Actividad | 2 |

Programa 2 (parte 1)

Desarrollo de Aplicaciones Móviles IV

Ingeniería en Desarrollo de Software

TUTOR: Marco Antonio Rodríguez

ALUMNO: Yanira Lizbeth Lopez Navarro

FECHA: 09/08/2024

Índice

Introducción

En la era digital actual, la banca en línea se ha convertido en una herramienta esencial para la gestión financiera personal. Los clientes demandan cada vez más accesibilidad y conveniencia, lo que impulsa a las instituciones bancarias a desarrollar soluciones innovadoras y eficientes. En este contexto, el Banco Mexicano busca crear una aplicación móvil de banca en línea utilizando el lenguaje de programación Swift. Esta aplicación permitirá a los usuarios realizar operaciones bancarias fundamentales, como depósitos, retiros y consultas de saldo, directamente desde sus dispositivos móviles.

La actividad planteada consiste en desarrollar un programa en Swift que ofrezca un menú interactivo con opciones claras y funcionales para cada tipo de operación bancaria. Además, se enfatiza la importancia de implementar validaciones y confirmaciones para garantizar la seguridad y la precisión de las transacciones. Este enfoque no solo mejorará la experiencia del usuario, sino que también proporcionará una plataforma segura y confiable para la gestión de sus cuentas bancarias.

A lo largo de esta actividad, se detallará el proceso de creación de esta aplicación, destacando los beneficios de utilizar Swift y la relevancia de ofrecer soluciones digitales eficientes en el sector bancario. La implementación de esta solución busca responder a las necesidades modernas de los clientes, fortaleciendo la relación entre el banco y sus usuarios al proporcionar una herramienta que combine conveniencia, seguridad y eficiencia.

Descripción

El Banco Mexicano necesita desarrollar una aplicación móvil de banca en línea utilizando el lenguaje de programación Swift. Esta app permitirá a los usuarios realizar operaciones esenciales como depósitos, retiros y consultas de saldo, además de ofrecer la opción de salir del sistema. La finalidad es proporcionar a los clientes una herramienta accesible y segura para gestionar sus finanzas personales desde sus dispositivos móviles.

La actividad consiste en crear un programa en Swift con un menú interactivo que incluya las siguientes opciones: Depósito, Retiro, Consulta de Saldo y Salir. Para la opción de Depósito, el usuario deberá ingresar la cantidad a depositar y el sistema preguntará si desea realizar otro depósito. Si no, le preguntará si desea realizar otra operación. En la opción de Retiro, si es la primera vez que el usuario intenta retirar dinero y no tiene saldo, el sistema debe informarle que no cuenta con fondos suficientes. Si tiene saldo, podrá ingresar la cantidad a retirar, y el sistema verificará si hay fondos suficientes, mostrando un mensaje si no es así. Después de cada retiro, se preguntará si desea realizar otro o hacer otra operación. En la Consulta de Saldo, el usuario podrá ver su saldo actual, y la opción Salir permitirá cerrar la sesión de manera segura.

Esta aplicación facilitará la gestión de cuentas bancarias de manera eficiente y segura, mejorando la experiencia del usuario y la accesibilidad a los servicios del banco.

Justificación

Implementar una aplicación de banca en línea utilizando Swift es una decisión estratégica y técnicamente sólida para el Banco Mexicano. Swift es un lenguaje de programación moderno y robusto, desarrollado por Apple, que se destaca por su eficiencia, seguridad y capacidad para crear aplicaciones rápidas y con un rendimiento óptimo en dispositivos iOS. Al desarrollar la aplicación en Swift, se asegura que los clientes del banco tengan una experiencia de usuario fluida y confiable en sus iPhone y iPads.

Además, una aplicación de banca en línea proporciona a los usuarios la comodidad de gestionar sus finanzas en cualquier momento y desde cualquier lugar. Esta accesibilidad es crucial en el mundo actual, donde los clientes valoran la posibilidad de realizar transacciones rápidas y seguras sin tener que visitar una sucursal bancaria física. La estructura del menú interactivo, que incluye opciones claras para depósitos, retiros, consultas de saldo y salir, facilita una navegación intuitiva. Las validaciones de saldo y las confirmaciones de operaciones mejoran la seguridad y la usabilidad, reduciendo la posibilidad de errores y asegurando que los usuarios estén siempre informados sobre el estado de sus cuentas.

Esta solución no solo mejora la experiencia del cliente al proporcionar una plataforma accesible y segura, sino que también fortalece la relación del banco con sus clientes al adaptarse a las demandas modernas de conveniencia y eficiencia.

Desarrollo
Codificación

El código comienza importando la biblioteca Foundation, que proporciona funcionalidades básicas esenciales para el desarrollo. Luego, se define la clase BankAccount, que representa una cuenta bancaria. Dentro de esta clase, se declara una propiedad balance de tipo Double, que almacena el balance de la cuenta. El inicializador init (balance: Double) permite crear nuevas instancias de BankAccount con un balance inicial especificado. La asignación self. balance = balance dentro del inicializador asegura que el valor del parámetro balance se asigne a la propiedad balance de la instancia actual de la clase

```
import Foundation

class BankAccount {
    var balance: Double

    init (balance: Double) {
        self. balance = balance
    }
```

La función deposit (amount: Double) en la clase BankAccount permite añadir una cantidad específica al balance de la cuenta. Al recibir un valor amount de tipo Double, la función incrementa el balance actual sumando esta cantidad (balance += amount). Después de actualizar el balance, imprime un mensaje que confirma la cantidad depositada y muestra el balance actualizado de la cuenta.

```
    func deposit (amount: Double) {
        balance += amount
        print ("Depósito de \(amount) completado. Balance actual: \(balance)")
    }
```

