# HOME SERVICES AND UTILITIES SYSTEM

## ( IOS APPLICATION)

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

**Computer Science and Engineering/Information Technology**

By

Naresh Motwani(121242)

Under the supervision of

Ms. Nishtha Ahuja

to


Department of Computer Science & Engineering and Information Technology
**Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh**

# CERTIFICATE

**Candidate's Declaration**

I hereby declare that the work presented in this report entitled **"HOME SERVICES AND UTILITIES SYSTEM"** is a insight to my **Training progress, Learning** and in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology, Waknaghat.

An authentic record of my own work carried out over a period from March 2016 to June 2016 under the supervision of Ms.Nishtha Ahuja(CSE Department)

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

(Student Signature)
Student Name:-Naresh Motwani
Rollno:-121242

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature)
Nishtha Ahuja
Department name- Computer Science Engineering
Dated: 10-June-2016

## ACKNOWLEDGEMENT

It was a great chance for learning and professional development I had with JUIT for the help and Support..

I would like to express our deepest gratitude and special thanks to Ms.Nishtha Ahuja who in spite of being extraordinarily busy with her duties, took time out to hear, guide and keep us on the correct path and allowing me to carry out my project. I choose this moment to acknowledge her contribution gratefully.

I would like to express our special gratitude and thanks to JUIT for giving me such facilities and to the People who have willingly supported us with the Project, invoking their own efforts as well.

# TABLE OF CONTENTS

# LIST OF FIGURES

## ABSTRACT

"HOME SERVICES AND UTILITIES SYSTEM" is an IOS Application to assist the user in all kind of services and utilities

1. To provide the user home appliances repair services by serviceman
2. To show nearby ATM's , Shops , Restaurants etc
3. To book electrician, plumber ,Carpenter etc

This application makes your life easy. With this app you can book services from qualified, background verified professionals near you for all your home needs from anywhere, anytime.

# CHAPTER – 1

# INTRODUCTION

## 1.1 INTRODUCTION

Moved into a new place? Maybe shifted cities? There's a lot of work you need done around the house, and you probably don't know local contractors who are trustworthy and efficient. Luckily, whether you need help in fixing a leaky tap, or drilling holes into a wall, or getting new fittings made, there are a dozen number of Internet startups in India that serve either as intermediaries, or direct providers of a standardized service to do your laundry, dry-cleaning, plumbing, carpentry, and other home services.

One of the effective strategies to successfully market these services is to do an app based ordering system where your customers can download an app and order the services straight from there, instead of using the cumbersome call center based ordering system

This application provide services that have a lot in common with personal assistants, except that instead of being only app or chat-based, they're accessible via other mediums, the Web, phone, and SMS, all trying to capture a bit of the value chain through aggregation, as listing platforms, or specialization, by offering standardized services with licensed, background professionals at your doorstep.

## 1.2 PROBLEM STATEMENT

To design and develop an IOS application that assists the user in all kind of services and utilities

1. **To provide the user home appliances repair services by serviceman**
2. **To show nearby ATM's , Shops , Restaurants etc**
3. **To book electrician, plumber ,Carpenter etc**
4. **To search for any place and view it in a map.**

This application makes your life easy. With this app you can book services from qualified, background verified professionals near you for all your home needs from anywhere, anytime

## 1.3 OBJECTIVE

With high income nuclear families in metropolitans becoming a norm the scope of maintenance services in India is high. This can be easily observed by entry of numerous players like easyfix, bromberg and handyman entering the market.

One of the challenges in this domain would be getting a pool of expert resources who can undertake a variety of tasks especially quite a number of these services would be done by semi skilled less education handyman.

One of the effective strategies to successfully market these services is to do an app based ordering system where your customers can download an app and order the services straight from there, instead of using the cumbersome call center based ordering system

Demand in Home services sector is rising yet there is huge potential due to large untapped market. While urban households are getting acquainted to the idea, the services have not yet reached tier-2 cities. The biggest challenge in this sector is to get professionals with right attitude onboard and providing uniform and reasonable pricing to the customers while maintaining quality.

There is no reason to break your head if you find a leaking tap in your kitchen in the morning or your microwave not functioning properly, as, you will find the services of professionals coming to your help at beckons call. Calling any Tom, Dick and harry may not solve your problems, as you will find a disorganized market where you can hardly rely on unprofessional service providers from whom expecting service quality, punctuality and safety would be not that much a prudent decision.

The winners will be defined by quality of service, ease of use of app, credibile brand building, training of manpower and ability to scale local geographical areas.

## 1.4 METHODOLOGY

### 1.4.1 TECHNOLOGY USED

Swift. A modern programming language that is safe, fast, and interactive.

Swift is a powerful and intuitive programming language for iOS, OS X, tvOS, and watchOS. Writing Swift code is interactive and fun, the syntax is concise yet expressive, and apps run lightning-fast. Swift is ready for your next project — or addition into your current app — because Swift code works side-by-side with Objective-C.

Swift is a new programming language for iOS, OS X, watchOS, and tvOS apps that builds on the best of C and Objective-C, without the constraints of C compatibility. Swift adopts safe programming patterns and adds modern features to make programming easier, more flexible, and more fun. Swift's clean slate, backed by the mature and much-loved Cocoa and Cocoa Touch frameworks, is an opportunity to reimagine how software development works.

Swift has been years in the making. Apple laid the foundation for Swift by advancing our existing compiler, debugger, and framework infrastructure. We simplified memory management with Automatic Reference Counting (ARC). Our framework stack, built on the solid base of Foundation and Cocoa, has been modernized and standardized throughout. Objective-C itself has evolved to support blocks, collection literals, and modules, enabling framework adoption of modern language technologies without disruption. Thanks to this groundwork, we can now introduce a new language for the future of Apple software development.

Swift feels familiar to Objective-C developers. It adopts the readability of Objective-C's named parameters and the power of Objective-C's dynamic object model. It provides seamless access to existing Cocoa frameworks and mix-and-match interoperability with Objective-C code. Building from this common ground, Swift introduces many new features and unifies the procedural and object-oriented portions of the language.

Swift is friendly to new programmers. It is the first industrial-quality systems programming language that is as expressive and enjoyable as a scripting language. It supports playgrounds, an innovative feature that allows programmers to experiment with Swift code and see the results immediately, without the overhead of building and running an app.

Swift combines the best in modern language thinking with wisdom from the wider Apple engineering culture. The compiler is optimized for performance, and the language is optimized for development, without compromising on either. It's designed to scale from "hello, world" to an entire operating system. All this makes Swift a sound future investment for developers and for Apple.

This project is developed in swift 2.1

### 1.4.2 INTEGRATED DEVELOPMENT ENVIRONMENT USED

**XCODE 7.3**

Xcode is an integrated development environment (IDE) containing a suite of software development tools developed by Apple for developing software for OS X, iOS, WatchOS and tvOS. First released in 2003, the latest stable release is version 7.3 and is available via the Mac App Store free of charge for OS X El Capitan users. Registered developers can download preview releases and previous versions of the suite through the Apple Developer website. However, Apple recently made a beta version of version 7.x of the software available to those of the public with Apple Developer accounts


## 1.5 MODULES

### 1.5.1 LOGIN AND SIGNUP MODULE

The user needs to login into the Application and if not registered , the user can also sign up and get registered as a new user

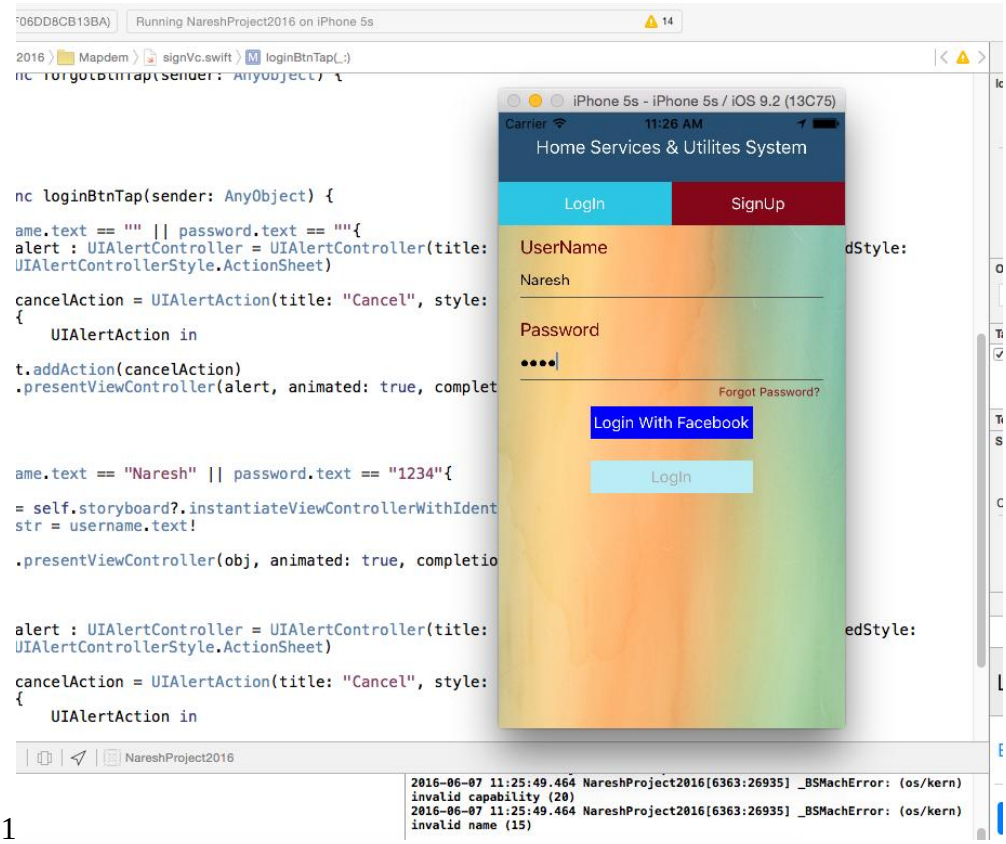

Running NareshProject2016 on iPhone 5s
Mapdem
signVc.swift
loginBtnTap(_:)
nc loginBtnTap(sender: AnyObject) {
ame.text == "" || password.text == ""{
alert : UIAlertController = UIAlertController(title:
UIAlertControllerStyle.ActionSheet)
cancelAction = UIAlertAction(title: "Cancel", style:
{
UIAlertAction in
t.addAction(cancelAction)
.presentViewController(alert, animated: true, complet
ame.text == "Naresh" || password.text == "1234"{
= self.storyboard?.instantiateViewControllerWithIdent
str = username.text!
.presentViewController(obj, animated: true, completio
alert : UIAlertController = UIAlertController(title:
UIAlertControllerStyle.ActionSheet)
cancelAction = UIAlertAction(title: "Cancel", style:
{
UIAlertAction in
iPhone 5s - iPhone 5s / iOS 9.2 (13C75)
Carrier
11:26 AM
Home Services & Utilites System
LogIn
SignUp
UserName
Naresh
Password
Forgot Password?
Login With Facebook
LogIn
NareshProject2016
2016-06-07 11:25:49.464 NareshProject2016[6363:26935] _BSMachError: (os/kern) invalid capability (20)
2016-06-07 11:25:49.464 NareshProject2016[6363:26935] _BSMachError: (os/kern) invalid name (15)

FIG 1

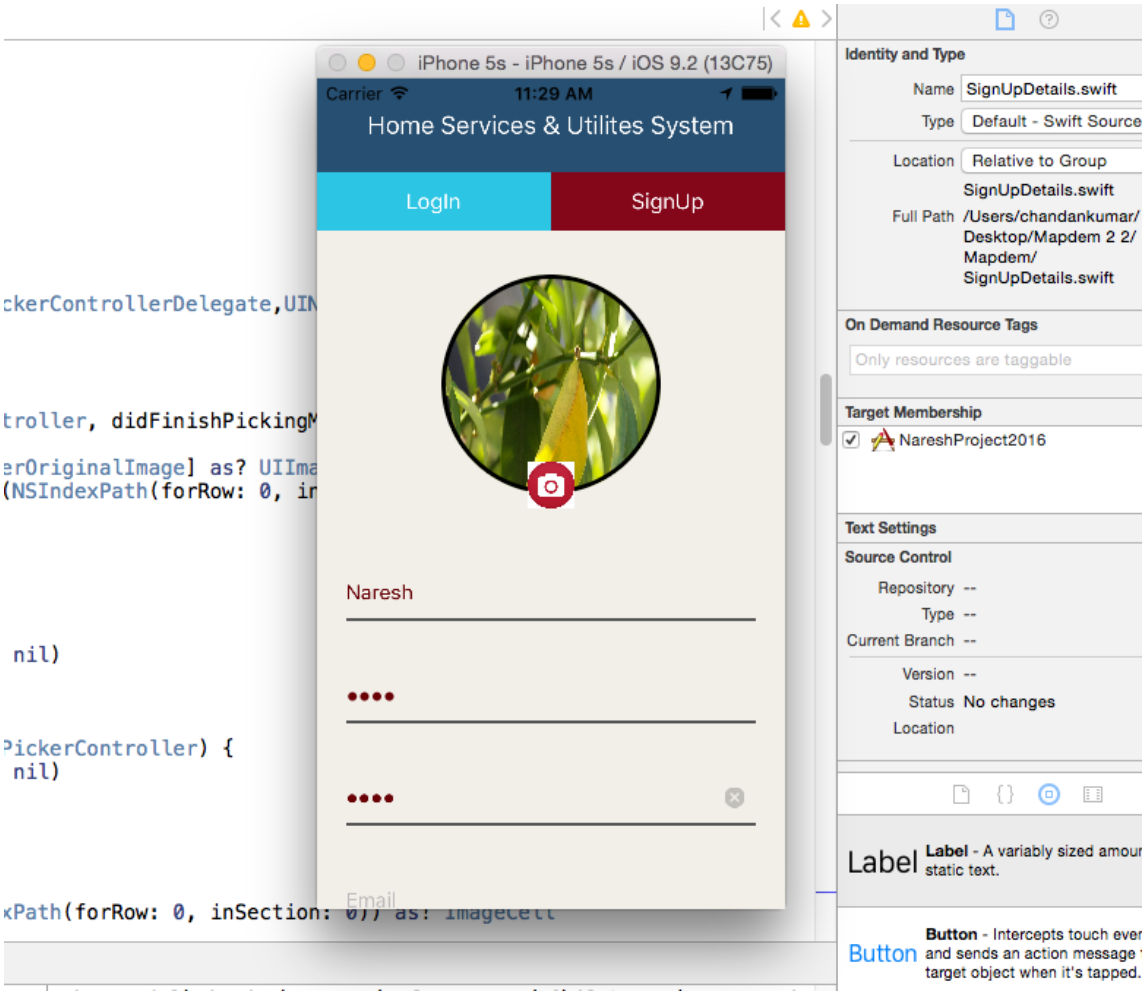

iPhone 5s - iPhone 5s / iOS 9.2 (13C75)
Carrier
11:29 AM
Home Services & Utilites System
LogIn
SignUp
Naresh
Email
Identity and Type
Name SignUpDetails.swift
Type Default - Swift Source
Location Relative to Group
SignUpDetails.swift
Full Path /Users/chandankumar/Desktop/Mapdem 2 2/Mapdem/SignUpDetails.swift
On Demand Resource Tags
Only resources are taggable
Target Membership
NareshProject2016
Text Settings
Source Control
Repository --
Type --
Current Branch --
Version --
Status No changes
Location
Label
Label - A variably sized amour static text.
Button
Button - Intercepts touch ever and sends an action message target object when it's tapped.

