МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ

Інститут комп'ютерних систем

Кафедра інформаційних систем

Лабораторно-практична робота №9

З дисципліни: «Технології створення програмних продуктів»

Тема: «Проектування тестових наборів»



1. КОНСТРУЮВАННЯ ПРОГРАМНОГО ПРОДУКТУ
   1. Особливості конструювання структур даних
      1. Особливості інсталяції та роботи з СУБД

Для розробки проекту була використана реляційна БД – PostgreSQL 13 версії, яка була встановлена на локальні комп’ютери. Для деплою була використана хмарна СУБД Heroku PostgreSQL, з якою робота відбувається лише через графічний інтерфейс у браузері.

* + 1. Особливості створення структур даних

Після налаштування підключення до БД, запити створення таблиць формуються автоматично за допомогою Django на основі написаних класів та за необхідності таблиці оновлюються при зміні програмного коду – також автоматично.

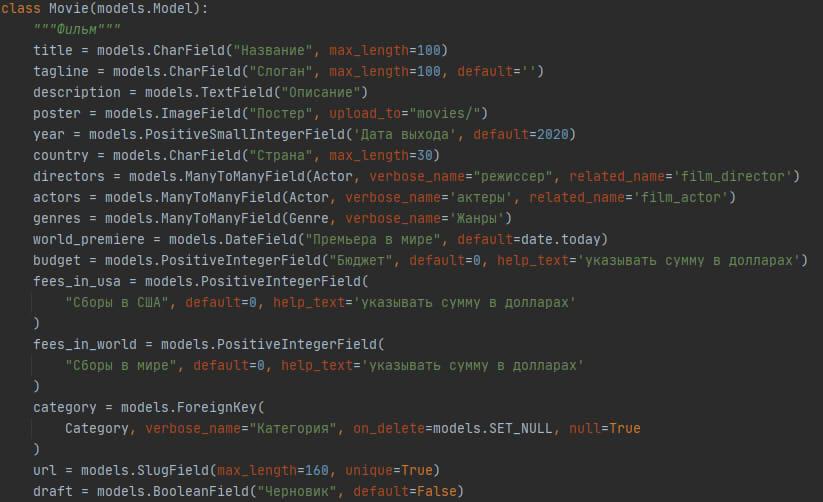


Рис. 1 – Клас “Movie”

Створення таблиці на основі класу Movie (рис. 1) буде відбуватися за допомогою спеціальних атрибутів, які сформують поля БД автоматично на основі описаних типів.

Наприклад:

* Атрибут models.CharField сформує звичайне поле типу Char з указаним розміром
* Атрибут models.TextField формує текстове поле
* Атрибут models.ImageField формує поле типу Char з прописаною адресою до теки з зображенням (в нашому випадку: movies/)
* Атрибут models.PositiveSmallInteger сформує поле типу Int з обмеженням на лише невід’ємність значення цього поля. Діапазон значень - з 0 до 32767
* Атрибут models.ManyToManyField формує зв’язок багато-до-багатьох з сутністю, указаною в цьому атрибуті, наприклад actors = models.ManyToManyField(Actor) сформує зв’язок з сутністю актора
* Атрибут models.DateField сформує поле типу дата
* Атрибут models.PositiveIntegerField сформує поле типу Int з обмеженням на лише невід’ємність значення цього поля
* Атрибут models.ForeignKey(<сутність>) сформує поле типу int з зовнішнім ключом - зв’язком з сутністю, вказаною у параметрах
* Атрибут models.Boolean сформує поле логічного типу даних, яке може приймати лише True або False
* Атрибут models.Slug - формує поле текстового типу в якому можуть бути лише букви цифри і особливі символи, які сформують адресу екземпляру класу у браузері.

Під’єднання до бд відбувається у файлі налаштування проекту settings.py у виді

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.postgresql',

'NAME': 'name',

'USER': 'django',

'PASSWORD': 'pass',

'HOST': 'localhost',

'PORT': '5432',

}

}

Після описання програмних класів створюються так звані міграції командою python manage.py makemigrations, що сформує запити створення усіх описаних програмних класів, після внесення усіх бажаних змін, якщо вони є, виконується команда python manage.py migrate, що виконає усі запити "міграцій".

* 1. Особливості конструювання програмних модулів
     1. Особливості роботи з інтегрованим середовищем розробки

Використовувалося середовище програмування PyCharm; інсталяція проводилася з офіційного сайту (jetbrains.com), ліцензія – студентська.

Включає зручні інструменти розробки. Використовувалися фреймворки Django, Bootstrap, psycopg2 – для значного пришвидшення написання програми.

* + 1. Особливості створення програмної структури з урахуванням спеціалізованого Фреймворку

Django створює головні конфігураційні файли, та дозволяє створювати так звані «застосунки» за допомогою команди python manage.py startapp <appname>, що створить у проекті нову папку з моделями та налаштуваннями для <appname>, потім цей застосунок допишеться до встановлених у головному файлі налаштувань проекту settings.py

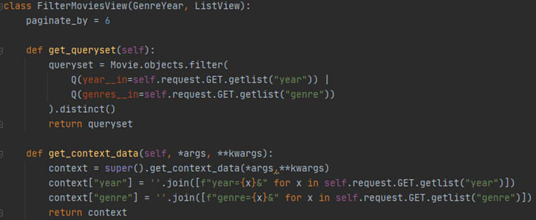
* + 1. Особливості створення програмних класів

Python у комбінації з Django дозволяє робити моделі зручними за допомогою наслідування класами вбудованої моделі. Розробнику не потрібно писати спеціальні поля типу id. Атрибути класів задаються без обмежень на публічність, та за допомогою models.<тип-атрибуту>.

Django має у своїй бібліотеці багато вбудованих та зручних типів даних, тож додавання типу ImageField для атрибуту «постер» через крапку створить автоматично зручне для взаємодії поле для постера у БД.

* + 1. Особливості розробки алгоритмів методів програмних класів або процедур/функцій

Алгоритми методів програмних класів можуть задаватися як методи, безпосередньо у класі, так і обробка екземплярів класів може відбуватися при побудові view повертаємого користувачу.



Розглянемо реалізацію метода get\_movie\_by\_genre(movie\_genre). Цей метод був об’єднаним з методом get\_movie\_by\_year(movie\_year) у більш загальний метод FilterMoviesView, що має можливість сортувати фільми водночас як за роком випуску, так і за жанрами для надання більш повної можливості вибору.

Завдяки елементу Q із queryset, метод має доступ до логічної операції «або», що дозволяє, наприклад, шукати водночас фільми різноманітних несумісних жанрів, або різних років виходу.

Також цей метод займається тим, що перебудовує url-адресу таким чином, щоб вона явно демонструвала те, що саме шукає користувач у даний момент часу.

У методу також наявний елемент пагінації – кожні 6 фільмів буде створена наступна сторінка пошуку для того, щоб не забруднювати екран і не робити потенційно нескінчених списків.

* + 1. Особливості використання спеціалізованих бібліотек та АРІ

Для спрощення розробки програмного продукту було використано готове API взаємодії з БД - psycopg2. API повністю автоматизований, тож для роботи з БД потрібно було лише його встановити.