**Distributed Tracing**

**Lecture 92/Step 35: Introduction to distributed Tracing**

Centralized tracing mechanism: using sleuth & zipkins.

The concept used id is :

**“attach a unique ID to each request to track which all micro services it passed through”**

**Spring Sleuth :** This component attaches a unique ID to the request

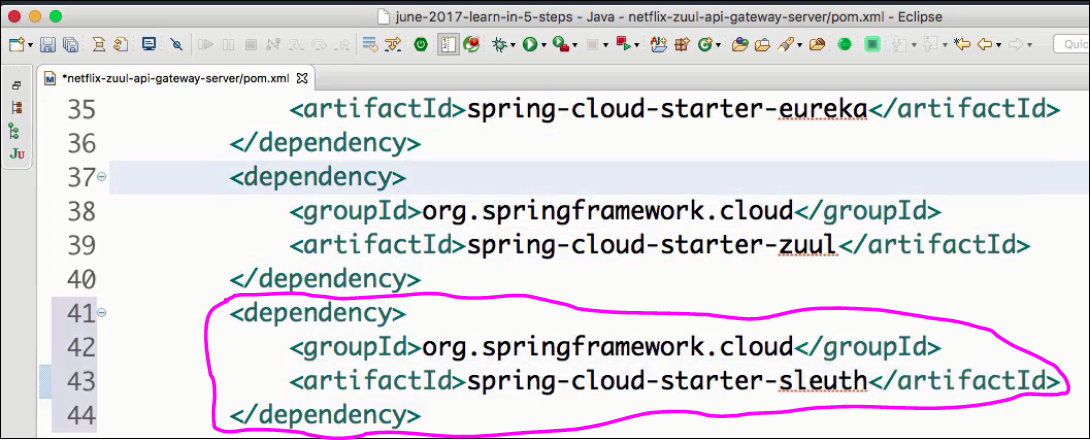
**Zipkins**: A distributed Tracing mechanism .

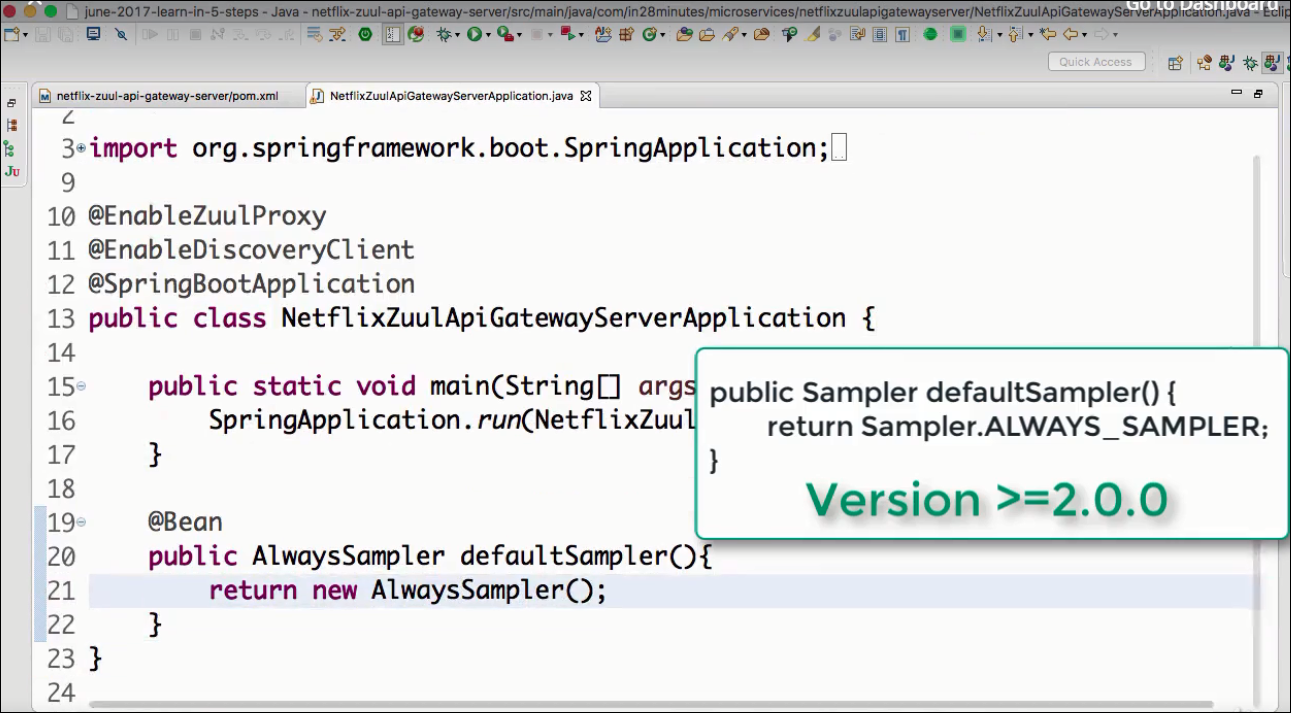
How this is achieved , is the logs for this request ( identified by the unique ID ) is put in a MQ implemented by Rabbit MQ , which is then transmitted to the Zipkins server.

