

All set up instruction can be found on **GitHub** page:

<https://github.com/MikeLiang2/Main-Capstone>

Capstone Project

A fullstack application with a Python FastAPI backend and a Vue 3 + Vite frontend.

Requirements

- **Python:** ≥ 3.13
 - **Node.js:** ≥ 22.17
 - **npm:** (comes with Node.js)
 - **Docker** and **Docker Compose** (for containerized setup)
-

Run the Project with Docker

Quick Start

If you're new to Docker, the easiest way to run the entire project is to execute the startup script:

```
./start.sh
```

This script handles building, migrating, and starting all the necessary containers.

Manual Docker Compose Commands

First Time Setup

Before running the containers manually, make sure the backend/alembic/versions directory exists. If it doesn't, create it:

```
mkdir -p backend/alembic/versions
```

Also, make sure Docker and Docker Compose are installed and running on your machine.

Then, from the project root directory, run:

```
docker compose up --build
```

This will:

- Build and start the backend FastAPI server on <http://localhost:8000>

- Build and start the frontend Vue app served via nginx on <http://localhost:5173>
- Start a PostgreSQL database container
- Start the database migration service to apply schema changes

Subsequent Runs (After Initial Build)

To start the containers without rebuilding, simply run:

```
docker compose up
```

Stop the containers

```
docker compose down
```

Remove the containers

```
docker compose down -v
```

Setup Instructions (Only if without Docker)

1. Backend Setup (Local development)

```
cd backend
python -m venv venv
# Activate virtual environment:
# Windows:
venv\Scripts\activate
# macOS/Linux:
source venv/bin/activate
pip install -r requirements.txt
```

2. Frontend Setup (Local development)

```
cd ../frontend
npm install
```

Run the Project Locally (without Docker)

1. Start Backend Server

In a new terminal:

```
cd backend
```

```
# Activate virtual environment:
# Windows:
venv\Scripts\activate
# macOS/Linux:
source venv/bin/activate
uvicorn app.main:app --reload
```

2. Start Frontend Dev Server

In another terminal:

```
cd frontend
npm run dev
```

Project Structure

```
Capstone/
├── backend/
│   ├── app/
│   ├── Dockerfile
│   ├── requirements.txt
│   └── ...
├── frontend/
│   ├── Dockerfile
│   ├── package.json
│   ├── src/
│   └── ...
├── dbm.sql
├── docker-compose.yml
├── package.json
└── README.md
```

Notes

- The backend runs on <http://localhost:8000>
- The frontend runs on <http://localhost:5173>

Validation (how to confirm the setup is successful).

All the deployment status will be reflected in the terminal. For detail check the terminal status.

For instance:

The following steps monitor the container build status in Docker.

```
[+] Running 6/6
✓ Network capstone_default Created
✓ Volume "capstone_pgdata" Created
✓ Container postgres_db Created
✓ Container migration Created
✓ Container backend Created
✓ Container frontend Created
```

Other start status will also posted in the terminal in different colors.

```
postgres_db | 2025-08-21 23:13:40.140 UTC [41] LOG: database system is shut down
postgres_db | done
postgres_db | server stopped
postgres_db |
postgres_db | PostgreSQL init process complete; ready for start up.
postgres_db |
postgres_db | 2025-08-21 23:13:40.213 UTC [1] LOG: starting PostgreSQL 15.13 on aarch64-unknown-linux-musl, compil
ed by gcc (Alpine 14.2.0) 14.2.0, 64-bit
postgres_db | 2025-08-21 23:13:40.213 UTC [1] LOG: listening on IPv4 address "0.0.0.0", port 5432
postgres_db | 2025-08-21 23:13:40.213 UTC [1] LOG: listening on IPv6 address ":::", port 5432
postgres_db | 2025-08-21 23:13:40.215 UTC [1] LOG: listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
postgres_db | 2025-08-21 23:13:40.217 UTC [59] LOG: database system was shut down at 2025-08-21 23:13:40 UTC
postgres_db | 2025-08-21 23:13:40.220 UTC [1] LOG: database system is ready to accept connections
frontend |
frontend | VITE v7.0.0 ready in 514 ms
frontend |
frontend | → Local: http://localhost:5173/
frontend | → Network: http://172.18.0.5:5173/
backend | INFO: Started server process [1]
backend | INFO: Waiting for application startup.
backend | Waiting for migration to create required tables... Attempt 1/10
migration | Connection to db (172.18.0.2) 5432 port [tcp/postgresql] succeeded!
migration | Postgres is up - continuing
migration | INFO [alembic.runtime.migration] Context impl PostgresqlImpl.
migration | INFO [alembic.runtime.migration] Will assume transactional DDL.
migration | INFO [alembic.autogenerate.compare] Detected added table 'process_categories'
migration | INFO [alembic.autogenerate.compare] Detected added index 'ix_process_categories_id' on '('id,')'
migration | INFO [alembic.autogenerate.compare] Detected added table 'roles'
migration | INFO [alembic.autogenerate.compare] Detected added index 'ix_roles_id' on '('id,')'
migration | INFO [alembic.autogenerate.compare] Detected added table 'users'
migration | INFO [alembic.autogenerate.compare] Detected added index 'ix_users_avatar' on '('avatar,')'
migration | INFO [alembic.autogenerate.compare] Detected added index 'ix_users_email' on '('email,')'
migration | INFO [alembic.autogenerate.compare] Detected added index 'ix_users_username' on '('username,')'
migration | INFO [alembic.autogenerate.compare] Detected added table 'processes'
migration | INFO [alembic.autogenerate.compare] Detected added index 'ix_processes_id' on '('id,')'
```

The services status (backend, frontend, database) can be viewed in Docker.

<input type="checkbox"/>	▼		capstone	-	-	-		:	
<input type="checkbox"/>			backend	09a91395905e	capstone-b	8000:8000 ↗		:	
<input type="checkbox"/>			frontend	249f85eee97e	capstone-fr	5173:5173 ↗		:	
<input type="checkbox"/>			migration	e6759aad4417	capstone-rr			:	
<input type="checkbox"/>			postgres_db	6580b7970561	postgres-1!	5432:5432 ↗		:	

Local deployment is easy, one lines of command should complete everything.

For cloud deployment:

Since different cloud service providers have different service policy and compacity, right now, auto deployment feature is only supported by Microsoft Azure.

For auto deployment on Microsoft Azure, run:

```
./deploy_azure.sh
```