## User Access Workflow and Design

Updated : 3-26-2013 ola lawal

Create user workflow

**general user login steps**

Two options for logging in

1. new users

i)normal account creation

ii)account creation via FaceBook

2) existing user

1) new user s

i) normal account creation

Steps:

1)gather registration data based on the order of the screens :

a) General profile data

-**Username** - add a tooltip explaining that this is what the user uses to log on to the app from the website or the app if they log out of the app.

-**Email Address** – pre-populate this value with the users ApplieId email address if accessible via API, in case of android attempt to use GoogleID which is also their google app store id . If that is not available leave blank and force user to enter it . Add tootip statting that this is needed for updates such as email and account notifications.

-**ScreenName**- add a tooltip explaining that since the username is a private value , screen name is the name other users will know you as. i.e a nickname.

-**Birthdate** – add tootip statting must be 18 and older.

-Gender – gender list is populated from this service call: <http://localhost/Shell.MVC2.Web.Common/LookupService.svc/Rest/getgenderlist>

json response is a generic list format below

"description":"String content",

"id":2147483647

}]

Where description is the text to display and the ID is the value stored back in the database and the value.

Help page for this service call:

<http://localhost/Shell.MVC2.Web.Common/lookupservice.svc/rest/help/operations/getgenderlist>

-**Ethnicity** – is also populated by the lookup service :

<http://localhost/Shell.MVC2.Web.Common/LookupService.svc/Rest/getethnicitylist>

json response is a generic list format below

"description":"String content",

"id":2147483647

}]

Where description is the text to display and the ID is the value stored back in the database and the value.

Help page for this service call:

<http://localhost/Shell.MVC2.Web.Common/lookupservice.svc/rest/help/operations/getethnicitylist>

b)Location data

Prelimindary step

Option a)

-access the current country list from the service since it is subject to change and possibly store it locallay on the app using the following service call store it in local storage:

<http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/help/operations/getcountrylist>

the JSON list returned is of the format :

[{

"countryindex":"String content",

"countryvalue":"String content",

"selected":true

}]

Where countryIndex is the CountryID and the CountryValue is the name of the country.

When using this step you have to get the postal code status of the country(i.e if this country has postal codes entered in our database or not) so that the postal code can be predicted and/or updated/added by the user.

To get postcode status by countryname user the method : [http://localhost/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getpostalcodestatusbycountryname/{COUNTRYNAME}](http://localhost/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getpostalcodestatusbycountryname/%7bCOUNTRYNAME%7d)

the COUNTRYNAME field is the name of the country without spaces allowed i.e united states would be “United States”

If the result is “**TRUE**” this country has a postal code in our database so the postal code field will have to be displayed . Population of postal code happens later in this workflow. For now record the postalcode status. Otherwise this country has not postalcode and we just need the Country,City,Lat,Long.

Obtion b)

Otherwise to save a service call down the line you can get a list of countries along with their postal code status :

<http://localhost/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/help/operations/getcountryandpostalcodestatuslist>

this returns a list of JSON object with the following structure

[{

"CountryCustomRegionID":2147483647,

"CountryID":255,

"CountryName":"String content",

"Country\_Code":"String content",

"Country\_Region":"String content",

"PostalCodes":True

}]

Where the PostalCodes value will be **1 or 0** 1 denotes the country has postal codes and 0 denotes it does not.

\*\*\* note this method is fixed and working now \*\*\*

-

Now that preliminary step is done moving forward :

i) IF access to location was allowed

prepopulatye country , city and store latitude and longitude in local storeage since it is a displayed field but is alos needed on the profile create.

**Country** – get from phone location – pre-polulate

**City** – get from phone location – pre populate

**PostalCode -**

If we have a country that has a postalcode (see **a** above) prepopulate the postal code using the following method

[http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getgeopostalcodebycountrynameandcity/{STRCOUNTRYNAME}/{STRCITY}](http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getgeopostalcodebycountrynameandcity/%7bSTRCOUNTRYNAME%7d/%7bSTRCITY%7d)

Again **StrCountryName** and **StrCity** are strings with no spaces allowed , this only returns a single postal code .

-If the user wants to fine tune this postal code you will have to dynamically populate a list of postal codes as they type in the numbers using this method : [http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getpostalcodesbycountryandcityprefixdynamic/{STRCOUNTRYNAME}/{STRCITY}/{STRPREFIXTEXT}](http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getpostalcodesbycountryandcityprefixdynamic/%7bSTRCOUNTRYNAME%7d/%7bSTRCITY%7d/%7bSTRPREFIXTEXT%7d)

StrPrefixtext – the prefix they enter for their postal code they are seeking i.e a USA postal code of 55555

As they type in 55 (allow min of two chars before pinging the web service ) a sample call would be :

<http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getpostalcodesbycountryandcityprefixdynamic/unitestates/minneapolis/55>

and returns a JSON list of :

[{

"PostalCode":"String content"

