



**UNIVERSITATEA TEHNICĂ**  
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICA ȘI CALCULATOARE

ADMINISTRAREA SISTEMELOR DE OPERARE

# Deploying Django and Docker

Student: Ioana Vîrnă

Profesor îndrumător: Andrei Bogdan  
Leucuta

## Introducere

În acest proiect, va fi realizat un web-site minimal și se va simula lansarea acestuia în producție.

Site-ul va fi realizat folosind framework-ul Django, bazat pe Python.

Realizarea sa va cuprinde 3 etape și anume:

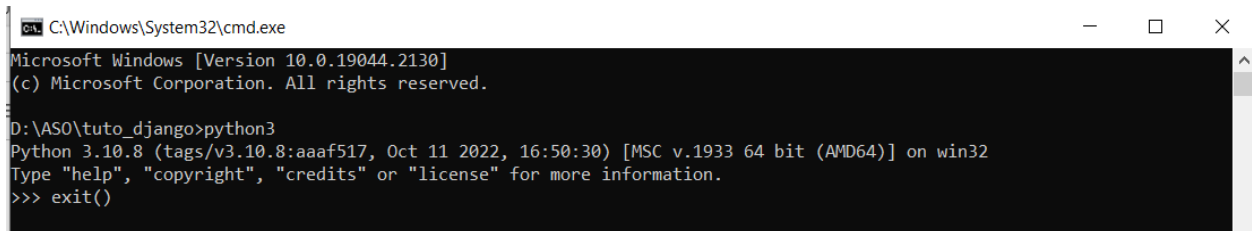
- Etapa 1: Instalare Python, Django și crearea unui site minimalist
- Etapa 2: Crearea unui chat minimalist
- Etapa 3: Presupunând că site-ul este funcțional, îl pregătim de lansarea în piață

## Etapa I

În această etapă se vor instala resursele necesare în realizarea temei și se vor implementa cerințele minime, având astfel o bază a proiectului.

Pasi pentru crearea proiectului de baza:

1. Verificarea existenței **python3**. Dacă nu există, vom fi ghidați să îl instalăm.



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.


D:\ASO\tuto_django>python3
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()
```

2. Deschidem linia de comandă din locația unde vom dori să creăm proiectul



**UNIVERSITATEA TEHNICĂ**  
DIN CLUJ-NAPOCA

3. Cream un director numit `venv_django`, utilizand comanda **`python3 -m venv venv_django`**

 C:\Windows\System32\cmd.exe

```
D:\ASO>cd tuto_django  
  
D:\ASO\tuto_django>python3 -m venv venv_django  
  
venv_django 24 oct. 2022 18:50
```

4. Activam virtual environment-ul creat. Se poate observa ca este activat atunci cand in fata prompt-ului obisnuit de CMD, avem intre paranteze numele al virtual environmentului.

```
D:\ASO\tuto_django>venv_django\Scripts\activate  
  
(venv_django) D:\ASO\tuto_django>pip list  
Package      Version  
-----  
pip           22.2.2  
setuptools    63.2.0  
  
[notice] A new release of pip available: 22.2.2 -> 22.3  
[notice] To update, run: python.exe -m pip install --upgrade pip
```

5. Instalam modulele Django si djangorestframework utilizand comanda **`pip`**

```
(venv_django) D:\ASO\tuto_django>cd venv_django  
  
(venv_django) D:\ASO\tuto_django\venv_django>pip install django djangorestframework  
Collecting django  
  Using cached Django-4.1.2-py3-none-any.whl (8.1 MB)  
Collecting djangorestframework  
  Using cached djangorestframework-3.14.0-py3-none-any.whl (1.1 MB)  
Collecting tzdata  
  Using cached tzdata-2022.5-py2.py3-none-any.whl (336 kB)  
Collecting asgiref<4,>=3.5.2  
  Using cached asgiref-3.5.2-py3-none-any.whl (22 kB)  
Collecting sqlparse>=0.2.2  
  Using cached sqlparse-0.4.3-py3-none-any.whl (42 kB)  
Collecting pytz  
  Using cached pytz-2022.5-py2.py3-none-any.whl (500 kB)  
Installing collected packages: pytz, tzdata, sqlparse, asgiref, django, djangorestframework  
Successfully installed asgiref-3.5.2 django-4.1.2 djangorestframework-3.14.0 pytz-2022.5 sqlparse-0.4.3 tzdata-2022.5  
  
