Journal of Current Research in Engineering and Science
JCRES
Bi-Annual Online Journal (ISSN : 2581 - 611X)

Volume 7- Issue 1, January 2024
Paper 81

MESH MESSAGING SERVICE USING BLUETOOTH

Aswini.J
Computer Science and Engineering
Sree Vidyanikethan Engineering College
Tirupati, India
aswini.j@vidyanikethan.edu

P. Hitesh Reddy
Computer Science and Engineering
Sree Vidyanikethan Engineering College
Tirupati, India
19121A05H1@vidyanikethan.asia

R.B.Moulya
Computer Science and Engineering
Sree Vidyanikethan Engineering College
Tirupati, India
19121A05J6@vidyanikethan.asia

P. Ruchitha
Computer Science and Engineering
Sree Vidyanikethan Engineering College
Tirupati, India
19121A05J4@vidyanikethan.asia

P. Ashish
Computer Science and Engineering
Sree Vidyanikethan Engineering College
Tirupati, India
19121A05G0@vidyanikethan.asia

S.Amisha
Computer Science and Engineering
Sree Vidyanikethan Engineering College
Tirupati, India
19121A05N2@vidyanikethan.asia

***Abstract*—**
The idea is to create a service that enables users to communicate with other users using a Mesh Network established between their devices. The Mesh network is to be built using the Bluetooth service available in the user devices. There are apps built using this technology such as Bridgefy, Mesh Messaging, Fire chat, etc. But the proposed application enables users to send messages to other users through the Bluetooth network established between the user devices. The application does not require any Internet connection. The idea is to transfer data from the sender to the receiver where it finds its way using a Dynamic routing algorithm. The middle devices (user devices that are not participating in the communication) act as routers in the network. The app encrypts the messages which are sent from the sender so that the middle devices cannot see them.

Keywords—Mesh Network, Blue Tooth, Routing, Data Transfer

I. INTRODUCTION

Bluetooth is an open standard for implementing short-range wireless communication. The data is transferred through the network with the help of mesh networking technology. The working principle is that the devices that take part in the network act as the endpoint devices as well as the routers of the network. The more the number of devices, the better the functionality of the network. It is deployed with a Distance Vector Routing algorithm, which will be customized to work with Bluetooth technology. To protect the messages from the third persons it will be deployed with a strong and efficient Encryption algorithm (Advanced Encryption). App has been built in VS code using Flutter for Android devices. With the developed application the users can message even without an internet connection or 2G, 3G, or 4G network coverage using Bluetooth. One of the most available modes of communication is messaging over either the internet or mobile network. Over 6 billion texts are sent every day. Platforms like Whats App, Twitter, Instagram have become the primary source of information through communication. Those are enabled through either an Internet connection or through a Mobile Network.

The proposed idea aims to provide an alternate way for communication using Bluetooth. It is to build an app that lets the user send messages without any of the regular connectivity. The users can connect to each other directly and pass messages over the network they create. The messages sent over this network hop across it until it reaches its destination. This service is also entirely free of cost. The effectiveness of this idea increases as the number of active users increases. Objective of this work is to designing a clean and simple user interface, so the users can use the application with ease. Also adding mobile number authentication during the initial run of the application, to verify the legitimacy of the users. Added with this to create a database to store the data locally on the user device, so that it can be available when offline. To implement an asymmetric encryption service to ensure that the privacy of the users is respected. To add a functionality which enables the users to start a new conversation by scanning a QR code on the other user's application. To design a dynamic routing algorithm that can respond to the constant changes in the topology of the network. Finally to prevent clogging up of the network by duplicate packets, adding a time to live for the network packets.

II. LITERATURE SURVEY

Bluetooth Low Energy: Gomez, Carles & Oller Bosch, Joaquim & Paradells, Josep. (2012). Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology. Sensors (Basel, Switzerland Bluetooth Low Energy or BLE is a wireless local area network technology that consumes lesser power than traditional Bluetooth technology. The devices that intend to transfer data through a BLE connection must first create a channel. Android provides APIs that applications can use to discover nearby devices, query for services, and transmit information. Mesh Network: Lee Barken et al [2]. A Mesh network is a kind of network where the devices that participate in the communication also act as the routers for the network. This allows various devices to link together branching off each other allowing the signal and network to spread further. There are two types of mesh networks: wired mesh networks and wireless mesh networks. Wireless mesh networks can easily and effectively connect large areas using inexpensive, existing technology. In this kind of network, the network connection is spread out among various wireless mesh nodes that "talk" to each other to share the network connection across a large area. RSA Encryption: Xin Zhou and Xiaofei Tang, "Research and implementation of RSA algorithm for encryption and decryption [3]. RSA algorithm is an asymmetric cryptography algorithm.

