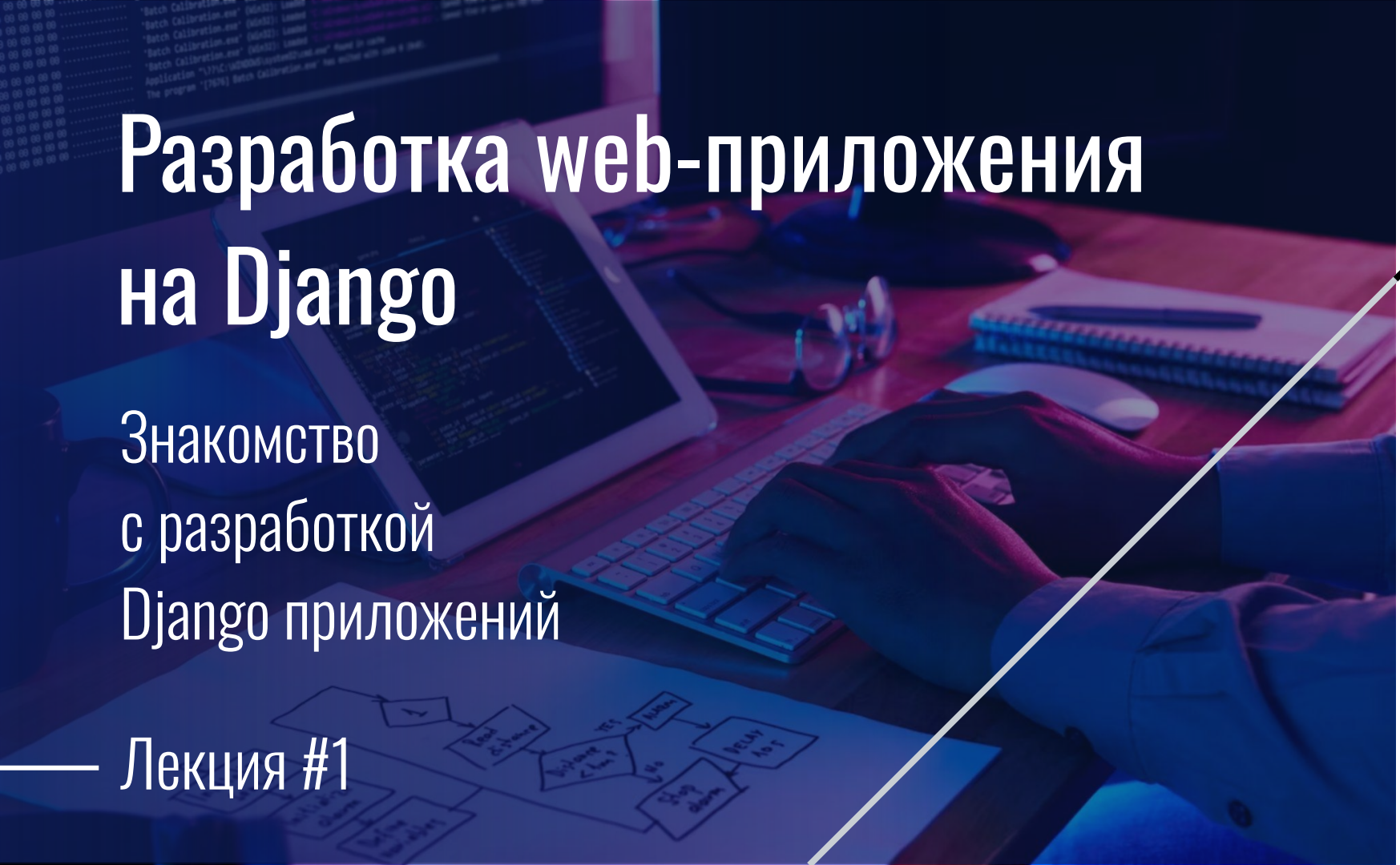


Разработка web-приложения на Django

Знакомство с разработкой Django приложений

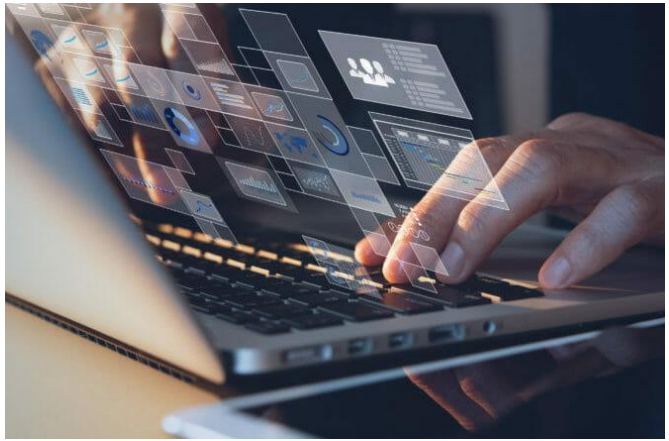
Лекция #1

UNIVERSITY OF ARTIFICIAL INTELLIGENCE

Структура лекции

1. Введение в Django
2. Работа с VSCode
3. Структура папок Django
4. MVT паттерн
5. Путь от запроса пользователя до появления сайта в браузере
6. Generic Views
7. Практическая часть

Django: фреймворк для веб-приложений на Python

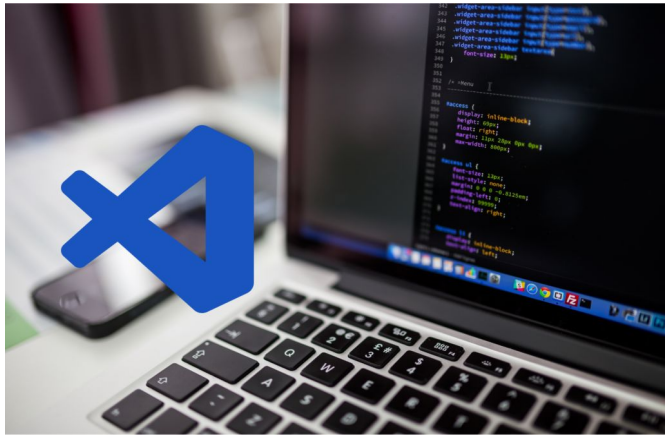


- Инструменты для быстрой разработки безопасных и масштабируемых приложений
- Удобная работа с базами данных, URL-ами, обработкой запросов
- Множество встроенных функций, упрощающих разработку

python™

django

Visual Studio Code (VSCode) для Python и Django



VSCode — популярный редактор кода с широкими возможностями для работы с Python и Django, включая интеграцию с Git и плагины для улучшения функциональности.

Рекомендуемые расширения:

- Python
- Django Snippets
- Django Template
- Pylance

Структура папок Django

- **projectname**/: Главная папка проекта Django. Содержит основные настройки проекта и маршрутизацию URL-ов.
