Computer Science Program

4th Semester 2016

Semester Project

Developing iOS P.O.C.A application

The report has been prepared by :

Alexandru Draghi

Ionut Danci-Bumbea

Richard Retezi

Supervisor:

Per Trosborg

Date of submission:

06-06-2015

Real Life Education

Contents

1. Introduction

POCA means Professional Oriented Communication Application. POCA wants to be an application that brings together people from with different backgrounds to work and create, to learn and have fun or just to meet experts around the world. POCA wants to bring together all the people from LinkedIn using an interface as simple as Tinder.

At the moment there is no easy way for people to search for want they need, when they need. Usually you can find freelancer or people with a specific kind of skills that are very far away or very expensive. POCA wants to solve this problem introduction the fun part in the equation. Around the world we can find a lot of people that want to share want they know, free or for cost. There was no way to do that, until now.

1.1 Problem statement

1.1.1 <u>Vision</u>

FOR people who share the same passion, WHO NEED to connect with like-minded thinkers, POCA IS A mobile app THAT provides the ultimate platform for connecting young professional thinkers. UNLIKE forums, social networks and mainstream communicating platforms, POCA OFFERS an easy and fast way to search and find people within the area of study and communicate with them.

1.1.2 <u>Sub-statement</u>

The problem with already existing applications is that their databases are huge and there is hard to find exactly what you are looking for. POCA should fix this problem. As an example: LinkedIn website is hard to use if you, as a user want to find future associates for your business and it is a requirement that they live nearby.

1.1.3 <u>Method</u>

In POCA we decided to make the user interface in a way that it will list the nearest users (using GPS and localization) with the same passions. The user can choose to look at their profile, add them to connections and start chatting with them once they accept the connection request.

2. System development

2.1 *Domain model*

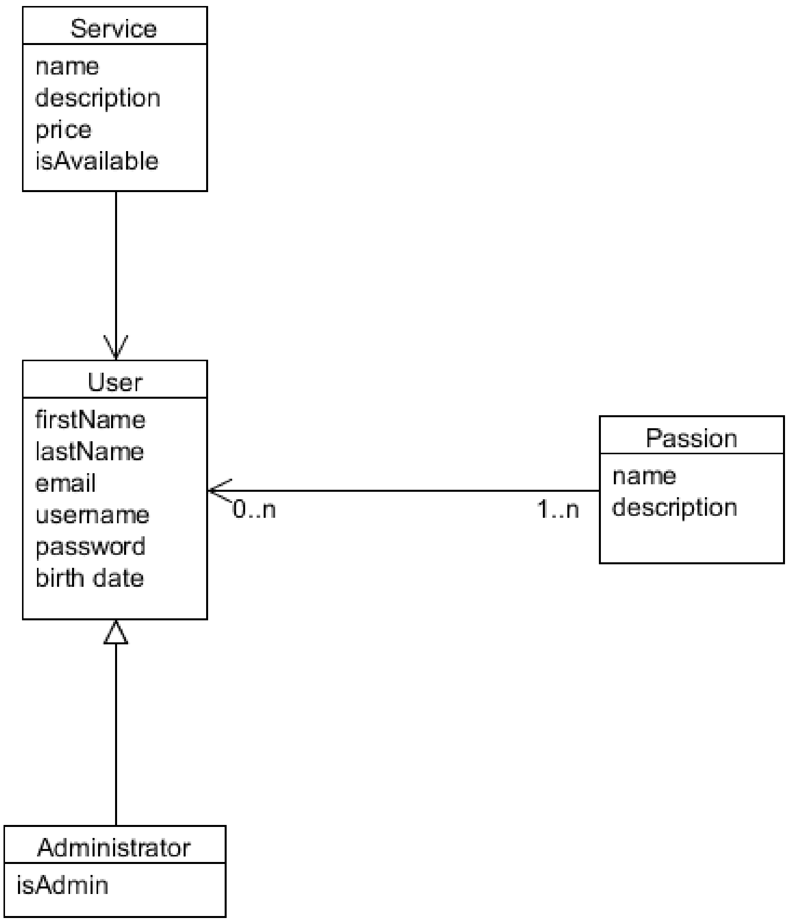
Domain Modelling is a way to describe and model real world entities and the relationships between them, which collectively describe the problem domain space.1

A domain model consists of domain classes, attributes for every class and associations them.

The domain model is one of the most important parts of a system because it describes the way the system works and it's the base of the architecture. The domain model shows how all system is connected within itself and with other systems.

In our case we have users, passions, services and administrators.

Every user can have one or more passions and a passion can be selected by none or more users. On our website version for this application a user can select if he is administrator and then an already existing admin can accept his request or deny it. A user can also book assistance online using the web version of the application.



2.2 *Relational model*

A relational table represents the relations between tables and fields inside a database.

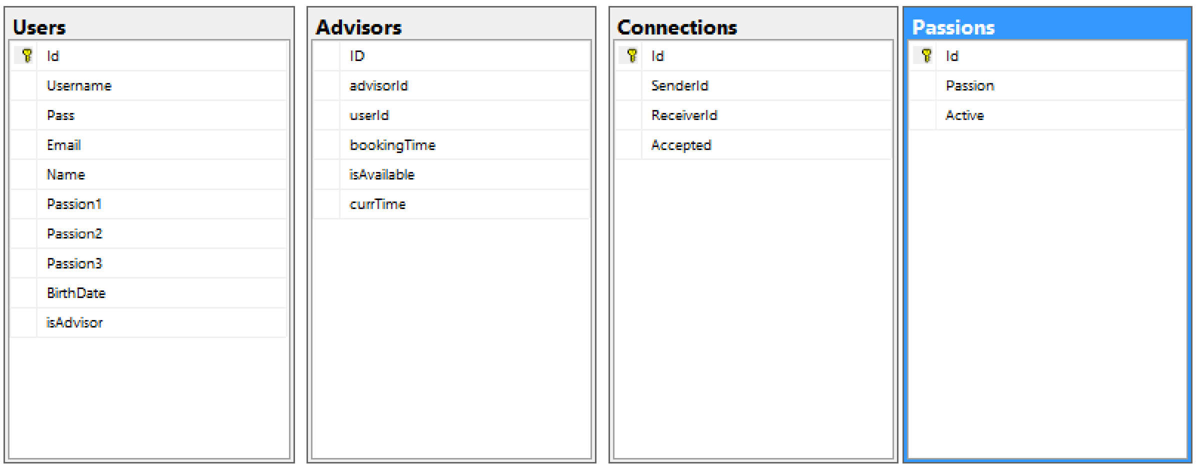
In our case we have 4 tables.

The first table is Users and it saves everything that is related to users. His username, password, email and name as well as passion ids. Every passion is represented by the id in the Passions table.

If a user is also an Advisor (available only in web version) then he can also advice users about different subjects. Users can book a meeting with an advisor at any booking time set by the advisor. A list is available on the website.

The Connections represents every connection of a User. The SenderId is the id of the user that is sending the request, the ReceiverId is the id of the user that gets invited. Accepted represents if the users are connected or not (0 and 1).

The Passions table represents a list with all the passions available for the user as well as if they are available for selection or not.



Users	Advisors	Connections	Passions
Id	ID	Id	Id
Username	advisorId	SenderId	Passion
Pass	userId	ReceiverId	Active
Email	bookingTime	Accepted	
Name	isAvailable		
Passion1	currTime		
Passion2			
Passion3			
BirthDate			
isAdvisor			

3. Implementation

3.1 System Architecture

For the architecture we tried to find a way to show everything we wanted in a view without asking too much from our users. From personal experiences we know that a user will not spend more than a few seconds trying to figure out how something works. Making the interfaces as simple as possible is a step forward in gaining that user's trust and making him continue using this app.

We want something complex but easy to use. We want to have the best application of this kind that puts the power in our users' hands.

