PLAN DE PRUEBAS

Integrantes:

- Juan Esteban Eraso
- Santiago Espinosa
- Nicolas Cardona
- Mateo Berrio Villa
- Thomas Brueck
- Santiago Cardenas

Tabla de contenido

Introducción

Objetivo de las pruebas

El propósito del plan de pruebas es garantizar la calidad, funcionalidad y estabilidad del software antes de su implementación definitiva. A través de un proceso sistemático de pruebas, se busca identificar y corregir errores, validar el cumplimiento de los requisitos definidos y asegurar una experiencia de usuario óptima.

El proceso de pruebas incluirá diferentes tipos de validaciones, como pruebas funcionales, de integración y de usabilidad, con el fin de evaluar el correcto comportamiento del sistema en distintos escenarios. Se establecerán criterios de aceptación para cada funcionalidad clave, asegurando que el software cumpla con las expectativas del usuario final y los objetivos del proyecto.

Este plan de pruebas permitirá detectar posibles riesgos y asegurar que la aplicación sea confiable, eficiente y escalable, minimizando problemas en etapas posteriores de desarrollo o producción en cuanto a las funcionalidades como la gestión de usuarios, la gestión de informes de autoevaluación, gestión de comentarios y preguntas orientadoras, consolidaciones informes y exportaciones, CRUD 12 factores (Plantillas base) y toda la gestión de CRUD por parte de los administradores para la gestión dentro de la plataforma en los componentes mencionados.

Estrategias de pruebas

Matriz de Niveles vs. Atributos de Calidad

Los diferentes niveles de prueba con los atributos de calidad que se deben verificar en cada nivel.

- **Funcionalidad:** ¿El sistema cumple con los requisitos funcionales?
- **Usabilidad:** ¿El sistema es fácil de usar?
- **Rendimiento:** ¿El sistema responde rápidamente bajo carga?
- **Seguridad:** ¿El sistema protege los datos sensibles?
- **Compatibilidad:** ¿El sistema funciona en diferentes entornos?
- **Confiabilidad:** ¿El sistema opera sin fallos?

Nivel de Prueba	Funcionalidad	Usabilidad	Rendimiento	Seguridad	Compatibilidad	Confiabilidad
Pruebas de Unidad	**X**					**X**
Pruebas de Integración	**X**		**X**			**X**
Pruebas de Sistema	**X**	**X**	**X**	**X**	**X**	**X**
Pruebas de Aceptación	**X**	**X**				

Matriz de Descomposición Funcional

ÉPICA	FUNCIONALIDAD	Pruebas de Funcionalidad	Pruebas de Rendimiento	Pruebas de Seguridad	Pruebas de Usabilidad	Pruebas de Interfaz de Usuario
Historias de Usuario de Planeación	Asignación de Roles	X		X		
Historias Funcionales Etapa 1	Autenticación credenciales universitarias	X		X		
	Creación de Roles y Permisos	X		X		
	Autenticación doble factor	X		X		
	Interfaz				X	X

	intuitiva y mensajes de conexión					
	Eliminación roles	X		X		
	Login usuarios	X		X		
	Login director de programa y asistente	X		X		
	Login comité de acreditación	X		X		
	Registro de todos los usuarios de la plataforma	X		X		
	Creación de informes de autoevaluación	X				
	Línea de tiempo general	X				
	Alertas de errores	X				
Historias Funcionales Etapa 3	Subida de archivo DOFA	X				
	Visualizar informes de autoevaluación	X				
	Filtrar informes de autoevaluaci	X				

	ón					
	Edición simultánea de informe	X				
	Almacenamiento de información de modificaciones	X	X			
	Revertir versiones en informe	X				
	Envío de notificaciones	X				
	Notificación de inicio del proceso	X				
	Asignación de Responsable y Fecha	X				
	Extensión de Tiempo y Recursos	X				
	Asignación de Fechas a Factores	X				
	Actualización de Barras de Progreso	X	X			
	Registro de Estado de Preguntas Orientadoras	X				

	Escritura libre	X				
	Escritura guiada	X				
	Aprobación o desaprobación de factores	X				
	Comentarios en factores	X			X	
	Edición de comentarios	X				
	Visualizar versiones de factores	X				
	Exportación de informe de autoevaluación	X	X			
	Registro de actividad	X				
	Notificaciones de actualización	X				
	Consolidación informe de autoevaluación	X	X			
	Comentarios en informes	X			X	
	Aprobación o desaprobación de comentarios	X				

