УДК 004.428

Д. О. Мельник
Д.І. Катєльніков

ОСОБЛИВОСТІ СТВОРЕННЯ ПРОТОТИПУ CI ПЛАТФОРМИ ЗА ДОПОМОГОЮ NODE.JS ТА DOCKER

Вінницький національний технічний університет

Анотація

В даній роботі розглянуто особливості створення CI(Continuous Integration) платформи з використанням таких технологій як Node.js та Docker. Досліджено предметну область та можливості реалізації програмного продукту за допомогою різних бібліотек та інструментів. Детальний огляд Docker API, як ключового інструменту створення сучасної CI платформи.

Ключові слова: CI/CD, Node.js, Docker, аналіз, Github webhooks.

Abstract

There are considered possibilities of implementing the methodology of mathematical analysis in programming. Also researched the advantages and disadvantages of different libraries, providing this functionality. The main requirements and criteria are formed according to following functionality: parsing mathematical expressions to data structures, differentiation of functions, calculating the integral sums of different types, performing a Fourier transforms, etc.

Keywords: CI/CD, Node.js, Docker, analysis, Github webhooks.

Безперервна інтеграція (англ. Continuous Integration) — практика розробки програмного забезпечення, яка полягає у виконанні частих автоматизованих складань проекту для якнайшвидшого виявлення та вирішення інтеграційних проблем. У звичайному проекті, де над різними частинами системи розробники працюють незалежно, стадія інтеграції є завершальною. Вона може непередбачувано затримати закінчення робіт. Перехід до неперервної (постійної) інтеграції дозволяє знизити трудомісткість інтеграції і зробити її передбачуваною за рахунок найбільш раннього виявлення та усунення помилок і суперечностей. Це концепція, яка реалізується як конвеєр, полегшуючи злиття тільки що закоміченного коду в основну кодову базу. Концепція дозволяє запускати різні типи тестів на кожному етапі.[1]

В якості мови програмування було обрано JavaScript, а саме його середовище виконання Node.js. Це динамічно типізована мова, що в даному випадку є перевагою, так як це дає можливість досить гнучко маніпулювати данними та постійно доповнювати й модифікувати програмний продукт протягом його розробки. Також Node.js це подійно-орієнтована мова програмування, що дає можливість одночасної обробки великої кількості запитів з мінімальними затратами ресурсів машини, так як ця концепція не відповідає застарілій “один потік - одне з’єднання”.[2]

Ключевым компонентом CI платформи є середовище виконання “кроків”, тобто етапів процесу інтеграції. Дуже важливим елементом процесу будування такого “кроку” є його кастомізація. Для цього було вирішено вдатися до застосування віртуалізації.

Прикладом використання віртуалізації є можливість запуску декількох операційних систем на одному комп'ютері: при тому кожен з екземплярів таких гостьових операційних систем працює зі своїм набором логічних ресурсів (процесорних, оперативної пам'яті, пристроїв зберігання), наданням яких із загального пулу, доступного на рівні обладнання, управляє хостової операційна система - гіпервізор.

На сьогоднішній день найпопулярнішим й найзручнішим інструментом віртуалізації є Docker. Він має зручний API який чудово підтримується вище обраним Node.js, та підтримку Windows та Linux контейнерів (в рамках розробленого проекту будуть використовуватися лише Linux контейнери). Також він дозволяє досить сильно кастомізувати середовище виконання - починаючи від налаштувань операційної системі в якій буде виконуватися “крок”, й закінчуючи командами які будуть виконуватися. Користувач зможе задати поведінку контейнера без жодних зовнішніх

інструментів, наприклад використавши bash script. Кастомізація такого рівня досягається за допомогою “образів”, які користувач може створювати сам.

Контейнери та віртуальні машини мають подібні переваги виділення та розподілу ресурсів, але функціонують по-різному, оскільки контейнери віртуалізують операційну систему замість апаратних засобів. Контейнери більш портативні та ефективніші.[3]

Образ контейнера Docker - це легкий, автономний, виконуваний пакет програмного забезпечення, який включає все необхідне для запуску програми: код, час виконання, системні інструменти, системні бібліотеки та налаштування.

Розглянувши Docker API більше детальніше можемо побачити, що там є все необхідне для його безболісного використання з Node.js. По-перше, Node.js має чудову бібліотеку для роботи з Docker - dockerode. Вона повністю дублює його API, адаптуючи його для роботи з Node.js. По-друге, Docker має всі інструменти для маніпулювання “контейнерами”.[4] Є можливість виконувати такі операції над ними:

- створення на основі “образу”
- запуск
- отримання логів
- отримання логів в режимі реального часу
- зупинка
- видалення

Наступним етапом створення платформи є інтеграція з системою контролю версій. Оскільки мова йде про безперервну інтеграцію потрібно якось реагувати на зміну в кодовій базу проекту. Наприклад при паралельній розробці (в окремих гілках) декількох компонент програмного продукту варто весь час перевіряти якість та правильність написання коду у обох розробників. Зручним інструментом в даному випадку можуть слугувати “вебхуки” (выд англ. webhook).

Webhook – це метод збільшення або розширення функціональності веб-сторінки або веб-застосунку за допомогою користувацьких зворотних викликів (callbacks). Ці зворотні виклики можуть обслуговуватися або керуватися користувачами або веб-розробниками, які не обов’язково пов’язані з вищезгаданим сайтом або веб-застосунком.[5]

Таким чином розроблена CI система зможе отримувати сповіщення на предмет зміни коду в репозиторії.[6] В результаті обробки “вебхуку” буде вирішено який описаний користувачем “конвеєр” буде запущено. Під час запуску першого “кроку” у файлову систему контейнера буде склоновано репозиторій та вибрана ревізія, яка відповідає тій що прийшла в “вебхуку”.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. CI essentials – [Електронний ресурс]. – Режим доступу до матеріалу: https://codeship.com/continuous-integration-essentials
2. The Positive and Negative Aspects of Node.js Web App Development – [Електронний ресурс]. – Режим доступу до матеріалу: https://www.mindinventory.com/blog/pros-and-cons-of-node-js
3. How is Docker different from a virtual machine? – [Електронний ресурс]. – Режим доступу до матеріалу: https://stackoverflow.com/questions/16047306/how-is-docker-different-from-a-virtual-machine
4. Docker API reference – [Електронний ресурс]. – Режим доступу до матеріалу: https://docs.docker.com/engine/api/v1.24/
5. Webhook – [Електронний ресурс]. – Режим доступу до матеріалу: https://uk.wikipedia.org/wiki/Webhook
6. Github webhooks API documentation – [Електронний ресурс]. – Режим доступу до матеріалу: https://developer.github.com/webhooks/

Мельник Денис Олександрович — студент групи 1ПІ-16б, факультет інформаційних технологій та комп’ютерної інженерії, Вінницький національний технічний університет, м. Вінниця, e-mail: denismel19@gmail.com

Науковий керівник – ***Катєльніков Денис Іванович***, кандидат технічних наук, доцент кафедри програмного забезпечення, Вінницький національний технічний університет, м. Вінниця, e-mail: fuzzy2dik@gmail.com.

Melnyk Denys O. — Faculty of Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: denismel19@gmail.com

Supervisor – ***Katielnikov Denys Ivanovych***, PhD, Associate Professor of Software Engineering Department, Vinnytsia National Technical University, Vinnytsia, E-mail: fuzzy2dik@gmail.com.