Project information

Created on
February 23, 2026

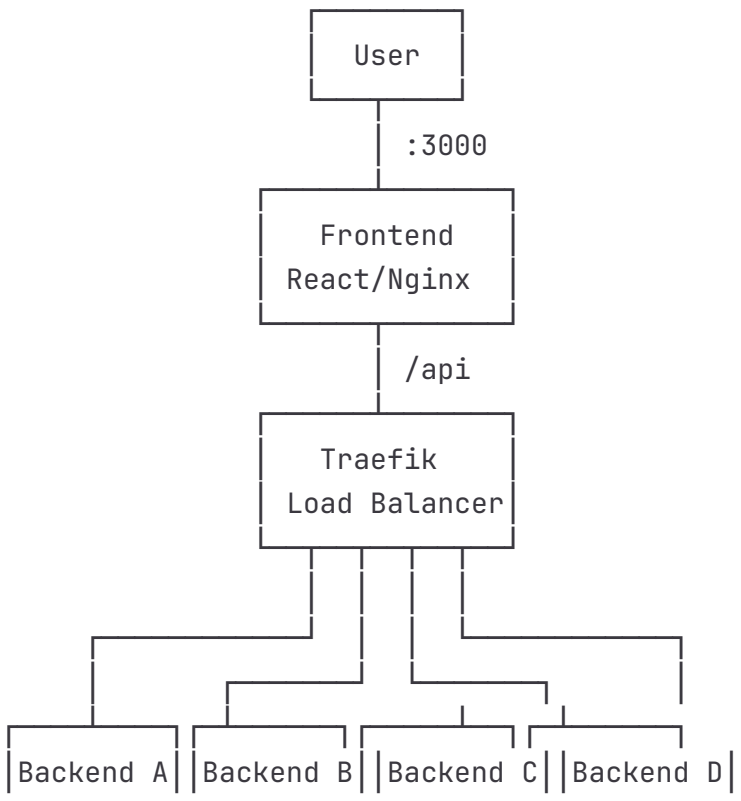
Name	Last update
backend	23 minutes ago
frontend	1 week ago
load-test	23 minutes ago
traefik	1 week ago
.gitignore	1 month ago
Arch.md	14 minutes ago
MILESTONE_01.md	1 month ago
MILESTONE_02.md	1 month ago
README.md	2 minutes ago
TODO	1 week ago
docker-compose.yml	23 minutes ago

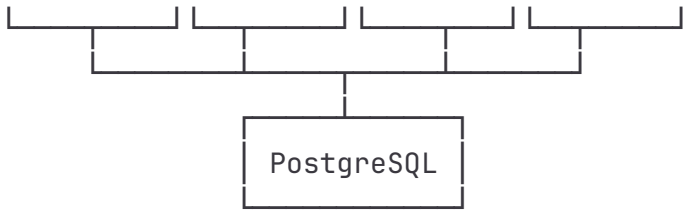
README.md

Distributed Stock Provider

A highly available, load-balanced stock data system built for high throughput and fault tolerance. It features 4 horizontally scaled backend instances behind a Traefik load balancer, real-time stock data from Yahoo Finance, dual-token JWT authentication, and a React frontend.

Architecture





Tech Stack

Layer	Technology
Frontend	React 19, Nginx, Axios
Backend	Node.js 24, Express 5
Database	PostgreSQL 18
Load Balancer	Traefik v3.6
Auth	JWT (access + refresh tokens), bcrypt
Caching	node-cache (in-memory, 90s TTL)
Orchestration	Docker Compose

Quick Start

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

Service	URL
Frontend	http://localhost:3000
Backend API	http://localhost:80/api
Traefik UI	http://localhost:8080

Development (Without Docker)

```
# Backend
cd backend
cp .env.example .env    # configure database connection, JWT secret, etc.
npm install
npm start

# Frontend (separate terminal)
cd frontend
npm install
npm start
```

Configuration

Key environment variables for the backend (see `backend/.env.example`):

Variable	Default	Description
`PORT`	`3000`	Server port
`JWT_SECRET`	—	JWT signing key (change in prod)
`ACCESS_TOKEN_EXPIRES_IN`	`15m`	Access token lifetime
`REFRESH_TOKEN_EXPIRES_IN`	`7d`	Refresh token lifetime
`DB_HOST`	`localhost`	PostgreSQL host

Variable	Default	Description
`DB_PORT`	`5432`	PostgreSQL port
`DB_NAME`	—	Database name
`DB_USER`	—	Database user
`DB_PASSWORD`	—	Database password
`INSTANCE_NAME`	—	Backend instance identifier

API Endpoints

Public

Method	Endpoint	Description
GET	`/health`	Health check
GET	`/api/instance`	Current instance info

Authentication

Method	Endpoint	Description
POST	`/api/auth/register`	Register (username, password)
POST	`/api/auth/login`	Login
POST	`/api/auth/refresh`	Refresh access token
POST	`/api/auth/logout`	Logout & revoke token

Protected (Bearer token required)

Method	Endpoint	Description
GET	`/api/stocks`	Live stock prices (Yahoo Finance)
GET	`/api/stocks/:symbol`	Single stock price
GET	`/api/stocks/cached/all`	All cached prices from DB
GET	`/api/heartbeat`	Authenticated health check

Tracked symbols: AAPL, GOOGL, MSFT, AMZN, TSLA, META, NVDA, BTC-USD, ETH-USD

Key Features

- **High Availability** — 4 backend instances with automatic failover (30s detection via Traefik health checks)
- **Multi-Layer Caching** — In-memory cache (90s TTL), DB query cache (5s TTL), batched writes (50-item buffer / 1s flush)
- **Dual-Token Auth** — Short-lived access tokens (15m) + long-lived refresh tokens (7d) with automatic frontend renewal
- **Connection Pooling** — PostgreSQL (20 max clients/instance), Traefik (200 idle connections/host)
- **Rate Limiting** — 1,000 RPS average, 2,000 burst (Traefik middleware)
- **Response Compression** — gzip for responses over 1KB

Load Testing

Load test scripts and the `hey` benchmarking tool are included in `load-test/`:

```
# Linux
./load-test/run-load-test.sh

# Windows
load-test\run-load-test.bat
```

Tests target `/api/stocks/cached/all` with auto-obtained auth tokens. Default test credentials: `testuser` / `testpass123`.

Performance

Under normal load (from benchmarking):

- **Response time:** 100–300ms
- **CPU usage:** <10% per instance
- **Memory:** ~150MB per backend container
- **DB query time:** <10ms for cached lookups

Database Schema

```
users (id, username, password, created_at)
stock_prices (id, symbol, price, change_percent, last_updated)
refresh_tokens (id, user_id, token, expires_at, created_at, revoked)
```

Project Documentation

- MILESTONE_01.md — Architecture planning & tech stack decisions
- MILESTONE_02.md — Initial implementation details
- Arch.md — Implementation progress & performance observations