	Historial de entregas y revisiones	X				
	Organización y Exportación de la Plantilla	X		X	X	X
	Visualizar DOFA	X				
	Exportar factores	X	X			
	Ponderación final	X				

Esta matriz descompone el sistema en sus módulos y subsistemas, y luego identifica los tipos de pruebas que se deben realizar en cada uno. Los tipos de pruebas pueden incluir:

- Pruebas de funcionalidad
- Pruebas de rendimiento
- Pruebas de seguridad
- Pruebas de usabilidad
- Pruebas de interfaz de usuario

Alcance del plan de pruebas

Épica	HU	Funcionalidad	Atributo de Calidad	Tipos de Pruebas
Autenticación y Datos	SCRUM-15	Autenticación credenciales universitarias	Funcionalidad, Seguridad	Funcionales, Seguridad
	SCRUM-22	Autenticación doble factor	Seguridad, Fiabilidad	Seguridad, Unitarias
	SCRUM-66	Login usuarios	Funcionalidad, Seguridad	Funcionales, Unitarias
	SCRUM-67	Login director de programa y asistente	Funcionalidad, Seguridad	Funcionales, Unitarias
	SCRUM-68	Login comité de acreditación	Funcionalidad, Seguridad	Funcionales, Unitarias
Gestión de informes	SCRUM-17	Creación de Roles y Permisos	Funcionalidad, Seguridad	Funcionales, Seguridad
	SCRUM-20	Asignación de Roles	Funcionalidad, Seguridad	Funcionales, Seguridad
	SCRUM-28	Exportación de informe de autoevaluación	Funcionalidad, Usabilidad	Funcionales, Usabilidad
	SCRUM-29	Visualizar informes de autoevaluación	Funcionalidad, Usabilidad	Funcionales, Usabilidad
	SCRUM-30	Filtrar informes de autoevaluación	Funcionalidad, Usabilidad	Funcionales, Usabilidad
	SCRUM-33	Consolidación informe de autoevaluación	Funcionalidad, Seguridad	Funcionales, Seguridad
	SCRUM-34	Edición simultánea de informe	Funcionalidad, Seguridad	Funcionales, Seguridad

	SCRUM-35	Almacenamiento de información de modificaciones en informe	Seguridad, Fiabilidad	Seguridad, Unitarias
Gestión de Plantillas	SCRUM-47	Organización y Exportación de la Plantilla	Funcionalidad, Usabilidad	Funcionales, Usabilidad
	SCRUM-48	Asignación de Responsable y Fecha	Funcionalidad, Usabilidad	Funcionales, Usabilidad
	SCRUM-49	Extensión de Tiempo y Recursos	Funcionalidad, Usabilidad	Funcionales, Usabilidad
	SCRUM-51	Actualización de Barras de Progreso	Funcionalidad, Fiabilidad	Funcionales, Usabilidad
	SCRUM-52	Registro de Estado de Preguntas Orientadoras	Funcionalidad, Usabilidad	Funcionales, Usabilidad
Plan de mejoramiento	SCRUM-63	Interfaz intuitiva y mensajes de conexión	Usabilidad, Interfaz	Usabilidad, UI
	SCRUM-65	Eliminación roles	Funcionalidad, Seguridad	Funcionales, Seguridad
Seguimiento y Progreso	SCRUM-71	Línea de tiempo general	Funcionalidad, Usabilidad	Funcionales, Usabilidad
	SCRUM-72	Alertas de errores	Funcionalidad, Usabilidad	Funcionales, Usabilidad
Notificaciones y Alertas	SCRUM-32	Notificaciones de actualización	Funcionalidad, Usabilidad	Funcionales, Usabilidad
	SCRUM-45	Envío de notificaciones	Funcionalidad, Usabilidad	Funcionales, Usabilidad
	SCRUM-46	Notificación de inicio del proceso	Funcionalidad, Usabilidad	Funcionales, Usabilidad

Estrategia de Pruebas Detallada

- **Pruebas de Unidad:**
  - Probar cada función y método individualmente.
  - Utilizar marcos de pruebas unitarias (JUnit, PyTest, etc.).
  - Asegurar que cada unidad cumpla con su especificación.
- **Pruebas de Integración:**
  - Probar la interacción entre los diferentes módulos y subsistemas.
  - Verificar el flujo de datos y la comunicación entre los componentes.
  - Utilizar mocks y stubs para simular dependencias externas.
- **Pruebas de Sistema:**
  - Probar el sistema completo en un entorno similar al de producción.
  - Realizar pruebas de funcionalidad, rendimiento, seguridad, usabilidad y compatibilidad.

