**INDUSTRIAL TRAINING REPORT**

**VIDEO RTC (Real Time Communication)**

**undertaken at**

**Addval Solutions Pvt. Limited**



Submitted by

**Varender Singh**

**UE-128093**

Under the guidance of

**Prateek Attrii**

**( General Manager )**

**Department of Information Technology**

**UNIVERSITY INSTITUTE OF ENGINEERING AND TECHNOLOGY**

**PANJAB UNIVERSITY , CHANDIGARH**

**JUNE-2016**

**DECLARATION**

I hereby declare that the project work entitled “ **Video RTC**” is an authentic work of my work carried out at “**Addval Solutions**” as requirement of six month industrial training for the award of degree of B.E. At University Institute of Engineering and Technology , Panjab University , Chandigarh under the guidance of **Prateek Attrii** during Six month industrial training.

**Varender Singh**

**UE-128093**

Date - 20/May/2016

Certified that the above statement made by the student is correct to the best of our knowledge and brief.

**Prateek Attrii**

**(General Manager)**

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Title** | **Page** |
| 1. | Introduction to IOS | **1-3** |
| 2. | IPhone Applications | **4-7** |
| 3. | Software and hardware requirements | **8-13** |
| 4. | Programming Language ( **Objective C** ) | **14-15** |
| 5. | Web RTC | **16-33** |
| 6. | Future Scope | **35-37** |
| 7. | References | **38-39** |

**LIST OF FIGURES**

|  |  |
| --- | --- |
| **Figure Title** | **Page No** |
| iOS homescreen on an iOS | **2** |
| Basic structure of an application in iOS | **6** |
| Various states of an application | **7** |
| Icon of XCode | **9** |
| StoryBoard in an Application | **9** |
| Work Space in XCode | **10** |
| Editor Area in Xcode | **11** |
| Utility area supports in XCode | **12** |
| Debug Area of XCode | **12** |
| Simulator of iOS device | **13** |
| Apple MacBook | **14** |
| Logo of WebRTC | **18** |
| How peers can get connected by WebRTC | **19,28** |
| Tabs Profile , Contact & Call Logs | **20** |
| Profile View Controller | **21** |
| The view Controller by which we can change the user name of App | **22** |
| Contacts that are visible | **23** |
| Multipeer view controller in which either you can accept request or decline the request | **24** |
| Contact view Controller if the devices get connected to each other successfully | **25** |
| Screen on other devices when you will call | **26** |
| View Controller if you start calling to next device | **27** |
| Call Log View Controller | **28** |
| How you can delete the row | **30** |
| File within the View Directory | **31** |
| Files Within the Model Directory | **32** |
| Files inside the Controller directory | **33** |

**CHAPTER – 1**

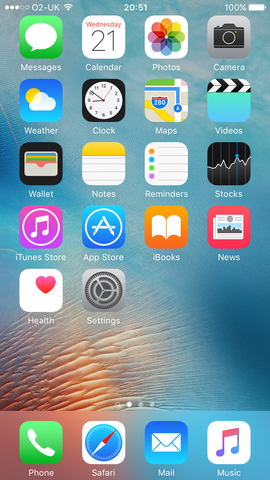
**Introduction To iOS**

**iOS** is a mobile operating system created and developed by Apple Inc. and distributed exclusively by Apple hardware. It is the operating system that powers many of the company’s mobile devices , including iPhone , iPad & iPod touch. It is the second most popular mobile operating system in the world by sales after Android. It was originally released in 2007 . iOS is derived from OS X, with which it shares the Darwin foundation. IOS is Apple’s mobile version of the OS X operating system used in Apple computers.

iOS shares the OS X some frameworks such as Core Foundation and Foundation kit; however its UI toolkit is CocoaTouch rather than OS X’s Cocoa,so that it provides the UIKit framework rather than the AppKit framework, It is therefore not compatible with OS X for applications. Also while iOS also shares the Drawin foundation with OS X, Unix like shell access is not available for users and restricted for apps,making iOS not fully Unix-compatible either.

There are four abstraction layers

* The core OS layer.
* The Core Service layer.
* The Media layer.
* The cocoa Touch layer.



*Figure: iOS home screen on an iOS 6S*

**SPECIFICATIONS OF iOS**

|  |  |
| --- | --- |
| **Developer** | Apple Inc. |
| **Written in** | C , C++ , Objective-C , Swift |
| **OS family** | Unix-like, based on Darwin (BSD) |
| **Initial release** | June 29,2007 |
| **Platforms** | 64 and 32-bit ARM architecture |
| **Official website** | [**http://www.apple.com/ios/**](http://www.apple.com/ios/) |

Some more features of iOS are as follow:

1. Maps
2. Siri
3. Multi-Touch
4. Accelerometer
5. GPS
6. High end processor
7. Camera
8. Safari
9. Powerful APIs
10. Game Center
11. In-App Purchase
12. Reminders
13. Wide range of gestures

**CHAPTER – 2**

**IPhone Applications**

**What is an Application?**

An applications program is a computer program designed to perform a group of coordinated functions,tasks or activities for the benefit of the user.

In Information technology ,an application is a computer program designed to help people perform an activity.An application thus differ from an operating system(which runs a computer), a utility (which performs maintenance or general-purpose chores),and a programming tool(with which programs are created).Depending on the activity for which it was designed,an application can manipulate text,numbers,graphics,or a combination of these elements.

**About iOS App Architecture?**

App need to work with the iOS ensure that they deliver a great user experience .Beyond just a good design for your apps design and user interface,a great user experience encompasses many other factors.Users expect iOS apps to be fast and responsive while executing the app to use a little power as possible.Apps need to support all of the latest iOS devices while still appearing as if the app was tailored for the current devices.

**App Life Cycle**

Apps are a sophisticated interplay between your custom code and the system frameworks.The system frameworks provide the basic infrastructure that all apps need to run ,and you provide the code required to customize that infrastructure and give the app the look and feel you want.

iOS framework rely on design patterns such as model view controller and delegations in their implementation.Understanding those design patterns is crucial to the successful creation of an app.

