

Distributed Systems

Chat Manual

Zhong-Xi Lu & Thomas Van Bogaert

1 Used Technologies

- Java EE
- HTML, CSS (Bootstrap), JS, AJAX
- Servlets, Beans, Entity Classes
- GlassFish Server
- JavaDB

2 Set up

TODO

3 Architecture

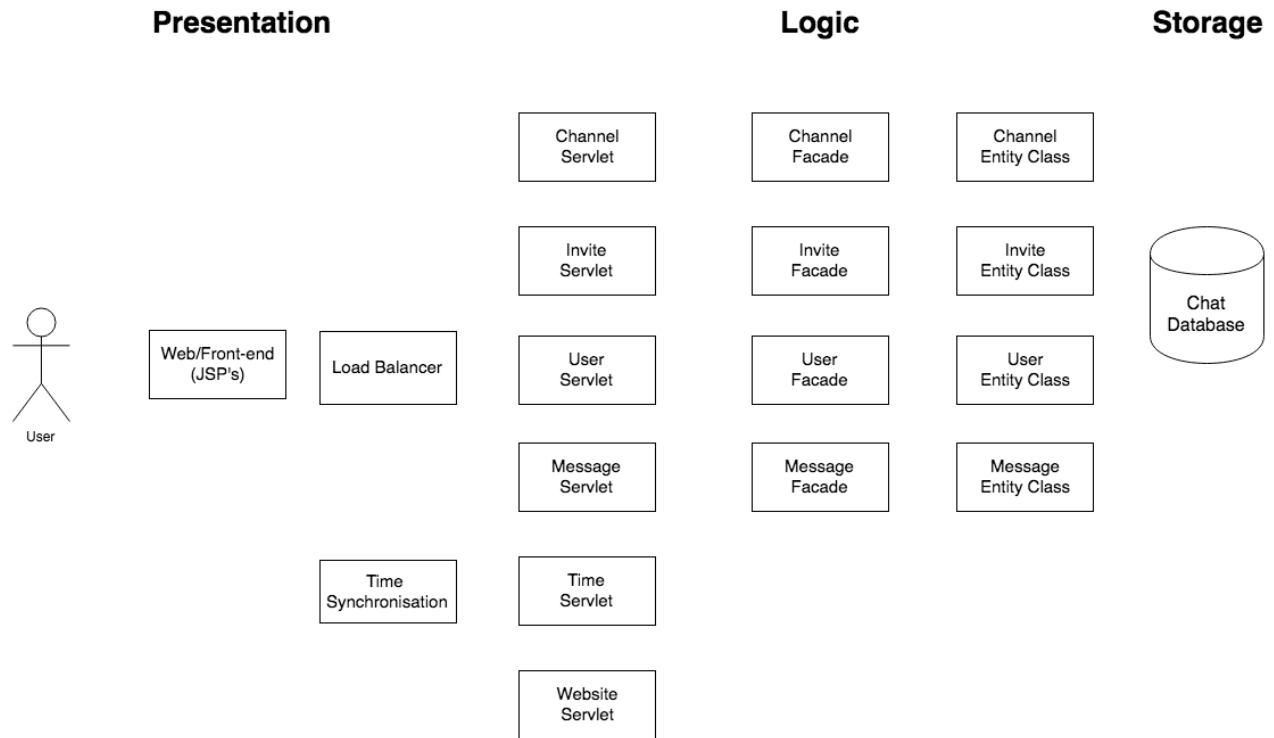


Figure 1: Chat Login Page

This is our service oriented architecture. In the front-end, we have jsp's that makes requests to the server through the load balancer. The load balancer forwards this request to the appropriate server/servlet. Each servlet then makes calls to the facades that is responsible for all the business logic. If a change to the database needs to be done, a facade can rely on an interface of the database (entity classes).


4 Design Choices

As previously mentioned, we tried to make a layered architecture (service-oriented). This way, each layer has one responsibility; servlets are responsible for handling requests, facades for doing all the logic, entity classes for the communication with the database. On the front-end, we have jsp's that can easily process all the data received from the server. Furthermore, we make ajax calls to the server, because it's reasonably fast and you can easily manipulate the responses and integrate it in your jsp's. Above that, those call are asynchronous, which makes it a lot easier to send different requests at roughly the same time.


5 Chat Tutorial

Chat!


