**User Access Workflow and Design**

Updated : 3-26-2013 ola lawal

\*\* add header below to rquest for JSON formatted responses \*\*

**Content-Type: text/json**

Create user workflow

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

User Data needed to create profile : (this can be got from the device directly or if not allowed

Entered in by the user) i.e if user signing in with twitter or facebook API’s a lot of that data can be got from there.

Username

Pssword

Confirms password

Email address

Email address confirms

Birth Date

Country

City

State Province (optional)

Zip or Postal

Screen Name

**Workflow steps :**

-Collect user data

-validate data

-create the profile

-collect user photo

-display activation page ( require activation code ) ALLOW user to BYPASS this I.e activate later

-send user to their profile home page after photo collected.

Data must be correct before the profile is created the validation details are explained in the validation section after the geographical data is explained.

**Initial user Creation Steps**

Two options for logging in

* 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://173.160.122.195/Common/LookupService.svc/Rest/getgenderlist](http://173.160.122.195/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://173.160.122.195/Common/LookupService.svc/Rest/getgenderlist](http://173.160.122.195/Shell.MVC2.Web.Common/LookupService.svc/Rest/getgenderlist)

[{"description":"Female","id":2},{"description":"Male","id":1}]

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

[http://173.160.122.195/Common/LookupService.svc/Rest/getethnicitylist](http://173.160.122.195/Shell.MVC2.Web.Common/LookupService.svc/Rest/getethnicitylist)

json response is a generic list format below

[{"description":"A few extra pounds","id":11,"selected":false},{"description":"Any","id":1,"selected":false},{"description":"Asian","id":3,"selected":false},{"description":"Black \/ African descent","id":7,"selected":false},{"description":"Caucasian \/ European descent","id":8,"selected":false},{"description":"East Indian","id":12,"selected":false},{"description":"Interacial","id":4,"selected":false},{"description":"Latino \/ Hispanic","id":2,"selected":false},{"description":"Middle Eastern","id":5,"selected":false},{"description":"Native American","id":9,"selected":false},{"description":"Other","id":6,"selected":false},{"description":"Pacific Islander","id":10,"selected":false}]

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://173.160.122.195/Common/lookupservice.svc/rest/help/operations/getethnicitylist](http://173.160.122.195/Shell.MVC2.Web.Common/lookupservice.svc/rest/help/operations/getethnicitylist)

b)Location data

Prelimindary step

b) Geographical data

**FIXED**

**- Country List**

-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/GeoService/GeoService.svc/Rest/help/operations/getcountryandpostalcodestatuslist](http://173.160.122.195/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

}]

