Name: 卓益帆 ID: 114550822

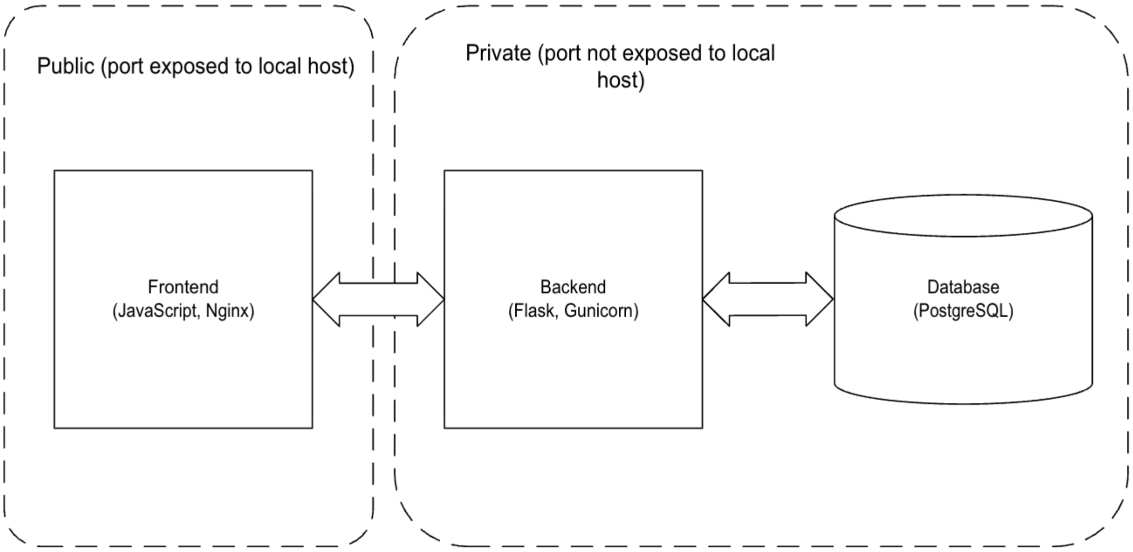
How do the 3 components work together?

The Frontend is exposed to the host (port 8080:80) and serves the HTML/JS files to the browser. Script.js makes API calls, then Nginx proxies these api calls to the backend service. The Backend runs inside its own container on port 8000 and exposes REST API endpoints under /api/names. It handles input validation and communicates with the database. And responds with JSON data (list of names, confirmation of added/deleted name, error messages). The database stores all data and persists data via a Docker volume. The backend container can reach the database container.

The request/response flow between containers.

When the user executes an action in the webpage, script.js sends a HTTP request. When Nginx sees the path /api/, Nginx proxies the request to the Backend. Flask then handles the HTTP request and executes the query in connection with the database. PostgreSQL updates the table then sends confirmation to Flask. Flask then sends a JSON response to script.js. The webpage then updates itself.

Diagram of the 3-tier architecture


Public (port exposed to local host)
Private (port not exposed to local host)
Frontend (JavaScript, Nginx)
Backend (Flask, Gunicorn)
Database (PostgreSQL)

Any challenges faced and how you solved them.

Docker compose up will start the containers in parallel. Hence, when I run the command, the backend container will try to connect with the database before the database is initialized. This led to race condition and I received an operational error.

I solved this by adding a Docker healthcheck to expose the database’s health, then I added a condition for the database's service to be healthy before running the backend container. This ensures that the database is ready to accept connections before starting the backend container.