Лабораторная работа № 4

Создание страницы определенной записи

Вы уже создавали страницу для всех постов, теперь создадим страницу отдельного поста. Для начала необходимо будет создать адрес будущей страницы, затем написать функцию-представление для нее и реализовать шаблон. Адрес страницы для одной записи обязательно должен нести в себе какое-то ключевое слово (ключ), чтобы можно было среди всей таблицы базы данных найти интересующий пользователя экземпляр. Для этого можно использовать **id** (уникальный идентификатор) сообщений, которые устанавливаются автоматически при создании каждого отдельного объекта сообщений. В итоге адрес страницы для статьи с идентификатором 5 должен выглядеть так: http://127.0.0.1:8000/article/5/.

В Django довольно часто используются регулярные выражения (формальный язык поиска и осуществления манипуляций с подстроками в тексте, основанный на использовании метасимволов), они нужны для того, чтобы было легче извлекать данные из адресов, по которым приходят запросы на сервер.

Для начала выполнения лабораторной работы, в директории **lab4** создайте новый проект **blog**, проведите первоначальную настройку и перенесите приложение articles из предыдущей работы (скопируйте папку в этот проект, добавьте нужные данные в INSTALLED_APP и настройте файл **urls.py**).

Для того, чтобы можно было бы прочитать значение идентификатора (id) отдельного поста укажите в качестве адреса следующее регулярное выражение (в файле urls.py):

```
from django.urls import path, re_path


urlpatterns = [
    re_path(r'^article/(?P<article_id>\d+)$', views.get_article, name='get_article'),]
```

В качестве первого аргумента передается регулярное выражение, которое возвращает именованную группу article_id (именованная группа – это переменные внутри регулярного выражения, которые имеют следующий синтаксис: “(?P<name> ...)”, где name вы должны указать сами, а вместо троеточия подставить нужный вам шаблон). Обратите внимание, что именованная группа идет после символов “article/”, и состоит она только из цифр,

продолжающихся до завершения адреса. Соответственно, если хоть одно из условий не подходит (например, адрес начинается с “artikle/” вместо “article/”, или та комбинация, что идет после “article/” состоит не только из цифр, но и из латинских букв), то регулярное выражение не подойдет и представление views.get_article не запустится.

После того, как в регулярном выражении вы объявили именованную группу, в представление можно добавлять именованный аргумент, куда и будет передано значение из адреса, по которому перешел пользователь. Создайте в файле **views.py** из директории **articles** помимо уже написанной в предыдущей работе функции archive ещё и функцию get_article:

```
from django.http import Http404
def get_article(request, article_id):
  try:
     post = Article.objects.get(id=article_id)
     return render(request, 'article.html', {"post": post})
except Article.DoesNotExist:
     raise Http404
```

Здесь использовался синтаксис обработки исключений, который помогает вручную обрабатывать возникающие ошибки (если ошибка была именно той, что указано после ключевого слова except). Если возникла ошибка Article.DoesNotExist (элемент с указанным id, который искался через функцию get(), отсутствует) возбуждается другая ошибка, системная, которая будет обработана автоматически платформой Django – ошибка Http404. В случае возникновения исключения Http404, Django вернет стандартную страницу ошибки 404 (Page not found).

Задание:

- Теперь у определенной записи есть отдельная страница, а значит на странице списка постов каждый заголовок можно сделать ссылкой. Сделайте так, чтобы при клике по названию статьи происходил переход на страницу указанной записи. Для этого достаточно воспользоваться тегом <a>, которому в атрибуте href указать адрес перехода.

Верстка обеих страниц в соответствии с макетом

В папке articles/templates/ создайте файл **article.html**. Содержимое его вы можете определить сами, оно будет проще, чем описанная в прошлой лабораторной работе страница вывода всех записей, ведь здесь запись всего одна.

Теперь вам необходимо создать правильные стили. Макет страницы одной записи должен выглядеть как на рисунке 5 (lab4_item), а макет главной страницы со списком всех постов должен выглядеть как на рисунке 6 (lab4_main).

