Realization and deployment

Realization

1. **Technology Stack Frontend**: We used Vue.js to build the user interface.

- **Front-end**: We used Vue.js to build the user interface. Vue.js is a very popular JavaScript framework to build responsive user interfaces quickly. We implemented real-time communication via WebSocket so that users can instantly see messages from other people.
- **Backend**: The backend uses Java and Spring Boot frameworks to handle WebSocket connections and business logic.Spring Boot is a lightweight framework for rapid application development and deployment.
- **Database**: We chose MySQL to store user messages and likes data, connecting to the Java application via JDBC.MySQL is an efficient and reliable relational database that can handle large-scale data storage requirements well.

2. **User ID generation**

- To ensure the uniqueness and privacy of each user, we use UUID (Universally Unique Identifier).UUID generates globally unique identifiers, which are ideal for identifying different users.

3. **Anti-Injection Security**

- In terms of user input, we have taken anti-injection security measures. We have escaped user input to prevent SQL injection and XSS attacks. For example, when sending a message, we perform HTML escaping on the message content to prevent malicious script execution.

4. **WebSocket**

- WebSocket technology enables instant messaging in our chat rooms. webSocket provides the ability to communicate in full duplex between the client and the server, ensuring that every message from the user is delivered in real time.
- On the Java backend, we manage the connection and message handling through Spring Boot's WebSocket support.

Deployment

1. **Kubernetes configuration**

- We use several YAML files to configure and manage Kubernetes resources, including services, deployments, and persistent storage volumes. For example, the `bcdep.yaml` file is used to deploy the back-end chat service, and the `bcsvc.yaml` file is used to define the ClusterIP service, which ensures that the back-end service can only be accessed from within the cluster.

- The front-end is configured via the `fsvc.yaml` and `fdep.yaml` files, and the LoadBalancer service is used to expose the front-end application to external access.

2. **Database management**

- We use `mpv.yaml` and `mpvc.yaml` files to define persistent storage volumes and storage volume declarations to ensure persistent storage of MySQL data. Database configuration and credentials are managed through ConfigMap and Secret to ensure security and configuration flexibility.