FOR IT PARTNERS AND DEVELOPERS

# Lynk – Technical overview for your *IT partner*

Architecture, stack, security, deployment, operations and extension – so you can decide whether you can run and extend Lynk in your own cloud account. Every statement is traceable to the source repository.

| 6 | 57 | ~10k | 34 | 9 |
|---|---|---|---|---|
| SERVICES | API MODULES | AUTOMATED TESTS | CI WORKFLOWS | UI LANGUAGES |

HP Capital GmbH · Mayrwiesstraße 11 · 5300 Hallwang (Salzburg) · Austria

Repository state: 24 Sep 2026

# Contents


| | | |
|---|---|---|
| 01 | The short version (for the agency owner) | 3 |
| 02 | Architecture | 4 |
| 03 | Technology stack and versions | 5 |
| 04 | Data, queues and storage | 5 |
| 05 | External integrations | 6 |
| 06 | AI layer: providers, model tiers, traceability | 6 |
| 07 | Security, access and multi-tenancy | 7 |
| 08 | Deployment in your own account | 7 |
| 09 | System requirements (estimate) | 8 |
| 10 | Configuration | 9 |
| 11 | Operations: monitoring, backup, updates | 10 |
| 12 | Testing and CI | 10 |
| 13 | Extending Lynk | 11 |
| 14 | Languages | 11 |
| 15 | Status: what exists today | 12 |
| 16 | Limits and constraints | 12 |
| 17 | Before go-live we configure and verify | 13 |
| 18 | Handover package | 14 |


All paths refer to the Lynk source repository (state 24 Sep 2026). No secrets or credentials are reproduced in this document.

## 01 The short version (for the agency owner)

- **What it is:** a web application made of six services (Python and Node.js) plus the databases MongoDB and Redis and an S3-compatible file store. Standard, widely used open-source technology – nothing exotic.
- **Where it runs:** in your own cloud or server account (a Linux server or Kubernetes cluster), in a data centre you choose – for EU agencies we recommend the EU. We set it up there during onboarding and hand everything over; we do not host it. Docker images and a Helm chart are part of the repository.
- **What you own:** the complete source code, the deployment files, the tests and the runbooks. Your IT partner can run, change and extend it without us.
- **What it needs from outside:** your own accounts for the server or cloud, Unipile (LinkedIn connection), an AI (LLM) provider, an e-mail server and file storage – contracted and paid by you (rough costs in section 9). Optional: Microsoft login, Stripe.
- **How honest is this document:** every statement refers to files in the repository. Where something is not finished, it says so (section "Status" and "Before go-live").

This overview reflects the repository state of 24 September 2026. File paths in `monospace` refer to that repository; the six services live under `lynk-code-backup-20260702-084926/<service>__container-0/` (a historical directory name, kept on purpose – see ADR-000 in `docs/adr/`). Paths below are abbreviated to the service name.