Рисунок 5. Макет страницы с одной статьей

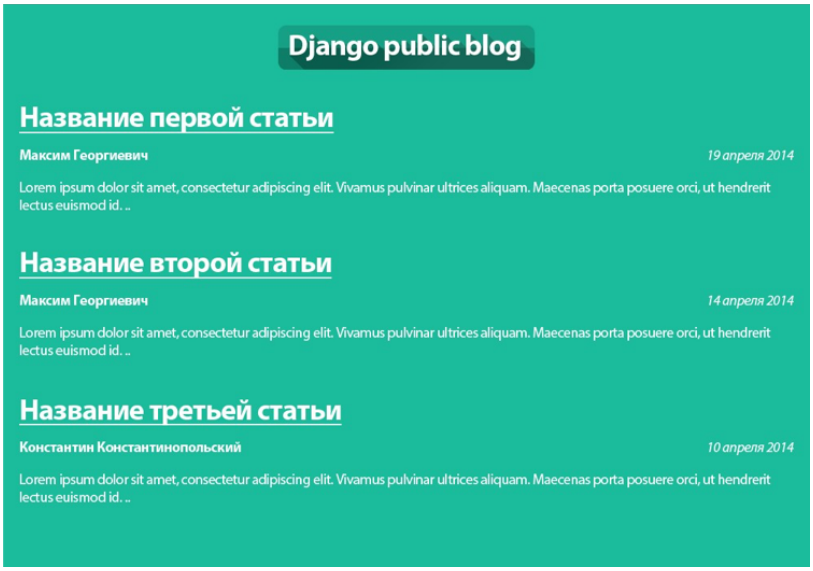

Рисунок 6. Макет страницы со списком всех статей

Для того, что придать внешние стили странице, нужно подключить css-файл. Создайте в директории articles/static/css/ файл **article.css**. Теперь подключите этот файл к своим html-шаблонам (articles/templates/archive.html и articles/templates/article.html) так, как вы это делали в лабораторной работе №2.

Для того, чтобы задавать стили, специфичные для одной из страниц (например, только для article.html) тегу <body> каждой страницы задайте уникальный класс. Пусть для **archive.html** это будет класс archive, а для страницы **article.html** будет класс article. Теперь в описании стилей можно указать селектор, например: `.article h1 {}` Тогда этот стиль применится ко всем заголовкам первого уровня на странице одной записи, но не на странице всех статей (архива).

Задайте общие стили страницы, в теге <body> укажите:

```
background: #1abc9c;
font-family: Tahoma, Arial, sans-serif;
color: #ffffff;
```

Чтобы горизонтально выровнять логотип (картинку), достаточно указать определенную ширину и боковые отступы, а также тегам img нужно явно указать, что это блочный элемент:

```
.header img {
     display: block;
     width: 318px;
     margin-left: auto;
     margin-right: auto;
}
```

Теперь логотип должен располагаться по центру. По такому же принципу необходимо пометить по центру остальной контент этой страницы:

```
.archive {
     width: 960px;
     margin-left: auto;
     margin-right: auto;
}
```

После того, как вы сделали названия всех постов ссылками, их цвет стал синий. Исправьте это, указав цвет белым:

```
.post-title a {
     color: #ffffff;
}
```

Вам необходимо, чтобы имя автора и время создания статьи находились на одном горизонтальном уровне. Имени автора задайте «обтекание слева», при этом указав ширину меньше, чем вся страница, иначе поле «время создания» не поместится справа. Например:

```
.article-author {
```

```
    width: 50%;
    float: left;
}
```

Чтобы поле «время создания» находилось в самой правой части экрана, задайте соответствующее выравнивание текста:

```
.article-created-date {
    text-align: right;
}
```

Задание:

- Создайте стили, соответствующие макету для страницы определенной записи. Не забудьте добавить ссылку «Все записи»;
- Загрузите ваш проект на любой гит-репозиторий (GitHub, GitLab, Google Code, Bitbucket и т.п.).