CONCEPTUAL ArchitecturE Document

Version 1.1

**Project Title:** **CHAT APPLICATION**

**Team Members:** Christopher Cargile

Jonathon Ng

Gayathri Parthasarathy

Nicklaus Rhodes

**VERSION HISTORY**

|  |  |
| --- | --- |
| **Version#** | **Revision Date** |
| 1.0 | 03/5/2014 |
| 1.1 | 04/19/2014 |

**Table of Contents Page No**

Architecturally Significant Requirements 3

Expected system qualities 3

Conceptual Architecture diagram 4

Client Component 5

Server Component 8

Database Component 9

MetaArchitecture Updates 9

CONCEPTUAL-ArchitecturE document

*Team Cupric seckel*

## ArchitecturalLY siGnificant requirements

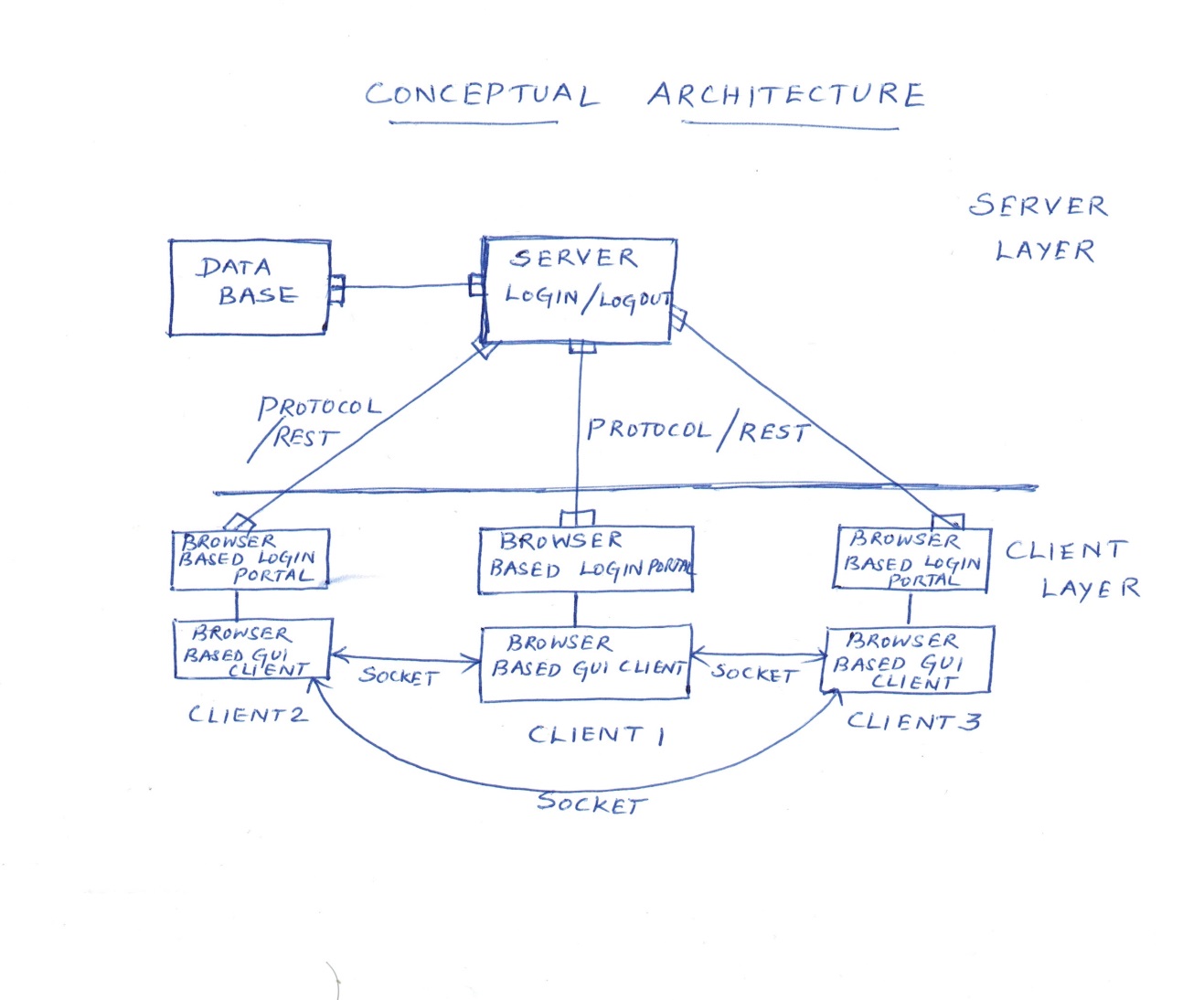
1. Chat application should be distributed and peer to peer application.
2. Server should provide the centralized control for the application.
3. Application should be browser based.
4. Clients should be able to send and receive texts.
5. Clients should be able to chat with all the online peers.
6. Server should notify all the online clients, whenever any client comes online or logs out.

**EXPECTED SYSTEM QUALITIES**

1. Chat Application should provide secure chatting environment.
2. Application should provide ease of use to the clients.
3. Application should provide privacy for the chat clients.

Conceptual Architecture of the chat application consists of three components – Client, Server and Database.

Figure 1 shows the conceptual architecture of the chat application with the components and their interactions.



**Figure1. Conceptual Architecture for Chat System**

**CLIENT COMPONENT**

**Responsibilities:**

* Allows client to initiate chat by logging in to the application through browser.
* Allows client to send and receive texts from online peers.
* Will maintain the current chat conversation.

