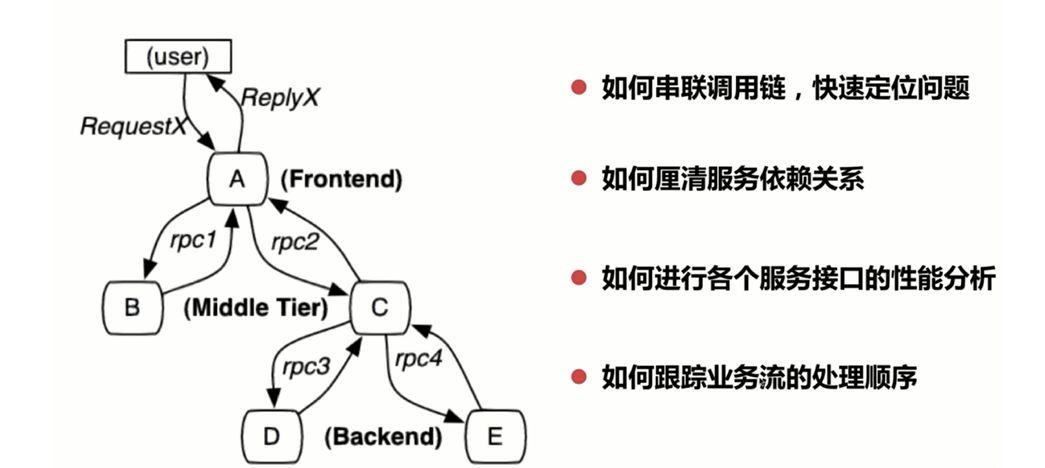
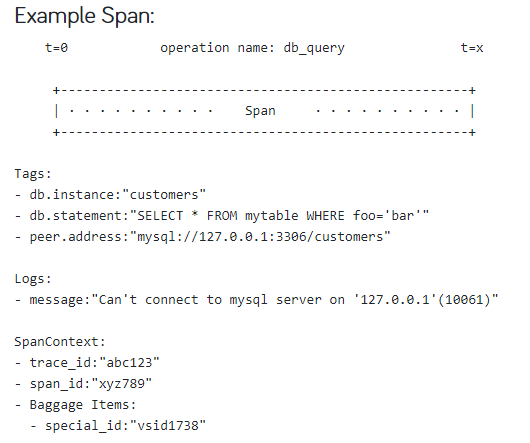
Knowledge Sharing for Jaeger

What is Distributed Tracing?

Distributed tracing, also called distributed request tracing, is a method used to profile and monitor applications, especially those built using a microservices architecture. Distributed tracing helps pinpoint where failures occur and what causes poor performance. Distributed tracking system is developing rapidly with various types, but the core steps are generally three: code embedding point(代码埋点), data storage(数据储存) and query display(查询展示)

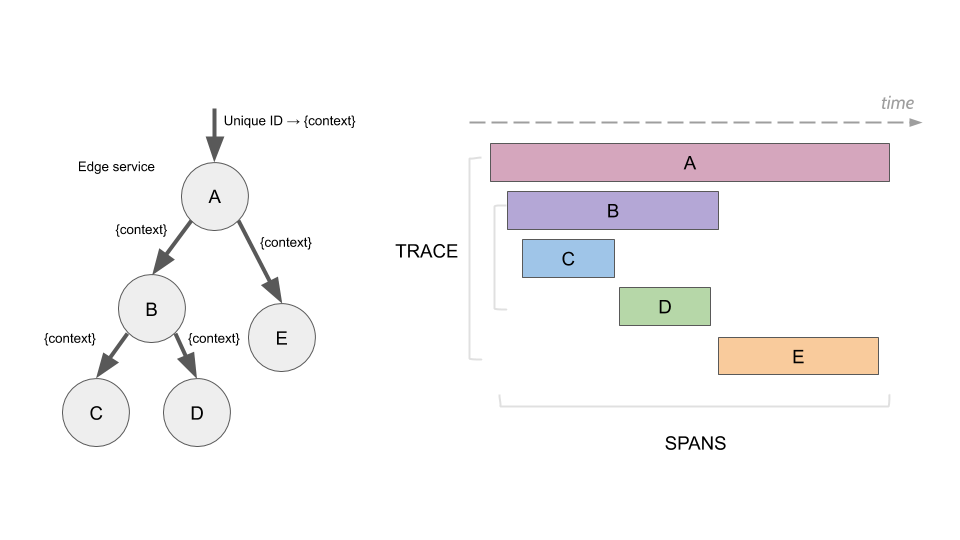
Span

A **span** represents a logical unit of work in Jaeger that has an operation name, the start time of the operation, and the duration. Spans may be nested and ordered to model causal relationships.



Trace

A **trace** is a data/execution path through the system, and can be thought of as a directed acyclic graph of [spans](https://www.jaegertracing.io/docs/1.18/architecture#span).



