Containerization Requirements Report (Automaspec)

Artifacts

- `Dockerfile` (multi-stage, pnpm, non-root runtime, exposes `3000`; pins `node:22-alpine`).
- `.dockerignore` (excludes build/cache artifacts; `.env` included for build/runtime).
- `docker-compose.yml` (builds `automaspec-web:local`, runs app on `3000`, consumes `.env`).
- `.env.example` (cloud libsql defaults; copy to `.env`).
- `README.md` (containerization section with run steps, env list, architecture, resources).

Build and Run

1) Copy `.env.example` to `.env` and fill `NEXT_PUBLIC_DATABASE_URL`, `DATABASE_AUTH_TOKEN`, AI keys as needed.
2) Build/run: `pnpm compose:up` (or `docker compose up --build`).
3) Stop: `pnpm compose:down`.

Requirement Checklist

- **Separate Dockerfile per service**: Frontend/serverless bundle has `Dockerfile`; database uses external cloud libsql (no image required).
- **Layered build**: Multi-stage (`base` → `deps` → `builder` → `runner`); dependencies cached; runtime is slim.
- **.dockerignore**: Present; excludes node_modules, build outputs, caches; keeps `.env` for build/run.
- **ENV for settings**: All config via env (`NEXT_PUBLIC_DATABASE_URL`, `DATABASE_AUTH_TOKEN`, AI keys, `VERCEL_URL`); no hardcoded secrets.
- **Volumes for persistence**: Database is external cloud libsql; persistence handled by provider (not local volume).
- **Ports**: `EXPOSE 3000` in image; Compose maps `3000:3000`.
- **Image size optimization**: Alpine base, pnpm, multi-stage, no dev deps in runner.
- **Temp/cache cleanup**: Dependency layer only installs runtime deps; no extra tools baked into runner.
- **Security**: Non-root user (`nextjs`); no secrets in image; `.env` supplied at build/run time.
- **Versioning**: Local tag `automaspec-web:local`; pin Node 22-alpine base.
- **Compose up**: `docker compose up` starts all services; `docker compose down` stops them.
- **Depends_on**: Not required because DB is external; app can start independently.

- **Networks**: Uses default bridge; app reachable on host port `3000`.
- **.env.example**: Provides sample values; `.env` consumed by Compose and included in build context.
- **Resource callout**: Targets <1 GB image; <500 MB compressed is expected for the app.
- **Non-root runtime**: `USER nextjs` in runner stage.

Compose

- Services: `app` only (expects external libsql). Use `.env` for connection details.
- Networks: default bridge.
- Restart policy: `unless-stopped`.

Environment Variables

- `NEXT_PUBLIC_DATABASE_URL`: cloud libsql endpoint (required).
- `DATABASE_AUTH_TOKEN`: libsql token (required).
- `OPENROUTER_API_KEY`, `GEMINI_API_KEY`: AI providers (optional unless using providers).
- `VERCEL_URL`: optional host for Better Auth trusted origins.

Resource Guidance

- App container: 1 CPU, 1.5 GB RAM baseline.

Architecture (containers)

```
[Host/Browser] -> 3000 -> [automaspec-web container] -> HTTPS -> [cloud libsql service]
```

Validation Checklist (how to verify)

- Build image: `docker compose build` succeeds without missing env errors.
- Run stack: `docker compose up` serves app on http://localhost:3000.
- Env wiring: runtime sees `NEXT_PUBLIC_DATABASE_URL` and `DATABASE_AUTH_TOKEN` via `env_file`.
- User: container runs as `nextjs` (non-root).
- Ports: `3000` exposed/mapped.
- Image size: check `docker images automaspec-web:local` (<1 GB target).
- Secrets: none hardcoded; `.env` supplied externally.