

6. Основи Django.

Контрольні запитання

- 06.01. Що таке шаблон проєктування MVC?
- 06.02. Який шаблон проєктування використовується в Django?
- 06.03. Як в Django відбувається URL-маршрутизація?
- 06.04. Як відбувається побудова моделей в Django?
- 06.05. Що таке менеджер моделей?
- 06.06. Для чого використовуються функції-представлення в Django?
- 06.07. Яким чином застосовуються файли шаблонів в Django?
- 06.08. Наведіть спосіб як використовувати змінні в шаблоні та основні керуючі конструкції шаблонізатора Django.

Завдання для аудиторної роботи

- [A]06.01. Створити Django-проєкт для бібліотеки. Проєкт має бути розміщений у Git-репозиторії. Кожен пункт повинен бути окремим комітом в Git-репозиторії з прикріпленням тегом, який можна додати командою

```
git tag <tagname> <commit-hash>
```

Для того щоб вивантажити на віддалений репозиторій коміти разом з тегами, потрібно виконати команду

```
git push origin <main-branch> --tags
```

1. Запустити проєкт Django.

- Встановити Django за допомогою команди

```
pip install django
```

- Створити теку **libraryproject** та всередині неї теку **app**

```
mkdir app
```

- Створити Django-проєкт **library** всередині директорії **app**

```
django-admin startproject library app
```

- Запустити проєкт, знаходячись в теці **app**

```
python manage.py runserver 8000
```

- Відвідати **http://localhost:8000/** щоб побачити сторінку зі злітанням ракети.

2. Створити застосунок **catalog** та головну сторінку для нього.

- Виконати команду для створення застосунку

```
python manage.py startapp catalog
```

- Описати каркас сторінок сайту **catalog/templates/catalog/base.html**.
- Описати власну головну сторінку **index.html**, яка розширює **base.html**.
- Описати представлення для переходу на головну сторінку.
- Описати маршрут до головної сторінки.
- Додати перенаправлення з адреси «/» на адресу «/catalog/».
- Зареєструвати новий застосунок у файлі **library/settings.py**

```
INSTALLED_APPS = [  
    ...  
    "catalog.apps.CatalogConfig",  
]
```

- Відвідати **http://localhost:8000/** щоб побачити нову головну сторінку.

3. Контейнеризувати проєкт.

- Створити **Dockerfile** для побудови проєкту Django на базі серверу **uvicorn**.

```
uvicorn library.asgi:application --host 0.0.0.0 --port 8000 --reload
```

- Описати **docker-compose.yml** для створення контейнерів проєкту Django, бази даних PostgreSQL та застосунку pgAdmin.
- Зареєструвати базу даних у файлі **library/settings.py** проєкту

```
DATABASES = {  
    "default": {  
        "ENGINE": "django.db.backends.postgresql",  
        "NAME": os.getenv("DB_NAME"),  
        "USER": os.getenv("DB_USER"),  
        "PASSWORD": os.getenv("DB_PASS"),  
        "HOST": "db",  
        "PORT": "5432",  
    }  
}
```