**Jaeger**

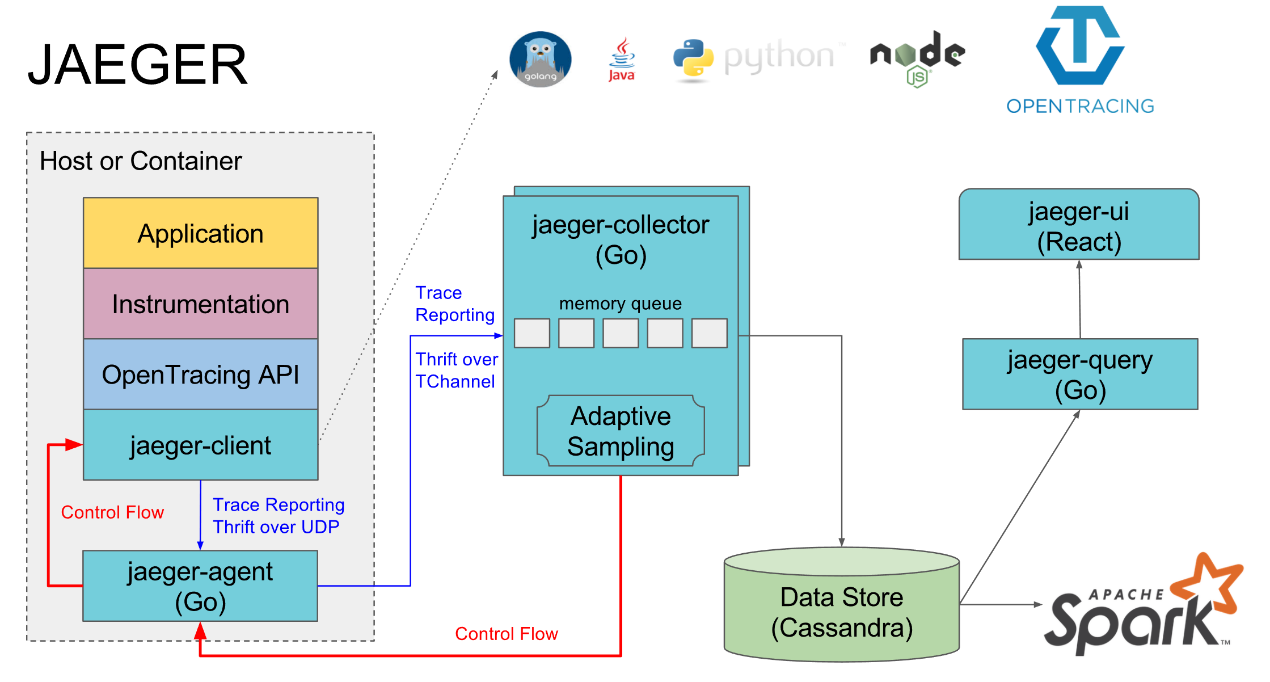
open source, end-to-end distributed tracing Monitor and troubleshoot transactions in complex distributed systems

Features

Jaeger is used for monitoring and troubleshooting microservices-based distributed systems, including:

* Distributed context propagation
* Distributed transaction monitoring
* Root cause analysis
* Service dependency analysis
* Performance / latency optimization

Architecture



jaeger-agent

The Jaeger **agent** is a network daemon that listens for spans sent over UDP, which it batches and sends to the collector.

jaeger-collector

Retrieve trace information from the queue and persist it (or direct memory storage), Jaeger’s storage is a pluggable component which currently supports [Cassandra](https://www.jaegertracing.io/docs/1.18/deployment#cassandra), [Elasticsearch](https://www.jaegertracing.io/docs/1.18/deployment#elasticsearch) and [Kafka](https://www.jaegertracing.io/docs/1.18/deployment#kafka).

jaeger-query

**Query** is a service that retrieves traces from storage and hosts a UI to display them

Installation and deployment

All in One(Not suitable for production deployment)

All-in-one is an executable designed for quick local testing, launches the Jaeger UI, collector, query, and agent, with an in memory storage component.

The simplest way to start the all-in-one is to use the pre-built image published to DockerHub (a single command line).

