TRiGlobe Part 4 – Login

So we have the basic structure of the game but we want to be able to handle multiple users. Each user will have to authenticate in order to play. We also want to be able to store the user’s credentials so he doesn’t have to enter them each time. NSUserDefaults is a pretty simple concept. An iOS app basically stores certain data locally. We will start by going to our ModalVIewController Class and adding some properties and methods. Then we will work on some php code for our backend.

First import the Santiapps Helper class created above into our .h file. While you’re there add some protocols, properties and methods:

#import <UIKit/UIKit.h>

#import "SantiappsHelper.h"

@interface ModalViewController : UIViewController <UINavigationControllerDelegate, UIImagePickerControllerDelegate>{

}

@property (nonatomic,retain) NSURLConnection \*myConnection;

@property (nonatomic,retain) NSMutableData \*incomingPostData;

@property (nonatomic,retain) IBOutlet UIButton \*registerUser;

@property (nonatomic,retain) IBOutlet UIButton \*loginUser;

@property (nonatomic,retain) IBOutlet UIButton \*dismissModal;

@property (nonatomic,retain) IBOutlet UITextField \*userName;

@property (nonatomic,retain) IBOutlet UITextField \*userPass;

@property (nonatomic,retain) IBOutlet UITextField \*userEmail;

-(IBAction)userWillDismiss;

-(IBAction)userWillLogIn;

-(IBAction)userWillRegister;

-(NSDictionary\*)addUser:(NSString \*)usuario withPass:(NSString \*)clave withAddress:(NSString\*)direccion;

-(NSString\*)checkUserLogin:(NSString \*)loginUsuario withPass:(NSString \*)loginClave;

@end

The navigation controller delegate is required along with the UIImagePickerControllerDelegate for picking images and we will use them for when the user uploads his picture.

As for properties, we will revisit our old friend, NSURLConnection, for which we require NSMutableData. Finally connect the buttons and textfields to their controls.

The methods will be used for when the user dismisses the login screen. Its always important to add a way out of a login screen. Even if that means the user will have limited access to the app. The login method will do just that whereas the register method will create a new user. Finally we will use the other methods for checking a login request.

Let’s review what these look like:

#import "ModalViewController.h"

@implementation ModalViewController

-(IBAction)userWillLogIn{

//Call checklogin.php to compare user/pass and return value

NSString \*loginSuccess = [self checkUserLogin:self.userName.text withPass:self.userPass.text];

if ([loginSuccess isEqualToString:@"SUCCESS"]) {

NSLog(@"We did it...");

//If all is well, then store userDefaults

NSUserDefaults \*prefs = [NSUserDefaults standardUserDefaults];

[prefs setObject:self.userName.text forKey:@"storedUser"];

[prefs setObject:self.userPass.text forKey:@"storedPass"];

[prefs synchronize];

[self dismissViewControllerAnimated:YES completion:nil];

} else {

NSLog(@"Authentication Failed..");

//Add UIAlertView

UIAlertView \*alertView = [[UIAlertView alloc] initWithTitle:@"Oops..." message:loginSuccess delegate:nil cancelButtonTitle:@"OK" otherButtonTitles:nil];

[alertView show];

[alertView release];

}

//if checklogin.php = OK then dismiss

}

-(IBAction)userWillRegister{

// Create uniqueIdentifier

//UIDevice \*device = [UIDevice currentDevice];

//NSString \*uniqueIdentifier = [device uniqueIdentifier];

//1. Get the LOGIN errorsDictionary

NSDictionary \*errorsDict = [self addUser:self.userName.text withPass:self.userPass.text withAddress:self.userEmail.text];

//2. if dictcount = 1 then dismiss && = 5926, then OK

if ([[errorsDict objectForKey:@"code1"] intValue] == 5926) {

NSLog(@"yeay");

//If all is well, then store userDefaults

NSUserDefaults \*prefs = [NSUserDefaults standardUserDefaults];

[prefs setObject:self.userName.text forKey:@"storedUser"];

[prefs setObject:self.userPass.text forKey:@"storedPass"];