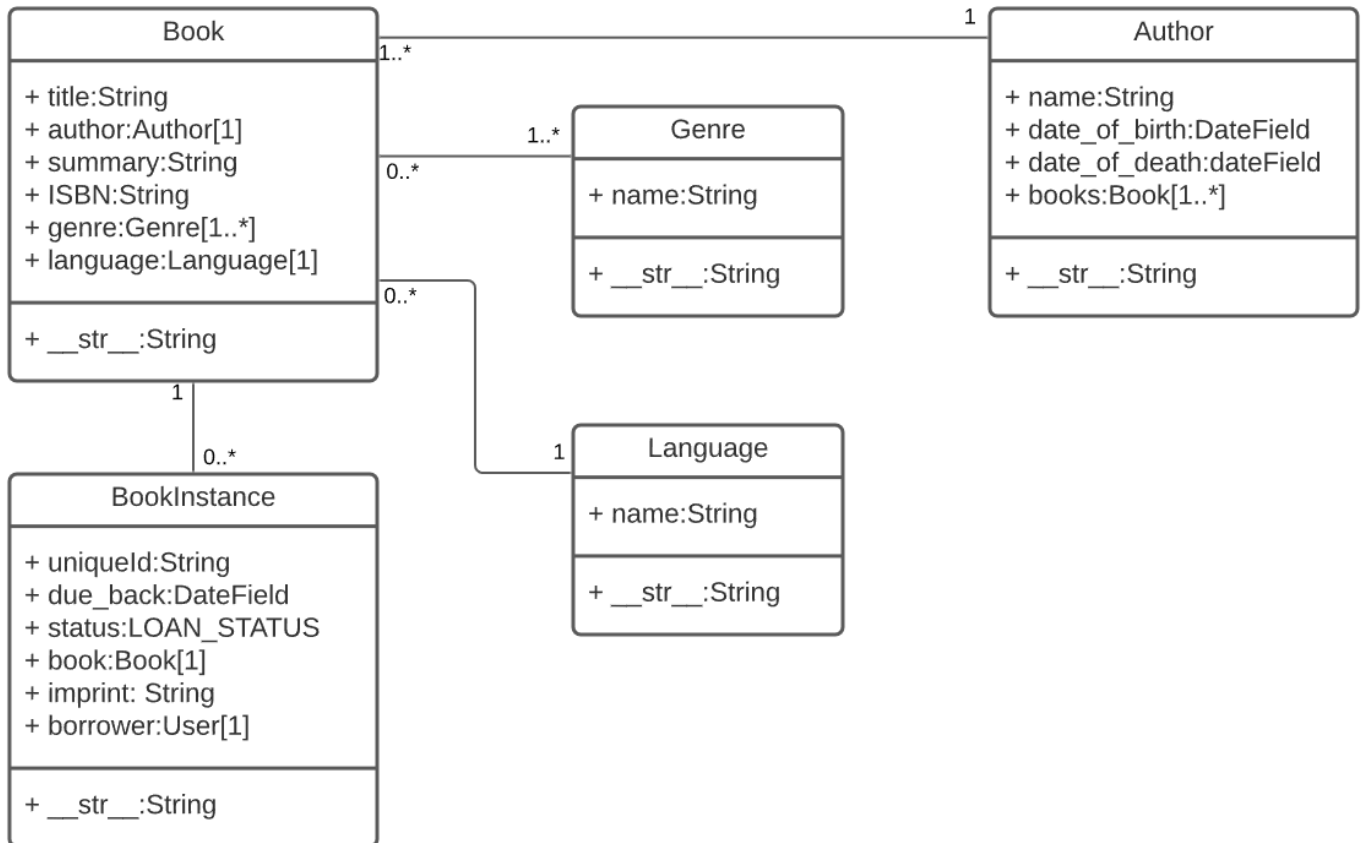
- Додати шлях для точки входу асинхронного веб-сервера

```
ASGI_APPLICATION = "library.asgi.application"
```

- Запустити контейнери і перевірити роботу веб-застосунку.

4. Описати моделі.

- Створити моделі згідно діаграми



- Додати до збірки проєкту міграцію бази даних за допомогою команд

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

- Запустити контейнери.
- Перевірити в застосунку pgAdmin чи створилися таблиці.

5. Додати сервер **nginx** як реверс-проксі та для опрацювання статичних файлів.

- Додати до **library/settings.py** пошук статичних файлів

```
STATIC_URL = "/static/"
STATIC_ROOT = BASE_DIR / "staticfiles"

MEDIA_URL = "/media/"
MEDIA_ROOT = BASE_DIR / "media"
```

- Створити файл **nginx/default.conf**

```
server {
    listen 80;

    location /static/ {
        alias /static/;
    }

    location /media/ {
        alias /media/;
    }

    location / {
```

```

    proxy_pass http://web:8000;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
  }
}

```

- Додати в **docker-compose.yml** контейнер для серверу **nginx**

```

nginx:
  image: nginx
  container_name: nginx
  ports:
    - "80:80"
  volumes:
    - ./nginx/default.conf:/etc/nginx/conf.d/default.conf
    - static_volume:/static
    - media_volume:/media
  depends_on:
    - web

```

- Застосувати команду збору статичних файлів у **docker-compose.yml**

```
python manage.py collectstatic --noinput
```

- Додати до сервісу **web** сховище для синхронізації статичних файлів

```

volumes:
  - static_volume:/app/staticfiles

```

- Створити власний статичний файл **style.css** для застосунку **catalog**.
- Додати на головну сторінку імпорт створених статичних файлів. Відвідати головну сторінку та переконатися, що статичні файли завантажуються.

6. Додати адміністративну панель.

