Social Media using Load Balance Microservices with Containers for Student

[1]Abhishek Shirke, [2]Akshay Jagalpure, [3]Shweta Deshpande, [4]Ashwini Raut, [5]Prof. Shrishail Patil
[1234]U.G. Student, Department of Computer Engineering, JSPM's BSIOTR, Wagholi, Pune, India.
[5]Prof, Department of Computer Engineering, JSPM's BSIOTR, Wagholi, Pune, India.

Abstract : Student social media using load balance microservices with containers, we have to use four models as Microservices, Load balance, Container, web server (NginX). Using these four models we have to implement the website. Microservices is a distinctive method of developing software systems that tries to focus on building single-function modules with well-defined interfaces and operations. Load balancing improves the distribution of workloads across multiple computing resources, such as computers, a computer cluster, network links, central processing units, or disk drives. Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource. Using multiple components with load balancing instead of a single component may increase reliability and availability through redundancy. Container basically used for storage purpose. container uses Docker algorithm. NginX is a web server. It used as a reverse proxy and load balancer.

Keywords:-Microservices, Load Balancer(Round Robin Algorithm),Containers(Docker Compose Algorithm),Web Server(NGINX)

I. INTRODUCTION

It is system in which we are using services like load balance, container, and microservices. In social media we are uses three micro-services as user service, post service, and university service. Login service is highly secured by using one way encryption with the help of hashing algorithm. In this we uses a web server as NginX. Load balanced is done with the help of docker compose algorithm. Container is used for store the programs and run on any platform, container created using docker.

To explore the role of design in software, consider improves modularity. This makes the application easier to understand, develop, test, and become more resilient to architecture model. It parallelizes development by enabling small autonomous teams to develop, deploy and scale their respective services independently. Services in a microservice architecture (MSA) are often processes that communicate over a network to fulfill a goal using technology-agnostic protocols such as HTTP. However, services might also use other kinds of inter-process communication mechanisms such as shared memory. Services in a microservice architecture are independently deployable. The services are easy to replace. Services can be implemented using different program languages, databases, hardware and software environment, depending on what fits best.

A load balancer is a device that distributes network or application traffic across a cluster of servers. Load balancing improves responsiveness and increases availability of applications. load balancing improves the distribution of workloads across multiple computing resources, such as computers, a computer cluster, network links, central processing units, or disk drives. Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource. Using multiple components with load balancing instead of a single component may increase reliability and availability through redundancy. Load balancing usually involves dedicated software or hardware, such as a multilayer switch or a Domain Name System server process.

Containers are easily packaged, lightweight and designed to run anywhere. Containers are frequently used to run each individual Microservice, serving as convenient, lightweight "envelopes" that allow software to be truly portable. As few or as many Containers needed for each Microservice can be dynamically created or destroyed based on load, so automation is crucial, as the rapid creation of Containers enables the scaling and high availability of Microservices. Containers are isolated workload environments in a virtualized operating system. They speed up workload processes and application delivery because they can be spun up quickly.

II. LITERATURE SURVEY

[2016] ALAN SILL:-The Design and Architecture of Microservices

The basic approach of separating services info functions that can interact via programming interfaces has been with us for same time.

[2017] PAOLO DI FRANCESCO:- Architecting Microservices

This paper reports on a PhD research project addressing three different challenges concerning MSA: (*i*) the identification of the key properties of microservice architectures, (*ii*) the identification and investigation on a description language for designing and analyzing architectures, (*iii*) the identification of the factors that impact the process of migrating existing applications towards MSA.

III. METHODOLOGY

1] MICROSERVICES:

Microservices are a software development technique a variant of the service oriented architecture (SOA) architectural style that structures an application as a collection of loosely coupled services. In a microservices architecture, services are fine grained and the protocols are lightweight.

The benefit of decomposing an application into different smaller services is that it improves modularity and makes the application easier to understand, develop, and test.[1]

- Services in a microservice architecture are independently deployable.
- The services are easy to replace.
- Services are organized around capabilities, e.g., user interface front-end, recommendation, logistics, billing, etc.
- Services can be implemented using different programming languages, databases, hardware and software environment, depending on what fits best
- Services are small in size, messaging enabled, bounded by contexts, autonomously developed, independently deployable, decentralized and built and released with automated processes.
- The services are small - fine-grained to perform single functionality.

Microservices is a specialization of an implementation approach for service-oriented architectures (SOA) used to build flexible, independently deployable software systems.

