Ryanair - Task 2 - Java/Spring - Interconnecting Flights

Write a Spring Boot based RESTful API application which serves information about possible direct and interconnected flights (maximum 1 stop) based on the data consumed from external APIs.

Given:

The application can consume data from the following two microservices:

- Routes API: https://services-api.ryanair.com/locate/3/routes which returns a list of all available routes based on the airport's IATA codes. Please note that only routes with: `connectingAirport` set to `null` and `operator` set to `RYANAIR` should be used.
  For example:

```
[
    {
        "airportFrom": "LUZ", # a departure airport IATA code
        "airportTo": "STN", # an arrival airport IATA code
        "connectingAirport": null, # a connecting airport IATA code
        "newRoute": false,
        "seasonalRoute": false,
        "operator": "RYANAIR",
        "group": "ETHNIC"
    },
    {
        "airportFrom": "CHQ",
        "airportTo": "SKG",
        "connectingAirport": null,
        "newRoute": false,
        "seasonalRoute": false,
        "operator": "RYANAIR",
        "group": "DOMESTIC"
    },

    (...)
]
```

- Schedules API: https://services-api.ryanair.com/timtbl/3/schedules/{departure}/{arrival}/years/{year}/months/{month} which returns a list of available flights for a given departure airport IATA code, an arrival airport IATA code, a year and a month.
  For example (https://services-api.ryanair.com/timtbl/3/schedules/DUB/WRO/years/2019/months/3):

```
{
    "month": 6, # a month of a year
    "days": [
        {
            "day": 1, # a day of a month
            "flights": [ # a list of flights for given day
                {
                    "number": "1926", # a flight number
                    "departureTime": "18:00", # a departure time in the
departure airport timezone
                    "arrivalTime": "21:35" # an arrival time in the arrival
airport timezone
                }
            ]
        },
        {
            "day": 3,
            "flights": [
                {
                    "number": "1926",
                    "departureTime": "17:25",
                    "arrivalTime": "21:00"
                }
            ]
        },

        (...)
    ]
}
```

Requirements:

- The source code of the application should be delivered (ideally shared through GitHub or Bitbucket).
- The application should build to an executable JAR file.
- The application should response to following request URI with given query parameters: `http://<HOST>/<CONTEXT>/interconnections?departure={departure}&arrival={arrival}&departureDateTime={departureDateTime}&arrivalDateTime={arrivalDateTime}` where:
    - departure – a departure airport IATA code
    - departureDateTime – a departure datetime in the departure airport timezone in ISO format
    - arrival – an arrival airport IATA code
    - arrivalDateTime – an arrival datetime in the arrival airport timezone in ISO format

  for example: `http://localhost:8080/somevalidcontext/interconnections?departure=DUB&arrival=WRO&departureDateTime=2018-03-01T07:00&arrivalDateTime=2018-03-03T21:00`

- The application should return a list of flights departing from a given departure airport not earlier than the specified departure datetime and arriving to a given arrival airport not later than the specified arrival datetime.
  The list should consist of:
    - all direct flights if available (for example: DUB – WRO)
    - all interconnected flights with a maximum of one stop if available (for example: DUB – STN – WRO)
- For interconnected flights the difference between the arrival and the next departure should be 2h or greater
- The list should be of the following form:

```
[
    {
        "stops": 0,
        "legs": [
            {
                "departureAirport": "DUB",
                "arrivalAirport": "WRO",
                "departureDateTime": "2018-03-01T12:40",
                "arrivalDateTime": "2018-03-01T16:40"
            }
        ]
    },
    {
        "stops": 1,
        "legs": [
            {
                "departureAirport": "DUB",
                "arrivalAirport": "STN",
                "departureDateTime": "2018-03-01T06:25",
                "arrivalDateTime": "2018-03-01T07:35"
            },
            {
                "departureAirport": "STN",
                "arrivalAirport": "WRO",
                "departureDateTime": "2018-03-01T09:50",
                "arrivalDateTime": "2018-03-01T13:20"
            }
        ]
    }
]
```

We are looking for the solution to be well factored and to adhere to the SOLID principles.