**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Факультет Информационных технологий**

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА № \_**10**\_**

**Дисциплина: Backend-разработка**

**Тема:** Создание простого веб-приложения на Django

**Выполнил(а): студент(ка) группы \_\_**221-379**\_\_**

\_\_\_\_\_Кодиров Жамшид Мурод угли\_\_\_\_\_\_

(Фамилия И.О.)

**Проверил: \_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_***

(Фамилия И.О., степень, звание) **(Оценка)**

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва**

**2025**

**Задание**

Цель работы: Получить практические навыки работы с фреймворком Django: установка и настройка, создание проекта и приложения, работа с представлениями, маршрутизацией, обработкой запросов и отправкой ответов.

Задание:

Часть 1: Настройка проекта и создание приложения

Установка Django:

Установите Django на свою локальную машину с помощью pip.

Проверьте установку, выполнив команду django-admin --version.

Создание проекта:

Создайте новый проект Django с именем my\_first\_django\_project.

Перейдите в директорию проекта и запустите сервер разработки командой python manage.py runserver.

Откройте браузер и убедитесь, что сервер работает, перейдя по адресу http://127.0.0.1:8000/.

Создание приложения:

Создайте новое приложение с именем my\_first\_app внутри проекта.

Зарегистрируйте приложение в файле settings.py проекта.

Часть 2: Работа с представлениями и маршрутизацией

Создание простого представления:

В файле views.py приложения my\_first\_app создайте функцию-представление hello\_world, которая возвращает простой HTML-ответ: <h1>Hello, World!</h1>.

Создайте маршрут для этого представления в файле urls.py приложения, используя функцию path.

Проверьте работу представления, перейдя по соответствующему URL в браузере.

Обработка параметров запроса:

Измените представление hello\_world, чтобы оно принимало параметр name из URL и возвращало ответ <h1>Hello, {name}!</h1>.

Создайте маршрут с параметром в файле urls.py, используя функцию path или re\_path.

Проверьте работу представления, перейдя по URL с разными значениями параметра name.

Получение данных из строки запроса:

Измените представление, чтобы оно также принимало параметр age из строки запроса и возвращало ответ <h1>Hello, {name}! You are {age} years old.</h1>.

Проверьте работу представления, перейдя по URL с параметрами в строке запроса.

Переадресация и отправка статусных кодов:

Создайте новое представление redirect\_example, которое перенаправляет пользователя на представление hello\_world с параметрами по умолчанию.

Используйте функцию redirect для перенаправления и HttpResponse для отправки статусного кода 302.

Проверьте работу переадресации, перейдя по соответствующему URL.

Отправка JSON-ответа:

Создайте новое представление json\_example, которое возвращает JSON-ответ с данными о пользователе (например, имя и возраст).

Используйте функцию JsonResponse для отправки JSON-ответа.

Проверьте работу представления, перейдя по соответствующему URL и просмотрев ответ в инструментах разработчика браузера.

Работа с куками:

Измените представление hello\_world, чтобы оно устанавливало куку с именем пользователя.

Создайте новое представление show\_cookies, которое отображает все куки, установленные в браузере.

Проверьте работу представлений, перейдя по соответствующим URL и просмотрев куки в инструментах разработчика браузера.

Часть 3: Вложенные маршруты и функция include

Создание вложенных маршрутов:

Создайте новое приложение nested\_app и зарегистрируйте его в проекте.

В приложении nested\_app создайте несколько представлений и маршрутов.

Используйте функцию include в файле urls.py проекта, чтобы включить маршруты из приложения nested\_app в общий маршрут проекта.

Проверьте работу вложенных маршрутов, перейдя по соответствующим URL.

**Ход работы:**

1. **Установка Django:**

В командной строке выполнили установку Django с помощью pip:

pip install django

Проверили успешность установки командой:

django-admin --version

После этого убедились, что Django установлен и готов к работе (выводится номер установленной версии).

1. **Создание проекта:**

В терминале выполнили команду для создания нового проекта Django:

django-admin startproject my\_first\_django\_project

Перешли в директорию проекта:

cd my\_first\_django\_project

Запустили сервер разработки:

python manage.py runserver

Перешли в браузер и открыли адрес <http://127.0.0.1:8000/>, чтобы убедиться, что сервер работает. Появилась стандартная стартовая страница Django.

1. **Создание приложения:**

В терминале выполнили:

python manage.py startapp my\_first\_app

В открывшемся каталоге приложения появились файлы: views.py, models.py, admin.py, apps.py и другие.

Приложение зарегистрировали в файле settings.py проекта, добавив его в список INSTALLED\_APPS:

INSTALLED\_APPS = [

...

