**Создание нового приложения. Работа с базой данных. Создание модели.**

В данной лабораторной работе мы создадим ещё одно новое Django-приложение, чтобы показать, что можно легко взаимодействовать между ними.

Для создания нового приложения *news* будем использовать изученные ранее команды входа в папку и вызова функции создания приложения с помощью файла *manage.py*.

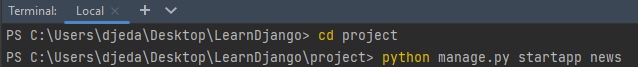


Рисунок 1 - Создание нового приложения news

После этой команды у нас создаётся папка, где та же самая структура, что и у приложения *main*.

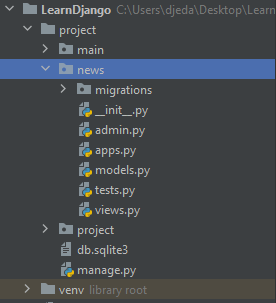


Рисунок 2 - Директория приложения news

Теперь его нам нужно зарегистрировать для самого проекта. Для этого заходим в файл *settings.py* и указываем название Django-приложения.

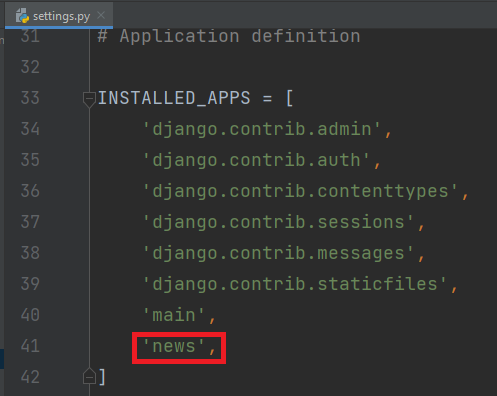


Рисунок 3 - Добавление приложения news в settings.py

После регистрации нам необходимо связать и URL-адреса между проектом и приложением. Для этого переходим в папку с проектом *project* и указываем имя и URL-файл, в котором будут находиться URL-адреса приложения *news*.

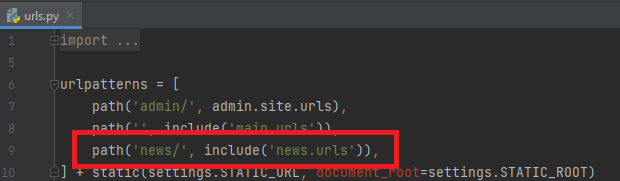


Рисунок 4 - Добавление URL-адреса приложения news в .../project/urls.py

Создаём этот URL-файл внутри папки приложения.

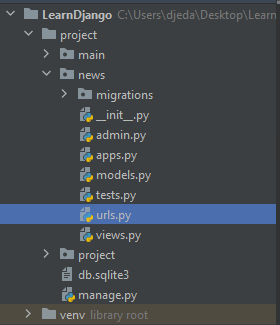


Рисунок 5 - Создание файла urls.py в директории news

Внутри созданного файла укажем пустой URL-адрес, название представления приложения и имя самого пути.

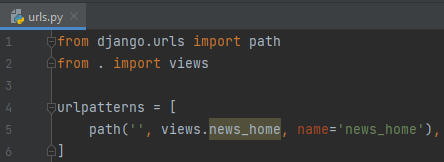


Рисунок 6 - Файл urls.py

Представление URL-адреса в файле *views.py* сделаем позже. Пока создадим новую папку для шаблонов *templates*, в которой будет находиться папка с тем же названием, что и у Django-приложения,и внутри этой папки сам HTML-файл *news\_home*. Так как название *templates* является ключевым и общим для всего Django-проекта, то не важно, сколько таких папок и где именно мы должны создать.

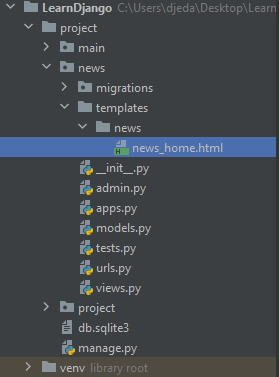


Рисунок 7 - Создание HTML-страницы news\_home

Заполним шаблон некоторой информацией о странице *news\_home*.

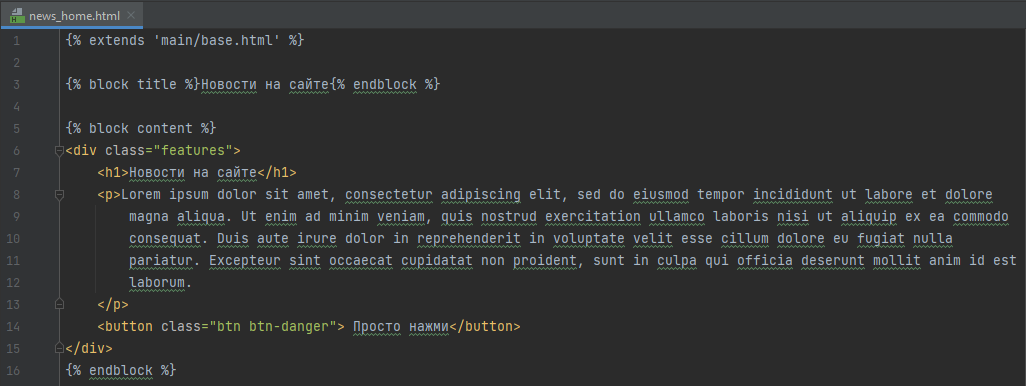


Рисунок 8 - Файл news\_home.html

Теперь перейдём к представлению шаблона в файл *views.py* Django-приложения *news*.

