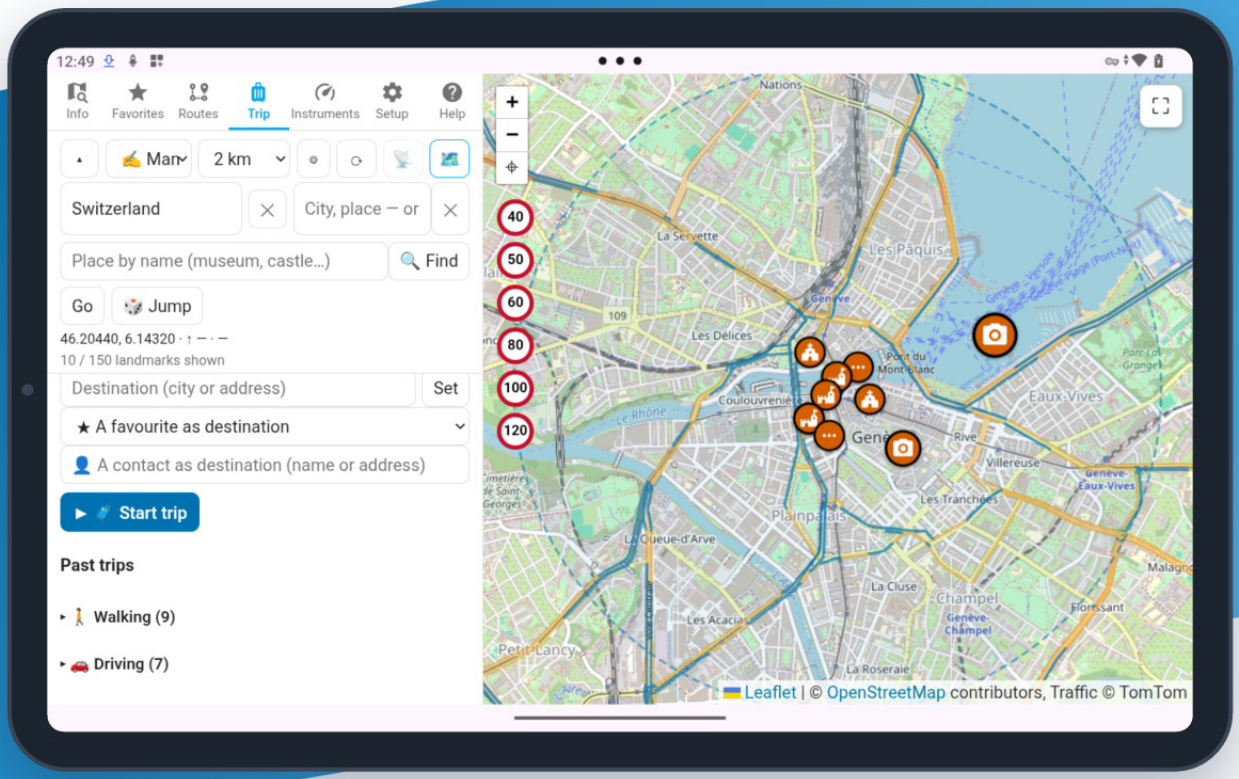


Nestour GPS

Technical Reference

Every place worth a stop, on your way



Version 0.15.0 · 5 November 2026

© 2026 The Development Team

Nestour GPS — Technical Reference

© 2026 The Development Team. All rights reserved.

Applies to Nestour GPS 0.15.0 (5 November 2026) · beta, expires 2027-02-28 · repository `tourist-explorer` · English

This is the maintainer's reference for **Nestour GPS**, the standalone edition of Tourist Explorer: a Tauri 2 app for Android (Windows for development, **NEW** Linux as an AppImage) that runs the same card as the Home Assistant integration, with an in-process port of the integration's Python engine. Code is pointed at by `file` + function or constant, which survives later edits (line numbers went stale with every release). The HA-era reference (`docs/TECHNICAL-REFERENCE.md`) still describes the shared parts (stars, place sources, navigation links, motorway points); this document repeats only what the app does differently or what a maintainer needs on hand. « **NEW** » marks what changed in 0.13.0 to 0.15.0.

NEW Changes in 0.15.0

0.15.0 (5 November 2026) — the beta key (app)

- **NEW** **Beta key = signed licence**: `base64url(payload JSON) "." base64url(Ed25519 signature over the first segment's bytes)`. Rust checks it offline (`lib.rs verify`, command `licence_check`, `ed25519-dalek`): the payload's `kid` picks the public key in `LICENCE_KEYS` (now `2026-11`, signed on nestour.app). `beta.ts licenceEdition` reads `v` = 1, `tier`, `exp` (unix s, required), `ends`, `lang`, `lim` into the edition; a `lim` key left out is not limited.
- **NEW** **Start**: `backend.ts loadLicence` (awaited by `main.ts` before the card) reads the key from localStorage `tourist-explorer-licence`. A beta build starts as `NO_KEY` (fails closed) and takes the key's edition when it is valid. Setup → 🔑 Keys & AI → 🎟️ Beta key: `licence_save` checks and stores it, the card reloads. A full build (no `EDITION`) ignores keys (shortcut, until the paid tiers).
- **NEW** **No key** = the simple GPS: `call` answers only `NO_KEY_SERVICES` (data, map, position, speed signs, recording a trip, backup), else `NO_KEY`; `tile_get` serves only `tile.openstreetmap.org`; `libraryOf` shows `map_style` standard and no hiking trails; the card hides ⛰️ (no `map3d`), the other map styles and the logbook exports.
- Not checked by the app: `cov`, `add`, `dev` (devices are counted on the server), `lid`, `iat`.

NEW Changes in 0.14.0

0.14.0 (5 November 2026) — the beta edition (app)

- NEW **Edition switch**: `EDITION=beta` at build time (`vite.config.ts` → `__EDITION__`) picks `BETA` in `app/src/beta.ts`; any other value, or none, picks `FULL` (no limits). One object: `{ tier, lang, lim: { trips, routes, plans, fav }, keyExport }`, the shape a signed licence key will carry later. Build: `EDITION=beta npx tauri build` / … `android build`; the Linux CI job sets it.
- NEW **Limits** (`beta.ts` `limitHit`, checked in `backend.ts` `call`): a new trip (`trip_start`; the running one counts but is never offered), route (`route_save` without id), plan (`plan_save` without id) or favourite (`favorite_add`, a new id) past `lim` throws `LIMIT {json}` with the oldest item and its delete service. The card's `call` turns that into a popup (`limitReached`): « Remove the oldest » deletes that one and retries, « See the licences » opens nestour.app/gallery.html. Nothing is deleted without the tap; edits, imports and data above a limit pass.
- NEW **Languages**: `lang` is the App language list (`main.ts` → card `languages`); a stored `ui_lang` outside it reads as "" (`allowedLang`, in `libraryOf` and `main.ts`), so the device's language, else English. Description languages (`languages` from `info.LANGUAGES`) are untouched.
- NEW **Key export**: `keyExport: false` greys out ⬇ Export keys file (card config `keyExport`) and `call` refuses `keys_export`.
- NEW **Feedback and new versions**: Help → « ✉ Send feedback » opens `nestour.app/feedback.html?v=&p=&l=` (`main.ts`, `about.feedback`). `release_check` (`backend.ts` `releaseCheck`) GETs `nestour.app/releases.json` at most once a day (cached in localStorage `tourist-explorer-release`; `version` checked as digits and dots), unless the setting `update_check` (Setup → 📦 Your data, default on) is off; Help shows « 0.x.y is out » with nestour.app/download.html. The card's new wording is in `card/src/edition-i18n.ts` (en, fr, fi, zh; other languages read English).

NEW Changes in 0.13.3

0.13.3 (5 November 2026) — export keys (app)

- NEW `keys_export` (`backend.ts` `keysExport`): this device's keys.json, without the Google sign-in in progress (`pending`), in the shape `keys_import` reads; « No keys on this device yet » when empty. Setup → 🔑 Keys & AI « ⬇ Export keys file » hands it to `saveFiles` (Download/<export folder>, Share in the popup). The card holds the text only for that call; Android gives no other way to the app's own files (release builds are not debuggable).

NEW Changes in 0.13.2

0.13.2 (5 November 2026) — automatic zoom (both editions), lanes and exit signs (app)

- NEW **Auto-zoom**: `card/src/zoom.ts` `targetZoom` (pure) and `settle` (calm-down: a new zoom must differ by 0.5 and hold `CALM_MS` 3 s; quarter levels, the map has `zoomSnap: 0.25`). `card.ts` `autoZoom` calls it in `setFix` while `following()`. Inputs: speed; `road.highway` / `road.limit` from `road_at` (TomTom's motorway sections on the route give `motorway`); `radius`; the map's shorter side; `guideAheadM` while the banner shows; `nearPlaceM` (trip stops and ★★★ places within `approach_km`, at most 1 km). Rules: `speedZoom` table (Waze-like, 0 km/h → 17, 130 → 13); motorway/trunk: the wider of that and the radius × `RADIUS_MARGIN` 1.3; town (residential, living street, service, unclassified, or a limit ≤ 50): at least 16; junction within max(300 m, 12 s): fits me + the junction + 150 m; place: fits me + it + 100 m; capped 17.5. A zoom by hand (wheel, two-finger touch, double tap, the ± control) sets `zoomHeld` until ⌖ (`recenter`); Leaflet's zoom events start on the next frame and cannot tell ours from the user's. The car stays centred (not in the lower third) so the radius circle shows whole. Settings `auto_zoom` (`off`/`speed`/`smart`), `zoom_radius`, `zoom_junctions` (app), `zoom_places`.
- NEW **Lanes**: `GuideStep.lanes` = [{dirs, take}] left to right. TomTom: `guideUrl` adds `sectionType=lanes`; `parseGuide` gives each manoeuvre the last `LANES` section starting after the previous manoeuvre and at or before it (`directions` → dirs lower-cased, `follow` present → take). OSRM: `steps[].intersections[0].lanes` (`indications`, `valid`; `none` → straight). The banner (`laneSign`) shows the lanes and, for an exit or fork with a number or towns, an exit sign (green in `GREEN_SIGNS` countries, else blue) from 1 km (exits; or the exit warning distance) / 300 m. Setting `guide_lanes`. `laneSeparators` are not drawn.

NEW Changes in 0.13.1

0.13.1 (5 November 2026) — the place's own article before English, « translate with my AI » (both editions)

- NEW **Native article nearby** (the team's Hamburg test, November 2026: places without Wikidata showed English or OpenTripMap text, never German). `info.fetchInfo` / `info.py` `fetch`: after the wanted language (sitelink, tag langlink, geosearch 500 m), the native language is resolved once (`nativeLang`, Nominatim) and tried as sitelink, then as a geosearch in that Wikipedia when it differs from the wanted one; then English, then OpenTripMap.
- NEW **`info_translate`** (default `true`, bool): Setup → 🔎 Places → « Translate other languages with my AI ». `backend.ts` `placeLang` / `__init__.py` `place_lang` return `mayTranslate =`

`info_translate !== false` (never with `native`). With it on and no AI, the card's « No X article » note adds `translateNeedsAi`. Cache keys `info6:` (app), `info5:` (HA).

NEW Changes in 0.13.0

0.13.0 (2 November 2026) — own Overpass, SCAN 25, a data limit in money, Show / hide, contacts in the app, TomTom web, Linux (app only unless noted)

- NEW **Speed signs on the route** (the team's ×60 Fairmont runs, 2 November 2026: the A1's 120 stayed 4 km into Nyon, the A9's 100 at the hotel, then with Overpass down the whole trip at ≈50). `guideUrl` asks TomTom for `sectionType=speedLimit&sectionType=motorway`; `parseGuide` returns `limits` [[from, to, km/h]] and `motorway` [[from, to]] as indexes into the route's points; `guide_route` keeps them (`routeLimits`, debug `route_limits`). `road_at`: `roads.routeRoadAt` (within 40 m of the route) wins over an estimated, TomTom-reverse or missing road; a posted OSM `maxspeed` still wins. A motorway stretch with no TomTom limit gets the country's motorway limit (≈). No road after `LOST_KEEP_S`: `unknownRoad` = the country's urban limit (≈), never the last road. `defaultLimit` treats `*_link` as an ordinary road. The card asks `road_at` on the speed signs' own clock (`signNow`: simulated during ▶ Play, sent as `t`; `steadyRoad` runs on it) and while stopped too. TomTom reverse answers are kept within `TT_SAME_M` = 300 m only (no time hold). `preloadRoads`: one Overpass `way(around:30, route…)` query per route (a point every 100 m, ≤ 2000), used where a tile is still loading. ▶ Play `SPEEDS` gains ×1 and ×5.
- NEW **Own Overpass server.** `discovery.ts` `setOwnOverpass(url)` (from `keys.json` `overpass` at start and on `keys_save` `overpass_url`); `overpassPost` tries it before `OVERPASS_URLS`, with the call's timeout capped at `OWN_TIMEOUT_S` = 15 s. A failure rests it `MIRROR_REST_MS` (5 min) like a mirror, and the public mirrors answer. An empty answer from it also asks the public ones; if they fail too, the empty answer stands (`shortcut:` an empty spot inside its region costs one public call). New service `overpass_test` {lat, lon} → {count} (`out count` within 500 m). `keys_save` checks `overpass_url` against `^https?://[^/]+/.*interpreter$` ("" = public only). The team's server: CT 138 on the Geneva homelab, Alps extract (CH, AT, LI, north IT, east FR, south DE) from Geofabrik, hourly updates, LAN and tailnet only. See 6.1.
- NEW **SCAN 25.** `map_style` gains `ign_scan25`: IGN's 1:25,000 raster, `data.geopf.fr/private/wmts` layer `GEOGRAPHICALGRIDSYSTEMS.MAPS` (`PM`, JPEG, maxNativeZoom 16) with the person's own key (`keys.json` `ign`, `keys_save` `ign_key`, a HASH key from cartes.gouv.fr « Mes clés d'accès », after joining the « Communauté d'usage des données SCAN 25/100 sous licence »; professionals and associations only, as the sign-up asks for a SIRET; geoservices.ign.fr closed on 2026-03-26). `traffic_tiles` returns the tile URL as `scan25`; without a

key the style falls back to the IGN plan (`ign_fr`). Offered in France only, like `ign_fr`.

- **NEW** **Mapy.com.** `map_style` gains `mapy_outdoor` and `mapy_winter`: Mapy.com's REST map tiles `api.mapy.com/v1/maptiles/{outdoor|winter}/256/{z}/{x}/{y}?apikey=…` (maxNativeZoom 19) with the person's own key (`keys.json` `mapy`, `keys_save` `mapy_key`, from developer.mapy.com). `traffic_tiles` returns the URL as `mapy` with a `{set}` hole the card fills; without a key the style falls back to the standard map. Offered everywhere once the key is saved. Mapy.com's terms: the attribution « © Seznam.cz a.s. a další » links to api.mapy.com/copyright, and a clickable Mapy.com logo (`api.mapy.com/img/api/logo.svg` → mapy.com) sits over the map (a Leaflet control, bottom left) while its tiles show. `tileKey` now also drops `apikey=` (IGN and Mapy.com), so a key is never part of a cached tile's key and a new key keeps the cache.
- **NEW** **Data limit in money.** New setting `data_limit_money` (null or 0-10000, `data.ts` and `data.py`): `datacap.limitMb(s)` = `data_limit_money / data_price × 1024` MB when both are set, else `data_limit_mb`; picking an MB size saves `data_limit_money: null`. Setup shows 5 · 10 · 20 · 50 · 100 and « = <size> this month » only with a `data_price`, else `dataMoneyHint`; the status line adds the month's use in money (≈). See 6.5.
- **NEW** **Instruments Show / hide.** « ⚙ Show / hide » (`gaugepick`) next to the coordinates; while picking, every figure gets a tick box in its caption (`gaugesPicking`, « ✓ Done »). The hidden ones are kept in localStorage `tourist-explorer-gauges-off` (card, per device): speedometer, altimeter, compass, climb rate, altitude chart, speed chart, trip strip, car, accuracy (DOP), sky, signal (SNR).
- **NEW** **Contacts in the app.** Besides Google: the phone's address book (setting `contacts_phone`, Android) and Nextcloud (`keys.json` `nextcloud` {url, username, password}, an app password; `keys_save` `nextcloud`, null = forget; `contacts_nextcloud` now used by the app too). `contacts.ts` ports the HA CardDAV code (`fetchNextcloud`) with the same contact ids as the HA edition, so favourites match; its XML reader is a regex (`shortcut:` no namespace URI check, any prefix accepted). Kotlin command `phone_contacts` (`READ_CONTACTS`, `ContactsContract.StructuredPostal`, at most 5000 rows; `peopleFromPhone`). Service `contacts` merges google, nextcloud and phone, with an error per source; kept 10 min in memory.
- **NEW** **TomTom web pill.** `card/src/nav.ts` `tomtomWebUrl` (NavApp `tomtom_web`), a fourth pill after « In the app · Google Maps · ViaMichelin »: `plan.tomtom.com/en/route/plan?r=(costModel:FASTEST,routingProvider:GLOBAL,sorted:(h~V<lat>~J<lon>~Vaddr~J<name>,…),travelMode:CAR|MOTORCYCLE)`, from the position through the stops (up to 9 via points), names ASCII with « _ ». Car and motorbike only; the pill is hidden for other modes. The format was read off plan.tomtom.com on 2026-11-02 and can change without notice.

- **NEW** **Mappy pill.** `card/src/nav.ts` `mappyUrl` (NavApp `mappy`), after TomTom: `fr.mappy.com/itineraire#/voiture/<lat>,<lon>/<lat>,<lon>/car` from the position to one point (`goTo`: next stop or destination, like Waze); without a position `fr.mappy.com/plan#/<lat>,<lon>`. Read off fr.mappy.com on 2026-11-02: stops live in the page, not the link, and the mode in the link is ignored (Mappy opens on the last one used), so the pill shows in every mode. Mappy has no public API; its terms (art. 14) forbid robots: a link only.
- **NEW** **Linux AppImage.** `.github/workflows/linux-appimage.yml` (by hand or on an `app-v*` tag, Ubuntu 22.04 runner, artifact `NestourGPS-linux-x86_64`). See 2.6.
- **NEW** http scope: `http(s)://*:*/api/interpreter`, `https://data.geopf.fr/*`, `https://*:*/*remote.php/dav/*` (6.2). Settings: 71 keys. Services: 81. Version 0.13.0 (`versionCode` 13000).

Changes in 0.12.1 to 0.12.6

0.12.6 — the team's dial and two faults from the 0.12.5 simulator run

- Sign wheel: `WHEEL_SLOT` 50 → 30 px (34 px signs overlap), `.wheelview` padded `WHEEL_PAD` 12 px so the 54 px centre sign is not clipped, edges faded with `mask-image`, each slot's z-index `20 - distance`.
- `.row.top .sourcesel` min-width 120 px (the row wraps): 0.12.5's `--te-ui-zoom` at M squeezed it to « | ».
- `stopPlay` ends with a parked fix (speed 0, no heading): the speed box no longer shows the last simulated speed over the arrival's limit, and `updateMapInfo` drops the ended trip's ETA.

0.12.5 — the team's leftovers (audit 1-7) and the simulator road test

- Speech ducking: `tourist-explorer-speak` carries `duck` (`settings.beep_duck`) → `android.ts speak(text, lang, duck)` → Kotlin `SpeakArgs.duck`; `say()` takes the « may duck » audio focus only when it is true.
- Button size: `--te-ui-zoom` (`UI_ZOOM` S 1 · M 1.12 · L 1.25 · XL 1.4) on every `button`, `select` and `a.button` outside `.mapwrap`, plus Leaflet popups; the map controls keep `--te-zoom` (up to 1.8).
- Simulator places: `playOpts.pois` is 0 (off), 30/60/120/300 real seconds, or -1 (start, middle, end; `playStage`); `playOptions()` reads the old boolean (on = 30). `trip_start.sim` takes `every` (0-3600 s, 0 = only when asked) and `sim_position` `search: true` sets `sim.ask`; `simMaySearch` / `sim_may_search` (HA too) honour both, a search clears `ask`. The card's own search during play bypasses Setup's « search every N min » (`search(false, true)`): in real minutes it allowed one search per ×60 trip, so the map kept Geneva's places. `drivenM(tl, t)` (`Timeline.v`, `Leg.p` = pause before) gives the km counter, which ran on during pauses.