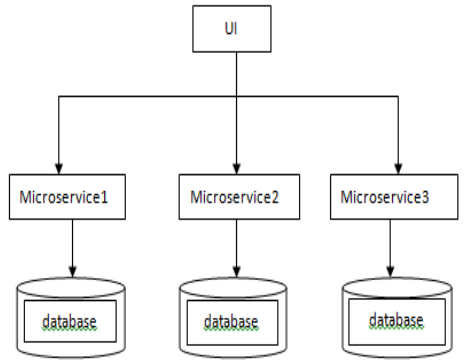

Fig 1:Microservice Architecture

2] NGINX WEB SERVER:

Nginx is a web server which can also be used as a reverse proxy, load balancer. Nginx's modular event-driven architecture can provide more predictable performance under high loads. Nginx default configuration file is nginx.conf.

The goal behind NGINX was to create the fastest web server around, and maintaining that excellence is still a central goal of the project.

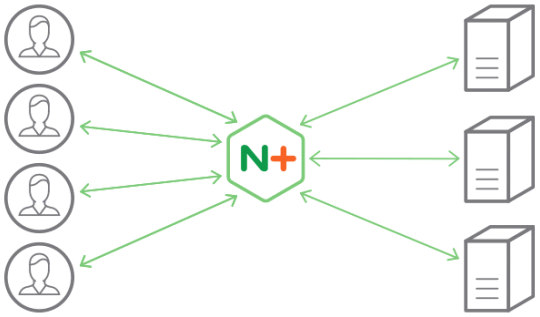

Fig 2:NGINX Server

3] LOAD BALANCER:

Modern high-traffic websites must serve hundreds of thousands, if not millions, of concurrent requests from users or clients and return the correct text, images, video, or application data, all in a fast and reliable manner. To cost-effectively scale to meet these high volumes, modern computing best practice generally requires adding more servers.

A load balancer acts as the "traffic cop" sitting in front of your servers and routing client requests across all servers capable of fulfilling those requests in a manner that maximizes speed and capacity utilization and ensures that no one server is overworked, which could degrade performance. If a single server goes down, the load balancer redirects traffic to the remaining online servers. When a new server is added to the server group, the load balancer automatically starts to send requests to it.

3.1] Load Balancing Algorithms

Different load balancing algorithms provide different benefits; the choice of load balancing method depends on your needs:[8]

- **Round Robin** – Requests are distributed across the group of servers sequentially.
- **Least Connections** – A new request is sent to the server with the fewest current connections to clients. The relative computing capacity of each server is factored into determining which one has the least connections.
- **IP Hash** – The IP address of the client is used to determine which server receives the request.

3.2] Load balancer performs the following functions:

- Distributes client requests or network load efficiently across multiple servers
- Ensures high availability and reliability by sending requests only to servers that are online
- Provides the flexibility to add or subtract servers as demand dictates

Load balancing refers to efficiently distributing incoming network traffic across a group of backend servers, also known as a server farm or server pool.

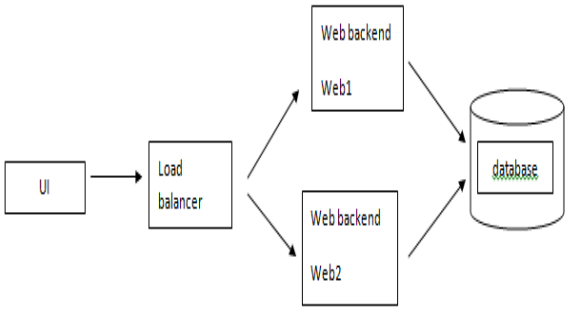

Fig 3: Load Balancer

4] CONTAINER:

Containers are easily packaged, lightweight and designed to run anywhere. Multiple containers can be deployed in a single VM. A container need not be used for a microservice. Containers hold the components necessary to run desired software. These components include files, environment variables, dependencies and libraries.

4.1] Application containers and system containers

Application containers, such as Docker, encapsulate the files, dependencies and libraries of an application to run on an OS. Application containers enable the user to create and run a separate container for multiple independent applications or multiple services constitute a single application. For example, an application container would be well-suited for a microservices application, where each service that makes up the application runs independently from one another.

4.2] Benefits of containers

Containers have better portability than other application hosting technologies: They can move among any systems that share the host OS type, without requiring code changes.

4.3] Container uses

Containers are well-adapted to work with microservices, as each service that makes up the application is packaged in an independently scalable container. Using containers may simplify the creation of highly distributed systems by allowing multiple

applications, worker tasks and other processes to run autonomously on a single physical machine or across multiple virtual machines.

