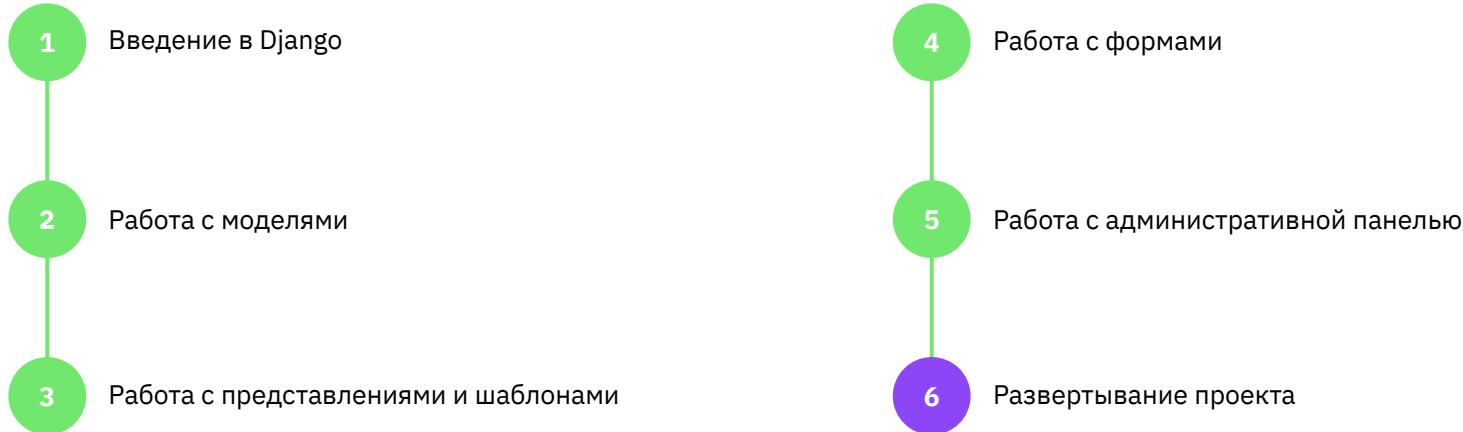


# Развертывание проекта

Урок 6



# План курса





# Содержание урока





## Что будет на уроке сегодня

- 📌 Узнаем о профилировании Django
- 📌 Разберёмся в развертывании проекта на сервере



# Сторонние пакеты для Django





## Дополнительные пакеты

В комплекте Django нет встроенного инструмента для профилирования кода. Но фреймворк позволяет подключать пакеты, созданные другими разработчиками.

Традиционно имена таких пакетов начинаются с приставки django- и устанавливаются командой  
`pip install django-addon-name`





# Профилирование и оптимизация в Django





## Описание и возможности Django Debug Toolbar

**Django Debug Toolbar** – это инструмент для отладки Django-приложений, который предоставляет дополнительную информацию о запросах и ответах, используя различные панели, которые можно настроить.

-  *Панель запросов:* отображает время выполнения каждого запроса к базе данных, а также общее количество запросов.
-  *Панель шаблонов:* показывает, какие шаблоны были использованы для генерации страницы и сколько времени заняло их выполнение.
-  *Панель SQL:* отображает все SQL-запросы, выполненные приложением, включая параметры запросов.
-  *Панель кэша:* показывает, какие запросы были сохранены в кэше и сколько времени они там находились.
-  *Панель профилирования:* предоставляет информацию о времени выполнения каждой функции приложения.
-  *Панель HTTP-заголовков:* отображает HTTP-заголовки запросов и ответов.
-  *Панель среды выполнения:* показывает информацию о системе, на которой работает приложение.



# Установка и настройка DjDT



## Установка

```
pip install  
django-debug-toolbar
```



## Проверка настроек

- INSTALLED\_APPS
- TEMPLATES
- STATIC\_URL



## Дополнительные настройки

- INTERNAL\_IPS
- INSTALLED\_APPS
- MIDDLEWARE



## Прописываем маршрут для подключения DjDT к проекту

```
urlpatterns = [  
    ...  
    path('__debug__/', include("debug_toolbar.urls")),  
]
```



## Оптимизация проекта

Рассмотрим как DjDT помогает оценивать код и оптимизировать его. Воспользуемся таблицей Продукты, которую создали на прошлой лекции. Выведем пользователю общее количество всех продуктов, т.е. сумму поля quantity для всех записей.

