МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ В. Н. КАРАЗІНА

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ КОМП’ЮТЕРНОЇ ФІЗИКИ ТА ЕНЕРГЕТИКИ

Кафедра комп`ютерної фізики

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломної роботи

за освітньо-кваліфікаційним рівнем магістр

«Програмне забезпечення для автоматизації та оптимізації процесів в енергетиці»

Виконав: студент 2 курсу, групи НФ-64

Анфімов О. К.

спеціальність:

105 Прикладна фізика та наноматеріали

освітня програма: комп'ютерна фізика

Керівник: к.т.н. Гарячевська І.В.,

Рецензент:

Харків – 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет імені В.Н. Каразіна

Навчально-науковий інститут комп’ютерної фізики та енергетики

Кафедра комп’ютерної фізики

Рівень вищої освіти (освітньо-кваліфікаційний рівень) магістр

Спеціальність: 105 Прикладна фізика та наноматеріали

Освітня програма: Комп’ютерна фізика

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ проф. Нємченко К.Е.

підпис

“____” _____2021 року

З А В Д А Н Н Я

НА ДИПЛОМНУ РОБОТУ МАГІСТРА

Анфімову Олегу Костянтиновичу

1. Тема роботи «Програмне забезпечення для автоматизації та оптимізації процесів в енергетиці»

керівник роботи Гарячевська Ірина Василівна, кандидат технічних наук,

затверджені наказом по університету від 10 лютого 2021 № 0210 – 05/374

2. Строк подання студентом роботи 12 травня 2021 р.

3. Перелік питань, які потрібно розглянути:

1) Провести аналіз існуючих підходів до проектування автоматизованих систем в енергетиці.
2) Аналіз пристроїв які використовуються на підприємствах.
3) Розробка моделі автоматизованої системи, конфігурація її і інтеграція з IoT пристроями.
4) Розробка алгоритму, структур даних, а також архітектури програми.

5) Розробка та проектування бази даних.

6) Розробка і тестування системи.

4. План роботи

№ з/п	Назви етапів роботи
1	Вибір теми, узгодження, закріплення за кафедрою
2	Робота над літературою та іншими джерелами
3	Написання вступу до роботи та першого розділу
4	Написання другого розділу
5	Написання третього розділу
6	Завершення написання дипломної роботи (висновки та література)
7	Подача науковому керівнику, доробка зауважень
8	Відгук, рецензія, підготовка до захисту
9	Попередній захист на кафедрі
10	Захист

5. Дата видачі завдання « »_____ 2021 р.

Студент _____ Анфімов О.К.

підпис

Керівник роботи _____ Гарячевська І.В.

підпис

РЕФЕРАТ

Дипломна робота магістра виконана студентом Анфімовим Олегом Костянтиновичем на тему «Програмне забезпечення для автоматизації та оптимізації процесів в енергетиці».

Пояснювальна записка містить 45 аркуш основного тексту, 14 рисунків, 41 джерел використаної літератури.

Об'єктом дослідження дипломної роботи магістра є розробка програмного для автоматизації та оптимізації процесів в енергетиці. Дослідження проблем централізованих мереж. Основним результатом роботи є розроблені сервіси для автоматизації, з використанням мікро-сервісної архітектури, Kubernetes інфраструктура для обслуговування та Circle CI/CD для автоматизації керування процесів розробки та перевірки. Використано новий інноваційний підхід до роботи з функціональними FaaS-рішеннями. Досягнуто мети універсальної автоматизації з центральним регулюванням.

КЛЮЧОВІ СЛОВА: БАЗА ДАНИХ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, АВТОМАТИЗАЦІЯ, ТАБЛИЦЯ, NUCLIO, KONG GATEWAY, PROFILE, GOLANG, SQL, NOSQL, POSTGRESQL, DOCKER, KUBERNETES, LINUX, BASH, RAFT, IOT, MICROSERVICES, DISASTER RECOVERING, FAAS, SAAS.

РЕФЕРАТ

Дипломная работа магистра выполнена студентом Анфимов Олегом Константиновичем на тему «Программное обеспечение для автоматизации и оптимизации процессов в энергетике».

Пояснительная записка содержит 45 лист основного текста, 14 рисунков, 41 источников литературы.

Объектом исследования дипломной работы магистра является разработка программного для автоматизации и оптимизации процессов в энергетике. Исследование проблем централизованных сетей. Основным результатом работы являются разработанные сервисы для автоматизации с использованием микро-сервисной архитектуры, Kubernetes инфраструктура для обслуживания и Circle CI / CD для автоматизации управления процессов разработки и проверки. Использован новый инновационный подход к работе с функциональными FaaS-решениями. Достигнуты цели универсальной автоматизации с центральной регулировкой.

КЛЮЧЕВЫЕ СЛОВА: БАЗА ДАННЫХ, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, АВТОМАТИЗАЦИЯ, ТАБЛИЦА, NUCLIO, KONG GATEWAY, PROFILE, GOLANG, SQL, NOSQL, POSTGRESQL, DOCKER, KUBERNETES, LINUX, BASH, RAFT, IOT, MICROSERVICES, DISASTER RECOVERING, FAAS, SAAS.

ABSTRACT

MSc Dissertation is done by the student Oleh Anfimov on the theme «Software for automation and optimization of processes in energy».

Explanatory note contains 45 pages of the main text, 14 figures, 41 sources of used literature.

The object of the study of MSc Dissertation is the development of software for automation and optimization of processes in energy. Investigate problems with centralized networks. The main result of the work is developed services for automation, using microservice architecture, Kubernetes infrastructure for maintenance and Circle CI / CD for automation of control of development and verification processes. A new innovative approach to working with functional FaaS-solutions is used. The goal of universal automation with central regulation has been achieved.

KEYWORDS: DATABASE, SOFTWARE, AUTOMATION, TABLE, NUCLIO, KONG GATEWAY, PROFILE, GOLANG, SQL, NOSQL, POSTGRESQL, DOCKER, KUBERNETES, LINUX, BASH, RAFT, IOT, MICROSERVICES, DISASTER RECOVERING, FAAS, SAAS.

ЗМІСТ

ВСТУП **7**

1 СИСТЕМИ АВТОМАТИЗАЦІЇ В ЕНЕРГЕТИЦІ ТА ЇХ ПОТОЧНІ ПРОБЛЕМИ **10**

1.1. Системи автоматизації в енергетиці 10

1.2. IoT в енергетиці 13

1.3. Типові реалізації автоматизованих систем в енергетиці та на підприємствах. Існуючі рішення. 16

1.4. Постановка задачі. 18

2 ПРОЄКТУВАННЯ СИСТЕМИ АВТОМАТИЗАЦІЇ **21**

2.1. Вимоги до системи автоматизації 21

2.2. Вибір БД 25

2.3. Проєктування системи автоматизації 28

3 УНІВЕРСАЛЬНА СИСТЕМИ АВТОМАТИЗАЦІЇ В ЕНЕРГЕТИЦІ **35**

ВИСНОВКИ **39**

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 40

ВСТУП

У сучасному світі неможливо уявити організацію, яка б не використовувала новітні інформаційні технології у своїй роботі. Наразі, майже кожне підприємство має частини систем smart-home, різноманітні датчики, свій власний документообіг, систему доступів та менеджменту задач, автоматизовані HR процеси, а також автоматизовані виробничі процеси. Автоматизація процесів використовується більшістю підприємств, які прагнуть використовувати свої ресурси такі, як час та гроші у найбільш ефективний спосіб. Попри те, що автоматизація є досить поширеною на підприємствах, водночас вона вузька та локалізована, і у більшості випадків не може бути масштабована. Питання комплексної автоматизації має велике значення, тому що її впровадження гарантує разючий економічний ефект. Так, комплексна автоматизація процесів на потокових лініях знижує затрати праці в 4–5 разів, і зменшує собівартість на 30–50 % у середньому. Тож, впровадження системи комплексної автоматизації з можливістю інтеграції IoT, Cloud services та severless додатків є ключовим для розвитку підприємств та галузі енергетики в цілому.

Виходячи з порядку денного цілей сталого розвитку ООН, енергоефективність є одним із ключових факторів сталого розвитку. Більше того, енергоефективність надає економічні вигоди в довгостроковій перспективі за рахунок зменшення вартості імпорту / постачання палива, виробництва енергії та зменшення викидів з енергетичного сектору. Для підвищення енергоефективності та більш оптимального управління енергією ключову роль у постачанні енергії відіграє ефективний аналіз даних у реальному часі. У енергопостачанні, а саме від видобутку ресурсів до їх доставки у корисній формі кінцевим споживачам, виділяють три основні частини:

1. Постачання енергії, включаючи процеси переробки нафти;
2. Процеси перетворення енергії, включаючи передачу та розподіл енергоносіїв;
3. Сторона попиту на енергію, що включає використання енергії у будівлях, транспортному секторі та промисловості.

Існує багато різних приладів, технологічних процесів та явищ, які вимагають автоматизації, тому, слід дослідити умови необхідні для їх автоматизації, яким чином можливо оптимізувати роботу підприємств та їх працівників. Інтеграція відновлюваних джерел енергії та оптимізація використання енергії є ключовими чинниками стійких енергетичних переходів та пом'якшення змін клімату. Сучасні технології, такі як Internet of Things (далі - IoT), пропонують широкий спектр застосувань в енергетичному секторі, тобто у постачанні, передачі та розподілі енергії та попиті. IoT може бути використаний для підвищення енергоефективності, збільшення частки відновлюваної енергії та зменшення наслідків використання енергії на навколишнє середовище. Крім того, система автоматизації може включати хмарні обчислення та різні платформи для аналізу даних.

IoT використовує датчики та комунікаційні технології для зондування та передачі даних у реальному часі, що дозволяє швидко обчислювати та приймати оптимальні рішення. Більше того, IoT може допомогти енергетичному сектору перетворитися з централізованої на розподілену, розумну та інтегровану енергетичну систему. Це ключова вимога при впровадженні місцевих розподілених енергетичних систем, таких як енергія вітру та сонячної енергії, а також перетворення багатьох дрібних кінцевих споживачів енергії у споживачів шляхом агрегування їх виробництва та оптимізації їх потреб, коли це корисно для мережі. Системи на основі IoT автоматизують, інтегрують та контролюють процеси за допомогою датчиків та комунікаційних технологій. Великий збір даних та використання інтелектуальних алгоритмів для аналізу даних у режимі реального часу можуть допомогти відстежувати схеми споживання енергії різними користувачами та пристроями в різних часових масштабах та контролювати це споживання більш ефективно.