Asymmetric actually means that it works on two different keys i.e., public key and private key. As the name describes that the public key is given to everyone and private key is kept private. The idea of RSA is predicated on the very fact that it's tough to factorise an outsized whole number. the general public key consists of 2 varietys wherever one number is multiplication of 2 massive prime numbers. and personal secret's conjointly derived from identical 2 prime numbers. So, if someone will factorise the big variety, the personal secret's compromised. Therefore, encryption strength totally lies on the key size and if we double or triple the key size, the strength of the encryption increases exponentially. RSA keys is usually 1024 or 2048 bits long however specialists believe that 1024-bit keys may well be tame the close to future. however until currently it appears to be associate unworkable task.

The security of RSA relies on the practical difficulty of factoring the product of two large prime numbers. Distance Vector Routing [4]. A distance vector routing protocol necessitates that a router update its neighbors of topology variations every so often. Historically known as the old ARPANET routing algorithm or the Bellman-Ford algorithm. Bellman Ford Basics - each router maintains a distance vector table containing the distance between itself and all possible destination nodes. Distances, based on a selected metric, area unit computed victimization info from the neighbors' distance vectors. related to every link connected to a router, there's a link value. A router transmits its distance vector to each of its neighbors in a routing packet. Each router receives and saves the most recently received distance vector from each of its neighbors. A router recalculates its distance vector when it receives a distance vector from a neighbor containing different information than before or when it discovers that a link to a neighbor has gone down.

III. IMPLEMENTATION DETAILS

The Android application provides a complete user interface for the users to send messages, receive messages, read the past messages and also manage all the conversations they have. The application was built with the Flutter toolkit and Dart libraries [5][6].

At the initial run of the application, the users are prompted to provide their mobile number, which will be used to create an account for them with the help of Firebase Phone Authentication. Upon a successful account creation, the application generates an RSA key pair. This is done with the help of the Pointy Castle dart library. Further details on how the keys are generated are provided in the Implementation chapter of this report.

Also, a shared preference is created to store the login state of the user, so that the next time the user opens the app, it automatically logs them in. As the application is intended to work offline, it stores all the data locally on the user's device. It does so by creating a local database on the user device. The application uses the SQLite dart library to create the database. The database maintains two tables, one for the conversations and one for the messages. A detailed relational model of the database is explained in the Design chapter of this system.

To start a conversation, the user must scan a QR code, which is uniquely generated for each user at the time of their account creation, of the other user. The QR code is encoded with a string that is a concatenation of both the RSA public key of the user, their Bluetooth address, their username and their mobile number. Upon scanning a QR code a conversation is created in the user's conversation list and also the details of the other person are stored into the local database. To send a message to a contact, the user taps on their conversation and types the message into the message field located at the bottom of the screen. When the user taps on the send button, the process of message transmission begins. This process includes a series of steps as: encryption, routing in the network, decryption. Also, the message is stored into the local database along with its details.

The encryption involves using the public key of the receiver to convert the plain text of the message to cipher text. The address of the receiver is appended to the processed message. The routing part involves looking for the Bluetooth address of the receiver in the nearby devices table. This table contains a list of all the nearby devices. If the address of the receiver is not in the list, then the processed message is flooded to all the nearby devices. The nearby devices then look for the address of the receiver in their tables. This flooding process is repeated at all the intermediate devices until the message reaches the receiver. To prevent the clogging of the network from all the flooded messages, we specify the time to live for all the messages. Once this time elapses, the message packet is destroyed[7].

The time that a message takes to reach the destination depends on the distance between the source and destination, and the number of hops the message takes along the network. The time of transmission between two immediate nodes of network is measured in real time, although it can be formulated as in equation

When there are three nodes involved in the transmission, the time is expected to increase two-fold, since the number of times the message is being processed has increased. application decrypts it by using the receiver's private key. This yields the text of the actual message. This message is then added to the local database of the receiver and also displayed in the corresponding conversation.

