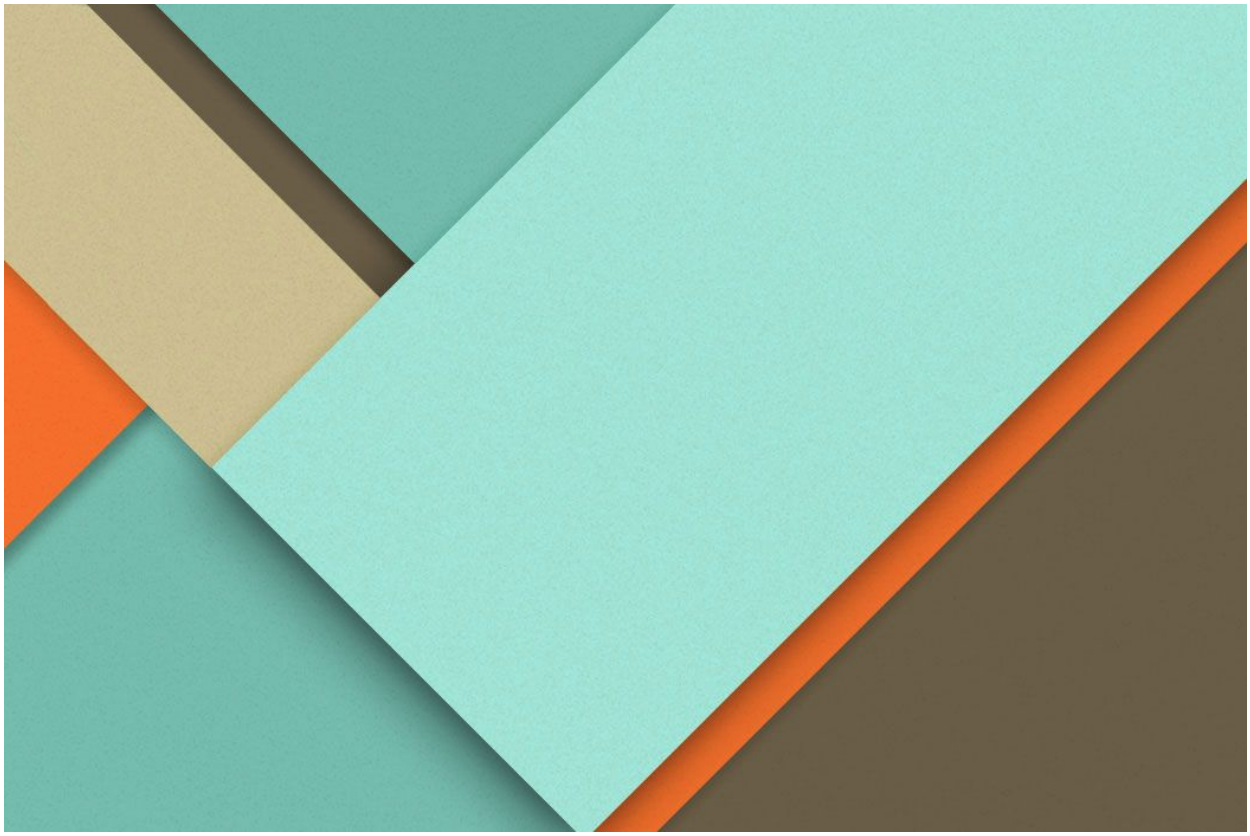


Facebook Graph API

20.08.2017

Suhas B S
Information Technology
3rd Year
Contact no: 7406482161

Overview

External API Integrations are of great significance these days. From providing login portals, to providing analytics, these APIs are a boon to developers.

In this document, integration of the Facebook Graph API with Python Django Server is discussed.

Goals

1. To Login with a Facebook account and display the user's details such as birthday, email, name, gender, profile picture etc. and display it / save it as JSON. Also to fetch the user's photo IDs with album names.
2. To search for another user and display the results

Specifications

In this project, Python v2.7 with Django v1.11 is being used.

Basic Flow

I. Login to your FB account, which generates a user access token on the client side

This is accomplished using the JavaScript SDK of the Graph API, which provides the login portal. After successful login, the access token in automatically generated.

