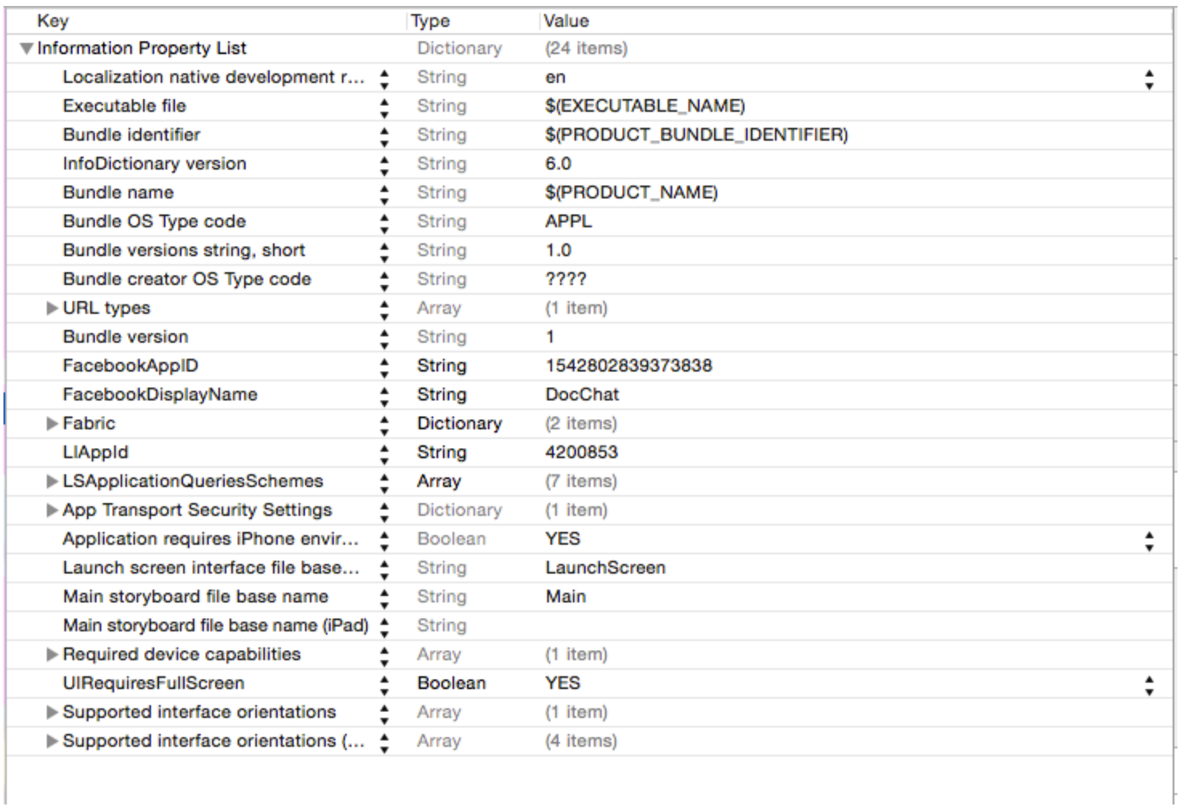

Key	Type	Value
▼ Information Property List	Dictionary	(24 items)
Localization native development r...	String	en
Executable file	String	$(EXECUTABLE_NAME)
Bundle identifier	String	$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle creator OS Type code	String	????
