UNICHECK
by Turnitin

User name:
Сігайов Андрій Олександрович

Check ID:
1016046933

Check date:
06.01.2024 13:57:15 EET

Check type:
Doc vs Internet + Library

Report date:
06.01.2024 14:04:51 EET

User ID:
100004595

File name: **PohrebenkoVO_TV22mp_magistr_2023**

Page count: **99** Word count: **12966** Character count: **106978** File size: **7.01 MB** File ID: **1015745798**

Text modifications detected (similarity score might be affected)

5.29% Matches

Highest match: **2.18%** with Library source (File ID: **1015745799**)

3.74% Internet sources	207	Page 101
5.01% Library sources	217	Page 106

3.71% Quotes

Quotes	5	Page 107
References	1	Page 107

0.41% Exclusions

Some exclusions were automatic (exclusion filters: matched word count less than **10 words** and **0%**)

0.14% Internet exclusions	77	Page 108
0.35% Library exclusions	156	Page 109

Modifind

Text modifications detected. Find more details in the online report.

Replaced characters	14
Suspicious formatting	26 Pages

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут атомної та теплової енергетики

Кафедра інженерії програмного забезпечення в енергетиці

«На правах рукопису»
УДК____004.4_____

ДО ЗАХИСТУ ДОПУЩЕНО
В.о. завідувача кафедри
_____Олександр КОВАЛЬ

"……"_....._202_р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою «Інженерія програмного забезпечення інтелектуальних кібер-фізичних систем в енергетиці»
зі спеціальності 121 Інженерія програмного забезпечення

на тему: «Система вибору вибіркових дисциплін в рамках кафедри та інтеграція її у університетську систему. Back End розробка бізнес-логіки додатку, підходи до розширення функціоналу існуючої системи»

Виконав: студент 2 курсу, групи ТВ-22мп
Погребенко Василь Олександрович _____ _____
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник: професор, д.н. Сігайов А. О._____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент:_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2024

Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»

Навчально-науковий інститут атомної та теплової енергетики
Кафедра інженерії програмного забезпечення в енергетиці
Рівень вищої освіти другий, магістерський
Спеціальність 121 Інженерія програмного забезпечення
Освітньо-професійна програма «Інженерія програмного забезпечення інтелектуальних кібер - фізичних систем в енергетиці»

ЗАТВЕРДЖУЮ
В.о.завідувача кафедри
_____ Олександр КОВАЛЬ
(підпис)
«____» _____ 202_р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ

_____Погребенку Василю Олександровичу_____
(прізвище, ім'я, по батькові)

1. Тема дисертації: «Система вибору вибіркових дисциплін в рамках кафедри та інтеграція її у університетську систему. Back End розробка бізнес-логіки додатку, підходи до розширення функціоналу існуючої системи»_____
_____
науковий керівник дисертації Сігайов Андрій Олександрович, професор, д.н. _
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від "06" листопада 2023 року № 5152-с____
2. Строк подання студентом дисертації 10 січня 2024 року_____
3. Об'єкт дослідження комп'ютерні інформаційні системи для вибору вибіркових дисциплін _
4. Предмет дослідження (Вихідні дані – для магістерської дисертації за ОПП) підходи до розширення функціоналу та забезпечення якості роботи системи вибору вибіркових дисциплін
_
5. Перелік питань, які потрібно розробити проаналізувати функціональну структуру сучасних аналогічних систем, проаналізувати методи для розширення функціоналу та забезпечення якості роботи сервісу, спроектувати та розробити програмне забезпечення, що буде забезпечувати сервіс такими параметрами як: доступність, безпека, швидкість, надійність _
6. Орієнтований перелік ілюстративного матеріалу діаграми результатів опитування, емблеми та інтерфейси аналогічних систем, емблеми, схеми та діаграми роботи системних архітектур та технологій, елементи інтерфейсу

розробленої системи

7. Орієнтований перелік публікацій _____
8. Дата видачі завдання «01» листопада 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Строки виконання етапів магістерської дисертації	Примітка
1	Отримання завдання	01.11.2022	виконано
2	Дослідження предметної області	01.11.2022 - 01.12.2022	виконано
3	Постановка вимог до проєктування системи	01.12.2022 - 15.12.2022	виконано
4	Дослідження існуючих рішень	15.12.2022 - 03.01.2023	виконано
5	Розробка програмного продукту	03.01.2023 - 20.09.2023	виконано
6	Тестування	20.09.2023 - 15.10.2023	виконано
7	Захист програмного продукту	16.10.2023 - 20.10.2023	виконано
8	Підготовка магістерської дисертації	21.10.2023 - 17.11.2023	виконано
9	Передзахист	18.11.2023 - 22.12.2023	виконано
10	Захист	15.01.2024 - 19.01.2024	виконано

Студент _____ Погребенко В. О.
(підпис) (прізвище та ініціали)

Науковий керівник _____ __Сігайов А. О.__
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Магістерська дисертація за темою «Система вибору вибіркових дисциплін в рамках кафедри та інтеграція її у університетську систему. Back End розробка бізнес-логіки додатку, підходи до розширення функціоналу існуючої системи» виконана студентом кафедри інженерії програмного забезпечення в енергетиці НН ІАТЕ Погребенком Василем Олександровичем зі спеціальності 121 Інженерія програмного забезпечення за освітньо-професійною програмою «Інженерія програмного забезпечення інтелектуальних кібер-фізичних систем і веб-технологій».

Структура і обсяг магістерської дисертації. Магістерська дисертація складається зі вступу, п'яти розділів, висновків та двох додатків. Робота містить посилання на 28 джерел та 76 рисунків. Основна частина роботи викладена на 90 сторінках.

Актуальність теми роботи зумовлена необхідністю підвищення вмотивованості студентів шляхом надання можливості вибору вибіркових дисциплін та відсутністю рішень, що пропонують надання потрібних послуг у належній якості.

Мета і задачі дослідження. Метою дослідження є розробка методів покращення та розширення системи вибору вибіркових дисциплін, а саме забезпечення системі таких якостей як: безпека, доступність, швидкість роботи, тощо.

Для досягнення поставленої мети були сформульовані наступні завдання дослідження, що визначили структуру дослідження:

- проаналізувати функціональну структуру сучасних аналогічних систем;
- проаналізувати методи для розширення функціоналу та забезпечення якості роботи сервісу;
- спроектувати та розробити програмне забезпечення, що буде забезпечувати сервіс такими параметрами як: доступність, безпека, швидкість, надійність.

Об'єкт дослідження. Комп'ютерні інформаційні системи для вибору вибіркових дисциплін.

Предмет дослідження. Підходи до розширення функціоналу та забезпечення якості роботи системи вибору вибіркових дисциплін.

Методи дослідження. У роботі було використано такі методи дослідження як порівняння та аналіз аналогів, моделювання системи, тестування та верифікація. Під час виконання завдання використовувались сучасні технології хмарних обчислень та контейнеризації, включно з GKE, Docker, Kubernetes, та Helm. Для досягнення мети використовувались паттерни sidecar, webhook, operator разом з технологіями налаштування topology та affinity. Для самого додатку використовувались мови програмування Golang, Python, фреймворк Angular, та YAML deployment файли. У якості сховища данних була використана БД PostgreSQL, а для аналізу статистичних даних було використано вбудовані методи GKE.

Практичне значення одержаних результатів полягає у створенні гнучкої, надійної та масштабованої системи, яка забезпечує послуги вибору вибіркових дисциплін у належній якості, що сприяє підвищенню вмотивованості студентів та якості освітнього процесу.

Ключові слова: ВИБІРКОВІ ДИСЦИПЛІНИ, KUBERNETES, DOCKER, WEBHOOK, АДАПТИВНІСТЬ, МАСШТАБОВАНІСТЬ.

ABSTRACT

The master's thesis titled «System for selecting elective subjects within the department and integrating it into the university system. Back End development of application business logic, approaches to expand the functionality of the existing system» was completed by Vasyl Pohrebenko, a student from the Department of Software Engineering at the National Institute of Energy of IATE from the specialty in 121 Software Engineering under the educational and professional program "Software Engineering of Intelligent Cyber-Physical Systems and Web Technologies".

The structure and volume of the thesis. The master's thesis includes an introduction, five chapters, conclusions, and two appendices. The work cites 28 sources and includes 76 figures. The main body of the work spans 90 pages.

The relevance of the work's topic is due to the need to increase student motivation by providing options for selecting elective subjects and the absence of solutions offering these needed services in proper quality.

Research goal and objectives. The aim of the research is to develop methods for improving and expanding the elective subject selection system, specifically ensuring the system's security, accessibility, speed of operation, etc.

To achieve this goal, the following research tasks were formulated, defining the study's structure:

- analyze the functional structure of modern analogous systems;
- investigate methods for expanding functionality and ensuring service quality;
- design and develop software that will provide the service with parameters such as accessibility, security, speed, and reliability.

Object of research. Computer information systems for selecting elective subjects.

Subject of research. Approaches to expanding the functionality and ensuring the quality of the elective subject selection system.

Research methods. The work employed research methods such as comparison and analysis of analogues, system modeling, testing, and verification. During the task execution, modern technologies of cloud computing and containerization were used, including GKE, Docker, Kubernetes, and Helm. To achieve the aim, sidecar, webhook, and operator patterns were used along with topology and affinity configuration technologies. The application itself utilized programming languages Golang and Python, Angular framework, and YAML deployment files. PostgreSQL database was used for data storage, and built-in methods of GKE were employed for statistical data analysis.

The practical value of the results lies in creating a flexible, reliable, and scalable system that provides quality elective subject selection services, thereby increasing student motivation and the quality of the educational process.

Keywords: ELECTIVE SUBJECTS, KUBERNETES, DOCKER, WEBHOOK, ADAPTABILITY, SCALABILITY.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

API – Application Programming Interface (Інтерфейс Програмування Додатків)

AWS – Amazon Web Services (Веб-Сервіси Amazon)

GCP – Google Cloud Platform (Хмарна Платформа Google)

GKE – Google Kubernetes Engine (Двигун Kubernetes від Google)

Go – Golang (Golang)

HTTP – HyperText Transfer Protocol (Протокол Передачі Гіпертексту)

IDE – Integrated Development Environment (Інтегроване Середовище Розробки)

IP – Internet Protocol (Інтернет Протокол)

JS – JavaScript (JavaScript)

PaaS – Platform as a Service (Платформа як Сервіс)

SIS – Student Information System (Система Інформації про Студентів)

SPOF – Single Point Of Failure (Одинарна Точка Відмови)

SQL – Structured Query Language (Мова Структурованих Запитів)

SSL – Secure Sockets Layer (Рівень Безпечних Сокетів)

SSL/TLS – Secure Sockets Layer/Transport Layer Security (Рівень Безпечних Сокетів/Безпека Транспортного Рівня)

SaaS – Software as a Service (Програмне Забезпечення як Сервіс)

YAML – Yet Another Markup Language (Ще Одна Мова Розмітки)

k8s – Kubernetes (Kubernetes)

ВСТУП

В умовах стандартної системи освіти студенти часто змушені вивчати предмети, які не відповідають їхнім інтересам чи професійним планам, що призводить до зниження мотивації та, як наслідок, до падіння академічних показників. Таким чином, в університетському навчальному процесі дедалі частіше виникає потреба у гнучкості та індивідуалізації навчального процесу. Це відображається в можливості студентів обирати ті чи інші вибіркові дисципліни, що відповідають їхнім інтересам і професійним планам. Однак, ефективний вибір таких дисциплін передбачає наявність інтегрованої, надійної та зручної в користуванні системи. Наявні аналоги сервісів вибіркових дисциплін, на жаль, не забезпечують належної гнучкості або якості послуг, що не дозволяє досягти бажаного позитивного впливу на навчальний процес.

З урахуванням цих потреб та викликів сучасного університетського навчання існує необхідність у створенні якісної системі вибору вибіркових дисциплін та забезпечення можливостей для подальшого її розвитку. Таким чином, ця робота націлена на розгортку, розробку методів покращення та розширення системи вибору вибіркових дисциплін, яка б мала такі ключові характеристики як безпека, доступність, швидкість роботи і т.д.

Наукова цінність роботи полягає у створенні системи, яка буде сприяти підвищенню мотивації студентів, а також у впровадженні автоматизованих методів контролю, оптимізації ресурсів та забезпеченні безпеки та найдійності додатку, що відсутні у альтернативних, вже реалізованих системах.

У процесі виконання практичної роботи було зосереджено особливу увагу на методах розгортки та оптимізації роботи додатку, які можуть бути використані не тільки у поточних системах. У результаті буде висвітленно цінність здобутих результатів, основні етапи розробки, технічні аспекти, можливості і функціональні характеристики створеної системи, та інше.

1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Актуальність теми

Актуальність роботи полягає в тому, що негнучка система навчання, де студенти не можуть обрати дисципліни, що їх насправді цікавлять, є однією з причин низької зацікавленості та вмотивованості студентів – серйозної проблеми, що сильно шкодить навчальному процесу та погіршує рівень підготовки спеціалістів.

З метою усунення цієї проблеми існуть системи для впровадження вибіркових дисциплін, наприклад, університетська система КПІ my.kpi.ua.

Проте, все ще існує значна частка студентів, що вважає, що практична складова навчання при навчанні може бути покращена. Наприклад, у 2023 році було проведено ряд анонімних опитувань серед студентів освітньо-професійної магістратури інженерії програмного забезпечення інтелектуальних кібер-фізичних систем в енергетиці. За результатами цих опитувань можна побачити, що більше 20% студентів не цілком задоволені практичною складовою при навчанні (рис. 1.1).

15

Рисунок 1.1 – Опитування, що відображає, чи задоволені студенти практичною складовою при навчанні

А більш ніж 30% студентів не повністю задоволені формами та методами навчання (рис. 1.2) [1].

Рисунок 1.2 – Опитування, що відображає, чи задоволені студенти формами та методами навчання

Це можна пов’язати з тим, що наявні системи та їх аналоги не можуть забезпечити належну якість надання послуг, і тому максимальний ефект від вибіркових дисциплін не був отриманий.

У результаті, метою дослідження є розробка методів покращення та розширення системи вибору вибіркових дисциплін, а саме забезпечення системі таких якостей як: безпека, доступність, швидкість роботи, і не тільки. Об’єктом дослідження є комп’ютерні інформаційні системи для вибору вибіркових дисциплін, а предметом – підходи до розширення функціоналу та забезпечення

якості роботи системи вибору вибіркових дисциплін, безпека, швидкість роботи, і т. д.

1.2 Аналіз існуючих систем для вирішення проблеми

Сьогодні на ринку освітніх технологій існує достатньо систем для управління інформацією про студентів та організації процесу навчання. Деякі з них мають широку популярність та використовуються учбовими закладами по всьому світу. Далі буде розглянуто кілька з них.

1.2.1 Banner by Ellucian

Banner by Ellucian є комплексною інтегрованою системою управління навчальним закладом, яка об'єднує в собі широкий спектр функціональностей, від управління реєстрацією на курси до фінансових операцій студентів (рис. 1.3).

Рисунок 1.3 – Емблема Banner by Ellucian

Ця система широко відома своєю інтегрованістю з різними аспектами університетської діяльності, пропонуючи рішення на основі SaaS для вищих навчальних закладів. Banner забезпечує університети можливістю стандартизації

процесів, підвищення ефективності роботи та покращення досвіду студентів завдяки єдиній базі даних студентської інформації (рис. 1.4).

Рисунок 1.4 – Користувацький інтерфейс Banner by Ellucian

Серед мінусів Banner by Ellucian можна виділити його комплексність та затратність, можуть бути необхідні великі зусилля для впровадження та підтримки системи. Також, певна застарілість деяких компонентів системи може бути проблемою, оскільки технологічний прогрес вимагає постійного оновлення та адаптації до сучасних стандартів. Це може призвести до додаткових витрат на оновлення та інтеграцію з іншими сучасними системами або рішеннями [2].

Таким чином, Banner by Ellucian є ефективним і широко використовуваним рішенням для управління університетськими процесами, але його складність та вартість можуть становити виклик для деяких закладів освіти.

1.2.2 SIS (Student Information Systems) by Blackbaud

SIS (Student Information Systems) by Blackbaud є інтегрованою платформою для управління всіма аспектами студентського життя від академічного планування до управління фінансами та відслідковування успішності (рис. 1.5).

Рисунок 1.5 – Емблема SIS by Blackbaud

Ця система використовує модель SaaS, що дозволяє навчальним закладам мінімізувати IT-інфраструктуру та зосередитися на наданні якісних освітніх послуг. SIS by Blackbaud надає користувацький інтерфейс, спрямований на зручність та інтуїтивне використання, а також підтримку мобільного доступу, що дозволяє студентам та викладачам залишатися в курсі навчальних питань навіть в русі (рис. 1.6).

