Cajero automático:

Especificación de requisitos

(SRS)

Índice

1. Historial de versiones del documento

Revisor	Estado	Descripción	Autores	Fecha
Gamaliel quiroz	No iniciada	Final Programación Avanzada	Gianni Baldizzone Victoria Troiano	08/03/2024

2. Objetivo

El objetivo de este documento es establecer los requisitos funcionales y no funcionales del sistema de cajero automático, cumpliendo con los criterios establecidos en la consigna final.

3. Beneficios

El sistema permitirá a los usuarios realizar operaciones bancarias de manera segura y eficiente, incluyendo la gestión de cuentas, consulta de saldo, retiro y depósito de efectivo, transferencia de fondos, cambio de PIN, y cierre de sesión. Esto proporcionará comodidad y facilidad de acceso a los servicios bancarios.

4. Alcance

El sistema de cajero automático abarca todas las funcionalidades descritas en la consigna final, incluyendo la apertura y cierre de sesión, gestión de cuentas, operaciones bancarias y seguridad de acceso. El sistema estará disponible para su uso mediante su interfaz gráfica, desarrollada con WindowsBuilder.

5. Limitaciones

-El aplicativo es local, no se conecta a internet ni consume recursos de él.

-El aplicativo únicamente funciona en computadoras, no está adaptado a móvil.

-El aplicativo no se complementa con otro software

-Las funciones del aplicativo están acotadas a lo que se puede desarrollar en el lenguaje de programación JAVA.

6. Requisitos

6.1. Requisitos funcionales

Apertura de conexión con la base de datos:

-El sistema debe ser capaz de establecer una conexión con la base de datos al iniciar la aplicación.

-Esta funcionalidad debe ser responsabilidad del componente Manager Connection.

Iniciar sesión:

-Los usuarios deben poder ingresar su número de cuenta y su PIN para acceder al sistema.

-El sistema debe autenticar las credenciales proporcionadas por el usuario y permitir el acceso si son válidas.

Consulta de saldo:

-Una vez iniciada la sesión, los usuarios deben poder verificar el saldo de su cuenta bancaria.

-El sistema debe recuperar el saldo de la cuenta bancaria del usuario desde la base de datos.

Retiro de efectivo:

-Los usuarios deben poder solicitar un retiro de efectivo ingresando la cantidad deseada.

-El sistema debe verificar si hay suficientes fondos en la cuenta para realizar el retiro solicitado.

-En caso afirmativo, el sistema debe dispensar el efectivo correspondiente y actualizar el saldo de la cuenta en la base de datos.

Depósito de efectivo:

-Los usuarios deben poder depositar efectivo en su cuenta bancaria.

-El sistema debe permitirles ingresar la cantidad de efectivo a depositar y reflejarla en el saldo de la cuenta en la base de datos.

Transferencia de fondos:

-Los usuarios deben poder transferir fondos de su cuenta a otra cuenta bancaria.

-El sistema debe permitirles ingresar el número de cuenta de destino y la cantidad a transferir.

-El sistema debe verificar si hay suficientes fondos en la cuenta para realizar la transferencia solicitada.

-En caso afirmativo, el sistema debe actualizar el saldo de ambas cuentas involucradas en la base de datos.

Cambio de PIN:

-Los usuarios deben tener la opción de cambiar su PIN actual por uno nuevo, para mantener la seguridad de su cuenta.

-El sistema debe permitir al usuario ingresar su PIN actual y el nuevo PIN deseado.

-El sistema debe validar el PIN actual y actualizarlo en la base de datos si es válido.

Salida del sistema:

-Los usuarios deben poder cerrar sesión y salir del sistema de manera segura.

-El sistema debe cerrar la conexión con la base de datos antes de finalizar la sesión.

6.2. Requisitos no funcionales

Diagramado por capas:

- El sistema debe estar diseñado siguiendo una arquitectura de capas, separando claramente la lógica de presentación, la lógica de negocio y el acceso a datos.

Manejo de base de datos:

-El sistema debe utilizar una unidad de persistencia de datos, ya sea una base de datos relacional o archivos indexados, para almacenar y recuperar la información de las cuentas bancarias y las transacciones.

Manejo de actividad y transacciones:

-El sistema debe gestionar las actividades de los usuarios de manera eficiente y segura, garantizando la integridad de los datos y la consistencia de las transacciones realizadas.

Herencia, interfaces, polimorfismo:

-El código Java del sistema debe hacer uso de conceptos de programación orientada a objetos como herencia, interfaces y polimorfismo para facilitar la modularidad y la reutilización del código.

Interfaz de usuario:

-El sistema debe proporcionar una interfaz de usuario intuitiva y amigable, que pueda ser utilizada tanto con Scanner por consola como con interfaz gráfica mediante Swing.

Diagrama de casos de uso —> **Pendiente**