Actividad |3| Programa 2 (parte 2).

Desarrollo de Aplicaciones Móviles IV.

Ingeniería en Desarrollo de Software.

TUTOR: Marco Alonso Rodriguez Tapia.

ALUMNO: Ramón Ernesto Valdez Felix.

FECHA: 29/11/2025.

Introducción.

En esta actividad final de la materia Desarrollo de Aplicaciones Móviles IV, nos enfocaremos en el Banco Mexicano que requiere el desarrollo de una aplicación fundamental de banca en línea utilizando el lenguaje de programación Swift para ofrecer servicios básicos a sus clientes. Este sistema digital es crucial para modernizar las operaciones y permitir a los usuarios gestionar su dinero de forma remota. En una fase inicial, se implementó la estructura principal del menú y las funcionalidades de "Depósito" y "Salir". La etapa actual de desarrollo se centrará en completar las funcionalidades esenciales pendientes: "Retiro" y "Saldo".

La clave de esta implementación será garantizar una experiencia de usuario robusta y segura. Para la opción de Retiro, el programa deberá validar el saldo disponible del cliente y manejar escenarios donde la cuenta esté vacía o el monto solicitado exceda los fondos. Por su parte, la opción Saldo proporcionará una consulta inmediata y clara del estado financiero. Ambos módulos incluirán prompts para la continuidad, asegurando un flujo de navegación eficiente en el sistema.

Descripción.

Este proyecto final de Desarrollo de Aplicaciones Móviles IV se centra en completar una aplicación fundamental de banca en línea para el Banco Mexicano, utilizando el lenguaje Swift. El objetivo es modernizar los servicios y permitir a los clientes gestionar su dinero de forma remota.

Tras haber implementado la estructura del menú, el Depósito y la opción Salir, la fase actual se dedica a añadir las funcionalidades esenciales restantes: Retiro y Saldo. La implementación prioriza una experiencia de usuario robusta y segura.

La función de Retiro exige una validación estricta del saldo, manejando errores si la cuenta no tiene fondos o si el monto excede el saldo actual. La opción Saldo debe mostrar de manera clara e inmediata el estado financiero del usuario. Ambos módulos están diseñados para ser eficientes, incluyendo

indicaciones para que el usuario decida si desea continuar con más operaciones. Esto garantiza un flujo de navegación lógico y completo dentro del sistema bancario.

Justificación.

La implementación de la banca en línea para el Banco Mexicano en Swift se justifica por la necesidad crítica de modernización y eficiencia operativa. En el contexto actual, ofrecer a los clientes la capacidad de gestionar sus finanzas de forma remota (depositar, retirar y consultar saldos) es esencial para la competitividad y la satisfacción del usuario. La actividad se centra en finalizar el núcleo transaccional del sistema, específicamente las opciones Retiro y Saldo, las cuales son vitales para la utilidad de la aplicación.

La justificación reside en garantizar la lógica de negocio y la seguridad de los fondos. La función de Retiro debe ser robusta, incluyendo validaciones obligatorias para evitar sobregiros y manejar la experiencia del usuario.

Al completar estas funciones, el programa pasa de ser una interfaz estática a una herramienta financiera funcional y confiable. Esto no solo cumple con los requisitos del banco, sino que también proporciona una base sólida para futuras expansiones del servicio, asegurando que las operaciones financieras se realicen con precisión y una interacción clara con el usuario.

Desarrollo.

En esta segunda parte de la materia Desarrollo de Aplicaciones Móviles IV, realizaremos el programa solicitado. Documentaremos el desarrollo de la aplicación, describiendo cada uno de sus requisitos. La aplicación debe incluir un menú de cuatro opciones de la aplicación de banco de méxico las cuales son (Depósito, Retiro, Saldo, Salir), así como un menú interactivo. Cada una de las funciones de la aplicación será la evidencia que se adjuntará a la documentación, e incluirá una breve explicación de su propósito.

