International Journal of Engineering Research-Online
A Peer Reviewed International Journal
Articles available online http://www.ijoer.in

Vol.3., Issue.3, 2015

REVIEW ARTICLE

ISSN: 2321-7758

DESIGN AND IMPLEMENTATION OF ROBUST INSTANT MESSENGER FOR SECURED REAL-TIME COMMUNICATION

SUDESH V KAMAT[1], SUNEEL C SHINDE[2]

[1]MCA Student, Department of MCA, KLEDRMSSCET, Belgaum, India

[2]Assistant Professor,Department of MCA, KLEDRMSSCET, Belgaum, India

Article Received:17/05/2015 | Article Revised on:28/05/2015 | Article Accepted on:08/06/2015

SUDESH V KAMAT

SUNEEL C SHINDE

ABSTRACT

In this paper, We introduce a robust instant messenger that is designed for efficient and real time communication. It is based on federation structure and uses client-server and server-server APIs for communication.It defines APIs for synchronising extensible JSON objects known as events between compatible clients, servers and services. The application has homeservers, every client has a home server and Each homeserver stores the communication history and account information for all of its clients, and shares data with other homeservers and their clients. The data exchanged are expressed as events and if the transmission is successful the server returns event-id. There are many options provided for efficient communication, these include creating new room, joining existing room, banning a room member and leaving a room. This application is platform independent that is it can work on various operating systems.

Keywords—Chat, IM, Communication, Computer application

I. INTRODUCTION

Instant messenger is an IM application that allows communication in a secured manner. A secured private IM is needed to ensure data transmission between sender and recipient more secure and our goal is to make real-time communication over IP without any interruption and should be able to communicate and exchange data with another system or device by providing a new open standard that allows communication services to interoperate.

There are many methods available that allows real time chatting over Internet. The purpose of this robust methodis to develop a chat application that will allow any user having Internet connection to have personal or even public communication with any other user. The implementation of this paper is centred on development of a user friendly application that will allow the users to easily login and communicate with each other.

Some of the principles that this method follows are:

- Web friendly APIs.
- Provides a simple architecture.
- Empowering the end user that is, user will be able to choose the server and he will be able to keep their communication private.
- Uses access tokens and event ids for authentication.
- Some of the functionality of this application includes:

- Creating and managing chat rooms with no point of failure.
- Sending and receiving messages in a room with encryption.
- Room management by inviting, joining rooms, leaving rooms, banning members.
- User management by Unique user id and password, User profile picture and display name

Architecture

This application has Federation architecture. Every client in this application is associated with a homeserver and it communicates with its home server using client-server API. The homeserver communicates with each other using server-server API. For example, if client A wants to send message to client B, client A performs HTTP PUT request on its homeserver using client-server API. A's homeserver then sends the message to B's homeserver by performing HTTP PUT request using the server-server API. B's homeserver authenticates the request. Client B then receives the message from its homeserver via GET request.

Users

All clients are associated with user accounts, which is identified using unique User-Id. This User-Id is namespaced to the home server which had allocated the account and is of the form: @userId:domain

Events

Data in this application is exchanged is expressed in the form of "event". Each client action (i,e sending a message) is an event. Each event has a type that is used to differentiate different types of data. For eg, m.room.message is the event type for message.

Room

Room is a place where users can communicate with each other by sending and receiving messages. Events are sent to the room and all members of the room will receive the event. Room Id's are case-sensitive and are not human readable. Each room has unique id called RoomId and is of the form:

!room_id:domain.

II. LITERATURE SURVEY

A. Existing system

In today's world there is a need of efficient communication between two or more people. There is need for fast and secure communication or for exchanging information. In existing system the communication is via text messages for exchanging data between two people and via conference calls for communication between many people. Some of the limitations of existing system are:

- Communication is not secure and efficient.
- Communication is costly.
- Maintaining communication history is difficult.
- No proper authentication of users.
- Managing users is not possible.