На сьогоднішній день програмне забезпечення для енергетичних систем знаходиться у початковій стадії розвитку, кожен додаток створений лише для того, щоб автоматизувати роботу окремого компоненту, сенсору, контролера, а не

всю енергетичну систему на підприємствах, будівлях або навіть містах.

1 СИСТЕМИ АВТОМАТИЗАЦІЇ В ЕНЕРГЕТИЦІ ТА ЇХ ПОТОЧНІ ПРОБЛЕМИ

1.1. Системи автоматизації в енергетиці

Автоматизовані системи контролю та обліку енергоресурсів (далі - АСКОЕ) являє собою масштабовану систему програмно-технічних засобів автоматизованого одночасного дистанційного вимірювання, збору, передачі, обробки, відображення і документування кількості споживання енергоресурсів (виробництво,передача, споживання) по множині територіально розподілених точок їх вимірювання у реальному часі з передачею інформаційних даних по ієрархічних рівнях. Тож, на сьогодні застосування систем АСКОЕ набуло широкої популярності в електроенергетиці.

Енергетичні ресурси як об'єкт купівлі-продажу потребують комерційного обліку всіх процесів їх виробництва, передачі та споживання. Саме використання систем автоматизації дозволяє реально отримати точний і швидкий контроль енергоспоживання, й підвищити достовірність обліку, при цьому, оптимізуючі витрати на енергоресурси.

Окрім того, запровадження такої системи дозволить зробити більш якісний аналіз споживання, який сприятиме оперативному прийняттю рішень по оптимізації. Маючи великий обсяг якісних даних та ретельно їх проаналізувавши, підприємство матиме змогу більш точно спрогнозувати енерговитрати.

Застосування систем автоматизації надає змогу виокремити поєднання планування енерговитрат із випуском готової продукції у системі обліку енергоресурсів, виділити собівартість енергетичної складової на кожному етапі виробництва, здійснити аналіз моментів перевантаження/простою обладнання. Розглянувши спрощену схему структурної організації автоматичного обліку енергоспоживання, то типова система автоматизації складається з двох рівнів: внутрішнього рівня, який передає дані споживача на сервер-контролер та зовнішнього рівня, який обробляє і зберігає дані, та дає можливість роботи з ними оператору-користувачу.



Рис. 1.1. Схема роботи автоматизованих систем в енергетиці

В останні роки зростання цін на енергоресурси зумовило різку зміну ставлення до організації енергообліку в промисловості. Тому наявність достовірного обліку комерційного обліку енергоресурсів розглядається як невід'ємний елемент, що сприяє зниженню фінансових втрат та економії енергоресурсів.

У сьогоднішніх умовах ринок енергоресурсів базується на використанні автоматизованого приладового енергообліку, що підвищує ефективність обліку електричної енергії та контролю. Для цього характерна мінімальна участь співробітника на всіх етапах обробки даних, і це у свою чергу забезпечує отримання достовірних, точних, гнучких, оперативних, адаптованих до різного роду тарифних облікових систем постачальника і споживача енергоресурсів. Для досягнення цілі енергозбереження постачальники та споживачі використовують автоматизовані системи контролю. Адже, за допомогою систем автоматизації контролю і обліку енергоресурсів постачальник енергоресурсів або підприємство-споживач задля мінімізації енерговитрат мають можливість всебічно контролювати всі етапи процесу енергоспоживання, і заодно у разі впливу різних економічних або ж інших факторів за згодою іншої сторони використовувати більш вигідну тарифну систему.

Загалом, щоб дійсно ефективно вирішити питання контролю обліку

енергоресурсів потрібно створити систему автоматизації, яка буде двох- або трьохрівнева.

Як показує практика, що зниження цін на мікроелектроніку посприяло зростанню установки різноманітних таймерів, датчиків, та іншого, що є найбільш оптимальним з точки зору економії споживання ресурсів і веде до відчутного зниження витрат. Використання вищезазначених електронних компонентів на базі мікроконтролерів одночасно є гнучким та багатофункціональним, і сприяє точному комерційному обліку. оперативність передачі даних різними каналами зв'язку

Така популярність у використанні цифрових лічильників електроенергії має значну пріоритетність, а саме: точність, якісні елементи, компактність, можливість збільшити термін повірки, різноманітність форм, легкість в технічному обслуговуванні та експлуатації, оперативність передачі даних різними каналами зв'язку у різних ситуаціях.

Так само, важлива можливість керування мікроконтролером цифровим дисплеєм лічильника, що дає змогу програмно налаштовувати режими введення/виведення інформації (спожитої енергії за розрахунковий період та ін.).

На ринку електроніки для автоматизованих систем випускаються лічильники-датчики на основі мікропроцесорів різного типу і призначення: одно і трифазні, однотарифні й багатотарифні, комбіновані інтелектуальні багатофункціональні.

Враховуючи специфіку використання автоматизованих систем, то сучасні високоточні електронні лічильники-датчики спроможні проводити виміри у широкому діапазоні та виконувати тарифні функції. Лічильники що включаються через трансформатори струму і напруги, реєструють енергію в обох напрямках. І за допомогою сервісної програми на ПК усі робочі параметри встановлюються індивідуально

Впровадження таких систем автоматизації є пріоритетним кроком у підвищенні ефективності енергетичного потенціалу України, що дозволяє виробити більш якісний аналіз та приймати найкращі рішення з точки зору

управління.

1.2. IoT в енергетиці

Інтернет речей (далі - IoT) представляє собою нову реальність. Висновки, отримані на основі даних, зібраних з нових підключених до Інтернету пристроїв, можуть бути використані для розробки нових послуг, підвищення продуктивності та ефективності, вдосконалення процесу прийняття рішень у режимі реального часу, вирішення критичних проблем та створення нових та інноваційних можливостей. Однак, коли чим більше пристроїв підключається, компанії стикаються з підвищеною фрагментацією, сумісністю та проблемами безпеки. Щоб вирішити цю проблему, слід використовувати інтегровані, масштабовані апаратні та програмні рішення, спеціально розроблені для задоволення різноманітних потреб ринку, які легко інтегруються в існуючу енергетичну інфраструктуру для підвищення безпеки, надійності та ефективності.

IoT описує мережу фізичних об'єктів - «речей», які вбудовані в датчики, програмне забезпечення та інші технології з метою підключення та обміну даними з іншими пристроями та системами через Інтернет. Ці пристрої варіюються від звичайних побутових предметів до складних промислових інструментів. Маючи сьогодні понад 7 мільярдів підключених пристроїв IoT, експерти очікують, що ця кількість зросте до 22 мільярдів до 2025 року.

Пристрій IoT зазвичай складається з друкованої плати та приєднаними датчиками, які використовують Wi-Fi для підключення до Інтернету. Наприклад:

1. Датчики температури та вологості у кондиціонері;
2. Датчик тиску на віддаленому масляному насосі;
3. Акселерометр у ліфті;
4. Датчики присутності у кімнаті.

Як правило, пристрої IoT передають телеметрію від датчиків до серверних служб у хмарі. Однак, можливі й інші типи зв’язку, наприклад, внутрішня служба, що надсилає команди на ваші пристрої. Нижче наведено кілька прикладів зв'язку

між пристроями та хмарами та хмари з пристроєм:

1. Мобільний рефрижератор відправляє температуру кожні 5 хвилин до IoT Hub;
2. Сервісна служба відправляє на пристрій команду змінити частоту, з якою вона надсилає телеметрію, щоб допомогти діагностувати проблему;
3. Пристрій надсилає попередження на основі значень, зчитування з його датчиків. Наприклад, пристрій для моніторингу періодичного реактора на хімічному заводі надсилає попередження, коли температура перевищує певне значення;
4. Ваші пристрої надсилають інформацію для відображення на інформаційній панелі для перегляду операторами-людьми. Наприклад, диспетчерська на нафтопереробному заводі може показувати температуру, тиск та обсяги потоку в кожній трубі, що дозволяє операторам контролювати установку.