5

Link: GitHub.

Codificación.

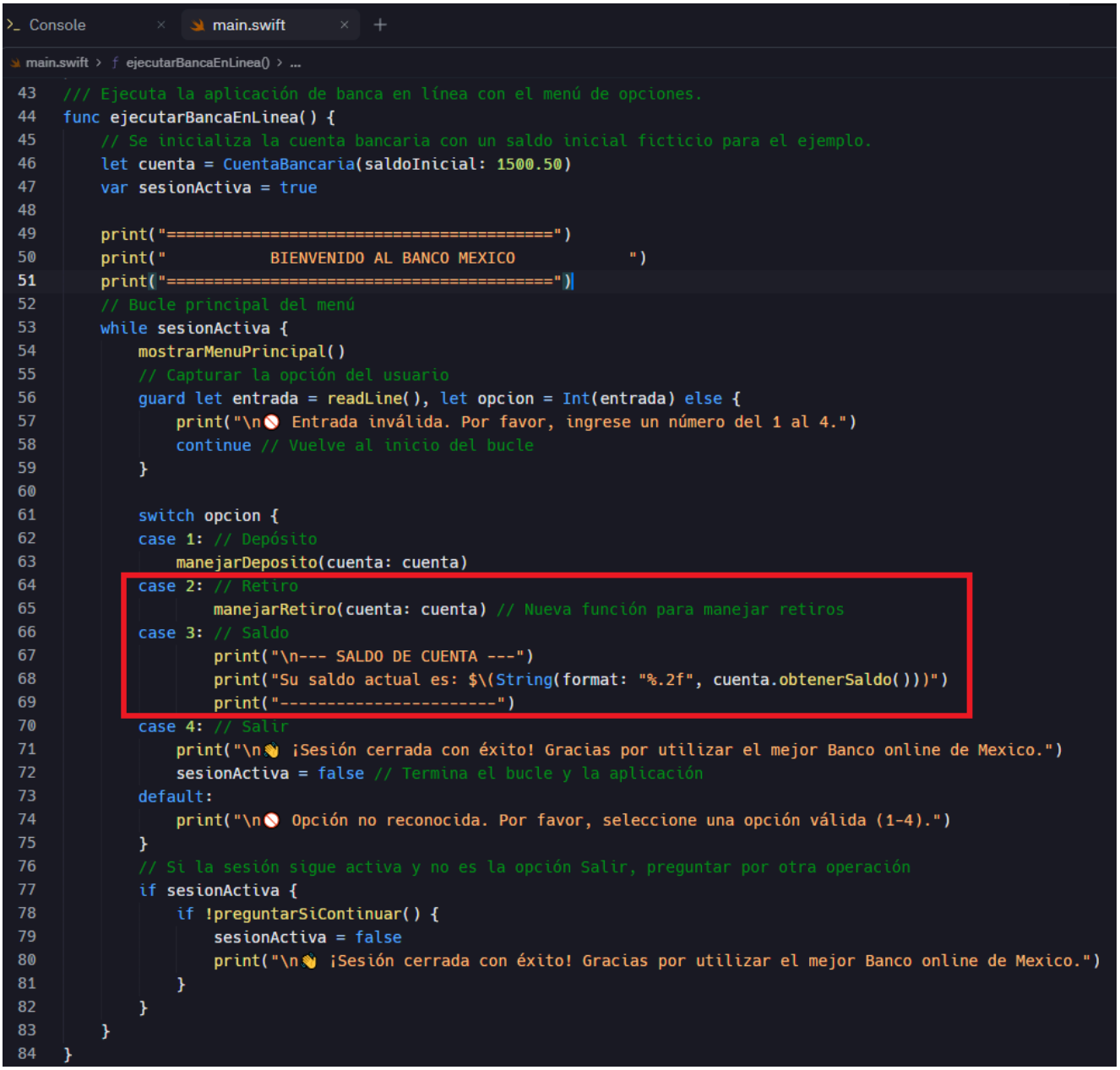
En el código Swift se implementa una simulación simple de un cajero de banco mexicano en línea. Su función principal es realizar la función de retiro y saldo de la aplicación dando funcionalidad a dos opciones del menú que nos pide la actividad final de la app del banco de mexico parte 2.

