Web Project

Cinema

20th March 2022

Deliverable 1

Degree in Computer Engineering

University of Lleida

Course 2021-2022

Members of the team:

1. Gerard Monsó - 48252494N

2. Daniel Bistuer - 78094395B

3. Marc Farran – 48059141C

4. Marc Vivas - 78101067J

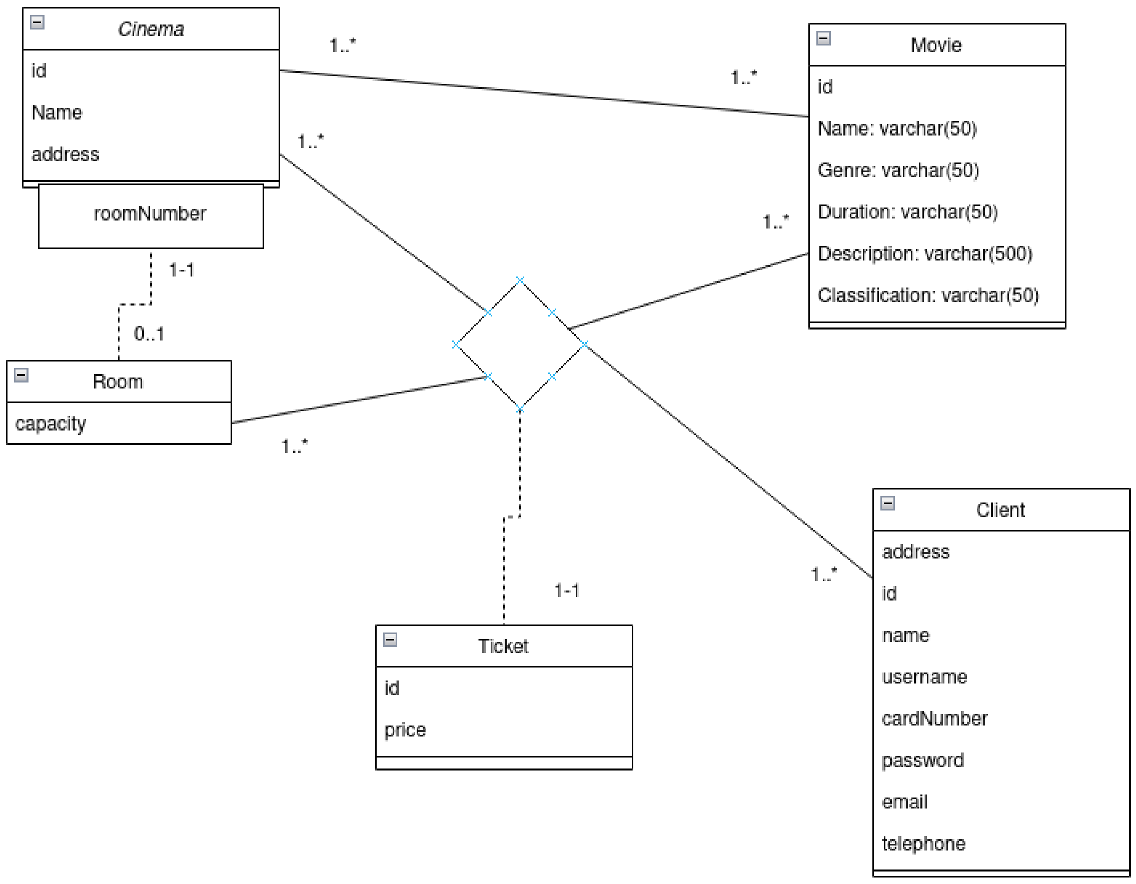
5. Adrià Rojas – 49262758R

6. Alexandre Pelegrina - 47981651V

Index

Introduction

The objective of this project is to develop a web application for data access using Django. In the following diagram you'll see the revised model diagram that has been implemented.



Github public address

The code of the project is available on the next link https://github.com/dbistuer/Projecte.git

Developer notes

Navigation:

You should be able to navigate through all pages without having to write URLs on the browser.