- Confirmations: `keyRemoveConfirm` (OpenTripMap, TomTom) and `debugClearConfirm`.
- By place › City: `regionOf` asks Nominatim at zoom 10 and caches `[cc, region, city]` under `place:` (was `region:`, zoom 5); `IP.stopPoints` and a trip's logged stops and towns carry `stop: true` and are filed under their city, the sampled track only under its region; a region with a city drops its city-less entry. Items whose `places` lack `city` are looked up again (card `fillItemPlaces` treats them as waiting). `touchTree` returns Country › Region › City. The HA card still has no « By place » for these lists.
- `car_reset` takes `departure` (the running trip's `trip.car`, error without a trip); `all` resets it too. `carStatus.departure` counts full tanks from `trip.car.since`.
- Parked heading: `rememberCourse` keeps the last course at ≥ 10 km/h (real sources only, saved to localStorage `tourist-explorer-course` every 10 s); `courseNow` sends it to `guide_route` when standing within 150 m of it.
- Errors: `asError` wraps Tauri's bare-string rejections (http, tile fetch) as `PluginError`, so every `.message` reads (« Routing unavailable: undefined » with the tablet's DNS down) and `fetchPlaces` sees a failed Overpass as failed; the dispatcher still prefixes those with the service name and hides keys; Google's token retry treats a `PluginError` as « no network yet ».

0.12.4 — map: ETA strip, right-hand buttons (app and HA card)

- `updateMapInfo` adds « ➡ next · 🏁 dest » arrival times from `tripEta` while a trip runs (called again when `trip_eta` answers).
- CSS: `.fullmap` and `.skipbox.skipmap .skipbtn` are 34 × 34 px with Leaflet's 2 px translucent border and 4 px corners (the left column's touch size), no shadow; `.skipbox.skipmap` top 52 px. The 📍 button's text is a `.lbl` span hidden on the map (title/aria-label keep « I'm here: … »). Version 0.12.4.

0.12.3 — the trip computer as the team listed it (app only)

- `car_set` takes `range` (0-3000 km): `Car.range = {km, odo, t}`; `typedRange` = km − km driven since, null once a fill-up is newer (then the tank formula). `carStatus.range_km` prefers it. `cleanCar` keeps it.
- Trip meter A always has `per100`: `mixPer100(a, consumption(fills, a.since, settings))` (estimated until two full tanks since the reset or a typed-in figure).
- Card: « Total km » (`carOdo`), the Trip A heading is « From reset (Trip A) », « Since departure » always shown (`carNoTrip` when no trip runs), the dialog has the autonomy field and greys « From start » without a trip.
- Tests: app 242 → 244. Version 0.12.3.

0.12.2 — ✎ From the car (app only)

- New service `car_set` {odo?, a?: {km?, per100?, avg_kmh?}, start?: {…}}: `fromCar(car, tripCar, d)` (`car.ts`) checks every field first (odometer 0-2,000,000 km, km 0-1,000,000, L/100 km 1-50, km/h 1-250; « start » without a running trip throws), then sets `car.odo.km` and shifts every fill-up's `odo` by the same amount (consumption between full tanks unchanged), sets a meter's km (scaling its times so the average speed holds), its times from the average speed (`drive_s = moving_s = km / kmh`), and `Meter.base = {km, per100}` from the consumption. `mixPer100` blends `base` with the app's own figure for the km since; `carStatus` gives trip meter A `per100`/`estimated` once it has a `base` and mixes the departure meter. `trip.car` is created on demand (`trips.ts` formula). Reset A clears `base`. `cleanCar` keeps `base`. Card: `fromCarDialog` (units shown, `toMetric`), button « ✎ From the car »; debug `car_set`.
- Strings `carFromCar`, `carFromCarHint`, `carFromReset`, `carFromStart`, `carAvgSpeed`. Tests: app 239 → 242. Version 0.12.2. The team had asked for it on 29 October (car stats, item 10); 0.10.0 had the meters but no way in.

0.12.1 — speed limits that hold, « I'm here », the sign wheel, route preview, IGN maps, Swiss departures (app only unless noted)

- **Speed limits that hold.** On the motorway at 100 the sign jumped to a 30 from a side road. Two guards in `roads.ts`: `roadAt(ways, pos, heading, country, keep)` counts the way the sign is on (`keep`) at half its distance, so a parallel way must be clearly nearer to win; and `steadyRoad(held, found, speedKmh, now)` holds a **suspicious drop** (new limit ≤ half the old one while the speed is above new limit + 20 km/h) for `DROP_CONFIRM_S` = 15 s: the candidate must be seen that long before it shows. A road lost for less than `LOST_KEEP_S` = 10 s keeps the last one. `road_at` (`backend.ts`, state `signRoad`) takes the card's `speed`; a held drop logs `road_held` (9). Only the sign; the road log (`trackRoad`) is unchanged. See 5.13.
- **« 📍 I'm here ».** New service `trip_here` {id, lat, lon}: `here(trip, id, now, pos)` (`trip.ts`) makes a pending waypoint visited (its planned entry becomes visited, or a log entry is added); `id: "dest"` logs the destination as visited (poi id `dest`), sets `dest_reached` and clears `dest`. `trip.still` is set to that entry. The card (`hereTarget`, `markHere`) shows the button in `skipBox` next to ⏭ within `HERE_M` (200 m walking or cycling, 300 m otherwise) of the next pending stop, else of `trip.dest`, and in that place's map window. Debug event `here`.
- **Sign wheel.** New setting `sign_view` ("fixed" | "3" | "4" default | "6") in `data.ts`, `data.py`. `renderSpeedButtons` builds `.wheelview > .wheelstrip > .slot`, every step (10, 5 in mph) up to the top value; `turnWheel` slides the strip so the current limit sits in slot `floor((slots-1)/2)`, fading and shrinking the others; no current limit = the sign nearest 50. Reduced motion: no transition. "fixed" = the old column.
- **Route preview in the app.** New service `route_preview` {points, mode?} → {path (thinned), km, drive_s, delay, traffic, litres, cost}: `routeLegs` from the device position (TomTom with a key, else OSRM); litres from the mean of

consumption_motorway and consumption_city, cost × energy_price. No tolls (shortcut: comment). The card's appChoice gets a first pill « 🏢 In the app » (previewRoute): the line (previewLine, #E69F00) is fitted on the map and a strip (ui.preview) shows the numbers, × removes both.

- **IGN maps.** map_style gains ign_fr (Géoplateforme WMTS PLANIGNV2, matrix PM, PNG, keyless, native zoom ≤ 19) and ign_es (www.ign.es/wmts/mapa-raster, layer MTN, GoogleMapsCompatible, JPEG, ≤ 17), both through IGN_WMTS in card.ts; offered in Setup only near France / Spain (inFrance, inSpain boxes). Credits: Etalab 2.0 / CC BY 4.0 (12). SCAN 25 (the French topo raster) needs a key: not used.
- **Swiss departures.** New engine module app/src/engine/transit.ts (stopsUrl, boardUrl, pickStops, parseBoard) and service departures {lat, lon} → {stops: [{name, icon, distance, departures: [{at, line, to, platform, delay, operator}]}]}: transport.opendata.ch (keyless), up to 3 stops within 1.5 km, the nearest train and ship stop first, 6 departures each, cached 60 s. The card's departuresButton (Info tab and map window, places in inSwitzerland) shows them in a dialog with « Timetable: opentransportdata.swiss ». Switzerland only: Transitous would cover the rest of Europe but forbids commercial use.
- New strings: imHere, imHereAt, hereToast, signsView, signsFixed, signsViewHint, inApp, departures, platform, noDepartures, noStopsNear (all 67 languages). Settings: 69 keys (sign_view).
- Tests: app vitest 234 → 239, card 161, pytest 120 passed, 1 skipped. Version 0.12.1.

0.12.0 — voice button, clock, button size, Share sheet, heading on routes (app only unless noted)

- **Voice button on the map.** renderVoiceBtn (card.ts) draws a button in the Leaflet left column under 🗻 showing 🔕 (off), 🔔 (beeps) or 🗣️ (voice); a tap cycles voice off → beeps → voice → off and saves it like Setup → Driving → Voice (same setting, 3.1). Called when settings change and when the setting is saved. Speech itself is unchanged (5.19, 7).
- **Clock.** New setting clock ("24" default | "12"), in data.ts and data.py. setClock / clock() in card/src/units.ts set the hour cycle (h23 / h12) used by every displayed time (ETA, timetable, logbook, GPS card, card notifications); card.ts calls setClock(next.clock) when settings load or change. App and HA card.
- **Button size.** New setting button_size ("S" default | "M" | "L" | "XL"). BUTTON_ZOOM in card.ts maps it to 1 / 1.25 / 1.5 / 1.8 and sets the CSS custom property --te-zoom on the card host; the CSS applies zoom: var(--te-zoom, 1) to .leaflet-left .leaflet-control, .fullmap and .skipbtn.skipmap (and the voice button). The speed-sign column grows with the left controls; place markers are not touched. Shown in Setup's simple view.
- **Share this trip.** shareData(tripId, pick, readable) (card.ts) now takes the readable files (the Markdown logbook) and passes them with the trip

JSON to `saveFiles(..., share = true)`, so the Share sheet opens at once with two files (before: a dialog with a ↗ Share button). On the desktop `app/src/main.ts` (`tourist-explorer-files` listener) has no share sheet: it saves both files under `Download/<export_folder>` and says where. Printing stays the PDF button (`print_html`, 7).

- **Export… all / none.** The past-trips « all / none » button in the Export dialog (`card.ts`) now sets every trip checkbox to one value; before, it flipped them one by one. Bug fix.
- **Route heading.** Service `guide_route` takes an optional `heading` (degrees, 0-359). The card sends it while moving at 10 km/h or more (`card.ts`, where it asks `guide_route`); the engine adds `vehicleHeading=<heading>` to the TomTom request (`tomtom.ts`) and `bearings=<heading>,45` to the OSRM one (`eta.ts`), so a reversed trip (⇄) or a reroute does not start with a U-turn. Traffic is unchanged (TomTom, `traffic=true`). The debug event `guide_route` now carries `heading` (9).
- Tests: app vitest 233 → 234, card unchanged, pytest 120 passed, 1 skipped. Version 0.12.0.

0.11.3 — routes, plans and past trips « By place » (app only)

- The Routes list, the Trip tab's saved plans and its past trips switch « 🚗 By mode · 🌍 By place » (`itemSections` in `card.ts`, one choice for the three lists: `localStorage tourist-explorer-items-by`). By mode is the old `modeSections`; by place is `touchTree` (`card/src/favgroups.ts`): Country › Region, an item under **every** country and region it touches, countries named by `countryName(cc)`, regions by `region_names`, else their ISO code; items not looked up yet fold under « … being placed ». See 3.2 Library.
- New service `item_places` {max} (app only; 4) fills `places` [{cc, region, lat, lon}] on routes, plans and ended trips step by step with the cached `regionOf`; new engine module `app/src/engine/places.ts` (`samplePath`, `tripPoints`, `stopPoints`); `Library.placeItems`, `TripManager.readTrack`.
- `library` routes and `trip_state` trips (summaries) carry `places` (null = not looked up yet). Version 0.11.3.

Changes in 0.10.2 → 0.11.3

0.10.2 (97c88e7) — reverse trips, free time in routes, tidier Setup

- ⇄ starts a route, plan or past trip the other way: `trip_start reverse: true`, `reverseStops` / `placeFree` in the new pure module `card/src/reverse.ts`, `tripOrigin` in `backend.ts`; « ⧉⇄ Save a reversed copy » card-side (`returnSuffix`). See ⇄ Reverse.
- Routes get ⏳ free-time stops: `route_save` takes `free` {freeId: note}; `library` lists `order` and `free` (`routes()`, `routeAsPlan` in `data.ts`). See 3.2 Library, 5.10.

- HA `route_save` accepts `pauses` and `free` (its schema refused the `pauses` the card sent); `data.py` checks them.
- The route editor's favourites and the ★ pickers follow the Favourites tab's « By kind · By place » switch (`favByPlace`, `favOptions`; `placeTree` moved to `card/src/favgroups.ts`). Setup group titles are coloured sticky bands with a line between topics (CSS). `tripAsPlan` keeps each place's visit time.

0.10.3 (52dc97e) — skip with undo

- New service `trip_unskip` (app and HA: `unskip` in `trip.ts` / `trip.py`); `trip_skip` takes `via`. See 4, 5.6.
- ⏭ Skip on the turn-by-turn banner or the map with an 8 s Undo toast; « ⏭ Skip » / « 🚫 Not interested » on the place card (`renderSkip`, `skipWaypoint`, `popupNode`). See 5.19.
- Debug log events `skip`, `unskip` (9).

0.10.4 (80a0a86) — steady popups, trips that stay, Simple Setup, colour-blind traffic

- The map holds still under a popup (`popupHold`, `following()`); a new trip never ends the running one without asking (`startTrip`, `askRunningTrip`, `startGuard` in `card/src/nav.ts`); Google reviews / TripAdvisor links (`reviewLinks`). See 5.20.
- New setting `setup_view` (simple | all; `SETUP_SIMPLE` in `card.ts`, `NEW_USER_SETTINGS` in `data.ts` / `data.py`): 66 keys then (68 since 0.12.0). See 3.1.
- Traffic recoloured in one pass by hue band with striped stopped traffic: `recolourTraffic` in `card/src/colours.ts`. See 6.1.
- The guide banner moves the map's top-left column below it and shows in the panel when the map is hidden or small (`showGuide`, `updateGuideMini`); `guide` debug note gains `tab`, `map`, « no next step ».

0.10.5 (c14b53e) — one preview line

- Routes and the plan editor: one « 🗺️ Preview route & costs » line (Google Maps and ViaMichelin side by side) replaces « Preview route » + « Route costs » (`appChoice` in `card.ts`, string `previewRoute`). Card only.

0.11.0 (b40af03) — experimental 3D map

- « ⛰️ » opens a MapLibre GL **6.14.0** view (new dependency, its own lazy chunk): `app/src/map3d.ts`, `app/src/map3d-style.ts` (protocol `nestour-tile://`, `style3d`, `zoom3d`), types in `card/src/map3d-api.ts`, glue in `card.ts` (`setMap3d`, `scene3d`, `update3d`). Terrain from Mapzen tiles on AWS, every tile through `tile_get`. See 5.21, 6.1, 12.

0.11.1 (2c0db81) — one country in every language

- A favourite's `place` carries `cc` (ISO code) and `lang`; « By place » groups by `cc` and names the country in the app language (`placeTree(favs, nowhere, countryName)`); `favorite_places` looks up again places named in another language. See 3.2, 4.

0.11.2 (4244420) — place types translated, 3D stays where you looked

- `kindNames` (72 type codes) in every language, `kindName` + `KIND_ALIAS` in `i18n.ts`. See 8.
- 3D opens at the 2D map's centre and zoom and follows only if 2D was following (`start3d`, `Open3dOptions.center` / `follow`); back in 2D the map is where 3D was (`View3d.view()`, `zoom2d`). See 5.21.

---

1. Overview and architecture

1.1 What runs where

The app has no server. Everything runs on the device:

- **The card** (`card/src/card.ts`, TypeScript + Leaflet) is the whole UI. It is the same custom element the HA integration serves (`tourist-explorer-card`). The app creates it with `standalone: true` (`card.setConfig` in `app/src/main.ts`) and hands it a fake `hass` object (`hass()` in `app/src/main.ts`) whose `callWS({type: "call_service"})` goes to the in-process dispatcher instead of Home Assistant.
- **The backend dispatcher** (`app/src/backend.ts`) implements the integration's services with the same names, inputs and outputs (`services` table and entry point `call()` in `backend.ts`): 81 services at 0.13.0 (41 of them app-only at 0.11.2).
- **The engine** (`app/src/engine/*.ts`) is a line-by-line TypeScript port of the Python modules in `custom_components/tourist_explorer/` (discovery, info, trip, gpslog, eta, motorway, geonames, facts, data, contacts) plus app-only modules (ai, tomtom, roads, schedule, weather, datacap, google).
- **The trip manager** (`app/src/trips.ts`) is the port of `tripsvc.py`: it follows this device's position and runs the trip engine on every fix.
- **The Tauri shell** (`app/src-tauri/`): Rust (`src/lib.rs`: plugin registration, a loopback OAuth listener, a desktop file writer), the official plugins (http, store, notification, opener, dialog) and one own plugin, **trip-location**, whose Android half is Kotlin (`plugins/trip-location/android/src/main/java/`): GPS, satellites, activity recognition, the foreground service, exports, PDF, beep, network and data counters.

1.2 Diagram

flowchart LR

```
 subgraph Webview["Webview (JS)"]
   Card["card/src/card.ts<br/>UI, map, guide, logbook"]
   Main["app/src/main.ts<br/>fake hass, DOM events"]
   Backend["app/src/backend.ts<br/>service dispatcher"]
   Trips["app/src/trips.ts<br/>TripManager"]
   Engine["app/src/engine/*.ts<br/>trip, roads, discovery, info,<br/>schedule, tomtom, ai, datacap"]
   Device["app/src/device.ts<br/>device_tracker.this_device"]
 end
 subgraph Rust["Tauri core (Rust)"]
   Http["plugin-http<br/>(allow-list)"]
   Store["plugin-store<br/>tourist.json, keys.json…"]
   Notif["plugin-notification"]
   Lib["lib.rs: oauth_listen,<br/>oauth_result, write_text_file"]
   TL["trip-location plugin"]
 end
 subgraph Android["Android (Kotlin)"]
   Plugin["TripLocationPlugin.kt<br/>LocationManager, GnssStatus,<br/>ActivityRecognition,
MediaStore, PDF, beep"]
   Service["TripLocationService.kt<br/>foreground service + wake lock"]
 end
 Card -- "hass.callWS(call_service)" --> Main --> Backend
 Backend --> Engine
 Backend --> Trips --> Engine
 Backend -- fetch --> Http --> Internet((Public APIs))
 Backend --> Store
 Trips --> Notif
 Plugin -- "position / gnss / activity events" --> TL --> Device --> Trips
 Device --> Main --> Card
 Backend -- "invoke plugin:trip-location|…" --> TL --> Plugin --> Service
 Card -- "DOM events: files, print, beep, gps" --> Main --> TL
```

In words: Kotlin pushes `position`, `gnss` and `activity` events through the plugin to `android.ts`, which feeds `device.ts`; `device.ts` notifies two listeners: `main.ts` (rebuilds `card.hass` so the card redraws) and the `TripManager` (runs the trip). The card calls services through `hass.callWS`; `backend.ts` answers from the engine, the stores and the network. Exports, PDF, beep and the GPS switch go from the card to `main.ts` as DOM `CustomEvent`s (`tourist-explorer-files`, `-print`, `-beep`, `-gps`), which call the plugin.

The Home Assistant path is the same card served by the integration: `hass.callWS` reaches real HA services (`SCHEMAS` and the `hass.services.async_register` loop in `custom_components/tourist_explorer/__init__.py`), positions come from HA entities (Companion app tracker), the trip runs in `tripsvc.py` on tracker state changes, notifications go through `notify.mobile_app_*`, tiles through HA's `map_tiles` proxy. See section 10.

