FACULTY OF AUTOMATION AND COMPUTER SCIENCE

COMPUTER SCIENCE DEPARTMENT

DISTRIBUTED SYSTEMS

Laboratory Assignment 2

ONLINE ENERGY UTILITY PLATFORM

Student: Horvath Ariana-Cristine
Group: 30441
Teacher assistant: Oana-Andreea Marin

Table of Contents

1. Requirements

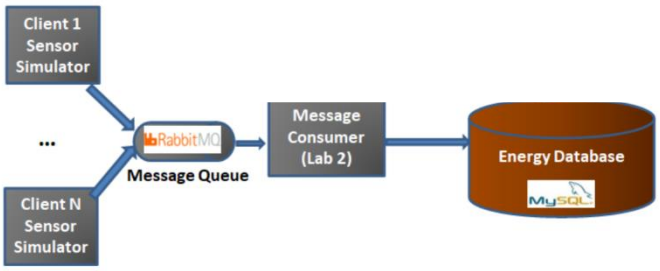
Implement a component for Assignment 1 application based on a message broker middleware that gathers data from the smart metering devices, pre-processes the data to compute the hourly energy consumption and stores it in the database.

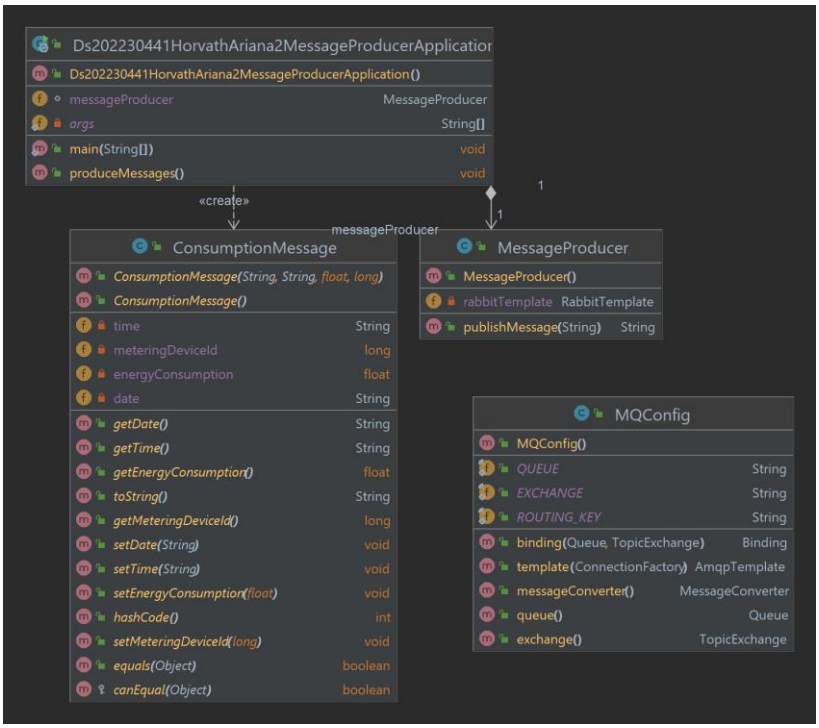
A Smart Metering Device Simulator module will be the Message Producer. It will simulate a sensor by reading energy data from a file (sensor.csv - one value at every 10 minutes) and sends data in the form < timestamp, device_id, measurement_value > to the Message Broker (i.e., the queue). The timestamp is taken from the local clock, the measurement_value is read from the file and represents the energy measured in kWh, and the device_id is unique to each instance of the Smart Metering Device Simulator and corresponds to the device_id of a user from the database (as defined in Assignment 1). The sensor simulator should be developed as a standalone application (i.e., desktop application).

A Message Consumer application will pre-process the data to compute the total hourly energy consumption and stores it in the database. If the computed total hourly energy consumption exceeds the smart device maximum value (as defined in Assignment 1) it notifies asynchronously the user on his/her web interface.

- The message broker allows Smart Metering Device Simulator to act as messages producer and send data tuples in a JSON format.
- The message consumer component of the system processes each message and notifies asynchronously using WebSockets the client application.
- Use the following technologies: RabbitMQ, WebSockets.

