

**A  
PROJECT REPORT  
ON  
  
AN ANDROID APP FOR  
CONFIDENTIAL CONVERSATION SYSTEM**

**By**

**Dhvanil Shah (ID No.: 14CEUOS041) (Roll no: CE108)  
Rushabh Shah (ID No.: 14CEUOS078) (Roll no: CE113)**

**For**

**BACHELOR OF TECHNOLOGY  
In  
Subject of  
Computer Engineering  
CE621 System Design Practice**

Prof. Jigar M. Pandya  
Assistant Professor  
Dept. of Comp. Engg.

Dr. C. K. Bhensdadia  
Head,  
Dept. of Comp. Engg.



**Faculty of Technology  
Department of Computer Engineering  
Dharmsinh Desai University  
April 2017**

# CERTIFICATE

This is to certify that the project work titled

Confidential Conversation System

is the bonafide work of

Dhvanil Shah (ID No.: 14CEUOS041) (Roll no: CE108)

Rushabh Shah (ID No.: 14CEUOS078) (Roll no: CE113)

Carried for the subject of System Design Practice of degree of Bachelor of Technology in  
Computer Engineering at Dharmsinh Desai University in the academic session  
December 2016 to April 2017

Prof. Jigar M. Pandya  
Asst. Prof  
Dept. of Comp. Engg.

Dr. C. K. Bhensdadia  
Head,  
Dept. of Comp. Engg.



**Faculty of Technology  
Department of Computer Engineering  
Dharmsinh Desai University  
April 2017**

## Acknowledgements

We, the developers of *Confidential Conversation System*, with immense pleasure and commitment would like to present the project assignment. The development of this app has given us wide opportunity to think, implement and interact with various aspects of management skills as well as the new emerging technologies.

Every work that one completes successfully stands on the constants encouragement, good will and support of the people around. We hereby avail this opportunity to express our gratitude to number of people who extended their valuable time, full support and cooperation in developing the project.

We express deep sense of gratitude towards our project guide **Prof. Jigar M. Pandya** towards their innovative suggestions. It is because of them, that we were prompted to do hard work, adopting new technologies.

We would also like to thank **Prof. Ankit P. Vaishnav** for his guidelines through the design and analysis phase of the development.

We are sincerely thankful to Head of CE Department, **Dr. C.K. Bhensdadia** for the support during the whole session of study and development.

They altogether provide us favorable environment, and without them it would not have been possible to achieve our goal.

Thanks,

Dhvanil Shah  
Rushabh Shah

# TABLE OF CONTENTS

CERTIFICATE .....	2
Acknowledgements .....	3
TABLE OF CONTENTS .....	4
Chapter 1 Abstraction .....	8
Chapter 2 .....	9
2.1 Project Details .....	9
2.2 Technology Used .....	9
2.2.1 Front End Tool: Android Studio .....	9
2.2.2 Language: Java .....	9
2.2.3 Diagram Tool: UMLET .....	9
2.2.4 Diagram Tool: ERDPlus .....	9
Chapter 3 Software Requirements Specification .....	10
3.1 Purpose .....	10
3.2 Scope .....	10
3.3 Project Definition .....	10
3.4 Definitions, Acronyms and Abbreviations .....	10
3.6 Product Perspective .....	10
3.6.1 System Interfaces .....	11
3.6.2 Hardware Interfaces .....	11
3.6.3 Software Interfaces .....	11
3.6.4 Communication Interfaces .....	11
3.6.5 Memory Constraints .....	11
3.6.6 System Adaptation Requirements .....	11
3.7 Product Functions .....	11
3.8 User Characteristics .....	11
3.9 Constraints .....	12
3.9.1 Language .....	12

3.9.2 Operating system .....	12
3.9.3 Required Memory .....	12
3.9.4 Tools Required for Development.....	12
3.10 Apportioning of requirement .....	12
3.11 Functional Requirement.....	12
3.12 Non-functional Requirement .....	13
3.12.1 Security Requirements.....	13
3.12.2 Other Requirements .....	13
Chapter 4 Design.....	15
4.1 Use-Case Diagram for Confidential Messaging System .....	15
4.2 Class Diagram for Confidential Messaging System .....	16
4.3 Sequence Diagrams for Confidential Messaging System.....	17
4.3.1 Login Sequence.....	17
4.3.2 Message Send Sequence .....	18
4.4 Activity Network .....	19
4.5 Entity Relationship Diagram.....	20
4.5.1 Entity-Relationship Diagram for <i>Confidential Messaging System</i> at Server.....	20
4.5.2 Entity-Relationship Diagram for <i>Confidential Messaging System</i> At Client .....	20
4.6 Data Dictionary .....	21
4.6.1 Server-side Tables.....	21
4.7 Data Definition Language (DDL).....	23
Chapter 5 Implementation.....	24
5.1 Main modules of the system .....	24
5.2 Division of Work .....	24
5.3 Module Details.....	25
5.3.1 Home Screen (Two Tabs Chats & Contacts).....	25
5.3.2 Chat Screen .....	26
5.3.3 Encryption Modules.....	26
5.3.4 Interaction with Server.....	28
5.3.5 Services .....	29
Chapter 6 Testing.....	34

6.1 Testing Plan .....	34
6.2 Testing Strategy .....	34
6.2.1 Unit Testing .....	34
6.3 Test Cases .....	35
6.3.1 Registration .....	35
6.3.2 Login .....	36
6.3.3 PIN .....	37
6.3.4 Change Passcode.....	38
6.3.5 Send Message.....	39
6.3.6 Screenshot .....	40
6.3.7 Merging Contacts.....	40
6.3.8 Logout .....	40
Chapter 7 Screenshots.....	41
7.1 Register Activity .....	41
7.2 Login Activity .....	42
7.3 Contact Fragment.....	43
7.4 Chat Fragment.....	44
7.5 Settings.....	45
7.6 Menu .....	46
7.7 Change Passcode.....	47
7.8 Chat Activity .....	48
7.9 Passcode Screen .....	49
Chapter 8 Conclusion.....	50
Chapter 9 Limitation and Future Extension.....	51
9.1 Limitation:.....	51
9.2 Future Extension: .....	51
Chapter 10 Miscellaneous.....	52
10.1 JSON (JavaScript Object Notation).....	52
10.2 Session Management in Android.....	53
10.2.1 Shared Preference .....	53
10.3 GitHub.....	53

10.4 Source Package/Directory Structure .....	55
10.4.1 The Overall Directory Structure .....	55
10.4.2 The Structure of app folder .....	56
10.4.3 The structure depicting the main files developed by us.....	57
10.5 Software Version Description Document .....	58
10.6 Deployment Procedure.....	59
Bibliography .....	60

# Chapter 1

## Abstract

---

Today, in the age of digitization, it is very important to be in touch with each other, i.e., communication is a basic necessity. But, with the increasing digitization, the thing which have also increased is the concern for security. Day-and-now, we keep hearing about the attacks on the security of any organization.

So, taking above thing in consideration, we have prepared an Android app, which helps to communicate with each other, but with great security. We have used various types of Encryption, to prevent almost every type of attack. In addition, we have added features like Auto-Deletion of messages after a given interval of time, preventing from taking a screenshot and some others.



# Chapter 2

## Introduction

---

### 2.1 Project Details

Confidential Conversation System is in a way simple Android App, similar to others. But, the thing that differentiates it from others

It utilizes the service created in php and also Firebase Cloud Messaging System, provided by Google. It allows the users to Register to the System. Once he is registered he has to log in once in the system, i.e. the App, and then he doesn't need to log in once again on that device.

This System only allows the conversation between two users only if both of them have each other in their contact list, and thus adding extra layer of security. Also, the messages are encrypted through Blowfish and RSA Algorithm. It also prevents the users from taking the screen shots, thus eliminating the potential threats. Furthermore, the messages get destroyed after the time specified by the sender. So, in all, it covers all the features, that a secured messaging app must have.

### 2.2 Technology Used

#### 2.2.1 Front End Tool: Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android platform development. It is used to develop Android Applications. It internally uses Java programming language and XML.

#### 2.2.2 Language: Java

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation.

#### 2.2.3 Diagram Tool: UMLET

UMLET is a UML tool for Linux and Windows. It is used for creating UML diagrams and other various types of diagram.

#### 2.2.4 Diagram Tool: ERDPlus

ERDPlus is an online tool to create Entity Relationship Diagram.

## Chapter 3

# Software Requirement Specifications

---

### 3.1 Purpose

The purpose of this document is to specify the detailed description of the Confidential Conversation System that we will design and implement. This document is intended to the Customers and Developers. It will explain the purpose and features of the system, what the system will do, the constraints which it must operate.

### 3.2 Scope

The Goal of the System is to develop a Secured Conversation Environment. The System provides one type of user. The system will allow user to securely message another user. The system will have database of application at server side and an in-built database in the app itself.

### 3.3 Project Definition

The Confidential Conversation System allows to send secure messages by using various encryption methods. It also restricts the user from taking a screen shot, and also receive messages from those only who are present in their contact list.

### 3.4 Definitions, Acronyms and Abbreviations

- Database: Collection of Information managed by the system.
- DBMS: Database Management System
- Developer: A person who develops applications.
- User: A person who uses the application developed by Developers. 1 .5 Overview

### 3.5 Overview

Following section of this document will focus on describing the system in terms of product, product functions, dependencies and assumptions. In the third section the specific requirements of the system will be addressed.

### 3.6 Product Perspective

The following subsections describe how the software operates inside various constraints.

### 3.6.1 System Interfaces

This system will not interact with any other outside interfaces.

### 3.6.2 Hardware Interfaces

This system does not have any hardware interfaces.

### 3.6.3 Software Interfaces

The System will interface with Database System that stores the information necessary for the application to operate. The DBMS must store all data related to account information. System uses SQL Server provided by *phpmyadmin* to interact with the database.

### 3.6.4 Communication Interfaces

The System will interface with an Internet Connection to maintain communication with all its devices. It will use a reliable type IP protocol such TCP/IP for maximum compatibility and stability. All devices it interfaces should contain standard compatible software and hardware accessible internet to maintain communication between the server and the mobiles.

### 3.6.5 Memory Constraints

This application is meant to run on devices with Random Access Memory (RAM) greater than 512 MB with internal free data space of at least 50 MB.

### 3.6.6 System Adaptation Requirements

The system requires an Android phone having the OS Jellybean or above.

## 3.7 Product Functions

- User Registration
- User Authentication
- Display Chat
- Send Message
- Read Message
- Auto Deletion of Message

## 3.8 User Characteristics

There is only one type of user. The whole system is meant for to be use of one person personally only. She/he can chat secretly with the person intended without caring about the leak of any information from the system side.

## **3.9 Constraints**

### **3.9.1 Language**

The software currently will operate in English (US) language only.

### **3.9.2 Operating system**

This system requires Android OS having version of Jellybean or above.

### **3.9.3 Required Memory**

This software will require a minimum of 512 MB RAM (Random Access Memory).

### **3.9.4 Tools Required for Development**

- Front End Tool: Android Studio
- A rich, integrated development environment for creating stunning applications for Android.
- Language: Java
- Diagram Tools: UMLET
- Sublime Text as an editor for developing php files
- Database Server: phpmyadmin MySQL
- Documentation Software: Microsoft Word 2016

## **3.10 Apportioning of requirement**

In the case that the project is delayed, there are some requirements that could be transferred to the next version of the application. Those requirements are to be developed in the next release.

