Software Specification Document

Java Messaging Application

Prepared by

Arvindkumar Thiagarajan

Yilin Xu

Chandupatla Vinay Reddy

Chuhan Zhang

1. Introduction

1.1 Purpose

The purpose of this project is to create a Java-based, platform-independent communication service allowing people to share messages with each other and in groups effectively and securely.

1.2 Scope

The communication system is meant to ease communication amongst people by creating a convenient and easy-to-use application for users. The server will be based entirely on Java and the project will extend the current system. Jenkins will be employed for CI and GitHub for Version Control.

1.2.1 Milestones

- 02/22 - Sprint 1: Basic version of the application with broadcast and point-to-point messaging implemented, SRS, UML diagram
- 03/14 - Sprint 2: Database setup, design interfaces
- 03/28 - Sprint 3: Implement communication between users and groups and features.
- 04/11 - Sprint 4: Deliver final prototype.

We will strive to provide a comfortable and dynamic user experience with the best pricing available.

1.3 Definitions, acronyms, and abbreviations

The conventions used in this document are provided in the below table:

UML	Unified Modeling Language
CI / CD	Continuous Integration / Continuous Development
DB	Database(s)
SRS	Software Specification Document
IP	Internet Protocol

1.4 References

IEEE standard 830-1998 : http://www.math.uaa.alaska.edu/~afkjm/cs401/IEEE830.pdf

2. Specific Requirements

2.1 Functional requirements

2.1.1 User actions

A user must be able to :

- Possess an avatar/icon.
- Know other people in their circle
- Find other people or groups, assuming they want to be found.
- Invite other users to join the user group
- Invite other group to join the user group
- Go into do not disturb mode (The user can't be found by other user)

A user can:

- Be a part of one or more groups.
- Be a moderator of one or more groups. A moderator has the following abilities:
  - Can remove one or more members from the group they're moderating.
  - Can delete a group they're moderating.
  - Can accept invitations to add one member
  - Can accept invitation to add group

2.1.2. User Communication

The following communication features for a user will be supported by the application:

- A user can send one or more messages to another user.
- A user can send one or more messages to a group.
- A user can delete a message before or after it's read by recipient or group members.
- A user can follow other users or groups.

Additional user requirement:

- A user may find out where the message goes i.e. who or what group the message is sent to.

- A user could forward messages.

2.1.3 Group Communication Requirement

A group must:

- Have at least one moderator.
- Be able to add other groups to the current group.

Additional group communication requirement:

- Have at least one thread.
- need an optional password to open the channel.

2.1.4 Message Requirement

Message must:

- Have the option to reply only to the sender or to a subset of the group.
- Be stored forever on the server.
- Have a timestamp based on the sender’s local time.
- Any message in a group can be privately replied to.
- Messages need to have lifetime. Messages are exposed for a certain amount of time before disappearing. Free tier user message lifetime is limited. Paid tier users can keep the message history forever.
- Multimedia content such as tunes/recordings, and video, can be part of a message.
- Messages need support Emoticons input.
- Users should be able to recall/delete a message. Messages that haven't been read should be deleted or not delivered. If messages have been read, messages should disappear/deleted.
- Messages need to be delivered in order that they were sent.
- Messages need to be queued on the server if the receiver isn’t online. Users can time-shift and receive messages in order they were sent.
- Message will be encrypted for additional fee (Premium feature)

Additional message requirement:

- Message can be set to self-destruct in 5 seconds upon read by the receiver.
- Message can be translated into receiver preferred language.
- Similar yes/no responses from different users may be combined together with numbers that indicate how many users give this kind of response.
- Low value messages disappear from the user quickly or upon reading but will remain in the database.

- Message needs to support non-English language input.

Additional requirement about thread

A Thread must support the following features:

- Be viewable by the entire group or part of the group.
- A message replied to any other message in a thread belongs to the same thread.
- A message in a thread must reply to exactly one message of the thread.
- A Thread should be identified by title (first few words in first message in thread)

2.1.5 Government Requirement

A server must be able to:

- Provide the chat history of a specific user within a specific time period.
- Send a copy of a specific user message to an authorized government agency in a fixed length of time.
- Report login activities of a specific user to an authorized government agency.
- Provide the original messages from a specific user.

A server might :

- Wrap the message with an IP address.