Full list of countries returnred along with postalcode status

[{"code":"AF ","customregionid":null,"**haspostalcode**":false,"id":null,"name":"Afghanistan","region":"ME "},{"code":"AS ","customregionid":null,"haspostalcode":false,"id":null,"name":"American Samoa","region":"OC "},{"code":"AD ","customregionid":null,"haspostalcode":true,"id":null,"name":"Andorra","region":"EU "},{"code":"AO ","customregionid":null,"haspostalcode":false,"id":null,"name":"Angola","region":"AF "},{"code":"AG ","customregionid":null,"haspostalcode":false,"id":null,"name":"Antigua and Barbuda","region":"NA "},{"code":"AR ","customregionid":null,"haspostalcode":true,"id":null,"name":"Argentina","region":"SA "},{"code":"AU ","customregionid":null,"haspostalcode":true,"id":null,"name":"Australia","region":"OC "},{"code":"AT ","customregionid":null,"haspostalcode":true,"id":null,"name":"Austria","region":"EU "},{"code":"AZ ","customregionid":null,"haspostalcode":false,"id":null,"name":"Azerbaijan","region":"EU "},{"code":"BS ","customregionid":null,"haspostalcode":false,"id":null,"name":"Bahamas","region":"CA "},{"code":"BH ","customregionid":null,"haspostalcode":false,"id":null,"name":"Bahrain","region":"AS "},{"code":"BD ","customregionid":null,"haspostalcode":true,"id":null,"name":"Bangladesh","region":"AS "},{"code":"BB ","customregionid":null,"haspostalcode":false,"id":null,"name":"Barbados","region":"CA "},{"code":"BE ","customregionid":null,"haspostalcode":true,"id":null,"name":"Belgium","region":"EU "},{"code":"BZ ","customregionid":null,"haspostalcode":false,"id":null,"name":"Belize","region":"SA "},{"code":"BM ","customregionid":null,"haspostalcode":false,"id":null,"name":"Bermuda","region":"NA "},{"code":"BR ","customregionid":null,"haspostalcode":true,"id":null,"name":"Brazil","region":"SA "},{"code":"VG ","customregionid":null,"haspostalcode":false,"id":null,"name":"British Virgin Islands","region":"CA "},{"code":"BG ","customregionid":null,"haspostalcode":false,"id":null,"name":"Bulgaria","region":"EU "},{"code":"CA ","customregionid":null,"haspostalcode":true,"id":null,"name":"Canada","region":"NA "},{"code":"CV ","customregionid":null,"haspostalcode":false,"id":null,"name":"Cape Verde","region":"AF "},{"code":"KY ","customregionid":null,"haspostalcode":false,"id":null,"name":"Cayman Islands","region":"CA "},{"code":"CL ","customregionid":null,"haspostalcode":false,"id":null,"name":"Chile","region":"NA "},{"code":"CN ","customregionid":null,"haspostalcode":false,"id":null,"name":"China","region":"AS "},{"code":"CO ","customregionid":null,"haspostalcode":false,"id":null,"name":"Colombia","region":"NA "},{"code":"CR ","customregionid":null,"haspostalcode":false,"id":null,"name":"Costa Rica","region":"NA "},{"code":"HR ","customregionid":null,"haspostalcode":true,"id":null,"name":"Croatia","region":"EE "},{"code":"CU ","customregionid":null,"haspostalcode":false,"id":null,"name":"Cuba","region":"NA "},{"code":"CY ","customregionid":null,"haspostalcode":false,"id":null,"name":"Cyprus","region":"EU "},{"code":"CZ ","customregionid":null,"haspostalcode":true,"id":null,"name":"Czech Republic ","region":"EU "},{"code":"DK ","customregionid":null,"haspostalcode":true,"id":null,"name":"Denmark","region":"EU "},{"code":"DO ","customregionid":null,"haspostalcode":true,"id":null,"name":"Dominican Republic","region":"NA "},{"code":"EG ","customregionid":null,"haspostalcode":false,"id":null,"name":"Egypt","region":"AF "},{"code":"ER ","customregionid":null,"haspostalcode":false,"id":null,"name":"Eritrea","region":"AF "},{"code":"FK ","customregionid":null,"haspostalcode":false,"id":null,"name":"Falkland Islands","region":"SA "},{"code":"FJ ","customregionid":null,"haspostalcode":false,"id":null,"name":"Fiji","region":"OC "},{"code":"FI ","customregionid":null,"haspostalcode":true,"id":null,"name":"Finland","region":"EU "},{"code":"FR ","customregionid":null,"haspostalcode":true,"id":null,"name":"France","region":"EU "},{"code":"GF ","customregionid":null,"haspostalcode":true,"id":null,"name":"French Guiana","region":"SA "},{"code":"PF ","customregionid":null,"haspostalcode":false,"id":null,"name":"French Polynesia","region":"OC "},{"code":"DE ","customregionid":null,"haspostalcode":true,"id":null,"name":"Germany","region":"EU "},{"code":"GH ","customregionid":null,"haspostalcode":false,"id":null,"name":"Ghana","region":"AF "},{"code":"GI ","customregionid":null,"haspostalcode":false,"id":null,"name":"Gibraltar","region":"EU "},{"code":"GL ","customregionid":null,"haspostalcode":true,"id":null,"name":"Greenland","region":"EU "},{"code":"GU ","customregionid":null,"haspostalcode":false,"id":null,"name":"Guam","region":"OC "},{"code":"GT ","customregionid":null,"haspostalcode":true,"id":null,"name":"Guatemala","region":"SA "},{"code":"GG ","customregionid":null,"haspostalcode":false,"id":null,"name":"Guernsey","region":"EU "},{"code":"GY ","customregionid":null,"haspostalcode":false,"id":null,"name":"Guyana","region":"SA "},{"code":"HT ","customregionid":null,"haspostalcode":false,"id":null,"name":"Haiti","region":"CA "},{"code":"HN ","customregionid":null,"haspostalcode":false,"id":null,"name":"Honduras","region":"SA "},{"code":"HK ","customregionid":null,"haspostalcode":false,"id":null,"name":"Hong Kong","region":"AS "},{"code":"HU ","customregionid":null,"haspostalcode":true,"id":null,"name":"Hungary","region":"EU "},{"code":"IS ","customregionid":null,"haspostalcode":true,"id":null,"name":"Iceland","region":"EU "},{"code":"IN ","customregionid":null,"haspostalcode":true,"id":null,"name":"India","region":"AS "},{"code":"ID ","customregionid":null,"haspostalcode":false,"id":null,"name":"Indonesia","region":"OC "},{"code":"IQ ","customregionid":null,"haspostalcode":false,"id":null,"name":"Iraq","region":"ME "},{"code":"IE ","customregionid":null,"haspostalcode":false,"id":null,"name":"Ireland","region":"EU "},{"code":"IM ","customregionid":null,"haspostalcode":false,"id":null,"name":"Isle Of Man","region":"EU "},{"code":"IL ","customregionid":null,"haspostalcode":false,"id":null,"name":"Israel","region":"ME "},{"code":"IT ","customregionid":null,"haspostalcode":true,"id":null,"name":"Italy","region":"EU "},{"code":"CI ","customregionid":null,"haspostalcode":false,"id":null,"name":"Ivory Coast","region":"AF "},{"code":"JM ","customregionid":null,"haspostalcode":false,"id":null,"name":"Jamaica","region":"CA "},{"code":"JP ","customregionid":null,"haspostalcode":true,"id":null,"name":"Japan","region":"AS "},{"code":"JE ","customregionid":null,"haspostalcode":false,"id":null,"name":"Jersey","region":"EU "},{"code":"JO ","customregionid":null,"haspostalcode":false,"id":null,"name":"Jordan","region":"ME "},{"code":"KE ","customregionid":null,"haspostalcode":false,"id":null,"name":"Kenya","region":"AF "},{"code":"KI ","customregionid":null,"haspostalcode":false,"id":null,"name":"Kiribati","region":"EU "},{"code":"LB ","customregionid":null,"haspostalcode":false,"id":null,"name":"Lebanon","region":"ME "},{"code":"LR ","customregionid":null,"haspostalcode":false,"id":null,"name":"Liberia","region":"AF "},{"code":"LI ","customregionid":null,"haspostalcode":true,"id":null,"name":"Liechtenstein","region":"EE "},{"code":"LU ","customregionid":null,"haspostalcode":true,"id":null,"name":"Luxembourg","region":"EU "},{"code":"MK ","customregionid":null,"haspostalcode":true,"id":null,"name":"Macedonia","region":"EU "},{"code":"MG ","customregionid":null,"haspostalcode":false,"id":null,"name":"Madagascar","region":"AF "},{"code":"ML ","customregionid":null,"haspostalcode":false,"id":null,"name":"Malaysia","region":"AS "},{"code":"MT ","customregionid":null,"haspostalcode":false,"id":null,"name":"Malta","region":"EU "},{"code":"MH ","customregionid":null,"haspostalcode":false,"id":null,"name":"MarshallIslands","region":"OC "},{"code":"MQ ","customregionid":null,"haspostalcode":true,"id":null,"name":"Martinique","region":"NA "},{"code":"YT ","customregionid":null,"haspostalcode":true,"id":null,"name":"Mayotte","region":"AF "},{"code":"MX ","customregionid":null,"haspostalcode":true,"id":null,"name":"Mexico","region":"SA "},{"code":"MD ","customregionid":null,"haspostalcode":true,"id":null,"name":"Moldova","region":"EE "},{"code":"MC ","customregionid":null,"haspostalcode":true,"id":null,"name":"Monaco","region":"EU "},{"code":"MA ","customregionid":null,"haspostalcode":false,"id":null,"name":"Morocco","region":"AF "},{"code":"NP ","customregionid":null,"haspostalcode":false,"id":null,"name":"Nepal","region":"AS "},{"code":"NL ","customregionid":null,"haspostalcode":true,"id":null,"name":"Netherlands","region":"EU "},{"code":"NZ ","customregionid":null,"haspostalcode":true,"id":null,"name":"New Zealand","region":"OC "},{"code":"NG ","customregionid":null,"haspostalcode":false,"id":null,"name":"Nigeria","region":"AF "},{"code":"MP ","customregionid":null,"haspostalcode":false,"id":null,"name":"Northern Mariana Islands","region":"OC "},{"code":"NO ","customregionid":null,"haspostalcode":true,"id":null,"name":"Norway","region":"EU "},{"code":"PK ","customregionid":null,"haspostalcode":true,"id":null,"name":"Pakistan","region":"AS "},{"code":"PG ","customregionid":null,"haspostalcode":false,"id":null,"name":"Papua New Guinea","region":"OC "},{"code":"PE ","customregionid":null,"haspostalcode":false,"id":null,"name":"Peru","region":"NA "},{"code":"PH ","customregionid":null,"haspostalcode":false,"id":null,"name":"Philippines","region":"AS "},{"code":"PL ","customregionid":null,"haspostalcode":true,"id":null,"name":"Poland","region":"EU "},{"code":"PT ","customregionid":null,"haspostalcode":true,"id":null,"name":"Portugal","region":"EU "},{"code":"PR ","customregionid":null,"haspostalcode":false,"id":null,"name":"Puerto Rico","region":"NA "},{"code":"QA ","customregionid":null,"haspostalcode":false,"id":null,"name":"Qatar","region":"ME "},{"code":"RE ","customregionid":null,"haspostalcode":true,"id":null,"name":"Reunion","region":"AF "},{"code":"RU ","customregionid":null,"haspostalcode":false,"id":null,"name":"Russia","region":"EU "},{"code":"PM ","customregionid":null,"haspostalcode":false,"id":null,"name":"Saint Pierreand Miquelon","region":"NA "},{"code":"VC ","customregionid":null,"haspostalcode":false,"id":null,"name":"Saint Vincent and the Grenadines","region":"EU "},{"code":"WS ","customregionid":null,"haspostalcode":false,"id":null,"name":"Samoa","region":"OC "},{"code":"SA ","customregionid":null,"haspostalcode":false,"id":null,"name":"Saudi Arabia","region":"ME "},{"code":"SN ","customregionid":null,"haspostalcode":false,"id":null,"name":"Senegal","region":"AF "},{"code":"SK ","customregionid":null,"haspostalcode":true,"id":null,"name":"Slovakia","region":"EU "},{"code":"SI ","customregionid":null,"haspostalcode":true,"id":null,"name":"Slovenia","region":"EU "},{"code":"SB ","customregionid":null,"haspostalcode":false,"id":null,"name":"Solomon Islands","region":"OC "},{"code":"ZA ","customregionid":null,"haspostalcode":true,"id":null,"name":"South Africa","region":"AF "},{"code":"ES ","customregionid":null,"haspostalcode":true,"id":null,"name":"Spain","region":"EU "},{"code":"LK ","customregionid":null,"haspostalcode":true,"id":null,"name":"Sri Lanka","region":"AS "},{"code":"SE ","customregionid":null,"haspostalcode":true,"id":null,"name":"Sweden","region":"EU "},{"code":"CH ","customregionid":null,"haspostalcode":true,"id":null,"name":"Switzerland","region":"EU "},{"code":"SY ","customregionid":null,"haspostalcode":false,"id":null,"name":"Syria","region":"ME "},{"code":"TW ","customregionid":null,"haspostalcode":false,"id":null,"name":"Taiwan","region":"AS "},{"code":"TZ ","customregionid":null,"haspostalcode":false,"id":null,"name":"Tanzania","region":"AF "},{"code":"TH ","customregionid":null,"haspostalcode":true,"id":null,"name":"Thailand","region":"AS "},{"code":"TT ","customregionid":null,"haspostalcode":false,"id":null,"name":"Trinidad and Tobago","region":"NA "},{"code":"TR ","customregionid":null,"haspostalcode":true,"id":null,"name":"Turkey","region":"EE "},{"code":"UG ","customregionid":null,"haspostalcode":false,"id":null,"name":"Uganda","region":"AF "},{"code":"UA ","customregionid":null,"haspostalcode":false,"id":null,"name":"Ukraine","region":"EU "},{"code":"AE ","customregionid":null,"haspostalcode":false,"id":null,"name":"United Arab Emirates","region":"ME "},{"code":"GB ","customregionid":null,"haspostalcode":true,"id":null,"name":"United Kingdom","region":"EU "},{"code":"US ","customregionid":null,"haspostalcode":true,"id":null,"name":"United States","region":"NA "},{"code":"VI ","customregionid":null,"haspostalcode":false,"id":null,"name":"US Virgin Islands","region":"CA "},{"code":"VE ","customregionid":null,"haspostalcode":false,"id":null,"name":"Venezuela","region":"NA "},{"code":"VN ","customregionid":null,"haspostalcode":false,"id":null,"name":"Vietnam","region":"AS "},{"code":"YE ","customregionid":null,"haspostalcode":false,"id":null,"name":"Yemen","region":"ME "}]