La función withdraw (amount: Double) -> Bool permite retirar una cantidad específica del balance de la cuenta bancaria. Primero, verifica si la cantidad a retirar (amount) es menor o igual al balance actual (balance). Si es así, resta la cantidad del balance, muestra un mensaje confirmando el retiro y el balance actualizado, y devuelve true para indicar que la transacción fue exitosa. Si la cantidad excede el balance, muestra un mensaje de fondos insuficientes y devuelve false, indicando que la transacción no pudo completarse.

```
    func withdraw (amount: Double) -> Bool {
        if amount <= balance {
            balance -= amount
            print ("Retiro de \(amount) completado. Balance actual: \(balance)")
            return true
        } else {
            print ("Fondos insuficientes. Balance actual: \(balance)")
            return false
        }
    }
}
```

La clase OnlineBanking se encarga de gestionar la interacción con una cuenta bancaria. Dentro de esta clase, se define una propiedad constante account de tipo BankAccount, que representa la cuenta bancaria con la que se trabajará. El inicializador init (account: BankAccount) recibe una instancia de BankAccount y la asigna a la propiedad account. Esto permite que cada instancia de OnlineBanking esté vinculada a una cuenta bancaria específica, facilitando la gestión de operaciones como depósitos y retiros.

```
class OnlineBanking {
    let account: BankAccount

    init (account: BankAccount) {
        self. account = account
    }
```

La función showMenu () en la clase OnlineBanking muestra un menú interactivo para el usuario. Inicialmente, establece la variable hasLoggedInBefore como false, para rastrear si el usuario ha realizado una operación de retiro antes. En un bucle infinito (while true), imprime un menú con opciones para depósito o retiro. Luego, lee la entrada del usuario y la convierte en un número entero (option). Según la opción seleccionada, llama a las funciones handleDeposit () para depósitos o handleWithdrawal (hasLoggedInBefore: hasLoggedInBefore) para retiros. Si la opción no es válida, muestra un mensaje de error. El bucle continúa ejecutándose para permitir múltiples operaciones hasta que el programa sea terminado.

```
func showMenu () {
    var hasLoggedInBefore = false

    while true {
        print ("""
        Bienvenido a la Banca en Línea
        1. Depósito
        2. Retiro
        Elige el número de la opción:
        """)

        if let choice = readLine (), let option = Int(choice) {
            switch option {
            case 1:
                handleDeposit ()
            case 2:
                handleWithdrawal (hasLoggedInBefore: hasLoggedInBefore)
                hasLoggedInBefore = true
            default:
                print ("Opción inválida. Por favor, intente nuevamente.")
            }
        } else {
            print ("Entrada inválida. Por favor, intente nuevamente.")
        }
    }
}
```

La función handleDeposit () gestiona el proceso de depósito en la cuenta bancaria. En un bucle infinito (while true), solicita al usuario que ingrese la cantidad a depositar. Si el usuario proporciona una entrada válida que puede convertirse a un número decimal (Double), se realiza el depósito mediante la función account. deposit (amount: amount). Si la entrada no es válida, muestra un mensaje de error y continúa solicitando una cantidad válida. Después del depósito, pregunta al usuario si desea realizar otro depósito. Si el usuario responde "n" (o "N"), el bucle se interrumpe y se sale de la función.

```
    private func handleDeposit () {
        while true {
            print ("Ingresa la cantidad a Depositar:")
            if let amountStr = readLine (), let amount = Double(amountStr) {
                account. deposit (amount: amount)
            } else {
                print ("Entrada inválida.")
                continue
            }

            print ("¿Deseas realizar otro depósito? s/S n/N")
            if let choice = readLine ()?lowercased (), choice == "n" {
                break
            }
        }
    }
```

La función handleWithdrawal (hasLoggedInBefore: Bool) gestiona el proceso de retiro de dinero de la cuenta bancaria. Primero, verifica si el usuario no ha realizado un retiro antes (hasLoggedInBefore es false) y si el balance de la cuenta es cero. Si ambas condiciones se cumplen, muestra el mensaje "No cuentas con saldo" y termina la función con return.

Si el usuario ha iniciado sesión previamente o la cuenta tiene saldo, entra en un bucle infinito (while true) donde solicita la cantidad a retirar. Si la entrada es válida y puede convertirse en un número decimal (Double), intenta realizar el retiro mediante account. withdraw (amount: amount). Si el retiro es exitoso, pregunta al usuario si desea realizar otro retiro. Si el usuario responde "n" (o "N"), el bucle se interrumpe. Si el retiro no es exitoso o la entrada es inválida, se muestra un mensaje de error y el bucle continúa solicitando una nueva cantidad.

```
    private func handleWithdrawal (hasLoggedInBefore: Bool) {
        if! hasLoggedInBefore && account. Balance == 0 {
            print ("No cuentas con saldo")
            return
        }
```

