IOS Guide

[iOS] Intro

Cardpay Unlimint mobile SDK for iOS (UnlimintSdk) helps you to:

Embed card data forms in the merchant's mobile app and securely collect and transmit the user's card data for:

- Card tokenization (without a payment) on the Unlimint side
- Making a mobile payment
- Making a payment with card token

Installation

UnlimintSDK is available through CocoaPods

To install UnlimintSDK-UI with CocoaPods, add the following lines to your `Podfile`.

```
source 'https://github.com/cardpay/ios-sdk-podspec.git'

platform :ios, '11.0'
use_frameworks!

pod 'UnlimintSDK-UI'

unlimintXCFramework = ['Alamofire', 'Moya', 'Swinject', 'UnlimintSDK-Core']

post_install do |installer|
    installer.pods_project.targets.each do |target|
        if unlimintXCFramework.include? "#{target}"
            target.build_configurations.each do |config|
                config.build_settings['BUILD_LIBRARY_FOR_DISTRIBUTION'] = 'YES'
            end
        end
    end
end
```

To install UnlimintSDK-Core with CocoaPods, add the following lines to your `Podfile`.

```
source 'https://github.com/cardpay/ios-sdk-podspec.git'

platform :ios, '11.0'
use_frameworks!

pod 'UnlimintSDK-Core'

unlimintXCFramework = ['Alamofire', 'Moya']

post_install do |installer|
    installer.pods_project.targets.each do |target|
        if unlimintXCFramework.include? "#{target}"
            target.build_configurations.each do |config|
                config.build_settings['BUILD_LIBRARY_FOR_DISTRIBUTION'] = 'YES'
            end
        end
    end
end
```

`Then run pod install command. For details of the installation and usage of CocoaPods, visit its official website.`

Basic Usage

Environment

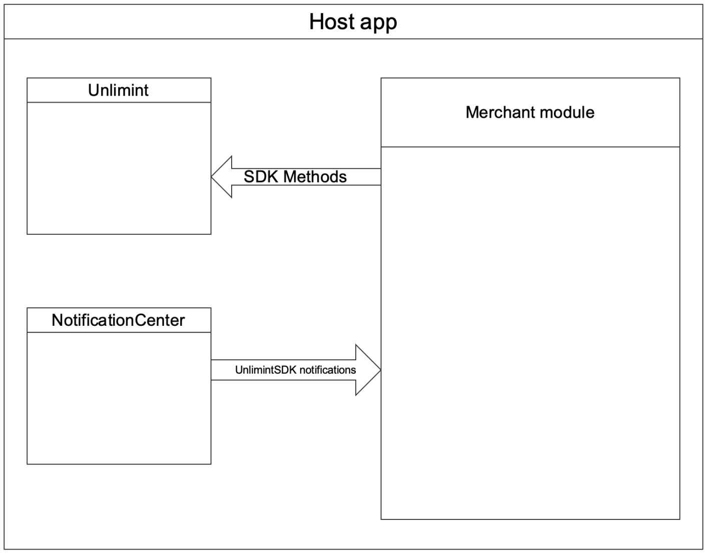
```
# For UnlimintSDK-UI
Unlimint.shared.environment = .sandbox

# For UnlimintSDK-Core
Environments.current = .sandbox
```

UI customization

Full info here. [iOS] Customization

```
'Unlimint.shared.theme'

public struct Theme {

    public var navigationStyle: NavigationBarStyle

    public var mainButtonStyle: MainButtonStyle

    public var viewControllerStyle: ViewControllerStyle

    public init(navigationStyle: NavigationBarStyle = .init(bar: .largeNavBar,
                                                            statusBarStyle: .default,
                                                            navigationBarColor: .transparentDark,
                                                            tintColor: .clear),
                                mainButtonStyle: MainButtonStyle = .init(cornerRadius: 2,
                                                         titleColor: (UIConstants.Colors.primaryBlack,
                                                                      UIConstants.Colors.primaryGray),
                                                         backgroundColor: (UIConstants.Colors.primaryGreen,
                                                                           UIConstants.Colors.primaryWhite)),
                                viewControllerStyle: ViewControllerStyle = .init(backgroundColor: .white))
}
```