II. Send the generated access token to the server

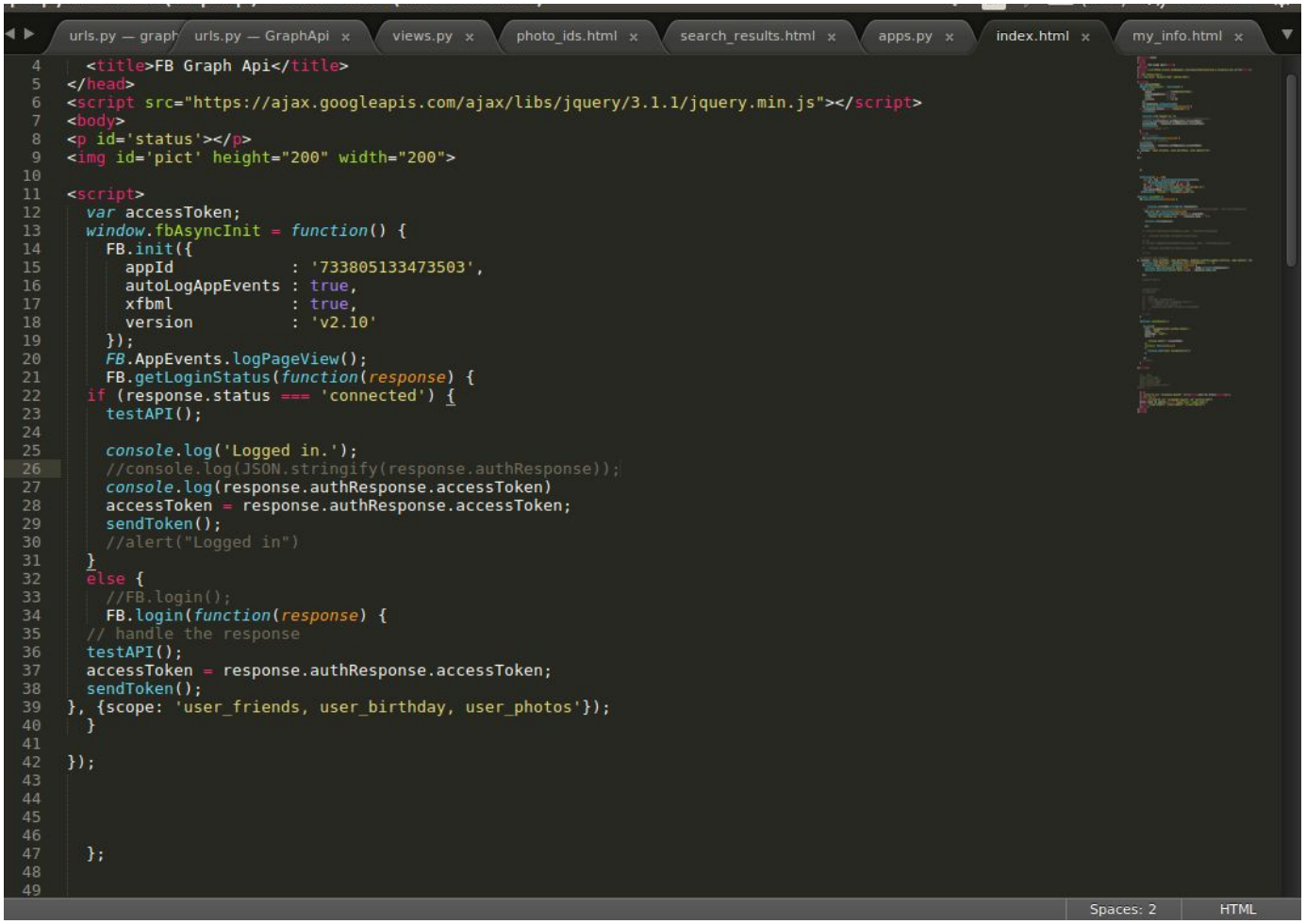
As the access token is now generated, it is best to send it to the server side for future server-server communication. This is accomplished through AJAX, where the access token is passed to server using 'POST' request.