▶ URL types	Array	(1 item)
Bundle version	String	1
FacebookAppID	String	1542802839373838
FacebookDisplayName	String	DocChat
▶ Fabric	Dictionary	(2 items)
LIAppId	String	4200853
▶ LSApplicationQueriesSchemes	Array	(7 items)
▶ App Transport Security Settings	Dictionary	(1 item)
Application requires iPhone envir...	Boolean	YES
Launch screen interface file base...	String	LaunchScreen
Main storyboard file base name	String	Main
Main storyboard file base name (iPad)	String	
▶ Required device capabilities	Array	(1 item)
UIRequiresFullScreen	Boolean	YES
▶ Supported interface orientations	Array	(1 item)
▶ Supported interface orientations (...	Array	(4 items)

```
<key>CFBundleURLTypes</key>
    <array>
        <dict>
            <key>CFBundleURLSchemes</key>
            <array>
                <string>li4200853</string>
                <string>fb1542802839373838</string>
            </array>
        </dict>
    </array>
    <key>CFBundleVersion</key>
    <string>1</string>
    <key>FacebookAppID</key>
    <string>1542802839373838</string>
    <key>FacebookDisplayName</key>
    <string>DocChat</string>
    <key>Fabric</key>
    <dict>
        <key>APIKey</key>
        <string>41807d4eabd028ed826385b1efc3bd654261bcf1</string>
        <key>Kits</key>
        <array>
            <dict>
                <key>KitInfo</key>
                <dict>
                    <key>consumerKey</key>
                    <string>x3WWfB6M1iNDwj5DasWoJy0mO</string>
                    <key>consumerSecret</key>

<string>7F9uBno7SmNcT89ZLa6wrHJltDTy1XZVUS8aXDAt76vUHQeF3N</string>
```

```
                </dict>
                <key>KitName</key>
                <string>Twitter</string>
            </dict>
        </array>
    </dict>
    <key>LIAppId</key>
    <string>4200853</string>
    <key>LSApplicationQueriesSchemes</key>
    <array>
        <string>linkedin</string>
        <string>linkedin-sdk2</string>
        <string>linkedin-sdk</string>
        <string>fbapi</string>
        <string>fb-messenger-api</string>
        <string>fbauth2</string>
        <string>fbshareextension</string>
    </array>
    <key>NSAppTransportSecurity</key>
    <dict>
        <key>NSAllowsArbitraryLoads</key>
        <true/>
    </dict>


//Appdelegate.h

@property (nonatomic, retain) NSDictionary *user;

+ (AppDelegate *)mainDelegate;


//Appdelegate.m

#import <linkedin-sdk/LISDK.h>
#import <Fabric/Fabric.h>
#import <TwitterKit/TwitterKit.h>
#import <FBSDKCoreKit/FBSDKCoreKit.h>

+ (AppDelegate *)mainDelegate {
    return (AppDelegate *)[UIApplication
sharedApplication].delegate;
}

- (BOOL)application:(UIApplication *)application
didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    // Override point for customization after application launch.
    [[FBSDKApplicationDelegate sharedInstance]
application:application

didFinishLaunchingWithOptions:launchOptions];
    [Fabric with:@[[Twitter class]]];

    return YES;
```

```
}

- (void)applicationDidBecomeActive:(UIApplication *)application {
    // Restart any tasks that were paused (or not yet started) while
the application was inactive. If the application was previously in
the background, optionally refresh the user interface.
    [FBSDKAppEvents activateApp];
}

- (BOOL)application:(UIApplication *)application openURL:(NSURL
*)url sourceApplication:(NSString *)sourceApplication annotation:
(id)annotation {
    if ([LISDKCallbackHandler shouldHandleUrl:url]) {
        return [LISDKCallbackHandler application:application
openURL:url sourceApplication:sourceApplication
annotation:annotation];
    }else{
        return [[FBSDKApplicationDelegate sharedInstance]
application:application

openURL:url

sourceApplication:sourceApplication

annotation:annotation];
    }

    return YES;
}


//Viewcontroller.m

#import <TwitterKit/TwitterKit.h>
#import <linkedin-sdk/LISDK.h>

#import <FBSDKCoreKit/FBSDKCoreKit.h>
#import <FBSDKLoginKit/FBSDKLoginKit.h>




//.h
{
IBOutlet UITextField *LEmail, *LPassword;
    IBOutlet UITextField *RFname, *RLname, *RDob, *RCountry,
*REmail, *RPassword;
    UIDatePicker *Picker;
    NSString *Profile_Pic_Url;
    NSMutableDictionary *profile;
}

- (IBAction)ConnectWithFacebook:(id)sender{
    FBSDKLoginManager *login = [[FBSDKLoginManager alloc] init];
    [login
     logInWithReadPermissions: @[@"public_profile", @"email"]
     fromViewController:self
     handler:^(FBSDKLoginManagerLoginResult *result, NSError *error)
{
         if (error) {
             NSLog(@"Process error");
         } else if (result.isCancelled) {
             NSLog(@"Cancelled");
         } else {
              [self GetFacebookProfile];
         }
     }];
}

- (void)GetFacebookProfile{
    FBSDKGraphRequest *request = [[FBSDKGraphRequest alloc]
                                  initWithGraphPath:@"/me"
                                  parameters:@{@"fields":
@"id,first_name,last_name,email,location"}
                                  HTTPMethod:@"GET"];
    [request
