УДК 004.9:33:62 DOI: 10.33111/mise.97.15

Остапович Т. В.,
аспірант кафедри інформаційних систем в економіці,
Київський Національний Економічний Університет
імені Вадима Гетьмана

Ostapovich T.V.,
Postgraduate Student of the Information System in Economic Department,
Kyiv National Economic University named after Vadim Hetman

ПРОЕКТУВАННЯ ІНФРАСТРУКТУРИ ІННОВАЦІЙНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КОМЕРЦІЙНОГО БАНКУ

THE DESIGN OF INFRASTRUCTURE OF INNOVATIVE COMMERCIAL BANK SOFTWARE

Анотація. *Стаття присвячена розробці архітектури програмного забезпечення. Головним завданням запропонованої архітектури є забезпечення можливості роботи пересувного банківського відділення. Технічна можливість зменшити кількість стаціонарних відділень банку існує вже сьогодні, це зручно для користувачів. В міру збільшення вартості утримання банківського відділення та збільшення концентрації населення у великих містах, актуальним постало питання діяльності пересувного банківського відділення. Прикладом реалізації можуть бути пересувні поштові відділення, які вже активно діють починаючи з 2019 року. З початком діяльності пересувного банківського відділення не потрібно буде їздити до відділення банку, яке часто знаходиться за десятки кілометрів. Для банків не потрібно буде утримувати стаціонарні відділення, які з кожним роком обслуговують усе менш кількість користувачів. Для вирішення такого питання є можливість практичного застосування такої інновації — як робота пересувного банківського відділення без підключення до інтернету. Розподілення навантаження від діяльності пересувних банківських відділень у статті запропоновано здійснити за рахунок використання Kubernetes, а саме сервісу, який запропонований на AWS. Використання сервісів AWS є хорошим тоном сучасної інфраструктури, так як існує можливість зменшити витрати та ефективніше використовувати обчислювальні можливості для вирішення робочих завдань. У статті описано переваги AWS і здійснено порівняння з іншими популярними хостинг провайдерами Digital Ocean та Heroku. За умови вибору AWS існує можливість використовувати штучний інтелект для прогнозування навантаження на сервер з урахуванням днів тижня та робочих годин функціонування банківських відділень. У статті також описано можливості використання AmazonMQ з метою зменшення навантаження на сервер бази даних.*
Ключові слова: *Kubernetes, банківські системи, база знань, EKS, AI.*

Abstract. *The article is devoted to the development of software architecture. The main objective of the proposed architecture is to provide mobile banking. The technical ability to reduce the number of stationary branches of the bank already exists today, it is convenient for users. As the cost of maintaining a bank branch and increasing the concentration of the population in large cities increased, the question of the activity of a mobile bank branch became urgent.*

An example would be mobile post offices, which are already active since 2019. With the start of the mobile banking branch, you will not need to go to the bank branch, which is often within tens of kilometers. Banks will not need to maintain fixed offices that serve a growing number of users every year. To address this issue, there is the possibility of the practical application of such an innovation — such as the work of a mobile banking branch without an Internet connection. The distribution of the load from the activity of mobile banking branches in the article is proposed to be made through the use of Kubernetes, namely the service offered by AWS. The use of AWS services is a good fit for today's infrastructure, as it is possible to reduce costs and use computing capabilities more efficiently to solve work problems. The article describes the benefits of AWS and compares it to other popular hosting providers Digital Ocean and Heroku. If AWS is selected, it is possible to use artificial intelligence to predict server load based on the days of the week and the operating hours of the branch offices. The article also describes how to use AmazonMQ to reduce the load on your database server.
Key words: *Kubernetes, banking systems, knowledge base, EKS, AI.*

Вступ. Стрімкий розвиток інформаційних технологій у 21-ому столітті, встановлює нові вимоги та потреби до програмного забезпечення та інфраструктури програмного забезпечення. Так, для прикладу, звичайний веб-сайт, який виконує функцію презентації компанії в Інтернеті, не має визначених меж. Не можна визначити межу, де сайт починається і де він завершується. Від технологій хмарних обчислень і до найновіших смартфонів з використанням штучного інтелекту, всі складові такого рішення матимуть вимір взаємодії з кінцевим користувачем і наскрізного використання технологій з урахуванням обмежень кожного елемента.

Поєднання всіх доступних нам технологій і покладено в основу проектування архітектури інноваційного програмного забезпечення. Надалі мова піде про те, як закласти вирішення ключових проблемних питаннь на етапі проектування архітектури програмного забезпечення. Так як в основі запропонованого архітектурного рішення буде сучасна інфраструктура — ми повинні визначити головні вимоги, щоб в подальшому краще використовувати наявні обчислювальні можливості.

Мета статті: аналіз та використання Kubernetes в банківських системах.

Виклад основного змісту. Для розгортання інфрастуркутури в 2020 році найчастіше обирають AWS (Heroku), Digital Ocean та Azure. Для кожного конкретного рішення — доцільно буде використати одну із перерахованих платформ. Для проектування пропонується обрати систему банку, так як вся розробка тепер є веб-розробкою — кінцевий результат це веб-сайт для працівників відділення банку. Відділення банку може бути виїзне, для прикладу на територіях близьких до лінії фронту, де немає мож-

ливості зробити повноцінне Інтернет-покриття. За допомогою використання програмних засобів є можливість обійти вказану проблему. Таким чином, ми маємо максимально складну для проектування систему з п'ятидесяти відділень без постійного підключення мережі Інтернет. Функціонування відділення полягає в здійснення визначеного числа операцій. Одним з видів операцій є готівкові операції, в більшості випадків буде здійснюватися видача пенсій, а також оплата комунальних платежів. У роботі банківського відділення запропоновано використовувати касовий апарат з вбудованою базою даних для того, щоб без підключеного Інтернету здійснити готівкові операції. У подальшому касовий апарат буде синхронізуватися з програмою банку, інноваційним у даному випадку буде використання бази даних безпосередньо на самому касовому апараті. Для цього необхідно створити структуру бази даних і нормалізувати таблиці, та мати актуальну інформацію. Наявність такої бази даних дає можливість використовувати касовий апарат без підключеного Інтернету в банківському відділенні, в тому числі пересувному. Синхронізація відбувається в момент проходження контрольно-пропускного пункту, для прикладу на лінії розмежування, це економія з точки зору розгортання інфраструктури. В момент синхронізації касовий апарат з'єднується з програмним забезпеченням банку у вигляді веб-сайту і нам саме час розглянути на наступний рівень, а саме веб-сайт. Як би це не звучало банально, але під веб-сайтом ми розуміємо комплексне програмне рішення з усіма його компонентами. До таких компонентів зокрема відноситься інтерфейс, який ще називають фронтендом, та адмінпанелль, де можна вносити зміни в фронтенд. Програмну логіку зазвичай реалізують на бекенді та додають так званий API (Прикладни́й програ́мний інтерфе́йс), це саме та частина де здійснюються обчислення. Обов'язковим компонентом є база даних (це може бути PostgreSQL, MSSQL, MongoDB), в якій зберігається інформація для роботи API. Для пришвидшення роботи бази даних використовують скорочену базу даних (Redis), або як її прийнято називати словник значень — це менша та швидша база даних яка дає можливість використати наявну в головній базі даних інформацію без мережевих затримок та використовувати збережену в Redis інформацію для обчислень. З метою надійного захисту інформації під час встановлення зв'зку з відділеннями обов'язково використовується VPN (Virtual Private Network). Такий підхід передбачає наявність в інфраструктурі VPN сервера, а на кожному касовому апараті використання VPN клієнта [4].

Детальніше розглянемо можливості запропонованого хостингу та його порівняння. Першим буде представлено Heroku, як найпростіший у використанні PaaS (Platform as a service). Heroku — це хостинг за принципом платформа як сервіс, у даному випадку ми приділяємо увагу написанню коду, а обслуговуванням інфраструктури займається платформа. Таким чином, ми не витрачаємо гроші на системного адміністратора. З наявного досвіду можна сказати, що впродовж останніх років дана платформа стабільно працювала і не було зафіксовано жодного випадку відключення інфраструктури. Головним компонентом Heroku є Dyno — так називається сервіс, який пропонує обчислювальні можливості хостинга, є різні тарифні плани. В межах нашого порівняння можемо вибрати мінімальні тарифні плани, так як в інфраструктурі буде передбачено функціонування в умовах навантаження. Під час збільшення навантаження інфраструктура повинна масштабуватися відповідно до потреб. Наше програмне рішення буде розміщуватися на Dyno. З сервісів, які пропонує платформа, можна вибрати базу даних та інші необхідні для роботи компоненти. Відсутність системних адміністраторів для роботи такого проекту також реальність, ми абстрагуємося від поняття сервер, так як Dyno це в першу чергу обчислювальні можливості, з технічної точки зору це контейнер.

Контейнер собою представляє ізольовану файлову систему, яка створюється із зображення ізольованої файлової системи так званого image. Image можна зберігати в репозиторі,ї який спеціально створений для зберігання image, це може бути docker hub [2], AWS ECR, Digital Ocean repository, також є можливості деплоїти Image напряму. Головна відмінність репозиторія призначеного для зберігання Image — це зберігання стану файлової системи. Репозиторій Image зручно використовувати для деплою (перенесення актуального коду з репозиторія коду на сервер), так Image складається з кількох рівнів файлової системи. У більшості випадків змінюється один або два рівні файлової системи Image, тому немає потреби кожен раз завантажувати Image повністю. Достатньо буде завантажувати ті рівні файлової системи, в яких є відмінність. Варто також зазначити, що Heroku має особливості в порівнянні з іншими хостинг провайдерами. Такою особливістю є білдпак — це спеціальна програма, за допомогою якої складається робоча версія програмного забезпечення. Інноваційність використання платформи Heroku на цьому не завершується тому, що є можливість автоматично вибирати білдпак з переліку наявних. Відбувається це за рахунок поперед-

нього аналізу нашого програмного рішення з подальшою спробою підібрати найоптимальнішу версію білдпаку. Явним недоліком є той факт, що Heroku використовує ресурси AWS, тому ціни на послуги матимуть більшу вартість [5]. Також слід зазначити, що Heroku передбачає використання динамічної ір-адреси та системи DNS-targets з подальшим використанням DNS записів типу CNAME. Відсутність публічно доступної, статичної ір-адреси, може бути проблемним моментом. У такому випадку доведеться додатково маршрутизувати трафік з відділення до головного офісу і потім до сервера. Зважаючи на такі обмеження, варто розглянути можливість використання хостингу Digital Ocean.

Головною перевагою використання хостингу Digital Ocean буде наявність виділеної, публічно доступної ір-адреси. На відміну від Heroku мова буде йти про віртуальну машину. Digital Ocean передбачає також присутність системного адміністратора, так як керувати інфраструктурою Digital Ocean потрібно вручну. Більшість завдань можна автоматизувати, але присутність у штаті співробітників системного адміністратора є обов'язковою. В даному випадку питання технологічного відставання Digital Ocean все ж таки поступово вирішується. Пропонуються сервіси, які дають можливість не думати про присутність системного адміністратора на постійній основі. Влітку 2019 року було запропоновано сервіс бази даних і сервіс Kubernetes на Digital Ocean. Kubernetes — це кластер пов'язаних між собою серверів [1]. Мастер нода є центром керування кластером, slave ноди представляють собою робочі одиниці сервісу Kubernetes. Сервіс складається з нод, на яких розміщується поди. Поди представляють собою одну копію запущеного аплікейшина в якому може розміщувати один працюючий контейнер. Також обов'язковим є присутність деплоймента. В межах Kubernetes, деплоймент — це опис процесу розгортання подів. Наступним елементом є сервіс. Сервіс — це набір інструкцій, які перевіряють роботу деплойментів і дають змогу до них звертатися. Найцікавіший момент — це наявність інгресу. За допомогою інгресу можна прив'язати публічно доступну ір-адресу до мережевого інтерфейсу в середині кластера. Наявність публічно доступної ір-адреси зробить аплікейшин доступним у мережі Інтернет. Таким чином Kubernetes дає можливість автоматизовано керувати великою кількістю серверів і абстрагуватися від роботи інфраструктури програмного забезпечення. Наявність ір-адреси в Digital Ocean дає можливість використовувати віртуальну приватну мережу. VPC (з анг. Virtual Private Cloud — віртуальна приватна мережа)

буде надійно захищати інформації канали зв'язку між головним відділенням і сервером [7].

Наступним для порівняння пропонується AWS і сервіси на його основі: EC2, ECR, s3, RDS, Elastic Cache, EKS, Autoscale Group, Spot request. В AWS обчислювальні ресурси називають EC2 — це віртуальні сервери. ElastickBenstalk. ElastickBenstalk — це сервіс, який дає можливість використовувати платформу як сервіс з подальшою організацією процесу деплою з репозиторію у вигляді архіву, який зберігається на s3 bucket. S3 bucket — сервіс для зберігання інформації на AWS. ElastickBenstalk розгортає архів, визначає його структуру і здійснює білд Image. Якщо Image успішно збілдився ElastickBenstalk запускає контейнер і сайт доступний в Інтернеті, цим процесом повністю керує ElastickBenstalk. Таким чином, в один момент часу в нас є включена одна, попередньо перевірена, версія програмного забезпечення. Якщо найновіша версія виявилась неробочою, ElastickBenstalk автоматично повертається до останньої робочої версії, проте має обмежений функціонал, а саме відсутність можливості використовувати репозиторій з Image. На AWS репозиторій зберігання Image представлений сервісом ECR, його зручно використовувати в поєднанні з EKS (це сервіс Kubernetes на основі обчислювальних можливостей AWS) (рис. 1).

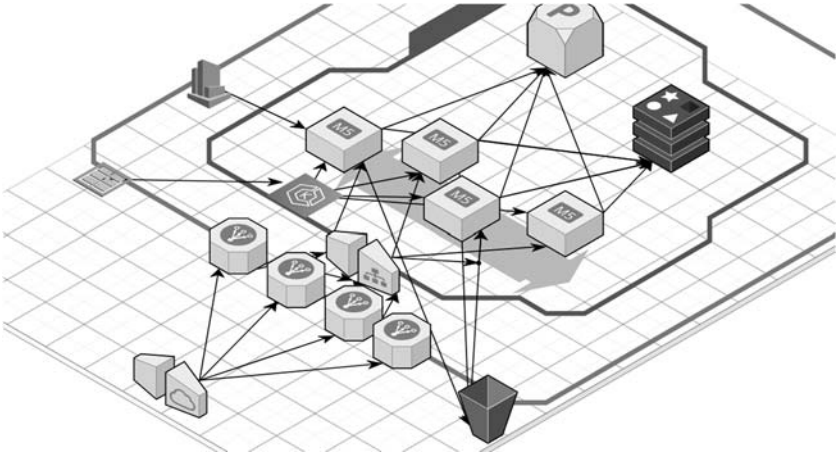

Рис. 1. Приклад інфраструктури EKS на AWS

В грудні 2019 року було презентовано сервіс Fargate. Призначення сервісу Fargate — забезпечувати обчислювальні ресурси

кластера на основі потреб HPA [8]. Horisontal Pod Autoscaler — це моніторинг необхідної кількості подів на основі ресурсів які використовує под [9]. Якщо ресурси наявних подів завершуються — EKS включає додаткові поди, а Fargate включає додаткові обчислювальні ресурси. Такий підхід забезпечує безвідмовну роботу аплікейшина. Якщо поглянути на принципи роботи великих компаній, то саме такий спосіб керування інфраструктурою лежить в основі хостингу аплікейшина, яким користується велика кількість відвідувачів. Національна банківська мережа є великим проектом, тому доцільно буде використовувати хостинг на основі сервісу EKS. EKS складається з різних компонентів і пропонується як окремий сервіс. На відміну від Kubernetes Digital Ocean, який пропонується як готове рішення, EKS потрібно налаштовувати вручну. CloudFormation є невід'ємною складовою EKS. CloudFormation дає можливість створювати кластер і включати нодгрупу (сервери одного типу які віддаються в розпорядження kubernetes як ноди). Всі ці та інші операції здійснюються з персонального комп'ютера за допомогою програми AWS CLI. Така можливість стала доступною завдяки тому, що сервіс інженери AWS детально прописали інструкції, за допомогою яких включається та налаштовується інфраструктура.

CloudFormation — це швидкий спосіб автоматизованого налаштування інфраструктури, який наочно демонструє визначення «інфраструктура як код». Сучасна інфраструктура являє собою 80 відсотків коду і тільки 20 відсотків це якісь апаратні рішення на основі яких працює код. Динамічна зміна навантаження передбачає використання різної кількості серверів у різний час. Навіть якщо припустити, що відділення банку працюють цілодобово [6]. Впродовж тижня навантаження коливатиметься залежно від того, яка кількість транзакцій відбувається та скільки пристроїв синхронізується. Для вирішення питання зміни навантаження, ми можемо використати AutoscaleGroup. AutoscaleGroup — це сервіс за допомогою, якого автоматично здійснюється зміна кількості серверів у нодгрупі. Важливими параметрами під час налаштування AutoscaleGroup є мінімальна кількість серверів, максимальна кількість серверів та умови, за яких включається додатковий сервер у нодгрупі. Важливим елементом інфраструктури на AWS є RDS. RDS — сервіс бази даних, який дає можливість використовувати такі переваги: бази даних як сервіс. Перевагою в порівнянні з розміщеним бази даних на сервері є наявність зображення файлової системи Shanshoot. Shanshoot у подальшому можна розгорнути в базу даних.

Такий підхід використання Shanshoot усуває проблеми несумісності версій, втрати бекапу, помилок під час зберігання бекапу. Snapshot є найнадійнішим варіантом створення бекапів. Надійнішим способом створення бекапів є тільки гаряча копія slave серверу бази даних. ElastickCach — сервіс зберігання словника. Словник значень потрібен для швидкого доступу до інформації, яка була попередньо збережена в базі даних. Вся інфраструктура AWS поділена по регіонах, на жаль в Україні немає датацентрів AWS, але найближчий до нас регіон — Франкфурт. У кожному регіоні є кілька зон, в яких розміщуються сервери, це забезпечує відмовостійкість.

Ключовою проблемою проектування інфраструктури такого рівня — буде розподіл навантаження. За умовою було порядку п'ятдесяти відділень і в кожному відділенні в нас у середньому 20 пересувних касових апаратів. Кількість касових апаратів може бути більшою. Можна взяти до уваги той факт, що не всі касові апарати працюють одночасно. Тим не менш ще на етапі проектування має бути передбачено можливість одночасної роботи всіх касових апаратів і можливості збільшення кількості касових апаратів, що можуть одночасно під'єднуватися для синхронізації. Запроектована кількість касових апаратів складатиме 1000 одиниць. Буде відбуватися синхронізація з базою даних 1000 одиниць, в нас буде здійснювати велика кількість одночасних під'єднань до бази. Необхідно продумати на перед як технічно забезпечити роботу такої інфраструктури. Зазвичай на продакшині бази мають можливість здійснювати максимум 500 конекшинів. Для прикладу 500 конекшинів це максимальний показник на Heroku. Виникає закономірне питання: яка доцільність тримати постійно включеним величезний сервер бази даних, який здатний витримати 1000 одночасних конекшинів. Вирішити дану проблему покликана інновація: використання програмного забезпечення з метою розподілити навантаження на інфраструктуру, для прикладу MailBroker. MailBroker являє собою чергу повідомлень та їхнє збереження з можливістю подальшого використання.

У хостингу AWS функціональні можливості MailBroker представлено сервісом AmazonMQ [3]. Кожна транзакція в базу даних може бути записана в повідомлення, дане повідомлення буде збережене в сервісі AmazonMQ і в подальшому оброблене і збережене в базу даних. Наявність такого рішення дає можливість використовувати менш потужну базу даних, яка буде ефективніше використовуватися. В такий спосіб ми усуваємо проблеми кількості ресурсів, які на даний момент підключені до хостингу.

Наступним етапом буде синхронізація з веб-сайтом. Синхронізація передбачає збереження всіх транзакцій в AmazonMQ, і паралельну обробку в міру можливості. Таким чином немає такого осяжного навантаження, яке б зробило веб-сайт для синхронізації недоступним. Всі спроби синхронізації буде записано та оброблено. Також у нагоді буде використання словника значень, так як не всі запити будуть направлятися до бази даних напряму. Інформація буде братися з сервісу ElactickCache, який зменшує мережеві затримки під час передачі інформації в базу даних. Саме наявність MailBroker якісно відрізнятиме архітектуру інноваційного програмного забезпечення.

Відсутність явних обмежень, як то кількість конекшинів до бази напряму, розподілення навантаження між різними сервісами та ефективніше використання наявних обчислювальних потужностей з можливістю машинного навчання — якісно відрізнятиме інноваційне рішення. Використання штучного інтелекту з метою здійснення прогнозів можливого навантаження, дасть можливість заздалегіть підготуватися і включити більше ресурсів до того, як навантаження зросте. В той момент коли використання веб-сайту мінімальне — виключити зайві ресурси. Такий підхід дає можливість зменшити витрати на інфраструктуру і в той самий момент максимально збільшити її функціональність залежно від наших потреб без компромісу на безпеку та надійність такого рішення.

Висновоки. Як підсумок можна зазначити, що технології не роблять нас заможними, натомість вдале використання технологій, ще на етапі проектування архітектури — може зробити дійсно якісні зміни кінцевого результату наших зусиль. На прикладі наданого проекту було продемонстровано використання інфраструктури kubernetes для вирішення питань побудови масштабованої інфраструктури. Така інфраструктура буде масштабуватися залежно від навантаження та необхідності опрацювання більшої кількості інформації у банківських системах.

Література

1. Kubernetes. Електронний ресурс. [Режим доступу:] https://uk.wikipedia.org/wiki/Kubernetes

2. Docker. Електронний ресурс. [Режим доступу:] https://uk.wikipedia.org/wiki/Docker

3. AmazonMQ. Електронний ресурс. [Режим доступу:] https://aws.amazon.com/amazon-mq/?nc1=h_ls&amazon-mq.sort-by=item.additionalFields.postDateTime&amazon-mq.sort-order=desc

4. VPN. Електронний ресурс. [Режим доступу:] https://uk.wikipedia.org/wiki/VPN

5. Операторська платформа надання послуг: Електронне навчальне видання. Конспект лекцій / Л. С. Глоба, О.М. Дяденко, В.Ф. Чердинцева. — К.: НН ІТС НТУУ «КПІ», 2013. — 191 с.

6. Банківська система : підруч. [для студентів, аспірантів, викладачів екон. спец.] / М. І. Крупка, Є. М. Андрущак, І. В. Барилюк та ін. ; за ред. М. І. Крупки ; М-во освіти і науки України, Львів. нац. ун-т ім. І. Франка. — Львів : ЛНУ, 2013. — 554, [2] с. : іл. — Бібліогр.: с. 530–546 (206 назв). — ISBN 978-966-613-813-5

7. Дудикевич В.Б., Хорошко В.О., Микитин Г.В., Банах Р.І., Ребець А.І. Інформцаійна модель безпеки технологій зв'язку // Інформатика та математичніметоди в моделюванні. 2014. Том 4, №2 Ст. 137–148.

8. Fargate AWS. Електронний ресурс. [Режим доступу:] https://aws.amazon.com/fargate/

9. Horizontal Pod Autoscaler. Електронний ресурс. [Режим доступу:] https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/

References

1. Kubernetes. Elektronnyi resurs. [Rezhym dostupu:] https://uk.wikipedia.org/wiki/Kubernetes

2. Docker. Elektronnyi resurs. [Rezhym dostupu:] https://uk.wikipedia.org/wiki/Docker

3. AmazonMQ. Elektronnyi resurs. [Rezhym dostupu:] https://aws.amazon.com/amazon-mq/?nc1=h_ls&amazon-mq.sort-by=item.additionalFields.postDateTime&amazon-mq.sort-order=desc

4. VPN. Elektronnyi resurs. [Rezhym dostupu:] https://uk.wikipedia.org/wiki/VPN

5. Operatorska platforma nadannia posluh: Elektronne navchalne vydannia. Konspekt lektsii / L. S. Hloba, O.M. Diadenko, V.F. Cherdyntseva. — K.: NN ITS NTUU «KPI», 2013. — 191 s.

6. Bankivska systema : pidruch. [dlia studentiv, aspirantiv, vykladachiv ekon. spets.] / M. I. Krupka, Ye. M. Andrushchak, I. V. Baryliuk ta in. ; za red. M. I. Krupky ; M-vo osvity i nauky Ukrainy, Lviv. nats. un-t im. I. Franka. — Lviv : LNU, 2013. — 554, [2] s. : il. — Bibliohr.: s. 530-546 (206 nazv). — ISBN 978-966-613-813-5

7. Dudykevych V.B., Khoroshko V.O., Mykytyn H.V., Banakh R.I., Rebets A.I. Informtsaiina model bezpeky tekhnolohii zviazku // Informatyka ta matematychnimetody v modeliuvanni. 2014. Tom 4, №2 St. 137–148.

8. Fargate AWS. Elektronnyi resurs. [Rezhym dostupu:] https://aws.amazon.com/fargate/

9. Horizontal Pod Autoscaler. Elektronnyi resurs. [Rezhym dostupu:] https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/

Статтю подано до редакції 14.03.2019 р.