[iOS] API description



Methods

Binding

```
     /**
     Use this method for binding card.
     */
    public func bindNewCardFor(
                for mobileToken: String,
                 with data: UnlimintSDK.BindingMethodData,
                presentationStyle style: PresentationStyle)
```

Payment

```
    /**
     Use this method for payment.
     */
    func payment(
                for token: String,
        with data: PaymentMethodData,
        presentationStyle style: PresentationStyle)
```

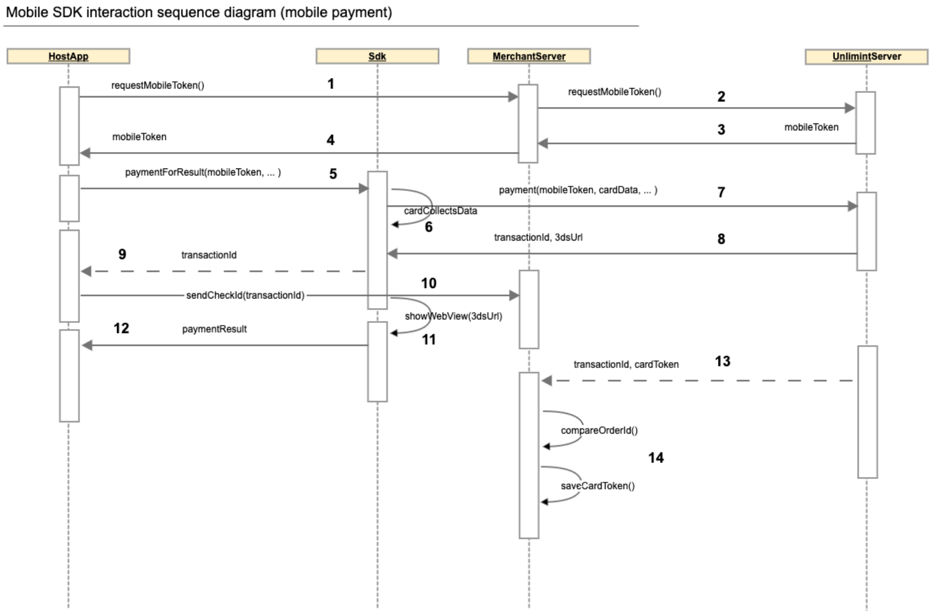
Payment with token

```
           /**
     Use this method for payment.
     */
    public func paymentWithToken(
                for mobileToken: String,
                with data: UnlimintSDK.PaymentTokenMethodData,
                presentationStyle style: PresentationStyle)
```

Payment with paypal

```
           /**
     Use this method for paypal payment.
    */
    func paypalPayment(
        for mobileToken: String,
        with data: PaypalPaymentMethodData,
        presentationStyle style: PresentationStyle)
```