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

\*\*\*Country is tied to postal code validation as country with postalcode = 1 must have a postal code entered and validated.\*\*\*

See the validation sections

\*\*\* 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 also 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 dropdown list using the following method

-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 or letters using this method :

[http://173.160.122.195/GeoService/GeoService.svc/Rest/getfilteredpostalcodesbycountrycityandfilter/{COUNTRY}/{CITY}/{FILTER}](http://localhost/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getfilteredpostalcodesbycountrycityandfilter/%7bCOUNTRY%7d/%7bCITY%7d/%7bFILTER%7d)

country : I.e unitedstates - spaces not allowed - required

city – city they are from - required

filter – the text 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 :

sample call

[http://173.160.122.195/GeoService/GeoService.svc/Rest/getfilteredpostalcodesbycountrycityandfilter/UnitedStates/Minneapolis/53](http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getfilteredpostalcodesbycountrycityandfilter/UnitedStates/Minneapolis/53)

and returns a JSON list of :

[{

"PostalCode":"String content"

}]

Response of abpve query

[{"postalcodevalue":"55419","selected":false},{"postalcodevalue":"55418","selected":false},{"postalcodevalue":"55417","selected":false},{"postalcodevalue":"55416","selected":false},{"postalcodevalue":"55415","selected":false},{"postalcodevalue":"55414","selected":false},{"postalcodevalue":"55413","selected":false},{"postalcodevalue":"55412","selected":false},{"postalcodevalue":"55411","selected":false},{"postalcodevalue":"55410","selected":false},{"postalcodevalue":"55409","selected":false},{"postalcodevalue":"55408","selected":false},{"postalcodevalue":"55407","selected":false},{"postalcodevalue":"55406","selected":false},{"postalcodevalue":"55405","selected":false},{"postalcodevalue":"55404","selected":false},{"postalcodevalue":"55403","selected":false},{"postalcodevalue":"55402","selected":false},{"postalcodevalue":"55401","selected":false},{"postalcodevalue":"55473","selected":false},{"postalcodevalue":"55449","selected":false},{"postalcodevalue":"55448","selected":false},{"postalcodevalue":"55434","selected":false},{"postalcodevalue":"55433","selected":false},{"postalcodevalue":"55432","selected":false}]

**Lattitude and Longitude -**

Will be populated from GPS of the mobile device so store this data into tempoary storage after country,city ,postalcode (if available) has been already gathered.

ii) IF access to location was **NOT** 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 the call below:

[http://173.160.122.195/GeoService/GeoService.svc/Rest/getcountryandpostalcodestatuslist/](http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getcountryandpostalcodestatuslist/)

**City** – must be dynamically populated since you do not want a list of ALL the cities in a country user this call the autocomplete that list :

[http://173.160.122.195/GeoService/GeoService.svc/Rest/getfilteredcitiesbycountryandfilter/unitedstates/new](http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getfilteredcitiesbycountryandfilter/unitedstates/new)

result :

[{"citystateprovincevalue":"Cheyenne Wells,Colorado","selected":false},{"citystateprovincevalue":"Conewango Valley,New York","selected":false},{"citystateprovincevalue":"East New Market,Maryland","selected":false},{"citystateprovincevalue":"East Newport,Maine","selected":false},{"citystateprovincevalue":"Goodnews Bay,Alaska","selected":false},{"citystateprovincevalue":"Hunnewell,Missouri","selected":false},{"citystateprovincevalue":"Kennewick,Washington","selected":false},{"citystateprovincevalue":"Lincoln'S New Salem,Illinois","selected":false},{"citystateprovincevalue":"Lone Wolf,Oklahoma","selected":false},{"citystateprovincevalue":"Minnewaukan,North Dakota","selected":false},{"citystateprovincevalue":"New Albany,Indiana","selected":false},{"citystateprovincevalue":"New Albany,Kansas","selected":false},{"citystateprovincevalue":"New Albany,Mississippi","selected":false},{"citystateprovincevalue":"New Albany,Ohio","selected":false},{"citystateprovincevalue":"New Albany,Pennsylvania","selected":false},{"citystateprovincevalue":"New Albin,Iowa","selected":false},{"citystateprovincevalue":"New Alexandria,Pennsylvania","selected":false},{"citystateprovincevalue":"New Almaden,California","selected":false},{"citystateprovincevalue":"New Almelo,Kansas","selected":false},{"citystateprovincevalue":"New Athens,Illinois","selected":false},{"citystateprovincevalue":"New Athens,Ohio","selected":false},{"citystateprovincevalue":"New Auburn,Minnesota","selected":false},{"citystateprovincevalue":"New Auburn,Wisconsin","selected":false},{"citystateprovincevalue":"New Augusta,Mississippi","selected":false},{"citystateprovincevalue":"New Baden,Illinois","selected":false}]

**PostalCode – ( same as when gps is available)**

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

-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 or letters using this method :

[http://173.160.122.195/GeoService/GeoService.svc/Rest/getfilteredpostalcodesbycountrycityandfilter/{COUNTRY}/{CITY}/{FILTER}](http://localhost/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getfilteredpostalcodesbycountrycityandfilter/%7bCOUNTRY%7d/%7bCITY%7d/%7bFILTER%7d)

country : I.e unitedstates - spaces not allowed - required

city – city they are from - required

filter – the text 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 :

sample call

[http://173.160.122.195/GeoService/GeoService.svc/Rest/getfilteredpostalcodesbycountrycityandfilter/UnitedStates/Minneapolis/53](http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getfilteredpostalcodesbycountrycityandfilter/UnitedStates/Minneapolis/53)

and returns a JSON list of :

[{

"PostalCode":"String content"

}]

