Лекція 11. Створення веб застосунку за допомогою фреймворку Django

Лектор – доцент, кандидат фізико-математичних наук
Кириченко Віктор Вікторович

ПЛАН ЛЕКЦІЇ

1. Встановлення та основні поняття Django. Створення та структура проекту.
2. Створення застосунку Django. Відстежування URL адрес. Вивід тексту на сайт.
3. Розробка моделей. Проведення міграцій. Панель адміністратора. Отримання та вивід даних із БД.

Встановлення та основні поняття Django. Створення та структура проекту.

Встановлення Django:
Для встановлення Django, скористайтеся командою:

```
pip install django
```

Перевірка версії:

```
django-admin --version
```

Щоб створити новий проект Django, використовується команда:

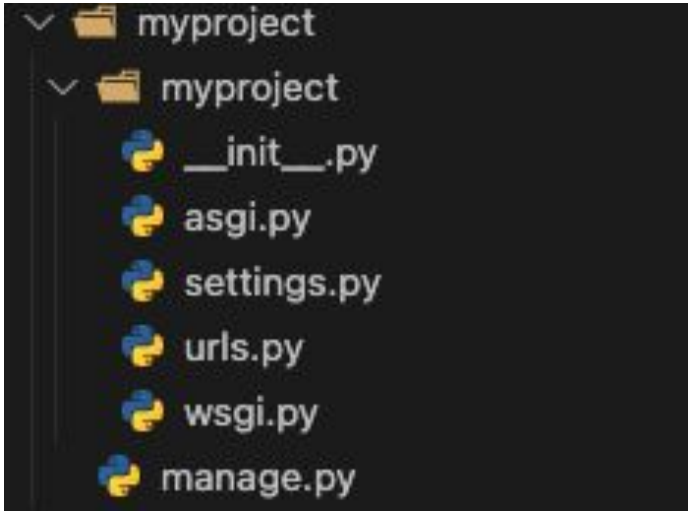
```
django-admin startproject myproject
```

Встановлення та основні поняття Django. Створення та структура проекту.

Структура файлів проекту, включаючи:

manage.py — утиліта для керування проектом.

Директорії проекту з файлами налаштувань (наприклад, settings.py, urls.py, wsgi.py).



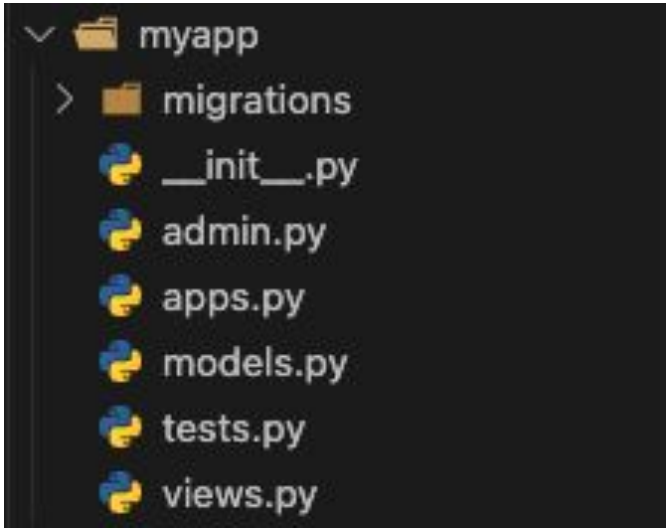
Встановлення та основні поняття Django. Створення та структура проекту.

Для створення застосунку в рамках проекту використовується команда:

```
python manage.py startapp myapp
```

Ця команда створює основну структуру для застосунку:

```
myapp/
    __init__.py
    admin.py
    apps.py
    models.py
    tests.py
    views.py
```



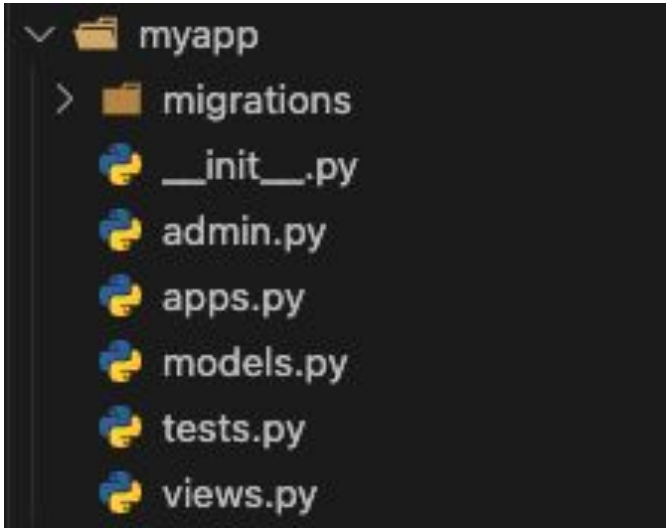
Встановлення та основні поняття Django. Створення та структура проекту.

Для створення застосунку в рамках проекту використовується команда:

```
python manage.py startapp myapp
```

Ця команда створює основну структуру для застосунку:

```
myapp/
    __init__.py
    admin.py
    apps.py
    models.py
    tests.py
    views.py
```



Встановлення та основні поняття Django. Створення та структура проекту.

Після створення нового проекту, перед запуском потрібно виконати міграцію БД:

```
python manage.py migrate
```

Встановлення та основні поняття Django. Створення та структура проекту.

Для запуску застосунку використовується команда:

```
python manage.py runserver
```

Створення застосунку Django. Відстежування URL адрес. Вивід тексту на сайт.