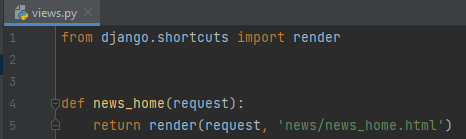


Рисунок 9 - Файл views.py

После этого можем проверить результат на запущенном локальном сервере. В адресную строку браузера вписываем название Django-приложения. Потом происходит переход на веб-страницу “Новость на сайте”.

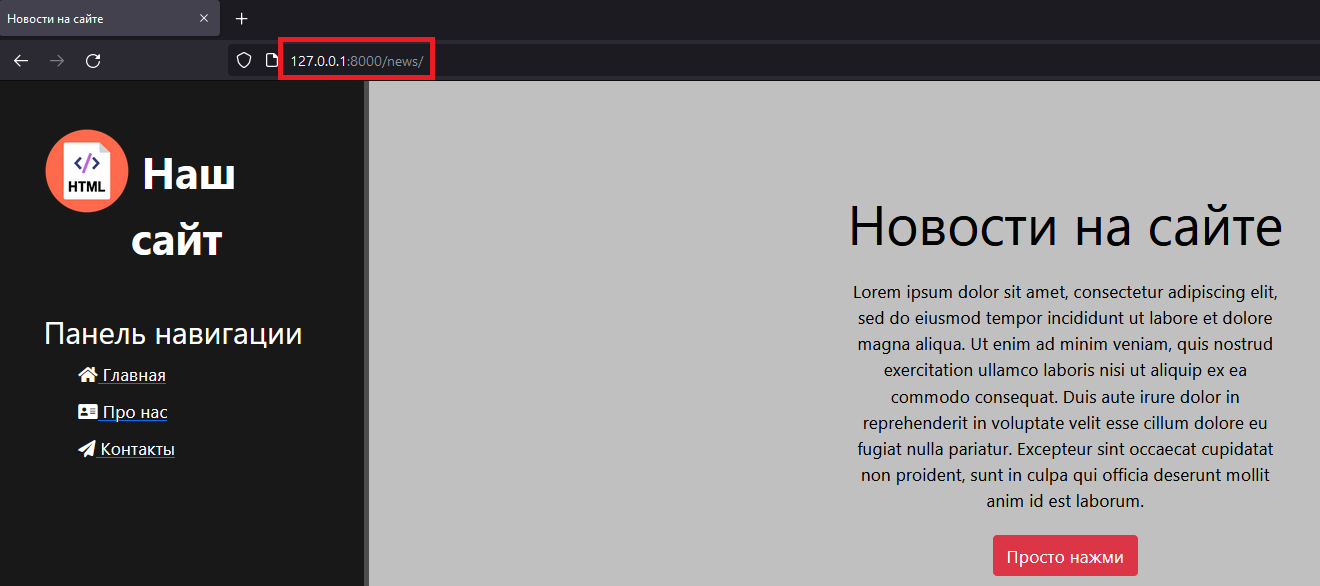


Рисунок 10 - Проверка новостной страницы

Теперь можно добавить в ранее созданную боковую панель навигации URL-адрес представления приложения *news* для создания гиперссылки.

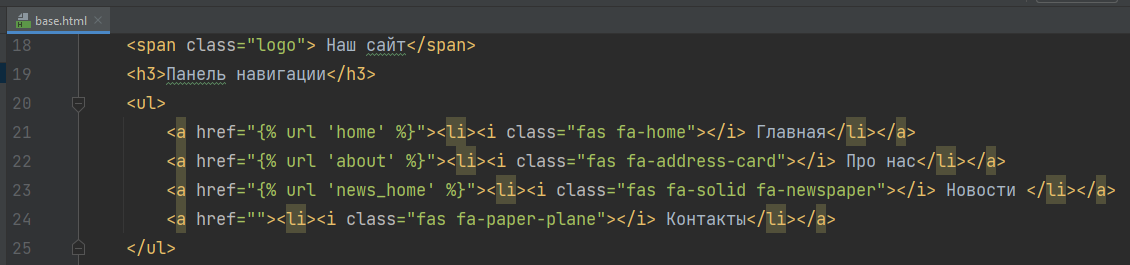


Рисунок 11 - Добавление гиперссылки “Новости” в base.html

Проверим результат работы обновив веб-страницу на локальном сервере.

