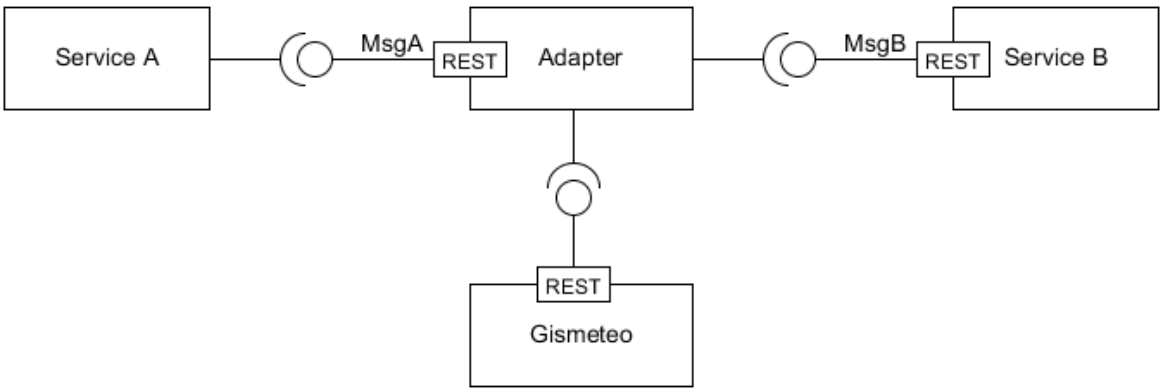
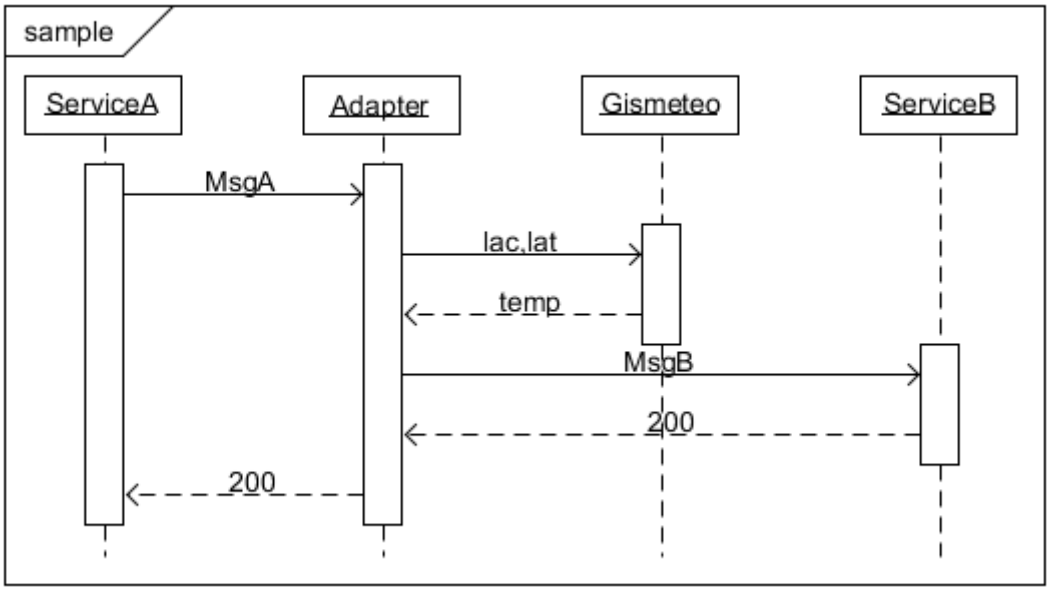
Тестовое задание необходимо выполнить с использованием стека: Apache Camel (ver. 3+), Spring Boot (ver. 2+)

Написать **только** микросервис «Adapter», который принимает сообщение из «Service A», производит преобразования, описанные ниже, и передает его в «Service B».

Ниже приведена схема взаимосвязей:



Ниже приведена схема взаимодействия:



1. Коммуникация между всеми микросервисами осуществляется с использованием архитектурного стиля REST. REST методы «Adapter» и «Service B» при успехе возвращают статус 200 с пустым телом. В рамках данной задачи не успешным ответом «Service B» будем считать любой ответ со статусом отличным от 200. Формат тела не успешного ответа «Adapter» в рамках данной задачи не специфицирован.

2. Формат сообщений, отправляемых «Service A» определен в формате RAML и имеет следующий вид:

```
MsgA:
 properties:
  msg:
   type: string
   description: Некое текстовое сообщение
   required: true
  lng:
   type: string
   enum: [ru, en, es]
   description: Язык сообщения
   required: true
  coordinates:
   properties:
    latitude:
     type: string
     description: Широта
     required: true
    longitude:
     type: string
     description: Долгота
     required: true
 example:
  {
   "msg": "Привет",
   "lng": "ru",
   "coordinates": {
    "latitude": "54.35",
    "longitude": "52.52"
   }
  }
```

3. Требуется обрабатывать сообщения только с признаком "lng": "ru" и, используя координаты, обогащать сообщение данными из сервиса погоды. Сообщения, не прошедшие условия фильтрации – игнорировать. Если сервис погоды недоступен – считать это ошибкой.

4. Одним из возможных форматов API сервиса погоды можно предложить формат Gismeteo (https://www.gismeteo.ru/api/). Реальный вызов сервиса погоды не обязателен, для разработки вполне достаточно использования MockRestServiceServer. Стоит учесть, что в дальнейшем список поддерживаемых сервисов погоды с их форматами будет расширяться, и возможно варианты сервисов будут определяться в момент развертывания сервиса адаптера.

5. Формат сообщений принимаемых сервисом «B»:

```
MsgB:
  properties:
    txt:
      type: string
      description: Некое текстовое сообщение
      required: true
    createdDt:
      type: datetime
      description: Дата формирования сообщения
      required: true
      format: rfc3339
    currentTemp:
      type: integer
      description: Температура по Цельсию
      required: true
  example:
    {
      "txt": "Привет",
      "createdDt": "2020-06-10T10:15:30Z",
      "currentTemp": "28"
    }
```

6. Добавить обработку ошибок при получении пустого сообщения из сервиса “A”. Пустым сообщением считать сообщение, не содержащее ни одного символа в поле “msg”

7. Написать компонентные тесты для проверки кода. Границы компонентного теста сформулированы тут: https://martinfowler.com/articles/microservice-testing/#testing-component-in-process-diagram. Примеры тестирования маршрутов Apache Camel можно найти тут: https://camel.apache.org/manual/latest/testing.html

8. Использовать gradle для сборки проекта

9. Настройка Spring с помощью Java-аннотаций