

Бахметьев В.В.¹, Киричек Г.Г.²

¹ студ. гр. КНТ-518, НУ «Запорізька Політехніка»

² канд. техн. наук, доц., НУ «Запорізька Політехніка»

ДОЦІЛЬНІСТЬ ВИКОРИСТАННЯ ТЕХНОЛОГІЇ SERVERLESS COMPUTING В JAVASCRIPT ДОДАТКАХ

Безсерверні обчислення (Serverless computing) є способом розгортання та виконання додатків у хмарній інфраструктурі з оплатою за фактичне використання, без оренди чи покупки серверів. За планування, масштабування, балансування та моніторингу обчислювальної потужності відповідає постачальник послуг безсерверних обчислень. Крім цього він може розглядати ваші програми як функції.

Попри свою назву, технологія безсерверних обчислень не виключає існування серверів. Для програмного забезпечення потрібні пристрої на яких воно буде виконуватися. Під «безсерверний» просто маємо на увазі, що компаніям, організаціям чи розробникам непотрібен фізичний сервер. Ви просто вивантажуєте програму в хмару постачальника послуг безсерверних обчислень, який подбає про те, щоб виділити сервери, сховища, налаштувати середовище виконання програми та запустити його [1].

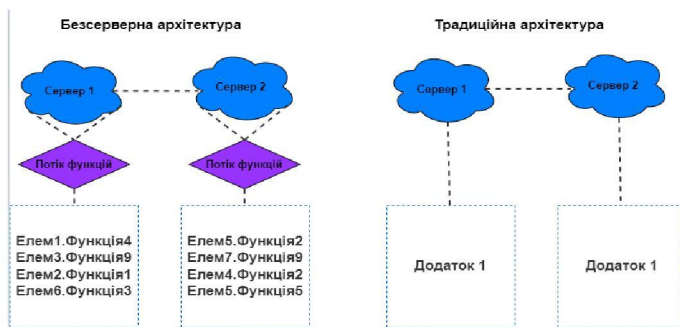


Рисунок 1 – Традиційна та безсерверна архітектура

Хмарний додаток стійкий до помилок та збоїв. Якщо з якоїсь причини апаратне забезпечення (сервер або маршрутизатор) виходить з ладу або в базі даних відбувається катастрофічна помилка, хмарна програма повинна виявити помилку та виправити її, можливо, шляхом створення нового інстансу на інший стійці або навіть в іншому хмарному дата-центрі в іншому регіоні [2].

Постачальник послуг збереже вашу програму в деякому контейнері, який представляє ізольоване оточення, що містить усе необхідне вашому додатку для роботи. Контейнер дозволяє постачальнику послуги

безсерверних обчислень безпечно переміщати та зберігати вашу програму, а також виконувати її та копіювати в залежності від ваших потреб. Але головна перевага безсерверної технології це відсутність необхідності виконувати будь-які налаштування, балансувати, масштабувати сервери та вирішувати завдання керування сервером. Це виконує провайдер і гарантує, що зі збільшенням навантаження на ваш додаток він створить достатню кількість копій контейнера з додатком для обробки всіх дзвінків. При цьому кожен контейнер є точною копією вихідного. Рішення про запуск ще однієї копії контейнера приймається провайдером, тільки коли кількість запитів до вашої програми стає настільки великим, що поточне число діючих контейнерів не встигає обробити їх усі. Якщо до вашої програми взагалі не надходить запитів, провайдер зупинить усі екземпляри, відповідно, не буде витрачатися процесорний час сервера [1].

Провайдер, який надає послуги безсерверних обчислень відповідає за всі деталі, що стосуються роботи: він знає, де зберігається ваш додаток, як і куди його копіювати, коли запускати нові контейнери та коли зменшувати кількість контейнерів при зниженні навантаження.

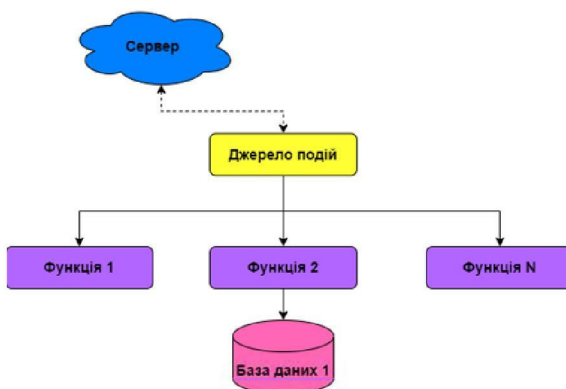


Рисунок 2 – Принцип роботи безсерверних обчислень

До переваг безсерверних обчислень при використанні їх у додатках на JavaScript, відносять:

- зниження витрат - безсерверні обчислення, як правило, вигідні, оскільки у багатьох великих провайдерів хмарних серверних послуг користувач платить за простір або час простою процесора, що не використовується;

- спрощена масштабованість - розробникам, які використовують безсерверну архітектуру, не потрібно турбуватися про деталі для

масштабування свого коду. Безсерверний постачальник виконує масштабування за запитом;

- спрощений внутрішній код - за допомогою функціоналу «хмари» розробники можуть створювати прості функції, які виконують одне завдання, наприклад, виконання виклику API;

- швидший оборот - безсерверна архітектура може значно скоротити час виходу на ринок. Замість того, щоб вимагати складного процесу розгортання для виправлення помилок та нових функцій, розробники можуть додавати та змінювати код частинами [3];

- безсерверні API можуть бути дуже потужними і відрізняються неймовірною здатністю до масштабування, але іноді традиційні API виявляються вдалим рішенням;

- додаткова затримка в обробці запиту неприпустима. Безсерверні програми не можуть гарантувати мінімальну затримку;

- важливо гарантувати певний рівень доступності. У більшості випадків AWS забезпечує досить високий рівень доступності, але іноді цього недостатньо;

- програма виконує важкі та тривалі обчислення;

- API має відповідати спеціальним стандартам. Хмарні сервери можуть виявитися недостатньо гнучкими для цього [1].

Висновки. Безсерверні обчислення абстрагують сервери від створення програмного забезпечення. Безсерверний JavaScript додаток відрізняється від традиційного тим, що керується подіями, автоматично розподіляється та масштабується. Така архітектура є дуже перспективною і має багато причин для використання, в тому числі і у JavaScript додатках, але вона не є панацеєю, і в деяких ситуаціях це найкращий вибір, наприклад, для програм реального часу з веб-сокетами.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Стоянович С., Симович А. Бессерверные приложения на JavaScript. 2020. 394 с.

2. Лашевски Т., Арора К., Фарр Э., Зонуз П. Облачные архитектуры: разработка устойчивых и экономичных облачных приложений. 2022. 320 с.

3. Hellerstein, J. M., Faleiro, J., Gonzalez, J. E., Schleier-Smith, J., Sreekanti, V., Tumanov, A., Wu, C. Serverless computing: One step forward, two steps back. arXiv preprint arXiv:1812.03651. University of California at Berkeley. 2018.