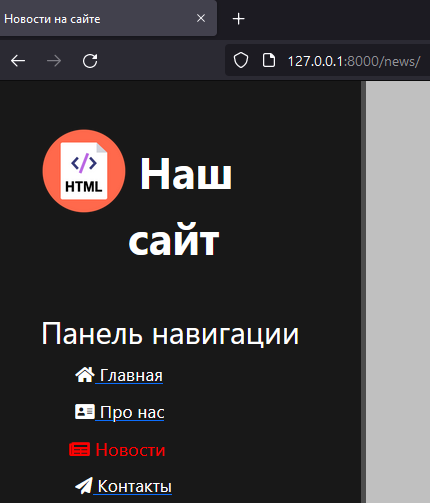


Рисунок 12 - Проверка гиперссылки “Новости”

Перейдём к созданию базы данных, а именно её моделей. Для этого внутри директории *news* создадим класс *Articles*, который будет содержать поля:

* *title* (служит для заголовка новости);
* *anons* (некоторая информация для анонса новости);
* *full\_text* (для самого текста статьи);
* *date* (дата публикации).

Типы полей мы задаём с учётом их задач. *CharField* имеет ограниченное количество символов, которое нам необходимо обязательно указывать. Тип *TextField* имеет практически неограниченный символьный размер, поэтому его мы и укажем в тексте для статьи. *DateTimeField* необходимо для указания времени и даты создания таблицы данных. Если не указывать в нём эти данные, то произойдёт автоматическая запись текущей дата и времени.

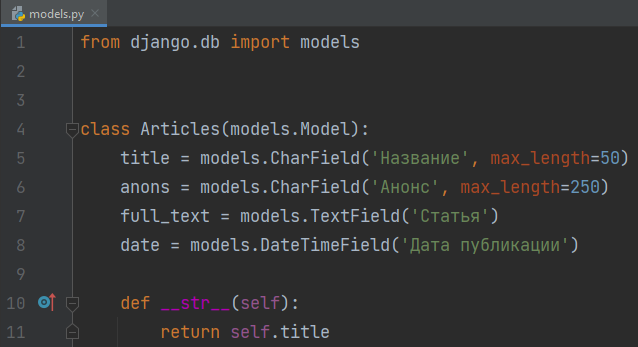


Рисунок 13 - Файл models.py

Для того, чтобы сформировать саму базу данных из созданной модели, пропишем команду в терминале.

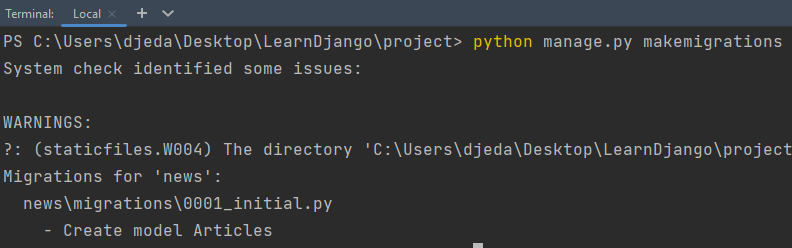


Рисунок 14 - Создание файла миграций

После ввода команды появляется автоматически созданный файл в директории *migrations* приложения *news*.

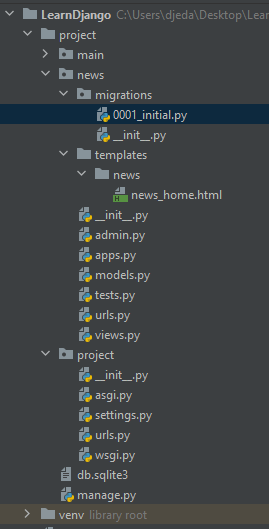


Рисунок 15 - Файл миграций в директории migrations

Если открыть данный файл, то в нем можно увидеть модель *Articles* вместе с созданными нами ранее полями. К данным четырем полям еще добавилось пятое поле *id*, которое является идентификатором самой модели.

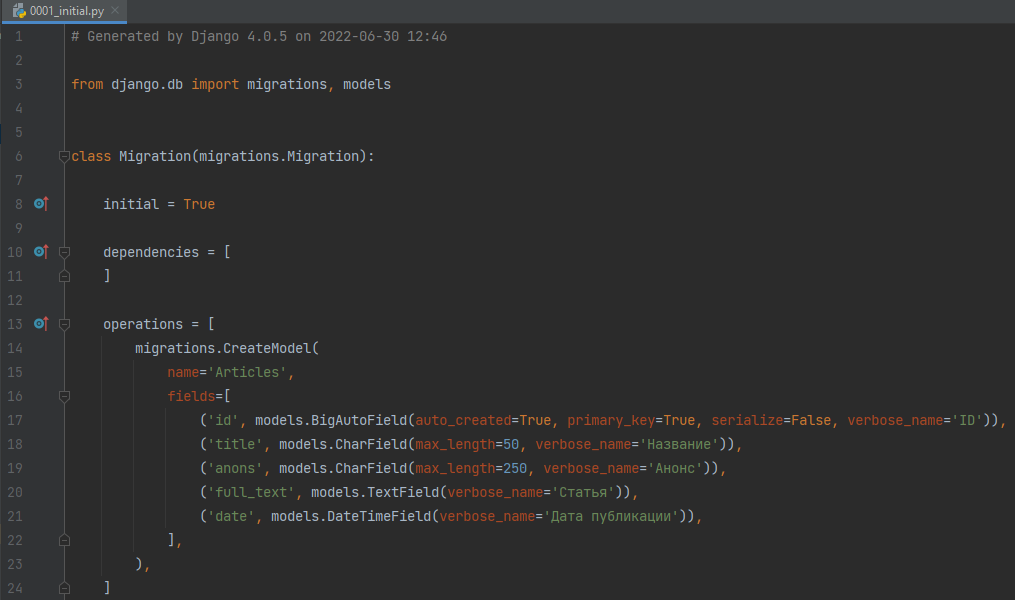


Рисунок 16 - Файл миграций

После этого нам необходимо дополнительно подтвердить миграцию, тем самым отправив данные на сервер SQLite, с помощью соответствующей команды.