1.3 Repository layout

Path	Content
`app/`	The standalone app. `package.json` (Vite 8.3.5, Vitest 5.0.5, TypeScript 7.0.2, Tauri JS packages; `maplibre-gl` 6.14.0,

Path	Content
	pinned), `vite.config.ts` (injects `__APP_VERSION__` from `tauri.conf.json`, serves `../card`), `index.html`
`app/src/main.ts`	Boot: language, fake `hass`, card config, DOM event bridges, GPS on/off
`app/src/map3d.ts`, `map3d-style.ts`	The experimental 3D map (MapLibre, loaded on first use; 5.21)
`app/src/backend.ts`	The service dispatcher, stores, keys, Nominatim queue, road tiles, data cap, trip services
`app/src/trips.ts`	`TripManager`: per-trip lock, `onPosition`, background work, search, notifications
`app/src/android.ts`	Typed wrappers for the trip-location plugin commands
`app/src/device.ts`	This device as `device_tracker.this_device` + `person.me`
`app/src/debuglog.ts`	The debug log (memory + `debug.json`)
`app/src/beta.ts`	`BETA_EXPIRES`, `VERSION`, services allowed after expiry
`app/src/langs.ts`	Lazy loading of the 60 extra languages
`app/src/notify.ts`	Trip notifications with Go / Not now buttons
`app/src/icons.ts`	Stand-in `<ha-icon>` with the dozen MDI paths the card uses
`app/src/engine/`	Engine modules (section 5, 6)
`app/test/`	Vitest tests of backend, engine and the 3D style (24 files)
`app/src-tauri/`	Rust crate `app_lib`, `tauri.conf.json`, `capabilities/default.json`, `icons/`
`app/src-tauri/plugins/trip-location/`	Own plugin: `build.rs` (command list), `permissions/`, `src/lib.rs`, `android/` (Kotlin)
`app/src-tauri/gen/android/`	Generated Android project (tracked: `app/build.gradle.kts` with the signing config, manifest, launcher icons)

Path	Content
app/icon-source/	Icon sources and icon.json for tauri icon
app/dist-apk/	Built APKs, NestourGPS-<version>-arm64.apk
card/	The shared card (Leaflet 1.9.4): src/ (card, logbook, guide, nav, units, i18n, lang/, sources, gps, favgroups, colours; reverse.ts ⇄, map3d-api.ts 3D types…), test/ (18 files)
custom_components/tourist_explorer/	The HA integration (Python) and its www/ card bundle and docs/ PDFs
tests/	Python tests of the integration (pytest) and fixtures (geneve_champex.json, walk_tour_du_bloc.json)
deploy/	HA deploy: deploy.sh, deploy-dev.sh, install.py, boxes/*.json, publish-docs.sh
docs/	Docs; docs/app/ for the app's own set

---

2. Build, sign and release

2.1 Windows development

```
cd app
npm install
npm run tauri dev        # Vite on http://localhost:1420, a desktop window "Nestour GPS" 1200×800
```

The Rust toolchain must be on PATH (~/.cargo/bin). On the desktop, positions come from the webview's navigator.geolocation (gps in main.ts), exports are written by the Rust command write_text_file into Download/<export_folder> (the tourist-explorer-files listener in main.ts; write_text_file in src-tauri/src/lib.rs), PDF goes through the webview's own print dialog, and there is no foreground service, no satellites and no activity recognition. Vite must be restarted if files are added to card/src/lang/extra/ (the glob in langs.ts is evaluated at start).

2.2 Android toolchain

Variable	Value on the build laptop
ANDROID_HOME	C:\Users\erikt\AppData\Local\Android\Sdk
NDK_HOME	C:\Users\erikt\AppData\Local\Android\Sdk\ndk\29.0.14206865
JAVA_HOME	C:\Program Files\Android\Android Studio\jbr
adb	%ANDROID_HOME%\platform-tools\adb.exe (not on PATH)

Gradle project: compileSdk 37, targetSdk 37, minSdk 24 (gen/android/app/build.gradle.kts, android { … }); the plugin library compiles against SDK 36 with Java/Kotlin target 1.8 and depends on play-services-location 21.3.0 (activity recognition only).

2.3 Release build and signing

```
cd app
npx tauri android build --apk --target aarch64
```

beforeBuildCommand runs npm run build (tsc && vite build) first. The signed APK lands under app/src-tauri/gen/android/app/build/outputs/apk/ (in the current tree: universal/release/app-universal-release.apk); copy it to app/dist-apk/NestourGPS-<version>-arm64.apk.

Signing key. The release signing configuration reads a properties file outside the repository (signingConfigs in build.gradle.kts): ~/.android-keys/tourist-explorer.properties (keys storeFile, password, keyAlias), or the file named by ANDROID_KEY_PROPERTIES (CI). The keystore lives in ~/.android-keys/ too. Both exist on the laptop; they are never committed and their contents never go into chat, logs or documents. Without the file, Gradle builds an unsigned release. Losing the key means every device must uninstall (and lose its data) to update: keep a backup.

Versioning. The one version that counts is version in app/src-tauri/tauri.conf.json (now 0.13.0). It feeds: the card's About box (vite.config.ts → __APP_VERSION__ → VERSION in beta.ts), Android versionName and versionCode (Tauri writes gen/android/app/tauri.properties: 0.13.0 → 13000) and the User-Agent (backend.ts, NestourGPS/<version>). app/package.json and the app crate (src-tauri/Cargo.toml) are kept equal to it by hand; the trip-location plugin crate stays 0.1.0. Each beta build carries its own end date: BETA_EXPIRES in app/src/beta.ts.

Identity. Product name and window title Nestour GPS; application id com.tjepkema.touristexplorer (kept from the old name, changing it would make a new app on every device).

Icons. Sources in `app/icon-source/` (`icon.json`: `app-icon.png`, adaptive `android-bg.png` / `android-fg.png`, background `#0072B2`). Regenerate every size with:

```
cd app
npx tauri icon icon-source/icon.json      # i.e. app/icon-source/icon.json from the repo root
```

The Tauri CLI is installed in `app/node_modules`, not at the repository root, so run it from `app/`.

NEW 2.6 Linux AppImage

Tauri cannot cross-compile Linux from Windows, so GitHub Actions builds it: `.github/workflows/linux-appimage.yml`, run by hand (Actions → Linux AppImage → Run) or on an `app-v*` tag. Ubuntu 22.04 runner (its older glibc runs on newer distributions), `libwebkit2gtk-4.1-dev` and friends, `npm ci` in `card` and `app`, then `npx tauri build --bundles appimage`; the run's artifact is `NestourGPS-linux-x86_64`. On the target: `chmod +x`, run; Arch and Omarchy need `webkit2gtk-4.1` (`sudo pacman -S webkit2gtk-4.1`). Position comes from the webview's geolocation; there is no USB GPS feed (11.1). Not yet tried on the team's Omarchy laptop.

2.4 Installing over adb (including Tailscale)

```
adb install -r app/dist-apk/NestourGPS-<version>-arm64.apk
```

- **Tablet (Lenovo Tab M9):** Play Protect blocks sideloading, so install over wireless adb.
- **Over Tailscale, from anywhere:** after each tablet restart, Wireless debugging on the tablet shows a pairing code and port. `adb pair <tablet-tailnet-ip>:<pair-port> <code>`, then `adb connect <tablet-tailnet-ip>:<tls-port>`, then `adb tcpip 5555` and `adb connect <tablet-tailnet-ip>:5555` (fixed port until the next reboot).
- **At home on the LAN:** Tailscale on the laptop routes 192.168.1.0/24 into the tunnel and wins over the local route; turn Tailscale off or use the tailnet address.
- **White screen after `adb install -r`** (Lenovo/MediaTek): the system may refuse the app's WebView renderer (« process is bad » / « suppress to start service » in logcat). Force stop from Settings → Apps clears it; `adb force-stop` and a reboot do not reliably (7).
- In Git Bash, `adb pull /sdcard/...` needs `MSYS_NO_PATHCONV=1`.
- Pull the debug export from the tablet with `adb pull "/sdcard/Download/<export_folder>/<file>.json"`.
- Debugging a release webview: temporarily call `WebView.setWebContentsDebuggingEnabled(true)` in the plugin, forward `localabstract:webview_devtools_remote_<pid>` and attach DevTools. Never commit it.

2.5 Test gates

Run all of these before a release build (app and card tests at 0.12.3, run on 2 November 2026; the bundle size is that of 0.11.2):

Gate	Command	Result
App unit tests	`cd app && npx vitest run`	**244 tests**
App type check	`cd app && npx tsc --noEmit` (also part of `npm run build`)	clean
Card unit tests	`cd card && npx vitest run`	**161 tests**
Card type check + bundle	`cd card && npm run build` (`tsc --noEmit && vite build`)	builds `custom_components/tourist_explorer/www/tourist-explorer-card.js` (809,829 bytes at 0.11.2)
Integration tests	`python -m pytest -q tests`	**120 passed, 1 skipped** (the drive simulations need HA installed)
Rust	`cd app/src-tauri && cargo test`	1 unit test (`redirect_query`), slow: compiles Tauri

Vitest does not type-check: a string missing from a language file (section 8) or a wrong type only fails in `tsc`. Node tests cannot catch a host missing from the http allow-list (section 6.2): try every new API once in the built app.

---

3. Data model

3.1 Settings

One settings object per user (the app has one user, `me`, `USER_ID` in `device.ts`), stored in the library. Defaults: `DEFAULT_SETTINGS` (`app/src/engine/data.ts`); every write is checked key by key against `SETTING_CHECKS` (`data.ts`): an unknown key or a bad value throws (`Library.saveSettings`). The HA integration has the same 71 keys (`data.py`; `setup_view` was the 66th, 0.10.4; `clock` and `button_size`, 0.12.0; `sign_view`, 0.12.1; **NEW** `contacts_phone` and `data_limit_money`, 0.13.0). "App" in the last column marks keys only the standalone app uses.

Key	Default	Allowed values	Use
`min_stars`	2	integer 0-3	Minimum stars on the map and for trip proposals
`radius`	5000	1000, 2000, 5000, 10000, 20000	Search radius (m)
`nav_app`	`google`	google, viamichelin, viamichelin_app, tomtom_go, waze, in_app, geo	Go: which app
`travel_mode`	`walking`	walking, driving, bicycling, transit, motorbike, auto	Default mode; `auto` (0.9.5, offered in the app only) = the detected one, see below
`categories`	all 8	subset of sight, museum, viewpoint, historic, religion, art, nature, other	Categories shown/proposed
`ai_entity`	null	null or `ai_task.*`	HA only (AI entity)
`sim_interval`	60	integer 5-1800 s	Random-walk simulator
`sim_move`	true	bool	Simulator moves
`country`	""	"" or ISO-2 upper case	Place search country, city list
`max_pois`	25	10, 25, 50, 100, 150	Places on the map
`trip_radius`	20000	10000, 20000, 30000	Trip look-ahead radius (applied to a running trip at once)

Key	Default	Allowed values	Use
trip_alerts	true	bool	Notifications for places ahead
fuel	GASOLINE	GASOLINE, DIESEL	ViaMichelin car, fuel price
energy_price	null	null or 0.3-6	Price per litre
caravan	false	bool	ViaMichelin
vm_account	false	bool	Use the ViaMichelin account's vehicle
info_lang	""	"", native, a code of info.LANGUAGES (63)	Language of descriptions and facts
NEW info_translate	true	bool	A text in another language is translated by the AI (not with native)
route_mode	stretch	stretch, eta	Long routes: 8 waypoints + 9th, or 7 + destination
tracker	""	"" or device_tracker.*/sensor.*/person.*	HA only
colours	colourblind	standard, colourblind, contrast	Palette; colourblind recolours TomTom flow (green, orange, red, stopped striped; 6.1)
backup_tracker	""	as tracker	HA only
contacts_nextcloud	false	bool	Nextcloud contacts; **NEW** the app too since 0.13.0 (turned on by saving the Nextcloud address book)
contacts_google	false	bool	Google contacts as destinations
contacts_phone **NEW**	false	bool	App, Android: the phone's

Key	Default	Allowed values	Use
			address book (postal addresses) as destinations (0.13.0)
`save_every`	30	15, 30, 60, 300 s	App: trip written to disk at least this often
`units`	`metric`	"", metric, us, uk	Display units ("" = from locale)
`clock`	`24`	"24", "12"	Times shown, 24 h or 12 h (am/pm); `setClock` in `card/src/units.ts`. App and HA card; checked in `data.ts` and `data.py`
`button_size`	`S`	S, M, L, XL	App: size of the map buttons and speed signs (×1 / 1.25 / 1.5 / 1.8), CSS `--te-zoom`; checked in `data.ts` and `data.py`
`ui_lang`	""	"" or a 2-3 letter code	App: UI language ("" = device)
`guide`	true	bool	App: turn by turn on the map
`exit_warn_m`	1000	0, 500, 1000, 2000, 3000	App: early exit/fork warning
NEW `guide_lanes`	true	bool	App: lanes and the exit sign in the banner
NEW `auto_zoom`	"smart"	`off`, `speed`, `smart`	The map's zoom while it follows
NEW `zoom_radius`, `zoom_junctions`, `zoom_places`	true	bool	Smart auto-zoom's three rules

Key	Default	Allowed values	Use
speed_buttons	true	bool	App: tap-to-set limit buttons
pictures	noroam	on, noroam, off	App: place photo in Infos
data_limit_mb	0	integer 0-100000	App: monthly data limit (0 = none)
data_limit_money **NEW**	null	null or 0-10000	App: the limit as money, with data_price; wins over data_limit_mb (datacap.limitMb()); an MB choice clears it (0.13.0)
data_scope	roaming	roaming, mobile	App: what the limit counts
go_target	next	next, dest	App: Waze/TomTom GO go to next stop or destination
traffic_layer	true	bool	App: TomTom flow overlay (with key)
pauses_on	true	bool	App: planned pauses count
pause_default_min	15	integer 1-60	App: a stop's pause without its own
drive_max_h	8	0, 4-10, 12	App: driving per day (0 = no limit)
drive_from	08:00	HH:MM	App: driving window start
drive_to	19:00	HH:MM	App: driving window end
break_every_h	2	0, 1, 1.5, 2, 2.5, 3, 4	App: break reminder (0 = off)
stop_minutes	5	integer 1-15	App: still this long = a stop/visit
gauge_size	full	full, half	App: dial size

Key	Default	Allowed values	Use
`beep_duck`	false	bool	App: lower the radio during the beep
`keep_screen`	`trip`	off, trip, always	App: screen stays on
`map_style`	`standard`	standard, satellite, terrain, tomtom, tomtom_night, swisstopo, ign_fr, ign_scan25 **NEW**, ign_es, mapy_outdoor **NEW**, mapy_winter **NEW**	App: base map; `ign_scan25` needs the IGN key, else it shows `ign_fr`
`route_lines`	true	bool	App: line between waypoints
`auto_search`	true	bool	App: map and trip search by themselves
`approach_km`	2	0, 2, 5	App: approach popup distance
`search_every`	walking 1, bicycling 2, driving 5, motorbike 5, transit 5	per mode: 1, 2, 5, 10, 15, 30 min	App: map's own search cadence
`hiking_trails`	false	bool	App: swisstopo hiking overlay
`split_view`	`auto`	auto, always, never	App: panel beside map
`export_folder`	`TouristExplorer`	`^[\w][\w .-]{0,39}$`, no `..`	App: `Download/<folder>`
`speed_signs`	true	bool	App: limit and speed on the map
`sign_view`	`4`	fixed, 3, 4, 6	App: the country's fixed sign column, or a turning wheel of 3, 4 or 6 signs (0.12.1); checked in `data.ts` and `data.py`
`speed_beep`	false	bool	App: beep when over limit + margin (also used by the

Key	Default	Allowed values	Use
			approach popup)
`speed_margin`	5	integer 0-20 km/h	App
`speed_beep_every`	0	0, 10, 20, 30, 60 s	App: repeat while too fast (0 = once)
`debug_log`	true	bool	App: record the debug log
`consumption_motorway`	null	null or 1-40 L/100 km	App: fuel cost
`consumption_city`	null	null or 1-40	App: fuel cost
`data_price`	null	null or 0-50 per GB	App: data cost
`voice`	`beeps`	off, beeps, voice	App: guidance out loud (5.19), also the 🔇 / 🔔 / 🗣️ button on the map (0.12.0, `renderVoiceBtn`): off = no guide/approach beeps (the over-speed beep still follows `speed_beep`), beeps, voice = spoken turns, places, limit
`vehicle`	`none`	none, van, motorbike, classic, auto	App: the « you are here » marker (5.19); `auto` by mode: driving → van, motorbike → motorbike, else the blue dot
`tank_l`	null	null or 5-200 L	App: fuel tank, for the range (Instruments → Car)
`map_rotate`	`north`	north, heading	App: map north up, or heading up while following above 5 km/h (the

Key	Default	Allowed values	Use
			map's ▲ button)
`tile_cache_mb`	500	100, 500, 2000	App: kept map tiles, oldest out past this
`setup_view`	`all` (a new user: `simple`, `NEW_USER_SETTINGS`)	simple, all	Setup shows the essentials (`SETUP_SIMPLE`, card) or everything (0.10.4, 5.20)

Travel mode `auto` (0.9.5). `MODES` (`data.ts`) accepts `auto` for `travel_mode` and for a plan's or route's own `mode` (HA too: `MODE_SETTINGS` in `data.py`); the card offers it only standalone. The mode in force comes from `effectiveMode(set, legs, lastDetected)` (`card/src/nav.ts`): a mode picked by hand wins; `auto` follows the running trip's last leg (bus, train, boat → transit), else the last leg any trip detected (`lastDetected`, `backend.ts`), else driving. The engine's side is `T.travelMode(trip)` (`engine/trip.ts`), used by `mapsUrl` and `trip_eta`; `plan_schedule` routes with `effectiveMode` too. A trip stores the setting (`mode: "auto"`), not the resolved mode.

Two comments in `DEFAULT_SETTINGS` (`data.ts`) are shifted by one line: the comment on `approach_km` also carries `auto_search`'s, and `search_every`'s carries `route_lines'`.

The webview's `localStorage` holds a few per-device UI keys outside the library: `tourist-explorer-lang` (read before the library loads, `language` in `main.ts`), `-gps-off`, `-source`, `-collapsed`, `-map-off`, `-setup-open`, the folds' `-fav-open`, `-route-open`, `-trip-open`, `-plan-open`, `-nextup-open`, `-fav-by` (Favourites « By kind » / « By place », 0.10.2) `-items-by` (Routes, plans and past trips « By mode » / « By place », 0.11.3) and **NEW** `-gauges-off` (the Instruments hidden by « Show / hide », 0.13.0); `sessionStorage` holds `-3d` and `-3d-hint` (5.21) (`card.ts`, the `*_KEY` constants).

3.2 Library

`tourist.json` → key `library` = `{ users: { me: User } }` (`Library`, `data.ts`). Shapes are the HA store's, so a library can move between the two.

```
User = { favorites: {id: Favourite}, routes: {id: Route}, plans: {id: Plan}, trips: {id: Trip},
         settings: {…}, active_trip: id | null, car: Car }
Car = { odo: Meter, a: Meter, fuel: {since}, fills: [{t, litres, full, price, odo}] }   (engine/car.ts, 0.10)
Meter = { km, drive_s, moving_s, since }
```

Car (the trip computer, Instruments → Car): counted by `TripManager.countCar` on the driving legs' own metres (walks, bus, train left out; test trips only on their own

`trip.car`). A new record starts from the km the device's trips already drove. Consumption = the litres put in after a full tank up to the next full one ÷ the km between them (≥ 20 km); without two full tanks since the reset: the last real figure, else Setup's consumption (≈). Range = fuel left (the last full tank − km since × consumption + partial fills after it) ÷ consumption. Goes out with « your data » (not with one trip) and comes in only to a device without a car record.

