

Documentation technique et Preuves d'utilisation de Supabase

Explication des Fonctionnalités Implémentées

Gestion des Messages

L'application permet aux utilisateurs de :

- Créer, Afficher leurs propres messages.
- Voir uniquement leurs propres messages.
- Associer un avatar à leur profil.

Authentification et Gestion des Profils

- Utilisation de Supabase Auth pour l'authentification des utilisateurs.
- Chaque utilisateur possède un profil avec un avatar stocké dans Supabase Storage.
- Seul l'utilisateur connecté peut modifier son avatar.

Sécurité et Accès aux Données

- Mise en place de Row-Level Security (RLS) pour restreindre l'accès aux données.
 - Un utilisateur ne peut modifier ou supprimer que ses propres messages.
-

Architecture du Projet

Schéma des Interactions

L'application suit une architecture basée sur **Next.js** et **Supabase** :

1. **Front-end (Next.js) :**
 - Interface utilisateur développée avec NextJs et Tailwind CSS.
 - Gestion des états avec useState et useEffect.
 - Appels API vers Supabase via supabase.from('table').select().

2. Back-end (Supabase - PostgreSQL) :

- Gestion des utilisateurs via Supabase Auth.
- Stockage des messages et avatars dans Supabase Database et Storage.
- Application des règles de sécurité avec RLS.

3. Flux des données :

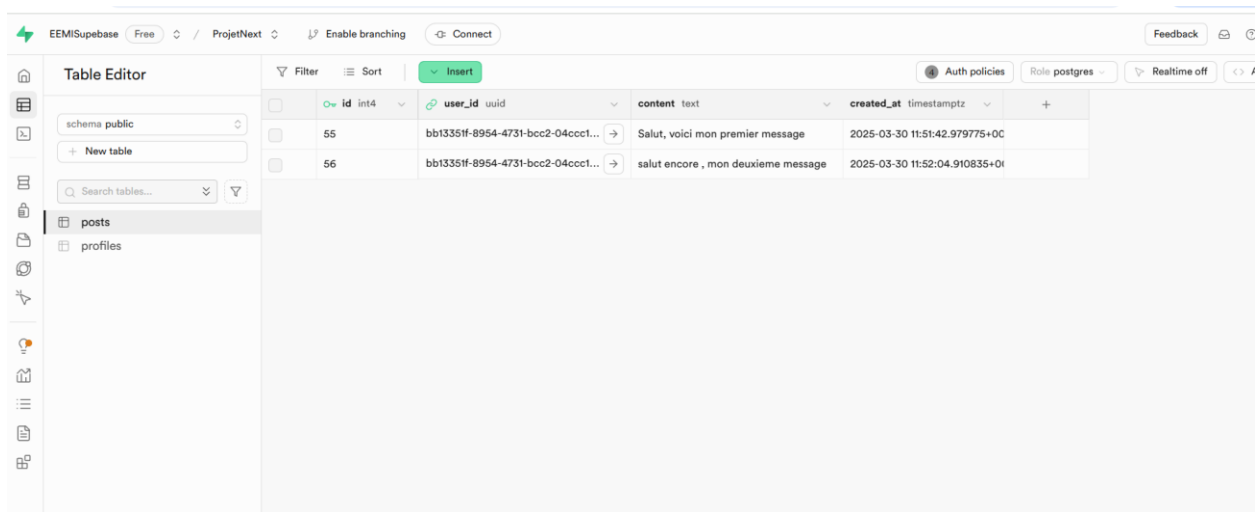
- Un utilisateur s'authentifie via Supabase Auth.
- Il peut poster un message qui est stocké dans la table `posts`.
- L'affichage des messages est filtré pour ne montrer que ceux de l'utilisateur connecté.
- Les avatars sont stockés dans Supabase Storage et accessibles via une URL sécurisée

Justification des règles de sécurité RLS mises en place :

Les règles de sécurité RLS sont mises en place pour restreindre l'accès aux données d'un utilisateur uniquement à ses propres informations. Par exemple, dans la table `profiles`, la règle `auth.uid() = id` permet à un utilisateur de voir et de modifier uniquement son profil. De même, pour la table `posts`, la condition `auth.uid() = user_id` garantit que chaque utilisateur peut interagir uniquement avec ses propres publications. Ces règles assurent la confidentialité des données et préviennent tout accès non autorisé.

