

MODERN SOFTWARE ENGINEERING CS 790

PROJECT DOCUMENTATION

TEAM 3

BATCHANGARI ROHITH CHANDRA GOUD

BHARGAV VECHA

VINAY KUMAR REDDY KANDULA

PRODUCT NAME: PRIVATE CHAT BOX

PRODUCT DESCRIPTION

Development of a mode of communication between users, by developing a chat box in order to provide privacy. The end user will have a user interface using which they can respond and create messages to be shared between each other. Technical capabilities required to perform this task are, having knowledge of Javascript, CSS (Cascading Style Sheets) and using Microsoft Visual Studio.

The potential audience of this product can be, students of a class, a group of friends, strangers who want to share their insights on a topic and for users wanting to share messages or communicate. This software will provide users with a chat room where about 25 people in the same time can communicate with each other using text and emojis.

The most important feature of this product is that it can be used without providing a mobile number, E-mail address or any social media account information unlike most of the applications that are found on the web. No personal information is shared on the internet and is known by the peer user as privacy is one of the most concerning issues in today's world. The users can join a personal chat room by entering a specific key shared among them, to communicate and the whole chat will be deleted once they leave the chat session.

TEAM DESCRIPTION

This product is developed using Javascript, where all the backend process is developed using NodeJS and frontend will be developed using ReactJS along with CSS, therefore the team members should exhibit knowledge in coding JS and also using Microsoft Visual Studio which will be used to write the code. The team members should also have knowledge in using Modified Waterfall Software Process Model, testing and debugging the software. Strong motivation, teamwork and communication between team members will ensure successful development of the software.

SOFTWARE PROCESS MODEL

Software Process Model:

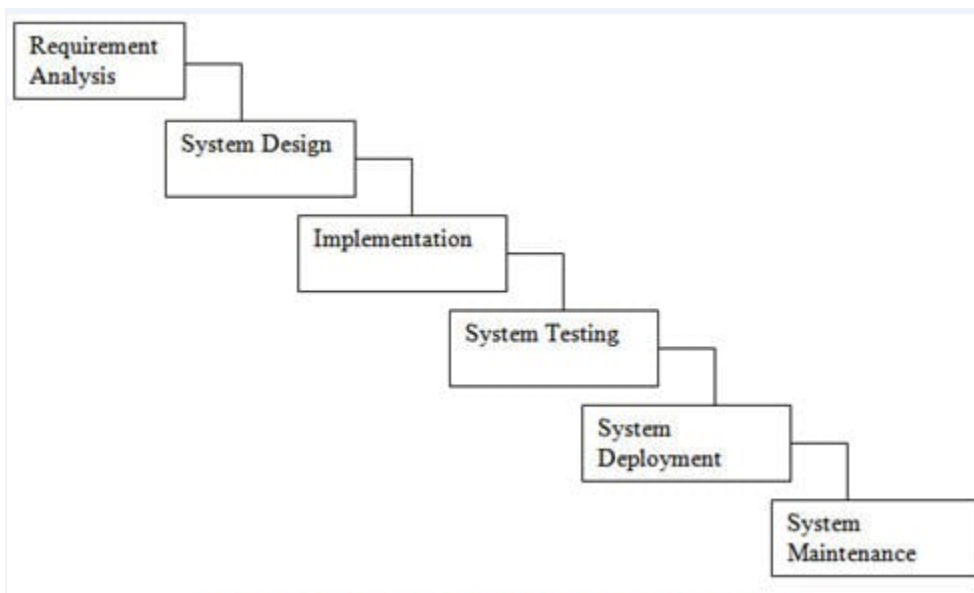
A software process model helps us follow a systematic procedure in developing software. The general process steps are:

- Requirements
- User Interface Design
- System Design
- Program Development (design and coding)
- Acceptance and release.

We will be using the Modified Waterfall Model.

Waterfall Model:

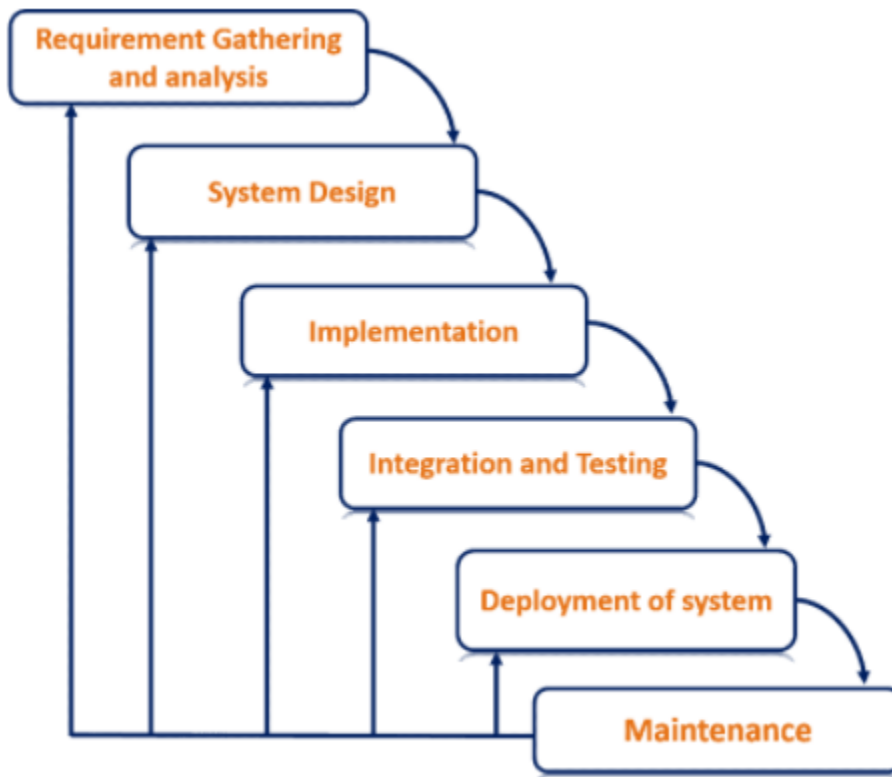
The waterfall model is a heavyweight process with full documentation of each process step. It is easy to understand and use. In the waterfall model, each phase must be completed before starting the new phase. This model illustrates the software development process in a linear sequential flow. In the waterfall approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. The following are different phases implemented in the waterfall model.



The disadvantage of the waterfall model is that it does not allow much revision. Once an application is in the testing phase, it is difficult to go back and change as the steps are documented before completion of the step.

Modified Waterfall Model:

The main difference between pure waterfall model and modified waterfall model is the feedback system which provides previous phases. This model works better when the requirements are well understood. It allows developers to return to a previous phase for verification or validation, confining to connected steps. The modified waterfall model uses the same phases as the pure waterfall model. This enables the phases to overlap when needed. The modified waterfall can also split into sub projects at an appropriate phase.



We have chosen the Modified Waterfall Model because of the feedback system. While in the developing phase, if we aren't clear about the requirements or if the requirements couldn't be implemented, this model provides the flexibility to go back to the previous steps and go through the requirements. If there is continuity between phases, then documentation can be reduced. While in the testing phase, if there are any errors in the software, they can be rectified by moving back to the development phase. Our product will be developed in Frontend and Backend, where frontend will be user interface and backend will be the implementation, therefore Modified Waterfall Method will allow the developers to perform more rigorous checking for this kind of project where both the operations can be monitored, altered and be updated as necessary.

