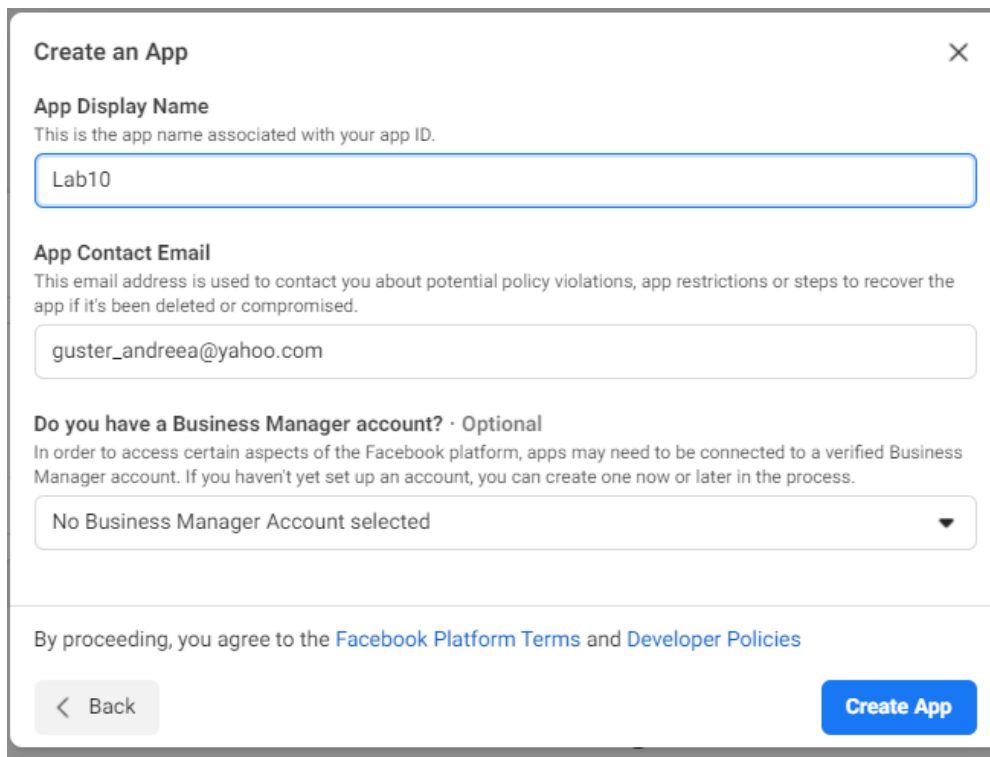


## Laboratorul 10 – Guster Andreea


Vom crea o aplicatie noua care are activat Individual User Accounts (autentificare).

### Autentificare Facebook

- Vom crea un cont pe Facebook for Developers la adresa <https://developers.facebook.com/>.  
Daca aveti **AddBlocker** trebuie sa il dezactivati pentru acest site.
- Din sectiunea **MyApps** (<https://developers.facebook.com/apps>) apasati pe butonul **Create App** si vom bifa **Build Connected Experiences**
- In sectiunea **App Display Name** introduceti numele aplicatiei si apasati pe **Create App**



- Din meniul aplicatiei, selectam **Dashboard**, din sectiunea **Add Products to Your App** mergem pe **Facebook Login** si apasam pe butonul de **Set Up**, apasam pe bulina cu **WWW (Web)**.
- Introduceti adresa site-ului vostru ( <https://localhost:44386/> ).

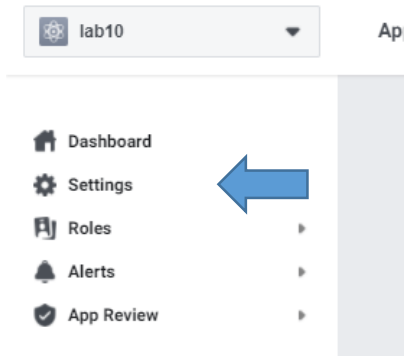


- Din meniu mergeti pe **Settings -> Basic**

De aici vom prelua informatiile de la **App ID** si **App Secret** pentru a le putea folosi în aplicatia web. Scrieti-le într-un fisier text.