}]

i) IF access to location was **NOT** allowed

can be generated

Validation of above Entered values must happen before data is submitted

[3/24/2013 10:55:25 PM] himanshi vishnoi: 2) validate screen name

[3/24/2013 10:55:27 PM] himanshi vishnoi: 3) create user

[3/24/2013 10:55:30 PM] himanshi vishnoi: 4) upload pic

Two was of creating a new user account

**1) Non Facebook or openID user**

-prompt use that this app requires access to your location

-gather user data and fill out the RegisterModel/creatuser:

-data validation :

email address

-----------

[http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/checkifemailalreadyexists/{EMAILADDRESS}](http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/checkifemailalreadyexists/%7bEMAILADDRESS%7d)

-returns a true if it exists i,e email address is invalid

screenanem

----------

must be at least six chars , can contain special chars

[http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/checkifscreennamealreadyexists/{SCREENAME}](http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/checkifscreennamealreadyexists/%7bSCREENAME%7d)

-reutrns true if screen name already exists -user must pick a new one

username

----------

must be at least six chars , can contain special chars within reason

[http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/checkifusernamealreadyexists/{USERNAME}](http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/checkifusernamealreadyexists/%7bUSERNAME%7d)

- returnes true if usenrame already exists- user must pick a new one

here is the JSON data needed

{

"birthdate":"\/Date(928167600000-0500)\/",

"city":"Minneapolis",

"country":"United States",

"email":"testdsdsdasda@yahoo.com",

"gender":"Male",

"isApproved":true,

"lattitude":1.26743233E+15,

"longitude":1.26743233E+15,

"openidIdentifer":"",

"openidProvidername":"",

"password":"kayode02",

"providerUserKey":{},

"screenname":"testofusername23",

"securityAnswer":"",

"securityQuestion":"",

"stateprovince":"Minnesota",

"status":0,

"username":"examplecreate",

"zippostalcode":"55411"

}

-location data city,country,longitude,lattitude stateprovince - can be generated from phone

or use the geoservice to get that data.

-if user opts to turn off gelocation on phone we still have to find a way to veryfy that the user enteres in

location data that matches what is on thier phone/tablet , so maybee not let them turn it off

-veryfy the geodata if manually entered in using the geoservice

http://localhost/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/verifyorupdateregistrationgeodata

also if you do not have lat and long info you can get that data using this method. Will provide sample calls in near future

once everythin is verifed call

http://localhost/Shell.MVC2.Web.AuthenticationService/MembershipService.svc/Rest/CreateUser

with the JSON body above and

if status is 0 on the return profile was created.

Option B. Facebook or openID user

-pretty much same as above , difference is you use the following JSON payload instead

:

{

"birthdate":"\/Date(928167600000-0500)\/",

"city":"Minneapolis",

"country":"United States",

"email":"testdsdsdasda@yahoo.com",

"gender":"Male",

"isApproved":true,

"lattitude":1.26743233E+15,

"longitude":1.26743233E+15,

"openidIdentifer":"1",

"openidProvidername":"Facebook",

"password":"kayode02",

"providerUserKey":{},

"screenname":"testofusername23",

"securityAnswer":"",

"securityQuestion":"",

"stateprovince":"Minnesota",

"status":0,

"username":"examplecreate",

"zippostalcode":"55411"

}

As you can see the openIDprovider and Identifier are specified in the JSOn payload.

- get profileinformation after creation

------------------------------------

[http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/getprofileidbyusername/{USERNAME}](http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/getprofileidbyusername/%7bUSERNAME%7d)

or

[http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/getprofileidbyscreenname/{SCREENAME}](http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/getprofileidbyscreenname/%7bSCREENAME%7d)

then use the profileID information to populate photos below...

photos - adding

-A Profile Photo is also required – the ability to allow the user to choose a photo from their device gallery or take a new one to add is needed. Addtionally users need to be able to upload more than a single photo on their initial logon since this functionality exists currently on the website version.

Media/Photos can be uploaded via two methods and there are

http://localhost/Shell.MVC2.Web.MediaService/PhotoService.svc/Rest/help/operations/addphotos

multiple photos or one phot can be added with this call:

Json body

{

"autoupload":true,

"multiple":true,

"photosuploaded":[{

"approvalstatusid":null,

"caption":"String content",

"creationdate":"\/Date(928167600000-0500)\/",

"image":[81,

109,

70,

122,

90,

83,

65,

50,

78,

67,

66,

84,

100,

72,

74,

108,

89,

87,

48,

61],

"imagename":"teest",

"imagetypeid":2147483647,

"photostatusid":2147483647,

"rejectionreasonid":null,

"size":2147483647

}],

"profileid":2

}

To get the imagetypeID and Phosotstatusid values call the lookup service and look at these two methods :

photostatusID

<http://localhost/Shell.MVC2.Web.Common/LookupService.svc/Rest/getphotostatusdescriptionlist>

the default status for a new photo is : 1

Profile Activation :

If the user created a profile withoute using Facebook or an other open ID provider we need to validate their activation code via email.

Using the following call :