# Software Design Document (SDD)

*Rev: A0-05*

*28th June, 2017*

# "Secure Calling"

**Revision History:**

| Date | Rev No. | Description | By |
|---|---|---|---|
| 03-04-2017 | A0-01 | First draft | VVDN |
| 04-04-2017 | A0-02 | Second draft | VVDN |
| 28-04-2017 | A0-03 | Third draft | VVDN |
| 19-06-2017 | A0-04 | Fourth draft | VVDN |
| 28-06-2017 | A0-05 | Fifth draft | VVDN |

# Table of Contents

***Figures***

# 1. Introduction

This document describes SW design details of the “Secure Calling Application”.

This SDD is made for the reference of

- Product managers at VVDN & Embedded Downloads to confirm the SW design & Architecture before implementation
- Engineering Team at VVDN for implementation and validation of the Secure Calling Application.
- System Integration and Verification team at VVDN/ Embedded Downloads.

## 1.1 Product Overview

This project is to design and develop a secure calling application between users via SIP server. Following are the key elements of the complete system proposed:

- Register with SIP
- Communicating with blockchain
- Secure calling via SIP server
- Bitcoin transfer to Embedded Downloads

## 1.2 Scope of the Document

The product is a Secure Calling application for BitVault as a native application. This document helps the developers/engineers and QA team to understand the flow and validating the features with use cases. It also describes the detail description about the technical corners, encryption and protocols used during the secure calling.

## 1.3 Project Architecture

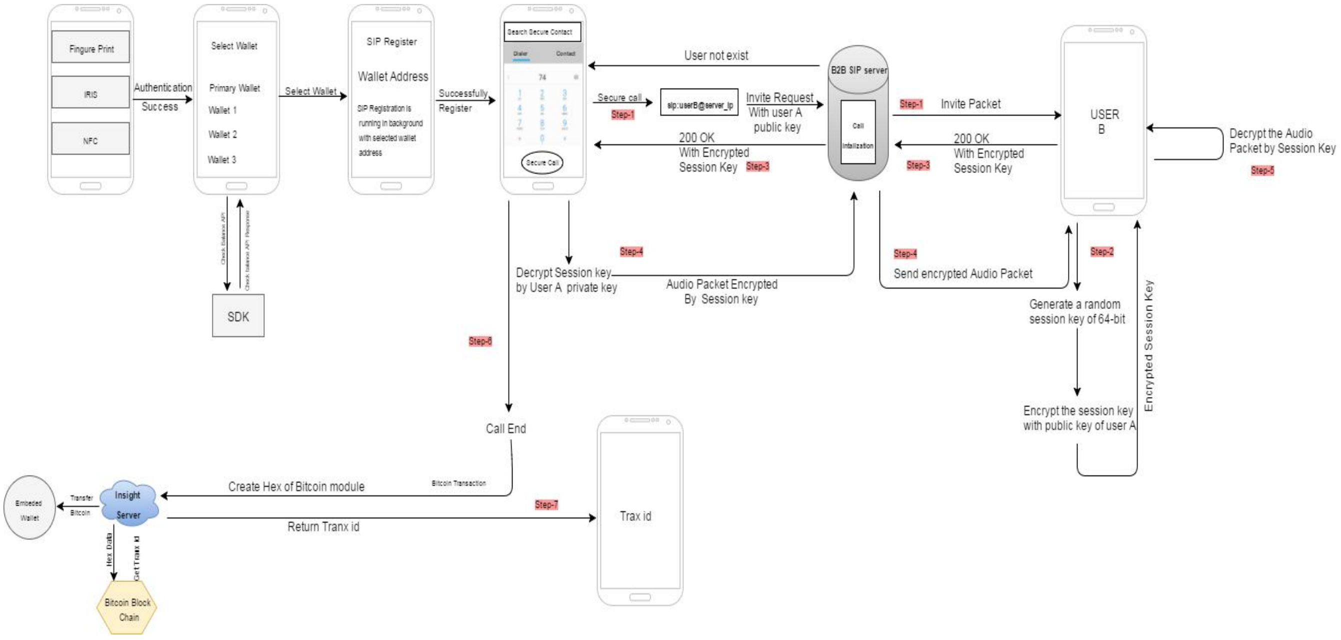


Figure 1 Project Architecture

## 1.4 Software Layer Architecture

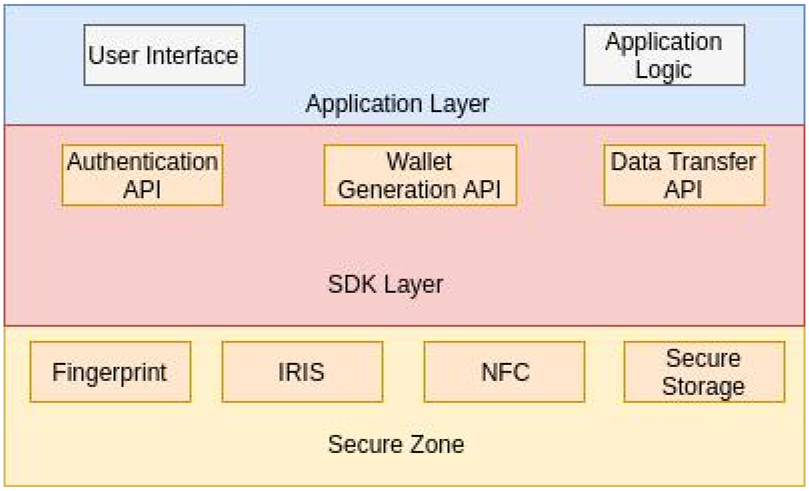


Figure 2 Software Architecture

There are three layers.

