Access 4 All

Senior Project - Winter/Spring 2019

Technical Documentation

Access 4 All
Spokane

WELCOME

Table of contents:

Team Contact Information:

Cameron Hilsmann
cameron.hilsmann@hotmail.com

Daniel Paulson
daniel.paulson20@eagles.ewu

Travis Cartmell
tcartmell@eagles.ewu.edu

Anthony Fairfax
apfairfax@yahoo.com

General Technical Information

Project Overview:

We made a mobile app for Access 4 All that displays accessibility information of various locations to a user. Our primary goal is to create a simple, intuitive interface to provide the user with data regarding a business's accessibility standards. The app features a category function and a search function. We prioritized accessibility first, taking into consideration outside software, such as screen readers. Additionally, the visual design was made as easily readable as possible: Intuitive design was prioritized over every other design aspect.

Development overview:

The mobile app retrieves the data from the database developed by the previous team that worked with Access4All, using PHP with Xamarin through Visual Studio. Our goal in designing it with Xamarin is for future proofing, so any future teams can easily work further on the project for other uses and other OS's.

Software Used:

Languages: C#, PHP
IDE: Visual Studio with Xamarin,
Repository: GitHub
NuGet package: Newtonsoft.Json, Xam.Plugin.Geolocator, Xamarin.Android.Support.Design, Xamarin.Essentials

Requirements:

Android 5.0 and up
At least 17 MB available on phone
Internet connection through Android device

Installation for Visual Studio:

First, ensure Visual Studio is installed and up-to-date. Then, with Visual Studio open, go to Tools, then "Get Tools and Features…". Find "Mobile development with .NET" under "Mobile & Gaming" and make sure it's installed. If not install it. Upon completion, you may need to restart your computer for the update to take effect. After that, download the project from Github, and open Access4All.sln in the Access4All-master folder.

Advanced Technical Information

Our program is written in C#, using Xamarin through Visual Studio, and thus, follows programming practices through the two cohesively. Additionally, it interacts with PHP, getting tables from the database. The following describes the basic functionality of each file:

AddressLocator.cs
AddressLocator is an object that contains name, addresses, longitude, and latitude information. Additionally, it can be compared to other AddressLocator objects in terms of the distance. AddressLocator is primarily used to calculate distance.

catAdapter.cs
catAdapter is an Expandable List View that contains all the Category objects in an easily readable format. This allows the selection of individual locations to be streamlined.

Categories.cs
Categories is an object that contains a single title and a list of locations.

detailAdapter.cs
Similar to the catAdapter, dettailAdapter is an Expandable List View that contains all the Detail objects in an easily readable format. This allows accessibility information to be streamlined.

Details.cs
Details is an object that stores a single string of title information.

Location.cs
Location stores a vast quantity of information for a location entry, the name, website, address information, contact information, ID information, and configuration information.

SplashActivity.cs
SplashActivity shows the user the Access4All logo, and while it does so, it loads the information from the database used by the app.

MainActivity.cs
MainActivity contains the main instructions that the app follows.

- *IBackButtonListener* listens for the back button of the phone to be pressed and calls *OnBackPressed*.
- *OnBackPressed* will navigate to whatever previous screen was open in the app.
- *OnCreate* checks and requests for permissions and checks internet connection.
- *OnSaveInstanceState* controls the state of the app when the user changes to a different app or goes to their home screen on their phone.
- *OnActivityResult* is used with voice recognition. It will detect your voice, convert it to a string, and then calls the search with that string.
- *OnNavigationItemSelected* returns a boolean, seeing if the item you've searched for is what you need.