[iOS] Interaction

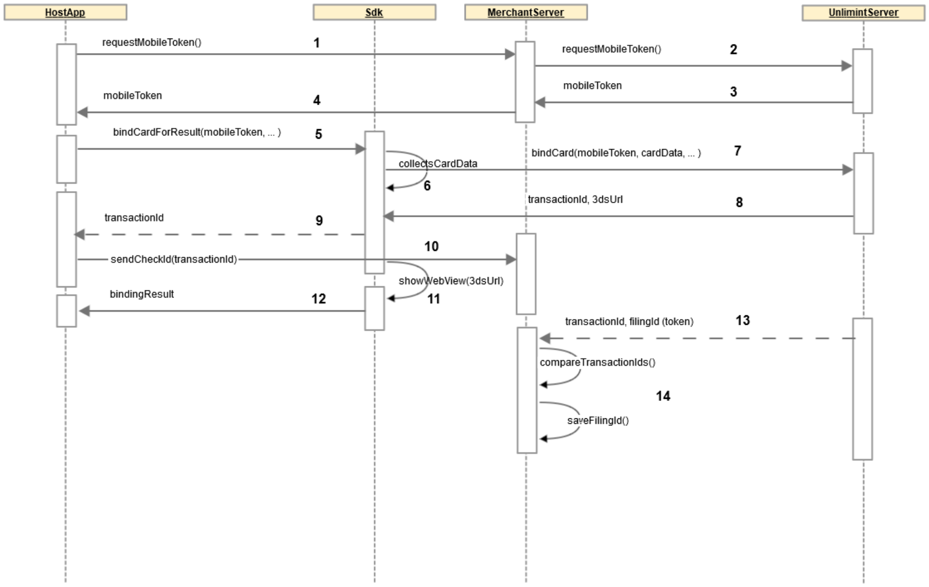


User scenario for mobile payment process:

ID	Requirement text
1	Host (merchant) mobile application sends request for getting mobile token to merchant backend (requestMobileToken)
2	Merchant backend sends POST request (JSON) with valid access token for getting mobile token to API v3 endpoint https://cardpay.com/api/mobile/token (Unlimint server): Header of request: valid access_token of merchant. Parameters of request: request.id - request ID, should be unique for time period of 30 minutes request.time - request attempt date and time up to milliseconds in ISO 8601 format (milliseconds is optional part) (example of format - yyyy-MM-dd'T'HH:mm:ss.SSS'Z'
3	API v3 returns response to merchant backend with requested mobile token: **-** mobile_token (string, unique identifier, 128 symbols) **-** expires (date and time of mobile token expiration in ISO 8601 format, example of format - yyyy-MM-dd'T'HH:mm:ss.SSS'Z') Lifetime of mobile token is (see below): 'lifetime of mobile token' <= 5 min and < 'Access_token' ('Bearer' token) life time. Granted LT of access token (for mobile token creation) should be less or equal 5 min but more or equeal than 4 min 4 min <= access token<= 5 min
4	Merchant backend sends mobile token to the host application
5	Host application calls paymentForResult(mobileToken) method of the mobile SDK
6	Customer fills in card data in card data form (in SDK)
7	Mobile SDK sends request with customer, card data, received mobile token (JSON) for making a payment to API v3 endpoint https://cardpay.com/api/mobile/* (here is presented masked endpoint)
8	API v3 sends a payment response with redirect URL and transaction id (for 3DS verification) to the mobile SDK
9	Mobile SDK returns transaction id to the host application, host application resend it to the merchant backend

10	Host application sends transaction id to the merchant server
11	Mobile SDK presents webview with 3dsUrl to the customer for 3-D Secure verification
12	3-D Secure verification procedure passes and customer redirects to success or decline url (paymentResult)
13	Unlimint API v3 sends callback to the merchant backend (with transaction id and card token (if it was requested by customer)
14	Merchant backend compares received transaction id's from the host application and callback and saves a received card token for a future use (recommendations to do)

Mobile SDK interaction sequence diagram (binding card)



User scenario for card binding process:

ID	Requirement text
1	Host (merchant) mobile application sends request for getting mobile token to merchant backend (requestMobileToken)
2	Merchant backend sends POST request (JSON) with valid access token for getting mobile token to API v3 endpoint https://cardpay.com/api/mobile/token: Header of request: valid access_token of merchant Parameters of request: request.id - request ID, should be unique for time period of 30 minutes request.time - request attempt date and time up to milliseconds in ISO 8601 format (milliseconds is optional part) (example of format - yyyy-MM-dd'T'HH:mm:ss.SSS'Z'
3	API v3 returns response to merchant backend with requested mobile token: **-** mobile_token (string, unique identifier, 128 symbols) **-** expires (date and time of mobile token expiration in ISO 8601 format, example of format - yyyy-MM-dd'T'HH:mm:ss.SSS'Z') Lifetime of mobile token is (see below): 'lifetime of mobile token' <= 5 min and < 'Access_token' ('Bearer' token) life time. Granted LT of access token (for mobile token creation) should be less or equal 5 min but more or equeal than 4 min 4 min <= access token<= 5 min
4	Merchant backend sends mobile token to the host application
5	Host application calls bindCardForResult(mobileToken) method of the mobile SDK

