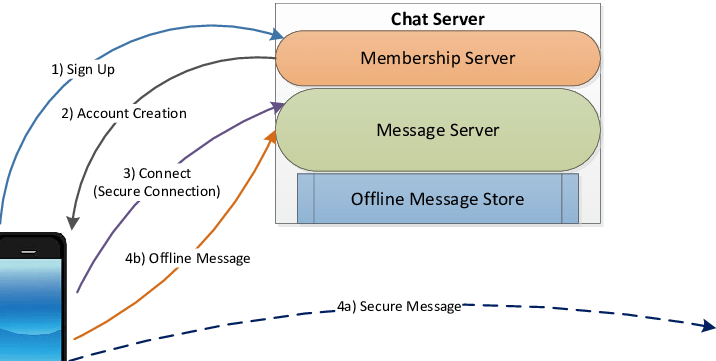
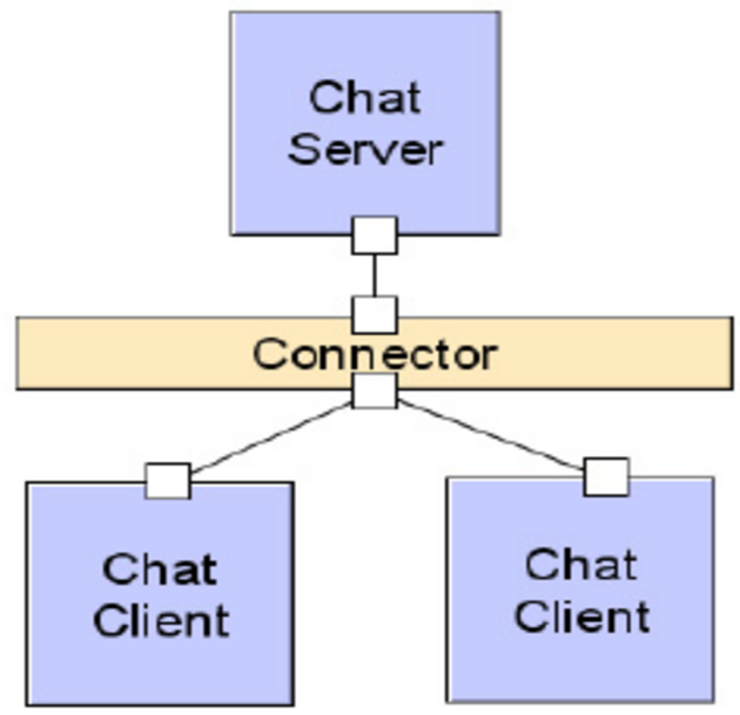
**CHAT BOX**

**Group members: 1. Durgesh Sharma (0827CS181072)**

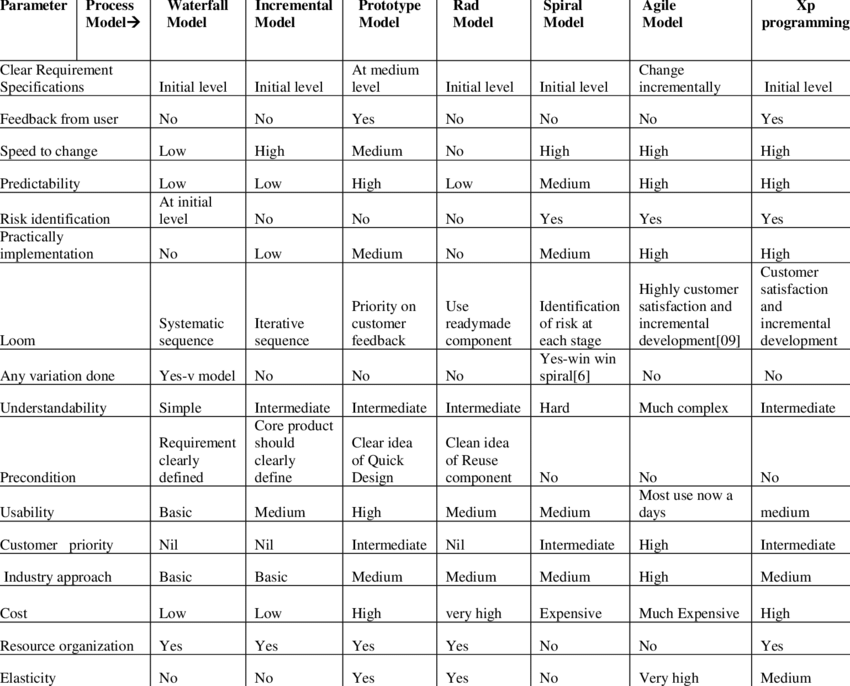
**2. Kartavya Verma (0827CS181103)**

**3. Harshala Gaikwad (0827CS181087)**

* **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 Agile 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