searchFragment.cs

- *OnCreate* obtains the user location if the user gave location permissions.
- *SearchByText* allows for an easily clickable button to call the search view's field and immediately pulls up the user's default keyboard for text search entries.
- *OnCreateView* sets up all the buttons on the view and their respective listeners. This function also disables the Near Me button if user locations aren't given.
- *searchNearMe* is only able to be called if the user location has been obtained. If called, the function will calculate and display all locations within a 15 mile radius of the user's current location. It displays the information cleanly in a list view.
- *MTv_ItemClick* is the listener for the list view after the locations have been pinged from the database and distances calculated.
- *submitQueryListener* takes a string (whether it's voice or keyboard input) and compares that string to location names in the database, and if the string matches or partially matches, it displays all locations with that name together into a listview.
- *InputsMatch* checks if the user's search query is close enough to a location name in the database and if so, returns true.
- *CalculateAddressDistance* takes the list of addresses that were populated in the search and calculates how far they are from the User's location in miles using latitude and longitude.
- *getLocation* gets called only if the User allows for the application to access their personal location. If so, the location gets stored for future calculations and the 'Near Me' button gets activated.
- *searchByVoice* starts an intent to use the phone's voice recognition software and allow for what the user says to be inputted as a string into the submit query.

- *GetData* is the method used to get the establishment ID's from the database, which is used in identifying and searching for locations.
- *RemoveSpecialCharacters* takes the input string from the User and removes any characters that may cause issues while comparing. Most importantly '&' and whitespace.

detaildepthFragment.cs

- *onCreate* gets the clicked on location and it's information selection from the previous view.
- *OnCreateView* sets up all the values to their initialized states and goes to the correct parser depending on the selection in the bundle.
- *getGeolocationAsync* obtains the address of the currently selected location and sends it to the OpenMaps function.
- *OpenMaps* takes the location's name and address and sends it to the maps application on the phone in order to look at it's location in Maps and to potentially get directions.
- *parseRestroomInfo* takes the restroom info table from the database and parses it to the string that will be used in the text view.This is done to make the information readable to the user.
- *parseSeating* takes the seating table from the database and parses it to the string that will be used in the text view. This is done to make the information readable to the user.
- *parseElevators* takes the elevators table from the database and parses it to the string that will be used in the text view. This is done to make the information readable to the user.
- *parseCommunication* takes the communication table from the database and parses it to the string that will be used in the text view. This is done to make the information readable to the user.
- *parseInterior* takes the interior table from the database and parses it to the string that will be used in the text view. This is done to make the information readable to the user.
- *parseMainEntrance* takes the main entrance table from the database and parses it to the string that will be used in the text view. This is done to make the information readable to the user.
- *parseExteriorData* takes the exterior data table from the database and parses it to the string that will be used in the text view. This is done to make the information readable to the user.

- *parseParkingInformation* takes the general table from the database and parses it to the string that will be used in the text view. This is done to make the information readable to the user.
- *parseGeneralInformation* takes the communication table from the database and parses it to the string that will be used in the text view. This is done to make the information readable to the user. It also creates a button to go to the phone's dial up that can be used to directly call the location. It also has a button that sends the user to the location's website, if that specific location has one.
- *convertPhone* takes the phone number string of a location and takes out the hyphens and returns with just numbers in the phone number.
- *parseTransitData* takes the exterior data table from the database and parses it to the string that will be used in the text view. This is done to make the information readable to the user.
- *onBackPressed* allows the phone to use the native back button to go back to the previous screen.
- *GetData* is the method used to get the establishment ID's from the database, which is used in identifying and searching for locations and database tables for parsing.
- *GetDataTable* is the method used to get the establishment ID's and the specific table from the database, which is used in identifying and searching for locations and database tables for parsing.

catagoriesFragment.cs

- *OnCreate* calls the setData function and creates a new catAdapter.
- *onCreateView* sets the adapter and the child click listeners.
- *setData* groups the categories into the right positions and sets the locations in the right groups.
- *HandleSelect* selects the children that have been clicked and sends the user into the detailFragment.

detailFragment.cs

- *OnCreate* calls the setTempData function and creates a new detailAdapter. It grabs the location name and previous view from the bundle.
- *setTempData* groups all the parsing data that will be used in the list view.
- *OnCreateView* initializes the list view and the adapter and the handleSelect click listener.
- *HandleSelect* selects the children that have been clicked and sends the user into the detaildepthFragment.