Django user inheritance [1]:

We've decided that the client model inherits the properties of Django default users to use the authentication methods they provide [2]. A client has got the same attributes as the Django default user and also some extra attributes.

Password reset:

When you enter the email to ask for the password reset, the e-mail won't be sent to your actual email address. Instead, on the terminal should appear a message with a link that takes you to a page where you should be able to restart the password.

List of clients and list of rooms:

The lists of clients and rooms don't have views due to the fact that it doesn't make sense to display this information to the public. This decision was taken while the view of the list of tickets was being developed, for the same reason the list of tickets also shouldn't be displayed to the public.

How to run the web application

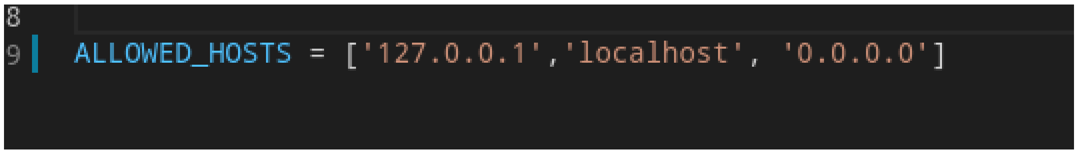
If you want to run the web application you have to choose one of these options.

1. Run the web application locally with the default server.

```
python3 manage.py runserver
```

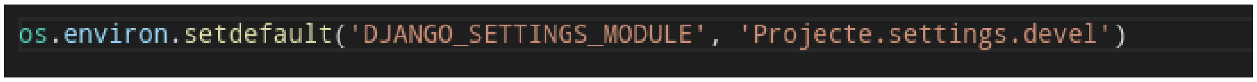
IMPORTANT: If you want to test the options 2 (gunicorn) and/or 3 (heroku local) you must do these changes.

1. **In Projecte/settings/devel.py insert the IP address 0.0.0.0 into the ALLOWED_HOST list. The list should look like this.**



```
ALLOWED_HOSTS = ['127.0.0.1','localhost', '0.0.0.0']
```

2. **In Projecte/wsgi.py replace 'Projecte.settings' for 'Projecte.settings.devel'. The option should look like this.**



```
os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'Projecte.settings.devel')
```

2. Run the web application with gunicorn.

```
gunicorn Projecte.wsgi
```

3. Run the web application locally with heroku.

```
heroku local
```

4. Run the deployed app.

https://cinema-web-project.herokuapp.com

How to deploy the web application

We have deployed on heroku as the teacher told us we have to do it.

First of all we created the requirements file with the command:

```
pip freeze > requirements.txt
```

That command searches all libraries on the environment and her version and writes it on a file specified on the command.

After we created the "Procfile" to say what commands must do on deployment and we added the lines below:

```
release: python manage.py migrate
```

```
web: gunicorn Projecte.wsgi
```

Then we modified the project structure adding one module "settings" on the main project to differentiate development of production. So we added the "settings.py" file on it and renamed the file to "base.py". Then we added "devel.py" and "prod.py" files in which we imported the base file and added the lines which contain database specification, allowed hosts and if the environment is on debug or not and deleted from the base file.