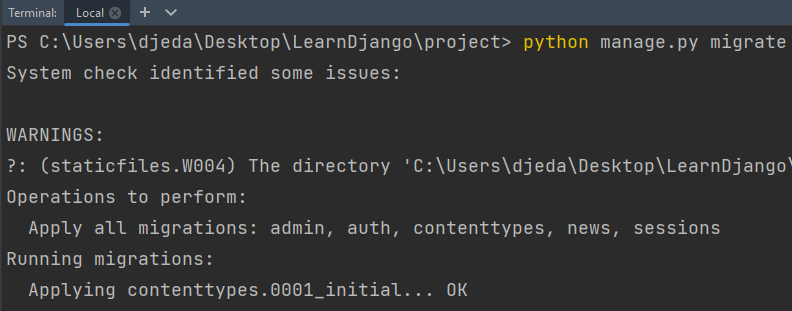


Рисунок 17 - Проведение миграции

После этого уже автоматически заполняется внутри корневой папки проекта сама база данных SQLite, в которых содержится созданная модель.

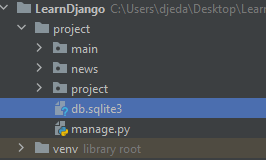


Рисунок 18 - Формирование базы данных на SQLite

После этого мы можем зайти в панель администратора, прописав в адресную строку браузера URL-адрес админ-панели.

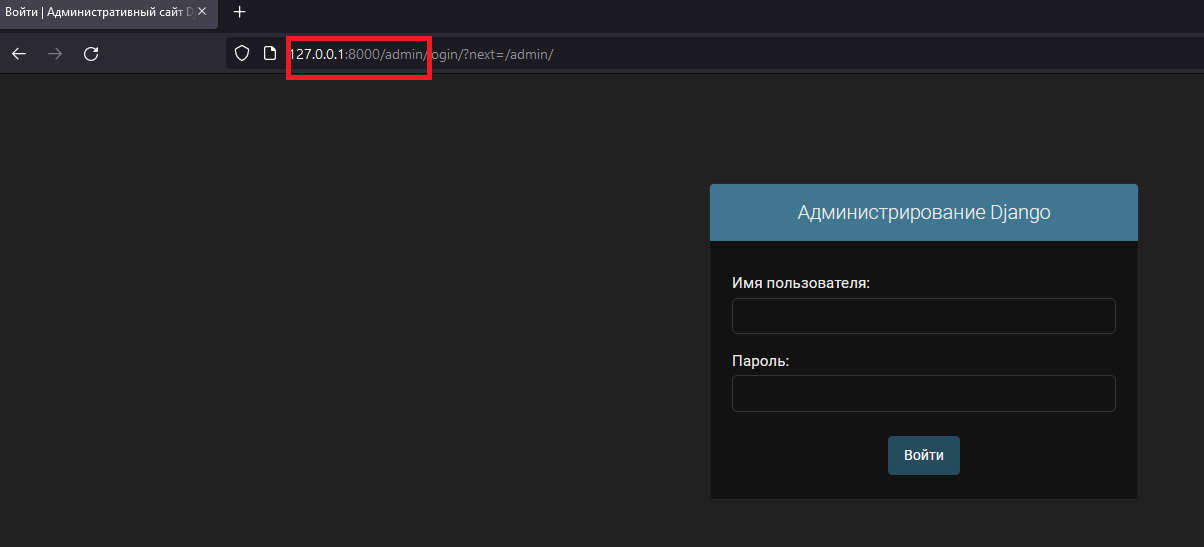


Рисунок 19 - Страница входа для администратора сайта

Для того, чтобы войти в панель администратора, необходимо создать его учетную запись. Для этого вводим в терминал команду. После необходимо ввести имя пользователя и пароль, адрес электронной почты можно не указывать, а затем подтвердить создание учетной записи.

