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

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

«На правах рукопису»
УДК _004.67_

До захисту допущено:
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2024 р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою
«Інформаційні управляючі системи та технології»

зі спеціальності 126 «Інформаційні системи та технології»

на тему: «Система підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку»

Виконав:
студент 2 курсу, групи ІС-32мп
Мінін Павло Андрійович _____

Керівник:
доцент каф. ІСТ, к.т.н.
Букасов Максим Михайлович _____

Рецензент:
доцент каф. ІПІ, к.т.н., доц.
Ліхоузова Тетяна Анатоліївна _____

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

Київ – 2024 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – другий (магістерський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма

«Інформаційні управляючі системи та технології»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2023 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Мініну Павлу Андрійовичу

1. Тема дисертації «Система підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку», науковий керівник дисертації Букасов Максим Михайлович, к.т.н, затверджені наказом по університету від «08» 11 2024 р. № 5016-с

2. Термін подання студентом дисертації «09» 12 2024 р.

3. Об’єкт дослідження: процес аналізу та оцінки портфельних кредитних ризиків банку

4. Вихідні дані: довідники НБУ для складання статистичної звітності та інша інформація у відкритому доступі

5. Перелік завдань, які потрібно розробити: розробити систему підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку, описати предметне середовище, пояснити мету розробки та проблему, що вирішується, обґрунтувати вибір використаного програмного та технічного забезпечення

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: структурна схема, діаграма розгортання, блок-схема імпорту файлів, блок-схема алгоритму парсингу, діаграма послідовності імпорту даних до

системи та створення дашборду по нормалізованим даним, діаграма сценаріїв використання, діаграма класів, ER-діаграма сховища даних.

7. Орієнтовний перелік публікацій: тези «Автоматизація аналізу портфельних кредитних ризиків банку», представлені на V Міжнародній науково-практичній конференції молодих вчених та студентів «Інженерія програмного забезпечення і передові інформаційні технології SoftTech-2023».

8. Дата видачі завдання 02.09.2024 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Прибуття здобувача вищої освіти на практику, оформлення і отримання перепусток	01.09.24	
2.	Проведення інструктажу з техніки безпеки та охорони праці	до 03.09.24	
3.	Виконання програми практики і індивідуального завдання відповідно до теми		
3.1	Аналіз предметної області	до 10.09.24	
3.2	Огляд наявних рішень для аналізу портфельних ризиків кредитного портфелю банку	до 16.09.24	
3.3	Постановка та формалізація задач підсистеми	до 20.09.24	
3.4	Розробка програмного забезпечення	до 15.10.24	
3.5	Аналіз функціоналу розробленого рішення	до 22.10.24	
4.	Оформлення щоденника, звіту і складання заліку з практики	до 26.10.24	
5.	Оформлення магістерської дисертації для проходження попереднього захисту	до 25.11.2024	
6.	Усунути зауваження, отримані на попередньому захисті, та надіслати роботу на нормоконтроль	до 06.12.2024	

Студент Павло МІНІН

Науковий керівник Максим БУКАСОВ

РЕФЕРАТ

Система підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку: 121 с., 25 табл., 9 рис., 8 дод., 20 джерел.

АНАЛІЗ РИЗИКІВ, БАНК, ВІЗУАЛІЗАЦІЯ, ІМПОРТ ДАНИХ, ІНТЕГРАЦІЯ, КРЕДИТНИЙ ПОРТФЕЛЬ, НОРМАЛІЗАЦІЯ ДАНИХ, СХОВИЩЕ ДАНИХ, ELT, BI-СИСТЕМА.

Актуальність теми. Управління кредитними ризиками є важливим аспектом діяльності банків, оскільки їх недостатній аналіз може призвести до значних фінансових втрат. На сьогодні частина українських банків використовує застарілі інструменти, такі як Excel, які не забезпечують швидкість і точність оцінок ризиків. Впровадження інтерактивних систем аналізу портфельних ризиків із сучасними аналітичними інструментами є актуальним завданням для підвищення ефективності банківської діяльності.

Зв’язок роботи з науковими програмами. Дисертація відповідає науковим планам кафедри інформаційних систем та технологій і узгоджена із сучасними тенденціями фінансового аналізу та вимогами НБУ.

Мета і задачі дослідження. Мета роботи – надати інтерактивну платформу, яка дозволить проводити глибокий аналіз портфельних кредитних ризиків, забезпечуючи більш ефективне управління банківськими активами. Основні завдання включають розробку додатку імпорту файлів звітності різних форматів, побудову сховища даних та інтеграцію BI-системи для візуалізації даних.

Об’єкт і предмет дослідження. Об’єктом є процес аналізу та оцінки портфельних кредитних ризиків банку, а предметом – система підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку.

Практичне значення. Система може бути впроваджена в банківській сфері для покращення якості управління кредитними портфелями, зменшення фінансових втрат та оптимізації процесів аналітики.

ABSTRACT

Interactive Analysis Support System for Bank Loan Portfolio Risks: 121 pages, 25 tables, 9 figures, 8 appendices, 20 references.

RISK ANALYSIS, BANK, VISUALIZATION, DATA IMPORT, INTEGRATION, LOAN PORTFOLIO, DATA NORMALIZATION, DATA WAREHOUSE, ELT, BI SYSTEM.

Relevance of the topic. Credit risk management is an important aspect of banks' operations, as insufficient analysis of credit risks can lead to significant financial losses. Today, some Ukrainian banks use outdated tools such as Excel, which do not ensure the speed and accuracy of risk assessments. Implementation of interactive portfolio risk analysis systems with modern analytical tools is an urgent task to improve the efficiency of banking activities.

Relationship to research programmes. The dissertation is in line with the research plans of the Department of Information Systems and Technologies and is consistent with current trends in financial analysis and the requirements of the NBU.

Purpose and Objectives of the Study. The purpose of this work is to provide an interactive platform that enables in-depth analysis of portfolio credit risks, ensuring more efficient management of banking assets. The primary objectives include developing an application for importing reporting files of various formats, building a data warehouse, and integrating a BI system for data visualization.

Object and Subject of the Study. The object of the study is the process of analysing and assessing the portfolio credit risks of a bank, while the subject is a system for supporting the interactive analysis of portfolio risks in a bank's credit portfolio.

Practical Significance. The system can be implemented in the banking sector to improve the quality of credit portfolio management, reduce financial losses, and optimize analytical processes.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

АБС – Автоматизована Банківська Система

Базель III – документ Базельського комітету з банківського нагляду, що містить методичні рекомендації в області банківського регулювання і затверджений в 2010–2011 рр.

НБУ – Національний банк України

ОС – операційна система

ПЗ – програмне забезпечення

СКБД – система керування базами даних

AI – Artificial Intelligence – штучний інтелект

ACID – Atomicity, Consistency, Isolation, Durability – набір властивостей, що гарантують надійну роботу транзакцій бази даних: атомарність, узгодженість, ізольованість, довговічність

API – Application Programming Interface – певний контракт, що визначає, якіз застосунком можуть взаємодіяти інші

BASEL – Basel Committee on Banking Supervision – Базельський комітет з питань банківського нагляду

BI – Business Intelligence – комп'ютерні методи і інструменти для організацій, що забезпечують переклад транзакційної ділової інформації в форму, придатну для бізнес-аналізу, а також засоби для роботи з обробленою таким чином інформацією.

CLI – Command-line Interface – інтерфейс командного рядка

CRM – Customer Relationship Management – управління відносинами з клієнтами, поняття, що охоплює концепції, котрі використовуються компаніями для управління взаємовідносинами зі споживачами, включаючи збір, зберігання й аналіз інформації про споживачів, постачальників, партнерів та інформації про взаємовідносини з ними

CRUD – Create Read Update Delete – 4 основні функції управління даними: створення, читання, оновлення і вилучення

CS – українська IT-компанія «CS Ltd.»

CSV – Comma Separated Value – формат структурованих текстових даних, у якому значення розділяються певним символом, зазвичай комою

DAX – Data Analysis Expressions – набір функцій, операторів і констант, які можна використовувати у формулі або виразі у Power BI або MS Excel для обчислення та повернення одного або кількох значень

DTO – Data Transfer Object – абстрактний інтерфейс до бази даних або іншого сховища

ELT – Extract, Load, Transform – підхід до обробки даних, за яким дані спочатку виймаються з джерел, завантажуються у сховище, а потім трансформуються безпосередньо у місці зберігання

ETL – Extract, Transform, Load – процес інтеграції даних, що включає їх виймання з джерел, трансформацію для очищення та структурування, а потім завантаження у цільове сховище.

Excel-файл – файл програми для роботи з електронними таблицями Microsoft Excel. Зазвичай має розширення файлу .xls або .xlsx

FinTech – фінансові технології

JDBC – Java DataBase Connectivity – прикладний програмний інтерфейс Java, який визначає методи, з допомогою яких програмне забезпечення на Java здійснює доступ до бази даних

JPA – Java Persistence API – стандартизований інтерфейс для Java Object-relational mapping фреймворків

JSON – JavaScript Object Notation – текстовий формат обміну даними між комп'ютерами

JVM – Java Virtual Machine – віртуальна машина для виконання байт-коду Java

KPI – Key Performance Indicators – ключові показники ефективності, фінансова та нефінансова система оцінки, яка допомагає організації визначити ступінь досягнення стратегічних цілей

Lime Systems – українська IT-компанія «Lime Systems»

MS – Microsoft

MVC – Model-view-controller – архітектурний шаблон, який використовується під час проєктування та розробки програмного забезпечення, що передбачає поділ системи на три взаємопов'язані частини: модель даних, вигляд (інтерфейс користувача) та модуль керування

NPL – Non-performing Loans – непрацюючі кредити, активи, за якими прострочення погашення боргу перевищує 90 днів (30 днів для банків-боржників)

REST – Representational State Transfer – стиль написання веб-застосунків, описує як користувачу слід організовувати написання коду веб-застосунку

SQL – Structured Query Language – мова написання запитів, за допомогою якої можна взаємодіяти із базами даних

TXT – формат файлів, що являють собою текстові документи, які містять простий текст у вигляді рядків

UNITY-BARS – українська IT-компанія «УНІТІ-БАРС»

VPN – Virtual Private Network – віртуальна приватна мережа

XML – Extensible Markup Language – формат структурованих текстових даних, який дозволяє зберігати різні об’єкти, описувати процеси тощо

ВСТУП

Аналіз ризиків є невід’ємною складовою управління будь-яким банком, якою займається підрозділ професійно навчених спеціалістів. Банки, як важливі учасники економіки, повинні постійно аналізувати ризики, пов'язані з кредитуванням, щоб мінімізувати можливі втрати. Однак ефективність такого аналізу часто залежить від інструментів, які використовуються для обробки даних. На сьогодні, хоч банківська галузь і активно впроваджує новітні технології, деякі установи все ще покладаються на застарілі підходи, що базуються на використанні офісних пакетів, які не завжди забезпечують необхідну гнучкість і швидкість. Однією з ключових складових кредитного аналізу є оцінка портфельних ризиків, яка дозволяє визначати потенційні загрози для банку на ранніх етапах і вживати необхідних заходів для їх мінімізації. Враховуючи швидкі зміни на фінансовому ринку, вдосконалення методів аналізу ризиків є критично важливим завданням.

Актуальність проблеми. На сьогодні актуальним завданням є розробка системи, яка б забезпечувала ефективний аналіз кредитних портфельних ризиків із використанням сучасних технологій обробки та візуалізації даних. Окремі банки все ще використовують застарілі засоби, такі як офісні пакети, що ускладнює процеси аналізу та управління ризиками. Це веде до зниження ефективності роботи аналітиків та збільшення часу на прийняття рішень. Актуальність створення системи підтримки інтерактивного аналізу портфельних кредитних ризиків полягає у необхідності підвищення якості оцінки ризиків та оптимізації процесів управління кредитними портфелями за рахунок автоматизації і впровадження сучасних аналітичних рішень.

Об'єктом роботи є процес аналізу та оцінки портфельних кредитних ризиків банку.

Предметом роботи є система підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку, яка використовує трирівневу архітектуру сховища даних і BI-системи для аналізу та візуалізації результатів.

Мета. У цьому контексті метою роботи є надати інтерактивну платформу, яка дозволить проводити глибокий аналіз портфельних кредитних ризиків, забезпечуючи більш ефективне управління банківськими активами. Система повинна інтегрувати сучасні інструменти бізнес-аналітики (BI), що дозволить проводити швидкий аналіз та візуалізацію результатів.

Для досягнення цієї мети необхідно було вирішити наступні завдання:

– розробити інтерфейс для введення параметрів аналізу, що цікавлять користувача;

– забезпечити сумісність системи з сучасними базами даних та технологіями візуалізації даних.

Практичне значення полягає в тому, що така система стане корисним інструментом для банківських аналітиків, допомагаючи швидко й ефективно визначати ризики, приймати управлінські рішення та знижувати фінансові втрати.

1 ВИЗНАЧЕННЯ ПРОБЛЕМАТИКИ

Аналіз ризиків – це комплексний процес оцінки потенційних загроз, які можуть вплинути на фінансову стабільність банку. Цей процес дозволяє ідентифікувати ризики, оцінити їхню ймовірність та вплив, а також розробити стратегії для мінімізації негативних наслідків. На перший погляд, аналіз ризиків може виглядати як звичайна оцінка фінансових показників, але коли це стосується великих банків з тисячами клієнтів та складними кредитними портфелями, процес стає значно більш багатогранним та комплексним.

У повсякденній діяльності банків аналіз ризиків є невід’ємною частиною управління фінансовими операціями, що охоплює кредитування, інвестиції, операції з цінними паперами та інші напрямки. Ризики можуть виникати як на рівні окремих угод, так і на рівні загального портфеля активів, що робить необхідним інтегрований підхід до їх оцінки та управління.

Аналіз ризиків є важливою складовою управління банківськими процесами, яка забезпечує безперебійну роботу фінансових установ і допомагає зменшити втрати від непередбачуваних ситуацій.