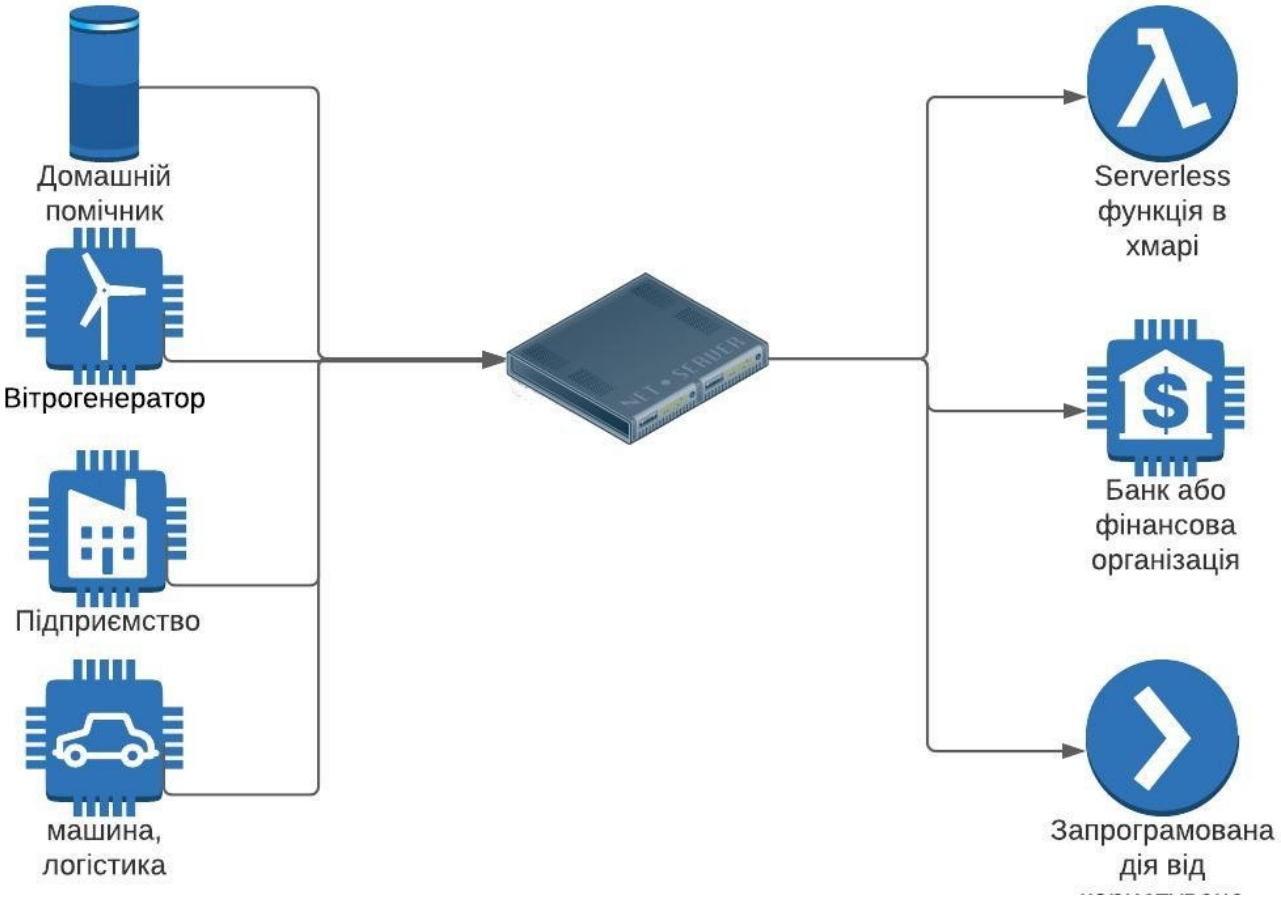


Рис. 1.2. Приклад IoT-мережі

IoT також надає можливість впроваджувати такі рішення, як інтелектуальні шлюзи, які спрощують інтеграцію, знижують витрати на розробку та пришвидшують час виходу на ринок кінцевого продукта або сировини. Від пристрою до центру обробки даних ці рішення включатимуть можливості, що підтримують широкий спектр вертикальних ринків, таких як промисловий контроль та розподілення енергетики, де пристрої працюють в областях з екстремальними вимогами до температури та безпеки.

Для енергетичних компаній це означає більшу гнучкість для розміщення нових джерел енергії, краще управління активами та операціями, більшу надійність, підвищений рівень безпеки, кращий сервіс для споживачів та можливість нових бізнес-моделей та послуг.

Приєднання датчиків IoT до обладнання для виробництва, передачі та розподілу може дозволити енергетичним компаніям дистанційно контролювати

його. Ці датчики вимірюють такі параметри, як вібрація, температура та знос для оптимізації графіків технічного обслуговування. Цей підхід до профілактичного обслуговування може значно покращити надійність, підтримуючи обладнання в оптимальному обладнанні та надаючи можливість робити ремонт до того, як він вийде з ладу.

Цифрова технологія близнюків, яка передбачає створення удосконаленої цифрової моделі існуючого обладнання, може також допомогти в цьому. Датчики IoT, приєднані до фізичного блоку, збирають дані про його роботу, які вони подають цифровому близнюку. На додаток до підтримки програм профілактичного обслуговування, ця технологія забезпечує віртуальне усунення несправностей та підтримку з віддалених місць.

Датчики IoT також можуть допомогти поліпшити безпеку. Прикріплення підключених до Інтернету датчиків до трубопроводів може допомогти виявити витоки, які, якщо їх не вирішити, можуть призвести до пожеж і вибухів. Витоки також становлять марно витрачені ресурси та сприяють глобальному потеплінню.

1.3. Типові реалізації автоматизованих систем в енергетиці та на підприємствах. Існуючі рішення.

Державний стандарт України 2226-93 (далі - ДСТУ 2223-93) встановлює терміни та визначення основних понять у галузі автоматизованих систем (далі - АС) та поширюється на АС, використовувані в різних сферах діяльності людини (керування, дослідження, проектування тощо), змістом яких є перероблення інформації. Стандарт не поширюється на терміни та визначення в галузі АС, призначених для безпосереднього оброблення (виготовлення, зберігання, транспортування) будь-яких виробів, матеріалів або енергії. Згідно даного державного стандарту автоматизована система – організаційно-технічна система, що складається із засобів автоматизації певного виду (чи кількох видів) діяльності, людей та персоналу, що здійснює цю діяльність. Згідно з ДСТУ 2941-94 автоматизована система контролю (далі - АСК) – це сукупність керованого об’єкта й автоматичних вимірювальних та керуючих пристроїв, у якій частину

функцій виконує людина.

Архітектура АС – концепція взаємозв'язку компонентів АС, що охоплює логічну, фізичну й програмну структури АС і принципи її функціонування. Послідовність дій компонентів АС під час виконання нею своїх функцій називається алгоритмом функціонування АС. Задача АС представляє собою функцію чи частину функцій АС, що є формалізованою сукупністю автоматичних дій, виконання яких приводить до результату заданого виду.

Керований елемент – це такий елемент об'єкту керування або АС, завдяки здійсненню керування яким реалізується одна чи кілька функцій АС.

Керівний елемент – елемент, що реалізує процес керування. Задачею АС є функція чи частина функції АС, що є формалізованою сукупністю автоматичних дій, виконання яких приводить до результату заданого виду.

Об'єкт керування – умовно виокремлена частина системи, на яку впливає система керування для досягнення необхідного результату.

Керування – сукупність цілеспрямованих дій, що включає оцінку ситуації та стану об'єкта керування, вибір керівних дій та їх реалізацію.

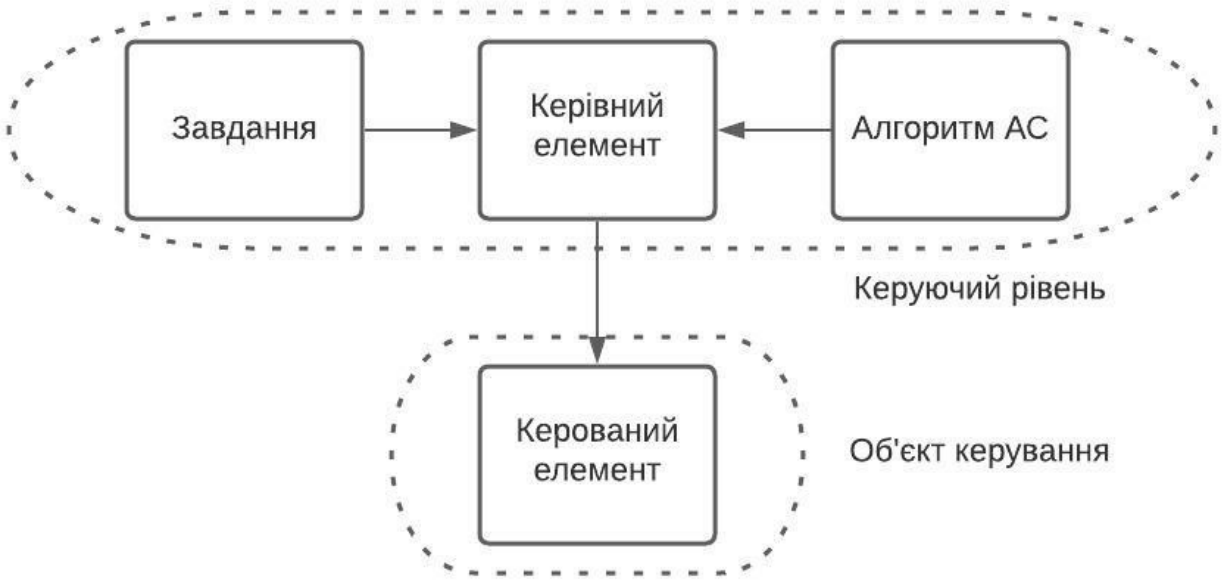


Рис. 1.3. Високорівнева структура системи керування

Згідно з дослідженням на підприємствах, де наявні автоматизовані елементи

виробництва та управління не має можливості розширити їх функціонал, у більшістості випадків автоматизації - є локальними, і не можуть бути розповсюджені на інші відділи, цехи, фабрики тощо.

Наприклад, існує риболовне підприємство, яке автоматизувало частину свого виробництва, а саме включення подачі кисню та його регулювання, подачі їжі, сповіщень про рівень забруднення води. Через два роки підприємство планує відкрити нову філію в іншому місці, але виникає проблема, що з часом встановлене обладнання зносилося або ж більше не продається, оскільки виробник не виготовляє його. Тому, виникає питання, як мати спільну адміністративну систему для застарілого та нового обладнання. У такому випадку доведеться адмініструвати на дві різні автоматизовані системи, без можливості централізованого контролю у одному центрі. Саме ця проблема є найбільшим недоліком сучасних автоматизованих систем. Наразі, майже не існує систем, які б змогли надати можливість за допомогою додатків встановлювати керуючі елементи, і - модульної архітектури підключати нових постачальників даних, працювати з ними одночасно, показувати вбудовану статистику, виконувати оцінку апаратного та програмного забезпечення, нотифікувати адміністраторів систем та мати кластерну архітектуру з можливістю реплікації і розподілення даних між філіалами.

1.4. Постановка задачі.

Нинішня архітектура автоматизованих систем для промисловості здійснила прогрес, але, щоб повністю реалізувати обіцянку четвертої промислової революції, потрібно принципово змінити технологічну модель. Потрібно прикласти чималі інженерні зусилля, відійти від встановлених бар'єрів, що перешкоджають впровадженню інновацій. Для цього необхідно покроково створити операційні вдосконалення за допомогою відкритої, заснованої на стандартах архітектури та екосистеми підключення і виготовлення додатків, здатної йти в ногу зі швидкістю ведення бізнесу у цифровому світі.

У світі IT стандартизовані, відкриті операційні системи, такі як Linux,

стимулюють активну, широку екосистему розробників до створення багатого та широкого асортименту портативних, інноваційних та економічно вигідних програмних рішень й послуг, які вирішують конкретні бізнес-проблеми. Сьогодні, ця парадигма не існує у світі промислової автоматизації. Для традиційних продавців промислової автоматизації природно протистояти змінам. Поняття «взаємодіючий» та «портативний» суперечать бізнес-моделям, які втратили актуальність, оскільки не відповідають цілям бізнесу в епоху діджиталізації. Однак, та сама динаміка призвела до скорочення кількості виробників комп'ютерів мейнфреймів, які продовжували забезпечувати функціонування застарілого програмного забезпечення.

