OGAP – Technical Documentation

1. Project Overview

- **Purpose**: OGAP is a community-driven parking finder that lets drivers hand over on-street parking spots in real time via a credit-based handshake flow and live maps.
- **Problem**: Reduces time wasted searching for parking by matching leaving drivers with arriving drivers and discouraging freeloading through credits.
- **Target users**: Urban drivers, campus visitors, and neighborhoods with scarce street parking; mobile users supported via Capacitor shells.

2. Feature Overview

- **Live map with availability (core)**: Displays `parking_spots` with age-based probability coloring and user location; hides spots already in a handshake flow (src/components/Map.tsx).
- **Handshake flow (core)**: Offer/request/accept/cancel/complete spot handovers; completion transfers the spot and updates credits (src/hooks/useHandshake.ts, supabase/functions/process-credits/index.ts).
- **Credits economy (core)**: Balance fetch, realtime updates, and credit transactions; deductions/awards handled via Supabase Edge Function (src/hooks/useCredits.ts, supabase/functions/process-credits/index.ts).
- **Presence (optional/nice-to-have)**: Online user count via Supabase realtime presence (src/hooks/usePresence.ts).
- **Session persistence (core)**: Local parking session saved to localStorage and restored on reload (src/pages/Index.tsx).
- **Account area (core)**: Profile (display name), credit balance/history, parking history, and placeholder credit purchase UI (src/pages/Account.tsx).
- **Landing & marketing (optional)**: Localized landing page with DE/EN toggle (src/pages/Landing.tsx, src/contexts/LanguageContext.tsx).
- **Mapbox token provisioning (core)**: Edge Function proxy with manual fallback input cached in localStorage (src/components/Map.tsx, supabase/functions/mapbox-proxy/index.ts).
- **Cleanup jobs (operational)**: Edge Function to prune stale parking spots and cancel expired handshakes (supabase/functions/cleanup-parking-spots/index.ts).
- **Mobile shells (optional)**: Capacitor iOS/Android scaffolding (ios/).

3. System Architecture

- **Client**: React (Vite) SPA with React Query for data fetching, React Router for routing, Tailwind + shadcn/ui for UI primitives.
- **Backend**: Supabase (Postgres, Auth, Realtime, Edge Functions). Tables include `parking_spots`, `handshake_deals`, `user_credits`, `credit_transactions`, `parking_history`, `credit_packages`, `profiles` (see supabase/migrations).
- **Realtime**:
  - Postgres changes channels for `parking_spots`, `handshake_deals`, `user_credits`.
  - Presence channel for online user count.
- **Edge Functions**:
  - `mapbox-proxy`: returns Mapbox token or proxies Mapbox requests.
  - `process-credits`: validates user via JWT, manages credit balance, records transactions, and completes handshakes.
  - `cleanup-parking-spots`: scheduled cleanup of stale spots and deals.

- **Map flow**: Mapbox GL JS initialized once token available; markers rendered from parking and handshake data; current location marker updates.
- **Data flow (simplified)**: User auth → Supabase session → hooks query tables → realtime subscriptions feed UI → user actions (offer/request/complete) hit Edge Functions or table updates → changes fan out via realtime channels to all clients.

4. Tech Stack

- **Languages**: TypeScript (frontend), SQL (Supabase), Deno TypeScript (Edge Functions).
- **Frameworks/Libraries**: React 18, Vite, React Router, React Query, Tailwind + shadcn/ui (Radix primitives), Mapbox GL JS, date-fns.
- **Backend services**: Supabase Auth, Postgres, Realtime, Edge Functions.
- **Mobile**: Capacitor (iOS/Android scaffolds).
- **Tooling**: ESLint, SWC React plugin, tailwindcss-animate.

5. Project Structure

- src/pages: Routes (`Landing` , `Index` , `Account` , `NotFound`).
- src/components: UI atoms/molecules (map, dialogs, credit display, probability bar, auth, etc.).
- src/hooks: Domain hooks (`useCredits` , `useHandshake` , `usePresence` , UI toast).
- src/integrations/supabase: Supabase client and typings.
- src/lib: Utilities such as haversine distance and parking probability model.
- supabase/functions: Edge Functions (`mapbox-proxy` , `process-credits` , `cleanup-parking-spots`).
- supabase/migrations: Database schema migrations.
- ios: Capacitor iOS project.
- Root configs: Vite, Tailwind, ESLint, Capacitor, npm scripts in package.json.

