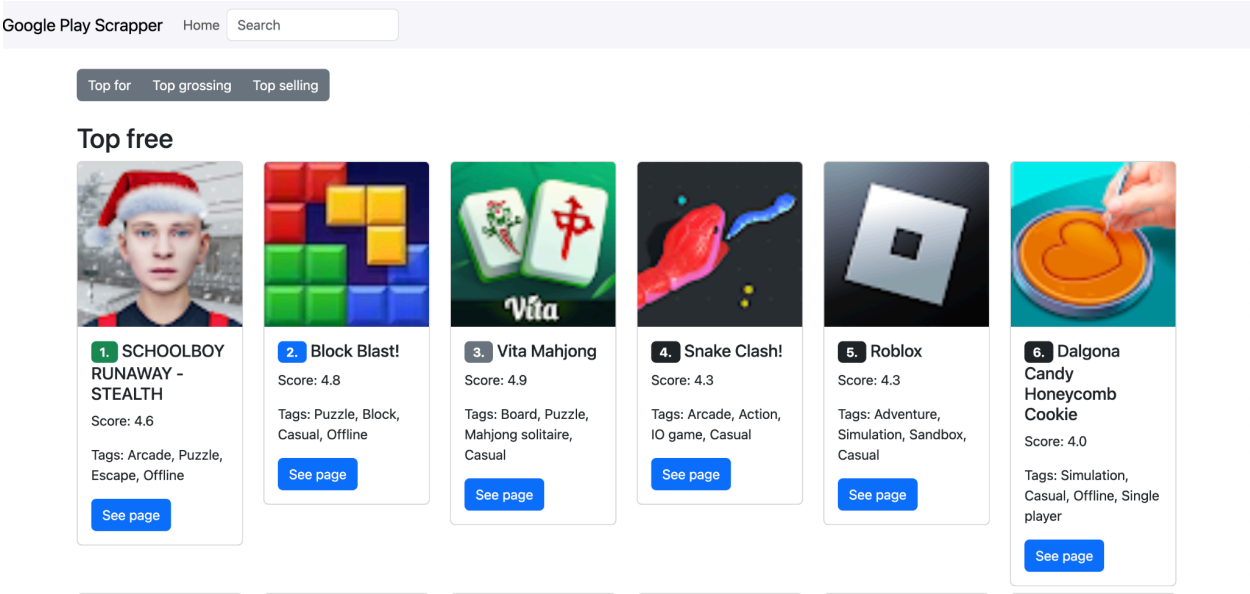
Student: Dulea Mihai-Alexandru
Proiect: Playstore Web Scrapper

Pentru acest proiect am avut de realizat un playstore scrapper care ar fi trebuit sa ia topul aplicatiilor per anumite categorii:

- Top Free
- Top Grossing
- Top Selling

Aplicatia trebuia sa aiba un sistem scheduled prin care sa isi updateze datele privind aplicatiile din baza de date.



Aplicatia a fost realizata folosing framework-ul de backend Django cu python, iar pentru standardizarea aplicatie am folosit Docker. Aplicatia poate afisa o pagina web cu datele despre aplicatii atunci cand navigam la path-ul **http://127.0.0.1:8000/playstore/top_apps/**

Aceasta pagina web este render-uita folosind datele care sunt deja existente in baza de date (este o baza de date simpla de tip mysqllite). Pentru a lua datele efective de pe site-ul celor de la playstore, am cautat prima data daca acestia ofera un API pentru a lua aceste detalii, insa nu exista disponibilitate pentru acest fel de API.

```python
def top_apps(request):
    payload = {}
    for category in Category.objects.all():
        apps = TopApps.objects.filter(category=category).values()
        payload[category.name] = apps

    return render(request, 'top_apps.html', {'payload': payload})
```