PROJECT DEFINITION

This product can be used and will be helpful to people who need to communicate with one other without providing their personal information. They can use this product to text and message one another for specific needs, for example a group of students in a class working on a small one-day project where they need to

communicate can easily join the chat room. They don't need to worry about privacy or the chat session as it will be deleted once they leave the session. Another use case can be a group of strangers who met at an event or at a library and found to be sharing the same interest in a technology or a topic or a book, can text one another, share their views without giving up their mobile number or E-mail ids. The users will be provided with an interactive chat interface where they can enter multiple line text messages in a text box and can use emojis and send it. The receiver will get the message instantly and can reply. The chat heads will be the names that can be given by the users themselves at the beginning of the chat session before they join the chat box.

VALIDATION PLAN

Firstly, the user will be prompted to enter their name of choice.



The image shows a rectangular form with a dark border. Inside the form, the text "Enter Your Name" is centered at the top. Below this text is a horizontal rectangular text input field. At the bottom right of the form, the word "Next" is displayed, likely representing a button or a link to proceed to the next step.

After the user enters his name, they will be asked for the chat room key they want to enter.

Enter Chat Room Key

Enter

Finally, the user can enter the chat room and text to other users.

Chat Room

User 1

Text....

User 2

Text...

Enter your message

😊

CONFIGURATION/ VERSION CONTROL

This project will be implemented and deployed with all the specifications mentioned, without any incremental additions after deployment, therefore no Revision Control Software is being used for this project.

TOOLS

The Software and Hardware tools required for this project are:

A computer with i5 or greater generation processor.

4GB or more RAM

512 GB of storage.

Windows 10 or Mac OS.

Microsoft Visual Studio.

Notepad++

The technologies used are: ReactJS, NodeJS, Express JS, CSS, Emojify

Technologies Used

React JS:

React is a JavaScript library for building user interfaces. It is maintained by Facebook and a community of individual developers and companies. React can be used as a base in the development of single-page or mobile applications.

Advantages:

1. It facilitates the overall process of writing components.
2. It boosts productivity and facilitates further maintenance.
3. It ensures faster rendering.
4. It guarantees stable code.
5. It is SEO friendly.
6. It comes with a helpful developer toolset.
7. There is React Native for mobile app development.

Node JS:

Node.js is a server-side JavaScript run-time environment. Node.js is primarily used for non-blocking, event-driven servers, due to its single-threaded nature. It's used for traditional web sites and back-end API services, but was designed with real-time, push-based architectures in mind.

Express JS:

Express JS, is a back end web application framework for Node.js. ExpressJS is a prebuilt NodeJS framework that can help you in creating server-side web applications faster and smarter. Simplicity, minimalism, flexibility, scalability are some of its characteristics

CSS

CSS stands for Cascading Style Sheets. CSS describes how HTML elements are to be displayed on screen, paper, or in other media. CSS saves a lot of work. It can control the layout of multiple web pages all at once.

Emojify:

Emojify is an extension that lets you view and input Emojis in chrome on webpages! How to enter emojis: Click Emojify's button in chrome's extension area and the emoji keyboard will pop up inside the webpage you're currently browsing.

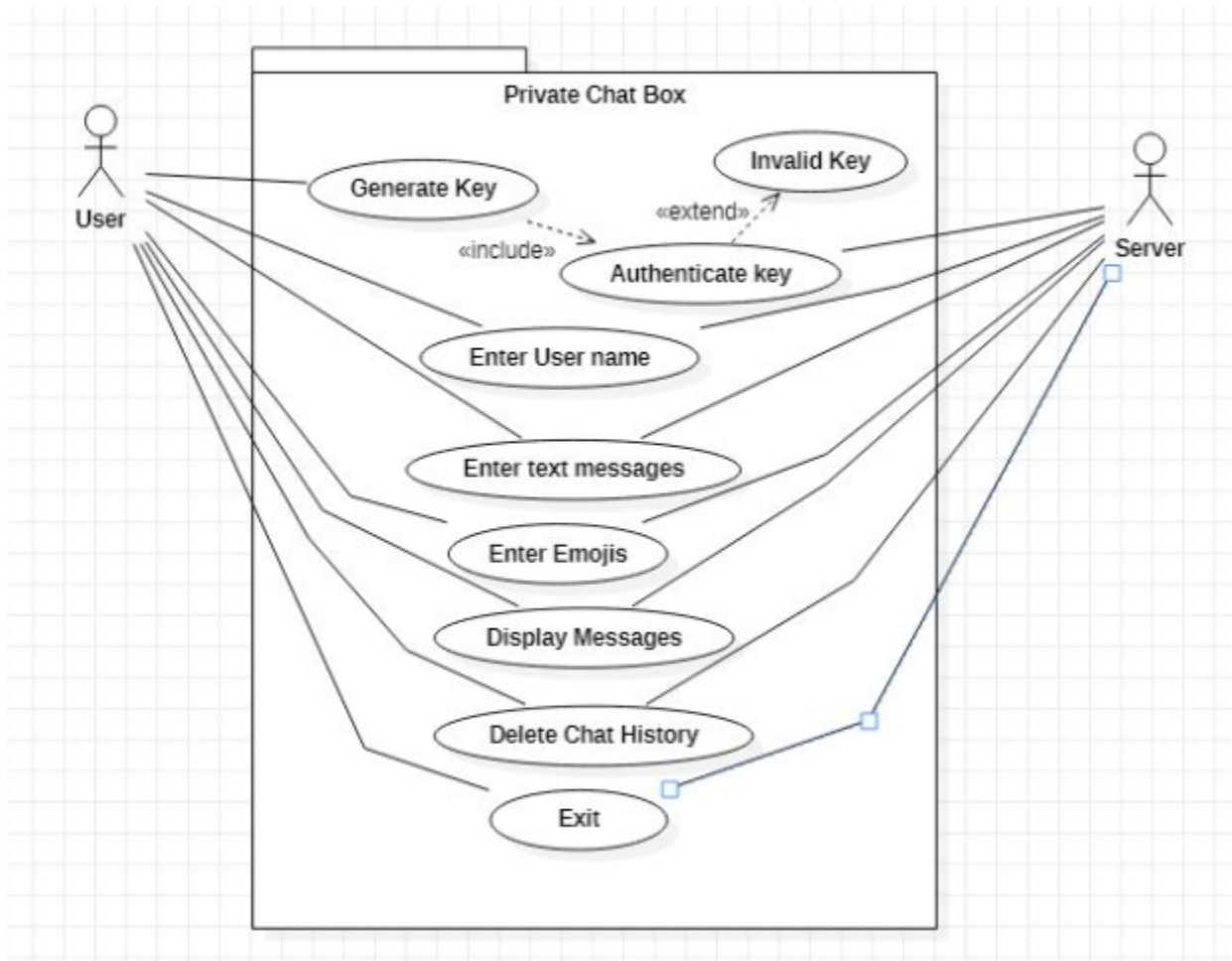
SYSTEM OVERVIEW:

Development of a mode of communication between users, by developing a chat box in order to provide privacy. The end user will have a user interface using which they can respond and create messages to be shared between each other. The potential audience of this product can be, students of a class, a group of friends, strangers who want to share their insights on a topic and for users wanting to share messages or communicate. This software will provide users with a chat room where about 25 people at the same time can communicate with each other using text and emojis. The most important feature of this product is that it can be used without providing a mobile number, E-mail address or any social media account information unlike most of the applications that are found on the web. No personal information is shared on the internet and is known by the peer user as privacy is one of the most concerning issues in today's world. The users can join a personal chat room by entering a specific key shared among them, to communicate and the whole chat will be deleted once they leave the chat session.

TECHNICAL REQUIREMENTS:

Technical capabilities required to perform this task are, having knowledge of Javascript, CSS (Cascading Style Sheets) and using Microsoft Visual Studio.

Functional Requirements:



Actors: User, Server.

Entry Condition: The user either needs to generate the key to create a new chat room or enter the key to an existing chatroom.