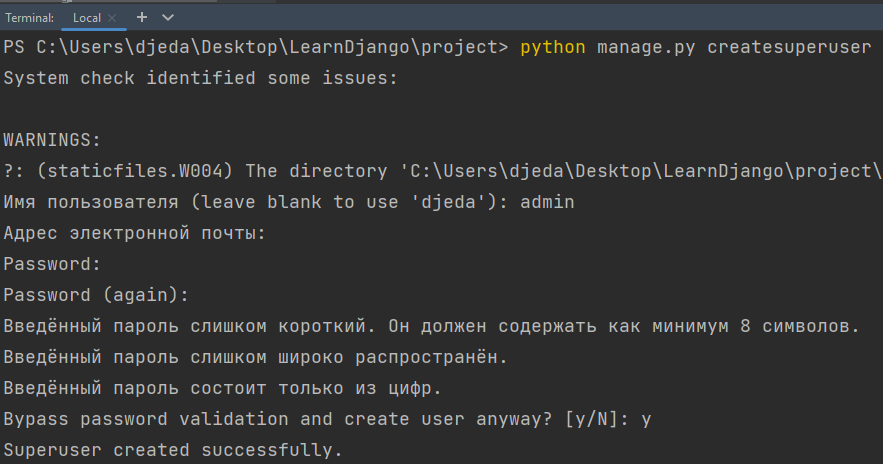


Рисунок 20 - Создание учетной записи администратора

Теперь можно ввести учетные данные на странице входа в панель администратора.

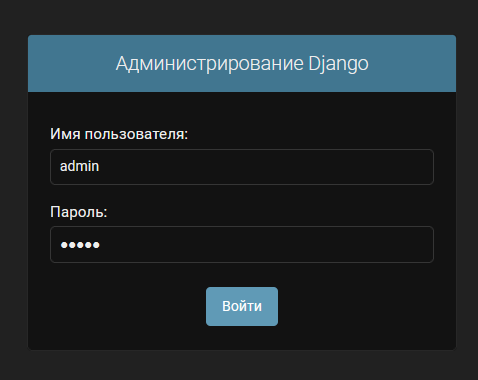


Рисунок 21 - Ввод имени пользователя и пароля администратора

После успешного входа в учетную запись отобразится страница администратора, на которой можно увидеть категории “Группы” и “Пользователи”.

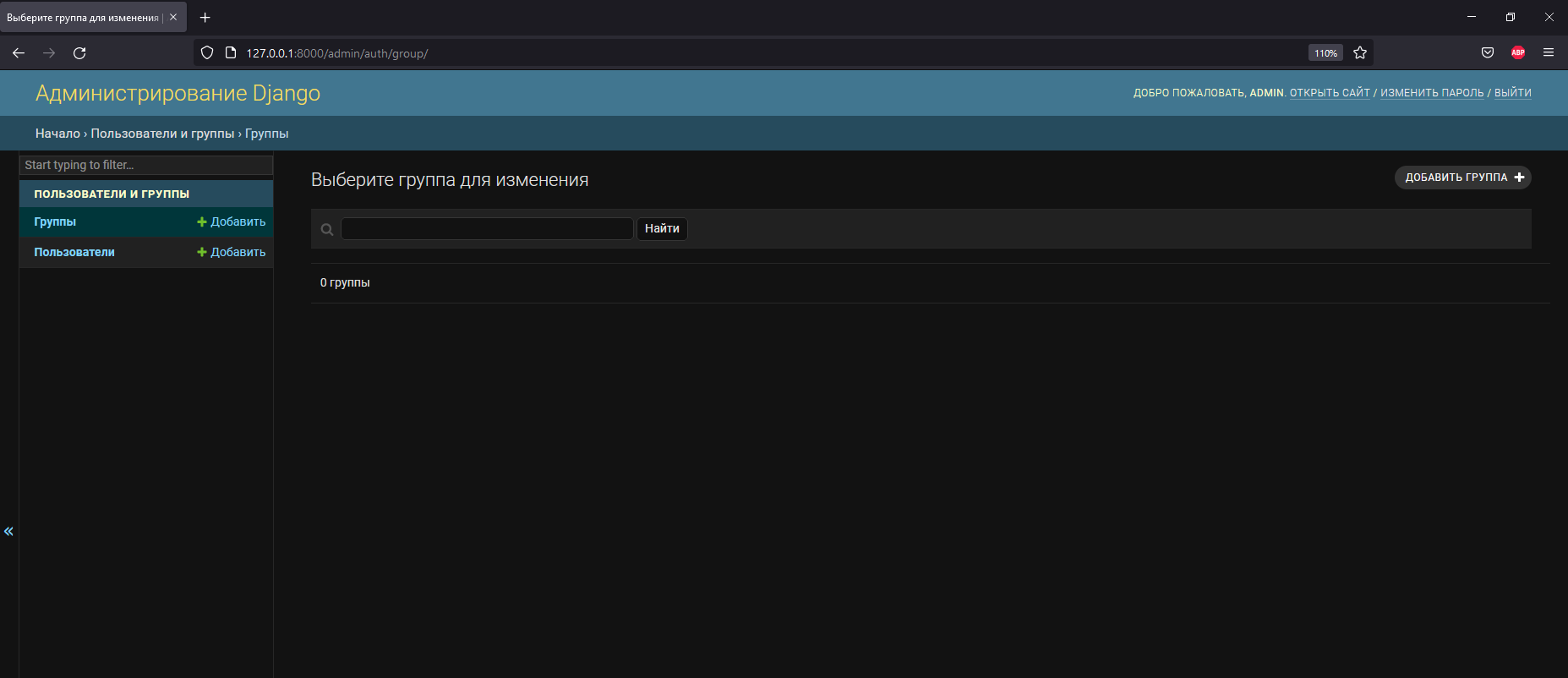


Рисунок 22 - Страница администратора

Мы можем перейти на страницу “Пользователи”, на ней будет отображен единственный пользователь - администратор сайта.