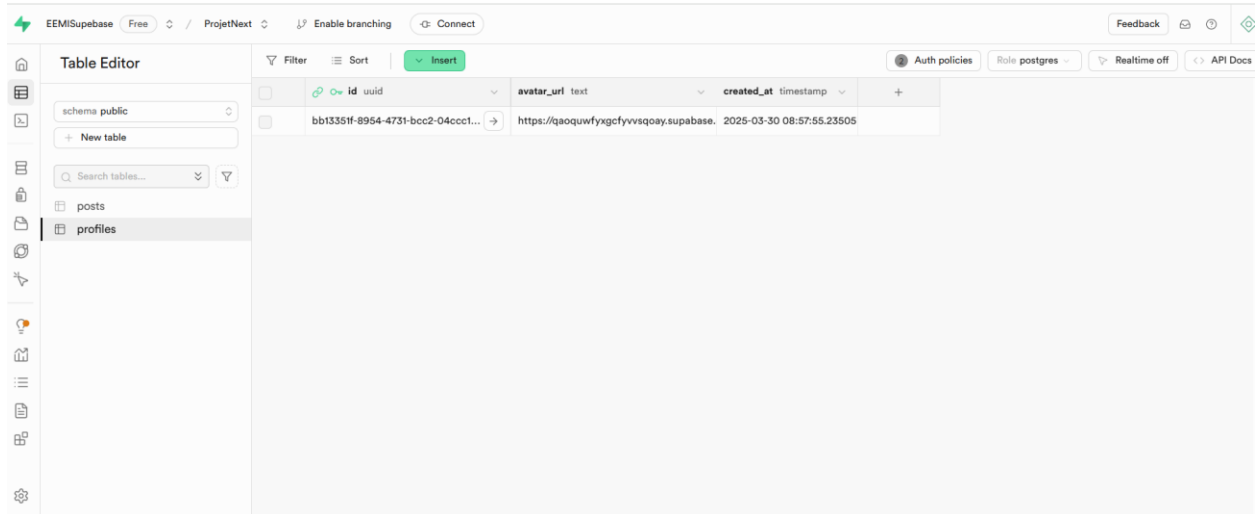
Capture d'écran des tables PostgreSQL configurées dans Supabase:

Table Posts :