startWithCompletionHandler:^(FBSDKGraphRequestConnection
*connection,
                                        id result,
                                        NSError *error) {
        // Handle the result
        Profile_Pic_Url = [NSString stringWithFormat:@"http://
graph.facebook.com/%@/picture?type=large",[result
objectForKey:@"id"]];
        [profile setObject:[NSString stringWithFormat:@"%@ %@",
[result objectForKey:@"first_name"],[result
objectForKey:@"last_name"]] forKey:@"NAME"];
        [profile setObject:[NSString stringWithFormat:@"%@",[result
objectForKey:@"email"]] forKey:@"EMAIL"];
        [profile setObject:[NSString stringWithFormat:@"%@",[result
objectForKey:@"id"]] forKey:@"ID"];
        [profile setObject:Profile_Pic_Url forKey:@"PIC"];

        [[AppDelegate mainDelegate] setUser:profile];

        [self performSegueWithIdentifier:@"Menu" sender:self];
    }];
}

- (IBAction)ConnectWithTwitter:(id)sender{
    [[Twitter sharedInstance] logInWithCompletion:^(TWTRSession *
_Nullable session, NSError * _Nullable error) {
        NSLog(@"%@",session.userName);
        if (error == nil) {
            Profile_Pic_Url = [NSString stringWithFormat:@"https://
twitter.com/%@/profile_image?size=original",session.userName];
            [profile setObject:[NSString
stringWithFormat:@"%@",session.userName] forKey:@"NAME"];
```

```
            [profile setObject:[NSString
stringWithFormat:@"%@",session.userID] forKey:@"ID"];
            [profile setObject:Profile_Pic_Url forKey:@"PIC"];

            [self GetTwitterProfile];
        }

    }];
}

- (void)GetTwitterProfile{
    if ([[Twitter sharedInstance] session]) {
        TWTRShareEmailViewController* shareEmailViewController =
[[TWTRShareEmailViewController alloc] initWithCompletion:^(NSString*
email, NSError* error) {
            NSLog(@"Email %@, Error: %@", email, error);
            [profile setObject:[NSString
stringWithFormat:@"%@",email] forKey:@"EMAIL"];
            [[AppDelegate mainDelegate] setUser:profile];
            [self performSegueWithIdentifier:@"Menu" sender:self];
        }];
        [self presentViewController:shareEmailViewController
animated:YES completion:nil];
    } else {
        // TODO: Handle user not signed in (e.g. attempt to log in
or show an alert)
    }
}

- (IBAction)connectWithLinkedIn:(id)sender {
    NSArray *permissions = [NSArray
arrayWithObjects:LISDK_BASIC_PROFILE_PERMISSION,
LISDK_EMAILADDRESS_PERMISSION, nil];
    [LISDKSessionManager createSessionWithAuth:permissions
                                         state:nil
                        showGoToAppStoreDialog:YES
                                  successBlock:^(NSString
*returnState) {
                                      NSLog(@"returned state
%@",returnState);
                                      [self GetLinkedinProfile];

                                  }
                                    errorBlock:^(NSError *error) {
                                        NSLog(@"%s %@","error
called! ", [error description]);
                                    }
     ];
}

- (void)GetLinkedinProfile{
    NSString *url = @"https://api.linkedin.com/v1/people/~:
(id,first-name,last-name,picture-url,email-address,location:
(country:(code)))?format=json";//email-address id first-name last-
```

```
name location:(country:(code))
    if ([LISDKSessionManager hasValidSession]) {
        [[LISDKAPIHelper sharedInstance] getRequest:url

success:^(LISDKAPIResponse *response) {
                                               // do something with
response
                                               NSDictionary *dict =
[NSJSONSerialization JSONObjectWithData:[response.data
dataUsingEncoding:NSUTF8StringEncoding]
options:NSJSONReadingMutableContainers error:nil];
                                               NSLog(@"%@",dict);

                                               Profile_Pic_Url =
[dict objectForKey:@"pictureUrl"];

                                               [profile setObject:
[NSString stringWithFormat:@"%@ %@",[dict objectForKey:@"first-
name"],[dict objectForKey:@"last-name"]] forKey:@"NAME"];
                                               [profile setObject:
[NSString stringWithFormat:@"%@",[dict objectForKey:@"email-
address"]] forKey:@"EMAIL"];
                                               [profile setObject:
[NSString stringWithFormat:@"%@",[dict objectForKey:@"id"]]
forKey:@"ID"];
                                               [profile
setObject:Profile_Pic_Url forKey:@"PIC"];

                                               [[AppDelegate
mainDelegate] setUser:profile];

                                               [self
performSegueWithIdentifier:@"Menu" sender:self];


                                           }
                                             error:^(LISDKAPIError
*apiError) {
                                                 // do something
with error
                                             }];
    }

}
```