При управлінні кредитними портфелями ризик-аналітики повинні знати про потенційні загрози, які можуть вплинути на стабільність банку, постійно інформувати керівництво та зацікавлені сторони, а також забезпечувати видимість усіх ризиків, пов'язаних з портфелем. Для цього у нагоді можуть стати спеціалізовані системи аналітики ризиків, які б допомогли автоматизувати процеси оцінки та управління ризиками, що робить їх більш ефективними та надійними.

Таким чином, аналітика ризиків у банку є ключовим елементом управління фінансовими потоками та стабільністю, що забезпечує контроль та зниження ризиків для ефективної діяльності банку.

1.1 Визначення проблем, що вирішуються

Аналітика ризиків у банку вирішує низку важливих проблем, пов'язаних з управлінням фінансовими ризиками та забезпеченням стабільності банківської діяльності.

Аналітика допомагає виявляти потенційні загрози, які можуть виникнути у кредитному портфелі, інвестиціях, операціях з активами або пасивами. Це можуть бути як внутрішні, так і зовнішні фактори, такі як коливання ринкових умов, зміни законодавства або економічні кризи.

Однією з ключових проблем банків є правильна оцінка ризиків: визначення ймовірності настання небажаних подій і їхнього впливу на фінансову стабільність банку. Аналітика дозволяє точно визначити розмір кредитного ризику, ринкового ризику (наприклад, валютного або процентного) та операційного ризику.

Аналітика ризиків допомагає моделювати й оцінювати можливі наслідки різних сценаріїв для банківського портфеля. Це дозволяє керівництву банку приймати проактивні рішення для мінімізації потенційних збитків.

Аналітичні інструменти дозволяють оптимізувати структуру кредитного портфеля, зменшувати концентрацію ризиків та підвищувати його ефективність. Це допомагає банкам приймати більш зважені рішення щодо видачі кредитів і управління активами.

Аналітика ризиків допомагає виявляти ознаки можливого дефолту позичальників на ранніх стадіях, що дозволяє банку вжити заходів для уникнення втрат або реструктуризації боргу.

Банки повинні відповідати вимогам регуляторів щодо управління ризиками, зокрема мати достатні резерви для покриття можливих збитків. Аналітика ризиків допомагає банкам ефективно виконувати ці вимоги, зокрема відповідати стандартам, таким як Базель III.

Аналітичні системи забезпечують детальну звітність для внутрішніх і зовнішніх користувачів (керівництва банку, акціонерів, регуляторних органів), що дозволяє підвищити прозорість діяльності та підвищити довіру до банку.

Ефективна аналітика ризиків дозволяє банкам знижувати кількість непередбачуваних збитків через кредитні або ринкові шоки, що, в свою чергу, підвищує їх фінансову стійкість.

Впровадження систем аналітики ризиків дозволяє автоматизувати значну частину процесів управління ризиками, що зменшує ймовірність людських помилок і підвищує ефективність банку.

Аналітика ризиків є важливим інструментом для ефективного управління банківською діяльністю, дозволяючи своєчасно виявляти загрози, мінімізувати ризики та приймати зважені рішення для забезпечення фінансової стабільності.

1.2 Головні принципи в аналітиці кредитних портфельних ризиків банку

Дотримання принципів дозволяє банкам здійснювати більш ефективний та стратегічний підхід до аналізу та управління ризиками, знижуючи ймовірність фінансових втрат та підвищуючи стабільність фінансових операцій.

Головні принципи в аналізі ризиків визначають підходи до ідентифікації, оцінки та управління ризиками у банківській діяльності. Вони допомагають забезпечити системний та ефективний аналіз ризикових ситуацій. Основні принципи включають:

– принцип ідентифікації ризиків. Першим кроком є виявлення всіх можливих загроз та ризиків, що можуть виникнути в банківській діяльності. Це охоплює аналіз зовнішніх і внутрішніх факторів, які можуть вплинути на банк. Важливо не пропустити жодного критичного ризику, оскільки недооцінка загроз може призвести до серйозних фінансових втрат;

– принцип системності. Аналіз ризиків має бути систематичним і охоплювати всі аспекти діяльності банку. Це означає, що ризики повинні оцінюватися комплексно, з урахуванням їх взаємозв'язків. Часто один ризик може впливати на інші, тому необхідно враховувати системний ефект для прийняття зважених рішень;

– принцип об'єктивності та точності. Ризики повинні оцінюватися на основі об'єктивних даних і достовірної інформації. Це дозволяє уникати суб'єктивних помилок та упереджень. Використання сучасних аналітичних інструментів, таких як моделі статистичного аналізу та прогнозування, дозволяє підвищити точність оцінки;

– принцип пріоритизації. Не всі ризики мають однаковий рівень важливості, тому необхідно розставляти пріоритети. Більшу увагу слід приділяти ризикам, які мають найвищу ймовірність виникнення і можуть спричинити значні втрати для банку. Це допомагає концентрувати ресурси на критично важливих аспектах управління ризиками;

– принцип динамічності. Ризики не є статичними, вони можуть змінюватися з часом під впливом економічних умов, ринкових тенденцій або внутрішніх змін у банку. Тому аналіз ризиків повинен бути постійним процесом, з регулярним переглядом та оновленням оцінок на основі нової інформації;

– принцип превентивності. Основною метою аналізу ризиків є попередження можливих загроз і мінімізація їхніх наслідків. Важливо розробляти превентивні заходи, які допоможуть уникнути або мінімізувати вплив ризиків до того, як вони реалізуються;

– принцип відповідності регуляторним вимогам. Банки повинні враховувати всі регуляторні вимоги щодо управління ризиками, зокрема вимоги, встановлені міжнародними стандартами, такими як Базель III, а також внутрішніми державними нормами. Це забезпечує відповідність політикам управління ризиками на всіх рівнях;

– принцип економічної ефективності. Аналіз ризиків та заходи з управління ними повинні бути економічно обґрунтованими. Це означає, що витрати на управління ризиками мають бути пропорційними до потенційних втрат. Необхідно обирати рішення, які приносять максимальний ефект за мінімальних витрат;

– принцип інтеграції. Аналіз ризиків повинен бути інтегрованим у загальну систему управління банком. Це забезпечує синергію між різними підрозділами банку та сприяє кращій координації дій для мінімізації ризиків;

– принцип прозорості та звітності. Процес аналізу ризиків має бути прозорим для всіх зацікавлених сторін. Регулярне звітування про ризики та заходи, вжиті для їхнього управління, допомагає підтримувати довіру з боку акціонерів, регуляторів і клієнтів.

Висновки до розділу 1

У цьому розділі визначено роль аналізу ризиків як комплексного процесу, що забезпечує ідентифікацію, оцінку та управління потенційними загрозами, які можуть вплинути на фінансову стабільність банку. Особливу увагу приділено багатогранності процесу в умовах великих банків з численними клієнтами та складними кредитними портфелями. Встановлено, що аналіз ризиків є фундаментальним елементом управління банківською діяльністю, спрямованим на мінімізацію збитків і підтримання безперервної роботи фінансових установ.

Охарактеризовано ключові проблеми, які вирішує аналітика ризиків. Зокрема, ідентифікація загроз, оцінка кредитного, ринкового та операційного ризиків, моделювання сценаріїв для прогнозування наслідків і дотримання регуляторних вимог, таких як Базель III. Відзначено важливість автоматизації аналітичних процесів і використання сучасних інструментів для підвищення ефективності управління ризиками.

Були систематизовані принципи аналізу кредитних портфельних ризиків. Особливий акцент зроблено на необхідності системності, об'єктивності, пріоритизації ризиків і регулярному оновленні оцінок. Принципи превентивності, економічної ефективності та прозорості визначено як основні для побудови надійної системи управління ризиками. Дотримання цих принципів дозволяє банкам знижувати ймовірність фінансових втрат, забезпечувати стабільність і підвищувати довіру з боку зацікавлених сторін.

2 ПОСТАНОВКА ЗАДАЧІ

На сьогоднішній день у сфері українського банківського сектору існує гостра проблема в аналітичній обробці великої кількості фінансової інформації, як зовнішньої ринкової, так і внутрішньобанківської. Хоча останні десять років показали стрімкий розвиток FinTech-сектору, зокрема таких напрямків, як CRM-системи, штучний інтелект, аналітичні інструменти для роботи з клієнтськими базами та інтерактивні засоби комунікації, аналітичні системи, пов'язані з оцінкою банківських ризиків, залишаються на менш розвиненому рівні. Це призводить до серйозних ускладнень у процесі управління кредитними ризиками та іншими видами ризиків, що можуть виникати під час банківських операцій.

Однією з основних проблем є значне збільшення потоку інформації внаслідок зростаючих обсягів транзакцій, постійної необхідності оновлення фінансових показників і швидкої зміни ринкових умов. У багатьох українських банках досі використовуються застарілі інструменти для ризик-менеджменту, такі як MS Excel, що обмежує можливості для швидкого і якісного аналізу великих масивів даних. Ручне оброблення даних є не тільки ресурсомістким, але й схильним до помилок, що створює додаткові ризики для банківської діяльності. Крім того, темпи зростання обсягу інформації випереджають можливості традиційних систем, що підвищує потребу в більш продуктивних аналітичних рішеннях.

Додатковою складністю є часті зміни у вимогах до звітності з боку Національного банку України (НБУ), що вимагає регулярного оновлення аналітичних інструментів і шаблонів. Ці зміни ускладнюють роботу ризик-аналітиків, які змушені адаптуватися до нових стандартів звітності. Більше того, в умовах мінливої економічної ситуації банкам стає дедалі важче відповідати не лише вимогам регуляторів, а й швидко адаптуватися до нових ринкових умов і загроз.

На тлі цих проблем злиття традиційних підходів до управління ризиками з інноваційними технологіями, такими як інтеграція штучного інтелекту та

автоматизація аналітичних процесів, може стати вирішальним фактором для підвищення ефективності ризик-менеджменту. Використання сучасних BI-систем і спеціалізованих аналітичних інструментів дозволить банкам ефективніше працювати з великими обсягами даних, виявляти потенційні ризики на ранніх етапах і своєчасно на них реагувати. Інтеграція інноваційних рішень також сприятиме підвищенню конкурентоспроможності банківських установ, забезпечуючи відповідність вимогам регуляторів і зменшуючи ймовірність фінансових втрат через непередбачені ризики.

Метою дипломної роботи було оптимізувати процес аналізу портфельних ризиків кредитного портфелю банку.

2.1 Актуальність і проблематика

Основним банківським бізнесом в Україні є надання кредитів як фізичним особам, так і юридичним суб'єктам. У світлі цього банки повинні ретельно контролювати рівень кредитного ризику, оскільки він є однією з ключових складових їхньої діяльності. Належне управління кредитними ризиками сприяє забезпеченню стійкості банківської системи і зниженню ймовірності настання фінансових втрат, що можуть виникати у разі неспроможності позичальників виконати свої зобов'язання. Відтак, для кожної банківської установи особливо важливо підтримувати достатній рівень капіталу для покриття можливих збитків, які можуть виникнути у випадку невиконання зобов'язань за кредитами.

З огляду на це, Національний банк України (НБУ) приділяє особливу увагу регулюванню кредитної діяльності банків, впроваджуючи строгі вимоги щодо адекватності капіталу. Одним із ключових показників, який визначає здатність банку покривати свої ризики, є норматив адекватності регулятивного капіталу (Н2). Цей норматив регулюється Національним банком України і має на меті забезпечити, щоб банки мали достатньо капіталу для покриття кредитних ризиків, що виникають у процесі їхньої діяльності. Наявність необхідного рівня капіталу

не лише сприяє фінансовій стабільності банку, а й допомагає зберегти довіру клієнтів та інвесторів до банківської системи загалом.

Регулятивні вимоги до капіталу, встановлені НБУ, також виконують важливу функцію підтримки стабільності фінансової системи країни. Забезпечення належного рівня капіталу дає змогу банкам витримати потенційні фінансові шоки, спричинені як внутрішніми, так і зовнішніми факторами, такими як економічні кризи або неспроможність позичальників виконати свої кредитні зобов'язання. Окрім цього, ці вимоги дозволяють банкам продовжувати функціонувати навіть в умовах непередбачених економічних змін, сприяючи зниженню ризику виникнення системних криз і забезпечуючи довготривалу стабільність фінансового сектору України.

Таблиця 2.1. – Структура навантаження на регулятивний капітал банків України станом на 01.11.2023 р., тис. грн [1]

Група ризику	Коефіцієнт ризику	Сума	Сума активів зважених на коефіцієнт ризику
1	*2*	*3*	*4=2*3*
I група ризику	0%	1 844 801 829	0
II група ризику	10%	335 692	33 569
III група ризику	20%	383 099 988	76 619 998
IV група ризику	30%	0	0
V група ризику	35%	0	0
VI група ризику	50%	104 540 358	52 270 179
VII група ризику	75%	0	0
VIII група Кредитний Портфель	100%	669 873 273	669 873 273
VIII група інші активи	100%	23 211 716	23 211 716
Валютний ризик	x	x	105 604 675
Операційний ризик	x	x	120 791 020

Група ризику	Коефіцієнт ризику	Сума	Сума активів зважених на коефіцієнт ризику
Сумарні активи зважені на відповідний коефіцієнт ризику залежно від групи ризику (Ар)	х	х	822 008 735
Величина непокритого кредитного ризику (НКР)	х	х	-3 476 963
Загальна сума регулятивного капіталу (РК) (Н1)	х	х	264 480 907
Середнє значення нормативу Н2	х	х	25,31%

Станом на 01.11.2023 р., середньозважене значення адекватності регулятивного капіталу банківської системи України становило 25,31%, що значно перевищує мінімальну вимогу в 10%, встановлену Національним банком України. Це свідчить про те, що банки мають значний запас капіталу для покриття можливих кредитних та інших ризиків, що виникають у процесі їхньої діяльності. Однак, незважаючи на високий рівень адекватності капіталу, залишається важливим питання щодо ефективного управління цим капіталом та мінімізації ризиків, пов'язаних з кредитуванням. Як показано в таблиці 1, високе значення адекватності капіталу забезпечує стабільність банківської системи, проте суттєва частка активів спрямовується на покриття ризиків, пов'язаних із кредитним портфелем.