- Kernel Layer
- SDK Layer
- Application Layer

## 2.1 Kernel layer

This layer will authenticate the iris, fingerprint and password. Kernel will be responsible for keys generation.

## 2.2 SDK layer

This layer will work as an interface between Application layer and Kernel. SDK will be having various sets of modules. Each module has different functionality, API and methods. Using this layer, application can do the following:

1. Authenticate user using the provided fingerprint, iris and password
2. SDK will expose an API that in turn calls kernel to generate keys and addresses.

## 2.3 Application layer

Application layer comprises of user interface and secure calling application’s logic.

# 3 Application Flow

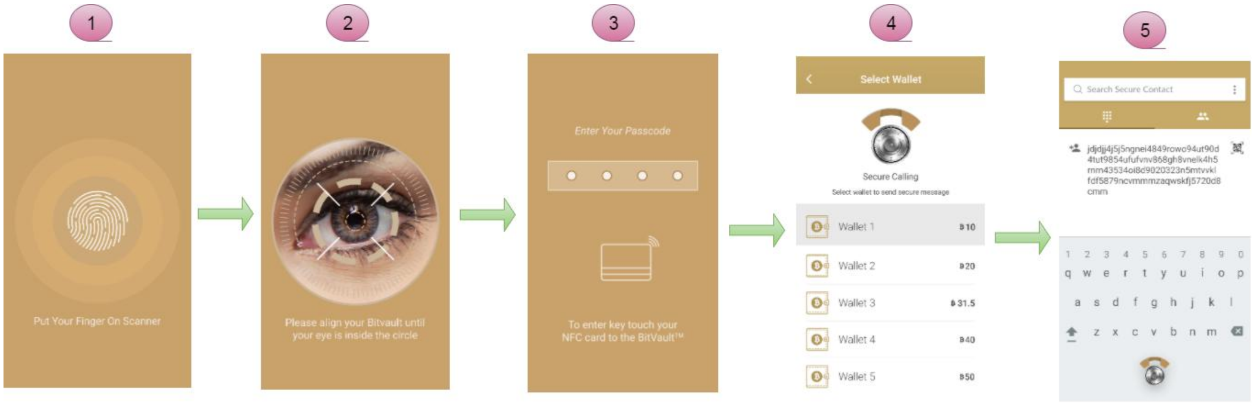
This section provides the complete understanding of application flow and design of the application.

The application flow has the following parts:

- Application Initialization
- Registration Process
- Calling Process

## 3.1 Application Initialization

Once the user starts the application, the authentication process would be done which includes Iris, Fingerprint and NFC card. If the user is authenticated then the application loads the select wallet screen will be shown with all the available wallets. User can choose the wallet to initiate the registration process with selected wallet address. After the registration process is successful, the application loads home screen and in this screen there are two options are dialer, contacts.



Put Your Finger On Scanner

Please align your Bitvault until your eye is inside the circle

Enter Your Passcode

To enter key touch your NFC card to the BitVault™

Secure Calling

Select wallet to send secure message

Wallet 1 ฿10
Wallet 2 ฿20
Wallet 3 ฿31.5
Wallet 4 ฿40
Wallet 5 ฿50

jdjdjj4j5j5ngnei4849rowo94ut90d4tut9854ufufvnv868gh8vnelk4h5mm43534oi8d9020323n5mtvvklfdf5879ncvmmmzaqwskfj5720d8cmm

Figure 3 Application Initialize

Authenticating the application will be the first step towards unlocking and accessing the application. Like any other app lock application where user sets either a pattern or set of numbers for unlocking the application, here user will have to provide all the three choices that are: iris, fingerprint and NFC card authentication.

### 3.1.1 Fingerprint Authentication

This screen is used to authenticate user by scanning fingerprint. Once the UI screen of iris is shown, Authenticate API will open the next UI of fingerprint where user will put the finger, fingerprint authentication will takes place by using SDK function *AuthUser()* and Kernel layer will authenticate the fingerprint by matching the input signatures with the recorded one. If authenticated, Kernel will return the **Success** else it will return **Failure**.