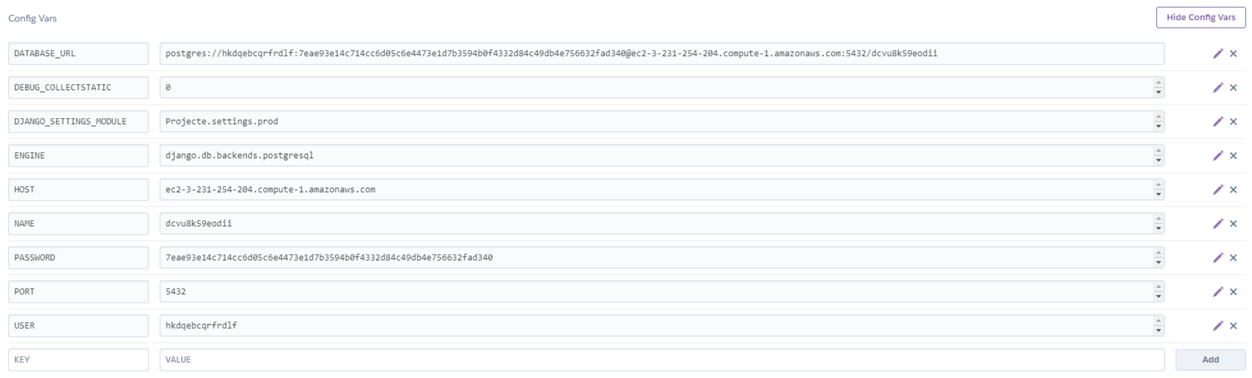
devel.py

```
from .base import *
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'Teatre.sqlite3',
    }
}
DEBUG = True
ALLOWED_HOSTS = ['127.0.0.1','localhost']
```

prod.py

```
from .base import *
import django_heroku
DATABASES = {
    'default': {
        'ENGINE': os.environ.get('ENGINE'),
        'NAME': os.environ.get('NAME'),
        'HOST': os.environ.get('HOST'),
        'PORT': os.environ.get('PORT'),
        'USER': os.environ.get('USER'),
        'PASSWORD': os.environ.get('PASSWORD')
    }
}
DEBUG = False
ALLOWED_HOSTS = ['https://cinemawebproject.herokuapp.com',
                'https://cinemawebproject.herokuapp.com/']
django_heroku.settings(locals())
```

As you can see on the prod file we have setted environment variables to make a little more secure the application. On heroku we have these variables:



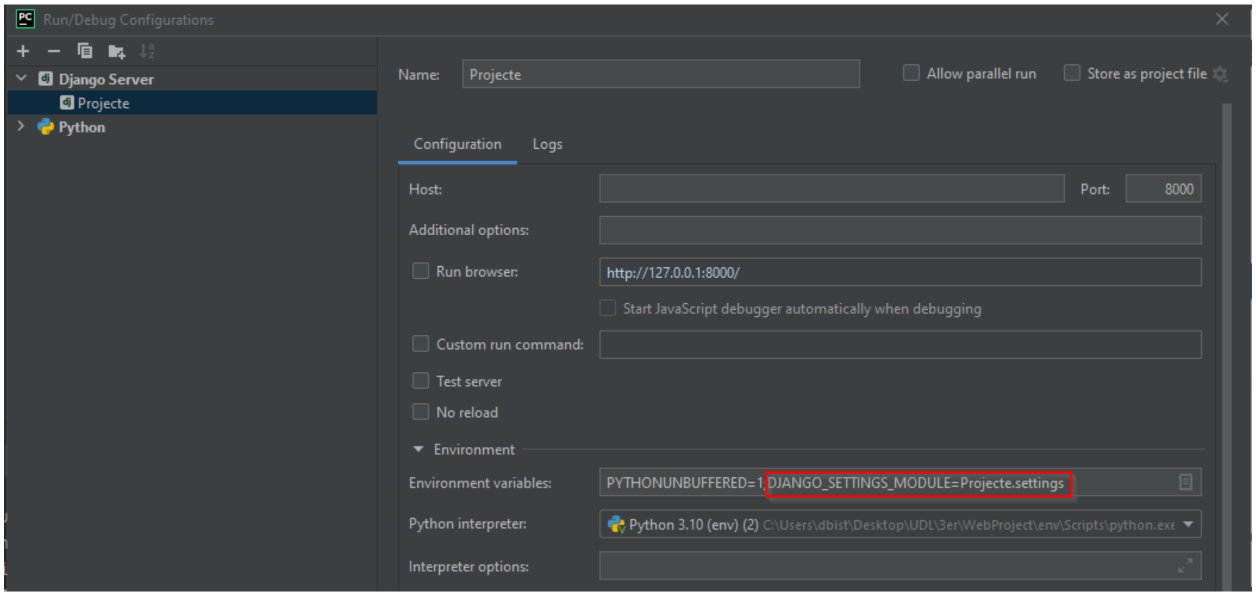
DATABASE_URL	postgres://hkdqebcqrfrdlf:7eae93e14c714cc6d05c6e4473e1d7b3594b0f4332d84c49db4e756632fad340@ec2-3-231-254-204.compute-1.amazonaws.com:5432/dcvu8k59eodii
DEBUG_COLLECTSTATIC	0
DJANGO_SETTINGS_MODULE	Projecte.settings.prod
ENGINE	django.db.backends.postgresql
HOST	ec2-3-231-254-204.compute-1.amazonaws.com
NAME	dcvu8k59eodii
PASSWORD	7eae93e14c714cc6d05c6e4473e1d7b3594b0f4332d84c49db4e756632fad340
PORT	5432
USER	hkdqebcqrfrdlf
KEY	VALUE