III. With the access token, the required information can be fetched from the FB server using Graph API

Code Specifics

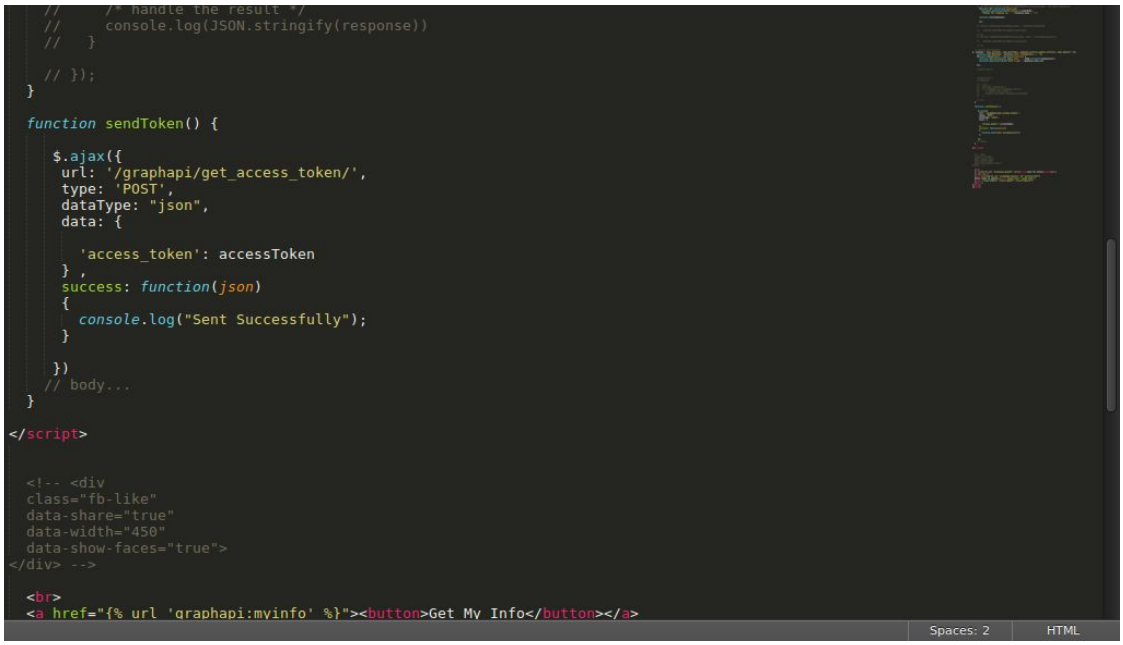
I. Client Side

The first step involves the creation of an app in the facebook developer console. This would provide an AppId which is required for the Graph API.



