A PROJECT REPORT

On

Your Garage Sale

Submitted in partial fulfillment of the
Requirements for the Degree of

MASTER OF COMPUTER APPLICATIONS

Submitted By

Abhishek Chaubey
University Roll. No.
1900290149002

Submitted to

Dr. Ankit Verma
(Associated Professor)
Computer Applications

Department of Computer Applications,
KIET Group of Institutions,
Delhi-NCR, Ghaziabad

(July 2021)

DECLARATION

I hereby declare that the work presented in this report entitled “Your Garage Sell”, was carried out by me. I have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute.
I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.
I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable.

Name	: Abhishek Chaubey
Roll. No.	: 1900290149002
Branch	: Master of Computer Applications

(Candidate Signature)

Abhishek chaubey.

CERTIFICATE

Certified that **Abhishek Chaubey** (**1900290149002**) has carried out the project work presented in this report entitled "**Your Garage Seell**" for the award of **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University, Lucknow under my supervision. The report embodies result of original work, and studies are carried out by the student himself and the contents of the report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University.

Dr. Ankit Verma
Associate Professor
Dept. of Computer Applications
KIET Group of Institutions, Ghaziabad

External Examiner

Dr. Ajay Kumar Srivastava
Professor & Head
Department of Computer Application
KIET Group of Institutions, Ghaziabad

Date:

ACKNOWLEDGEMENT

Success in life is never attained single handedly. My deepest gratitude goes to my thesis supervisor, **Dr. Ankit Verma** for her guidance, help and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to Dr. Ajay Kumar Shrivastava, Professor and Head, Department of Computer Applications, for his insightful comments and administrative help at various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Abhishek Chaubey
1900290149002

Table of Contents

ABSTRACT

This project focuses on to build the internal project & Garage Sale. The main purpose of this project is to develop the iPhone base application, which is helpful to the customer in purchasing and selling estate and garage item.

Your Garage Sale is an IPhone based application developed using IPhone and XMLs as back end. This application is developed for the internal development of scripts for the company TheAppGuruz situated at Rajkot.

The project is primarily divided into two major modules

1. Search Item
2. Post Garage Sale

Search Item

The Search module is a heart of this project. This will suggest user to search Items according to their requirement like free, sale, estate, garage etc. also in each type user can search item according to city, state, country and category of an item.

Post Garage Sale

This is a personal Module of the user. In this module user can check its messages, profile, add item to sale, insert its interesting word and category for search, and for support request and feedback

LITERATURE REVIEW

ios :-

Apple iOS (AAPL, GOOG) is the second-most popular mobile computing platform. In September 2017, Apple iOS held a 32 percent share of the mobile market, second only to Android which held a 65.5 percent share.

The first version of iOS was released in June 2007, when the iPhone debuted on the market. iOS, an acronym for iPhone Operating System, is a Unix-based operating system powering all of Apple's mobile devices, but the name iOS was not officially applied to the software until 2008, when Apple released the iPhone software development kit (SDK), enabling any app makers to create applications for the platform.

The overwhelming popularity of the iPhone was certainly driven by the effectiveness of iOS. Nearly 218 million iPhones were sold by the end of 2018, making the device the single most successful product ever released into the market. Some estimates show that since launch in 2007, iOS has been responsible for more than $1 trillion in revenue for Apple.

Over the years, iOS has enabled many advancements which have rippled throughout the culture, impacting those who own iPhones as well as those who do not.

The first version of iOS introduced the culture to the touch-screen smartphone, a significant cultural shift away from flip-phones and Blackberry-style devices. The iPhone combined many functions within a single device, including a camera, internet browser and media player alongside the phone and messaging, and the world would never be the same.

Apple finally gave iOS its name in the second version, when the company also released its SDK to developers looking to build apps for the platform. FaceTime, Apple's video chat software, was released in iOS 4. Version 4 also introduced multitasking capabilities in iOS devices.

iOS 5 delivered Siri, a voice-enabled personal assistant, as well as iMessage as a central messaging system and the iOS Notification Center. Subsequent releases of the software added Airdrop, Touch ID, Apple Pay, and the much-derided Apple Maps mapping system as well as myriad improvements on functionality and design.

Apple released its iOS version 12 on June 4, 2018, delivering myriad improvements to Siri, FaceTime and other key iOS features.

Introduction :-

IOS stands for iPhone operating system. It is a proprietary mobile operating system of Apple for its handheld. It supports Objective-C, C, C++, Swift programming language. It is based on the Macintosh OS X. After Android, it is the world's second most popular mobile operating system. Many of Apple's mobile devices, including the iPhone, iPad, and iPod, run on this operating system. To control the device, iOS employs a multi-touch interface, such as sliding your finger across the screen to advance to the next page or pinching your fingers to zoom in or out of the screen.

Features of IOS Platform :-

iOS has become popular because of its prominent features. The following are the popular features of iOS. Let's get into details.

1. **Multitasking:** iPhone offers multitasking features. It started with iPhone 4, iPhone 3GS. By using the multitasking feature on an iOS device or using a multi finger gesture on an iPad, you can swiftly go from one app to another at any moment.
2. **Social Media:** Sharing content and displaying an activity stream are just a few of the ways iOS makes it simple to integrate social network interactions into the app.
3. **iCloud:** Apple's iCloud is a service that offers Internet-based data storage. It works on all Apple devices and has some Windows compatibility, and it handles most operations in the background. It is highly encrypted. It offers the backup option to help the user to not lose any of their data.
4. **In-App purchase:** In-app purchases, which are available on all Apple platforms, provides users with additional material and services, such as digital

items(iOS, iPadOS, macOS, watchOS), subscriptions, and premium content, right within the app. You may even use the App Store to promote and sell in-app purchases.

5. **Game Center:** Game Center, Apple's social gaming network, adds even more pleasure and connection to your games. Game Center provides access to features such as leaderboards, achievements, multiplayer capability, a dashboard, and more.
6. Notification **Center:** Notification Center is a feature in iOS that shows you all of your app alerts in one place. Rather than needing immediate resolution, it displays notifications until the user completes an associated action. However we can control the notification settings.
7. **Accelerometer:** An accelerometer is a device that detects changes in velocity along a single axis. A three-axis accelerometer is built into every iOS device, providing acceleration readings in each of the three axes. LIS302DL 3-axis MEMS-based accelerometer is used in the original iPhone and first-generation iPod touch.
8. **Gyroscope:** The rate at which a gadget rotates around a spatial axis is measured using a gyroscope. A three-axis gyroscope is found in many iOS devices, and it provides rotation data in each of the three axes.
9. **GPS:** To detect your location, the iPhone uses an inbuilt Assisted GPS (AGPS) chip. You don't even need to instal this function because it's already integrated into your iPhone. Because it provides an approximation of your location based on satellite information, this system is faster than standard GPS.
10. **Accessibility:** Every Apple product and service is built with one-tap accessibility capabilities that work the way you do.
11. **Bluetooth:** Apple supplies the Core Bluetooth framework, which includes classes for connecting to Bluetooth-enabled low-energy wireless technology.
12. **Orientations:** The iOS apps can be used in both portrait and landscape modes. Apple, on the other hand, provides size classes in XCode for creating interfaces in landscape and portrait orientations.
13. **Camera integration:** In iOS, Apple provides the AVFoundation Capture Subsystem, which is a standard high-level architecture for audio, image, and video capture.

14. **Location services:** The Location Services enables applications and websites to access the user’s device location with the user’s permission. When location services are operational, the status bar displays a black or white arrow icon.
15. **Maps:** Apple offers an online mapping service that can be utilised as the iOS default map system. It has a variety of functions, such as a flyover mode. Apple’s MapKit may be used to create applications that utilise maps

INTRODUCTION

In our day to day life, we purchase many things, in them some become becomes useless for us, as time passes we put them in our garage or store room as we don‘t want to throw them out as it costs much and reselling is not that much easy for us.

Same thing happens for buying second hand things, if people want to cut the cost, then they probably prefer good quality second hand things to purchase. But how can we know that who want to sell their old goods nearby our place..?

YourGarageSale is a very good solution to this problem...

One just need to install YGS in his mobile and he can purchase or sell any second hand item from anywhere and to anywhere. YGS helps its users to find out any secondhand item nearby the specified location.

In addition to this, YGS allows users to find out location of any item using Google maps. User can also link his/her account with Facebook and/or eBay. User can directly contact the item owner he/she wants to purchase as well as can email item details to his friends directly from the application.

YGS also lists the items its users want to sell. This means you can both purchase and sell second hand things by using his internet enabled, iPhone or android mobile phone...

This application is made for android and iPhone mobile phones and a website is also available on web named www.yourgaragesale.com.

Once registered, user can access his/her account from any of these medium.

FEATURES OF YOUR GARAGE SALE

Listed below are the features of Your Garage Sale: -

Login & Registration Features

The Login & Registration Features is a heart of this project. Only User can use this module. User manages entire application.

Login Functionality Registration (Create Account) Forgot Password Functionality

Selling Feature :-

:- The user has choice to sale its product
:- The selling include on rent, sale, free and more
:- User can suggest any item to friend
:- User can post any item to eBay or facebook a/c from this app. User can add its favorite criteria, category to search item

Feedback Features :-

:- The Feedback Feature is give information about people response about products this module having functionality like.

:- User can write feedback to particular product
:- User can suggest to Friends About Product.
:- User can also see their own feedback list.
:- User can Navigate different category list and select User can ask for support request also.

Search Feature :-

:- Search Feature is having functionality like
:- User can Search free products, on rent product etc.
:- User can search product Near Its location
:- User can Search liquidation products
:- User can Search product in country, state, city and category wise.

Post Garage Feature:-

:- The Post Garage Feature is having functionality like. User can add its product for sale.

:- User can also add the new feedback for the same product, which is in his stuff. User can post in Facebook or eBay and also check messages,
User can enter its words of interest and favorite category

Web Services Feature:-

:- The Web Services is very useful Feature to manage the database. The entire database is handle through the Web Services.

:- All the Web Services is written in this Feature.
:- With the help of this module user can retrieve data from the server.
:- User can also send the data to the Server.
:- There is some calculation also done in this feature like Desirability Value.

:- All the web services in this module return the data in the JSON or XML format.

ADVANTAGES OF Your Garage Sale WORKS?

:- Improve users service and satisfaction.
:- User can allow giving their choice or suggestion about products.

:- User can efficiently sale their Products.
:- User can see their feedback response and support request. Time saving process.

:- Reduce Advertisement cost.
:- To enhance the better quality of product.

PROJECT DEFINITION

YOUR GARAGE SALE is a best IPhone application for user. They can simple register with their iPhone to navigate registration process for take the benefit of YOUR GARAGE SALE. User sees the various products list and navigate through various category of that product.

User register with this application to get more benefit of product usage and buy good quality product from the list of choice in the market and get more joy of use good brand product and also can sale its own product.

User can also see the various new brand of the company and get more feature of the product also get time save from the old product what they use.

User can add feedback on particular products and also create stuff.

User can also share its product On Facebook and eBay.

SCOPE

The scope of the project is spread among the registered users. Users get fully benefit after the registration. YOUR GARAGE SALE provides the best services for product sale and purchase. This application records all the transaction into the server database. User can see all the list of product meet its criteria.

YOUR GARAGE SALE is open to all users, who get benefit of products. Your Garage sale is tied with eBay and allows user to sale its product in eBay from this app.

YGS work at the interaction of mobile, social and search. YGS develops and applies emotional technologies to benefit brands and consumers.

YGS focus on emotional engagement and satisfaction. Our data helps consumers Sale and discover great experiences. And lets Buy product According to their need.

YGS are digital ethnographers. Our members can give sale and purchase products using a smartphone.

Experience begins long before a purchase: researching, choosing, and using are also part of them.

SYSTEM REQUIRMENT SPECIFICATION

USER CHARACTERISTICS:-

YOUR GARAGE SALE is very simple for anyone to use. Any users who want to use this application that user have to simple knowledge of how iPhone works and Internet.

HARDWARE AND SOFTWARE REQUIREMENT

Hardware Requirement

1. 2 GB RAM.
2. Intel Core 2 Duo Processor.
3. 160 GB HDD Space.

Software Requirement

System Development Tools

1. XCODE IDE 4.0.
2. MySQL 5.1.3.6.
3. Operating System: MAC OS X10.6 Snow Leopard.

System Deployment Tools

1. IPhone Simulator 4.3.
2. IPhone 4.
3. IPad.

System Design Tools

1. http://www.gliffy.com UML Diagrammed.
2. Adobe Photoshop CS4.

System Documentation Tools

1. MS Word 2007.
2. MS PowerPoint 2007.

Coding Architecture

• COCOA Model View Controller (MVC) Architecture.

TOOLS & TECHNOLOGY DETAILS

Develop for Mac OS X:-

Mac OS X is the world's most advanced operating system. The Mac is built upon a proven UNIX® foundation, coupled with a GPU- accelerated desktop, intuitive networking services, and system- wide optimization for multicore CPUs. Using the Cocoa frameworks, every application automatically inherits the native look, feel, and behaviors of the Mac platform. Beautiful, fast, and exciting applications are not an accident on Mac OS X — the entire system was crafted to deliver amazing experiences.

The best part is that your Mac already has everything you need to create world-class applications. The same Xcode toolset used to develop Mac OS X and iOS are included with every Mac. Code editor. Debugger. Compilers. GUI designer. Performance analysis. It's all there. The tools and Cocoa frameworks work seamlessly together, making it easy to create gorgeous, feature-rich Mac OS X applications.

Develop for IOS :-

IOS is the world's most advanced mobile platform, redefining what can be done with a mobile device. The iOS SDK combined with Xcode tools make it easy to create apps that perform feats never before attempted. IOS 4 delivers several new multitasking services that allow your apps to perform tasks in the background while preserving battery life and performance. With the App Store present on every iOS device, and localized around the world, there is simply no platform more compelling for mobile developers.

What Is Cocoa? :-

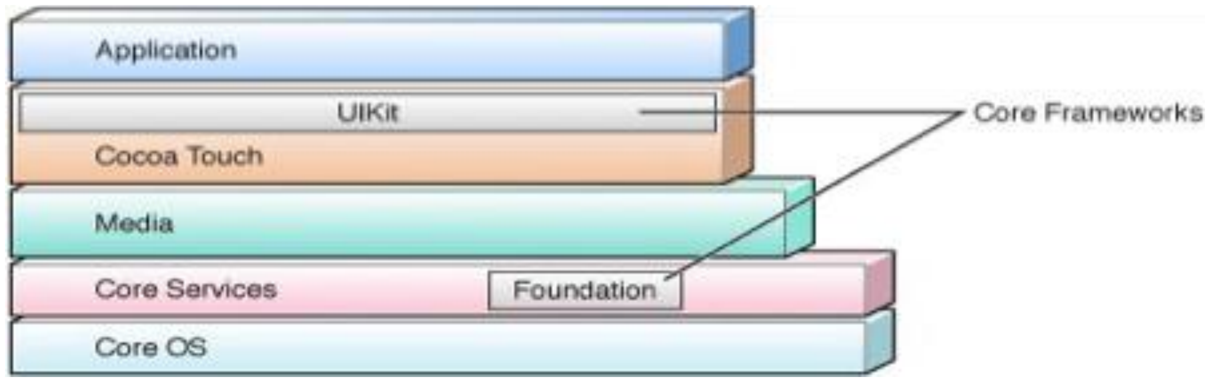
Cocoa is an application environment for both the Mac OS X operating system and iOS, the operating system used on Multi-Touch devices such as iPhone, iPad, and iPod touch. It consists of a suite of object-oriented software libraries, a runtime system, and an integrated development environment.

The Cocoa Environment :-

Cocoa is a set of object-oriented frameworks that provides a runtime environment for applications running in Mac OS X and iOS. Cocoa is the preeminent application environment for Mac OS X and the only application

Environment for iOS. (Carbon is an alternative environment in Mac OS X, but it is a compatibility framework with procedural programmatic interfaces intended to support existing Mac OS X code bases.) Most of the applications you see in Mac OS X and iOS, including Mail and Safari, are Cocoa applications. An integrated

Development environment called Xcode supports application development for both platforms. The combination of this development environment and Cocoa makes it easy to create well- factored, full-featured application.



Core OS. This level contains the kernel, the file system, networking infrastructure, security, power management, and a number of device drivers. It also has the libSystem library, which supports the POSIX/BSD 4.4/C99 API specifications and includes system-level APIs for many services.

Core Services. The frameworks in this layer provide core services, such as string manipulation, collection management, networking, URL utilities, contact management, and preferences. They also provide services based on hardware features of a device, such as the GPS, compass, accelerometer, and gyroscope. Examples of frameworks in this layer are Core Location, Core Motion, and System Configuration.

This layer includes both Foundation and Core Foundation, frameworks that provide abstractions for common data types such as strings and collections. The Core Frameworks layer also contains Core Data, a framework for object graph management and object persistence.

Media. The frameworks and services in this layer depend on the Core Services layer and provide graphical and multimedia services to the Cocoa Touch layer. They include Core Graphics, Core Text, OpenGL ES, Core Animation, AVFoundation, Core Audio, and video playback.

Cocoa Touch. The frameworks in this layer directly support applications based in iOS. They include frameworks such as Game Kit, Map Kit, and iAd.

What is Xcode Project? :-

Xcode is a powerful professional tool that
allows you to perform the most common
software development tasks simply, quickly, and
in a way that should be familiar. Its capabilities,
however, are much more powerful than what is
needed to build just a single type of application.
Xcode is designed to let you build any kind of
software product you can dream of. From Cocoa
and Carbon applications to kernel extensions and Spotlight importers, Xcode is

up to the task. Xcode's unique interface is designed to help you navigate your source code in a variety of ways and provides access to the large amount of functionality in the underlying toolset—including GCC, javac, jikes, and GDB—that is used to build software products. It's a tool designed by professionals for professionals.

Xcode Projects Defined :-

All activity in Xcode, from creating and editing files to building and debugging applications, revolves around projects. Xcode projects organize and give access to all of the files and resources that are used to build a software product. Regardless of what you are building, Xcode manages three kinds of information to build your product:

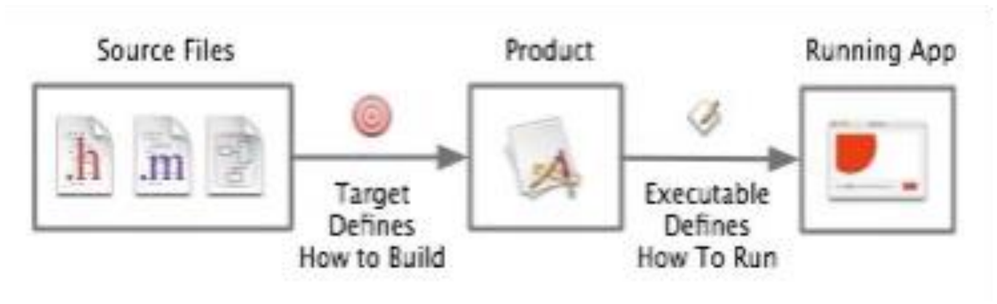
Source file references that include source code, images, localized string files, data models, and more.

Targets that define the products to build. A target organizes the files and instructions needed to build a product into a sequence of build actions that can be taken.

Executable environments in which you can run and test a software product. An executable environment defines the program that should be used to run the product with. In many cases, this will be the product itself, but doesn't have to be. In addition, the executable environment defines any command-line arguments and environment variables, which should be used.

These three elements of an Xcode project come together as shown in diagram

Xcode builds projects from source code written in C, C++, Objective-C, and Objective- C++. It generates executables of all types supported in Mac OS X, including command-line tools, frameworks, plug-ins, kernel extensions, bundles, and applications. (For iOS, only application executables are possible.) Xcode permits almost unlimited customization of build and debugging tools, executable packaging (including information property lists and localized bundles), build processes (including copy-file, script-file, and other build phases), and the user interface (including detached and multi view code editors). Xcode also supports several source-code management systems—namely CVS, Subversion, and Perforce—allowing you to add files to a repository, commit changes, get updated versions, and compare versions.



Xcode builds projects from source code written in C, C++, Objective-C, and Objective- C++. It generates executables of all types supported in Mac OS X, including command-line tools, frameworks, plug-ins, kernel extensions, bundles, and applications. (For iOS, only application executables are possible.) Xcode permits almost unlimited customization of build and debugging tools, executable packaging (including information property lists and localized bundles), build processes (including copy-file, script-file, and other build phases), and the user interface (including detached and multi view code editors). Xcode also supports several source-code management systems—namely CVS, Subversion, and Perforce—allowing you to add files to a repository, commit changes, get updated versions, and compare versions.

Xcode Project development environment screen :-

The Objective-C Programming Language

History of Objective-C :-

The Objective-C programming language has had a humble history. Created by Brad Cox in the early 1980s as an extension of the venerated C, pioneered a decade earlier by Dennis Ritchie, the language was based on another called SmallTalk-80.NeXT Software licensed the language in the 1988, and developed a code library called NeXTSTEP. NeXTSTEP in all its Glory.

When Apple Computer acquired neXT in 1996, the NeXTSTEP code library was built into the core of Apple's operating system, Mac OS X. NeXTSTEP provided Apple with a modern OS foundation, which Apple could not produce on its own.

The iPhone's operating system, currently dubbed iOS, is based off of a reduced version of OS X. Therefore; iOS inherits most of the NeXTSTEP code library, along with extensive modernization and optimizations. Because NeXTSTEP was built from Objective-C, iOS mirrors the language choice. This made it easy for OS X developers to begin creating apps for the iPhone and iPod Touch.

Apple added a number of features to the Objective-C language, extending its functionality to parallel that of other languages that were beginning to arise. This major update was labeled Objective-C 2.0, and remains the language of choice for both OS X and iOS. This iteration will be covered in this tutorial.

What is Objective-C? :-

The Objective-C language is a simple computer language designed to enable sophisticated object-oriented programming. Objective-C is defined as a small but powerful set of extensions to the standard ANSI C language. Its additions to C are mostly based on Smalltalk, one of the first object-oriented programming languages. Objective-C is designed to give C full object-oriented programming capabilities, and to do so in a simple and straightforward way.

Most object-oriented development environments consist of several parts: An object-oriented programming language

:- A library of objects

:- A suite of development tools

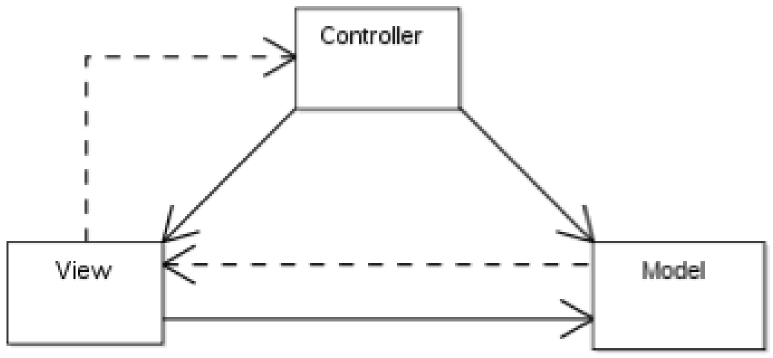
:- A runtime environment

Model View Controller (MVC) :-

Model–View–(MVC) is an architectural pattern used in software engineering. Successful use of the pattern isolates business logic from user interface considerations, resulting in an application where it is easier to modify either the visual appearance of the application or the underlying business rules without affecting the other. In MVC, the model represents the information (the data) of the application; the view corresponds to elements of the user interface such as text, checkbox items, and so forth; and the controller manages the communication of data and the business rules used to manipulate the data to and from the model.

In simpler words-

1. Model handles all our database logic. Using the model we connect to our database and provide an abstraction layer.
2. Controller represents all our business logic i.e. all if and else.
3. View represents our presentation logic i.e. our HTML/XML/JSON code.



MySQL :-

MySQL is an open source relational database management system (RDBMS) That uses Structured Query Language (SQL), the most popular language for adding, accessing, and processing data in a database MySql is noted mainly for its speed, reliability, and flexibility. It is commonly employed with most of the popular server-side scripting languages including PHP, JSP and ASP. It is a multithreaded, multi-user, SQL (Structured Query Language) relational database server (RDBMS). MySql is available either under the GNU General Public License (GPL) or under other licenses when the GPL is inapplicable to the intended use. MySQL is a freely available third-party database engine designed to provide fast access to stored data. Data can be stored, updated and deleted using languages such as PHP. The data can be retrieved from the database to allow the generation of dynamic Web Pages.

