Laborator 1

In folderul **proiectului**:

urls.py

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path("aplicatie_exemplu/", include("aplicatie_exemplu.urls")),
]
```

Fisierele din folderul **aplicatiei:**

views.py

```
from django.shortcuts import render

# Create your views here.
from django.http import HttpResponse


def index(request):
    return HttpResponse("Primul raspuns")


def pag1(request):
    return HttpResponse(2+3)


l=[]
def pag2(request):
    global l
    a=request.GET.get("a",10)
    print(request.GET)
    l.append(a)
```

```
    return HttpResponse(f"<b>Am primit</b>: {l}")
```

urls.py

```
from django.urls import path

from . import views

urlpatterns = [
    path("", views.index, name="index"),
    path("pag1", views.pag1, name="f"),
    path("pag2", views.pag2, name="g"),
]
```

Pentru obtinerea ip-ului unei cereri

```
def get_ip(request):
    req_headers = request.META
    str_lista_ip = request.META.get('HTTP_X_FORWARDED_FOR')
    if str_lista_ip:
        return str_lista_ip.split(',')[-1].strip()
    else:
        return request.META.get('REMOTE_ADDR')
```

Cerințe

1. Alegeți o temă de proiect. Site-ul va fi unul comercial care vinde produse dintr-o anumită categorie restrânsă (exemple: cofetărie, librărie, restaurant, magazin de instrumente muzicale, magazin de hardware etc.). Deschideți un google docs în care scrieți numele, prenumele și grupa ca titlu (atat în document cât și numele fisierului). Ca subtitlu precizați numele temei de proiect. Veți scrie o descriere scurtă de un paragraf cu privire la ce va conține proiectul. Dați share documentului astfel încât oricine are linkul să poate vizualiza documentația. Adăugați linkul în tabelul de prezență în coloana corespunzătoare.
2. Instalați Django cu ajutorul utilitarului pip. Dacă aveți erori așa, folosiți un mediu virtual. Să se creeze un proiect nou. În cadrul proiectului creați o aplicație cu nume corespunzător temei voastre de proiect. Toate paginile care urmează să fie create vor fi în cadrul acestei aplicații.
3. La accesarea paginii principale (rădăcina aplicației) afișati textul cu descrierea proiectului vostru. Să se scrie o pagina `/info` care generează un html și afișează titlul "Informații despre server" (atât titlul paginii cât și ca heading de nivel 1).

4. Dacă pagina primește parametrul `data`, se afișează o secțiune, cu titlul "Data și ora" în care se afiseaza data și ora curentă pentru server. Data se va afișa în limba română, în fomatul:
   `[zi din saptamana], [zi din luna] [numele lunii] [an],`
   de exemplu `Joi, 2 Octombrie 2025`. Dacă parametrul data primește valoarea `zi`, se afișează doar data, dacă valoarea parametrului e `timp`, se afișează doar ora. Pentru implementare, veți crea o funcție ajutătoare afis_data() care returnează stringul html corespunzător secțiunii (acesta va fi concatenat la textul paginii). Funcția va primi ca parametri valorile prin care decide ce string să returneze.
5. Creați o clasă numită `Accesare` cu proprietățile id, ip_client, url și data. Proprietatea id se autoincrementeaza la crearea unei noi instante a clasei Accesare. Creați metodele:
   a. `lista_parametri()` care returnează o listă de tupluri unde prima valoare e numele parametrului și a doua e valoarea (None daca nu e setat)
   b. `url()` care returnează tot url-ul împreuna cu query string
   c. `data()` care returnează un obiect de tip datetime și primește ca parametru un string de formatare
   d. `pagina()` care returnează doar numele paginii accesate din cadrul aplicatiei (fara domeniu sau parametri). De exemplu, `/` sau `/info`.
6. Sa se scrie o pagină `/log` care afiseaza toate cererile către paginile din site (prima pagina, info si chiar, e însăși, log). Pagina poate primi si parametrul `ultimele` cu o valoare numerică întreagă (altfel pagina va afișa un mesaj de eroare). Se vor afișa ultimele *n* accesări unde n e valoarea parametrului ultimele. Dacă se cere o valoare mai mare decât numărul de accesări, se vor afișa toate și la final un mesaj de eroare cu textul "Exista doar k accesari fata de n accesari cerute", înlocuind k și n cu valorile corespunzătoare.
7. Pagina `/log` poate primi ca parametru și `accesari` cu valoarea "nr" (string) (`/log?accesari=nr`) caz în care va afișa o secțiune suplimentară care afișeaza numarul de accesari de cand a fost pornit serverul (in sesiunea curenta). Se mai pot da si parametrii iduri care primește o enumerare de id-uri separate prin virgula și dubluri, implicit cu valoarea false (deci daca lipsește din query string e considerat false). Parametrul iduri poate sa apare de mai multe ori: /log?iduri=2,3&iduri=5&iduri=4,2,1 In această situație se vor afișa accesarile cu id-urile 2,3,5,4,1 (în această ordine, 2 nefiind afișat a doua oară). Dacă parametrul dubluri e setat la true, se afișează și 2.
8. Parametrul accesari poate primi si valoarea "detalii" afișând sub formă de listă neordonată html data și ora la care s-a realizat fiecare accesare. Pagina info va avea și o sectiune numită "Parametri" în care afiseaza cati parametri a primit si numele acestora.
9. Sa se admita si parametrul tabel care afiseaza intr-un tabel html datele accesarilor. Parametrul tabel poate primi ori valoarea tot (afisand coloane cu toate proprietatile accesarilor), ori o enumerare de nume de proprietati (tabel=id,url va afisa doar coloana cu id-ul si cea cu url-ul)
10. Sub afisarea listei sau, daca e cazul, a tabelului, veti afisa numele paginii care a fost cea mai putin accesata, respectiv a paginii care a fost cel mai mult accesata.