уровня опасности производства, уменьшению вероятности отказа ответственного и дорогостоящего оборудования, а так же к улучшению качества выпускаемой продукции.

Литература

1. Системы вибрационного контроля и диагностики, [Электронный ресурс] Информационный сайт «Сумма Технологий». Режим доступа: http://www.summatechnology.ru/solutions/sistemy-vibratsionnogo-kontrolya-i-diagnostiki/ - Дата доступа 10.06.2018.

2. Вибрационная диагностика, как элемент технического обслуживания оборудования [Электронный ресурс] Дата доступа 10.06.2018: http://p3s.ru/upload/iblock/0d6/0d6d9b672f83ebaaf3c000f6c0dcbd36.pdf

УДК 004.42

ИНФРАСТРУКТУРА ДЛЯ РАЗВЕРТЫВАНИЯ .NET ПРИЛОЖЕНИЙ В DOCKER-КОНТЕЙНЕРАХ И ИХ ОРКЕСТРАЦИИ

магистрант Якимович П. С.
Научный руководитель – к. ф.-м. н. Козадаев К. В.
Белорусский государственный университет
Минск, Беларусь

Сегодня на платформе .NET часто разрабатывают крупные веб-приложения, которые состоят из нескольких сервисов, а чаще из микросервисов, так как платформа .NET предоставляет много точек расширения и возможностей для построения масштабируемых приложений. Все эти сервисы должны разворачиваться отдельно и независимо, ведь работают они абсолютно автономно. Для этого используются Docker-контейнеры. Контейнеры отличаются от виртуальных машин тем, что разные контейнеры могут использовать одно ядро ОС, что делает контейнеризацию более легковесным способом изоляции приложений друг от друга [1]. Не должно быть никаких проблем при обновлении одного из контейнеров. Поэтому встает проблема изоляции этих контейнеров. Также появляется новая проблема: как управлять этими изолированными контейнерами. Автоматическое развертывание, координация и управление контейнерами называется оркестрацией [2].

Для решения этих проблем часто используются Docker-контейнеры и Kubernetes для оркестрации. При разработке больших приложений разными командами необходим иметь какой-то общий шаблон, который позволит быстро создать необходимую инфраструктуру для развертывания приложений в контейнере и их оркестрации.

На сегодня есть несколько средств для обеспечения шаблонами команд, разрабатывающих одно и то же приложение, но разные сервисы.

Например, существует приложение, разработанное компанией Microsoft, eShopOnContainers. Это приложение для интернет-магазина, построенное на микросервисной архитектуре с примерами файлов для конфигурации Kubernetes и Docker. Недостаток этого приложения в том, что оно решает определенные задачи. В этом приложении есть спорные решения, которые могут не подойти в качестве шаблонной инфраструктуры.

Также можно опираться на документацию используемых технологий. Docker и Kubernetes предлагают информативную документацию, которой будет достаточно для развертывания небольшого приложения в контейнере и управления им [1] [2]. Однако чаще всего мы сталкиваемся с более сложными ситуациями, чем описываемые в базовой документации.

Еще можно пользоваться решениями, предлагаемыми в каталоге Yeoman. Yeoman – это технология для выстраивания удобного и современного процесса разработки. Состоит она из трех компонентов:

- Yo — технология для быстрой установки основы приложения.
- Grunt — Javascript технология выстраивающая процесс сборки.
- Bower — менеджер зависимостей/пакетов клиентских Javascript библиотек.

Это неплохой вариант, однако существующие шаблоны предлагают гораздо больше, чем часто требуется для разработки микросервиса. Помимо инфраструктуры, шаблоны предлагают также и структуру проекта, что не всегда подходит разным командам.

Также для использования Yeoman нужно устанавливать на машину разработчиков дополнительные компоненты, необходимые для работы Yeoman.

В данной работе ставится задача разработки шаблонной инфраструктуры, которая предлагает конфигурационные файлы для Docker и Kubernetes, чтобы с их помощью можно было быстро и беспроблемно развернуть свой сервис и управлять им в последствии.

Инфраструктуру можно применять для приложений с абсолютно разными задачами, так как она не ограничивает проект в своем функционале, не предлагает готовую структуру для проекта, а лишь предоставляет конфигурационные файлы для развертывания приложения в контейнере и оркестрации этих контейнеров.

Шаблонная инфраструктура включает в себя dockerfile, который используется для конфигурации Docker-контейнеров [1]. Он включает в себя информацию, нужную для подготовки образов, исходя из которых будет запускаться контейнер. Там содержится информация о портах, которые будет слушать сервис, о месте нахождения и копирования артефактов программы. Dockerfile содержит набор инструкций с аргументами. Каждая инструкция пишется заглавными буквами. Инструкции обрабатываются сверху вниз. Каждая инструкция добавляет новый слой в образ. Docker исполняет инструкции, следуя процессу.

Также инфраструктура включает в себя файл для конфигурации Kubernetes. Подключив веб-интерфейс для Kubernetes, мы сможем управлять контейнерами, останавливать их или запускать, проводить обновление контейнеров так, чтобы не останавливать целое приложение, а лишь один из контейнеров, обновлять его и снова запускать работать [2].

Всё это позволит разработчикам легко начинать работу над новыми сервисами, имея шаблон инфраструктуры, который, конечно, можно будет изменить и настроить под нужды конкретного приложения.

Литература

1. Моут Э., Использование Docker / Эдриен Моут – ДМК Пресс, 2017. – 354 с.
2. Poulton N., The Kubernetes Book / Nigel Poulton – Independently Published, 2017 – 137 p.

УДК 621.382

СПОСОБ ПОЛУЧЕНИЯ ОРЕБРЕНИЯ БЕЗОТХОДНЫМ ВИХРЕВЫМ РЕЗАНИЕМ

аспирант кафедры «Технологическое оборудование» Ермалович В.И.
Научный руководитель – к. т. н. Якимович А.М.
Белорусский национальный технический университет
Минск, Беларусь