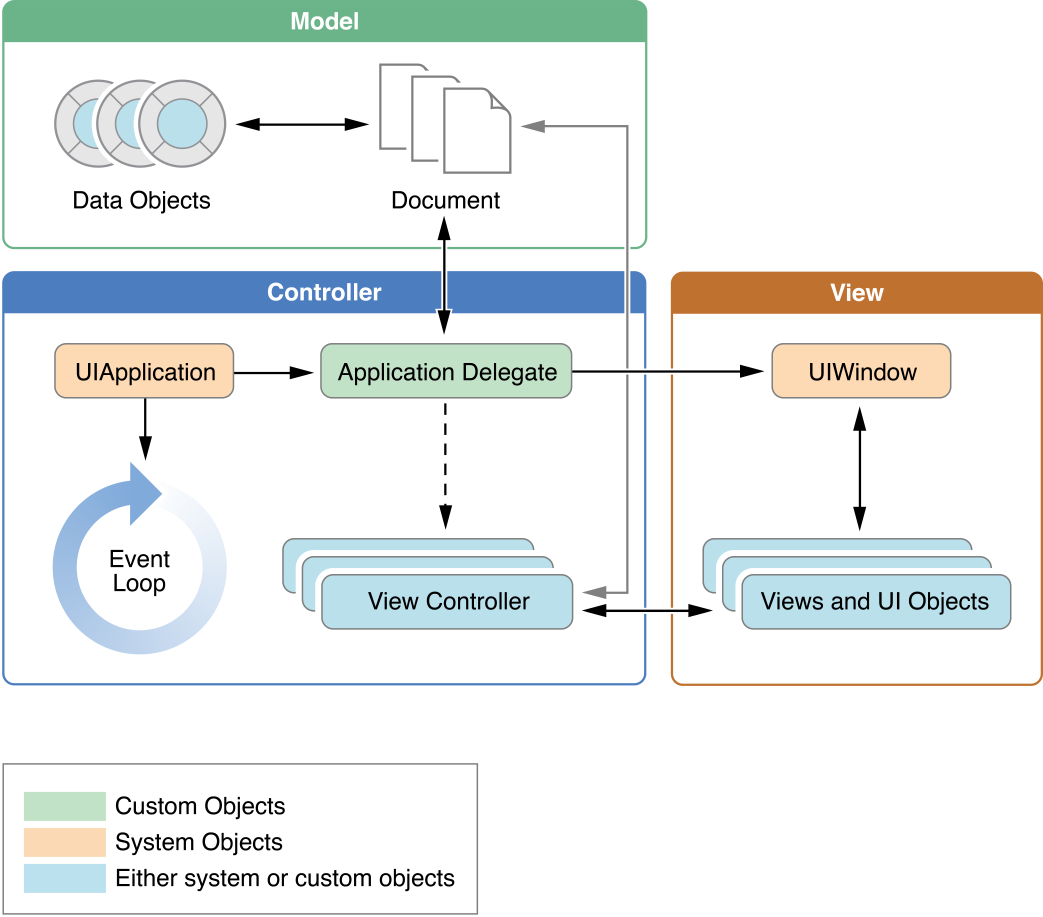
**The Main Function**

The entry point for every C-based app is the main function and iOS apps are no different.What is different is that for apps you do not write the main function yourself. X-code creates this functions as part of your basic project.

**Structure of the App**

During startup ,the UIApplicationMain function sets up several keys objects and starts the app running.At the heart of every iOS app is the UIApplication object,whose job us to facilitate the interaction between the system and other objects in the app.

This first thing to notice is that iOS apps use model-View-Controller architecture.The pattern separates apps data and business logic from the visual presentation of that data.This architecture is crucial to creating apps that can run on different devices with different screen sizes.

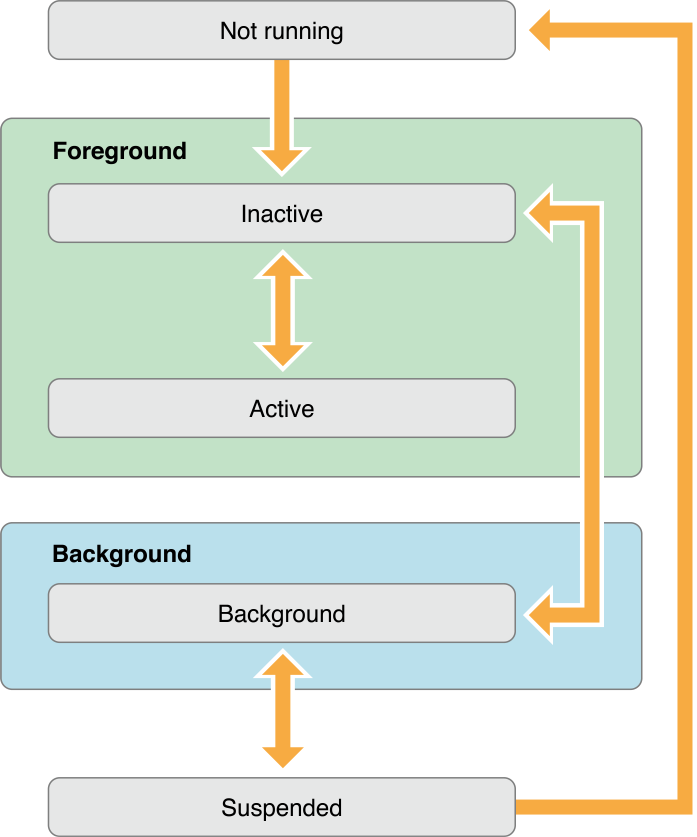


*Figure:Represents basic structure of an application in iOS*

**Execution states for Apps**

At any given moment,your app is in one of the states .The system moves your app from state to state in response to actions happening throughout the system.For example ,when the user presses Home button,a phone call comes in,or any of several other interruptions occurs,the currently running apps change state in response.

* **application:willFinishLaunchingWithOptions**:-This method is your app’s first chance to execute code at launch time.
* **application:didFinishLaunchingWithOptions**:-This method allows you to perform any final initialization before your app is displayed to the user.
* **applicationDidBecomeActive**:-Lets your app know that it is about to become the foreground app.Use this method for any last minute preparation.
* **applicationWillResignActive**:-Lets you know that your app is transitioning away from being the foreground app.Use this method to put your app into quiescent state,
* **applicationDidEnterBackgound**:-Lets you know that you app is running in the background and may be suspended at any time.
* **applicationWillEnterForegrund**:-Lets you know that your app is moving out of the background and back into the foreground.but that it in not yet active,
* **applicationWillTerminate**:-Lets you know that your app is being terminated.This method is not called if you app is suspended.



*Figure:Representing various states of an application*

**CHAPTER -3**

**Software and Hardware requirements to build an iOS App**

**X-Code**

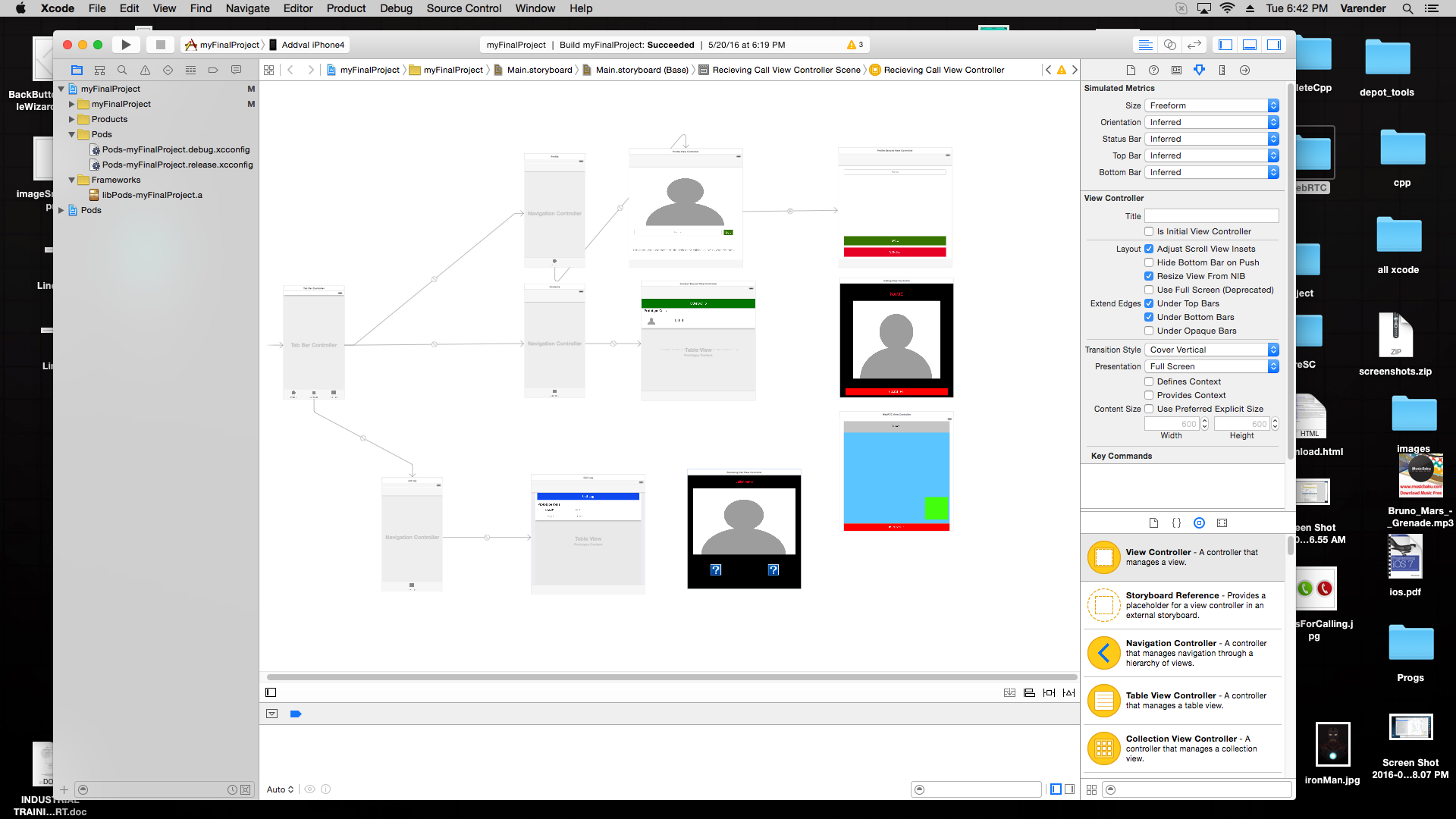
Xcode is an integrated development environment(IDE) containing a suite of software development tools developed by Apple for developing software for OS X, iOS ,WatchOS and tvOS. First released in 2003 ,the latest stable release is version 7.3,1 and is available via the Mac App Store free of charge for OS X El Captian users.



