What are Microservices?

Description

Microservices are small autonomous services that work together to achieve a business feature. Before microservices we used to have applications that are monolithic, meaning the entire team used to work on a one single huge code base which was actually difficult to maintain.

If you have worked as a developer then you must have noticed that teams start with well written code at the beginning, but as the time passes the code base grows huge with new and changing requirements and it becomes hard to maintain. When the issue arises itâ€™s difficult to find where to fix it and sometimes fix in the code breaks some other feature. More and more emphasis should be given to write more cohesive code. Cohesion means grouping the related code together. One of the design principles â€œSingle Responsibility Principleâ€• states that â€œGather together those things that change for the same reason, and separate those things that change for different reasons.â€•

Microservices takes the approach of independent services. We focus our service boundaries on business boundaries, making it obvious where code lives for a given piece of functionality. By keeping this service focused on an explicit boundary, we avoid the temptation for it to grow too large.

Microservices ecosystem is a platform of services, and each service encapsulates a business capability. Business capability means what a particular business does in that domain to fulfill its objectives and responsibilities. Microservices have an independent life cycle. Developers can build, test and release each microservice independently. We can have small autonomous teams each responsible for one or multiple services. Contrary to general perception and â€˜microâ€™ in microservices, the size of each service matters least and may vary depending on the operational maturity of the organization.

One thing we have to keep in mind is that migrating to microservices is not a good idea for some monolithic applications E.g. Applications which are used by a smaller audience and do not have any scope to grow. Also you can think of scalability as one of the factors i.e. you have to ask yourself that, is your application used excessively at some point of the year that you require multiple instances of your application? In that case break your monolithic into microservices and only scale that feature alone instead of scaling all the microservices or entire monolith. Scaling the entire monolith is also fine if it’s a small application.

Tough this post included minimal information, I would recommend you to go over the benefits of microservices post to get a more clear picture.

Category

1. Microservices

Date Created
March 29, 2021
Author
kk-ravi144gmail-com