## 02 Architecture

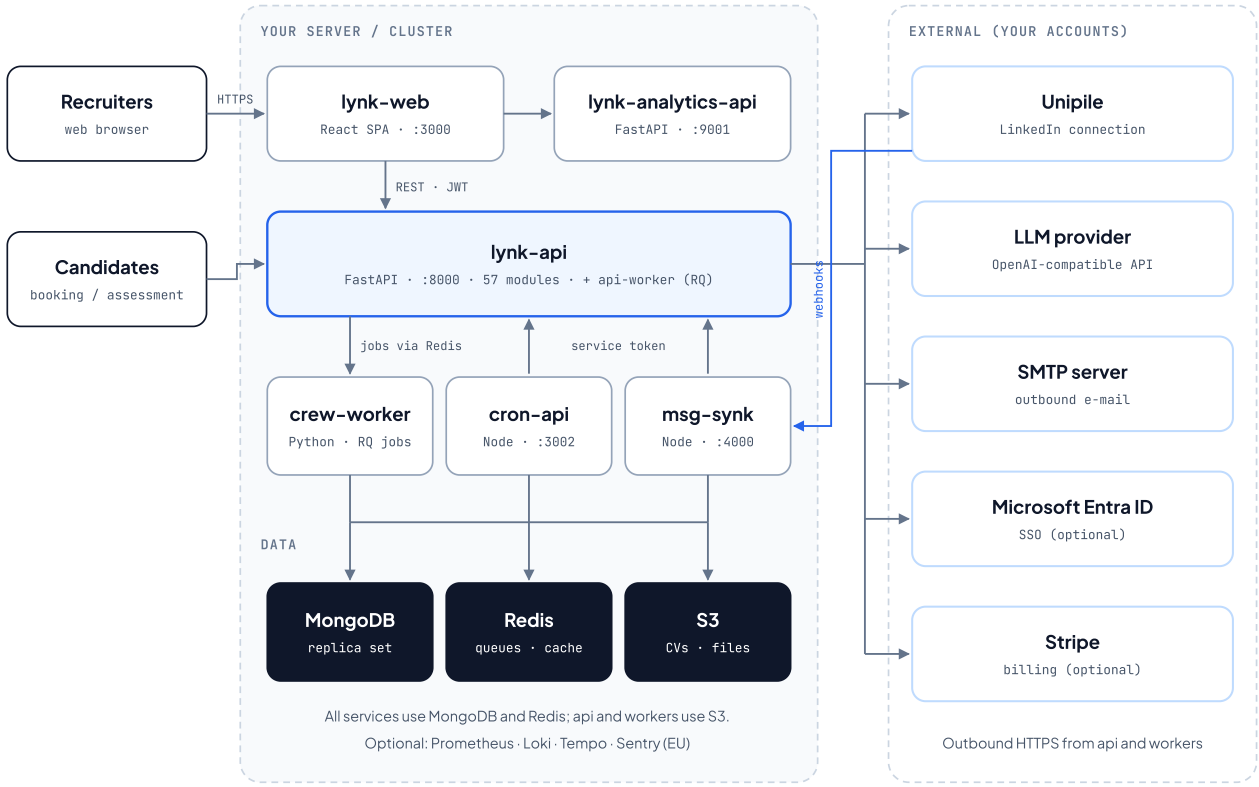



Figure 1 – Services, data stores and external services. Ports are the canonical ones from docker-compose.yml and the Helm chart.

| SERVICE | STACK | RESPONSIBILITY |
|---|---|---|
| `lynk-web` | React 19 · Vite 6 · TypeScript | Browser UI (single-page app), served on :3000. Talks to lynk-api and lynk-analytics-api; Microsoft sign-in via MSAL. |
| `lynk-api` | Python · FastAPI | Core REST API (:8000): auth, workspaces, jobs, candidates, matching, messaging, assessments, booking, billing. 57 feature modules. The same image also runs as `lynk-api-worker` (RQ queue `api_tasks`). |
| `lynk-crew-worker` | Python · RQ · CrewAI | Background jobs from priority-ordered Redis queues: LinkedIn search and import, CV/profile extraction, AI profile analysis, job-profile tasks. |
| `lynk-analytics-api` | Python · FastAPI | Read-side analytics and activation/cohort metrics (:9001). |
| `lynk-cron-api` | Node 20 · TypeScript · Express | Scheduler (croner): follow-up schedules and periodic tasks; calls lynk-api with a service token (:3002). |
| `lynk-msg-synk` | Node 20 · TypeScript · Express | Messaging sync: receives Unipile webhooks (HMAC-verified), stores LinkedIn messages and connection events (:4000). |

Sources: `DEVELOPMENT.md`, `docker-compose.yml`, `deploy/helm/lynk/values.yaml`, each service's `Dockerfile` and `package.json` / `pyproject.toml`.

## 03 Technology stack and versions

| LAYER | TECHNOLOGY | VERSION / PIN |
|---|---|---|
| **Backend (Python)** | FastAPI, Pydantic 2, Beanie ODM + mongoengine, RQ, structlog, OpenTelemetry | Python 3.10.12 pinned via `uv` (`pyproject.toml`, `uv.lock`); FastAPI 0.139.2 (api) / 0.115.8 (analytics) |
| **AI orchestration** | OpenAI Python SDK, LangChain core, CrewAI | `openai ⩾ 2.46`, `crewai ⩾ 1.15.3` |
| **Backend (Node)** | Express 4, Mongoose 8, ioredis, pino, prom-client | Node 20, TypeScript 5 |
| **Frontend** | React 19, Vite 6, Ant Design 5, Tailwind CSS 3, Redux Toolkit, TanStack Query 5, React Router 6, i18next | TypeScript 5.9, Node 20 build image |
| **Data** | MongoDB, Redis | `mongo:8`, `redis:7-alpine` in compose; MongoDB 8.0 on staging |
| **Containers** | Docker (multi-stage, non-root user), Helm 3–compatible chart, ArgoCD | Images per git commit on GHCR (`build-push-ghcr.yml`) |

Server-rendered HTML is not used; the UI is a static bundle. The api image includes a headless Chromium (Playwright) for rendering external job pages, which makes that image larger than the others.

