

# Tokenization

## **Q: What is the difference between Pre-Digitization and Push-Provisioning?**

A: Pre-digitization and push provisioning are two different processes in the tokenization workflow.

Pre-digitization refers to a set of processes that enable the generation of digital payment tokens. It involves turning a payment card into a digital token to facilitate secure and simplified digital payment experiences. During pre-digitization, the Verestro Token Management Platform (TMP) takes care of all the requirements from Token Requestors.

The pre-digitization process involves the following steps:

1. User enters the card details into a Token Requestor wallet (e.g., Apple Pay, Google Pay).
2. The TMP receives an Authorize Service request from the Token Service Provider (TSP) with the card details and other tokenization data provided by the Token Requestor.
3. The TMP checks device score, existing active tokens, and velocity controls.
4. The TMP sends a request to the Issuer's Card Verification API to validate the card details and receives the Card Status, Card ID, User Phone Number, CVC validation Result, and Product Category.
5. Based on the verification result, the TMP returns a decision to the TSP (APPROVED/REQUIRE\_ADDITIONAL\_AUTHENTICATION/DECLINED).

Push provisioning refers to the process of delivering the digitized card profile to the user's device after a successful pre-digitization. It is a crucial step in tokenization where the payment token is provisioned on the user's device, enabling secure transactions.

During push provisioning, the following steps occur:

1. After successful pre-digitization, the digitized card profile is delivered to the user's device.
2. If the pre-digitization decision is APPROVED, the token is activated instantly, and the Verestro TMP may notify the issuer if required.
3. If the pre-digitization decision is REQUIRE\_ADDITIONAL\_AUTHENTICATION, the user is prompted with activation options (e.g., SMS OTP), and the token is activated after successful user authentication.
4. If the pre-digitization decision is DECLINED, the token remains inactive and cannot be activated.

In summary, pre-digitization focuses on the initial token generation process, while push provisioning involves delivering and activating the digitized card profile on the user's device. Both processes are essential for enabling secure and convenient digital payments.

**Q: What are the steps for having Push Provisioning in application? I already have manual provisioning.**

A: Android - Push Provisioning to Google Pay

1. In order to start the Push Provisioning requesting process company responsible for deployment of the application (Verestro or customer) should visit following website: <https://developers.google.com/pay/issuers/apis/push-provisioning/android/launch-process> and select "Request access". Company should enter all necessary details in the form that should pop-up.
2. Once the access from step 1 will be granted company has to fill in next form and pass to Google:
  - screenshots from application - especially those that shows how "Google Pay" button was placed in the application.
  - screen recording showing user flow for adding card to Google Pay
3. Passing additional, paid certification is not needed in case of Google Pay.

Verestro activities:

- in case when Verestro is owner of the application: configure SDK responsible for Push Provisioning internally
- in case customer is owner of the application: provide customer with SDK and ask them to configure it

iOS - Push Provisioning to Apple Pay

In order to start the Push Provisioning requesting process company responsible for deployment of the application (Verestro or customer) should write directly to Apple using Apple Pay

[provisioning@apple.com](mailto:provisioning@apple.com) and [applepayentitlements@apple.com](mailto:applepayentitlements@apple.com).

Email example:

"Hi,

I would like to start the in-app push provisioning process with FIME certification. Can you enable the in-app push provisioning on appstoreconnect for testing purposes for XYZ application  
<https://apps.apple.com/pl/app/XYZ>

Regards,

XYZ"

We provide the data of the application that is on the App Store, and we perform tests on TestFlight using the same application (the same ADAM ID)

Then company should contact with FIME to pass required certification and follow their instructions. Last time we checked with Fime cost was 3125 euro.

Verestro should:

- in case when Verestro is owner of the application: configure SDK responsible for Push Provisioning internally
- in case customer is owner of the application: provide customer with SDK and ask them to configure it

**Q: Can you provide exact sequence of steps for whole tokenisation process?**

A: The sequence in the documentation presents the flow which you should follow.

<https://developer.verestro.com/books/token-requestor/page/use-cases> In each diagram Issuer = PhonePe in your case

**Q: What is the difference between push provisioning and in-app provisioning?**

A: There is no difference between push and in-app provisioning.