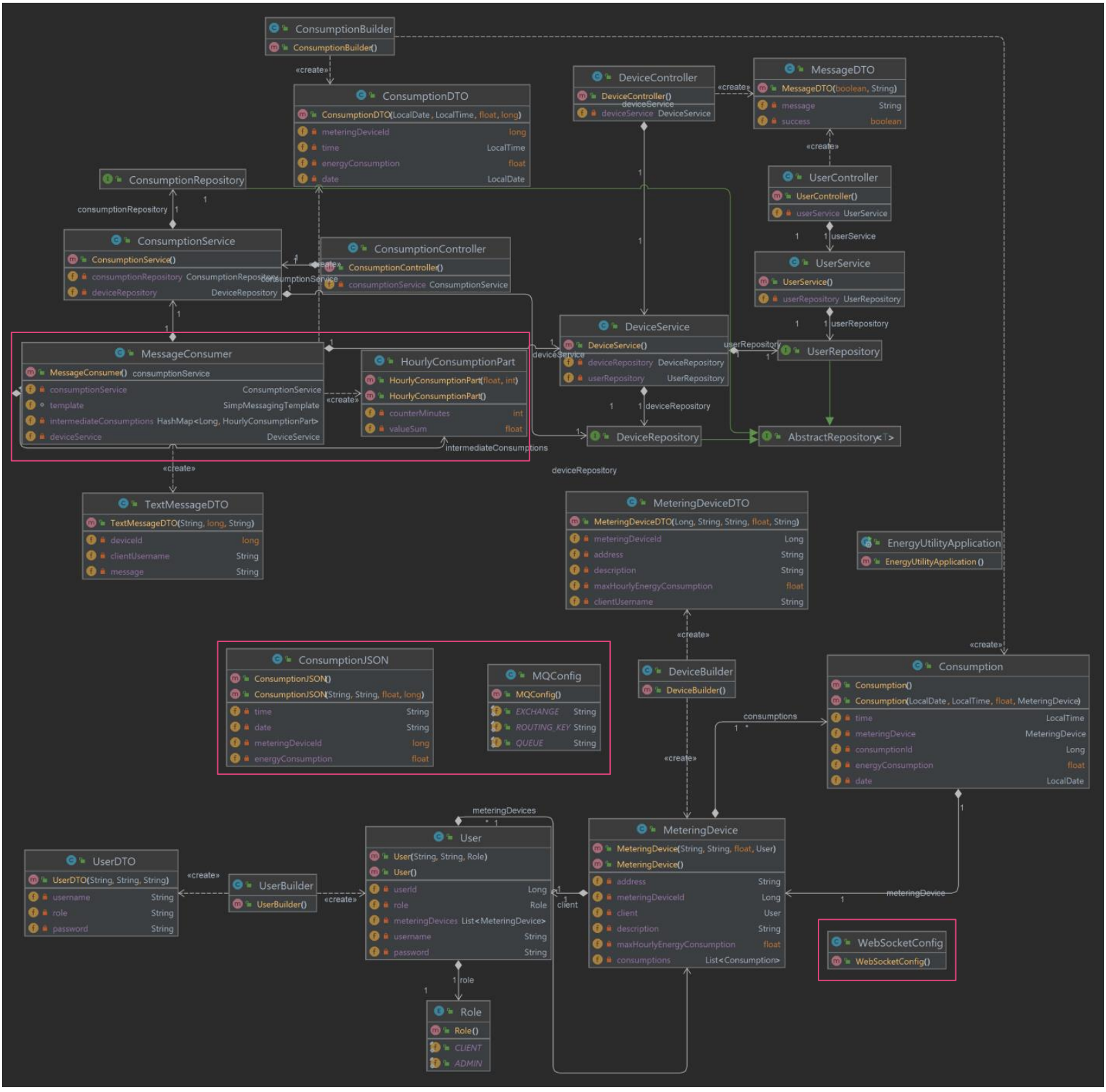
2. Design



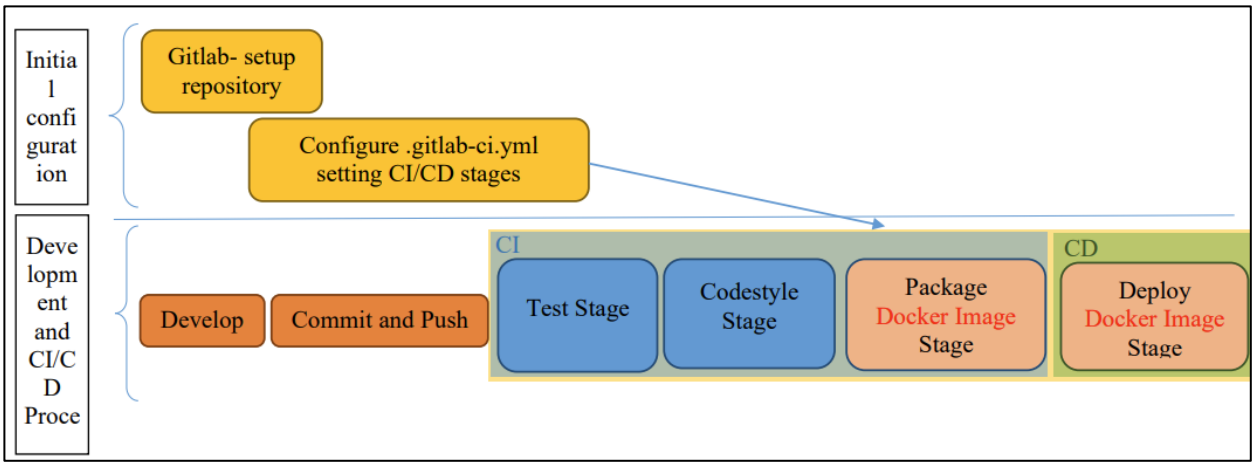


2.1 Message Producer

2.2 Message Consumer



3. CI/CD Deployment



3.1 Configuration Files

3.1.2 Backend

- docker-compose.yml

```
version: "3.4"

services:
   api:
      image: "containerregistryarianahorvath30441.azurecr.io/arianahorvath30441backend:latest"
      domainname: "arianahorvath30441backend"
      ports:
        - 8080:8080
      environment:
        SPRING_RABBITMQ_HOST: rabbitmq
        DB_IP: demo-db
        RABBIT_IP: demo-rabbit
        DB_PORT: 3306
        DB_USER: root
        DB_PASSWORD: root123
        DB_DBNAME: energy_utility
      deploy:
        resources:
          reservations:
            cpus: '1'
            memory: 2G

   db:
      image: "containerregistryarianahorvath30441.azurecr.io/db:latest"
      environment:
```

```
      MYSQL_DATABASE: energy_utility
      MYSQL_ROOT_PASSWORD: root123
      MYSQL_HOST_AUTH_METHOD: trust
    domainname: "arianahorvath30441backend"
    ports:
      - 3306:3306
    deploy:
      resources:
        reservations:
          cpus: '1'
          memory: 2G
  rabbitmq:
    image: "containerregistryarianahorvath30441.azurecr.io/rabbitmq:latest"
    domainname: "arianahorvath30441backend"
    ports:
      - 15672:15672
      - 5672:5672
    deploy:
      resources:
        reservations:
          cpus: '1'
          memory: 2G
```

- azure-pipelines.yml

```
# Docker
# Build and push an image to Azure Container Registry
# https://docs.microsoft.com/azure/devops/pipelines/languages/docker

trigger:
- master

resources:
- repo: self

variables:
  # Container registry service connection established during pipeline creation
  dockerRegistryServiceConnection: '5e9f645c-6548-46fc-bb7f-25853a03db67'
  imageRepository: 'arianahorvath30441backend'
  containerRegistry: 'containerregistryarianahorvath30441.azurecr.io'
  dockerfilePath: '$(Build.SourcesDirectory)/energyUtility/Dockerfile'