## 04 Data, queues and storage

| STORE | USED FOR | NOTES FOR OPERATORS |
|---|---|---|
| **MongoDB** | All application data: users, workspaces, jobs, candidates, messages, scores, decision log, LLM usage, feature flags. 71 document models registered in the async ODM (`src/db/beanie_connect.py`). | Run as a **replica set** in production: money-path writes use transactions and fail closed on a standalone server (`src/db/tx.py`). Managed (e.g. MongoDB Atlas, EU region) or self-hosted both work. |
| **Redis** | RQ job queues (priority-ordered), scheduler locks, cache, cluster-wide rate-limit counters. | Enable persistence (AOF) so queued jobs survive a restart; recovery steps in `RQ-RECOVERY.md`. |
| **S3-compatible object storage** | CVs and uploaded documents. | Any S3 API (AWS, Cloudflare R2, MinIO, Hetzner …) via `AWS_ENDPOINT`. Keep the bucket private. |

**Schema changes** run through a small built-in migration runner (`lynk-api/migrations/`, `python -m migrations.runner status|up`); applied versions are recorded in the `_migrations` collection so each runs exactly once (`docs/migrations-runbook.md`). The codebase is mid-way through moving from mongoengine to the async Beanie ODM (ADR-002); both layers are in use today.

## 05 External integrations

| SERVICE | PURPOSE | HOW IT CONNECTS |
|---|---|---|
| **Unipile (LinkedIn)** | LinkedIn search, profile import, messages, invitations. Each recruiter connects their own LinkedIn account. | REST (`LINKEDIN_*` variables) from api/workers; inbound webhooks to msg-synk, HMAC-signed (`WEBHOOK_HMAC_ENFORCE` defaults to on). Per-account sending limits in Lynk. |
| **LLM provider** | CV extraction, profile analysis, match reasoning, message drafts. | Any OpenAI-compatible HTTP API; three model tiers (section 6). |
| **SMTP** | Transactional mail: invitations, OTP codes, booking confirmations with .ics. | Standard SMTP (`EMAIL_*`). Configure SPF/DKIM for your sending domain. |
| **S3 storage** | CVs, documents. | `AWS_*` / `AWS_ENDPOINT` |
| **Microsoft Entra ID** | Single sign-on for recruiters (optional). | MSAL in the browser (`VITE_CLIENT_ID`, `VITE_TENANT_ID`); token verified server-side via OIDC discovery/JWKS. A generic OIDC path exists behind the runtime flag `sso.oidc`. |
| **Your ATS / CRM** | Import of jobs, applications and candidates; write-back where your system and plan allow it. One connection is included in the 8-week program (set up and tested); each further connection $2,500 one-time (net). Read/write scope is agreed in writing before purchase. Bullhorn only on request and subject to Bullhorn's approval for AI vendors. | Connector framework in lynk-api (`ats_connect/`: capability model, registry, sync, ingest). Existing adapters: d.vinci, Personio, Teamtailor, SAP SuccessFactors. Other systems are connected per customer via the vendor's API or, where useful, an integration service. You connect with your own account at the vendor. |
| **Stripe** | Subscription billing code exists (checkout, webhooks, dunning). | Optional for a self-owned installation; `STRIPE_*`. |

## 06 AI layer: providers, model tiers, traceability

All LLM calls go through one client (`lynk-api/src/common/openai_llm.py`) that speaks the OpenAI-compatible chat API. The provider is configuration, not code: point `LLM_BASE_URL` at any compatible endpoint (a commercial EU provider, Azure OpenAI, or a self-hosted vLLM/Ollama server). Each call is assigned a *purpose*, and each purpose maps to a model tier that can have its own model, endpoint and key:

| TIER | USED FOR | VARIABLES |
|---|---|---|
| **FAST** | mechanical outputs (numbers, flags, short lists) | `LLM_MODEL_FAST`, `LLM_BASE_URL_FAST`, `LLM_API_KEY_FAST` |
| **QUALITY** | extraction (CVs, job descriptions) | `LLM_MODEL_QUALITY`, `…_QUALITY` |
| **PREMIUM** | candidate-facing text (message drafts) | `LLM_MODEL_PREMIUM`, `…_PREMIUM` |

Unset tiers fall back to `LLM_MODEL` / `LLM_BASE_URL`.

