

Barbie Financiera

Versión 0.1

Martes, 15 de octubre de 2024

Ricardo José Cubias Sigarán

Genesis Beraly Parada Ventura

Daniela Rocío Pineda Pineda

Karla Melissa Torres Solórzano

Melanie Giovanna Valle Benavides

Prof. Patricia Rodríguez Polanco

Índice

Introducción	3
Objetivo del Plan de Pruebas	3
Alcance del Plan de Pruebas	4
Estrategia de Pruebas	6
Tipos de Pruebas	6
Enfoque de Pruebas.....	7
Criterios de Entrada y Salida	7
Planificación de Recursos	8
Roles y Responsabilidades.....	8
Recursos Necesarios	11
Cronograma de pruebas	13
Cronograma de Actividades.....	13
Hitos de las Actividades Claves: Fechas importantes y sus entregables.	14
Historias de Usuario.....	14
Riesgos y Mitigación	15
Identificación de Riesgos.....	15
Planes de Mitigación: Estrategias para reducir o manejar los riesgos.	16
Control de Cambios	16
Gestión de Cambios: Cómo se manejarán los cambios en los requisitos o en el plan de pruebas.	16
Aprobaciones y Revisiones: Proceso de revisión y aprobación de los cambios.....	16
Requisitos de Desarrollo de Pruebas.....	17
Entorno de Pruebas	17
Configuraciones Especiales	18
Conclusiones	18
Recomendaciones	18
Bibliografía	20
Anexos	21
Apariencia de la aplicación.....	21
Diagrama UML de Barbie Financiera.....	26
Glosario de Términos.....	26

Introducción

El presente documento, realizado para la materia titulada “Pruebas de software”, describe el enfoque y las actividades necesarias para realizar pruebas en la aplicación Barbie Financiera. La aplicación antes mencionada fue desarrollada en un ciclo de carrera previo cursado por los autores.

El objetivo general de la aplicación es permitir a los usuarios crear presupuestos mensuales según sus ingresos netos, ingresando sus gastos mensuales asignándolos a diferentes categorías según la preferencia de cada usuario, también, la aplicación permite también realizar reportes mensuales de los gastos e ingresos realizados.

Con este proyecto, se busca mejorar el trabajo previo a través de diferentes herramientas, especialmente desde el ángulo de pruebas. Ahora bien, para poder realizar estas actividades de la mejor forma posible, se consideró adecuado realizar este documento que engloba todo el proceso de planeación. Las pruebas expuestas garantizarán que la aplicación cumple con los requisitos funcionales y de calidad, y que ofrece una experiencia de usuario sin fallos críticos.

En resumen, este proceso de pruebas permitirá validar la funcionalidad y robustez de la aplicación, contribuyendo a que sea una herramienta confiable y eficiente para sus usuarios.

Objetivo del Plan de Pruebas

Descripción general del propósito del documento y sus metas.

El objetivo de este plan de pruebas es garantizar que el software cumpla con los requisitos tanto funcionales como no funcionales establecidos desde el inicio del proyecto. Se busca asegurar que todas las funcionalidades operen correctamente y que el sistema mantenga un alto nivel de calidad, revisando aspectos como la estabilidad, usabilidad y rendimiento bajo diferentes condiciones. Además, dado que el software maneja información financiera, es fundamental que garantice la protección de estos datos sensibles, minimizando riesgos para los usuarios finales. Uno de los propósitos clave es la identificación temprana de defectos que puedan comprometer la calidad del software. Para esto, se llevarán a cabo pruebas exhaustivas en distintas fases del desarrollo, con el fin de detectar problemas técnicos o funcionales que podrían derivar en errores críticos. Solucionar estos defectos a tiempo evitará complicaciones futuras, reduciendo costos y tiempos de corrección, y asegurando una experiencia más satisfactoria para los usuarios.

De esta manera, el plan busca optimizar los recursos de tiempo y presupuesto mediante una planificación y ejecución eficiente de las pruebas. Detectar fallos lo antes posible reducirá riesgos y gastos asociados con correcciones tardías, además de minimizar posibles retrasos en el lanzamiento del producto. De esta forma, pretendemos entregar un software confiable y de alta calidad dentro del plazo y presupuesto acordados.

Alcance del Plan de Pruebas

Como ya se mencionó anteriormente, el proyecto se trata de un software diseñado para ayudar a los usuarios a monitorear sus ingresos y egresos, con diferentes funcionalidades que facilitan este proceso. Para dar seguridad de que funcionen correctamente, se espera

cubrir y probar las siguientes funcionalidades, así mismo, se enlistan otras que no serán probados.

