**Introduction to Microservices to Beginners from a Beginner**

Hello reader, in this blog/article I will try to convey of what I have understood of microservices. Please note that I am still a student of this technology and I am nowhere near to the professionals in this field. This will be my attempt to explain microservices to someone who is point blank about this topic. So let us start, sit back , relax and enjoy hopefully.

**The Beginning**

Lets start from the beginning to fully understand why do we even need microservices. When software production was booming, people started to realize the power and efficiency which came with working on computers. Software engineers were developing massive applications for businesses, government institutions and so much more. These applications were huge in size and were not easy to maintain. The experts in the field came up with the idea of **Divide and Conquer.** In order to maintain the readability of the code, developers started to modularize their files and applications. This means that the functionality of the software was distributed in different files and modules. Programmers kept their modules **Loosely Coupled and Highly Cohesive.** Every module in the program had very little interference with other module and each of them had unique functionality.

But chaos is unavoidable, as the applications started to grow the boundary between the modules started depleting. This means that a little change to one of the code in a module would start a chain of change on code in other files and modules. This type of software development architecture was known as **Monolithic Architecture.** This architecture has been around for quite a while and effective for producing trustable software but with the modernization and development of high end technology, monolithic architecture is not effective. All of the software is written in a single language and composed of millions of lines of code. With good high level languages in the market which gets the job very well done, adaption of these languages into the developed software is quite difficult and time consuming.

But wait, the problems keep coming.

New developers consume too much time to fully understand the whole application which decreases the productivity. Each new update requires the developers to test every functionality and modules for bugs. As the application becomes tightly coupled instead of loosely coupled there is high chance that a little addition to one function can cause errors in other 10.

**Not a solution, An Effective Strategy**

With modernization taking over the world drastically, software world was affected too. Experts felt that the monolithic architecture is a hurdle in between the achievement of modern technology. It lacked adaptability, was very memory intensive, difficult to maintain. With all of these problems in mind **Microservices Architecture** was developed. The simple definition of this architecture would be: “The compartmentalization of functionality so that no functionality affects the other in anyway”. Well this means that one module of a software and can run on a different server and the other module can run on other. They all are connected to each other with a physical connection.