НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

До захисту допущено
В.о. завідувача кафедри

_____ **Микола ГРАЙВОРОНСЬКИЙ**
(підпис)

«_____»_____2021 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною программою «Системи, технології та математичні методи кібербезпеки»
спеціальності: 125 «Кібербезпека»

на тему: «Захищена автоматизація роботи в сфері фінансового менеджменту»

Виконав: здобувач вищої освіти **IV** курсу, групи ФБ-72
(шифр групи)

Христиченко Дмитро Миколайович
(прізвище, ім'я, по батькові) (підпис)

Керівник к.ф.-м.н., доцент Грайворонський Микола Владленович
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище, ім'я, по батькові) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Здобувач вищої освіти _____
(підпис)

Київ - 2021 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 125 «Кібербезпека»
Освітньо-професійна програма «Системи, технології та математичні методи кібербезпеки»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Микола ГРАЙВОРОНСЬКИЙ
(підпис)
«___»_____2021 р.

ЗАВДАННЯ
на дипломну роботу здобувачу вищої освіти

Христиченка Дмитра Миколайовича
(прізвище, ім'я, по батькові)

1. Тема роботи «Захищена автоматизація роботи в сфері фінансового менеджменту»,
керівник роботи Грайворонський Микола Владленович к.ф.-м.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від « » _____ 2021 р. №

2. Термін подання здобувачем вищої освіти роботи 07 червня 2021 р.

3. Вихідні дані до роботи: програмне забезпечення захищеної автоматизації роботи, яке можна використовувати у фінансових організаціях зі схожим направленням.

4. Зміст роботи: загальний огляд систем автоматизації робочих процесів, огляд існуючих механізмів та сервісів безпеки, програмна реалізація застосунку з використанням обраних технологій.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): презентація до захисту дипломної роботи, яка відображає ключові моменти виконаної роботи.

6. Дата видачі завдання 22.09.2020

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів дипломної роботи	Примітка
1	Отримання завдання	22.09.2020	виконано
2	Вивчення та аналіз літератури	22.09.2020 – 19.12.2020	виконано
3	Аналіз існуючих механізмів та сервісів безпеки	19.12.2020 – 17.02.2021	виконано
4	Набуття навичок інтеграції обраних механізмів та сервісів безпеки	17.02.2021 – 15.03.2021	виконано
5	Розробка програмного забезпечення	15.03.2021 – 10.04.2021	виконано
6	Відкрите тестування застосунку	10.04.2021 – 18.04.2021	виконано
7	Проходження переддипломної практики	12.04.2021 – 16.05.2021	виконано
8	Написання дипломної роботи	23.04.2021 – 02.06.2021	виконано
9	Отримання допуску до захисту	03.06.2021	
10	Захист дипломної роботи	16.06.2021	

Здобувач вищої освіти _____ Дмитро ХРИСТИЧЕНКО
(підпис) (Власне ім’я, ПРІЗВИЩЕ)

Керівник роботи _____ Микола ГРАЙВОРОНСЬКИЙ
(підпис) (Власне ім’я, ПРІЗВИЩЕ)

РЕФЕРАТ

Робота обсягом 61 сторінку складається з 3 розділів, містить 11 ілюстрацій, 2 таблиці та 19 літературних посилань.

Завданням роботи є огляд основних технологій автоматизації та підтримки бізнесу, огляд та вибір сервісів безпеки, технологій та технічних рішень, реалізація застосунку з використанням обраних технологій та сервісів безпеки, перевірка ефективності.

Мета даної дипломної роботи полягає в розробці застосунку для автоматизації роботи в сфері фінансового менеджменту з використанням сучасних методів захисту, що дозволяє залишати всі джерела інформації конфіденційними.

Об’єктом дослідження у даній роботі є існуючі механізми безпеки сервісів.

Предметом дослідження є розробка застосунку з використанням сучасних методів захисту та аналіз отриманих результатів.

Актуальність дипломної роботи полягає у тому, що комунікація між користувачем та компанією відбувається більш оперативно за рахунок чого можна обслуговувати більшу кількість клієнтів, що в свою чергу підвищить результативність роботи фінансової організації. Деяка робота зі статичними і динамічними даними стає автоматизованою, так як відправка звітів, розрахунок прибутковості не буде вимагати дій співробітника компанії. Користувач, коли йому зручно може запросити цю інформацію, при цьому всі джерела інформації залишаються конфіденційними що не порушує регламент роботи деяких компаній.

Відмінною рисою самого сервісу є те, що формування інвестиційного портфеля відбувається в режимі реального часу, при цьому актуальні дані автоматично оновлюються.

Методами дослідження у даній дипломній роботі є огляд літературних джерел за обраним направленням, вибір найбільш підходящих сервісів безпеки та технічних рішень, їх інтеграція та аналіз отриманих результатів.

Наукова новизна полягає в обґрунтованому виборі та комбінації різних механізмів безпеки сервісів в одному застосунку, які в подальшому можуть пропонуватись як готове рішення.

Практичне значення даної роботи полягає у тому, що отриманий застосунок у ході роботи може бути використаний у фінансових організаціях зі сходим направленням. Користувач буде потребувати мінімальну кількість часу на обслуговування своїх потреб.

Ключові слова: автоматизація, сервіс, захист, конфіденційність.

ABSTRACT

The work of 61 pages consists of 3 sections, contains 11 illustrations, 2 tables and 19 literature references.

The task of the work is to review the main technologies of automation and business support, review and selection of security services, technologies and technical solutions, implementation of the application using the selected technologies and security services, efficiency testing.

The purpose of this thesis is to develop an application to automate work in financial management using modern security methods, allows you to leave all sources of information confidential.

The object of the study in this work is the existing mechanisms of security services.

The subject of the study is to develop an application using modern methods of protection and analysis of the results.

The relevance of the thesis is that the communication between the user and the company is more rapid by which you can serve more customers, which in turn will increase the performance of the financial organization. Some work with static and dynamic data becomes automated, as the sending of reports, profitability calculation will not require the actions of an employee of the company. The user can request this information whenever he wants, and all sources of information will remain confidential without violating the regulations of some companies.

A distinctive feature of the service itself is that the formation of an investment portfolio is done in real time and the actual data is automatically updated.

Research methods in this thesis are literature review on the chosen direction, selection of the most suitable security services and technical solutions, their integration and analysis of obtained results.

The scientific novelty lies in the reasonable choice and combination of different service security mechanisms in a single application, which can then be offered as a

ready-made solution. Practical value of this work is that the received application in the course of work can be used in the financial organizations with the similar direction. The user will require a minimum amount of time to service their needs.

Keywords: automation, service, protection, privacy.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 9

Вступ 10

1 Системи автоматизації робочих процесів 13

1.1 Визначення робочого процесу 13

1.2 Переваги автоматизованих робочих процесів 14

1.3 Практичне застосування автоматизованих робочих процесів 17

1.4 Порівняння систем для автоматизації робочих процесів 19

Висновок до розділу 1 28

2 Огляд механізмів та сервісів безпеки 29

2.1 Google Drive в якості сервісу хмарного сховища 29

2.2 Реляційна база даних MySQL, як сервіс безпеки 31

2.3 Корпоративна платформа, як додатковий сервіс безпеки 33

2.4 Telegram Bot у якості клієнтської частини сервісу 34

2.5 Визначення та огляд політики безпеки 36

Висновок до розділу 2 39

3 Реалізація застосунку з використанням обраних технологій 40

3.1 Налаштування сервісу бази даних MySQL 40

3.2 Налаштування сервісу хмарного сховища Google Drive 41

3.3 Програмна реалізація застосунку 42

3.4 Аналіз отриманих результатів у ході публічного тестування 44

Висновки до розділу 3 45

Висновки 46

Перелік джерел посилань 47

Додаток А 50

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ПЗ – програмне забезпечення

WYSIWYG – What You See Is What You Get

AES – Advanced Encryption Standard

TLS – Transport Layer Security

ITU – International Telecommunication Union

PKI – Public Key Infrastructure

AWS – Amazon Web Services

SQL – Structured Query Language

IGE – Infinite Garble Extension

API – Application Programming Interface

JSON – JavaScript Object Notation

РСУБД – Реляційна система управління базами даних

ВСТУП

На сьогоднішній день існує безліч компаній, які пропонують свої послуги у різних сферах, тому виникає потреба впевнити клієнтів у тому, що саме ви пропонуєте послуги на найвищому рівні, так як саме це допомагає виділитись серед конкурентів.

У наш час швидкість, конфіденційність та зручність це ключові фактори, які визначають чи буде клієнт користуватись послугами вашої компанії, чи ні. Але надання якісних послуг, ще й швидко, потребує велику кількість кваліфікованого персоналу, що доволі важко отримати без додаткових витрат зі сторони компанії, тому використання програмного забезпечення, яке може задовольнити всі ці потреби належним чином, є досить актуально.

Загалом під час створення подібного програмного забезпечення приділяється велика кількість часу для розробки та адаптації інтерфейсу користувача. Саме інтерфейс в момент знайомства користувача с сервісом грає визначну роль у подальшому виборі на користь компанії, адже він формує загальні критерії швидкості та зручності програмного забезпечення в цілому.

Адаптивний та інтуїтивно зрозумілий інтерфейс застосунку дійсно може зацікавити велику кількість користувачів, проте в парі з технічними методами захисту конфіденційних даних, кількість активних клієнтів зростатиме.

Сервіси безпеки - це сукупність механізмів та заходів управління, що спрямовані на зменшення ризиків, пов’язаних з загрозою втрати або розкриття даних. Одні сервіси забезпечують захист від загроз, інші виявляють слабкі місця в системі безпеки. Відсутність базових сервісів безпеки у застосунках може нести за собою не тільки втрату особистих даних клієнта, а й викрадення конфіденційної інформації компанії, яку заборонено розголошувати третім особам.

Актуальність роботи зумовлена тим, що комунікація між користувачем та компанією відбувається більш оперативно за рахунок чого можна

обслуговувати більшу кількість клієнтів, що в свою чергу підвищить результативність роботи фінансової організації. Деяка робота зі статичними і динамічними даними стає автоматизованою, так як відправка звітів, розрахунок прибутковості не буде вимагати дій співробітника компанії. Користувач, коли йому зручно може запросити цю інформацію, при цьому всі джерела інформації залишаються конфіденційними що не порушує регламент роботи деяких компаній.

Відмінною рисою самого сервісу є те, що формування інвестиційного портфеля відбувається в режимі реального часу, при цьому актуальні дані автоматично оновлюються.

Метою роботи є розробка застосунку для автоматизації роботи в сфері фінансового менеджменту з використанням сучасних методів захисту, що дозволяє залишати всі джерела інформації конфіденційними.

Завданням дослідження є:

- Огляд основних технологій автоматизації та підтримки бізнесу;
- Огляд та вибір сервісів безпеки, технологій та технічних рішень;
- Реалізація застосунку з використанням обраних технологій та сервісів безпеки, перевірка ефективності.

Об’єктом дослідження є існуючі механізми безпеки сервісів.

Предметом дослідження є розробка застосунку з використанням сучасних методів захисту та аналіз отриманих результатів.

