**Методические указания по выполнению практического задания**

**Целью** занятия является практическое освоение теоретических знаний, полученных в процессе изучения темы 5 «Веб-формы в Django. Валидация. JavaScript».

**Методические указания по выполнению практического задания 1**

1. Прежде чем приступить к выполнению задания 1, необходимо изучить презентацию, конспект и глоссарий к лекции 7.
2. Для закрепления представления о понятиях, введенных на лекции 7, найдите в конспекте к лекции информацию о создании форм средствами языка HTML и о классах библиотеки Django, обеспечивающих автоматическую генерацию HTML-разметки форм и валидацию полей форм. По материалам лекции 7 ознакомьтесь с обработкой данных форм в методах действия.
3. Для освоения технологии создания форм в веб-проекте Django в среде Visual Studio 2017:

* прочитайте теорию для практического занятия;
* выполните [**задание 1**](https://courses.openedu.ru/courses/course-v1:spbstu+WEBPYT+spring_2021/courseware/56dd3a021e4745a6b09b3644d9c0a5b8/671b9760d4264c4696bcec6e475a0a81/?child=first#06).

**Теория для практического занятия**

Для создания формы в веб-проекте Django в среде Visual Studio 2017 необходимо:

1. Запустить Visual Studio 2017.
2. Открыть в Visual Studio 2017 ранее созданный веб-проект Django.
3. Добавить описание класса новой формы в файл **app\forms.py**.
4. Добавить разметку формы в файл шаблона HTML-страницы в папке app\templates\app.
5. В файл **app\views.py** после строк **from ... import ...** добавить строку импорта созданного класса формы:  
   **from .forms import имя\_класса\_формы**
6. В файл **app\views.py**добавить метод действия, обрабатывающий запрос к странице с формой, выполняющий создание объекта класса формы и его передачу методу**render** по ключу **‘form’** в составе словаря, передаваемого третьим аргументом метода.

Для настройки валидации полей формы необходимо:

1. Для полей, заполнение которых не обязательно, в описании класса формы передать в функцию создания поля аргумент **required=False.**
2. Для ограничения минимальной и максимальной длины текстовых полей передать в функции создания таких полей аргументы **min\_length**и**max\_length**с соответствующими значениями**.**
3. Для ограничения минимального и максимального значения числовых полей передать в функцию создания таких полей аргументы **min\_value**и**max\_value**с соответствующими значениями**.**

Для приема данных из заполненной и отправленной пользователем формы необходимо:

1. Проверить в методе действия значение атрибута **request.method***.*Если его значение**‘POST’**, то создать экземпляр класса формы, передав в функцию создания объекта формы атрибут **request.POST**.
2. Если на шаге 1 объект формы был создан, вызвать метод **form.is\_valid()**. Если метод вернет True, получить данные формы из словаря **form.cleaned\_data**по ключам, являющимся текстовыми строками, содержащими имена полей класса формы.
3. Если метод **form.is\_valid()** вернул False, передать объект form методу render для дальнейшего редактирования содержания формы пользователем.
4. Если на шаге 1 атрибут **request.method** не имел значение **‘POST’**, создать новый экземпляр класса формы и передать его методу **render**для отображения пустой формы, которую должен будет заполнить пользователь.

Для передачи набора данных в шаблон страницы для отображения пользователю необходимо:

1. Создать в методе действия переменную-словарь вызовом функции **dict()**и сохранить в ней по ключам набор данных, который будет передаваться в шаблон страницы.
2. Включить созданный словарь как значение, записанное по определенному ключу, в словарь, который передаётся в методе действия методу**render**.

Для отображения в составе веб-страницы данных, переданных из метода действия необходимо:

Включить в вёрстку страницы переменные, содержащие значения из словаря с набором данных, переданным из метода действия. Например, если ключ словаря был **‘data’**, а ключ значения, содержащего имя пользователя - **‘name’**, то для вставки этого имени в шаблон страницы используется переменная **{{ data.name }}**.

Для создания шаблона страницы, отображающей либо форму, либо другие данные, переданные из метода действия, необходимо:

Использовать в шаблоне страницы оператор **if** для проверки наличия данных формы в переданном в шаблон словаре: **{% if form != None %}**. После этого может размещаться верстка формы, завершаемая**{% else %}**, после которого до **{% endif %}** будет размещена вёрстка данных, передаваемых в отсутствие данных формы.

**Задание 1**

Цель задания – освоить технологию создания форм и обработки данных, передаваемых через формы, в веб-проекте Django в среде Visual Studio 2017.

1. Добавить в проект страницу для отзывов пользователей сайта – pool.html.
2. Добавить соответствующий пункт, например, «Обратная связь», для перехода на страницу отзывов, в меню сайта.
3. Сформировать состав формы отзыва пользователя сайта, включающий в себя ряд вопросов к пользователю о его оценках различных сторон сайта и его пожеланиях. Форма должна включать в себя поля ввода, флажки, радиокнопки, список, блок ввода текста.
4. Описать в файле forms.py класс формы, содержащий требуемые поля формы. Настроить валидацию полей формы, соответствующую их назначению.
5. Добавить в файл views.py метод действия, который будет создавать объект формы и принимать данные из формы.
6. В случае успешного приема данных передавать из метода действия на страницу отзыва список принятых данных и выводить на этой странице благодарность за заполнение отзыва и оформленный список принятых от пользователя данных.