CashBabaSDK Integration Guide

Version: 1.0.0
Platform: iOS 15.0+
Language: Swift 5.0+

Table of Contents

1. Introduction

CashBabaSDK is a secure iOS SDK that provides PIN management and payment functionality for CashBaba wallet integration. The SDK handles all UI flows internally and returns results through callback closures.

Key Features:

- Set PIN for new users
- Change existing PIN
- Forgot PIN recovery flow
- Secure payment processing

2. Requirements

Requirement	Version
iOS Deployment Target	15.0+
Swift	5.0+
Xcode	14.0+

3. Installation

3.1 CocoaPods

Add the following to your `Podfile`:

```
platform :ios, '15.0'

target 'YourAppName' do
```

```
  use_frameworks!

  pod 'CashBabaSDK', :path => '/path/to/CashBabaSDK'
  # Or from remote repository:
  # pod 'CashBabaSDK', :git => 'https://github.com/your-org/CashBabaSDK
end
```

Then run:

```
pod install
```

Open the `.xcworkspace` file to continue development.

3.2 Swift Package Manager (SPM)

1. In Xcode, go to **File → Add Packages...**
2. Enter the repository URL
3. Select the version and click **Add Package**

Or add to your `Package.swift`:

```
dependencies: [
    .package(url: "https://github.com/your-org/CashBabaSDK.git", from: "
]
```

3.3 Manual XCFramework Integration

1. Download `CashBabaSDK.xcframework` from the release package
2. Drag and drop the `.xcframework` into your Xcode project
3. In your target's **General** tab, ensure the framework is listed under **Frameworks, Libraries, and Embedded Content**
4. Set **Embed** to **Embed & Sign**

Steps in Xcode:

Step	Action
1	Select your project in the Navigator
2	Select your app target
3	Go to **General** tab
4	Scroll to **Frameworks, Libraries, and Embedded Content**
5	Click **+** and add `CashBabaSDK.xcframework`
6	Set **Embed** dropdown to **Embed & Sign**

4. SDK Initialization

Before using any SDK features, you must initialize the SDK with your environment configuration.

Import the SDK

```
import CashBabaSDK
```

Initialize

```
do {
    try CashBaba.shared.initialize(
        environment: .demo,      // or .live for production
        languageCode: "en"       // "en" for English, "bn" for Bengali
    )
} catch {
    print("SDK Initialization failed: \(error.localizedDescription)")
}
```

Initialization Parameters

Parameter	Type	Required	Description
`environment`	`Environment`	Yes	`.demo` for sandbox, `.live` for production
`languageCode`	`String`	Yes	`"en"` for English, `"bn"` for Bengali

5. Available Features

The SDK provides four main features. Each feature is presented as a full-screen modal flow.

5.1 Set PIN

Allows new users to set up their wallet PIN.

```
let args = NavigationArgs(
    type: .SET_PIN,
    wToken: "your-w-token",
    languageCode: "en",
    environment: .demo,
    clientId: "your-client-id",
    clientSecret: "your-client-secret",
    phone: "01XXXXXXXXX"  // User's phone number
)

SDKPresenter.present(
    from: self,
    args: args,
    onSuccess: { result in
        // PIN set successfully
        if let response = result.setPinResponse {
            print("PIN Set - Code: \(response.code ?? 0)")
        }
    },
```

```
    onFailed: { failure in
        print("Error: \(failure.errorMessage)")
    },
    onUserCancel: {
        print("User cancelled the flow")
    }
)
```

5.2 Change PIN

Allows existing users to change their PIN.

```
let args = NavigationArgs(
    type: .CHANGE_PIN,
    wToken: "your-w-token",
    languageCode: "en",
    environment: .demo,
    clientId: "your-client-id",
    clientSecret: "your-client-secret"
)

SDKPresenter.present(
    from: self,
    args: args,
    onSuccess: { result in
        // PIN changed successfully
        if let response = result.changePinResponse {
            print("PIN Changed - Code: \(response.code ?? 0)")
        }
    },
    onFailed: { failure in
        print("Error: \(failure.errorMessage)")
    },
    onUserCancel: {
        print("User cancelled the flow")
    }
)
```

5.3 Forgot PIN

Allows users to recover/reset their PIN using NID and Date of Birth verification.

```
let args = NavigationArgs(
    type: .FORGET_PIN,
    wToken: "your-w-token",
    languageCode: "en",
    environment: .demo,
    clientId: "your-client-id",
    clientSecret: "your-client-secret"
)

SDKPresenter.present(
    from: self,
    args: args,
    onSuccess: { result in
        // PIN reset successfully
        if let response = result.forgetPinResponse {
            print("PIN Reset - Code: \(response.code ?? 0)")
        }
    },
    onFailed: { failure in
        print("Error: \(failure.errorMessage)")
    },
    onUserCancel: {
        print("User cancelled the flow")
    }
)
```