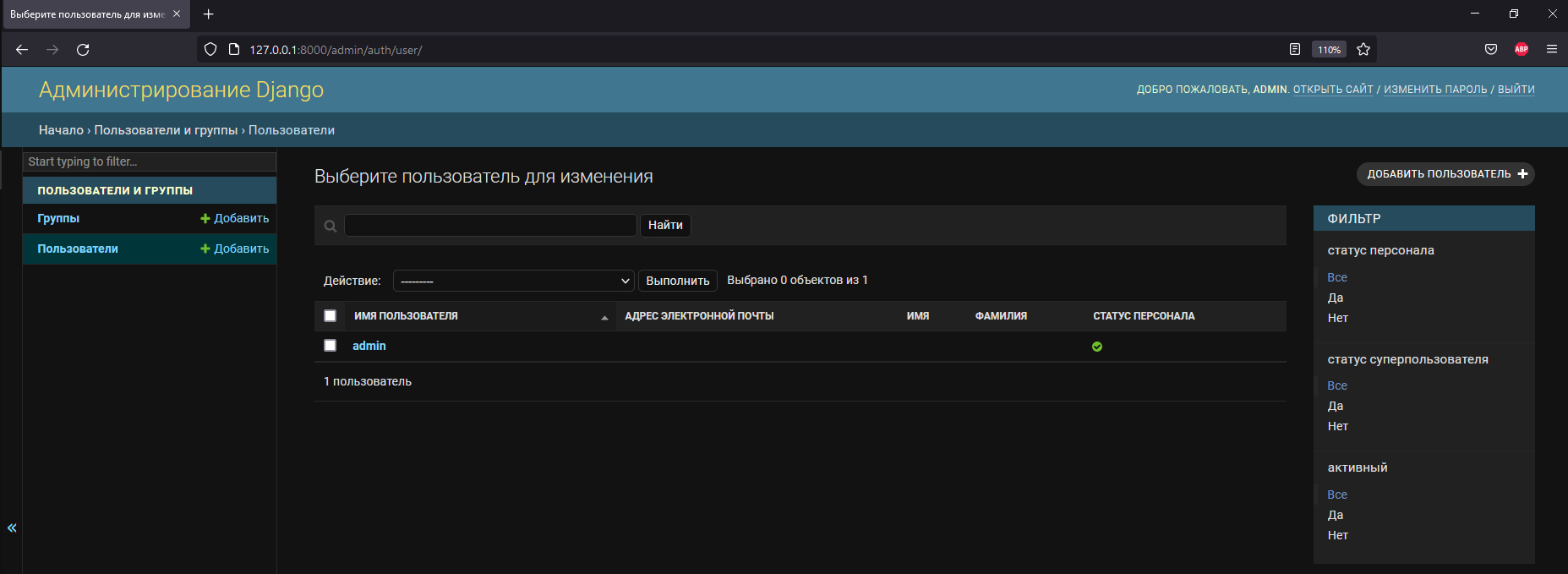


Рисунок 23 - Страница “Пользователи”

Также мы можем перейти на страницу “Группы”.

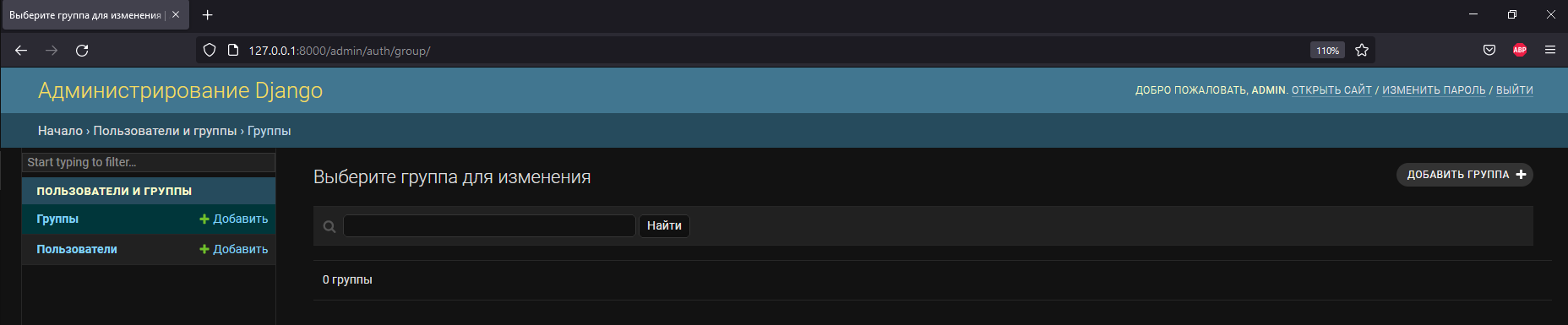


Рисунок 24 - Страница “Группы”

Чтобы мы могли взаимодействовать с нашей созданной моделью базы данных через панель администратора, нам необходимо зарегистрировать её в файле *admin.py* Django-приложения *news*.

