

Corporate cards: create, issue and share

This guide describes how to create a corporation, create its balance, issue a virtual corporate card and share that card with an existing end user.

At a glance

Follow the five steps in order. The corporation remains the owner of the balance and card throughout the process.

UML sequence diagram

```
@startuml
actor Integrator
participant "User Lifecycle API" as Lifecycle
participant "Card Issuing API" as Issuing

Integrator -> Lifecycle: POST /lifecycle/v1/corporations
Lifecycle --> Integrator: dcCorporationId
Integrator -> Issuing: POST /secure/balances (dcCorporationId)
Issuing --> Integrator: balanceld
Integrator -> Issuing: POST /secure/cards/virtual
Issuing --> Integrator: cardId
Integrator -> Lifecycle: POST /lifecycle/v1/corporations/{id}/users (dcUserId)
Integrator -> Issuing: PATCH /secure/cards/{cardId}/shares (dcUserId)
Issuing --> Integrator: 204 No Content
@enduml
```

What you will build

At the end of this flow, a corporation owns a balance and a virtual card. A corporate member can access and use the card after it has been shared with their `dcUserId`.

Before you start

- Use the server-to-server API channel. Secure endpoints are protected with an x509 client certificate.
- Have a configured card programme and the required card configuration ID.
- Ensure that each end user already exists in DataCore and has a `dcUserId`.
- Complete KYB where it is required by your programme before enabling financial operations.

Basic flow

Labels are optional. Use this flow when all users in the corporation require the same access level.

1. Create the corporation.

Call `POST /lifecycle/v1/corporations`. Store the returned `dcCorporationId`.

2. Create a corporate balance.

Call `POST /secure/balances` with `dcCorporationId`. Do not use `dcUserId` as the balance owner.

3. Issue a virtual corporate card.

Call `POST /secure/cards/virtual` with the balance ID, `dcCorporationId` and the card configuration ID.

4. Add the end user to the corporation.

Call `POST /lifecycle/v1/corporations/{corporationId}/users` with the user's `dcUserId`.

5. Share the card with the corporate member.

Call `PATCH /secure/cards/{cardId}/shares` with the member's `dcUserId`. The user must be a member first.

Ownership and access

The card is issued to the corporation. Sharing grants a corporation member access without changing ownership.

UML class diagram

@startuml

title Corporate card ownership and access

rectangle "Company" as company

rectangle "Corporate budget" as budget

rectangle "Corporate card" as card

actor "Employee" as employee

company --> budget : owns

budget --> card : funds

company --> employee : adds as member

card --> employee : shares access

note right of employee

The employee can use the card

after access is shared.

The company remains the owner.

end note

@enduml

Key identifiers

Identifier	Purpose
dcCorporationId	Identifies the corporation.
balanceId	Identifies the corporate balance.
cardId	Identifies the issued card when creating a share.
dcUserId	Identifies the end user to add and to receive the card share.

Optional role-based visibility

For different visibility levels, retrieve labels with `GET /lifecycle/v1/labels` and assign the appropriate label while adding a user to the corporation. For example, a `HEAD` user can receive extended visibility according to programme authorisation rules.

Implementation notes

- The card remains owned by the corporation; sharing does not transfer ownership.
- Create corporation membership before calling the share endpoint.
- Use `dcCorporationId` with the corporate `balanceId`; do not substitute the member's `dcUserId`.

API reference

- [User Lifecycle and Card Management API](#) — corporation, membership and labels endpoints.
 - [Card Issuing and Core Banking technical documentation](#) — secure balance, virtual-card and card-share endpoints.
-

Revision #13

Created 2026-10-13 08:04:03 UTC by Wiktor Kowalczyk

Updated 2026-10-13 09:30:25 UTC by Wiktor Kowalczyk