

Partea 2

Proiect ASO 2021

Rus Mihai Tudorel

Grupa 30643

1.Cerinte rezolvate:

Crearea unui mediu de dezvoltare pentru a crea o aplicatie simpla, un chat public si anume instalarea unei masini virtuale, pe care am instalat sistemul de operare Ubuntu 20.04. Instalarea mediului Django, pentru a putea modela aplicatia si toate metodele din aceasta. Pornirea serverului destinate aplicatiei si rularea catorva comenzi simple prezente pe pagina de ajutor a mediului Django.

Pentru cea de a doua parte a proiectului, am realizat un chat minimalist, unde utilizatorii se pot loga cu username ul propriu intr o anumita camera de chat, pentru a putea comunica cu ceilalti.

2.Modul de rezolvare:

In prima faza, am verificat sa fie prezent pe sistem, biblioteca python fara de care nu am reusi sa rulam codul in Django prin scrierea linii `python3 --version`. Dupa ce ne am asigurat ca avem prezent mediul python instalat, putem incepe prin a instala mediul Django, dar doar intr un virtual environment, pentru a nu instala toate bibliotecile si dependintele Django, daca nu avem nevoie de toate acestea, prin urmatoarele comenzi :

`-pip install virtualenv`

`-python -m virtualenv env`

`-env\Scripts\activate`

`-pip install django`

Odata rulate aceste comenzi, am instalat Django in environment ul env si l am activat. Pentru a crea un nou proiect gol, care are ca si functionalitate pornirea unui server si a unei aplicatii web simpliste este nevoie de urmatoarele comenzi :

`-django-admin startproject asoProject`

`-source activate(in folderul environmentului deja instalat)`

`-python manage.py runserver(in folderul aplicatiei deja create)`

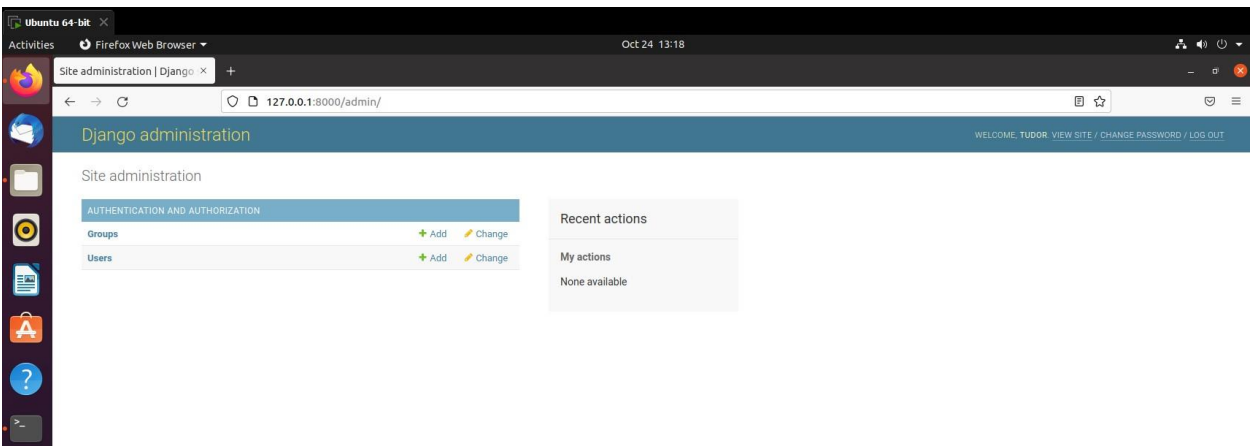
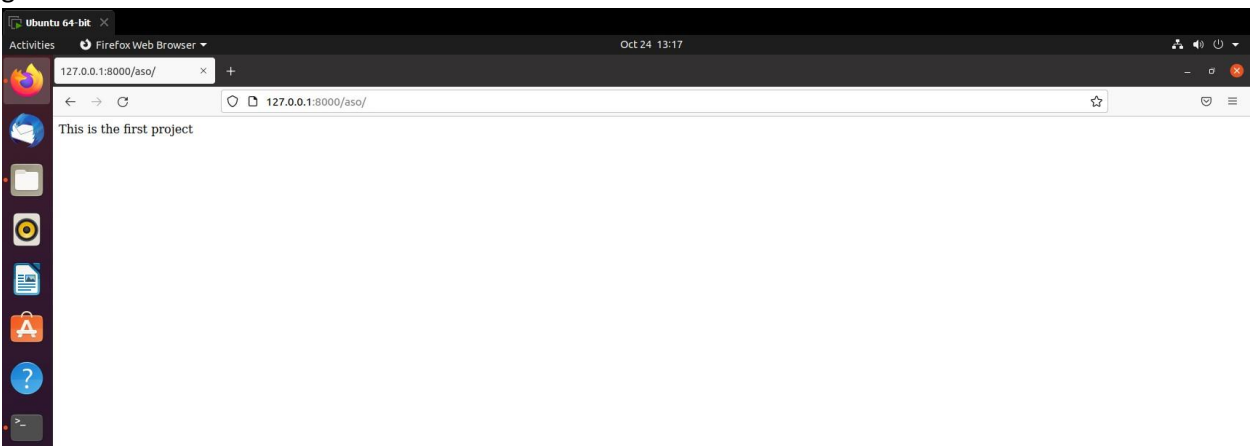
```
Ubuntu 64-bit x
Activities Terminal Oct 24 13:15
tudor@ubuntu: ~/django/aso_proj

tudor@ubuntu:~/django/project_django/bin$ source activate
(project_django) tudor@ubuntu:~/django/project_django/bin$ cd ..
(project_django) tudor@ubuntu:~/django/project_django$ cd ..
(project_django) tudor@ubuntu:~/django$ cd aso_proj
(project_django) tudor@ubuntu:~/django/aso_proj$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
October 24, 2021 - 20:11:42
Django version 3.2.8, using settings 'aso_proj.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.

Not Found: /
[24/Oct/2021 20:12:41] "GET / HTTP/1.1" 404 2165
Not Found: /favicon.ico
[24/Oct/2021 20:12:58] "GET /aso/ HTTP/1.1" 200 25
[24/Oct/2021 20:13:04] "GET /admin/ HTTP/1.1" 200 3326
[24/Oct/2021 20:13:04] "GET /static/admin/css/base.css HTTP/1.1" 304 0
[24/Oct/2021 20:13:04] "GET /static/admin/css/nav_sidebar.css HTTP/1.1" 200 2271
[24/Oct/2021 20:13:04] "GET /static/admin/js/nav_sidebar.js HTTP/1.1" 304 0
[24/Oct/2021 20:13:04] "GET /static/admin/css/dashboard.css HTTP/1.1" 304 0
[24/Oct/2021 20:13:04] "GET /static/admin/css/responsive.css HTTP/1.1" 304 0
[24/Oct/2021 20:13:04] "GET /static/admin/css/fonts.css HTTP/1.1" 304 0
[24/Oct/2021 20:13:04] "GET /static/admin/img/icon-addlink.svg HTTP/1.1" 200 331
[24/Oct/2021 20:13:04] "GET /static/admin/img/icon-changelink.svg HTTP/1.1" 200 380
[24/Oct/2021 20:13:04] "GET /static/admin/fonts/Roboto-Regular-webfont.woff HTTP/1.1" 304 0
[24/Oct/2021 20:13:04] "GET /static/admin/fonts/Roboto-Light-webfont.woff HTTP/1.1" 304 0
[24/Oct/2021 20:13:04] "GET /static/admin/fonts/Roboto-Bold-webfont.woff HTTP/1.1" 304 0
```