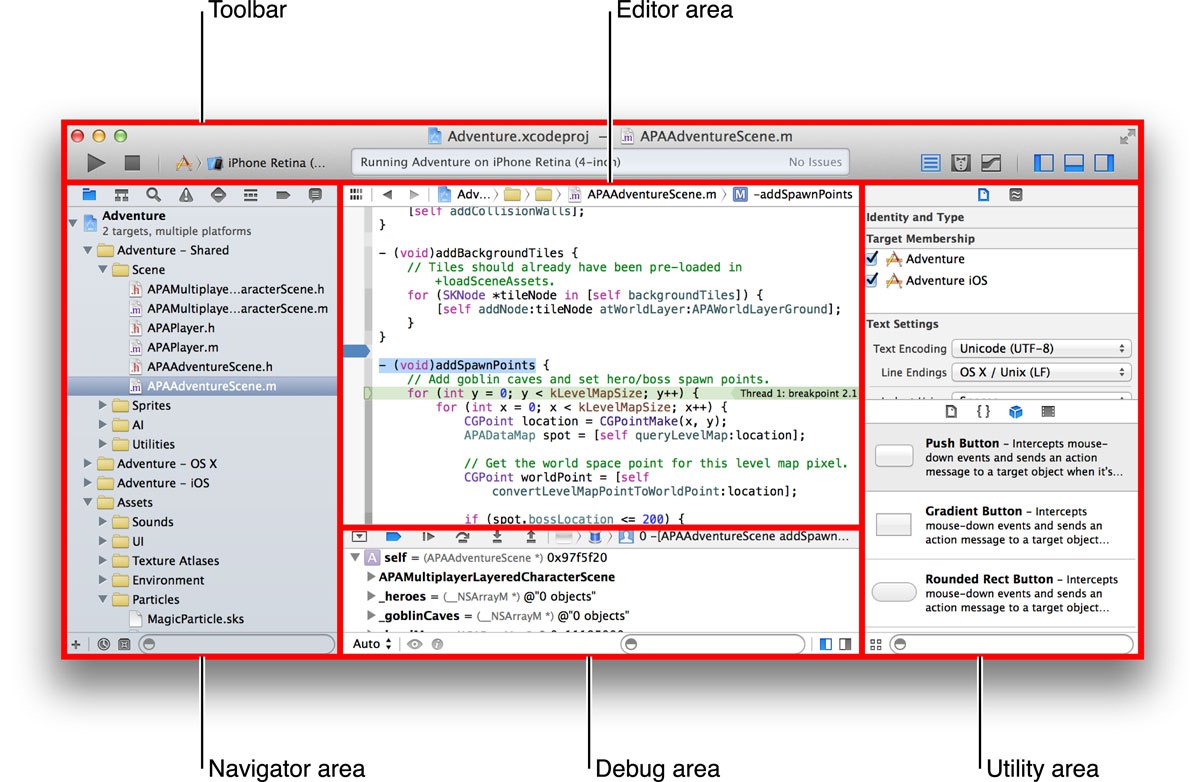
*Figure:- represents icon of Xcode*

**Main Features**

Xcode supports source code for the [programming languages](https://en.wikipedia.org/wiki/Programming_language) [C](https://en.wikipedia.org/wiki/C_(programming_language)), [C++](https://en.wikipedia.org/wiki/C++), [Objective-C](https://en.wikipedia.org/wiki/Objective-C), [Objective-C++](https://en.wikipedia.org/wiki/Objective-C++), [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [AppleScript](https://en.wikipedia.org/wiki/AppleScript), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)), and [Swift](https://en.wikipedia.org/wiki/Swift_(programming_language)), with a variety of programming models, including but not limited to [Cocoa](https://en.wikipedia.org/wiki/Cocoa_(API)), [Carbon](https://en.wikipedia.org/wiki/Carbon_(API)), and Java.



*Figure: represents storyboard in an application*



*Figure:Represents various work spaces in Xcode*

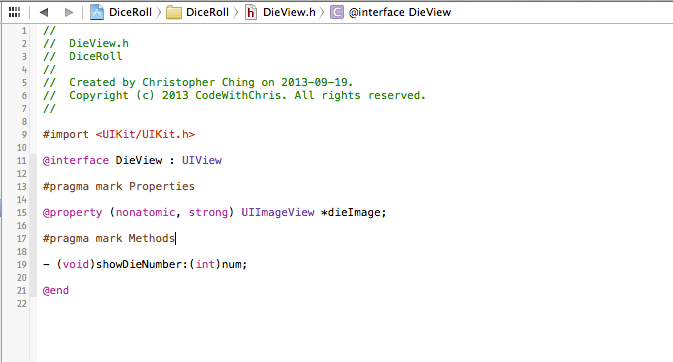
**XCode WorkSpaces and their explanation**

1. **Navigation Area:-** in this pane, there are a bunch of different navigators that you can switch between using the *Navigator selector bar (refer to diagram)*. The three navigators that you will use a lot are the Project, Search and Issue navigators.Within the project navigator, you can also create “Groups” to organize your files in.

You can also right click on the project navigator and create new files or add existing files to your project. Or you can drag folders or files from your computer directly onto the project navigator and will popup a dialog asking how you want to add the files.

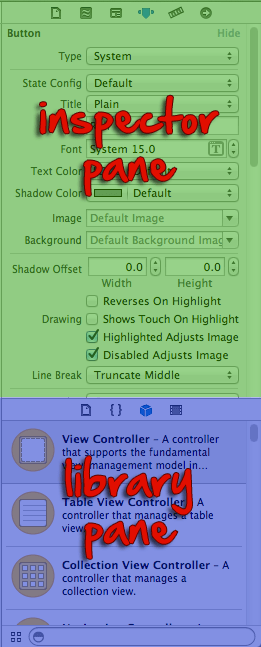
If you click on a file in the project navigator, it will display the file’s contents in the Editor area. If you double click a file instead, it will open a new window which can be useful when you have dual monitors.The root node of the project navigator is your XCode project file (indicated by the blue icon). If you click that, the project properties will open in the editor area.

1. **Editor Area:-** This is the area where coding happens.