Зокрема, частка навантаження на регулятивний капітал, спричинена кредитним портфелем, досягла 64% станом на вищевказану дату (див. рис. 2.1.). Це підкреслює важливість ефективного моніторингу та управління кредитними ризиками для підтримки стійкості банківської системи. Велика частина ресурсів банків спрямовується на управління цими ризиками, що знову ж таки підтверджує

критичну потребу у впровадженні надійних аналітичних інструментів, які дозволять банкам точніше прогнозувати можливі ризики та ухвалювати обґрунтовані рішення.

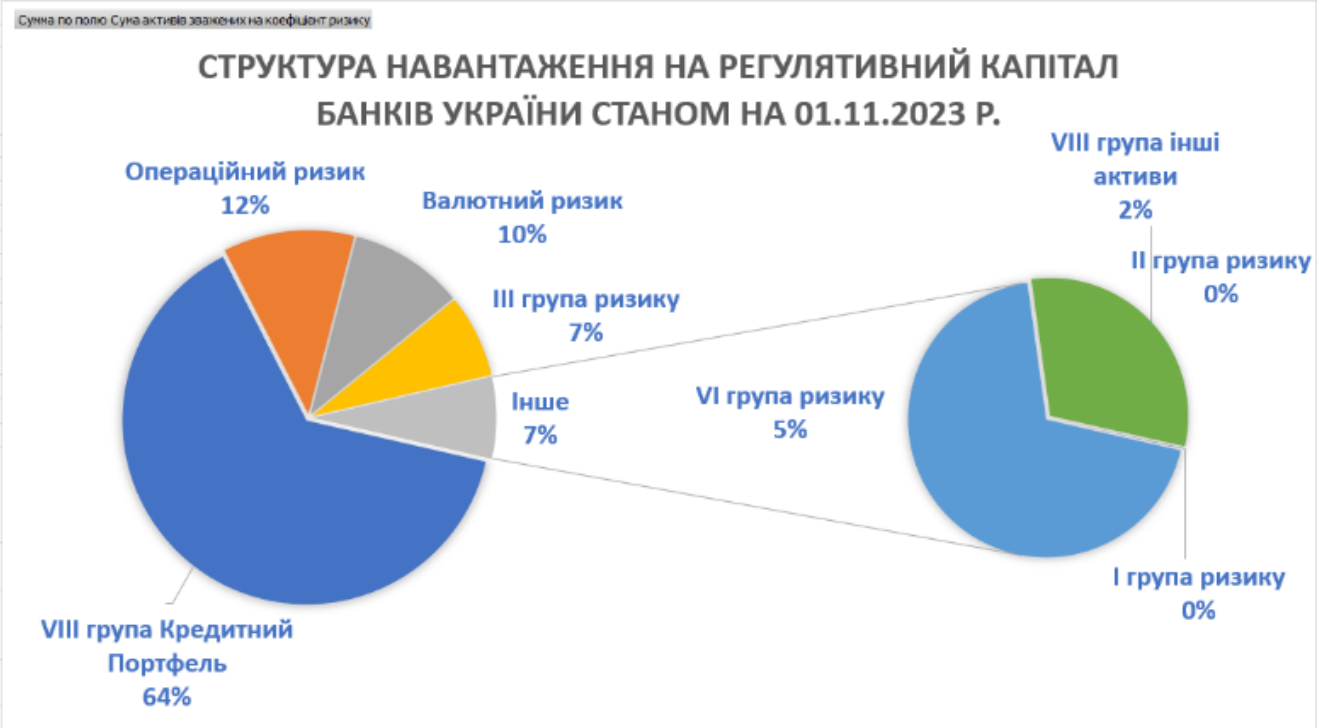

Рисунок 2.1 – Структура навантаження на регулятивний капітал банків України

Крім того, варто звернути увагу на структуру доходів банків. Станом на 01.11.2023 р. питома вага процентних доходів від кредитування фізичних та юридичних осіб складала 48% від загальних доходів банківської системи (див. таблицю 2). Це означає, що майже половина прибутків банків безпосередньо пов'язана з їхньою кредитною діяльністю. У зв'язку з цим, управління кредитними ризиками стає не лише питанням стабільності, а й ключовим фактором прибутковості банківської діяльності.

Вищезазначені дані підтверджують критичну необхідність впровадження спеціалізованих аналітичних інструментів для аналізу портфельних кредитних ризиків. Ці інструменти дозволять банкам ефективно оцінювати ризики, які виникають при наданні кредитів, а також забезпечать можливість оперативного

коригування кредитної політики в залежності від змін економічних умов або специфічних ризиків, пов'язаних із певними категоріями позичальників. Відтак, наявність належного інструментарію дозволить банкам не лише мінімізувати втрати, але й оптимізувати управління капіталом, підвищуючи тим самим загальну ефективність діяльності.

Таблиця 2.2. – Структура доходів банків України станом на 01.11.2023 р., тис. грн [1]

Показник	Сума	Питома вага
Процентні доходи від КП юридичних осіб	75 024 498,13	30,87%
Процентні доходи від КП фізичних осіб	41 733 182,15	17,17%
Процентні доходи від КП	116 757 680,28	48,03%
Інші доходи	126 315 204,51	51,97%
Всього доходів	243 072 884,79	100,00%

Станом на момент написання цієї роботи, частина банків України продовжує використовувати програмний комплекс MS Office Professional для аналізу портфельних кредитних ризиків на рівні окремих операцій. Такий підхід до організації процесу аналізу ризиків зображено на рисунку 2.2. Офісний пакет застосовується не тільки для внутрішнього аналізу, але й для обробки інформації, отриманої з зовнішніх джерел, таких як сайт Національного банку України, інші банківські установи, сайт Міністерства статистики та інші.

Незважаючи на широке використання MS Office у банківському середовищі, існують суттєві недоліки цього підходу, особливо коли мова йде про роботу з великими обсягами даних і складними інформаційними потоками.

Для обміну фінансовою звітністю багато банків користуються електронною поштою. Це швидкий та дієвий спосіб обміну файлами, проте це невдалий варіант для зберігання важливої інформації. У випадку, коли файл залишається тільки у поштовій скриньці, дані може бути втрачено серед незупинного потоку листів. Отриманий файл можна зберегти на пристрої, проте це вимагає місця на диску і не забезпечує спільного доступу до інформації з іншими співробітниками.

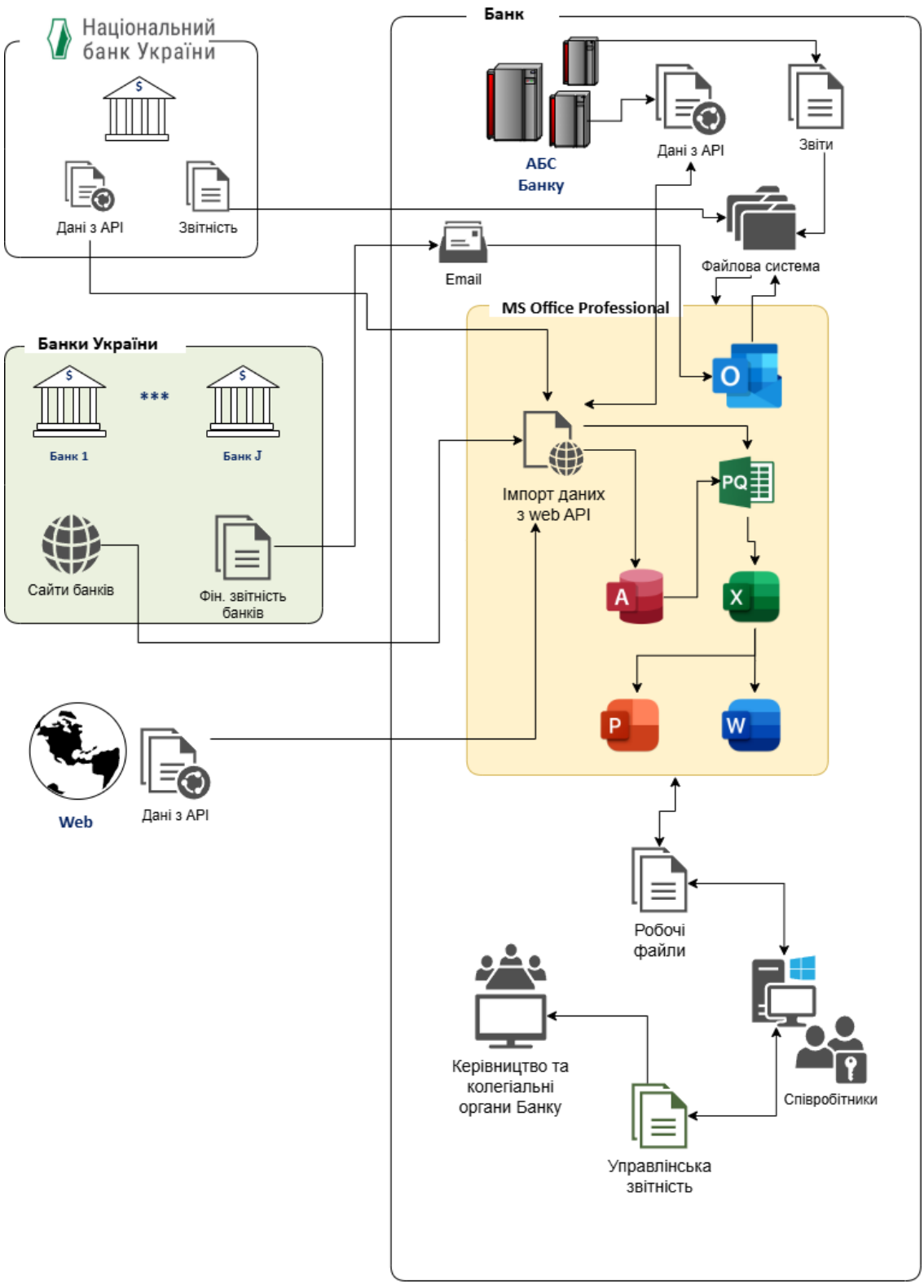

Рисунок 2.2 – Загальна структура автоматизації аналізу ризиків банку засобами MS Office

Ще більшою проблемою є обмежені можливості MS Office щодо обробки великих масивів інформації. Для проведення аналізу часто створюється безліч

окремих робочих файлів, кожен з яких може містити величезні обсяги даних. Наприклад, файл MS Excel для обробки від 250 до 300 тисяч рядків даних може займати до 250 МБ дискового простору. Відкриття такого файлу, навіть на сучасному комп'ютері, займає значний час – від 2,5 до 3 хвилин, що призводить до затримок в роботі ризик-аналітиків.

Такі тривалі процеси завантаження і обробки інформації знижують ефективність роботи фахівців, оскільки їм доводиться чекати на відкриття і маніпуляції з даними, що відбирає час для глибокого аналізу ризиків. До того ж, чим більше даних міститься у файлі, тим більша ймовірність виникнення технічних помилок або перевантаження системи, що може призвести до втрати інформації або порушення цілісності даних.

Крім цього, використання MS Office не забезпечує достатнього рівня автоматизації і інтеграції з іншими системами, що ускладнює синхронізацію даних між різними підрозділами банку. У результаті виникає необхідність у ручному введенні даних або їх перенесенні між системами, що збільшує ризик помилок і знижує загальну продуктивність. Така фрагментарність підходу робить процес аналізу кредитних ризиків громіздким і менш ефективним, особливо в умовах зростаючого обсягу інформації і ускладнення фінансових операцій.

2.2 Головні задачі засобу

Головною задачею було розробити систему, яка не лише допомогла б працівникам банків пришвидшити та покращити аналіз портфельних кредитних ризиків, але й забезпечила б інтеграцію з наявними процесами банку. Аналіз ризиків є ключовим компонентом успішної фінансової стратегії банку, однак, у багатьох банках досі використовуються застарілі підходи до аналізу, які передбачають ручну обробку даних у великих файлах MS Excel, що сповільнює процес та створює значні ризики помилок.

Розроблена система підтримки інтерактивного аналізу портфельних ризиків кредитного портфеля банку надає можливість автоматизованого збереження,

обробки і моделювання даних. Завдяки цьому аналітики можуть більше зосередитися на прийнятті рішень, а не на рутинних операціях з підготовки даних. Автоматизовані моделі прогнозування показників дозволяють прогнозувати фінансові ризики з більшою точністю та надавати результати в зручному форматі звітів і дашбордів, що спрощує презентацію інформації для керівництва. Це, в свою чергу, дозволяє приймати обґрунтовані рішення швидше та на більш надійній основі.

Ключовою перевагою впровадження такої системи є те, що вона дозволяє відмовитися від використання громіздких робочих файлів і значно пришвидшує перебіг внутрішніх процедур банку. Наприклад, замість зберігання і обробки даних у кількох великих Excel-файлах, система дозволяє зберігати дані централізовано і доступно, що зменшує ймовірність дублювання або втрати інформації. Це також сприяє покращенню співпраці між різними підрозділами банку, оскільки система забезпечує доступ до актуальних даних у режимі реального часу, що є критичним для ефективного управління кредитними ризиками.

Таким чином, впровадження цієї системи сприятиме значному підвищенню ефективності роботи банку, оскільки вона дозволяє покращити точність прогнозування фінансових результатів, забезпечити більш комплексне управління ризиками та відповідати сучасним вимогам ринку. Вона не тільки оптимізує процес аналізу кредитних ризиків, але й дозволяє банку краще контролювати свою фінансову стабільність, знижуючи ризики та підвищуючи прозорість прийняття рішень [2].

Висновки до розділу 2

У цьому розділі окреслено основні проблеми аналітики ризиків у банківському секторі України, що пов'язані із застарілими підходами до обробки великих обсягів даних, такими як використання MS Excel. Зростання кількості транзакцій, зміни в ринкових умовах і часті оновлення регуляторних вимог

створюють значне навантаження на ризик-аналітиків та ускладнюють управління кредитними ризиками. Визначено, що традиційні методи не забезпечують належної швидкості, автоматизації та інтеграції для ефективного моніторингу ризиків, що знижує стабільність і прибутковість банківської діяльності.