MySQL is an open source relational
management system (RDBMS) That uses

Query Language (SQL), the most popular
for adding, accessing, and processing data in
MySql is noted mainly for its speed, reliability,
flexibility. It is commonly employed with most of the popular server-side scripting languages including PHP, JSP and ASP. It is a multithreaded, multi-user, SQL (Structured Query Language) relational database server (RDBMS). MySql is available either under the GNU General Public License (GPL) or under other licenses when the GPL is inapplicable to the intended use. MySQL is a freely available third-party database engine designed to provide fast access to stored data. Data can be stored, updated and deleted using languages such as PHP. The data can be retrieved from the database to allow the generation of dynamic WebPages.

Proposed System Over Current System :-

Current System :-

- Our current Sale system is very time consuming and headache. It is very hard to Sale or purchase desired product near our location.
- If someone only wants to see all the product of company which is available in the market and which having good functionality it is not possible through current system.
- It is not physible for every user to get advice from their clings.
- When user don't know about product type selling category and how seller want to sale. Also user can add its own product with 10 photos of single product. User can search product near its location.

Proposed System :-

YGS is such great things get more and more products and Suggestions from the Friends and the different user.

- To search product is very easy task in YGS application.
- User can start to Search Product In USA and Canada. User can see the location of product in map.
- YGS work at the interaction of mobile, social and search. YGS develops and applies emotional technologies to benefit User.
- YGS focus on emotional engagement and satisfaction. Our data helps consumers share and Sale products. And lets brand Purchase products.
- YGS are digital ethnographers. Our members can Sale and purchase products using a smartphone.

Feasibility Study :-

Feasibility study means to determine whether the current system is feasible in the following respects:

Feasibility analysis is a cross life cycle activity, which has to be continuously performed throughout the system development. By using the creeping commitment approach, feasibility is measured at different times in the system development. This evaluation ensures that the project is beneficial and practical to an organization.

There are four categories of feasibility analysis:

- Technical Feasibility
- Economic Feasibility
- Schedule Feasibility
- Operational Feasibility

Technical Feasibility :-

Technically the systems configuration is less complex. This system is technically feasible as tools used are easy to use and comfortable as well as user can easily operate the system.

Economical Feasibility :-

Economic analysis is the most frequently used technique for evaluating the effectiveness of a proposed system. More commonly known as Cost / Benefit analysis the procedure is to determine the benefits and savings that are expected from a proposed system. The system is economically feasible when the tools used are easily available in the market.

Operational Feasibility :-

It is mainly related to the aspects to operate the system. The points to be consider is: The system should be easy to operate by the system user and user does not require any special knowledge to operate this system.

Schedule Feasibility :-

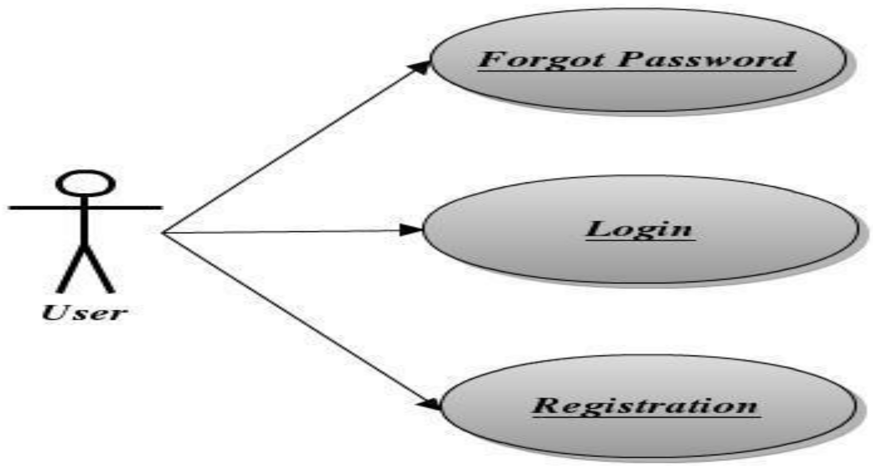
It is very important that the project gets completed within the time limit. With three members in the team, this division of work will allow us to hold the time limit and

complete the project within this time limit. So, with this we conclude that the project is feasible with respect to schedule.

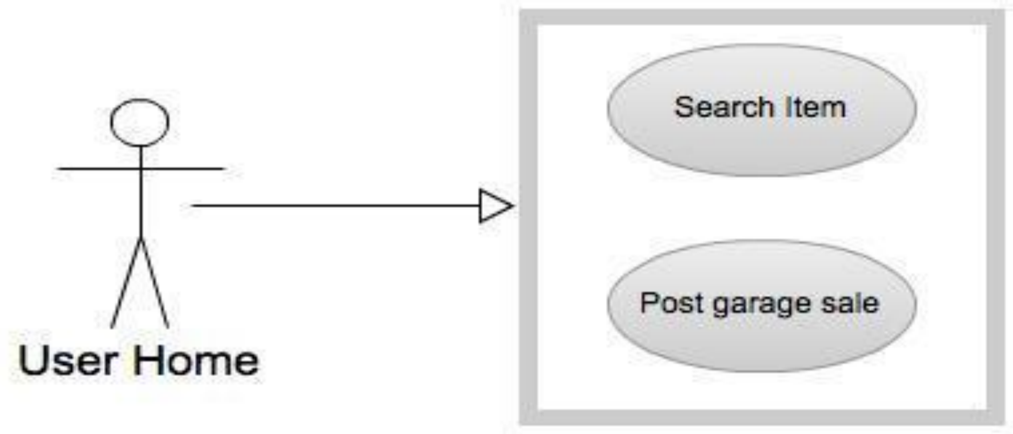
System Design :-

Use Case Diagram :-

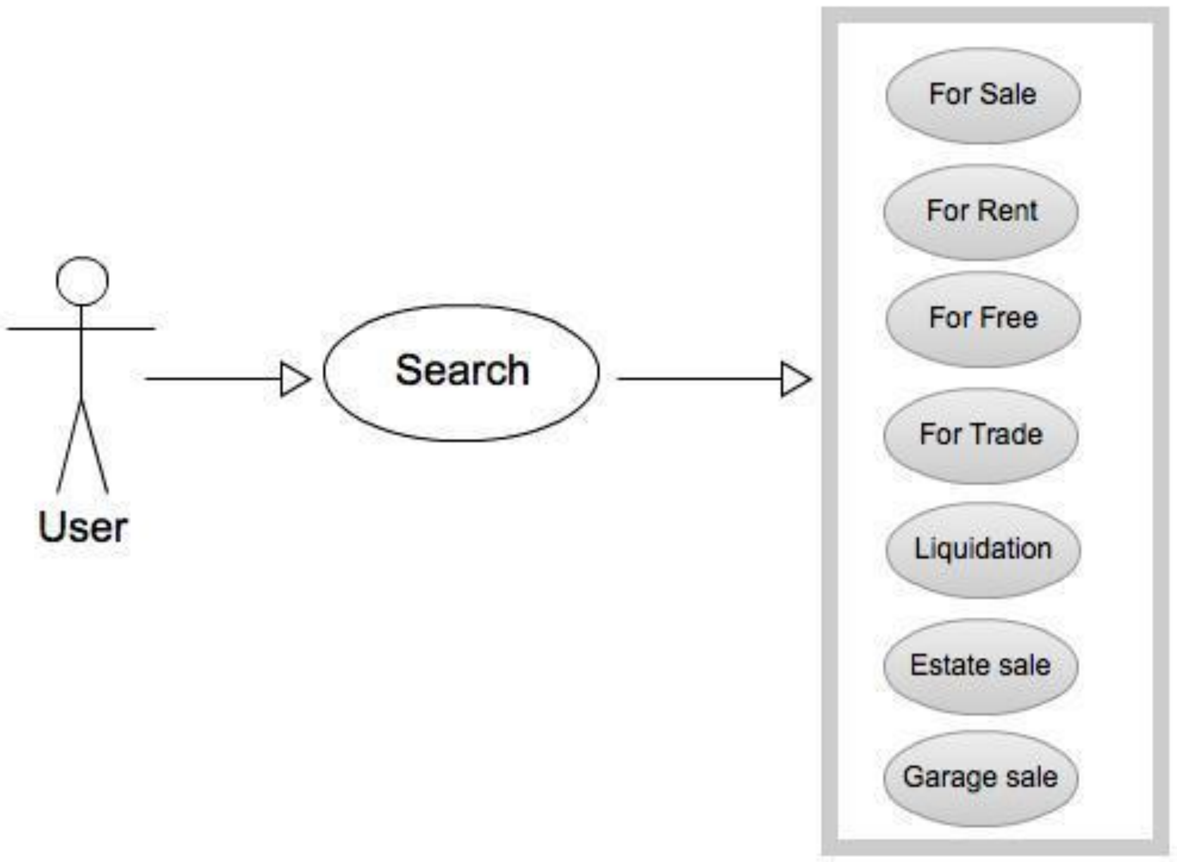
Use Case Diagram For User Registration



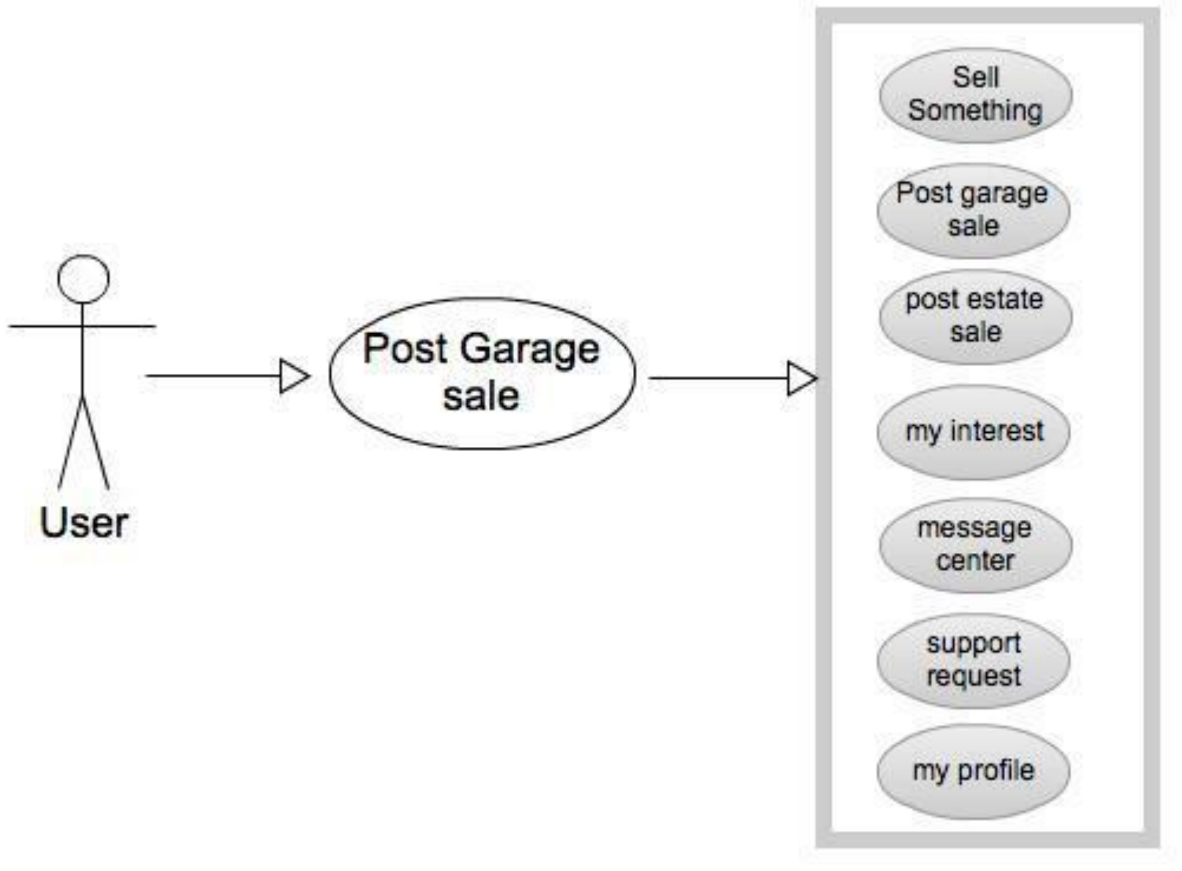
Use Case Diagram User Home :-



Use Case Diagram for Search Menu :-

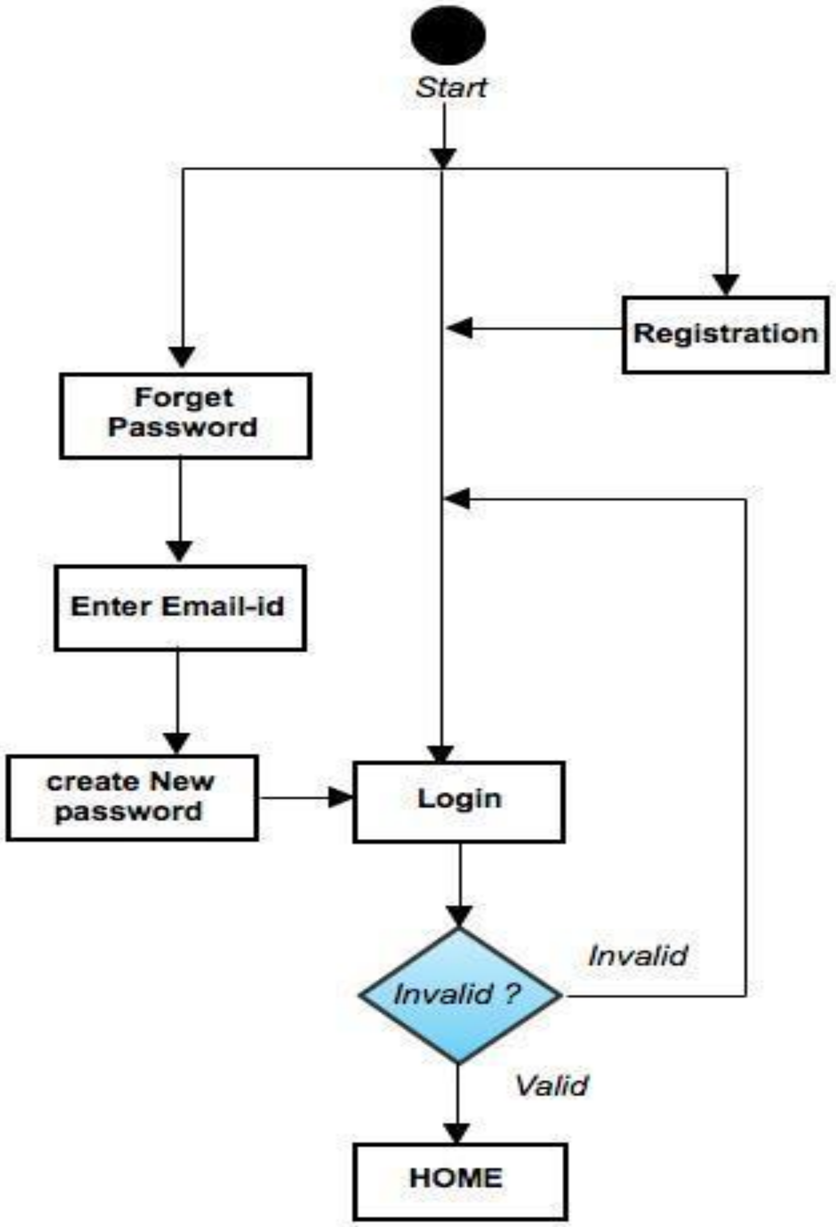


Use Case Diagram for post garage menu :-

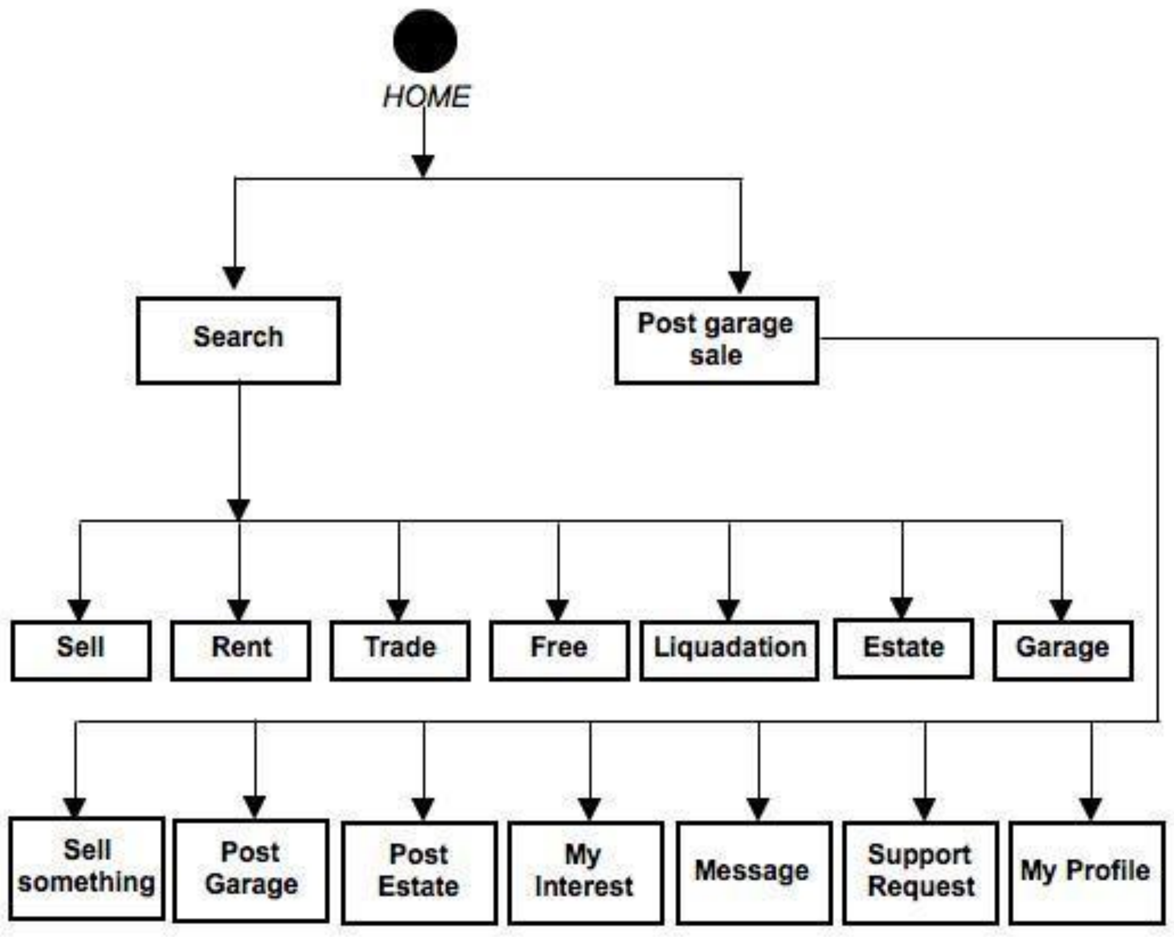


Activity Diagram :-

Activity Diagram for User :-


Start
Registration
Forget Password
Enter Email-id
create New password
Login
Invalid ?
Invalid
Valid
HOME

Activity Diagram for Home :-



CODING

Classes

/*

This file is generated and isn't the actual source code for this

managed global class.

This read-only file shows the class's global constructors,

methods, variables, and properties.

To enable code to compile, all methods return null.