Use Cases:

- **Generate Key:** When the user needs to create a new chat room, he needs to generate a key.
- **Authenticate Key:** The key is then authenticated in the server to check whether the key exists or not.
- **Invalid Key:** If the key entered is not present in the server, the user receives an error message stating the key is invalid.

- **Enter Username:** The user needs to enter a username which is a display name. It appears in the chat when we send the message.
- **Enter Text Message:** The user enters his message in the text area and presses enter. The message gets delivered in the chat room and other members can see it.
- **Enter Emojis:** The user can also communicate with emojis.
- **Display Messages:** The user can see the messages sent by other users.
- **Delete Chat:** When the users leave the chat room, the messages are all cleared in the server.

Nonfunctional Requirements:

Security: As no personal information is taken, there is no issue of data theft. The messages in the chat room are also deleted when the room is left. This increases the security of data.

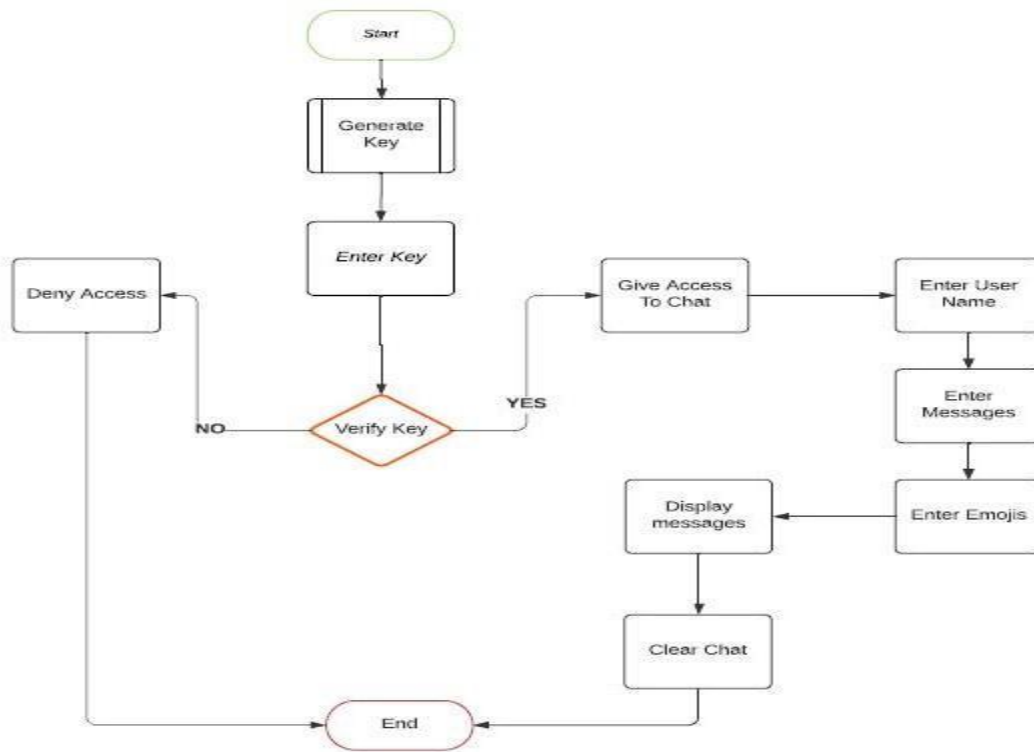
Capacity: The number of people allowed per room are restricted to 25.

Performance: As we use socket, it helps in connecting the threads and response time will be less. It also increases throughput as it orders the threads and processes them.

Reliability: As no information is taken from the user, there is no issue of loss of personal information.

- **User Interface Specification:** In the user interface, we can observe there will be an option to enter the message, send emojis and view the messages.
- **User Task Flow:**

Task Flow Diagram



- **Input/ Output Data Specification:** The user enters the message in the text area and sends it into the chat box. The user can also communicate through emojis. He can also view the messages sent by other users present in the same room.

Acceptance Criteria:

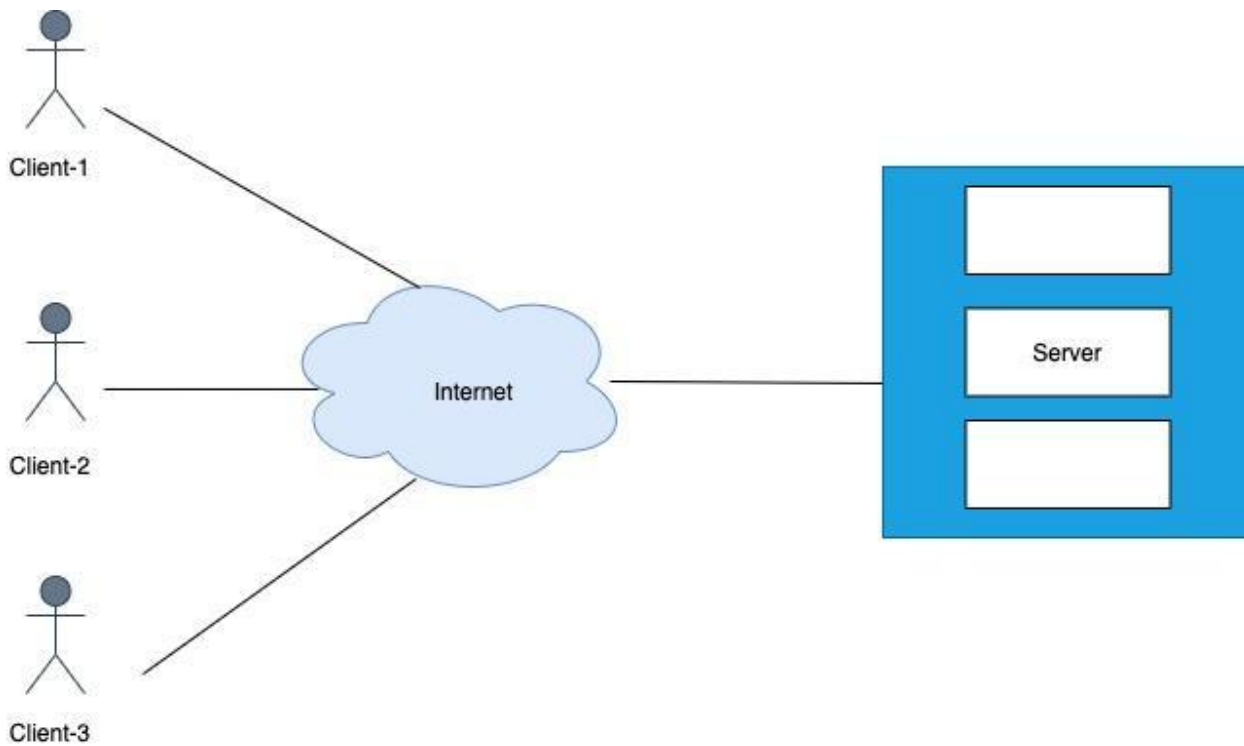
- How can I access the chat box? A user generates a key and shares it with his classmates or friends. When they enter the same key, they all enter a chat room where they can talk to others through messages and emojis.
- How secure is the chat? Once we exit the chat room, the messages are cleared from the server.
- What happens when members in the chat room message at the same time? Based on the incoming timestamp of the message in the server, the messages are displayed in that order.

Verification/ Validation:

- The software undergoes testing on all its features.
- We check whether the key is generated and is getting validated or not.
- We check for messages getting delivered and received properly.
- The clearing of the messages is done at the server. It is done with utmost care as it must be cleared totally from the server. This feature undergoes rigorous testing.

