NAME: ANAM FREDRICK OTIENO

REG NO: CS282-0756/2011

CLIENT/SERVER CHAT SYSTEM DOCUMENTATION

FOURTH YEAR SEMESTER I

Introduction

This document provides a detailed description of the Chat system developed. It includes both the analysis and design descriptions for the system.

General system description

The system is a chat application system that allows users to chat with each other. It implements the client/server architecture model whereby the server application is accessed over the network using front-end client applications and only registered users are allowed access to the system. The system allows for both private chatting and broadcast/public chatting (i.e. group chatting).

Figure 1: Application Architecture

Concepts demonstrated

This project aims at demonstrating the following concepts:

- Networking
- Socket programming
- Client/server computing model

The system implementation is achieved using Java programming language. There is no persistent data storage for the system as this acts just as a demonstration of the concepts. However, user accounts are stored in an array list that only persists as long as the server is running.

SYSTEM ANALYSIS

System requirements

These are the statements of what the system must do or the characteristics it must have. The system requirements are divided into 2 categories:

- Functional: Entails what the system does
- Non-functional: The characteristics that the system must possess.

A. Functional requirements

The system accomplishes the following as per user expectations:

1. To enable the user to engage in a group chat after a successful login.
2. To provide new accounts' registration functionality for new users.

B. Non-functional requirements

The system exhibits the following characteristics:

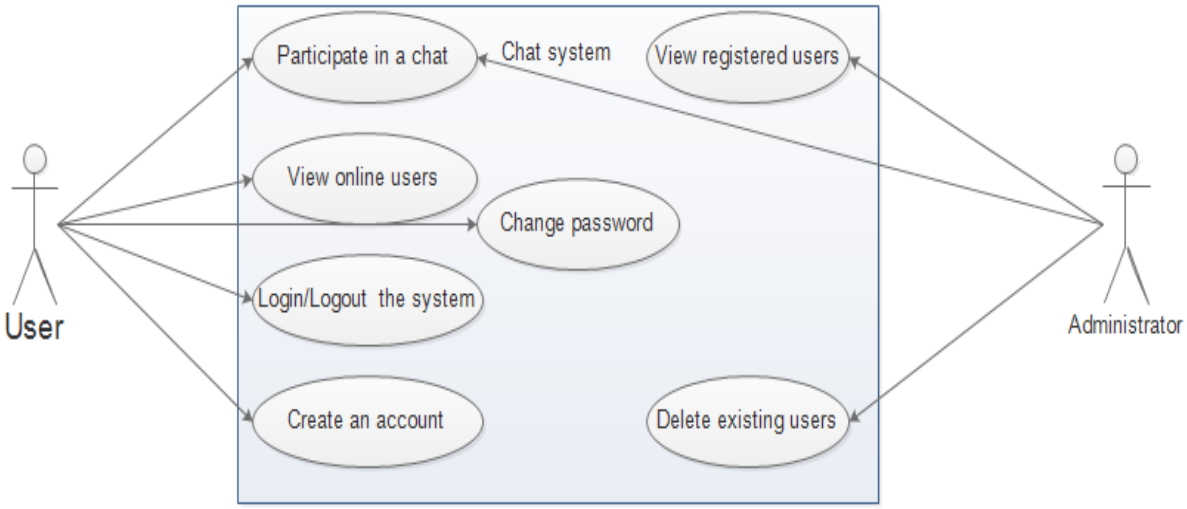
- Operational: The system operates in all operating system environments. Being a desktop application it cannot be accessed over the web. A network connection is mandatory for using the system.
- Performance: Response times for the system is less than 5 seconds and it's available for use continually (24 hours a day). Users are also informed for interruptions in case of system failures.
- Security: No two users can be registered with the same username into the system. Usernames must be unique to individuals, only the administrator is allowed to view all registered members and ability to delete registered members.

Use cases

The following use cases exist for the system:

i. Send a chat message
ii. View chat messages
iii. View online/available users
iv. Log in to the system
v. Logout of the system
vi. Register/create an account

Use Case diagram



SYSTEM DESIGN

This section explains how the proposed chat system will provide its functionality. It illustrates the various aspects of the system and involves the following among others:

- Architectural design
- Interface design
- Program design

To achieve these, object oriented design is employed for modeling the various system features.