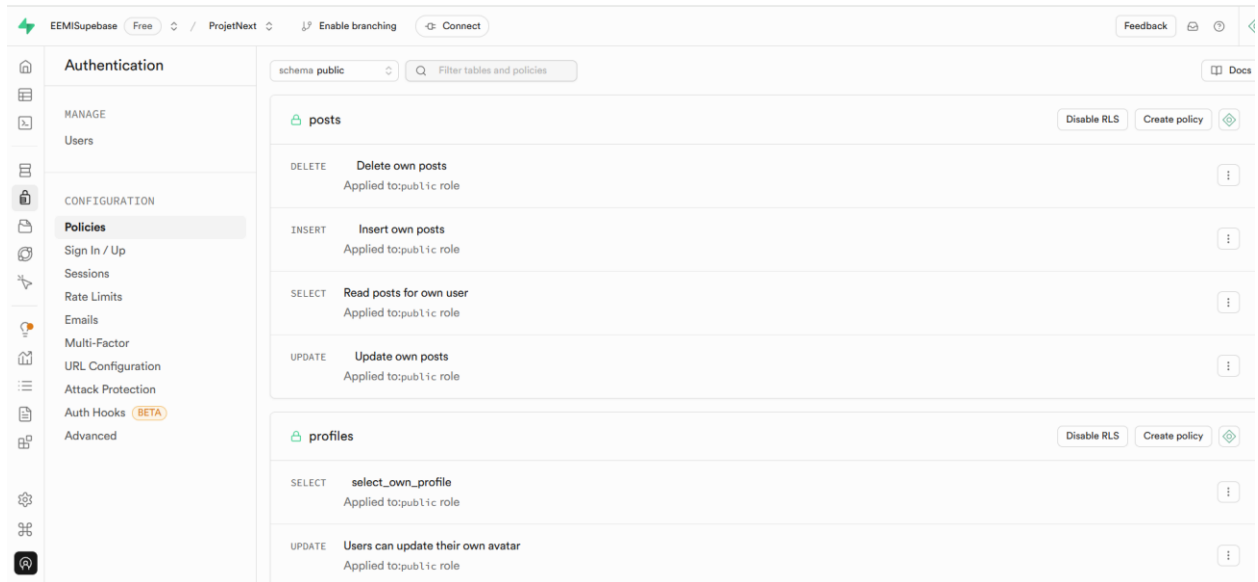
	id int4	user_id uuid	content text	created_at timestampz	
	55	bb13351f-8954-4731-bcc2-04ccc1...	Salut, voici mon premier message	2025-03-30 11:51:42.979775+00	
	56	bb13351f-8954-4731-bcc2-04ccc1...	salut encore , mon deuxieme message	2025-03-30 11:52:04.910835+01	

Table Profiles :