- *OnBackPressed* allows the phone to use the native back button to go back to the previous screen.

Activity_main.axml
Controls the layout for the Main Activity and, thus, sets the categories and search navigation buttons at the bottom of the screen.

catagoriesLayout.axml
Controls the Expandable List View layout and id in a Linear Layout.

Detail_layout.axml
Controls how the parsed information page will look like. It is a Linear Layout, with a scrollView, that has a single Text View for the parsed information and 3 buttons that are hidden at default. These buttons send the user to the maps application, the phone dialer and the establishment's website if they are able to obtain.

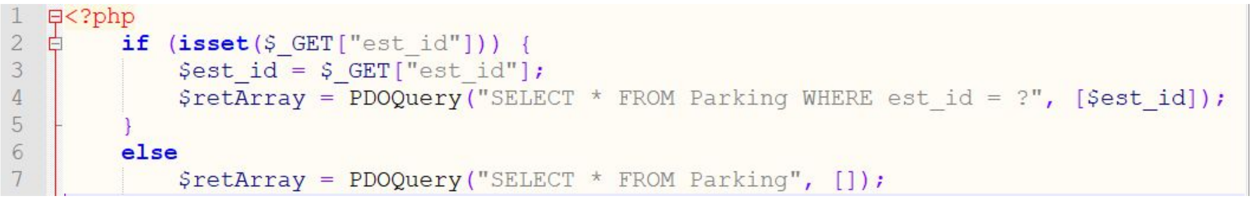
Group_item.axml
Controls the Text View that holds the group name in a Linear Layout.

Item_layout.axml
Controls the Text View that holds an item in the group in a Linear Layout.

searchLayout.axml
Controls the Search View that allows the user to search for a specific location. It has three buttons that can be used to find the nearest locations, search through voice and search through text. This view is in a relative layout.

get.php: for each table on the database

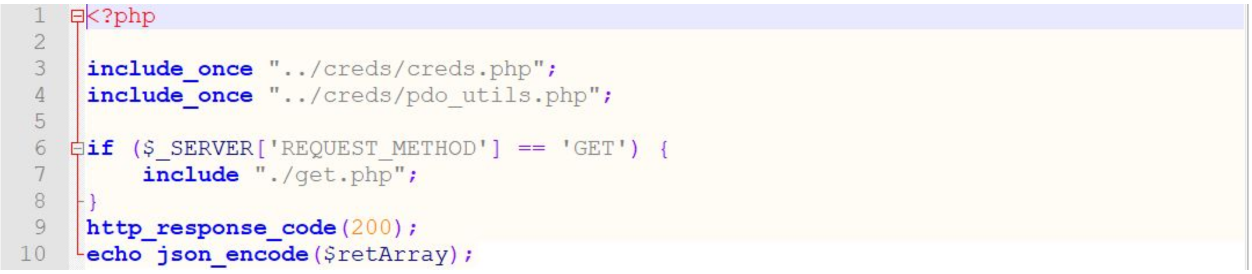
PHP was used behind the website in order to return information from the database. All 22 tables have a similar format:



```php
<?php
    if (isset($_GET["est_id"])) {
        $est_id = $_GET["est_id"];
        $retArray = PDOQuery("SELECT * FROM Parking WHERE est_id = ?", [$est_id]);
    }
    else
        $retArray = PDOQuery("SELECT * FROM Parking", []);
```

