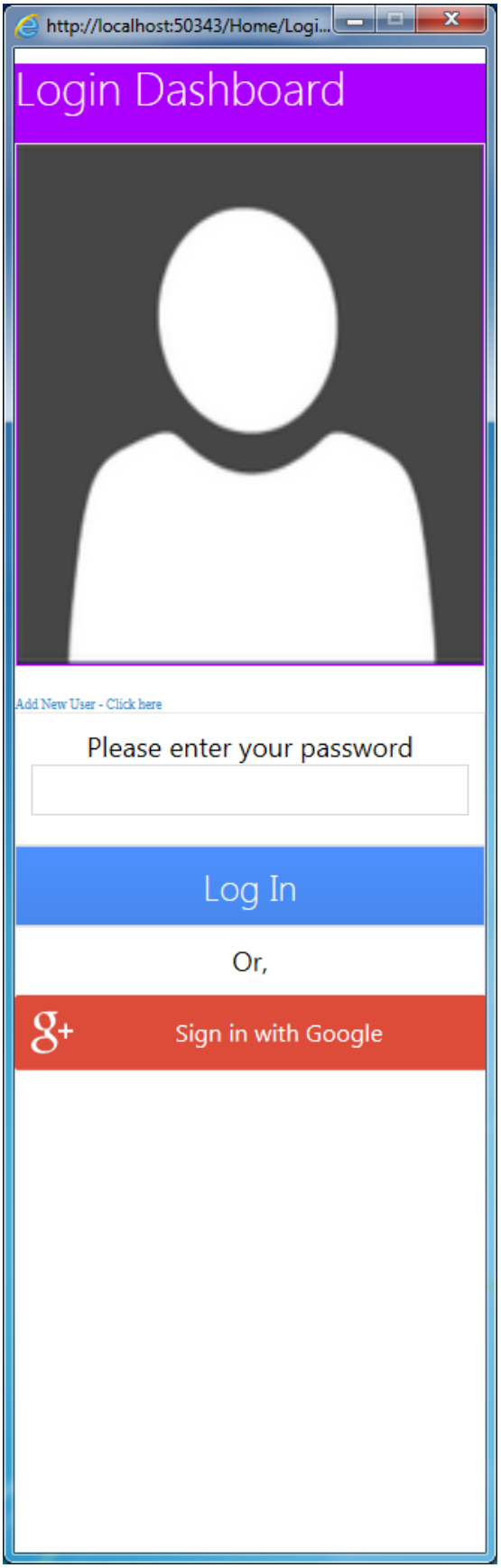
Adding Google SignIn to my MVC application



Signing in and authentication methods are different from social network to another social network. From application, to another web or desktop application.

In almost all cases, we need to enter our user name, followed by a password, after a registration procedure that requires us to enter our first name, last name, nickname sometimes, phone, email etc…

From user point of view, simplicity and transparency is all that matters. The simpler the registration, and then the login parts of an application is, the better.

To software designer, who designs these applications, security of an identity is a big issue. Why ? Because an application needs to identify user, know exactly who logged, or is logging in, and who is the other side to which it will be providing services.

Biometrics rock ! No doubt about that ! It includes card swipe sign in, fingerprint sign in , voice recognition sign in, and sometimes eye retina scan. But these solutions are expensive, highly personal and individual, and in many cases custom built to the needs of a client.

But how about a regular app. An app that has users, contains blogs, has picture upload features, etc… , that are intended for massive user base across the world.

From software designer perspective, the picture to the left provides very convenient way, both to the user, and software developer, to log in and authenticate securely. Why ?

The first section is users picture. It is users choice to identify himself with whatever picture he wants. From computer software point of view,pictures are more unique than words, because no two similar pictures have identical pixel, and layer layout and formation. Meaning that if there are two users with same names, there are almost no two identical pictures !!! A picture is unique in the entire world !

Another unique thing that we may identify users with this login is address. No two users can have same address. But what if two people want to be on the same social network or app. Their addresses will be identical, and we will lose our uniqueness to identify a user. Addresses work on a global scale with ratios, meaning the approach to identify uniquely a person by his address is not unique. It might work in some countries, but not in all of the countries.

If we look at the picture to the left, we have removed unnecessary log in name, or login nickname. It is not needed at all. The picture is there. And a password. In case that some may argue

that passwords could be the same, we could notify a user to put another password. But that is not necessary. So here we give a user two options to log in. Either by entering a password, only password, or by signing in with Google Account, which has many positive fruitful benefits, or, and this is the part where I like it the most, using both !