Architectural Design

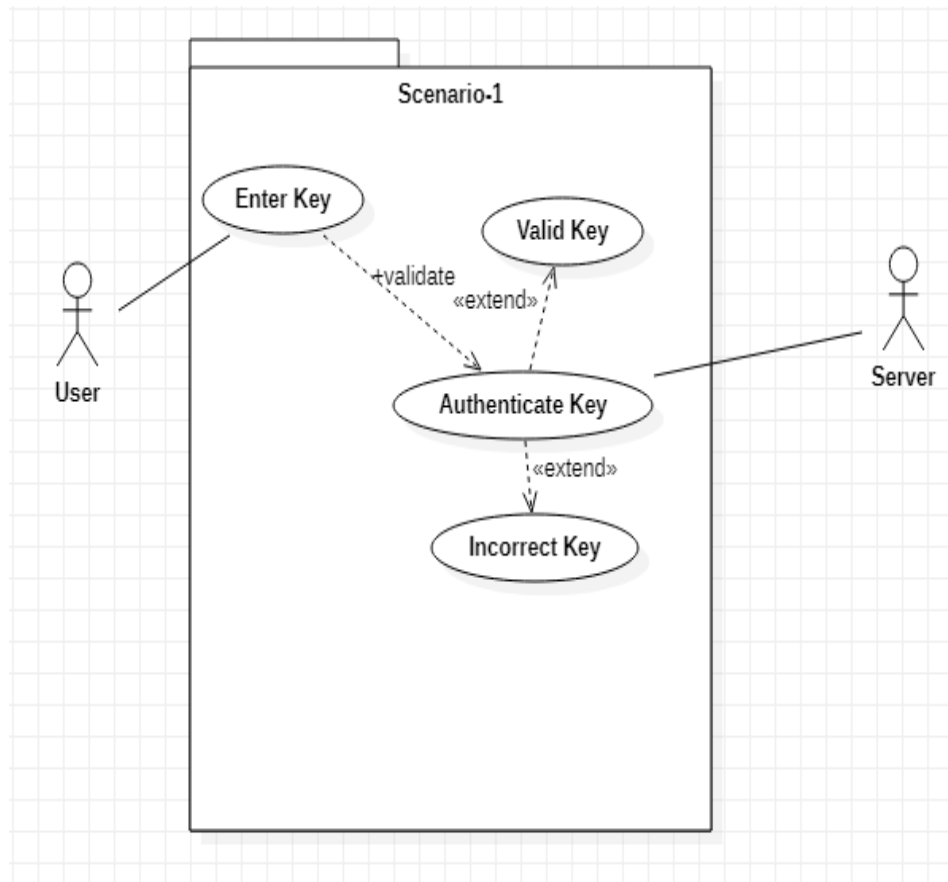
The following diagram shows the Client Server architecture between clients and server. Clients using application interfaces communicate with the server.



Use-Case Scenario

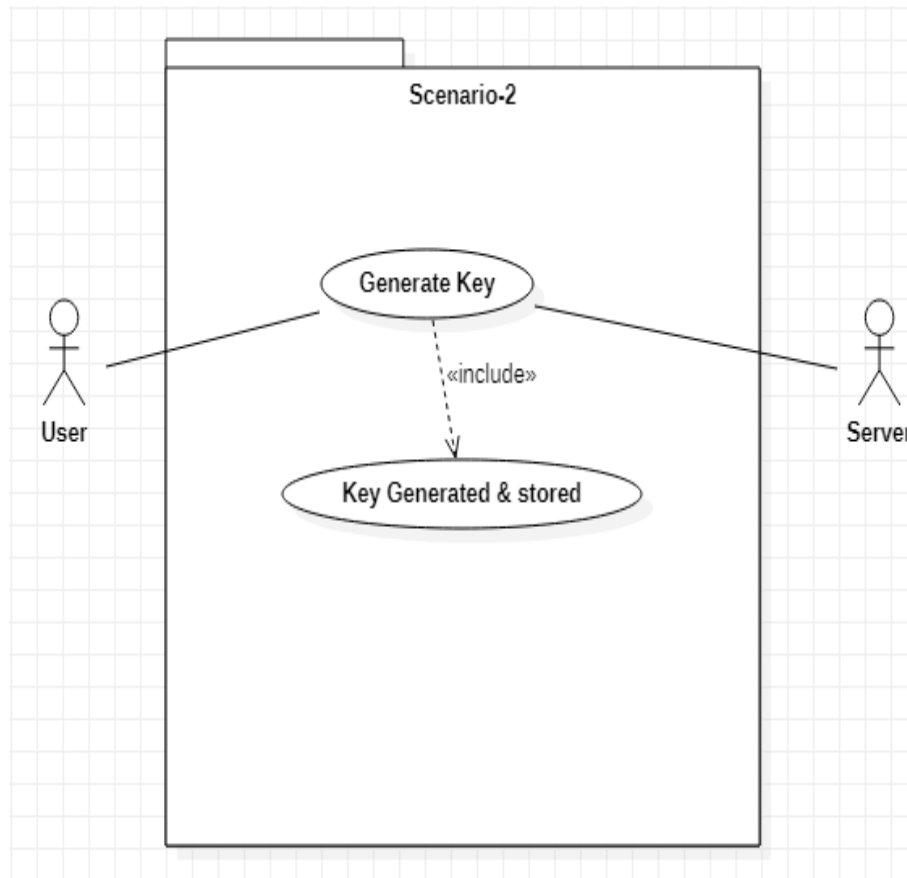
Scenario-1:

The user tries to enter the key to enter into the chat room. The key entered by the user is validated by the server. If the key exists in the server, the user is directed to the chat room or else the user is warned as an incorrect key.



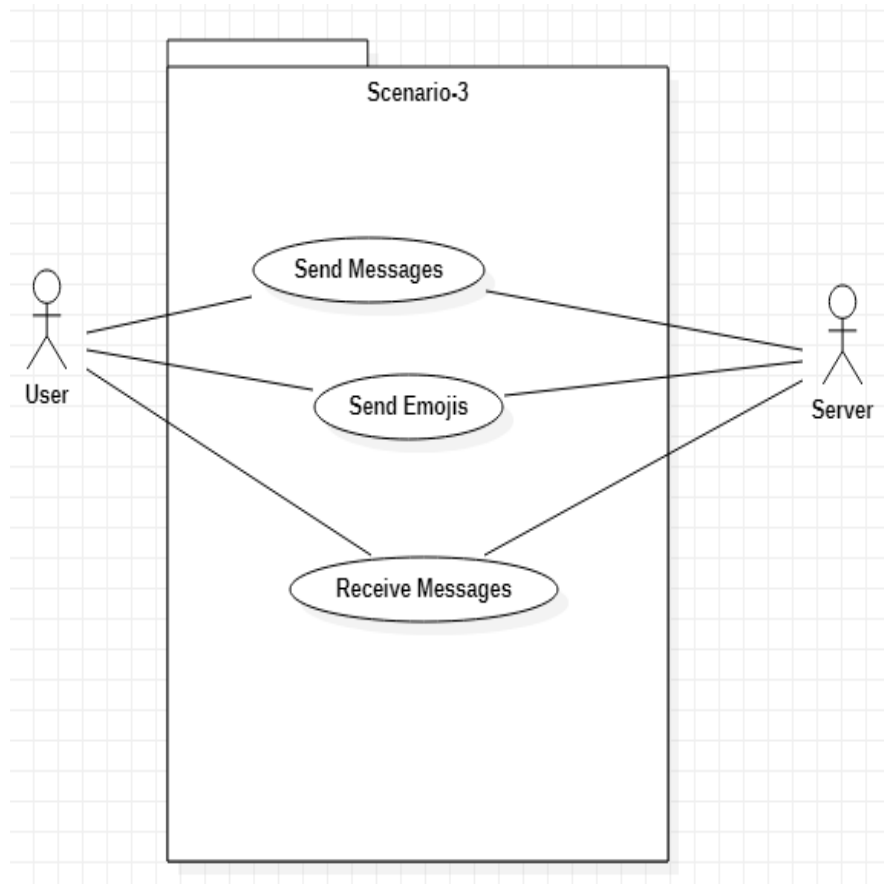
Scenario-2:

When a user wants to create a new chat room, he clicks on the generated key. A key will be generated and stored in the server. When other users enter the same key, it is verified in the server and they enter the chat room.