**Lecture 93/Step 36: Add Spring Sleuth**

**Perform the below action to NZ-API Gateway Service & to** **all the services which should have DT enabled ( here CES,CCS)**

**Step 1) :** → Add spring-sleuth **dependency** in pom.xml (to NZ-API-GW,CES,CCS )

**Step 2):**→ Add **Sampler** ( as a @Bean ) in the Application class to tell which type/pattern of request should be filtered/tracked. (to NZ-API-GW,CES,CCS )



**Step 3):**→ Add logger at respective methods in respective controllers of the m/s. I.e add loggers to ( ZuleLoggingFilter.java,CurrencyExchangeController.java,CurrencyConversionController.java) of (NZ-API,CES,CCS ).

**Step 4):** Launch the services in the order :

--------> Launch Eureka Naming Server(EUKA)

--------> Launch Currency Exchange Service(CES)

--------> Launch Currency Conversion Service(CCS)

--------> Launch Netflix-Zuul-API Gateway service (NZ-API-GW)

Give a call to CCS via API Gateway and you can see the logs in NZ-API-GW,CES,CCS with a specific ID. Now , Traversing this through various console is difficult when there are 100s or m/s. Thus wee need to send this to a **centralized Logging service.**

**Lecture 95/Step 38: Centralize the Logging server:**

Note: **----------------**

There are many centralized Logging Solutions available , most popular are :

1. ELK stack [Elastic Search, Logstash , Kibana ]- managed by Elastic

Lucene → indexing+ search

ElasticSearch → distributed search capabilities

Logstash → to all kind of Tanseries data

Kibana → Visualization tool

Output => QBOX ( Capabilities of Tablue, Jaspersoft etc )

