Flask - это легкая веб‑платформа Python, которая предоставляет полезные инструменты и функции для создания веб‑приложений на языке Python. Это дает разработчикам гибкость и является доступной средой для новых разработчиков, поскольку вы можете быстро создать веб‑приложение, используя только один файл Python. Flask также является расширяемым и не требует определенной структуры каталогов и не требует сложного шаблонного кода перед началом работы.

Знание Flask позволит вам быстро создавать веб‑приложения на Python. Вы можете воспользоваться библиотеками Python для добавления расширенных функций в свое веб‑приложение, таких как хранение данных в базе данных или проверка веб‑форм.

На этом шаге вы активируете среду Python и установите Flask с помощью установщика пакета pip.

После активации среды программирования установите Flask с помощью команды pip install:

pip install flask

После завершения установки вы увидите список установленных пакетов в последних частях вывода, аналогичный следующему:

...

Installing collected packages: Werkzeug, MarkupSafe, Jinja2, itsdangerous, click, flask

Successfully installed Jinja2-3.0.1 MarkupSafe-2.0.1 Werkzeug-2.0.1 click-8.0.1 flask-2.0.1 itsdangerous-2.0.1

Это означает, что установка Flask также установила несколько других пакетов. Эти пакеты являются зависимостями, которые необходимы Flask для выполнения различных функций.

Вы создали папку проекта, виртуальную среду и установили Flask. Теперь вы можете перейти к настройке простого приложения.

Теперь, когда у вас настроена среда программирования, вы начнете использовать Flask. На этом этапе вы создадите небольшое веб‑приложение Flask внутри файла Python, в котором вы напишете HTML‑код для отображения в браузере.

В каталоге **flask\_app** откройте файл с именем **app.py** для редактирования, используйте IDLE или ваш любимый текстовый редактор напишите следующий код:

**from** flask **import** Flask

app = Flask(\_\_name\_\_)

@app.route('/')

**def** hello():

**return** '<h1>Hello, World!</h1>'

Сохраните и закройте файл.

После создания файла, содержащего приложение Flask, вы запустите его с помощью интерфейса командной строки Flask, чтобы запустить сервер разработки и отобразить в браузере HTML‑код, который вы написали в качестве возвращаемого значения для функции просмотра hello() в предыдущем разделе. шаг.

Первый, находясь в каталоге **flask\_app** с активированной виртуальной средой, сообщите Flask, где найти приложение (**app.py** в вашем случае), используя переменную среды FLASK\_APP с помощью следующей команды (в Windows используйте set вместо export):

export FLASK\_APP=app

Затем укажите, что вы хотите запустить приложение в режиме разработки (чтобы вы могли использовать отладчик для обнаружения ошибок) с помощью переменной среды FLASK\_ENV:

export FLASK\_ENV=development

Наконец, запустите приложение, используя команду flask run:

flask run

После запуска приложения вы увидите примерно это:

\* Serving Flask app "app" (lazy loading)

\* Environment: development

\* Debug mode: on

\* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

\* Restarting **with** stat

\* Debugger **is** active!

\* Debugger PIN: 296-353-699

Предыдущие выходные данные содержат несколько частей информации, например:

* Название приложения, которое вы запускаете (**«app»**).
* Среда, в которой выполняется приложение (**development**).
* **Debug mode** (Режим отладки): **on** означает, что отладчик Flask запущен. Это полезно при разработке, поскольку предоставляет подробные сообщения об ошибках, когда что-то идет не так, что упрощает поиск и устранение неисправностей.
* Приложение работает локально по URL-адресу **http://127.0.0.1:5000/**. **127.0.0.1** — это IP-адрес, который представляет локальный хост вашего компьютера, а: **5000** — это номер порта.

Откройте браузер и введите URL-адрес **http://127.0.0.1:5000**/. Вы увидите текст **Hello, World!** в заголовке <h1> в качестве ответа. Это подтверждает, что ваше приложение успешно работает.