Item	Fields	Rules
Favourite	`LANDMARK_FIELDS` (`data.ts`): id, name, lat, lon, kind, category, stars, wikidata, wikipedia, website, otm_xid, carriageways, exit, note, w3w, group, place	id/name/lat/lon required; stars 0-3; note ≤ 2000; website http(s); w3w three dotted words; group ∈ people, junctions, culture, food, places; `place` = {country, region, city, cc, lang} filled by `favorite_places` (0.11.1: cc = ISO country code: « By place » groups by it and names the country in the app language, so « Suisse » and « Switzerland » are one; lang = the language the names were looked up in; a place whose `lang` is not the app language is looked up again, a few at a time)
Route	id, name, stops (favourite ids and ⏳ free-time ids, order), dest, description, mode, pauses {stopId: minutes}, free {freeId: note}, places	stops must be favourites or `free:<1-24 of [0-9a-z]>` ids with an entry in `free` (note ≤ 200), ≤ 100 stops, at least one favourite; `pauses` and `free` keep only ids in the stops (pause 0-600); deleting a favourite removes it from routes. Same checks in `data.py` (0.10.2)
Plan	id, name, description, dest, stops [{id, name, lat, lon, stars?, pause_min (0-600)?, free?, carriageways?}], favorite, mode, start {kind leave/arrive, at}, saved, places	≤ 50 stops, name ≤ 120, description ≤ 2000; `free: true` = a ⏳ free time, not a place

Item	Fields	Rules
Trip	see below	one active trip (`active_trip`); a 🧪 test trip refuses to start while a real one runs

Route free times (0.10.2, app only in the card). A route's ⏳ free time is an id `free:…` in `stops`, its note in `free`, its length in `pauses` like a place's visit time. `routes()` (the `library` answer) keeps `stops` as the favourites only (maps, links, simulator: no place to drive to) and adds `order` (every stop id in order, free times included) and `free`; the route editor works on `order`. `routeAsPlan` turns each free time into a plan stop `{id, name: note, free: true, pause_min?}` placed at the place before it (`placeFree`, `card/src/reverse.ts`), so a trip started from a route gets it in `free_plan` as a plan's would. HA's `data.py` checks and stores the same fields and lists `order`/`free`, but its `route_as_plan` skips free times (HA trips have no free time).

By kind · By place everywhere (0.10.2, app only). The Favourites tab's switch (`favByPlace`, `localStorage tourist-explorer-fav-by`) also shapes the route editor's favourite list and the ★ pickers (`favOptions`: one `<optgroup>` per country › region › city, the ones not placed yet last). The tree is `placeTree(favs, nowhere, countryName)` in `card/src/favgroups.ts` (tested in `card/test/favgroups.test.ts`); since 0.11.1 it groups by `place.cc` and names the country with `countryName(cc)` (`Intl.DisplayNames` in the app language), so a place looked up in French and one in English land in one country.

Routes, plans and past trips by place (0.11.3, app only). `places` on a stored route, plan or ended trip = [{cc, region, lat, lon}]: each country (ISO code) and region (ISO 3166-2 code, `ISO3166-2-lvl4`, or null) it touches, with one point in it (2 decimals, for the region's name). Language-neutral, so a change of app language never looks them up again (only their names). Written only by `item_places`; nothing from the card or an import is taken: `saveRoute`, `savePlan` and `importTrip` build the item field by field, so a save drops `places` and the item is looked up again; `removeFavorite` drops it on the routes it changes and `endTrip` on the trip. The points (`app/src/engine/places.ts`): a route's favourites and destination (`stopPoints`), a plan's stops (⏳ free times left out) and destination, a trip's logbook entries visited / stop / parked / night / passed / free, its `towns`, and its GPS track sampled by `samplePath`: the start, a point every 15 km along it (`SAMPLE_M`), the end (`tripPoints`; the track read by `TripManager.readTrack`, not kept in memory). The running trip is never looked up; it waits under « … » until it ends. HA has no `item_places` (its card shows « By mode » only; the switch is app-only, like Favourites').

No city level: a town needs a zoom-10 look-up per point (the region one is zoom 5, cached per ~1 km and shared with the speed signs' country), a trip touches dozens of towns, and the trip's town list is already in its logbook.

Trip. Created by `T.newTrip` (`trip.ts`) and grown by the engine. Persistent fields:

Field	Meaning
id, name, user_id, tracker, dest, radius, started, ended, mode	Identity; `mode` from the plan, else Setup (may be `auto`, 3.1); `dest` can be set later by `trip_dest`
vehicle	0.9.5: `van`, `motorbike`, `classic` or null, fixed at `trip_start` from `vehicleFor(settings.vehicle, effectiveMode(mode, null, lastDetected))`; the logbook heading shows its marker
sim {pois, now, searched} / record_only / route_id	Test trip on a simulated clock / no notifications / started from a route
planned, planned_at	**List** of POI ids in waypoint order; `planned_at` {poiId: arrival, dest} = the plan's timetable, written in the background by `tripStart` (read by `trip_eta` for « late »)
log	Logbook entries (below)
legs	[{mode, start, end, from, to, distance, manual?}]
roads	Road log (≤ 1000): {key, ref, name, highway, tunnel, toll, limit, estimated, from, to, distance, max_kmh, town}
towns	[{t, name, lat, lon}] (≤ 1000)
places	Ended trips only: the countries and regions it went through, [{cc, region, lat, lon}], filled by `item_places` (3.2, 0.11.3); dropped by `endTrip`
traffic	TomTom incidents met and ↻ deviations (≤ 200): {t, id, kind, text, road, from, to, delay, length, lat, lon}
weather	Readings (≤ 200): {t, temp, code, wind, rain, lat, lon}
next / seen / nearby	Proposed places / ids already proposed or dismissed / rated places for arrival detection (≤ 200)
pauses {poiId: s}, free_plan [{id, name, pause_min}], start_rule, free {since, entry}, night {since, entry}, days [t]	Timetable and multi-day state
data_start, data_bytes	App traffic counter at start; bytes used by the trip
car	The trip's own trip-computer meter {km, drive_s, moving_s, since} (« since departure »)
plan_path	The planned route's line at the start ([[lat, lon]], a point every ≥ 80 m): the logbook draws it dashed under the track and lists the deviations

Field	Meaning
	(`card/src/deviation.ts`: > 300 m off for > 1 km). Not for test trips
imported	Set by `importTrip`: driven by another device, left out of this device's odometer
stretch_end, prev, fix_t, heading, last, anchor, still, closest, main_road_at, wheel_since, break_told, leg_last, leg_probe, road_probe, town_m, town_last, traffic_at, weather_at	Engine working state

`slim()` (`backend.ts`) removes `nearby, seen, still, anchor, last, leg_probe, leg_last, road_probe` before a trip goes to the card.

Log entry kinds (`addLog`, `trip.ts`; base fields id, kind, arrived, left, lat, lon, poi, name, rating, note):

Kind	Icon	Written when
`planned`	🧭	A place added as a waypoint (`planStop`)
`visited`	📍	Still near a known place for min(5 min, stop time); replaces its passed/skipped/seen lines
`stop`	⏸	Still elsewhere for the stop time; named by `stopName` (POI within 120 m, else street and town)
`passed`	⤳	A pending waypoint driven past within the pass radius
`skipped`	⤼	By hand (`trip_skip`), behind a later reached waypoint (`skipBehind`), or `missed: true` (`markMissed`)
`seen`	👀	A proposed (not planned) sight driven past
`parked`	🅿️	A driving leg turned into walking
`gap`	⚡	The app was not running (resumed after an interruption)
`night`	🛏	🌙 End of day, named by the town

Kind	Icon	Written when
`free`	⏳	⏳ Free time; `name` = the comment, `note` = distance walked (≥ 50 m), `pause` = planned length

Extra entry fields: `weather`, `detour_km` (visits, computed after the trip), `pause_s`, `pause_told`, `pause_skipped`, `missed`.

3.3 Storage files

All through `tauri-plugin-store`, in the app's data directory (Windows: `%APPDATA%\com.tjepkema.touristexplorer\`; Android: the app's private storage). Every store is opened with `autoSave: false` and saved explicitly: the plugin's auto-save is a debounce restarted by every change, so with one GPS fix a second it never wrote during a trip and a power cut lost the whole trip (2026-10-27, the store comment in `backend.ts`).

File	Keys	Written
`tourist.json`	`library`, `cache`	Library after every change; during a trip at most every `save_every` unless something important happened (`tripSave`, `backend.ts`); cache when it changes
`keys.json`	`otm`, `tomtom`, `ign` **NEW**, `overpass` **NEW**, `nextcloud` **NEW**, `ai`, `ais`, `google`, `pending`	On key changes and during Google sign-in
`track_<tripId>.json`	`points`	On incoming points, at least every `save_every` (counted on points, not a timer: Android throttles timers in the background); at once on flush; emptied (file kept) on delete
`data.json`	`month`	Every 30 s on Android (`dataTick`)
IndexedDB `nestour-tiles` (webview)	`blob` (tile by URL without its key), `meta` {key, size, t}	Every base-map tile downloaded (`engine/tilecache.ts`); refreshed after 30 days when online; oldest out past `tile_cache_mb` (down to 90 %). TomTom traffic tiles never

File	Keys	Written
`debug.json`	`events`	Every 60 s and on clear
`cities.json`	`cities`	Once, after the GeoNames download

keys.json fields (`keysFile`, `backend.ts`). The card only ever learns whether a key is set (`keys_status`).

Field	Content
`otm`	OpenTripMap API key ("" = none)
`tomtom`	TomTom API key
`ign` **NEW**	IGN Géoplateforme key for SCAN 25 ("" = none)
`mapy` **NEW**	Mapy.com API key for the hiking and winter maps ("" = none)
`overpass` **NEW**	The person's own Overpass server, `http(s)://host/…interpreter` ("" = public mirrors only)
`nextcloud` **NEW**	{url (https), username, password (an app password)} or null
`ai`	The AI in use: {provider, model, key, url}
`ais`	Every provider's own setup by provider name (0.9.3: switching providers keeps each one's key); a pre-0.9.3 `ai` is copied in on read
`google`	{client_id, client_secret, refresh_token, email} of the person's own OAuth client (type Desktop)
`pending`	A Google sign-in in progress: {port, redirect, verifier, state, until, answer?}, so it survives a webview reload

`keys_import` takes another device's `keys.json` (same checks as typing the keys, plus the Google refresh token).

Cache (`Cache`, `data.ts`): `{key: {v, exp}}`, pruned at start. Keys and lifetimes:

Prefix	Content	TTL
`search3:<lat3>,<lon3>,<r>/…:fast`	Places (full / fast pass)	24 h; partial or fast 10 min
`otm:`	OpenTripMap places	24 h
`info5:<lang>:<t/n>:<id>`	Description, source, photo file	30 d
`facts:<lang>:<provider>:<model>:<id>`	AI facts	forever

Prefix	Content	TTL
`pic1:<file>`	Commons photo URL and credit	30 d
`region:<lat2>,<lon2>` / `country:<lat1>,<lon1>`	Country and region	90 d
`regname:<lang>:<ISO 3166-2>`	A region's name in that language (« » = none), for `item_places` (written by `regionOf` too)	90 d
`reverse:<lang>:<lat4>,<lon4>`	Street and town	90 d
`geocode:` / `find:`	Nominatim search	90 d
`fuel:`	Fuel stations	1 h
`simpath:`	Simulator route line	1 d

Road tiles (speed limits) are kept **in memory** only, the last 40 (`TILES_KEPT`, `backend.ts`). Contacts (Google, Nextcloud, the phone) are kept in memory 10 minutes, never on disk.

3.4 Import and export formats

Export	Made by	Format
Your data / one trip	`library_export` → card saves `tourist-explorer-data-YYYY-MM-DD.json` / `-trip-…`	`{format: "tourist-explorer", version: 1, favorites, routes, plans, trips, exported, tracks: {tripId: points}}`; `pick` selects trips, favourites, routes (with the favourites they use), plans
Logbook	card `logbook.ts`	Markdown (`text/markdown`), GPX (`application/gpx+xml`, track + stops), CSV of the track (`text/csv`)
Logbook PDF	card HTML → `save_pdf` (Android) or print dialog	A4, written by `PdfWriter.kt`
Debug log	`debug_export`	`tourist-explorer-debug-<stamp>.json`, section 9

Import (`library_import`, `Library.importData`, `data.ts`): only `format: "tourist-explorer"`. Adds what the person does not have yet (same id = same item, kept as is); each item goes through the same validation as an edit; routes keep only stops that are favourites or free times listed in their `free`, with their `pauses` and `free` (0.10.2: before, an imported route lost its visit times); trips are rebuilt field by field by `importTrip` (`data.ts`) and always arrive finished; tracks are taken only for trips without one, at most 100,000 points, each point checked (`cleanPoint`, `backend.ts`). Since 0.10.1 `importTrip` also keeps, each checked: on entries `missed`, `pause`, `pause_s`, `pause_skipped`, `pause_told`; on the trip `route_id`, `pauses`, `planned_at`, `free_plan`, `free`. `vehicle` is kept since 0.9.5 (only `van`, `motorbike`, `classic`; anything else becomes null), and `mode` accepts `auto`.

---

4. Services

`call(service, data)` (`backend.ts`): an unknown name throws `<service>: not available in the app yet`; after `BETA_EXPIRES` only `AFTER_EXPIRY` services answer (library, trip_state, trip_get, trip_track, trip_end, library_export, keys_status, settings_save, tile_get, debug_export, debug_clear); the `CAPPED` services (search, facts, picture, fuel_price) throw `DATA_CAP` when the data limit is reached; every error becomes an `Error` with the API keys removed from its text.

"Net" lists what the call may reach; "–" = local only. The dispatcher has **81 services** at 0.13.0 (75 at 0.11.2, 80 at 0.12.6; **NEW** `overpass_test`), every one in the table; 41 were app-only at 0.11.2. HA has 36.

Service	Input	Output	Net
`library`	–	{favorites, routes, settings, languages}	–
`trip_state`	–	{active (slim), trips (summaries), plans, interrupted {since} or null}	–
`screen_apply`	–	null (re-applies keep-screen when the app is back in front)	–
`search`	lat, lon, radius (5000), fast	{landmarks (ranked, ≤ 150), partial, complete}	Overpass, OpenTripMap, Wikipedia, Wikidata; capped

Service	Input	Output	Net
`info`	landmark	{title, extract, url, website, source, lang, image, translated_from?}	Wikidata, Wikipedia, OpenTripMap, Nominatim (native language), AI (translation)
`facts`	landmark, extract	{facts (≤ 3), ai_entity: null, source}	AI provider; capped
`picture`	file (Commons file name)	{url, page, author, licence}, {skipped: true}, or null	Commons; capped, skipped when roaming (`noroam`) or past 80 %
`keys_status`	–	{otm, tomtom, ign **NEW** (set or not), overpass **NEW** (the address), nextcloud **NEW** {url, username, password_set}, ai {provider, model, url, key_set}, providers, saved, default_models, ollama_url, google {client_id, secret_set, signed_in, email}}	Google token (finishes a pending sign-in)
`keys_save`	otm_key, tomtom_key, ign_key **NEW**, overpass_url **NEW** (must match `^https?://[^/]+/.*interpreter$`, "" = none), nextcloud **NEW** ({url (https), username, password; a missing	as keys_status	–

Service	Input	Output	Net
	password keeps the saved one} or null), ai ({provider, model, key, url} or null), google ({client_id, client_secret} or null)		
`overpass_test` **NEW**	lat, lon	{count}: map objects within 500 m (15 s)	The own Overpass server (« still importing? » on 503)
`keys_import`	data (a keys.json object)	as keys_status	–
`ai_test`	–	{answer, source}	AI provider
`otm_test`	lat, lon	{places} within 2 km	OpenTripMap
`tomtom_test`	lat, lon	{incidents} within 25 km	TomTom incidents
`google_signin`	–	as keys_status, after the browser comes back (≤ 5 min)	Google OAuth
`google_signout`	–	as keys_status	–
`google_test`	–	{addresses, email}	Google token, People API
`contacts`	query, refresh	{contacts, total, errors {google?, nextcloud? **NEW**, phone? **NEW**}}	People API, **NEW** Nextcloud CardDAV, the phone (`phone_contacts`); merged, memory 10 min
`find_place`	query, lat, lon, country	{places} (Wikidata first, transport last)	Nominatim search
`geocode`	query, country	{lat, lon, name}	Nominatim search
`where`	lat, lon	{name}	Nominatim reverse
`motorway_points`	query (town) or lat, lon; country	{points [{id, name, kind, lat, lon, carriageways, exit,	Nominatim, Overpass (15 km), cached 30 d

Service	Input	Output	Net
		distance}], center}	
`motorway_snap`	lat, lon	{carriageways (≤ 2), ref}	Overpass (300 m), cached 30 d
`cities`	country	{countries} or {country, cities}	GeoNames (once, ~3.4 MB zip)
`fuel`	lat, lon, radius (10000)	{stations} by distance; throws « Station search unavailable » when every full source failed (6.1)	Overpass, OpenTripMap; TomTom search, prix-carburants as fallbacks
`fuel_price`	lat, lon	{price} (France, within 10 km)	data.economie.gouv.fr; capped
`road_at`	lat, lon, heading	{road {id, highway, name, ref, tunnel, toll, limit, estimated, source?} or null, max, country} (`source: "tomtom"` and `country` since 0.9.5)	Overpass road tile and Nominatim country, both in the background; TomTom reverse geocoding when no OSM road (5.13)
`guide_route`	points (2-60), heading? (degrees, the car's course, sent while moving at 10 km/h or more)	{path, steps, distance, duration, by?}	TomTom routing with guidance (key; `vehicleHeading`), else OSRM (`bearings=<heading>,45`)
`plan_schedule`	plan {stops, dest, mode, start}	{start, end, drive_s, traffic, delay, distance, stops [{id, name, arrive, leave, pause}], nights [{at,	TomTom routing (car, key) or OSRM; Nominatim

Service	Input	Output	Net
		resume, lat, lon, name}]}	
`trip_start`	one of plan_id, route_id, trip_id, dest_point, destination; name, tracker, sim {pois}, record_only, reverse (0.10.2, with plan_id / route_id / trip_id), reason (debug log only)	trip_state	Nominatim; plan timetable in the background
`trip_dest`	dest_point {name, lat, lon}	trip_state	– (app only, 0.9.5: sets the running trip's destination; « Guide me to… » on a recorded trip, 5.19)
`trip_end`	at_interruption	trip_state	Background after the answer: Nominatim (end town), Overpass rails (train), OSRM (detours)
`trip_resume`	–	trip_state	–
`trip_day_end` / `trip_day_start`	–	trip_state	Nominatim (night's town) / –
`trip_free_start` / `trip_free_end`	note / –	trip_state	–
`trip_leg`	mode (walking, bicycling, driving, bus, train, boat)	trip_state	–
`trip_plan`	landmark	trip_state	–
`trip_skip`	id, via (`banner`, `card`, else `trip`: debug log only, 0.10.3)	trip_state	–
`trip_here`	id (a waypoint, or `dest`), lat, lon	trip_state (the waypoint or destination visited)	– (0.12.1, « 📍 I'm here »)
`route_preview`	points (1-58), mode?	{path, km, drive_s, delay, traffic, litres, cost}	TomTom routing with traffic (key), else OSRM (0.12.1)
`departures`	lat, lon	{stops: [{name, icon, distance,	transport.opendata.ch (Switzerland, 0.12.1)

Service	Input	Output	Net
		departures: [{at, line, to, platform, delay, operator}]}]}	
`trip_unskip`	id	trip_state (the waypoint pending again; nothing when it was not skipped)	– (0.10.3, also in HA)
`trip_dismiss`	id	trip_state	–
`trip_log`	trip_id, entry_id, rating (1-5), note (≤ 1000)	trip_state + {trip}	–
`trip_get`	id	{trip}	–
`trip_delete`	id (not the running trip)	trip_state	–
`trip_detour`	lat, lon, text	{ok} (one line per 2 min)	–
`trip_eta`	–	{active, at, next, dest ({name, distance, duration, late}), nights, traffic, delay, error?}	TomTom or OSRM, cached 150 s (15 s in a test trip) or until the next waypoint changes
`trip_track`	id, limit (1500, 10-5000)	{trip_id, points (downsampled), stats}	–
`gps_log`	points (≤ 500)	{kept}	–
`traffic_ahead`	path ([[lat, lon]], ≤ 400)	{incidents (≤ 10, within 200 m, nearest first), available}	TomTom incidents
`traffic_tiles`	–	{url (flow), day, night} with the TomTom key, or nulls; **NEW** `scan25` (the SCAN 25 tile URL) with the	– (the card then loads tiles)