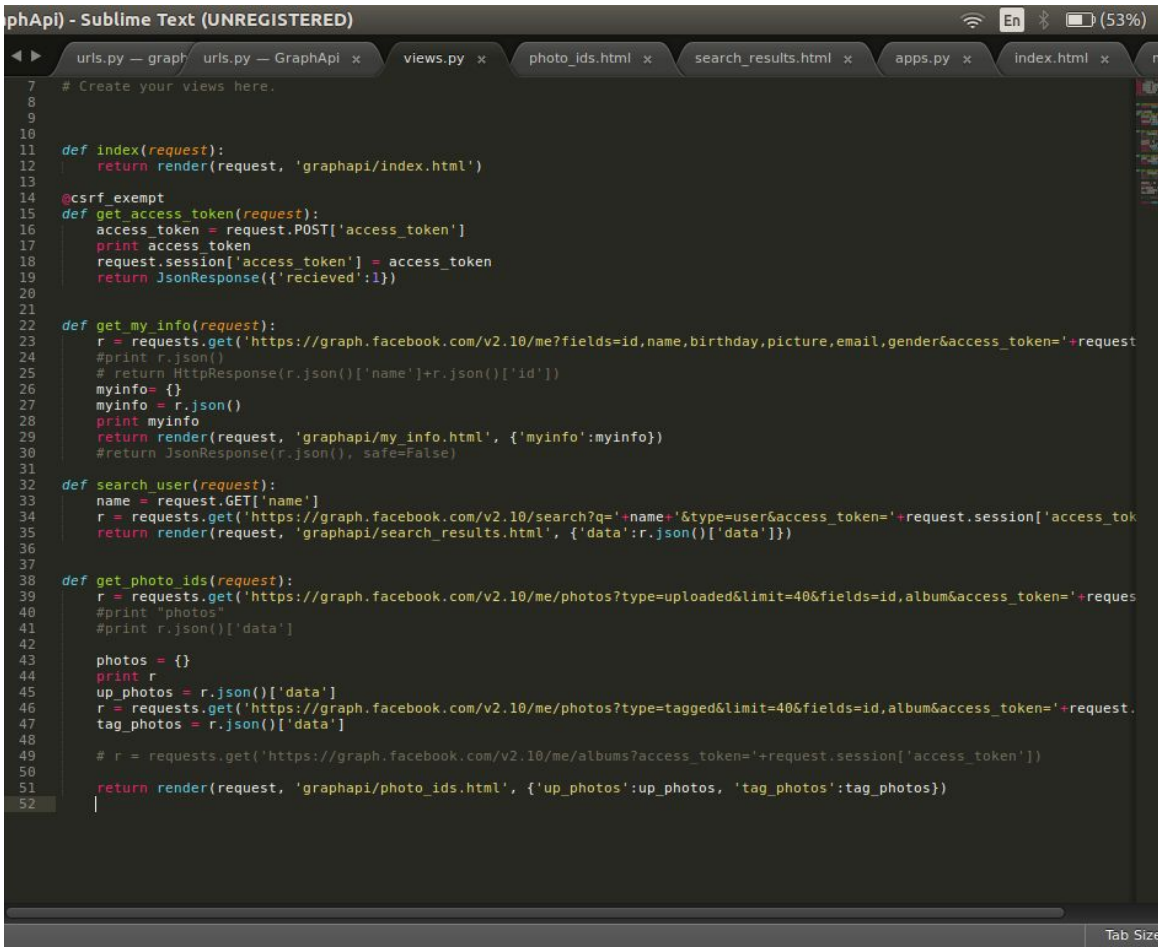
The above screenshot shows a part of the html page which is rendered at startup. The function ***FB.init()*** inside the ***script*** tag initialises the graph API. The first thing which it does is checks whether the user is connected. If not, ***FB.login()*** is called, which opens a login page to login the user.

Now that the access token is generated, it needs to be sent to the server side. This happens when the ***sendToken()*** function is called.



The access token is ‘POST’ed to the server.

II. Server Side

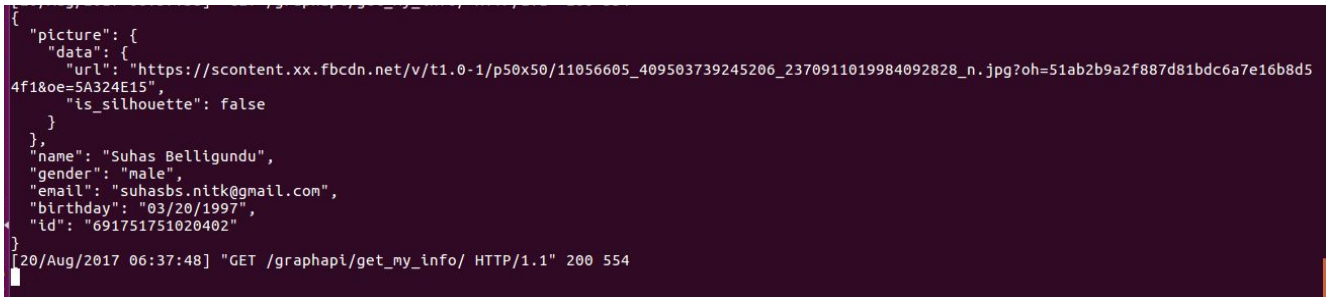


The above screenshot shows the functions in ***views.py*** , which basically performs the actions upon receiving requests and renders respective templates. The function names describe the action performed by each function.