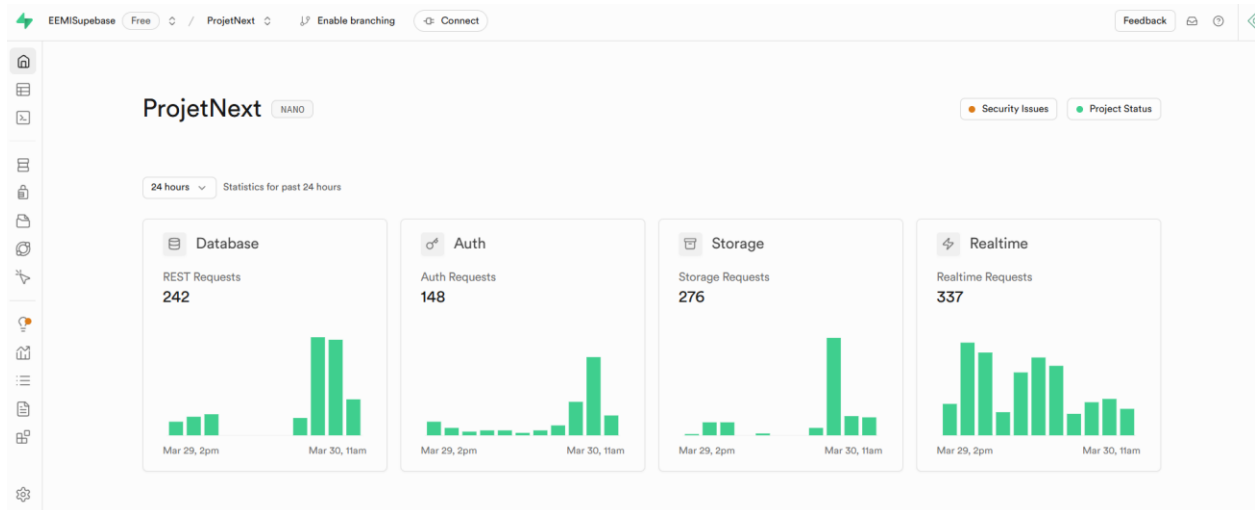


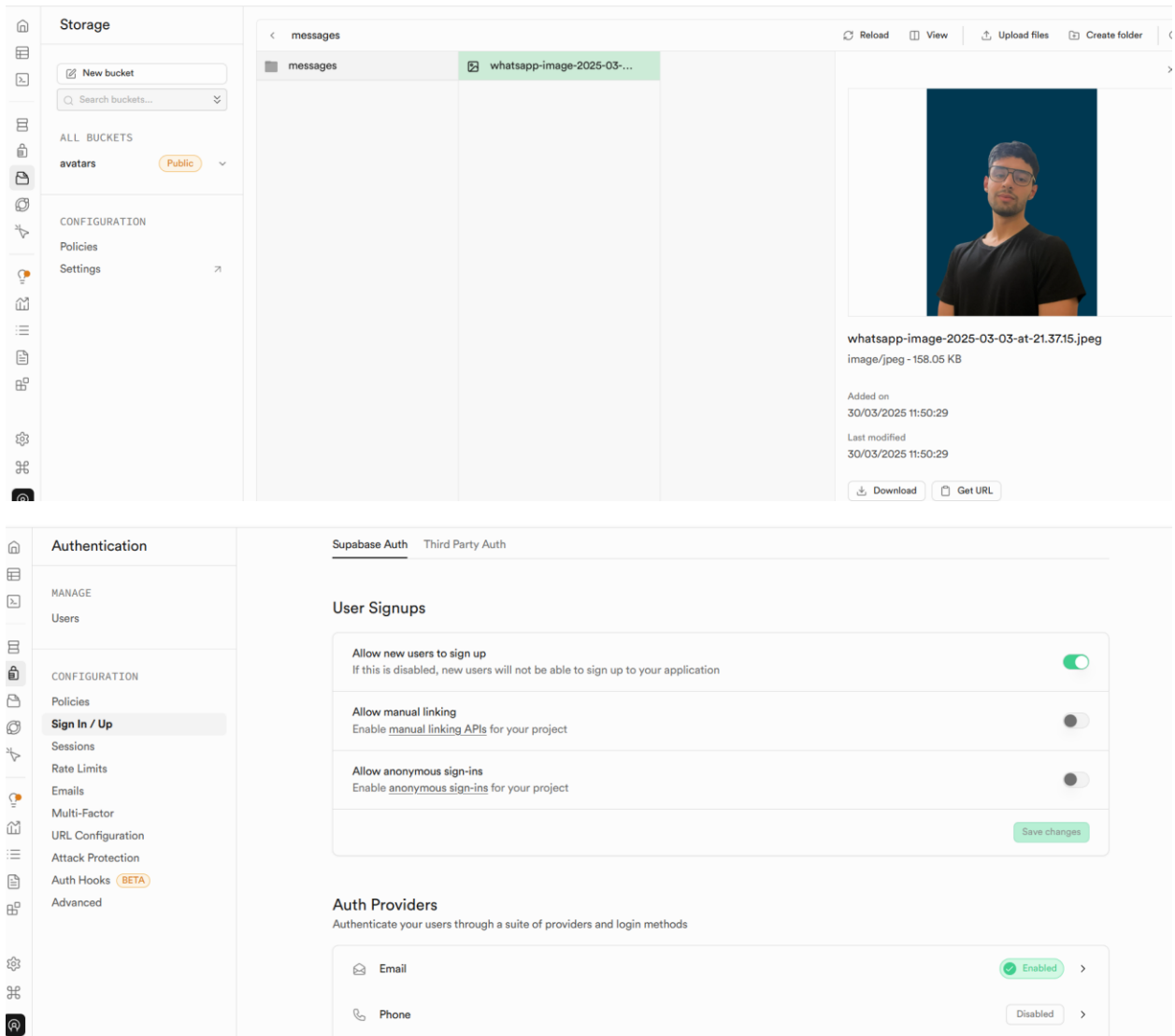
	id uuid	avatar_url text	created_at timestamp	
	bb13351f-8954-4731-bcc2-04ccc1...	https://qaoquwfyxgcfyvvsqay.supabase.	2025-03-30 08:57:55.23605	

2- Capture d'écran des règles de sécurité RLS appliquées:



Capture d'écran du dashboard de Supabase avec les paramètres de l'authentification et du stockage.





Résultat de l’affichage dans la webView page d’accueil d’un user authentifié :

