**Title** – **Real Time Chat Application**

**About application –**

According to the problem statement in the title, the task is to create and deploy real-time responsive web-based, mobile, or other device chat applications.

**Why does this application require cloud?**

* Real-time chat application can offer consumers a dependable and scalable real-time communication solution by utilizing cloud infrastructure.
* Scalability: Cloud architecture enables real-time chat apps to scale up or down depending on the user base and usage patterns, making sure that the programme is responsive even during times of high traffic.
* Flexibility: Depending on the unique needs of the chat application, cloud architecture allows users to combine the best possible amount of storage, compute, and network resources.

**Literature Review –**

* The author [1] describes the use of AWS Lambda for building a serverless Chat Application. It allows for scaling without adding more servers. The relationship between various AWS services like S3, DynamoDB, CloudWatch, etc., and serverless technologies
* The author [2] the usage of encryption, access control, and other security measures to secure sensitive information are just a few of the problems that real-time chat applications face when it comes to security.
* The author [3] Integration with Other Apps: Research demonstrates the advantages of combining real-time chat applications running on cloud infrastructure with other cloud-based applications, such as storage and analytics services, to offer a more complete solution.
* The author [4] the use of failover strategies to maintain sustained operation in the case of hardware failures or network outages is discussed in papers as an advantage of leveraging cloud infrastructure to assure high availability and dependability of real-time chat apps.
* The author [5] papers examine how real-time chat apps can handle a range of traffic levels thanks to cloud infrastructure's capacity to scale up or down dependent on the number of users and their usage habits.

**Methodology –**

**Block diagram**

Server

Server sending client1 message into chat

Client1 sending message in chat and that message going to server

Other Clients connected

Client1

(Message Sender)

**Fig 1:** Server – client block diagram