PocketBase Project Structure

The open-source backend in one file. A pure Go application with embedded SQLite and Svelte Admin UI, designed for portability and simplicity.

Updated 2025-12-30

#pocketbase #go #sqlite #svelte #monolith #embedded #baas

PNG PDF Copy Prompt

Project Directory

```
pocketbase/
  core/ Core framework …
    app.go Main applicatio…
    db.go DB connection l…
    record_model.go Dynamic record …
  apis/ HTTP API Handle…
    record_crud.go
    realtime.go
  cmd/ CLI Commands
    serve.go Start server co…
  ui/ Admin Dashboard…
    src/
    embed.go Embeds UI into …
    package.json
  tools/ Utilities
    auth/ OAuth providers
    mailer/ Email sending
    cron/ Job scheduling
  plugins/ Extensions
    jsvm/ JavaScript runt…
  migrations/ Go-based DB mig…
  main.go Entry point
  go.mod Go dependencies
```

Repository Info

Repository - pocketbase/pocketbase

Stars - 40k+

License - MIT

Last Analyzed - December 2025

Tech Stack

Language - Go

Database - SQLite (Embedded)

Admin UI - Svelte 4 (SPA)

Scripting - JavaScript (Goja)

Build - Vite (Frontend)

Architecture Notes

PocketBase is a true 'Monolith'. The Go backend, SQLite database engine, and Svelte frontend are all compiled into a single executable binary. It uses the `embed` package to bundle the compiled frontend assets. The `core` package serves as a framework that can be imported and extended in other Go projects.

Key Directories

core/ - The heart of PocketBase. It defines the `App` struct, the DB wrapper, and the event hooks system.

apis/ - Contains the REST API handlers. Each file roughly corresponds to a resource (records, collections, auth).

ui/ - The Admin Dashboard source code. It's a standard Svelte project that gets built and then embedded into the Go binary.

Why This Structure?

PocketBase demonstrates the power of the 'One Binary' philosophy. By avoiding external dependencies like a separate database server or object storage, it drastically simplifies deployment. It's a perfect example of how to build a portable, self-contained application using Go.