-------------------------

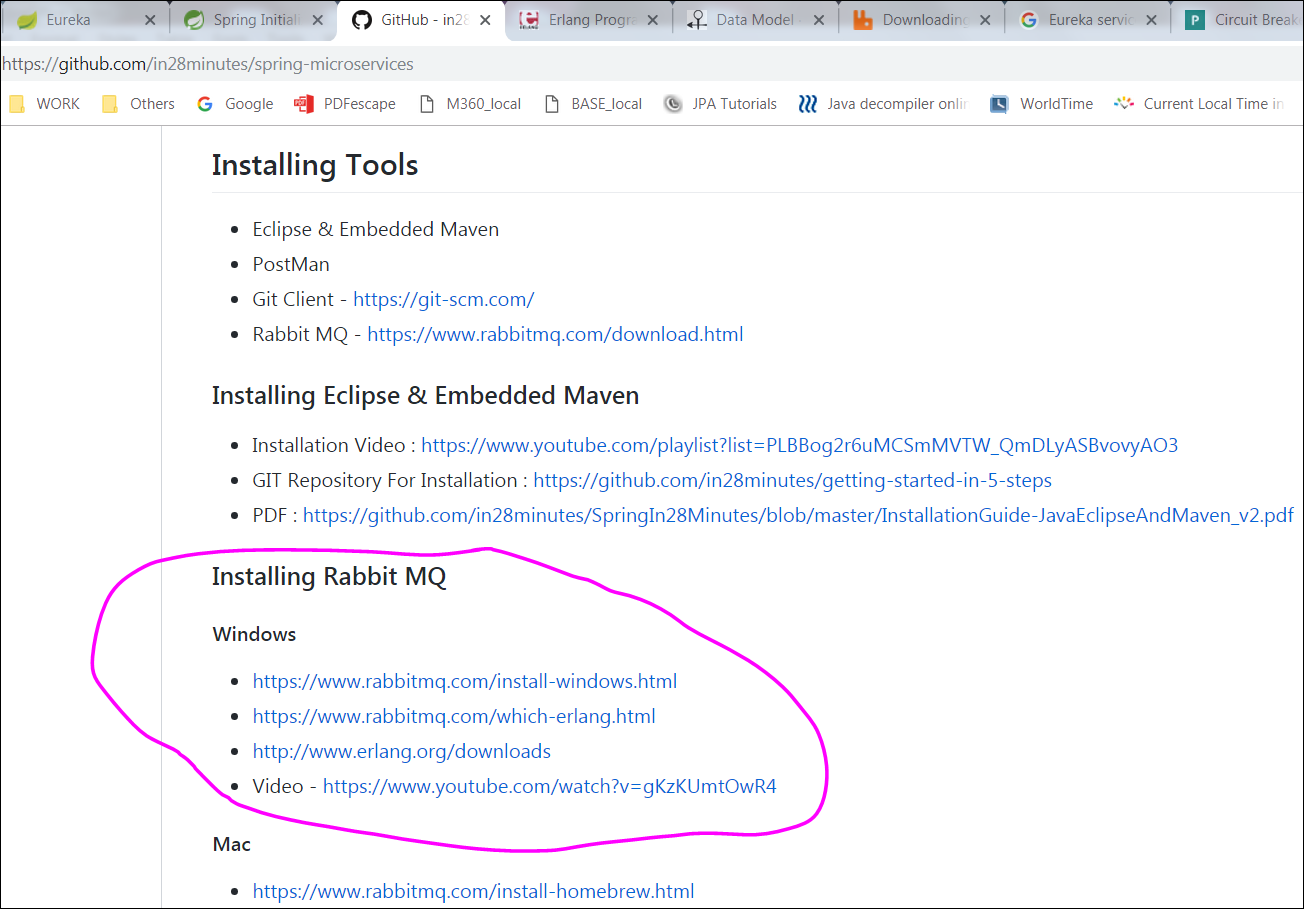
Zipkins uses the above technology to centralize this . The topology we’ll use is as follows:

**M/S ---> RabbitMQ ---> Zipkins ---> Database**

Micro services will push all their logs to RabbitMQ , Zipkins will collect it from it & push it to its DB)

-------------------------------------------------------------------------------------------------------

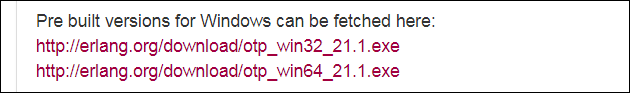
Refer the following links for installation from the tutorial github site:



**Step1: Install MQ**

**( For Windows)**

→ Pre-requsites : Install a version of **Erlang** . Which version to install , see the links in tutorial(screenshot above). <http://www.erlang.org/downloads>



→ download and install Rabbit MQ.( Any version is fine ,from **Github** or **Bintray**)

[**https://www.rabbitmq.com/download.html**](https://www.rabbitmq.com/download.html) **>> Windows:** [**Installer (recommended)**](https://www.rabbitmq.com/install-windows.html)



**(For Mac)**

**→ Pre-requisites: Erlang ( Homebrew or MacPorts)**



→ Use **Homebrew**.

→ make sure u download the latest version of Homebrew

→ add the folder where it is installed to your path

**Now , Start MQ Server**

**Step2: Install ZipKins**

--Config & Download the Zipkins from springs initializer: This is no more supported from spring Finchley : 2.0.0.M3

– We’ll download from Zipkins from zipkins.io page ( <https://zipkin.io/pages/quickstart>)

– Docker version

– Java version ( Recomended )

– copy the link provided on the Quick start page and paste it on browser , this will download a **zipkins-server-xxx-exec.jar**

**Step3: Connect RabbitMQ to Zipkins**

– ( On windows):( we can put the below command in **zipkin\_start.bat** file & execute the bat

prompt$> **SET RABBIT\_URI=amqp://localhost**

prompt$> **java -jar zipkin-server-2.11.7-exec.jar**

– verify that it is installed → <http://localhost:9411/zipkin/> or <http://localhost:9411/>

(Note the ‘/’ after zipkin )