[prefs synchronize];

//Dismis ModalVC after users first login

[self dismissViewControllerAnimated:YES completion:nil];

} else {

NSLog(@"Errors");

// extract nsdict

NSMutableString \*resultString = [NSMutableString string];

for (NSString\* key in [errorsDict allKeys]){

if ([resultString length]>0)

[resultString appendString:@"&"];

[resultString appendFormat:@"%@=%@", key, [errorsDict objectForKey:key]];

}

NSLog(@"rS:%@",resultString);

//Add UIAlertView

UIAlertView \*alertView = [[UIAlertView alloc] initWithTitle:@"Check your user,pass or email" message:resultString delegate:nil cancelButtonTitle:@"OK" otherButtonTitles:nil];

[alertView show];

[alertView release];

}

}

-(IBAction)userWillDismiss{

[self dismissViewControllerAnimated:YES completion:nil];

}

//registering user----moved to modalVC

-(NSString\*)checkUserLogin:(NSString \*)loginUsuario withPass:(NSString \*)loginClave{

//CREATE URL TO SEND

NSString \*urlString = [NSString stringWithFormat:@"username=%@&password=%@",loginUsuario,loginClave];

NSLog(@"login string:%@",urlString);

//POST THE STRING

NSData \*postData = [urlString dataUsingEncoding:NSUTF8StringEncoding allowLossyConversion:YES];

NSString \*postLength = [NSString stringWithFormat:@"%d", [postData length]];

NSMutableURLRequest \*request = [[[NSMutableURLRequest alloc] init] autorelease];

NSURL \*url = [NSURL URLWithString:[NSString stringWithFormat:@"http://www.yourserver.com/iGlobe/login2/checklogin.php"]];

[request setURL:url];

[request setHTTPMethod:@"POST"];

[request setValue:postLength forHTTPHeaderField:@"Content-Length"];

[request setValue:@"application/x-www-form-urlencoded" forHTTPHeaderField:@"Content-Type"];

[request setHTTPBody:postData];

NSURLResponse \*response = nil;

NSError \*error = nil;

// We should probably be parsing the data returned by this call, for now just check the error.

NSData \*myData = [NSURLConnection sendSynchronousRequest:request returningResponse:&response error:&error];

//NSError \*outError = NULL;

//NSDictionary \*tempDict = [NSDictionary dictionaryWithJSONData:myData error:&outError];

NSString \*string=[[NSString alloc] initWithData:myData encoding:NSUTF8StringEncoding ];

//NSLog(@"string es:%@, %i, %C",string,[string length],[string characterAtIndex:7]);

NSLog(@"string ess:%@",string);

//NSLog(@"Dict of errors:%@",tempDict);

return string;

}

-(NSDictionary\*)addUser:(NSString \*)usuario withPass:(NSString \*)clave withAddress:(NSString\*)direccion{

//CREATE URL TO SEND

NSString \*urlString = [NSString stringWithFormat:@"username=%@&password=%@&email=%@",usuario,clave,direccion];

NSLog(@"user registration string:%@",urlString);

//POST THE STRING

NSData \*postData = [urlString dataUsingEncoding:NSUTF8StringEncoding allowLossyConversion:YES];

NSString \*postLength = [NSString stringWithFormat:@"%d", [postData length]];

NSMutableURLRequest \*request = [[[NSMutableURLRequest alloc] init] autorelease];

NSURL \*url = [NSURL URLWithString:[NSString stringWithFormat:@"http://www.yourserver.com/iGlobe/login2/user\_add\_save.php"]];

[request setURL:url];

[request setHTTPMethod:@"POST"];

[request setValue:postLength forHTTPHeaderField:@"Content-Length"];

[request setValue:@"application/x-www-form-urlencoded" forHTTPHeaderField:@"Content-Type"];

[request setHTTPBody:postData];

NSURLResponse \*response = nil;

NSError \*error = nil;

// We should probably be parsing the data returned by this call, for now just check the error.

