Практическое занятие

Тема: Использование специальных тегов в шаблонах HTML-страниц.

Цель: научиться пользоваться специальными тегами в шаблонах HTML-страниц.

Оборудование: персональный компьютер, IDE PyCharm, Visual Studio Code, методические рекомендации к проведению лекционных занятий, Литература: Дронов В. А. Django. Практика создания веб-сайтов на Python - Санкт-Петербург : БХВ-Петербург, 2023. - 800 с., Постолит А. Python, Django и PyCharm для начинающих – - Санкт-Петербург : БХВ-Петербург, 2021. – 464 с., METANIT.COM: сайт о программировании. - URL: https://metanit.com/sharp/ – режим доступа: свободный, инструкционная карта для проведения лекционного занятия.

В Django предоставлена возможность использовать в шаблонах ряд специальных тегов, которые упрощают вывод некоторых данных. Рассмотрим некоторые наиболее часто используемые теги.

Тег для вывода текущих даты и времени. Для вывода дат в Django используется следующий тег: {% now "формат_данных" %}

Тег now позволяет вывести системное время. В качестве параметра тегу now передается формат данных, который указывает, как форматировать время и дату. Для форматирования времени и дат используются следующие символы

№ п/п	Символ	Значение даты и времени
1	Y	Год в виде четырех цифр (2021)
2	y	Год в виде последних двух цифр (21)
3	F	Полное название месяца (Июль)
4	M	Сокращенное название месяца — 3 символа (Июл)
5	m	Номер месяца — две цифры (07)
6	N	Аббревиатура месяца в стиле Ассошизйтед Пресс
7	n	Номер месяца — одна цифра (7)
8	j	День месяца (1–31)
9	l	День недели — текст (среда)
10	h	Часы (0–12)— 9:15
11	H	Часы (0–24)— 21:15
12	i	Минуты (0–59)
13	s	Секунды (0–59)

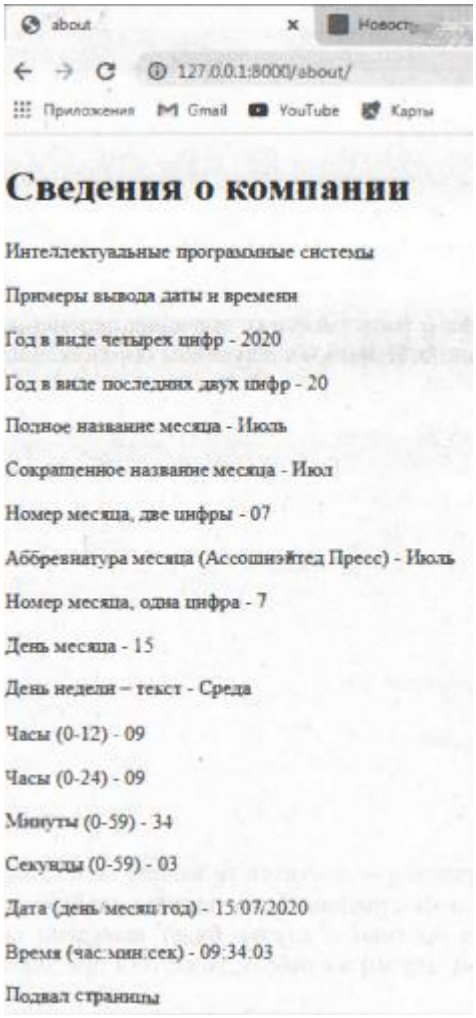
Все возможные форматы для вывода даты и времени можно посмотреть в оригинальной документации на Django.