**FIG 2**

### 1.5.2 UPDATE PROFILE

The user is also allowed to update the information

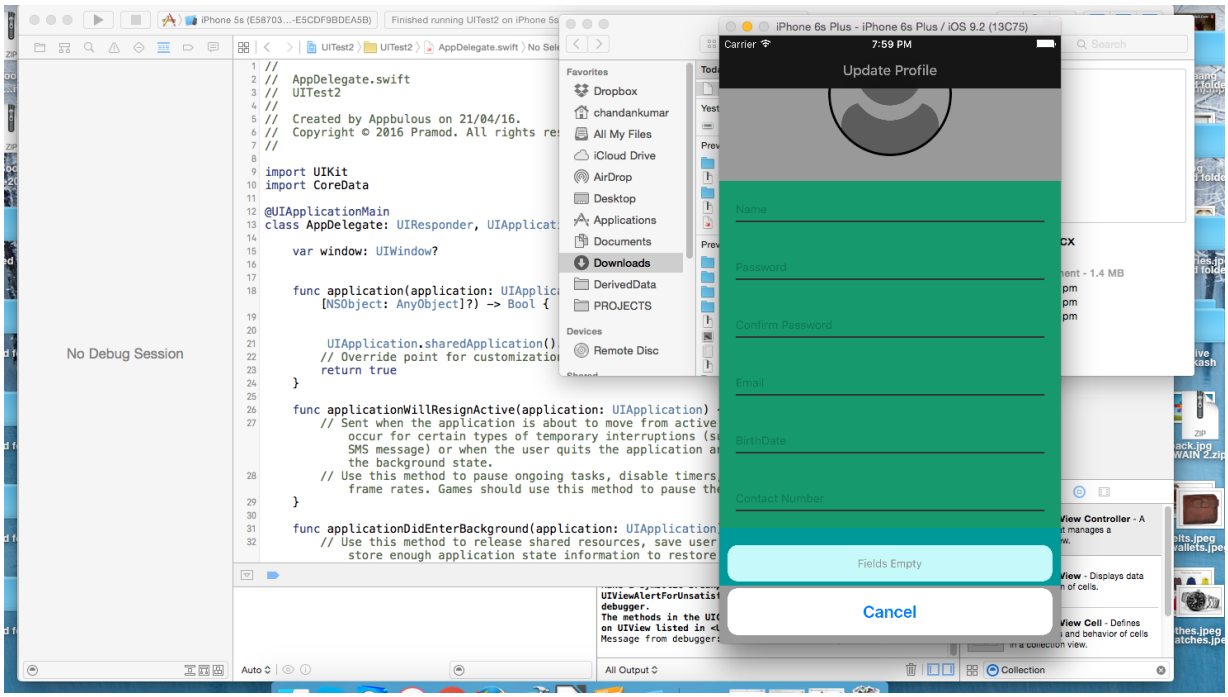



FIG 3

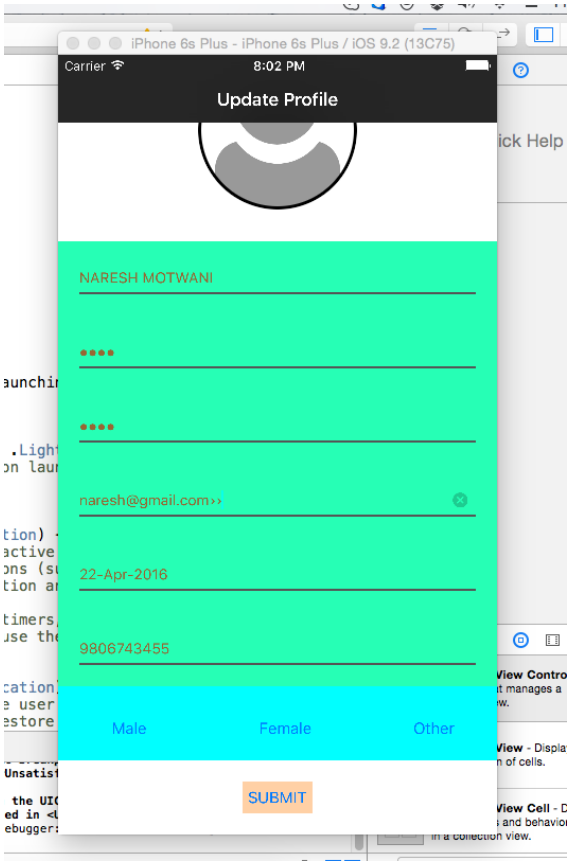


FIG 4

### 4.4.3 DISPLAY THE PROVIDED SERVICES

This module includes display of all the services

- **To provide the user home appliances repair services by serviceman**
- **To show nearby ATM's , Shops , Restaurants etc**
- **To book electrician, plumber ,Carpenter etc**
- **To show all places in list as well as map view**
- **To search for any place**

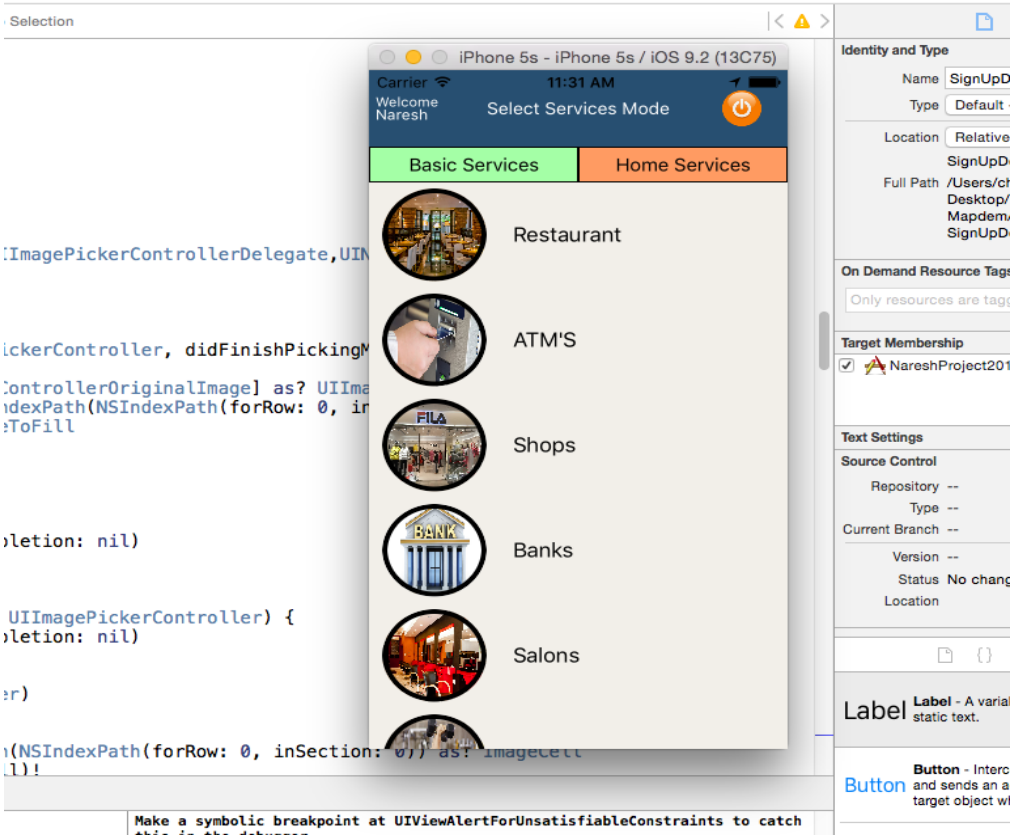

iPhone 5s - iPhone 5s / iOS 9.2 (13C75)
Carrier
11:31 AM
Welcome
Naresh
Select Services Mode
Basic Services
Home Services
Restaurant
ATM'S
Shops
Banks
Salons
Identity and Type
Name
Type
Location
Full Path
On Demand Resource Tags
Target Membership
Text Settings
Source Control
Repository
Type
Current Branch
Version
Status
Location
Label
Button
Make a symbolic breakpoint at UIViewAlertForUnsatisfiableConstraints to catch

FIG 5

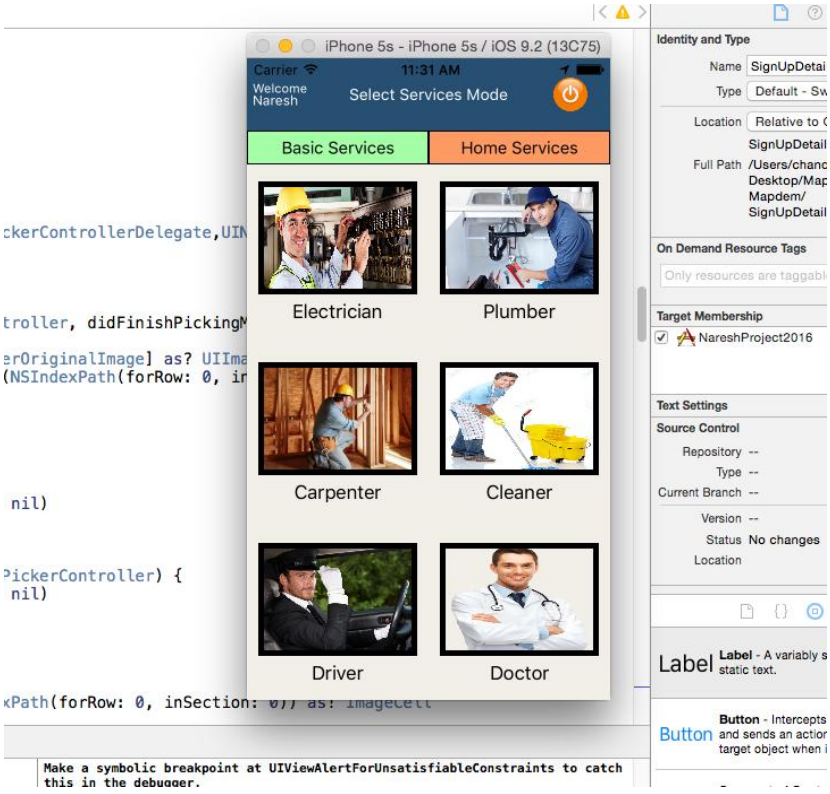
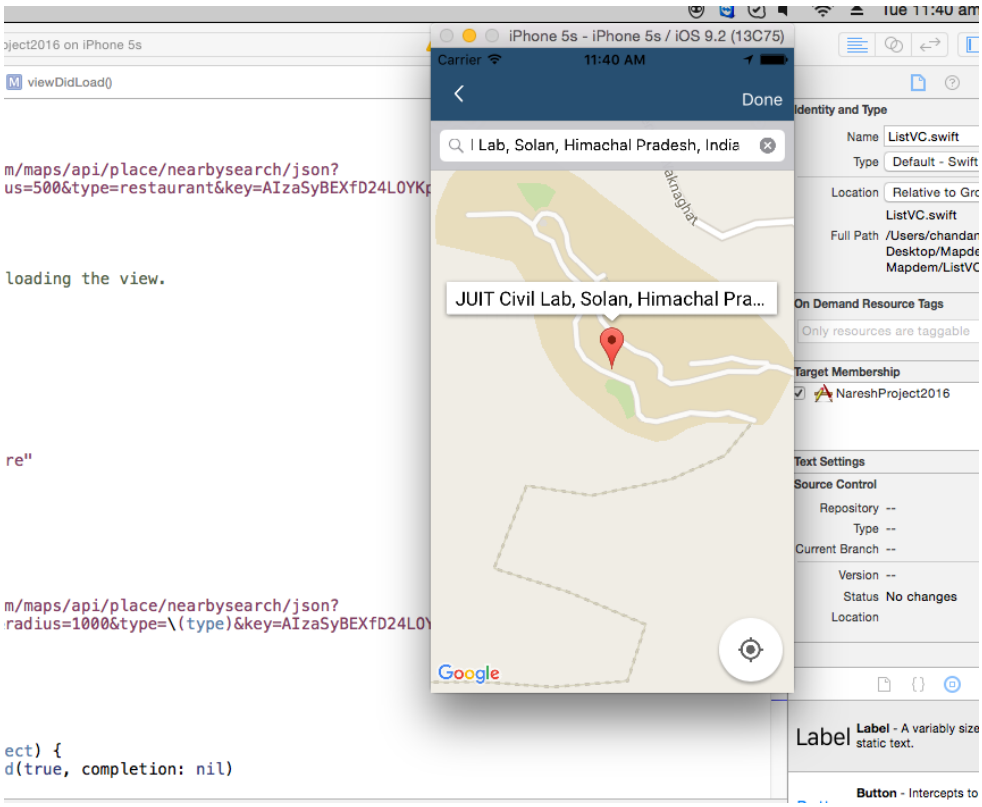

iPhone 5s - iPhone 5s / iOS 9.2 (13C75)
Carrier
11:31 AM
Welcome
Naresh
Select Services Mode
Basic Services
Home Services
Electrician
Plumber
Carpenter
Cleaner
Driver
Doctor
Identity and Type
On Demand Resource Tags
Target Membership
NareshProject2016
Text Settings
Source Control
Repository
Type
Current Branch
Version
Status No changes
Location
Label
Button
Make a symbolic breakpoint at UIViewAlertForUnsatisfiableConstraints to catch

FIG 6


iPhone 5s - iPhone 5s / iOS 9.2 (13C75)
Carrier
11:40 AM
Done
l Lab, Solan, Himachal Pradesh, India
JUIT Civil Lab, Solan, Himachal Pra...
Google
viewDidLoad()
loading the view.
Identity and Type
Name ListVC.swift
Type Default - Swift
Location
ListVC.swift
Full Path
On Demand Resource Tags
Only resources are taggable
Target Membership
NareshProject2016
Text Settings
Source Control
Repository --
Type --
Current Branch --
Version --
Status No changes
Location
Label
Button - Intercepts to
FIG 7

### 4.4.4 BOOKING

The next module is to book a service for a user and display the information

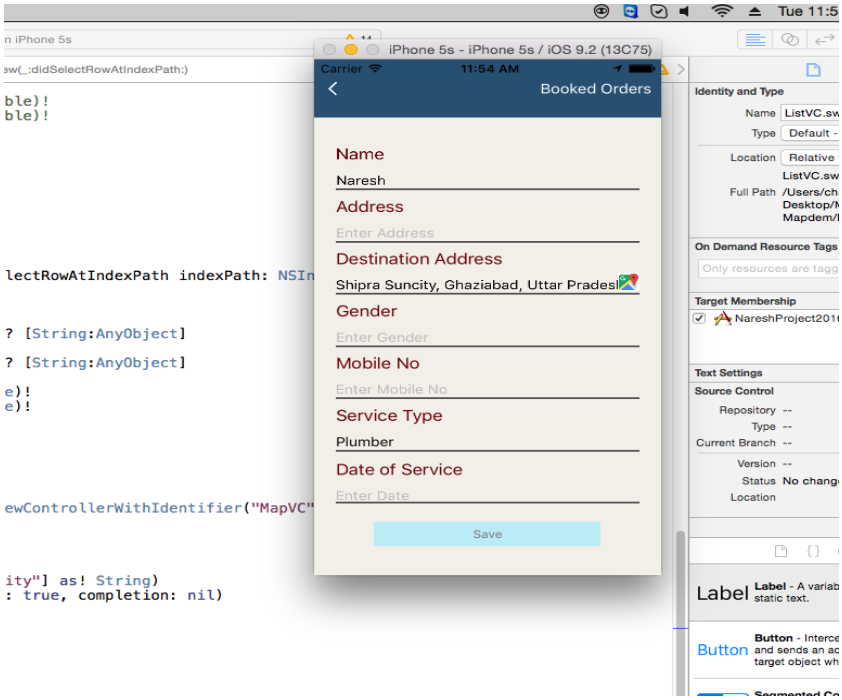



FIG 8

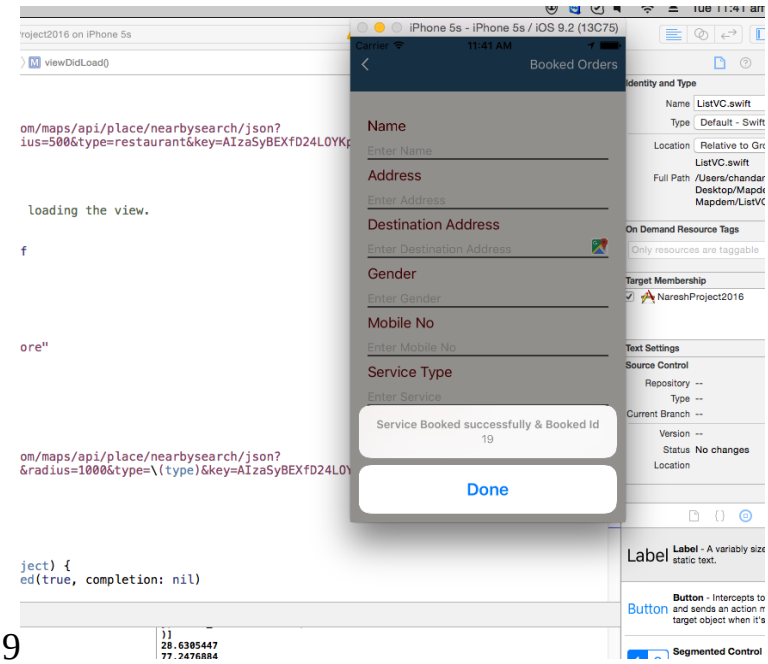



FIG 9

# CHAPTER 2
# LITERATURE SURVEY

## 2.1 OVERVIEW

Services provided at home are becoming increasingly important. Hence, there is a need for information on home care in India. The objective of this literature review was to respond to this need by systematically describing what has been reported on home care in Europe in the scientific literature over the past decade.