*/

```
//
//  ViewController.swift
//  switchInCell
//
//  Created by Abhishek Chaubey'S MacBook Pro on 05/05/21.
import UIKit

class MySecondViewController: UIViewController, UITableViewDelegate,
UITableViewDataSource {


    @IBOutlet weak var myTableView: UITableView!

    let options = ["All Notification", "Buy Notification", "Reedem Notification", "Bids
Notification", "News Notification", "Recieve Notification"]

    override func viewDidLoad() {
        super.viewDidLoad()
    }
    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int)
-> Int {
        return options.count

    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -
> UITableViewCell {
        let cell  = tableView.dequeueReusableCell(withIdentifier: "cell")!
        cell.textLabel?.text = options[indexPath.row]
        let swicthView = UISwitch(frame: .zero)
        swicthView.setOn(false, animated: true)
        swicthView.tag = indexPath.row
        swicthView.addTarget(self, action:#selector(self.swicthChanged(sender:)),
for: .valueChanged)
        cell.accessoryView = swicthView
        return cell
    }
    @objc func swicthChanged(sender: UISwitch!){

        print("Table Row Switch Changed\(sender.tag)")
        print("The Switch Is\(sender.isOn ? "ON" : "OFF")")
    }


}

class headViewController: UIViewController {

    var edata = [

        EntertainmentApp(sectionType: "sandeep kumar", imageGallary:
["PUBG","PUBG","PUBG"]),
        EntertainmentApp(sectionType: "pradeep  kumar", imageGallary:
["PUBG","PUBG","PUBG"]),

    ]

    var images = ["PUBG","PUBG2","PUBG3","FreeFire"]
    var games = ["Trending Games","Live Bidding Game"]
    var currentcellindex = 0
    var timer:Timer?
    @IBOutlet weak var tableView: UITableView!


    @IBOutlet weak var headPageControl: UIPageControl!

    @IBOutlet weak var collectionView: UICollectionView!
    override func viewDidLoad() {
        super.viewDidLoad()
        collectionView.delegate = self
        collectionView.dataSource = self
        headPageControl.numberOfPages = images.count
        /*Timer.scheduledTimer(timeInterval: 3.0, target: self,
selector:#selector(slidetoNext), userInfo: nil, repeats: true)*/

    }
   @objc func slidetoNext()
   {
    if currentcellindex < images.count-1
    {
        currentcellindex = currentcellindex + 1
    }
    else{
        currentcellindex = 0
    }
    headPageControl.currentPage = currentcellindex
    collectionView.scrollToItem(at: IndexPath(item: currentcellindex, section: 0), at:
.right, animated: true)
   }

}
extension headViewController:
UICollectionViewDelegate,UICollectionViewDataSource,UICollectionViewDelegate
FlowLayout{
    func collectionView(_ collectionView: UICollectionView,
numberOfItemsInSection section: Int) -> Int {
        return images.count
    }

    func collectionView(_ collectionView: UICollectionView, cellForItemAt
indexPath: IndexPath) -> UICollectionViewCell {
        let cell = collectionView.dequeueReusableCell(withReuseIdentifier:
"headCollectionViewCell", for: indexPath) as! headCollectionViewCell
        cell.headImg.image = UIImage(named: images[indexPath.row])
        return cell
    }
    func collectionView(_ collectionView: UICollectionView, layout
collectionViewLayout: UICollectionViewLayout, sizeForItemAt indexPath:
IndexPath) -> CGSize {
        return CGSize(width: collectionView.frame.width, height:
collectionView.frame.height)
    }


}
extension headViewController: UITableViewDelegate,UITableViewDataSource{

    func tableView(_ tableView: UITableView, heightForRowAt indexPath:
IndexPath) -> CGFloat {
        return 200
    }
    func numberOfSections(in tableView: UITableView) -> Int {
        return edata.count
    }
    func tableView(_ tableView: UITableView, titleForHeaderInSection section: Int) -
> String? {
        return edata[section].sectionType
    }

    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int)
-> Int {
        return 2
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -
> UITableViewCell {
        let  cell = tableView.dequeueReusableCell(withIdentifier:
"gamesTableViewCell", for: indexPath) as! gamesTableViewCell
        cell.gamesCollectionView.tag = indexPath.section
        return cell
    }
    func tableView(_ tableView: UITableView, willDisplayHeaderView view: UIView,
forSection section: Int) {
        view.tintColor = .yellow
    }


}

class ViewController: UIViewController {


    @IBOutlet var cities: [UIButton]!

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
    }


    @IBAction func bttnSelectCity(_ sender: UIButton) {
        cities.forEach { (Button) in
            UIView.animate(withDuration: 0.3,animations:{
                Button.isHidden = !Button.isHidden
                self.view.layoutIfNeeded()
            })
        }

        }



    @IBAction func bttnCity1(_ sender: UIButton) {
    }


    @IBAction func bttnCity2(_ sender: UIButton) {
    }


    @IBAction func bttnCity3(_ sender: UIButton) {
    }



    @IBAction func bttnCity4(_ sender: UIButton) {
    }

}

class TableViewCell: UITableViewCell {

    @IBOutlet weak var nameLbl: UILabel!

    @IBOutlet weak var inputTextField: UITextField!


    @IBOutlet weak var AlertLbl: UILabel!

    @IBOutlet weak var textfield: UITextField!
    @IBOutlet weak var txtFName: UITextField!
    @IBOutlet weak var txtEmail: UITextField!
    @IBOutlet weak var txtAddress: UITextField!
    @IBOutlet weak var txtCity: UITextField!
    @IBOutlet weak var txtCountry: UITextField!
    @IBOutlet weak var txtPassword: UITextField!
    @IBOutlet weak var txtConPassword: UITextField!

    @IBOutlet weak var AlertName: UILabel!
    @IBOutlet weak var AlertFName: UILabel!
    @IBOutlet weak var AlertEmail: UILabel!
    @IBOutlet weak var AlertAddress: UILabel!
    @IBOutlet weak var AlertCity: UILabel!
    @IBOutlet weak var AlertCountry: UILabel!
    @IBOutlet weak var AlertPassword: UILabel!
    @IBOutlet weak var AlertConPassword: UILabel!

class signUpViewController: UIViewController{

    //var universityBlocks = ["Physics","Chemistry","Mathematics","Hindi","CSE"]//

    var inputPlaceHolder = ["Name", " Father's name",
"Email","Address","City","Country","Password","Confirm Password"]


    var titleLbl = ["Name", " Father's name",
"Email","Address","City","Country","Password","Confirm Password"]

    var alertLbl = ["please Enter Your Name","please Enter Your Father's
Name","please Enter Your Email ","please Enter Your Address","please Enter Your
City","please Enter Your Country","please Enter Your Password","please Enter
Your Confirm Password",]




    @IBOutlet var subjects: [UIButton]!







    @IBOutlet weak var tableView: UITableView!

    @IBOutlet weak var maleImg: UIImageView!


    @IBOutlet weak var femaleImg: UIImageView!

    @IBOutlet weak var femaleBttn: UIButton!
    @IBOutlet weak var maleBttn: UIButton!
    override func viewDidLoad() {
        super.viewDidLoad()
        tableView.delegate = self
        tableView.dataSource = self

        // Do any additional setup after loading the view.
    }


    @IBAction func submitBttnClicked(_ sender: UIButton) {









        let vc = storyboard?.instantiateViewController(identifier: "newViewController")
as! newViewController
        self.navigationController?.pushViewController(vc, animated: true)






    }
    @IBAction func buttnSelectSubjects(_ sender: UIButton) {
        subjects.forEach { (Button) in
            UIView.animate(withDuration: 0.2,animations:  {
                Button.isHidden = !Button.isHidden
                self.view.layoutIfNeeded()
            })
        }
    }


    @IBAction func maleBttnAction(_ sender: UIButton) {

        self.maleBttn.setTitle("male tapped", for: .normal)
    }

    @IBAction func femaleBttnAction(_ sender: UIButton) {
        self.femaleBttn.setTitle("female tapped", for: .normal)

    }

}
extension signUpViewController : UITableViewDataSource,UITableViewDelegate{
    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int)
-> Int {
        return inputPlaceHolder.count

    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -
> UITableViewCell {



        let cell = tableView.dequeueReusableCell(withIdentifier: "TableViewCell", for:
indexPath) as! TableViewCell

        cell.inputTextField.placeholder = inputPlaceHolder[indexPath.row]
        cell.nameLbl.text = titleLbl[indexPath.row]
        cell.AlertLbl.text = alertLbl[indexPath.row]





        return cell


}

    }


class FirstViewController: UIViewController {






    @IBOutlet weak var lEmailTextField: UITextField!
    @IBOutlet weak var lPassTextField: UITextField!

    override func viewDidLoad() {
        super.viewDidLoad()


    }



    @IBAction func LoginUser(_ sender: UIButton) {
        if let email = lEmailTextField.text, let Password = lPassTextField.text {
            if !email.isValidEmail {
                let alert = UIAlertController(title: "Alert", message: "Please Enter Your
Valid Email or Phone No. ", preferredStyle: UIAlertController.Style.alert)
                alert.addAction(UIAlertAction(title: "OK", style:
UIAlertAction.Style.default, handler: nil))
                self.present(alert, animated: true, completion: nil)
            }
            else if !Password.isValidPassword(){
                let alert = UIAlertController(title: "Alert", message: "Please Enter Your
Valid Password", preferredStyle: UIAlertController.Style.alert)
                alert.addAction(UIAlertAction(title: "OK", style:
UIAlertAction.Style.default, handler: nil))
                self.present(alert, animated: true, completion: nil)
            }
            else{

            }
        }else{
            //openAlert ki coding for Nil value
        }
    }

    @IBAction func forgotPasswordClicked(_ sender: Any) {


    }



    @IBAction func signUpClicked(_ sender: Any) {
        let vc = self.storyboard?.instantiateViewController(identifier:
"signUpViewController") as! signUpViewController
        self.navigationController?.pushViewController(vc, animated: true)
    }

   //Extension for all the Validatiuons
```

```
}
extension String {
  var isValidEmail: Bool {
    let regularExpressionForEmail = "[A-Z0-9a-z._%+-]+@[A-Za-z0-9.-]+\\.[A-Za-
z]{2,64}"
    let testEmail = NSPredicate(format:"SELF MATCHES %@",
regularExpressionForEmail)
    return testEmail.evaluate(with: self)
  }
  var isValidPhone: Bool {
    let regularExpressionForPhone = "^\\d{3}-\\d{3}-\\d{4}$"
    let testPhone = NSPredicate(format:"SELF MATCHES %@",
regularExpressionForPhone)
    return testPhone.evaluate(with: self)
  }

   func isValidPassword() -> Bool {
      // least one uppercase,
      // least one digit
      // least one lowercase
      // least one symbol
      //  min 8 characters total
      let password = self.trimmingCharacters(in: CharacterSet.whitespaces)
      let passwordRegx = "^(?=.*?[A-Z])(?=.*?[a-z])(?=.*?[0-
9])(?=.*?[#?!@$%^&<>*~:`-]).{8,}$"
      let passwordCheck = NSPredicate(format: "SELF MATCHES
%@",passwordRegx)
      return passwordCheck.evaluate(with: password)

}

}

class ViewController: UIViewController {


   @IBOutlet var cities: [UIButton]!

   override func viewDidLoad() {
      super.viewDidLoad()
      // Do any additional setup after loading the view.
   }


   @IBAction func bttnSelectCity(_ sender: UIButton) {
      cities.forEach { (Button) in
         UIView.animate(withDuration: 0.3,animations:{
               Button.isHidden = !Button.isHidden
               self.view.layoutIfNeeded()
         })
      }

      }



   @IBAction func bttnCity1(_ sender: UIButton) {
   }


   @IBAction func bttnCity2(_ sender: UIButton) {
   }


   @IBAction func bttnCity3(_ sender: UIButton) {
   }



   @IBAction func bttnCity4(_ sender: UIButton) {
   }

}

import UIKit

class sideMenuViewController: UIViewController {

   @IBOutlet weak var mainBgView: UIView!

   @IBOutlet weak var profilePic: UIImageView!

   override func viewDidLoad() {
      super.viewDidLoad()

      self.setupHamburgerUI()


   }

   @IBAction func clickOnButton(_ sender: Any) {
   }
```

```
private func setupHamburgerUI()
{
    self.mainBgView.layer.cornerRadius = 30

    self.mainBgView.clipsToBounds = true


    self.profilePic.layer.cornerRadius = 35
    self.profilePic.clipsToBounds = true

}
}

class ViewController: UIViewController, UIPickerViewAccessibilityDelegate,
UIPickerViewDataSource {

    var modelObj = modelClass()

    @IBOutlet weak var tableView: UITableView!


    @IBOutlet weak var checkboxBtn: UIButton!


    @IBOutlet weak var TnCbtn: UIButton!


    @IBOutlet weak var registerBtn: UIButton!

    @IBOutlet weak var loginBtn: UIButton!


    @IBOutlet weak var gender: UITextField!

    @IBOutlet weak var country: UITextField!

    @IBOutlet weak var state: UITextField!


    @IBOutlet weak var city: UITextField!

    var textFieldPlaceHolder = ["Username","Email Address", "First Name","Last
Name"]

    var gndr = ["Male", "Female"]
    var cntry = ["United States of America", "Australia", "Russia", "China", "India",
"Japan", "United Kingdom"]
```

```
    var states = ["Bihar", "Gujrat", "Panjab", "Maharastra", "Uttar Pradesh", "West
Bangal", "Madhya Pradesh", "Kerla"]

    var cities = ["Delhi", "Mumbai", "Chennai", "Kolkata", "Ahemdabad", "Varanasi",
"Lucknow", "Ghaziabad", "Meerut"]
    var pickerView = UIPickerView()



    override func viewDidLoad() {
        super.viewDidLoad()

        pickerView.delegate = self
        pickerView.dataSource = self

        // Do any additional setup after loading the view.
        city.inputView = pickerView
        city.textAlignment = .center
        city.placeholder = "Select City"

        gender.inputView = pickerView
        gender.textAlignment = .center
        gender.placeholder = "Select Gender"

        country.inputView = pickerView
        country.textAlignment = .center
        country.placeholder = "Select Country"

        state.inputView = pickerView
        state.textAlignment = .center
        state.placeholder = "Select City"



    }

    public func numberOfComponents(in pickerView: UIPickerView) -> Int {
        return 1
    }
    func pickerView(_ pickerView: UIPickerView, numberOfRowsInComponent
component: Int) -> Int {
        return cities.count
    }

    func pickerView(_pickerView: UIPickerView, titleForRow row: Int, forComponent
component: Int) -> String? {
        return cities[row]

    }

    func pickerView(_pickerView: UIPickerView, didSelectRow row: Int,
inComponent component: Int)  {
        city.text = cities[row]
        city.resignFirstResponder()

    }

}

extension ViewController : UITableViewDelegate,UITableViewDataSource {
    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int)
-> Int {
        return textFieldPlaceHolder.count
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -
> UITableViewCell {

        switch indexPath.row{

            case 0:

            let cell = tableView.dequeueReusableCell(withIdentifier:
"registerTableViewCell", for: indexPath as IndexPath) as!  registerTableViewCell
                cell.textName.tag = 100
                cell.textName.tag = indexPath.row + 100
                cell.textName.placeholder = textFieldPlaceHolder[indexPath.row]
                cell.textName.delegate = self
                cell.textName.autocapitalizationType = .words
                cell.textName.text = modelObj.firstname
                cell.textName.isSecureTextEntry = false
                cell.errorLbl.text = (indexPath.row == modelObj.errorIndex ?
modelObj.errormessage : "")

                return cell



            case 1:
            let cell = tableView.dequeueReusableCell(withIdentifier:
"registerTableViewCell", for: indexPath as IndexPath) as! registerTableViewCell

                cell.textName.tag = indexPath.row + 100
                cell.textName.placeholder = textFieldPlaceHolder[indexPath.row]
                cell.textName.delegate = self
                cell.textName.keyboardType = .emailAddress
                cell.textName.isSecureTextEntry = false
                cell.textName.text = modelObj.email
                cell.errorLbl.text = (indexPath.row == modelObj.errorIndex ?
modelObj.errormessage : "")

            return cell

                 case 2:
                 let cell = tableView.dequeueReusableCell(withIdentifier:
"registerTableViewCell", for: indexPath as IndexPath) as! registerTableViewCell

                 cell.textName.tag = indexPath.row + 100
                 cell.textName.placeholder = textFieldPlaceHolder[indexPath.row]
                 cell.textName.delegate = self
                 cell.textName.keyboardType = .phonePad
                 cell.textName.isSecureTextEntry = false
                 cell.textName.text = modelObj.phonenumber
                 cell.errorLbl.text = (indexPath.row == modelObj.errorIndex ?
modelObj.errormessage : "")

                 return cell


            case 3:

            let cell = tableView.dequeueReusableCell(withIdentifier:
"registerTableViewCell", for: indexPath as IndexPath) as! registerTableViewCell

                cell.textName.tag = indexPath.row + 100
                cell.textName.placeholder = textFieldPlaceHolder[indexPath.row]
                cell.textName.delegate = self
                cell.textName.keyboardType = .asciiCapable
                cell.textName.isSecureTextEntry = true

                cell.errorLbl.text = (indexPath.row == modelObj.errorIndex ?
modelObj.errormessage : "")

            return cell

            case 4:
            let cell = tableView.dequeueReusableCell(withIdentifier:
"registerTableViewCell", for: indexPath as IndexPath) as! registerTableViewCell

                cell.textName.tag = indexPath.row + 100
```

```
                cell.textName.placeholder = textFieldPlaceHolder[indexPath.row]
                cell.textName.delegate = self
                cell.textName.keyboardType = .asciiCapable
                  cell.textName.isSecureTextEntry = true

                cell.errorLbl.text = (indexPath.row == modelObj.errorIndex ?
modelObj.errormessage : "")

            return cell


        default :
            break

    }
        return UITableViewCell()


}

}

extension ViewController : UITextFieldDelegate {
func textField(_ textField: UITextField, shouldChangeCharactersIn range:
NSRange, replacementString string: String)-> Bool{
    if let text = textField.text as NSString? {

        let newstring = text.replacingCharacters(in: range, with: string)
        let numofchar = newstring.count

       switch textField.tag - 100{

        case 0:
            modelObj.firstname = newstring
            if numofchar > 16 || (textField.textInputMode?.primaryLanguage == "emoji")
{
                return false
            }
            if range.location == 0 && (string == " ") {
                return false
            }
            else {

                let allOverChar = CharacterSet(charactersIn:
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz")
                let characterSet = CharacterSet(charactersIn: string)
                if numofchar == 0 || numofchar == 1 {
```

```
               let cell : registerTableViewCell = (tableView.cellForRow(at:
IndexPath(row: 0, section: 0)) as? registerTableViewCell)!
               cell.errorLbl.text = "Please enter your Full name."
                }

               else {

               let cell : registerTableViewCell = (tableView.cellForRow(at:
IndexPath(row: 0, section: 0)) as? registerTableViewCell)!
               cell.errorLbl.text = ""
                }
            return allOverChar.isSuperset(of: characterSet)
            }
       case 1:

         modelObj.email = newstring
          _ = isAllFieldForEmail()
          if let cell : registerTableViewCell = tableView.cellForRow(at: IndexPath(row:
1, section: 0)) as? registerTableViewCell {
              if(modelObj.errorIndex == 1)
              {
                  cell.errorLbl.text = modelObj.errormessage
              }
                      else if (numofchar < 8)
                  {
                     cell.errorLbl.text = "*Please enter valid email address."
                  }
                  else {
                  cell.errorLbl.text = ""
              }
          }
          return true

        case 2:
           modelObj.phonenumber = newstring
            _ = isAllFieldForPhone()
            if let cell : registerTableViewCell = tableView.cellForRow(at:
IndexPath(row: 2, section: 0)) as? registerTableViewCell {

                if(modelObj.errorIndex == 2){

                    cell.errorLbl.text = modelObj.errormessage

                }
                else if (numofchar < 8) {
```

```
                cell.errorLbl.text = "*Please enter correct phone number."
            }
            else if (numofchar == 15){

                return false
            }
           else {

                cell.errorLbl.text = ""
            }
        }
        return true


        case 3:
                modelObj.password = newstring

                                if let cell : registerTableViewCell = tableView.cellForRow(at:
IndexPath(row: 3, section: 0)) as? registerTableViewCell

                                {
                                    if(modelObj.errorIndex == 3){
                                        cell.errorLbl.text = modelObj.errormessage

                                    } else if (numofchar < 6)
                                    {
                                       cell.errorLbl.text = "*Please enter your password more
than 6-digits."
                                   }

                                    else if (numofchar == 12){

                                        return false
                                    }
                                   else {

                                        cell.errorLbl.text = ""
                                    }
                                }
                                return true

        case 4:
                modelObj.confirmpassword = newstring

                                if let cell : registerTableViewCell = tableView.cellForRow(at:
IndexPath(row: 4, section: 0)) as? registerTableViewCell
```

```
                                   {
                                       if(modelObj.errorIndex == 4){
                                           cell.errorLbl.text = modelObj.errormessage

                                       } else if (modelObj.password !=
modelObj.confirmpassword)
                                       {
                                          cell.errorLbl.text = "*Confirm password does not match
with password."
                                      }

                                       else if (numofchar == 12){

                                           return false
                                       }
                                      else {

                                           cell.errorLbl.text = ""
                                       }
                                   }
                                   return true




           default:
                       return true
           }


           }


       return true

}


   func textFieldShouldReturn(_ textField: UITextField) -> Bool {
              if textField.returnKeyType == .next {
                  let nexttextfield = self.view.viewWithTag(textField.tag + 1) as!
UITextField
                  nexttextfield.becomeFirstResponder()

              }
              else{
                  textField.resignFirstResponder()
```

```
            }
            return true
        }
    }

extension ViewController {

    func isAllFieldForEmail() -> Bool {
        var isVerified = false

        if modelObj.email.count == 0 {

            modelObj.errorIndex = 1
            modelObj.errormessage = "*Please enter your email address."
        }
        else {
            isVerified = true
            modelObj.errorIndex = -1
            modelObj.errormessage = ""
            let isValid = isValidEmail(testStr: modelObj.email)
            if isValid {

            } else {
                isVerified = false

                modelObj.errorIndex = 1

                modelObj.errormessage = "*Please enter your correct email address."
            }
        }

       // self.TableView.reloadData()
        return isVerified
    }
    func isValidEmail(testStr:String) -> Bool {

        let emailRegEx = "^(((([a-zA-Z]|\\d|[!#\\$%&'\\*\\+\\-\\/=\\?\\^_`{\\|}~]|[\\x{00A0}-
\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-\\x{FFEF}])+(\\.([a-zA-Z]|\\d|[!#\\$%&'\\*\\+\\-
\\/=\\?\\^_`{\\|}~]|[\\x{00A0}-\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-
\\x{FFEF}])+)*)|((\\x22)((((\\x20|\\x09)*(\\x0d\\x0a))?(\\x20|\\x09)+)?(([\\x01-
\\x08\\x0b\\x0c\\x0e-\\x1f\\x7f]|\\x21|[\\x23-\\x5b]|[\\x5d-\\x7e]|[\\x{00A0}-
\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-\\x{FFEF}])|(\\([\\x01-\\x09\\x0b\\x0c\\x0d-
\\x7f]|[\\x{00A0}-\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-
\\x{FFEF}]))))*(((\\x20|\\x09)*(\\x0d\\x0a))?(\\x20|\\x09)+)?(\\x22)))@((([a-zA-
Z]|\\d|[\\x{00A0}-\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-\\x{FFEF}])|(([a-zA-
Z]|\\d|[\\x{00A0}-\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-\\x{FFEF}])([a-zA-Z]|\\d|-
|\\.|_|~|[\\x{00A0}-\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-\\x{FFEF}])*([a-zA-
Z]|\\d|[\\x{00A0}-\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-\\x{FFEF}])))\\.)+(([a-zA-
Z]|[\\x{00A0}-\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-\\x{FFEF}])|(([a-zA-
Z]|[\\x{00A0}-\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-\\x{FFEF}])([a-zA-Z]|\\d|-
|_|~|[\\x{00A0}-\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-\\x{FFEF}])*([a-zA-
Z]|[\\x{00A0}-\\x{D7FF}\\x{F900}-\\x{FDCF}\\x{FDF0}-\\x{FFEF}])))\\.?$"
        let email = NSPredicate(format:"SELF MATCHES %@", emailRegEx)
        let result = email.evaluate(with: testStr)
        return result
    }

    func isAllFieldForPhone() -> Bool {
        var isVerified = false


        if modelObj.phonenumber.count == 0 {
            modelObj.errorIndex = 2
            modelObj.errormessage = "*Please enter phone number."
        }
        else if modelObj.phonenumber.count < 8 {
            modelObj.errorIndex = 2
            modelObj.errormessage = "*Please enter your correct phone number."
        }
        else {
            isVerified = true
            modelObj.errorIndex = -1
            modelObj.errormessage = ""
            let isValid = isContainsAllZeros(testStr: modelObj.phonenumber)
            if !isValid {
                // correct mail
            } else {
                isVerified = false
                modelObj.errorIndex = 2
                modelObj.errormessage = "*Please enter your correct mobile number."
            }

        }
         // self.TableView.reloadData()
        return isVerified
    }
    func isContainsAllZeros(testStr: String) -> Bool {


        let mobileNoRegEx = "^0{2,15}$"

        let mobileNoTest = NSPredicate(format: "SELF MATCHES %@",
mobileNoRegEx)
```

```
        return mobileNoTest.evaluate(with: testStr)

    }
}

extension ViewController {

    func registration() -> Bool {

        var register = false


        if (modelObj.firstname.isEmpty)
        {
            modelObj.errorIndex = 0
            modelObj.errormessage = "*Please enter your first name."
            self.tableView.scrollToRow(at : IndexPath(row: 0, section: 0), at: .top,
animated: true)

        }



        else if (modelObj.email.isEmpty){

            modelObj.errorIndex = 1
            modelObj.errormessage = "*Please enter your e-mail address."
            self.tableView.scrollToRow(at : IndexPath(row: 1, section: 0), at: .top,
animated: true)

        }   else if (!isAllFieldForEmail()){

            modelObj.errorIndex = 1
            modelObj.errormessage = "*Please enter your correct e-mail address."
            self.tableView.scrollToRow(at : IndexPath(row: 1, section: 0), at: .top,
animated: true)

            }
        else if (modelObj.phonenumber.isEmpty){

            modelObj.errorIndex = 2
            modelObj.errormessage = "*Please enter your phone number."
            self.tableView.scrollToRow(at : IndexPath(row: 2, section: 0), at: .top,
animated: true)

        }
        else if (modelObj.password.isEmpty){

            modelObj.errorIndex = 3
            modelObj.errormessage = "*Please enter password."
            self.tableView.scrollToRow(at : IndexPath(row: 3, section: 0), at: .top,
animated: true)
        }

        else if (modelObj.confirmpassword.isEmpty) {

            modelObj.errorIndex = 4
            modelObj.errormessage = "*Please enter confirm password ."
            self.tableView.scrollToRow(at : IndexPath(row: 4, section: 0), at: .top,
animated: true)

        }

        else if (modelObj.confirmpassword != modelObj.password ) {

            modelObj.errorIndex = 4
            modelObj.errormessage = "*Confirm password should be same with the
password field."
            self.tableView.scrollToRow(at : IndexPath(row: 4, section: 0), at: .top,
animated: true)

            }

        else{
            register = true

        }
          self.tableView.reloadData()
        return register
    }

}

class walletViewController: UIViewController {

    @IBOutlet weak var tableView: UITableView!
    var pending = ["Pending", "Success", "Pending", "Success", "Pending",
"Pending", "Success", "Success", "Pending", "Success", "Pending", "Pending",
"Pending", "Success"]
    override func viewDidLoad() {
         super.viewDidLoad()
    }




}

extension walletViewController: UITableViewDelegate, UITableViewDataSource
{
    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int)
-> Int {
        return 14
        return pending.count

    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -
> UITableViewCell {

        let cell = tableView.dequeueReusableCell(withIdentifier:
"walletTableViewCell", for: indexPath) as! walletTableViewCell

        cell.pendingLbl.text = pending[indexPath.row]
        return cell
    }


}


var sportsData = [sportsApp(sectionType: "Bolloywood Movies", imageGallary :
["araya","araya","araya","araya"]),
                    sportsApp(sectionType: "Hindi Webseries", imageGallary:
["araya","araya","araya ","araya"]),

  ]
 var actorNames = ["sxsxsx","xsxs","xsqx","sjsjcxsx"]
var actresnames = ["dssdd","sdsdsd","sdsde","dsdsdde"]


class ViewController: UIViewController {



    @IBOutlet weak var tableView: UITableView!
    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
    }


}
extension ViewController: UITableViewDelegate, UITableViewDataSource {
    func tableView(_ tableView: UITableView, heightForRowAt indexPath:
IndexPath) -> CGFloat {
        if indexPath.row == 0{
            return 200
        }
        else{
            return UITableView.automaticDimension
        }
    }
    func numberOfSections(in tableView: UITableView) -> Int {
        return sportsData.count
    }
    func tableView(_ tableView: UITableView, titleForHeaderInSection section: Int) -
> String? {
        return sportsData[section].sectionType
    }

    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int)
-> Int {
        return actorNames.count
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -
> UITableViewCell {
        if indexPath.row == 0{
            let cell = tableView.dequeueReusableCell(withIdentifier:
"imgTableViewCell", for: indexPath) as! imgTableViewCell

            cell.mycollectionView.tag = indexPath.row
            return cell
        }
        else{
            if sportsData.count == 0{
            let cell = tableView.dequeueReusableCell(withIdentifier: "lblTableViewCell",
for: indexPath) as!lblTableViewCell
            cell.showLbl.text = actorNames[indexPath.row-1]
            return cell
        }
            else{
                let cell = tableView.dequeueReusableCell(withIdentifier:
"lblTableViewCell", for: indexPath) as! lblTableViewCell
                cell.showLbl.text = actresnames[indexPath.row-1]
                return cell
            }
            }

        //return UITableViewCell()

    }

    func tableView(_ tableView: UITableView, willDisplayHeaderView view: UIView,
forSection section: Int) {
        view.tintColor = .yellow
    }




}

class imgTableViewCell: UITableViewCell {

    @IBOutlet weak var mycollectionView: UICollectionView!
    override func awakeFromNib() {
        super.awakeFromNib()
        mycollectionView.delegate = self
        mycollectionView.dataSource = self
    }

    override func setSelected(_ selected: Bool, animated: Bool) {
        super.setSelected(selected, animated: animated)

        // Configure the view for the selected state
    }

}
extension
imgTableViewCell:UICollectionViewDelegate,UICollectionViewDataSource{
    func collectionView(_ collectionView: UICollectionView,
numberOfItemsInSection section: Int) -> Int {
        return sportsData[mycollectionView.tag].imageGallary.count
    }

    func collectionView(_ collectionView: UICollectionView, cellForItemAt
indexPath: IndexPath) -> UICollectionViewCell {
       let cell = mycollectionView.dequeueReusableCell(withReuseIdentifier:
"myCollectionViewCell", for: indexPath) as! myCollectionViewCell
        cell.myImage.image = UIImage(named:
sportsData[mycollectionView.tag].imageGallary[indexPath.row])
        return cell
    }

class signUpViewController: UIViewController {




    @IBOutlet weak var signupBttn: UIButton!

    var textFieldPlaceHolder = ["Full Name","E-mail","Mobile","Password"]

    var modelObj = modelClass()

    @IBOutlet weak var TableView: UITableView!
    override func viewDidLoad() {
        super.viewDidLoad()

     signupBttn.layer.cornerRadius = 4



    func showAlert(usermessage: String)
    {
        let alert = UIAlertController(title: title, message: usermessage, preferredStyle:
.alert)
        let action = UIAlertAction(title: "Ok", style: .cancel, handler: nil)
        alert.addAction(action)
        DispatchQueue.main.async {
            self.present(alert, animated: true, completion: nil)
        }


    }

    }
    @IBAction func signUpTapped(_ sender: UIButton) {

       if self.universityUser() {

        signUpAPI()
       }
    }

}
extension signUpViewController : UITableViewDelegate,UITableViewDataSource
{
    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int)
-> Int {
        return textFieldPlaceHolder.count
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -
> UITableViewCell {

        switch indexPath.row{

            case 0:

            let cell = tableView.dequeueReusableCell(withIdentifier:
"signUpTableViewCell", for: indexPath as IndexPath) as! signUpTableViewCell
                cell.textField.tag = 100
                cell.textField.tag = indexPath.row + 100
                cell.textField.placeholder = textFieldPlaceHolder[indexPath.row]
                cell.textField.delegate = self
                cell.textField.autocapitalizationType = .words
                cell.textField.text = modelObj.firstname
                cell.textField.isSecureTextEntry = false
                cell.errorLabel.text = (indexPath.row == modelObj.errorIndex ?
modelObj.errormessage : "")
                cell.textField.attributedPlaceholder = NSAttributedString(string: "Please
Enter FirstName",
                                                                    attributes:
[NSAttributedString.Key.foregroundColor: UIColor.white])





                return cell




            case 1:
```

```
            let cell = tableView.dequeueReusableCell(withIdentifier:
"signUpTableViewCell", for: indexPath as IndexPath) as! signUpTableViewCell

                cell.textField.tag = indexPath.row + 100
                cell.textField.placeholder = textFieldPlaceHolder[indexPath.row]
                cell.textField.delegate = self
                cell.textField.keyboardType = .emailAddress
                cell.textField.isSecureTextEntry = false
                cell.textField.text = modelObj.email
                cell.errorLabel.text = (indexPath.row == modelObj.errorIndex ?
modelObj.errormessage : "")
                cell.textField.attributedPlaceholder = NSAttributedString(string: "Please
Enter your Email",
                                                                     attributes:
[NSAttributedString.Key.foregroundColor: UIColor.white])

            return cell

                  case 2:
                  let cell = tableView.dequeueReusableCell(withIdentifier:
"signUpTableViewCell", for: indexPath as IndexPath) as! signUpTableViewCell

                      cell.textField.tag = indexPath.row + 100
                      cell.textField.placeholder = textFieldPlaceHolder[indexPath.row]
                      cell.textField.delegate = self
                      cell.textField.keyboardType = .phonePad
                      cell.textField.isSecureTextEntry = false
                      cell.textField.text = modelObj.phonenumber
                      cell.errorLabel.text = (indexPath.row == modelObj.errorIndex ?
modelObj.errormessage : "")
                    cell.textField.attributedPlaceholder = NSAttributedString(string:
"Please Enter MobileNumber",
                                                                         attributes:
[NSAttributedString.Key.foregroundColor: UIColor.white])

                  return cell


            case 3:

            let cell = tableView.dequeueReusableCell(withIdentifier:
"signUpTableViewCell", for: indexPath as IndexPath) as! signUpTableViewCell

                cell.textField.tag = indexPath.row + 100
                cell.textField.placeholder = textFieldPlaceHolder[indexPath.row]
                cell.textField.delegate = self
                cell.textField.keyboardType = .asciiCapable
                cell.textField.isSecureTextEntry = true
                cell.textField.text = modelObj.password
                cell.errorLabel.text = (indexPath.row == modelObj.errorIndex ?
modelObj.errormessage : "")
                cell.textField.attributedPlaceholder = NSAttributedString(string: "Please
Enter password",
                                                                   attributes:
[NSAttributedString.Key.foregroundColor: UIColor.white])

            return cell



        default :
            break

    }
        return UITableViewCell()


}

}

 extension signUpViewController : UITextFieldDelegate {
 func textField(_ textField: UITextField, shouldChangeCharactersIn range:
NSRange, replacementString string: String)-> Bool{
     if let text = textField.text as NSString? {

         let newstring = text.replacingCharacters(in: range, with: string)
         let numofchar = newstring.count

        switch textField.tag - 100{

         case 0:
             modelObj.firstname = newstring
             if numofchar > 16 || (textField.textInputMode?.primaryLanguage ==
"emoji") {
                 return false
             }
             if range.location == 0 && (string == " ") {
                 return false
             }
             else {

                 let allOverChar = CharacterSet(charactersIn:
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz")
```

```
                let characterSet = CharacterSet(charactersIn: string)
                if numofchar == 0 || numofchar == 1 {

               let cell : signUpTableViewCell = (TableView.cellForRow(at:
IndexPath(row: 0, section: 0)) as? signUpTableViewCell)!
               cell.errorLabel.text = "Please enter your Full name."
                }

               else {

               let cell : signUpTableViewCell = (TableView.cellForRow(at:
IndexPath(row: 0, section: 0)) as? signUpTableViewCell)!
               cell.errorLabel.text = ""
                }
            return allOverChar.isSuperset(of: characterSet)
            }
        case 1:

         modelObj.email = newstring
          _ = isAllFieldForEmail()
          if let cell : signUpTableViewCell = TableView.cellForRow(at: IndexPath(row:
1, section: 0)) as? signUpTableViewCell {
             if(modelObj.errorIndex == 1){
                cell.errorLabel.text = modelObj.errormessage
             }
                  else if (numofchar < 8)
                {
                   cell.errorLabel.text = "*Please enter valid email address."
                }
                else {
                cell.errorLabel.text = ""
             }
          }
          return true

         case 2:
           modelObj.phonenumber = newstring
            _ = isAllFieldForPhone()
            if let cell : signUpTableViewCell = TableView.cellForRow(at:
IndexPath(row: 2, section: 0)) as? signUpTableViewCell {

                if(modelObj.errorIndex == 2){

                   cell.errorLabel.text = modelObj.errormessage

                }
                else if (numofchar < 8) {
```

```
                cell.errorLabel.text = "*Please enter correct phone number."
            }
            else if (numofchar == 15){

                return false
            }
           else {

                cell.errorLabel.text = ""
            }
        }
        return true


       case 3:
               modelObj.password = newstring

                                 if let cell : signUpTableViewCell = TableView.cellForRow(at:
IndexPath(row: 3, section: 0)) as? signUpTableViewCell

                                 {
                                     if(modelObj.errorIndex == 3){
                                         cell.errorLabel.text = modelObj.errormessage

                                     } else if (numofchar < 6)
                                     {
                                        cell.errorLabel.text = "*Please enter your password
more than 6-digits."
                                    }

                                     else if (numofchar == 12){

                                         return false
                                     }
                                    else {

                                         cell.errorLabel.text = ""
                                     }
                                 }
                                 return true




       default:
                   return true
            }


            }


        return true

    }


    func textFieldShouldReturn(_ textField: UITextField) -> Bool {
            if textField.returnKeyType == .next {
                let nexttextfield = self.view.viewWithTag(textField.tag + 1) as!
UITextField
                nexttextfield.becomeFirstResponder()

            }
            else{
                textField.resignFirstResponder()
            }
            return true
        }
    }

extension signUpViewController {

    func isAllFieldForEmail() -> Bool {
        var isVerified = false

        if modelObj.email.count == 0 {

            modelObj.errorIndex = 1
            modelObj.errormessage = "*Please enter your email address."
        }
        else {
            isVerified = true
            modelObj.errorIndex = -1
            modelObj.errormessage = ""
            let isValid = isValidEmail(testStr: modelObj.email)
            if isValid {

            } else {
                isVerified = false

                modelObj.errorIndex = 1
                modelObj.errormessage = "*Please enter your correct email address."
            }
        }

       // self.TableView.reloadData()
        return isVerified
    }
    func isValidEmail(testStr:String) -> Bool {

        let emailRegEx = "[A-Z0-9a-z._%+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,64}"
        let email = NSPredicate(format:"SELF MATCHES %@", emailRegEx)
        let result = email.evaluate(with: testStr)
        return result
    }

    func isAllFieldForPhone() -> Bool {
        var isVerified = false


        if modelObj.phonenumber.count == 0 {
            modelObj.errorIndex = 2
            modelObj.errormessage = "*Please enter phone number."
        }
        else if modelObj.phonenumber.count < 8 {
            modelObj.errorIndex = 2
            modelObj.errormessage = "*Please enter your correct phone number."
        }
        else {
            isVerified = true
            modelObj.errorIndex = -1
            modelObj.errormessage = ""
            let isValid = isContainsAllZeros(testStr: modelObj.phonenumber)
            if !isValid {
                // correct mail
            } else {
                isVerified = false
                modelObj.errorIndex = 2
                modelObj.errormessage = "*Please enter your correct mobile number."
            }

        }
         // self.TableView.reloadData()
        return isVerified
    }
    func isContainsAllZeros(testStr: String) -> Bool {
        let mobileNoRegEx = "^0{2,15}$"

        let mobileNoTest = NSPredicate(format: "SELF MATCHES %@",
mobileNoRegEx)

        return mobileNoTest.evaluate(with: testStr)

    }
}

extension signUpViewController {

    func universityUser() -> Bool {

        var universityvalidation = false


        if (modelObj.firstname.isEmpty)
        {
            modelObj.errorIndex = 0
            modelObj.errormessage = "*Please enter your first name."
            self.TableView.scrollToRow(at : IndexPath(row: 0, section: 0), at: .top,
animated: true)

        }



        else if (modelObj.email.isEmpty){

            modelObj.errorIndex = 1
            modelObj.errormessage = "*Please enter your e-mail address."
            self.TableView.scrollToRow(at : IndexPath(row: 1, section: 0), at: .top,
animated: true)

        }   else if (!isAllFieldForEmail()){

            modelObj.errorIndex = 1
            modelObj.errormessage = "*Please enter your correct e-mail address."
            self.TableView.scrollToRow(at : IndexPath(row: 1, section: 0), at: .top,
animated: true)

            }
        else if (modelObj.phonenumber.isEmpty){

            modelObj.errorIndex = 2
            modelObj.errormessage = "*Please enter your phone number."
```

```
          self.TableView.scrollToRow(at : IndexPath(row: 2, section: 0), at: .top,
animated: true)

        }



        else if (modelObj.password.isEmpty){

            modelObj.errorIndex = 3
            modelObj.errormessage = "*Please enter password."
            self.TableView.scrollToRow(at : IndexPath(row: 3, section: 0), at: .top,
animated: true)
        }


        else{
            universityvalidation = true

        }
          self.TableView.reloadData()
        return universityvalidation
    }

}


//webservice integration
extension signUpViewController{

    func signUpAPI(){
    var params = [String:Any]()
    params["userName"] = modelObj.firstname
    params["email"] = modelObj.email
    params["name"] = "abc"
        params["countryCode"] = "123"
        params["mobileNumber"] = modelObj.phonenumber
        params["password"] = modelObj.password
        params["dateOfBirth"] = "123"

        Debug.log("Request parametres ==\(params)")
              // self.showBlankIndicator()
               ServiceHelper.request(params, method: .post, apiName:
API_ADD_BANKACCOUNT, hudType: .simple) { (result, error, code) in
                  // self.hideIndicator()
            if result == nil{  AlertController.alert(title: "", message: "Something went
wrong.");return}

                let responseDict = result as! [String:Any]

                let responseCode = responseDict.validatedValue("responseCode",
expected: 0 as AnyObject) as! Int

                let responseMessagefromServer =
responseDict.validatedValue("responseMessage", expected: "" as AnyObject) as!
String


                if responseCode == 200{

                    Debug.log("Response  ==\(responseDict)")



                }else {
                 AlertController.alert(title: "", message: responseMessagefromServer)
                }
            }


}
}
```

