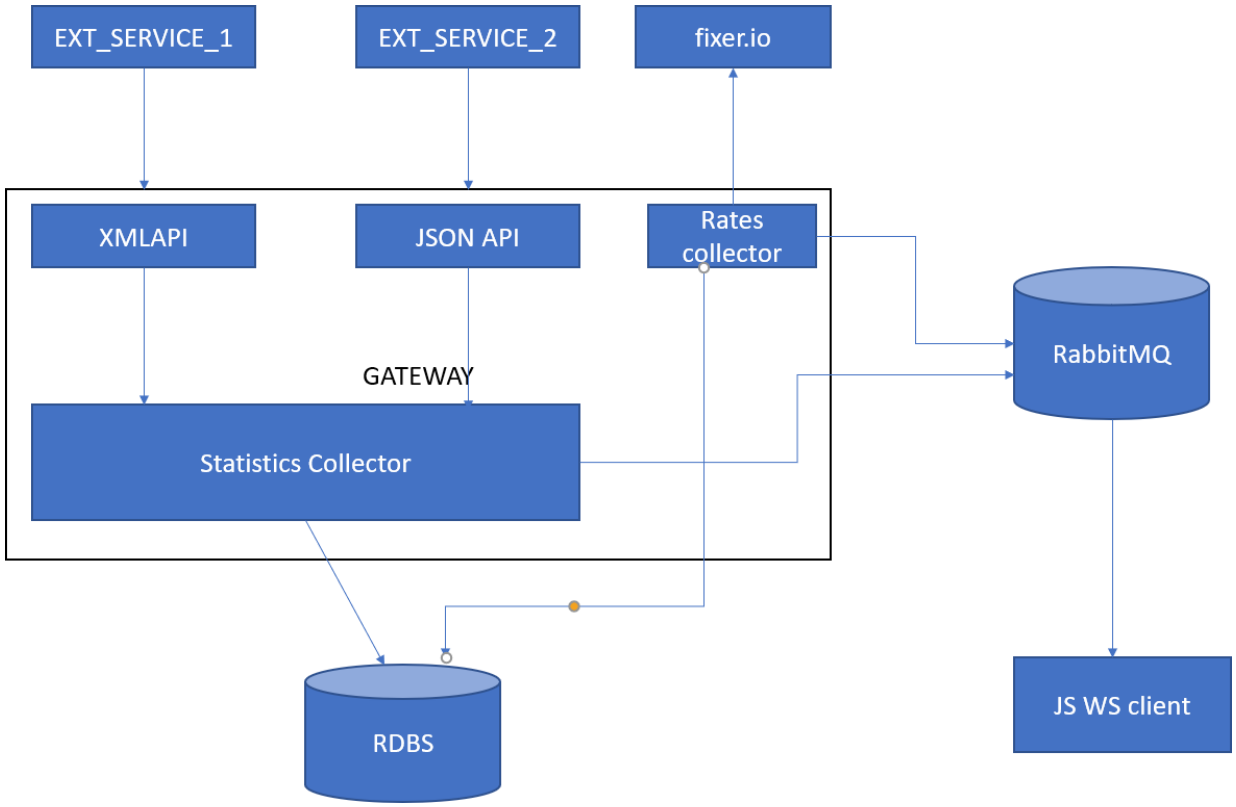
ЗАДАНИЕ И РАЗЯСНЕНИЯ

Цел на заданието

Да се създаде приложение фасада (наричано GATEWAY), което да предоставя данни за валути на различен тип клиенти.

Схема

Следната диаграма илюстрира инфраструктурата на приложението



Функционална спецификация

Системата се състои от следните компоненти:

1. Събира текущи данни за валутите предоставяни от https://fixer.io/, които да съхранява в релационна база данни. Обновяването на данните трябва да се случва на предефиниран интервал от време, посредством конфигурация

2. Предоставя две публични REST API за външни сервизи EXT_SERVICE_1 и EXT_SERVICE_2 работещи съответно с Content-type application/json и application/xml. Връщаната информация съдържа данни от fixer.io, като структурата на респонса е по твой избор, но в съответствие с Content-type поддържан от външния сервиз. Следват подробности и примери:

   JSON API: Поддържа два ендпойнта, които приемат следните POST рикуести:

   /json_api/current

   ```
   {
    "requestId": "b89577fe-8c37-4962-8af3-7cb89a245160",
    "timestamp": 1586335186721, // UTC
    "client": "1234",
   “currency”:”EUR”
   }
   ```

   Обработката на този рикуест изисква проверка за дублиране на рикуест с даденото id. В случай на дублиране системата връща грешка. Полето client, идентифицира краен клиент. При нормално изпълнение на рикуеста, респонса съдържа последните постъпили в системата данни за съответната валута

   /json_api/history

   ```
   {
    "requestId": "b89577fe-8c37-4962-8af3-7cb89a24q909",
    "timestamp": 1586335186721,
    "client": "1234",
   “currency”:”EUR”,
    "period": 24
   }
   ```

   Този рикуест изисква отново проверка за дублиране на рикуест. Респонса съдържа списък с натрупаните данни, за съответната валута през зададения период. Времевия интервал се интерпретира като часове и е цяло число.

XML API: Предоставя единствен ендпойнт за пост рикуести, които могат да са в следния формат:

/xml_api/command:

- За текущи данни

```
<command id="1234" >
        <get consumer="13617162" >
                <currency>EUR</currency>
        </get>
</command>
```

- Статистика за период

```
<command id="1234-8785" >
        <history consumer="13617162" currency="EUR" period="24" />
</command>
```

Атрибутът id в тага command, идентифицира уникално рикуеста. Атрибутът consumer, да се интерпретира като ид на краен потребител. Проверка за дублиране на рикуести също трябва да се имплементира.

3. Да събира **унифицирана** статистическа информация (service name/id - EXT_SERVICE_X, request id, time (UTC), end client id) в релационна база данни, относно постъпилите рикуести от EXT_SERVICE_1 и EXT_SERVICE_2

4. Да препраща през уеб сокет (RabbitMQ), унифицираната информация за постъпващи рикуести. Името на exchange е предефинирано в конфигурация. Сокет клиента не е обект на това задание.

Технологичен стек

Разполагаш е Java 8+, Spring boot 2.4+, Hibernate, Redis, RabbitMQ, Postgres/MySQL, Maven. Напълно приемливо е да използваш Hibernate за генериране на дб схема. Особено важно е да използваш максимално възможностите, които предоставя Spring, т.е. да не включваш странични библиотеки, ако Spring предоставя решение.

Имплементацията трябва да се направи с презумпцията, че това е силно натоварена система откъм брой рикуести за единица време, постъпващи от EXT_SERVICE_X.

Изисква се да осигуриш достъп до git репозитори по твой избор.