It shows the waterfall agile model from one phase to the next in a cascading method. It means the following phase should not 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 Agile 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**

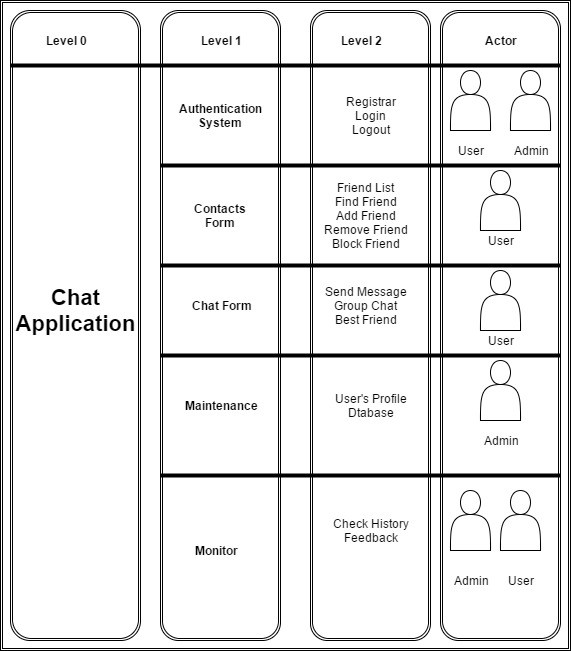
* + ->Messages shared between the users must be encrypted so that privacy is maintained.

**2.ROBUSTNESS**

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

**3. PERFORMANCE**

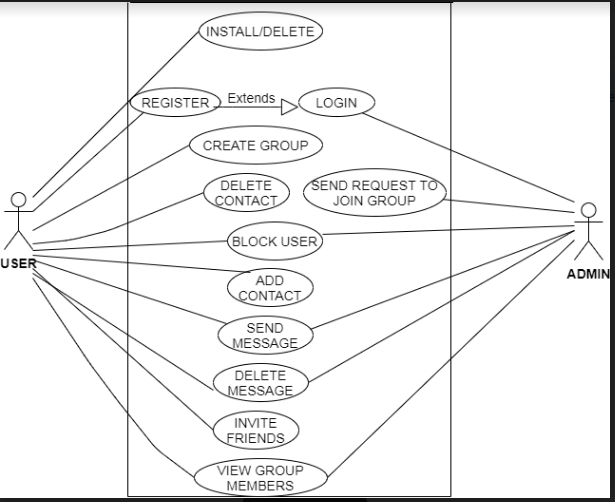
* + ->Application must be light weight and the messages should be delivered instantly.