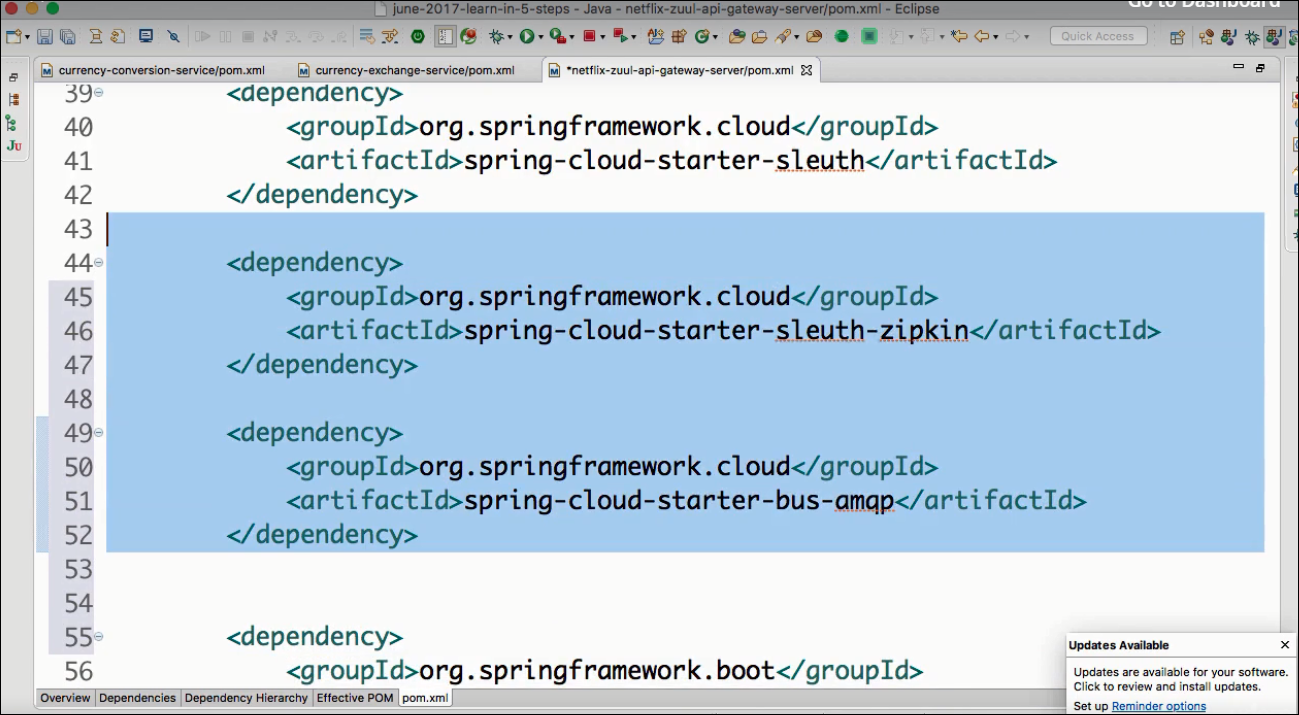
**Step4: Add snippets to m/s to push logs to rabbit MQ**

– add the following dependency in pom.xml of all these m/s ( CES,CCS & NZ-API-GW )

spring-cloud-starter-sleuth: attaches aa ID to each request( ignore , if already added )

spring-cloud-sleuth-zipkin: tells spring to log messages in zipkins format

spring-cloud-starter-bus-amqp: tells springs that the default MQ is rabbit MQ.



**Step5: Using Zipkins UI**

**Step 5.1 : Launching order :**

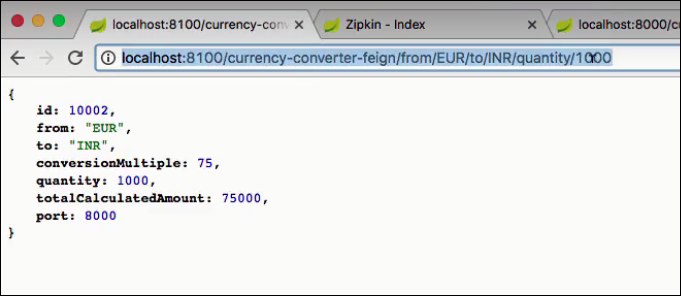
1. Eureka Naming Server
2. Zipkins Distributed Tracing Server
3. CES
4. CCS
5. NZ-API Gateway Service

allow enough time between each application launch.

