**Chat:**

1. Utente crea un gruppo
2. N Utenti possono collegarsi a quel gruppo
3. Tutte le comunicazioni inviate dagli utenti su quel gruppo vengono ricevute sul browser di tutti gli utenti collegati.

**Soluzione WebSocket:**

1. Js Standard HTML5 + JSR356 (Tomcat 7)
2. SockJs Client e Server (java)

**UC:  
  
/app = /service  
/topic = /channels**

1. **StompClientDemo**.js -> Controller.metodo
   1. Connect/Disconnect
   2. Subscribe (Existing Topic) -> broker?
   3. Unsubscribe (Existing Topic) -> broker?
   4. Send -> Controller.metodo
      1. TestData (console.log)
   5. Receive
      1. Tabella
2. **SpringWsStompServer**
   1. Controller (/app/)
      1. receiveMessage (Message) -> add to Map
      2. forwardMessage(Message) -> forward to brokerChannel
      3. route (inoltra su destinazione dinamica)
      4. processMessage (Message) -> Output
   2. DataProducer (/broker/out)
      1. Thread produce messaggi e li inoltra sul outChannel

Generazione dati server side: (WebSocket session.send)  
<https://dzone.com/articles/sample-java-web-socket-client>

Descrizione con immagini:  
<http://javasampleapproach.com/spring-framework/spring-websocket/create-spring-websocket-application-springboot-sockjs-stomp>