Response of abpve query

[{"postalcodevalue":"55419","selected":false},{"postalcodevalue":"55418","selected":false},{"postalcodevalue":"55417","selected":false},{"postalcodevalue":"55416","selected":false},{"postalcodevalue":"55415","selected":false},{"postalcodevalue":"55414","selected":false},{"postalcodevalue":"55413","selected":false},{"postalcodevalue":"55412","selected":false},{"postalcodevalue":"55411","selected":false},{"postalcodevalue":"55410","selected":false},{"postalcodevalue":"55409","selected":false},{"postalcodevalue":"55408","selected":false},{"postalcodevalue":"55407","selected":false},{"postalcodevalue":"55406","selected":false},{"postalcodevalue":"55405","selected":false},{"postalcodevalue":"55404","selected":false},{"postalcodevalue":"55403","selected":false},{"postalcodevalue":"55402","selected":false},{"postalcodevalue":"55401","selected":false},{"postalcodevalue":"55473","selected":false},{"postalcodevalue":"55449","selected":false},{"postalcodevalue":"55448","selected":false},{"postalcodevalue":"55434","selected":false},{"postalcodevalue":"55433","selected":false},{"postalcodevalue":"55432","selected":false}]

**Lattitude and Longitude -**

use this service call to get the longidtude and lattitude :

TWO options

1. If the user selected a country that DOES NOT have postal codes (HaspostalCodes == false ) use this call:

[http://173.160.122.195/GeoService/GeoService.svc/Rest/getpostalcodesbycountrynamecity/{COUNTRY}/{CITY}](http://localhost/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getpostalcodesbycountrynamecity/%7bCOUNTRY%7d/%7bCITY%7d)

-sample call

[http://173.160.122.195/GeoService/GeoService.svc/Rest/getgpsdatalistbycountrycity/Philippines/Visoria](http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getgpsdatalistbycountrycity/Philippines/Visoria)

sample result

[{"lattitude":15.566666603088379,"longitude":120.86666870117187,"postalcode":null,"selected":false,"stateprovince":""}]

2. if the user slected a country that HAS a postal code (HasPostalcodes == true)

[http://173.160.122.195/GeoService/GeoService.svc/Rest/getgpsdatabycitycountrypostalcode/{COUNTRY}/{CITY}/{POSTALCODE}](http://localhost/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getgpsdatabycitycountrypostalcode/%7bCOUNTRY%7d/%7bCITY%7d/%7bPOSTALCODE%7d)

Sample call

[http://173.160.122.195/GeoService/GeoService.svc/Rest/getgpsdatabycitycountrypostalcode/UnitedStates/Minneapolis/55411](http://173.160.122.195/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getgpsdatabycitycountrypostalcode/UnitedStates/Minneapolis/55411)

-result

{"lattitude":44.999599456787109,"longitude":-93.300498962402344,"postalcode":null,"selected":false,"stateprovince":"Minnesota"}

**\*\*\* Location and gps infor complete after this**

Validation of above Entered values must happen before data is submitted

Validation details :

-data validation :

email address - must be valid email address format !

----------- [http://173.160.122.195/MembersService/MembersService.svc/Rest/checkifemailalreadyexists](http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/checkifemailalreadyexists/%7bEMAILADDRESS%7d)

sample JSON body

{

"email":"ola\_lawal@yahoo.com"

}

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

screenname

----------

must be at least six chars , can contain special chars

[http://173.160.122.195/MembersService/MembersService.svc/Rest/checkifscreennamealreadyexists](http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/checkifscreennamealreadyexists/%7bSCREENAME%7d)

{

"screenname":"myname",

}

-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://173.160.122.195/MembersService/MembersService.svc/Rest/checkifusernamealreadyexists](http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/checkifusernamealreadyexists/%7bUSERNAME%7d)

{

"username":"firstuser"

}

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

**Validation of geo data**

-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://173.160.122.195/GeoService/GeoService.svc/Rest/verifyorupdateregistrationgeodata](http://173.160.122.195/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

end of updates for this section

manual validation:

**Country** – must come from the list of countries in the call:

<http://173.160.122.195/GeoService/GeoService.svc/Rest/help/operations/getcountryandpostalcodestatuslist>

**City**

**StateProvinice**

<http://173.160.122.195/GeoService/GeoService.svc/Rest/help/operations/getcountryandpostalcodestatuslist>

[{

"lattitude":1.26743233E+15,

"longitude":1.26743233E+15,

"postalcode":"String content",

"selected":true,

"stateprovince":"String content"

}]

You can get the list of state provinces as well from the above call

**ZipOrPostal**

(is required if country is in the list from this call)

<http://173.160.122.195/GeoService/GeoService.svc/Rest/help/operations/getcountryandpostalcodestatuslist>

if the country is in that list and postalcode status = 1 postal code is required

validate the postal code agains the call :

**best way to do this is to allow user to slect only postal codes from this service call:**

[http://173.160.122.195/GeoService/GeoService.svc/Rest/getpostalcodesbycountrynamecity/{COUNTRY}/{CITY}](http://localhost/Shell.MVC2.Web.GeoService/GeoService.svc/Rest/getpostalcodesbycountrynamecity/%7bCOUNTRY%7d/%7bCITY%7d)

using the city and country to create the filtered list of postal codes.