$ docker run -d --name jaeger \

-e COLLECTOR\_ZIPKIN\_HTTP\_PORT=9411 \

-p 5775:5775/udp \

-p 6831:6831/udp \

-p 6832:6832/udp \

-p 5778:5778 \

-p 16686:16686 \

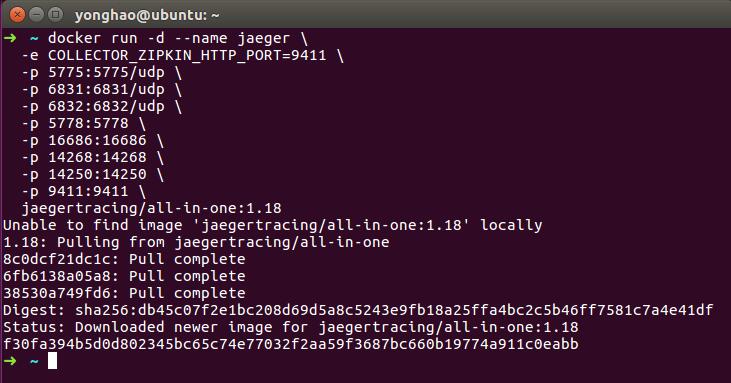
-p 14268:14268 \

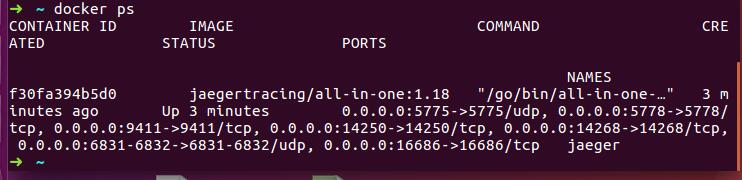
-p 14250:14250 \

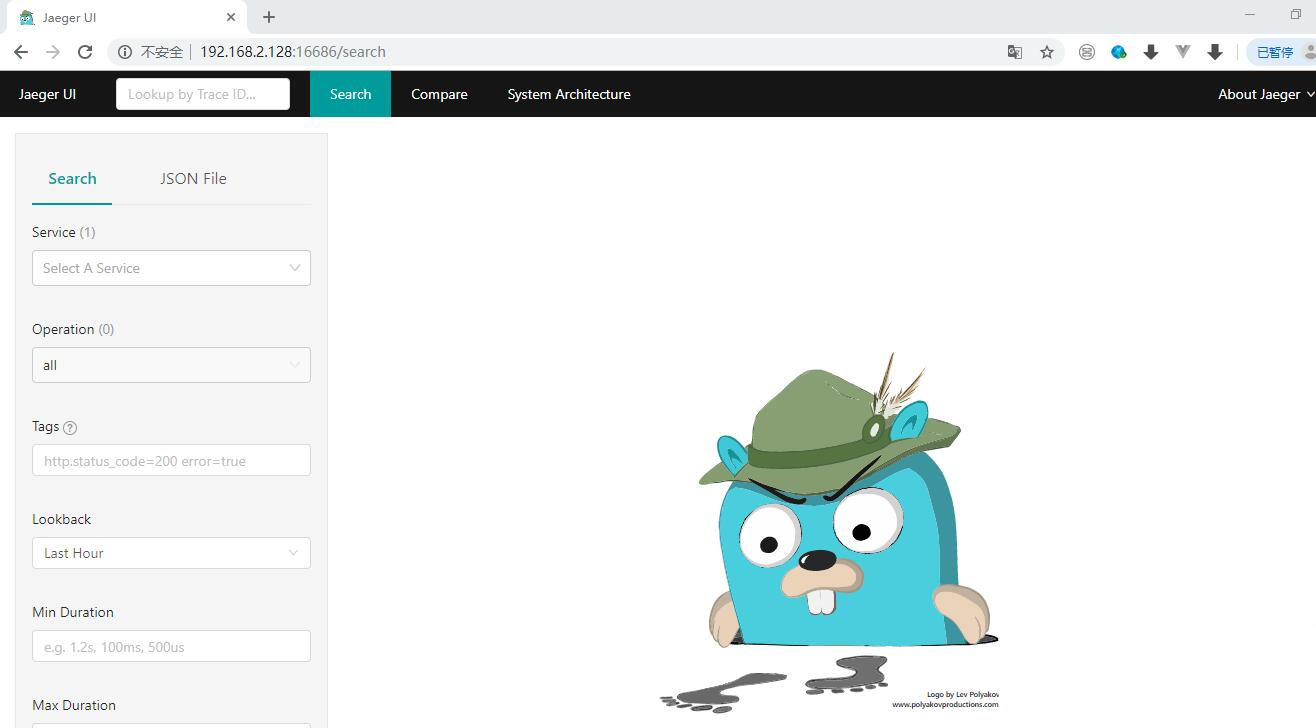
-p 9411:9411 \

jaegertracing/all-in-one:1.18





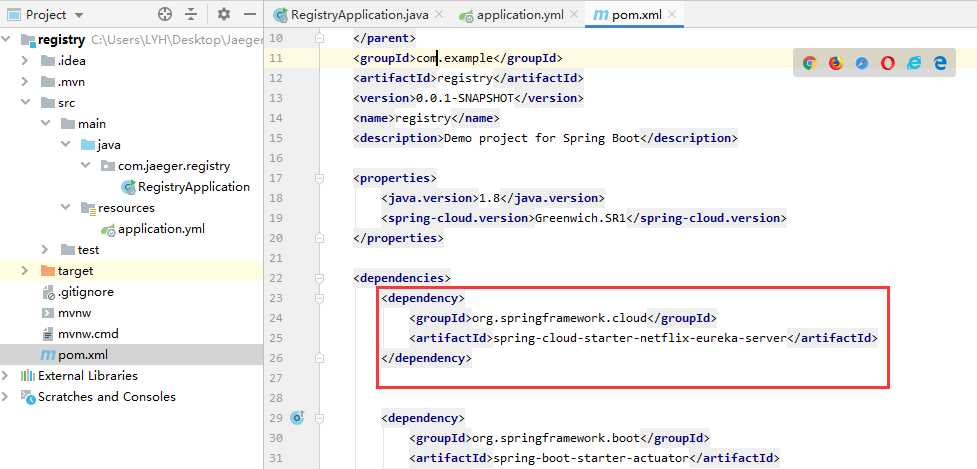
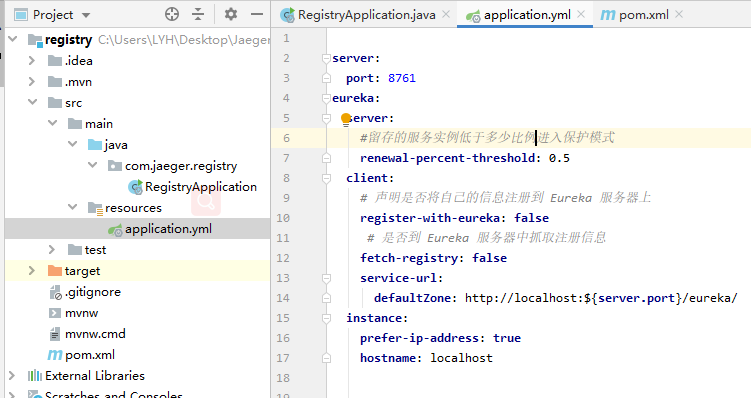
You can then navigate to http://localhost:16686 to access the Jaeger UI.