**Methods**

A systematic literature search was performed for papers on home care published in English, using the following data bases: Cinahl, the Cochrane Library, Embase, Medline, PsycINFO, Sociological Abstracts, Social Services Abstracts, and Social Care Online. Studies were only included if they complied with the definition of home care, were published between January 1998 and October 2009, and dealt with at least one of the 31 specified countries. Clinical interventions, instrument developments, local projects and reviews were excluded. The data extracted included: the characteristics of the study and aspects of home care 'policy & regulation', 'financing', 'organisation & service delivery', and 'clients & informal carers'.

**Results**

Seventy-four out of 5,133 potentially relevant studies met the inclusion criteria, providing information on 18 countries. Many focused on the characteristics of home care recipients and on the organisation of home care. Geographical inequalities, market forces, quality and integration of services were also among the issues frequently discussed.

**Conclusions**

Home care systems appeared to differ both between and within countries. The papers included, however, provided only a limited picture of home care. Many studies only focused on one aspect of the home care system and international comparative studies were rare.

Furthermore, little information emerged on home care financing and on home care in general in Eastern Europe. This review clearly shows the need for more scientific publications on home care, especially studies comparing countries. A comprehensive and more complete insight into the state of home care in Europe requires the gathering of information using a uniform framework and methodology.

## 2.2 METHOD AND STRATEGY

The following electronic databases were searched: Cinahl, the Cochrane Library, Embase, Medline, PsycINFO, Sociological Abstracts, Social Services Abstracts, and Social Care Online. The search was limited to papers published between January 1998 and October 2009 and to studies involving persons aged 18 years and over when possible.

The search strategy (see Additional file 1 for an example) was first devised for use in Medline and subsequently adapted for the other databases. The search term 'home care services' was used for Medline; terms associated with 'home care' were used for the title or abstract in the other databases if required (MeSH term; major focus and/or exploded depending on the database). In the case of Sociological Abstracts and Social Services Abstracts for instance, the keywords 'home care', 'home help', 'home maker' and 'domiciliary care' were used. In the case of the Social Care Online database for example, the term 'home care' was searched for as a topic or in the title and countries were searched for as a topic only (searching in abstracts was impossible and a search in the free text led to too many irrelevant hits). The search was carried out for the 31 countries covered by the EURHOMAP project, i.e. the 27 EU countries and Switzerland, Norway, Iceland and Croatia. The names of these countries were also included as search terms.

Methods of screening and article selection criteria

An initial screening of publications, based on titles, was performed by two researchers (first AB, then NG). In the second screening round of the remaining publications, titles and abstracts were evaluated by pairs of reviewers independently (NG, DSK, WGWB, AB, WD, ALF, CF, MGM). As a final screening step, the full texts of the remaining publications were

independently assessed for inclusion by pairs of reviewers once more (NG, DSK, WGWB, AB, WD, ALF, CF, MGM). Any discrepancies between reviewers were resolved through consensus and, if necessary, by consulting a third reviewer.

Papers were excluded on the basis of the following criteria:

- published in languages other than English
- not related to the countries specified
- not relevant to the study question
- not in line with the working definition
- reports of effects of specific clinical interventions
- books, reports and dissertations
- reviews (as relevant individual papers would be included)
- published before 2003 (if describing organization or financing of home care)
- studies on which more recent publications were available
- covering instrument developments (e.g. needs assessment instruments)
- covering local (unstructured) projects, personal opinions and experiences

**Data extraction**

After final selection of the papers, information was extracted from the full texts. The following information was extracted from the articles that met the inclusion criteria: the study results, country, author, year of publication, study design, study population, study focus and the home care domains they covered.

The framework used to identify and categorise the features of home care was based on an international comparison previously conducted in EU Member States . This study used the following framework: the context of home care; the specific organisation of home care; and aspects of financing. The 'organisation of home care' comprised the organisations that provide home care, manpower, client populations, provision of services/needs assessment, problems and recent developments, and relations between home nursing and home help services. The data gathered on financing focused on payment and insurance, funding of organisations and

payment of home care professionals. This framework was adjusted after consultation with the EURHOMAP consortium (8 international experts in health services research) and taking into account the information from the studies that met our study criteria.

This resulted in the following four key domains that were used in this literature overview to organise the information: policy & regulation (PR); financing (FI); organisation & service delivery (OS); and clients & informal carers (CI). We distinguished policy and regulation as a separate dimension from financing, organisation and delivery. The area of clients & informal carers was added, because client choice and client-centeredness have become core issues in Europe . Formal acknowledgement of informal care and professional support for informal carers have recently become major policy issues and the demand side (client and informal caregiver) was therefore established as a separate area

## 1.3 ORGANIZATION AND SERVICE DELIVERY

The following key aspects of organisation and service delivery in a home care system were identified in the literature.

***The type of home care providers***: A variety of provision models was found, including monopolist agencies providing comprehensive services in an area; agencies for specific services, such as nursing or domestic care (e.g. in Sweden ); competing commercial and non-commercial private providers and public providers. Private provision (including non-profit) was growing in several countries, such as Ireland , Finland , Sweden and England , either replacing public provision or compensating for its absence. The introduction of market mechanisms in some countries appeared to have weakened co-governance between the third sector (voluntary sector) and the public sector The for-profit private providers may have been better adapted to the new market forces than the voluntary organisations, as was the case in the UK, where managers of voluntary organisations were more likely to have greater problems with negotiating contracts than private provider managers

***Home care integration with other types of services***: Integrated provision of services was reported to be a major challenge in some articles, e.g. in Portugal and for personal budget

holders in the Netherlands  Integration problems are: poor service coordination as too many professionals are involved in caring for one client; multiple entry points for those seeking home care; and different jurisdictions and budgets applicable to health and social home care. Integration could be achieved by having different disciplines working within one agency and by the use of case managers. Case managers for the coordination of home care services were reported in five countries out of 11, i.e. England, Iceland, Sweden, Italy and Finland  Other integration methods were integrated care teams, reported from Norway [, integrated care trusts in the UK, organisations providing multiple types of home care, such as some domiciliary support services in Portugal , and most Danish  and some Swedish  municipalities. Problems with integrating complementary services and regular home care services were also reported, such as different financial conditions in England and Wales with regard to intermediary care Another issue is the coordination between home care and other services. Coordination between hospital and home care is an issue in the UK ,where intermediary care (home care as well as residential care) has been introduced to speed up hospital discharge and to prevent unnecessary re-admissions. In Finland, home helps also delivered care in residential care units and assisted living arrangements .In Poland, home nurses were often employed by family doctors , thus becoming part of the primary health care system.

***Accessibility of home care*:** In Sweden, geographical variation in access to home care was related to different needs across regions .It is possible that such differences are also related to differences in available resources between regions, as is the case in Spain  and Slovenia Variation in access may also be related to the absence of formalised needs assessment instruments .Assessment was more formalised in France than in the UK and Sweden, where assessors had wider discretionary powers. In Italy and Spain, assessment depended on the region and the assessment team. Lack of standardisation of assessment was also a point of concern in the Netherlands .In general, countries differed in the formalisation of the procedure, the instruments used, the professionals involved, and whether social needs were taken into account in addition to physical needs The organisations performing the assessment could be independent assessment agencies (the Netherlands ) municipal teams independent of provision (Norway ), or governmental organisations (the local social service departments in the UK and the municipal care teams in Finland ). In France, Germany, the UK, Sweden and Spain, needs assessment was followed by the preparation of a care plan that included the

services to be provided and the number of hours . In two countries with public provision, Sweden and Finland , a shift in focus over time was reported from low level needs to those with the highest level of needs.

***Monitoring of care needs***: Several studies mentioned monitoring and reassessment of clients' needs after a period of time. In Sweden, care was only assigned for a few months and was regularly monitored . In the UK, care provided was examined for adequacy after 6 weeks and care plans were adapted every 6 months, while needs were re-assessed after 6 months in Finland .

***Quality of home care***: The quality of home care was discussed for several countries, such as Norway , UK and Sweden . Reported instruments for quality improvement were: strict supervision; use of protocols; and user surveys. An international comparison, restricted to urban sites in 11 countries, showed the quality of home care to be most problematic in the Czech Republic and Italy, and least problematic in the Nordic countries . Satisfaction surveys were used by almost two-thirds of the municipalities in Norway. Quality improvement initiatives in Norway were generally not focused on technical quality .

***Working conditions for home care workers***: Working conditions were also discussed in several papers. A study in Northern Ireland showed that home care workers were dissatisfied with irregular working hours, lack of management support and workload pressures . Burn-outs were reported among home care workers in the Netherlands . The position of workers in the private sector was weaker than in the public sector in Ireland, in terms of payment, working conditions and qualifications .

***Increasing self-care ability***: 'Re-ablement' programmes were set up in the UK, with the objective of enhancing self-care among dependent people and hence empowering them to live at home. Municipalities in Denmark were legally obliged to carry out preventive home visits to citizens over the age of 75 with the aim of fostering the functional abilities of these citizens and improving the use of their own resources . In a Polish rural area, most home visits by family nurses were devoted to health education .

## 1.4 CONCLUSION

This review has provided a systematic overview of the recent scientific knowledge on home care. It had a broader geographical scope than previous overviews of home care systems [4, 7. The focus on scientific literature was chosen to safeguard the quality of the information, but it may have been a limitation as well. Good studies may also appear as grey literature, in particular those published in other languages than English, which is the usual language of many international scientific journals. So, the conclusions of this review are not based on possible relevant publications in other languages than English. Another limitation has been that no free text search for 'home care' was carried out in the final design of the review. Such a search has been done provisionally but has been rejected because it resulted in unmanageable large numbers of hits with no or minor relevance to home care. The use of a limited number of databases for the search could be another source of missed information. To ensure coverage of the two main areas of home care, that is, social services and health care services, data bases from both areas were used; three for social services and five for health care. Given the coverage of these data bases, it is reasonable, however, to assume that major articles on home care would be retrieved with this approach.

The field is developing day by day, many applications are now available on various platforms to provide home services and utilities for the complete support.

# CHAPTER 3

# SYSTEM DEVELOPMENT

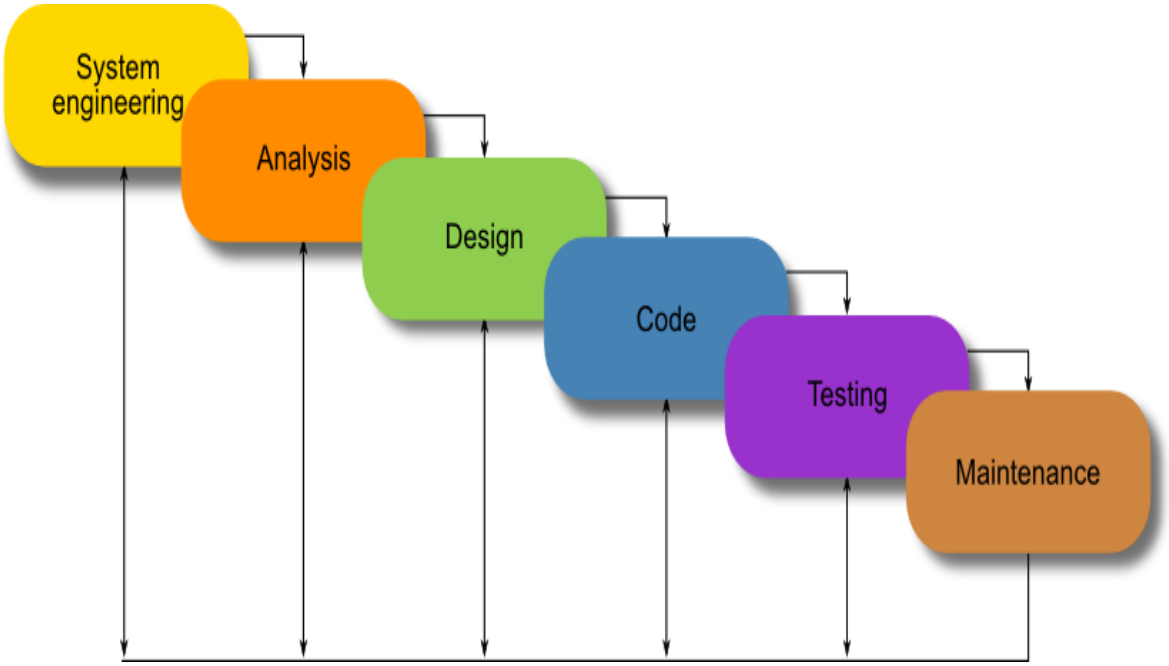
## 3.1 DEVELOPMENT LIFE CYCLE

### Waterfall Model:-

The waterfall model is a sequential design process, used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, production/implementation and maintenance.

The waterfall development model originates in the manufacturing and construction industries: highly structured physical environments in which after-the-fact changes are prohibitively costly, if not impossible.

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.




The sequential phases in Waterfall model are:

- **Requirement Gathering and analysis:** All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.

- **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

- **Implementation:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

- **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

- **Deployment of system:** Once the functional and non- functional testing is done, the product is deployed in the customer environment or released into the market.

- **Maintenance:** There are some issues which come up in the client environment. To fix those issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model phases do not overlap.

## 3.2 SYSTEM REQUIREMENTS

**1. Software Requirements**-

- Client Side:-

    1) Apple device such as i-phone or i-pad
    2) IOS Operating system

- Developer Side :-

    1) SDK
    2) Swift 2.1
    3) XCODE 7.3
    4) Emulator

**2. Hardware Requirements:**

- Client Side:

    1) 512 MB RAM or More
    2) 1 GHz or above advanced processors

- Developer Side :

    1) 4 GB RAM
    2) Resolution of 1024x768.
    3) Mac OS X
    4) Hard disk 20 GB free space
    5) Apple System for development

## 3.3 ALGORITHM AND CONCEPTS

### 3.3.1 USE OF CORE CONCEPTS AND ELEMENTS

- UI TABLE VIEW
- UI COLLECTION VIEW
- DATE PICKER
- UI PICKER VIEW
- UI VIEW
- DATA VERIFICATION
- FIELD VALIDATIONS
- INTEGRATING GOOGLE MAPS
- WEB SERVICES
- MAPS API
- PLACES API
- PLACE PICKER
- PLACES AUTOCOMPLETE API
- JSON PARSING
- DATA TRANSFER
- CORE DATA

## 3.4 API'S USED

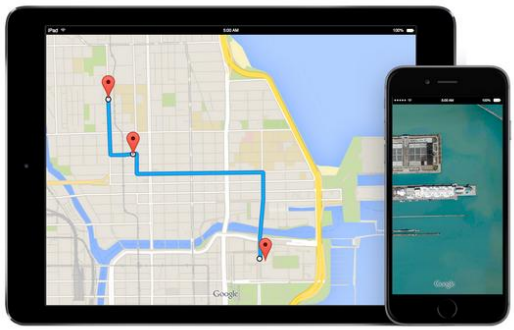
### GOOGLE MAPS API

Always lead the way

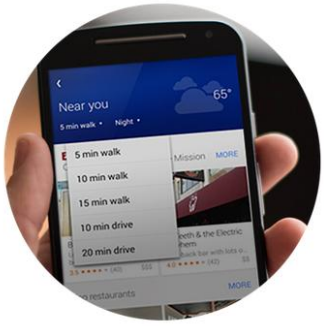
Build full-featured iOS apps for your users. Google Maps APIs for iOS are available via CocoaPods so your app can be location-aware, include data-rich maps, find relevant places nearby and more.