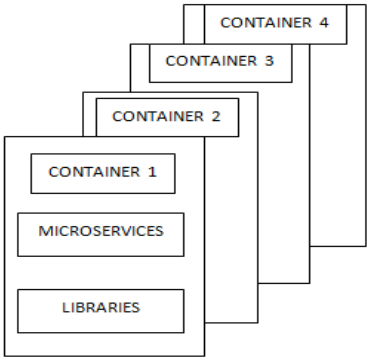

Fig 4:Containers

5] DOCKER:

Docker was released as open source in March 2013. On March 13, 2014, with the release of version 0.9, Docker dropped LXC as the default execution environment and replaced it with its own lib container library written in the Go programming language. Docker is developed primarily for Linux, where it uses the resource isolation features of the Linux kernel such as c groups and kernel namespaces, and a union-capable file system such as Overlay FS and others to allow independent "containers" to run within a single Linux instance, avoiding the overhead of starting and maintaining virtual machines.

Docker is an open source application container platform designed for Linux and, more recently, Windows, Apple and mainframe OSes. Docker utilizes resource isolation features, such as cgroups and Linux kernels, to create isolated containers.

5.1] Operation:

Docker implements a high-level API to provide lightweight containers that run processes in isolation. Because Docker containers are lightweight, a single server or virtual machine can run several containers simultaneously.

IV. ALGORITHMS

1] Round Robin Algorithm:

Round-robin algorithm is pre-emptive algorithm. The name of the algorithm comes from the round-robin principle known from other fields.A round-robin scheduler generally employs time-sharing, giving each job a time slot or quantum, and interrupting the job if it is not completed by then. The job is resumed next time a time slot is assigned to that process. If the process terminates or changes its state to waiting during its attributed time quantum, the scheduler selects the first process in the ready queue to execute.

2] Hash function algorithm:

A cryptographic hash function is a special class of hash function that has certain properties which make it suitable for use in cryptography. It is mathematical algorithm that maps data of arbitrary size to a bit string of a fixed size (hash) and is designed to be a one-way function, that is, a function which is infeasible to invert.

IV. PROPOSED SYSTEM

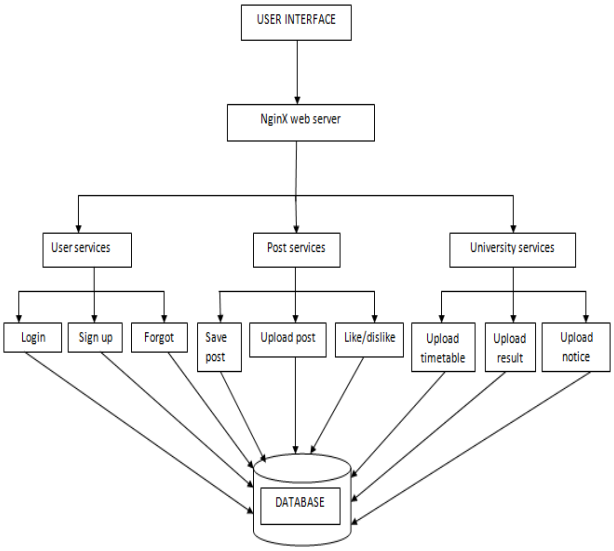

Fig 5: System Architecture

V. CONCLUSION

The system is developed with modular approach and also based on advanced technologies like Microservices, load balancer, containers. Because of using these technologies, system will reduce time consumption and also system provides high security for that one way encryption (hash function) is used.

REFERENCES

1. https://microservices.io/

2. https://en.wikipedia.org/wiki/Microservices

3. ALAN SILL Texas Tech University, alan.sill@ standards-now.org https://www.linkedin.com/ in/alan-sill-ttu

4. Paolo Di Francesco, Patricia Lago, Ivano Malavolta Gran Sasso Science Institute, L'Aquila, Italy - paolo.difrancesco@gssi.it

5. N. Alshuqayran, N. Ali, and R. Evans. A Systematic Mapping Study in Microservice Architecture. In Proc.of the 9th International Conference on Service-Oriented Computing and Applications. IEEE, IEEE, 2016.

6. M. Fowler and J. Lewis. Microservices a definition of this new architectural term. http://martinfowler.com/articles/microservices.html

7. P. Di Francesco, P. Lago, and I. Malavolta. Research on architecting microservices: Trends, focus, and potential for industrial adoption. IEEE International Conference on Software Architecture (ICSA), 2017.

8. https://en.wikipedia.org/wiki/Load_balancing

9.https://www.computerweekly.com/feature/What-are-containers-and-microservices

10. https://github.com/algorithm-ninja/cms-docker

11. https://www.researchgate.net/publication/323960272

12. Davide Taibi and Valentina Lenarduzzi and Claus Pahl Tampere University of Technology, Finland Free University of Bozen-Bolzano, Bolzano, Italy

13.https://www.studytonight.com/operatingsystem/round-robin-scheduling