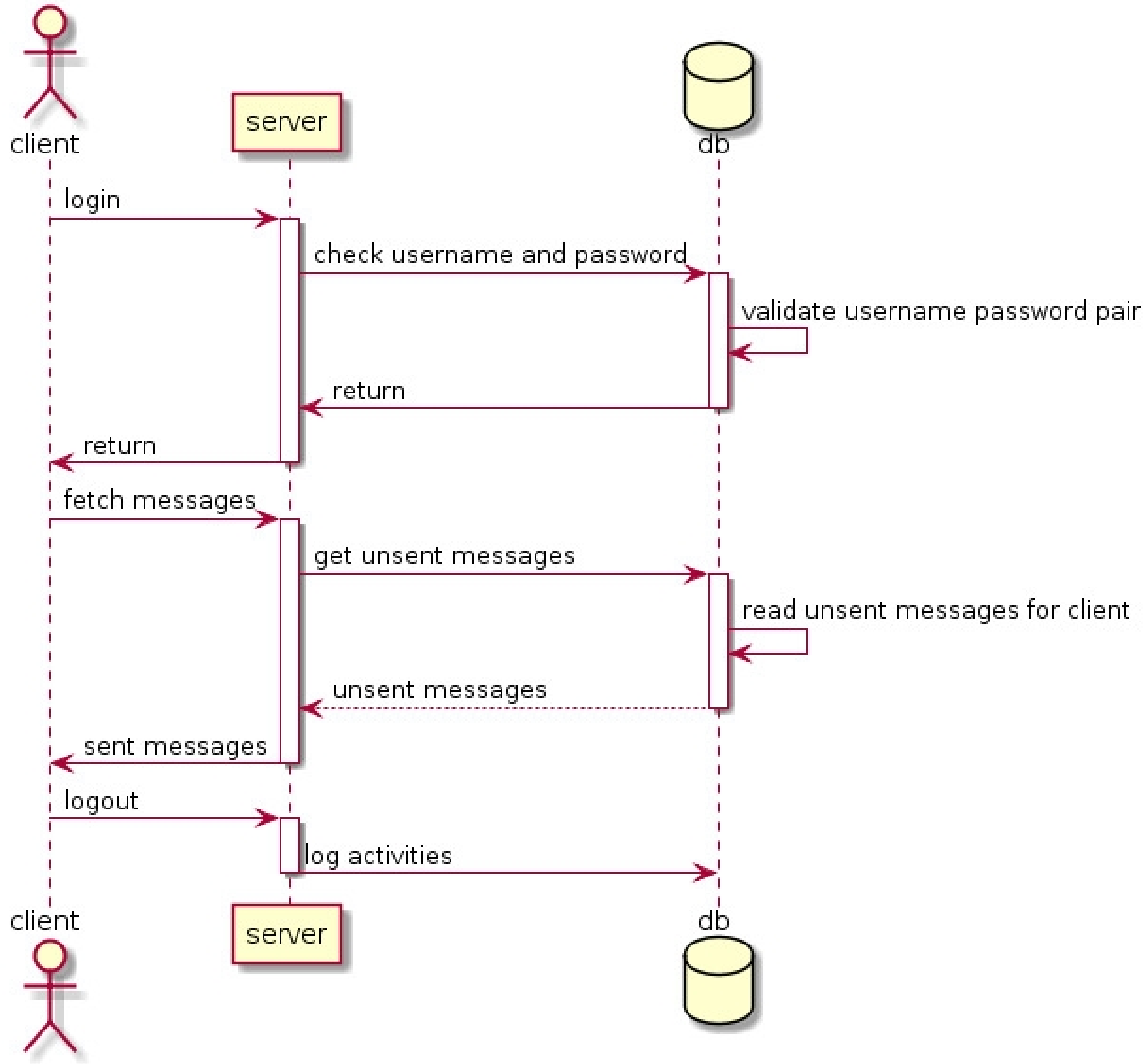
2.2 Non-Functional requirements

- A single text message should take no longer than a second to deliver to the receiver provided the receiver is online.
- Every message displayed must be suffixed by the time of day in which it was received at the server.
- Every user is associated with a status - green dot for online, grey dot for offline, and red dot for Do-not-disturb mode. In a group, a star next to the username of a user signifies that they are a moderator of that group.
- Every user and group name has a follow/unfollow button next to it to enable a user to follow them.
- A moderator can also see a delete button next to the group name to delete the group.
- A group must also have a ‘View members’ button next to its name to enable users to view members of the group. This will bring up a list of users in a vertical list format who are part of that group.

