Functional Requirements Specification

for

Hermes Chat Application

Authored by: Martin Ting
Partner: Kevin Suh

June 08, 2014

Version 1.0

1.0 **Overview**

This document contains the specifications for Hermes. There is information about behavior specification, implementation requirements, and architecture details.

2.0 **Architecture & Administration**

This section contains the details for choice of technology for our software stack as well as the interactions between them and interactions of team members.

2.1 **Java**

Java was chosen for the programming language of choice because database interface tools are easily available. The memory management features of Java was another deciding factor for Java.

2.2 **PostgreSQL**

Postgres was chosen for the data storage. It is a previously used technology within our context and have confidence in it's ability to perform. Our database engineers will also have a more extensive knowledge of optimization.

3.3 **Team Roles**

We used a side by side approach for this project. Martin had responsibilities to improve GUI but we decided to scrap that due to time and minor implementation constraints. We probably would have chosen to continue to complete the GUI if the deadline followed the specification in the submission guideline.

3.0 **Implementation Requirements**

These are the high level descriptions of details about implemented features.

3.1 **User Capabilities**

FT001: User account registration -- Non-existing users will be able to create new accounts.
FT002: User authentication -- Existing users will be able to login to the system until logout.
FT003: Account deletion -- A user can delete his own account. See behaviour specification for more details about account deletion.

3.2 **Social Services**

FT001: Contact/Block Lists -- Contact lists and block lists are maintained for each user.
FT002: Notifications -- Notifications of new messages will be sent to recipients.
FT003: Chats -- Users can start new chats with friends or view existing chats.

3.1 **Chat Manager**

FT001: Chat viewer -- Shows message log of the chat. See behaviour specification for more information.
FT002: Message attachments -- Users will be able to attach media to messages.

4.0 **Behaviour specification**

This section lists the specifications and requirements that were considered when architecting and designing Hermes.

4.1 **Account Deletion**

All users are allowed to delete their accounts. However, there are a few assumptions made when deleting the account. It must be true that all messages by the user are deleted and all the chats where the user is an initial sender must also be cleared. We give the user an option to list chats they are apart of and delete themselves from the chat. This will delete all their publications in that chat. Users must delete themselves from all their chats before they can delete their own account.

4.2 **List administration**

Contact Lists are maintained for all users. Only the user who owns the certain list would be able to add and delete contacts and blocks.

4.3 **Chat moderation**

Only the user who created the chat is allowed to moderate the chat. They can invite anybody in the system to the chat. The program will not allow you to add blocked users to chat.The ability to moderate allows a user to add and delete members. Also, only chat moderators are allowed to delete whole chats. The result of deleting a whole chat is: it will delete all subsequent messages and attachments and it will not appear in any of those users chat lists.

4.4 **Chat Viewer**

Our chat viewer will only display 10 messages at a time. A user will enter ‘next’ after each iteration of listing chat history to get more messages. They are ordered in descending order.

4.5 **Chat types**

Users will be able to create two different types of chats. Single chats where only two users exchange messages back and forth or group chats where a group of users send messages to the entire group. It is possible for a chat moderator to add people into a chat. When users are added to a single chat, that single chat becomes a group chat and all previous messages become visible to the group.

4.6 **Message Self Destruct**

The messages in the system are tagged with a self destruction timestamp. They will be removed from the system once the date-time is reached. The system sets the default destruction timestamp to one day.

4.7 Triggers and Stored Procedures

N/A

4.8 Performance Tuning

N/A

5.0 Implementation challenges/observations

There were many portions to this project that presented challenges. One of the biggest challenges was listing the chatlist by most recently edited. This involved gathering data from different sources and then sorting them for appropriate output. We were having difficulty figuring out why we had duplicated and other misinformation. Also, foreign key restrictions were a big problem for us. There was a lot of trial and error until we figured out the correct deletion protocols for getting rid of dependencies before correct deletion.

6.0 Documentation and misc details

At time of writing, the document specifications stated for a 1-2 page document. Considering the format/headings/whitespace we decided to extend this specification to three pages in order to provide a better amount of quality information for documentation.