Using Supabase in Python

Supabase is an open-source alternative to Firebase that provides a PostgreSQL database, authentication, storage, and serverless functions. In this guide, we will see how to use Supabase with Python to manage data in a database.

Installation

First, let's install the Supabase Python client:

```
pip install supabase
```

Configuration

To connect to Supabase, you need to obtain the URL and API key from your Supabase project.

```
from supabase import create_client, Client

url = "https://your-project.supabase.co"
key = "your-anon-key"
supabase: Client = create_client(url, key)
```

Inserting Data

To add data to a table, we use the `insert` method:

```
data = {"nome": "Mario", "cognome": "Rossi", "email":
"mario.rossi@example.com"}
```

```
response =
supabase.table("utenti").insert(data).execute()
print(response)
```

Reading Data

To read data from a table, we use the `select` method:

```
response =
supabase.table("utenti").select("*").execute()
print(response.data)
```

Updating Data

To update a specific record, we can use the `update` method:

```
response = supabase.table("utenti").update({"email":
"nuova.email@example.com"}).eq("nome",
"Mario").execute()
print(response)
```

Deleting Data

To delete a record, we use the `delete` method:

```
response =
supabase.table("utenti").delete().eq("nome",
"Mario").execute()
print(response)
```

Authentication Management

Supabase provides an authentication system based on email and password. To register a user:

```
response = supabase.auth.sign_up({"email":
"utente@example.com", "password": "password123"})
print(response)
```

To log in:

```
response =
supabase.auth.sign_in_with_password({"email":
"utente@example.com", "password": "password123"})
print(response)
```

Conclusion

Supabase is a powerful platform that simplifies database and authentication management in Python. With this guide, you have the basics to start using Supabase in your project.