Service	Input	Output	Net
		IGN key, or null; **NEW** `mapy` (the Mapy.com tile URL, `{set}` = outdoor or winter) with the Mapy.com key, or null	
`sim_path`	points (2-60), mode	{path, stops, legs, distance, duration}	OSRM / FOSSGIS, cached 1 d
`sim_position`	lat, lon, speed, t	trip_state	via the trip engine
`favorite_places`	max (5, ≤ 20)	library + {left}	Nominatim reverse; also looks up again the places named in another app language (0.11.1)
`item_places`	max (5, ≤ 20: new Nominatim look-ups per call; cache hits are free)	library + trip_state + {region_names {code: name}, left (items and names still to do), asked (look-ups made)}	Nominatim reverse zoom 5 through `regionOf` (cached) and, for a region name not cached in the app language, one more at a point of it. App only (0.11.3). The card asks while an item has no `places` or once per app language, and stops when a call asks nothing and something is still left (no network) until the app restarts
`favorite_add` / `favorite_remove`	landmark / id	library	–
`route_save` / `route_delete`	name, stops (favourite ids and `free:` ids), id, dest, description, mode, pauses {stopId: min}, free {freeId: note} (0.10.2) / id	library (each route with `order` and `free`)	–
`plan_save` / `plan_delete`	plan / id	trip_state + {plan}	–

Service	Input	Output	Net
`settings_save`	settings	library	–
`library_export`	trip_id, pick	export file object	–
`library_import`	data	{added, … library, … trip_state}	–
`data_status`	–	{used, limit, level, counted, roaming, mobile, month}	–
`location_access`	{ask?}	{background: true / false / null}	Android « all the time » location: state, or asks for it (`ask`); null off Android (0.9.4)
`data_more`	–	as data_status, +50 MB this month	–
`debug_export`	–	debug file object	–
`tile_get`	url (https)	Blob (the tile, kept or downloaded)	The map's tile server; answers after the beta's end too
`tile_cache_status` / `tile_cache_clear`	–	{bytes, count, limit}	–
`car_status`	–	{total_km, a, fuel {per100, estimated, since}, departure, tank_l, litres_left, range_km, fills (last 5)}	–
`car_fill`	litres (0.5-300), full, price (per litre, 0.1-10, optional)	as car_status	–
`car_set`	odo?, range? (0.12.3), a? {km, per100, avg_kmh}, start? {…} (blank = unchanged)	car_status	– (0.12.2, ✎ From the car)
`car_reset`	what (a, fuel, all)	as car_status	–

Service	Input	Output	Net
`logbook_photo`	landmark, inline	{url (330 px), author, licence, data?} / {skipped} / null	Wikipedia (once, if the description was never fetched), Commons; capped
`debug_clear`	–	{}	–
`debug_note`	k (`guide`, `no_fix`, `speak`, `gps_source`: `DEBUG_NOTES`, else « unknown debug note »), data	{}	– (app only, 0.9.5: card-side events into the debug log; ≤ 8 fields, names ≤ 20 chars, strings ≤ 120)

HA services the app does **not** have: `contacts_status`, `contacts_setup`. HA has `trip_unskip` since 0.10.3 (`trip_unskip` in `__init__.py`, `unskip` in `trip.py`). HA's `route_save` accepts `pauses` and `free` since 0.10.2 (before, its schema refused the `pauses` the card sends); HA's `trip_start` has no `reverse`.

⇵ **Reverse** (0.10.2). `trip_start` with `reverse: true` starts the plan, route or past trip the other way: `reverseStops(stops, dest, origin)` (`card/src/reverse.ts`, pure, shared with the card, tested in `card/test/reverse.test.ts`) reverses the stops (each keeps its `pause_min`; free times move with the places around them and are placed again at the place before them). The new destination: the origin when one is known, else the first original place taken out of the stops when there was a destination, else none (the reversed stops end at the first original one). The original destination is dropped. Origins: a past trip's first GPS point, else its first leg's `from`, named by its town (reverse geocoding, "" when offline) — `tripOrigin`, `backend.ts`; plans and routes have no start place (a plan's `start` is a time), so they use the destination rule. The trip is named `<name> (return)` (`trips.ts` `ret`, translated), a plan's `start` time is dropped, `route_id` is kept (■ on the Routes tab), and the debug log's `trip_start` gets `reverse: true`. The card shows ⇵ beside ▶ on routes, plans and past trips in the app only. « ⧉⇄ Save a reversed copy » (routes and plans, app and HA) is card-side with the same function through `route_save` / `plan_save`, named `<name> (return)` (`returnSuffix`); a one-place route with a destination reversed keeps the place as its stop, without destination. `tripAsPlan` keeps each planned place's visit time (from `trip.pauses`) since 0.10.2, for ▶ and ⇵ alike.

---

5. The trip engine

5.1 Position flow

1. **Kotlin** (`TripLocationPlugin.kt` `watch`, `startUpdates`): `watch` asks for location (and, on Android 10+, activity recognition), then `startUpdates()` uses **`LocationManager`** directly: `GPS_PROVIDER`, or `NETWORK_PROVIDER` only when GPS is switched off, every 1000 ms, 0 m. It first emits `getLastKnownLocation()`, which can be an hour old. There is no FusedLocationProvider; `play-services-location` is used for activity recognition only. Each fix is a `position` event {lat, lon, alt, speed m/s, acc, heading, t ms}. `GnssStatus` gives `gnss` events (at most 1/s); `ActivityRecognition` every ~10 s gives `activity` events through `ActivityReceiver.kt`.
2. **`android.ts` `watchPositions`**: subscribes once; logs `fix` in the debug log; converts speed to km/h and t to seconds; attaches the last activity if younger than 3 minutes; calls `device.update`.
3. **`device.ts`**: keeps `current`, calls the listeners: `main.ts` rebuilds `card.hass` (the card reads `device_tracker.this_device` with HA units: speed m/s, `course`), and `TripManager.watch` (`trips.ts`, via `followDevice`, which first replays the current position).
4. **`TripManager.onPosition(userId, tripId, pos)`** (`trips.ts`) under the trip's lock.

On the desktop, step 1-2 is `navigator.geolocation.watchPosition` (`gps` in `main.ts`). GPS on/off: the card's 📡 switch (`tourist-explorer-gps`), and when the app goes to the background without an active trip the GPS is switched off (the `visibilitychange` listener in `main.ts`); during a trip the foreground service keeps it going.

Trip start takes the live GPS (0.9.5, card `liveGpsForTrip`, called when the active trip changes): on 2026-10-30 a trip was started while the map sat on « Manual location » and recorded nothing. For a real trip (not 🧪 / 🎬, not in its night) a map source `manual`, `sim` or `play` is switched to `phone` (debug note `gps_source`) and a 📡 switched off is switched on. If no fix of this device's tracker newer than the start arrives within `FIX_WAIT_MS` (30 s), the Trip tab shows `gpsNoFix` (« No GPS position since the trip started… ») until one comes (`fixSince`), and a `no_fix` note {trip, last (s since the last fix, or null)} goes to the debug log. In the app the source list has one « 📱 This device — live GPS » fed by the app's own tracker (`readEntitySource`), not a second « This device » tracker or person.

5.2 The per-trip lock and the background work

`locked(tripId, fn)` (`trips.ts`) chains positions per trip: one at a time, in order, like the HA `asyncio.Lock`. **Nothing that needs the network may be awaited inside it.** On the 2026-10-29 road test (Champex → Genève, 0.9.0) `onPosition` awaited the place search (Overpass, up to 75 s per mirror), weather,

reverse geocoding and the train check under the lock: positions queued up behind them for ~160 s at a time, the track had 23 holes and 2-3 km jumps, and the first fix after a hole was stale. Since 0.9.1:

- `bg(work)` (`trips.ts`) runs such work detached, logs its failure, and tracks it in `background`; `idle()` waits until all of it is done (tests).
- The place search runs in the background and applies its result under the lock later (`search`, `trips.ts`); one search per trip at a time (`searching`).
- The road lookup is non-blocking: `waysAt` answers `null` while a tile loads (`waysAt`, `backend.ts`), and the country is the cached one or the last known one (`countryNow`, `backend.ts`); Nominatim is never awaited by a fix.
- The track point uses the GPS fix time, not the handling time. A fix ≥ 600 s old (Android's last known position) is dropped as `stale`; a fix older than the last one handled is dropped; handling > 5 s after the fix is logged as `lag`, handling > 1 s as `slow_pos`.
- `trip_end` returns at once; network work after it runs in `afterEnd` (`backend.ts`).

The traffic log runs through `bg()` too since 0.10.1 (before, `void this.track(…)` escaped `idle()`).

5.3 What `onPosition` does, in order

1. Drop stale fixes; record the point `[t, lat, lon, alt, km/h, acc, src]` (unless an open GPS page pushed richer fixes in the last 30 s, `CARD_PRIORITY_S`). Thinning: `gpslog.append` keeps a point ≥ 5 s after the last if it moved ≥ 15 m or climbed ≥ 5 m, always one a minute; beyond 30,000 points every second point goes.
2. **Legs** (`T.trackLeg`): a change of mode; a finished driving leg is checked for a train in the background.
3. **Heading** (`updateHeading`): bearing after 200 m of movement.
4. **Road log** while driving: `roadAt` (non-blocking), main-road timestamp, `roads.trackRoad`; a new road gets its town in the background; every 1000 m driven a town lookup in the background.
5. **Stretch end** fixed; **passed** waypoints (`markPassed`, with `pause_skipped` if the stop had a planned pause); **seen** sights (`markSeen`); **missed** waypoints while driving (`markMissed`).
6. **Traffic** every 5 min of driving; **weather** at the start and hourly (background).
7. **Visits and stops** (`trackMotion`), weather and stop name in the background.
8. **Pauses and break reminder** (`pauseAndBreak`).
9. `skipBehind`; if the stretch end is gone, the next stretch and its notification.
10. **Search** ahead, if `auto_search` is on and the look-ahead says so (background).
11. Save (at once if something important happened, else at the `save_every` interval).

5.4 Legs and modes

`trackLeg` (`trip.ts`). Evidence per fix: the activity (`on_foot`, `walking`, `running` → walking; `on_bicycle` → bicycling; `in_vehicle` → driving; `still`/`tilting` → no evidence), else the speed (≤ 7 km/h walking, ≥ 30 km/h driving, in between no evidence, < 1.5 km/h standing). A change counts after 60 s of the same evidence (`LEG_SWITCH_S`); the new leg starts where the change began. The very first leg starts at once and is corrected if wrong within 60 s. Driving → walking writes a 🅿️ `parked` entry. A finished driving leg ≥ 3 km whose points (~40 sampled) lie within 25 m of OSM railways for 80 % becomes `train` (`isTrain`, Overpass `around:40`). The person can set the leg by hand (`trip_leg`, `setLeg`): bus, train and boat count as "vehicle" for detection, so a manual leg survives until a walk. Radii follow the current leg (`radii` and `RADII`, `trip.ts`):

Mode	Passed within	Visit within	Still within
driving (default)	500 m	150 m	200 m
walking	60 m	75 m	60 m
bicycling	150 m	100 m	100 m

5.5 Visits, stops and the main-road rule

`trackMotion` (`trip.ts`). While within the "still" radius of where we stopped:

- near a known place (proposals, logbook places, `nearby`) for `min(5 min, stop_minutes)` → **visited** (a planned entry turns into visited; passed/skipped/seen lines of that place are removed);
- elsewhere for `stop_minutes` → **stop** (or the open `parked` entry is reused).

Moving out closes the entry (`left`). **Main-road rule** (0.9.1): if the road under the car was motorway, trunk, primary or secondary (or their links) in the last 30 s, standing still is traffic, not a visit nor a stop (`MAIN_ROAD` in `trips.ts`). On 2026-10-29 a jam beside the botanical garden in Genève had become a visit.

5.6 Passed, missed, skipped waypoints

- **Pending waypoints** (`waypoints`, `trip.ts`): `planned` minus visited, passed, skipped, in order.
- **Passed** (`markPassed`): a pending waypoint within the pass radius of the segment from the previous fix to this one (not just the point: at motorway speed fixes are far apart); skipped while we are still within the radius (we may stop).
- **Missed** (`markMissed`, 0.9.1, driving only): the closest approach to each waypoint is remembered (`closest`); once it came within 15 km and we are now 5 km farther away, it is logged `skipped` with `missed: true`. On 2026-10-29 the route kept leading back to Chillon, Montreux, the Olympic Museum, Rolle, Nyon from the A9/A1: « off route » twenty times.
- **Skipped behind** (`skipBehind`): once a later waypoint is passed or visited, earlier unreached ones are skipped.

- **By hand**: `trip_skip` (× in the Trip tab, ⏭ on the driving screen, 5.19). **Undo** (0.10.3): `trip_unskip` → `unskip` (`trip.ts`, HA `trip.py`) removes the waypoint's `skipped` lines (a `missed` one too) and forgets its `closest` approach, or `markMissed` would skip it again at the next fix; null for a place not skipped or not planned. `skipBehind` may still skip it again once a later waypoint is reached.

5.7 Seen places, proposals and search

`searchCenter` (`trip.ts`): the first search waits for a heading or 3 minutes; then the circle (trip radius) is centred half a radius ahead and the next search comes after driving half a radius. `propose` keeps places with ≥ max(1, `min_stars`) in the chosen categories, ahead (±90°) or within 2 km, never twice; at most 3 notifications per search. `nearby` keeps up to 200 rated places for arrival detection. **Seen** (`markSeen`): a proposed, not planned, place driven past within 150 m (★★★: within the pass radius) is logged once as `seen` (0.9.1 narrowed it after ~70 noisy ★★ lines on the road test). `auto_search: false` makes the trip a plain recorder.

5.8 Stretches

Navigation links carry at most 8 waypoints (`STRETCH`, ViaMichelin's limit). `stretch()` / `stretchEnd()` (`trip.ts`): with more, the link ends at the 9th; in `eta` mode, 7 waypoints then always the destination. The current stretch's end is fixed in `stretch_end`; when it leaves the pending list a « 🧭 Next stretch » notification carries the next link (`notifyStretch`).

5.9 Pauses, visit times and the break reminder

`trip_start` turns each plan stop's `pause_min` (or `pause_default_min`, 0 when `pauses_on` is off) into `trip.pauses[poiId]` seconds (`tripStart`, `backend.ts`). `pauseAndBreak` (`trips.ts`): at a stop with a planned pause the entry gets `pause_s`, and once over, one notification « ⏱ The visit at … is over ». A waypoint passed without stopping gets `pause_skipped`. Break: `wheel_since` starts at ≥ 10 km/h and resets after a stop of 15 minutes; after `break_every_h` hours of driving, one « ☕ » notification. Visit lengths up to 10 h (`pause_min` ≤ 600).

5.10 Free time

⏳ Free time (`trip_free_start`/`trip_free_end`: `tripFreeStart`/`tripFreeEnd`, `backend.ts`): a `free` entry at the current position, named by the typed comment or the plan's next free stop (`free_plan`, consumed in order), with `pause` from the plan. At its end, `left` is set and the distance walked since (from the track) goes into `note` when ≥ 50 m. Plan stops with `free: true` never become waypoints. Routes have free times too since 0.10.2 (3.2); in both editors the « ⏳ Free time » button sits beside the add-a-stop controls with a hint, and a free time's length picker shows even with `pauses_on` off.

5.11 Nights (multi-day trips)

🌙 `trip_day_end` (`tripDayEnd`, `backend.ts`): closes an open stop, stops following the device, stops the foreground service, ends the current leg and road, writes a `night` entry named by the town, sets `trip.night`. ☀ `trip_day_start`: closes the night entry, appends the day's start to `days`, resets the leg state and resumes. Ending the trip during the night removes the night entry and ends the trip at the night's start. At app start a trip in its night is not offered for resume.

5.12 Interruptions and durability

At start (the `state` start-up in `backend.ts`, `TripManager.startAll`), test trips are ended; a real active trip whose last position (track or leg) is < 120 s old resumes silently; otherwise `interrupted` is set and the card asks: resume (`trip_resume`, writes a ⚡ `gap` entry from the interruption to now) or end at the interruption (`trip_end {at_interruption: true}`).

5.13 Road log and speed limits

`roads.ts`. Roads come in tiles of 0.01° (~1.1 km): one Overpass query per tile for motorway…living_street ways with `out tags geom`, timeout 15 s server / 20 s client (0.9.5; urgent in the Overpass queue, 6.1); the last 40 tiles in memory; a failed tile is retried after a minute. `roadAt` takes the nearest way within 30 m whose segment bearing is within 50° of the heading (either direction). The way the sign is already on counts at half its distance (`keep`, 0.12.1), and `steadyRoad` holds a limit that drops to half or less while the speed is still 20 km/h above it for 15 s before it shows (debug `road_held`); a road lost for under 10 s keeps the last one. Test drive, 2 November 2026: at 100 on the motorway the sign showed 30, a side road. Limit: the posted `maxspeed` (`50`, `50 mph`, `CH:urban`), else the country's default for the kind (`defaultLimit`: motorway, trunk, living street; any other kind gets the urban limit, the lowest). Country table: CH, FR, DE (motorway none), IT, AT, NL, BE, ES, GB. `estimated: true` marks an implied limit. `road_at` also returns `max`, the country's motorway limit, for the speed buttons, and (0.9.5) `country`, for the sign set.

TomTom fallback (0.9.5, `roadHere` → `tomtomRoad`, `backend.ts`): on 2026-10-30 Overpass loaded 4 road tiles in two hours and the signs stayed blank. When no OSM road matches (no tile, or no way within the snap rules) and a TomTom key is set, the road comes from TomTom Reverse Geocoding (`tomtom.speedLimitUrl`, `returnSpeedLimit=true&returnRoadUse=true`, `&heading=`; `parseSpeedLimit` turns « 80.00KPH » / « 50.00MPH » into km/h). Never awaited by a fix: a call only when the last one is ≥ 10 s and ≥ 100 m away (`TT_EVERY_MS`, `TT_EVERY_M`), in the background, 10 s timeout; its answer serves fixes within 300 m of where it was asked (`TT_SAME_M`; **NEW** 0.13.0: no longer for 12 s, which at ×60 is 12 km). The road gets `source: "tomtom"`, `estimated: false`, an id = minus a hash of ref or name (never an OSM id, so a hand-set limit holds on it as on a way), and a kind from TomTom's road use (LimitedAccess → motorway, Ramp → motorway_link, Arterial/Rotary → primary, Terminal/LocalStreet → residential, else

unclassified), so the road log and the main-road rule work on it too. The card marks such a limit (`.speedbtn.tt`: dotted blue outline, tooltip `limitTomTom` « Limit from TomTom »). A change of source logs `road_src` (9).

Sign sets per country (0.9.5, `signSet(country, mph)`, `card/src/units.ts`): the sign column shows the limits the country posts, plus the motorway `max` and the road's own limit when different, in order (km/h; GB and US in mph):

Country	Signs
FR	30 50 70 80 90 110 130
CH, LI	30 50 60 80 100 120
DE, AT	30 50 70 100 130
IT	30 50 70 90 110 130
ES, PT	30 50 70 90 100 120
BE	30 50 70 90 120
NL	30 50 60 80 100 130
LU	30 50 70 90 130
GB	20 30 40 50 60 70 mph
US	25 35 45 55 65 75 mph
other	40 50 60 80 100 km/h, or 20 30 40 50 mph with `us`/`uk` units

The **road log** (`trackRoad`, `roads.ts`): one entry per road (key = ref, else name, else kind). A new road must be seen again ≥ 20 s after it was first seen; several candidates are tracked at once and one not seen for 60 s is forgotten (0.9.1: on the A9 a single candidate was reset by every slip road and tunnel, and one « residential ≈50 » line ran 101 km). Each entry keeps distance and top speed; the town comes from a background reverse lookup.

5.14 Towns

Every 1000 m of driving (`TOWN_EVERY_M`) a reverse lookup (Nominatim, queued 1/s, background); a name different from the last one is appended to `towns` (≤ 1000). A road entry's town counts as the last town.