- **Usage metering:** every call writes an `LLMUsage` record (model, purpose, tokens, cost estimate from `llm_pricing.json`) – including calls from the crew-worker.
- **Config fingerprint:** on every boot each service logs one `LLM_CONFIG_FINGERPRINT` line and stores an `LLMConfigFingerprint` document (SHA-256 over provider and per-tier model/host; never the key, host name only). This answers "which model scored this candidate?".
- **Model changelog:** any model or endpoint change requires a row in `MODEL_CHANGELOG.md`. This is preparation for EU AI Act record-keeping (recruitment scoring is listed as high-risk in Annex III).
- **Human in the loop:** the match score is labelled as an automated estimate, not a hiring decision; messages are drafts that a recruiter reviews and sends; assessments are not graded by AI.

## 07 Security, access and multi-tenancy

| AREA | IMPLEMENTATION |
|---|---|
| **Authentication** | Short-lived JWT access tokens (HS256) plus refresh tokens; a missing secret fails closed, and lynk-analytics-api refuses placeholder secrets or ones shorter than 32 characters. E-mail OTP login supported. |
| **SSO** | Microsoft (Entra ID) via MSAL/OIDC; generic OIDC behind a flag. |
| **MFA** | TOTP (authenticator apps), secrets encrypted at rest with a field-encryption key (`src/modules/mfa/`). |
| **Multi-tenancy** | All data is scoped to a **workspace**; helpers in `src/common/tenancy.py` scope queries and return 404 for other tenants' IDs. Cross-tenant isolation tests run in CI. |
| **Service-to-service** | Signed service tokens ("service principals") between cron-api / msg-synk and lynk-api (`SERVICE_PRINCIPALS_ENABLED`, `SERVICE_JWT_SECRET`). |
| **Hardening** | Rate limiting (Redis-backed), request body limits, SSRF guard for outbound fetches, security headers, non-root containers, HMAC on inbound webhooks, secrets only server-side (`VITE_*` values are public by design). |
| **Supply chain** | CI runs secret scanning (Gitleaks), SAST (Semgrep), container scanning (Trivy), dependency audit + Dependabot, and generates an SPDX SBOM per main build (Syft). |
| **GDPR** | Retention CronJob (`deploy/gdpr/`), export/deletion endpoints (`gdpr` module), marketing double opt-in behind a flag. |

Vulnerability reporting and response times: `SECURITY.md` (acknowledgement within 3 business days, fix or mitigation target 30 days).

## 08 Deployment in your own account

Lynk runs in **your own cloud or server account**, opened in your agency's name with a provider of your choice. During onboarding we install it there with temporary admin access, connect your providers, test it with you and hand over all credentials; then we remove our access. Your IT team or IT partner runs it day to day. We do not host Lynk.

| OPTION | WHAT IS IN THE REPOSITORY | SUITABLE FOR |
|---|---|---|
| **Docker Compose** | `docker-compose.yml`: MongoDB 8, Redis 7 and all six services built from their Dockerfiles, with health checks. | Local development and evaluation. The file is marked "not for production" (standalone MongoDB, no TLS, no backups). |
| **Kubernetes (Helm)** | `deploy/helm/lynk/`: umbrella chart for all services incl. `lynk-api-worker`; per-environment values, HPA/KEDA, PodDisruptionBudgets, probes. `deploy/argocd/`: GitOps applications (manual sync). `infra/`: network policies, monitoring, backup schedules, External Secrets. | Production. Reference environment: single-node k3s on a Hetzner VM (staging). Any conformant Kubernetes should work. |
| **Single VM with containers** | Same Docker images; configuration via env files. | Small installations. A production-grade setup (replica-set MongoDB, TLS reverse proxy, backups) is configured during onboarding; it is not a ready-made file in the repository today. |

**Secrets** are never committed. The reference setup uses Doppler → External Secrets Operator → one Kubernetes Secret mounted by every service (`deploy/secrets/`); any secret store that can populate environment variables works. **Health endpoints:** api `/health` (liveness) and `/ready` (503 with per-dependency breakdown), analytics `/health`, web `/`, msg-synk `/`.

