

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ

к.т.н, доц. Михелєв І.Л.

" ____ " _____ 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти "магістр"

**на тему: Дослідження методів аналізу тимчасових рядів та розробка системи
моніторингу фінансових операцій**

Виконала: студентка групи 6144 м

_____ ***Волянський Ю.С.***
(підпис)

Керівник роботи:

доцент кафедри ІУСТ, к.т.н.,
(посада, науковий ступінь вчене звання)

_____ ***Партас В.К.***
(підпис)

м. Миколаїв – 2023 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами

Кафедра інформаційних управляючих систем та технологій

Спеціальність 126 "Інформаційні системи та технології"

Освітня програма "Інформаційні системи та технології"

"ЗАТВЕРДЖУЮ"

Гарант освітньої програми

к.т.н, доц. Чернова Лб.С.

" ____ " _____ 2023 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ на здобуття ступеня вищої освіти "магістр"

Студенту Волянському Юліану Сергійовичу

(прізвище, ім'я, по батькові)

1. Тема роботи: Дослідження методів аналізу тимчасових рядів та розробка системи моніторингу фінансових операцій

Керівник роботи к.т.н., доц. Парнас В.К

Затверджені наказом ректора № 1098-уч. від "18" вересня 2023 року.

2. Термін подання роботи: 15 грудня 2023 р.

3. Вихідні дані до проекту (роботи) ДСТУ щодо обробки інформації, літературні джерела, технічна документація на існуючі аналоги систем, матеріали наукового стажування

4. Перелік питань, що належать до розробки (найменування розділів)

Вступ, 1 Аналіз і дослідження предметної області, постановка завдання моніторингу проведення платіжних транзакцій 2 Розробка проектних рішень, 3 Реалізація проекту системи моніторингу 4 Розрахунок економічної ефективності, 5 Охорона праці, 6 Охорона навколишнього середовища, Висновки, Список використаних джерел. Додатки

5. Перелік презентаційних матеріалів Мета роботи, Методи дослідження та наукова новизна одержаних результатів, Задачі. Діаграма варіантів використання, Діаграма компонентів, Діаграма послідовності, Діаграма класів Результати розробки. Висновки

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	к.т.н., доц. Паргас В.К.		
2	к.т.н., доц. Паргас В.К.		
3	к.т.н., доц. Паргас В.К.		
4	ст. викл. Трушлякова А.Б.		
5	ст. викл. Гурець Н.В.		
6	ст. викл. Гурець Н.В.		

7. Дата видачі завдання 05 вересня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Аналіз предметної області та методів дослідження тимчасових рядів проведення платіжних транзакцій	04.09.2012–30.11.2012	
2	Розробка проектних рішень	01.12.2022–28.02.2023	
3	Реалізація проекту системи моніторингу	01.03.2023–27.06.2023	
4	Розрахунок економічної ефективності	03.09.2023–18.09.2023	
5	Вирішення питань охорони праці та охорони навколишнього середовища	19.09.2023–30.09.2023	
6	Виконання графічних документів	01.10.2023–31.10.2023	
7	Оформлення пояснювальної записки	01.11.2023–15.12.2023	
8	Подання кваліфікаційної магістерської роботи (МР) на кафедральний захист	16.12.2018	
9	Подання МР рецензенту	17.12.2018	
10	Подання МР на державний захист	21.12.2018	

Студент

(підпис)

Волянський Ю.С.

(ПІБ)

Керівник роботи

(підпис)

Паргас В.К.

(ПІБ)

РЕФЕРАТ

Волянський Юліан Сергійович

«Дослідження методів аналізу тимчасових рядів та розробка системи моніторингу фінансових операцій»

Кваліфікаційна магістерська робота на здобуття ступеня вищої освіти магістра зі спеціальності 126 «Інформаційні системи та технології».

Національний ун-т кораблебудування ім. адмірала Макарова. м. Миколаїв, 2023.

Обсяг роботи. 114 с., 16 табл., 31 рис., 40 використаних джерел, 2 додатка.

Актуальність теми роботи. Великий потік транзакцій також накладає обмеження на зберігання історії, по якій можна проводити аналіз. Для зберігання у базі даних потрібно нормалізацію, тобто індексацію за усіма необхідними критеріями для подальшого аналізу в реальному часі частенько просто неможлива. Проблема збереження і обробки даних, що містять інформацію про платіжні транзакції, що проводяться, для відображення агрегованих метрик на графіках в реальному часі, для аналізу отриманих метрик і генерації сповіщень у разі зареєстрованих відхилень при аналізі є актуальною.

Мета дослідження. Дослідження тимчасових рядів, зберігання й обробки даних, що містять інформацію про платіжні транзакції, що проводяться, їх аналізу і можливості сповіщення про нестандартну поведінку потоку транзакцій.

Завдання дослідження:

- Провести аналіз існуючих методів аналізу тимчасових рядів.
- Запропонувати найкращий;
- Дослідити ефективність обраного методу

Об'єкт дослідження. : методи аналізу тимчасових рядів.

Предметом дослідження. Засоби розробки систем моніторингу платіжних транзакцій.

Наукова новизна одержаних результатів. Полягає в обґрунтуванні та вирішенні проблеми підвищення якості обробки поведінки потоку транзакцій як інформаційних продуктів шляхом урахування тимчасових залежностей даних .

Практичне значення одержаних результатів. Розроблена система моніторингу скоротить фінансові і тимчасові витрати на аналіз виникаючих проблем в компаніях, що займаються електронною комерцією і проводять платежі з використанням зовнішніх платіжних систем. Можливий розвиток розробленої системи в окремий комерційний продукт, який універсально використовуватиметься компаній з подібною сферою роботи.

Апробація результатів досліджень. Під час . V Міжнародної студентської наукової конференції «Наука сьогодення: від досліджень до стратегічних рішень» (22.12.2023, м. Чернівці, Україна)

Публікації. Одна друкована праця у збірнику тез конференції:

1. Проєкт системи моніторингу фінансових операцій.

Ключові слова: програмне забезпечення, моніторинг, метрики, часові ряди, платіжні системи, аналіз

ABSTRACT

Volianskyi Yulian Serhiyovych

«Research of methods of time series analysis and development of a system for monitoring financial transactions»

The qualification work for the degree of higher education Master's degree in specialty 122 «Information systems and technologies». The Admiral Makarov National University of Shipbuilding. Mykolaiv, 2022.

The scope of the work. 114 p., 16 tabl., 31 fig., 40 references, and 2 appendices.

Relevance of the topic of work. A large flow of transactions also imposes a limit on the amount of history that can be analyzed. For storage in the database, normalization is required, that is, indexing according to all the necessary criteria for further analysis in real time is often simply impossible. The problem of saving and processing data containing information about the payment transactions being carried out, for displaying aggregated metrics on graphs in real time, for analyzing the obtained metrics and generating notifications in case of registered deviations during the analysis is relevant.

The aim of the study. Study of time series, storage and processing of data containing information about ongoing payment transactions, their analysis and the possibility of notification of non-standard behavior of the transaction flow.

Objectives of the study:

- conduct an analysis of existing methods of time series analysis.
- offer the best;
- investigate the effectiveness of the chosen method

The object of the. Study methods of time series analysis

The subject of study. Tools for developing payment transaction monitoring systems.

The scientific novelty of the obtained results. It consists in substantiating and solving the problem of improving the quality of processing the behavior of the flow of transactions as information products by taking into account the temporal dependencies of the data.

The practical value of the results. The developed monitoring system will reduce financial and time costs for analyzing emerging problems in companies engaged in electronic commerce and making payments using external payment systems. It is possible to develop the developed system into a separate commercial product, which will be universally used by companies with a similar field of work.

Approbation results of the research During the V International Student Scientific Conference "Today's Science: From Research to Strategic Decisions" (December 22, 2023, Chernivtsi, Ukraine).

Publications 1 printed work was published with the abstracts for the conferences.

1. Project of the monitoring system of financial operations.

Keywords: software, monitoring, metrics, time series, payment systems, analysis.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ.....	8
ВСТУП.....	9
РОЗДІЛ 1 ДОСЛІДЖЕННЯ МЕТОДІВ АНАЛІЗУ ТИМЧАСОВИХ РЯДІВ, ПОСТАНОВКА ЗАВДАННЯ МОНІТОРИНГУ ПРОВЕДЕННЯ ПЛАТІЖНИХ ТРАНЗАКЦІЙ	10
1.1 Теоретичні основи проведення платіжних та методів дослідження тимчасових рядів сфері.....	10
1.2 Черги повідомлень	11
1.3 InfluxDB – база даних для збереження тимчасових рядів	13
1.4 Telegram та використання його ботів для оповіщення.....	15
1.5 Аналіз метрик та функцій агрегації.....	17
1.6 Використовувані технології.....	20
1.7 Аналітичний огляд існуючих рішень.....	21
1.8 Аналіз вимог до системи моніторингу проведення платіжних транзакцій.....	25
РОЗДІЛ 2 РОЗРОБКА ПРОЕКТНИХ РІШЕНЬ	30
2.1 Рішення з інформаційного забезпечення	30
2.2 Рішення з інформаційної системи	36
2.3 Рішення з програмного забезпечення	42
РОЗДІЛ 3 РОЗРОБКА ПРОЕКТУ СИСТЕМИ МОНІТОРИНГУ	65
3.1 Вибір мови розробки проекту та технологій	65
3.2 Результати розробки	67
3.3 Перевірка працездатності системи	70
РОЗДІЛ 4. РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ	71
РОЗДІЛ 5. ОХОРОНА ПРАЦІ	75
РОЗДІЛ 6. ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА.....	90
ВИСНОВКИ	99
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	101

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД - База Даних;
ОС - Операційна Система;
ПО - Програмне Забезпечення;
ПК - Програмний Комплекс;
СУБД - Система Управління Базою Даних;
API - Application Programming Interface;
CRUD - Create, Read, Update, Delete;
CSS - Cascading Style Sheets;
DB - Data Base;
DDL - Data Definition Language;
DI - Dependency Injection;
IDE - Integrated Development Environment;
JS - JavaScript;
JSON - JavaScript Object Notation;
GUI - Graphical User Interface;
HTML - HyperText Markup Language;
HTTP - HyperText Transfer Protocol;
MQ - Message Queue;
MVC - Model View Controller;
POM - Project Object Model;
REST - Representational State Transfer;
RMI - Remote Method Invocation;
UDP - User Datagram Protocol;
URL - Uniform Resource Locator;
XML - eXtensible Markup Language.

ВСТУП

Сучасні компанії, працюючі у сфері електронної комерції, мають досить складні взаємозв'язки з різними платіжними системами. Проведення платіжних транзакцій пов'язане з фінансовими ризиками як для компанії, так і для її клієнтів. Тому одним з головних завдань в цій області є мінімізація можливих проблем і прискорення знаходження проблемних місць.

Чим більше у компанії клієнтів і чим більше географічне охоплення, тим складніше завдання аналізу транзакцій в реальному часі. А якщо говорити про великий потік транзакцій, то частенько ручний аналіз і зовсім неможливий. При такій ситуації можна пропустити момент, коли якась з платіжних систем перестала працювати тільки для вузького круга користувачів, наприклад, тільки в одній з країн. У такому разі, загальна картина платежів може змінитися зовсім трохи, а значить і проблема буде виявлена тільки після певного інтервалу часу, наприклад, коли один з користувачів сам звернеться в технічну підтримку.

Подібні проблеми викликають не лише явні фінансові втрати, але і, у тому числі, відтік клієнтів до конкуруючих компаній і інші супутні витрати.

Великий потік транзакцій також накладає обмеження на зберігання історії, по якій можна проводити аналіз. Для зберігання у базі даних потрібно нормалізацію, тобто індексацію за усіма необхідними критеріями для подальшого аналізу в реальному часі частенько просто неможлива.

Проблема збереження і обробки даних, що містять інформацію про платіжні транзакції, що проводяться, для відображення агрегованих метрик на графіках в реальному часі, для аналізу отриманих метрик і генерації сповіщень у разі зареєстрованих відхилень при аналізі є актуальною.

Метою магістерської роботи є дослідження тимчасових рядів, зберігання й обробки даних, що містять інформацію про платіжні транзакції, що проводяться, їх аналізу і можливості сповіщення про нестандартну

поведінку потоку транзакцій та розробка системи моніторингу проведення транзакцій.

Об'єкт дослідження: методи аналізу тимчасових рядів.

Предмет дослідження: засоби розробки систем моніторингу платіжних транзакцій.

Новизна роботи полягає у науковому обґрунтуванні та вирішенні проблеми підвищення якості обробки поведінки потоку транзакцій як інформаційних продуктів шляхом урахування часових залежностей даних..

Для досягнення поставленої мети необхідно вирішити наступні основні завдання:

- Провести аналіз існуючих методів аналізу тимчасових рядів.
- Запропонувати найкращий;
- Дослідити ефективність обраного методу

Розроблена система повинна дозволити автоматизувати складні операції аналізу даних і надати зручний інструмент для персоналу, відповідального за область аналітики.

Система моніторингу транзакцій потрібна для міжнародної компанії, одним з головних аспектів якої є робота з постійним проведенням платіжних транзакцій в країнах по всьому світу, з використанням великої кількості зовнішніх платіжних систем

РОЗДІЛ 1 ДОСЛІДЖЕННЯ МЕТОДІВ АНАЛІЗУ ТИМЧАСОВИХ РЯДІВ, ПОСТАНОВКА ЗАВДАННЯ МОНІТОРИНГУ ПРОВЕДЕННЯ ПЛАТІЖНИХ ТРАНЗАКЦІЙ

1.1 Теоретичні основи методів дослідження тимчасових рядів

Існує комплекс, що надає комерційні послуги для клієнтів і інтегрований з великою кількістю зовнішніх платіжних систем. Використовуючи дані платіжні системи користувачі сервісу проводять платіжні транзакції: зарахування грошей зі свого рахунку у зовнішній платіжній системі на внутрішній рахунок сервісу, переказ з рахунку всередині сервісу на свій особистий рахунок зовнішньої платіжної системи і т.п. При обробці даних транзакцій можливе виникнення будь-яких помилок, через які транзакції можуть завершуватися некоректно або не завершуватися зовсім.

Основною проблемою при пошуку даних помилок є складність в поділі коректної поведінки системи, коли платежі дійсно не повинні відбуватися в силу певної логіки і функціональності яких умов зовнішнього світу, і поведінки, пов'язаного з реальними проблемами в логіці системи, зовнішніх платіжних сервісів, інфраструктури і т. д.

Розділити транзакції на успішні і неуспішні також не представляється можливим, тому що неуспішною вона може бути в силу яких-небудь дійсно існуючих умов (наприклад, у користувача недостатньо грошей на рахунку), так і в силу помилок системи (наприклад, з рахунку клієнта гроші списали, а при обробці даної транзакції сталася непередбачувана внутрішня помилка, яка призвела до ситуації з неможливим відкотом транзакції).

Пошуком і аналізом таких проблем займаються фахівці аналітики, застосовуючи для цього певні алгоритми і евристики. Але для цього аналізу потрібні вже підготовлені дані, на основі яких можна побудувати графік залежності значень застосованих функцій агрегації над даними від часу. Грунтуючись на побудованих графіках, фахівці можуть зробити будь-які

висновки, порівняти з впливом зовнішніх факторів на систему і вирішити, наскільки коректна була її поведінка.

Але, як уже було сказано вище, для такого аналізу потрібні підготовлені і відфільтровані за певними критеріями дані, для отримання яких потрібно виконувати досить складні запити до бази даних, що може займати дуже великий час (в залежності від складності запиту) в силу особливостей і обсягів зберігання даних (нестача індексів, розподіленість БД), що критично для аналізу транзакцій в реальному часі. А іноді підготувати дані для аналізу і зовсім неможливо, якщо потрібні будь-які нестандартні параметри, наприклад, для фільтра по країні, валюті, юрисдикції, типу платіжної системи і провайдеру послуг і т. д.

Саме для цих вирішення цих проблем, в рамках даної роботи, повинна бути розроблена система моніторингу, яка дозволить здійснювати зберігання та обробку даних за платіжними транзакціями, буде відображати в реальному часі графіки на основі функцій агрегації платіжних транзакцій за певними метриками, зможе самостійно здійснювати певний аналіз і оповіщати про «нестандартних ситуаціях» в разі відхилень потоку платіжних транзакцій від розрахункового обсягу. Дана система зможе надати інструмент для фахівців в даній області, що зможе мінімізувати час виявлення проблем, а значить і час їх рішення, що допоможе знизити фінансові втрати компанії і витрати ресурсів.

1.2. Черги повідомлень

Черга повідомлень (message queue, MQ) - це архітектура обміну повідомленнями між компонентами програмних систем в асинхронному режимі, що означає, що повідомлення може бути відправлено одним компонентом системи в один час, а отримано та оброблено другим компонентом-одержувачем в зовсім інше. Складаються такі системи з producer'a (відправника) і consumer'a (одержувача) які взаємодіють між собою

через постачальника (broker), який по-іншому можна визначити, як MQ-сервер. MQ-архітектура показана на рис. 1.1.

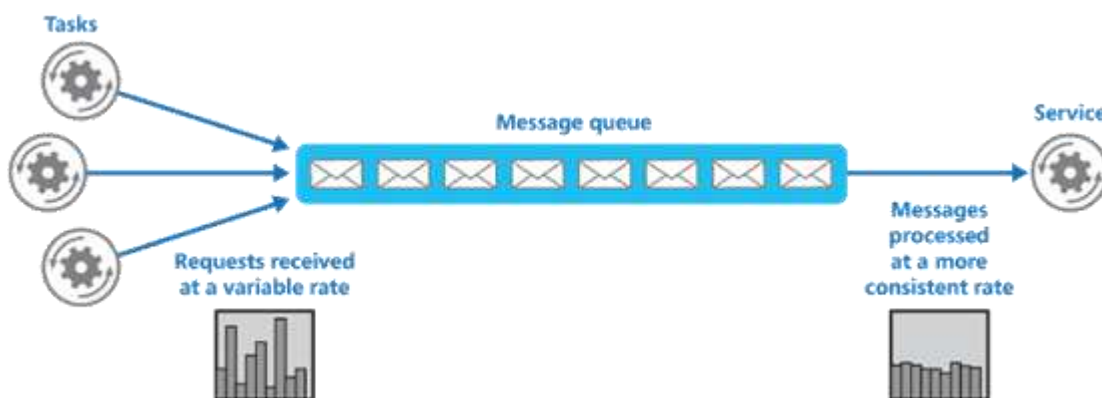


Рис. 1. 1 – Архітектура MQ-взаємодії

Також, слід зазначити, що таке черга (queue):

- Черга - структура даних з доступом до елементів, який можна описати, як «перший прийшов - першим вийшов». Елемент можна додати тільки в кінець черги, а вибірка проводиться з її початку, при отриманні елемента з черги він буде знищений.
- Використання MQ-архітектури має такі особливості:
 - слабке зв'язування - створюються неявні інтерфейси обміну даними, які дозволяють процесам бути незалежними один від одного і зберігати необхідний формат повідомлень;
 - надмірність - можливість уникати можливості нераціонального використання різних ресурсів системи і мережі, зберігаючи необроблену інформацію;
 - масштабованість - можливість збільшувати продуктивність MQ-сервера за допомогою розподілу процесів обробки інформації;
 - еластичність і можливість витримувати пікові навантаження - черги повідомлень можуть служити, в якомусь плані, буфером для накопичення інформації при високому навантаженні на додаток, дозволяючи розтягнути обробку інформації і, тим самим, знизити загальне навантаження на систему;

- відмовостійкість - можливість відділення один від одного процесів додавання і отримання повідомлень, завдяки чому, в разі відмови процесу-обробника, з'являється можливість відкласти обробку повідомлення до відновлення робочого стану системи;
- гарантована доставка - повідомлення будуть доставлені і оброблені в будь-якому випадку, незалежно від робочого стану системи-відправника та системи-одержувача повідомлень (як тільки з'явиться хоча б один обробник), що досягається використанням асинхронного зв'язку і можливістю довготривалого зберігання повідомлення на MQ-сервері, поки воно не буде оброблено;
- гарантований і одноразовий порядок доставки - велика частина MQ-серверів забезпечує доставку даних саме в тому порядку, в якому вони були відправлені, гарантуючи при цьому, що після прочитання повідомлення буде видалено з черги;
- буферизація - гарантія максимальної ефективності доставки повідомлень, яка досягається незалежністю швидкості обробки повідомлень від швидкості їх відправки завдяки використанню черзі - своєрідного буфера між системою-обробником і системою-відправником.

1.3 InfluxDB – база даних для збереження тимчасових рядів

Для початку варто визначити, що таке тимчасові ряди і чим відрізняються бази даних для їх зберігання.

Часовий ряд (англ. Time series) - набір показників (або метрик), що описують значення будь-якого з параметрів певного процесу в певні моменти часу. Кожна з таких метрик обов'язково має час, при якому вона була виміряна, і значення параметра. Використовуючи такий набір (часовий ряд), можна збирати статистику зміни значень параметрів у часі.

Для зберігання подібних метрик існують спеціальні бази даних. Їх головна особливість полягає в оптимізації для зберігання тимчасових рядів за рахунок ефективної індексації даних на основі часу метрик [5].

Головною причиною неефективної роботи реляційних баз даних з тимчасовими рядами полягає в поганій сумісності основ реляційної моделі і природи тимчасового ряду.

Однією з фундаментальних основ реляційної моделі є поняття відносини - неупорядкованого безлічі кортежів [6].

В основі ж часового ряду лежить впорядкованість його елементів. Тому для зберігання подібних метрик існують спеціальні бази даних, особливість яких полягає в оптимізації зберігання тимчасових рядів за рахунок ефективної індексації даних на основі часу метрик. До того ж, в подібних базах даних існують оптимізації на основі припущення про CRUD операціях: основними операціями є вставка і читання даних, а операції зміни і видалення даних досить рідкісні або взагалі не використовуються.

Однією з таких баз даних є «InfluxDB» - база даних з відкритим вихідним кодом від компанії «InfluxData», написана на мові програмування «Go» і використовується для зберігання і аналізу різних часових метрик [7].

Основними особливостями InfluxDB є [8]:

- API, побудований на основі HTTP інтерфейсу;
- Можливість використання тегів;
- Для опису запитів до БД використовується SQL-подібна мова;
- InfluxDB не має зовнішніх залежностей, що дозволяє легко і просто встановити і налаштувати базу даних;
- Оскільки кожна метрика індексується відразу після вставки і відразу є доступною для читання, це дозволяє відповідати на запити за константне час;
- Можливість налаштування автоматичного видалення застарілих даних.

При зберіганні метрик використовуються теги - так звані необов'язкові додаткові дані для кожної з метрик, що дозволяють при запитах до бази даних проводити пошук і порівняння ще й по ним.

Для нашої задачі одним із прикладів метрик можуть служити набори успішних / неуспішних транзакцій, тобто час транзакції і сума транзакції. Крім того, як було сказано в розділі аналізу вимог, нам будуть потрібні додаткові параметри для зберігання, такі як валюта, країна і т.п., які як раз є тегами для InfluxDB.

1.4. Telegram та використання його ботів для оповіщення

Telegram - популярний сервіс для обміну повідомленнями. Крім версії для браузерів існують версії для ОС Linux, Mac і Windows, а також для популярних мобільних ОС, таких як Android, iOS і Windows Phone.

Однією з функціональних можливостей сервісу є можливість створення так званих «ботів» - сторонніх додатків, які можуть взаємодіяти з користувачами, відповідати на будь-які команди від них, а також відправляти їм повідомлення. Наприклад, вже зараз існує величезна безліч ботів для отримання новин, зведень погоди, для створення заміток і нагадувань, для проведення грошових операцій і т.д.



Рис. 1.2 – Приклад інтерфейса клієнта Telegram для платформи iOS.

Повідомлення, команди і запити, надіслані користувачами, передаються на сервер розробників спамерських пошукових роботів. Сервер Telegram є тільки посередником і надає свій API для розробників, доступний за допомогою HTTPS-інтерфейсу, який пропонує спрощену версію API Telegram'a, званий «Bot API» [9].

Для того, щоб почати взаємодіяти з ботом, користувач повинен спочатку додати його до себе в чат або в групу. Або відправляючи команди з чату безпосередньо якомусь боту, використовуючи символ «@» перед його ім'ям в повідомленні з командою.

Детальніше про ботах можна почитати на офіційному сайті Telegram, в даній роботі нас цікавить тільки можливість додавання бота в чат або групу, після чого він зможе надсилати туди оповіщення при будь-яких відхиленнях аналізованих метрик.

Слід уточнити етап створення нового бота і використання API, що надається Telegram.

Згідно з інструкцією з офіційного сайту, для створення бота мінімально потрібно виконати наступні пункти:

- Додати бота «BotFather» до себе в чат, який дозволяє управляти виробництвом і налаштуванням нових ботів всередині Telegram;
- Надіслати боту команду «/ newbot», після чого той запросить повне і коротке ім'я створюваного бота (повне ім'я використовується для відображення користувачеві назви бота в заголовку чату, коротке ім'я - для посилань і швидких команд) і згенерує унікальний токен авторизації, який буде потрібен для ідентифікації бота і взаємодії через надається Telegram API при відправці запитів.

Після виконання описаних вище пунктів бот буде створений і доступний для використання і додавання в чати до користувачів.

Для більш докладної настройки існують і інші команди бота «BotFather», які можна дізнатися, виконавши команду «/ start», ми це не будемо розглядати.

1.5. Аналіз метрик та функцій агрегації

Так як в роботі розглядаються метрики, їх аналіз та функції агрегації, що будуються на метриках, слід докладніше зупинитися на них.

Метрика - якась величина, як уже говорилося, що характеризує параметр системи в певний момент часу. Метрику можна представити у вигляді пари ключ-значення, де ключем буде час.

Функції агрегації, це вид математичних функцій, що застосовуються до безлічі значень, результатом виконання яких є єдине значення для всієї множини. Якщо ми говоримо про функції агрегації, застосованої до тимчасового ряду метрик, результатом її виконання можна назвати «згортання» значень метрик на інтервалі часу в одне [10].

Функції агрегації використовуються в багатьох сферах ПО, наприклад, в теорії баз даних можна навести приклади таких функцій агрегації, так чи інакше використовуються в багатьох СУБД: [11]

- AVG - середнє арифметичне значення стовпця;
- COUNT - підрахунок кількості значень;
- GROUP_CONCAT - конкатенація значень стовпця;
- MIN - мінімальне значення стовпця;
- MAX - максимальне значення стовпця;
- SUM - сума всіх значень стовпця;
- STD - середньоквадратичне відхилення значень стовпця;
- VARIANCE - розбіжність (розкид) значень стовпця.

Для аналізу тимчасових метрик застосовуються різні методи, розглянемо деякі з них [12].

Перший метод використовує так звані коридори прогнозованих значень, які можна відобразити на графіку, наведеному на рис. 1.3, де:

- Ось абсцис, t - час вимірювань;
- Вісь ординат, v - значення параметра системи;
- t_0 - поточний час;

- t_1 - час останньої відомої метрики;
- крива, позначена синім кольором, - умовне позначення змінюється параметра системи в часі;
- дві прямі лінії - умовне позначення верхньої і нижньої межі «нормальних значень», розраховані за певними математичними алгоритмами (тому що це завдання математики та дані модулі розробляються окремо від системи моніторингу, ми це не розглядаємо в даній роботі) на інтервалі часу від початку графіка до t_0 .

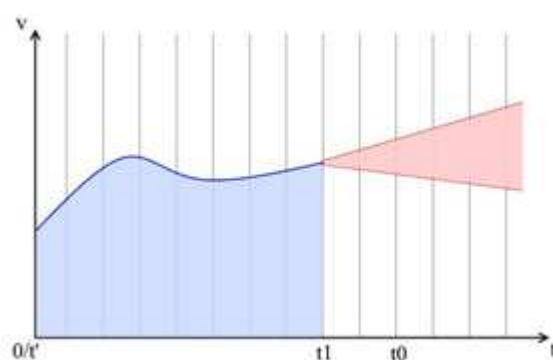


Рис. 1.3 –Графік розподілення часових міток та коридору прогнозуємих значень

При отриманні метрики для точки t_0 і зміщення отриманого значення за межі коридору «нормальних значень», вважається, що поведінка системи відрізняється від нормального і приймаються дії до розслідування і усунення неполадки, якщо такі є, або рішення, що ситуація коректна і будь-які дії не потрібні.