6	Customer fills in card data in card data form (in SDK)
7	Mobile SDK sends request with customer, card data, received mobile token (JSON) for card binding to API v3 endpoint https://cardpay.com/api/mobile/* (here is presented masked endpoint)
8	API v3 sends a card binding response with redirect URL and transaction id (for 3DS verification) to the mobile SDK
9	Mobile SDK returns transaction id to the host application, host application resend it to the merchant backend
10	Host application sends transaction id to the merchant server
11	Mobile SDK presents webview with 3dsUrl to the customer for 3-D Secure verification
12	3-D Secure verification procedure passes and customer redirects to success or decline url (bindingResult)
13	Unlimint API v3 sends callback to the merchant backend (with transaction id and filing id)
14	Merchant backend compares received transaction id's from the host application and callback and saves a received filing id for a future use (recommendations to do)

[iOS] Customization

General

At the moment, the design system is implemented as a theme. You can use a standard theme or configure his own. Below is a table with the existing fields.

Theme:

Field	Component
navigationStyle	NavigationBarStyle
mainButtonStyle	MainButtonStyle
viewControllerStyle	ViewControllerStyle

Components

NavigationBarStyle

<table>
<tr><th>Field</th><th>Data Type</th><th></th></tr>
<tr><td>bar</td><td><table><tr><th>Bar</th></tr><tr><td>largeNavBar</td></tr><tr><td>small</td></tr><tr><td>smallTranslucent</td></tr></table></td><td>Style of UINavigationBar</td></tr>
<tr><td>statusBarStyle</td><td><table><tr><th colspan="2">UIStatusBarStyle</th></tr><tr><td>default</td><td>A dark status bar, intended for use on light backgrounds.</td></tr><tr><td>lightContent</td><td>A light status bar, intended for use on dark backgrounds.</td></tr><tr><td>darkContent</td><td>A dark status bar, intended for use on light backgrounds.</td></tr></table></td><td>The style of the device's status bar.</td></tr>
<tr><td>navigationBar</td><td><table><tr><th>Color</th></tr><tr><td>light</td></tr><tr><td>dark</td></tr><tr><td>transparentDark</td></tr><tr><td>transparentLight</td></tr><tr><td>custome (title, barTint, backgroundImage, background)</td></tr></table></td><td>Theme of NavigationBar</td></tr>
<tr><td>tintColor</td><td>RGB color</td><td>Tint color of UINavigationBar</td></tr>
</table>

