1. Introducción

Este plan de pruebas para la aplicación FlowLance tiene como objetivo garantizar la calidad y el correcto funcionamiento de todas sus funcionalidades clave. Basado en un enfoque híbrido que combina la flexibilidad de las pruebas post-desarrollo (TAD) con los principios de TDD, este plan permite al equipo ajustar las pruebas a medida que la implementación evoluciona. Este enfoque asegura que las pruebas se adapten a las realidades del desarrollo ágil, evitando la rigidez de las pruebas anticipadas que pueden volverse obsoletas ante cambios en la implementación.

El plan cubre pruebas unitarias, de integración, manuales, automatizadas y de aceptación, priorizando las áreas críticas de la aplicación. A lo largo de los tres sprints de desarrollo, cada uno de aproximadamente cuatro semanas, las pruebas se ejecutarán en distintas etapas para verificar la correcta interacción entre los componentes y asegurar una experiencia de usuario óptima. Se utilizará PostgreSQL como base de datos y Selenium para las pruebas automatizadas, asegurando un entorno de pruebas robusto y alineado con las tecnologías utilizadas en el desarrollo. A través de este enfoque, se busca un equilibrio entre la eficiencia en el desarrollo y la solidez en la calidad del producto final, garantizando que cada componente funcione de manera coherente y responda adecuadamente a los cambios y ajustes del proyecto.

2. Alcance

El plan abarca todas las funcionalidades críticas de la aplicación FlowLance, incluyendo pruebas unitarias, de integración, manuales, automatizadas y de aceptación. Dado que el proyecto se desarrolla en tres sprints, el plan adapta las pruebas a las distintas etapas del desarrollo, priorizando áreas críticas que impactan directamente en la estabilidad y usabilidad de la aplicación. Las pruebas automatizadas con Selenium, junto con la base de datos PostgreSQL, permitirán validar tanto la integridad de los datos como la interacción entre módulos clave, asegurando que la aplicación cumpla con los requisitos funcionales y no funcionales establecidos.

3. Estrategia de Pruebas

3.1. Desarrollo Guiado por Pruebas (TDD) Adaptado

Este enfoque híbrido comienza con el desarrollo de funcionalidades clave, priorizando aquellas basadas en las historias de usuario y los requisitos específicos. Durante las primeras etapas del desarrollo, se pueden escribir pruebas exploratorias para validar hipótesis o comportamientos básicos del sistema. A medida que la implementación se estabiliza, se desarrollan pruebas unitarias e integradas más detalladas, ajustadas al contexto real de la implementación, evitando la necesidad de reescribir pruebas cuando la implementación cambia significativamente.

Una vez que una funcionalidad ha sido completamente desarrollada, se procede a una validación exhaustiva mediante pruebas que cubren todos los casos principales. Estas pruebas aseguran que la funcionalidad cumpla con los requisitos especificados y que no introduzca regresiones en el sistema. Además, se automatizan pruebas clave con Selenium

para garantizar que las nuevas características o cambios en el código no afecten negativamente las funcionalidades existentes.

3.2. Pruebas Unitarias

El objetivo de las pruebas unitarias es asegurar que cada unidad de código funcione correctamente dentro de su propio contexto. Las pruebas se escriben después de haber desarrollado una funcionalidad clave o cuando se tiene una implementación estable, permitiendo ajustes según las necesidades y la evolución del código. Esto garantiza una validación precisa y relevante para cada componente individual, utilizando PostgreSQL como base de datos para verificar la correcta manipulación y almacenamiento de datos.

3.3. Pruebas de Integración

Las pruebas de integración se centran en verificar la correcta interacción entre los diferentes módulos del sistema. Una vez que se ha completado la integración de varias unidades, se redactan estas pruebas para asegurar que los módulos funcionen en conjunto de manera efectiva, validando los flujos de usuario y asegurando que la base de datos PostgreSQL maneje correctamente las transacciones y relaciones entre entidades. Las pruebas de integración se implementarán y completarán antes de la semana 8 para garantizar la estabilidad del sistema antes de las pruebas de aceptación.

3.4. Pruebas Manuales

Las pruebas manuales se enfocan en evaluar la usabilidad y consistencia de la interfaz de usuario. Estas pruebas se realizarán una vez que las funcionalidades principales hayan sido implementadas y validadas mediante pruebas automatizadas. El equipo revisará la experiencia del usuario en distintos dispositivos y navegadores, asegurando una interfaz intuitiva y accesible.

3.5. Pruebas Automatizadas

Las pruebas automatizadas serán un componente clave para garantizar la eficiencia y consistencia del proceso de pruebas. Usando Selenium, se automatizarán las pruebas de regresión, validación de flujos de usuario críticos y pruebas de interfaz de usuario. Estas pruebas se integrarán en el flujo de trabajo a medida que las funcionalidades se estabilicen y se completen las pruebas de integración, garantizando que cualquier cambio en el código sea verificado automáticamente, reduciendo el riesgo de introducción de errores.

3.6. Pruebas de Aceptación

Las pruebas de aceptación se llevarán a cabo en la semana 8, involucrando a usuarios finales o representantes del cliente para validar que la aplicación cumple con los requisitos y expectativas definidos. Estas pruebas son cruciales para asegurar que el producto final no solo funciona correctamente, sino que también satisface las necesidades del usuario y cumple con los estándares de calidad esperados.

4. Planificación y Cronograma

El plan de pruebas se desarrollará en las 11 semanas restantes del proyecto, distribuyéndose de la siguiente manera:

- **Semana 1-2:** Desarrollo inicial de funcionalidades clave, incluyendo pruebas exploratorias donde sea necesario.
- **Semana 3-4:** Ajuste y escritura de pruebas unitarias e integradas para funcionalidades estabilizadas, utilizando PostgreSQL y preparando la integración con Selenium para pruebas automatizadas.
- **Semana 5-6:** Implementación y ejecución de pruebas de integración, asegurando que los módulos interactúan correctamente entre sí y que los flujos de usuario operan según lo esperado.
- **Semana 7-8:** Finalización de pruebas de integración, automatización de pruebas con Selenium y ejecución de pruebas de aceptación con usuarios finales o stakeholders.
- **Semana 9-10:** Pruebas de regresión automatizadas con Selenium, asegurando que las nuevas implementaciones no afecten las funcionalidades existentes, y validación final de la aplicación.
- **Semana 11:** Cierre de pruebas, documentación de resultados y preparación para el lanzamiento, asegurando la calidad y estabilidad del código.

5. Criterios de Aceptación y Cierre

Para que una funcionalidad sea aceptada, todas las pruebas unitarias, de integración, automatizadas y de aceptación deben ser ajustadas y completadas conforme a la implementación final, asegurando que cubran al menos los casos críticos y que las funcionalidades principales operen sin problemas. El ciclo de pruebas se considerará cerrado cuando todas las pruebas estén completas y documentadas, y todos los defectos críticos hayan sido resueltos, garantizando que el software esté listo para su liberación.

6. Documentación

Los informes de defectos se registrarán y priorizarán en JIRA, asignando responsables para su corrección. Esta documentación asegura un seguimiento detallado y estructurado de todos los problemas encontrados durante el ciclo de pruebas, facilitando su resolución eficiente y manteniendo un alto estándar de calidad en el desarrollo del proyecto.