Наукова новизна полягає в обґрунтованому виборі та комбінації різних механізмів безпеки сервісів в одному застосунку, який в подальшому може пропонуватись як готове рішення.

Практичне значення полягає у тому, що отриманий застосунок у ході роботи може бути використаний у фінансових організаціях зі схожим направленням. Користувач буде потребувати мінімальну кількість часу на його обслуговування.

Методами дослідження є огляд літературних джерел за обраним направленням, вибір найбільш підходящих сервісів безпеки та технічних рішень, їх інтеграція та аналіз отриманих результатів.

Апробація результатів роботи буде проведена з використанням отриманих результатів у ході відкритого тестування програмного забезпечення.

1 СИСТЕМИ АВТОМАТИЗАЦІЇ РОБОЧИХ ПРОЦЕСІВ

1.1 Визначення робочого процесу

Виконання дрібних і повторюваних завдань не тільки забирає багато часу, а й може привести компанію до значних витрат. Але можна автоматизувати певні завдання за допомогою шаблонів, і тоді керівники та учасники команд зможуть приділяти більше уваги найбільш важливим проектам.

Шаблони дозволяють компаніям автоматизувати робочі процеси, оптимізувати операції і ефективно розширюватися відповідно до зростаючого попиту.

Робочий процес - це запланований, послідовний і повторюваний шаблон дій або операцій. Це кульмінація подій, дій, ресурсів і конкретних завдань, які в кінцевому підсумку будуть пов'язані між собою для досягнення бажаного результату [1]. Робочі процеси приймають безліч форм на робочому місці і можуть зайняти від одного дня до декількох місяців (Рисунок 1.1).

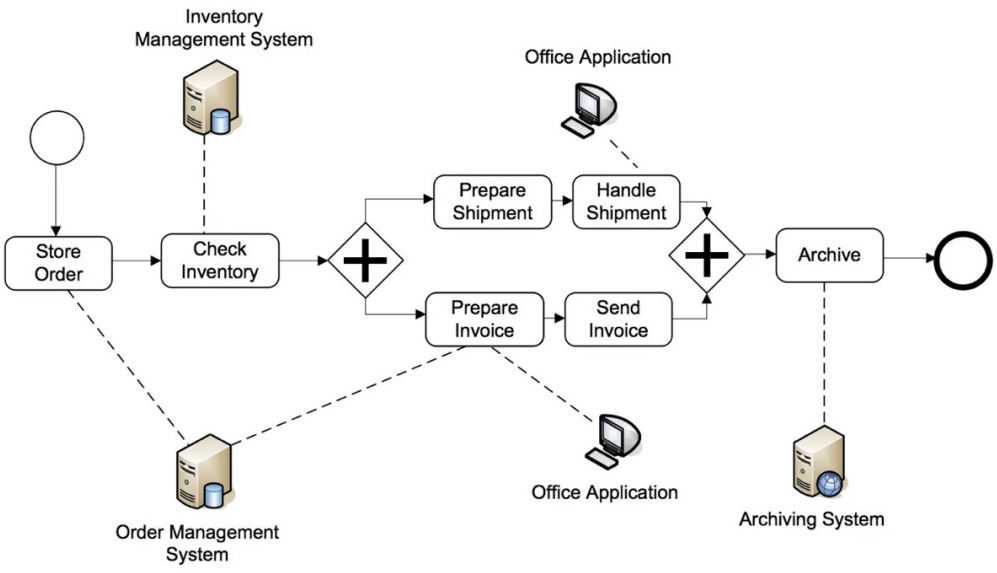

Рисунок 1.1 – Приклад схеми робочого процесу

Формування робочого процесу проходить в певній послідовності логічно пов’язаних прийомів, дій та рухів:

1. Аналіз ситуації (визначення проблем, технологій, плану процесу та результатів)
2. Розробка робочого процесу (реалізація подій та операцій за визначеним планом та технологіями)
3. Аналіз отриманих результатів (порівняння очікуваного та отриманого результату процесу)
4. Інтеграція процесу в роботу

Далі з розробленим робочим процесом знайомляться співробітники, які будуть відповідальні в подальшому за всі операції, що відбуваються у ньому.

В рамках автоматизованого робочого процесу зникає необхідність вручну передавати робочі матеріали від одного учасника до іншого. Процес автоматизований таким чином, що після завершення одного етапу, моментально запускається наступний етап.

1.2 Переваги автоматизованих робочих процесів

Автоматизація робочого процесу націлена на збільшення швидкості, точності та ефективності операцій. В ході автоматизації робочих процесів ви скорочуєте обсяги ручної роботи, яку доводиться робити співробітникам, звільняючи їх для більш важливих завдань.

Розглядаючи автоматизовані робочі процеси ми можемо виділити ряд переваг на відміну від неавтоматизованих робочих процесів [2]:

1. Впорядкування внутрішньої комунікації

Однією з найбільших переваг автоматизації робочого процесу є те, що вона покращує внутрішню комунікацію (Рисунок 1.2). Це зменшує темпи

плинності працівників, оскільки однією з найбільших причин, через яку працівники залишають організацію, є відсутність зв'язку з керівництвом. Автоматизуючи робочий процес, ви також автоматизуєте спілкування в компанії, що піде на користь загальної роботи компанії.

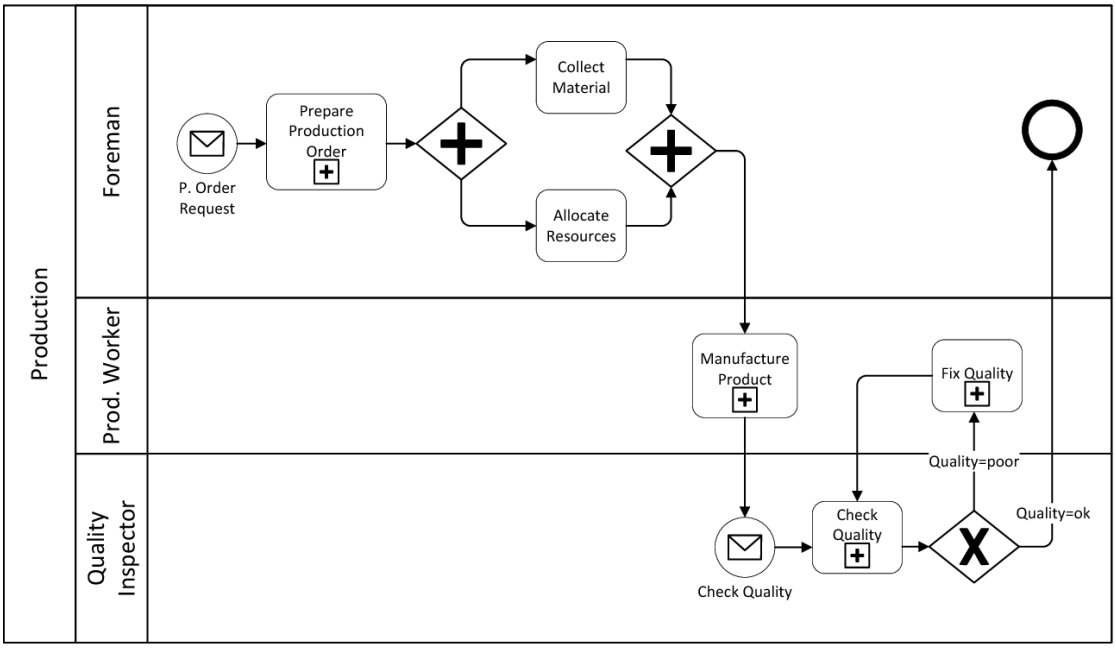

Рисунок 1.2 – Приклад автоматизованого процесу внутрішньої комунікації

2. Організація підзвітності

Автоматизуючи робочий процес, ви ефективно призначаєте співробітника, який відповідає за кожну частину процесу. Для кожного кроку в процесі є одна особа, яка виконує певну дію. Роблячи це, ви створюєте систему підзвітності, де кожен знає, за які конкретні завдання відповідає. Більш того, з даною системою підзвітності можливо зручно виявити, які завдання займають найбільше часу і де процес затримується частіше. Автоматизація робочих процесів дозволяє приймати подальші рішення для створення більш ефективних процесів і відповідно розподіляти роботу.

3. Зниження ймовірності помилок та додаткових витрат

Автоматизація робочих процесів зменшує кількість помилок, оскільки вони працюють автоматично, тим самим знижують ймовірність помилки через людський фактор. Ще однією з переваг автоматизації є те, що вона рятує компанії від додаткових витрат, пов'язаних з помилками працівників, а також може зменшити витрати на адміністративну працю.

4. Можливість працівників керувати власним часом

Оскільки працівник має можливість відстежувати одразу всі поточні задачі, тим самим зручно розподіляти свій робочий час, завдяки чому знижувати ризик прострочення дедлайну. Більше керівникам не доведеться слідкувати за роботою кожного працівника або витрачати час на перевірку їх прогресу.

5. Підвищення ефективності робочого процесу

Автоматизований робочий процес одразу несе в собі всі необхідні інструкції щодо виконання самого процесу та його очікуваного результату. У випадку, коли в одному робочому процесі задіяно більш ніж один співробітник, це дає можливість скоротити час передачі завдання з одного етапу процесу на інший. Всі операції з оформлення та передачі звітів сам процес делегує на себе, тим самим звільняє працівника від додаткових витрат часу на інформування колеги про готовність.

Застосовуючи автоматизацію робочого процесу, компанії можуть зменшити кількість ручних завдань, що виконуються працівниками. Це дозволяє виконати більше задач за коротший проміжок часу, що очевидно підвищить загальну продуктивність праці (Рисунок 1.3).

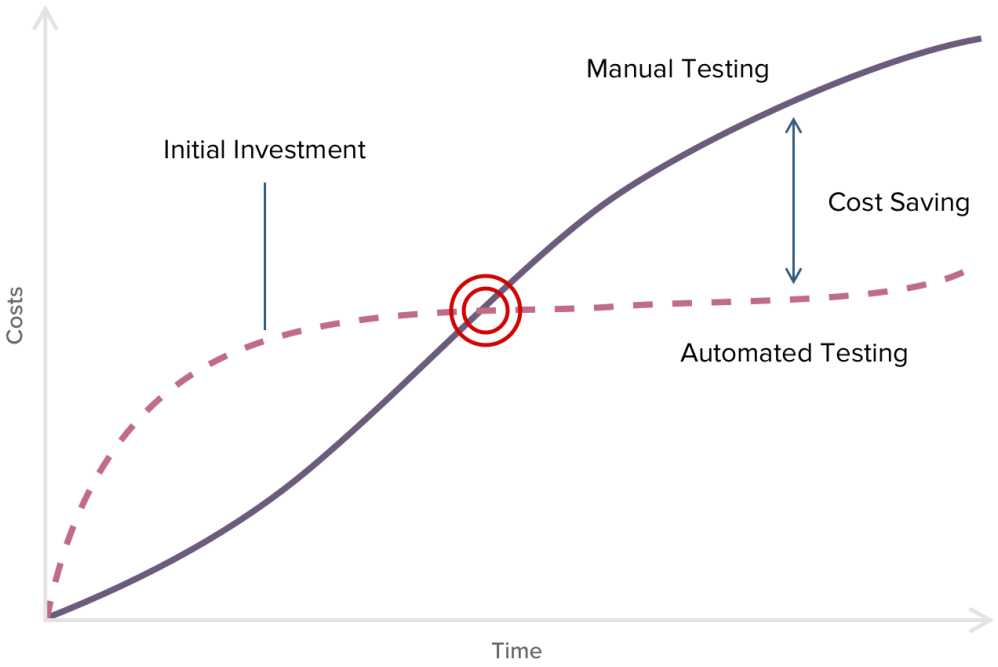

Рисунок 1.3 – Графік тестування автоматизованих процесів у порівнянні з ручними

1.3 Практичне застосування автоматизованих робочих процесів

Завдяки тому, що майже кожен робочий процес можна автоматизувати, практично застосувати це можна у будь-якій робочій сфері задля підвищення результативності роботи [3].

