Хмельницький національний університет
Факультет інформаційних технологій
Кафедра інженерії програмного забезпечення

ДИПЛОМНА РОБОТА

Метод та програмні засоби моніторингу адміністрування хмарних сервісів

Рівень вищої освіти Другий (магістерський)

Галузь знань 12 «Інформаційні технології»

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

Шифр ДРІПЗ.170110.01.07.ПЗ

Виконав студент 2 курсу група ІПЗм-21-1 _____ О.В. Максимів
Підпис Ініціали, прізвище

Керівник канд. техн.наук, доцент _____ Ю.В. Форкун
Науковий ступінь, звання Підпис Ініціали, прізвище

Нормоконтролер канд. пед. наук, доцент _____ Н.І. Праворська
Підпис Ініціали, прізвище

До захисту допускаю:
Завідувач кафедри інженерії програмного забезпечення _____ Л. П. Бедратюк
Підпис Ініціали, прізвище

5 грудня 2022 р.

Хмельницький 2022

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій
Кафедра Інженерії програмного забезпечення
Рівень вищої освіти Другий (магістерський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Л. П. Бедратюк
01 09 2022 р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ)

Максимову Олександру Володимировичу
Прізвище, ім'я, по батькові студента

1. Тема проєкту (роботи) Метод та програмні засоби моніторингу адміністрування хмарних сервісів

Керівник проєкту (роботи) Форкун Юрій Вікторович к.т.н., доц.
Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 01.07.2022 р. № 83

2. Строк подання студентом проєкту (роботи) на кафедру 01.12.2021 р.

3. Вихідні дані до проєкту (роботи) Матеріали переддипломної практики, методичні вказівки до виконання дипломної роботи

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)

1. Теоретичні аспекти досліджуваної галузі
2. Аналіз методів монторингу хмарних обчислень
3. Програмна реалізація системи
4. Дослідження та тестування ефективності удосконалення методу

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)

Презентаційні матеріали (слайди)

6. Консультанти розділів дипломного проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Антиплагіат	Гурман І. В., доцент		
Нормоконтроль	Праворська Н.І., доцент		

7. Дата видачі завдання « 01 » липня 2022р.

КАЛЕНДАРНИЙ ПЛАН

Назва етапів (розділів) дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Прим
1 Вивчення предметної області; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження; формування логістичної структури дипломної роботи	01.09-10.09.2022	
2 Робота над розділом 1 дипломної роботи – вивчення літературних та Інтернет-джерел; аналіз відомих моделей, методів та засобів за темою роботи; визначення методологічних підходів до вирішення задачі; висновки до розділу та постановка задач дослідження	11.09-25.09.2022	
3 Робота над розділом 2 дипломної роботи – розробка моделей, методів та алгоритмів вирішення задачі; висновки до розділу	26.09-10.10.2022	
4 Робота над науковими статтями	11.10-30.10.2022	
5 Робота над розділом 3 дипломної роботи – розробка інформаційної технології вирішення задачі (аналіз вимог до програмного засобу та його проектування, аналіз та вибір засобів реалізації програмного засобу тощо); висновки до розділу	11.10-26.10.2022	
7 Попередній захист дипломної роботи	Листопад (згідно графіка)	
8 Узгодження постановки задачі, отриманих результатів та висновків; написання вступу, загальних висновків, оформлення джерел посилання та додатків; оформлення пояснювальної записки та графічних матеріалів згідно вимог чинних стандартів	18.12-30.11.2022	
9 Перевірка роботи на наявність плагіату; нормоконтроль; брошурування пояснювальної записки; підготовка супровідних документів	01.12-04.12.2022	
10 Підготовка до захисту дипломної роботи	з 01.12.2022 р	

Студент _____ Маханьків О. В.
Підпис Ініціали, прізвище

Керівник проекту (роботи) _____ Корнага О.В.
Підпис Ініціали, прізвище

РЕФЕРАТ

Тема дипломної роботи: «Метод та програмні засоби моніторингу адміністрування хмарних сервісів».

Автор роботи: Максимів Олександр Володимирович.

Керівник роботи: Форкун Юрій Вікторович.

Пояснювальна записка: 80 с., 11 рис., 4 табл., 6 дод., 29 джерел. 80 p., 11 pc., 4 tb., 6 add., 29 srs.

МЕТОДИ МОНІТОРИНГУ, ПРОГРАМНІ ЗАСОБИ, МОНІТОРИНГ, ХМАРНІ СЕРВІСИ, ХМАРНІ ОБЧИСЛЕННЯ, АДМІНІСТРУВАННЯ

Об'єкт дослідження — процеси моніторингу хмарних сервісів та забезпечення їх адміністрування.

Мета дослідження — удосконалення методу який використовується для моніторингу хмарних ресурсів та розробка і впровадження програмного комплексу на базі вдосконаленого методу. Це дозволить проводити моніторинг та аналіз використання хмарних технологій та хмарних сервісів співробітниками та відділами організації.

У поданій роботі використано такі методи дослідження і технічні засоби:

- спостереження, аналіз, синтез, доведення, формалізація та експеримент;
- комп'ютерні засоби проектування та розробки, програмування, налагодження і тестування;
- персональні комп'ютери, сервери, хмарні технології.

В ході роботи над кваліфікаційною роботою було досліджено галузь аналізу та моніторингу хмарних сервісів, а також сучасні методи та засоби моніторингу використання ресурсів хмарних сервісів та їх використання користувачами. В ході роботи було окреслено можливі напрямки вдосконалення існуючих методів та підходів до їх покращення і вдосконалення. З огляду на визначені підходи і невирішені задачі в цій області було запропоновано покращений метод моніторингу хмарних сервісів з допомогою

інформаційно панелі, наведено алгоритм його роботи, архітектуру системи та сам програмний комплекс.

При реалізації програмного комплексу використано серверну операційну систему Linux, мову C, мову серверного програмування Bash та мови програмування веб-сайтів PHP, HTML, JavaScript.

Загалом, у даному досліджені нами розглянуто різні підходи до розробки інформаційної панелі для моніторингу хмарних ресурсів. Зокрема, розглянуто різні програмні засоби моніторингу адміністрування хмарних сервісів і пов'язані з ним роботи з різними інформаційними панелями для віртуальних машин в середовищі хмарних обчислень. Здійснено аналіз методів та програмних засобів, що використовуються при розробці інформаційної панелі, та запропоновано внутрішню архітектуру для інформаційної панелі моніторингу хмарних сервісів.

Підпис

02.12.2021
Дата

ABSTRACT

Master's thesis: «The method and software tools for monitoring the administration of cloud services».

Author: Maksymiv Oleksandr.

Head of research: ForkunYuriy.

Master's thesis consists of: 85 p., 17 pc., 2 tb., 6 add., 21 srs.

MONITORING METHODS, SOFTWARE, MONITORING, CLOUD SERVICES, CLOUD COMPUTING, ADMINISTRATION

The object of the study is the processes of monitoring cloud services and ensuring their administration.

The purpose of the research is to improve the method used for monitoring cloud resources and to develop and implement a software complex based on the improved method. This will allow monitoring and analysis of the use of cloud technologies and cloud services by employees and departments of the organization.

The following research methods and technical means were used in the submitted work:

– observation, analysis, synthesis, proof, formalization and experiment;

– computer tools design and development, programming, debugging and testing;

– personal computers, servers, cloud technologies.

In the course of work on the qualification work, the field of analysis and monitoring of cloud services was investigated, as well as modern methods and means of monitoring the use of cloud services resources and their use by users. During the work, possible areas of improvement of existing methods and approaches to their improvement and improvement were outlined. In view of the identified approaches and unsolved problems in this area, an improved method of monitoring cloud services with the help of an information panel was proposed, the algorithm of its operation, the system architecture and the software complex itself were given.

The software complex was implemented using the Linux server operating system, the C language, the Bash server programming language, and the website programming languages C, PHP, HTML, and JavaScript.

In general, in this research we considered different approaches to the development of a dashboard for monitoring cloud resources. In particular, various software tools for monitoring the administration of cloud services and related work with various information panels for virtual machines in the cloud computing environment are considered. An analysis of the methods and software tools used in the development of the information panel was carried out, and an internal architecture for the information panel of monitoring cloud services was proposed.

Signature

02.12.2022
Date

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД	–	база даних
ПЗ	–	програмний засіб
ПК	–	персональний комп’ютер
ООП	–	об’єктно-орієнтований підхід
CC		Cloud computing
IT	–	Information Technology
IaaS		Infrastructure as a service
ISO	–	International Organization for Standardization
RP		Resource pooling
PaaS		Platform as a service
SSoD		Self service on demand
Saas		Software as a Service
url		Uniform Resource Locator
VM		Virtual Machine
XaaS		Anything-as-a-service

ВСТУП

У мовах розвитку інформаційного суспільства використання хмарних обчислення та хмарних сервісів – це великий крок у використанні сучасних інформаційних технологій, який набуває все більшого свого розвитку. Загалом, хмарні обчислення, це тип обчислень, в основі якого закладено обмін даними та ресурсами, керування даними та виконання обчислень за допомогою спільних ресурсів. В даний час різні компанії, в різних галузях економіки, енергетики, машинобудування тощо, використовують хмарні обчислення для задоволення потреб, в підтримку власної обчислювальної інфраструктури, а часто навіть і на її заміну.

При використанні ресурсів, та даних, доступних в хмарі, компаніям доводиться постійно контролювати та коригувати їх для визначення навантаження на кожен тип визначеного ресурсу.

Темою магістерської роботи є дослідження та аналіз методів і програмних засобів моніторингу адміністрування хмарних сервісів.

У роботі представлено розробку та вдосконалення методу моніторингу хмарних сервісів та розробку на його основі інформаційної панелі яку можна використовувати для моніторингу хмарних сервісів та хмарних ресурсів, які використовує організація. Використання хмарних обчислень дозволяють організація здійснювати обмін даними та, відповідно, керувати ними, а також здійснювати виконання обчислень на спільних ресурсах з допомогою різноманітних веб-технологій.

Отже, реалізація нового підходу до організації даної інформаційної панелі вимагає визначення основної системи стандартів та певних правил. Вона повинна бути легкою та зрозумілою в використанні і, відповідно, збалансованою. А також, відповідати загальноприйнятим парадигмам якими буде здійснюватися її реалізація.

Окрім того, сама інформаційна панель повинна відповідати ряду вимог до програних систем такого типу, зокрема:

– має бути універсальною - підтримувати роботу з різними типами хмарних обчислень і не обмежуватись тільки певним видом хмарних сервісів;

– здійснювати збір та наліз даних для різного типу хмарних ресурсів, як всього ресурсу так і окремих його складових.

Актуальність нашої теми роботи полягає вдосконаленні методу моніторингу хмарних сервісів та у розробці програмного комплексу, який би дозволяв проводити моніторинг та аналіз використання хмарних сервісів та хмарних обчислень, які використовуються для вирішення певних задач в певній організації.

Об’єкт дослідження – процеси моніторингу хмарних сервісів та забезпечення їх адміністрування.

Мета дослідження – удосконалення методу який використовується для моніторингу хмарних ресурсів та розробка і впровадження програмного комплексу на базі вдосконаленого методу. Це дозволить проводити моніторинг та аналіз використання хмарних технологій та хмарних сервісів співробітниками та відділами організації.

Предметом дослідження виступають методи, механізми та основні принципи моніторингу та аналізу використання хмарних сервісів та хмарних обчислень співробітниками та відділами організації.

Завдання роботи полягають у наступному:

– здійснення теоретичного аналізу методів та засобів моніторингу хмарних обчислень;

– –здійснення аналізу та порівняння програмних засобів моніторингу адміністрування хмарних сервісів;

– проведення підсумків роботи, на основі яких можу бути сформований об’єкт, предмет та завдання для подальших розробок;

– розробки програмного комплексу, який реалізуватиме розроблені алгоритми та розроблений метод;

– провести практичне тестування роботи розробленого програмного комплексу.

У нашій дипломній роботі використано різні методи дослідження та технічні засоби:

– спостереження, аналіз, синтез, доведення, формалізація та експеримент;
– комп’ютерні засоби проектування та розробки, програмування, налагодження і тестування;
– персональні комп’ютери, сервери, хмарні технології.

Наукова новизна отриманих рішень полягає, з огляду на визначені підходи і невирішені задачі в цій області, що нами було вперше запропоновано покращений метод моніторингу хмарних сервісів з допомогою інформаційної панелі, наведено алгоритм його роботи, архітектуру системи та сам програмний комплекс. Крім того, Наукова новизна полягає у тому, що метод моніторингу адміністрування хмарних сервісів отримав свій подальший розвиток, а це дозволило загалом підвищити роботу веб-інформаційної панелі для адміністрування хмарних сервісів та збільшило продуктивність і ефективність роботи програного комплексу та інформаційної системи певної організації загалом.

Практичне значення нашої роботи полягає у застосуванні покращеного розробленого методу адміністрування хмарних технологій при розробці інформаційних панелей такого роду, здійснювати їх реалізацію таким чином, щоб інформаційна панель дозволяла їх користувачам відстежувати, аналізувати та контролювати різні типи ресурсів, зокрема такі параметри як навантаження процесору, використовувану оперативну пам'ять, вільне мсіце на диску тощо. Це надає можливість користувачам аналізувати отриману інформацію про використання різних типів ресурсів та на основі цього приймати певні рішення щодо покращення роботи цих ресурсів, та роботи всієї інформаційної системи організації загалом.

За результатами проведеного дослідження нами опубліковано тези доповіді у Збірнику наукових праць за матеріалами XIV Всеукраїнської науково-практичної конференції «Актуальні проблеми комп’ютерних наук АПКН-2022» та здійснено апробацію при виступі з доповіддю на конференції.

1. ТЕОРЕТИЧНІ АПЕКТИ ДОСЛІДЖУВАНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз предметної області, дослідження невирішених проблем в наявних сучасних дослідженнях

В умовах сьогодення хмарні технологій є досить є досить затребуваною і значимою в світовій економіці. Гостра необхідність організацій при побудові своїх інформаційних систем та середовищ вимагає великих затрат, як часу, матеріальних ресурсів, людських ресурсів та відповідно значних коштів. Використання хмарних технологій, навіть частково, дає змогу на підприємстві покращити рівень використанні сучасних інформаційних технологій та в певній мірі знизити затрати при їх впровадженні та використанні. В свою чергу це дозволяє підвищити рівень роботи самої організації чи підприємства та робить саме підприємство більш гнучким при його функціонуванні та розвиту, що, в свою чергу, збільшує його конкурентну здатність на ринку в порівнянні з конкурентами та дозволяє отримувати більший прибуток при менших затратах самого підприємства.

Загалом хмарні обчислення - це тип обчислень, який покладається на обмін ресурсами, управління даними та виконання різного роду обчислень за допомогою цих спільних ресурсів. На сьогодні різні компанії використовують хмарні обчислення та хмарні сервіси для споживання та задіяння в своїй роботі обчислювальних ресурсів, часто навіть замість того, щоб вибудовувати та підтримувати свою обчислювальну інфраструктуру власними силами.

Проте, як показує практика, багато організацій та підприємств використовують хмарні сервіси та технології поряд з власними інформаційними системами, які тісно інтегровані в їх інформаційні системи, або використовуються окремо для вирішення певного роду задач. Серед таких задач найчастіше використовують ресурси для забезпечення зберігання великих обсягів даних, забезпечення їх цілісності, безпеки, збереження

резервних копій тощо. Також серед хмарних обчислень часто використовуються технічні ресурси (процесори, оперативна пам'ять, фізична пам'ять тощо), для забезпечення потреб організації щодо розв’язання певного роду задач.

Отже, як бачимо, основні питання та невирішені завдання полягають у тому, що для побудови розвинутої інформаційно системи організації, як зазначалось, потрібно використання великої кількості ресурсі та матеріальних і часових затрат. Використання хмарних сервісів та технологій дозволяє в певній мірі це питання вирішити, проте постають питання в реалізації моніторингу хмарних ресурсів. А це, зокрема, не тільки їх контроль за використанням, а побудова складної системи, яка давала б можливість вирішити, які хмарні технології потрібні організації та в яких обсягах. Після вирішення цих питань відкриваються нові завдання, а саме:

– питання яким чином інтегрувати хмарний сервіс чи певний ресурс використовувати;

– питання збору і передачі інформації про використання хмарних сервісів та ресурсів;

– питання моніторингу ресурсу, а саме, яким чином, і що саме потрібно моніторити використовуючи певний хмарний сервіс;

– питання пов’язані з аналізом отриманих даних інформаційною панеллю та їх обробку і збереження.

Основним головним моментом при використанні хмарних ресурсів та хмарних обчислень, доступних в хмарі чи хмарних сервісах, є те що компаніям доводиться постійно контролювати та коригувати їх використання та роботу, для визначення навантаження на кожен тип ресурсу ресурс та їх подальший аналіз і обробку.

Таким чином, використання спеціального програного продукту, як у нашому випадку веб-інформаційної панелі і зокрема інформації про ресурси, які надаються вказаною інформаційною панеллю, допоможе не тільки в

моніторингу, а й аналізі використання хмарних ресурсів, а також в міграції хмарних обчислень, що і є одним і завдань та метою дипломної роботи.

При розробці програного комплексу, для забезпечення його якості нами було враховано, основні підходи до розробки таких систем з урахуванням основного стандарту моделі якості, а саме «Якість програмного забезпечення ISO/IEC 25010:2015) в якому виділено один з елементів моделі якості зокрема зручність супроводження та використання програмного засобу [5]. У таблиці 1.1 наведено детально характеристики якості цього стандарту.

Таблиця 1.1 – характеристики якості стандарту ISO/IEC 25010 зручності використання та супроводу програмного комплексу.

Сумісність	Здатність ПЗ виконувати однакові програми з отриманням таких самих результатів
Модульність	Розділення ПЗ на окремі модулі таким чином, що рівень їх залежності був найменшим при зміні в одному модулі
Надійність	Здатність зберігати протягом певного часу значення визначених параметрів
Захищеність	Здатність до захисту від взлому стороннім ПЗ та їх втручанням в роботу системи
Супроводжуваність	Здатність до покращення та оптимізації програмного забезпечення

Модифікованість	Здатність до зміни без виникнення пошкоджень в роботі, або погіршення роботи функціоналу
Тестованість	Можливість виконання тестових наборів та можливість виконання тестів для перевірки виконання вимог до функціонування

Отже, як бачимо, існує нагальна потреба в розробці та покращені методології розробки програмних комплексів моніторингу хмарних ресурсів. До тепер розроблено багато різних підходів до реалізації таких методи сервісів, проте для кращого забезпечення ефективності та покращення їх роботи доцільно використовувати підхід з зосередженням на забезпечення їх вдалого поєднання та реалізації кращої роботи моніторингу хмарних сервісів.

1.2 Порівняльний аналіз переваг та недоліків існуючих рішень

Для забезпечення моніторингу таких ресурсів найчастіше використовують так звані веб-інформаційні панелі, які дозволяють оперувати даними та ресурсами на сучасних платформах хмарних обчислень. Інформація про дані та ресурси, що надається такою інформаційною панеллю, допомагає в організації та моніторингу хмарних обчислень, що є завданням даного дослідження.

Основне завдання розробки інформаційної панелі – надання користувальницького інтерфейсу, з допомогою якого можна впорядкувати та надати інформацію таким чином, щоб її вона була наглядною та зрозумілою для інтерпретації та аналізу даних. Вона повинна допомагати адміністратору системи, або певному користувачеві взаємодіяти з необхідною інформацією без необхідності переглядати різні веб-ресурси тощо. Інформаційна панель

також надає можливість понизити складність та витрати при роботі з великими обсягами інформації з допомогою складових, які надають інформацію як про важливі події, як то оперативна обробка даних і видача результатів та загальної інформації – історія обчислень, статистика тощо.

При проектуванні таких систем нами розглянуто ряд технологій та методів вирішення поставленого завдання моніторингу хмарних сервісів.

Так, у роботі [1] розглянуто фреймворк для створення інформаційної панелі, суть якого полягає у захоплення всіх елементів комплексного фреймворку управління хмарними рішеннями. Розроблена інформаційна панель надає можливість користувачам керувати віртуальними машинами кожна окремо, або в групами. Вона володіє такими функціями, як можливість користувача зробити знімки однієї віртуальної машини або декількох віртуальних машин та перезавантажувати віртуальні машини з допомогою інструментів інформаційної панелі. Крім того, фреймворк інформаційної панелі також використовується для управління та перевірки рішень.

Інформаційна панель розміщувалась в хмарному сервісі IBM Smart Cloud. Вона забезпечувала перевірку дзвінків, що направлялися від одного абонента до іншого, а потім перевіряла такі виклики на основі знімків зробленими інформаційною панеллю.

Наведений приклад інформаційної панелі підтверджує, що поставлена нами задача відповідає дійсності. Окрім того, ефективність наведеної інформаційної панелі вимірювалася за допомогою відеотрансляцій. На відміну від інформаційної панелі Smart Applications Virtual Infrastructure [2], в нашому проекті відстежується різні обмеження для наданих ресурсів та надається перегляду на різних ресурсах.