Console | main.swift

main.swift › ƒ ejecutarBancaEnLinea() › ...

```swift
import Foundation
// CLASE de nombre CuentaBancaria
/// Representa una cuenta bancaria con funcionalidades básicas.
class CuentaBancaria {
    // Variable para almacenar el saldo actual de la cuenta, inicializado en 0.
    private var saldo: Double = 0.0
    // Función para obtener el saldo actual (para futuros usos, como la opción 3)
    func obtenerSaldo() -> Double {
        return saldo
    }
    // Inicialización
    init(saldoInicial: Double = 0.0) {
        self.saldo = saldoInicial
    }
    // Operaciones
    /// Realiza un depósito en la cuenta.
    /// Parameter cantidad: La cantidad a añadir al saldo.
    func depositar(cantidad: Double) {
        if cantidad > 0 {
            saldo += cantidad
            print("\n✅ ¡Depósito exitoso! Se han depositado $\(String(format: "%.2f", cantidad))")
            print("Nuevo saldo: $\(String(format: "%.2f", saldo))")
        } else {
            print("\n❌ Error: La cantidad a depositar debe ser positiva.")
        }
    }
    // Función de Retiro (Implementación básica para completar la estructura)
    func retirar(cantidad: Double) -> Bool {
        if cantidad > 0 && saldo >= cantidad {
            saldo -= cantidad
            print("\n✅ ¡Retiro exitoso! Se han retirado $\(String(format: "%.2f", cantidad))")
            print("Nuevo saldo: $\(String(format: "%.2f", saldo))")
            return true
        } else if cantidad <= 0 {
            print("\n❌ Error: La cantidad a retirar debe ser positiva.")
            return false
        } else {
            print("\n❌ Error: Saldo insuficiente. Saldo actual: $\(String(format: "%.2f", saldo))")
            return false
        }
    }
}
```

En esta parte del código define un objeto (class) llamado CuentaBancaria en Swift, que encapsula la nformación (saldo) y las acciones (depositar/retirar) relacionadas con una cuenta.



La porción del código que has compartido define la función central que ejecuta y controla el flujo de toda tu aplicación de banca: el menú principal y la interacción con el usuario e inicializando el saldo con un valor de $1500.

Esencialmente, esta función ejecutarBancaEnLinea() es el motor del programa.

- Inicialización de la Cuenta.
- Bucle principal del menú.
- Captura y validación de la entrada.
- Selección de operación.
- Continuidad de la sesión.

Adicional se anexa la evidencia del código agregado para el funcionamiento correcta de la aplicación enmarcando en un color rojo.

Console | main.swift

main.swift > ejecutarBancaEnLinea() > ...

```swift
}
/// Muestra las opciones del menú principal.
func mostrarMenuPrincipal() {
    print("\n--- MENÚ DE OPERACIONES ---")
    print("1. Depósito")
    print("2. Retiro")
    print("3. Saldo")
    print("4. Salir")
    print("----------------------------")
    print("Ingrese su opción (1-4): ", terminator: "")
}
/// Pregunta si el usuario desea realizar otra operación (genérica) y devuelve un booleano.
func preguntarSiContinuar() -> Bool {
    while true { // El bucle se repite hasta que haya una respuesta válida
        print("\n----------------------------")
        print("¿Desea realizar otra operación? (Sí/No): ", terminator: "")

        guard let respuesta = readLine()?.lowercased().trimmingCharacters(in: .whitespacesAndNewlines) else {
            continue
        }
        if respuesta == "si" || respuesta == "sí" {
            return true
        } else if respuesta == "no" {
            return false
        } else {
            print("🚫 Respuesta inválida. Por favor, ingrese 'Sí' o 'No'.")
        }
    }
}
/// Pregunta si el usuario desea realizar otro depósito o retiro.
func preguntarPorOtraOperacion(tipo: String) -> Bool {
    var respuestaValida = false
    var deseaContinuar = false

    while !respuestaValida {
        print("¿Desea realizar otro \(tipo)? (Sí/No): ", terminator: "")
        let respuesta = readLine()?.lowercased().trimmingCharacters(in: .whitespacesAndNewlines) ?? ""

        if respuesta == "si" || respuesta == "sí" {
            deseaContinuar = true
            respuestaValida = true
        } else if respuesta == "no" {
            deseaContinuar = false
            respuestaValida = true
        } else {
            print("\n⚠️ Respuesta no válida. Por favor, escriba 'Sí' o 'No'.")
        }
    }
    return deseaContinuar
}
```