Рисунок 1.6 – Користувацький інтерфейс SIS by Blackbaud

Мінусами SIS by Blackbaud є висока вартість ліцензування та впровадження, що може бути значним бар'єром. Окрім того, комплексність системи також може вимагати додаткових зусиль для налаштування та адаптації до специфічних потреб інституцій, що включає необхідність у кваліфікованому IT-персоналі та тривалий час на імплементацію [3].

В результаті, ця система є гарним вибором, але не без своїх нюансів, які треба враховувати.

1.2.3 Kuali

Kuali є відкритою платформою для управління навчальними та адміністративними процесами в університетах та коледжах (рис. 1.7).

Рисунок 1.7 – Емблема Kuali

Як відкрита система, Kuali надає навчальним закладам можливість модифікувати та адаптувати функціонал системи до власних унікальних потреб без жорстких обмежень. Вона сприяє співпраці між інституціями, дозволяючи їм обмінюватися кращими практиками та розвивати нові функціональні можливості. Завдяки модульній структурі, Kuali дозволяє закладам вибирати та інтегрувати лише ті компоненти, які вони вважають необхідними, забезпечуючи гнучкість у відповідності до змінюваних вимог та ресурсів (рис. 1.8).

Рисунок 1.8 – Користувацький інтерфейс Kuali

Серед проблем Kuali виділяється її технічна складність, яка може становити виклик для налаштування. Ресурсозатратність, пов'язана з необхідністю індивідуальної адаптації та розвитку системи, також може бути важливим чинником, особливо для менших закладів або тих, що обмежені в бюджеті. Крім того, ефективне використання Kuali часто передбачає активну участь у співтоваристві користувачів та співпрацю з іншими інституціями, що може вимагати додаткових зусиль та ресурсів для координації та спільної роботи [4].

Тому Kuali є гнучкою системою, що може підійти у багатьох випадках, але вона може не підходити для всіх користувачів з огляду на її складність та ресурсозатратність.

1.2.4 PeopleSoft Campus Solutions

PeopleSoft Campus Solutions є комплексною інтегрованою системою управління, розробленою Oracle для вищих навчальних закладів (рис 1.9).

Рисунок 1.9 – Емблема PeopleSoft Campus Solutions

Ця система надає широкий спектр функцій, включаючи управління зарахуванням студентів, фінансами, навчальними результатами та іншими ключовими аспектами університетського життя. Як інтегроване рішення, воно пропонує університетам можливість централізовано керувати всіма студентськими записами та адміністративними процесами. Серед переваг можна виділити її

гнучкість та масштабованість, які дозволяють університетам налаштовувати функціонал відповідно до своїх унікальних потреб та зростати разом із системою (рис 1.10).

Рисунок 1.10 – Користувацький інтерфейс PeopleSoft Campus Solutions

Недоліки PeopleSoft Campus Solutions включають її складність у налаштуванні та управлінні, високу вартість впровадження та підтримки, а також потребу в значних часових та фінансових інвестиціях для впровадження. Сильна залежність від постачальника та потреба в постійній підтримці та оновленнях також можуть стати викликом для деяких навчальних закладів [5].

В заключенні, PeopleSoft Campus Solutions залишається популярним вибором серед вищих навчальних закладів завдяки своїй широкій функціональності та гнучкості. Але, так само як і попередні системи, це не спеціалізоване рішення на вибіркові дисципліни, та має ряд схожих недоліків, на які варто звертати увагу.

1.2.5 Jenzabar

Jenzabar є інтегрованою системою управління для вищих навчальних закладів, пропонуючи широкий спектр функцій для реєстрації, фінансового управління, обліку студентів, і академічного планування (рис 1.11).

Рисунок 1.11 – Емблема Jenzabar

Це рішення вирізняється своєю гнучкістю та адаптивністю до специфічних потреб закладу, надаючи можливість інтеграції з широким спектром інших систем та технологій (рис. 1.12).

Рисунок 1.12 – Користувацький інтерфейс Jenzabar

Переваги Jenzabar включають її модульну структуру, що дозволяє навчальним закладам обирати та інтегрувати необхідні компоненти, а також її здатність до налаштування і адаптації, що дозволяє закладам ефективно реагувати на змінювані вимоги та обставини. Однак, ця модульність та гнучкість можуть призвести до складностей управління та налаштування системи, вимагаючи додаткових ресурсів для інтеграції та адаптації [6].

Недоліки Jenzabar включають потенційну складність у налаштуванні та управлінні, вимогу до ресурсів для індивідуальної адаптації та розвитку системи, а також можливу високу вартість впровадження та підтримки. Також, ефективне використання системи часто передбачає активну участь та співпрацю з іншими закладами, що може вимагати додаткових зусиль для координації.

Таким чином, Jenzabar продовжує бути вибором багатьох навчальних закладів через свою здатність до адаптації та індивідуалізації. Але має і ряд застережень щодо використання.

1.2.6 CampusNexus Student

CampusNexus Student, розроблена компанією Campus Management, є комплексною інтегрованою системою управління, яка надає широкий спектр функцій для вищих навчальних закладів (рис. 1.13).

Рисунок 1.13 – Емблема CampusNexus Student

CampusNexus Student також є універсальною системою, як і попередні системи, що було розглянуто. Вона дозволяє університетам централізовано керувати зарахуванням студентів, фінансами, навчальними результатами та іншими ключовими аспектами університетського життя. Система є cloud рішенням, має достатню гнучкий та різноманітний функціонал та може масштабуватись. Це забезпечує високу адаптивність до специфіки кожного навчального закладу та є важливим компонентом для ефективного управління університетом в сучасних умовах (рис. 1.14).

Рисунок 1.14 – Користувацький інтерфейс CampusNexus Student

Однак, CampusNexus Student має свої складнощі у налаштуванні та управлінні. Система також є не орієнтованою тільки на вибіркові дисципліни, та має високу вартість впровадження та підтримки, а також значні часові та фінансові інвестиції, необхідні для її впровадження, є серйозними викликами для багатьох навчальних закладів. Також, слід пам’ятати, що система залежить від постачальника послуг [7].

Незважаючи на ці недоліки, CampusNexus Student продовжує бути популярним вибором серед вищих навчальних закладів через свою здатність адаптуватися до різних потреб та надання широкого спектра функцій. Це не спеціалізоване рішення, але воно пропонує гнучкість та функціональність, які можуть бути налаштовані для різних типів навчальних закладів і їх специфічних потреб.

1.2.7 Університетська система «my.kpi.ua»

Сучасний університетський портал "https://my.kpi.ua/" із часом перетворився на одну з важливих систем університету, забезпечуючи студентів та викладачів важливою інформацією та функціональністю (рис. 1.15).

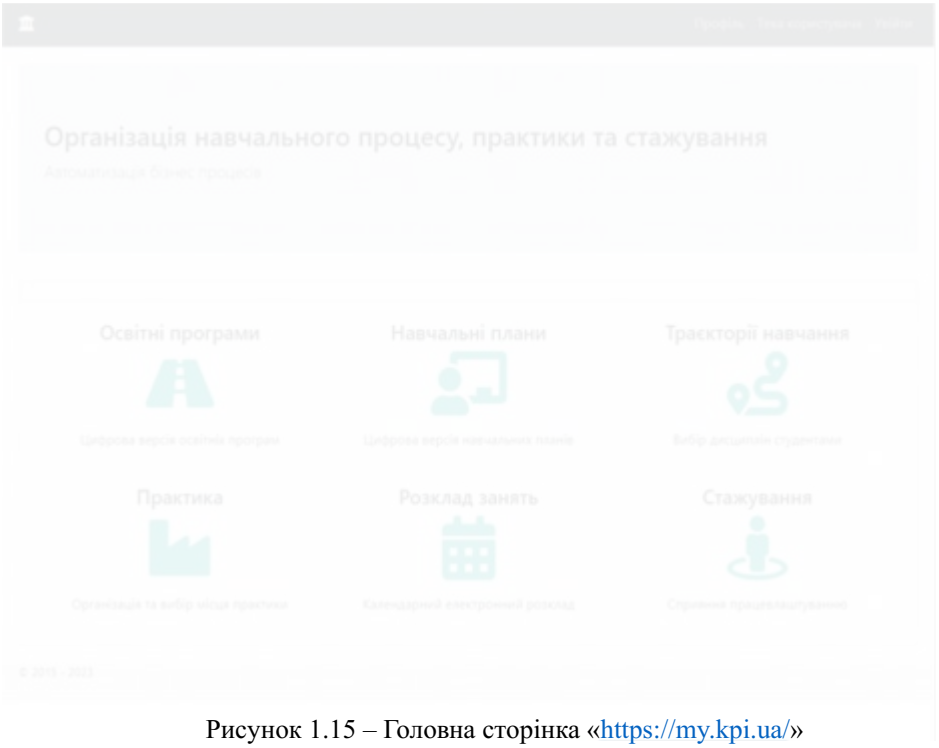

Рисунок 1.15 – Головна сторінка «https://my.kpi.ua/»

Зробимо короткий огляд системи:

- історія розвитку – в початковий період свого існування, цей ресурс не був орієнтований на вибіркові дисципліни. Але з часом, із введенням нових законодавчих ініціатив, що надали студентам право вибору вивчаємих дисциплін, ресурс почав адаптуватися під нові потреби;
- технічна характеристика – на сьогоднішній день "my.kpi.ua" є монолітною системою, написаною на мові програмування PHP.

Монолітна архітектура, хоча і спрощує процес розробки, може призводити до проблем зі скейлінгом, особливо коли ресурс зазнає великого навантаження від відвідувачів;

- проблеми стабільності – однією з основних проблем системи є її тенденція до відмов під час пікових навантажень. Це особливо відчутно під час періодів вибору дисциплін;
- хостинг – портал розміщений на виділеному сервері, наданому компанією DataGroup. Хоча таке рішення забезпечує достатню продуктивність та гнучкість для більшості задач, спеціально орієнтовані рішення як GKE краще підходять для розподілених та високонавантажених систем.

Загалом, університетська система "my.kpi.ua" є важливим інструментом для студентського навчання, але в її архітектурі та виконанні є місце для оптимізацій та покращень.

1.3 Постановка задачі

При загальному огляді аналогів було виділено наступні недоліки, які мають бути усунуті:

- висока вартість ліцензії;
- потреба в адаптації під структуру університету;
- зазвичай, головний акцент зосереджений на загальному управлінні студентами, а не на вибіркових дисциплінах;
- вимагають значних зусиль для впровадження налаштування, та підтримки.

Окрім того, унверситетьска система КПІ мала такі додаткові недоліки як проблеми зі стабільністю, підтримкою, хостингом. Як наслідок, хоч і існує більш

ніж достатньо рішень, знайти ідеальне під конкретний заклад може бути проблематично. Отже, існує необхідність у запровадженні своєї системи.

Також система вибору вибіркових дисциплін для студентів має відповідати ряду ключових вимог, а саме:

- безпека – забезпечення захисту особистих даних студентів та надійна протидія можливим зовнішнім атакам;
- доступність – система повинна бути легко доступна з будь-якого пристрою;
- швидкість роботи – забезпечення високої швидкості відгуку системи на дії користувача;
- інтегрованість – співпраця системи з іншими університетськими системами для обміну інформацією;
- гнучкість та розширюваність – система повинна легко адаптуватися до змін у вимогах та масштабуватися за потребами;
- автоматизація процесів – впровадження автоматизованих систем для оптимізації вибору дисциплін та контролю ресурсів.

Дотримання зазначених вимог допоможе забезпечити ефективну роботу системи і високий рівень задоволеності користувачів.

Таким чином, основні задачі, які потрібно розв'язати для досягнення мети, будуть включати в себе такі пункти як: аналіз методів для розгортки та розширення функціоналу та забезпечення якості роботи сервісу, проектування та розробка програмного забезпечення, що буде забезпечувати сервіс необхідними якостями.

Висновки до розділу 1

У цьому розділі було визначено актуальність роботи з використанням наявної статистики по факультету, а також здійснено аналіз сучасних систем для

вибору вибіркових дисциплін, які використовуються у вищих навчальних закладах, зокрема такі як Banner by Ellucian, SIS by Blackbaud, Kuali, система my.kpi.ua, тощо. Розглянувши кожну з систем, було виявлено, що вони пропонують різноманітні можливості для управління навчальними планами, реєстрацією на курси та іншими аспектами студентського життя, одночасно маючи свої унікальні переваги та обмеження. Відкрита платформа, співпраця, модульність, а також інтегрованість із різними аспектами університетської діяльності виступають як ключові позитивні аспекти, тоді як комплексність, висока вартість та технічні виклики є основними недоліками. Таким чином, у результаті аналізу було зроблено висновок про необхідність запровадження власної системи для вибору вибіркових дисциплін, а також була виконана постановка задачі для роботи.

2 АПАРАТ ВИРІШЕННЯ ПОСТАВЛЕНОЇ ЗАДАЧІ

2.1 Аналіз можливих архітектур розміщення додатку

2.1.1 Використання власного виділеного сервера (On-Premise)

У контексті вибору архітектурних рішень для інформаційних систем, використання власного виділеного сервера займає особливе місце, оскільки це один з найбільш старих та традиційних підходів до розміщення та управління IT-ресурсами. Такий підхід також відомий як «On-Premise», на відміну від cloud рішень (рис. 2.1).

Рисунок 2.1 – Візуалізація різниці On-Premise від Cloud

Переваги використання власного виділеного сервера наведені нижче.

1. Повний контроль: власники мають повний контроль над апаратним та програмним забезпеченням, що дозволяє точно налаштовувати систему під специфічні потреби проекту.

2. Висока налаштовуваність: можливість оптимізувати сервер під конкретні завдання, вибираючи відповідні компоненти та налаштування.
3. Незалежність від третіх сторін: відсутність залежності від зовнішніх провайдерів може знижувати ризики пов'язані з простоями та доступністю послуг.
4. Безпека: можливість імплементації індивідуальних рішень безпеки та політик конфіденційності.

Але така архітектура має ряд недоліків, з причини яких все більше компаній відмовляються від неї.

1. Висока вартість: початкові витрати на придбання та налаштування обладнання, а також тривалі витрати на його обслуговування та оновлення.
2. Технічне обслуговування: потреба в кваліфікованому персоналі для постійного обслуговування, ремонту та оновлення обладнання.
3. Масштабування: фізичні обмеження на розширення ресурсів, особливо в умовах різкого зростання навантаження.
4. Ризики пов'язані з надійністю: потенційні збої в роботі та необхідність самостійного забезпечення високої доступності та резервного копіювання [8].

Таким чином, під час вибору архітектурного рішення на основі використання власного виділеного сервера необхідно враховувати комплексний баланс між повним контролем і налаштовуваністю та з іншого боку - високими витратами та відповідальністю за обслуговування. Цей варіант може бути ідеальним для сценаріїв, де критично важливі специфічні вимоги до апаратури, безпеки та налаштувань, але вимагає значних інвестицій та ресурсів для ефективного управління та підтримки.

2.1.2 Оренда інфраструкатури (IaaS)

Оренда інфраструктури, наприклад, віртуальної машини у провайдера хмарних послуг (як це зроблено для системи my.kpi.ua) є популярним рішенням для розгортання інформаційних систем, що пропонує гнучкість та масштабованість без необхідності інвестувати у власне фізичне обладнання (рис 2.2).

Рисунок 2.2 – Візуалізація аспектів, про які дбає IaaS

Оренада інфраструктури, також відома як Infrastructure as a Service (IaaS), має ряд переваг.

1. Мінімізація капітальних витрат: немає потреби у придбанні фізичного обладнання; платіж здійснюється за фактично використані ресурси.
2. Гнучкість та швидке масштабування: можливість швидкого збільшення або зменшення ресурсів відповідно до поточних потреб.

3. Управління та обслуговування: провайдери зазвичай пропонують управління інфраструктурою, безпекою та підтримку, знімаючи це навантаження з користувача.
4. Доступність та надійність: висока доступність та резервне копіювання забезпечується на стороні провайдера, знижуючи ризики втрати даних та простою.

Але існує і ряд недоліків, на які варто звернути увагу.

1. Залежність від провайдера: потенційна залежність від конкретного провайдера хмарних послуг та їх політик, ціноутворення та доступності послуг.
2. Вартість за тривалим користуванням: хоча початкові витрати нижчі, тривале користування може бути дорожчим у порівнянні з власними серверами, особливо при стабільному високому навантаженні.
3. Непередбачуваність витрат: складність у передбаченні вартості, оскільки вона залежить від фактичного використання ресурсів, що може відрізнятися.
4. Питання безпеки та конфіденційності: хоча провайдери зазвичай пропонують сильні заходи безпеки, користувачі можуть мати побоювання щодо безпеки даних при розміщенні їх у хмарі.
5. Обмеження налаштовуваності на рівні апаратного забезпечення: у той час як віртуальні машини пропонують значну гнучкість, вони все ж таки обмежені можливостями та конфігураціями апаратного забезпечення, наданого хмарним провайдером [9].

