→ How does API composer work.

→ we will write API composer ~~host~~ service here. which will intern talks to individual micro services like raw material service, space-production service and marketing-management service.

→ Now the API composer microservice have demand and supply. (or) (production analysis service) how much we can produce. we have production analytic data. If is dummy micro service it makes service Call. Each of these microservices, Aggregate the data and provides the data to the caller.

i.e. aggregate the data for 3 microservices and will provides ~~caller~~ data to caller.

(Asynchronous Call → All 3 microservices we want to do parallel and aggregate the data parallely → i.e. Restful services) (not related).

→ our automobile production service will now Call whoom production analysis service (API composer microservice) which will get the aggregate the data always.

→ Never we don't know this information is coming which micro service. so the microservice not ~~tightly~~ coupled to other.

→ loosely coupled to each other. through API Composer.

api composer

→ I have automobile production service is try to communicate raw-material management service spare-distribution-service like to get some information for example demand-and-supply information. what is the demand in the market how much we supply in the raw material to plan for a production

→ automobile needs to manufacture because to plan for production. So it demands and supply. Suppliers how many we have and demand/distribution we have marketing service.

→ we Automobile service need to query the data from raw-material service, spare parts, marketing service aggregate the data has to used and combine all data has to used.

→ Now such a information we are deriving here. So many microservices need all the data we have combined and used, so data should be duplicated.

→ Each microservice directly talk to the other microservice is tightly coupled. If there is change in one microservice is other microservices will be impacted.

→ that is where API Composer coming to picture

If more no of network calls.

→ The data we are trying to access now production analysis service (api composer) every 10 ms data.

→ If more no of times if we are calling to Api composer what will happen eventually degrades the performance system because more no of calls and more no of aggregation is there ~~Off~~ only. which result in poor performance ~~of~~ the application.

→ How to overcome this CQRS is introduced

CQRS (command query responsibility segregation)

→ How can we microservice is communicate each other.

→ Depends on our requirement when our talking to other microservices 2 to 3 microservices we want to aggregate the data if the data sets ~~calls~~. are ~~and~~ very less and less frequently used the we can for API composer.

→ If we want to get the data from one microservice means directly make a service ~~through the rest template.~~ call, dont go for API composer, ~~if we~~ dont have data aggregation requirement we dont want to talk multiple micro services getting the data when one micro service is talk to another microservice through the rest template we can call.

if we want to combine the data and want to access then only we can go for API Composer.

CQRS

→ we are frequently access the data from across the microservices we are aggregate and taking. suppose we go for API Composer large volumes of data inefficient memory and out of memory exception (CPU) more no of calls performance issue came.

ans for CQRS design pattern.

→ How do we need to design one view only database with respect to tables we wanted to aggregate the data. this database will never be performed update/delete operation only querying the data we processing the data only populated here. view only database here. we will take.

we will take a microservice here we try to write product trends and analysis microservices. that required to all the micr will serves what product trends data i.e. required by whoom all the microservices. most demanding products / popular products (marketing information and Sales information) we need in this way multiple other requ both together we can find popular products otherwise we cannot i.e. How can

aggregate by using Product trends and analysis service in this way multiple other requirements also we have.

Now How does this micro service we have to get now which has to talk to multiple other microservices (like marketing service, inventory management service which products are which stocks, spares Products service), to get this information if all our product trends and analysis service is talk to 3 other micro services.

Every 5 minutes the request will comes to product trends and analysis service for querying the data, every 5 minutes the request will coming now How many times we need call the other microservices i.e. Every request the product trends analysis service it has to make a service a call and aggregating the data from other microservices and pulling the data.

→ frequently update quite oftenly thats the data of the marketing service, like remaing micro services keep on frequently get updated may not be.

→ Even then also for each service to call the product trends service repeatedly query the data of other micro service se Performance issue coming.

→ aggregate data. } CQRS, API composa.

→ Data base per service design pattern we dont need.

→ Instead of frequently called.

