

TABLE OF CONTENTS

ACKNOWLEDGEMENT	i
ABSTRACT.....	ii
LIST OF FIGURES	iii
LIST OF ABBREVIATION	iv
1. CHAPTER ONE	1
1.1. INTRODUCTION	1
1.2. PROBLEM STATEMENT	1
1.3. OBJECTIVE	2
1.4. SCOPE AND APPLICATION	3
2. CHAPTER TWO	4
2.1. RELATED THEORY	4
2.1.1. Node.js	4
2.1.2. Passport.js	4
2.1.3. Socket.IO	5
2.1.4. jQuery	6
2.1.5. CSS	6
2.1.6. MongoDB	7
3. CHAPTER THREE	8
3.1. LITERATURE REVIEW	8
4. CHAPTER FOUR.....	10
4.1. METHODOLOGY	10
4.1.1. Waterfall Model.....	10
4.1.2. Block Diagram	12
4.1.3. Sequence Diagram	13
4.1.4. Level-0 DFD	13
4.1.5. Level-1 DFD	14

4.1.6.	ER Diagram	15
4.1.7.	Use Case Diagram.....	15
4.1.8.	Gantt Chart.....	16
5.	CHAPTER FIVE	17
5.1.	EPILOGUE.....	17
5.1.1.	Output	17
5.2.	DISCUSSION AND CONCLUSION.....	18
5.3.	LIMITATION	19
5.4.	FUTURE ENHANCEMENT.....	19
6.	CHAPTER SIX.....	20
6.1.	REFERENCES	20
6.2.	BIBLIOGRAPHY	21

ACKNOWLEDGEMENT

If words are considered as a symbol of approval and token of appreciation, then let the words play the heralding role expressing my gratitude. The satisfaction that accompanies that the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success. We are grateful to our project guide **Er. Anup Karmacharya** and **Mr. Dipraj Kayastha** for the guidance, inspiration and constructive suggestions that helped us in the preparation of this project. We also thank our colleagues who have helped in successful completion of the project.

We also take this opportunity to express a sense of gratitude to our project coordinator, **Er. Anju Khanal** for her cordial support, valuable information and guidance. We would like to appreciate the creative program of preparing of the proposal which not only boosts up the knowledge about the subject but also provides the experience and encouragement for preparing other reports on our long journey of life on which we are about to embark.

Lastly, we thank our parents and friends for their constant encouragement without which this proposal would not be possible.

ABSTRACT

Chatting system is very common communication tools that have been used in human in this modern cutting edge technology world. This chatting system has become one of the important intermediate tools for everyone to share knowledge and materials via network. So this Chatter is developed in partial fulfillment of the requirements for the Bachelor of Engineering. The design of the system depends on socket concept where there is software endpoint that establishes bidirectional communication between a server program and one or more client programs. This chatting system incorporates peer-to-peer chat as well as group chat technique. This system is built with Node.js along mongoDB as database end. The output from this system enables the users to chat with each other.

LIST OF FIGURES

Figure 1 Enhanced Waterfall Model.....	12
Figure 2 Block Diagram of Chatter	12
Figure 3 Sequence Diagram of Chatter.....	13
Figure 4 Level-0 DFD of Chatter.....	13
Figure 5 Level-1 DFD of Chatter.....	14
Figure 6 ER Diagram of Chatter	15
Figure 7 Use Case Diagram of Chatter	15
Figure 8 Gantt Chart	16

LIST OF ABBREVIATION

API	: Application Program Unit
BSON	: Binary Structured Object Notation
CSS	: Cascading Style Sheet
DFD	: Data Flow Diagram
DOM	: Domain Object Model
GUI	: Graphical User Interface
HTML	: Hypertext Markup Language
ICQ	: I See You
JSON	: Javascript Object Notation
NoSQL	: Not Only Sequence Query Language
NPM	: Node Package Manager
SMS	: Short Message Service
SVG	: Scalable Vector Graphics
TCP	: Transfer Control Protocol
UDP	: User Datagram Protocol
XHTML	: Extensive Hypertext Markup Language

1. CHAPTER ONE

1.1.INTRODUCTION

Chatting or teleconferencing is a method of using technology to bring people and ideas together despite of the geographical barriers in real-time. Online chat is a form of communication that utilizes computer program that allow for two-way conversation between users in real time (event that occur in cyber space at a same speed that they would occur in real life).

The technology has been available for years but the acceptance it was quite recent. Our project is an example of chat server. Typically, the user will connect to a chat server using a chat client and meet in a chat room. Once the users are in the same chat room, they can converse with one another by typing messages into a window where all of the other users in the chat room can see the message. The user can also see all of the messages entered by the other user. Conversation are then carried on by reading the messages entered by the other user in the chat room and responding to them.

In this paper, we focus on such a mobile chat service. We have built a framework for such a service and then evaluate the technical challenges involved in implementing it, to provide a proof-of-concept and understand any potential technical issues which may restrict such features from being implemented by mainstream mobile chat service providers.

Implementing a chat server application provides a good opportunity for a beginner to design and implement a network-based system. The design is very simple. It is implemented in node.js, since is easy to program in, it precludes the need to deal with low-level memory management and includes powerful libraries for sockets and threads.

1.2.PROBLEM STATEMENT

The mobile phone platform has evolved a long way from the original simple medium of voice and text communication to become the hub of the digital world. Mobile phones. along with being an entertainment hub, have also developed into a social construct that has affiliations and emotional attachments for individuals. It is also becoming the

predominant medium for connection with the world through social media sites/applications.