Odata ce serverul a pornit, putem sa accesam un browser web, si sa ne conectam la adresa de ip: <http://127.0.0.1:8000/> cu local-host pe portul 8000, unde putem vedea un site minimalist, pentru gestiunea unui site web.



In final pentru a putea avea toate setarile la proiect prezente pe site dar si pe local, ar trebui rulata comanda :

-python manage.py migrate (ca toate datele ce au fost introduce in browser sa fie prezente si in cod)

3. Pentru a putea realiza proiectul am reusit sa creez o serie de urls care sa fie caile catre paginile care se pot deschide in cadrul aplicatiei:

```
chatapp > chat > 🐍 urls.py > ...
1  from django.urls import path
2  from . import views
3
4  urlpatterns = [
5      path('', views.home, name="home"),
6      path('<str:room>/', views.room, name="room"),
7      path('checkview', views.checkview, name="checkview"),
8      path('send', views.send, name="send"),
9      path('getMessages/<str:room>/', views.getMessages, name="getMessages")
10 ]
```

Pentru fiecare URL in parte exista o metoda care implementeaza apelurile de GET, POST si UPDATE pentru useri si camera de chat:

```
def home(request):
    return render(request, 'home.html')

def room(request, room):
    username = request.GET.get('username') # henry
    room_details = Room.objects.get(name=room)
    return render(request, 'room.html', {
        'username': username,
        'room': room,
        'room_details': room_details,
    })

def checkview(request):
    room = request.POST['room_name']
    username = request.POST['username']

    if Room.objects.filter(name=room).exists():
        return redirect('/')+room+'/?username='+username
    else:
        new_room = Room.objects.create(name=room)
        new_room.save()
        return redirect('/')+room+'/?username='+username)
```

4. Stilizarea aplicatiei am realizat o in fisierele home.html si room.html unde am imbinat notiuni de html css si javascript pentru a putea face aplicatia una responsive si user-friendly.