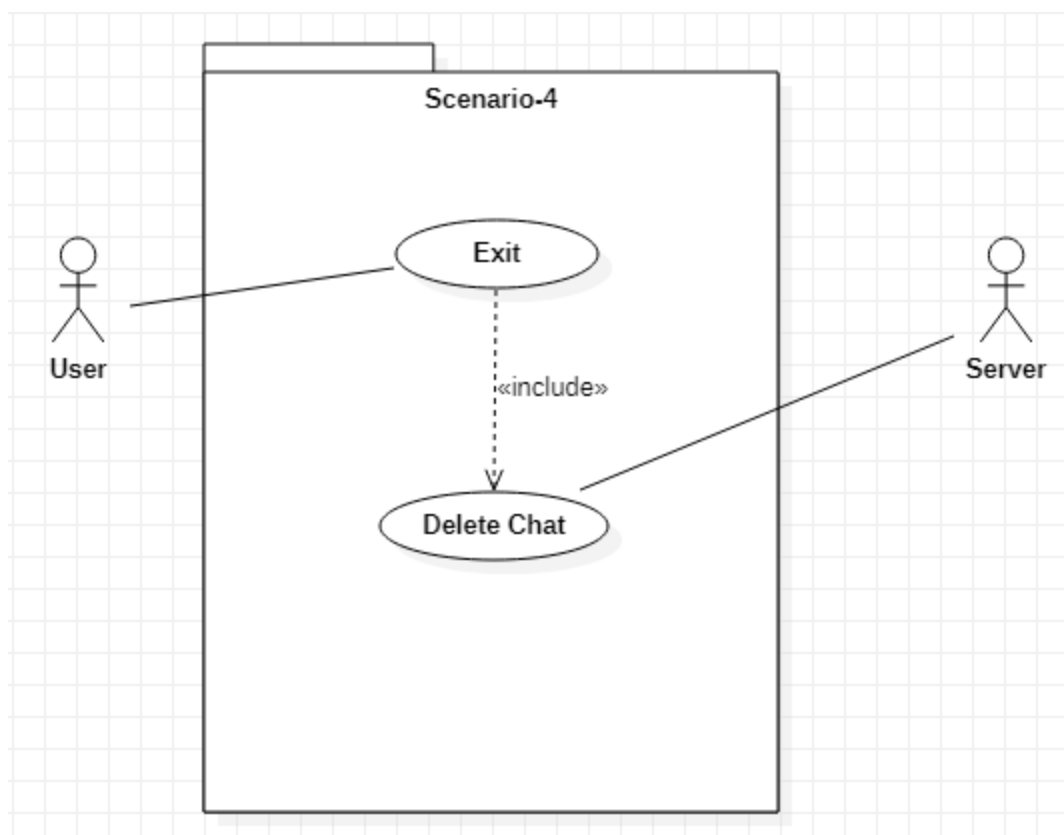
Scenario-3:

When the users enter the chat room, they can type their messages in the text area and send the message in the room. Whoever is present in the chat room, they will be able to see the message sent by a user. The users can also communicate using emojis.



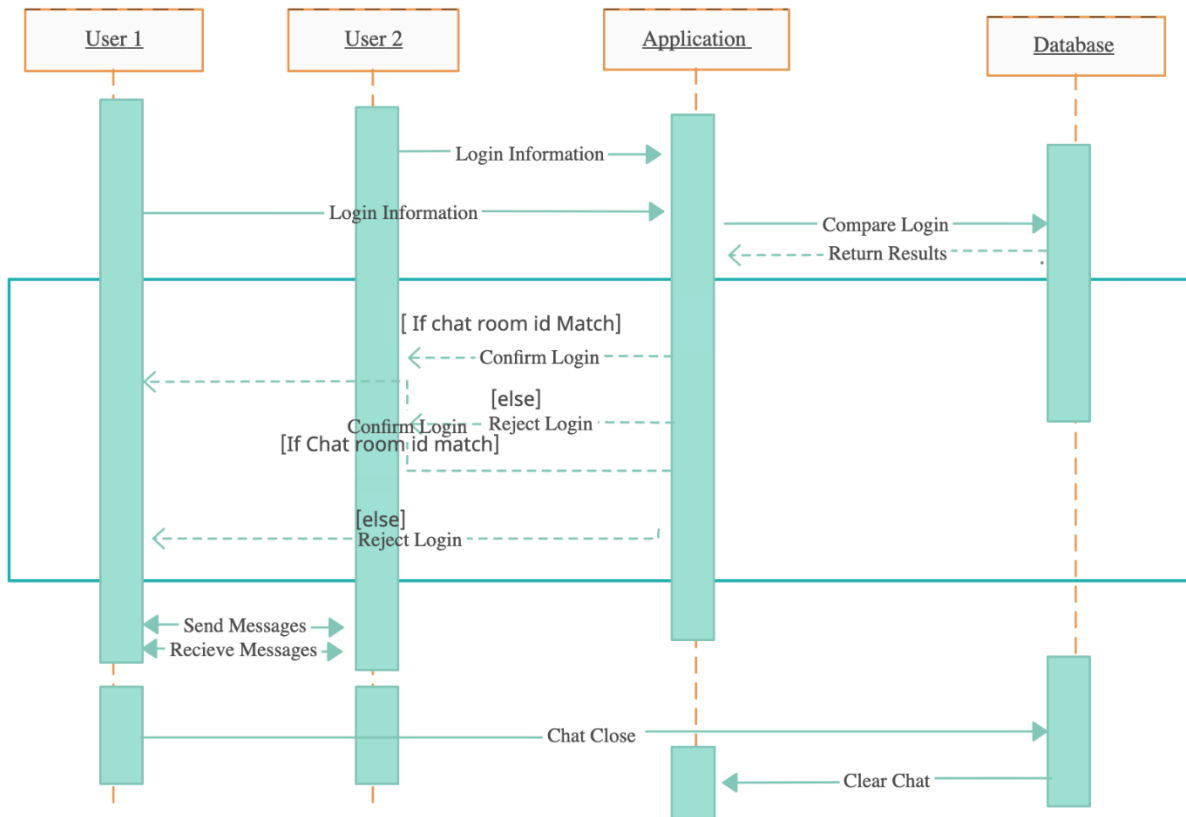
Scenario-4:

When all the users exit the chat room, the chat is cleared from the server.