El código que presentamos está compuesto por tres funciones auxiliares en Swift, cuyo propósito principal es manejar la interacción con el usuario en la aplicación de banca en línea, específicamente en la presentación del menú y en la gestión de la continuidad de las operaciones.

Console | main.swift

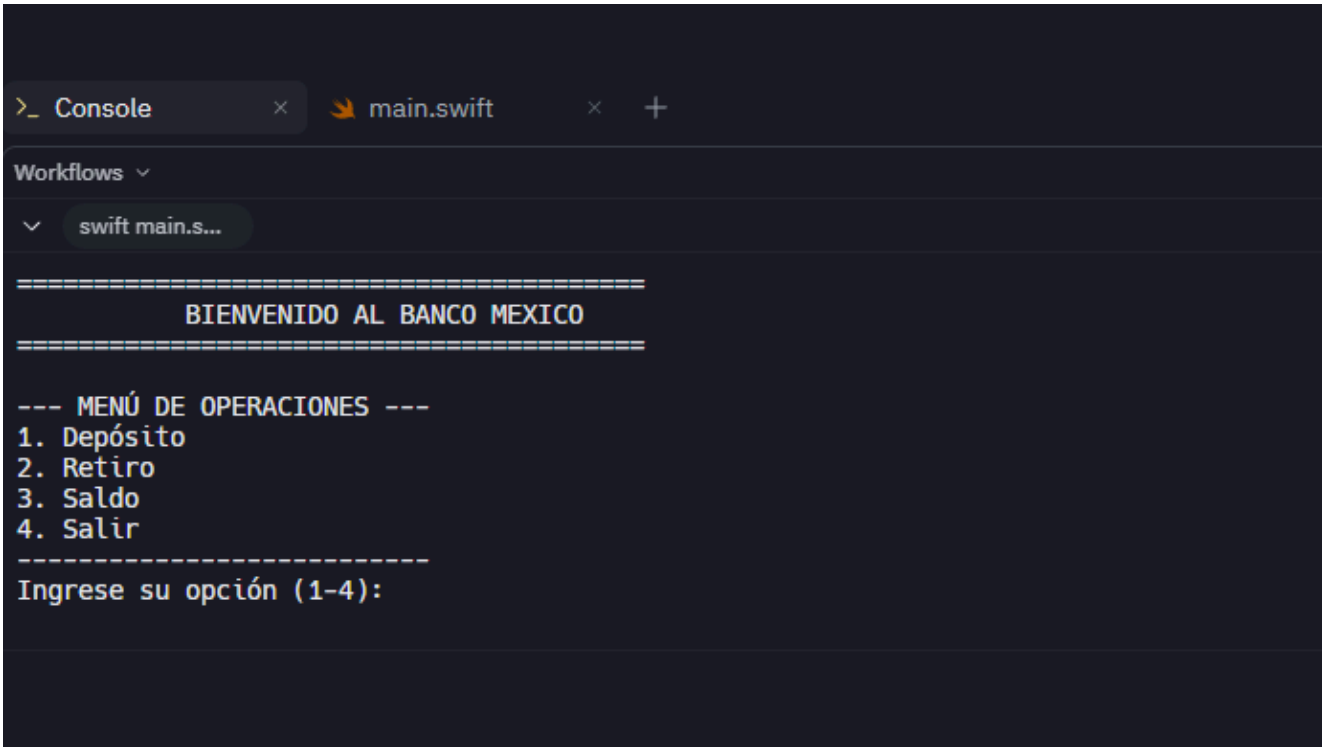
main.swift > ejecutarBancaEnLinea() > ...

```swift
}
/// Maneja el flujo de depósitos, incluyendo depósitos múltiples.
func manejarDeposito(cuenta: CuentaBancaria) {
    var deseaOtroDeposito = true
    while deseaOtroDeposito {
        print("\n--- OPERACIÓN DE DEPÓSITO ---")
        print("Ingrese la cantidad a depositar: $", terminator: "")
        // Capturar y validar la cantidad
        guard let entradaCantidad = readLine(),
              let cantidad = Double(entradaCantidad),
              cantidad > 0
        else {
            print("\n❌ Error: Cantidad inválida. Debe ser un número positivo.")
            // Usar la función auxiliar corregida
            deseaOtroDeposito = preguntarPorOtraOperacion(tipo: "depósito")

            continue
        }
        // Llama al método depositar de la clase CuentaBancaria.