## 09 System requirements (estimate)

Estimate – not a measured benchmark. Based on the Helm resource settings and staging observations; validate with your own load.

| SCENARIO | COMPUTE | DATA |
|---|---|---|
| **Evaluation / development (Docker Compose)** | 1 VM, 4 vCPU, 8 GB RAM, 40 GB SSD | Local MongoDB + Redis containers |
| **Production, one agency (about 5–20 recruiters)** | Kubernetes with 1–3 nodes of 4 vCPU / 8 GB each, or one VM with 4–8 vCPU / 16 GB | MongoDB 3-member replica set (managed or self-hosted), 20–50 GB to start; Redis with AOF, 1 GB; S3 bucket, grows with CV volume |

For reference: the Helm defaults request about 1.9 GB RAM and 0.7 vCPU across the seven app deployments, with limits of about 4.5 GB and 7 vCPU (`deploy/helm/lynk/values.yaml`). lynk-api is memory-bound (about 470 MB observed on staging). Monitoring (Prometheus, Grafana, Loki) adds roughly 1–2 GB if you run it in-cluster. LLM inference is not included – it runs at your LLM provider unless you self-host a model, which needs GPU capacity sized separately.

### What you pay directly

You contract these providers in your own name and pay them directly, not us. Rough guide values at the date of this document; the providers set the prices.

| ITEM | ROUGH COST | NOTE |
|---|---|---|
| **Server or cloud account** | ≈ €/$ 50–150 per month | Depends on provider, region and size (see above); includes database and file storage if managed by the same provider. |
| **AI (LLM) provider** | ≈ €/$ 200–300 per month for 10 recruiters | Usage-based; Lynk meters every call so you see the cost per task. |
| **LinkedIn connection (Unipile)** | ≈ €/$ 5 per recruiter per month | One connected account per recruiter. |
| **E-mail sending (SMTP)** | small | Depends on your provider and volume. |

## 10 Configuration

Every service ships an `.env.example` that lists every variable the code reads, with safe local defaults where possible ( `DEVELOPMENT.md` ). The Kubernetes key set is templated in `infra/secrets/lynk-secrets.example.yaml` . No real values are in either. Variable groups:

| GROUP | EXAMPLES (NAMES ONLY) |
|---|---|
| **Server / environment** | `HOST` , `PORT` , `ENVIRONMENT` , `FRONTEND_URL` , `BACKEND_URL` , `CRON_SERVICE_URL` |
| **MongoDB** | `DATABASE_URI` (Python), `MONGO_URL` + `MONGODB_URI` (cron), `MONGO_URI` (msg-synk) – must all point to the same database |
| **Redis** | `REDIS_HOST` , `REDIS_PORT` , `REDIS_USERNAME` , `REDIS_PASSWORD` |
| **Auth** | `JWT_TOKEN_SECRET` , `ACCESS_TOKEN_TTL_MIN` , `REFRESH_TOKEN_EXPIRES_IN` , `FIELD_ENCRYPTION_KEY` , `SERVICE_PRINCIPALS_ENABLED` , `SERVICE_JWT_SECRET` |
| **LLM** | `LLM_PROVIDER` , `LLM_BASE_URL[_FAST\|_QUALITY\|_PREMIUM]` , `LLM_MODEL[_…]` , `LLM_API_KEY[_…]` , `LLM_API_TIMEOUT` |
| **LinkedIn / Unipile** | `LINKEDIN_API_KEY` , `LINKEDIN_BASE_URL` , `UNIPILE_WEBHOOK_SECRET` , `WEBHOOK_CALLBACK_URL` |
| **S3** | `AWS_ACCESS_KEY_ID` , `AWS_SECRET_ACCESS_KEY` , `AWS_REGION` , `AWS_ENDPOINT` , `AWS_STORAGE_BUCKET_NAME` |
| **E-mail (SMTP)** | `EMAIL_HOST` , `EMAIL_PORT` , `EMAIL_USER` , `EMAIL_PASS` , `EMAIL_SENDER_EMAIL` , `EMAIL_SENDER_NAME` |
| **Stripe** | `STRIPE_SECRET_KEY` , `STRIPE_WEBHOOK_SECRET` |
| **Web build (public)** | `VITE_API_URL` , `VITE_ANALYTICS_API_URL` , `VITE_CLIENT_ID` , `VITE_TENANT_ID` , `VITE_MS_REDIRECT_URL` |
| **Observability** | `SENTRY_DSN` , `OTEL_EXPORTER_OTLP_ENDPOINT` , `LOG_LEVEL` |
| **Feature switches** | `*_ENABLED` environment switches, see section 15 |

