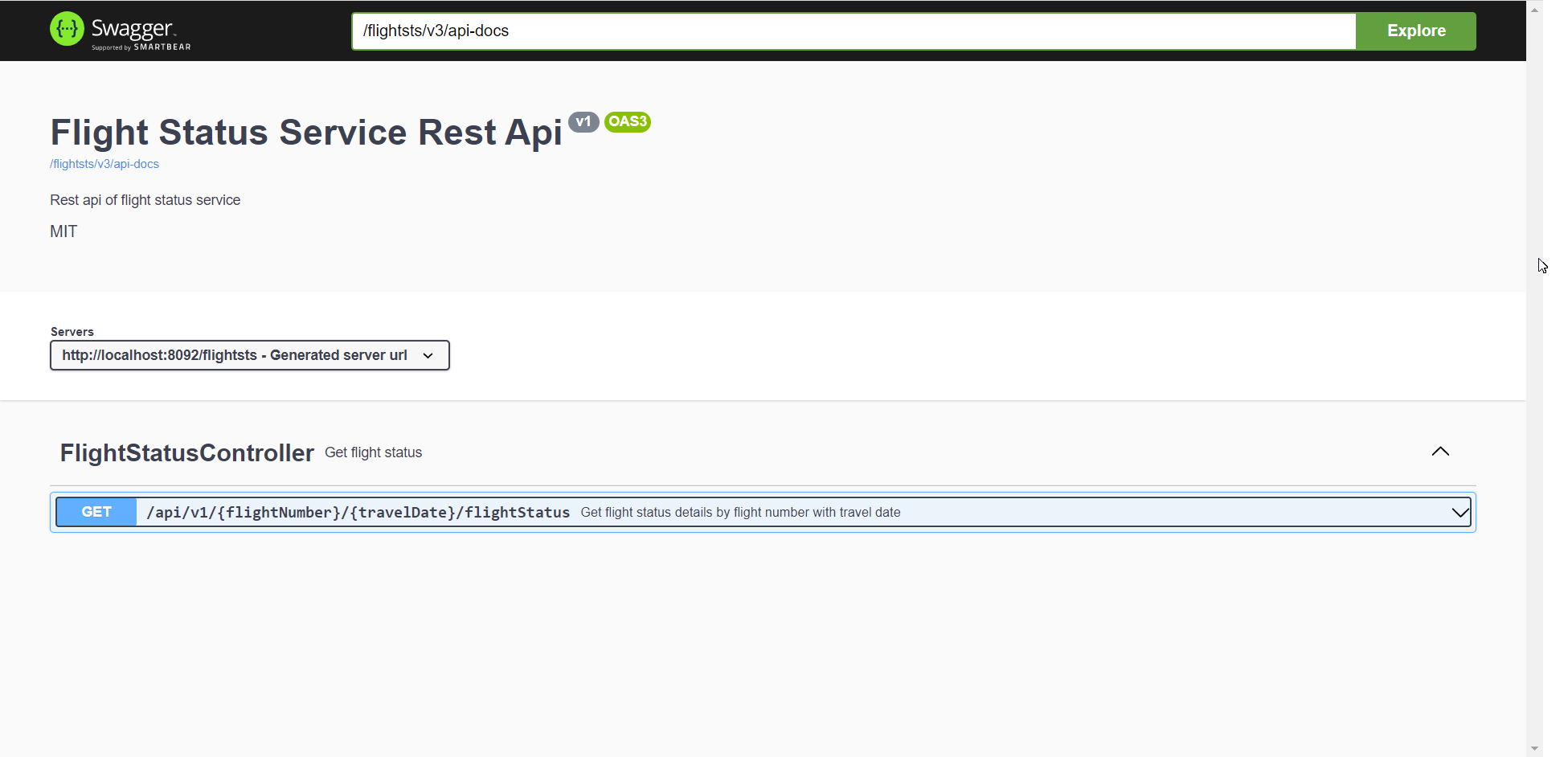
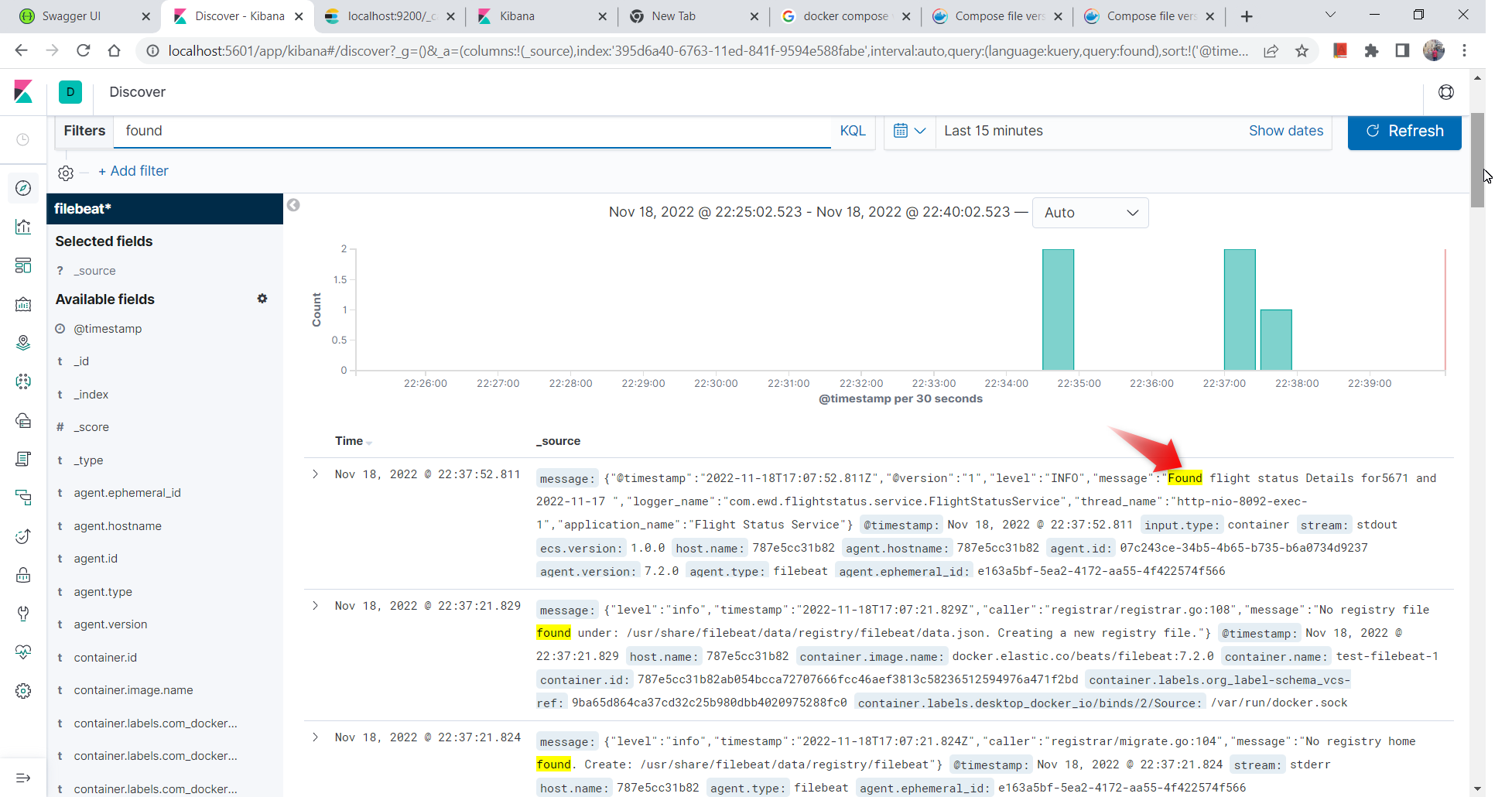
1. I have used spring boot for developing the microservice.
2. I assumed that flight status information will be pulled from 3rd party system and stored in our service, but I am not clear on the 3rd party system, so I have used [**https://aviationstack.com/**](https://aviationstack.com/)and pulled the information and stored it in the in-memory cache to the status search logic.
3. PFB table for the request and response details.
4. **Request:** flightNumber, travelDate **Response:** as JSON with all the required details (departure time, arrival time, flight route, flight number, status e.g. on time or delayed)
5. 
6. I have defined a business event that if the flight status is found or not which will be shown in the Kibana.
7. 
8. **Deployment Steps:**
   1. Build microservice using **mvn clean install** inside the project folder
   2. To get the Junit test report run **mvn site** command.
   3. Once the build is done. Build the image for the microservice inside the project folder🡪 **docker build -t ewd/flight-status-service: latest.**
   4. run the docker-compose file
   5. create an index in Kibana(http://localhost:5601/) as **filebeat-\***
   6. Add a filter as a container.image.name: "ewd/flight-status-service: latest"
9. Principles and code practices:
   1. I have used OpenAPI standards for API definitions.
   2. Used Swagger for API details.
   3. Covered the logic using JUNITs > 90 %
   4. Followed clean code principles.
   5. Logged necessary information for debugging.
   6. Packaged the services as docker.
   7. Used custom exception handling logic.