Push Provisioning (or in-app provisioning) provides the ability to initiate the card provisioning process for Apple/Google Wallet directly from the issuer's app.

Users will find the Push Provisioning feature an extremely convenient method to provision their cards or passes into their devices by avoiding the need to input those details manually.

**Q: How we will use Verestro TMP for M4M tokenization's, do we need to configure something on MDES/Verestro side or else?**

A: From Verestro side - we are ready to accept M4M tokenizations. You will have to ask MC for MDES side of configuration.

**Q: Where can we find Verestro TMP public key used for encryption? Can you send us an example of encryption as we cannot find it in documentation?**

A: The JWE certificate will be different for beta and production environment. We will be sending CV requests from the domain <https://bifrost.upaid.pl>

You can get this public key from the API:

```
curl --location 'https://bifrost.upaidtest.pl/issuer/public-keys' \
```

```
--header 'Authorization: Basic ***\
```

**Q: Can you please send, if you have, the example of code written in .Net which is an alternative to this one written in Java, as we cannot use much from this one (we use .Net)?**

A: We don't have any example code written in .Net, only in Java. You can find some examples on the internet - JWE is a common standard and there should be some library for .Net that supports it. But please make sure that you are providing the necessary headers in JWE (ex. iat and kid).

**Q: Can we have native NFC payments (issuer wallet) on Apple devices?**

A: No, Apple doesn't allow such solution. It is possible to pay with NFC on iOS only by using ApplePay. NFC payments (issuer wallet) is only available on Android devices.

**Q: What is manual provisioning and push provisioning to XPays (Apple Pay, Google Pay)?**

A: If you apply for manual provisioning it means that your users will be able to add card and later use it for NFC Payments to XPays by filling in card data manually in XPay application. If you apply for push provisioning it means that your users will be to add card directly from your application by pressing developed button "Add card to <name of the XPay>".

**Q: In what step we get the paymentTokenId?**

A: The paymentTokenId you can get after digitization.

**Q: Is there a callback which inform about SDK initialisation completion?**

A: There is lack of callback related to "SDK initialisation completion" , it is synchronic function, once it is finished it means it was conducted.

**Q: Once we add the card and complete the whole tokenization process , does the pan no (actual card number) gets deleted?**

A: Yes, we do not store PAN after tokenization.

**Q: When user came again to do tokenisation again for same card how can I map this with instrumentId?**

A: When you try to add same PAN again LC API should return error CARD\_ALREADY\_EXISTS Passed card is in the system. Cards number for issuer is unique. You can check detail under below link: <https://developer.verestro.com/books/user-lifecycle-card-management-api-sdk/page/technical-documentation> Method Add Card

**Q: Can delete card only once token is successfully created on app and backend is notified about it? What will be the tokenization flow?**

1. The main flow for tokenization is add card digitize card.
2. When card is added you can get it by MobileDC:CardsService:getAll().
3. Card deletion is possible by MobileDC:CardsService:deletecard(id) or LC API : Delete Card. If removed card was tokenized it automatically removes token on backend side.

**Q: What's difference between below methods?**

1. restart()
2. userService.logout()
3. userService.deleteUser()
4. deviceService.unPairDevice()

Answers:

1. Clear only UCP context like payment tokens, user, cards and session for Mobile DC remains on device
2. Just clears user session token - next actions requires login
3. delete user, tokens, cards and clears SDK data as there is no user
4. Remove binding between user and device and clear SDK data. Requires pairDevice to use SDK

**Q: Is there any method to release SDK (MDC,UCP) resource(like SSE etc) for better resource utilisation ?**

A: SDK doesn't provide such methods. It should be out of the box in JVM. The objects are not cleared, but objects do not keep any data during lifecycle. The data is cleared by GC in context of method usage. Only HCE related methods keeps data context until contactless payment is finished.

---

Revision #2

Created 4 June 2025 10:59:56 by Alicja Switkowska

Updated 4 June 2025 12:16:13 by Alicja Switkowska