**Методические указания по выполнению практического задания**

**Целью** занятия является практическое освоение теоретических знаний, полученных в процессе изучения темы 7 «Использование баз данных в Django. SQLite. Модели в Django».

**Теория для практического занятия**

Для того, чтобы добавить возможность редактирования и создания статьи блога в административном разделе, необходимо:

1. В начало файла **models.py** необходимо добавить строку импорта модуля административного раздела: *from django.contrib import admin*
2. В файл **models.py** после объявления модели необходимо добавить строку регистрации модели блога (*class* *Blog*) в административном разделе: *admin.site.register(Blog)*

Для того, чтобы добавить в поле даты модели значение по умолчанию равное текущей дате, необходимо:

1. В начало файла **models.py** необходимо добавить строку импорта класса datetime из модуля datetime: *from datetime import datetime*
2. В поле даты добавить значение по умолчанию равное текущей дате: *posted = models.DateTimeField(default = datetime.now(), db\_index = True, verbose\_name = "Опубликована")*

При добавлении метода **get\_absolute\_url** в класс модели необходимо:

1. В начало файла **models.py** добавить строку импорта метода **reverse**: *from django.core.urlresolvers import reverse*

Для того, чтобы добавить статью блога в административном разделе и сохранить её в базе данных веб-проекта db.sqlite3, необходимо:

1. Запустить веб-проект на выполнение.
2. Войти в административный раздел (см. тему 6).
3. В строке «Статьи блога» нажать кнопку «**+Добавить**».
4. В загруженной форме добавления статьи блога необходимо ввести **Заголовок** статьи блога, **Краткое содержание** статьи, **Полное содержание** статьи, **Дату публикации** статьи (поле **Опубликована**).
5. Нажать кнопку «**Сохранить**». Статья добавляется в список статей и может быть **отредактирована при нажатии на гиперссылку названия статьи**.
6. Добавить следующую новую статью блога можно нажав на кнопку  и выполнить пункты 4 и 5.
7. Выйти из административного раздела.

**Задание 1**

Цель задания – получить практические навыки создания модели Django и построения базы данных веб-проекта на её основе в среде Visual Studio 2017.

1. Добавить в веб-проект Django модель для статей блога. Данные статьи блога должны добавляться в административном разделе. Каждая статья блога должна иметь заголовок, краткое содержание, полное содержание и дату публикации.
2. Создать таблицу в БД веб-проекта db.sqlite3 для сохранения статей блога.
3. Добавить в административном разделе три тестовые статьи блога (на определенную тему) и сохранить в базе данных веб-проекта db.sqlite3.

**Пошаговый пример решения задания**

Рассмотрим пример решения задания 1 практического занятия по теме 7.

1. На основе материалов лекции и теории в подразделе **Теория для практического занятия** необходимо:

a. Открыть, созданный ранее, веб-проект Django в Visual Studio 2017. В нём открыть файл **models.py**.

b. В файл **models.py** добавить код класса модели блога – **Blog**, содержащий:

* + - * Поля модели для Заголовка, Краткого содержания, Полного содержания и Даты публикации статьи блога (см. рисунок 1).
      * Методы **get\_absolute\_url** и **\_\_str\_\_** (см. рисунок 1).
      * Метаданные – **class Meta** (см. рисунок 1).

c. В начало файла **models.py** необходимо добавить строку импорта класса **datetime** из модуля datetime (см. рисунок 1).

d. В начало файла **models.py** добавить строку импорта метода **reverse** (см. рисунок 1).

e. В начало файла **models.py** необходимо добавить строку импорта модуля административного раздела (см. рисунок 1).

f. В конец файла **models.py** после объявления модели (class Blog) необходимо добавить строку регистрации модели блога в административном разделе (см. рисунок 1):