Повне оцифрування виробничої та технологічної галузей вимагатиме відкритих систем автоматизації, які підтримують:

- Гнучку економічну взаємодію підприємства з IT-системами протягом усього життєвого циклу - від проєктування до операцій;
- Додатки, які можна створювати з використанням перевірених у використанні програмних компонентів, незалежно від базового обладнання, на якому буде працювати програмне забезпечення. Оскільки, це зменшує витрати та час виходу на ринок, одночасно, збільшуючи рентабельність інвестицій, гнучкість та загальну функціональність.

Впровадження автоматизованих систем контролю і обліку енергоресурсів є стратегічним напрямом підвищення ефективності енергетичного потенціалу країни. Також, необхідно пам’ятати, що в енергетичній галузі, як і в будь-якій іншій галузі значна кількість часу витрачається не на виробництво продукту, а на адміністрування ресурсів, документів, тощо.

Стандарт IEC 61499, який був прийнятий на заміну стандарту IEC 61131-3, адже він вирішує проблеми забезпечення переносимості, конфігурації та сумісності прикладного програмного забезпечення між постачальниками, та водночас незалежності програмного та апаратного забезпечення. IEC 61499

визначає мову дизайну систем високого рівня для розподіленої інформації та систем управління. Тим самим, це являє собою технічну основу універсальної автоматизації, оскільки вона:

1. Дозволяє створювати додатки для автоматизації з використанням портативних перевірених у користуванні програмних компонентів, незалежно від основного обладнання для автоматизації;
2. Дозволяє користувачеві розповсюджувати додаток на будь-яку обрану апаратну архітектуру системи - високорозподілену, централізовану або ж комбіновану. Апаратні машини можуть бути настільки малими, як інструменти та виконавчі механізми, або такими ж великими, як потужні комп'ютери;
3. Підтримує передовий досвід загальнодоступного програмного забезпечення, що спрощує створення програм автоматизації, які взаємодіють з IT-системами. Власна IT-конвергенція та легка портативність сприяє довгостроковому переходу від програмування власних контролерів до високоцінних систем автоматизації plug-and-produce, що використовують перевірені у використанні компоненти програмного забезпечення для автоматизації за низьку вартість.

Стандарт IEC 61499 уже працює на сучасних платформах на базі ПК та IPC, а також на обчислювальних пристроях, таких як Raspberry Pis. Керуючись попитом організацій кінцевих користувачів, тенденція до таких відкритих апаратних платформ продовжує зростати.

2 ПРОЄКТУВАННЯ СИСТЕМИ АВТОМАТИЗАЦІЇ

2.1. Вимоги до системи автоматизації

Сучасна система автоматизації повинна бути:

- модульною;
- надавати можливість додавати нові джерела даних та працювати з декількома такими джерелами одночасно;
- надавати можливість вбудовувати нові додатки для роботи з апаратним та програмним забезпеченням із зовнішніх ресурсів;
- мати мікросервісну архітектуру;
- надавати статистику виконань завдань, нотифікувати користувача у разі збоїв та несправностей;

Модульність програмного забезпечення - це розкладання програми на менші програми зі стандартизованими інтерфейсами. Поки що мікросервіси є тенденцією, і це по суті, про невеликі модулі, які вбудовані у цілу програмну систему. Uber та Netflix розповідають про те, як вони працюють з архітектурами мікросервісів, і до цього існувала подібна тенденція під назвою Service-Oriented Architecture (далі - SOA), яка орієнтувалась на більші модулі. Модульність програмного забезпечення у значній мірі поділяє ті самі визначення, що і апаратна модульність, із стратегічними чи функціональними модулями, які керуються потребами замовника та мають спільний стандартизований інтерфейс.

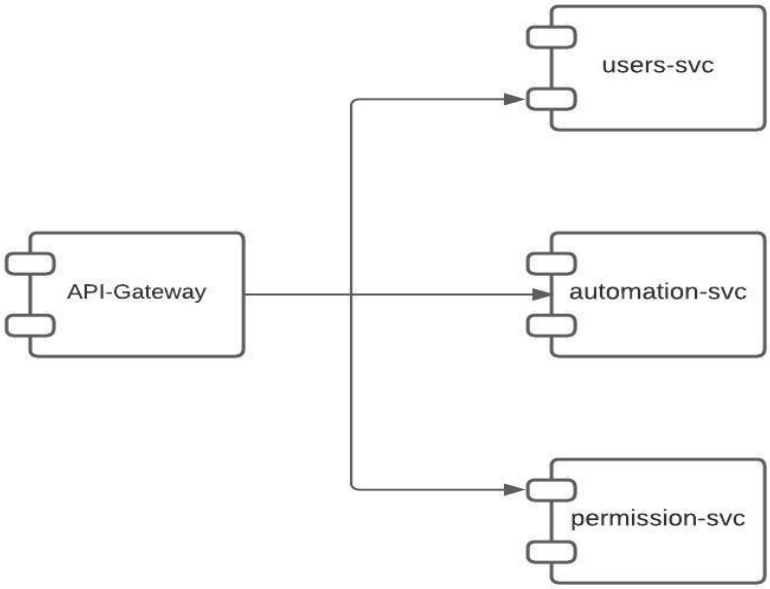


Рис. 2.1. Приклад мікро сервісної архітектури

На рисунку 2.1 зображена типова Back-End схема мікро-сервісної архітектури, де API-Gateway виступає у ролі головного proxy компонента, який пропускає усі запити від клієнтів до відповідних мікросервісів. У ролі функціональних мікросервісів виступають: users-svc, automation-svc, permission-svc.

Основні вимоги до мікро-сервісів:

- слабка зв'язність;
- компонетізація;
- єдина відповідальність;
- автономність;
- автоматичний CI / CD;

Відповідно система автоматизації повинна надавати можливість працювати з різними джерелами даних. Таку можливість можна реалізувати двома способами:

1. Розробити універсальну систему для отримання даних у компоненті автоматизації. Розробити універсальний, формалізований протокол для визначення формату вхідних даних на основі JSON, XML;

2. Розробити гнучку систему додатків (плагінів), які могли б бути встроєними в основну систему в період компіляції, або навіть на етапі запущеного додатку за допомогою спеціалізованого реєстру плагінів у системі автоматизації.

Перший варіант не потребує спеціальних технологій, ресурсів для того, щоб передати дані, для розробників та користувачів системи не буде важко розібратися із цим протоколом і форматом даних. Другий варіант більш складний, оскільки потребує технології, на якій створювалась система автоматизації, у контексті наукової роботи буде потрібно використовувати мову програмування Go. І, це потребує додаткових вмінь компілювання SO додатків та оптимізації програм, а також потребує розробки універсального інтерфейсу з описом можливих викликів цього інтерфейсу, а також створення спеціально реєстру додатків, плагінів з можливістю CRUD операцій на записах.

Перший формат є більш простим у реалізації та використанні, й не потребує специфічних навичок від адміністраторів та розробників. У даний час JSON формат є найпопулярнішим для опису запитів та відповідей від API. JSON - це текстовий формат обміну даними між комп'ютерами. JSON базується на тексті, може бути прочитаним людиною. Формат дає змогу описувати об'єкти та інші структури даних. Цей формат використовується переважно для передачі структурованої інформації через мережу (завдяки процесу, що називають серіалізацією).

Для того, щоб виконувати різні команди у системі, такі як управління світлом, енергоресурсами, турбінами, механізмами, датчиками необхідно використовувати Serverless підхід, та працювати з функціями, які можливо виконати під час роботи автоматизованої системи та окремо, автономно розгорнути функції в закритій, приватній хмарній технології на закритому сервері.

Serverless надає можливість сфокусувати свої зусилля на розробці, замість того, щоб витрачати цінні ресурси на створення вторинних можливостей, таких як

інфраструктура та обслуговування серверів, розробники можуть зосередитись на пропозиції своєї основної вартості. Швидше, ніж будуючи речі з нуля, розробники можуть з’єднувати збірні деталі та обрізати додаткову складність від основної програми. Ці самі serverless функції будуть викликатися з системи автоматизації, і будуть повністю передавати дані із джерела даних.

Найкорисніший спосіб визначити і зрозуміти Serverless - це зосередження уваги на декількох основних атрибутах, які відрізняють Serverless обчислення від інших обчислювальних моделей, а саме:

- Serverless модель не вимагає управління та експлуатації інфраструктури, що надає розробникам більше часу для оптимізації коду та розробки нових інноваційних функцій та функціональних можливостей;
- Serverless обчислення, як правило виконують код лише на вимогу, у контейнері без стану, на основі кожного запиту та прозоро масштабуються відповідно до кількості поданих запитів;
- Serverless обчислення дозволяють кінцевим споживачам платити лише за використовувані ресурси, ніколи не платячи за простої.

Виходячи із даного дослідження, найбільш ефективним рішенням для системи автоматизації буде використання фреймворка Nuclio. Nuclio - це високопродуктивний «Serverless» фреймворк, орієнтований на дані, введення / виведення та обчислення інтенсивних навантажень. Він добре інтегрований із популярними інструментами з обробки даних, такими як Jupyter та Kubeflow; підтримує різноманітні джерела даних та потокового передавання; і підтримує виконання над процесорами та графічними процесорами. Проєкт Nuclio було запущено у 2017 році, який постійно та швидко розвивається; багато стартапів та підприємств зараз використовують Nuclio у виробництві.

Для метрик та нотифікацій підійде такий інструмент як Grafana.



Рис 2.2. Приклад інтерфейсу Grafana

Grafana - це мультиплатформений веб-додаток для аналітики та інтерактивної візуалізації, який надає діаграми, графіки та оповіщення для Інтернету при підключенні до підтримуваних джерел даних.

2.2. Вибір БД

Існує багато різних видів баз даних, але на сьогодні є два основних лідери, які є найбільш поширеними у проектах:

1. Реляційні SQL бази даних;
2. Не реляційні NoSQL бази даних;

Реляційна база даних - це тип бази даних, що зберігає та забезпечує доступ до точок даних, пов’язаних між собою. Реляційні бази даних засновані на реляційній моделі, інтуїтивно зрозумілому, прямому способі представлення даних у таблицях. У реляційній базі даних кожен рядок таблиці є записом з унікальним ідентифікатором, який називається ключем. Стовпці таблиці містять атрибути даних, і кожен запис зазвичай має значення для кожного атрибута, що полегшує встановлення взаємозв’язків між точками даних. Проста, але потужна реляційна модель використовується організаціями всіх типів та розмірів для широкого спектру інформаційних потреб. Реляційні бази даних використовуються для

відстеження запасів, обробки транзакцій електронної комерції, управління величезними обсягами критично важливої інформації про клієнтів та багато іншого. Реляційну базу даних можна розглядати для будь-якої інформаційної потреби, у якій точки даних відносяться один до одного і повинні керуватися безпечним, заснованим на правилах, послідовним способом.

