**Тема занятие № 58-59: Практическая работа: Лист задач**

1. Формулировка технического задания

Джанго это мощный фреймворк для создания веб-приложений. Изначально Django был создан для того, чтобы быстро создавать, например, новостные сайты (или другие сайты, который нужно создавать максимально быстро). И после нативного PHP не покидает ощущение, что ты едешь на очень быстрой машине разработки. Чтобы посмотреть все его возможности для быстрой разработки, мы с вами попробуем создать простое Todo — приложение.

Начнем с формулировки краткого т.з. У нас будет небольшое веб-приложение с версткой на Bulma. У нас (пока) нет авторизации, и пользователь может создавать, редактировать и удалять либо категорию дел, либо карточку todo, которая связана с какой-либо категорией, которую создал пользователь.

Todo карточку или категорию можно удалить, поставив галочку на checkbox и нажав кнопку удалить.

1. Основные концепции Django

Немного поговорим о Django. Django реализует архитектурный паттерн MVT (Модель Представление Шаблон), которая немного отличается от знакомого большинству MVC (Модель Представление Контроллер) на котором работает Ruby on Rails и Laravel.

Модель (Model) Модель в Django описывает схему данных в БД. С Django ORM, вы можете самостоятельно описывать поля и любые другие типы данных, и совершать миграции для упрощения разработки.

Представление (View) В представлении в Django вы задаете основную логику и алгоритмы приложения, получаете различные данные из базы данных или манипулируете ими. Представление обычно базируется на функциях request\response. Response представляет из себя обычно HTTP redirect, HTTP error(404), MimeTypes или какой-либо шаблон.

Шаблон Шаблон в Django это простой HTML код со специальным шаблонным языком Django. DTL (Django Template Language) — это язык, с помощью которого вы можете динамически менять содержимое страницы (к примеру, изменять имя пользователя на странице, в зависимости от того, как зовут авторизовавшегося пользователя).

Настройки Файл настроек в Django, в котором находятся все настройки вашего веб-приложения. Он включает в себя секретный ключ, папки с шаблонами, middlewares (которые отвечают, например за то, чтобы ваши приватные альбомы не увидели другие пользователи), подключение к базе данных, и много всего остального.

Url Файл настройки роутинга — примерно то же самое, что и в Angular или Laravel. Это связывает представление с url запросами.

Страница Админа так как Django изначально был спроектирован для быстрого прототипирования и развертывания новостных сайтов, админка включена в комплект по умолчанию.

1. Установка Python

До последнего времени активно поддерживались и развивались две основные ветки Python: 2.7 и 3.x. Тут мы будем использовать версию 3.x, но на самом деле это не так важно. Если вы правда хотите знать разницу между ними, есть специальная вики. С другой стороны, сейчас уже нет никакого смысла использовать Python версии 2.7 — обновление языка остановилось на 2.7.17 (если я правильно понимаю документацию на официальном сайте).

Это означает, что есть смысл переводить проекты написанные на Python 2.7.x на новую ветку, а вот новые писать на 2 версии совсем бессмысленно.

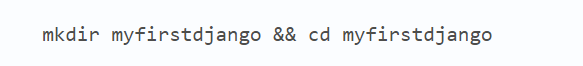
Если вы работаете на Mac или Ubuntu — у вас уже скорее всего установлен Python, однако 2 версии. Python третьей версии придется скачивать отдельно, и вызывать его в командной строке вы сможете через python3.

В любом случае, лучше всего скачать последний релиз на сайте https://www.python.org/

1. Создание виртуального окружения и папки проекта

На самом деле первое приложение на Django вы можете начать разрабатывать и не создавая свое виртуальное окружение, однако навык создания виртуального окружения может пригодится если вы, например, разрабатываете приложение с определенной версией библиотеки и не хотите устанавливать библиотеки глобально и замусоривать ваш system.