Proposed system

The proposed application is designed to be secure and efficient. It follows some standards for communication. It uses API's for communication and is based on federation architecture, which provides efficiency. In this application the communication takes place within chat rooms, and any user can create these rooms. Here the communication can be private or public. Private communication can take place only between two people and public communication can include any number of people in a public room. Some of the advantages of this system are :

- User management and room management is possible.
- Every user has unique user-id and password.
- Communication is not costly.
- Good interface with many options.
- Users can see a list of online users in a room.

B. Feasibility study

Organizational feasibility

There are various project teams working in an organisation on different projects. All the members of a project team can use this application to discuss various issues related to their project. Instead of calling a meeting for every small issue, they can make use of this application.

Economic feasibility

This application does not require any extra hardware apart from computer and Internet for communication. For example when there are issues that are to be discussed by a project team, the project manager has to call a meeting or to make a conference call to discuss the issues. So there is a

waste of time and money. This application does not involve any telephone calls, as a result it is cost effective and also lot of time is saved as there is no need for wasting time in deciding a schedule for meeting, inviting all members for meeting and so on. With this application the project manager can create a room for discussing the issue, which will save cost as well as time and provides an efficient and secure way of communication.

Technical feasibility

This application requires a computer and an internet connection, Apart from this it does not require any extra hardware or any high level software. This application can be upgraded by adding many features using the current technology.

C.Tools and Technology

Java programming language is a general purpose, class based, object oriented language. Java language is similar to C and C++ but is organised differently. Java language is strongly and statically typed, this helps to distinguish between compile time errors and run time errors. Compilation includes translation of programs into machine independent byte codes. Run time activities consist of loading and linking classes to execute the program. [2]

Java language is statically typed means every variable and expression has a type that is known at compile time. Types are divided into two categories primitive and reference. Primitive types are Boolean type and numeric types. Reference types are class, interface and array types. [2] [13]

Java programs are organised as sets of packages and each package has a unique name. The names for packages are hierarchical. Package members are class types and interface types. Package can be stored in file system and database. Package consists of number of compilation units. Package member are its sub packages and all top level class types and top level interface types. [2]

JavaFX[3]

JavaFX is a set of packages of graphics and media that allows developer to create, design, test, debug and deploy client applications. JavaFX written as java API, its code can ref APIs from any java library. The appearance of javaFX application can be customized. If you want to separate user interface and back end logic, then you can develop the presentation aspects of user interface in FXML scripting language and use java code for application logic.

JavaFX releases are fully intregrated with java SE 7 runtime environment and java development kit. Since the JDK is available for all major desktop platforms, javaFX applications complied to JDK 7 and later run on all major desktop platforms. Cross platform compatibility allows a consistant runtime experience for javaFX application developers and users. Some of the features are :

Java APIs: JavaFX is a java library that contains interfaces and classes written in native java code. Its APIs are designed to be alternative to JVM languages such as scala and ruby.

Scene builder and FXML: FXML is a XML based declarative markup language for constructing a JavaFX application UI. Scene builder can generate FXML markup that can be ported to an IDE where business logic can be added.

WebView: It is a web component that uses WebKitHTML technology that allows it to embed web page within a javafx app. JavaScript that is running in web view can call java APIs, and java APIs can call javascript that is running in Web view.

Swing interoperability: Swing applications can be easily updated with new JavaFX features, such as embedded Web content and rich graphics media playback.

Kotlin[4] [5]

Kotlin is a new language for JVM unvield by JetBrains in the year 2011. One of the goal of this language is to compile as fast as java. It is designed to be an industrial strength object oriented language and fully interoperable with java code allowing a gradual migration from Java to Kotlin.

It is a statically typed programming language that runs on JVM and can be compiled to javascript source code. It is developed by a team of JetBrain programmers based in St Petersburg. Some of the features of Kotlin include:

Concise: Reduces number of lines of code.

Versatile: Server side apps, android app, or front end code running in the browser are built.

Safe: Avoid errors, such as null pointer exceptions.