Підкреслено важливість інтеграції інноваційних технологій, таких як системи бізнес-аналітики (BI), які дозволять автоматизувати процеси оцінки та прогнозування ризиків. Використання таких рішень сприятиме зменшенню кількості помилок, пришвидшенню аналізу даних і підвищенню ефективності прийняття управлінських рішень.

Актуальність дослідження підтверджується критичною роллю кредитного ризику в банківській діяльності, зокрема у формуванні доходів і підтримці фінансової стабільності. Регулятивні вимоги встановлені НБУ, зокрема норматив адекватності регулятивного капіталу (Н2), підкреслюють важливість аналізу портфельних ризиків кредитного портфелю, що підтверджує потребу у вдосконалених інструментах аналітики.

Визначено ключові задачі для розробки системи підтримки аналізу кредитних ризиків, яка має автоматизувати збереження, обробку та моделювання даних. Розроблене рішення передбачає використання сучасних підходів до централізації даних, скорочення ручної роботи й інтеграції з існуючими процесами банку.

Описано структуру системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку з використанням структурної схеми. Визначено основні елементи системи, та їх взаємодію між собою та з зовнішніми елементами.

3 ОГЛЯД ІСНУЮЧИХ РЕАЛІЗАЦІЙ

Існує широкий спектр рішень для аналітики ризиків у банках України, що відрізняються функціональними можливостями та вартістю. У цьому розділі буде розглянуто представлений на сучасному ринку інструментарій, який має необхідний функціонал для аналізу кредитних портфельних ризиків банку.

Можна виділити три основних категорії:

– безкоштовні BI-системи;

– Power BI;

– рішення на замовлення від українських компаній (CS, UNITY-BARS, Lime Systems).

Безкоштовні BI-системи підходять для банків з обмеженими бюджетами, але з наявністю потреб в аналізі даних. Такі системи забезпечують базові можливості візуалізації та аналітики.

Power BI – це універсальне рішення, яке логічно доповнює офісний пакет Microsoft Office, який використовується багатьма банками України. Power BI пропонує значну гнучкість і функціональність для середніх та великих банків, маючи при цьому доступну вартість у порівнянні з рішеннями, виготовленими на замовлення.

Рішення на замовлення від українських компаній – це спеціалізовані системи, які враховують особливості банківської діяльності в Україні та пропонують широкий спектр можливостей для інтеграції з внутрішніми системами, проте мають значно вищу цінову категорію

Огляд цих рішень дозволить порівняти їх за вартістю, можливостями та адаптивністю до потреб банків різного масштабу.

3.1 Огляд безкоштовних BI-систем

Безкоштовні BI-системи пропонують базовий, але досить функціональний набір інструментів для аналізу даних, що робить їх привабливими для банків з

обмеженими фінансовими можливостями або для невеликих проєктів. Хоча ці системи мають обмежену кількість функцій у порівнянні з комерційними аналогами, їх можливості можна ефективно використовувати для аналізу портфельних кредитних ризиків банку. Розглянемо основний функціонал таких систем і їхнє застосування.

Основні функції безкоштовних BI-систем:

– збір та обробка даних;

– дашборди та візуалізація даних;

– фільтрація та сегментація даних;

– моделювання сценаріїв;

– звітування та обмін інформацією.

Безкоштовні BI-системи дозволяють з'єднуватися з різними джерелами даних, такими як бази даних, Excel-таблиці, текстові файли та інші. Це надає змогу банку зібрати інформацію про кредитні портфелі, фінансові показники та історію платежів клієнтів.

Основною перевагою таких систем є можливість створювати інтерактивні дашборди. Вони дозволяють ризик-аналітикам візуалізувати складні дані за допомогою графіків, діаграм і таблиць, що спрощує аналіз показників ризику, таких як рівень дефолту або зміни у кредитних рейтингах.

У безкоштовних BI-системах зазвичай доступні інструменти для сегментації та фільтрації даних. Це допомагає аналізувати окремі групи позичальників або кредитів за різними критеріями – тип кредиту, строки погашення, регіони або галузі. Наприклад, можна легко сегментувати кредити за рівнем ризику і відстежувати найнебезпечніші сегменти.

Хоча безкоштовні BI-системи обмежені в аналітичних можливостях порівняно з преміальними версіями, вони часто підтримують прості інструменти для моделювання сценаріїв. Це дозволяє оцінювати потенційний вплив різних ринкових умов на кредитний портфель банку (наприклад, аналіз сценаріїв за умов зниження або підвищення відсоткових ставок).

Системи зазвичай дозволяють створювати регулярні звіти з результатами аналізу, які можуть бути використані для звітування перед керівництвом банку або регуляторними органами.

Застосування до аналізу портфельних кредитних ризиків:

– оцінка ризику дефолту;

– сегментація кредитного портфеля;

– прогнозування ризиків;

– моніторинг показників ефективності (KPI);

– аналіз ринкових трендів.

Збираючи дані про кредитну історію позичальників і поточні фінансові показники, система може допомогти ідентифікувати клієнтів з високим ризиком дефолту. Наприклад, за допомогою дашбордів можна відслідковувати відсоток прострочених платежів у кожному сегменті клієнтів.

BI-системи дозволяють розбивати кредитний портфель на групи за рівнем ризику, типом кредиту (іпотечний, споживчий, корпоративний), регіонами або галузями економіки. Це дає змогу банку визначити найбільш ризикові області для подальшої роботи з цими клієнтами.

Використовуючи моделі сценаріїв, банк може оцінити, як зміна економічних факторів (інфляція, зміна процентних ставок) вплине на кредитний портфель. Це допоможе спрогнозувати, як певні зміни вплинуть на ймовірність дефолтів або реструктуризацій кредитів.

Безкоштовні BI-системи дозволяють постійно відстежувати ключові показники кредитних ризиків, такі як відсоток прострочень, рівень NPL (Non-performing loans), та ефективність роботи відділів ризик-менеджменту. Регулярний моніторинг допомагає своєчасно реагувати на негативні зміни.

Дані про ринкові умови можуть бути інтегровані в систему для оцінки загроз і ризиків, що виникають через макроекономічні фактори, такі як коливання ринкових цін або зміни в законодавстві.

Популярні безкоштовні BI-системи:

– Looker Studio: підтримує візуалізацію даних і створення дашбордів, з можливістю інтеграції з Google Sheets, базами даних та іншими джерелами;

– Metabase: простий у використанні інструмент, який забезпечує базовий аналіз і візуалізацію даних;

– Tableau Public: безкоштовна версія популярної платформи для візуалізації, що дозволяє створювати дашборди та інтерактивні звіти, хоча обмежена в можливостях зберігання та інтеграції з деякими джерелами даних.

Переваги:

– низька вартість або безкоштовність;

– доступність для банків будь-яких розмірів;

– простота використання і швидке налаштування.

Недоліки:

– обмежені можливості налаштувань та інтеграції;

– менший вибір аналітичних інструментів;

– можливі обмеження щодо кількості даних або доступних функцій у безкоштовних версіях;

– потреба в джерелі упорядкованих даних;

– відсутність дозволу на збереження результатів роботи у локальному сховищі у деяких безкоштовних BI-систем.

Looker Studio прив’язує активи до акаунтів користувачів, що може призвести до їх втрати, якщо ця людина покине компанію [3]. Tableau Public вимагає поширення виконаної роботи у публічний доступ [4], що є критично небажаним для банків.

Таким чином, безкоштовні BI-системи можуть стати стартовою точкою для банків, які хочуть покращити аналіз кредитних ризиків і візуалізувати дані без значних фінансових вкладень. Однак, їх варто розглядати як тимчасове або базове рішення для банків з менш складними процесами. У великій кількості випадків, зрештою банкам доведеться переходити на платні версії BI-систем. Також банкам все одно доведеться витрачати кошти та часові ресурси на навчання користувачів, що теж варто враховувати при обрахунку вартості інтеграції.

3.2 Огляд системи Power BI

Power BI є потужним інструментом для бізнес-аналітики, який надає широкі можливості для візуалізації, аналізу і моніторингу даних. Це платна система, яка органічно доповнює офісний пакет Microsoft Office, що робить її популярним вибором для банків, особливо тих, що вже використовують Microsoft Excel або інші інструменти Microsoft. Завдяки знайомому інтерфейсу та інтеграції з іншими продуктами компанії, Power BI стає зручним і функціональним рішенням для аналізу кредитних ризиків.

Основний функціонал Power BI:

– збір та інтеграція даних;

– розширені можливості візуалізації;

– фільтрація та сегментація даних;

– моделювання та аналітика на основі сценаріїв;

– вбудована підтримка AI та машинного навчання;

– інтеграція з іншими Microsoft продуктами;

– спільний доступ та автоматизація звітності.

Power BI може підключатися до різноманітних джерел даних, включаючи SQL Server, Excel, Access, сторонні бази даних, веб-сервіси і навіть платформи типу Google Analytics. Це дозволяє легко інтегрувати інформацію з різних підрозділів банку (наприклад, з відділів кредитування або фінансової аналітики) для централізованого аналізу. У контексті портфельного аналізу можна збирати дані про всі активні кредити, фінансові показники клієнтів та їхню історію платежів.

Power BI пропонує широкий набір інструментів для створення інтерактивних дашбордів і звітів. Він підтримує різноманітні типи графіків, діаграм, теплових карт і картографічних візуалізацій. Це дозволяє ризик-аналітикам візуалізувати складні дані, що допомагає швидше оцінювати фінансові ризики. Наприклад, можна побудувати дашборд, який показує рівень дефолту в різних кредитних портфелях або відображає географічний розподіл ризиків.

За допомогою вбудованих інструментів фільтрації та сегментації даних Power BI дозволяє легко аналізувати окремі категорії кредитів або групи позичальників. Наприклад, можна швидко створити звіт, що відображає ризикові кредити за галузями, строками кредитування або рівнями ризику, що дає змогу ризик-менеджерам приймати більш обґрунтовані рішення.

Power BI підтримує вбудовані можливості моделювання даних, що дозволяє аналізувати вплив різних факторів на кредитний портфель. Використовуючи функціонал DAX (Data Analysis Expressions), можна створювати складні обчислювальні моделі, прогнозувати ймовірність дефолтів у різних ринкових умовах, наприклад, при зміні процентних ставок або макроекономічної ситуації.

Power BI легко інтегрується з іншими продуктами Microsoft, такими як Excel, Azure, SQL Server та інші. Це дозволяє банкам ефективно використовувати вже існуючу IT-інфраструктуру. Наприклад, можна автоматизувати процес передачі даних з банківських систем до Power BI, створювати звіти без необхідності імпортувати їх вручну.

Однією з найсильніших сторін Power BI є його інтеграція з інструментами штучного інтелекту та машинного навчання, що дозволяє будувати прогностичні моделі і автоматизувати аналіз даних. Це може бути корисно для оцінки портфельних ризиків, оскільки допомагає автоматично визначати патерни поведінки клієнтів, які можуть вказувати на підвищений рівень ризику.

Power BI дозволяє створювати спільні звіти, якими можуть користуватися кілька відділів банку, і забезпечує автоматичне оновлення даних у реальному часі. Це полегшує обмін інформацією між ризик-аналітиками та керівництвом банку, оскільки звіти можуть бути автоматично розіслані на регулярній основі, наприклад, щотижня або щомісяця.

Застосування Power BI до аналізу портфельних кредитних ризиків:

– оцінка дефолтних ризиків;

– сегментація портфеля за рівнем ризику;

– моніторинг ключових показників ефективності (KPI);

– прогнозування та моделювання;

– інтеграція з іншими системами ризик-менеджменту.

Power BI дозволяє побудувати звіти і дашборди для моніторингу ризику дефолту позичальників. Аналітики можуть легко відстежувати важливі показники, такі як кількість прострочених платежів, кредитний рейтинг клієнтів або зміни в поведінці позичальників, і прогнозувати ймовірність дефолту в майбутньому.

Використовуючи фільтрацію та сегментацію даних, Power BI допомагає банкам розбивати кредитний портфель на групи за рівнем ризику. Наприклад, можна ідентифікувати найбільш ризикові сегменти (малі підприємства, споживчі кредити) і проводити глибокий аналіз щодо того, які фактори підвищують ризики у кожній категорії.

Power BI дозволяє створювати і моніторити KPI для управління ризиками. Наприклад, можна регулярно відстежувати рівень NPL (кредити, що не приносять доходу), коефіцієнт покриття кредитів резервами або інші показники, що допомагають ризик-менеджерам реагувати на проблеми в реальному часі.

За допомогою інструментів моделювання, доступних у Power BI, можна прогнозувати потенційні збитки у разі зміни макроекономічних умов. Наприклад, аналітики можуть створювати сценарії "що, якщо", моделюючи вплив підвищення процентних ставок на рівень дефолтів або збільшення прострочених кредитів.

Завдяки інтеграції з SQL Server або Azure, дані з банківських систем управління кредитними ризиками можуть автоматично завантажуватися до Power BI, що дозволяє ризик-аналітикам працювати з актуальними даними без необхідності проводити ручний імпорт. Це значно підвищує ефективність аналізу та дозволяє банку оперативно реагувати на зміни в кредитному портфелі.

Power BI має певні переваги та недоліки.

Переваги:

– широкі можливості візуалізації та інтеграції даних;

– інтуїтивно зрозумілий інтерфейс, схожий на інші продукти Microsoft;

– інтеграція з офісними додатками (Excel, Outlook);

– потужні інструменти для аналітики та прогнозування;

– можливість масштабування та підтримка великих обсягів даних.

Недоліки:

– Power BI Premium, який скоріш за все буде потрібний, враховуючи об’єми даних, які обробляють бізнес аналітики, коштує 20$ на місяць за кожного користувача;

– потреба в початковому налаштуванні і належному управлінні для максимальної ефективності;

– потреба в джерелі упорядкованих даних.