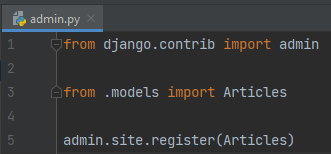


Рисунок 25 - Файл admin.py

После этого она отправляется на сервер админ-панели.

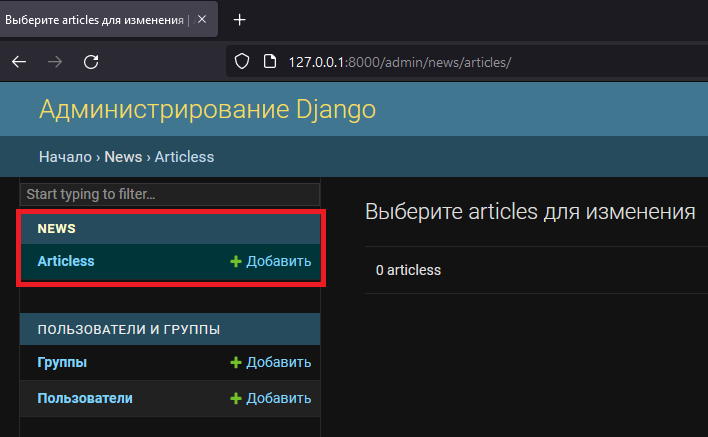


Рисунок 25 - Отображение новостной таблицы

Мы видим, что название у модели в админ-панели не очень хорошее. Изменим его, добавив псевдонимы в мета-класс в файле *models.py*

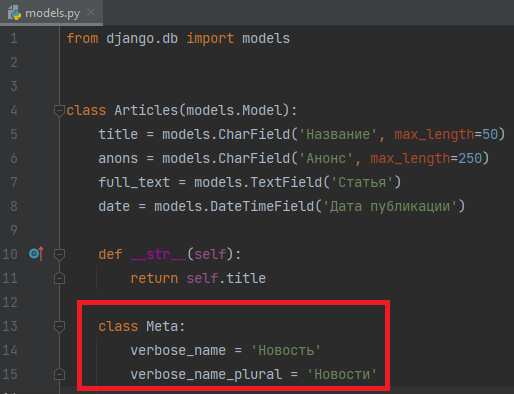


Рисунок 26 - Добавление метаданных в models.py

После добавления метаданных название новостной таблицы отобразится так, как нам необходимо.

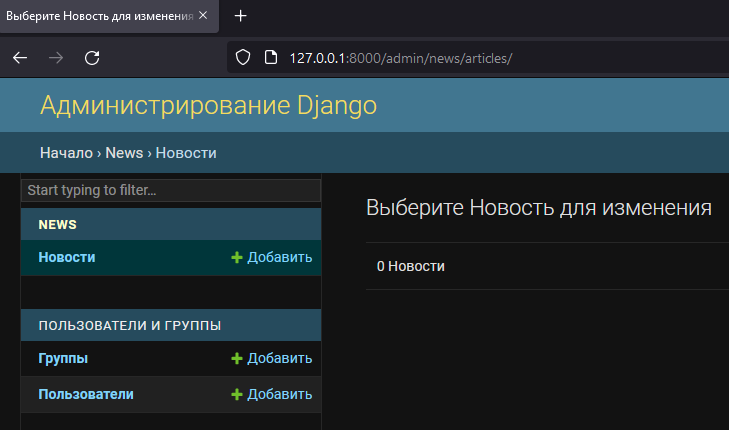


Рисунок 27 - Отображение метаданных новостной таблицы

Далее мы можем добавить запись в базу данных, указав название, анонс, статью и дату публикации.