MainButtonStyle

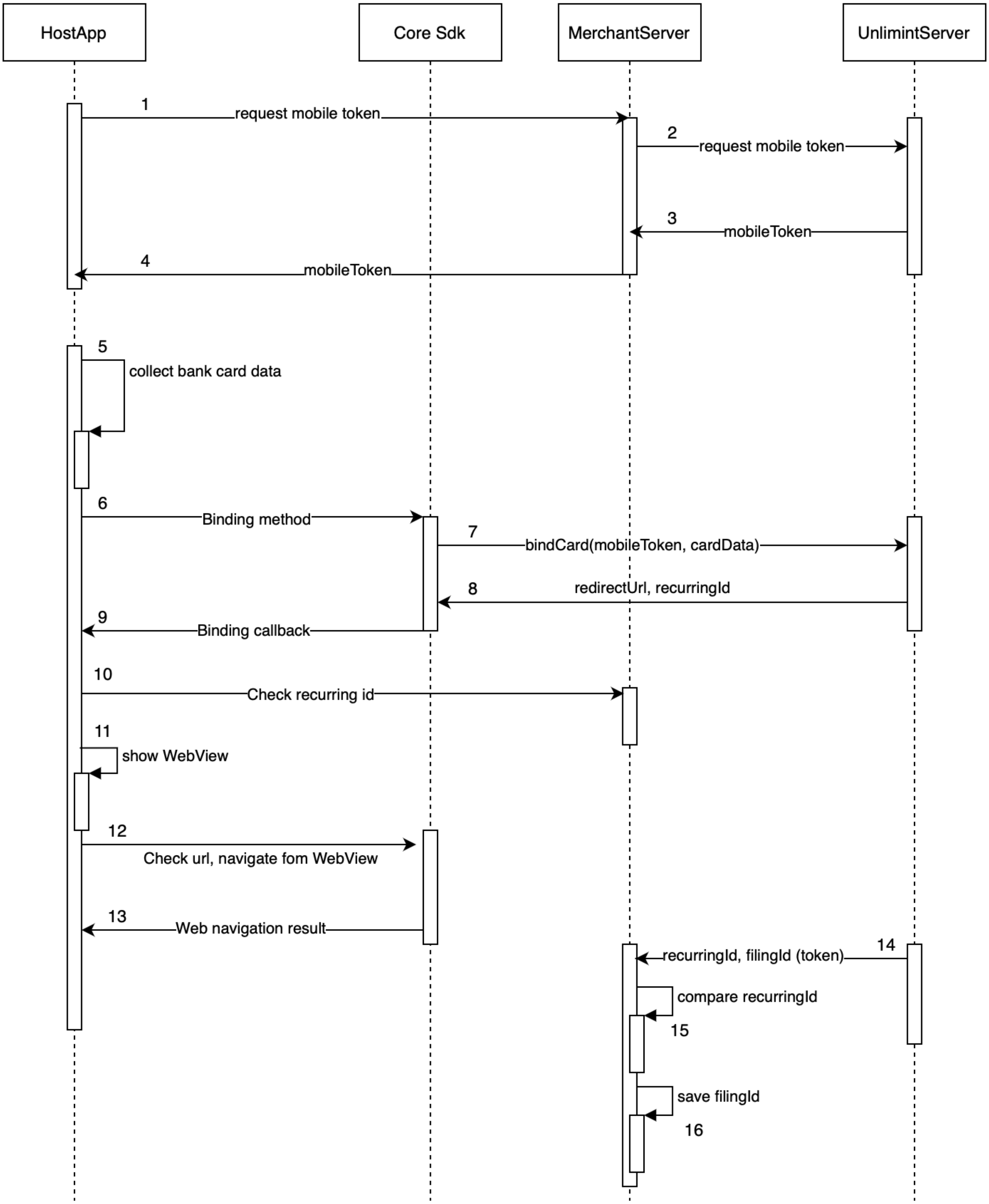
<table>
<tr><th></th><th></th><th></th></tr>
<tr><td>cornerRadius</td><td>Float</td><td>Rounding radius of view.</td></tr>
<tr><td>titleColor</td><td>RGB color</td><td>Color of button text.</td></tr>
<tr><td>backgroundColor</td><td><table><tr><th colspan="2">Color</th></tr><tr><td>normal</td><td>RGB color</td></tr><tr><td>disabled</td><td>RGB color</td></tr></table></td><td>Button background color.</td></tr>
</table>

ViewControllerStyle

backgroundColor	RGB color	The view’s background color.