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:

**\*\*\*\*Upadated 7/12/2013 \*\*\*\*\***

**ProfileModel** replaces common strings like email since you cannot pass special chars like @ and / as well as GUIDS in a rest url

all the member actions user a common json body to pass items to the server here is the format. you only need to pass the values realted to the call you are making.

{ "activationcode":"String content",

"email":"String content"

, "openididentifier":"String content",

"openidprovider":"String content",

"photoid":"1627aea5-8e0a-4371-9022-9b504344e724",

"profileid":2147483647,

"screenname":"String content",

"securityanswer":"String content",

"securityquestion":"String content",

"sessionid":"String content",

"username":"String content"

}

**\*\*\*\*Updates end 7/12/2013 \*\*\*\*\***

**CREATE user after the data is validated**

Method call:

[http://173.160.122.195/AuthenticationService/MembershipService.svc/Rest/createuser](http://173.160.122.195/Shell.MVC2.Web.AuthenticationService/MembershipService.svc/Rest/createuser)

this is a POST method…

here is a sample of the POST JSON data needed for the createprofile

{

"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"

}

For now password is sent in plain text but will be over SSL later

\*\*\*NEW \*\*\*\*

I added a new response object that returns JSON data in the following format :

{

"Documents":[{

"document":[81,

109,

70,

122,

90,

83,

65,

50,

78,

67,

66,

84,

100,

72,

74,

108,

89,

87,

48,

61],

"documenttype":0,

"filename":"String content",

"mimetype":"String content"

}],

"ResponseMessages":[{

"dataelement":"String content",

"errormessage":"String content",

"message":"String content"

}],

"currentstatuscode":0,

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

"email":"String content",

"profileid1":"String content",

"profileid2":"String content",

"profilestatus":0,

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

"requestreturnflag":true,

"responsecreatedate":"\/Date(928167600000-0500)\/"

}

ResponseMessages will contain any error or other messages regarding if the profile was created or activated or any other type of action. IF there were any problems the errormessage field will be populated as well.

Also

The createuser service call validates unequie email address and username only, in the future we will validate long and lat to make sure they match city and country :

For eample a duplicate email will return the following JSON

{

"Documents":[],

"ResponseMessages":[{"dataelement":"","message":"","errormessage":"Duplicate email : the email :testdsdsdasda@yahoo.comalready exists"}],

"currentstatuscode":0,"currentstatusdate":"\/Date(-62135575200000-0600)\/","email":"","profileid1":"","profileid2":"","profilestatus":0,"profilestatusdate":"\/Date(-62135575200000-0600)\/","requestreturnflag":false,"responsecreatedate":"\/Date(-62135575200000-0600)\/"}

And duplicate username:

{"Documents":[],"ResponseMessages":[{"dataelement":""," errormessage":"Duplicate username : the username :examplecreatealready exists","message":"Unable to create profile"}],"currentstatuscode":0,"currentstatusdate":"\/Date(-62135575200000-0600)\/","email":"","profileid1":"","profileid2":"","profilestatus":0,"profilestatusdate":"\/Date(-62135575200000-0600)\/","requestreturnflag":false,"responsecreatedate":"\/Date(-62135575200000-0600)\/"}

On success ful created profile :

{"Documents":[],"ResponseMessages":[{"dataelement":""," errormessage":"” message":"profile creatred succesfully"}],"currentstatuscode":0,"currentstatusdate":"\/Date(-62135575200000-0600)\/","email":"","profileid1":"","profileid2":"","profilestatus":0,"profilestatusdate":"\/Date(-62135575200000-0600)\/","requestreturnflag":false,"responsecreatedate":"\/Date(-62135575200000-0600)\/"}

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://173.160.122.195/MembersService/MembersService.svc/Rest/getprofileidbyusername](http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/getprofileidbyusername/%7bUSERNAME%7d)

smaple json body

{

"username":"String content"

}

or

[http://173.160.122.195/MembersService/MembersService.svc/Rest/getprofileidbyscreenname/](http://localhost/Shell.MVC2.Web.MembersService/MembersService.svc/Rest/getprofileidbyscreenname/%7bSCREENAME%7d)

sample json body

{

"screenname":"String content",

}

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://173.160.122.195/MediaService/PhotoService.svc/Rest/help/operations/addphotos](http://173.160.122.195/Shell.MVC2.Web.MediaService/PhotoService.svc/Rest/help/operations/addphotos)

