**Health Care**

**All controllers are based on table views.**

**Each cell is customized.**

**Most of cells are used in all project.**

**This method is described at https://github.com/imodeveloper/ImoDynamicTableView**

**The reason of this method is to make dynamical UI programing.**

**Each cell has it's source which feed it with information.**

**When You create a new cell You have to show to source what cell will be completed with.**

**Example :**

#import <ImoDynamicTableView/ImoDynamicDefaultCell.h>

@interface SpaceCellSource : IDDCellSource

@property (nonatomic,retain) UIColor \* backgroundColor;

@end

@interface SpaceCell : ImoDynamicDefaultCellExtended

- (void)setUpWithSource:(SpaceCellSource\*)source;

@end

**each cell interface shall contain (**setUpWithSource:**) method**

**in method** setUpWithSource: **we will have**

- (void)setUpWithSource:(SpaceCellSource\*)source

{

self.backgroundColor = source.backgroundColor;

}

(1.1)

**the property  *backgroundColor* will be passed to cell property**

**these where crumbs**

**we have the login controller**

**there is not to say , I customized text fields to have one of them with image and customized placeholder color.**

**Macintosh HD:Users:mc309:Desktop:Screen Shot 2017-01-05 at 02.41.35.png**

if (\_textField.placeholder.length) {

\_textField.attributedPlaceholder = [[NSAttributedString alloc] initWithString:\_textField.placeholder attributes:@{NSForegroundColorAttributeName: [AppUtils placeholdersColor]}];

}

(1.2)

it delegates to its view controller

Register phase has 5 steps:

1st step Particular or CNA from where apear ClientType

typedef enum {

\_individual,

\_profesional,

\_unknown\_type

} ClientType;

messages where written like

{

BlueTitleAndSubtitleCellSource \*cellSource = [BlueTitleAndSubtitleCellSource new];

cellSource.title = @"Registration: Step 1";

cellSource.subtitle = @"Choose your account type";

[section addObject:cellSource];

}

(1.3)

**the price also**

{

PriceItemCellSource \*cellSource = [PriceItemCellSource new];

cellSource.price = @"$9.99/month";

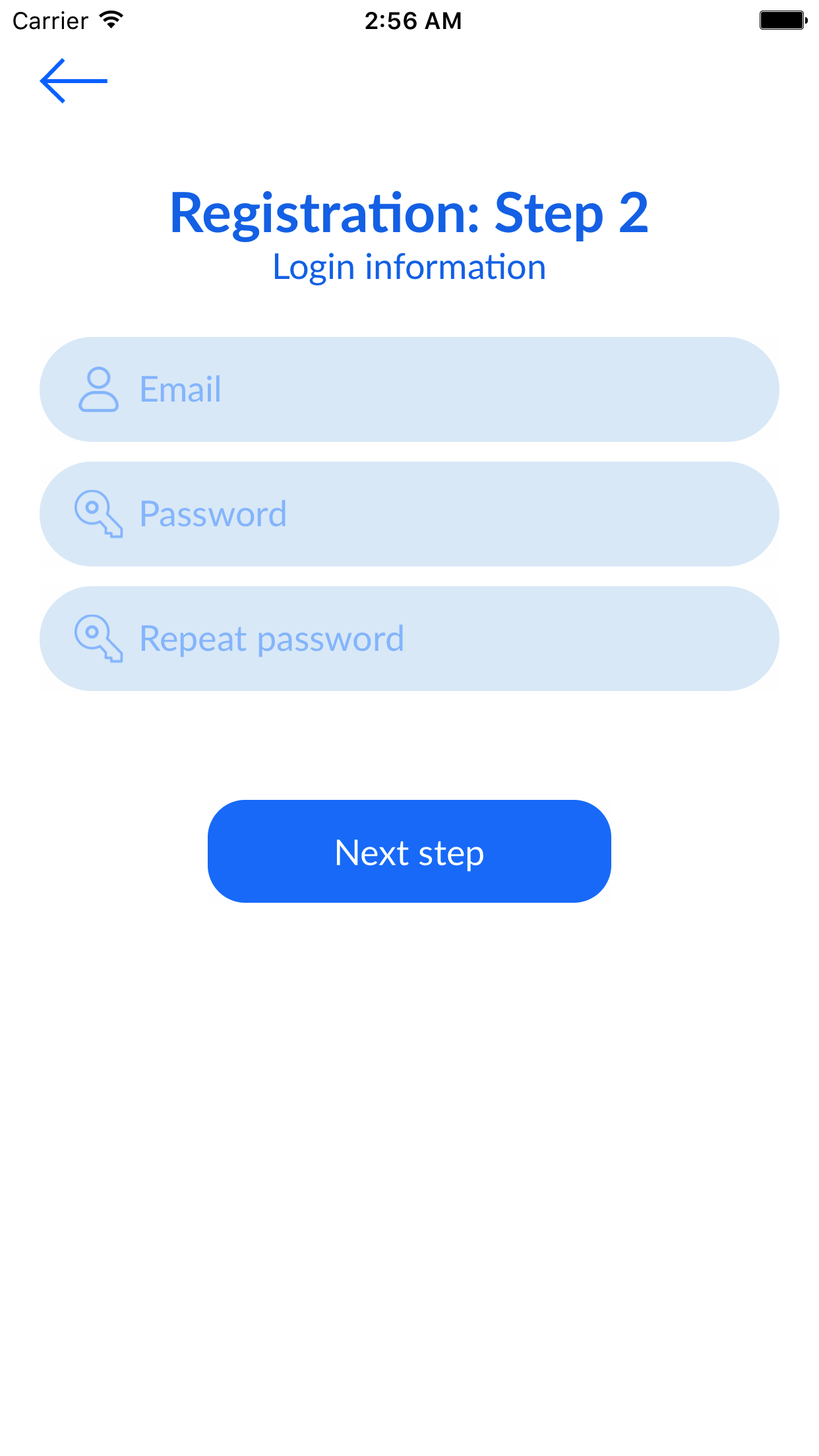
cellSource.staticHeightForCell = 40;

[section addObject:cellSource];

}

****

**each source feeds it's cell**

****

**text fields are** (1.2) point and title refers to (1.3). But the blue cornered buttons have their place in project

**Macintosh HD:Users:mc309:Desktop:Screen Shot 2017-01-05 at 03.00.31.png**

**are customized to optimize all the project**

**all errors are validated in**

+(void)alertUserForError:(NSError\*)error

{

UIApplication \* application = [UIApplication sharedApplication];

if ( application.applicationState == UIApplicationStateInactive

|| application.applicationState == UIApplicationStateBackground ) {

return;

}

NSLog(@"\n\n\n%@\n\n\n",error);

if ([error.domain isEqualToString:appErrorDomain]) {

NSString \*messageString = error.userInfo[@"message"];

UIAlertController \*alert = [UIAlertController alertControllerWithTitle:appErrorDomain message:messageString preferredStyle:UIAlertControllerStyleAlert];

UIAlertAction\* okButton = [UIAlertAction

actionWithTitle:@"OK"

style:UIAlertActionStyleDefault

handler:^(UIAlertAction \* action) {

}];

[alert addAction:okButton];

[[AppUtils topViewController] presentViewController:alert animated:YES completion:nil];

}

else

{

NSString \*messageString = error.localizedDescription;

if ([[HTTPClient sharedHTTPClient] connected] == NO || [messageString isEqualToString:@"The network connection was lost."]) {

if ([AppUtils topViewController] == [HTTPClient sharedHTTPClient].requestViewController) {

[[AppUtils topViewController] presentViewController:[CheckInternetConnectionViewController new] animated:YES completion:^{

}];

}

return;

}

UIAlertController \*alert = [UIAlertController alertControllerWithTitle:@"Server error!" message:messageString preferredStyle:UIAlertControllerStyleAlert];

UIAlertAction\* okButton = [UIAlertAction

actionWithTitle:@"OK"

style:UIAlertActionStyleDefault

handler:^(UIAlertAction \* action) {

}];

[alert addAction:okButton];

[[AppUtils topViewController] presentViewController:alert animated:YES completion:nil];

}

}

**like an example in RegistrationStep2ViewController.m**

-(IBAction)nextStep:(id)sender

{

NSString \*messsageString = @"";

if (\_password.length < 6) {

messsageString = @"Password must have at least 6 characters";

}

else if ([AppUtils isEmailValid:self.user.email] == NO) {

messsageString = @"Invalid email adddress";

}

else if ([\_password isEqualToString: \_repeatPassword] == NO) {

messsageString = @"Password fields does not match";

}

if (messsageString.length) {

NSError \*error = [NSError errorWithDomain:appErrorDomain code:0 userInfo:@{@"message" : messsageString}];

[DataLoader alertUserForError:error];

return;

}

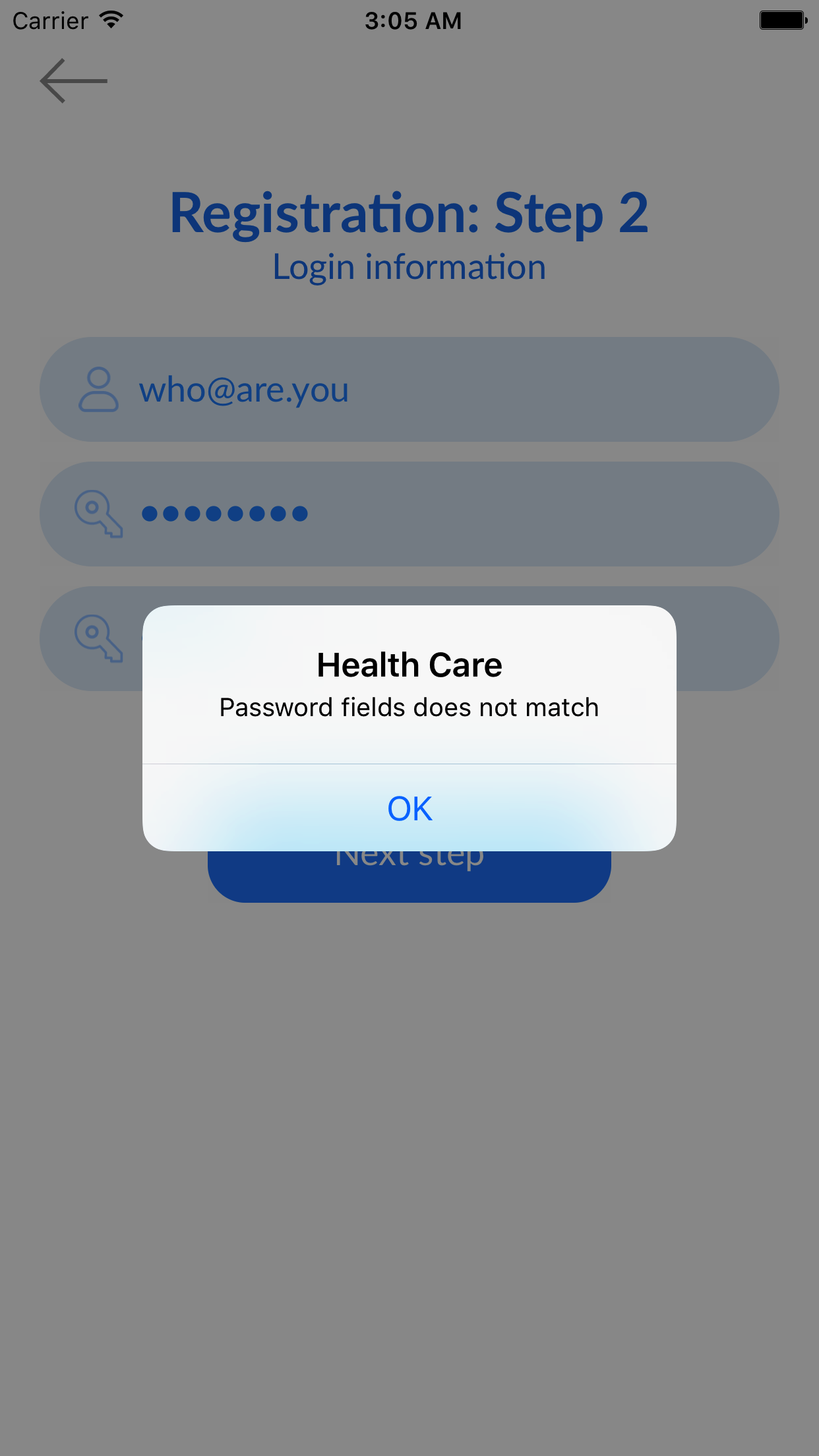
[DataLoader checkEmail:self.user.email success:^(id responseObject) {