Другий метод ґрунтується на розподілі Пуассона. Для початку варто зупинитися на визначенні. Розподіл Пуассона - імовірнісний розподіл дискретного типу, моделює випадкову величину, яка була число подій, що сталися за фіксований час, за умови, що дані події відбуваються з деякою фіксованою середньою інтенсивністю і незалежно один від одного [13]. Іншими словами, використовуючи розподіл Пуассона можна обчислити розподіл ймовірностей для подій, які:

- відбуваються з часовим інтервалом;

- дискретні;
- не залежать одне від одного.

Для нашої задачі використання розподілу Пуассона (1.1) дає можливість спрогнозувати поведінку процесів, пов'язаних з проведенням платіжних транзакцій. Побудувавши імовірнісний розподіл на основі відомого розподілу подій (метрик) на часовому інтервалі, можна припустити час наступної події. При значних відхиленнях прогнозованого часу від фактичного будемо вважати, що поведінка програмної системи відрізняється від нормального і робити які-небудь заходи.

$$P_m \approx \frac{\lambda^m}{m!} \cdot e^{-\lambda}, \text{ де } \lambda = np \quad (1.1)$$

Наприклад, маючи стабільний потік платіжних транзакцій і побудувавши на його основі Пуассоновим розподіл ймовірностей, ми зможемо розрахувати, що наступна транзакція повинна відбутися не більше, ніж через 5 секунд, якщо через 5 секунд транзакція не буде зареєстрована системою, генерувати сповіщення про те, що поведінка системи змінилося, і відсилати його зацікавленим особам. Самі математичні розрахунки, як уже говорилося вище, в даній роботі розглядатися не будуть.

1.6 Використовувані технології

Вимоги, висунуті до використовуваних технологій, в основному залежать від вже використовуваних в компанії технологій і рішень. Оскільки розробляється в даній роботі система моніторингу буде лише маленькою частиною програмної системи, для забезпечення максимальної сумісності, простоти розробки і підтримки слід дотримуватися вже сформованих корпоративних правил і стандартів. Виходячи з цього, були висунуті наступні вимоги до технологій:

- Для передачі метрик від програмної системи в моніторинг повинен використовуватися брокер черг повідомлень «RabbitMQ»;

- При використанні бази даних для зберігання тимчасових поруч повинна використовуватися «Influx Database»;
- При використанні реляційної бази даних, вибір повинен бути відданий на користь «Oracle Database»;
- Розробка серверної частини повинна проводитися на мові Java з використанням, при необхідності, Spring Framework (особливо, його модулів: «Spring Boot», «Spring Remote» і «Spring MVC»);
 - Взаємодія між модулями через RMI;
 - Збірка повинна проводитися з використанням «Maven»;
 - Для розгортання веб-додатки повинен використовуватися контейнер «Apache Tomcat»;
- При розробці веб-інтерфейсу пріоритет повинен бути відданий таким технологіям:
 - о «Twitter Bootstrap» для візуального оформлення;
 - о «JavaScript» і «AngularJS» для розробки динамічної частини;
 - о «Google Charts» для відображення графіків

1.7 Аналітичний огляд існуючих рішень

На етапі аналітики і формування вимог до розроблюваної системи було вирішено, що жоден з існуючих аналогів ПО не підходить під вимоги. Зупинимося на цьому докладніше.

Основними вимогами були:

- Зберігання метрик в time-series базі даних тільки в «сирому» вигляді (під «сирим» видом ми розуміємо оригінальні показники часу, кількості і т.п., в тому вигляді, в якому вони були доставлені в розроблювану систему моніторингу, на які ще не були накладені ніякі математичні функції), тому що аналізувати метрики передбачається в реальному часі;
- Можливість в реальному часі накладати математичні функції, тобто функції агрегації, на «сирі» метрики, зчитуються з бази даних;

- Відображення графіків з вибором функції агрегації, історією в часі, з вибором відображуваного історичного періоду і розмірності на екрані;
- Можливість задавати пороги критичних значень і відхилень метрик індивідуально для кожного графіка, при досягненні яких будуть спрацьовувати і відображатися так звані «Алерт», тобто оповіщення;
- Можливість додати оповіщення по спрацьовує «Алерт», які будуть надсилатися за передплатою, як мінімум, на електронну пошту і в Telegram-канал. Сповіщення повинні доставлятися, як при досягненні критичної позначки певної метрики, так і при нормалізації ситуації;
- Якщо ПЗ буде стороннім, у нього повинна здійснюватися підтримка, тобто при виникненні помилок в його роботі, вони повинні бути виправлені або силами розробників з боку компанії, що надає ПО, або власними силами (у такому разі, ПЗ повинно бути з відкритим вихідним кодом і актуальною технічною документацією).

На поточний момент, існує не так багато рішень, що дозволяють покрити хоча б частину функціоналу. Розглянемо відповідні рішення для висунутих вимог і відзначимо плюси і мінуси для нас:

- «Zabbix» від Zabbix SIA;
- «Grafana» від Grafana Labs;
- «Nagios» від Nagios Enterprises;
- «Prometheus» від Soundcloud;

«Zabbix» є, мабуть, найпоширенішим і функціональним безкоштовним рішенням з існуючих на ринку. До того ж, він вже використовується всередині компанії для моніторингу інших сервісів.

Плюси [1]:

- Безкоштовний;
- Багато підтримуваних джерел метрик, в тому числі, їм може бути time-series база даних, така як «InfluxDb»;
- Для доставки метрик в сервіс можна використовувати широкий програмний інтерфейс, в тому числі, HTTP REST;

- Підтримує пакетування запитів при відправці;
- Є платна технічна підтримка;
- Є баг-трекер з можливістю відслідковувати прогрес по заведеним завданням на виправлення помилок.

Мінуси:

- Для відображення метрик і сповіщень по ним в базі вже повинні зберігатися оброблені за допомогою функцій агрегації метрики, що вже суперечить основним вимогу;
- Оповіщень по «Алерт» в Telegram-канал немає, потрібно дописувати окремий функціонал для цього;
- Не можна організувати поділ метрик за тегами;
- Налаштування відображення графіків дуже складна і обмежена.

Висновок: «Zabbix» не підходить, тому що, як мінімум, для роботи з метриками і їх відображення він вже повинен мати в своїй базі даних «підготовлені» метрики.

«Grafana» є другим рішенням для моніторингу, що вже використовуються компанією. Використовується тільки для візуалізації підготовлених даних, що зберігаються в time-series базі даних.

Плюси [2]:

- Зручний і інтуїтивно зрозумілий інтерфейс;
- Гнучкість в налаштуванні графіків;
- Можливість групування метрик за тегами;
- Найпоширеніші функції агрегації вбудовані в програму і доступні для застосування безпосередньо перед відображенням даних на графіку;
- Стабільно виходять оновлення і виправлення знайдених помилок.

Мінуси:

- Хоч функції агрегації і доступні прямо поверх «сирих» даних перед візуалізацією метрик, але можливість додавання нових функцій обмежена. Неможливо додати складні математичні перетворення, тобто знову потрібно

зберігання в базі даних вже підготовлених метрик для складних функцій агрегації.

- Повністю відсутня функціонал сповіщень;
- Чи не реалізовано розмежування прав доступу - абсолютно будь-який користувач системи має доступ до даних (графіки, інформаційні панелі) і може їх змінити.

Висновок: «Grafana» підходить більше, ніж «Zabbix», але цього все одно недостатньо: потрібно написання стороннього сервісу, який буде аналізувати метрики і оповіщати при «Алерт». Крім цього, знову потрібно «підготовка» метрик і зберігання в базі даних перед відображенням їх на графіку.

«Nagios» - платний, ціни починаються від +1995 \$ / рік за мінімальну функціональність [3].

Плюси:

- Можливість роботи з різними підключаються плагінами, що розширюють базовий функціонал;
- Є розширені можливості сповіщень при відхиленнях показників від «норми»;
- Є платна / безкоштовна підтримка.

Мінуси:

- Для комерційного використання є тільки платна версія, ціни починаються від тисяча дев'яносто дев'яносто п'ять \$ / рік за мінімальну функціональність

Відсутність вбудованих засобів візуалізації ;

- Відсутність можливості угруповання за тегами;
- Функції агрегації застосовуються за таймером, після чого розраховані метрики зберігаються в базі даних;
- При будь-якій зміні конфігурації потрібно перезапуск сервісу;
- Якщо потрібно відстежувати те, що «Nagios» не вміє, потрібна серйозна доробка і написання додаткових сервісів прошарок;

- Всі перевірки збираються в живу чергу і виконуються в строгій черговості.

Це накладає певні обмеження на масштабованість - сервер утримає велика кількість перевірок без праці, але чим їх більше, тим більше часу проходить між ними.

Висновок: «Nagios» точно так само вимагає зберігання оброблених метрик, дуже складний в конфігурації і вимагає постійних перезапусків сервісу, вимагає окремо встановленого ПО для візуалізації даних, повідомлень.

Мінусів набагато більше, ніж у «Zabbix» і «Grafana», вимагає серйозного доопрацювання, що недоцільно, тим більше, при платній ліцензії.

«Prometheus» - повноцінна система моніторингу.

Плюси [4]:

- Безкоштовний;
- Є мінімальна система «алертів», описувана правилами виду «якщо значення менше X, надсилати сповіщення»;
- Є вбудована візуалізація метрик з поділом за тегами;
- Для візуалізації графіків можлива інтеграція з «Grafana».

Мінуси:

- «Prometheus» сам отримує метрики від сервісів, значить для цього йому потрібна прошивка до кожного сервісу, а крім цього, вже на цьому етапі потрібні підготовлені і агреговані метрики;
- Зберігається вся історія, а не тільки «сирі» дані;
- Система сповіщення не має права продовжувати;
- Відсутність інформації про будь-яку підтримку.

Висновок: використання «Prometheus» не підходить по більшості пунктів.

1.8 Аналіз вимог до системи моніторингу проведення платіжних транзакцій

На етапі аналітики і формування вимог до системи, що розробляється, було вирішено, що жоден з існуючих аналогів ПЗ не підходить під вимоги. Зупинимось на цьому детальніше. Так як систему, що розробляється, планується впроваджувати в уже існуючий програмний комплекс, це накладає певні обмеження на архітектуру і використання технології, що також буде розглянуто в цьому розділі.

Виходячи з поставлених вимог, потрібна розробка системи, яка:

- Буде приймати дані про проведені платіжних транзакціях за певним протоколу;
- Буде зберігати отримані дані в спеціалізованій БД, оптимізованої для зберігання метрик і часових рядів в цілому;
- Буде певним чином зберігати налаштування, структуру відображаються графіків;
- Чи дозволить підключати і використовувати різні функції агрегації, що розробляються окремо від даної системи;
- Чи дозволить використовувати один з методів аналізу потоку транзакцій для можливості оповіщення в разі будь-яких відхилень;
- Буде мати можливість оповіщення на кілька різних каналів зв'язку;
- Буде надавати графічний веб-інтерфейс з можливістю перегляду графіків в реальному часі.

Тепер розглянемо детальніше кожен з описаних вище вимог і опишемо його деталізацію.

Прийом здійснених транзакцій здійснюватиметься з черги повідомлень, брокером якої є «RabbitMQ», дані матимуть формат JSON і міститимуть наступні обов'язкові параметри:

- Час, в якому була вироблена транзакція;
- Сума транзакції;

- Валюта транзакції.

Окрім цього, в кожному повідомленні можуть міститися необов'язкові параметри у форматі ключ-значення. Функціонал по прийому даних з черги повинен міститися в окремому модулі системи, що розробляється

Для зберігання прийнятих з черги транзакцій має бути використана time - series БД, логіка роботи з якою так само міститиметься в окремому модулі системи.

До зберігання налаштувань, графіків і так далі пред'являються наступні вимоги:

- На одному графіку може відображатися відразу декілька залежностей метрик від часу, тобто декілька графічних ліній;

- Має бути передбачене редагування графіків з наступними можливостями:

- Додавати нові;
- Видаляти старі;
- Змінювати існуючі;
- Додавати нові теги(по яких фільтруються графіки);
- Видаляти теги;
- Змінювати теги;
- Додавати лінії на графік(лінії будуються на основі тегів);
- Видаляти лінії з графіку;
- Вибирати функції агрегації для кожного з графіків.

Сервіс для зберігання усіх налаштувань так само повинен розроблятися, як окремий модуль додатка.

Функції агрегації розроблятимуться окремо від усього застосування, треба спроектувати і створити загальний для них інтерфейс, використовуючи який вони однаково підключатимуться до сервісу моніторингу, для цього треба створити окремий модуль додатка, який вирішуватиме саме цю задачу.

Аналіз потоку транзакцій повинен використати розподіл Пуассона для прогнозування часу подальших транзакцій, при будь-яких відхиленнях, як і при нормалізації ситуації, мають бути створені сповіщення. При цьому сповіщення повинні відсилатися в спеціально створену для цього чергу в RabbitMQ. Розрахунок повинен робитися для кожного із спеціально створених для цього шаблонів сповіщень. Для цього завдання, так само, потрібний окремий модуль додатка.

Сповіщення повинні відсилатися по наступних каналах зв'язки:

- По електронній пошті, у вигляді листа на адресу тих, що підписалися на конкретне сповіщення користувачам;
- У месенджер (сервіс обміну текстовими повідомленнями) «Telegram».

Модуль розсилки сповіщень має бути розроблений окремо.

Веб-інтерфейс повинен мати можливість відображення графіків з наступними параметрами:

- Змінюваним інтервалом часу (, що відображається, 30 хвилин і 1, 2, 3, 6, 24 години);
- Можливістю вибору часу і дати початку графіків(для можливості відображення історичних даних).

Окрім цього, у нього має бути можливість редагування налаштувань і перегляду сповіщень, що спрацювали. Для надання даних графічному інтерфейсу має бути розроблений спеціальний «прошарок» у вигляді окремого модуля додатка. Для зниження навантаження на БД, цей модуль повинен уміти кешувати прошені дані.

Усі описані вище модулі додатка, при необхідності, повинні використати RMI, як інтерфейс комунікації між собою.

Модулі повинні запускатися незалежно.

Уся системна конфігурація повинна зберігатися в окремій теці в текстових файлах з розширенням «properties».

Розроблений сервіс моніторингу, при реплікації даних з черги «RabbitMQ» в time - series БД, повинен забезпечувати пропускну спроможність до 50 повідомлень в секунду.

Для візуалізації призначених для користувача вимог можна побудувати діаграму варіантів використання.

Сама система робить перевірку метрик, що приходять в неї, по певних алгоритмах, і, у разі потреби, розсилає сповіщення на електронну пошту і в Telegram.

Основними вимогами були:

- Зберігання метрик в time - series базі даних тільки в «сирому» вигляді(під «сирим» виглядом ми розуміємо оригінальні показники часу, кількості і так далі, в тому вигляді, в якому вони були доставлені в систему моніторингу, що розроблялася, на які ще не були накладені ніякі математичні функції), оскільки аналізувати метрики передбачається в реальному часі;

- Можливість в реальному часі накладати математичні функції, тобто функції агрегації, на «сирі» метрики, що зчитуються з бази даних;

- Відображення графіків з вибором функції агрегації, історією в часі, з вибором історичного періоду, що відображається, і розмірності на екрані;

- Можливість задавати пороги критичних значень і відхилень метрик індивідуально для кожного графіку, досягши яких спрацьовуватимуть і відображатимуться так звані «Алерти», тобто сповіщення;

- Можливість додати сповіщення по тих, що спрацьовують «алертам», які відсилатимуться по підписці, як мінімум, на електронну пошту і в Telegram- канал. Сповіщення повинні доставлятися, як досягши критичного значення певної метрики, так і при нормалізації ситуації;

- Якщо ПЗ буде стороннім, у нього повинна здійснюватися підтримка, тобто при виникненні помилок в його роботі, вони мають бути виправлені або силами розробників з боку компанії, що надає ПЗ, або

власними силами(у такому разі, ПЗ повинно бути з відкритим початковим кодом і актуальною технічною документацією).

Система моніторингу планується до використання компанією увесь подальший час, при цьому можливе подальше розширення функціональності системи до не розглянутих при первинному аналізі вимог, будь-які складнощі при використанні існуючих рішень приведуть або до серйозних втрат в ресурсах/часу на розробку і пошук рішень виникаючих проблем, або до зміни використовуваного програмного забезпечення, що обійдеться в засоби, неспіввимірні з вартістю поточної розробки.

РОЗДІЛ 2 РОЗРОБКА ПРОЕКТНИХ РІШЕНЬ

2.1 Рішення з інформаційного забезпечення

Проектування - невід'ємна частина розробки будь-якої програмної системи. Умовно, цей процес можна розділити на декілька послідовних, пов'язаних один з одним етапів: формулювання цілей (збір вимог), аналіз предметної області і створення прототипу системи.

Аналіз предметної області має на увазі глибше вивчення вимог, що пред'являються до функціонала проектованої системи, опрацювання сценаріїв взаємодії з системою і алгоритмів у бізнес логіки.

Чітко сформульовані цілі і завдання, ретельно проведений аналіз вимог, опрацювання сценаріїв взаємодії з системою і проектування бізнес-логіки - усе це ключ успіху в досягненні необхідного результату.

Таким чином, проектування збільшує шанси успішної здачі проекту, а також економить час і гроші, адже внесення змін на початковому етапі розробки не так критично, як, наприклад, на етапі відладки вже розробленої системи.

2.1.1 Вибір засобів моделювання

Для побудови складної інформаційної системи необхідним етапом є проектування. В останнє десятиліття в комп'ютерному світі намітилася тенденція проектування систем візуальними (наочними) моделями.

Головна особливість індустрії сучасних програмних комплексів складається в концентрації складності на початкових етапах аналізу вимог і проектування специфікацій при відносно невисокій складності і трудомісткості наступних етапів. Фактично тут і приходить розуміння того, що буде робити майбутня програма і яким чином вона буде функціонувати, щоб задовольняти пред'явленим їй вимогам. Саме нечіткість і неповнота системних вимог, невирішені питання і помилки, допущені на етапах аналізу

та проектування, породжують на наступних етапах важкі, часто нерозв'язні проблеми і, в кінцевому підсумку, призводять до неуспіху всієї роботи в цілому.

Для подолання труднощів початкових етапів розробки, насамперед, і призначений структурний аналіз – метод дослідження, який починається із загального огляду системи і потім деталізується, набуваючи ієрархічну структуру з дедалі більшим числом рівнів. Для таких методів характерна розбивка на рівні абстракції з обмеженням числа елементів на кожному з рівнів (зазвичай від 3 до 6–7), обмежений контекст, який включає лише важливі на кожному рівні деталі, використання строгих формальних правил записи, і як наслідок - послідовне наближення до кінцевого результату.

У цей час відомо близько 90 різновидів структурного системного аналізу, які можуть бути класифіковані за призначенням (для моделювання програмних систем або систем взагалі), за порядком побудови моделі (декларують первинність функціонального або інформаційного моделювання), за типом цільових систем (інформаційні системи або системи реального часу). Незважаючи на різноманітність методів, практично у всіх з них використовуються три групи засобів:

- а) DFD (Data Flow Diagrams) – діаграми потоків даних або SADT-діаграми, які ілюструють функції, які система повинна виконувати;
- б) ERD (Entity Relationship Diagrams) – діаграми "сутність-зв'язок", що моделюють відносини між даними;
- в) STD (State Transition Diagrams) – діаграми переходів станів, які моделюють залежне від часу звернення системи (аспекти реального часу).

Перераховані моделі в сукупності дають повний опис системи незалежно від того, є вона розробленою або розробляється.

2.1.2 Розробка архітектури підсистеми

Для візуалізації призначених для користувача вимог можна побудувати діаграму варіантів використання.

На представлений на рисунку 2.1 діаграмі варіантів використання були показані основні сценарії використання, ґрунтуючись на яких ми можемо спроектувати систему, що розробляється.

Діаграма варіантів використання(Use Case Diagram) призначена, в першу чергу, для того, щоб визначити функціональні вимоги до системи.

Під час аналізу і проектування варіанти використання дозволяють зіставити очікувані результати і їх вплив на архітектуру системи, а також зрозуміти поведінку системи при реалізації необхідної функціональності.

А на етапі тестування з'являється можливість точніше оцінити відповідність розробленої функціональності поставленим вимогам.

Діаграма варіантів використання складається з акторів, а також дій, які робить актор або система.

Актори позначаються значком людини, а дії - овалом.

У системі є три актори: аналітик, адміністратор і черга повідомлень «RabbitMQ».

Аналітик може працювати з графіками, редагувати їх, видаляти, переглядати і так далі.

Окрім цього, він може редагувати відстежувані сповіщення і дивитися історію по них.

Адміністратор займається безпосередньо налаштуванням самого середовища для правильної роботи застосування, що розробляється.

Черга повідомлень «RabbitMQ» забезпечує потік даних для моніторингу.

Сама система робить перевірку метрик, що приходять в неї, по певних алгоритмах, і, у разі потреби, розсилає сповіщення на електронну пошту і в Telegram.

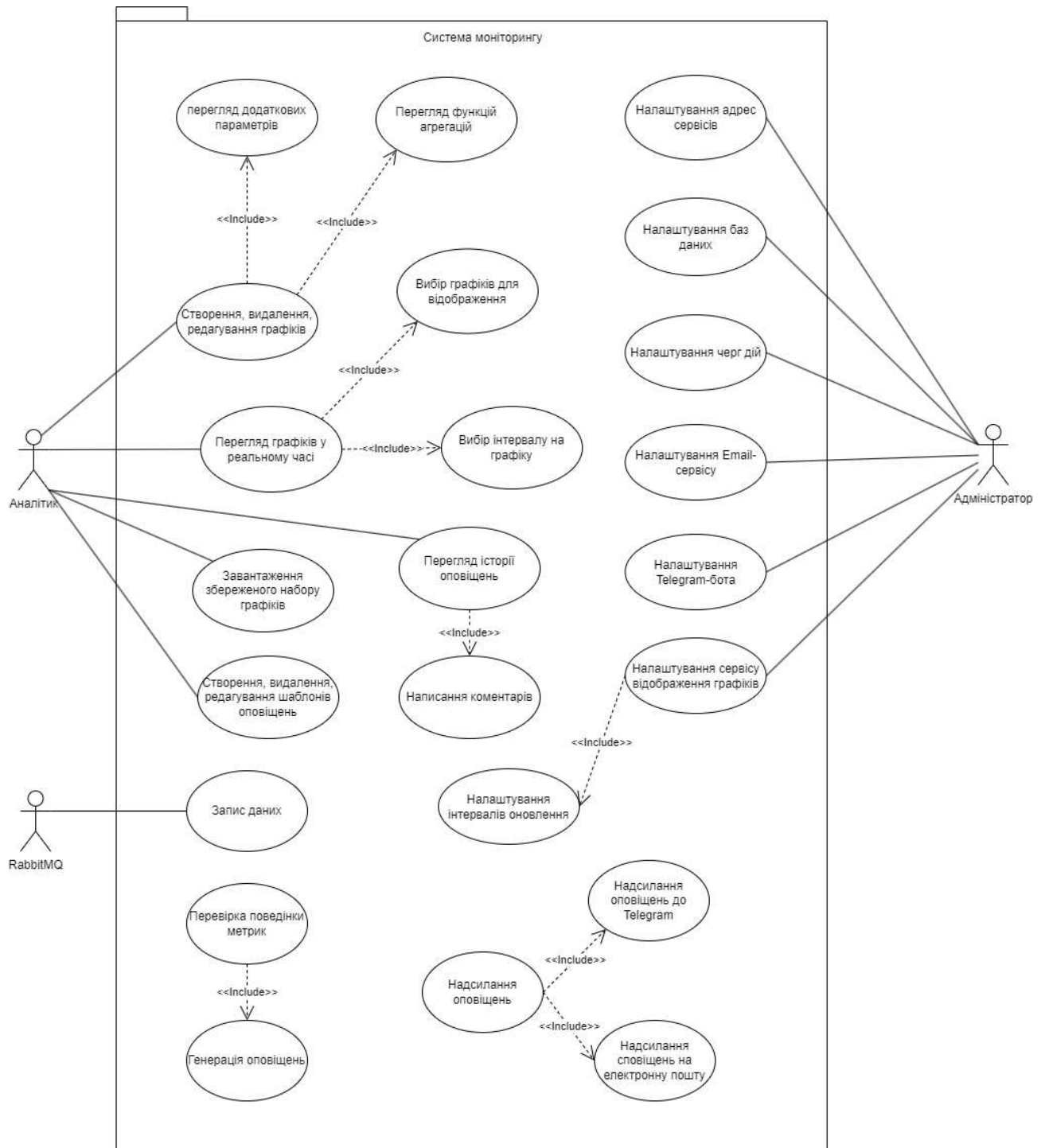


Рис. 2.1 – Діаграма варіантів використання

Перед тим, як перейти до основного етапу проектування програмної системи, вимагається розглянути існуючу архітектуру і визначити місце для компонента, що розробляється.

Існуюча архітектура представлена діаграмою компонентів на рис. 2.2.

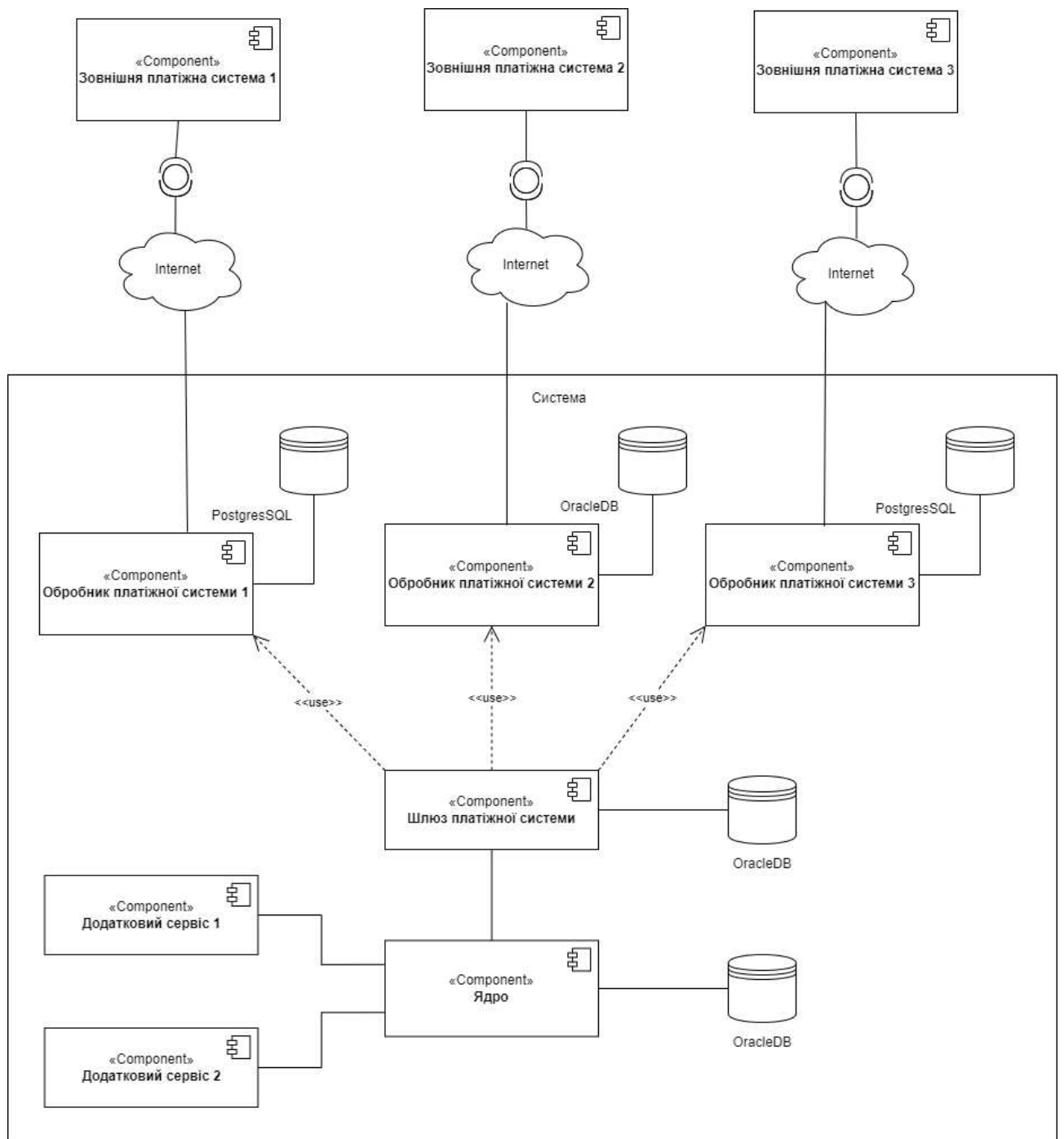


Рис. 2.2 – Діаграма компонентів існуючого програмного комплексу

На приведеній вище діаграмі показані наступні вузли:

- Ядро системи, що інкапсулює в собі усю основну логіку роботи;
- Шлюз платіжних систем, що оперує усіма процесами по обробці платіжних транзакцій;

- Обробники для кожної із зовнішніх платіжних систем, які інкапсують в собі усю логіку по роботі з API кожної конкретної платіжної системи;
- Зовнішні платіжні системи - зовнішні сервіси надаючи послуги з переказу платіжних коштів з одного рахунку на інший і працюючі по певному API;
- Додаткові сервіси - сервіси програмного комплексу, які не мають значення для цієї роботи;
- Різні БД, які(зазвичай) є у кожного з сервісів програмного комплексу.