В результаті, оренда інфраструктури є вигідним рішенням для багатьох сценаріїв, особливо коли потрібна швидка розгортка, гнучкість управління ресурсами та відсутність необхідності у значних капітальних інвестиціях. Важливо зважувати переваги та недоліки, включаючи потенційні довгострокові витрати та залежність від провайдера, при виборі віртуальної машини як платформи для розгортання інформаційних систем.

2.1.3 Використання хмарних платформ (PaaS)

Хмарні платформи як сервіс (Platform as a Service, PaaS) надають середовище та інструменти для розробки, тестування, доставки та управління програмними додатками без необхідності займатися складним управлінням інфраструктурою. PaaS забезпечує розробників широким спектром інструментів та сервісів, що дозволяють зосередитися на створенні додатків, замість управління апаратним забезпеченням, мережею, сховищами, базами даних та іншими ресурсами (рис. 2.3).

Рисунок 2.3 – Аспекти, про які треба дбати користувачу (позначені темно-синім), та аспекти, про які подбає провайдер (позначені світло-синім)

Популярні хмарні платформи включають Google Kubernetes Engine (GKE), Amazon Web Services (AWS) та Microsoft Azure. У користь використання хмарних платформ (PaaS) можна віднести ряд перваг.

1. Ефективність розробки: PaaS пропонує готові до використання середовища розробки та інструменти, що значно прискорюють процес розробки та тестування.
2. Автоматизація технічного обслуговування: хмарні платформи автоматично управляють більшістю аспектів інфраструктури, включаючи безпеку, масштабування, балансування навантаження та відновлення після збоїв.
3. Масштабованість: PaaS дозволяє ще легше та швидше масштабувати додатки, адаптуючи ресурси до змінних вимог та навантаження.
4. Інтеграція та співпраця: багато PaaS пропонують вбудовану підтримку для інтеграції з іншими сервісами та інструментами, спрощуючи співпрацю у командах.

Але слід зважати і на ряд недоліків, що можуть викликати проблеми.

1. Залежність від провайдера: використання PaaS може створити залежність від конкретного провайдера хмарних послуг, ускладнюючи міграцію або інтеграцію з іншими сервісами.
2. Обмежений контроль: хоча PaaS знімає багато технічних тягарів, це також може обмежувати контроль над інфраструктурою та глибину налаштувань.
3. Вартість: використання PaaS може бути дорожчим у порівнянні з самостійним управлінням інфраструктурою, особливо при великих обсягах використання.
4. Безпека та відповідність: необхідно враховувати, що політики безпеки та відповідності PaaS можуть варіюватися і потребують ретельного оцінювання та можливої додаткової налаштувань [10].

Як наслідок, використання хмарних платформ (PaaS) є потужним рішенням для розробки та розгортання додатків, забезпечуючи широкий спектр сервісів та інструментів для спрощення управління інфраструктурою та процесу розробки. Вибір правильної PaaS може значно підвищити продуктивність, масштабованість та ефективність розробки. Проте, важливо враховувати потенційні обмеження та залежності, що вони вносять, для вибору найбільш відповідного хмарного рішення, що відповідає конкретним потребам вашого проекту.

2.2 Технології розгортання сервісів

2.2.1 Docker як основа контейнеризації

Docker є однією з найпопулярніших платформ для контейнеризації, дозволяючи розробникам пакувати додатки та їх залежності у контейнери, які можуть бути легко транспортовані та виконані на будь-якій системі, що підтримує Docker.

Існує ряд причин, чому Docker використовується по всьому світу.

1. Багатоплатформність: контейнери Docker забезпечують консистентність через всі етапи розробки, тестування та виробництва.
2. Легкість використання: Docker контейнери можуть бути створені, запущені, зупинені, переміщені, та видалені з простими командами, знижуючи складність управління додатками.
3. Швидкість: контейнери забезпечують швидке розгортання, оскільки їм не потрібно завантажувати повноцінну ОС для роботи.
4. Ізоляція та безпека: кожен контейнер ізольований та обмежений; якщо один контейнер зазнає збою, він не впливає на інші контейнери.

Ці дуже вагомі переваги, зокрема особливості його роботи (рис. 2.4), забезпечили Docker всесвітньою популярністю серед розробників.

Рисунок 2.4 – Контейнери (праворуч) потребують менше ресурсів, бо не потребують віртуалізації операційної системи

Але також варто звертати увагу на деякі нюанси використання.

1. Управління контейнерами: при великій кількості контейнерів управління ними стає складним без додаткових інструментів оркестрації, таких як Kubernetes.
2. Безпека: незважаючи на ізоляцію, контейнери ділять ядро ОС з хост-машини, що може створювати потенційні вразливості.
3. Зберігання даних: тривале зберігання та управління даними може бути складним, оскільки дані в контейнерах зазвичай є тимчасовими та втрачаються при зупинці контейнера.
4. Продуктивність: хоча контейнери зазвичай мають менше накладних витрат, ніж віртуальні машини, існують сценарії, коли вони можуть сповільнювати продуктивність, особливо при великих обсягах вхідно-вихідних операцій [11].

Тому Docker став основою сучасної контейнеризації завдяки своїй портативності, швидкості та ефективності. Проте, в реальних виробничих умовах, Docker рідко використовується самостійно. Зазвичай він інтегрований з іншими технологіями оркестрації та управління, такими як Kubernetes, для ефективнішого розгортання, масштабування та управління контейнеризованими додатками.

2.2.2 Docker Compose для управління багатоконтейнерними додатками

Docker Compose (рис. 2.5) є інструментом для визначення та управління багатоконтейнерними Docker додатками. Використовуючи простий файл YAML для конфігурації служб, він дозволяє користувачам оркеструвати, стартувати, зупиняти та масштабувати всі контейнери та їх залежності одночасно.

Рисунок 2.5 – Емблема Docker Compose

Docker Compose широко використовується з ряду причин.

1. Спрощення конфігурації: Docker Compose дозволяє описати ціле багатоконтейнерне середовище в одному файлі, зменшуючи складність та спрощуючи процеси налаштування та розгортання.
2. Єдина точка управління: він надає єдиний інтерфейс для управління всіма контейнерами та їхніми залежностями, роблячи процеси оркестрації більш зрозумілими та управліними.
3. Розробка та тестування: завдяки можливості швидкого старту та зупинки всіх служб, Docker Compose є ідеальним для розробки та тестування, дозволяючи розробникам ефективно тестувати взаємодію між контейнерами.
4. Локальна розробка: Docker Compose часто використовується для локального налаштування та розгортання додатків, що дозволяє розробникам працювати в умовах, наближених до виробничого середовища.

Але він не є універсальним рішенням з огляну на деякі моменти.

1. Масштабування: хоча Docker Compose добре підходить для локальних та невеликих середовищ, він може не впоратися з великими та складними системами, особливо в середовищах з високим навантаженням.
2. Обмежені можливості оркестрації: для складних сценаріїв оркестрації та автоматичного масштабування може знадобитися використання більш потужних інструментів, таких як Kubernetes.
3. Залежність від версій: завдяки швидкому розвитку технологій Docker, конфігурації Docker Compose можуть вимагати оновлень для сумісності з новими версіями Docker.
4. Управління ресурсами: потреба вручну керувати ресурсами для кожного контейнера та визначати залежності між ними може бути складним та часозатратним [12].

Таким чином Docker Compose є міцним інструментом для розробників, що забезпечує легкість і зручність у визначенні, оркестрації та управлінні багатоконтейнерними додатками. Він особливо корисний у розробці, тестуванні та малих виробничих середовищах. Проте, для більш великомасштабних та складних розгортань, розробники часто вдаються до використання більш потужних інструментів оркестрації та управління, таких як Kubernetes.

2.2.3 Kubernetes як система оркестрації контейнерів

Kubernetes (K8s) є відкритим джерелом та однією з найбільш популярних систем оркестрації контейнерів, розробленим для автоматизації розгортання, масштабування та управління додатками в контейнерах.

Kubernetes використовується майже всюди у важких та складних додатках завдяки його визначним перевагам.

1. Автоматичне масштабування: Kubernetes може автоматично масштабувати кількість контейнерів залежно від навантаження без необхідності втручання з боку оператора.
2. Самовідновлення: він може автоматично перезапускати контейнери, що зазнали збоїв, замінювати та рескедулювати контейнери при виході з ладу вузлів, а також убивати контейнери, які не відповідають заданим користувачем здоров'ям.
3. Розподілення навантаження: Kubernetes може розподіляти вхідний трафік між контейнерами для підтримки оптимального навантаження та відмовостійкості.
4. Управління конфігурацією та секретами: Kubernetes дозволяє зручно керувати конфігурацією та секретами.

Але при його використанні може виникнути ряд складнощів.

1. Складність: Kubernetes є дуже потужною системою, але разом з тим і досить складною для розуміння та управління, особливо для новачків.
2. Ресурсоємність: для запуску Kubernetes необхідний відносно великий набір ресурсів, що може бути недоцільним для дуже малих проектів або тестових середовищ.
3. Високі вимоги до налаштування: Kubernetes складається з багатьох компонентів та потребує ретельного налаштування та оптимізації для досягнення оптимальної продуктивності та безпеки [13].

Тому Kubernetes є вибором багатьох компаній та розробників завдяки його гнучкості, масштабованості та потужним можливостям автоматизації. Він ідеально підходить для управління складними, великомасштабними, багатосервісними додатками та є ключовим компонентом у сучасних DevOps практиках. Проте, варто пам’ятати і про складність цієї технології (рис. 2.6).

Рисунок 2.6 – Схема основних компонентів Kubernetes

Однак, необхідно врахувати високі вимоги до технічної експертизи та ресурсів для ефективного використання Kubernetes.

2.2.4 Helm як менеджер пакетів для Kubernetes

Helm вважається пакетним менеджером Kubernetes, що спрощує процеси встановлення, конфігурації та оновлення додатків, які розгортаються на Kubernetes. Він використовує пакети, відомі як "charts", для визначення, встановлення та оновлення Kubernetes ресурсів (рис. 2.7).

Рисунок 2.7 – Схема взаємодії Helm з Kubernetes

Helm використовують разом з Kubernetes з ряду причин.

1. Стандартизація розгортання: Helm дозволяє розробникам і системним адміністраторам використовувати попередньо визначені шаблони для розгортання додатків, забезпечуючи консистентність і зменшуючи помилки.
2. Легкість управління: Helm значно спрощує управління життєвим циклом додатків, дозволяючи легко встановлювати, оновлювати та видаляти додатки в Kubernetes.
3. Гнучкість: Helm charts можуть бути легко налаштовані для відповідності конкретним потребам та середовищам, дозволяючи детальне управління конфігураціями.
4. Спільнота та підтримка: Helm має велику та активну спільноту, яка розробляє та підтримує велику кількість готових до використання charts для різних додатків і сервісів.

Проте також варто звернути увагу на ряд моментів.

1. Крива навчання: для ефективного використання Helm та написання власних charts може знадобитися час на вивчення його особливостей та синтаксису.
2. Безпека: використання Helm може ввести додаткові ризики безпеки, особливо при управлінні доступом до ресурсів та виконанні скриптів. Важливо забезпечити, що charts та Helm сам по собі належним чином конфігуруються та оновлюються.
3. Залежність від Helm сервера (Tiller): у попередніх версіях Helm використовував компонент на стороні сервера під назвою Tiller, який міг створювати додаткові проблеми з управлінням та безпекою. Проте, в останніх версіях Helm 3, Tiller було видалено, що спростило архітектуру та управління.
4. Управління залежностями: керування залежностями та версіями може бути складним, особливо коли використовується багато charts з різними залежностями [14].

Таким чином, Helm є суттєвим інструментом для спрощення розгортання та управління додатками в Kubernetes, надаючи стандартизовані та легко налаштовувані рішення для оркестрації контейнерів. Він відіграє ключову роль у забезпеченні ефективного управління ресурсами та додатками, але вимагає ретельного планування та управління з точки зору безпеки та конфігурації. Незважаючи на свої складності, інтеграція Helm з Kubernetes може значно підвищити ефективність та гнучкість розгортання сервісів.

2.3 Платформи для розгортання

2.3.1 Microsoft Azure

Microsoft Azure (рис. 2.8) є однією з провідних хмарних платформ, яка пропонує широкий спектр обчислювальних, сховищевих, мережевих та інших сервісів. Це інтегроване рішення підтримує різноманітні технології та мови програмування, даючи можливість розробникам і компаніям швидко розгортати, управляти та масштабувати додатки.

Рисунок 2.8 – Емблема Microsoft Azure

До переваг Microsoft Azure можна віднести ряд властивостей.

1. Широкий спектр послуг: Azure пропонує все від віртуальних машин, контейнерів, баз даних, аналітики до штучного інтелекту та IoT послуг, забезпечуючи рішення для майже будь-яких потреб.
2. Інтеграція з Microsoft продуктами: Azure легко інтегрується з широким спектром продуктів Microsoft, включно з Office 365, SharePoint та Dynamics 365, що робить його ідеальним рішенням для бізнесів, які вже використовують ці продукти.
3. Глобальне охоплення: Azure має одну з найширших мереж центрів обробки даних, забезпечуючи низьку затримку та високу доступність послуг по всьому світу.
4. Безпека та дотримання стандартів: Azure підкреслює безпеку та приватність даних, пропонуючи суворі політики безпеки та великий набір сертифікацій відповідності.

Але існує і ряд недоліків.

1. Складність: широкий спектр послуг та можливостей може ускладнити навігацію та використання платформи, особливо для нових користувачів.
2. Вартість: хоча Azure пропонує гнучкі та конкурентоспроможні ціни, повне розуміння та оптимізація витрат може бути складною задачею через складність тарифних планів та білінгу.

3. Інтерфейс користувача: деякі користувачі відзначають, що портал Azure може бути перевантаженим і менш інтуїтивно зрозумілим порівняно з аналогічними сервісами.
4. Технічна підтримка: хоча Azure пропонує різні плани підтримки, швидкість та якість реакції можуть варіюватись, і деякі користувачі зазначають, що для отримання швидкої допомоги потрібно вищий рівень підтримки [15].

Microsoft Azure є потужною хмарною платформою, яка підходить для широкого спектру задач, де потрібно масштабоване та інтегроване рішення для управління своїми додатками. Його охоплення, інтеграція з продуктами Microsoft, та суворі стандарти безпеки роблять його привабливим вибором, хоча потенційні користувачі повинні також враховувати його виклики зі складністю та вартістю.

2.3.2 Google Kubernetes Engine

Google Kubernetes Engine (рис 2.9), також відомий як GKE, є хмарним сервісом від Google Cloud, який забезпечує потужну та легко управляєму інфраструктуру для розгортання, управління та масштабування додатків на основі контейнерів з використанням Kubernetes.

Рисунок 2.9 – Емблема Google Kubernetes Engine

GKE може запропонувати ряд переваг.

1. Інтегроване середовище: GKE тісно інтегровано з усім екосистемою Google Cloud, забезпечуючи легкий доступ до різних сервісів та засобів Google Cloud, включаючи сховища, бази даних та сервіси аналітики.
2. Автоматизоване управління: GKE автоматизує багато задач управління Kubernetes, включаючи автоматичне масштабування, оновлення та управління життєвим циклом.
3. Безпека та надійність: GKE використовує потужну інфраструктуру Google для забезпечення високого рівня безпеки та надійності, включаючи захист від DDoS атак, шифрування даних та ізольовані мережі.
4. Глобальне охоплення: завдяки глобальній інфраструктурі Google, GKE забезпечує широкий охоплення та низьку затримку, дозволяючи розгортати додатки ближче до користувачів у будь-якій частині світу.

Але слід звернути увагу і на нюанси.

1. Вартість: хоча GKE пропонує багато автоматизованих та оптимізованих функцій, це може призвести до вищих витрат порівняно з самостійним управлінням Kubernetes, особливо при високому навантаженні або великому масштабі використання.
2. Комплексність налаштування: незважаючи на спрощення багатьох аспектів управління Kubernetes, GKE все ще може вимагати значного розуміння Kubernetes для оптимального налаштування та управління середовищами.
3. Залежність від провайдера: подібно до інших хмарних рішень, використання GKE створює певну залежність від Google Cloud, що може ускладнювати міграцію сервісів або збільшувати ризики пов'язані з локальними перебоями чи змінами у політиці Google.

4. Ресурсоємність: для оптимальної продуктивності та надійності GKE може потребувати використання високопродуктивних (і відповідно дорогих) ресурсів, особливо для великих та інтенсивних додатків [16].

GKE пропонує потужну платформу для розгортання та управління контейнеризованими додатками, забезпечуючи легке масштабування, високу доступність та тісну інтеграцію з сервісами Google Cloud. Воно ідеально підходить для організацій, які шукають високоавтоматизоване рішення для управління Kubernetes та хочуть використовувати переваги хмарної інфраструктури Google. Проте, важливо врахувати потенційні витрати, технічні складнощі та залежність від хмарного провайдера перед вибором GKE як основи для середовища Kubernetes.

