver.findElement(By.css("div:nth-child(7) .ant-picker-next-icon")).click()
    await driver.findElement(By.css("div:nth-child(7) tr:nth-child(1) > .ant-picker-cell:nth-child(6) > .ant-picker-cell-inner:nth-child(1)")).click()
    await driver.sleep(2000)
    await driver.findElement(By.id("finRecepcion")).click()
    await driver.findElement(By.css("div:nth-child(8) .ant-picker-header-next-btn")).click()
    await driver.findElement(By.css("div:nth-child(8) tr:nth-child(4) > .ant-picker-cell:nth-child(6) > .ant-picker-cell-inner:nth-child(1)")).click()
    await driver.sleep(2000)
    await driver.findElement(By.css(".ant-btn-primary > span")).click()
    await driver.sleep(1000)
    await driver.findElement(By.css(".ant-timeline-item:nth-child(1) > .ant-timeline-item-content")).click()
    await driver.sleep(4000)
    await driver.findElement(By.css(".ant-btn:nth-child(2) > span")).click()
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);

    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "El proceso de titulación se creó con éxito.";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
  })
})
```

```
crearPeriodoAcademico
  ✓ crearPeriodoAcademico (33362ms)


1 passing (36s)
```

Subir trabajo final cuando el título no coincide (Estudiante) [4]

Descripción	Datos de entrada	Resultados esperados
Se sube la información de su proyecto integrador (Palabras clave, ODS, Resumen, Archivo Final y Entregables) y se verifica que el título del archivo coincida con el del sistema.	Periodo académico, usuarios secretaria, estudiante, y proyecto creado en el admin.	Archivo coincide con el título registrado en el sistema. Archivo final enviado correctamente.

```
describe('TituloNoCoincide', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('TituloNoCoincide', async function() {
    await driver.sleep(3000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/revisionGrupal');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000);
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).sendKeys("rarivera")
    await driver.findElement(By.id("password")).sendKeys("Zki250501..")
    await driver.sleep(2000); // Pausa de 3 segundos
    await driver.findElement(By.id("password")).sendKeys("Zki250501..")
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(5000); // Pausa de 3 segundos
    let estudiante = await driver.findElement(By.id("idestudiante"));
    await estudiante.click();
    let inicio = await driver.findElement(By.id("inicio"));
    await inicio.click();
    await driver.sleep(6000); // Pausa de 3 segundos
    let continuar = await driver.findElement(By.id("continuar"));
    await continuar.click();
    await driver.sleep(3000); // Pausa de 5 segundos

    let fileInput = await driver.findElement(By.id("upload"));
    await fileInput.sendKeys('/home/oem/Descargas/Untitled.pdf');
    let alertElement = await driver.wait(until.elementLocated(By.css("div.ant-alert-message")), 10000);
    let alertMessage = await alertElement.getText();
    let expectedMessage = "El título de su archivo de trabajo final no coincide con el título registrado en el sistema de proyectos.";
    assert.strictEqual(alertMessage, expectedMessage, "El mensaje no coincide con el esperado");
    await driver.sleep(3000); // Pausa de 5 segundos

    await driver.findElement(By.css("input[type=\"text\"]")).sendKeys("palabras", Key.TAB);
    await driver.findElement(By.css("input[type=\"text\"]")).sendKeys("claves", Key.TAB);
    await driver.sleep(1000);
    await driver.switchTo().frame(0);
    await driver.findElement(By.css("p")).click();
    {
      const element= await driver.findElement(By.css("#tinymce"));
      await driver.executeScript("if(arguments[0].contentEditable==='true'){arguments[0].innerText='Abstract de prueba'}", element);
    }
    await driver.switchTo().defaultContent();
    await driver.findElement(By.id("opciones")).click()
    {
      const dropdown = await driver.findElement(By.id("opciones"))
      await dropdown.findElement(By.xpath("//option[. = 'Trabajo decente y crecimiento económico']")).click();
    }

    await driver.sleep(2000);
    let fileInput2 = await driver.findElement(By.id("upload-ent"));
    await fileInput2.sendKeys('/home/oem/Descargas/tests.txt');
    await driver.sleep(3000);

    await driver.findElement(By.css("button[type=\"submit\"]")).click()
    await driver.sleep(1000)
    await driver.findElement(By.id("opcion-aceptar")).click();
    await driver.sleep(3000)
    await driver.findElement(By.id("btn-continuar")).click();
    await driver.sleep(7000)
  })
})
```

Subir trabajo final cuando el título no coincide (Estudiante) [4]

Revisión Tutor (Tutor) [5]

Descripción	Datos de entrada	Resultados esperados
Revisar y aprobar el archivo que subió el estudiante.	Proyecto estudiante en estado entregado	Proyecto integrador aprobado exitosamente.

```
describe('RevisionTutor', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('RevisionTutor', async function() {
    await driver.sleep(2000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/revisionTutor');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.sleep(3000)
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).sendKeys("tutor-fiec1")
    await driver.findElement(By.id("password")).sendKeys("jdhfl215", Key.RETURN)
    await driver.sleep(2000)
    await driver.findElement(By.id("password")).sendKeys("jdhfl215", Key.RETURN)
    await driver.sleep(3000)
    await driver.findElement(By.css("#notification > .ant-badge path")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn > span")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-list-item:nth-child(1) .ant-list-item-meta-title")).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("Tutor")).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("Inicio")).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("PROYECTO INTEGRADOR DE PRUEBA")).click()
    await driver.sleep(4000)
    await driver.executeScript("window.scrollTo(0,451.81817626953125)")
    await driver.findElement(By.css("button[type=\"submit\"]")).click()
    await driver.sleep(6000)
    await driver.findElement(By.css(".ant-btn > span:nth-child(2)")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css("input#contenido")).click()
    await driver.findElement(By.css(".ant-btn-primary")).click()
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);
    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Proyecto integrador aprobado exitosamente";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
    await driver.sleep(4000)
  })
})
```

```
RevisionTutor
  ✓ RevisionTutor (51560ms)


1 passing (54s)
```

Revisión profesor (Profesor) [6]

Descripción	Datos de entrada	Resultados esperados
Revisar y aprobar el archivo que subió el estudiante.	Proyecto estudiante en estado entregado y aprobado por tutor	Proyecto integrador aprobado exitosamente.