[self performSegueWithIdentifier:segue\_showRegistrationStep\_3 sender:nil];

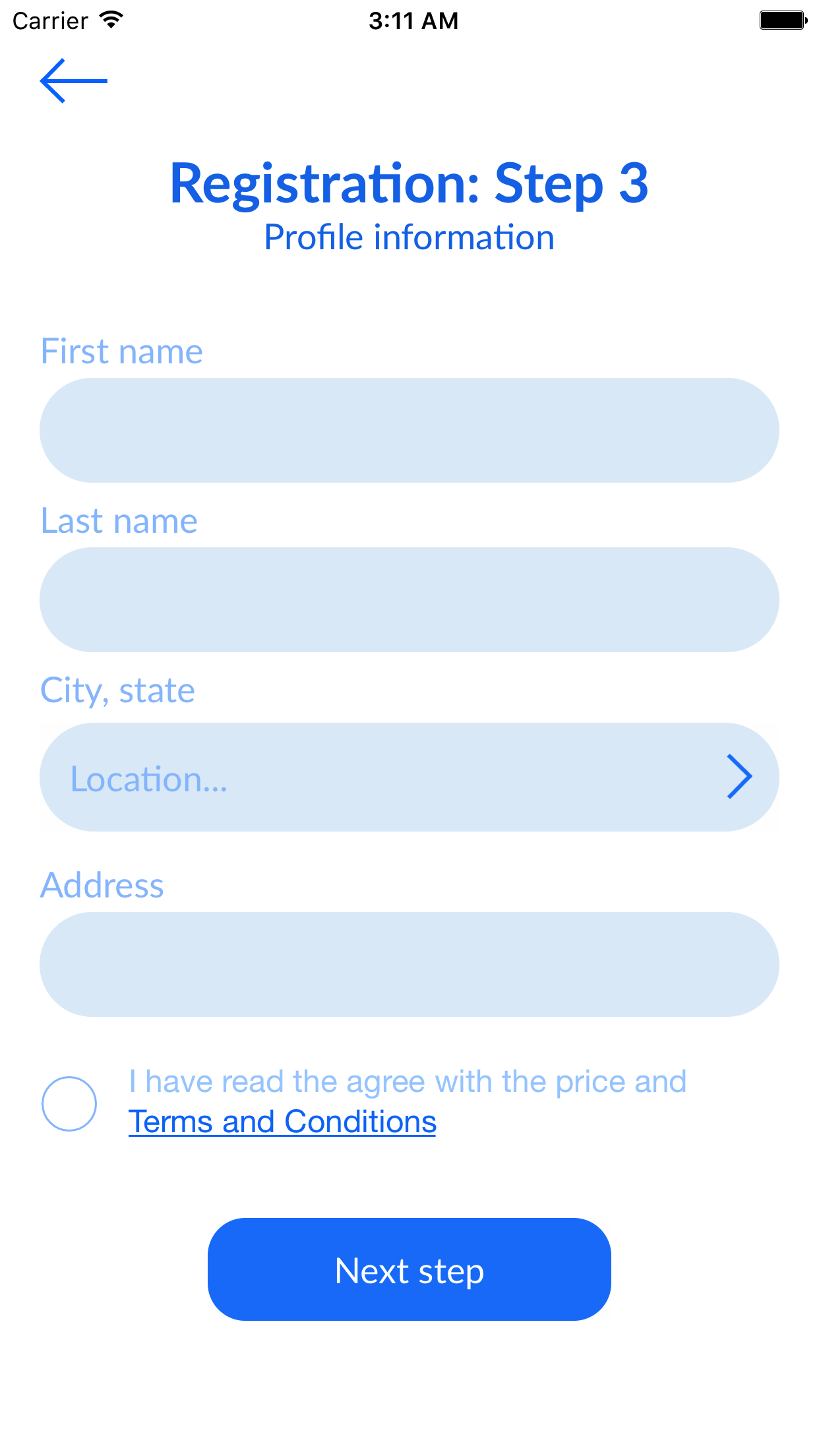
} fail:^(NSError \*error) {

}];

}

****

(1.4)



Like I said in (1.2) each text field delegates in its view controller then it is verified by tag

- (void)textFieldDidChange:(UITextField \*)textField

{

//ewe

NSInteger tag = textField.tag;

switch (tag) {

case 0:

self.user.email = textField.text;

break;

case 1:

\_password = textField.text;

break;

case 2:

\_repeatPassword = textField.text;

break;

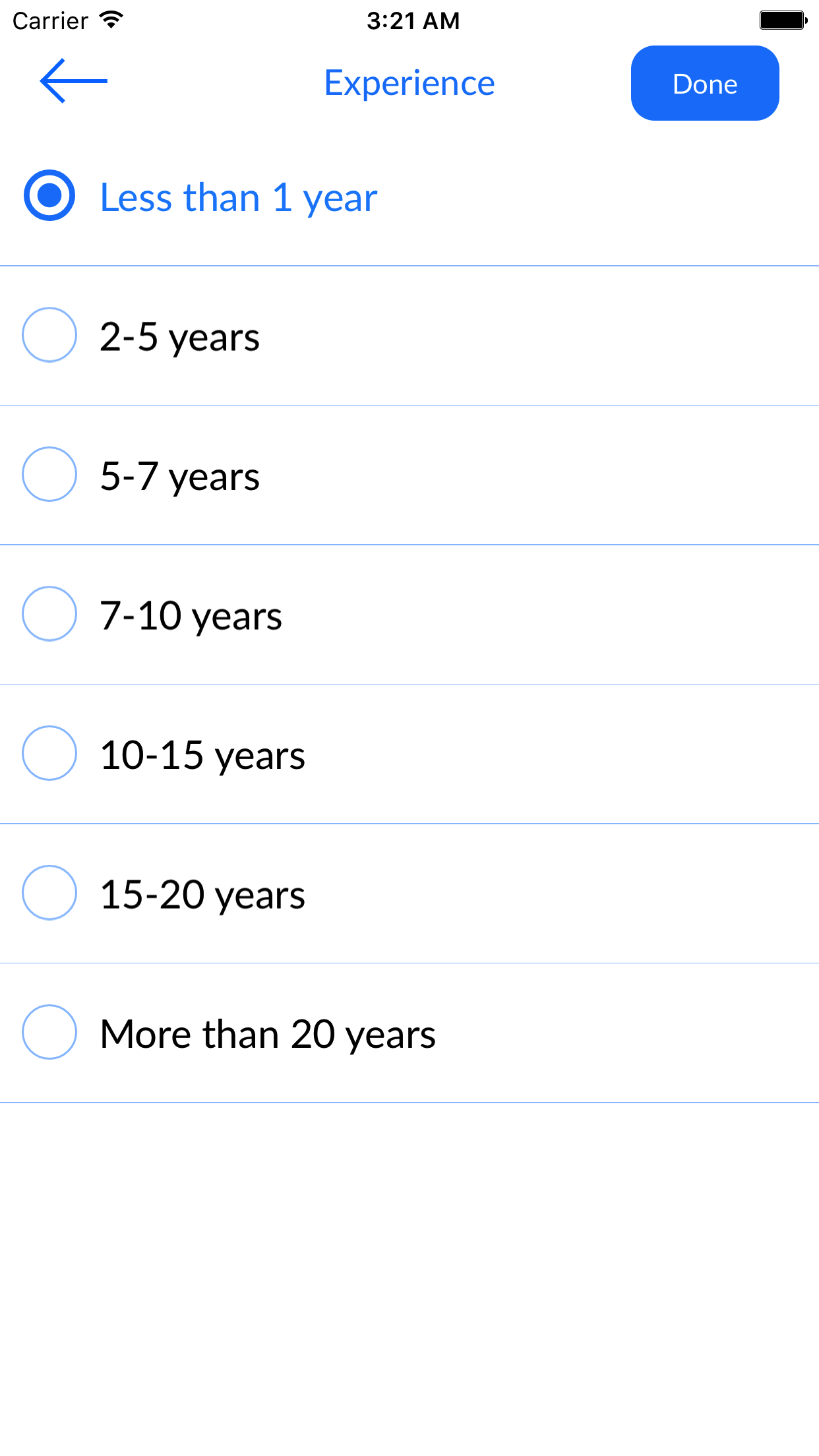
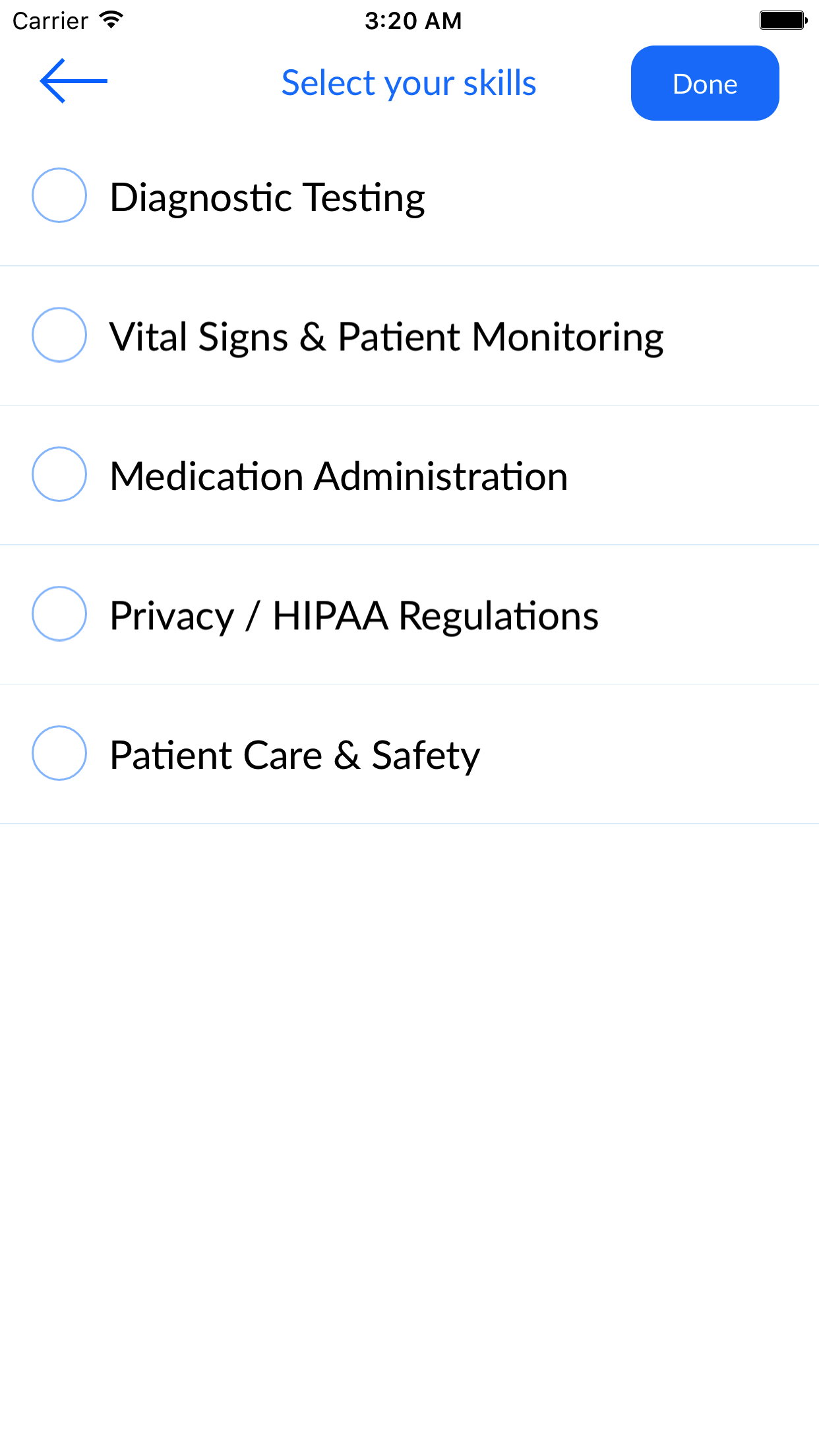
default:

break;

}

}

but we have another cell which pass us to location or another view controller like Skils or Experience.



these need beter attention because You have to set for about done user each of this proprieties

RadioButtonCellSource \*cellSource = [RadioButtonCellSource new];

cellSource.title = experienceArray[i];

cellSource.isSelected = [self.user.experience integerValue] == i;

cellSource.selector = @selector(itemSelected:);

cellSource.target = self;

cellSource.itemTag = i;

[section addObject:cellSource];

}

but in RadioButtonCell hapens

NSString \* imageName = [@"radioButton" stringByAppendingFormat:@"%d",source.isSelected];

\_radioImage.image = [UIImage imageNamed:imageName];

the same in location You press select and view controller take user location

another thing we have to select just 2 choices



in DoubleRadioButtonCell we have

[\_leftItemButton addTarget:source.target action:source.selector forControlEvents:UIControlEventTouchUpInside];

[\_rightItemButton addTarget:source.target action:source.selector forControlEvents:UIControlEventTouchUpInside]

int choice = source.selectedItem == 0;

NSString \* imageName = [@"radioButton" stringByAppendingFormat:@"%d",choice];

\_leftItemRadioImage.image = [UIImage imageNamed:imageName];

imageName = [@"radioButton" stringByAppendingFormat:@"%d",!choice];

\_rightItemRadioImage.image = [UIImage imageNamed:imageName];

all selectors are passed from viewcontroller

DoubleRadioButtonCellSource \*cellSource = [DoubleRadioButtonCellSource new];

cellSource.selectedItem = self.user.available\_time;

cellSource.selector = @selector(selectAvailability:);

cellSource.leftItemName = @"Full time";

cellSource.rightItemName = @"Part time";

cellSource.target = self;

[section addObject:cellSource];

and view controller catches them

-(IBAction)selectAvailability:(UIButton\*)sender

{

self.user.available\_time = (AvailableTime)sender.tag;

[self buildInterface];

}

Users are registered as CNA and profesional. So we have the list of CNA. It is created for Individual users Is the same table view but with another cell sources. I is built in :

Macintosh HD:Users:mc309:Desktop:Screen Shot 2017-01-09 at 09.57.49.png

There is used pagination to load by al limited number of items.

- (void)refresh:(UIRefreshControl \*)refreshControl {

NSInteger fromPage = 0;

if (\_items.count != 0) {

User \* user = \_items[0];

if (user.fromPage - 1 == -1) {

[self loadItemsForPage:0 position:\_top];

return;

}

fromPage = user.fromPage - 1;

}

[self loadItemsForPage:fromPage position:\_top];

}

- (void)refreshBottom:(UIRefreshControl \*)refreshControl {

if (\_items.count != 0) {

User \* user = \_items[\_items.count - 1];

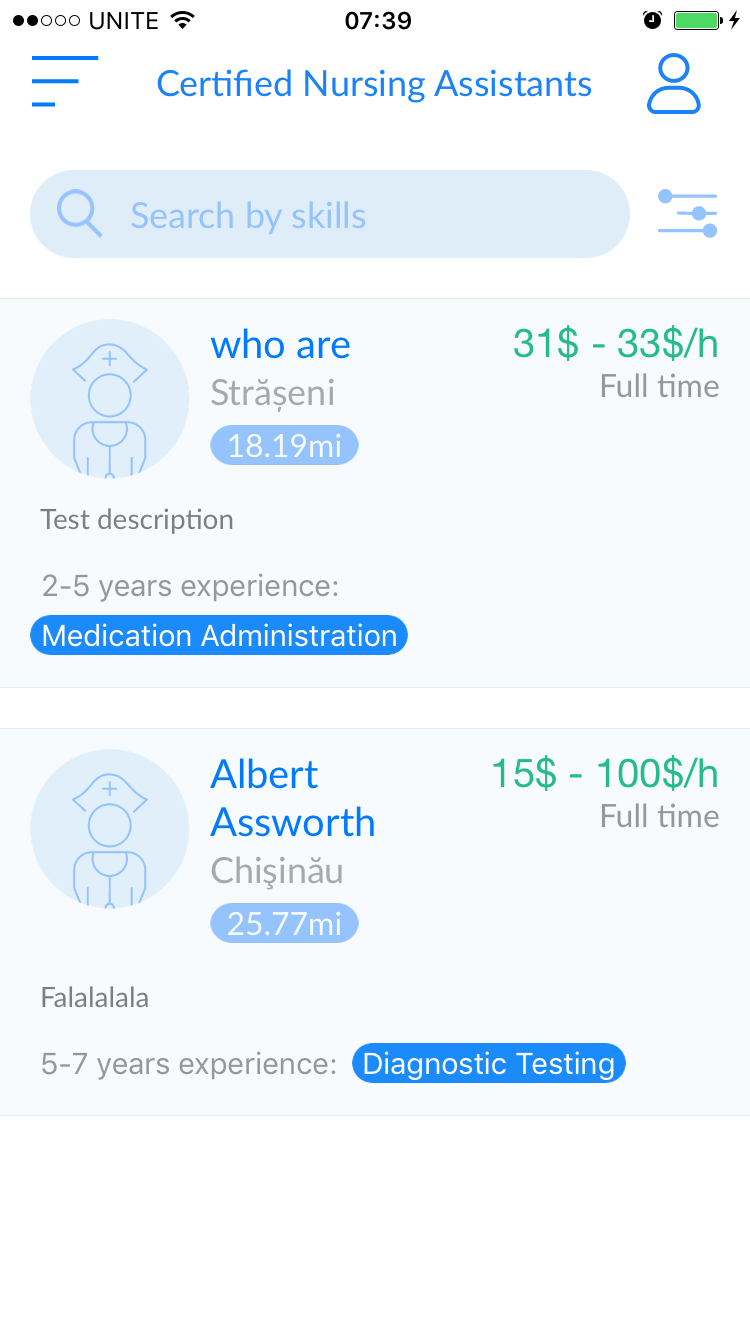
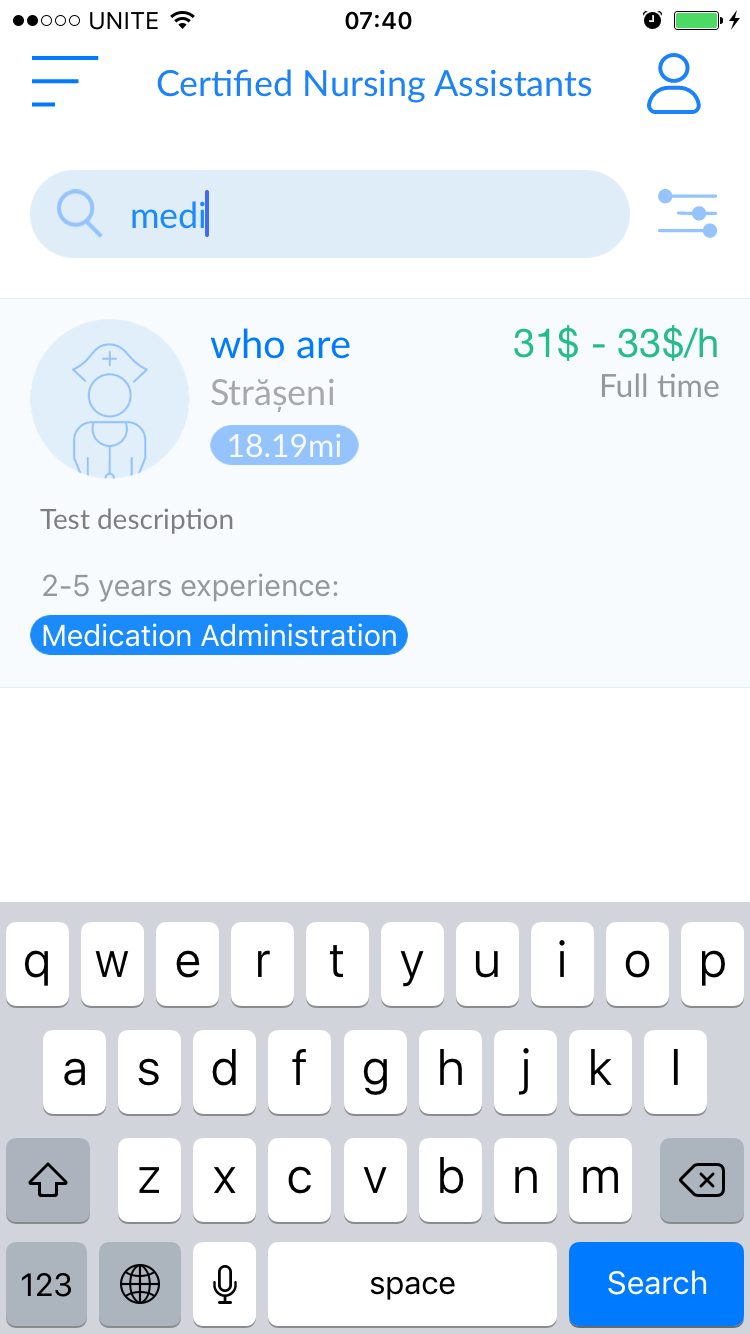
NSInteger fromPage = user.fromPage + 1;

[self loadItemsForPage:fromPage position:\_bottom];

};

}

each item is marked with its page it was received and when we ask another page we increment or decrement this marker. Also it contains the search option which finds Assistants by their skils. Is the same request but server verifies if the search string is empty or not.