6. Setup & Installation

- **Prerequisites**: Node.js 20+, npm; Supabase project; Mapbox public token; optional Xcode/Android Studio for mobile builds.
- **Environment variables**:
  - App: `VITE_SUPABASE_URL` , `VITE_SUPABASE_PUBLISHABLE_KEY` in `.env.local` .
  - Supabase Edge Functions: `MAPBOX_ACCESS_TOKEN` (mapbox-proxy), `SUPABASE_URL` , `SUPABASE_ANON_KEY` , `SUPABASE_SERVICE_ROLE_KEY` .
- **Install**: `npm install` .
- **Run dev server**: `npm run dev` (open printed localhost URL).
- **Lint**: `npm run lint` .
- **Build**: `npm run build` ; preview: `npm run preview` .
- **Capacitor sync**: `npm run build && npx cap sync` .

7. Core Logic & Key Modules

- **Map rendering & token flow**: src/components/Map.tsx fetches Mapbox token via `mapbox-proxy` , falls back to manual input, renders parking and handshake markers, and keeps a manual pin aligned to map center.
- **Parking session & orchestration**: src/pages/Index.tsx handles geolocation, parking session persistence, toast feedback, dialogs for auth/handshake/leaving, and ties map interactions to spot selection and handshake flows.
- **Credits**: src/hooks/useCredits.ts fetches balance from `user_credits` , subscribes to realtime changes, and invokes `process-credits` for deductions/refunds/awards.

- **Handshake lifecycle**: src/hooks/useHandshake.ts fetches active deals, subscribes to realtime updates, and exposes actions (offer/request/accept/decline/complete/cancel) backed by `handshake_deals` and `process-credits` .
- **Probability model**: src/lib/parkingProbability.ts computes availability probability based on elapsed minutes and peak/off-peak ranges; used for marker coloring and UI bars.
- **Cleanup**: supabase/functions/cleanup-parking-spots/index.ts deletes stale `parking_spots` and cancels expired open `handshake_deals` , re-opening spots.
- **Mapbox proxy**: supabase/functions/mapbox-proxy/index.ts serves token and proxies Mapbox requests with CORS headers.
- **Credit processing**: supabase/functions/process-credits/index.ts authenticates user, ensures a credits record, and handles `check_balance` , `parking_used` , `new_spot_reported` (+2), `complete_handshake` (giver +20, receiver -10, close spot, record history).

8. API / Interfaces

- **Supabase tables (selected)**:
  - `parking_spots` : `{id, latitude, longitude, available, available_since}` .
  - `handshake_deals` : `{id, spot_id, giver_id, receiver_id, status, latitude, longitude, departure_time, created_at}` .
  - `user_credits` , `credit_transactions` , `parking_history` , `profiles` , `credit_packages` .
- **Edge Functions**:
  - `mapbox-proxy` (no auth): `POST { action: 'get-token' }` → `{ token }` ; or proxy `{ url }` .
  - `process-credits` (JWT required): `POST { action: 'check_balance' | 'parking_used' | 'new_spot_reported' | 'complete_handshake', spotId?, dealId? }` .
  - `cleanup-parking-spots` (service role): scheduled/cron invocation; not intended for client calls.
- **Authentication**: Supabase email/password (UI via AuthDialog) with session stored in Supabase client; auth state gates protected flows.
- **Realtime**: Supabase `channel` subscriptions for changes on `parking_spots` , `handshake_deals` , `user_credits` , plus presence channel for online users.

9. Configuration & Customization

- **Mapbox**: Set `MAPBOX_ACCESS_TOKEN` secret; users can override via in-app token input stored in localStorage.
- **Credit amounts**: Hardcoded in `process-credits` (+20/-10 handshake, +2 report); adjust in the Edge Function.
- **Cleanup thresholds**: Day vs night thresholds (30 vs 90 minutes) configured in supabase/functions/cleanup-parking-spots/index.ts.
- **UI theme**: Tailwind and shadcn; adjust globals in src/App.css and src/index.css.
- **Language copy**: Extend translations in src/contexts/LanguageContext.tsx and component-level text.

10. Known Limitations & Assumptions

- Mapbox token is returned to the client; relies on a public token and local caching.
- Credit purchase UI is disabled; no payment integration.
- Geolocation denial falls back to default coordinates; experience degrades without location.

- Handshake completion assumes giver triggers completion; receiver debit happens server-side without explicit confirmation.
- Cleanup requires external scheduling (cron) to invoke `cleanup-parking-spots`.
- Limited offline support; errors are toast-first with minimal retry/backoff.

11. Future Improvements & Roadmap (inferred)

- Integrate payments for purchasing credit packages.
- Enhance handshake UX: timers, auto-complete based on departure time, dual confirmation before credit transfer.
- Secure Mapbox access by proxying tiles/styles and adding rate limiting; avoid exposing token to clients.
- Add scheduled cron in Supabase for `cleanup-parking-spots`.
- Expand analytics: heatmaps and predictive models beyond static probability ranges.
- Improve offline/caching (service workers, optimistic updates).