        cuenta.depositar(cantidad: cantidad)
        // Preguntar si desea realizar otro depósito y validar la respuesta
        deseaOtroDeposito = preguntarPorOtraOperacion(tipo: "depósito")
    }
}
/// Maneja el flujo de retiros, incluyendo retiros múltiples.
func manejarRetiro(cuenta: CuentaBancaria) {
    var deseaOtroRetiro = true
    while deseaOtroRetiro {
        print("\n--- OPERACIÓN DE RETIRO ---")
        print("Ingrese la cantidad a retirar: $", terminator: "")
        // Capturar y validar la cantidad
        guard let entradaCantidad = readLine(),
              let cantidad = Double(entradaCantidad),
              cantidad > 0
        else {
            print("\n❌ Error: Cantidad inválida. Debe ser un número positivo.")
            // Usar la función auxiliar corregida
            deseaOtroRetiro = preguntarPorOtraOperacion(tipo: "retiro")

            continue
        }
        // Llama al método retirar de la clase CuentaBancaria.
        let _ = cuenta.retirar(cantidad: cantidad)
        // Preguntar si desea realizar otro retiro y validar la respuesta
        deseaOtroRetiro = preguntarPorOtraOperacion(tipo: "retiro")
    }
}
// Iniciar la aplicación
ejecutarBancaEnLinea()
```