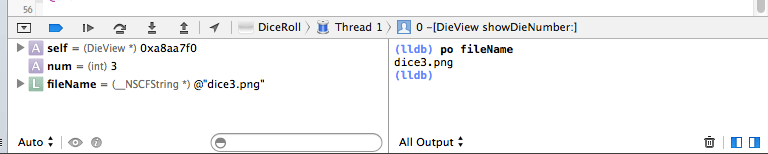
*figure : editor area in an xcode*

**3.Utility Area:-**The XCode utility area is comprised of two panes: the inspector pane and the library pane.



*Figure : Represents the Utility area sub-parts in XCode*

**The Debug Area :-** The debug area will show you console output and the state of various variables when you run your application.



*Figure:represents debug area of XCode*

**iOS Simulator:-** XCode 5 comes bundled with a wonderful iOS Simulator for you to test your application on. In fact, you can use the iOS simulator for most of your development and then find a device to test on when you’re nearly done.

You can actually do a lot with the simulator including:  
-Device rotation  
-Simulating various GPS coordinates  
-Device shake  
-Simulating



*Figure:represents simulator of iOS device*

**Hardware requirements**

**MacBook :-**The MacBook is a [brand](https://en.wikipedia.org/wiki/Brand) of [notebook computers](https://en.wikipedia.org/wiki/Laptop) manufactured by [Apple Inc.](https://en.wikipedia.org/wiki/Apple_Inc.) from early 2006 to late 2011, and relaunched in 2015. It replaced the [iBook](https://en.wikipedia.org/wiki/IBook) series and 12-inch [PowerBook](https://en.wikipedia.org/wiki/PowerBook) series of notebooks as a part of the Apple-Intel transition from PowerPC. Positioned as the low end of the [MacBook family](https://en.wikipedia.org/wiki/MacBook_family), below the premium ultra-portable [MacBook Air](https://en.wikipedia.org/wiki/MacBook_Air) and the powerful [MacBook Pro](https://en.wikipedia.org/wiki/MacBook_Pro),the MacBook was aimed at the consumer and education markets. It was the best-selling [Macintosh](https://en.wikipedia.org/wiki/Macintosh) ever. For five months in 2008, it was the best-selling laptop of any brand in US retail stores.Collectively, the MacBook brand is the "world's top-selling line of premium laptops.



*Figure:represents Apple MacBook*

**CHAPTER – 4**

**Programming Language ( Objective - C )**

**Objective-C**

Objective-C is a general-purpose, object-oriented programming language that adds Smalltalk-style messaging to the C programming language. This is the main programming language used by Apple for the OS X and iOS operating systems and their respective APIs, Cocoa and Cocoa Touch.

The programming language Objective-C was originally developed in the early 1980s. It was selected as the main language used by [NeXT](https://en.wikipedia.org/wiki/NeXT) for its [NeXT STEP](https://en.wikipedia.org/wiki/NeXTSTEP) operating system, from which OS X and iOS are derived.[[2]](https://en.wikipedia.org/wiki/Objective-C#cite_note-2) Portable Objective-C programs that do not use the [Cocoa](https://en.wikipedia.org/wiki/Cocoa_(API)) or [Cocoa Touch](https://en.wikipedia.org/wiki/Cocoa_Touch) libraries, or those using parts that may be ported or reimplemented for other systems, can also be compiled for any system supported by [GNU Compiler Collection](https://en.wikipedia.org/wiki/GNU_Compiler_Collection) (GCC) or [Clang](https://en.wikipedia.org/wiki/Clang).

Objective-C source code 'implementation' program files usually have .m filename extensions, while Objective-C 'header/interface' files have .h extensions, the same as C header files. Objective-C++ files are denoted with a .mm file extension.

### Apple development and Objective-C

After acquiring NeXT in 1996, [Apple Computer](https://en.wikipedia.org/wiki/Apple_Computer) used OpenStep in its new operating system, Mac [OS X](https://en.wikipedia.org/wiki/OS_X). This included Objective-C, NeXT's Objective-C based developer tool, [Project Builder](https://en.wikipedia.org/wiki/Project_Builder), and its interface design tool, [Interface Builder](https://en.wikipedia.org/wiki/Interface_Builder) (both now merged into one [Xcode](https://en.wikipedia.org/wiki/Xcode) application). Most of Apple's present-day [Cocoa API](https://en.wikipedia.org/wiki/Cocoa_(API)) is based on OpenStep interface objects, and is the most significant Objective-C environment being used for active development.

**Features of Objective-C**

* Classes are objects
* Dynamic typing and optional static typing
* Categories
* Message sending
* Expressive message syntax
* Introspection
* Dynamic run-time
* Automatic garbage collection
* C inside
* C++ fluent
* Simplicity
* Access to Apple technologies

**CHAPTER – 5**

**WebRTC**

**(Real Time Communication)**

**WEB RTC ( Real Time communication )**

**WebRTC** (**Web Real-Time Communication**) is an [API](https://en.wikipedia.org/wiki/Application_programming_interface) definition drafted by the [World Wide Web Consortium](https://en.wikipedia.org/wiki/World_Wide_Web_Consortium) (W3C) that supports [browser](https://en.wikipedia.org/wiki/Web_browser)-to-browser applications for [voice calling](https://en.wikipedia.org/wiki/Voice_calling), [video chat](https://en.wikipedia.org/wiki/Video_chat), and [P2P file sharing](https://en.wikipedia.org/wiki/P2P_file_sharing) without the need of either internal or external [plugins](https://en.wikipedia.org/wiki/Plug-in_(computing)).



*figure: represents webRTC logo*

**Major Components of WebRTC**

1. getUserMedia :- which allows a web browser to access the camera and microphone and to capture media.

2.RTCPeerConnection:- which sets up audio/video calls

3.RTCDataChannel:- which allow browsers to share data via peer-to-peer.

4.didChangeState:- method being used to get the states that can be either connecting , connected , disconnected

**Benefits of webRTC**

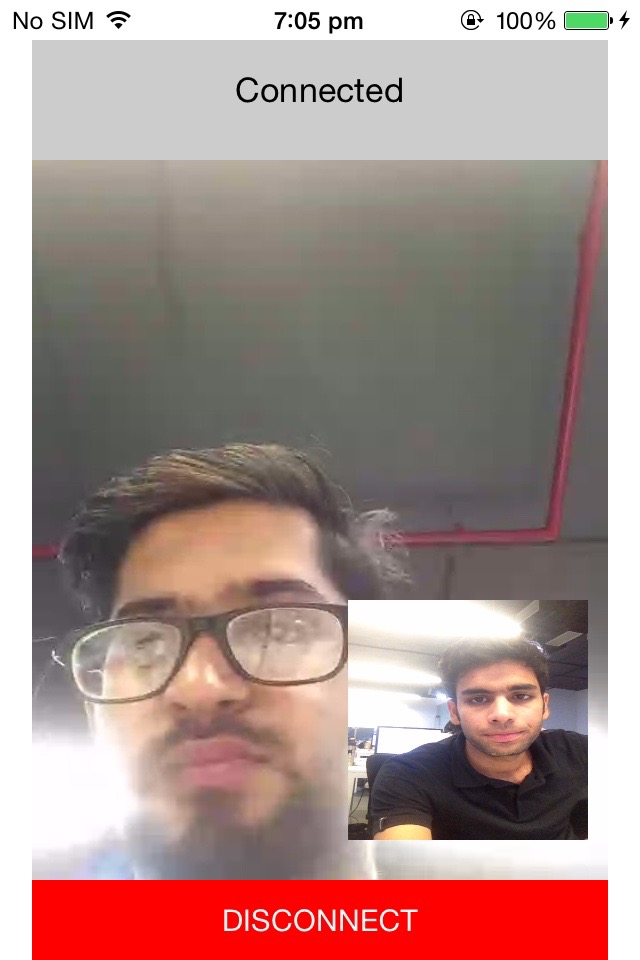
1.**Ease to use:** Real Time communication is supported without the need for additional applications or plug-ins

2.**Security:** webRTC enforces the usage of encryption for media .Thereby WebRTC provides a high security level than most currently available commercial telephony systems.