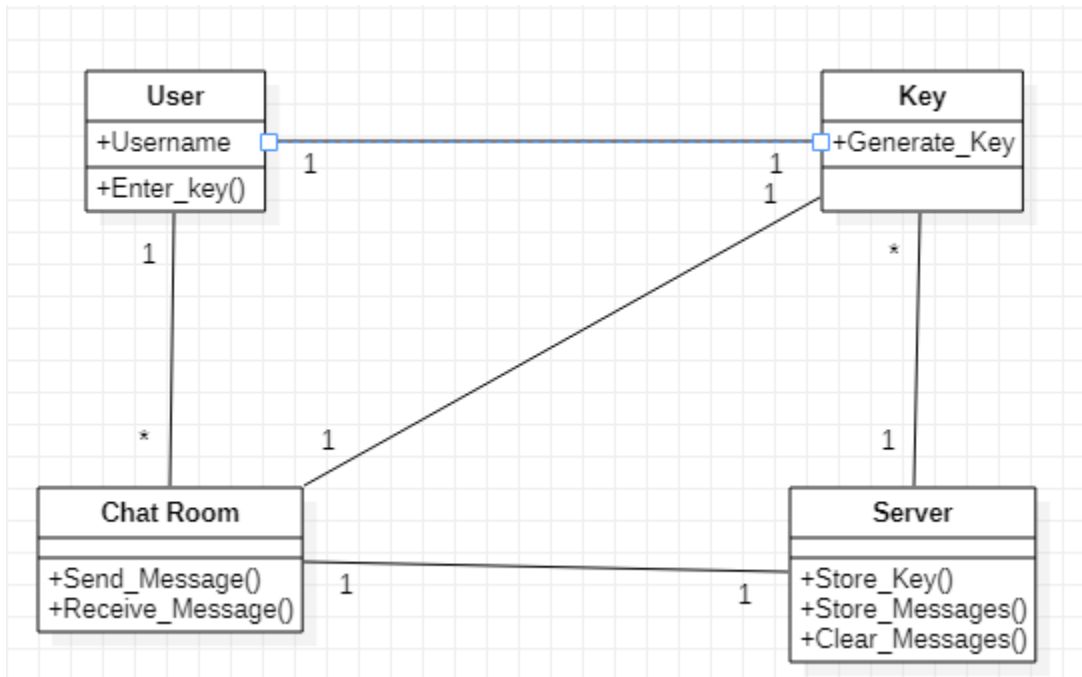
Sequence Diagram

The following figure shows the Sequence diagram of users, application and Database. Users enter the login information in Application the Application checks the details in the database if the chat room id matches then the user is allowed into the chat room. once the users leave the chat room then the messages are deleted automatically.



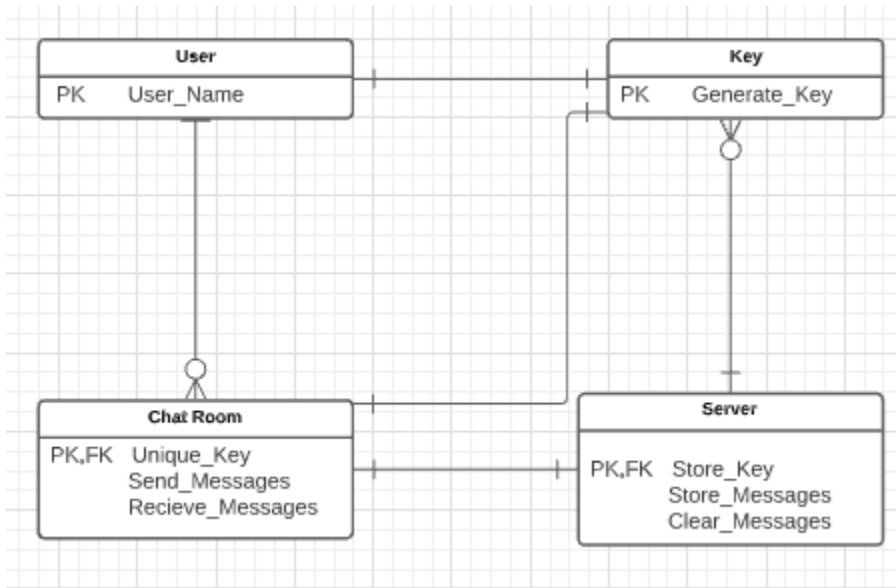
Class Diagram

In the Class diagram there are 4 different classes User, Key, Chat Room and Server. The following diagram shows the relation between classes.



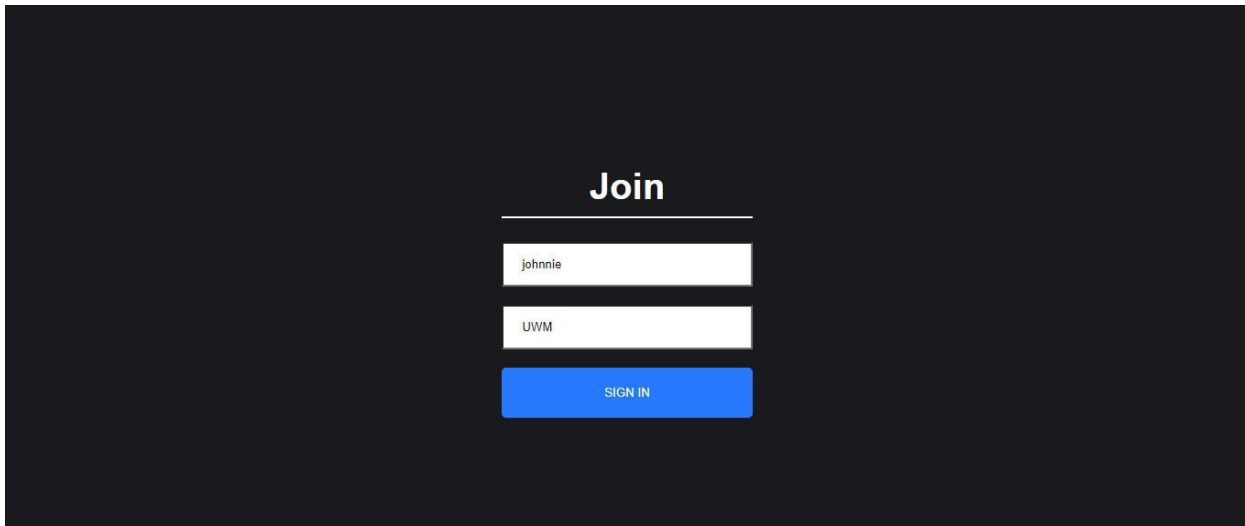
Relational Database Design

The following diagram shows the Relationship design of the database



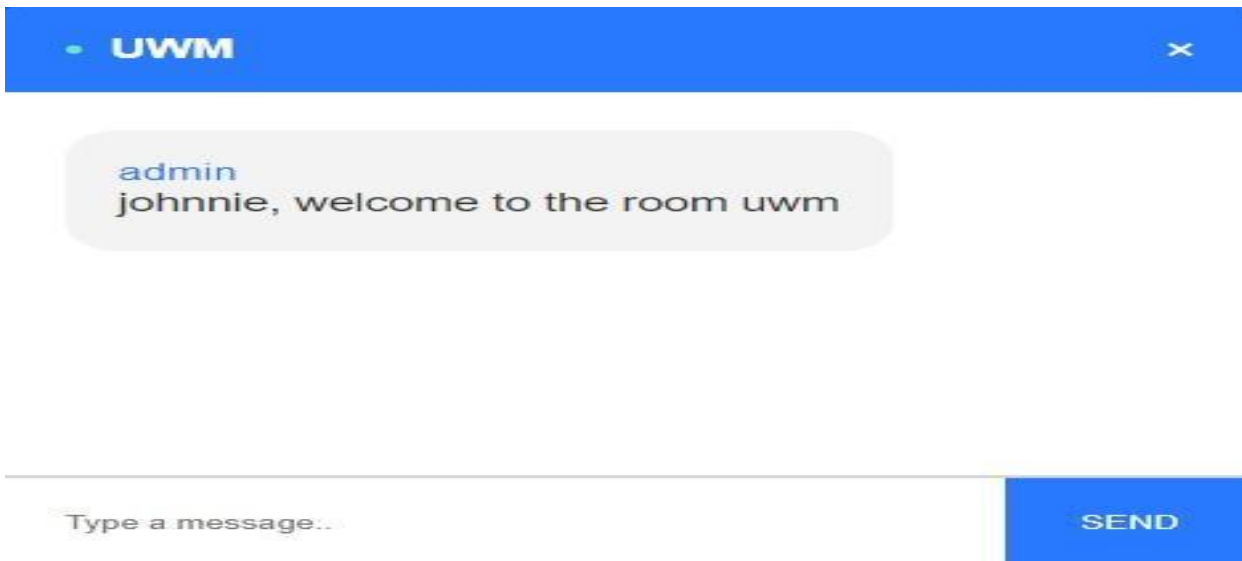
User- Interface Design

- First User enters chat screen and enters name and chat room id



A dark-themed 'Join' screen for a chat application. The title 'Join' is centered at the top. Below it are two input fields: the first contains the name 'johnnie' and the second contains the room ID 'UWM'. A blue 'SIGN IN' button is positioned below the input fields.