Funciones que se probarán:

- Registro e Inicio de sesión:
 - Se tiene que asegurar que todos los usuarios puedan ingresar y registrarse de manera eficiente al programa, realizando pruebas de autenticación.
- Creación de Presupuestos:
 - Verificar que los usuarios puedan crear presupuestos mensuales basados en diferentes rangos de ingresos.
- Cálculo de Gastos e Ingresos/ Visualización de Informes:
 - Evaluar que el cálculo sea preciso y claro y se pueda ver en una especie de informe.
- Interfaz de usuario:
 - Comprobar que el diseño sea fácil de usar y que sea visualmente atractivo.
- Seguridad:
 - Asegurarse que los datos financieros de los usuarios estén protegidos y sean almacenados en una base de datos.
- Funciones que no se probarán:
 - Compatibilidad con Múltiples Dispositivos:
 - No se probará que el diseño sea responsivo en diferentes pantallas.
- Integraciones con otros Servicios:
 - No se realizarán pruebas en integraciones con otros servicios, como bancos o sistemas financieros.

Con este plan se busca garantizar que las funcionalidades del software sean probadas para garantizar la integridad de este mismo. Las funcionalidades que no se probarán serán consideradas en futuras del proyecto.

Estrategia de Pruebas

Tipos de Pruebas

Para garantizar que Barbie Financiera cumpla con los requisitos especificados y funcione de manera óptima, se aplicarán los siguientes tipos de pruebas:

Pruebas Funcionales:

- Verificar que las funcionalidades principales (como la creación de presupuestos, el cálculo de ingresos y gastos) funcionen según lo especificado en los requisitos.
- Pruebas de validación de entradas: asegurarse de que los usuarios no puedan introducir datos erróneos (por ejemplo, caracteres no numéricos en campos de ingreso).

Pruebas No Funcionales:

- Verificar la usabilidad de la interfaz, asegurando que los usuarios puedan navegar y realizar acciones sin dificultad.
- Probar la seguridad del sistema, asegurando la protección de los datos personales y financieros.

Pruebas de Regresión:

- Asegurarse de que nuevas funcionalidades o correcciones de errores no afecten negativamente a otras áreas previamente verificadas.

Pruebas de Integración:

- Verificar la interacción entre diferentes áreas del sistema (por ejemplo, cómo interactúan los cálculos de presupuesto con las bases de datos de usuarios).

Pruebas Exploratorias:

- Realizar pruebas sin guiones predefinidos para descubrir errores que pueden no haber sido considerados en los casos de prueba iniciales.

Enfoque de Pruebas

Pruebas de Caja Negra:

- Se probará la funcionalidad del sistema sin conocer la estructura interna del código. Los casos de prueba estarán basados en los requisitos del sistema y escenarios de usuario.

Pruebas de Caja Blanca:

- Estas pruebas se aplicarán en módulos críticos del código, como los cálculos de presupuesto. Se examinará la estructura interna para asegurarse de que todos los caminos del código están siendo ejecutados correctamente.

Pruebas Exploratorias:

- Se permitirá a los evaluadores explorar la aplicación libremente para identificar posibles problemas no contemplados en los casos de prueba formales.

Criterios de Entrada y Salida

Criterios de Entrada

- Los requisitos funcionales deben estar completos y aprobados.
- La versión del software a probar debe estar disponible y desplegada en el entorno de pruebas.
- Las herramientas de prueba deben estar configuradas y listas para su uso.
- Los datos de prueba deben estar preparados y verificados.

Criterios de Salida

- Todos los casos de prueba deben haberse ejecutado al menos una vez.
- No deben quedar errores críticos abiertos.
- Los defectos encontrados deben estar registrados y priorizados para corrección.
- Se debe haber alcanzado un nivel de cobertura de pruebas aceptable en función de los objetivos del proyecto.

Planificación de Recursos

Roles y Responsabilidades

Descripción de los miembros del equipo y sus funciones.

Para que el proyecto se realice de la mejor forma posible, todo el equipo estará involucrado en cada parte del proceso. Sin embargo, no es una tarea fácil, por lo que se han asignado ciertos roles a cada miembro de forma que cada uno pueda desarrollar actividades y pruebas individualmente que luego se integren de la mejor forma posible.

Por otro lado, es importante mencionar que estos roles fueron elegidos a raíz de dos grandes categorías vistas en clase: líder de pruebas y testers. En cuanto a los tipos de testers, estos fueron escogidos en base a categorías de pruebas que consideramos más relevantes

realizar, así como luego de una investigación en fuentes externas que permiten tener ejemplos e ideas más amplias del proceso de prueba de software. Habiendo aclarado estos puntos, nuestro equipo está formado de la siguiente manera:

Líder de pruebas de software y tester regresión: Daniela Pineda.