Попробуем посчитать сумму разными способами:

- в базе данных
- в представлении
- в модели через вызов в шаблоне



## Этапы разработки

**Перечислим этапы для создания трёх вариантов решения задачи:**

 Готовим данные в базе данных

-  Создаём представления
- Представление для суммирования в БД
  - Представление для суммирования в представлении
  - Представление для суммирования в модели из шаблона

 Создаём свойство модели

 Создаём универсальный шаблон

 Настраиваем маршруты

 Сравниваем результаты в DjDT



# Развертывание проекта на сервере





## Регистрация на платформе

В качестве площадки будем использовать <https://www.pythonanywhere.com/>

- Для регистрации нажмите кнопку  
Start running Python online in less than a minute!
- На открывшейся страницы выбирайте  
Create a Beginner account
- Заполняем форму регистрации
- Не забудьте подтвердить электронную почту, которую вы ввели при регистрации.

Бесплатный тариф создаст для вашего Django проекта адрес в интернете вида  
username.pythonanywhere.com. Введи в качестве username при регистрации то имя,  
которое хотите видеть в адресной строке браузера



# Подготовка проекта к развертыванию

Подготовим код, который мы писали в рамках лекций к развертыванию на сервере. Внесём правки:

## Настройки безопасности

```
DEBUG = False
...
SESSION_COOKIE_SECURE = True
CSRF_COOKIE_SECURE = True
...
import os
...
SECRET_KEY =
os.getenv('SECRET_KEY')
```

## Настройки доступа

```
ALLOWED_HOSTS = [
    '127.0.0.1',
    ...
    'username.pythonanywhere.com',
]
...
STATIC_ROOT = BASE_DIR / 'static/'
```



## Настройка подключения к базе данных

Откроем страницу Базы данных на сайте pythonanywhere.

Бесплатно создаём БД MySQL и после инициализации открываем консоль БД.

Вводим команду смены кодировки с латиницы по умолчанию, на UTF-8:

```
ALTER DATABASE username$default CHARACTER SET utf8 COLLATE utf8_general_ci;
```

```
...
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        ...
    }
}
```



## Формируем список пакетов

В терминале где мы обычно запускали сервер Django введём команду  
pip freeze > requirements.txt

Запишем в requirements.txt ещё пару строк в конец файла

```
mysqlclient
python-dotenv
```

Отключаем DjDT

```
urlpatterns = [
    ...
    # path('__debug__/', include("debug_toolbar.urls")),
    ...
]
```



## Репозиторий GitHub

Скорее всего вы уже создали репозиторий на `github` с вашим проектом.

Добавьте в него финальные изменения проекта, чтобы клонировать в `pythonanywhere` командой

```
git clone https://github.com/myusername/myproject.git
```



# Шпаргалка по Git

Список команд и действий для тех, кто не использовал GitHub

- 💡 git init
- 💡 .gitignore
  - /media/
  - /static/
  - \*.sqlite3
  - \*.log
  - \*.pyc
- 💡 git add \*
- 💡 git commit -m "Initial commit"
- 💡 Создаём пустой репозиторий на сайте <https://github.com/>
- 💡 git remote add origin <https://github.com/username/myproject.git>
- 💡 git push -u origin master
- 💡 git clone <https://github.com/myusername/myproject.git>



## Настройка проекта на сервере

После завершения клонирования остаёмся в консоли, запускаем команду на создание виртуального окружения и добавления пакетов.

По завершению работы в консоли создаём веб приложение с ручной конфигурацией

```
mkvirtualenv --python=/usr/bin/python3.10 virtualenv
```

```
cd myproject  
pip install -r requirements.txt
```



## Настройки веб приложения



**Виртуальное окружение и  
рабочие каталоги**

Заполняем поля на сайте



**wsgi файл облачного сервера**

Правим wsgi файл в редакторе



**Сохраняем “секреты” для  
приложения и для консоли**

- Генерируем секреты
- Сохраняем в .env
- Подключаем .env в консоли



## Финальные команды

Завершаем настройку веб-приложения в облаке.

```
python manage.py migrate
```

```
python manage.py collectstatic
```

```
python manage.py createsuperuser
```



Помним о обязательной перезагрузке  
для применения изменений



# Итоги занятия





## На этой лекции мы

- 📌 Узнали о профилировании Django
- 📌 Разобрались в развертывании проекта на сервер



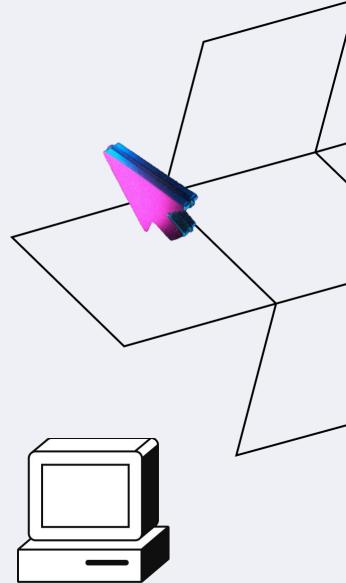
## Задание



1. Для закрепления материалов лекции попробуйте самостоятельно набрать и запустить демонстрируемые примеры.



Спасибо за внимание





Финальная  
лекция курса  
пройдена



# Фреймворк Django