Python Practice

Размещение сайта на heroku. Краткое резюме

Автор курса

Кременtарь Ксения

Ведущий Python разработчик

Системный архитектор

в компании K-Solutions

После урока обязательно

Повторите этот урок в видео формате на ITVDN.com

Проверьте как Вы усвоили данный материал на TestProvider.com

Размещение сайта на heroku. Краткое резюме.

На этом уроке

1. Узнаем о вариантах деплоймента проектов в production среде.
2. Познакомится с процедурой разворачивания сервисов на платформе Heroku.
3. Развернем проект на Heroku.
4. Подведем итоги курса.

Проект реализован — что же дальше?

Проект Golden-Eye реализован в полном объеме согласно постановке задачи, но он должен быть запущен где-то, кроме вашего собственного стационарного компьютера или ноутбука.

Деплоймент — это процесс развертывания проекта на production сервер. Для этого код проекта должен быть помещен на специальные компьютеры, сервера, которые доступны в интернете 24/7. При этом, на серверах должны быть установлены все необходимые программы — Python интерпретатор и сторонние библиотеки, СУБД и другие сервисы и программные компоненты.

Варианты размещения проектов

Есть несколько вариантов развертывания web проекта, основные это:

- развертывание на сервере
- с использованием платформ PaaS(Platform-as-a-service)

Развертывание на сервере подразумевает, что разработчик имеет доступ к серверу, самостоятельно выбирает ОС, устанавливает все необходимые библиотеки, дополнительные сервисы (СУБД, web сервер) и т.п. и поддерживает все эти компоненты системы в дальнейшем, в ходе эксплуатации проекта.

Можно разворачивать проекты на собственных серверах и самостоятельно обеспечивать их доступность и надежность, можно воспользоваться услугами специальных компаний. Примерами таких компаний являются — DigitalOcean, Linode, Vultr, OVH, Amazon Web Services и т.п.

Платформы PaaS

Платформы PaaS, например, Heroku, PythonAnywhere, Google App Engine и тп избавляют разработчика от настройки сервера, а позволяют ему сосредоточиться только на коде приложения.

Такой подход делает процесс разворачивания web приложений гораздо более удобным и простым для разработчика.

heroku

Основные этапы разворачивания на Heroku

- Завести аккаунт на сайте Heroku.
- Установить CLI, Heroku Command Line Interface, все дальнейшие действия по управлению проектом на Heroku будут осуществляться через Heroku CLI.
- Создать приложение heroku из папки с кодом и git репозиторием приложения.
- Добавить Procfile в проект, для указания Heroku, каким именно образом запускать python приложение (указать тип процесса - web).
- запустить приложение с помощью CLI команды **heroku ps:scale web=1** и проверить в браузере.

Добавление периодической задачи

Для добавления процесса обновления курсов, необходимо добавить в Procfile новый тип процесса — clock и указать команду, которая должна запускаться: **python tasks.py**.
После внесения изменений в Procfile необходимо их закоммитить и обновить на сервере heroku, а после этого запустить новый процесс с помощью команды — **heroku ps:scale clock=1**

После внесенных изменений задача с обновлением курсов будет запускаться с указанной периодичностью.

Полезные команды Heroku CLI

heroku create

Создает приложение на heroku и подготавливает репозиторий на heroku, где будет размещен наш код. Можно указать имя приложения (через пробел) или оно будет сгенерировано случайным образом

heroku ps:scale web=1

Запускает приложение

heroku open

Открывает в браузере сайт с приложением

heroku logs —tail

Отображает последние логи с сервера

heroku run

Запуск командной строки для выполнения команд на сервере heroku (напримре, python скрипт по инициализации БД и т.п.)

Добавление логирования в консоль

Для добавления логирования в консоль, для дальнейшего просмотра логов с помощью heroku CLI необходимо просто добавить новый обработчик типа logging.StreamHandler и добавить его в список хендлеров для каждого логгера в нашем конфигурационном файле.

```
'handlers': {
    'console': {
        'class': 'logging.StreamHandler',
        'formatter': 'default',
    },
    'file': {
        'class': 'logging.FileHandler',
        'formatter': 'default',
        'filename': 'new.log',
    },
},
```

```
'loggers': {
    'GoldenEye': {
        'handlers': ['file', 'console'],
        'level': logging.DEBUG
    },
    'Api': {
        'handlers': ['file', 'console'],
        'level': logging.DEBUG
    },
    'Tasks': {
        'handlers': ['file', 'console'],
        'level': logging.DEBUG
    },
},
```

Просмотр логов приложения

krementar@MacBook-Pro-Ksenia:~/projects/golden_eye$ heroku logs --tail

```
2019-02-24T20:50:36.574548+00:00 app[web.1]: 2019-02-24 20:50:36,574 [INFO] - flask.app: Started ViewAllRates
2019-02-24T20:50:36.595569+00:00 app[web.1]: 10.63.102.120 - - [24/Feb/2019:20:50:36 +0000] "GET /xrates HTTP/1.1" 200 1659 "https://golden-eye.herokuapp.com/
xrates" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/71.0.3578.98 Safari/537.36"
2019-02-24T20:50:36.597341+00:00 heroku[router]: at=info method=GET path="/xrates" host=golden-eye.herokuapp.com request_id=3e796b29-c632-4e9b-bb0b-8cad4b4bff
1f fwd="176.115.99.211" dyno=web.1 connect=0ms service=25ms status=200 bytes=1821 protocol=https
2019-02-24T20:50:48.795023+00:00 app[web.1]: 2019-02-24 20:50:48,794 [INFO] - flask.app: Started UpdateRates
2019-02-24T20:50:49.425303+00:00 heroku[router]: at=info method=GET path="/update/840/980" host=golden-eye.herokuapp.com request_id=d89fd4a3-73fb-43af-b540-85
74b17c9024 fwd="176.115.99.211" dyno=web.1 connect=0ms service=633ms status=302 bytes=436 protocol=https
2019-02-24T20:50:49.426568+00:00 app[web.1]: 10.63.102.120 - - [24/Feb/2019:20:50:49 +0000] "GET /update/840/980 HTTP/1.1" 302 221 "https://golden-eye.herokua
pp.com/xrates" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/71.0.3578.98 Safari/537.36"
2019-02-24T20:50:49.670249+00:00 app[web.1]: 2019-02-24 20:50:49,670 [INFO] - flask.app: Started ViewAllRates
2019-02-24T20:50:49.681478+00:00 heroku[router]: at=info method=GET path="/xrates" host=golden-eye.herokuapp.com request_id=4d0cdee1-d494-4657-9266-7de1672988
e8 fwd="176.115.99.211" dyno=web.1 connect=0ms service=13ms status=200 bytes=1822 protocol=https
2019-02-24T20:50:49.682727+00:00 app[web.1]: 10.63.102.120 - - [24/Feb/2019:20:50:49 +0000] "GET /xrates HTTP/1.1" 200 1660 "https://golden-eye.herokuapp.com/
xrates" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/71.0.3578.98 Safari/537.36"
```

Работа с файлами в Heroku

Как мы говорили в начале курса, база данных Sqlite очень удобна благодаря своей простоте, но не рекомендуется для использования на production среде. При этом она не применима при размещении проектов на heroku, так как из-за особенностей файловой системы Heroku файлы не сохраняются и не сохраняют содержимое постоянно в ходе работы приложения.

Но к счастью, Heroku предлагает решение для проектов, требующих СУБД — это СУБД Postgres. Для ее подключения нам нужно внести небольшие изменения в код, а также добавить библиотеки в requirements.txt. И конечно, добавить соответствующее расширение в наше приложение на Heroku.

Переход на использование PostgreSQL

Библиотека python для работы с Postgres — psycopg2. Добавим ее в requirements.txt.

В модуле models.py необходимо будет использовать другой класс БД, вместо SqliteDatabase создавать экземпляр PostgresqlDatabase.

Функцию init_db переименуем в start_db, ее задачей будет проверять, если еще не создана таблица курсов XRate, создавать ее, а также другие таблицы и заполнять таблицу курсов нужными нам данными. Добавим вызов этой функции в модуль app.py после создания экземпляра Flask приложения.

После внесения изменений в проект, закоммитим их и отправим на heroku. Теперь останется только добавить расширение с помощью команды.

```
$ heroku addons:create heroku-postgresql:<PLAN_NAME>
```

В качестве плана укажем бесплатный hobby-dev.

Итоги курса Python Practice

- В ходе работы над курсом мы познакомились с таким понятием как постановка задачи и убедились в нее важности при реализации проекта.
- Мы познакомились и научились использовать много полезных встроенных модулей python, а также сторонних библиотек — logging, requests, traceback, unittest, importlib и др.
- Мы узнали, что такое базы данных и что такое ORM библиотеки для работы с ними, научились использовать библиотеку peewee.
- Узнали, что такое парсинг и научились парсить xml документы с помощью встроенной библиотеки xml и сторонней xmltodict.
- Мы узнали, что такое web приложения, как их создавать и как в этом могут помочь web фреймворки. Начали знакомство с web фреймворком Flask, основными его элементами.
- Написали полноценное web приложение на базе Flask и разместили его на платформе Heroku, а также добавили фоновую задачу по обновлению курсов валют.

Что дальше?

Наш курс подошел к концу, но конечно изучение Python на это мне останавливается. Для более осознанного проектирования и создания web приложений можно порекомендовать:

- Более глубокое изучение Flask.
- Более детальное понимание работы СУБД, языка запросов SQL.
- Работа с БД Postgres или другой полноценной СУБД.
- Знакомство с другой ORM библиотекой.
- Более глубокое понимание протокола HTTP.

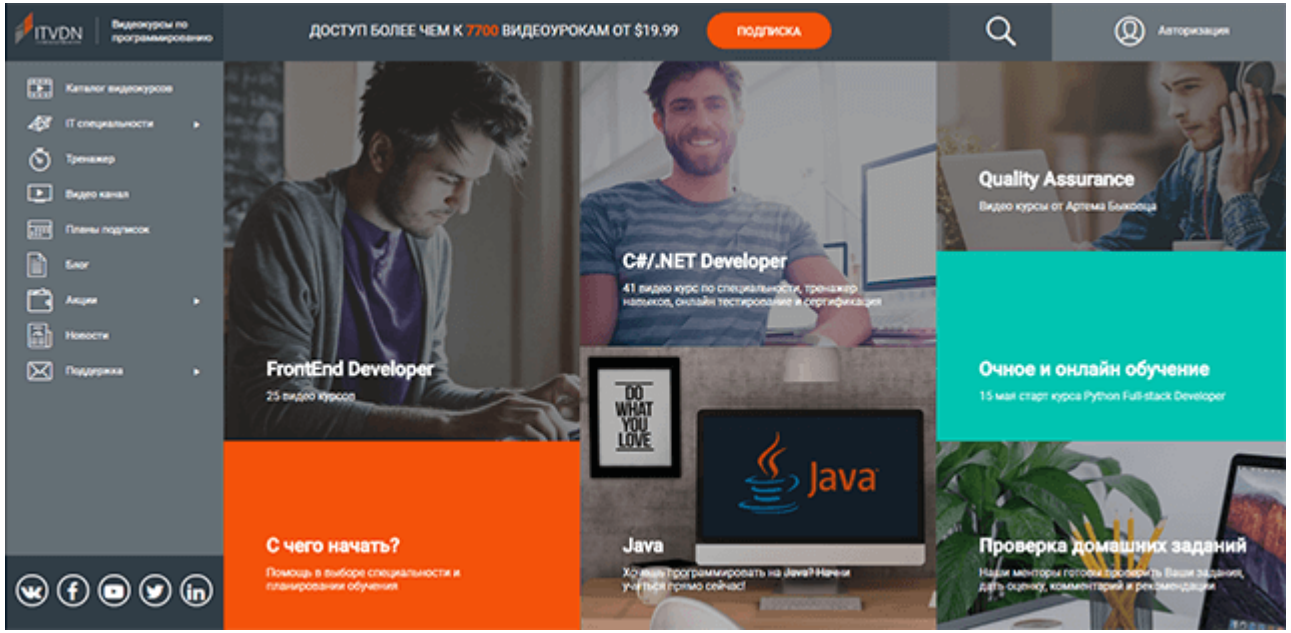
Также в рамках курса мы не коснулись таких интересных моментов как:

- Работа с потоками в python.
- Асинхронность в python.
- Генераторы и итераторы.
- Анализ и визуализация данных с помощью python библиотек.