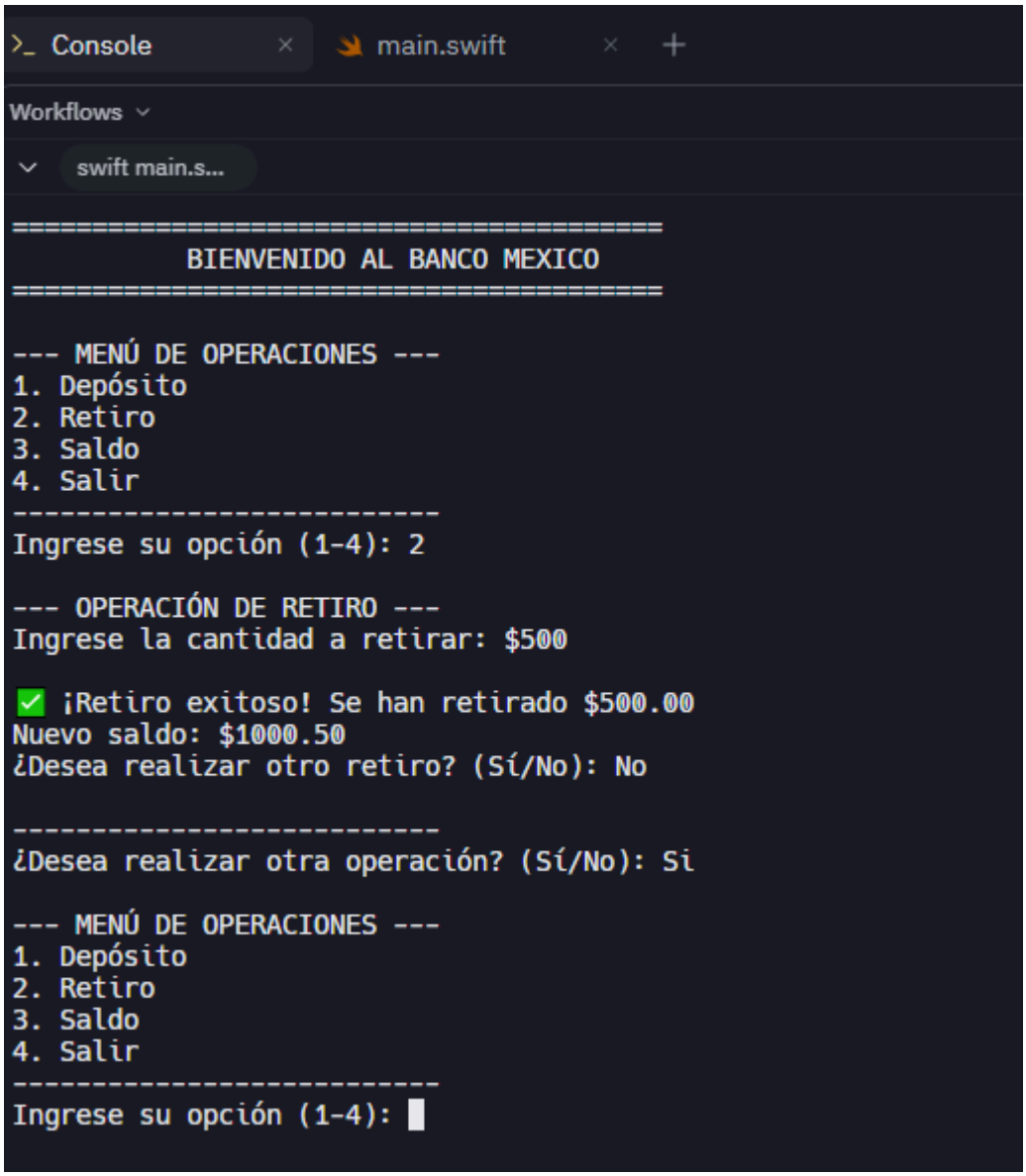
El código proporcionado define dos funciones clave, manejarDeposito y manejarRetiro, que gestionan la lógica de las transacciones financieras en la aplicación de banca en línea. Ambas funciones siguen un patrón similar para asegurar una interacción fluida y repetible con el usuario.