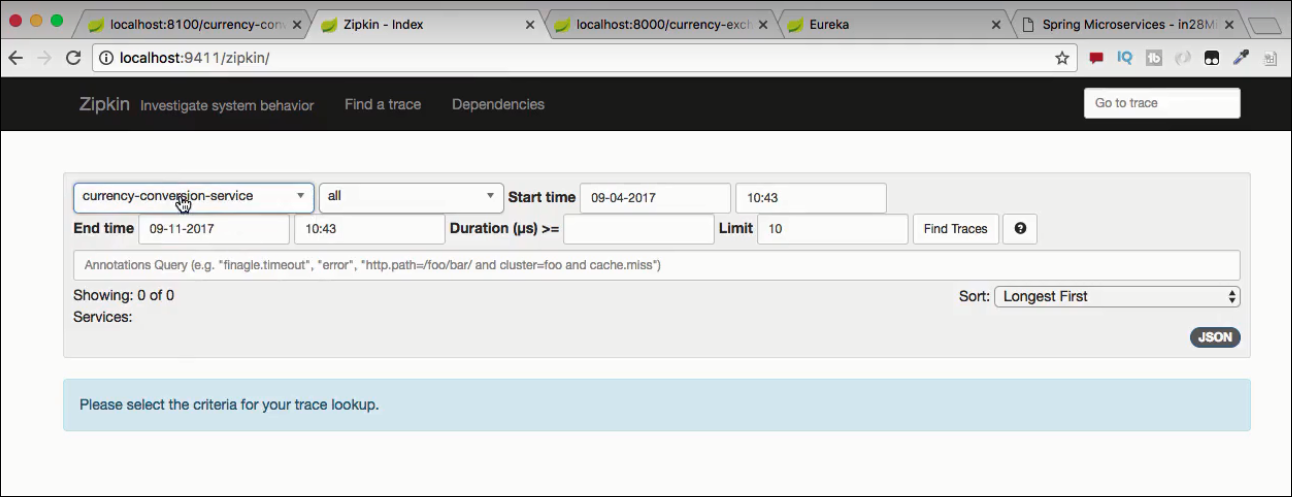
**Step5.2 : Verify on Eureka Naming Server**

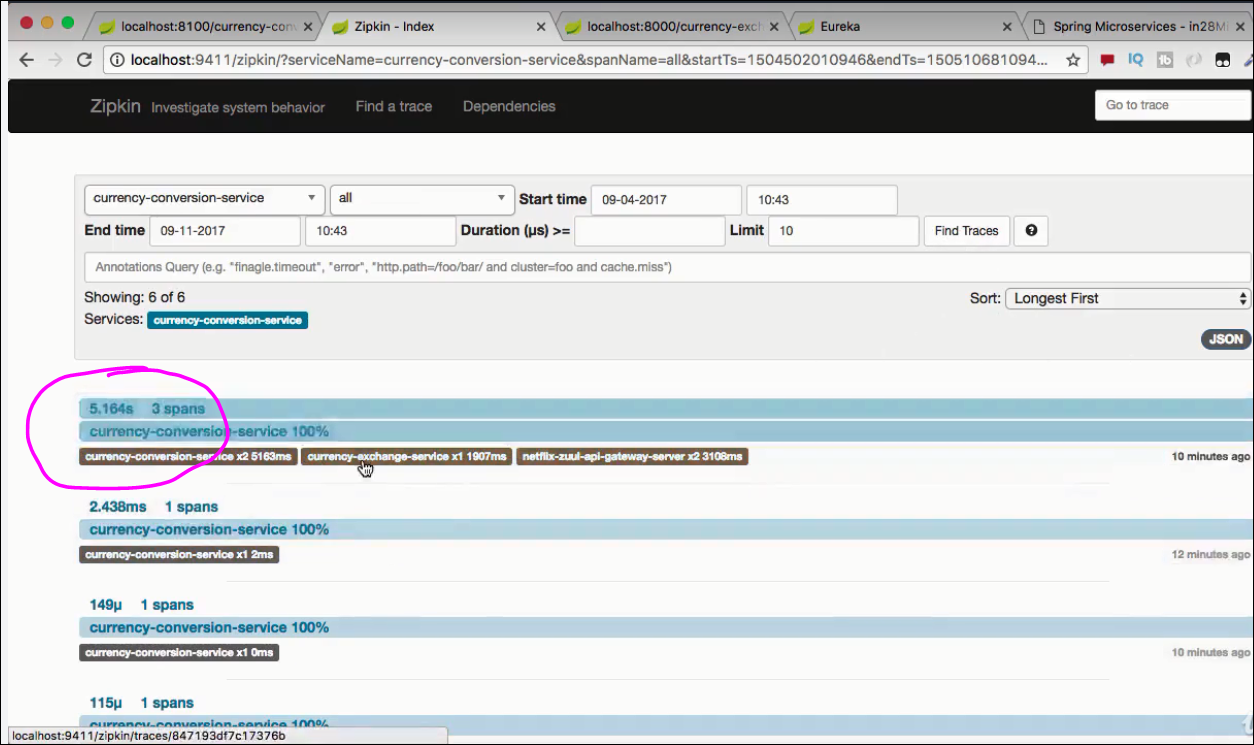
[**http://localhost:8761/**](http://localhost:8761/)