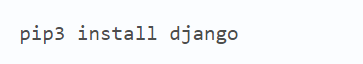
Во, первых, создаем папку, в которой мы будет творить:



Дальше вводим следующие команды для активации venv, где django\_env имя нашего виртуального окружения:



Далее наше виртуальное окружение активировалось. Можем поставить необходимые пакеты. В нашем случае это Django:



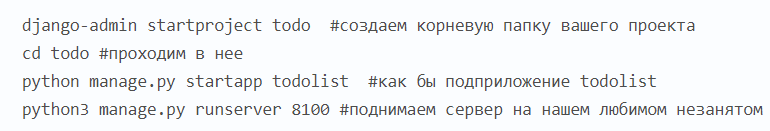
Если вы захотите выключить виртуальное окружение, чтобы вернуться в ваш глобальный python (вернуться в контекст system), введите следующую команду:



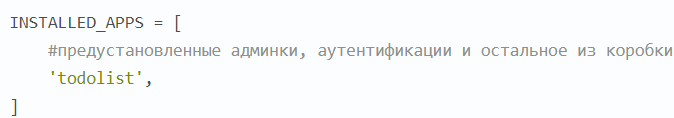
1. Создание и настройка проекта, определение маршрутов

**Создание самого проекта**

Допустим вы выбрали какой-то из способов создания своего виртуального окружения (или даже делаете все глобально, что же, никто не запрещает вам это делать). Теперь проходим в папку проекта и начинаем его создание:



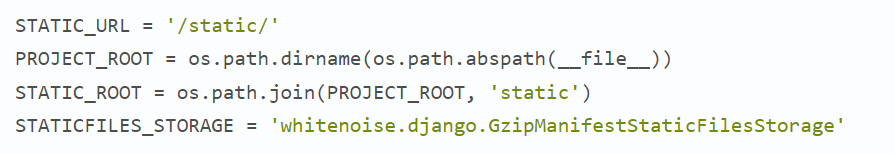
Так, после того как Django открыл стартовую страницу, необходимо проинсталлировать наше приложение todolist в основное приложение. Открываем settings.py и добавляем в уже имеющийся список приложений наш собственный todolist:



Следующим шагом будет связывание приложения с базой данных. Если базы данных — это не то, с чем вы хотите возиться, вам стоит воспользоваться решением по умолчанию — SQlite. Мы воспользуемся PostgreSQL — она популярна и классически связана с Django, кроме того, потом мы можем захотеть увеличить производительность приложения. Инструкций как устанавливать PostgreSQL на все операционные системы достаточно много.

Кроме того, пришлось поставить psycopg2 в виртуальное окружение проекта (без этого драйвера с БД ваше приложение работать не будет).

Дальше нужно настроить работу статики. По-прежнему редактируем файл settings.py, теперь в самом конце добавляем работу со статикой:



Для того, чтобы у вас заработала статика, проверьте что в списке INSTALLED\_APPS находился пакет, отвечающий за статику:

django.contrib.staticfiles, на тот случай, если произойдет ошибка.

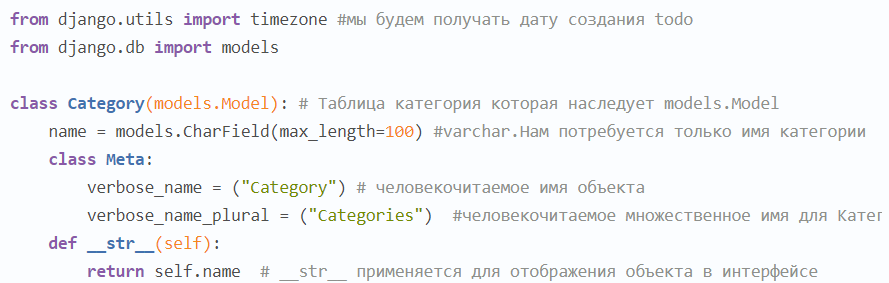
И последнее в подготовительных работах, нам нужно ещё настроить базовую работу url в проекте:



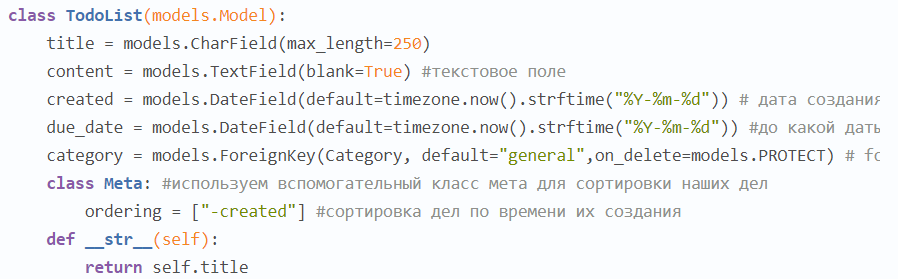
Мы добавили перенаправление, так как хотим, чтобы с дефолтной страницы localhost сразу переходил на подстраницу category (чтобы пользователь не потерялся). Также у нас есть роутинг на две страницы: категорий и дел.

1. Создание моделей

Дальше приступим к созданию модели, которая будет базово взаимодействовать с нашей базой данных. Для создания модели открываем файл models.py в нашем todolist и начинаем творить. Начнем с того, что создадим таблицу категорий:



Отлично! Да, здесь у нас будет только две колонки в таблице Категорий: id и name. Дальше создадим таблицу для наших дел. Думаю, из комментариев все понятно:



После того, как ваша модель будет готова, необходимо создать миграции:

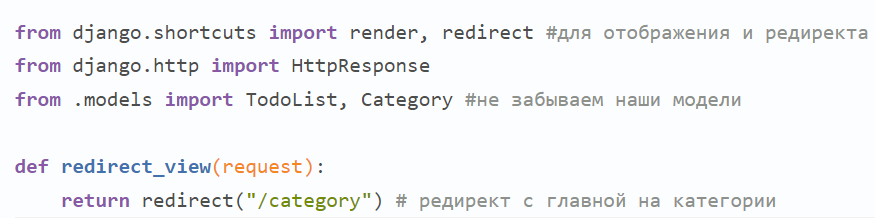


И потом запускаете сами миграции:

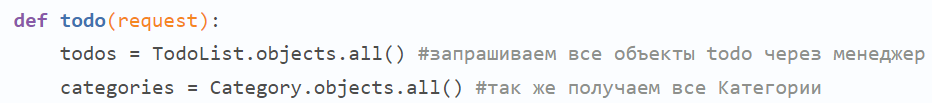


1. Создание контроллеров

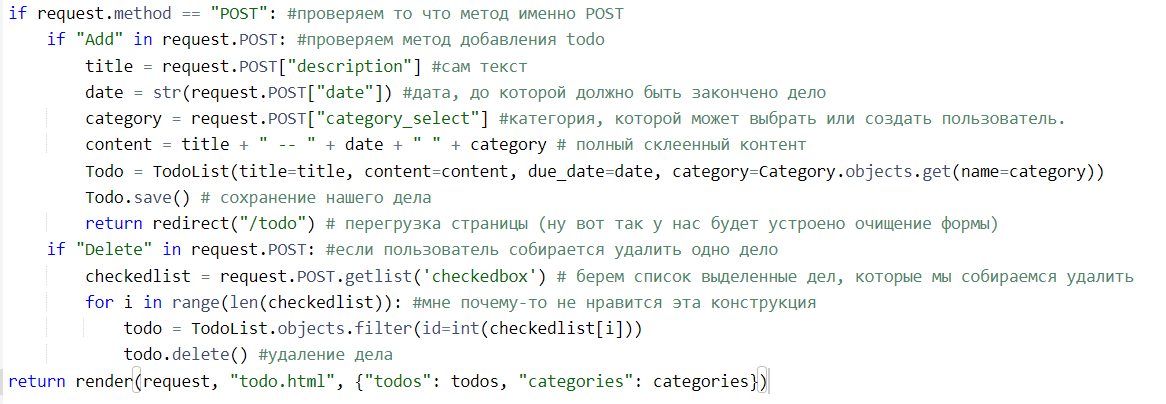
Откроем файл view.py в todolist и отредактируем его. Для начала добавим необходимые импорты и редирект с главной на category:



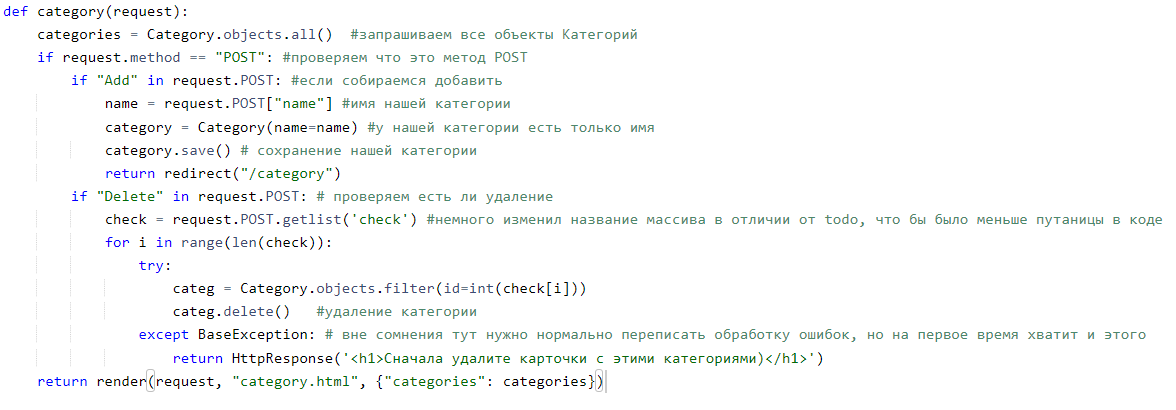
Потом начинаем создание нашего дела. У экземпляра дела будут поля самого текста, даты, до которой должно быть закончено дело, категория дела, и объединенный контент:



После этого добавим функции добавления и удаления дел:



С тудушками все. Дальше можем перейти к странице Категорий. Создаем функцию категорий, в которой у нас тоже будет функция добавления и удаления категории. Принципиально здесь ничего нового не будет, у нас так же здесь будет возможность добавления и удаления:

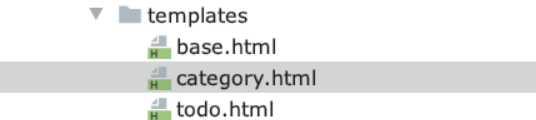


На этом мы заканчиваем с файлом view и можем переходить к шаблонам.

1. Создание шаблонов

Как вы помните, чтобы не писать css лишний раз, я воспользовался bulma.css для упрощения верстки. Т.к. наши страницы категорий и todo буду очень схожи, я создал три файла:

base.html, который будет включать в себя все одинаковое, что у нас есть на страницах, а в category.html, todo.html будут располагаются отличия:

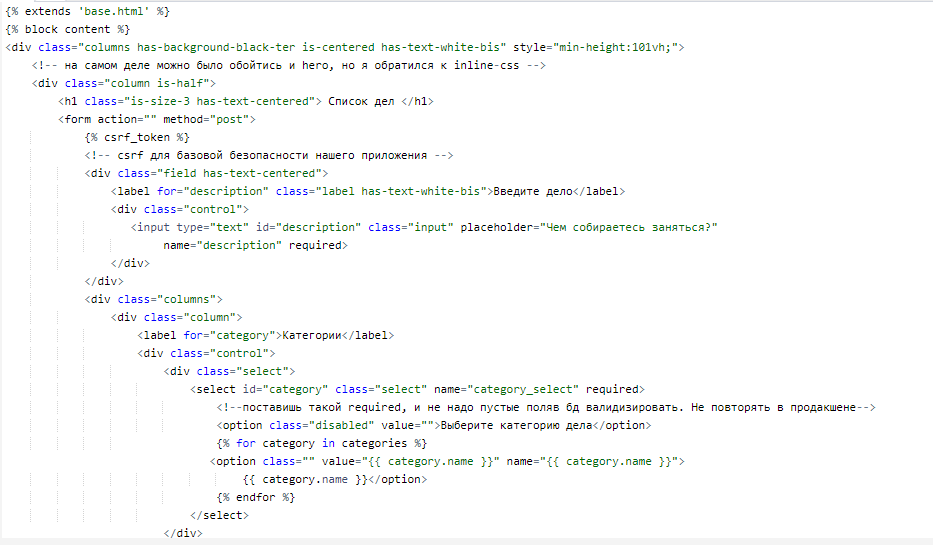


Создаем base.html и редактируем его:

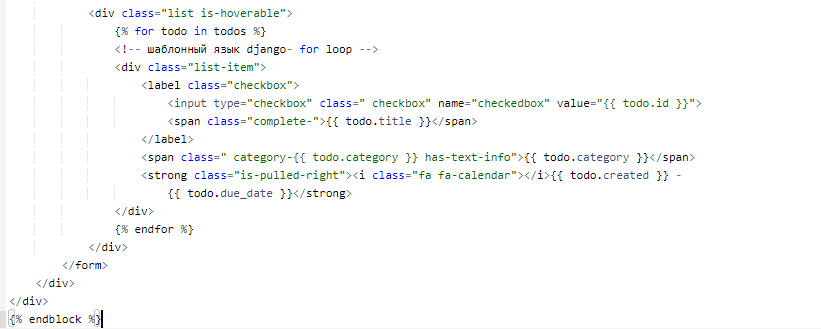


Дальше у нас пойдут страницы todo.html и category.html:

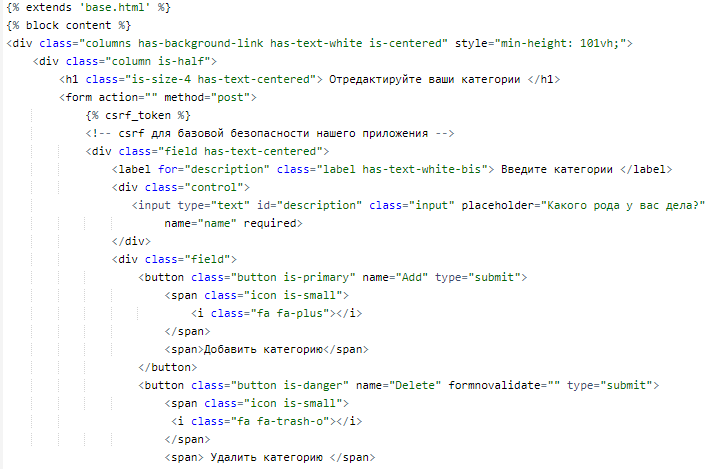
Тудушка:

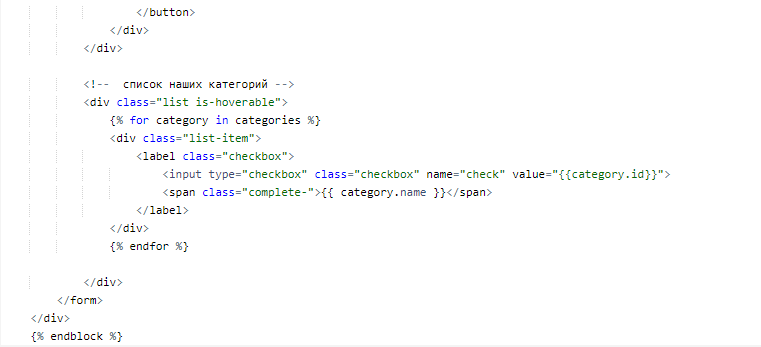






И category.html. В ней у нас не особо много чего меняется, принципиально никак не отличается от todo.html:





На этом все. Возможно, где-то не идеальна верстка, можно дополнить по желанию. А подготавливать приложение к выпуску мы будем в следующем проекте.