The database is designed to have two tables. The first table is to store the conversations and its details. The second table stores the messages, both sent and received, along with their details.

The attribute 'id' acts as the primary key for the 'persons' table. This is an arbitrary integer value, and it has the auto increment property, which means its value automatically increments for each new row added to the table. All other attributes of this table are of string type.

In the 'messages' table, the 'id' attribute acts as the primary key. This attribute is similar to the one in the 'persons' table. The 'recipient id' attribute is a foreign key that references to the 'id' attribute of 'persons' table. All other attributes of this table are of string type. The application uses the following Dart libraries i) firebase_core 1.10.0 ii) firebase_auth 3.2.0 iii) sqflite 2.0.0+3 iv)pointycastle 1.0.0-rc4 v) asn1lib 1.0.3 vi) shared_preferences 2.0.11 .These dependencies must be specified in the 'pubspec.yaml' file in the flutter application root directory. firebase_core and firebase_auth libraries add the authentication functionality to the application. sqflite library is used to implement the local database for the application. pointycastle and asn1lib libraries provide the encryption services for the application[10]. Shared preferences library lets the application to create shared preferences, which stores small amount of primitive data as a key/value pair.

Table 3.1 Application Screens and their files

Screen name	File name
Welcome Screen	welcome_scren.dart
Registration Screen	signin.dart
Conversations Screen	chats.dart
Chat Screen	chat.dart
Edit Profile Screen	edit_profile.dart
Settings Screen	setting.dart
Profile Screen	profile.dart

Additionally, there are other dart files that provide functional support to the files that are mentioned in Table 3.1. These files and their functions are specified in Table 3.2.

Table 3.2 Back-end code files and their functionalities

File name	Function
main.dart	It contains the dart script that runs the application.
constants.dart	It contains the pre-defined values such as theme data and also definitions of custom data types.
chat_database.dart	It contains the definition of the SQLite database that is used by the application and also the methods to perform operations on the database.
encrypt_service.dart	It contains the methods that are used to generate RSA key pairs,
router.dart	It contains the routing table

The application starts from the ‘main.dart’ file, which is basically is runner code for the application. It returns a MaterialApp object, which actually is the main application.

At startup, the application starts in the welcome screen While in this page, the application checks if the ‘LoggedIn’ shared preference has been set or not. If it is set, then the application navigates the conversations page, else the application navigates to the registration screen.

In the registration screen, the user interface consists of a text field that accepts mobile numbers. Once the user enters their mobile number and taps on the submit button, an OTP will be sent to the provided mobile number and a new text field to enter the OTP appears on the screen. If the user enters a valid OTP and taps on the verify button, their login state will be persisted by setting a shared preference name ‘LoggedIn’. Upon successful authentication, the application navigates to an empty conversations screen.

To start a new conversation, the user must tap on a floating action button at the bottom right of the screen, which opens the camera of the device. The user must scan the QR code of the other user with whom they intend to start a new conversation, which is available in the settings page of the application. To navigate to the settings screen, the user can tap on the settings icon available at the top right of the application screen. The QR code encodes a string which contains the Bluetooth address of the user, RSA public key, mobile number, username[8]. Scanning the QR code decodes the string and adds the conversation into the ‘persons’ table in the database. The display picture for this conversation must be manually set by the user, else a default display picture will be displayed.

The edit profile screen can be opened by tapping on the profile icon in the appbar of the conversations screen. In the edit profile screen, the user is provided with the ability to change their display picture, username, and add some lines about them. To send a message, the user must tap on the conversation of interest, which navigates the application to the chat screen. The chat screen consists of an appbar which displays the username and display picture of that particular conversation, along with a back button. The main body of the chat screen is a list that displays the messages of the conversation. The sent messages are displayed to the right of the screen, while the received messages are displayed to the left of the screen[9]. They are further differentiated by having a different bubble color. At the bottom of the chat screen is a bar with the text field to type the message in, and the send message. The user can type the message in the text field and tap on the send button to start the process of transmitting the message to the recipient. Tapping on the appbar navigates the application to the profile screen.

Volume 7- Issue 1, January 2024
Paper 81

The profile screen displays the display picture of the conversation, the username, mobile number and its Bluetooth address.

The database consists of two tables, one to store the conversations and one to store the messages. The table that stores the conversations is named as *‘persons’* and the table that stores the messages is named as *‘messages’* in the database.