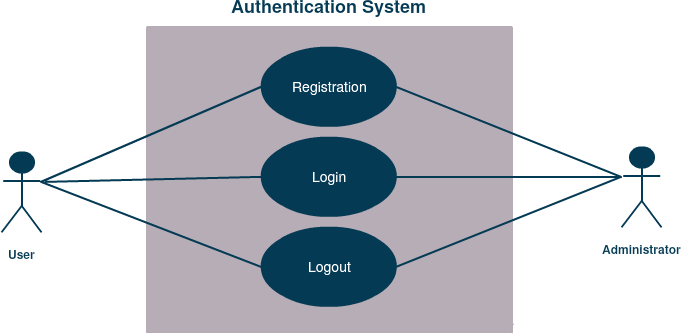
**Use Case Table:-**

Figure 1: Use Case Table of Chat Application

**USE CASE DIAGRAM:-**



**Authentication System:-**

 Figure 2:Use Case Diagram Of Authentication System

### Contacts Form:-

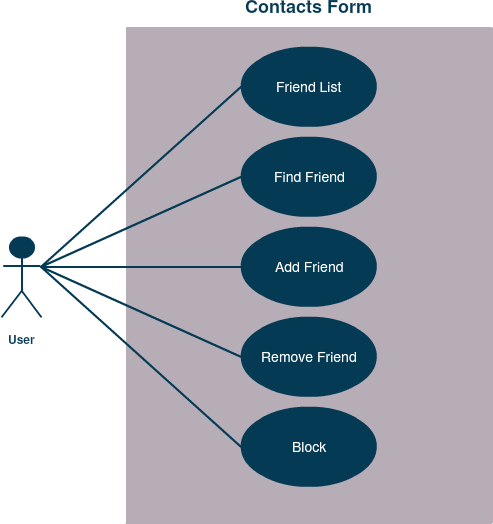


Figure 3:Use Case Diagram Of Contact Form

### Chat Form:-

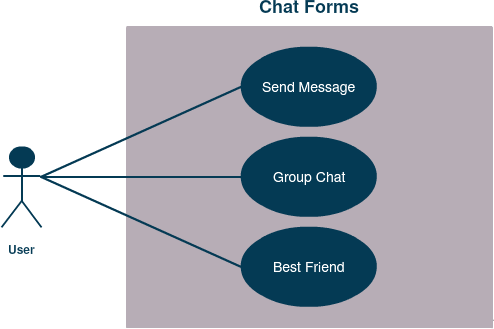


Figure 4:Use Case Diagram Of Chat Forms

**Maintenance:-**

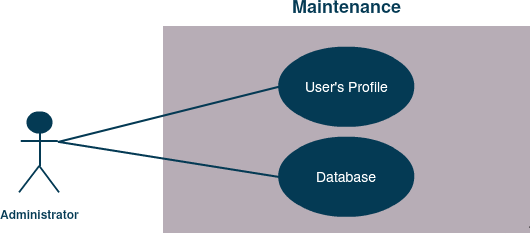


Figure 5:Use Case Diagram Of Maintenance

## Activity Diagram:-

### 1.Authentication System, Maintenance, Monitor:-

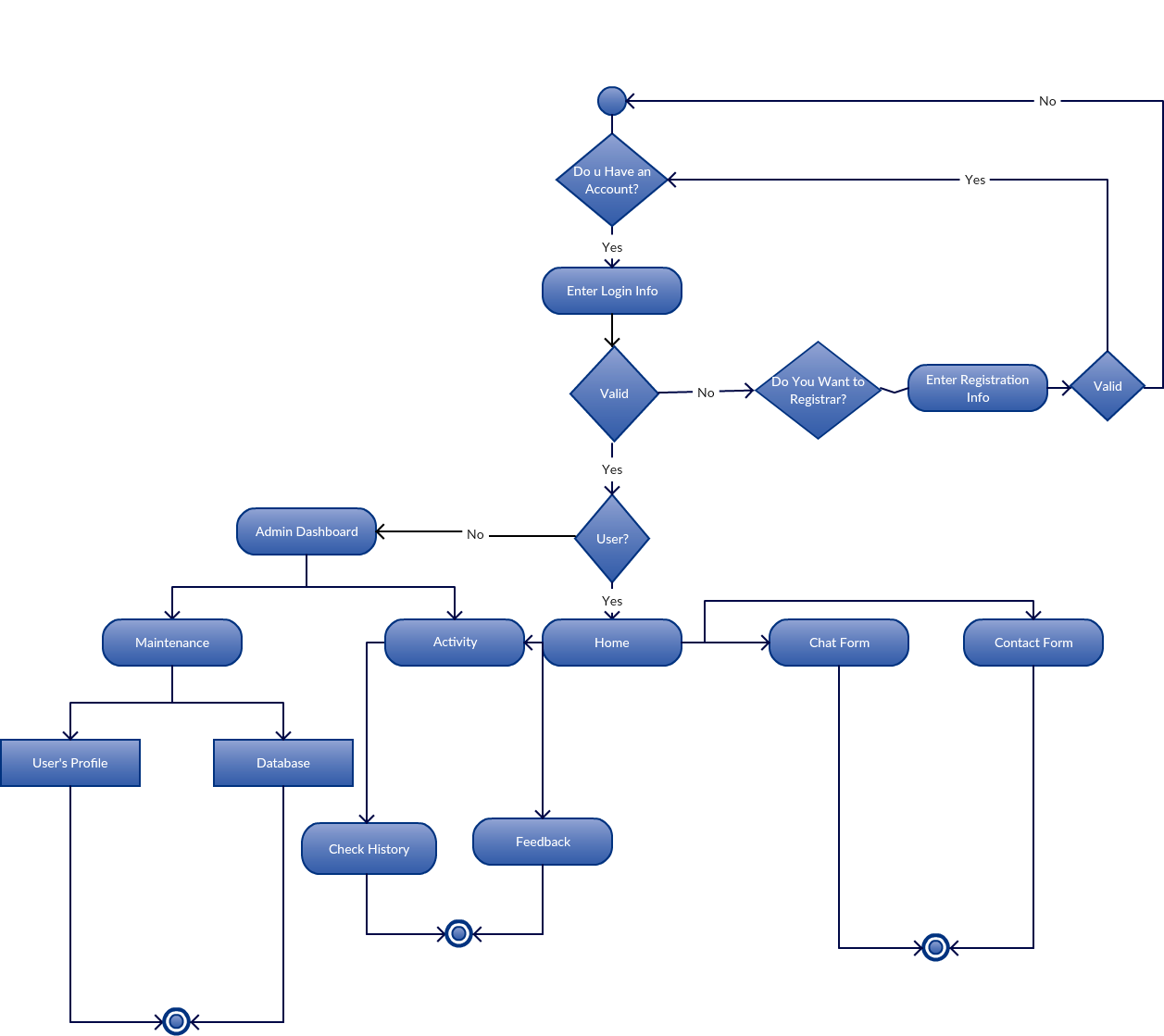


Figure 6: Activity Diagram of Authentication System, Maintenance, Monitor

### 2.Contacts Form, Chat Form:-

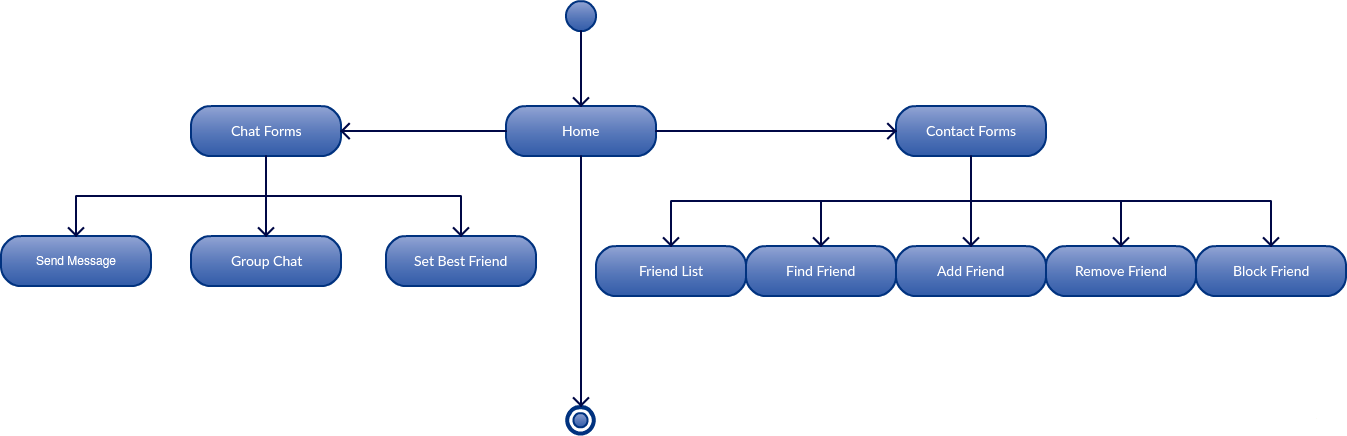


Figure 7: Activity Diagram of Contacts Form, Chat Form

## Entity Relationship Diagram:-

## 

Figure 8: Entity Relationship Diagram of Chat Application

## Data Flow Diagram:-



Figure 9: Data Flow Diagram of Chat Application

**Implementation:-**

Generally, in real-world communication is done directly using voice in an ideal situation (distance between communicating parties, identification of parties)  where **sender** speak out and the intended **receiver** respond after listening. So, what is the **medium of communication**here? Indeed, it is the air which helps our voice to travel to the **receiver** and successful communication depends upon air (high wind and long-distance can cause trouble). In online or digital communication the role of air is played by network channel (coaxial cable, fiber optics, etc.) and communication is controlled by a **server**. A **server** is a program which regulates the communication between **sender** and **receiver**.

So, to create a **Python** **Chat Application**, one has to write a **server** program and **client** program/s (**sender** and **receiver**). Suppose, two parties Alice and Bob want to chat with each other and ask you to develop a chat application then being a developer you have to write a **server** **program** and a **client** **program** (different instance of the same program will be used by both Alice and Bob or even more users).

we will demonstrate the aforementioned scenario and will develop a **Python** **Chat Application**for Alice and Bob. **Python** has many modules which can help us to create network-related application, the **socket** is one of such popular default **Python** modules for **low-level network programming**. We will first list and explain the steps for **server** and **client** programs and then implement the same using **Python**.

