# Практическая работа №11-12

# Введение

Python – отличный инструмент для анализа данных, с его помощью можно выполнять разнообразные операции и задачи, связанные с обработкой и анализом данных.

Django один из наиболее популярных фреймвоков для написания веб-приложений на языке Python, использующий такие шаблоны проектирования как MVC и MVT, он облегчает создание мощных и масштабируемых веб-приложений.

В первой части практической работе вам предстоит написать простой аналитический сервис, который должен будет брать данные из сервисов, написанных в прошлых практиках (на Go, Java).  
Во второй части работы вам предстоит установка Django и настройка venv – виртуальной среды интерпретатора Python, а так же написание простого веб-приложения и выгрузка его в Docker контейнер.

# Теория

Python код может выполняться разными способами. Можно написать файл c расширением .py, который бы содержал некий скриптовый код, далее его можно исполнить командой python, например, мы находимся в некотором каталоге где лежит main.py в котором есть код:

Листинг 1 – код примера

**print("Hello world")**

Далее необходимо запустить код командой:

Листинг 2 – запуск кода на Python

D:\RSCHIR> python main.py

Hello world

PS D:\RSCHIR>

Но такой подход обычно используется простыми программами или скриптами и использует глобальную виртуальную среду интерпретатора Python. В этой среде обычно определены пакеты, получаемые с помощью пакетного менеджера python – pip. pip берет пакеты из собственного удаленного репозитория пакетов [PyPi](https://pypi.org/). Эти пакеты представляют собой zip архивы которые содержат в себе код Python некоторую метаинформацию, определяющую зависимости библиотеки и тому подобные вещи.

Соответственно, пакеты, которые будут стоять глобально, могут быть использованы всеми программами, которые выполняются глобально как было описано выше, но проблема в том, что пакеты постоянно обновляются, выходят новые патчи, исправления и, если устанавливать эти обновления глобально, какие-то из проектов могут перестать корректно запускаться, не только из-за несовместимости версий используемых пакетов в конкретных проектах, но и из-за несовместимой версии самого интерпретатора Python.

Поэтому был придуман механизм изоляции приложений, написанных на языке Python – это создание отдельных виртуальных сред интерпретатора Python venv. Такой подход позволяет использовать только те зависимости, которые нужны проекту. Можно сказать, что это своя папка для каждого проекта на Python. Такой подход является наиболее рекомендуемым.

По большому счету пакеты являются библиотеками для языка Python. А pip это способ доставки этих пакетов, по аналогии с Gradle/Maven для Java/Kotlin или npm для Node.js.

Таким пакетом и является Django. Чтобы его установить необходимо создать виртуальную среду интерпретатора Python. Для начала воспользуемся командной строкой и создадим рабочую папку и создадим в ней виртуальную среду

Листинг 3 – Создание виртуальной среды и ее активация

PS D:\RSCHIR\helloworld> python -m venv .venv

PS D:\RSCHIR\helloworld> ./.venv/Scripts/activate.bat

(.venv) PS D:\RSCHIR\helloworld>

Далее нужно скачать пакет Django. Делается это с помощью pip

Листинг 4 – Установка пакета

(.venv) PS D:\RSCHIR\helloworld> pip install django

Django установлен последней возможной версии. Далее необходимо создать проект Django, делается это с помощью утилиты django-admin, которая входит в поставку пакета django.

Листинг 5 – Создание проекта

(.venv) PS D:\RSCHIR\helloworld> django-admin startproject example

# Описание файловой архитектуры созданного проекта

Файловая архитектура организации джанго (стандартная) приведена на рисунке 1.

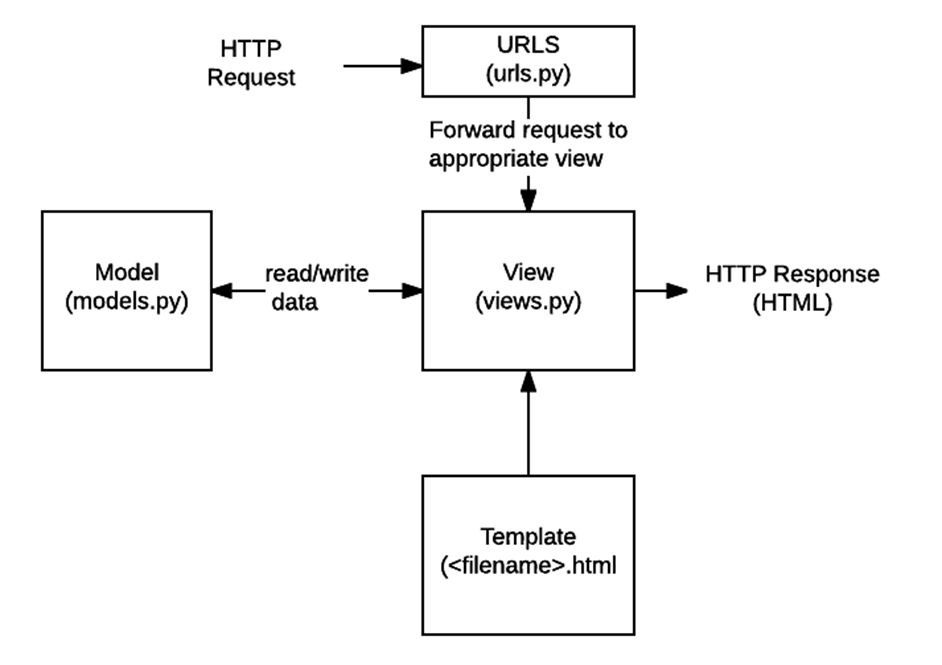


Рисунок 1.  Файловая архитектура организации джанго (стандартная)

Python имеет множество библиотек и фреймворков, созданных специально для анализа данных. NumPy, Pandas, Matplotlib и SciPy предоставляют мощные инструменты для работы с числовыми данными, выполнения операций с массивами, обработки и визуализации данных.