index.php

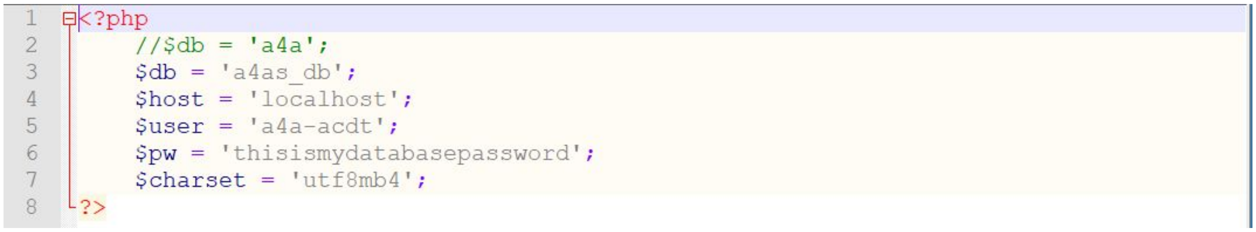
Directs the GET request to where it needs to go. It includes two important files prior to that as well.



```php
<?php

include_once "../creds/creds.php";
include_once "../creds/pdo_utils.php";

if ($_SERVER['REQUEST_METHOD'] == 'GET') {
    include "./get.php";
}
http_response_code(200);
echo json_encode($retArray);
```

creds.php

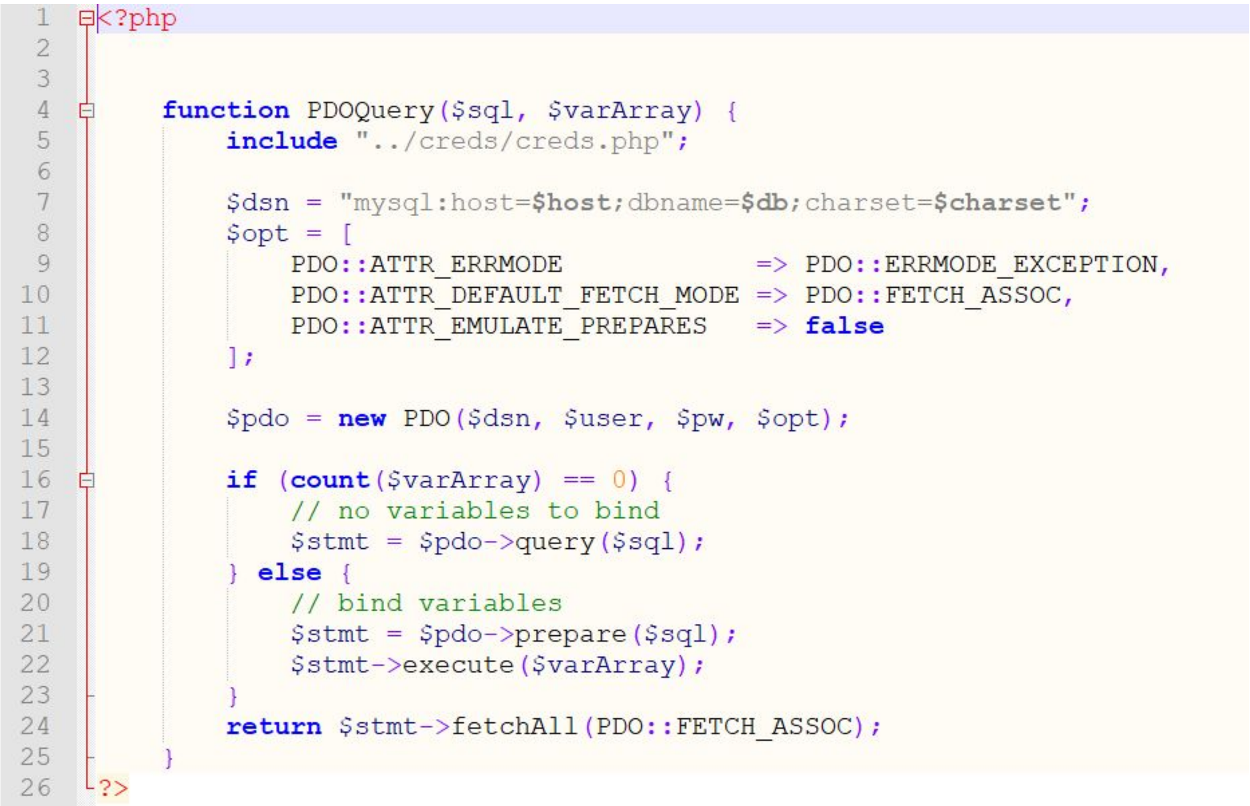
The creds.php contains login info the database and what database it needs to connect to.



```php
<?php
    //$db = 'a4a';
    $db = 'a4as_db';
    $host = 'localhost';
    $user = 'a4a-acdt';
    $pw = 'thisismydatabasepassword';
    $charset = 'utf8mb4';
?>
```