```
describe('RevisionProfesor', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('RevisionProfesor', async function() {
    await driver.sleep(2000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/revisionProfesor');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.sleep(4000)
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).sendKeys("prof-fiec1")
    await driver.findElement(By.id("password")).sendKeys("jdhfl2155", Key.RETURN)
    await driver.sleep(2000)
    await driver.findElement(By.id("password")).sendKeys("jdhfl2155", Key.RETURN)
    await driver.sleep(4000)
    await driver.findElement(By.css("#notification > .ant-badge path")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn > span")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-list-item:nth-child(1) .ant-list-item-meta-title")).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("Profesor")).click()
    await driver.sleep(500)
    await driver.findElement(By.linkText("Inicio")).click()
    await driver.sleep(1000)
    await driver.findElement(By.linkText("PROYECTO INTEGRADOR DE PRUEBA")).click()
    await driver.sleep(2000)
    await driver.executeScript("window.scrollTo(0,451.81817626953125)")
    await driver.sleep(2000)
    await driver.findElement(By.css(".ant-btn-primary > span")).click()
    await driver.sleep(6000)
    await driver.findElement(By.css(".ant-btn > span:nth-child(2)")).click()
    await driver.sleep(2000)
    await driver.findElement(By.css("input[type=\"file\"]")).sendKeys('/home/oem/Descargas/Documento6.pdf')
    await driver.sleep(4000)
    await driver.executeScript("window.scrollTo(0,451.81817626953125)")
    await driver.findElement(By.css("input#contenido")).click()
    await driver.sleep(1000)
    await driver.findElement(By.css("input#formato")).click()
    await driver.sleep(1000)
    await driver.findElement(By.css("input#titulo")).click()
    await driver.sleep(1000)
    await driver.findElement(By.css("input#plagio")).click()
    await driver.sleep(1000)
    await driver.findElement(By.css(".ant-btn-primary")).click()
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);

    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Proyecto integrador aprobado exitosamente";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
    await driver.sleep(4000)
  })
})
```

```
RevisionProfesor
  ✓ RevisionProfesor (55467ms)


1 passing (58s)
```

Revisión secretaría grupal (Secretaría) [7]

Descripción	Datos de entrada	Resultados esperados
Revisar y aprobar el archivo que subió el estudiante.	Proyecto estudiante en estado entregado y aprobado por tutor y profesor	Proyecto integrador aprobado exitosamente.

```
describe('revisionSecretariaGrupal', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('revisionSecretariaGrupal', async function() {
    await driver.sleep(3000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/revisionSecretaria');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).sendKeys("secre-fiec1")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(3500)
    await driver.findElement(By.css("#notification > .ant-badge path")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn > span")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-list-item:nth-child(1) .ant-list-item-meta-title")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css('#idestudiante')).click()
    await driver.findElement(By.linkText("Proyectos")).click()
    await driver.sleep(2000)
    await driver.findElement(By.linkText("PROYECTO INTEGRADOR DE PRUEBA")).click()
    await driver.sleep(4000)
    await driver.findElement(By.css(".ant-btn-primary")).click()
    await driver.sleep(4000)
    await driver.findElement(By.css("#integrador")).click()
    await driver.findElement(By.css(".ant-btn-primary")).click()
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);
    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Proyecto integrador aprobado exitosamente";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
    await driver.sleep(4000)
  })
})
```

```
revisionSecretariaGrupal
  ✓ revisionSecretariaGrupal (43584ms)


1 passing (46s)
```

Devolver trabajo (Tutor) [8]

Descripción	Datos de entrada	Resultados esperados
Revisar archivo del estudiante y devolverlo con un comentario.	Proyecto estudiante en estado entregado	Trabajo devuelto exitosamente.

```javascript
describe('DevolverTrabajoTutor', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('DevolverTrabajoTutor', async function() {
    await driver.sleep(3000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/revisionTutor');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.sleep(3000)
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).sendKeys("tutor-fiec1")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(5000);
    await driver.findElement(By.id("idestudiante")).click()
    await driver.sleep(3000)
    await driver.findElement(By.id("inicio")).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("PROYECTO INTEGRADOR DE PRUEBA")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn-primary")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-collapse-header-text")).click()
    await driver.sleep(3000)
    await driver.findElement(By.id("comentario")).click()
    await driver.findElement(By.id("comentario")).sendKeys("Debe corregir el trabajo.")
    await driver.sleep(4000)
    let fileInput = await driver.findElement(By.id("upload"));
    await fileInput.sendKeys('/home/oem/Descargas/Untitled.pdf');
    await driver.sleep(5000)
    await driver.findElement(By.css(".ant-btn-danger")).click()
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);

    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Trabajo devuelto exitosamente";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
    await driver.sleep(4000)
  })
})
```

```
DevolverTrabajoTutor
  ✓ DevolverTrabajoTutor (54853ms)


1 passing (57s)
```

Devolver trabajo (Profesor) [9]

Descripción	Datos de entrada	Resultados esperados
Revisar archivo del estudiante y devolverlo con un comentario.	Proyecto estudiante en estado entregado y aprobado por tutor	Trabajo devuelto exitosamente.