**Google Maps SDK for iOS**
Add Google Maps to your iOS app.

**Google Places API for iOS**
Implement device place detection, auto-complete and add information about millions of locations to your app.



### GOOGLE PLACES API

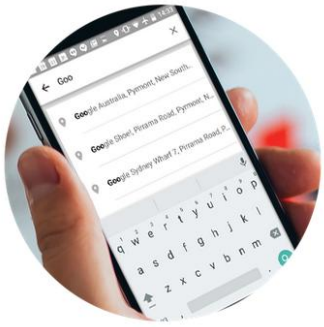



**Location Awareness**

Use the power of mobile to give your users contextual information about where they are, when they're there.


**Search Everywhere**

Search for and retrieve rich information about local businesses and points of interest, available on every screen.



**Autocomplete**

Add autocomplete to any application, providing type-ahead location-based predictions like the search on Google Maps.

## 3.5 UI DESIGN AND DEEVELOPMENT

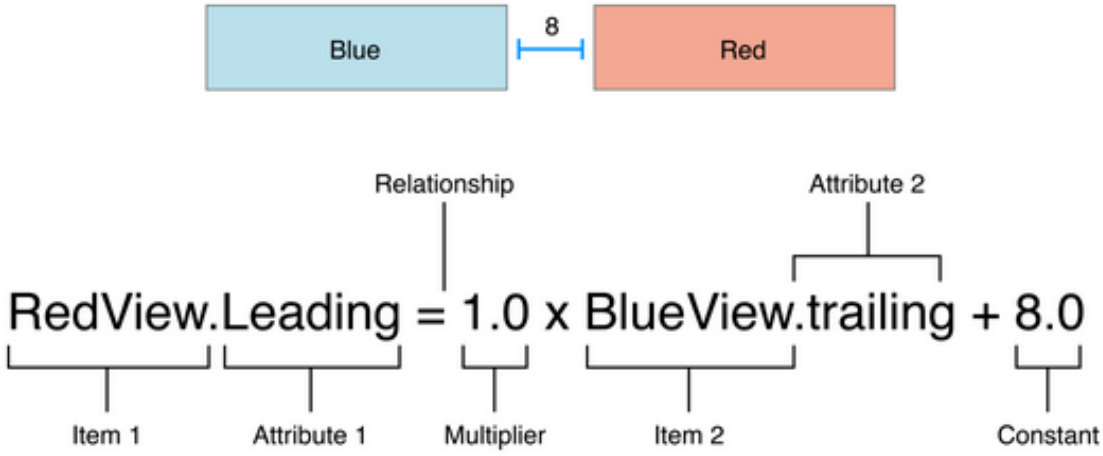
### 3.5.1 AUTO LAYOUT AND CONSTRAINTS

Auto Layout dynamically calculates the size and position of all the views in your view hierarchy, based on constraints placed on those views. For example, you can constrain a button so that it is horizontally centered with an Image view and so that the button's top edge always remains 8 points below the image's bottom. If the image view's size or position changes, the button's position automatically adjusts to match.

This constraint-based approach to design allows you to build user interfaces that dynamically respond to both internal and external changes

The layout of your view hierarchy is defined as a series of linear equations. Each constraint represents a single equation. Your goal is to declare a series of equations that has one and only one possible solution.

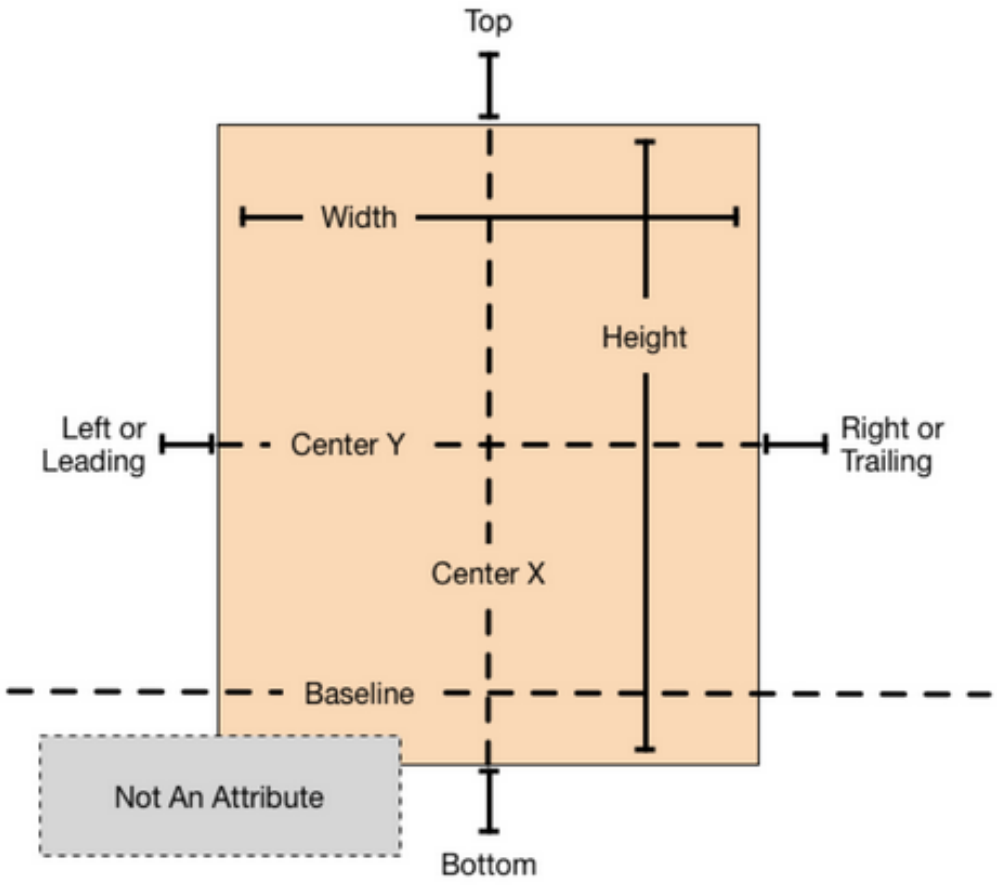
A sample equation is shown below.



This constraint states that the red view's leading edge must be 8.0 points after the blue view's trailing edge. Its equation has a number of parts:

**Item 1**- The first item in the equation—in this case, the red view. The item must be either a view or a layout guide.
**Attribute 1** - The attribute to be constrained on the first item—in this case, the red view's leading edge.
**Relationship** - The relationship between the left and right sides. The relationship can have one of three values: equal, greater than or equal, or less than or equal. In this case, the left and right side are equal.
**Multiplier** - The value of attribute 2 is multiplied by this floating point number. In this case, the multiplier is 1.0.
**Item 2** - The second item in the equation, the blue view. Unlike the first item, this can be left blank.
**Attribute 2**- The attribute to be constrained on the second item—in this case, the blue view's trailing edge. If the second item is left blank, this must be Not an Attribute.
**Constant** - A constant, floating-point offset—in this case, 8.0. This value is added to the value of attribute 2.

Most constraints define a relationship between two items in our user interface. These items can represent either views or layout guides. Constraints can also define the relationship between two different attributes of a single item, for example, setting an aspect ratio between an item's height and width. You can also assign constant values to an item's height or width. When working with constant values, the second item is left blank, the second attribute is set to Not An Attribute, and the multiplier is set to 0.0




**Working with Constraints**

There are three main options for setting up Auto Layout constraints in Interface Builder: You can control-drag between views, you can use the Pin and Align tools, and you can let Interface Builder set up the constraints for you and then edit or modify the results. Each of these approaches has its own set of strengths and weaknesses. Most developers find that they prefer one approach over the others; however, being familiar with all three approaches lets you quickly switch between tools based on the task at hand

To create a constraint between two views, Control-click one of the views and drag to the other.


When you release the mouse, Interface Builder displays a HUD menu with a list of possible constraints.

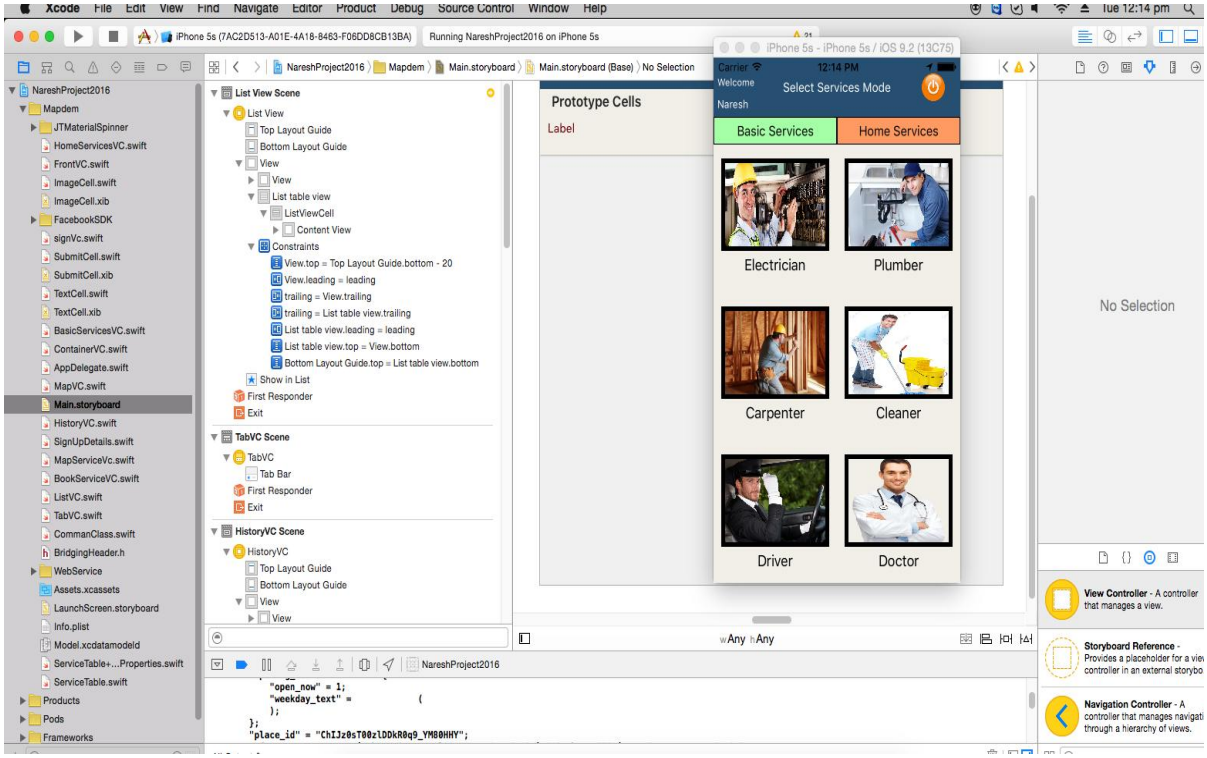



FIG 10

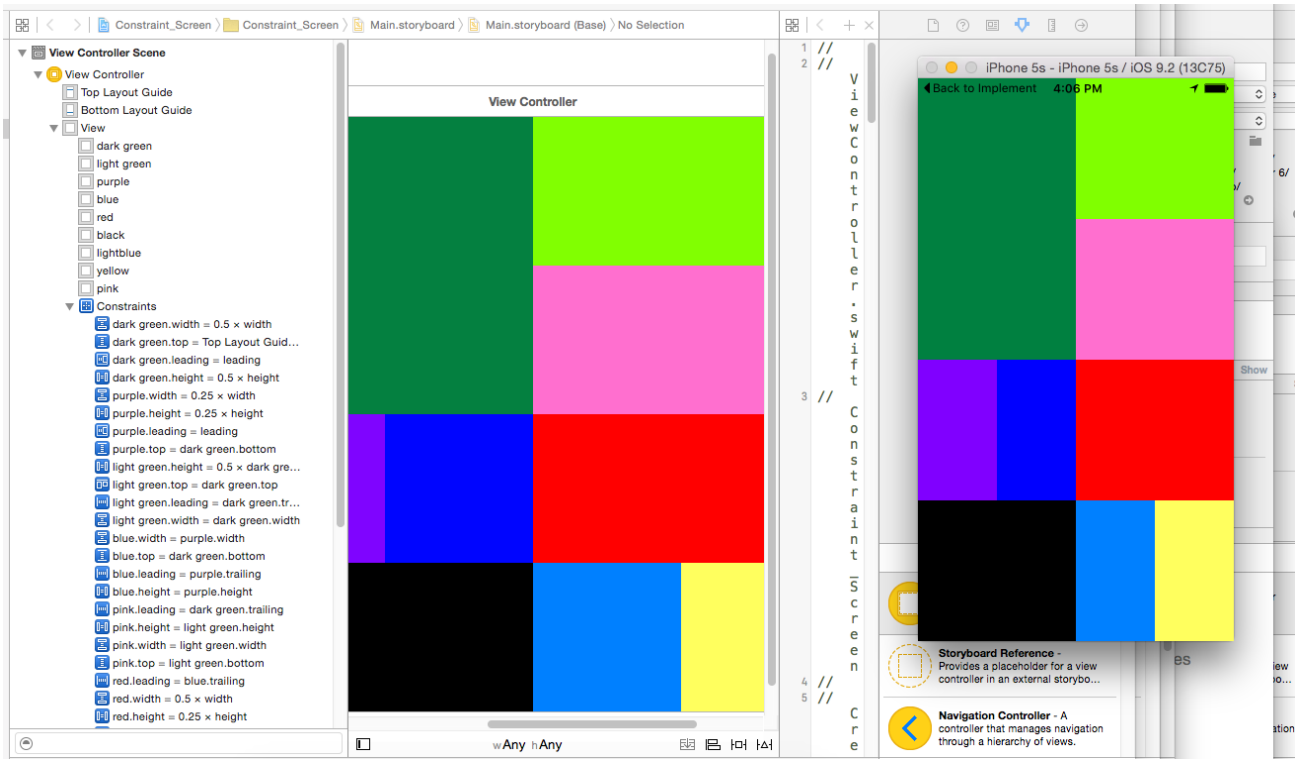



FIG 11

### 3.5.2 UI TABLE VIEW

An instance of UITableView (or simply, a table view) is a means for displaying and editing hierarchical lists of information.

A table view displays a list of items in a single column. UITableView is a subclass of UIScrollView, which allows users to scroll through the table, although UITableView allows vertical scrolling only. The cells comprising the individual items of the table are UITableViewCell objects; UITableView uses these objects to draw the visible rows of the table. Cells have content—titles and images—and can have, near the right edge, accessory views. Standard accessory views are disclosure indicators or detail disclosure buttons; the former leads to the next level in a data hierarchy and the latter leads to a detailed view of a selected item. Accessory views can also be framework controls, such as switches and sliders, or can be custom views. Table views can enter an editing mode where users can insert, delete, and reorder rows of the table.