pdo_utils.php

Contains the PDOQuery method, which takes a string and converts it to MySQL while taking care to prevent SQL-Injection.



```php
<?php


    function PDOQuery($sql, $varArray) {
        include "../creds/creds.php";

        $dsn = "mysql:host=$host;dbname=$db;charset=$charset";
        $opt = [
            PDO::ATTR_ERRMODE            => PDO::ERRMODE_EXCEPTION,
            PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
            PDO::ATTR_EMULATE_PREPARES   => false
        ];

        $pdo = new PDO($dsn, $user, $pw, $opt);

        if (count($varArray) == 0) {
            // no variables to bind
            $stmt = $pdo->query($sql);
        } else {
            // bind variables
            $stmt = $pdo->prepare($sql);
            $stmt->execute($varArray);
        }
        return $stmt->fetchAll(PDO::FETCH_ASSOC);
    }
?>
```

All these work together so the website access4allspokane.org can function like an api. For example typing in http://access4allspokane.org/RESTapi/establishment in the URL bar returns a string formatted as JSON that's easy to manipulate:

[{"est_id":"1","name":"Indaba Coffee Roasters","website":"http:\/\/www.IndabaCoffee.com","subtype":"Half Elf","date":"2016-05-10","street":"1425 W. Broadway","city":"Spokane","state":"WA","zip":"99201","phone":"509-443-3566","tty":"","contact_fname":"Hannah","contact_lname":"","contact_title":"Bobby","contact_email":"","user_id":"15","cat_id":"5","config_id":"6","config_comment":"Apartment building"},

{"est_id":"116","name":"Top of the Line Seafood & Burgers","website":"","subtype":"","date":"2016-08-26","street":"24 W 1st St","city":"Cheney","state":"WA","zip":"99004","phone":