NSData \*myData = [NSURLConnection sendSynchronousRequest:request returningResponse:&response error:&error];

NSError \*outError = NULL;

NSDictionary \*tempDict = [NSJSONSerialization JSONObjectWithData:myData options:NSJSONReadingMutableContainers error:&outError];

NSLog(@"Dict of errors:%@",tempDict);

return tempDict;

}

///////////////////////////////////----------------PHOTO UPLOAD-----

-(IBAction)uploadPhoto:(id)sender{

NSLog(@"picker");

UIImagePickerController \*picker = [[UIImagePickerController alloc] init];

picker.delegate = self;

picker.sourceType = UIImagePickerControllerSourceTypePhotoLibrary;

[self presentViewController:picker animated:YES completion:nil];

}

-(void)imagePickerController:(UIImagePickerController \*)picker didFinishPickingMediaWithInfo:(NSDictionary \*)info{

NSLog(@"pickerPicked");

UIImage \*profile\_image = [info objectForKey:UIImagePickerControllerOriginalImage];

[self uploadImage:UIImageJPEGRepresentation(profile\_image, 1.0) filename:@"globe57.png"];

[picker dismissViewControllerAnimated:YES completion:nil];

}

-(void)imagePickerControllerDidCancel:(UIImagePickerController \*)picker

{

[picker dismissViewControllerAnimated:YES completion:nil];

}

- (BOOL)uploadImage:(NSData \*)imageData filename:(NSString \*)filename{

NSLog(@"uploading");

NSString \*urlString = @"http://www.yourserver.com/iGlobe/photos/uploadPhoto.php";

NSMutableURLRequest \*request = [[[NSMutableURLRequest alloc] init] autorelease];

[request setURL:[NSURL URLWithString:urlString]];

[request setHTTPMethod:@"POST"];

NSString \*boundary = @"0xKhTmLbOuNdArY";

NSString \*contentType = [NSString stringWithFormat:@"multipart/form-data; boundary=%@",boundary];

[request addValue:contentType forHTTPHeaderField: @"Content-Type"];

NSMutableData \*body = [NSMutableData data];

[body appendData:[[NSString stringWithFormat:@"\r\n--%@\r\n",boundary] dataUsingEncoding:NSUTF8StringEncoding]];

//Set the filename

[body appendData:[[NSString stringWithString:[NSString stringWithFormat:@"Content-Disposition: form-data; name=\"userfile\"; filename=\"%@\"\r\n",filename]] dataUsingEncoding:NSUTF8StringEncoding]];

[body appendData:[@"Content-Type: application/octet-stream\r\n\r\n" dataUsingEncoding:NSUTF8StringEncoding]];

//append the image data

[body appendData:[NSData dataWithData:imageData]];

[body appendData:[[NSString stringWithFormat:@"\r\n--%@--\r\n",boundary] dataUsingEncoding:NSUTF8StringEncoding]];

[request setHTTPBody:body];

NSData \*returnData = [NSURLConnection sendSynchronousRequest:request returningResponse:nil error:nil];

NSString \*returnString = [[[NSString alloc] initWithData:returnData encoding:NSUTF8StringEncoding] autorelease];

NSLog(@"returningOKString");

return ([returnString isEqualToString:@"OK"]);

}

//////////////////////////////

- (void)didReceiveMemoryWarning {

// Releases the view if it doesn't have a superview.

[super didReceiveMemoryWarning];

// Release any cached data, images, etc. that aren't in use.

}

- (void)viewDidUnload {

[super viewDidUnload];

}

- (void)dealloc {

[super dealloc];

}

@end

This is a lot of code so we will review it method by method. Here we go, method by method:

1. userWillLogin. This method makes a call to the web fetch method that sends the submitted data to the server and runs a check (checkUserLogin). If authentication was successful then we store the user data in NSUserDefaults so that the user doesn’t have to log in again. Otherwise we display an alert view with the server error response.
2. checkUserLogin. This method actually makes the web fetch and returns the server response.
3. userWillRegister. This is the method that calls to create a new user when the user taps on the register button. It calls the addUser:withPass:withAddress:.
4. addUser:withPass:withAddress:. This method actually makes the web post to add that user to our web database.
5. userWillDismiss is pretty self explanatory ☺
6. uploadPhoto is in charge of calling the UIImagePicker so that the user can select his or her image to upload.
7. imagePickerController:didFinishPickingMediaWithInfo. This method returns the image selected by the user and passes it to uploadImage. However if you notice, for simplicity sake we are not using the picked image, instead we have hardcoded a placeholder image just to make things move along more smoothly.
8. uploadImage:filename. This is the method that actually makes the web post to upload the image to the server.