TESTING

System testing is a critical process that takes as much 50% of the system development time. The common view of testing held by users is that it is performed to prove that there are no errors in a program.

Testing is too important for success of the system. If all parts of the system are correct, the goal will be achieved successfully. System testing is also called the —put it all together‖ phase where the elements of the system are put tighter to examine its validity and reliability.

Hardware, Software, Manpower and live data are combined in an effort to produce the necessary result.

The test strategy will include four different types of testing as described below.

Logical Testing :-

This is used to test every aspects of each function as soon as it is implemented. In this test results are compared with expected results.

Fundamental Testing :-

Each function should be tested in the HCP to ensure that no functionalities have been missed out.

System Testing :-

When the system is complete, the whole range of test should be carried out again to ensure that no errors have been introduced.

Acceptance Testing :-

The user will be involved and asks to test all the capabilities of the system to ensure that all required functions are present and working in the manner expected. The user involvement gives the final look to system.

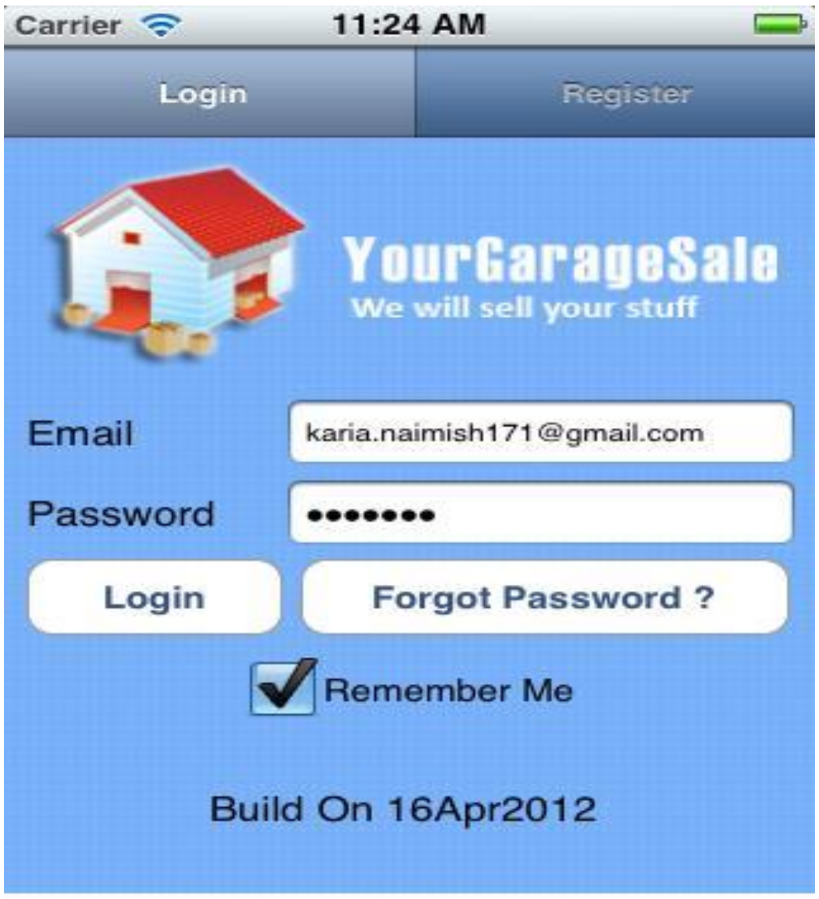
Interface Testing :-

Once the database is gets finalized then interface design is started and after whole design it is tested in context of database. Checks whether all the tables are gets affected and no table is missed out.

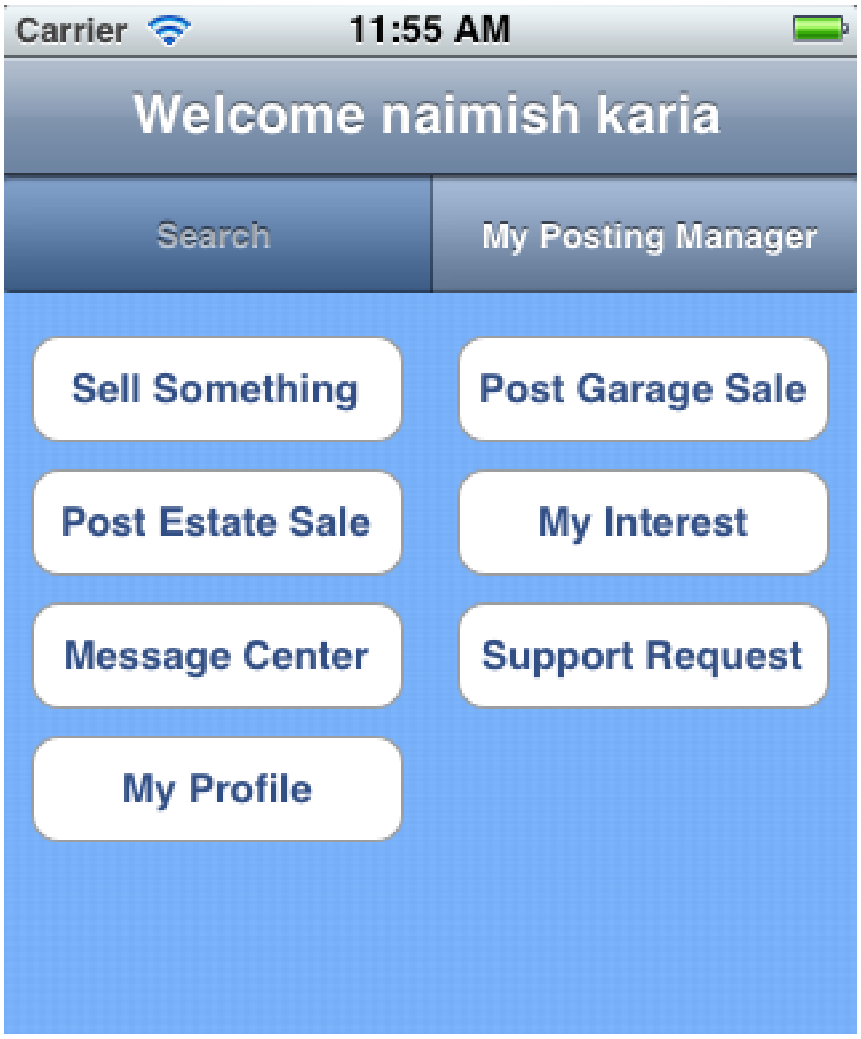
The flow of data is same as the database and sequence is planned property or not? All the controls used in the design are appropriate and looks well or not? Colour Combinations are also finalized after testing with different colours.

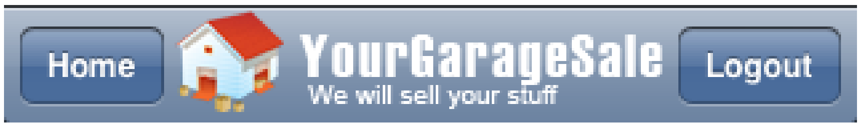
SCREEN LAYOUT

Login screen with registration and forget password functionality and also remember your user name functionality.

