Segmentify Coding Challenge

Segmentify integrates with online stores by offering different integration methods, which are detailed at https://segmentify.github.io/segmentify-dev-doc, one of them is by using REST API which is detailed under: https://segmentify.github.io/segmentify-dev-doc/integration_rest/

You are requested to create a simple web application which will mimic "Add Event" endpoint in the documentation. Your task is to create a web application that has one end point to accept event requests in JSON data format and persist them asynchronously.

We are expecting you to support following capabilities in the web application:

- Accept event requests with given end-point (POST request)
- Accept and parse JSON input
- Persist events to storage asynchronously in a separate thread without blocking event handling

Platform

We are asking you to create a web application by using **Java** (version >= 8) and advise you to use **Spring Boot** framework. You can use either Apache Tomcat or Netty for application server.

Requirements

Following list is the requirements of the challenge:

- Fully comply with asked capabilities in input and output format (please check example inputs below)
- Your code should compile and run in one step. It would be nice to have use maven and have build, test and run (mvn spring-boot:run)
- You **must** include unit tests.
- Your end-point should accept POST request with url: /add/events/v1.json?apiKey=<APIKEY>
- You must accept input in JSON format and return results in JSON (please check response format below)
- You must implement **validation** for event format (please check response format below for validations)

- Return codes and formats should follow these rules:
    - 200 – OK for valid input, and success response
    - 400 – Bad Request for malformed input, and error response
    - 404 – Not Found for requests different than given endpoint, and error response
- Event should be defined as an Abstract class with common fields (please check: https://segmentify.github.io/segmentify-dev-doc/integration_rest/#common-parameters)
- We expect you to implement **only Page View and Product View** events, you can skip all other events (please check: https://segmentify.github.io/segmentify-dev-doc/integration_rest/#page-view and https://segmentify.github.io/segmentify-dev-doc/integration_rest/#product-view)
- Please don't use any Database storage, but **use file system** to store events (e.g. PostgreSQL, MSSql, Mysql etc. not allowed, also using a design pattern is a good idea)
- Data persist job should be run **asynchronously** without blocking request handling
- It would be nice to use reactive programming with Spring Reactor

Response Format

Request response should be in the following format and should be returned as JSON:

```
{
  "timestamp": "1501142345512",
  "statusCode": "<STATUSCODE>"
}
```

Fields and definitions are given as:

- **timestamp:** current unix timestamp at request processing
- **statusCode:** string for return code (use the appropriate one defined below)

Following validations should be done for input and statusCode should be changed to error code

1- If there is **no event** at the input, statusCode should be **NO_EVENT**

2- If the input event is **malformed** (ex: Invalid JSON), statusCode should be **BAD_INPUT**

3- If there is **no userId** field present in the input, statusCode should be **NO_USERID**

4- If there is **no sessionId** field present in the input, statusCode should be **NO_SESSIONID**

For valid inputs, statusCode should be **SUCCESS**

Example Request - No Event Request – NO_EVENT

```
curl \
     --url 'https://localhost:8080/add/events/v1.json?apiKey=XXXXXXXX-XXXX-XXXX' \
     --request POST \
     --header 'Content-Type: application/json' \
     --header 'Accept: application/json' \
     --data-binary @- <<BODY
[]
BODY
```

Example Request – No User ID Event Request – NO_USERID

```
curl \
     --url 'https://localhost:8080/add/events/v1.json?apiKey=XXXXXXXX-XXXX-XXXX' \
     --request POST \
     --header 'Content-Type: application/json' \
     --header 'Accept: application/json' \
     --data-binary @- <<BODY
{
     "name":"PAGE_VIEW",
     "sessionId":"YYYYYYYYYYYYYYYYYYYY",
     "pageUrl":"https://www.mysite.com/",
     "category":"Home Page",
     "device":"iOS"
}
BODY
```

Example Request – No Session ID Event Request – NO_SESSIONID

```
curl \
     --url 'https://localhost:8080/add/events/v1.json?apiKey=XXXXXXXX-XXXX-XXXX' \
     --request POST \
     --header 'Content-Type: application/json' \
     --header 'Accept: application/json' \
     --data-binary @- <<BODY
{
     "name":"PAGE_VIEW",
     "userId":"XXXXXXXXXXXXXXXXXXX",
     "pageUrl":"https://www.mysite.com/",
     "category":"Home Page",
     "device":"iOS"
}
BODY
```