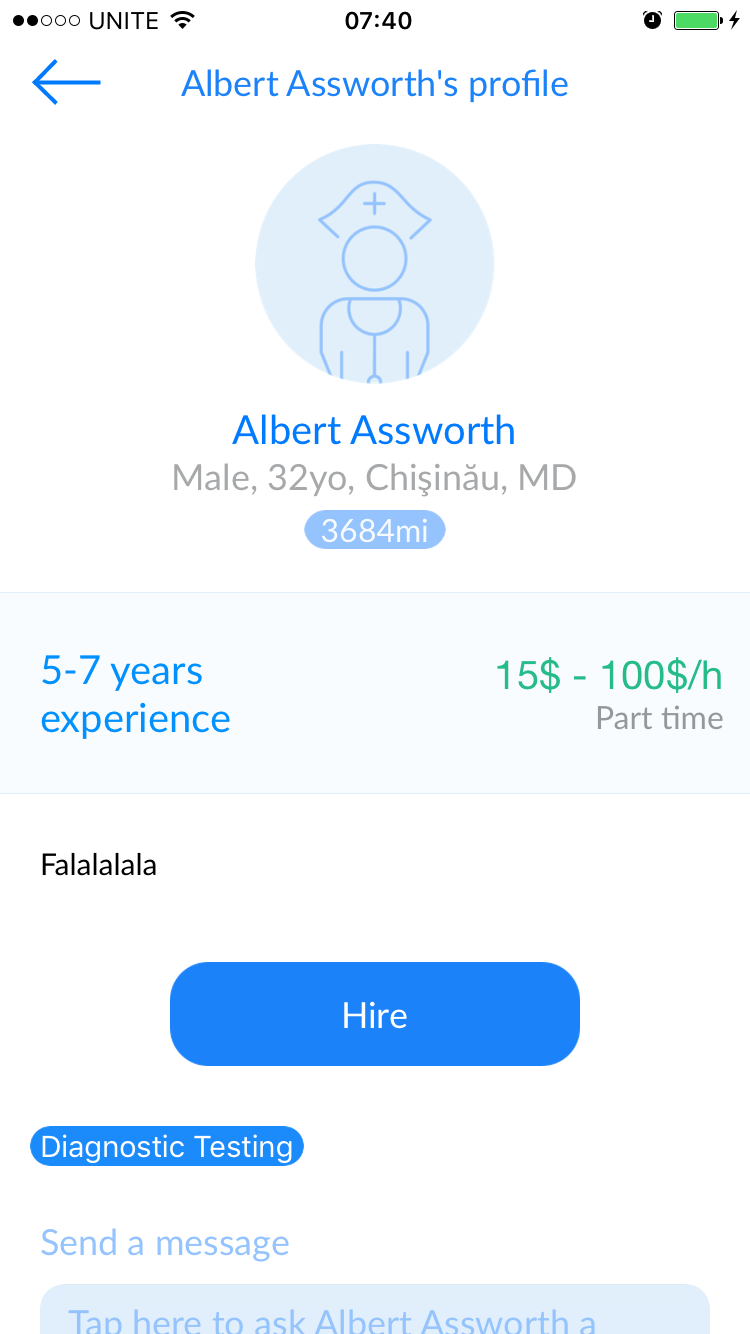
 

(1.5)

For bottom refresh control we are using

#import <CCBottomRefreshControl/UIScrollView+BottomRefreshControl.h>

another pod from github.

Macintosh HD:Users:mc309:Desktop:Screen Shot 2017-01-09 at 10.20.34.png

This ccontroller gives us information about CNA and we can hire him/her for some assistance.

#import "MyProfileCredentialsCell.h"

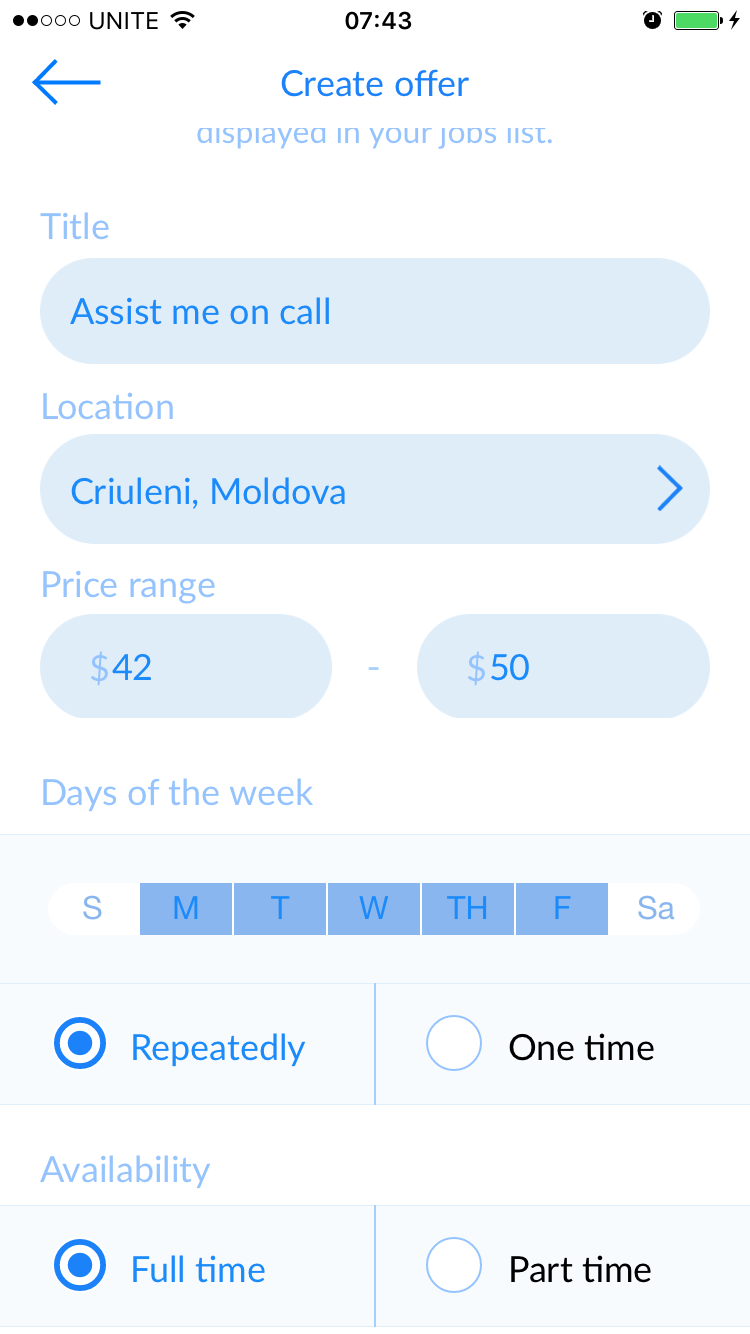
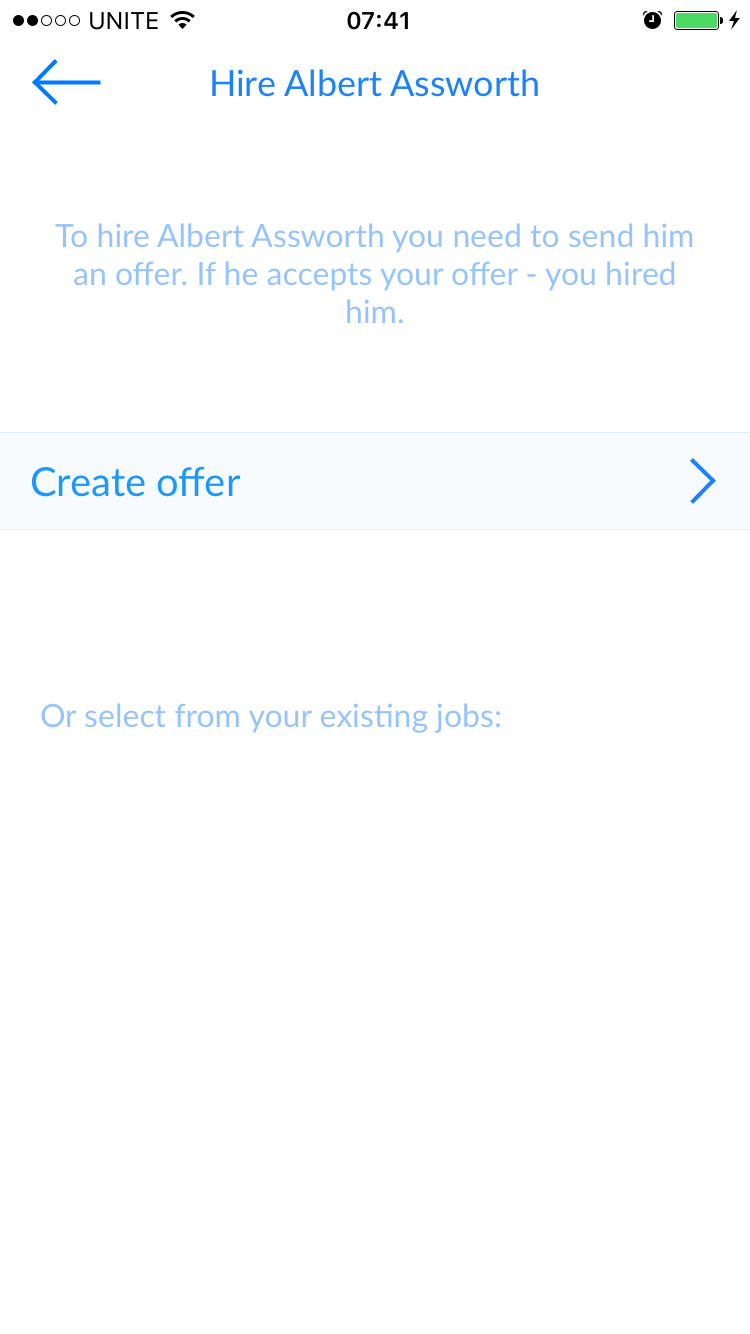
#import "ProfileExperienceAndPriceRangeCell.h"

#import "TaggedStringCell.h"

#import " BlueButtonRoundedCell.h"