3.**Cost saving:** save on the costs of toll free telephone numbers for call centres

4.**Rich communication:** Enhance the communication to users and between employers with video and messaging without the need for special applications and servers.

5.**Un-interrupted communication:** keep the customers on the web page and at the same time start a voice and video call with customers.



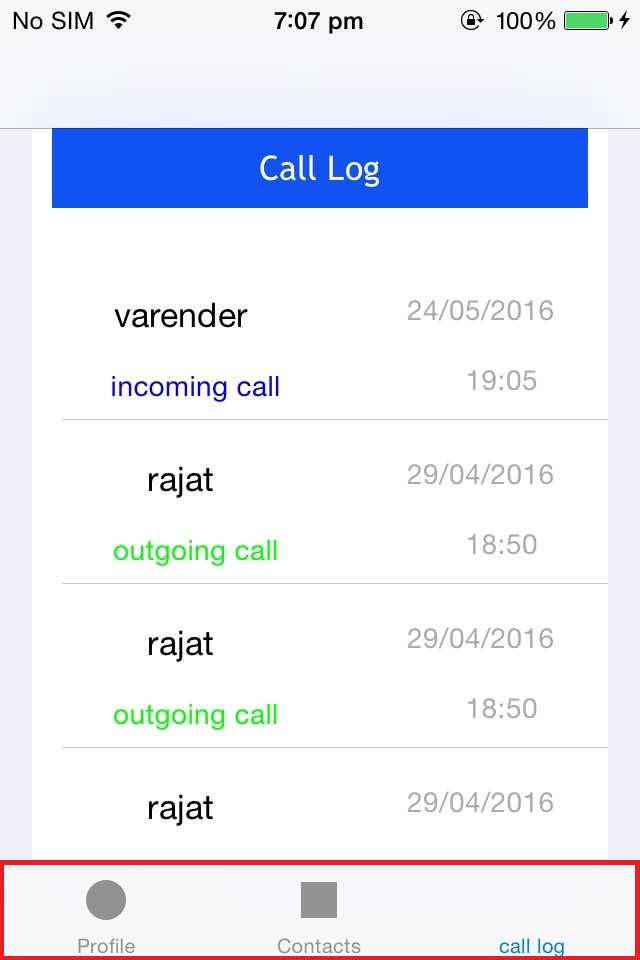
*Figure: represent how peers can get connected by webRTC*

**Steps that i have followed to prepare this project are as follow**

1. Information gathering about webRTC and its architecture.
2. Finding for an previously built library
3. Implementing it in iOS in its programming language that is objective-C. Its an advance topic so generally you have to search about it a lot i you have to implement it.
4. Fixing bugs
5. Adding constraints so that my project can run on devices having different screen sizes.
6. Generally in webRTC you have to enter the same room from both the peers so that connection can be built via the server but i have used MPC framework i.e MultiPeer Connectivity framework for calling because its a tough task to setup an dedicated servers individually.
7. Multipeer Connectivity frame work has one limitation that it can send calls only to near by devices using wifi and bluetooth only.
8. I have integrated both of webRTC framework and MPC framework to build this project.
9. I also have manually tested mine application.

**Various sceenshots of my application are as follow**

1. In my application i have 3 tabs Profile,Contacts,Call Logs as represented in the figure



*figure : tabs Profile , Contacts , Call Log*

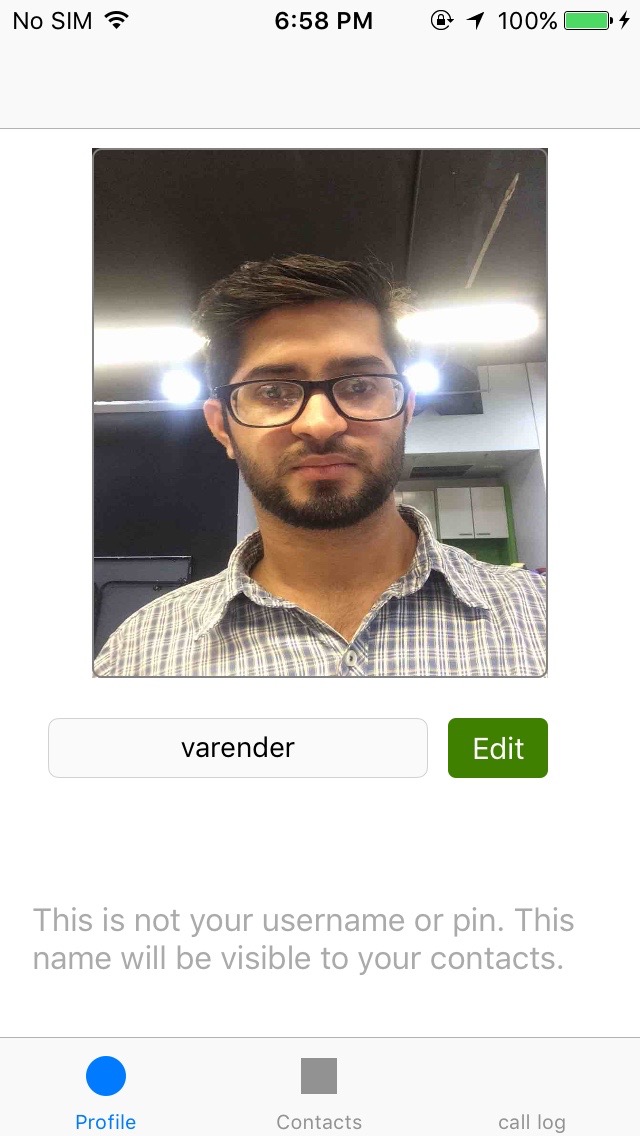
**Profile-** In this profile we can add image of the app user and his name .This name he will enter will be seen by the person.

**Contacts -**  In this contacts i have show i have shown all the devices that are connected to your devices through Multipeer connectivity.So that we can make calls with each other only by clicking on the contact in our screen.

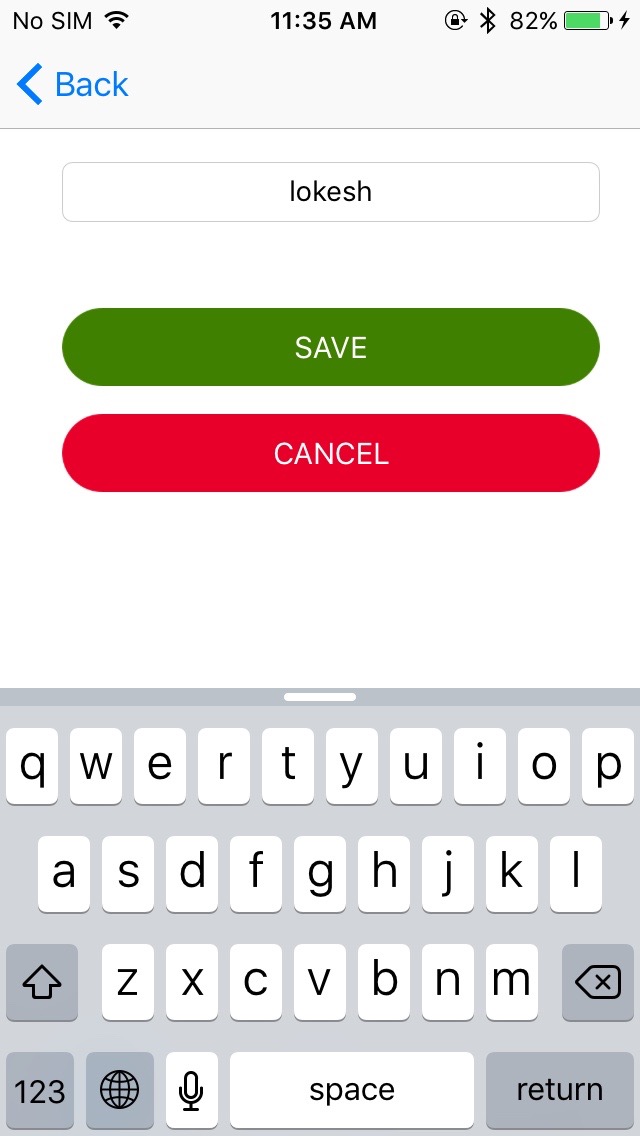
**Call Logs -** Call logs shows you all the calls you have made and are being received by you.