2.3.3 Amazon Web Services (AWS)

Платформа Amazon Web Services (рис 2.10), також відома як AWS, є однією з провідних провайдерів хмарних обчислень на світовому ринку, пропонуючи широкий спектр обчислювальних ресурсів, сховищ, баз даних, аналітичних, машинного навчання, мережевих та інших сервісів для підтримки великих та малих бізнесів.

Рисунок 2.10 – Amazon Web Services

Переваги використання Amazon Web Services наведені нижче.

1. Широкий спектр послуг: AWS пропонує один з найширших асортиментів хмарних сервісів, що підтримують майже будь-які обчислювальні потреби, від віртуальних серверів та контейнерів до штучного інтелекту, Інтернету речей та квантових обчислень.
2. Масштабованість та гнучкість: AWS дозволяє легко масштабувати ресурси вгору або вниз згідно з потребами бізнесу, пропонуючи гнучкість управління навантаженням та оптимізацією витрат.
3. Надійність та відмовостійкість: завдяки глобальній інфраструктурі та надійним технологіям, AWS забезпечує високу доступність та відмовостійкість сервісів.
4. Безпека та відповідність стандартам: AWS приділяє велику увагу безпеці, надаючи розширені інструменти та сервіси для захисту даних, ідентифікації та доступу, а також дотриманню регулятивних та галузевих стандартів.

Але слід звертати увагу і на недоліки, які можуть призвести до проблем.

1. Складність: через велику кількість сервісів та опцій, нові користувачі можуть зіткнутися з високим порогом входу та складнощами у навігації та оптимізації сервісів.
2. Вартість: хоча AWS пропонує гнучку модель ціноутворення, повне розуміння і ефективне управління витратами вимагає уважного планування та моніторингу використання ресурсів.
3. Залежність від провайдера: при використанні специфічних AWS рішень та інструментів може виникнути залежність від платформи, що ускладнює міграцію або інтеграцію з іншими сервісами або провайдерами.
4. Оптимізація ресурсів: необхідність постійної оптимізації використання ресурсів для уникнення надмірних витрат, що може бути складною задачею у великих та складних середовищах [17].

Amazon Web Services продовжує бути одним з найбільш потужних та популярних хмарних рішень на ринку, пропонуючи великий вибір сервісів для розгортання та управління обчислювальними ресурсами. Воно підходить для різноманітних використань, від стартапів до великих корпорацій, завдяки своїй масштабованості, безпеці та гнучкості. Проте, успішне використання AWS вимагає ретельного планування, управління та оптимізації для забезпечення ефективності та економічності обчислень.

2.4 Технології розширення функціоналу

2.4.1 Реалізація Webhook для специфічної логіки обробки подій

Webhook є потужним інструментом, який дозволяє додаткам або сервісам отримувати миттєві повідомлення про події або зміни стану в інших системах, сервісах або компонентах. Вони діють як зворотні виклики, що активуються певними подіями (рис. 2.11).

Рисунок 2.11 – Схема життєвого циклу запиту до API Kubernetes, ди видно принцип роботи webhook

Можна виділити список причин, чому варто використовувати Webhook.

1. Реактивність: Webhook дозволяє системам швидко реагувати на події в режимі реального часу, забезпечуючи своєчасну обробку та відгук.
2. Ефективність: замість постійного опитування джерела на наявність змін, Webhook сповіщають систему про конкретні події, ефективно знижуючи навантаження та зайве використання ресурсів.
3. Гнучкість інтеграції: їх можна легко інтегрувати з багатьма веб-службами, інструментами та кастомними рішеннями, дозволяючи налаштувати обробку подій під конкретні бізнес-процеси.
4. Кастомізація обробки подій: розробники можуть визначати специфічну логіку обробки для кожної події, оптимізуючи процеси та поведінку системи.

Але у такої реалізації є ряд моментів, на які треба звертати увагу.

1. Безпека: вебхуки можуть створювати потенційні вектори атак для системи, особливо якщо повідомлення не належним чином захищені або перевіряються на автентичність та цілісність.
2. Надійність: Webhook залежать від зовнішніх сервісів, і їх успішна доставка може залежати від багатьох факторів, включаючи мережеву інфраструктуру та наявність вебхуку приймаючого сервера.
3. Комплексність управління: управління, моніторинг та відладка вебхуків може бути складним, оскільки вони часто використовуються в розподілених та динамічних середовищах.
4. Обмеження в контролі: приймаюча сторона має менше контролю над тим, коли та як події будуть відправлені, що може призвести до проблем з синхронізацією або обробкою подій [18, 19].

Webhook є важливим інструментом у сучасній архітектурі програмного забезпечення, забезпечуючи високу реактивність та інтеграцію серед різних сервісів і додатків. Вони відіграють ключову роль у створенні гнучких та ефективних систем, здатних швидко реагувати на зміни та події. Однак, для ефективного використання вебхуків, необхідно ретельно планувати їх

імплементацію, забезпечуючи належну безпеку, надійність та управління. Розуміння того, як вони інтегруються та взаємодіють з іншими компонентами системи, є ключовим для їх ефективного використання.

2.4.2 Використання Sidecar патерну

Sidecar патерн є популярним структурним патерном у мікросервісній архітектурі, що використовується для додавання функціональності до основного контейнера без безпосереднього змінення його. Sidecar працює як допоміжний контейнер, що запускається поряд з основним контейнером, дозволяючи ізолювати та управляти певними аспектами додатка (рис 2.12).

Рисунок 2.12 – Схема відносин між основним сервісом та sidecar'ом.

Переваги використання Sidecar патерну мають багато пунктів.

1. Модульність та ізоляція: Sidecar дозволяє ізолювати специфічні завдання або сервіси, такі як логування, моніторинг або безпека, від основного додатку, спрощуючи управління та оновлення.

2. Легкість інтеграції: за допомогою sidecar можна легко додавати нові компоненти та функціональності до існуючих додатків без необхідності зміни основного контейнера.
3. Відновлюваність та масштабованість: оскільки sidecar є окремими контейнерами, їх також можна масштабувати та оновлювати, що забезпечує високу відновлюваність та гнучкість системи.
4. Повторне використання: спільні функції, такі як автентифікація, шифрування або конфігурація, можуть бути реалізовані в sidecar, що дозволяє їх легко перевикористовувати між різними додатками.

Але є і перелік певних недоліків, на які слід звертати увагу.

1. Збільшення складності: використання sidecar може збільшити загальну складність системи, оскільки кожен sidecar потребує окремого управління, моніторингу та налаштування.
2. Використання ресурсів: кожен sidecar використовує додаткові обчислювальні ресурси, що може призводити до збільшення витрат на хостинг та розгортання, особливо у великих системах.
3. Управління мережею: Sidecar часто комунікують з основними контейнерами через локальну мережу, що може вимагати додаткових налаштувань та оптимізації мережевого трафіку.
4. Консистентність конфігурації: підтримка консистентності конфігурації та версій між основними контейнерами та їх sidecars може бути викликом у динамічно міняючихся середовищах [20].

Sidecar патерн є важливою складовою у розробці мікросервісних архітектур, забезпечуючи модульність, гнучкість та ефективність управління окремими аспектами додатків. Використання sidecar підвищує здатність системи до масштабування, ізоляції завдань та повторного використання компонентів. Проте, при впровадженні sidecar необхідно уважно зважити потенційне збільшення складності та ресурсоємності для забезпечення оптимальної працездатності та вартісної ефективності системи.

2.4.3 Розробка та впровадження Operators у Kubernetes

Operators у Kubernetes є методом розширення функціональності платформи, щоб автоматизувати розгортання та управління складними додатками. Цей підхід використовує специфічні знання про додатки для створення програм, які діють як роботи-адміністратори в середовищі Kubernetes (рис 2.13).

Рисунок 2.13 – Схема роботи Kubernetes оператору

Нижче наведено ряд причин чому варто використовувати оператори Kubernetes.

1. Автоматизація управління: Operators дозволяють автоматизувати складні та повторювані задачі управління додатками, такі як розгортання, оновлення, резервне копіювання та відновлення.

2. Специфічна логіка додатку: вони вбудовують знання про конкретний додаток або сервіс прямо у Kubernetes, дозволяючи тонко налаштовувати поведінку системи під конкретні вимоги.
3. Зниження помилок: автоматизація за допомогою Operators може зменшити людські помилки при управлінні складними додатками.
4. Масштабування та самовідновлення: Operators можуть відслідковувати стан додатків та автоматично вживати заходів для масштабування або відновлення при виявленні проблем.

Також нижче наведемо аспекти, до яких треба ставитись з обережністю.

1. Складність розробки: створення власного Operator вимагає глибокого розуміння Kubernetes API та додатка, що може бути складною та часозатратною задачею.
2. Обмежена універсальність: кожен Operator зазвичай націлений на конкретний додаток або задачу, що обмежує його застосування лише визначеними сценаріями.
3. Управління та підтримка: Operators потребують постійного управління, моніторингу та оновлення для забезпечення їх актуальності та ефективності.
4. Залежність від платформи: Operators тісно інтегровані з Kubernetes, що створює залежність від цієї платформи та її майбутніх оновлень [21].

Operators у Kubernetes представляють собою потужний інструмент для автоматизації управління додатками та сервісами, забезпечуючи високий рівень контролю та ефективності. Вони дозволяють перенести специфічні знання та процедури управління в середовище Kubernetes, спрощуючи та оптимізуючи роботу. Проте, для успішної імплементації Operators важливо враховувати потреби в технічній експертизі, ресурсах на розробку та підтримку, а також визначити правильний баланс між автоматизацією та управлінням у вашому конкретному середовищі.

2.4.4 API Gateway для маршрутизації та управління API сервісів

API Gateway є ключовим компонентом в архітектурі мікросервісів, який діє як єдина точка входу для управління, маршрутизації та оптимізації запитів до різних API сервісів (рис. 2.14).

Рисунок 2.14 – Схема роботи Kubernetes оператору

Переваги використання API Gateway:

1. Агрегація та маршрутизація: API Gateway дозволяє агрегувати різні сервіси та маршрутизувати запити до відповідних API, спрощуючи інтеграцію та управління мікросервісами.
2. Аутентифікація та авторизація: він може обробляти аутентифікацію користувачів та авторизацію запитів, забезпечуючи безпеку та контроль доступу до додатків.
3. Обмеження та квотування: API Gateway часто використовують для впровадження обмежень на використання API, включаючи квотування, обмеження швидкості та інші політики.

4. Моніторинг та аналітика: він може збирати та аналізувати дані про трафік API, дозволяючи оптимізувати роботу та реагувати на зміни у використанні.

Недоліки використання API Gateway:

1. Точка відмови: як єдина точка входу, API Gateway може стати потенційною точкою відмови, що підвищує ризики для доступності всієї системи.
2. Складність конфігурації: налаштування та управління API Gateway може бути складним, особливо в великих та динамічних середовищах з численними сервісами.
3. Затримка: введення додаткового шару маршрутизації може спричинити додаткову затримку в обробці запитів.
4. Обмеження гнучкості: тісна інтеграція з конкретним API Gateway може обмежити можливості міграції або інтеграції з іншими системами та сервісами [22].

API Gateway є важливим елементом у створенні ефективної та безпечної мікросервісної архітектури, надаючи рішення для управління API, маршрутизації, безпеки та моніторингу. Він спрощує розробку та інтеграцію мікросервісів, але вимагає ретельного планування та управління для забезпечення високої продуктивності та надійності. При впровадженні API Gateway необхідно зважити на потенційні виклики, пов'язані з конфігурацією, затримками та точками відмови, щоб максимізувати його переваги та ефективність у вашому середовищі.

2.5 Мови програмування для реалізації допоміжних сервісів

2.5.1 Мова програмування Golang

Golang, або просто Go, є сучасною мовою програмування, розробленою в Google, яка стала популярною через свою простоту, продуктивність та підтримку паралелізму (рис. 2.15). Вона широко використовується у розробці системного та мережевого софту, включно з такими відомими проектами як Docker та Kubernetes, що є написаними на Go.

Рисунок 2.15 – Логотип мови програмування Golang

Нижче наведено список найбільш визначних переваг роботи на golang.

1. Простота та читабельність: Go має чистий синтаксис, що сприяє легкому читанню та написанню коду, роблячи мову доступною для новачків, а також забезпечує швидку розробку.
2. Висока продуктивність: компільована мова з вбудованою підтримкою паралелізму за допомогою горутин, Go забезпечує високу продуктивність та ефективність виконання.
3. Підтримка паралелізму: модель паралелізму в Go, заснована на горутинах і каналах, спрощує написання багатопотокових програм.
4. Широке використання в індустрії: Go є мовою вибору для системних додатків, веб-серверів, інструментів для розгортання та оркестрації, зокрема Docker та Kubernetes. Це означає, що вміння розробляти на Go відкриває доступ до великої кількості інструментів та систем, а також спільноти.

Але, також, є і свій ряд недоліків, що наведено нижче.

1. Обмежена стандартна бібліотека: на відміну від деяких інших мов, стандартна бібліотека Go може бути сприйнята як більш обмежена, вимагаючи зовнішніх пакетів для певних завдань.
2. Специфіка мови програмування Golang: Golang має ряд нюансів в своїх підходах, що можуть здатись незручними програмістам, що звикли до інших мов програмування. Наприклад, специфічна обробка помилок, відсутність класичного спадкування, тощо.
3. Управління залежностями: історично управління залежностями в Go було складним, хоча із введенням модулів ця проблема значно зменшилась [23].

Розробка на Golang пропонує сильну комбінацію простоти, продуктивності та ефективності для паралельного програмування, роблячи її відмінним вибором для сучасних розробок, особливо у сфері мережевих сервісів та системних додатків. Її широке використання в індустрії, зокрема в таких проектах як Docker та Kubernetes, свідчить про її надійність та ефективність. Однак, як і будь-яка технологія, Golang має свої особливості та обмеження, які важливо враховувати при виборі мови для конкретного проекту.

2.5.2 Мова програмування Java

Java є однією з найбільш використовуваних мов програмування у світі, відомою своєю стабільністю, переносимістю та багатим набором функцій. Її широке застосування в різних сферах, особливо у корпоративних додатках та великих системах, робить Java популярним вибором для створення надійних та масштабованих систем.

До переваг використання Java можна віднести наступні пункти.

1. Багата стандартна бібліотека: Java пропонує широкий спектр готових до використання бібліотек та фреймворків, що підтримують розробку у всіх аспектах, від веб-додатків до системного програмування.
2. Багатоплатформність: як частина філософії "Write Once, Run Anywhere", Java додатки можуть бути запущені на будь-якій платформі з підтримкою Java без змін коду.
3. Безпека: Java надає сильні можливості безпеки, включаючи виключення, управління пам'яттю, та доступ до криптографічних інструментів, забезпечуючи надійне середовище для розробки.

Java здобула всесвітне визнання та популярність завдяки цим перевагам, і тисячі найрізноманітніших систем по всьому світу використовують саме її з цих причин. В тому числі завдяки її багатоплатформності досягнутої особливостями виконання Java коду (рис. 2.16).

Рисунок 2.16 – Схема виконання коду на Java

Але є і певні недоліки використання Java:

1. Швидкість: хоча Java часто достатньо швидка, існують сценарії, де вона може бути повільнішою від компільованих мов, таких як C++ або Go, через використання віртуальної машини.

2. Складність мови: деякі розробники вважають Java надто великою та складною через її об'ємну стандартну бібліотеку та строгу структуру.
3. Споживання ресурсів: додатки Java можуть вимагати значних ресурсів по пам’яті, особливо в порівнянні з деякими більш легковажними мовами та фреймворками.
4. Зміни у технологічному стеку: незважаючи на свою стабільність, світ Java постійно еволюціонує, і розробники повинні триматися в курсі останніх змін і трендів [24].

Java залишається однією з найбільш надійних та широко використовуваних мов програмування для створення масштабованих та надійних систем. Її переносимість, підтримка багатопоточності та сильні можливості безпеки роблять її відмінним вибором для корпоративних додатків, великих розподілених систем та інших вимогливих завдань. Однак, при виборі Java для нового проекту важливо враховувати потреби в продуктивності, складності та ресурсах для забезпечення оптимального балансу між потужністю, ефективністю та легкістю розробки.

2.5.3 Мова програмування Python

Python є однією з найпопулярніших мов програмування у світі завдяки своїй універсальності, легкості читання та великій спільноті. Його широке прийняття у веб-розробці, наукових обчисленнях, аналізі даних, штучному інтелекті та автоматизації роблять Python вибором для різних проектів та рішень (рис. 2.17).

Рисунок 2.17 – Емблема мови програмування Python

Python може запропонувати ряд переваг при його використанні.

1. Гнучкість та широке застосування: Python можна використовувати в широкому діапазоні додатків, від веб-розробки до наукових досліджень та машинного навчання, завдяки багатому набору бібліотек та фреймворків.
2. Читабельність та легкість написання: Python відомий своїм чистим і простим синтаксисом, що робить код легким для розуміння та написання, що сприяє швидкій розробці.
3. Сильна спільнота та підтримка: Python має одну з найбільших розробницьких спільнот, забезпечуючи обширну підтримку, численні ресурси для навчання та велику кількість відкритого коду.
4. Міжплатформеність: Python працює на багатьох операційних системах і легко інтегрується з іншими мовами та інструментами.