У роботі [3] авторами запропоновано розробку інформаційної панелі, яка динамічно отримує доступ до стійкості сервісів хмарних обчислень. Дана інформаційна панель хмарних ресурсів використовується для доступу до загальної роботи сервісів, що розміщені у хмарі. Модель оцінки даної інформаційної панелі базується на чотирьох факторах стійкості: технічних,

економічних, екологічних та соціальних факторах. Основні переваги інформаційної панелі полягають у можливості користувачам вибрати власні фактори стійкості для моніторингу. У роботі представлені лише моделі оцінки на основі перерахованих факторів та наведено загальну архітектуру додатку, однак, не наводиться поглиблена інформація про саму реалізацію інформаційної панелі.

Наведені рішення використовуються для певного виду хмарних сервісів, а інформаційна панель OpenStack не забезпечує моніторинг багатьох сервісів для певного ресурсу та не може бути налаштованим для моніторингу інформації, яка стосується певного ресурсу.

1.3 Постановка задачі реалізації покращення методу моніторингу хмарних сервісів

Реалізація нового підходу до вдосконалення методу моніторингу хмарних сервісів інформаційної панелі вимагає визначення основної системи стандартів та певних правил. Покращення методу має за собою на меті збалансувати навантаження на саму систему доступу та моніторингу хмарних сервісів та надати можливість користувачам оперативно опрацьовувати необхідну їм інформацію. Сама інформаційна панель повинна бути легкою та зрозумілою в використанні і, відповідно, збалансованою. А також вона повинна відповідати загальноприйнятим парадигмам якими буде здійснюватися її реалізація.

Окрім того, сама інформаційна панель повинна відповідати ряду вимог до програних систем такого типу, зокрема:

- має бути універсальною - підтримувати роботу з різними типами хмарних обчислень і не обмежуватись тільки певним видом хмарних сервісів;
- здійснювати збір та наліз даних для різного типу хмарних ресурсів, як всього ресурсу так і окремих його складових.

Актуальність нашої теми роботи полягає вдосконаленні методу моніторингу хмарних сервісів та у розробці програмного комплексу, який би дозволяв проводити моніторинг та аналіз використання хмарних сервісів та хмарних обчислень, які використовуються для вирішення певних задач в певній організації.

Мета дослідження – удосконалення методу який використовується для моніторингу хмарних ресурсів та розробка і впровадження програмного комплексу на базі вдосконаленого методу. Це дозволить проводити моніторинг та аналіз використання хмарних технологій та хмарних сервісів співробітниками та відділами організації.

Предметом дослідження виступають методи, механізми та основні принципи моніторингу та аналізу використання хмарних сервісів та хмарних обчислень співробітниками та відділами організації.

Завдання роботи полягають у наступному:

- здійснення теоретичного аналізу методів та засобів моніторингу хмарних обчислень;
- –здійснення аналізу та порівняння програмних засобів моніторингу адміністрування хмарних сервісів;
- проведення підсумків роботи, на основі яких можу бути сформований об’єкт, предмет та завдання для подальших розробок;
- розробки програмного комплексу, який реалізуватиме розроблені алгоритми та розроблений метод;
- провести практичне тестування роботи розробленого програмного комплексу.

У нашій дипломній роботі використано різні методи дослідження та технічні засоби:

- спостереження, аналіз, синтез, доведення, формалізація та експеримент;
- комп’ютерні засоби проектування та розробки, програмування, налагодження і тестування;

– персональні комп’ютери, сервери, хмарні технології.

Наукова новизна отриманих рішень полягає, з огляду на визначені підходи і невирішені задачі в цій області, що нами було вперше запропоновано покращений метод моніторингу хмарних сервісів з допомогою інформаційної панелі, наведено алгоритм його роботи, архітектуру системи та сам програмний комплекс. Крім того, Наукова новизна полягає у тому, що метод моніторингу адміністрування хмарних сервісів отримав свій подальший розвиток, а це дозволило загалом підвищити роботу веб-інформаційної панелі для адміністрування хмарних сервісів та збільшило продуктивність і ефективність роботи програного комплексу та інформаційної системи певної організації загалом.

Практичне значення нашої роботи полягає у застосуванні покращеного розробленого методу адміністрування хмарних технологій при розробці інформаційних панелей такого роду, здійснювати їх реалізацію таким чином, щоб інформаційна панель дозволяла їх користувачам відстежувати, аналізувати та контролювати різні типи ресурсів, зокрема такі параметри як навантаження процесору, використовувану оперативну пам'ять, вільне місце на диску тощо. Це надає можливість особам відповідальним за використання інформаційної системи та програмного забезпечення загалом аналізувати отриману інформацію про використання різних типів ресурсів. На основі проведеного моніторингу та аналізу діяльності це дасть змогу їм приймати певні рішення щодо покращення роботи співробітників та відділів при використанні цих хмарних технологій та роботи всієї інформаційної системи організації загалом.

Питання практичного використання обговорювалось на XIV Всеукраїнській науково-практичній конференції «Актуальні проблеми комп’ютерних наук АПКН-2022» де було здійснено апробацію при виступі з доповіддю на конференції та її обговорення.

1.4 Висновок

В першому розділі нашої дипломної роботи було показано, що в умовах сьогодення хмарні технологій є досить є досить затребуваною, необхідною та значимою.

В цьому розділі, зокрема, було проведено аналіз предметної області, дослідження невирішених проблем в наявних сучасних дослідженнях. Наведено приклади та показано, що різним організаціям потрібен такий продукт. Було окреслено проблему невирішеного раніше рішення, а саме показано, що, основним головним моментом при використанні хмарних ресурсів та хмарних обчислень є те, що компаніям доводиться постійно контролювати та коригувати їх використання та роботу, для визначення навантаження на кожен тип ресурсу ресурс при використанні їх в своїй діяльності використання IT-технологій, а також необхідність аналізу і обробки використання хмарних сервісів.

У розділі також проведено порівняльний аналіз переваг та недоліків існуючих рішень на основі аналізу наукових праць по даній темі.

Здійснено постановку задачі реалізації покращення методу моніторингу хмарних сервісів. Наведено обґрунтування реалізація нового підходу до вдосконалення методу моніторингу хмарних сервісів інформаційної панелі на основі визначених основних систем стандартів та певних правил.

У розділі зазначено, що практичність розробки нового підходу до реалізації методу моніторингу хмарних сервісів полягає в затребуваності організацій та окремих користувачів такого роду рішеннями та необхідності засобів для покращення та автоматизації своє роботи в IT-сфері загалом.

2. АНАЛІЗ МЕТОДІВ МОНТОРИНГУ ХМАРНИХ ОБЧИСЛЕНЬ

2.1 Аналіз підходів, методології та технологій для вирішення завдання

Як було вказано нами хмарні технології - це тип хмарних обчислень, який покладається на обмін ресурсами, управління даними та виконання обчислень за допомогою цих спільних ресурсів.

Загалом майже бідь-яке IT-рішення на сьогодні позначать терміном хмарні обчислення або хмарні рішення (Cloud Computing (CC)). Хмарні обчислення чи хмарні рішення є метафорою пропозиції та споживання організаціями IT-ресурсів.

IT-ресурси та сервіси в хмарі користувачі не можуть бачити безпосередньо; між ними знаходяться так звані шари абстракцій. Рівень цих абстракцій, які пропонує певний хмарний сервіс мають багато різних варіацій: від пропозицій надання в оренду чи користування віртуальних машин (Virtul Mchine(VM)) до надання в оренду чи тимчасове користування програмного забезпечення як послуги (Software as a Service (SaaS)), інфраструктури як сервісу (Infrastructure as a service (IaaS)),платформи як сервісу Platform as a service (PaaS), а також розглядають модель «все як сервіс» Anything-as-a-service (XaaS). Вони базуються на основі на основі різних розподілених систем та хмарних рішень.

Загалом структура моделі Cloud Computing може бути представлена структурою, яка наведена на рисунку 2.1

Для забезпечення моніторинг таких хмарних сервісів, веб-інформаційні панелі вважаються досить корисним інструментом. Такий інструмент дозволяє здійснювати оперативну обробку інформації при використанні сучасних платформ хмарних сервісів.

Дамо визначення нашого розуміння інформаційної панелі моніторингу хмарних сервісів – загалом це користувальницький інтерфейс, який надає

змогу впорядковати та надати в сприйнятому вигляді інформацію таким чином, щоб її можна було легко читати, інтерпретувати та аналізувати. Це допомагає адміністратору іт-структури підприємства, або звичайному користувачеві взаємодіяти з необхідною та важливою інформацією з однієї сторінки без необхідності здійснювати перегляд інформації про кожен такий ресурс. Інформаційна веб-панель також дає змогу знизити складність і затрати на роботу персоналу з великими обсягами інформації. Також інформаційна панель зазвичай містить усі необхідні елементи, які вимагають швидкого відображення та сприйняття інформації, в подальшому переходячи проведення аналізу, здійснення статистики та прийняття рішень тощо.

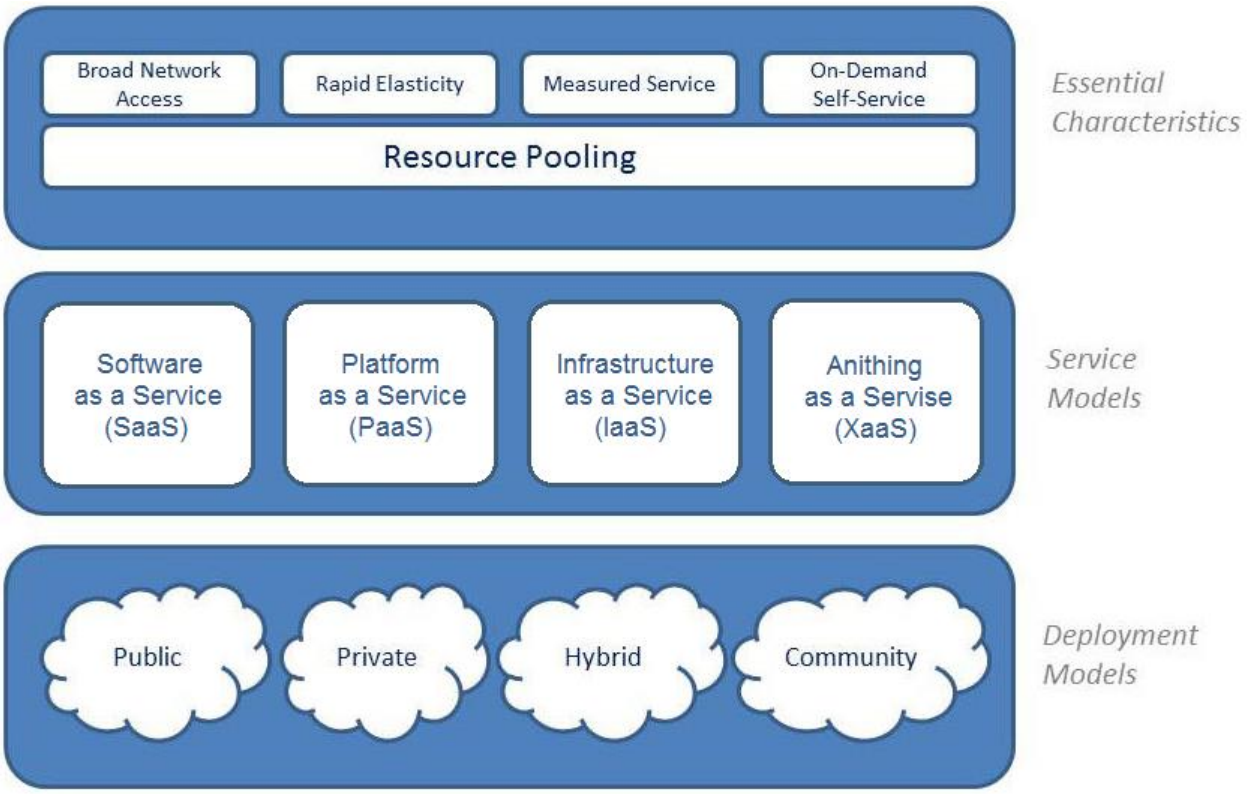


Рисунок 2.1 - Структура моделі Cloud Computing

Перед початком роботи над розробкою нашого програмного комплексу та вдосконалення існуючих методів нами було розглянуто ряд різних підходів, методології та технологій для реалізації нашої моделі та програмного забезпечення моніторингу хмарних сервісів. Розглянемо основні з них.

2.2 Аналіз платформ для побудови програмного комплексу моніторингу хмарних сервісів.

OpenStack - це платформа для хмарних технологій та хмарних обчислень з відкритим вихідним кодом. Організації NASA та Rackspace для цієї платформи розробили спеціалізоване програмне забезпечення. Дана технологія має досить не складну реалізації та являє собою відкриту масштабовану інфраструктуру як сервіс IaaS [18].ЦЯ платформа надає послуги хмарних обчислень з будь-яким потрібним для цього обладнанням, яке б забезпечувало сумісність, гнучкість, сумісним та масштабованість. Платформа OpenStack, загалом, це набір різних програмних засобів для створення та керування платформами хмарними сервісами, як для публічних та різних приватних послуг хмарних технологій. В якості ядра для бчислень платформа OpenStack використовує базу даних основана на стандартах SQL для збереження, отримання та обробки інформації про стан хмарного сервісу.

Перерахуємо основні компоненти платформи OpenStack (Рисунок 2.2).

Одним із основних компонентів платформи виступає так звана Nova, яка є основним обчислювальним середовищем для платформи OpenStack. Nova побудована на загальній архітектурі, яка заснована на повідомленнях. Його використання призначено для розгортання та управління різною кількістю віртуальних машин та інших хмарних технологій для обробки обчислювальних процесів. Драйвери Nova можливість надають функціонал для зв'язку з різними програмами, які працюють на різних компютерах з різними операційними системами, і забезпечують їх взаємодію через через Інтернет. Даний двигун має компоненти, які здійснюють обмін даними через повідомлення. Його компоненти можуть бути розподілені в мережі, що робить його гарно масштабованим. Усі запити, які надходять на ресурси хмарних сервісів, обробляються з допомогою цього двигуна. Далі він використовує

інші програмні засоби для їх подальшої обробки. Після обробки знову відбувається обмін інформацією між хостами.

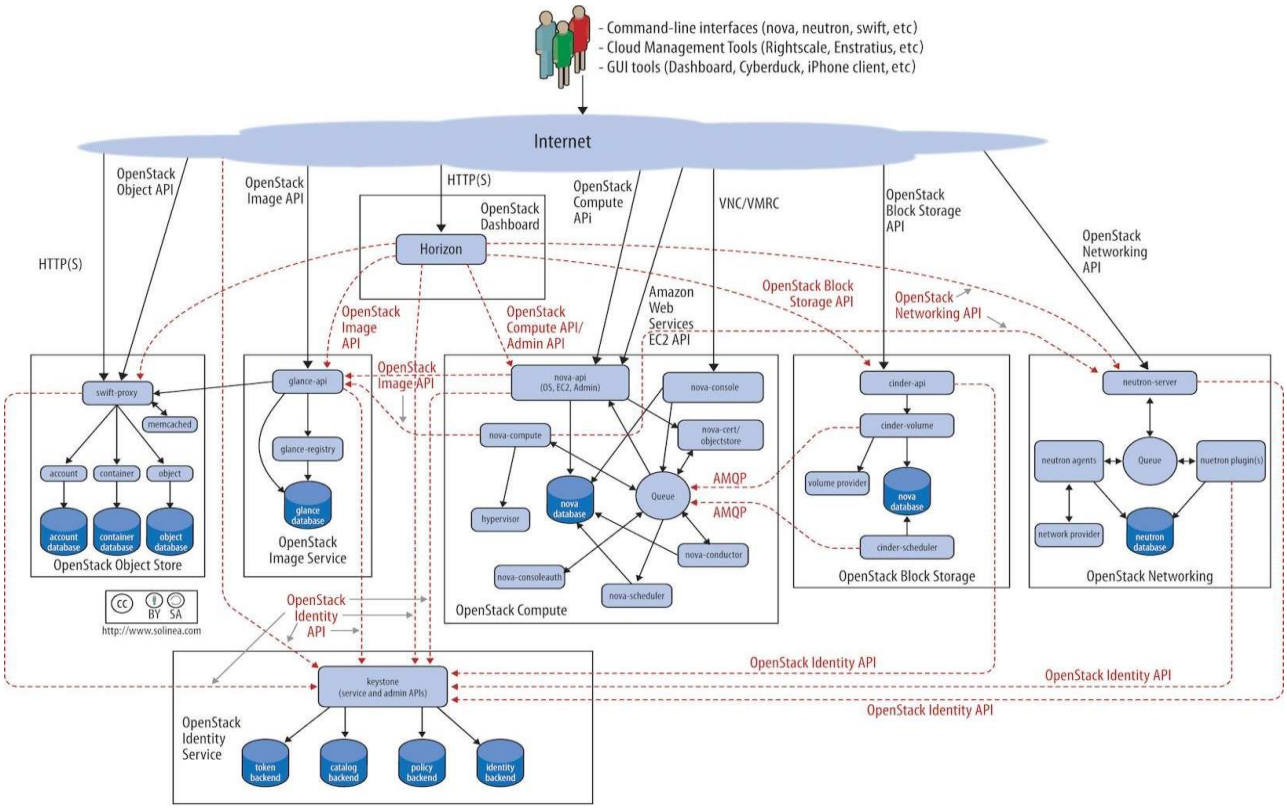


Рисунок 2.2 - Архітектура платформи OpenStack-

Платформа Swift надає засоби зберігання об’єктів і файлів. Посилання на файли в Swift здійснюються на основі унікального ідентифікатора, що посилається на конкретний файл, або деякий фрагмент інформації, незалежно від того, де зберігається сам файл. Такий підхід дає значні переваги, серед яких масштабування та здійснення системою резервного копіювання даних у разі апаратного збою, або неполадок в мережевого трафіку Він використовує Glance для зберігання своїх резервних копій і Cinder для створення резервної копії для всіх дисків віртуальної машини.

Cinder кінцевим користувачам надає послуги блочного зберігання. Він використовується також для надання ресурсів віртуальним машинам, а також має функцію резерних копій, де в конкретний момент часу може зберігатися повний стан віртуальної машини, та використовуватися для клонування

віртуальної машини. Такий підхід є корисним в сценаріях, в яких швидкість доступу до даних є досить важливим фактором.

Keystone надає послугу ідентифікації для OpenStack. По суті, це головний усіх користувачів хмариного сервісу OpenStack, який співставляється з усіма послугами, що надаються сервісом, де порівнєть права на дозвіл використання. Він забезпечує аутентифікацію на основі токенів. Всі виконані запити відправляються в Keystone для аутентифікації. Таким чином усі компоненти залежать від Keystone при перевірці усіх облікових даних.

Нейтрон забезпечує мережевий контроль для OpenStack. Він забезпечує роботу в мережі як послугу між різними інтерфейсними пристроями. Це дає гарантію, що кожен компонент розгортання OpenStack може швидко, надійно та ефективно взаємодіяти між собою.

Horizon - це інформаційна панель, надана для OpenStack. Загалом це графічний інтерфейс, який надає всю інформацію про різні складові OpenStack. Загалом це інтерфейс для кінцевого споживача який забезпечує інформацією про всі компоненти інтерфейсу веб-додатку. Це досить важливо для системних адміністраторів та аналітиків, оскільки дозволяючи їм стежити за певним сервісом та керувати ним.

Glance надає послуги резервних копій Для OpenStack. В даному випадку під «резерними копіями» маються на увазі віртуальні копії дисків та томів системи. Glance дозволяє використовувати ці резервні копії як шаблони при розгортанні нових екземплярів віртуальних машин.

Ceilometr надає послуги моніторингу, які дозволяють хмарному сервісу надавати послуги виставлення рахунків окремим користувачам хмарного сервісу. Він також веде облік системного використання різними користувачами кожного з різних складових OpenStack. Він використовується для обліку використання хмарних сервісів та послуг для звітності та аналізу про використання.

SAVI розшифровується як розумні додатки на віртуальній інфраструктурі, яка є мережею стратегічних досліджень NSERC. Зараз вони є

дослідницькими та тестовими платформи веб-застосунків. Метою дослідження мережі SAVI є вирішення питань проектування майбутніх прикладних платформ, які побудовані на універсальній та гнучкій інфраструктурі, з допомогою якої можна легко розгортати та підтримувати великомасштабні розподілені додатки для майбутніх застосувань [21].

Національна платформа розподілених додатків розроблена SAVI для створення та доставки майбутніх інтернет-додатків. Тестове середовище SAVI забезпечує гнучку, віртуалізовану конвергентну інфраструктуру для проведення експериментальних досліджень у прикладно-орієнтованих мережах, хмарних обчисленнях, інтегрованому бездротовому зв'язку та архітектурою інтернету.

Тестовий стенд SAVI - це мережа віртуалізованих кластерів спеціалізованих ресурсів. Він використовується SAVI для контролю та управління певною віртуальною інфраструктурою. Тестове середовище SAVI забезпечує доступ до різнорідних ресурсів, таких як віртуальних процесорів, пам’яті, дисків та доступ до мереж. Тестовий стенд SAVI також буде підтримувати експериментування в додатках, побудованих на передових сервісах, які забезпечують розвідку за допомогою аналітики та розширеної обробки медіа. На рисунку 2.3 представлена архітектура тестового середовища SAVI.

Тестовий стенд SAVI складається з Орендаря, Середовища та Екземплярів.

Орендар використовується для диференціації експериментаторів на високому рівні, оскільки імена орендарів є назвою організації в тестовому блоці SAVI. Експериментатори з цієї ж організації групуються під одним і тим же ім'ям орендаря. Групування користувачів за іменем організації полягає в тому, щоб класифікувати користувачів у тестовому блоці SAVI та обмежити користувачів у їхніх діях для виконання певного завдання. Наприклад, користувач з певної організації, який має ідентифікатор клієнта як "orgunit12", має право переглядати інформацію про іншого користвача цієї організації.

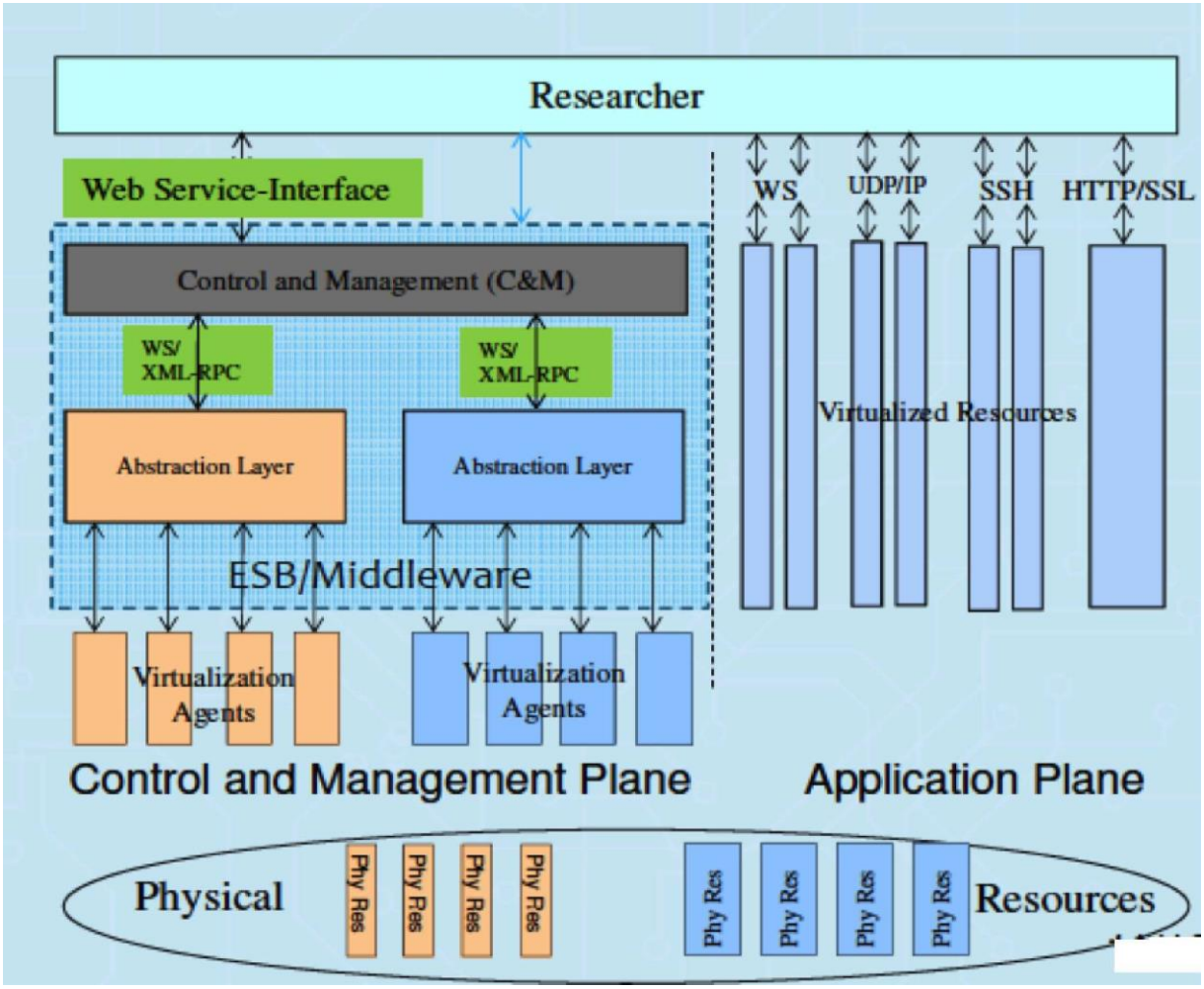


Рисунок 2.3 - Архітектура тестового стенду SAVI

Середивище в SAVI визначається як фізичне розташування, де знаходиться все фізичне обладнання. Коли користувачі дають запит на розподіл ресурсів, їм потрібно визначити ім'я, і за їх поданням видається ресурси середовища де і відбувається розподіл ресурсів.

Екземпляри - це ресурси, призначені користувачам SAVI. Екземпляри - це ті ресурси, які будуть контролюватися інформаційною панеллю для певного хмарного середовища. Під екземплярами розуміють віртуальні машини або фізичні машини.

Мотивація цього проекту полягала в тому, що OpenStack не має інформаційної панелі моніторингу ресурсів. OpenStack надає користувачам інформаційну панель тестового центру SAVI, відомої під назвою «Горизонт», але ця інформаційна панель є лише засобом перегляду ресурсів на високому

рівні. Він не надає жодної інформації про використання ресурсу. Отже, розробка такого веб-додатка є досить важливою для тестового стенду SAVI.

Документація, надана щодо використання Ceilometera для випробувального стенду SAVI, досить обмежена, що є ще однією мотивацією для того, щоб спроектувати інформаційну панель моніторингу хмарних ресурсів панель і надати розробникам документацію про те, як ним користуватися. Крім того, основна документація пояснює деталі реалізації проекту, які можуть служити керівництвом користувача для використання Ceilometera для тестового стенду SAVI.

2.2 Порівняльний аналіз переваг та недоліків існуючих рішень

До цих пір ресурсні обмеження тестового стенду SAVI контролюються Horizon, який є інформаційною панеллю, наданою OpenStack. Horizon є абстрактним видом ресурсів і не забезпечує детальною інформацією уявлення про надані ресурси.

Отже, щоб вирішити дану проблему, інформаційна панель яку пропонуємо розробити, може відстежувати інформацію про надання ресурсів хмарних сервісі для кожного виду хмарних сервісів, та надавати детальнішу інформацію про деталі надання ресурсів дійсно корисно.

Ще однією причиною розробки цієї інформаційної панелі є надання інформації про використання хмарних сервісів, які забезпечать адміністраторам контроль над переміщенням ресурсів з одного средовища до іншого на основі таких факторів, як навантаження на певний середовище та місткість цього віртуального сервісу. Це значною мірою зменшило б навантаження на віртуальні сервіси, які обмежені у надані ресурсів через свої технічні можливості.

OpenStack використовує «Горизонт» для інформаційної панелі OpenStack, яка є веб-інтерфейсом, щодопомагає користувачам керувати ресурсами та послугами. Він забезпечує середовище для користувачів, щоб знайти інформацію про ресурс, і обравши конкретний ресурс, вони можуть отримати інформацію про цей ресурс.

Інформаційна панель OpenStack використовує «Горизонт» як підкреслюючу основу, яка забезпечує веб-інтерфейс для сервісів OpenStack, включаючи Nova, Swift, Keystone, Glance, Neutron, Cinder.

Деякі з ключових особливостей «Горизонт» полягають в таких характеристиках його роботи:

– основна підтримка - Він підтримує всі основні проекти OpenStack;

– розширюваність - додавання нового компонента та ресурсу може зробити будь-який кроситувач;

– керованість - простий в користуванні і легко масштабується на різні платформи;

– наслідуваність – працює з усіма парадигмами та має стандартизацію при цьому;

– стабільність – реалізується через спеціально розроблений API;

– використовуваність - одна з ключових особливостей, яка забезпечує популярність до «Горизонту», оскільки вона забезпечує сприйнятний користувацький інтерфейс, в якому легко орієнтуватися.

Огляд ресурсів на дозволяє користувачам, які ввійшли в систему з ідентифікатором клієнта, подивитися на інші ресурси.

Інформаційна панель OpenStack використовує «Горизонту» і відображає інформацію про хмарні сервіси для кожного ресурсу, але вона не надає однакову інформацію для однотипних ресурсів.

Це одне з обмежень інформаційної панелі OpenStack, яка підкреслює необхідність розробки інформаційної панелі, яка б надавала інформацію про хмарні сервіси яка є специфічною для певного ресурсу.

2.2.1 Інші підходи

Розглянемо деякі з різних підходів, які були прийняті при розробці інформаційних панелей інформаційних панелей для моніторингу хмарних сервісів іншими дослідниками.

В [6] автори розробили фреймворк у вигляді приладової панелі, яка використовується для захоплення всіх елементів комплексного фреймворку управління хмарними рішеннями. Створена інформаційна панель надає користувачам можливість керувати віртуальними машинами окремо або в групах. Інформаційна панель поставляється з такими функціями, як можливість користувача зробити резервну копію однієї віртуальної машини, або декількох віртуальних машин, або перезавантажити окрему або кілька віртуальних машин з інформаційної панелі одним клацанням миші.

Повна версія інформаційної панелі була розміщена на IBM Smart Cloud. Інформаційна панель, яка використовується авторами, забезпечувала перевірку дзвінків, що направлялися з точки А в точку Б , а потім перевіряла ці виклики на основі знімків забезпечується приладовою панеллю приладів.

Ця приладова панель є простим способом візуально переконатися, що умови, вказаних в експерименті виконуються. Інформаційна панель використовується для управління та перевірки хмарних сервісів. Ефективність цієї інформаційної панелі також вимірювалася за допомогою відеотрансляції. На відміну від інформаційної панелі SAVI, тут відстежувались різні характеристики для одного хмарного сервісу та надала користувачеві можливість переглядати на різних ресурсах, заснованих на конкретному ресурсі.

Автори в [7] розробили інформаційну панель, яка динамічно отримує доступ до стійкості сервісів хмарних обчислень. Хмарна панель] використовується для доступу до загального впливу сервісів , які розміщені в хмарі. Модель оцінки інформаційної панелі базується на чотирьох факторах стійкості, економічних причинах, екологічних факторах та соціальних факторах. Значення приладової панелі найкраще реалізується, коли різні користувачі можуть вибрати власні переваги стійкості для вимірювання. Однак у роботі не наводиться поглиблена інформація про деталі реалізації приладової панелі, а представлені лише моделі оцінки на основі перерахованих факторів.

Архітектура Cloudlet для інформаційних панелей в хмарі пропонувала деякі методи розробки інформаційних панелей, які можуть бути використані виробничим сектором, якому потрібна така приладова панель. Автори показують, як презентаційні шари додатку тепер структуровані в набір віджетів. Кожен віджет має власне графічне представлення, яке підтримує сервіс, який можна легко включити в інформаційну панель, тим самим покращуючи прийняття рішень шляхом капіталізації. Архітектура для базового дизайну була розроблена на MVC (Model View Controller), а інтерфейс використовував шаблон дизайну презентації RIA (Rich Interface Architecture). Приладова панель була побудована з використанням технологій web 3.0 (HTML5, JQuery, CSS3).

Після того, як розроблена архітектура проекту, наведена вище для проектування інформаційної панелі для хмарних сервісів, стає очевидним, що вони використовуються для певного хмарного сервісу. Наявна інформаційна панель OpenStack не забезпечує моніторинг обмежень ресурсів для певного ресурсу та не настроюється для інформації про конкретний ресурс. Усі перераховані вище архітектури дуже добре визначені у своїй області та області досліджень, але створення інформаційної панелі для тестового середовища SAVI, характерного для конкретного ресурсу та має перехованні недоліики, які варто доопрацювати та реалізувати.

Наша інформаційна панель побудована з використанням технологій Веб 3.0 та використовує фреймворк Node.js. Причиною використання Node.js стало те, що на цій платформі легко масштабувати веб-додаток, а дещо відіграє досить важливу роль у роботі з хмарними сервісами. В подальшому приладова панель може бути реалізована для моніторингу декількох різних ресурсів.

2.3 Аналіз архітектури методу моніторингу даних хмарних сервісів

Основним інструментом для побудови методологічного підходу нами обрано Ceilometer, який є відповідно компонентом платформи OpenStack і надає своїм користувачам послуги моніторингу хмарних сервісів. Послуги з моніторингу - це технологія передачі і прийоме різних вимірюваних величин, розроблена з метою віддаленого моніторингу хмарних сервісів та їх послуг для збору та обробки даних.

Ceilometer був випущений з новою версією платформи OpenStack. Він використовується для збору інформації про використання послуг хмарних сервісів та віртуальних машин, які розгорнуті в Cloud Computing. Дані він зберігає в базі даних, а потім для аналізування та обробки даних ініціює певні події дій, коли виконуються наперед визначені критерії.

Даний компонент платформи OpenStack використовується в основному різними системами для встановлення певної точки доступу, щоб забезпечити їм надання інформації усіх компонентів OpenStack.

Ceilometer є спеціальним компонентом платформи OpenStack і допомагає в вирішенні певних задач, що полягають в наступному:

– забезпечує ефективний збір різних даних обліку ресурсів, таких наприклад, як центральний процесор, пам’ять і мережевий трафік тощо;

– дані збираються за допомогою повідомлень, які надіслаються з хмарних сервісів, а також шляхом опитування їхньої інфраструктури;

– розробники можуть налаштувати різні типи даних, які їм потрібно моніторити;

– зібрані таким чином дані стають доступними через API іншим користувачам програмного комплексу.

Архітектура інструменту Ceilometer розроблена таким чином, що кожну його службу можна легко змінювати, для включення чи виключення різних компонентів, додаючи співробітників та комп'ютери до відповідного архітектурного рішення. Він надає декілька послуг: опитування, сповіщення, збір, API та обробку сигналів.

Ці служби працюють незалежно від збору та обробки інформації та даних, але вони, також, можуть працювати спільно, для забезпечення вирішення різного типу завдань.

Архітектура Ceilometer складається з таких складових компонентів:

– агент опитування: Робота виборчого цього агента призначена для мніторингу різних служб платформи OpenStack та побудови різних лічильників для цих служб.

– агент оповіщення: Це фонова комп'ютерна програма, яка працює як фоновий процес, і призначена для постійного прослуховування і моніторингу повідомлень, що знаходяться в черзі повідомлень. Він перетворює дані цих повідомлень на зразки та різні події, а потім застосовує до них дії конвеєра.

– збирач: всі дані, які виробляються агентом опитування та сповіщення, збираються та записуються цим демоном. Він може працювати на одному або декількох центральних серверах управління для моніторингу черг повідомлень.

– API – комплекс програмних засобі та бібліотек, який надає доступ до даних бази даних та інших сервісів. Колектор і сервер API відповідно мають доступ до Системи курування базами даних та бази даних загалом. Цей тип сервісу використовується для запиту та перегляду даних, який він отримує від збирача.

– Правила - набір правил, на яких базується моніторинг, оцінка та самі повідомлення.

Дані в сховищі записуються в базу даних для оптимізації зберігання, обробки та модифікації запитів. Облік ресурсів наданих хмарними сервісами , також записується у базу даних.

З наведеного нижче рисунка 2.4 можемо бачищо, що вимірювач має прив'язку до моніторингу сервісу Glance, а лічильник «networking.incoming.bytes» має прив'язку до моніторингу сервісу Neutron

.

Name	Type	Unit
cpu	cumulative	ns
cpu_util	gauge	%
disk.ephemeral.size	gauge	GB
disk.read.bytes	cumulative	B
disk.read.requests	cumulative	request
disk.root.size	gauge	GB
disk.write.bytes	cumulative	B
disk.write.requests	cumulative	request
image	gauge	image
image.size	gauge	B
instance	gauge	instance
instance.scheduled	delta	instance
instance:m1.large	gauge	instance
instance:m1.medium	gauge	instance
instance:m1.small	gauge	instance
instance:m1.tiny	gauge	instance
instance:m1.xlarge	gauge	instance
ip.floating	gauge	ip
ip.floating.create	delta	ip
ip.floating.update	delta	ip
memory	gauge	MB
network.incoming.bytes	cumulative	B
network.incoming.packets	cumulative	packet
network.outgoing.bytes	cumulative	B
network.outgoing.packets	cumulative	packet
port	gauge	port
port.create	delta	port
port.update	delta	port
vcpus	gauge	vcpu

Рисунок 2.4 - Список назв ресурсів хмарних сервісів

Ceilometer має змогу контролювати ресурси платформи OpenStack. Ресурс - це певний об'єкт, який має певний хмарний сервіс та використовується метою його моніторингу. Вимірювальні дані в ньому поділяються на три типи: калібрувальні (дискретні структури),

накопичувальні (які збільшуються за розміром за певний проміжок часу) та дельта (які змінюються за розміром за певний проміжок часу). Для цього Ceilometer використовує, як було вище сказано спеціальний набір команд узгоджений з платформою OpenStack.

Наприклад, наведена команда може бути використана при отримані списку назв ресурсів хмарного сервісу з їх типами та юнітами:

ceilometer check_list_param;

2.4 Розробка вдосконаленого методу моніторингу хмарних сервісів

В порівняні з попередніми рішеннями та методами ми пропонуємо новий вдосконалений метод моніторингу хмарних сервісі та їх ресурсів. Так, зокрема він полягає в розширені роботі API, здійсненню асинхронного доступу до даних в двосторонньому зв’язку з всіма модулями програмного комплексу а відповідними хмарними сервісами тощо.

Опишемо детально суть та методологію вдосконаленого методу.

Збір даних здійснюють колекціонери та агенти, які збирають дані з кількох ресурсів. На рисунку 2.5 показано, як опитувальний агент і агент повідомлень створюють збір даних з хмарних сервісів з допомогою модуля обробки запитів та модуля оброки повідомлень, та як відбувається взаємодія між цими модулями програмного комплексу.

Обмін запитами модуля обробки даних з хмарними сервісами здійснюється за допомогою API для кожного окремого хмарного сервісу.

За збір даних відповідають модулі обробки запитів та модуль обробки повідомлень, які збирають дані з різних хмарних сервісів. Як видно з рисунку 1 дані збираються двома методами, а саме з допомогою модуля запитів і модуля повідомлень здійснюється збір даних.

У модулі обробки запитів метод опитування проводиться через певний проміжок часу і дані збираються на кожному цьому проміжку часу.

Опитування проводиться за допомогою розробленого API та інструментів для збору даних та інформації.

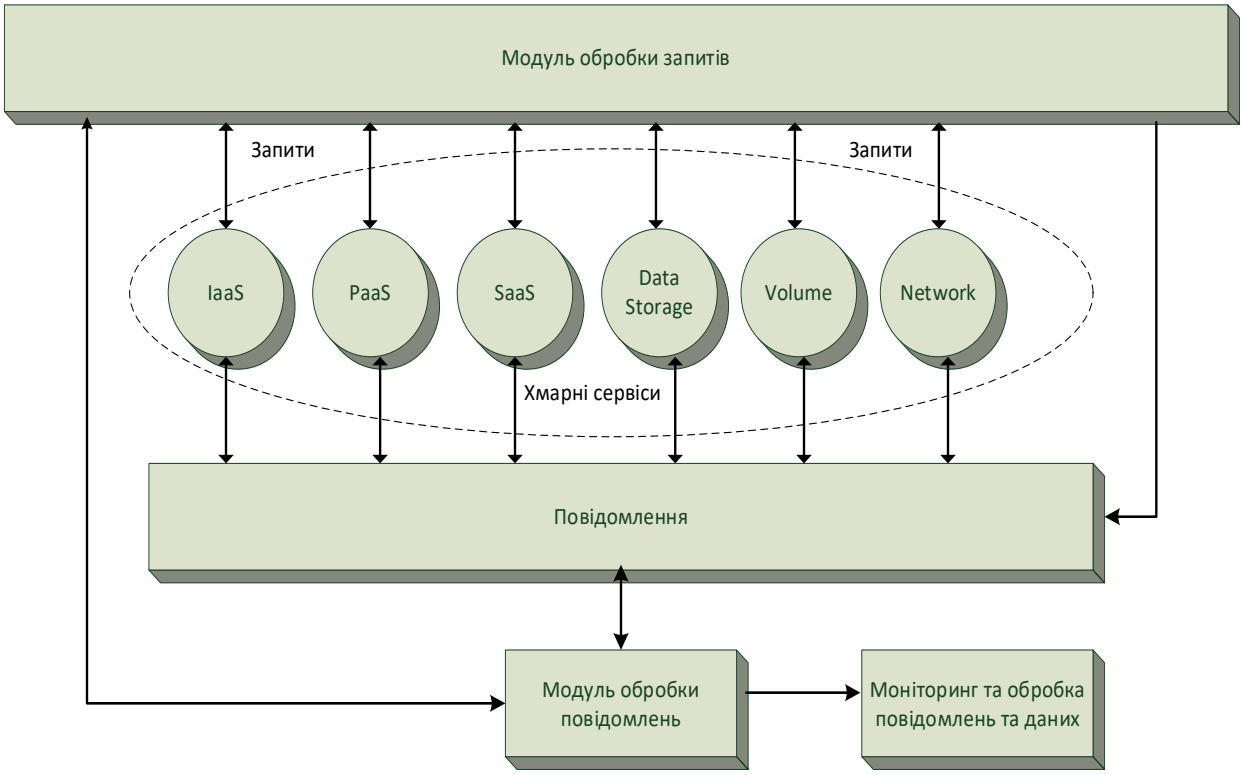


Рисунок 2.5 - Збір даних збирачами та агентами та взаємодія модулів

Модуль повідомлень налаштований на опитування локального або віддаленого хмарного сервісу з допомогою API. Цей модуль також запитує інформацію у хмарних сервісів про послуги для отримання даних. Він працює в обчислювальному модулі, а модуль опитування здійснює обробку повідомлень для інформаційної системи. Модуль повідомлень відповідає за запити даних, які пов'язані з обчислювальними ресурсами. Для іншого виду ресурсів модуль здійснює обробку повідомлень. Частота опитування може контролюватися згідно з налаштування API.

У модулі збору повідомлень події, що генеруються з повідомлень зібраних з хмарних сервісів, збираються та спотворюються в дані придатні для локальної інформаційної системи. У цьому методі модуль повідомлень відстежує черги повідомлень для інформаційної системи компанії.

Модуль обробки повідомлень є головним модулем системи збору даних, який відстежує повідомлення на вміст даних, що надаються хмарними сервісами, а також забезпечує внутрішню комунікацію. Повідомлення захоплюються модулем повідомлень, а потім перерозподіляються на основі структуризації даних кінцевим споживачам у вигляді придатним для обробки інформаційною системою. Певні повідомлення також у модулі можна конвертувати в певні події, які фільтруються за типами.

До модуля обробки даних надходять дані, які зібрані іншими модулями, де він здійснює їх обробку та публікує ці дані за допомогою декількох конвеєрів. На модуль повідомлення покладена відповідальність за обробку самих даних. Доступ до даних здійснюється через сервіси API розроблені для кожного окремого хмарного сервісу.

Модуль моніторинг і обробки повідомлень і даних здійснює публікацію даних за допомогою двох методів: повідомлювача та видавця, на основі повідомлень, які може опублікувати дані в стеку повідомлень

Отримані дані, які зібрані інформаційною панеллю, можуть зберігатися в базі даних, файлі, базі даних інформаційної системи компанії тощо. Дані збираються інформаційною панеллю, перевіряються та зберігаються потім в базу даних.

Доступ до даних через сервіс API. Зібрані дані стають доступними для розробника завдяки використанню API.

Таким чином вдосконалений метод моніторингу хмарних сервісів надає більш гнучку та масштабовану систему, яка покращує не тільки збір та обробку інформації, а й надає зручний інтерфейс для роботи з самою системою.

2.5 Постановка технічного завдання проекту

Основною задачею для розробки програмного комплексу є реалізація інформаційної веб-панелі, в функції якої має входити збір та обробка даних від сервера додатків, реалізованого через бібліотеку API з однієї сторони та забезпечення інтерфейсу взаємодії користувача з інформаційною панеллю.

Програмний комплекс повинен загалом виконувати такі функції:

– забезпечувати візуальний інтерфейс користувача для моніторингу та опрацювання даних;

– здійснювати обмін даними зі всіма складовими програмного комплексу і їх модулями;

– здійснювати вибірку та обробку даних згідно поставлених цілей до програмного комплексу і його складових;

– забезпечувати зручну та зрозумілу подачу запитуваних даних користувачам програмного комплексу;

– асинхронно опрацьовувати дані на будь якому рівні та передачу їх між модулями програмного комплексу.

При плануванні розробки системи та розробці її архітектури важливим є питання, що виникає при розробці будь-якого типу додатків, є вибір засобів та мов програмування. Серед безлічі існуючих інструментів, засобів та різних мов програмування, які по своїх можливостях можуть забезпечити реалізацію можливостей подібних програмних комплексів шляхом реалізації різноманітних плагінів, фреймворків та готових систем потрібно обирати для кожної складової системи ті, які найкраще справляють з такими задами на своєму рівні. Для розробки нашого програмного комплексу нами було обрано ряд технологій та мов програмування. Оскільки наш програмний комплекс є модульним, де кожен модуль працює на окремому незалежному програмно-технічному комплексі то, відповідно нами було обрано і різні мови програмування та програмні рішення.

Поряд з цим постає питання в виборі типу додатку яке часто виникає при плануванні розробки програмних систем, а саме якого типу має бути додаток: десктопна версія чи веб-додаток. Обидва типи додатків мають ряд своїх

переваг та недоліків в порівнянні з іншим. Програмні рішення реалізовані у вигляді веб-додатків працюють через браузер і їх не потрібно встановлювати на персональний комп'ютер. Іншою перевагою веб-додатків є те, що вони виконують основні обчислення на стороні серверу, що може значно зменшити навантаження на пристрій користувача. Основним недоліком веб-датків є вимога постійного підключення до мережі інтернет. Але, в нашому випадку ми маємо роботу з хмарними сервісами, тому навіть реалізація програми у вигляді десктопного рішення всерівно буде вимагати постійного підключення до мережі інтернет.

Оскільки в нашому проекті ми будемо мати роботу з хмарними технологіями, то, на нашу думку, реалізацію даного проекту слід також використати хмарні технології та сервіси і реалізувати кінцевий продукт саме з допомогою цих технологій. Створення веб-додатків які доступні з будь-якого комп'ютера, смартфона, планшета – це найкраща інвестиція в розвиток будь-якої компанії

Так у нашому проекті, для реалізації програмного комплексу нами було обрано серверну операційну систему Linux, мову програмування C/C++, мову серверного програмування bash, мови програмування веб-сайтів PHP, HTML, JavaScript та фреймворк Node.js.

Такі рішення дозволяють найкраще спростити роботу бізнесу та і оптимізувати різні бізнес-процеси для власників та клієнтів компанії. Основна перевага таких рішень полягає у тому, що усі дані розміщуються в масштабованому, відмовостійкому та захищеному хмарному сховищі, а доступ до даних та відповідних модулів здійснюється через будь-який браузер з будь-якого місця з різних пристроїв.

Щоб відповідати запитам часу і бути конкурентноздатним веб-ресурс компанії має бути динамічним, надавати не певну статичну інформацію за запитами користувачів, а саме динамічно генерувати інформацію на веб сторінках з тими даними, які необхідні користувачеві веб-ресурсу..

Саме тому зараз на зміну створення десктопних версій програмного забезпечення та розробки статичних веб-сторінок на сайтах потрібна розробка веб-додатків.

Отже, нами було окреслено вимоги та шляхи реалізації нашого програмного комплексу. Загалом, це має бути програмний комплекс на основі хмарних технологій з кінцевим інтерфейсом у вигляді веб-додатку, який виконує функції моніторингу хмарних сервісів та повинен забезпечувати виконання наступних функцій:

– забезпечувати введення-введення користувачами програмного комплексу необхідної інформації щодо ресурсів хмарних сервісів;

– здійснювати моніторинг, а саме здійснювати збір даних про ресурси хмарних сервісів;

– виконувати обробку запитів з допомогою сервісів API;

– виконувати опрацювання отриманої інформації та даних згідно до розробленого методу;

– забезпечувати якісне надання інформації у браузері через-веб-додаток для користувача у графічному форматі.

Виходячи з поставлених задач, можна основні моменти роботи програмного комплексу. Основним застосуванням розроблюваної системи є моніторинг роботи хмарних сервісів, які використовує організація. Для забезпечення моніторинг роботи хмарних сервісів необхідно окреслити необхідні параметри для роботи програмного комплексу згідно до розробленого нами методу.

Виходячи з цього, нами було визначено основні типи варіантів використання системи. Графічно роботу програмного комплексу загалом можна зобразити за допомогою діаграми використанняUse Case, яка наведена нижче на рисунку 2.6.

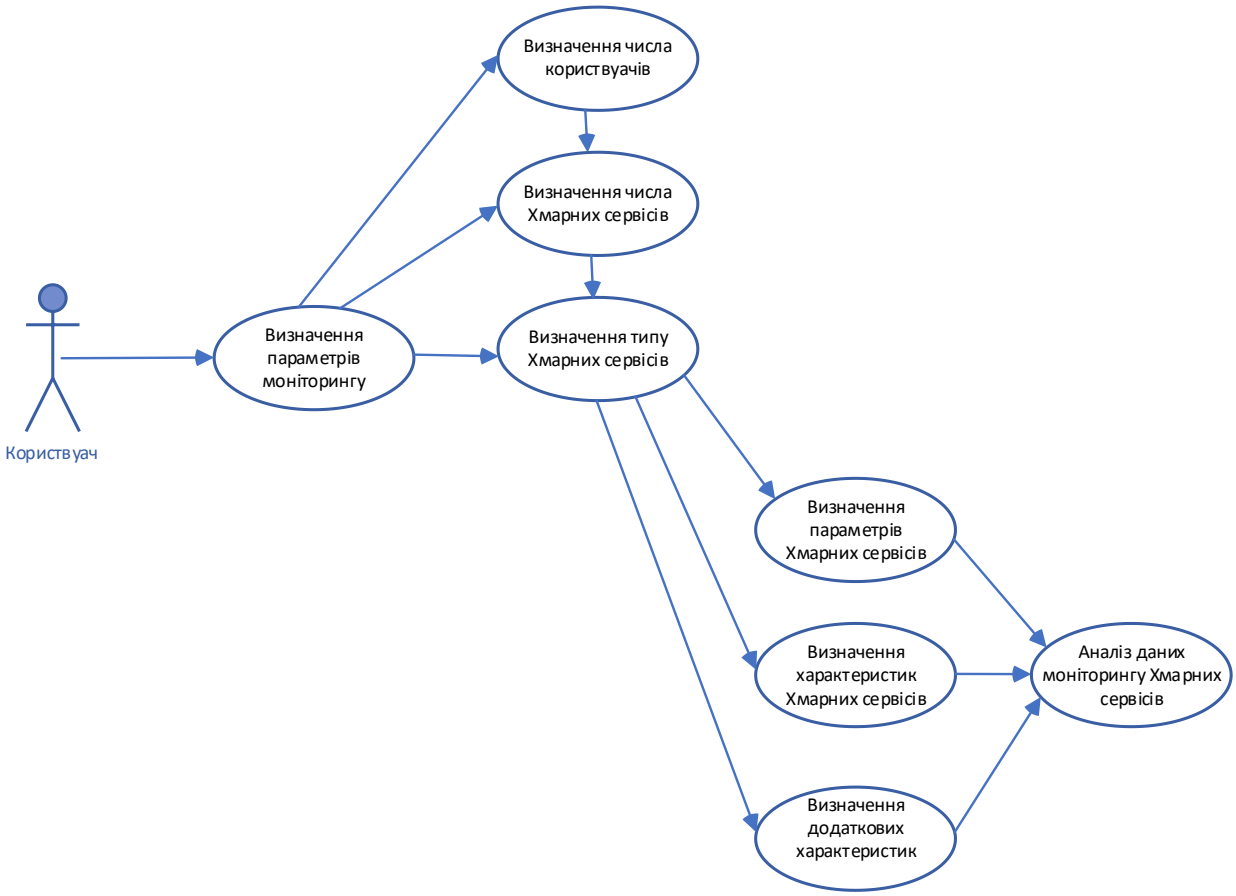


Рисунок 2.6 – Діаграма варіантів використання

Окреслені на даму етапі проектування системи варіанти використання Use Case нашого програмного комплексу вказують на подальшу розробку функціональних вимог до модулів програмного комплексу та структури програмного комплексу загалом.

Наступним кроком проектування програмного комплексу буде становлення послідовності дій при роботі користувача з програмним комплексом. Приведені варіанти використання Use Case мають декілька кроків, одні етапи повинні здійснюватися асинхронно поряд з іншими, інші повинні здійснюватися, як видно з діаграми, тільки після виконання попередніх, оскільки попередні етапи містять інформацію та дані, без яких неможливе виконання наступних етапів. На першому кроці стоїть визначення типу розроблюваного програмного комплексу. Саме визначення типу програмного комплексу буде визначати можливі варіанти використання на

подальших етапах, зокрема вибору різних платформи для його розробки. У нашому випадку згідно розробленого методу усі чотири складові платформи будуть виконувати роботу для систем ,як працюють у вимірі реального часу. Для інших типів модулів програмного комплексу тип платформ будуть визначатися згідно їх особливостей використання.

Подальшою дією при проектуванні програмного комплексу стає визначення параметрів його роботи. Так, параметрами нашого програмного комплексу системи, які визначені при розробці нашого методу буде кількість потоків, які підтримують між різними складовими програмного комплексу та, відповідно, апаратними і програними вимогами до них.

Наступним кроком, реалізації програмного комплексу необхідно спланувати інформацію про кількість користувачів, пристроїв сервісів задіяних в системі, а також додаткові параметри проектованого комплексу, серед яких, типи взаємодії, обробка специфічні типів запитів та даних на основі яких працює програмних комплекс.

Ключовим етапом роботи з програмним комплексом є отримання даних та результатів обробки інформації та аналізу їх користувачами. Користвуачі на основі виведеної інформації повинні мати змогу здійснювати аналіз та оцінку отриманих параметрів хмарних сервісів та забезпечити виведеня результатів їх обробки.

Для побудови даної моделі відображення послідовності дій для роботи з програмним комплексом використаємо UML-діаграму послідовностей, яка наведена на рисунку 2.7. Дана діаграма послідовностей відображає, як здійснюються етапи розробки архітектури та проектування програмного комплексу розробки, а також на етап тестування системи. Вона відображає основну послідовність дій роботи додатку. Також дана діаграма буде використана при виконанні тест-кейсів на етапі тестування.

Подальшим кроком проектування є планування станів модулів програмного комплексу. Початковий стан програмного комплексу це стан, в якому знаходиться програмний засіб до того моменту, як користувач здійснив

введення певних параметрів в систему. В такому стані програмний комплекс приймає параметри про типи хмарних сервісів їх ресурси та дані.

Рисунок 2.6 – Діаграма послідовності

2.6 Висновок

Таким чином, у розділі проведено дослідження та аналіз існуючих методів моніторингу хмарних сервісів та хмарних технологій. На основі

досліджених методологій було розроблено власну методологію, які допоможе її користувачам в моніторингу хмарних сервісів.

Загалом нами розглянуто різні підходи до розробки інформаційної панелі для моніторингу хмарних ресурсів. Зокрема, розглянуто різні програмні засоби моніторингу адміністрування хмарних сервісів і пов'язані з ним роботи з різними інформаційними панелями для віртуальних машин в середовищі хмарних обчислень. Здійснено аналіз методів та програмних засобів, що використовуються при розробці інформаційної панелі, та запропоновано внутрішню архітектуру для інформаційної панелі моніторингу хмарних сервісів.

Наступним етапом розробки дипломної роботи є програмна реалізація розробленого вдосконаленого методу та реалізація програмного комплексу.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ МОНІТОРИНГУ ХМАРНИХ СЕРВІСІВ

3.1 Програмна реалізація програмного комплексу

Проектована нами інформаційна панель моніторингу хмарних сервісів та ресурсів являє собою веб-додаток, надає доступ надає доступ до детальної інформації про використання хмарних сервісів співробітниками та відділами організації. Загалом вона забезпечує досить зручний інтерфейс, проста у користуванні з різноманітною інформацію про ресурс, де відображається відображаються дані про використання ресурсів користувачами, для забезпечення надання менеджерам та керівникам іт-відділів бачення про ситуацію з наявними сервісами. Інформаційну панель панель розроблено з використанням сучасних технологій розробки на технології веб-3.0. Дані технології допомагають поставлених рішень наведених у першому розділі базуючись на стандарті ISO/IEC 25010:2011 [5].

В даному розділі подано розробку та реалізацію інформаційної панелі моніторингу хмарних сервісів та ресурсів. Так, реалізація програмного комплексу розділена на чотири етапи, де він детально описано усі фази:

– проектування архітектури інформаційної веб-панелі;.

– використання технологій та інструментів, які використовуються при розробці інформаційної панелі;

– опис використовуваних API, які працюють на різних рівнях взаємодії інформаційної панелі, як з хмарними сервісами, так і інформаційною системою організації;

– розробка інтерфейсу користувача інформаційної панелі.

Загалом програмний комплекс складається з певного набору модулів, та собою програмний комплекс, що складається з базового вікна інтерфейсу, через який здійснюється моніторинг та доступ до інформації про використання хмарних сервісів та ресурсів.

Інформаційна веб-панель забезпечує моніторинг ресурсів на використання API доступу до них та відображає використання і обмеження для використання ресурсів доступних користувачам. Високорівнева архітектура отримання та обміну даних за допомогою API для нашої панелі наведена на рисунку 4.1 та складається з наступних компонентів:

– інтерфейс користувача, а саме інформаційна панель інструментів моніторингу хмарних сервісів;

– Apllication Server (сервер додатків), на якому розміщуються основні модулі та для роботи з хмарними сервісами для їх моніторингу та аналізу для подальшого використання;

– компоненти API OpenStack, на якому розміщено модулі для взаємодії з хмарними сервісами та сервером додатків;

– хмарні сервіси, які надають виконання запитів даних, пов'язаний з середовищем OpenStack.

В поданій архітектурі сервер додатків розташовується в основі наведеної архітектури високого рівня. Даний сервер розміщений на виділеному сервері з під керівництвом окремої операційної системи. Це основний оброблювач потоку даних між хмарними сервісами та користувальницьким інтерфейсом.

Інтерфейс користувача а саме інформаційна панель, відображає всю інформацію, пов'язану з хмарними сервісами.

Хмарні сервіси, працюють через API OpenStack, - це також хмарні сервіси, робота яких ресурси орієнтована на обмін даними з інформаційною веб-панеллю нашого програмного комплексу.

API OpenStack забезпечує послуги телеметрії та доступ до даних хмарних сервісів, а також здійснює обмін даними хмарними сервісами для моніторингу використання та встановлених параметрів хмарних ресурсів. Саме в ньому працюють модулі, які використовується для запиту та передачу інформації про ресурси.

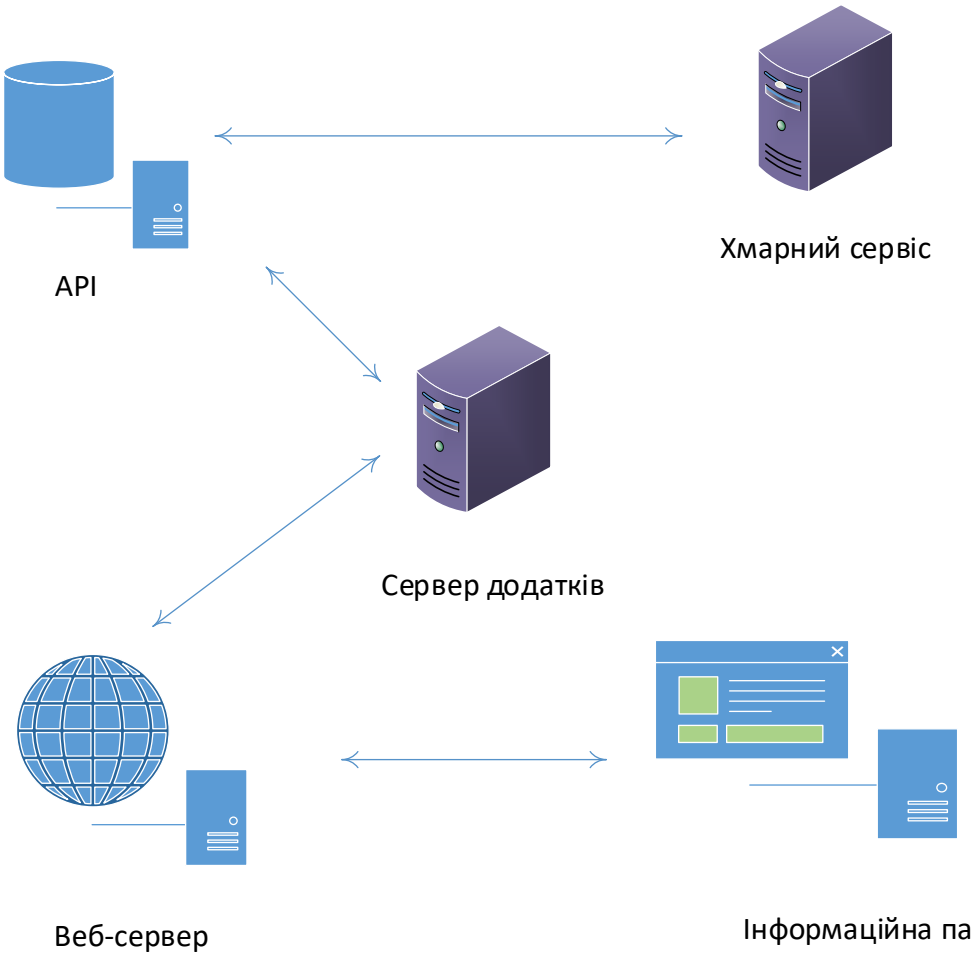


Рисунок 3.1- Високорівнева архітектура інформаційної веб-панелі

Спроектована інформаційна веб-панель використовує різноманітні технології, засоби та інструменти, для забезпечення масштабування оптимізованої інформаційної панелі моніторингу хмарних ресурсів.

Сервер додатків – платформа в основі якої лежить фреймворк Node.js, який в свою чергу, побудована на мові JavaScript і призначений для розробки масштабованих, зручних, швидких та ефективних мережевих додатків. Це досить розповсюджений фреймворк, який зарекомендував себе досить великою надійністю. Сам сервер додатків в основному реалізований на мові C++, де основна увага приділена продуктивності і невисокого споживання процесорного навантаження та споживання пам'яті [21].

Сервер додатків використовує неблоковану модель вводу-виводу інформації, яка керується подіями основаними на об’єктно-орієнтованому

підході, що, в свою чергу, робить його досить легким та ефективним для використання та ідеально підходить для такого типу програм в реальному часі з досить інтенсивним обміном та обробкою даних, які знаходяться на розподілених сервісах та пристроях.

Сервер, де працює інформаційна панель працює з допомогою фреймворку Node.js. Який, в свою чергу, є серверним середовищем java-скриптів, і представляє собою вузол Node.

Даний фреймворк містить власну бібліотеку API-lib, яка дає змогу запускати модулі програмам та працювати, як веб-сервер без спеціалізованого програмного забезпечення, зокрема веб-серверів Internet Information Server компанії Microsoft чи веб-сервера Apache. Встановлення пакетів у фреймворк здійснюється за допомогою спеціального менеджера пакетів (NPM) (менеджера пакетів вузлів). Даний менеджер призначений для встановлення модулів Node.js з його реєстру. Для становлення самого фреймворку також використовується інсталятор пакетів Node.

Сама веб-сторінка інформаційної панелі панель також розроблена з допомогою фреймворка Node.js з врахуванням усіх перелічених вище переваг. Проте, слід зауважити, що окрім основної проблеми пов’язаної з моніторингом даних, саме віддалені віртуальні системи можуть швидко витрачати свої ресурси. Завдяки, розробленому покращеному нами методу наша інформаційна панель буде надійною навіть при дуже великих масштабах моніторингу, коли кількість ресурсів, які моніторяться та контролюються, перевищують декілька десятків тисяч.

Сам фреймворк, який використовується нашою системою є сполучною ланкою. Він забезпечує саме функціональність маршрутизації. Це відноситься до визначення кінцевих точок адрес хмарних сервісів до нашої системи і того, як він реагує на запити різних клієнтів.

Про всі вхідні запити даних, які йдуть з інформаційної веб-панелі через API відповідає відповідно сам фреймворк. Кожен запит з допомогою розробленого API спрямовуються в потрібний потік та виходячи з того який

типу запиту використовується, кожному клієнту у відповідь надсилається потік затребуваної інформації.

Application Program Interface в розробленій системі це бібліотека розроблена на мовах C та JavaScript, яка забезпечується керуванням командами операційної системи Linux з допомогою скриптової мови Bash та, яка допомагає створювати потужні, адаптивні інтерфейси зв'язку та передачі і оброки даних з інформаційною панеллю та інтерфейси користувача з базовими режимами обробки даних. Саме він відповідає за те, що кожного разу, як тільки певні модулі інтерфейсу користувача інформаційної панелі змінюються або поновлюються, що у нашому випадку відбувається постійно, то він застосовується саме для цього. API забезпечує механізм прямої і зворотної прив'язки. Тобто, будь-які зміни, які зроблені в інтерфейсі користувача, відображаються відразу в моделі даних та навпаки.

Деякі з ключових особливостей API полягають у тому, що він забезпечує повне відстеження залежностей. Він автоматично оновлює ту частину інтерфейсу інформаційної панелі, коли відбуваються певні зміни в моделях даних. Також він відповідає за взаємодію з браузером користувача і він може бут застосовний поверх існуючого веб-додатку.

Наведений нижче фрагмент коду показує використання бібліотеки API в взаємодії інформаційної панелі з даними хмарних сервісів.

Даний фрагмент наведеного коду складається з трьох моделей, кожен з яких використовується для прив'язки даних інформаційної веб-панелі.

```
var serverDataModel = function () {
this.message = ko.observable   (" показує дані сервера");
shouter.subscribe(function(newValue)
{
        this.message(newValue);
}, this, "messageToPublish");
};
var resourceListModel = function(resourceList)
```

```
{
        resourceListOptions = ko.observableArray();
        var resourceListArray = Object.keys(resourceList).map(function (key)
        {
                return resourceList[key]
        });
for(var resourceIndex in resourceListArray)
{
        resourceListOptions.push(resourceListArray[resourceIndex]);
}
        resourceData = ko.observable();
        resourceData.subscribe(function(newValue) {
        shouter.notifvSubscribers(newValue. "messageToPublish");
        console.log(ˊnewValueˊ);
};
};
        var getResourcelDData = function (resourceName)
        {
         console.log('Selected Value ' + resourceName):
        };
$.get('/resource', function(data)
{
        ko.applyBindings(new masterVM(data));
});
var masterVM = function(data)
{
        resourceListModel = new resourceListModeli data): serverDataModel = new
        serverDataModeli); d3Model = new d3Model();
});
```

Головною моделлю даних виступає Master_VM. Вона містить підмоделі, кожна з яких є незалежною та забезпечує масштабованість програного комплексу, реалізуючи можливість додавання різних моделей до самої основної моделі.

Модель ResourceListModel складається зі списку всіх доступних ресурсів, що використовуються зокрема в якості критеріїв вибору та моніторингу для перегляду характеристик певного хмарного ресурсу.

Модель ServerDataModel містить інформацію про надані ресурси для кожного хмарного сервісу.

Модель D3Model містить інформацію, пов'язану з прив'язкою цих даних до різних характеристик хмарних сервісів.

Також розроблена бібліотека API надає послуги передачі даних телеметрії для OpenStack. Для того, щоб API забезпечувало запит даних, які пов’язані з пов'язані з характеристиками наданих послуг хмарних сервісів, розроблено спеціальні функції, які забезпечують виконання різних команд. Детальний огляд та роботу таких дій наведено у попередньому розділі.

Розробка механізму, пов'язаного з отриманням та передачею даних для характеристик хмарних сервісів полягає у розробці модуля маркера аутентифікації для кожного такого запиту. Так, зокрема, аутентифікація використовується для перевірки чи авторизований користувач є дійсно користувачем зареєстрованим в інформаційній системі.

Наведений нижче фрагмент коду - це bash-скрипт, який виконується при кожному запиті на авторизацію в системі.

```
export IP_USERNAME=Maxym
export IP_PASSW0RD=****
export IP_REGI0N_NAME=KHMEL
export IP_PAIN_BACK=Oleg
```

З наведеного фрагмента коду бачимо, що в якості користувача надано ім’я користувача та пароль. Так, ім'я клієнта – “Maxym”, а в назві регіону вказано поле “KHMEL”, характеристики сервісу якого потрібно моніторити та відстежувати для кожного такого запиту, який буде здійснюватися.

Проте, в наведеному коді, є один нюанс для відображення паролю в цей скрипт. Оскільки даний файл в bash не зашифрований то , кожного разу, коли здійснюється запит, даний файл bash надсилається мережею, і він може бути перехвачений шкідливим програмним забезпеченням зловмисниками для отримання конференційних даних.

Далі, після того, як користувач системи буде автентифікований в ній, вводячи логін та пароль при кожному заходів в систему, команди списку використовують для того, щоб отримати інформацію про ідентифікатор кожного ресурсу для кожного сервісу, в даному випадку це ім’я «Maxym».

Для отримання даних, які містять інформацію про хмарні сервіси необхідно отримати всю інформацію про ідентифікатор кожного ресурсу, який в подальшому передається скрипту вже для визначеного хмарного сервісу, щоб отримати інформацію сервісу для нього. В наведеному нижче коді скрипту надаються персональні дані для запиту, а вже потім виконується команда зі списку. У наведеному сценарії, при використанні списку, програма запитує перших чотири стовпці з бази даних.

```
export IP_USERNAME=Maxym
export IP_PASSW0RD=****
export IP_REGI0N_NAME=KHMEL
export IP_TENANT_NAME=oleg
export IP_HOST=http://oblok.com.ua:5677
export IP_AUTH_URL=http:// test.com.ua.ca:8755/v3.1/
opens list|awk 'PS>4'|cut -a'|' -file1 -file2 -file3-file4|send \$ask
```

На нижче наведеному фрагменті запиту (Рисунок 3.3) показано отримані дані запиту отриманий в результаті виконання вище наведено скрипту мовою bash. У даному запиті коду в першому стовпці вдображається

інформація про ідентифікатор сервісу, а у наступному стовпці – назва ідентифікатора сервісу.

```
kadyans-MacBook-Pro:~ kadyan$ ./resourcename.sh
| 394d7a0b-2e09-4114-8553-165d04203432 | FDAServiceTester
| cc93da5a-4df2-4ae8-9039-495f03f54289 | GreenPower
| 53009673-2084-44db-bef9-fcc0f8cbabab | Kaleidoscope Gstreamer Services
| cf75600e-6624-4dd0-991a-0d2ce03919d9 | Kaleidoscope Locality Service
| a57cd697-631e-47ae-a911-b57f4a403b5b | Kaleidoscope Media Service II
| fd6797bf-cbe9-4225-a45f-f2719cbc08d3 | Kaleidoscope Registry Service
| 327dc476-40dc-4778-99ab-46ac7d61eda0 | Kaleidoscope SnapChat Billboard
| 993c5519-b4fa-4f05-b202-a02ec7cc2424 | Kaleidoscope Web Application
| 9005a84b-b937-4e2b-8b9d-42e9b6c892bf | Kaleidoscope-PubSubService
| fccd16f0-4e2b-4bae-b08f-98e9d7564241 | LivelyViz
| 3f810263-6093-4907-99c4-4736b758ad2d | SDN-Waseem
| 8059fb3a-f317-487d-8a60-d2c7b7697102 | TestKubernetes
| 8b5e4215-9e69-4817-a23b-0e0f590b9d3f | andikaleidoscope
| 35b4db70-9157-4a29-8da3-af718c62c610 | yakkit demo
```

Рисунок 3.3 – Фрагмент запиту інформації про хмарні сервіси

Наступним кроком, після зібрання інформації про ідентифікатор та ім’я сервісу для визначеного ресурсу, який отримуємо після виконання скрипта мовою bash списку opens відбувається передача інформації новому скрипту про ідентифікатор сервісу скрипту для конкретного ресурсу щоб отримати всю інформацію про послуги хмарного сервісу, як описано нами в попередньому розділі роботи.

У нижче наведеній частині програмного коду наведені команди bash в якому команди виконуються скриптом з обліковими даними, які ми отримали перед їх виконанням скриптом. Код наведений в цьому скрипті надає всю інформацію про характеристики хмарних сервісів, які необхідно моніторити та здійснювати їх аналіз. Зокрема, виконання цього скрита отримуємо результати, які пов'язані з певним ідентифікатором хмарного сервісу, що здійснюється кожною командою даного скрипту.

```
source /users /Maxym/ R_config ResourceId=$I
echo $ResourceId
```

```
bsh sample-list -• cpu -q resource_id=$ResourceId awk 'NR=4{  $7$11}‘
bsh sample-list hp* cpu_util -q resource_id $ResourceId awk 'NR=5{
$5$11}'
bsh sample-list -«disk.main.size -q resource_id»$ResourceId awk 'NR=5{
S7$11>'
bsh sample-list -a disk.read.bytes -q resource_id*$ResourceId awk 'NR=5{
$7$11}’
bsh sample-list hi disk.main.size -q resource_id $ResourceId awk 'NR=5{
S7$11>'
bsh sample-list hi disk.readwrite.bytes -q resource_id $ResourceId awk
'NR=*5{      $7$11>'

bsh sample-list -a disk.writeread.requests -q resource_id $ResourceId awk
'NR=5{       $7$11}'
bsh sample-list hi instance -q resource_id $ResourceId awk 'NR=5{
$7*11}'
bsh sample-list -m memory -q resource_id $ResourceId awk 'NR=5{
$7$11}'
bsh sample-list -■ vcpus -q resource_id»$ResourceId awk 'NR—5{
$7$11}>'
bsh sample-list -m network.incoming.bytes -q resource_id $ResourceId
awk 'NR==5{$7$11}'
bsh sample-list - network.size.bytes -q resource_id $ResourceId awk
'NR==5{{$7$11}'
```

Фрагмент програмного коду отриманої інформації (Рисунок 3.4) демонструє отримані характеристики для певного ідентифікатора сервісу. Список доступних послуг, який отриманий для певного ресурсу, це фактично

список лічильників наданий API при запиті даних. Список таких лічильників є унікальним для кожного хмарного сервісу.

```
kadyans-MacBook-Pro:~ kadyan$ ./resourcespecific.sh 394d7a0b
394d7a0b-2e09-4114-8553-165d04203432
1.7348047e+14ns
3.06557377049%
0.0GB
324574720.0B
20.0GB
10426356736.0B
426778.0request
1.0instance
2048.0MB
1.0vcpu
```

Рисунок 3.4 - Список використання хмарних сервісів

При виконанні таких запитів на сервері здійснюється асинхронне виконання скриптів, що продемонстровано в коді зображеному на рисунку 4.4 де наведений скрипт мовою bash здійснює публікацію отриманих даних за їх унікальним локатором хмарних сервісів.

```
var zapyt=function (motion, areos, callBack)
{
        var spaim=require('процес підпотоку').spawn;
        var iodyt_p=spawn(motion, areos);
        var resp= "_";
        var source_id_mas();
        iodyt_p.stdout.on('кількість', function (buffer)
                {
                        resp buffer.toStrring)
                        });
```

```
        iodyt_p.stdout.on('вихідний потік', function(error)
        {
        var lines=resp.split('\n');
        for (var source_id_mas in lines)
                {
                        source-id_mas.push(lines [source_id_mas]);
                }
                callBack (source_id_mas);
                });
        }
function source_nm
        run_cmd ("/Users/Maxym/sourcennm.sh",[], function (source_id_mas)
        {
        for (var source_id in (source_id_mas))
        {
                var motionFiltered= source_id_mas[source_id).replace(/[|]a/on, "");
                var motion Split= motion Filtered.split!(\n');
                if(motion Split(0))
                        {
                        source_nm [motionSplit ("_")[0].split("_")[1]
                        + '_' + motionSplit [0].split("_")[0].split('_')[0];
                        source_nm_z[motionSplit[0].split("_")[1]  ]
                        + '_' + motionSplit [0].split("_")[0].split('_')[0]
                        = motionSplit[0].split("_")[0];
                        if (motionSplit [config. source_id == '_')
                        {
                        return;
                        }
                }
        }
    });
});
```

3.2 Програмна реалізація інформаційної веб-панелі

У даному пункті опишемо як здійснена реалізація інформаційної панелі та, зокрема, як здійснюється обробка інформації нею з інших складових програмного комплексу. У другому розділі ми описували , як виводяться інформація за допомогою скриптів і запитів API та виконання їх на сервері додатків. Тут покажемо роботу розробленої нами інформаційної панелі, зокрема як реалізований інтерфейс користувача та способи моніторингу хмарних сервісів та їх параметрів.

Сервер додатків та сама інформаційна система розроблялась з допомогою фреймворку Node.js, який розгортається в харному сервісі, що надає послуги віртуальних машин.

На рисунку 3.5 наведено інтерфейс інформаційної веб-панелі панелі, яка відображає дані про використання певним користувачем хмарного сервісу та відображає інформацію про використання його ресурсів.

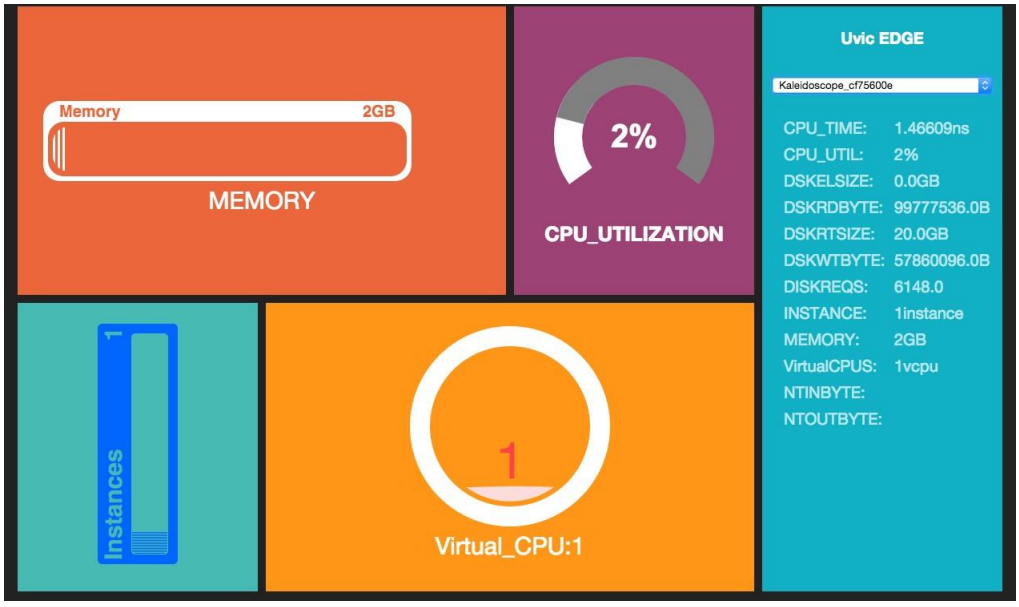


Рисунок 3.5 – Інформаційна веб-панель моніторингу хмарних сервісів.

Як бачимо наша інформаційна веб-панель має п'ять віджетів – окремих веб-застосунків, які розміщені на динамічній веб-сторінці. Такий віджет надає інформацію в графічному вигляді на сторінці що відображається браузером.

Кожен такий віджет надає інформацію характерну для певного параметра хмарного сервісу. Зокрема, на даному рисунку відображаються п'ять віджетів наступні характеристики хмарного сервісу, який надає послуги по використанню віртуальних машин:

– Пам'ять - відображає інформацію про кількість виділеної пам’яті, та використану пам'ять користувачем цього хмарного сервісу. Інормація про пам'ять виводиться гігабайтах;

– Процесор - відображає інформацію про викристання ресурсів процесора. Даний віджет можна налаштувати в інформаційні панелі за допомогою спеціальних команд. Одицями вимірювання виступають відсотки навантаження процесора.

o Диск – надає інформацію про використання місця на диску. Інформація представлена у вигляді графіку заповнення диску;

– Віртуальний процесор – надає інформацію про кількість використаних віртуальних процесорів. Інформація про їх кількість подається у вигляді звичайних цифр;

– Загальна інформація – в текстовому вигляді надається детальна інформація про використання всіх ресірсів хмарного сервісу;

Для отримання такої інформації про використання певного хмарного сервісу потрібно з меню інформаційної панелі обрати тип хмарного сервісу і далі обрати вже конкретний хмарний сервіс, щоб переглянути інформацію для нього.

Для більш детального розгляду роботи інформаційної панелі, розглянемо принцип її роботи на прикладі публікації даних. Публікація даних.

Так, інформаційна панель здійснює запит даних та інформації для отримання всіх хмарних сервісів використовуючи метод GET. Наведений нижче фрамент скрипта здійснює запит на вед-адресу конкретного хмарного сервісу звідки він бере всі імена про надані ресурси.

```
$.get('/resource', function(data)
```

```
{
ko.applyBindings(new masterVM(data));
};
var masterVM = function(data) {
resourceListModel = new resourceListModel(data);
serverDataModel = new serverDataModel();
d3Model = new d3Model();
};
```

Сам запит відправляється на URL-адресу хмарного сервісу, де розташована інформація про ключ, та ім'я ресурсу. Запит співставляє відправлені значення з ідентифікатором певного ресурсу. Після виконання запиту, інформація про імена ресурсів передаються в інформаційну панель та відображаються на веб-сторінці у відповідних віджетах. Наведений нижче фрагмент програмного коду скрипта показує веб-адресу, де розміщено містить всі параметри хмарних сервісів та відображає дані у табличній формі у вигляді двох полів – ключ та ім'я.

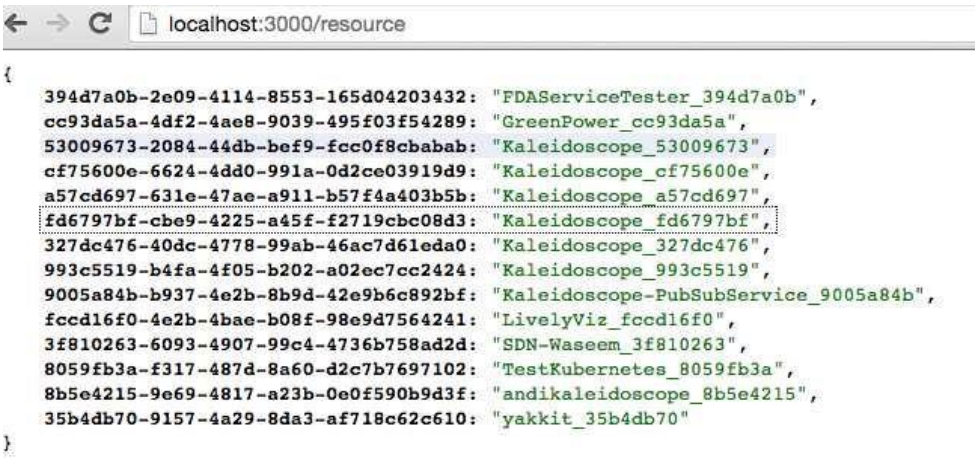


Рисунок 3.6 - Інформація про надані ресурси

Запит також надходить на сервер додатків з допомогою скрипта та на основі запитуваної веб-адреси ресурсу відповідь надається користувачеві через веб-інтерфейс.

Наведений нижче фрагмент програмного коду скрита демонструє показує яким чином обробник сервера додатків запиту GET робить запит даних та яким чином отримані відповіді з інформацією надходить до кожного певного вхідного запиту. Тобто, коли запит відправляється на певний веб-ресурс, прописаний в запиті GET, то сервер додатків дає відповідь з інформацією із запитуваної в запиті веб-адреси.

```
app.get(’/serverdata1, function(reg, res) { res.json(serverData);
});
app.get('/serverdata/:id', function(reg. res)
        {
         res.j son{serverData[ reg.pa rams.id]);
        });
        app. get( ’/resource1. function (reg. res)
        {
        res.j son(resourceName);
        });
app. get(' /resourceNameInverse1.function(reo. res)
{
// parses запаитує url
var theUrl = url.parse{ reg. url );
// надає запит частини URL і parses створє цей обєкт
        var resourceName = theUrl.query;
        var ResourceID=resourceWameInverse[resourceName];
        var serverSpecificData=serverData[ResourceID];
        res.send(serverSpecificData);
        console.log(serverSpecificData);
        res.json(resourceNameInverse);
};
app.get(’/getServers', function(rea. res)
{
run_cmd{"source panel config".[1.function(text) { console.loo(text)
});
});
```

Крім вище описаних скриптів на мові bash при розробці нашої інформаційної панелі було використано технологію AJAX для виконання асинхронних запитів (мовою JavaScript мовою розмітки і опрацювання даних XML). Ця технологія використовується для розробки асинхронного веб-застосунку, який виконується не на сервері, а на стороні клієнта. Технологія AJAX дозволяє веб-застосунку відправляти та отримувати дані з сервера додатків в основі якого лежить фреймворк Node.js та асинхронно, без передачі інформації на екран і відповідно веб- сторінки. Це, в свою чергу дозволяє додатку на стороні клієнта змінювати веб-сторінки або їх частини без необхідності втручання в роботі іншої частини веб-сторінки, яка відображається в браузері. Скрипти AJAX можуть працювати як явно так і в фоновому режимі не порушуючи роботу інших віджетів веб-сторінки інформаційної панелі..

В нашому випадку скрипти написані з використанням технології AJAX використовуються в роботі інформаційної панелі для передачі даних та отримання даних з сервером додатків.

В нижче наведеному фрагменті скрипта наведено запит з використанням технології AJAX, де запит надходить на веб-адресу яка вказану програмному коді. Поряд з запитами AJAX ми відбувається перехід на сервера за іменем ресурсу, який запитує інформацію про ресурси хмарного сервісу. Скрипти на технології AJAX мають різні формати викликів. Так тип запиту може приймати відповідно значення двох типів GET або POST. В цих типах запитів вказується веб-адреса, з якої буде запитуватись необхідна інформація. Після вдалого виконання такого запиту завжди виконується функція про успішне завершення виконання програмного коду. Продемонстрований програмний код показує, як в інформаційну панель надходить інформація про всі використані ресурси хмарного сервісу одразу при запиті назви ресурсу.

```
shouter.subscribe(function(serverName)
{
$.ajax
```

```
{
type: 'GET;
data: serverName;
contentType: "application/j son";
dataType:json;
url:"http://localhost:3000/resourceNameInverse"f success: function(data)
        {
        updateServerdata(data); canvas(data);
        barstackeridata); liquidFillGaugeidata);
        console.log(data);
        };
error: function(error)
        {
        console.log("some error in fetching the notifications");
        }
});
        console.log('D3 Model Refreshed');
        {
        this, "messageToPublish");
        }
};
```

3.3 Висновки

Уданому розділі загалом було здійснено програмну реалізацію програмного комплексу моніторингу хмарних сервісів та здіснено програмну реалізацію інтерфейсу у вигляді інформаційної панелі.

Так, зокрема, було подано продемонстровано розробку програмного комплексу за чотирма етапами: проектування архітектури, використання технологій та інструментів задіяних в розробці, розробки API для реалізації програмного комплексу та розробки інтерфейсу користувача інформаційної панелі.

В розділі наведено спроектовану високорівневу архітектуру інформаційної веб-панелі та програмного комплексу загалом. Описано роботу всіх складових зпроектованої архітектури.

Крім того у розділі наведено та описано реалізацію програмного комплексу на основі вибраних програмних та технічних засобів розробки та показано взаємодію модулів з розробленим API.

Детально описано реалізацію інтерфейсу користувача інформаційної панелі моніторингу хмарних сервісів та їх складових.

4 ДОСЛІДЖЕННЯ ТА ТЕСТУВАННЯ ЕФЕКТИВНОСТІ УДОСКОНАЛЕННЯ МЕТОДУ АДМІНІСТРУВАННЯ ХМАРНИХ СЕРВІСІВ

4.1 Перевірка методу на тестових даних

Оцінку розробленого нами вдосконаленого методу моніторингу хмарних сервісів та розробленого нами програмного комплексу у вигляді інформаційної веб-панелі будем здійснювати в порівняні з єдиною доступною в веб-інформаційною панеллю створеною на основі OpenStack, яка на відміну від спроектованої не надає такої детальної інформації про використання ресурсів сервісів. Оцінку будемо здійснювати на основі показників стандарту якості програмних продуктів визначених у таблиці 1.1 першого розділу нашої дипломної роботи.

А саме, нами було здійснювалась оцінка розробленого програмного комплексу за такими параметрами:

– сумісність, як здатність ПЗ виконувати однакові програми з отриманням таких самих результатів;

– модульність - розділення ПЗ на окремі модулі таким чином, що рівень їх залежності був найменшим при зміні в одному модулі;

– надійність - здатність зберігати протягом певного часу значення визначених параметрів;

– захищеність - здатність до захисту від взлому стороннім ПЗ та їх втручанням в роботу системи;

– супроводжуваність - здатність до покращення та оптимізації програмного забезпечення;

– модифікованість - здатність до зміни без виникнення пошкоджень в роботі, або погіршення роботи функціоналу;

– тестованість - можливість виконання тестових наборів та можливість виконання тестів для перевірки виконання вимог до функціонування.

Крім цих основних властивостей, також, розроблений наш комплекс оцінювали більш детально ще й за такими параметрами як продуктивність, безпека, та інтервали оновлення

Так продуктивність розробленої інформаційної панелі оцінювалась на базі часу відгуку для кожного набору параметрів хмарних сервісів, якіі моніторились системою. Для пришвидшення роботи додатку увага приділялась як розробленому API, а також значна увага приділялась налаштуванню бази даних для пришвидшення роботи запитів, які пришвидшили роботу як вихідних запитів так і запитів на вибірку даних.

Безпека роботи програми перевірялась нами, оскільки вона головною проблемою будь-яких застосунків такого характеру. В саме перевірялась робота скриптів які містить ім'я користувача, пароль та інші облікові користувача. Проблема полягала в тому, що такі скрипти написані мовою bash, а вони не шифруються не шифрується при надсиланні за кожним запитом. Ця проблема була вирішена на рівні захисту мережевих протоколів даних.

Інтервал оновлення нами визначався як проміжок часу визначений між отриманням даних та відображанням їх далі в інформаційній панелі. Загалом наш програний комплекс був налаштований так, щоб інформація обпитувалась в межах однієї хвилини та менше. Даний проміжок часу був обраний нами для отримання даних одного хмарного сервісу. Мінімальний час, для запитів було нами далі зведено до 30-60 секунд. Це досить прийнятний час при використанні інформаційної панелі користувачами, враховуючи, що дані для аналізу та обробки не використовують синхронно, а лише менеджерами системи в певний час для прийняття рішень, щодо збільшення чи зменшення наданих ресурсів користувачам та ресурсів для забезпечення роботи інформаційної системи підприємства.

При проведені тестувань ми використовували дані для дослідження, які генерувалися на основі реальних опрацьованих даних, отриманих на основі використання організацією та користувачами організації хмарних сервісів та хмарних технологій.

4.2 Оцінка за критерієм якості часу відгуку

Перше наше тестування проводилось для визначення часу реагування шляхом збільшення сервісів на тестовому середовищі, де саме було встановлено час реакції для отримання інформації про дані хмарних сервісів. Тестування проводилось багато разів для того, щоб перевірити масштабованість системи та визначити, яким чином збільшувався час роботи та відгуку в залежності від кількості збільшення наданих ресурсів, які відстежувались інформаційною панеллю згідно вдосконаленого алгоритму адміністрування хмарних сервісів та хмарних обчислень проведених у нашому дослідженні в ході роботи над дипломною роботою.

У наведеній таблиці 4.1 подано результати роботи інформаційної веб-панелі за в ході моніторингу від одного до п’ятдесяти хмарних сервісів та хмарних обчислень .

Таблиця 4.1 – Результати тестування хмарних сервісів

Номер тесту	Кількість хмарних сервісів	Реакція системи
1	1	21 секунда
2	5	23 секунд
3	10	31 секунда
4	15	35 секунд
5	20	37 секунд
6	25	39 секунд
7	30	41 секунда
8	35	42 секунди
9	40	42 секунди
10	45	42 секунди
11	50	42 секунди

У даній таблиці приведено результати тестування, які проводились одинадцять раз, зі зміною кількості сервісів та часу відгуку при кожній зміні кількості сервісів. З даної таблиці ми можемо бачити, що для одного сервісу час, який необхідний для отримання даних про надані послуги, становить двадцять одну секунду. Проте, як видно з таблиці, якщо кількість сервісів зростає то час стає стабільним і час відгуку не зростає. Тобто, як видно з таблиці, вже при збільшені кількості сервісів до 20 і більше – час відгуку вже не збільшується. Дане дослідження підтверджує наше припущення, про досконаліть нашого вдосконаленого методу адміністрування хмарних сервісів та обчислень.

Основною переваго даного методу, на нашу думку, а також як підтверджують результати експерименту, полягає у тому, що скрипти виконуються на сервері додатків, і саме це забезпечує швидкодію виконання запитів даних до бази даних. Згідно стандартів якості оцінки програмного забезпечення робить нашу інформаційну веб-панель масштабованою та надійною. При збільшені кількості сервісів не відбувається збільшення навантаження на сам сервер додатків, оскільки, згідно нашої моделі, опитування здійснюється асинхронно.

Головним недоліком, в тестуванні було обмеження тестування пов’язане з OpenStack, яке не дозволяло провести тестування не більше 50 сервісів. Проте, надана можливість протестувати його на такій кількості сервісів доводить надійність нашого методу.

4.3 Оцінка за критерієм якості масштабованості

Дана оцінка була проведена для визначення критерію якості масштабованості від кількості користувачів системи, які мають можливість можуть спільно користуватися розробленим нашим сервісом.

Для виконання даного тесту, сервер додатків був запущений на хмарному сервісі, який надає послуги віртуальних комп’ютерів з виділеними процесорами, оперативною пам’яттю та фізичним сховищем даних. Після налагодження цього сервера, на ньому було встановлено веб-сервер, який здійснював запуск нашої інформаційної панелі. Саме на цьому сервері було здійснене дане тестування масштабованості.

Головною метою даного експерименту нами було поставлено завдання тестування навантаження роботи розробленої інформаційної веб-панелі. Основним параметром, який було визначено для тестування навантаження гідно стандарту є кількість користувачів інформаційної панелі. Як покаказали результати тестування, навантаження на сервер досить невеликі, навіть при великій кількості користувачів. У таблиці 4.2 наведено результати тестування навантаження.

Таблиця 4.2 – Результати тестування навантаження

Номер тесту	Кількість користувачів	Час відгуку
1	1	0,2 секунди
2	5	0,2 секунди
3	10	0,1 секунди
4	15	0,1 секунди
5	20	0,1 секунди
6	25	0,1 секунди
7	30	0,1 секунди
8	35	0,1 секунди
9	40	0,1секунди
10	50	0,1 секунди

З даної таблиці видно, що при збільшені кількості користувачів інформаційної панелі затримки в часі немає, за винятком затримок для малої кількості користувачів, а саме, коли сервер обробляти дані. Дана початкова

затримка в отримані даних для сервісів обумовлена саме роботою сервісів в операційній системі та роботою обладнання, обумовлених їх технічними характеристиками та обмеженнями трафіку.

4.4 Легкість використання користувачами та зворотній зв'язок з розробником

У даному проведеному експерименті ми оцінюється досвід використання користувачами. Даний тест показує наскільки якісний аналіз пророблені роботи для користувача і яким чином забезпечується зворотний з розробниками про поліпшення та вдосконалення програмного засобу.. Задоволеність користувачів від користування є досить важливим критерієм оцінювання, який розробники застосовують для оцінки розробленого програмного забезпечення. Зокрема, нами було проведено аналіз, на основі відгуків користувачів про використання розробленої інформаційної панелі. Основна увага при вимірювані задоволеності користувачів приділялась параметрам які наведені у таблиці 4.3 де булав встановлена шкала для кожного параметра від одного до десяти балів, де нуль – повне незадоволення, десять – повністю задоволений.

Таблиця 4.3 – Параметри оцінювання користувачами розробленої системи моніторингу хмарних сервісів

Параметр	Опис
Корисність	Функціонал системи має відповідати вимогам користувача та відповідати вимогам до використання подібних систем
Здатність до використання	Розроблений програмний засіб повинен бути легким у розумінні

	функціоналу та зручним у використанні
Задоволеність	Такі елементи інтерфейсу, як меню, значки, піктограми та графічнні зображення не повинні викликати негативних емоцій і бути легко сприйнятливими користувачем
Здатність до легкого пошуку необхідної інформації та функціональних можливостей	Весь функціонал системи повинен мати легку навігацію та легко локалізуватись
Доступність	Наскільки легко можна користувачу дістатись потрібного функціоналу
Достовірність	Користувачі мають бути впевнені в достовірності інформації, яка відображається в панелі

За результатами опитуваного проведення нами було отримано результати наведені у таблиці 4.4

Таблиця 4.4 – Оцінка роботи інформаційної панелі користувачами

Параметр	Оцінка
Корисність	10
Здатність до використання	9
Задоволеність	9
Здатність до легкого пошуку необхідної інформації та функціональних можливостей	8
Доступність	10
Достовірність	10

Як видно з таблиці, за результатами опитування користувачів, розроблена нами інформаційна панель моніторингу хмарних сервісів отримала досить високу оцінку. Дещо занижена оцінка за параметром «Здатність до легкого пошуку необхідної інформації та функціональних можливостей» обумовлена на нашу думку тим, що таких продуктів на ринку IT-технологій досить мало і користувачі не звикли користуватись такими додатками.

4.5 Висновки

При проведені дослідження та оцінку вдосконаленого методу адміністрування хмарних сервісів в даному розділі нами було використано експериментально отримані тестові набори даних з метою перевірки розробленої інформаційної панелі адміністрування хмарних сервісів та ефективності розробленого вдосконаленого методу.

У розділі було здійснено тестування методу, дослідження його роботи та оцінка розробленого вдосконаленого методу моніторингу хмарних сервісів організацією. Результати тестування вказують, що розроблений нами вдосконалений метод на моніторингу хмарних сервісів має високу корисність, здатність до використання, задоволеність від роботи користувачами, здатність до легкого пошуку необхідної інформації та функціональних можливостей, доступність та достовірність.

В результаті отриманих даних експериментальних досліджень запропонованого нами методу було з’ясовано, що він також має високу швидкість роботи без втрати своєї ефективності, а також тестованість та масштабованість за рахунок використання розробленого нами вдосконаленого методу моніторингу хмарних сервісів.

ВИСНОВОК

В ході дипломної роботи було досліджено що реалізація моніторингу хмарних сервісів надала можливість виокремити певні ключові недоліки сучасних методів моніторингу хмарних сервісів.

В результаті виконання дипломної роботи нами було реалізовано та вдосконалено метод моніторингу хмарних сервісів, розроблено та протестовано програмний комплекс.

Так, зокрема, в першому розділі нашої дипломної роботи було показано, що в умовах сьогодення хмарні технологій є досить є досить затребуваною, необхідною та значимою технологією .

В першому цьому розділі було проведено аналіз предметної області, дослідження невирішених проблем в наявних сучасних дослідженнях. Наведено приклади та показано, що різним організаціям потрібен такий продукт. Було окреслено проблему невирішеного раніше рішення, а саме показано, що, основним головним моментом при використанні хмарних ресурсів та хмарних обчислень є те, що компаніям доводиться постійно контролювати та коригувати їх використання та роботу, для визначення навантаження на кожен тип ресурсу ресурс при використанні їх в своїй діяльності використання IT-технологій, а також необхідність аналізу і обробки використання хмарних сервісів.

У першому розділі також проведено порівняльний аналіз переваг та недоліків існуючих рішень на основі аналізу наукових праць по даній темі. Здійснено постановку задачі реалізації покращення методу моніторингу хмарних сервісів. Наведено обґрунтування реалізація нового підходу до вдосконалення методу моніторингу хмарних сервісів інформаційної панелі на основі визначених основних систем стандартів та певних правил.

У другому розділі проведено дослідження та аналіз існуючих методів моніторингу хмарних сервісів та хмарних технологій. На основі досліджених

методологій було запропоновано власну методологію, яка допоможе краще користувачам здійснювати моніторинг хмарних сервісів.

Загалом у розділі розглянуто різні підходи до розробки програмного комплексу для моніторингу хмарних ресурсів. Розглянуто різні програмні засоби моніторингу для адміністрування хмарних сервісів і пов'язані з ним роботи з різними інформаційними панелями для віртуальних машин в середовищі хмарних обчислень. У розділі здійснено аналіз методів та програмних засобів, що використовуються при розробці інформаційної панелі, та запропоновано внутрішню архітектуру для програмного комплексу моніторингу хмарних сервісів.

У третьому розділі загалом було здійснено програмну реалізацію програмного комплексу моніторингу хмарних сервісів та здійснено програмну реалізацію інтерфейсу у вигляді інформаційної панелі.

Так, зокрема, було подано продемонстровано розробку програмного комплексу за чотирма етапами: проектування архітектури, використання технологій та інструментів задіяних в розробці, розробки API для реалізації програмного комплексу та розробки інтерфейсу користувача інформаційної панелі. Наведено спроектовану високорівневу архітектуру інформаційної веб-панелі та програмного комплексу загалом. Описано роботу всіх складових зпроектованої архітектури.

Також у третьому розділі наведено та описано реалізацію програмного комплексу на основі вибраних програмних та технічних засобів розробки та показано взаємодію модулів з розробленим API. Детально описано реалізацію інтерфейсу користувача інформаційної панелі моніторингу хмарних сервісів та їх складових.

У четвертому розділі здійснено тестування методу, досліджено його роботу та здійснена оцінка розробленого вдосконаленого методу моніторингу хмарних сервісів організацією. Як видно з результатів тестування, що розроблений нами вдосконалений метод на моніторингу хмарних сервісів має високу корисність, здатність до використання, задоволеність від роботи

користувачами, здатність до легкого пошуку необхідної інформації та функціональних можливостей, доступність та достовірність.

В результаті отриманих даних експериментальних досліджень у четвертому розділі з запропонованим нами методом було з’ясовано, що він має високу швидкість роботи без втрати своєї ефективності, високу тестованість, масштабованість саме завдяки використанню розробленого нами вдосконаленим методом моніторингу хмарних сервісів.

Таким чином, в ході дипломного проектування було вдосконалено метод моніторингу хмарних сервісів та здійснено реалізацію програмного комплексу на основі розробленого методу програмного комплексу моніторингу хмарних сервісів та їх складових.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Deshpande P. Use-case centric dashboard for cloud solutions / P.Deshpande, S. Sharma, A. Acharya, K.Beaty, A. Kundu. In 2014 IEEE International Conference on Services Computing (SCC), pp. 840–841. IEEE, 2014.

2. Kai Lei. Performance comparison and evaluation of web development technologies in php, python, and node.js./ Kai Lei, Yining Ma, Zhi Tan. Computational Science and Engineering, 2014 IEEE 17th International Conference, pp. 661–668, Dec 2014.

3. Rosado T.. An overview of openstack architecture/ T. Rosado, J. Bernardino. In Proceedings of the 18th International Database Engineering & Applications Symposium, pp. 366–367. ACM, 2014.

4. Ferreira L. Cloudlet architecture for dashboard in cloud and ubiquitous manufacturing./ L. Ferreira, G.Putnik, M. Cunha, Z. Putnik, H. Castro, C. Alves, Vaibhav Shah, and Maria Leonilde R Varela. Procedia CIRP, pp. 366–371, 2013.

5. System Architecture https://docs.openstack.org/ceilometer/latest/contributor/architecture.html

6. Ceilometer Documentation https://docs.openstack.org//ceilometer/latest/doc-ceilometer.pdf

7. Хуан Анхель Лоренцо дель Кастільо, Кейт Маллічан і Яхья Аль-Хазмі. Федерація Openstack в експериментах мультихмарних тестових стендів. In Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5-й міжнародний висновок С, том 2, с.51–56. IEEE, 2013.

8. Пол Беррі.Head First. Python. Харків:Фабула. – 2021. – 624с. ISBN : 978-617-522-019-1

9. Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE — Режим доступу до ресурсу: https://iso25000.com/index.php/en/iso-25000-standards (дата звернення 11.09.2022). – Назва з екрану.

10. Роберт Мартін. Чиста архітектура. Харків:Фабула. – 2019. – 368с. ISBN : 978-617-09-5286-8

11. Роберт Мартін. Чистий кода. Харків:Фабула. – 2019. – 416с. ISBN : 978-617-09-5285-1

12. Олексій Васильєв. Програмування мовою Python. Тернопіль: НАВЧАЛЬНА КНИГА – БОГДАН. - 2019. – 504 с. ISBN : 9789661056113

13. Елізабет Робсон , Ерік Фрімен. Head First. Патерни проєктування. Харків:Фабула. – 2020. – 672с. ISBN : 978-617-09-6159-4

14. Елізабет Робсон, Ерік Фрімен. Head First. Програмування на JavaScript. Харків:Фабула. – 2022. – 672с. ISBN : 978-617-522-047-4

15. Роб Коул. Блискучий Agile. Харків:Фабула. – 2020. – 192с. ISBN : 978-617-09-6381-9

16. Берт Бейтс , Кеті Сьєрра. Head First. Java. Харків:Фабула. – 2022. – 720 с. ISBN : 978-617-522-033-7

17. Юрій Рамський, Василь Олексюк, Анатолій. Адміністрування комп’ютерних мереж і систем. Тернопіль: НАВЧАЛЬНА КНИГА – БОГДАН. - 2020. – 196 с. ISBN : 9789661015615

18. Надія Балик, Віктор Мандзяк. MySQL: лабораторний практикум. Тернопіль: НАВЧАЛЬНА КНИГА – БОГДАН. - 2018. – 88 с. ISBN : 9789664084267

19. Емі Вебб. Велика дев'ятка. Як IT-гіганти та їхні розумні машини можуть змінити людство. Харків:Vivat. – 2020. – 352 с. ISBN : 9789669822185

20. Денис Каплунов. Королі соціальних мереж. .Київ: BookChef – 2022. – 432 с. ISBN: 9786175480922

21. Openstack. Базова архітектура openstack. http://docs.openstack.org/ розробник/цейометр/архітектура.html.

22. Форкун Ю. В. Методичні вказівки до виконання курсового проекту для студентів напряму підготовки “Інженерія програмного забезпечення” з дисципліни «Архітектура та проектування програмного забезпечення» / Ю. В. Форкун. – Хмельницький: ХНУ, 2018. – 42 с..

23. Томашевський В.М. Моделювання систем: Підручник.-К.:Видавнича група BHV.-2015.-352с.-Інформатика.-966-552-120-9

24. Мінухін С.В., Беседовський О.М., Знахур С.В. Методи і моделі проектування на основі сучасних CASE- засобів: навч. посіб.-Харків:ХНЕУ.-2018.-272с.-978-966-676-301-6

25. Тіаго Росадо і Хорхе Бернардіно. Огляд архітектури openstack. У працях 18-го Міжнародного симпозіуму з розробки баз даних та додатків, сторінки 366–367. ACM, 2014.

26. Омар Сефрауї, Мохаммед Айссауї та Мохсін Елеульдж. Openstack: до рішення з відкритим кодом для хмарних обчислень. Міжнародний журнал комп'ютерних застосувань, 55(3):38–42, 2012

27. Пралхад Дешпанде, Шачі Шарма, Аруп Ачар'я, Кірк А Біті та Ашіш Кунду. Інформаційна панель, орієнтована на використання, для хмарних рішень. У 2014 році Міжнародна конференція IEEE з сервісних обчислень (SCC), сторінки 840–841. IEEE, 2014.

28. Мартв Арлітті, Суджата Банерджі, Каллен Баш, Юань Чен, Даніель Гмах, Крістофер Гувер, Прія Махадеван, Деян Мілоїчич, Ерік Пеллетьє, Р. Н. Вішванатх та ін ал. Інформаційна панель сталого розвитку хмари. In Sustainable Systems and Technology (ISSST), 2010 IEEE International Симпозіум на, сторінки 1–1. IEEE, 2010..

29. Максимів О.В., Форкун Ю. В. АДМІНІСТРУВАННЯ ХМАРНИХ СЕРВІСІВ //Збірник наукових праць за матеріалами XIV Всеукраїнської науково-практичної конференції «Актуальні проблеми комп’ютерних наук АПКН-2022». – Хмельницький. – 2022. – С. 181-185..

ДОДАТОК А

(обов'язковий)

ПРОГРАМНИЙ КОД

А. 1 Програмний код використання бібліотеки API

```
var serverDataModel = function () {
this.message = ko.observable    (“ показує дані сервера  ");
shouter.subscribe(function(newValue)
{
        this.message(newValue);
}, this, "messageToPublish");
};
var resourceListModel = function(resourceList)
{
        resourceListOptions = ko.observableArray();
        var resourceListArray = Object.keys(resourceList).map(function (key)
        {
                return resourceList[key]
        });
for(var resourceIndex in resourceListArray)
{
        resourceListOptions.push(resourceListArray[resourceIndex]);
}
        resourceData = ko.observable();
        resourceData.subscribe(function(newValue) {
        shouter.notifvSubscribers(newValue. "messageToPublish");
        console.log(‘newValue’);
};
};
        var getResourcelDData = function (resourceName)
        {
         console.log('Selected Value ' + resourceName):
        };
```

```
$.get('/resource', function(data)
{
        ko.applyBindings(new masterVM(data));
});
var masterVM = function(data)
{
        resourceListModel = new resourceListModeli(data): serverDataModel = new
        serverDataModeli); d3Model = new d3Model();
});
```

A.2. bash скрипт

```
export IP_USERNAME=Maxym
export IP_PASSW0RD=****
export IP_REGI0N_NAME=KHMEL
export IP_PAIN_BACK=Oleg

export IP_USERNAME=Maxym
export IP_PASSW0RD=****
export IP_REGI0N_NAME=KHMEL
export IP_TENANT_NAME=oleg
export IP_HOST=http://oblok.com.ua:5677
export IP_AUTH_URL=http:// test.com.ua.ca:8755/v3.1/
opens list|awk 'PS>4'|cut -a'|' -file1 -file2 -file3-file4|send \$ask
```

А.3. Програмний код виведення інформації про характеристики хмарних сервісів

```
source /users /Maxym/ R_config ResourceId=$l
echo $ResourceId
bsh sample-list -• cpu -q resource_id=$ResourceId awk 'NR=4{   $7$11}‘
bsh sample-list hp* cpu_util -q resource_id $ResourceId awk 'NR=5{
$5$11}'
bsh sample-list -«disk.main.size -q resource_id»$ResourceId awk 'NR=5{
S7$11>'
bsh sample-list -a disk.read.bytes -q resource_id*$ResourceId awk 'NR=5{
$7$11}’
bsh sample-list hi disk.main.size -q resource_id $ResourceId awk 'NR=5{
S7$11>'
bsh sample-list hi disk.readwrite.bytes -q resource_id $ResourceId awk
'NR=*5{      $7$11>'
bsh sample-list -a disk.writeread.requests -q resource_id $ResourceId awk
'NR=5{       $7$11}'
bsh sample-list hi instance -q resource_id $ResourceId awk 'NR=5{
$7*11}'
bsh sample-list -m memory -q resource_id $ResourceId awk 'NR=5{
$7$11}'
bsh sample-list -■ vcpus -q resource_id»$ResourceId awk 'NR—5{
$7$11}>'
bsh sample-list -m network.incoming.bytes -q resource_id $ResourceId
awk 'NR==5{$7$11}'
bsh sample-list - network.size.bytes -q resource_id $ResourceId awk
'NR==5{{$7$11}'
```

А.5. скрипт мовою bash публікації отриманих даних

```
var zapyt=function (motion, areos, callBack)
{
        var spaim=require('процес підпотоку').spawn;
        var iodyt_p=spawn(motion, areos);
        var resp= "_";
        var source_id_mas();
        iodyt_p.stdout.on('кількість', function (buffer)
                {
                        resp buffer.toStrring)
                        });
        iodyt_p.stdout.on('вихідний потік', function(error)
        {
        var lines=resp.split('\n');
        for (var source_id_mas in lines)
                {
                        source-id_mas.push(lines [source_id_mas]);
                }
                callBack (source_id_mas);
                });
        }
function source_nm
        run_cmd ("/Users/Maxym/sourcennm.sh",[], function (source_id_mas)
        {
        for (var source_id in (source_id_mas))
        {
                var motionFiltered= source_id_mas[source_id).replace(/[|]a/on, "");
                var motion Split= motion Filtered.split!(\n');
                if(motion Split(0))
                        {
                        source_nm [motionSplit ("_")[0].split("_")[1]
```

```
                    + '_' + motionSplit [0].split("_")[0].split('_')[0];
                    source_nm_z[motionSplit[0].split("_")[1]  ]
                    + '_' + motionSplit [0].split("_")[0].split('_')[0]
                    = motionSplit[0].split("_")[0];
                    if (motionSplit [config. source_id == '_')
                    {
                    return;
                    }
                }
            }
        });
});
```

А.6 фрамент скрипта запиту на веб-адресу конкретного хмарного сервісу

```
$.get('/resource', function(data)
{
ko.applyBindings(new masterVM(data));
};
var masterVM = function(data) {
resourceListModel = new resourceListModel(data);
serverDataModel = new serverDataModel();
d3Model = new d3Model();
};
```

А.7 фрамент скрипта обробника сервера додатків запиту GET

```
app.get(’/serverdata1, function(reg, res) { res.json(serverData);
});
app.get('/serverdata/:id', function(reg. res)
        {
         res.j son{serverData[ reg.pa rams.id]);
        });
        app. get( ’/resource1. function (reg. res)
        {
        res.j son(resourceName);
        });
app. get(' /resourceNameInverse1.function(reo. res)
{
// parses запаитує url
var theUrl = url.parse{ reg. url );
// надає запит частини URL і parses створє цей обєкт
        var resourceName = theUrl.query;
        var ResourceID=resourceWameInverse[resourceName];
        var serverSpecificData=serverData[ResourceID];
        res.send(serverSpecificData);
        console.log(serverSpecificData);
        res.json(resourceNameInverse);
};
app.get(’/getServers', function(rea. res)
{
run_cmd{"source panel config".[1.function(text) { console.loo(text)
});
});
```

А.8. фрагмент скрипту виведення інформації про всі використані ресурси хмарного сервісу

```
shouter.subscribe(function(serverName)
{
$.ajax
{
type: 'GET;
data: serverName;
contentType: "application/j son";
dataType:json;
url:"http://localhost:3000/resourceNameInverse"f success: function(data)
        {
        updateServerdata(data); canvas(data);
        barstackeridata); liquidFillGaugeidata);
        console.log(data);
        };
error: function(error)
        {
        console.log("some error in fetching the notifications");
        }
});
        console.log('D3 Model Refreshed');
        {
        this, "messageToPublish");
        }
};
```

ДОДОАТОК Б

(обов’язковий)

КОПІЇ НАУКОВИХ ПУБЛІКАЦІЙ

#Paper_APKN_2022_059

УДК 004.65

Максимів О.В., Форкун Ю.В.

Хмельницький національний університет

МЕТОДИ ТА ПРОГРАМНІ ЗАСОБИ МОНІТОРИНГУ АДМІНІСТРУВАННЯ ХМАРНИХ СЕРВІСІВ

Розглянуто основні методи та програмні засоби моніторингу адміністрування хмарних сервісів.. Виконано порівняння та аналіз найбільш популярних різних підходів, які були прийняті при створенні інформаційних панелей для моніторингу хмарних сервісів іншими дослідниками. Здійснено аналіз методів та програмних засобів, що використовуються при розробці інформаційної панелі, та запропоновано внутрішню архітектуру для інформаційної панелі моніторингу хмарних обчислень з використанням технологій Веб 3.0.

The main methods and software tools for monitoring the administration of cloud services are considered. A comparison and analysis of the most popular different approaches, which were adopted when creating information panels for monitoring cloud services by other researchers, is performed. An analysis of the methods and software tools used in the development of an information panel was carried out, and we offer an internal architecture for a cloud computing monitoring information panel using Web 3.0 technologies.

У мовах розвитку інформаційного суспільства використання хмарних обчислення та хмарних сервісів - це великий крок у використанні сучасних інформаційних технологій, який набуває все більшого свого розвитку. Загалом, хмарні обчислення, це тип обчислень, в основі якого закладено обмін даними та ресурсами, керування даними та виконання обчислень за допомогою спільних ресурсів. В даний час різні компанії в різних галузях енергетики, машинобудування тощо, використовують хмарні обчислення для задоволення потреб, в підтримку власної обчислювальної інфраструктури, а часто навіть і на її замін у. При використанні ресурсів, та даних, доступних в хмарі, компаніям доводиться постійно контролювати та коригувати їх для визначення навантаження на кожен тип визначеного ресурсу.

Для забезпечення моніторингу таких ресурсів найчастіше використовують так звані веб-інформаційні панелі, які дозволяють оперувати даними та ресурсами на сучасних платформах хмарних обчислень. Інформація про дані та ресурси, що надається такою інформаційною панеллю, допомагає в організації та моніторингу хмарних обчислень, що є завданням даного дослідження.

Основне завдання розробки інформаційної панелі – надання користувальницького інтерфейсу, з допомогою якого можна впорядкувати та надати інформацію таким чином, щоб її вона була наглядною та зрозумілою для інтерпретації та аналізу даних. Вона повинна допомагати адміністратору системи, або певному користувачеві взаємодіяти з необхідною інформацією без необхідності переглядати різні веб-ресурси тощо. Інформаційна панель також надає можливість понизити складність та витрати при роботі з великими обсягами інформації з допомогою складових, які надають інформацію як про важливі події, як то оперативна обробка даних і видача результатів та загальної інформації – історія обчислень, статистика тощо.

При проектуванні таких систем розглянемо ряд технологій та методів вирішення поставленого завдання моніторингу хмарних сервісів.

Так, у роботі [1] розглянуто фреймворк для створення інформаційної панелі, суть якого полягає у захоплення всіх елементів комплексного фреймворку управління хмарними рішеннями. Розроблена інформаційна панель надає можливість користувачам керувати віртуальними машинами кожна окремо, або в групами. Вона володіє такими функціями, як можливість користувача зробити знімки однієї віртуальної машини або декількох віртуальних машин та перезавантажувати віртуальні машини з допомогою інструментів інформаційної панелі. Крім того, фреймворк інформаційної панелі також використовується для управління та перевірки рішень.

Інформаційна панель розміщувалась в хмарному сервісі IBM Smart Cloud. Вона забезпечувала перевірку дзвінків, що направлялися від одного абонента до іншого, а потім перевіряла такі виклики на основі знімків зробленими інформаційною панеллю.

Наведений приклад інформаційної панелі підтверджує, що поставлена нами задача відповідає дійсності. Окрім того, ефективність наведеної інформаційної панелі вимірювалася за допомогою відеотрансляцій. На відміну від інформаційної панелі Smart Applications on Virtual Infrastructure [2], в нашому проекті відстежується різні обмеження для наданих ресурсів та надається перегляду на різних ресурсах.

У роботі [3] авторами запропоновано розробку інформаційної панелі, яка динамічно отримує доступ до стійкості сервісів хмарних обчислень. Дана інформаційна панель хмарних ресурсів використовується для доступу до загальної роботи сервісів, що розміщені у хмарі. Модель оцінки даної інформаційної панелі базується на чотирьох факторах стійкості: технічних, економічних, екологічних та соціальних факторах. Основні переваги інформаційної панелі полягають у можливості користувачам вибрати власні фактори стійкості для моніторингу. У роботі представлені лише моделі оцінки на основі перерахованих факторів та наведено загальну архітектуру додатку, однак, не наводиться поглиблена інформація про саму реалізацію інформаційної панелі.

ТЕХНІЧНА ВЕРСІЯ

Наведені рішення використовуються для певного виду хмарних сервісів, а інформаційна панель OpenStack не забезпечує моніторинг багатьох сервісів для певного ресурсу та не може бути налаштованим для моніторингу інформації, яка стосується певного ресурсу.

Наша інформаційна панель спроектована з використанням технологій Веб 3.0 [8] з допомогою фреймворку ASP.NET з використанням мови Pyton. Використання ASP.NET обумовлено легкістю масштабуння та реінжинірингу додатку, що досить суттєво відіграє важливу роль у хмарних сервісах та технологіях.

Для розробки методологічного підходу, як основний інструмент нами було обрано один з інструментів OpenStack, а саме інструмент Ceilometer. , який користувачам послуги телеметрії – технологія передачі і прийому вимірюваних даних з метою віддаленого моніторингу ресурсів хмарних обчислень для збору та аналізу даних з різних потоків.

Загальна архітектура модулів спроектованої інформаційної панелі, зокрема потоків даних та їх обробки і моніторингу зображено на рисунку 1.

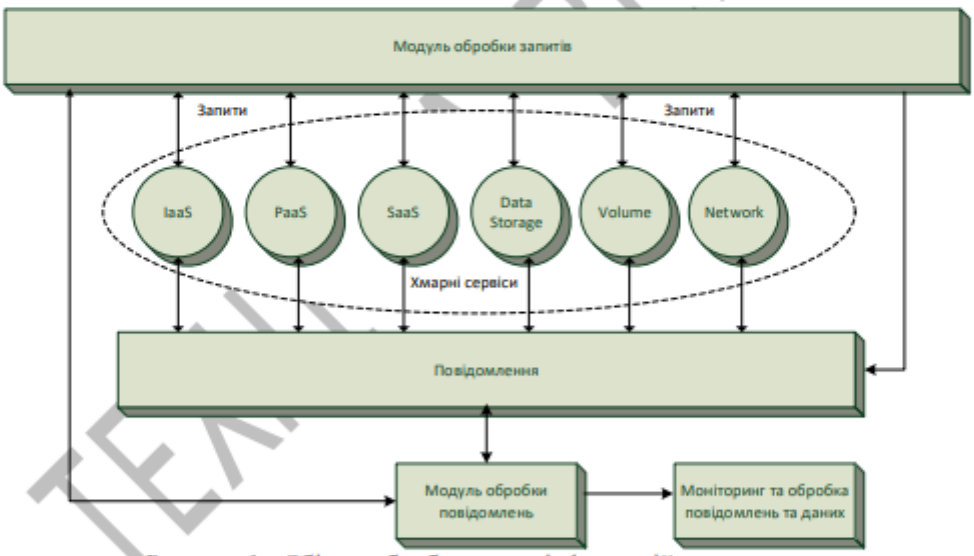


Рисунок 1 – Збір та обробка даних інформаційною панеллю

Обмін запитами модуля обробки даних з хмарними сервісами здійснюється за допомогою API для кожного окремого хмарного сервісу.

За збір даних відповідають модулі обробки запитів та модуль обробки повідомлень, які збирають дані з різних хмарних сервісів. Як видно з рисунку 1 дані

збираються двома методами, а саме з допомогою модуля запитів і модуля повідомлень здійснюється збір даних.

У модулі обробки запитів метод опитування проводиться через певний проміжок часу і дані збираються на кожному цьому проміжку часу. Опитування проводиться за допомогою розробленого API та інструментів для збору даних та інформації.

Модуль повідомлень налаштований на опитування локального або віддаленого хмероного сервісу з допомогою API. Цей модуль також запитує інформацію у хмарних сервісів про послуги для отримання даних. Він працює в обчислювальному модулі, а модуль опитування здійснює обробку повідомлень для інформаційної системи. Модуль повідомлень відповідає за запити даних, які пов'язані з обчислювальними ресурсами. Для іншого виду ресурсів модуль здійснює обробку повідомлень. Частота опитування може контролюватися згідно з налаштування API.

У модулі збору повідомлень події, що генеруються з повідомлень зібраних з хмарних сервісів, збираються та петворюються в даніпридатні для локальної інформаційної ситеми. У цьому методі модуль повідомлень відстежує черги повідомлень для інформаційної системи компанії.