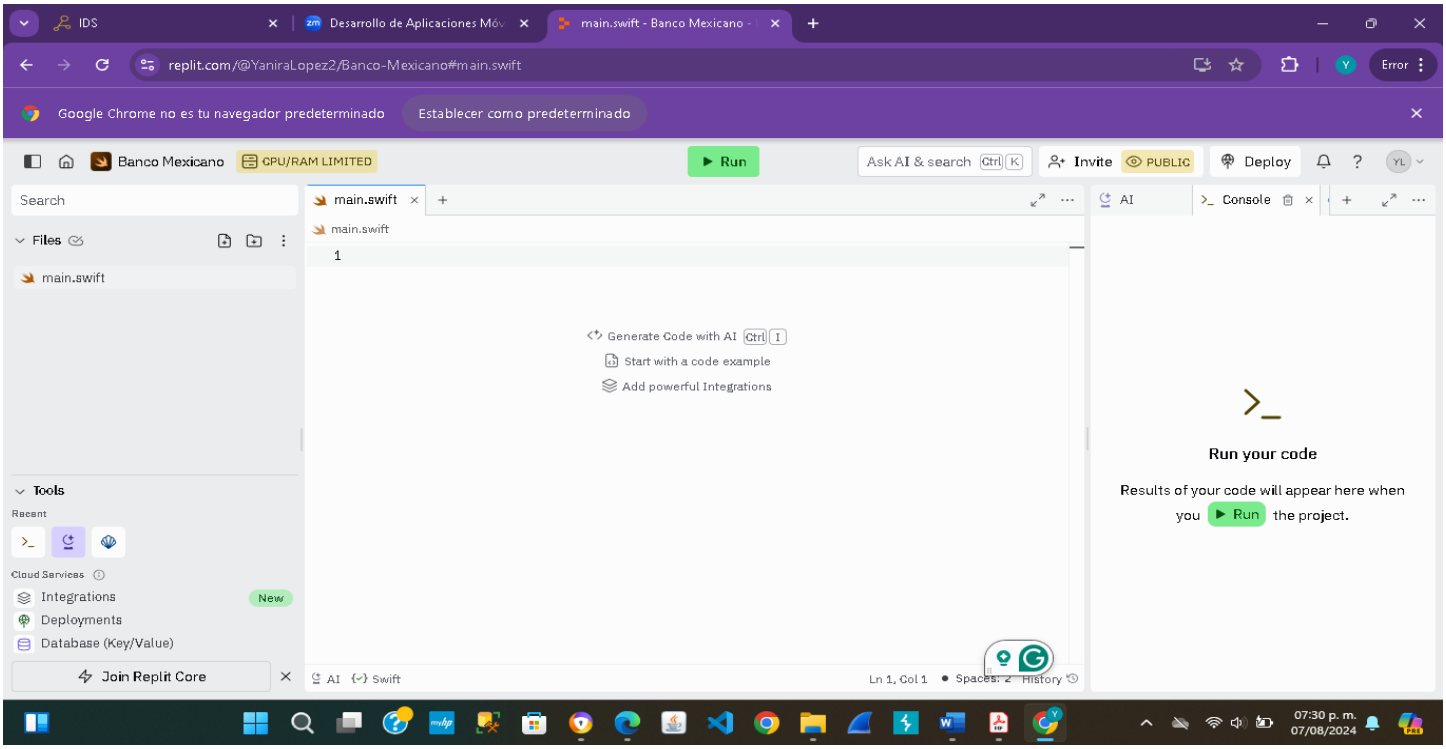
```
        while true {
            print ("Ingresa cantidad a retirar:")
            if let amountStr = readLine (), let amount = Double(amountStr) {
                let success = account. withdraw (amount: amount)
                if success {
                    print ("¿Deseas realizar otro retiro? s/S n/N")
                    if let choice = readLine ()?lowercased (), choice == "n" {
                        break
                    }
                } else {
                    break
                }
            } else {
                print ("Entrada inválida.")
            }
        }
    }
}
```

Finalmente, las líneas de código fuera de la función crean una instancia de BankAccount con un balance inicial de 0.0 y una instancia de OnlineBanking vinculada a esta cuenta. Luego, se llama a showMenu () para iniciar el menú interactivo.