**Feature flags** exist on two levels: deploy-time environment switches ( `*_ENABLED` , `VITE_*_ENABLED` for the UI) and runtime flags stored in MongoDB ( `FeatureFlag` collection, 30-second cache). A broken flag store always degrades to "off" ( `src/common/feature_flags.py` ). This separates deploying code from releasing a feature.

## 11 Operations: monitoring, backup, updates

### Logging and monitoring

Python services log JSON via structlog, Node services via pino, all to stdout (ready for Loki or any log shipper). Prometheus `/metrics` endpoints, OpenTelemetry traces (OTLP, e.g. Tempo) and Sentry error reporting are built in and optional. `infra/monitoring/` contains Prometheus alert and SLO rules (with unit tests), Grafana dashboards and Alertmanager configuration; `uptime-check.yml` probes from outside the cluster. The LLM config fingerprint (section 6) is part of this trail.

### Backup and disaster recovery

`deploy/backup/mongo-backup-cronjob.yaml` runs a nightly `mongodump`, encrypts it against a public certificate (RSA-4096 / AES-256; the cluster cannot decrypt its own backups), writes a SHA-256 manifest and uploads to S3-compatible storage (default retention 14 days); a success timestamp feeds a "no recent backup" alert. A scheduled restore-drill job and `restore.sh` are included, as are Velero schedules (`infra/backup/`). `DR-RUNBOOK.md` covers rebuild from zero, node loss, Redis loss, credential compromise, bad deploy and database corruption, and marks each step as proven, unverified or a gap. A managed MongoDB with point-in-time recovery is recommended in addition.

### Updates and upgrade path

1. Updates arrive as commits/tags in your repository, with a changelog.
2. CI builds one image set per commit (`build-push-ghcr.yml`), tagged with the git SHA.
3. Deploy to staging first; the smoke suite (`tests/staging/run.sh baseline`) must pass.
4. Promote the same SHA to production (`promote-prod.yml` pins `values-prod.yaml`; a parity guard checks staging ran it), then sync in ArgoCD.
5. Run pending database migrations (`python -m migrations.runner up`).
6. Rollback: re-pin the previous SHA (`docs/prod-promote-rollback.md`).

If you change the code yourself, merge our updates like any upstream branch; the module structure (section 13) keeps conflicts local.

Updates are included for 12 months after go-live. After that, an optional maintenance contract (updates and support) is available, price on request; without it, your installation keeps running as it is.

## 12 Testing and CI

| KIND | WHERE | APPROX. COUNT |
|---|---|---|
| **Python unit and integration tests (pytest)** | `lynk-api/tests`, `lynk-crew-worker/tests`, `lynk-analytics-api/tests` | ~7,600 test functions in ~550 files (more cases via parametrisation) |
| **TypeScript tests (Vitest)** | lynk-web, lynk-cron-api, lynk-msg-synk | ~2,600 tests in ~275 files |
| **Container tests (Testcontainers)** | `container-tests.yml` | real MongoDB/Redis |
| **Staging smoke / end-to-end / visual** | `tests/staging`, `tests/e2e`, `tests/visual` | markers: smoke, writes, live_only, perf, destructive |

**34 GitHub Actions workflows** (`.github/workflows/`): lint and type checks, unit and container tests, OpenAPI drift and completeness, generated API types freshness, bundle-size and web-quality budgets, accessibility/visual checks, compose smoke, infrastructure validation, security scanners (Gitleaks, Semgrep, Trivy, dependency audit), SBOM, image build/push and retention, staging smoke and rollback, prod promotion, nightly e2e, load test, flake report and uptime check. In total roughly 10,000 automated tests.