## ****SERVER SCRIPT:-****

**Server program** has all the logic to control and regulate the **Chat**, so most of the chat logic is implemented with a **server** program. So first step of communication is to identify the users, how server do this? In network communication, users are identified by a **socket** which is nothing but a combination of **IP address** and **port address**. So, for human understanding, Alice and Bob will be chatting but for a **network**, it is two **sockets** process which is sending and receiving bytes.  Steps involved in this process is as follows:

1. Create socket
2. Communicate the socket address
3. Keep waiting for an incoming connection request/s
4. Connect to client
5. Receive the message
6. Decode the destination user and select the socket
7. Send a message to the intended client
8. Keep repeating step 5 & 6 as per users wish
9. Exit i.e. end the communication by terminating the connection

**Client script:-**

**Client** script is run by the user, so the same **client** code will be run by a different user but each will have a separate **socket** so they will have their unique **communication channel**. Client script uses to be thin because it has very less work i.e. it only connect with the **server** and send and receive messages. The steps involved in **client script** are:

1. Create a unique client socket per instance/user
2. Connect to the server with given socket address (IP and port)
3. Send and receive messages
4. Repeat step 3 as per configuration
5. Close the connection

## Testing:-

**Test Plan:-**

### Chat Application will be the interpreter to bring people and ideas together. We have been designing our Chat Application with well- equipped technology. This project is now at development phase, so readers can read the Software Requirement Specification document for details. This document presents the Master Test Plan of IIT Website. As we know, master test plan is a living and breathing document that summarizes the overall effort required to test a software product. Master test plan will actually contain the details of individual tests to be run during the testing cycle like unit test, system test, beta test etc. However, our document will categorize and describe each test case. It will also outline pass-fail criteria and indicate the planned run day or week. This is a quick-reference tracking document for what has to be tested, the priority of test items, what is left to test etc. We followed IEEE-829 format to develop our test plan. We strictly follow the instructions provided by our respective course teacher. This is our first test plan documentation, so we also read some sample test plan to gather knowledge about test plan documentation. The estimated time line for this project is a semester. The testing activities are to be done in parallel with the development process.

### Test Items:-

Higher Level Test Items:-

Items to be tested:

1.Chat Application Released version 1.0 and supporting infrastructure

2.Application runnig on different client’s device

Items Not To be Tested:-

1.SRS of Chat Application

2.User Manual of Chat Application

3.Already Exist Chat Application

4.Manual process related to the application

5.Any Legacy System

Lower Level Test Items:

Items to be tested:-

1.User Profile

2.Chatting

3.Group Chatting

4.Add Friend

5.Remove Friend

6.Block Friend

7.Unblock Friend

8.Find Friend

9.Registrar

10.Log In

11.Log Out

Items not to be tested:

1.User

2.Admin

Some more modules were included and changed during development phase, which are not fully recognized or documented yet. So, test items of those modules will be included in the next version of the test plan.

### Software Risk Issues:-

There is several risk issues recognized which can have direct impact on the Chat Application and need to be handled carefully.