Now we need the backend to respond to these 2 requests. Specifically we need 2 php files (checklogin.php & user\_add\_save.php) and their helper files. Here is checklogin.php:

<?php

include"login\_config.php";

//Connection String Variables\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

// connect to the server

mysql\_connect( $db\_host, $db\_username, $db\_password )

or die( "Error! Could not connect to database: " . mysql\_error() );

// select the database

mysql\_select\_db( $db )

or die( "Error! Could not select the database: " . mysql\_error() );

//IBM suggested scrub for URL request

$urlun = strip\_tags(substr($\_REQUEST['username'],0,32));

$urlpw = strip\_tags(substr($\_REQUEST['password'],0,32));

$cleanpw = md5($urlpw);

//echo"Cleanpw: $cleanpw<br>";

//$sql="SELECT \* FROM agents WHERE username='$urlun' and password='$urlpw'";

$sql="SELECT \* FROM users WHERE username='$urlun' and password='$cleanpw'";

$result=mysql\_query($sql);

// Mysql\_num\_row is counting table rows

$count=mysql\_num\_rows($result);

// If result matches $myusername and $mypassword, table row must be 1 row

//echo"Count:$count<br>";

if($count==1){

// Register $myusername and redirect to file designated success file

$cookie\_name ="$cookiename";

$cookie\_value ="$urlun";

//set to 24 hours

$cookie\_expire ="86400";

setcookie($cookie\_name,$cookie\_value,time() + (86400),"/", $cookie\_domain);

header("location:$successful\_login\_url");

}else{

header("location:$failed\_login");

}

?>

and user\_add\_save.php:

<?

include"master\_inc.php";

//--------------------------------------------------------------------------RECEIVE LOCAL VARIABLES FROM FORM

$lastname = strip\_tags(substr($\_POST['lastname'],0,32));

$firstname = strip\_tags(substr($\_POST['firstname'],0,32));

$phone = strip\_tags(substr($\_POST['phone'],0,32));

$password\_hint=$\_REQUEST['password\_hint'];

$noERROR=1;

$udidposted = 9;

//---------------------------------------------------------------------------CHECK IF USERNAME IS LONG ENOUGH

$username = strip\_tags(substr($\_POST['username'],0,32));

if(trim($username)!=='' && strlen(trim($username)) >= 4){

//---------------------------------------------------------------------------IF LONG ENOUGH THEN RUN A QUERY GETTING ALL DATA FROM THAT USER

$sql="SELECT \* FROM users WHERE username='$username'";

$result=mysql\_query($sql);

$count=mysql\_num\_rows($result);

//---------------------------------------------------------------------------IF $sql GOTTEN HAS ROW COUNT > 1 THEN USER ALREADY EXISTS----------------SET EXISTING USER 104 FLAG

if($count>0){

$username\_already\_in\_use = 3141;

}

//---------------------------------------------------------------------------ELSE USERNAME IS TOO SHORT?!?!-------------------------------------------------------SET USER TOO SHORT 104 FLAG

}else{

$username\_too\_short = 3142;}

//---------------------------------------------------------------------------EMAIL FORMAT CHECK

$email\_raw = $\_REQUEST['email'];

if(eregi("^[\_a-z0-9-]+(\.[\_a-z0-9-]+)\*@([a-z0-9-]{2,3})+(\.[a-z0-9-]+)\*(\.[a-z]{2,3})$", $email\_raw))

{

$email = $email\_raw;

}else{

//---------------------------------------------------------------------------IF INVALID EMAIL THEN----------------------------------------------------------------------SET INVALID EMAIL 104 FLAG

$bad\_email=3143;

}

//email unique?