## 13 Extending Lynk

lynk-api is organised as feature modules under `src/modules/<name>/`. At startup `load_routers()` (`src/__init__.py`) mounts every module package that has a `router.py`; modules without one are libraries. A typical module:

```
src/modules/myFeature/
  __init__.py      # marks the package
  router.py        # FastAPI APIRouter - HTTP surface, auth dependencies
  handler.py       # business logic
  schema.py        # Pydantic request models
  responses.py     # typed response models (feed the OpenAPI checks)
  model.py         # Beanie document(s) - register in src/db/beanie_connect.py
  tasks.py         # RQ background jobs (optional)
tests/test_my_feature.py
```

- **Where to add:** a new capability goes into a new module; shared helpers into `src/common/`. Scope every query to the workspace (`tenancy.py`). Guard new features with a flag so they can ship dark.
- **Background work:** enqueue an RQ job; long-running AI tasks belong in the crew-worker, API-adjacent tasks in the `api_tasks` queue.
- **Frontend:** pages in `lynk-web/src/Pages`, API calls in `src/API`, state in `src/Slices` / `src/Stores`, strings in the i18n catalogues (all 9 locales, enforced by a parity test).
- **Conventions** (`CLAUDE.md`): never commit to `main` – branch → pull request → review; Conventional Commit titles (`feat:`, `fix:`, `refactor:`, `chore:`, `test:`, `docs:`, `ci:`, `perf:`); all code and docs in English; never weaken assertions or skip tests to get green; flaky tests are findings; ruff and mypy for Python, ESLint and `tsc` for TypeScript; LLM model changes need a `MODEL_CHANGELOG.md` row.
- **Local start:** `docker compose up mongo redis`, then run a service with `uv run fastapi dev src` or `yarn dev`; `tests/staging/ci_seed.py` seeds two test tenants and prints a working JWT.

## 14 Languages

The UI and the outreach message templates are available in **9 locales**: English, German, French, Spanish, Italian, Dutch, Polish, Portuguese (Portugal) and Portuguese (Brazil). The locale registry is `lynk-api/src/config/locales.json`; seven further codes are registered but switched off for the UI. Web catalogues use i18next; tests enforce key parity and plural forms across locales. Adding a language means adding a catalogue and switching the locale on – no code change in the modules. Automatic machine translation (DeepL) is not wired up.

## 15 Status: what exists today

**LIVE – ALL FEATURES SWITCHED ON AT GO-LIVE**

- LinkedIn search and one-click import via Unipile (per-recruiter connection)
- CV parsing; duplicate detection (skips duplicates, does not merge)
- 5-factor match score with breakdown (skills 40 %, experience 25 %, title 15 %, location 10 %, education 10 %)
- AI first-message drafts, scheduled follow-ups, inbox, per-account sending limits
- Assessments, booking links with .ics, decision log, analytics
- Workspaces, Microsoft SSO, MFA, 9 UI languages
- Any OpenAI-compatible LLM, tiered models, per-call metering, config fingerprint
- Written reasons behind every score (Match v2), cross-job re-matching
- Approval queue & bulk actions, per-job pause, Today queue
- ATS/CRM connection (one included, section 5)
- E-mail intake, Outlook calendar, candidate status page, AI notice to candidates

**SWITCHES SET AT GO-LIVE (FORMERLY BEHIND FEATURE FLAGS)**

- Match v2 rubric with written reasons `MATCH_V2_RUBRIC_API_ENABLED`
- Cross-job re-matching `MATCH_V2_CROSS_JOB_ENABLED`
- Approval queue & bulk actions `BULK_ACTIONS_ENABLED`
- Per-job pause `AUTOMATION_CONTROL_ENABLED`
- Today queue `VITE_WORK_QUEUE_ENABLED`
- ATS/CRM connector framework and sync `ATS_CONNECTIONS_API_ENABLED`, `ATS_SYNC_ENABLED`
- Outlook calendar push `booking.ms_graph`
- E-mail intake `EMAIL_INTAKE_ENABLED`
- Candidate portal `CANDIDATE_PORTAL_ENABLED`
- AI disclosure to candidates `AIA_DISCLOSURE_ENABLED`

**NOT INCLUDED**

- Further ATS/CRM connections ($2,500 each); write-back unless agreed
- Microsoft Teams
- DeepL machine translation
- Automatic merging of duplicates
- Client (customer) portal

All features in the middle column are built and are switched on in every installation at go-live, after the acceptance tests. The switch names are listed so your IT partner knows where they live; some features also have a matching `VITE_*` switch for the UI.