- **appname**/: Папка каждого Django приложения в проекте. Содержит views, models, templates и другие файлы.
- **manage.py**: Управляющий скрипт Django для различных задач.
- **settings.py**: Основные настройки проекта Django, такие как база данных и статические файлы.
- **urls.py**: Определяет URL-адреса приложения и их обработчики.

Разделение логики на приложения в Django

Каждое приложение в Django должно отвечать за конкретную функциональность.

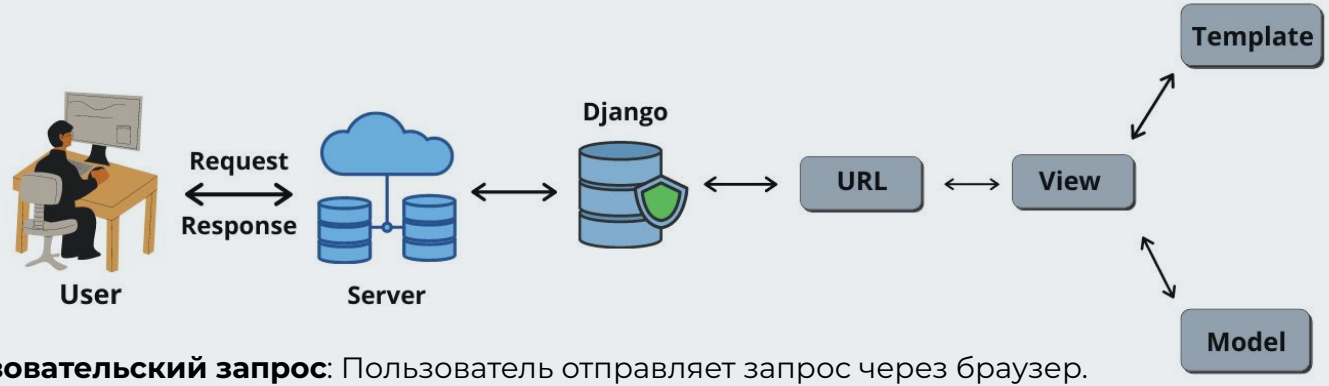
- Улучшает масштабируемость и повторное использование кода
- Повышает читаемость и облегчает совместную разработку
- Каждое приложение отвечает за определенную функциональность
- Обеспечивает более четкую структуру проекта и эффективное управление кодом
- Делает проект более гибким, модульным и легким в поддержке

MVT паттерн

- Model: Представляет данные и правила доступа к ним.
- View: Обрабатывает запросы, взаимодействует с данными и формирует ответы для отображения.
- Template: Отображает данные и составляет пользовательский интерфейс.