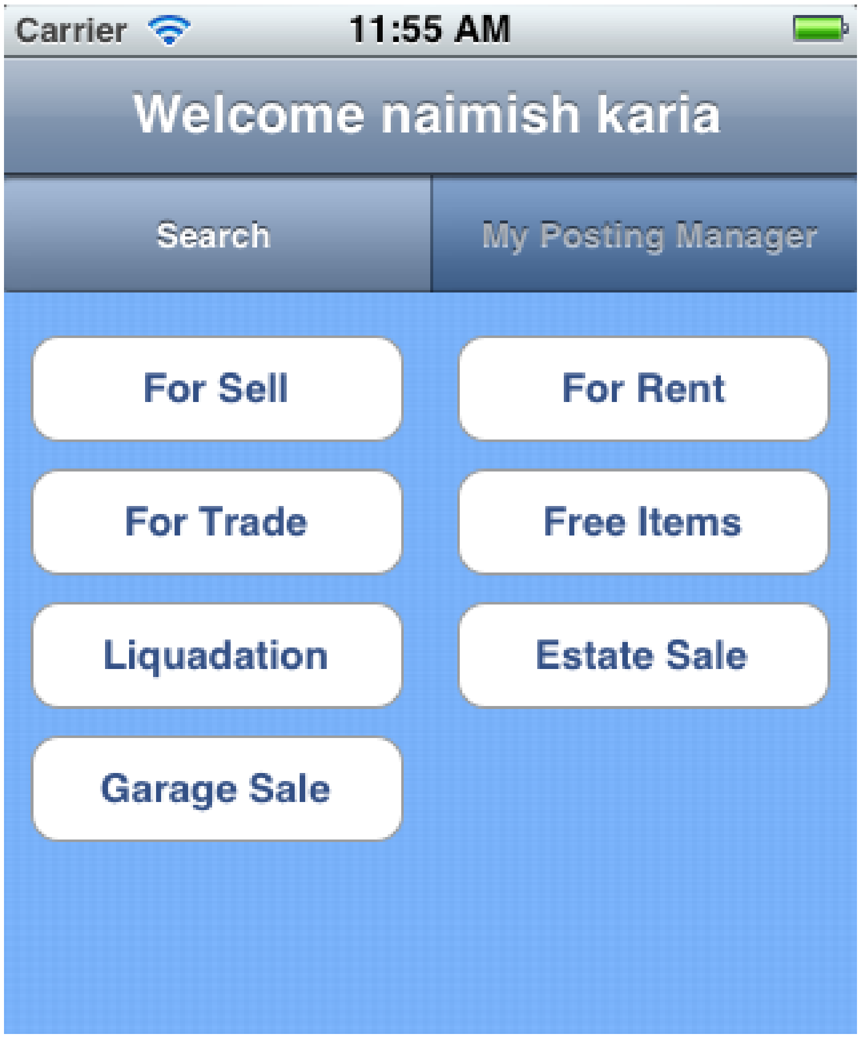


Home Screen for post garage sale :-

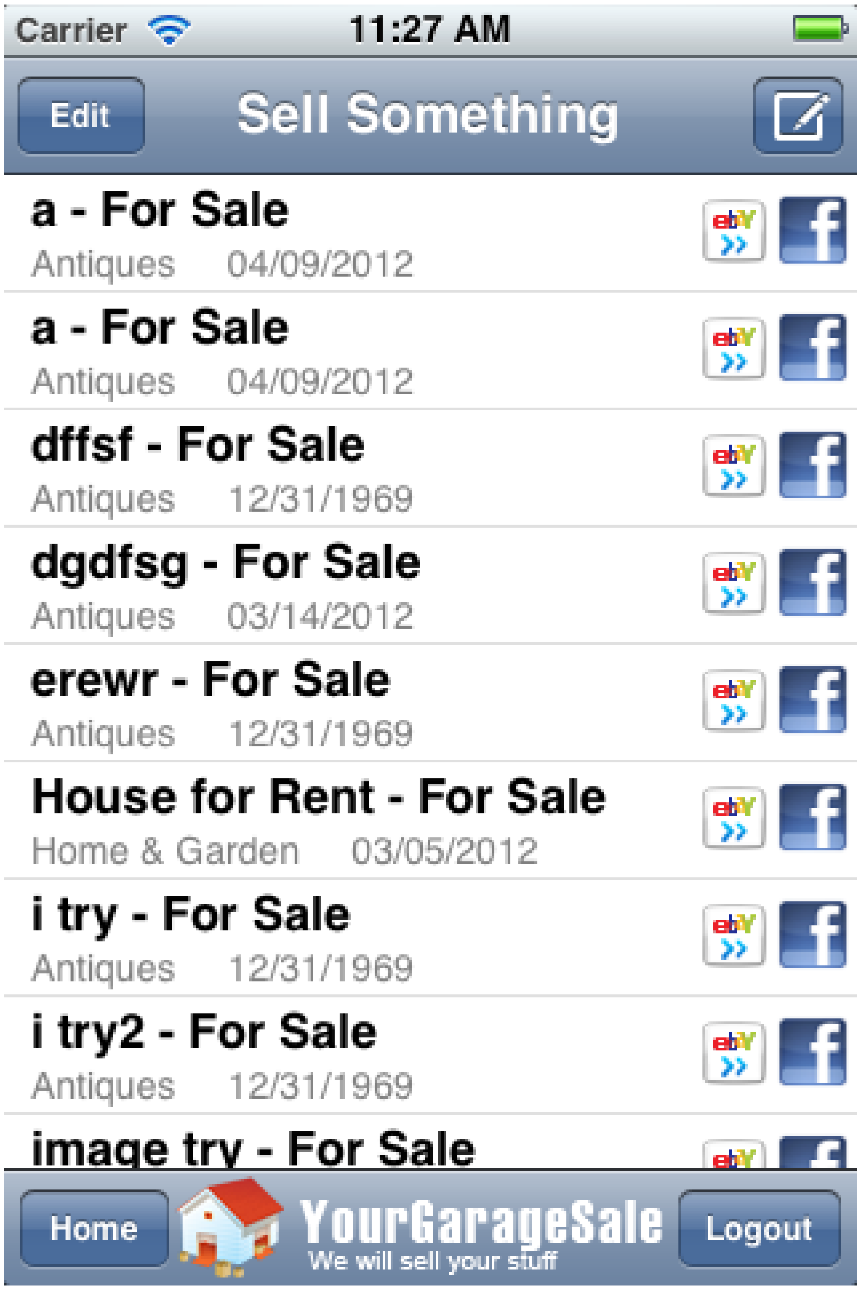




Home screen for search :-



Sell something in post garage sale :-



a - For Sale
Antiques 04/09/2012

a - For Sale
Antiques 04/09/2012

dffsf - For Sale
Antiques 12/31/1969

dgdfsg - For Sale
Antiques 03/14/2012

erewr - For Sale
Antiques 12/31/1969

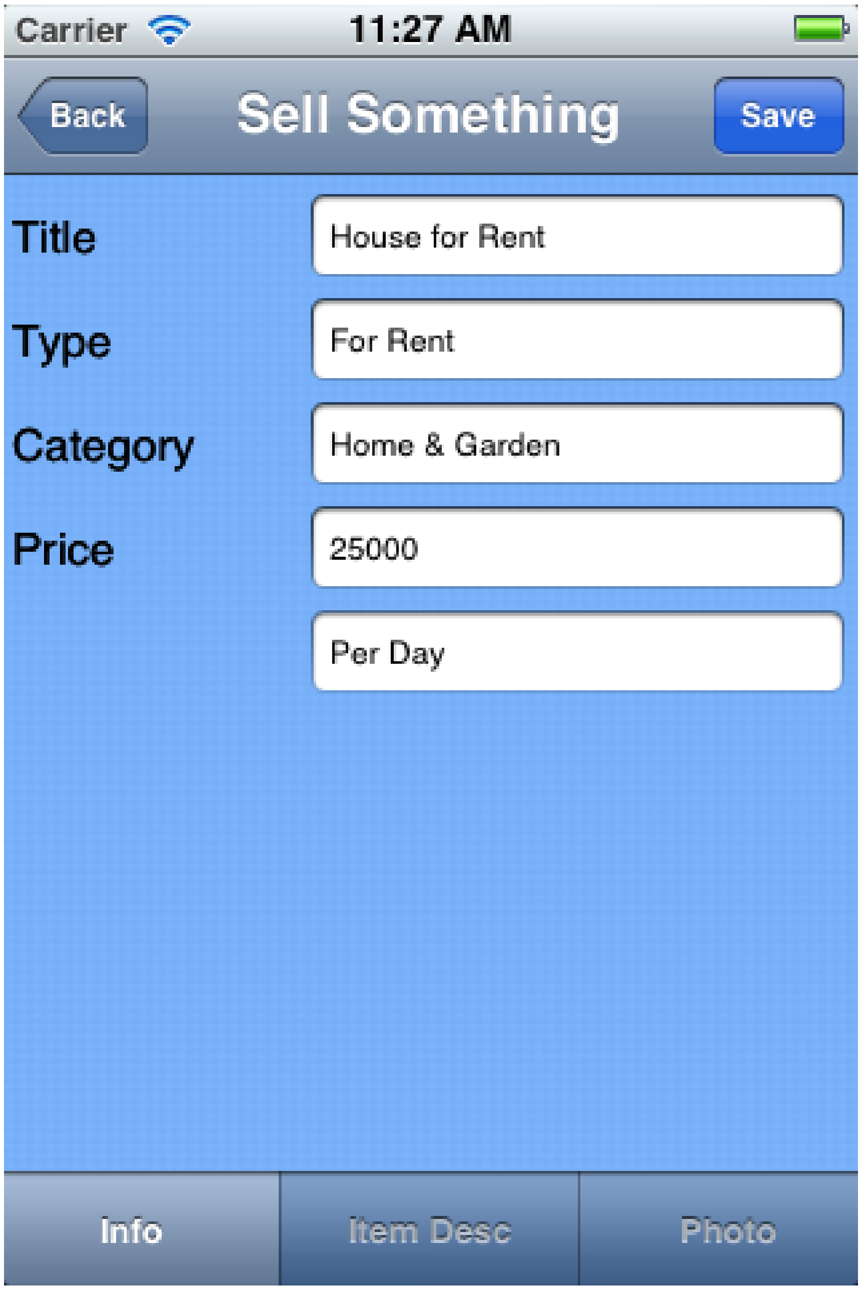
House for Rent - For Sale
Home & Garden 03/05/2012

i try - For Sale
Antiques 12/31/1969

i try2 - For Sale
Antiques 12/31/1969

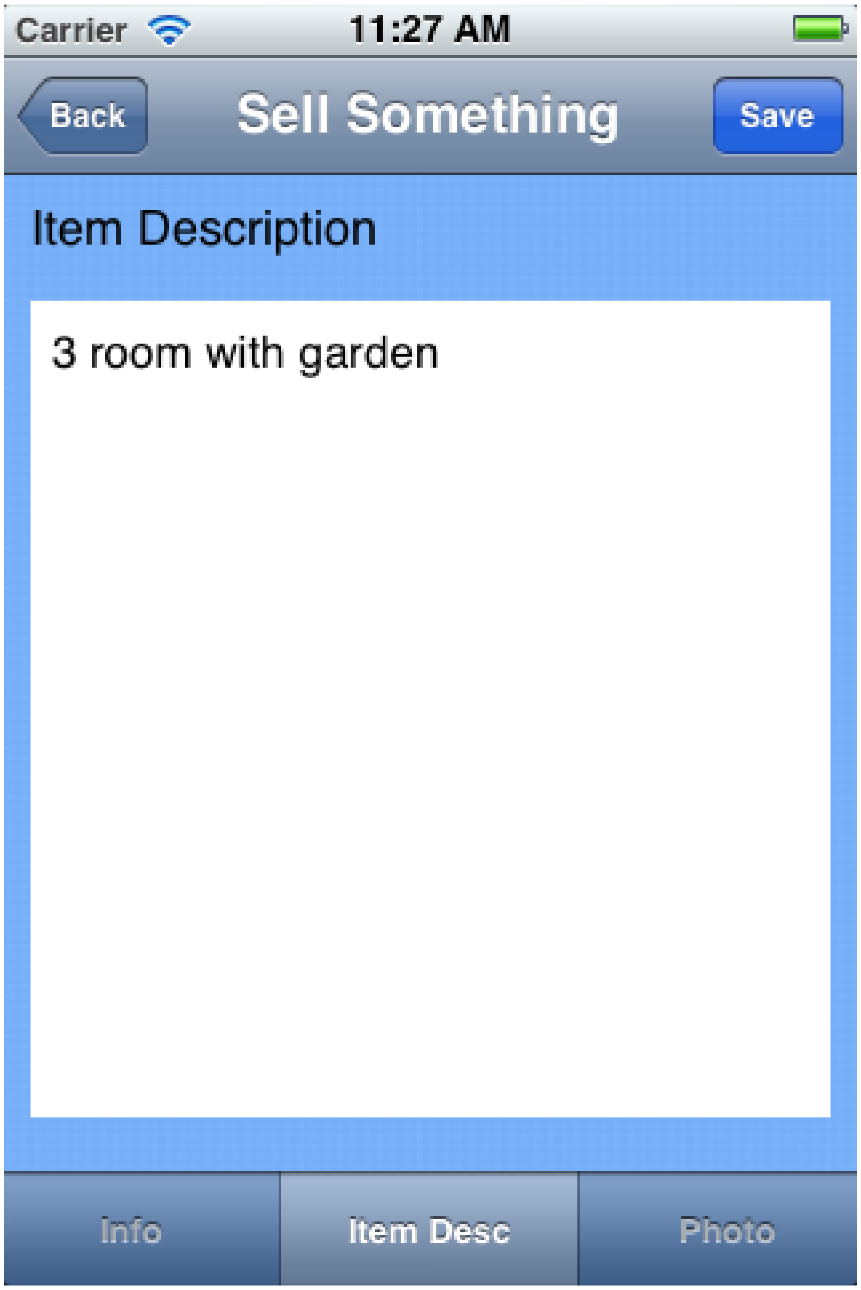
image try - For Sale

Particular item Information screen :-



Title	House for Rent
Type	For Rent
Category	Home & Garden
Price	25000
	Per Day

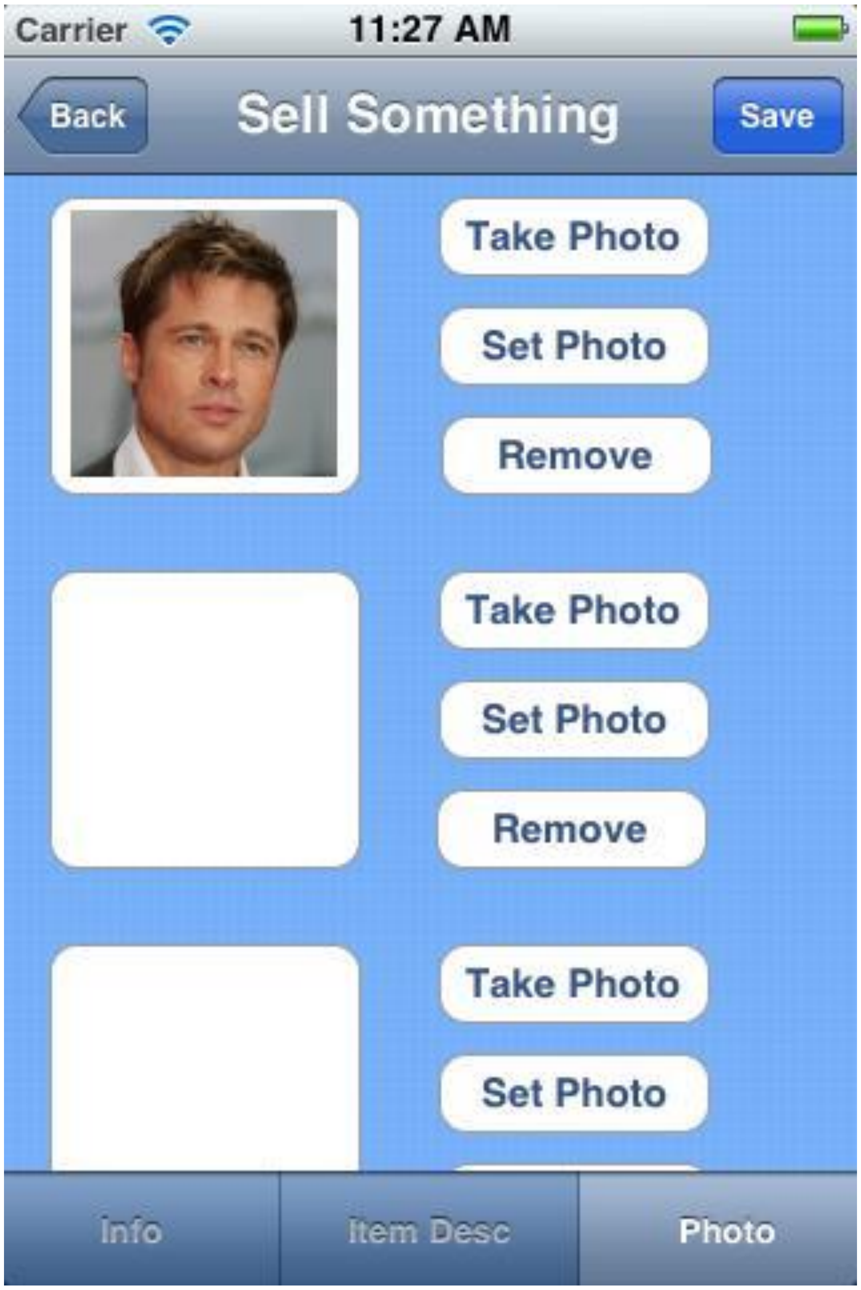
Particular item Description screen :-



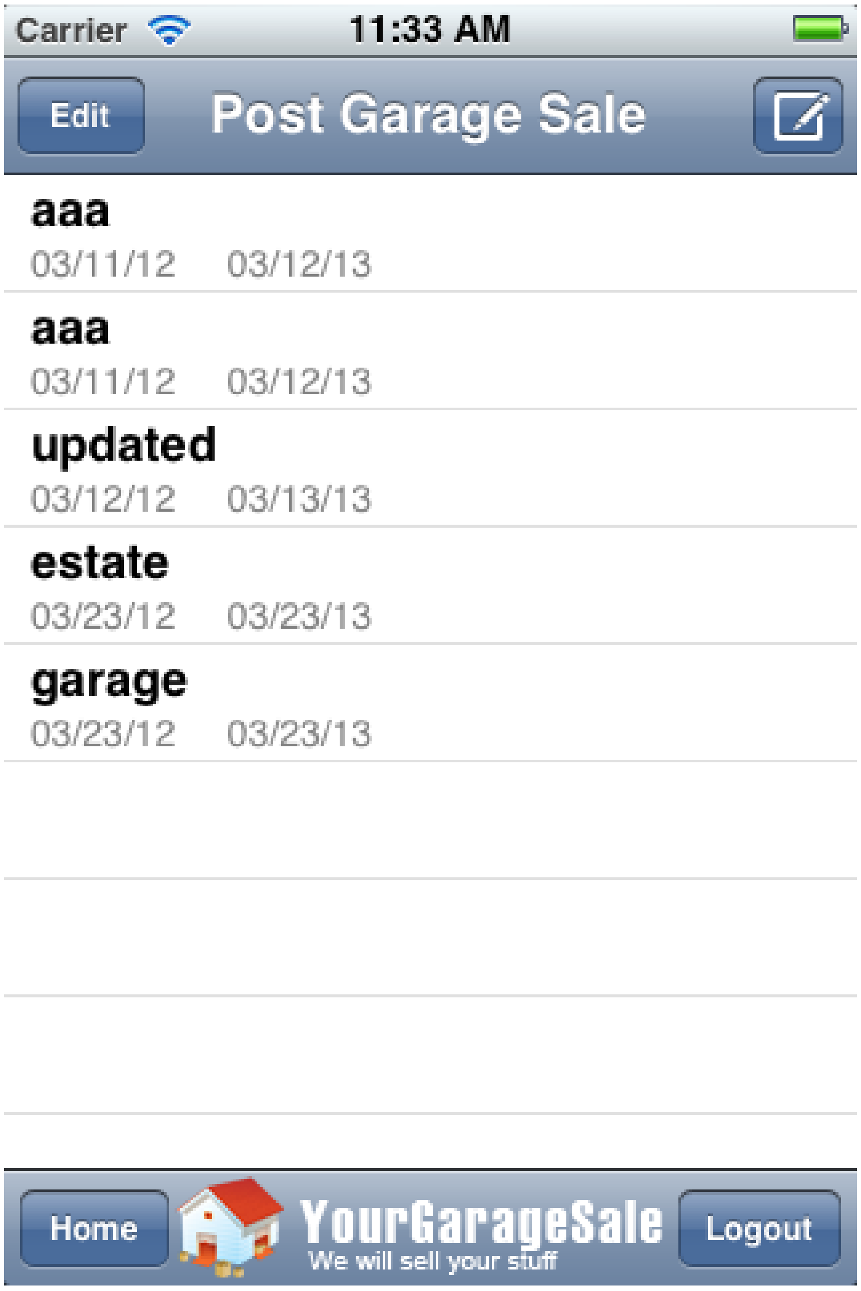
Item Description

3 room with garden

Particular item Available Photos :-

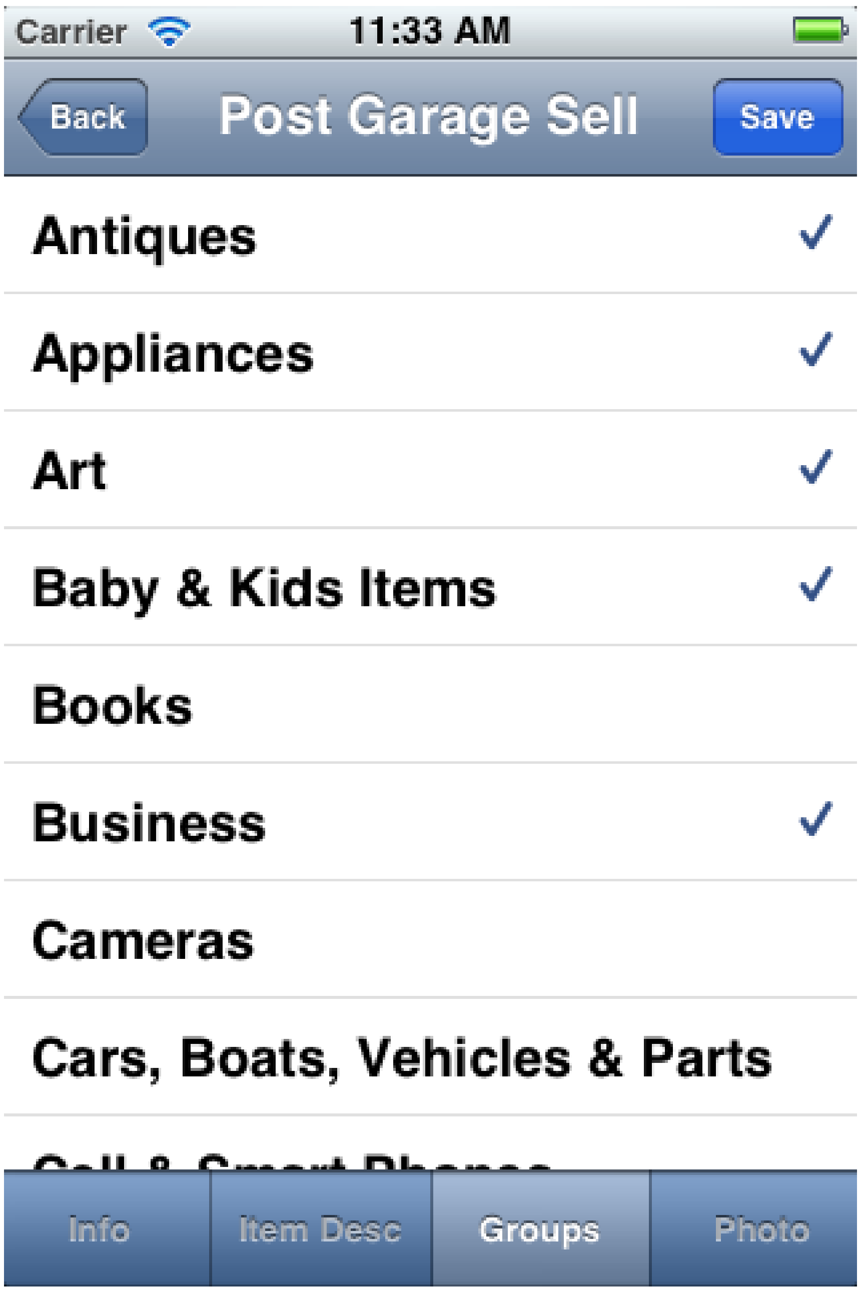


Post Garage sale :-



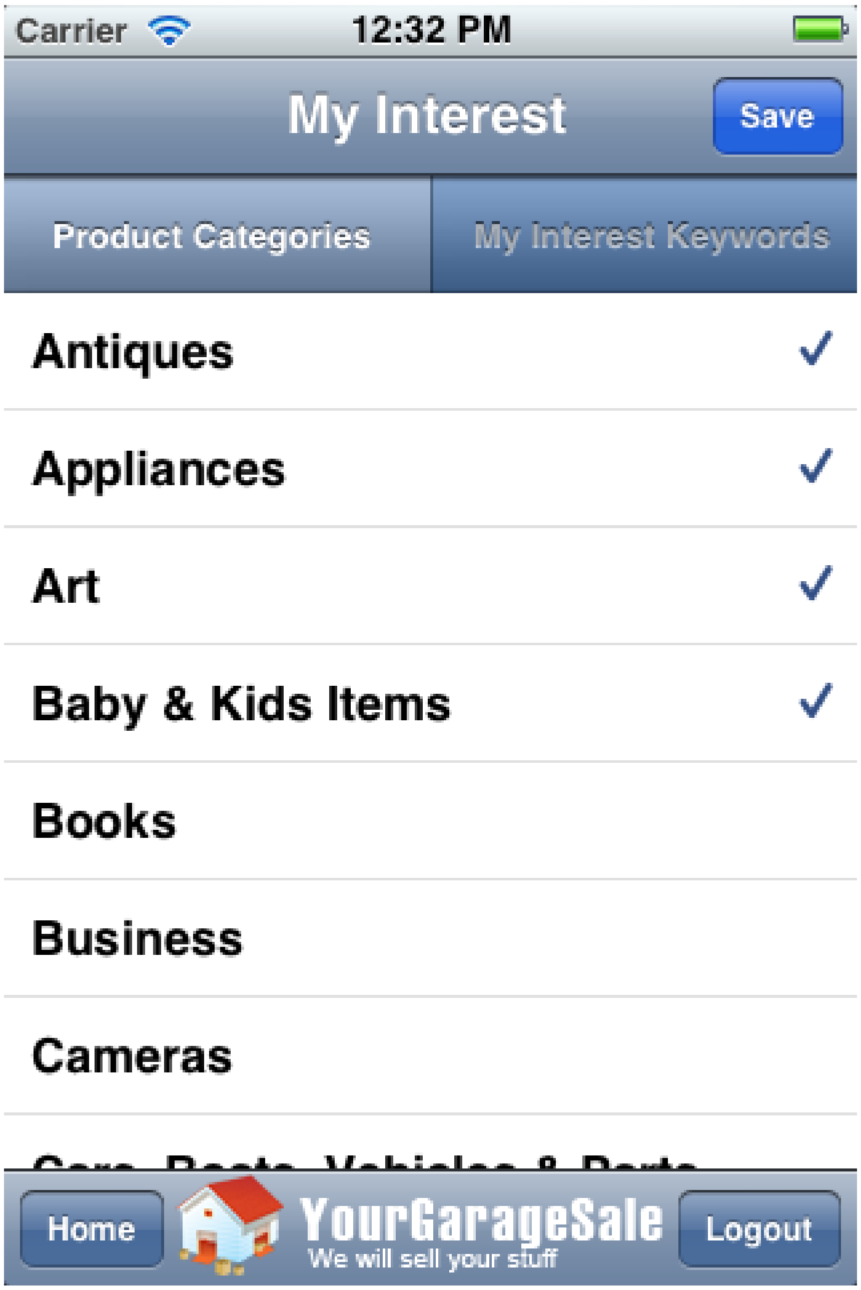
aaa	03/11/12	03/12/13
aaa	03/11/12	03/12/13
updated	03/12/12	03/13/13
estate	03/23/12	03/23/13
garage	03/23/12	03/23/13