У сфері фінансів за допомогою автоматизації деяких робочих процесів можна різко зменшити об’єми роботи з паперовими документами, що одразу знизить кількість витраченого часу на очікування, обробку і схвалення розпоряджень тощо. Також можливо розширити спектр послуг, наприклад, надати можливість користувачу самостійно налаштовувати автоматичні платежі у застосунку, це заощадить час клієнта та звільнить від обробки додаткових запитів фінансову організацію. Більш того, можливо реалізувати розумну систему управління виплатами заробітної плати та інтегрувати її з

програмним забезпеченням для бухгалтерського обліку, в свою чергу це допоможе розгрузити бухгалтерський відділ та знизити ризик помилок при нарахуванні коштів.

Особлива увага розробці автоматизованих процесів приділяється у різних комерційних сферах (Рисунок 1.4). У маркетингу, зазвичай, вони використовуються для делегування таких важких задач як супроводження рекламних кампаній та фільтрування якісних потенційних клієнтів у реальному часі для подальшої передачі їх у відділ продажів, що дозволяє керувати та контролювати процеси за допомогою єдиної інформаційної панелі.

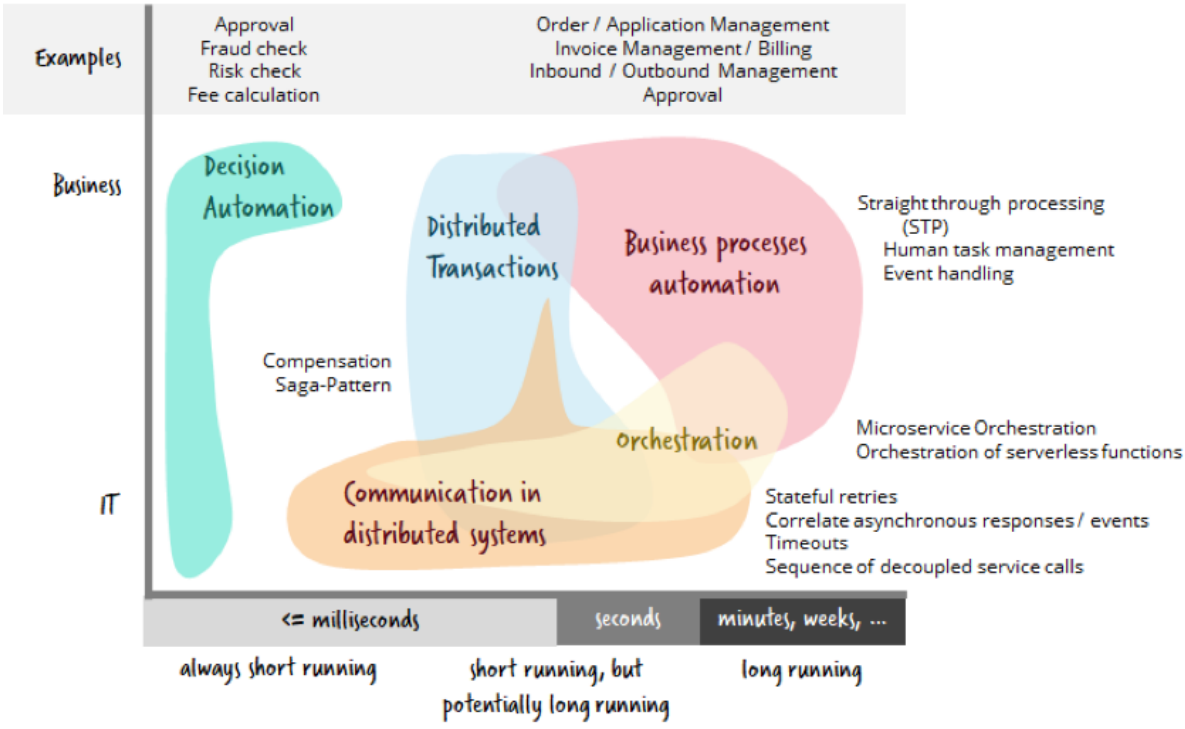

Рисунок 1.4 – Випадки використання автоматизованих робочих процесів

У відділах продажу ми можемо використовувати автоматизовані процеси для того, щоб працювати з більшою кількістю клієнтів, підвищуючи загальну результативність роботи відділу. Співробітник компанії може не витрачати свій робочий час на інформування, щодо нових пропозицій та продуктів від

компанії, а направити основний трафік потенційних та діючих клієнтів у загальний канал розсилок, у якому буде публікуватись ця інформація, як результат, співробітник відділу продажів зможе більше сконцентруватися на успішному завершенні угоди.

У сферах інформаційних технологій зазвичай автоматизовані процеси беруть на себе модерацію запитів до технічної підтримки, щоб уникнути їх дублікатів та не витрачати час на аналіз повторних задач. Також ці процеси займаються автоматичним оновленням програмного забезпечення на пристроях підприємства в неробочий час, що виключає факт оновлення пристрою під час роботи співробітника, як наслідок, втрати важливої інформації, яка редагувалась в даний момент.

В цілому, практичне застосування автоматизованих робочих процесів надає можливість:

- менше працювати з паперовими документами, пропонуючи вести всю відповідну звітність в електронному варіанті;
- уникати ризиків, спричинених людським фактором;
- повний моніторинг виконуваних завдань, що допомагає проаналізувати той чи інший процес на ефективність;

1.4 Порівняння систем для автоматизації робочих процесів

Правильно підібране програмне забезпечення (ПЗ) для автоматизації робочого процесу може допомогти вашій команді пришвидшити прогрес у виконанні завдань та проектів, а може навіть включати вбудовану аналітику, яка допоможе вам визначити ефективність.

Більшість інструментів автоматизації робочих процесів надають візуальний інтерфейс, за допомогою якого ви можете створювати робочі процеси без кодування або великої кількості технічних знань. Функція Drag-

and-Drop, електронні форми або блок-схеми роблять це програмне забезпечення зручним для маркетологів, продавців, медичних та промислових працівників та навіть для навчальних закладів. Також сервіси автоматизації процесів пропонують гнучкі настройки для малого бізнесу та готові шаблони схем для процесів.

Функція Drag-and-Drop являється функціональністю, за допомогою якої користувачі можуть виділити об’єкт або фрагмент тексту, а також перемістити його у потрібне місце та «відпустити» туди. Drag-and-Drop є частиною більшості графічних користувальницьких інтерфейсів та іншого програмного забезпечення [4].

Також перед порівнянням застосунків для автоматизації робочих процесів слід знати, що таке електронні форми. Електронні форми (англ. E-Form) надають користувальницький інтерфейс для даних та послуг, як правило, через інтерфейс на основі браузера. Електронні форми дозволяють користувачам взаємодіяти з корпоративними додатками та пов'язаними з ними внутрішніми системами. Веб-програми, рішення для електронного уряду та електронної комерції викликали попит на практичні веб-форми, які підтримують більш насичену та динамічну взаємодію, ніж це можливо з формами HTML. Нові додатки електронної форми включають ідентифікацію вмісту XML, декілька виносок даних, валідацію полів та вбудовану логіку процесу, що містяться в захищеному форматі, який часто є портативним [5].

В таблиці 1.1 наведені найпопулярніші системи для автоматизації робочих процесів та їх порівняння [6]. Перераховані системи у таблиці порівнюються за наявністю такого базового функціоналу як: функція Drag-and-Drop, адаптивна мобільна версія та електронні форми. Проте, кожна із систем має свої особливості користувальницького інтерфейсу та функціоналу.

Таблиця 1.1 – Порівняння найкращих застосунків для автоматизації робочих процесів

Назва сервісу	Функція Drag-and-Drop	Мобільна версія	Електронні форми
ProcessMaker	+	+	+
Integrify	+	+	+
Comindware Tracker	+	+	+
Flokzu	+	+	+
Zapier	–	–	–
Kissflow	+	+	+
Nintex	+	+	+
Process Director	–	–	+
TrackVia	+	+	+
Gravity Flow	+	+	–

1.4.1 ProcessMaker

ProccessMaker - це інструмент автоматизації робочих процесів з відкритим кодом, відомий своєю простотою використання, так як потребує мінімальні знання написання коду. Візуальні блок-схеми допомагають створювати робочі процеси, а сповіщення вбудовуються в кожен із них. Вся система заснована на базі веб-застосунку та використовує функції WYSIWYG, що розшифровується як «що-ти-бачиш-є-те-що-ти-отримуєш» (англ. what-you-see-is-what-you-get), щоб зменшити бар'єр входу для користувачів будь-якої галузі, будь то виробництво, освіта, охорона здоров'я або телекомунікації.

Доступ до API для розробників відкриває програмне забезпечення для налаштування для великих організацій та більш складних процесів.

1.4.2 Integrify

Integrify використовує підхід, який базується на послугах, що включають підтримку та консультацію щодо найкращих практик та вдосконалення процесів. Програмне забезпечення розроблено для зручності використання та пропонує редактор Drag-and-Drop для більшості робочих процесів. Усі інструменти розроблені на базі веб-застосунку та доступні для будь-якого мобільного пристрою. Вірний своїй сервісно-орієнтованій моделі, веб-сервіс містить корисні приклади робочого процесу та повну базу знань для користувачів. Можливі корпоративні оновлення та розширення функціоналу для підприємств, що мають конкретні або складні потреби.

1.4.3 Comindware Tracker

Більшість представників електронної комерції, малого бізнесу та навчальних закладів вважають це програмне забезпечення ідеальним рішенням, так як воно легко масштабується для корпоративних організацій та потребує мінімальне використання коду. Застосунок розроблений для гнучких задач та пропонує створення автоматизованих процесів, заснованих на завданнях, які недостатньо добре обслуговуються більш структурованими системами. Робочі процеси легко переносяться з комп’ютера на мобільний, а конструктор Drag-and-Drop сумісний з іншими веб-сервісами, такими як Outlook, Excel тощо. Comindware Tracker дозволяє працювати та зберігати інформацію як локально, так і в хмарі, що робить це програмне забезпечення безпечним вибором для високорегульованих медичних або фінансових організацій.

