

HAPPY DAYS README GUIDE

Note!

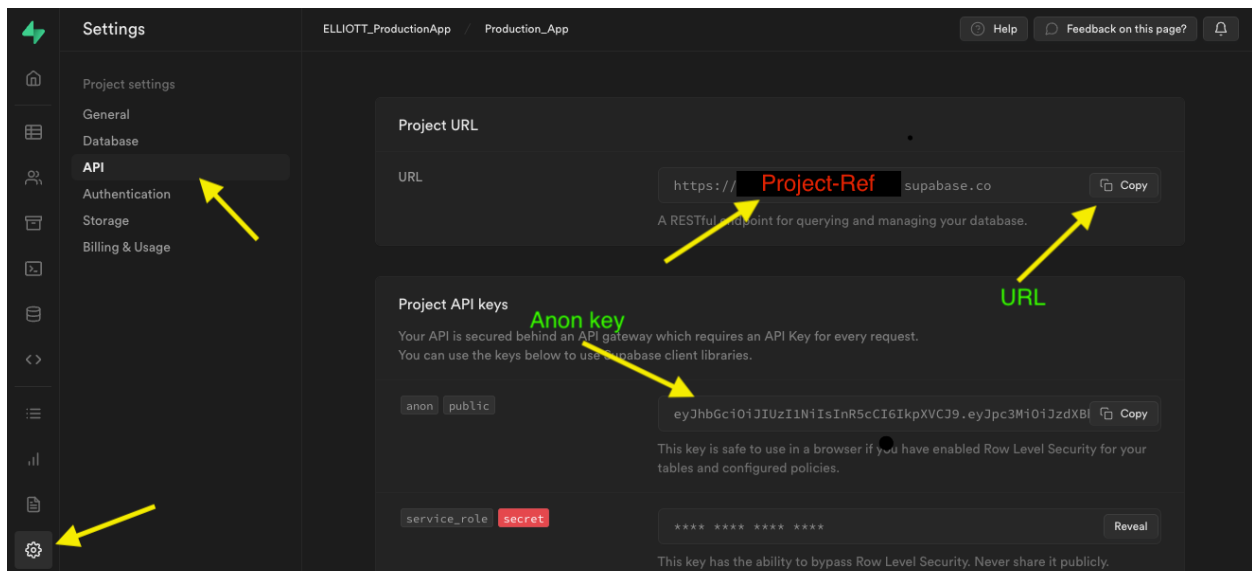
Use a reference guide and notes. I haven't had time to test everything out and ensure things are bug-free and correct. So don't drive yourself crazy debugging because there are probably some errors. Use as notes, and check work. Let me know if there are any problems or add your own notes.

ACCOUNTS REQUIREMENTS

- Supabase free account
- Cloudflare workers
- GitHub

SUPABASE REFERENCES (Copy these to note pad will be used many times in the guide)
(See screenshots below as a visual reference to locate)

- Supabase URL (supabase/settings/api)
- Supabase Anon Key (Supabase/setting/api)
- Supabase Project Ref (supabase/settings/api)



SUPABASE GITHUB AUTH PROVIDER (GITHUB SETUP)

- In your GitHub profile, on the right side of the nav bar, go to Settings / Developer Settings / OAuth apps
- Register a New OAuth application
- Name: your choice

- Homepage URL = Supabase URL
- Authorization Callback = Supabase URL + /auth/v1/callback

Register a new OAuth application

Application name *

Happy-days-Remix-App

Something users will recognize and trust.

Homepage URL *

https://ProjectRef.supabae.co

The full URL to your application homepage.

Application description

Application description is optional

This is displayed to all users of your application.

Authorization callback URL *

https://ProjectRef.supabae.co/auth/v1/callback

Your application's callback URL. Read our [OAuth documentation](#) for more information.

☐ Enable Device Flow

Allow this OAuth App to authorize users via the Device Flow. Read the [Device Flow documentation](#) for more information.

Register application Cancel

Settings / Developer Settings / OAuth Apps

Supabase URL

Supabase URL + /auth/v1/callback

Settings / Developer settings / Supabase-happydays-testing

General

Optional features

Advanced

Supabase-happydays-testing

Etdev2 owns this application.

Transfer ownership

You can list your application in the [GitHub Marketplace](#) so that other users can discover it.

List this application in the Marketplace

1 user

Revoke all user tokens

Client ID

44cb339c0db3362d2cfe

Generate a key

Client secrets

Generate a new client secret

*****34091a96

Added 15 hours ago by Etdev2

Last used within the last week

You cannot delete the only client secret. Generate a new client secret first.