  tag: 'latest'

  # Agent VM image name
  vmImageName: 'ubuntu-latest'

stages:
- stage: Build
  displayName: Build and push stage
```

```
jobs:
- job: Build
  displayName: Build
  pool: 'local'
  steps:
  - task: Docker@2
    displayName: Build and push an image to container registry
    inputs:
      command: buildAndPush
      repository: $(imageRepository)
      dockerfile: $(dockerfilePath)
      containerRegistry: $(dockerRegistryServiceConnection)
      tags: |
        $(tag)
```

3.1.2 Frontend

- docker-compose.yml

```
version: "3.4"

services:

  react:
    image: "containerregistryarianahorvath30441.azurecr.io/arianahorvath30441frontend:latest"
    domainname: "arianahorvath30441frontend"
    ports:
      - 80:80
    deploy:
        resources:
          reservations:
            cpus: '1'
            memory: 2G
```

- azure-pipelines.yml

```
# Docker
# Build and push an image to Azure Container Registry
# https://docs.microsoft.com/azure/devops/pipelines/languages/docker

trigger:
- master

resources:
```

```
- repo: self

variables:
  # Container registry service connection established during pipeline creation
  dockerRegistryServiceConnection: 'c2d2c1a7-db5d-49c1-a26d-982391bf363e'
  imageRepository: 'arianahorvath30441frontend'
  containerRegistry: 'containerregistryarianahorvath30441.azurecr.io'
  dockerfilePath: '$(Build.SourcesDirectory)/Dockerfile'
  tag: 'latest'

  # Agent VM image name
  vmImageName: 'ubuntu-latest'

stages:
- stage: Build
  displayName: Build and push stage
  jobs:
  - job: Build
    displayName: Build
    pool: 'local'
    steps:
    - task: Docker@2
      displayName: Build and push an image to container registry
      inputs:
        command: buildAndPush
        repository: $(imageRepository)
        dockerfile: $(dockerfilePath)
        containerRegistry: $(dockerRegistryServiceConnection)
        tags: |
          $(tag)
```