The methods that are used to perform operations on the database are listed in Table 3.3.

Table 3.3 Database methods

Method	Parameters	Returns	Functionality
createPerson	A Person object	id of the row	Inserts a row into the ‘persons’ table
removePerson	id of the person	None	Removes a row from the ‘persons’ table
readPerson	mobile number of the person	Person object	Selects a row from the ‘persons’ table
readAllPersons	None	List of Person objects	Selects all rows from the ‘persons’ table
postMessage	A ChatMessage object	ChatMessage object	Inserts a row into the ‘messages’ table
deleteMessage	id of the message	None	Removes a row from the ‘messages’ table
readMessage	id of the message	ChatMessage object	Selects a row from the ‘message’ table
readMessages	id of the recipient	List of ChatMessage objects	Selects all rows of a particular conversation from the ‘messages’ table
getLastMessage	id of the recipient	content of a ChatMessage object	Retrieves the content the last message of a particular conversation from the ‘messages’ table

deleteAllMessages	id of the recipient	None	Removes all rows of a particular conversation from the 'messages' table

Encryption:

At the time of account creation, the application runs a piece of code that initializes the encryption service for the application. It generates a RSA key pair and stores them with shared preferences. The public key is also used to generate the QR code for the user. Firstly, the parameters for the key generator are initialized.

Routing:

After the encryption service generates the cipher text, it passes it to the routing mechanism of the application. The routing mechanism of the application has three jobs to do:

1. Crafting the network packet.
2. Searching for the recipient in its routing table.
3. Forwarding the packet to its successor node(s).

Along with these jobs, the mechanism constantly updates its routing table by scanning for new Bluetooth devices in its range.

The structure of the network packet that is used in this mechanism is described as follows:

Destination address + Cipher text + Sent time + Time to live

The destination address part of the packet is the Bluetooth address of the recipient. If the address is not available in the current device's routing table, it means the recipient is not the range of the device. The device then floods the packet to all the devices in its routing table.

The cipher text part of the packet contains the actual content of the message in its encrypted format. In a routing algorithm where flooding is involved, it is customary to include the packet's time to live. This ensures that the network is not clogged by duplicate copies of a packet that could not reach its destination. Every device that the packet arrives at checks the sent time and time to live of the packet. If the sum of the sent time and time to live exceeds the current time, it means that the packet's lifetime has expired. The packet is then discarded. If the packet still has time left, it is again forwarded to the successor nodes of the network. When a packet reaches its destination, the routing mechanism extracts the cipher text from the packet and forwards it to encryption service of the application.

IV. RESULTS AND EVALUATION

Volume 7- Issue 1, January 2024
Paper 81

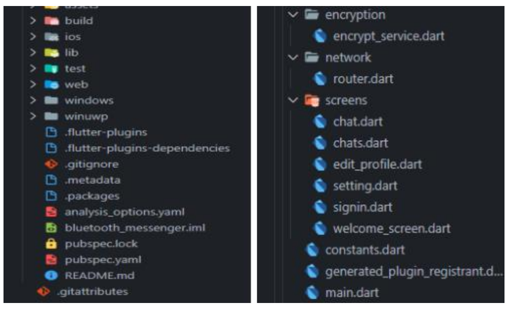
The application is developed with the Flutter toolkit and is programmed in Dart programming language. Both the frontend and backend of the application is written in Dart language. The details the environment in which we developed the application is as follows:

● Visual Studio code with Flutter and Dart plugins as the text editor.

● Android emulator provided by the Android Studio SDK (Pixel 4XL API 30 running Android 11.0).

To run the application on an emulator, create a virtual android device. This can be done with the help of AVD manager in Android Studio. Choose any of the provided real-world smartphones to create a virtual android device of it.

