Restaurant open/closed microservice - code challenge

Description
In our quest to move away from a monolithic application and to benefit from the scalability of an event driven architecture, we want a new microservice to be responsible for a restaurant's open/closed state.

Open/closed microservice
The open/closed microservice will be the new source of truth in JUST EAT Takeaway regarding the restaurant's open state. It must keep the state in-memory.

We can set a restaurant open or closed by the API that this microservice exposes, as well as getting the current state of a restaurant.

When a restaurant's opening state has changed, the microservice emits an event on Kafka.

Frontend service
Our hungry customers are looking at a list of restaurants provided by the frontend service. They want to see which restaurants are open and which ones are closed.

The restaurant service consumes events from the open/closed microservice and keeps changes in-memory (to save you some time, you don't have to implement a database).

You are free to choose your favorite free-tier online message BUS for this challenge. (Kafka, RabbitMQ, SNS/SQS)

Must haves

- The frontend is not affected by an outage on the open/closed microservice.
- The microservices are horizontally scalable.
- Your code has tests.
- A readme is included with your explanation and thoughts, possible struggles, as well as instructions on how to get your challenge running.
- You're using micro-commits

Nice to haves (Bonus points)

- Use real databases instead of in-memory.
- Separate the frontend service into 2 microservices.
    - A microservice for the UI
    - A worker that listens and writes updates to the frontend database (or API if you use in-memory)
- Message BUS included in the local docker setup instead of externally managed.
- An interface where we can open/close restaurants.
- A diagram of how the services are connected

Required tech:

- Microservices, which use their own storage (in-memory, or a real DB)
- Preferably spring framework
- Docker, docker-compose

Important

We'll only have Docker running on our machines and should not have to install any additional software to get this micro-service running.

You have one week to turn in this challenge.

Done?

Great! Please push everything to a Github repository and share the URL with us, or zip the project (don't forget to include the git files) and share it with the recruiter.

Best of luck!