Se decidió asignar un líder de pruebas pues se considera muy importante que exista un rol que se dedique a la coordinación de todas las pruebas realizadas por el equipo, con el fin de mantener el orden y comunicación de todos. Ahora bien, como se mencionaba anteriormente, es muy importante que cada persona esté involucrada en las diferentes etapas que se pasará en este proceso, es por esto por lo que se decidió que el líder también se hará cargo del rol de tester de regresión, que se refiere a la realización de todas aquellas pruebas que tienen que ver con comprobar que los elementos del sistema ya probados no se dañen con la incorporación de nuevos cambios. En esa misma línea, es importante mencionar que estos dos roles no fueron combinados al azar, se consideró que ya que el líder de pruebas es quien tiene una mejor idea de todas las pruebas que se están llevando a cabo podrá identificar y realizar las respectivas pruebas para asegurar que lo aprobado anteriormente mantiene su orden.

Tester de Usabilidad (UX Tester): Melissa Torres.

Al iniciar el desarrollo de esta aplicación teníamos muchas ideas respecto a cómo se quería que se viera. Sin embargo, a lo largo del proceso se enfrentó con un gran problema, y es que algunas de las funcionalidades que se habían establecido resultaron inútiles o poco importantes para los usuarios, mientras que otras que ni siquiera se habían tomado en consideración eran vitales para la experiencia de usuario. De esta experiencia, se extrajo

lo vital que es poner al usuario como centro de todas las actividades, pues no tiene sentido trabajar en exceso si al final todo ese progreso va a resultar en una pérdida de recursos. Además, la consideración de las pruebas de usabilidad en el producto genera un resultado de sistema fácil e intuitivo de usar, que reduce aspectos tediosos del desarrollo de software como lo es el soporte y formación para los usuarios. Debido a estas razones, se ha considerado vital asignarle a un miembro del equipo realizar todas las pruebas que tengan que ver con UX, teniendo como objetivo final la reducción de errores en la usabilidad del producto, así como su adopción y lealtad de parte de los usuarios, considerando otros factores como sistemas parecidos ya existentes en el mercado.

Tester de funcionalidad y exploratorio: Ricardo Cubias.

Esta posición se relaciona mucho con la anterior, tester de UX pues con este rol se busca poder validar que las funcionalidades determinadas realmente estén cumpliendo lo que se quiere así como también lo que el usuario final quiere. El tester UX y el tester de funcionalidad se diferencian esencialmente en que, el UX se enfoca en “cómo” el sistema hace lo que hace, mientras que el tester de funcionalidad en “qué”. Ahora bien, como se mencionaba, la funcionalidad y el UX se relacionan profundamente, esto debido a la conexión que se ha creado asignando al tester de funcionalidad el rol de tester exploratorio, el cual se refiere a probar y buscar formas creativas y quizá distintas a las pensadas originalmente, tanto de “qué hace” como de “cómo lo hace”.

Tester de Datos de Pruebas: Génesis Parada.

Este rol se refiere a la prueba del sistema a través de datos concretos, ya sean reales o estimados.

Tal vez, para otros tipos de sistemas estas pruebas no sean tan fundamentales como para

este proyecto; sin embargo, en este caso, todo el proyecto se basa en la recolección de datos y la exposición de estos de una forma útil para el usuario. Debido a esto, aprobar cualquier parte del software sin antes probarlo con datos reales o estimados resultaría un grave error. Saber cómo el sistema reacciona a los diferentes tipos y cantidades de datos que se ingresen ayudará a mejorar aspectos quizá más concretos de código y demás.

Tester de Aceptación de Usuarios (UAT Tester): Giovanna Valle.

Como ya se menciona en varias partes anteriores, el objetivo con este proyecto es poner al usuario al centro de todas las actividades que se realizan, pues se ha comprobado con experiencias anteriores que no hacerlo de esta forma resulta en un gasto inútil de recursos. Debido a esto, se considera que este rol es fundamental para obtener el éxito del proyecto, en este, se engloba muchas de las acciones descritas anteriormente y es uno de los últimos pasos antes de sacar al aire el sistema. Se ha pensado que el encargado de este rol será quien se encargue de mostrar el sistema a usuarios un tanto extraños para el equipo, de forma que estos puedan brindar feedbacks los más acertados posibles para pasarlos a todas las otras pruebas anteriormente descritas.

Recursos Necesarios

Herramientas, entornos de prueba, y cualquier recurso técnico o humano.

Herramientas:

Trello: los autores de este documento han contemplado el ciclo de vida del desarrollo de software en múltiples materias, incluida esta. Es por esto por lo que se sabe que para que el desarrollo se complete de la mejor forma posible, debemos utilizar una metodología, en

este caso, elegimos ágil y para trabajar en esta línea utilizaremos Trello para mejor distribución de las tareas.

Cypress: se trata de una herramienta de automatización de pruebas de software, si bien el equipo no cuenta con demasiada experiencia en la utilización de esta herramienta, se considera que la realización de este proyecto es una oportunidad perfecta para comenzar a conocerla con el objetivo de lograr la mejor aplicación posible.

Entornos de prueba:

Ambiente controlado local: se planea realizar todas las pruebas en las computadoras del equipo y también solicitar la colaboración en ciertas pruebas de compañeros en sus respectivas computadoras.

Recursos técnicos:

- Previa documentación de la aplicación: debido a que este se podría decir que es una continuación de un proyecto anterior, el equipo ya posee cierta documentación del desarrollo original de la aplicación que fue solicitada en forma de proyecto. Gracias a la seriedad con la que fue creada esta información, se sabe que será de mucha ayuda.
- Documentación de Python y base de datos correspondiente: todo programador sabe que sin esta documentación no se hace nada, siempre es necesario consultarla para verificar que algo está bien hecho o si se puede mejorar.
- Visual Studio Code: el editor de código de preferencia por consenso de todo el equipo.
- Recursos humanos
- Equipo de prueba/desarrollo: el equipo que cumple tanto funciones de testers como de desarrolladores de código.

- Profesores: el auxilio de profesores para las distintas etapas de desarrollo, especialmente en la fase de pruebas.
- Usuarios finales: con el fin de realizar todos los tipos de prueba, se planea solicitar la colaboración de personas externas al equipo de pruebas/desarrollo con el fin de obtener feedback que probablemente el equipo ya no es capaz de percibir.

Cronograma de pruebas

Cronograma de Actividades

Fase	Actividad	Duración (tiempo)	Fecha de Inicio	Fecha de Finalización	Responsable	Entregable
Planificación	Revisión del proyecto	2 días	13/10/2024	15/10/2024	Todo el equipo	Plan de Pruebas
Diseño de Pruebas	Diseño de casos de prueba	5 días	16/10/2024	21/10/2024	Todo el equipo	Casos de prueba basados en requerimientos y escenarios de usuario
Configuración del entorno	Configuración de entorno de pruebas	1 día	22/10/2024	22/10/2024	Líder de equipo	Informe de configuración de entorno
Ejecución de Pruebas	Funcionales, no funcionales, regresión, integración y exploratorias	5 días	23/10/2024	28/10/2024	Todo el equipo	Reporte de resultados de pruebas

Análisis y gestión de defectos	Registro y gestión de defectos	3 días	29/10/2024	31/10/2024	Todo el equipo	Documentación, análisis y seguimiento de defectos
Cierre de Pruebas	Revisión de resultados	0.5 días	01/11/2024	01/11/2024	Todo el equipo	Revisión final de los resultados de pruebas

Hitos de las Actividades Claves: Fechas importantes y sus entregables.

Planificación:

Es la primera fase de todas las actividades a realizar. En esta fase se identifican los objetivos bajo los que se trabajan y las funcionalidades que se pondrían a prueba. Sobre esto, se genera el plan de pruebas que permite que el resto de las fases sean posibles.

Ejecución de pruebas:

Esta fase permitirá que se descubra todos los defectos o cosas a mejorar tanto del código como de funcionalidades estéticas y prácticas.

Historias de Usuario

- Como usuario de Barbie Financiera, quiero ingresar mis ingresos netos mensuales y asignar montos a categorías de gastos (alquiler, alimentación, entretenimiento, etc.) para crear un presupuesto personalizado que me ayude a gestionar mis finanzas de manera efectiva.
- Como usuario, quiero poder modificar los montos de las categorías de mi presupuesto actual para ajustarlo cuando mis ingresos o gastos cambien.

- Como usuario, quiero ver un resumen de mi presupuesto, incluyendo cuánto he gastado y cuánto me queda disponible por categoría para llevar un control claro de mis finanzas.
- Como usuario, quiero poder comparar mis presupuestos actuales con los de meses anteriores para analizar cómo han cambiado mis hábitos de gasto y mejorar mi planificación financiera.
- Como usuario, quiero poder observar cuando mis gastos en una categoría sobrepasen el presupuesto asignado, para poder tomar acciones y controlar mis finanzas de manera oportuna.

Riesgos y Mitigación

Identificación de Riesgos

Durante la fase de pruebas de Barbie Financiera, existen algunos riesgos potenciales que podrían afectar la calidad o el cronograma de las pruebas:

- Problemas de integración: La aplicación puede presentar problemas al integrar áreas clave como los cálculos financieros con la base de datos o con otras herramientas, lo que podría afectar la precisión de los resultados.
- Cambios de última Hora en los Requisitos: Cambios inesperados en los requisitos funcionales o no funcionales cerca del final del proceso de desarrollo podrían afectar negativamente las pruebas, requiriendo reevaluación de casos de prueba y replanteamiento de estrategias.
- Recursos insuficientes para la Ejecución de Pruebas: Limitaciones en el número de personas asignadas al equipo de pruebas o en las herramientas de prueba disponibles podrían retrasar la ejecución y finalización del proceso de pruebas.

Planes de Mitigación: Estrategias para reducir o manejar los riesgos.

- Problemas de Integración: Se realizarán pruebas de integración tempranas y continuas para identificar y corregir problemas de interacción entre herramientas desde el inicio.

Control de Cambios**Gestión de Cambios: Cómo se manejarán los cambios en los requisitos o en el plan de pruebas.**

Para incorporar cualquier modificación sugerida o cambio a los requisitos, estos deberán atravesar un estricto proceso de revisión, garantizando así con seguridad la transparencia y la trazabilidad del proyecto. Todos los cambios serán analizados en grupo, pues la meta es cuantificar tanto su efecto como su peligro, generando de esta manera una visión nítida y la posibilidad de tomar decisiones acordadas. Así, una vez que cada modificación sea aprobada, se notificará a todos los participantes y/o implicados en el proyecto, con el fin de coordinar el trabajo y esfuerzo de todos hacia un objetivo compartido.

Aprobaciones y Revisiones: Proceso de revisión y aprobación de los cambios.

Todos los cambios sugeridos deberán atravesar un proceso de evaluación en el que se involucrarán los integrantes fundamentales del equipo de pruebas, administración de proyectos y otros interesados. Una vez evaluado el efecto del cambio o el riesgo que pueda traer, se pedirán autorizaciones formales de los involucrados a través de encuentros o procedimientos digitales. Ninguna modificación se llevará a cabo sin la aprobación explícita de todos los participantes. Este procedimiento garantiza que las modificaciones sean claras y que todos los integrantes del equipo estén al tanto con los cambios.

Requisitos de Desarrollo de Pruebas

Entorno de Pruebas

Un entorno de prueba según algunos usuarios de LinkedIn es una réplica del entorno de producción, donde el software será implementado y utilizado por los usuarios finales y para este proyecto se necesita ambientar el hardware, el software y algunas configuraciones necesarias de la siguiente manera:

- Hardware: Para crear un entorno de pruebas ideal se necesita tener las mismas especificaciones de procesamiento, memoria y almacenamiento en comparación con el entorno de producción, esto según algunos sitios web consultados. Para ello necesitamos como mínimo los siguientes requisitos:
 - Intel i5 o superiores.
 - Al menos 8GB de RAM.
 - Un disco duro de 256GB par almacenamiento eficiente.
 - Conexión a internet estable para garantizar la correcta ejecución de prueba.
- Software: El software debe incluir un sistema operativo compatible con la aplicación. Este entorno debe de tener los siguientes requisitos:
 - Sistema operativo como mínimo de Windows 10 o mayor
 - Servidor de base de datos tentativo de MariaDB 10.4
 - Entornos de desarrollo como Visual Studio Code.

Configuraciones Especiales

- Seguridad y Autenticación: Se implementarán mecanismos de autenticación para asegurarnos que el usuario este ingresando en su cuenta y no en la de otro, y así garantizar la seguridad de los datos financieros de cada persona.
- Datos de Prueba: Utilizaremos datos ficticios para imitar un usuario modelo para que se asemeje a un caso real que la aplicación vaya a manejar. Se generarán con diferentes tipos de usuarios con diferentes roles y perfiles de acceso a fin de verificar la integridad del sistema.

Conclusiones

- Implementar las pruebas permite identificar errores en etapas tempranas del desarrollo, minimizando los costos y gastos innecesarios a futuro.
- La combinación de pruebas funcionales y no funcionales permiten dar seguridad de que el software cumple con requerimientos técnicos y de experiencia de usuario
- Integrar a los usuarios en las pruebas permite obtener retroalimentación valiosa para mejorar las funcionalidades.
- Trabajar mediante metodologías ágiles optimiza la distribución de tareas y comunicación entre el equipo.

Recomendaciones

- Seguir aplicando pruebas de regresión e integración de manera periódica.

- Fomentar que el equipo se capacite en herramientas que faciliten llevar a cabo las pruebas.
- Incluir pruebas que no fueron realizadas en esta fase, como la compatibilidad con diversos dispositivos

Bibliografía

Estrategia de pruebas. (s. f.). ¿Cómo se definen y documentan los requisitos del entorno de prueba? LinkedIn. <https://www.linkedin.com/advice/1/how-do-you-define-document-your-test-environment?lang=es&originalSubdomain=es>

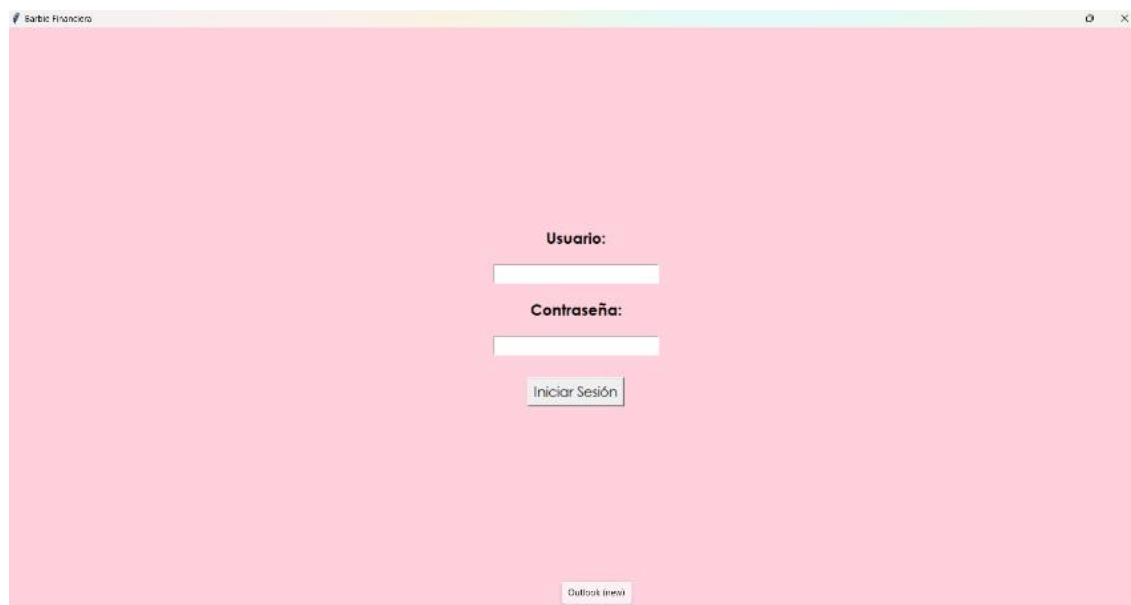
Jonic, N. (2024, Octubre). El Alcance de una Prueba de Software. Test IO Academy. <https://academy.test.io/es/articles/2683803-el-alcance-de-una-prueba-de-software#>

Mohit, V. Barate, P. (s. f.). ¿Cómo se puede determinar el alcance de las pruebas de un sistema de software? LinkedIn. <https://www.linkedin.com/advice/0/how-can-you-determine-scope-testing-software-lofjc?lang=es&originalSubdomain=es>

Anexos

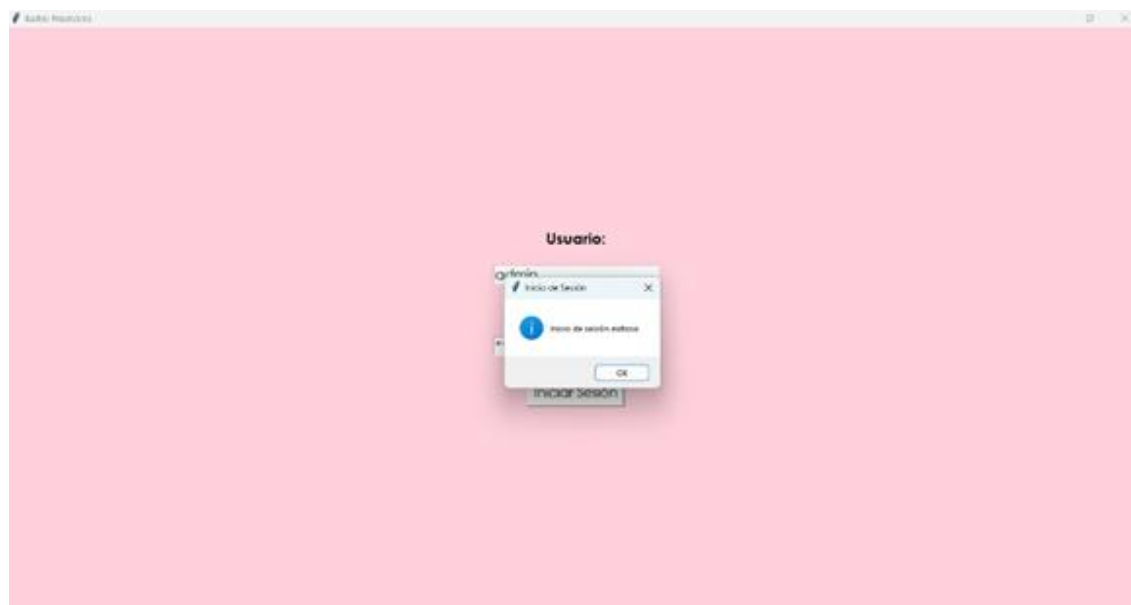
Apariencia de la aplicación

Figura 1. Inicio de sesión



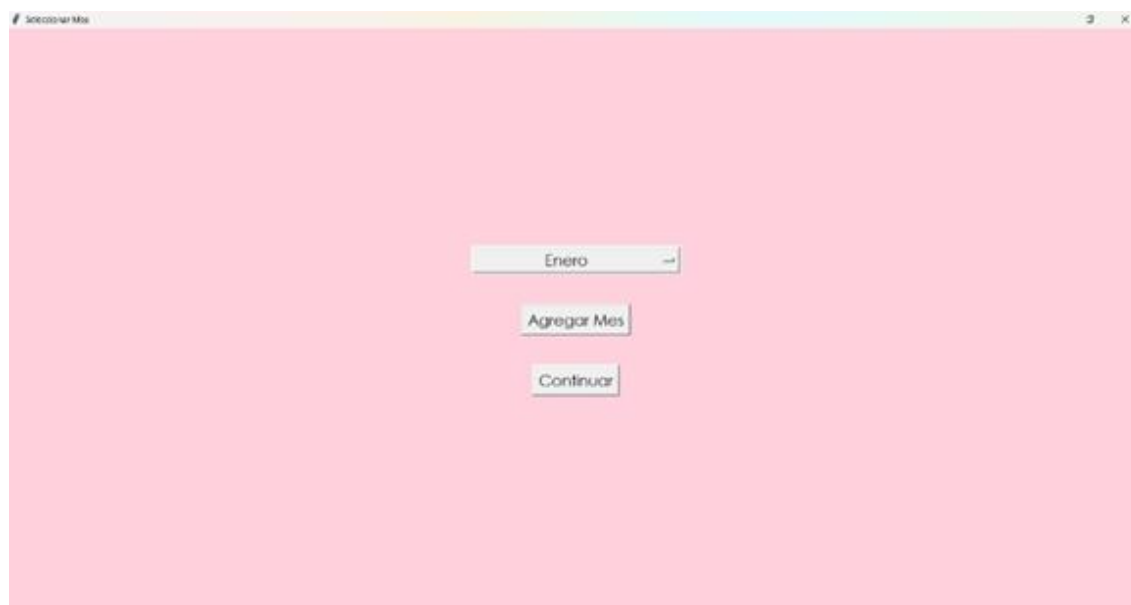
Fuente. Elaboración propia

Figura 2. Validación de inicio de sesión



Fuente. Elaboración propia

Figura 3. Selección de mes de presupuesto



The screenshot shows a web application window titled "Seleccionar Mes". The background is a solid light pink color. In the center, there is a form with three elements: a dropdown menu displaying "Enero", a button labeled "Agregar Mes", and a button labeled "Continuar".

Fuente. Elaboración propia

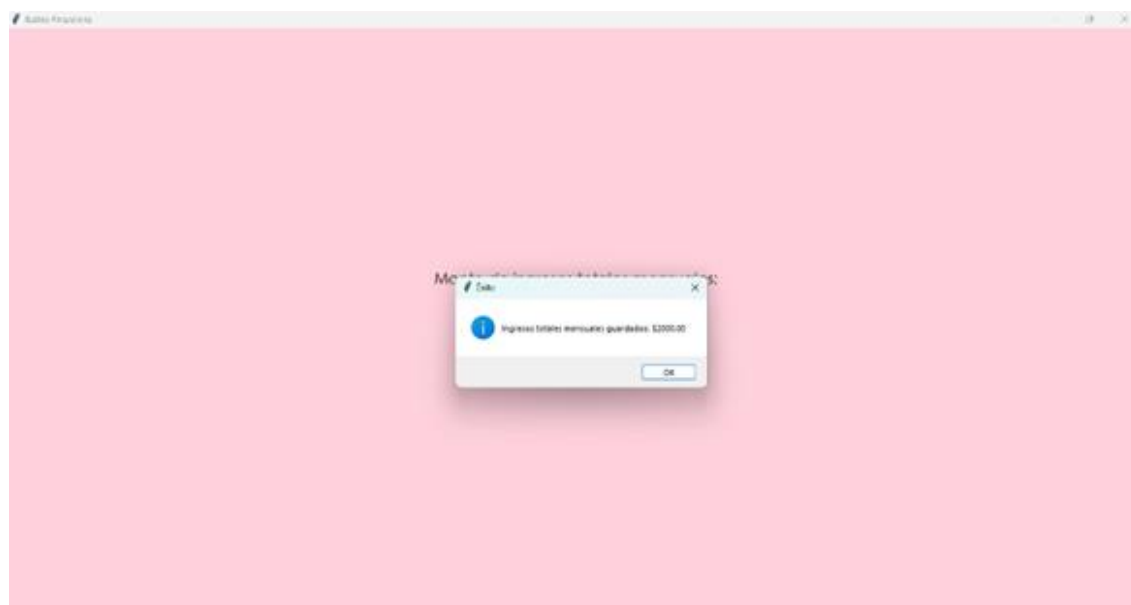
Figura 4. Monto de ingresos mensuales



The screenshot shows a web application window titled "Monto de Ingresos". The background is a solid light pink color. In the center, there is a form with two elements: a text input field containing the number "2000" and a button labeled "Continuar". Above the input field, the text "Monto de ingresos totales mensuales:" is displayed.