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

[http://173.160.122.195/MediaService/PhotoService.svc/Rest/help/operations/addphotos](http://173.160.122.195/Shell.MVC2.Web.MediaService/PhotoService.svc/Rest/help/operations/addphotos)

Json body

{

"autoupload":true,

"multiple":true,

"photosuploaded":[{

"approvalstatusid":1,

"caption":"String content",

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

"imagebytes":null,

"imageb64string":" ”,

"imagename":"String content",

"imagetypeid":1,

"photostatusid":1,

"rejectionreasonid":null,

}],

"profileid":2

}

the paramters you need :

"**autoupload":true** - set this to true

"**multiple":true,** - true if we have multiple photos to upload false if only one

the generic list of objects will be the **photosuploaded** object

parameters :

**"approvalstatusid":null, -** this should be null for each photo

**caption":"String content**" - the caption of the photo i.e " me and my sister"

**creationdate":"\/Date(928167600000-0500)\/"** - optional date if you want to pass a specific date you can if you want but the default date is date of the server.

**imageb64string ":** <- you must locally convert the image into a byte array and pass that as string (base 64 string)

"**imagename":"teest", - image name just the name of the image i.e meandmysister.jpg**

**"imagetypeid":**1,

\*\*this is from a list value : usee the call to get the image types from this call : \*i

**"photostatusid":1,**

\*\*this is from a list value : usee the call to get the image types from this call : \*i

**"rejectionreasonid":null**, - set as null

i,e you could be uploading a bunch of photos at once

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

For now just pass 1’s for both values

**\*\*\*\*\*\*Updated 7/12/2013 \*\*\*\*\***

**Display Profile Activation message at this point :**

After re-directing them to their home page display the message : *“Please check your email address for your activation code and click the link to activate your profile. Activating your profile verifys your email address and ensures you can rescive email notifications when changes are made to your profile.”*

*p.s*

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

On mobile and web we still allow them temporary access to their homepage even withoute activation. The cannot communicate/update their profiles and use mail features until they activate however. This will be built into the service calls. They will only be able to search and view user profiles.

steps

to Activate the profile they can click on thier email and redicret to the activate profile page or type in the activate code from thier email address along with email addres

if the forgot thier activation code they can call this method to have it re-emailed to them:

[http://173.160.122.195/AuthenticationService/MembershipService.svc/Rest/recoveractivationcode](http://173.160.122.195/Shell.MVC2.Web.AuthenticationService/MembershipService.svc/Rest/recoveractivationcode)

{ "activationcode":"String content", "emailaddress":"String content", "photostatus":true, "photouploadviewmodel":{ "autoupload":true, "multiple":true, "photosuploaded":[{ "approvalstatusid":2147483647, "caption":"String content", "creationdate":"\/Date(928167600000-0500)\/", "imageb64string":"String content", "imagebytes":[255], "imagename":"String content", "imagetypeid":2147483647, "photostatusid":2147483647, "rejectionreasonid":2147483647 }], "profileid":2147483647 }}

Sample call

{

"emailaddress":"**ola\_323@yahoo.com**",

"photostatus":true,

"photouploadviewmodel":null,

}

**only the email address is required for this call.**

to activate profile us the call :

[http://173.160.122.195/AuthenticationService/MembershipService.svc/Rest/activateprofile](http://173.160.122.195/Shell.MVC2.Web.AuthenticationService/MembershipService.svc/Rest/activateprofile)

you can also upload photos with this activation model but for now do the photo upload separate. required feilds are. you can skip the photo upload method if you upload photos in activation or vice versa.

-activationcode,

-email address

all the rest are null;

{ "activationcode":"String content", "emailaddress":"String content", "photostatus":true, "photouploadviewmodel":{ "autoupload":true, "multiple":true, "photosuploaded":[{ "approvalstatusid":2147483647, "caption":"String content", "creationdate":"\/Date(928167600000-0500)\/", "imageb64string":"String content", "imagebytes":[255], "imagename":"String content", "imagetypeid":2147483647, "photostatusid":2147483647, "rejectionreasonid":2147483647 }], "profileid":2147483647 } }

sample call

{

"activationcode":"23232132132134343432434t",

"emailaddress":"oladsadas@yahoo.com",

"photostatus":true,

"photouploadviewmodel":{}

}

end of updates for this section **\*\*\*\*Upadated 7/12/2013 \*\*\*\*\***

2. Creation of FACEBOOK, YAHOO and GOOGLE and other Open ID type profile starts here