Interoperable: Works well with existing framework and libraries of JVM with java interoperability.

CSS [6]

CSS is an acronym for cascading style sheets. It defines layout of docs eg :colors, margins,

height, width and background images etc. HTML can be used to add layouts but CSS is more efficient. CSS is used by most websites to create visually engaging web pages, UI for web apps and UI for many mobile apps.

CSS is design to allow separation of documents content from its presentation along with elements such as layouts, colors and fonts. This separation allows to present the same mark up page in diff style for different methods. It is also used to display webpage in a different way. It has a simple syntax and uses English words to specify names of style properties.

Some of the advantages of CSS are :

They control the layout of document from a single style sheet.

They have more precise control of layout.

They apply different layout to different media type.

It has many advanced and sophisticated technique.

IntelliJ IDEA [7] [8]

IntelliJ IDEA java integrated development environment (IDE) for developing computer software. It is developed by JetBrains and is available as an Apache2 Licensed community edition and in a proprietary commercial edition.

Some of the features are:

Enterprise frameworks: It offers framework specific coding assistance and productivity boosting features for Java EE, Spring, GWT, Grails, Play and other frameworks, along with deployment tools for most application servers.

Built in developer tools: It includes set of tools that support for Gradle, Maven, integration with git etc.

Web development : In addition to framework support it provides advanced coding assistance for HTML, CSS and JavaScript technologies.

III. SOFTWARE REQUIREMENT SPECIFICATION

A software requirement specification is a description of software that is being developed. It describes the purpose of the software and how it is expected to perform. SRS minimizes the development cost and the time and effort of the developers to achieve their goals. It defines how the application will interact with system hardware, users etc. It lays out functional and non functional requirements and include set of use cases. SRS also provides a base line for verification and validation of the software that is developed.

A. Users

Users include project manager, project team members and it may also include other employees who want to discuss their issues or get a solution for some problem or help others in the room.

Project Manager

A project manager may can create new room, mark it as public and ask its team members to join that room. The manager can then discuses on various topics with their team members, assign them some work, if they have some issues or doubt then it can be addressed.

Project team members

All the members of a project team can communicate with each other or with their manager by creating a room or by joining already created room. They can join some room discus their issues and problems and then leave that room after they are done with their work.

Other employees

Other employees of the company can communicate with each other by creating or joining the rooms. If an employee is an admin of the room he can also ban a member, if he is misbehaving.

B. Functional Requirements

Keep it simple

The user interface is simple and user friendly. Anyone with less technical knowledge can login, easily communicate with other users, can easily use options that are provided.

Room management

The user should be able to manage the room with options such as creating new rooms which can be public or private, joining rooms which are already present, banning room members and leaving rooms after the work is done.

User management

Users should have their unique id and password. Users should be able to see other online users, should be able to see other users who are typing, Should have display name and profile picture.

C. Non Functional Requirements

Safe and secure

Creation and management of fully distributed chat rooms with no single point of failure or error. Access tokens are used for authentication of the user. The server returns event ids if the

message is sent successfully. Every room has a room id in non-human readable form.

Performance

User should be able to login and logout easily without any delay. Communication should be fast without any error or delay. User should be able to create, join and leave rooms without any delay.

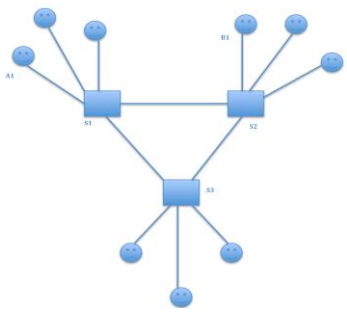
Platform independent

The software should be able to run efficiently and effectively on various platform, that is the software should be flexible and platform independent. Since this application is built using javaFX it can run on various operating system.