До головних переваг реляційних баз даних належать:

- можливість блокування таблиць;
- можливість працювати із даними в базі одночасно;
- можливість використовувати транзакції, які обладають ACID властивостями;
- консистентність даних.

Нереляційна база даних - це база даних, яка не використовує табличну схему рядків і стовпців, знайдену в більшості традиційних систем баз даних. Натомість нереляційні бази даних використовують модель зберігання, яка оптимізована для конкретних вимог до типу даних, що зберігаються. Наприклад, дані можуть зберігатися як прості пари ключ / значення, як документи JSON або як графік, що складається з ребер і вершин.

Усім цим сховищам даних є притаманне те, що вони не використовують реляційну модель. Крім того, вони, як правило, більш конкретні щодо типу даних, які вони підтримують, і способу отримання даних. Наприклад, сховища даних часових рядів оптимізовані для запитів за часовими послідовностями даних, тоді як сховища даних графіків оптимізовані для дослідження зважених взаємозв’язків між сутностями. Жоден формат не може добре узагальнити завдання управління даними транзакцій.

Термін NoSQL позначає сховища даних, які не використовують SQL для запитів, а замість цього використовують інші мови програмування та конструкції для запиту даних. На практиці NoSQL означає «нереляційна база даних», навіть незважаючи на те, що багато з цих баз даних підтримують SQL-сумісні запити. Однак, основна стратегія виконання запитів зазвичай сильно відрізняється від того, як традиційна СУБД виконує той самий запит SQL.

Зберігання даних документу управляє набором іменованих полів рядків та значень даних об'єкта, в сутності, яку називають документом. Ці сховища даних зазвичай зберігають дані у формі документів JSON. Кожне значення поля може бути скалярним елементом, таким як число, або складним елементом, таким як список або колекція батьків-дочірніх елементів. Дані в полях документа можуть кодуватися різними способами, включаючи XML, YAML, JSON, BSON або навіть зберігатись як звичайний текст. Поля в документах піддаються системі управління сховищем, що дозволяє додатку запитувати та фільтрувати дані за допомогою значень у цих полях.

Зазвичай документ містить усі дані для сутності. Елементи, які складають сутність є специфічними для кожної конкретної програми. Наприклад, організація може містити дані про клієнта, замовлення або їх комбінацію. Окремий документ може містити інформацію, яка буде розподілена між кількома реляційними таблицями в системі управління реляційними базами даних (СУБД). Зберігання документів не вимагає, щоб усі документи мали однакову структуру. Цей підхід у вільній формі забезпечує високу гнучкість. Наприклад, програми можуть зберігати різні дані у документах у відповідь на зміну бізнес-вимог.

Зберігання ключа / значення - це, по суті, велика хеш-таблиця. Ви пов'язуєте кожне значення даних з унікальним ключем, і сховище ключа / значення використовує цей ключ для зберігання даних за допомогою відповідної функції хешування. Функція хешування обрана для рівномірного розподілу хешованих ключів по сховищу даних.

Більшість сховищ ключів / значень підтримують лише прості операції запиту, вставки та видалення. Щоб змінити значення (частково або повністю), програма повинна переписати наявні дані для всього значення. У більшості реалізацій читання або запис одного значення є атомною операцією. Якщо значення велике, написання може зайняти деякий час.

Для цілей автоматизації на підприємствах підійде змішаний варіант, через те що основні дані стосовно автоматизації повинні бути структуровані, згруповані та пріоритезовані, а дані для функцій в Nuclio повинні бути

динамічними.

Найбільш правильним варіантом було б обрати PostgreSQL. PostgreSQL - це потужна об'єктно-реляційна система баз даних з відкритим кодом, яка активно працює понад 30 років, що забезпечило їй сильну репутацію надійності, надійності та продуктивності. PostgreSQL має безліч функцій, спрямованих на те, щоб допомогти розробникам створювати додатки, адміністраторам захищати цілісність даних та створювати відмовостійкі середовища, а також допомагати вам керувати вашими даними незалежно від того, наскільки великим чи малим є набір даних. На додаток до того, що PostgreSQL є безкоштовним та відкритим кодом, він дуже розширюваний. Наприклад, ви можете визначати власні типи даних, створювати власні функції, навіть писати код з різних мов програмування, не перекомпілюючи базу даних.

2.3. Проєктування системи автоматизації

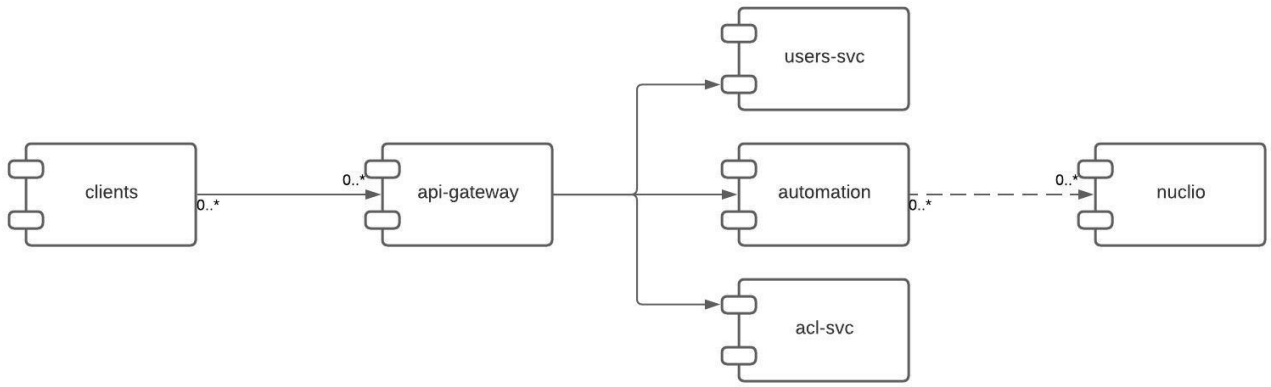


Рис 2.3. UML діаграма мікросервісів в системі автоматизації

У ролі клієнтів виступають: мобільні пристрої, веб-додатки, стороннє програмне забезпечення, апаратні пристрої, зовнішні сервера і т.п.

API-Gateway - використовуються для захисту від загроз, безпеки повідомлень, транспортної безпеки, організації, маршрутизації та трансформації. Шлюз API - це сервер, який є єдиною точкою входу в систему. Це схоже на фасадний малюнок з об’єктно-орієнтованого дизайну. Шлюз API інкапсулює внутрішню архітектуру системи та надає API, призначений для кожного клієнта.

Він може мати інші обов'язки, такі як автентифікація, моніторинг, балансування навантаження, кешування, формування та управління запитами та обробка статичних відповідей. API-Gateway відповідає за маршрутизацію запитів, композицію та переклад протоколів. Усі запити від клієнтів спочатку проходять через шлюз API. Потім він направляє запити до відповідної мікросервісу. Шлюз API часто обробляє запит, викликаючи кілька мікро-сервісів та агрегуючі результати. Він може перекладати між веб-протоколами, такими як HTTP та WebSocket, та веб-недружніми протоколами, які використовуються всередині на Backend стороні.

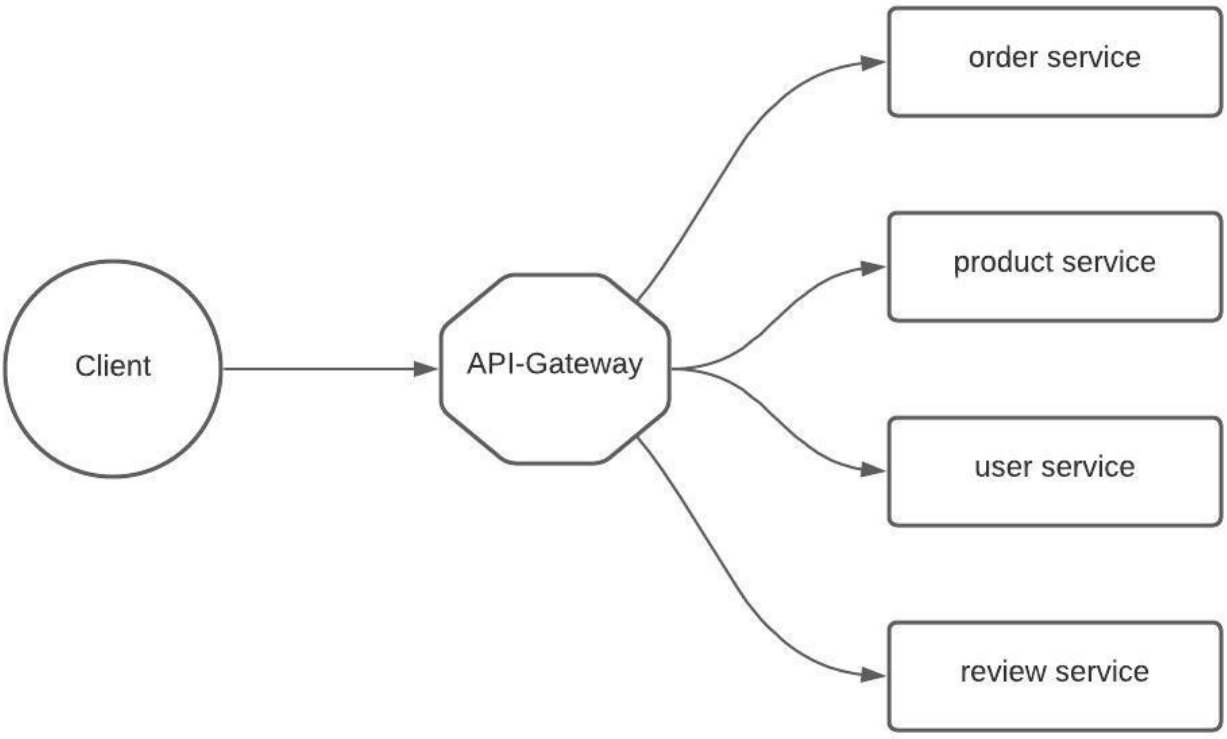


Рис 2.4. Діаграма функціонала API-Gateway

Мікросервіс users-svc - відповідає за реєстрацію користувачів, зберігання даних та управління даними, відновлення паролю та аутентифікацію.