(① cpu cycles wasted ② network connection more)

CQRS comes into picture. (Kafka where is we will used in microservices)

→ There is an change/update in marketing management service data has been modified due to some operation the immediatly marketing management service has to publish an event. Now there should be bunch of event handlers will registered in ~~bunch~~ product trends and analysis ~~[illegible]~~ microservice (aggregate logic service).

(where are we using Kafka. this the place we are using kafka in our project.)

→ when ever event has been to published now who will be called now (when marketing service has published event to change/update data who will called now) event handler.

Inventory management service there is a change in the stock value immediatly ~~the [illegible]~~ he will publish an event, when he publish an event where does request go to event handler will consume the ~~[illegible]~~ event. similarly the spares production service will change in

change in new stores in product, one event will triggered / published. event.

→ when ever there is a change in underlying data for every action/operation. he will publish an event with the relavent data.

→ when ever individual microservices of publish an event when there is a change now the corresponding event handler i.e. product trends and and analysis micro service called. The product trends and analysis service when it was called with latest data and it will aggregate the data and will store the data in view only database now.

A change in the data will be updated in view only database.

→ when the people are asking popular products now when the send the request to the product trends and analysis service for getting the popular products now immediatly product analysis micro service should make a ~~should~~ service callout (or) always they have latest (i.e. running microservice) snapshot of data view only database now?

we use view only database.

→ they can simply get the data and return, so where is the data aggregation happen now? at

the database level

→ 30 calls we got with in the 5 minutes for getting the product analysis data 30 calls,

→ New 30 micro service calls and 30 times we need to do data aggregation? NO

→ when ever there is change in data individual microservices there are publishing that changes to the product trends and analysis service which has computed and stored in view only database so that we can fetch that existing data can return quickly here where is the performance issue. so large volumes of data also handle by using Cqrs design pattern : so to avoid network consumption, recompute the same data CPU cycles will increase so one time data will be querying and one time only data should be transmitted for every change. and same data will be cached and will be serve to the clients.

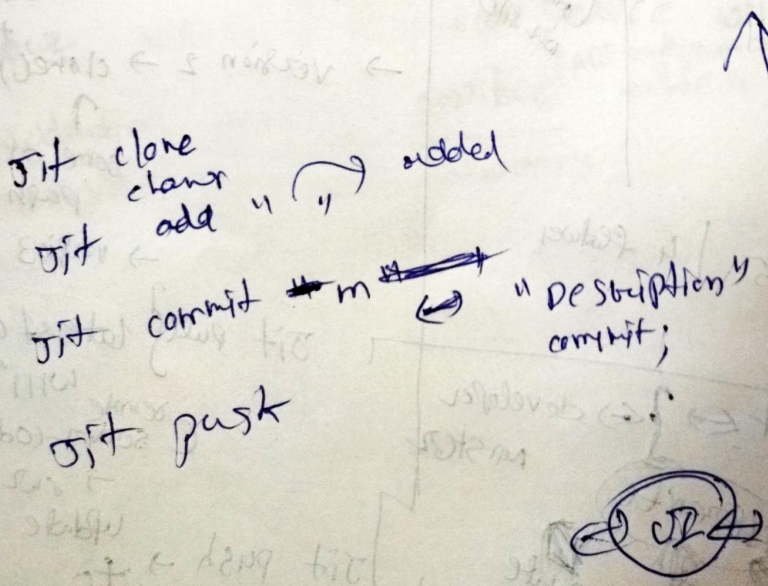
So that all the issues are resolved using Cqrs .