5.15 Traffic log

With a TomTom key, every 300 s of driving (`TRAFFIC_EVERY_S`, 12 requests/hour): incidents in the bounding box of the stretch driven since the last check (pad 0.005°), kept if one of their points is within 150 m of the track and, where it meets the track, runs within 60° of our direction (`onTrack`); each TomTom id once; ≤ 200 kept. ↻ deviations from turn by turn come from the card (`trip_detour`). Road-test finding G (incidents in both directions and on side roads) is what the direction filter addresses. Only incidents met on the route are kept, with the trip (logbook, exports); nothing else from TomTom is stored (the header comment of `tomtom.ts`).

5.16 Timetable

`engine/schedule.ts`. `timetable(legs, pauses, start, rules, drivenToday)` walks the legs' driving seconds, inserting pauses at the stops and nights when the day's budget (`drive_max_h`) or the driving window (`drive_from`-`drive_to`) runs out; a night falls part-way through a leg (`fraction`) and resumes the next morning at `drive_from` (DST-safe midnight). `leaveFor` (« arrive by ») bisects the departure over 14 days, 40 iterations, to the minute. `plan_schedule` routes the plan (TomTom with traffic for car/motorbike and a key, else OSRM), places each night on the route line (`along`) and names its town. `trip_eta` reruns the timetable from the current position with today's driving so far and the time left in a running pause, and reports minutes late against the plan.

5.17 Approach popups

Card side (`approach` in `card.ts`, 0.9.3): during a trip, within `approach_km` of a pending waypoint or a ★★★ place, its popup opens by itself, once per trip and place, and closes after 8 s unless touched. Never below 3 km/h, while the screen was touched in the last 30 s, while another popup is open, or when a manoeuvre is less than 1 km away. With `speed_beep` on it beeps once (not with `voice: off`); with `voice: voice` it says « {name}, {dist} ahead » instead. While driving the place opens docked (5.19).

5.18 Notifications

`notify.ts`: `@tauri-apps/plugin-notification`, permission asked at the first one. Two action types: `trip-place` (🧭 Go → opens the link, Not now → `trip_dismiss`) and `trip-stretch` (Go). A tag hashes to a fixed id, so the same place replaces its notification. Texts: `TEXT` in `trips.ts` (en, fr), else the language pack's `trip` strings. Quiet trips (record only, 🧪 test) send none.

5.19 The driving view: docked place card, destination, voice, vehicle (0.9.5)

All card side (`card.ts`), standalone only.

- **Docked place card** (`docked`, `showDock`, `closePlace`): while the running trip's mode in force is driving or motorbike, a place opens in `.placedock` over the bottom of the map (max 48 % high; in landscape on the right, ≤ 340 px), with a ×, instead of a Leaflet popup pinned to the place: each fix moved the map and the popup jumped (2026-10-30). The map keeps following underneath. On foot and without a trip: the popup, with the map held still under it (0.10.4, 5.20). Approach popups (5.17) use the same path (`placeOpen`).
- **Turn by turn needs a destination** (`guideToLine`): a trip with no destination and no pending waypoint (a recorded one) gives the guide nothing to follow. With `guide` on, driving/motorbike, not 🧪 and not at night, the Trip tab says

guideNeedsDest and offers « 🧭 Guide me to… » (destChooser: favourite, contact, address), which calls trip_dest. The banner's hide reasons (no trip, night, guide off, mode, route done) go to the debug log (hideGuide, note guide).

- **Voice** (voice setting, speak): voice speaks the manoeuvre at 300 m and, for an exit or fork, also at the exit_warn_m warning, each once (said, keyed by route, step and level); places on approach (sayAhead, « {name}, {dist} ahead ») instead of the beep; « Speed limit {limit} » instead of the over-speed beep. Distances are rounded for the ear (spokenMetres: 50 m steps below 1 km, then 100 m); spokenText (guide.ts) turns « → » into « towards », « · » into a pause and drops arrows and emoji. The card fires tourist-explorer-speak {text, lang (app language)}; main.ts cancels it on Android and calls the plugin's speak (7); elsewhere Web Speech (speechSynthesis.cancel() then speak), falling back to a beep. Each spoken line is a speak debug note.
- **My vehicle** (vehicle setting, vehicleNow, placeMe): instead of the blue dot, a 40 px marker from app/public/vehicles/ (VEHICLE_ICON: van → marcopolo-marker.png, motorbike → gs1200-marker.png, classic → healey-marker.png), kept upright, with an orange (#E69F00) wedge on its ring turned to the heading (hidden without a heading). auto follows the mode in force (the trip's, else Setup's). The trip's vehicle (3.2) shows as an icon before the logbook heading.
- ⏭ **Skip** (0.10.3, renderSkip, skipWaypoint, rerouteSoon): one 48 px button, shown whenever the running trip (not at night) has a pending waypoint. It sits in the turn-by-turn banner (the banner's text is now its own span, the live region) when the banner shows, else alone on the map under ⛶ (.skipmap; hidden in landscape while the dock is open, which has its own ⏭). One tap skips pendingStops(trip)[0], the waypoint the guide heads to (tripStops starts from the same list), with trip_skip {via: "banner"}. Then: a toast over the bottom of the map (.skiptoast) « Skipped: {name} » with **Undo** (trip_unskip) for 8 s; with voice: voice, « {name} skipped »; and 1.5 s after the last skip or undo the guide's 20 s throttle is reset and updateGuide runs, so guide_route is asked for the next waypoint or the destination at once (the stops key changed). The Trip tab's × is unchanged (via: "trip").
- **The place card during a trip** (0.10.3, popupNode, docked or popup): a pending waypoint gets « ⏭ Skip » (via: "card"); a proposed place (in trip.next, or one that opened by itself on approach) gets « 🚫 Not interested » → trip_dismiss (it goes into seen, never proposed again on this trip). The × only closes.

5.20 0.10.4: the popup holds the map, no silent trip replacement, reviews, Setup « Simple », list bands

All of this section is new in 0.10.4 (0.10.5's one line: last bullet). All card side unless said.

- **The map holds still under a popup** (`popupHold`, `following()`, the map's `popupopen` / `popupclose`): a popup pinned to a place near the edge auto-panned the map, the next fix pulled it back (follow mode) and the popup jumped up and down, in any mode (the team, 2026-10-31). While any Leaflet popup is open (place, map-tap menu) `following()` is false: no `setView` per fix, no pan on resize, and the map is north up (`applyTurn`; if the turn changed, the popup's auto-pan runs again on the upright map). `follow` itself is untouched, so a drag still stops following and a « look at this place » click (which sets `follow = false`) is not undone. On close (×, a tap on the map, the approach popup's 8 s) the map centres on me again if still following; deferred by a tick, because a popup replacing another closes the first just before the second opens. ⌖ (`recenter`) closes the popup first. Auto-pan stays on (a popup that grows when its photo arrives is pulled into view; with the map held there is no loop). The docked card (5.19) is unchanged: nothing is pinned to the map, so the map keeps following under it. Chosen over « dock during every trip »: it also covers the no-trip case and the tap menu, and keeps the walker's popup by the place.
- **A new trip never ends the running one without asking** (`startTrip`, `askRunningTrip`, `startGuard` in `nav.ts`): on 2026-10-30 picking the Gare Cornavin favourite ended a test trip (`trip_end` reason `new trip`). Every card call of `trip_start` except the 🧪 simulator goes through `startTrip`: the Trip tab's ▶ Start (with or without a destination) and ● Record, ▶ / ⇄ of a plan, a route and a past trip, the plan editor's ▶, and Go « in this app » (`guideHere`, which starts a trip when none runs). It asks `trip_state` afresh (the card's copy may be old), then `startGuard(active, data, standalone)`: no trip, or a 🧪 one → start; a real trip and a destination (app only: HA has no `trip_dest`) → a dialog « Set as this trip's destination » (focused, the default: `trip_dest`, the trip and its log kept) / « End this trip and start a new one » (then a `confirm`) / Cancel; a real trip and anything else → « End this trip and start a new one » / Cancel (Cancel focused). The backend's `trip_start` is unchanged (it still ends the running trip); `backend.test.ts` « new destination during a trip » checks both sides.
- **Reviews** (`reviewLinks`): « ⭐ Google reviews » (`googlePlaceUrl`: Google Maps search by name at the place's coordinates) and « TripAdvisor » (`tripadvisorUrl(name, town)`: the favourite's `place.city` when known, never twice) in the Infos tab, the place popup / docked card, and each logbook entry (which already had « Google Maps » / « TripAdvisor », same URLs). Plain `target="_blank"` links, opened by `main.ts` with `openLink`. No API.

- **Setup « Simple · All »** (`setup_view`, 3.1; `SETUP_SIMPLE` near the top of `card.ts`): a segmented switch at the top of Setup. Simple shows only the topics in `SETUP_SIMPLE`, named by their title's i18n key (`section(key, …)` in `renderSetup`), plus `vehicle` (the car: fuel, tank, consumption) and whole groups by id (`keys`: AI and other keys, keys import; `data` is not in it); a group left empty is not shown. Today: `uiLang`, `infoLang`, `unitsTitle`, `travelMode`, `navigation`, `approachTitle`, `voiceTitle`, `myVehicle`, `vehicle`, `colours`, `mapStyle`, `picturesTitle`, `dataLimitTitle`, `keys`. Default `all` (`DEFAULT_SETTINGS`), but a user who never saved a setting (a new install) reads `simple` (`NEW_USER_SETTINGS` in `data.ts` / `data.py`), written with the first save so it stays: existing users keep the whole Setup.
- **Fold titles as bands, lines between items** (CSS only): `button.favhead` (every `foldSection`: Favourites by kind and by place, the route editor's favourites, Routes, the Trip tab's plans and previous trips; also « Next up ») shares one rule with `details.setgroup > summary` (0.10.2): bold, rounded, 18 % of `--primary-color` into the card background (`--band`), plain fallback; not sticky (nested place levels would pile up); inside `.foldin` the band is 11 %, one level deeper 6 % (country › region › city). Items of a list in a fold (`.favgroup .list li`) get 6 px padding and a line of 22 % `--primary-text-color` (fallback `--divider-color`, which was lost among the buttons' own borders), none after the last.
- **Turn by turn seen** (`showGuide`, `updateGuideMini`): while the banner shows, the map's top-left column (zoom, speed signs, my speed; z-index 1000, it covered the banner's start) moves below it (`.guiding .leaflet-top.leaflet-left`, `--guide-bottom` measured at each update). When the map is hidden (Instruments, Setup, Help, 🗺 off) or under 260 px high (portrait with the panel open), the same line also shows at the top of the panel (`.guidemini`, warn colours alike); a tap opens the full map (⛶). Debug log: the `guide` note says `tab` and `map` when shown, and once per route « no next step » with `off`, `along`, `steps` (9). `locate` always answers once a route is held, so a « no position on route » case cannot occur and is not logged.
- **Colour-blind traffic** (6.1): orange, red and stopped traffic are recoloured too.
- **One preview line** (0.10.5): in Routes and the plan editor « 🗺 Preview route & costs » (`previewRoute`) offers Google Maps and ViaMichelin side by side (`appChoice`, kind `route`); the separate « Route costs » line is gone (ViaMichelin's page shows tolls and fuel).

5.21 The experimental 3D map (0.11.0, 0.11.2)

New in 0.11.0; opening and closing changed in 0.11.2. App only (`config.standalone` and `config.map3d`). Card glue in `card.ts` (`setMap3d`, `scene3d`, `update3d`), types in `card/src/map3d-api.ts` (types only), the MapLibre view in `app/src/map3d.ts`, its pure parts in `app/src/map3d-style.ts` (tested in `app/test/map3d.test.ts`).

- **Button**: « ⛰️ » in the map's zoom column (under ⌖ and 🧭). The 3D view (`.map3d`) lies over the 2D map's box (`mapWrap`, z-index 1000: over Leaflet and its controls, under ⛶, the docked card and the fuel question), so ⛶ full screen works in 3D too. Its own « 2D » and « ⌖ » buttons sit top left. The Leaflet map keeps running underneath; leaving 3D calls the view's `destroy()` (MapLibre's `map.remove()`), then Leaflet's `invalidateSize()` and (0.11.2) `setView` to where 3D was looking, unless 3D was still following. The last choice is kept in `sessionStorage` (`tourist-explorer-3d`) for this session only; a reload in the same session reopens 3D.
- **Loading**: `main.ts` passes `map3d: () => import("./map3d");` MapLibre is fetched only on the first ⛰️ (its own chunk, `dist/assets/map3d-*.js`, 1,127,506 bytes, ~291 kB gzip at 0.11.2; MapLibre's CSS imported with `?inline` and put into the card's shadow root by `addCss`; the worker is a separate asset `maplibre-gl-worker-*.mjs`, 508,314 bytes, set once by `setup()` with `setWorkerUrl` from a Vite `?url` import: same origin, no `blob:`). The HA card never imports it: it carries only the glue and strings (the HA bundle is 809,829 bytes at 0.11.2; 787,186 before 0.11.0).
- **Engine and style** (`style3d`): MapLibre GL JS **6.14.0** (pinned), WebGL. Sources: `base` (raster, the base map chosen in Setup, the same URL and attribution as the 2D map: `base3d` is set by `applyMapStyle`; Leaflet's `{s}` becomes a, b, c), `terrain` and `shade` (raster-dem, AWS Terrarium `s3.amazonaws.com/elevation-tiles-prod/terrarium/{z}/{x}/{y}.png`, encoding terrarium, tileSize 256, maxzoom 15; two sources because MapLibre warns when hillshade and terrain share one), `lines` (GeoJSON). Layers: base, hillshade (exaggeration 0.25), solid and dashed lines. Terrain exaggeration 1.3, pitch 60° (max 75°), sky and fog. MapLibre zoom = Leaflet zoom − 1 (`zoom3d`, 512 px vs 256 px tiles).
- **Tiles**: every tile URL is rewritten `https://` → `nestour-tile://` (`toCached`); `addProtocol("nestour-tile")` turns it back and asks the backend's `tile_get` (cache first, counted in the data cap, TomTom's key stripped from the cache key by `tileKey`). Requests for the same URL in flight are shared (terrain and hillshade ask the same tile), each caller gets its own copy of the buffer (MapLibre hands it to its worker). A failed tile is a `console.warn`, not an error.
- **Content** (`scene3d`, sent on each fix, each marker refresh and each map-style change): me (the blue dot `.medot`, or My vehicle's picture `.vehmark`, with the orange wedge turned to heading − map bearing); the plan line, the driven track and the guide line (same colours, widths, dashes as 2D); the places shown on the 2D map (same `markerSvg`, title = stars and name) and the trip's pending waypoints (amber). A tap on a place calls `openPlacePopup`, which in 3D always uses the docked card (`.placedock`, raised over the 3D view): no Leaflet popup can show there. Approach popups go the same way.

- **Where it opens, and follow** (0.11.2): `open(el, scene, opts)` gets the 2D map's `zoom`, `center` and `follow` (`Open3dOptions`). `start3d(follow, me, center)` (`map3d-style.ts`) puts the camera on me only while 2D was following, else on the 2D map's centre (a favourite shown in 2D stays shown; before, 3D jumped to the « Manual location »); zoom `zoom3d(zoom)`, bearing the heading when following, else 0. While following, the camera eases to me with bearing = the 2D map's smoothed heading (`mapHeading`, from `smoothHeading`: the heading above 5 km/h, else the last one); when not, 3D never moves by itself. A drag stops following; the 3D ⌖ (`recenter`) follows again. On « 2D » `View3d.view()` returns {lat, lon, zoom: `zoom2d(zoom)` (= round(zoom + 1)), follow}: if 3D no longer followed, 2D sets `follow = false` and opens there. Tested in `app/test/map3d.test.ts`.
- **Not in 3D**: speed signs, the speed buttons, the turn-by-turn banner, ⏭ on the map, « Search this area », the map info line. A hint says so once a session (`map3dHint`, 7 s).
- **Errors**: no WebGL (checked on a test canvas) → `map3dNoWebgl` in the status line; the import or `new Map` failing → `map3dFailed (reason)`; both stay in 2D.
- **Tauri**: `tauri.conf.json` has `"csp": null`, so neither the worker nor the custom protocol needed a CSP change (the protocol is MapLibre's own, inside JS: no request leaves the webview with that scheme). Elevation tiles go out through the http plugin like every tile; `https://*/*` in `capabilities/default.json` already covers `s3.amazonaws.com`. Tauri serves `.mjs` as `text/javascript` (tauri-utils), as a module worker needs.
- **Data**: 3D at 60° pitch shows far more ground than 2D: roughly 2-4 times the tiles per screen, plus one elevation tile (~50-150 kB) per terrain tile. All of it is kept, so a road seen once costs nothing the next time.

---

6. External services and their limits

6.1 Services used

All requests go through `@tauri-apps/plugin-http` from Rust (no CORS; the webview's fetch cannot set a User-Agent) with `User-Agent: NestourGPS/<version> (https://github.com/eriktjepkema/ha-tourist-explorer)` (the version from `tauri.conf.json` via `__APP_VERSION__`) (`USER_AGENT` and `http` in `backend.ts`); Wikimedia answers 403 without contact information.

Service	Endpoint(s)	Used for	Limits and timeouts
Overpass	**NEW** the person's own server first (`keys.json` `overpass`, 15 s cap), then `overpass-api.de`,	Places, road tiles, railways,	Through `overpassPost` (below).

