

Лабораторная работа №2.

Создание модели. Администрирование Django.

1. Создание приложения.

Веб-приложение или проект Django состоит из отдельных приложений. Вместе они образуют полноценное веб-приложение. Каждое приложение представляет какую-то определенную функциональность или группу функциональностей. Один проект может включать множество приложений. Это позволяет выделить группу задач в отдельный модуль и разрабатывать их относительно независимо от других. Кроме того, мы можем переносить приложение из одного проекта в другой независимо от другой функциональности проекта.

При создании проекта он уже содержит несколько приложений по умолчанию.

- `django.contrib.admin`
- `django.contrib.auth`
- `django.contrib.contenttypes`
- `django.contrib.sessions`
- `django.contrib.messages`
- `django.contrib.staticfiles`

Список всех приложений можно найти в проекте в файле `settings.py` в переменной `INSTALLED_APPS`

Для создания приложения необходимо набрать следующую инструкцию в командной строке (из директории `djangodir`, где находится файл `manage.py`):

```
(lab1)      C:\.....\djangodir>      python      manage.py  
startapp my_app
```

В результате выполнения команды в проекте появилась новая папка `my_app`, которая содержит некоторые файлы. Таким образом, структура проекта теперь выглядит так:

```

djangodir
├── my_app
│   ├── __init__.py
│   ├── admin.py
│   ├── apps.py
│   ├── migrations
│   │   └── __init__.py
│   ├── models.py
│   ├── tests.py
│   └── views.py
├── db.sqlite3
├── manage.py
└── lab_1_site
    ├── __init__.py
    ├── settings.py
    ├── urls.py
    └── wsgi.py
```

После того, как приложение создано, нужно сообщить Django, что теперь он должен его использовать. Сделаем это с помощью файла `lab_1_site/settings.py`. Нужно найти `INSTALLED_APPS` и добавить к списку `'my_app'`, прямо перед `]`. Конечный результат должен выглядеть следующим образом:

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
```

```
'django.contrib.contenttypes',  
'django.contrib.sessions',  
'django.contrib.messages',  
'django.contrib.staticfiles',  
'my_app',
```

```
]
```

2. Создание модели

Для создания, изменения, удаления моделей необходимо редактировать файл `my_app/models.py`.

Для примера создадим модель для текстовых статей на сайте:

```
from django.db import models
from django.utils import timezone

class Post(models.Model):
    author = models.ForeignKey('auth.User',
on_delete=models.CASCADE)

    title = models.CharField(max_length=200)
    text = models.TextField()
    created_date = models.DateTimeField(
        default=timezone.now)
    published_date = models.DateTimeField(
        blank=True, null=True)

    def publish(self):
        self.published_date = timezone.now()
        self.save()

    def __str__(self):
        return self.title
```

В результате по данной модели в базе данных будет создана таблица `Post` со следующими полями:

- author – автор статьи. Внешний ключ к таблице auth.User. Связь один-ко-многим
- title – заголовок статьи
- text – текст статьи
- created_date – дата создания
- published_date – дата публикации

Также в данной модели реализованы два метода:

`publish()` – при выполнении изменяет дату публикации статьи

`__str__()` – переопределение магического метода Python для данной модели.

3. Создание таблицы моделей в базе данных

Созданные модели необходимо добавить в базу данных. Необходимо дать Django знать, что файл `models.py` был изменен:

```
(lab1) C:\.....\djangodir> python manage.py
makemigrations my_app
```

Результате выполнения команды будет следующим:

```
Migrations for 'my_app':
  my_app/migrations/0001_initial.py:
    - Create model Post
```

Django создал файл с миграцией для базы данных. Теперь необходимо выполнить:

```
(lab1) C:\.....\djangodir> python manage.py migrate
my_app
```

Результат должен быть следующим:

```
Operations to perform:
```

```
  Apply all migrations: my_app
```

```
Running migrations:
```

```
  Rendering model states... DONE
```

```
  Applying my_app.0001_initial... OK
```

В итоге модель статьи теперь в базе данных.

4. Администрирование Django

Чтобы добавлять, редактировать и удалять записи, для которых мы только что создали модель, мы используем панель управления администратора Django.

Для доступа к странице администрирования нужно создать *суперпользователя* (англ. *superuser*) – пользователя, который имеет полный доступ к управлению сайтом. Для этого выполняем команду:

```
(lab1) C:\.....\djangodir> python manage.py  
createsuperuser
```

```
Username: admin
```

```
Email address: admin@admin.com
```

```
Password:
```

```
Password (again):
```

```
Superuser created successfully.
```

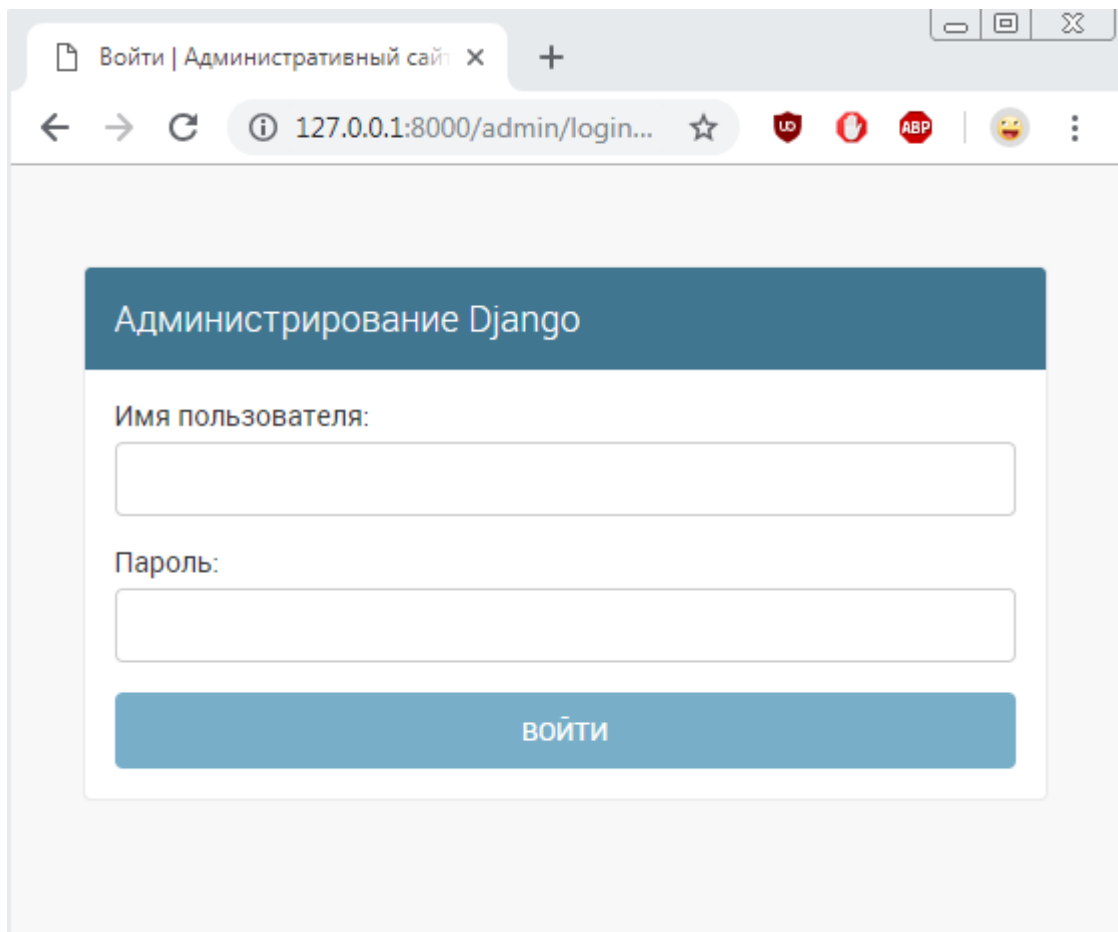
Чтобы созданная модель стала доступна на странице администрирования, нужно зарегистрировать её. Для этого необходимо отредактировать файл `my_app/admin.py`:

```
from django.contrib import admin
from .models import Post

admin.site.register(Post)
```

Теперь можно открыть в браузере адрес 127.0.0.1:8000/admin/

Откроется страница авторизации:

A screenshot of a web browser window showing the Django Admin login page. The browser's address bar displays '127.0.0.1:8000/admin/login...'. The page has a dark blue header with the text 'Администрирование Django'. Below the header, there are two input fields: 'Имя пользователя:' (Username) and 'Пароль:' (Password). At the bottom of the form is a blue button labeled 'ВОЙТИ' (Login). The browser's tab shows 'Войти | Административный сайт'.

Необходимо войти в систему с помощью логина и пароля суперпользователя, созданного ранее.

После авторизации откроется панель управления Django:

Администрирование Django

ДОБРО ПОЖАЛОВАТЬ, JUSTAWAY. [ОТКРЫТЬ САЙТ](#) / [ИЗМЕНИТЬ ПАРОЛЬ](#) / [ВЫЙТИ](#)

Администрирование сайта

BLOG

Posts

[+ Добавить](#)

[✎ Изменить](#)

ПОЛЬЗОВАТЕЛИ И ГРУППЫ

Группы

[+ Добавить](#)

[✎ Изменить](#)

Пользователи

[+ Добавить](#)

[✎ Изменить](#)



Теперь можно открыть раздел Posts и добавить пару статей:

Администрирование Django
ДОБРО ПОЖАЛОВАТЬ, **JUSTAWAY**. [ОТКРЫТЬ САЙТ](#) / [ИЗМЕНИТЬ ПАРОЛЬ](#) / [ВЫЙТИ](#)

Начало » Blog » Posts » Добавить post

Добавить post

Author:

justaway ▾  


Title:


Прикладное ПО 1

Text:


Документация Django: <https://docs.djangoproject.com/en/2.0/>
Документация Django: <https://docs.djangoproject.com/en/2.0/>
Документация Django: <https://docs.djangoproject.com/en/2.0/>
Документация Django: <https://docs.djangoproject.com/en/2.0/>
Документация Django: <https://docs.djangoproject.com/en/2.0/>
Документация Django: <https://docs.djangoproject.com/en/2.0/>


Created date:

Дата: 30.09.2018  Сегодня

Время: 12:42:18  Сейчас

Published date:

Дата: 30.09.2018  Сегодня

Время: 12:45:21  Сейчас

Сохранить и добавить другой объект

Сохранить и продолжить редактирование

СОХРАНИТЬ

Задание к лабораторной работе.

1. Выбрать вариант задания.
2. Создать несколько моделей в соответствии со своим вариантом.
3. Изучить страницу администрирования Django.

Варианты заданий

1. Сайт расписания занятий группы
2. Сайт высшего учебного заведения
3. Блог с публикациями

4. Новостной сайт
5. Сайт-визитка предприятия с информацией о продукции
6. Сайт – энциклопедия
7. Сайт заметок
8. Сайт «Список дел»
9. Сайт «Доставка еды»
10. Сайт подсчета голосов
11. Текстовый квест