Delete

Copy

SUPABASE GITHUB AUTH PROVIDER (SUPABASE SETUP)

- After setting up GitHub OAuth copy **Client ID** AND copy the generated **client secrete**

-
- Authentication
- ELLIOTT_ProductionApp / Production_App
- Help Feedback on this page?
- General
- Users
- Policies
- Logs
- Configuration
- Settings
- Email Templates
- Providers
- Azure Disabled
- Bitbucket Disabled
- Discord Disabled
- Facebook Disabled
- GitHub Enabled
- GitHub enabled
- Client ID
- b6ede413312f2186
- Client Secret
-
- Redirect URL
- https://acdhwunlmjletxnddzq.supabase.co/auth/v1/callback Copy
- Cancel Save
- GitLab Disabled
- Authentication
- Settings
- Client id GitHub
- Generated Client Secret GitHub
- Scroll down

- <https://github.com/dijonmusters/happy-days>

Search or jump to... Pull requests Issues Marketplace Explore

dijonmusters / happy-days Public

Clone or Download Zip

<> Code Issues 2 Pull requests 1 Actions Projects Security Insights

main 2 branches 0 tags

Go to file Add file Code

About

No description, website, or topics provided.

Readme 16 stars 3 watching 7 forks

Releases

No releases published

Packages

No packages published

Contributors 3

dijonmusters Jon Meyers

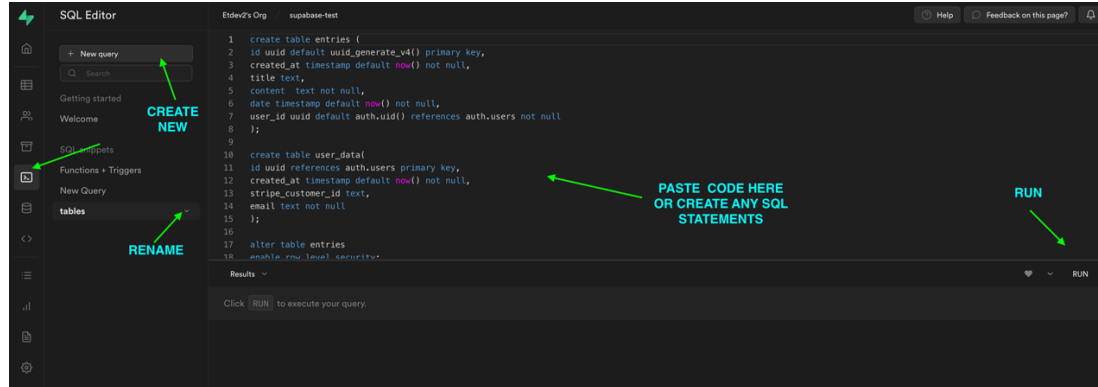
pryley Paul Ryley

etonomick Valery Stepanov

File	Commit	Time
app	Happy Hour #18	6 days ago
public	happy hour #10	2 months ago
styles	happy hour #10	2 months ago
supabase	Happy Hour #18	6 days ago
.eslintrc	happy hour #10	2 months ago
.gitignore	add auto-generated css file to gitignore	6 days ago
README.md	happy hour #10	2 months ago
package-lock.json	happy hour #17	13 days ago
package.json	happy hour #17	13 days ago
remix.config.js	happy hour #10	2 months ago
remix.env.d.ts	happy hour #10	2 months ago
server.js	happy hour #10	2 months ago
tailwind.config.js	Basic styles on all entries pages	28 days ago
tsconfig.json	happy hour #10	2 months ago
wrangler.toml	Happy Hour #18	6 days ago

- In wrangler.toml under vars replace SUPABASE_URL and SUPABASE_ANON_KEY with your supabase keys found in the dashboard.
- Continue with VS code and remix after the database is configured.