Privacy is a big issue for our users and we have to protect how we can better. This application is designed to ask only for very basic information from the every users. The only requirement that we have as an extra is sharing locations.

The system is constructed in such a way that every window can be access from a specific point making navigation easy.

The first window that opens is the Login screen if that user is not logged in or the Search windows if opposite. From the login window there is the possibility to create an account or to recover your password.

3.1.1 Web

In our previous project we have created a website using c#, asp.net and html. The website was created using Visual Studio and all the tools it offers. Besides that we used Azure to store the website and the WCF Service, as well as the database.

The initial decision was to create a mobile application for Windows Phone, unfortunately the WCF Service was not compatible with phone so we had to create a website. Now we have the chance to create the mobile application and link the together for a better user experience. And this is what we have done.

3.1.2 Mobile

We had to choose between Windows Phone OS, Android and iOS. After checking every platform we come to a conclusion that we could learn more from developing for iOS. The windows phone platform is basically unknown and there is no future promised for it. For now at least we decided not to develop application for Microsoft's OS.

The second choice is Android. To develop for android we would have used the Android Studio IDE and JAVA as programming language. We have been studying java for 1 year and we certainly wanted something new, something fresh to learn.

The last but now least is iOS with SWIFT programming language. Since this is a new programming language, with millions of devices around the world capable of running the application is clear that this should be our choice. Swift is just a better version of Objective-C, the old programming language used for developing iOS applications.

3.1.3 WCF service

Windows Communication Foundation (WCF) is a framework for building service-oriented applications. Using WCF, you can send data as asynchronous messages from one service endpoint to another. A service endpoint can be part of a continuously available service hosted by IIS, or it can be a service hosted in an application2.

An endpoint can be a client of a service that requests data from a service endpoint.

In this project the WCF service is used to call the database every time a user is logging in, logging out, registering and so on. The WCF service is hosted on Azure and it's connected to the website as a reference. The WCF service has also access to the database used to store all the needed information.

We used WCF service in our previous project because it was a requirement but since the mobile applications are hardly compatible with it we had to find an alternative solution. The best solution we could come up with is using a PHP script on a server end-based application. This way we can transfer data between our iOS application written in Swift and the SQL database.

3.1.4 PHP

PHP is a script language and interpreter that is freely available and used primarily on Linux Web servers. PHP, originally derived from Personal Home Page Tools, now stands for PHP: Hypertext Preprocessor, which the PHP FAQ describes as a "recursive acronym."3. We are using PHP scripts in order to connect the application to the SQL database.

We are using the post method from our iOS application to send the data to the server where the PHP takes it and writes a query based on the information received from the application. The PHP then executes the query and writes the response in a single string that is later send to the application.

Considering that on Azure free account the transfer speed is really slow and the application always timed out we had to store the files on a private paid hosting account. The code is split in several files in order to have a good management of the methods and to keep the functionality separated.

3.1.5 Database

SQL stands for Structured Query Language. SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems.4

At first we tried to host the database on Azure servers but unfortunately the transfer speed is really slow and because we found a free hosting for the database we decided to try this way. Hosting the database on this free database hosting server it was a success so we kept the database here.

The database for our project consists of four tables (Users, Passions, Connections, Advisors) and it's used for both application and website.

The "Users" table has ten different columns including: "Username", "Password", "Email", "Name" (as a full real name), "Passion1", "Passion2", "Passion3" and "isAdvisor" (This is an int value 0/1 which represents if the user is an advisor or just a simple user).

The "Passions" table has only three columns: "Id","Passion" and "Active". The "Active" field it's an int value 0/1 which represents if the passion is active in the system or not.

The "Connections" table has four columns: "Id","SenderId","ReceiverId" and "Accepted"(This column can have only 2 values 0/1, 0 in case the connection request is sent but not accepted or 1 if it is already accepted).

The "Advisors" table has six columns: "Id", "advisorId", "userId", "bookingTime" (the time when the advisor it's available), "isAvailable" (when someone choose an hour that is available for the booking this is setted to 0 if the person does not finish the booking or 1 if the booking is finished), "currTime" (Based on the currTime we lock an available booking for five minutes when someone click it. If the customer does not finish the booking and if more than five minutes pasted the booking for this hour is available again).

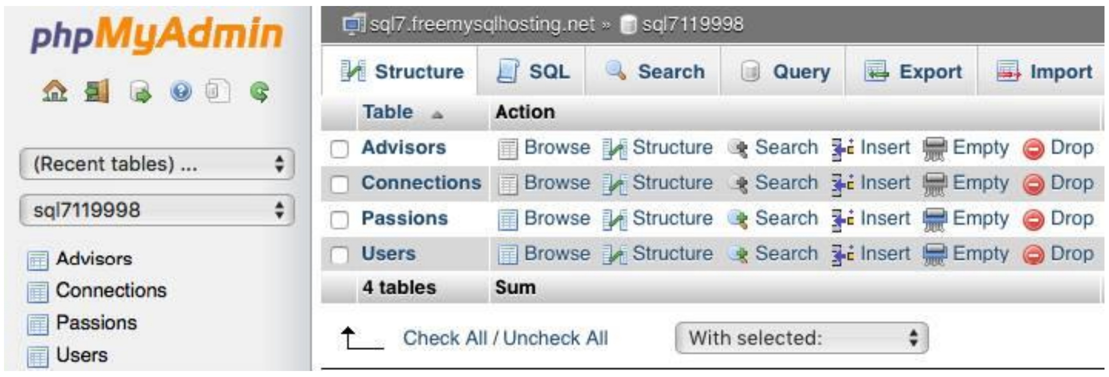
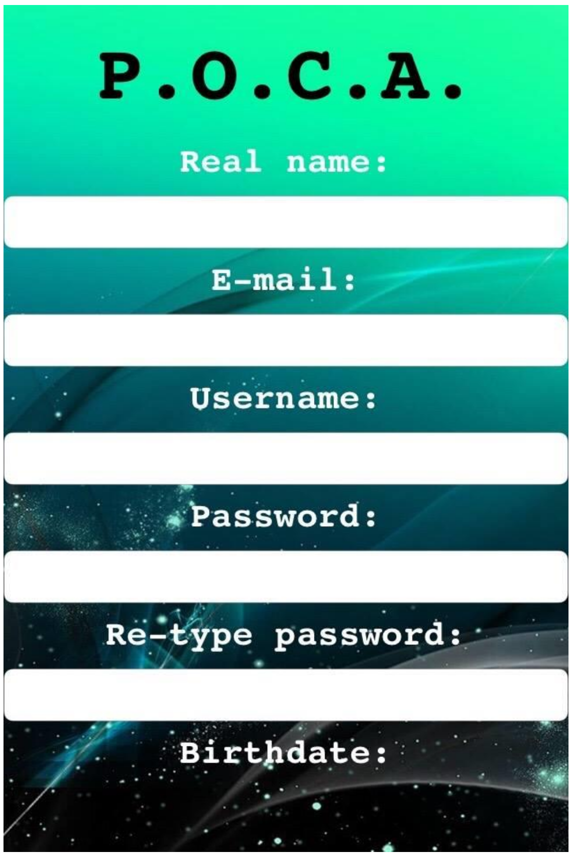


	Table	Action					
☐	Advisors	Browse	Structure	Search	Insert	Empty	Drop
☐	Connections	Browse	Structure	Search	Insert	Empty	Drop
☐	Passions	Browse	Structure	Search	Insert	Empty	Drop
☐	Users	Browse	Structure	Search	Insert	Empty	Drop
	4 tables	Sum					

3.2 Interface and design

Since we are talking about a mobile application the interface needs to be simple. It has to be intuitive and easy to use. People hate when they get lost in a lot of information or have too many form to fill in. We ask from our users exactly what we need and nothing more.

Register


P.O.C.A.

