Administrarea Sistemelor de Operare
Deadline 2- Django Chat

Ardelean Radu - Petru
Grupa 30641 An 4 TI

Cerintele rezolvate

Dezvoltarea unei aplicatii chat web minimalista, in care utilizatorii se pot loga, pot crea o camera de chat si pot trimite mesaje pe chat.

Implementare

Conexiunea cu baza de date SQLite3 se realizeaza cu setarile implicite din *settings.py.*

Pentru implementarea entitatilor de *room* si *message* din baza de date s-au creat modele in *models.py*

```
class Room(models.Model):
    name = models.CharField(max_length=1000)

class Message(models.Model):
    value = models.CharField(max_length=1000000)
    date = models.DateTimeField(default=datetime.now(),blank=True)
    user = models.CharField(max_length=1000000)
    room = models.CharField(max_length=1000000)
```

Pentru salvarea modelelor in baza de date se folosesc comenzile din terminal

```
python manage.py makemigrations , python manage.py migrate
```

Makemigrations va crea migrarile necesare pentru schimbarile din baza de date iar *migrate* va aplica migrarile respective.

Pentru dezvoltarea mai facila a partii de front end am folosit doua templateuri pentru formul de selectare a userului si a camerei si pentru chatul propriu-zis, calea catre *templates* este specificata in *settings.py*

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [BASE_DIR/'templates'],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
```

```
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

In *views.py* am definit views pentru pagina de home si camera de chat.

```
# Create your views here.
def home(request):
    return render(request, 'home.html')

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

Totodata, am definit si views pentru a trimite mesaje, pentru a citi mesajele deja existente in baza de date pentru camera aleasa si pentru a verifica existenta unei camere de chat/crearea acesteia in cazul in care nu exista.

```
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)

def send(request):
    message = request.POST['message']
    username = request.POST['username']
    room_id = request.POST['room_id']

    new_message =
Message.objects.create(value=message,user=username,room=room_id)
    new_message.save()
    return HttpResponse('Message sent.')

def getMessages(request,room):
    room_details = Room.objects.get(name=room)

    messages = Message.objects.filter(room=room_details.id)
    return JsonRespon
```

In fisierul *urls.py* am trecut *url-urile* folosite in aplicatie, folosind url dynamic pentru conexiunea la un chat room specific.

```
urlpatterns = [
    path('',views.home, name='home'),
    path('<str:room>/',views.room,name='room'),
    path('checkview', views.checkview, name='checkview'),
    path('send', views.send, name='send'),
    path('getMessages/<str:room>/',views.getMessages,name='getMessages')

]
```

Bibliografie

1. https://docs.djangoproject.com/en/3.2/intro/tutorial02/#creating-models
2. https://docs.djangoproject.com/en/3.2/intro/tutorial03/
3. https://github.com/tomitokko/django-chat-app