Изменим следующим образом код шаблона страницы about.html, которая была создана ранее

```
{% extends "firstapp/base.html" %}
{% block title %}about{% endblock title %}
{% block header %}Сведения о компании{% endblock header %}
{% block content %}
```

```
<p>Интеллектуальные программные системы</p>
<p>Примеры вывода даты и времени</p>
<p>Год в виде четырех цифр  - {% now "Y" %}</p>
<p>Год в виде последних двух цифр - {% now "y" %}</p>
<p>Полное название месяца  - {% now "F" %}</p>
<p>Сокращенное название месяца  - {% now "M" %}</p>
<p>Номер месяца, две цифры - {% now "m" %}</p>
<p>Аббревиатура месяца (Ассошиэйтед Пресс) - {% now "N" %}</p>
<p>Номер месяца, одна цифра - {% now "n" %}</p>
<p>День месяца - {% now "j" %}</p>
<p> День недели – текст - {% now "l" %}</p>
<p>Часы (0-12) - {% now "h" %}</p>
<p>Часы (0-24) - {% now "H" %}</p>
<p>Минуты (0-59) - {% now "i" %}</p>
<p>Секунды (0-59) - {% now "s" %}</p>
<p>Дата (день/месяц/год) - {% now "j/m/Y" %}</p>
<p>Время (час:мин:сек) - {% now "H:i:s" %}</p>
{% endblock content %}
```

Если обратиться к странице about.html после этих изменений, то мы получим следующий результат



Пример страницы сайта about.html (варианты вывода текущих даты и времени)

Тег для вывода информации по условию. Тег для вывода информации в зависимости от соблюдения какого-либо условия выглядит следующим образом: {% if %} {% endif %}

В качестве параметра тегу if передается выражение, которое должно возвращать True ИЛИ False. Предположим, что из представления view в шаблон передаются некоторые значения - например, возраст клиента (age). Изменим следующим образом текст функции def index () в файле view.py

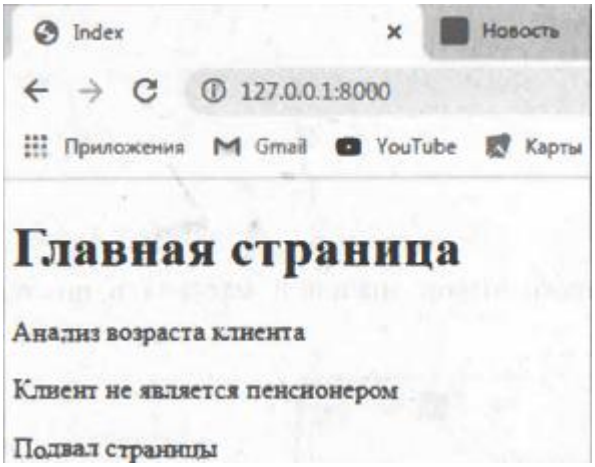
```
from django.http import *
from django.shortcuts import render


def index(request):
    data = {"age": 50}
    return render(request, "firstapp/index.html", context=data)
```

В шаблоне страницы firstapp\index.html в зависимости от значения переменной age мы можем выводить разную информацию. Изменим следующим образом текст в файле firstapp\index.htm

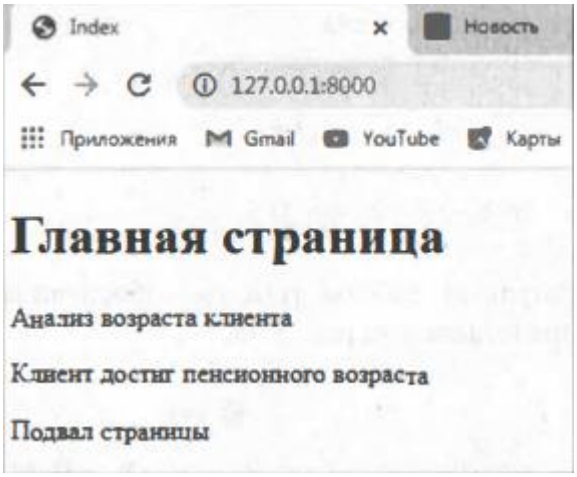
```
{% extends "firstapp/base.html" %}
{% block title %}Index{% endblock title %}
{% block header %}Главная страница{% endblock header %}
{% block content%}
<p>Анализ возраста клиента</p>
{% if age > 60 %}
    <p>Клиент достиг пенсионного возраста</p>
{% else %}
    <p>Клиент не является пенсионером</p>
{% endif %}
{% endblock content %}
```

На этой странице мы проверяем условие - является ли клиент пенсионером. Если возраст клиента больше 65 лет, то на странице будет выдано сообщение Клиент достиг пенсионного возраста, в противном случае будет выведено сообщение Клиент не является пенсионером. Проверим работу этого тега при значении возраста: {"age" : 50)