SETUP SUPABASE WITH THE SQL



CREATING A TYPE THAT DOES NOT EXIST IN SUPABASE WITH SQL

- RUN first in a separate query.

```
create type subscription_tier as enum('FREE', 'STADARD', 'PREMIUM')
```

Full SQL (Tables, function, triggers)

- Run either FULL SQL or SQL FOR TABLES AND POLICIES.
- Manually Add [] as default. For the assets_urls column
-

```
create table entries (
  id uuid default uuid_generate_v4() primary key,
  created_at timestamp default now() not null,
  title text,
  content text not null,
  date timestamp default now() not null,
  user_id uuid default auth.uid() references auth.users not null,
  asset_urls text [] not null
);

create table user_data
(
  id uuid references auth.users primary key,
  created_at timestamp default now() not null,
  stripe_customer_id text,
  email text not null,
  subscription_tier subscription_tier default 'FREE' not null
);

alter table entries
  enable row level security;

CREATE POLICY "Authenticated users can see their own entries" ON "public"."entries"
AS PERMISSIVE FOR SELECT
TO authenticated
USING (user_id = auth.uid());

CREATE POLICY "users can update their entry" ON "public"."entries"
AS PERMISSIVE FOR UPDATE
TO authenticated
```

```

USING (user_id = auth.uid())
WITH CHECK (user_id = auth.uid());

CREATE POLICY "Authenticated users can insert own data" ON "public"."entries"
AS PERMISSIVE FOR INSERT
TO authenticated
WITH CHECK (user_id = auth.uid());

alter table user_data
  enable row level security;

create function public.handle_new_user()
returns trigger as
$$
begin
  insert into public.user_data(id,email)
  values(new.id,new.email);
  return new;
end;

$$
language plpgsql security definer;

create trigger on_insert_auth_user
after insert on auth.users
for each row
execute procedure public.handle_new_user();

```

SQL FOR TABLES and POLICES Only (user_data, and entries)

- Run will have to manually add handle_new_user function and on insert

```

create table entries (
id uuid default uuid_generate_v4() primary key,
created_at timestamp default now() not null,
title text,
content text not null,
date timestamp default now() not null,
user_id uuid default auth.uid() references auth.users not null
);

create table user_data(
id uuid references auth.users primary key,
created_at timestamp default now() not null,
stripe_customer_id text,
email text not null
);

alter table entries
enable row level security;

CREATE POLICY "Authenticated users can see their own entries" ON "public"."entries"
AS PERMISSIVE FOR SELECT
TO authenticated
USING (user_id = auth.uid());

CREATE POLICY "users can update their entry" ON "public"."entries"
AS PERMISSIVE FOR UPDATE
TO authenticated
USING (user_id = auth.uid())
WITH CHECK (user_id = auth.uid());

CREATE POLICY "Authenticated users can insert own data" ON "public"."entries"
AS PERMISSIVE FOR INSERT
TO authenticated

```

```
WITH CHECK (user_id = auth.uid());

alter table user_data
enable row level security;
```

Run Remix to create a new user trigger.

- If you have not cloned the repository, follow the directions above.
- Before starting make sure you completed (OAuth, Created tables (entries, user_data, RLS)
- From the cloned repository replace SUPABASE_KEY and Anon_key in wangler.toml.
- RUN npm install to install packages, npm audit fix to install dependencies
- RUN **npm run dev to check if OAuth works**
- If a user is created the user will show up in the database in supabase/authentication/users.
- If the user is created continue to set up a function, that creates a user in the user_data table, triggered by the login.

SETTING UP FUNCTIONS AND TRIGGERS (MANUALLY)

- In the supabase dashboard go to Supabase/ Database / In the menu below, you will see Triggers / Functions and Database Webhooks.
- Create the **handle_new_users** function **first**
- After the function is created you can connect the **handle_new_users** function to the **on_insert_auth_user** trigger. To connect the trigger-function use the drop-down menu “functions to trigger” in the trigger and you will see the function handle_new_users.

Add a new function

Return type: **trigger**

Arguments
Arguments can be referenced in the function body using either names or numbers.

+ Add a new argument

Definition
The language below should be written in 'plpgsql'.
Change the language in the Advanced Settings below.

```

1
2
3
4
5
6
7
begin
  insert into public.user_data(id,email)
  values (new.id,new.email);
  return new;
end;

```

plpgsql function that inserts id and email into user_data table

☒ Show advanced settings
These are settings that might be familiar for postgres heavy users

Language: **plpgsql**

Behavior: **volatile**

Config Params
+ Add a new config

Type of security

SECURITY INVOKER
Function is to be executed with the privileges of the user that calls it.

SECURITY DEFINER
Function is to be executed with the privileges of the user that created it.

Add a new Trigger

Name of trigger: **on_insert_auth_user**
The name is also stored as the actual postgres name of the trigger. Do not use spaces/whitespace.

Conditions to fire trigger

Table: **users**
This is the table the trigger will watch for changes. You can only select 1 table for a trigger.