A. *Architectural Design*

The chat system is a client/server system whereby the system is accessed over a network and it's implemented in 3-tier architecture: client application (installed at client machine) and a server application (managing the chats and hosted in an application server). (**Refer to figure 1 above**).

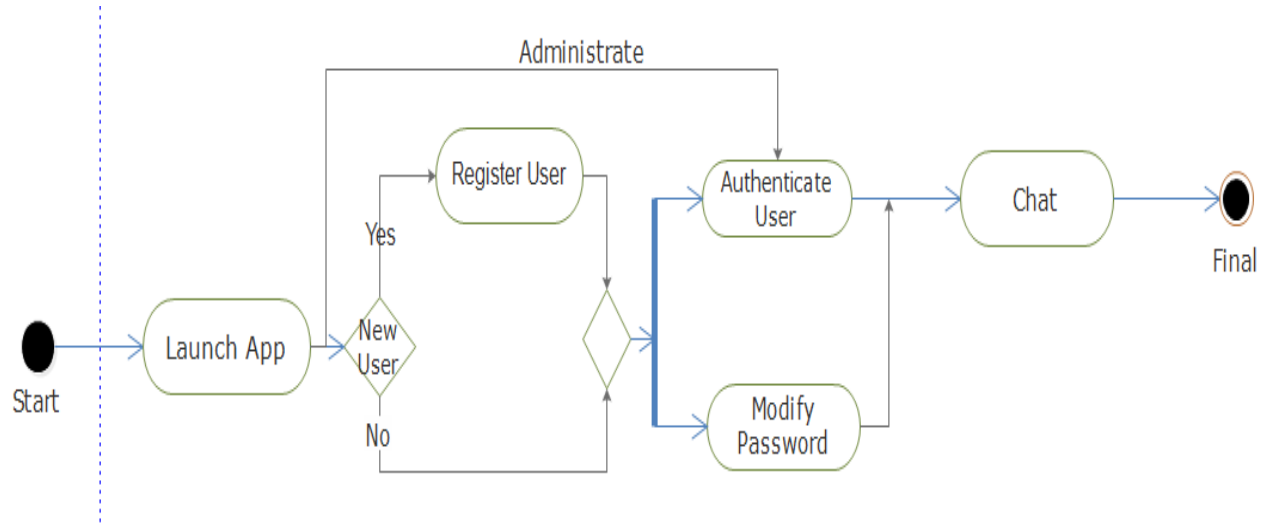
B. Program Design

To effectively achieve its objectives, the proposed system has been divided into the following modules for proper maintenance:

1. **Registration** component: This is to enable new users create their user accounts.
2. **Chat component:** To manage the sending and receiving of the chat messages.

3. **Authentication** component: For user login process before they use the system.
4. **Network** component: Establishes the connection for network communication and closes the connections (sockets) at the end of communication.

Chat System Activity Diagram



Program classes

Chat server

1. DbConnection: Manages user accounts information by storing them in an array list data structure.
2. ChatServerController: Launches and terminates the chat server end.
3. ChatServer: Listens for client connection requests and allows the establishment of connections
4. ChatServerHelper: Enables the server to handle simultaneous client connections as each handles individual client communication
5. ChatServerLogic: Contains the functions that perform the various chat tasks for an individual user e.g. authentication, sending and receiving chats.

Chat Client

1. ChatConnection: Establishes a dedicated network connection to the server for the sending and receiving of chat messages.
2. ChatDisplay: Handles the user interface of the system to allow users send and receive chats.
3. ChatClientLogic: Contains functions that process the sending and receiving of messages

4. ChatLauncher: Lauches the application (main entry in the application on client end).

The project structure

This class project consists of the system projects in separate folders:

i. ChatServer project
ii. ChatClient project

Each project has an executable jar file that can be used to run the application outside an IDE. The server executable must be run first. As for the development, the system uses localhost as the server host and it is listening on port 1234.
The system is compatible with java runtime environment 1.7 or higher.
The default user account is;
Username: admin
Password: admin