Dev Setup

Prerequisites:

- Java 1.8 (tested with java 1_8_302; you can install it using sdkman; link is in reference)
- IntelliJ (tested with 2020.3)
- Docker Desktop(tested with 3.4.0) (make sure you allocate atleast 6gb memory)
- Project code
- maven (tested with 3.5.4)
- Necessary configs (For Kafka Influx Connector)

Steps Involved

- Import the java code into Intellij as **maven project**. Make sure build is successful by clicking on build button under *Build* option. For double check, run `mvn clean package` in terminal and check its success or not. (check here how to import guidelines)

- Edit the **volumes** in the docker-compose and provide a local path . Below are the three volumes which needs to be updated as per local path. Note that right side path of ':' no need to be changed.

> Take the absolute path of the root of the project and paste in the localpath like shown below

For Influx:

- <***/localpath***>:/var/lib/influxdb

Eg: **/Users/Documents/Spark-Streaming**:/var/lib/influxdb

For Grafana:

- <***/localpath***>:/var/lib/grafana

Eg: **/Users/Documents/Spark-Streaming**:/var/lib/grafana

For Kafka-connect:

- <**/localpath**>:/usr/share/confluent-hub-components (take the absolute path of the folder **config/kafka-influx** from the project root and paste it in localpath)

Eg: **/Users/Documents/Spark-Streaming/config/kafka-influx:**/usr/share/confluent-hub-components

> Without above config change code will not run

- Open a terminal in the IDE or in CLI and go to the root of the project. and run command `docker-compose up -d .` The result of the command should look something like below:

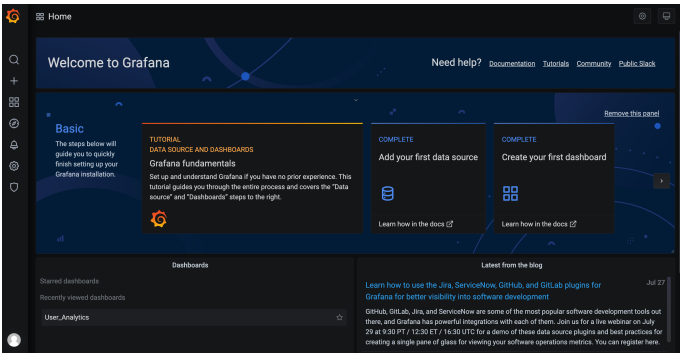
```
Starting mysql          ... done
Starting influx_grafana ... done
Starting zookeeper      ... done
Starting broker         ... done
Starting schema-registry ... done
Starting rest-proxy      ... done
Starting connect         ... done
Starting ksqldb-server   ... done
Starting ksql-datagen    ... done
Starting ksqldb-cli      ... done
Starting control-center  ... done
```

- Check whether all containers are in UP state by running `docker-compose ps`. If any of the service is not in upstate, then run `docker-compose up -d` again. Once the services (every docker container) is up, Kafka takes sometime to be up as it starts zookeeper, broker, etc.

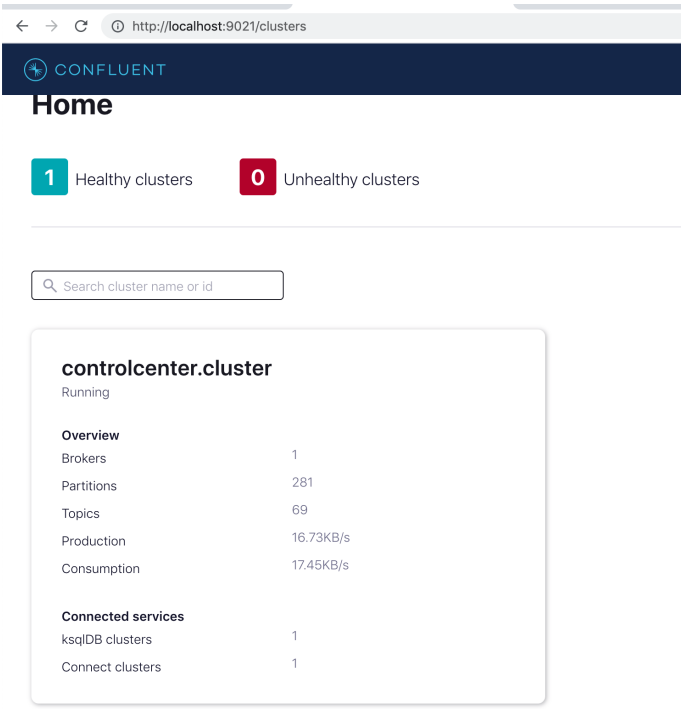
Validation

Access UI of services (Grafana, Kafka) via browser

- For Grafana: http://localhost:3003/ . login creds are user: **admin**, password: **admin**



- For Kafka-Control-Centre: http://localhost:9021/clusters . Takes sometime to be up .



- Connect mysql with any mysql client tools and check the connectivity with below details. I use intellij dbconnect plugin. It can be anything DBeaver per se. Provide a jdbc jar to the tool as well. The test should be successful.
    - host : **localhost**
    - port: **3306**

    - user: **admin**
    - password: **example**
- Connect influx shell: login to the shell of the influx docker container from the root of the project.
    - `docker-compose exec influx_grafana sh`
    - Once logged in, run `influx` . It should open a shell like below.

Troubleshooting

- If any docker service is not coming up. you can check the logs of a docker container running this command at project root: `docker-compose logs -f <servicename>` . Eg: `docker-compose logs -f broker` .
- Make sure all the ports are available for the docker services. Refer the docker-compose.yml

file to check all the necessary ports under **port** section under every **service**.

Reference Links

- Install Java using SDKMAN (command: **sdk install java 8.302.08.1-amzn**)
- Docker Desktop
- Intellij
- Spark Documentation
- Kafka Confluent
- Kafka Quickstart
- Influx
- Grafana
- install maven
- install maven with default settings.xml