В myapp/views.py додамо нове відображення з назвою home

```
from django.http import HttpResponse

from django.shortcuts import render


# Create your views here.

def home(request):

    return HttpResponse("Hello, Django!")
```

Створення застосунку Django. Відстежування URL адрес. Вивід тексту на сайт.

Налаштуємо раутинг для стовренного додатку.

В myapp потрібно створити файл urls.py

myapp/urls.py

```
from django.urls import path

from . import views


urlpatterns = [

    path('', views.home, name='home'),

]
```

Створення застосунку Django. Відстежування URL адрес. Вивід тексту на сайт.

В myproject у файл settings.py додамо myapp до INSTALLED_APPS

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'myapp',  # Add your app here
]
```

Створення застосунку Django. Відстежування URL адрес. Вивід тексту на сайт.

В myproject у файл urls.py підключаємо urls з myapp

myproject/urls.py

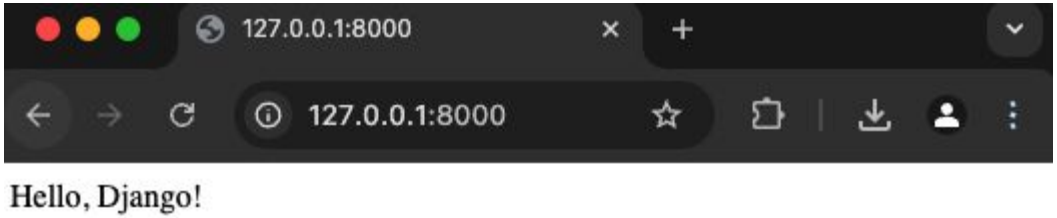
```
from django.contrib import admin
from django.urls import include, path


urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('myapp.urls'))
]
```

Створення застосунку Django. Відстежування URL адрес. Вивід тексту на сайт.

Для запуску застосунку використовується команда:

```
python manage.py runserver
```

Тепер у браузері відображається відображення home.



Створення застосунку Django. Відстежування URL адрес. Вивід тексту на сайт.

Для відображення HTML сторінки використовується templates та функція render.

Потрібно створити папку myapp/templates і файл myapp/templates/home.html з довільним HTML кодом.

```
<h1>
  Hello, world!
</h1>


<style>
  h1 {
    color: blue;
  }
</style>
```

Створення застосунку Django. Відстежування URL адрес. Вивід тексту на сайт.

Для відображення HTML сторінки використовується templates та функція render.

В myapp/views.py замінити HttpResponse на функцію render

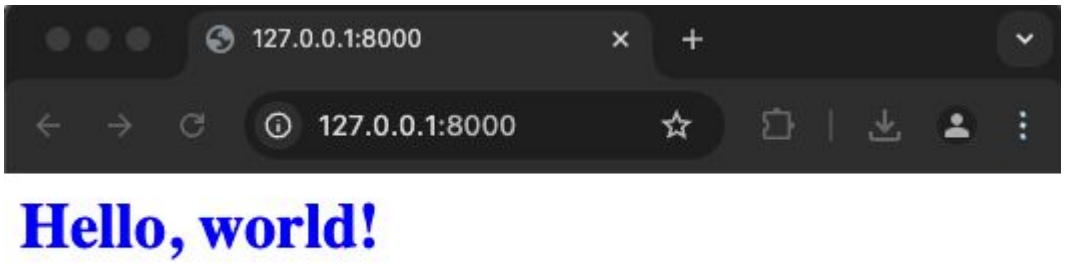
```
from django.http import HttpResponse
from django.shortcuts import render

# Create your views here.
def home(request):
    return render(request, 'home.html')
```

Створення застосунку Django. Відстежування URL адрес. Вивід тексту на сайт.

Для запуску застосунку використовується команда:

```
python manage.py runserver
```

Тепер у браузері відображається відображення home.



Розробка моделей. Проведення міграцій. Панель адміністратора. Отримання та вивід даних із БД.

Міграція баз даних (БД) - це процес переміщення даних та структури бази даних з одного середовища або системи у інше. Цей процес може включати в себе різні аспекти, такі як:

- Переміщення даних
- Переміщення схеми
- Актуалізація даних
- Тестування та верифікація

Розробка моделей. Проведення міграцій. Панель адміністратора. Отримання та вивід даних із БД.

У Django, моделі відповідають за структуру бази даних. Вони визначають таблиці та зв'язки між ними.

```
from django.db import models


class Post(models.Model):
    title = models.CharField(max_length=100)
    content = models.TextField()

    def __str__(self):
        return self.title
```

Розробка моделей. Проведення міграцій. Панель адміністратора. Отримання та вивід даних із БД.

Для отримання даних з бази даних у Django використовується ORM (Object-Relational Mapping).

```
from .models import Post

def home(request):
    posts = Post.objects.all()
    return render(request, 'home.html', {'posts': posts})
```

У шаблоні home.html можна вивести дані:

```
<ul>
    {% for post in posts %}
        <li>{{ post.title }}</li>
    {% endfor %}
</ul>
```

Розробка моделей. Проведення міграцій. Панель адміністратора. Отримання та вивід даних із БД.

Міграції використовуються для створення або оновлення таблиць в базі даних, коли змінюються моделі.

Для створення міграцій:

```
python manage.py makemigrations
```

Для застосування міграцій:

```
python manage.py migrate
```

ДЯКУЮ ЗА УВАГУ