DevOps, Software Evolution and Software Maintenance

May 2021

Course code: KSDSESM1KU

Project assignment by:

name	email
Vlad Croitoru	vlcr@itu.dk
Anders Friis Kaas	anfk@itu.dk
Rasmus Dilling Møller	rdmo@itu.dk
Sean Wachs	sewa@itu.dk
Jonas William Gohn	jgoh@itu.dk

System

Overview

The core of the Minitwit application is written in Go. It is split into a frontend system that responds to HTTP requests coming in on destination port 8080 and a backend API that responds to HTTP requests received on port 8081. The frontend system responds to client GET requests with properly formatted HTML responses and is what users would interact with when they visit our website. The backend system responds only with raw JSON and is what the simulator interacts with. The two systems share a PostgreSQL database that stores user information, user following relationships, messages and the 'latest' value. The systems interface with the database through the ORM library GORM instead of through raw SQL queries. The systems are instrumented with Prometheus metrics which allows us to monitor them using Grafana.

The whole system is deployed to three nodes on DigitalOcean. One node acts as a manager and load balancer and the other two are replicated worker nodes. The nodes are managed with Docker Swarm. The runtime environment of the worker nodes are Docker containers running Ubuntu v. 20.4.

Dependencies

On the lowest level, our Go applications depend on the following libraries:

- fmt - used for string formatting and basic I/O.
- errors - useful functions for error handling and error manipulation .
- os - used to interface with the operating system. We use this to obtain environment variables.
- encoding/json - used to encode code objects as JSON strings.
- strconv - conversion to and from string representations
- strings - string manipulation
- time - used for function execution timing.
- net/http - used to listen for HTTP requests and serve responses.
- log - used for basic logging of our application. We log system errors and HTTP requests and responses.
- gorilla/mux - used to implement request routing. We use mux to direct specific requests to appropriate handler functions.
- gorilla/sessions - provides a cookie system.
- godotenv - used to read environment variables from a file.
- gorm - Gorm is the ORM library that we use to abstract the database to objects in code.
- postgres - used by Gorm to interact with the database.
- prometheus - prometheus is our metrics system that stores information about numbers of requests per endpoint as well as function execution times.
- promhttp - this submodule of prometheus is used to expose an endpoint called /metrics that is used by Grafana.
- bcrypt - used to hash passwords.
- html/template - used for generating valid HTML from templates