## **3.11 Functional Requirement**

### **3.11.1 Functional Requirement 1**

System should facilitate user to login or register with the app when using first time

- System should provide facility for user to login using his username and password.
- System should provide facility to add new user.
- System should provide interface to change password.

#### **3.11.2 Functional Requirement 2**

System should facilitate user to secure app by PIN.

- System should provide interface to change PIN.

#### **3.11.3 Functional Requirement 3**

System should provide facility to send message to other user which are in contacts using end to end encryption.

#### **3.11.4 Functional Requirement 4**

System should provide facility to user to self-destruct message immediately after user provided time.

#### **3.11.5 Functional Requirement 5**

System should provide facility to user to find other users via contact information.

#### **3.11.6 Functional Requirement 6**

System should provide facility to user to logout from application.

### **3.12 Non-functional Requirement**

#### **3.12.1 Security Requirements**

1. System will not allow user to take screenshots of any chats.
2. System will use Blowfish Encryption to store messages in user's mobile device.
3. System will use RSA Encryption while sending message to other users.
4. System will lock application with pin as soon as user close application.
5. System will check if user(receiver) is online then only user(Sender) can send him messages.
6. System will not allow user to login from two devices.

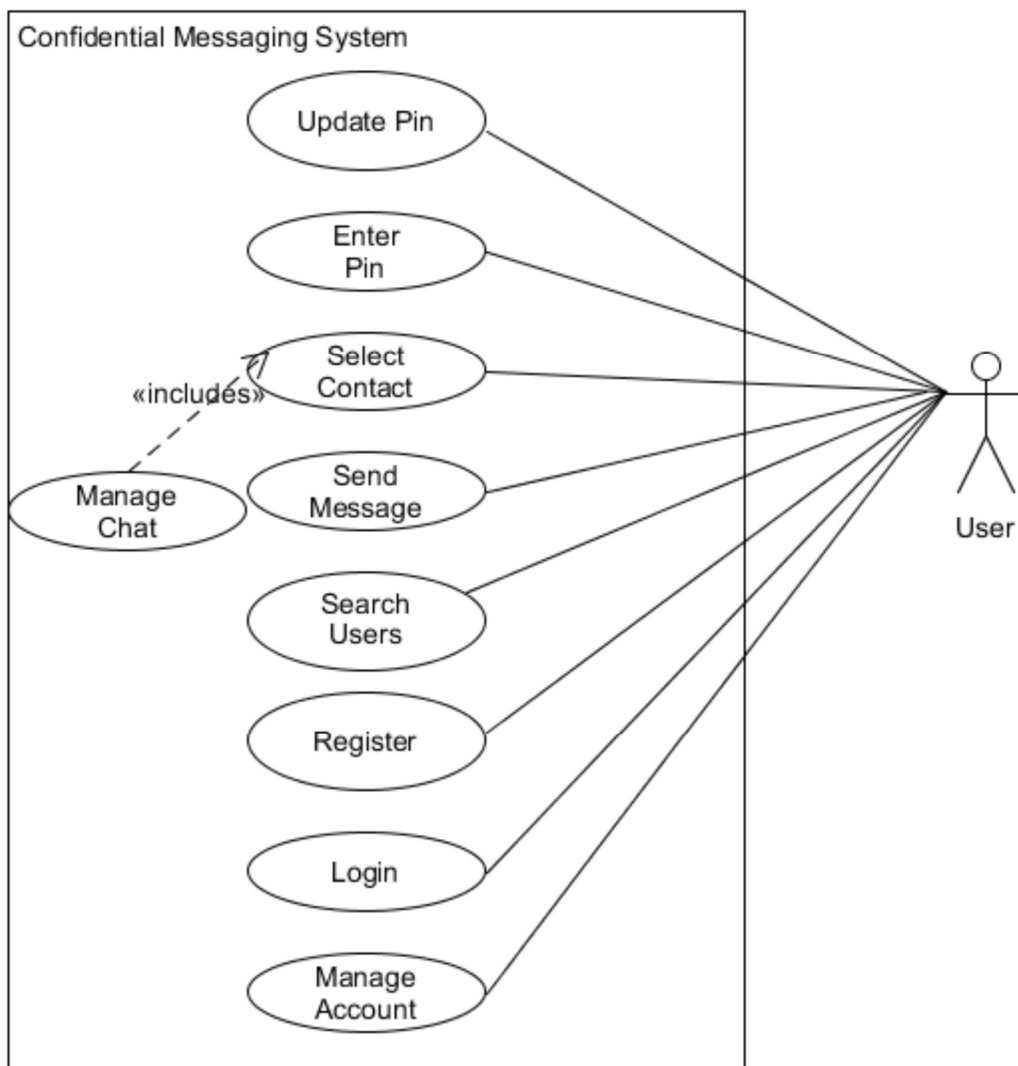
#### **3.12.2 Other Requirements**

1. System (Android application) will use REST based services of PHP Server for all functionality except getting notification.
2. System (Android application) will use Google's Firebase Service to get notification.
3. System (PHP Server) will send message to other users using Google's Firebase Service.

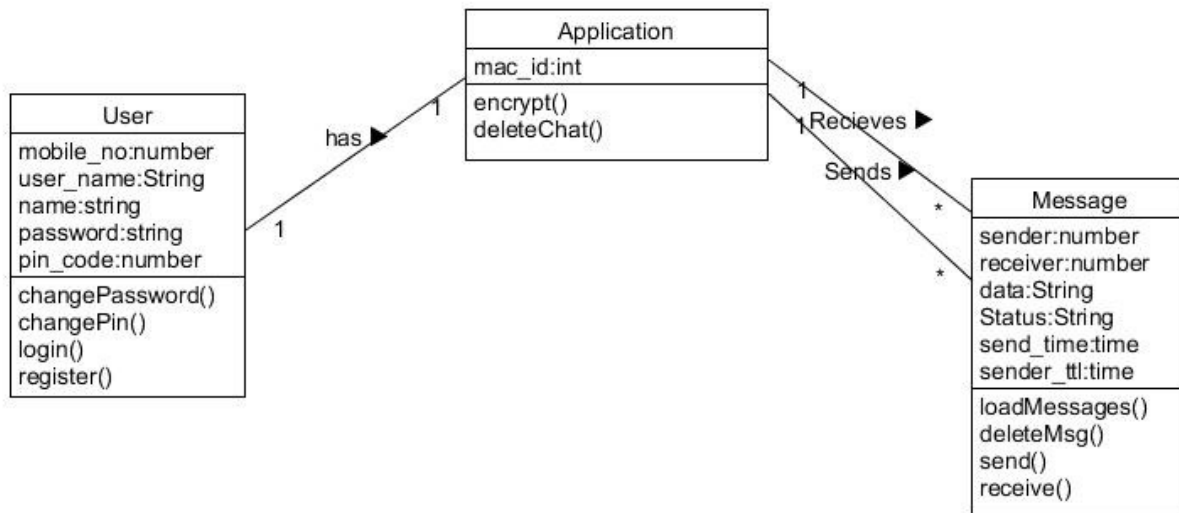
## Chapter 4

### Design

#### 4.1 Use-Case Diagram for Confidential Messaging System



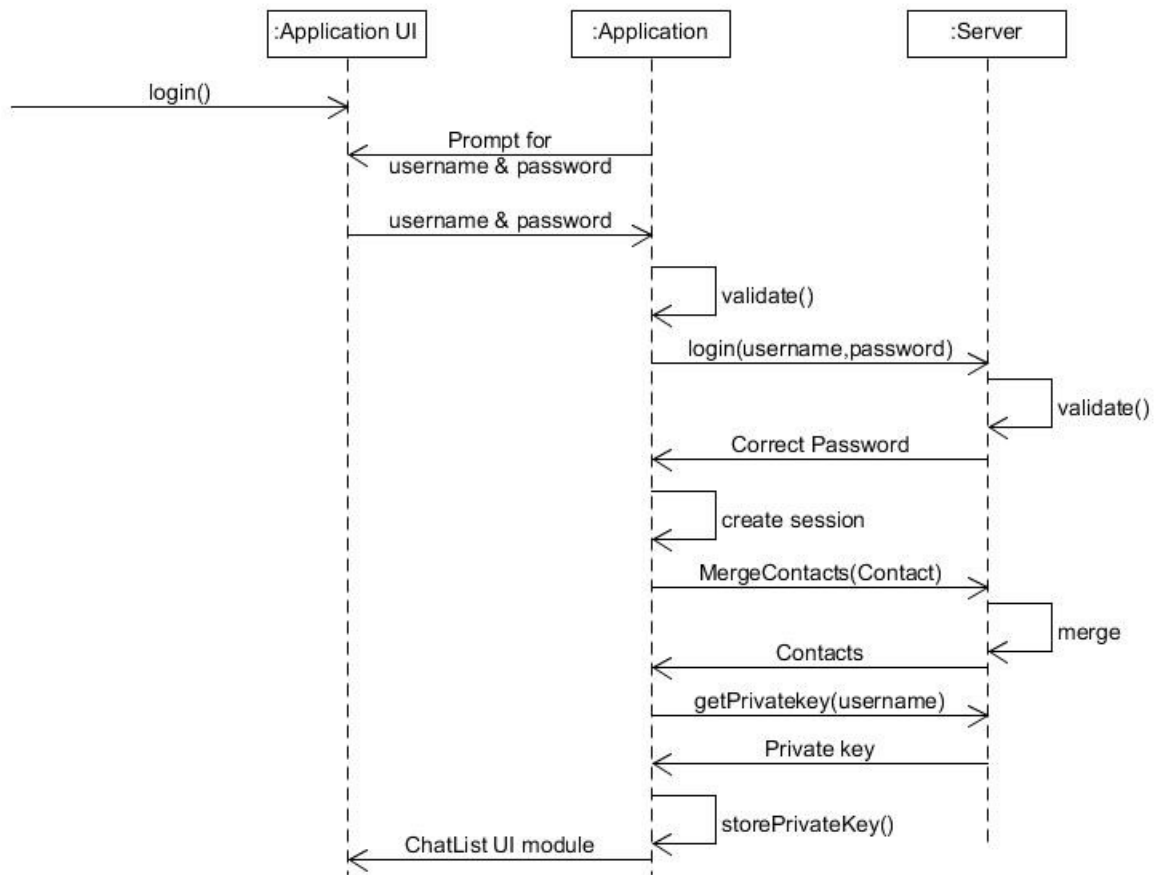
## 4.2 Class Diagram for Confidential Messaging System



