j’ai mis en place un mécanisme complet d’authentification et de gestion des accès afin de sécuriser les données sensibles (clients, comptes, transactions).
L’approche retenue repose sur quatre étapes :
(a) l’activation de l’authentification multi-facteur,
(b) la mise en place d’un système de rôles,
(c) l’implémentation du Row Level Security (RLS),
(d) et la réalisation de tests de permissions pour valider le bon fonctionnement du dispositif.

a) Configurez l'authentification multi-facteur (MFA)

Nous avons configuré l’authentification multi-facteur directement dans Supabase Auth.
Nous avons activé le mécanisme TOTP (Authenticator App), permettant aux utilisateurs d’utiliser une application d’authentification comme second facteur lors de la connexion.

Cette configuration renforce la sécurité des comptes utilisateurs en ajoutant une couche de protection supplémentaire au mot de passe.

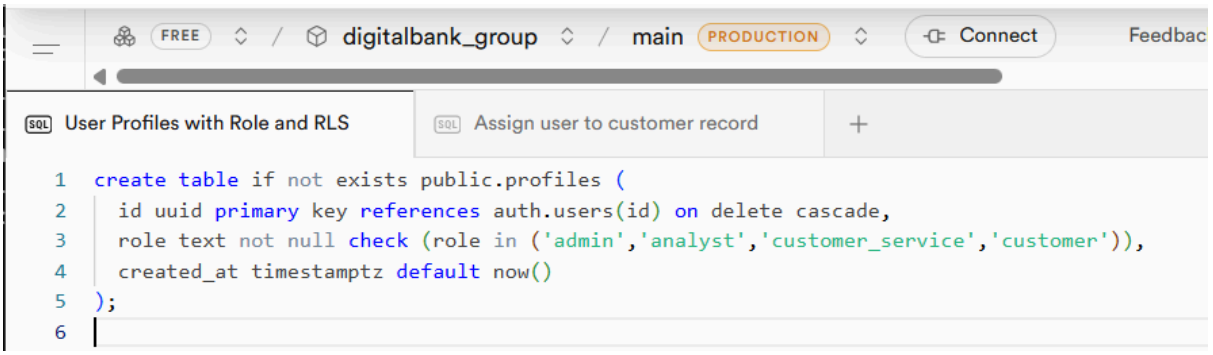
Configuration réalisée

- Interface : Authentication → Multi-Factor
- Option activée : TOTP (App Authenticator)
- SMS MFA non activé (fonctionnalité réservée au plan Pro)

Les captures d’écran jointes montrent que le TOTP est bien activé.

b) Créez un système de gestion des rôles :
Nous avons mis en place un système de gestion des rôles en créant une table applicative profiles, liée à la table auth.users de Supabase.

Création de la table profiles



Attribution des rôles aux utilisateurs

```
insert into public.profiles (id, role)
values
  ('df4e0f78-e2f4-4976-a34d-d9fccd8291d0', 'admin'),
  ('da94d0ce-59a9-433d-b024-5f69c85133f9', 'analyst'),
  ('d34374e6-d8ec-4288-84b6-b8eb6d56b84c', 'customer_service'),
  ('c722a53e-0ca6-402c-b344-d3a49dee3304', 'customer')
on conflict (id) do update set role = excluded.role;
```

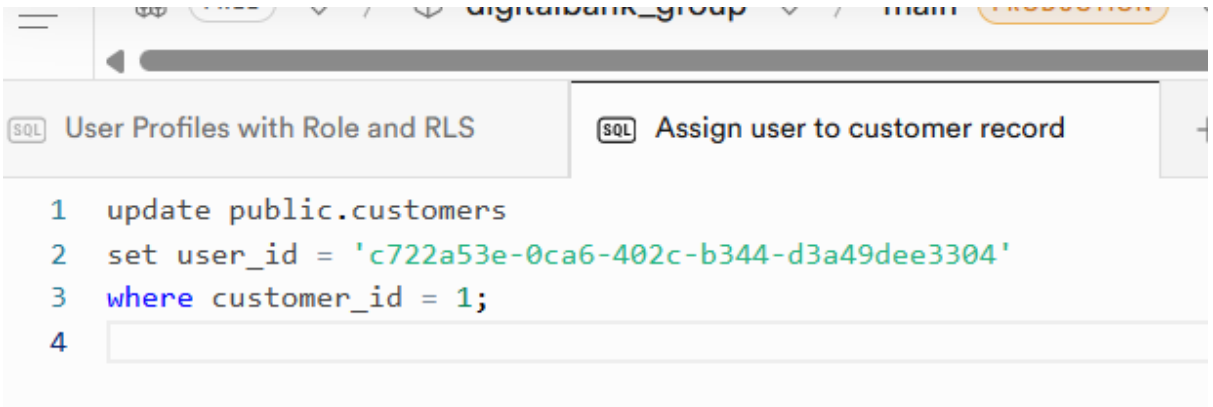
Cette table permet d'associer chaque utilisateur authentifié à un rôle précis.

c) Implémentez Row Level Security (RLS)

Nous avons implémenté le Row Level Security afin de restreindre l'accès aux données directement au niveau de la base de données.

Pour permettre au client d'accéder uniquement à ses propres données, nous avons lié la table customers à l'utilisateur authentifié via le champ user_id.

Liaison d'un client à un utilisateur



Activation du RLS sur les tables sensibles

```sql

alter table public.customers enable row level security;
alter table public.accounts enable row level security;
alter table public.cards enable row level security;
alter table public.transactions enable row level security;

```

Fonction SQL pour récupérer le rôle de l'utilisateur connecté

```sql
create or replace function public.current_role()
returns text
language sql
stable
as $$
  select role from public.profiles where id = auth.uid()
$$;
```

Policies RLS mises en place

Accès aux clients

```sql
create policy "customers_select_policy"
on public.customers
for select
using (
  public.current_role() in ('admin','analyst','customer_service')
  OR (public.current_role() = 'customer' and customers.user_id = auth.uid())
);
```

Accès aux comptes

```sql
create policy "accounts_select_policy"
on public.accounts
for select
using (
  public.current_role() in ('admin','analyst','customer_service')
  OR (
    public.current_role() = 'customer'
    and exists (
      select 1 from public.customers c
      where c.customer_id = accounts.customer_id
        and c.user_id = auth.uid()
    )
  )
);
```

Accès aux transactions

```sql
create policy "transactions_select_policy"
on public.transactions
for select
using (
  public.current_role() in ('admin','analyst','customer_service')
  OR (
    public.current_role() = 'customer'
    and exists (
      select 1
      from public.accounts a
      join public.customers c on c.customer_id = a.customer_id
      where a.account_id = transactions.account_id
        and c.user_id = auth.uid()
    )
  )
);
```

Modification des cartes (droits limités)

```
97
98  create policy "cards_update_policy"
99  on public.cards
100 for update
101 using (
102   public.current_role() in ('admin','customer_service')
103 )
104 with check (
105   public.current_role() in ('admin','customer_service')
106 );
107
```

d) Testez les permissions :

nous avons les permissions en me connectant avec chaque rôle utilisateur et en effectuant des requêtes via l'API REST Supabase, en utilisant un JWT utilisateur.

Le Table Editor n'a pas été utilisé pour ces tests, car il utilise le service role et contourne le RLS.