```
describe('DevolverTrabajoProfesor', function() {
  this.timeout(60000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('Untitled', async function() {
    await driver.sleep(3000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/revisionProfesor');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.sleep(3000)
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).sendKeys("prof-fiec1")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl2155")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl2155")
    await driver.sleep(2000);
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(5000);
    await driver.findElement(By.id("idestudiante")).click()
    await driver.sleep(4000)
    await driver.findElement(By.id("inicio")).click()
    await driver.sleep(4000)
    await driver.findElement(By.linkText("PROYECTO INTEGRADOR DE PRUEBA")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn-primary")).click()
    await driver.sleep(4000)
    await driver.findElement(By.css(".ant-collapse-header-text")).click()
    await driver.sleep(3000)
    await driver.findElement(By.id("comentario")).click()
    await driver.findElement(By.id("comentario")).sendKeys("Debe corregir el trabajo.")
    await driver.sleep(4000)
    let fileInput = await driver.findElement(By.id("upload"));
    await fileInput.sendKeys('/home/oem/Descargas/Untitled.pdf');
    await driver.sleep(4000)
    await driver.findElement(By.css(".ant-btn-danger")).click()
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);

    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Trabajo devuelto exitosamente";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
    await driver.sleep(4000)
  })
})
```

```
DevolverTrabajoProfesor
  ✓ Untitled (54991ms)


1 passing (58s)
```

Devolver Trabajo (Secretaría)[10]

Descripción	Datos de entrada	Resultados esperados
Revisar archivo del estudiante y devolverlo con un comentario.	Proyecto estudiante en estado entregado y aprobado por tutor y profesor	Trabajo devuelto exitosamente.

```
describe('devolverTrabajoSecretaria', function() {
  this.timeout(60000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('devolverTrabajoSecretaria', async function() {
    await driver.sleep(3000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/revisionSecretaria');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).sendKeys("secre-fiec1")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(5000);
    await driver.findElement(By.css('#idestudiante')).click()
    await driver.findElement(By.linkText("Proyectos")).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("PROYECTO INTEGRADOR DE PRUEBA")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn-primary")).click()
    await driver.sleep(4000)
    await driver.findElement(By.css(".ant-collapse-header-text")).click()
    await driver.sleep(3000)
    await driver.findElement(By.id("comentario")).click()
    await driver.findElement(By.id("comentario")).sendKeys("Debe corregir el trabajo.")
    await driver.sleep(3000)
    let fileInput = await driver.findElement(By.id("upload"));
    await fileInput.sendKeys('/home/oem/Descargas/Untitled.pdf');
    await driver.sleep(4000)
    await driver.findElement(By.css(".ant-btn-danger")).click()
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);

    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Trabajo devuelto exitosamente";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
    await driver.sleep(4000)
  })
})
```

```
devolverTrabajoSecretaria
  ✓ devolverTrabajoSecretaria (44482ms)


1 passing (47s)
```

Notificación a estudiante (Secretaría) [11]

Descripción	Datos de entrada	Resultados esperados
Notificar al estudiante por correo electrónico recordatorios de fechas del proceso.	Proyecto del estudiante subido.	Notificación enviada exitosamente.

```
describe('notificarEstudiante', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('notificarEstudiante', async function() {
    await driver.sleep(2000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/revisionSecretaria');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.sleep(4000)
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).click()
    await driver.findElement(By.id("username")).sendKeys("secre-fiec1")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.id("idestudiante")).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("Notificar")).click()
    await driver.sleep(3000)
    await driver.executeScript("window.scrollTo(0,31.111112594604492)")
    await driver.findElement(By.css(".ant-table-cell > .ant-checkbox-wrapper .ant-checkbox-input")).click()
    const campoDeEntrada = await driver.findElement(By.id('asunto'));
    await campoDeEntrada.sendKeys("Revisión");
    await driver.sleep(3000)
    const campoEditable = await driver.findElement(By.css('.jodit-wysiwyg'));
    await campoEditable.sendKeys("Esta es una notificación de prueba");
    await driver.sleep(3000)
    await driver.findElement(By.id("testenviar")).click()
    await driver.sleep(4000)
    await driver.findElement(By.css(".ant-btn:nth-child(2) > span")).click()
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);

    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Notificación enviada";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
    await driver.sleep(4000)
  })
})
```

```
notificarEstudiante
  ✓ notificarEstudiante (40851ms)
```

Subir documentos personales (Estudiante) [12]

Descripción	Datos de entrada	Resultados esperados
Subir nombres completos y hoja de vida.	Proyecto aprobado por tutor, profesor y secretaria.	Información subida exitosamente.

```
describe('faseIndividualEstudiante', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('faseIndividualEstudiante', async function() {
    await driver.sleep(2000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/subirHV');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).sendKeys("rarivera")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("Zki250501..")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("Zki250501..")
    await driver.sleep(2000);
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(5000);
    await driver.findElement(By.css("#idestudiante")).click()
    await driver.findElement(By.css("#inicio")).click()
    await driver.sleep(2000)
    await driver.findElement(By.css("#continuar > span")).click()
    await driver.sleep(2000)
    await driver.findElement(By.css("#continuar > span")).click()
    await driver.sleep(2000)
    await driver.findElement(By.css("#nombresCompletos")).sendKeys("Ricardo Aurelio Rivera Guerra")
    await driver.findElement(By.css("input[type=\"file\"]")).sendKeys('/home/oem/Descargas/201714722_hoja_vida.pdf')
    await driver.findElement(By.css("button[type=\"submit\"]")).click()
    await driver.sleep(200)
    await driver.findElement(By.css("div > .ant-modal-footer > .ant-btn-primary")).click()
    await driver.sleep(200)
    await driver.findElement(By.css(".ant-modal-confirm-btns > .ant-btn-primary")).click()
    await driver.sleep(8000)
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-success")), 10000);
    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Información enviada exitosamente";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
  })
})
```

```
faseIndividualEstudiante
  ✓ faseIndividualEstudiante (35464ms)


1 passing (39s)
```

Aprobar documentos personales (Secretaría) [13]

Descripción	Datos de entrada	Resultados esperados
Revisar y aprobar nombres completos y hoja de vida del estudiante	Documentos personales subidos por el estudiante.	Hoja de vida aprobada exitosamente.