При такій архітектурі, загальна інформація про проведені платіжні транзакції зберігається у БД, що відноситься до ядра системи, і саме по ній будуються аналітичні звіти.

Загальну архітектуру програмного комплексу після додавання нової системи моніторингу можна представити оновленою діаграмою компонентів, зображеною на Рис. 2.3.

Як видно з приведеної діаграми, модуль моніторингу має бути представлений наступними компонентами:

- Модуль для підключення до черги повідомлень;
- Модуль для підключення до бази даних InfluxDB;
- Модуль для підключення до реляційної бази даних Oracle;
- Модуль для зберігання і зміни графіків і налаштувань;
- Модуль для обробки даних для подальшої побудови графіків і сповіщень;
- Модуль для відправки e - mail сповіщень;
- Модуль для відправки повідомлень в Telegram;
- Модуль підготовки даних для відображення графічного вебінтерфейса.

Дані про проведені транзакції складатимуться в черги RabbitMQ ядром системи, після чого до них матиме доступ модуль підключення до RabbitMQ. Окрім цього, цей модуль надаватиме можливість збереження і читання даних про сповіщення в черзі повідомлення для тих модулів, яким це необхідно.

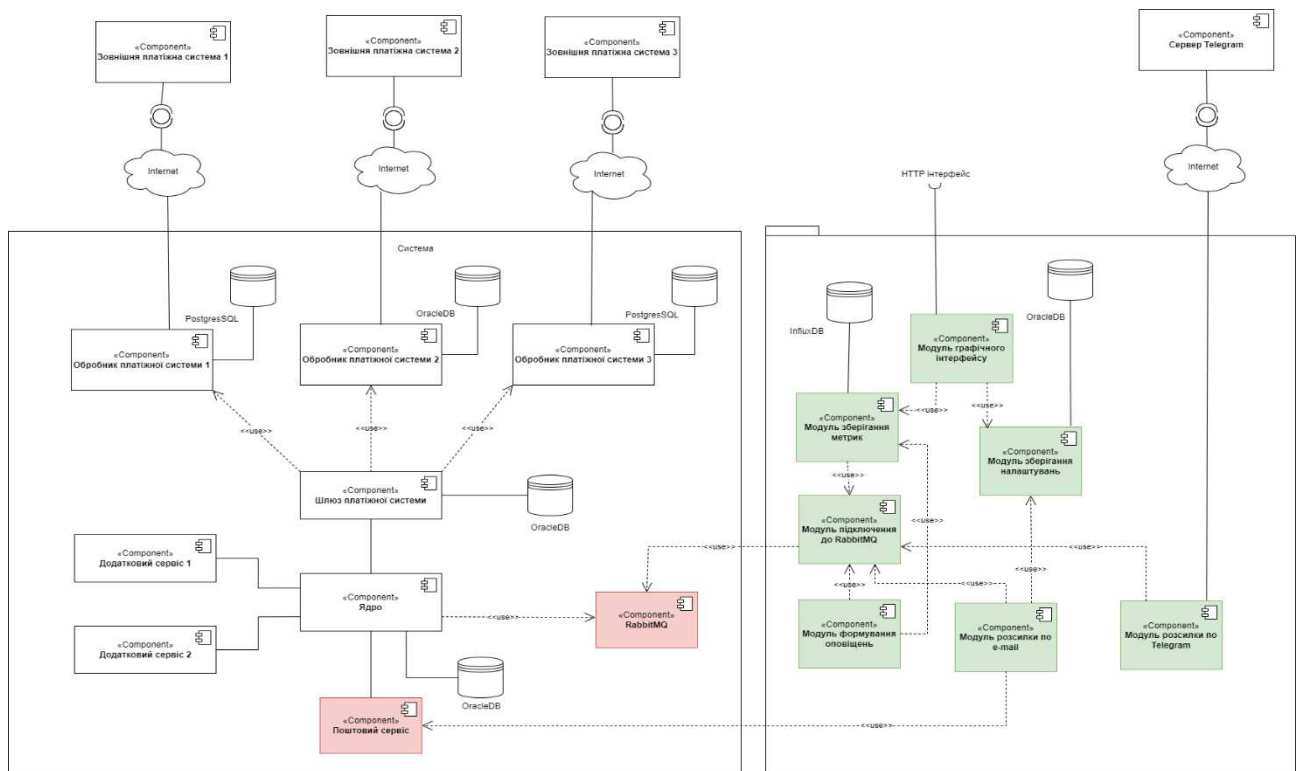


Рис.2.3 – Діаграма компонентів програмного комплексу після додавання необхідного сервісу

Для розсилки сповіщень по електронній пошті використовуватиметься вже розроблений і використовуваний сервіс, що входить до складу програмного комплексу

Крім усього іншого, з'являться два зовнішні інтерфейси: веб, для доступу з браузера, і інтерфейс обміну даними з сервером Telegram.

2.2 Рішення з інформаційної системи

Для детальнішого опрацювання системної архітектури, слід розглянути основні процеси, що відбуваються в системі :

- Процес налаштування відображення і оновлення графіків;
- Процес вибірки даних для графіків;
- Процес додавання нового графіку;
- Процес реплікації даних з RabbitMQ в InfluxDB;
- Процес розрахунку максимального часу очікування наступної транзакції;
- Процес генерації і відправки сповіщень.

Для більшої наочності, були побудовані діаграми послідовності для кожного з цих алгоритмів. Діаграми приведені на рис. 2.4.-2.10 починаючи з наступної сторінки. Діаграми не вимагають яких-небудь доповнень, оскільки розроблені досить детально.

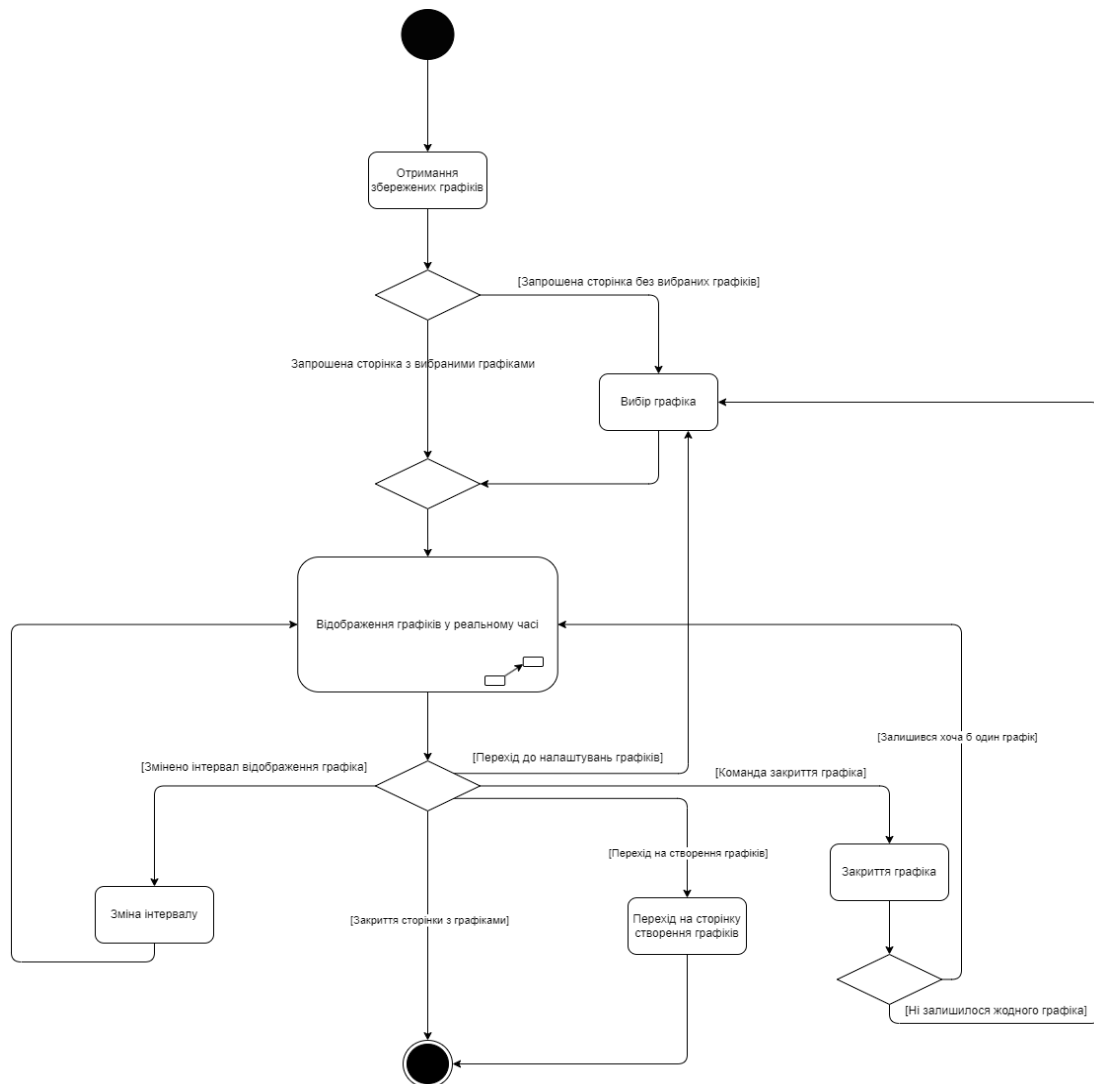


Рис.2.4. – Діаграма діяльності для процесу налаштування відображення графіків у веб-інтерфейсі

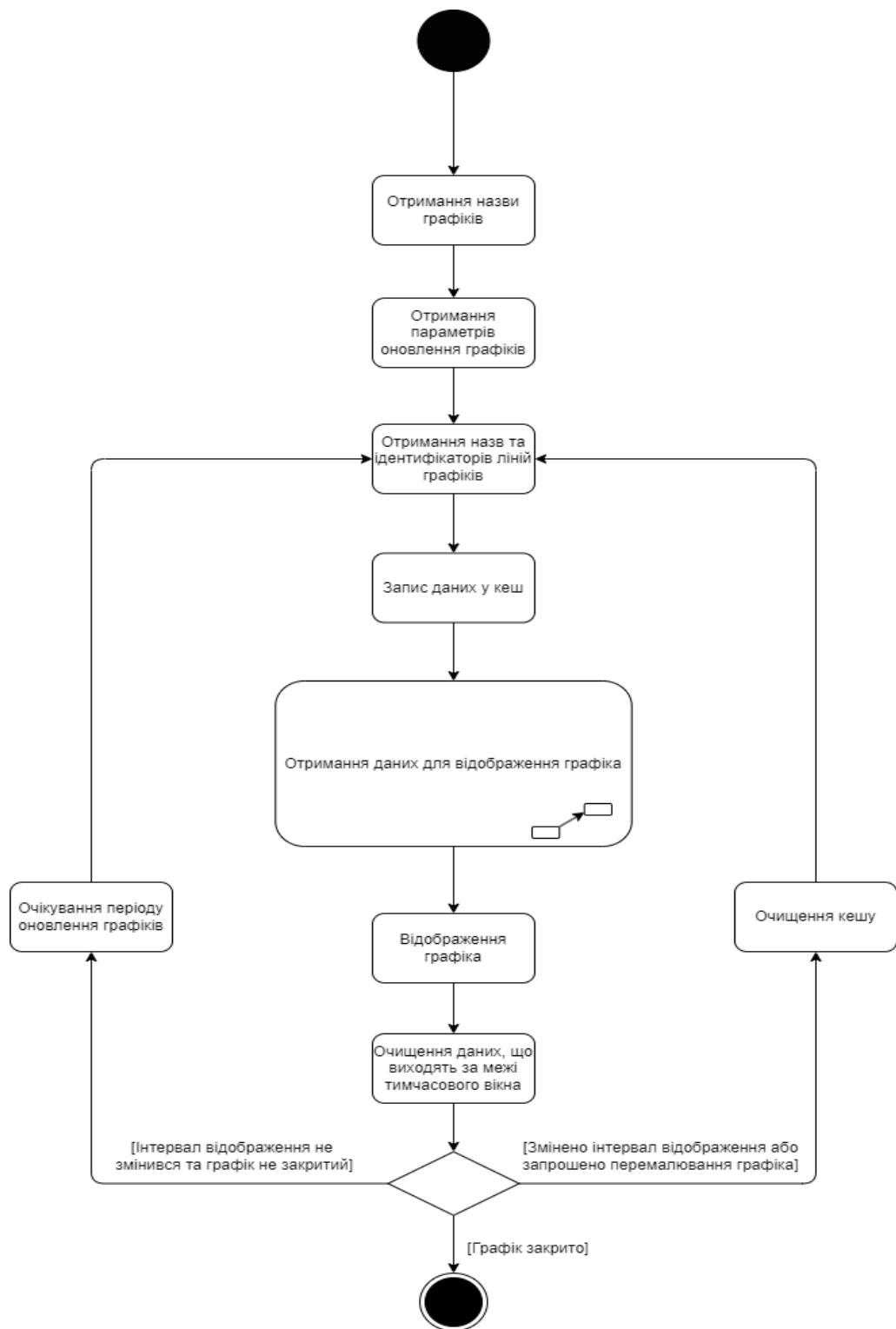


Рис.2.5 – Деталізована діаграма діяльності для процесу відображення і оновлення графіків

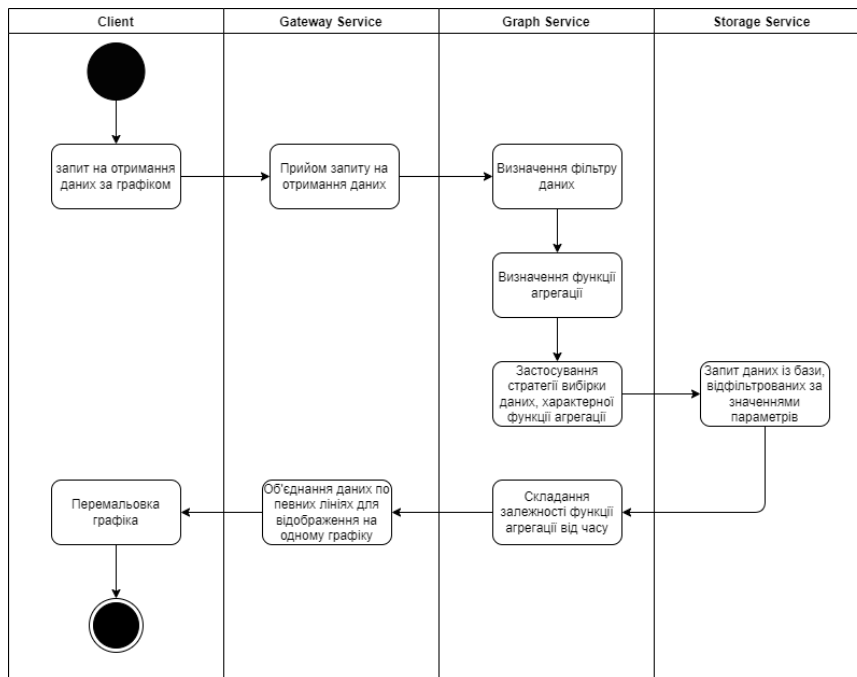


Рис 2.6 – Деталізована діаграма діяльності для процесу отримання вибірки даних для графіків

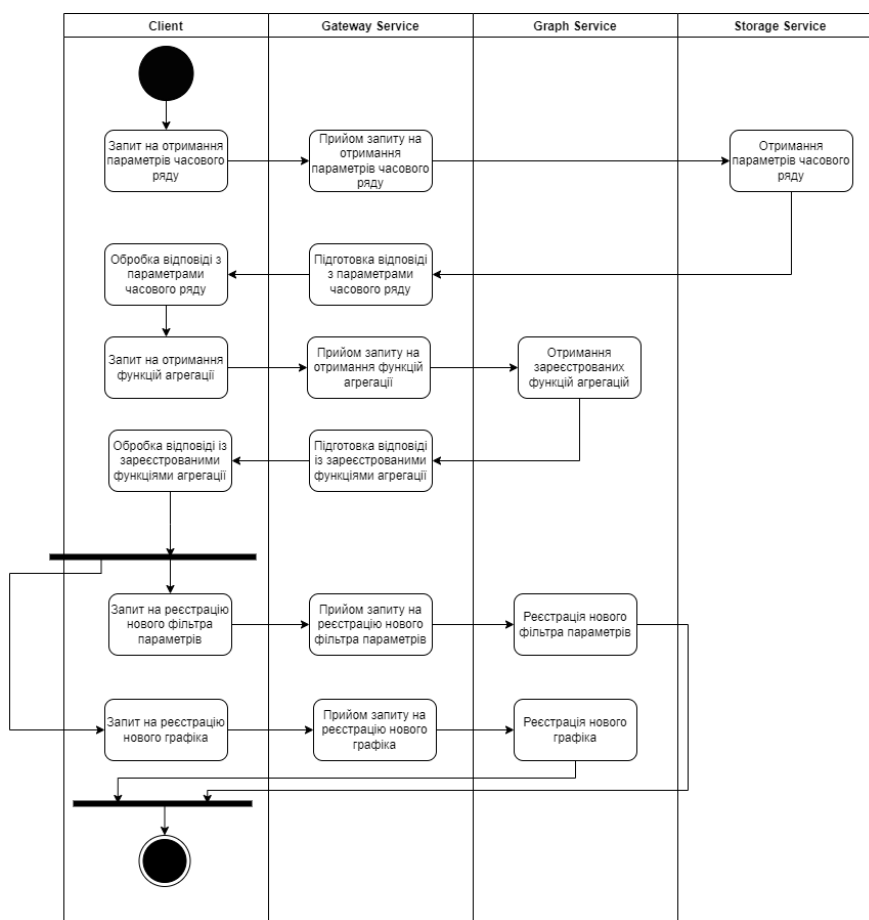


Рис 2.7. – Діаграма діяльності для процесу створення нового графіку



Рис 2.8 – Діаграма діяльності для процесу реплікації даних з черги повідомлень у базу даних

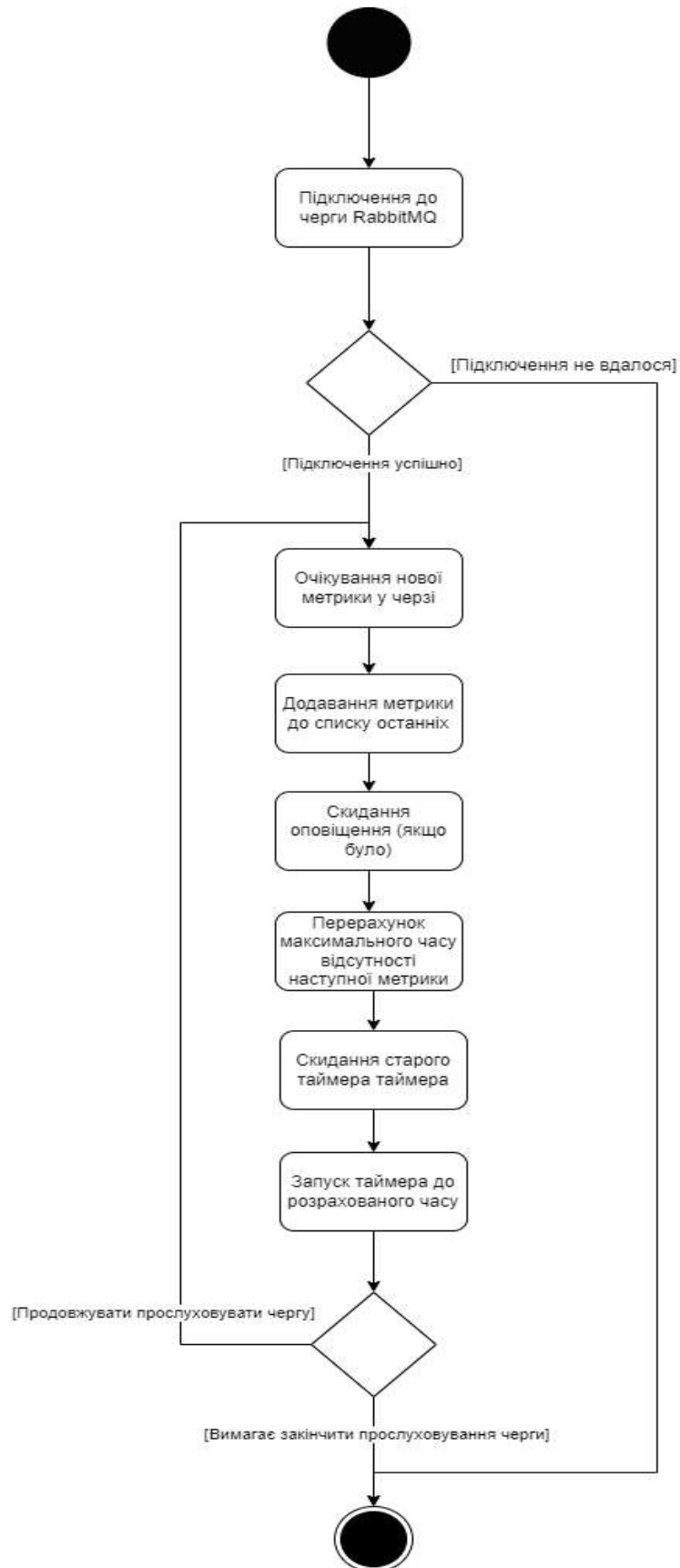


Рис 2.9 – Діаграма діяльності для процесу підрахунку максимального часу відсутності транзакції



Рис 2.10 – Діаграма діяльності процесу генерації сповіщення на основі розрахованого часу

2.3 Рішення з програмного забезпечення

Оскільки в системі потрібно не лише зберігання метрик, але і зберігання призначених для користувача налаштувань, таких як графіки, функції агрегації та ін., то звідси витікає необхідність використання відразу двох СУБД: для зберігання тимчасових рядів і для зберігання налаштувань і інших даних, необхідних для роботи додатка. На етапі аналізу вимог, були висунені

вимоги до технологій, у тому числі, і до баз даних, тому для використання були вибрані наступні рішення:

- InfluxDB для зберігання «сирих» метрик і тегів;
- Oracle Database для зберігання супутніх даних і налаштувань.

Оскільки ми використовуємо стандартні інструменти, InfluxDB, що надаються, змінювати сам спосіб зберігання даних не треба. Тому розглянемо лише способи їх збереження і читання, які будуть описані в наступному розділі, а в цьому розділі відразу перейдемо до Oracle Database.

Розглянемо, які сутності знадобляться для зберігання в реляційній БД. Оскільки там зберігатиметься уся структура графіків, то першою суттю буде сам «Графік», кожен графік може відображати відразу профілів - агрегованих метрик, означає потрібна суть «Профіль». Для опису кожного з профілів знадобиться один або декілька тегів з його ім'ям і значенням, що зберігаються в кожній з метрик, - отримуємо суть «Тег». Для підтримки сповіщень і можливості їх посилати, знадобиться суть «Шаблон сповіщення», завдяки якій можлива буде прив'язка сповіщень до профілів для їх постійного відстежування. Для розсилки сповіщень на електронну пошту знадобляться ще дві суті: «Група розсилки» і «Адресат», адресат має на увазі під собою мету розсилки з вказівкою електронної пошти, група ж включатиме декілька шаблонів сповіщень і адресатів, яким буде відправлено електронний лист у разі спрацьовування сповіщень з цієї групи. «Історія сповіщень» зберігатиме історію створених сповіщень з прив'язкою до «знімків» додатковим даним, що зберігаються по суті «Історичний профіль»

Отримані сутності можна відобразити на ER- діаграмі, приведеній на рис. 2.11.

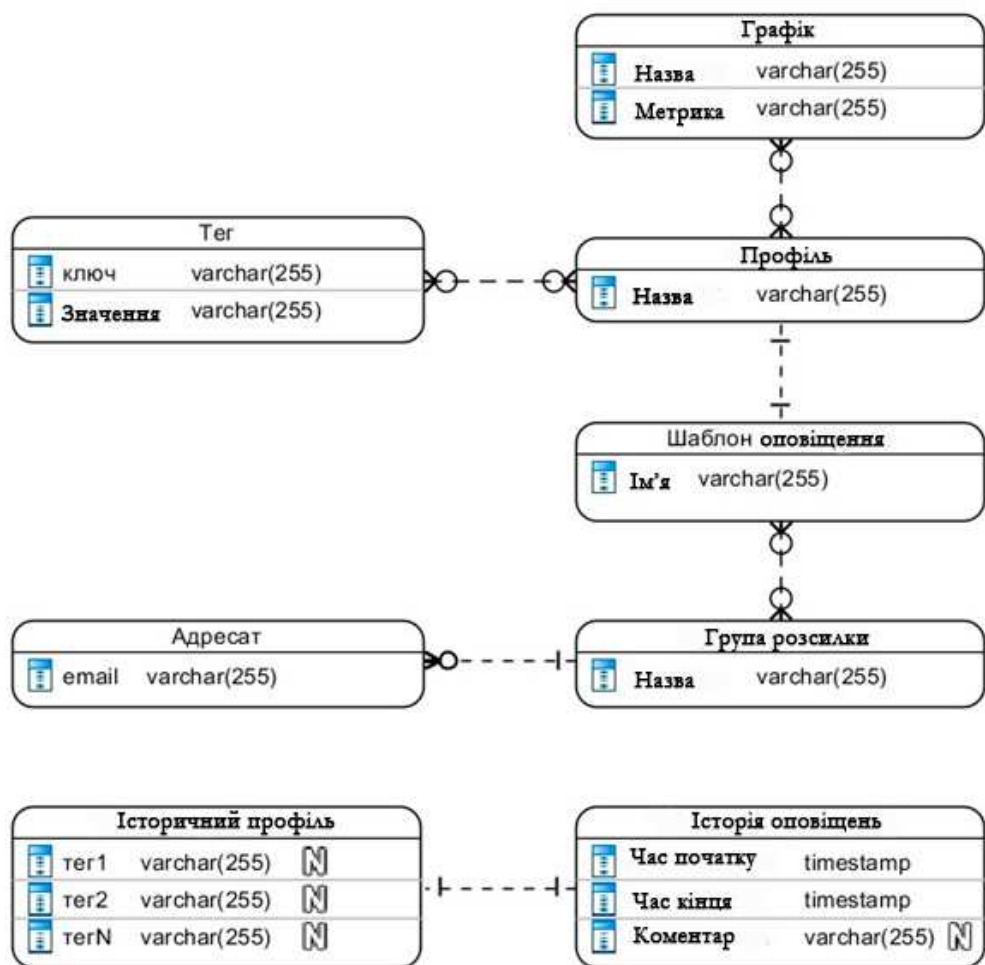


Рисунок 2.11 – Діаграма суть-зв'язок.

InfluxDB використовує закриту внутрішню структуру для зберігання метрик, тому для збереження даних треба використати Influx 'ом HTTP API або бібліотеку, що надається, для мови Java, що реалізовує цей API.

Передбачається, що зберігати треба структуру наступного виду : «<metric>, <value>, <time>, <tag _ key = tag _ value>*», де:

- <metric> - ім'я метрики;
- <value> - значення метрики;
- <time> - час, коли була зібрана метрика;
- <tag _ key = tag _ value> - ключ і значення додаткових параметрів-тегів;

- Символ зірочки «*» означає, що тегів може як бути декілька, так і не бути зовсім.