***Tab No 1 - Profile***

1. In Profile you can set your image as well as change your image. You can also edit you name by clicking onto that edit button that you can see on the right side of the name.



*figure : represents the profile view controller*



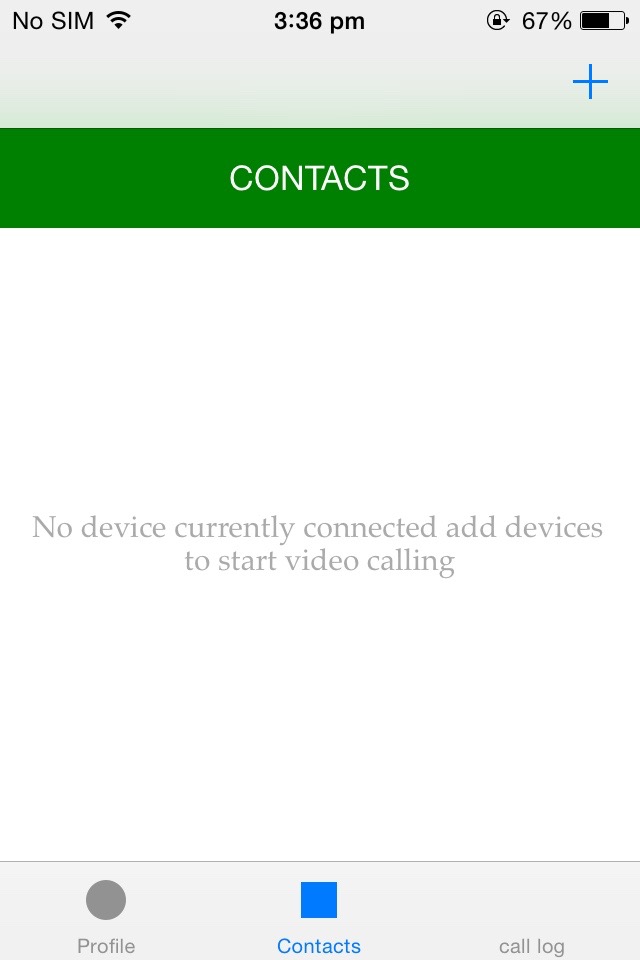
*figure: represents the view controller by which we can change the username of the app.*

**Edit name View Controller**

In this you can change your user name which will be show to other users. This view controller has 2 button that is either you can save the edited text or not. In such a case when you leave the text Field of name as empty the username will be sent as the device name.

***Tab No-2 Contacts***

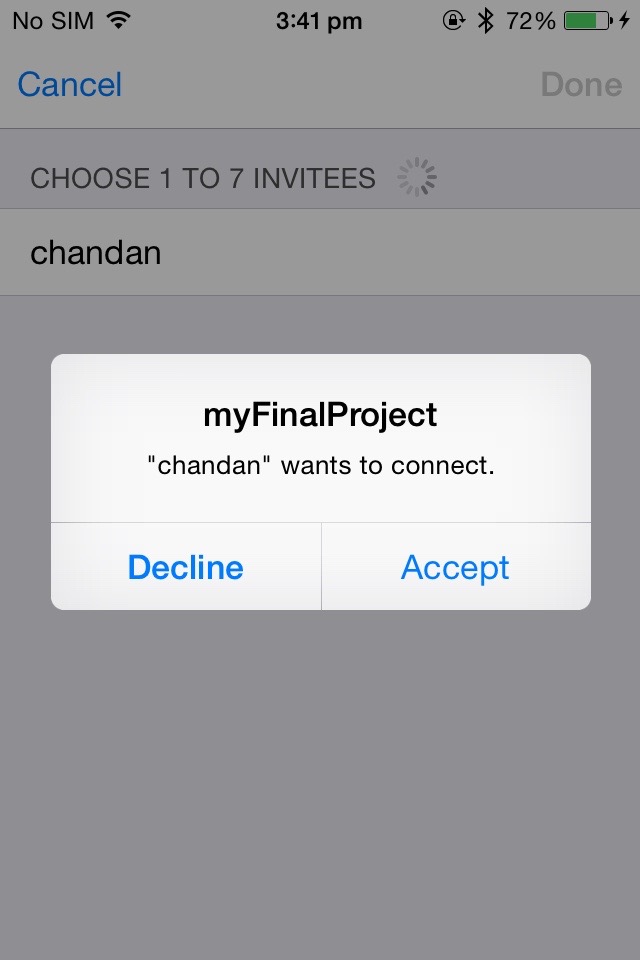
1. The second tab button represents contacts which are added to the list when they get connected by Bluetooth/ Wifi and the list get updated when the connection to that person get lost.



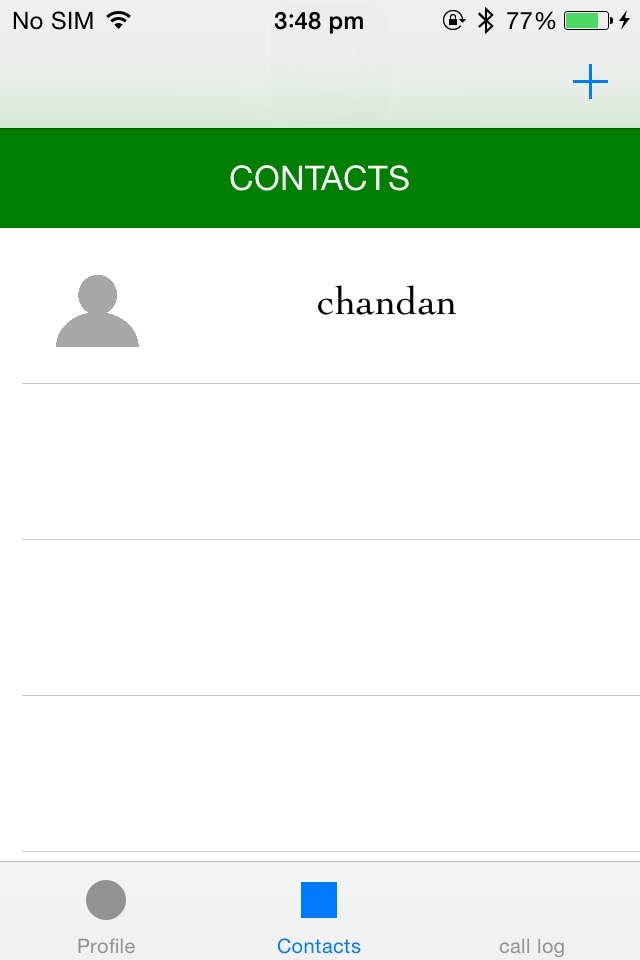
*figure : represents contacts that are visible*

On clicking the plus button you can get the last of other devices which you wanna get connected to. Then you can click on the cell which will then send request to other device.

The an alert message will be pop up if you want to connect then you accept the request else you can decline it.