```
describe('apDocsPersonales', function() {
  this.timeout(100000)
  let driver
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }

  })
  afterEach(async function() {
    await driver.quit();
  })
  it('apDocsPersonales', async function() {
    await driver.sleep(2000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/revisionIndividual');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.sleep(4000)
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).click()
    await driver.findElement(By.id("username")).sendKeys("secre-fiec1")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css('#idestudiante')).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("Estudiantes")).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("PROYECTO INTEGRADOR DE PRUEBA")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn > span")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn > span")).click()
    await driver.sleep(3000)
    await driver.findElement(By.id("cantidad")).click()
    await driver.sleep(3000)
    await driver.findElement(By.id("contenido")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn-primary:nth-child(1) > span")).click()
    await driver.sleep(3000);
  })
})
```

```
apDocsPersonales
  ✓ apDocsPersonales (41933ms)


1 passing (45s)
```

Extraer Acta (Secretaría) [14]

Descripción	Datos de entrada	Resultados esperados
Se extrae el acta de evaluación del estudiante del api del GTSI	Documentos personales subidos por el estudiante y aprobados por la secretaria.	Acta descargada.

```
describe('extraerACta', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(2000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('extraerActa', async function() {
    await driver.sleep(2000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/extraerActa');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(2000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.sleep(4000)
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).click()
    await driver.findElement(By.id("username")).sendKeys("secre-fiec1")
    await driver.findElement(By.id("password")).click()
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000)
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(3000)
    await driver.findElement(By.id("idestudiante")).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("Estudiantes")).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("PROYECTO INTEGRADOR DE PRUEBA")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn-primary > span")).click()
    await driver.sleep(3000)
    await driver.findElement(By.id("btn-acta")).click()
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);

    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Acta extraída exitosamente";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
  })
})
```

```
extraerACta
  ✓ extraerActa (58024ms)


1 passing (1m)
```

Firmar acta (subdecano) [15]

Descripción	Datos de entrada	Resultados esperados
Descargar acta del estudiante y proceder a firmarlas en el sistema.	Documentos personales subidos por el estudiante y aprobados por la secretaria, extracción de acta.	Actas firmadas.

```
describe('firmaracta', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(3000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('firmaracta', async function() {
    await driver.sleep(3000)
    try {
        await axios.post('http://200.9.176.97/api/test/datos/firmarActa');
      } catch (error) {
        console.error('Error en la creación de datos de prueba:', error);
      }
    await driver.sleep(3000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.sleep(4000)
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).click()
    await driver.findElement(By.id("username")).sendKeys("subde-fiec1")
    await driver.findElement(By.id("password")).click()
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(3000)
    await driver.findElement(By.id("idestudiante")).click()
    await driver.findElement(By.linkText("Firmar actas pendientes")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-table-cell > .ant-checkbox-wrapper .ant-checkbox-input")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn > span")).click()
    await driver.sleep(3000)
    let fileInput = await driver.findElement(By.id("upload"));
    await fileInput.sendKeys('/home/oem/Descargas/archivo.p12');
    await driver.sleep(3000)
    const inputElement = driver.findElement(By.id('register_password'));
    const textoAEscribir = '2505';
    inputElement.sendKeys(textoAEscribir);
    await driver.sleep(3000)
    await driver.findElement(By.id("firmar")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn-primary:nth-child(2)")).click()
    await driver.wait(until.elementLocated(By.css('[data-testid="modal-resumen"]')), 5000);
    await driver.wait(until.elementIsVisible(driver.findElement(By.css('[data-testid="modal-resumen"]'))), 5000);
    await driver.sleep(3000)
    await driver.findElement(By.css('[data-testid="modal-resumen"] .ant-modal-close')).click();
    await driver.sleep(5000)
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);
    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Actas firmadas exitosamente";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
    await driver.findElement(By.css(".navbar-elem:nth-child(1) > .anticon")).click()
    await driver.sleep(4000)
    await driver.get("http://localhost:3000/inicio")
    await driver.sleep(4000)
    await driver.findElement(By.css("span")).click()
    await driver.sleep(4000)
    await driver.findElement(By.id("username")).click()
    await driver.findElement(By.id("username")).sendKeys("secre-fiec1")
    await driver.findElement(By.id("password")).click()
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(3000)
    await driver.findElement(By.id("idestudiante")).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("Actas firmadas")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn-default")).click()
    await driver.sleep(4000)
  })
})
```

```
firmaracta
  ✓ firmaracta (77120ms)


1 passing (1m)
```

Subir proyecto al Dspace (Secretaría)[16]

Descripción	Datos de entrada	Resultados esperados
Subir trabajo final al Dspace	Documentos personales subidos por el estudiante y aprobados por la secretaria, extracción de acta.	Trabajo final subido al Dspace.

```
describe('subirDSpace', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(3000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('subirDSpace', async function() {
    await driver.sleep(3000)
    try {
      await axios.post('http://200.9.176.97/api/test/datos/subirDspace');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(3000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).click()
    await driver.findElement(By.id("username")).sendKeys("secre-fiec1")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.id("password")).sendKeys("jdhfl215")
    await driver.sleep(2000);
    await driver.findElement(By.name("submit")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css('#idestudiante')).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("Estudiantes")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-table-row > .ant-table-cell:nth-child(3)")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn:nth-child(2) > span")).click()
    await driver.sleep(3000)
    await driver.findElement(By.id("btn-dspace")).click()
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);
    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Información súbida al dspace";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
  })
})
```

```
subirDSpace
  ✓ subirDSpace (82709ms)