1.4.4 Flokzu

Візуальні робочі процеси Flokzu містять графіки на основі значків, де ви можете спроектувати свої бізнес-процеси без знань програмування. Робочі процеси адаптуються індивідуально на основі конкретних завдань та потреб команди. Flokzu надає шаблони робочих процесів, що може прискорити їх розробку. Ця хмарна платформа обіцяє інформаційну безпеку завдяки наскрізному шифруванню, а повний набір статистичних даних процесів полегшує аналіз та вдосконалення процесів для вашої команди.

1.4.5 Zapier

Zapier є чудовим вибором для сфери маркетингу та управління проектами завдяки адаптивному інтерфейсу та простому підключенню сторонніх застосунків. Нещодавно платформа розширила свій функціонал, який дозволяє створювати цілі робочі процеси, використовуючи понад 500 інтеграцій - від сервісів маркетингу електронної пошти та обміну документами до соціальних мереж та управління проектами.

1.4.6 Kissflow

Простота Kissflow робить його одним з найпопулярніших інструментів автоматизації робочих процесів на ринку. Редагування за допомогою Drag-and-Drop та інтуїтивна візуалізація робочого процесу роблять програмне забезпечення простим у засвоєнні. Інтеграції API, WebHooks пов’язують всі ваші технології в одному місці та допомагають відстежувати процеси за допомогою сповіщень. За допомогою правильних налаштувань ви можете встановити дозволи для різних рівнів користувача та складати докладні звіти на основі показників, які мають значення для вашої команди.

1.4.7 Nintex

Nintex - це платформа з мінімальним використанням коду зі сторони користувача, що пропонує інтуїтивно зрозумілий конструктор Drag-and-Drop, який полегшує візуалізацію робочих процесів під час їх створення. Платформа дозволяє легко створювати та підписувати документи прямо в застосунку, а потім зберігати їх у своїх робочих процесах для зручного доступу. Компанії можуть застосовувати Nintex у хмарі, локально або в гібридному середовищі.

1.4.8 Process Director

Process Director поєднує в собі обмін документами, сповіщеннями, візуальними робочими процесами та звітами. Це перспективне рішення має дизайн, подібний до проекту Microsoft, і не вимагає блок-схем або кодування. Форми та серверні процеси, спрямовані на клієнта, інтегруються з аналітикою та звітуваннями.

1.4.9 TrackVia

Хмарна візуалізація робочого процесу TrackVia полегшує автоматизацію використання додатків, які ви вже вбудували у ваші виробничі та складські процеси. Програмне забезпечення створює зв’язки між існуючими закритими програмами, дозволяючи їм спілкуватися між собою та надсилати повідомлення відповідному співробітнику. Наскрізне шифрування даних означає, що конфіденційна інформація безпечна на хмарних серверах TrackVia, при цьому доступ до API доступний розробникам.

1.4.10 Gravity Flow

Засоби автоматизації Gravity Flow інтегрують клієнтські та внутрішні робочі процеси безпосередньо на сайти, які працюють на WordPress. Хоча дане програмне забезпечення являється плагіном, воно має в собі більшу частину із вищеперерахованого функціоналу. У якості недоліку можна виділити відсутність електронних форм в даному плагіні, натомість розробники додали можливість генерації PDF та блок-схем у реальному часі.

Порівнявши функціонал вищеперерахованих систем, також важливою характеристикою є захист особистих даних користувачів та компаній, які користуються цими застосунками. На сьогоднішній день обрані застосунки вважаються найкращим рішенням на ринку, відповідно в них використовуються найсучасніші методи захисту, шифрування, системи виявлення вторгнень та інші технології безпеки, які змогли вже зарекомендувати себе і користувачі з впевненістю можуть довіряти свої особисті дані.

Якщо говорити про шифрування особистих даних і комунікацій клієнтів, то майже усі програмні забезпечення у цьому направленні використовують шифрування AES-256 (Рисунок 1.5).

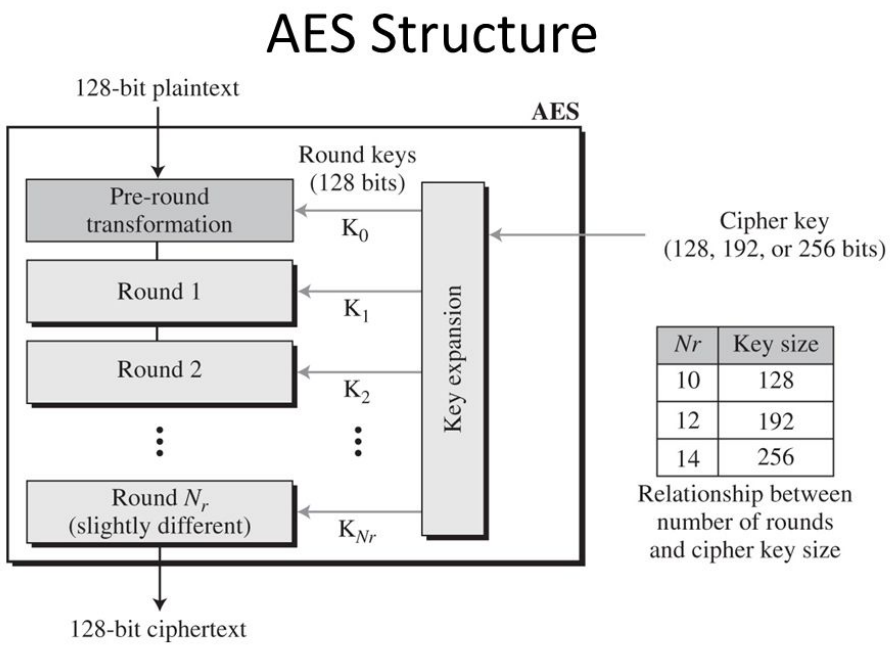

Рисунок 1.5 – Структура схеми розширеного стандарту шифрування

Розширений стандарт шифрування (AES) - це перший і єдиний загальнодоступний шифр, затверджений Агентством національної безпеки США (NSA) для захисту надсекретної інформації. AES вперше отримав назву Rijndael на честь двох розробників - бельгійських криптографів Вінсента Ріймена та Джоан Дамен. AES використовує симетричне шифрування ключів, яке передбачає використання лише одного секретного ключа для шифрування та розшифровки інформації [7].

Під час автентифікації користувачі отримують доступ до свого облікового запису лише з коректним поєднанням імені користувача та пароля, яке зашифровується через TLS під час передачі. Користувачам заборонено вибирати слабкі або очевидні паролі. При цьому зашифрований файл cookie ідентифікатора сесії використовується для однозначної ідентифікації кожного користувача. Для додаткової безпеки ключ сеансу автоматично регулярно шифрується та відновлюється у фоновому режимі.

Слід не забувати про безпеку операційної та внутрішніх систем. Всередині периферійних брандмауерів системи захищені перекладом мережевих адрес, перенаправленням портів, маскуванням IP, схемами нерегульованої IP-адреси тощо. Також забезпечується жорсткий захист на рівні операційної системи, використовуючи мінімальну кількість точок доступу до всіх виробничих серверів. Облікові записи операційної системи захищаються надійними паролями та двофакторною автентифікацією. Безпека операційних систем підвищується шляхом відключення та/або видалення непотрібних користувачів, протоколів та процесів.

У всіх багатокористувацьких застосунках мають бути добре налагоджені механізми резервного копіювання, системи для автоматизації робочих процесів не є винятком. Усі дані клієнта зберігаються у зашифрованих томах. Дані бази даних про замовника автоматично резервуються до останньої здійсненої транзакції [8].

Висновок до розділу 1

В даному розділі проведено детальний огляд систем автоматизації робочих процесів, а саме визначено поняття робочих процесів та методу їх формування. Також проаналізовано переваги автоматизованих процесів у порівнянні з ручними.

В розділі розглянуто практичне застосування автоматизованих робочих процесів у різних сферах діяльності, таких як: фінансових організацій, електронних комерціях, відділах маркетингу, інформаційних технологій та продажів задля підвищення їх результативності.

Також порівняно найкращі на сьогоднішній день системи автоматизації робочих процесів, виділено їх функціональні аспекти та оглянуто використані в них технічні методи захисту такі як: шифрування AES-256, автентифікація з шифруванням TLS, шифрування сесії користувача у файлах cookie, захист внутрішніх та операційних систем, а також резервне копіювання даних користувачів.

2 ОГЛЯД МЕХАНІЗМІВ ТА СЕРВІСІВ БЕЗПЕКИ

Механізми безпеки - це технічні засоби та прийоми, які використовуються для реалізації служб безпеки. Механізм може працювати як сам, так і разом з іншими, для надання певної послуги. Прикладами загальних механізмів безпеки є такі [9]:

- Криптографія
- Цифрові сертифікати
- Інфраструктура відкритих ключів (PKI)
- Ідентифікація та автентифікація
- Авторизація
- Аудит

Сервіс безпеки - це сервіс, що надається рівнем комунікацій відкритих систем, що забезпечує належну безпеку застосунків та програмних забезпечень або передачі даних, як визначено Рекомендацією ITU-T X.800 [10].

2.1 Google Drive в якості сервісу хмарного сховища

Хмарне зберігання - це сервісна модель, при якій дані передаються та зберігаються у віддалених системах зберігання, де вони підтримуються, керуються, створюються резервні копії та робляться доступними для користувачів через мережу - як правило, Інтернет [11].

На сьогоднішній день існує досить багато різних сервісів збереження статичних і динамічних файлів, які надають свої послуги мільйонам користувачів, але кожен з них має свої особливості, які визначають сферу їх

використання. У ході розробки програмного забезпечення стояв вибір між Google Drive та Amazon Web Services (AWS3).

Позитивне рішення було прийняте на користь Google Drive, так як саме він пропонує більш захищені протоколи передачі даних ніж AWS3. Сервіс від Google шифрує файли AES-128 (розмір блоку 128 біт) під час передачі та 256-бітною у режимі спокою, з додатковою двофакторною автентифікацією, біометричні сканери забезпечують додатковий захист усіх файлів. Amazon в свою чергу пропонує багатофакторну автентифікацію. Інші переваги Google Drive перед AWS3 на період розробки сервісу зображені на рисунку 2.1 [12, 13].

Також Google пропонує зручний контроль доступу до файлів, що допоможе знайти співробітника з поганими намірами при спробі передачі інформації третім особам.