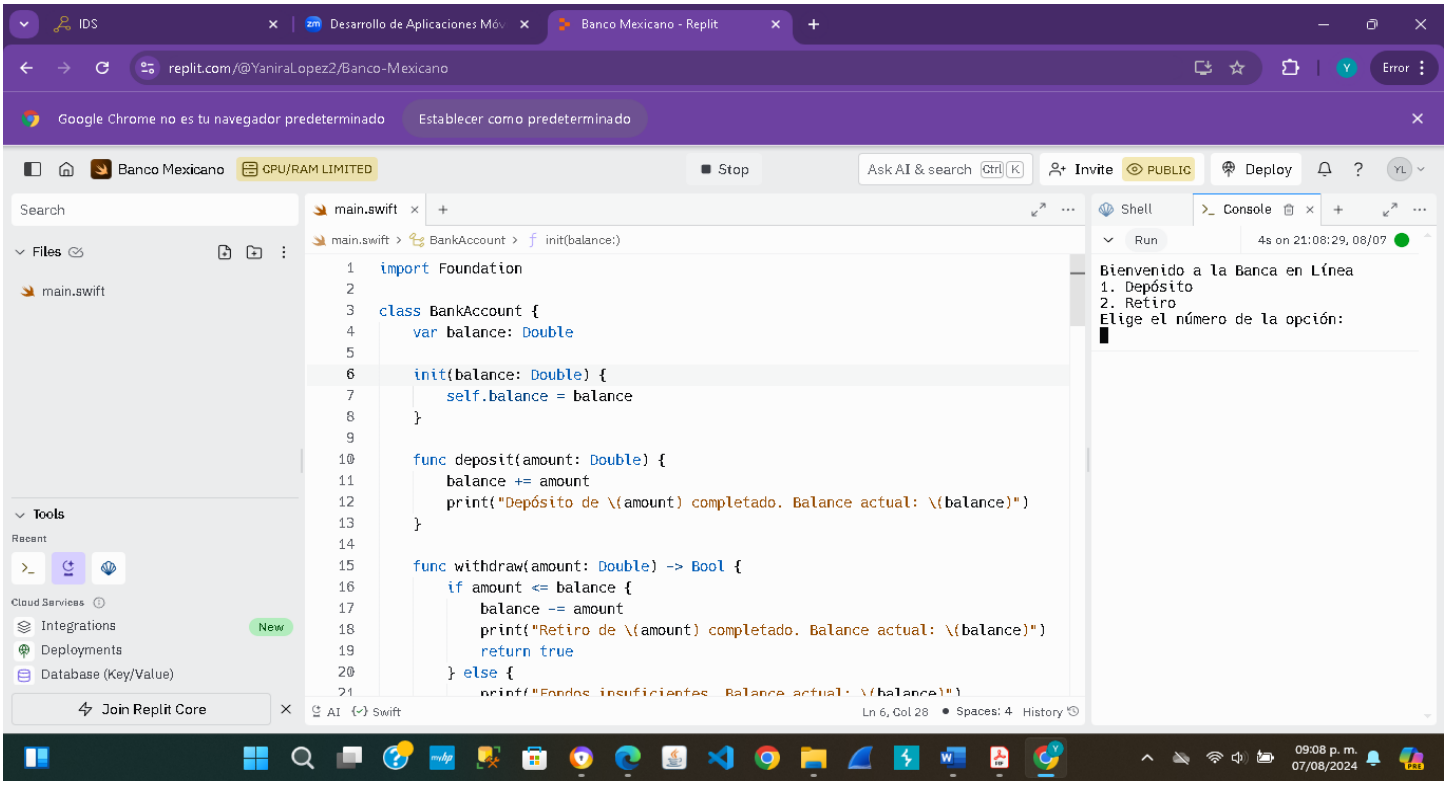
```
let myAccount = BankAccount (balance: 0.0)
let myOnlineBanking = OnlineBanking (account: myAccount)
myOnlineBanking.showMenu()
```

Prueba del programa

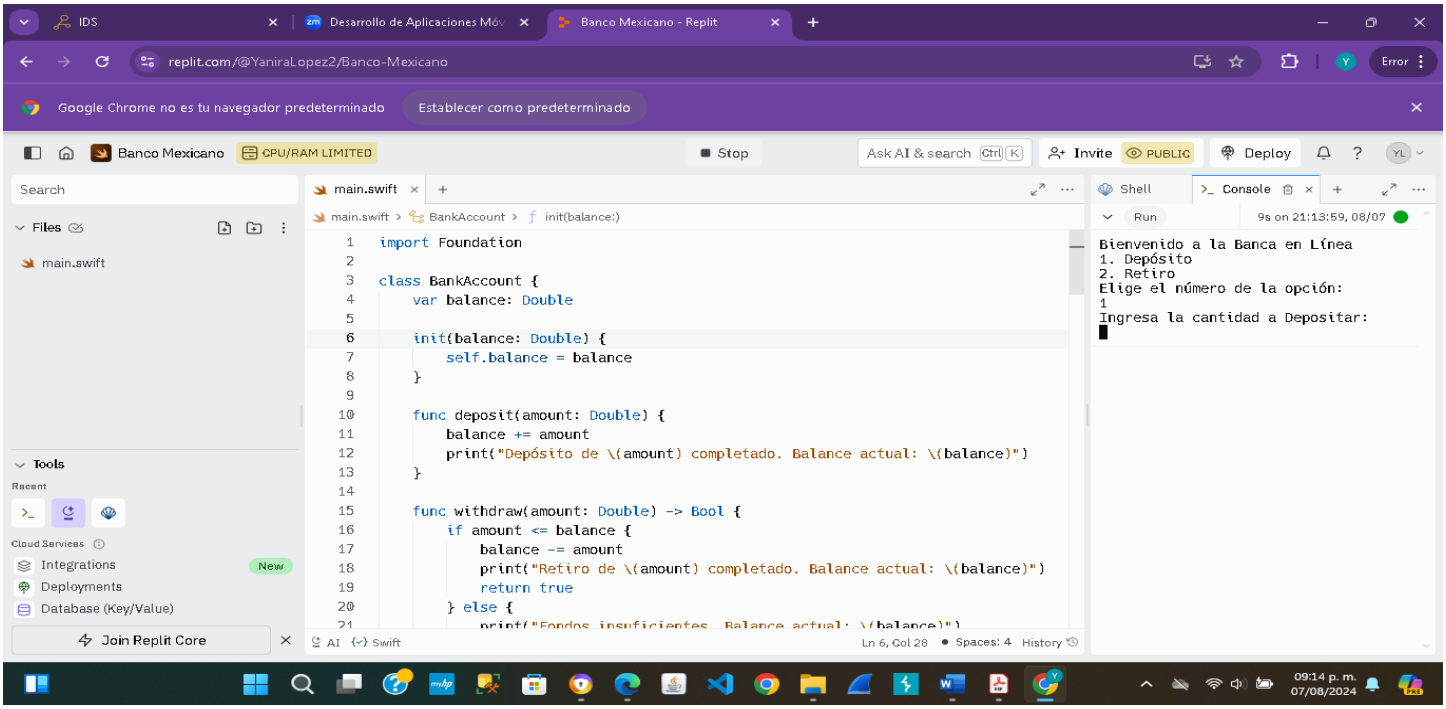
En la siguiente imagen podemos observar el inicio en el compilador online de Replit.



