UNDERSTANDING RELATIONSHP AND DIFFERENCES BETWEEN APIs AND MICRO-SERVICES

[1]Nitheesh, [2]Prof. Suma B

[1]Student, [2]Assistant Professor,
[1]Computer Science and Engineering,
[1]RV College of Engineering, Bangalore, India.

Abstract : When discussing software architecture and integrations one've probably heard two extravagant sounding terms tossed around: APIs and microservices. The two ideas are vital to web application improvement and plan today,and their uses certainly overlap. However, it's important to recognize the differences between microservices and APIs and how they're applied. This way, one can have more productive and insightful conversations with developers and better understand the applications you use and sell. In this paper, we'll start by defining APIs and microservices separately: what they do, how they work, and why they're important. Hence, we'll perceive how APIs and microservices fit together and differences among them..

***IndexTerms* - APIs, Micro-Services, Monolithic.**

I. INTRODUCTION

For the past couple of years, microservices have beena hot topic of conversation in the field of informationtechnology. The hype around microservices is that they make everything better, and as an engineer whoworks on an application, we are enthusiastic promoter of the Micro-Services and Web-Services such as APIs. APIs being used extensively in the software industry. Many Companies are shifting from Monolithic to Micro-service Architecture at fast pace. Therefore understanding the Working of APIs and Microservices is very much necessary.
.

II. APIs

An application programming interface (API) is the part of an application that communicates with other applications. An API is a group of protocols and methods that define how two applications share and modify each other's data An easy way to think about the API is to think of a contract of work that you can ask for a particular service. APIs are used in various web applications such as today's social media and banking software. External applications can interact with other applications through standardized contracts. For instance, let's say a developer is building an application that's going to integrate with the system A. Hr would be able to use the data inside system A, through A's API. The API simplifies the complexity of trying to use the data inside Facebook and provides an easy-to-use way for the developer to access that data. An API sits between the core software components and the audience, and external developers can access partsof the app backend without understanding how things work inside the app. This is what makes an API an interface for programmers.

Extensively speaking, APIs permit designers inside and outer to get one of two things done: access an application's information, or utilize an application's functionalities. At last, this is the means by which the world's gadgets, applications, and website pages are connected up to communicate with each other and work together. While numerous APIs are made for outsiders to utilize — alleged public APIs — the expanding fame of a microservice architecture has prompted the making of increasingly more private APIs. In this case, APIs act as a lightweight solution allowing individual microservices to communicate with each other. From a technical point of view, APIs typically send data using HTTP requests. These responses return a text response, usually in JSON format, which developers can use as they see fit. Types of API design styles include REST, SOAP, GraphQL, gRPC and others.

III. MICROSERVICE ARCHITECTURE.

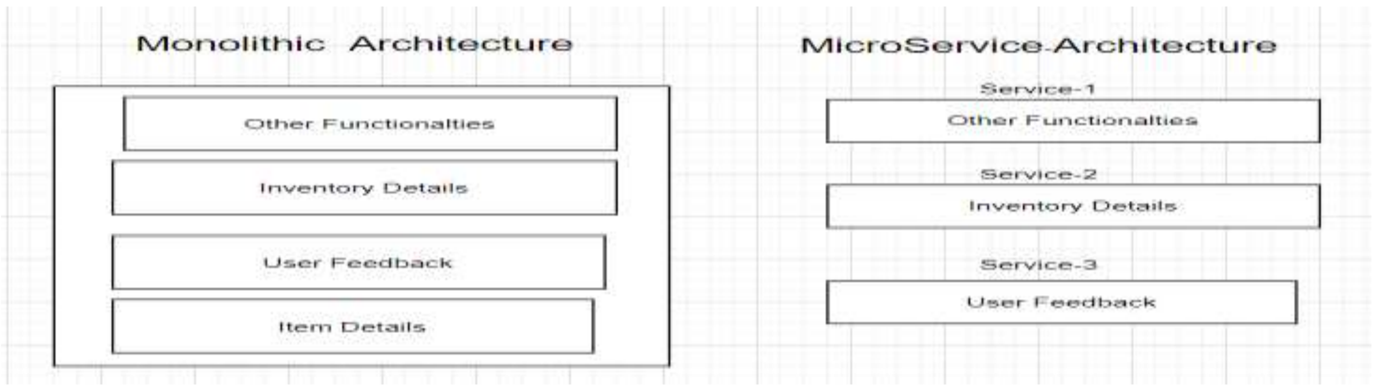

Fig-1:Comparision between Monolithic andMicroservice Architecture

Microservices are a type of software architecturethat divides the various functions of an application into smaller components called "services". When an application is built this way, it is said to follow a microservices architecture[2]. The way of thinking behind microservices is that your application is made out of an assortment of units, every one of which gives the usefulness to a solitary businesstask. To illustrate this let's consider an example ofE- Commerce website , when a person clicks on a item page various details arises such as item details hether it's available in stock or not and also if person already bought that item he can give review to it.Previous to Micro-service Architecture we had Monolithic architecture where all these functionalities embedded under one single unit (fig- 1) where as in Micro-service each of the functionalities will be treated as different units which are called services.As one can see inventory update is a different service and it is independent from services like item details and user feedback services.

And these different units can communicate with each other with the help of APIs. Benefit of this approach is that if one wants to make a change in inventory Details services only that service needs to be updated and deployed, instead of the entire application as in Monolithic-service.