[notice] A new release of pip available: 22.2.2 -> 22.3  
[notice] To update, run: python.exe -m pip install --upgrade pip
```

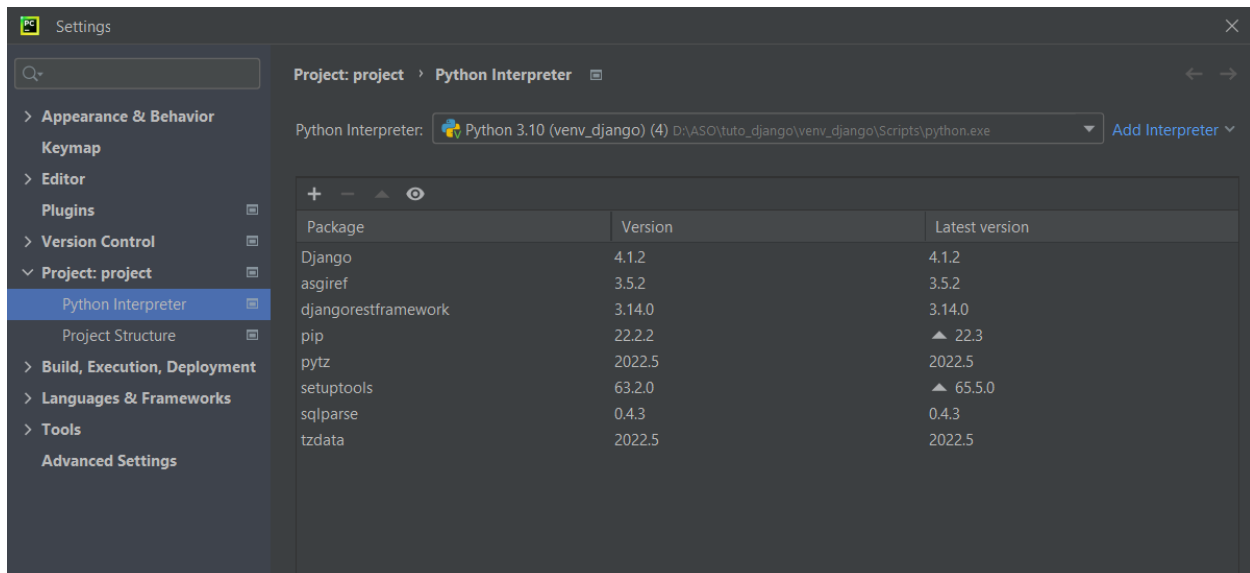


**UNIVERSITATEA TEHNICĂ**  
DIN CLUJ-NAPOCA

6. Din directorul care contine directorul cu virtual environmental creat, initiem un proiect Django cu ajutorul comenzii **Django-admin startproject project**

```
(venv_django) D:\ASO\tuto_django\venv_django>cd ..  
  
(venv_django) D:\ASO\tuto_django>django-admin startproject project
```

7. Folosind mediul de dezvoltare **PyCharm**, deschidem directorul tuto\_django/project. Configuram din setari interpretorul de python (mediul virtual creat mai devreme)



8. Invocam pornirea serverului utilizand comanda **python manage.py runserver** din terminalul PyCharm. Se va crea o aplicatie web Django, insa fara o baza de date.

```
PS D:\ASO\tuto_django\project> python manage.py runserver  
Watching for file changes with StatReloader  
Performing system checks...  
  
System check identified no issues (0 silenced).  
  
You have 18 unapplied migration(s). Your project may not work properly until you apply t  
he migrations for app(s): admin, auth, contenttypes, sessions.  
Run 'python manage.py migrate' to apply them.  
October 24, 2022 - 18:57:03  
Django version 3.2, using settings 'project.settings'  
Starting development server at http://127.0.0.1:8000/  
Quit the server with CTRL-BREAK.
```



9. Cream aplicatia  
**python**

**scumboard**. Dupa ce s-a creat, in sectiunea **INSTALLED\_APPS** din **setting.py** vom adauga si 'scumboard'.

scumboard prin comanda  
**manage.py startapp**

```
PS D:\AS0\tuto_django\project> python.exe .\manage.py startapp scumboard
PS D:\AS0\tuto_django\project> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...
```

Name	Date modified	Type
Quick access		
Desktop		
Downloads		
.idea	24 oct. 2022 19:02	File folder
project	24 oct. 2022 18:58	File folder
scumboard	24 oct. 2022 19:02	File folder

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'scumboard',
]
```

10. Definim modelele aplicatiei, **List** si **Card** in **models.py** din directorul **scumboard**.

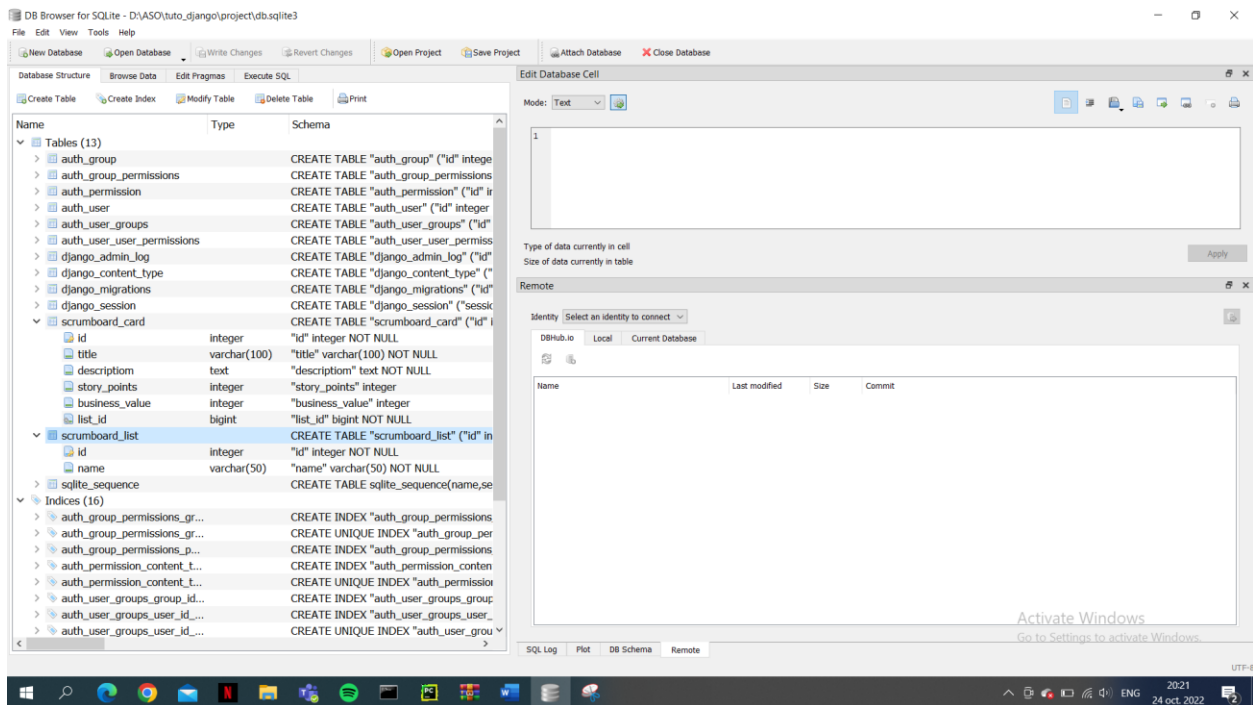
```
settings.py x models.py x
1 from django.db import models
2
3 # Create your models here.
4 class List(models.Model):
5     name=models.CharField(max_length=50)
6
7 class Card(models.Model):
8     title = models.CharField(max_length=100)
9     description=models.TextField(blank=True)
10
11 # relation with foreign key below
12 # Each Card must belong to a list
13 list=models.ForeignKey(List, related_name="cards", on_delete=models.CASCADE)
14 story_points = models.IntegerField(null=True, blank=True)
15 business_value=models.IntegerField(null=True, blank=True)
```

