УДК 004.075:004.04:51-3
Науменко Т. О., аспірант
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

БЕЗСЕРВЕРНА ТЕХНОЛОГІЯ (FUNCTIONS AS A SERVICE) ДЛЯ СТВОРЕННЯ ХМАРНИХ МІКРОСЕРВІСНИХ ДОДАТКІВ.

Науменко Т. О. Безсерверна технологія (Functions as a SERVICE) для створення хмарних мікросервісних додатків. У статті автор робить детальний аналіз необхідності та актуальності використання технологій, а саме у вигляді «Функція як сервіс». Досліджується використання веб-сервісів та контейнерів для хмарних сервіс-орієнтованих прикладних додатків.
Ключові слова: хмарні сервіси, мікросервісні додатки, обробка даних, метод віртуалізації.

Науменко Т. А. Безсерверна технология (Functions as a Service) для создания облачных микросервисних приложений. В статье автор делает подробный анализ необходимости и актуальности использования безсерверних технологий, а именно в виде «Функция как сервис». Исследуется использование веб-сервисов и контейнеров для облачных сервис-ориентированных прикладных приложений.
Ключевые слова: облачные сервисы, микросервисни приложения, обработка данных, метод виртуализации.

Naumenko T. A. Non-server technology (Functions as a Service) to create cloud-based micro-service applications. In the article, the author make a detailed analysis of the necessity and relevance of using non-server technologies, namely in the form of "Function as a service". The use of Web services and containers for cloud-based service-oriented applications is explored.
Keywords: cloud services, microservice applications, data processing, virtualization method.

Вступ. Науковий розвиток інноваційних технологій, розширює спектр використання інформаційних засобів, які є результатом інтелектуальної діяльності висококваліфікованої частини сучасного суспільства, що в свою чергу викликає необхідність розвитку сервісів хмарних обчислень.

Використання та зберігання великих обсягів інформації є просто необхідним в сучасному суспільстві, але не завжди можливе в рамках одного підприємства у зв'язку з особливостями діяльності та наявністю необхідних апаратних та трудових ресурсів [4].

З цієї причини стає актуальною розробка безсерверної технології (Functions as a Service) для створення хмарних мікросервісних додатків, оскільки це в свою чергу дозволить клієнтам розробляти, запускати та керувати функціональними можливостями додатків без складності створення та підтримки інфраструктури, яка зазвичай пов'язана з розробкою та запуском додатка [3]. Перехід на автоматизацію зберігання та обробки інформації дозволить підвищити продуктивність праці в рамках конкретної галузі промисловості. У зв'язку з розвитком інформаційно-комунікаційних технологій стало можливим використання онлайн-сервісів у всіх ланках виробництва.

Аналіз останніх досліджень і публікацій. Історія зародження хмарних обчислень починається з 1950-х років вона тісно пов'язана із розвитком електроенергетики, перші кроки в розвитку безсерверних технологій та їх порівняння із можливими електронними носіями можна знайти в роботах Дугласа (Douglas Parkhill) зокрема в книзі «The Challenge of the Computer Utility» 1966 року. На думку іншого вченого тих часів Херба Гроша, необхідно близько 15 масштабних центрів для обробки всіх даних, які виникають в процесі діяльності людства в цілому світі. Якщо згадати початок 1990-х років то людство мало змогу оперувати лише фізичними серверами, які можна було охарактеризувати як обмежений ресурс, і для того щоб отримати нове необхідний значний проміжок часу. Формування методу віртуалізації припадає на кінець 1990-х на початок 2000-х років, що докорінно змінив основні підходи в роботі серверних систем. Аналітики «Гартнер Груп» («Gartner Group») провели детальний аналіз та шляхи розвитку хмарних обчислень, у результаті безсерверні технології є найбільш перспективним напрямком вдосконалення, а саме тільки на найближчі 5-7 років значна частина існуючих інформаційних систем перейде в хмари [5,6].

У 2015 (або хтось каже, що у 2012 рік) настав час комп'ютерної парадигми. Існує ряд інтерпретацій, запропонованих для терміна "безсерверний" та його наслідків. Одна школа приписує його до технології Бекенд як служба (Backend as a Service, BaaS). Наприклад, служби автентифікації, що запропоновані сторонніми постачальниками, як Google або Facebook. Інша

школа посилається на концепцію, в якій серверна сторона логіки запускається за допомогою контейнерів без статусу (stateless containers), керованих стороннім провайдером у повному обсязі, яка має назву Функція як служба (Function as a Secrice, FaaS). [8] Однією з перших широко відомих реалізацій останньої є представлений в 2014 році сервіс AWS Lambda [3], аналогічні [2] пропозиції серед публічних FaaS є у Google (Cloud Functions [5]), IBM (на Apache OpenWhisk в складі платформи Bluemix) і Microsoft (Azure Functions).

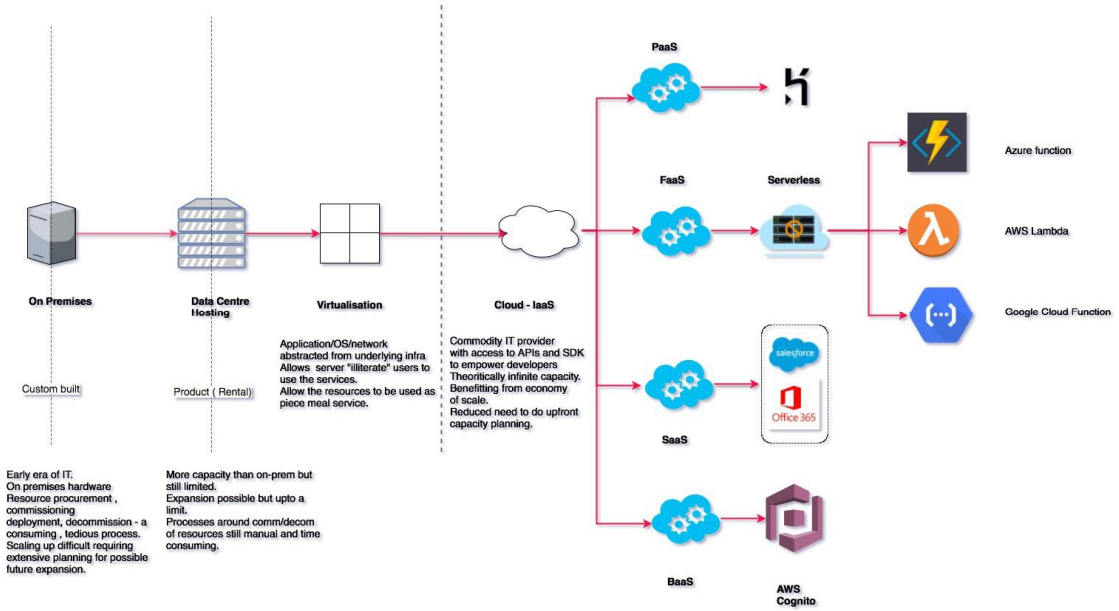


Рис. 1. Коротка еволюція парадигми хмарних обчислень [8]

Хмарні обчислення володіють багатьма перевагами в порівнянні з традиційними рішеннями для побудови інфраструктур підприємств, в пропозиції сервісів і послуг і т.п. [2]. Даний напрям дослідження є досить актуальним про що свідчить поява, останнім часом, великої кількості публікацій по даній тематиці, зокрема з рядом нових результатів можна ознайомитися в роботах А. В. Богданова, А. А. Лобанова, Г. І. Радченка та ін. Аналізуючи наукові розробки, бачимо, що вкрай непростою та актуальною задачею є правильний вибір хмарних платформ, оскільки кожна з них має власні особливості та переваги [1]. Виникає потреба у грунтовному аналізі та систематизації матеріалу стосовно розробки безсерверної технології (Functions as a Service) для створення хмарних мікросервісних додатків.

Формулювання цілей статті (постановка завдання). Дослідити теоретико-практичні основи створення ПЗ, що базується на безсерверній технології, у вигляді «Функція як сервіс» для хмарних сервіс-орієнтованих прикладних додатків на основі використання веб-сервісів і контейнерів.

Виклад основного матеріалу дослідження. Використання безсерверних технологій це своєрідна модель хмарних обчислень де платформа динамічно керує виділенням машинних ресурсів. Хмарні обчислення (англ. cloud computing) представляють собою систему розподіленої обробки даних, де необхідні ресурси представлені як інтернет-сервіс. Під час роботи з хмарними технологіями, зберігання даних, обробка інформації, проведення обчислень та інші операції переходять на сервери Всесвітньої Мережі. Оскільки мережа Інтернет, на сьогодні, може задовольнити попити широкого спектру користувачів причому в доволі громіздких запитах та великих діапазонах даних.

Досліджувані безсерверні обчислення носять назву «Функція як послуга» (англ. *Function as a Service, FaaS*), це можна пояснити тим що в основі коду є функція, яка обробляється певною платформою. Мається на увазі, для обробки та виконання одного запиту необхідно створити, а після завершення видалити окремий контейнер.[3] Проведемо аналіз переваг використання безсерверної технології Functions as a Service, перш за все це зниження потреби в підтримці і програмному забезпеченні. Хмарні обчислення, як правило, не вимагають чисельного IT-

персоналу, оскільки зникає потреба в управлінні програмним забезпеченням. Хмарні сервіси не вимагають потужних комп'ютерів, тож не має необхідності часто оновлювати техніку. Повна абстракція серверів від розробника, тобто є можливість працювати в будь-якому місці і в будь-який час. Хмарне програмне забезпечення дозволяє співробітникам працювати з дому. Крім того, хмарні інструменти полегшують співпрацю з людьми, які перебувають на великій відстані, тобто береться до уваги платежі на основі витрат та виконання, а не розміри екземплярів сервера. Також до переваг варто віднести те, що автор програми не буде перейматися про адміністрування серверів на яких запущені в роботу його додатки.

В процесі еволюції безсерверних обчислень з'являється гіпервізор, таким чином формується віртуалізація [3]. Розглянемо метод віртуалізації, його концепція полягає в тому, що ядро операційної системи може підтримувати не один ізольований примірник простору користувача, а декілька в залежності від запитів користувача. Примірники носять назву контейнери або зони, їх можна співставити з реальним сервером. Так як програми взяті з різних контейнерів, це забезпечує повну незалежність в їх роботі і виключає можливість впливу одна на одну.

Дослідники напрямку інформаційних технологій, детально вивчають особливості роботи контейнерів, так як має місце, на відміну від гіпервізорів, інший метод розподілу ресурсів, оскільки контейнерні двигуни, наприклад Docker [7], можуть обертати одиниці обчислення за лічені секунди. Такі інноваційні зміни в роботі мікросервісів надають ряд переваг та функціональних можливостей при роботі з великими масивами даних.

FaaS (*Function as a Service*) можна означити, як еволюцію контейнерів. Розглянемо детально основні етапи використання вказаної вище безсерверної технології. Якщо уявити, що вже існує кілька десятків контейнерів, які вже встановлені, наприклад, Python, Java або NodeJS, але без розгорнутих в них мікросервісів, які потрібно виконати в певному тимчасовому режимі. Принцип роботи полягає в тому, що при виникненні події, побудові додатку або виклику API, двигун FaaS завантажує код мікросервісу, обробляє код і вимикає контейнер. Можливий алгоритм, коли контейнер з кодом в ньому продовжує бути ввімкненим, для щоб на наступний виклик швидше реагувати, це значно оптимізує роботу над певною подією.

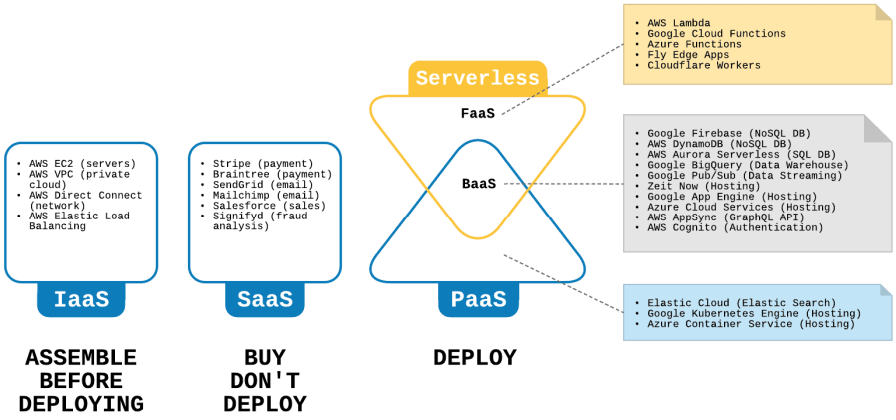
Таким чином, можна стверджувати, що Function as a service – архітектурний шаблон, який передбачає можливість виклику примірника керуючого коду без необхідності управління серверами і серверним додатком; ключовий компонент безсерверних обчислень (англ. serverless computing) [1].

Ідея полягає в тому, що серверна частина розбивається на набір незалежних від стану (stateless функцій для обробки подій (якими можуть бути приходять HTTP-запити, поява нових повідомлень в черзі, наступ зазначеного в планувальнику часу і тому подібні події). Стан у даному випадку розуміється в контексті розгортання (deployment), тобто результат виконання функції не повинен залежати від стану пам'яті сервера (за вирахуванням переданих параметрів виклику) і вмісту локальної файлової системи. При таких обмеженнях, масштабування виконується автоматично провайдером, який може використовувати будь-який доступний йому в даний момент обчислювальний пристрій, що забезпечує необхідний рівень продуктивності. Також, як правило, час виконання функції примусово обмежується кількома хвилинами.

Головним завданням є правильна ієрархія при розробці програми в основі якої лежатиме така модель, адже таким чином користувачі мають змогу використовувати безсерверну архітектуру, яка є фундаментом при розробці мікросерверних розподілених додатків, де кожен компонент є ізольованим від впливу інших та виконує запрограмовану функцію. Використання схем робочих потоків дозволяє координувати роботу мікросервісів може, основаних на оркеструванні контейнерів і налагодженні API Gateway з метою співпраці сервісів.

Навіть з точки зору екології є вагомі переваги, оскільки величезні центри зберігання інформації вимагають багато електроенергії, однак вони: 1) мають високий рівень незалежності; 2) тривіальні в заміні однієї реалізації сервісу іншою: 3) організовані відносно бізнес логіки яку вони виконують; 4) можуть незалежно від інших кожен сервіс реалізувати за допомогою будь-якої мови програмування; 5) архітектурно побудовані за симетричним принципом (виробник-споживач).

Реалізація перерахованих вище можливостей засобів нових інформаційних технологій дозволяє організувати такі види діяльності як: 1) робота з даними представленими в великих обсягах; акумулювання інформації стосовно досліджуваних об'єктів та передача інформації, яка може бути представлена у різних формах; 2) продуктивна співпраця користувача з програмною (програмно-апаратною) системою, яку можна охарактеризувати тим що має місце обмін



Ілюстрація 2: Скорочене описання різниці між IaaS, SaaS, PaaS, BaaS, FaaS та Serverless [9]

текстовими командами (запитами) і відповідями (запрошеннями), широкий спектр засобів ведення діалогу; представлення альтернативи вибору варіантів змісту, режиму роботи; 3) можливість керування реальними об'єктами; 4) візуалізація даних та управління відображенням на екрані моделей різних об'єктів, явищ, процесів, навіть тих що реально відбуваються в часі [4]. Для обслуговування та підтримки роботи безсерверних технологій необхідно мати апаратне забезпечення і цей термін не варто розуміти буквально. Ця назва використовується тому, що користувачам такої платформи не треба мати справу з налаштуванням серверів для запуску коду: усі серверні налаштування, планування обчислювальних ресурсів цілком приховані від користувачів і керуються платформою. Безсерверний код може бути частиною застосунка побудованого на традиційній архітектурі, наприклад, на мікросервісах.

Розглянемо вид сервісу, який носить назву «безсерверні обчислення» (Serverless computing) своєю особливість має те, що, хоча користувач і може розгортати таким чином своє ПЗ і може вносити деякі зміни у конфігурацію розгортання – користувач не може обрати ні віртуальну машину, ні оточення, тощо. Користувач може лише розгорнути своє ПЗ і вказати ситуації, у яких воно може бути запущене (наприклад за таймером), все інше бере на себе постачальник хмарних сервісів. Це дозволяє ще сильніше спростити розроблення й розгортання ПЗ з використанням хмарних сервісів.

Використовуючи хмарне програмне забезпечення більша частина роботи яка пов'язана із опрацюванням даних перекладається на центр обробки даних, який розташований в іншому місці.Тобто користувач бачить тільки результати роботи на власному ПК.Можна із впевненістю стверджувати, що «хмарні обчислення» є ніби онлайн альтернативою тим процесам, які зазвичай організації виконують самостійно. Хмара може означати як програмне забезпечення, так і інфраструктуру. З точки зору досконалості технології, програмне забезпечення в хмарах розвинуте значно краще, ніж апаратна складова. Варто відмітити, що хмарні технології забезпечують високий рівень обслуговування споживачів різних сфер діяльності.На сьогодні, існує безліч додатків та сервісів у вільному доступі через вікно будь-якого браузера, за наявності підключення до мережі Інтернет. Для прикладу розглянемо Google Apps – служби, надавані компанією Google. Головним є наявність доменного ім'я для цього потрібно зареєструватися через реєстратора, авторизованого компанією Google. Google Apps суміщає в собі безкоштовний базовий та професійний пакети. Для роботи з конструктором сайтів представлено сервіс сайти Google де можливо працювати з відео, зображеннями, документами. Мета сервісу –організувати єдиний інтернет-простір, де користувачі будуть ділитися інформацією.

Повсякденний ритм життя вимагає від користувачів зтак званого «онлайн режиму» для цього можливе використання хмарних сервісів через спеціальний мобільний додаток або браузер на смартфоні та планшеті. Хмарні сервіси не вимагають складного обладнання, яке працює на спеціалізованому програмному забезпеченні. Для прикладу, веб-сервіс пошти – це онлайн-альтернатива хостингу серверу електронної пошти. Більшість послуг «хмарних обчислень» можна отримати через веб-браузери: Microsoft Internet Explorer, Microsoft Edge, Mozilla Firefox або Google Chrome. Служба сайти Google дозволяє додавати на сайт найрізноманітнішу інформацію:

календарі, відео, зображення, документи й ін.; визначати параметри доступу до сайту. Легко зрозуміти, що сервіси Google завдяки хмарам дозволяють уникнути багатьох проблем і витрат, пов'язаних з обслуговуванням програмного й апаратного забезпечення.

Хмарні технології – це організація роботи, при якій все обчислювальне навантаження припадає на сервери. У хмарних обчисленнях звичайно виділяють три окремі категорії або рівня: 1. «Інфраструктура як послуга» (IaaS, infrastructure as a service). На цьому рівні користувачі одержують базові обчислювальні ресурси – наприклад, процесори й пристрої для зберігання інформації – і використовують їх для створення своїх власних операційних систем і додатків. Перевагою такого підходу є гнучкість конфігурації. Недоліки: складність конфігурації й ціна. 2. «Платформа як послуга» (PaaS, platform as a service). Тут користувачі мають можливість установлювати власні додатки на платформі, що надається провайдером послуги. Користувач може внести мінімум змін у їхню конфігурацію, він не має прямого доступу до віртуальних машин, де вони розгорнуті. Перевагами такого підходу (у порівнянні з IaaS) є спрощення розгортання й конфігурації, а також менша ціна. Недоліки: нижча гнучкість конфігурації, потенційно більше ризиків у безпеці системи. 3. «Програмне забезпечення як послуга» (SaaS, software as a service). У цьому типі користувач має виділені йому CRM, CMS, тощо. Користувач не має доступу до налаштувань веб-серверів, СКБД і, тим більше, ОС. Найшвидший і найпростіший у розгортанні тип сервісу. Але тут користувач майже немає можливостей до змінення конфігурації, налаштування безпеки сервісу, тощо. На цьому рівні в «хмарі» зберігаються не тільки дані, але й пов'язані з ними додатки, а користувачеві для роботи потрібно тільки веб-браузер [5].

Основна частина програмного забезпечення при роботі легко поєднується з моделью SaaS, відмітимо найпоширеніші – Google Apps Education Edition і Microsoft Live@edu, адже з їх використанням можливі різносторонні послуги, і як засоби підтримки комунікації, так і як офісні додатки (електронна пошта й електронні таблиці).Зокрема мова йде про хмарні технології керування проектами. Хмарна технологія керування проектами Clarizen надає можливість зрозуміти в повній мірі, що таке проект, його значимість, основні етапи, застосування на практиці. Це онлайн-система керування проектами, що являє собою проектно- орієнтований робочий простір для ведення одного або декількох проектів, доступна всім учасникам через Інтернет. При виникненні апаратних проблем в роботі користувача на кожному сервері є резервний жорсткий диск, яким можна замінити диска, що вийшов з ладу, в складі массиву RAID. Заміну диску повинен проводити висококваліфікований спеціаліст, для того щоб уникнути повного припинення роботи сервера. Якщо все ж сервер остаточно вийшов з ладу то для відновлення потрібно використати резервну копію та поетапно виконати в ручному режимі план аварійного відновлення.

Проведемо аналіз і порівняння можливостей найбільш поширених на сьогоднішній день платформ хмарних обчислень: Amazon Web Services, Google App Engine і Microsoft Windows Azure. Для наочності та кращого сприйняття наведеного матеріалу, представимо характеристичні властивості платформ у вигляді таблиці

Таблиця 1 – Порівняння можливостей платформ хмарних обчислень: Amazon Web Services, Google App Engine і Microsoft Windows Azure

Назва платформи	**Інтерфейс доступу користувача**	**Віртуалізація**
Google App Engine	Web-консоль адміністрування	Контейнер додатків Docker
Microsoft Windows Azure	Портал Microsoft Windows Azure	Рівня ОС з Hyper-V-virtualized multitenant compute (Virtual Machines)
Amazon Web Services	Утиліти консолі Amazon	Рівня операційної системи (ОС), з гіпервізором Xen

**сформовано автором за джерелами [1;2; 4; 5]*

По типу всі платформи однакові — IaaS, PaaS та мають підтримку Web API, а також схожі за мовами програмування: Java, Node.js, Ruby, Python and PHP. Відмінне за останнім критерієм лише те, що Google App Engine підтримує C# та GO, а Microsoft Windows Azure та Amazon Web Services - Microsoft .Net.

Підсумовуючи вище сказане, відмітимо, що хмарні технології забезпечують високий рівень обслуговування споживачів. Користувачі одержали можливість спільно працювати з документами, створювати окремі сайти для робочих груп, управляти проектами, обмінюватися миттєвими

повідомленнями, організовувати вебконференції, користуватися електронною поштою з розширеним функціоналом та ін. Є можливість установити локальні версії програм на своїх власних пристроях – ноутбуках, планшетах, смартфонах – і залишатися на зв'язку, спілкуватися й учитися де завгодно. Виходячи з цього можна із впевненістю стверджувати, що при виборі платформи необхідно враховувати бюджет програмного проекту, призначення, технології розробки.

Висновки. Залишаючись монолітним в сучасному світі, важко змінюватись разом з потребами примхливого ринку. Ось чому створення програмного забезпечення у вигляді мікросервісів шалено набирає популярність на противагу монолітному підходу. Детальний аналіз наукових досягнень в даному напрямку, дав змогу систематизувати отримані, на даний час, новітні результати та узагальнити конкретні рекомендації з удосконалення технологій та методів, які активно застосовуються для подальшої розробки безсерверної технології Functions as a Service для створення хмарних мікросервісних додатків.

1. Богданов, А. В. Сравнение нескольких платформ облачных вычислений. – Киев: Академпериодика, 2016. – 472 с.
2. Эммерих В. Конструирование распределенных объектов. Методы и средства программирования интероперабельных объектов в архитектурах OMG/CORBA, Microsoft/COM и Java/RMI / Пер. с англ. – М.: Мир, 2012. – 510 с.
3. Радченко, Г.И. Распределенные вычислительные системы /Г.И. Радченко. – Челябинск:Фотохудожник, 2012. – 184 с.
4. Маклаков С.В. BPwin ERwin CASE-средства разработки информационных систем. – М.: Диалог МИФИ, 2011. – 304 с.
5. Пасічник В.В. Глобальні інформаційні системи та технології: моделі ефективного аналізу,опрацювання та захисту даних. Монографія / В.В.Пасічник, П. І. Жежнич, Р. Б. Кравець, А. М.
6. Пелещишин, Д. О. Тарасов – Львів: Видавництво Львівської політехніки, 2017. – 348 с. ISBN: 966-553-578-1.
7. Plummer D. C. Cloud Computing Confusion Leads to Opportunity / Daryl C. Plummer, David W. Cearley, David Mitchell Smith – Report № G00159034. – Gartner Group,2017 – [Electronic resourse]. – Access mode: http://www.gartner.com/it/content/868800
8. https://www.docker.com
9. 8. Himanshu Pant. A brief history of serverless (or, how I learned to stop worrying and start loving the cloud) ... – [Electronic resourse]. – Access mode: https://medium.freecodecamp.org/a-brief-history-of-serverless-or-how-i-learned-to-stop-worrying-and-start-loving-the-cloud-7e2fc633310d
10. 9. Nicolas Dao. THE SERVERLESS SERIES—What Is Serverless?... – [Electronic resourse]. – Access mode: https://hackernoon.com/the-serverless-series-what-is-serverless-d651fbacf3f4