Таким чином, Power BI є потужним рішенням для банків, що прагнуть покращити аналіз портфельних кредитних ризиків, використовуючи знайомий інтерфейс та наявні можливості інтеграції з вже існуючими системами Microsoft. Це інструмент, що може значно підвищити ефективність ризик-менеджменту та допомогти банкам оперативно реагувати на зміни в кредитних портфелях.

3.3 Огляд рішень на замовлення від українських компаній

Рішення на замовлення від українських компаній, таких як CS, UNITY-BARS і Lime Systems, є спеціалізованими BI-системами, які розробляються під конкретні потреби банку. Такі рішення пропонують глибоку інтеграцію з існуючими системами обміну даними та процесами банку, що дозволяє використовувати всі наявні дані максимально ефективно. Вони надають гнучкість у налаштуванні під специфічні вимоги банку, але це також означає високі витрати на впровадження та підтримку.

Основний функціонал рішень на замовлення:

– повна інтеграція з банківськими системами;

– гнучке налаштування під вимоги банку;

– індивідуальна візуалізація даних та аналітичні інструменти;

– розширені можливості аналітики і прогнозування;

– підтримка великого обсягу даних та масштабування;

– підтримка різних стандартів безпеки та конфіденційності.

Однією з головних переваг рішень на замовлення є можливість інтеграції з будь-якими існуючими системами банку, такими як АБС (Автоматизована Банківська Система), CRM, системи управління кредитними ризиками та бази даних. Наприклад, рішення від CS може бути інтегроване з системами управління кредитним портфелем, що дозволить використовувати всі наявні дані для створення індивідуальних аналітичних звітів та дашбордів.

На відміну від готових продуктів, кастомні рішення створюються під конкретні завдання банку. Це означає, що система може бути повністю налаштована під специфіку аналізу ризиків банку, включаючи структуру портфеля, фінансові індикатори, специфічні сценарії аналізу. Наприклад, рішення від UNITY-BARS дозволяє банку налаштовувати алгоритми обчислення ризиків під конкретні методології або стандарти, такі як стандарти BASEL

Такі рішення надають можливість створення унікальних дашбордів і звітів, які відповідають вимогам аналітиків та керівництва банку. Рішення від Lime Systems може надавати динамічні звіти, що відображають всі важливі показники в режимі реального часу, а також інтерактивні інструменти для детального аналізу кредитного портфеля. Це допомагає керівництву швидко оцінювати ризики та приймати обґрунтовані рішення.

Рішення на замовлення можуть включати складні аналітичні моделі та алгоритми прогнозування, розроблені спеціально для потреб банку. Вони дозволяють прогнозувати ризики на основі історичних даних і сучасних тенденцій. Наприклад, CS пропонує модулі, які використовують машинне навчання та алгоритми штучного інтелекту для прогнозування кредитних ризиків і ймовірності дефолтів, що дозволяє банку діяти проактивно.

Рішення на замовлення розробляються з урахуванням потреб великих фінансових установ, які працюють з великими обсягами інформації. Ці рішення дозволяють ефективно обробляти великі дані, що циркулюють у банку, і при цьому забезпечувати швидкість та ефективність аналітики. Наприклад, системи від UNITY-BARS можуть працювати з великими банками, де кількість транзакцій і клієнтів є значною, дозволяючи обробляти ці дані без втрати продуктивності.

Важливою перевагою таких рішень є можливість налаштування систем під конкретні вимоги до безпеки. Для банків критично важливо, щоб всі дані залишалися конфіденційними і захищеними. Компанії, такі як CS та Lime Systems, надають рішення з повною підтримкою стандартів інформаційної безпеки (наприклад, ISO/IEC 27001), забезпечуючи надійний захист даних банку від зовнішніх загроз.

Застосування рішень на замовлення до аналізу портфельних кредитних ризиків:

– аналіз структури кредитного портфеля;

– побудова прогнозів і сценаріїв;

– моніторинг і управління кредитними ризиками в реальному часі;

– автоматизація звітності для регуляторів.

Рішення на замовлення дозволяють глибоко аналізувати структуру кредитного портфеля банку. Наприклад, аналітики можуть відстежувати рівень концентрації ризиків за різними секторами економіки, типами кредитів (іпотечні, споживчі, корпоративні кредити) або географічними регіонами. Це дозволяє виявляти слабкі місця портфеля та вчасно реагувати на ризики.

Використовуючи історичні дані і прогностичні моделі, кастомні рішення можуть допомогти банкам оцінити можливі наслідки різних економічних сценаріїв. Наприклад, можна моделювати вплив підвищення процентних ставок на загальний рівень дефолтів, що дасть змогу банку завчасно вжити заходів для зменшення ризиків.

Завдяки інтеграції з банківськими системами, рішення на замовлення дозволяють створювати дашборди для моніторингу ключових показників ризиків у реальному часі. Це дозволяє керівництву банку мати доступ до актуальної інформації щодо ризиків і приймати рішення на основі останніх даних, а не чекати завершення обробки інформації.

Спеціалізовані рішення можуть автоматизувати підготовку звітів для регуляторів (НБУ, BASEL) щодо ризиків і кредитного портфеля. Наприклад, рішення від CS може автоматично генерувати звіти щодо рівня NPL (кредити, що

не приносять доходу) або резервування коштів, що допомагає банку відповідати вимогам регуляторів без додаткових зусиль.

Рішення на замовлення мають свої переваги та недоліки.

Переваги:

– глибока інтеграція з існуючими банківськими системами;

– максимальна гнучкість у налаштуванні під специфічні потреби банку;

– потужні інструменти прогнозування та сценарного аналізу;

– підтримка стандартів безпеки та великих обсягів даних.

Недоліки:

– критично висока вартість впровадження та підтримки;

– потреба в тривалому налаштуванні та адаптації;

– залежність від постачальника рішення в питаннях підтримки та оновлення.

Інформація про вартість інтеграції такої системи варіюється і не є публічною. Відомо, що річна технічна підтримка рішення на замовлення для НБУ від UNITY-BARS виходить в 5 млн гривень станом на 2023 рік [5].

Рішення на замовлення від CS, UNITY-BARS та Lime Systems є ефективним інструментом для аналізу портфельних кредитних ризиків банків. Вони дозволяють банкам максимально адаптувати аналітику під свої потреби, інтегруватися з наявними системами та забезпечувати високу якість управління ризиками. Однак, висока вартість таких рішень робить їх доступними переважно для найбільших банківських установ України, які можуть дозволити собі масштабні інвестиції в технології.

Висновки до розділу 3

У цьому розділі проаналізовано сучасні рішення для аналітики ризиків у банківській сфері, зокрема безкоштовні BI-системи, Power BI і спеціалізовані рішення на замовлення. Результати огляду демонструють різноманіття доступних інструментів, які задовольняють потреби банків різного масштабу та бюджету,

але суттєво відрізняються за функціоналом, рівнем інтеграції та вартістю впровадження.

Безкоштовні BI-системи, такі як Looker Studio або Tableau Public, забезпечують базову аналітику, візуалізацію даних і створення звітів, але мають обмеження щодо функціоналу та інтеграції. Power BI надає розширені можливості, включаючи інтеграцію з екосистемою Microsoft, моделювання даних і автоматизацію звітності, що робить його популярним вибором серед банків середнього рівня. Рішення на замовлення, представлені такими компаніями, як CS, UNITY-BARS і Lime Systems, пропонують максимально адаптивні аналітичні інструменти з інтеграцією до внутрішніх банківських систем, але мають високу вартість і потребують тривалого налаштування.

Розглянуто плюси та мінуси кожного з наявних рішень для визначення характеристик, які могли б вирізнити систему підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку серед конкурентів.

4 ПРОЄКТУВАННЯ СИСТЕМИ

4.1 Сценарії використання

Для планування функціоналу системи було побудовану діаграму сценаріїв використання, яку представлено у додатку Ж.

Діаграма сценаріїв використання є ключовим компонентом у проєктуванні програмного забезпечення, що допомагає наочно відобразити основні взаємодії між користувачами та системою. У випадку системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфеля банку, діаграма сценаріїв використання розроблена для представлення всіх можливих дій, які можуть виконувати різні категорії користувачів, включаючи гостей та авторизованих користувачів.

Гість у системі має мінімальні права доступу, однак йому надається можливість зареєструватися та створити обліковий запис. Якщо гість ще не має акаунта, він може перейти до процесу реєстрації, де вводить свої персональні дані, такі як ім'я, електронна пошта та пароль. Після створення акаунта гість отримує статус зареєстрованого користувача. Крім того, гість може авторизуватися за допомогою свого облікового запису, якщо вже зареєстрований, і таким чином отримати доступ до функцій системи, доступних лише авторизованим користувачам.

Після авторизації користувач отримує можливість взаємодіяти з системою на глибшому рівні.

Однією з основних функцій є створення конфігурацій імпорту файлів. Ця функція дозволяє користувачам налаштовувати процеси імпорту даних з різних джерел. Наприклад, користувач може створити конфігурацію імпорту файлів з файлового серверу, де зберігаються робочі документи, або налаштувати імпорт даних через веб-API. У конфігураціях імпорту користувач має змогу визначити параметри обробки даних, такі як формат файлів, правила обробки даних та специфікації для їх інтеграції до сховища.

Окрім створення нових конфігурацій, користувач має можливість редагувати вже існуючі конфігурації. Це особливо корисно, якщо параметри імпорту потребують оновлення через зміну вимог або умов роботи. Система дозволяє користувачеві легко вносити зміни у налаштування конфігурацій, що підвищує їх гнучкість та адаптивність.

Щоб спростити роботу з великою кількістю конфігурацій, система надає функції сортування та фільтрації. Користувач може відфільтрувати конфігурації імпорту файлів за кожним з основних параметрів, наприклад, джерелом даних або типом файлів. Сортування також полегшує навігацію та управління, дозволяючи впорядкувати конфігурації у потрібному порядку.

Окрім цього, користувач має змогу обирати одну або кілька конфігурацій для подальшого використання. Наприклад, обрані конфігурації можуть бути видалені, якщо вони більше не потрібні, або запущені для імпорту даних. Цей підхід забезпечує зручність у роботі з системою та дозволяє мінімізувати ризик помилок під час імпорту.

Запущені імпорти починають низку автоматичних процесів у системі. Бек-енд частина зчитує та обробляє файли. Далі створюється необхідна таблиця у базі даних у схемі source_data. Таблиця заповнюється записами з файлів. Заповнення таблиці викликає тригери, які можуть запустити процедуру, що заповнить значення таблиць структурованої схеми analytics. Таблиці цієї схеми містять нормалізовані дані, які готові до використання у візуалізації за допомогою інтегрованої системи Metabase.

Система підтримує вивантаження даних зі сховища та сформованих дашбордів у вигляді файлів та через API-запити. Якщо користувач має окреме середовище для виконання аналізу, якому він надає перевагу, наприклад Excel, то нормалізовані дані можна легко імпортувати в це середовище.

Інтеграція з BI-системою Metabase є ще однією важливою функцією системи. Вона надає користувачам можливість будувати дашборди з нормалізованих даних, які були імпортовані та структуровані для зберігання у сховищі. Завдяки Metabase користувачі також можуть виконувати складні запити

до бази даних та отримувати звіти у зручному форматі. Це значно полегшує процес формування звітності, роблячи його швидким та інтуїтивно зрозумілим.

Після успішного імпорту файлів користувач може переглядати імпортовані файли у файловому архіві. Файловий архів виступає централізованим сховищем для всіх імпортованих даних, де кожен файл має метадані, такі як дата завантаження, джерело, відповідна конфігурація та статус обробки. Це допомагає користувачам швидко знаходити потрібну інформацію та контролювати процеси, пов'язані з обробкою даних.

Система забезпечує повний цикл роботи з даними – від їх імпорту до представлення. Усі дії користувача відображаються на діаграмі сценаріїв використання як взаємодії між актором (користувачем) та функціями системи. Головна ідея полягає у створенні безшовного робочого процесу, де користувач може працювати з даними у єдиному середовищі, без необхідності використовувати зовнішні інструменти.

4.2 Документообіг системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку

Було створено схему документообігу, яка об'єднує всі ключові джерела даних та елементи системи. Її можна побачити на рисунку 4.1.

Організований таким чином документообіг забезпечує ефективну взаємодію між компонентами та інтеграцію всіх функцій, необхідних для комплексного аналізу ризиків.

Основою системи є файловий сервер, який виконує роль централізованого сховища робочих файлів. Його використання дозволяє забезпечити зручний і безпечний доступ до даних для всіх користувачів. Файловий сервер слугує єдиним місцем збереження інформації, що усуває проблему дублювання даних і сприяє підвищенню рівня організації роботи. Завдяки цьому співробітники можуть швидко знаходити необхідні файли, а ризик втрати інформації знижується

до мінімуму. Крім того, інтеграція файлового сервера з іншими компонентами системи забезпечує автоматизацію імпорту даних.