Real name:

E-mail:

Username:

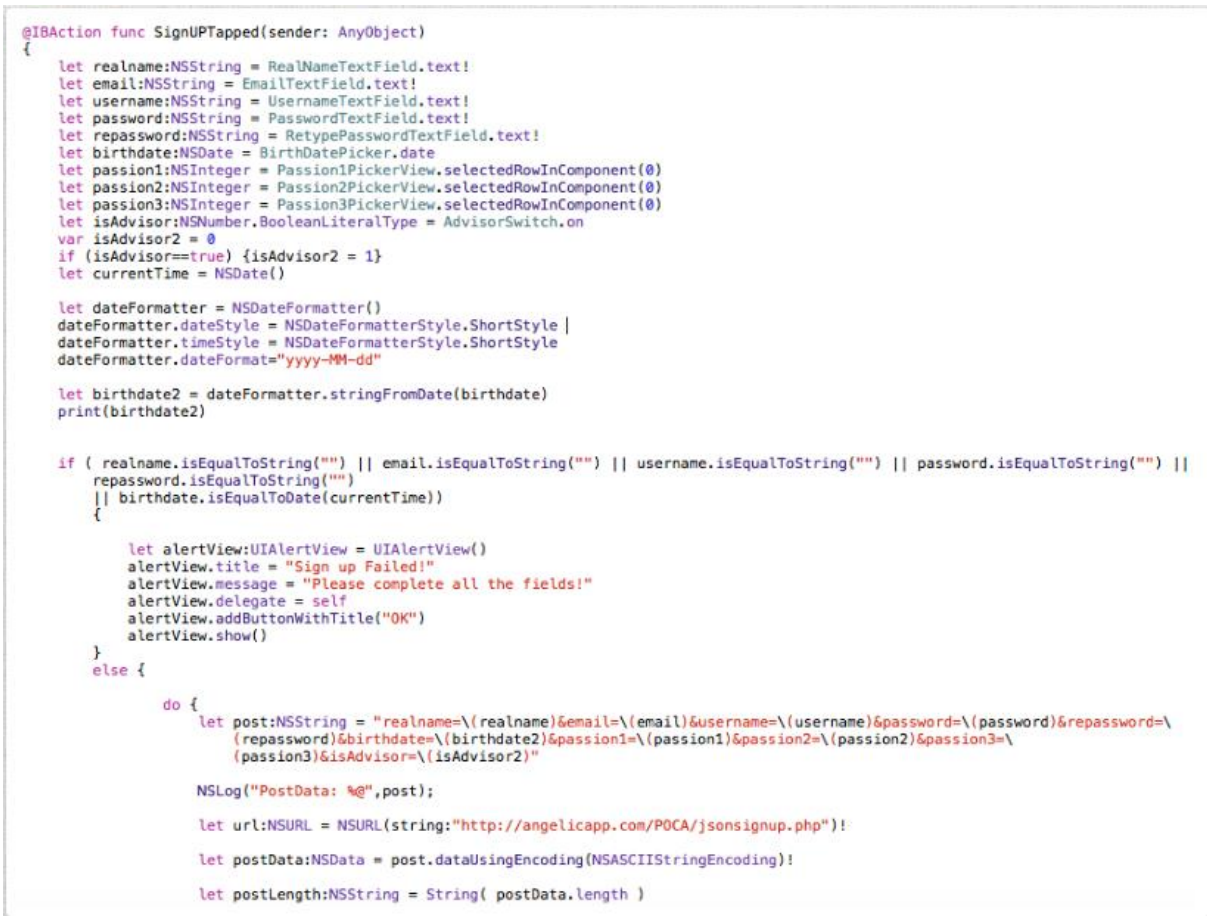
Password:

Re-type password:

Birthdate:

The register view represents the way a user creates an account to be able to use our application. In this view every user is requested to fill in a few forms to describe himself and his passions. The requested data is basic data as username, e-mail and so on. What we are interested on are 3 passions.

We decided to limit the access to only 3 passions, this way every user has to think about it and to choose wisely. This passions can be updated an anytime but a user will feel obligated to put the most relevant data first.



```
@IBAction func SignUPTapped(sender: AnyObject)
{
    let realname:NSString = RealNameTextField.text!
    let email:NSString = EmailTextField.text!
    let username:NSString = UsernameTextField.text!
    let password:NSString = PasswordTextField.text!
    let repassword:NSString = RetypePasswordTextField.text!
    let birthdate:NSDate = BirthDatePicker.date
    let passion1:NSInteger = Passion1PickerView.selectedRowInComponent(0)
    let passion2:NSInteger = Passion2PickerView.selectedRowInComponent(0)
    let passion3:NSInteger = Passion3PickerView.selectedRowInComponent(0)
    let isAdvisor:NSNumber.BooleanLiteralType = AdvisorSwitch.on
    var isAdvisor2 = 0
    if (isAdvisor==true) {isAdvisor2 = 1}
    let currentTime = NSDate()

    let dateFormatter = NSDateFormatter()
    dateFormatter.dateStyle = NSDateFormatterStyle.ShortStyle |
    dateFormatter.timeStyle = NSDateFormatterStyle.ShortStyle
    dateFormatter.dateFormat="yyyy-MM-dd"

    let birthdate2 = dateFormatter.stringFromDate(birthdate)
    print(birthdate2)


    if ( realname.isEqualToString("") || email.isEqualToString("") || username.isEqualToString("") || password.isEqualToString("") ||
        repassword.isEqualToString("")
        || birthdate.isEqualToDate(currentTime))
        {

            let alertView:UIAlertView = UIAlertView()
            alertView.title = "Sign up Failed!"
            alertView.message = "Please complete all the fields!"
            alertView.delegate = self
            alertView.addButtonWithTitle("OK")
            alertView.show()
        }
        else {

                do {
                    let post:NSString = "realname=\(realname)&email=\(email)&username=\(username)&password=\(password)&repassword=\
                        (repassword)&birthdate=\(birthdate2)&passion1=\(passion1)&passion2=\(passion2)&passion3=\
                        (passion3)&isAdvisor=\(isAdvisor2)"

                    NSLog("PostData: %@",post);

                    let url:NSURL = NSURL(string:"http://angelicapp.com/POCA/jsonsignup.php")!

                    let postData:NSData = post.dataUsingEncoding(NSASCIIStringEncoding)!

                    let postLength:NSString = String( postData.length )
```