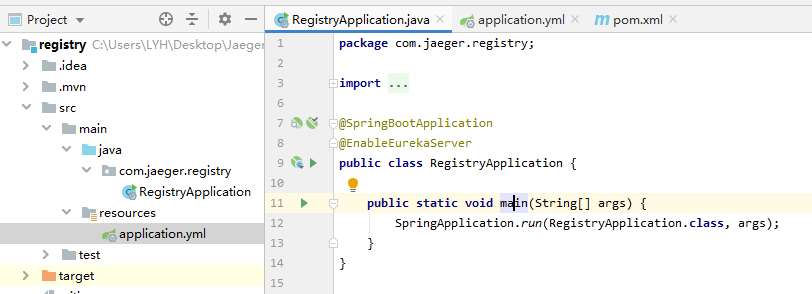


JAVA Project Practice

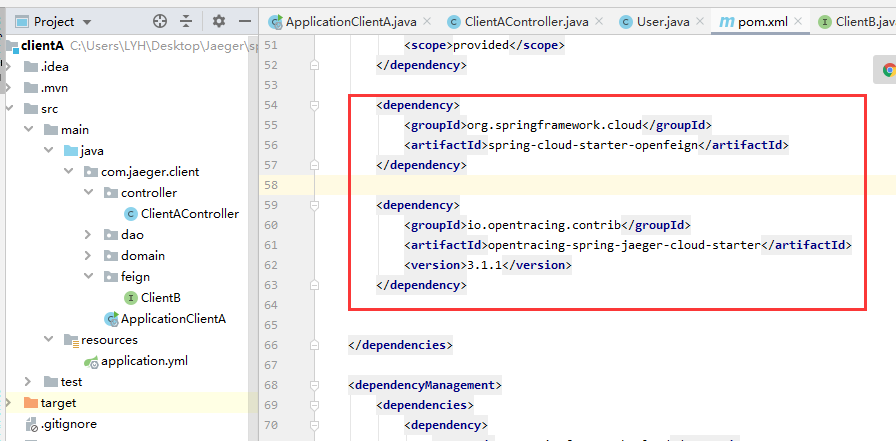


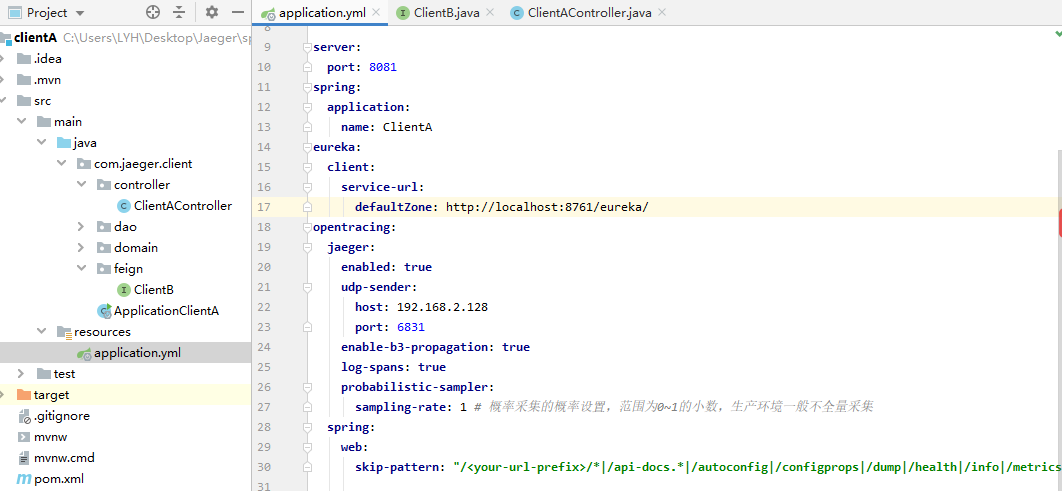
Registry:

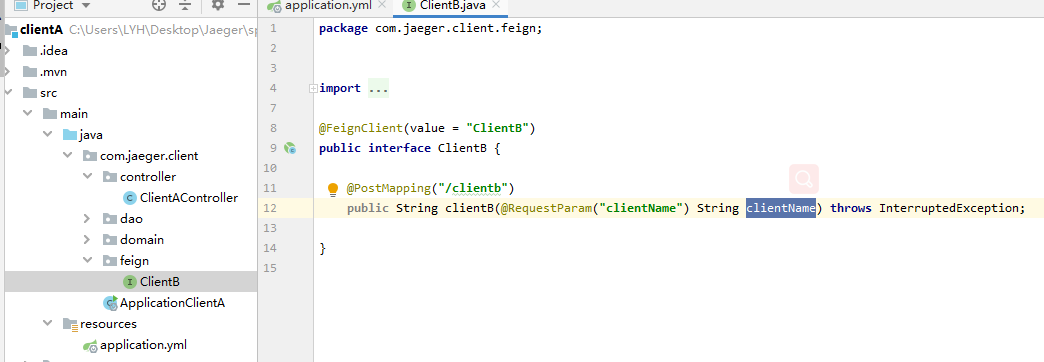
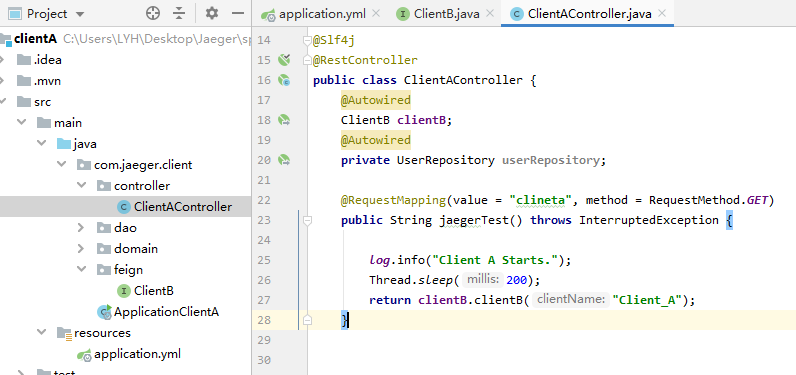
 



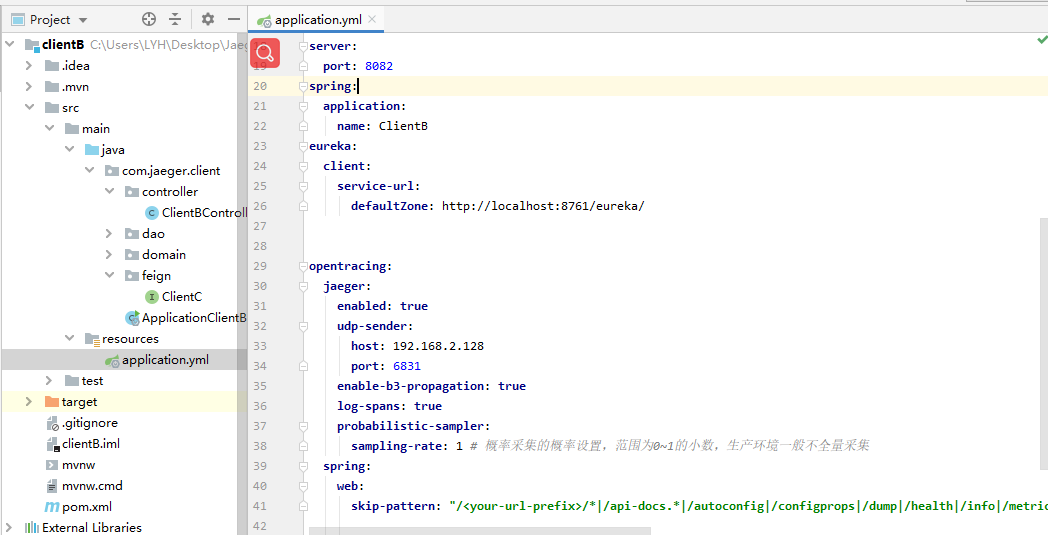
ClientA：

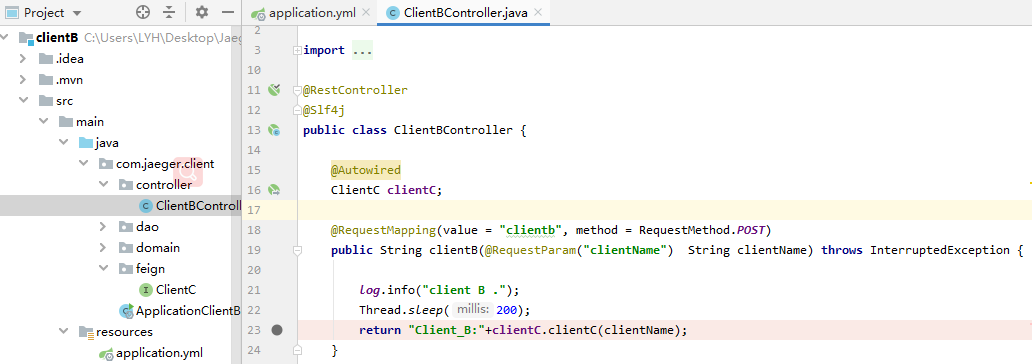




ClientB:



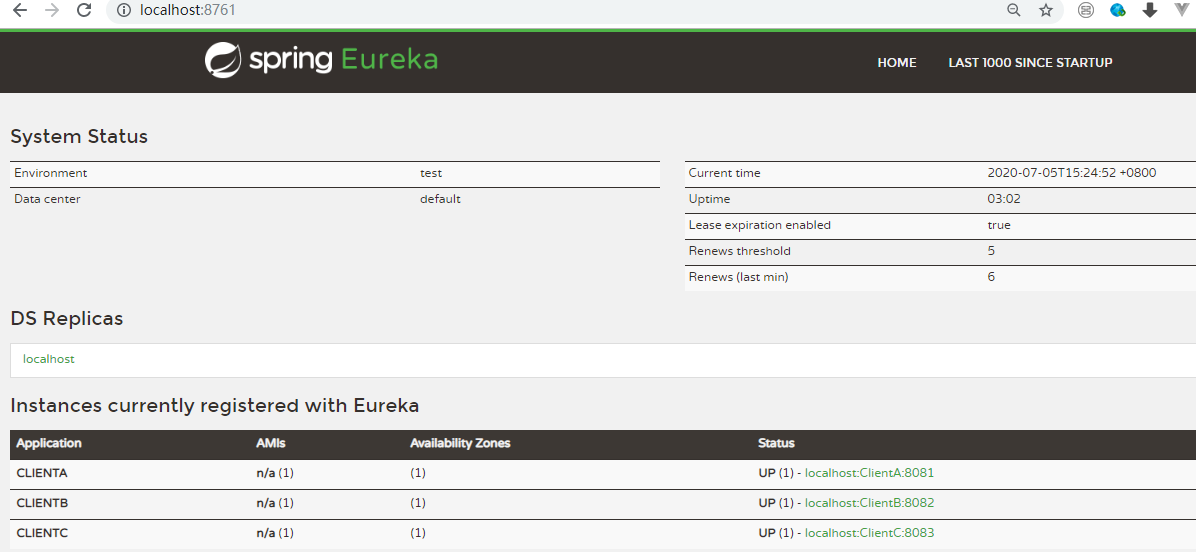


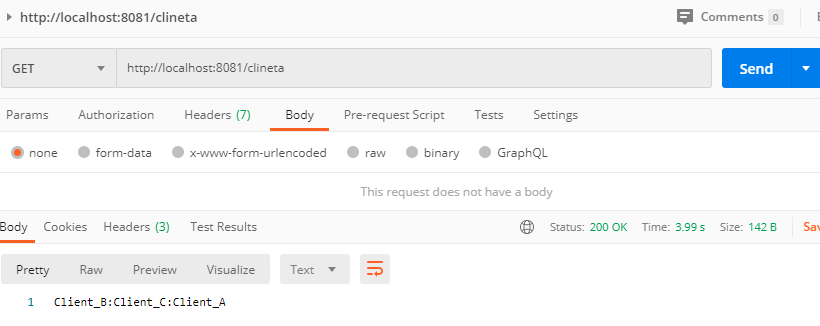


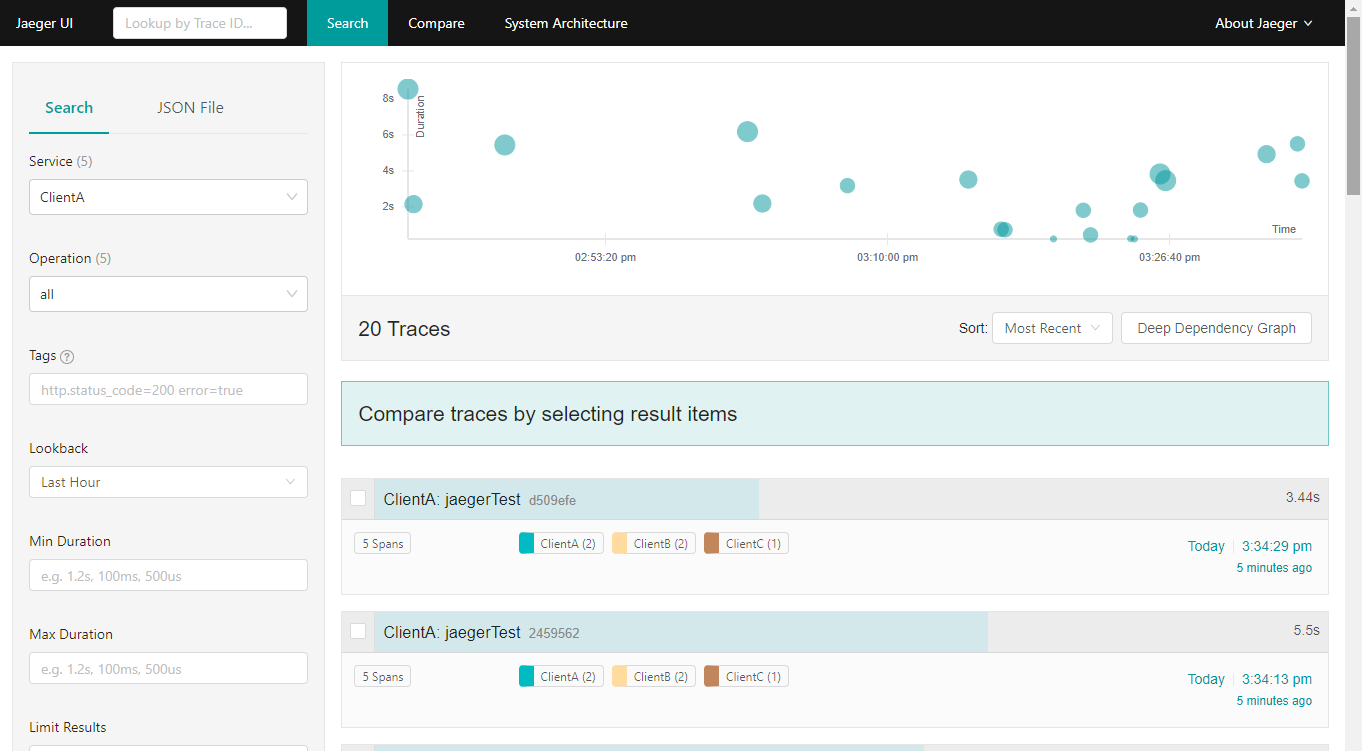
ClientC:

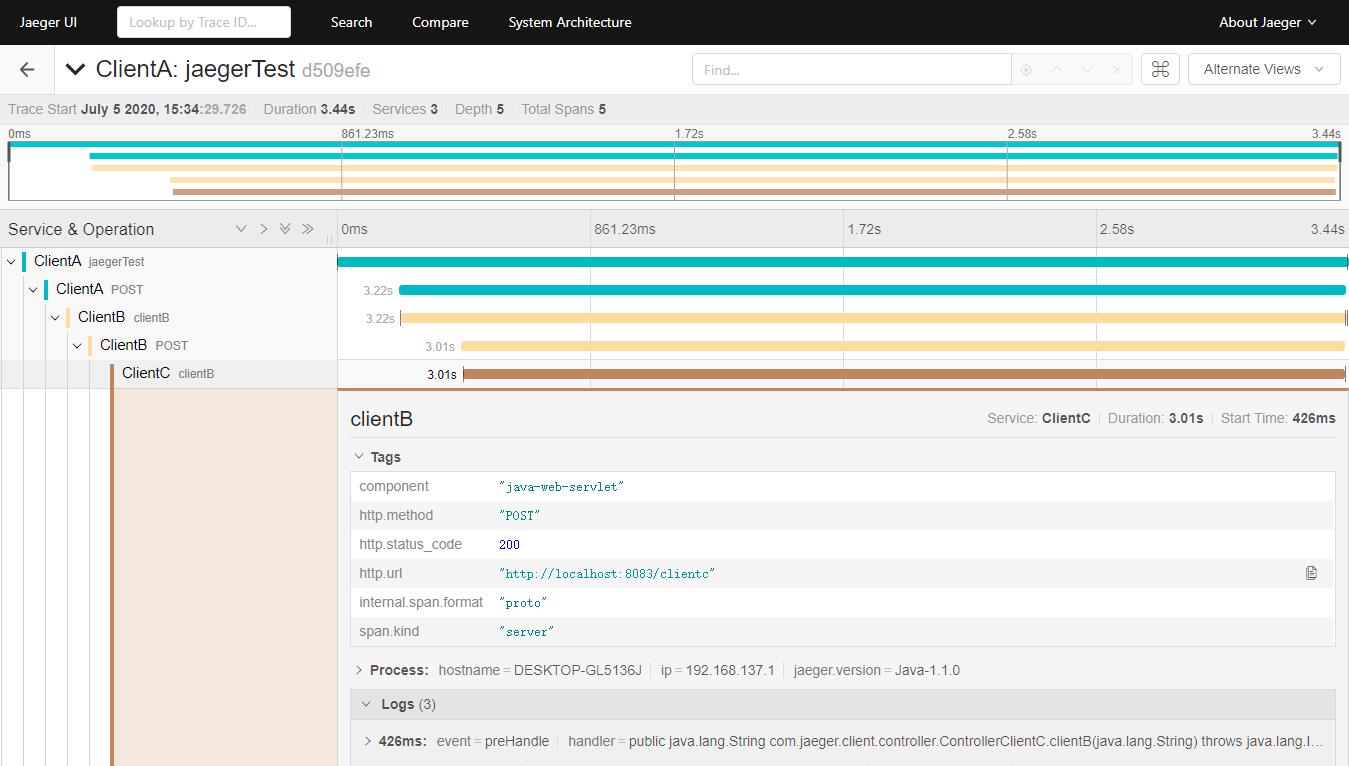


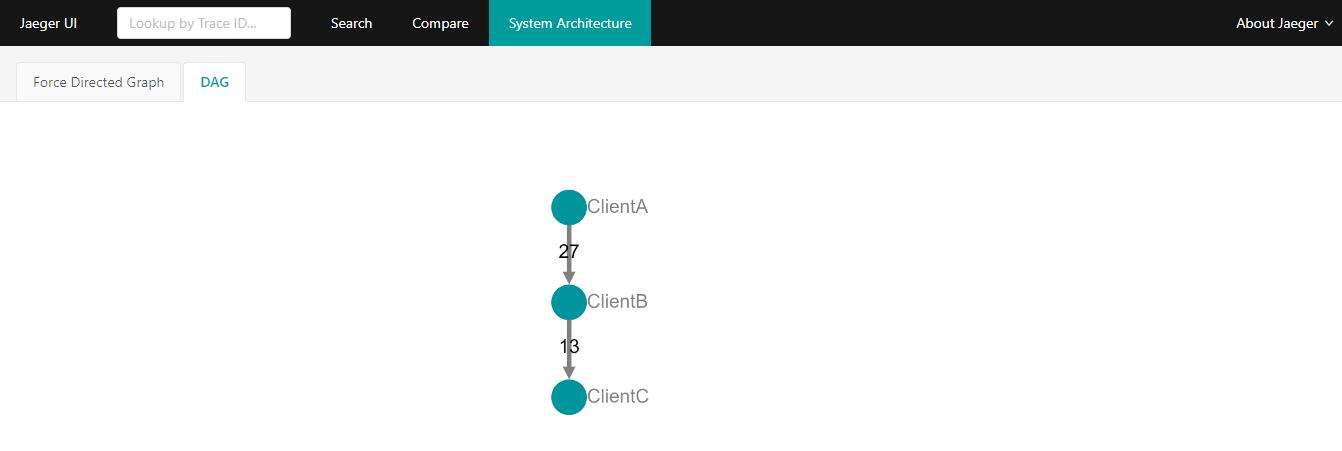
After the project starts:

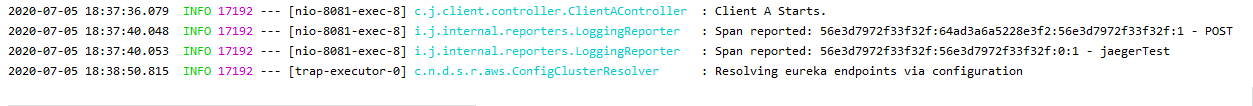