Prueba del Programa.

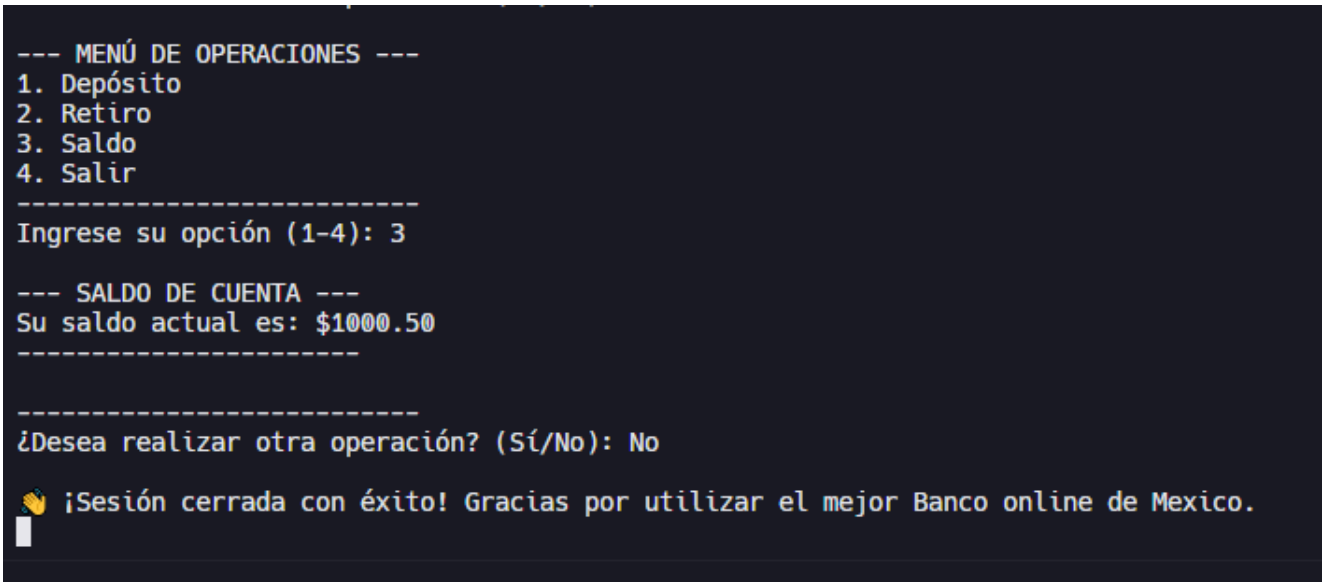
En esta etapa realizaremos la prueba del funcionamiento de la ejecución del código Swift anterior donde anexaremos los screenshot de evidencia del funcionamiento correcto de aplicación del banco de méxico en línea y solamente mostraremos el funcionamiento de la parte dos de la actividad final.



En esta parte de la ejecución nos muestra la estructura del menú principal del banco de méxico en línea.



En esta parte de la ejecución el flujo de la segunda opción de retiro en donde realizamos un retiro de $500 mostrando la ejecución de manera correcta y realizan la pregunta si desea seguir retirando en el caso de ser si regresa al menú de cantidad a retirar sino se muestra una segunda pregunta que dice si se desea hacer otra operación .



A continuación en esta captura de evidencia ingresamos a la opción 3 que es consultar saldo donde nos muestra en pantalla la cantidad de saldo con el cual cuenta el cliente y realiza una pregunta si desea realizar otra operación, si es que si regresa al menú para que elijas la operación y si se decide no continuar sale de la app del banco de méxico.

Conclusion.

En conclusión: La creación de una aplicación bancaria utilizando Swift representa una importancia dual, tanto en el ámbito laboral como en la vida cotidiana.

En el ámbito laboral, el desarrollo de estas aplicaciones consolida la demanda de ingenieros de software especializados en Swift (para iOS/macOS), una habilidad altamente valorada en la industria tecnológica. Esto impulsa la capacitación en desarrollo móvil y la especialización en arquitecturas seguras. Además, para la institución financiera, la app se convierte en una ventaja competitiva crucial, reduciendo costos operativos y extendiendo su alcance de servicio 24/7.

En la vida cotidiana, la aplicación transforma la gestión financiera, ofreciendo comodidad y acceso inmediato a servicios esenciales como depósitos, retiros y consulta de saldo, eliminando la necesidad de visitar sucursales. La elección de Swift garantiza que la aplicación sea rápida, segura y nativa en

dispositivos Apple, cumpliendo con los altos estándares de rendimiento y seguridad que los usuarios esperan para el manejo de su dinero. Así, el desarrollo en Swift no es solo tecnología, sino un factor clave para la digitalización de la economía y la autonomía financiera personal.

Referencias.

Google. (n.d.). Gemini. Retrieved November 29, 2025, from https://gemini.google.com/