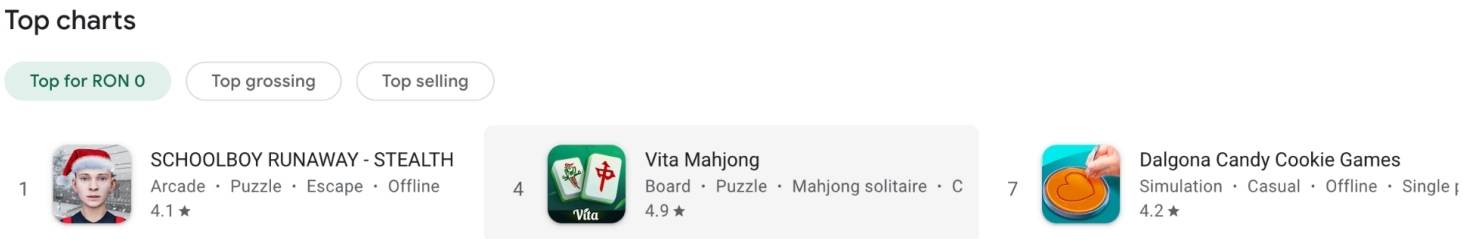
Pentru a face rost totusi de aceste detalii de pe pagina m-am folosit de libraria numita selenium pentru a naviga pe pagina, a apasa pe butoanele existente pe pagina de playstore si a descarca sursa paginii. Dupa care cu ajutorul altei librarii numite BeautifulSoup4 am extras datele necesare din sursa paginii web folosindu-ma de clasele CSS.

```python
def get_top_apps(output_queue: queue.Queue) -> None:
    url = 'https://play.google.com/store/games?hl=en_GB'
    options = webdriver.ChromeOptions()
    options.add_argument('--headless')
    driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()), options=options)
    driver.get(url)

    WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.CLASS_NAME, 'sT93pb.DdYX5.OnEJge'))
    )

    tops = {
        'Top for': [],
        'Top grossing': [],
        'Top selling': []
    }
```

In prima parte a functiei de extragere a datelor din pagina de playstore, am initializat driver-ul care urmeaza sa navigheze pe pagina, l-am pus in modul headless ca sa ne deschida de fapt browser-ul in background, dupa care astept dupa un element anume din pagina ca sa stiu ca pagina a fost generata pana la capat inainte ca sursa sa fie descarcata. Apoi voi declara un dictionar **tops**, care va contine topurile mentionate.

```python
for top_name in tops.keys():
    button = WebDriverWait(driver, 10).until(
        EC.element_to_be_clickable((By.XPATH, f"//span[contains(text(), '{top_name}')]"))
    )
    button.click()
    time.sleep(2)
    soup = BeautifulSoup(driver.page_source, 'html.parser')

    parent_list = soup.select_one('div[jsname="fh1pXc"]')
    if parent_list:
        for application in parent_list.select('.Si6A0c.itIJzb'):
            app = {
                'name': application.select_one('div[jsname="fh1pXc"] .sT93pb.DdYX5.OnEJge').text if application.select_one('div[jsname="fh1pXc"] .sT93pb.DdYX5.OnEJge') else None,
                'score': float(application.select_one('span.sT93pb.CKzsaf .w2kbF').text) if application.select_one('span.sT93pb.CKzsaf .w2kbF') else None,
                'icon': application.select_one('img').get('src') if application.select_one('img') else None,
                'tags': ', '.join([tag.text for tag in application.select('div.ubGTjb .sT93pb.w2kbF')[:-1]]) if application.select('div.ubGTjb .sT93pb.w2kbF') else None,
                'webpage': application.get('href') if application.get('href') else None
            }
            tops[top_name].append(app)
```

Mai departe voi parcurge dictionarul in functie de chei si voi incarca valorile acestuia. Driver-ul o sa apese pe butoanele care contin ca si text valoarea cheii din dictionar, adica va apasa pe butoanele pe care scrie “top for”, “top grossing” si “top selling”



Apoi folosindu-ne de clasele css in care se afla fiecare informatie de care avem nevoie, cum ar fi numele aplicatiei, scorul, categoria, etc., vom umple dictionarul.

In final un dictionar cu aceste valori va contine topurile impreuna cu aplicatiile din acele topuri.

```python
def set_database(top_apps: dict) -> None:
    TopApps.objects.all().delete()

    for category_name, apps in top_apps.items():
        category, _ = Category.objects.get_or_create(name=category_name)
        for app in apps:
            app_obj = TopApps.objects.create(
                name=app['name'],
                category=category,
                score=app['score'],
                icon=app['icon'],
                tags=app['tags'],
                webpage=app['webpage']
            )
            app_obj.save()
```

Metoda de set_databse este folosita pentru a umple baza de date cu valorile din acest dictionar.