## 16 Limits and constraints

- **LinkedIn** is reached only through Unipile and the recruiter's own account. LinkedIn's own usage limits apply; Lynk adds per-account sending limits but cannot lift LinkedIn's.
- **AI output** depends on the model you configure. Scores are estimates to support a recruiter, not decisions; there is no automated rejection.
- **ATS/CRM:** read only or also write-back depends on the system and your plan (Personio, for example, is read-only); scope confirmed in writing before you buy, Bullhorn only with Bullhorn's approval.
- **Compose** is for development; several Dockerfiles may need small per-service adjustments for your registry and base images ( `DEVELOPMENT.md` ).

- **Scaling:** most services scale horizontally; `lynk-cron-api` runs as a single replica by design (scheduled jobs must run once).
- **Repository layout** keeps the historical snapshot directory names; two ODM layers (mongoengine, Beanie) coexist while the migration continues.
- **No performance guarantee** beyond the sizing estimate in section 9; a load-test workflow and a baseline exist ( `loadtest.yml` , `docs/perf-baseline-2026-08-27.md` ).

---

## 17 Before go-live we configure and verify

Each licensee runs its own instance in its own cloud account; we do not operate customer instances. Release status and deployment references are available on request. Every installation goes through this checklist during setup, together with your IT partner:

- [ ] Open your cloud or server account in your agency's name and give us temporary admin access; deploy your instance there from a tagged release of the handed-over repository; health endpoints green; staging smoke suite passes against it.
- [ ] Choose the LLM provider (EU hosting recommended) and **sign a data processing agreement (GDPR Art. 28)** with it; set the tier models; record them in `MODEL_CHANGELOG.md` ; confirm the fingerprint line in the logs.
- [ ] Enable service-to-service auth: `SERVICE_PRINCIPALS_ENABLED` and a strong `SERVICE_JWT_SECRET` on api, cron-api and msg-synk – **required for scheduled follow-up sending**.
- [ ] Generate fresh secrets (JWT ≥ 32 characters, field-encryption key, webhook secrets); store them in your secret manager only.
- [ ] MongoDB as a replica set (transactions); Redis with persistence and a password; private S3 bucket.
- [ ] Nightly encrypted backup running; backup private key kept in two separate places; **one restore test performed and documented**.
- [ ] Unipile account and webhook URL; each recruiter connects their LinkedIn account; sending limits reviewed.
- [ ] SMTP sender domain with SPF/DKIM; Microsoft SSO app registration if used.
- [ ] ATS/CRM connected via your own vendor account; agreed scope tested.
- [ ] Alert receivers and dashboard access set up; external uptime check pointed at your URLs.
- [ ] All features switched on after the acceptance tests; handover of all credentials; our temporary admin access removed.

## 18 Handover package

Common handover checklists for custom software ask for the source code with full history, an architecture overview, deployment and operations runbooks, environment templates, and a list of third-party components with their licences. This is what you receive:

| ITEM | CONTENT | STATUS |
|---|---|---|
| **Source code** | Git repository with full history: six services, `deploy/`, `infra/`, `tests/`, `scripts/` | exists |
| **Documentation** | This overview; `DEVELOPMENT.md`, `SECURITY.md`, 5 architecture decision records, observability, multi-tenancy, migrations, LLM provider docs, `MODEL_CHANGELOG.md` | exists |
| **Runbooks** | `DR-RUNBOOK.md`, deploy/promote/rollback, staging roll, backup/restore, Redis job recovery (`RQ-RECOVERY.md`) | exists |
| **Environment templates** | `.env.example` per service; Kubernetes secret key template – names only, no values | exists |
| **Deployment** | Dockerfiles, `docker-compose.yml`, Helm chart, ArgoCD applications, backup and GDPR CronJobs, monitoring rules and dashboards | exists |
| **CI** | 34 GitHub Actions workflows (portable to other CI systems with effort) | exists |
| **SBOM** | SPDX software bill of materials generated per main build (`sbom.yml`) | exists |
| **Open-source licence list** | Licence overview of all third-party components, derived from the SBOM | **to be produced before handover** |
| **Test data** | Seed script for two isolated test tenants (`tests/staging/ci_seed.py`) | exists |
| **Not included** | Third-party accounts and credentials (Unipile, LLM provider, SMTP, storage, Stripe, Microsoft): these are opened in your name and stay with you. | – |

Questions from your IT partner are welcome before any decision. Contact: HP Capital GmbH, Paul Hagler and Philipp Habring · hello@lynk-recruiting.com