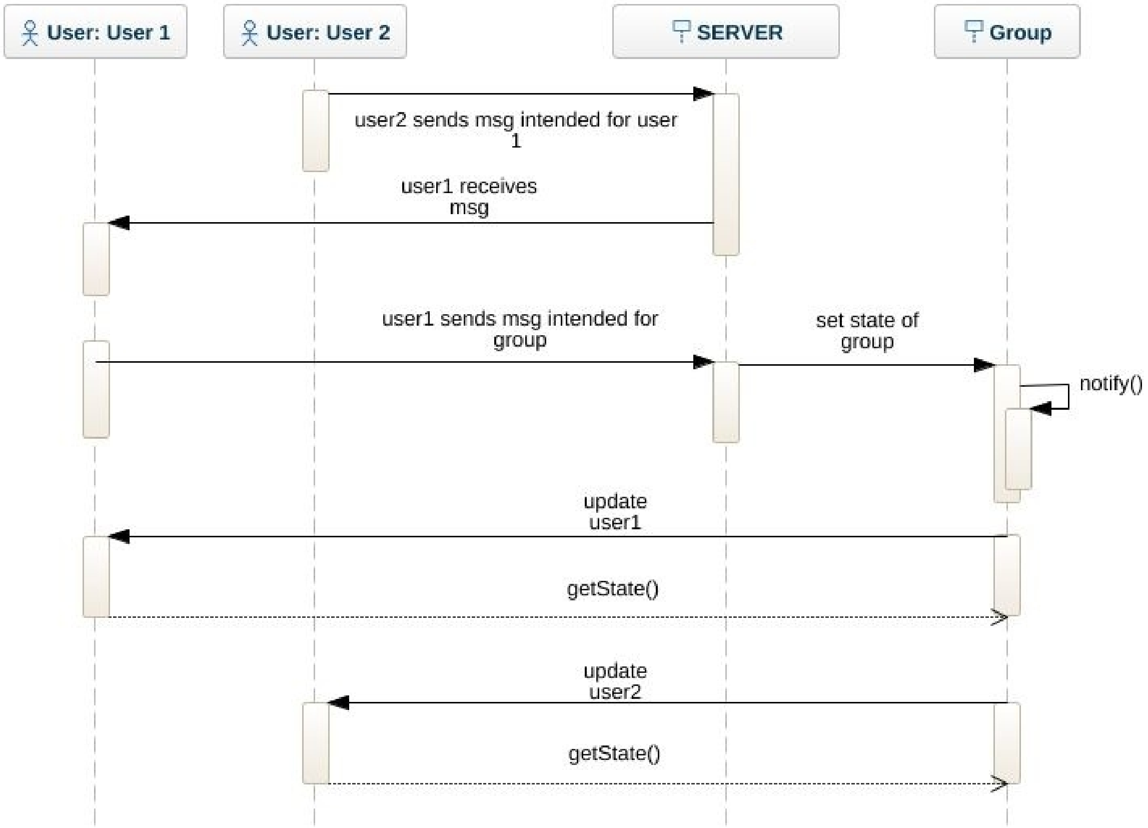
- A moderator can see a ‘Remove’ button next to each user in the above-mentioned list. This button enables a moderator to remove members of the group.
- Each message sent (either to a single user or to a group) must be associated with 2 buttons:
    - Forward button - to forward the message to another user or group.
    - Delete button - to delete the message
- For every message a user sends, he can view the places where it’s forwarded to by clicking on another button associated with the message.
- A user can see a ‘reply’ button on hovering over a message in order to reply individually to that message.
- On deleting a message, only the user who deleted the message can see a button in its place - A recall button, which will allow other users to view the message again if/when selected.

3. Sequence diagram

3.1 Authentication sequence diagram



3.2 Communication sequence diagram


User: User 1
User: User 2
SERVER
Group
user2 sends msg intended for user 1
user1 receives msg
user1 sends msg intended for group
set state of group
notify()
update user1
getState()
update user2
getState()