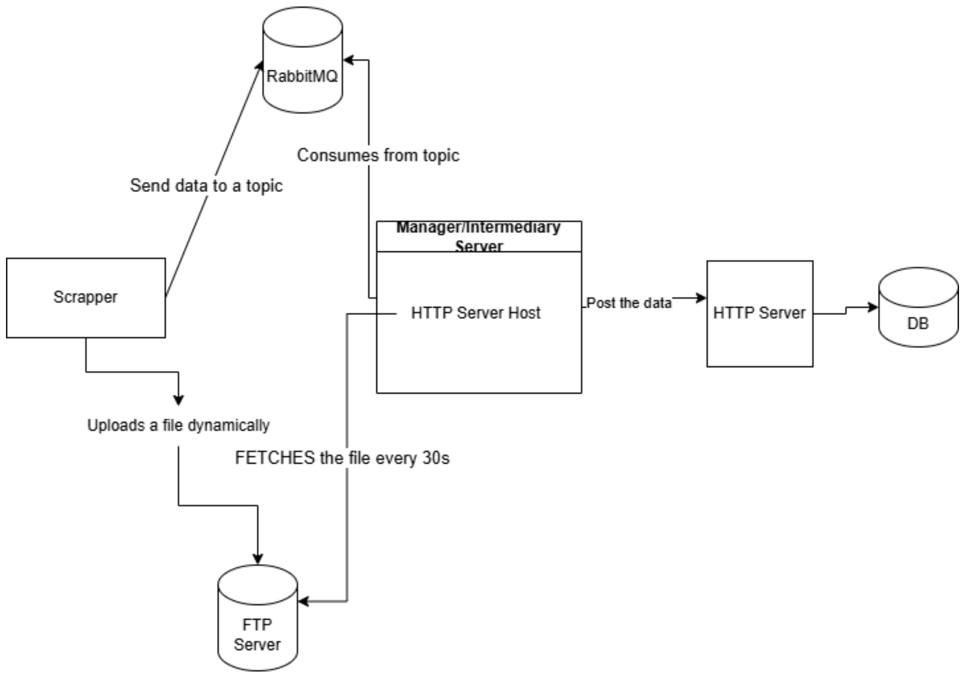
PR Lab 3

Deadline: 3 weeks; until 13.12.2024;