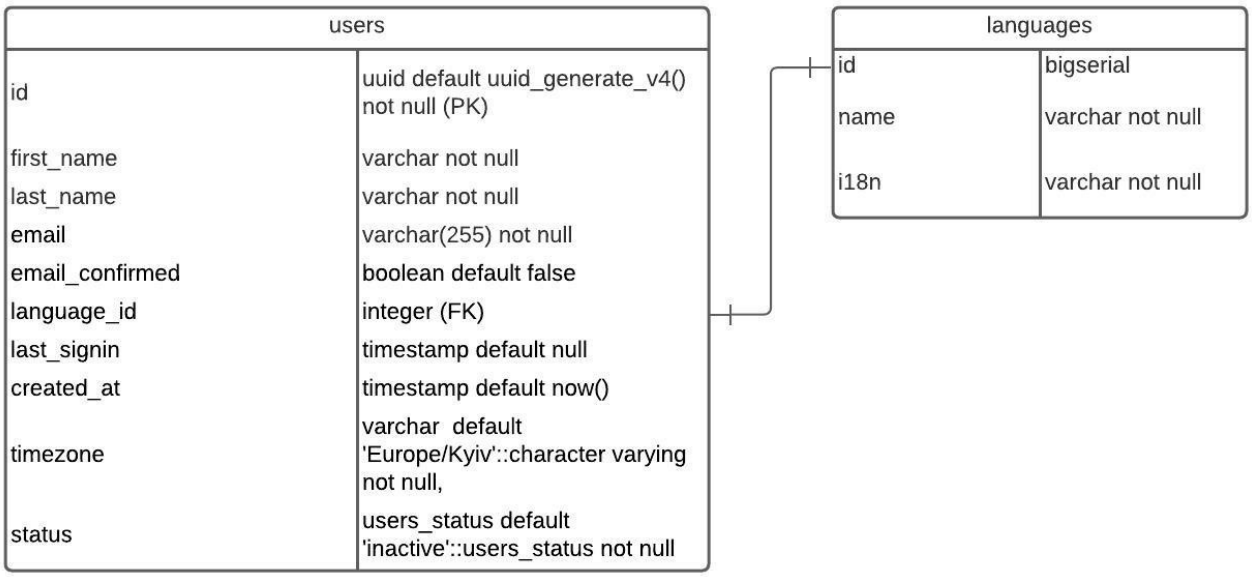


Рис 2.5. ERD бази даних для мікросервіса users-svc

Мікросервіс acl-svc контролює доступ користувачів до інших компонентів за допомогою протокола RBAC. Розробка цього компонента буде за допомогою фреймворка Casbin. Casbin - це потужна та ефективна бібліотека контролю доступу з відкритим кодом для проектів Golang. Він забезпечує підтримку забезпечення авторизації на основі різних моделей контролю доступу. У Casbin модель контролю доступу абстрагується у файл CONF на основі метамоделі PERM (Policy, Effect, Request, Matchers). Отже, переключення або оновлення механізму авторизації для проекту так само просто, як зміна конфігурації. Ви можете налаштувати власну модель контролю доступу, об’єднавши доступні моделі. Наприклад, ви можете отримати ролі RBAC та атрибути ABAC разом в одній моделі та спільно використовувати один набір правил політики. У безпеці комп’ютерних систем контроль за рольовим доступом або рольова безпека є підходом до обмеження доступу до системи авторизованим користувачам. Він використовується більшістю підприємств, що мають понад 500 працівників, і може запровадити обов’язковий контроль доступу (MAC) або дискреційний контроль доступу (DAC).

Мікросервіс automation-svc створює сценарії автоматизації, управляє

даними, викликає функції, веде статистику виконань, опрацьовує помилки виконань, активує та деактивує сценарії.

Рис 2.6. UML мікросервіса автоматизації та взаємодії з Nuclio та командами

Функції можна викликати через різні джерела подій, які визначені у функції (наприклад, HTTP, RabbitMQ, Kafka, Kinesis, NATS, DynamoDB, потік платформи наукових даних Iguazio або графік). Джерела подій поділяються на кілька класів подій (req / rep, async, stream, pooling), які визначають поведінку джерел. Різні джерела подій можуть без проблем підключатись до однієї і тієї ж функції без шкоди для продуктивності, забезпечуючи портативність, повторне використання коду та гнучкість.

Процесори функцій забезпечують середовище для виконання функцій. Процесор подає функції з подіями, забезпечує контекст та дані, збирає журнали та статистику та управляє життєвим циклом функції. Процесори можуть бути скомпільовані в один бінарний файл (при використанні Go або C) або упаковані в контейнер з усіма необхідними залежностями коду. Контейнери процесора можуть працювати як окремі контейнери Docker, так і поверх платформи для організації контейнерів, наприклад Kubernetes. Кожна функція має власні процесори функцій. Процесори функцій будуть масштабовані автоматично (шляхом додавання більше екземплярів контейнера) для вирішення високої частоти подій.

Слухачі джерела подій можуть прослуховувати сокети та черги повідомлень або періодично отримувати події із зовнішніх джерел подій або даних. Отримані події мають спільну схему, яка відокремлює логіку функції від реалізації джерела події або конкретної структури та передається одному або декільком паралельним робочим середовищем виконання. Прослуховувачі подій також гарантують виконання рівно одного разу або принаймні одного разу та обробку помилок - наприклад, зберігаючи контрольні точки потоку, підтверджуючи або повторюючи події черги повідомлень або відповідаючи на запити клієнта HTTP.

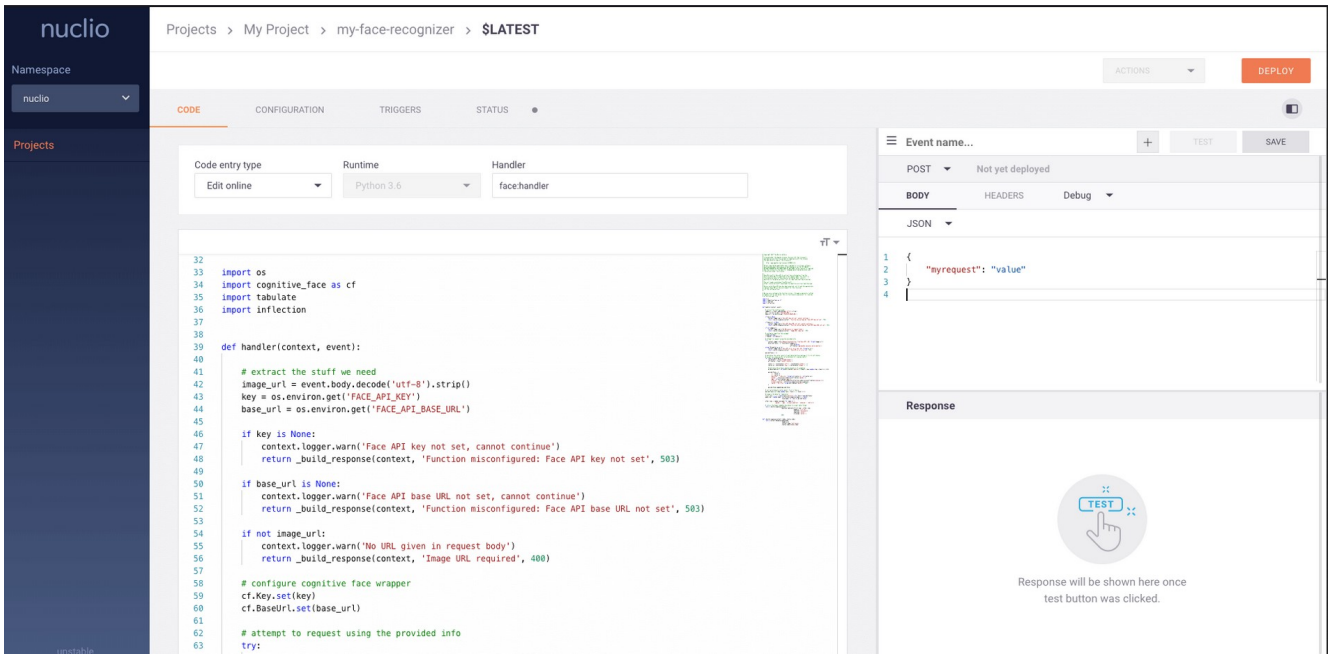


Рис 2.8. Інтерфейс Nuclio

Функції можуть отримати вигоду від постійних з'єднань даних із зовнішніми файлами, об'єктами, базами даних або системами обміну повідомленнями. Час виконання ініціалізує з'єднання служби даних на основі типу, URL-адреси, властивостей та облікових даних, зазначених у специфікації функції, і подає їх функції через об'єкт контексту.

Прив’язка даних спрощує розробку, позбавляючи потреби інтегруватися з SDK або керувати зв’язками та обліковими даними. Вони також уможливлюють повторне використання та перенесення функцій, оскільки різні служби даних одного класу відображаються у функцію за допомогою однакових API. Прив'язки даних також можуть обробляти аспекти попереднього отримання, кешування та мікро-пакетного управління, щоб зменшити затримку виконання та підвищити продуктивність вводу-виводу. Прив'язки даних та джерела подій розроблені з нульовим копіюванням, нульовою серіалізацією та неблокуючими операціями, і забезпечують роботу в режимі реального часу без необхідності будь-якого спеціального коду функції.