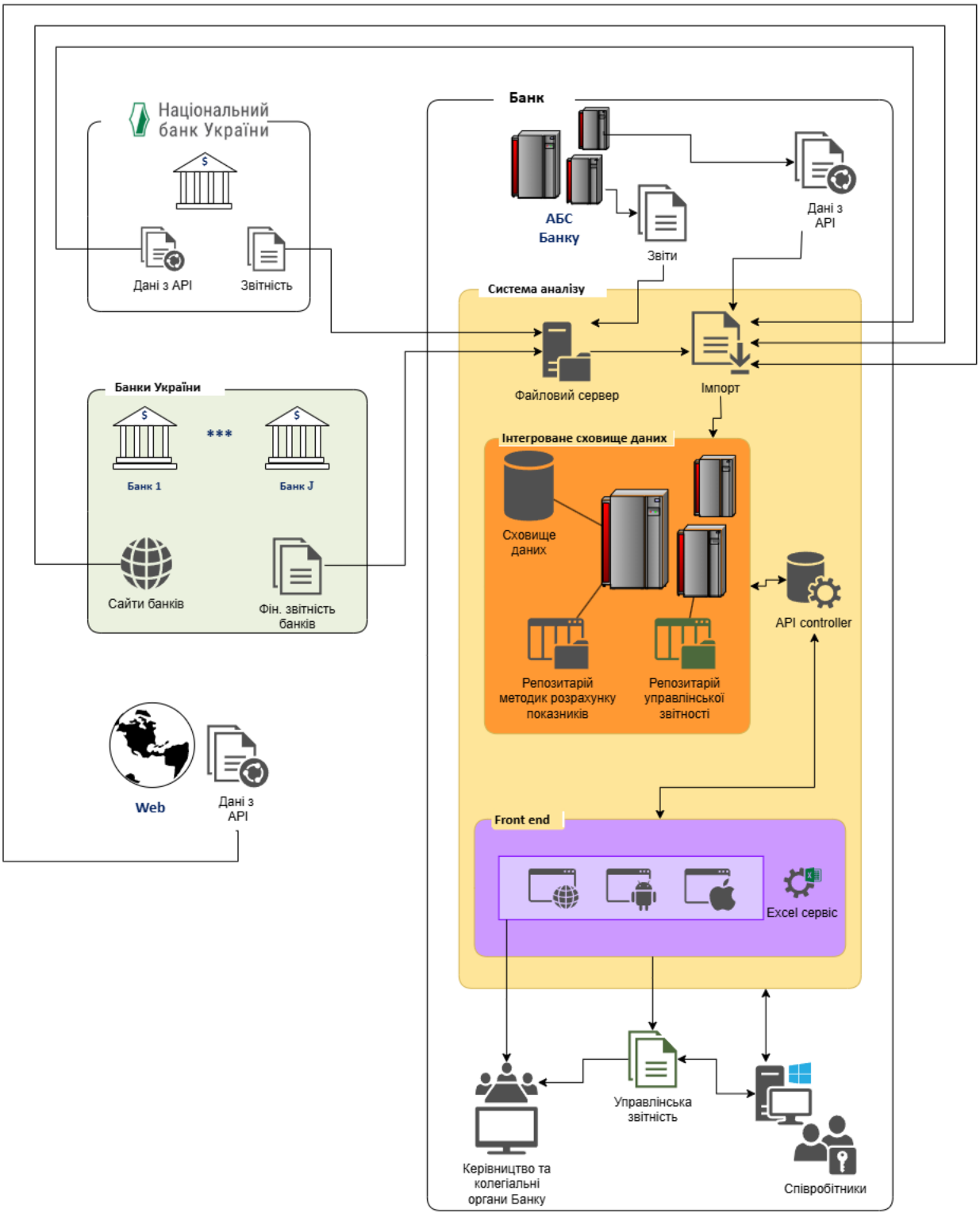

Рисунок 4.1 – Схема документообігу системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку

Дані можуть бути імпортовані у сховище з файлового сервера або через веб-API. Це дозволяє автоматизувати передачу інформації, зменшуючи необхідність ручної обробки. Таким чином, система підтримує усі головні джерела даних: Національний банк України, комерційні банки України, АБС банку та дані з інших веб-API, що зменшує час на підготовку інформації для аналізу, що є критичним фактором в умовах банківського середовища.

Для збереження і нормалізації великих обсягів даних у системі використовується сховище даних, яке є основним елементом для накопичення і управління інформацією. Сховище працює на основі сучасних технологій баз даних, що дозволяє швидко обробляти великі масиви інформації і забезпечує високу швидкість доступу до даних. Усі імпортовані файли піддаються структуризації, що спрощує подальший аналіз і візуалізацію.

Щоб забезпечити взаємодію між сховищем даних і фронт-ендом системи, було розроблено серверний бек-енд, який керується API-контролером. Цей елемент відповідає за обробку запитів від користувачів, їхню передачу до сховища, виконання необхідних операцій з даними і повернення результатів. Архітектура серверного бек-енду побудована так, щоб забезпечити масштабованість системи, високу продуктивність і можливість адаптації до змін у процесах аналізу.

Однією з ключових функцій вебзастосунку є можливість імпорту інформації з робочих файлів у сховище даних. Користувачі мають змогу виконувати ці операції через єдиний інтерфейс, що значно спрощує робочі процеси.

Крім того, система дозволяє виконувати певний аналіз даних безпосередньо у вебзастосунку, забезпечуючи інтеграцію всіх необхідних інструментів в одній платформі. Одною з основних переваг системи є інтеграція BI-системи для візуалізації даних. Користувачі системи, а саме ризик-аналітики, можуть сформувати звітність у вигляді сучасних дашбордів з інтерактивними фільтрами та графіками прямо в системі.

Важливим аспектом розробленої системи є її інтеграція з існуючими процесами банку. Вона дозволяє не лише проводити аналіз ризиків, але й

зберігати та обробляти дані централізовано. Це усуває недоліки традиційних підходів, таких як використання MS Excel для аналізу даних, що часто призводить до втрати інформації та зниження ефективності роботи аналітиків.

Загалом, система забезпечує високу продуктивність, простоту використання та масштабованість. Це дозволяє банкам адаптуватися до змін економічних умов і забезпечувати стабільність своєї діяльності, знижуючи фінансові ризики та підвищуючи довіру клієнтів і партнерів.

4.3 Структура системи

Для проєктування необхідних модулів системи та планування їх взаємозв’язків було створено структурну схему, яка наочно демонструє логіку функціонування всієї системи. Ця схема наведена в додатку А. Основним завданням під час проєктування було забезпечення взаємодії та логічної послідовності дій модулів, яка враховує потреби користувачів і забезпечує зручний доступ до функціоналу.

Користувач взаємодіє з клієнтською частиною системи, яка об’єднує функціонал BI-системи Metabase та частину, що забезпечує підтримку імпорту файлів. Важливою особливістю реалізації стало безшовне поєднання графічних інтерфейсів цих підпрограм, що дозволило створити цілісне та зручне середовище для роботи. Такий підхід сприяє інтуїтивно зрозумілій взаємодії користувача з різними компонентами системи та підвищує її ефективність.

Архітектурно, модулі, що відповідають за взаємодію із клієнтською частиною, реалізовано окремо для кожної складової системи. Це рішення забезпечило гнучкість у реалізації та спростить імплементацію оновлень Metabase в майбутньому. Бізнес-логіка Metabase залишалася незмінною. Цій BI-системі був наданий прямий доступ до двох ключових схем сховища даних: analytics та source_data. Такий підхід дозволив інтегрувати як нормалізовані, так і не нормалізовані дані у візуалізації, дашборди та інші засоби аналітики. Відсутність необхідності змінювати внутрішню логіку Metabase сприяла збереженню

стабільності роботи BI-системи, мінімізувала ризики помилок і скоротила час на інтеграцію.

Модуль взаємодії з клієнтською частиною для підтримки імпорту файлів повертає користувачу збережені конфігурації імпорту, та отримує від нього команди. Далі вони передаються або до модуля конфігурацій імпорту файлів, або до модуля зчитування файлів. Модуль конфігурацій імпорту файлів виконує необхідні дії для отримання поточних даних з бази та внесення до неї змін.

Модуль конфігурацій імпорту файлів виконує читання, редагування та видалення конфігурацій імпорту, збережених у схемі analytics сховища даних, базуючись на потребах клієнтської частини та командах користувача.

Процес імпорту даних починається з команди до модуля зчитування файлів. У залежності від джерела даних, система працює з веб-ресурсами або локальними файлами. Якщо файл розташований на веб-ресурсі, модуль зчитує його через веб-API, тоді як для локальних файлів використовується робоча директорія файлового серверу. Цей підхід забезпечує універсальність роботи з даними незалежно від їхнього джерела.

Після зчитування файл передається до модуля архівації, який забезпечує його збереження у файловому архіві. Файловий архів також знаходиться на файловому сервері, що спрощує управління даними.

Наступним етапом є обробка даних у модулі парсингу. Цей модуль трансформує зчитану інформацію у формат, придатний для подальшого аналізу та зберігання у базі даних. Задача парсингу полягає в тому, щоб стандартизувати дані та забезпечити їхню готовність до інтеграції в структуру сховища.

Після підготовки даних модуль імпорту створює необхідні SQL-запити та завантажує інформацію до схеми source_data. Ця схема відповідає за зберігання сирих даних у їхньому первісному вигляді. Крім того, імпорт може активувати тригери реструктуризації даних. Ці тригери запускають збережені процедури, які автоматично нормалізують дані та переносять їх до структурованих таблиць у схемі analytics. Такий підхід дозволяє автоматизувати процес підготовки даних для аналітики та значно скорочує час, необхідний для аналізу.

Таким чином, створена система забезпечує повний цикл роботи з даними, починаючи від їх зчитування та обробки до інтеграції у сховище даних і подальшого використання в аналітиці. Модульна архітектура сприяє простоті підтримки, оновлення та розширення функціональності, що робить систему ефективним інструментом для банківських аналітиків.

4.4 Ключові процеси в системі

Діаграма послідовності, наведена в додатку Д, ілюструє основний процес взаємодії користувача з фронт-ендом системи, а також обмін даними між елементами системи під час виконання ключових операцій. Ця діаграма є візуальним представленням динамічної поведінки системи під час обробки запитів, які ініціює користувач, що є необхідним для розуміння її роботи.

Коли користувач виявляє нові файли або веб-API ресурси, які містять дані, що ще не були імпортовані в систему, він переходить до відповідної вкладки імпорту в інтерфейсі користувача. Ця вкладка служить стартовою точкою для ініціювання процесу додавання нових джерел даних до системи. На цьому етапі користувач може створити нову конфігурацію імпорту, яка представлена у вигляді рядка таблиці імпорту, яку потім можна бути використати для імпорту файлів з такою самою структурою.

Під час створення конфігурації система реалізує механізм автоматичного збереження. Це означає, що будь-які зміни, внесені користувачем, негайно зберігаються до бази даних, що мінімізує ризик втрати даних у разі переривання роботи. Збережені конфігурації стають доступними для подальшого використання та редагування. Фронт-енд через відповідний API-запит передає параметри нової конфігурації на сервер, де вони обробляються і додаються до бази даних. Цей процес гарантує, що всі введені дані залишаються структурованими і готовими до використання.

Коли підготовка конфігурації завершується, користувач має можливість обрати необхідні конфігурації для запуску імпорту. Обрані конфігурації

ініціюють формування POST-запиту, який передає серверу всі параметри, потрібні для обробки. Сервер, отримавши цей запит, починає зчитувати дані з визначених джерел. Якщо джерело є файловим сервером, зчитуються файли у форматах, які підтримує система. У разі використання веб-API сервер виконує HTTP-запит, отримуючи JSON-відповідь з даними, що є необхідними для подальшої трансформації.

Зібрані дані проходять етап трансформації, під час якого вони перетворюються у формат, сумісний із вимогами бази даних. Це може включати зміну структури даних, очищення, а також додавання додаткових атрибутів, що полегшують аналітику. Оброблені дані далі розбиваються на пачки по 1000 рядків, після чого формується SQL-запит для вставки цих даних до сховища.

Коли дані завантажуються до бази даних, вони спочатку потрапляють до схеми source_data, яка виступає проміжним етапом зберігання необроблених даних. У цій схемі спрацьовує тригер бази даних, що автоматично виконує їх нормалізацію. Нормалізація включає реструктуризацію даних у відповідності до заздалегідь визначених правил, що готує їх до подальшого використання у аналітиці. Результатом цього процесу є заповнення новою інформацією необхідних таблиць у схемі analytics, що містить уже структуровані та підготовлені до аналізу дані.

Після завершення процесу імпорту та нормалізації даних користувач може перейти до використання інтегрованої BI-системи Metabase, яка забезпечує потужний функціонал для візуалізації та аналізу. Metabase, як частина загальної архітектури системи, дозволяє користувачам виконувати складні SQL-запити, створювати інтерактивні дашборди, аналізувати тренди та виявляти закономірності у даних. Важливою особливістю є можливість роботи з нормалізованими даними, які вже підготовлені системою, що знімає з користувачів необхідність у ручній підготовці інформації.

4.5 Процедура імпорту даних

Проаналізувавши структуру системи, вимоги до функціоналу та послідовності виконання дій у системі, було побудовано блок-схеми виконання імпорту, що представлено у додатках В та Г.

При натисканні на кнопку імпорту користувачем, на бек-енд надсилається POST-запит з обраною конфігурацією імпорту, яка містить усі необхідні для виконання алгоритму параметри.

Для всіх конфігурацій імпорту особливими є такі параметри:

– джерело інформації (Source);

– тип файлу (Data Format);

– тип джерела (Source Type);

– шлях з файловою маскою для імпортів з файлового сервера або посилання на веб-API ресурс (Path);

– шлях до директорії архіву на файловому сервері (Archive Path);

– назва таблиці у базі даних у схемі source_data (Table Name).

Для імпорту локальних файлів також можна заповнити параметр формату дати з назви файлу (Date Format).

JSON- та XML-файли не потребують додаткових параметрів, у той час як TXT має таких 2: рядок, з якого починаються табличні дані (First Row) та список індексів розбиття стовпців (Column Indexes).

Для CSV-файлів можна вказати який роздільник використовується у документі (Separator).

Excel має 5 додаткових параметрів. Sheet Number вказує які номери аркушів файлів обробляти, а First Row, Last Row, First Column, Last Column дозволяють вказати границі для таблиці, якщо потрібно обробити тільки частину даних.

В залежності від обраного типу джерела подальші дії системи відрізняються.

При імпорті з веб ресурсу спочатку ми трансформуємо шлях до файлового архіву в залежності від ОС та шляху до директорії. Це включає в себе підстановку

локального для серверу шляху до директорії та зміну роздільників при розбіжностях в операційних системах між сервером та пристроєм користувача.

Система зчитує відповідь на GET-запит, який вона надіслала на заданий у Path шлях і зберігає отриманий JSON як файл у архів.

При імпорті з файлового серверу користувач може надати файлову маску, щоб обробити декілька схожих за структурою файлів та зберегти їх у одній таблиці.

Спочатку шляхи до файлів та до архіву трансформуються на локальні шляхи для операційної системи сервера.