Example Request – Malformed Event Request – BAD_INPUT

```
curl \
     --url 'https://localhost:8080/add/events/v1.json?apiKey=XXXXXXXX-XXXX-XXXX' \
     --request POST \
     --header 'Content-Type: application/json' \
     --header 'Accept: application/json' \
     --data-binary @- <<BODY
{
     "name":"PAGE_VIEW",
     "userId:"XXXXXXXXXXXXXXXXXXX",
     "pageUrl":"https://www.mysite.com/"
     "category":"Home Page"
}
BODY
```

Example Request – Single Page View Event Request – SUCCESS

```
curl \
     --url 'https://localhost:8080/add/events/v1.json?apiKey=XXXXXXXX-XXXX-XXXX' \
     --request POST \
     --header 'Content-Type: application/json' \
     --header 'Accept: application/json' \
     --data-binary @- <<BODY
{
     "name":"PAGE_VIEW",
     "userId":"XXXXXXXXXXXXXXXXXXX",
     "sessionId":"YYYYYYYYYYYYYYYYYYY",
     "pageUrl":"https://www.mysite.com/",
     "category":"Home Page",
     "device":"PC"
}
BODY
```

Example Request – Single Product View Event Request – SUCCESS

```
curl \
     --url 'https://localhost:8080/add/events/v1.json?apiKey=XXXXXXXX-XXXX-XXXX' \
     --request POST \
     --header 'Content-Type: application/json' \
     --header 'Accept: application/json' \
     --data-binary @- <<BODY
{
     "name":"PRODUCT_VIEW",
     "userId":"XXXXXXXXXXXXXXXXXXX",
     "sessionId":"YYYYYYYYYYYYYYYYYYYY",
     "pageUrl":"https://www.mysite.com/",
     "referrer":"",
     "lang":"TR",
     "currency":"TRY",
     "device":"PC",
     "productId":"EXAMPLE_PRODUCT_1",
     "title":"Example Product",
     "inStock":true,
     "url":"https://www.mysite.com/example-product-1/",
     "image":"https://cdn.mysite.com/example-product-1.png",
     "category":"Men>Sports>Shoes",
     "brand":"Example Brand",
     "price":349.99,
     "oldPrice":449.99,
```

```
    "gender":"M",
    "colors":["black","red","blue"],
    "sizes":["small","medium","large"],
    "labels":["new-comer","top-seller"],
    "params":{
        "field1":"value1",
        "field2":"value2"
    }
}
BODY
```

Example Request – Multiple Event Request – SUCCESS

```
curl \
    --url 'https://localhost:8080/add/events/v1.json?apiKey=XXXXXXXX-XXXX-XXXX' \
    --request POST \
    --header 'Content-Type: application/json' \
    --header 'Accept: application/json' \
    --data-binary @- <<BODY
[
    {
        "name":"PAGE_VIEW",
        "userId":"XXXXXXXXXXXXXXXXXXX",
        "sessionId":"YYYYYYYYYYYYYYYYYYY",
        "pageUrl":"https://www.mysite.com/",
        "category":"Home Page",
        "device":"PC"
    },
    {
        "name":"PRODUCT_VIEW",
        "userId":"XXXXXXXXXXXXXXXXXXX",
        "sessionId":"YYYYYYYYYYYYYYYYYYYY",
        "pageUrl":"https://www.mysite.com/",
        "referrer":"",
        "lang":"TR",
        "currency":"TRY",
        "device":"PC",
        "productId":"EXAMPLE_PRODUCT_1",
        "title":"Example Product",
        "inStock":true,
        "url":"https://www.mysite.com/example-product-1/",
        "image":"https://cdn.mysite.com/example-product-1.png",
        "category":"Men>Sports>Shoes",
        "brand":"Example Brand",
        "price":349.99,
        "oldPrice":449.99,
        "gender":"M",
        "colors":["black","red","blue"],
        "sizes":["small","medium","large"],
        "labels":["new-comer","top-seller"],
        "params":{
            "field1":"value1",
            "field2":"value2"
        }
    }
]
BODY
```