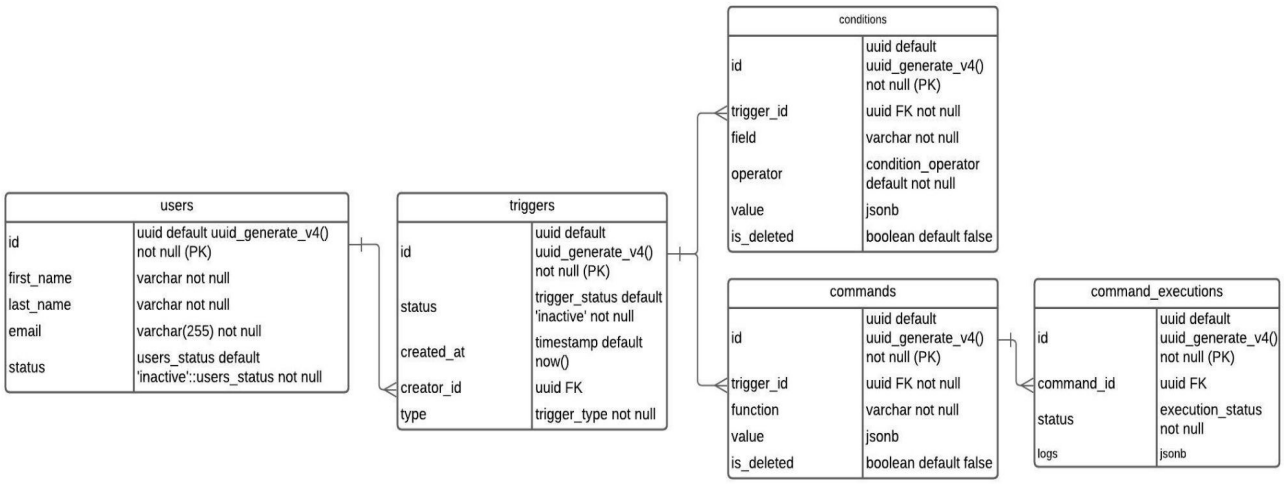


Рис 2.9. ERD мікросервіса automation

На рис. 1.8. зображена діаграма даних та взаємовідносин між ними. Кожен об’єкт - це окрема таблиця в PostgreSQL базі даних. Таблиця users реплікується через NATS-сповіщення від мікросервіса users-svc, ця таблиця слугує для того, щоб відображати дані користувачів які створили тригери. Таблиця triggers - відображає кожен сценарій автоматизації. В цій таблиця є наступні стовбці: id, status, created_at, creator_id, type. Стовпець status може мати кілька значень: active, deleted. Таблиця conditions - описує можливі умови, які повинні бути перевірені у вхідних даних. Conditions таблиця містить наступні стовпці: id, trigger_id, operator, datafield, value, is_deleted. Де trigger_id - це зовнішній ключ на таблицю triggers, field - поле в структурі JSON, яке необхідно перевірити, operator - оператор який

треба виконати на цільових полях, такі як: is, is not, in, contains, greater, less. fold equal. Value - динамічні значення, котрі задають користувачі. Таблиця commands - описує команді які мають буду виконані, якщо всі умови описані в таблиці conditions - виконані. Таблиця складаються з наступних стовпців: id, trigger_id, function, value, is_deleted. Де function - це ім’я-ключ яке використовується для виклику Nuclio функцій. Таблиця command_executions слугує для того, щоб записати результати виконання функцій в Nuclio, вона зберігає результати у вигляді відповіді від функцій та її логів в Nuclio системі.

3 УНІВЕРСАЛЬНА СИСТЕМИ АВТОМАТИЗАЦІЇ В ЕНЕРГЕТИЦІ

Сучасне суспільство щодня напряму залежить від управління енергією, газом, водою та відходами. Кожне домогосподарство, кожна людина потребує надання цих послуг для задоволення своїх щоденних потреб, що у свою чергу надає енергетичному та комунальному секторам пріоритетне становище. Як було сказано раніше, процеси виробництва електроенергії у більшості випадків мають багато чого спільного з виробництво продукції та товарів у будь-якій іншій сфері. Зокрема, енергетичні компанії часто мають справу з великою кількістю дуже повторюваних процесів, таких як прогнозування цін, передача файлів та звітування. Наявність можливості приймати ці повторювані, трудомісткі ручні процеси та перетворювати їх на автоматизовані, динамічні робочі процеси означає велику економію часу та ресурсів.

Давайте розглянемо деякі найпоширеніші процеси, які енергетичні компанії можуть автоматизувати, щоб отримати віддачу від своїх інвестицій в автоматизацію.

Послуга вдосконаленого обладнання - допоможе службі обслуговування на місцях забезпечити найкращий досвід клієнтів, зробивши дзвінки в служби більш ефективними. Саме така послуга автоматизує процес створення робочих замовлень, звітів про обслуговування обладнання та дозволів на повернення, і найголовніше, що, при цьому вся відповідна інформація з’являється автоматично. Автоматизація дозволяє зробити більший обсяг роботи за короткий час. За допомогою автоматизації можливо відобразити весь виробничий процес і визначити можливості автоматизації у формі запиту на технічне обслуговування на затвердження, відправлення, ремонт, оновлення записів та виставлення рахунків. Затримки усуваються, оскільки ключові дані миттєво надходять до працівника, якому вони потрібні для виконання його обов'язків у процесі, і автоматично передаються наступному працівнику після завершення.

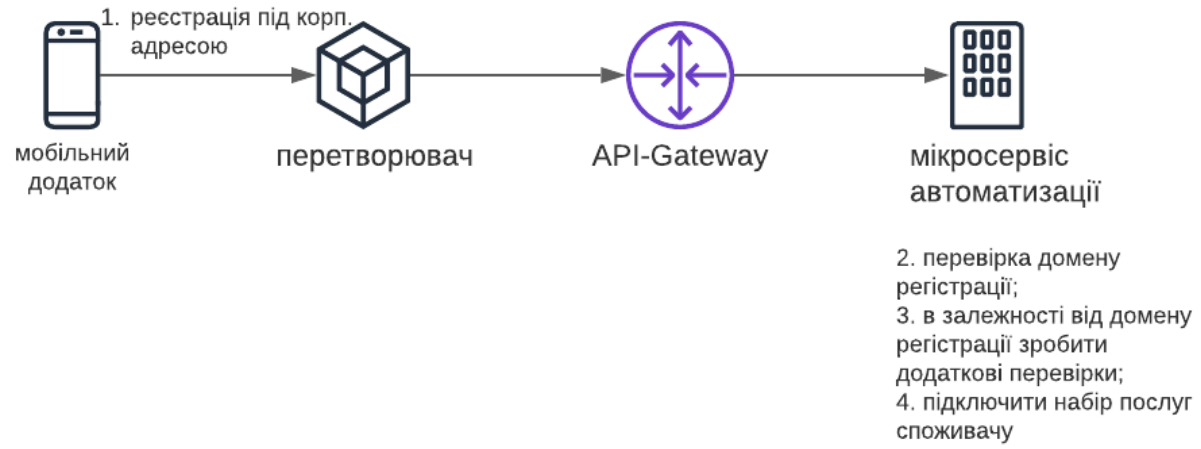


Рис 3.1. Автоматизація обслуговування користувачів

На рис. 3.1. зображена реєстрація користувача, до якого застосовуються спеціальні умови автоматизації.

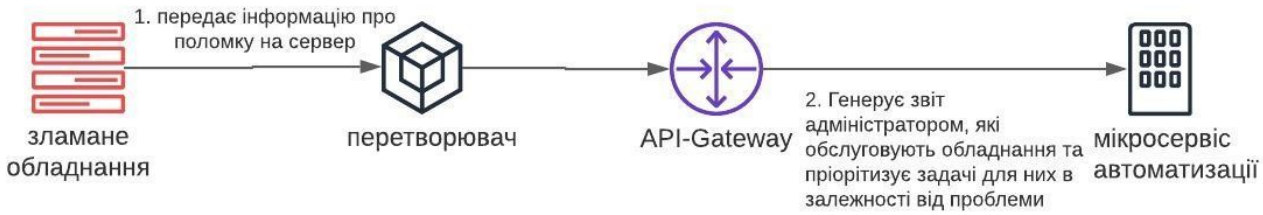


Рис 3.2. Автоматизація нотифікацій про зламане обладнання

На рис 3.2. проілюстровано процес автоматизації для зламаного обладнання, який створює звіт для адміністраторів мережі, створює для них задачі та пріорітизує їх.

У нинішніх умовах важливе значення у енергетичній галузі має така властивість як швидке реагування на різноманітні поломки, катастрофи, оскільки, тут найголовнішим критерієм усунення проблеми є швидкість її усунення, тобто час. Нехтування даним критерієм може бути згубним для навколишнього середовища та завдати колосальних збитків компанії.

Для ефективного реагування у таких ситуаціях енергетичні компанії можуть здійснити автоматизацію завдання негайного виправлення. За таких умов динамічні форми можуть забезпечити якнайшвидший збір критичної інформації, а генерація документів може використовувати ці дані для створення детальних робочих замовлень, щоб можна було вжити відповідні заходи.

Автоматизована реєстрація клієнтів - Як правило, важливу роль відіграє автоматизована реєстрація клієнтів, адже сучасні споживачі очікують швидкого та простого досвіду роботи. Даний процес передбачає можливості самообслуговування під час створення та управління своїми рахунками. Саме застосування автоматизованої реєстрації допомагає покращити зворотній зв'язок між сторонами за допомогою автоматизованих повідомлень та опитувань.

Обслуговування обладнання з підтримкою IoT - Однією з найголовніших частин управління енергетичною організацією є перевірка ефективності та безпеки роботи обладнання, що використовується.

