Шпаргалка Первое приложение

models

Данными в Django управляют через файлы *models.py*, эти файлы хранят в директории приложения.

```
icecream
├── models.py  # данными приложения управляют здесь
├── views.py  # view-функции
└── urls.py  # список url приложения
```

HTML-теги

В HTML всю информацию на веб-странице размечают тегами (англ. *tag*, «метка»); вид страницы в браузере зависит от разметки.

Теги бывают

- одиночные (например, `<br>`);
- парные: у них есть открывающий и закрывающий тег, между которыми пишется содержимое тега (например, `<p>` Это содержимое тега *p* `</p>`).

Теги можно вкладывать друг в друга.

Теги, используемые в проекте:

```
<!DOCTYPE html> <!-- На странице используется 5-я версия HTML -->
<html> <!-- Начало HTML-кода -->
    <head> <!-- Тут служебная информация. Она не будет отображаться на экране -->
        <meta charset="UTF-8"> <!-- На странице корректно отобразится русский язык -->
    </head>
    <body> <!-- Тело страницы. Всё, что в этом теге, будет показано на странице  -->
        <h1>Текст большого заголовка</h1> <!-- Заголовок первого уровня, крупный -->
        <h2>Текст заголовка поменьше</h2> <!-- Заголовок второго уровня, помельче, но тоже крупный -->
        <a href="https://praktikum.yandex.ru/">Практикум</a>
        <!-- Ссылка: клик по слову "Практикум" отправит пользователя
        на страницу praktikum.yandex.ru -->
        <p> <!-- абзац -->
           Текст
            <br> <!-- перенос строки -->
           абзаца
        </p>
    </body>
</html>
```

HTML-шаблоны

Шаблоны — это заготовки для создания HTML-страниц, файлы с HTML-кодом, в который можно вставить данные.

Применение шаблонов позволяет вернуть пользователю красиво сформатированную и оформленную веб-страницу с нужной информацией.

Структура проекта с шаблонами:

```
├── icecream # приложение
├── home  # приложение
├── anfisa4friends  # общие настройки проекта
└── templates
    ├── icecream # шаблоны приложения icecream
    │   ├── icecream-detail.html
    │   └── icecream-list.html
    └── homepage # шаблоны приложения homepage
        └── index.html
```

Render

Функция `render()` «монтирует» веб-страницу из HTML-шаблона и данных, которые надо внедрить в этот шаблон.

```
from django.shortcuts import render # Импорт функции render в код

def view_function(request):
    # Готовим нужную информацию для передачи в шаблон: складываем всё в словарь context
    # 'ключ': значение
    context = {
        'key1': value1,
        'key2': value2,
        ...
    }
    # Передаём в render() имя шаблона и данные, которые надо в него вставить
    return render(request, 'путь_к_шаблону', context)

# Например: return render(request, 'icecream/icecream-detail.html', context)
```

А в шаблон выводим названия ключей, на эти места будут вставлены значения этих ключей:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
</head>
<body>
    <h2>{{ key1 }}</h2>
    <p>{{ key2 }}</p>
</body>
</html>
```

Параметры в urls

Однотипные URL можно обрабатывать одной view-функцией.

Для адресов

`album/1/`

`album/2/`

`album/3/`

можно создать общее правило с шаблоном адреса `album/<int:pk>/` и вызывать одну для всех view-функцию `detail()`:

```
# Файл urls.py

urlpatterns = [
    ...
    path('album/<int:pk>/', views.detail),
]
```

А view-функция будет принимать изменяемую часть адреса и как-то обрабатывать её:

```
# Файл views.py
def detail(request, pk):
    output=''
    if(pk==1)
        output='Первый великий музыкальный альбом'
    ...
```