A table view is made up of zero or more sections, each with its own rows. Sections are identified by their index number within the table view, and rows are identified by their index number within a section. Any section can optionally be preceded by a section header, and optionally be followed by a section footer

### 3.5.2.1 IMPLEMENTATION

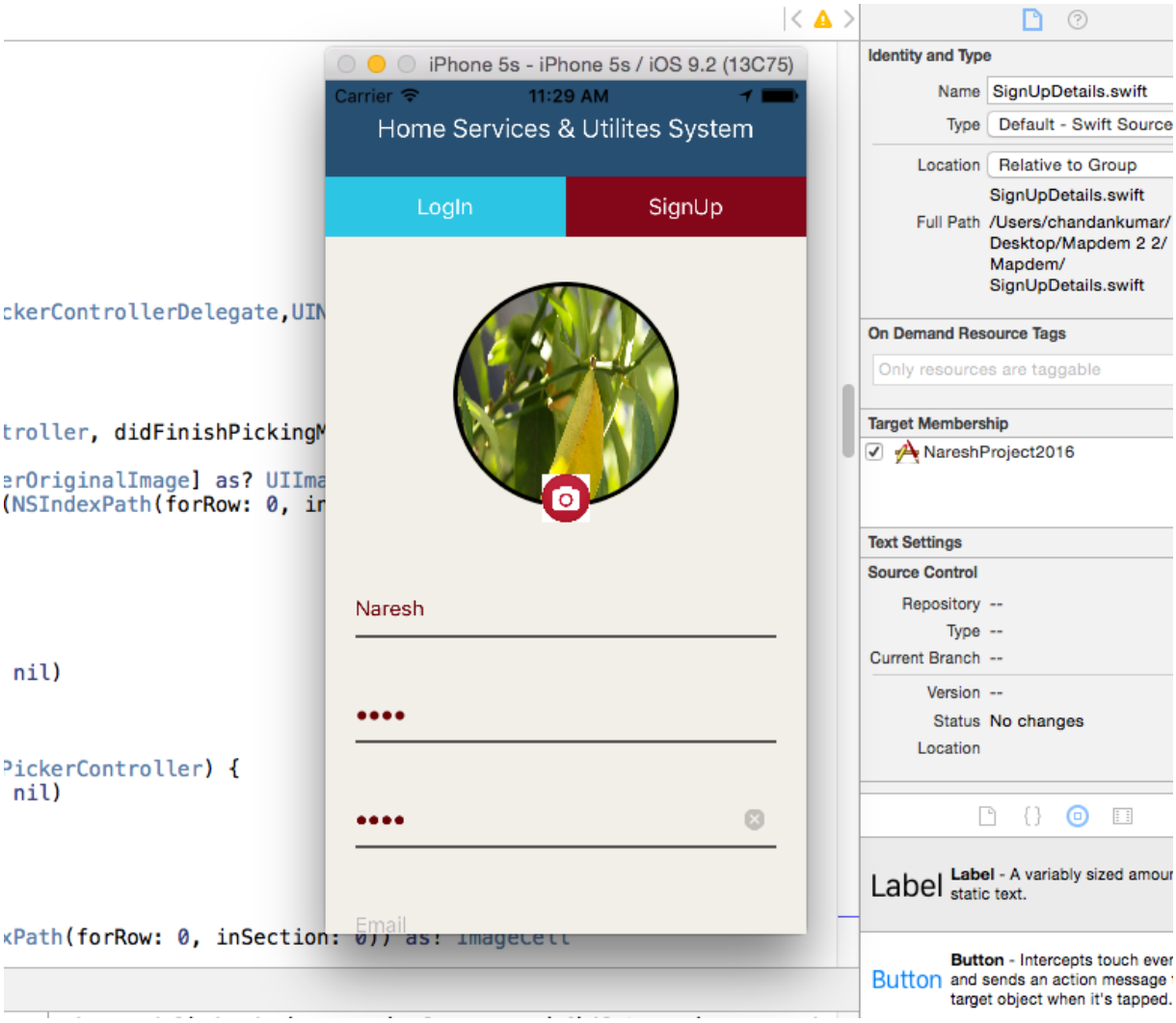



**FIG 12**

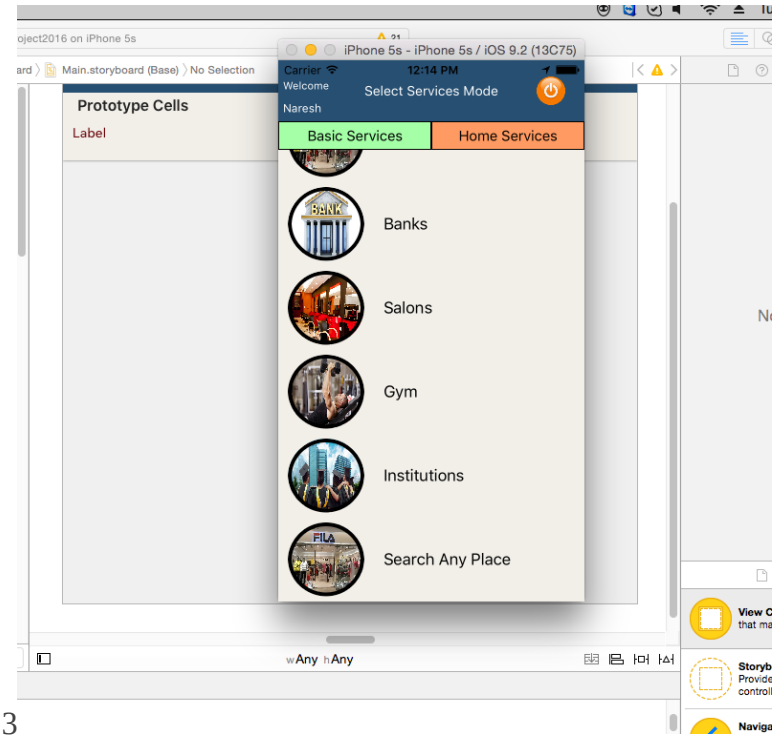



FIG 13

### 3.5.3 UI COLLECTION VIEW

The UICollectionView class manages an ordered collection of data items and presents them using customizable layouts. Collection views provide the same general function as table views except that a collection view is able to support more than just single-column layouts. Collection views support customizable layouts that can be used to implement multi-column grids, tiled layouts, circular layouts, and many more. You can even change the layout of a collection view dynamically if you want.

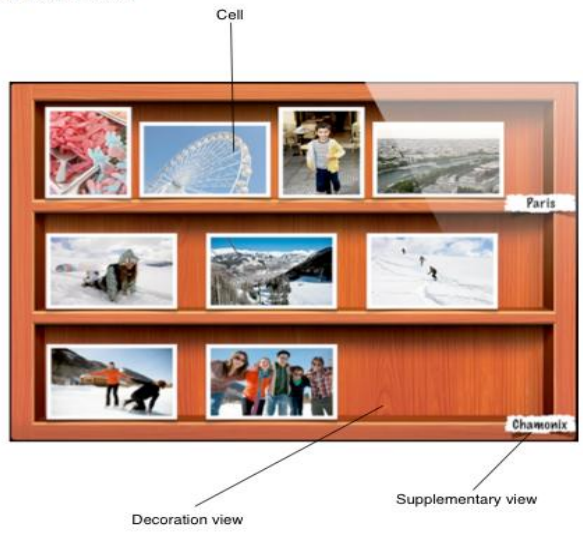
When adding a collection view to your user interface, your app's main job is to manage the data associated with that collection view. The collection view gets its data from the data source object, which is an object that conforms to the UICollectionViewDataSource protocol and is provided by your app. Data in the collection view is organized into individual items, which can then be grouped into sections for presentation. An item is the smallest unit of data you want to present. For example, in a photos app, an item might be a single image. The

collection view presents items onscreen using a cell, which is an instance of the UICollectionViewCell class that your data source configures and provides.

In addition to its cells, a collection view can present data using other types of views too. These supplementary views can be things like section headers and footers that are separate from the individual cells but still convey some sort of information. Support for supplementary views is optional and defined by the collection view's layout object, which is also responsible for defining the placement of those views.

Besides embedding it in your user interface, you use the methods of UICollectionView object to ensure that the visual presentation of items matches the order in your data source object. Thus, whenever you add, delete, or rearrange data in your collection, you use the methods of this class to insert, delete, and rearrange the corresponding cells. You also use the collection view object to manage the selected items, although for this behavior the collection view works with its associated delegate object