    - Utilizar casos de prueba basados en los requisitos del sistema.
- **Pruebas de Aceptación:**
    - Realizar pruebas con usuarios finales para validar que el sistema cumple con sus necesidades.
    - Utilizar escenarios de prueba basados en casos de uso reales.
    - Obtener la aprobación del cliente antes de la implementación en producción.
- **Pruebas de Seguridad:**
    - Pruebas de penetración: Simulación de ataques para encontrar vulnerabilidades.
    - Pruebas de autorización: Verificar que los roles y permisos funcionan correctamente.
    - Pruebas de inyección SQL: Verificar que el sistema es resistente a ataques de inyección de código.
    - Pruebas de vulnerabilidad de software de terceros: Verificar que las librerías y dependencias sean seguras.

Entorno de Pruebas

- Crear un entorno de pruebas aislado del entorno de producción.
- Utilizar datos de prueba realistas para simular el uso en producción.
- Configurar el entorno de pruebas para reflejar la infraestructura de producción.

Esquema de trabajo

Definición de Roles y Responsabilidades

Para garantizar un flujo de trabajo estructurado en las pruebas, cada miembro del equipo asumirá un rol específico:

- QA Tester: Diseñar y ejecutar los casos de prueba, documentar hallazgos y validar correcciones.
- Desarrolladores: Resolver defectos identificados en las pruebas y optimizar funcionalidades.
- Líder de Pruebas: Coordinar la ejecución de pruebas, gestionar prioridades y consolidar reportes de avance.

Coordinación del Equipo

- Planificación de Pruebas: Se definirán objetivos de prueba en cada sprint, alineados con los requerimientos del sistema.

- Asignación de Responsabilidades: Se distribuirán las tareas de pruebas entre los testers y se establecerán prioridades.
- Seguimiento de Avances: Se registrarán los hallazgos y se documentarán los defectos en herramientas de gestión.

Metodología de Trabajo y Herramientas de Gestión

- Ejecución Iterativa de Pruebas: Las pruebas se realizan en paralelo con el desarrollo para detectar errores tempranos.
- Gestión de Incidencias: Se utilizarán herramientas como JIRA o Trello para documentar fallos, asignar tareas y monitorear avances.
- Automatización de Pruebas: Se emplearán herramientas como Selenium o Postman cuando sea viable, para optimizar el proceso.

Estrategia de Comunicación Interna

- Reuniones de Sincronización: Se realizarán sesiones breves para revisar avances, problemas encontrados y acciones a seguir.
- Registro de Cambios: Se documentan ajustes en pruebas y hallazgos críticos para mejorar la trazabilidad.
- Uso de Canales de Comunicación: Se utilizarán herramientas como Slack o Microsoft Teams para mantener la colaboración del equipo.

Herramientas de apoyo

Para garantizar la calidad y eficiencia en el desarrollo y validación del software, se utilizarán diversas herramientas de apoyo que facilitarán la gestión de pruebas, la automatización, el control de versiones y la integración continua. A continuación, se detallan las herramientas seleccionadas y su propósito dentro del proyecto:

1. Jira – Gestión de Tareas y Pruebas

- **Uso:** Jira se utilizará para la planificación, seguimiento y gestión de incidencias en el proceso de desarrollo y pruebas del software.
- **Propósito:**
    - Crear y asignar tareas relacionadas con pruebas.
    - Documentar y gestionar defectos encontrados durante la ejecución de pruebas.
    - Monitorear el estado de cada prueba dentro del ciclo de desarrollo.

2. Git/GitHub – Control de Versiones y Colaboración

- **Uso:** Git y GitHub permitirán el control de versiones del código fuente, asegurando un desarrollo colaborativo y organizado.
- **Propósito:**
    - Mantener un historial de cambios en el código fuente y scripts de pruebas.
    - Facilitar la integración de nuevas funcionalidades y correcciones sin afectar la estabilidad del proyecto.
    - Permitir la revisión y auditoría del código mediante pull requests y ramas específicas para pruebas.