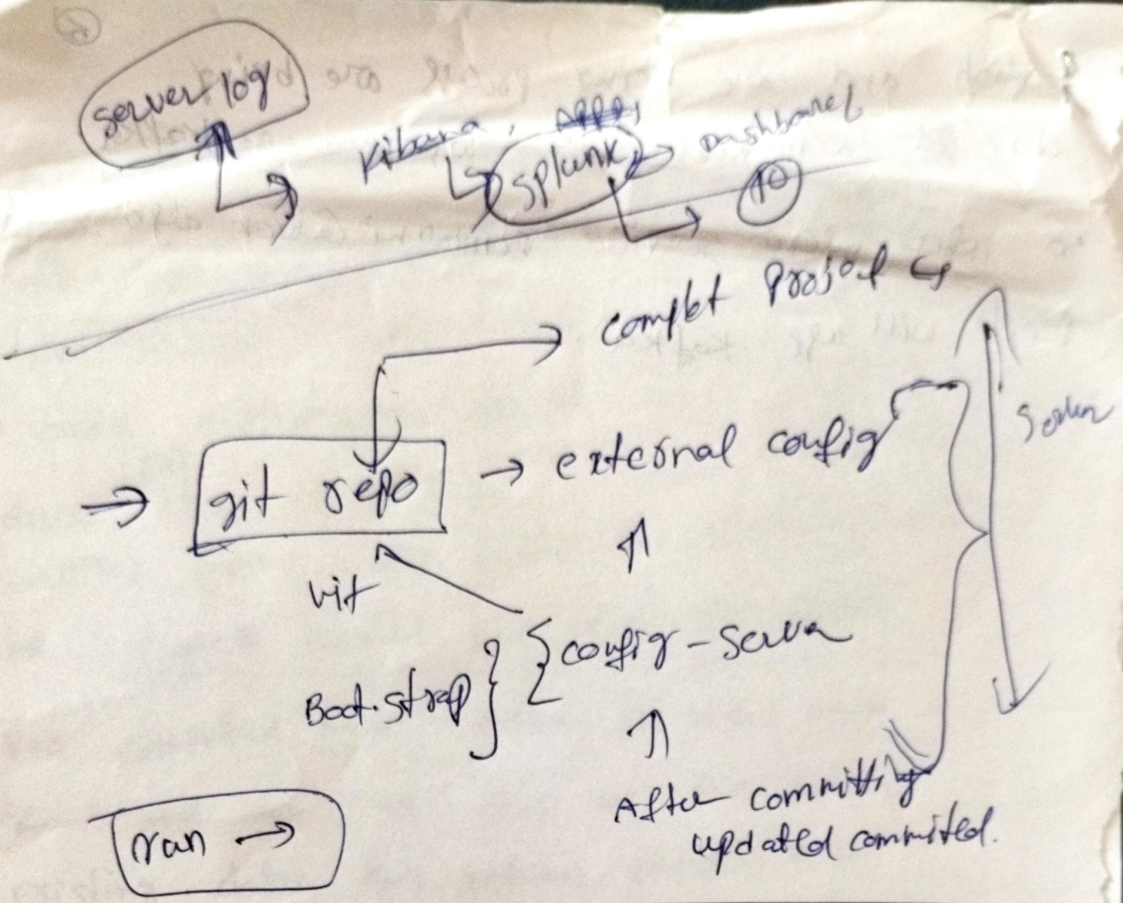
→ that is where for Cqrs microservices.

→ How does the every change in any of the microservices notified to the Cqrs service through event and we consume the events, the frequency which the events are published and are

publish and are being process are being done at a massive scale. we need use kafka.

so inter micro service communication also people will use kafka.

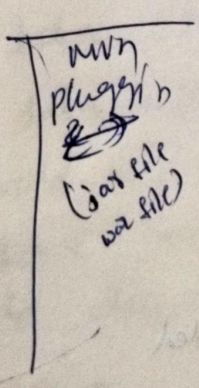


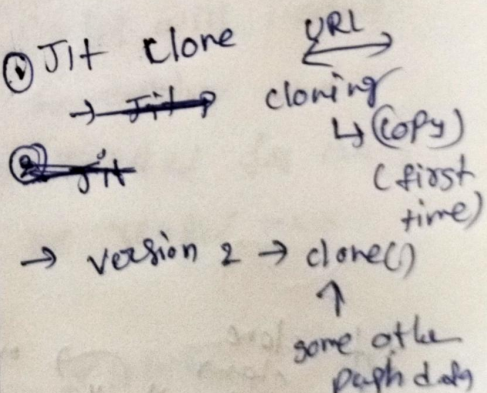


→ we will user cloud-config-server to externalize all the configuration data.

we will

we will give





Junit | 5 | 4 features

@Mock