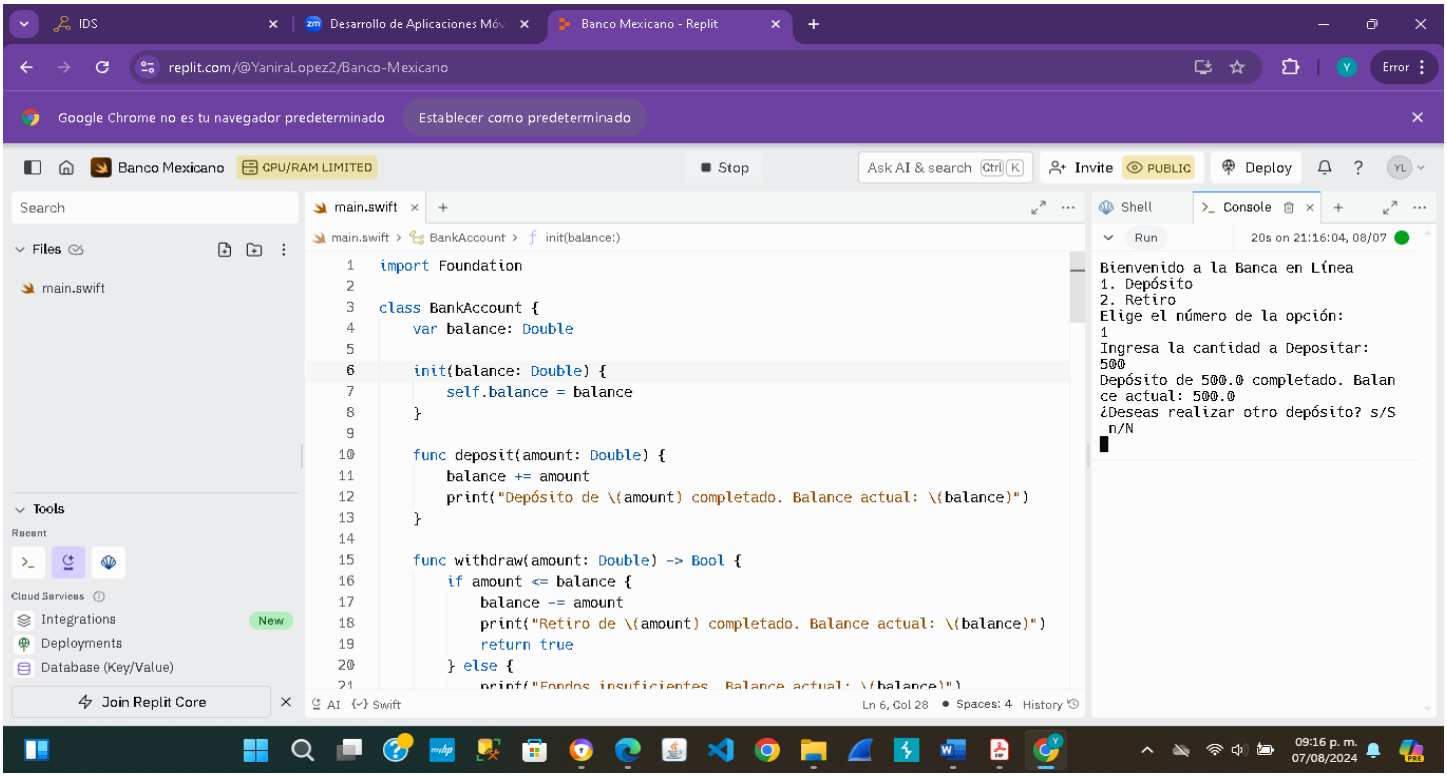
En la siguiente imagen, se puede observar la interfaz del menú del programa de banca en línea, que permite al usuario interactuar con su cuenta bancaria a través de las siguientes opciones:



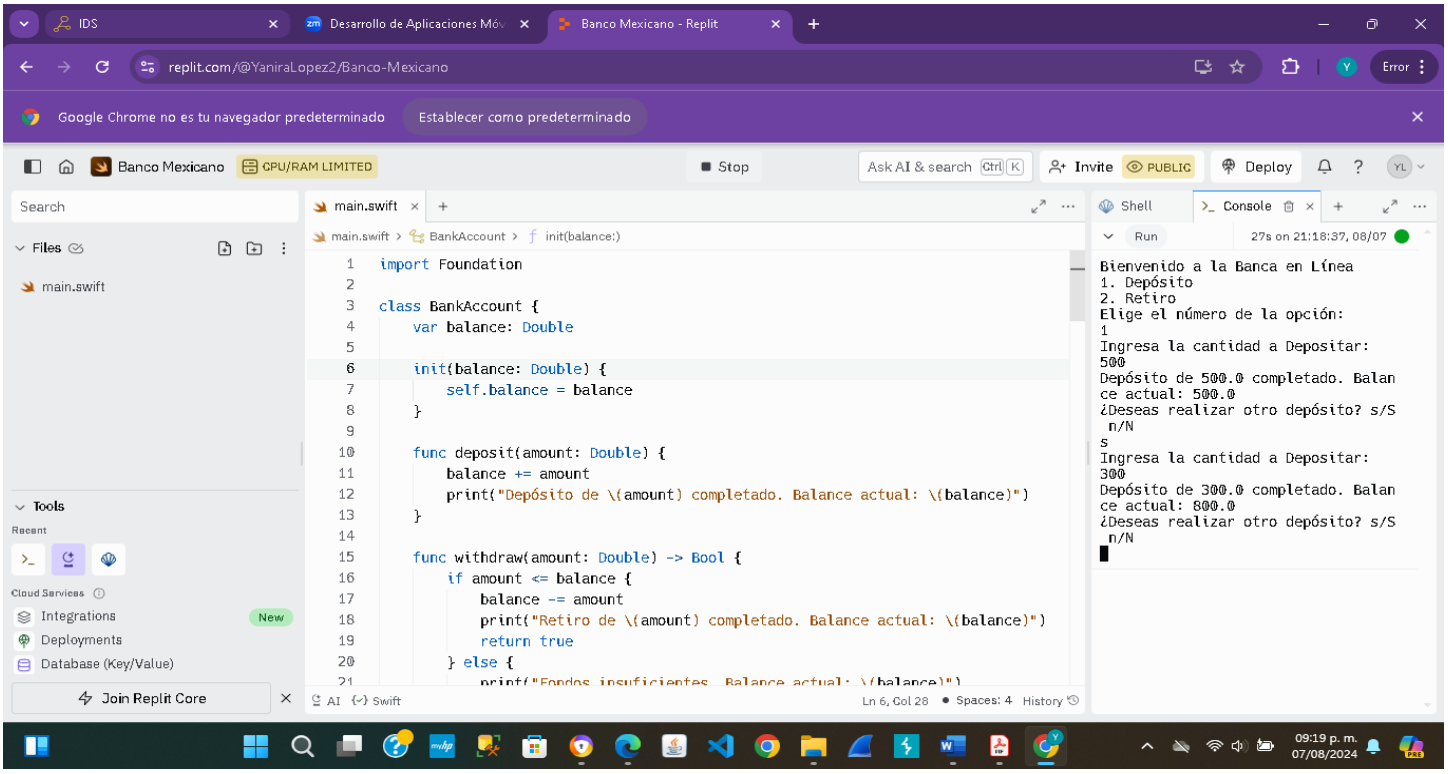
A continuación, se muestra que, al seleccionar la "Opción 1: Depósito", el sistema presenta un mensaje indicando al usuario que ingrese la cantidad de dinero que desea depositar.



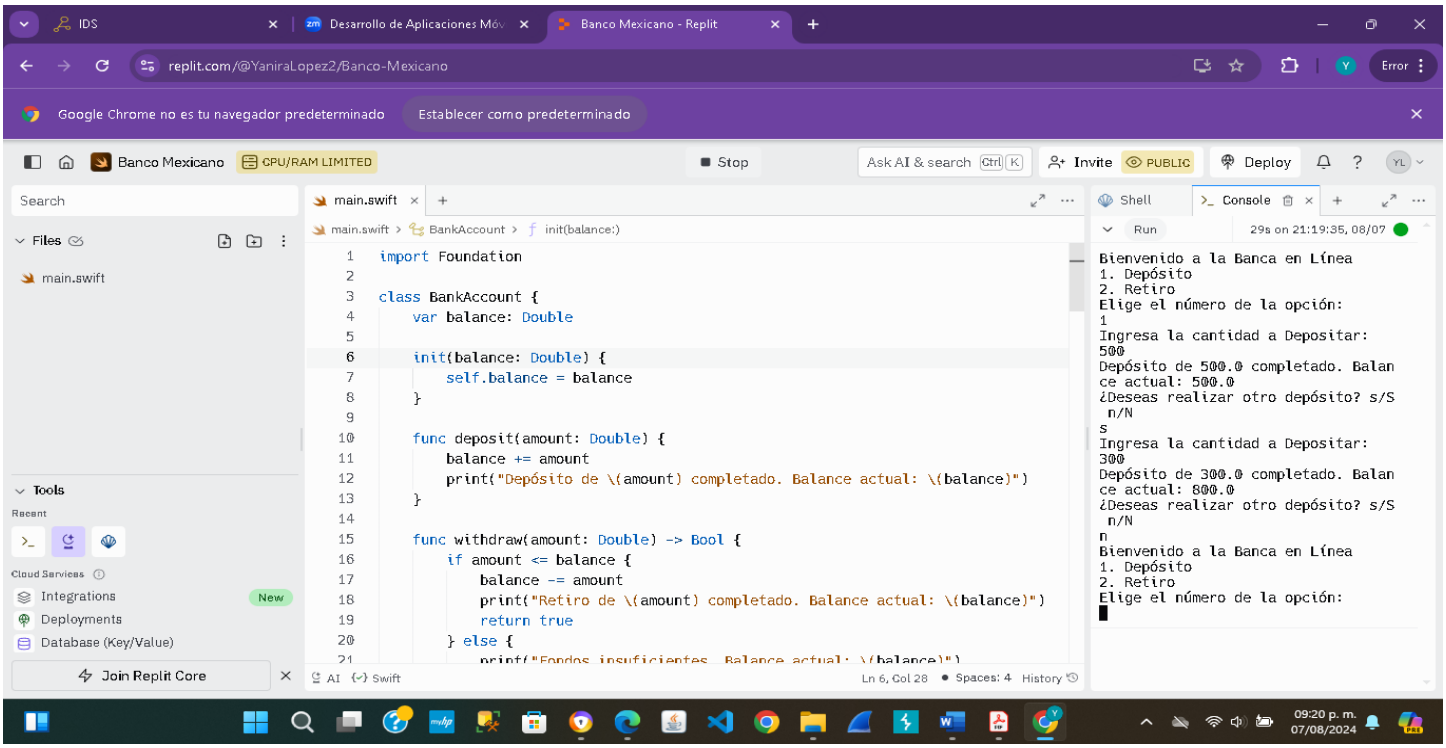
En la siguiente imagen se observa que, después de ingresar la cantidad y presionar Enter, el sistema mostrará un mensaje que pregunta: **"¿Deseas realizar otro depósito? s/S n/N:"**.