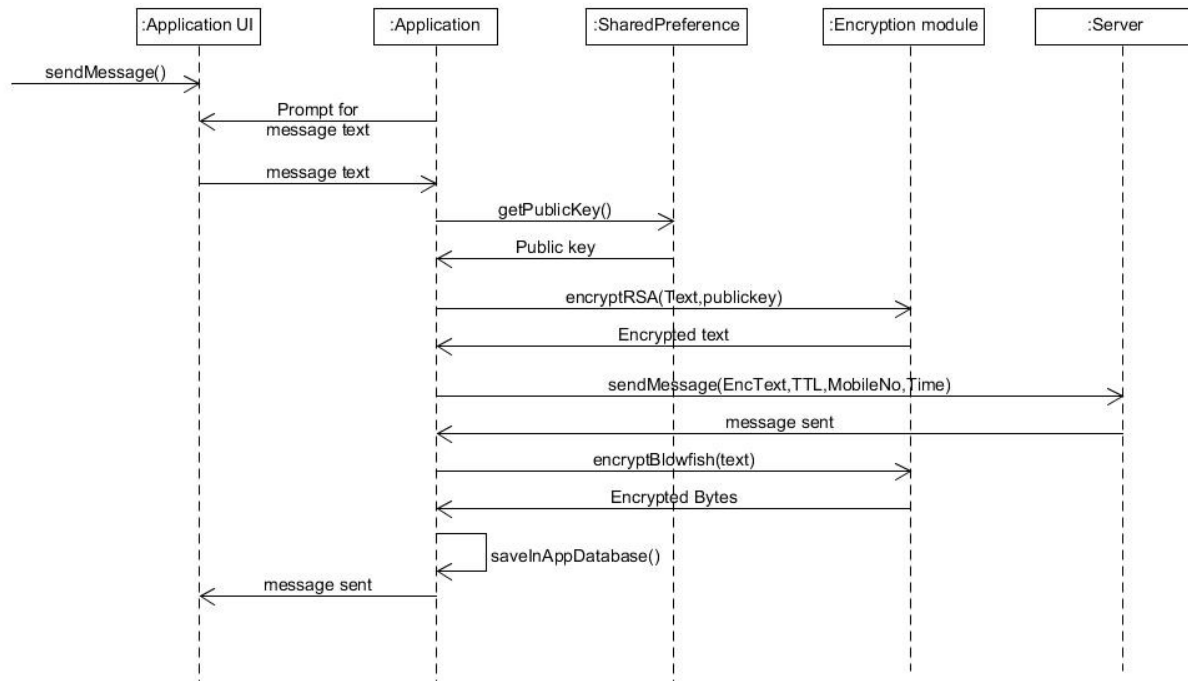


## 4.3 Sequence Diagrams for Confidential Messaging System

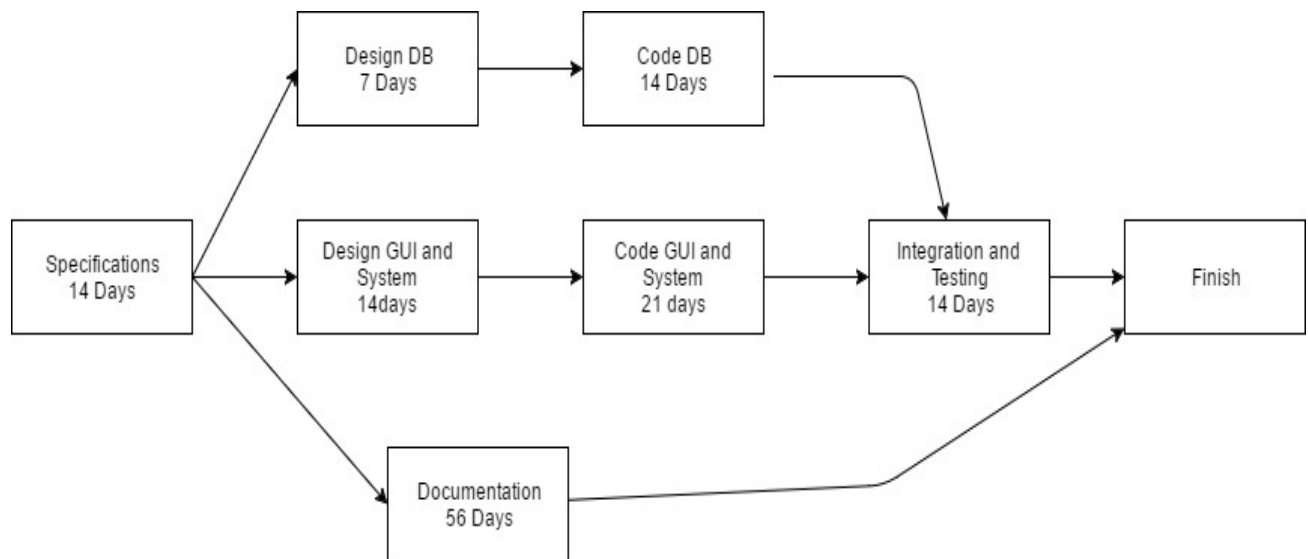
### 4.3.1 Login Sequence



### 4.3.2 Message Send Sequence

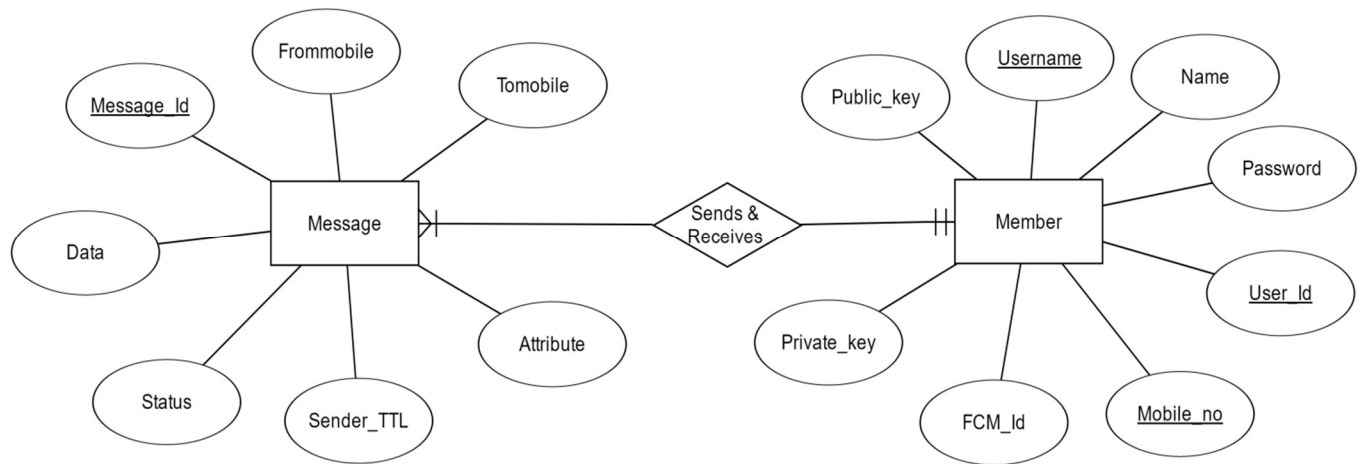


#### 4.4 Activity Network

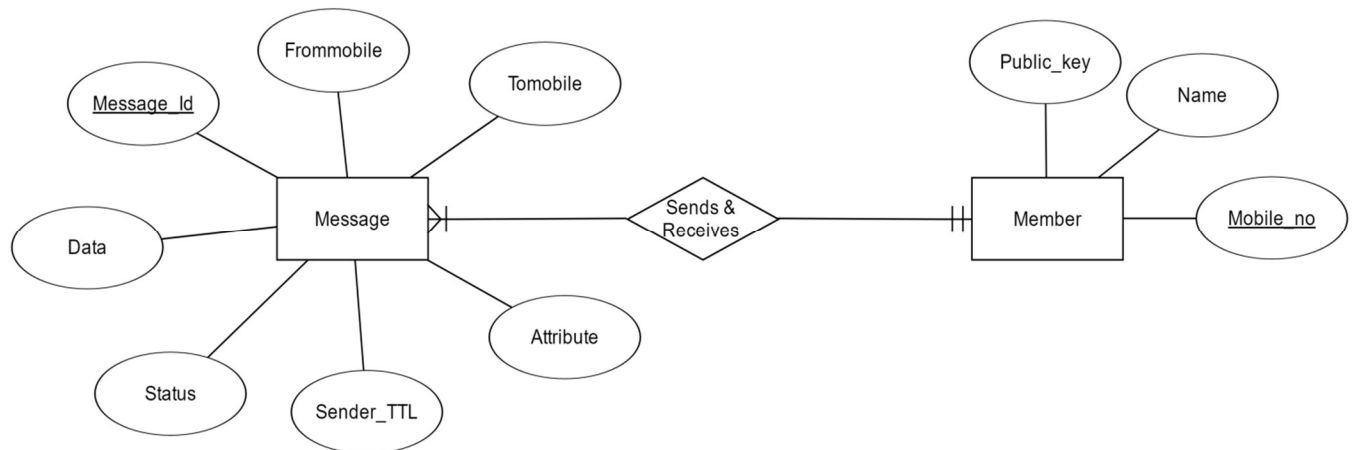


## 4.5 Entity Relationship Diagram

### 4.5.1 Entity-Relationship Diagram for *Confidential Messaging System* at Server



### 4.5.2 Entity-Relationship Diagram for *Confidential Messaging System* At Client



## 4.6 Data Dictionary

### 4.6.1 Server-side Tables

#### *Member Table*

<b>Name</b>	<b>Type</b>
Username	Varchar (20)
Password	Varchar (20)
Id (Primary Key)	Int (11)
Email	Varchar (25)
Mobile	Decimal (10,0)
Name	Text
Fcmid	Varchar(1000)
Public_key	Mediumtext
Private_key	Mediumtext

#### *Message Table*

<b>Name</b>	<b>Type</b>
Id	Int(11)
Frommobile	Varchar (20)
Tomobile	Varchar (20)
Creationtime	Int (11)
Senderttl	Int(11)
Data	mediumtext
Status	Varchar(1000)

## Client-side Tables

### Message Table

id	name	type	notnull	dflt_value	pk
0	id	integer	0		1
1	frommobile	TEXT	0		0
2	tomobile	text	0		0
3	data	text	0		0
4	creationtime	text	0		0
5	senderttl	int	0		0
6	status	text	0		0
7	action	text	0		0

### Member Table

id	name	type	notnull	dflt_value	pk
0	Name	TEXT	0		0
1	Number	TEXT	0		0
2	key	String	0		0

## 4.7 Data Definition Language (DDL)

### Create Member Table

```
CREATE TABLE `member` (  
  `username` varchar(20) NOT NULL,  
  `password` varchar(20) NOT NULL,  
  `id` int(11) NOT NULL,  
  `email` varchar(25) NOT NULL,  
  `mobile` decimal(10,0) NOT NULL,  
  `name` text NOT NULL,  
  `FCMID` varchar(1000) NOT NULL,  
  `public_key` mediumtext NOT NULL,  
  `private_key` mediumtext NOT NULL  
);
```

### Create Message Table

```
CREATE TABLE `message` (  
  `id` int(11) NOT NULL,  
  `frommobile` varchar(20) DEFAULT NULL,  
  `tomobile` varchar(20) DEFAULT NULL,  
  `creationtime` int(11) DEFAULT NULL,  
  `senderttl` int(11) DEFAULT NULL,  
  `data` mediumtext,  
  `status` varchar(100) NOT NULL  
);
```

## Chapter 5

# Implementation

---

### 5.1 The main modules present in the system are:

- Login, Register
- Find friends
- Send Message
- Blowfish Encryption
- RSA Encryption
- Message Self Destruction
- Pin Code Locking

### 5.2 Division of Work

Dhvanil Shah

#### Web Service

- Login
- Contact merging

#### App Modules

- Login Activity
- Contacts fragment
- Chat Activity
- Blowfish implementation
- Settings activity

Rushabh Shah

#### Web Service

- Register
- Send message



### App modules

- Register Activity
- Chatlist fragment
- Lock screen
- RSA implementation
- Chat activity

### *Notes*

This application communicates with PHP server using REST messages and receives notification via Google's Firebase Service.

## **5.3 Module Details**

### **First time use**

User will have to click on register link then register using email, mobile no, username, password and application will use register service at server. Then he will login using username and password which will be checked by server and then all mobile contacts will be sent to server and server will send back contacts which are already registered at server.

