

DigitalBank France

Documentation API REST

Version 1.0 - Janvier 2026

Table des Matières

- 1. Introduction
- 2. Authentification
- 3. Endpoints - Customers
- 4. Endpoints - Accounts
- 5. Endpoints - Transactions
- 6. Endpoints - Cards
- 7. Codes d'Erreur
- 8. Exemples Complets

1. Introduction

Cette API REST fournit un accès sécurisé aux données bancaires de DigitalBank France. Elle est construite sur Supabase (PostgreSQL) avec Row Level Security (RLS) pour garantir que chaque utilisateur n'accède qu'aux données autorisées.

1.1 URL de Base

`https://tzoipnuhurrrqwhjpus.supabase.co/rest/v1`

1.2 Format des Réponses

Toutes les réponses sont au format JSON. Les listes sont retournées sous forme de tableaux.

2. Authentication

Toutes les requêtes (sauf signup/login) nécessitent deux headers obligatoires :

Header	Description
apikey	Clé publique anon de votre projet Supabase
Authorization	Bearer <JWT_TOKEN> obtenu après login

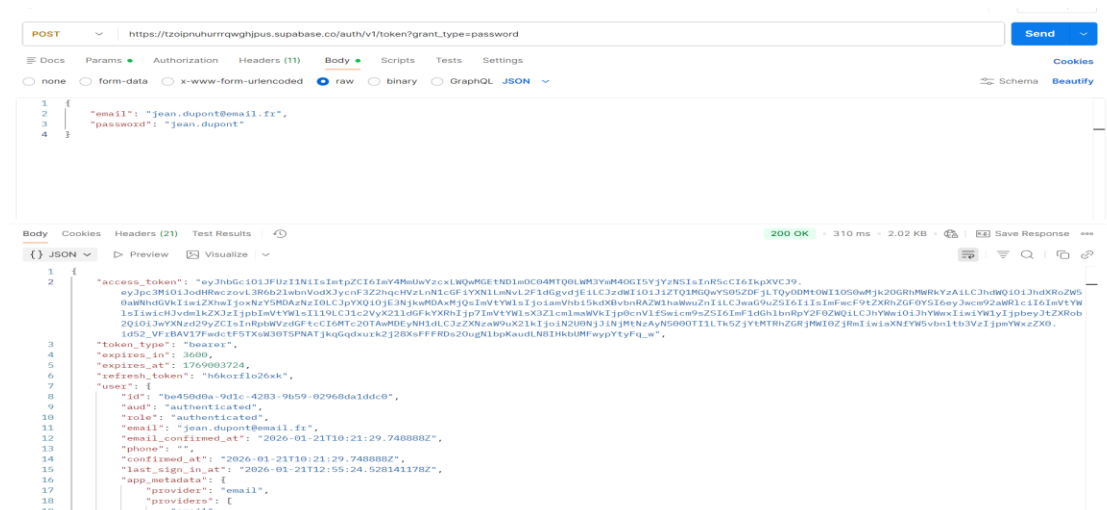
2.1 Login (Obtenir un JWT)

POST /auth/v1/token

Body (JSON) :

```
{  "email": "jean.dupont@email.fr",    "password": " jean.dupont",    "grant_type":  
"password" }
```

Réponse (200 OK) :



3. Endpoint: Customers

3.1 GET /customers

Description:

Récupère la liste des clients selon les permissions RLS.

Permissions:

- admin : Tous les clients
- customer_service : Tous les clients (lecture seule)
- client : Uniquement son propre profil

Paramètres de requête:

Paramètre	Type	Description
select	string	Colonnes à sélectionner (défaut: *)
email	string	Filtrer par email (format: eq.email@domain.com)
limit	integer	Nombre maximum de résultats

Exemple de requête:

GET /customers?select=*&email=eq.jean.dupont@email.fr

Réponse (200 OK):

GET

https://tzoipnuhrrrqwhjpus.supabase.co/rest/v1/customers?select=*&email=eq.jean.dupont@email.fr

Send

Docs

Params

Authorization

Headers (9)

Body

Scripts

Tests

Settings

Cookies

Headers

6 hidden

Key	Value	Description
apikey	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJkdXBhYmFz...	
Authorization	Bearer eyJhbGciOiJIUzI1NiIsImtpZCI6ImY4MmUwYzcxLWQwMG...	
Content-Type	application/json	