The so-called “App Culture” promoted by Apple Inc., has enabled users to seamlessly download any application they desire. This has opened the smartphone platform to a wide range of companies and services. One of the most prominent services provided by different applications on smart phones is mobile chat. It provides a potentially convenient and cost-effective alternative to traditional voice and SMS. Mobile chat services have the potential to eclipse SMS communication and this trend is becoming more obvious on a daily basis. With consumers’ increasing reliance on mobile chat services, security and privacy features are becoming serious concerns. [1]

Consumers use a mobile chat service to communicate with each other, a process that can include relaying personal information. The security and privacy of such communications should be taken seriously.

This paper deals with the security and privacy-related challenges faced in the design, development and maintenance of a chat service.

1.3.OBJECTIVE

- Server-client communication using sockets

This involves developing infrastructure to use network sockets for communication. It incorporates network sockets and figure out TCP and UDP connection using them. This will help us understand sockets in detail (with very little socket programming experience before).

- Multiple client support & GUI

We have added the support for multiple clients connecting to the server and having a chatroom kind of feature between the client (and the server acting as a relay/broadcast agent). Also we have developed a simple GUI interface for easeness for users.

1.4.SCOPE AND APPLICATION

This project can be mainly divided into two modules.

- Server
- Client

This project is mainly depended on single server – multiple client models. The client requests the server and server responses by granting the clients request. The following specifications are implemented:

- The server and client are two separate programs.
- Multiple clients are able to connect to a single server.
- All input and output is via I/O Interface.
- Client:
 - Client is able to choose a nickname on connection.
 - Client is show when another client connects or disconnects with the server.
 - Client is able to send messages to the server before arriving in other client.
 - Client is able to receive messages while writing.
- Server:
 - The server does not allow for more than one client to get the same nickname.
 - Server is able to return messages again to all clients (including source).
 - Client connected and disconnected with the server does not crash the server.

2. CHAPTER TWO

2.1.RELATED THEORY

2.1.1. Node.js

Node.js is a runtime environment and library for running JavaScript applications outside the browser. Node.js is mostly used to run real-time server applications and shines through its performance using non-blocking I/O and asynchronous events. A complete web ecosystem has been built around node.js with several web app frameworks and protocol implementations available for usage. It's definitely one of the easiest and fastest way to develop real-time applications on the web today. JavaScript is an extremely popular language and is credited with being one of the driving forces that turned the web into the dynamic wonderland that it is today. What you can do in a browser nowadays, rivals all others! JavaScript arose on the frontend but - thanks to the V8 JavaScript engine and the work of Ryan Dahl - you can now run networked JavaScript applications outside of the browser precisely to build web apps. Node.js lets you unify the programming language used by your app - no longer do you need a different language for your backend, you can use JavaScript throughout. If your background is in building and design websites and web app frontends in HTML, CSS and JavaScript, you don't need to pick up another language to develop complex data-driven back-ends for your apps. Node.js plays a critical role in the advancement of WebSockets as a method for real-time communication between the front and back ends. The use of WebSockets and the libraries building on that protocol such as Socket.IO have really pushed what is expected of web applications and lets us developers explore new ways to create the web. [2]

2.1.2. Passport.js

Passport is authentication middleware for Node. It is designed to serve a singular purpose: authenticate requests. When writing modules, encapsulation is a virtue, so Passport delegates all other functionality to the application. This separation of concerns keeps code clean and maintainable, and makes Passport extremely easy to integrate into an application. [3]

In modern web applications, authentication can take a variety of forms. Traditionally, users log in by providing a username and password. With the rise of social networking, single sign-on using an OAuth provider such as Facebook or Twitter has become a popular authentication method. Services that expose an API often require token-based credentials to protect access.

Passport recognizes that each application has unique authentication requirements. Authentication mechanisms, known as strategies, are packaged as individual modules. Applications can choose which strategies to employ, without creating unnecessary dependencies.

2.1.3. Socket.IO

Socket.IO is a JavaScript library for realtime web applications. It enables realtime, bi-directional communication between web clients and servers. It has two parts: a client-side library that runs in the browser, and a server-side library for node.js. Both components have a nearly identical API. Like node.js, it is event-driven.

Socket.IO primarily uses the WebSocket protocol with polling as a fallback option, while providing the same interface. Although it can be used as simply a wrapper for WebSocket, it provides many more features, including broadcasting to multiple sockets, storing data associated with each client, and asynchronous I/O. It can be installed with the npm tool.

Socket.IO provides the ability to implement real-time analytics, binary streaming, instant messaging, and document collaboration. Notable users include Microsoft Office, Yammer, and Zendesk.

Socket.IO handles the connection transparently. It will automatically upgrade to WebSocket if possible. This requires the programmer to only have Socket.IO knowledge.

Socket.IO is not a WebSocket library with fallback options to other realtime protocols. It is a custom realtime transport protocol implementation on top of other realtime protocols. Its protocol negotiation parts cause a client supporting standard WebSocket to not be able to contact a Socket.IO server. And a Socket.IO implementing client cannot talk to a non-Socket.IO based WebSocket or Long Polling Comet server.

Therefore, Socket.IO requires using the Socket.IO libraries on both client and server side. [4]

2.1.4. jQuery

jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. jQuery is the most popular JavaScript library in use today, with installation on 65% of the top 10 million highest-trafficked sites on the Web. jQuery is free, open-source software licensed under the MIT License.

jQuery's syntax is designed to make it easier to navigate a document, select DOM(Dynamic Object Model) elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, themeable widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and Web applications.

jQuery also provides a paradigm for event handling that goes beyond basic DOM element selection and manipulation. The event assignment and the event callback function definition are done in a single step in a single location in the code. jQuery also aims to incorporate other highly used JavaScript functionality (e.g. fade ins and fade outs when hiding elements, animations by manipulating CSS properties). [5]

2.1.5. CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language. Although most often used to change the style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any kind of XML document, including plain XML, SVG and XUL. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of document content from document presentation, including elements such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity

and repetition in the structural content, such as semantically insignificant tables that were widely used to format pages before consistent CSS rendering was available in all major browsers. CSS makes it possible to separate presentation instructions from the HTML content in a separate file or style section of the HTML file. For each matching HTML element, it provides a list of formatting instructions. For example, a CSS rule might specify that "all heading 1 elements should be bold", leaving pure semantic HTML markup that asserts "this text is a level 1 heading" without formatting code such as a `<bold>` tag indicating how such text should be displayed.

