Plan de Pruebas - FlowLance

Contexto del Proyecto

FlowLance es una plataforma en línea diseñada para facilitar la interacción entre freelancers y clientes. Este plan de pruebas ha sido desarrollado para asegurar que las funcionalidades principales del sistema operen según lo esperado y cumplan con los estándares de calidad establecidos. La aplicación se desarrollará utilizando Django para el backend, con PostgreSQL como base de datos, y Selenium se empleará para las pruebas automatizadas.

Objetivo de las Pruebas

El propósito del plan de pruebas de FlowLance es asegurar que la plataforma cumpla con los estándares de calidad funcional, seguridad, y experiencia de usuario. El objetivo principal de las pruebas es validar que las funcionalidades clave de la plataforma, como la autenticación, gestión de perfiles, gestión de proyectos y procesamiento de pagos, operen sin errores y satisfagan las expectativas de los usuarios finales, tanto freelancers como clientes.

Objetivos Específicos:

1. **Verificación de Funcionalidades Críticas**: Garantizar que las áreas esenciales de la plataforma, como el registro de usuarios, creación de proyectos y gestión de pagos, funcionen correctamente bajo condiciones normales y de carga.

2. **Detección y Corrección de Defectos**: Identificar y corregir defectos en una fase temprana del desarrollo, asegurando una calidad continua a lo largo del ciclo de vida del proyecto. Se prioriza la detección temprana de defectos en módulos críticos para minimizar el impacto en el desarrollo global.

3. **Pruebas de Integración**: Asegurar que los distintos módulos de la plataforma, como la interacción entre freelancers y clientes, se integren de manera fluida, garantizando la coherencia de datos y la funcionalidad interdependiente.

4. **Validación de Interacciones del Usuario**: Evaluar la experiencia del usuario para garantizar que la plataforma sea intuitiva, fácil de usar y accesible desde diferentes dispositivos, asegurando la satisfacción de los usuarios finales.

5. **Cumplimiento de los Requisitos de Seguridad**: Validar que se han implementado las medidas necesarias para la protección de los datos de los usuarios, asegurando que el sistema sea robusto contra vulnerabilidades de seguridad.

Este plan se integrará en el ciclo de vida del proyecto a través de pruebas continuas, combinando pruebas manuales y automatizadas, y asegurando que se cumplan los criterios de calidad antes de cada entrega.

Estrategia de Pruebas

La estrategia de pruebas de FlowLance está diseñada para asegurar la calidad del software en términos de funcionalidad, rendimiento, usabilidad, seguridad, fiabilidad y compatibilidad. Esta estrategia se basa en la ejecución de diferentes tipos de pruebas, cada una dirigida a cubrir aspectos específicos de la aplicación. A continuación, se presentan las matrices de niveles vs. atributos de calidad y la descomposición funcional para garantizar que se cubran todas las áreas críticas del sistema.

Matriz de Niveles vs. Atributos de Calidad

La matriz de niveles vs. atributos de calidad especifica qué tipos de pruebas se aplicarán en cada nivel del sistema, en función de los atributos de calidad requeridos. Esto garantiza una cobertura exhaustiva de las pruebas en todas las dimensiones relevantes del sistema.

Niveles / Atributos de calidad	Funcionalidad	Rendimiento	Compatibilidad	Usabilidad	Fiabilidad	Seguridad
Pruebas Unitarias	X			X	X	
Pruebas de Integración	X					
Pruebas de Regresión	X			X	X	
Pruebas de Sistema	X				X	
Pruebas de Aceptación	X				X	
Pruebas de concurrencia		X	X	X	X	
Prueba de seguridad						X

Matriz de Descomposición Funcional

La matriz de descomposición funcional describe los diferentes procesos y subprocesos de negocio en la aplicación FlowLance, detallando las funcionalidades que serán probadas, los atributos de calidad que se evaluarán y los tipos de pruebas que se utilizarán para asegurar la calidad del sistema.