3. Selenium – Automatización de Pruebas Funcionales

- **Uso:** Selenium se emplea para la automatización de pruebas en la interfaz de usuario de la aplicación.
- **Propósito:**
    - Automatizar la ejecución de casos de prueba en navegadores web.
    - Validar el correcto funcionamiento de la aplicación desde la perspectiva del usuario final.
    - Reducir el tiempo de ejecución de pruebas repetitivas y mejorar la eficiencia del proceso de validación.

4. TDD (Test-Driven Development) – Desarrollo Guiado por Pruebas

- **Uso:** Se aplicará la metodología TDD para desarrollar el código basado en pruebas antes de su implementación.
- **Propósito:**
    - Escribir pruebas antes del código para asegurar su correcto funcionamiento desde el inicio.
    - Mejorar la calidad del software al fomentar un diseño modular y más robusto.
    - Reducir la cantidad de errores en etapas avanzadas del desarrollo, optimizando la detección temprana de defectos.

5. Postman – Pruebas de API

- **Uso:** Postman será utilizado para la validación de APIs y servicios web.
- **Propósito:**
    - Probar y validar peticiones HTTP (GET, POST, PUT, DELETE) a la API.
    - Automatizar pruebas de API para asegurar su correcto funcionamiento y respuesta esperada.

- Identificar errores en la comunicación entre el backend y el frontend, garantizando la integridad de los datos.

TIPOS DE PRUEBAS A APLICAR

Para garantizar la calidad y correcto funcionamiento del sistema desarrollado para la reacreditación, se aplicarán las siguientes pruebas:

Pruebas de Unidad

Cada módulo del sistema será probado de forma individual para garantizar que cada función, clase o método se comporta como se espera. En este caso, se evaluarán funcionalidades clave como la carga de documentos, validación de datos y generación de reportes.

- **Ejemplo:** Se probará que la carga de archivos en el sistema almacene correctamente los documentos sin corromper los datos.
- **Responsable:** El equipo de desarrollo, con cada miembro evaluando las unidades de código que ha implementado. El líder del equipo coordinará la ejecución de estas pruebas.

Pruebas de Integración

Se verificará que los diferentes módulos del sistema trabajen juntos correctamente, asegurando una comunicación fluida entre ellos, especialmente en la transferencia de información entre el módulo de gestión de documentos y el de generación de reportes.

- **Ejemplo:** Se validará que los datos cargados en el sistema sean reflejados correctamente en los reportes generados, sin pérdida ni alteración de información.
- **Responsable:** El equipo de desarrollo diseñará y ejecutará pruebas para validar la integración. El líder del equipo supervisará este proceso.

Pruebas Funcionales

Se comprobará que todas las funcionalidades del sistema cumplen con los requisitos establecidos, asegurando que cada acción dentro del sistema se realice correctamente según lo esperado.

- **Ejemplo:** Se evaluará que los usuarios puedan cargar, editar y eliminar documentos correctamente, y que los reportes generados contengan la información exacta.
- **Responsable:** El equipo de desarrollo diseñará y ejecutará pruebas funcionales basadas en los criterios de aceptación. El líder del equipo asegurará el cumplimiento de estos requisitos.

Pruebas de Seguridad

Se evaluará la seguridad del sistema en términos de autenticación de usuarios, gestión de permisos y protección de datos sensibles. Se identificarán vulnerabilidades y se implementarán medidas para mitigarlas.

- **Ejemplo:** Se probará que solo los usuarios autorizados puedan acceder a documentos específicos y que los datos almacenados estén cifrados.
- **Responsable:** El equipo de desarrollo aplicará pruebas de seguridad y corregirá vulnerabilidades. El líder del equipo supervisará este proceso y garantizará que se implementen medidas de protección adecuadas.

Pruebas de Usabilidad

Se evaluará la facilidad de uso del sistema y la experiencia de los usuarios al interactuar con la interfaz. Se recopilará retroalimentación de los usuarios finales para realizar mejoras.

- **Ejemplo:** Se llevará a cabo una prueba con usuarios administrativos y docentes para verificar que puedan navegar fácilmente por la plataforma y realizar las acciones sin complicaciones.
- **Responsable:** Usuarios finales participarán en las pruebas, proporcionando comentarios sobre la interfaz y la experiencia. El equipo de desarrollo analizará la retroalimentación y realizará mejoras.