MVT в Django позволяет эффективно разделять и организовывать функциональность для более простой разработки, тестирования и поддержки веб-приложений.

Путь от запроса до появления сайта в браузере



1. **Пользовательский запрос**: Пользователь отправляет запрос через браузер.
2. **Обработка в Django**: Запрос обрабатывается через URL-маршрутизацию до представления (view) Django.
3. **Логика и данные**: Представление взаимодействует с данными, обрабатывает логику и генерирует контент для ответа.
4. **Отправка ответа**: Сформированный ответ отправляется на сервер и возвращается пользователю.
5. **Обработка браузером**: Браузер получает ответ, парсит HTML и загружает связанные файлы (css, js, картинки и др. статика).
6. **Отображение страницы**: Полученная страница отображается на экране браузера.

Generic Views

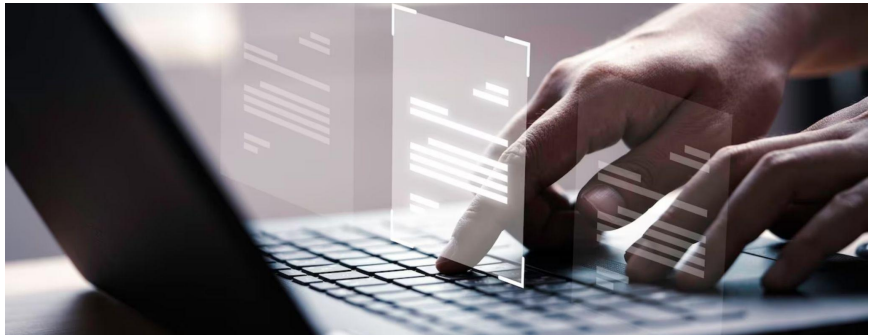
Предварительно созданные представления в Django, упрощают разработку веб-приложений.

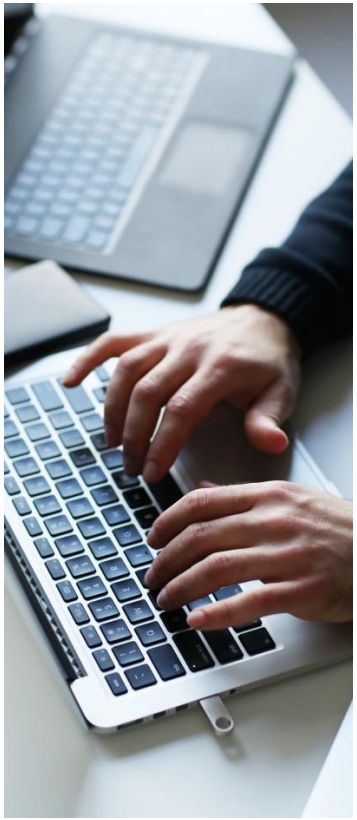
Generic Views:

- Упрощенная разработка: Быстрая создание CRUD операций без длинного кода.
- Гибкость и масштабируемость: Быстрая реализация функций обработки запросов.
- Сокращение кода: Уменьшение повторяющегося кода и улучшение его читаемости.

Стандартный Подход:

- Ручное создание представлений: Необходимость кодирования для каждой операции CRUD.
- Больше кода: Повторное создание кода для различных операций.
- Увеличенные усилия: Больше времени на разработку и отладку.





Практическая часть

1. Создание проекта и необходимых приложений

2. Настройка моделей для каталога товаров:
   - Создадим модель Product с полями, такими как название, описание, цена и изображение.
   - Определим связи между моделями, если необходимо.

3. Создание шаблонов для отображения списка товаров и страницы товара:
   - Создадим шаблон для списка товаров (products_list.html) с использованием шаблонного синтаксиса Django для отображения списка товаров.
   - Создадим шаблон для страницы товара (product_detail.html) с детальной информацией о товаре.

4. Использование Generic Views для реализации логики просмотра товаров:
   - Используем Generic Views, например, ListView для отображения списка товаров.
   - Свяжем Generic Views с нашими созданными моделями Product.
   - Добавим логику для отображения страницы товара с помощью Generic Views, например, DetailView.