Серед недоліків Python можна виділити наступні пункти.

1. Швидкість виконання: як інтерпретована мова, Python може бути повільнішим за компільовані мови, хоча це часто компенсується його гнучкістю та продуктивністю розробки.
2. Мобільна розробка: хоча Python використовується для серверної сторони та скриптів, він не є популярним вибором для розробки мобільних додатків порівняно з мовами, спеціалізованими на цьому, як Swift або Kotlin.
3. Використання пам'яті: Python може бути досить вимогливим до пам'яті, особливо у великих або складних додатках.
4. Залежність від сторонніх бібліотек: багато функцій в Python залежать від сторонніх бібліотек та модулів, що може ускладнити управління залежностями та сумісністю [25].

Python є гнучкою, високорівневою мовою, яка підходить для широкого спектру завдань та проектів завдяки своїй читабельності, широкій підтримці та міжплатформенності. Він залишається одним з найбільш затребуваних мов

програмування, особливо в областях, де важлива швидкість розробки та ітерацій. Однак, важливо розуміти його обмеження щодо швидкості виконання та ресурсоємності для вибору правильного інструменту для вашого конкретного випадку використання.

2.5.4 Мова програмування Node.js

Node.js є популярним середовищем виконання JavaScript, яке дозволяє розробляти швидкі та масштабовані мережеві застосунки. Це відкрите середовище є особливо популярним для створення веб-серверів, мікросервісів та інших систем, які потребують високої продуктивності при обробці асинхронних подій (рис. 2.18).

Рисунок 2.18 – Емблема мови програмування Node.js

Нижче приведено переваги, що може надати Node.js.

1. Швидкодія: Node.js побудовано на V8 JavaScript Engine від Google, що забезпечує високу швидкість виконання коду. Його неблокуюча архітектура оптимізована для асинхронних операцій, що робить його ідеальним для мережевих застосунків.
2. Один мова по всьому стеку: використання JavaScript на як фронтенді, так і бекенді спрощує розробку та підтримку застосунків.
3. Великий екосистема: NPM (Node Package Manager) надає величезну кількість модулів та бібліотек, що полегшують інтеграцію різноманітних функцій і сервісів.

4. Спільнота та підтримка: Node.js має велику та активну спільноту, що надає широкий спектр ресурсів, інструментів та підтримки для розробників.

А також нижче наведено проблеми, що можуть виникнути під час розробки на Node.js.

1. Обробка тяжких обчислень: Node.js менш ефективний для застосунків, що вимагають інтенсивних обчислень, через свою однопоточну природу.
2. Зворотна сумісність: швидкий розвиток Node.js та його бібліотек може призводити до проблем зі зворотною сумісністю та необхідністю частого оновлення коду.
3. Управління ресурсами: Node.js вимагає уважного управління пам'яттю та ресурсами, особливо в великих та складних застосунках [26].

Node.js продовжує бути популярним вибором для розробки легких, швидких та ефективних мережевих застосунків. Його асинхронна натура, разом з великою спільнотою та екосистемою модулів, роблять його ідеальним для створення різноманітних веб-сервісів та API. Проте, важливо розуміти обмеження та виклики, пов'язані з Node.js, для ефективної розробки та підтримки вашого застосунка.

Висновки до розділу 2

У цьому розділі було представлено широкий огляд архітектур, технологій, патернів та мов програмування, які є актуальними для розробки сучасних, надійних та масштабованих систем. Таким чином, аналіз кожної технології допоміг визначити їхні переваги та недоліки, що є ключовим для інформованого вибору інструментів розробки. Отже, від архітектур хмарних рішень, таких як AWS, GKE та Azure, до мов програмування, як Golang, Java та Python, кожен

інструмент чи метод був оцінений з огляду на його придатність для певних сценаріїв використання.

3 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

3.1 Загальна структура системи

У системі активно використовуються хмарні рішення з метою забезпечення масштабованості, надійності та гнучкості. Хмарна інфраструктура дозволяє ефективно розгортати та управляти сервісами, оптимізувати витрати та забезпечити високу доступність системи.

Для розгортання використано Docker, Kubernetes та Helm. Ці технології були обрані через їх здатність забезпечувати консистентність середовища, спрощення оркестрації контейнерів та ефективне управління конфігураціями. Docker забезпечує легку контейнеризацію додатків, Kubernetes — ефективну оркестрацію та масштабування, а Helm спрощує управління пакетами та розгортанням.

У якості хмарної платформи було обрано GKE. Google Kubernetes Engine було обрано як основу для хмарної інфраструктури через його інтегровані можливості управління Kubernetes (наприклад, автопілот, що спрощує налаштування системи), автоматизоване масштабування, високу надійність та тісну інтеграцію з іншими сервісами Google Cloud.

Використання Webhook для розгортання: Основною технологією для динамічного управління конфігурацією та додавання специфічної логіки обробки подій обрано Webhook. Це рішення дозволяє реалізувати необхідні налаштування, такі як topology та affinity, і тим самим забезпечити високу доступність та уникнути Single Point Of Failure. Використовуючи цю технологію як основний інструмент буде досягнуто мету роботи.

Також, окрім цього, була проведена інтеграція оператора для PostgreSQL. Для ефективного управління базами даних у Kubernetes середовищі використовується PostgreSQL Operator. Це забезпечує автоматизацію багатьох

задач, пов'язаних з управлінням, резервним копіюванням, відновленням та масштабуванням баз даних.

Також система включає Ingress для ефективної маршрутизації та управління зовнішнім трафіком. Це не тільки забезпечує доступність послуг, але й створює основу для можливої інтеграції API Gateway, що дозволить оптимізувати управління API та маршрутизацію запитів у майбутньому.

За допомогою продуманої архітектури, систмема має повну готовність до впровадження Sidecar'ів. Система спланована таким чином, що в будь-який момент може бути інтегровано Sidecar для додаткового моніторингу, логування чи безпеки, наприклад, бази даних. Впровадження Sidecar може значно підвищити ефективність та гнучкість системи, забезпечуючи більш глибоке вбудоване управління та аналітику (рис. 3.1).

Рисунок 3.1 – Загальна схема роботи запропонованого Sidecar

Завдяки вдалому вибору технологій та платформ, система забезпечує стабільну основу для розгортання, масштабування та управління мікросервісною архітектурою, водночас зберігаючи гнучкість для майбутньої інтеграції та оптимізації. Це можливо також за допомогою менеджеру сертифікатів, що забезпечує ще більше можливостей для подальшого розвитку системи (рис. 3.2).

Рисунок 3.2 – Загальна схема роботи webhook, що допоможе досягти заявленої мети роботи

Там чином, система задовільняє найвищім критеріям якості та може мати багато перспектив у подальшому розвитку.

3.2 GKE кластер та його налаштування

Створення та розгортання GKE кластеру обов'язково складається з ряду необхідних кроків.

Перш за все наобхідно необхідно мати реєстрацію на Google Cloud Platform (GCP), створити аккаунт на платформі GCP, що є передумовою для користування її сервісами (рис. 3.3).

Рисунок 3.3 – Завершена реєстрація у GCP

Після реєстрації необхідно мати новий проект, бо на платформі GCP це є бажаною дією для кожного нового завдання чи додатку (рис 3.4). Кластер, що буде містити додаток та всі надбудови названо «kpi-iate-course-picker».

Рисунок 3.4 – Новий проект, де буде розміщено додаток

Окрім цього, також треба мати всі необхідні для роботи API активованими. Таким чином, для коректної роботи кластеру було активовано декілька необхідних API, включно з Kubernetes API (рис 3.5).

Рисунок 3.5 – Активоване АПІ GKE

Окрім цього, ще будуде заразервована IP адреса (рис. 3.6), та активоване АПІ для зберігання контейнерів (рис. 3.7). Все це дозволить отримати повністю працюючий додаток з доступом до мережі.

Рисунок 3.6 – Процес створення статичної IP адреси

Рисунок 3.7 – АПІ для контейнерів перед активацією

Основним елементом є GKE кластер, де буде розміщенно додаток (рис 3.8).

Рисунок 3.8 – Новий створений клстер, де буде розміщено додаток

Щоб отримати доступ до цього кластеру, необхідно мати зарезервовану IP-адресу. Таким чином можна гарантувати стабільний доступ до ресурсів кластеру (рис 3.9).

Рисунок 3.9 – Зареєстрована статична IP адреса, за якою буде доступен додаток

Тепер, щоб мати можливість повноцінно працювати з кластером як розробник, необхідна установка та конфігурація інструментів. Для цього необхідні такі інструменти як gcloud (рис 3.10) та kubectl (рис 3.11), а також повинна бути наявна їх первинна конфігурація та авторизація на платформі GCP.

Рисунок 3.10 – Установлений gcloud cli

Рисунок 3.11 – Установлений kubectl cli

Тепер, маючи все необхідне, треба створити та налаштувати Ingress. Це необхідно для того, щоб забезпечити доступ до сервісів кластеру за допомогою зарезервованого зовнішнього IP.

Щоб мати коректне налаштування, нам буде необхідно додати у конфігураційний файл ingress посилання на зарезервований IP, що буде виглядати як: «kubernetes.io/ingress.global-static-ip-name: course-picker-ip-v1». І, після цього, за допомогою команди «kubectl apply» цей інгресс буде розгорнуто на кластері (рис. 3.12).

Рисунок 3.12 – Успішно запущений у кластері ingress

Таким чином, за допомогою статичного IP та Ingress, до кластеру був отриманий стабільний доступ.

3.3 Deployment файли та розгорнуті мікросервіси додатку

Для реалізації сервісу, основаного на архітектурі мікросервісів, важливим кроком є створення та розгортання конфігураційних файлів для кожного з цих мікросервісів.

Першочерговою дією є створення необхідних deployment файлів. Для кожного мікросервісу було написано окремий конфігураційний файл у форматі YAML. Ці файли вказують на необхідні ресурси, змінні оточення, відкриті порти та інші специфікації для коректної роботи мікросервісів (рис 3.13).

Рисунок 3.13 – Частина написаних deployment файлів

До важливих змінних, що будуть у таких файлах, можна віднести:

- selector – механізмом для ідентифікації та вибору ресурсів в системах, що працюють з метаданими, наприклад «app: backend»;
- посилання на Docker «image» – місце де зберігається сам мікросервіс, наприклад: «europe-west3-docker.pkg.dev/kpi-iate-course-picker/kpi-iate-course-picker-registry/backend:latest»;
- порт контейнеру – порт, на якому буде «слухати» контейнер, наприклад «containerPort: 5000»;
- виділені ресурси – кількість ресурсів кластера, що може використовувати додаток, наприклад: «cpu: 100m»;
- лейбли – на них, наприклад, буде орієнтуватись webhook, що буде написано: «affinity-inject: any».

Також варто не забути про написання deployment файлу для сервісу – сутності, яка необхідна для того, щоб мікросервіси могли обмінюватись інформацією між собою. До їх важливих налаштувань можна віднести:

- selector – той самих механізм для ідентифікації та вибору ресурсів в системах, наприклад «app: frontend»;
- port – це порт, на якому служба або додаток буде доступний ззовні, наприклад «80»;

- targetPort – це порт контейнера, до якого буде перенаправлятися трафік, який приходить на зовнішній порт, наприклад «8080».

Також варто окремо виділити змінні середовища, в яких буде передаватись конфігурація для самого додатку. Таким чином буде передано пароль та логін до бази даних з секретного сховища K8s: «env: - name: POSTGRES_PASSWORD» (рис 3.14).

Рисунок 3.14 – Файл типу Secret, де будуть зберігатись облікові дані користувача БД

Щоб отримати цей секрет да доступ до БД, необхідно розгорнути оператор PostgreSQL. Оператори Kubernetes дозволяють автоматизувати рутинні завдання з управління ресурсами в кластері. Установка оператора PostgreSQL пройшла за допомогою helm, оператор був доданий до репозиторію та розгорнутий за допомогою команди: «helm install postgres-operator postgres-operator-charts/postgres-operator». За допомогою цього було полегшено управління базою даних PostgreSQL, та гарантували її стабільну роботу та автоматизацію бекапів (рис 3.15).

Рисунок 3.15 – Успішно розгорнутий PostgreSQL оператор

Щоб за допомогою оператору створити БД, достатньо просто написати та використати конфігураційний файл YAML, основними параметрами якого будуть такі поля як:

- унікальне ім'я PostgreSQL кластера в середовищі Kubernetes, наприклад «name: acid-minimal-cluster»;

- обсяг дискового простору, виділеного для даного кластера, наприклад «size: 1Gi»;
- база даних та її користувачі, такі як «courses_db: backend_user»;
- версія бази даних, наприклад «version: "15"».

В результаті цих дій можна отримати функціонуючу БД (рис. 3.16).

Рисунок 3.16 – Успішно розгорнута за допомогою оператора PostgreSQL БД

Після цього можна почати розгортання мікросервісів. За допомогою інструменту kubectl, YAML конфігураційні файли були застосовані для розгортання мікросервісів в середовищі GKE кластеру (рис 3.17).

Рисунок 3.17 – Успішно розгорнутий фронтенд мікросервіс

Окрім того, буде потрібна інфраструктура для подальших дій. Для цього необхідно встановлення cert-manager. Спочатку необхідно взяти основні файли з офіційного репозиторію (рис. 3.18), та розгонути їх за допомогою kubectl (рис 3.19).

Рисунок 3.18 – Файли, необхідні для розгортки cert-manager

Рисунок 3.19 – Успішно розгорнутий cert-manager

Завдяки цим діям, сервіс буде успішно розгорнуто в Kubernetes середовищі, що готове до масштабування та подальшого розвитку.

3.4 Injector service'y та webhook

Для розширення можливостей сервісу, підвищенню його надійності, необхідно створити webhook.

Почати це варто з створення кодової бази для нього, зроблено це буде за допомогою мови програмування golang та середовища Goland IDE (рис. 3.20).

Рисунок 3.20 – Структура проекту mutating webhook'а на golang

Цей Webhook буде служити для перехоплення запитів до Kubernetes API та модифікації об'єктів перед їхнім створенням чи оновленням. Для реалізації webhook'у були використані такі необхідні бібліотеки як, наприклад, k8s.io.

Для забезпечення безпечного з'єднання між Kubernetes API сервером та webhook'ом, та нормальної роботи додатку було згенеровано TLS сертифікати (tls.key та tls.crt) за допомогою спеціалізованого скрипта. Роботу цього скрипта можна коротко описати наступними кроками:

1. Параметри APP та NAMESPACE беруться як аргументи командного рядка і використовуються для генерації імені CSR (Certificate Signing Request).
2. За допомогою openssl генерується приватний ключ (${APP}.key).

79

3. Створюється CSR (${APP}.csr) з конфігураційним файлом, який визначає атрибути сертифіката.
4. Створюється новий об'єкт CertificateSigningRequest в Kubernetes з генерованим CSR.
5. Скрипт очікує, поки CSR буде присутнім в Kubernetes та затверджений.
6. За допомогою kubectl отримується затверджений сертифікат.
7. Затверджений сертифікат декодується з base64 і зберігається у файл ${APP}.pem.

Після використання цього скрипта, було згенеровано ряд файлів, серед яких є потрібні нам сертифікати (рис. 3.21).

Рисунок 3.21 – Файли, що було згенеровано скриптом

Результуючі файли варто завантажити у Kubernetes (рис. 3.22).

Рисунок 3.22 – Створений секрет з сертифікатами

І тепер, коли вся необхідна інфраструктура готова, можна почати створення конфігураційних файлів для webhook. Для реєстрації та коректної роботи webhook'y в Kubernetes середовищі було створено додаткові YAML конфігураційні файли. Один з найважливіших серед них – MutatingWebhookConfiguration: цей

файл визначає налаштування mutating webhook'y та вказує, які об'єкти та події мають бути перехоплені. Одними з його головних параметрів можна виділити:

- унікальне ім'я конфігурації webhook'y, наприклад «name: sidecar-injector-configuration»;
- анотації, що вказують джерело сертифіката, наприклад «cert-manager.io/inject-ca-from: default/selfsigned-cert»;
- параметри доступу до написаного сервісу, який обробляє запити з такими полями, як «name: sidecar-injector-v2», «path: /mutate», «port: 8443» та «namespace: "default"»;
- правила, що визначають, які ресурси та операції слід обробляти, наприклад, операції «CREATE» та «UPDATE» для ресурсів «pods».

З завершенням цих кроків, injector service та Webhook були успішно інтегровані в архітектуру сервісу, гарантуючи додатковий рівень контролю та безпеки при роботі з ресурсами в Kubernetes середовищі (рис. 3.23).

Рисунок 3.23 – Успішно розгорнутий Webhook сервіс

У результаті було отрмано повністю функціонуючий додаток, що задовільняє усім поставленим вимогам.