[IOS] SDK-Core interaction

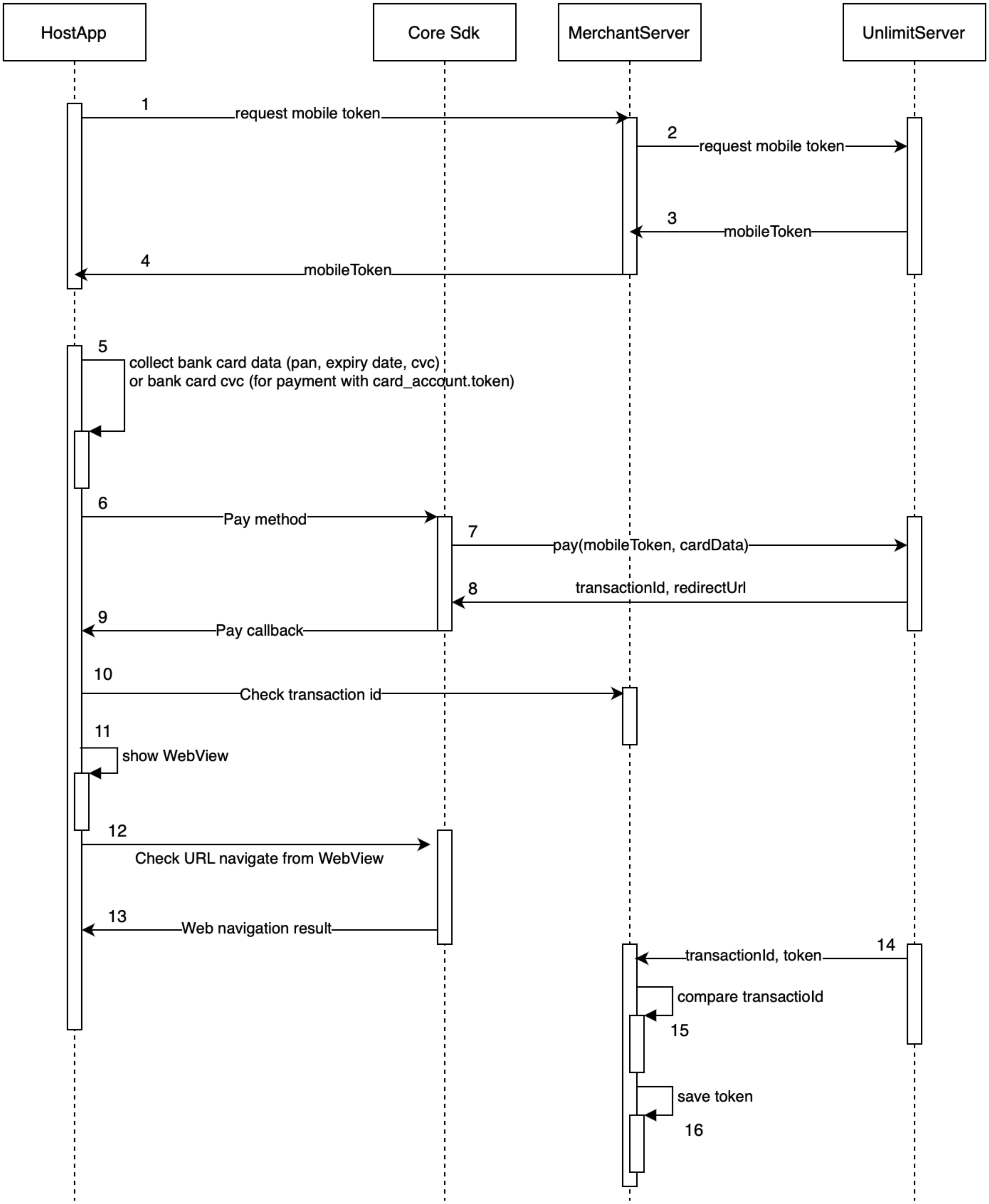
Bank card binding (sequence diagram)



User scenario for the mobile binding process:

Step	Description
1	Host (merchant) mobile application sends a request for getting a mobile token to merchant backend (request mobile token)
2	Merchant backend sends POST request (JSON) with a valid access token for getting a mobile token to API v3 endpoint https://cardpay.com/api/mobile/token Header of request: valid access_token of merchant Parameters of request: request.id - request ID, should be unique for time period of 30 minutes request.time - request attempt date and time up to milliseconds in ISO 8601 format (milliseconds is optional part)(example of format - yyyy-MM-dd'T'HH:mm:ss.SSS'Z'
3	API v3 returns a response to merchant backend with requested mobile token: **-** mobile_token (string, unique identifier, 128 symbols) **-** expires (date and time of mobile token expiration in ISO 8601 format, an example of format - yyyy-MM-dd'T'HH:mm:ss.SSS'Z') A lifetime of the mobile token is (see below): 'lifetime of mobile token' <= 5 min and < 'Access_token' ('Bearer' token) lifetime. Granted LT of the access token (for mobile token creation) should be less or equal 5 min but more or equal than 4 min 4 min <= access token<= 5 min
4	Merchant backend sends mobile token to the host application
5	A customer fills in card data in card data form (in Host Application)
6	Host application calls bindCard(mobileToken, cardData) method of the mobile Core Sdk
7	Core Sdk sends a request with customer, card data, received mobile token (JSON) for card binding to API v3 endpoint https://cardpay.com/api/mobile/* (here is presented masked endpoint)
8	API v3 sends a card binding response with redirect URL and recurringId to the Core Sdk
9	Core Sdk returns recurringId, redirect URL (for WebView), and last 4 digits of card pan to the host application
10	Host application sends recurringId to the merchant server
11	Host application shows WebView (with redirect URL) to the customer for passing binding process
12	On every user redirection (for example, when the user navigates in pages) host application sends this url to Core Sdk. Sdk checks these URLs if the user has passed binding process
13	Binding procedure passes and customer redirects to success or decline url. Core sdk return result of binding procedure
14	Unlimint API v3 sends a callback to the merchant backend (with recurringId and filingId)
15-16	Merchant backend compares received recurring id's from the host application and callback and saves a received filing id for future use (recommendations to do)

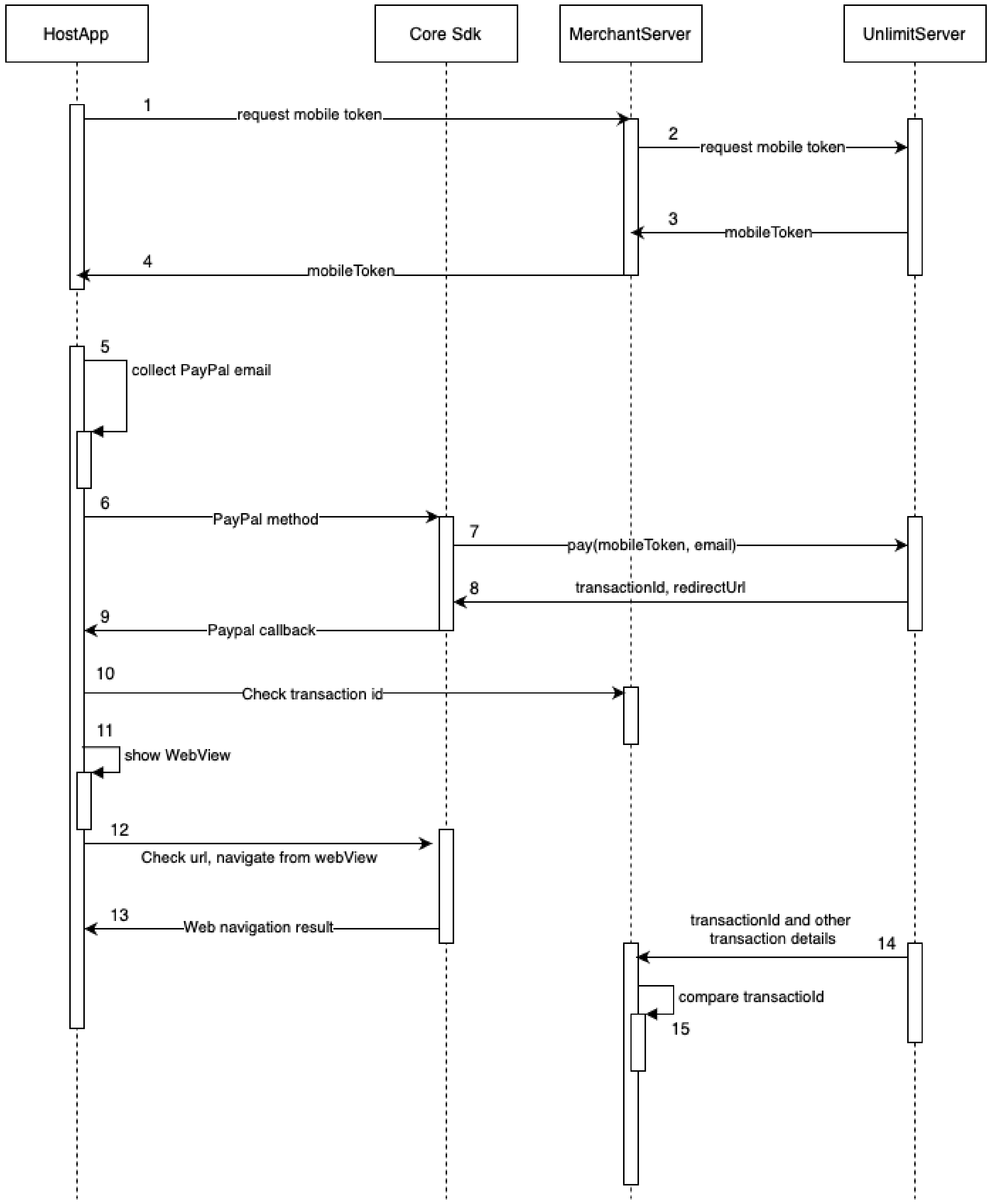
Bank card payment (sequence diagram)