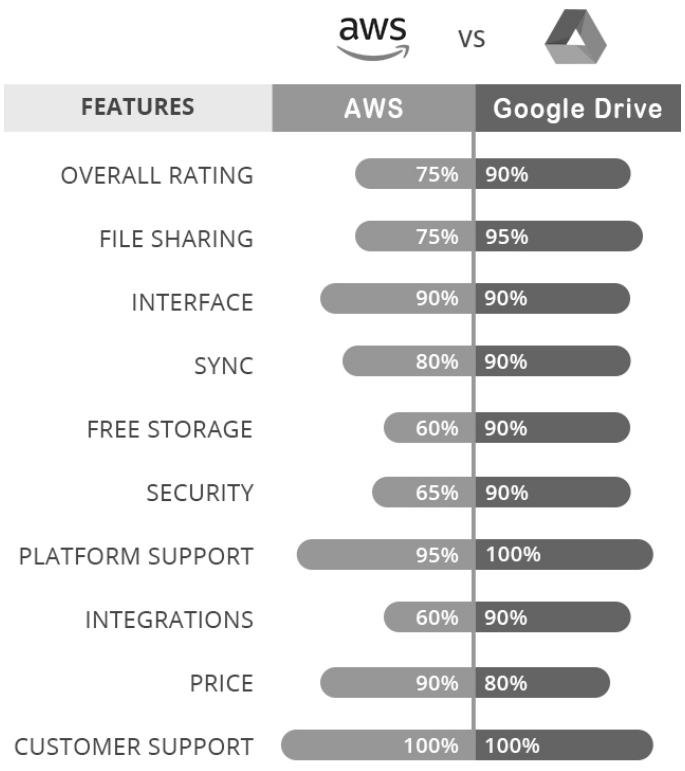

Рисунок 2.1 – Порівняння Google Drive та Amazon Web Services

2.2 Реляційна база даних MySQL, як сервіс безпеки

Під час вибору бази даних було обрано MySQL, так як вона має явні переваги перед SQLite та PostgreSQL. Для того, щоб впевнитись у цьому потрібно визначити поняття реляційних баз даних та виділити ключові моменти кожної з них [14].

Реляційна база даних - це тип бази даних, що зберігає та забезпечує доступ до даних, пов’язаних між собою. Такі бази даних базуються на реляційній моделі, інтуїтивно зрозумілому, прямому способі представлення даних у таблицях. У реляційній базі даних кожен рядок таблиці є записом з унікальним ідентифікатором, який називається ключем. Стовпці таблиці містять атрибути даних, і кожен запис зазвичай має значення для кожного атрибута, що полегшує встановлення взаємозв’язків між цими даними [15].

2.2.1 SQLite

SQLite доволі портативна безсерверна база даних, яка ідеально підходить для мікросервісів, або для тестування на період розробки. Головний її недолік це те, що в ній немає керування користувачами, єдиними чинними дозволами на доступ є типові права користувача в операційній системі. Це робить SQLite невдалим вибором для сервісів, яким потрібно кілька користувачів зі спеціальними дозволами на доступ, відповідно в даному випадку неможливо налаштувати додатково політику безпеки у базі даних, що робить сервіс менш захищеним.

Обмежений паралелізм, що пропонує SQLite також робить її не підходящою для даного сервісу. Хоча декілька процесів можуть отримати доступ до бази даних і запросити її одночасно, тільки один процес може вносити зміни в базу даних у будь-який момент часу. Це означає, що SQLite підтримує більше паралелізму, ніж більшість інших вбудованих систем

управління базами даних, але не так, як клієнт/серверні РСУБД, такі як MySQL або PostgreSQL.

2.2.2 PostgreSQL

PostgeSQL позиціонує себе як "сама передова реляційна база даних в світі з відкритим вихідним кодом". Вона має велику кількість переваг перед MySQL та SQLite, але якщо говорити про продуктивність, то для кожного нового клієнтського підключення PostgreSQL ініціалізується новий процес. Кожному новому процесу виділяється близько 10 МБ пам'яті, тому база даних буде більш вимоглива до пам'яті.

Відповідно, для простих операцій з великим об'ємом читання PostgreSQL, як правило, менш продуктивна, ніж інші РСУБД, такі як MySQL.

2.2.3 MySQL

MySQL вважається найпопулярнішою РСУБД (Рисунок 2.2). MySQL був розроблений для швидкості і надійності за рахунок повного дотримання стандартного SQL. На відміну від сервісів, що використовують SQLite, сервіси, що використовують базу даних MySQL, отримують доступ до нього через окремий процес. Оскільки серверний процес знаходиться між базою даних і застосунком, це дозволяє більше контролювати тих, хто має доступ до бази даних. При цьому вона не є такою ресурсозатратною, як PostgreSQL.

Отже, база даних MySQL більш всього підходить для розроблюваного сервісу, як з міркувань безпеки, так і з міркувань продуктивності сервісу в цілому.

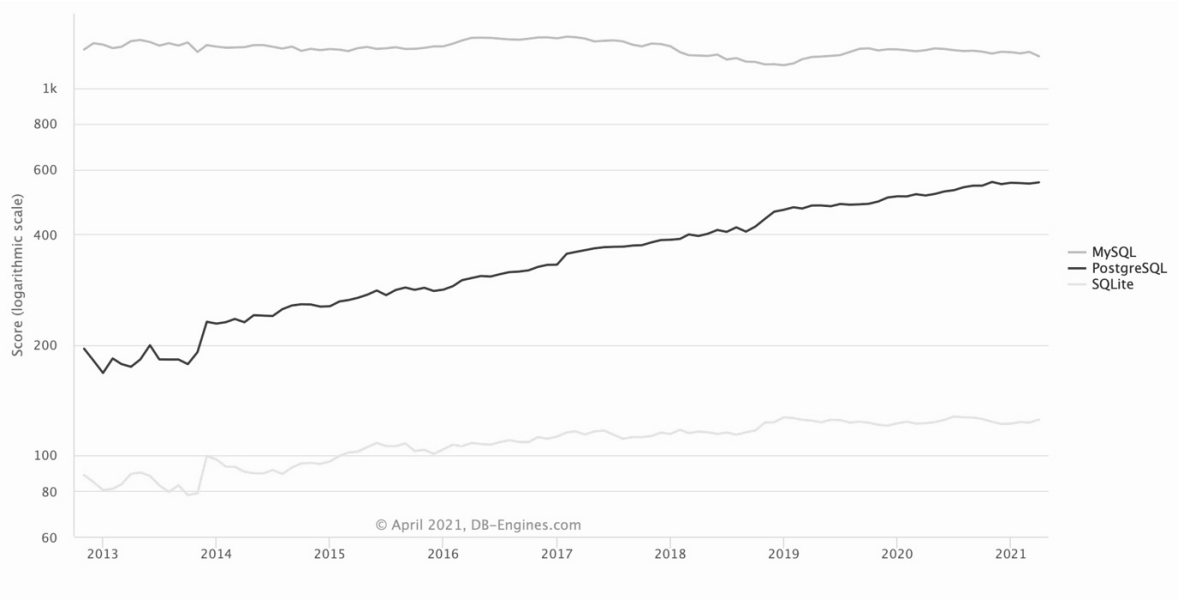

Рисунок 2.2 – Тенденція популярності MySQL, PostgreSQL та SQLite

2.3 Корпоративна платформа, як додатковий сервіс безпеки

В умовах фінансової організації, де тестуватиметься даний сервіс, надана корпоративна платформа, де зберігаються особисті дані діючих клієнтів, які підтвердили свою особистість за допомогою документів та пройшли моніторинг по всім банкам країни та по базам терористів. Це значно спростить механізми автентифікації користувачів в сервісі. Відповідно надання доступу до сервісу буде працювати за принципом: якщо користувач верифікований на корпоративній платформі, тоді доступ до сервісу дозволено.

В майбутньому, у випадку публічного розширення сервісу, поточний механізм автентифікації можна замінити перевіркою та моніторингом напряму по базам банків та терористів.

2.4 Telegram Bot у якості клієнтської частини сервісу

В якості клієнтської частини обрано один із найпопулярніших месенджерів – Telegram, який надає публічну можливість створення ботів, у якому доволі зручно створити інтерфейс для будь-яких потреб.

Для того, щоб користуватись месенджером та розробленим сервісом потрібно авторизуватись через особистий номер телефону, що в комбінації з корпоративною платформою дає додаткову впевненість у тому, що тільки сам користувач можете отримати доступ до свого облікового запису в сервісі.

В якості клієнтської частини могли бути обрані і інші месенджери, такі як WhatsApp, Viber і т.д., також є можливість розробити додаток для мобільних пристроїв, або навіть повноцінний сайт. Перевагою використовувати Telegram є те, що месенджер пропонує захист на основі протоколу MTProto, який побудований на перевірених часом алгоритмах, щоб зробити безпеку сумісною з високошвидкісною доставкою та надійністю при слабких з’єднаннях [16].

2.4.1 Протокол MTProto

Схема принципу роботи протоколу MTProto наведена на рисунку 2.3. Перш ніж повідомлення передається через мережу за допомогою транспортного протоколу, воно зашифровується певним чином, а у верхній частині повідомлення додається зовнішній заголовок, який складається з 64-розрядного ідентифікатора ключа auth_key_id (що однозначно ідентифікує ключ авторизації для сервера, а також користувача) та 128-бітний ключ повідомлення msg_key.