Висновки до розділу 3

В результаті роботи над розділом, було проведено процеси розгортання системи в GCP, що включали в себе такі кроки як: реєстрацію на платформі, створення нового проєкту та кластеру, активацію АПІ, резервацію IP адреси, розгортання додатків та необхідних елементів інфраструктури, тощо. Отже, було успішно виконано налаштування проєкту, а також написано та розгорнуто додаткові служби, що будуть забезпечувати додаткову надійність системи. Таким

чином процес розгортання системи було здійснено за допомогою deployment файлів у форматі YAML, де для кожного мікросервісу задано свої специфікації.

4 ІНСТРУКЦІЯ КОРИСТУВАЧА

Розглянемо функціональність, яку надала системі розгортка на GKE. Головна сторінка відображає основну доступну функціональність, наприклад, інформацію про виставлення рахунків за користування GKE, кількість запитів до кластеру по часу, назву проєкту, тощо (рис 4.1).

Рисунок 4.1 – Головна сторінка проекту

З головної сторінки можна перейти на інші сторінки з інформацією про проєкт. На сторінці з інформацією про кластер можна побачити панелі, у яких будуть відображатись працюючі поди, сервіси, секрети, конфігурації, тощо. Наприклад, у панелі Services & Ingress можна побачити сервіси, що забезпечують зв'язок по мережі для подів (рис. 4.2).

Рисунок 4.2 – Панель «Services & Ingress»

У тій самій панелі знаходиться інгресс, за допомогою якого можна сконфігурувати доступ до кластеру з інтеренету, налаштувати IP адреси, тощо. Кожен запис про сервіс чи інгресс може бути відкритий та модифікований за потреби. Так, наприклад, можна переглянути деталі про налаштування, змінити конфігурацію, подивитись записи логів, події, що сталися, і т.д. (рис. 4.3). Аналогічні дії можна виконати і з записами на інших панелях з інформацією про об’єкти системи.

Рисунок 4.3 – Деталі об’єкту «app-ingress»

Окрім цього, важливим досягненням є можливість чітко контролювати права доступу у кожного користувача системи, адміністратора, тощо. Це дає змогу забезпечити високий рівень безпеки та гнучкості в управлінні ресурсами. Досягається це за допомогою політик Identity and Access Management (IAM), що можна налаштувати у відповідному розділі (рис. 4.4). IAM системи дозволяють детально визначати, хто має доступ до яких ресурсів і в який спосіб цей доступ може бути використаний.

Рисунок 4.4 – Сторінка налаштування політик доступу IAM

Це дозволить не перйматись за безпеку та надійність збереження своїх данних, бо на відміну від залежності від зовнішнього провайдера послуг, який буде контролювати дані системи, тут контроль буде забезпечуватись на значно більш детальному рівні.

Також, розгорнуті мікросервіси додатку мають ряд важливих можливостей, що було досягнуто. Наприклад, для frontend мікросервісу можливо подивитись на кількість помилок, що виникають в залежності від запиту, на загальну кількість різних запитів, затримку запитів, утилізацію ресурсів, тощо. Статистику можна дивитись за різні періоди, наприклад: за останню годину, за останній день, за остінній місяць, та не тільки (рис 4.5).

Рисунок 4.5 – Статистика по мікросервісу «frontend»

За необхідності, по кластеру можна подивитись і інші характеристики, наприклад, такі як використання додатком процессора, кількість задіяних ядер, і т.д. (рис 4.6). За необхідності можливо підключити профайлери, аналітику даних, і інші корисні інструменти для догляду за станом додатку.

Це є дуже важливим досягненням, бо гарний моніторинг системи дозволяє завчасно бачити та уникати великої кількості проблем. Наприклад, на відміну від наявної системи my.kpi.ua, це дозволить чітко бачити та аналізувати піки навантажень та адаптовуватись під них. Також, застосування передових методів моніторингу дає можливість прогнозувати потенційні збої та автоматично реагувати на них, забезпечуючи високу доступність та надійність сервісів. Все це в сукупності забезпечує вищу якість обслуговування користувачів та сприяє підвищенню загальної продуктивності системи.

Рисунок 4.6 – Статистика утилізації ресурсов процесора

Так само як і з іншими об’єктами, мікросервіси додатку можна конфігурувати різноманітними способами. Це можна зробити у панелі «Workloads» (рис. 4.7).

Рисунок 4.7 – Панель «Workloads» з інформацією про мікросервіси системи

Переглянути та змінити конфігурацію можна в будь-який момент у відповідній об’єкт мікросервісу (рис. 4.8). Наприклад, можна додати змінні

середовища для конфігурації самого додатку, чи додати налаштування для забезпечення додаткової стабільності та найдійності.

Рисунок 4.8 – Конфігурація мікросервісу «frontend» з можливістю коригування

Також, окрім моніторингу показників самого сервісу, можна побачити ревізії будь яких мікросервісів (рис. 4.9).

Рисунок 4.9 – Ревізії версій мікросервісу «sidecar-injector»

Так само можна побачити різні події, що трапилися з деплойментом, наприклад, горизонтальне розширення чи зменшення кількості одночасно працюючих сервісів (рис. 4.10).

Рисунок 4.10 – Події, що сталися з сервісом «frontend»

Також варто зазначити сховище різних конфігурацій та секретів, де можна в будь який момент швидко передивитись та поправити різноманітні конфігурації системи (рис. 4.11).

Рисунок 4.11 – Сховище конфігурацій та секретів

Додатку необхідна база даних, яка була розгорнута за допомогою оператора. Таким чином, база даних буде дуже просто розгортатись, підтримувати стабільний

стан, і не тільки. Деталі про фізичне місце збереження даних можна подивитись у відповідній панелі GKE (рис. 4.12)

Рисунок 4.12 – Конфігурація мікросервісу «frontend» з можливістю коригування

Саму БД, як і будь які іниші об’єкти в системі, можна скейлити, тобто збільшувати кількість одночасно працюючих баз даних, що дозволяє підвищити продуктивність додатку (рис. 4.13). Це можна робити як і вручну, так і довіритись автоматичному скейлеру.

Рисунок 4.13 – Горизонтальне розширення бази даних

Це надає базі даних такі важливі аспекти як легкість у контролюванні, надійність, можливість горизонтального розширення та можливість легкої інтеграції з іншими додатками системи.

Основний сервіс, який був написаний для досягення мети роботи, можна так само побачити серед інших об’єктів системи. Для нього теж можна передивитись записи логів, деталі, події, що сталися, і т.д. (рис 4.14).

Рисунок 4.14 – Детальна інформація про історію ревізій Webhook сервісу

Також можна налаштувати події та типи об’єктів на які буде працювати цей Webhook (рис 4.15).

Рисунок 4.15 – Можливі поля конфігурації правил роботи webhook

Це забезпечує гнучкість для системи, що дозволяє як впровадити інший функціонал за потреби, так і відкорегувати поточний.

Поточний функціонал, за результатами тестування, працює справно, та забезпечує додаткову надійність системі уникаючи єдиної точки відмови за допомогою налаштувань Kubernetes, що є відмінною рисою у порівнянні з аналогами (рис. 4.16).

Рисунок 4.16 – Інформація про додані налаштування

За необхідності, Docker image’и, які є запакованими додатками, готовими до розгортки у системі, можна побачити у відповідному реєстрі. Тут можна переглянути інформацію про них, видалити, організувати їх, тощо (рис. 4.17).

Рисунок 4.17 – Інформація про додані налаштування

Таким чином можна провести ревізію наявних «артефактів», що наявні у системі.

У окремому реєстрі можна побачити IP адреси, що зарезервовано для системи. Тут можна забронювати нові, видалити старі, і т.д. (рис. 4.18).

Рисунок 4.18 – Реєстр IP адрес

За допомогою них можна відкрити сам додаток до інтернету (рис. 4.19).

Рисунок 4.19 – Головна сторінка розгорнутого додатку

На окремій вкладці слід зазначити наявність платіжної інформації, де буде інформація про те, скільки коштує розміщення додатку, переглянути статистику витрат, плани витрат, та не тільки (рис. 4.20).

Рисунок 4.20 – Сторінка з рахунками по проекту

В результаті, було детально розглянуто функціональність розробленого проєкту.

Висновки до розділу 4

У результаті роботи над розділом була продемонстрована ефективність запропонованої системи, що базується на хмарному рішенні GKE та використовує Docker, Kubernetes, та Helm для розгортання. Результати тестування підтверджують значне покращення в управлінні ресурсами, досутпом, моніторингом та у масштабуванні. Також вони вказують на гнучкість, надійність та масштабованість системи, підкреслюючи новизну та ефективність рішення порівняно з існуючими аналогами. Таким чином, дослідження підтверджує потенціал системи для подальшого розширення та адаптації в освітньому середовищі, відкриваючи нові можливості для ефективного та сучасного управління навчальними процесами.

ВИСНОВКИ

У магістерській дисертації було досягнуто значних результатів у розробці та покращенню системи вибору вибіркових дисциплін для університетського середовища, що базується на сучасних технологіях та архітектурах. Робота визначила актуальність та потребу у розробці системи, аналізуючи існуючі аналоги та їх обмеження. Було проведено детальний огляд аналогів, сучасних технологій та платформ, що дозволило обрати оптимальні інструменти для розробки системи, зокрема GKE, Docker, PostgreSQL, та інші.

Реалізація системи включала налаштування та розгортання системи на хмарній платформі GKE, оптимізацію ресурсів за допомогою webhook, що впроваджує topology та affinity налаштування для запобігання ризикам Single Point Of Failure, а також створення надійної та ефективної системи баз даних. Результати тестування та аналізу підтверджують збільшення продуктивності, масштабованості та надійності системи, підкреслюючи її переваги перед існуючими аналогами.

У розділі, присвяченому ринковій стратегії та маркетинговій програмі, було визначено ключові цільові аудиторії та стратегії позиціонування продукту, які вказують на потенціал для подальшого розширення та зростання системи. Аналіз викликів та можливостей дозволяє сконцентруватися на подальшому удосконаленні та адаптації системи до потреб освітнього середовища.

Таким чином, робота не лише вирішила актуальну наукову проблему розробки надійної, масштабованої та ефективної системи для вибору вибіркових дисциплін, а й забезпечила підґрунтя для подальшого розвитку та інтеграції системи у університетську інфраструктуру. Результати демонструють не лише наукову цінність роботи, але й її практичну значущість та ефективність, підкріплену кількісними показниками та обґрунтуванням достовірності результатів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Опитування щодо якості навчання Магістри Аналітика. URL: https://ipze.kpi.ua/wp-content/uploads/2023/10/Опитування_щодо_якості_навчання_Магістри_Аналітика_1.pdf. (дата звернення: 02.01.2023).

2. Ellucian Banner. URL: https://www.ellucian.com/solutions/ellucian-banner. (дата звернення: 02.01.2023).

3. Student Information System. URL: https://www.blackbaud.com/solutions/organizational-and-program-management/student-information. (дата звернення: 02.01.2023).

4. KUALI STUDENT. URL: https://www.kuali.co/products/student. (дата звернення: 02.01.2023).

5. PeopleSoft Campus Solutions. URL: https://docs.oracle.com/en/applications/peoplesoft/campus-solutions/index.html. (дата звернення: 02.01.2023).

6. The Smart Choice for Higher Education. URL: https://jenzabar.com/about. (дата звернення: 02.01.2023).

7. CampusNexus Student Features. URL: https://www.campusmanagement.com/products/campusnexus-student-features/. (дата звернення: 02.01.2023).

8. A Comprehensive Guide For On-premise vs Cloud Computing Tutorial. URL: https://www.edureka.co/blog/on-premise-vs-cloud-computing/. (дата звернення: 02.01.2023).

9. Infrastructure as a Service (IaaS) in Cloud Computing. URL: https://www.mongodb.com/cloud-explained/iaas-infrastructure-as-a-service. (дата звернення: 02.01.2023).

10. What is PaaS (Platform as a Service) in Cloud Computing? URL: https://www.simplilearn.com/what-is-paas-article. (дата звернення: 02.01.2023).
11. Turnbull J. The Docker Book. Lulu.com, 2023. 386 C.
12. Gkatziouras E. A Developer's Essential Guide to Docker Compose: Simplify the development and orchestration of multi-container applications. Packt Publishing Ltd; 2022. C. 264.
13. Poulton N. The Kubernetes Book. NIGEL POULTON LTD; 2023. C. 311.
14. Using Helm. URL: https://helm.sh/docs/intro/using_helm/. (дата звернення: 02.01.2023).
15. Collier M, Shahan R. Microsoft Azure Essentials - Fundamentals of Azure. Microsoft Press; 2015. C. 246.
16. The most scalable and fully automated Kubernetes service. URL: https://cloud.google.com/kubernetes-engine?hl=en. (дата звернення: 02.01.2023).
17. Burnett SM. Aws for Beginners: The Ultimate Guide to the Fundamentals of Aws. Independently Published; 2021. C. 118.
18. Dynamic Admission Control. URL: https://kubernetes.io/docs/reference/access-authn-authz/extensible-admission-controllers/. (дата звернення: 02.01.2023).
19. Writing a very basic kubernetes mutating admission webhook. URL: https://medium.com/ovni/writing-a-very-basic-kubernetes-mutating-admission-webhook-398dbbcb63ec. (дата звернення: 02.01.2023).
20. Sidecar Containers. URL: https://kubernetes.io/docs/concepts/workloads/pods/sidecar-containers/. (дата звернення: 02.01.2023).
21. Operator pattern. URL: https://kubernetes.io/docs/concepts/extend-kubernetes/operator/. (дата звернення: 02.01.2023).
22. Gateway API. URL: https://kubernetes.io/docs/concepts/services-networking/gateway/. (дата звернення: 02.01.2023).

23. GoLang — The Good, the Bad and the Ugly. URL: https://medium.com/geekculture/golang-the-good-the-bad-and-the-ugly-880270a85848. (дата звернення: 02.01.2023).
24. Overview of Java. URL: https://docs.oracle.com/en/database/oracle/oracle-database/19/jjdev/Java-overview.html. (дата звернення: 02.01.2023).
25. What is Python Language? Overview of the Python Language. URL: https://www.scholarhat.com/tutorial/python/overview-of-the-python-language. (дата звернення: 02.01.2023).
26. Introduction to Node.js. URL: https://nodejs.org/en/learn/getting-started/introduction-to-nodejs. (дата звернення: 02.01.2023).
27. Education ERP Market Forecast & Report Summary, Global Size. URL: https://www.marketsandmarkets.com/Market-Reports/education-erp-market-225190725.html. (дата звернення: 02.01.2023).
28. How to measure the ROI of tech in higher education. URL: https://www.fullfabric.com/articles/how-to-measure-the-roi-of-tech-in-higher-education. (дата звернення: 02.01.2023).