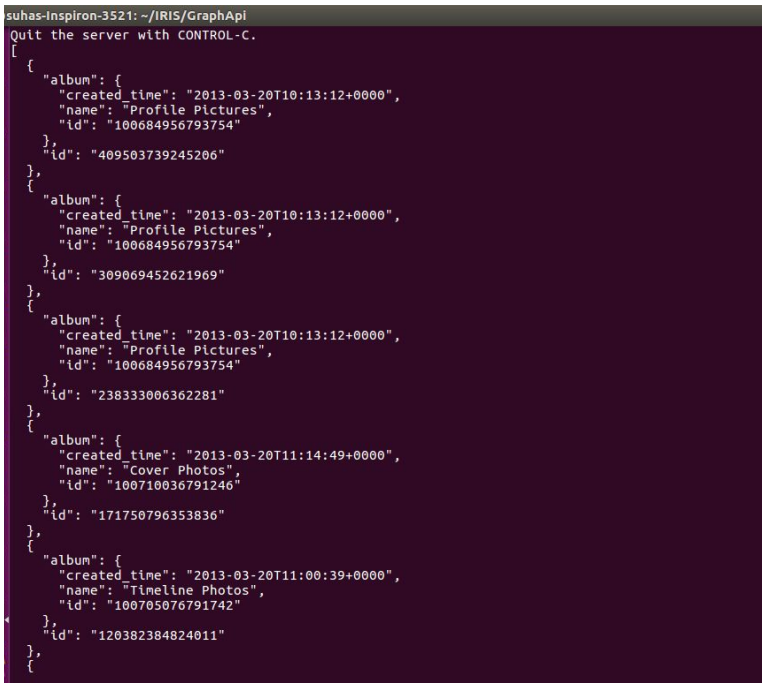
Outputs

JSON Output

1. Get My Information



2. Get Photo IDs



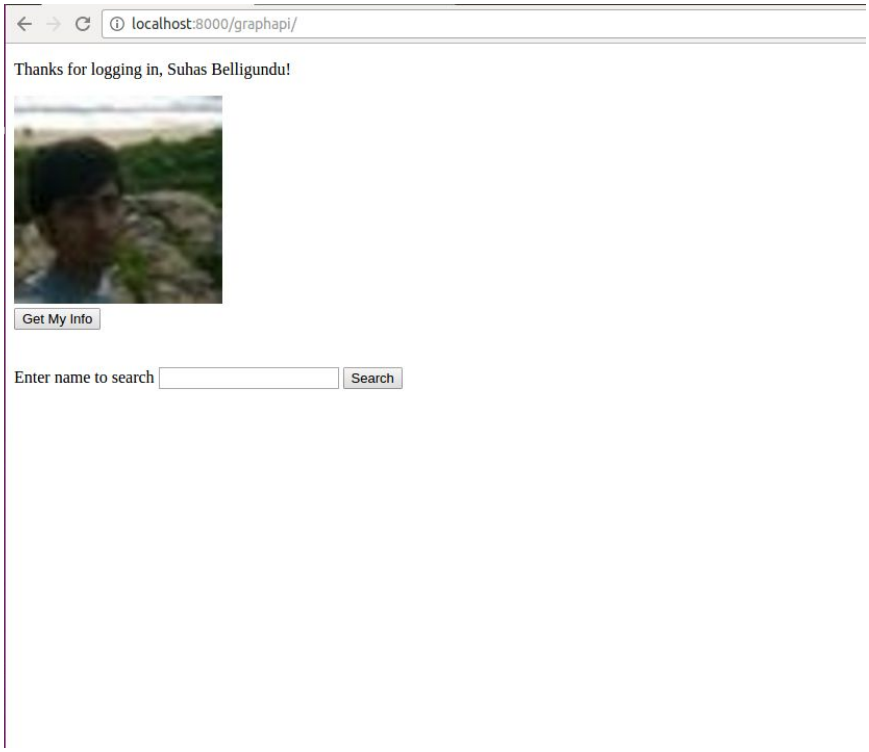
3. Search User

suhas-Inspiron-3521: ~/IRIS/GraphApi

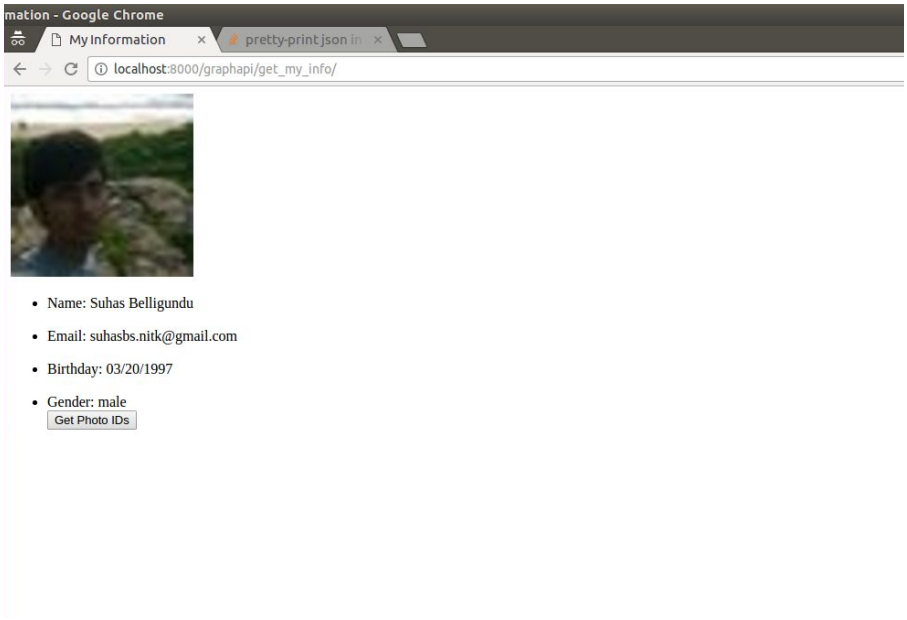

```
[
  {
    "name": "Akshay",
    "id": "1952806954995471"
  },
  {
    "name": "Akshaya",
    "id": "1555298044521007"
  },
  {
    "name": "Gabru Akshay Jat",
    "id": "1087738994697015"
  },
  {
    "name": "Akshaya",
    "id": "1908327319490580"
  },
  {
    "name": "Akshay AK",
    "id": "2019973541568539"
  },
  {
    "name": "Akshaya",
    "id": "472837239738802"
  },
  {
    "name": "Tiwari Akshay",
    "id": "1975946529341497"
  },
  {
    "name": "Akshay Shelke",
    "id": "1477132462381240"
  },
  {
    "name": "Akshay Akshi",
    "id": "1424740237594101"
  },
  {
    "name": "Kartik Akshay Kumar",
    "id": "256278571548814"
  },
  {
    "name": "Vijay Akshay",
```