IV. SYSTEM DESIGN

A. System Perspective

The application is based on federation architecture and uses client server and server-server APIs. Each client has a homeserver and each homeserver may have one or more clients.When the user logs in using the id and password, the server returns the access token if the login is successful. Every time the user logs in, he receives a different access token from the server. Suppose a user wants to communicate with another person, then the user creates a room by entering a room name. This room name is sent to the server and the server returns the id known as roomid for this room, which is in non-human readable form. When the user enters a message, along with the message, the access token and roomid is sent to the server. If the message is sent successfully then an event id is returned by the server.



A1, B1 -> Clients S1, S2, S3 -> Home servers

Fig 1.1Architecture

The architecture in figure 4.1 shows how the application works. Here s1, s2, s3 are home servers. Every client in this application is associated with a homeserver and it communicates with its home server using client-server API. The homeserver communicates with each other using server-server API.

For example, if client A1 wants to send message to client B1, client A1 performs HTTP PUT request on its homeserver s1 using client-server API. A1's homeserver then sends the message to B1's homeserver s2 by performing HTTP PUT request using the server-server API. B1's homeserver authenticates the request. Client B1 then receives the message from its homeserver via GET request.

B. Context Diagram

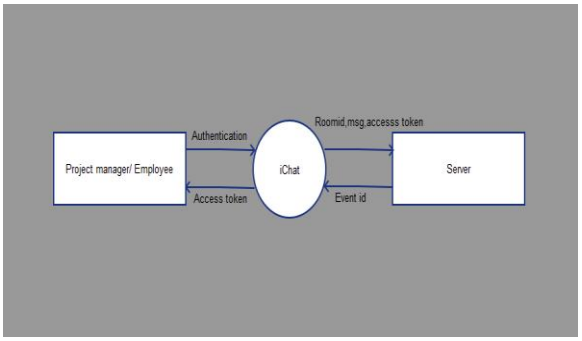


Fig 1.2 Context diagram

The context diagram shows the workflow in brief. The project manager or other employee will login using their userid and password, if the login is successful then access token is returned. After login the user can create new room, join existing room, ban a member or leave a room. When the user sends the message, along with the message room id and access token is sent to the server. If the message is sent successfully the server returns an event id.

V. DETAILED DESIGN

A. Use Case Diagram

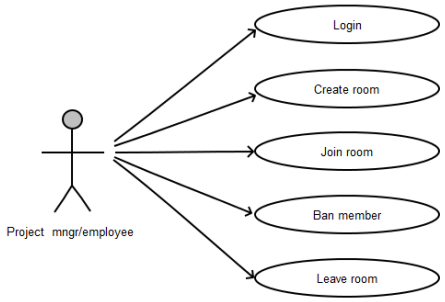


Fig 1.3 Use case diagram

The use case diagram shows that the project manager or other employees can have access to many options after logging in. They can create a new room and start communication, join a room by entering the room name, ban a member by

entering room name and the member id and leave a room after their work is done.

B. Sequence Diagram

The sequence diagram in fig 5.2 shows the sequence of activities that occur after logging in.

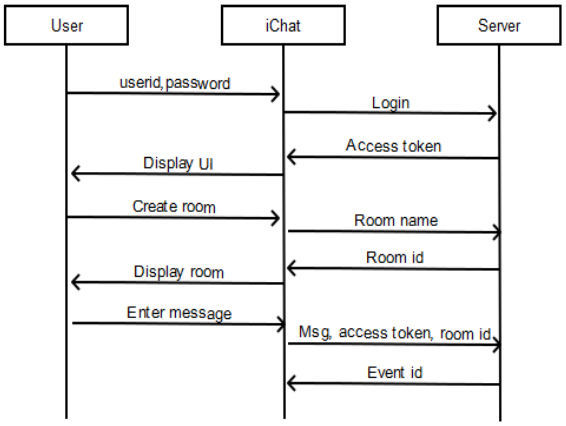


Fig 1.4 Sequence diagram

C. Data flow Diagram

Level 0 DFD

Fig 1.5 Level 0 DFD

This is a level 0 data flow diagram for project manager or other employees who has access to the home page with valid user id and password.