1.Database security and safety

2.Backup of files

3.Reliability of Hosting Service

4.Lack of test tools

5.Lack of training

6.Lack of User Involvement

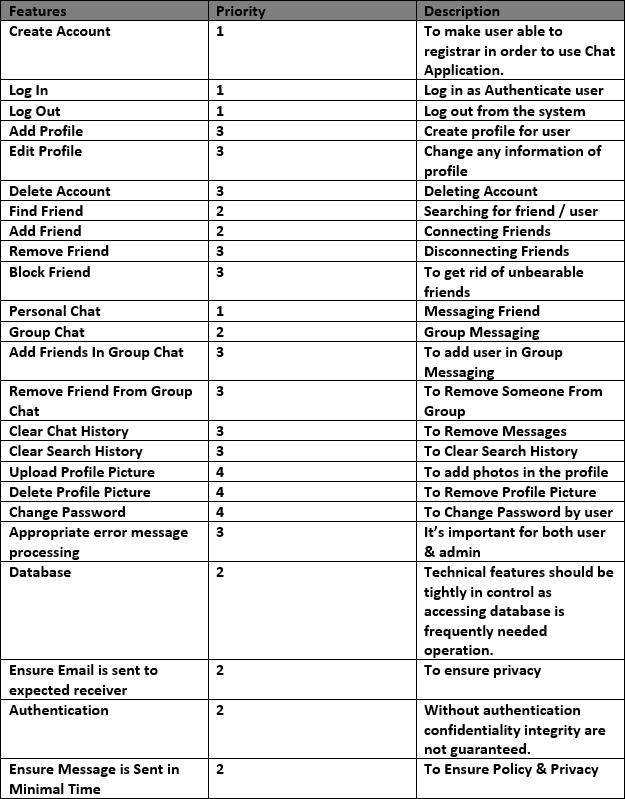
7.Lack of Schedule

8.Lack of Budget

9.Rapid Change in Taste

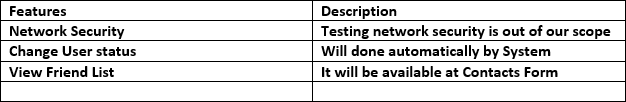
### Features To Be Tested:-

### The feature and attributes to be focused on during testing of the application.



### Features Not To Be Tested:-

We will try to test every features. But If we cant manage Schedule. Then we may skip those Features and attribute with low priority below:



### Testing Levels:-

The testing approach for Chat Application project is Master Test Plan or MTP. Which includes unit testing, integration testing and acceptance testing. In this project testing is done by developer, tester and user. Unit testing will be done by developers. Developer must provide unit testing data to tester. Integration testing will be done by tester. After all major and critical defects are solved it will go through acceptance test which will done by end user.

**Software Maintenance:-**

Software Maintenance is the process of modifying a software product after it has been delivered to the customer. The main purpose of software maintenance is to modify and update software application after delivery to correct faults and to improve performance.

**Need for Maintenance –**  
 Software Maintenance must be performed in order to:

1.Correct faults.

2.Improve the design.

3.Implement enhancements.

4.Interface with other systems.

5.Accommodate programs so that different hardware, software, system features, and telecommunications facilities can be used.

6.Migrate legacy software.

7.Retire software.

  There are number of reasons, why modifications are required, some of them are briefly mentioned below:-

**1.Market Conditions:-**

Policies, which changes over the time, such as taxation and newly introduced constraints like, how to maintain bookkeeping, may trigger need for modification.

**2.Client Requirements:-**

Over the time, customer may ask for new features or functions in the software.

**3.Host Modifications:-**

If any of the hardware and/or platform (such as operating system) of the target host changes, software changes are needed to keep adaptability.

**4.Organization Changes:-**

If there is any business level change at client end, such as reduction of organization strength, acquiring another company, organization venturing into new business, need to modify in the original software may arise.

## Cost of Maintenance:-

On an average, the cost of software maintenance is more than 50% of all SDLC phases.

****

There are various factors, which trigger maintenance cost go high, such as:

### **Real-world factors affecting Maintenance Cost:-**

1.The standard age of any software is considered up to 10 to 15 years.

2.Older softwares, which were meant to work on slow machines with less memory and storage capacity cannot keep themselves challenging against newly coming enhanced softwares on modern hardware.

3.As technology advances, it becomes costly to maintain old software.

4.Most maintenance engineers are newbie and use trial and error method to rectify problem.

5.Often, changes made can easily hurt the original structure of the software, making it hard for any subsequent changes.

6.Changes are often left undocumented which may cause more conflicts in future.

### Software-end factors affecting Maintenance Cost:-

1.Structure of Software Program

2.Programming Language

3.Dependence on external environment

4.Staff reliability and availability

## Maintenance Activities:-

IEEE provides a framework for sequential maintenance process activities. It can be used in iterative manner and can be extended so that customized items and processes can be included.



**Cost Estimation:-**

For any new software project, it is necessary to know how much it will cost to develop and how much development time will it take. These estimates are needed before development is initiated, but how is this done? Several estimation procedures have been developed and are having the following attributes in common.

1.Project scope must be established in advanced.

2.Software metrics are used as a support from which evaluation is made.

3.The project is broken into small PCs which are estimated individually.  
To achieve true cost & schedule estimate, several option arise.

5.Delay estimation

6.Used symbol decomposition techniques to generate project cost and schedule estimates.

7.Acquire one or more automated estimation tools.

**Uses Of Cost Estimation:-**

1.During the planning stage, one needs to choose how many engineers are required for the project and to develop a schedule.

2.In monitoring the project's progress, one needs to access whether the project is progressing according to the procedure and takes corrective action, if necessary.