HTML Page

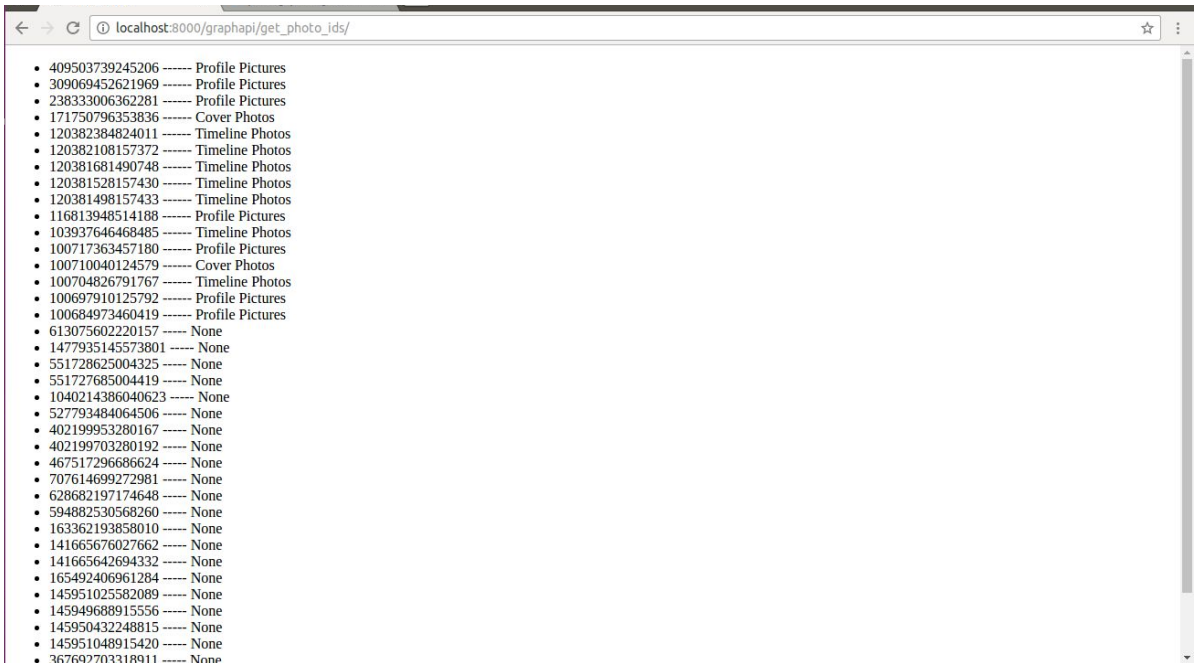
1. Index



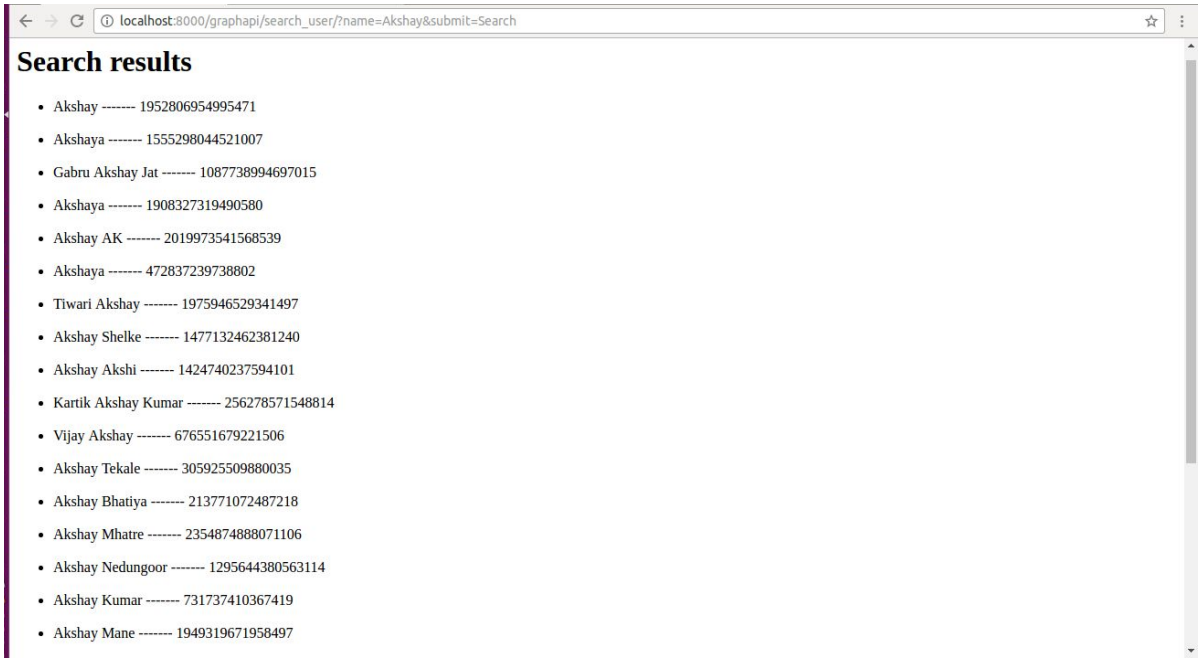
2. Get My Information



3. Get Photo IDs



4. Search User



How to Make it Work

1. Install Django
2. Install requests library for python (pip install requests)
3. Inside the IRIS/GraphApi folder, run the command: ***python manage.py runserver***
4. Open Browser and in the address bar: ***localhost:8000/graphapi***
5. First time, login page will be pop up and the user will be asked to give certain permissions.