1 passing (1m)
```

Subir acta firmada (Secretaría) [17]

Descripción	Datos de entrada	Resultados esperados
Subir acta al SAAC con hoja agregada con los nombres del estudiante.	Documentos personales subidos por el estudiante y aprobados por la secretaria, extracción de acta, trabajo subido al Dspace y deuda no valor eliminada	Archivo guardado exitosamente.

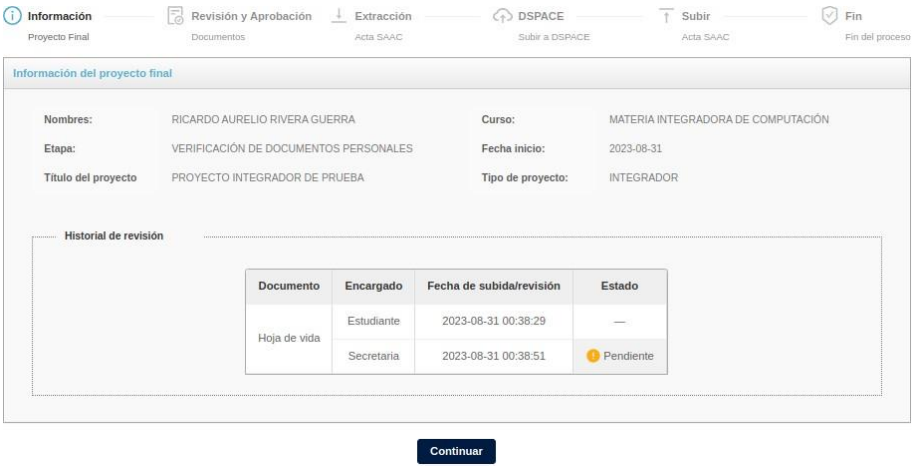
```javascript
describe('subirActaFirmada', function() {
  this.timeout(100000)
  let driver
  let vars
  beforeEach(async function() {
    driver = await new Builder().forBrowser('chrome').build()
    await driver.sleep(3000)
    try {
      await axios.delete('http://200.9.176.97/api/test/limpieza');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    vars = {}
  })
  afterEach(async function() {
    await driver.quit();
  })
  it('subirActaFirmada', async function() {
    await driver.sleep(2000)
    // 1 | open | http://localhost:3000/inicio |
    try {
      await axios.post('http://200.9.176.97/api/test/datos/subirActa');
    } catch (error) {
      console.error('Error en la creación de datos de prueba:', error);
    }
    await driver.sleep(3000)
    await driver.get("http://200.9.176.116/inicio")
    await driver.manage().window().setRect({ width: 1846, height: 1053 })
    await driver.findElement(By.css("span")).click()
    await driver.findElement(By.id("username")).click()
     await driver.findElement(By.id("username")).sendKeys("secre-fiec1")
     await driver.sleep(2000);
     await driver.findElement(By.id("password")).sendKeys("jdhfl215")
     await driver.sleep(2000);
     await driver.findElement(By.id("password")).sendKeys("jdhfl215")
     await driver.sleep(2000);
     await driver.findElement(By.name("submit")).click()
     await driver.sleep(3000)
    await driver.findElement(By.css('#idestudiante')).click()
    await driver.sleep(3000)
    await driver.findElement(By.linkText("Estudiantes")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-table-row > .ant-table-cell:nth-child(3)")).click()
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn:nth-child(2) > span")).click()
    await driver.sleep(3000)
    let fileInput = await driver.findElement(By.id("upload"));
    await fileInput.sendKeys('/home/oem/Descargas/201714722_actaFirmada.pdf');
    await driver.sleep(3000)
    await driver.findElement(By.css(".ant-btn-primary")).click()
    await driver.sleep(3000)
    let messageElement = await driver.wait(until.elementLocated(By.css("div.ant-message-custom-content")), 10000);
    let messageTextElement = await messageElement.findElement(By.css("span:nth-child(2)"));
    let messageText = await messageTextElement.getText();
    let expectedMessage = "Acta subida exitosamente";
    assert.strictEqual(messageText, expectedMessage, "El mensaje no coincide con el esperado");
  })
})
```

```
subirActaFirmada
  ✓ subirActaFirmada (37741ms)