5.4 Payment

Process a payment transaction with PIN verification.

```
let args = NavigationArgs(
    type: .PAYMENT,
    wToken: "your-w-token",
    languageCode: "en",
    environment: .demo,
```

```
    paymentReference: "your-payment-reference-id",
    clientId: "your-client-id",
    clientSecret: "your-client-secret"
)

SDKPresenter.present(
    from: self,
    args: args,
    onSuccess: { result in
        // Payment successful
        if let response = result.paymentResponse {
            print("Transaction ID: \(response.data?.transactionID ?? ""
            print("Amount: \(response.data?.totalAmount ?? 0)")
            print("Status: \(response.data?.transactionStatus ?? "")")
        }
    },
    onFailed: { failure in
        print("Payment failed: \(failure.errorMessage)")
    },
    onUserCancel: {
        print("User cancelled payment")
    }
)
```

6. Handling SDK Responses

The SDK communicates results through three callback closures.

6.1 Success Callback

Called when the operation completes successfully.

```
onSuccess: { (result: CBSuccessModel) in
    // Handle success based on operation type
    if let setPinResponse = result.setPinResponse {
        // Set PIN completed
    }
    if let changePinResponse = result.changePinResponse {
        // Change PIN completed
```

```
    }
    if let forgetPinResponse = result.forgetPinResponse {
        // Forgot PIN completed
    }
    if let paymentResponse = result.paymentResponse {
        // Payment completed
    }
}
```

6.2 Failure Callback

Called when an error occurs during the operation.

```
onFailed: { (failure: CBFailedModel) in
    let errorMessage = failure.errorMessage
    // Display error to user or log it
    showAlert(message: errorMessage)
}
```

Common Error Scenarios:

- Network connectivity issues
- Invalid credentials
- Session expired
- Server errors

6.3 User Cancel Callback

Called when the user manually closes the SDK without completing the operation.

```
onUserCancel: {
    // User closed the SDK
    // You may want to show a message or take appropriate action
}
```

7. Response Models

CBSuccessModel

The unified success response containing operation-specific data.

```
public struct CBSuccessModel: Codable {
    public let setPinResponse: SetPinResponse?
    public let changePinResponse: ChangePinResponse?
    public let forgetPinResponse: ForgetPinResponse?
    public let paymentResponse: PaymentResponse?
}
```

CBFailedModel

```
public struct CBFailedModel: Codable {
    public let errorMessage: String
}
```

SetPinResponse / ChangePinResponse / ForgetPinResponse

```
public struct SetPinResponse: Codable {
    public let code: Int?
    public let messages: [String]?
    public let details: String?
}
```

PaymentResponse

```
public struct PaymentResponse: Codable {
    public let code: Int?
    public let messages: [String]?
    public let data: PaymentData?
    public let details: String?
}
```

```
public struct PaymentData: Codable {
    public let responseCode: Int?
    public let message: String?
    public let transactionID: String?
    public let paymentReference: String?
    public let transactionCurrencyCode: String?
    public let transactionAmount: Double?
    public let feeAmount: Double?
    public let totalAmount: Double?
    public let availableBalance: Double?
    public let transactionStatus: String?
}
```

8. Environment Configuration

The SDK supports two environments:

Environment	Usage	Base URL
`.demo`	Development & Testing	Sandbox server
`.live`	Production	Production server

```
// For development/testing
try CashBaba.shared.initialize(environment: .demo, languageCode: "en")

// For production
try CashBaba.shared.initialize(environment: .live, languageCode: "en")
```

> ⚠️ **Important:** Always use `.demo` environment during development and testing. Switch to `.live` only for production releases.