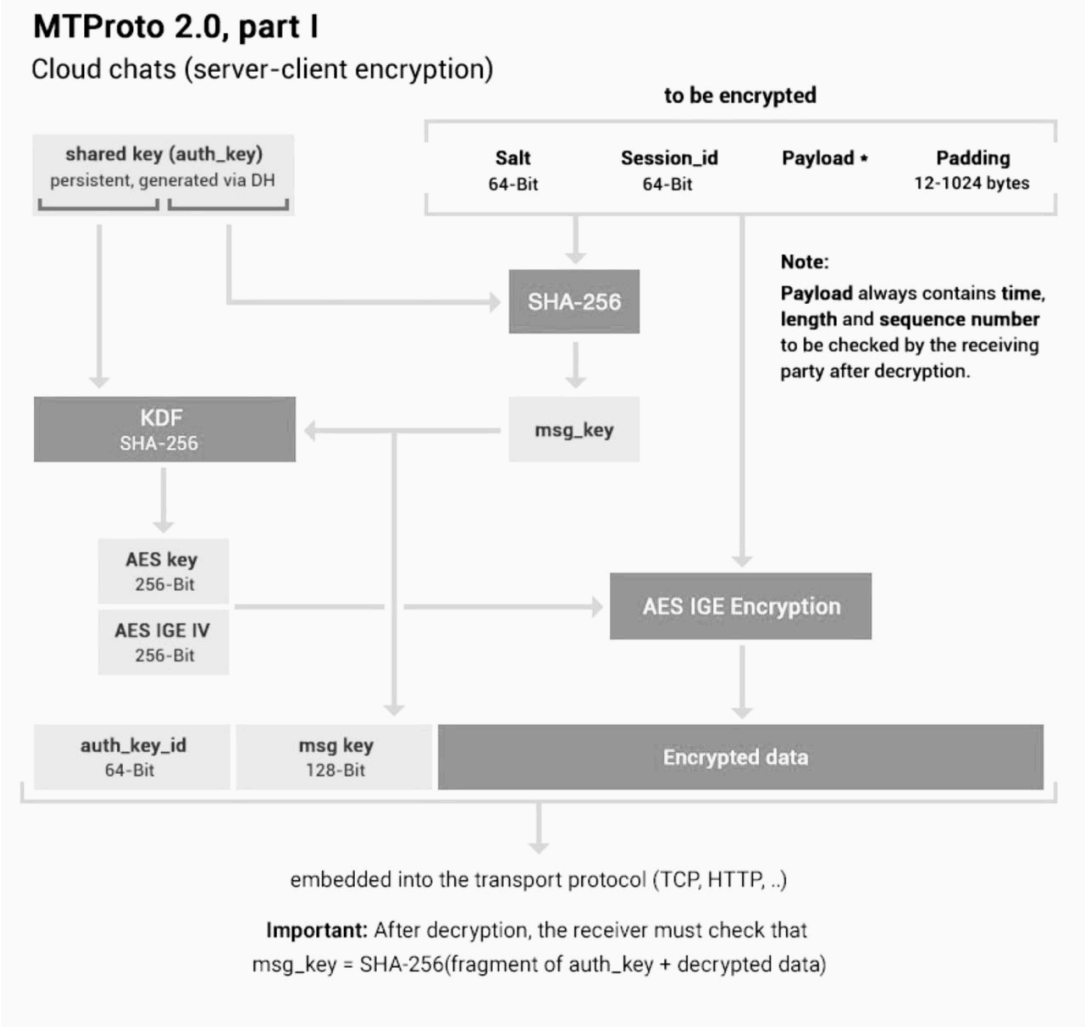

Рисунок 2.3 – Схема протоколу MTProto 2.0

Ключ авторизації auth_key в поєднанні з ключем повідомлення msg_key визначає фактичний 256-бітний ключ aes_key та 256-бітний вектор ініціалізації aes_iv, які використовуються для шифрування повідомлення за допомогою шифрування AES-256 з розширенням невизначеного спотворення (IGE) (Рисунок 2.4). Слід звернути увагу, що початкова частина повідомлення, яка підлягає зашифруванню, містить змінні дані (сеанс, ідентифікатор повідомлення, порядковий номер, salt сервера), що очевидно впливає на ключ повідомлення (і, отже, ключ AES та iv). У MTProto 2.0 ключ повідомлення визначається як 128 середніх бітів SHA-256 тіла повідомлення (включаючи

сеанс, ідентифікатор повідомлення, відступ тощо), додані 32 байтами, взятими з ключа авторизації [17].

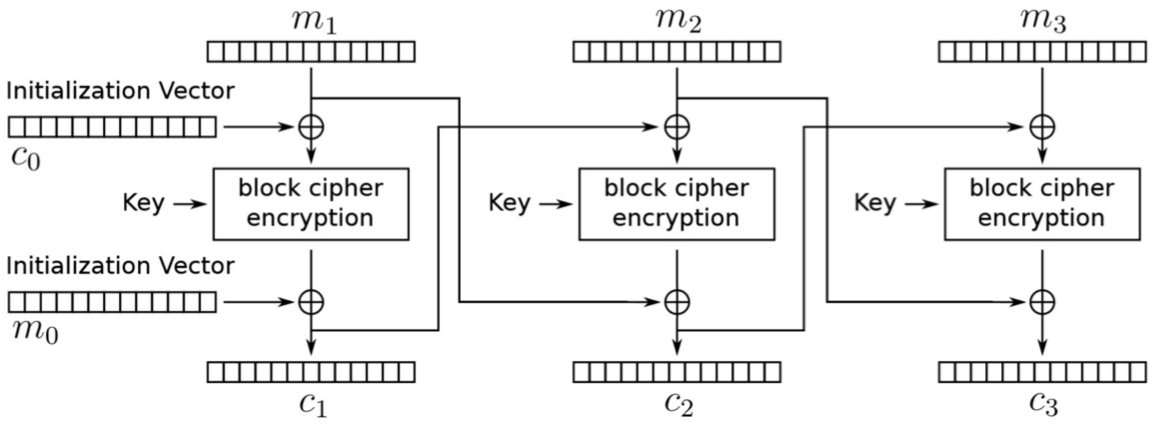

Рисунок 2.4 – AES-256 шифрування з розширенням невизначеного спотворення

2.5 Визначення та огляд політики безпеки

Політика безпеки - це набір правил, політик та процедур, призначених для забезпечення того, щоб усі користувачі та мережі в організації відповідали мінімальним вимогам щодо безпеки та захисту даних [18].

Створення ефективної політики безпеки є важливим кроком для запобігання порушень безпеки. Щоб зробити політику безпеки по-справжньому ефективною, слід розробити її у відповідності з регламентом роботи компанії та оновляти відповідно до змін цього регламенту. Також потрібно постійно покращувати її з появленням нових загроз та проводити аналіз отриманої інформації з логу застосунку в результаті попередніх порушень [19].

Для того, щоб зробити політику безпеки практичною та придатною для застосування, потрібно опрацювати та встановити систему виключень, яка враховує більшість вимог.

У розроблюваному застосунку буде визначено три ролі: Користувач, Адміністратор та Менеджер.

Користувач має найнижчий рівень доступу, в його можливості входить лише користуватись сервісом в межах налаштувань програмного забезпечення. У користувача існує 2 рівні доступу:

- Перший рівень має обмежений функціонал 2-го рівня. У можливості користувача першого рівня входить отримання актуальних інвестиційних ідей на 3 дні та подача заявки на отримання повного доступу в сервісі.
- Другий рівень має повний функціонал користувача, а саме: отримання актуальних інвестиційних ідей на 2 тижні, фільтрація інвестиційних ідей по секторам, формування інвестиційного портфелю в реальному часі, перегляд профілю, навчальні матеріали, актуальні новини та можливість зв'язатись з персональним менеджером.

Адміністратор має найвищий рівень доступу, йому доступний функціонал користувача 2-го рівня, а також: управління користувачами (додати користувача, додати менеджера, заблокувати/розблокувати, змінити менеджера користувачу), отримати інформацію по користувачу та відправити повідомлення всім користувачам сервісу.

Також у адміністратора є доступ повністю до всіх даних сервісу, які зберігаються в хмарі. Завдяки можливостям Google Drive, при будь-якій зміні даних записується хто змінив, що змінив і зберігається резервна копія документа в разі, якщо потрібно повернутись на минулу версію. При будь-якій

зміні прав і статусів користувачів відбувається повідомлення суб'єкта та адміністраторів про зміни.

Менеджер в свою чергу має обмежений функціонал адміністратора, він може надати повний доступ до сервісу, заблокувати/розблокувати користувача та отримати інформацію по ньому. Також менеджер має можливості користувача 2-го рівня.

На прикладі даної політики безпеки вдалося отримати вдалу комбінацію ролей та прав, що зробило модерацію програмного забезпечення доволі зручною і безпечною, усі зміни у базі знань та у правах користувачів фіксуються, що автоматично підвищує рівень безпеки.

2.5.1 Технічні рішення для політики безпеки

Для того, щоб у разі несанкціонованого доступу до коду програми не були вилучені дані доступу до бази знань, дані для авторизації у базу даних та в цілому до її серверної частини є сенс використовувати системні змінні (змінні оточення), в яких зберігається ця інформація. Більш того, дані для доступу до хмари та до клієнтської частини сервісу зберігаються в вигляді згенерованих токенів, що підвищує рівень безпеки при передачі, або отриманні інформації.

Висновок до розділу 2

В даному розділі було визначено та розглянуто механізми та сервіси безпеки. Також було визначено поняття хмарного сховища та порівняно Google Drive та Amazon Web Services на основі їх протоколів передачі даних.

В розділі розглянуто поняття реляційних баз даних та порівняно найпопулярніші із них: MySQL, PostgreSQL та SQLite. Проаналізувавши дану інформацію було обрано бази даних MySQL для програмної реалізації.

Також визначено поняття політики безпеки та запропоновано перелік правил для кожної ролі, які використовуватимуться в ході розробки застосунку.

В якості клієнтської частини програмного забезпечення було розглянуто месенджер Telegram, так як саме він пропонує функціонал створення ботів зі зручним в налаштуванні користувацьким інтерфейсом та сучасний захист особистих даних при передачі за допомогою протоколу MTProto 2.0, в основі якого лежить шифрування AES-256 з розширенням невизначеного спотворення.

3 РЕАЛІЗАЦІЯ ЗАСТОСУНКУ З ВИКОРИСТАННЯМ ОБРАНИХ ТЕХНОЛОГІЙ

Для реалізації застосунку було використано технології та сервіси безпеки, які розглянуті у другому розділі. У якості локального середовища для розробки використовувались операційні системи MacOS версії 11.3.1 (Big Sur) та Raspios версії 5.10 для Raspberry Pi в якості системи для продакшену.

3.1 Налаштування сервісу бази даних MySQL

Як було показано у попередньому розділі, вибір було зроблено на користь баз даних MySQL. Для її встановлення у середовище розробки спершу було встановлено менеджер пакетів Homebrew з публічного репозиторію GitHub розробників за допомогою команди:

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.
com/Homebrew/install/master/install)"
```

Встановлення MySQL було виконано за допомогою команди:

```
brew install mysql
```

Після встановлення баз даних було запущено сервіси MySQL та виконано базові налаштування за допомогою mysql_secure_installation.

Для створення відповідних таблиць для подальшої роботи застосунку було виконано SQL запит для створення таблиці користувачів:

```
CREATE TABLE users (id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, Last_Name VARCHAR(40), First_Name VARCHAR(40), Telegram_ID INT, Manager_ID INT, Phone_Number BIGINT, Role INT, Status INT, Subscriptions VARCHAR(1000));
```

А також для таблиці менеджерів:

```
CREATE TABLE managers (id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, Last_Name VARCHAR(40), First_Name VARCHAR(40), Telegram_ID INT, Phone_Number BIGINT, Telegram_Name VARCHAR(100), Email VARCHAR(100), Status INT);
```

3.2 Налаштування сервісу хмарного сховища Google Drive

Було прийнято рішення використовувати сервіс Google Drive в якості хмарного сховища для застосунку. Для налаштування сервісів сховища використано публічний API.

Першим кроком є отримання ключів доступу. Для цього потрібно мати діючий обліковий запис у системі Google та перейти у консоль управління хмарним сховищем. У пункті меню API та Сервіси натиснувши на кнопку «Створити ключі доступу» та вибравши у відповідному меню «Сервісний обліковий запис» (Рисунок 3.1) було введено Ім’я та ID сервісного облікового запису. Після виконаних дій ми отримали файл формату JSON, у якому знаходиться вся необхідна інформація для доступу до нашого хмарного сховища у подальшій розробці та роботі застосунку (Рисунок 3.2).

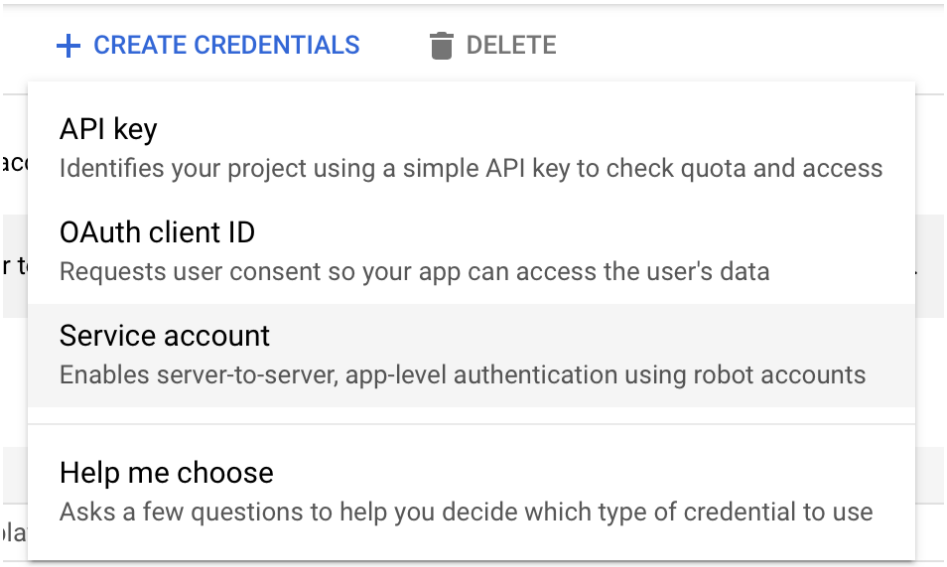

Рисунок 3.1 – Створення сервісного облікового запису

Рисунок 3.2 – Приклад отриманого файлу JSON

3.3 Програмна реалізація застосунку

Після налаштованих сервісів баз даних та хмарного сховища були реалізовані методи комунікації застосунку та цих сервісів на мові Python. У цих методах було описано всі основні запити до сервісів, такі як: перевірка існування облікового запису користувача, отримання статусу та ролі користувача, блокування, підпис на інвестиційні ідеї, автоматичне оновлення даних з хмари тощо [Додаток А, 1].

Так як, клієнтська частина сервісу базується на звичайному WEB-сервері, тобто він очікує до себе запит, у якому знаходиться ідентифікатор користувача, після чого визначає, яку відповідь і кому він повинен дати. Для коректної роботи цього механізму було описано всі дії сервера від моменту отримання запиту від користувача до відповіді сервера на запит. При цьому наш застосунок, у ході обробки запиту користувача, може самостійно запросити додаткову інформацію для успішної його обробки, до прикладу, коли клієнт бажає залишити заявку на отримання повного доступу сервер просить надати номер телефону та місто, у якому він проживає. Якщо введені дані користувачем введені коректно (у застосунку розроблені та інтегровані методи валідації отриманих даних), то заявка успішно передається для подальшої обробки [Додаток А, 2].

Більш того, сам застосунок в ході обробки певних запитів від користувача може самостійно робити запити до інших сервісів, будь то сервіс безпеки чи хмарне сховище. Для цього були реалізовані методи міжсервісної комунікації [Додаток А, 3].

Завдяки тому, що у ході розробки програмного забезпечення використовується бібліотека aiogram, яка базується на асинхронній бібліотеці asyncio у сервісі було реалізовано ряд методів, які циклічно працюють у фоновому режимі. Це дозволило виконувати ряд автоматизованих процесів, які не примушують очікувати користувача повного їх завершення та дає змогу користуватись сервісом та оновлювати дані у реальному часі [Додаток А, 4]. До таких методів відносяться автоматичне публікування інвестиційних новин, щотижневі розсилки, оновлення інформації по інвестиційним ідеям тощо.

3.4 Аналіз отриманих результатів у ході публічного тестування

В ході тестування програмного забезпечення було залучено більше ніж 170 активних користувачів та 5 менеджерів з якими вдалось знайти деякі вразливості, а також проаналізувати результативність сервісу в цілому. На кожного менеджера припадало в середньому по 34 клієнти, які тестували сервіс, при цьому менеджери займались обслуговуванням клієнтів які не приймали участь у тестуванні. Сумарно за 8 днів тестування нашим програмним забезпеченням було оброблено 20197 запитів від користувачів, з них 7058 запитів було оброблено нашим сервісом.

Провівши аналіз даних з таблиці 3.1 ми розуміємо, що за період тестування наше програмне забезпечення обробило 34,95% від загальної кількості запитів, що дає можливість обслуговувати значно більшу кількість клієнтів, відповідно підвищувати результативність в цілому.

Таблиця 3.1 – Результати роботи сервісу за 8 днів

Всього запитів	20197
Оброблено сервісом	7058
Запитів за 1 день оброблено сервісом	882
Запитів за день на 1-го менеджера оброблених сервісом	176

Висновки до розділу 3

Даний розділ присвячений програмній реалізації застосунку для автоматизації роботи в сфері фінансового менеджменту та аналізу результатів продуктивності застосунку.

Запропонований застосунок надає найкращі у цьому випадку методи захисту, що будуть забезпечувати якомога більшу конфіденційність та захищеність в ході користування сервісом, при цьому піднімає ефективність роботи фінансової організації на 34,95% в рамках тесту на одному з підрозділів. У випадку інтеграції даного застосунку у всіх підрозділах організації можливо досягти досить великий приріст ефективності компанії в цілому не ризикуючи втратою конфіденційних даних організації та користувачів застосунку.

ВИСНОВКИ

У роботі було оглянуто системи автоматизації робочих процесів, практичне їх застосування та порівняно найкращі системи на сьогоднішній день. На основі огляду зроблено висновки про основні аспекти, які мають бути притаманні розроблюваному програмному забезпеченню.

На основі отриманої інформації, під час огляду сервісів безпеки, технологій та технічних рішень, розглянуто і обрано механізми та сервіси безпеки, які будуть використовуватись в ході розробки застосунку.

В результаті огляду обрані такі технології та сервіси:

- Google Drive в якості сервісу хмарного сховища;
- база даних MySQL була обрана як основний сервіс зберігання і захисту особистих даних користувачів;
- завдяки MySQL вдалося реалізувати вдалу модель політики безпеки, яка себе зарекомендувала у ході публічного тестування програмного забезпечення;
- Telegram Bot був взятий у якості клієнтської частини сервісу, так як месенджер Telegram використовує протокол MTProto 2.0, який побудований на перевірених часом алгоритмах, щоб зробити безпеку сумісною з високошвидкісною доставкою та надійністю при слабких з’єднаннях.

В ході виконання дипломної роботи реалізовано застосунок з використанням вищеперерахованих технологій та сервісів безпеки. Перевірено ефективність роботи фінансової організації в рамках тесту на одному з підрозділів. Під час тесту розроблений застосунок підвищив ефективність на 34,95%, що визначено на основі зібраних даних.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Как оптимизировать работу с помощью автоматизированных рабочих процессов [Електронний ресурс] – Режим доступу до ресурсу: https://www.wrike.com/ru/blog/kak-optimizirovat-rabotu-nad-zadachami-s-pomoshhyu-avtomatizirovannyh-rabochih-protsessov
2. The Top 5 Benefits of Workflow Automotion [Електронний ресурс] – Режим доступу до ресурсу: https://tallyfy.com/workflow-automation
3. What does workflow automation help you? [Електронний ресурс] – Режим доступу до ресурсу: https://kissflow.com/workflow/workflow-automation
4. Wikipedia / Drag and drop [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Drag_and_drop
5. Electronic Forms (e-Forms) [Електронний ресурс] – Режим доступу до ресурсу: https://www.gartner.com/en/information-technology/glossary/electronic-forms-e-forms
6. 10 of the Best Options for Workflow Automation Systems [Електронний ресурс] – Режим доступу до ресурсу: https://technologyadvice.com/blog/information-technology/top-10-workflow-automation-software
7. Secure your data with AES-256 encryption [Електронний ресурс] – Режим доступу до ресурсу: https://www.atpinc.com/blog/what-is-aes-256-encryption
8. Security Statement [Електронний ресурс] – Режим доступу до ресурсу: https://www.processmaker.com/security-statement
9. Security concepts and mechanisms [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/docs/en/ibm-mq/7.5?topic=overview-concepts-mechanisms

10. Wikipedia Security service (telecommunication) [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Security_service_(telecommunication)#cite_note-Stal-3
11. What is cloud storage? [Електронний ресурс] – Режим доступу до ресурсу: https://searchstorage.techtarget.com/definition/cloud-storage
12. Andrew C. Pincher Amazon Drive vs Google Drive [Електронний ресурс] – Режим доступу до ресурсу: https://www.jotform.com/blog/amazon-drive-vs-google-drive
13. Difference between Google Drive and Amazon S3 [Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/difference-between-google-drive-and-amazon-s3
14. SQLite vs MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems [Електронний ресурс] – Режим доступу до ресурсу: https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems
15. What is a Relational Database (RDBMS)? [Електронний ресурс] – Режим доступу до ресурсу: https://www.oracle.com/database/what-is-a-relational-database
16. How secure is Telegram? [Електронний ресурс] – Режим доступу до ресурсу: https://telegram.org/faq#q-how-secure-is-telegram
17. MTProto Mobile Protocol [Електронний ресурс] – Режим доступу до ресурсу: https://core.telegram.org/mtproto/description
18. What is an Information Security Policy? [Електронний ресурс] – Режим доступу до ресурсу: https://www.upguard.com/blog/information-security-policy

19. The 8 Elements of an Information Security Policy [Електронний ресурс] – Режим доступу до ресурсу: https://www.exabeam.com/information-security/information-security-policy

ДОДАТОК А

А.1 Методи запитів по сервісів

```
def __connect__(self):
        self.connect = pymysql.connect(host=self.host, user=self.user,
                                       password=self.password, db=self.database)
        self.cur = self.connect.cursor()
def __disconnect__(self):
            self.connect.close()

def __exists__(self, telegram_id):
            self.__connect__()
            self.cur.execute('SELECT EXISTS(SELECT * FROM users
  WHERE Telegram_ID = {0});'.format(telegram_id))
            self.valid = re.sub("\D", "", str(self.cur.fetchone()))
            self.__disconnect__()
            return self.valid

def __exists_number__(self, number):
            self.__connect__()
            self.cur.execute('SELECT EXISTS(SELECT * FROM users
  WHERE Phone_Number = {0});'.format(number))
            self.valid_num = re.sub("\D", "", str(self.cur.fetchone()))
            self.__disconnect__()
            return self.valid_num
def __exists_manager__(self, telegram_id):
            self.__connect__()
```

```
            self.cur.execute('SELECT EXISTS(SELECT * FROM managers
  WHERE Telegram_ID = {0});'.format(telegram_id))
            self.valid = re.sub("\D", "", str(self.cur.fetchone()))
            self.__disconnect__()
            return self.valid

def __exists_number_manager__(self, number):
            self.__connect__()
            self.cur.execute('SELECT EXISTS(SELECT * FROM managers
  WHERE Phone_Number = {0});'.format(number))
            self.valid_num = re.sub("\D", "", str(self.cur.fetchone()))
            self.__disconnect__()
            return self.valid_num

def __if_block__(self, telegram_id):
            self.__connect__()
            self.cur.execute('SELECT Status FROM users WHERE
  Telegram_ID = {0};'.format(telegram_id))
            self.block_status = re.sub("\D", "", str(self.cur.fetchone()))
            self.__disconnect__()
            return self.block_status

def __if_admin__(self, telegram_id):
            self.__connect__()
            self.cur.execute('SELECT Role FROM users WHERE
  Telegram_ID = {0};'.format(telegram_id))
            self.admin_status = re.sub("\D", "", str(self.cur.fetchone()))
            self.__disconnect__()
            return self.admin_status
```

