

Привет! Давай разберём, как создать простое приложение на Django шаг за шагом.

## 1. Установка Django

Bash

# Создаем виртуальное окружение

```
python -m venv myenv
```

```
source myenv/bin/activate # Linux/Mac
```

# или

```
myenv\Scripts\activate # Windows
```

# Устанавливаем Django

```
pip install django
```

## 2. Создание проекта

Bash

```
django-admin startproject myproject
```

```
cd myproject
```

## 3. Создание приложения

Bash

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

## 4. Структура проекта

копировать

```
myproject/
```

```
├─ manage.py
```

```
├─ myproject/
```

```
| ├── __init__.py
| ├── settings.py
| ├── urls.py
| └── wsgi.py
└── myapp/
    ├── __init__.py
    ├── admin.py
    ├── apps.py
    ├── models.py
    ├── tests.py
    ├── views.py
    └── migrations/
```

## 5. Настройка приложения

myproject/settings.py:

Python

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'myapp', # добавляем наше приложение
]
```

## 6. Создание модели

myapp/models.py:

Python

```
from django.db import models
```

```
class Article(models.Model):
```

```
    title = models.CharField(max_length=200)
```

```
    content = models.TextField()
```

```
    created_at = models.DateTimeField(auto_now_add=True)
```

```
    updated_at = models.DateTimeField(auto_now=True)
```

```
    def __str__(self):
```

```
        return self.title
```

## 7. Создание представлений (views)

myapp/views.py:

Python

```
from django.shortcuts import render, get_object_or_404
```

```
from django.http import HttpResponse
```

```
from .models import Article
```

```
def home(request):
```

```
    articles = Article.objects.all()
```

```
    return render(request, 'myapp/home.html', {'articles': articles})
```

```
def article_detail(request, article_id):
```

```
article = get_object_or_404(Article, id=article_id)

return render(request, 'myapp/article_detail.html', {'article': article})
```

## 8. Настройка URL-маршрутов

myproject/urls.py:

Python

```
from django.contrib import admin
```

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

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

myapp/urls.py (создайте этот файл):

Python

```
from django.urls import path
```

```
from . import views
```

```
urlpatterns = [
    path('', views.home, name='home'),
    path('article/<int:article_id>/', views.article_detail, name='article_detail'),
]
```

## 9. Создание шаблонов

Создайте папку myapp/templates/myapp/

myapp/templates/myapp/home.html:

Markup

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
    <title>Мой блог</title>
```

```
</head>
```

```
<body>
```

```
    <h1>Статьи</h1>
```

```
    {% for article in articles %}
```

```
        <div>
```

```
            <h2><a href="{% url 'article_detail' article.id %}">{{ article.title }}</a></h2>
```

```
            <p>{{ article.created_at }}</p>
```

```
        </div>
```

```
    {% empty %}
```

```
        <p>Статьи пока не добавлены</p>
```

```
    {% endfor %}
```

```
</body>
```

```
</html>
```

myapp/templates/myapp/article\_detail.html:

Markup

```
<!DOCTYPE html>

<html>

<head>

    <title>{{ article.title }}</title>

</head>

<body>

    <h1>{{ article.title }}</h1>

    <p><em>{{ article.created_at }}</em></p>

    <div>{{ article.content }}</div>

    <a href="{% url 'home' %}">Назад к списку</a>

</body>

</html>
```

## 10. Настройка административной панели

myapp/admin.py:

Python

```
from django.contrib import admin

from .models import Article

@admin.register(Article)

class ArticleAdmin(admin.ModelAdmin):

    list_display = ['title', 'created_at', 'updated_at']

    search_fields = ['title', 'content']
```

## 11. Миграции и запуск

Bash

```
# Создаем миграции
```

```
python manage.py makemigrations
```

```
# Применяем миграции
```

```
python manage.py migrate
```

```
# Создаем суперпользователя для админки
```

```
python manage.py createsuperuser
```

```
# Запускаем сервер
```

```
python manage.py runserver
```

## 12. Проверка работы

- Откройте <http://127.0.0.1:8000/> - главная страница

- Откройте <http://127.0.0.1:8000/admin/> - админка

## Дополнительные возможности

### Формы

```
myapp/forms.py:
```

```
Python
```

```
from django import forms
```

```
from .models import Article
```

```
class ArticleForm(forms.ModelForm):
```

```
    class Meta:
```

```
        model = Article
```

```
        fields = ['title', 'content']
```

### Статические файлы

Создайте папку myapp/static/myapp/ и добавьте CSS, JS файлы.

Полезные команды

Bash

# Создать приложение

```
python manage.py startapp appname
```

# Запустить сервер

```
python manage.py runserver
```

# Создать миграции

```
python manage.py makemigrations
```

# Применить миграции

```
python manage.py migrate
```

# Создать суперпользователя

```
python manage.py createsuperuser
```

Это базовая структура Django приложения. По мере изучения вы сможете добавлять аутентификацию, REST API, кеширование и другие функции!