'my\_first\_app',

]

1. **Создание простого представления:**

В файле my\_first\_app/views.py создали функцию:

from django.http import HttpResponse

def hello\_world(request):

return HttpResponse("<h1>Hello, World!</h1>")

Затем в корне приложения создали файл urls.py (если он отсутствует) и добавили маршруты:

from django.urls import path

from . import views

urlpatterns = [

path('hello/', views.hello\_world),

]

В файле my\_first\_django\_project/urls.py подключили маршруты приложения:

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

path('admin/', admin.site.urls),

path('', include('my\_first\_app.urls')),

]

После этого открыли http://127.0.0.1:8000/hello/ и увидели сообщение: Hello, World!.

1. **Обработка параметров URL:**

Модифицировали функцию hello\_world, чтобы она принимала параметр name:

def hello\_world(request, name='World'):

return HttpResponse(f"<h1>Hello, {name}!</h1>")

Обновили маршруты:

path('hello/<str:name>/', views.hello\_world),

Проверили результат по адресу, например: http://127.0.0.1:8000/hello/Alex/ → Hello, Alex!.

1. **Получение данных из строки запроса:**

Изменили hello\_world, добавив чтение параметра age:

def hello\_world(request, name='World'):

age = request.GET.get('age', 'unknown')

return HttpResponse(f"<h1>Hello, {name}! You are {age} years old.</h1>")

Перешли по адресу, например: http://127.0.0.1:8000/hello/Alex/?age=22

1. **Переадресация (редирект):**

Создали представление redirect\_example, которое выполняет редирект:

from django.shortcuts import redirect

def redirect\_example(request):

return redirect('/hello/Guest/?age=25')

Добавили маршрут:

path('redirect/', views.redirect\_example),

Проверили по адресу: http://127.0.0.1:8000/redirect/

1. **Отправка JSON-ответа:**

Создали представление json\_example:

from django.http import JsonResponse

def json\_example(request):

data = {'name': 'Alice', 'age': 30}

return JsonResponse(data)

Добавили маршрут:

path('json/', views.json\_example),

Проверили по адресу: http://127.0.0.1:8000/json/. В браузере отобразился JSON-объект.

1. **Работа с куками:**

Изменили hello\_world, чтобы установить куку username:

def hello\_world(request, name='World'):

age = request.GET.get('age', 'unknown')

response = HttpResponse(f"<h1>Hello, {name}! You are {age} years old.</h1>")

response.set\_cookie('username', name)

return response

Создали новое представление show\_cookies:

def show\_cookies(request):

return HttpResponse(f"<pre>{request.COOKIES}</pre>")

Добавили маршрут:

path('cookies/', views.show\_cookies),

Проверили по адресу: http://127.0.0.1:8000/cookies/ — отобразились установленные куки.

1. **Создание вложенных маршрутов:**

Создали новое приложение:

python manage.py startapp nested\_app

Добавили его в INSTALLED\_APPS:

'nested\_app',

В nested\_app/views.py создали два представления:

def nested\_home(request):

return HttpResponse("<h2>This is nested app home.</h2>")

def nested\_about(request):

return HttpResponse("<h2>About nested app.</h2>")

В nested\_app/urls.py:

from django.urls import path

from . import views

urlpatterns = [

path('', views.nested\_home),

path('about/', views.nested\_about),

]

В my\_first\_django\_project/urls.py добавили:

path('nested/', include('nested\_app.urls')),

Проверили по адресам:

* http://127.0.0.1:8000/nested/
* <http://127.0.0.1:8000/nested/about/>

Теперь можно переходить по маршрутам:

* http://127.0.0.1:8000/hello/Alice/?age=30
* http://127.0.0.1:8000/redirect/
* http://127.0.0.1:8000/json/
* http://127.0.0.1:8000/cookies/
* http://127.0.0.1:8000/nested/
* http://127.0.0.1:8000/nested/about/

**Вывод:**

В ходе лабораторной работы был пошагово реализован полноценный Django-проект с двумя приложениями. Мы изучили:

* установку Django и запуск проекта;
* создание приложений и их регистрацию;
* создание различных представлений с поддержкой URL-параметров и строки запроса;
* перенаправления пользователей с использованием redirect;
* отправку JSON-ответов через JsonResponse;
* работу с куками: установка и чтение;
* организацию маршрутов с использованием функции include и реализацию вложенной маршрутизации.

Результатом работы стал функциональный Django-проект, содержащий набор представлений и маршрутов, демонстрирующих базовые возможности фреймворка. Все части задания были выполнены успешно, представления работают корректно, маршруты обрабатываются согласно логике, заданной в коде.

Ссылка на Github: <https://github.com/QodirovJM/BackendPython>