- Створити суперкористувача

```
python manage.py createsuperuser
```

- Зареєструвати моделі в адміністративній панелі за допомогою функції

```
admin.site.register()
```

- Створити в адміністративній панелі декілька екземплярів кожного класу.
- Додати на головній сторінці відображення кількості авторів, книг, екземплярів книг та доступних екземплярів книг.

7. Додати сторінку для повного переліку книг.

8. Додати сторінку для детального опису книги разом з її екземплярами.

9. Додати сторінку для переліку авторів.

10. Додати сторінку для детального опису автора та переліку його книг.

Завдання для самостійної роботи

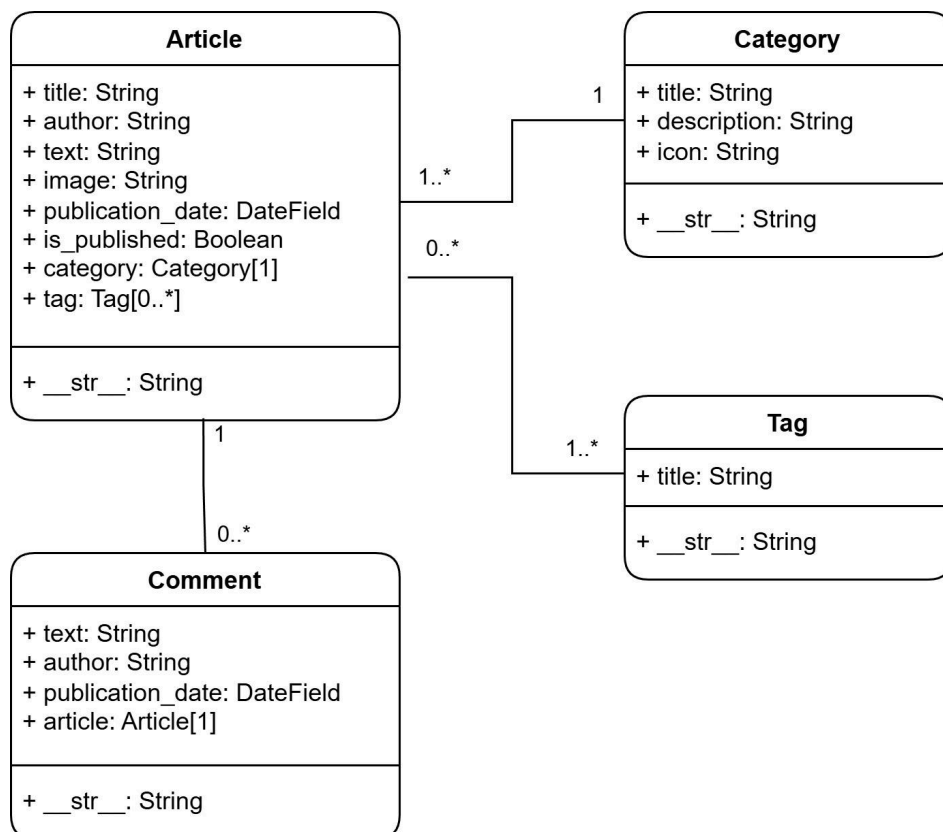
[В]06.01. Створити Django-проект для блогу. Проект має бути розміщений у Git-репозиторії. Кожен пункт повинен бути окремим комітом в Git-репозиторії з прикріпленим тегом, який можна додати командою

```
git tag <tagname> <commit-hash>
```

Для того щоб вивантажити на віддалений репозиторій коміти разом з тегами, потрібно виконати команду

```
git push origin <main-branch> --tags
```

1. Запустити проект Django.
2. Створити застосунок **blog** проекту та головну сторінку для нього.
3. Контейнеризувати проект. Під'єднати базу даних MySQL.
4. Описати моделі згідно діаграми



Поле **icon** означає код іконки у **fontawesome** для відображення.

Поле **image** – це покликання на картинку для відображення у статті.

Автора реалізувати як звичайне рядкове поле.

5. Додати сервер **nginx** як реверс-проксі та для опрацювання статичних файлів.
6. Додати адміністративну панель.
7. Додати на головну сторінку список із трьох останніх опублікованих статей.
8. Додати сторінку для списку всіх опублікованих статей.
9. Додати сторінку для детального опису статті та переліку коментарів до неї.
10. Додати сторінку для списку категорій.