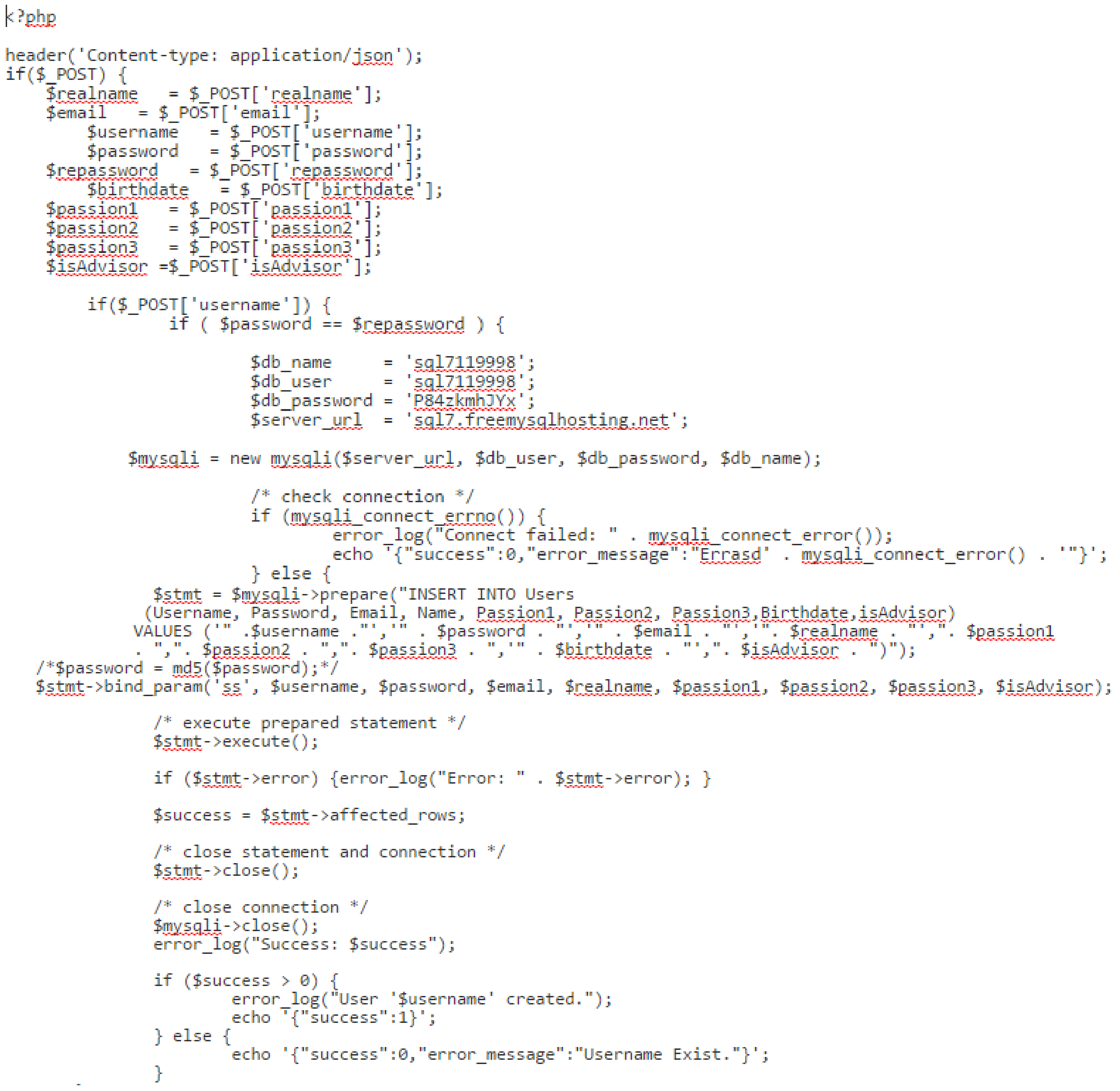
When pressing the register account button a series of checks are made. If the username exists, if the passwords are identical and many more. If any of the checks fails a pop-up will be shown on the screen informing that user about the problem.

We start by asking the user to fill in the information and then when the register button is pressed we collect the data, store it in variables. We process all the data in order to fit in the database, for example we remove the time from the date picker and we only store the date itself. In order to display the passions we had to create the delegates and data sources for every one of the three picker views.

Before the data is sent to the server we check if the information written in every field is accepted. If any field was left empty a pop-up up alert will warn the user.

Otherwise we create a post message that sends all the information to the server as a single string, every variable getting a new name.

A connection to the server is created. The type is NSURL. Every error is caught and shown on the screen. After the data gets processed we get back a response with a success or failed message. If succeeded the new user is stored in the database and he can now use it to login.

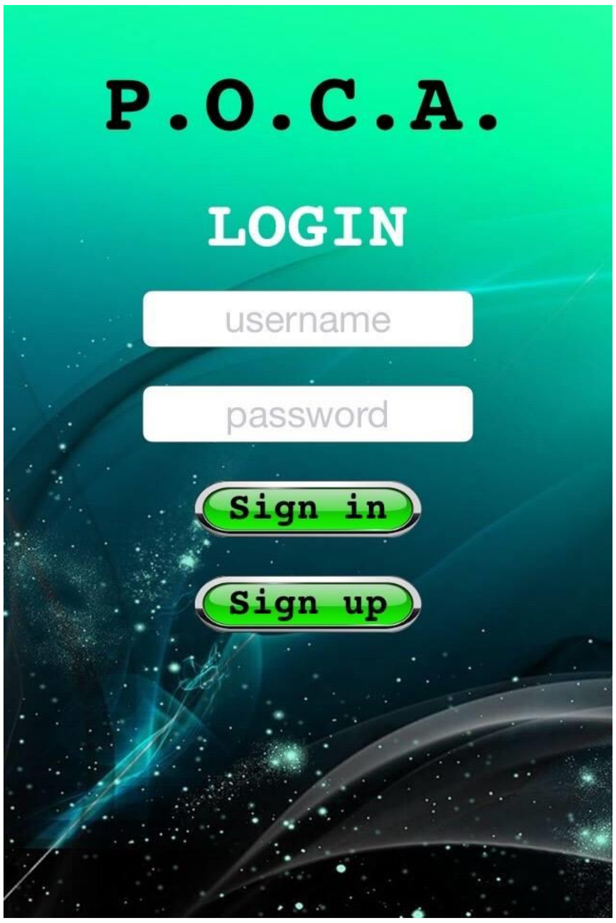


```php
<?php

header('Content-type: application/json');
if($_POST) {
    $realname   = $_POST['realname'];
    $email    = $_POST['email'];
        $username    = $_POST['username'];
        $password    = $_POST['password'];
    $repassword    = $_POST['repassword'];
        $birthdate    = $_POST['birthdate'];
    $passion1    = $_POST['passion1'];
    $passion2    = $_POST['passion2'];
    $passion3    = $_POST['passion3'];
    $isAdvisor =$_POST['isAdvisor'];

        if($_POST['username']) {
                if ( $password == $repassword ) {

                        $db_name     = 'sql7119998';
                        $db_user     = 'sql7119998';
                        $db_password = 'P84zkmhJYx';
                        $server_url  = 'sql7.freemysqlhosting.net';

            $mysqli = new mysqli($server_url, $db_user, $db_password, $db_name);

                        /* check connection */
                        if (mysqli_connect_errno()) {
                                error_log("Connect failed: " . mysqli_connect_error());
                                echo '{"success":0,"error_message":"Errasd' . mysqli_connect_error() . '"}';
                        } else {
               $stmt = $mysqli->prepare("INSERT INTO Users
              (Username, Password, Email, Name, Passion1, Passion2, Passion3,Birthdate,isAdvisor)
             VALUES ('" .$username ."','" . $password . "','" . $email . "','". $realname . "',". $passion1
             . ",". $passion2 . ",". $passion3 . ",'" . $birthdate . "',". $isAdvisor . ")");
   /*$password = md5($password);*/
   $stmt->bind_param('ss', $username, $password, $email, $realname, $passion1, $passion2, $passion3, $isAdvisor);

               /* execute prepared statement */
               $stmt->execute();

               if ($stmt->error) {error_log("Error: " . $stmt->error); }

               $success = $stmt->affected_rows;

               /* close statement and connection */
               $stmt->close();

               /* close connection */
               $mysqli->close();
               error_log("Success: $success");

               if ($success > 0) {
                       error_log("User '$username' created.");
                       echo '{"success":1}';
               } else {
                       echo '{"success":0,"error_message":"Username Exist."}';
               }
```

On the server side we have a php code that is directly connected to our swift code based application. After sending the data from the inside of the application to the server, we store and split the string into variables.

We try to connect to our database, and, if we succeed then we execute a query that tries to store the new data into the database. If succeeded the data is saved, otherwise an error state is send to the phone and the error statement is called.