Particular item Group This item belongs to all the ticked groups All other three tabs are same as sell something :-



Antiques	✓
Appliances	✓
Art	✓
Baby & Kids Items	✓
Books	
Business	✓
Cameras	
Cars, Boats, Vehicles & Parts	

My Interested categories :-



Antiques ✓

Appliances ✓

Art ✓

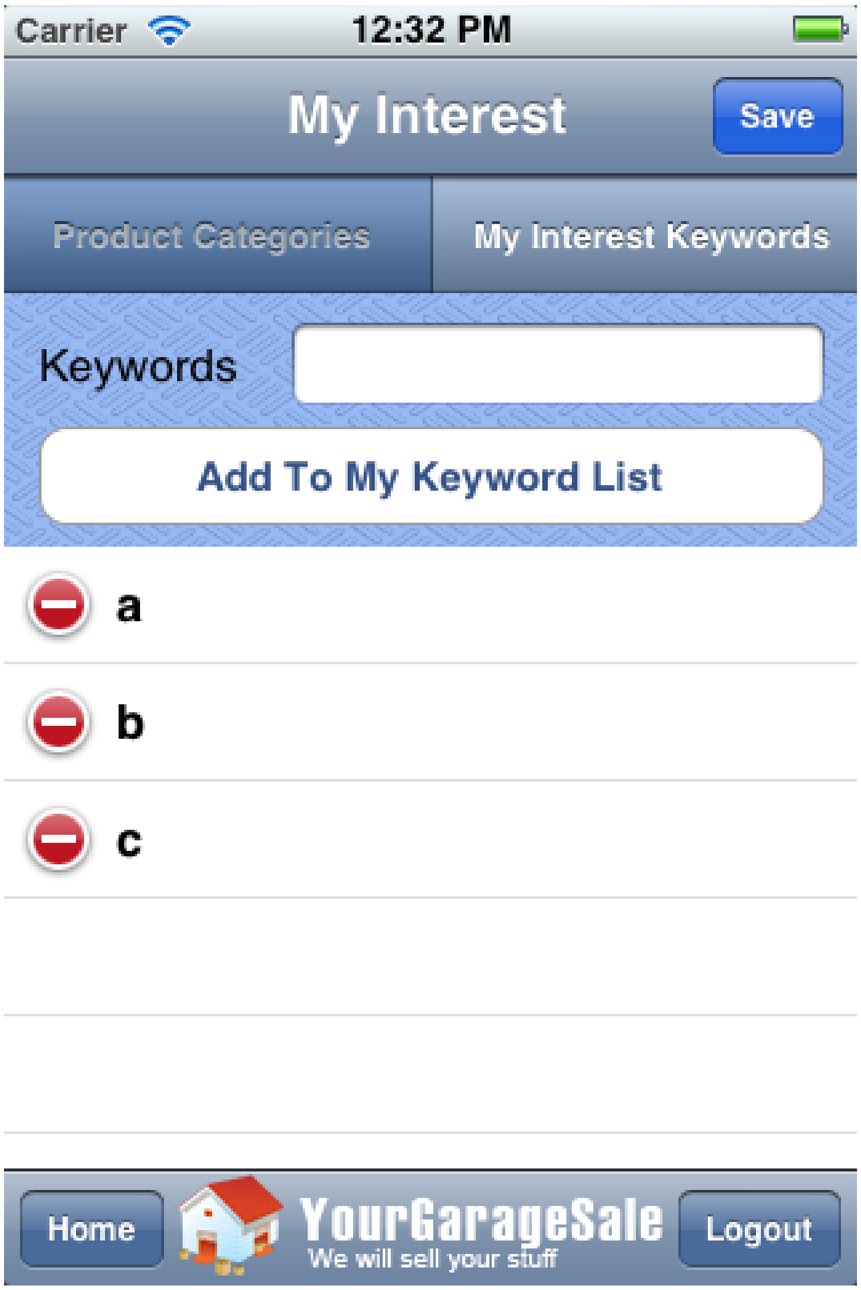
Baby & Kids Items ✓

Books

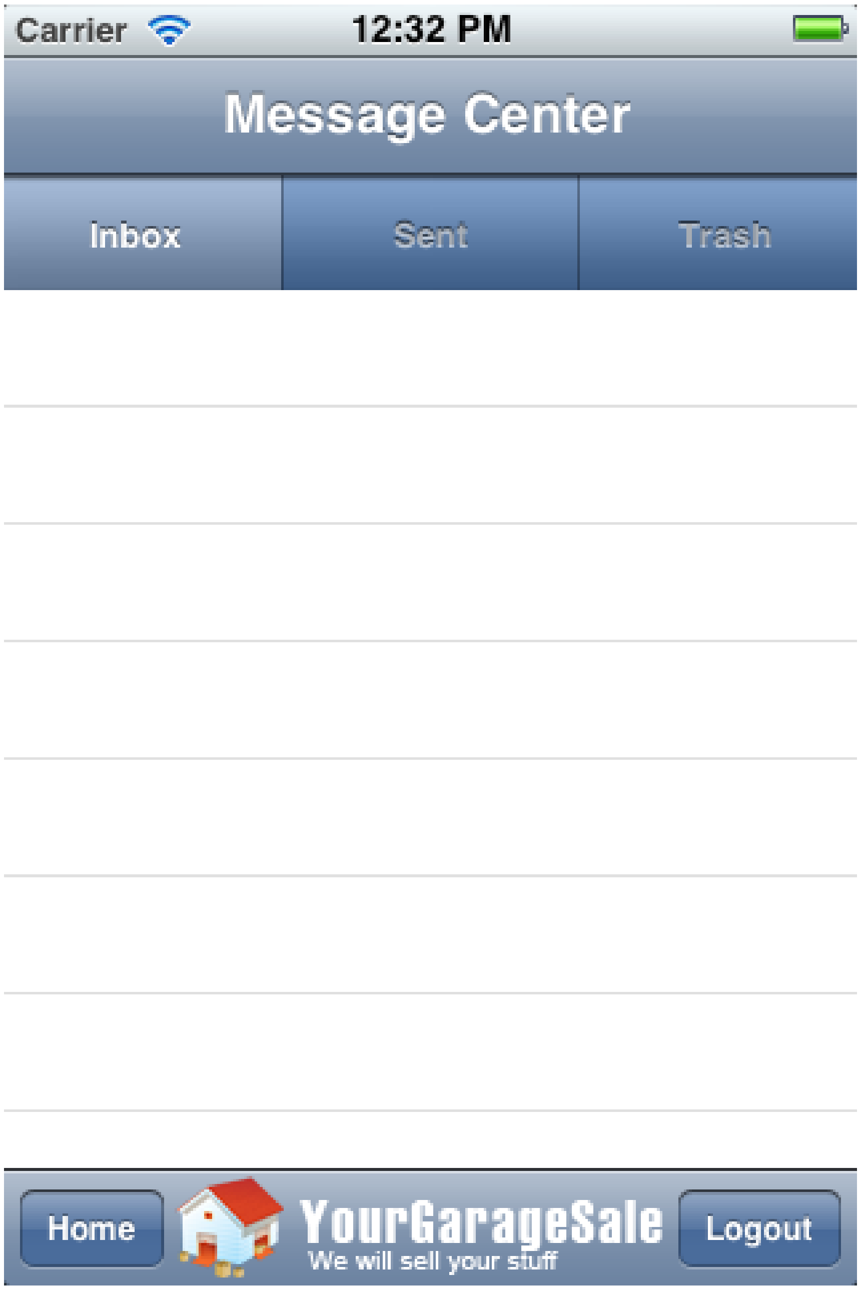
Business

Cameras

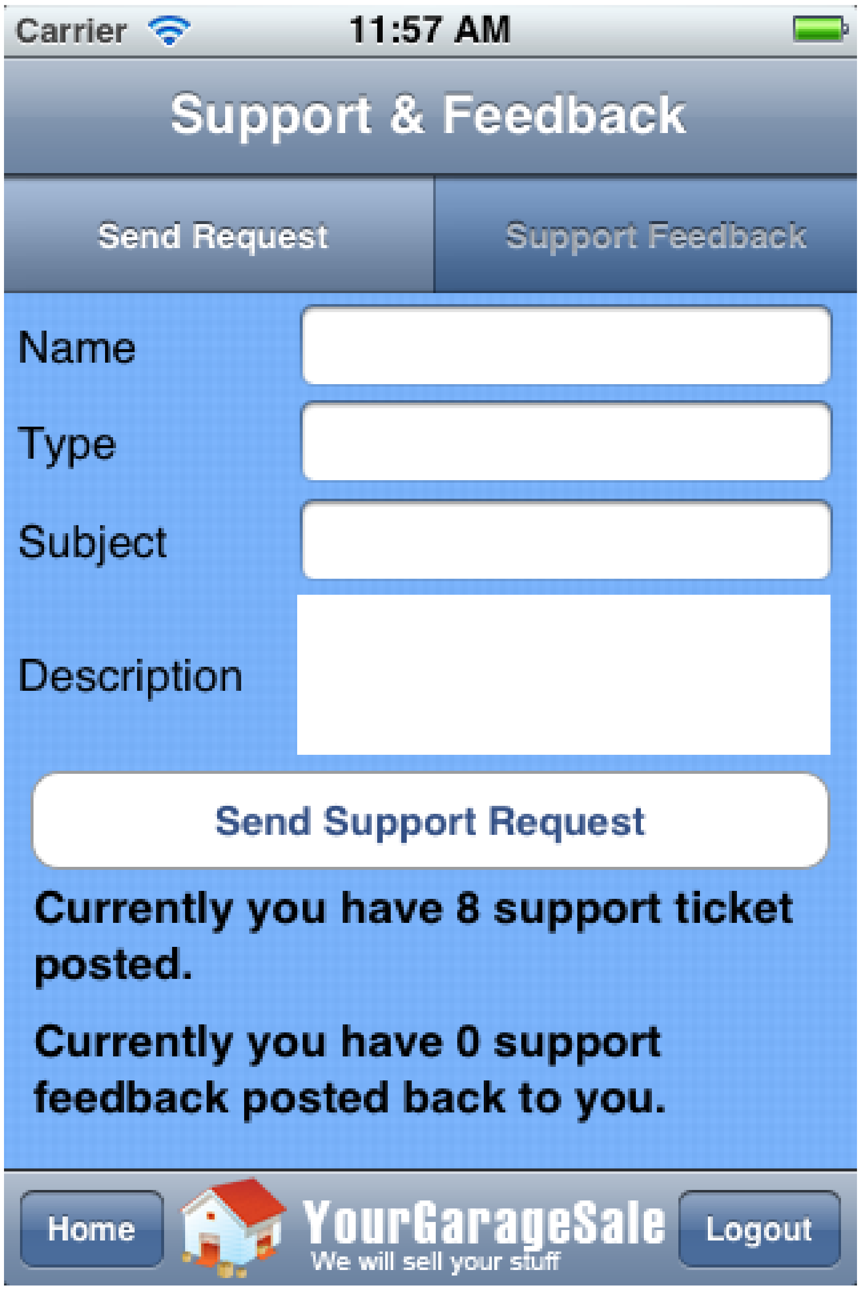
My interested Words :-



My Message inbox Sent item and trash are same in this UI Table view :-



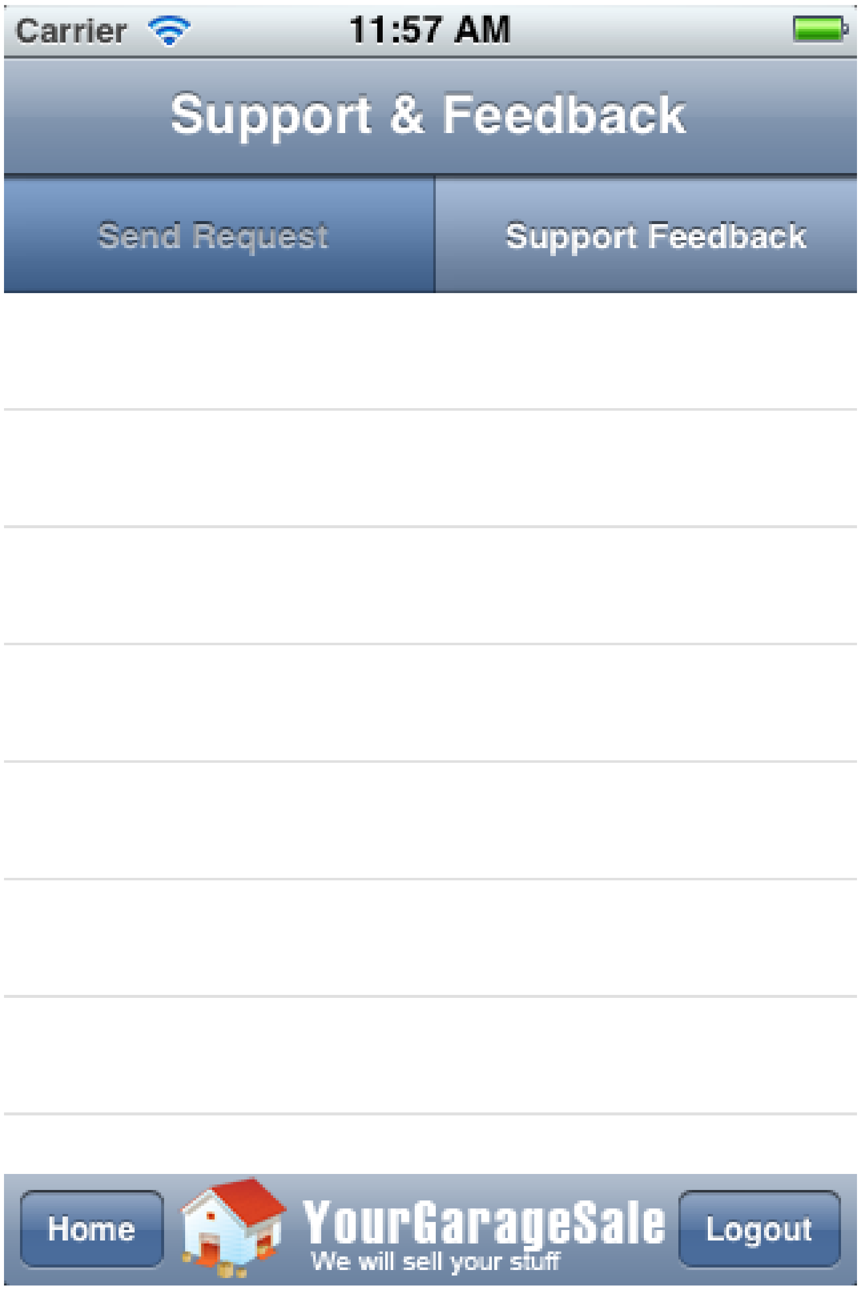
List of Support Request and send new Request :-



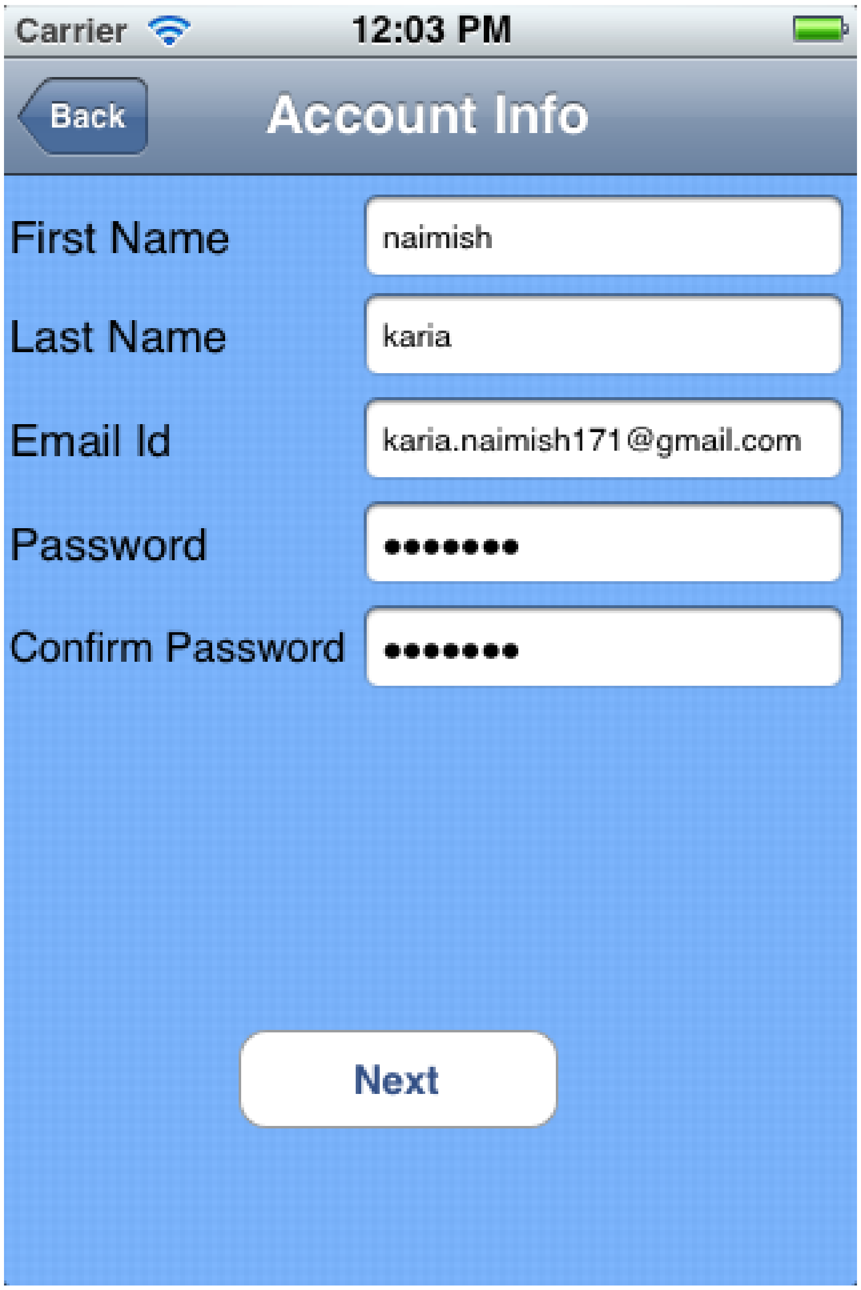
Currently you have 8 support ticket posted.

Currently you have 0 support feedback posted back to you.