Пример работы тега if на странице сайта index.html (при возрасте клиента 50 лет)

Теперь изменим следующим образом текст функции def index () в файле view.py - укажем возраст клиента 66 лет: data = {"age" : 66). В этом случае на главной странице firstapp\index.html мы получим другое сообщение



Пример работы тега if на странице сайта lndex.html (при возрасте клиента 66 лет)

Тег для вывода информации в цикле. Тег for позволяет создавать циклы. Этот тег принимает в качестве параметра некоторую коллекцию и пробегается по этой коллекции, обрабатывая каждый ее элемент. Тег имеет следующую структуру:
{% for "Индекс элемента" in "Коллекция элементов" %)
{% endfor %}

Предположим, что из представления view в шаблон передается массив значений - например, категории товаров (cat). Изменим следующим образом текст функции def index () в файле view.py

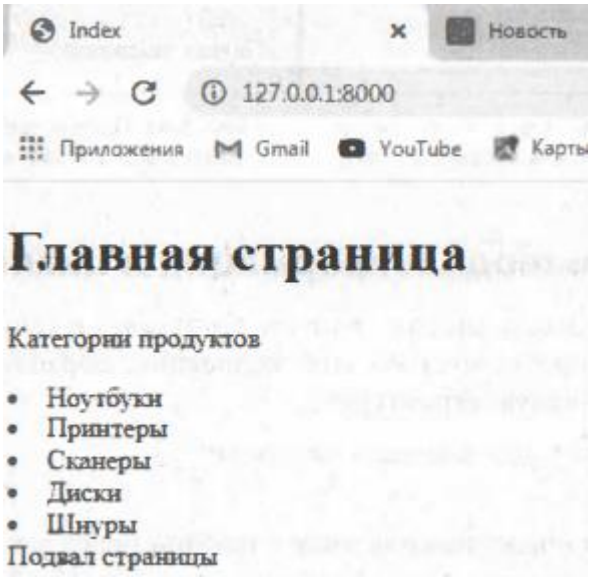
```
from django.http import *
from django.shortcuts import render


def index(request):
    cat = ["Ноутбуки", "Принтеры", "Сканеры", "Диски", "Шнуры"]
    return render(request, "firstapp/index.html", context={"cat": cat})
```

Чтобы в шаблоне страницы firstapp\index.html в цикле выводить информацию из массива cat, изменим следующим образом код в файле firstapp\index.html

```
{% extends "firstapp/base.html" %}
{% block title %}Index{% endblock title %}
{% block header %}Главная страница{% endblock header %}
{% block content%}
<p>Категории продуктов</p>
{% for i in cat %}
        <li>{{ i }}</li>
    {% endfor %}
{% endblock content %}
```

Результат работы тега for, обеспечивающего вывод значений массива в цикле, представлен на рисунке:

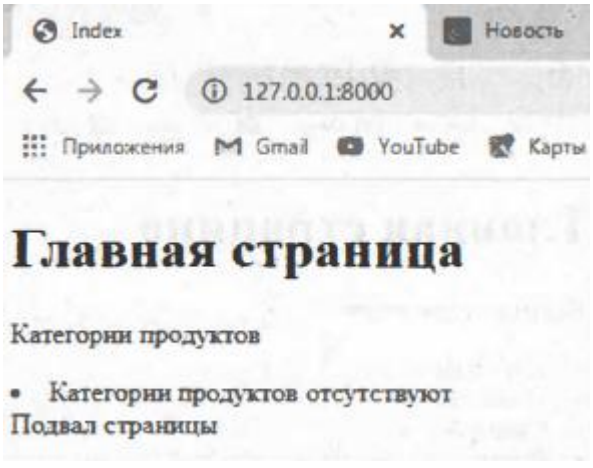


Пример работы тега for на странице сайта index.html

Вполне возможно, что массив, переданный из представления view в шаблон, окажется пустым. На этот случай мы можем использовать дополнительный тег: {% empty %}. Для этого блок content на странице firstapp\index.html можно изменить следующим образом

```
{% block content%}
<p>Категории продуктов</p>
{% for i in cat %}
      <li>{{ i }}</li>
   {% empty %}
     <li>Категории продуктов отсутствуют</li>
   {% endfor %}
{% endblock content %}
```