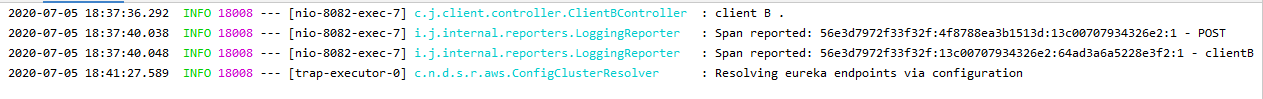


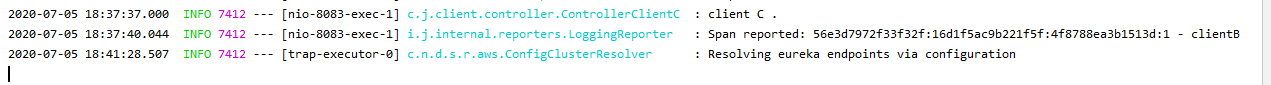




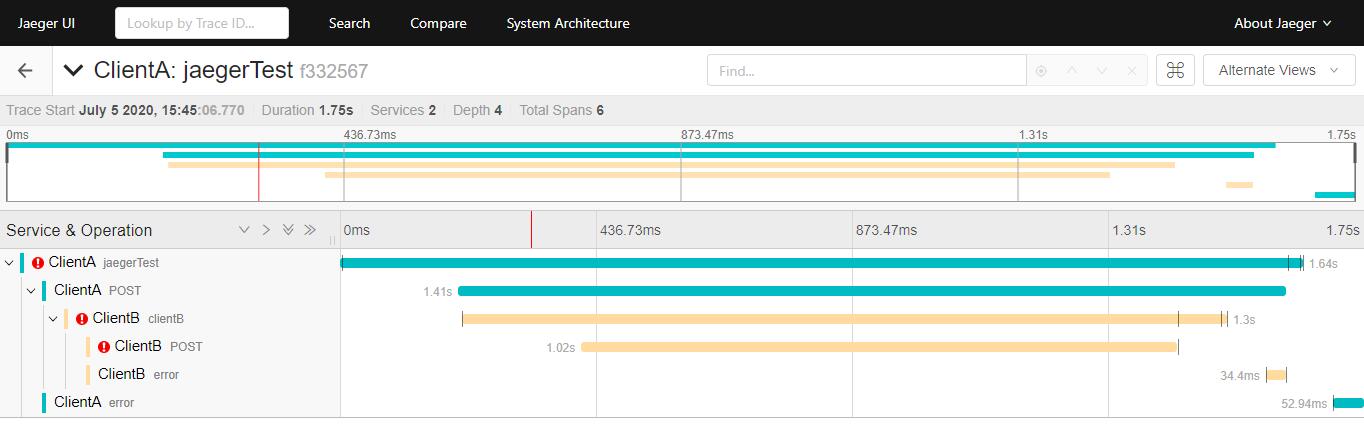






Client C error scenario:



Jdbc scenario:

