Мацапура В. О., студент,
Вакалюк Т. А., д.пед.н., проф.,
професор кафедри інженерії програмного забезпечення
Державний університет «Житомирська політехніка»

МОЖЛИВОСТІ ВИКОРИСТАННЯ СУЧАСНИХ ХМАРНИХ ТЕХНОЛОГІЙ ДЛЯ МІКРО-СЕРВІСНИХ ДОДАТКІВ

На сьогоднішній день багатьом компаніям або користувачам необхідно обробляти великі обсяги інформації даних або потрібно виконати складні математичні обчислення, і їм потрібна велика кількість обчислювальних потужностей для їх вирішення. Можливо, потужність персонального комп'ютера може бути недостатньо, щоб виконати за певний час одне завдання, а може це завдання вимагає більших потужностей ніж має компанія, і для компанії немає сенсу купувати апаратні засоби, необхідні для його виконання, адже існують хмарні рішення, які можна отримати в любий час, в зручному для компанії регіоні та не турбуватись, що дані можуть бути втрачені внаслідок будь-яких явищ.

Сам термін архітектури мікро-сервісів ще почав зароджуватись в 2010 році, цей метод описував інший підхід до побудови архітектури при розробці програмних додатків. Мета архітектури мікро-сервісів полягає в ідеї, що додаток повинен будуватись як сукупність невеликих сервісів, що спілкуються між собою за допомогою стандартних механізмів як HTTP, HTTPS. Самі сервіси побудовані на основі бізнес потреб компанії та розгортаються в автоматизованому середовищі, а також сервіси можуть бути написані на різних мовах програмування та використовувати різні сховища даних.

З розвитком хмарних технологій на ринку з’явилось чимало хмарних рішень від різних провайдерів, тому постає актуальність проблеми в аналізі хмарних рішень від хмарного провайдеру, та за допомогою яких сервісів від хмарних провайдерів компанії можуть швидко розробляти та доставляти до користувачів свої програмні додатки з архітектурою мікро-сервісів.

У монолітних архітектурах всі процеси тісно пов'язані між собою і працюють як єдиний сервіс. Це означає, що якщо один процес додатку має сплеск запитів, вся архітектура повинна бути масштабована одразу. Додавання або поліпшення функцій монолітного додатку стає більш складним у міру зростання кодової бази. Ця складність обмежує експерименти і ускладнює реалізацію нових ідей. Монолітні архітектури збільшують ризик для доступності додатків, оскільки багато залежних і тісно пов'язаних між собою процесів збільшують вплив збою одного процесу. В архітектурі мікро-сервісів додаток будується як незалежні компоненти, які виконують кожен прикладний процес як сервіс. Ці сервіси взаємодіють через чітко визначений інтерфейс з використанням легких API. Сервіси створюються для бізнес-можливостей, і кожен сервіс виконує одну функцію. Оскільки вони запускаються незалежно, кожна служба може бути оновлена, розгорнута і масштабована для задоволення попиту на конкретні функції програми.

Для аналізу можливостей використання сучасних хмарних технологій було обрано рішення від Amazon Web Services [1]. Для використання обчислювальні потужностей для мікро-сервісів можливе використання високомасштабованого, високопродуктивного сервісу управління контейнерами Amazon Elastic Container Service[2], який підтримує контейнери Docker і дозволяє легко запускати додатки на керованому кластері Amazon EC2. Можливе використання обчислювальних потужностей сервісу AWS Lambda, який дозволяє запускати код без резервування або управління серверами. Просте завантажування коду, а AWS Lambda керує всім, що потрібно для запуску та масштабування коду з високою доступністю.

Для аналізу використання сховища та бази даних AWS надає такі сервіси для використання як Amazon ElastiCache , який підвищує продуктивність сервісу, дозволяючи отримувати інформацію з швидких, керованих кешів в пам'яті, замість того, щоб повністю покладатися на більш повільні дискові бази даних.

Amazon S3 надає розробникам та IT-командам високонадійне, безпечне та масштабоване об'єктне сховище для всіх їхніх даних, як великих, так і малих.

Amazon DynamoDB - повністю керований, швидкий і гнучкий сервіс баз даних NoSQL для всіх додатків, які потребують послідовної, однозначної, мілісекундної затримки при будь-якому масштабуванні сервісу.

Amazon RDS - легке налаштування, експлуатація та масштабування реляційної бази даних у хмарі. Можливе обрання з шести знайомих механізмів баз даних, включаючи Oracle, Microsoft SQL Server, PostgreSQL, MySQL та MariaDB.

Amazon Aurora - Реляційна СУБД, яка поєднує в собі швидкість і надійність висококласних комерційних баз даних з простотою і економічністю баз даних з відкритим вихідним кодом. Забезпечує до 5 разів більшу пропускну здатність, ніж стандартний MySQL, що працює на тому ж обладнанні.

Провівши аналіз можливостей мережевих послуг з високою пропускною здатністю та з мінімальною затримкою від AWS було виокремлено такі можливі сервіси для мікро-сервісного додатку, як AWS Cloud Map - це служба виявлення всіх хмарних ресурсів. За допомогою Cloud Map можна визначати власні імена для ресурсів додатків, і сервіс підтримує оновлене розташування цих ресурсів, що динамічно змінюються.

Існує також AWS App Mesh, який полегшує моніторинг і управління мікро-сервісами, що працюють на AWS. App Mesh стандартизує взаємодію мікро-сервісів, надаючи наскрізну видимість і допомагаючи забезпечити високу доступність додатків. Найбільш потужними сервісами для мережевих рішень від AWS є Application Load Balancer , який балансує трафік HTTP і HTTPS на прикладному рівні (рівень 7 моделі OSI), забезпечуючи розширену маршрутизацію запитів, орієнтовану на доставку сучасних архітектур додатків, включаючи мікро-сервіси і контейнери, а також Network Load Balancer, який пропонує високопродуктивне балансування

навантаження, яке працює на рівні мережевого з'єднання (рівень 4 моделі OSI) і дозволяє маршрутизувати з'єднання з мікро-сервісами на основі даних IP-протоколу. Network Load Balancer може обробляти мільйони запитів в секунду, зберігаючи при цьому наднизькі затримки.

Також важливими сервісами при мережевих налаштувань додатку є Amazon Route 53, який є високодоступним і масштабованим хмарним веб-сервісом системи доменних імен (DNS), який ефективно з'єднує запити з інфраструктурою, що працює в AWS. Він може використовуватися для перевірки працездатності IP та виявлення сервісів для роботи мікро-сервісів. Можливе використання Amazon API Gateway, який пропонує комплексну платформу для управління API. Amazon API Gateway дозволяє обробляти сотні тисяч паралельних викликів API та здійснює управління трафіком, авторизацією та контролем доступу, моніторингом та управлінням версіями API.

Список використаних джерел

1. Amazon. Amazon Web Services [Електронний ресурс] / Amazon – Режим доступу до ресурсу: https://aws.amazon.com/.
2. Amazon Web Services. AWS Cloud Products [Електронний ресурс] / Amazon Web Services – Режим доступу до ресурсу: https://aws.amazon.com/products/.