Наприклад, структура для транзакції, проведеної по банківській карті в 10 : 00 01.01.2023, з сумою платежу рівної 500, для країни Україна, валюти гривня і операції на поповнення рахунку, може виглядати таким чином:

Таблиця 2.1

Приклад метрики, що зберігається

```
«amount, 500, 10: 00 01.01.2023, type=bankcard, operation=deposit, currency=UAH, country=UA»
```

Для збереження подібної метрики, при використанні HTTP API, треба буде виконати наступний запит [14]:

Таблиця 2.2

Приклад запиту до API InfluxDB для додавання метрики

```
curl -i -XPOST 'http://localhost:8086/write?db=mydb' --data-binary 'amount, type=bankcard, operation=deposit, currency=UAH, country=UA value=500.00 1434055562000000000'
```

Читання так само можливо або через HTTP API, використовуючи SQLподібний мова запитів, опис якої є в документації на офіційному сайті InfluxDB, або, знову ж таки, використовуючи готову Javaбібліотеку.

Таблиця 2.3

Приклад запиту до InfluxDB на читання

```
curl -G 'http://localhost:8086/query?pretty=true' --data-urlencode "db=mydb" --data-urlencode "q=SELECT \"value\" FROM \"amount\" WHERE \"country\"='UA' AND \"currency\"='UAH' "
```

При запиті на читання через HTTP API відповідь повернеться у форматі JSON приблизно наступного виду :

Приклад відповіді від InfluxDB при запиті на читання

```
{ "results": [
  { "statement _id": 0,
    "series": [
      { "name": "amount"
        "columns": [
          "time"
          "value"
        ], "values": [
          [
            "2017-01-01T10: 00: 00.000000000Z",
            500
          ]
        ]
      }
    ]
  }
]
```

Виходячи з цього, при розробці буде прийнято рішення про використання або HTTP API і написанні обробника для запитів, або готового рішення.

Первинна генерація сповіщень здійснюватиметься з використанням цього методу.

Таким чином, якщо узяти певну вірогідність події, то можна розрахувати приблизний час, через який ця подія станеться з цією вірогідністю.

При постійному потоці платіжних транзакцій, відхилення поведінки системи від «нормального» можна виявити, побудувавши розподіл Пуассона на підставі розподілу транзакцій за останній відомий інтервал часу, після чого

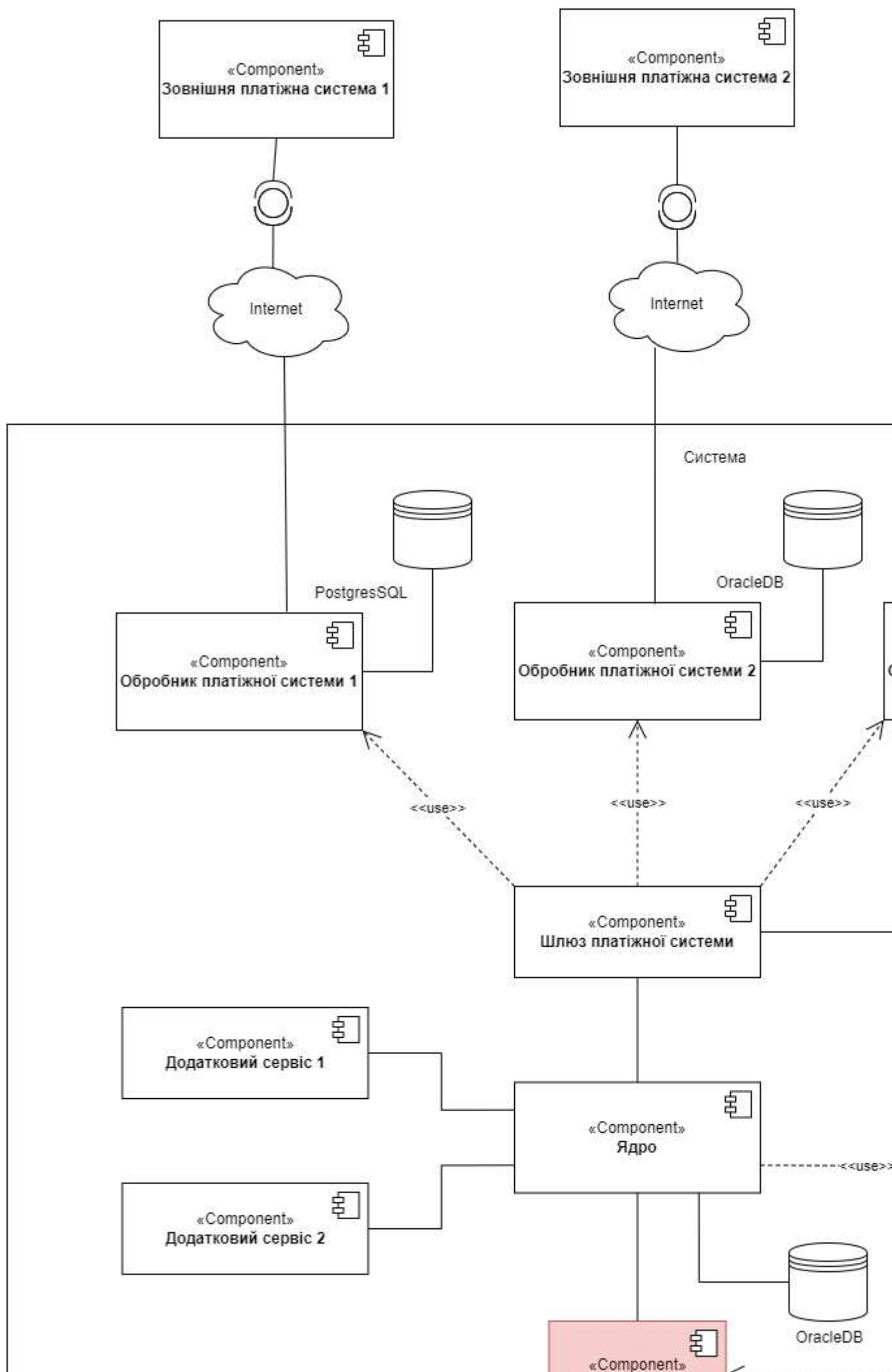
можна розрахувати максимальний інтервал, на якому транзакцій може не бути. Для цього нам знадобиться деяка вірогідність P , яка означатиме мінімальну вірогідність події досягши якої(тобто досягнення межі інтервалу, що розраховується) і вважатиметься, що поведінка системи вийшла за рамки «нормальної».

Візуалізувати загальну діаграму класів не представляється можливим в силу великої кількості розроблених класів і маленького розміру листа А4, тому будуть розглянуті більше деталізовані діаграми класів для кожного з модулів.

Для процесів, пов'язаних з побудовою графіків, були виділені особливості, ґрунтуючись на яких було створено програмне API для міжмодульної взаємодії. Діаграма класів для API представлена на рисунку 2.12.

Процес побудови графіків включає отримання метрик з БД, відібраних за певними критеріями(тегам), застосування функцій агрегації. Окрім цього, кожен графік підтримує відразу декілька ліній, що відображаються одночасно. Для підтримки описаних вимог, розроблені класи пакету graph :

- Graph - суть, що представляє сам графік;
- GraphLine - служить для підтримки декількох ліній для одного графіку;
- GraphApi забезпечує усі необхідні операції редагування для графіків;
- LineDataApi - забезпечує отримання даних за певними критеріями - профілем.



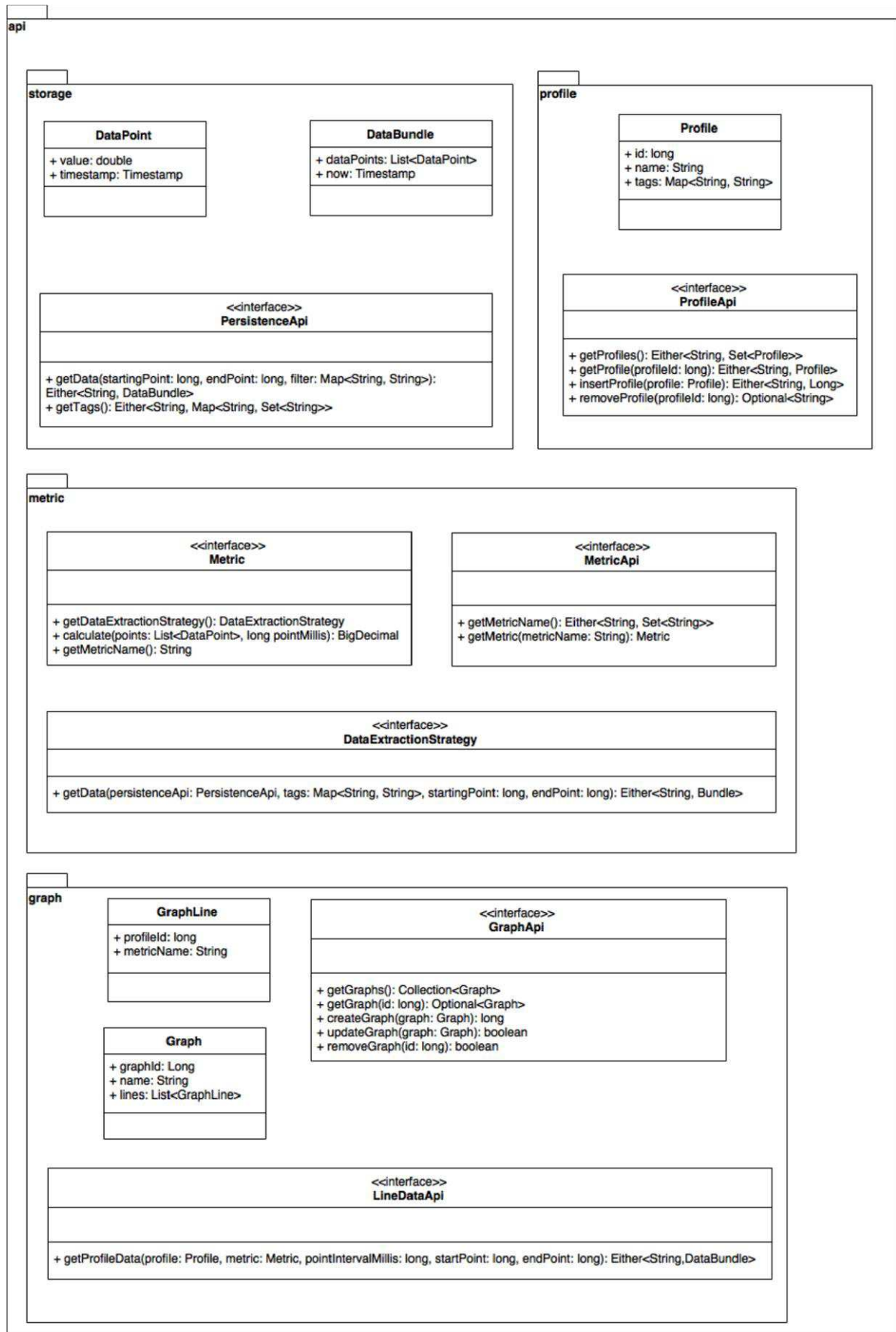


Рисунок 2.12 – Діаграма класів внутрішнього API

Профіль представлений однойменним інтерфейсом в пакеті `profile`. За профілем, якраз, і забезпечується фільтрація, - він представляє з себе набір тегів, по якому можна буде відбирати значення метрик при запитах до InfluxDB. Операції редагування профілів представлені в інтерфейсі `ProfileApi`.

У пакеті `metric` представлені інтерфейси для роботи з агрегованими метриками, а саме:

- `DataExtractionStrategy` описує спосіб отримання даних з БД для можливості вибірки даних на необхідному інтервалі і використовуватиметься у функціях агрегації, що розробляються;
- `MetricApi` дозволяє отримати доступні функції агрегації;
- `Metric` є інтерфейсом для подальшої його імплементації у функціях агрегації.

У пакеті `storage` описані інтерфейси для роботи з «сирими» метриками і можливості отримання доступних ключів і значень для тегів.

Для можливості читання/збереження даних з/в черзі повідомлень брокера RabbitMQ був реалізований наступний модуль, діаграма якого представлена на рис. 2.13

Для можливості прослуховування черг RabbitMQ, отримання даних з них у вигляді JSON- повідомлень і перетворення в об'єкти, створений пакет `storage`.

Робота безпосередньо з InfluxDB здійснюється через `InfluxDbApi`.

Для реалізації зберігання даних в InfluxDB був розроблений наступний модуль, представлений на рис. 2.14

У пакеті `influxdb` представлена реалізація обгортки над вже існуючим рішенням `org.influxdb.influxdb - java`, яка дозволить спростити наступні завдання:

- Створення шаблонних запитів;
- Перетворення JSON, що отримується від InfluxDB в об'єкти;

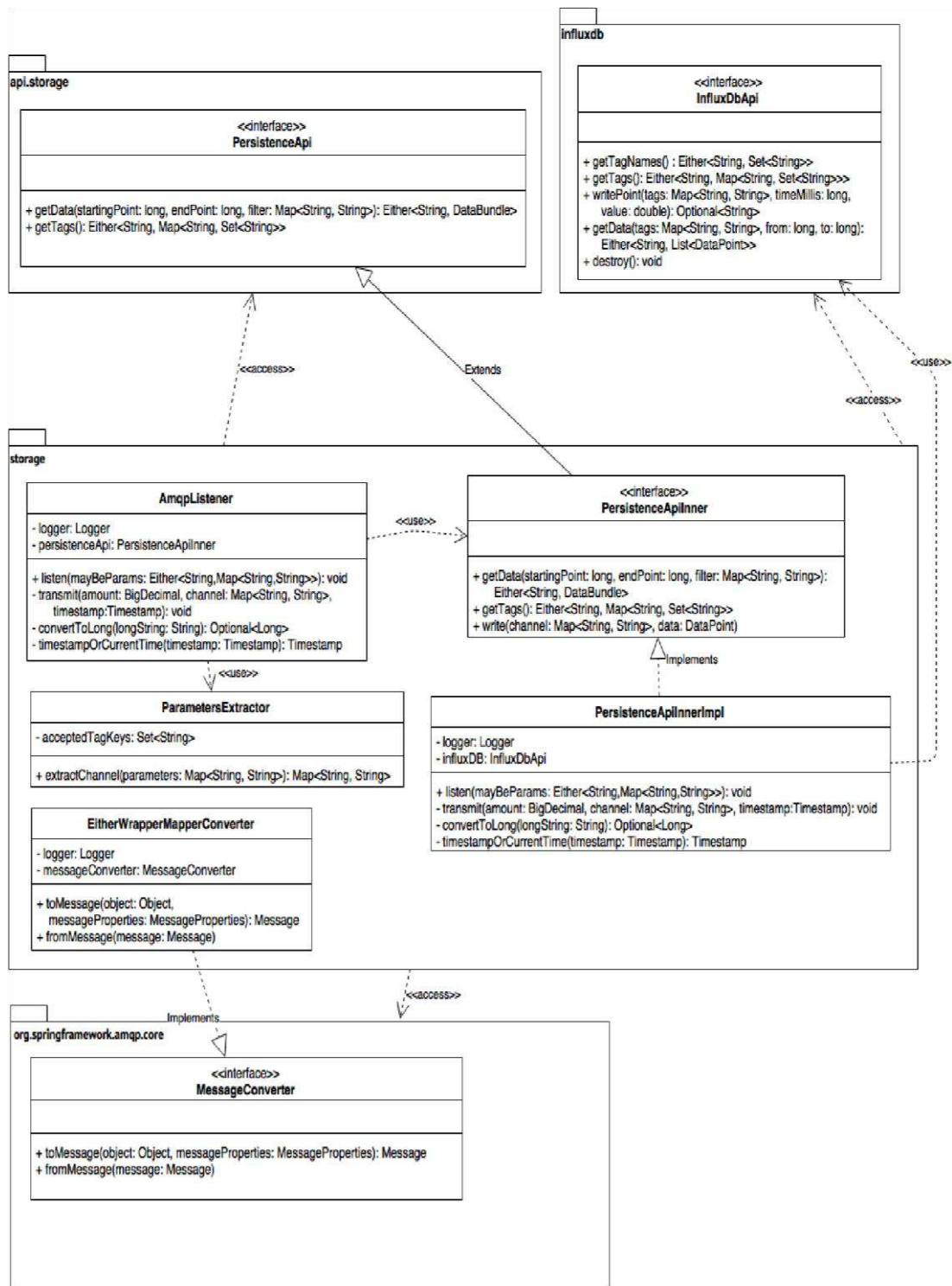


Рис. 2.13 – Діаграма класів модуля, що здійснює взаємодію з RabbitMQ

- Пакетування запитів для зниження навантаження на бд, з можливістю обробки помилок, що виникають при відправці «пакетів»;
- Логгирование інформації про запити до БД.

На рис. 2.15 представлена діаграма класів з урахуванням декількох розроблених функцій агрегації, на якій можна побачити, як вони взаємодіють з іншою системою. IntervalExtractor служить для отримання надмірних даних на інтервалі часу для можливості правильного відображення графіків, оскільки в кожній точці часу на графіці має бути визначене значення кожної з ліній, що відображаються.

Діаграма класів модуля, призначеного для зберігання і зміни графіків і налаштувань, представлена на рис. 2.16.

Для зниження навантаження на БД застосовується кешування, здійснюване класами пакетів cached і cache, що дозволяють кешувати прошені дані для одних і тих же точок на графіці і при повторному зверненні віддавати дані з пам'яті, а не просити з БД. Кеш реалізований на основі LoadingCache від Guava.

У пакеті dao представлені класи, що безпосередньо відповідають за взаємодію з Oracle Database і зберігання налаштувань графіків в ній. Модуль дозволяє розмежувати зберігання графіків, профілів і тегів.

Для взаємодії клієнтської частини і серверною, був розроблений модуль, ґрунтований на фреймворке «Spring MVC», що дозволяє організувати логіку обробки запитів до додатка з використанням HTTP запитів. Діаграма класів цього модуля представлена на рис. 2.17.

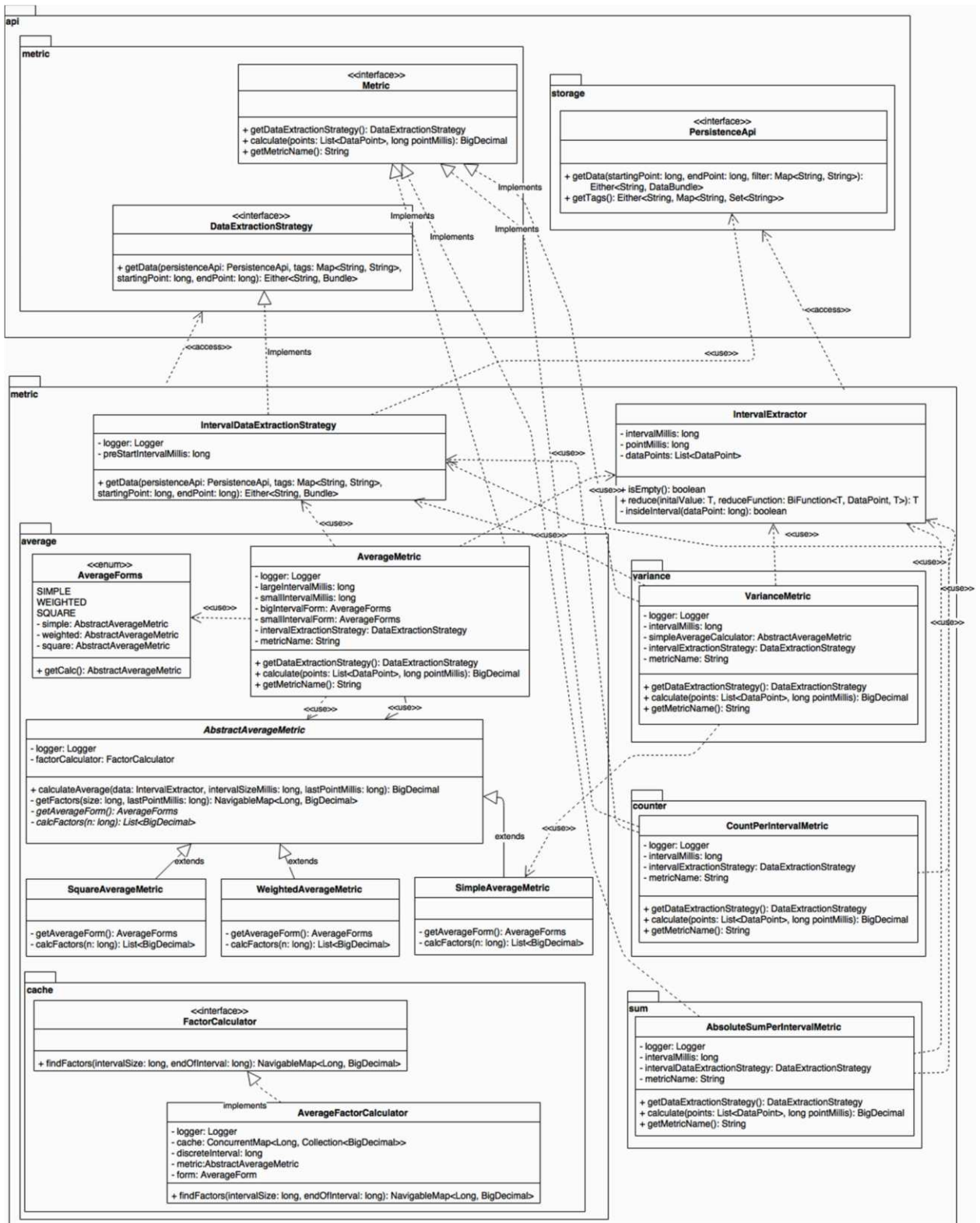


Рис 2.15 – Діаграма класів модуля, призначеного для роботи з функціями агрегації

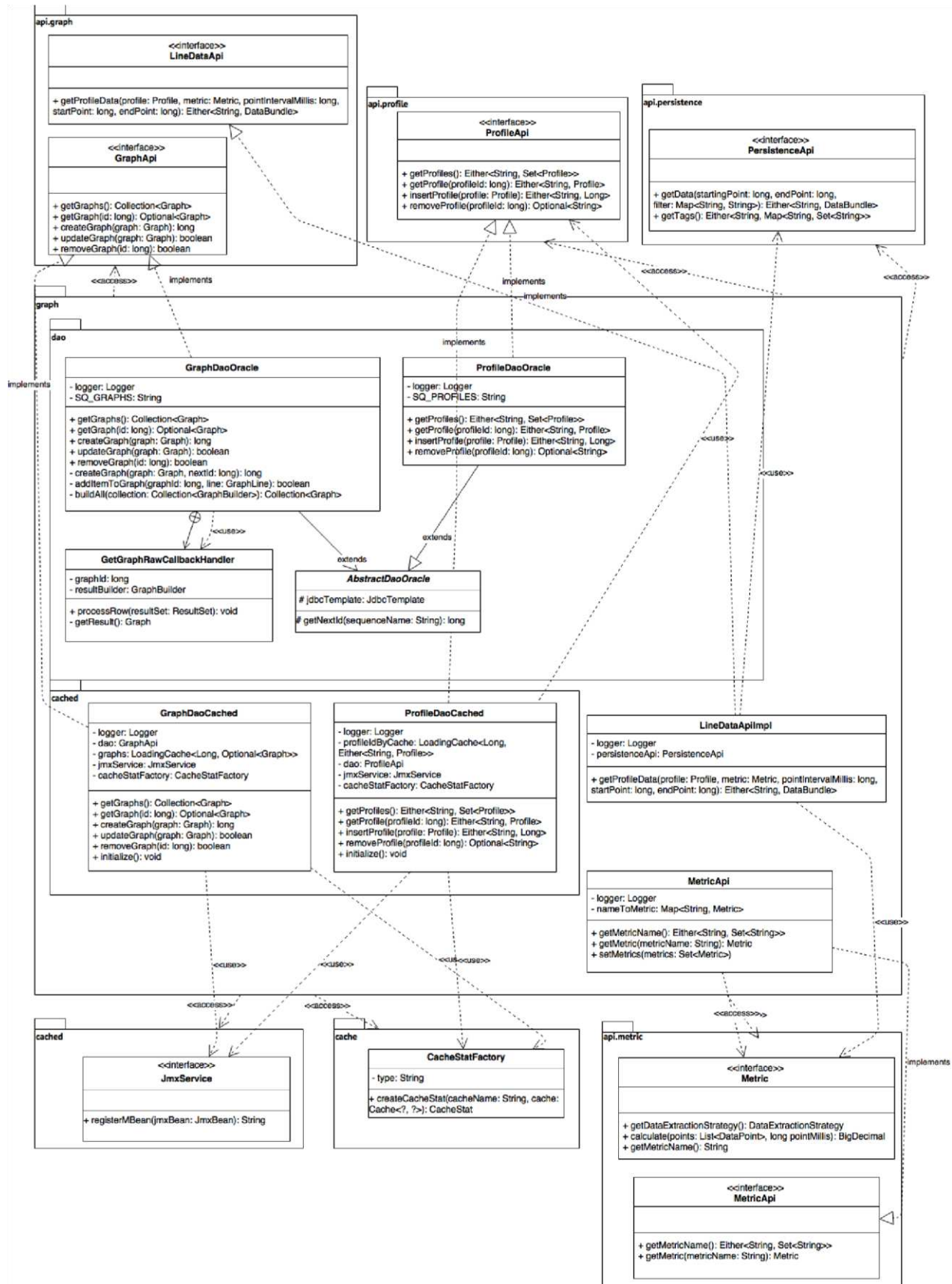


Рис. 2.16 – Діаграма класів модуля, що відповідає за редагування графіків

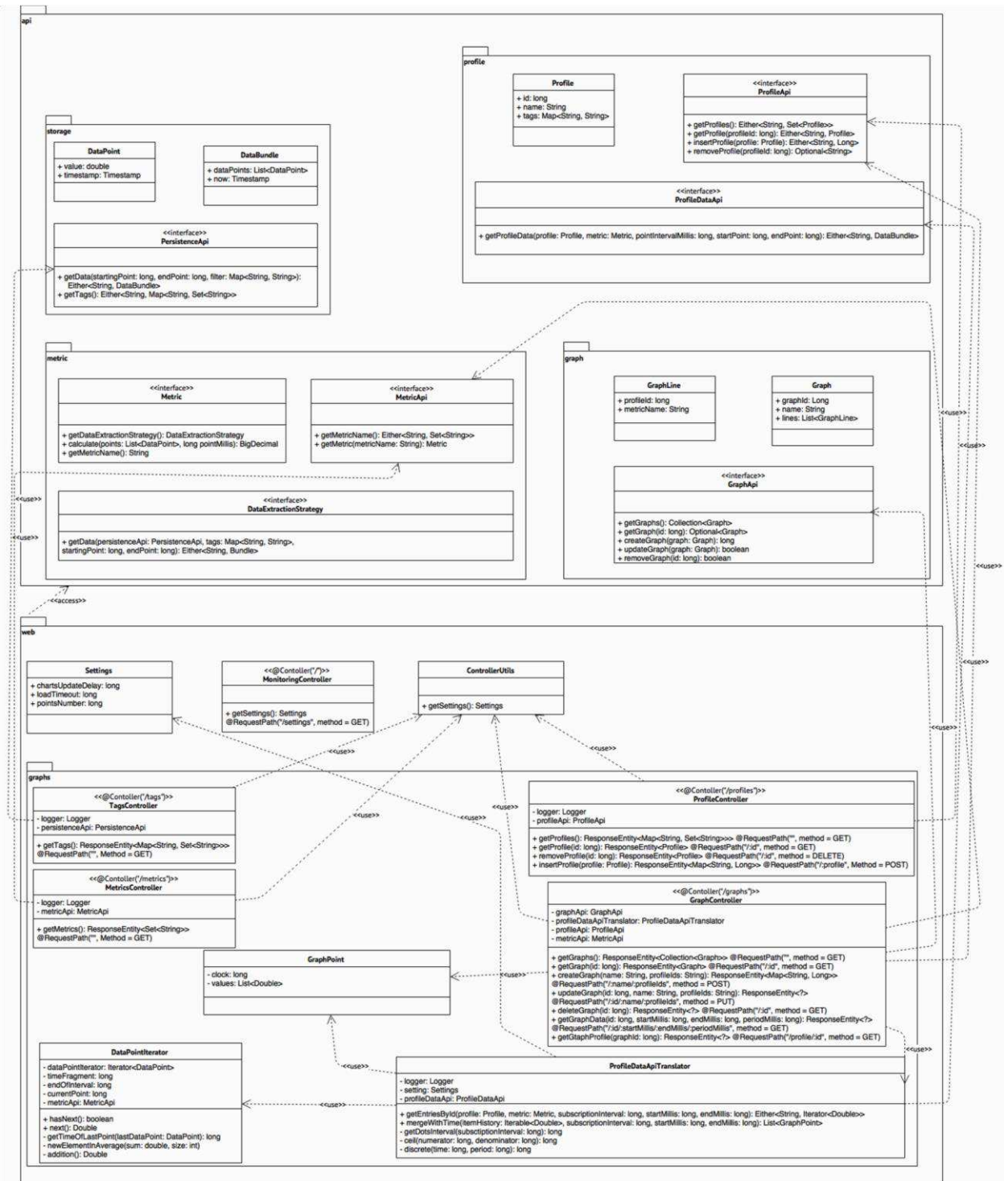


Рис. 2.17 – Діаграма класів модуля, що відповідає за взаємодію клієнтської і серверної частин додатка

Зберігання метрик

Як вже говорилося при описі технологій, уся взаємодія з InfluxDB відбувається через бібліотеку «org.influxdb.influxdb - java», що надає доступ до InfluxDB через HTTP/UDP API. на момент розробки проекту використовувана версія цієї бібліотеки була 2.1.