Body

Cookies

Headers (21)

Test Results

200 OK

615 ms

1.26 KB

Save Response

JSON

Preview

Visualize

```
1 [
2   {
3     "customer_id": 1,
4     "email": "jean.dupont@email.fr",
5     "password_hash": "$2b$12$abcdefghijklmnopqrstuvwxy12345",
6     "first_name": "Jean",
7     "last_name": "Dupont",
8     "date_of_birth": "1985-03-15",
9     "phone": "0601020304",
10    "address": "12 Rue de la Paix",
11    "city": "Paris",
12    "postal_code": "75001",
13    "country": "France",
14    "created_at": "2026-01-21T10:54:03.740324",
15    "last_login": null,
16    "status": "active"
17  }
18 ]
```

terminal

Runner

Start Proxy

Cookies

Vault

Trash

4. Endpoint: Accounts

4.1 GET /accounts

Récupère les comptes bancaires selon les permissions RLS.

Permissions:

- admin : Tous les comptes
- analyst : Tous les comptes (lecture)
- customer_service : Tous les comptes
- client : Uniquement ses comptes

Exemple : Client récupère ses comptes

```
GET /accounts?select=*
```

Réponse (200 OK):

The screenshot shows a REST client interface for the 'DigitalBank API' with the endpoint 'Client2_Access-Only-Their-Accounts'. The request is a GET to 'https://tzoipnuhrrqwgijpus.supabase.co/rest/v1/accounts'. The response is a 200 OK with a 107 ms latency and 1.11 KB of data. The response body is a JSON array containing one account object.

```
{
  "account_id": 12,
  "customer_id": 9,
  "account_number": "FR7612345678901234567890134",
  "account_type": "checking",
  "balance": 4500.25,
  "currency": "EUR",
  "opened_at": "2026-01-21T10:54:03.740324",
  "status": "active"
}
```

4.2 PATCH /accounts (Modifier)

Modifie le statut d'un compte. Réservé aux admins et customer_service.

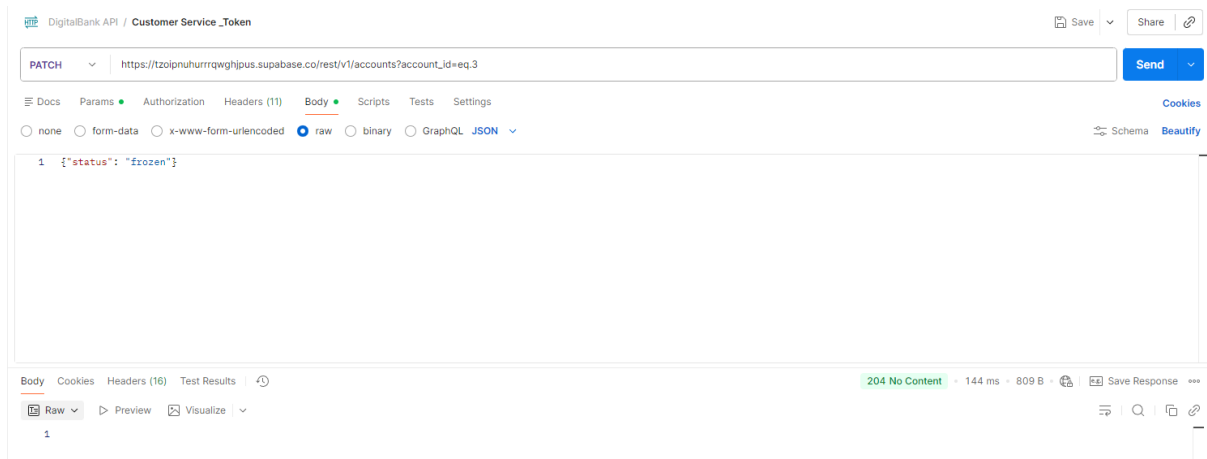
Exemple : Geler un compte

```
PATCH /accounts?account_id=eq.3
```

Body (JSON):

```
{ "status": "frozen" }
```

Réponse (204 No Content) : Modification réussie



5. Endpoint: Transactions

5.1 GET /transactions

Endpoint le plus important pour la détection de fraude.