NumPy

NumPy это open-source модуль для python, который предоставляет общие математические и числовые операции в виде пре-скомпилированных, быстрых функций. Они объединяются в высокоуровневые пакеты. Они обеспечивают функционал, который можно сравнить с функционалом MatLab. NumPy (Numeric Python) предоставляет базовые методы для манипуляции с большими массивами и матрицами.

SciPy

SciPy (Scientific Python) расширяет функционал numpy огромной коллекцией полезных алгоритмов, таких как минимизация, преобразование Фурье, регрессия, и другие прикладные математические техники.

Matplotlib

Библиотека на языке программирования Python для визуализации данных двумерной и трёхмерной графикой. Получаемые изображения могут быть использованы в качестве иллюстраций в публикациях. Matplotlib написан и поддерживался в основном Джоном Хантером и распространяется на условиях BSD-подобной лицензии.

**Описание файлов приложения**

**URLs**: Хотя можно обрабатывать запросы с каждого URL-адреса с помощью одной функции, гораздо удобнее писать отдельную функцию для обработки каждого ресурса. URL-mapper используется для перенаправления HTTP-запросов в соответствующее представление на основе URL-адреса запроса. URL-mapper также может извлекать данные из URL-адреса в соответствии с заданным шаблоном и передавать их в соответствующую функцию в виде аргументов.

**View:** Представление (view) - это функция обработчика запросов, которая получает HTTP-запросы и возвращает ответы. View имеет доступ к данным через модели (необходимым для удовлетворения запросов и делегирования ответа в шаблоны).

**Models:** Модели представляют собой объекты Python, которые определяют структуру данных приложения и предоставляют механизмы для управления (добавления, изменения, удаления) и выполнения запросов в базу данных.

**Templates:** Template (шаблон) - это текстовый файл определяющий структуру или разметку страницы (например HTML страницы), с полями для подстановки используемыми для представления актуального содержимого. View может динамически создавать HTML страницы, используя HTML шаблоны и заполняя их данными из модели (model). Шаблон может быть использован для определения структуры файлов любых типов, не обязательно HTML.

Далее вы можете пользоваться любой средой разработки для написания кода приложения.

# Полезные ссылки

1. Оф. документация Django - <https://docs.djangoproject.com/en/4.2/>
2. RESTful на Django - <https://habr.com/ru/articles/722054/>
3. Views в Django - <https://ru.hexlet.io/courses/python-django-basics/lessons/views/theory_unit>
4. Установка Python - <https://devpractice.ru/python-lesson-1-install/>
5. Docker - <https://habr.com/ru/articles/253877/>
6. Docker compose - <https://habr.com/ru/companies/ruvds/articles/450312/>
7. <https://habr.com/ru/articles/352678/> - NumPy
8. <https://pythonworld.ru/novosti-mira-python/scientific-graphics-in-python.html> - Matplotlib
9. <https://django.fun/ru/articles/tutorials/sozdajte-rest-api-za-30-minut-s-pomoshyu-django-rest-framework/> - REST на Django
10. <https://habr.com/ru/articles/729610/> - Posgres, Django, Docker
11. <https://blog.skillfactory.ru/glossary/pandas/> - Pandas

# Задание

* + - 1. Вам предлагается написать простое веб приложение Hello world, которое должно получать имя пользователя и выводить на экран, “Hello, {полученное имя}”. Ваша фамилия обязательно должна фигурировать в пути к приложению (localhost:8080/ivanov). Обязательно обработать случай, когда пользователь не передал имя.

Код обязательно должен быть написан на языке Python c использованием фреймворка Django.

Код обязательно должен иметь отдельную venv Python, и должен быть запущен в контейнере Docker.

* + - 1. Вам предлагается реализовать простой сервис аналитики, которому бы на вход поступали данные из предыдущих практик (Java и Go), на выходе сервис выдает картинку-диаграмму. Картинки обязательно сохранять в БД(Рекомендуемая – Postgres), и при получении данных которые уже были, использовать график, который уже был сгенерирован.

Необходимо реализовать сервис на основе предыдущей работы с использованием REST.

Сервис обязательно должен работать на Docker контейнере, и написан на Python Django с объединением с предыдущими работами. Приветствуется использование схожих библиотек, описанных в введении для создания картинок и анализа данных.

# Контрольные вопросы

1. Из каких частей состоит обычный проект на Django?
2. Что такое Отображения (views) в рамках Django?
3. Что такое модель в рамках Django?
4. Что нужно сделать чтобы развернуть Django приложение внутри Docker?
5. Какие преимущества предоставляет Django для разработки аналитических сервисов?
6. Как можно визуализировать аналитические данные в Django-приложении?
7. Зачем нужна библиотека matplotlib?
8. Какие действия нужно произвести чтобы Django мог работать с Postgres?