$sql="SELECT \* FROM users WHERE email='$email'";

$result=mysql\_query($sql);

$count=mysql\_num\_rows($result);

if($count>0){

//---------------------------------------------------------------------------IF SQL FOR EMAIL RETURNS A ROW THEN------------------------------------------------SET EMAIL 104 FLAG

$email\_already\_in\_use=3144;

}

//Secure Password Format Checks

$pw\_clean = strip\_tags(substr($\_POST['password'],0,32));

if (preg\_match("/^.\*(?=.{4,})(?=.\*[0-9])(?=.\*[a-z])(?=.\*[A-Z]).\*$/", $pw\_clean, $matches)) {

}else{

//---------------------------------------------------------------------------IF PW NOT IN FORMAT THEN-----------------------------------------------------------------SET PW 104 FLAG

$pw\_insecure = 3145;

}

//---------------------------------------------------------------------------IF ERROR FLAGS ARE SET THEN LOG HEADERS----------------------------

if($username\_already\_in\_use==3141 OR $email\_already\_in\_use==3144 OR $pw\_insecure==3145 OR $bad\_email==3143 OR $username\_too\_short==3142){

header(

"location:user\_add\_errors.php?pw\_insecure=$pw\_insecure&email\_already\_in\_use=$email\_already\_in\_use&username\_already\_in\_use=$username\_already\_in\_use&bad\_email=$bad\_email&username\_too\_short=$username\_too\_short");

die();

}

else {header("location:user\_add\_errors.php?noERROR=$noERROR");}

//End Error Checks\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//-------------------------------------------------------------------INSERT INTO SQL

//Encrypt Password

$encrypted\_pw = md5($pw\_clean);

$query = "INSERT INTO `users` (`username`,

`password`,

`lastname`,

`firstname`,

`email`,

`phone`,

`password\_hint`,

`udid`,

`userCreated`)

VALUES

(

'$username',

'$encrypted\_pw',

'$lastname',

'$firstname',

'$email',

'$phone',

'$password\_hint',

'$udidposted',

now())";

// save the info to the database

$results = mysql\_query( $query );

// print out the results

if( $results )

{

if($username\_too\_short==3142){echo"ShortUser=".$username\_too\_short;}

if($username\_already\_in\_use==3141){echo"UserTaken=".$username\_already\_in\_use;}

if($email\_already\_in\_use==3144){echo"EmailTaken=".$email\_already\_in\_use;}

if($pw\_insecure==3145){echo"ShortPass=".$pw\_insecure;}

if($bad\_email==3143){echo"BadEmail".$bad\_email;}

//echo( "<font size='2' face='Verdana, Arial, Helvetica, sans-serif'>Your changes have been made sucessfully. <br><br><a href='login.php'>Back to login</a></font> " );

}

else

{

die( "Trouble saving information to the database: " . mysql\_error() );

}

//--------------------------------------

$sql="SELECT \* FROM users";

$result=mysql\_query($sql);

$count=mysql\_num\_rows($result);

if($count==1){

$query = "UPDATE `users` SET `permissions`='5' WHERE `email`='$email'";

//---------------------------------------SAVE the info to the database

$results = mysql\_query( $query );

//---------------------------------------JUST PRINT CODE

if( $results )

{ echo( "ADMINCREATED" );

}

else

{

die( "ERRORSAVINGADMIN" . mysql\_error() );

}

}

?>

We won’t review all of what these files do. You can download them here\_\_\_\_\_! But checklogin basically queries the database for the user and password submitted. Whereas user\_add\_save.php verifies the user and password submitted for registration, as well as the email, and then returns a result accordingly.

You must configure your database, user and password accordingly. This is done in the login\_config.php file. We wont go too much into the php files but another important one is the user\_add\_errors.php. This file takes care of any errors during registration such as too short a username or too simple a password or invalid email or user already exists.

Great! Now we can start putting the pieces together, see you in the next part!