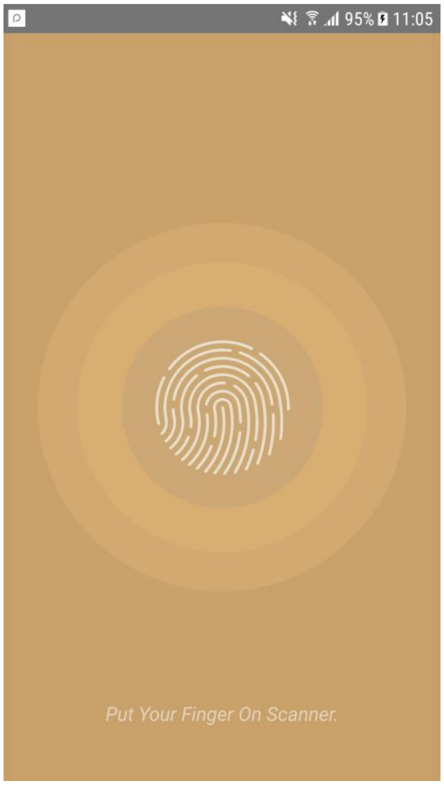
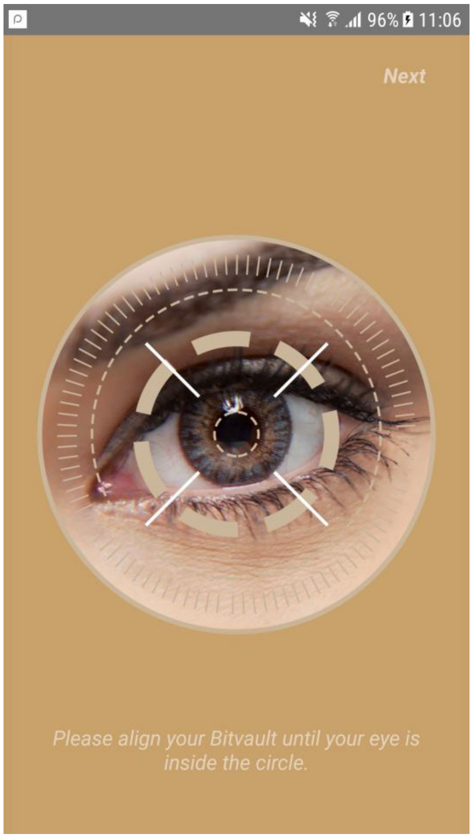


Figure 4 Fingerprint authentication screen

### 3.1.2 Iris Authentication

This screen is used to authenticate user by scanning iris.



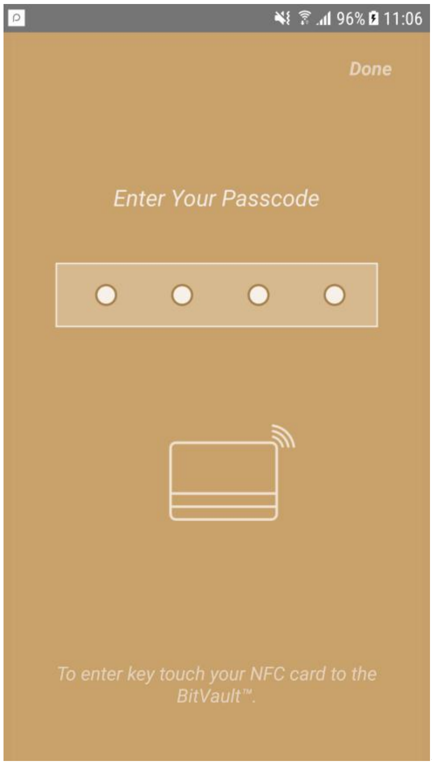
Please align your Bitvault until your eye is inside the circle.

Figure 5 IRIS authentication screen

Authenticate API of SDK layer will open the UI screen of Iris Authentication. When user puts eye, IRIS authentication will be the input parameter that will be sent to the Kernel layer in the function *AuthIris(),* which in return will provide **Success/Failure**.

### 3.1.3 Near Field Communication (NFC) Card

The next UI screen to be opened by the Authenticate API of SDK layer will be the NFC card screen. The user authenticates using NFC card if successful then user navigated to Select wallet screen.



Enter Your Passcode

○ ○ ○ ○

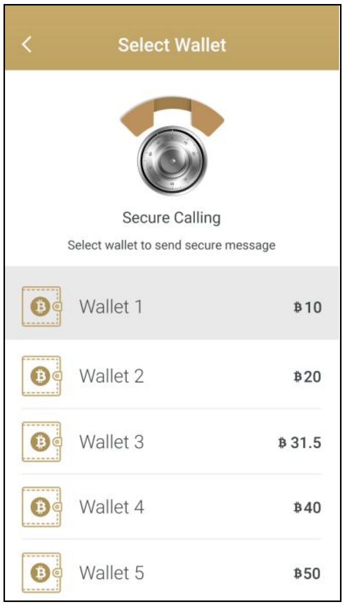
To enter key touch your NFC card to the BitVault™.

.

