Distributed Software Architecture Lab 3 Report

# GOALS

* Orchestrate multiple microservices with Docker Compose
* MongoDB basics
* Reliability concepts: Retry & Circuit-Breaker
* Availability: Data Caching & Request Queuing

# SOURCE CODE

<https://github.com/tvph1996/third_lab>

# SYSTEM SETUP

## Supports 4 methods

* AddItem (POST): Duplicate 'id' and 'name' is not allowed
* GetItem (GET): Search by 'id' gets single item, search by 'name' gets item stream (wrap around)
* UpdateItem (PUT): Change name (no duplicate) of an item by 'id'
* DeleteItem (DELETE): Remove item by 'id'

## REST-service

* Provides human response to CURL request
* Uses FastAPI, continue from Lab 2
* Calls gRPC-service using Environment Variable in Docker Compose
* Retry & Circuit-Breaker
  + Design as the lab guideline, the only differences:
    - 2 retries after 1 sec & 2 sec to avoid IO display issue with minimal time
    - Shorten reset timeout to 6 sec for easier testing
* Supports: Request Queuing
  + When gRPC-service or MongoDB down, requests are put in a queue and will be processed when services are healthy again.
  + This is done with a process checking the availability of required services by sending dummy request periodically in the background every 15 seconds.

## gRPC-service

* Rule check for methods
* Calls MongoDB using Environment Variable in Docker Compose
* Data Caching of MongoDB using python dictionary

## MongoDB

* Data is stored locally using Docker Volume
  + Configured in Docker Compose

# OUTPUT

# REFLECTION