9. Language Support

The SDK supports two languages:

Code	Language
"en"	English
"bn"	Bengali (বাংলা)

```
// English
let args = NavigationArgs(
    type: .SET_PIN,
    languageCode: "en",
    // ... other parameters
)

// Bengali
let args = NavigationArgs(
    type: .SET_PIN,
    languageCode: "bn",
    // ... other parameters
)
```

10. Complete Integration Example

Here's a complete example of integrating the Set PIN feature:

```
import UIKit
import CashBabaSDK

class PaymentViewController: UIViewController {

    // Your credentials (store securely, not hardcoded)
    private let clientId = "your-client-id"
    private let clientSecret = "your-client-secret"
    private let wToken = "user-w-token"

    override func viewDidLoad() {
        super.viewDidLoad()
```

```
        initializeSDK()
    }

    private func initializeSDK() {
        do {
            try CashBaba.shared.initialize(
                environment: .demo,
                languageCode: "en"
            )
        } catch {
            showAlert(title: "Error", message: "SDK initialization fail
        }
    }

    @IBAction func setUpPINTapped(_ sender: UIButton) {
        let args = NavigationArgs(
            type: .SET_PIN,
            wToken: wToken,
            languageCode: "en",
            environment: .demo,
            clientId: clientId,
            clientSecret: clientSecret,
            phone: "01XXXXXXXX"
        )

        SDKPresenter.present(
            from: self,
            args: args,
            onSuccess: { [weak self] result in
                self?.handleSuccess(result)
            },
            onFailed: { [weak self] failure in
                self?.showAlert(title: "Error", message: failure.errorM
            },
            onUserCancel: { [weak self] in
                self?.showToast("Operation cancelled")
            }
        )
    }

    @IBAction func makePaymentTapped(_ sender: UIButton) {
        let args = NavigationArgs(
            type: .PAYMENT,
```

```
            wToken: wToken,
            languageCode: "en",
            environment: .demo,
            paymentReference: "payment-reference-id",
            clientId: clientId,
            clientSecret: clientSecret
        )

        SDKPresenter.present(
            from: self,
            args: args,
            onSuccess: { [weak self] result in
                if let payment = result.paymentResponse?.data {
                    self?.showAlert(
                        title: "Payment Successful",
                        message: "Transaction ID: \(payment.transaction
                    )
                }
            },
            onFailed: { [weak self] failure in
                self?.showAlert(title: "Payment Failed", message: failu
            },
            onUserCancel: { [weak self] in
                self?.showToast("Payment cancelled")
            }
        )
    }

    private func handleSuccess(_ result: CBSuccessModel) {
        if result.setPinResponse != nil {
            showAlert(title: "Success", message: "PIN set successfully!
        } else if result.changePinResponse != nil {
            showAlert(title: "Success", message: "PIN changed successfu
        } else if result.forgetPinResponse != nil {
            showAlert(title: "Success", message: "PIN reset successfull
        } else if result.paymentResponse != nil {
            showAlert(title: "Success", message: "Payment completed!")
        }
    }

    private func showAlert(title: String, message: String) {
        let alert = UIAlertController(title: title, message: message, p
        alert.addAction(UIAlertAction(title: "OK", style: .default))
```

```
            present(alert, animated: true)
        }

        private func showToast(_ message: String) {
            // Implement your toast/snackbar
        }
    }
```

11. Troubleshooting

Common Issues

Issue	Solution
SDK initialization fails	Ensure you're calling `initialize()` before any SDK operations
Assets not loading	Verify the XCFramework is properly embedded with "Embed & Sign"
Session expired error	The access token has expired; re-authenticate the user
Network errors	Check internet connectivity and firewall settings

Xcode Build Error: Sandbox rsync.samba deny

If you encounter the following error when building your project:

```
Sandbox: rsync.samba(13105) deny(1) file-write-create /Users/.../Librar
```

This is a known Xcode issue. Follow these steps to resolve it:

Solution:

1. Open your project in Xcode
2. Select your **project** in the Navigator (not the target)
3. Go to **Build Settings** tab
4. Search for `ENABLE_USER_SCRIPT_SANDBOXING`
5. Set **User Script Sandboxing** to **No**

Alternatively, you can add this to your Podfile's `post_install` hook:

```
post_install do |installer|
  installer.pods_project.targets.each do |target|
    target.build_configurations.each do |config|
      config.build_settings['ENABLE_USER_SCRIPT_SANDBOXING'] = 'NO'
    end
  end
end
```

Then run `pod install` again.

> **Note:** This issue occurs in Xcode 15+ due to stricter sandboxing rules for build scripts.

Debug Tips

1. **Check Environment:** Ensure you're using `.demo` for testing
2. **Verify Credentials:** Confirm `clientId`, `clientSecret`, and `wToken` are correct
3. **Check iOS Version:** SDK requires iOS 15.0+
4. **Clean Build:** Try **Product → Clean Build Folder** (Cmd+Shift+K) if you encounter unexpected issues

Support

For technical support, contact:

Channel	Contact
Email	support@cashbaba.com
Website	https://www.cashbaba.com