Figure 6 NFC card authentication screen

### 3.1.4 Wallet Selection

All the wallets available to the user are displayed in this screen. Wallet's generation and updation of the wallets will be done by Wallet generation API of SDK layer. Wallets are updated using the API *UpdateWalletBalance(Wallet Object*). The user then selects any one wallet that user wants to use for making a secure call using *GetTransactionHistory(Wallet Address).*



| | |
|---|---|
| Wallet 1 | ฿10 |
| Wallet 2 | ฿20 |
| Wallet 3 | ฿ 31.5 |
| Wallet 4 | ฿40 |
| Wallet 5 | ฿50 |

Figure 7 Wallet selection screen

| | |
|---|---|
| **Description** | Updates wallet balance by contacting the bit-coin block-chain |
| **Method** | UpdateWalletBalance(Wallet Object) |
| **Input Parameters** | Wallet<br>Type: Object |
| **Output Parameters** | Bit-coins<br>Type: long |

### 3.1.5 Register Screen

Register process will be created when the app starts up . In this process the user will be sending the REGISTER requested using TLS with these input parameter mention below:-

1. User selected wallet address.

| | |
|---|---|
| **Description** | Register the user's with the selected wallet address stored in SIP server |
| **Method** | UserRegister() |
| **Input Parameters** | User mobile no. |
| **Output Parameters** | True/False<br>Type: Boolean |

### 3.1.6 Refresh register process

After the successful registration process, the keep alive process will be started after every 60s for sending the refresh packet to the server in encrypt form.

| | |
|---|---|
| **Description** | Update the user information over the sip server |
| **Method** | KeepAliveProcess() |
| **Input Parameters** | none |
| **Output Parameters** | True/False<br>Type: Boolean |

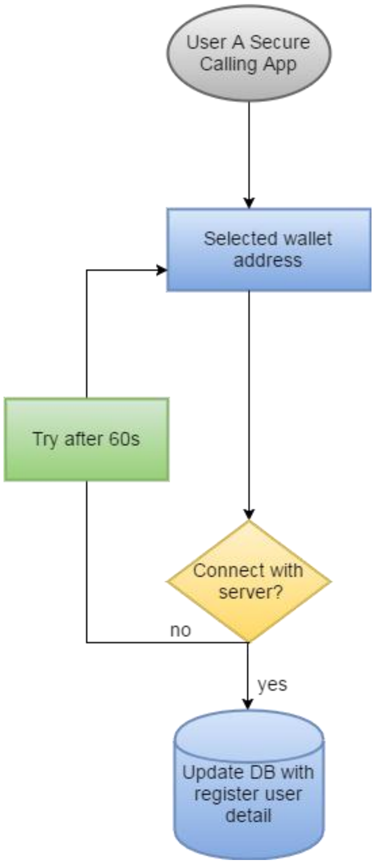
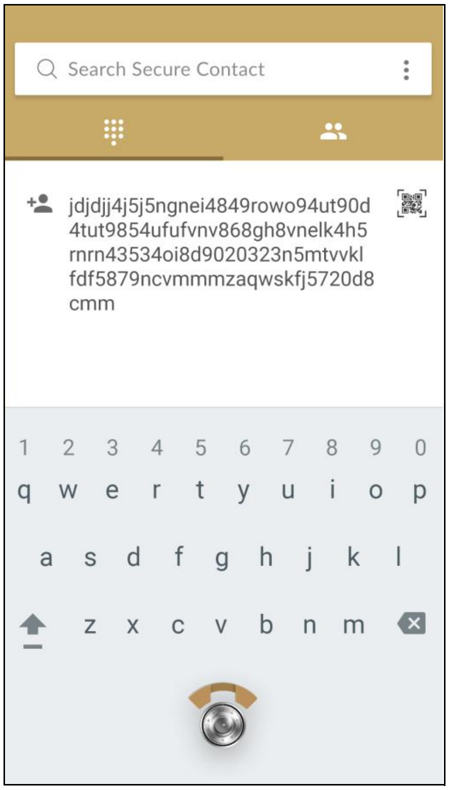


Figure 8 Keep Alive Process Sequence Diagram

## 3.2 Secure Call

### 3.2.1 Dialer Screen

- Calling:User can type the wallet address where the call needs to be made.This will redirect the user to the Calling Screen (Figure 11).
- Side menu(Setting): Setting option to redirect the user Setting screen.
- QR code scan: User can read the QR code image.
- Add contact : User redirect to add contact screen.



