**Microservices: -**

* Microservices is an architecture for software development.
* Microservices is not a first architecture while previous many architecture in software development.
* Previous architecture is monolithic, in this architecture what limitation, short comings, advantages, disadvantages and what reasoning those software developer or computer scientist or researcher move to new or another architecture as like microservices

**First commit……!**

**Background: -**

* During the early days of computer science/software development, barriers to entry in programming languages were high level languages
* Cannot understand any graduate or master level of science/computer students
* In 1960s, it was a time using a programming and software development its much difficult or complex task
* Most limited people can use programming languages
* Only PhD in science and computer can use these programming languages
* Or mathematics and researcher can use these programming languages
* As at that time, nearly all use of computer required writing custom software
* Because limited people have computer and early era computer does not efficient as compare to now a day’s computer.

**Second commit……!**

**Developed BASIC programming languages: -**

* In 1964, a general -purpose programming languages **BASIC** was develop.
* It is low level programming language, so easily can use non-PhD and all department students and also write programs.
* Now day by day is rapidly growth up in computing appliance in the 1960s, software became large and complex.
* Computer scientist tried to tackle the complexity of software systems with the ancient and proven technique **Divide and conquer**.
* **David Parnas** published his paper **“On the Criteria to be used in decomposing systems into Modules”.**
* In 1972, **David Parnas** introduced the concept of **Modularity** and **Information hiding.**
* **Edsger W. Dijkstra** introduced the concept **“Separation of Concern** in his paper **“On the role of scientific throught”** published in 1974.

**Third commit…..!**

**The Work Of two scientist/researcher: -**

* The work of **David Parnas, Edsger W. Dijkstra** and other lead to the **Modular Software Development** in 1970.
* With the principle of decomposing a large, complex software systems into **“Loosely coupled, highly cohesive”.**
* In simple terms, **“Loosely coupled**”means the dependency between modules should be very low and **“highly cohesive”** means that one module should focus on single or similar functionality.
* In 1990s, rise the internet and web, Software growth up in business applications and became even more complex and large.
* As such we use the modularity and reduct the complexities of software applications.

**Fourth commit…….!**

**Period of Internet: -**

* So mostly people use the internet and those are move to desktop applications to web application
* But it did not help full as the soft modular boundaries of software sub-systems are easy to **cross and minus.**
* The computer scientist think is and developed the new architecture became a very popular architecture during the 1990s, the new architecture name is **Layered Architecture,** it is purpose of business applications.
* Normally, a business Web Application is divided into several layers:

1. Presentation layers
2. Business layers
3. Data base layers

* In 1997, **Brian Foote and Joseph Yoder** has analyzed many Business application and published the **“Big Ball of Mud”** papers.

**Fifth commit…….!**

**Differentiate Architectures**

**Monolithic Architecture: -**

**Application Scaling: -**

* As such web scaling are rise up and software belongs companies growth up and also enjoy the benefits.
* We also need to supported high horizontal scalability in software
* E.g. CPU intensive or **I/O intensive** need to be scaled and handled separately (implemented with polyglot programming).
* **Monolithic software** works as a single unit and developed in a single programming language using a single **Tech Stack.**
* **Monolithic Software** only supports one programming languages, it is impossible to implement one single module of it in order programming language or in other **Teck Stack.**

**Sixth commit…..!**

**Development Velocity: -**

* It is very fast development velocity and growing up the software companies.
* To shorten time to market, every company nowadays wants u have **fast feature development.**
* It’s very huge **Monolithic Applications**, **adding new feature** slowly slowly because such a Monolithic Application gives **huge Cognitive Load** to the Developer.
* Modules of giant Monolithic application are tightly coupled and provide an additional challenge to add new features
* As a result, **adding new features** in a Monolithic application become **very expensive**

**Seventh commit………!**

**Microservice Architecture: -**

* In the 2010s, near the during years much disclosure in technologies with arise which impact the software Development landscape in a significant way.
* As like new technologies **Cloud Computing, Containerization (Docker, Kubernetes), DevOps**.
* During this year developed the new programming languages and **highly productive, lightweight languages e.g. Golang, Rust, Swift** comes to scenario.
* New development languages this time period and some **highly productive, easy to use, lightweight** programming languages just like **JavaScript, Python** become mainstream
* Major changes in technologies and **IT** department, we also change **water fall** to **Agile Software development**.
* Desktop Hardware also changed heavy with cheaper, faster main memory ram/rom and rise of **Multi-Core CPU, GPU.**
* Another new **Database** technologies as like **NoSQL, NewSQL** emerges and become mainstream
* To handle the complexity of modern software application
* To take the advantages of **Cloud Computing, Containerization, DevOps**.
* To get benefits from modern Programming languages

**Eight commit………!**

**Definition: -**

**Microservice Architecture is about decomposing a Software System into autonomus Units which are independently deployable and which communicates via lightweight , language agnostic way and together they fulfill the business goal.**

**Advantages: -**

**Application Scaling: -**

* Firstly, Microservices are often **Stateless** and if they are carefully deployed
* Microservices using **Docker, Kubernetes** or using other…
* Microservices high horizontal scaling within seconds and which leads companies as like **Netflix, Spotify, Uber, Google** and other companies to move from Monolithic to Microservice Architecture.
* It could be implemented in a CPU optimized programming languages like C, C++, Rust where other microservice can be implemented in an interpreted languages like Java, PHP.

**Application Speed: -**

* Microservices are often quite small in size.
* **Several hundred to several thousand**
* Due to the size, adding new feature in Microservices are usually faster

**Development Scaling: -**

* As u know Microservices are **autonomous** and independently works or independently development
* Software developer easily work on different microservices **autonomous** without bumping into each other’s code.
* Microservices puts **small Cognitive load** on newly hired Developer or fresh graduates.
* New developer can normally write **productive code.**

**Ninth commit……..!**