STARTUP

Inhaltsverzeichnis

docs/STARTUP.md

← Back to README

Goal

Run the study system locally (Next.js + LangGraph + PostgreSQL).

Prerequisites

- Node.js + pnpm
- Docker + Docker Compose

Step 1: Configure environment

Copy the template and fill required values:

```
# from repo root
copy .example.env .env
# or on mac/linux:
# cp .example.env .env
```

Minimum required (typical local setup):

- AUTH_SECRET
- OPENAI_API_KEY

- DATABASE_URL (App DB)
- LANGGRAPH_POSTGRES_URL (Checkpoint DB)
- DATASET_POSTGRES_URL (Dataset DB)
- NEXT_PUBLIC_ASSISTANT_ID (usually: dataAwareLLMSystem)

Details: `docs/CONFIGURATION.md`

Important:

- Do NOT commit `.env` (use `.example.env` as template).

Step 2: Put the dataset CSVs in place (IMPORTANT BEFORE DOCKER)

The dataset DB container (`postgres_esg`) imports CSV files **on first startup** using init SQL scripts:

- `db/esg/init/00_schema.sql` (creates schema/tables)
- `db/esg/init/10_load.sql` (COPY from `/data/...`)

In Docker Compose, `./data` is mounted into the container as **/data** (read-only).
That means the CSV files must exist locally **before** you start the containers, otherwise the import will fail.

Required files (minimum):

- `data/raw/companies.csv`
- `data/raw/indicator_metadata.csv`
- `data/processed/esg_indicators_postprocessed.csv`

You can obtain the dataset from OSF:

- https://osf.io/q2jpv/
- https://osf.io/q2jpv/files/osfstorage

Or request it from the author (see README contact).

Optional (sanity check):

```
# check that files exist (PowerShell)
ls .\data\raw\companies.csv
ls .\data\raw\indicator_metadata.csv
ls .\data\processed\esg_indicators_postprocessed.csv
```

Note on Data Insights UI:

- The file `data/dataset-manifest.json` maps dataset files to SQL tables.
- This mapping is used by the app/assistant to display “which dataset files were used” when showing data insights.

Step 3: Start database containers

```
docker compose up -d
```

Docker Compose starts three Postgres containers:

- `postgres_langgraph` on localhost:55432
  Purpose: LangGraph checkpointing/state
- `postgres_app` on localhost:55433
  Purpose: App DB (Prisma; participants/surveys/chat logs etc.)
- `postgres_esg` on localhost:55434
  Purpose: Dataset DB (ESG tables; CSV import via init scripts)

Step 4: Install dependencies

```
pnpm install
```

Step 5: Create Prisma Client

```
pnpm exec prisma generate
```

Step 6: Migrate App DB Schema

```
pnpm exec prisma migrate dev --name init
```

Step 7: Run in development mode

```
pnpm dev
```

What `pnpm dev` does (high level):

- runs `dataset:catalog:sync` (keeps dataset catalog metadata in sync)
- starts Next.js dev server
- starts LangGraphJS dev server (port 2024, no browser)

Step 8: Build Next.js application

```
pnpm build
```

Step 9: Run in production/study mode

```
pnpm start
```

Useful scripts (package.json)

Common:

- `pnpm dev`
  Next.js + LangGraph dev servers
- `pnpm start`
  Production start (after build) + dataset catalog sync
- `pnpm build`
  Next.js build

LangGraph:

- `pnpm langgraph:dev`
  LangGraphJS dev server (opens browser)
- `pnpm langgraph:dev:nobrowser`
  LangGraphJS dev server without browser
- `pnpm lg:db:setup`
  Setup Postgres schema for tavilyAgent checkpointing (LangGraph DB)
- `pnpm dataset:db:setup`
  Setup Postgres schema for dataAwareLLMSystem checkpointing (LangGraph DB)

Dataset:

- `pnpm dataset:catalog:sync`
  Sync dataset catalog metadata (based on `data/dataset-manifest.json`)

Prisma (App DB):

- See `prisma/commands.md` for local DB commands.

Troubleshooting

Dataset DB import fails on first startup:

- Verify required CSVs exist under `./data/...`
- Check container logs:

```
docker logs postgres_esg
```

LangGraph server not reachable:

- Verify `LANGGRAPH_API_URL` (default: http://localhost:2024)
- Verify `postgres_langgraph` is running and `LANGGRAPH_POSTGRES_URL` is correct
- Ensure schema variables exist:
  - LANGGRAPH_POSTGRES_SCHEMA_TAVILY
  - LANGGRAPH_POSTGRES_SCHEMA_DATA_AWARE

← Back to README