This separation of formatting and content makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on Braille-based, tactile devices. It can also be used to display the web page differently depending on the screen size or device on which it is being viewed. Although the author of a web page typically links to a CSS file within the markup file, readers can specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author has specified. The CSS specification describes a priority scheme to determine which style rules apply if more than one rule matches against a particular element. [6]

2.1.6. MongoDB

MongoDB is a free and open-source cross-platform document-oriented database. Classified as a NoSQL database, MongoDB avoids the traditional table-based relational database structure in favor of JSON-like documents with dynamic schemas (MongoDB calls the format BSON), making the integration of data in certain types of applications easier and faster. MongoDB is developed by MongoDB Inc. and is free and open-source, published under a combination of the GNU Affero General Public License and the Apache License. As of July 2015, MongoDB was the fourth most widely mentioned database engine on the web, and the most popular for document stores. [7]

3. CHAPTER THREE

3.1.LITERATURE REVIEW

Internet communication is getting more and more popular among the public. Apart from using telephones and sending mails, people can now communicate with each other through the chat technology. The chat, in fact, is a kind of Internet technology that supports human-to-human communication. ICQ, for instance, is one of the latest chat. Over the past two years, with the advanced level of technology, there is an increasing trend of using ICQ for communication. With ICQ, users can chat, send messages, files and URL's or play games with others users in real time. Because of the proliferation of using the chat, studies have been focused mainly on its impact on our society.

Much of the work stresses the good impact of the chat. Hauben's writing suggested that as the impact or influence of first impressions is removed, users are free to communicate without fears, limits or apprehension through the chat. This statement actually points out the main reason for the increasingly use of the chat. Only one advantage, however, seems inadequate to attract such a huge number of users to use the chat, so it seems that there may be other benefit. [8] Accordingly, Licklider claimed that people can communicate online with others who have similar goals and interests, thus their life will be enriched and communication will be more productive and more enjoyable then. Although Licklider is actually the prophet of the Net, it seems that the chat really has this benefit. [9]

In Randall's viewpoint, on the other hand, the purpose of people who use the chat is for socializing. But he emphasized that such kind of socializing is different from that in the real world, as the former only involves the exchange of words with other users but the latter means to interact with others face-to-face. While the trend of using the chat is increasing, Randall suggested that children and youths will be discouraged from the normal social contact but will adopt cyberspace contact instead. However, although the chance of happening this phenomenon is quite high, it seems neither right nor wrong according to Randall. [10]

To sum up, the chat has good impact on the society but problems exist at the same time. However, these problems are not serious in fact. Therefore, even if these problems exist continuously, the chat technology will still become central to our lives and it has already begun actually.

Today, most chat rooms use the current connection by sending messages. Some chat rooms feature the ability to use audio and video to communicate and upload images almost instantly. Most chat rooms require registration and a password. It all depends on who is running the chat room, which monitors, if anyone. As an example Yahoo has a long list of chat rooms by topic. Also Pogo site is a popular site allows people to chat while playing games online. To use the chat rooms, the user must first register and choose a user name, and will be introduced at the talks. Username is usually not the legal name of the person. Once the user enters the chat rooms (by typing your user name and password), and he could see a list of people who are already in the room. To talk with these people, the user must write a message in the text box. Once the user clicks on the key "enter", you can see anyone else his or her message. Then other people may respond, and the development of the conversation. Sometimes, people walk in chat rooms and just read messages without taking part in the conversation.

4. CHAPTER FOUR

4.1.METHODOLOGY

4.1.1. Waterfall Model

For this project we have decided on using enhanced waterfall model because as we go on developing software there may come a time when we need to go to initial phase and make some changes. Here is the brief description of this model:

The Waterfall Model was first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.

Waterfall Model design

Waterfall approach was first model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

The sequential phases in Waterfall model are:

Requirement Gathering and analysis: All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.

System Design: The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

Implementation: With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

Integration and Testing: All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

Deployment of system: Once the functional and non functional testing is done, the product is deployed in the customer environment or released into the market.

Maintenance: There are some issues which come up in the client environment. To fix those issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model phases do not overlap.

Waterfall Model Application

Every software developed is different and requires a suitable approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are:

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

Advantages of waterfall model:

The advantage of waterfall development is that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one. Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