Pe langa aceasta parte de web, am implementat si 2 API-uri:

- POST: **api/update-top-apps/** -> O sa faca din nou scrapping si o sa updateze baza de date
- GET: **api/get-top-apps/** -> O sa returneze intr-un format JSON topurile si aplicatiile

```python
def fetch_top_apps():
    payload = {}
    for category in Category.objects.all():
        apps = TopApps.objects.filter(category=category).values()
        pay (variable) payload: dict (apps)  # Convert queryset to list
    return payload

@api_view(['POST'])
def update_top_apps(request):
    top_apps = queue.Queue()
    apps_thread = threading.Thread(target=get_top_apps, args=(top_apps,))
    apps_thread.start()
    apps_thread.join()
    set_database(top_apps.get())
    return JsonResponse({'status': 'success', 'message': 'Top apps updated successfully'})

@api_view(['GET'])
def get_apps(request):
    payload = fetch_top_apps()
    return JsonResponse(payload)
```

```yaml
docker-compose.yml - The Compose specification es
services:
  ▷ Run Service
  django:
    build:
      context: .
      dockerfile: Dockerfile
    container_name: django_app
    ports:
      - "8000:8000"
    volumes:
      - ./django_app:/app

  ▷ Run Service
  flask:
    build:
      context: ./flask
      dockerfile: Dockerfile
    container_name: flask_app
    ports:
      - "5000:5000"
    volumes:
      - ./flask_app:/app
```

Pentru a rula proiectul, in acesta se afla un fisier **docker-compose.yml.** Cu terminalul focusat pe folderul unde se afla acest fisier, daca rulam comanda **docker compose up** o sa ne creeze 2 containere, unul cu aceasta aplicatie de scrapping si una de **flask** care o sa dea scheduled la un minut un request POST catre backend-ul aplicatiei de scrapping, pentru a updata topul aplicatiilor din playstore

(Este posibil sa nu va mearga, eu am lucrat de pe un macbook cu procesor pe arhitectura ARM si este posibil sa nu va creeze imaginea docker bine - daca nu este ok pot sa va prezint pe un meet proiectul**).**

Iar baza de date db.mysqlite arata:

Tabela de categorii:

Rows: 3

	id	name
1	1	Top for
2	2	Top grossing
3	3	Top selling
4		

Tabela cu topul aplicatiilor:

	id	name	score	icon	categ...	webpage
1	441	SCHOOLBOY RUNAWAY - STEALTH	4.6	https://play-lh.googleusercontent.com/yl9hJnKHS9w...	1	/store/apps/details?id=com.LinkedSquad.SchoolBoyR..
2	442	Block Blast!	4.8	https://play-lh.googleusercontent.com/R0qgNDYYHb...	1	/store/apps/details?id=com.block.juggle
3	443	Vita Mahjong	4.9	https://play-lh.googleusercontent.com/uqRUH5NPLig...	1	/store/apps/details?id=com.vitastudio.mahjong
4	444	Snake Clash!	4.3	https://play-lh.googleusercontent.com/Ps9ble2ja0F1...	1	/store/apps/details?id=io.supercent.linkedcubic
5	445	Roblox	4.3	https://play-lh.googleusercontent.com/WNWZaxi9Rd...	1	/store/apps/details?id=com.roblox.client
6	446	Dalgona Candy Honeycomb Cookie	4	https://play-lh.googleusercontent.com/x8yDCrL2O29...	1	/store/apps/details?id=com.dobrogamesdg.squid.can..
7	447	456 Cat Survival Master 3D	4.7	https://play-lh.googleusercontent.com/IWU-gBsc5zK...	1	/store/apps/details?id=com.abi.banana.survival.master
8	448	Extreme Car Driving Simulator	4.4	https://play-lh.googleusercontent.com/ZJHc4l1wm9I...	1	/store/apps/details?id=com.aim.racing
9	449	Mini Games: Calm & Relax	4.6	https://play-lh.googleusercontent.com/pGVAsLdeJSJ...	1	/store/apps/details?id=com.uc.minigame.relax