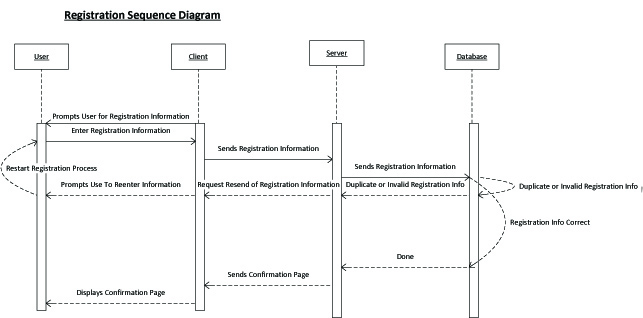
**Collaborators:**

* Server verifies the client with the credentials stored in the database.
* User list may be refreshed for the current user list from the server.

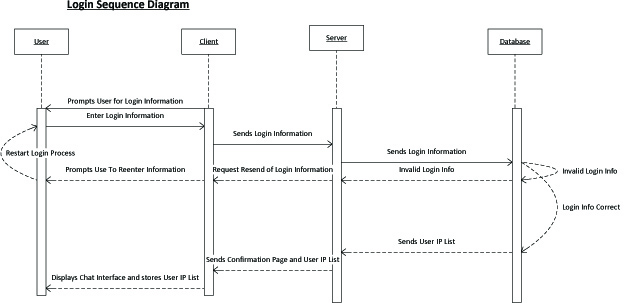
**Rationale:**

* Provides independence between chat clients and facilitates distributed system.

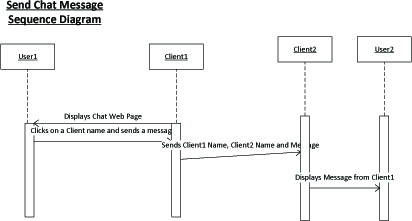
Figure 2 to 6 provides the various sequence of interaction that takes places in the chat application

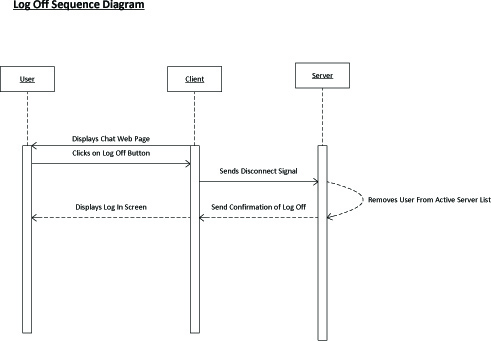
****

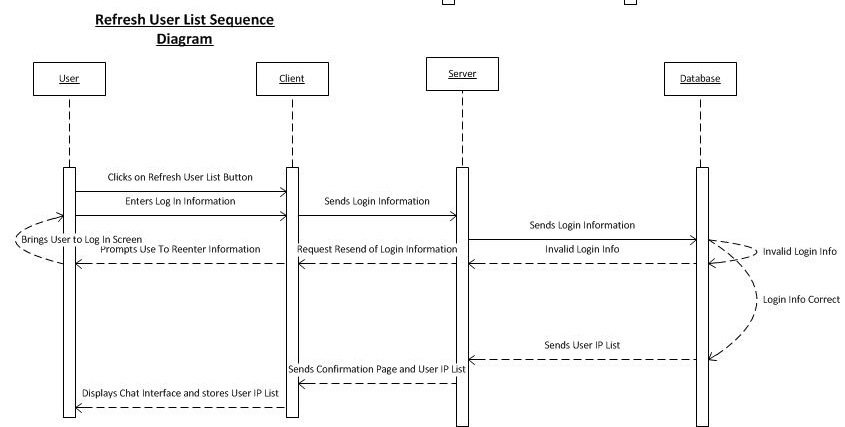
**Figure 2.Registration Sequence Diagram**

****

**Figure 3.Log in Sequence Diagram**

**Figure 4. Send Message Sequence Diagram**

**Figure 5. Log Off Sequence Diagram**

****

**Figure 6.Refresh User List Sequence Diagram**

Client Component includes Browser based login portal and Browser based GUI chat application. The typical sequence of events for usage is as follows:

1. User types in the server URL in the browser client and requests for the login page.
2. To log in, the User types into the browser client the username and password into the required fields and clicks the log in button as seen in Figure 3.
3. If new user, they have to register with the server to participate in the chat and log in with the credentials provided by the server as seen in Figure 2.
4. When the login credentials are verified, the client enters into the chat application as seen in Figure 3.
5. Application GUI shows chat history, List of users online, Text area to type messages and a submit button, Logout button.
6. Once a user wants to communicate with a peer (we shall label user2), the user will access the client, choose user2 from the user list, type a message into the text area box and click submit as shown in Figure 4. In order to gain connection to user2, the current user’s client establishes a socket with user2’s client for a peer to peer connection.
7. When the user wants to log out of the chat client, the user will click the logout button in the client. The server disables the connection and the socket connection between the peers is also disabled as shown in Figure 5.

**SERVER COMPONENT**

**Responsibilities:**

* Provides authentication of the clients.
* Allows centralized control of the chat application. Real world use of this model may use replication to replicate the server in case of failures.
* Notifies clients about all online users and clients going offline.

**Collaborators:**

* Interacts with the database, to store the user information.
* Verifies the client login username and password with the information in the database.
* Client submits login and logout requests to the server.

**Rationale:**

* Provides centralized access to the application and control of the application
* There may be single point failure, but in future enhancements it will be taken care of by considering measures like server replication during peak load.

**DATABASE COMPONENT**

**Responsibilities:**

* Provides data storage - user information

**Collaborators:**

* Server access the database through JDBC.

**Rationale:**

* Provides centralized data control and storage