List of Feedback :-



User Profile Basic Detail :-



First Name naimish

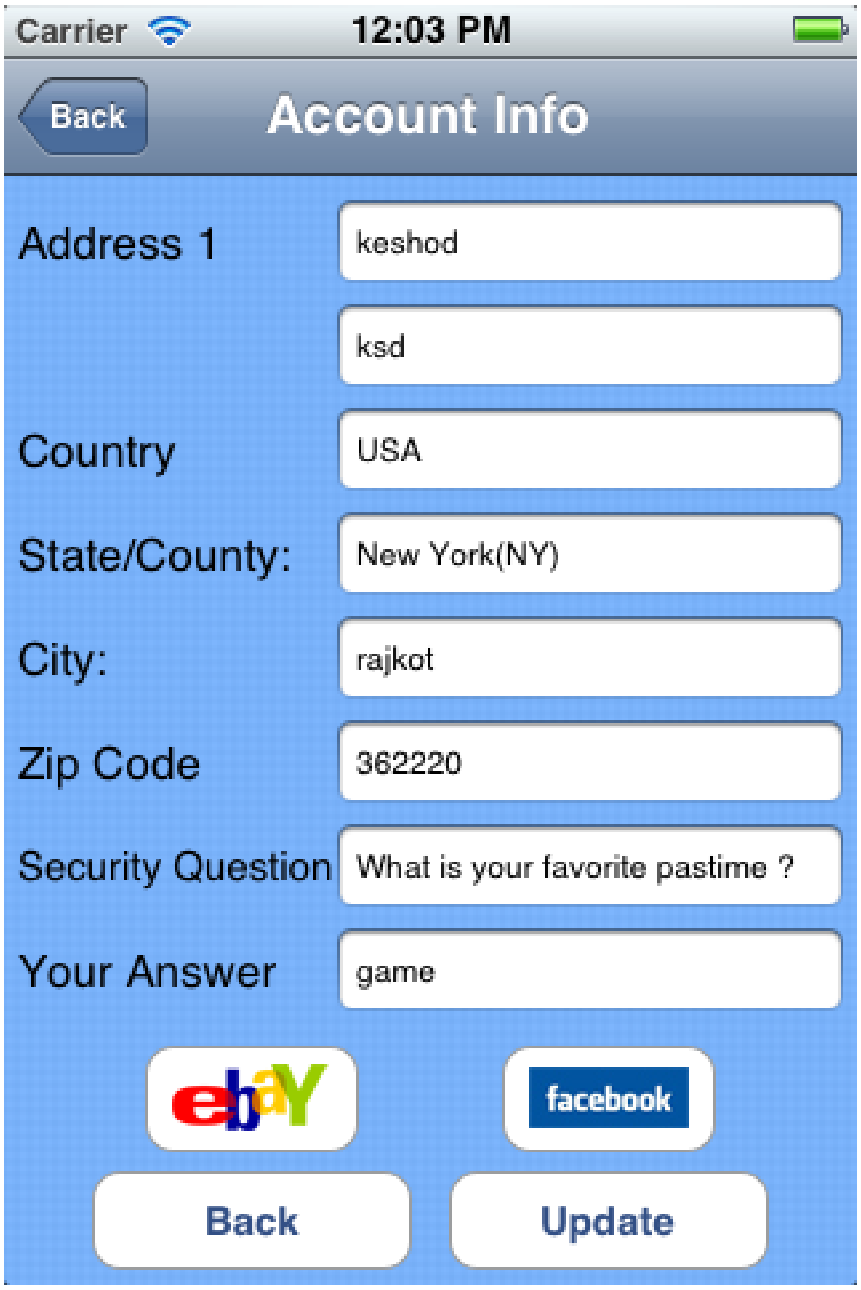
Last Name karia

Email Id karia.naimish171@gmail.com

Password ●●●●●●●

Confirm Password ●●●●●●●

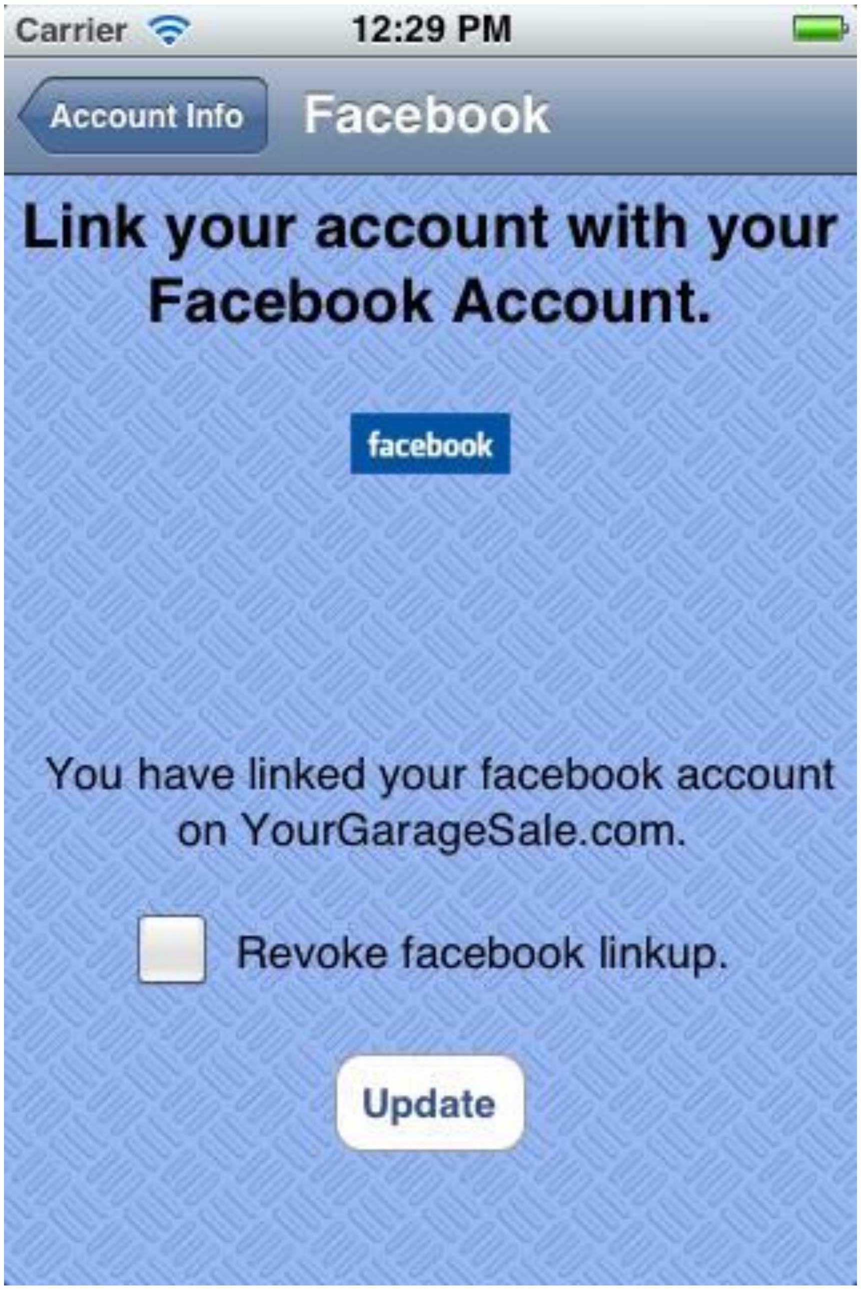
User Profile Full Detail :-



Account Info

Address 1	keshod
	ksd
Country	USA
State/County:	New York(NY)
City:	rajkot
Zip Code	362220
Security Question	What is your favorite pastime ?
Your Answer	game

Link Your Account with Facebook :-

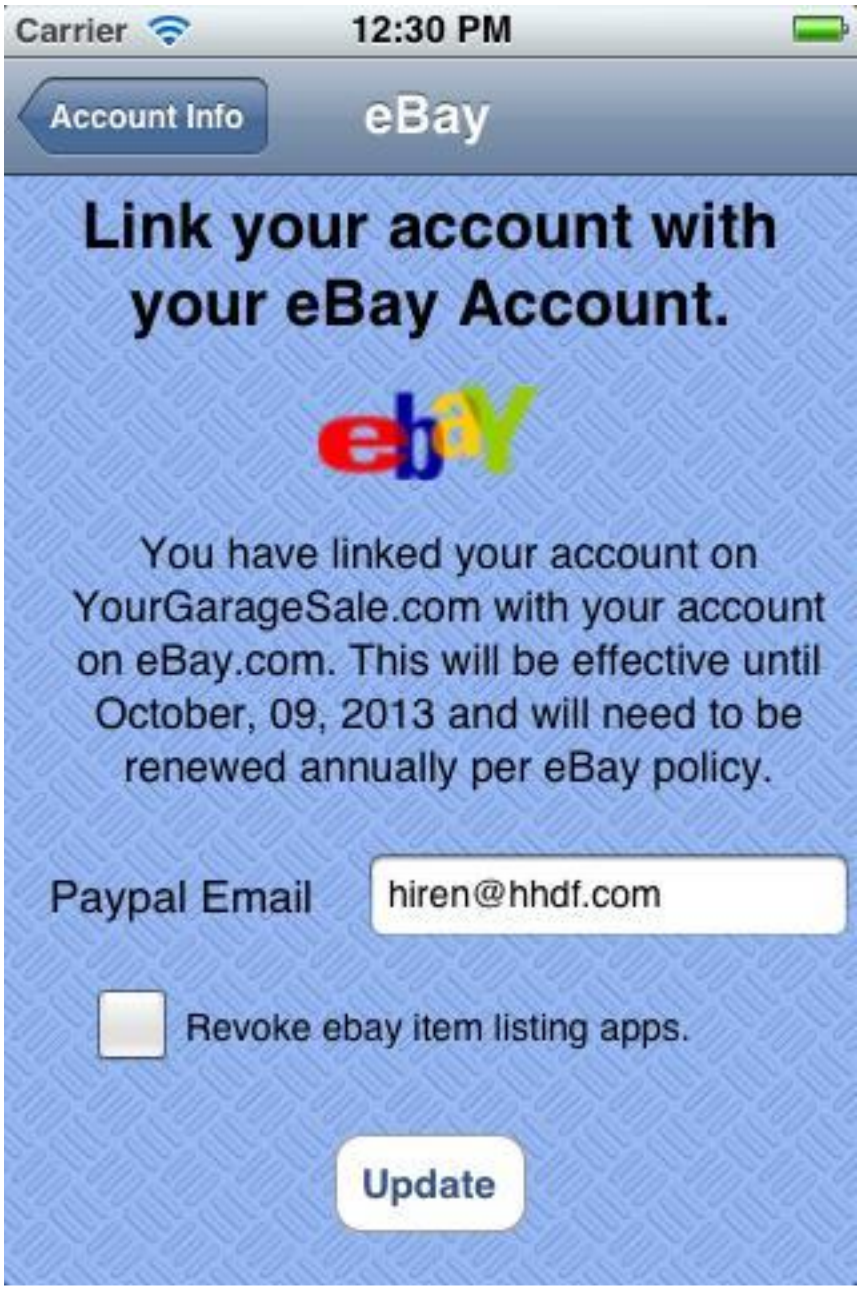


Link your account with your Facebook Account.

You have linked your facebook account on YourGarageSale.com.

☐ Revoke facebook linkup.

Link Your Account with eBay :-



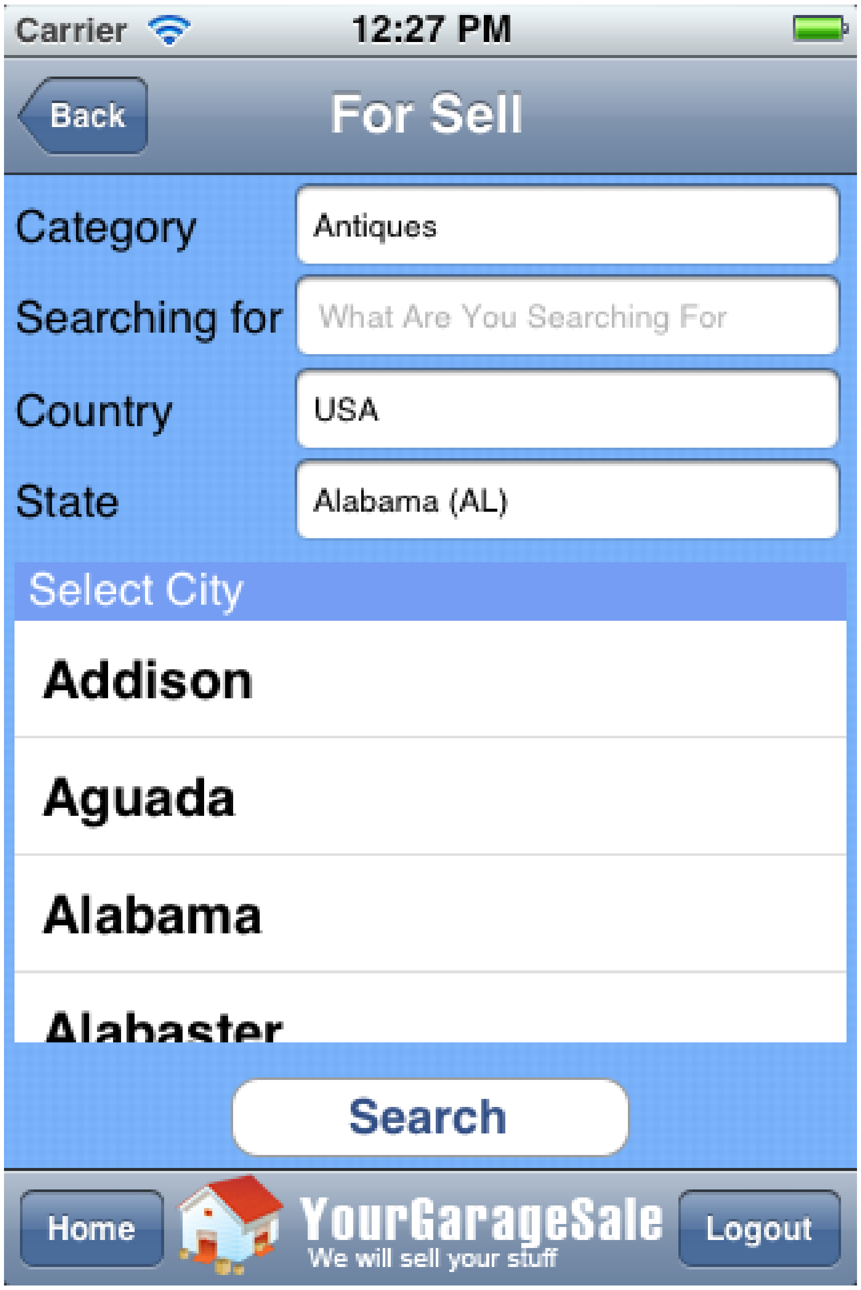
Link your account with your eBay Account.

You have linked your account on YourGarageSale.com with your account on eBay.com. This will be effective until October, 09, 2013 and will need to be renewed annually per eBay policy.

Paypal Email: hiren@hhdf.com

☐ Revoke ebay item listing apps.

Search Screen for sale, rent, trade, free and liquidation all are same with this field for searching criteria :-


Carrier 12:27 PM

Back For Sell

Category	Antiques
Searching for	What Are You Searching For
Country	USA
State	Alabama (AL)

Select City

Addison

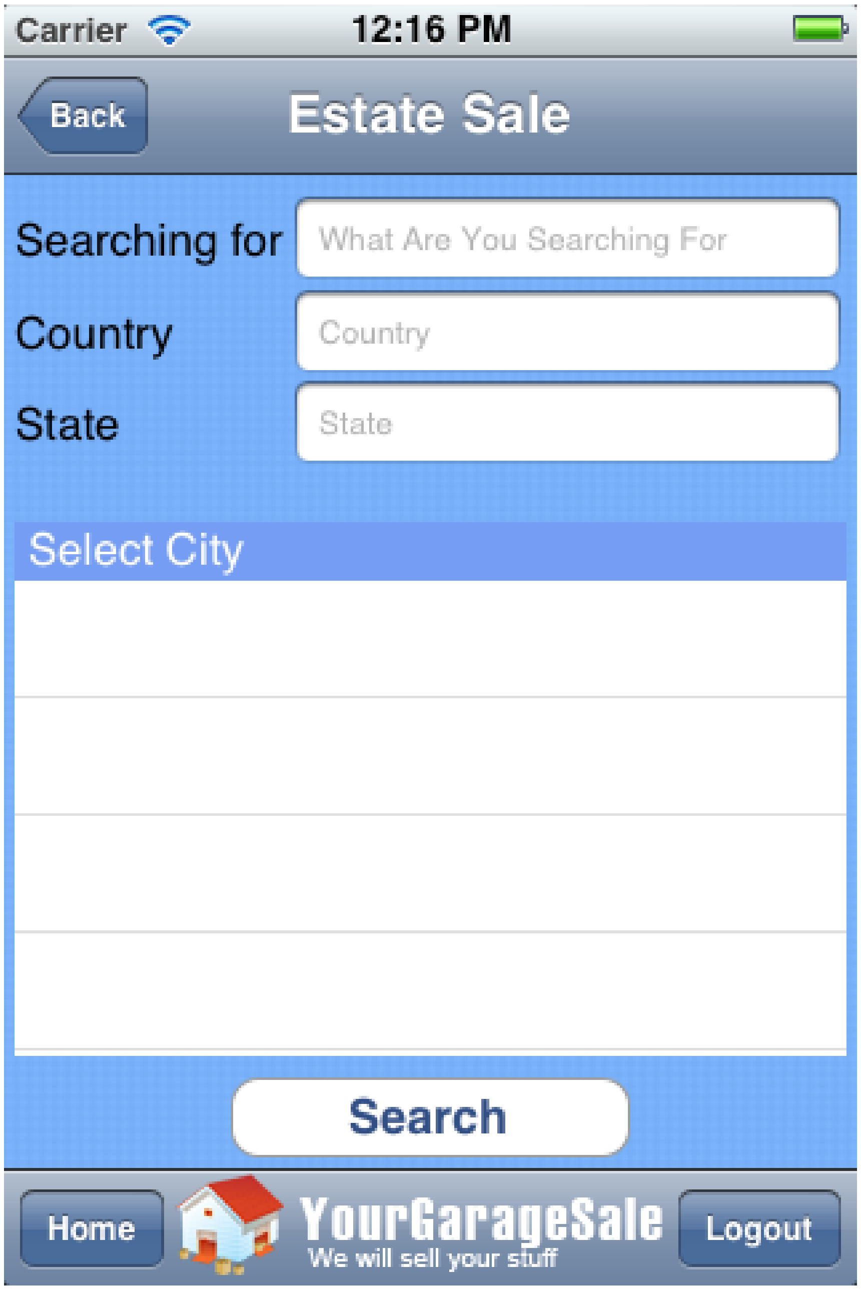
Aguada

Alabama

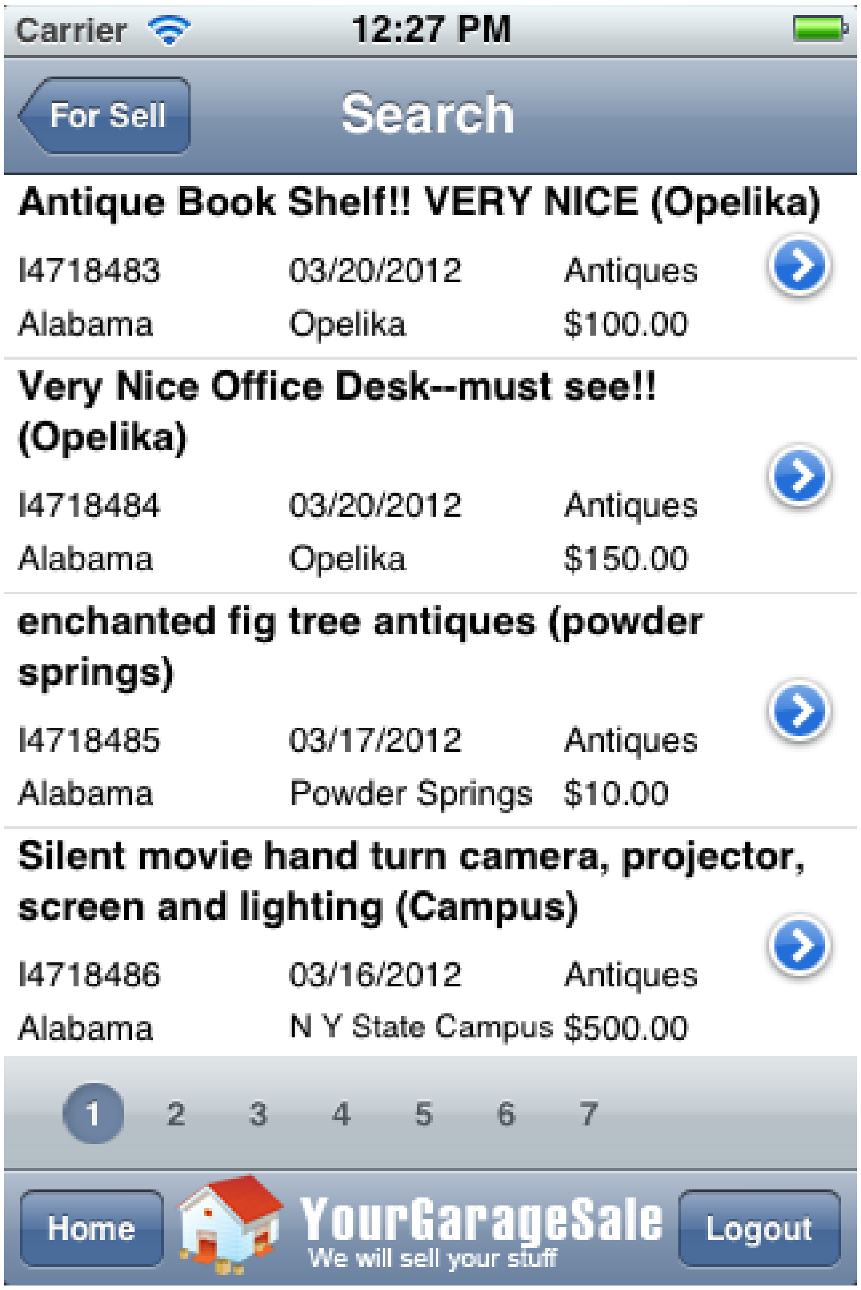
Alabaster

Search

Search Screen for Estate and garage sale are same with this field for searching criteria :-



After filling searching criteria in any field the item meets that criteria are displayed as above screen. It displays all the items, which meet your searching criteria in rent, free, liquidation, trade and sale only :-



Antique Book Shelf!! VERY NICE (Opelika)

I4718483	03/20/2012	Antiques
Alabama	Opelika	$100.00

Very Nice Office Desk--must see!! (Opelika)

I4718484	03/20/2012	Antiques
Alabama	Opelika	$150.00

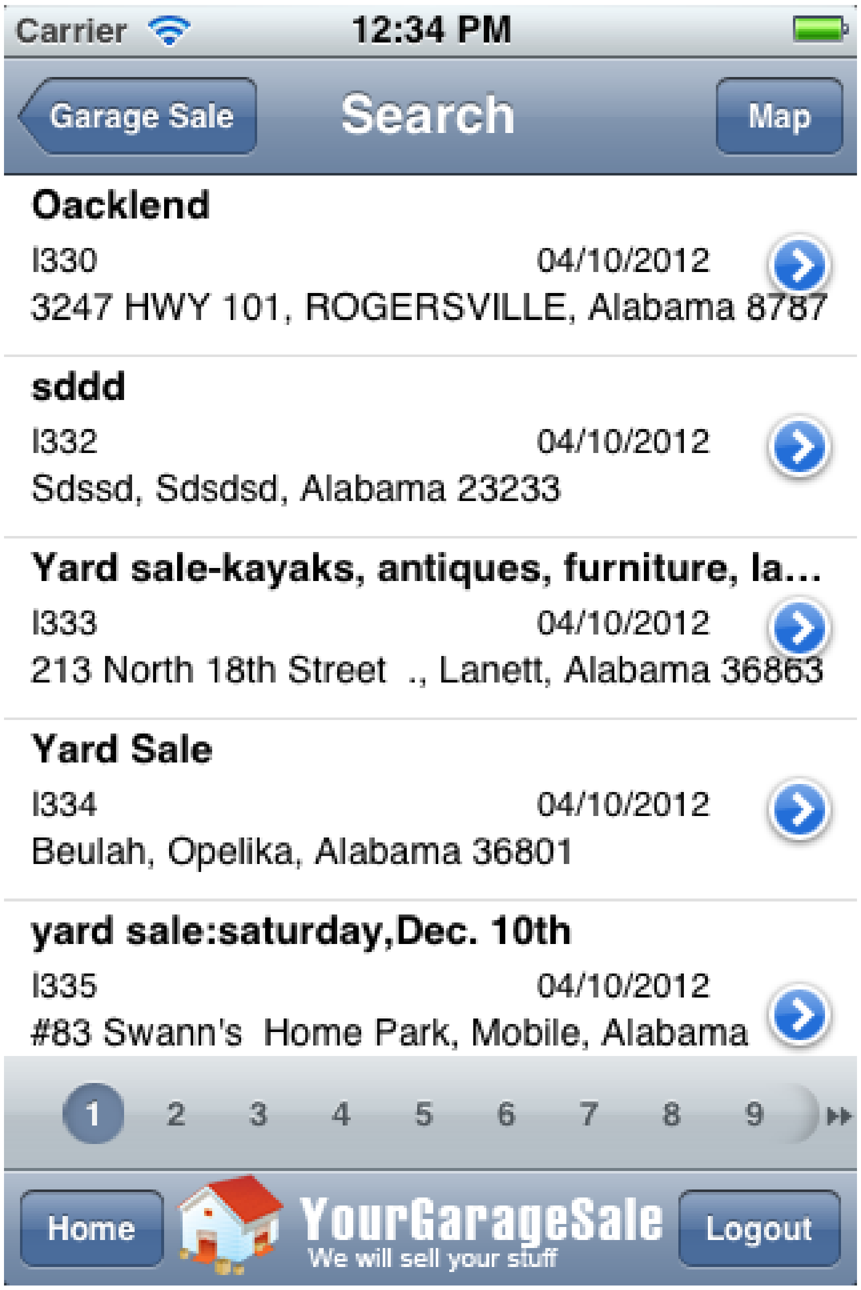
enchanted fig tree antiques (powder springs)

I4718485	03/17/2012	Antiques
Alabama	Powder Springs	$10.00

Silent movie hand turn camera, projector, screen and lighting (Campus)

I4718486	03/16/2012	Antiques
Alabama	N Y State Campus	$500.00

After filling searching criteria in any field the item meets that criteria are displayed and one extra feature is added for just garage and estate sale that is MAP which displays location of that item :-