App ID: 714359399466498

App Secret: cdd9b8a566bb5fe5390a7e6e8722bc45

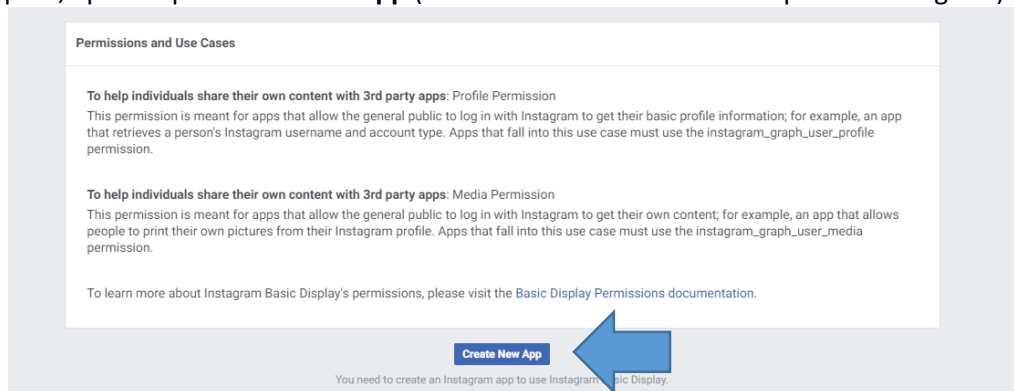


Acum vom deschide proiectul in Visual Studio si in fisierul **App\_Start/Startup.Auth.cs** vom introduce aceste date, mai precis vom comenta liniile ce contin apelul la **UseFacebookAuthentication** si vom introduce în acel apel parametrii obtinuti de pe site, astfel:

```
app.UseFacebookAuthentication(
    appId: "714359399466498",
    appSecret: "cdd9b8a566bb5fe5390a7e6e8722bc45");
```

## Autentificare Instagram

- Vom merge inapoi la **Dashboard** (din meniul din stanga al paginii), iar din sectiune **Add Products to Your App** vom selecta **Instagram Basic Display** apasand pe butonul de **Set up**. Pe pagina care apare, apasam pe **Create New App** (deoarece trebuie sa cream o aplicatie Instagram).



- Vom introduce, ca nume, numele aplicatiei Facebook si vom apasa pe **Create App**. Ca URL-uri introducem un URL standard ce returneaza codul de succes 200, cum ar fi: <https://httpstat.us/200>

Client OAuth Settings

Valid OAuth Redirect URIs

<https://httpstat.us/200>
<https://localhost:44386/Instagram/Show>

Deauthorize

Deauthorize Callback URL

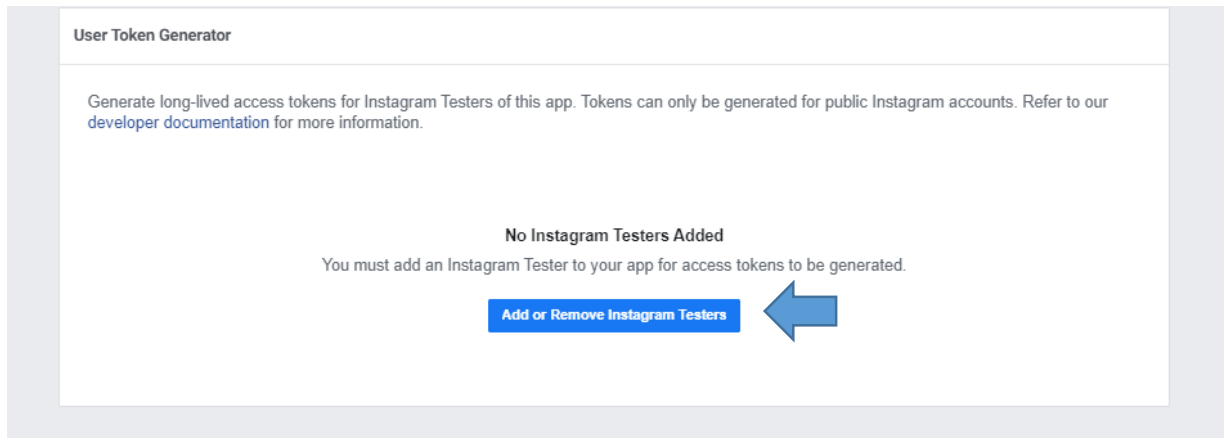
https://localhost:44386/Instagram/Show

Data Deletion Requests

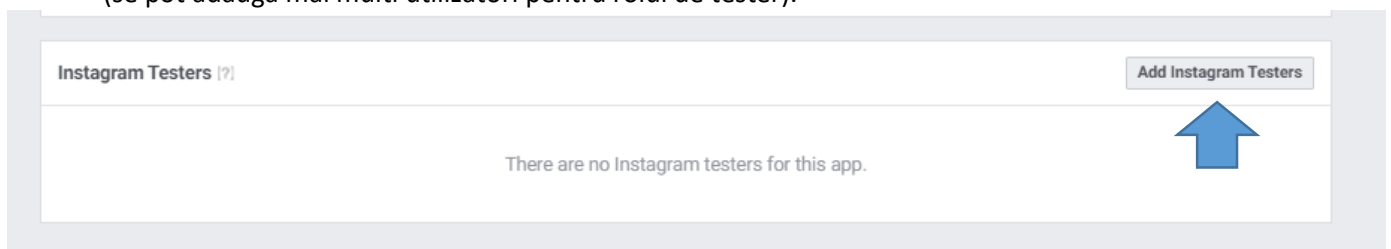
Data Deletion Request URL

https://localhost:44386/Instagram/Show

- În continuare vom adauga un utilizator Instagram ca tester. În meniul din stânga, selectam **Roles, Roles**, sau vom apasa pe butonul **Add or Remove Instagram Tester**, conform imaginii.