Too on base file we added or modified the following lines to allow static files:

```
STATIC_URL = '/static/'
```

```
STATIC_ROOT = BASE_DIR
STATICFILES_DIRS = (os.path.join(BASE_DIR, 'static'), )
STATICFILES_STORAGE                                                    =
'whitenoise.storage.CompressedManifestStaticFilesStorage'
BASE_DIR = Path(__file__).resolve().parent.parent.parent
MIDDLEWARE = [
   'django.middleware.security.SecurityMiddleware',
   'django.contrib.sessions.middleware.SessionMiddleware',
   'django.middleware.common.CommonMiddleware',
   'django.middleware.csrf.CsrfViewMiddleware',
   'django.contrib.auth.middleware.AuthenticationMiddleware',
   'django.contrib.messages.middleware.MessageMiddleware',
   'django.middleware.clickjacking.XFrameOptionsMiddleware',
   'whitenoise.middleware.WhiteNoiseMiddleware',
]
```

Also we have deleted environment variable to allow run application on develop deleting what is marked on the image:



Name: Projecte

Host:
Port: 8000
Additional options:
Run browser: http://127.0.0.1:8000/
Start JavaScript debugger automatically when debugging
Custom run command:
Test server
No reload
Environment
Environment variables: PYTHONUNBUFFERED=1;DJANGO_SETTINGS_MODULE=Projecte.settings
Python interpreter: Python 3.10 (env) (2) C:\Users\dbist\Desktop\UDL\3er\WebProject\env\Scripts\python.exe
Interpreter options:

Then we modified “manage.py” to set develop settings if there no is that variable replacing this line:

```
os.environ.get('DJANGO_SETTINGS_MODULE')
```

For this lines:

```
if os.environ.get('DJANGO_SETTINGS_MODULE') is None:
                              os.environ.setdefault('DJANGO_SETTINGS_MODULE',
'Projecte.settings.devel')
```

Before we have installed heroku, her client, gunicorn and whitenoise:

```
pip install heroku

pip install heroku-client

pip install gunicorn

pip install whitenoise
```

Then we have logged on heroku:

```
heroku login
```

After login successfully we create a heroku app.

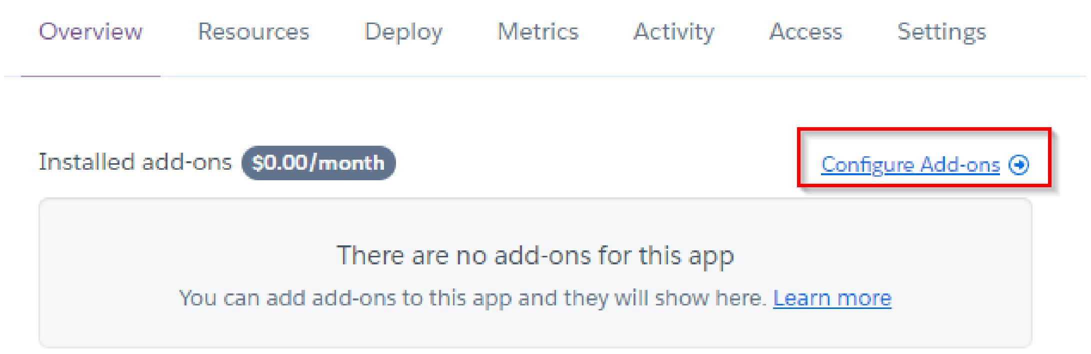
```
heroku create cinema-web-project

heroku git:remote -a cinema-web-project
```