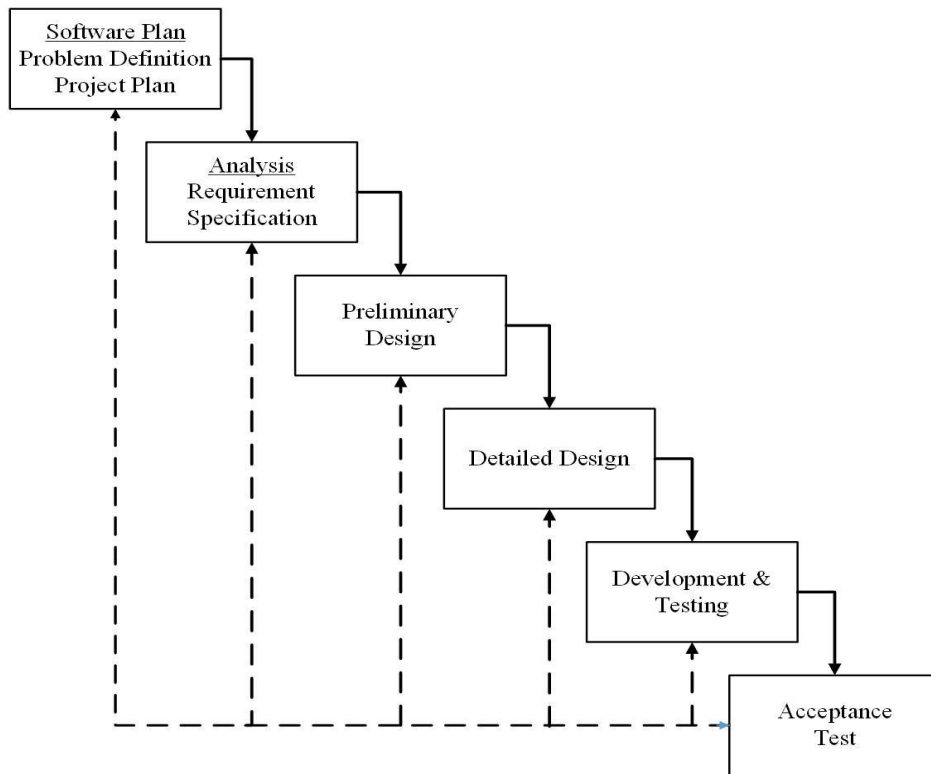


Figure 1 Enhanced Waterfall Model

4.1.2. Block Diagram

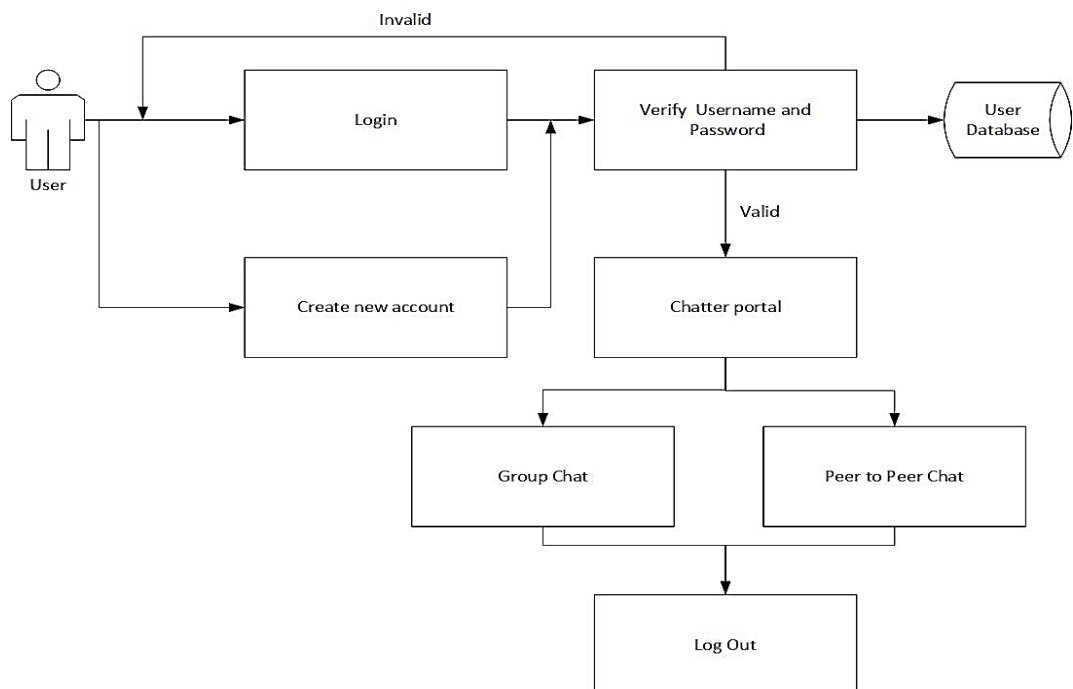


Figure 2 Block Diagram of Chatter

4.1.3. Sequence Diagram

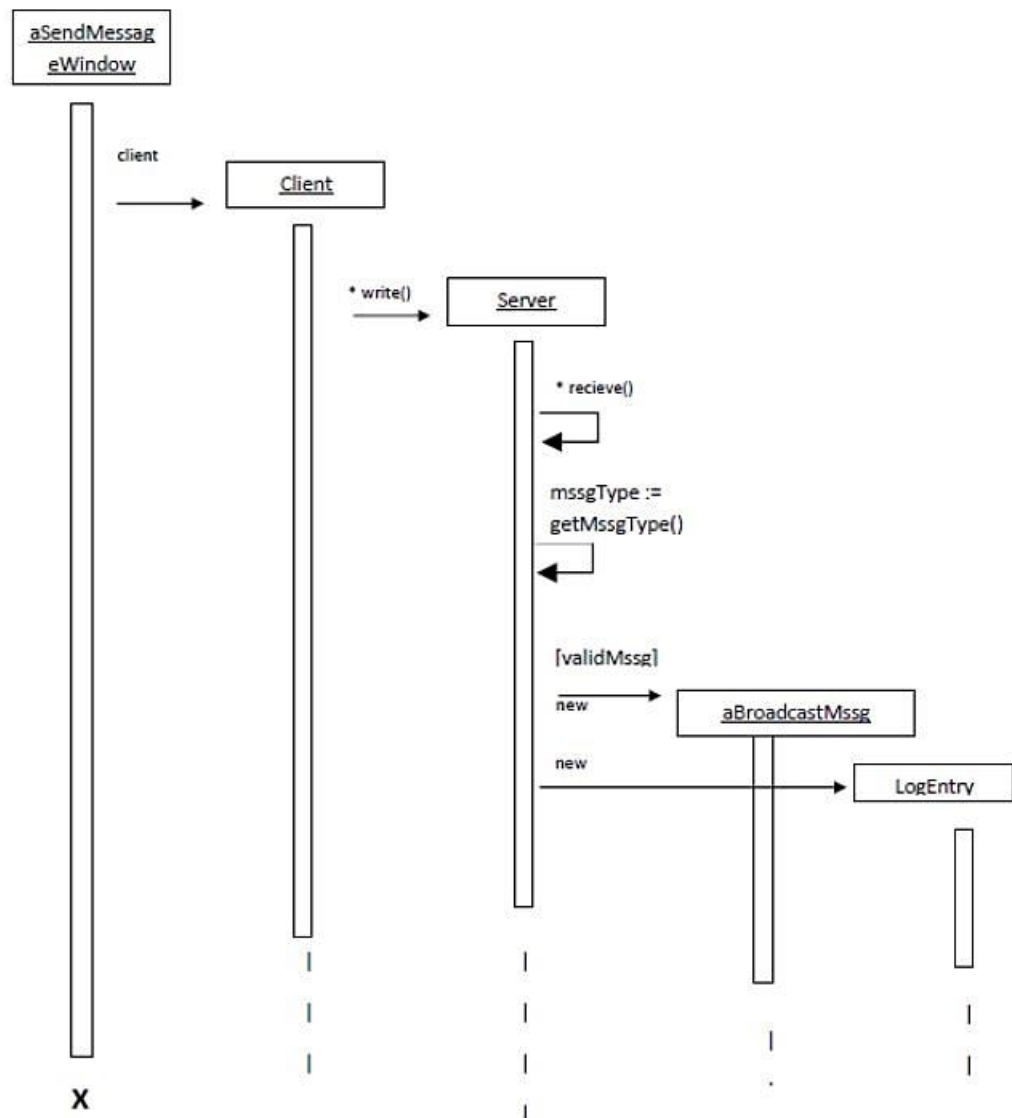


Figure 3 Sequence Diagram of Chatter

4.1.4. Level-0 DFD

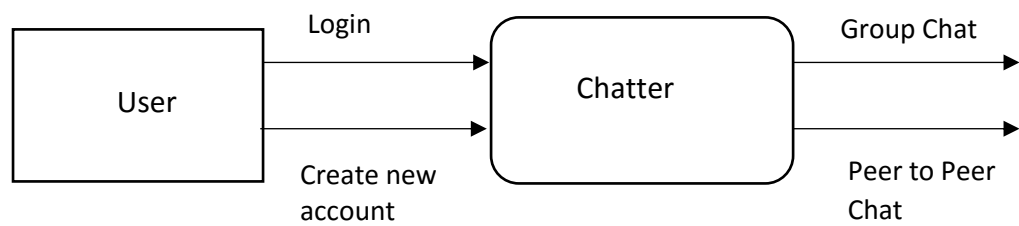