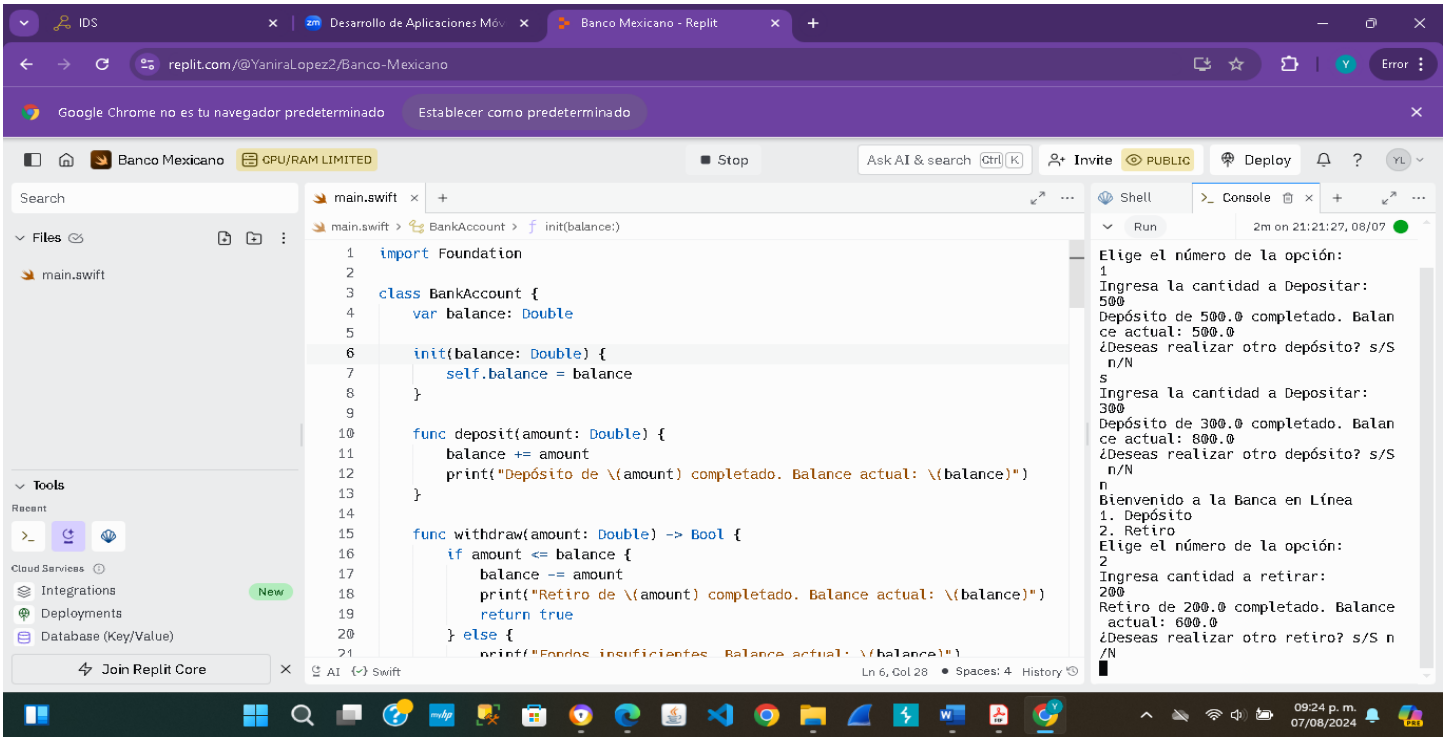
En la siguiente imagen se ilustra que, si se ingresa la letra "S", el sistema volverá a mostrar la pantalla solicitando al usuario que **"Ingrese la cantidad a Depositar:"**.



