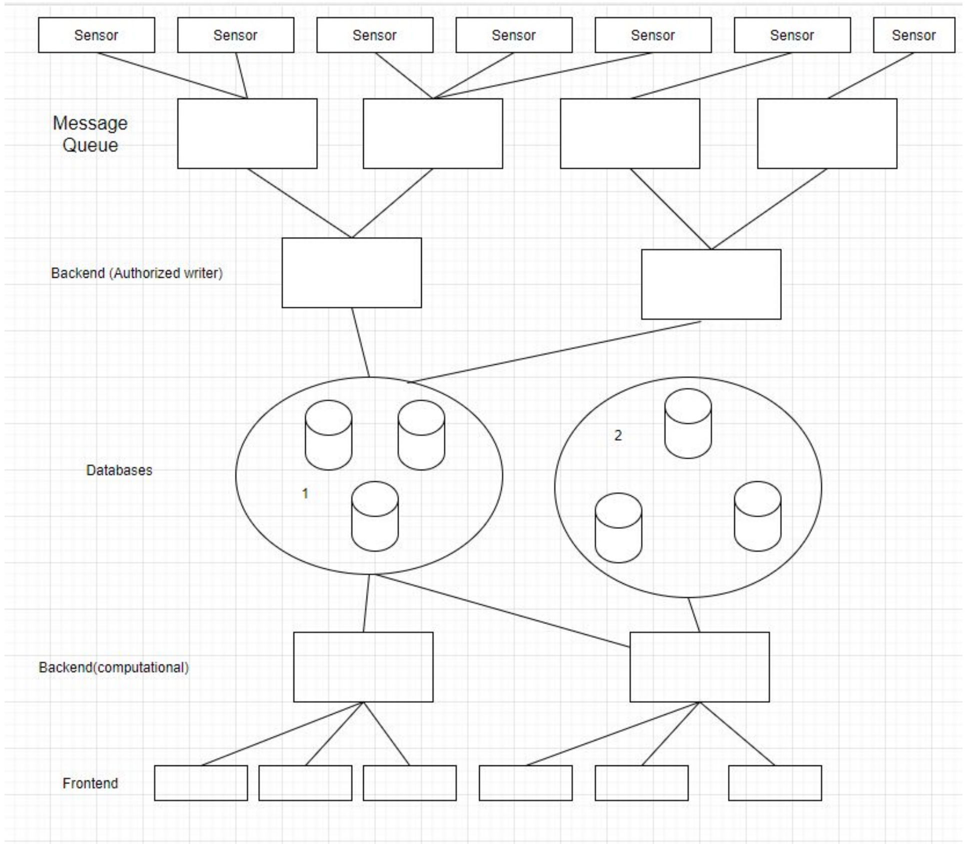
Web and cloud computing

Group 03

Architecture



- Where every box lives in a Docker container with an image that's pulled from a container registry
- Every layer is replicated
- Every connection can be load balanced

Sensors

- Programming language: Python
- Reason of picking technology: OOP constructs that allows for simple development of multiple types of sensors
- Producers that write different types of simulated sensor data to Message Queue

Message Queue

- Technology: Apache Kafka
- Reason for picking technology: Distributed message queue with persistent storage and fault tolerance
- Serves as an intermediary between sensors and authorized backend consumers
- Maintained by Zookeeper

Backend

- Programming language + Framework: NodeJS + Express (w/TypeScript) + (Go or Rust or Spark) for computation-heavy routes
- Reason of picking technology: Easy and fast framework that's tried and tested + async await + promises + callbacks functionality
- From all consumers of Kafka, Node.JS ones will write to the database
- For data preprocessing, services written in Go or Rust might be used as they are faster (not necessary, but nice to have)
- For data modelling Spark might be used if a use case arises
- Socket.io for chat functionality
- JWT for user authentication
- Encryption library: bcryptjs

Databases

- Apache CassandraDB
- Reason: Fast read writes for key-value pairs; ideal for sensor data due to SStables with consistent hashing
- Can be distributed and replicated for backups and fault-tolerancy (available and partition tolerant, but not consistent “AP”)

- MongoDB
- Reason: Fast development time because it is a Document-based NoSQL db
- Consistency is key when dealing with sensitive administrative information
- Can (also) be distributed, replicated, sharded with master/slave strategy

Deployment

- Technology: Docker + docker-compose + Kubernetes + Github Actions + DockerHub
- Reason of picking technology: Everything lives on Docker, managed locally during development by docker-compose, then deployed on multiple machines using K8s, automated through a CI/CD pipeline (Github Actions + DockerHub as a container registry)
- Automation testing through Github Actions (dependent on programming language)
- Deployment platform: Microsoft Azure / GCP (arbitrary pick, they give money)
- Load balancer from K8s (although NGINX could also be used)

Front-end

- Framework: Javascript + Vue.js (enforces SPA, modern, easy to learn)
- CSS Framework: Buefy(Bulma) / Bootstrap
- Webserver: NGINX
- Visualization tools: Prometheus + Grafana
- Requests library: Axios

Questions

- Should we use JQuery if we use Vue.js and can handle everything within Vue.js?
- JWT vs SciTokens - Worth the trouble / is there any use case?
- Can we use Prometheus + Grafana?
- Do we get bonus points if we use scala in the preprocessing pipeline (potentially even with Spark)?