Figure 4 Level-0 DFD of Chatter

4.1.5. Level-1 DFD

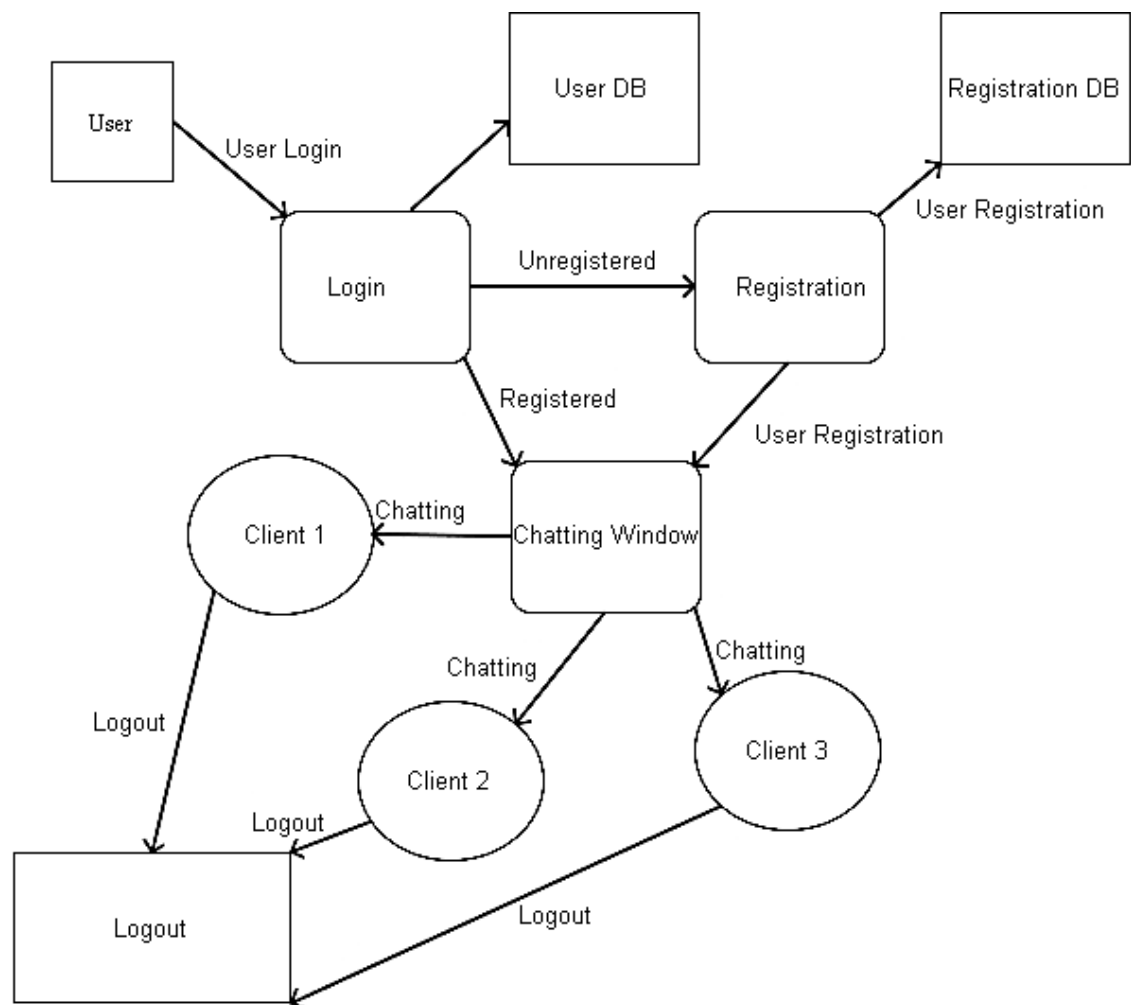


Figure 5 Level-1 DFD of Chatter

4.1.6. ER Diagram

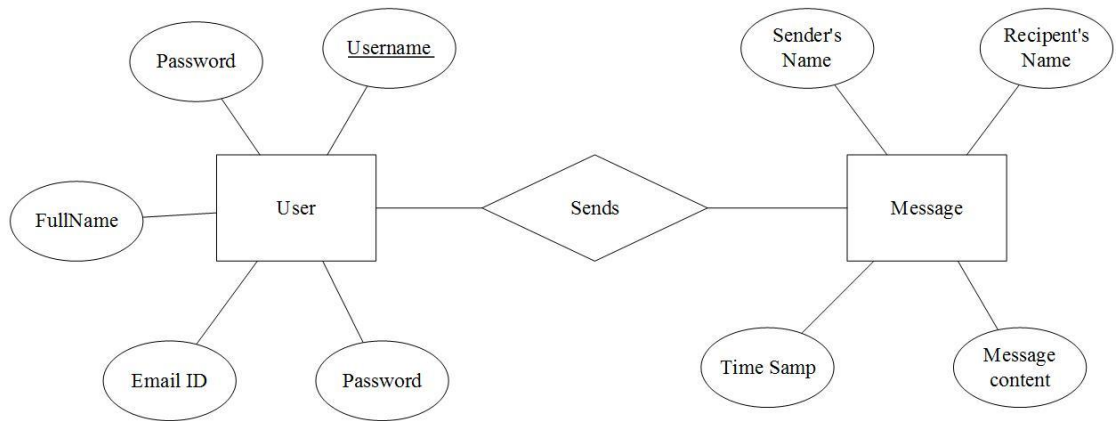


Figure 6 ER Diagram of Chatter

4.1.7. Use Case Diagram

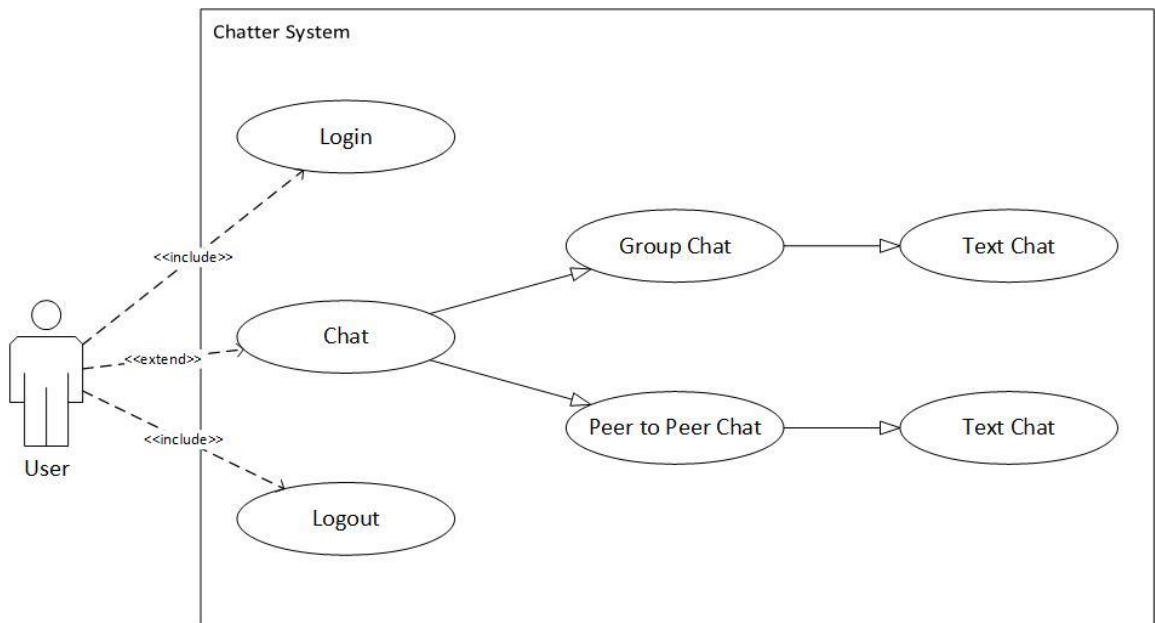


Figure 7 Use Case Diagram of Chatter

4.1.8. Gantt Chart

Preliminary Study									
Detail system study									
System analysis									
System design									
Coding									
Testing and debugging									
Documentation									
	Poush 2072	Magh 2072	Falgun 2072	Chaitra 2072	Baishak 2073	Jestha 2073	Ashad 2073	Shrawan 2073	Bhadra 2073

