

12-reitti.md

kstack: book: Centerpoint Home
Lab chapter: Documents &
Organization page: Reitti tags:
[reitti, fitness, activity-tracking,
postgis, rabbitmq, redis]

Overview

Reitti is a self-hosted activity and fitness tracking platform. It records GPS tracks, check-ins, and location history and displays them on interactive maps. The platform is a 6-container stack: the core Java application, a PostGIS spatial database, Redis cache, RabbitMQ message queue, a tile proxy for map tiles, and Photon for geocoding.

External access is at `https://reitti.jeeves5454.ddns.net`. Check-in data from Foursquare/Swarm is synced into Reitti via the separate **swarm-reitti-bridge** service (see next page).

Access

Type	URL	Auth
External	<code>https://reitti.jeeves5454.ddns.net</code>	Reitti own auth + letsencrypt

No internal `home.local` route — external-only.

Containers

Container	Image	Role
reitti-app	dedicatedcode/reitti:latest	Core Java application
reitti-postgis	postgis/postgis:17-3.5-alpine	Spatial database (PostgreSQL 17 + PostGIS 3.5)
reitti-redis	redis:7-alpine	Cache and session store
reitti-rabbitmq	rabbitmq:3-management-alpine	Message queue for async jobs
reitti-tiles	nginx:alpine	Map tile caching proxy
reitti-photon	rtuszik/photon-docker:1.3.0	Local geocoding (Nominatim-based)

reitti-app

The main Spring Boot application serving the Reitti web UI and REST API. Ingests location data via the OwnTracks-compatible endpoint used by swarm-reitti-bridge.

Key environment variables:

Variable	Value / Notes
POSTGIS_HOST	postgis
POSTGIS_PORT	5432
POSTGIS_DB	reittidb
POSTGIS_USER	reitti
POSTGIS_PASSWORD	REDACTED
REDIS_HOST	redis
REDIS_PORT	6379
RABBITMQ_HOST	rabbitmq
RABBITMQ_PORT	5672
RABBITMQ_USER	reitti
RABBITMQ_PASSWORD	REDACTED
PHOTON_BASE_URL	http://photon:2322

Volumes:

Volume / Mount	Container Path	Purpose
Docker volume reitti_reitti-data	/data	Application data and uploads

Traefik labels:

```
traefik.http.routers.reitti.rule: Host(`reitti.jeeves5454.ddns.net`)
traefik.http.routers.reitti.entrypoints: websecure
traefik.http.routers.reitti.tls.certresolver: letsencrypt
traefik.http.services.reitti.loadbalancer.server.port: 8080
```

reitti-postgis

PostgreSQL 17 with PostGIS 3.5 spatial extension. Stores all GPS track data, waypoints, and geospatial indices.

Image: `postgis/postgis:17-3.5-alpine`

Variable	Value
<code>POSTGRES_USER</code>	<code>reitti</code>
<code>POSTGRES_DB</code>	<code>reittidb</code>
<code>POSTGRES_PASSWORD</code>	REDACTED

Volumes:

Volume	Container Path
<code>reitti_postgis-data</code> (named)	<code>/var/lib/postgresql/data</code>

reitti-redis

Redis 7 (Alpine) — used for caching and session management.

Image: `redis:7-alpine`

Volumes:

Volume	Container Path
<code>reitti_redis-data</code> (named)	<code>/data</code>

No authentication configured — network-isolated to the Reitti internal stack network.

reitti-rabbitmq

RabbitMQ 3 with management plugin. Handles async processing of imported tracks and location events.

Image: rabbitmq:3-management-alpine

Variable	Value
RABBITMQ_DEFAULT_USER	reitti
RABBITMQ_DEFAULT_PASS	REDACTED

Volumes:

Volume	Container Path
reitti_rabbitmq-data (named)	/var/lib/rabbitmq

RabbitMQ management UI is available internally on port 15672 (not exposed via Traefik).

reitti-tiles

NGINX-based map tile proxy/cache. Serves map tiles from upstream tile providers with local caching to reduce external requests.

Image: nginx:alpine (NGINX 1.29.8)

Variable	Value
NGINX_CACHE_SIZE	1g

No persistent volumes — tile cache is in-memory. No external Traefik route.

reitti-photon

Local geocoding service using the Photon engine (Nominatim/OSM-based). Configured for the ca (Canada) region with parallel update strategy.

Image: rtuszik/photon-docker:1.3.0

Variable	Value
REGION	ca
UPDATE_STRATEGY	PARALLEL

Volumes:

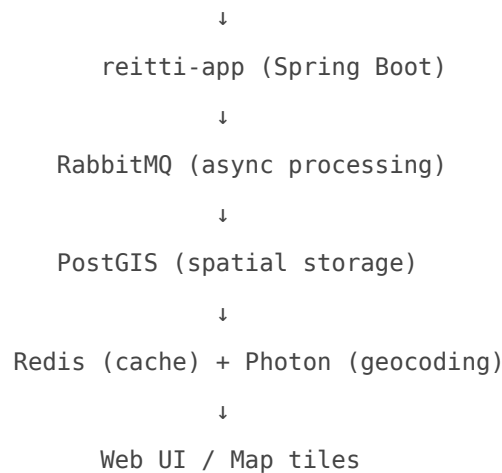
Volume	Container Path
<code>reitti_photon-data</code> (named)	<code>/photon/data</code>

The photon data volume holds the downloaded Nominatim extract for the CA region. It can be large (several GB); allow time for the initial data download on first start.

Data Flow

GPS device / OwnTracks → Reitti OwnTracks ingest API

Foursquare/Swarm check-ins → swarm-reitti-bridge → Reitti OwnTracks ingest API



Notes / Gotchas

- All named Docker volumes (`reitti_reitti-data`, `reitti_postgis-data`, `reitti_redis-data`, `reitti_rabbitmq-data`, `reitti_photon-data`) must be included in any backup strategy — none are bind-mounted to host paths.
- PostGIS 17 requires spatial extensions to be enabled in the database on first run — `dedicatedcode/reitti:latest` handles this automatically on startup.
- The Photon geocoding data download is region-specific (`REGION: ca`). If geocoding stops working after a restart, check that the photon data volume is intact and the photon container started successfully.
- RabbitMQ management UI (port 15672) is available inside the Docker network but is not exposed externally. Access via `docker exec` or Portainer console if needed.
- Check-in data from Foursquare/Swarm flows through **swarm-reitti-bridge** (see next page). Reitti itself has no Foursquare integration — the bridge translates and delivers data via the OwnTracks ingestion endpoint.

Revision #2

Created 2026-06-17 04:45:56 UTC by Admin

Updated 2026-06-17 04:45:56 UTC by Admin