*admin.site.register(Blog)*

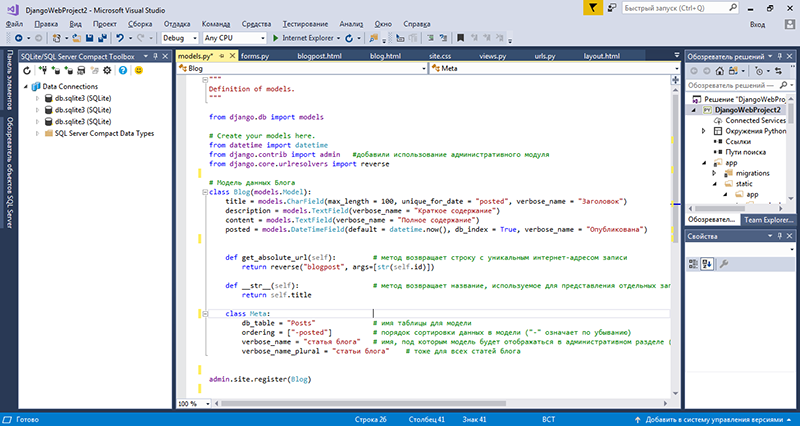


Рисунок 1 – Добавление в веб-проект модели статьи блога

2. На основе материалов лекции необходимо:

a. В меню **Проект** выбрать команду **Миграция сборок Django**

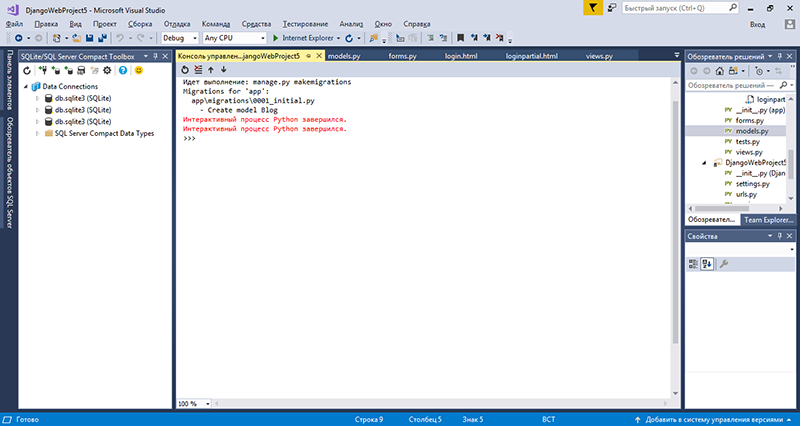


Рисунок 2 – Результат выполнения команды Миграция сборок Django

b. В меню **Проект** выбрать команду **Миграция Django...**

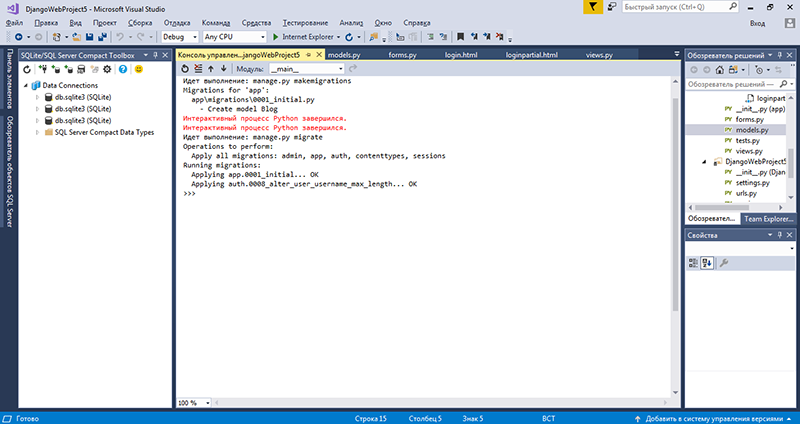


Рисунок 3 – Результат выполнения команды Миграция Django...

3. На основе теории в подразделе **Теория для практического занятия** необходимо:

a. Запустить проект на выполнение.

b. Войти в административный раздел (под учетной записью ранее созданного администратора).

c. В строке «**Статьи блога**» нажать кнопку «**+ Добавить**» (см. рисунок 4).

