

Supabase Edge Function Project Structure

TypeScript edge functions on Deno with Supabase client, shared code, and local dev workflow.

Updated 2025-12-22

#supabase #edge-function #deno #typescript #serverless

PNG

PDF

Copy

Prompt

Project Directory

supabase/

- functions/ Edge functions ...
 - hello-world/ Example function
 - index.ts Function entry ...
 - send-email/ Email sending f...
 - index.ts
 - stripe-webhook/ Payment webhooks
 - index.ts
 - _shared/ Shared code (un...
 - supabase-client.ts Initialized cli...
 - cors.ts CORS headers he...
 - types.ts Shared type def...
- supabase/ Supabase config
 - config.toml Project configu...
- migrations/ Database migrat...
 - 20240101000000_init.sql
 - seed.sql Seed data
- .env.local Local secrets
- import_map.json Deno import ali...
- deno.json Deno workspace ...
- README.md

Why This Structure?

Each function lives in its own folder under `functions/`. The `_shared/` folder (prefixed with underscore) contains reusable code that won't be deployed as a separate function. Functions run on Deno Deploy's edge network for low latency globally.

Key Directories

- `functions/` - Each subfolder becomes a deployed function
- `functions/_shared/` - Shared utilities, not deployed independently
- `supabase/` - Database migrations and project config

Basic Edge Function

```
import { serve } from "https://deno.land/std/http/server.ts"
import { createClient } from "jsr:@supabase/supabase-js@2"
import { corsHeaders } from "../_shared/cors.ts"

serve(async (req) => {
  const supabase = createClient(
    Deno.env.get("SUPABASE_URL")!,
    Deno.env.get("SUPABASE_SERVICE_ROLE_KEY")!
  )
  const { data } = await supabase.from("users").select()
  return new Response(JSON.stringify(data), {
    headers: { ...corsHeaders, "Content-Type": "application/json" }
  })
})
```

Getting Started

- `supabase init` to create project
- `supabase functions new hello-world`
- `supabase start` for local dev with Docker
- `supabase functions serve` to test locally
- `supabase functions deploy hello-world`

Best Practices

- Use `_shared/` for common code like CORS and clients
- Keep functions small and focused (one responsibility)
- Use `Deno.env.get()` for secrets, never hardcode
- Return proper CORS headers for browser requests
- Use `import_map.json` for cleaner imports

When To Use This

- Serverless API endpoints without managing servers
- Webhook handlers for Stripe, GitHub, etc.
- Background tasks triggered by database changes
- Auth-related logic like custom JWT claims
- Integrating third-party services securely

Trade-offs

- Cold starts** - First request may be slower (~50-200ms)
- Execution limits** - 2-second wall time for free tier
- Deno only** - No Node.js APIs, use Deno or npm: specifier

Local Development

Run `supabase start` to spin up local Postgres, Auth, Storage, and Edge Functions. Use `supabase functions serve --env-file .env.local` to test functions with hot reload.