We are living in a technological world where; we attach to the internet and devices associated to it. Nowadays we use devices and the internet not only to make transactions, send, receive data, but to confess our feelings. Chat via networks, mainly via internet has become a vital part with evolution of the technology. The above example shows how technology made things easy, in addition technology has brought everything to our fingertips. Project for this module also based on internet, which will illustrate the impact of a secure chat application to our day-to-day life. Chat applications such as Whatsapp, Messenger, telegram is popular among smartphone users but most of them are not reliable and secure to use. This paper is all about a new secure live chat application which I created using MERN. All new chat application allows people to securely chat individually and publicly using group chat option. Majority of the chat applications that are available in the general market are free and have unique features but failed to serve with security features to the customers

Security is of the biggest necessity while developing any kind of program, especially a live chat application. Live chat applications are particularly vulnerable to security threats since they include the transmission of sensitive information, such as financial and personal details as well as other significant data. It is essential to implement the proper security measures in order to protect the application and its users from any security breaches.

This paper will discuss the best methods for securing a live chat application built using the MERN stack.

Validation of Input: Another crucial step in securing a live chat application is input validation. User data is cleaned up before being handled by the program thanks to input validation. By doing this, attacks like SQL injection, cross-site scripting, and other frequent web application assaults are less likely to occur.

Secure Data Storage: The first step in securing a live chat service is making sure that all user data is stored securely. Access control and SSL/TLS encryption should be implemented for the MongoDB authentication process.

Access Control: Depending on the user's position or privileges, access control limits access to particular application features or resources. By doing this, sensitive data is protected from unauthorized access.

User authentication is the procedure used to confirm a user's identity when they access an application. Username and password, two-factor authentication, and social login are some of the methods that can be used to accomplish this.

Security for Real-time Communication: To guard against eavesdropping and man-in-the-middle attacks, real-time communication between users must be encrypted with SSL/TLS and authenticated using authentication methods.

Updates: Updates are necessary to preserve security and must be done frequently. For the purpose of preventing vulnerabilities from being exploited, updates should also include security patches and bug fixes.

Verifying the identification of users who access the program is done through the process of user authentication. Utilizing methods like username and password, two-factor authentication, or social login, this can be done.

Monitoring and logging: Monitoring and logging assist in quickly identifying and responding to security breaches. The application must record all pertinent events, and the logs must be routinely checked for irregular behavior.