A collection view using the flow layout



### 3.5.3.1 IMPLEMENTATION

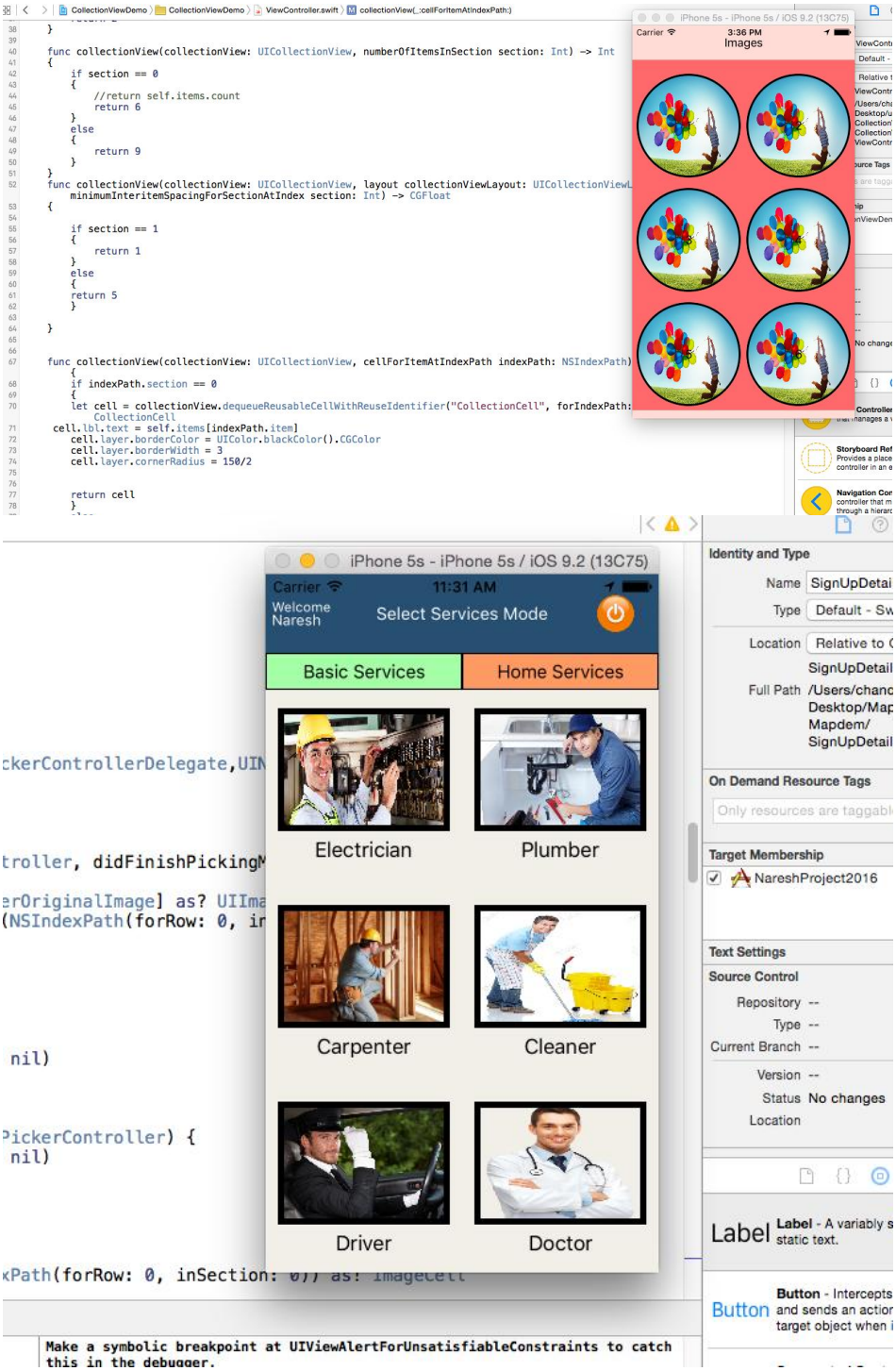



FIG 14

## 3.6 UML MODELLING

### 3.6.1 USE CASE

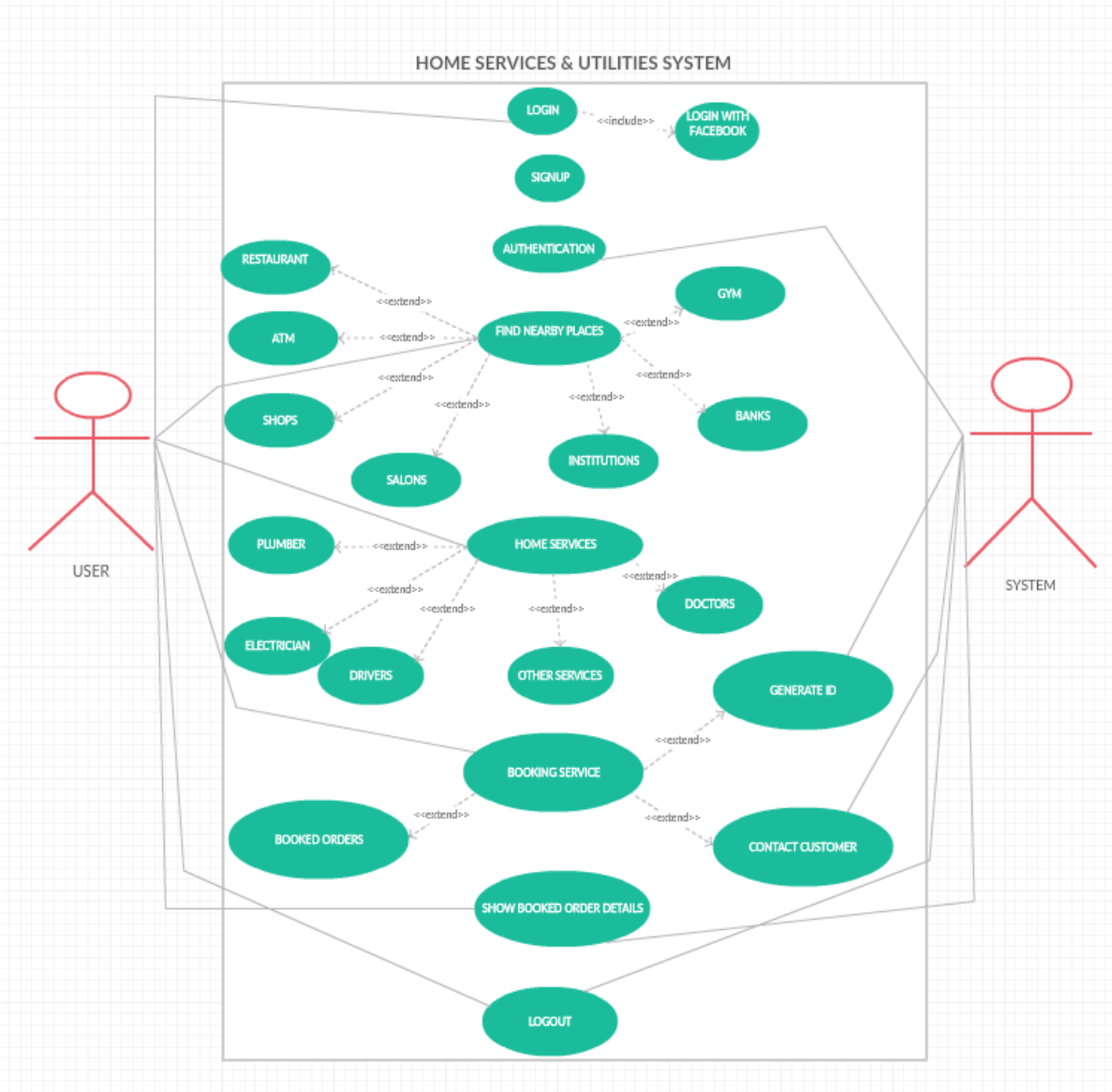



**FIG 15**

### 3.6.2 DFD

#### 3.6.2.1 0 LEVEL DFD

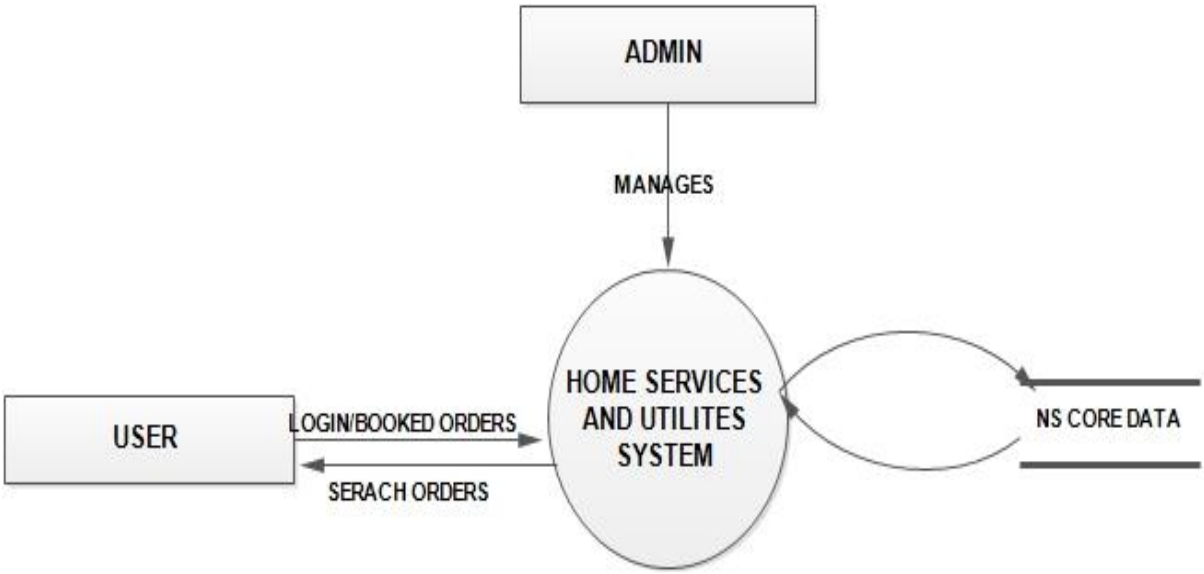



**FIG 16**

### 3.6.2.2 1 LEVEL DFD

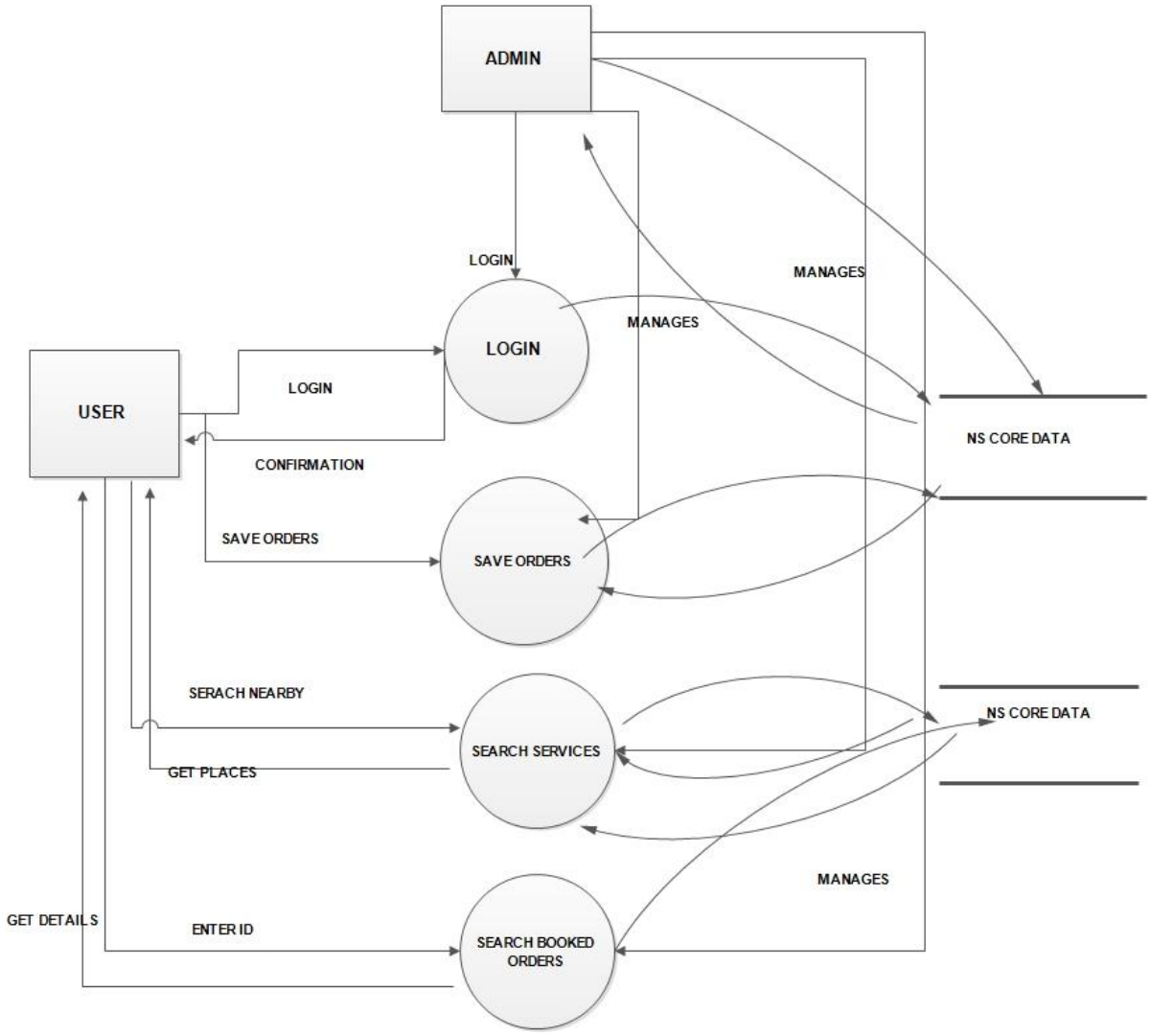



**FIG 17**

### 3.6.3 CLASS DIAGRAM

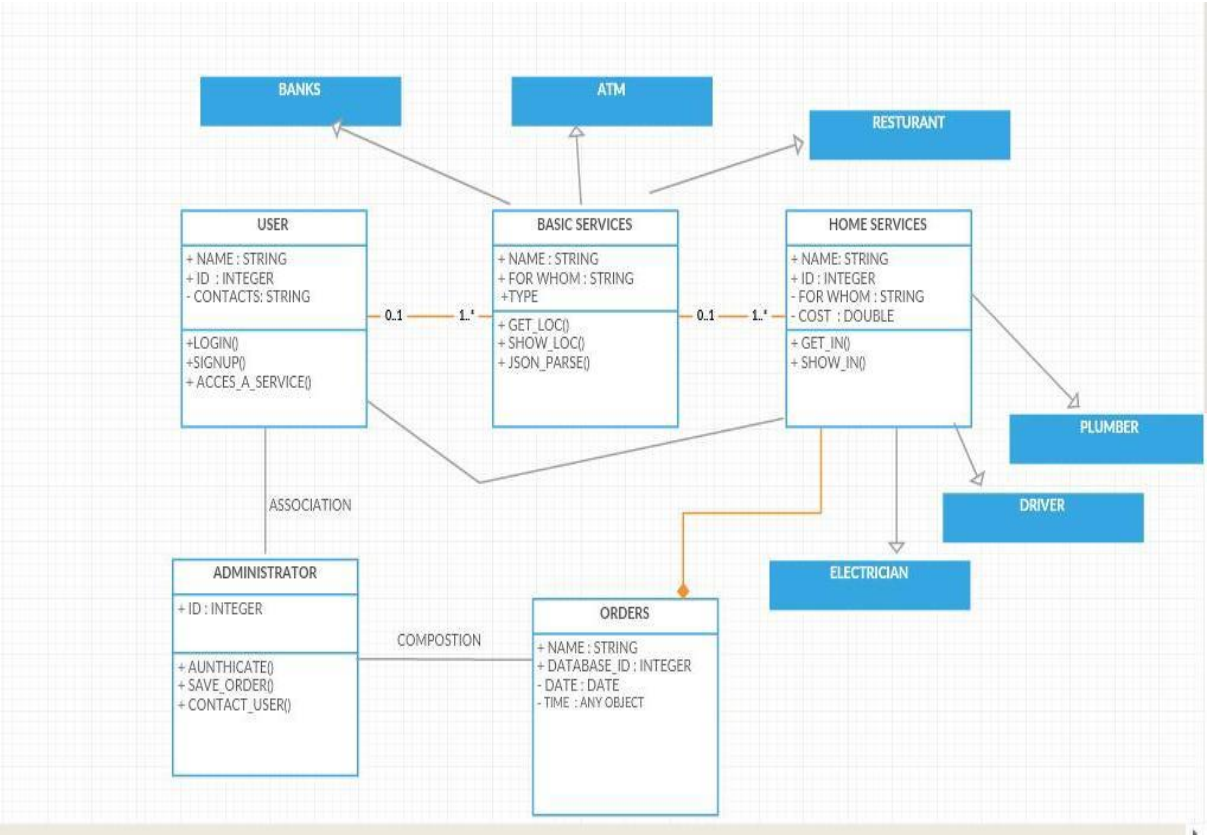



**FIG 18**

# CHAPTER 4

# PERFORMANCE ANALYSIS

## 4.1 OUTPUTS AT VARIOUS STAGES

### 1. LOGIN MODULE IS WORKING ACCORDING TO FETCHED DATABASE

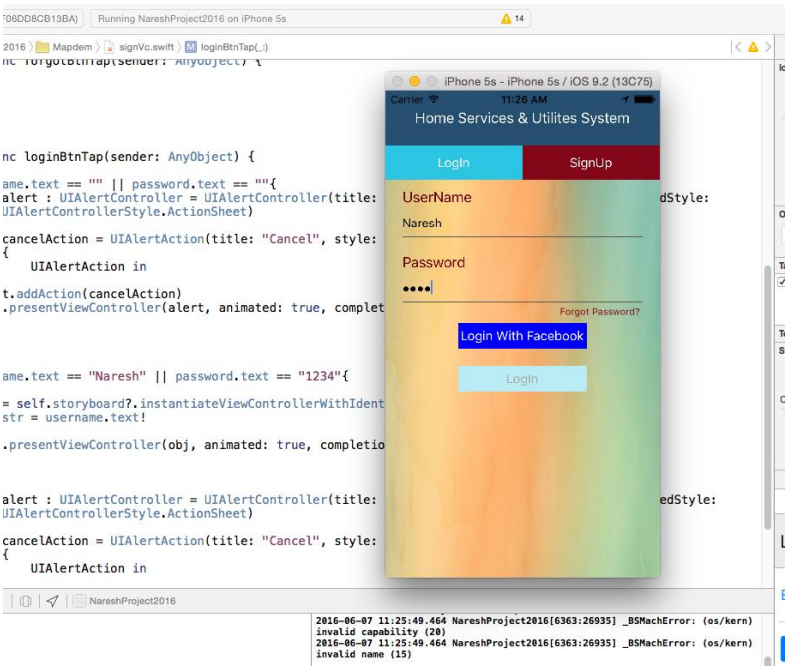



FIG 19

### 2. WHEN USER REGISTERS DATA IS SAVED ACCORDINGLY

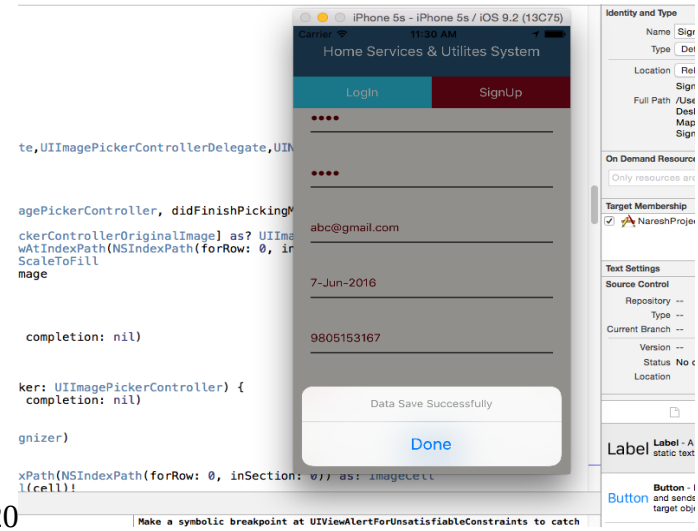