Events
The type of events that will trigger your trigger

☒ **Insert**
Any insert operation on the table

☐ **Update**
Any update operation, of any column in the table

☐ **Delete**
Any deletion of a record

These are the events that are watched by the trigger, only the events selected above will fire the trigger on the table you've selected.

Trigger type: **After the event**
This determines when your Hook fires

Orientation: **Row**
Identifies whether the trigger fires once for each processed row

Function to trigger
Choose a function to trigger

SETTING UP FUNCTION + TIGGER IN SQL EDITOR

```

create function public.handle_new_user()
returns trigger as
$$
begin
insert into public.user_data(id,email)
values(new.id,new.email);
return new;
end;

$$
language plpgsql security definer;

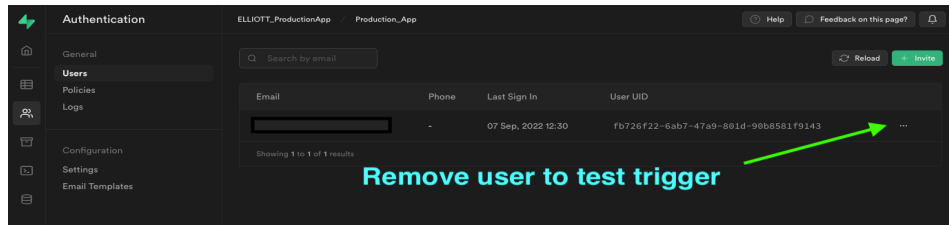
create trigger on_insert_auth_user
after insert on auth.users
for each row
execute procedure public.handle_new_user();

```


Removing user data to triggers, functions, Edge functions

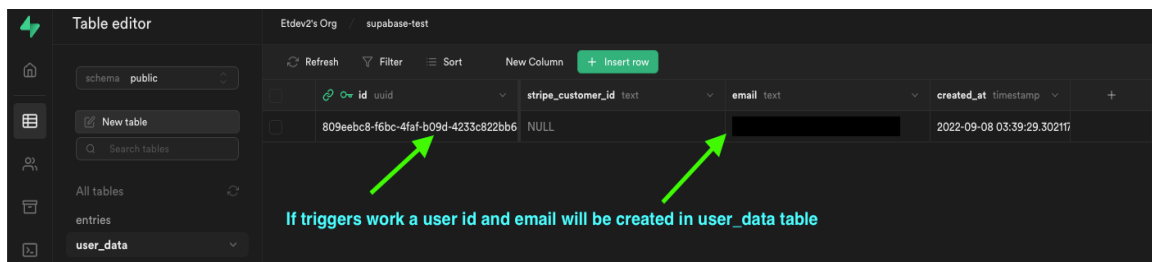
- **REMOVE** all the user data from the supabase including the user before testing the trigger

-



-

- **RUN** `npm run dev`
- If everything is working properly after signing in to the remix app. A user with `user_id` will be written into the `user_data` base with `stripe_customer` null.



-

SETUP STORAGE

- Create a new storage bucket named **assets**
- Allow all operations
- Added the below definitions in Storage policies

```
bucket_id = 'assets'  
and auth.uid()::text = (storage.foldername(name))[1]
```

Adding new policy to assets

Policy name

A descriptive name for your policy

34/50

Allowed operation

Based on the operations you have selected, you can use the highlighted functions in the [client library](#).

☒ SELECT

☒ INSERT

☒ UPDATE

☒ DELETE

upload

download

list

update

move

copy

remove

createSignedUrl

createSignedUrls

getPublicUrl

Target roles

Apply policy to the selected roles

×

Policy definition

Provide a SQL conditional expression that returns a boolean.

Defines the folder structure

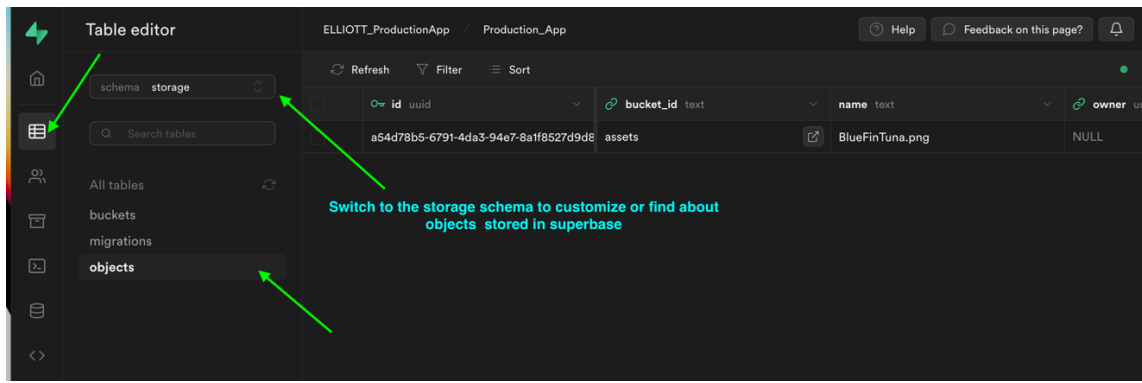
```
1 bucket_id = 'assets'
2 and auth.uid()::text = (storage.foldername(name))[1]
```