- La sectiunea **InstagramTesters**, vom adauga username-ul contului vostru de Instagram si vom apasa pe **Submit**. Adica, am trimis o invitatie pentru testare catre un utilizatorul de Instagram (se pot adauga mai multi utilizatori pentru rolul de tester).



- Acum, utilizatorii care au fost invitati pentru testare trebuie sa acceseze urmatorul link [https://www.instagram.com/accounts/manage\\_access/](https://www.instagram.com/accounts/manage_access/). Cu alte cuvinte, va veti loga in Instagram si veti accepta cererea primita.

## Apps and Websites

ACTIVE EXPIRED REMOVED **TESTER INVITES**

lab10

Decline

Accept

## Accesarea API-ului Instagram dintr-o aplicatie web

Vom crea un controller de tipul MVC numit **InstagramController** cu doua actiuni, **Index** si **Show**. Actiunea **Index** (ce nu are view asociat) va folosi metoda Redirect pentru a accesa URL-ul prin care vrem sa obtinem codul initial de autorizare, însa vom pune în parametrul `redirect_uri` adresa actiunii **Home/Index** (de asemenea, trebuie sa avem grija ca acea adresa sa fie trecuta în setarile aplicatiei la sectiunea Valid Oauth Redirect URIs).

Asadar, actiunea Show va primi ca parametru acel code. Ea va transmite catre view-ul sau o lista de string-uri ce vor reprezenta URL-urile imaginilor ce vor fi afisate de acesta.

```

using Newtonsoft.Json;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Web;
using System.Web.Mvc;

namespace lab10.Controllers
{
    public class InstagramController : Controller
    {
        public ActionResult Index()
        {
            return
Redirect("https://api.instagram.com/oauth/authorize?client_id=1&redirect_uri=https://localhost:44386/Home/
Index&scope=user_profile,user_media&response_type=code");
        }

        public ActionResult Show(string code)
        {
            /*Vom folosi clasa WebClient, aflata în namespace-ul System.Net:*/
            WebClient cl = new WebClient();

            /*Pentru a trimite parametrii sub forma Form URL Encoded, vom
specifica acest lucru în header-ul clientului/
            cl.Headers[HttpRequestHeader.ContentType] = "application/x-www-form-urlencoded";

            /*Vom stoca parametrii ca query string (adaugând parametrul code)
si vom folosi metoda UploadString a clientului pentru a-i
transmite via POST. Rezultatul cererii va fi stocat ca string.*/
            string para =
"redirect_uri=https://localhost:44388/Home/Index&app_id=1&app_secret=2&grant_type=authorization_co
de&code=" + code;
            string HtmlResult = cl.UploadString("https://api.instagram.com/oauth/access_token", para);

            /*transformam stringul JSON într-un dictionar
cheie-valoare. Pentru aceasta, folosim metoda de deserializare a
clasei JsonConvert, situata în namespace-ul Newtonsoft.Json.
Din acest dictionar, vom prelua token-ul de acces.*/
            Dictionary<string, string> dic = JsonConvert.DeserializeObject<Dictionary<string,
string>>(HtmlResult);
            string access_token = dic["access_token"];
            List<string> list_mediaurls = new List<string>();

            /*accesam URL-ul corespunzator pentru a obtine
lista id-urilor postarilor (insa acum vom folosi metoda HTTP
GET, ca urmare folosim metoda DownloadString a clientului):*/
            HtmlResult = cl.DownloadString("https://graph.instagram.com/me/media?access_token=" +
access_token);

            Dictionary<string, object> dic2 = JsonConvert.DeserializeObject<Dictionary<string,
object>>(HtmlResult);

            /* Pe noi ne intereseaza câmpul data, asadar pe acela îl vom converti
la string si îl vom deserializa în continuare la o lista de dictionare
obisnuite:*/
            List<Dictionary<string, string>> list = JsonConvert.DeserializeObject<List<Dictionary<string,
string>>>(dic2["data"].ToString());

            // accesam fiecare postare si sa ii preluam URL-ul:
            foreach (Dictionary<string, string> p in list)
            {
                HtmlResult = cl.DownloadString("https://graph.instagram.com/" + p["id"] +
"?fields=media_url&access_token=" + access_token);
            }
        }
    }
}

```

```

        Dictionary<string, string> dic3 = JsonConvert.DeserializeObject<Dictionary<string,
string>>(HtmlResult);
        list_mediaurls.Add(dic3["media_url"]);
    }
    // vom transmite lista de URL-uri view-ului
    return View(list_mediaurls);
}
}
}

```

In folder-ul **Views/Instagram** vom crea view-ul **Show**.

```
@model List<string>
```

```

@foreach (var p in Model)
{
    
}

```