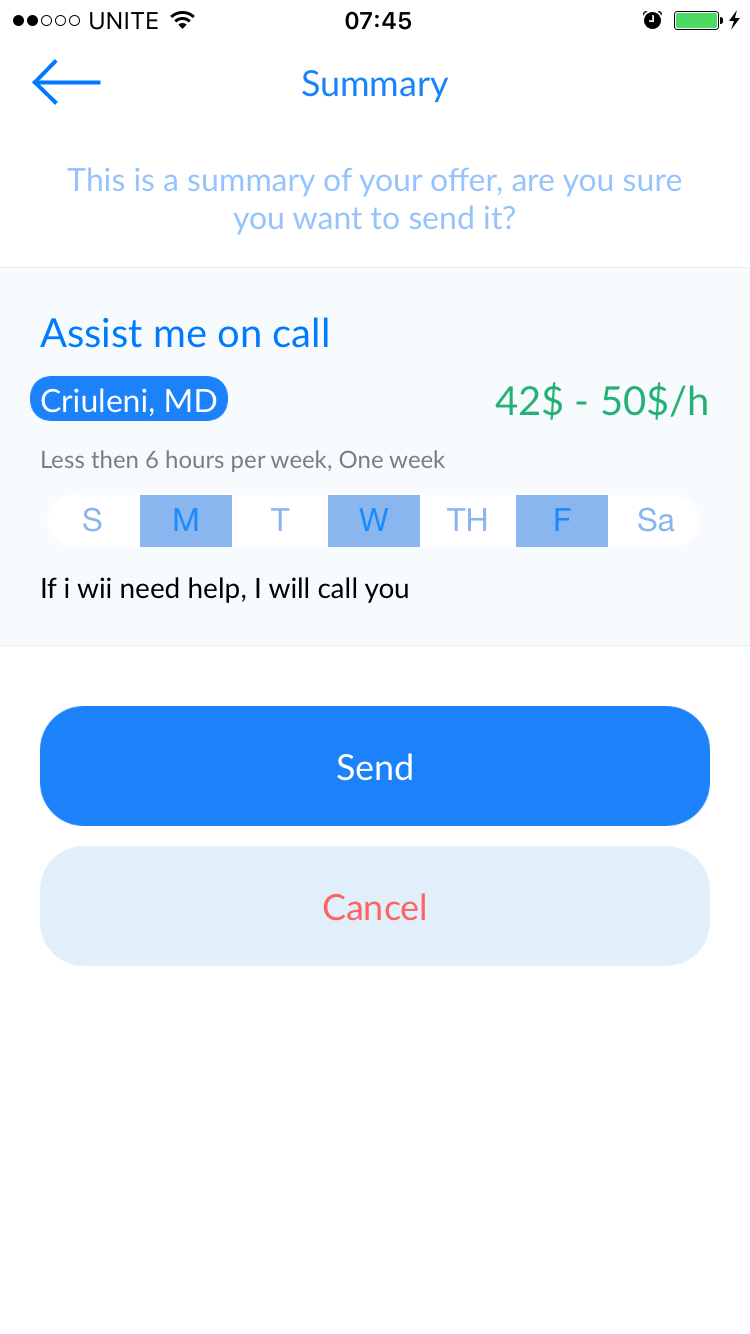
#import "JobDetailsSendMessageCell.h"

#import "ReportButtonCell.h"

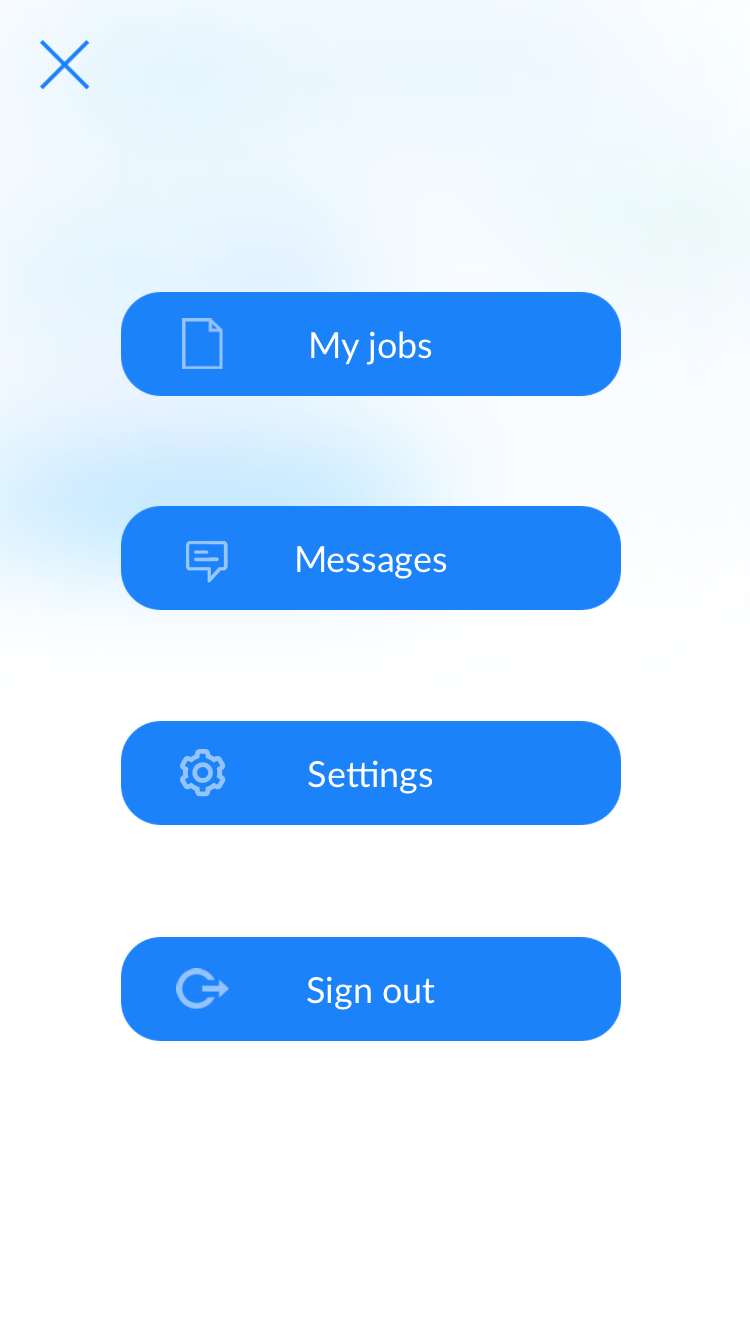


To hire this assistant we can create an offer or choose one of what we have created before.

This controller also uses pagination to show user jobs list . Nothing complicated or interface agorithms added.



After offer or job is created we can see it's summary. Also this summary can be called from our job list .



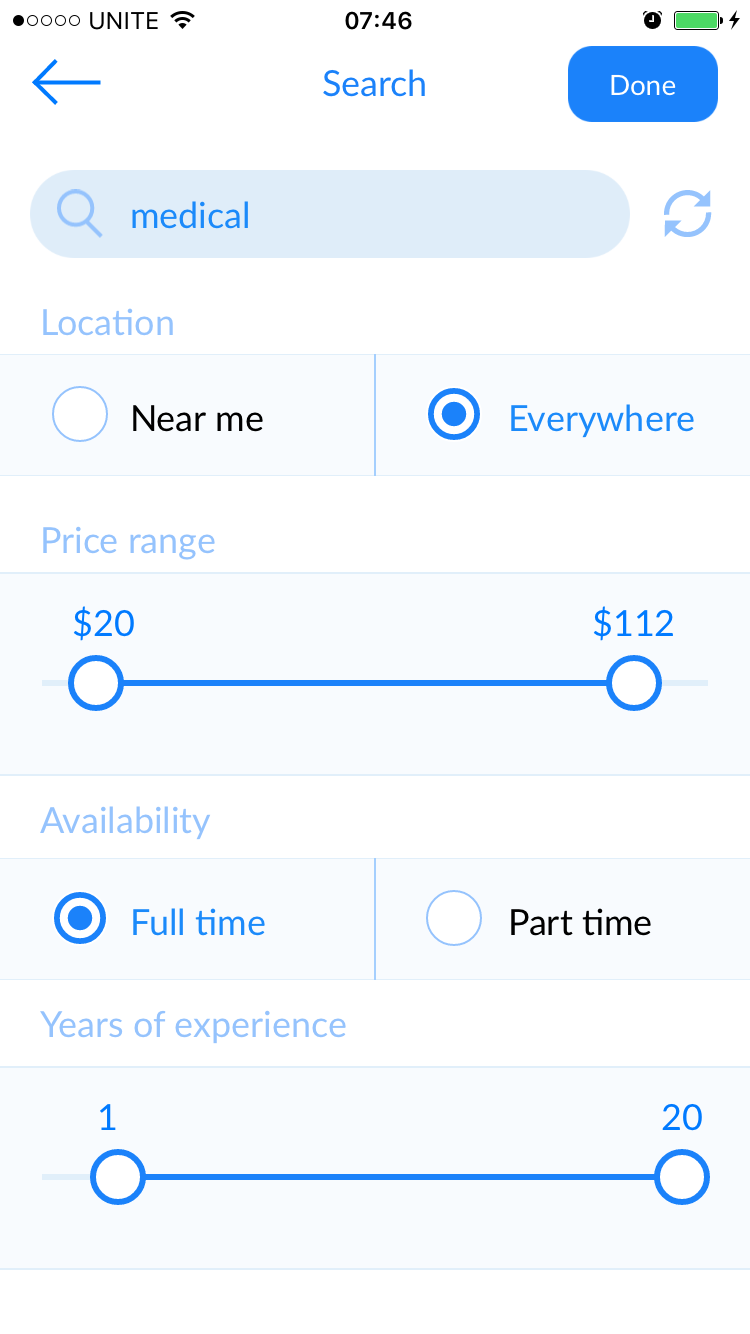
The menu from upper left corner is used to open menu which gives us the opportunity to see more user options.

(1.6)

The menu is not based on table view controller because there is no reason to use it and also it fits nice on each of apple mobile device screen.

The user form in upper right corner opens user's applications (created or received offers). Also uses pagination and is sorted by timestamp to see the newest applications

Macintosh HD:Users:mc309:Desktop:Screen Shot 2017-01-09 at 10.42.08.png

Search options are saved in user

defaults.