Login

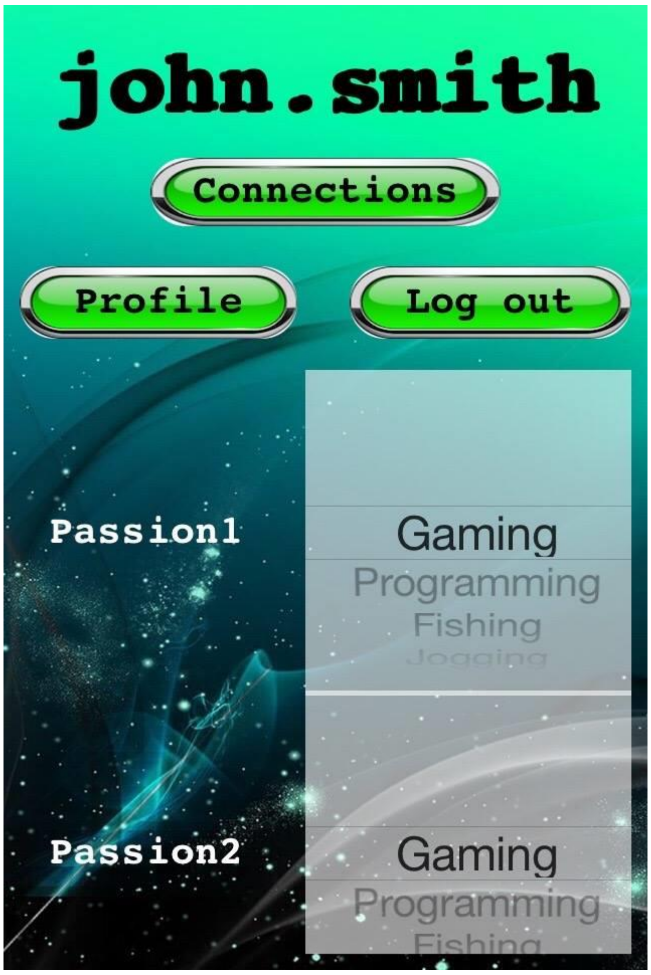


This is just a basic view with two fields, a button and a create account link. Here every user can login to use the features of this app.

After pressing the "Sign in" button the data is send to the server and checked to see if that users exists. If yes, the password is checked. For any error a pop-up is shown to inform the user.

If a user doesn't have an account or if he want to create a new one he can use the register link ("Sign up") that will show him the Register view described above.

Search



This represents the home view once the user is logged in. From this view the user can go in any other window except Register. He can log out (obviously), go to connections (to be explained), his profile (to be explained) or search for users or advisors that he is not already connected.

John.smith in this case is a label that changes his value to the username of the account that is logged in. This is there only for you to be sure that you are logged in the correct account in case you have more than one account. Starting from here everything else it’s included into a scroll view in order to be able to see the views that will be later created by the search button.

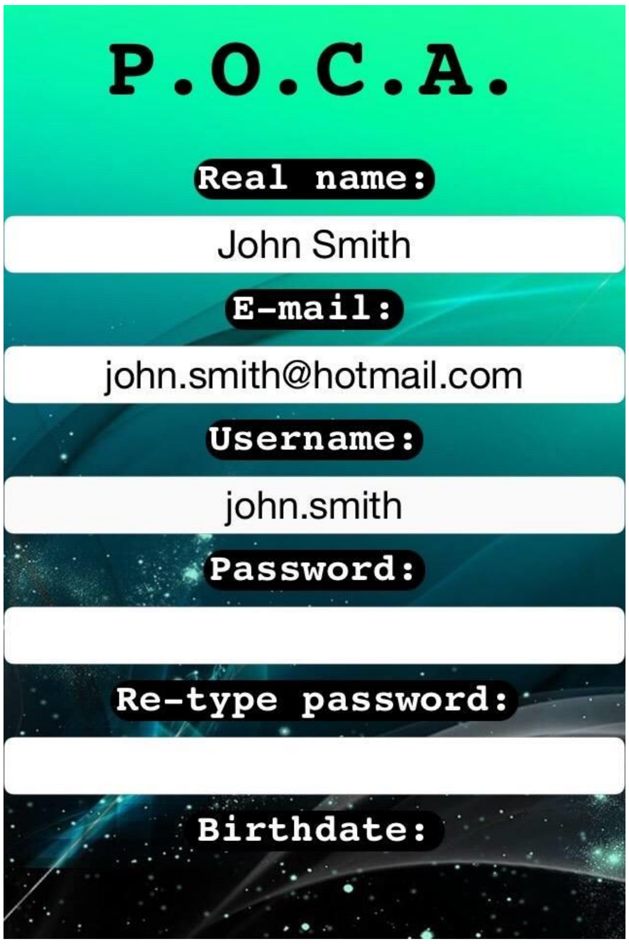
Connections, Profile and Log out are buttons that use different methods. The Connections button sends you to the Connections controller where you will be able to see your connection and in case you have any request you will be able to accept or decline them. The Profile button sends you to the Profile Controller where you are able to Update your profile information except from “username” and the status of your account (if you are an advisor or not).

The Log out button disconnects your account from the program so if you want to give the phone to someone else he can connects into his account.

Then we have 3 different labels one for each passion positioned in the centre of the passion picker view so you know which one picker view corresponds to its passion.

Then there is the search button that dynamically creates view for each of the user that has at least one of the passions that you selected.

Profile


P.O.C.A.

Real name:

John Smith

E-mail:

john.smith@hotmail.com

Username:

john.smith

Password:

Re-type password:

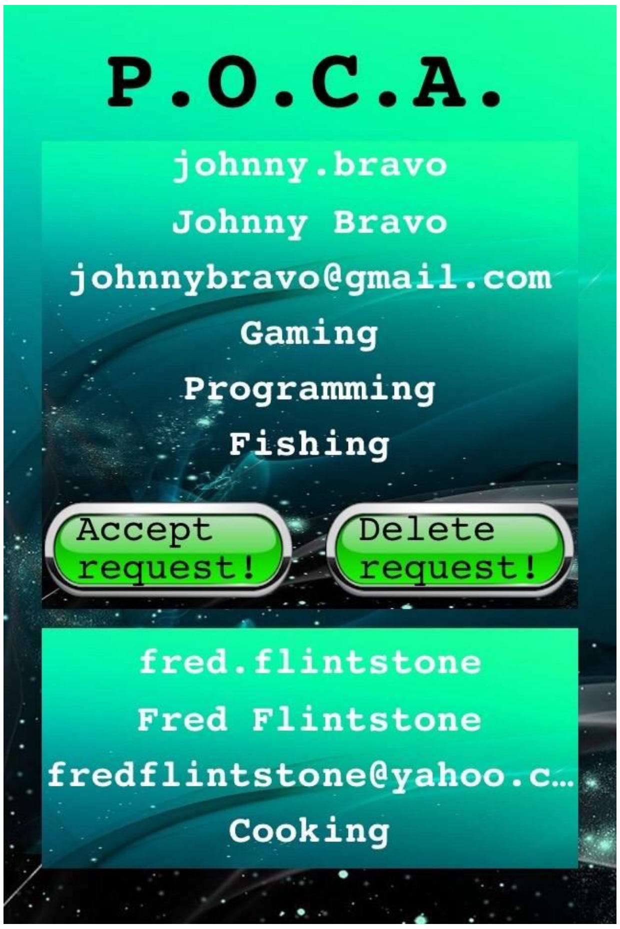
Birthdate:

In this view a user can update all the information about his profile except his username and his advisor status that must stay the same.

Everything else can be changed at any given moment. The data is exactly the same as the one in the register view except the isAdvisor option.

This way a user can update his e-mail address, his passions or his passwords.

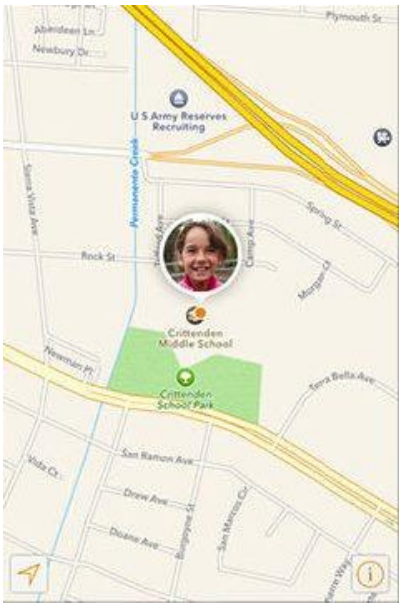
Connections



From here every view includes: The first label represents the username (john.smith), the second one represents the real name (John Smith), the third one represents the e-mail (john.smith@hotmail.com) and then there are the 3 labels for the passions.

