Data Engineering Assignment

Develop an Apache Spark application that can process the source dataset to a couple of derivatives and calculate a metric. The dataset is formatted as JSON.

The application should support two execution modes specified by the command line parameters:

Parse Mode

The *parse* mode only selects the `app_loaded` and `registered` events, and saves them on a disk in the parquet format, e.g, `/home/user/events/app_loaded/*.parquet` and `/home/user/events/registered/*.parquet`. The resulted events should comply to the specification provided below.

Source File

The source file consists of event records. Every record has properties which are different for every event type the source.

The common properties for every event type are:

- "timestamp" - the time when the event occurred (ISO-8601 format string),
- "event" - event type,
- "initiator_id" - user identity.

Event Specification

1. User registration - user registered in the service

```
"event": "registered"
"time": timestamp
"initiator_id": long
"channel": string
```

2. Application loading - recorded on application loading into browser

```
"event": "app_loaded"
"time": timestamp
"initiator_id": long
"device_type": string
```

Statistics Mode

The *statistics* mode computes a metric based on the `app_loaded` and `registered` events produced by the *parse* mode. The metric represents the percentage of all users who loaded the application in the calendar week

immediately after their registration week. The calendar week starts on Monday and finishes on Sunday. The application should work correctly in case the source data covers several years of history! The end result should be written to the console.

Example

For a given source file below,

```
{"event":"registered", "timestamp":"2020-02-11T06:21:14.000Z",
"initiator_id":1} //week#1: Monday
{"event":"registered", "timestamp":"2020-02-11T07:00:14.000Z",
"initiator_id":2} //week#1: Monday
{"event":"app_loaded", "timestamp":"2020-03-11T06:24:42.000Z",
"initiator_id":1, "device_type":"desktop"} //week#1: Tuesday
{"event":"registered", "timestamp":"2020-03-11T07:00:14.000Z",
"initiator_id":3} //week#1: Wednesday
{"event":"app_loaded", "timestamp":"2020-11-11T10:13:42.000Z",
"initiator_id":2, "device_type":"desktop"} //week#2: Wednesday
{"event":"app_loaded", "timestamp":"2020-12-11T11:08:42.000Z",
"initiator_id":2, "device_type":"desktop"} //week#2: Thursday
{"event":"app_loaded", "timestamp":"2020-17-11T11:08:42.000Z",
"initiator_id":3, "device_type":"mobile"} //week#3: Tuesday
```

the metric's calculation step should produce the following result:

```
Metric: 33%
```

Instructions

You are free to use any programming language of your choice, e.g. Java, Scala or Python. Please submit the completed assignment as a link to a **private** Git repository (Github, Bitbucket, etc.) or as a zipped archive by email. In the solution, we expect you to demonstrate the knowledge of commonly used engineering practices such as clean code, testing and documentation.

As part of the solution, please explicitly provide the following information:

- the instruction on how to run the application
- the design considerations explaining your approach, constraints, etc.
- *(optional)* the feedback on the assignment: what could we have improved in the assignment or description?

Notice:

Please **don't share** the assignment or your solution publicly.