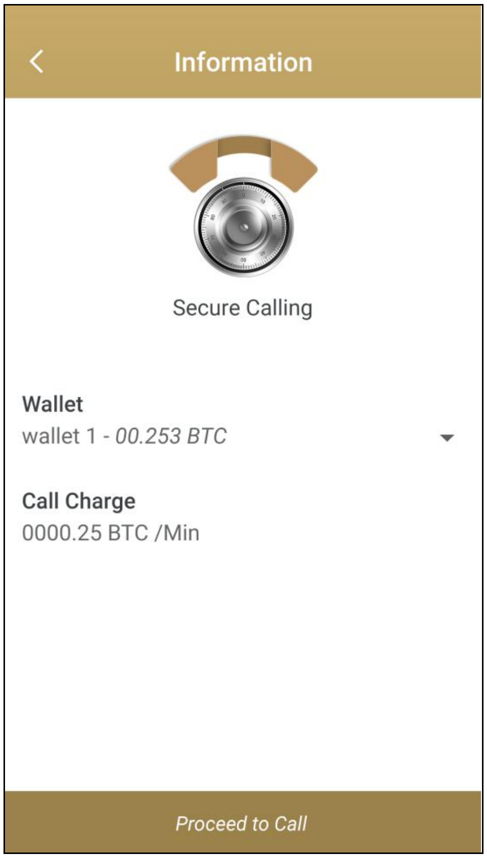
jdjdjj4j5j5ngnei4849rowo94ut90d4tut9854ufufvnv868gh8vnelk4h5rnrn43534oi8d9020323n5mtvvklfdf5879ncvmmmzaqwskfj5720d8cmm

Figure 9 Dialer Screen

### 3.2.2 Wallet Information Screen

In this screen user can change the wallet address from which he wants detected  the bit-coin and can see the call charge info also.

**Process to call button**: Its redirect the user to calling screen.



Wallet
wallet 1 - *00.253 BTC*

Call Charge
0000.25 BTC /Min

Figure 10 Dialer Screen

### 3.2.3 Secure call initialization process

In the call initialization process before call initial the wallet balance check will be there for just to check the wallet have the sufficient balance to make the call or not

After the call start there will be a separate thread running along with the call to calculate the call duration in order to deduct bit-coins in the end of the call,in case call duration exceeds wallet balance call is disconnected automatically.

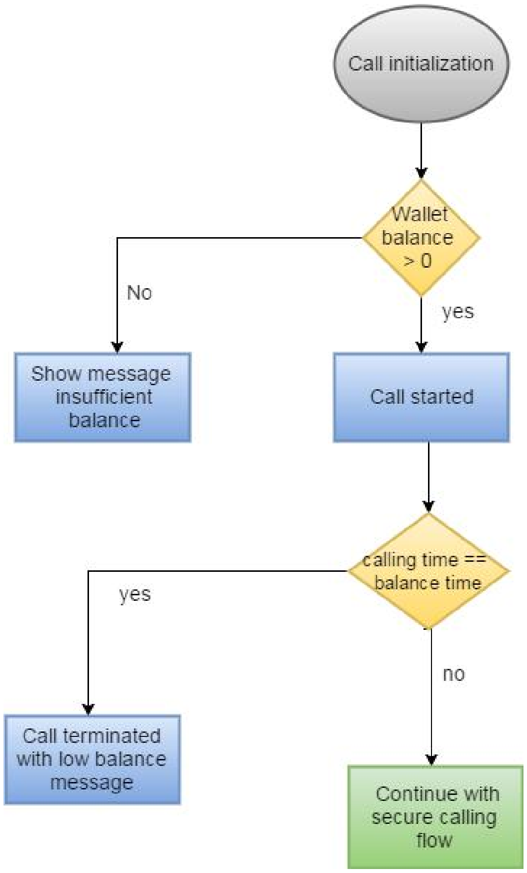

Call initialization
Wallet balance > 0
No
yes
Show message insufficient balance
Call started
calling time == balance time
yes
no
Call terminated with low balance message
Continue with secure calling flow

Figure 11 Call initialization process

### 3.2.4 Secure call process

The secure calling will be initiated between one end user to other via SIP server. The SIP server will be responsible for setting up the session between both parties. The server performs user authentication steps to make connection between both parties.while one user agrees to end the call, the request is made to SIP server for closing the calling session..

