**Problem Statement:-**

->Email, newsgroup and messaging applications provide means for communication among people but these are one-way mechanisms and

they do not provide an easy way to carry on a real-time conversation or discussion with people involved.

->Suppose you have to use a chat application where multiple users can chat together.

->Rather than each user sending the message directly to other users,we can use mediator pattern to implement this design.

->The problem with the internet messaging is Data loss or viewed by third party while transferring data or infromation.

->Some untrusted staff will denial the task that given to them,take time to receive the data or information through internet.

->Chat room extends the one-way messaging concept to accommodate multi-way communication among a set of people.

->Chat rooms have become a popular way to support a forum for n-way conversation or discussion among a set of people with

interest in a common topic.

->Chat applications range from simple, text-based ones to entire virtual worlds with exotic graphics.

**Solution:-**

->In this project we are implementing a simple text-based chat client/server application.

->This project is to create a chat application with a server and users to unable the users to chat with each other.

->To develop an instant messaging solution to enable users to seamlessly communicate with each other.

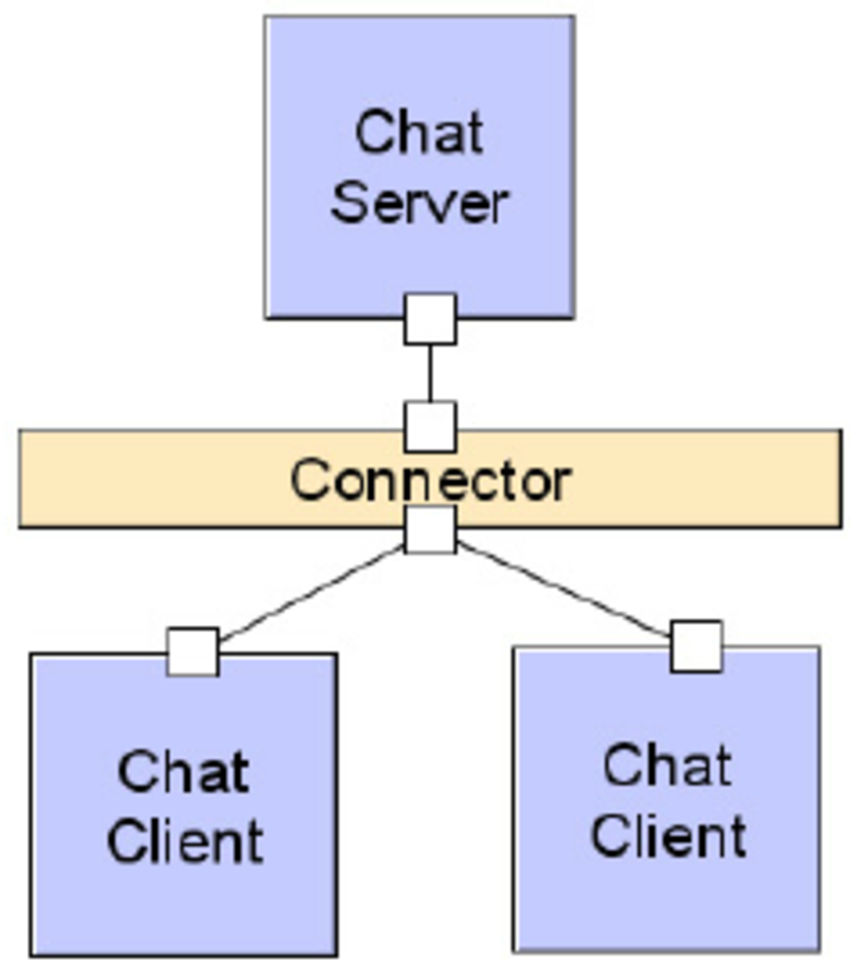
->The chat application we are going to make will be more like a chat room, rather than a peer to peer chat.

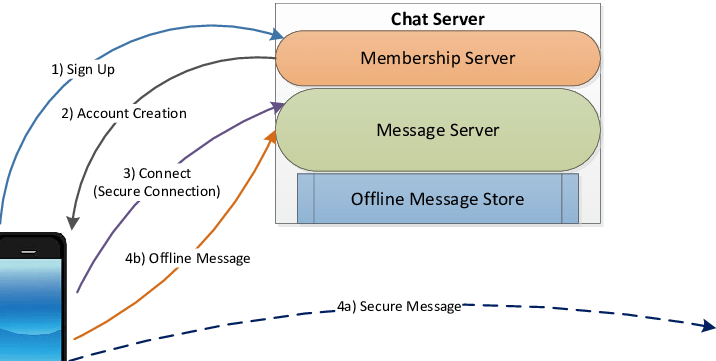
->So this means that multiple users can connect to the chat server and send their messages.

->Every message is broadcasted to every connected chat user

->This project is very easy to use enabling even a novice person to use it.

**Software Layer Architecture:-**





**Tools and Methods:-**

->Chat program is like any other Client-server application so it would require all those tools/platforms that are required for a simple client-server application (eg. the most basic form of website hosted on a server and accessed by a thin Client or a thick client).