@interface Storage : NSObject

+(void)setDefaultSearchSettings;

+(BOOL)defaultsWereSet;

+(void)setSearchSettingsSearchString:(NSString\*)searchString;

+(NSString\*)getSearchSettingsSearchString;

+(void)setSearchSettingsMinExperience:(NSInteger)experience;

+(NSInteger)getSearchSettingsMinExperience;

+(void)setSearchSettingsMaxExperience:(NSInteger)experience;

+(NSInteger)getSearchSettingsMaxExperience;

+(void)setSearchSettingsMinPriceRange:(NSInteger)minPriceRange;

+(NSInteger)getSearchSettingsMinPriceRange;

+(void)setSearchSettingsMaxPriceRange:(NSInteger)maxPriceRange;

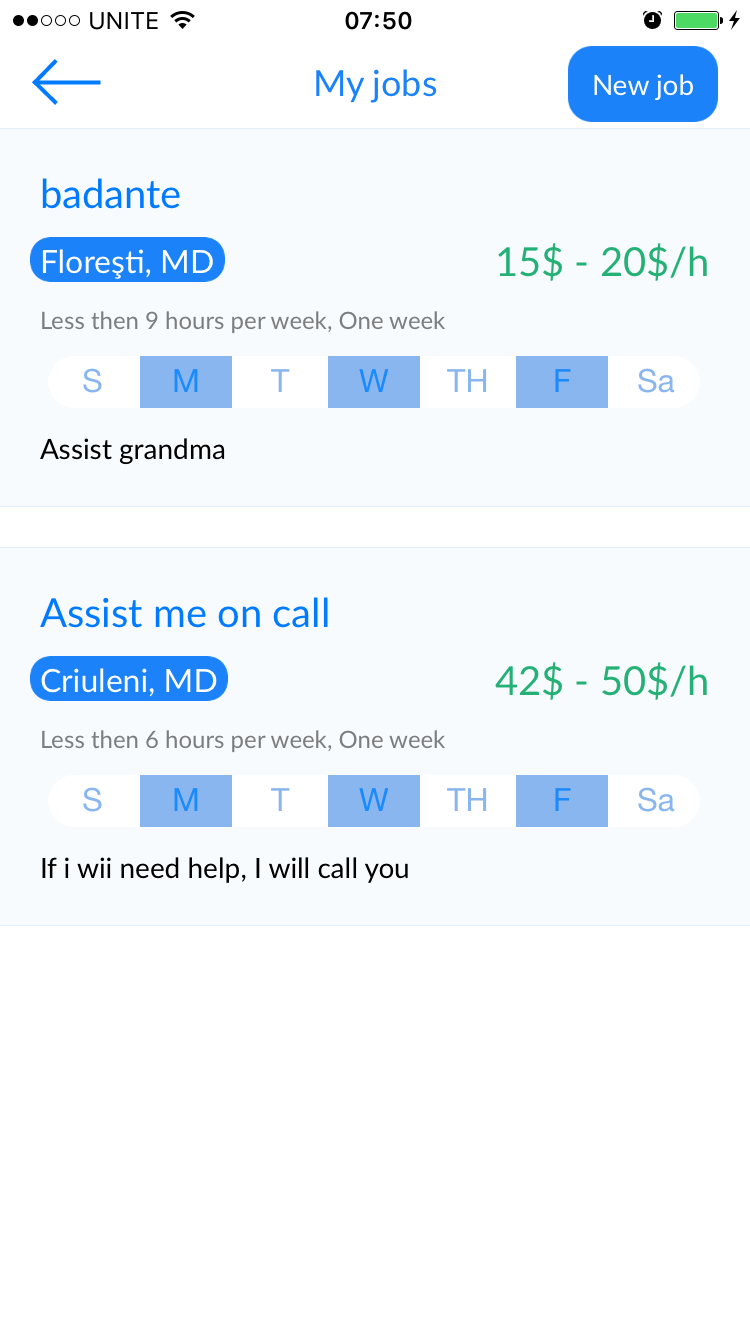
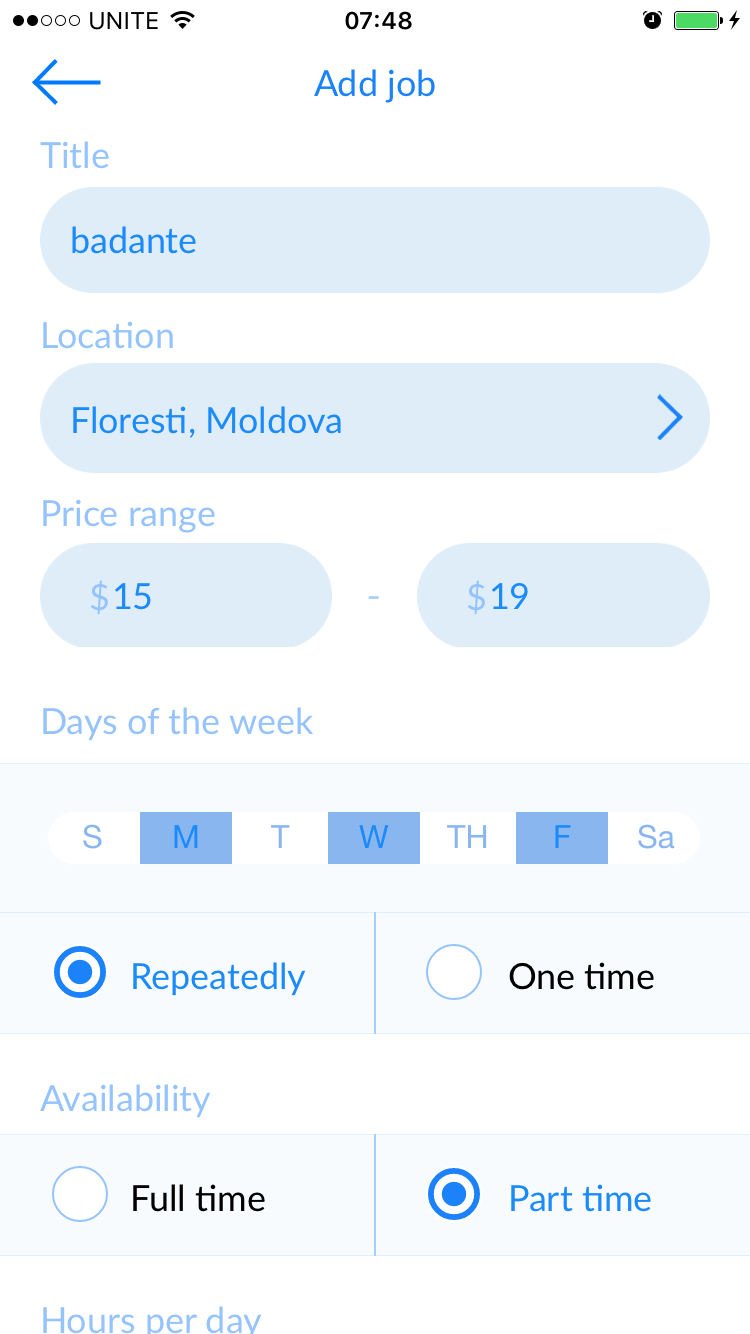
+(NSInteger)getSearchSettingsMaxPriceRange;

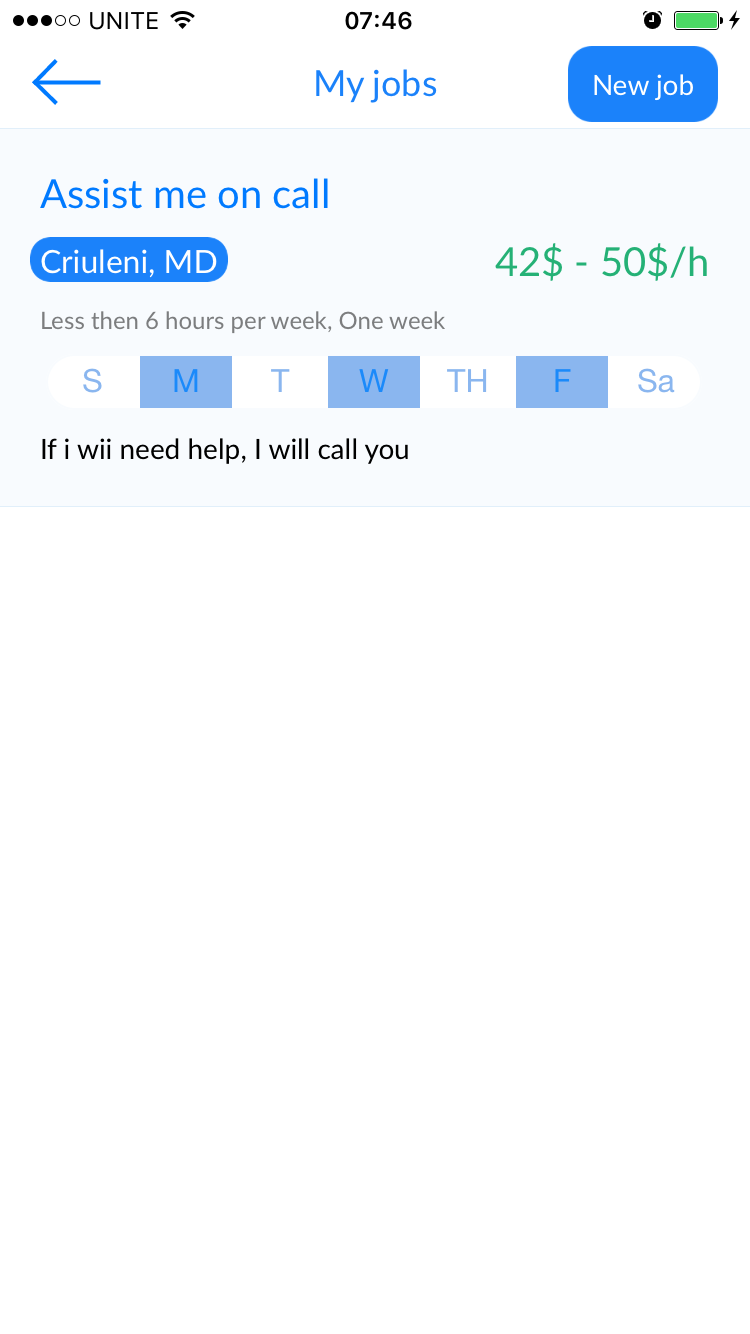
+(void)setSearchSettingsLocation:(NSInteger)type;

+(NSInteger)getSearchSettingsLocation;