### **Not First time use**

If user is not logged out then Passcode Lock Screen will appear. If user has not set passcode then he has to enter initial passcode or his stored passcode. And if user is logged out then Login Screen will appear and he has to again log in.

### **After Login or Lock Screen**

#### **5.3.1 Home Screen (Two Tabs Chats & Contacts)**

If user has already logged in then application will retrieve contacts from local database else contacts will be retrieved from server. User can refresh contact list by click on refresh button on settings.

If user selects one of that contacts new chat will be initiated. If other user is not logged in at server then user cannot send message to him.

User can clear all chats by clear all chats setting. User can change time to live message in setting. User can change passcode and account information in settings menu.

When user press logout in settings users phones database will be cleared and user will also be logged out at server and he cannot receive messages.

### 5.3.2 Chat Screen

User will find messages which are delivered to him or he sent to other user by him on this screen with its sending time or receiving time. User will send message by send message button then message will be encrypted using *RSA* algorithm and it will be stored at phone's database after encrypted by *Blowfish* Algorithm.

When user selects contact from chat list or contact list screen application will fetch public key of receiver from server. This key will be used while sending message.

### 5.3.3 Encryption Modules

#### RSA Encryption

Given Code Snippet is used for encoding of keys.

```
public static PublicKey stringToPublicKey(String s) {

    byte[] c = null;
    KeyFactory keyFact = null;
    PublicKey returnKey = null;

    try {
        c = Base64.decode(s, Base64.DEFAULT);
        keyFact = KeyFactory.getInstance("RSA");
    } catch (Exception e) {
        System.out.println("Error in Keygen");
        e.printStackTrace();
    }

    X509EncodedKeySpec x509KeySpec = new X509EncodedKeySpec(c);
    try {
        returnKey = keyFact.generatePublic(x509KeySpec);
    } catch (Exception e) {

        System.out.println("Error in Keygen2");
        e.printStackTrace();
    }

    return returnKey;
}

public static PrivateKey stringToPrivateKey(String s) {

    byte[] c = null;
    KeyFactory keyFact = null;
    PrivateKey returnKey = null;

    try {

        c = Base64.decode(s, Base64.DEFAULT);
        keyFact = KeyFactory.getInstance("RSA");
    } catch (Exception ex) {

        Log.e("Error", ex.toString());
        ex.printStackTrace();
    }
}
```

```

    }

    PKCS8EncodedKeySpec x509KeySpec = new PKCS8EncodedKeySpec(c);
    try { //the next line causes the crash
        returnKey = keyFact.generatePrivate(x509KeySpec);
    } catch (Exception ex) {

        Log.e("Error", ex.toString());
        ex.printStackTrace();
    }

    return returnKey;
}

```

Given Code Snippet is used for encryption and decryption.

```

public static String publicKeyToString(PublicKey p) {

    byte[] publicKeyBytes = p.getEncoded();
    return Base64.encodeToString(publicKeyBytes, Base64.DEFAULT);

}

public static String privateKeyToString(PrivateKey p) {

    byte[] privateKeyBytes = p.getEncoded();
    return Base64.encodeToString(privateKeyBytes, Base64.DEFAULT);

}

public static KeyPair getKeys() throws NoSuchAlgorithmException,
NoSuchPaddingException, InvalidKeyException,
IllegalBlockSizeException, BadPaddingException, InvalidAlgorithmParameterException
{
    KeyPairGenerator kpg;
    RSAKeyGenParameterSpec spec = new RSAKeyGenParameterSpec(1024,
RSAKeyGenParameterSpec.F4);
    kpg = KeyPairGenerator.getInstance("RSA");
    kpg.initialize(spec);
    KeyPair kp = kpg.genKeyPair();
    return kp;
}

public static String ServerEncrypt (String plain,String public_key) throws
NoSuchAlgorithmException, NoSuchPaddingException, InvalidKeyException,
IllegalBlockSizeException, BadPaddingException, InvalidKeySpecException
{
    byte[] encryptedBytes;
    String encrypted;
    Cipher cipher;
    cipher = Cipher.getInstance("RSA");
    PublicKey pb= stringToPublicKey(public_key);
    cipher.init(Cipher.ENCRYPT_MODE,pb);
    encryptedBytes = cipher.doFinal(plain.getBytes());
    encrypted = Base64.encodeToString(encryptedBytes,Base64.DEFAULT);
    return encrypted;
}

public static String ServerDecrypt (String result,String private_key) throws
NoSuchAlgorithmException, NoSuchPaddingException, InvalidKeyException,
IllegalBlockSizeException, BadPaddingException, InvalidKeySpecException

```

```

{
    byte[] decryptedBytes;
    String decrypted;
    Cipher cipher1;
    cipher1=Cipher.getInstance("RSA");
    PrivateKey pr= stringToPrivateKey(private_key);
    cipher1.init(Cipher.DECRYPT_MODE,pr);
    decryptedBytes = cipher1.doFinal(Base64.decode(result,Base64.DEFAULT));
    decrypted = new String(decryptedBytes);
    return decrypted;
}

```

## Blowfish Encryption

```

public static byte[] encryptClient(String plainText) throws GeneralSecurityException {

    SecretKey secret_key = new SecretKeySpec(KEY_CLIENT.getBytes(), ALGORITHM_CLIENT);

    Cipher cipher = Cipher.getInstance(ALGORITHM_CLIENT);
    cipher.init(Cipher.ENCRYPT_MODE, secret_key);

    return cipher.doFinal(plainText.getBytes());
}

public static String decryptClient(byte[] encryptedText) throws
GeneralSecurityException {

    SecretKey secret_key = new SecretKeySpec(KEY_CLIENT.getBytes(), ALGORITHM_CLIENT);

    Cipher cipher = Cipher.getInstance(ALGORITHM_CLIENT);
    cipher.init(Cipher.DECRYPT_MODE, secret_key);

    byte[] decrypted = cipher.doFinal(encryptedText);

    return new String(decrypted);
}

```

### 5.3.4 Interaction with Server

Given Code snippet is used for communication with server.

```

public static String getData(RequestPackage p) {
    int statusCode = 0;
    BufferedReader reader = null;
    String uri = p.getUri();
    if (p.getMethod().equals("GET")) {
        uri += "?" + p.getEncodedParams();
    }

    try {
        URL url = new URL(uri);
        HttpURLConnection con = (HttpURLConnection) url.openConnection();
        con.setRequestMethod(p.getMethod());

        if (p.getMethod().equals("POST")) {
            con.setDoOutput(true);
            OutputStreamWriter writer = new OutputStreamWriter(con.getOutputStream());
            writer.write(p.getEncodedParams());
            writer.flush();
        }
    }
}

```

```

    }
    statusCode = con.getResponseCode();
    StringBuilder sb = new StringBuilder();
    reader = new BufferedReader(new InputStreamReader(con.getInputStream()));
    String line;
    while ((line = reader.readLine()) != null) {
        sb.append(line);
    }
    return sb.toString();
} catch (Exception e) {
    if(statusCode == 404){
        return "Requested resource not found";

    }else if(statusCode == 500){
        return "Something went wrong at server end";

    }else{
        return "Unexpected Error occurred! [Most common Error: Device might not be
connected to Internet]"+" " +
            "statusCode"+statusCode+e.toString();

    }
} finally {
    if (reader != null) {
        try {
            reader.close();
        } catch (IOException e) {
            e.printStackTrace();
            return null;
        }
    }
}
}
}

```

### 5.3.5 Services

#### • PHP Server's Services

Server provide login, register, contact merging, encryption key fetching and message sending services.

Here Are Code snippet of some services

#### Login

```

if($type=='login'){

    $user_name = $_REQUEST['un'];
    $user_pass = $_REQUEST['pw'];
    $fcid = $_REQUEST['fcid'];
    $mysql_qry = "select id,username,password,private_key from member where
username='$user_name' and password = '$user_pass'";

```

```

$result = mysqli_query($conn,$mysql_gry);

//$row = mysqli_num_rows($result);
if($row = $result->fetch_assoc())
{
    $mysql_gry2 = "update member set FCMID='$fcmid' where
username='$user_name'";
    $result2 = mysqli_query($conn,$mysql_gry2);
    echo $row['private_key'];
}
else
{
    echo "0";
}
}

```

### Register

```

if($type=='register'){
    $name = $_REQUEST['name'];
    $email = $_REQUEST['email'];
    $mobile = $_REQUEST['mobile'];
    $pw = $_REQUEST['pw'];
    $un = $_REQUEST['un'];
    $pb_key = $_REQUEST['public_key'];
    $pr_key = $_REQUEST['private_key'];
    $q = "insert into member(username, password, name, email,
mobile,public_key,private_key)
values('$un','$pw','$name','$email','$mobile','$pb_key','$pr_key')";
    $result = mysqli_query($conn,$q);
    if($result){
        echo "1";
    }
    else echo "0";
}
else if($type=='getuserlist'){
    $q = "select mobile from member";
    $json1 = $_REQUEST['json1'];
    $earray = array();
    $earray = json_decode($json1,true);
    $result = mysqli_query($conn,$q);
    $semparray = array();
    while($row = mysqli_fetch_assoc($result))
    {
        $semparray[] = $row["mobile"];
    }
    $r = array_intersect($earray,$semparray);
    $array1=array_keys($r);
    echo json_encode($array1);
}

```

### Send Message

```

else if($type=='sendmessage'){
    $frommobile = $_REQUEST['frommobile'];
    $tomobile = $_REQUEST['tomobile'];
    $senderttl = $_REQUEST['senderttl'];
    $data = $_REQUEST['data'];
    $time =time();

```

```

$q = "insert into message(frommobile, tomobile, senderttl, data, creationtime,
status) values('$frommobile','$tomobile','$senderttl','$data','$time','Pending')";
$result = mysqli_query($conn,$q);
$msg_id = mysqli_insert_id($conn);
if($result){
    $q2 = "select * form member where tomobile= ".$tomobile;
    $result2 = mysqli_query($conn,$q2);
    send_multipal_push_notification_FCM
    (array_GCMIDs_by_AppGroupID($tomobile)
    ,json_Message_by_ID($msg_id));
    echo "1";
}
else echo "0";
}

```

### • Notification Service

It uses Google's Firebase Service to receive notification. After Receiving them it Decrypt using *RSA* Algorithm and then store it at phone's database after encrypting using *Blowfish* Algorithm and prompts user about new message. And Starts auto delete message service with time to leave from message.