To run the application for testing, open the root directory of the flutter project in VS code. The editor must also be able to detect the presence of any created virtual devices on the computer. It provides a list of available virtual devices for us to choose one from. Once a virtual device has been chosen, it is booted and connected to the editor. The directory structure of the project is as mentioned in Figure 4.1.


```
> build
> ios
> lib
> test
> web
> windows
> winuwp
  .flutter-plugins
  .flutter-plugins-dependencies
  .gitignore
  .metadata
  .packages
  analysis_options.yaml
  bluetooth_messenger.iml
  pubspec.lock
  pubspec.yaml
  README.md
.gitattributes
```

```
v encryption
    encrypt_service.dart
v network
    router.dart
v screens
    chat.dart
    chats.dart
    edit_profile.dart
    setting.dart
    signin.dart
    welcome_screen.dart
  constants.dart
  generated_plugin_registrant.d...
  main.dart
```

Figure 4.1 Directory structure

The main code of the application is in the /lib/ directory in the root directory of the flutter project. The file named ‘main.dart’ is where the application starts from. Debugging this file runs the application on the virtual device in debug mode. The application can be used in the same way one would do on a real smartphone. Testing: The entire application was thematically divides into 4 modules: ● Front-end ● Database ● Encryption Service ● Routing .

Each of these modules were tested individually to verify if they work as intended. The testing was done by running the app on the Android Virtual Device. The Front-end module consists of the entire user interface of the applications. The application has been tested on different screen sizes to ensure that the application UI is adaptive and looks as imagined.

The Database module was tested by performing all the operations on the SQLite database and verifying if those operation have the desired effect, on the database. Encryption Service was tested by

Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)

building some test cases where plain text is passed to it and the output was the encrypted cipher text and the decrypted plain text.

The routing module was tested by sending dummy packets over the network and observing how they were distributed. A test was performed to check if the packets were actually reaching the intended destination. All these modules were integrated into the final application and were tested again, to check if they work as intended when integrated with the other modules.

The test we conducted on the application yielded the following results.

Front-end test result:

The UI of the application is responsive and simple to read. The application was run on different screen sizes to check if the elements of the application are sized correctly and it was found that the layout of the application fits as intended on every screen size.

Database test result:

The local database that was implemented for the application successfully created the relational model that was designed for the application. The operations on the database also returned expected results. The runtime of all of the operations is fairly quick. The initialization of the database during the application startup took a bit longer than estimated, but this issue is not experienced in the consequent application launches.

Encryption test result:

The RSA encryption algorithm is known to be a slow encryption algorithm. A test was conducted to observe the time it takes to generate the RSA key pair and also the time it takes to encrypt and decrypt text. The generation of the key pair takes a significant amount of time, but since it happens only one time for every user, it was decided not to optimize it. The time to encrypt and decrypt texts is insignificant.

Routing test result:

The test that was conducted on the routing mechanism verified if the packets reach their destination, the time it takes for the packet to reach its destination, and how these factors change with the distance the communicators.

The results that were observed in the test are tabulated in Table 4.1. It was observed that when under 10 m, the message is transmitted directly from the sender device to receiver device, but for a distance greater than 10m, a third device had to act as the middle device. The inclusion of a third device increases the processing time two-fold.

Table 4.1 Routing test result

Distance	Time	Reachability
1 m	500 ms	Yes
2 m	600 ms	Yes
5 m	800 ms	Yes
10 m	1000 ms	Yes
11 m	1500 ms	Yes
12 m	1600 ms	Yes
15 m	1800 ms	Yes
20 m	2000 ms	Yes

V. APPLICATIONS

1. Natural Calamities:

During the aftermath of a natural disaster, the basic means of communication might not be available, since there is a high probability that the signal towers, cables, and routers are damaged. At times like these, a service like the proposed one can come in handy and might help save lives.

2. Public protests:

During the recent Hong Kong protests the government took down internet services to prevent the protesters from communicating. To overcome this problem the protesters used similar technology to create a platform of their own to plan and organize themselves.

3. Remote locations:

People that live in remote areas like hill stations, or in the middle of a forest might not always have a network reception and neither do they have access to proper internet. This not only cuts them off from the rest of the world but also prevents them from communicating among themselves. This idea can help them establish a network within their community and use it to communicate with ease.

4. Closed communities:

In schools or offices, creating a network only for themselves saves a lot of time that might otherwise be spent walking corridors or climbing stairs.

5. Unavailability of other media:

Everyone may have faced this situation when someone is trying to send some important message (like office messages) to another, suddenly their carrier signals may be lost or get a low signal that leads to frustration then This application becomes handy.