**Step5.3 :**

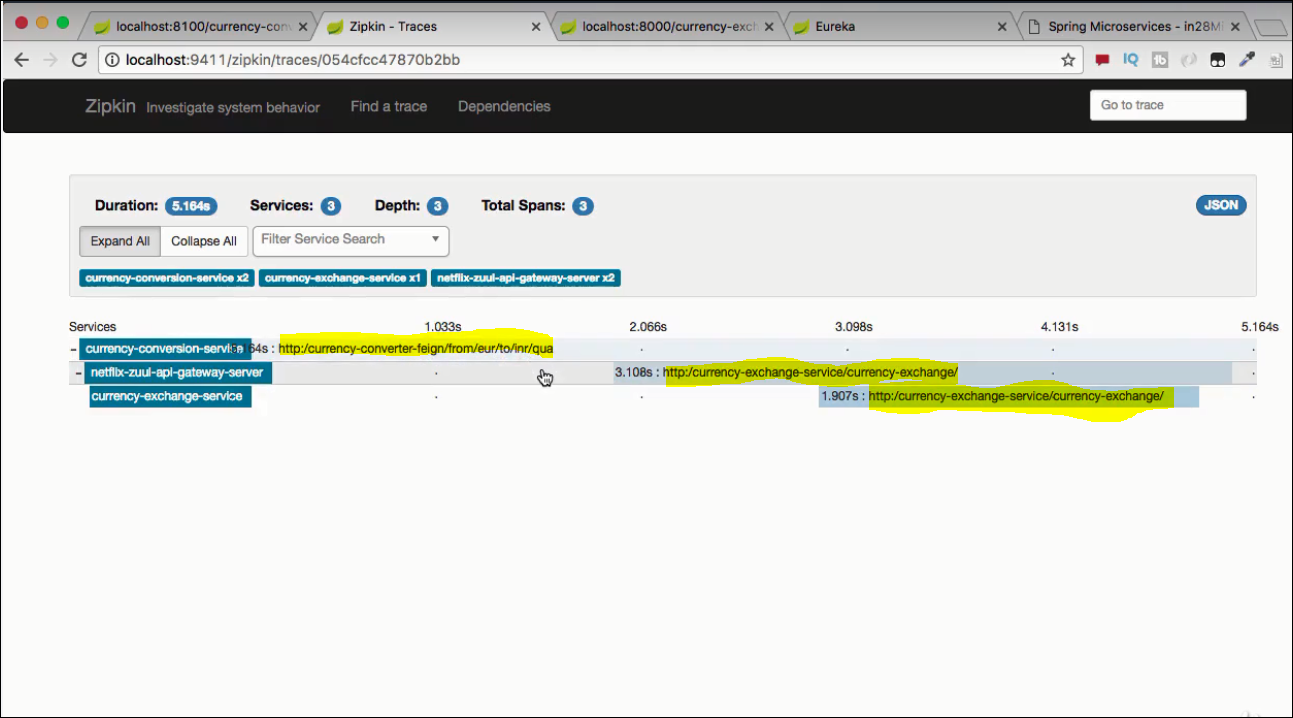
****

Step5.4:





Now u can see the trace of that request:



Click on the CES to see the details of this request inside the CES:

