26-5-2024

Proyecto Final

Mantenimiento De Software

Carlos Antonio Arce Mendivil
ING SISTEMAS COMPUTACIONAELS

Contenido

Resumen de las Técnicas de Mantenimiento Aplicadas y Resolución de Detalles Encontrados

Proyecto: Configuración y mantenimiento de software

Introducción

El código trata de una simulación de un cajero automático tiene como objetivo proporcionar una aplicación funcional que permita realizar operaciones básicas como retiro y transferencia de dinero. Durante el desarrollo del proyecto, se aplicaron varias técnicas de mantenimiento de software para garantizar la calidad y mantenibilidad. Este documento resume las técnicas de mantenimiento aplicadas y los detalles corregidos.

Técnicas de Mantenimiento Aplicadas

Refactorización del Código

La refactorización implica reorganizar y limpiar el código existente sin alterar su funcionalidad externa. Esta técnica mejora la legibilidad, reduce la complejidad y facilita el mantenimiento futuro.

Se reorganizó el código para separar las responsabilidades en diferentes clases, como Banco, Cliente, Cuenta, Transacción, y Cajero Automático. Esto hizo que cada clase tuviera una responsabilidad única y clara, mejorando el modularidad del sistema.

Encapsulamiento

El encapsulamiento es una práctica de programación que protege los datos sensibles y controla el acceso a los atributos de las clases mediante métodos get y set.

Casi todos los atributos de las clases se hicieron privados, y se proporcionaron métodos get y set para acceder y modificar esos atributos. Esto aseguró que los datos solo pudieran ser modificados de manera controlada, previniendo errores y aumentando la seguridad.

Validación de Datos

La validación de datos es crucial para asegurar que el sistema funcione correctamente con entradas válidas y manejadas adecuadamente.

Se implementaron validaciones para asegurar que los montos de retiro y transferencia no excedieran el saldo disponible. Esto previno errores operacionales y mejoró la robustez del sistema.

Manejo de Errores

El manejo adecuado de errores mejora la robustez del sistema y proporciona una mejor experiencia al usuario.

Se implementaron controles para manejar situaciones como saldo insuficiente y opciones de menú inválidas. Esto permitió proporcionar mensajes claros y útiles al usuario, guiándolos para corregir sus entradas.

Control de Versiones con Git

El control de versiones permite gestionar los cambios en el código fuente, facilitando la colaboración y el mantenimiento del historial de modificaciones.

Se utilizó Git para el control de versiones. Se crearon ramas para diferentes funcionalidades y se hizo merge a la rama principal después de completar y revisar los cambios. Esto permitió una gestión organizada y segura del código fuente.

Pruebas de Funcionalidad

Las pruebas de funcionalidad aseguran que el sistema funcione según lo esperado antes de ser entregado.

Se realizaron pruebas manuales para cada funcionalidad del cajero automático, como retiros, transferencias y consultas de saldo. Esto garantizó que el sistema respondiera correctamente en cada caso.

Conclusión

La aplicación de estas técnicas de mantenimiento resultó en un sistema de cajero automático modular y fácil de mantener.

La refactorización y encapsulamiento mejoraron la estructura del código, mientras que la validación de datos y el manejo de errores aseguraron la fiabilidad del sistema.

El uso de control de versiones facilitó la colaboración y el seguimiento de cambios, y las pruebas de funcionalidad garantizaron que el sistema funcionara correctamente.

Estas mejoras no solo corrigieron los detalles encontrados, sino que también prepararon el sistema para futuras extensiones y mantenimientos.

Links GitHub:

Repositorio de prueba:

https://github.com/AntonioArce123/Proyecto_Terminado

(Abrir el READMI.md)