Service	Endpoint(s)	Used for	Limits and timeouts
	`overpass.kumi.systems`, `overpass.private.coffee` (`OVERPASS_URLS`, `discovery.ts`), tried in order	fuel, motorway points	Server/client timeouts: places `timeout:60`/75 s; roads 15/20 s; rails 25/30 s; fuel 15/20 s; motorway points 75 s, snap 30 s. Public fair use
OpenTripMap	`api.opentripmap.com/0.1/en/places/radius`, `/xid/`	Places + rating, fuel, descriptions	Key; 20 s; free plan non-commercial, 5000 req/day
Wikipedia	`<lang>.wikipedia.org/w/api.php`, `/api/rest_v1/page/summary/`	Geosearch fallback, langlinks, summaries	20 s
Wikidata	`www.wikidata.org/w/api.php` (sitelinks, 50 per call), `Special:EntityData`, `query.wikidata.org/sparql`	Fame, heritage, class, P18 photo, website	20 s; SPARQL 15 s (failure = partial result)
Wikimedia Commons	`commons.wikimedia.org/w/api.php` (`imageinfo`, `iiurlwidth=640`)	Place photo and credit	15 s
Nominatim	`nominatim.openstreetmap.org/search`, `/reverse`	Geocoding, place by name, street/town, region/country, stop names, the countries and regions of routes, plans and trips (`item_places`)	**One request at a time, ≥ 1.1 s apart** (`nominatim()`, `backend.ts`), 15 s, `accept-language` = app language
OSRM	`router.project-osrm.org` (car), `routing.openstreetmap.de/routed-foot`, `/routed-bike`	Turn by turn, ETA, timetable, simulator, detours	20-30 s; public, light use; max 60 points
TomTom	`api.tomtom.com/traffic/services/5/incidentDetails`; `/routing/1/calculateRoute` (traffic, `departAt`, `instructionsType=coded` for	night`	Incidents, routing with traffic, guidance, flow overlay, TomTom maps; since

Service	Endpoint(s)	Used for	Limits and timeouts
	guidance); `/traffic/map/4/tile/flow/relative0`; `/map/1/tile/basic/main		0.9.5 `/search/2/nearbySearch` (`categorySet=7311`, radius ≤ 50 km, 50 results: fuel fallback) and `/search/2/reverseGeocode` (`returnSpeedLimit`: speed-limit fallback, 5.13)
Open-Meteo	`api.open-meteo.com/v1/forecast?current=…`	Weather in the logbook	15 s, no key
French fuel prices	`data.economie.gouv.fr/…/prix-des-carburants…/records`	Average price within 10 km; since 0.9.5 also the stations themselves within the search radius (`fuelStationsUrl`, ≤ 50, `within_distance(geom, …)`): fuel fallback	20 s, France only
GeoNames	`download.geonames.org/export/dump/cities15000.zip`	Country/city lists	120 s, once; unzipped in JS (`unzipText`)
NEW Nextcloud	`<url>/remote.php/dav/…` (CardDAV PROPFIND and REPORT, `contacts.ts fetchNextcloud`), Basic auth with an app password	Contacts as destinations	The person's own server; https only
NEW TomTom web planner	`plan.tomtom.com/en/route/plan?r=(…)` (a link opened outside the app, `tomtomWebUrl`)	The « TomTom » route pill	No key, no request from the app; unofficial link format, can change without notice

Service	Endpoint(s)	Used for	Limits and timeouts
NEW Mappy route page	`fr.mappy.com/itineraire#/voiture/…/car` (a link opened outside the app, `mappyUrl`)	The « Mappy » route pill	No key, no request from the app; unofficial link format; Mappy's terms forbid automated use
Google	`accounts.google.com` (browser), `oauth2.googleapis.com/token`, `people.googleapis.com/v1/people/me/connections`	Contacts as destinations	Own Desktop OAuth client, PKCE, loopback; 20/30 s
AI	see 6.3	Facts, translations	60 s; own-address providers 180 s
Tiles (webview, not the http plugin)	`tile.openstreetmap.org` (`card.setConfig` in `main.ts`), Esri `server.arcgisonline.com/…/World_Imagery`, `{s}.tile.opentopomap.org`, swisstopo `wmts.geo.admin.ch/1.0.0/ch.swisstopo.pixelkarte-farbe/…/3857` and `ch.swisstopo.swisstlm3d-wanderwege`, TomTom, IGN `data.geopf.fr/wmts` (keyless plan) and **NEW** `data.geopf.fr/private/wmts` (SCAN 25, the person's key), **NEW** Mapy.com `api.mapy.com/v1/maptiles/…` (the person's key)	Base maps and overlays (`MAP_STYLES`, `card.ts`)	Public servers; swisstopo offered in or near Switzerland
Mapzen terrain tiles (0.11.0)	`s3.amazonaws.com/elevation-tiles-prod/terrarium/{z}/{x}/{y}.png`	Elevation for the 3D map (5.21), through `tile_get`	Public open data on AWS, no key; z ≤ 15

The Overpass queue (0.9.5, `overpassPost(http, query, timeoutS, urgent)`, `engine/discovery.ts`). On 2026-10-30 public Overpass was down all evening, and sight search, road tiles, fuel and the train check each walked every mirror again and again: ~110 failed calls of 40-75 s that only added to the overload. Every Overpass request now goes through one slot, app-wide (`overpassSlot`): one request in flight, the others wait; urgent ones (road tiles, fuel, motorway snap:

small and wanted now) jump to the front of the queue. Inside the slot the mirrors are tried one at a time, in order; a mirror that fails rests 5 minutes (`MIRROR_REST_MS`) and is skipped meanwhile (« (resting) » in the error), a success clears its rest. When every mirror is resting the call fails at once (« OpenStreetMap (Overpass) unavailable: … ») instead of waiting. `overpassReset` (tests) clears the rests and the own server.

NEW **Own server** (0.13.0, `setOwnOverpass`). When set, it is tried first inside the slot with the call's timeout capped at `OWN_TIMEOUT_S` = 15 s (away from home without Tailscale it fails fast), and rests 5 min on failure like a mirror. An empty answer from it (outside the region it holds) goes on to the public mirrors; if every one of them fails, the empty answer stands (`shortcut:` comment in `overpassPost`: an empty spot inside its region costs one public call).

Fuel stations (0.9.5, `fetchFuel`, `discovery.ts`; `fuel` in `backend.ts`). OSM (Overpass, brand and hours) and OpenTripMap (with a key) in parallel; only when OSM failed or found none: TomTom's search (category 7311, with a key) and, in France or where the country is not known yet, the prix-carburants stations, also in parallel. All lists are merged by `mergeFuel` (a station within 60 m of one already listed is the same). The answer distinguishes « none around » from « could not ask »: it throws « Station search unavailable » only when nothing was found and every source that lists every station (OSM, TomTom, prix-carburants; OTM does not count) failed or was not available. The card then shows `fuelUnavailable` with a « try again » button (`stations: "failed"`) instead of an empty list; `DATA_CAP` keeps its own notice. Each search logs `fuel` with the count per source or its error (9). A non-empty result is cached 1 h.

The TomTom flow tiles are redrawn in the colour-blind palette by `RecolouredTiles` (`card.ts`), which calls `recolourTraffic` (`card/src/colours.ts`, tested in `card/test/colours.test.ts`) on each tile's pixels in one pass, by hue band (0.10.4): green (70-170°) → `#0072B2`, orange (20-70°) → `#E69F00`, red (330-20°) → `#D55E00`, and the darkest red (brightest channel < 150: stopped, closed) → `#D55E00` with black diagonal stripes `((x+y)>>2)&1` (4 px bands), so a jam is not told by colour alone. Greys, white, the dark outlines (saturation < 0.35 or max < 40) and transparent pixels stay. The bands were not sampled from real tiles: they follow TomTom's documented relative0 legend (green, orange, red, dark red), and the test uses colours close to it. The colour-blind layer asks `&thickness=10` (TomTom's default, as far as known: a `shortcut:` comment in `applyTrafficLayer`; raise it if the stripes blur). TomTom's tiles allow CORS, so a canvas may read them.

6.2 The http allow-list

`app/src-tauri/capabilities/default.json`: `http:default` with an `allow` list of URL patterns. A host not on it fails inside the app with « url not allowed on the configured scope », **although every Node test passes** (the tests mock the plugin). Lesson 0.8.4: `api.tomtom.com` was missing, so every TomTom feature failed on the phone while the live tests outside the app worked. Rule: a new API host goes into this file, and is tried once in the built app.

NEW 0.13.0 adds `http://*:*/api/interpreter` and `https://*:*/api/interpreter` (an own Overpass server, cleartext on the LAN), `https://data.geopf.fr/*` (SCAN 25), **NEW** `https://api.mapy.com/*` (Mapy.com tiles, for the map cache) and `https://*:*/*remote.php/dav/*` (Nextcloud CardDAV on any port).

Since 0.9.2 the list ends with `"https://*/*"` (for « custom » AI at any address) and contains `http://*:11434/*` (Ollama on any host, cleartext). With the wildcard every https host is allowed and the per-host entries above it no longer restrict anything; Mistral, DeepSeek, OpenRouter and Groq are reachable only through it. Kept on purpose (0.10.1): a « custom » AI address is typed by the person at run time, and a Tauri scope cannot be widened after the build.

Other capabilities: `opener:allow-open-path` limited to `$RESOURCE/docs/*` (the HA-era help PDFs bundled from `custom_components/tourist_explorer/docs/*.pdf`), and the defaults of core, opener, store, notification, dialog, trip-location. Since 0.10.1 the app's Help tab no longer links those PDFs (it shows a short « User guide » section, `userGuideTitle`/`userGuideBody`); they are still bundled (`tauri.conf.json resources`) until the new guides have a home.

6.3 AI providers

`engine/ai.ts`. One provider in use at a time, each provider's setup kept in `keys.json` `ais`.

Provider	Request	Default model
`claude`	`POST https://api.anthropic.com/v1/messages`, headers `x-api-key`, `anthropic-version: 2023-06-01`, `anthropic-dangerous-direct-browser-access: true` (the request carries the webview's Origin); body `{model, max_tokens: 4096, messages}`	`claude-haiku-5-5`
`chatgpt`, `gemini`, `mistral`, `deepseek`, `openrouter`, `groq`	`POST <base>/chat/completions`, `Authorization: Bearer <key>`, body `{model, messages}` (bases in `OPENAI_STYLE`, `ai.ts`; Gemini through its OpenAI endpoint)	gpt-5-mini, gemini-2.5-flash, mistral-small-latest, deepseek-chat, openrouter/auto, llama-3.3-70b-versatile
`ollama`	`<url or http://localhost:11434>/v1/chat/completions`, key optional	llama3.2

Provider	Request	Default model
custom	<url>/chat/completions (a trailing /chat/completions is accepted), key optional, model required	–

max_tokens is 4096 because newer Claude models think first: 1024 left no text. <think>…</think> is stripped (reasoning models). An empty answer cut by the length limit throws a clear error. The facts prompt (facts.ts) asks for exactly three verifiable facts in the target language, or one line saying it is not sure.

6.4 Keys in errors and logs

- Error messages: call() replaces the value of key=, apikey=, api_key=, access_token=, token= in any URL inside the message with … (call, backend.ts). The Tauri plugins reject with plain strings that contain the full URL; without this the TomTom key appeared on screen (0.8.4).
- Debug log: requests are logged by host and the first 80 characters of the path, never the query; an error text is cut at the first ? (http, backend.ts).
- Debug export: only which keys exist (keys: {opentripmap: true, …, ai: "claude/model"}), never their values.
- traffic_tiles hands the card tile URLs that carry the TomTom key (kept in the card's memory; the webview requests them).

6.5 Data limit and roaming

engine/datacap.ts + dataTick, dataCap, dataStatus in backend.ts. **NEW** The limit in MB is limitMb(settings): data_limit_money / data_price × 1024 when both are set (0.13.0), else data_limit_mb. Every 30 s on Android, dataTick reads the app's own byte total since boot (dataUsed: TrafficStats.getUidRxBytes + getUidTxBytes) and the network (networkState: cellular transport; roaming = cellular without NET_CAPABILITY_NOT_ROAMING, Android 9+), and adds the bytes since the last reading to the month's mobile and/or roaming counter (a reboot resets the total: nothing is added). A gap between readings goes to the network of the next reading (shortcut: at the top of datacap.ts). State: off (no limit), ok, warn (≥ 80 %), over (≥ 100 %), and only on the counted network (data_scope: roaming or all mobile); on Wi-Fi nothing is held back. At warn, pictures stop; at over, the capped services (search, facts, picture, fuel_price) throw DATA_CAP and the card shows a notice once per crossing (tourist-explorer-datacap event). Map, route, turn by turn and speed signs keep working. data_more adds 50 MB to the month. A trip also stores its own data_bytes (difference of the total from start to end).

---

7. Android specifics

Topic	How
Permissions (plugin manifest)	`ACCESS_FINE_LOCATION`, `ACCESS_COARSE_LOCATION`, `FOREGROUND_SERVICE`, `FOREGROUND_SERVICE_LOCATION`, `POST_NOTIFICATIONS`, `WAKE_LOCK`, `ACCESS_NETWORK_STATE`, `ACTIVITY_RECOGNITION` (+ the GMS one), **NEW** `READ_CONTACTS` (0.13.0); app manifest adds `INTERNET`. Runtime: location + activity at the first `watch`, notifications when the service starts (Android 13+), **NEW** contacts at the first `phone_contacts` (once `contacts_phone` is on). « While using the app » location is enough: the foreground service keeps the process in use
Foreground service	`TripLocationService.kt`: type `location`, channel « Trip » / « Trajet » (named in the app's language, low importance), ongoing notification « Trip running » / « Trajet en cours » + trip name, tap opens the app, `START_STICKY`. Started by `background {on: true}` at trip start/resume (not for test trips), stopped at night and at the end
Wake lock	One `PARTIAL_WAKE_LOCK` (`TouristExplorer:trip`) for the service's life: the trip engine is JavaScript in the webview and must keep running with the screen off
Battery optimisation	Not handled: the app does not ask to be exempted (`REQUEST_IGNORE_BATTERY_OPTIMIZATIONS`). Samsung's power saving held back background network on the road test; set « Unrestricted » battery use by hand on test devices
Command names	Tauri calls a Kotlin `@Command` by the **camelCase** form of the JS name: `keep_screen` → `keepScreen`. Snake_case Kotlin methods are never found, silently (0.4.1 fixed screen, share, print, data). `build.rs` lists the snake_case names for the permission files (`permissions/autogenerated/commands/*.toml`, `default.toml`)
Commands	`watch`, `unwatch`, `background`, `background_state`, `ask_background`, `keep_screen`, `open_intent`, `beep`, `speak` (0.9.5), **NEW** `phone_contacts` (0.13.0: `ContactsContract StructuredPostal` rows, at most 5000; null when the permission is refused), `save_files`, `save_pdf`, `share_files`, `print_html`, `data_used`, `network_state`, plus Tauri's `register_listener` / `remove_listener` (`build.rs` `COMMANDS`)
Spoken guidance	`speak {text, lang}` (`TripLocationPlugin.kt` `speak`/`say`, JS `android.ts` `speak`, from the card's `tourist-explorer-speak` event, 5.19): Android `TextToSpeech`, started once on the first sentence (a call while it starts is rejected; a failed start is retried next time).

Topic	How
	Audio attributes `USAGE_ASSISTANCE_NAVIGATION_GUIDANCE` / `CONTENT_TYPE_SPEECH`; a transient « may duck » focus (API 26+) lowers the radio and is released on done or error. Language = `Locale.forLanguageTag(lang)` when the engine has it, else the engine's own. `QUEUE_FLUSH`: a newer instruction replaces one not yet said. Android 11+ package visibility: the plugin manifest declares `<queries><intent><action android:name="android.intent.action.TTS_SERVICE"/></intent></queries>`, without which no speech engine is found and nothing is said
Listeners after a reload	The plugin's event listeners outlive the page: after `location.reload()` (a new app language) Kotlin kept sending fixes and satellites to the dead page's callbacks, 12,658 « Couldn't find callback id » warnings in one evening (2026-10-30). Since 0.9.5 `android.ts` `listen` keeps each listener's {event, channelId} in `sessionStorage` (`trip-location-listeners`), and the next page's first `watchPositions` calls `dropOldListeners`: `plugin:trip-location|remove_listener` for each, then logs `listeners_removed` {n}
Vehicle markers	`app/public/vehicles/marcopolo-marker.png` (van), `gs1200-marker.png` (motorbike), `healey-marker.png` (classic), served at `vehicles/…` (5.19)
White screen after `adb install -r` (Lenovo/MediaTek)	The system may refuse to start the app's WebView renderer: logcat shows « process is bad » / « suppress to start service », and the app opens white. Settings → Apps → Nestour GPS → **Force stop** clears it; `adb shell am force-stop` and a reboot did not reliably
Exports	`saveFiles` → `MediaStore.Downloads` with `RELATIVE_PATH = Download/<export_folder>` (Android 10+); answers the name Android gave (`x (1).gpx`). Android 9 and older: the app's external files folder (`shortcut:`). MIME type from the extension so Android keeps it
Share	Files written to `cache/exports`, offered through the app's `FileProvider` (`${applicationId}.fileprovider`) in a chooser
PDF	`savePdf`: an off-screen `WebView` loads the HTML, its print adapter is written to an A4 300 dpi PDF without the print screen by `android.print.PdfWriter` (in that package to reach package-private callbacks, `shortcut:`), then copied to Downloads. On failure the app falls back to `printHtml` (Android's print screen, « Save as PDF »)
Beep / ducking	`ToneGenerator` on the notification stream (no audio focus: the radio plays on); two beeps (too fast) or one (reminder).

Topic	How
	With `beep_duck`, a transient « may duck » focus first, released after 700 ms
Intents	`openIntent` parses `intent:` links (TomTom GO), strips component/selector, falls back to a chooser
Keep screen	`FLAG_KEEP_SCREEN_ON` and the view's own flag (some skins drop the window's), only while in front
Network state	`networkState` for the data limit (6.5); a `net` debug event on every change
Satellites	`GnssStatus.Callback`: constellation, PRN, elevation, azimuth, C/N0, used in fix, at most 1/s; shown on the Instruments tab

---

8. Internationalisation

67 languages: English and French in `card/src/i18n.ts` (objects `en`, `fr`, and `appEn`/`appFr` for the app's own wording), five built-in packs `card/src/lang/{de,it,es,nl,fi}.ts` that HA's card also carries, and 60 packs in `card/src/lang/extra/*.ts` that only the app has. `app/src/langs.ts` loads the one in use at start (`import.meta.glob`, lazy) and registers it (`registerLang`). A pack is a `LangPack` `{ui: Strings, app: Partial<Strings>, gps: GpsStrings, help: HelpSection[], trip: Record<string,string>}` (`LangPack` in `i18n.ts`): the card's strings, the app's overrides, the GPS pages (`gps-i18n.ts`), the Help tab (`help.ts`) and the trip notifications (keys of `TEXT` in `trips.ts`). Device codes are aliased (`nb`/`nn` → `no`, `iw` → `he`, `fil` → `tl`, `in` → `id`); `ar`, `he`, `fa`, `ur` mirror the panel (RTL), not the map; `NATIVE` names each language in its own words. Descriptions and facts have their own language list (`info.LANGUAGES`, 63) and the `native` option.

Adding a UI string:

1. Add the key to `en` in `i18n.ts` (this defines the `Strings` type) and to `fr`. App-only wording goes in `appEn`/`appFr`.
2. Every pack's `ui` is typed `Strings`, so `tsc` (card `npm run build`, app `npm run build`) now fails until all 65 other files have the key. The owner's routine: a small script (kept in the session scratchpad, not in the repository) copies the English text into every pack, then the texts are translated in batches.
3. Placeholders `{name}` must stay identical: the i18n test checks it.
4. `cd card && npx vitest run test/i18n.test.ts`: the 7 built-in languages keep every placeholder; every extra pack has the same placeholders in `ui` and `gps`, 8 compass points, the help's sections (by icon), every notification key (`ahead, go, skip, stop, visited, here, next_stretch, stretch_to, stretch_last, pause_over, break`), no stray spaces in `gps.ago`; every language has a native name.

Notification texts inside `trips.ts` exist in en/fr; other languages come from the pack's `trip`. The foreground service's title and channel name (`running`, `channel`) come from the same texts, in Setup's app language, passed to `background` by `keepAlive` (0.10.1; other languages fall back to English unless their `trip` pack has them).

Units (`card/src/units.ts`): everything is stored and computed metric; `metric`, `us`, `uk` only change display and exports; `""` derives from the locale (en-US → US, en-GB → UK).

Place types (0.11.2): `kindNames` in `Strings` maps 72 type codes (OSM values, app-made `motorway_*`/`contact`) to a map-legend noun in every language; `kindName(kind, t)` (`i18n.ts`) takes an OpenTripMap list's first kind, folds OTM plurals via `KIND_ALIAS`, hides `yes` and makes unknown codes readable (`tourist_trap` → « Tourist trap »). Used by the Info card, search results and the favourite editor; stored data and grouping stay on codes; the i18n test checks every pack has en's codes.