This project model is primarily targeted on the populations that have a difficulty in communicating through the conventional media of communication. This can prove to be very useful where there is little to no carrier signal and internet service. Although, this application can be used by anyone as it has very little dependency on the external factors and is also free of cost when compared to the other modes of communication. Bluetooth uses short-wavelength UHF radio waves. These waves can easily be disrupted or blocked by solid objects. This leads to the inability of the connection between devices. Limitations of this work is that an application must keep checking for changes in the network topology to update its routing table. This requires a decent amount of constant memory utilization as well as battery power. When it is considered as performance, speed of the delivery of the message entirely depends upon the availability of middle devices (more users). This uncertainty of the delivery of the message is a drawback to this model. Since the networks are formed entirely between the user devices, the usage of this model can lead to multiple networks that are unrelated to each other when two populations are divided by a distance. A nnetwork doesn't exactly specify the location of the receiver, the routing algorithm must implement the flooding technique, which has a reputation of clogging up the network if not optimized.

CONCLUSION

The proposed system was built based on the experiences and situations faced by different people in the society in the real time while sending messages that extend providing communication at the time of natural calamities, providing communication in remote places, and providing communication in the closed places. So, the complete ideology was built to solve this problem and built this application on it. This application has been built using flutter framework and Dart Programming Language in the view of extending it to the iOS in future. This application uses QR code to get the details of the receiver and save their chat ID and uses Distance Vector routing algorithm for the routing of messages between the devices. This application uses RSA encryption algorithm to make messages secured using encryption and decryption mechanisms. This application would be widely useful for all the people and make them overcome the problem faced while transmitting messages. Future work of this system are discussed as follows: i) The application is now currently deployed for the Android OS only. This code can be reused to deploy the app for the iOS environment too. Ii) The application can be released into the market of the already existing messaging platforms like WhatsApp messenger, Facebook Messenger, Instagram messenger etc.iii) A situation where the users are disconnected from the communication network either because of lack of connectivity or a server-side issue of the application can cause disruption in the society, when using the regular messaging platforms. This issue can be solved by creating an API which can create Bluetooth mesh networks between the user devices when there is no other form of connection. The backend of this application can be used to develop such an API.iv) With the proposed API, networks an be established without the internet. This API can also be developed in such a way that it

Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)

Volume 7- Issue 1, January 2024
Paper 81

toggles between the way the network is established – network through the internet, when an internet connectivity is available or Bluetooth mesh network when there is no internet connectivity.

REFERENCES

[1] Gomez, Carles & Oller Bosch, Joaquim & Paradells, Josep. (2012). Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology. Sensors (Basel, Switzerland). 12. 11734-53. 10.3390/s120911734.

[2] Lee Barken, Eric Bermel, John Eder, Matthew Fanady, Michael Mee, Marc Palumbo, Alan Koebrick, Wireless Hacking, Syngress, 2004, Pages 215-224, ISBN 9781931836371.

[3] Xin Zhou and Xiaofei Tang, "Research and implementation of RSA algorithm for encryption and decryption," Proceedings of 2011 6th International Forum on Strategic Technology, 2011, pp. 1118-1121, doi: 10.1109/IFOST.2011.6021216.

[4] Misra, S. and Goswami, S. (2017). Fundamental Routing Protocols. In Network Routing (eds S. Misra and S. Goswami). https://doi.org/10.1002/9781119114864.ch3 .

[5] W. Pan, F. Luo and L. Xu, "Research and design of chatting room system based on Android Bluetooth" 2012 2nd International Conference on Consumer Electronics, Communications and Networks
(CECNet), 2012, pp. 3390-3393, doi: 10.1109/CECNet.2012.6201484.

[6] El-Seoud, Samir & Taj-Eddin, Islam. (2016). Developing an Android Mobile Bluetooth Chat Messenger as an Interactive and Collaborative Learning Aid. 10.1007/978-3-319-50340-0_1.

[7] Deb, Amrita & Sinha, Swarnabha. (2014). Bluetooth Messenger: an Android Messenger app based on Bluetooth, Connectivity. IOSR Journal of Computer Engineering. 16. 61-66. 10.9790/0661-16336166.

[8] http://ethesis.nitrkl.ac.in/6762/1/Bluetooth_banerjee_2015.pdf .

[9] https://in.mashable.com/tech/9554/firechat-will-be-your-go-to-messaging-app-when-the-internet-is-blocked .

[10] https://indianexpress.com/article/technology/tech-news-technology/ what-is-bridgefy-the-offline-messaging-app-with-over-1-million-installs-in-48-hours-7172736.