Matches

Internet sources 207

5	https://ela.kpi.ua/bitstream/123456789/51707/1/Komisarov_magistr.pdf		1.2%
6	https://ela.kpi.ua/handle/123456789/51601		1.13%
7	https://ela.kpi.ua/bitstream/123456789/58305/1/Svyrydenko_magistr.pdf		1.08%
11	https://ela.kpi.ua/handle/123456789/51713		0.94%
12	https://ela.kpi.ua/handle/123456789/51869		0.94%
13	https://ela.kpi.ua/bitstream/123456789/59069/1/Boldyr_bakalavr.pdf		0.93%
15	https://ela.kpi.ua/bitstream/123456789/51713/1/Roienko_magist.pdf		0.85%
16	https://ela.kpi.ua/bitstream/123456789/51869/1/Kuzmenchuk_magistr.pdf		0.85%
17	https://ela.kpi.ua/handle/123456789/51702		0.81%
18	https://ela.kpi.ua/handle/123456789/26996		0.77%
19	https://ela.kpi.ua/bitstream/123456789/51600/1/Khluplianets_magistr.pdf		0.77%
20	https://ela.kpi.ua/bitstream/123456789/59128/1/Havryliuk_bakalavr.pdf		0.77%
22	https://ela.kpi.ua/bitstream/123456789/51708/1/Korotych_magistr.pdf	3 Sources	0.73%
23	https://ela.kpi.ua/handle/123456789/51699		0.73%
24	https://ela.kpi.ua/bitstream/123456789/51714/1/Yakubovskii_magistr.pdf		0.73%
28	https://ela.kpi.ua/handle/123456789/25521		0.68%
29	https://ela.kpi.ua/bitstream/123456789/59110/1/Ivashchenko_bakalavr.pdf		0.69%
38	https://ela.kpi.ua/handle/123456789/28874		0.63%
39	https://ela.kpi.ua/bitstream/123456789/57383/1/Magisterska_dysertatsiia_organizatsiia_vykonannia_zahystu_vymohy_do_stru...		0.62%
40	https://ela.kpi.ua/handle/123456789/25791		0.62%

41	https://ela.kpi.ua/handle/123456789/51600	2 Sources	0.61%
43	https://ela.kpi.ua/handle/123456789/49769		0.6%
44	https://ela.kpi.ua/bitstream/123456789/32344/1/Boichuk_magistr.pdf	2 Sources	0.6%
45	https://ela.kpi.ua/handle/123456789/59077?mode=full	6 Sources	0.6%
51	https://ela.kpi.ua/handle/123456789/59076?mode=full	12 Sources	0.58%
55	https://ela.kpi.ua/bitstream/123456789/46017/1/Syrov_magistr.pdf		0.56%
56	https://ela.kpi.ua/bitstream/123456789/26437/1/Romashchenko_magistr.pdf		0.53%
57	https://ela.kpi.ua/handle/123456789/51704?mode=full		0.52%
59	https://ela.kpi.ua/handle/123456789/51865		0.52%
60	https://ela.kpi.ua/handle/123456789/31863	2 Sources	0.51%
61	https://ela.kpi.ua/bitstream/123456789/59078/1/Samko_bakalavr.pdf		0.51%
63	https://ela.kpi.ua/bitstream/123456789/59125/1/Pyrohovska_bakalavr.pdf	2 Sources	0.47%
64	https://ela.kpi.ua/bitstream/123456789/51700/1/Valchun_magistr.pdf		0.47%
65	https://ela.kpi.ua/handle/123456789/59129?mode=full	2 Sources	0.47%
66	https://ela.kpi.ua/bitstream/123456789/57422/1/Bakalavrska_kvalifikatsiyna_robota.pdf		0.46%
67	https://ela.kpi.ua/bitstream/123456789/52686/1/Gukovskiy_magistr.pdf		0.46%
69	https://ela.kpi.ua/handle/123456789/59117?mode=full	5 Sources	0.43%
71	https://ela.kpi.ua/bitstream/123456789/51515/1/Kovyniev_magistr.pdf		0.4%
72	https://ela.kpi.ua/handle/123456789/31855	4 Sources	0.39%
74	https://ela.kpi.ua/handle/123456789/51617	3 Sources	0.39%
76	https://ela.kpi.ua/bitstream/123456789/59076/1/Yermosin_bakalavr.pdf	3 Sources	0.38%
78	https://ela.kpi.ua/bitstream/123456789/52685/1/Khorets_magistr.pdf		0.38%

82	https://ela.kpi.ua/bitstream/123456789/53687/1/Susiekov_magistr.pdf	5 Sources	0.37%
83	http://cad.kpi.ua/attachments/093_%D0%94%D0%B8%D0%BF%D0%BB%D0%BE%D0%BC_%D0%9F%D0%B8%D1%81%D1%8C...		0.36%
85	https://ela.kpi.ua/bitstream/123456789/59077/1/Yermosina_bakalavr.pdf		0.35%
87	https://ela.kpi.ua/bitstream/123456789/52615/1/Yurchenko_magistr.pdf	2 Sources	0.35%
90	https://ela.kpi.ua/bitstream/123456789/59129/1/Volynets_bakalavr.pdf		0.33%
91	https://ela.kpi.ua/bitstream/123456789/56622/1/Polupan_mahistr.pdf		0.34%
97	https://ela.kpi.ua/handle/123456789/59084?mode=full		0.31%
99	https://ela.kpi.ua/bitstream/123456789/51712/1/Radchuk_magistr.pdf		0.3%
100	https://ela.kpi.ua/bitstream/123456789/56994/1/Fedko_magistr.pdf	2 Sources	0.31%
101	https://ela.kpi.ua/handle/123456789/31997		0.3%
104	https://ua-referat.com/uploaded/nacionalenij-tehnichnij-universitet-ukrayini-kiyivsekij-polite-vkyw1/index1.html		0.28%
109	https://ela.kpi.ua/handle/123456789/59082?mode=full	3 Sources	0.27%
110	https://ela.kpi.ua/handle/123456789/31864	3 Sources	0.27%
116	https://ela.kpi.ua/bitstream/123456789/57109/1/magistr_Kalachnykov_Oleksandr.pdf	2 Sources	0.26%
120	https://ela.kpi.ua/bitstream/123456789/27205/1/Kolot_magistr.pdf		0.25%
122	https://ela.kpi.ua/bitstream/123456789/31803/1/Shkoliar_magistr.pdf	7 Sources	0.24%
123	https://ela.kpi.ua/handle/123456789/32127		0.24%
125	https://ela.kpi.ua/bitstream/123456789/39293/1/Dorosh_magistr.pdf		0.23%
126	https://ela.kpi.ua/bitstream/123456789/26909/6/Masechko_magistr.docx		0.23%
127	https://ela.kpi.ua/handle/123456789/25800		0.23%
128	https://tef.kpi.ua/files/pdf/1tezi_tom2_1524728034.pdf		0.23%
131	https://ela.kpi.ua/bitstream/123456789/52690/6/Huz_magistr.pdf		0.22%

132	https://ela.kpi.ua/bitstream/123456789/26984/8/Kovalchuk_magistr.pdf		0.22%
133	https://ela.kpi.ua/handle/123456789/54530	2 Sources	0.22%
138	http://tk-its.kpi.ua/sites/default/files/2019-03/Voloshyn_magistr.pdf		0.21%
139	https://ela.kpi.ua/bitstream/123456789/32305/1/Pavlovskii_magistr.pdf	2 Sources	0.2%
152	https://ela.kpi.ua/bitstream/123456789/25668/1/Bakharev_magistr.pdf		0.2%
153	https://ela.kpi.ua/handle/123456789/44196		0.2%
155	https://ela.kpi.ua/bitstream/123456789/41204/1/Shostak_magistr.pdf		0.2%
156	https://ela.kpi.ua/bitstream/123456789/25831/1/Korniiachenko_magistr.pdf		0.19%
157	https://ela.kpi.ua/bitstream/123456789/51868/1/Kotlov_magistr.pdf		0.2%
158	https://ela.kpi.ua/handle/123456789/25553		0.19%
162	https://ela.kpi.ua/handle/123456789/51758		0.19%
163	https://ela.kpi.ua/handle/123456789/33741		0.18%
164	https://osvita.kpi.ua/sites/default/files/opfiles/121_OPPB_IPZIKFSVT_2021.pdf	3 Sources	0.16%
165	http://uchika.in.ua/rozvitok-mehanizmu-obslugovuvannya-bankom-korporativnih-kliyen.html		0.16%
166	https://ela.kpi.ua/bitstream/123456789/52556/1/Laskaviy_magistr.pdf		0.16%
167	https://ela.kpi.ua/bitstream/123456789/31878/1/Nerodenko_magistr.pdf		0.15%
168	https://ela.kpi.ua/handle/123456789/26921	14 Sources	0.16%
169	https://ela.kpi.ua/handle/123456789/29727	2 Sources	0.16%
170	https://ela.kpi.ua/bitstream/123456789/36439/1/Pavlenko_bakalavr.pdf		0.15%
175	https://ela.kpi.ua/bitstream/123456789/38256/1/Zatulenko_magistr.pdf		0.15%
176	https://ela.kpi.ua/handle/123456789/27355		0.15%
177	https://ela.kpi.ua/bitstream/123456789/25822/1/Mishchuk_magistr.pdf		0.15%

179	http://cad.kpi.ua/attachments/093__Master_theses_Kovtun-DA-32m.pdf		0.15%
180	https://ela.kpi.ua/handle/123456789/46078		0.14%
181	https://ela.kpi.ua/handle/123456789/51703	2 Sources	0.15%
182	https://ela.kpi.ua/handle/123456789/51712		0.14%
183	https://ela.kpi.ua/bitstream/123456789/26988/8/Lysenko_magistr.pdf		0.15%
184	https://uchika.in.ua/nacionalenij-tehnichnij-universitet-ukrayini-kiyivsekij-polite-v15.html		0.15%
185	https://ela.kpi.ua/bitstream/123456789/45782/1/Alokhin_magistr.pdf		0.14%
186	http://eprints.library.odeku.edu.ua/5010/1/ZosimchukJV_KEOD_MKR_2018.pdf		0.14%
190	https://ela.kpi.ua/bitstream/123456789/52626/1/Mariukhin_magistr.pdf		0.13%
193	http://lib.iitta.gov.ua/716724/1/tmn-2015-3.pdf	22 Sources	0.14%
197	https://ela.kpi.ua/handle/123456789/23795		0.13%
198	https://ela.kpi.ua/bitstream/123456789/32106/1/Zavistovska_magistr.pdf	2 Sources	0.13%
199	https://ela.kpi.ua/handle/123456789/41624	3 Sources	0.12%
200	http://ekmair.ukma.edu.ua/bitstream/handle/123456789/17750/Naumenko_Analiz_ta_modeliuvannia_dii_transmisiinoho_mek...		0.13%
202	https://ela.kpi.ua/handle/123456789/49664	2 Sources	0.13%
210	https://ela.kpi.ua/bitstream/123456789/52156/3/Kizym_magistr.pdf		0.13%
213	https://ir.library.knu.ua/server/api/core/bitstreams/6ab89b19-a935-4f1a-9982-aaa013b41dab/content		0.11%
214	http://ir.nmu.org.ua/handle/123456789/154347		0.1%
223	https://ela.kpi.ua/handle/123456789/30607		0.09%
224	https://ela.kpi.ua/bitstream/123456789/35022/1/Kovalenko-V-O_bakalavr.pdf		0.09%
225	https://ela.kpi.ua/handle/123456789/27113?mode=full		0.09%
226	http://elibrary.donnuet.edu.ua/1940/1/Shtyk_article_3.pdf		0.09%

231	https://dspace.lboro.ac.uk/2134/13173	0.08%
234	https://core.ac.uk/download/pdf/223009996.pdf	0.08%
235	https://journal.ie.asm.md/en/contents/electronni-jurnal-456-2022	0.08%
236	https://ela.kpi.ua/bitstream/123456789/58893/1/Kalaida_bakalavr.pdf	0.08%

Library sources 217

1	**Student submission**	File ID: **1015745799**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		2.18%
2	**Student submission**	File ID: **1015742130**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		1.61%
3	**Student submission**	File ID: **1015745067**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		1.6%
4	**Student submission**	File ID: **1015745796**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		1.27%
8	**Student submission**	File ID: **1015732324**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic**	7 Sources	1.07%
9	**Student submission**	File ID: **1015743338**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		1.07%
10	**Student submission**	File ID: **1015745797**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		1.03%
14	**Student submission**	File ID: **1012991049**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic**	2 Sources	0.86%
21	**Student submission**	File ID: **1015077465**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic**	2 Sources	0.74%
25	**Student submission**	File ID: **1011525299**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.73%
26	**Student submission**	File ID: **1015059530**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic**	2 Sources	0.73%
27	**Student submission**	File ID: **1013095581**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.71%
30	**Student submission**	File ID: **1013028806**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.69%
31	**Student submission**	File ID: **1013090177**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.67%
32	**Student submission**	File ID: **1015036306**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic**	11 Sources	0.66%
33	**Student submission**	File ID: **1000050250**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic**	12 Sources	0.65%
34	**Student submission**	File ID: **1012983247**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.64%

35	Student submission	File ID: **1012943212**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.63%
36	Student submission	File ID: **1012965753**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.64%
37	Student submission	File ID: **1012981673**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.64%
42	Student submission	File ID: **1015018155**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.6%
46	Student submission	File ID: **1013029167**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.6%
47	Student submission	File ID: **1012951220**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.59%
48	Student submission	File ID: **1014982740**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.59%
49	Student submission	File ID: **1012965746**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.59%
50	Student submission	File ID: **1013055092**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.58%
52	Student submission	File ID: **1000047857**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	6 Sources	0.57%
53	Student submission	File ID: **1000035014**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.56%
54	Student submission	File ID: **1000046603**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.56%
58	Student submission	File ID: **1009476398**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.52%
62	Student submission	File ID: **1000039717**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.48%
68	Student submission	File ID: **1009491583**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	3 Sources	0.44%
70	Student submission	File ID: **1000755753**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.42%
73	Student submission	File ID: **1015731195**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.38%
75	Student submission	File ID: **1009480507**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	3 Sources	0.38%
77	Student submission	File ID: **1000745187**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	3 Sources	0.38%
79	Student submission	File ID: **1009433921**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.35%
80	Student submission	File ID: **1015700165**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.37%
81	Student submission	File ID: **1009503380**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	3 Sources	0.36%

84	Student submission	File ID: **1005704345**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.36%
86	Student submission	File ID: **1009495857**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.35%
88	Student submission	File ID: **1009504569**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.35%
89	Student submission	File ID: **1012942227**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.35%
92	Student submission	File ID: **1009496065**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.33%
93	Student submission	File ID: **1015628614**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.31%
94	Student submission	File ID: **1000749058**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.33%
95	Student submission	File ID: **1000792549**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.32%
96	Student submission	File ID: **1000723812**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.32%
98	Student submission	File ID: **1009545124**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.31%
102	Student submission	File ID: **1015698377**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	5 Sources	0.28%
103	Student submission	File ID: **1015226154**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.27%
105	Student submission	File ID: **1005766349**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.27%
106	Student submission	File ID: **1003223780**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.27%
107	Student submission	File ID: **1005715975**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.27%
108	Student submission	File ID: **1000807162**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.26%
111	Student submission	File ID: **1000745711**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	4 Sources	0.27%
112	Student submission	File ID: **1000758419**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.26%
113	Student submission	File ID: **1009496565**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.26%
114	Student submission	File ID: **1000733827**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.26%
115	Student submission	File ID: **1000754634**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.26%
117	Student submission	File ID: **1000756805**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.26%

118	Student submission	File ID: **1000748496**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.25%
119	Student submission	File ID: **1000747888**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.26%
121	Student submission	File ID: **1005707052**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.24%
124	Student submission	File ID: **1005784727**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.23%
129	Student submission	File ID: **1003312065**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.22%
130	Student submission	File ID: **1009469123**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.23%
134	Student submission	File ID: **1015344789**	Institution: **Lviv Polytechnic National University**		0.22%
135	Student submission	File ID: **1003344595**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.22%
136	Student submission	File ID: **1009419960**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.21%
137	Student submission	File ID: **1003289815**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.21%
140	Student submission	File ID: **1005754245**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	3 Sources	0.21%
141	Student submission	File ID: **1003053908**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.21%
142	Student submission	File ID: **1009652403**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.21%
143	Student submission	File ID: **1009117025**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.21%
144	Student submission	File ID: **1015707369**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.21%
145	Student submission	File ID: **1000012757**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.21%
146	Student submission	File ID: **1009683571**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.2%
147	Student submission	File ID: **1009698001**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.21%
148	Student submission	File ID: **1011340965**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.21%
149	Student submission	File ID: **1011343797**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.21%
150	Student submission	File ID: **1015679570**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.21%
151	Student submission	File ID: **1009497506**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.13%

154	Student submission	File ID: **1000066703**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.2%
159	Student submission	File ID: **1000787996**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.19%
160	Student submission	File ID: **1009421956**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.19%
161	Student submission	File ID: **1000050719**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.19%
171	Student submission	File ID: **1005743321**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	3 Sources	0.15%
172	Student submission	File ID: **1015079598**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.15%
173	Student submission	File ID: **1012883219**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.15%
174	Student submission	File ID: **1007947745**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.15%
178	Student submission	File ID: **1015134675**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.15%
187	Student submission	File ID: **1009467260**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.14%
188	Student submission	File ID: **1003994987**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic**	15 Sources	0.14%
189	Student submission	File ID: **1015149669**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.14%
191	Student submission	File ID: **1005672566**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.14%
192	Student submission	File ID: **1009693978**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.14%
194	Student submission	File ID: **1000044158**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.14%
195	Student submission	File ID: **1009731863**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.13%
196	Student submission	File ID: **1009712565**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.13%
201	Student submission	File ID: **1000070462**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.13%
203	Student submission	File ID: **1000062912**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.13%
204	Student submission	File ID: **1005679856**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.13%
205	Student submission	File ID: **1008273301**	Institution: **National Aviation University**		0.13%
206	Student submission	File ID: **1008296552**	Institution: **National Aviation University**		0.13%

207	Student submission	File ID: **1005679866**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.13%
208	Student submission	File ID: **1005656131**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	2 Sources	0.13%
209	Student submission	File ID: **1009503370**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.12%
211	Student submission	File ID: **1009734766**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.13%
212	Student submission	File ID: **1015075515**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.13%
215	Student submission	File ID: **1014835466**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.1%
216	Student submission	File ID: **1003701143**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.1%
217	Student submission	File ID: **1000780893**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.1%
218	Student submission	File ID: **1000083559**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic I**	6 Sources	0.1%
219	Student submission	File ID: **1014833091**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.1%
220	Student submission	File ID: **1015293436**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.09%
221	Student submission	File ID: **1015169115**	Institution: **National Aviation University**		0.09%
222	Student submission	File ID: **1005982434**	Institution: **State University Kyiv National Economic University Vadym Hetman**		0.09%
227	Student submission	File ID: **5474533**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Inst**	7 Sources	0.08%
228	Student submission	File ID: **1015682331**	Institution: **Zaporizhzhya National University**	2 Sources	0.08%
229	Student submission	File ID: **1015242942**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.08%
230	Student submission	File ID: **1015723725**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.08%
232	Student submission	File ID: **1005736192**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institu...**		0.08%
233	Student submission	File ID: **1015081158**	Institution: **National University of Life and Environmental Sciences of Ukraine**		0.08%