Here is code snippet of this service

```

public class MyFirebaseMessagingService extends FirebaseMessagingService {
    String frommobile, tomobile, data, status, decText;
    int senderttl;
    long creationtime;
    private Handler h=new Handler();
    LocalBroadcastManager broadcaster;
    Context context;
    final static public String COPA_RESULT =
    "com.example.imdhv.blumed.MyFirebaseMessagingService.REQUEST_PROCESSED";
    @Override
    public void onMessageReceived(RemoteMessage remoteMessage) {
        SharedPreferences sp=PreferenceManager.getDefaultSharedPreferences(this);
        int caid=sp.getInt("caid",0);
        broadcaster = LocalBroadcastManager.getInstance(this);
        Intent intent = new Intent(this, LockScreenActivity.class);
        intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0,
        intent, PendingIntent.FLAG_ONE_SHOT);
        Uri notificationSound =
        RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
        NotificationCompat.Builder mBuilder = new
        NotificationCompat.Builder(this);
        mBuilder.setSmallIcon(R.mipmap.ic_shortcut_chat_bubble);
        mBuilder.setTitle("Blumed");
        mBuilder.setText("New Message");
        mBuilder.setAutoCancel(true);
        mBuilder.setSound(notificationSound);
        mBuilder.setContentIntent(pendingIntent);
        mBuilder.setPriority(Notification.PRIORITY_HIGH);
        NotificationManager mNotificationManager = (NotificationManager)
        getSystemService(Context.NOTIFICATION_SERVICE);
    }
}

```

```

mNotificationManager.notify(0, mBuilder.build());
Log.e("FCM MSG", "From: " + remoteMessage.getData());
String msgobj = remoteMessage.getData().get("newmessage");
try {
    JSONArray arr = new JSONArray(msgobj);
    SQLiteDatabase database =
openOrCreateDatabase("/sdcard/userlists.db", SQLiteDatabase.CREATE_IF_NECESSARY,
null);

    database.execSQL("CREATE TABLE IF NOT EXISTS MESSAGE (id integer
primary key autoincrement,frommobile TEXT, tomobile text, data text, creationtime
text,senderttl int,status text,action text);");
    for (int i = 0; i < arr.length(); i++) {
        JSONObject obj = arr.getJSONObject(i);
        frommobile = obj.get("frommobile").toString();
        tomobile = obj.get("tomobile").toString();
        data = obj.get("data").toString();
        data=Utility.ServerDecrypt(data,sp.getString("private_key",""));
        byte[] enc = Utility.encryptClient(data);
        String a = obj.get("creationtime").toString();
        creationtime = Long.parseLong(a);
        String b = obj.get("senderttl").toString();
        senderttl = Integer.parseInt(b);
        String status = obj.get("status").toString();
        boolean chatresult;
        Log.e("FCM MSG", "Data Received");
        chatresult = checkforchatlist();
        Log.e("FCM MSG", "Result of chatresult" + chatresult);
        if (chatresult == true) {
            ContentValues cv = new ContentValues();
            cv.put("frommobile", frommobile);
            cv.put("tomobile", tomobile);
            cv.put("data", enc);
            cv.put("creationtime", creationtime);
            cv.put("senderttl", senderttl);
            cv.put("status", status);
            cv.put("action", "r");
            database.insertOrThrow("MESSAGE", null, cv);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    intent = new Intent(COPA_RESULT);
    broadcaster.sendBroadcast(intent);
}

```

### • Auto Delete Message Service

It sets alarm at current time + time to leave. And when application alarmed by it application will delete that message from database.

Here is code snippet for this service



```

public class OnAlarmReceive extends BroadcastReceiver {

    protected String TAG = "Bound";
    LocalBroadcastManager broadcaster;
    final static public String COPA_RESULT =
"com.example.imdhv.blumed.MyFirebaseMessagingService.REQUEST_PROCESSED";

    @Override
    public void onReceive(Context context, Intent intent) {

        String id = intent.getStringExtra("id");
        Log.e(TAG, "On the verge of deleting the message with id: "+id);
        SQLiteDatabase database = context.openOrCreateDatabase("/sdcard/userlists.db",
SQLiteDatabase.CREATE_IF_NECESSARY, null);
        database.execSQL("DELETE FROM " + "MESSAGE" + " WHERE " + "id" + "= '" + id +
        "'");

        broadcaster = LocalBroadcastManager.getInstance(context);
        intent = new Intent(COPA_RESULT);
        broadcaster.sendBroadcast(intent);
    }
}

```

## Chapter 6

# Testing

---

### 6.1 Testing Plan

Technique that's going to be used is black box. The expected inputs to the system are applied and only the outputs are checked.

### 6.2 Testing Strategy

The development process repeats this testing process for the following phases:

1. Unit Testing
2. Integration Testing

Unit testing is checked for a given module after it has been prepared. It is vigorously checked for various test cases to find for errors, if any. It is the most basic and important step.

Other test is Integration testing which checks for the programs prepared till now, are working error free or not after integration. System testing ensures that the system meets its stated design specifications. Alpha and Beta testing are carried out by users for the final testing of the system.

We have done Unit Testing and System Testing, along with alpha and beta testing and fixed the errors in the process.

#### 6.2.1 Unit Testing

The objective of the unit testing is to test a unit of code using the unit test specification after the coding has finished.

## 6.3 Test Cases

### 6.3.1 Registration of Users (3.11.1 Functional Requirement 1)

- 1) Pre-condition: "Imdhvanil" username already exists

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Registration Activity	Name: David	Already Registered	User is already registered	Pass
2	Provide Details	Email: imdhvanil@gmail.com			
		Mobile: 7383557789			
		Username: Imdhvanil			
		Password: Imdhvanil			
3	Click Register				

- 2) Pre-condition: "dhv" username does not exists

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Registration Activity	Name: David	Username is too short	Username is short	Pass
2	Provide Details	Email: imdhvanil@gmail.com			
		Mobile: 7383557789			
		Username: dhv			
		Password: Imdhvanil			
3	Click Register				

- 3) Pre-condition: "Imdhvanil" username does not exists

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Registration Activity	Name: David	Password is too short	Password is short	Pass
2	Provide Details	Email: imdhvanil@gmail.com			

		Mobile: 7383557789			
		Username: Imdhvanil			
		Password: imdhv			
3	Click Register				

4) Pre-condition: "Imdhvanil" username does not exists

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Registration Activity	Name: David	Invalid email address	Invalid email address	Pass
2	Provide Details	Email: imdhvanilgmail.com			
		Mobile: 7383557789			
		Username: Imdhvanil			
		Password: imdhvnil			
3	Click Register				

### 6.3.2 Login (3.11.1 Functional Requirement 1)

1) Pre-condition: "Rushabh5x" username doesn't exists

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Login Activity	Username: Rushabh5x	Username or Password incorrect	Username or Password incorrect	Pass
2	Provide Details	Password: Rushabh5x			

- 2) Pre-condition: Password of user “Rushabh5x” is Rushabh5x doesn’t exists

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Login Activity	Username: Rushabh5x	Username or Password incorrect	Username or Password incorrect	Pass
2	Provide Details	Password: Rushabh5x			

### 6.3.3 PIN (3.11.2 Functional Requirement 2)

- 1) Pre-condition: The pin is 1234

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	PIN Activity	PIN: 1234	Go to the Home Activity (Home page)	Home Page	Pass
2	Provide Details				

- 2)Pre-condition: The pin is 1234

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	PIN Activity	PIN: 2345	Incorrect Pin	Incorrect PIN	Pass
2	Provide Details				

- 3)Pre-condition: The pin is 1234

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	PIN Activity	PIN: -	Enter the PIN	Enter PIN	Pass
2	Provide Details				

### 6.3.4 Change Passcode (3.11.2 Functional Requirement 2)

1)Pre-condition: The pin is 1234

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Change PIN Activity	Old Passcode: 1234	Passcode changed	Passcode changed	Pass
2	Provide Details	New Passcode: 2345			
		Repeat new passcode: 2345			

2)Pre-condition: The pin is 2345

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Change PIN Activity	Old Passcode: 1234	Wrong Passcode	Wrong Passcode	Pass
2	Provide Details	New Passcode: 2222			
		Repeat new passcode: 2222			

3)Pre-condition: The pin is 2345

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Change PIN Activity	Old Passcode: 2345	Repeat Passcode doesn't match.	Repeat Passcode doesn't match.	Pass
2	Provide Details	New Passcode: 1233			

		Repeat new passcode: 2233			
--	--	------------------------------------	--	--	--

### 6.3.5 Send Message (3.11.3 Functional Requirement 3)

#### 1) Internet connection is up and running

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Chat Activity	Message: Hi	Message is sent	Message is sent.	Pass
2	Enter Message				
3	Send				

#### 2) Internet connection is unavailable

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Chat Activity	Message: Hello	No internet connection	No internet Connection	Pass
2	Enter Message				
3	Send				

#### 3) Internet connection is up and running but the user is not signed in

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Chat Activity	Message: Hi	That user is not signed in	That user is not signed in	Pass
2	Enter Message				
3	Send				

### 6.3.6 Screenshot (3.11.4 Functional Requirement 4)

#### 1) Chat activity or Home Activity is Opened

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Chat List Fragment	User tries to take a screenshot	Screenshot is not allowed	Screenshot is not allowed	Pass

### 6.3.7 Merging Contacts (3.11.5 Functional Requirement 5)

#### 1) Home Activity is Opened

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Contact List Fragment	User's phones contacts	Merged Contacts	Merged Contacts	Pass

### 6.3.8 Logout (3.11.6 Functional Requirement 6)

#### 1) User is logged in

Step	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Home Activity	Click Logout Button	Logs out and all messages are cleared	Logs out and all messages are cleared	Pass



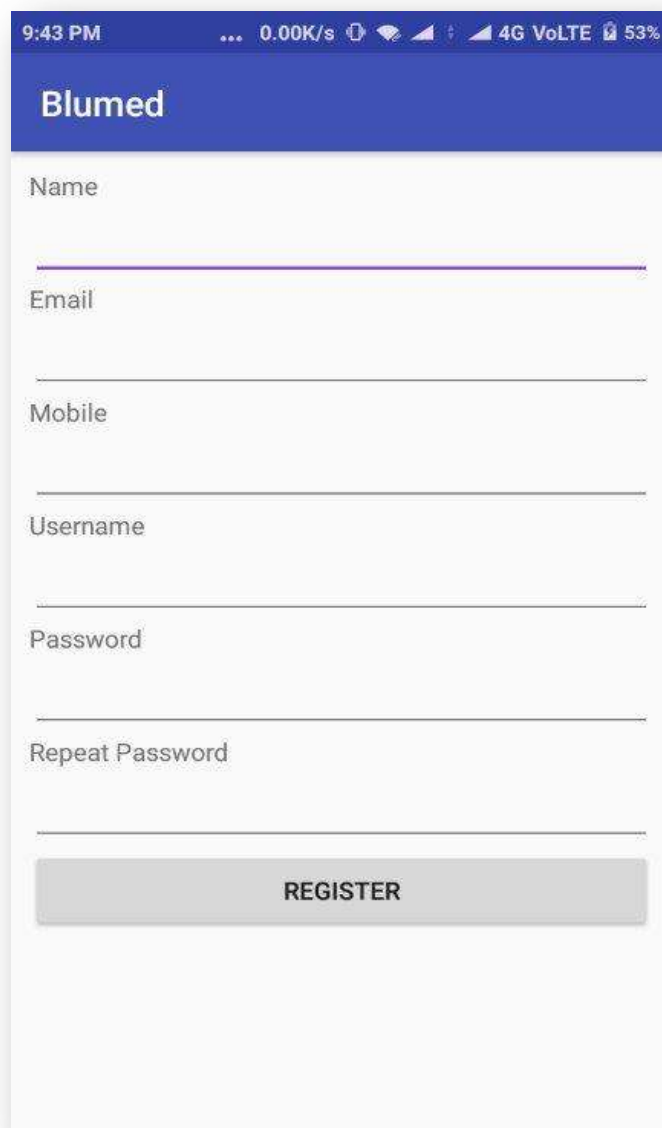
## Chapter 7

### Screenshots

---

#### 7.1 Register Activity

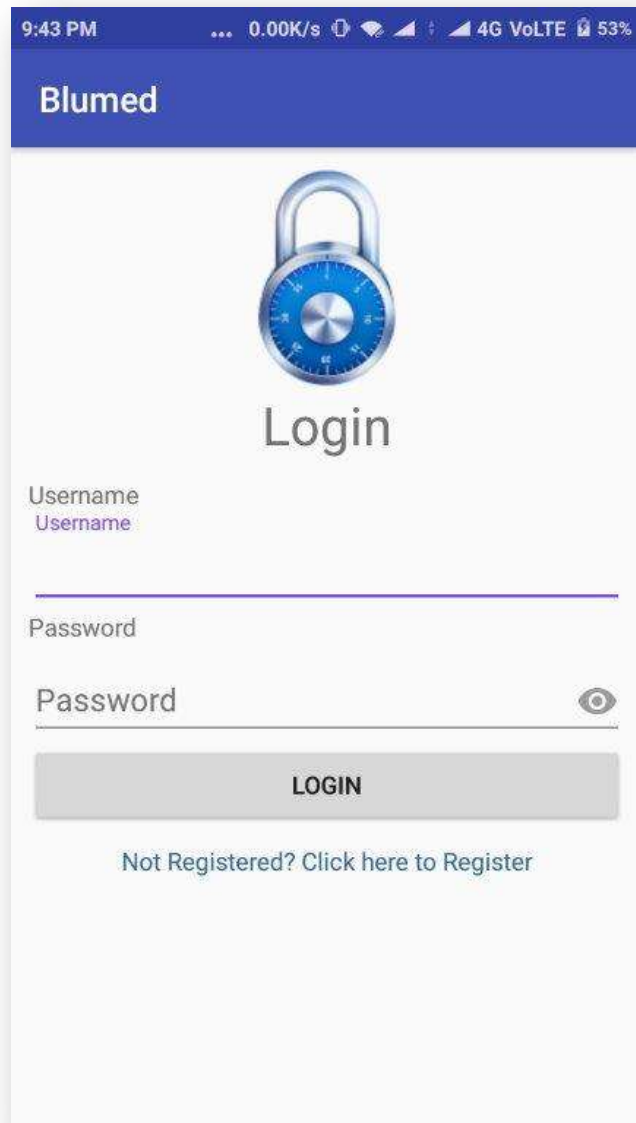
Here Users must enter all the necessary information. The username and mobile must not have been used before. On touching register button, the user gets registered to the system.



The screenshot shows a mobile application interface for a registration activity. At the top, a blue header bar displays the app name "Blumed". Below the header, the screen contains several input fields for user registration: "Name", "Email", "Mobile", "Username", "Password", and "Repeat Password". Each field is represented by a text label followed by a horizontal input line. At the bottom of the form, there is a prominent grey button labeled "REGISTER". The top status bar of the phone shows the time as 9:43 PM, network status as 4G VoLTE, and battery level at 53%.

## 7.2 Login Activity

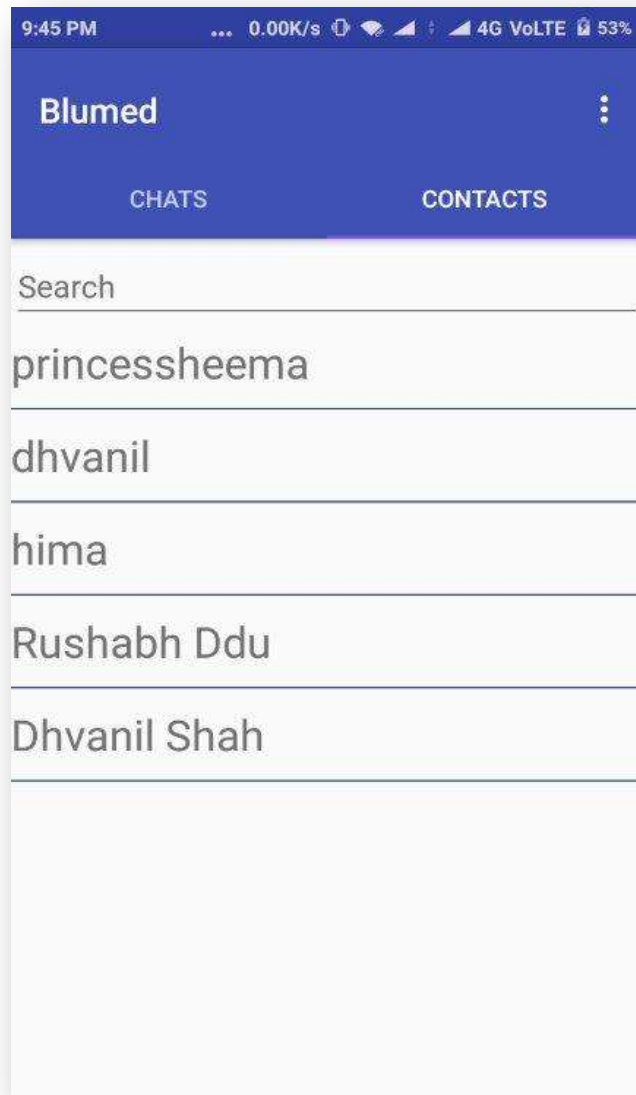
Here User must enter the username and password to Log into the system. It should be noted that this is one time process required when starting the app for first time.



A screenshot of the Blumed mobile application's login screen. The status bar at the top shows the time as 9:43 PM, a speed of 0.00K/s, and network status 4G VoLTE with 53% battery. The app's header is a blue bar with the text "Blumed". Below the header is a large blue padlock icon. Under the padlock, the word "Login" is displayed in a large, dark font. There are two input fields: the first is labeled "Username" with a placeholder "Username" and a red underline; the second is labeled "Password" with a placeholder "Password" and a red underline. To the right of the password field is an eye icon for toggling visibility. Below the input fields is a grey button with the text "LOGIN". At the bottom, there is a link that says "Not Registered? Click here to Register".

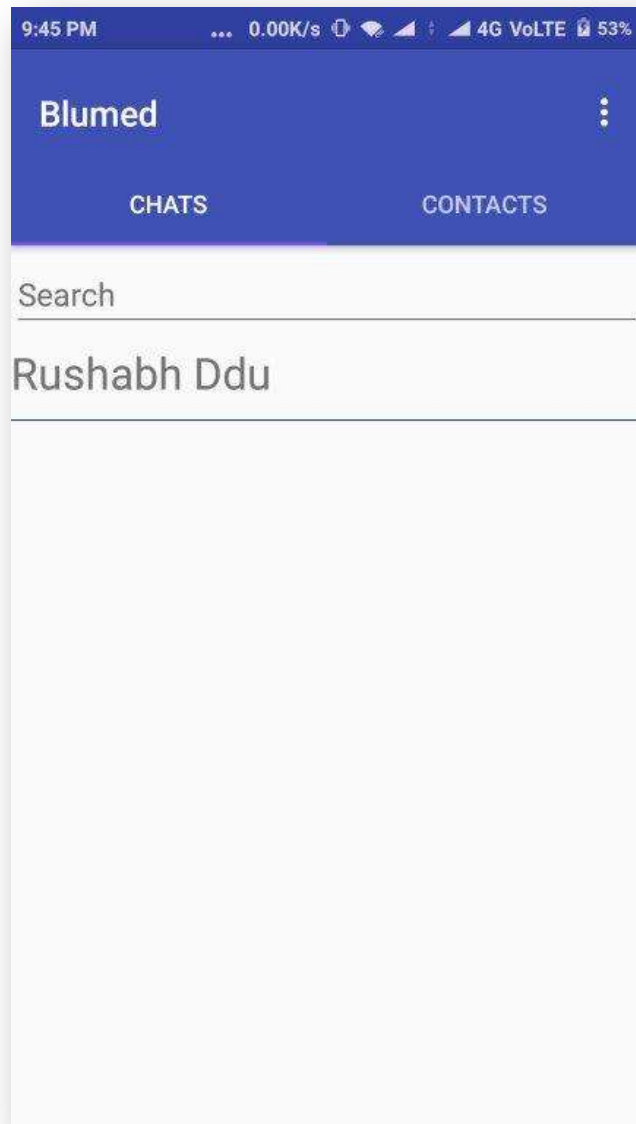
### 7.3 Contact Fragment

This Screen displays all the contacts which are registered with the System and also present in the user contact list. User can Click on any contact to start conversation with the same.



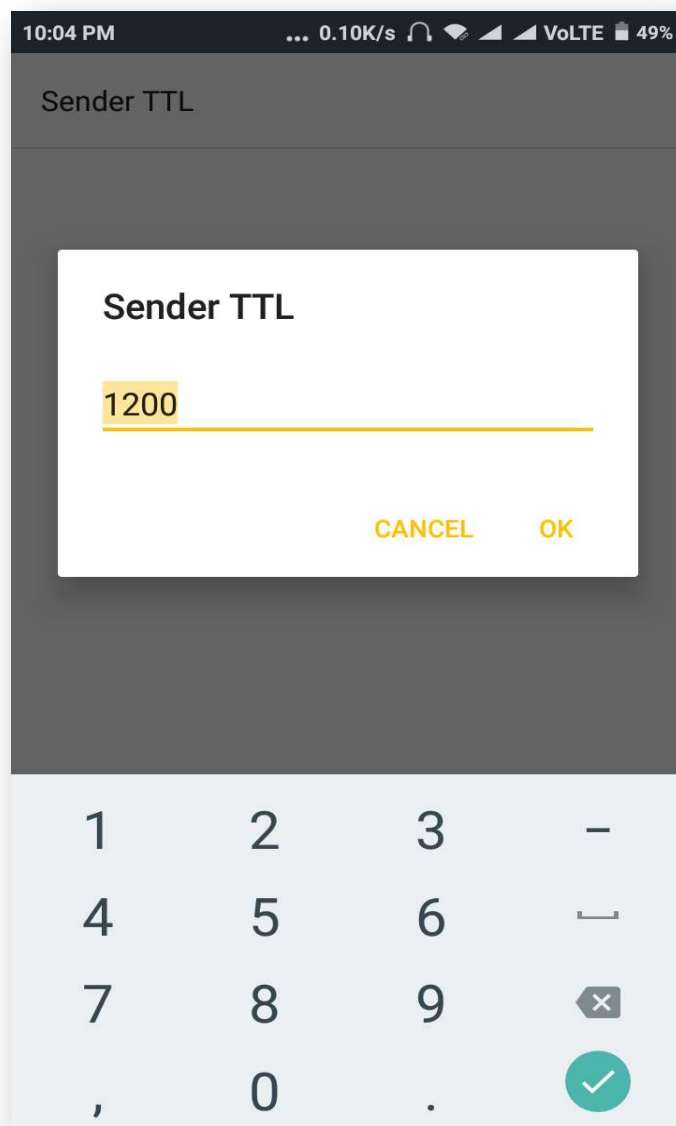
## 7.4 Chat Fragment

This screen displays all the active chats of the user. User can click on them to start the Conversation.