View templates

Review

Storage File Structure Defined by policy

- Assets/
 - /user uuid
 - /entries uuid
 - /supabeats.mp3
 - /user2 uuid
 - /entries uuid
 - Dog-pic.png



SUPABASE EDGE FUNCTIONS AND SUPABASE CLI (Run Code In Vscode Terminal Exclude Run In The Terminal)

STEPS TO THE EDGE

Follow the steps below are more detailed (You will have to mess around with things because sometimes the functions will not work as expected when adding **DATABASE WEBHOOKS**)

1. If you forked or cloned happy-days repository. Then Remove index.ts from the supabase folder, place it somewhere or copy code from the clipboard.

(Run the following in the terminal)

2. supabase init
3. supabase link
4. supabase functions new create-stripe-customer
5. supabase functions deploy create-stripe-customer
6. curl -L -X POST invoke the function
7. Create database webhook in supabase
8. Add back the code form index.ts from the create-stripe-custom folder
9. Deploy supabase edge function again
10. If there are users in the database, delete users.
11. Npm run dev and login
12. If the user is found in user_data with stripe_customer_id = null triggers are working continue and remove all user data from auth and user_data table again.
13. Kill server
14. supabase secrets list
15. supabase secrets set STRIPE_KEY = sk_13234
16. supabase secrets list again and check
17. Copy code from code in supabase/functions/index from Happys-14 repo (the first function on top of index starts with const stripe = stripe(Deno.env.get....))
18. supabase functions deploy create-stripe-customer

19. Finally, `npm run dev`, start server, log in. If no errors everything probably works, check in supabase dashboard in the `user_data` table to see if the user was created with a `stripe_customer_id`, if yes, then a customer will be inserted into the stripe dashboard.

- <https://supabase.com/docs/guides/cli>
- **Move temperately** create-stripe-customer out of the superbase file before creating the edge function.
- **RUN** `supabase INIT`
- **RUN** `supabase link --project-ref [Project-ref]`
- After linking the project you can exclude your project ref when deploying the edge function
- **RUN** `supabase functions new create-stripe-customer` (Creates)
- **RUN** `supabase functions deploy create-stripe-customer --project-ref [project-ref]` (deploys)

Edge Functions Beta create-stripe-customer

Metrics Details Invocations Logs

Function Name	create-stripe-customer
Status	active
Endpoint URL	https://bnhiugttacqojodmxsnf.functions.supabase.co/create-stripe-customer
Created At	Wednesday, September 7, 2022 10:10 PM
Updated At	Wednesday, September 7, 2022 10:10 PM
Version	v1
Regions	Earth All functions are deployed globally

Command line access

Deployment management

- > Deploy a new version
\$ `supabase functions deploy create-stripe-customer`
- > Delete the function
\$ `supabase functions delete create-stripe-customer`