Для збереження лічених з RabbitMQ метрик в InfluxDB, дані спочатку валидируються, а потім перетворюються в необхідний формат для використовуваного API. Валидація і перетворення відбуваються в класі com.company.monitoring.input.amqp.AmqpListener, код якого представлений в додатку А. Читання даних також відбувається через API, для отримання даних використовуються наступні запити, параметри яких підставляються безпосередньо під час виконання :

Таблиця 2.4

Запити до InfluxDB

SHOW TAG KEYS FROM %s
SHOW TAG VALUES FROM %s WITH KEY = %s
SHOW TAG VALUES FROM %s WITH KEY IN(%s)
SELECT %s FROM %s WHERE time>=%sms AND time<%sms %s ORDER BY time ASC
SELECT %s FROM %s WHERE time < %sms %s ORDER BY time DESC LIMIT %s

Перші три запити використовуються для отримання тегів і їх значень для певної метрики. Останні два - для отримання списку самих метрик з обмеженням по інтервалу часу або кількості записів і тегам. Теги заздалегідь підставляються перед конструкцією «ORDER BY» і перетворюються на конструкцію виду «AND <tag1 _ name> = <tag1 _ value> AND <tag2 _ name> = <tag2 _ value> AND », де tagN _ name - ім'я тега, а tagN _ value - його значення.

У поточній реалізації, дані, що зберігаються в InfluxDB, не піддаються видаленню, оскільки це не потрібно. За бажання, це легко налаштується в планувальнику.

Зберігання супутніх даних

На етапі проектування була створена ER- діаграма. Розглянемо структуру таблиць, що вийшла на її основі, для Oracle Database.

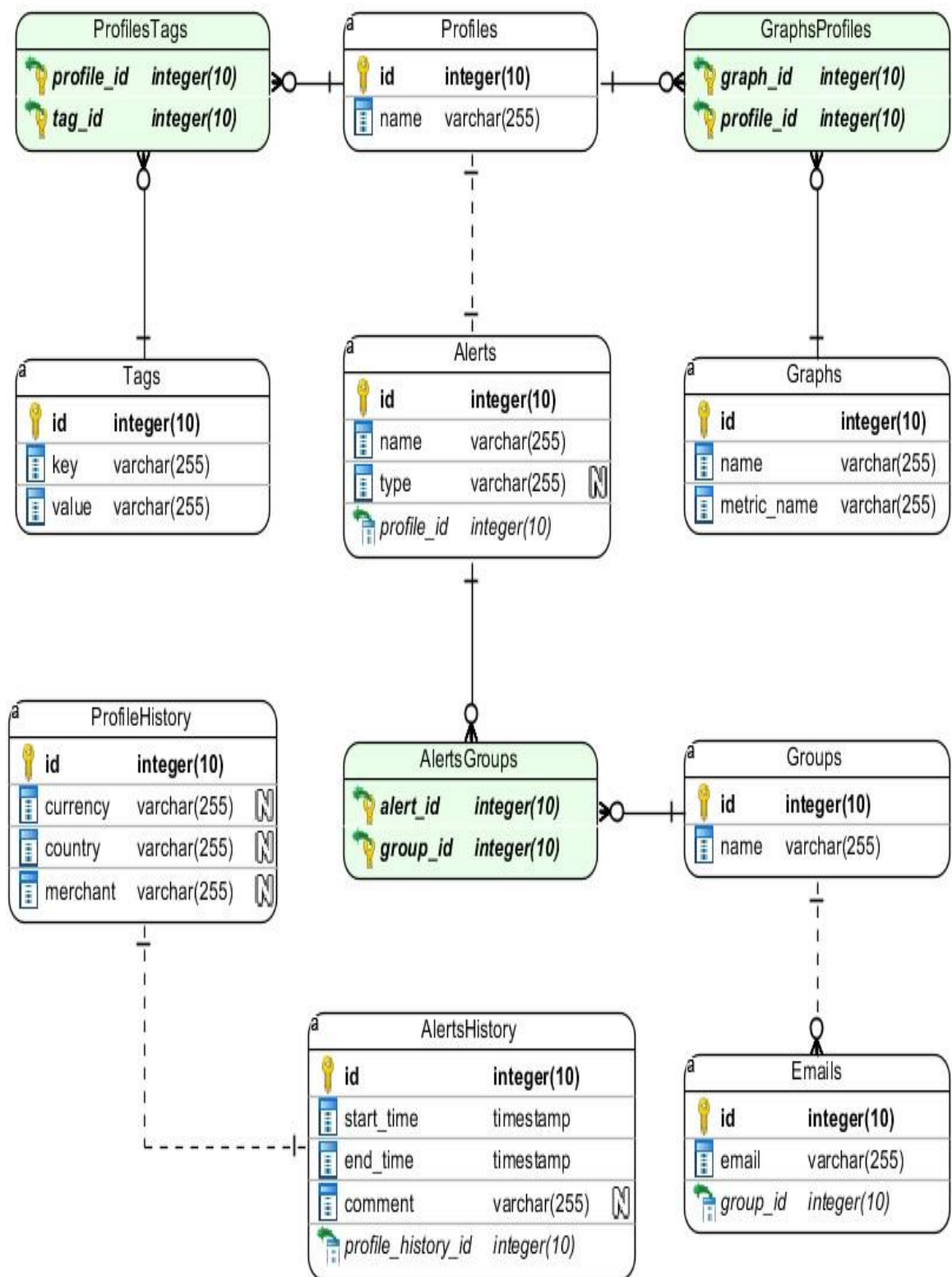


Рис 2.18 – Структура таблиц для Oracle Database

Приведена схему описана в таблиці 2.2:

Опис схеми бази даних Oracle

Ім'я таблиці	Опис
Tags	Зберігає теги для метрик(Успішна/транзакція неуспіху, країна, валюта, юрисдикція, тип платіжної системи і так далі) у вигляді пар ключ-значення
Profiles	Зберігає профілі, які складаються з декількох тегів
ProfilesTags	Забезпечує зв'язок багато-багатьом між таблицями Profiles і Tags
Graphs	Зберігає графіки
GraphsProfiles	Забезпечує зв'язок між графіками і профілями, як на одному графіку може відображатися відразу декілька профілів по одній і тій же метриці, то GraphsProfiles забезпечує зв'язок багато-багатьом між Graphs і Profiles
Alert	Зберігає шаблони сповіщень, профілі яких треба відстежувати
Groups	Зберігає групи шаблонів сповіщень, які використовуватимуться при розсилці
AlertsGroups	Забезпечує зв'язок багато-багатьом між таблицями Alert і Groups, забезпечуючи можливість зберігати в одній групі декілька шаблонів сповіщень
Emails	Зберігає підписки адрес електронної пошти на групи сповіщень
AlertsHistory	Зберігає історію створених сповіщень
ProfileHistory	Зберігає копію профілю, по якому було створено сповіщення, і додаткові відомості

DDL- схема для створення бази даних приведена в додатку.

Використання розподілу Пуассона для генерації сповіщень

З використанням розподілу Пуассона можна вичислити максимальний час до наступної події(тобто транзакції) з відомою вірогідністю P .

Розроблений сервіс дозволяє розрахувати цей час, а також дозволяє використати таймери, за допомогою яких можна відстежити настання розрахованого часу, після чого згенерувати сповіщення за відсутності транзакцій.

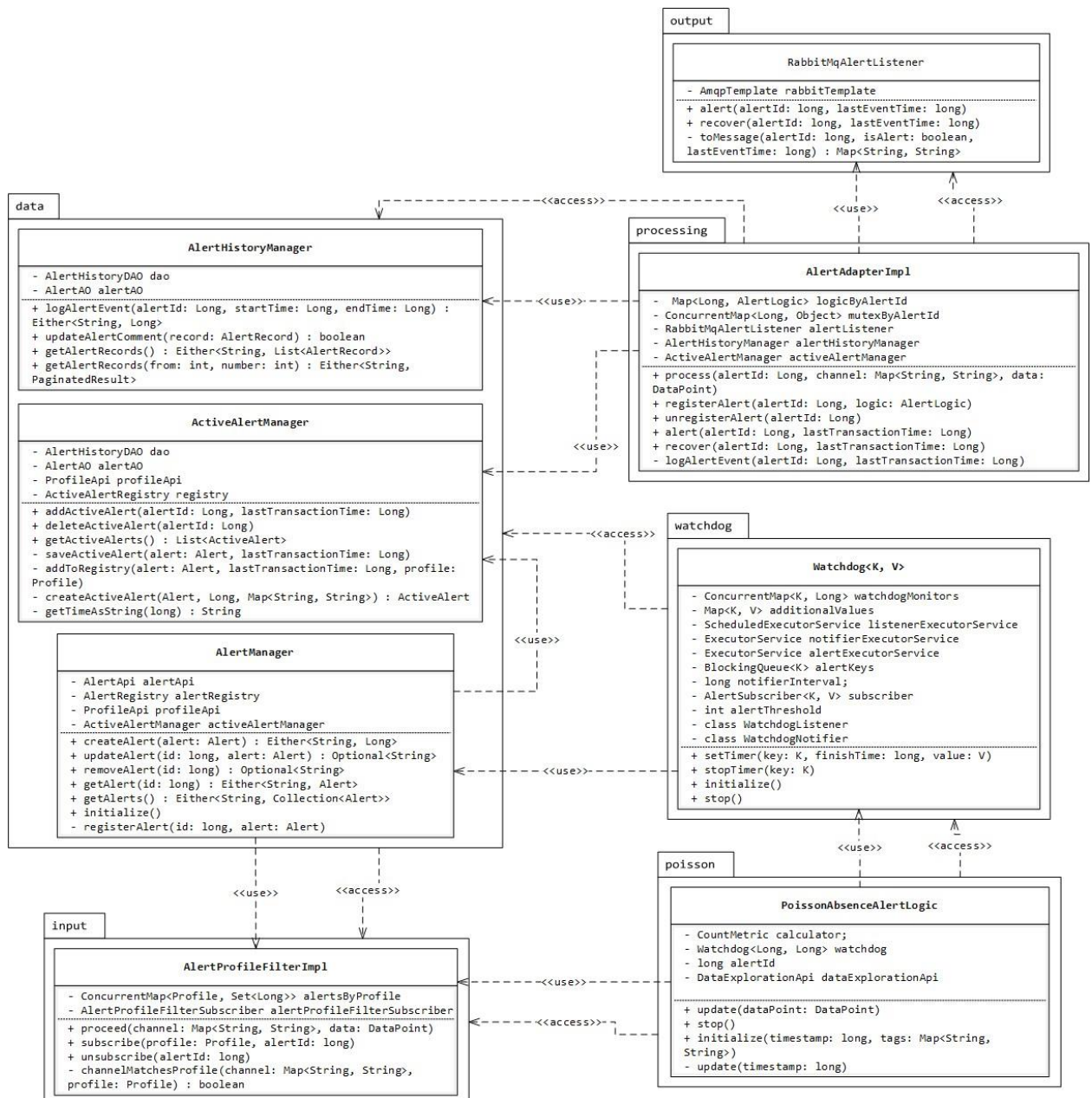


Рис 2.19 – Діаграма основних класів модуля сповіщень.

Розрахунок максимального часу відсутності транзакції робиться в класі в класі `PoissonAbsenceLogic`.

WatchDog - таймер, що відстежує настання розрахованого часу.

Взаємодія з чергою повідомлень і вивантаження сповіщень відбуваються в класі RabbitMqAlertListener.

Клас AlertHistoryManager відповідає за історію сповіщень. Він використовує AlertHistoryDAO використовується для взаємодії з Oracle Database.

Початковий код сервісу сповіщень представлений в додатку.

Відправка e - mail сповіщень

Для відправки повідомлень по електронній пошті був створений сервіс, що дозволяє робити розсилку по створених групах, які зберігаються в Oracle Database в таблицях Groups і EmailGroups.

Діаграма класів розробленого сервісу приведена на рис 2.20.

Клас AlertMailListener призначений для читання збережених сповіщень з черги RabbitMQ, після чого вони конвертуються за допомогою MailMessageConverter 'а і вирушають за допомогою MailSender 'а на адреси електронної пошти.

Підписка на групи сповіщень здійснюється через Mail Subscription Managere.

Для тестування відправки повідомлень розроблена заглушка Mail Service Mock, реалізуюча API існуючого віддаленого сервісу для відправки повідомлень на електронну пошту.

Код розробленого сервісу представлений в додатку А.

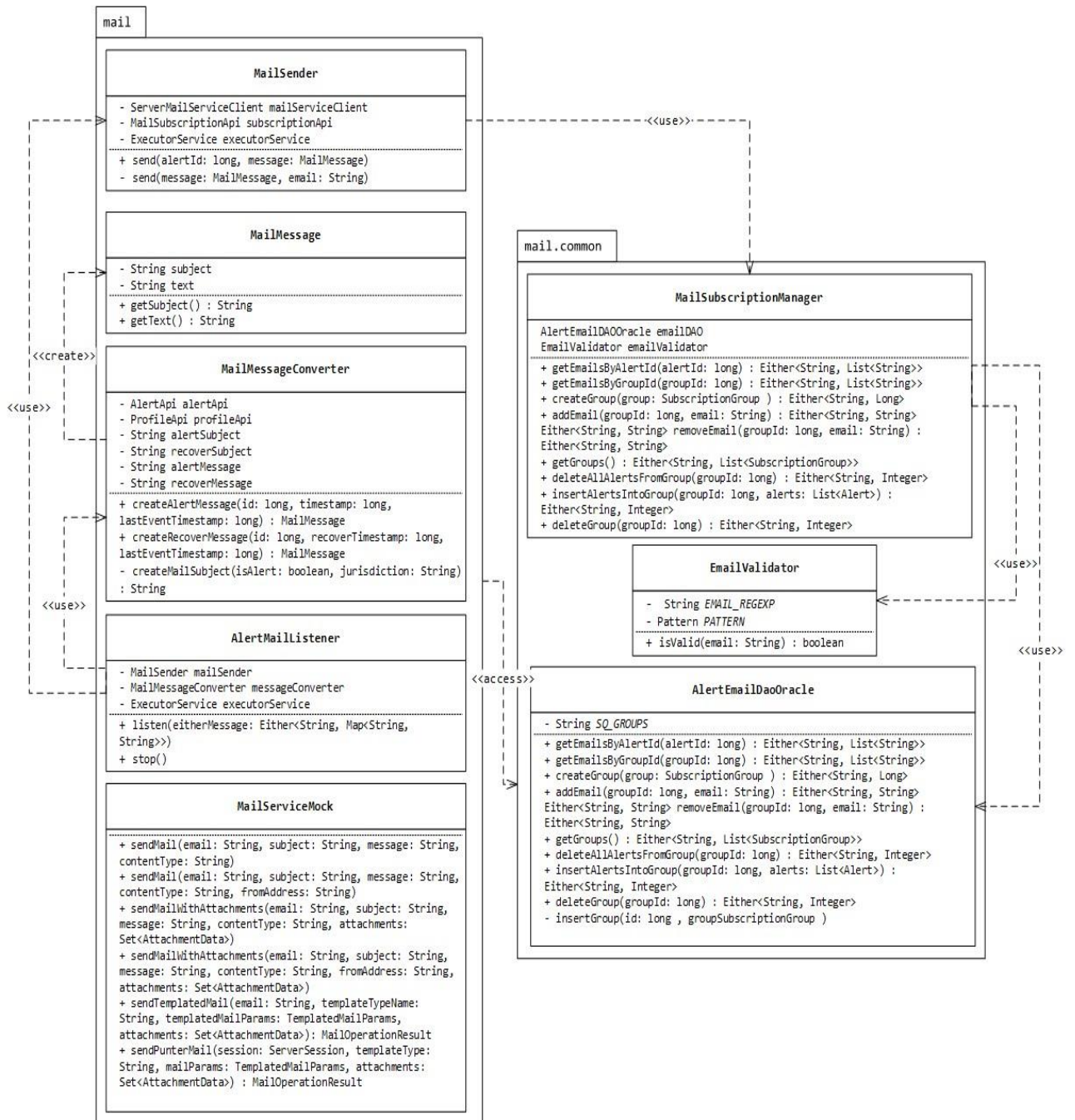


Рис 2.20 – Діаграма класів сервісу відправки email- повідомлень

Відправка сповіщень в Telegram- бот

Для відправки сповіщень в Telegram був створений бот. Код серверної частини бота знаходиться в класі TelegramSender і приведений в таблиці

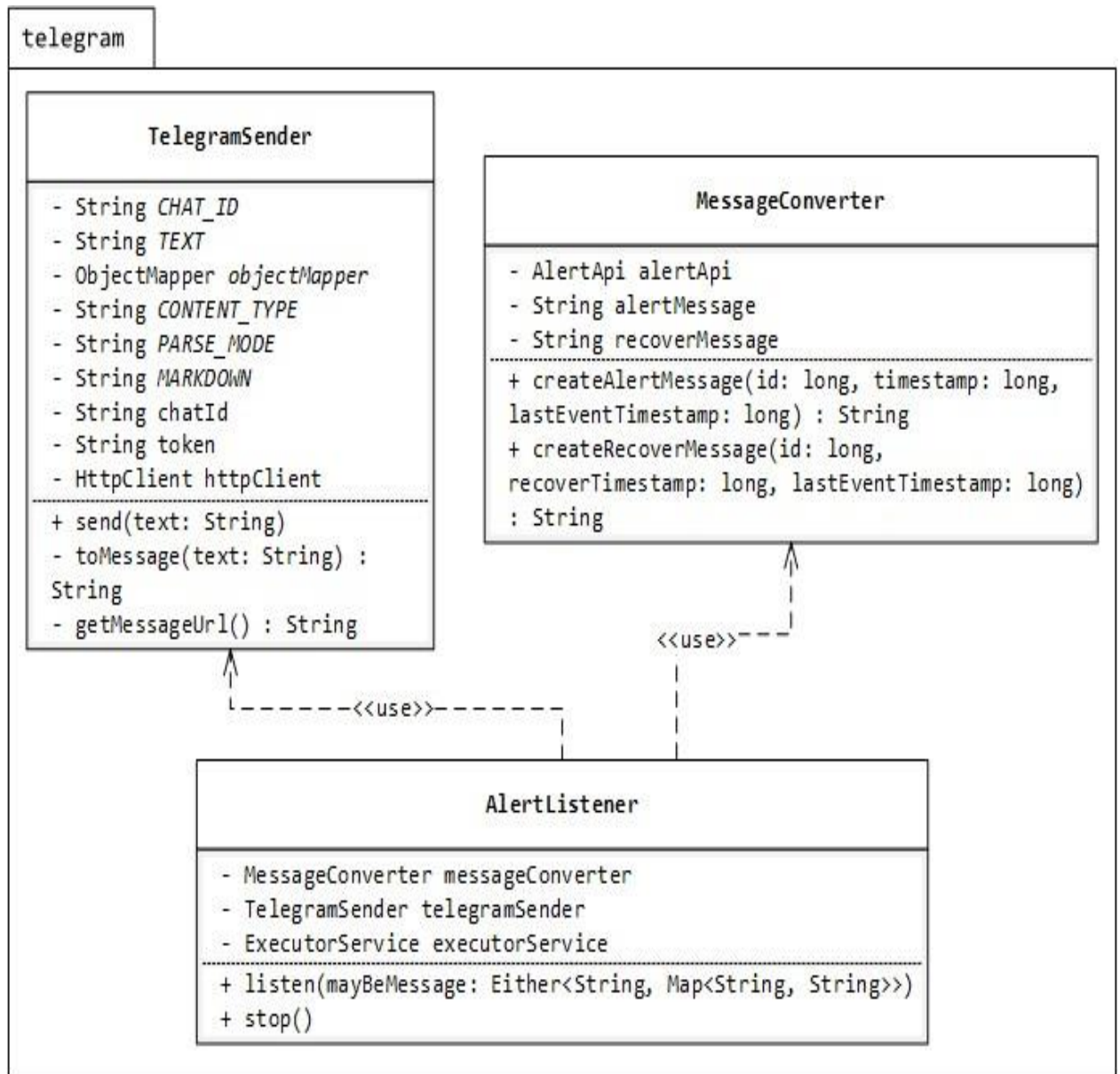


Рис 2.21 - Діаграма класів сервісу відправки повідомлень в Telegram

Клас `AlertMailListener`, по аналогії з класом `AlertMailListener`, призначений для читання збережених сповіщень з черги RabbitMQ. `MessageConverter` перетворить повідомлення у вид, потрібний для подальшої відправки, а клас `TelegramSender` відправляє повідомлення в API Telegram, для подальшої відправки їх від імені бота в потрібний груповий чат.

Для відправки повідомлення через HTTP протокол використовується об'єкт класу `HttpClient`, що проініціалізував в контексті додатка з необхідними параметрами для доступу до видаленому Telegram API.

Метод `getMessageUrl(.)` повертає HTTP URL для Telegram API

Проектування інтерфейсу

Проектування інтерфейсу полягає в створенні прототипу інтерфейсу. Оскільки проект призначений для внутрішнього використання в компанії, дизайн інтерфейсу не так важливий, як його функціональність.

Як було сказано на етапі аналізу вимог, у веб-інтерфейсі потрібна можливість вибору графіків і їх перегляду в реальному часі. Окрім цього, потрібна можливість конфігурувати саму систему моніторингу - додавати графіки, профілі, теги і так далі. Окрім цього, потрібний інтерфейс для конфігурації сповіщень, перегляду створених сповіщень і підписки на них.

Для візуалізації вимог до інтерфейсу буде досить розглянути прототипи трьох основних сторінок, які приведені на Рис 2.22-2.24.

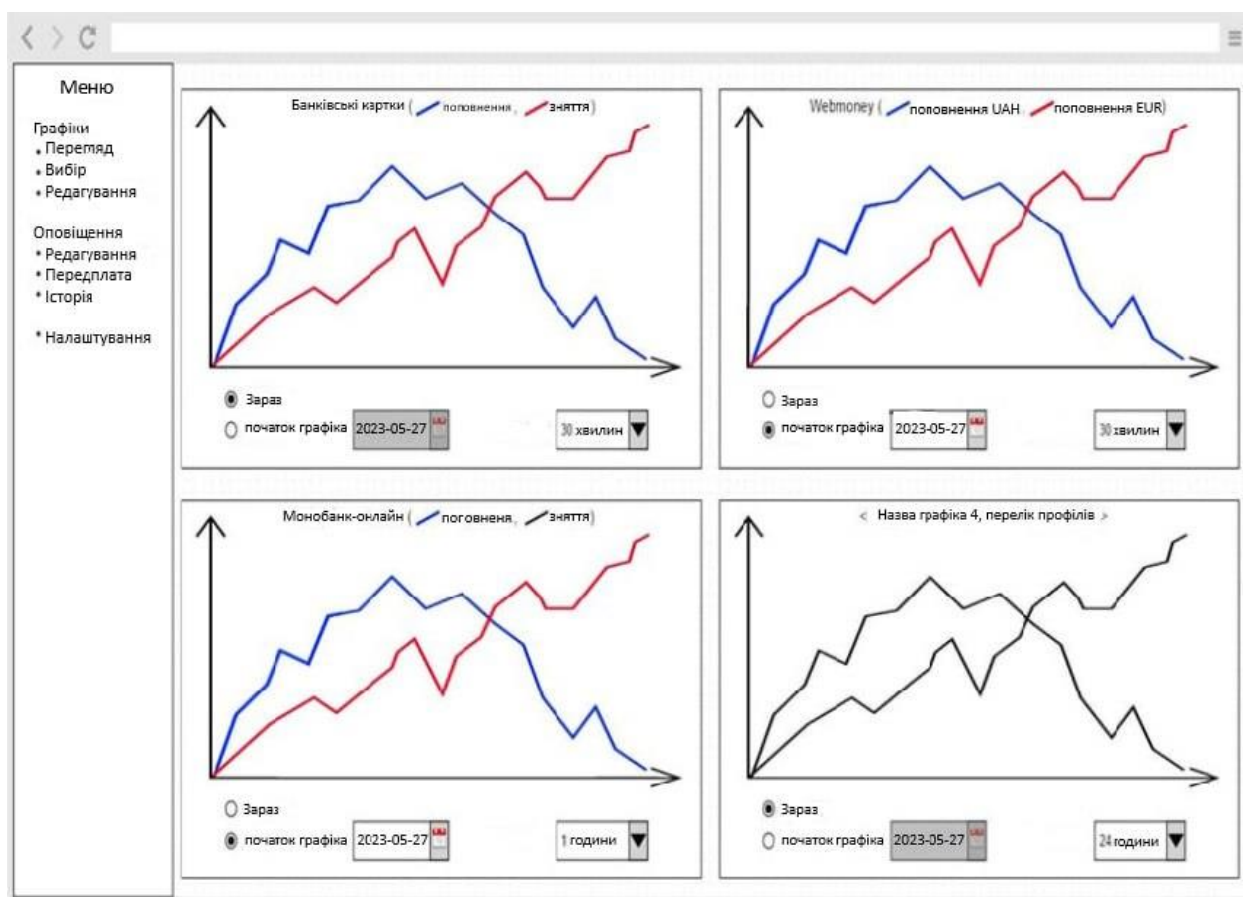


Рис. 2.22 - Прототип сторінки інтерфейсу відображення графіків

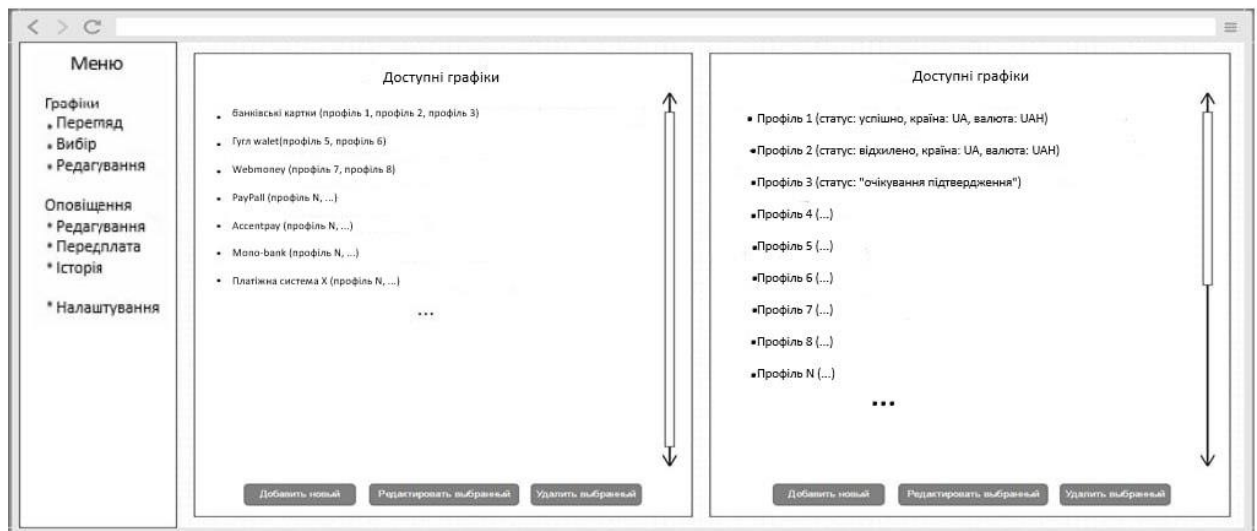


Рис .2.23 - Прототип сторінки інтерфейсу редагування графіків і профілів

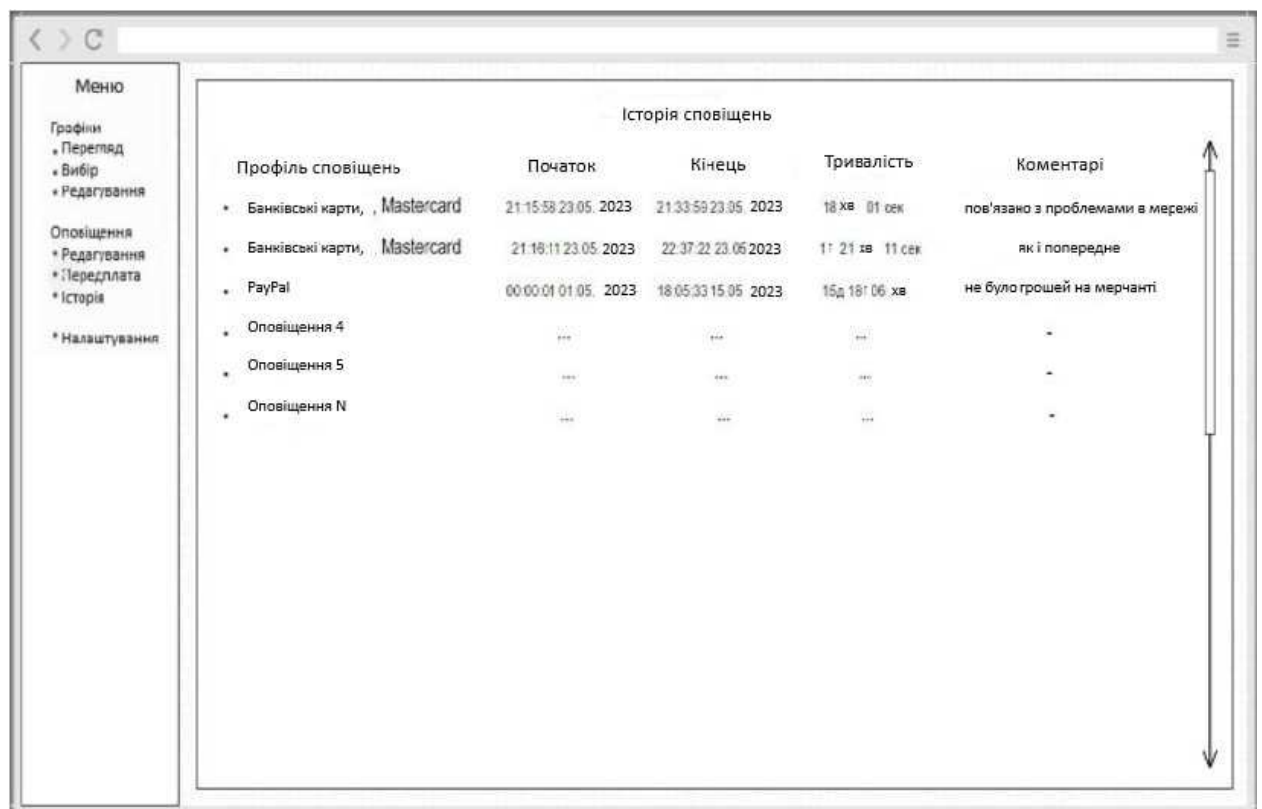


Рис 2.24 - Прототип сторінки інтерфейсу історії сповіщень

РОЗДІЛ 3 РЕАЛІЗАЦІЯ ПРОЕКТУ СИСТЕМИ МОНІТОРИНГУ

3.1. Вибір мови розробки проекту та технологій

Розглянути інструменти та технології які будуть використані при розробці.

В якості середовища розробки на мові Java була вибрана комерційна IDE «IntelliJ IDEA» від компанії «JetBrains», що дозволяє автоматизувати рутинні операції і що має можливості розширеного рефакторингу.

Була використана версія 2017.1.3.

Сервером додатків був вибраний вже розгорнутий і налагоджений «Apache Tomcat» від компанії «Apache Software Foundation», розроблений спеціально для розгортки Java веб-застосунків. Використовувана версія - 8.0.27.

Як сервер черг повідомлень використовується «RabbitMQ» компанії «Pivotal Software» і плагін «RabbitMQ-Management» для можливості переглядати стан сервера і черг через вебінтерфейс. Версія використовуваного сервера - 3.6.4.

В якості СУБД вибрано, так само вже використовуване, рішення «Oracle Database(RDBMS) », а в якості IDE для управління БД був вибраний «Oracle SQL Developer». Використовувана версія СУБД - 12c Release 1(12.1.0.1), версія IDE - 4.2.0.17.089.1709.

В якості бази даних для зберігання метрик вибраний «InfluxDB». Остання доступна стабільна версія InfluxDB на даний момент - 1.2.2(випущена 14 березня 2017 року [15]) і вже використовується усередині компанії, що і обґрунтовує її використання в цій роботі.

Як система контролю версій використовувався «Git», розробка велася в локальному репозиторії з подальшим вивантаженням проекту у видалений(репозиторій).

Збирачем проекту був «Maven» - фреймворк для автоматизації складання проектів на основі опису їх структури у файлах на мові POM(англ.

Project Object Model). Використовувана версія(вбудована в IDE «IntelliJ IDEA») - 3.3.9.

При розробці програмної системи стояла мета знайти такі технології, які б дозволили спростити реалізацію системи і мінімізувати повторення вже існуючих рішень при розробці.

В результаті проведеного аналізу існуючих рішень і технологій для Java, вибір був зроблений на наступних:

Мова програмування - Java 1.8 з використанням можливостей SE(Standard Edition) версії.

Для архітектури усього застосування була вибрана програмна платформа «Spring MVC Framework», що реалізовує патерн модель-видконтроллер і дозволяє відділяти шари додатка за допомогою Javaаннотаций.

Для взаємодії з API InfluxDB вибраний клієнт для мови Java «org.influxdb.influxdb - java», реалізуючий HTTP/UDP API і що дозволяє використати CRUD- операції. Цей клієнт доступний для підключення, як залежність Maven проекту.

Для реалізації взаємодії між модулями додатка вибрано рішення «Spring Remote», що реалізовує підхід RMI з використанням Hessian протоколу для передачі даних.

Для оформлення статичних частин веб-інтерфейсу використовується фреймворк «Twitter Bootstrap», зокрема, використовуються його CSS- стилі і JavaScript рішення.

Для отрисовки графіків у веб-інтерфейсі використовується бібліотека «Google charts», що дозволяє відображати відразу декілька ліній на кожному з графіків.

Для побудова іншої динамічної логіки на клієнтові використовується AngularJS в сукупності з JavaScript. Це дозволяє зменшити зв'язність коду, розбити логіку на частини відповідно до моделі MVC і використати підхід з так званою інверсією управління - DI.

3.2 Результати розробки

Для первинної версії, щоб уникнути використання бота сторонніми користувачами, можливість відправляти повідомлення була обмежена тільки одним груповим чатом, ідентифікатор якого передається в конструктор класу TelegramSender у вигляді строкового параметра chatId. Треба додати створеного бота в груповий чат, після чого бот зможе отримати цей ідентифікатор і відправляти в цей чат повідомлення. Початковий код розроблених класів приведений в додатку .

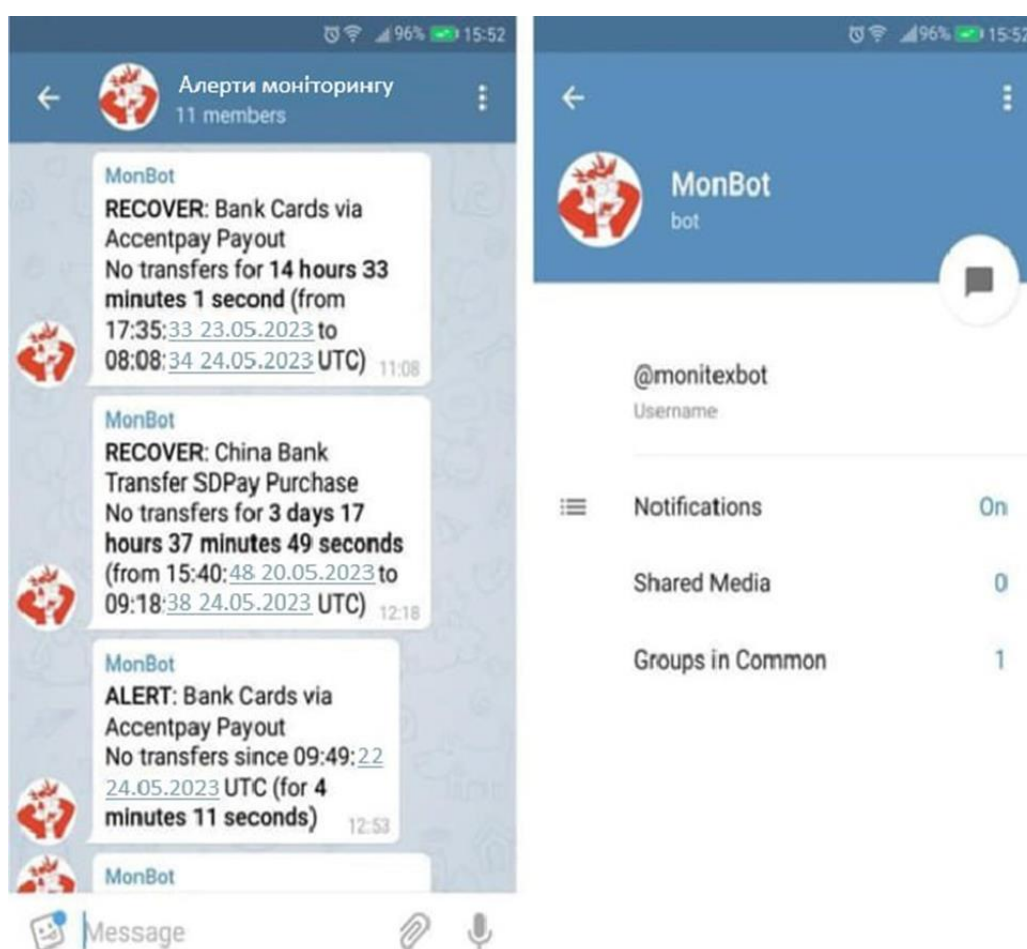


Рис – 3.1 Інтерфейс розробленого бота з прикладом оповіщення

Інтерфейс веб-клієнта

При розробці графічного інтерфейсу використаний набір інструментів «Twitter Bootstrap», що включає в різні рішення з використанням CSS, JavaScript і HTML для оформлення веб-сторінок.

Окрім цього, оскільки статичних сторінок недостатньо для відображення графіків, для розробки динамічної частини інтерфейсу застосовані наступні бібліотеки і технології, описані на початку цієї глави :

- AngularJS;
- JavaScript;
- Ajax;
- Google - Charts.

При розробці з'ясувалося, що Google - Charts зобов'язує передавати йому дані по усіх точках на графіці для кожного з профілів. Це означає, що в кожній точці графіку мають бути визначені значення кожної з ліній, що відображаються.

Приклад розробленого графічного інтерфейсу наведений на рис.3.2-3.4. Основні частини коду і розмітки сторінок приведені в додатку.

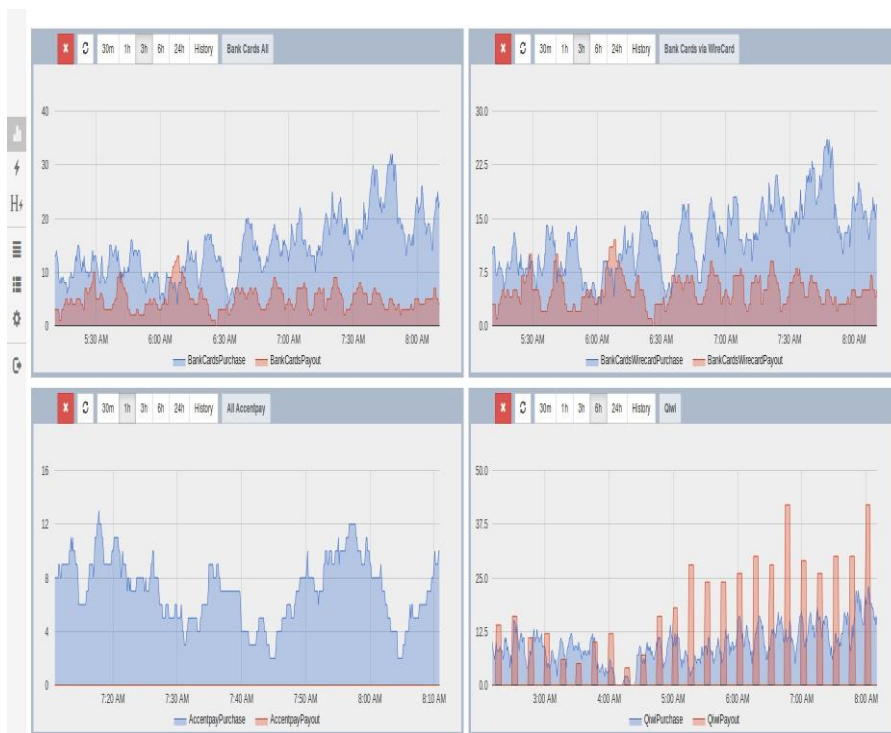


Рис 3.2 – Знімок екрану з відображенням графіків

Alert history								
ID	Jurisdiction	PS	Merchant ID	Operation	Start Time	End Time	Duration	Comment
1110		ecopayz		purchase	29-01-49 14:15:2023	01-29-23 15:05:2023	2 hours 27 minutes 34 seconds	
1109		sportsbetting		cost	01-17:55 25.05.2023	01-19:21 25.05.2023	1 minute 21 seconds	empty
1108		bankcard	59	purchase	01-00:03 25.05.2023	01-13:23 25.05.2023	13 minutes 20 seconds	empty
1107		sportsbetting		cost	01-01:42 25.05.2023	01-11:57 25.05.2023	10 minutes 14 seconds	empty
1106		bankcard	84	payout	15-17:54 24:05:2023	15-21:45 24:05:2023	3 minutes 51 seconds	empty
1105		sportsbetting		cost	14-11:42 24.05.2023	14-11:44 24:05:2023	1 second	empty
1104		sportsbetting		cost	12-11:23 24:05:2023	12-11:24 24:05:2023	1 second	empty
1103		bankcard	84	payout	08-49:22 24:05:2023	10-00:17 21.05.2023	11 minutes 24 seconds	empty
1102			60	purchase	15-40:48 20.05.2023	08-13:38 24.05.2023	3 days 17 hours 37 minutes 49 seconds	empty
1101		bankcard	84	payout	17-35:33 23.05.2023	08-08:31 24.05.2023	14 hours 33 minutes 1 second	empty
1100		chinaonlinepay	148	purchase	19-15:34 23:05:2023	23-23:07 23.05.2023	3 hours 12 minutes 32 seconds	empty
1099		qiwi		purchase	20-22:06 23.05.2023	20-26:40 24.05.2023	4 minutes 43 seconds	empty
1098		sportsbetting		cost	17-48:44 23:05:2023	17-38:56 23.05.2023	1 second	empty
1097		bankcard	84	payout	16-47:13 23.05.2023	17-35:33 23.05.2023	48 minutes 20 seconds	empty
1096		sportsbetting		cost	11-30:19 23.05.2023	11-30:21 23.05.2023	1 second	empty

Рис 3.3 – Приклад розробленого інтерфейсу, сторінка історії сповіщень

Graphs		Profiles	
Add new graph or select existing to change his profiles		Click on profiles to add or remove them from graph. Order matters!	
Graph name Add		New profile name Add	
Bank Cards All	Edit metric Remove	AccentpayPayout filtered by [{"result":"success","merchantId":"84","operation":"payout"}]	Remove
Bank Card via Credorax	Edit metric Remove	AccentpayPurchase filtered by [{"result":"success","merchantId":"84","operation":"purchase"}]	Remove
Bank Cards via WireCard	Edit metric Remove	All filtered by []	Remove
Bank Cards via WinPay	Edit metric Remove	BankCardAlderneyPayout filtered by [{"result":"success","jurisdiction":"alderney","type":"bankcard","operation":"payout"}]	Remove
Bank Card via Accentpay	Edit metric Remove	BankCardAlderneyPurchase filtered by [{"result":"success","jurisdiction":"alderney","type":"bankcard","operation":"purchase"}]	Remove
All Visa/Visa Electron	Edit metric Remove	BankCardEntropayPayout filtered by [{"result":"success","cardType":"entropay visa","type":"bankcard","operation":"payout"}]	Remove
All MasterCard/Maestro	Edit metric Remove	BankCardEntropayPurchase filtered by [{"result":"success","cardType":"entropay visa","type":"bankcard","operation":"purchase"}]	Remove
Entropay	Edit metric Remove	BankCardEurowebPurchase filtered by [{"result":"success","sessionSource":"euroweb","type":"bankcard","operation":"purchase"}]	Remove
All Accentpay	Edit metric Remove	BankCardMaestroPayout filtered by [{"result":"success","cardType":"maestro","type":"bankcard","operation":"payout"}]	Remove
Qiwi	Edit metric Remove	BankCardMaestroPurchase filtered by [{"result":"success","cardType":"maestro","type":"bankcard","operation":"purchase"}]	Remove
Yandex Direct	Edit metric Remove	BankCardMasterCardPayout filtered by [{"result":"success","cardType":"mastercard","type":"bankcard","operation":"payout"}]	Remove
Yandex via Accentpay	Edit metric Remove	BankCardMasterCardPurchase filtered by [{"result":"success","cardType":"mastercard","type":"bankcard","operation":"purchase"}]	Remove
Comepay via Accentpay	Edit metric Remove	BankCardsAccentpayPayout filtered by [{"result":"success","merchantId":"84","type":"bankcard","operation":"payout"}]	Remove
Comepay Terminals	Edit metric Remove	BankCardsAccentpayPurchase filtered by [{"result":"success","merchantId":"84","type":"bankcard","operation":"purchase"}]	Remove
WebMoney Direct	Edit metric Remove	BankCardsCredoraxPayout filtered by [{"result":"success","merchantId":"70","type":"bankcard","operation":"payout"}]	Remove
WebMoney via DO	Edit metric Remove	BankCardsCredoraxPurchase filtered by [{"result":"success","merchantId":"70","type":"bankcard","operation":"purchase"}]	Remove
ecoPayz	Edit metric Remove	BankCardsOchopayPayout filtered by [{"result":"success","merchantId":"154","type":"bankcard","operation":"payout"}]	Remove
Moneta	Edit metric Remove	BankCardsOchopayPurchase filtered by [{"result":"success","merchantId":"154","type":"bankcard","operation":"purchase"}]	Remove
Soyuz Telekom	Edit metric Remove	BankCardSpainPayout filtered by [{"result":"success","jurisdiction":"spain","type":"bankcard","operation":"payout"}]	Remove

Рис 3.4 – Знімок екрану зі списком графіків і профілів

3.3 Перевірка працездатності

У рамках цього етапу тестування здійснювалася перевірка виконання усіх поставлених вимог, описаних в главі 2 [16].

Були перевірені вимоги, пред'явлені до наступних модулів :

- Модулю для підключення до черги повідомлень;
- Модулю для підключення до бази даних InfluxDB;
- Модулю для підключення до реляційної бази даних;
- Модулю для зберігання і зміни графіків і налаштувань;
- Модулю для обробки даних для подальшої побудови графіків;
- Модулю генерації сповіщень;
- Модулю для відправки е - mail сповіщень;
- Модулю для відправки повідомлень в Telegram.

Перевірені наступні функціональні можливості графічного інтерфейсу :

- Редагування графіків;
- Перегляд графіків в режимі реального часу;
- Редагування шаблонів сповіщень;
- Перегляд сповіщень, що спрацювали;
- Підписка на email- розсилку.

Тестування навантаження здійснювалося за допомогою розробленого робота на робочій машині з наступними характеристиками:

- 32Гб ОЗУ;
- 1Тб жорсткий диск 1000 IOPS;
- 4-х ядерний процесор Intel Core i5 - 6400T, 2.2 ГГц;
- OS Linux.

За результатами тестування можна зробити висновок, що заявлене у вимогах навантаження додаток витримує. При 1000 запитів в секунду споживання пам'яті і задіювання ЦПУ приведені на графіках в додатку Б.

РОЗДІЛ 4 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ

4.1 Опис впроваджуваної підсистеми

У зв'язку з вузькою спеціалізацією питання, не існує готового програмного забезпечення, яке б в повній мірі вирішувало всі необхідні задачі. Впровадження проекту підсистеми автоматизованої відновлення зображень, перш за все, націлене на зменшення витрат, підвищення ефективності роботи.

Створення програмного забезпечення вимагає одноразових витрат на розробку, придбання необхідних технічних засобів і поточних витрат на функціонування. Економія від функціонування комплексу визначається з урахуванням витрат на його експлуатацію. Відношення цієї економії до витрат на створення підсистеми моніторингу характеризує економічну ефективність капітальних вкладень. Економічні показники визначаються за діючими на момент розрахунку оптовими цінами, тарифами і ставками заробітної плати в Україні.

4.2 Розрахунок витрат на створення й експлуатацію -програмного комплексу

Витрати на розробку системи складаються з витрат на заробітну плату розробника, на амортизацію ЕОМ, на якій виконується розробка програмного забезпечення, на експлуатацію цієї ЕОМ, на засоби розробки та витрат на матеріали і комплектуючі.

Розробка виконується протягом двох місяців фахівцем І категорії, місячний оклад якого дорівнює місячній мінімальній зарплаті в Україні, яка з 01.01.2018 складає 3723 грн/міс. [27]. Додаткова заробітна плата складає 20 % від основної. Виходячи з цього, основна і додаткова заробітна плата розроблювача системи 4467,60 грн/міс., вартість сучасної ЕОМ складає 7610 грн [28].

При вартості електроенергії, рівної 167,514 коп. за 1 кВт • год грн [26], розраховується вартість розробки комплексу.

Витрати на допоміжні матеріали наведені у табл. 4.1.

Таблиця 4.1

Витрати на допоміжні матеріали

Статті витрат	Сума, грн
Папір 500 арк. 80 г/см ²	83,00
Заправлення картриджа до струйного принтера	79,00
Послуги Інтернет	100,00
Непередбачені витрати	25,00
Разом допоміжні матеріали	287,00

Вартість апаратно-програмного комплексу розраховуємо по формулі (4.1):

$$C_{\text{пр}} = (V_{\text{зп}} + V_{\text{сф}} + V_{\text{зг}} + V_{\text{е}}) * T + V_{\text{м}}, \quad (4.1)$$

де $V_{\text{зп}}$ – основна і додаткова заробітна плата обслуговуючого персоналу, грн;

$V_{\text{св}}$ – єдиний соціальний внесок (ЄСВ) 22 % від основної і додаткової заробітної плати [29], грн;

$V_{\text{зг}}$ – загальногосподарські витрати (10 % від основної заробітної плати), грн;

$V_{\text{е}}$ – витрати на електроенергію, грн;

T – тривалість розробки, міс.;

$V_{\text{м}}$ – витрати на основні і допоміжні матеріали, грн.

$V_{\text{е}}$ при споживанні потужності апаратно-програмним комплексом

$$P = U_{\text{сп}} * I \quad (4.2),$$

де $U_{\text{сп}}$ – напруга електроживлення, В, $U_{\text{сп}} = 3,3$ В;

I – струм, мА, $I = 200..300$ мА.

$$P = 3,3 * 300 \approx 1 \text{ Вт},$$

При тривалості роботи на місяць, рівної $20 \cdot 8 = 160$ годин, вартості кіловат-години електроенергії 1,68 грн [26] витрати на роботу підсистеми моніторингу становлять:

$$Be = 160 \cdot 1,68 \cdot 0,001 = 0,27 \text{ грн.}$$

Відповідно до формули (4.1) вартість апаратно-програмного комплексу:
 $С_{пр} = (4467,60 + 982,87 + 372,30 + 0,27) \cdot 2 + 287,00 + 639,00 = 1257,08 \text{ грн.}$

Результати розрахунків зведені у табл. 4.2.

Таблиця 4.2

Витрати на розробку системи

Найменування витрат	Одиниця виміру	Кількість
Тривалість розробки	міс.	2
Основна і додаткова заробітна плата	грн/міс.	4467,60
Відрахування в соціальні фонди	грн/міс.	982,87
Загальногосподарські витрати	грн/міс.	372,30
Витрати на допоміжні матеріали	грн	287,00
Витрати на комплектуючі	грн	639,00
Витрати на електроенергію	грн/міс.	0,27
РАЗОМ:		12572,08

Експлуатаційні витрати для ЕОМ розраховуємо по формулі (4.3):

$$В_{зр} = А_{об} + В_{м} + В_{е}, \quad (4.3)$$

де $A_{об}$ – амортизаційні відрахування, грн;

$В_{м}$ – річні витрати на основні і допоміжні матеріали, грн;

$В_{е}$ – витрати на електроенергію, грн;

Амортизаційні відрахування на обладнання групи 4 основних засобів (ОЗ) за прямолінійним методом нарахування складають [30]:

$$A_{об} = C_a : T,$$

де C_a – вартість об'єкта ОЗ, що амортизується;

T – термін корисного використання об'єкта ОЗ (років), для ЕОМ $T = 5$ р.

$$A_{об} = 7610 : 5 = 1522,00 \text{ грн.}$$

У масштабах підприємства річні витрати на основні і допоміжні матеріали (носії, папір, тощо) визначаються в розмірі 5 % вартості основного устаткування:

$$B_m = 7610 * 0,05 = 380,00 \text{ грн.}$$

Річний обсяг робіт ЕОМ у годинах визначається в такий спосіб:

$$\Phi_m = 253,3 * T_z, \quad (4.4)$$

де T_z – середнє денне завантаження устаткування (близько 4 годин);

253,3 – середня кількість робочих днів у році.

Отже, річний обсяг роботи ЕОМ складе:

$$\Phi_m = 253,3 * 4 = 1013 \text{ годин.}$$

Витрати на електроенергію B_e при 1013 годинах роботи устаткування в рік та потужністю ЕОМ 750 Вт складуть

$$B_e = 1013 * 1,68 * 0,75 = 1276,38 \text{ грн.}$$

Експлуатаційні витрати для ЕОМ за рік складуть:

$$B_{зр} = 1522 + 380 + 1276,38 = 3178,38 \text{ грн.}$$

Отже, у перший рік витрати на створення й експлуатацію підсистеми моніторингу автоматизованої системи управління освітленням складуть:

$$B_{се} = 12572,08 + 3178,38 = 15750,46 \text{ грн.}$$

4.3 Економічна ефективність розробки та впровадження

Основним показником економічної ефективності функціонування є зменшення витрат, підвищення ефективності роботи.

За даними п. 3.3, за тиждень (5 робочих днів) завдяки впровадження проекту тільки в одному приміщенні підприємства зменшуються витрати на спожиту електроенергію на 9 грн 65 коп., або за місяць (20 днів) приблизно 38 грн 60 коп. (див. табл. 4.1).

Крім того, впровадження апаратно-програмного комплексу зменшує кількість працівників на 1 штатну посаду

Зарплата 1 працівника на рік складає (із розрахунку на рівень мінімальної зарплати в Україні з 01.01.2018 у сумі 3723 грн/міс. [27].):

$$3723,00 * 12 = 44676,00 \text{ грн.}$$

Розрахуємо пряму економічну ефективність від використання підсистеми моніторингу автоматизованої системи управління освітленням.

Економічний ефект першого року розраховується за формулою (4.5):

$$A_{\text{екон.эф.}} = \Delta C_n - E_n * k, \quad (4.5)$$

де ΔC_n – вивільнені кошти в підприємства з 20 приміщень після впровадження підсистеми моніторингу автоматизованої системи управління освітленням (772,00 грн/міс. * 12 міс. = 9264,00 грн) та скорочення посади «Енергоінспектор» (44676,00 грн) мінус експлуатаційні витрати (3178,38 грн.):

$$\Delta C_n = 9264,00 + 44676,00 - 3178,38 = 50761,62 \text{ грн.}$$

E_n – коефіцієнт ефективності (дорівнює коефіцієнту амортизації), $E_n = 0,2$;

k – одноразові витрати на впровадження продукту, $k = 12572,08$ грн.

$$A_{\text{екон.эф.}} = 50761,62 - 0,2 * 12572,08 = 48247,20 \text{ грн.}$$

Строк окупності системи розраховується по формулі (4.6):

$$T = \frac{k}{\Pi}, \quad (4.6)$$

де Π – вивільнені кошти після впровадження комплексу (53940,00 грн);

k – одноразові витрати на впровадження системи (12572,08 грн).

$$T = \frac{12572,08}{53940,00} = 0,233 \text{ (року).}$$

Отже строк окупності підсистеми моніторингу автоматизованої системи управління освітленням складає приблизно 2 місяці.

Економічний ефект проявляється в отриманні економії витрат шляхом зменшення витрат на освітлення виробничих приміщень.

З обрахованих вище даних видно, що розроблені апаратно-програмний комплекс та веб-інтерфейс для підсистеми моніторингу автоматизованої системи управління освітленням з точки зору річних експлуатаційних даних може мати практичне застосування.

Реалізація даного апаратно-програмного продукту є доцільною, про що свідчать отримані значення періоду окупності та показників ефективності

РОЗДІЛ 5 ОХОРОНА ПРАЦІ

5.1 Основні положення Закону України «Про охорону праці»

Аналіз стану охорони праці в господарській діяльності України свідчить, що система керування цією важливою сферою трудових відносин, форми й методи роботи не відповідали тим процесам, які почали набирати сили у напрямі реформування економіки та всієї системи державного та господарського керування. Методи адміністративно-командного впливу на посадових осіб та працівників за порушення вимог охорони праці вже не діяли, а інших важелів впливу не було. Трудова, виконавська, технологічна дисципліни істотно знижувалися. Невизначеність обов'язків та повноважень з охорони праці новостворюваних структур у процесі роздержавлення, приватизації та поступової відмови від галузевого принципу керування господарською діяльністю ще більше ускладнювала стан справ. Негативний вплив справляла і відсутність законодавчо закріплених обов'язків з охорони праці для органів державної виконавчої влади різного рівня - від Уряду до державних адміністрацій областей, районів, міст та інших територіальних формувань. Тому прийняття Закону України «Про охорону праці» (надалі - Закон) було об'єктивно зумовлене ситуацією, що склалася на тій годині у суспільстві [31].

Специфічною особливістю українського Закону, що регламентує правову основу охорони праці, є високий рівень прав і гарантій працівників. Уперше в історії держави працівнику було надано право відмовитися від дорученої роботи, якщо створилася виробнича ситуація, небезпечна для його життя чи здоров'я, або для людей, які його оточують, і навколишнього природного середовища. Розширено права працівників у соціальних гарантіях відшкодування збитків у випадку ушкодження їх здоров'я на виробництві. Передбачається нова система фінансування охорони праці, формування системи страхування від нещасних випадків і профзахворювань, посилюється

централізація планування. Договірне регулювання з питань охорони праці поставлене на високий рівень. Передбачається значна доля громадських інституцій у цьому процесі.

З позицій законодавчої регламентації прав і гарантій працівників у сфері охорони праці та їх забезпечення Закон України «Про охорону праці» та нормативно- правові акти щодо його реалізації взяли високу оцінку експертів Міжнародної підприємства праці.

До позитивних аспектів Закону України «Про охорону праці», безперечно, належить закріплення за державою функції нагляду за охороною праці.

У умовах роздержавлення, приватизації, утворення великої кількості суб'єктів підприємницької діяльності з різними формами недержавної власності роль держави у вирішенні завдань охорони праці суттєво зростає. Держава виступає гарантом створення безпечних та нешкідливих умів праці для працівників підприємств, установ, організацій усіх форм власності.

5.2 Організація охорони праці на підприємстві

У організації охорони праці на підприємстві беруть доля роботодавці, їх заступники, головні спеціалісти, керівники виробничих дільниць, окремих структурних підрозділів та служб, профспілки та інші органи, що певним чином впливають на організацію охорони праці [32].

Основним завданням з питань охорони праці на підприємстві є створення здорових і безпечних умів праці. Цього можна досягти:

- навчанням всіх працюючих на підприємстві, перевіркою їх знань та пропагандою охорони праці;
- розробкою і виконанням комплексних(перспективних), річних та оперативних планових заходів з охорони праці;
- аналізом показників і причин виробничого травматизму та захворювань;

- оперативним контролем стану охорони праці на підприємстві і негайним усуненням шкідливостей та небезпек, виявлених на робочих місцях;
- проведенням паспортизації санітарно - технічного стану виробничих приміщень, технологічного обладнання та окремих робочих місць;
- впровадженням заходів морального і матеріального заохочення за зразковий стан охорони праці на робочому місці, дільниці, структурному підрозділі;
- проведенням спеціальних заходів з охорони праці жінок та молоді, виховної роботи з питань охорони праці та трудової дисципліни, а також притягненням до відповідальності осіб, які руйнували існуючі норми і правила охорони праці;
- забезпеченням усіх працюючих необхідними захисними засобами згідно з існуючими нормами.

Виконання цих заходів необхідно здійснювати на основі новітніх досягнень науки та передового досвіду, включаючи технічні засоби інформатики, спеціальні засоби сигналізації, блокування та ін.

5.3 Аналіз шкідливих та небезпечних факторів, які супроводжують роботові програміста

Використання персональних електронно-обчислювальних машин повинне здійснюватися у відповідності з НПАОП 0.00-1.28-10 «Правила охорони праці під година експлуатації електронно-обчислювальних машин» [33] та Державних санітарних правилах і нормах роботи з візуальними дисплейними терміналами(ВДТ) електронно-обчислювальних машин(ДСанПіН 3.3.2.007-98), затверджених Постановою Головного державного санітарного лікаря України від 10 грудня 1998 р. № 7 [34]. Відповідно до встановлених гігієнічно - санітарних вимог, визначених у цих документах, роботодавець зобов'язаний забезпечити в приміщеннях з ВДТ оптимальні параметри виробничого середовища(таблиця. 5.1 - таблиця. 5.4).

Таблиця 5.1

Норми мікроклімату для приміщень з ВТД

Пора року	Категорія робіт	Температура повітря, $^{\circ}\text{C}$, не більше	Відносна вологість повітря %	Швидкість руху повітря, м/с
Холодна	Легка - 1а	22...24	4...6	0,1
	Легка - 1б	21...23	4...6	0,1
Тепло	Легка - 1а	23...25	4...6	0,1
	Легка - 1б	22...24	4...6	0,2

Таблиця 5.2

Рівні іонізації повітря приміщень при роботі на ВДТ

Рівні	Число іонів в 1 см ³ повітря	
	п+	п-
Мінімально необхідні	400	600
Оптимальні	1500–30000	3000–5000
Максимально допустимі	50000	50000

Таблиця 5.3

Допустимі рівні звуку, еквівалентні рівні звуку і рівні звукового тиску в октавних смугах частот

Вид трудової діяльності	Рівні звукового тиску в дБ в октавних смугах із середньгеометричними частотами, Гц								
	63	125	250	500	1000	2000	4000	8000	Рівні звуку еквівалентні рівні звуку дБА/дБАекв.
Програмісти ЕОМ	7	61	54	49	45	42	40	38	50
Оператори в залах обробки інформації на ЕОМ та оператори комп'ютерного набору	83	74	68	63	60	57	55	54	65

Продовження таблиці 5.3

У приміщеннях для розташування шумних агрегатів ЕОМ	91	83	77	73	70	68	66	64	75
---	----	----	----	----	----	----	----	----	----

Таблиця 5.4

Допустимі параметри електромагнітних випромінювань і електричного поля

Види поля	Допустимі параметри поля		Допустима поверхнева щільність потоку енергії(інтенсивність потоку енергії), Вт/м ²
	за електричною складовою(Е), В/м	за магнітною складовою(Н), А/м	
Напруженість електромагнітного поля при частоті: 6 кГц...3 МГц	50	5	—
3 МГц...30 МГц	2	—	—
30 МГц...5 ГГц	—	—	10
Електромагнітне поле оптичного діапазону в ультрафіолетовій частині спектру: УФ-С(220...280 нм)			0,001
УФ-В(280...320 нм)			0,01
УФ-А(320...400 нм)			10,0
у інфрачервоній частині спектру: 0,76.10,0 мкм			35,0...70,0
Напруженість електричного поля ВДТ			20 У/м

5.4 Визначення категорії праці за енерговитратами з урахуванням особливості роботи за комп'ютером

Категорії праці за енерговитратами з урахуванням особливості роботи за комп'ютером визначаються відповідно до міждержавного ГОСТ 12.1.005-88

«ССБТ. Загальні санітарно-гігієнічні вимоги до повітря робочої зони» [35],
чинного в Україні станом на 18.02.2018(таблиця. 5.5).

Таблиця 5.5

Загальні енерговитрати організму

Категорія робіт	Джоул ь/з(Вт)	Ккал/рік	Характеристика робіт	Професії (приклади)
Легка - Іа	105– 140	90–120	Роботи, які виконуються сидячи та не потребують фізичного напруження	Управлінець, оператор ЕОМ
Легка - Іб	141– 175	121–150	Роботи, які виконуються сидячи, стоячи - супроводжуються деяким фізичним напруженням	Інженерно - технічний персонал
Середньої важкості - Іа	176– 232	151–200	Роботи, які пов'язані з постійним ходінням, переміщенням дрібних(до 1 кг) виробів або предметів у положенні стоячи або сидячи - потребують певного фізичного напруження	Працівники ремонтних майстерень
Середньої важкості - Іб	233– 290	201–250	Роботи, які виконуються стоячи, пов'язані з ходінням, переміщенням невеликих(до 10 кг) вантажів - супроводжуються помірним фізичним напруженням	Зварювальники
Важка - ІІІ	291– 349	251–300	Роботи, які пов'язані з постійними переміщеннями, перенесенням значних(понад 10 кг) вантажів - потребують великих фізичних зусиль	Вантажники, різнороби, тваринники

5.5 Розробка заходів щодо зменшення впливу шкідливих небезпечних факторів. Техніка безпеки на робочому місці

5.5.1 Засоби регулювання метеорологічних умів в приміщеннях, де працюють програмісти

Параметри мікроклімату можуть мінятися в широких межах, у тієї ж години як необхідною умовою життєдіяльності людини є підтримка постійності температури тіла завдяки терморегуляції, тобто здатності організму регулювати віддачу тепло в навколишнє середовище. Принцип нормування мікроклімату - створення оптимальних умів для теплообміну тіла людини з навколишнім середовищем.

Обчислювальна техніка є джерелом істотних тепловиділень, що може привести до підвищення температури і зниження відносної вологості у приміщенні. У санітарних нормах «Гігієнічні вимоги до мікроклімату виробничих приміщень»(СанПіН 2.2.4.548-96) встановлені величини параметрів мікроклімату, що створюють комфортні умови [36]. Ці норми встановлюються в залежності від шмагай року, характеру трудового процесу і характеру виробничого приміщення(таблиця. 5.6 - таблиця. 5.7).

Об'єм приміщень, в яких розміщені працівники обчислювальних центрів, відповідно до вимог п. 1.12 НПАОП 0.00-1.28-10 не повинний бути меншим 20,0 м³ на людину з урахуванням максимального числа одночасно працюючих в зміну [33].

Таблиця 5.6

Параметри мікроклімату для приміщень де встановлені комп'ютери

Період року	Параметр мікроклімату	Величина
Холодний	Температура повітря у приміщенні	22...24°C
	Відносна вологість	40...60 %
	Швидкість руху повітря	до 0,1 м/с
Теплий	Температура повітря у приміщенні	23 ... 25°C
	Відносна вологість	40 ... 60 %
	Швидкість руху повітря	0,1 ... 0,2 м/с

Таблиця 5.7

Норми подачі свіжого повітря в приміщення де розташовані комп'ютери

Характеристика приміщення	Об'ємна витрата подається в приміщення свіжого повітря, м3/на одну людину в годину
Об'єм до 20 м3 на особу	Не менше 30
20 ... 40 м3 на особу	Не менше 20
Більш 40 м3 на особу	Природна вентиляція

Для забезпечення комфортних умов використовуються як організаційні методи(раціональна організація проведення робіт поклад від шмагай року і доби, чергування праці і відпочинку), так і технічні засоби(вентиляція, кондиціонування повітря, опалювальна система).

5.5.2 Визначення розряду зорової праці відповідно нормативним вимогам

Правильно спроектоване і виконане виробниче освітлення покращує умови зорової роботи, знижує стомлюваність, сприяє підвищенню продуктивності праці, благотворно впливає на виробниче середовище, надаючи позитивну психологічну дію на працюючого, підвищує безпеку праці і знижує травматизм.

Недостатність освітлення приводить до напруги зору, ослабляє увагу, приводить до настання передчасної стомленості. Надмірно яскраве освітлення викликає засліплення, роздратування і різь в очах. Неправильний напрямок світла на робочому місці може створювати різкі тіні, відблиски, дезорієнтувати працюючого. Всі ці заподій можуть призвести до нещасного випадку або профзахворювань, тому такий важливий правильний розрахунок освітленості.

Існує три відіа освітлення - природне, штучне і поєднане(природне і штучне разом).

Природне освітлення - освітлення приміщень денним світлом, що потрапляє через світлові прорізи в зовнішніх конструкціях, що огорожують

приміщення. Природне освітлення характеризується тим, що змінюється в широких межах від години покладу дня, шмагай року, характеру області і ряду інших чинників.

Штучне освітлення застосовується при роботі в темний годину доби і втягай, коли не вдається забезпечити нормовані значення коефіцієнта природного освітлення(похмура погода, короткий світловий день). Освітлення, при якому недостатнє за нормами природне освітлення доповнюється штучним, називається змішаним освітленням.

Штучне освітлення підрозділяється на робоче, аварійне, евакуаційне, охоронне. Робоче освітлення, у свою чергу, може бути загальним або комбінованим. Загальне - освітлення, при якому світильники розміщуються у верхній зоні приміщення рівномірно, або, як розташоване устаткування. Комбіноване - освітлення, при якому до загального додається місцеве освітлення.

У приміщеннях обчислювальних центрів необхідно застосувати систему комбінованого освітлення [16, 37].

При виконанні робіт категорії високої зорової точності(найменший розмір об'єкту розрізнення 0,3 ... 0,5 мм) величина коефіцієнта природного освітлення(КПО) повинна бути не нижче 1,5 %, а при зоровій роботі середньої точності(найменший розмір об'єкту розрізнення 0,5 ... 1,0 мм) КПО повинний бути не нижче 1,0 %. В якості джерел штучного освітлення звичайно використовуються люмінесцентні лампи типу ЛБ, або ДРЛ, які попарно об'єднуються у світильники, які повинні розташовуватися рівномірно над робочими поверхнями.

Вимоги до освітленості в приміщеннях, де встановлені комп'ютери, наступні: при виконанні зорових робіт високої точності загальна освітленість повинна складати 300 лк, а комбінована - 750 лк; аналогічні вимоги при виконанні робіт середньої точності - 200 і 300 лк відповідно.

Крім того усе поле зору повинне бути освітлено достатньо рівномірно - ця основна гігієнічна вимога. Іншими словами, ступінь освітлення приміщення

і яскравість екрану комп'ютера повинні бути приблизно однаковими, оскільки яскраве світло в районі периферійного зору значно збільшує напруженість очей і, як наслідок, призводить до їх швидкої стомлюваності.

5.5.3 Допустимі рівні шуму та вібрації на робочих місцях з урахуванням застосування розмножувальної техніки

Шум погіршує умови праці надаючи шкідливу дію на організм людини. Працюючі в умовах тривалої шумової дії випробовують дратівливість, головні болі, запаморочення, зниження пам'яті, підвищену стомлюваність, зниження апетиту, біль у вухах і т. п. Такі порушення в роботі ряду органів і систем організму людини можуть викликати негативні зміни в емоційному стані людини аж до стресових. Під впливом шуму знижується концентрація уваги, порушуються фізіологічні функції, з'являється втома у зв'язку з підвищеними енергетичними витратами і нервово-психічним напруженням, погіршується мовна комутація. Все це знижує працездатність людини та її продуктивність, якість і безпеку праці. Тривала дія інтенсивного шуму (вище 80 дБА) на слух людини приводить до його часткової або повної втрати [38].

У таблиці. 5.8 вказані граничні рівні звуку поклад від категорії тяжкості і напруженості праці, що є безпечними відносно збереження здоров'я і працездатності.

Таблиця 5.8

Граничні рівні звуку на робочих місцях(дБ)

Категорія напруженості праці	Категорія важкості праці			
	I. Легка	II. Середня	III. Важка	IV. Дуже важка
I. Малонапружений	80	80	75	75
II. Помірно напружений	70	70	65	65
III. Напружений	60	60	—	—
IV. Дуже напружений	50	50	—	—

Рівень шуму на робочому місці математиків - програмістів і операторів відеоматеріалів не повинний перевищувати 50 дБА, а в залах обробки

інформації на обчислювальних машинах - 65 дБА. Для зниження рівня шуму стіни і стелючі приміщень, де встановлені комп'ютери, можуть бути облицьовані звукопоглинальними матеріалами. Рівень вібрації в приміщеннях обчислювальних центрів може бути понижень шляхом встановлення устаткування на спеціальні віброізолятори.

5.5.4 Потужність електричних приладів за ступенем небезпеки

Більшість вчених вважають, що як короткочасний, так і тривалий вплив усіх видів випромінювання від екрану монітора є небезпечним для здоров'я персоналу, що обслуговує комп'ютери. Проте, вичерпних даних щодо небезпеки дії випромінювання від моніторів на працюючих з комп'ютерами не існує, і дослідження в цьому напрямі продовжуються .

Допустимі значення параметрів неіонізуючих електромагнітних випромінювань від монітора комп'ютера представлені в таблиці. 5.9.

Максимальний рівень рентгенівського випромінювання на робочому місці оператора комп'ютера звичайно не перевищує 10 мкбер/рік, а інтенсивність ультрафіолетового і інфрачервоного випромінювань від екрану монітора лежить в межах 10...100 мВт/м².

Для зниження дії цих видів випромінювання рекомендується застосовувати монітори на електронно-променевих трубках(англ. CRT - Cathode Ray Tube) або більш сучасними рідкокристалевими LCD - моніторами зі зниженим рівнем випромінювання(ТСО - 92 та вище, до ТСО'03, ТСО'04, ТСО'05, ТСО'06 включно), встановлювати на них захисні екрани, а також дотримуватися регламентованих режимів праці та відпочинку

Таблиця 5.9

Допустимі значення параметрів неіонізуючих електромагнітних випромінювань(відповідно до ДСанПіН 3.3.2-007-98)

Найменування параметра	Допустимі значення
------------------------	--------------------

Напруженість електричної складової електромагнітного поля на відстані 50см від поверхні відеомонітора	10 У/м
Напруженість магнітної складової електромагнітного поля на відстані 50см від поверхні відеомонітора	0,3 А/м
Напруженість електростатичного поля не повинна перевищувати: для дорослих користувачів	20 кВ/м
для дітей дошкільних установ і таких, які вчаться у середніх спеціальних і вищих навчальних закладів	15 кВ/м

5.5.5 Ергономічні вимоги до робочого місця програміста

Проектування робочих місць, забезпечених ВДТ, відноситься до числа важливих проблем ергономічного проектування в області обчислювальної техніки.

Робоче місце і взаємне розташування всіх його елементів повинне відповідати антропометричним, фізичним і психологічним вимогам. Велике значення має також характер роботи. Зокрема, при підприємства робочого місця програміста повинні бути дотримані наступні основні умови : оптимальне розміщення устаткування, що до складу робочого місця і достатній робочий простір, що дозволяє здійснювати всі необхідні рухи і переміщення.

Ергономічними аспектами проектування робочих місць з ВДТ, зокрема, є: висота робочої поверхні, розміри простору для ніг, вимоги до розташування документів на робочому місці(наявність і розміри підставки для документів, можливість різного розміщення документів, відстань від очей користувача до екрану, документу, клавіатури і т. д.), характеристики робочого крісла, вимоги до поверхні робочого столу, урегульованість елементів робочого місця.

Головними елементами робочого місця програміста є стіл і крісло. Основним робочим положенням є положення сидячи.

Ергономічні вимоги до робочого місця оператора ЕОМ визначені у ДСТУ ISO 9241-5 : 2004 «Вимоги до компонування робочого місця та до робочої пози» [39].

Робоча поза сидячи викликає мінімальне стомлення програміста. Рациональне планування робочого місця передбачає чіткий порядок і сталість розміщення предметів, засобів праці і документації. Ті, що потрібно для виконання робіт частіше, розташоване в зоні легкої досяжності робочого простору [40].

Положення екрану визначається:

- відстанню зчитування(0,6 ... 0,7 м);
- нижче горизонталі до центру екрану, причому екран(кутом зчитування, напрямком погляду на 20(перпендикулярно цьому напрямку;
- Повинна також передбачатися можливість регулювання екрану;
- по висоті +3 см;
- щодо вертикалі від мінус 10(до +20(нахилу;
- у лівому і правому напрямках.

Велике значення також надається правильній робочій позі користувача. При незручній робочій позі можуть з'явитися болі в м'язах, суглобах і сухожиллях. Вимоги до робочої пози користувача відеотермінала наступні :

- голова не повинна бути нахилена більш ніж на 20(;
- плечі повинні бути розслаблені;
- лікті - під кутом 80(... 100(.

РОЗДІЛ 6 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Вплив людини на навколишнє середовище та навпаки

Охорона навколишнього природного середовища є системою державних і суспільних заходів, направлених на збереження, відтворення і раціональне використання природних ресурсів і поліпшення полягання природного середовища, і є частиною прикладної екології.

Відносини у галузі охорони навколишнього природного середовища в Україні регулюються Законом України від 26.06.1991 № 1268 - XII «Про охорону навколишнього природного середовища» [41], а також розроблюваними відповідно до нього земельним, водним, лісовим законодавством, законодавством про надра, про охорону атмосферного повітря, про охорону і використання рослинного і тваринного світу та іншим спеціальним законодавством.

У міру прискорення темпів науково-технічного прогресу дія людей на природу стає все більш могутньою. Але це приводить до якісної зміни співвідношення сил між суспільством і природою. На сучасному етапі людство поставлено перед чинником виникнення в природі незворотних процесів, нових шляхів переміщення і перетворення енергії і речовини.

Останніми роками при проектуванні(конструюванні), виготовленні(будівництві) і експлуатації технічних і систем управління в різних сферах діяльності надзвичайно широко застосовуються персональні комп'ютери і всілякі комп'ютерні програми.

Основну загрозу для навколишнього середовища в межах галузі розробки апаратно-програмного забезпечення становлять комплекси на базі ЕОМ, а саме електромагнітні випромінювання від них. Тому їх рівень обмежується у відповідних стандартах [42]. Але, на шкода, вплив випромінювання на оточуюче середовище досліджений недостатньо, тому рівні обмеження суттєво відрізняються між країнами.

Дослідження останніх років показали, що електромагнітні випромінювання вищезгаданих електронних пристроїв містять торсіонну компоненту, котра несе інформацію про процеси, що відбуваються в тому чи іншому електронному пристрої [43].

Торсіонні поля мають високу проникаючу здатність і їх неможливо екранувати. Вплив торсіонного поля може проявлятися в деякому фізичному дискомфорті, занепаді сил, іноді больових відчуттях. Можна сказати, що люди старшого віку більш чутливі, ніж молоді, жінки більш чутливі, ніж чоловіки. Однак, найбільш об'єктивним індикатором впливу ТП на живе є поведінка тварин. Тварини добрі відчують області поширення лівого торсіонного(інформаційного) поля і намагаються уникати їх.

Працюючий комп'ютер створює електромагнітне поле, що шкідливо діє на організм людини. Це поле може викликати радіоперешкоди, тобто заважати роботі телерадіоапаратури, що призводить до зниження надійності технічних систем або систем управління, до збільшення ризику виникнення аварійної ситуації у виробничому середовищі. Для забезпечення безпеки роботи з комп'ютером розроблені і повинні повсюдно застосовуватися стандарти на монітори, вимоги до приміщень для експлуатації комп'ютерів і до підприємства і устаткування робочих місць [33-40].

6.2 Сучасні технології утилізації відходів комп'ютерної техніки

6.2.1 Облік та переробка дорогоцінних металів зі зношених комп'ютерних компонентів

У зв'язку з тим, що вітчизняне виробництво сучасних компонентів інформаційних технологій дуже обмежено, із-за відсутності інформації про вміст дорогоцінних металів в елементах устаткування, строгий облік не представляється можливим і має бути покладений на фахівців експортних фірм. У умовах ринкової економіки підприємства мають бути самі зацікавлені

у вторинній переробці, що містять дорогоцінні метали вузлів і елементів за умови неможливості їх використання.

Сумарна маса дорогоцінних металів в ПЕОМ приблизно складає: золото - 0,22968 г; срібло - 5, 091336 р.

Технологічний процес виділення дорогоцінних металів з друкарських плат здійснюється за наступною схемою. Друкарські плати сортуються по переважанню в них кількості дорогоцінних металів, дробляться і подрібнюються, обпалюються і плавляться. У процесі випалення розкладанню піддається пластмасова основа, а основа дорогоцінних металів у вигляді металевих залишків відновлюється до оксидів. Металевий залишок подрібнюється, гранулюється, проходить магнітну сепарацію і відбувається відділення магнітних від немагнітних часток. Отриманий таким чином порошок, розділений після видів дорогоцінних металів, у вигляді гранул розплавляється в індукційних плавильних печах з подальшим розділенням шкідливого металу окремо.

6.2.2 Особливості утилізації елементів пам'яті

Використання комп'ютерів вимагає вирішення таких важливих питань, як утилізація відходів (мікросхеми з вмістом кольорових металів, плати, дискети).

При утилізації старих комп'ютерів відбувається їх розділ на шість фракцій: метали, пластмаси, штекери, дроти, батареї, скло. Жодна деталь не йде для повторного використання, оскільки не можна гарантувати їх надійність, але у формі вторинної сировини смороду йдуть на виготовлення нових комп'ютерів або інших пристроїв.

Детально розглянемо декілька прикладів переробки відходів обчислювальної техніки [44].

Гадолінієво - галеві гранати (ГГГ) використовуються у виробництві компонентів пристроїв, що запам'ятовують. У ході обробки біля 80 % вихідного матеріалу перетворюється на відходи або відбраковується. ГГГ

мають високу вартість і їх виділення з відходів представляє інтерес з економічної точки зору.

При здобутті досить чистих продуктів можливі повторне їх використання як вихідний матеріал. При цьому значно підвищується економічність виробництва заготовок з ГГГ. Під терміном «відходи» маються на увазі кристалічні залишки(залишки середовища для зростання кристалів, частини кристалів, виробництва, що утворюються на різних стадіях), а також дрібний порошок, що виходить при різанні, шліфовці, поліровці кристалів граната або подібних матеріалів.

Переробка цих відходів протягом ряду останніх років викликає труднощі і повністю не вирішена до цих пір. Існує процес, розроблений Е. Гуссетом(патент США 4-198231 від 15 квітня 1980 року, фірма «Свісс Алюмініум Лтд»., Швейцарія), призначений для виділення галію і гадолінію з відходів, що містять обидва ці елементи у вигляді оксидів або з'єднань, що перекладаються в оксиди. Відходи дрібно подрібнюються і потім розчиняються в сильних мінеральних кислотах. Гадоліній осідає з очищених розчинів у вигляді оксалату, галій виділяється в металевому вигляді електролітично. Електролітичне виділення галію може проводитися до виділення гадолінію у вигляді оксалату з розчину.

Розглянемо приклад проведення такого процесу. Відходи піддаються переробці, є залишки завантаження в пристрої для зростання кристалів, розколені частини кристалів зі всіх стадій переробки, дрібнозернисті порошки і пудри після операції різання, шліфовки і поліровки гранатів $GdGaO$. Подрібнений порошок після обробки кристалів ГГГ висушується при $1200^{\circ}C$ і потім нагрівається при $6000^{\circ}C$ протягом декількох годин для розкладання летких забруднень.

Дрібнозернисті відходи в кількості 1000 г розміром менше 40 мкм., що містять 34 % галію і 46 % гадолінію кип'ятяться із зворотним холодильником протягом двох годин в 2100 мл 35 % соляної кислоти. Це відповідає 99 % завантаження. Після кип'ятіння частина відходів, що не розчинилася,

фільтрується і промивається. Після висушування залишок важить 20 р. Залишки такого типу об'єднуються і піддаються повторній обробці кіп'ятінням. Фільтрат, що включає промивні води, об'ємом 2300 мл, обробляється 4000 г металевого галію при 500°C в течії 45 хвилин. Метав має максимально диспергувати.

У ході цієї операції благородніші елементи, присутні в розчині, виділяються і частково розчиняються в галії до його насичення і далі охолоджуються у вигляді інтерметалевих включень або в елементарній формі. Висадження можна проводити з меншою кількістю галію. Метав може періодично замінюватися на нову порцію до досягнення необхідної міри очищення. У результаті процесу очищення виходить розчин з вмістом галію 140 г/л і гадолінію 190 г/л.

Встановлюється величина $pH = 1$ шляхом додавання 900 мл 4 % розчину перекису водню для окислення домішок заліза. Осадження гадолінію проводиться при 500°C шляхом додавання 1500 г кристалічної технічної щавлевої кислоти $COH \cdot 2HO$; суспензія акуратно перемішується 12 годин для повторного осадження у вигляді оксалату гадолінію.

Оксалат гадолінію $Gd(CO) \cdot 10HO$ відділяється центрифугуванням, промивається 20 мл розбавленої щавлевої кислоти (6 г/л) і висушується при 1300°C; перетворення на оксид гадолінію досягається прожаренням при 8000°C. Чимдалі очищення може проводитися розчиненням в кислоті і осадженням у вигляді оксалату гадолінію. До рідини після обробки центрифугою і промивань облогу (3300 мл) додають 2300 г KOH при інтенсивному перемішуванні до $pH = 12,6$, 6000 мл розчину з вмістом галію 55 г/л і гадолінію 1 г/л піддається електролізу при 600 Із з використанням катода з нержавіючої сталі і графітового анода при щільності струму близько 0,1 А/кв. см Після 48 годин осідає 325 г галію і залишається розчин з вмістом 0,4 г/л, що не піддається подальшій переробці. Обкладень метав має чистоту 99,99 % і може бути безпосередньо перетворений в оксид.

6.2.3 Особливості утилізації елементів живлення

Використання акумуляторів різних типів, у тому числі і літієво - іонних, вже давно викликає занепокоєння органів охорони здоров'я та різних екологічних організацій. На шкода, у нашій країні проблема утилізації Li - іон акумуляторів стоїть найбільш гостро із заподій екологічної безграмотності населення, а також через відсутність налагодженої схеми утилізації, що контролюється державними органами. Використані літій - тіонілхлоридні елементи живлення можуть завдати шкоди довкіллю, а при тривалому неконтрольованому зберіганні стають вибухонебезпечними [45].

Вибухонебезпечною можна назвати ситуацію, при якій в результаті короткого замикання всередині акумулятора реагенти розігріваються до 450оС. У сучасних акумуляторах цього типу встановлена спеціальна система для підвищення рівня безпеки. Як тільки тиск досягає певної позначки в дію наводиться клапан, який випускає парі розчинника і зупиняє систему.

У таких батареях використовуються інертні матеріали(фосфат заліза, вуглець та інші) і речовини, що переробляються на 90 %.

Правильна утилізація Li - іон акумуляторів знижує ризик порушення роботи цієї системи. У великих літієвих акумуляторах витік тіону хлориду або діоксиду сірки при неписьменної утилізації стані причиною забруднення довкілля парами соляної кислоти, шкідливого впливу діоксиду сірки, продуктів горіння літію та інших неприємних наслідків.

Зазвичай процес утилізації Li - іон акумуляторів складається з декількох стадій:

- акумулятор розкривається в спеціальному сухому приміщенні і з нього витягується вміст;
- виробляється вимивання електроліту, що містить солі літію;
- далі відбувається поділ катодних і анодних пластинів;
- проводиться розчинення адгезії та видалення з пластинів близько 70 % матеріалу анода і катода;
- пластини міді і алюмінію підлягають переплавці;

– пластиковий корпус подрібнюється і переплавляється, далі його можна використовувати як добавку для покриттів автодоріг.

Таким чином, в середньому, близько 80 % матеріалів в акумуляторі, що підлягає утилізації, може використовуватися повторно в процесі виробництва.

Щоб уникнути небажаних наслідків, використані літієві елементи підлягають збору та транспортування на спеціалізовані підприємства, що мають ліцензію на утилізацію Li - ion акумуляторів.

6.3 Заходь для раціонального використання ресурсів на робочих місцях з автоматизованими комп'ютерними системами

При експлуатації персонального комп'ютера витрачаються наступні ресурси:

- електроенергія;
- папір для принтера;
- картриджі з фарбувальною стрічкою, чорнилами або тонером.

Для раціонального використання електроенергії не слід залишати включеним комп'ютер і принтер, якщо сморід не потрібні в даний годину.

Витрати на папір навряд чи вдасться скоротити удвічі, проте економія буде вельми істотною, якщо друкувати з двох сторін. Проблему з утилізацією паперових відходів може вирішити вторинна переробка.

Для повторного використання картриджів їх можна перезаправляти, і тоді картриджі можна буде використовувати 20-40 разів.

Сучасна технологія виготовлення елементів засобів обчислювальної техніки дозволяє досягти дуже низького рівня відмов елементів під година експлуатації. У зв'язку з цим відпадає необхідність проведення ремонтних робіт на місці експлуатації сучасних засобів обчислювальної техніки, при цьому, несправні мікросхеми, що містять дорогоцінні і рідкоземельні метали, також мають бути утилізовані.

Згідно з наказом Мінфіну від 22.06.2015 № 587, в Україні втратили чинність Інструкції 1998-2013 рр. про порядок одержання, використання, обліку та зберігання дорогоцінних металів і дорогоцінного каміння [46].

Таким чином, в Україні з 2015 р. облік дорогоцінних металів і дорогоцінного каміння проводитиметься на підставі внутрішніх інструкцій, що розробляються суб'єктами господарської діяльності самостійно, з урахуванням методичних рекомендацій, затверджених Мінфіном.

6.4 Організація утилізації компонентів комп'ютерної техніки в Україні

Утилізація літій - іонних батарей поки що не отримала в Україні такого широкого розвитку, як на Заході. Так чи інакше, але утилізація Li - ion акумуляторів на необхідному рівні вже налагоджено на окремих промислових підприємствах, які мають отримати відповідну ліцензію за кодами 38.32.1 «Утилізування відсортованих матеріальних ресурсів» ДК 016 : 2010 та 90513000-6 «Послуги з поводження із безпечними сміттям і відходами та їх утилізація» ДК 021 : 2015.

Збір використаних батарей у населення носити одноразовий характер, зазвичай ці предмети потрапляють на загальні сміттєві звалища, хоча природоохоронні органи настійно рекомендують громадянам здавати використані батареї в спеціалізовані пункти прийому. У якості позитивного прикладу можна навести встановлені в магазинах деяких стільникових операторів урни, куди люди можуть викинути використані акумулятори від своїх телефонів.

У сервісних центрах, що спеціалізуються на ремонті і технічному обслуговуванні ЕОМ, має бути організований збір і облік матеріалів, що містять кошовні метали, з подальшою обробкою цих матеріалів на спеціалізованих заводах з метою їх вилучення.

Однією з компаній з найбільшим досвідом на українському ринку в області утилізації промислових відходів, є ПП «ЮрЕко», яка має ліцензію на право поводження з небезпечними відходами [47].

У Україні також здійснюються такі види утилізації відходів, що утворюються за результатами діяльності в ІТ - галузі:

Утилізація моніторів

- утилізація ноутбуків;
- утилізація жорстких дисків;
- утилізація серверного обладнання;
- утилізація різноманітних електронних відходів;
- утилізація офісної техніки;
- утилізація картриджів;
- утилізація старих комп'ютерів(ПЕОМ);
- утилізація радіоелектронної апаратури;
- утилізація різноманітної комп'ютерної техніки;
- утилізація електронного обладнання;
- утилізація літій - іонних акумуляторів.

ВИСНОВКИ

В ході виконання магістерській роботи були вивчені технології, використовувані для розробки інформаційних систем моніторингу , отримані знання в області розробки веб-застосунків на мові Java , також досліджено методи аналізу метрик і тимчасових рядів.

Був складений список вимог до підсистеми, та проведений огляд існуючих програмних засобів, що виконують завдання. На підставі цього було прийнято рішення про створення власної підсистеми.

В процесі дослідження був виконаний огляд існуючих програмних засобів, що реалізують алгоритми для реконструкції реальних об'єктів. Був реалізований програмний продукт.

Розроблена система моніторингу скоротить фінансові і тимчасові витрати на аналіз виникаючих проблем в компаніях, що займаються електронною комерцією і проводять платежі з використанням зовнішніх платіжних систем.

Можливий розвиток розробленої програмної системи в окремий комерційний продукт, який універсально використовуватиметься компаній з подібною сферою роботи.

Реалізовано такі модулі системи моніторингу фінансових операцій:

- Модуль для підключення до черги повідомлень;
- Модуль для підключення до бази даних InfluxDB;
- Модуль для підключення до реляційної бази даних;
- Модуль для зберігання і зміни графіків і налаштувань;
- Модуль для обробки даних для подальшої побудови графіків;
- Модуль генерації сповіщень;
- Модуль для відправки e - mail сповіщень;
- Модуль для відправки повідомлень в Telegram.

Також Реалізований графічний інтерфейс для управління функціоналом розробленої програмної системи з наступними можливостями:

- Редагування графіків;

- Перегляд графіків в режимі реального часу;
- Редагування шаблонів сповіщень;
- Перегляд сповіщень, що спрацювали;
- Підписка на email- розсилку.

Проведено тестування розробленого ПО показали, що усі поставлені завдання виконані.

Розширення існуючої функціональності Telegram- бота, додавання команд для управління сповіщеннями безпосередньо з чату з ботом, вибір необхідних типів сповіщень безпосередньо кожним з користувачів і так далі

Отримані результати мають практичне значення.

Але необхідно заявити і про недоробки, серед яких недружелюбний графічний інтерфейс , неоптимізована робота програми і висока залежність якості реконструкції від вхідних даних. Треба зазначити, що отримана система показала себе як працездатне рішення і має право на подальше вдосконалення і більш детальне тестування. Все це говорить про те, що без сумніву дані отримані при виконанні магістерської роботи є предметом подальшого глибокого дослідження. Подальший розвиток проекту можливий в декількох варіантах:

- Розширення підтримуваних типів метрик;
- Додавання нових функцій агрегації;
- Додавання альтернативних методів аналізу тимчасових рядів і генерації сповіщень;
- Додавання нових каналів сповіщення

Виконано розрахунок економічної ефективності від розробки. Доведено економічне обґрунтування розробки такого програмного продукту. Розглянуто питання з охорони праці та охорони навколишнього середовища.

Результати роботи представлені у вигляді доповіді на конференції.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Опис ПО Zabbix // Оф. сайт Zabbix. URL: <http://www.zabbix.com/ru/product> (дата звернення 03.04.2017).
2. Опис ПО Grafana // Оф. сайт GrafanaLabs. URL: <https://grafana.com/grafana> (дата звернення 03.04.2017).
3. Опис ПО Nagios XI // Оф. сайт Nagios. URL: <https://www.nagios.com/products/nagios-xi> (дата звернення 03.04.2017).
4. Опис і документація ПО Prometheus // Оф. сайт Prometheus. URL: [https://prometheus.io/docs/introduction/getting started](https://prometheus.io/docs/introduction/getting_started) (дата звернення 03.04.2017).
5. Стаття Андрія Прохорова «Часовий ряд як об'єкт зберігання в СУБД» // Сайт Центру Інформаційних Технологій «ЦИТ Форум» URL : http://citforum.ru/seminars/cbd2001/day_2_5_informix.shtml (дата звернення 15.05.2017).
6. Дейт К. Джоуль. Введення в системи баз даних, 8-е видання; пер. з англ. М. : «Вільямс», 2017. 1328 с.
7. Опис можливостей InfluxDB // Оф. сайт InfluxData. URL: <https://www.influxdata.com> (дата звернення 17.05.2017).
8. Документація по InfluxDB версії 1.2 // Оф. сайт InfluxData. URL: <https://docs.influxdata.com/influxdb/v1.2> (дата звернення 17.05.2017).
9. Опис Telegram API // Оф. сайт Telegram. URL: <https://core.telegram.org/api> (дата звернення 12.04.2017).
10. Мишулина О. А. Статистичний аналіз і обробка тимчасових рядів. М.: МИФИ, 2004. 180 с.
11. Функції агрегації // ІТ енциклопедія CodeQ(itsCode). URL: <http://codeq.ru/code/Функции агрегации> (дата звернення 29.04.2017).
12. Клаптів А. Ю. Аналіз тимчасових рядів, курс лекцій. М.: Фізичний факультет МГУ. 113 с.

13. Розподіл Пуассона // Портал знань, Глобальний інтелектуальний ресурс. URL: <http://statistica.ru/theory/raspredelenie-puassona>(дата звернення 23.04.2017).

14. Офіційний репозиторій InfluxDB // Платформа для розробки GitHub. URL: <https://github.com/influxdata/influxdb>(дата звернення 17.05.2017).

15. Історія версій InfluxDB // Платформа для розробки GitHub. URL: <https://github.com/influxdata/influxdb/blob/master/CHANGELOG.md>(дата звернення 01.05.2017) .

16. Криспин Л. Гнучке тестування: практичний посібник для тестувальників ПЗ і гнучких команд; пер. з англ. М. : «Вільямс», 2010. 464 с.

17. Евгений Соколов. Технологии Java для разработки веб-приложений. 2003. [<http://www.techinfo.net.ru/docs/web/iavawebdev.html>.5 l.Neal Ford. Art of Java Web Development. Manning, 2004.]

18. Василий Стрельников. Основы Web-технологий: Учеб. пособие / ВолгГ-ТУ. Волгоград, 2002. - 72 с.

19. Комп'ютерний моніторинг і інформаційні технології [<http://megabook.ru/article/Web>] : матеріали студент. наук.-практ. конф., 25 квіт. 2005, Донецьк / Донец. нац. техн. ун-т, Каф. комп'ютер. систем моніторингу. – Текст. і граф. дані (250 МБ). – Донецьк, 2005.

20. Лафоре Р. Структуры данных и алгоритмы в Java. Классика Computers Science. 2-е изд. Питер, 2011. – 695 с.

21. Роздрібні тарифи на електроенергію для побутових споживачів на лютий 2018 року. Національна комісія, що здійснює державне регулювання у сферах енергетики та комунальних послуг (НКРЕКП) : веб-сайт. URL : <http://www.nerc.gov.ua/?id=30742> (дата звернення : 19.02.2018).

22. Мінімальна зарплата в Україні з 2000 по 2018 рр. Мінфін : веб-сайт. URL : <https://index.minfin.com.ua/labour/salary/min/> (дата звернення : 19.02.2018).

23. Компьютеры для работы и учебы. Rozetka : Интернет-супермаркет. URL : <https://hard.rozetka.com.ua/computers/c80095/filter/preset=workteaching;sort=cheap/> (дата обращения : 19.02.2018).

24. Пример №1 расчет налогов на зарплату в 2018 году. URL : <http://www.buhoblik.org.ua/rizni/2018/primer-nalogov-1.html> (дата обращения : 19.02.2018).

25. Амортизація основних засобів. Вісник. Офіційно про податки : офіц. видання Держ. фіскальн. служби України. URL : <http://www.visnuk.com.ua/ua/pubs/id/6707> (дата звернення : 19.02.2018).

26. Про охорону праці : Закон України від 14.10.1992 №2694-XII. Дата оновлення : 20.01.2018. URL : <http://zakon2.rada.gov.ua/laws/show/2694-12/page> (дата звернення : 19.02.2018).

27. Охрана праці (практикум) : навч. посіб. / за заг. ред. канд. техн. наук, доц. І. П. Пістуна. Львів : Тріада плюс, 2011. 436 с.

28. НПАОП 0.00-1.28-10. Правила охорони праці під час експлуатації електронно-обчислювальних машин : затв. наказом Держгірпромнагляду України від 26.03.2010 № 65. URL : <http://zakon3.rada.gov.ua/laws/show/z0293-10> (дата звернення : 19.02.2018).

29. ДСанПіН 3.3.2-007-98. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин : затв. постановою Голов. держ. саніт. лікаря України від 10.12.1998 № 7. URL : http://kharkiv.medprof.org.ua/uploads/media/S_Gig1.pdf (дата звернення : 19.02.2018).

30. ГОСТ 12.1.005-88. ССБТ. Общие санитарно-гигиенические требования к воздуху рабочей зоны [Дата введ. 1989-01-01]. Київ : Держспоживстандарт України, 2006. 77 с.

31. СанПіН 2.2.4.548-96. Гігієнічні вимоги до мікроклімату виробничих приміщень : затв. постановою Госкомсанепіднадзора Росії від

1 жовтня 1996 р. № 21. URL : <http://pid.atwebpages.com/sanpin-224548-gigienicheskie-trebovaniya.html> (дата звернення : 19.02.2018).

32. ДСН 3.3.6-039-99. Державні санітарні норми виробничої загальної та локальної вібрації : затв. постановою Голов. держ. саніт. лікаря України від 1 грудня 1999 р. № 39. URL : http://www.dnaop.com/html/1643/doc-ДСН_3.3.6.039-99 (дата звернення : 19.02.2018).

33. ДСТУ ISO 9241-5:2004. Ергономічні вимоги до роботи з відеотерміналами в офісі. Частина 5. Вимоги до компонування робочого місця та до робочої пози. Київ : Держспоживстандарт України, 2004. 27 с.

34. Охорона праці в офісі. Вимоги до робочого місця офісного працівника. Global Consulting Corporation. URL : <http://gc.ua/business-news/oxorona-praci-v-ofisi-vimogi-do-robochogo-miscya-ofisnogo-pracivnika/> (дата звернення : 19.02.2018).

35. Про охорону навколишнього природного середовища : Закон України від 26.06.1991 № 1268-ХІІ. Дата оновлення : 25.02.1994. URL : <http://zakon5.rada.gov.ua/laws/show/1268-12> (дата звернення : 19.02.2018).

36. Шорошев В. В., Пающик І. І. Фізична безпека комп'ютерних систем. Захист інформації : наук.-техн. журн. 2008. № 3. С. 4–10. URL : <http://jrn1.nau.edu.ua/index.php/ZI/article/viewFile/5394/6109> (дата звернення : 19.02.2018).

37. Цветков В. Я. Естественные и искусственные информационные поля. Международный журнал прикладных и фундаментальных исследований. 2014. №5, Ч. 2. С. 178–180.

38. Извлечение металлов и неорганических соединений из отходов URL : <http://www.kar-met.su/izvlechenie-metallov-i-neorganicheskikh-soedinenii-iz-otkhodov/izvlechenie-metallov-i-neorganicheskikh-soedinenii-iz-otkhodov-str141.html> (дата обращения : 19.02.2018).

39. Утилізація Li-ion акумуляторів. 2017, 20 лютого. URL : <http://akumulatory.com/utilizatsiya-li-ion-akumulyatoriv/> (дата звернення : 19.02.2018).

40. Утилізація оргтехніки та офісних меблів. Дозвільні документи Міністерства екології та природних ресурсів України. 2017, 20 червня. URL : <http://youreko.com.ua/> (дата звернення : 19.02.2018).

ТЕХНІЧНЕ ЗАВДАННЯ НА РОЗРОБКУ

1. Загальні відомості

1.1 Повне найменування системи та її умовне позначення

Повне найменування системи: «Система моніторингу фінансових операцій».

1.2 Найменування організації замовника та організацій-виконавців робіт :

Замовником системи є Комерційна фірма «XXX»

Адреса: м. Київ

1.3 Перелік документів, на основі яких створюється система

Підставою для розробки є наступний документ:

– завдання на магістерську роботу зі спеціальності 126 «Інформаційні системи та технології».

1.4 Планові строки початку та закінчення робіт зі створення системи

Плановий строк початку робіт зі створення системи – 16 жовтня 2023 року.

Плановий строк закінчення робіт зі створення системи – 15 грудня 2023 року

1.5 Порядок оформлення та пред'явлення замовнику результатів робіт зі створення системи

Система передається у вигляді комплексу на базі засобів обчислювальної техніки Замовника та Виконавця у встановлені строки. Приймання системи здійснюється комісією у складі вповноважених представників Замовника та Виконавця.

Порядок пред'явлення системи, її випробувань та остаточного приймання визначений у додатку В «Програма й методика випробувань» даної розрахунково-пояснювальної записки. Спільно з пред'явленням системи здійснюється здача розробленого Виконавцем комплексу документації.

2. Призначення та цілі створення системи

2.1 Призначення системи

Система призначена для зберігання й обробки даних, що містять інформацію про платіжні транзакції, що проводяться, їх аналізу і можливості сповіщення про нестандартну поведінку потоку транзакцій.

2.2 Цілі створення системи

Створення модернізованої системи має на меті вирішення наступних завдань:

- Контроль проведення транзакцій;
- Оповіщення р нестандартних ситуаціях;
- Графічне зображення даних
- Аналіз даних;

3. Характеристика об'єкта автоматизації

Комерційне підприємство проводить свою діяльність у сфері електронної комерції з постійним проведенням платіжних транзакцій в країнах по всьому світу, з використанням великої кількості зовнішніх платіжних систем.

На підприємстві існує комплекс, що надає комерційні послуги для клієнтів і інтегрований з великою кількістю зовнішніх платіжних систем. При обробці даних транзакцій можливе виникнення будь-яких помилок, через які транзакції можуть завершуватися некоректно або не завершуватися зовсім.

Основною проблемою при пошуку даних помилок є складність в поділі коректної поведінки системи, коли платежі дійсно не повинні відбуватися в силу певної логіки і функціональності яких умов зовнішнього світу, і поведінки, пов'язаного з реальними проблемами в логіці системи, зовнішніх платіжних сервісів, інфраструктури і т. д.

За пропонована система моніторингу проведення платіжних транзакцій, буде допомагати працівникам розділити транзакції на успішні і неуспішні та проаналізувати проблеми та наскільки коректна була поведінка платіжної системи.

Розроблений комерційний продукт повинен зміцнити лідируючі позиції підприємства на цільовому ринку,

4. Необхідні вимоги до використаних технологій:

- Для передачі метрик від програмної системи в моніторинг повинен використовуватися брокер черг повідомлень «RabbitMQ»;
- При використанні бази даних для зберігання тимчасових поруч повинна використовуватися «Influx Database»;
- При використанні реляційної бази даних, вибір повинен бути відданий на користь «Oracle Database»;
- Розробка серверної частини повинна проводитися на мові Java з використанням, при необхідності, Spring Framework (особливо, його модулів: «Spring Boot», «Spring Remote» і «Spring MVC»);
- Взаємодія між модулями через RMI;
- Збірка повинна проводитися з використанням «Maven»;
- Для розгортання веб-додатки повинен використовуватися контейнер «Apache Tomcat»;
- При розробці веб-інтерфейсу пріоритет повинен бути відданий таким технологіям:
 - o «Twitter Bootstrap» для візуального оформлення;
 - o «JavaScript» і «AngularJS» для розробки динамічної частини;
 - o «Google Charts» для відображення графіків.

В результаті проведеного аналізу існуючих рішень і технологій для Java, вибір був зроблений на наступних:

Мова програмування - Java 1.8 з використанням можливостей SE (Standard Edition) версії.

Для архітектури всього програми була обрана програмна платформа «Spring MVC Framework», що реалізує патерн модель-відконтроллер і дозволяє відокремлювати шари додатки за допомогою Javaаннотацій.

Для взаємодії з API InfluxDB обраний клієнт для мови Java «org.influxdb.influxdb-java», який реалізує HTTP / UDP API і дозволяє використовувати CRUD-операції. Даний клієнт доступний для підключення, як залежність Maven проекту.

Для реалізації взаємодії між модулями програми вибрано рішення «Spring Remote», що реалізує підхід RMI з використанням Hessian протоколу для передачі даних.

Для оформлення статичних частин веб-інтерфейсу використовується фреймворк «Twitter Bootstrap», зокрема, використовуються його CSS-стилі і JavaScript рішення.

Для відтворення графіків в веб-інтерфейсі використовується бібліотека «Google charts», що дозволяє відображати відразу кілька ліній на кожному з графіків.

Для побудова іншої динамічної логіки на клієнті використовується AngularJS в сукупності з JavaScript. Це дозволяє зменшити зв'язність коду, розбити логіку на частини відповідно до моделі MVC і використовувати підхід з так званої інверсії управління - DI.

5. Загальні вимоги до приймання робіт

Здача-приймання робіт проводиться поетапно, у відповідності з робочою програмою та календарним планом. Здача-приймання здійснюється комісією, до складу якої входять представники Замовника і Виконавця. За результатами приймання підписується акт приймальної комісії.

Кожна стадія розробки повинна бути представлена в зазначені рядки та узгоджена із замовником. Випробування системи відбувається згідно з програмою та методикою випробувань. У випадку виявлення недоліків усунення їх відбувається протягом тижня

Всі створювані в рамках цієї роботи програмні вироби (за винятком покупних) передаються Замовнику, як у вигляді готових модулів, так і у вигляді вихідних кодів, що подаються в електронній формі на стандартному машинному носії.

5.1 Статус приймальної комісії

Статус приймальної комісії визначається замовником до проведення випробувань.

6. Вимоги до документації

Разом з розробленими програмними виробами Виконавець повинен передати Замовникові наступний перелік відповідно оформлених документів:

- програма та методика випробувань, протокол випробувань системи;
- пояснювальна записка до проекту системи;
- опис застосування системи;
- керівництво з обслуговування
- інструкція користувача;
- опис програми;
- текст програми;

ІНСТРУКЦІЇ ПО ВИКОРИСТАННЮ РОЗРОБЛЕНОГО ПРОДУКТУ(ПІДГОТОВКА ДОКУМЕНТАЦІЇ, РОЗРОБКА РЕКОМЕНДАЦІЙ ПО ЕКСПЛУАТАЦІЇ)

Б.1 Установка і налаштування InfluxDB

Для установки потрібний завантажений з офіційного сайту дистрибутив під потрібну ОС, далі розглядатиметься установка дистрибутива під ОС Ubuntu. Завантажити дистрибутив можна по наступній команді, описаній в таблиці Б.1.

Таблиця Б.1

Завантаження дистрибутива

```
$ wget http://s3.amazonaws.com/influxdb/influxdb latest amd64.deb
```

Установка виконується за допомогою команди для пакетного менеджера, описаної в таблиці Б.2.

Таблиця Б.2

Установка дистрибутива

```
$ sudo dpkg -i influxdb _ latest _ amd64.deb
```

Після установки досить запустити InfluxDB командою, приведеною в таблиці Б.3.

Таблиця Б.3

Запуск InfluxDB

```
$ sudo /etc/init.d/influxdb start
```

Уся конфігурація InfluxDB зберігається у файлі /opt/influxdb/current/config.toml і підрозділяється на наступні розділи:

- Логгирование - рівень і шлях до теки для зберігання лівг;
- Налаштування доступу до GUI;
- Налаштування Influx API, що надається;
- Налаштування експорту даних від зовнішніх джерел;
- Налаштування для створення кластера;
- Налаштування випереджаючого введення журналу.

- Різні додаткові налаштування, що не входять ні в одну з категорій.

Усе інше можна зробити через веб-інтерфейс: створити нову базу даних, налаштувати періодичне видалення і так далі

Б.2 Конфігурація розробленого застосування

Усі потрібні для коректної роботи налаштування мають бути змінені в теці «config», загальною для кожного з модулів, файли з розширенням «.properties» можна змінювати на будь-якому етапі життя проекту, файли конфігурації контексту підлягають зміні тільки до складання проекту.

Допустимі налаштування:

- Налаштування модуля обмінника RabbitMQ :
- Налаштування для підключення; o Ім'я черги для читання; o Ім'я черги для збереження сповіщень.
- Налаштування модуль відправки повідомлень в Telegram:
- Токен для підключення до бота;
- Ідентифікатор групового чату для відправки повідомлень;
- Шаблони повідомлень.
- Налаштування БД :
- Дані для підключення до Oracle Database;
- Дані для підключення до InfluxDB;
- Налаштування модуля генерації сповіщень :
- Вірогідність, по якій розраховуватиметься максимальний час очікування наступної метрики;
- Кількість останніх метрик, по яких відбуватиметься розрахунок;
- Поріг спрацьовування сповіщень, в мс.
- Налаштування для веб-інтерфейсу:
- Налаштування щільності точок;
- Налаштування відображення функцій агрегації.
- Допустимі теги;
- Адреси підключення до кожного з модулів;
- Налаштування логгирования.