Fuente. Elaboración propia

Figura 5. Validación de ingresos



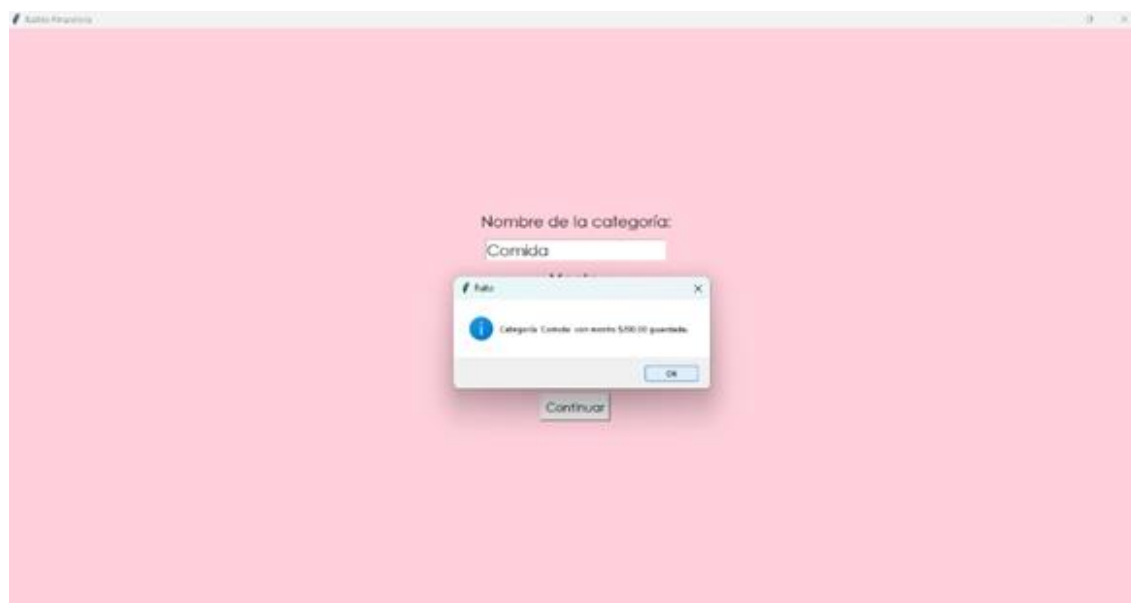
Fuente. Elaboración propia

Figura 6. Ingreso de gastos por categorías



Fuente. Elaboración propia

Figura 7. Validación de gasto.



Fuente. Elaboración propia

Figura 8. Menú de opciones



Fuente. Elaboración propia

Figura 9. Balance General



Fuente. Elaboración propia

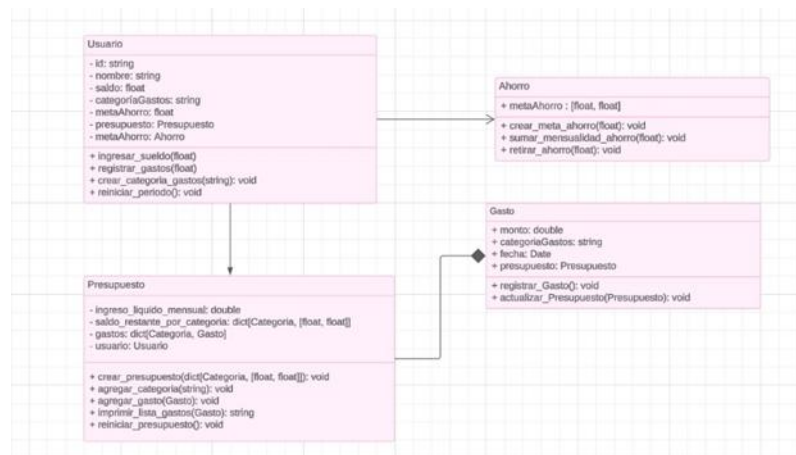
Figura 10. Detalle de gastos



Fuente. Elaboración propia

Diagrama UML de Barbie Financiera

Figura 11. Diagrama UML



Fuente. Elaboración propia

Glosario de Términos

- Caja Blanca: Técnica de prueba en la que se conoce la estructura interna del sistema o código que se está probando.
- Caja Negra: Técnica de prueba donde se evalúa la funcionalidad de un sistema sin conocer su estructura interna.
- Cobertura de Pruebas: Medida utilizada para describir el alcance de las pruebas ejecutadas en un sistema en relación con el número de condiciones de prueba.
- Criterios de Entrada: Condiciones que deben cumplirse antes de comenzar una actividad de prueba.
- Criterios de Salida: Condiciones que deben cumplirse para finalizar una actividad de prueba.
- Testers: Personas encargadas de realizar las pruebas de software, identificando errores y sugerencias de mejora.

- Cypress: Herramienta de automatización utilizada para realizar pruebas de software, especialmente pruebas de interfaz de usuario (UI)