Далі виконується пошук файлів за заданою файловою маскою. Для кожного файлу визначається його кодування за допомогою штучного інтелекту, файл потім зчитується за цим кодуванням у текстовий рядок. Також локальні файли можуть мати дату своєї останньої зміни, що є корисною інформацією для зберігання у сховищі, тому якщо вдається її зчитати, вона передається далі по алгоритму. Зчитані файли переміщуються у архів.

Контролер визначає який тип файлу за параметром Data Format і викликає відповідну функцію обробки.

JSON-и зазвичай мають просту структуру даних. Файли цього типу представляють собою масив елементів, де кожний елемент має набір пар ключ-значення. Якщо уявити кожний елемент як окремий рядок таблиці, а ключ-значення як пару назва поля – значення поля для відповідного рядка, то можна легко конвертувати такий файл у необхідну для імпорту структуру List<Map<String, Object>>.

Однак, серед деяких файлів НБУ поширена і інша структура даних. Ці файли представляють собою двовимірний масив об’єктів, де зовнішній масив можна представити як рядки таблиці, а внутрішні масиви зберігають, окрім пар ключ – значення, ще й історію змін цих рядків таблиці. Кожний об'єкт внутрішнього масиву налічує 4 атрибути: date_b, date_e, attribute, value. Date_b та date_e – це дати початку та кінця актуальності даних для цього поля, attribute – це

назва поля, а value – це значення, яке було у цього поля для цього запису в проміжок часу від date_b до date_e.

Спочатку конвертуємо наш текстовий рядок у двовимірний список List<List<Map<String, Object>>>. Створимо List<Map<String, Object>> для збереження результату. Так як така структура є незвичайною, визначаємо множину унікальних атрибутів, пройшовшись по кожному зі значень attribute.

Для кожного внутрішнього масиву, створюємо необхідну кількість рядків (Map<String, Object>), щоб відобразити історію змін. Поля для цього використовуються інші, а саме D_OPEN, D_MODI та D_CLOSE. Ці поля використовуються в інших файлах НБУ, і ними простіше користуватися у базі даних. Зрозуміло, що рядки, у яких D_MODI має не порожнє значення, більше не є актуальними, а рядки, у яких D_CLOSE має не порожнє значення, були видалені остаточно.

XML-файли також є досить прямолінійними в їх обробці, проте неочікуваною проблемою стало використання КЕП-підписів до таких файлів. Це заважає автоматичному визначенню кодування файлу та парсингу даних.

Тому першим чином у алгоритмі перевизначається кодування файлу, якщо воно явно прописане в файлі. Далі створюємо List<Map<String, Object>>, очищаємо файл від можливої непотрібної інформації та зчитуємо дані з XML тегів. Більш детально цей процес розглянуто у підпункті 5.1.5 в методі parseXML.

CSV-файли є файлами, у яких один рядок – це один запис у таблиці. Тому можна просто розбити цей файл по рядкам, створити об’єкт, в який буде записано результат, та розділити кожний рядок за роздільником (параметр Separator). Перший рядок вважатимемо назвами стовпців і заповнимо List<Map<String, Object>> зчитаними значеннями.

Файли Excel можуть мати декілька аркушів, тому кожний з аркушів обробляється окремо в циклі.

Для кожного аркуша перевіряється коректність введення параметрів, визначаються номери останнього рядка та стовпця , якщо в параметрах не було обмежено границі таблиці.

Назви стовпців визначаються з першого рядка таблиці, зважаючи на можливість використання формул та поєднаних клітинок. Іноді після роботи з Excel-файлом, у ньому можуть залишитись проіндексовані клітинки, тому система це враховує і може не обробити стовпці з пустими назвами, які йдуть великою кількістю в ряд.

Обов’язково для SQL потрібно забезпечити унікальність назв стовпців. Для цього до тих стовпців, у яких дублюються назви, у назву додається порядковий номер.

Далі зчитуємо з кожної клітинки дані у відповідний рядок List<Map<String, Object>> і також додаємо назву аркуша, з якого взяті ці дані.

Для TXT-файлів один рядок також є одним записом у таблиці, проте розбиття на стовпці є менш очевидним. Система автоматично може визначити індекси розбиття стовпців за роздільником “|”, якщо їх не було вказано в параметрах.

Так як це TXT-файл, то для табличного зовнішнього вигляду та забезпечення стабільності індексації стовпців використовується велика кількість пробільних елементів. Всі пробільні елементи, крім пробілів між словами, видаляються і ми залишаємось тільки зі значеннями полів. Якщо поле повністю з пробілів, то таким чином воно виходить пустим і його слід замінити на NULL.

Зчитані дані з рядків додаються у новий List<Map<String, Object>>.

Не зважаючи на тип файлу, до рядків додається дата звіту, якщо вона наявна в назві файлу.

Виконується з’єднання з базою даних.

Якщо задана в параметрі Table Name таблиця ще не існує у схемі source_data, то потрібно її створити. Для цього визначаються назви стовпців, що беруться з ключів мапи Map<String, Object>. Для того, щоб визначити тип даних перевіряється якого з класів є значення Object у кожній мапі. Якщо у мапах трапляється розбіжність, то по замовченню створюється поле типу varchar.

Додаються системні поля bri_id (id запису), bri_time (єдиний час запису для файлу), bri_uid (id користувача), bri_f_time (дата з назви файлу). Надсилається запит на створення таблиці.

Далі зчитуються значення і формується відповідний SQL-запит на внесення нової інформації в таблицю. Нові записи вставляються пакетами по 1000 рядків.

Якщо імпортуються файли з файлового серверу, то процес повторюється для кожного файлу, що відповідає файловій масці.

Висновки до розділу 4

У цьому розділі було проаналізовано ключові аспекти розробленої системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфеля банку, її функціональні можливості та архітектуру. Зокрема, описано сценарії використання, структуру системи, документообіг і механізми взаємодії між компонентами.

Діаграма сценаріїв використання відобразила всі основні взаємодії користувачів із системою, включаючи процес реєстрації, створення конфігурацій імпорту, обробку даних та інтеграцію з BI-системою Metabase. Це дало змогу забезпечити чітке розуміння функціоналу системи та її ключових операцій. Особливу увагу приділено можливостям автоматизації процесів імпорту, редагування, сортування та запуску конфігурацій, що підвищує ефективність роботи користувачів.

Схема документообігу продемонструвала інтеграцію всіх ключових компонентів системи, включаючи файловий сервер, сховище даних, бекенд і BI-систему. Така організація забезпечує безперервний цикл роботи з даними: від їх імпорту з усіх головних джерел даних, а саме Національного банку України, комерційних банків України, АБС банку та веб-API, та зберігання до аналітичної обробки і формування звітів. Використання сучасних технологій дозволяє ефективно працювати з великими обсягами даних, зберігаючи їхню структурованість і доступність.

У структурній схемі було розглянуто модулі системи, їх взаємозв’язки між собою, з файловим сервером та зі сховищем даних. Також продемонстровано трансформацію даних у сховищі для їх нормалізації та збереження в окремій схемі.

Діаграма послідовності деталізувала динамічну поведінку системи під час виконання ключових операцій, таких як створення конфігурацій імпорту, зчитування даних та їх інтеграція до сховища. Це допомагає зрозуміти взаємодію між елементами системи і сприяє оптимізації її роботи.

У блок-схемах розглянуто алгоритм імпорту файлів, що детально пояснює різниці в підході до обробки файлів різних типів та форматів.

Загалом, розроблена система забезпечує комплексну підтримку аналізу портфельних ризиків, надаючи банкам можливість працювати з великими обсягами даних у зручному та ефективному середовищі. Вона сприяє підвищенню продуктивності ризик-аналітиків, зниженню фінансових ризиків і забезпечує стабільність банківських операцій.

5 ЗАСОБИ РОЗРОБКИ

Для створення системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку було вирішено використати:

– на бек-енд частині мову програмування Java та Spring Framework;

– на фронт-енд частині JavaScript, з бібліотекою Tabulator в основі.

В якості системи керування базами даних було обрано PostgreSQL.

5.1 Бек-енд частина

5.1.1 Мова програмування Java

Java – це мова програмування та обчислювальна платформа, вперше випущена компанією Sun Microsystems у 1995 році. Вона пройшла шлях від скромних початків до значної частини сучасного цифрового світу, забезпечуючи надійну платформу, на якій побудовано багато сервісів та додатків. Нові, інноваційні продукти та цифрові послуги, призначені для майбутнього, також продовжують покладатися на Java [6].

Java залишається однією з найпопулярніших мов для розробки як корпоративних, так і веб-додатків, мобільних програм та наукових обчислень. Її універсальність, надійність та широкі можливості інтеграції роблять її ідеальним вибором для розробки системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку.

5.1.2 Spring Framework

Spring спрощує створення корпоративних додатків на Java. Він надає все необхідне для використання мови Java в корпоративному середовищі, з підтримкою Groovy та Kotlin як альтернативних мов на JVM, а також з гнучкістю для створення багатьох видів архітектур залежно від потреб програми [7] [8].

Spring Framework – це комплексне рішення, яке підходить для розробки сучасних корпоративних додатків, а його гнучкість та інструменти роблять його ідеальним вибором для аналізу портфельних кредитних ризиків у банківському середовищі.

5.2 Фронт-енд частина

5.2.1 Вибір BI-системи для інтеграції

Business Intelligence (BI) системи є важливим компонентом сучасної банківської аналітики, що забезпечує ефективне управління даними та їх аналіз. Вони відіграють ключову роль в обробці великих обсягів інформації, дозволяючи банкам структурувати, зберігати та аналізувати різні типи даних, такі як фінансові, кредитні та операційні показники. BI-системи мають високу продуктивність у зборі даних з різних джерел, їх інтеграції та візуалізації, що дає змогу ризик-аналітикам швидко отримувати інсайти, аналізувати ризики та приймати обґрунтовані рішення на основі достовірної інформації.

Однією з головних переваг BI-систем є їх здатність автоматизувати процес аналізу кредитних портфелів, відстежувати зміни в показниках та надавати візуалізовані звіти. Це значно спрощує роботу з великими масивами даних та дозволяє зосередитися на аналізі тенденцій та прогнозуванні ризиків. Інтерактивні дашборди, які можуть бути налаштовані відповідно до потреб аналітиків, допомагають візуалізувати ключові показники ефективності (KPI), аналізувати тренди та розуміти складні залежності між даними. Завдяки цьому BI-системи стають важливим інструментом у процесі прийняття стратегічних рішень для управління кредитними ризиками.

Крім того, BI-системи дозволяють інтегруватися з існуючими базами даних та IT-інфраструктурою банку, забезпечуючи зручний доступ до актуальних даних для всіх рівнів менеджменту. Це допомагає уникнути дублювання інформації та зменшує ризики помилок при обробці даних. Таким чином, використання BI-систем забезпечує оптимізацію процесів аналітики та дозволяє банкам швидко

реагувати на зміни ринкових умов і ризиків, що робить їх ідеальними для використання на верхньому рівні нашої архітектури.

Для знаходження найкращого комплексу для системи автоматизації аналізу ризиків банку було розглянуто три безкоштовні BI-системи з відкритим вихідним кодом. Результати порівняльного аналізу представлено у таблиці 3.1.

Таблиця 3.1 – Порівняльний аналіз BI-систем з відкритим кодом

Критерії порівняння	Metabase	BIRT	KNOWAGE
Операційна система (ОС)	Windows, macOS, Linux, Docker	Windows, macOS, Linux	Windows, Linux, macOS, Docker
Підтримка баз даних	MySQL, PostgreSQL, SQLite, декілька інших	Різноманітні, включаючи MySQL, PostgreSQL, Oracle, SQLite	MySQL, PostgreSQL, Oracle, MS SQL Server, та інші
Підтримка виводу дашбордів	+	+	+
Ведення та контроль KPI	-	-	+
Аналітичні інструменти	Обмежені	Розширені	Розширені
Спільнота та підтримка	Активна	Активна	Активна
Візуальний інтерфейс	Сучасний	Застарілий	Сучасний

Однією з таких систем є аналітична та візуалізаційна платформа з відкритим вихідним кодом Metabase, яка швидко набирає популярність завдяки простоті використання та широкому функціоналу. Ця система була розроблена компанією Metabase, штаб-квартира якої розташована в Сан-Франциско, Каліфорнія, і

заснована у 2014 році Саміром Аль-Сакраном. Metabase надає можливість створювати інтерактивні дашборди, автоматизовані звіти та аналізувати великі обсяги даних без потреби у складному програмуванні, що робить її доступною для широкого кола користувачів.

Платформа інтегрується з різними типами баз даних, такими як MySQL, PostgreSQL, SQL Server та інші, дозволяючи користувачам отримувати дані безпосередньо з їхніх сховищ і візуалізувати їх у реальному часі. Вона підтримує створення різноманітних типів графіків і діаграм, що спрощує процес аналізу та допомагає швидко виявляти ключові інсайти. Завдяки можливості легкої інтеграції та гнучким параметрам налаштувань, Metabase є ідеальним інструментом для підприємств, які потребують якісного аналізу даних при мінімальних витратах.

Metabase надає зручний інтерфейс, який дозволяє користувачам швидко орієнтуватися в структурі даних та створювати візуалізації буквально за кілька кліків. Завдяки можливості спільного використання дашбордів між колегами, платформа сприяє ефективній командній роботі та швидкому обміну інформацією [9]. Тому Metabase є особливо привабливим рішенням для організацій, які потребують доступної, надійної та функціональної системи для аналітики даних, не витрачаючи значних ресурсів на ліцензії чи розробку кастомного програмного забезпечення.

Іншою системою для візуалізації даних є BIRT (Business Intelligence and Reporting Tools) – проєкт з відкритим вихідним кодом, який надає потужну технологічну платформу для створення інтерактивних звітів, дашбордів та графіків для візуалізації великих масивів даних. Розробка BIRT була ініційована в межах Eclipse Foundation за підтримки компанії Actuate, що активно співпрацювала з IBM та Innovent Solutions для вдосконалення функціоналу та забезпечення високої продуктивності. Проєкт розпочався як платформа для бізнес-аналітики та звітності, а сьогодні є одним із популярних інструментів для компаній, які прагнуть ефективно аналізувати свої дані.