- The Users get into the chat screen and once the user leaves the chat room then all the messages get deleted automatically.



A chat room interface with a blue header bar displaying 'UWM' and a close button. A message from 'admin' says 'johnnie, welcome to the room uwm'. At the bottom is a text input field with the placeholder 'Type a message...' and a blue 'SEND' button.

Goals and Exit Criteria

Quality Goals that need to be met for the test phase to exit.

Quality goals of testing include Written test cases for requirements: In this phase the test cases are written for requirements that are to be met. In this project, the application the following are the expected requirements to work as expected.

1. Proper Functionality and response from the server.
2. Saving the user's id.
3. Authorizing the user with respective chat room names.
4. The text messages should be displayed as soon as they are sent.

The other quality is of written test cases that are successfully executed: Here the test cases that are written to test whether the application is working as expected without any errors and maximum efficiency.

Finally, quality of the application will also depend on the bugs that are yet to be verified, when discovered in the final phase of testing before releasing the product or at the time of user testing.

Robustness goals of the product.

The application depends on a server in order to store the information of the user and the messages that are being sent from one another, hence the application must be robust in doing the following tasks. The application should not halt anytime when being used. It must make sure that all the text messages are displayed properly and based on the time they are sent so that the messages are in chronological order and not in a random fashion. The application is expected to handle multiple users using it at the same time.

Schedule goals of the project

Project Phase	Phase Goals
Planning of project	A detailed plan on the project process
Design of application	Design of the UI and ML model architecture
Phase 1: Development of application	Initial development of Application
Phase 1: Testing the product	Testing of initial development
Phase 2: Development of application	Resolving and addressing remaining issues in the application that have been left out in the first development phase.
Phase 2: Testing	Testing the application.
Final product implementation	Final changes, improvements and deployment of the application

The schedule of the project can be depicted as different phases, the following is a table showing abstract information of Software Development Life Cycle of the project.

The software is expected to be completed within the mentioned time and schedule, considering any minor bugs and errors, with taking necessary measures to rectify and correct them.

Performance and efficiency goals of the product

The key aspects of performance and efficiency of the product being developed are as follows:

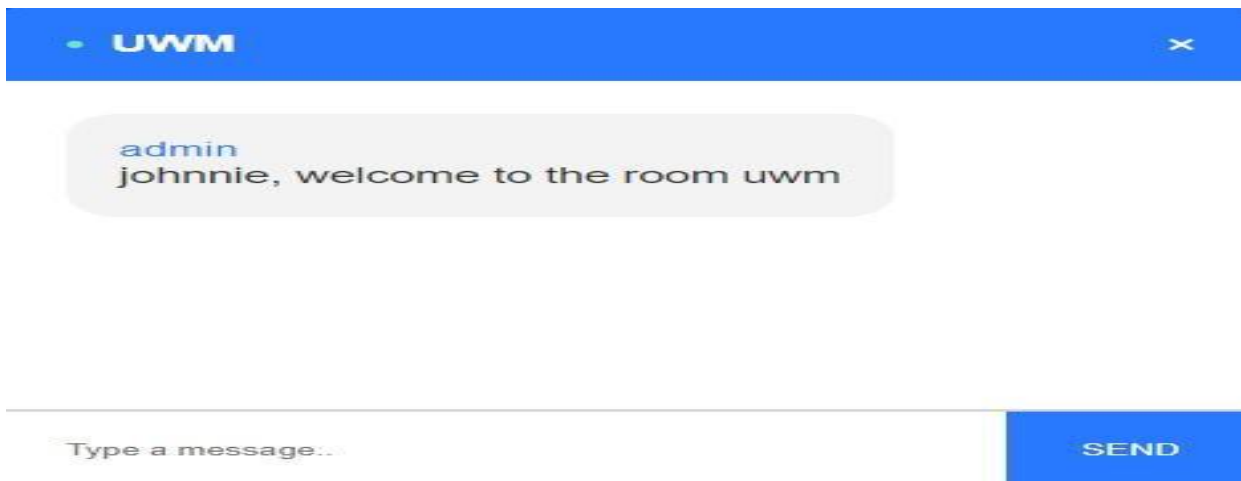
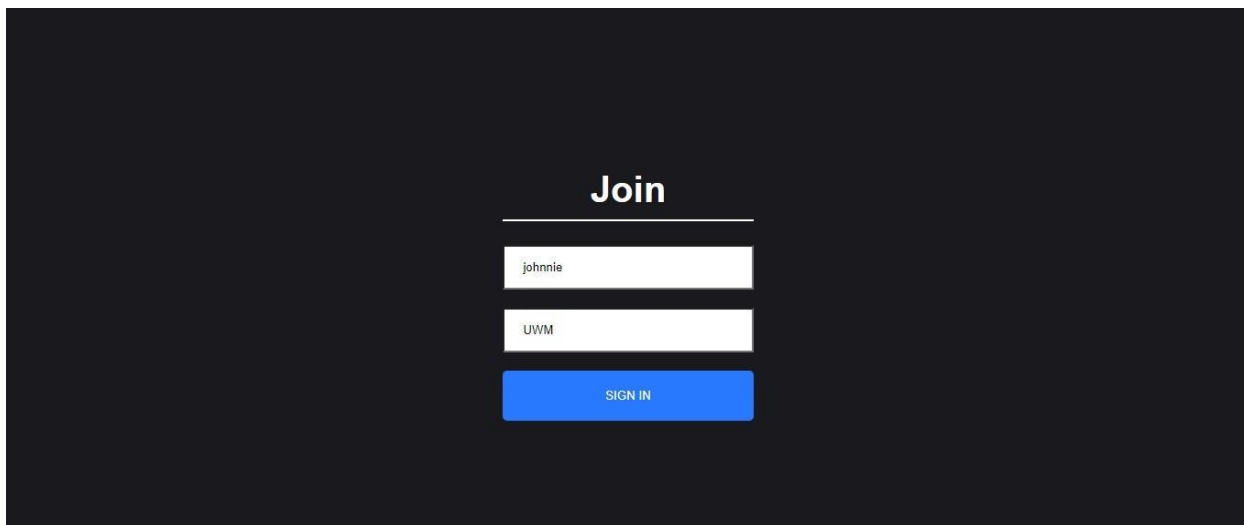
As this software is for texting and sending information through a medium, it must do the assigned task quickly, there must not be any lag between the user and the server. The messages that are being sent must be displayed synchronously, without any delay. The information should not be misplaced or lost. The messages should not be grouped and be sent at once from a single user that has been sent over time. It must be highly responsive and should exhibit high performance. Efficiency goals of this application is that it should be able to handle multiple or a greater number of users using a smaller number of resources. It must be able to handle a large number of messages that are being exchanged between the users very efficiently.

Items to be Tested/ Inspected:

Initially, we check whether the key is generated and stored in the server. When a user wants to join a chat room, he enters the room key. If the key already exists, then the user is added to the chat room. We check for the name of the user whether it is saved correctly or not. In the chat room, we check whether the messages sent are

displayed to every user and the emoji's whether they are displayed in graphic representation or in symbols. We also investigated the server issues. We checked if the server is getting timed out and fixed it.

User Interface:



Test Process/ Methodologies:

Unit Testing

In this type of testing, individual modules of the software are tested. Its main goal is to verify each module is working as per the requirements. Unit testing is done during the development phase of each module.

