DayBoard Build Guide

Overview

DayBoard is a local?first personal dashboard application for macOS/iOS (SwiftUI) with a Go backend and Supabase Postgres database. It integrates Google Calendar, Plaid, and Google Distance Matrix to show your next meeting, upcoming subscription bills, commute estimate, and after?tax pay outlook. This guide explains how to set up the required external accounts, configure environment variables, run the backend, and build the SwiftUI client.

1. External accounts

Supabase

Create a project at `https://supabase.com`. From the project?s API settings, record your **Project URL** (`SUPABASE_URL`) and **Anon/public API Key** (`SUPABASE_SERVICE_KEY`). For server?side operations you may prefer the Service Role key (found under Settings ? API ? Service role key) because it bypasses row?level security.
Database URL ? In Supabase, open Settings ? Database ? Connection string and copy the `postgresql://?` connection string. This will be used as `DATABASE_URL`.

Google Cloud

Create or select a project at `https://console.cloud.google.com`. Enable the **Google Calendar API** and **Distance Matrix API**. Under APIs & Services ? Credentials click ?Create Credential ? OAuth Client ID.? Choose ?Web application.? Set authorized redirect URIs to `http://localhost:8080/auth/google/callback` for local development and your production URL for deployment (e.g. `https://yourbackend.fly.dev/auth/google/callback`). Record the **Client ID** (`GOOGLE_CLIENT_ID`) and **Client Secret** (`GOOGLE_CLIENT_SECRET`). Create an API key for the Distance Matrix API (`MAPS_API_KEY`).

Plaid

Sign up for a Plaid developer account at `https://dashboard.plaid.com`. Create a new development application and note the **Client ID** (`PLAID_CLIENT_ID`) and **Secret** (`PLAID_SECRET`). Set the environment to `sandbox` for testing or `development` for live data. Define a redirect URI `http://localhost:8080/auth/plaid/callback` and record it as `PLAID_REDIRECT_URI`.

2. Configure environment

Copy `dayboard/backend/.env.example` to `.env` and fill in the placeholders:

```
PORT=8080
DATABASE_URL=postgresql://...             # Supabase connection string
SUPABASE_URL=https://...supabase.co       # your project URL
SUPABASE_SERVICE_KEY=...                  # service role key or anon key
GOOGLE_CLIENT_ID=...
GOOGLE_CLIENT_SECRET=...
GOOGLE_REDIRECT_URI=http://localhost:8080/auth/google/callback
PLAID_CLIENT_ID=...
PLAID_SECRET=...
PLAID_ENV=sandbox
PLAID_REDIRECT_URI=http://localhost:8080/auth/plaid/callback
MAPS_API_KEY=...
JWT_SECRET=supersecretkey                 # choose any random secret
```

Make sure your `.env` file remains private and never commit it to version control.

3. Database migrations

The backend uses Postgres tables defined in `dayboard/backend/migrations/0001_create_tables.sql`. To create them:

```
cd dayboard/backend
# install goose (Go database migration tool) if not already installed
go install github.com/pressly/goose/v3/cmd/goose@latest
# run migrations
goose -dir migrations postgres "$DATABASE_URL" up
```

Alternatively, you can open the Supabase SQL editor and execute the contents of the SQL file directly.

4. Run the backend locally

From the `dayboard/backend` directory:

```
go mod tidy
go run cmd/server/main.go
```

The server listens on `PORT` (default `8080`). Available endpoints include:

- `/agenda/today?user_id=<uuid>` ? return today?s calendar events with join URLs.
- `/subs?user_id=<uuid>` ? get subscriptions; `POST /subs` creates a manual subscription.
- `/estimate/taxes` ? compute taxes given income, state, and filing status.
- `/commute/estimate` ? estimate commute distance/time and cost.
- `/profile` ? get or update your profile (addresses, pay, hours/week, food cost).

Use curl or Postman to test: `curl 'http://localhost:8080/agenda/today?user_id=YOUR_UUID'`.

5. Build and run the SwiftUI client

---

1. Open Xcode and create a new ?App? project. Replace the default `App` and `ContentView` with the contents of `dayboard/client/DayBoardApp.swift`.
2. Add a view model file (e.g., `DayBoardViewModel.swift`) using the provided view model code. Set `baseURL` to your backend URL (e.g., `http://localhost:8080`) and `userID` to a UUID that corresponds to an existing user.
3. Grant network permissions in the macOS/iOS app (App Sandbox / App Transport Security if needed).
4. Build and run the macOS target. You should see a menu bar item showing your next meeting, commute estimate, bills, and pay outlook. For iOS, run on a device via Xcode or distribute through TestFlight (requires Apple Developer Program).

Testing tips

- Use Plaid sandbox to link a dummy bank account and generate recurring transactions.
- Create events in Google Calendar for today with Meet or Zoom links to test agenda display.
- Provide sample addresses (e.g., ?123 Main St, Indianapolis, IN? ? ?456 Company Rd, Indianapolis, IN?) to test commute estimation.
- Watch backend logs in the terminal; errors are printed when external API calls fail.

Deployment

When you are ready to deploy, push the Go backend to Fly.io, Render, Railway, or Cloud Run. Set the same environment variables in your hosting provider?s configuration panel. Update OAuth redirect URIs in Google and Plaid to point to your deployed backend (e.g., `https://dayboard.fly.dev/auth/google/callback`). Finally, notarize your macOS app and attach the .dmg to a GitHub Release. If you wish to offer an iOS version to testers, enroll in the Apple Developer Program and use TestFlight.