IV. RELATIONSHIP BETWEEN APIs AND MICRO-SERVICES.

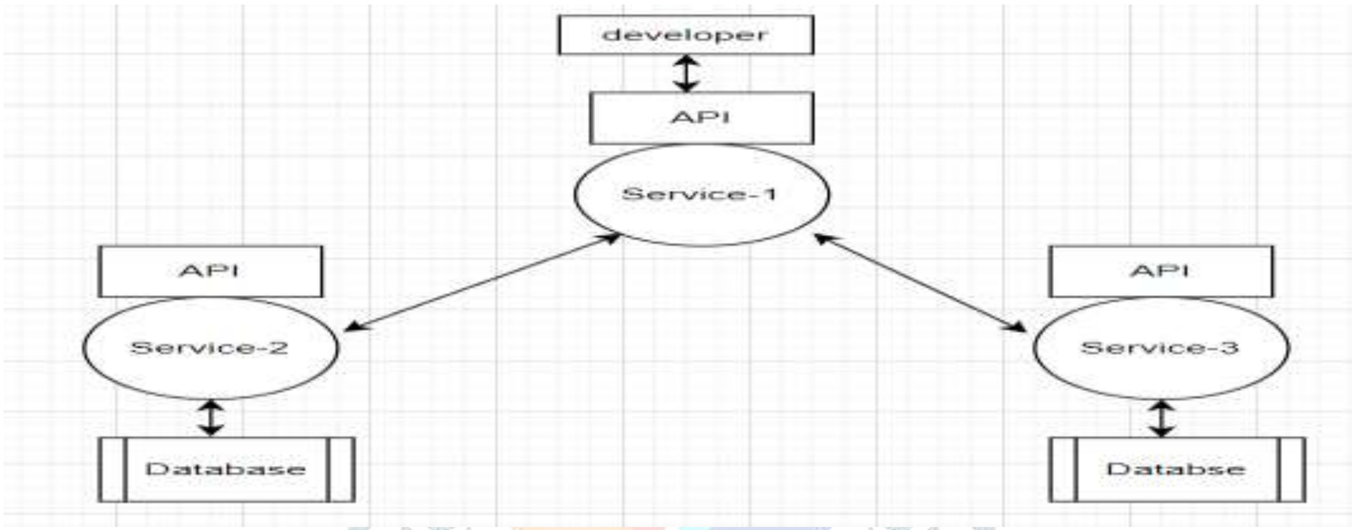

Fig-2:Micro-Services communicating with each other.

API has contract details about how a services in Micro-service architecture can be handled.API sits on top of ofa service as shown in fig-2 and services communicatewith each other with help of the APIs.

For instance now a developer needs information regarding item's inventory details ,item details .Firstly he will make one call to service 1 which internally communicates with service-2 and service-3 to get details regarding inventory and item details aggregates the data and adds some other information toit and sends it to developer.
An added benefit of having an API is that you can update the inner workings of your service, and as long as the service still meets the API contract specifications, customer services will not be affectedby the updates.

V. DIFFERENCES BETWEEN APIs AND MICRO-SERVICES.

An API is a contract that provides guidance to consumers to use basic services. Microservices are an architectural design that separates parts of a (generally monolithic) application into smaller self- contained services. According to the definition, this means that APIs are generally part of a microserviceand can interact with the microservice itself. Another way to think about this is for the API to act as a contract for interactions within the microserviceand provide options that can be used to interact with the microservice. An API is the part of a web application that communicates with other applications. A software API defines a set of acceptable requests made to the API and responsesto those requests. Microservices is a method for building applications that decomposes the application's functionality into modular, apartment- like programs as well as how applications use public APIs to integrate with other applications, one component of a microservice uses a private API to access another component of the same microservice. In microservices, each service has its own API that determines which requests and responses it can receive[1].

Microservices are a type of software architecturethat divides the various functions of an application into smaller components called "services". When an application is built this way, it is said to follow a microservices architecture. The way of thinking behind microservices is that your application is made out of an assortment of units, every one of which gives the usefulness to a solitary businesstask. To illustrate this let's consider an example of ECommerce website and when a person clicks on a item page various details arises such as item details, whether it's available in stock or not and also if person already bought that item he can give review to it.Previous to Micro-service Architecture we had Monolithic architecture where all these functionalities embedded under one single unit (fig- 1) where as in Micro-service each of the functionalities will be treated as different units which are called services.As one can see inventory update is a different service and it is independent from services like item details and user feedback services. And these different units can communicate with each other with the help of APIs. Benefit of this approach is that if one wants to make a changein inventory Details services only that service needsto be updated and deployed, instead of the entire application as in Monolithic-service.

VI. CONCLUSION

Maintainability and code quality are two important elements of a successful IT strategy. Microservices help everyone to stay with them. They keep team agile and help engineers to meet Microservices is an architectural style for web applications where functionality is broken down into small web services. while APIs are frameworks through which developers can interact with a web application. As we mentioned, there is certainly some overlap between these two services, as many microservices use APIs to communicate with each other.

REFERENCES

[1] Kapil Bakshi,"On Microservices based Architecture Approaches " IEEE Aerspace conference 2017.

[2] Sam NewMan ,"On Building the Micro-services." Texbook published on 2015.

[3] Andy Neumann, Nuno Laranjeiro, Jorge Bernardino "On Analysis of public Rest APIs" June 2018,IEEE transactions on Service Computing.

[4] .Jacek Kopecký, Paul Fremantle, Rich Boakes,on "A history and future of Web APIs" january 2014 ,Information Technology 56/3.