The “Accept request!” button sets the connection status in the database as completed while “Delete request!” deletes the connection from the database.

Location



POCA offers a way to see all the users next to you on a map inside the app. This map shows the users in their last share position. This position is shared automatically when he/she logs in and an internet connection allows this. The access for this feature is done from the Search view via the Locations button.

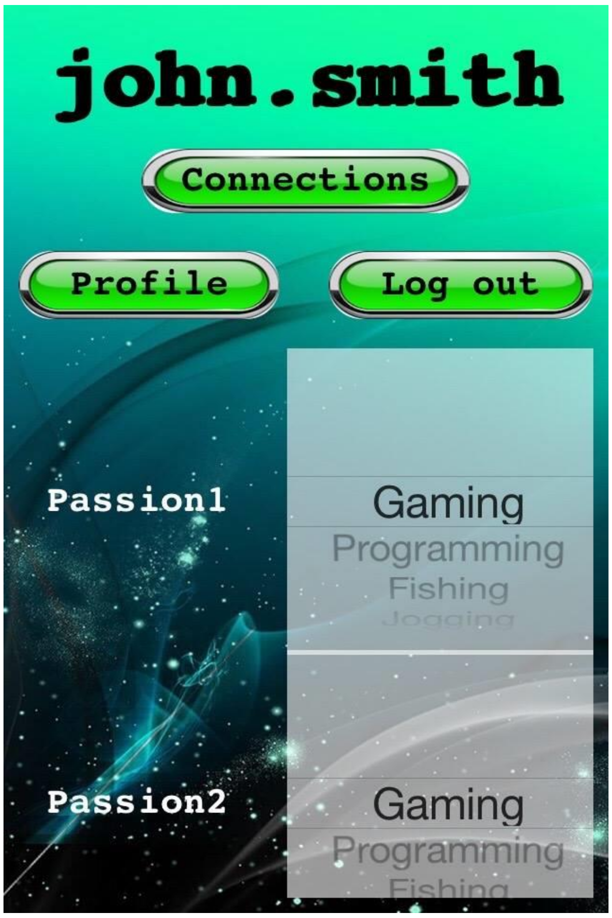
Every user that can be shown will be represented by a small dot on the map. The map is not showing their real names or any other information to protect that person's identity.

The purpose of the map is to inform a user about his options. More dots on the map means more connections to choose from therefore more possible future collaborations.

To display the map we implemented Apple Maps and we are using Apple Maps API to control the dots on the map and update the location.

Inside the map view there is a setting to disable Location Sharing. This way the user has control over his personal data, he can control when to share and with whom.

Log out



A simple button that disconnects the user from the app and logs him out.

3.3 Problems and errors

The biggest problem, by far, was the very slow database connection to azure SQL database. We had the web version of POCA connected to azure and everything worked well. Even though the connection was slow a login was still possible. A login took almost 30 seconds on web. Once we've connected the iOS app to the azure database the only response we got was: "Connection timed out".

We've tried to modify our scripts, optimize the code but nothing worked. The only solution left was replacing the hosting for our database. We replaced azure with freemysqlhosting.net and everything worked well.

In swift there is no default way to connect to a SQL Database. We had to come up with a fast solution if we wanted this project to be done in time. After a lot of searching we found a great solution. Swift can be connected to SQL via php. A php is called and asked to execute a query that returns information as a string. Then we use that information to show the data on screen.

Also in swift if you create an outlet for a label (example) and then you delete it only from one side (code or interface) but not both, it gives you an error at runtime. Plus the constrains was a pain at the beginning but after we found out that Xcode can actually set your constrain automatically it was easier.

The scroll view is automatically resizing only if you create the content from the interface but if you want to add something dynamically then you will have to increase the scroll view size also otherwise you will not be able to see all the content that was created.

Also you cannot be sure that if something works well in the simulator will work on the phone too because sometimes the collider of the buttons remains the same even if the text is made smaller and because of that some of the buttons will appear over the other ones and the second ones will be impossible to touch.

3.4 Learning approach

The three members of our group have taken different paths this semester.

The team has different set of skills, and have the variety of different knowledge can help immensely in a team work and in order to deliver a project.

The goal was that each of us selects a path, specialization that is in his interest and helps him in his further development as well as helps us to become experts and to create a great project.

Alexandru and Ionut are more like programmer types and they love coding, while Richard is more interested in the IT Business and System Design segment of IT. We believe this is a great combination of a team to enforce each other in order to work well.

In the implementation of the POCA app, the Team has learnt how to adapt to unexpected errors and obstacles, how complex can even the simplest mobile application be, how the different "parts" (database, server, user interface, maps, search, other functional codes, etc.) of an application work together and connect to each other.

In this project the Team tried to learn from past mistakes, correct errors and find new ways to solve the challenges. On the basis of what we have learned about the mobile applications, databases, servers, WCF, We used our experience to enhance our previous project as well as making it more complex and also adding new layers and reaching new learning outcomes (such as php, swift and experience in the participation of real-life business cases and projects).

We believe the Team was able to achieve the goals that were set for this semester and gain lots of new experiences in order to become professionals in the IT area.

4. Student profiles

4.1 Alexandru

Fourth semester as a student at UCN:

In this semester we studied at school different programs. Some of them was new (like Android Studio, Xcode) and some was just advancing into them (like Visual Studio). At ASP.Net course we learned how to host your webpage on Azure, new ways to do a better design and new things like processing images via code (resizing, adding colour effects and other things).

At the Android course we learned the basics of how to create an Android app in Android Studio and also how to create an Android developer account in order to publish our app on the Google PlayStore. At iOS of course it was a total new experience because it was a new program and also a new programming language (SWIFT). Swift is a modern programming language that came in order to help the developers for iOS that considered Objective-C as hard to use.

Xcode and iOS:

The iOS course was a really good experience. At the beginning it was harder because if you remove an outlet from code you have to remove it from the interface to. Also if you have 2 outlets at the same object there is going to give a runtime error. Managing the constraints was a really pain before I found that you can use Xcode methods in order to do that by only pressing 2 clicks on the right buttons.

P.O.C.A as an iOS APP:

Because we like the challenges we decided to make an iOS application for our last project. At the beginning we tried to connect our app directly to the WCF service but we found out that this is a really painful thing and that it’s much easier to use server side methods (PHP) in order to connect the program to the SQL database.

Since we did not learned PHP at school courses we took this as a challenge too and we decided to connect our program to the SQL database in this way. We learned how to use post methods in order to send parameters to the PHP script and then the script will query the database and save the results into a string that will be passed back to the client (iOS app in our case).

Another hard thing was to find out how to dynamically create views with every client that is registered in the database when you use the search methods or when you want to see your connections. We created a view model for the clients so when you present a client you create an empty model and just add the information in every label.

Since the scrollView does not know how to adapt its size for the content that is dynamically created we had to manage the script in that way that the scrollView size will be increased with the height of a view plus some space for every client that is registered and it's going to be presented after the search applies.

As this is a school project application and because we do not intend to publish it on the AppStore we did not focused on graphics as much as on functionality. I always considered that a programmer should know how to do the functionality and some simple design but the proper design in order to release an application on the AppStore or Google PlayStore should be created by a professional designer.

4.2 Ionut

Our application consists of an iPhone app created in Swift. Swift is a programming language created and developed by Apple inc. The iOS course consisted of 7 modules where we've learned the basics of Swift, xCode and the OSX operating system.

Every lesson was an introduction to a new subject. In the beginning we learned how to create a new project, how to add libraries. Then we went a step forward and we started to add objects into your controllers and views.