Esfuerzo estimado

HU	FUNCIONALIDAD	BAJA	MEDIA	ALTA
1.1	Autenticación credenciales universitarias	1H	2H	3H

1.2	Alertas y Notificaciones de Actualización	1H	2H	3H
1.3	Envío de Notificaciones	1H	2H	3H
1.4	Subida de archivo DOFA	1H	2H	3H
1.5	Registro de Actividad del Sistema	1H	2H	3H
1.6	Creación de Roles y Permisos	1H	2H	3H
1.7	Historial de Entregas y Revisión	1H	2H	3H
1.8	Asignación de Roles	1H	2H	3H
1.9	Autenticación doble factor	1H	2H	3H
1.10	Exportación de informe de autoevaluación	1H	2H	3H
2.1	Visualizar informes de autoevaluación	1H	2H	3H
2.2	Filtrar informes de autoevaluación	1H	2H	3H
2.3	Registro de actividad	1H	2H	3H
2.4	Consolidación informe de autoevaluación	1H	2H	3H
2.5	Edición simultánea de informe	1H	2H	3H
2.6	Almacenamiento de información de modificaciones en informe	1H	2H	3H

2.7	Revertir versiones en informe	1H	2H	3H
2.8	Comentarios en informes	1H	2H	3H
2.9	Aprobación o desaprobación de comentarios en informe	1H	2H	3H
3.1	Organización y Exportación de la Plantilla	1H	2H	3H
3.2	Asignación de Responsable y Fecha	1H	2H	3H
3.3	Extensión de Tiempo y Recursos	1H	2H	3H
3.4	Asignación de Fechas a Factores	1H	2H	3H
3.5	Actualización de Barras de Progreso	1H	2H	3H
3.6	Registro de Estado de Preguntas Orientadoras	1H	2H	3H
4.1	Interfaz intuitiva y mensajes de conexión	1H	2H	3H
4.2	Eliminación de roles	1H	2H	3H
4.3	Login usuarios	1H	2H	3H
4.4	Login director de programa y asistente	1H	2H	3H
4.5	Login comité de acreditación	1H	2H	3H

4.6	Registro de todos los usuarios de la plataforma	1H	2H	3H
4.7	Creación de informes de autoevaluación	1H	2H	3H
4.8	Línea de tiempo general	1H	2H	3H
4.9	Alertas de errores	1H	2H	3H
4.10	Visualizar DOFA	1H	2H	3H
5.1	Escritura libre	1H	2H	3H
5.2	Escritura guiada	1H	2H	3H
5.3	Aprobación o desaprobación de factores	1H	2H	3H
5.4	Comentarios en factores	1H	2H	3H
5.5	Exportar factores	1H	2H	3H
5.6	Ponderación final	1H	2H	3H
5.7	Edición de comentarios	1H	2H	3H
5.8	Visualizar versiones de factores	1H	2H	3H

ENTREGABLES DEL PROCESO

Para asegurar un proceso de pruebas organizado y documentado, se generarán los siguientes entregables:

1. Plan de Pruebas

- **Descripción:** Documento que define los objetivos, el alcance, la estrategia y los recursos necesarios para la validación del sistema. Contendrá información sobre los tipos de pruebas aplicadas, los criterios de aceptación y las responsabilidades del equipo.
- **Propósito:** Garantizar que las pruebas sigan una planificación estructurada y alineada con los requisitos del proyecto.

2. Casos de Prueba

- **Descripción:** Documentos que describen las condiciones, pasos, datos necesarios y resultados esperados para cada prueba. Estos casos de prueba estarán orientados a la funcionalidad clave del sistema, como la carga de documentos, validación de datos y generación de reportes.
- **Propósito:** Servir como guía para la ejecución de pruebas y asegurar que todas las funcionalidades del sistema se validan correctamente.

3. Informes de Pruebas

- **Descripción:** Reportes que resumen los resultados de las pruebas realizadas, indicando el estado de cada una, los problemas detectados y su impacto en el sistema.
- **Propósito:** Proporcionar una visión clara del estado del software tras las pruebas y ayudar en la toma de decisiones sobre correcciones y mejoras.