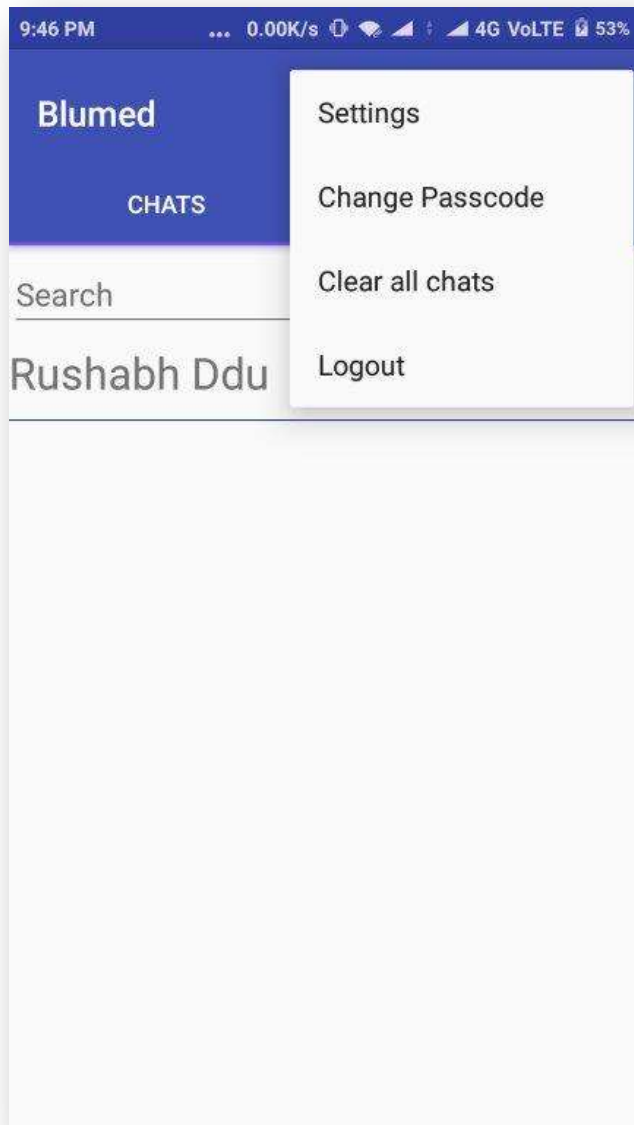
## 7.5 Settings

Here the user can set the default time in seconds after which the message sent by him or her gets deleted.



## 7.6 Menu

Here user can explore various options like Change Passcode to change the PIN lock, and Clear all chats to simply clear all the conversations. Logout may be used to Log out from the system. In this case, all the owner info associated with the app gets deleted.



## 7.7 Change Passcode

User can change his old password here...

9:46 PM ... 0.00K/s 4G VoLTE 53%

**Blumed**

**Old Passcode**

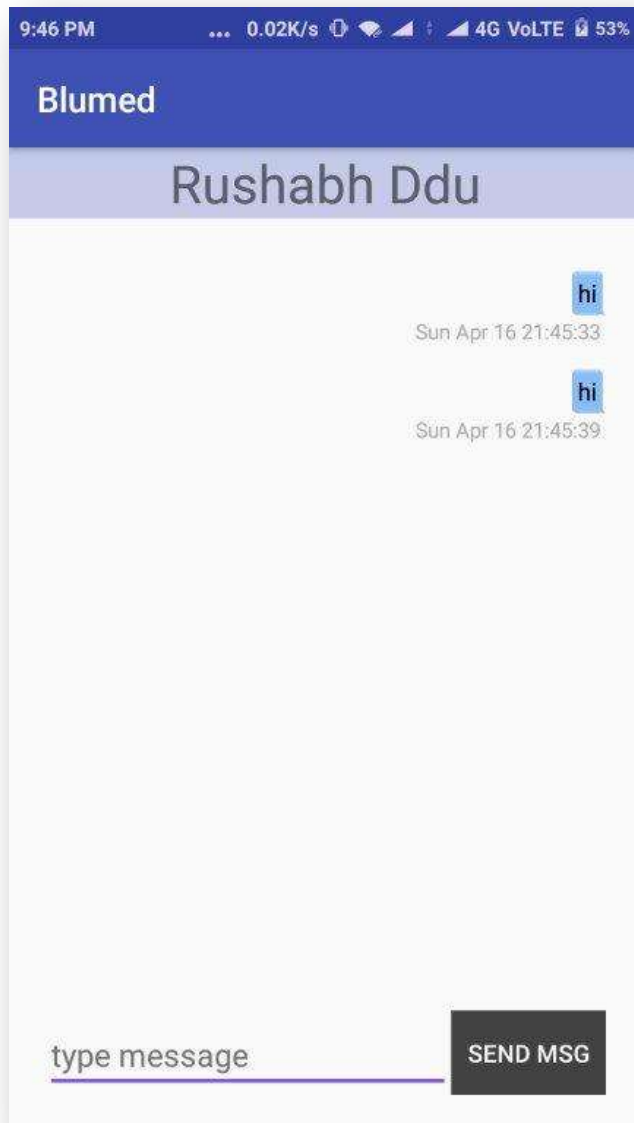
**New Passcode**

**Repeat New Passcode**

**SUBMIT**

## 7.8 Chat Activity

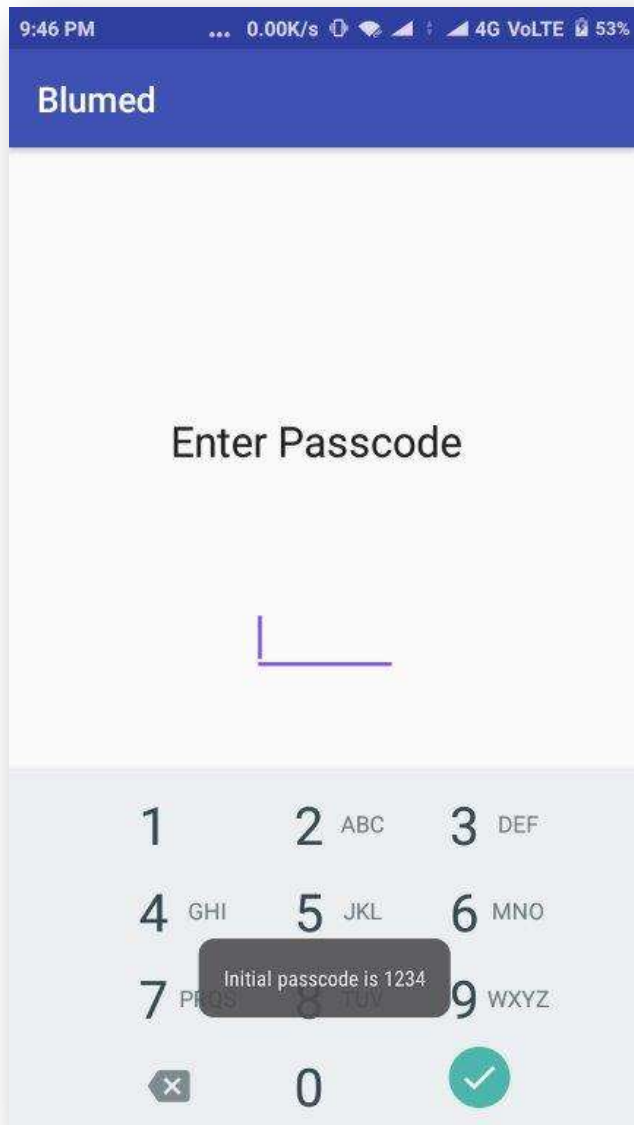
Here user can actually carry out the conversation. The message sent by the user will get deleted on the recipient's app in the time specified in the settings, after the recipient has viewed the message.





## 7.9 Passcode Screen

This is the first screen, that user will visit every time he exists the app. It is for added security to prevent intruder from using the app.



## Chapter 8

### Conclusion

---

Hereby, we declare that we have performed a project by understanding every module of this project. We checked the feasibility and requirement for this system. Then we defined overall look and flow of control among modules in paper. After this, we started actual design of our modules of system in Java. All modules of system were developed separately. Then we integrated all modules by means of control flow among all modules.

After finishing Coding and integrating of all modules, we tested all modules separately This is Unit Testing of modules. Test cases were designed by performing black box testing. Also, Alpha testing was carried out.

Thus, we were basically able to perform all the functions mentioned in the Functional Requirements along with the enhanced security.

## Chapter 9

# Limitation and Future Extensions

---

### 9.1 Limitation:

- 1) Anyone who has web address of server can send message to user by sending REST message to server which will not be encrypted and user will get a spam message. Which will not be shown to his chats but his application will get a false notification.
- 2) Also, there may be possibility of some performance enhancement in the speed of the app.

### 9.2 Future Extension:

- 1) Provide support for photos, audios, videos and documentation.
- 2) Contacts with photos in user interface.
- 3) Web Application or Mobile Application in other platforms.

## Chapter 10

### Miscellaneous

---

#### 10.1 JSON (JavaScript Object Notation)

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON is built on two structures:

- A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
- An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

These are universal data structures. Virtually all modern programming languages support them in one form or another. It makes sense that a data format that is interchangeable with programming languages also be based on these structures.

In JSON, they take on these forms:

An *object* is an unordered set of name/value pairs. An object begins with { (left brace) and ends with } (right brace). Each name is followed by : (colon) and the name/value pairs are separated by , (comma).

A string is a sequence of zero or more Unicode characters, wrapped in double quotes, using backslash escapes. A character is represented as a single character string. A string is very much like a C or Java string.

A number is very much like a C or Java number, except that the octal and hexadecimal formats are not used.

Examples:

```
1. { "name" : "John" }
```

```
2. {"employees":[
    { "firstName":"John", "lastName":"Doe" },
    { "firstName":"Anna", "lastName":"Smith" },
    { "firstName":"Peter", "lastName":"Jones" }
  ]}
```

The file type for JSON files is ".json"

The MIME type for JSON text is "application/json"

## 10.2 Session Management in Android

### 10.2.1 Shared Preference

Android provides many ways of storing data of an application. One of this way is called Shared Preferences. Shared Preferences allow you to save and retrieve data in the form of key, value pair.

In order to use shared preferences, you have to call a method `getSharedPreferences()` that returns a `SharedPreferences` instance pointing to the file that contains the values of preferences.

Shared Preference can also be used for session management purpose to save some important data.

Example:

```
SharedPreferences sp = PreferenceManager.getDefaultSharedPreferences(this);
sp.edit().putString("private_key",key).apply();
mynumber = sp.getString("mynumber", "");
rpctl = sp.getString("pref_sender_ttl", "");
```

## 10.3 GitHub

**GitHub** is a web-based Git or version control repository and Internet hosting service. It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project.

GitHub offers both plans for private and free repositories on the same account which are commonly used to host open-source software projects.

Projects on GitHub can be accessed and manipulated using the standard Git command-line interface and all of the standard Git commands work with it. GitHub also allows registered and

non-registered users to browse public repositories on the site. Multiple desktop clients and Git plugins have also been created by GitHub and other third parties that integrate with the platform.

The site provides social networking-like functions such as feeds, followers, wikis (using wiki software called Gollum) and a social network graph to display how developers work on their versions ("forks") of a repository and what fork (and branch within that fork) is newest.

A user must create an account in order to contribute content to the site, but public repositories can be browsed and downloaded by anyone. With a registered user account, users are able to discuss, manage, create repositories, submit contributions to others' repositories, and review changes to code.