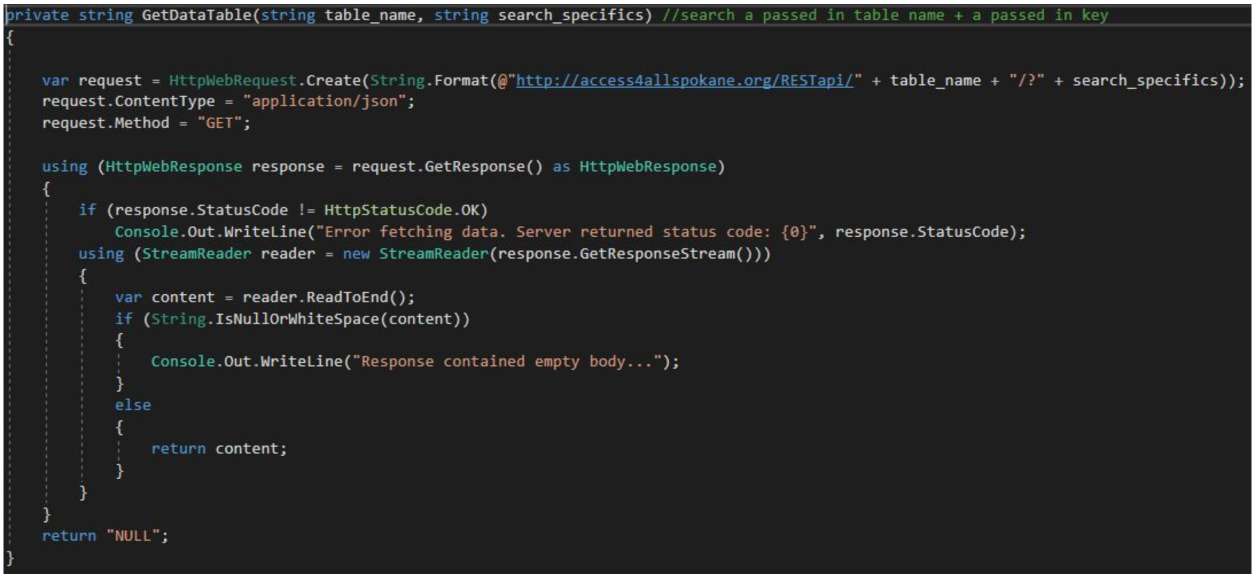
```
"5092354559","tty":"","contact_fname":"LaShay","contact_lname":"
Germain","contact_title":"Owner","contact_email":"","user_id":"1
5","cat_id":"5","config_id":"3","config_comment":""},
{"est_id":"117","name":"Spark
Central","website":"spark-central.org",
"subtype":"","date":"2016-05-16","street":"1214 W. Summit
Parkway","city":"Spokane","state":"WA","zip":"99201","phone":"50
92790299","tty":"","contact_fname":"","contact_lname":"","contac
t_title":"","contact_email":"","user_id":"14","cat_id":"1","conf
ig_id":"1","config_comment":null},

{"est_id":"118","name":"C & H Foreign Auto","website":
"candhforeignauto.com","subtype":"","date":"2016-08-31","street"
:"E. 620 N Foothills Dr.","city":"Spokane","state":"WA",
"zip":"99207",
"phone":"5094879683","tty":"","contact_fname":"","contact_lname"
:"","contact_title":"","contact_email":"","user_id":"14","cat_id
":"2","config_id":"1","config_comment":null},

{"est_id":"119","name":"Goldenwest
Mobility","website":"goldenwestmobility.com","subtype":"","date"
:"2016-05-16","street":"1815 E.
Francis","city":"Spokane","state":"WA",
"zip":"99208","phone":"5094843842","tty":"","contact_fname":"","
contact_lname":"","contact_title":"","contact_email":"","user_id
":"14","cat_id":"2","config_id":"1","config_comment":null},

{"est_id":"120","name":"Grocery
Outlet","website":"groceryoutlet.com","subtype":"grocery
store","date":"2016-05-27","street":"12115 E. Sprague
Avenue","city":"Spokane
Valley","state":"WA","zip":"99206","phone":"5097037185","tty":""
,"contact_fname":"","contact_lname":"","contact_title":"","conta
ct_email":"","user_id":"14","cat_id":"5","config_id":"1","config
_comment":null}]
```

Similarly, it can handle queries as well such as http://access4allspokane.org/RESTapi/parking/?est_id=1 which returns the JSON following string:

[{"park_id":"1","lot_free":"N\/A","street_metered":"Metered","parking_type":"","total_num_spaces":"0","num_reserved_spaces":"0","num_accessable_space":"0","num_van_accessible":"0","reserve_space_sign":"N\/A","reserve_space_obstacles":"N\/A","comment":"Metered Parking - Curb cuts only on Main & Spokane Falls","recommendations":"","est_id":"1"}]

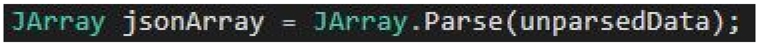
detaildepthFragment.cs

We handle gettings these string in our c# code:



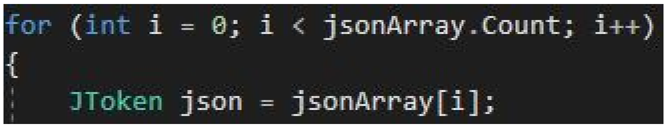
```
private string GetDataTable(string table_name, string search_specifics) //search a passed in table name + a passed in key
{

    var request = HttpWebRequest.Create(String.Format(@"http://access4allspokane.org/RESTapi/" + table_name + "/?" + search_specifics));
    request.ContentType = "application/json";
    request.Method = "GET";

    using (HttpWebResponse response = request.GetResponse() as HttpWebResponse)
    {
        if (response.StatusCode != HttpStatusCode.OK)
            Console.Out.WriteLine("Error fetching data. Server returned status code: {0}", response.StatusCode);
        using (StreamReader reader = new StreamReader(response.GetResponseStream()))
        {
            var content = reader.ReadToEnd();
            if (String.IsNullOrWhiteSpace(content))
            {
                Console.Out.WriteLine("Response contained empty body...");
            }
            else
            {
                return content;
            }
        }
    }
    return "NULL";
}
```

The only problem with this is that it returns a string, thankfully we can convert it to JSON by importing Newtonsoft.Json.Linq and then converting it into a JArray with:



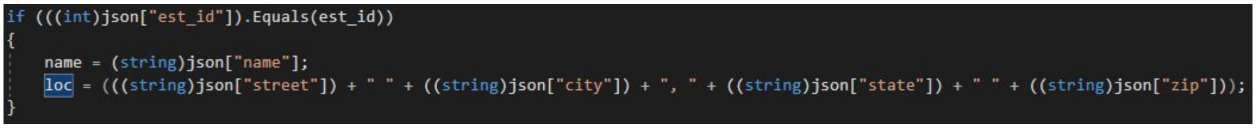
```
JArray jsonArray = JArray.Parse(unparsedData);
```

Then we can convert it to a JToken using:



```
for (int i = 0; i < jsonArray.Count; i++)
{
    JToken json = jsonArray[i];
```

And then finally it can be accessed like a normal JSON object:



```
if (((int)json["est_id"]).Equals(est_id))
{
    name = (string)json["name"];
    loc = (((string)json["street"]) + " " + ((string)json["city"]) + ", " + ((string)json["state"]) + " " + ((string)json["zip"]));
}
```

File Structure:

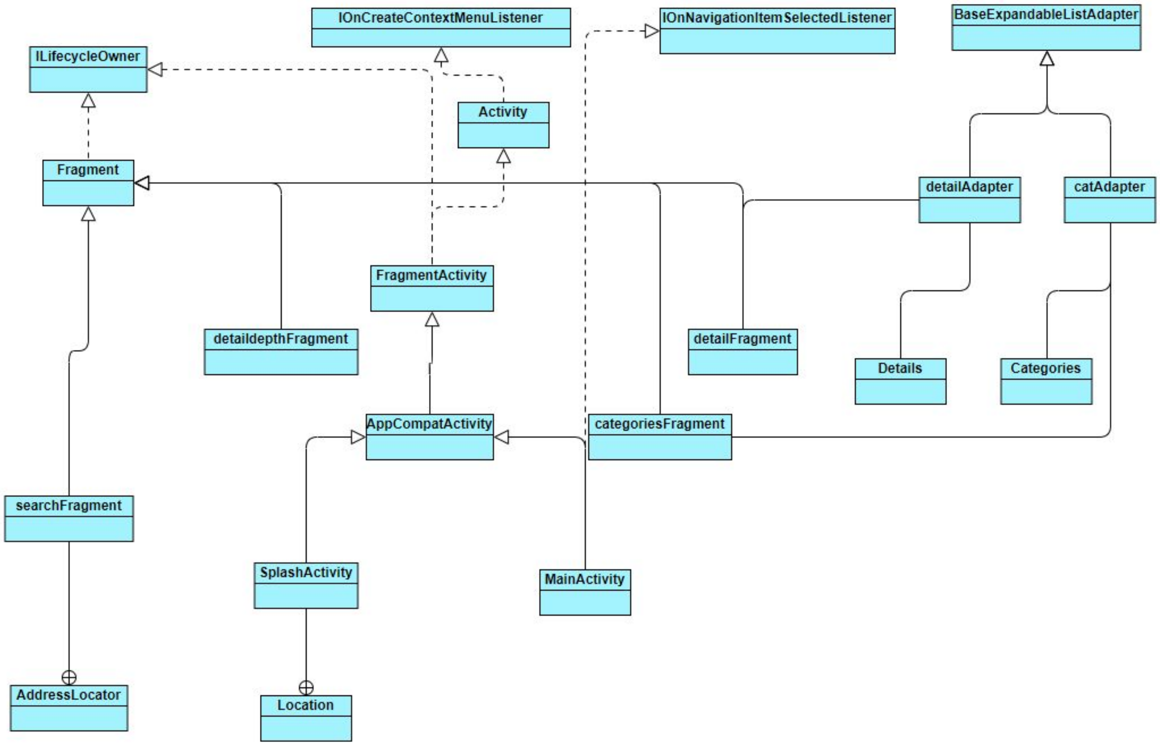
- *Access4All-master*
    - .gitattributes
    - .gitignore
    - Access4All.sln
    - *Access4All*
        - Access4All.csproj
        - catAdapter.cs
        - Categories.cs
        - detailAdapter.cs
        - Details.cs
        - Location.cs
        - MainActivity.cs
        - SplashActivity.cs
        - *Assets*
            - AboutAssets.txt
        - *Fragments*
            - categoriesFragment.cs
            - detaildepthFragment.cs
            - detailFragment.cs
            - searchFragment.cs
        - *ObjectClasses*
            - Category.cs
            - Communication.cs
        - *Properties*
            - AndroidManifest.xml
            - AssemblyInfo.cs
        - *Resources*
            - AboutResources.txt
            - Resource.designer.cs
            - *drawable*
                - A4AS.png
                - access4all_logo.png
                - access4all_name.png
                - ic_dashboard_black_24dp.xml
                - ic_home_black_24dp.xml
                - ic_launcher_background.xml

- ic_notifications_black_24dp.xml
- splash_screen.xml

- *layout*
    - activity_main.axml
    - categoriesLayout.axml
    - detail_layout.axml
    - group_item.axml
    - item_layout.axml
    - searchLayout.axml
- *menu*
    - navigation.xml
- *mipmap-anydpi-v26*
    - ic_launcher.xml
    - ic_launcher_round.xml
    - icon.xml
    - icon_round.xml
- *mipmap-hdpi*
    - ic_launcher.png
    - ic_launcher_foreground.png
    - ic_launcher_round.png
- *mipmap-mdpi*
    - ic_launcher.png
    - ic_launcher_foreground.png
    - ic_launcher_round.png
- *mipmap-xhdpi*
    - ic_launcher.png
    - ic_launcher_foreground.png
    - ic_launcher_round.png
- *mipmap-xxhdpi*
    - ic_launcher.png
    - ic_launcher_foreground.png
    - ic_launcher_round.png
- *mipmap-xxxhdpi*
    - ic_launcher.png
    - ic_launcher_foreground.png
    - ic_launcher_round.png
- *values*
    - colors.xml
    - dimens.xml

- ic_launcher_background.xml
- Strings.xml
- styles.xml

File Structure diagram:


IOnCreateContextMenuListener
IOnNavigationItemSelectedListener
BaseExpandableListAdapter
ILifecycleOwner
Activity
Fragment
detailAdapter
catAdapter
FragmentActivity
detaildepthFragment
detailFragment
Details
Categories
AppCompatActivity
categoriesFragment
searchFragment
SplashActivity
MainActivity
AddressLocator
Location