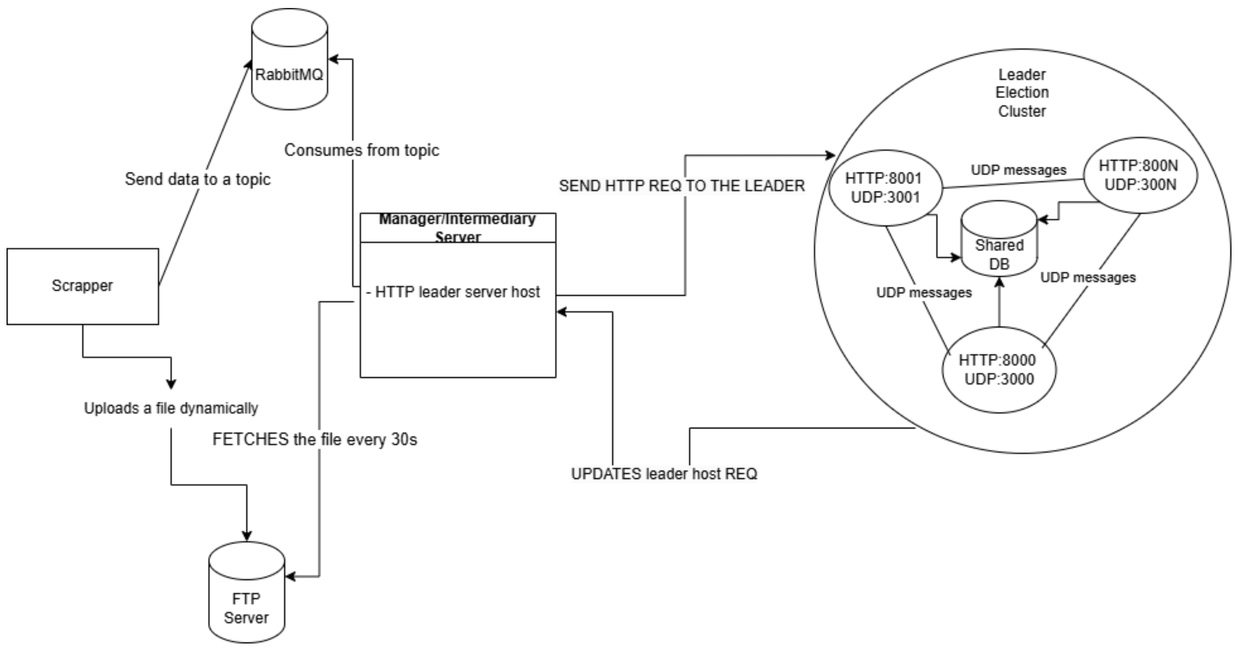
Grade	Conditions
1 - 4	Study the Leader Election process in the RAFT Consensus Algorithm. [1]
5	Implement a simulation of the Leader Election process from the RAFT Consensus Algorithm using the User Datagram Protocol (via UDP sockets). You may run the servers (acting as nodes) asynchronously, in threads, in Docker, or start them with a script, but it must be a functional simulation with logging/printing for leader election, including handling received heartbeats.
6	Use RabbitMQ (or any message broker). You can utilize Docker Compose to run the RabbitMQ image.[2] Connect your scraper from LAB1 (as the publisher to RabbitMQ server) and your webserver from LAB2 using **a manager/intermediary server** acting as a Consumer. The consumer will read from a topic/queue and make POST requests to your LAB2 webserver with the necessary data.
7	Implement a separate thread on **your manager/intermediary server** to fetch a file from an FTP server every 30 seconds. An FTP server Docker image is provided here[3]. The thread should then send the file as a multipart request to your LAB2 webserver. To populate or update the FTP server with the file, save the processed information (after map/filter/reduce in LAB1) into a file dynamically and upload it to the FTP server.
8	Similar to Task 5 and implement the Leader Election process on multiple copies of your web server. All web servers will operate on the same database. This means that you should start your servers, and they must elect a leader over the UDP interface. Similar to Task 6 from LAB2, start your UDP handler in a thread. Once the leader election process finishes, the leader should make a request to your **intermediary/manager server** to update the host to which web server requests are redirected.
9	Dockerize every component of your lab, except the scraper (as it will serve as a live test client during your presentation).
10	Separately implement a simple SMTP client and demonstrate successfully sending an email.

Tasks 9 and 10 are interchangeable; the order of completion does not matter. Additionally, you may skip Task 5 if you implement Task 8.

If you do untill 7, inclusive:



If you do more than 7:



[1] RAFT Algorithm and Leader Election articles:

– https://raft.github.io/

– https://sulavpanthi.medium.com/raft-a-consensus-algorithm-for-distributed-systems-67f46c6ed030

– https://medium.com/@govinda.attal/raft-consensus-leader-election-with-golang-89bfdbd471cb

– https://medium.com/geekculture/raft-consensus-algorithm-and-leader-election-in-mongodb-vs-coachroachdb-19b767c87f95

[2] docker-compose.yaml file for RabbitMQ

```yaml
version: '3.8'
services:
    rabbitmq:
        image: 'rabbitmq:3-management-alpine'
        container_name: iepure_MQ
        ports:
          - 5672:5672
          - 15672:15672
```

[3] docker-compose.yaml file for a FTP server

```yaml
version: '3.8'
services:
  ftp_server:
    image: stilliard/pure-ftpd:hardened
    container_name: ftp_server
    ports:
      - "21:21"
      - "30000-30009:30000-30009" # Passive ports for FTP
    environment:
      FTP_USER_NAME: testuser          # FTP username
      FTP_USER_PASS: testpass          # FTP password
      FTP_USER_HOME: /home/testuser    # FTP user home directory
    volumes:
      - ./ftp_data:/home/testuser      # Map local directory to FTP home
```