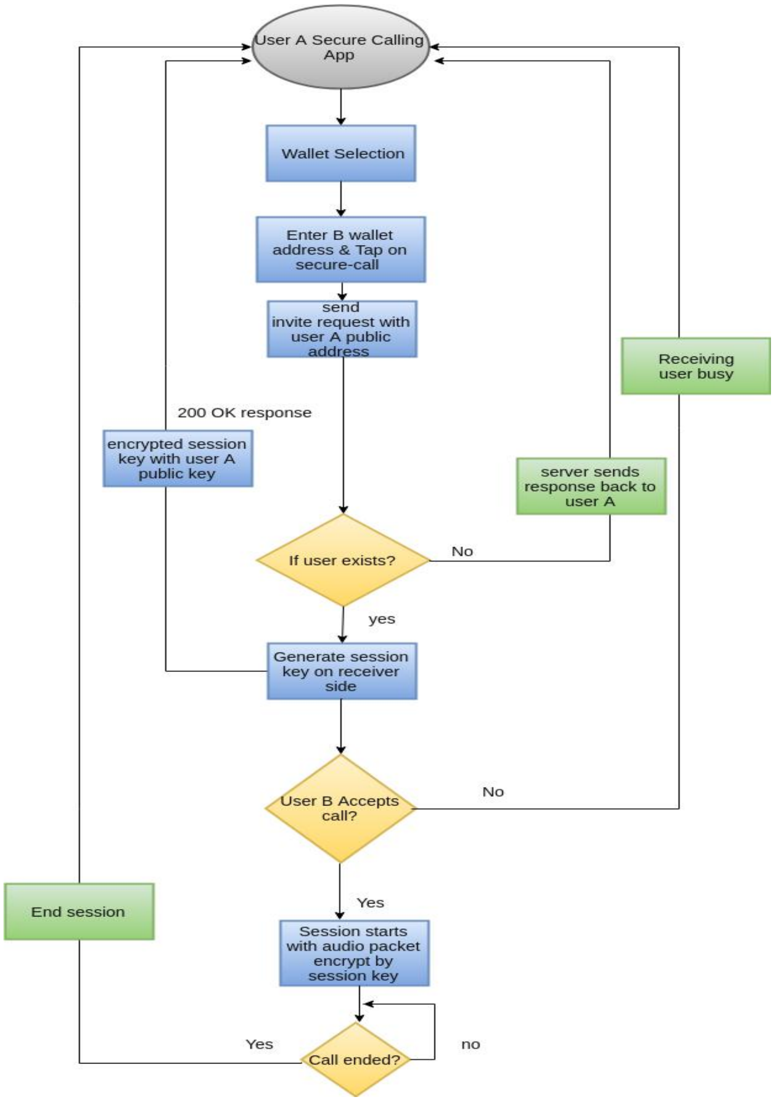
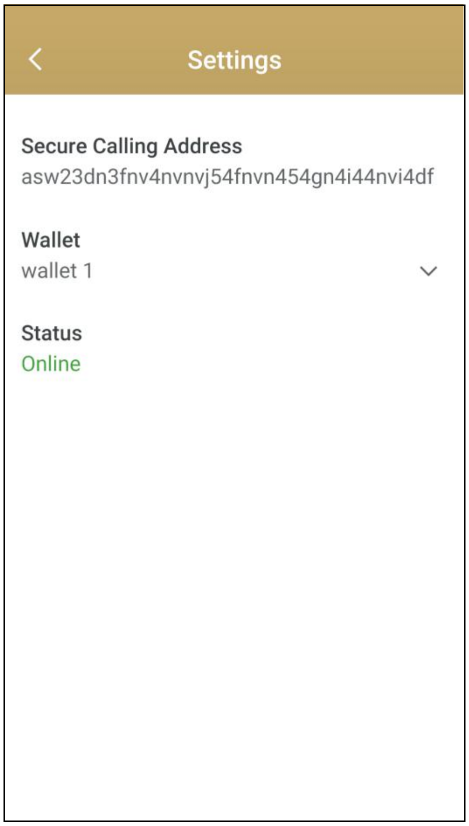

User A Secure Calling App
Wallet Selection
Enter B wallet address & Tap on secure-call
send invite request with user A public address
Receiving user busy
200 OK response
encrypted session key with user A public key
server sends response back to user A
If user exists?
No
yes
Generate session key on receiver side
User B Accepts call?
No
Yes
End session
Session starts with audio packet encrypt by session key
Yes
Call ended?
no

Figure 12 Sequence Diagram for Secure Calling

### 3.2.5 Settings Screen

In this screen user is able to change the wallet address and with the selected wallet address the user will again register on SIP server.

User can also able to check the device status.



Secure Calling Address
asw23dn3fnv4nvnvj54fnvn454gn4i44nvi4df

Wallet
wallet 1

Status
Online

Figure 13 Setting Screen

### 3.3 Calling Screen

When the user makes a call from the dialer screen, it will be navigated to the calling screen, where user will have the following provisions :-

- **Name & Photo display**: Dialed contact number or name and corresponding photo will be displayed on the screen. Photo option can be changed as per discussion.
- **Speaker (On/Off)**: User can enable or disable the speaker by clicking the speaker icon as per his convenience during the call.
- **Microphone(On/Off)** : User can enable or disable the microphone by clicking the Microphone icon as per his convenience.
- **Disconnect Call**: User can disconnect the outgoing call by clicking the "Disconnect Call" button.
- **Pause Button** : User can make put the ongoing call to hold by pressing the pause button.