FIG 20

## 3. FIELD VALIDATIONS ARE WORKING PROPERLY

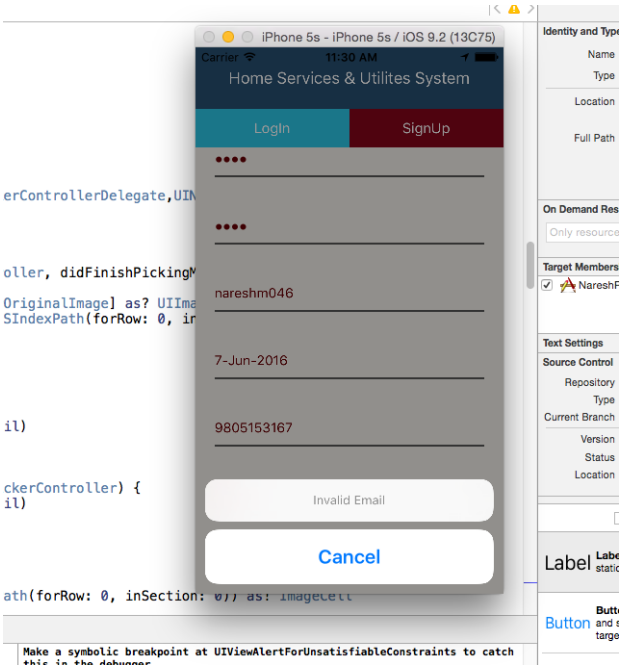



**FIG 21**

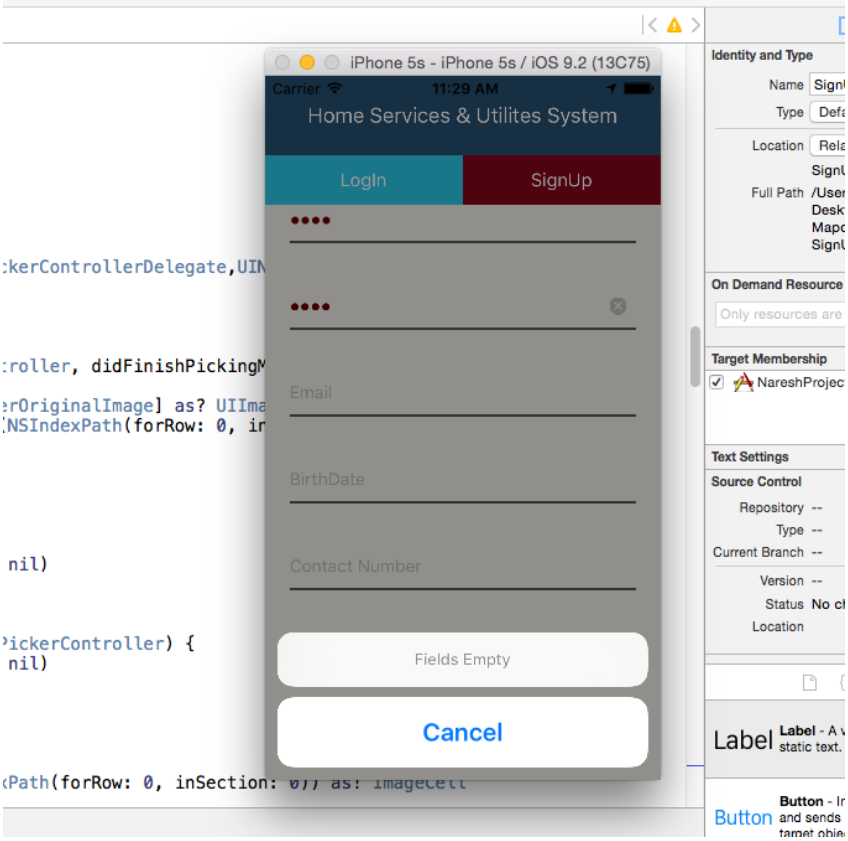



**FIG 22**

## 4. SERVICES ARE DISPLAYED AND UI WORKING PROPERLY

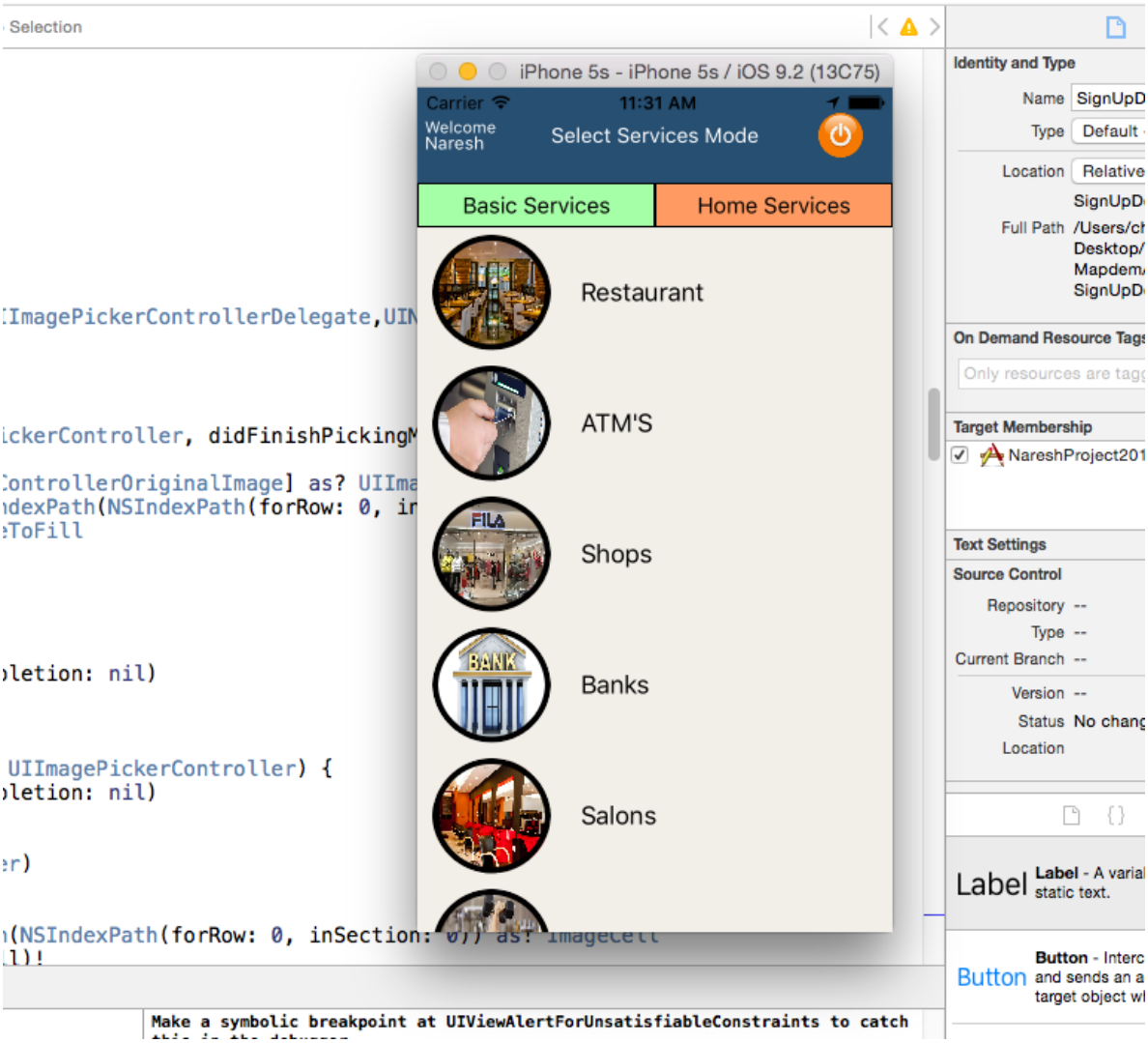



**FIG 23 BASIC SERVICES MODE**

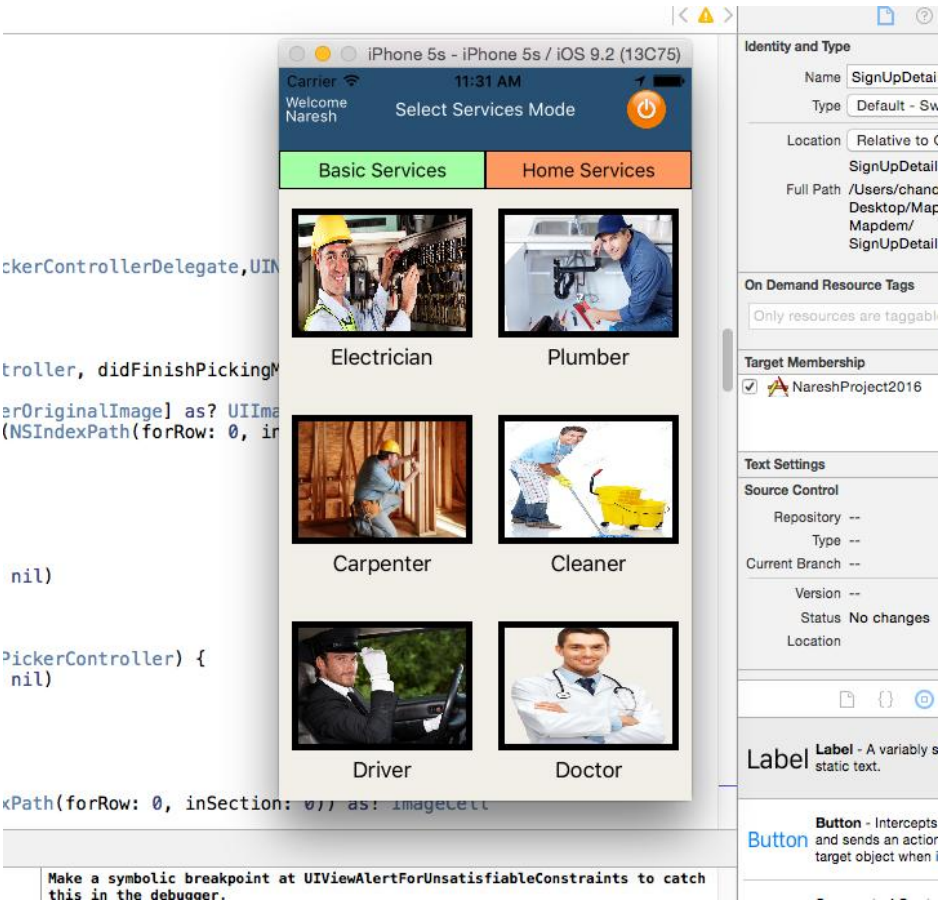



FIG 24 HOME SERVICES MODE

## 5. SERVICES AND API'S ARE WORKING PROPERLY

### NEARBY RESTAURANTS

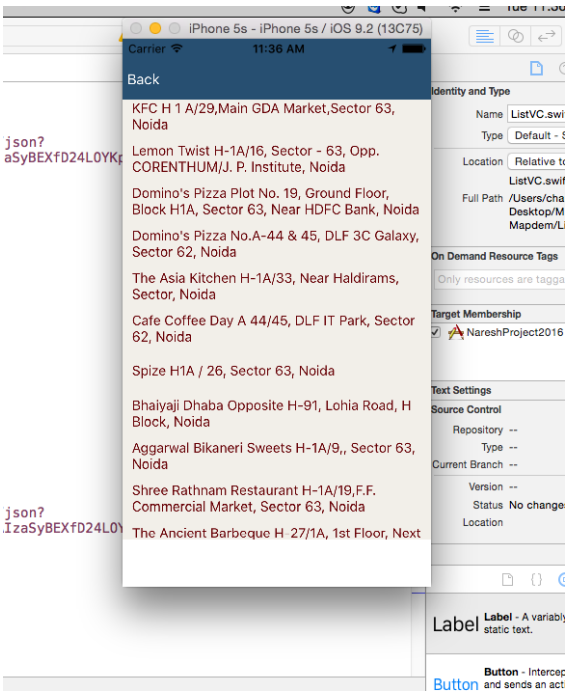



**FIG 25**

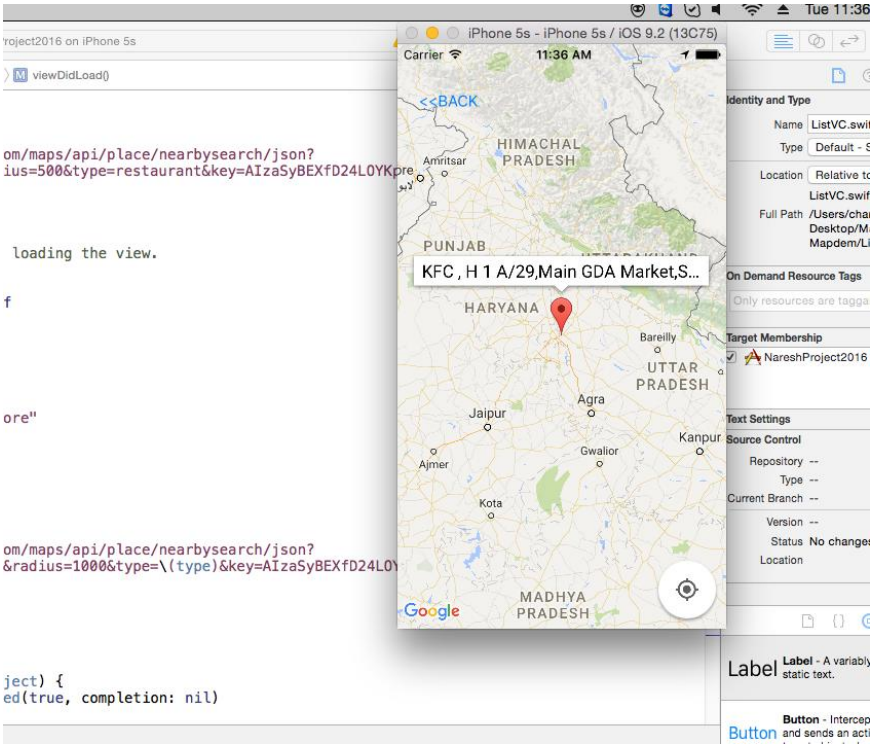



**FIG 26**

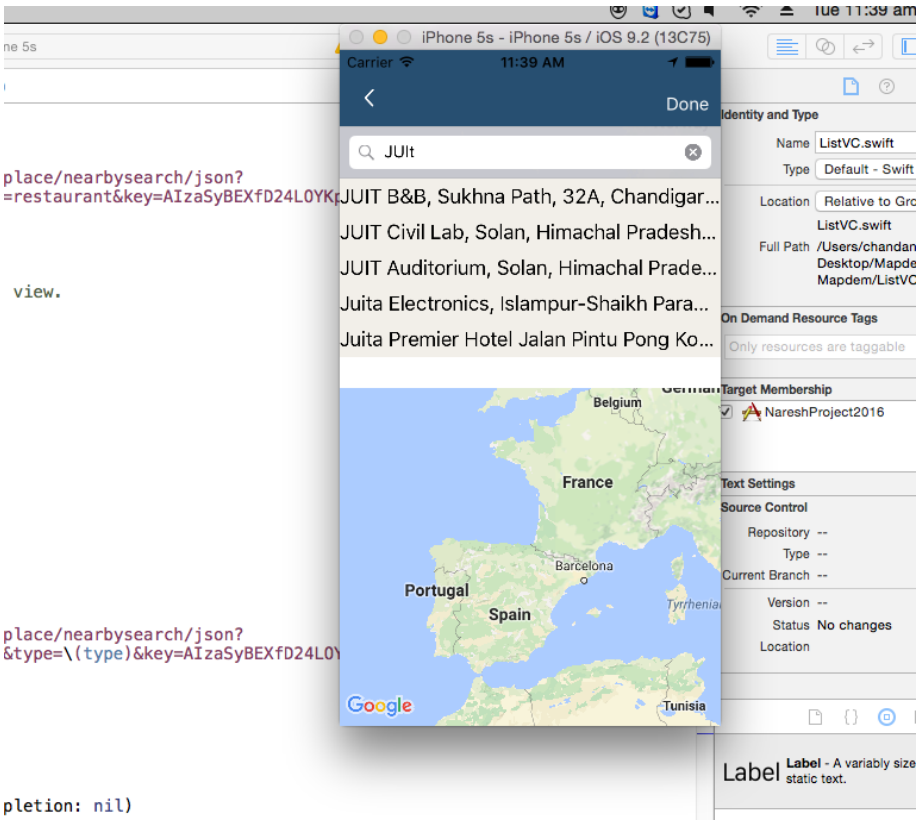



FIG 27 TO VERIFY SEARCH IS AUTOCOMPLETED AND MAPS ARE WORKING

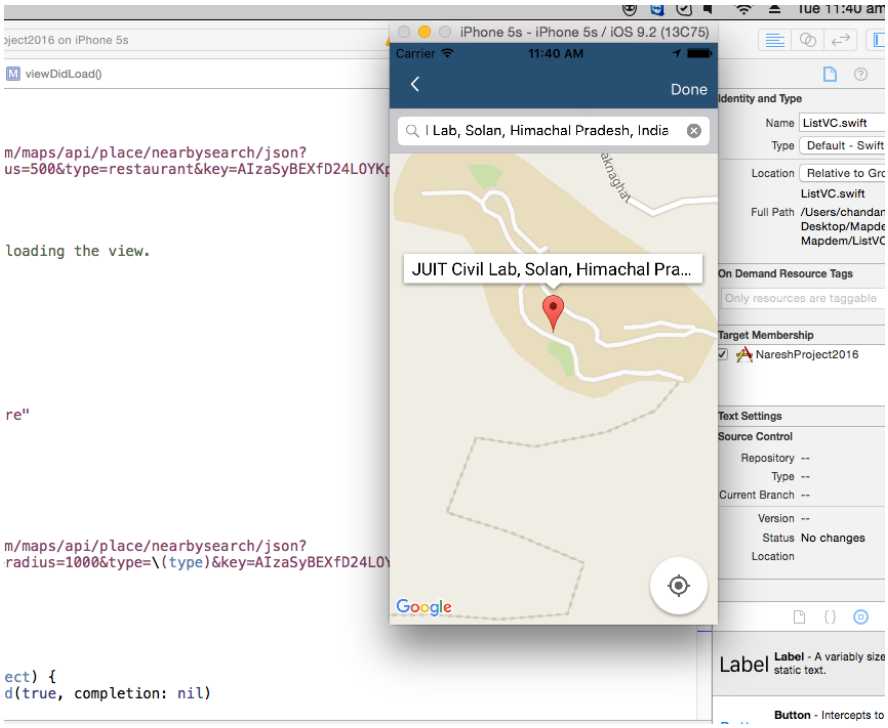
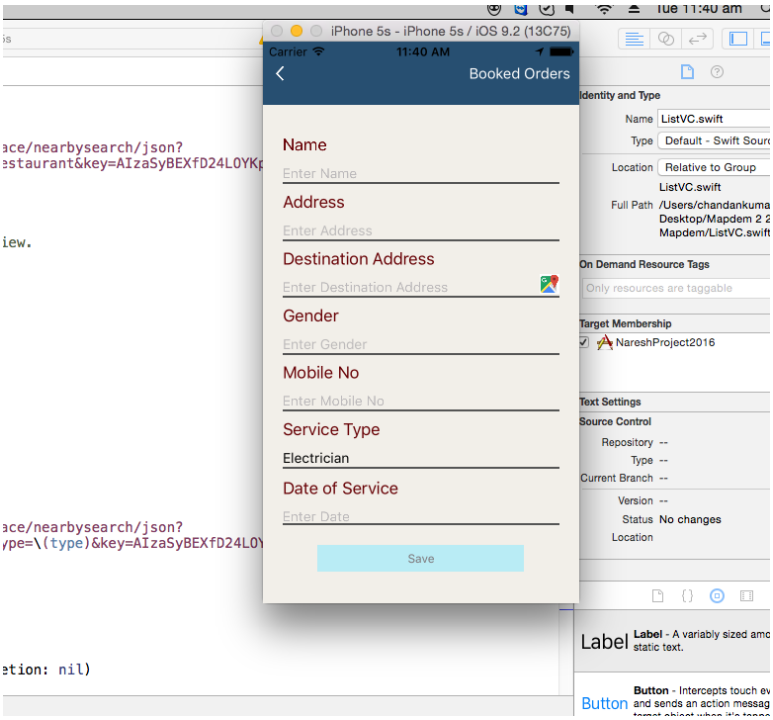