From a developer perspective, using both methods to authenticate a user and identify him, is a security heaven ! Why? Well, let's look further into what google offers developers.

```
function onSignIn(googleUser) {
  var profile = googleUser.getBasicProfile();
  console.log('ID: ' + profile.getId());
  console.log('Name: ' + profile.getName());
  console.log('Image URL: ' + profile.getImageUrl());
  console.log('Email: ' + profile.getEmail());
```

source: https://developers.google.com/identity/sign-in/web/sign-in

For the sake of example, after implementing it in my MVC application, it looks like this :

```
    function renderButton() {
          gapi.signin2.render('my-signin2', {
              'scope': 'https://www.googleapis.com/auth/plus.login',
              'width': 315,
              'height': 50,
              'longtitle': true,
              'theme': 'dark',
              'onsuccess': onSignIn,
              'onfailure': onFailure
          });
      }

        function onSignIn(googleUser) {
            var profile = googleUser.getBasicProfile();
            $("#myhiddenfield").val(profile.getName());
            $('form#myForm').submit();
        }


<script src="https://apis.google.com/js/platform.js?onload=renderButton" async defer></script>
```

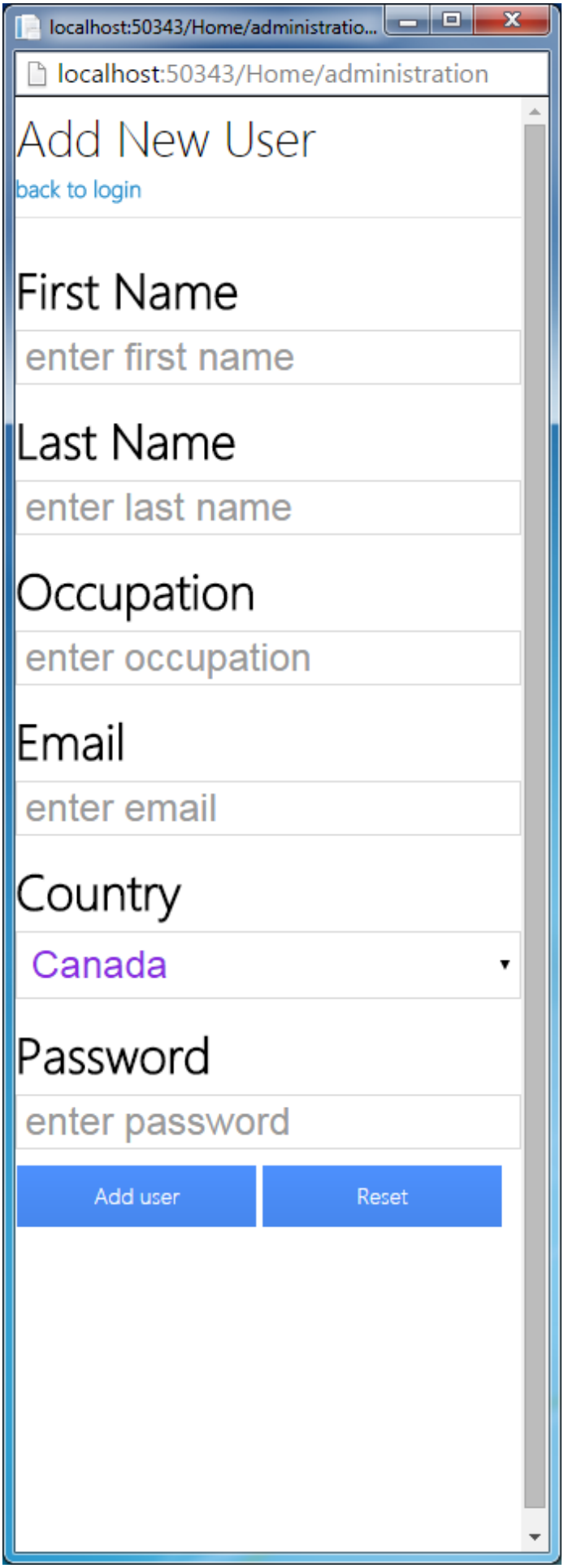
The final script is for button rendering, to get the right specifications for Google Sign in button, renderButton function renders the specifics of a Google Sign in button, and 'on success', or once authenticated successfully with google acocunts, we trigger onSignIn function, which will get the users details, including his name, ID, his Image, and his email.

If he is authenticated, he will proceed directly to other parts of my MVC application.