Oacklend
I330 04/10/2012
3247 HWY 101, ROGERSVILLE, Alabama 8787

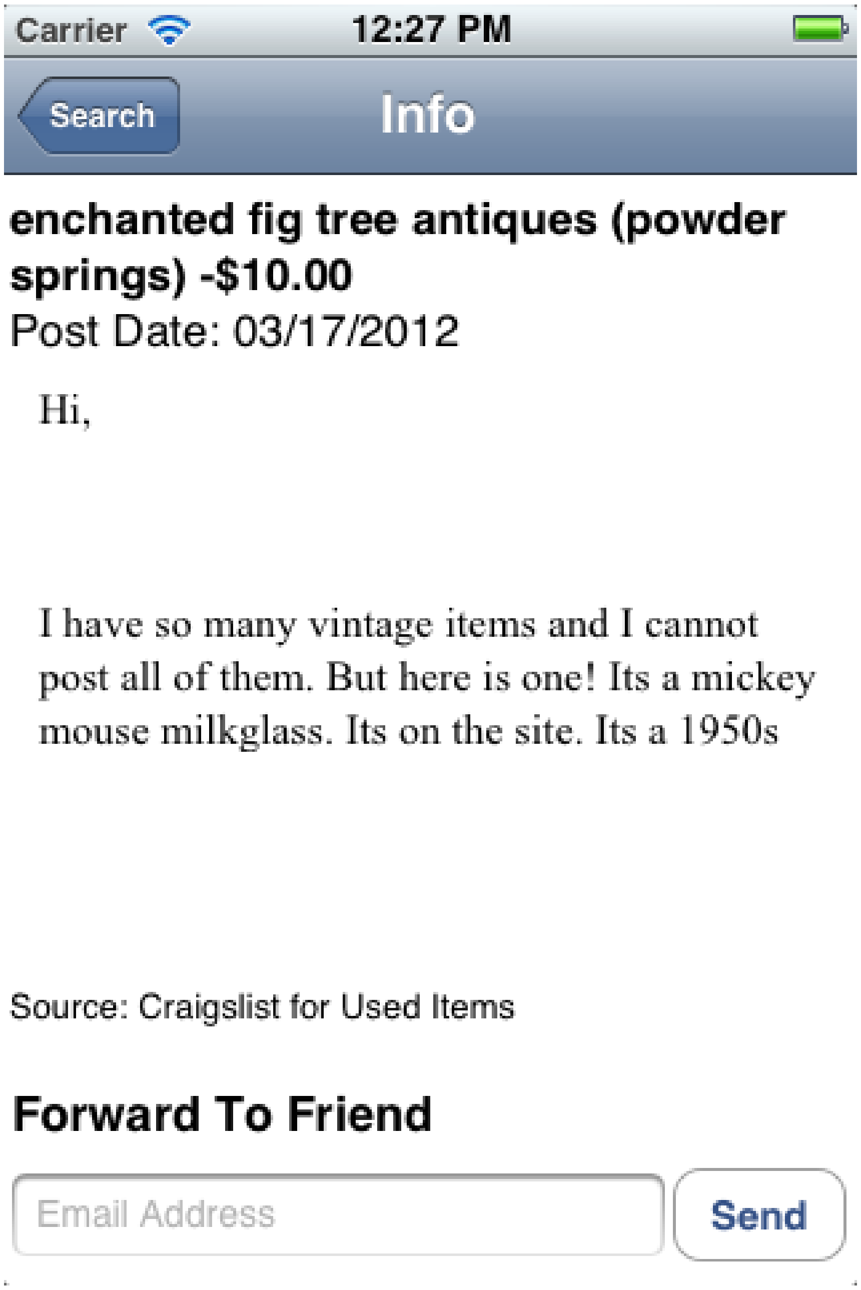
sddd
I332 04/10/2012
Sdssd, Sdsdsd, Alabama 23233

Yard sale-kayaks, antiques, furniture, la...
I333 04/10/2012
213 North 18th Street ., Lanett, Alabama 36863

Yard Sale
I334 04/10/2012
Beulah, Opelika, Alabama 36801

yard sale:saturday,Dec. 10th
I335 04/10/2012
#83 Swann's Home Park, Mobile, Alabama

After selecting in any from any of above two searched item screens the detail of any item selected by clicking on it will display in info screen and allow user to suggest this to friend or contact item owner for purchase :-



enchanted fig tree antiques (powder springs) -$10.00
Post Date: 03/17/2012

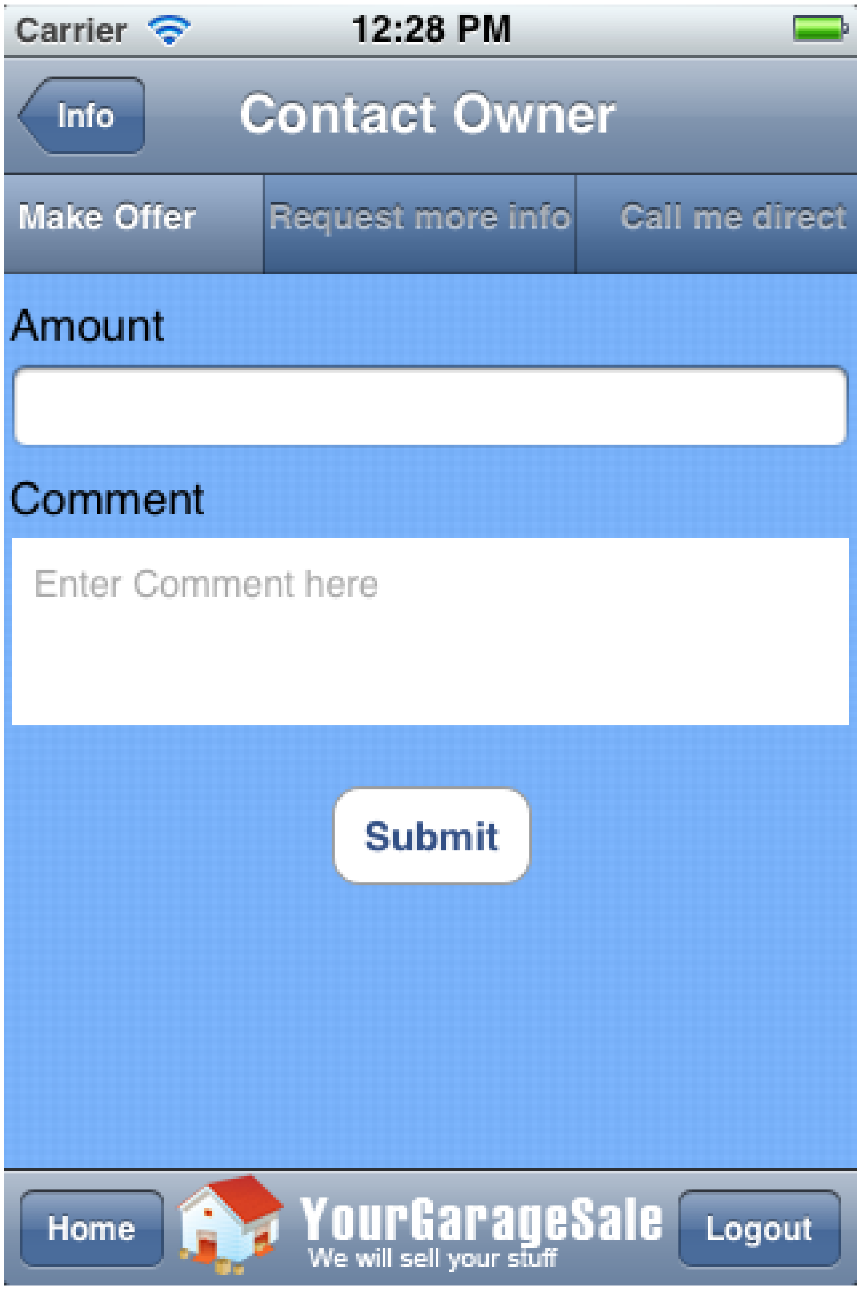
Hi,

I have so many vintage items and I cannot post all of them. But here is one! Its a mickey mouse milkglass. Its on the site. Its a 1950s

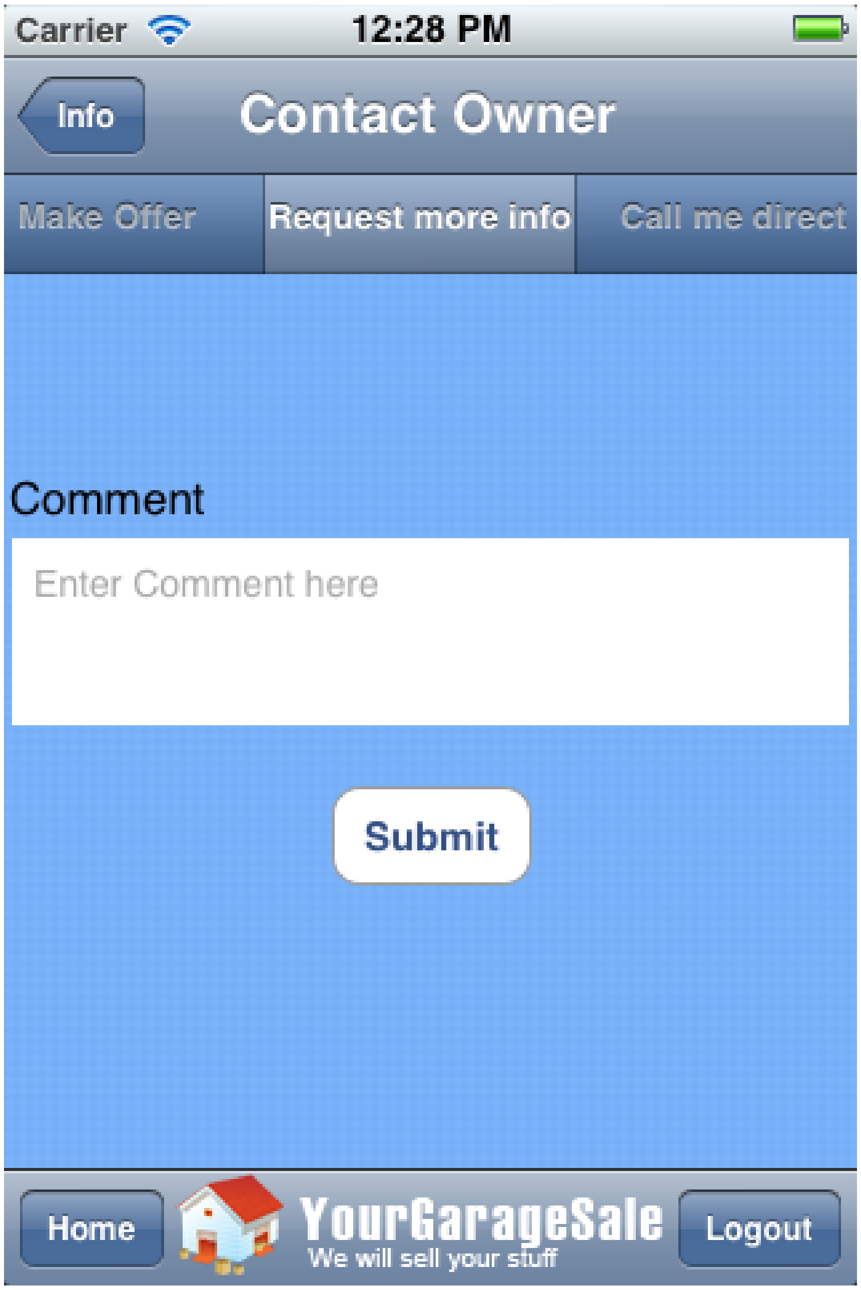
Source: Craigslist for Used Items

Forward To Friend

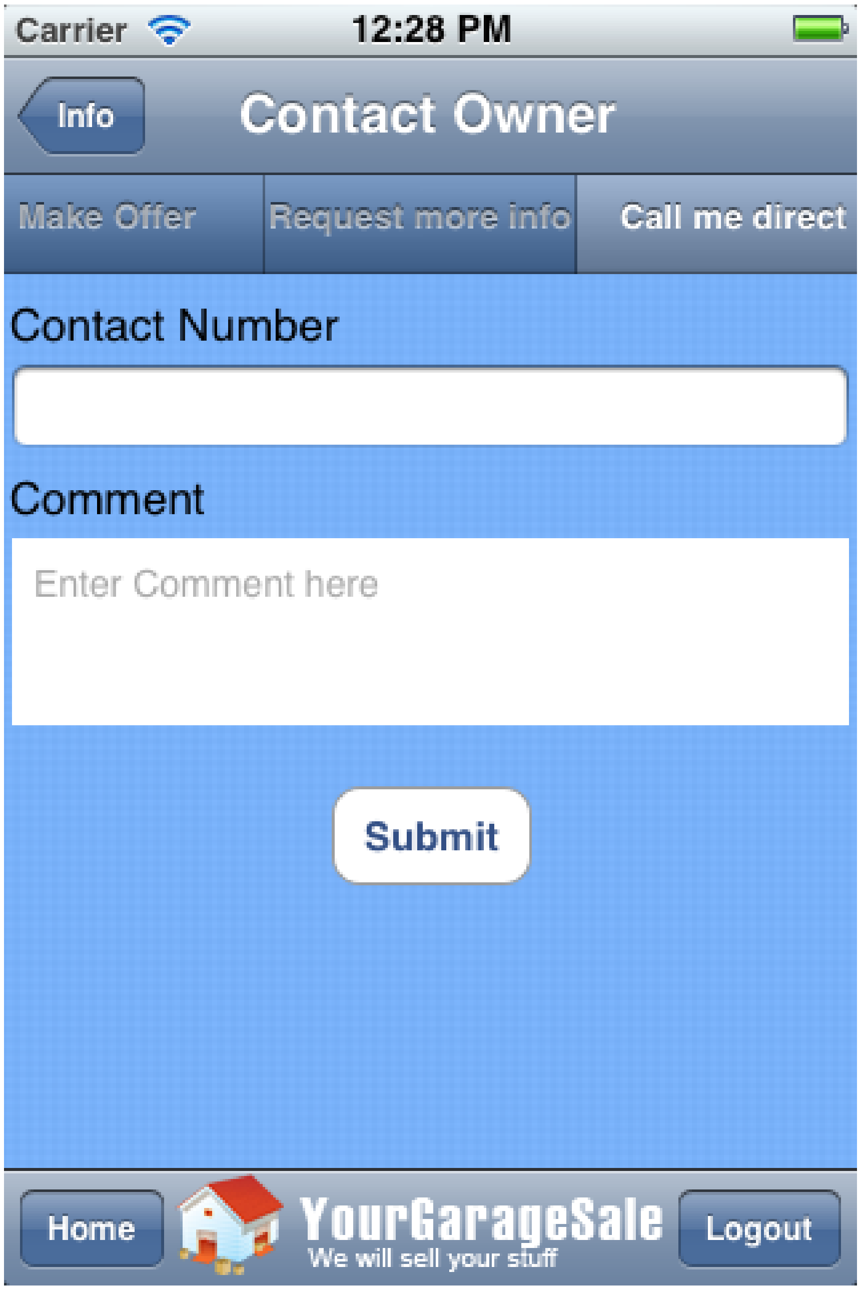
This screen allows the user to make offer for selected things to the owner of that selected thing :-



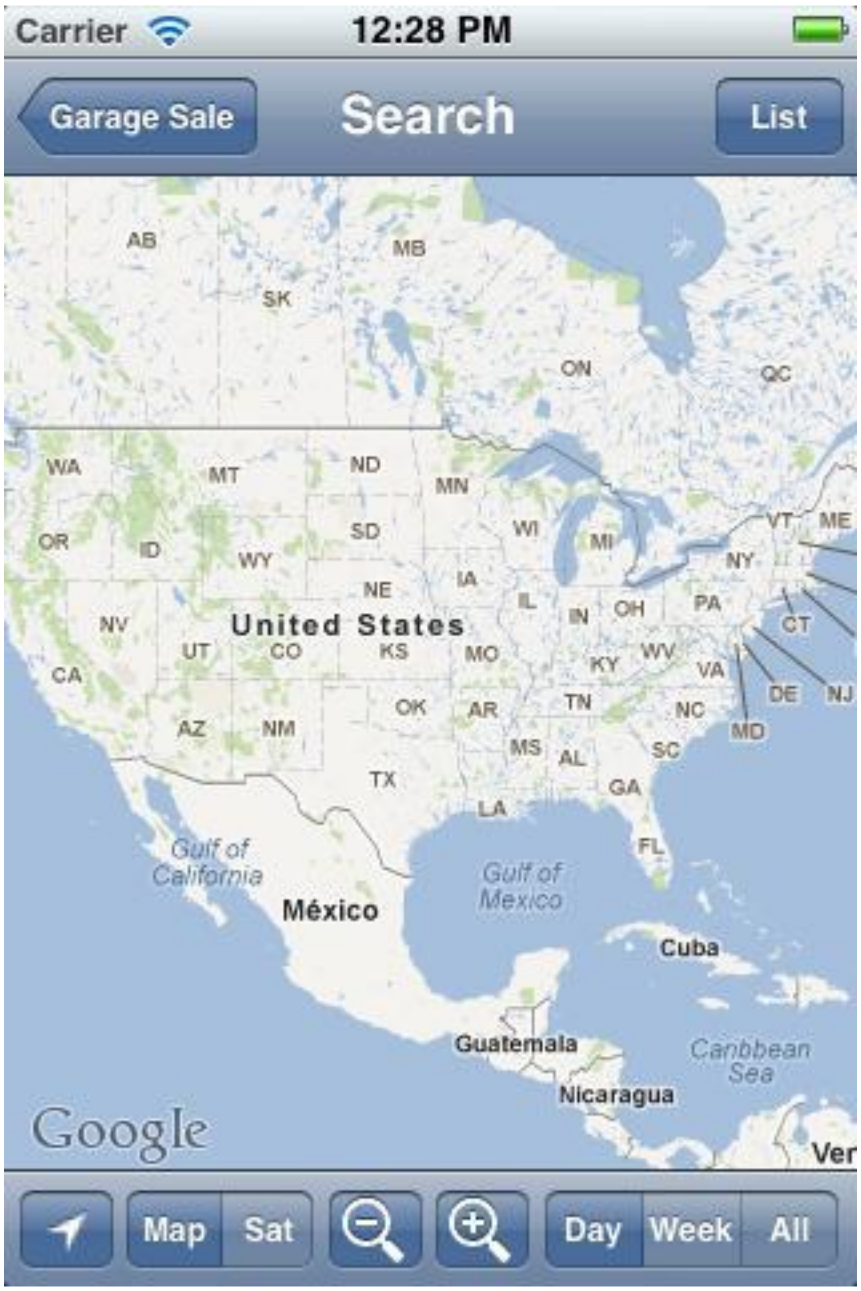
This screen allows the user to comment on particular thing or request to owner for more information on that item :-



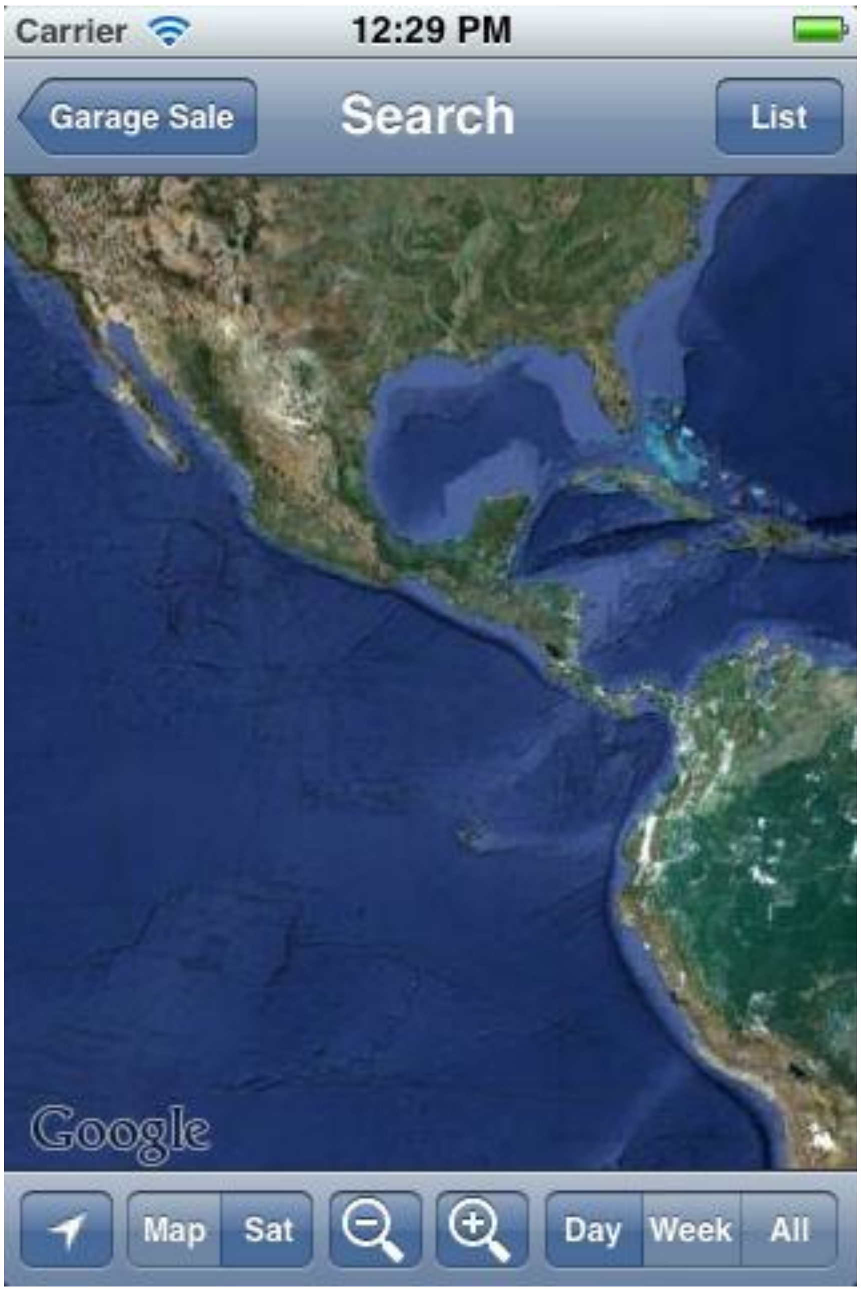
This screen allows the user to give his/her contact information to the owner of that selected thing :-



This screen shows you the location of that selected item in satellite view (only garage and estate item) with other map functionalities :-



This screen shows you the location of that selected item in Hybrid view (only garage and estate item) with other map functionalities :-


Carrier
12:29 PM
Garage Sale
Search
List
Google
Map
Sat
Day
Week
All

LIMITATIONS AND FUTURE ENHANCEMENT

Limitations :-

It is important to note that YGS cannot be considered to replace traditional system. In many ways, YGS shares some similarities to ―life-coaching.‖ They can‗t offer guidance and advice to people experiencing problems in relationships, work, or life in face-to-face communication.

Future Enhancement :-

- YOUR GARAGE SALE will open to all users, who get benefit of products. YGS will may be open for online shopping of estate and other products.
- YGS will work at the interaction of mobile, social and search to all. YGS develops and applies emotional technologies to benefit brands and consumers.
- YGS focus on Product selling and purchasing. Our data helps users to discover items/products, which he want to buy. And also allow them to sale his/her own products.

REFERENCES

http://www.iphonesdkarticles.com/ is good site for iPhone fundamental

Learning iPhone Programming (OREILLY) - Alasdair Allan

It is good book for objective - C Basic Learning Programming.

List Of Books Referred :-

1 - Book Name : Learn iPhone Programming.

Author Name : Alasdair Allan

Publish Date :- March 2010

Best book for basic object oriented programming (OOP) in Objective - C. It is conducted a very basic knowledge for learning object oriented programming concept using C.

2 – Book Name :- Professional iphone and ipad application development .

Author Name :- Gene Backlin .

Publisher :- Wiley Publishing.

Publish Date :- January 2010

This book is providing a very powerful programming concept for Objective - C. It also includes a high level concept.

List Of Website Referred

1 :- Web-Site URL: http://www.iphonesdkarticles.com/

This site is provides information and example of

MVC (Model-View-Controller) pattern.

2 :- Web-Site URL: http://www.iphonedevx.com

The best resource for Objective - C tutorial, and

Video tutorials.

3] Web-Site URL: http://iphonedevelopertips.com

Best site having good Objective-C tips and tricks with examples and

User community.

4] Web-Site URL: http://www.iphonedevsdk.com

This is good site for giving lots of example of objective –C and give

5] Web-Site URL: http://stackoverflow.com/

This site give good example and suggestion of iPhone and iOS development.

6] Web-Site URL: http://apple.com/iPhone

This is Apple site for iPhone developer who is newly joint iPhone program

7] Web-Site URL: http://apple.com/iPad

This site give the various latest information about iPad application and newly

Launch tools and applications.

8] Web-Site URL: http://lynda.com/iPohne-Learning-in-New-Ways

This is Good site give very basic concept video and go through high level of apple application development.