**Глава 1 Введение в Django**

**Создаём проект на Django**

Создаём новый репозиторий на github по названию будущего проекта friender

**Virual env и установка пакетов**

Создаём в папке виртуальное окружение: py –m venv имя[.venv]

Не забудьте активировать виртуальное окружение: .venv\Scripts\activate

Устанавливаем django. Pip install Django

**Startproject**

django-admin startproject имя[каталог] .

Создает структуру каталогов проекта Django для данного имени проекта в текущем каталоге или заданном месте назначения. Если указано только имя проекта, и каталог проекта, и пакет проекта будут называться <projectname>, а каталог проекта будет создан в текущем рабочем каталоге. Если указано необязательное место назначения,Django будет использовать этот существующий каталог в качестве каталога проекта и создаст в нем manage.py и пакет проекта.  
  
  
MTV:

Model-temlates-views; шаблон программного обеспечения, похожий по стилю на MVC, но лучше описывающий то, как работает Django. Модель-представление-контроллер; программный шаблон Django в некоторой степени следует MVC.

Функция представления, или сокращенно Вьюха, — это функция python, которая принимает веб-запрос и возвращает веб-ответ. Этот ответ может быть НТМL содержимым веб-страницы, перенаправлением, ошибкой 404, XML - документом или изображением… или что-нибудь, на самом деле. Само представление содержит любую произвольную логику, необходимую для возврата этого ответа. Этот код может жить где угодно, пока он находится на вашем пути python. Других требований нет — никакой «магии», так сказать.

Ради размещения кода где-нибудь принято размещать представления в файле views.py, расположенном в каталоге вашего проекта  
или приложения.

Пример функции представления, которое возвращает текущую дату и время в виде HTML- документа:

***from django.http import HttpResponse***

***import datetime***

***def current\_datetime(request):***

***now = datetime.datetime.now()***

***html = "<html><body>It is now %s.</body></html>" % now***

***return HttpResponse(html)***

Во-первых, мы импортируем класс HttpResponse из модуля django.httpвместе с библиотекой datetime python. Затем мы определяем функцию под названием current\_datetime. Это функция просмотра. Каждая функция представления принимает в качестве первого параметра объект [HttpRequest](https://django.fun/ru/docs/django/4.1/ref/request-response/#django.http.HttpRequest), который обычно называется request.

Обратите внимание, что имя функции представления не имеет значения; его не нужно называть определенным образом, чтобы Django мог его распознать. Мы называем это здесь current\_datetime, потому что это имя четко указывает на то, что он делает.

Представление возвращает объект [HttpResponse](https://django.fun/ru/docs/django/4.1/ref/request-response/#django.http.HttpResponse), содержащий сгенерированный ответ. Каждая функция представления отвечает за возврат объекта [HttpResponse](https://django.fun/ru/docs/django/4.1/ref/request-response/#django.http.HttpResponse). (Есть исключения, но мы вернемся к ним позже.)

**Функция render:**

**render(*request*, *template\_name*, *context=None*, *content\_type=None*, *status=None*, *using=None*)**

Объединяет данный шаблон с заданным словарем контекста возвращает [HttpResponse](https://docs.djangoproject.com/en/4.1/ref/request-response/#django.http.HttpResponse)объект с этим визуализированным текстом. Django не предоставляет функцию быстрого доступа, которая возвращает a, [TemplateResponse](https://docs.djangoproject.com/en/4.1/ref/template-response/#django.template.response.TemplateResponse)потому что конструктор [TemplateResponse](https://docs.djangoproject.com/en/4.1/ref/template-response/#django.template.response.TemplateResponse)предлагает тот же уровень удобства, что и [render()](https://docs.djangoproject.com/en/4.1/topics/http/shortcuts/#django.shortcuts.render).

**Конфигурация шаблонов:**

Шаблон Django — это текстовый документ или строка Python, размеченная с использованием языка шаблонов Django. Некоторые конструкции распознаются и интерпретируются механизмом шаблонов. Основными из них являются переменные и теги. Шаблон отображается с контекстом. Рендеринг заменяет переменные их значениями, которые просматриваются в контексте, и выполняет теги. Все остальное выводится как есть. Механизмы шаблонов настраиваются с помощью параметра TEMPLATES. Это список  
конфигураций, по одной для каждого движка. Значение по умолчанию пусто.

**Маршутизация:**

Для работы с маршрутизацией, вы создаете модуль python , неофициально называемый  
URLconf (конфигурация URL- адреса). Этот модуль представляет собой чистый код python и представляет собой сопоставление между выражениями URL-пути и функциями python (вашими представлениями).  
Это сопоставление может быть настолько коротким или длинным, насколько это необходимо. Он может ссылаться на другие сопоставления. А поскольку это чистый код python , его можно создавать динамически.  
Этот модуль настраивается через settings.py. По умолчанию маршрутизация собрана в urls.py в папке проекта. Пример содержимого urls.py:

***from django.urls import path***

***from . import views***

***urlpatterns = [***

***path('articles/2003/', views.special\_case\_2003),***

***path('articles/<int:year>/', views.year\_archive),***

***path('articles/<int:year>/<int:month>/', views.month\_archive),***

***path('articles/<int:year>/<int:month>/<slug:slug>/', views.article\_detail),***

***]***

**Задание:**

Использую в качестве примера созданный нами ранее проект с виртуальным окружением! Создайте Django приложение

1. Создать 2 маршрута с путями к 2 страницам, главной странице содержащей информацию о правилах сайта знакомств и информации где будут в дальнейшем выводиться все заведения
2. Создать 2 функции представления связанные с путями написанными в 1 пункте.
3. Подключить приложение к проекту в настройках и в файле с путями.