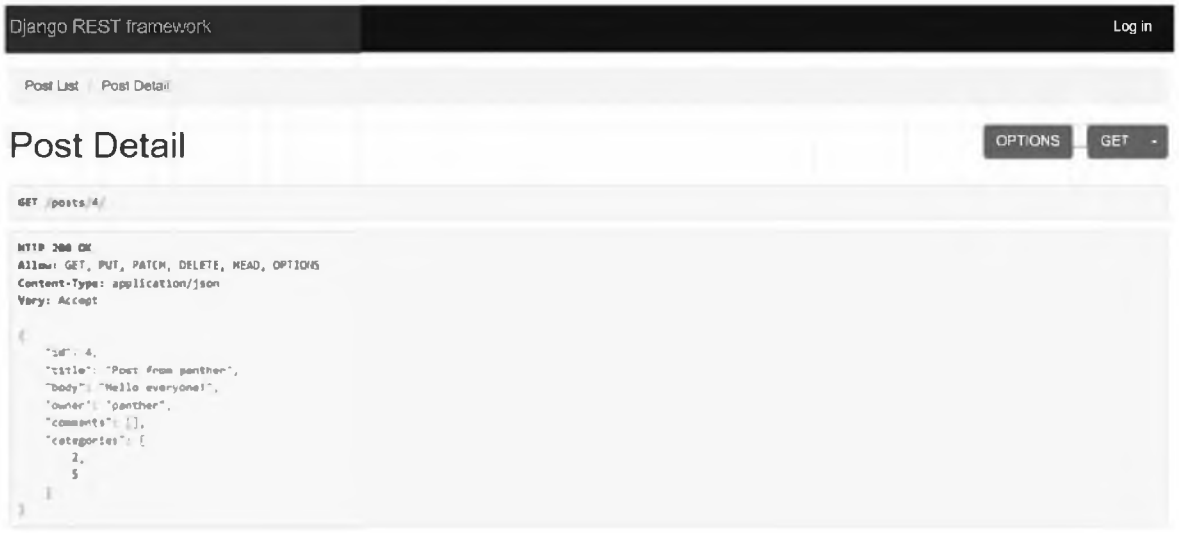

Рисунок 2 – Интерфейс фреймворка для тестирования API

Список использованных источников

1. Wilde E., Pautasso C. REST: From Research to Practice. Springer New York, 2011.
2. Bendoraitis A. Web Development with Django Cookbook: Over 70 Practical Recipes to Create Multilingual, Responsive, and Scalable Websites with Django. Packt Publishing Limited, 2014.

УДК 004.415.25

КОНТЕЙНИРИЗАЦИЯ И РАЗВЕРТЫВАНИЕ ПРИЛОЖЕНИЙ С ПОМОЩЬЮ DOCKER И DOCKER-COMPOSE

Захарчёнок В.Ф., студ., Бизюк А.Н., ст. преп.
Витебский государственный технологический университет,
г. Витебск, Республика Беларусь

Реферат. *В статье рассмотрены вопросы использования Docker и docker-compose для разработки быстро разворачиваемых приложений. Разработаны Docker-файл и compose-файл для компиляции и запуска веб-приложения, использующего базу данных.*

Ключевые слова: веб-сервис, C#, Docker, doccker-compose.

Docker впервые увидел свет в 2011 году во Франции, когда Соломон Хайкс предвидел создание независимых от платформы механизмов для развертывания всех ваших двоичных файлов разработки таким образом, чтобы все библиотеки, зависимости уровня ОС и коды могли быть развернуты на любом компьютере. Облако или система, дающая разработчику уверенность в том, что его программное обеспечение будет работать должным образом, без каких-либо переделок и сбоев [1].

Основная причина концептуализации контейнера докеров заключалась в том, чтобы упростить жизнь разработчика. Настройка конфигурации рабочей среды для поддержки различных версий программного обеспечения на любой конкретной машине была большой проблемой. Но с докером вам просто нужно создать изолированный пакет, состоящий из всех конфигураций (называемых контейнером докеров), поддерживающих разные версии, и каждый контейнер здесь содержит отдельную конфигурацию изолированной среды, которая не влияет на конфигурации уровня ОС или любой другой работающий контейнер.

Чем отличается Docker от Виртуальных Машин?

Docker и виртуальные машины (VM) имеют одну общую цель – изолировать и запускать приложения. Однако у них есть ряд важных отличий:

1. Размер. Docker контейнеры гораздо легче и меньше, чем виртуальные машины. Это обусловлено тем, что Docker использует общую операционную систему для запуска контейнеров, в то время как VM требуют свою собственную операционную систему для каждой машины.

2. Скорость запуска. Docker контейнеры запускаются гораздо быстрее, чем виртуальные машины, так как Docker использует общую операционную систему и не требует загрузки отдельной ОС для каждой машины.

3. Изоляция. Docker контейнеры предоставляют лучшую изоляцию, чем виртуальные машины. Контейнеры могут иметь свои собственные файловые системы, библиотеки и настройки, но они все еще могут обмениваться ресурсами с хост-системой, что позволяет контейнерам работать более эффективно и сокращает накладные расходы.

4. Управление. Docker контейнеры легче управлять, чем виртуальные машины. Docker имеет более простой и удобный интерфейс управления контейнерами, а также более широкое сообщество разработчиков и экосистему.

5. Нагрузка. Виртуальные машины могут обрабатывать более тяжелые и вычислительно сложные задачи, так как они имеют доступ к полной вычислительной мощности хост-системы, в то время как Docker контейнеры обладают ограниченной вычислительной мощностью и могут быть неэффективными при обработке тяжелых нагрузок.

Docker подобен тому огромному грузовому кораблю, загруженному большими ящиками(контейнеры), который предлагает разработчикам программную платформу, которая позволяет им быстро создавать, тестировать и развертывать приложения.

Docker упаковывает программное обеспечение в стандартизированные блоки, называемые контейнерами, которые содержат все необходимое для работы программного обеспечения, включая библиотеки, системные инструменты, код и среду выполнения.

Контейнеры изолированы друг от друга, поэтому можно запускать множество контейнеров одновременно на одном хосте.

На рисунке 1 изображена архитектура Docker.

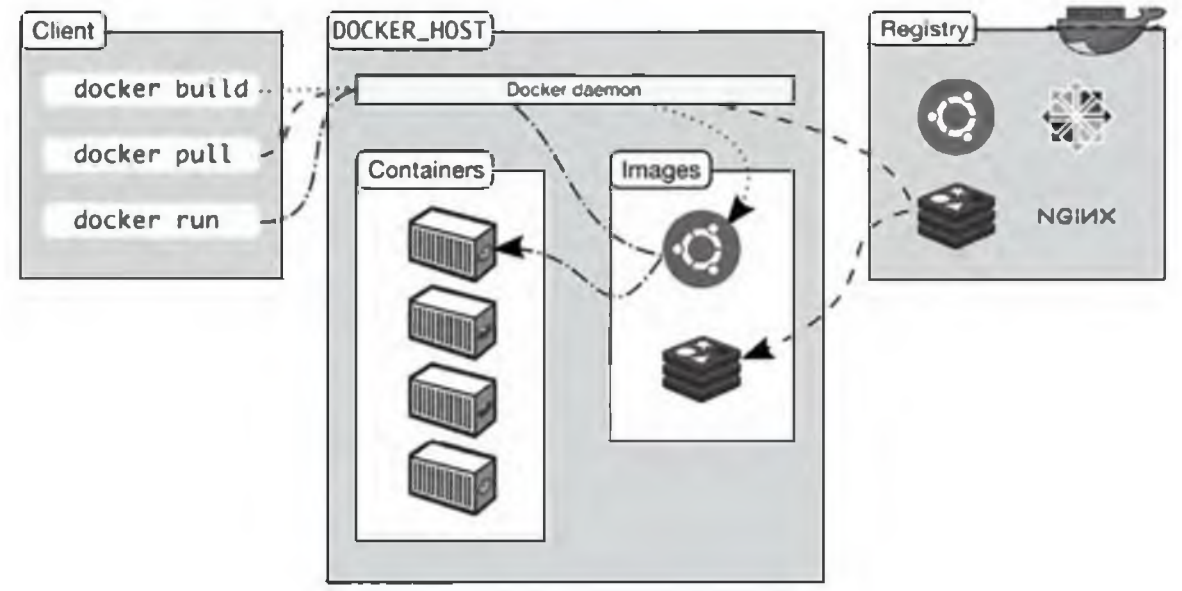

Рисунок 1 – Архитектура Docker

Compose – это инструмент для определения и запуска многоконтейнерных приложений Docker и позволяет легко обрабатывать несколько контейнеров одновременно. С Compose вы используете файл YAML для настройки служб вашего приложения. Затем с помощью одной команды вы создаете и запускаете все службы из вашей конфигурации.

Compose работает во всех средах: производстве, подготовке, разработке, тестировании, а также рабочих процессах непрерывной интеграции.

Использование Compose в основном состоит из трех шагов:

1. Определите среду вашего приложения с помощью Dockerfile, чтобы его можно было воспроизвести где угодно.

2. Определите службы, из которых состоит ваше приложение, в файле docker-compose.yml,

чтобы их можно было запускать вместе в изолированной среде.

3. Запустите docker-compose up, и Compose запустит и запустит все ваше приложение.

Compose-файл предоставляет способ настроить все зависимости службы приложения (базы данных, очереди, кэши, API-интерфейсы веб-службы и т. д.). С помощью инструмента командной строки Compose вы можете создать и запустить один или несколько контейнеров для каждой зависимости с помощью одной команды (docker-compose up).

Важной частью любого процесса непрерывного развертывания или непрерывной интеграции является набор автоматизированных тестов. Для автоматизированного сквозного тестирования требуется среда, в которой можно запускать тесты. Compose предоставляет удобный способ создания и уничтожения изолированных сред тестирования для вашего набора тестов. Определив полную среду в compose-файле, вы можете создавать и уничтожать эти среды всего несколькими командами.

Compose традиционно был сосредоточен на рабочих процессах разработки и тестирования, но с каждым выпуском docker-compose делает успехи в более ориентированных на производство функциях.

Для демонстрации возможностей Docker и docker-compose был реализован проект для развертывания и запуска веб-приложения на языке C# [2] с использованием СУБД PostgeSQL. Проект состоит из двух контейнеров Docker. В одном контейнере происходит компиляция и запуск веб-приложения с использованием dotnet-sdk и сервера ASP.Net. Во втором контейнере разворачивается СУБД. Контейнеры связываются между собой с помощью docker-compose.

Полученные файлы Docker и docker-compose можно использовать для быстрого разворачивания приложения на любом сервере, где установлен Docker, без необходимости ручной установки и настройки веб-сервера и сервера баз данных.

Список использованных источников

1. Merkel D. Docker: lightweight linux containers for consistent development and deployment // Linux journal. 2014. Vol. 2014, № 239. P. 2.
2. Hejlsberg A., Wiltamuth S., Golde P. C# language specification. Addison-Wesley Longman Publishing Co., Inc., 2003.

УДК 004.43:378

СРЕДСТВА АНАЛИЗА ДАННЫХ В PYTHON

Ринейская Р.К., маг., Казаков В.Е., к.т.н., доц., Корниенко А.А., д. ф.-м. н., проф.
Витебский государственный технологический университет,
г. Витебск, Республика Беларусь

Реферат. *В этой статье рассмотрены некоторые инструменты анализа данных в Python и возможности их применения в разработке системы автоматизации кластерного анализа успеваемости учащихся.*

Ключевые слова: Python, кластерный анализ, обработка данных.

Python – один из самых популярных языков программирования, который на сегодняшний день широко используется для разработки программного обеспечения и анализа данных. Рассмотрим основные библиотеки Python для анализа и обработки данных.

1. Pandas – это библиотека программного обеспечения для языка программирования Python, которая используется для работы с данными. Она очень популярна в мире анализа данных благодаря своим функциям, позволяющим обрабатывать данные, очищать данные, агрегировать данные, группировать данные, и многое другое [1].

2. NumPy – это библиотека программного обеспечения для языка программирования Python, которая является базовой для многих других библиотек, таких как Pandas и SciPy. NumPy предоставляет широкий спектр функций для работы с массивами и матрицами. Эта библиотека ускоряет вычисления и используется для численных вычислений [2].

3. Matplotlib – это библиотека для Python, которая помогает создавать графики, диаграммы и другие визуализации данных. Она поддерживает множество различных типов диаграмм и