И да пребудет с Вами сила!

Смотрите наши уроки в видео формате

ITVDN.com



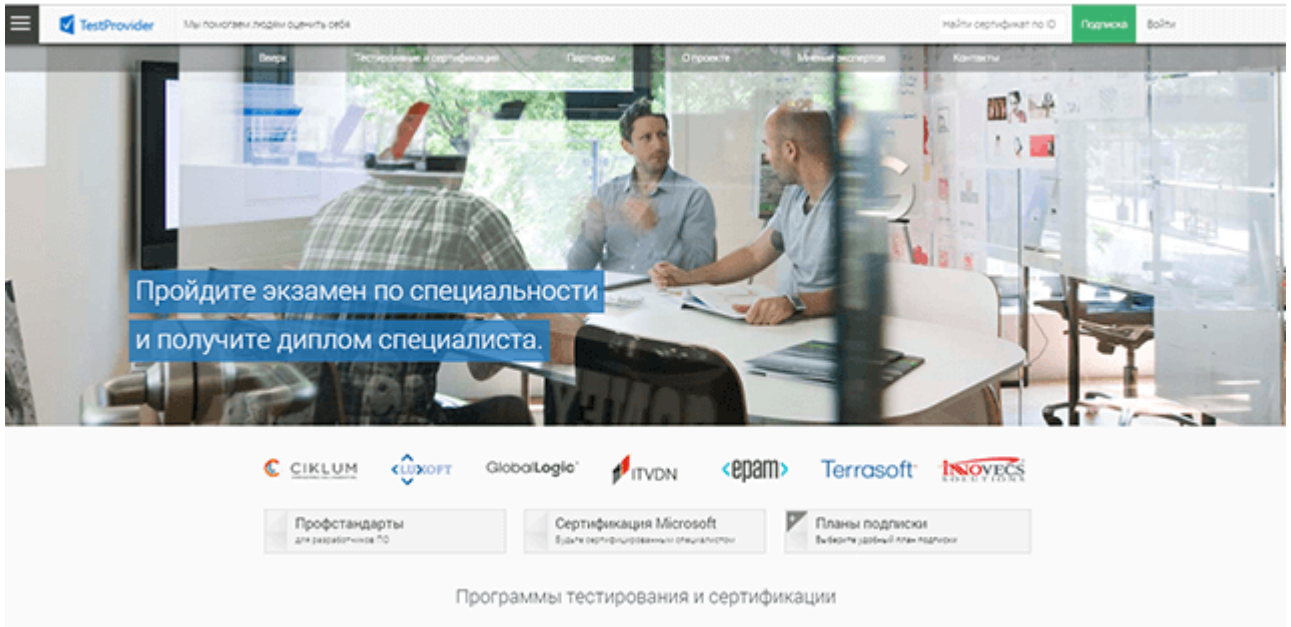
Посмотрите этот урок в видео формате на образовательном портале ITVDN.com для закрепления пройденного материала.

Курсы записаны сертифицированными тренерами, которые работают в учебном центре CyberBionic Systematics и другими высококвалифицированными разработчиками.

ITVDN
IT VIDEO DEVELOPERS NETWORK

Проверка знаний

TestProvider.com



TestProvider – это online сервис проверки знаний по информационным технологиям. С его помощью Вы можете оценить Ваш уровень и выявить слабые места. Он будет полезен как в процессе изучения технологии, так и для общей оценки знаний IT специалиста.

После каждого урока проходите тестирование для проверки знаний на TestProvider.com

Успешное прохождение финального тестирования позволит Вам получить соответствующий Сертификат.

TestProvider

Q&A

Информационный видеосервис для разработчиков программного обеспечения

ITVDN

IT VIDEO DEVELOPERS NETWORK

TestProvider PROMETRIC TEST CENTER CyberBionic systematics Microsoft Partner Silver Learning Windows Azure Cloud Partner EBA European Business Association