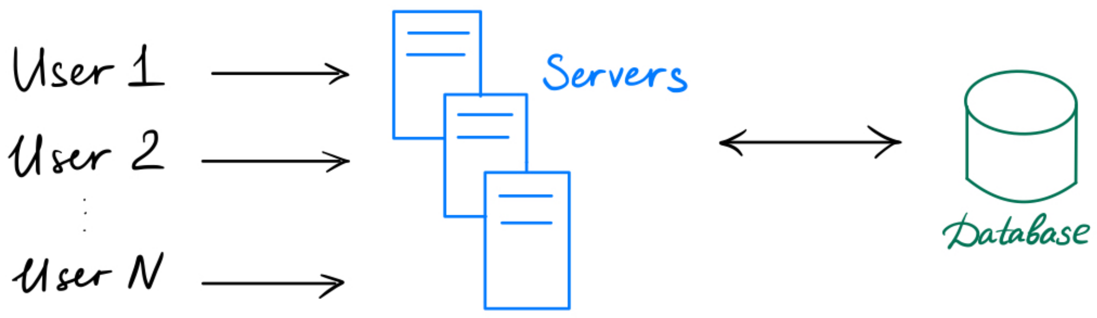
Simple messenger design

Proposed architecture is a client-server architecture where clients (users) communicate through multiple servers (servers are needed between the users instead of direct connection because User 1 is unaware of the IP address of user 2). For instance, if User 1 wants to send a message to User 2, it will connect to the server, that server will store the message in the database and switch it to User 2. The database stores all the necessary info for that action.



The basic element is one-to-one chat which I described above. It has some more detailed features like *last seen*. When User 1 is online it periodically sends request to the server and the server helps to store the info when it was online (timestamp, for instance). If User 2 enters the chat with User 1, through the server it can obtain info about the timestamp. Another element is *delivered/read flag*. When User 1 sends something to User 2, after going through the server and further to database and User 2, the acknowledgment is sent back to the server when the message is delivered to User 2. The server sends this info back to User 1. As soon as User 2 reads the message, another acknowledgment will be sent to the server.
This design can be extended by adding more complex elements like group chats (or channels in slack).