Permissions:

- admin : Toutes les transactions
- analyst : Toutes les transactions (détection fraude)
- customer_service : Toutes les transactions
- client : Uniquement les transactions de ses comptes

Filtres disponibles:

Paramètre	Exemple
is_fraud	?is_fraud=eq.true
account_id	?account_id=eq.1
order	?order=timestamp.desc
transaction_type	?transaction_type=eq.payment
limit	?limit=10

Exemple 1 : Analyst - Toutes les fraudes

```
GET /transactions?is_fraud=eq.true&order=timestamp.desc
```

Réponse : ~10 transactions frauduleuses

6. Endpoint: Cards

6.1 GET /cards

Récupère les cartes bancaires.

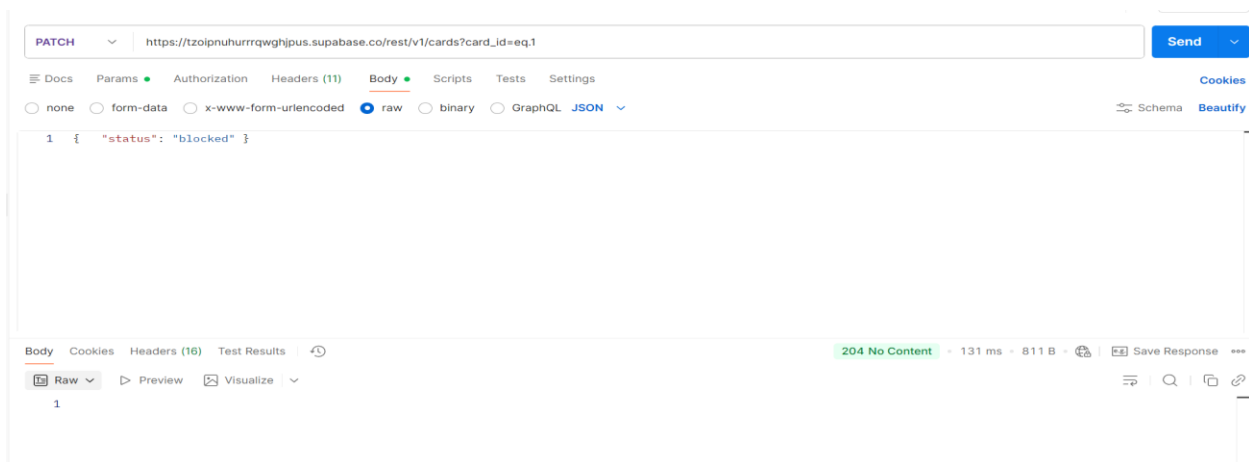
Exemple : Bloquer une carte (customer_service)

```
PATCH /cards?card_id=eq.1
```

Body:

```
{  "status": "blocked" }
```

Réponse : 204 No Content (succès)



7. Codes d'Erreur

Code	Signification
200	OK - Requête réussie, données retournées
204	No Content - Modification réussie (PATCH, DELETE)
400	Bad Request - Paramètres invalides ou body malformé
401	Unauthorized - Token JWT manquant, invalide ou expiré
403	Forbidden - Permissions insuffisantes (RLS bloque l'accès)
404	Not Found - Ressource inexistante
500	Internal Server Error - Erreur serveur PostgreSQL

8. Exemples Complets

8.1 Scénario : Analyst détecte une fraude

Étape 1 : Login

```
POST /auth/v1/token Body: { "email": "analyst@digitalbank.fr", "password": " analyst" } → Récupère access_token
```

Étape 2 : Filtrer les fraudes

```
GET /transactions?is_fraud=eq.true&order=fraud_score.desc&limit=20 Headers: apikey: <anon_key> Authorization: Bearer <access_token> → Retourne les 20 transactions les plus suspectes
```

8.2 Scénario : Customer Service bloque une carte

Étape 1 : Rechercher le client

```
GET /customers?email=eq.jean.dupont@email.fr
```

Étape 2 : Voir ses cartes

```
GET /cards?account_id=eq.1
```

Étape 3 : Bloquer la carte

```
PATCH /cards?card_id=eq.1 Body: { "status": "blocked" } → 204 No Content (succès)
```

8.3 Scénario : Client consulte son solde

```
GET /accounts?select=account_number,account_type,balance,currency Réponse : Uniquement les 2 comptes du client (RLS filtre automatiquement)
```