Level 1 DFD

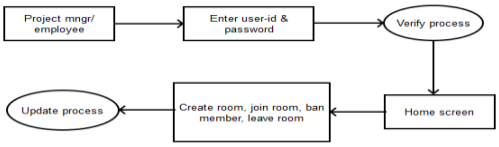


Fig 1.6Level 1 DFD

The user here logs in using the user id and password and get access to various options. The user can use these options according to its requirements. After finishing with the work the user then logs out.

D. Activity Diagram

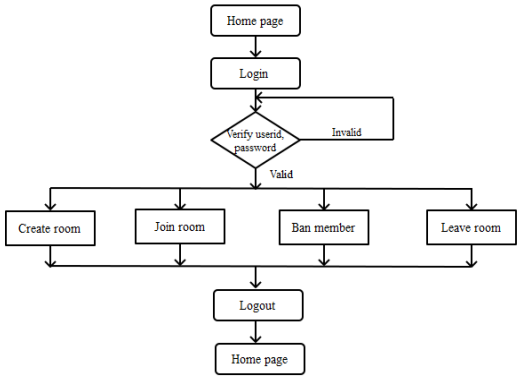


Fig 1.7 Activity diagram

Here the user will enter the unique user id and password, after login the user can use the options provided to communicate or their issues. If the user has authority then he can also ban a member from the room. The banned member will be automatically removed from that room and that member cannot join that room again.

If the user is no longer interested in a room then he can leave that room.

VI. CONCLUSION

The purpose of this application is to allow user having Internet connection to have personal or even public communication with any other user. This application is safe and secure for communication as it uses access tokens, event ids and federation structure.

This application makes it possible for many users to communicate with each other, discus their issues and solve their problems. This application also allows you to make private and public communication with another users. This application is efficient, easy to use and has good user interface. The main advantage of this application is that it is platform independent that is it can work on any operating system.

VII. FUTURE ENHANCEMENTS

This application can be enhanced in future by adding many modules which can make communication more efficient

and easy. That will also make the communication cost effective. Some of the modules that can be added are :

- Inviting a member to a room.
- Setting topic for room.
- Sharing images and files in a room.
- Making video calls.

- Profiles for each user displaying their information.

REFERENCES

[1.]The complete reference JAVA 2, fifth edition.
[2.]https://docs.oracle.com/javase/specs/jls/se8/html/jls-1.html
[3.]https://docs.oracle.com/javafx/2/overview/jfxpub-overview.htm
[4.]http://en.wikipedia.org/wiki/Kotlin_(programming_language)
[5.]http://kotlinlang.org/
[6.]http://en.wikipedia.org/wiki/Cascading_Style_Sheets
[7.]http://en.wikipedia.org/wiki/IntelliJ_IDEA
[8.]https://www.jetbrains.com/idea
[9.]https://stackoverflow.com/users/signup?returnurl=http%3a%2f%2fstack overflow.com%2f
[10.]http://www.java2s.com/Tutorials/Java/JavaFX/0400__JavaFX_Label.htm
[11.]http://tutorials.jenkov.com/java/index.html
[12.]http://www.w3schools.com/css
[13.]http://stackoverflow.com/questions/10430582/primitive-data-types-in-java
[14.]Zhen Xiao ; T. J. Watson Res. Center, IBM, Hawthorne, NY ; Guo, L. ; Tracey, J. "Understanding Instant Messaging Traffic Characteristics", Distributed Computing Systems, 2007. ICDCS '07. 27th International Conference.
[15.] Zhen Xiao ; T. J. Watson Res. Center, IBM, Hawthorne, NY ; Guo, L. ; Tracey, J. "Understanding Instant Messaging Traffic Characteristics", Distributed Computing Systems, 2007. ICDCS '07. 27th International Conference.
[16.] Zhen Xiao ; T. J. Watson Res. Center, IBM, Hawthorne, NY ; Guo, L. ; Tracey, J. "Understanding Instant Messaging Traffic Characteristics", Distributed Computing Systems, 2007. ICDCS '07.27th International Conference.