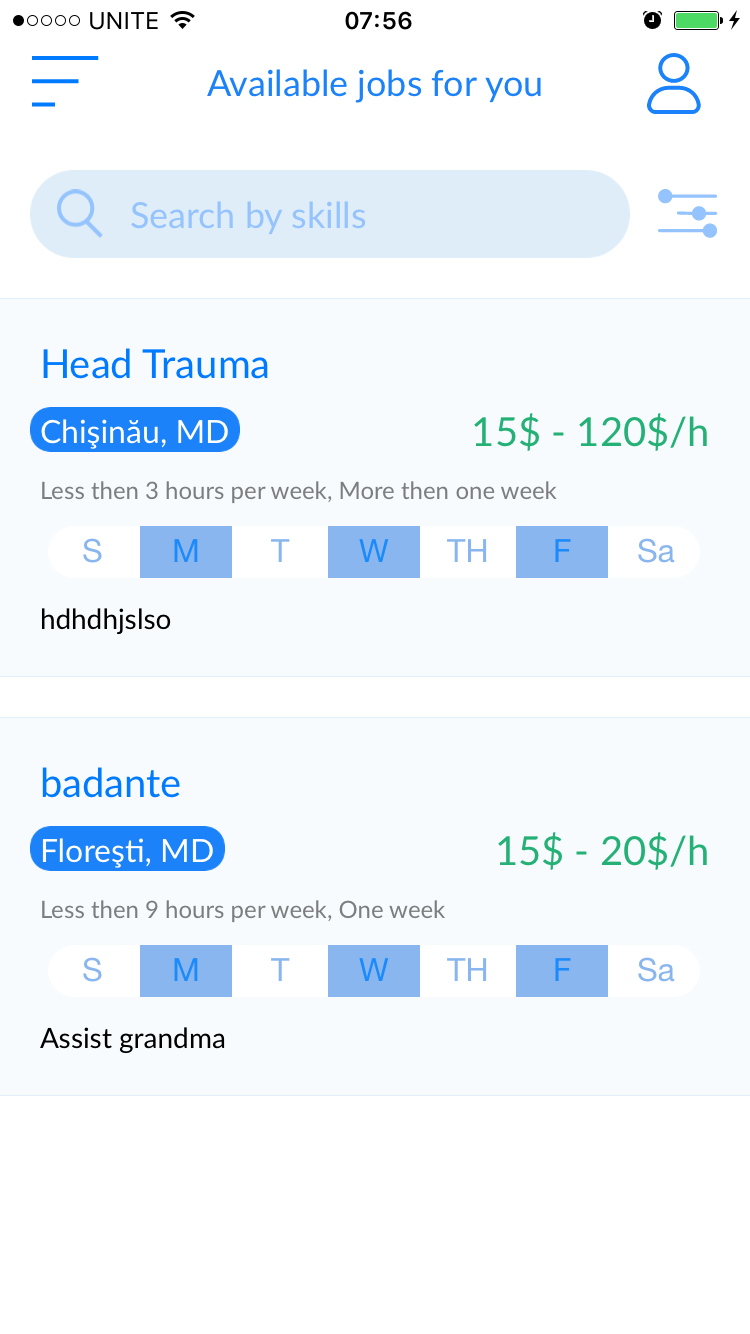
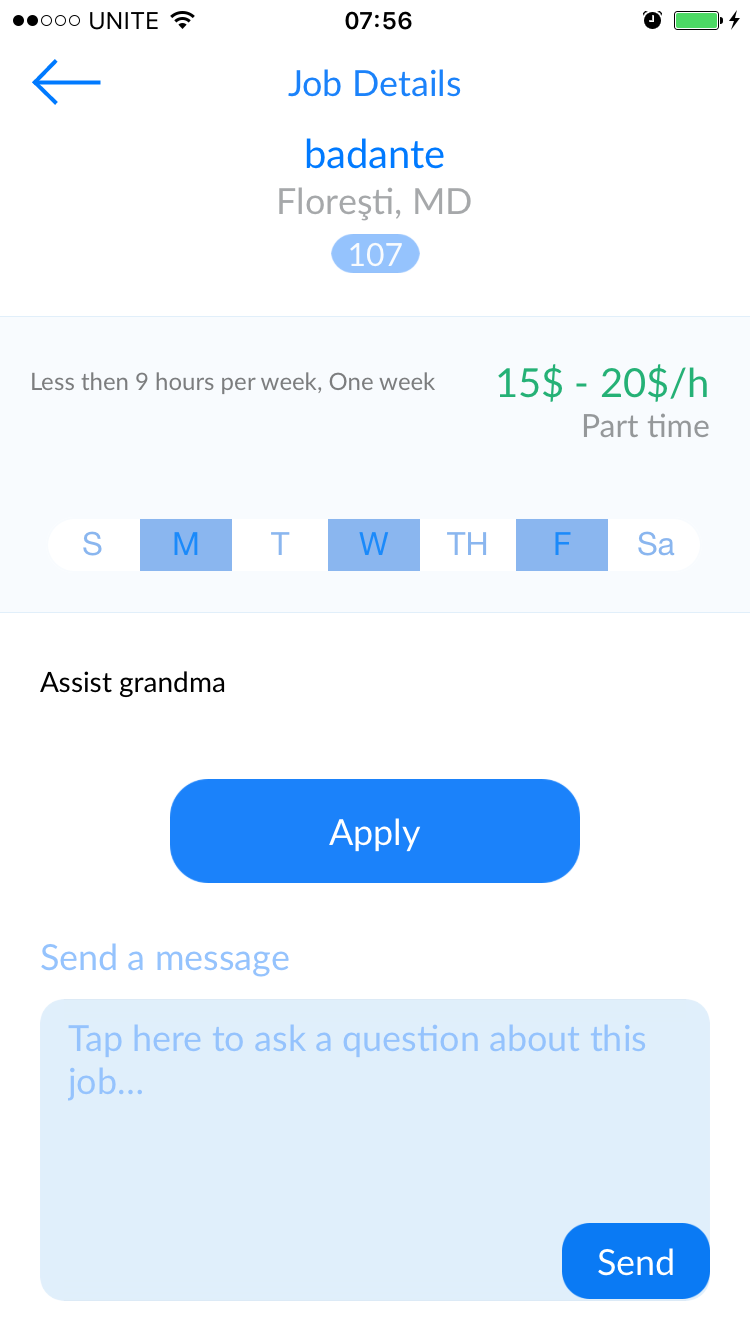
+(void)setSearchSettingsAvalability:(NSInteger)type;

+(NSInteger)getSearchSettingsAvalability;



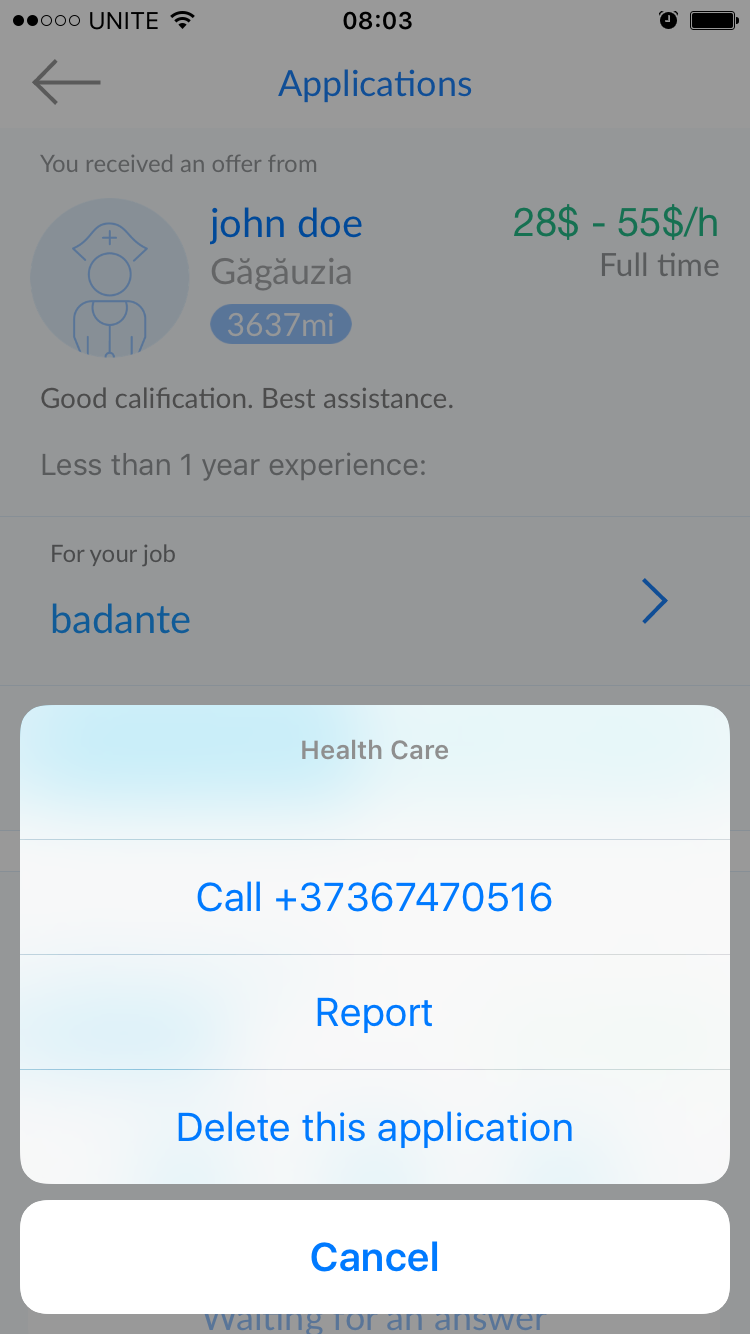


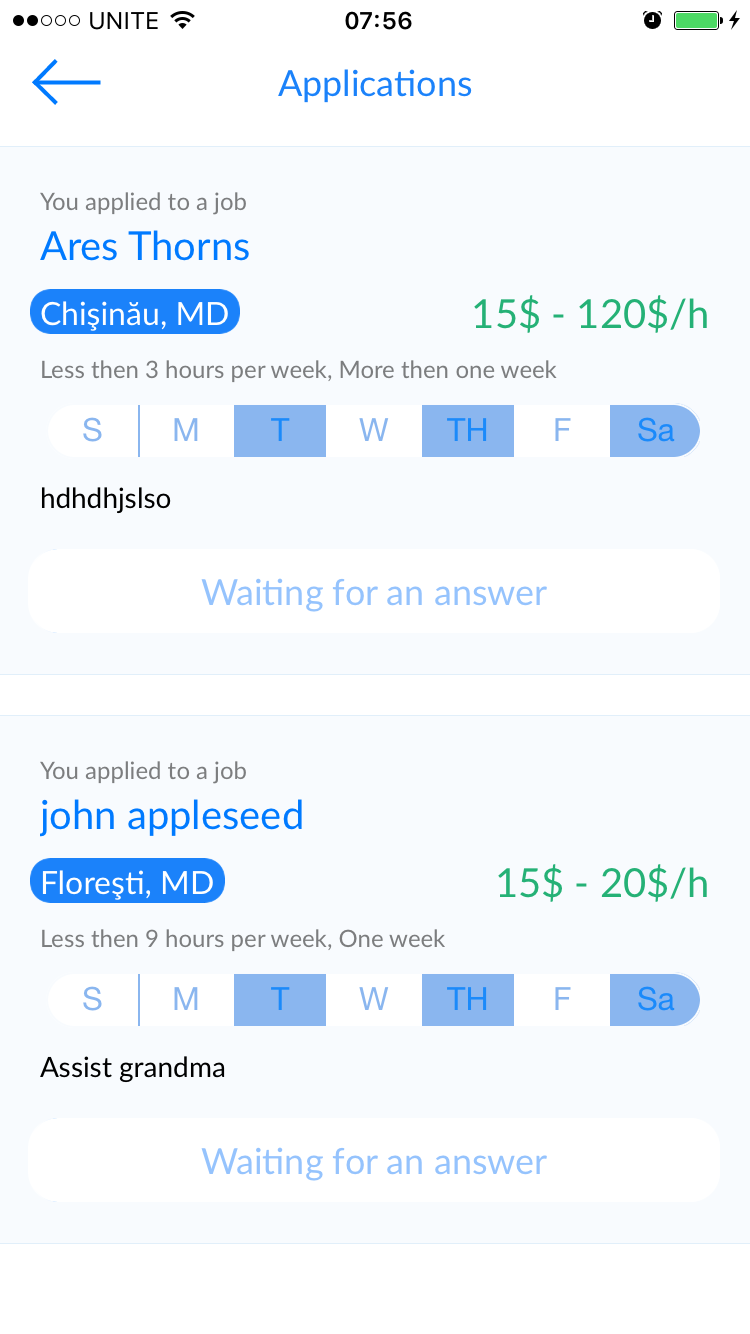
My jobs list. Is an table view with jobs custom cells an uses pagination , on press it opens jobs summary and we can edit.