Log In




Username



Password



Initial Time (HH:mm)



Drift Value (in sec/min)

Log In

Don't have an account? [Sign Up](#)

1

2

Figure 2: Chat Login Page

If a user browses to our chat application the page above will be shown.

1. A user can login by filling in the necessary information. Note that an initial time (HH:mm) and drift value must be specified.
2. Or, if the user doesn't have an account yet, he/she can create one here.

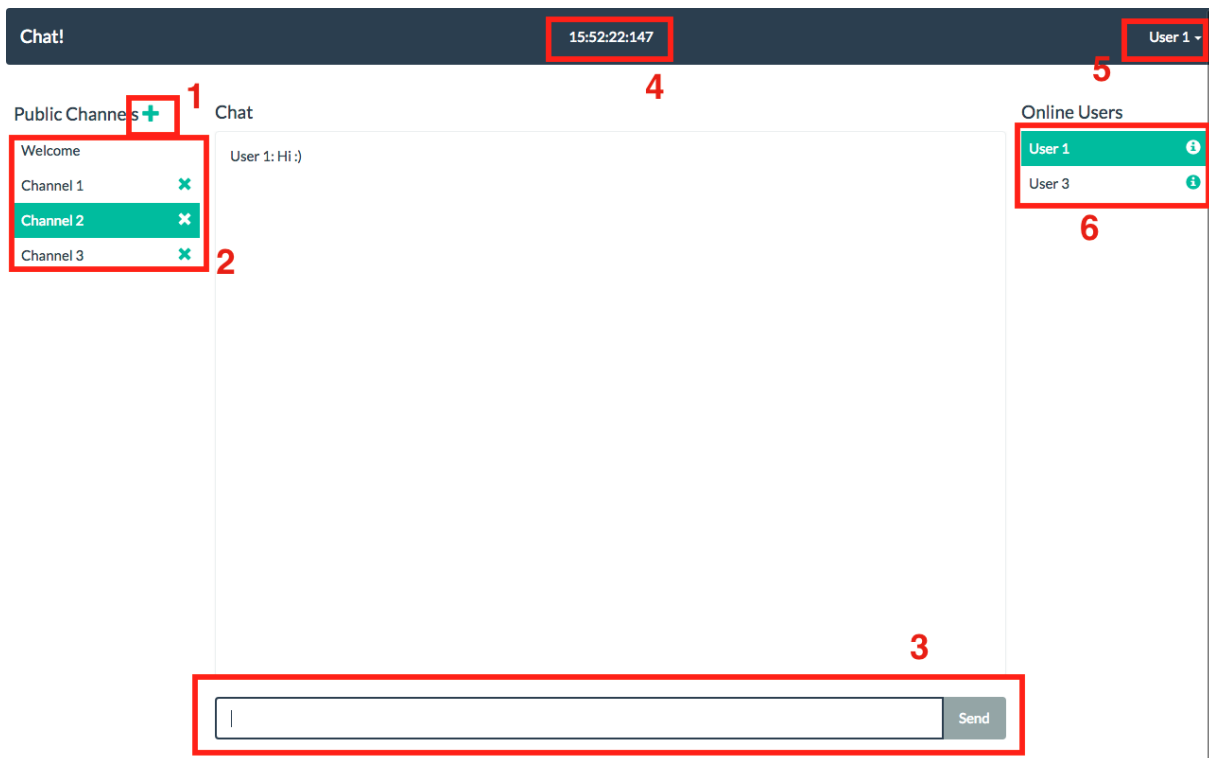


Figure 3: Chat Homepage

The figure above shows the main screen when the user has logged in or registered. The user can do the following:

1. If the user is a moderator, a new public channel can be added by clicking the plus-sign. After this, a window will pop up where a name for the channel must be specified.
2. Here, a list of all the active public channel are shown (green indicates the channel of the user); the user can simply join a channel by clicking on it. If the user is a moderator, the channel can be removed as well by clicking the X-sign next to the channel name.
3. This is the input field to send a message to the current channel of the user. The message will then appear in the chat box displayed above.
4. The current time for this specific user is shown here and is synchronised every 10 seconds with the server.
5. A user can log out by clicking on his username and selecting the "Log out" option.
6. All the current online users are shown here (green indicates the name of the user and red indicates a moderator). By clicking on a user, you can invite them to a private

channel (a channel name must be given as well), the other user can then accept or decline the invite. In addition, a moderator can also view all the messages a user sent (in all public/private channels) by clicking the info-sign next to the name. This will open a new page with the history of that user.