Universidad de San Carlos de Guatemala
Escuela de Ciencias y Sistemas
Facultad de Ingeniería Introducción a la programación y Computación 1
Primer Semestre 2025
Catedrático: Herman Igor Veliz Linares
Auxiliar: Lesther Kevin Federico López Miculax

PROYECTO 1

USAC INTERNATIONAL BANK

Axel David González Molina
202402074
Fecha de entrega:

16/3/2025

Introducción

El proyecto "USAC International Bank" tiene como objetivo desarrollar una aplicación de simulación de un sistema bancario, diseñada para facilitar la comprensión de los conceptos fundamentales de la programación orientada a objetos y la creación de interfaces gráficas en Java. Este sistema permitirá a los usuarios interactuar con diversas funcionalidades típicas de una entidad bancaria, como la gestión de cuentas, la realización de depósitos y retiros, y la generación de reportes sobre transacciones.

La aplicación está estructurada bajo el patrón Modelo-Vista-Controlador (MVC), lo que garantiza una separación adecuada de las responsabilidades y mejora la mantenibilidad del código. A lo largo del desarrollo del proyecto, se implementarán validaciones rigurosas para asegurar la integridad de los datos y la correcta ejecución de las operaciones, así como una interfaz amigable que permita a los usuarios navegar con facilidad por las diferentes funcionalidades disponibles.

Este manual técnico documenta los requerimientos del sistema, la arquitectura del software, y proporciona una descripción detallada de las funcionalidades implementadas, así como las pruebas realizadas para garantizar el correcto funcionamiento del sistema. Se espera que este proyecto no solo sirva como una herramienta de aprendizaje para los estudiantes, sino que también les brinde una experiencia práctica en el desarrollo de aplicaciones de escritorio.

Manual técnico

A continuación se describirán los métodos creados y requerimientos de la aplicación.

Requerimientos:

- Los requerimientos generales son:
    - Tamaño máximo de usuarios en el sistema → 50.
    - En ningún momento y por ninguna razón el saldo de la cuenta deberá ser negativo.
    - En ningún momento y por ninguna razón una transacción tendrá valores 0 en el "monto debitado" o en el "monto acreditado" al mismo tiempo. Así como tampoco deberá tener valores mayores 0 al mismo tiempo. Estos campos deberán ser mutuamente exclusivos: uno u otro, pero no ambos

Requerimientos por apartado:

- **Inicio de Sesión**:
    - Validar credenciales de usuario predefinido.
    - Mensajes de error para credenciales incorrectas.
- **Gestión de Usuarios**:
    - Registro de nuevos clientes con validación de CUI.
    - Restricción de creación de clientes según la cantidad permitida por sección.
- **Gestión de Cuentas**:
    - Creación de cuentas asociadas a clientes.
    - Validación de saldo para depósitos y retiros.
    - Registro y visualización de transacciones.

- **Reportes**:
    - Generación de reportes de transacciones, depósitos y retiros en formato PDF.
    - Nombres de archivos con fecha y hora de generación.
- **Bitácora**:
    - Registro de actividades del sistema con detalles de cada acción.
- **Seguridad**:
    - Protección de datos sensibles (CUI, contraseñas).
    - Validaciones para evitar transacciones no autorizadas.
- **Arquitectura**:
    - Descripción del patrón MVC (Modelo-Vista-Controlador) utilizado.
    - Diagramas de clases y flujo del sistema.

Métodos creados:

Para el desarrollo de la aplicación se trabajó con el patrón MVC (Modelo-Vista-Controlador) y se crearon varios métodos auxiliares para poder simplificar y trabajar de manera más ordenada y eficiente el código, a continuación se mencionaran algunos de los métodos principales para el desarrollo del programa.

- **Registrar Cliente (RegistrarUsuario_1):** Registra un nuevo cliente en la base de datos con sus datos personales.
- **Crear Cuenta (RegistroCuenta_2):** Genera una nueva cuenta bancaria asociada a un cliente recién registrado.
- **Actualizar Tabla de Cuentas (BuscarCuentas_3):** Llena automáticamente la tabla de cuentas cuando un cliente se registra.
- **Validar Datos del Cliente:** Verifica que los datos ingresados cumplan con los requisitos necesarios antes del registro.
- **Buscar Cuentas por Cliente:** Permite recuperar todas las cuentas asociadas a un cliente en la base de datos.
- **guardarUsuario(ModeloRegistroUsuario usuario):**Agrega el usuario a la lista estática listaClientes.
- **btnRegistrarActionPerformed():** Valida datos (CUI, nombre, formato de correo). Llama a guardarUsuario() para almacenar en ArrayList<ModeloRegistroUsuario>.
- **btnBuscarCuentasActionPerformed()**:Obtiene el CUI ingresado y Verifica existencia con listaClientes. Llama a obtenerCuentasAsociadas() para mostrar en tblcuentasAsociadas.
- **agregarcuentas(String cui, String cuenta)**:Añade cuentas nuevas (hasta 3 por usuario) y Muestra alertas con JOptionPane.
- **btnDepositarActionPerformed()**:Valida que el monto sea positivo y Actualiza saldo en una estructura de datos.
- **mostrarConfirmacion()**: Muestra diálogo con detalles del depósito.
- **Depositos()**: Campos para número de cuenta y monto.
- **CrearCuenta()**: Incluye combobox para tipo de cuenta (Ahorro/Corriente).
- **Login ()**: Inicializa campos para usuario/contraseña.

Conclusiones

El desarrollo del proyecto "USAC International Bank" ha permitido aplicar y consolidar los conocimientos adquiridos en el curso de Introducción a la Programación y Computación. A través de la implementación de un sistema bancario simulado, se ha logrado comprender la importancia de la programación orientada a objetos y su aplicabilidad en el desarrollo de software real.

Se ha demostrado que la separación de responsabilidades mediante el patrón MVC no solo facilita la organización del código, sino que también mejora la capacidad de mantenimiento y escalabilidad del sistema. Las validaciones implementadas han asegurado que las operaciones realizadas sean seguras y confiables, minimizando el riesgo de errores y garantizando la integridad de los datos.

Además, la creación de una interfaz gráfica amigable ha sido fundamental para mejorar la experiencia del usuario, permitiendo una interacción fluida con el sistema. La generación de reportes y la bitácora de actividades han añadido un nivel adicional de funcionalidad y trazabilidad, lo que es esencial en cualquier aplicación bancaria.