1 passing (42s)
```

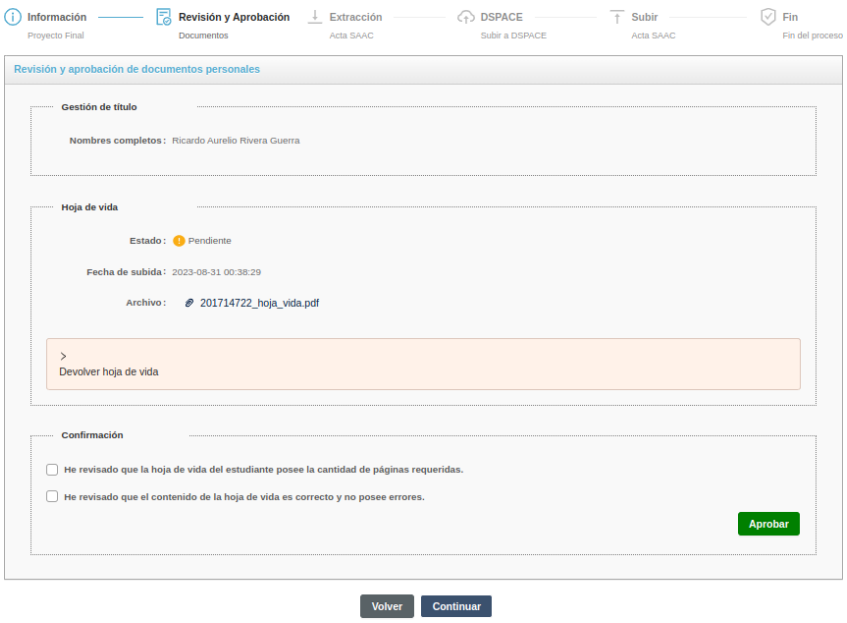
Apéndice C

Evidencia de correcciones

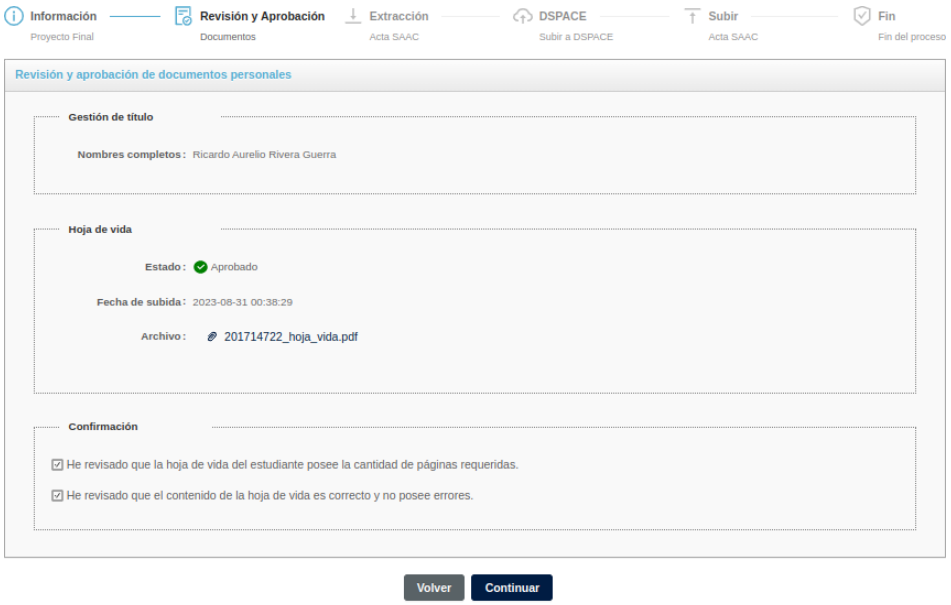
Flujo revisión individual parte 1 (Secretaría)



Flujo revisión individual parte 2 (Secretaría)



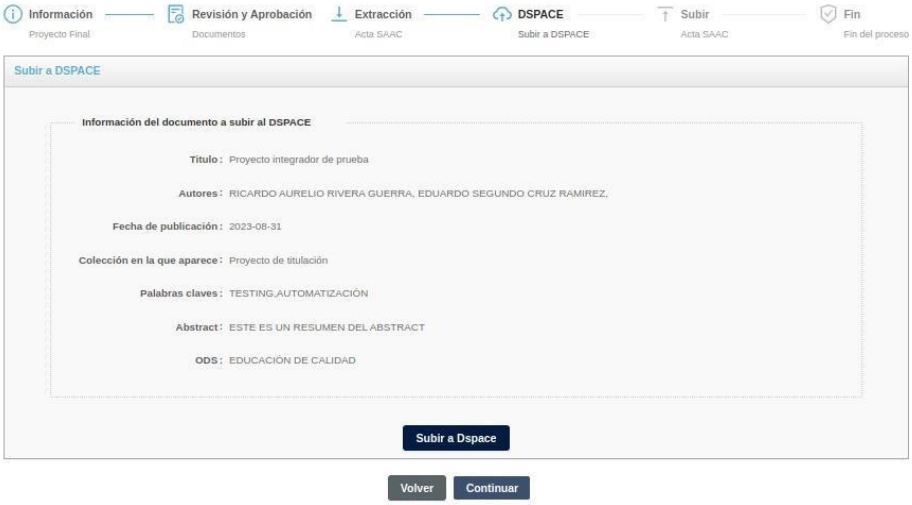
Flujo revisión individual parte 3 (Secretaría)



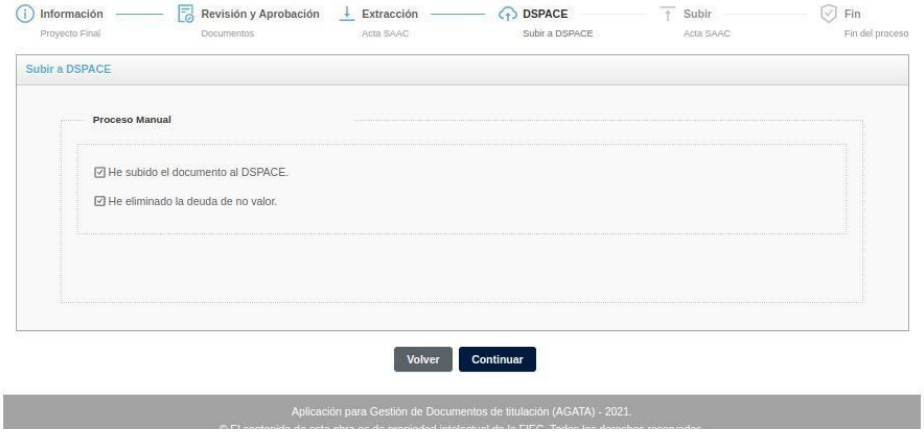
Flujo revisión individual parte 4 (Secretaría)

Flujo revisión individual parte 5 (Secretaría)

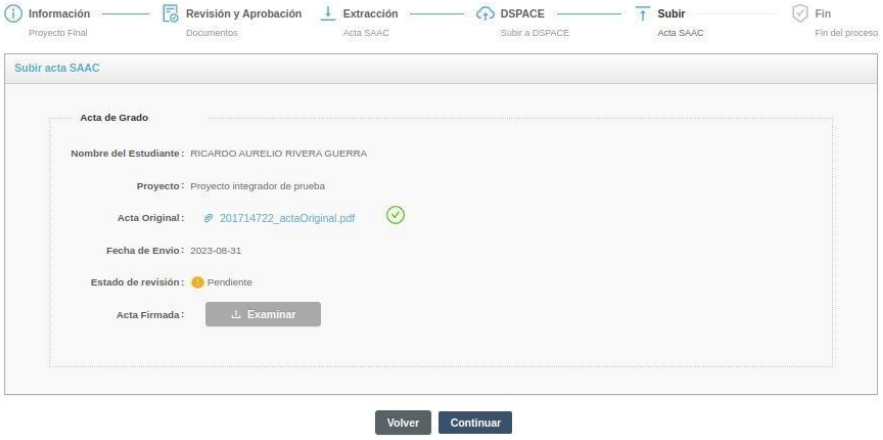
Flujo revisión individual parte 6 (Secretaría)



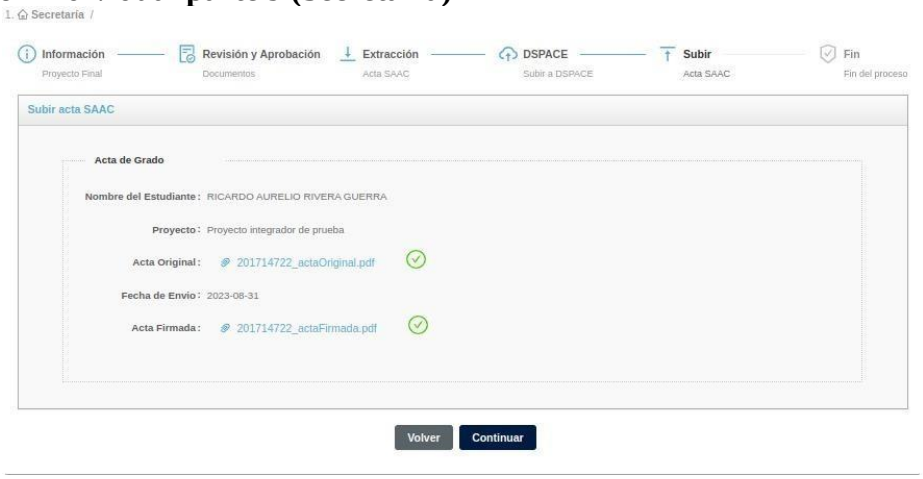
Flujo revisión individual parte 7 (Secretaría)



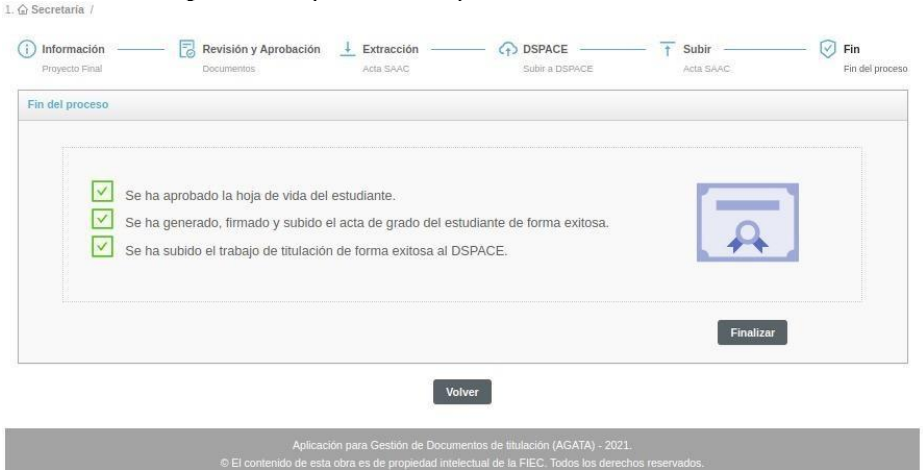
Flujo revisión individual parte 8 (Secretaría)



Flujo revisión individual parte 9 (Secretaría)

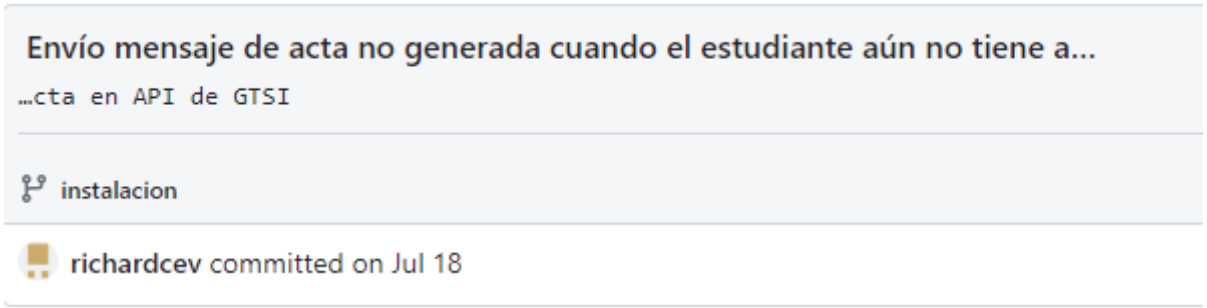


Flujo revisión individual parte 10 (Secretaría)



Código modificado en github.

Código de manejo de error en extraer acta de GTSI parte 1