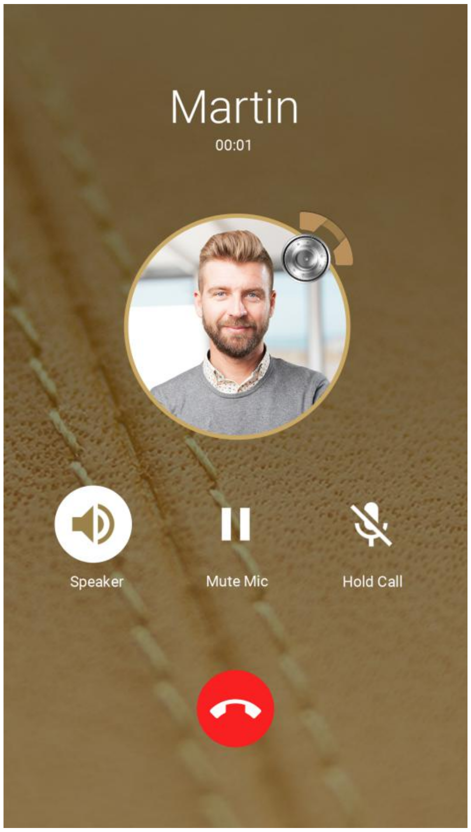
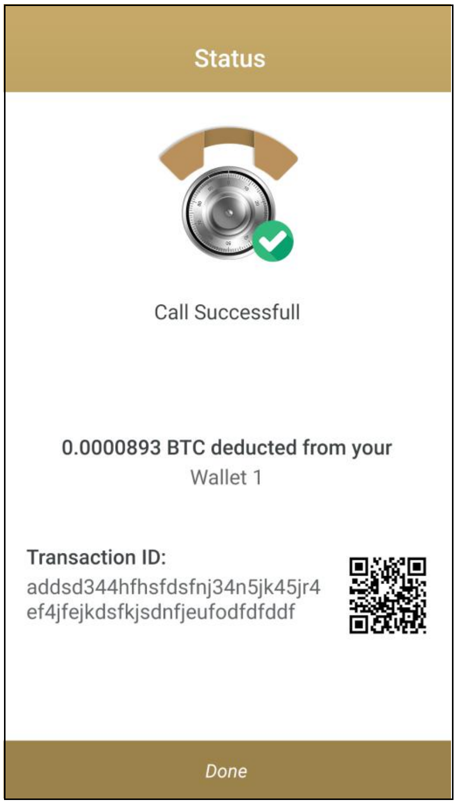


Figure 14 Calling Screen

### 3.4 Call charge screen

In this screen user able to see the bit-coin detected  amount based on the duration from selected wallet with transaction id.



Call Successfull

0.0000893 BTC deducted from your
Wallet 1

Transaction ID:
addsd344hfhsfdsfnj34n5jk45jr4
ef4jfejkdsfkjsdnfjeufodfdfddf

Figure 15 Status Screen

## 3.5 Incoming call screen

Incoming call screen will be displayed when a user will get a call from another user. User can see the following details on this screen :-

- Secure Contact Number of the caller.
- Status – Incoming call.

User will have the provision to ***Answer or Decline*** the incoming call by sliding the middle call icon to the up or down respectively.

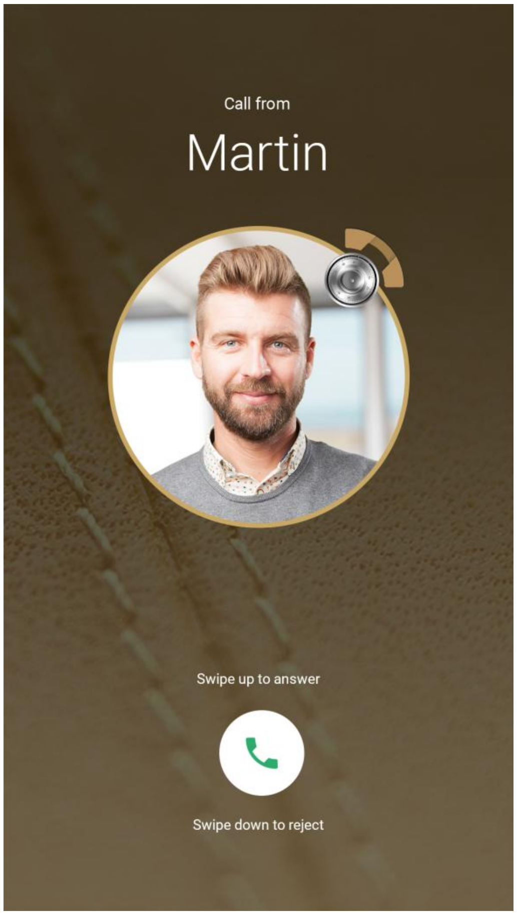


Figure 16 Incoming Call

### 3.6 Contact screen

This screen would be the Contact register for the application.Contacts will be sorted alphabetically. Contact person's name and photo(if needed) will be shown in the list.

User would be having the following provision :-

- **Add Button :-** User can add new contact to his contact register by clicking the "+" icon. This would redirect the user to the Contact details screen(Figure 8).
- **Search Bar :-** User can make a search for a desired contact person by typing relevant alphabets in the search bar. The list will get sorted accordingly.
- **Contact List item click :-** By clicking any of the contact list item, user can see the details of the existing contact. This would redirect the user to the Contact details screen(Figure 22). On this screen, user can edit the information of an existing contact.