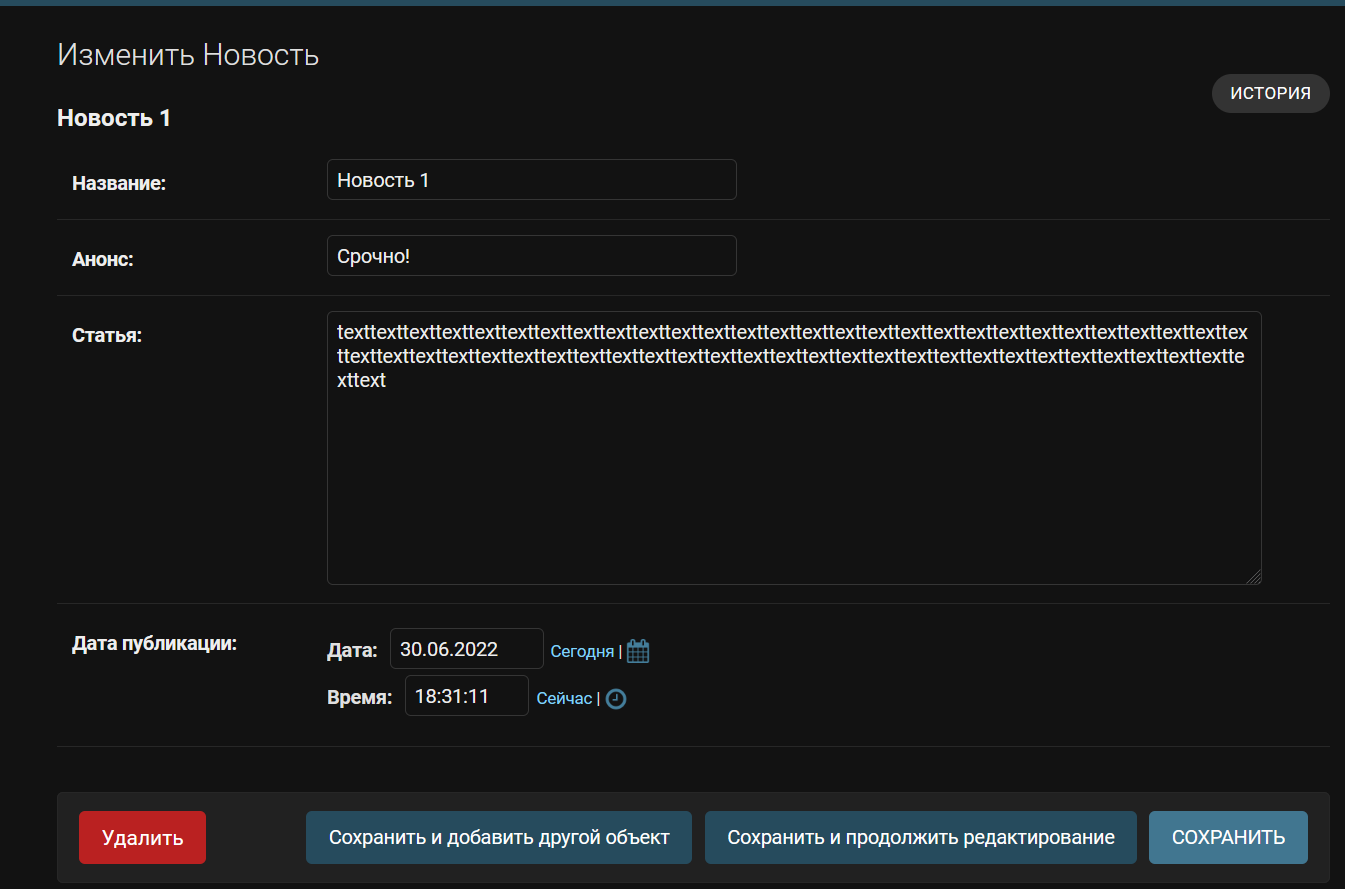


Рисунок 28 - Создание новости

Созданную запись можно увидеть, если перейти в базу данных “Новости”.

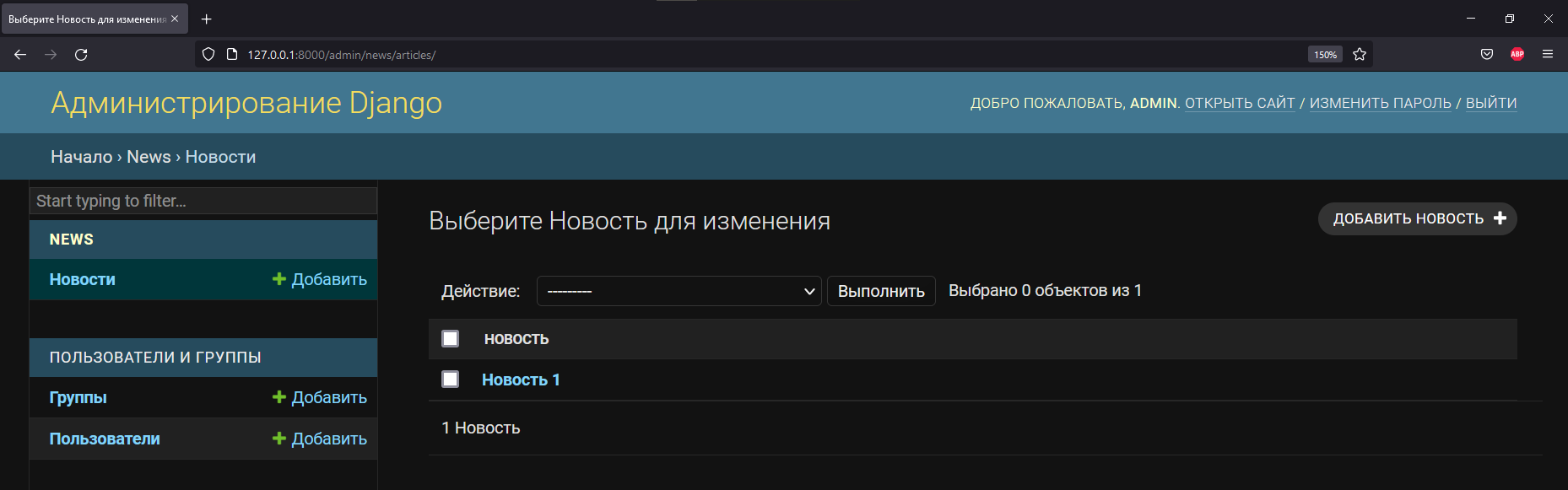


Рисунок 29 - Отображение новости в базе данных