```
def __add_admin__(self, last_name, first_name, telegram_id, manager_id,
 phone_number):
        self.__connect__()
        self.cur.execute("INSERT INTO users (Last_Name, First_Name,
 Telegram_ID, Manager_ID, Phone_Number, Role, Status) "
                         "VALUES ('{0}', '{1}', '{2}', '{3}', {4}, 1,
 1);".format(last_name, first_name, telegram_id,
                                                    manager_id,
 phone_number))
        self.connect.commit()
        self.__disconnect__()

def __add_user__(self, last_name, first_name, telegram_id, manager_id,
phone_number):
        self.__connect__()
        self.cur.execute("INSERT INTO users (Last_Name, First_Name,
 Telegram_ID, Manager_ID, Phone_Number, Role, Status) "
                         "VALUES ('{0}', '{1}', '{2}', '{3}', {4}, 0,
 1);".format(last_name, first_name, telegram_id,
                                                    manager_id,
 phone_number))
        self.connect.commit()
        self.__disconnect__()

def __add_manager__(self, last_manager, first_manager, tg_id,
phone_manager, telegram_name, email_manager):
        self.__connect__()
        self.cur.execute(
```

```
                "INSERT INTO managers (Last_Name, First_Name,
  Telegram_ID, Phone_Number, Telegram_Name, Email, Status) "
                "VALUES ('{0}', '{1}', '{2}', '{3}', '{4}', '{5}',
  1);".format(last_manager, first_manager, tg_id,
                                                    phone_manager,
  telegram_name, email_manager))
                self.connect.commit()
                self.__disconnect__()

def __remove_user__(self, number):
                self.__connect__()
                self.cur.execute("DELETE FROM users WHERE Phone_Number
  = {0};".format(number))
                self.connect.commit()
                self.__disconnect__()

def __block_user__(self, number):
                self.__connect__()
                self.cur.execute("UPDATE users SET Status = 0 WHERE
  Phone_Number = {0};".format(number))
                self.connect.commit()
                self.cur.execute("SELECT Telegram_ID FROM users WHERE
  Phone_Number = {0};".format(number))
                self.__disconnect__()
                return re.sub("\D", "", str(self.cur.fetchone()))

def __unblock_user__(self, number):
                self.__connect__()
```

```
            self.cur.execute("UPDATE users SET Status = 1 WHERE
  Phone_Number = {0};".format(number))
            self.connect.commit()
            self.cur.execute("SELECT Telegram_ID FROM users WHERE
  Phone_Number = {0};".format(number))
            self.__disconnect__()
            return re.sub("\D", "", str(self.cur.fetchone()))

def __show_info__(self, phone_number):
            self.__connect__()
            self.cur.execute("SELECT * FROM users WHERE
  Phone_Number = {0};".format(phone_number))
            self.all_client_info = self.cur.fetchone()
            self.cur.execute("SELECT Last_Name, First_Name FROM
  managers WHERE Telegram_ID = {0};".format(self.all_client_info[4]))
            self.manager_info = self.cur.fetchone()
            self.__disconnect__()
            return self.all_client_info, self.manager_info

def __show_info_user__(self, tg_id):
            self.__connect__()
            self.cur.execute("SELECT * FROM users WHERE Telegram_ID
  = {0};".format(tg_id))
            self.all_client_info = self.cur.fetchone()
            self.cur.execute("SELECT Last_Name, First_Name FROM
  managers WHERE Telegram_ID = {0};".format(self.all_client_info[4]))
            self.manager_info = self.cur.fetchone()
            self.__disconnect__()
            return self.all_client_info, self.manager_info
```

```
def __get_all_ids__(self):
        self.__connect__()
        self.cur.execute("SELECT Telegram_ID FROM users;")
        self.__disconnect__()
        return self.cur.fetchall()

def __change_manager__(self, manager_id, client_phone):
        self.__connect__()
        self.cur.execute("SELECT Last_Name, First_Name, Telegram_ID,
 Manager_ID FROM users WHERE Phone_Number =
 {0}".format(client_phone))
        self.client_name = self.cur.fetchone()
        self.cur.execute("SELECT Last_Name, First_Name FROM
 managers WHERE Telegram_ID = {0}".format(manager_id))
        self.new_manager_name = self.cur.fetchone()
        self.cur.execute("UPDATE users SET Manager_ID = {0} WHERE
 Phone_Number = {1}".format(manager_id, client_phone))
        self.connect.commit()
        self.__disconnect__()
        return self.client_name, self.new_manager_name

def __find_by_first_and_last_name__(self, last_name_change,
first_name_change):
        self.__connect__()
        self.cur.execute("SELECT Telegram_ID FROM managers
 WHERE Last_Name = '{0}' AND First_Name =
 '{1}';".format(last_name_change,
```

```
  first_name_change))
                self.manager_change_id = self.cur.fetchone()
                self.__disconnect__()
                return self.manager_change_id[0]

def __all_managers__(self):
                self.__connect__()
                self.cur.execute('SELECT * FROM managers;')
                self.all_info_managers = self.cur.fetchall()
                return self.all_info_managers

def __manager__(self, client_chat_id):
                self.__connect__()
                self.cur.execute('SELECT Last_Name, First_Name, Manager_ID
  FROM users WHERE Telegram_ID = {0}'.format(client_chat_id))
                self.clients_manager_id = self.cur.fetchone()
                self.cur.execute('SELECT Last_Name, First_Name,
  Phone_Number, Telegram_Name, Email FROM managers WHERE
  Telegram_ID = {0}'.format(self.clients_manager_id[2]))
                self.manager_cl_info = self.cur.fetchone()
                self.__disconnect__()
        return self.clients_manager_id, self.manager_cl_info
```

А.2 Методи запитів у користувача додаткової інформації для подальшої обробки

```
@dp.message_handler(lambda message: not int(re.findall(r'\w+', str(message.text))[0]) in range(1, 10000001), state=PortfolioConstruct.enter_sum)
    async def process_enter_sum_invalid(message: types.Message, state: FSMContext):
        if int(DBFreedom().__exists__(message.chat.id)) and int(DBFreedom().__if_block__(message.chat.id)):
    await message.answer("Некорректный ввод! Пожалуйста, введите желаемую сумму (только цифры до 10 млн.).")

@dp.message_handler(lambda message: message.text.isdigit(), state=PersonalPortfolio.enter_city)
async def process_enter_city_invalid(message: types.Message):
  if int(DBFreedom().__exists__(message.chat.id)) and int(DBFreedom().__if_block__(message.chat.id)):
    await message.answer("Некорректный ввод! Пожалуйста, введите Ваш город.")


@dp.message_handler(lambda message: message.text not in ["Предыдущие", "Предстоящие", "Все IPO", "FWD_IPO", "Посмотреть статистику", "В меню"], state=IPO.select_date)
async def process_ipo_invalid(message: types.Message):
  if int(DBFreedom().__exists__(message.chat.id)) and int(DBFreedom().__if_block__(message.chat.id)):
```

```
    await message.answer("Некорректный ввод! Пожалуйста, выберите вариант из списка.")


@dp.message_handler(lambda message: message.text not in ["10000$", "50000$", "100000$", "Конструктор Портфеля", "В меню"], state=Portfolio.select_port)
async def process_port_invalid(message: types.Message):
    if int(DBFreedom().__exists__(message.chat.id)) and int(DBFreedom().__if_block__(message.chat.id)):
        await message.answer("Некорректный ввод! Пожалуйста, выберите вариант из списка.")


@dp.message_handler(lambda message: message.text not in ["Long term ideas", "Mid term ideas", "Назад"], state=Stocks.select_term)
async def process_term_invalid(message: types.Message):
    if int(DBFreedom().__exists__(message.chat.id)) and int(DBFreedom().__if_block__(message.chat.id)):
        await message.answer("Некорректный ввод! Пожалуйста, выберите вариант из списка.")
```

А.3 Методи міжсерверної комунікації

```
 scope = [ "https://spreadsheets.google.com/feeds",
 'https://www.googleapis.com/auth/spreadsheets',
        "https://www.googleapis.com/auth/drive.file",
 "https://www.googleapis.com/auth/drive" ]

 creds = ServiceAccountCredentials.from_json_keyfile_name("credits.json",
 scope)

 client = gspread.authorize(creds)

class News:
   def __init__(self):
      self.sheet_news = client.open("STOCK NEWS").sheet1
      self.news_content = self.sheet_news.get_all_records()
      self.news_content.reverse()

   async def print_news(self, message):

      for self.news in self.news_content:
         if self.news['date'] != '' and datetime.strptime(self.news['date'],
'%d.%m.%Y').date() == datetime.now().date():
            await message.answer(parse_mode='HTML', text='<b>Новости на
сегодня:</b> \n'
                                 '{0}'.format(self.news['article']))
```

```
        elif self.news['date'] != '' and self.news == self.news_content[0]:
            await  message.answer(parse_mode='HTML',  text='<b>Последние
новости:</b> \n'

'{0}'.format(self.news_content[0]['article']))

class Actual:
    def __init__(self):
        self.iterator = 0
        self.all_ideas = []
        self.week_day = datetime.now().isocalendar()[2]
        self.start_date = datetime.now() - timedelta(days=self.week_day + 7 - 1)
        self.sheet_idea = client.open("ИДЕИ").sheet1

        self.dates_full = [str((self.start_date + timedelta(days=i)).date()) for i in
range(14)]
        self.dates_demo = [str((self.start_date + timedelta(days=i)).date()) for i in
range(3)]
    async def attract_people(self, bot, wait_for):
        try:
            while True:
                await asyncio.sleep(wait_for)
                if  datetime.today().strftime('%d')  ==  '15'  and  time(13,  0,  0)  <=
datetime.now().time() <= time(13, 0, 30):
                    for self.id in DBFreedom().__get_all_ids__():
                        try:
                            await bot.send_message(self.id[0], parse_mode='HTML',
                                                   text="")
```

```
                except exceptions.BotBlocked:
                    print(datetime.now().strftime('%d.%m.%Y %H:%M:%S') +
                          ": User {} loses attract notification!".format(self.id))

                except exceptions.UserDeactivated:
                    print(datetime.now().strftime(
                        '%d.%m.%Y %H:%M:%S') + ": User {} is
deactivated!".format(self.id))
                except exceptions.ChatNotFound:
                    print(datetime.now().strftime(
                        '%d.%m.%Y %H:%M:%S') + ": Chat {} is not
found!".format(self.id))
```

А.4 Методи, які працюють асинхронно у фоновому режимі

```
if __name__ == '__main__':
    dp.loop.create_task(DBFreedom().__ipo_scheduled__(bot, 27))
    dp.loop.create_task(DBFreedom().__ipo_deadline__(bot, 27))
    dp.loop.create_task(all_tickers_update(30))
    dp.loop.create_task(DBFreedom().__subs_alert__(bot, 1800))
    dp.loop.create_task(start())
    dp.loop.create_task(DBFreedom().__activity_report__(27))
    dp.loop.create_task(Actual().new_idea(bot, 3600))
    dp.loop.create_task(Actual().attract_people(bot, 27))
    executor.start_polling(dp, skip_updates=True)
```