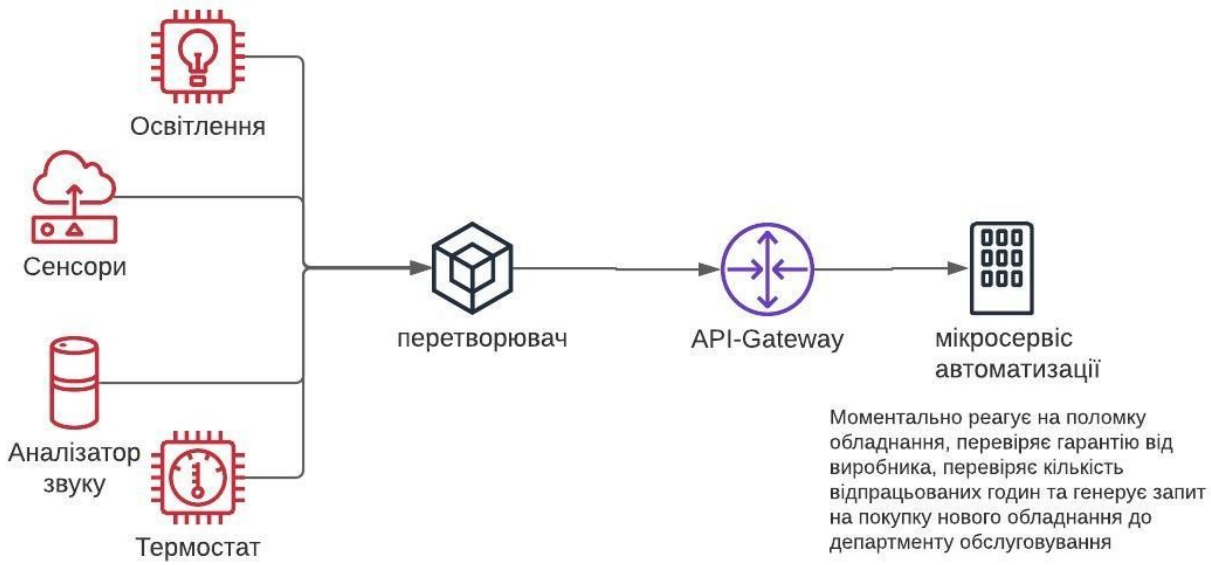


Рис 3.3. Реагування системи автоматизації на катастрофи

Однак, моніторинг обладнання вручну на наявність несправностей є

повільним і важким завданням - не найкращим використанням часу ваших працівників. Енергетичні організації з метою економії своїх ресурсів (час, гроші) можуть здійснювати обслуговування обладнання та застосовувати інтелектуальне та профілактичне обслуговування за допомогою поєднання IoT й автоматизації процесів. Завдяки автоматичному плануванню технічного обслуговування та динамічному формуванню службового документа ви можете вирішити будь-яку проблему до її поломки та спричинення простоїв, збитків.

ВИСНОВКИ

У ході виконання дипломної роботи магістра була досягнута мета розробити універсальне програмне забезпечення для автоматизації процесів в енергетиці та на підприємствах, яке складається з багатьох пластів та мікро-сервісів, а також баз даних, окрім цього були вирішені наступні поставлені завдання:

1. Проведено аналіз предметної області та існуючих автоматизованих систем, відзначені недоліки і знайдені шляхи усунення їх;

2. Розроблено алгоритм роботи сервісів і алгоритм обслуговування таких систем;

3. Розроблено структуру бази даних зберігання інформації про елементах форм, користувачах і користувальницьких формах, реалізована БД в PostgreSQL;

4. Розроблені сервіси для автоматизації, з використанням мікросервісної архітектури, Kubernetes інфраструктура для обслуговування та Circle CI/CD для автоматизації керування процесів розробки та перевірки. Використано новий інноваційний підхід до роботи з функціональними FaaS-рішеннями;

5. Протестована роботоспособність мережі, були проведені unit та інтеграційні та end-to-end тести.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Офіційна документація мови програмування Go [електронний ресурс] // https://golang.org/
2. Офіційна документація Kubernetes [електронний ресурс] // https://kubernetes.io/
3. Автоматизація рутинних процесів у енергетичній сфері [електронний ресурс] // https://www.advsyscon.com/blog/automation-energy-company/
4. Роботизація процесів автоматизації у енергетиці та комунальному секторі [електронний ресурс] // https://www.crmt.com/en/resources/blog/robotic-process-automation-in-energy-utilities-industry/
5. Чому автоматизація процесів у енергетичній галузі є суттєвою [електронний ресурс] // https://www.nintex.com/blog/why-process-automation-in-the-energy-industry-is-essential/
6. Інтелектуальні електричні мережі електроенергетичних систем та їхнє технологічне забезпечення / Б.С. Стогній, О.В. Кириленко, С.П. Денисюк [електронний ресурс] // http://dspace.nbuv.gov.ua/handle/123456789/61922
7. Автоматизація технологічних процесів і системи автоматичног керування / О.В. Барало, П.Г. Самойленко, С.Є. Гранат, В.О. Ковальов [електронний ресурс] // https://asutp.in.ua/wp-content/uploads/2019/12/ebook_automatizacija_tehnologichnih_procesiv_i_sistem_avtomatichnogo_keruvannya.pdf
8. SMART системи як одна із основних складових сталого розвитку енергетики / Ю.Т. Якименко, В.В. Прокопенко, С.П. Денисюк, О.М. Закладний [електронний ресурс] // https://ela.kpi.ua/bitstream/123456789/2757/1/1.pdf
9. Дослідження системи автоматизованого керування енергетичним

обладнанням / Є.А. Пилипенко, В.Г. Здоренко [електронний ресурс] // https://er.knutd.edu.ua/bitstream/123456789/11490/1/NRMSE2018_V2_P295-296.pdf

10. Багатокритеріальний підхід до оцінки альтернативних варіантів гібридних енергетичних систем / О.В. Бойко, В.В. Шендрик, Ю.В. Парфененко, С.О. Шендрик [електронний ресурс] // http://dspace.khntusg.com.ua/bitstream/123456789/10329/1/34.pdf
11. Управління системою виробництва енергії - інтеграція відновлювального джерела енергії із системою автоматизації будівель [електронний ресурс] // https://www.sciencedirect.com/science/article/abs/pii/S0196890409005263
12. Мережі зв'язку ліній електропередач для широкомасштабних систем управління та автоматизації [електронний ресурс] // https://ieeexplore.ieee.org/abstract/document/5439083
13. Інтегрована система енергоефективної автоматизації будівель: застосування у третинному секторі [електронний ресурс] // https://www.sciencedirect.com/science/article/abs/pii/S0306261912004059
14. Інтелектуальне управління енергією на основі знань за допомогою системи автоматизації будівель [електронний ресурс] // https://ieeexplore.ieee.org/abstract/document/5696994
15. Нова система бездротового живлення для пристроїв бездротового зв'язку в системах промислової автоматизації [електронний ресурс] // https://ieeexplore.ieee.org/abstract/document/1185473
16. Шлях до універсальної автоматизації chttps://www.arcweb.com/industry-best-practices/road-universal-automation-0
17. Програмне забезпечення для вбудованих систем [електронний ресурс] // https://www.arcweb.com/market-studies/embedded-systems-software
18. Програмне забезпечення для управління енергією [електронний ресурс] // https://www.arcweb.com/market-studies/energy-management-software
19. Розподілені системи управління [електронний ресурс] // https://www.arcweb.com/market-studies/distributed-control-systems

20. Сучасні автоматизовані системи контролю та обліку енергоресурсів (АСКОЕ) [електронний ресурс] // http://energetika.in.ua/ua/books/book-5/part-4/section-6
21. Internet of Things та енергетична промисловість [електронний ресурс] // https://www.intel.com/content/www/us/en/energy/energy-iot.html
22. Як IoT перетворює енергетичну промисловість [електронний ресурс] // https://www.renewableenergymagazine.com/emily-folk/how-iot-is-transforming-the-energy-industry-20190418
23. Системи автоматизації промислових процесів: проєктування та впровадження / B.R. Mehta, Y. Jaganmohan Reddy [електронний ресурс] // https://books.google.com.ua/books?hl=uk&lr=&id=Iz_LAwAAQBAJ&oi=fnd&pg=PP1&dq=implementation+of+automation+energy+system&ots=sMFjYlBUHh&sig=fDnItG5vY8LS-L3kT8KdoPLIUDU&redir_esc=y#v=onepage&q=implementation%20of%20automation%20energy%20system&f=false
24. Автоматизація робочого процесу для комбінованого моделювання будівель та центральних енергетичних систем [електронний ресурс] // https://www.sciencedirect.com/science/article/abs/pii/S0360544216304297
25. Розбиття динамічного руйнування ексергії в енергетичній системі будівлі на ендогенні та екзогенні частини за допомогою вимірюваних даних системи автоматизації будівлі [електронний ресурс] // https://onlinelibrary.wiley.com/doi/full/10.1002/er.5213
26. Архітектура комунікаційної мережі підстанції IEC 61850 для ефективної автоматизації енергетичної системи [електронний ресурс] // https://www.tandfonline.com/doi/full/10.1080/23317000.2015.1043475
27. Що таке IoT? [електронний ресурс] // https://www.oracle.com/internet-of-things/what-is-iot/
28. Що таке Azure Internet of Things (IoT)? [електронний ресурс] // https://docs.microsoft.com/en-us/azure/iot-fundamentals/iot-introduction
29. Нафтові компанії підключаються до Інтернету, щоб стати більш

оперативно ефективними [електронний ресурс] // https://www.businessinsider.com/the-iot-in-the-upstream-oil-and-gas-industry-report-companies-are-looking-to-the-iot-to-become-more-operationally-efficient-2015-11

30. Універсальна автоматизація [електронний ресурс] // https://download.schneider-electric.com/files?p_enDocType=White+Paper&p_File_Name=998-21066447_UniversalAutomation_eReport.pdf&p_Doc_Ref=998-21066447
31. IEC 61499 [електронний ресурс] // https://en.wikipedia.org/wiki/IEC_61499
32. Автоматизовані системи керування технологічними процесами / А.О. Бобух [електронний ресурс] // https://core.ac.uk/download/pdf/11319166.pdf
33. Визначення та принцип будови автоматизованих систем. Поняття про АСУП (автоматизована система управління підприємством), САПР (система автоматизованого проектування) [електронний ресурс] // http://hpsl.in.ua/images/stories/2020_Inf_texnologii_distan_nav/OK_Gr_191_1603_1604/%D1%83%D1%80%D0%BE%D0%BA%206.pdf
34. 5 основних елементів модульного дизайну програмного забезпечення [електронний ресурс] // https://shanebdavis.medium.com/the-5-essential-elements-of-modular-software-design-6b333918e543
35. Мікросервіси [електронний ресурс] // https://martinfowler.com/articles/microservices.html
36. JSON [електронний ресурс] // https://uk.wikipedia.org/wiki/JSON
37. Nuclio [електронний ресурс] // https://nuclio.io/
38. Що таке безсерверність? [електронний ресурс] // https://www.redhat.com/en/topics/cloud-native-apps/what-is-serverless
39. Що таке реляційна база даних [електронний ресурс] // https://www.oracle.com/database/what-is-a-relational-database/
40. Microsoft стаття про не реляційні бази даних [електронний ресурс] // https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/non-relational-data

41. Система контролю доступу користувачів [електронний ресурс] // https://casbin.org/