*figure : represents multi peer view controller in which either you can accept request or decline the request.*



*figure: represents the contacts view controller if the device gets connected to each other successfully.*

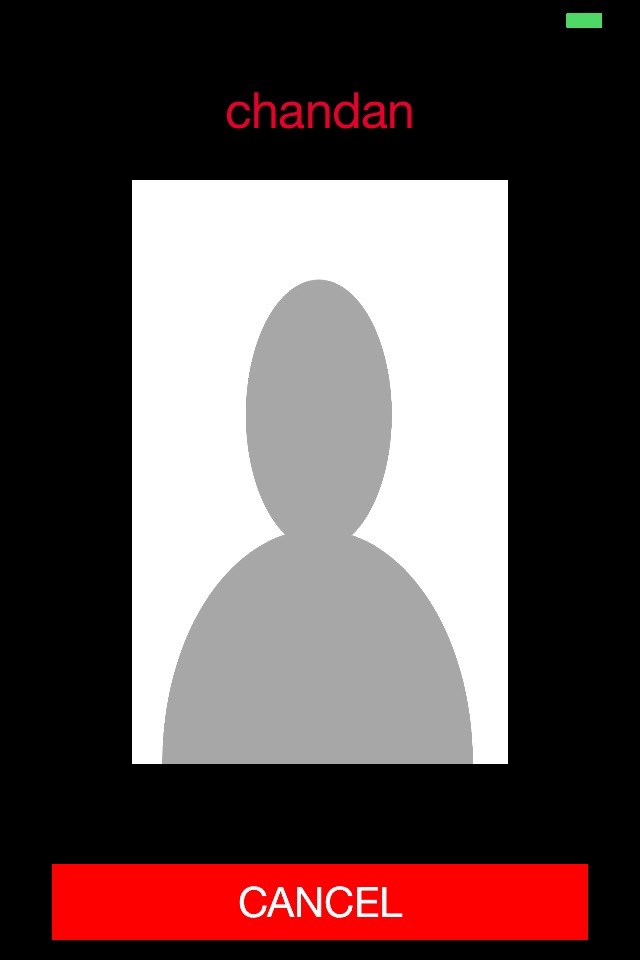
Now if you will tap on the cell the call will be received by the next device and the screen on the next device .



*figure : represent the screen on the other device when you will call .*

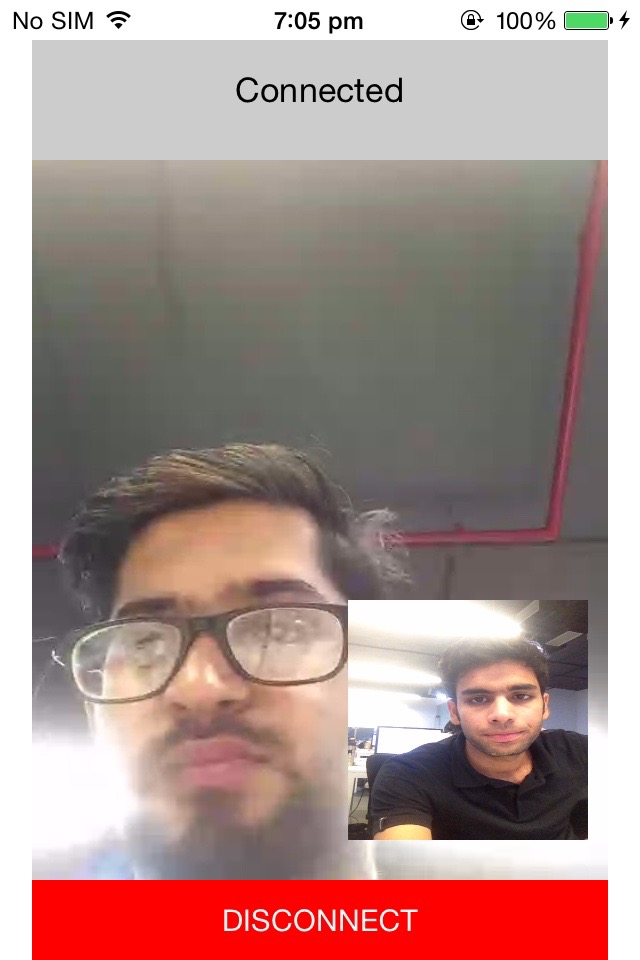
This view controller will pop up on the screen of next device . Where you can either accept the call or decline the call. If you will accept the call then video calling will start else it would not.

The view controller under this will pop on your sceen when you will tap on the Contact. If you will press the cancel button then the calling will stop you get back returned to contact view controller.



*figure: represents the view controller if you start calling to next device,*

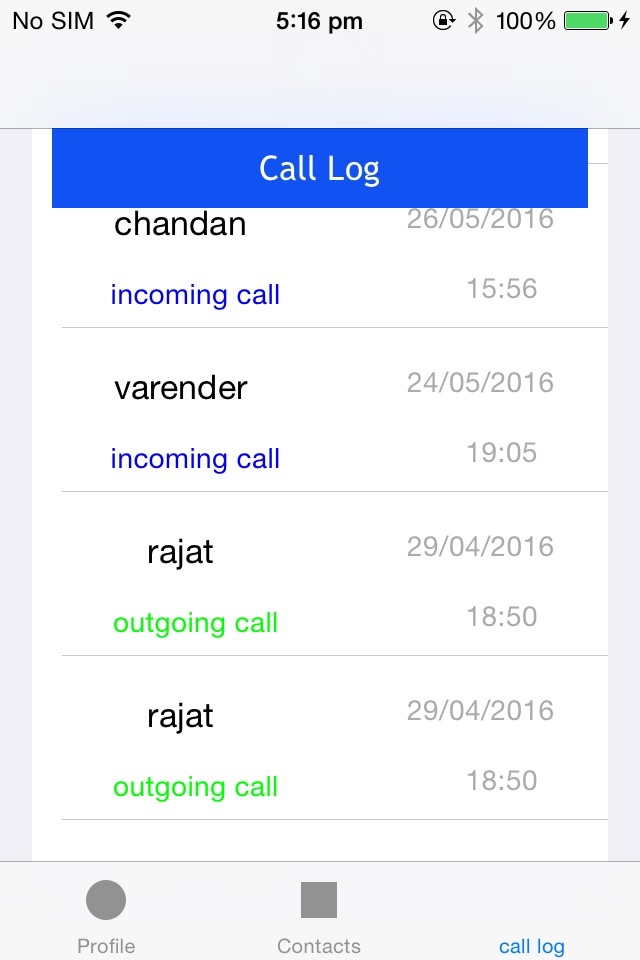
In case the built connection through multipeer got disconnected the call will get terminated and you will find yourself on the contacts view controller.



when you will accept the call then view calling will be start between the peers. If you will press the disconnect button the view controller will dismiss and you will get your self in contact view controller.

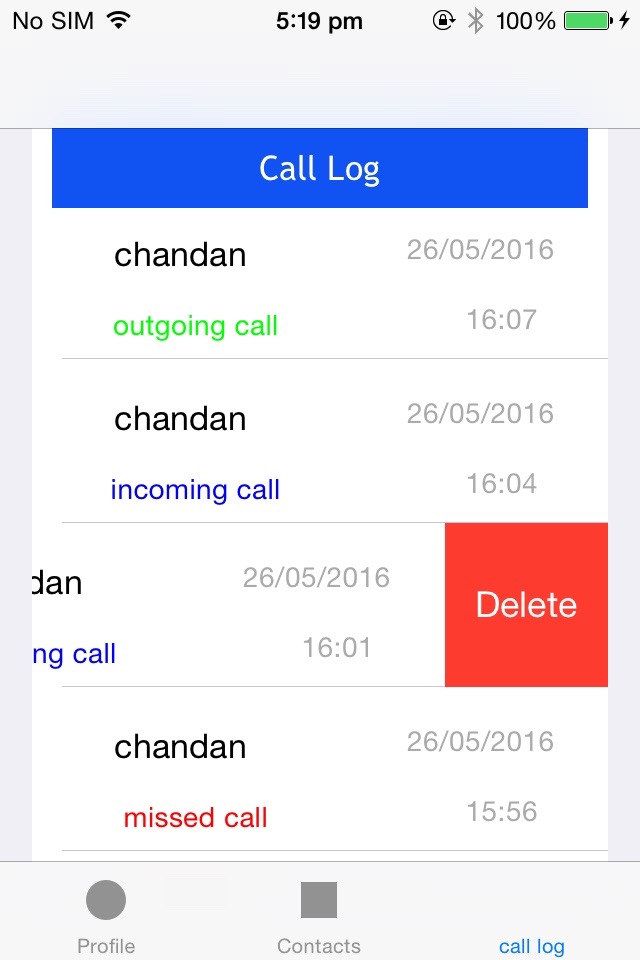
***Tab No -3 Call Log***

This view controller will give you the detail of all the calls that is either incoming or outgoing calls.