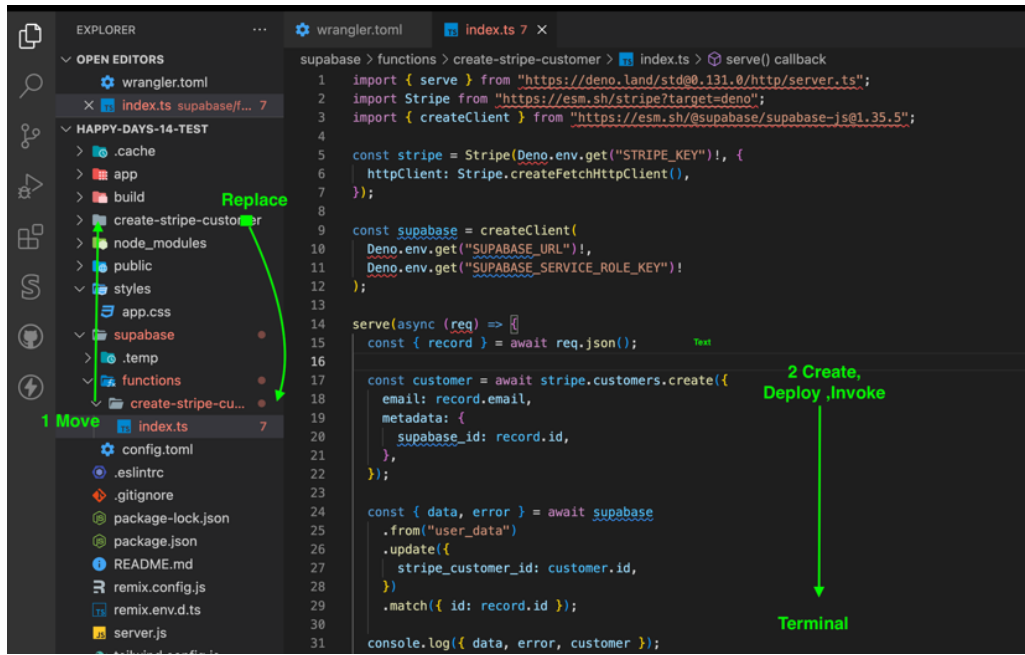
Invoke

- > Invoke your function
\$ `curl -L -X POST 'https://bnhiugttacqojodmxsnf.functions.supabase.co/create-stripe-customer' -H 'Authorization: Bearer [YOUR ANON KEY]' --data '{"name":"Functions"}'`

Secrets management

- > View all secrets
\$ `supabase secrets list`
- \$ `supabase secrets set NAME1=VALUE1 NAME2=VALUE2`
- > Unset secrets for your project
\$ `supabase secrets unset NAME1 NAME2`

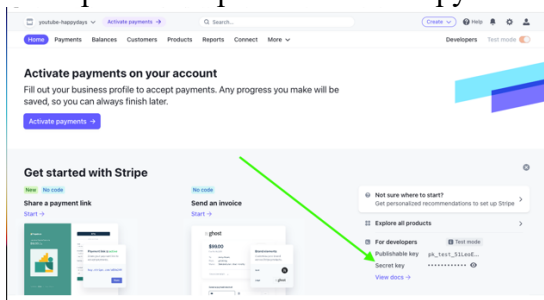
- **RUN** `curl -L -X POST 'https:// [project-ref].functions.supabase.co/create-stripe-customer' -H 'Authorization: Bearer Anon_Key --data '{"name":"Functions"}'` (Invokes)



- **Replace** the create-stripe-customer folder, that was moved earlier in vs code

CREATE A STRIPE SECRETE KEY THAT CAN BE CALLED BY FUNCTION

- Create a stripe account
- In stripes developer dashboard copy secret key



- In vscode terminal **Run: supabase secrets list** (get a list of keys stored in supabase)
- To set stripe key in Supabase **Run: supabase secrets set STRIPE_KEY=sk_1234** in terminal
- **Run: supabase secrets list** to check if stripe key was stored

CREATE A SUPABASE DATABASE WEBHOOK

- **Setting up this webhook, there is you could run into problems, I don't know why but I had issues, getting everything to get these working. Everything was right but a customer stripe id would not be created. So you may need to tweak things around**
- Create a webhook to call the Edge function that will create a stripe customer in stripe and in supabase when a new user is inserted into the database.
- Delete user_data run npm run dev, login and if everything works you should have created a stripe customer in user_data and stripe dashboard

Environment Variables, keys, and things

(FILL OUT THESE use as clipboard)

Supabase

SUPABASE_PROJECT_REF=

SUPABASE_URL =

SUPABASE_ANON_KEY=

GITHUB

CLIENT ID=

SECRET=

Supabase variables used by Edge functions

(Set in the terminal with **supabase secrets set name2=value**)

- Supabase secrets set **STRIPE_SIGNING_SECRET=**
- supabase secrets set **STRIPE_KEY=**

.env

(set in a .env in root)

STRIPE_SECRET=

wrangler.toml

SUPABASE_URL =

SUPABASE_ANON_KEY=

Supabase Edge functions

supabase functions new **create-stripe-customer**

create-stripe-customer_URL =

supabase functions new **create-stripe-checkout**

create-stripe-checkout_URL =

supabase functions new **stripe-webhooks.**

create-stripe-webhooks_URL =

