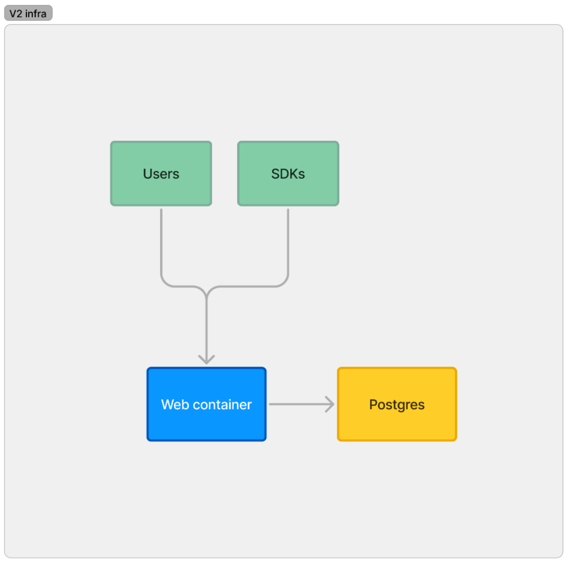
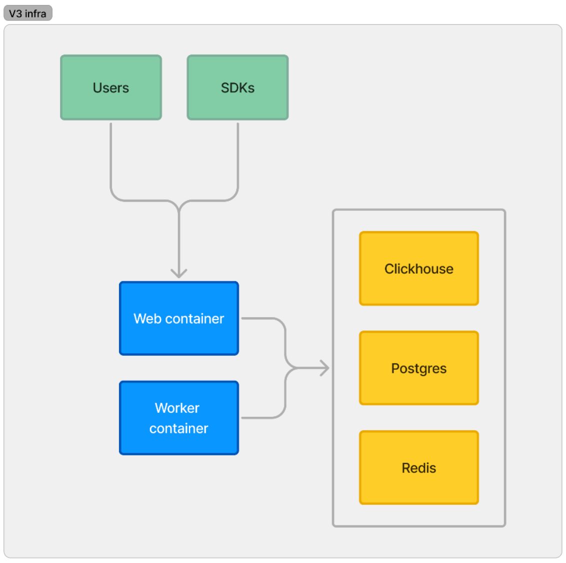
Langfuse v3 Architecture

Architectures v2 vs v3





- Containers
  - `web` container: hosts public api, and all resources for the user interface
  - `worker` container: asynchronous processes, no exposed ports
- Databases
  - `Redis` used as cache and queue
  - `Postgres` stores transactional data such as projects or API keys
  - `Clickhouse` stores tracing data generated by the SDKs. This database will do most of the processing as our server will insert all the SDK data and read it for tables and dashboards.

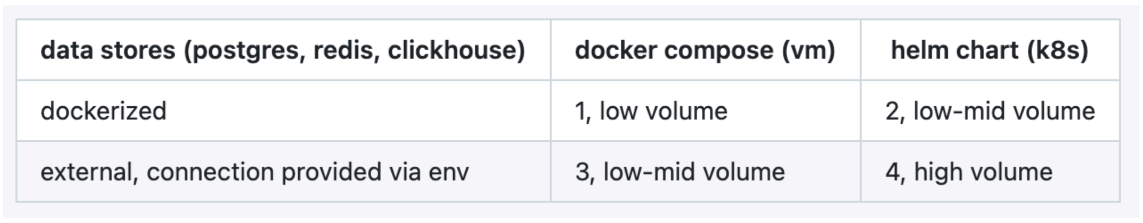
Next to the core application, an **application load balancer** for TLS termination and routing of requests to the `Web` container is necessary. We use nginx but you can also use the fully managed AWS load balancer.

Upgrade path from v2 to v3

Thousands of teams run on Langfuse (~400k docker pulls)
→ we aim to offer the easiest migration experience that is automated and documented

- Guidance on scalability of dockerized datastores, when are fully-managed datastores necessary
- Guidance of reliable auto-scaling configuration of the application and databases
- Migration script to move data from Postgres to Clickhouse

Application deployment



data stores (postgres, redis, clickhouse)	docker compose (vm)	helm chart (k8s)
dockerized	1, low volume	2, low-mid volume
external, connection provided via env	3, low-mid volume	4, high volume

As there is no unique complexity, deployment on ECS follows the instructions of deploying on K8S.

For low-volume/non-production deployments, dockerized DBs + docker compose is a sensible option to keep complexity low. We will publish guidance on when options 3 and 4 are necessary.

DB deployment

Databases (see above): redis, postgres, clickhouse

Low-volume

- All datastores are available as single docker containers
- Can be bundled in docker compose or EKS/ECS/K8S deployment

High-volume / fully-managed → databases external of application cluster

- Redis
    - No special requirements
    - AWS: we chose ElastiCache (Redis OSS) for a fully managed Redis instance
- Postgres
    - No special requirements compared to other applications.
    - AWS: we chose RDS for a fully-managed Postgres instance
- Clickhouse
    - Main data store, thus this database will scale the most.
    - AWS:
        - There is no RDS-equivalent for Clickhouse on AWS
        - We use Clickhouse Cloud which is the fully managed Clickhouse database managed by the Clickhouse team. This can be run on an AWS region of your choice with VPC peering for private access and data security.
        - There are potentially additional vendors for managed Clickhouse such as DoubleCloud or Altinity.
        - Alternatively, you can run Clickhouse yourself in ECS, e.g. by using this template.

We will provide guidance at which scale high-volume/fully-managed clickhouse is necessary. As discussed, we expect dockerized Clickhouse to perform well at least up to 50 GB of tracing data (10s of millions of traces).