```
1242  1242                if actaSAAC == 500:
1243  1243                    return JsonResponse({'msg': "Error al intentarse conectar al servidor, por favor comuniquese con el administador"}, status=500)
      1244  +             elif actaSAAC == 404:
      1245  +                 response = HttpResponse(content_type='application/pdf')
      1246  +                 response['Content-Disposition'] = 'Acta no generada'
      1247  +                 return response
1244  1248                v_nuevo = json.loads(actaSAAC.content)
1245  1249                uri = v_nuevo['uri']
1246  1250                cod_act = v_nuevo['cod_acta']

      @@ -7921,6 +7925,8 @@ def buscarActaGTSI(matricula):
7921  7925      print("el valor del response", response.content, response.status_code)
7922  7926      if (response.status_code == 500):
7923  7927          return 500
      7928  +   elif(response.status_code == 404):
      7929  +       return 404
7924  7930      return HttpResponse(response)
```

Código de manejo de error en extraer acta de GTSI parte 2

```
10 ■■■■■ src/ModuloSecretaria/Fase1/ExtraerActa.js

      @@ -138,9 +138,15 @@ class ExtraerActa extends React.Component {
138   138             if (response.status === 200) {
139   139               let header = response.headers["content-disposition"];
140   140               console.log(header);
141         -           if (header === null) {
      141   +           if (header == "Acta no generada"){
      142   +             message.info({
      143   +               content: "El acta aún no ha sido generada, intente en otro momento.",
      144   +               style: FuncionesAuxiliares.layoutInfo(),
      145   +               duration: 5,
      146   +             });
      147   +           } else if (header === null) {
142   148               throw Error;
143         -           } else{
      149   +           } else{
```

Código de solución de error de pasos en revisión individual parte 1

Arreglo de botón continuar en revisión individual

instalacion

richardcev committed on Jul 27

Showing 1 changed file with 10 additions and 14 deletions.

24 administrador/views.py

