Mozout

Generated on 15/11/2026 00:39

KNOWLEDGE BASE · PUBLIC DOCUMENT

Evolution API with PostgreSQL and Redis: sessions that survive a restart

Evolution API

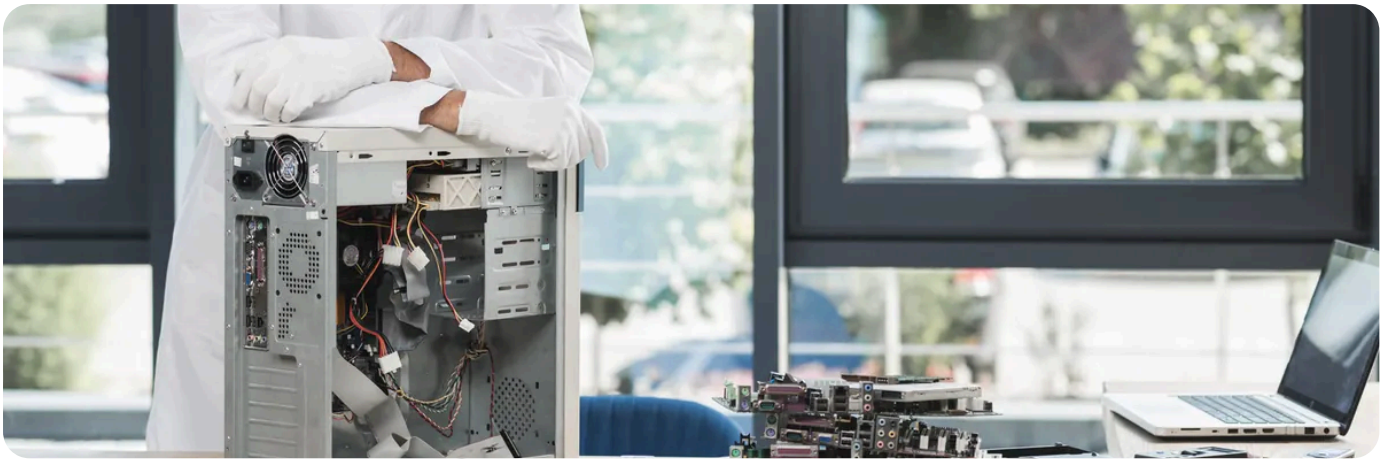
CODE	VERSION	COUNTRY · LANGUAGE
RLX-INT-KB-3048	v2026.10.10	International · English

MozOut International
RLX Holdings (Pty) Ltd. · Group RLX Holdings
Tax no. (SARS) 943 907 6234 · Registration no. (CIPC) 2025/667953/07

https://www.my.mozout.com/knowledgebase/3048

Evolution API with PostgreSQL and Redis: sessions that survive a restart

Evolution API Created on **October 10, 2026** 3 min read



An Evolution API install that loses its sessions at every restart is an install with no **persistent volume**. The database keeps the instances and messages; Redis, which many guides add, is support; and Docker volumes make sure none of it vanishes when the container is recreated. This article shows how the pieces fit and how to **prove** that sessions survive.

What each piece does

Piece	What it is for	If it is missing
Evolution API	Receives the requests and talks to WhatsApp.	Nothing works.
PostgreSQL or MySQL	Keeps the instances and data, according to the project's documentation.	The API will not start, or starts with no memory of what existed.
Redis	Many guides use it as support. The documentation says what your version requires.	Depending on the setup, the service complains or runs with less support.
Docker volumes	Keep on disk what must not disappear.	Every restart wipes the sessions and makes you scan the QR codes again.

Both databases are supported. PostgreSQL is the one most guides show, but MySQL or MariaDB are fine if that is what you already know. If you are unsure, see **PostgreSQL or MySQL: which one for your project**.

Building it to survive

1. **Put the database in a compose service** with a named volume for its data. The volume is what remains when the container disappears. For the layout, see **Docker Compose with a database**.

2. **Give the API's instances a volume too**, as the project's example file shows.

3. **Make the services talk by name.** On the same Compose network, the database address is the service name, not `localhost`. The user and password must match on both sides.

4. **Do not publish the database or Redis ports.** Only the API needs to reach them, inside the Docker network.

5. **Set the restart policy** so the services come back by themselves after a server reboot. Compose has an option for it, and the project's example usually includes it.

Proving it works

1. **Link a test instance** and confirm it sends.

2. **Restart the services** with `docker compose restart`. The instance should come back linked, with no QR.

3. **Stop and recreate:** `docker compose down` then `docker compose up -d`. This deletes the containers but not the volumes. The instance is still there.

4. **Reboot the VPS itself** once, at a quiet moment, and see whether everything comes back on its own.

A volume is not a backup.
It protects against a restart, not against a failing disk, a container removed by mistake or a `docker compose down -v`. Take database dumps (see **backups with mysqldump and pg_dump**) and keep them off the server.

Now and then check the space the volumes and logs take. A full database brings everything down at once: **Docker filling the disk**. And to update without losing anything, **updating and backing up Evolution API**.

RECOMMENDED
Need more disk for the database? Have a look at the VPS plans.
See VPS servers →

SEE ALSO

- Docker Compose with a database
- Backups with mysqldump and pg_dump
- Updating and backing up Evolution API

MozOut International

RLX Holdings (Pty) Ltd. · Group RLX Holdings

TAX NO. (SARS)	943 907 6234
REGISTRATION NO. (CIPC)	2025/667953/07
REGISTERED OFFICE	1111 Olive Street Saint Louis 63101, Missouri, United States
CONTACT	info@mozout.com · +27 78 823 3325 · www.mozout.com
OPENING HOURS	Monday to Friday - 08 AM to 04 PM Saturday & Sunday - Closed

Online version of this document

RLX-INT-KB-3048 · v2026.10.10 · Generated on 15/11/2026 00:39