



# Data Engineering Case Study

## Basic Spring Application with Kafka

We want you to write a Java application using Spring Framework that integrates with Apache Kafka. There is a package table that involves fields as described below. Package instances should be accessed through a JPA Repository implementation. Application has a core functionality of getting packages converting them into instances of MappedPackage class and then sending them to a kafka topic of your choice in your local machine. This functionality must be exposed in a controller with two different endpoints. The first endpoint takes the id of a package and sends a single message to Kafka, the second one retrieves all the package records and sends them to Kafka.

- First endpoint's URI should be /kafka/send/{packageId}
- Second endpoint's URI should be /kafka/bootstrap
- Cancelled packages should be filtered and not be sent to the topic.
- If the package is not completed, you can set related fields to null.
- "eta" is the estimated time of arrival in minutes from package creation to package completion. It is calculated during package creation and never changes.
- "lead\_time" is the realized minutes from package creation to package completion.
- "order\_in\_time" value points whether the package is delivered within the range of eta value.
- We do not require to run this on a cloud-based server, it is enough to be runnable on a local machine.
- Format of created\_at and last\_updated\_at fields of MappedPackage object should be "yyyy-MM-dd HH:mm:ss" or "yyyy-MM-dd HH:mm:ss.SSSSSS".

**Sample package data:**

```
{
  "id": 19604181,
  "arrival_for_delivery_at": "2021-11-13 11:05:58.045640",
  "arrival_for_pickup_at": "2021-11-13 10:48:37.032078", "cancel_reason": null,
  "cancelled": 0,
  "completed_at": "2021-11-13 11:40:15.314340",
  "created_at": "2021-11-13 10:47:52.675248",
  "customer_id": 20002011575015,
  "in_delivery_at": "2021-11-13 11:05:56.861614",
  "last_updated_at": "2021-11-13 11:40:15.314340",
  "eta": 277,
  "status": "COMPLETED",
  "store_id": 20000000004103,
  "origin_address_id": 999000020443388, "type": "REGULAR",
  "waiting_for_assignment_at": "2021-11-13 10:47:52.675248",
  "user_id": 50002010395213,
  "collected": 1,
  "collected_at": "2021-11-13 10:47:52.828692",
  "cancelled_at": null,
  "picked_up_at": "2021-11-13 10:49:50.278087",
  "reassigned": null, "order_id": 123972783,
  "delivery_date": "2021-11-13"
}
```

### MappedPackage:

Field Name	Type
id	Long
createdAt	String
lastUpdatedAt	String
collectionDuration	Integer
deliveryDuration	Integer
eta	Integer
leadTime	Integer
orderInTime	Boolean

## Architecture Question

We want to populate data continuously to the Kafka topic mentioned in the previous section. How can we achieve this? You can use the application implemented previously or make suggestions using alternative approaches.