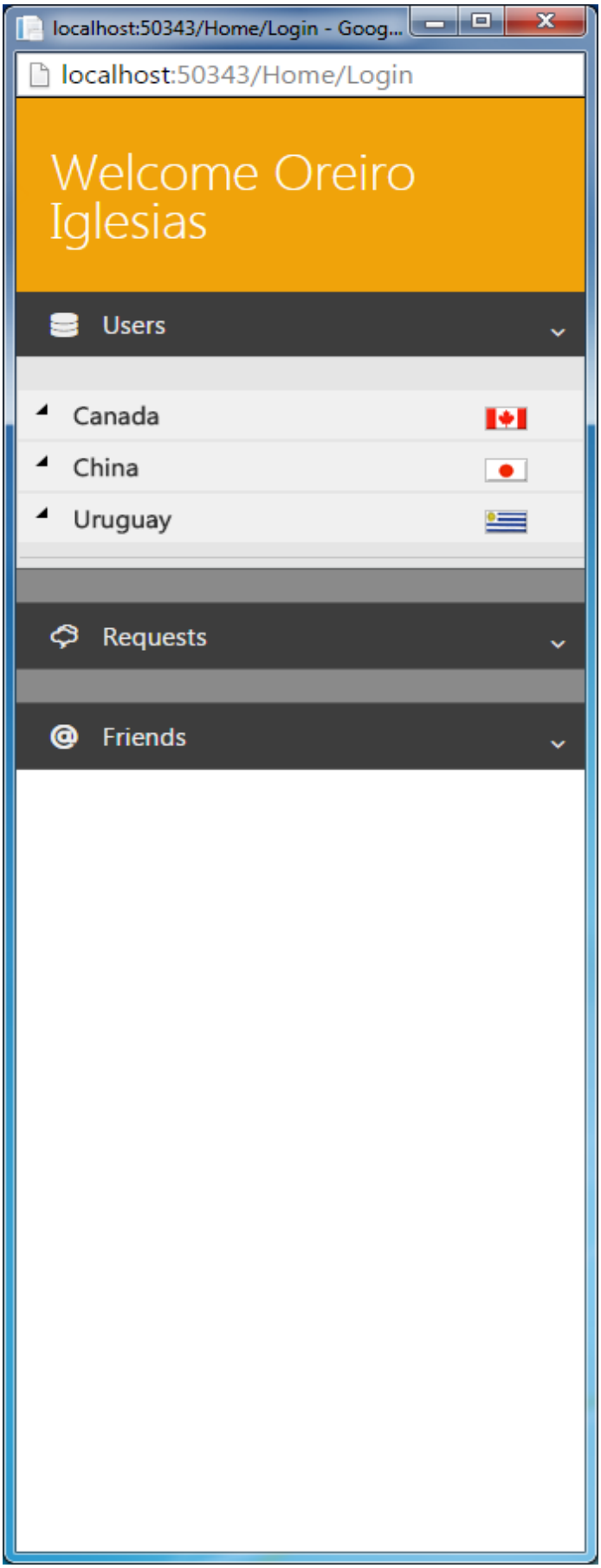
Another option given to a user, is entering a password, and pressing our custom blue log in button (on the picture to the left) Once the log in is pressed, we authenticate against sql server to match the user with out database. As an example, I used this code below:

```
DataTable userdetails = new DataTable();

string connectionString = "Data Source=\\;Initial Catalog=Chat;Integrated Security=True";
string query = "SELECT * FROM [Chat].[dbo].[User] WHERE [upassword] = '"+userPassword+"'";

        System.Data.SqlClient.SqlConnection conn = new
        System.Data.SqlClient.SqlConnection(connectionString);

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand(query, conn);

conn.Open();
System.Data.SqlClient.SqlDataAdapter da = new System.Data.SqlClient.SqlDataAdapter(cmd);
da.Fill(userdetails);
conn.Close();

da.Dispose();

return userdetails;
```

And to be able to do this I used the form below to register a user with my database. So in reality, this user is stored in my database, and has a google account which my application can verify. If both names and IDs match, we have a good solid security checkup ! Below is the registration screen:


localhost:50343/Home/administratio...
localhost:50343/Home/administration
Add New User
back to login
First Name
enter first name
Last Name
enter last name
Occupation
enter occupation
Email
enter email
Country
Canada
Password
enter password
Add user
Reset

And after a user is authenticated, he could be redirected to the MVCs application business logic and as an example, this is a screen showing users requests to join a game:



Thank you,

Jinan Kordab

Programmer Analyst - Software Developer

Mobile: +1 438 401 9096