11. Apelam comanda **python manage.py makemigrations**, urmata de **python manage.py migrate**. Makemigrations doar pregateste baza de date pentru manipulare, migrate o modifica efectiv.

```
You have 18 unapplied migration(s). Your project may not work properly until you apply t
he migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.

Applying auth.0006_require_contenttypes_0002... OK
Applying auth.0007_alter_validators_add_error_messages... OK
Applying auth.0008_alter_user_username_max_length... OK
Applying auth.0009_alter_user_last_name_max_length... OK
Applying auth.0010_alter_group_name_max_length... OK
Applying auth.0011_update_proxy_permissions... OK
Applying auth.0012_alter_user_first_name_max_length... OK
Applying scrumboard.0001_initial... OK
Applying sessions.0001_initial... OK
PS D:\ASO\tuto_django\project>
```

12. Daca ne uitam in SQLite, vom putea vedea tabelele scrumboard\_card si scrumboard\_list unde vom putea vedea campurile definite in clasele List si Card din models.py



The screenshot displays the DB Browser for SQLite interface. On the left, the 'Database Structure' pane shows a list of tables and their schemas. The 'Tables (13)' section includes:

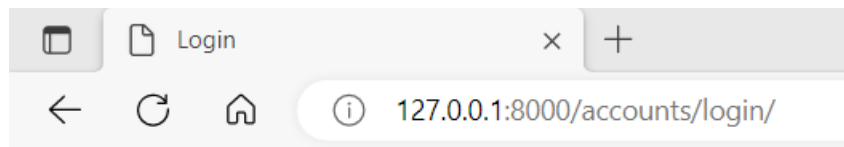
- auth\_group: CREATE TABLE "auth\_group" ("id" integer NOT NULL, "name" varchar(150) NOT NULL, PRIMARY KEY ("id"))
- auth\_group\_permissions: CREATE TABLE "auth\_group\_permissions" ("group\_id" integer NOT NULL, "permission\_id" integer NOT NULL, PRIMARY KEY ("group\_id", "permission\_id"))
- auth\_permission: CREATE TABLE "auth\_permission" ("id" integer NOT NULL, "content\_type\_id" integer NOT NULL, "codename" varchar(100) NOT NULL, PRIMARY KEY ("id"))
- auth\_user: CREATE TABLE "auth\_user" ("id" integer NOT NULL, "username" varchar(150) NOT NULL, "password" varchar(128) NOT NULL, "email" varchar(254) NULL, PRIMARY KEY ("id"))
- auth\_user\_groups: CREATE TABLE "auth\_user\_groups" ("user\_id" integer NOT NULL, "group\_id" integer NOT NULL, PRIMARY KEY ("user\_id", "group\_id"))
- auth\_user\_user\_permissions: CREATE TABLE "auth\_user\_user\_permissions" ("user\_id" integer NOT NULL, "permission\_id" integer NOT NULL, PRIMARY KEY ("user\_id", "permission\_id"))
- django\_admin\_log: CREATE TABLE "django\_admin\_log" ("id" integer NOT NULL, "user\_id" integer NOT NULL, "content\_type\_id" integer NOT NULL, "object\_id" integer NOT NULL, "action\_flag" integer NOT NULL, "change\_message" text NOT NULL, PRIMARY KEY ("id"))
- django\_content\_type: CREATE TABLE "django\_content\_type" ("id" integer NOT NULL, "app\_label" varchar(100) NOT NULL, "model" varchar(100) NOT NULL, PRIMARY KEY ("id"))
- django\_migrations: CREATE TABLE "django\_migrations" ("app" varchar(255) NOT NULL, "migration" varchar(255) NOT NULL, PRIMARY KEY ("app", "migration"))
- django\_session: CREATE TABLE "django\_session" ("session\_key" varchar(40) NOT NULL, "session\_data" text NOT NULL, "expire\_date" datetime NOT NULL, PRIMARY KEY ("session\_key"))
- scrumboard\_card: CREATE TABLE "scrumboard\_card" ("id" integer NOT NULL, "title" varchar(100) NOT NULL, "description" text NOT NULL, "story\_points" integer NOT NULL, "business\_value" integer NOT NULL, "list\_id" bigint NOT NULL, PRIMARY KEY ("id"))
- scrumboard\_list: CREATE TABLE "scrumboard\_list" ("id" integer NOT NULL, "name" varchar(50) NOT NULL, PRIMARY KEY ("id"))