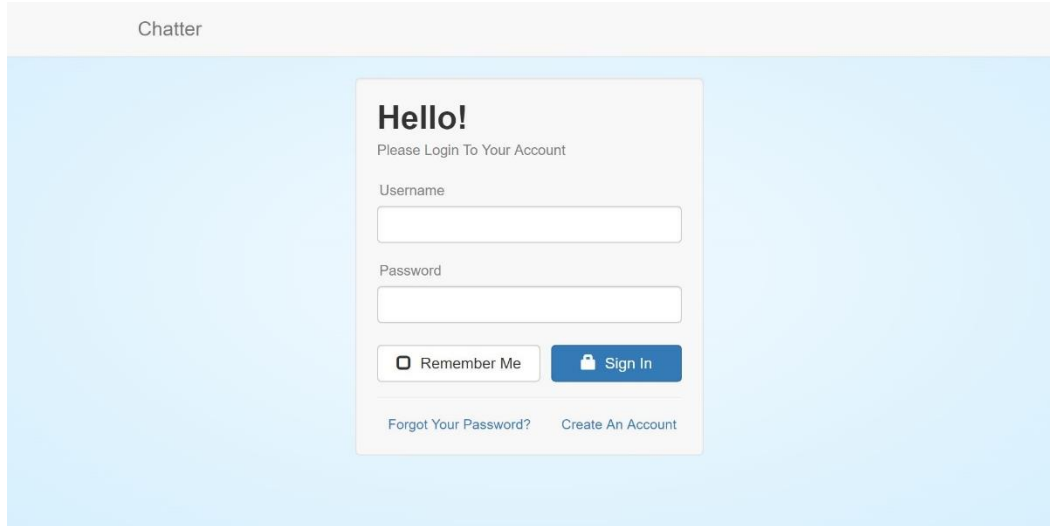
Figure 8 Gantt Chart

5. CHAPTER FIVE

5.1.EPILOGUE

5.1.1. Output

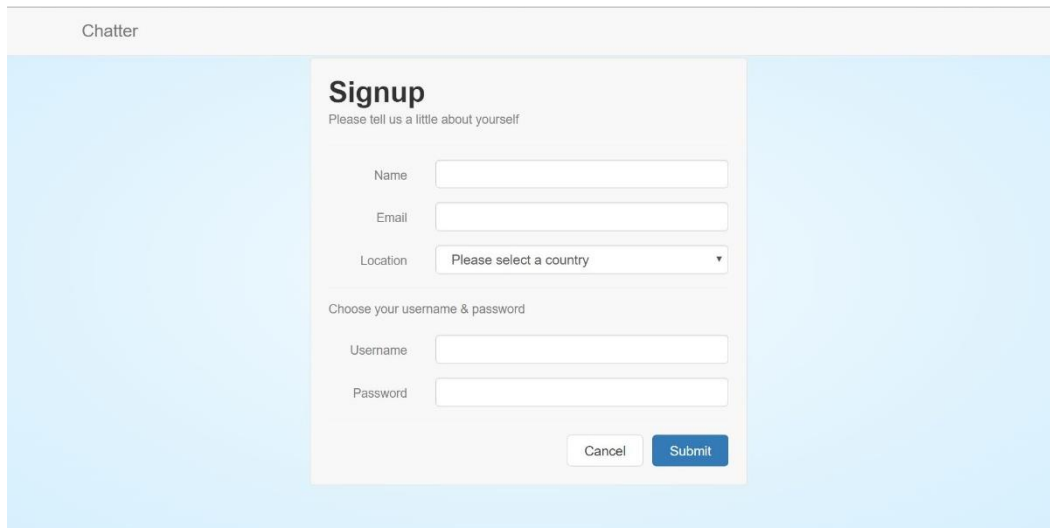
Login Page



The screenshot shows a web application interface with a light blue background. At the top, there is a header bar with the word "Chatter" in a small, dark font. Below the header, there is a central white box containing the login form. The form is titled "Hello!" in a bold, dark font, followed by the instruction "Please Login To Your Account". Below this, there are two input fields: "Username" and "Password". To the right of the "Password" field, there is a "Remember Me" checkbox with the text "Remember Me" next to it. Below the input fields, there is a blue button with the text "Sign In". At the bottom of the form, there are two links: "Forgot Your Password?" and "Create An Account".