FIG 28 SERACHED LOCATION IS TRACED PROPERLY IN THE MAP

TO BOOK A HOME SERVICE BOOKING PAGE OPENS UP



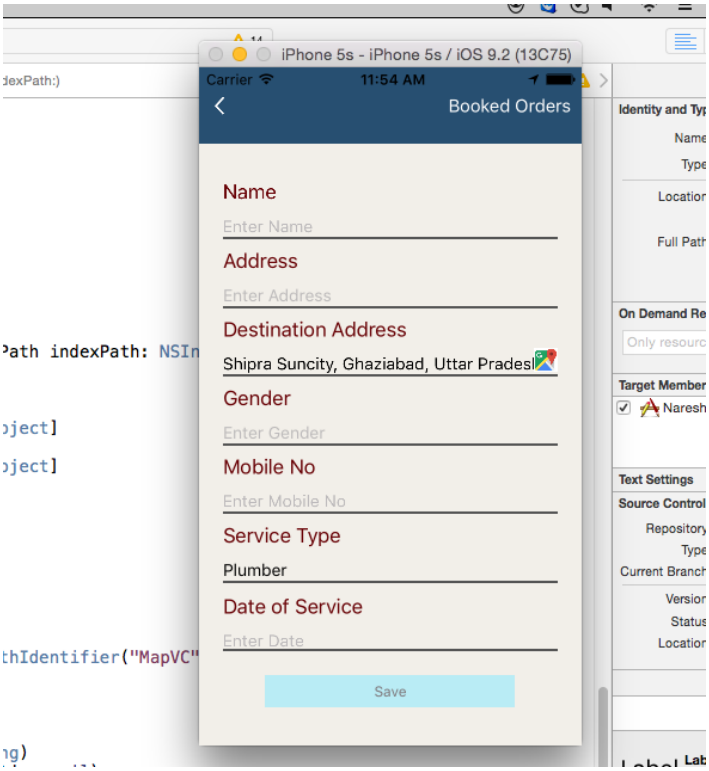
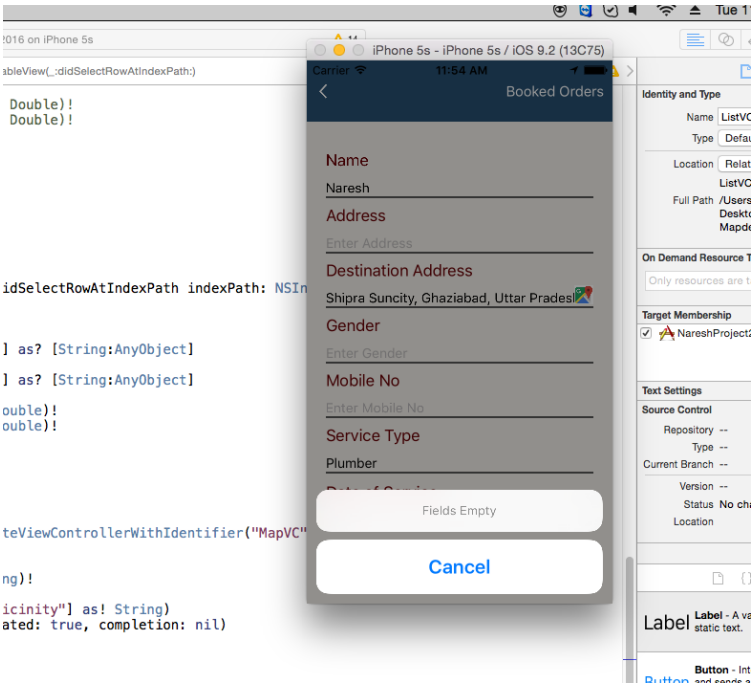




FIG 29 DESTINATION OF SERVICECAN BE PICKED FROM MAP

VALIDATIONS ALSO WORK ON BOOKING PAGE



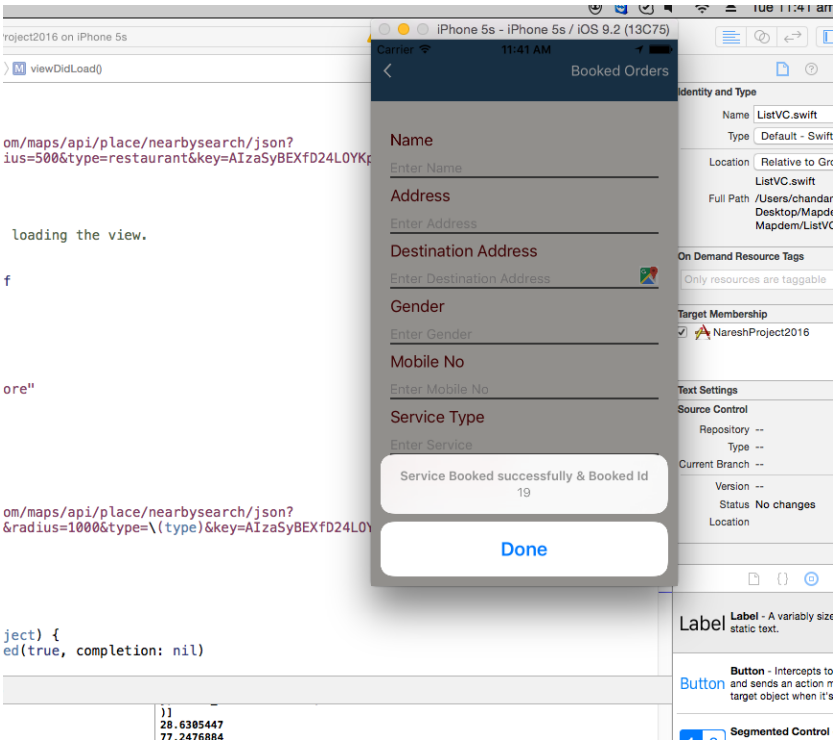




FIG 30 SERVICE IS BOOKED AND ORDER ID GENERATED

BOOKED ORDERS ARE SHOWN FROM DATABASE

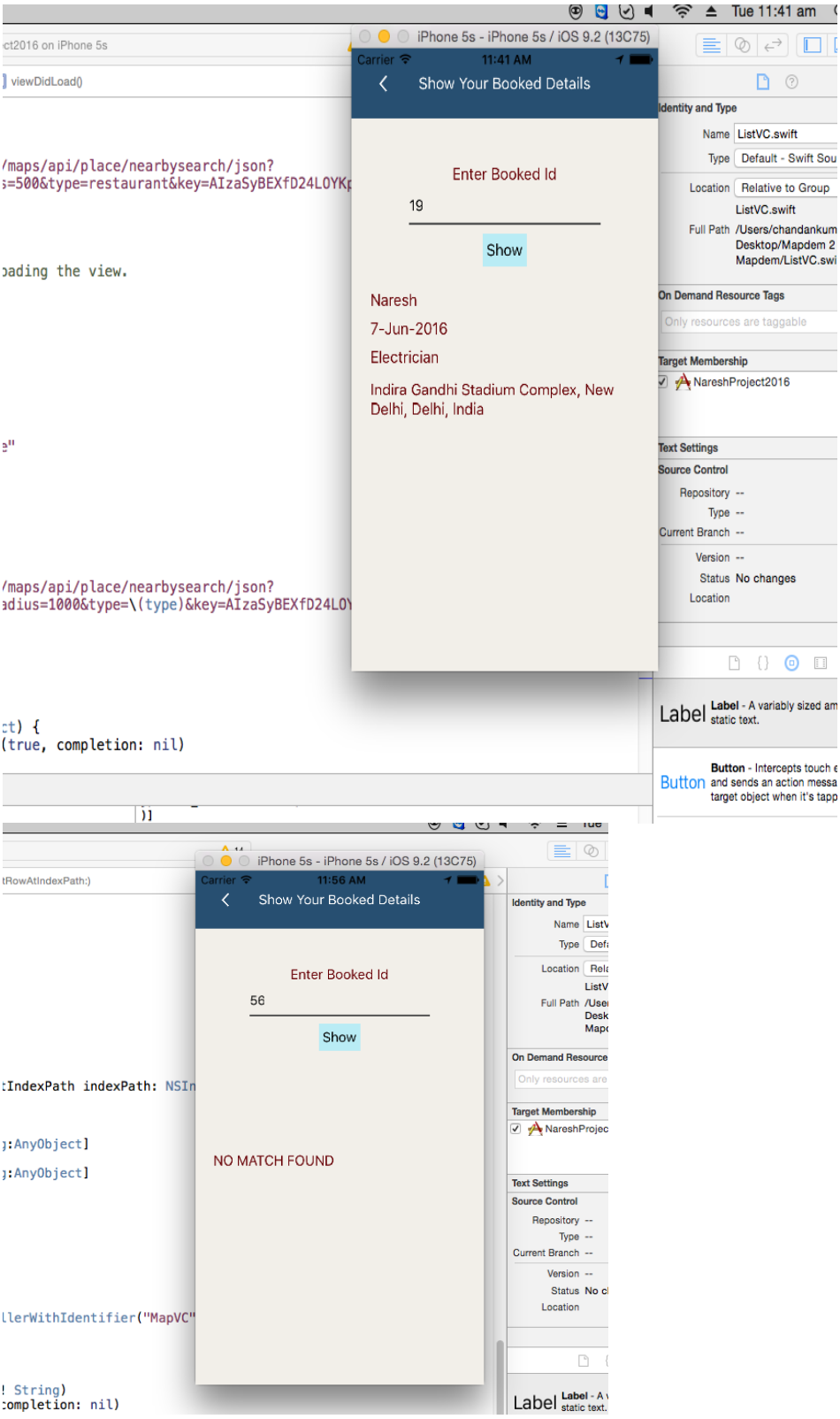



FIG 31 DETAILS ARE NOT SHOWN FOR INVALID ORDER ID

## 4.2 ANALYSIS AND TESTING

**Software testing** is an investigation conducted to provide stakeholders with information about the quality of the product or service under test.Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to the process of executing a program or application with the intent of finding software bugs (errors or other defects).

Software testing can be stated as the process of validating and verifying that a computer program/application/product:

- meets the requirements that guided its design and development,
- works as expected,
- can be implemented with the same characteristics,
- and satisfies the needs of stakeholders.

Software testing, depending on the testing method employed, can be implemented at any time in the software development process. Traditionally most of the test effort occurs after the requirements have been defined and the coding process has been completed, but in the Agile approaches most of the test effort is on-going. As such, the methodology of the test is governed by the chosen software development methodology.

**White-box Testing**

White-box testing (also known as clear box testing, glass box testing, and transparent box testing and structural testing) tests internal structures or workings of a program, as opposed to the functionality exposed to the end- user. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT).

While white-box testing can be applied at the unit, integration and system levels of the software testing process, it is usually done at the unit level. It can test paths within a unit,

paths between units during integration, and between subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it might not detect unimplemented parts of the specification or missing requirements.

Techniques used in white-box testing include:

- API testing (application programming interface) – testing of the application using public and private APIs
- Code coverage – creating tests to satisfy some criteria of code coverage (e.g., the test designer can create tests to cause all statements in the program to be executed at least once)
- Fault injection methods – intentionally introducing faults to gauge the efficacy of testing strategies
- Mutation testing methods
- **Static testing methods**

Code coverage tools can evaluate the completeness of a test suite that was created with any method, including black-box testing. This allows the software team to examine parts of a system that are rarely tested and ensures that the most important function points have been tested. Code coverage as a software metric can be reported as a percentage for:

- Function coverage, which reports on functions executed
- Statement coverage, which reports on the number of lines executed to complete the test

100% statement coverage ensures that all code paths or branches (in terms of control flow) are executed at least once. This is helpful in ensuring correct functionality, but not sufficient since the same code may process different inputs correctly or incorrectly.

**Black-box Testing**

Black-box testing treats the software as a "black box", examining functionality without any knowledge of internal implementation. The testers are only aware of what the software is

supposed to do, not how it does it. Black-box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, state transition tables, decision table testing, fuzz testing, model-based testing, use case testing, exploratory testing and specification-based testing.

Specification-based testing aims to test the functionality of software according to the applicable requirements. This level of testing usually requires thorough test cases to be provided to the tester, who then can simply verify that for a given input, the output value (or behavior), either "is" or "is not" the same as the expected value

Specified in the test case. Test cases are built around specifications and requirements, i.e., what the application is supposed to do. It uses external descriptions of the software, including specifications, requirements, and designs to derive test cases. These tests can be functional or non-functional, though usually functional.

Specification-based testing may be necessary to assure correct functionality, but it is insufficient to guard against complex or high-risk situations.

One advantage of the black box technique is that no programming knowledge is required. Whatever biases the programmers may have had, the tester likely has a different set and may emphasize different areas of functionality. On the other hand, black-box testing has been said to be "like a walk in a dark labyrinth without a flashlight. Because they do not examine the source code, there are situations when a tester writes many test cases to check something that could have been tested by only one test case, or leaves some parts of the program untested.

This method of test can be applied to all levels of software testing: unit, integration, system and acceptance. It typically comprises most if not all testing at higher levels, but can also dominate unit testing as well

### 4.3 PARAMETERS TO TEST

1. LOGIN DETAILS ARE FETCHED FROM DATABASE AND VERIFIED
2. WHEN USER SIGN UPS DATA IS SAVED
3. WHEN USER LOGS IN VALIDATIONS WORK TO PREVENT UNAUTHORIZED ACCESS
4. WHEN USER SIGNN UPS AND LOGS IN FIELD VALIDATIONS WORK FOR PROPER DATA ENTRY
5. UI ELEMENTS WORK PROPERLY
6. SERVICES DISPLAYED PROPERLY
7. LOCATION IS SIMULATED PROPERLY
8. WEB SERVICES AND API'S WORK PROPERLY
9. SEARCH IS AUTOCOMPLETED AND MAP MARKERS SHOWS PROPER PLACE
10. NEARBY PLACES ARE ACCURATE AND IN RADIUS
11. WHEN USER WANT TO BOOK A SERVICE BOOKING PAGE OPENS
12. DATA IS VALIDATED AND SAVED PROPERLY FOR BOOKING DETAILS
13. ORDER DETAILS ARE SHOWN ACCORDING TO ORDER ID
14. ORDER DETAILS ARE NOT SHOWN FOR INVALID ID
15. LOGOUT BUTTONS LOGS YOU OUT OF THE APPLICATION
16. UI CONSTRAINTS WORK PROPERLY
17. APPLICATION IS CHECKED IN VARIOUS SCREEN SIZES OF IPHONES

# CHAPTER 5

# CONCLUSION

## 5.1 USE AND SCOPE

With high income nuclear families in metropolitans becoming a norm the scope of maintenance services in India is high. This can be easily observed by entry of numerous players like easyfix, bromberg and handyman entering the market.

One of the challenges in this domain would be getting a pool of expert resources who can undertake a variety of tasks especially quite a number of these services would be done by semi skilled less education handyman.

One of the effective strategies to successfully market these services is to do an app based ordering system where your customers can download an app and order the services straight from there, instead of using the cumbersome call center based ordering system

Demand in Home services sector is rising yet there is huge potential due to large untapped market. While urban households are getting acquainted to the idea, the services have not yet reached tier-2 cities. The biggest challenge in this sector is to get professionals with right attitude onboard and providing uniform and reasonable pricing to the customers while maintaining quality.

There is no reason to break your head if you find a leaking tap in your kitchen in the morning or your microwave not functioning properly, as, you will find the services of professionals coming to your help at beckons call. Calling any Tom, Dick and harry may not solve your problems, as you will find a disorganized market where you can hardly rely on

unprofessional service providers from whom expecting service quality, punctuality and safety would be not that much a prudent decision.

The winners will be defined by quality of service, ease of use of app, credibile brand building, training of manpower and ability to scale local geographical areas.

## 5.2 FUTURE INCREMENTS

- The services provided by the application can be increased , The application can be made to be used for complete customer support and assistance system.

- It can even be used as a place finder, search for locations and even book a taxi in its incremented and updated versions.

- There has been a boom in health care services so medical and healthcare services can also be included so that the application qualifies to set a proper example of complete services and support application.

- The mobile development field has a wide scope in future and with increasing dependency and growth of mobile applications , the application can do well in both Android and IOS platforms

## REFERENCES


1. [1]developer.apple.com.[Online]. Available:https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/LocationAwarenessPG/CoreLocation/CoreLocation.html
. [Accessed:2016]

2.[2] iphonefaq.com 2016.[Online]. Available: http://www.iphonefaq.org/archives/974149
. [Accessed:2016]

3.[3] developer.google.com 2016.[Online]. Available: https://developers.google.com/places/
. [Accessed:2016]

4.[4] stackoverflow.com 2016.[Online]. Available: http://stackoverflow.com/questions/16647871/embed-collection-view-in-custom-table-view-cell
. [Accessed:2016]

5.[5] developer.google.com 2016.[Online]. Available: *https://developers.google.com/maps/documentation/ios-sdk/start#step 2 get cocoapods*
. [Accessed:2016]

6.[6] raywenderlich.com 2016.[Online]. Available: *https://www.raywenderlich.com/*
. [Accessed:2016]


END OF DOCUMENT