Speaking for myself, having a mac and being passionate about create mobile games, I had some experience with all of this. The hard part come in play when we started coding.

I had no idea how to use swift but slowly we discovered how to connect outlets, how to send information from one view to the other one using segues.

I was spending most of my time learning at home, I like to have quiet and no people around me when accumulating knowledge. In class I like to pay attention at what the teacher is saying, what he's showing, how he solved errors and bugs.

I like to try everything myself, reproduce all projects and solving all the errors. This way, when I encounter then in a real life project I know how to solve them.

Regarding the group, we meet a few times a week, either at school or at someone's house and we talked about that week's subject, what kind of problems we had, what we've tried new.

The learning experience was really pleasant and I really like talking about it.

4.3 Richard

I have decided to follow an alternative learning approach for this semester. I have received the possibility to work in the IT department of the biggest company of my home country, Hungary. The company is MOL GROUP, which is an Energy/Oil company, one of the key players in the Central Eastern Europe region.

I have joined the Office Applications Team, which as its name is referring to, deals with the corporate implementation of internal applications let it be purchase, licensed or self-developed.

I had the pleasure of getting insights into real-life projects and implementations, such as the roll-out of a Fuel Station Finder App, upgrading corporate phones from Windows Mobile 8.1 to Windows Mobile 10, migrating sites and documents from Microsoft SharePoint 2007 to 2013, creating a feasibility study to Office 365, etc.

My biggest task and learning experience during the semester was implementing an MDM (Mobile Device Management) solution for the company. The project was called EMM ("Enterprise Mobility Management") and it included the implementation of MobileIron's MDM solution.

The MDM solution basically means that the employees on their corporate mobiles can have a public and a corporate profile. On the public profile they can have their private apps and data. By using a second authentication they can access to the Corporate Profile where they can use corporate apps and open, edit corporate documents securely, without risking a leakage, virus or the theft of sensitive documents. If a mobile is stolen for example, the employee or the company's Help Desk (after notification of course) can 24/7 wipe out and remotely make a factory reset on the phone clearing it from the corporate profile and deleting every company data. In this way security of company mobiles is assured and it also helps the now very accurate mobility trend in order to create a modern, mobile workplace for employees.

I think this semester was very useful for me to work with real developers, real project managers on real projects, helping with the implementation of several applications. My main responsibilities were writing business cases (called IPPs -> Individual Project Proposals), making sure that the implementation is on track (discussing with developers, project managers and vendors), creating communication campaigns towards colleagues about the incoming new applications or updated, etc.

During this semester I have learned a lot, especially about planning and design which I believe is a great asset and helps a lot to my group in our project and implementation of the POCA app.

5. Conclusions

POCA was an idea for a modern, simple mobile application that in the meantime provides great challenges and deep complexity in the making. As the Team had discovered the criteria and requirements for the project, we were unknowingly heading towards and agile development which consisted of a dynamic approach in order to handle sudden risks.

As we progressed last semester, we were facing lots of challenges and obstacles because of previously unknown technical limitations. The Team had to make great effort in order to derive from the original plan and not to fail, but to deliver a great, complex and challenging project (this was to change our original scope from Windows Phone 8.1 to Web Application).

This semester, after learning from past errors and in order to enhance and further improve the functionality of our core idea app, we decided to finally develop the app as a mobile app and deploy it on iOS. For this, the Team had to learn basically two new languages for them (Swift and php), as well as making sure of the right connections to servers.

We do believe that this project was the perfect choice. It contained everything that an IT student needs in order become a competitive professional: Complexity, challenges, obstacles, technical problems, massive errors, finding new solutions, learning new codes and platforms, workarounds, etc.

We believe that the fact that we were able to make this app work is a great achievement and team effort, and we can proudly stand behind it.

The Team – on purpose – did not want to play securely and go for an easy, simple project and thus we have suffered a lot when we faced the connection problems, but we managed it and that was a great learning curve to all of us.

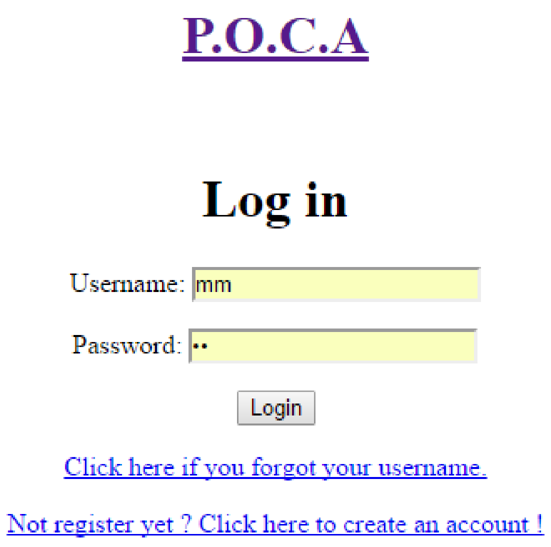
6. References

1. http://www.scaledagileframework.com/domain-modeling
2. https://msdn.microsoft.com/en-us/library/ms731082(v=vs.110).aspx
3. http://searchenterpriselinux.techtarget.com/definition/PHP
4. http://www.sqlcourse.com/intro.html

7. Appendix

Website: http://poca.azurewebsites.net/login.aspx

7.1 Log in



P.O.C.A

Log in

Username: mm

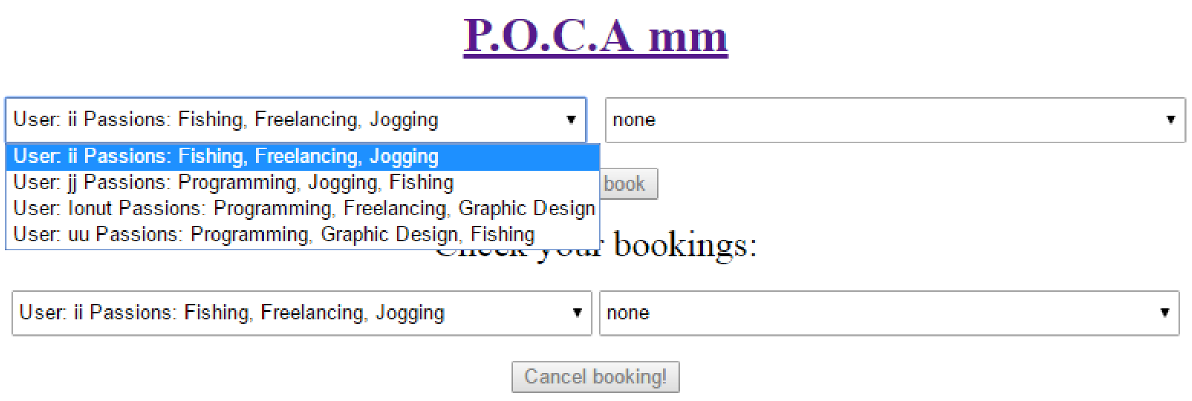
Password: ••

Login

Click here if you forgot your username.

Not register yet ? Click here to create an account !