Screenshot 1: Login Page/Home Page

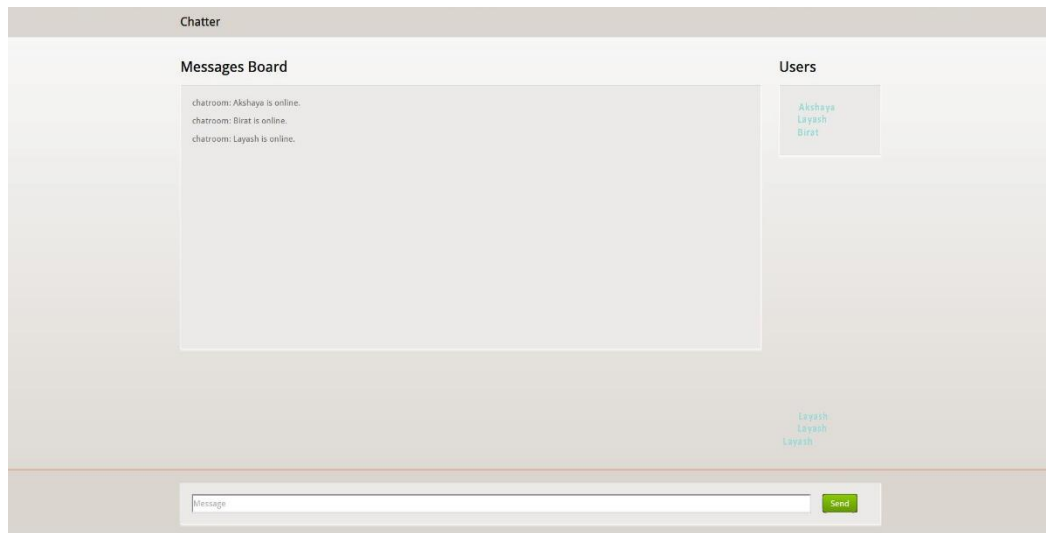
New User Sign Up Page



The screenshot shows a web application interface with a light blue background. At the top, there is a header bar with the word "Chatter" in a small, dark font. Below the header, there is a central white box containing the sign-up form. The form is titled "Signup" in a bold, dark font, followed by the instruction "Please tell us a little about yourself". Below this, there are three input fields: "Name", "Email", and "Location". The "Location" field is a dropdown menu with the text "Please select a country" and a downward arrow. Below the input fields, there is a section titled "Choose your username & password" with two input fields: "Username" and "Password". At the bottom of the form, there are two buttons: "Cancel" and "Submit".

Screenshot 2: New User Sign Up Page

Chat Portal



Screenshot 3: Chat Portal

5.2.DISCUSSION AND CONCLUSION

After a year worth of hard work in submitting proposal, design, coding and development, our team finally completed the project based on node.js, socket.io and mongoDB.

In this process, some difficulties came up mostly about programming language and connecting with database. We found out relational information on internet and researched open source coding. The problems were solved after we understood and we modified code in few other node.js projects.

After carefully research few node projects, we learned a good professional knowledge to integrated use. We got a deeper understanding from many of abstract and theoretical knowledge before. At the same time, we learned how to design and development the socket programs using the structured programming and models to achieve functions.

But also we still have many additional changes that can be added We think it is lack of insufficient programming skills and limited time. We would add these changes to make this system even more effective when we have enough time. It is very useful for future work to us.

Thus in this way with team effort and co-ordination we were able to successfully complete our project. This project is a result of a lot of hard work and in this process we also gained a lot of knowledge regarding node.js and databases.

5.3.LIMITATION

Following are the few limitations of our project in the first version.

- Users are limited to text chat only.
- The emoji feature is not yet available.
- “Forgot your Password?” option in home screen is not yet fully completed. So in case, users forget his/her password, he/she has to create a new credential.

5.4.FUTURE ENHANCEMENT

We know that no system is perfect at first and gradually with time it can be improved and updated regularly. Our system is also no exception. It still has room for improvements and the following enhancements can be added to it:

- Addition of file transfer, voice chat and video conferencing.
- Add new keyboards to the portal.
- Listing all the users.
- Searching users.
- Send offline messages to users.
- Group the users in the basis of workplace or other category.
- Implement drag-drop for file transfer.

6. CHAPTER SIX

6.1.REFERENCES

- [1] Waleed, Farah. *Developing a Chat Server*. 2000.
- [2] Mardan, Azat. *Practical Node.js: Building Real-World Scalable Web Apps*. Apress, July 10, 2014
- [3] "passportjs.org." n.d. *passportjs.org/docs*. 04 March 2016.
- [4] Mardan, Azat. *Practical Node.js: Building Real-World Scalable Web Apps*. Apress, July 10, 2014.
- [5] Castledine, Earle and Craig Sharkie. *jQuery: Novice to Ninja: New Kicks And Tricks*. Sitepoint, 2013.
- [6] Mozilla Developer Network. "mozilla.org." n.d. *developer.mozilla.org/en-US/docs/Web/Guide/CSS*. 10 March 2016.
- [7] MongoDB Developer Manual. "mongodb.com." n.d. *docs.mongodb.com/manual/introduction/*. 010 March 2016.
- [8] Hauben. M (1997). Chapter 6: Cybernetics, Time-sharing, Human-Computer Symbiosis and Online Communities: Creating a Supercommunity of Online Communities, Netizens(76-95).Los Alamitos: Computer Society Press.
- [9] Hauben .M (1997). Chapter 1:The Net and Netizens: The Effect the Net Has on People's Lives, Netizens(3-34). Los Alamitos: Computer Society Press.
- [10] Randall. N (1997). Chapter 19: Future Communities: Socializing and Educating in the Internet's Future, The soul of the Internet: net gods, netizens and the wiring of the world(pp.333-344). London: Computer Press.

6.2.BIBLIOGRAPHY

- Freier, P. Karlton, and P. Kocher. (2011, August) RFC:6101 - The Secure Sockets Layer (SSL) Protocol Version 3.0. Online. IETF.
- M. La Polla, F. Martinelli, and D. Sgandurra, A Survey on Security for Mobile Devices, Communications Surveys & Tutorials, IEEE, pp. 446471, 2013.
- <https://www.alljoyn.org>
- Open Source Chat Servers in Java <http://java-source.net/open-source/chat-servers>
- <http://blogs.msdn.com/b/cdn devs/archive/2014/09/04/node-js-tutorial-series-a-chatroom-for-all-part-1-introduction-to-node.aspx>
- Paper on, "Web-based Mobile Apps of the Future Using HTML 5, CSS and JavaScript",HTMLGoodies, Retrieved October 2014
- H. Korth and A. Silberschatz, Database system concepts. New York: McGraw-Hill, 1986.
- R. Nixon, Learning PHP, MySQL, JavaScript, and CSS. Sebastopol, CA: Oreilly, 2012.
- Dayley, Brad. *Node.js, MongoDB, and AngularJS Web Development (Developer's Library)*. Addison-Wesley Professional, June 28, 2014.
- MongoDB Developer Manual. "mongodb.com." n.d. docs.mongodb.com/manual/introduction/. 010 March 2016.
- *npmjs.com*. n.d. 04 March 2016.