Micro Service

یک طراحی بیرونی است که به معنی شکستن یک برنامه بزرک به قطعات کوچک است.

فرض کنید که یک اتاق داریم پر از کارمند که با پارتیشن جدا کردیم مثلا فنی، حسابداری، ارتباطات، و در کنار هم کار می کنند و خروجی را به مدیر عامل می دهند.

اما با ماکروسرویس می توانیم هر بخش را جدا کرده به ساختمان های در محل های دیگر بفرستیم و در نهایت آن بخش ها کار خود را خروجی گرفته و به مدیر عامل بدهند.

**Soa**

معماری قدیمی که نام آن service oriented است که اپلیکیشن را به بخش های کوچکتری تبدیل می کند. در واقع میکروسرویس وام گرفته از آن است با این تفاوت که در میکروسرویس سرویس ها ما خورد تر میشود اما در soa در حد یک اپلیکیشن می تواند مستقل شوند.

**نکته :** در میکروسرویس می توان یک انبار را به سرویس های مستقلی تبدیل کرد به شرط این که مستقل بودن حفظ شود.

میکروسرویس در واقع External Architect است.

فرض کنید یک پروژه داریم با 3 تا سرویس a,b,c و هر کدام از این ها خود دارای یک معاری هستند.

یکی clean یکی union می تواند باشد.

**بین سرویس های می تواند ارتباط باشد به صورت 2 طرفه.**

**روش های ارتباط بین سرویس ها:**

GRPC

Google Remote Procedure Class

راه دوم Message Broker

به این صورت که کار ارتباط بین سرویس ها را انجام می دهد

چه کسی بگیرد.

ارسال و دریافت دیتا در یک صف و بررسی به ترتیب.

یک نرم افزار realTime است که صف را گوش می دهد و داده می گیرد و می فرستد.

RabbitMq – Kafka – redis – zmq – massTransiaent

**همه این ها از پروتکل Maqp استفاده می کنند.**

مانند یک تسمه عمل می کند داده را می برد و پاسخ را برمیگرداند.

**Standard**

Exchnage : مثل یک سری ادم هتسند که بسته ها رو تحویل آن ها می دهیم و می گوییم بزار تو فلان صف یعنی مشخص می کند که بسته ما در کدام صف قرار بگیرد چون می توانیم صف های مختلفی داشته باشیم و بگویم با چه قانونی در صف قرار بگیرد بسته.

برای ارتباط بین سرویس ها مثل سرویس user – information از روش prodeuce , consume یعنی ارائه دهنده و مصرف کننده که با rabbitMq حل است.

ابزار های مورد نیاز را می توانیم بر روی Docker بالا بیاوریم.

**Seq یک دیتا بیس مناسب برای ذخیره لاگ ها است.**

در این پروژه از Redis برای اطلاعات سبد خرید و elasticSearch نیز برای search کردن استفاده می کنیم.

**نکته: باید در میکروسرویس هر کدام از سرویس ها مستقل از هم باشند.**

**نکته »** باید توکن jwt در همه سرویس های تزریق شود به صورت common و در همه معتبر باشد.

**یک ApiGateWay می سازیم تا با استفاده از توکن از به سرویس های خود دسترسی داشته باشیم.**

**Cqrs** برای جدا کردن کوئری های select و insert می توان از database های مختلف به صورت cluster استفاده کرد به صورت always on avalibility group چند سرور داریم با چند sql که یک سرویس بین این ها run است به نام Failover یعنی یک سرور Primary داریم و مابقی secondery که به صورت sync و async می تواند باشد.

داده های نوشته شده روی جدول های primary بر روی secondery ها نیز نوشته می شود و اگر primary از بین برود یکی از secondery ها تبدیل به primary می شود و با ip به نام listener با هم در ارتباط هستند.

در Cqrs مثلا به یکی از secondery ها به صورت readOnly برای select زدن استفاده می کنیم.

MediatR برای پیاده سازی Cqrs کاربرد دارد.

ApiVersion هم برای ورژن بندی api ها کاربرد دارد