Proceso de negocio	Subproceso	Funcionalidades	Atributo de calidad	Tipo de prueba
Registro y Autenticación	Registro de usuarios	Creación de cuentas para freelancers y clientes	Seguridad	Prueba de seguridad
Registro y Autenticación	Autenticación de usuarios	Inicio de sesión con credenciales	Funcionalidad	Pruebas Unitarias
Gestión de Proyectos	Creación de proyectos	Publicación de proyectos por parte de clientes	Funcionalidad	Pruebas de Integración
Gestión de Proyectos	Asignación de freelancers	Asignación de freelancers a proyectos	Usabilidad	Pruebas de Aceptación
Gestión de Pagos	Procesamiento de pagos	Pago seguro a freelancers	Fiabilidad, Seguridad	Pruebas de concurrencia
Panel de Control	Gestión de proyectos y pagos	Seguimiento de proyectos y pagos	Rendimiento	Pruebas de Regresión
Panel de Control	Notificaciones automáticas	Alertas para hitos y pagos pendientes	Compatibilidad	Pruebas de concurrencia
Mensajería	Comunicación interna	Sistema de mensajería entre freelancers y clientes	Usabilidad	Pruebas de Sistema
Perfil de Freelancer	Gestión de habilidades	Creación y actualización de perfiles	Usabilidad	Pruebas Unitarias
Seguridad	Autenticación de doble factor	Activación y uso de doble autenticación	Seguridad	Prueba de seguridad

Esquema de Trabajo

La estrategia del grupo de trabajo en FlowLance está diseñada para asegurar una colaboración eficiente entre los diferentes actores del proyecto: el equipo de desarrollo, el equipo de pruebas, el cliente, y otros stakeholders relevantes. La coordinación entre estos grupos es clave para el éxito del proceso de pruebas y la calidad del producto final.

1. Roles y Responsabilidades del Equipo de Trabajo

Se han definido los siguientes roles dentro del equipo para facilitar la correcta implementación del plan de pruebas:

- **Scrum Master:** El Scrum Master es responsable de coordinar el equipo de pruebas y asegurar que el proceso de pruebas esté alineado con el plan general del proyecto. Se encargará de organizar las ceremonias Scrum (reuniones diarias, revisiones y retrospectivas de sprint) y de garantizar que cualquier problema en el proceso de pruebas se resuelva rápidamente.

- **Product Owner:** El Product Owner se asegura de que las funcionalidades críticas del sistema sean evaluadas adecuadamente. También valida los casos de prueba y coordina para garantizar que las pruebas estén alineadas con las expectativas y los criterios de aceptación del proyecto.

- **Ingenieros de Pruebas:** El equipo de ingenieros de pruebas (incluyendo desarrolladores con responsabilidades en pruebas) será responsable de la ejecución de pruebas unitarias, de integración y automatizadas, así como de la realización de pruebas manuales para asegurar la cobertura total del sistema. Los ingenieros de pruebas colaborarán estrechamente con los desarrolladores para reportar y corregir defectos.

2. Coordinación con el Cliente y Stakeholders

La interacción constante con el cliente y otros stakeholders como la profesora es fundamental para validar los avances del proyecto y asegurarse de que las pruebas estén alineadas con los requisitos del sistema. Se implementará el siguiente esquema de coordinación:

- **Revisión de Casos de Prueba y Criterios de Aceptación:** Durante el proceso de pruebas, se realizarán revisiones conjuntas de los casos de prueba para asegurar que estén alineados con los requisitos del sistema. El cliente validará los criterios de aceptación para garantizar que los casos de prueba cubran todas las áreas críticas del proyecto.

- **Entregas Iterativas y Retroalimentación Continua:** Se realizarán entregas iterativas del producto, integrando las pruebas dentro del ciclo de desarrollo continuo. Cada entrega incluirá un conjunto de funcionalidades probadas que serán evaluadas por el cliente, quien proporcionará retroalimentación para las siguientes iteraciones del proyecto.

3. Flujo de Trabajo del Equipo de Pruebas

El flujo de trabajo del equipo de pruebas estará completamente integrado con el ciclo de desarrollo ágil del proyecto, siguiendo la metodología Scrum. A continuación, se describen los aspectos clave del esquema de trabajo:

- **Sprints de Pruebas y Desarrollo Simultáneos:** Las pruebas serán ejecutadas en paralelo con el desarrollo, siguiendo el enfoque de Desarrollo Guiado por Pruebas (TDD). Cada sprint incluirá la creación de casos de prueba, la ejecución de pruebas automatizadas y manuales, y la corrección de defectos encontrados.

- **Pruebas Continuas e Integración Continua (CI/CD):** El proceso de pruebas se integrará con el pipeline de CI/CD para asegurar que las pruebas automatizadas se ejecuten de forma continua. Cada vez que se realice un cambio en el código, se ejecutarán pruebas de regresión para garantizar que no se introduzcan defectos en el sistema.

- **Colaboración entre Desarrollo y Pruebas:** Los desarrolladores y los ingenieros de pruebas trabajarán en estrecha colaboración, revisando los resultados de las pruebas de manera diaria. Los defectos encontrados serán asignados a los desarrolladores para su resolución. Una vez corregidos, se volverán a ejecutar las pruebas para asegurar que el problema esté resuelto.

4. Herramientas de Comunicación y Colaboración

Para garantizar una comunicación fluida y eficiente dentro del equipo y con los stakeholders, se utilizarán las siguientes herramientas:

- **JIRA:** Para la gestión de tareas, el seguimiento de defectos y la coordinación de las actividades del equipo de pruebas.
- **GitHub Actions:** Para la automatización del pipeline de pruebas e integración continua.
- **Microsoft Teams:** Para las reuniones con el cliente.
- **Google Docs / Confluence:** Para la documentación colaborativa y la creación de informes de pruebas.

5. Plan de Coordinación y Reporte

- **Revisión de Progreso Diario (Daily Scrum):** El equipo se reunirá diariamente para discutir el progreso de las pruebas, compartir hallazgos y coordinar los próximos pasos. Los defectos encontrados se discutirán para definir la prioridad de corrección.
- **Reuniones de Sprint Review:** Al finalizar cada sprint, se realizará una reunión de revisión con el cliente para discutir las correcciones necesarias. El cliente revisará las funcionalidades probadas y proporcionará retroalimentación sobre las pruebas de aceptación.

Herramientas de Apoyo

El proceso de pruebas de FlowLance se apoyará en una serie de herramientas de software y artefactos de ingeniería, que facilitan la gestión, ejecución y automatización de las pruebas. Estas herramientas asegurarán que el proceso sea eficiente, rastreable y adaptable a los diferentes entornos de desarrollo. A continuación, se detallan las principales herramientas que se utilizarán:

1. Selenium

- **Descripción:** Selenium es una herramienta de automatización de pruebas para aplicaciones web que permite simular interacciones de usuario en navegadores.
- **Uso en FlowLance:** Será utilizado principalmente para automatizar las pruebas de regresión y las pruebas funcionales enfocadas en la interfaz de usuario (UI), como flujos críticos de inicio de sesión, gestión de proyectos y procesamiento de pagos.

- **Beneficios:** Automatización de tareas repetitivas, como la ejecución de pruebas funcionales y de regresión, asegurando que los cambios en el código no introduzcan errores.

2. GitHub Actions

- **Descripción:** GitHub Actions es una herramienta de integración continua y entrega continua (CI/CD) que permite automatizar flujos de trabajo en el desarrollo de software.
- **Uso en FlowLance:** Se integrará con el pipeline de pruebas, ejecutando automáticamente las pruebas automatizadas y de regresión cada vez que se realice un cambio en el código fuente.
- **Beneficios:** Facilita la ejecución continua de pruebas, asegurando que cualquier cambio en el código pase por un ciclo completo de validación antes de ser integrado en la rama principal.

3. Pytest

- **Descripción:** Pytest es un marco de pruebas de Python que facilita la escritura de pruebas unitarias y de integración.
- **Uso en FlowLance:** Los desarrolladores utilizarán Pytest para crear y ejecutar pruebas unitarias que validen el correcto funcionamiento de componentes individuales de la plataforma, como la autenticación, la gestión de proyectos, y el procesamiento de pagos.
- **Beneficios:** Mejora la cobertura de las pruebas y permite una validación rápida y eficaz del código mediante la automatización de las pruebas unitarias.

6. Git

- **Descripción:** Git es un sistema de control de versiones distribuido que permite el seguimiento y gestión de cambios en el código fuente.
- **Uso en FlowLance:** Git será utilizado junto con GitHub para gestionar el código fuente del proyecto y colaborar entre los miembros del equipo. Cada versión del código será registrada, permitiendo a los desarrolladores y testers rastrear cambios y asegurarse de que las pruebas sean realizadas en la versión correcta.
- **Beneficios:** Facilita la colaboración entre desarrolladores y testers, permitiendo un control preciso sobre los cambios en el código y asegurando que se puedan revertir cambios problemáticos cuando sea necesario.

7. Visual Studio Code (VS Code)

- **Descripción:** Visual Studio Code es un editor de código ligero pero potente, con soporte para múltiples lenguajes y herramientas de desarrollo.
- **Uso en FlowLance:** Se utilizará para escribir el código de las pruebas automatizadas y scripts personalizados para el proceso de pruebas. Los ingenieros de pruebas usarán VS Code para la creación y edición de los scripts de automatización con Selenium y Pytest.
- **Beneficios:** Facilita la escritura de código de prueba eficiente y permite integrar diversas extensiones que mejoran la productividad en la creación de pruebas.

8. Django Test Framework

- **Descripción:** Django cuenta con un marco de pruebas integrado que permite crear y ejecutar pruebas unitarias y de integración dentro del contexto de la aplicación Django.
- **Uso en FlowLance:** Se utilizará para validar el backend de la plataforma, asegurando que las funcionalidades clave como la gestión de usuarios, la gestión de proyectos, y el procesamiento de pagos funcionen correctamente.
- **Beneficios:** Integra las pruebas directamente en el flujo de desarrollo de Django, facilitando la validación del comportamiento de la aplicación en cada cambio de código.

Tipos de Pruebas a Aplicar

El proceso de pruebas de FlowLance incluirá una variedad de tipos de pruebas diseñadas para validar todos los aspectos del sistema, desde funcionalidades individuales hasta la integración completa, el rendimiento bajo carga y la seguridad. Cada tipo de prueba se enfoca en un área específica de la plataforma para asegurar que se cumplen los requisitos funcionales, de rendimiento, de seguridad y de experiencia de usuario.

1. Pruebas Unitarias

- **Descripción:** Las pruebas unitarias se enfocan en validar el correcto funcionamiento de componentes individuales del sistema, como funciones, clases y métodos.
- **Objetivo:** Asegurar que cada componente independiente del sistema funcione correctamente por sí solo.

2. Pruebas de Integración

- **Descripción:** Las pruebas de integración validan que varios componentes del sistema funcionen correctamente cuando se combinan. Se busca identificar defectos relacionados con la interacción entre módulos o servicios.
- **Objetivo:** Garantizar que las interacciones entre los diferentes módulos del sistema, como la gestión de proyectos, pagos y mensajería, se realicen de manera correcta.

3. Pruebas de Regresión

- **Descripción:** Las pruebas de regresión se ejecutan para verificar que las nuevas actualizaciones o correcciones en el código no hayan introducido errores en las funcionalidades existentes.
- **Objetivo:** Asegurar que la adición de nuevas funcionalidades no afecte negativamente a las funcionalidades previamente desarrolladas.

4. Pruebas de Sistema

- **Descripción:** Estas pruebas se encargan de validar el sistema completo, verificando que todas las funcionalidades trabajen como se espera cuando se ejecutan juntas.
- **Objetivo:** Validar que el sistema como un todo funcione correctamente bajo condiciones de uso real.

5. Pruebas de Aceptación

- **Descripción:** Las pruebas de aceptación se realizan para validar que el sistema cumpla con los criterios de aceptación definidos por el cliente y otros stakeholders. Estas pruebas se enfocan en validar los requisitos funcionales y no funcionales desde la perspectiva del usuario final.
- **Objetivo:** Asegurar que el producto final cumple con las expectativas y requisitos del cliente.

6. Pruebas de Concurrencia

- **Descripción:** Las pruebas de concurrencia verifican cómo el sistema maneja múltiples usuarios accediendo y utilizando la plataforma al mismo tiempo.
- **Objetivo:** Validar que el sistema soporte un número elevado de usuarios concurrentes sin afectar su rendimiento ni provocar fallos.

7. Pruebas de Seguridad

- **Descripción:** Las pruebas de seguridad tienen como objetivo identificar vulnerabilidades en el sistema que podrían ser explotadas por atacantes. Se validan aspectos como la protección de datos, autenticación segura y prevención de accesos no autorizados.
- **Objetivo:** Asegurar que el sistema proteja adecuadamente la información sensible de los usuarios y resista ataques externos.

8. Pruebas de Usabilidad

- **Descripción:** Las pruebas de usabilidad evalúan la experiencia del usuario, verificando que el sistema sea fácil de usar, intuitivo y eficiente para los usuarios finales.
- **Objetivo:** Asegurar que la plataforma sea comprensible y fácil de utilizar, tanto para freelancers como para clientes.

Esfuerzo Estimado

El esfuerzo estimado para el proceso de pruebas de FlowLance se ajusta a la carga horaria disponible, que es de aproximadamente 3 horas semanales. A continuación, se presenta la estimación de tiempo necesario para ejecutar las pruebas en función de la duración de cada sprint y el tipo de pruebas a aplicar.

Distribución del Esfuerzo por Sprint

1. Sprint 0 (3 semanas):

- **Objetivo:** Configuración de las herramientas de pruebas y realización de pruebas unitarias en componentes básicos del sistema (autenticación y registro de usuarios).
- **Esfuerzo semanal:** 3 horas.
- **Duración total del Sprint 0:** 9 horas (3 horas/semana x 3 semanas).
- **Actividades:**
    - Configuración de herramientas de pruebas.
    - Pruebas unitarias de las funcionalidades básicas.

2. Sprint 1 (3 semanas):

- **Objetivo:** Extender las pruebas unitarias a módulos adicionales (gestión de proyectos, perfiles) y empezar con pruebas de integración.
- **Esfuerzo semanal:** 3 horas.
- **Duración total del Sprint 1:** 9 horas (3 horas/semana x 3 semanas).
- **Actividades:**
    - Pruebas unitarias adicionales para nuevos módulos.
    - Pruebas de integración iniciales entre frontend y backend.

3. Sprint 2 (4 semanas):

- **Objetivo:** Continuación de pruebas unitarias y de integración, además de iniciar pruebas de regresión para asegurar que las funcionalidades anteriores siguen funcionando correctamente.
- **Esfuerzo semanal:** 3 horas.
- **Duración total del Sprint 2:** 12 horas (3 horas/semana x 4 semanas).
- **Actividades:**
    - Pruebas de integración más avanzadas entre módulos.
    - Pruebas de regresión sobre las funcionalidades ya implementadas.

4. Sprint 3 (4 semanas):

- **Objetivo:** Realización de pruebas de sistema, pruebas de aceptación, pruebas de concurrencia y seguridad, junto con las pruebas finales de usabilidad.
- **Esfuerzo semanal:** 3 horas.
- **Duración total del Sprint 3:** 12 horas (3 horas/semana x 4 semanas).
- **Actividades:**
    - Pruebas completas del sistema (pruebas de sistema y aceptación).
    - Pruebas de concurrencia, seguridad y usabilidad final.

Esfuerzo Total del Proceso de Pruebas por Sprint

Sprint	Duración (semanas)	Esfuerzo Total (Horas)
Sprint 0	3 semanas	9 horas
Sprint 1	3 semanas	9 horas
Sprint 2	4 semanas	12 horas
Sprint 3	4 semanas	12 horas

Distribución del Esfuerzo por Tipo de Prueba

Tipo de Prueba	Esfuerzo Total (Horas)
Pruebas Unitarias	12 horas
Pruebas de Integración	10 horas
Pruebas de Regresión	8 horas
Pruebas de Sistema	8 horas
Pruebas de Aceptación	6 horas
Pruebas de Concurrencia	4 horas
Pruebas de Seguridad	4 horas
Pruebas de Usabilidad	6 horas

Esfuerzo Estimado Total

Actividad	Esfuerzo Total (Horas)
Configuración y Preparación	3 horas
Pruebas durante los Sprints	42 horas
Pruebas de Sistema y Aceptación	14 horas
Pruebas de Concurrencia	4 horas
Pruebas de Seguridad	4 horas
Pruebas de Usabilidad	6 horas
Total	**73 horas**

Entregables del Proceso

El proceso de pruebas de FlowLance generará una serie de entregables clave que permitirán realizar un seguimiento del progreso de las pruebas, evaluar los resultados y garantizar la calidad del producto. Estos entregables incluirán informes detallados sobre las pruebas ejecutadas, la documentación de los defectos identificados y la validación de los criterios de aceptación por parte del cliente y stakeholders.

1. Plan de Pruebas

- **Descripción:** Documento inicial que describe el plan general del proceso de pruebas, incluyendo los objetivos, tipos de pruebas, herramientas de apoyo, y estrategia de pruebas.
- **Momento de entrega: Al inicio del proyecto.**
- **Responsable:** Scrum Master e Ingenieros de Pruebas.

2. Casos de Prueba

- **Descripción:** Conjunto de casos de prueba detallados que cubren todas las funcionalidades críticas del sistema, así como escenarios para pruebas unitarias, de integración, regresión, y sistema.
- **Responsable**: Ingenieros de Pruebas.
- **Objetivo:** Asegurar la cobertura de todas las funcionalidades y escenarios relevantes para la validación del sistema.

3. Registro de Defectos

- **Descripción:** Documento o tablero en JIRA donde se registran todos los defectos encontrados durante las pruebas, incluyendo descripciones detalladas, pasos para reproducirlos, el entorno en que se produjeron, y la severidad y prioridad del defecto.
- **Responsable:** Ingenieros de Pruebas.
- **Objetivo:** Facilitar la identificación y corrección de defectos, y asegurar que se mantenga un seguimiento claro de los problemas que afectan al sistema.

4. Informes de Ejecución de Pruebas

- **Descripción:** Informe que documenta los resultados de los casos de prueba ejecutados en cada sprint. Incluye:
    - Funcionalidades probadas.

  - Resultados de cada caso de prueba (éxito o fallo).
  - Descripción de los defectos identificados y su estado (abierto, en progreso, resuelto).

- **Responsable:** Ingenieros de Pruebas.
- **Objetivo:** Proporcionar visibilidad sobre el estado de las pruebas y la calidad del sistema en cada sprint.

5. Informe de Pruebas de Integración

- **Descripción:** Documento específico que detalla los resultados de las pruebas de integración entre los diferentes módulos del sistema. Incluye:
  - Descripción de los módulos probados.
  - Flujo de datos entre módulos.
  - Resultados de las pruebas de integración (éxito o fallo).
  - Defectos relacionados con la integración.
- **Responsable:** Ingenieros de Pruebas.
- **Objetivo:** Validar la correcta interacción entre los diferentes componentes del sistema y garantizar la cohesión del sistema en su conjunto.

6. Informe de Pruebas de Aceptación

- **Descripción:** Documento que resume los resultados de las pruebas de aceptación realizadas junto con el cliente y los stakeholders. Este informe incluye:
  - Criterios de aceptación validados.
  - Funcionalidades probadas durante la aceptación.
  - Defectos o problemas identificados durante la validación.
- **Responsable:** Ingenieros de Pruebas y Product Owner.
- **Objetivo:** Confirmar que el producto cumple con los requisitos funcionales y no funcionales acordados con el cliente, y que está listo para su despliegue.

7. Informe de Pruebas de Seguridad

- **Descripción:** Informe detallado de los resultados de las pruebas de seguridad, que incluye:
    - Vulnerabilidades encontradas.
    - Riesgos potenciales y su impacto.
    - Medidas correctivas tomadas.
- **Responsable:** Ingenieros de Pruebas.
- **Objetivo:** Validar que el sistema sea seguro y esté protegido contra posibles ataques o vulnerabilidades.

8. Informe Final de Pruebas

- **Descripción:** Documento de cierre del ciclo de pruebas que recopila todos los resultados de las pruebas realizadas a lo largo del proyecto, incluyendo:
    - Resumen de los defectos identificados y resueltos.
    - Estado final de las pruebas.
    - Criterios de aceptación cumplidos.
    - Evaluación de la calidad general del sistema.
- **Responsable:** Ingenieros de Pruebas y Scrum Master.
- **Objetivo:** Proporcionar una visión completa del estado del producto y el proceso de pruebas, con el objetivo de dar cierre al ciclo de pruebas.

Mecanismos de Seguimiento y Control

Para asegurar un control riguroso y eficaz del proceso de pruebas en FlowLance, se implementarán mecanismos de seguimiento y control que incluyen la definición del Oráculo de las pruebas, la clasificación de fallos y defectos, y la utilización de indicadores clave que permitan evaluar tanto el proceso de pruebas como la calidad final del producto de software.

1. Oráculo de las Pruebas

El Oráculo de pruebas es un mecanismo que permite determinar si los resultados de las pruebas son correctos o no, basándose en los requisitos funcionales y no funcionales del sistema. En FlowLance, el oráculo se define a partir de los siguientes elementos:

- **Requisitos funcionales:** La funcionalidad del sistema debe cumplir con los casos de uso y las historias de usuario definidas. Cada prueba estará alineada con estos requisitos para verificar que la funcionalidad entregada corresponde con las expectativas del cliente.

- **Criterios de aceptación:** Los criterios de aceptación definidos servirán como referencia para validar el correcto comportamiento del sistema.

- **Diseño UI/UX:** Se utilizará la especificación de diseño que se hizo en figma para validar que la interfaz de usuario cumpla con las expectativas de usabilidad y presentación establecidas.

- **Buenas prácticas de seguridad y rendimiento:** El sistema debe cumplir con estándares de seguridad y rendimiento, incluyendo la protección de datos, la prevención de vulnerabilidades y la capacidad de manejar múltiples usuarios concurrentes sin degradación significativa del rendimiento.

2. Clasificación de Fallos y Defectos

Para garantizar una priorización eficiente de los defectos encontrados durante las pruebas, se utilizará la siguiente clasificación basada en la severidad y el impacto en el sistema:

Por Severidad

1. **Crítico:** Defectos que causan fallos catastróficos en el sistema, impidiendo su uso o poniendo en riesgo la integridad de los datos.

2. **Alto:** Defectos que afectan funcionalidades importantes del sistema, pero permiten continuar utilizando la plataforma con limitaciones.

3. **Medio:** Defectos que afectan funcionalidades no críticas o que tienen soluciones temporales, pero deben corregirse antes del lanzamiento final.

4. **Bajo:** Defectos menores, generalmente relacionados con problemas estéticos o inconsistencias menores que no afectan la funcionalidad central del sistema.

Por Impacto

1. **Funcional:** Defectos relacionados directamente con el incumplimiento de una funcionalidad específica del sistema.

2. **Seguridad:** Defectos que exponen vulnerabilidades o riesgos relacionados con la seguridad del sistema.

3. **Rendimiento:** Defectos que afectan la velocidad, estabilidad o capacidad del sistema para manejar la carga.

4. **Usabilidad:** Defectos que dificultan o complican la experiencia de usuario, aunque no impidan directamente el uso del sistema.

3. Indicadores Clave

Para evaluar el proceso de pruebas y estimar la calidad del software, se utilizarán los siguientes indicadores clave de desempeño:

Indicadores del Proceso de Pruebas

1. **Porcentaje de Casos de Prueba Ejecutados:**
    - **Descripción:** Mide el número de casos de prueba ejecutados respecto al total de casos planificados.
    - **Fórmula:** (Casos de prueba ejecutados / Casos de prueba planificados) * 100.
    - **Objetivo:** Garantizar que todas las funcionalidades críticas del sistema sean probadas antes del final del sprint o del ciclo de pruebas.
2. **Porcentaje de Casos de Prueba Exitosos:**
    - **Descripción:** Mide el número de casos de prueba que han pasado exitosamente.
    - **Fórmula:** (Casos de prueba exitosos / Casos de prueba ejecutados) * 100.
    - **Objetivo:** Asegurar que la mayoría de las funcionalidades probadas cumplan con los criterios de aceptación.
3. **Tasa de Detección de Defectos:**
    - **Descripción: I**ndica el número de defectos encontrados por cada ciclo de pruebas.
    - **Fórmula:** Número de defectos encontrados / Ciclo de pruebas.
    - **Objetivo:** Medir la efectividad de las pruebas y detectar posibles áreas problemáticas en el sistema.

Indicadores de Calidad del Producto

1. **Cobertura de Pruebas:**
    - **Descripción:** Mide el porcentaje de código o funcionalidades que han sido cubiertas por pruebas.

- **Fórmula:** (Funcionalidades probadas / Total de funcionalidades) * 100.
- **Objetivo:** Asegurar que todas las áreas clave del sistema sean probadas adecuadamente antes de la entrega final.

2. **Tasa de Defectos por Severidad:**
   - **Descripción:** Mide la cantidad de defectos encontrados, clasificados por severidad (Crítico, Alto, Medio, Bajo).
   - **Fórmula:** (Número de defectos por severidad / Total de defectos) * 100.
   - **Objetivo:** Evaluar la calidad general del sistema y priorizar la corrección de defectos críticos y altos.

4. Herramientas de Seguimiento

Para gestionar y medir estos indicadores, se utilizarán las siguientes herramientas:

- **JIRA:** Para el seguimiento de defectos y la generación de informes de casos de prueba ejecutados.
- **GitHub Actions:** Para integrar el pipeline de pruebas y realizar un seguimiento automatizado de los casos de prueba ejecutados.