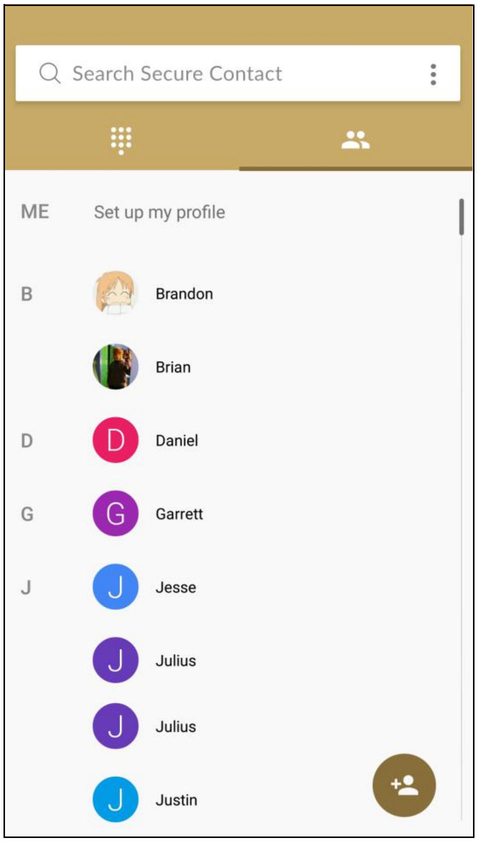


Figure 17 Contacts Screen

### 3.6.1 Add contact screen

This screen would appear when user clicks on “+” button of the Contact register screen. User have to fill the following information about the new contact:-

- **First Name**
- **Last Name**
- **Contact Number**

By clicking on the “Save” button, the new contact will be saved in the Contact register.

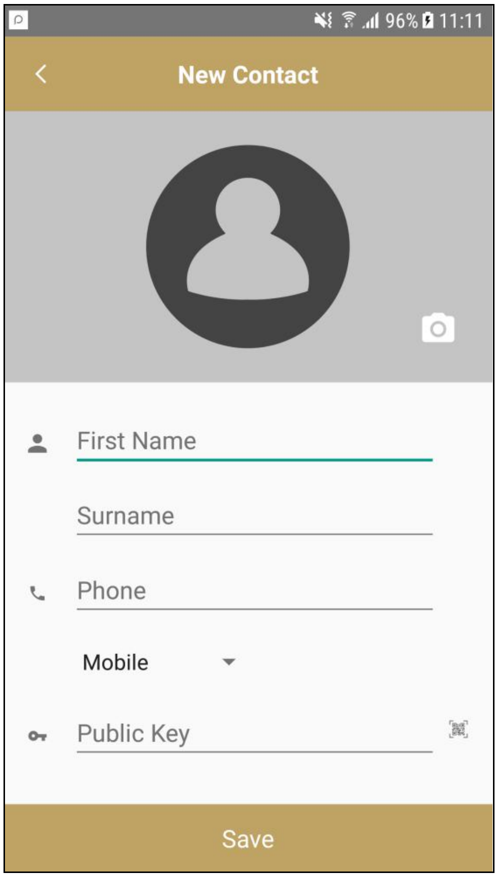
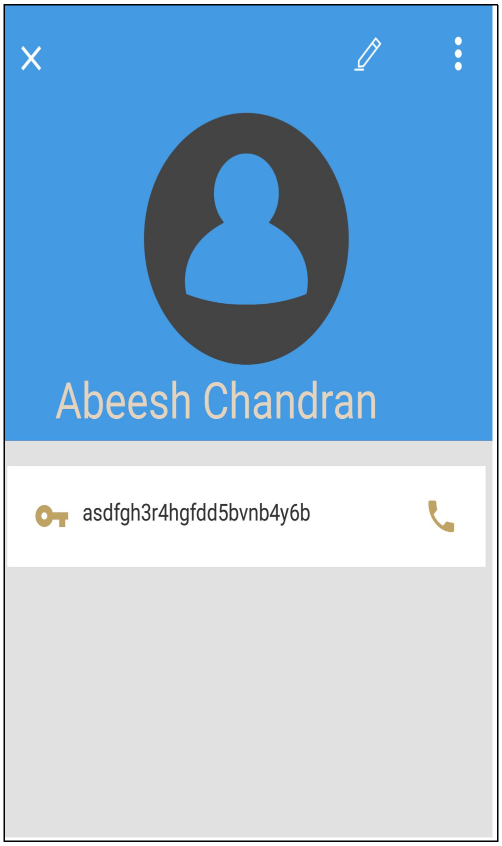


**Figure 18 Add Contacts Screen**

### 3.6.2 Contact details screen

This screen will be displayed on clicking any of the Contact list row. Corresponding contact details will be shown in the screen.

User have provision to edit these details by clicking the “Edit” button present on the top right corner of this screen. All the fields will be editable and the edit button will be transformed in the “Save” button. After editing the contact details, user can click on the “Save” button, to save the edited information of an existing contact.



Abeesh Chandran

asdfgh3r4hgfdd5bvnb4y6b

Figure 19 Contact Details Screen