Модуль обробки повідомлень є головним модулем системи збору даних, який відстежує повідомлення на вміст даних, що надаються хмарними сервісами, а також забезпечує внутрішню комунікацію. Повідомлення захоплюються модулем повідомлень, а потім перерозподіляються на основі структуризації даних кінцевим споживачам у вигляді придатним для обробки інформаційною системою. Певні повідомлення також у модулі можна конвертувати в певні події, які фільтруються за типами.

До модуля обробки даних надходять дані, які зібрані іншими модулями, де він здійснює їх обробку та публікує ці дані за допомогою декількох конвеєрів. На модуль повідомлення покладена відповідальність за обробку самих даних. Доступ до даних здійснюється через сервіси API розроблені для кожного окремого хмарного сервісу.

Модуль моніторинг і обробки повідомлень і даних здійснює публікацію даних за допомогою двох методів: повідомлювача та видавця, на основі повідомлень, які може опублікувати дані в стеку повідомлень

Отримані дані, які зібрані інформаційною панеллю, можуть зберігатися в базі даних, файлі, базі даних інформаційної системи компанії тощо. Дані збираються інформаційною панеллю, перевіряються та зберігаються потім в базу даних.

Загалом нами розглянуто різні підходи до розробки інформаційної панелі для моніторингу хмарних ресурсів. Зокрема, розглянуто різні програмні засоби моніторингу адміністрування хмарних сервісів і пов'язані з ним роботи з різними інформаційними панелями для віртуальних машин в середовищі хмарних обчислень. Здійснено аналіз методів та програмних засобів, що використовуються

ТЕХНІЧНА ВЕРСІЯ

при розробці інформаційної панелі, та запропоновано внутрішню архітектуру для інформаційної панелі моніторингу хмарних сервісів.

Перелік посилань

1. P.Deshpande, S. Sharma, A. Acharya, K.Beaty, A. Kundu. Use-case centric dashboard for cloud solutions. In 2014 IEEE International Conference on Services Computing (SCC), pp. 840–841. IEEE, 2014.
2. Kai Lei, Yining Ma, Zhi Tan. Performance comparison and evaluation of web development technologies in php, python, and node.js. Computational Science and Engineering, 2014 IEEE 17th International Conference, pp. 661–668, Dec 2014.
3. T. Rosado, J. Bernardino. An overview of openstack architecture. In Proceedings of the 18th International Database Engineering & Applications Symposium, pp. 366–367. ACM, 2014.
4. L. Ferreira, G.Putnik, M. Cunha, Z. Putnik, H. Castro, C. Alves, Vaibhav Shah, and Maria Leonilde R Varela. Cloudlet architecture for dashboard in cloud and ubiquitous manufacturing. Procedia CIRP, pp. 366–371, 2013.
5. Емі Вебб. Велика дев'ятка. Як IT-гіганти та їхні розумні машини можуть змінити людство. Харків:Vivat. – 2020. – 352 с. ISBN : 9789669822185

ТЕХНІЧНА ВЕРСІЯ

ДОДАТОК В

(обов’язковий)

ПРЕЗЕНТАЦІЙНІ МАТЕРІАЛИ

ДИПЛОМНИЙ ПРОЕКТ

1

Метод та програмні засоби моніторингу адміністрування хмарних сервісів

Виконав: студент Максимів О.В.

Керівник: канд. тех. наук, доцент Форкун Ю.В.

2

Вступ

У мовах розвитку інформаційного суспільства використання хмарних обчислення та хмарних сервісів - це великий крок у використанні сучасних інформаційних технологій, який набуває все більшого свого розвитку. Загалом, хмарні обчислення, це тип обчислень, в основі якого закладено обмін даними та ресурсами, керування даними та виконання обчислень за допомогою спільних ресурсів.

В даний час різні компанії в різних галузях енергетики, машинобудування тощо, використовують хмарні обчислення для задоволення потреб, в підтримку власної обчислювальної інфраструктури, а часто навіть і на її замін у. При використанні ресурсів, та даних, доступних в хмарі, компаніям доводиться постійно контролювати та коригувати їх для визначення навантаження на кожен тип визначеного ресурсу.

Для забезпечення моніторингу таких ресурсів найчастіше використовують так звані веб-інформаційні панелі, які дозволяють оперувати даними та ресурсами на сучасних платформах хмарних обчислень. Інформація про дані та ресурси, що надається такою інформаційною панеллю, допомагає в організації та моніторингу хмарних обчислень, що є завданням даного дослідження.

3

Мета, об'єкт та предмет дослідження

Головною метою проекту є розробка методу та програмного засобу моніторингу адміністрування хмарних сервісів

Об'єктом дослідження роботи виступають хмарні ремурси.

Предметом дослідження виступають методи та програмні засоби моніторингу адміністрування хмарних сервісів

Практичне значення

Практичне значення нашої роботи полягає у застосуванні покращеного розробленого методу адміністрування хмарних технологій при розробці інформаційних панелей такого роду, здійснювати їх реалізацію таким чином, щоб інформаційна панель дозволяла їх користувачам відстежувати, аналізувати та контролювати різні типи ресурсів, зокрема такі параметри як навантаження процесору, використовувану оперативну пам'ять, вільне місце на диску тощо.

Апробація результатів

Максимів О.В., Форкун Ю. В. АДМІНІСТРУВАННЯ ХМАРНИХ СЕРВІСІВ //Збірник наукових праць за матеріалами XIV Всеукраїнської науково-практичної конференції «Актуальні проблеми комп'ютерних наук АПКН-2022». – Хмельницький . – 2022. – С. 181-185.

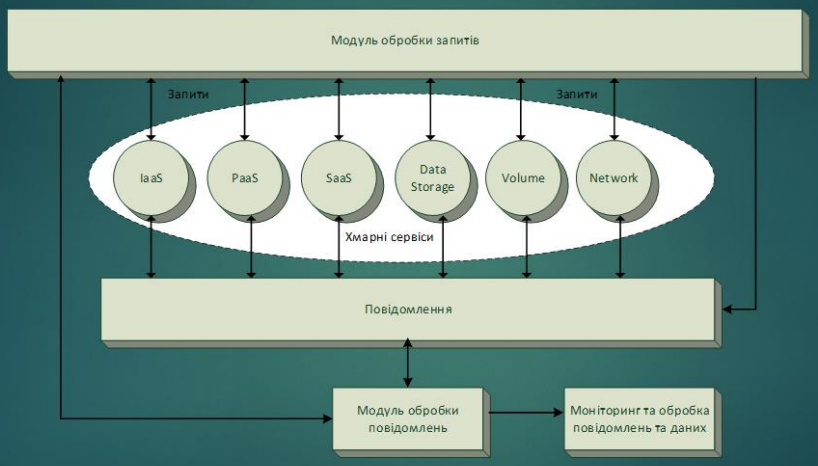
4

Основне завдання розробки інформаційної панелі – надання користувальницького інтерфейсу, з допомогою якого можна впорядкувати та надати інформацію таким чином, щоб її вона була наглядною та зрозумілою для інтерпретації та аналізу даних. Вона повинна допомагати адміністратору системи, або певному користувачеві взаємодіяти з необхідною інформацією без необхідності переглядати різні веб-ресурси тощо. Інформаційна панель також надає можливість понизити складність та витрати при роботі з великими обсягами

Інформаційна панель також надає можливість понизити складність та витрати при роботі з великими обсягами інформації з допомогою складових, які надають інформацію як про важливі події, як то оперативна обробка даних і видача результатів та загальної інформації – історія обчислень, статистика тощо.

При проектуванні таких систем розглянуто ряд технологій та методів вирішення поставленого завдання моніторингу хмарних сервісів.

Загальна архітектура модулів спроектованої інформаційної панелі



У модулі обробки запитів метод опитування проводиться через певний проміжок часу і дані збираються на кожному цьому проміжку часу. Опитування проводиться за допомогою розробленого API та інструментів для збору даних та інформації.

Модуль повідомлень налаштований на опитування локального або віддаленого хмероного сервісу з допомогою API. Цей модуль також запитує інформацію у хмарних сервісів про послуги для отримання даних. Він працює в обчислювальному модулі, а модуль опитування здійснює обробку повідомлень для інформаційної системи. Модуль повідомлень відповідає за запити даних, які пов'язані з обчислювальними ресурсами. Для іншого виду ресурсів модуль здійснює обробку повідомлень. Частота опитування може контролюватися згідно з налаштування API.

У модулі збору повідомлень події, що генеруються з повідомлень зібраних з хмарних сервісів, збираються та петворюються в даніпридатні для локальної інформаційної ситеми. У цьому методі модуль повідомлень відстежує черги повідомлень для інформаційної системи компаніїї.

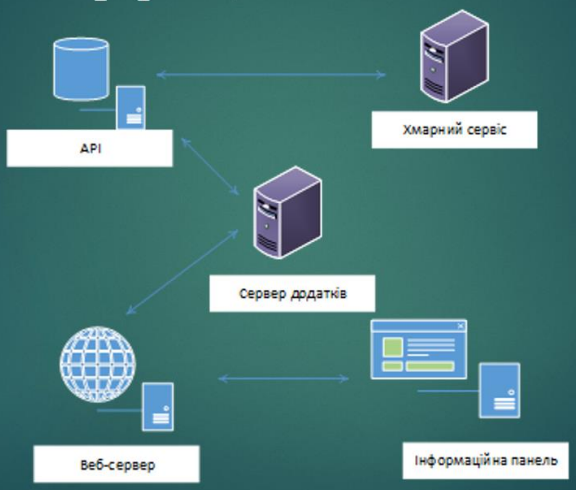
Модуль обробки повідомлень є головним модулем системи збору даних, який відстежує повідомлення на вміст даних, що надаються хмарними сервісами, а також забезпечує внутрішню комунікацію. Повідомлення захоплюються модулем повідомлень, а потім перерозподіляються на основі структуризації даних кінцевим споживачам у вигляді придатним для обробки інформаційною системою. Певні повідомлення також у модулі можна конвертувати в певні події, які фільтруються за типами.

До модуля обробки даних надходять дані, які зібрані іншими модулями, де він здійснює їх обробку та публікує ці дані за допомогою декількох конвеєрів. На модуль повідомлення покладена відповідальність за обробку самих даних. Доступ до даних здійснюється через сервіси API розроблені для кожного окремого хмарного сервісу.

Модуль моніторинг і обробки повідомлень і даних здійснює публікацію даних за допомогою двох методів: повідомлювача та видавця, на основі повідомлень, які може опублікувати дані в стеку повідомлень

Отримані дані, які зібрані інформаційною панеллю, можуть зберігатися в базі даних, файлі, базі даних інформаційної системи компанії тощо. Дані збираються інформаційною панеллю, перевіряються та зберігаються потім в базу даних.

Високорівнева архітектура інформаційної веб-панелі



Наукові публікації

Максимів О.В., Форкун Ю.В.
МЕТОДИ ТА ПРОГРАМНІ ЗАСОБИ МОНІТОРИНГУ АДМІНІСТРУВАННЯ ХМАРНИХ СЕРВІСІВ
XIV Всеукраїнська науково-практична конференція "Актуальні проблеми комп'ютерних наук (АПКН – 2022)", м.Хмельницький, ХНУ, 21-22 жовтня 2022.

Завідувачу кафедри інженерії програмного забезпечення проф. Бедратюку Л. П.

здобувача вищої освіти
Максимова О.В.
Прізвище, ініціали

факультет ІТ, 2 курс, група ІПЗм-21-1

ЗАЯВА

З правилами чинного Положення «Про дотримання академічної доброчесності в Хмельницькому національному університеті» від 26.09.2020 (зі змінами від 26.11.2020), згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлений. Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщений та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Unicheck та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

02.12.2022
дата

підпис

Mon Dec 05 11:14:06 EET 2022, Хіврич Володимир Русланович, Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

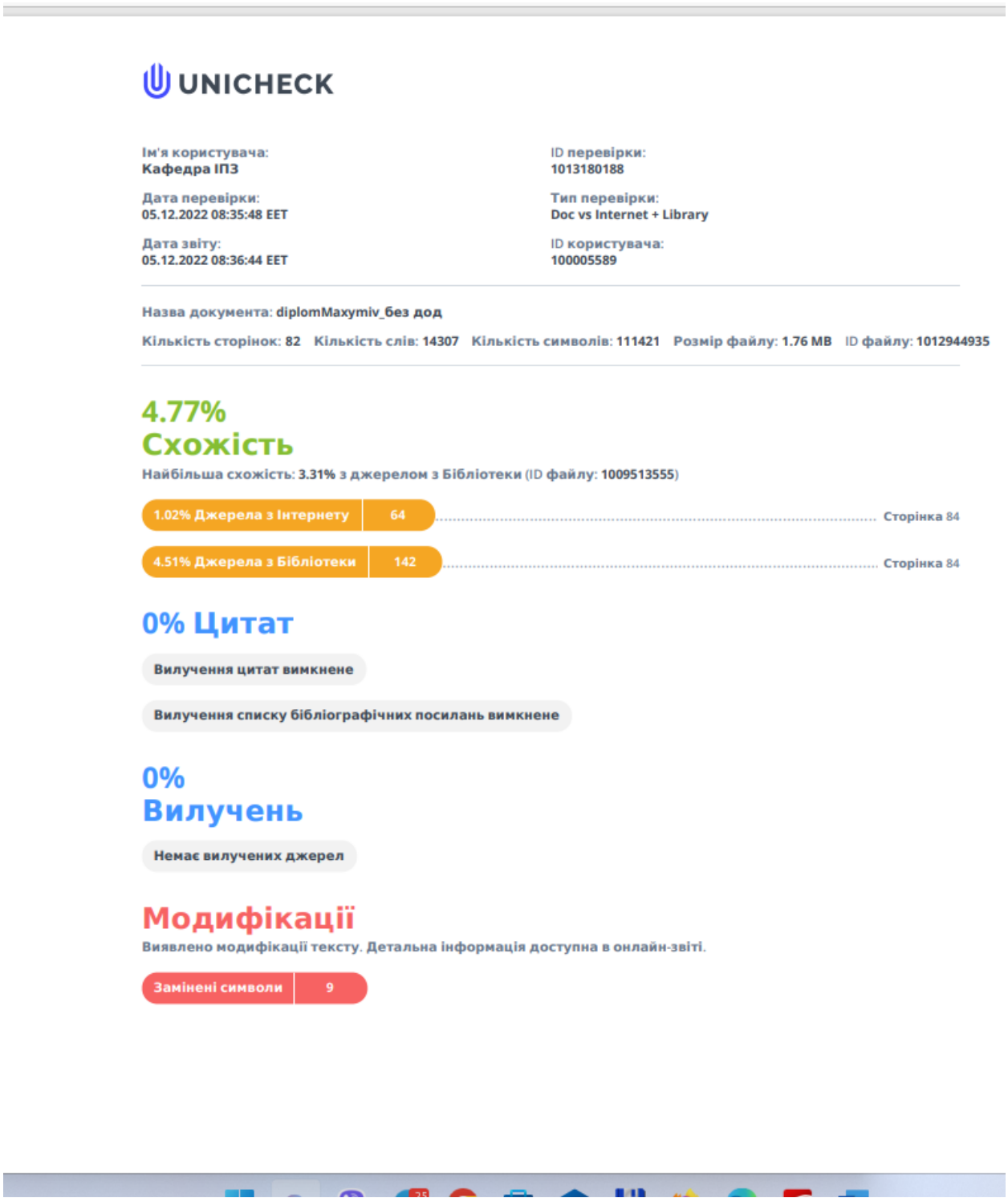
Максимальне співпадіння з одним документом 4.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 10%**

ID: 108935 Назва: КРМ на тему: «Метод та програмні засоби моніторингу адміністрування хмарних сервісів» Додано в БД: 2022-12-05 Автора: Максимів О.В. Керівники: Форкун Ю.В. Консультанти: Опоненти:	Документ		Сумарний збіг по Базі Даних	
	Символи	Лексеми	Символи	Лексеми
	101218	722	7841 (8%)	84 (12%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми



UNICHECK

Ім'я користувача:
Кафедра ІПЗ

ID перевірки:
1013180188

Дата перевірки:
05.12.2022 08:35:48 EET

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
05.12.2022 08:36:44 EET

ID користувача:
100005589

Назва документа: **diplomMaxymiv_без дод**

Кількість сторінок: **82** Кількість слів: **14307** Кількість символів: **111421** Розмір файлу: **1.76 MB** ID файлу: **1012944935**

4.77%
Схожість

Найбільша схожість: **3.31%** з джерелом з Бібліотеки (ID файлу: **1009513555**)

1.02% Джерела з Інтернету | 64 ..... Сторінка 84

4.51% Джерела з Бібліотеки | 142 ..... Сторінка 84

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи | 9

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА ДИПЛОМНИЙ ПРОЕКТ
освітнього ступеня «Магістр»

Дипломник Максимів Олександр Володимирович

Тема Метод та програмні засоби моніторингу адміністрування хмарних сервісів

Спеціальність 121 – Інженерія програмного забезпечення

Обсяг дипломного проекту:

Кількість листів креслень 19 : кількість сторінок записки 85

1.Короткий зміст пояснювальної записки та прийнятих рішень У дипломному проекті проведено аналіз предметної області та інформаційних потоків, розроблено метод адміністрування хмарних сервсів. Для розробки програмного продукту використано засоби операційних систем, мова програмування серверних систем, веб-сервер та СКБД MySQL. За допомогою цих засобів розроблено програмний комплекс адміністрування хмарних сервсів організації.

2. Висновок про відповідність проекту поставленому завданню Дипломний проект виконаний відповідно до поставленого завдання та з дотриманням всіх вимог.

3. Характеристика виконання кожного розділу проекту, ступінь використання останніх досягнень науки і техніки та передових методів роботи У вступі доведено актуальність теми, визначено мету та завдання дипломного проектування. У першому розділі здійснено аналіз предметної області. Визначено основні задачі, загальні функції та методологічні засади. Проведено аналіз останніх публікацій та наявного програмного забезпечення, з подальшим визначенням задач необхідних для реалізації проекту. Здійснено розробку методу адмінітрування та визначено функціональне призначення та вимоги до програмного комплексу. Наведено розроблену архітектуру системи та спроектовано базу даних. Виконано практичну розробку програмних модулів з описом їх особливостей, в результаті чого створено програмний комплекс. Далі було виконано тестування веб-системи та проведено його у відповідності до функціональних вимог, в результаті було підтверджено коректну роботу програмного комплексу.

4. Позитивні сторони проекту Тематика дипломного проекту є актуальною, оскільки на робота з хмарними технологіями є досить затребуваною.

5. Негативні сторони проекту У проекті опис готових рішень не є повним, доцільно додати розширений аналіз прийнятих рішень.

6. Оцінка графічного оформлення та пояснювальної записки проекту Графічне оформлення виконано відповідно до теми дипломної роботи та подано у вигляді діаграм і рисунків. Пояснювальна записка оформлена згідно вимог чинних стандартів

7. Відгук про дипломний проект в цілому Дипломна робота заслуговує позитивної оцінки. Матеріал пояснювальної записки структурований, послідовний, чіткий та простий, що дозволяє чітко зрозуміти викладений матеріал у рамках тематики дипломної роботи. Графічний матеріал дає можливість наочно побачити деталі проектування програмного засобу.

8. Інші зауваження

9. Оцінка дипломного проекту Дипломна робота виконана на достатньому рівні, відповідає поставленій задачі та заслуговує на оцінку «добре».

РЕЦЕНЗЕНТ

Завідувач кафедри комп'ютерної інженерії та інформаційних систем Доктор технічних наук, професор Товстоконь Тетяна Олександрівна

"03" грудня 2022 р. _____
(підпис)

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ
КАФЕДРИ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ
ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: «Метод та програмні засоби моніторингу адміністрування хмарних сервісів»
Автор: Максимів Олександр Володимирович
Спеціальність: 121 – Інженерія програмного забезпечення
Освітня програма: Освітньо-професійна програма «Інженерія програмного забезпечення»
Науковий керівник: Форкун Юрій Вікторович, кандидат технічних наук, доцент
Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	**відповідає**
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи. Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
5	Інше:	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) у тексті дипломної роботи системами перевірки на плагіат виявлено схожість з деякими документами в частині загальновживаних обов'язкових словосполучень у стандартних бланках (титулка, бланк завдання), у структурі змісту, назвах розділів/підрозділів тощо та в назвах публікацій у переліку джерел посилання;

2) в якості запозичень системою було зафіксовано деякі послідовності вихідного коду і посиланя на бібліотеки, які є стандартними мовними конструкціями програмування та не можуть розглядатися як об'єкт авторських прав і, відповідно, їх порушення;

3) усі запозичення є фрагментарними або мають належним чином оформленні посилання;

4) виявлені модифікації тексту не впливають на відсоток схожості.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів ідентичності/ схожості, складає 8% і адресується до 1 джерела, а саме звіту з переддипломної практики самого студента, що, з урахуванням характеру звіту з переддипломної практики, який включає в себе частину дипломного проекту відповідає характеру теми і свідчить на користь дипломної роботи.

Керівник _____ Ю.В. Форкун

Гарант ОП _____ О.М. Яшина

Завідувач кафедри _____ Л.П. Бедратюк