```
@@ -1274,7 +1274,7 @@ def get(self, request, matricula, tipo, id_periodo):
                        idusuario=estudiante[0].id_usuario.id)
                print(proyecto)
                Pasostitulacion.objects.filter(
-                    idproyecto=proyecto[0].idproyecto.id, idusuario=estudiante[0].id_usuario.id).update(num_step=3, max_step=4)
+                    idproyecto=proyecto[0].idproyecto.id, idusuario=estudiante[0].id_usuario.id).update(num_step=2, max_step=3)
                etapa = Etapa.objects.filter(
                    idperiodo=id_periodo, idnombreetapa=obtenerNombreEtapa(extraerActaGrado))
                log = Logtitulacion(fechaevento=datetime.date.today(), descripcion="El acta de grado ha sido extraída exitosamente",
@@ -1354,7 +1354,7 @@ def post(self, request, matricula, tipo, id_periodo):
                    idusuario=estudiante[0].id_usuario.id)
                print(proyecto)
                Pasostitulacion.objects.filter(
-                    idproyecto=proyecto[0].idproyecto.id, idusuario=estudiante[0].id_usuario.id).update(num_step=3, max_step=4)
+                    idproyecto=proyecto[0].idproyecto.id, idusuario=estudiante[0].id_usuario.id).update(num_step=2, max_step=3)
                etapa = Etapa.objects.filter(
```

Código de solución de error de pasos en revisión individual parte 2

```
@@ -1357,7 +1357,7 @@ def post(self, request, matricula, id, dspace, deuda, id_periodo):
                    idproyecto=proyecto[0].idproyecto.id, idusuario=estudiante[0].id_usuario.id).update(num_step=2, max_step=3)
                etapa = Etapa.objects.filter(
                    idperiodo=id_periodo, idnombreetapa=obtenerNombreEtapa(extraerActaGrado))
                log = Logtitulacion(fechaevento=datetime.date.today(), descripcion="El acta de grado ha sido subido al sistema",
@@ -2139,17 +2139,13 @@ def post(self, request, matricula, id, dspace, deuda, id_periodo):
            estudiante = Estudiante.objects.get(matricula=matricula)
            pt = Pasostitulacion.objects.filter(
                idproyecto=id, idusuario=estudiante.id_usuario.id)
-            print("valor de pasos titulacion", pt[0])
-            if pt[0].num_step == 3:
-                print("estamos en el num_step = 3")
-                pt[0].num_step=4
-                pt[0].max_step=6
-                pt[0].save()
+            print("VALOR DE PASOS TITULACION", pt[0])
+            if pt[0].num_step == 2:
+                print("estamos en el num_step = 2")
+                pt.update(num_step=3, max_step=4)
            else:
                print("estoy de regreso para completar el proceso")
-                pt[0].num_step=6
-                pt[0].max_step=6
-                pt[0].save()
+                pt.update(num_step=5, max_step=5)
            return JsonResponse({'msg': 'Se ha guardado correctamente'}, status=200)
        except Estudiante.objects.get(matricula=matricula).DoesNotExist:
            raise Http404("El estudiante no existe")
```

Código de solución de error de pasos en revisión individual parte 3

```
3608  3604
3609  3605                      # Configuración pasos titulación.
3610  3606                      pasos = Pasostitulacion(
3611        -                       idproyecto=proyecto, idusuario=usuario, num_step=0, max_step=2, estadorevision=obj_pend.nombre)
      3607  +                     idproyecto=proyecto, idusuario=usuario, num_step=0, max_step=1, estadorevision=obj_pend.nombre)
3612  3608                      pasos.save()
3613  3609
3614  3610                      # Crea una notificación web para enviar a secretaría.
                @@ -6498,13 +6494,13 @@ def envioCorreoActualizacionTitulacion(correos, info_proyecto, request, rol_usua
6498  6494              estudiante = Estudiante.objects.get(
6499  6495                  matricula=matriculaInd)
6500  6496              Pasostitulacion.objects.filter(
6501        -                 idusuario=estudiante.id_usuario.id, idproyecto=documento.idproyecto.id).update(num_step=2, max_step=3)
      6497  +                 idusuario=estudiante.id_usuario.id, idproyecto=documento.idproyecto.id).update(num_step=1, max_step=2)
6502  6498      elif 'matricula' in request.data:
6503  6499          estudiante = Estudiante.objects.get(
6504  6500              matricula=request.data["matricula"])
6505  6501          correos.append(estudiante.id_usuario.correo)
6506  6502          Pasostitulacion.objects.filter(
6507        -             idusuario=estudiante.id_usuario.id, idproyecto=documento.idproyecto.id).update(num_step=2, max_step=3)
      6503  +             idusuario=estudiante.id_usuario.id, idproyecto=documento.idproyecto.id).update(num_step=1, max_step=2)
6508  6504      # Si aprueba el tutor, le llega un correo al profesor (aparte de los estudiantes)
6509  6505      if rol_usuario == id_rol_tutor:
6510  6506          correos.append(info_proyecto.tutor.correo)
```

Código de solución de error de pasos en revisión individual parte 4

Arreglo botón continuar en revisión individual y cambios en los en en...

...dpoints de prueba

instalacion

richardcev committed 2 weeks ago

Showing **1 changed file** with **204 additions** and **15 deletions.**

219 ■■■■□ administrador/views.py

```
                @@ -1232,6 +1232,7 @@ def get(self, request, matricula, tipo, id_periodo):
1232  1232              else:
1233  1233                  if (DEBUG_CONFIG):
1234  1234                      matriculad = "201705100"
      1235  +                   print("ENTRÉ A ACTA")
1235  1236                  else:
1236  1237                      matriculad = matricula
1237  1238                  # matriculad = matricula
                @@ -1317,6 +1318,10 @@ def post(self, request, matricula, tipo, id_periodo):
1317  1318                          matricula=matricula)
1318  1319                  acta.subidasaac=True
1319  1320                  acta.save()
      1321  +               proyecto = Proyectousuario.objects.filter(
      1322  +               idusuario=estudiante[0].id_usuario.id)
      1323  +               Pasostitulacion.objects.filter(
      1324  +               idproyecto=proyecto[0].idproyecto.id, idusuario=estudiante[0].id_usuario.id).update(num_step=4, max_step=5)
1320  1325                  secretaria = Rolusuario.objects.filter(
1321  1326                      idrol=id_rol_secretaria)
```