Quotes

Quotes **5**

1 Infrastructure as a Service (IaaS) in Cloud Computing.

2 What is PaaS (Platform as a Service) in Cloud Computing?

3 **« Система вибору вибіркових дисциплін в рамках кафедри та інтеграція її у університетську систему**. Back End **розробка бізнес-логіки додатку, підходи до розширення функціоналу існуючої системи »**

4 **«Система вибору вибіркових дисциплін в рамках кафедри та інтеграція її у університетську систему**. Back End **розробка бізнес-логіки додатку, підходи до розширення функціоналу існуючої системи»**

5 "15"». **В результаті цих дій можна отримати функціонуючу БД** (**рис**. 3.16). **Рисунок** 3.16 – **Успішно розгорнута за допомогою оператора** PostgreSQL **БД Після цього можна почати розгортання мікросервісів. За допомогою інструменту** kubectl, YAML **конфігураційні файли були застосовані для розгортання мікросервісів в середовищі** GKE **кластеру** (**рис** 3.17). **Рисунок** 3.17 – **Успішно розгорнутий фронтенд мікросервіс Окрім того, буде потрібна інфраструктура для подальших дій. Для цього необхідно встановлення** cert-manager. **Спочатку необхідно взяти основні файли з офіційного репозиторію** (**рис**. 3.18), **та розгонути їх за допомогою** kubectl (**рис** 3.19). **Рисунок** 3.18 – **Файли, необхідні для розгортки** cert-manager 77 **Рисунок** 3.19 – **Успішно розгорнутий** cert-manager **Завдяки цим діям, сервіс буде успішно розгорнуто в** Kubernetes **середовищі, що готове до масштабування та подальшого розвитку**. 3.4 Injector service'**у та** webhook **Для розширення можливостей сервісу, підвищенню його надійності, необхідно створити** webhook. **Почати це варто з створення кодової бази для нього, зроблено це буде за допомогою мови програмування** golang **та середовища** Goland IDE (**рис**. 3.20). 78 **Рисунок** 3.20 – **Структура проекту** mutating webhook**'а на** golang **Цей** Webhook **буде служити для перехоплення запитів до** Kubernetes API **та модифікації об'єктів перед їхнім створенням чи оновленням. Для реалізації** webhook'**у були використані такі необхідні бібліотеки як, наприклад,** k8s.io. **Для забезпечення безпечного з'єднання між** Kubernetes API **сервером та** webhook'**ом, та нормальної роботи додатку було згенеровано** TLS **сертифікати** (tls.key **та** tls.crt) **за допомогою спеціалізованого скрипта. Роботу цього скрипта можна коротко описати наступними кроками**: 1. **Параметри** APP **та** NAMESPACE **беруться як аргументи командного рядка і використовуються для генерації імені** CSR (Certificate Signing Request). 2. **За допомогою** openssl **генерується приватний ключ** (${APP}.key). 79 3. **Створюється** CSR (${APP}.csr) **з конфігураційним файлом, який визначає атрибути сертифіката**. 4. **Створюється новий об'єкт** CertificateSigningRequest **в** Kubernetes **з генерованим** CSR. 5. **Скрипт очікує, поки** CSR **буде присутнім в** Kubernetes **та затверджений**. 6. **За допомогою** kubectl **отримується затверджений сертифікат**. 7. **Затверджений сертифікат декодується з** base64 **і зберігається у файл** $ {APP}.pem. **Після використання цього скрипта, було згенеровано ряд файлів, серед яких є потрібні нам сертифікати** (**рис**. 3.21). **Рисунок** 3.21 – **Файли, що було згенеровано скриптом Результуючі файли варто завантажити у** Kubernetes (**рис**. 3.22). **Рисунок** 3.22 – **Створений секрет з сертифікатами І тепер, коли вся необхідна інфраструктура готова, можна почати створення конфігураційних файлів для** webhook. **Для реєстрації та коректної роботи** webhook'**у в** Kubernetes **середовищі було створено додаткові** YAML **конфігураційні файли. Один з найважливіших серед них** – MutatingWebhookConfiguration: **цей** 80 **файл визначає налаштування** mutating webhook'**у та вказує, які об'єкти та події мають бути перехоплені. Одними з його головних параметрів можна виділити: унікальне ім'я конфігурації** webhook'**у, наприклад «**

References **1**

1 1. **Опитування щодо якості навчання Магістри Аналітика**. URL: https://ipze.kpi.ua/wp-content/uploads/2023/10/**Опитування_щодо_якості_навчання_Магістри_Аналітика**_ 1.pdf. (**дата звернення**: 02.01.2023). 2. Ellucian Banner. URL: https://www.ellucian.com/solutions/ellucian-banner. (**дата звернення**: 02.01.2023). 3. Student Information System. URL: https://www.blackbaud.com/solutions/organizational-and-program- management/student-information. (**дата звернення**: 02.01.2023). 4. KUALI STUDENT. URL: https://www.kuali.co/products/student. (**дата звернення**: 02.01.2023). 5. PeopleSoft Campus Solutions. URL: https://docs.oracle.com/en/applications/peoplesoft/campus-solutions/index.html. (**дата звернення**: 02.01.2023). 6. The Smart Choice for Higher Education. URL: https://jenzabar.com/about. (**дата звернення**: 02.01.2023). 7. CampusNexus Student Features. URL: https://www.campusmanagement.com/products/campusnexus-student-features/. (**дата звернення**: 02.01.2023). 8. A Comprehensive Guide For On-premise vs Cloud Computing Tutorial. URL: https://www.edureka.co/blog/on-premise-vs-cloud-computing/. (**дата звернення**: 02.01.2023). 9. Infrastructure as a Service (IaaS) in Cloud Computing. URL: https://www.mongodb.com/cloud-explained/iaas-infrastructure-as-a-service. (**дата звернення**: 02.01.2023). 98 10. What is PaaS (Platform as a Service) in Cloud Computing? URL: https://www.simplilearn.com/what-is-paas-article. (**дата звернення**: 02.01.2023). 11. Turnbull J. The Docker Book. Lulu.com, 2023. 386 **С**. 12. Gkatziouras E. A Developer's Essential Guide to Docker Compose: Simplify the development and orchestration of multi-container applications. Packt Publishing Ltd; 2022. **С**. 264. 13. Poulton N. The Kubernetes Book. NIGEL POULTON LTD; 2023. **С**. 311. 14. Using Helm. URL: https://helm.sh/docs/intro/using_helm/. (**дата звернення**: 02.01.2023). 15. Collier M, Shahan R. Microsoft Azure Essentials - Fundamentals of Azure. Microsoft Press; 2015. **С**. 246. 16. The most scalable and fully automated Kubernetes service. URL: https://cloud.google.com/kubernetes-engine?hl=en. (**дата звернення**: 02.01.2023). 17. Burnett SM. Aws for Beginners: The Ultimate Guide to the Fundamentals of Aws. Independently Published; 2021. **С**. 118. 18. Dynamic Admission Control. URL: https://kubernetes.io/docs/reference/access- authn-authz/extensible-admission-controllers/. (**дата звернення**: 02.01.2023). 19. Writing a very basic kubernetes mutating admission webhook. URL: https://medium.com/ovni/writing-a-very-basic-kubernetes-mutating-admission- webhook-398dbbcb63ec. (**дата звернення**: 02.01.2023). 20. Sidecar Containers. URL: https://kubernetes.io/docs/concepts/workloads/pods/sidecar-containers/. (**дата звернення**: 02.01.2023). 21. Operator pattern. URL: https://kubernetes.io/docs/concepts/extend- kubernetes/operator/. (**дата звернення**: 02.01.2023). 22. Gateway API. URL: https://kubernetes.io/docs/concepts/services- networking/gateway/. (**дата звернення**: 02.01.2023). 99 23. GoLang — The Good, the Bad and the Ugly. URL: https://medium.com/geekculture/golang-the-good-the-bad-and-the-ugly-880270a85848. (**дата звернення**: 02.01.2023). 24. Overview of Java. URL: https://docs.oracle.com/en/database/oracle/oracle-database/19/jjdev/Java-overview.html. (**дата звернення**: 02.01.2023). 25. What is Python Language? Overview of the Python Language. URL: https://www.scholarhat.com/tutorial/python/overview-of-the-python-language. (**дата звернення**: 02.01.2023). 26. Introduction to Node.js. URL: https://nodejs.org/en/learn/getting- started/introduction-to-nodejs. (**дата звернення**: 02.01.2023). 27. Education ERP Market Forecast & Report Summary, Global Size. URL: https://www.marketsandmarkets.com/Market-Reports/education-erp-market- 225190725.html. (**дата звернення**: 02.01.2023). 28. How to measure the ROI of tech in higher education. URL: https://www.fullfabric.com/articles/how-to-measure-the-roi-of-tech-in-higher-education. (**дата звернення**: 02.01.2023). 100

Exclusions

Internet exclusions 77

https://ela.kpi.ua/bitstream/123456789/44246/1/Valchun_bakalavr.pdf		0.07%
https://ela.kpi.ua/handle/123456789/45782		0.07%
http://eadnurt.diit.edu.ua/bitstream/123456789/12812/1/Davydenko_dyp_2020.pdf		0.07%
http://conf.fizmat.tnpu.edu.ua/media/arhive/25.11.22_%D0%97%D0%B1%D1%96%D1%80%D0%BD%D0%B8%D0%BA_%D1%82%D...		0.07%
http://tst.stu.cn.ua/issue/download/14505/7895		0.07%
http://hdl.handle.net/2134/32268		0.07%
https://ela.kpi.ua/jspui/bitstream/123456789/29896/1/Kovalenko_bakalavr.pdf	3 Sources	0.07%
https://ela.kpi.ua/handle/123456789/28916	5 Sources	0.07%
https://ela.kpi.ua/jspui/bitstream/123456789/30438/1/Adakh_magistr.pdf	3 Sources	0.07%
https://ela.kpi.ua/bitstream/123456789/27009/14/Marchyshyna_magistr.pdf		0.07%
https://ela.kpi.ua/bitstream/123456789/52542/1/Chornenkiy_magistr.pdf		0.07%
https://ela.kpi.ua/bitstream/123456789/26973/8/ZubarchukOD_magistr.pdf	2 Sources	0.06%
https://elartu.tntu.edu.ua/bitstream/lib/39650/1/SAm-61_Bielousov_K_K.pdf	8 Sources	0.06%
https://ela.kpi.ua/handle/123456789/31258		0.06%
https://ela.kpi.ua/bitstream/123456789/41693/1/Kosach_magistr.pdf		0.06%
http://lib.iitta.gov.ua/107152/1/%D0%93%D0%BB%D0%BE%D1%81%D0%B0%D1%80i%D0%B9_%D0%BD%D0%B0_%D1%81%	3 Sources	0.06%
https://er.nau.edu.ua/bitstream/NAU/52032/1/%d0%a4%d0%9c%d0%92%20293%20%d0%91%d0%b5%d1%80%d0%b5%d1%81%...		0.06%
http://repository.kpi.kharkov.ua/handle/KhPI-Press/21048	3 Sources	0.06%
https://elartu.tntu.edu.ua/handle/lib/31373?mode=full		0.06%
https://uchika.in.ua/senchishak-l-v-metodist-nmc-osnovi-naukovo-doslidnicekoyi-diya.html	8 Sources	0.06%

http://ir.nmu.org.ua/bitstream/handle/123456789/160206/%d0%9f%d0%97_%d0%9a%d0%be%d0%bc%d0%b0%d1%80%d0%b8%...		0.06%
http://www1.cnri.reston.va.us	4 Sources	0.06%
http://ekhsuir.kspu.edu/bitstream/handle/123456789/18214/122_PraskoAV_fknfm_2023.pdf?isAllowed=y&sequence=1	7 Sources	0.06%
https://ela.kpi.ua/bitstream/123456789/31416/1/Korzhenevskyi_magistr.pdf	2 Sources	0.06%
https://openarchive.nure.ua/bitstream/document/15083/1/2020_M_INF_Petuhova_KS.pdf		0.06%
https://ela.kpi.ua/bitstream/123456789/23045/3/Kravinskyi_magistr.pdf	2 Sources	0.06%
http://geum.ru/next/art-116542.php	8 Sources	0.06%
https://ela.kpi.ua/bitstream/123456789/56982/1/Kostiuchyk_magistr.pdf		0.06%
https://blog.mate.academy/python/learning-python-2023		0.06%
https://ela.kpi.ua/bitstream/123456789/26171/3/Dokashenko_magistr.pdf	3 Sources	0.06%

Library exclusions **156**

Student submission	File ID: **7647889**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institute"**		0.07%
Student submission	File ID: **1003704671**	Institution: **National University of Life and Environmental Sciences of Ukrai**	3 Sources	0.07%
Student submission	File ID: **1013100024**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	14 Sources	0.07%
Student submission	File ID: **1003891492**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	5 Sources	0.07%
Student submission	File ID: **1000086949**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	2 Sources	0.07%
Student submission	File ID: **1015720878**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institute"**		0.07%
Student submission	File ID: **8237602**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institute"**		0.07%
Student submission	File ID: **1008101121**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	3 Sources	0.07%
Student submission	File ID: **1015149978**	Institution: **Lviv Polytechnic National University**		0.07%
Student submission	File ID: **1015664882**	Institution: **Lviv Polytechnic National University**		0.07%
Student submission	File ID: **1015226156**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institute"**		0.07%

Student submission	File ID: **1014744613**	Institution: **Taras Shevchenko National University of Kyiv**	4 Sources	0.07%
Student submission	File ID: **1015307544**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	3 Sources	0.07%
Student submission	File ID: **1015273268**	Institution: **Poltava National Technical Yuri Kondratyuk University**		0.06%
Student submission	File ID: **5692629**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institute**	2 Sources	0.06%
Student submission	File ID: **1015311871**	Institution: **Lviv Polytechnic National University**	3 Sources	0.06%
Student submission	File ID: **1005699861**	Institution: **National University of Water Management and Natural Resour**	10 Sources	0.06%
Student submission	File ID: **1011458452**	Institution: **Lviv Polytechnic National University**	9 Sources	0.06%
Student submission	File ID: **1015342708**	Institution: **Taras Shevchenko National University of Kyiv**	9 Sources	0.06%
Student submission	File ID: **1003120987**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	2 Sources	0.06%
Student submission	File ID: **1015227481**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	10 Sources	0.06%
Student submission	File ID: **1015336425**	Institution: **Lviv Polytechnic National University**	4 Sources	0.06%
Student submission	File ID: **1005754328**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institute"**		0.06%
Student submission	File ID: **1005676665**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institute"**		0.06%
Student submission	File ID: **1015482783**	Institution: **National Aviation University**	2 Sources	0.06%
Student submission	File ID: **1011461674**	Institution: **Yuriy Fedkovych Chernivtsi National University**	2 Sources	0.06%
Student submission	File ID: **1015226603**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	2 Sources	0.06%
Student submission	File ID: **1000756062**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institute"**		0.06%
Student submission	File ID: **1004279799**	Institution: **Lviv Polytechnic National University**	2 Sources	0.06%
Student submission	File ID: **1015605469**	Institution: **Lviv Polytechnic National University**		0.06%
Student submission	File ID: **2005872**	Institution: **National University Ostroh Academy**	2 Sources	0.06%
Student submission	File ID: **1015566597**	Institution: **National University of Life and Environmental Sciences of Ukraine**		0.06%
Student submission	File ID: **8306024**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institute"**		0.06%

Student submission	File ID: **1015226843**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	2 Sources	0.06%
Student submission	File ID: **1005710415**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	3 Sources	0.06%
Student submission	File ID: **1009505202**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institute"**		0.06%
Student submission	File ID: **1009609480**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	9 Sources	0.06%
Student submission	File ID: **1015254518**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	6 Sources	0.06%
Student submission	File ID: **1015698656**	Institution: **Lviv Polytechnic National University**	3 Sources	0.06%
Student submission	File ID: **1011397212**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Institute"**		0.06%
Student submission	File ID: **1007212286**	Institution: **Interregional Academy of Personnel Management**	5 Sources	0.06%
Student submission	File ID: **1007887560**	Institution: **Taras Shevchenko National University of Kyiv**	8 Sources	0.06%
Student submission	File ID: **1008320983**	Institution: **National Aviation University**	2 Sources	0.06%
Student submission	File ID: **8571276**	Institution: **Sumy State University**		0.06%
Student submission	File ID: **1015093358**	Institution: **National Technical University of Ukraine "Kyiv Polytechnic Insti**	8 Sources	0.06%
Student submission	File ID: **1015690973**	Institution: **Lviv Polytechnic National University**		0.06%