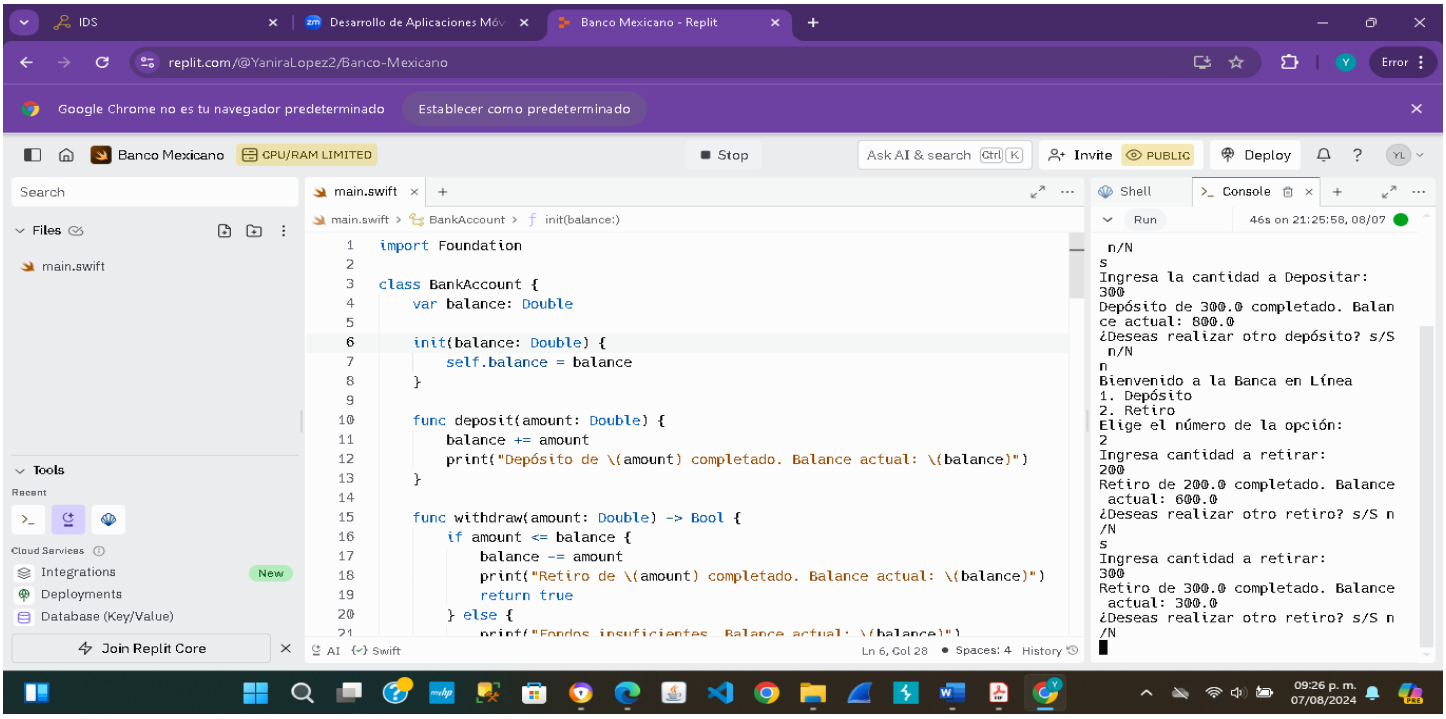
En la siguiente imagen podemos observar que, en caso de teclear la letra “N”, el sistema volverá a mostrar la siguiente pantalla:**"¿Desea continuar con otra operación? s/S n/N:"**



En la siguiente imagen podemos observar que, al momento de ingresar al sistema por segunda vez y ya tener dinero en la cuenta, aparece el siguiente mensaje: **“Ingresa cantidad a retirar:"**



En la siguiente imagen podemos observar que, al ingresar la cantidad a retirar, si se cuenta con la cantidad suficiente, aparecerá la pantalla: **"¿Deseas realizar otro retiro? s/S n/N:"**. En caso de teclear la letra “S”, se mostrará nuevamente la pantalla: **"Ingresa cantidad a Retirar:"**.



Conclusión

La creación de una aplicación de banca en línea utilizando Swift, como la descrita en esta actividad, tiene un impacto significativo tanto en el campo laboral del desarrollo de software como en la vida cotidiana de los usuarios. Para los desarrolladores y profesionales del sector tecnológico, esta actividad representa una oportunidad para aplicar sus conocimientos en un proyecto real que responde a las demandas del mercado actual. Utilizar Swift, un lenguaje moderno y eficiente, no solo garantiza el desarrollo de una aplicación robusta y segura, sino que también permite a los desarrolladores mejorar sus habilidades y mantenerse al día con las tecnologías emergentes.

Para los usuarios del Banco Mexicano, la disponibilidad de una aplicación de banca en línea transforma la manera en que gestionan sus finanzas. La capacidad de realizar depósitos, retiros y consultas de saldo desde cualquier lugar y en cualquier momento mejora significativamente la conveniencia y la accesibilidad. Esto se traduce en una experiencia bancaria más cómoda y eficiente, eliminando la necesidad de visitar sucursales físicas y permitiendo una mejor gestión del tiempo y los recursos personales.

La implementación de esta solución tecnológica no solo fortalece la competitividad y la innovación en el sector bancario, sino que también mejora la calidad de vida de los usuarios al proporcionarles herramientas modernas y accesibles para la gestión de sus finanzas personales. Este tipo de desarrollo tecnológico es esencial para responder a las necesidades cambiantes de los clientes y para fomentar un entorno bancario más dinámico y centrado en el usuario.

https://drive.google.com/file/d/15TBHBnfZIrdCzQspBYmz2WQiyg6MLg3B/view?usp=drive_link

Referencias

Ingeniería en desarrollo de software. Universidad México Internacional. Recuperado el día 25 de junio de 2024 https://umi.edu.mx/coppel/IDS/mod/scorm/player.php

Video conferencing, web conferencing, webinars, screen sharing. (s. f.). Zoom. https://academiaglobal-mx.zoom.us/rec/share/nTWXaqNCZBAFPLiWG6IWqlgPgKFWr3FSO4ylTCDqhIUKNhrw3RIxyIQm9Njig.GIMns9WCVCDtdW1Q

Video conferencing, web conferencing, webinars, screen sharing. (s. f.-b). Zoom. https://academiaglobal-mx.zoom.us/rec/play/pSGwnoUlTgl_mawFlrgrJB_yYAgpygo_3p5O21Rh1eapK2-ipQGT4kgOdCrZnNh3-7uGzPQaNgDwgcDE.9oer0iuZGwvAuWhB?canPlayFromShare=true&from=share_recording_detail&continueMode=true&componentName=rec-play&originRequestUrl=https%3A%2F%2Facademiaglobal-mx.zoom.us%2Frec%2Fshare%2FU1sVMAB-my2VQZPAEqgo98RrNPwHMZR5kTs11J5CQrb_aS-M3AMh4DIdJeu9YCrQ.RsaqulHdKjjy79QD