For CNA we have available jobs which were created by Individual users. On press we will have job details and assistant can apply for it.

Also We can send Messages to client.



CNA has its own applications list and can manage them to refuse or accept offers



Messages

To optimize and customize the code we create some classes

@interface AppUtils : NSObject

+(BOOL) isEmailValid:(NSString \*)checkString;

+(UIColor\*)placeholdersColor;

+(UIColor\*)inputFieldTextColor;

+(UIColor\*)separatorsColor;

+(UIColor\*)grayTextColor;

+(UIColor\*)lightBlueColor;

+(UIColor\*)priceGreenColor;

+(UIColor\*)spinerColor;

+(UIColor\*)buttonBlueColor;

+(UIColor\*)buttonRedTextColor;

+(void)setApplicationAppearence;

+ (UIViewController\*)topViewController;

+(NSArray\*)experienceArray;

+(BOOL)isValidObject:(id)object;

+(BOOL)isValidString:(NSString\*)object;

+(NSString\*)validateString:(NSString\*)object;

+(BOOL)isIphoneVersion:(int)iPhoneVersion;

+(BOOL)validatePhone:(NSString \*)phoneNumber;

@end

These are used mmostly in project and if we will change let say one color all elements using this collor will be changed

Segues identifiers, font names , enums are defined in AppConstants.

@interface AppConstants:NSObject

@end

//Used Fonts

#define \_lato\_font\_regular @"Lato-Regular"

#define \_helveticaNeue\_font\_regular @"HelveticaNeue"

#define itemsPerPage 10

// used segues

#define segue\_showRegistrationStep\_2 @"showRegistrationStep2"

#define segue\_showRegistrationStep\_3 @"showRegistrationStep3"

#define segue\_showRegistrationStep\_4 @"showRegistrationStep4"

#define segue\_showRegistrationStep\_5 @"showRegistrationStep5"

#define segue\_showTermsAndConditions @"showTermsAndConditions"

#define segue\_showProfesionalRegistrationStep\_3 @"showProfesionalRegistrationStep3"

#define segue\_showUserSkils @"showUserSkils"

#define segue\_showExperienceTime @"showExperienceTime"

#define segue\_showJobDetails @"showJobDetails"

#define segue\_showMyJobDetails @"showMyJobDetails"

#define segue\_showEditMyProfile @"showEditMyProfile"

#define segue\_showUserMesages @"showUserMesages"

#define segue\_showEditMyProfileProfesional @"showEditMyProfileProfesional"

#define segue\_showApplications @"showApplications"

#define segue\_showExpiredSubscribtion @"showExpiredSubscribtion"

#define segue\_showCertifiedNursingAssistantProfile @"showCertifiedNursingAssistantProfile"

#define segue\_showHireAssistant @"showHireAssistant"

#define segue\_showAddNewJob @"showAddNewJob"

#define segue\_showEditJob @"showEditJob"

#define segue\_showLocationMap @"showLocationMap"

#define segue\_showCreateOffer @"showCreateOffer"

#define segue\_showJobSummary @"showJobSummary"

#define segue\_showCertifiedNursingAssistants @"showCertifiedNursingAssistants"

#define segue\_showAvailablejobs @"showAvailablejobs"

#define segue\_showLogin @"showLogin"

#define segue\_showLostPasswordInstructions @"showLostPasswordInstructions"

#define appErrorDomain @"Health Care"

#define SUBSCRIPTION\_ID @"com.tusk.healthcare.subscription"

#define SYSTEM\_VERSION\_GRATERTHAN\_OR\_EQUALTO(v) ([[[UIDevice currentDevice] systemVersion] compare:v options:NSNumericSearch] != NSOrderedAscending)

typedef enum {

\_experience\_time,

\_price\_range,

} SliderType;

typedef enum {

\_top,

\_bottom,

} Position;

typedef enum {

\_individual,

\_profesional,

\_unknown\_type

} ClientType;

typedef enum {

\_fullTime,

\_partTime,

} AvailableTime;

typedef enum {

\_female,

\_male,

} Gender;

typedef enum {

\_NearMe,

\_Everywhere

} LocationSearch;

typedef enum {

\_waiting,

\_acccepted,

\_declined,

\_accept\_decline,

} ApplicationsState;

Macintosh HD:Users:mc309:Desktop:Screen Shot 2017-01-09 at 11.13.27.pngData loader contains all requests to server

+(void)recoverUserPasswordForEmail:(NSString\*)email success:(SuccessRH)success fail:(FailRH)fail;

+(void)registerOrEditUser:(User\*)user withPassword:(NSString\*)password andUserPhoto:(UIImage\*)userPhoto isEditing:(BOOL)isEditing success:(SuccessRH)success fail:(FailRH)fail;

+(void)setAvailableUser:(User\*)user available:(NSNumber\*)available success:(SuccessRH)success fail:(FailRH)fail;

+(void)getAvailableJobsForUser:(User\*)user searchString:(NSString\*)searchString forPage:(NSInteger)page success:(SuccessRH)success fail:(FailRH)fail;

+(void)getUserMessageList:(NSString\*)userId success:(SuccessRH)success fail:(FailRH)fail;

+(void)getUserConversation:(NSString\*)userId withPartner:(NSString\*)partnerId success:(SuccessRH)success fail:(FailRH)fail;

+(void)sendMessageFromUser:(NSString\*)userId toPartner:(NSString\*)partnerId text:(NSString\*)text success:(SuccessRH)success fail:(FailRH)fail;

+(void)deleteConversationUser:(NSString\*)userId withPartner:(NSString\*)partnerId success:(SuccessRH)success fail:(FailRH)fail;

+(void)deleteApplication:(ApplicationItem\*)applicationItem success:(SuccessRH)success fail:(FailRH)fail;

+(void)getApplicationsForUser:(NSString\*)userId page:(NSInteger)page success:(SuccessRH)success fail:(FailRH)fail;

+(void)reportJobFromUser:(NSString\*)userId jobId:(NSString\*)jobId success:(SuccessRH)success fail:(FailRH)fail;

+(void)getApplicationsForProfesionalUser:(NSString\*)userId page:(NSInteger)page success:(SuccessRH)success fail:(FailRH)fail;

+(void)reportUserFromUser:(NSString\*)userId reportedUserId:(NSString\*)reportedUserId success:(SuccessRH)success fail:(FailRH)fail;

+(void)getMyJobs:(NSString\*)userId forPage:(NSInteger)page success:(SuccessRH)success fail:(FailRH)fail;

+(void)addJobEdit:(Job\*)job withMyUserId:(NSString\*)userId isEditing:(BOOL)isEditing isPrivate:(BOOL)isPrivate success:(SuccessRH)success fail:(FailRH)fail;

+(void)sendJobTouser:(NSString\*)userId byUser:(NSString\*)byUser jobId:(NSString\*)jobId invite:(BOOL)invite success:(SuccessRH)success fail:(FailRH)fail;

+(void)sendOfferTouser:(NSString\*)userId byUser:(NSString\*)byUser job:(Job\*)job invite:(BOOL)invite success:(SuccessRH)success fail:(FailRH)fail;

+(void)acceptApplication:(NSString\*)userId applicationId:(NSNumber\*)applicationId success:(SuccessRH)success fail:(FailRH)fail;

+(void)refuseApplication:(NSString\*)userId applicationId:(NSNumber\*)applicationId success:(SuccessRH)success fail:(FailRH)fail;

+(void)individualUserProfile:(NSString\*)userId success:(SuccessRH)success fail:(FailRH)fail;

+(void)getCertifiedNursingAssistantsList:(User\*)user searchString:(NSString\*)searchString forPage:(NSInteger)page success:(SuccessRH)success fail:(FailRH)fail;

+(void)deleteUserJob:(NSString\*)userId withJobId:(NSString\*)jobId success:(SuccessRH)success fail:(FailRH)fail;

+(void)deletePush:(NSString\*)userId success:(SuccessRH)success fail:(FailRH)fail;

+(void)sendPush:(NSString\*)pushString fromUser:(NSString\*)userId success:(SuccessRH)success fail:(FailRH)fail;

+(void)changeUserPassword:(NSString\*)oldPassword newPassword:(NSString\*)newPassword forUser:(NSString\*)userId success:(SuccessRH)success fail:(FailRH)fail;

+(void)userCanApplyToJob:(NSString\*)userId jobId:(NSString\*)jobId success:(SuccessRH)success fail:(FailRH)fail;

+(void)checkEmail:(NSString\*)email success:(SuccessRH)success fail:(FailRH)fail;

+(void)sendUserDidSubscribe:(NSString\*)userId success:(SuccessRH)success fail:(FailRH)fail;

+(void)checIfUserIsSubscribed:(NSString\*)userId success:(SuccessRH)success fail:(FailRH)fail;

Macintosh HD:Users:mc309:Desktop:Screen Shot 2017-01-09 at 11.14.30.png

Initialize a session for requests.

@interface HTTPClient : AFHTTPSessionManager

@property (nonatomic , strong) BaseViewController \* requestViewController;

// Initialisation Methods

+ (HTTPClient \*)sharedHTTPClient;

- (instancetype)initWithBaseURL:(NSURL \*)url;

- (BOOL)connected;

@end

+ (HTTPClient \*)sharedHTTPClient

{

static HTTPClient \*\_sharedHTTPClient = nil;

\_sharedHTTPClient.requestViewController = (BaseViewController\*)[AppUtils topViewController];

static dispatch\_once\_t onceToken;

dispatch\_once(&onceToken, ^{

\_sharedHTTPClient = [[HTTPClient new] initWithBaseURL:[NSURL URLWithString:BASE\_URL]];

});

return \_sharedHTTPClient;

}

- (instancetype)initWithBaseURL:(NSURL \*)url

{

self = [super initWithBaseURL:url];

if (self) {

AFNetworkActivityIndicatorManager.sharedManager.enabled = YES;

[[AFNetworkReachabilityManager sharedManager] startMonitoring];

[self.requestSerializer setTimeoutInterval:120.0];

}

return self;

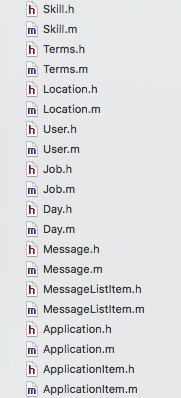
}

- (BOOL)connected {

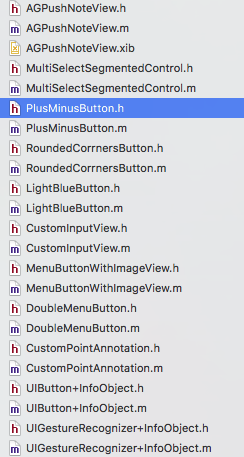
return [AFNetworkReachabilityManager sharedManager].reachable;

}

These are all objects used by users in program.



In application also we use diferent custom butttons , input fields or categories which help us to send objects on touch or to adapt them to design



All base view and navigation controllers used in application



All main and common methods are written here, then the controller which inherit them can use these methods and to add other personal or private methods.