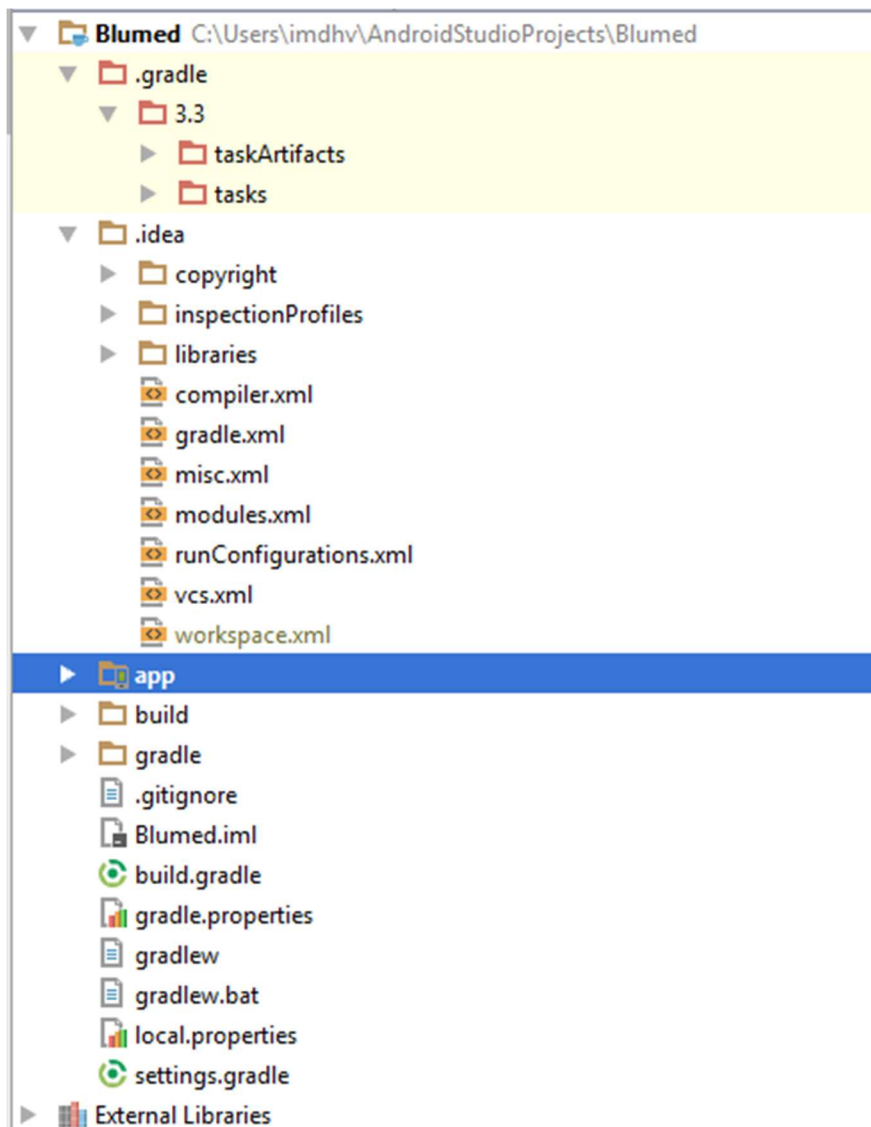
GitHub is mostly used for code.

In addition to source code, GitHub supports the following formats and features:

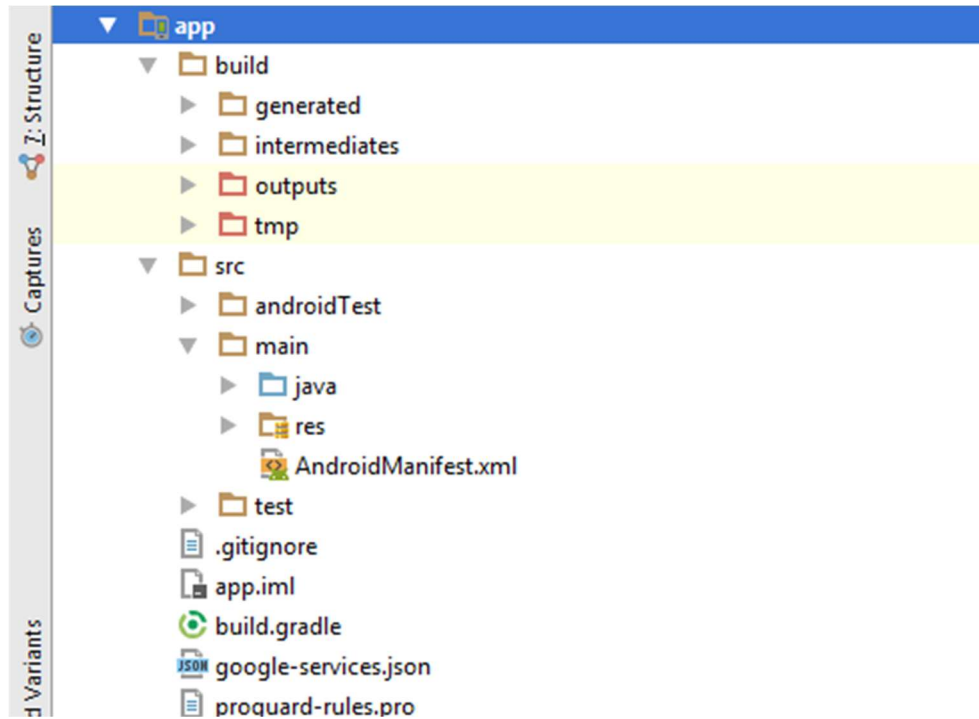
- Documentation, including automatically rendered README files in a variety of Markdown-like file formats (see README files on GitHub)
- Issue tracking (including feature requests) with labels, milestones, assignees and a search engine
- Wikis
- Pull requests with code review and comments
- Commits history
- Graphs: pulse, contributors, commits, code frequency, punch card, network, members
- Integrations Directory
- Unified and split diffs
- Email notifications
- Option to subscribe someone to notifications by @ mentioning them.
- GitHub Pages: small websites can be hosted from public repositories on GitHub. The URL format is `http://username.github.io`.
- Nested task-lists within files

## 10.4 Source Package/Directory Structure

### 10.4.1 The Overall Directory Structure

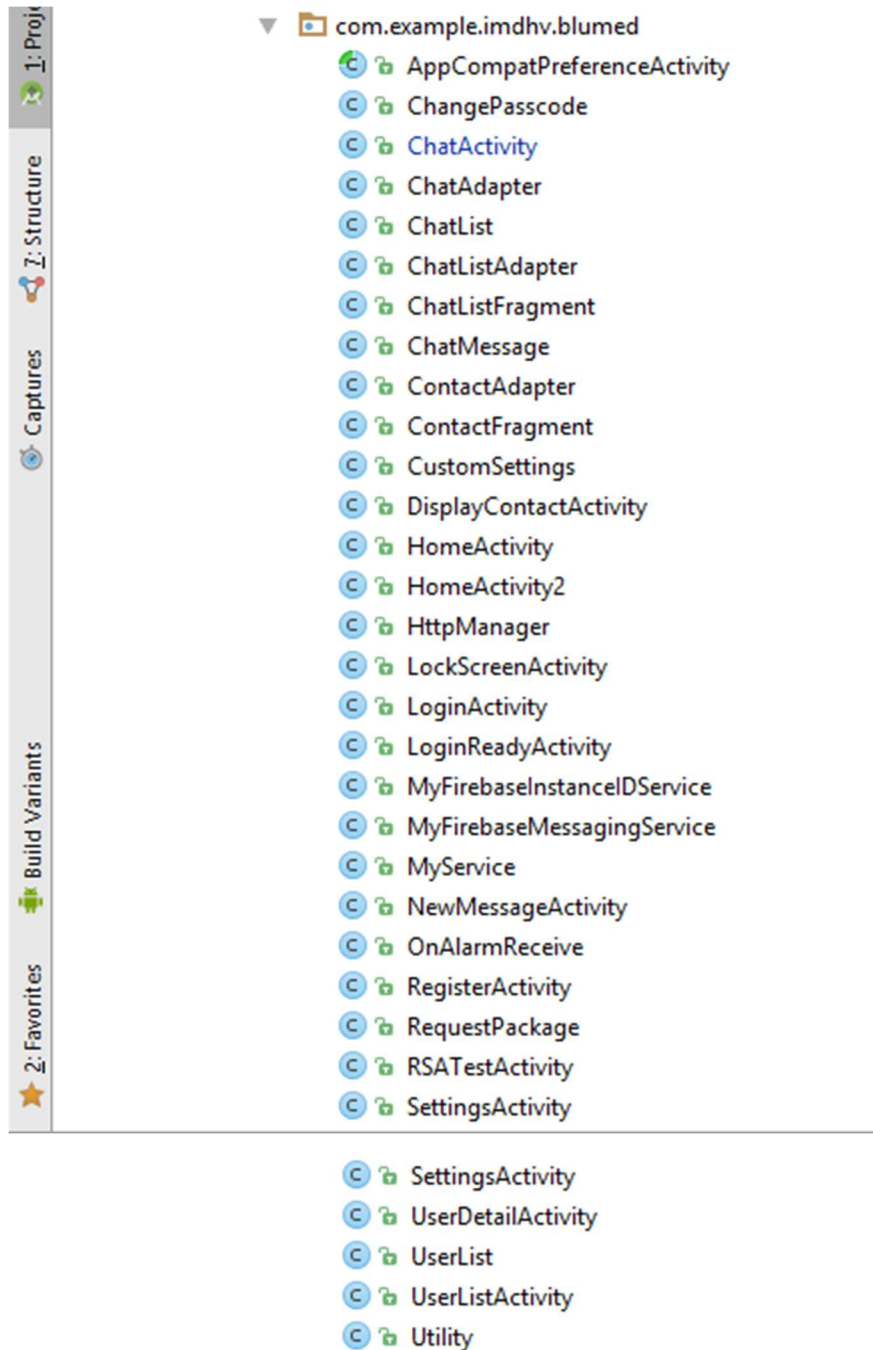


### 10.4.2 The Structure of app folder





### 10.4.3 The structure depicting the main files developed by us..



## 10.5 Software Version Description Document

This is to document the following steps which will be required for installing the application for the first time:

Basic Requirement in the System:

1. Android Studio
2. Xampp
3. Internet connection
4. Android Phone
5. Android ADB

We have to follow this procedure for installation of web service on local pc.

1. For connection to server with PHP services we have to enter following code.

```
<?php
$db_name = "blumeddb";
$mysql_username="root";
$mysql_password="";
$server_name="localhost";
$conn = mysqli_connect($server_name,$mysql_username,$mysql_password,$db_name);

if($conn)
{
    //echo "Connection Successfull!";
}
else
{
    //echo "Connection Fail!!";
}
?>
```

2. For connection of application with web service you have to add web address of web service on android app.

```
public static final String serverurl =
"https://blumed.000webhostapp.com/androidsupport.php";
```

## 10.6 Deployment Procedure

To deploy this Application in Android Phone follow this steps

- 1) On ADB Debugging Settings in your phone
- 2) Connect your phone to PC via micro USB cable.
- 3) In Android Studio On run command select connected phone and press ok.

To Deploy PHP Server to Web Hosting website

- 1) Register on Web Hosting Website (000WebHosting.com)
- 2) Export Database and use it to create database on Hosting Site.
- 3) Create a PHP web application and add required files to web application
- 4) To connect with online database add following file.

```
<?php
$db_name = "id1144005_blumeddb";
$username="id1144005_dhvanil1976";
$password="abc123";
$server_name="localhost";
$conn = mysqli_connect($server_name,$username,$password,$db_name);

if($conn)
{
    //echo "Connection Successfull!";
}
else
{
    //echo "Connection Fail!!";
}
?>
```

- 5) Start web application.

## Bibliography

### **Sources for Knowledge:**

Tutorials Point

Stack Overflow

Developer.Android.com