Если мы теперь в файле view.py обнулим массив cat, т. е. сделаем его «пустым»: cat = [] то тег for сработает следующим образом



Пример работы тега for на странице сайта index.html при «пустом» массиве категорий продуктов

Тег для задания значений переменным. Если требуется определить переменную и использовать ее внутри шаблона, то для этого можно использовать тег: { % with % 1. Изменим следующим образом код в файле firstapp\index.html

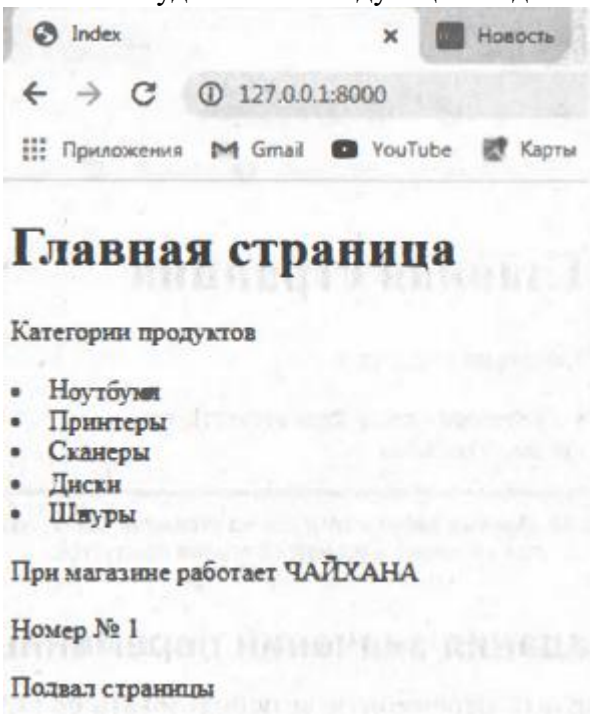
```
{% extends "firstapp/base.html" %}
```

```
{% block title %}Index{% endblock title %}
{% block header %}Главная страница{% endblock header %}
{% block content%}
<p>Категории продуктов</p>
{% for i in cat %}
       <li>{{ i }}</li>
   {% empty %}
      <li>Категории продуктов отсутствуют</li>
   {% endfor %}

{% with name="ЧАЙХАНА" nom=1 %}
   <div>
      <p>При магазине работает {{ name }}</p>
      <p>Номер № {{ nom }}</p>
   </div>
{% endwith %}

{% endblock content %}
```

Здесь внутри шаблона мы определили и вывели значения двух переменных: narne и nom. После этих изменений страница firstapp\index.html будет иметь следующий вид



Пример работы тега with на странице сайта lndex.html

Отчет к практическому занятию

Требования к отчету: скомпилированный проект со скриншотами хода работы, ответы на контрольные вопросы – файл в формате .docx, при необходимости блок-схема к созданной программе в MS Visio.

Литература: методические рекомендации к проведению лекционных занятий, Литература: Дронов В. А. Django. Практика создания веб-сайтов на Python - Санкт-Петербург : БХВ-Петербург, 2023. - 800 с., Постолит А. Python, Django и PyCharm для начинающих – - Санкт-Петербург : БХВ-Петербург, 2021. – 464 с., METANIT.COM: сайт о программировании. - URL: https://metanit.com/sharp/ – режим доступа: свободный, инструкционная карта для проведения лекционного занятия.

Критерии оценки:

Оценка «отлично» - работа выполнена в полном объеме с соблюдением необходимой последовательности ее проведения; самостоятельно и рационально загрузил необходимое программное обеспечение, все задания выполнил в условиях и режимах, обеспечивающих получение результатов и выводов с наибольшей точностью.

Оценка «хорошо» - работа выполнена в полном объеме с соблюдением необходимой последовательности ее проведения; самостоятельно и рационально загрузил необходимое программное обеспечение, но задания выполнил в условиях, не обеспечивающих достаточной точности результатов, или допущено 2-3 недочета.

О**ценка «удовлетворительно»** - работа выполнена не полностью, но объем выполненной части позволяет получить правильные результаты и выводы по основным, принципиально важным задачам работы.

Оценка «неудовлетворительно» - работа выполнена не полностью, объем выполненной части не позволяет сделать правильных выводов.