If you want to update the web application you only have to push the last commit you did:

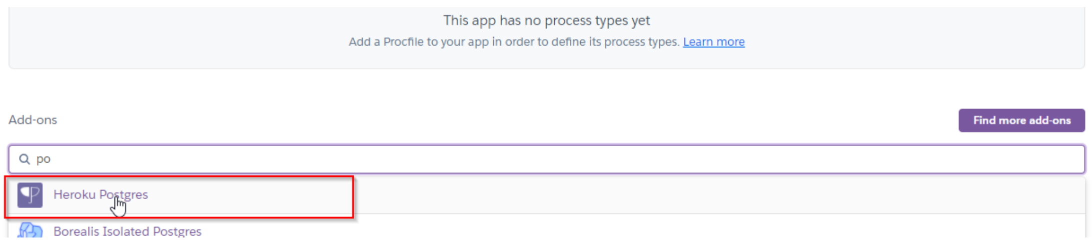
```
git push heroku master
```

We have added Postgres add-on



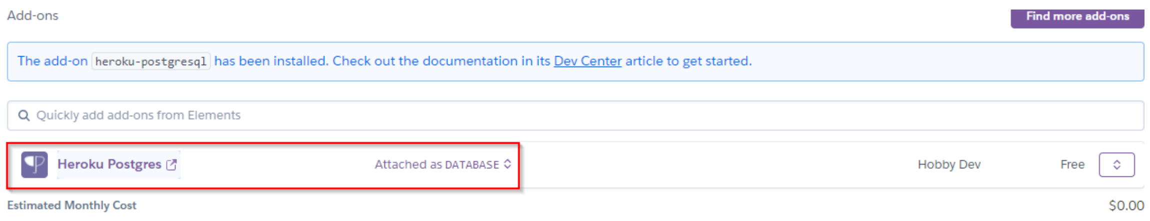
There are no add-ons for this app

You can add add-ons to this app and they will show here. Learn more


This app has no process types yet

Add a Procfile to your app in order to define its process types. Learn more

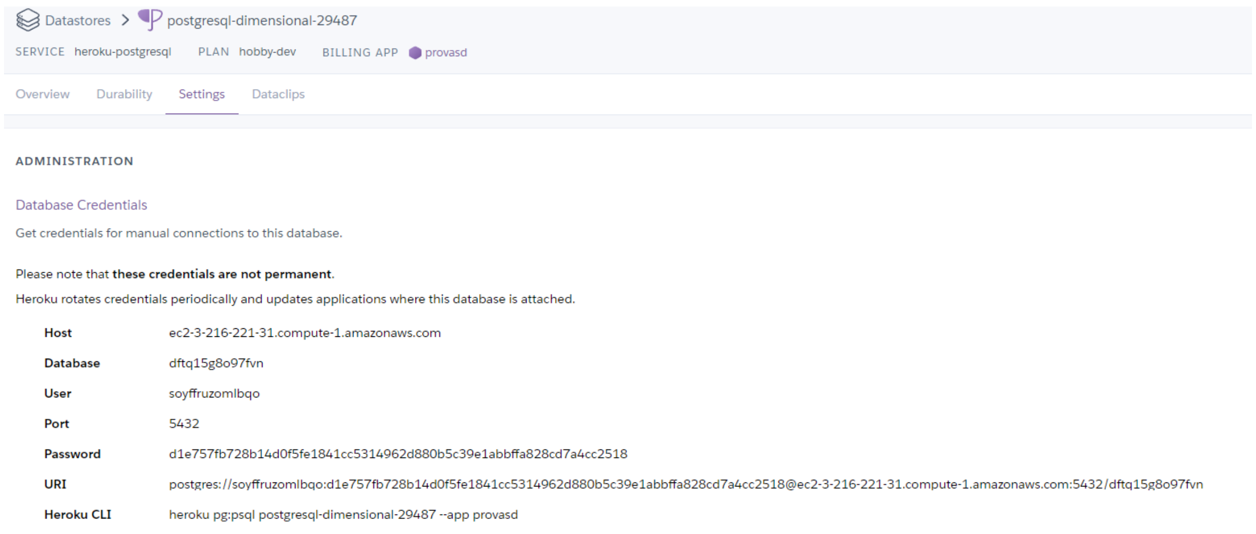
Once we have added postgres we can add the environment variables on heroku entering on it