Inspection

The testing, when done during the development phase, is verified by other team members. They go through the module's execution and verify whether it is working as per the requirements and look for any bugs which might affect during the integration of the modules.

Black Box Testing

It is also called Behavioral/Specification-Based/Input-Output Testing. Black Box Testing is a software testing method in which testers evaluate the functionality of the software under test without looking at the internal code structure. When the user enters an existing key, he is redirected to the chat room. If the key entered is not available in the server, then a new chat room is created, and the user is considered the admin.

White Box Testing

It is also called Glass Box, Clear Box, and Structural Testing. White Box Testing is based on the application's internal code structure. In white-box testing, an internal perspective of the system, as well as programming skills, is used to design test cases. This testing is usually done at the unit level. This type of testing is done when the tester has knowledge regarding the code structure and designs test cases based on his knowledge.

Test Metrics

Code coverage is the percentage of code which is covered by automated tests. Code coverage measurement simply determines which statements in a body of code have been executed through a test run, and which statements have not. The number of bugs or errors occurred is also one of the test metrics. In our project we were able to identify two errors with respect to text boxes which were not accepting numbers, then made necessary changes to the code to fix the problem.

Test-bug report-fix-retest process

We have used a spreadsheet with multiple access to report the bugs that occurred. They are reported so that all the team members can view them and perform the necessary action. We made necessary changes to the code and to the UI design in order to fix the problems and performed testing again, to check if the issue was resolved. We followed this process after every development process until all the errors that have been identified and reported were solved and obtained the results that were expected from the application.

Resources

People

This project is being developed by a group of three people having knowledge and skills in Java, JavaScript, ReactJS, NodeJS, Express, and CSS. Basic knowledge of software testing is required to perform testing to rectify bugs and errors.

Tools

We will be using GitHub as our management tool as it can store all the previous versions of the software that will help in management, testing and developing as it supports version control. Microsoft visual studio will also be used as it supports a vast number of programming languages and allows debugging which is user friendly.

Systems

For test case development we will be using multi access spreadsheets which can be accessed by all the group members to develop and implement the test cases. Subsequent changes and updates can be reported into these spreadsheets.

Schedule:

Test-case development:

The testing for this project can be performed in multiple phases as development is done in multiple phases. The testing can be carried after each development phase has completed to highlight any bugs or errors occurred after the intimal or intermediate development. The team members will create multiple test cases depending on their knowledge and what they have developed for the application, which are updated to a spreadsheet that can be accessed by all the team members and can perform those test cases. Development of a good number of test cases that are efficient in testing the software will result in correctness of the product with less time and budget. For this multiple test cases can be developed for example: Response from the server, messages that are being received and sent, short and very long messages should be equally important and be given the same priority. All the text information should be sent spontaneously when sent by the user. The emojis entered by the user should be displayed in graphical format rather than symbols.

Test Execution:

All the test cases that are developed must be executed in order to obtain good results. There might be a few bugs and errors that might be missed if these test cases are ignored. After implementing the test cases, errors and bugs need to be addressed and then perform testing again to measure their success.

Problem reporting and fixing:

All the bugs and errors that have been discovered must be informed to other team members. They will be uploaded into the test case development spreadsheet which can be accessed by all the team developers. The member who has developed the specific functionality, in which an error or bug has occurred, will be notified and will be asked to solve it with the help of other team members, by considering all the inputs and selecting the optimal solution.

Risks

Missing Goals: The product is working as expected and satisfies the requirements that were established during the initial stage of the development.

Backup Resources Needed: As the product is a web application, we have made a copy of the application on our personal computers, and on the cloud storage so that it can be deployed easily if any loss happens.

Major Test Scenarios and Test Cases

Boundary value and input domain test cases:

BVT is used to test the boundaries between the equivalence partitions. BVT is most appropriate where the input is a continuous range of values, simply because this is where so many defects exist.

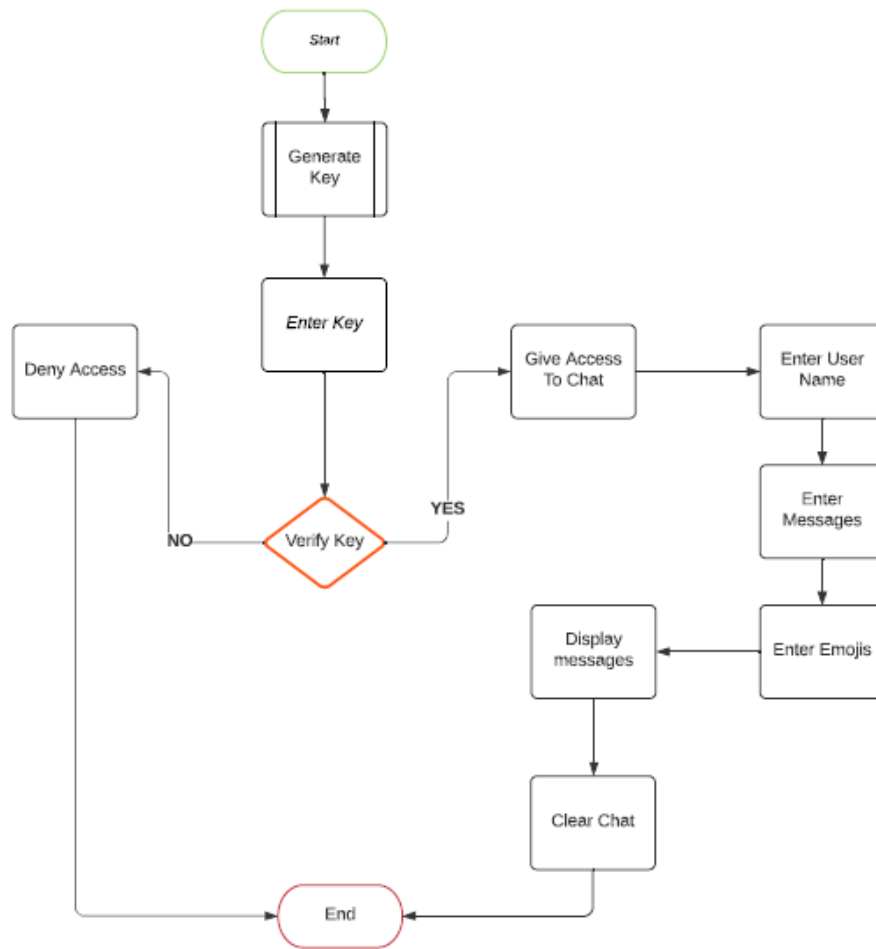
Test Scenario

Consider the following situation where a text box which is used to enter the room key allows a character to be entered:

If the user A enters @UWM to create a chat room, then @UWM becomes the unique key which will be used by other users to join that chat room.

If another user B enters @UWm in the text box (room key) then they will not be able to join the chat room created by User A as there is a mismatch of a character “m” as they are case sensitive. Case sensitivity completely depends on the preference of the client, but it will allow more unique key or chat room generations.

Control path and data flow test cases



Control path and data flow test cases will allow us to check the flow of control and data flow which make sure that the data is provided to the next task correctly.

Test cases

1. Consider the following test case where the chat room key that has been created must be verified before giving access to the other user.
2. The messages that are being entered by the user should only be displayed after pressing the enter key, which can be made sure by using action listener, which will check if the user has pressed enter then only the message or text will be sent through the server and then it will be visible to the other users that are present in that chat room.

Integration and intermodular test cases:

Integration testing will make sure that the different modules of the software application are integrated correctly so that there are no gaps between the modules and can perform as a single system. Modularity will allow the developers to bind multiple functionalities of the system that can be developed by multiple people. In our project we made sure that the User Interface is perfectly integrated with the backend system(server) so that there are no delays or errors while communicating or data transfer.