4. Documentación de Defectos

- **Descripción:** Registros detallados de los errores encontrados durante las pruebas, incluyendo su descripción, los pasos para reproducirlos, su nivel de criticidad y el estado de su resolución.
- **Propósito:** Facilitar la gestión y corrección de errores para mejorar la estabilidad y funcionalidad del sistema.

5. Métricas de Pruebas

- **Descripción:** Datos cuantitativos sobre el proceso de pruebas, como el número de pruebas ejecutadas, pruebas exitosas y fallidas, tiempo invertido y porcentaje de cobertura de pruebas.

- **Propósito:** Evaluar la efectividad del proceso de pruebas y proporcionar información clave para optimizar la calidad del sistema.

Mecanismos de Seguimiento y Control

Para asegurar que el proceso de pruebas se desarrolle de manera eficiente y alineada con los objetivos del proyecto, se implementarán diversos mecanismos de monitoreo y control.

1. Revisiones Periódicas del Progreso

- **Frecuencia:** Una vez por semana.
- **Participantes:** Todo el equipo de trabajo.
- **Objetivo:** Evaluar los avances en las pruebas, detectar posibles inconvenientes y definir ajustes en la estrategia si es necesario.
- **Dinámica:** Se compartirán informes de estado con métricas clave y se discutirán los desafíos encontrados para tomar decisiones oportunas.

2. Métricas de Avance

- **Descripción:** Indicadores clave que permitirán medir el estado y desempeño de las pruebas en relación con el cronograma del proyecto. Se analizarán aspectos como la cantidad de pruebas ejecutadas, la tasa de éxito/fallo y el número de errores detectados.
- **Frecuencia de Revisión:** Se actualizarán constantemente, con reportes formales en las reuniones semanales.

3. Informes de Evaluación de Pruebas

- **Periodicidad:** Se generará un reporte cada semana.
- **Elementos incluidos:**
    - Cantidad de pruebas realizadas y pendientes.
    - Resultados obtenidos (pruebas exitosas vs. fallidas).
    - Clasificación de errores según su nivel de impacto.
    - Tiempo estimado de resolución de defectos críticos.
    - Alcance de la cobertura de pruebas en los diferentes módulos.
- **Finalidad:** Facilitar la toma de decisiones basada en datos y garantizar que el sistema cumpla con los estándares de calidad requeridos.

4. Registro y Seguimiento en Plataforma de Gestión

- **Actualización:** Diaria.
- **Contenido registrado:** Fallos detectados, avances en la resolución de errores y progreso en la ejecución de pruebas.

- **Objetivo:** Mantener un control detallado del estado de cada tarea dentro del proceso de pruebas y mejorar la comunicación interna del equipo.

Tipo de Prueba	Métrica	Descripción	Fórmula	Herramienta	Frecuencia
Pruebas de Unidad	Cobertura de Código	Mide la cantidad de líneas de código ejecutadas durante las pruebas unitarias.	$\frac{\text{Líneas de código cubiertas}}{\text{Total de líneas de código}} * 100$	Coverage.py, SonarQube	Semanal
Pruebas de Integración	Tasa de Integración Exitosa	Indica el porcentaje de integraciones que no presentan errores y funcionan correctamente.	$\frac{\text{Integraciones exitosas}}{\text{Integraciones totales}} * 100$	Postman, JUnit	Mensual
Pruebas Funcionales	Porcentaje de Casos de Prueba Pasados	Mide qué proporción de las pruebas ejecutadas han sido exitosas.	$\frac{\text{Casos de prueba exitosos}}{\text{Casos de prueba ejecutados}} * 100$	Selenium, Cypress	Semanal
Pruebas de Seguridad	Nivel de Riesgo de Vulnerabilidades	Evalúa la criticidad de los problemas de seguridad detectados,	$\frac{\text{Suma de niveles de severidad de vulnerabilidades}}{\text{Total de vulnerabilidades encontradas}}$	OWASP ZAP, Burp Suite	Mensual

		clasificándolos por severidad.			
Pruebas de Usabilidad	Tiempo Promedio de Tarea	Indica cuánto tiempo le toma a un usuario completar una acción en la aplicación.	$\frac{\text{Tiempo total para completar tareas}}{\text{Número de tareas evaluadas}}$	Hotjar, Google Analytics	Cada dos semanas