User scenario for mobile payment process:

Step	Description
1 - 4	Similar to the binding scenario
5	Host Application collects bank card data (pan, expire date, cvc) for payment with a bank card or collects only bank card cvc for pay with bank card token
6	Host application calls pay(mobileToken, cardData) method of the mobile Core Sdk
7	Core Sdk sends a request with customer, card data, received mobile token (JSON) for card payment to API v3 endpoint https://cardpay.com/api/mobile/* (here is presented masked endpoint)
8	API v3 sends a card payment response with redirect URL and transactionId to the Core Sdk
9	Core Sdk returns transactionId, redirect URL (for WebView), and last 4 digits of card pan to the host application
10	Host application sends transactionId to the merchant server
11	Host application shows WebView (with redirect URL) to the customer for passing payment process
12	On every user redirection (for example, when the user navigates in pages) host application sends this url to Core Sdk. Sdk checks these URLs if the user has passed payment process
13	Payment procedure passes and customer redirects to success or decline url. Core sdk returns result of payment procedure
14	Unlimint API v3 sends a callback to the merchant backend (with transaction id and card token). Unlimint API v3 send card token only if generateToken=true will be send on 6 step
15-16	Merchant backend compares received transaction id's from the host application and callback and saves a received token for future use.

PayPal payment (sequence diagram)



User scenario for mobile PayPal payment process:

Step	Description
1 - 4	Similar to the payment scenario with bank card
5	Host Application collects PayPal email
6	Host application calls pay(mobileToken, email) method of the mobile Core Sdk
7	Core Sdk sends a request with customer, email, payment data, received mobile token (JSON) for Paypal payment to API v3 endpoint https://cardpay.com/api/mobile/* (here is presented masked endpoint)
8	API v3 sends a payment response with redirectUrl and transaction id to the Core Sdk
9	Core Sdk returns transaction id, redirectUrl (for WebView) to the host application
10	Host application sends transaction id to the merchant server
11	Host application shows WebView (with redirectUrl) to the customer for payment with PayPal
12	On every user redirection (for example, when the user navigates in pages) host application sends this url to Core Sdk. Sdk checks these URLs if the user has passed payment procedure
13	PayPal payment procedure passes and customer redirects to success or decline url. Core sdk returns result payment process
14	Unlimint API v3 sends a callback to the merchant backend (with transaction id and other transaction details)
15	Merchant backend compares received transaction id's from the host application and callback