*Figure represents the call log view controller*

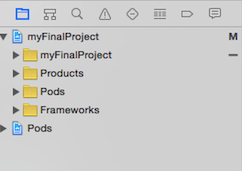
By swiping on the cell to the left side the delete button will appear on clicking that the row will be deleted.



*figure : represents how you can delete the row*

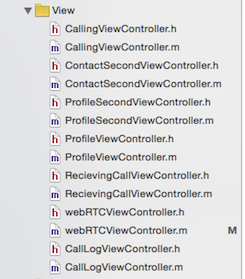
**PROJECT STRUCTURE IN XCODE**

**The Model View Controller in my project are as follow:-**



**VIEW**

These all files are related to the view and each file function is explained below



*figure: represent the files within the view directory in my project*

**ProfileViewController –** in this view controller we have information of username and image of the user. Only the username will be seen on the next device.

**ProfileSecondViewController –** in this view controller we can edit the username as well as save.

**ContactSecondViewController –** in this view controller you will see the devices that you are connected with via wifi and Bluetooth through MPC framework.

**CallLogViewController –**  in this view controller you can watch the list of missed calls ,incoming calls and outgoing calls.

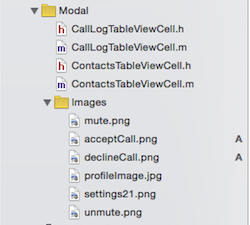
**CallingViewController –** this view will appear if you will start calling other device

**RecievingCallViewController –** this view controller will appear if an call is being received.

**webRTCViewController –** this view controller will open when call is being accepted by you or being accepted by other one.

**Model**

Model represents the cells that are beng used.In my project I have used sqlite or core data so I don’t need any data model class. I have user NSUserDefault to store the data within the device this NSUserDefault is perisistant data class.



*figure: files within the Model Directory*

**IMAGES**

In Images folder I have 6 images that I have used within this project. Most of them are icons.

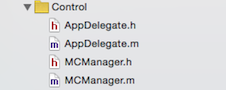
Other files functions are as follow:-

**CallLogTableCell –** it is the custom cell I have created in my project and used I CallLogViewController

**ContactsTableViewCell -**  in this custom cell we have an image and a label which is being replicated on the number of devices that are been connected top mine device.This cell is being used in ContactsSecondViewController.

**CONTROLLER**

The controller will take data from view to model and model to view. It acts as an intermediate between both of them.



*figure: files inside Control directory*

The files between this directory are explained as follow:-

**AppDelegate –** it is the default file that has app delegate method like app is in background, app enter foreground, app will enter foreground ,app terminated etc

**MCManager –** it is the file I have created that receives the messages of 2 peers connected via MPC framework.It send these messages to other controllers that is either call id disconnected or accepted ,calling,decline etc etc

**A LITTLE ABOUT MPC ( Multipeer connectivity ) FRAMEWORK**

The Multipeer Connectivity Framework is one of the many new frameworks iOS 7 introduces, and it broadens the range of the applications that can be built for the new operating system. Its aim is to enable developers to create applications that will allow devices being in close distance to get connected, simply using Wi-Fi networks or Bluetooth. Great tools are offered for establishing communication between nearby devices, and for exchanging data and other resources without much effort to be required.

Talking about data, there are three kinds of data that can be sent and received using the Multipeer Connectivity framework. These are:

1.Messages (including text, images, and everything else that can be converted to a*NSData* object

2. Streaming

3.resources

Going into some more details about messages, there are two modes that can be used for data transmission: The reliable and the unreliable mode. Using the reliable mode to send data, the framework makes sure that everything will arrive to the receiver, and in the order been sent. The cost of this reliable data sending though, is that is required more time for the transmission to be completed. On the other hand, when using the unreliable mode, data is send in no-time, really fast, but there is no guarantee that everything will arrive to the other peer, and of course, data is not sent in order. Which way is the most preferable clearly depends on the requirements of each application, so it’s up to every developer to decide what mode will use.

**CHAPTER – 6**

**Future Scope of WebRTC**

**FUTURE SCOPE OF IOS APPLICATION DEVELOPMENT**

The iPhone and iPad are made what they are because of the App Store. Apple admits this itself, even going so far as to include it in their advertising.

Moreover, the App Store is made by developers. Some great apps have been made by Apple (some had to be), but one of the distinguishing things about the App Store is the promise it offers to hardworking developers and entrepreneurs.

Here you have a way to get your app in front of tens of millions of users, many of whom are happy to spend their money on brilliant (and sometimes terrible) apps. Here is a way to make money by building on some already pretty incredible devices.

The comScore report, and the following wall of opinion, should only worry iOS users if the rise of Android necessitates a loss of good development for the iOS platform; as the market share of other Smartphones (and tablets) increases, will developers to begin to drift away from the iPhone and iPad in favour of Android (or other) devices.

Fortunately, I don’t believe that this will be the case. At least not anytime soon. The development prospects for iOS are too promising.

**ABOUT THE MOBILE INDUSTRY REVOLUTION**

The mobile application market is rapidly growing and gaining popularity as an emerging job opportunities market. India is the third fastest growing app market in the world. As per Gartner report, Indias active mobile connections will exceed 900 million by 2016, which translates to 72% penetration. According to Trai in January 2012 the number of active mobile connections in India has reached 659.99 million and the total mobile subscriber base stands at 903.72 million.  
  
It is estimated that every Indian user spends nearly 52 minutes per-day using mobile apps. Similarly, mobile app economy-estimated at nearly $4 billion (Rs 20,000 crore) in 2009 and expected to grow to $18 billion in half a decade-continues to expand exponentially.

**WEB RTC FUTURE**

The web has revolutionised communication, and [WebRTC](http://www.webrtc.org/) promises to take the revolution a step further. The free, open-source project enables compliant web browsers to communicate in real-time using simple JavaScript APIs.

The project's mission: "To enable rich, high quality, real-time communication applications to be developed in the browser via simple JavaScript APIs and HTML5." Think Skype or FaceTime, built into the browser-and far more once developers start tinkering.

Distinguished Engineer at Avaya, Inc. Alan Johnston co-wrote a [comprehensive book on the technology](http://webrtcbook.com/), and he's truly excited about its potential: "WebRTC is about unleashing the power of real-time communication to any web developer or application. Before WebRTC, to add real-time communication, a developer needed to know protocols and select stacks, often requiring licensing.

Now, WebRTC browsers such as Chrome and Firefox have all this built in, and developers can utilise them with a few simple API calls in their JavaScript." Net benefits, says Johnston, will be lower development costs, richer interaction on websites, and potentially happier end users.

**CHAPTER – 7**

**References**

For the references i have used the following sources:-

1. Wikipedia
2. Web RTC doc by google
3. Multi Peer connectivity frame from wikipedia and apple reference book
4. Objective-C resources i have abstracted from Tutorial point and other online teaching websites
5. Images from google Images.

For the rest of the Images i have took screenshots of my project.

You can download mine project from github .

[**https://github.com/varen1994/webRTCProjectMajor**](https://github.com/varen1994/webRTCProjectMajor)