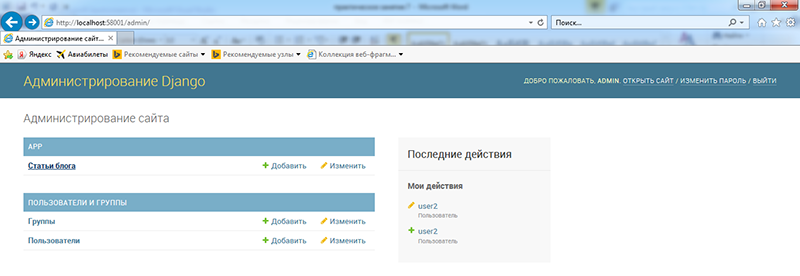


Рисунок 4 – Добавление статьи блога

d. В форме добавления статьи блога необходимо ввести *Заголовок статьи блога*, *Краткое содержание статьи*, *Полное содержание статьи*, дату публикации статьи – поле *Опубликована* (см. рисунок 5)

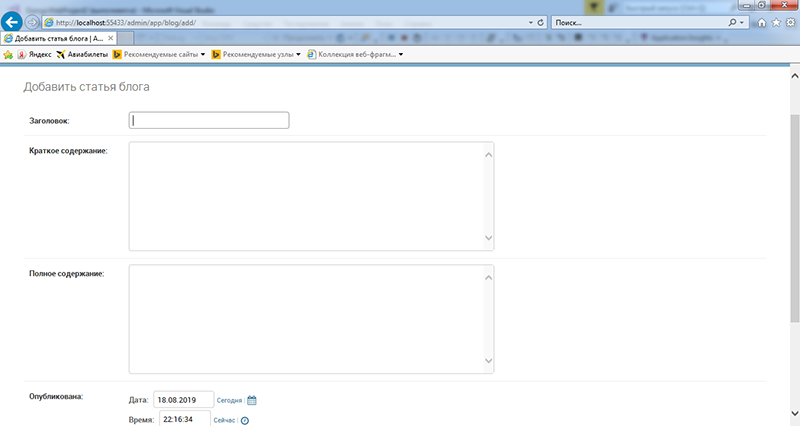


Рисунок 5 – Заполнение формы Добавления статьи блога

e. Нажать кнопку «**Сохранить**» (результат на рисунке 6).

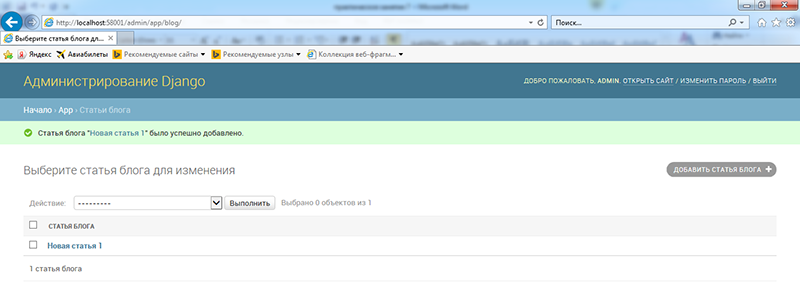


Рисунок 6 – Добавленная статья в списке статей блога

f. Добавить следующую новую статью блога, нажав на кнопку "**ДОБАВИТЬ СТАТЬЯ БЛОГА** **+**"

g. В форме ввести данные новой статьи и сохранить в таблице базы данных (пункты d, e).

h. Добавить следующую новую статью блога, нажав на кнопку "**ДОБАВИТЬ СТАТЬЯ БЛОГА** **+**"

i. В форме ввести данные новой статьи и сохранить в таблице базы данных (пункты d, e).

j. Выйти из административного раздела, нажав на ссылку **ВЫЙТИ**.

**Дополнительный материал**

**Добавление поля автора в модель *Blog* из связанной модели *User*:**

1. Поле для автора в модели *Blog* из модели *User* должно иметь тип ***ForeignKey*** (так как у каждой статьи будет только один автор, но у автора может быть много статей). Поле для автора также должно иметь атрибуты *null, blank, on\_delete* и *verbose\_name (атрибуты null, blank обязательны для установки, т.к. поле автор не заполнено у ранее сохраненных статей)*:

*author = models.ForeignKey(User, null=True, blank=True, on\_delete = models.SET\_NULL, verbose\_name = "Автор")*

2. Так как поле автора добавляется из связанной модели, то в начало файла **models.py** необходимо добавить строку импорта этой модели:

*from django.contrib.auth.models import User*

3. Затем необходимо обновить таблицы базы данных, выполнив миграцию.