Home Task – Travix

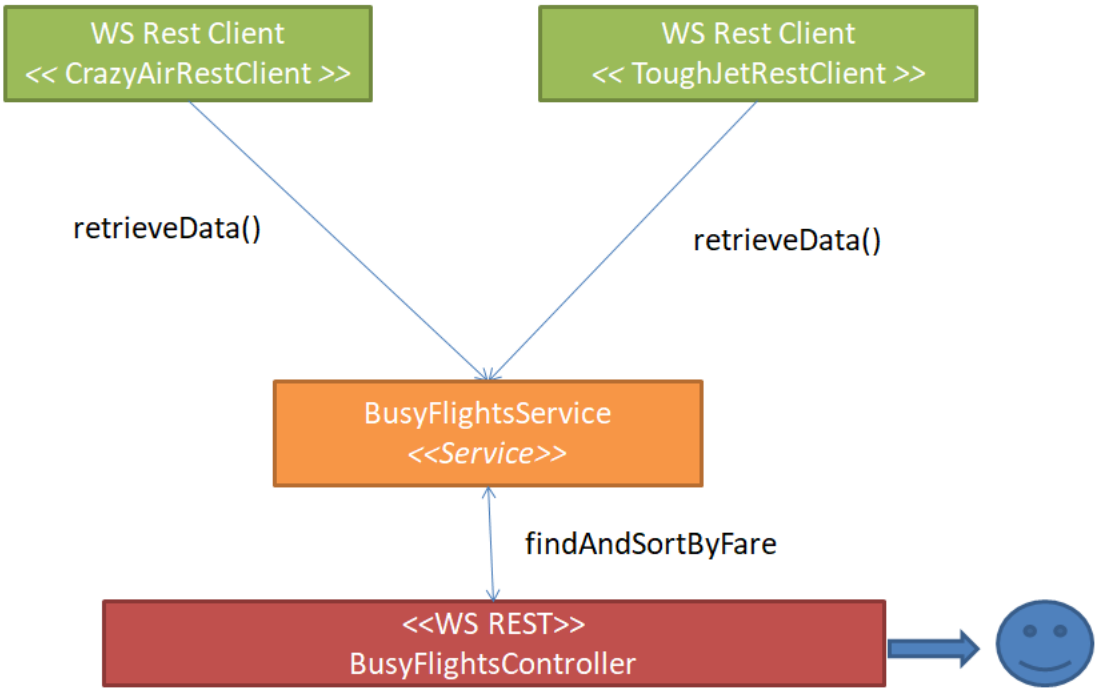
João Carlos Sousa do Vale

Jcarlos.vale@gmail.com

https://www.linkedin.com/in/joaocarlosvale/

https://github.com/jcarlosvale/interview-test

1. Architecture



Applied technologies:

- SpringBoot v2;
- Spring CDI;
- RestTemplate by Spring;
- Mockito for Unit Testing;
- Javax Validation;
- Java 8 Lambda;
- Gson;
- SpringBoot autoloaders: Web, Test

2. Code

- The created constructors are important to be used in the Tests.
- Lists were created to represent result responses of WS Clients.
- Equals and hashcode implemented to compare objects.
- It was created an Exception Controller to manage exceptions and WS Rest responses.
- The method post was chosen because it is not idempotent.
- Created a mass of tests.

    a. New classes
        i. BusyFlightsResponseList: represents a List of Buys Flights Response.
        ii. CrazyAirResponseList: represents a List of Crazy Air Responses.
        iii. CustomizedResponseEntityExceptionHandler: a shared across multiple Controller classes to manage Exceptions.
        iv. ExceptionResponse: an Exception Response Bean to represent exceptions.
        v. BusyFlightsController: main WS Rest Controller.
        vi. CrazyAirRestClient: CrazyAir WS Rest Client.
        vii. ToughJetRestClient: ToughJet WS Rest Client.
        viii. BusyFlightsDataConvertService: An util service class to convert data.
        ix. BusyFlightsService: responsible by retrieve data of WS Rest Clients, sort the response and send to User through our WS Rest Service.
    b. Modified
        i. Created methods equals, hashcode, toString in the Classes of Domain.
        ii. Created some constructors in Responses used in the tests.
    c. Javax Validations
        i. BusyFlightsRequest
    d. Tests
        i. BusyFlightsControllerTests

3. Improvements
    a. Scalability.
        i. Necessary JMeter to verify test stress of the current application;
        ii. Study and try an proof of concept integrating Apache Spark + Apache Kafka + Apache Cassandra (distributed, scalable technology + big data processing + pipelines and streaming)
    b. Future work
        i. Experiment the Java 9 reactive programming.

ii. Create an Interface to represent the WS Clients used to retrieve data from the air flights suppliers.
iii. Improve the documentation.
iv. Create the log, adding errors logs.
v. Improve tests, implementing more integration, unit and some functional test.
vi. Create exceptions to represent some fails like invalid responses, invalid responses against the request (dates, origin, destination), WS errors.
vii. Try to use LocalDateTime API to represent Dates, Big Decimal as currency value