The 'Indices (16)' section shows various indexes and unique constraints for these tables. The 'Edit Database Cell' window is open, showing a text input field.

## **Etapa II**

Scopul acestei etape este de a realiza un chat minimalist unde se pot conecta utilizatori si pot trimite si primi mesaje.

Am inceput cu crearea sistemului de conectare la chat. Utilizatorul este creat doar de un admin, ulterior putandu-se sa se conecteze accesand <http://127.0.0.1:8000/accounts/login/>



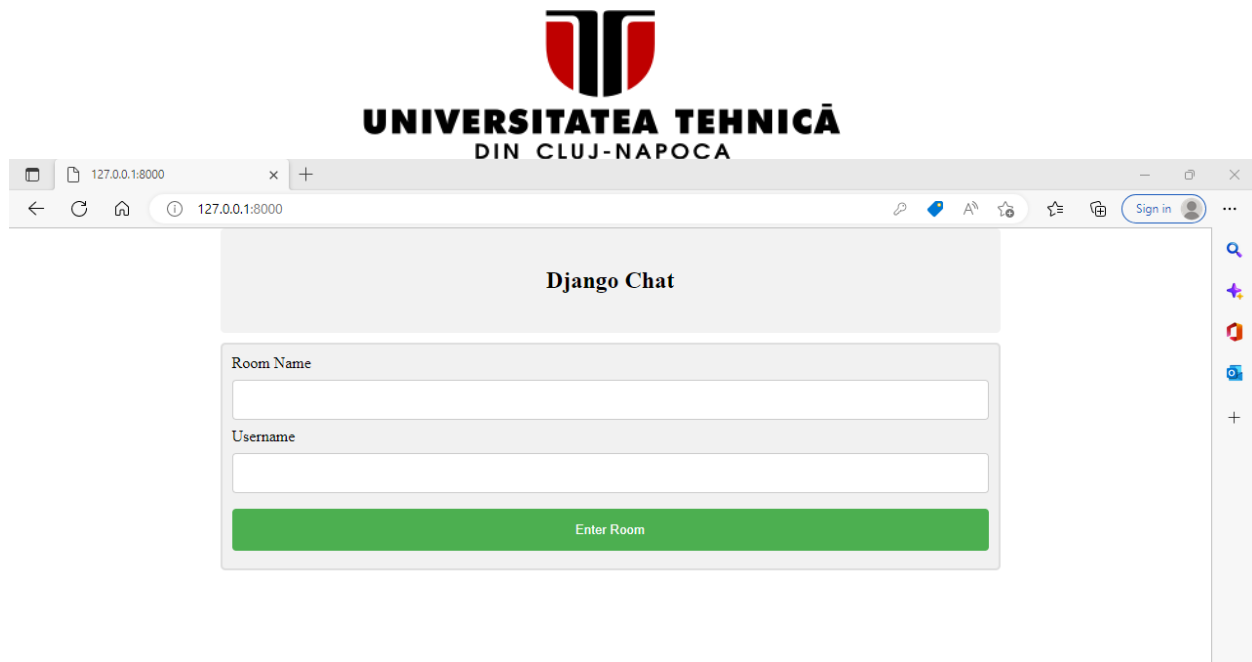
### **Log In**

Username:

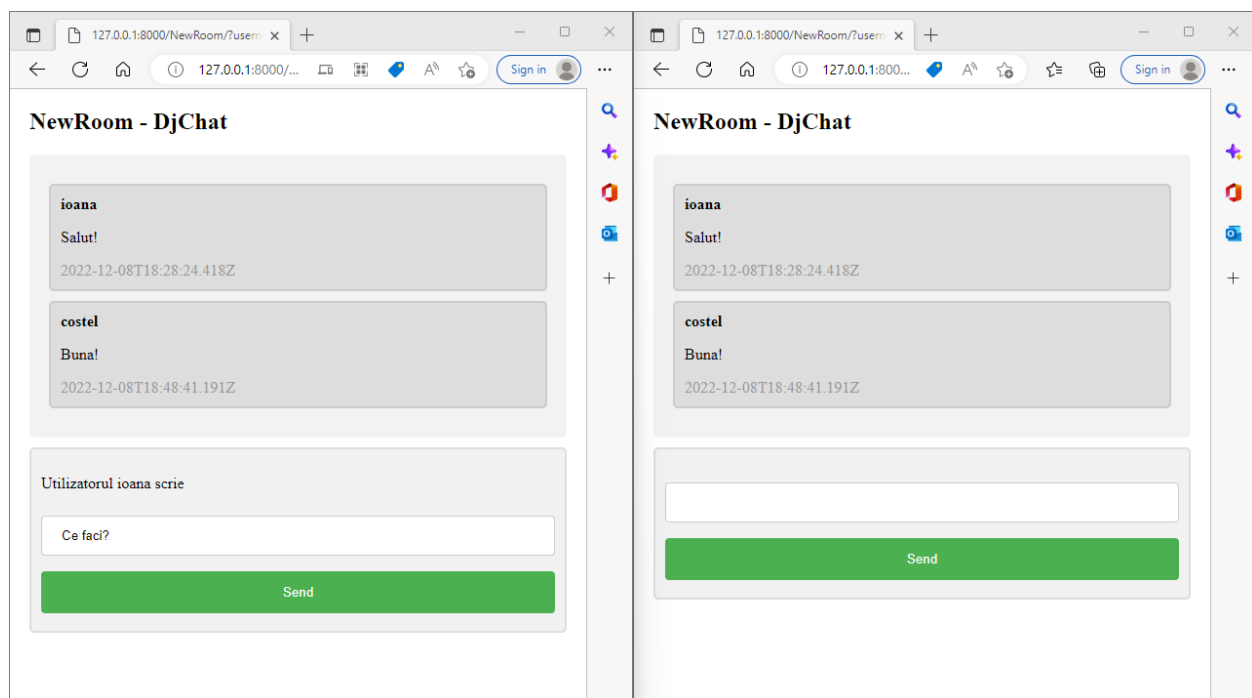
Password:

Dupa conectare, acesta poate intra intr-o camera de chat. Camera poate exista sau nu, insa in cazul in care camera ceruta nu e create, in momentul in care se introduce un nume nou de camera, aceasta se creaza.

De asemenea, utilizatorul poate alege numele sub care sa se conecteze la chat.



Camera de chat este prezentata in urmatoarea poza, unde se poate vedea ca in momentul in care un utilizator scrie, este semnalat acest lucru.



Rularea aplicatiei se va face cu comanda: `python manage.py runserver`





**UNIVERSITATEA TEHNICĂ**  
DIN CLUJ-NAPOCA

### **Probleme intalnite si modul de rezolvare**

M-am ghidat dupa 2 tutoriale gasite pe internet (se va face referire la ele in bibliografie). Am avut probleme cu cateva dintre metodele de trimis/primit mesaje, insa si cu partea de login.

Am incercat sa realizez sa apara la ambii utilizatori cand cineva scrie ceva, insa nu am reusit sa leg prin websocket-uri asta.

### **Concluzii**

Am dobandit cunostinte stabile despre rutare si implementarea unei aplicatii de chat in timp real 😊