---

9. The debug log

`app/src/debuglog.ts`. On by default during the beta (Setup → Your data → Debug, setting `debug_log`). Events live in memory (last 20,000, about 5 hours at one fix a second, ~2 MB), are saved to `debug.json` every minute, and come back at the next start (an app killed by Android keeps what it saved). Console warnings and errors, uncaught errors and unhandled rejections are captured too.

k	**Fields**	**From**
`app_start`	–	`debugStart`
`fix`	ft (GPS time, s), lat, lon, v (km/h), acc	Every Android fix (`watchPositions`, `android.ts`)
`stale`	age (s) or t	Fix ≥ 600 s old, or older than the last handled (`onPosition`, `trips.ts`)
`lag`	s	Handled > 5 s after the fix
`slow_pos`	ms	`onPosition` took > 1 s
`search`	ms, found	Trip search finished
`road`	key, limit, est	New road in the road log
`http`	host, path, status or error, ms	Every request
`service`	on	Foreground service on/off
`net`	mobile, roaming	Network change
`visible`	on	App to front / background

k	Fields	From
`warn`, `error`	msg (≤ 300), repeated?	Console; the same message at most once a minute (below)
`crash`	msg, at	Uncaught error / rejection
`trip_start`	id, name, mode, reason	`tripStart`: `reason` = the caller's, else simulation, plan, route, past trip, record, destination, button
`trip_end`	id, reason	`tripEnd` (button, at interruption, during the night, or the caller's), a running trip replaced (new trip), a 🧪 trip ended at app start (`startAll`, test trip at app start)
`trip_day_end`	id	🌙 End of day
`trip_dest`	id, name	« Guide me to… » (`trip_dest`)
`skip`	id, name, via (banner, card, trip)	A waypoint skipped by hand (`trip_skip`, 0.10.3)
`unskip`	id, name	« Undo » after a skip (`trip_unskip`, 0.10.3)
`guide_route`	by (tomtom, osrm), heading, points, steps, km; or by, error	Every turn-by-turn route asked; a TomTom HTTP error before the OSRM fallback
`guide`	shown: true, text (≤ 60), tab, map (0.10.4); or shown: false, why; or (0.10.4, once per route) shown: false, why `no next step`, off, along (m), steps	Card note: the turn-by-turn banner shown / hidden and why (no trip, night, guide off, mode …, route done); `map` false = the map was hidden (Instruments, Setup, Help)
`speak`	text (≤ 80)	Card note: each spoken line
`gps_source`	to, trip	Card note: trip start switched the map to the live GPS (5.1)
`no_fix`	trip, last	Card note: no fix 30 s after a trip start (5.1)
`fuel`	found, osm, otm, tomtom, fr (count or error per source asked); or error	Every fuel search not served from the cache (6.1)
`road_held`	kept, limit, seen, seen_limit, speed	A sudden drop of the limit held back (0.12.1, 5.13)

k	Fields	From
here	id, name	« 📍 I'm here » marked a waypoint or the destination (trip_here)
road_src	src (osm, tomtom, none), was; or src: tomtom, error	The speed-limit source changed, or a TomTom lookup failed (5.13)
listeners_removed	n	Old plugin listeners removed after a reload (7)

Every event has t (wall clock, seconds). The card-side events come through debug_note (4), limited to the names in DEBUG_NOTES (backend.ts): guide, no_fix, speak, gps_source; any other k throws « unknown debug note ».

Repeated warnings (0.9.5, repeats, debuglog.ts): one warning flooding the log (the 12,658 « callback id » lines, section 7) had filled the 20,000 events and pushed the evening's drive out. The same warn/error text, digits ignored (« callback id 4021 » = « 4022 »), is logged at most once per 60 s (REPEAT_MS); the next one logged carries repeated: how many were left out since. The console itself still gets every one; at most 500 texts are tracked.

Export (debug_export: debugExport, backend.ts), saved as tourist-explorer-debug-<stamp>.json:

```
{ format: "tourist-explorer-debug", version: 1, app, beta_until, exported,
  device: {ua, screen, lang, tz}, settings, keys: {opentripmap, tomtom, ai, google},
  network: {mobile, roaming, month}, trip: {…the running or last real trip…, track: [...]}, events: [...] }
```

trip is the whole trip as stored (log, legs, roads, towns, traffic, weather, working state; not slimmed), the running one or else the most recent non-test one. Before 0.9.5 it was the trip list's summary (id, name, dates, stop count).

Analysing a road test. Pull the export (2.4) and look, in this order:

1. **Recording gaps:** sort fix events by t; a gap > 10 s while service was on and visible false points at Android (the service was frozen or the GPS stopped). Compare with trip.track timestamps: a gap only in the track means the engine dropped or delayed points.
2. **Lag:** t − ft per fix, lag and slow_pos events: anything near the 0.9.0 ~160 s means work is back under the lock.
3. **Network:** group http by host: error counts and ms; Overpass mirrors timing out explain partial searches and missing road tiles (then road events stop and, without a TomTom key, speed signs go blank; with one, road_src says tomtom). fuel events show which station sources answered.
4. **Road log:** road events vs trip.roads; a long single entry with no road events means no tile loaded there.
5. **Stops and visits:** trip.log against the track: a visit on a main road should no longer happen.
6. **Data:** network.month vs trip.data_bytes.

Example (Python, any machine):

```
import json; d = json.load(open("tourist-explorer-debug.json", encoding="utf-8"))
fx = [e for e in d["events"] if e["k"] == "fix"]
gaps = [(a["t"], b["t"] - a["t"]) for a, b in zip(fx, fx[1:]) if b["t"] - a["t"] > 10]
print(len(fx), "fixes;", len(gaps), "gaps > 10 s;", max((g for _, g in gaps), default=0), "s longest")
```

---

10. Home Assistant edition and deploy

10.1 Differences

	Nestour GPS (app)	**Tourist Explorer (HA)**
UI	Same card, `standalone: true`, extra tabs and Setup groups	Same card, plus `tourist-gps-card` views
Services	`backend.ts`, 81 services (0.13.0), in process	`__init__.py`, 36 response-only services over HA's websocket (`trip_unskip`; `route_save` with `pauses`/`free`)
Users	One (`me`)	Per HA user
Position	This device (Android GPS / webview; **NEW** on Linux the webview's geolocation), USB GPS not yet	HA entities: Companion app tracker, Android Auto, mesh, backup tracker; browser GPS and USB GPS on open pages
Trip runtime	`trips.ts` on device fixes; background work since 0.9.1	`tripsvc.py` on tracker state changes; the place search runs as a background task since 0.10.1, reverse geocoding and the train check are still awaited under its lock; no road log, no missed/seen waypoints, no main-road rule, no pauses, free time or nights
Notifications	Local, plugin-notification	`notify.mobile_app_*` with actions
Logbook calendars	None	`calendar.create_event` per finished stop
AI	Own key, 9 providers	HA `ai_task` entity
Keys	`keys.json` on the device	Config entry options
Contacts	Google (own OAuth client, loopback), **NEW** Nextcloud CardDAV (app password in	Nextcloud CardDAV + Google via HA's application credentials

	Nestour GPS (app)	Tourist Explorer (HA)
	keys.json) and the phone's address book (Android)	
Motorway points	Not wired (11.3)	motorway_points, motorway_snap
TomTom, timetable, turn by turn, data limit, debug log, photos, speed signs	Yes	No
Tiles	Direct from the tile servers	HA's map_tiles proxy
Storage	Tauri stores (3.3)	.storage/tourist_explorer.*

10.2 Deploy (HA only)

```
HOMELAB=/c/Users/erikt/ansible-homelab ./deploy/deploy.sh -t dev|prod|emma|jonas|demo|… [-o deploy/boxes/<box>.json]
./deploy/deploy-dev.sh      # = deploy.sh -t dev
```

deploy.sh builds the card, tars custom_components/tourist_explorer, pushes it (and the options JSON) with ansible-homelab's ha-env.sh (rpush), then runs deploy/install.py on the box through inject.sh, which stops HA cleanly, runs the script, starts HA **even on failure**, waits and checks. install.py replaces the component, adds or merges the « Tourist » dashboard and grafts the config entry; idempotent, with backups. HOMELAB defaults to $HOME/dev/ansible-homelab; on this laptop the clone is ~/ansible-homelab, so set it. Each run is one HA stop window (about a minute) on that box.

- **No locking in the scripts:** never run two deploys to the same box at once, and run fleet deploys in the background so a killed foreground run cannot leave HA stopped (it once left Nernier down 11 hours).
- **Ledger rule** (after any live homelab change): append an entry to INSTALLER-PENDING-CHANGES.md in ansible-homelab, commit, push, git -C /root/ansible-homelab pull --ff-only on the Proxmox host, then refresh the NFS mirror with /usr/local/lib/homelab-cron/nfs-clone-mirror.sh --now.
- deploy/publish-docs.sh copies docs/ into ansible-homelab docs/tourist-explorer/ for the Outline mirror.

---

11. Known limits, shortcut: debts and backlog

11.1 Known limits

- Public Overpass, Nominatim, OSRM and tile servers: fine for family and testers, not for a product (their terms forbid heavy or commercial use). Own

servers or paid providers before any sale (`docs/LICENSING-PROPOSAL.md` §5).

- No battery-optimisation exemption; on Samsung and similar the background service can still be throttled. The 0.9.0 phone recorded 6 minutes, then nothing until the end.
- No offline area download: map tiles (2D and 3D) are kept once seen (IndexedDB, `tile_cache_mb`, 3.3), but nothing fetches an area ahead of a trip.
- The 3D map (5.21) shows no speed signs, speed buttons, turn-by-turn banner, ⏭ or « Search this area », and costs roughly 2-4 times the tiles of 2D; it needs WebGL.
- No USB GPS feed in the app yet; satellites and DOP come from Android's `GnssStatus` only.
- **NEW** The Linux AppImage (2.6) builds, but has **not yet been tried on the team's Omarchy laptop**: install, webview, geolocation and exports there are untested. Position on a laptop comes from the webview's geolocation only.
- **NEW** The « TomTom » route pill uses plan.tomtom.com's own link format, read off the site on 2026-11-02, not a published API: TomTom can change it without notice. Car and motorbike only.
- **NEW** The « Mappy » pill goes to one point only (Mappy's links carry no stops) and opens on Mappy's last-used mode; its link format is unofficial and can change without notice.
- **NEW** The own Overpass server (the team's CT 138) answers on the LAN and the tailnet only; elsewhere the public mirrors answer, after one 15 s wait per 5 minutes.
- **NEW** Android 17 (target SDK 37) closes the LAN to apps without `ACCESS_LOCAL_NETWORK` (« Nearby devices »): requests to the own Overpass server time out (« Request canceled », seen on the team's S24 Ultra 2026-11-03). The plugin declares it and asks (alias `lan`) with location in `watch` and through `ask_lan` when the address is saved or tested; refused, `overpass_test` says so. Android 16 and older: nothing asked.
- Trip statistics drop segments faster than 300 km/h as GPS jumps (`MAX_PLAUSIBLE_KMH`, `gpslog.ts`); after a recording gap with a stale first fix the catch-up segment is lost from the distance (road-test finding B). The 0.9.1 lock fix removes the cause.
- `getLastKnownLocation` at start may be old; fixes ≥ 10 min old are dropped, younger stale ones are kept.
- TomTom requires the person's own key (Traffic and Routing APIs enabled for it).
- *(Eased in 0.9.5.)* An Overpass outage used to blank the speed signs, empty the fuel list (« no station ») and multiply the load by retrying every mirror per request. Now the queue and mirror rests (6.1) cap the load, and TomTom (with a key) and prix-carburants (France) stand in; without a TomTom key, signs still go blank outside cached OSM tiles.
- *(Fixed in 0.9.5.)* A trip started on « Manual location » recorded nothing (5.1); a recorded trip showed no turn by turn without saying why (5.19); the place popup jumped with every fix while driving (5.19); the debug export carried only the trip's summary (9); old plugin listeners flooded the log after a language change (7).

- TomTom speed limits come only through Reverse Geocoding (no snap-to-road or speed-limit API is used): the limit of the road TomTom finds at one point, at most one call per 10 s and 100 m, kept 12 s or 50 m. Above about 80 km/h the shown limit can look stale between calls (a change of limit shows up to ~10 s late).
- Spoken guidance depends on the device's speech engine and its voices: a language the engine lacks is spoken with the engine's default voice.

11.2 `shortcut:` debts in the code

Where	Debt	Upgrade when
`app/src/main.ts`, top (help PDFs)	`openPath` for help PDFs untested on Android	Check the help PDFs on Android
`app/src/main.ts`, `tourist-explorer-files` listener	On the desktop « Share this trip » saves the files to Download/<export folder> (no share sheet)	If the desktop becomes a target
`app/src/main.ts`, above `card.setConfig`	Public tile servers (kept on the device by `tile_get` since 0.10)	Own tiles before selling
`app/src/main.ts`, after `card.setConfig`	No USB GPS feed	Step 3b
`app/src/backend.ts`, the track store's `remove`	A deleted trip's track file stays on disk, emptied	When the fs plugin is added
`app/src/backend.ts` `guideRoute`	Turn by turn on the public OSRM server	Own server in Phase D
`app/src/engine/datacap.ts`, top	A gap between readings is counted on the next reading's network	If the counts are off by more than a few MB
`app/src/engine/discovery.ts` `overpassPost` **NEW**	An empty answer from the own Overpass server also asks the public mirrors: an empty spot inside its region costs one public call	If the public load matters
`app/src/engine/contacts.ts`, the CardDAV reader **NEW**	Regex, not an XML parser (no DOMParser in node); any namespace prefix accepted, no	If a server's XML breaks it

Where	Debt	Upgrade when
	namespace URI check	
`app/src/engine/trip.ts` `setLeg`	Manual legs only in the app, not in HA	If HA users ask
`card/src/card.ts`, above `MAP_STYLES`	Public tile servers	Own tiles before selling
`card/src/card.ts` `applyTrafficLayer`	Colour-blind flow tiles ask `&thickness=10`, unverified on the phone (0.10.4)	Raise to 14-16 if the stripes blur
`card/src/help.ts`	Other languages carry only the app's help wording	–
`TripLocationPlugin.kt` `saveToDownloads`	Android 9 and older save exports in the app's own folder	If an old device appears
`PdfWriter.kt`, top	Relies on package-private print callbacks	If an Android version refuses it (fallback: print screen)
`custom_components/…/__init__.py` `google_accounts`	HA: an unclaimed Google account is offered to every user	–

11.3 Inconsistencies found while writing this reference

1. *(Fixed since: the timetable writes `trip.planned_at`, `tripStart` in `backend.ts`.)* **`trip.planned` overwritten:** after `trip_start` from a plan, a route or a past trip, the background `plan_schedule` result replaced `trip.planned` (the waypoint id **list**) with a map `{poiId: arrival, dest: …}`, breaking `waypoints()`, `planStop()`, `skipBehind()` and the card's `pendingStops()`. Now the map goes to `planned_at` (read by `trip_eta`, kept by `importTrip`).
2. *(Fixed in 0.10.1: both services ported, `engine/motorway.ts` `parse`.)* **Motorway points not wired in the app:** the card's motorway picker (`motorwayPicker`) and the map-tap `motorway_snap` failed with « not available in the app yet ».
3. *(Accepted in 0.10.1, see 6.2.)* **The http allow-list is open:** `"https://*/*"` (0.9.2) makes the per-host entries meaningless (6.2), and `http://*:11434/*` allows cleartext to any host on that port.
4. *(Fixed in 0.10.1: the comment now says only met incidents are kept.)* **TomTom incidents stored:** `tomtom.ts` said incidents are « shown live, never stored (TomTom's terms) », but the traffic log stores them in `trip.traffic` and in exports.
5. *(Search moved to a background task in 0.10.1; the rest stands at 0.11.2.)* **HA trip runtime lags behind:** `tripsvc.py` `on_position` still awaits the train check, the stop's reverse geocoding and the calendar entry under its per-trip

lock (the 0.9.1 root cause) and has no missed waypoints, seen places, main-road rule or road log.
6. *(Fixed in 0.10.1.)* **Lossy trip import:** `importTrip` drops `missed`, `pause_s`, `pause_skipped`, `pause` on entries and `pauses`, `free_plan`, `route_id` on trips; pause notes and « missed » marks disappear on the other device.
7. *(Fixed in 0.10.1.)* **Traffic work outside `bg()`:** `trips.ts` used `void this.track(…)`, so `idle()` did not wait for it.
8. **Stale comments** (still at 0.11.2): the header of `data.ts` says contacts are not in the app (Google contacts are); in `DEFAULT_SETTINGS` the comment on `approach_km` also carries `auto_search`'s, and `search_every`'s carries `route_lines'`; `Library.exportData` has two doc comments; in `backend.ts` a stray doc comment (« at_interruption ») sits above the multi-day section, after `tripEnd`.
9. *(Fixed: `LogEntry.kind` in `card/src/logbook.ts` has `"free".`)* **Types:** `LogEntry.kind` lacked `"free"` although `ICON` had it.
10. *(Fixed in 0.10.1 except the crate description.)* **Names and versions:** User-Agent `TouristExplorer-App/0.1` (`USER_AGENT`, `backend.ts`); `app/package.json` and both Cargo crates at `0.1.0`, the main crate described as « Tourist Explorer, standalone edition ».
11. *(Fixed in 0.10.1.)* **Foreground notification language** followed `navigator.language`, not Setup's app language, and had only FR/EN (`android.ts`).
12. *(Fixed in 0.10.1, with `debug_clear`.)* **After beta expiry** `debug_export` was not in `AFTER_EXPIRY` (`beta.ts`): the debug log could not be exported then.
13. *(Fixed in 0.10.1.)* **Deploy defaults:** `deploy.sh` `HOMELAB` default (`$HOME/dev/ansible-homelab`) does not match the clone on this laptop; its usage line lists dev, prod, emma, jonas, demo only.

11.4 Backlog pointers

`docs/BACKLOG.md`: done in 0.9.x: place picture, data limit, debug export, road-test fixes A, C-F, own AI, swisstopo, colour-blind traffic, route lines. Done in 0.10.x: tile cache with size limit (`tile_cache_mb`), north-up/heading-up map (`map_rotate`), deviation from the planned route in the logbook (`plan_path`), car statistics (fill-ups, consumption, range: `car_*`), photos in the logbook PDF (`logbook_photo inline`), repeated over-speed beep (`speed_beep_every`). Done in 0.10.2-0.11.2: see the changes section at the top. **NEW** Done in 0.13.0: own Overpass server, data limit in money, Instruments show / hide, SCAN 25, contacts in the app (phone, Nextcloud), TomTom web pill, Linux build. Phase D: licensing (`docs/LICENSING-PROPOSAL.md`), own servers, Bluetooth car detection, Android Auto later. Website plan: `docs/app/WEBSITE-PLAN.md`.

---

12. Credits and licences

Full list: `CREDITS.md`, shown in the app under Setup → About → Credits (`CREDITS`, `card.ts`).

Source	Licence / terms	Attribution
OpenStreetMap (tiles, Overpass, Nominatim)	ODbL 1.0 data, CC BY-SA 2.0 tiles	© OpenStreetMap contributors
swisstopo	Free use with attribution (OGD)	© swisstopo
Esri World Imagery	Esri terms	Esri, Maxar, Earthstar Geographics, GIS User Community
OpenTopoMap	CC BY-SA 3.0	© OpenTopoMap
TomTom	Developer terms, user's own key	© TomTom
NEW Mapy.com	REST API terms, user's own key	« © Seznam.cz a.s. a další » (linked) and the Mapy.com logo over the map
Wikipedia / Wikidata / Wikimedia Commons	CC BY-SA 4.0 / CC0 / per file	Link; author · licence under each photo
OpenTripMap	ODbL data, service terms	OpenTripMap
OSRM	BSD-2 software, ODbL data	–
Open-Meteo	CC BY 4.0	« Weather data by Open-Meteo.com »
prix-carburants.gouv.fr	Licence Ouverte 2.0	–
GeoNames	CC BY 4.0	–
AI providers	The user's own account and key	–
Tauri 2 / Leaflet / Home Assistant	MIT or Apache-2.0 / BSD-2-Clause / Apache-2.0	–
MapLibre GL JS 6.14.0 (3D map)	BSD-3-Clause	« MapLibre (BSD-3) » in the 3D view
Mapzen terrain tiles on AWS (3D map)	Open data, various sources	« Terrain: Mapzen terrain tiles on AWS (various sources, see their attribution) »
MDI icon paths (`icons.ts`)	Apache-2.0 (@mdi/js 7.4.47)	–
App icon	Pin drawn in VectorCraft/PhotoCraft on « Le lac de Genève », Adolphe Joanne, 1888 (BnF, public domain)	`app/icon-source/PHOTO-SOURCES.md`

Routes, speed limits and traffic are guidance only. The software is provided « as is », without warranty of any kind. Enquiries: hello@nestour.app.

© 2026 The Development Team. All rights reserved.



Nestour GPS

Every place worth a stop, on your way

IN SHORT

- Places worth a stop along your route, rated with stars
- Plans for the days ahead, a logbook of where you have been
- Turn-by-turn guidance with voice, in 67 languages
- Your data stays on your device

Technical Reference

Version 0.15.0 · 5 November 2026

> Beta — free until 28 February 2027

CONTACT

hello@nestour.app

© 2026 The Development Team. All rights reserved.

Map data © OpenStreetMap contributors · swisstopo · TomTom
Wikipedia/Wikimedia Commons (CC BY-SA) · Terrain: Mapzen/AWS · MapLibre