The add-on `heroku-postgresql` has been installed. Check out the documentation in its Dev Center article to get started.

Heroku Postgres	Attached as DATABASE	Hobby Dev	Free
Estimated Monthly Cost			$0.00

Before we enter it we must go to settings and view credentials. Tere we have the values to set environment variables as you can see on the image:



Datastores > postgresql-dimensional-29487
SERVICE heroku-postgresql PLAN hobby-dev BILLING APP provasd
Overview
Durability
Settings
Dataclips
ADMINISTRATION
Database Credentials

Get credentials for manual connections to this database.

Please note that **these credentials are not permanent**.

Heroku rotates credentials periodically and updates applications where this database is attached.

Host	ec2-3-216-221-31.compute-1.amazonaws.com
Database	dftq15g8o97fvn
User	soyffruzomlbqo
Port	5432
Password	d1e757fb728b14d0f5fe1841cc5314962d880b5c39e1abbffa828cd7a4cc2518
URI	postgres://soyffruzomlbqo:d1e757fb728b14d0f5fe1841cc5314962d880b5c39e1abbffa828cd7a4cc2518@ec2-3-216-221-31.compute-1.amazonaws.com:5432/dftq15g8o97fvn
Heroku CLI	heroku pg:psql postgresql-dimensional-29487 --app provasd

Then we entered the following commands to add static files and deploy style and js files

```
python manage.py collectstatic
```

The following command to add heroku environment variable to set production settings file

```
heroku config:set DJANGO_SETTINGS_MODULE=Projecte.settings.prod
```

Finally run:

```
git add .

git commit -m “(message)”

git push heroku master
```

Docker creation:

To create the docker we have created the following files: “docker-compose.yml” and “Dockerfile”

“docker-compose.yml” contains the necessary data to say to docker version, database which use to add it and which web application contains, how to run it, name of that app ports that have to use and her dependences:

```
version: "3"
services:
 db:
   container_name: "django-postgre"
   image: postgres
   ports:
     - "5432:5432"
   environment:
     POSTGRES_PASSWORD: "secret"
     POSTGRES_USER: "django"
     POSTGRES_DB: "no"
   restart: "no"
 web:
   container_name: "django"
   build: .
```

```
    command: python manage.py runserver 0.0.0.0:8000
    volumes:
      - .:/app
    ports:
      - "8000:8000"
    depends_on:
      - db
```

“Dockerfile” sets the language, create a folder, sets working directory, copies the requirements file run’s installation of all libraries on each file and copy all files to directory

```
FROM python
RUN mkdir /app
WORKDIR /app
ADD requirements.txt /app/
RUN pip install -r requirements.txt
ADD . /app/
```

Finally once have installed Docker on computer have run:

```
docker-compose up
```

This command create the docker using the files specified

References

[1] V. Freitas. How to extend the Django user model. simpleisbetterthancomplex.
https://simpleisbetterthancomplex.com/tutorial/2016/07/22/how-to-extend-django-user-model.html#onetoone

[2] MDN contributors. Django authentication. developer.mozilla
https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Authentication

[3] Pretty Printed. Deploy a Django App to Heroku.
https://www.youtube.com/watch?v=GMbVzl_aLxM

[4] A. Ortega Mármol. Web project teacher.

[5] Heroku documentation page. https://devcenter.heroku.com/categories/reference

[6] Django documentation page. https://docs.djangoproject.com/en/4.0/