7.2 Search



P.O.C.A mm

User: ii Passions: Fishing, Freelancing, Jogging

none

User: ii Passions: Fishing, Freelancing, Jogging
User: jj Passions: Programming, Jogging, Fishing
User: Ionut Passions: Programming, Freelancing, Graphic Design
User: uu Passions: Programming, Graphic Design, Fishing

book

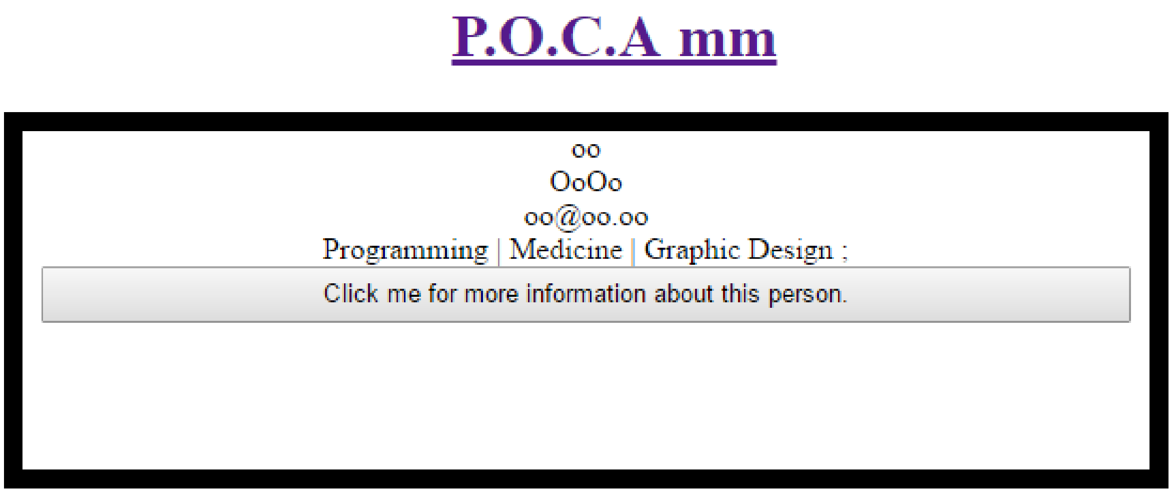
Check your bookings:

User: ii Passions: Fishing, Freelancing, Jogging

none

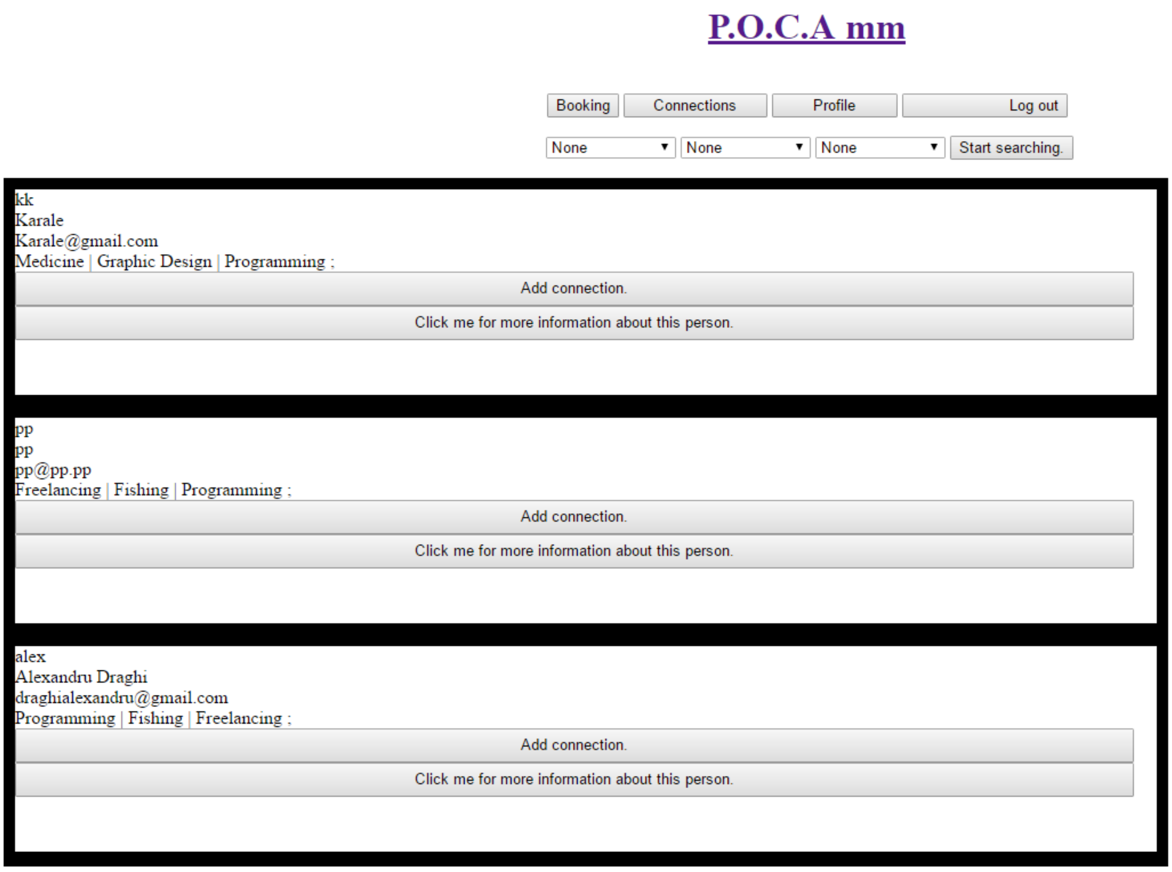
Cancel booking!

7.3 Connections


P.O.C.A mm

oo
OoOo
oo@oo.oo
Programming | Medicine | Graphic Design ;

7.4 Search results


P.O.C.A mm

Booking
Connections
Profile
Log out
None
None
None
Start searching.

kk
Karale
Karale@gmail.com
Medicine | Graphic Design | Programming ;

Add connection.
Click me for more information about this person.

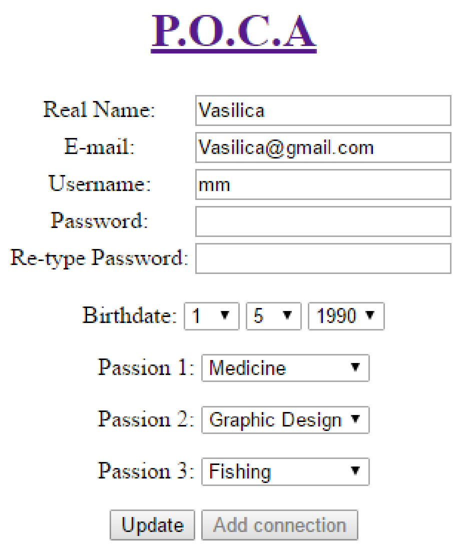
pp
pp
pp@pp.pp
Freelancing | Fishing | Programming ;

Add connection.
Click me for more information about this person.

alex
Alexandru Draghi
draghialexandru@gmail.com
Programming | Fishing | Freelancing ;

Add connection.
Click me for more information about this person.

7.5 Update profile



P.O.C.A

Real Name:	Vasilica
E-mail:	Vasilica@gmail.com
Username:	mm
Password:	
Re-type Password:	

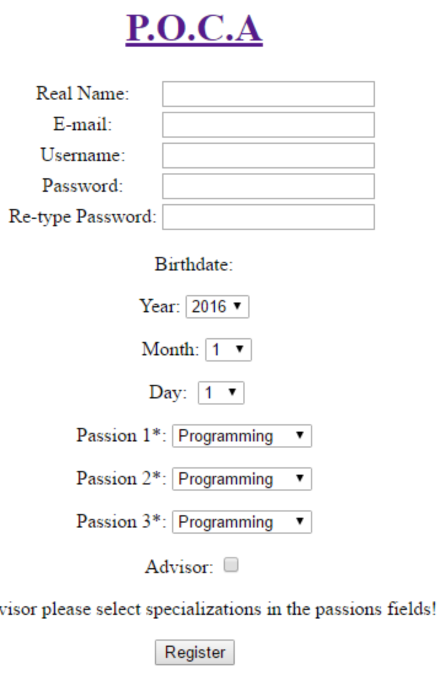
Birthdate: 1 5 1990

Passion 1: Medicine

Passion 2: Graphic Design

Passion 3: Fishing

7.6 Register



P.O.C.A

Real Name:	
E-mail:	
Username:	
Password:	
Re-type Password:	

Birthdate:

Year: 2016

Month: 1

Day: 1

Passion 1*: Programming

Passion 2*: Programming

Passion 3*: Programming

Advisor: ☐

*If advisor please select specializations in the passions fields!