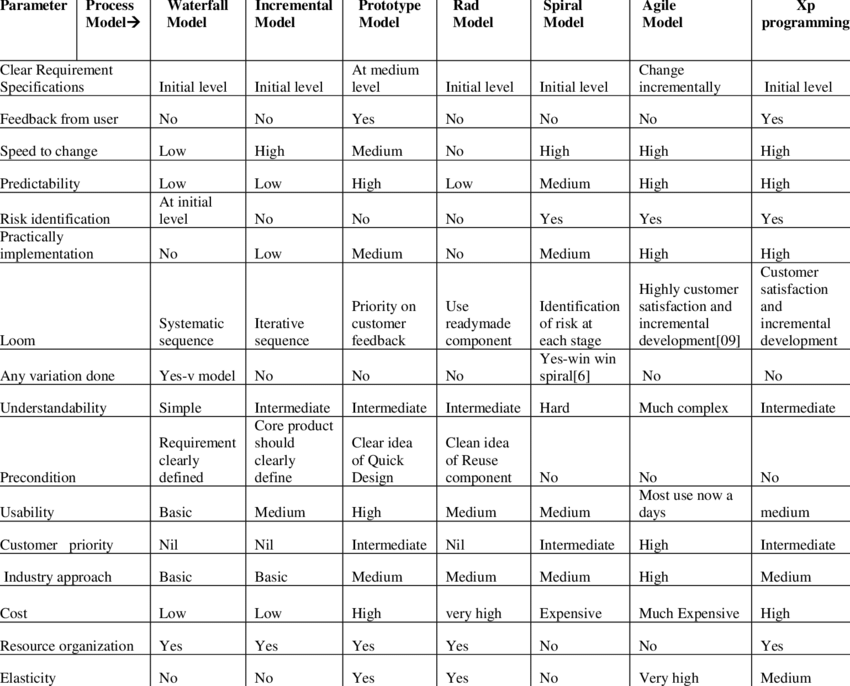
->If you have basic knowledge of any Programming language, then using the web based APIs of that language you can create a simple chat program.

->We will be using socket programming for the development of this software.We will create a chat program which did a trivial task of group chat.

->This means a user posts a message on his browser and that message is received by the server and relayed to all the other users accessing my chat server at same time.

->We will be using Python and its frameworks for developing this project.

**Comparative Study Of Software Process Model:-**



**Study Of Our Software Process Model:-**

In the system development, there are number of methodologies, known as SDLC (System Development Life Cycle), which can be applied such as Waterfall, Evolutionary Development and Component-Based Software Engineering. The selected-approach reflects the success of the system.

In the waterfall model, the system development process cascades from one phase to another.

It comprises six phases, namely:

1.Phase one : System Planning

2.Phase two : Problem Analysis

3.Phase three : System Design

4.Phase four : System Implementation

5.Phase five : System Testing

6.Phase six : Operation and Maintenance

In Figure 1 shows the waterfall sequential model from one phase to the next in a cascading method.It means the following phase shouldnot start until the previous phase has finished. For instance, one first completes systems planning, which are set in stone.

When the requirements are fully completed,one proceeds to problem analysis. Next, after the problem analyses are fully completed, it goes to system design. When the design is fully completed, a system implementation of that design is made by coders.

During the final life-cycle phase, the software is put into use. Errors and omissions are discovered.Therefore, the waterfall model maintains that one should move to a phase only when it’s proceeding phase is completed and perfected.

**COMPARITIVE STUDY:-**

**1.Socket programming:-**

Sockets can be thought of as endpoints in a communication channel that is bi-directional, and establishes communication between a server and one or more clients. Here, we set up a socket on each end and allow a client to interact with other clients via the server.

The socket on the server side associates itself with some hardware port on the server side. Any client that has a socket associated with the same port can communicate with the server socket.

**2.Multi-Threading:-**

A thread is sub process that runs a set of commands individually of any other thread. So, every time a user connects to the server,a separate thread is created for that user and communication from server to client takes place along individual threads based on socket objects created for the sake of identity of each client.

We will require two scripts to establish this chat room. One to keep the serving running, and another that every client should run in order to connect to the server.

**3.Server Side Script:-**

The server side script will attempt to establish a socket and bind it to an IP address and port specified by the user (windows users might have to make an exception for the specified port number in their firewall settings, or can rather use a port that is already open). The script will then stay open and receive connection requests, and will append respective socket objects to a list to keep track of active connections. Every time a user connects,a separate thread will be created for that user.

In each thread, the server awaits a message, and sends that message to other users currently on the chat. If the server encounters an error while trying to receive a message from a particular thread, it will exit that thread.

**FUNCTIONAL REQUIREMENTS:-**

**1. USER REGISTRATION**

->User must be able to register to the application through a valid phone number.

->On starting the application user must be prompted to register their phone numbers.

**2.ADDING NEW CONTACTS**

->The user must be able to add the contacts to have live chat.

**3. SEND MESSAGE**

-> User must be able to send messages to the other people in contact.

->User must be notified the delivery of the message.

**4. MESSAGE STATUS**

->User must be notified about the delivery of message,reading of the message(whether the message has been read by the people in contact or not).

**NON FUNCTIONAL REQUIREMENTS**

**1. PRIVACY**

->Messages shared between the users must be encrypted so that privacy is maintained.

**2.ROBUSTNESS**

->In case useres devive crashes, there should be a backup and a chat history.

**3. PERFORMANCE**

->Application must be light weight and the messages should be deliverd instantly

**Use Case Diagram:-**