BIRT відзначається розширюваністю та можливістю інтеграції з різними сховищами даних, включаючи SQL, NoSQL та інші бази. Він надає користувачам гнучкі інструменти для розробки звітів різної складності – від простих таблиць до складних інтерактивних дашбордів, які можна вбудовувати безпосередньо у веб-застосунки чи робочі процеси компанії. Завдяки високому рівню налаштування, BIRT може адаптуватися до конкретних потреб бізнесу, що робить його універсальним рішенням для організацій із різноманітними запитами на аналітику та звітність.

Проєкт BIRT також відомий своєю активною спільнотою розробників, яка постійно впроваджує нові функції та вдосконалює існуючі [10]. Це дозволяє підтримувати актуальність проєкту відповідно до нових вимог ринку. Платформа є особливо корисною для фінансових організацій та банків, які працюють з великими масивами даних та потребують надійних інструментів для формування звітів, що відповідають стандартам регуляторів і бізнес-вимогам.

Ще однією BI-системою, доступною на ринку, є KNOWAGE, яка пропонує широкі можливості для бізнес-аналітики з повністю відкритим вихідним кодом. Цей проєкт був раніше відомий як SpagoBI у версіях до 6.x, після чого він був перейменований та отримав значне функціональне оновлення. KNOWAGE підтримується європейською спільнотою відкритих кодів OW2, що забезпечує його актуальність і динамічний розвиток у відповідь на потреби користувачів. Управління платформою здійснює італійська компанія Engineering, яка була заснована в 1980 році й має великий досвід у розробці корпоративного програмного забезпечення.

KNOWAGE виділяється своїм широким функціоналом і гнучкістю, що дозволяє ефективно обробляти великі обсяги даних, здійснювати глибоку аналітику та візуалізувати результати у вигляді зручних дашбордів, звітів і аналітичних панелей. Система підтримує різноманітні джерела даних, включаючи реляційні та нереляційні бази, файлові сховища та веб-сервіси, що дозволяє легко інтегрувати її в наявну IT-інфраструктуру компаній. KNOWAGE надає інструменти для аналітики в реальному часі, підтримує багатомовність і пропонує

різноманітні модулі, такі як аналіз даних, інтерактивні звіти, географічна інформаційна система (GIS), а також інструменти для прогнозної аналітики та Big Data [11].

Завдяки відкритому коду та підтримці широкої спільноти, KNOWAGE легко адаптується до специфічних потреб організацій, надаючи можливість налаштовувати окремі компоненти під вимоги конкретного бізнесу. Це робить систему привабливим рішенням для фінансових установ, зокрема банків, які прагнуть ефективно здійснювати аналіз ризиків і відслідковувати фінансові показники в умовах динамічного ринку.

За результатами порівняльного аналізу можна зробити висновок, що KNOWAGE має декілька переваг над альтернативними системами.

Однак, KNOWAGE, хоч і пропонує потужні аналітичні інструменти, виявилася занадто складною для інтеграції шляхом доопрацювання програмного коду та його розширення для побудови системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфеля банку. Хоча ця платформа передбачає високий рівень налаштування і підтримку широкого спектра бізнес-функцій, вона також вимагає значних зусиль для налагодження процесу розробки користувацьких модулів, що ускладнює побудову системи.

При роботі з відкритим кодом KNOWAGE виникали проблеми, зокрема труднощі з компіляцією, відмінності в роботі з різними операційними системами та недосконала документація. Серед технічних викликів – процес компіляції, який вимагав ретельного налаштування середовища, а також окремих параметрів, що ускладнювало інтеграцію на різних платформах. Зокрема, виникали труднощі із забезпеченням стабільної роботи при побудові проєкту з відкритого коду на різних ОС, таких як Windows та Linux.

Крім того, наявна документація часто виявлялася неповною, що затримувало процес інтеграції KNOWAGE для використання як основний компонент візуалізації системи. Відсутність детальних інструкцій щодо налаштування середовища, компіляції проєкту, налаштування інтеграційних модулів та кастомізації інтерфейсу Відсутність детальних інструкцій щодо

налаштування середовища, компіляції проєкту, конфігурації інтеграційних модулів та кастомізації інтерфейсу створювала значні труднощі для реалізації цілісного рішення. Ці обмеження переважували потенційні переваги KNOWAGE, що, в свою чергу, зробило цю систему менш бажаним вибором для побудови інтерактивного аналізу портфельних ризиків кредитного портфелю банку.

Враховуючи ці аспекти, вибір було зроблено на користь Metabase, яка завдяки своїй простоті інтеграції та структурованій документації дозволяє швидко адаптувати її під фронт-енд частину системи.

Metabase є ефективним інструментом для інтеграції у систему підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку завдяки своїм широким функціональним можливостям і простоті використання. Ця BI-платформа підтримує інтеграцію з популярними базами даних, такими як PostgreSQL, MySQL і SQL Server, що забезпечує сумісність із різними інфраструктурами банків.

Однією з ключових переваг Metabase є інтуїтивний інтерфейс, який дозволяє ризик-аналітикам створювати звіти та дашборди без глибоких технічних знань. Завдяки інтерактивній візуалізації даних, аналітики можуть швидко ідентифікувати тренди та потенційні ризики. Metabase також забезпечує можливість автоматизації звітності, що знижує ручну працю і мінімізує ризик помилок.

5.2.2 Бібліотека Tabulator

Мовою для фронт-енд розробки було обрано JavaScript, що стало зваженим рішенням завдяки його універсальності, широким можливостям для інтерактивного та динамічного відображення даних, а також через JavaScript бібліотеку Tabulator. Бібліотека Tabulator забезпечує широкий функціонал для побудови табличних інтерфейсів і ефективно працює з великими обсягами даних, що є ключовим при реалізації компонента імпорту файлів. У нашому випадку вона дозволяє зручно керувати даними імпорту та надає інструменти для

налаштування параметрів імпорту файлів, створення конфігурацій та повторного запуску процесів імпорту.

Застосування Tabulator дало можливість розробити таблицю з інтерактивними елементами, що підтримує такі функції, як сортування, налаштування колонок і редагування даних безпосередньо в інтерфейсі. Редагування прямо у таблиці забезпечує зручність у внесенні змін до даних без необхідності повторного завантаження файлів, що значно економить час і підвищує ефективність роботи користувачів.

Однією з особливих переваг використання Tabulator стало те, що цей інструмент підтримує динамічні зміни у конфігураціях колонок і структурі таблиць, що дозволяє налаштовувати інтерфейс для роботи з різними форматами файлів та з урахуванням специфіки їхнього вмісту. Це забезпечує можливість створення окремих конфігурацій для кожного типу файлів, що імпортуються. Кожна така конфігурація може бути збережена і легко відтворюється під час повторного імпорту подібних файлів, що значно спрощує робочий процес і мінімізує можливість помилок під час обробки даних.

Крім цього, Tabulator пропонує зручні механізми роботи з даними через API, що дозволяє легко інтегрувати його з іншими частинами системи і забезпечити синхронізацію даних між фронт-енд і бек-енд частинами. Завдяки цьому, таблиці, побудовані з Tabulator, можуть автоматично оновлюватися за даними з сервера, що робить систему актуальною і інтерактивною в реальному часі.

Таким чином, використання JavaScript у поєднанні з бібліотекою Tabulator для фронт-енд розробки дозволило створити гнучкий і функціональний інтерфейс для управління імпортом файлів у системі підтримки аналізу портфельних кредитних ризиків.

5.3 База даних

PostgreSQL була обрана як основна система керування базами даних (СКБД) з кількох вагомих причин. Це потужна, надійна та високо масштабована реляційна база даних з відкритим кодом, яка широко використовується у різних галузях завдяки своїм перевагам у зберіганні та обробці великих обсягів даних. PostgreSQL є ідеальним вибором для створення ELT (Extract, Load, Transform) сховища даних, що дозволяє ефективно здійснювати збирання, завантаження та обробку даних для подальшого аналізу ризиків.

PostgreSQL відома своєю стабільністю, що є критичним фактором при роботі з великими обсягами фінансових даних. Вона підтримує транзакції ACID (Atomicity, Consistency, Isolation, Durability), що гарантує надійність і цілісність даних навіть у випадку збоїв системи або помилок при обробці запитів. Для банківської системи, де точність і стабільність даних є життєво важливими, це є безперечним плюсом.

Оскільки система потребує зберігання та обробки великих обсягів інформації (наприклад, даних про портфельні ризики, фінансові показники тощо), PostgreSQL забезпечує необхідну масштабованість як за рахунок вертикального, так і горизонтального масштабування. Вона підтримує кластеризацію та реплікацію, що дозволяє організувати ефективне зберігання даних і швидкий доступ до них навіть при зростанні обсягів інформації.

PostgreSQL підтримує широкий спектр складних SQL-запитів, що дає можливість ефективно виконувати аналіз даних без необхідності додаткових обчислювальних ресурсів. Вона має потужні можливості для обробки та аналізу структурованих даних, що є критичним для завдань кредитного аналізу та прогнозування ризиків.

Однією з основних переваг PostgreSQL є її здатність інтегруватися з іншими популярними інструментами та системами, що використовуються в процесах збору та обробки даних. Наприклад, PostgreSQL легко взаємодіє з різними ETL/ELT інструментами та BI-системами, такими як Metabase, що дозволяє

безперешкодно передавати дані між системами для подальшого аналізу та візуалізації. Це забезпечує ефективний процес підготовки даних та їх інтеграцію в єдину систему.

PostgreSQL підтримує широкий спектр типів даних, зокрема географічні та мультимедійні типи, що може бути корисним для аналітичних систем, де необхідно працювати з не тільки числовими, а й текстовими або іншими типами даних. Крім того, вона дозволяє зберігати дані у вигляді JSON, що зручно для обробки структурованих, напівструктурованих і навіть неструктурованих даних.

PostgreSQL має потужні інструменти для захисту даних, такі як підтримка шифрування даних на рівні таблиць, використання ролей і прав доступу, а також підтримка багатофакторної автентифікації. Для фінансових установ, де забезпечення конфіденційності та безпеки є критичним, ці функції є дуже важливими.

Постійне оновлення великих обсягів даних і виконання складних запитів є частими вимогами до системи, особливо в контексті ELT-процесів. PostgreSQL дозволяє проводити складні трансакції і обробляти великі набори даних із мінімальними затримками завдяки своїм можливостям оптимізації запитів та підтримці індексації.

Для системи підтримки інтерактивного аналізу портфельних кредитних ризиків PostgreSQL став ключовим елементом для ефективного зберігання, обробки та трансформації фінансових даних. Використання ELT-підходу дозволяє спочатку витягувати (Extract) дані з різних джерел (внутрішніх і зовнішніх), завантажувати їх (Load) у базу даних, а потім трансформувати (Transform) для подальшого аналізу і візуалізації.

Висновки до розділу 5

У цьому розділі детально розглянуто засоби розробки системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфеля банку. Вибір

технологій було обґрунтовано їх перевагами, гнучкістю та відповідністю вимогам проекту.

По бек-енд частині було встановлено, що використання мови Java і Spring Framework забезпечує надійність, масштабованість і гнучкість у розробці серверної частини. Spring Framework дозволив інтегрувати різноманітні модулі для автоматизації процесів обробки даних, забезпечення безпеки і масштабованості, що є критично важливим для фінансових систем.

Для фронт-енд розробки обрано JavaScript з використанням бібліотеки Tabulator. Ця комбінація дозволила створити динамічний і зручний інтерфейс для інтерактивного управління даними, включаючи функції сортування, редагування й імпорту файлів. Tabulator забезпечує високий рівень інтеграції з бек-ендом і автоматизацію обробки даних.

BI-система Metabase була інтегрована завдяки її простоті використання, потужним інструментам візуалізації і легкості налаштування. Хоча KNOWAGE пропонувала ширший функціонал, її інтеграція виявилася складною через обмеження документації і складність налаштувань.

Вибір PostgreSQL як системи керування базами даних обумовлений її стабільністю, підтримкою транзакцій ACID і масштабованістю. PostgreSQL ефективно обробляє великі обсяги фінансових даних і забезпечує надійне зберігання та трансформацію даних у рамках ELT-процесів.

Завдяки широкій підтримці типів даних, інтеграції з BI-системами і сильним інструментам безпеки, PostgreSQL стала ключовим компонентом для роботи з великими обсягами структурованих і напівструктурованих даних.

Сформована технологічна основа дозволяє створити систему, яка забезпечує швидку обробку та аналіз даних, інтеграцію з внутрішніми та зовнішніми джерелами інформації, а також високий рівень безпеки. Вибір Java, Spring Framework, PostgreSQL і Metabase робить систему ефективною, масштабованою та легкою у використанні для потреб банківської аналітики.

6 ПРОГРАМНА РЕАЛІЗАЦІЯ

Програмна реалізація системи включає такі етапи:

- реалізація бек-енд частини;
- створення сховища даних;
- реалізація фронт-енд частини;
- інтеграція з обраною BI-системою.

6.1 Бек-енд частина

Серверна частина системи була реалізована на мові програмування Java з використанням фреймворку Spring Framework. Діаграму класів бек-енд частини представлено у додатку К.

Архітектура системи базувалась на моделі MVC, що забезпечило чітке розділення логіки на кілька рівнів:

- Data Layer (JPA Repositories);
- Service Layer;
- Controller Layer;
- View Layer (у фронтенд-частині). [12]

Для доступу до бази даних використовувались JPA Repository, створені для кожної окремої моделі даних [13] [14]. Логіка обробки даних була реалізована в сервісних класах, а потім передавалась до контролерів, які забезпечували зв’язок із клієнтською частиною системи.

5.1.1 Репозиторії

У системі пристуні 3 репозиторії:

- ImportRepository;
- SourceRepository;
- FileInsertRepository.

Інтерфейси ImportRepository та SourceRepository були реалізовані як похідні від CrudRepository, що входить до складу Spring Framework. Такий підхід значно спрощує взаємодію з базою даних, оскільки вимагає лише налаштування відповідних моделей сутностей, таких як Import і Source, з урахуванням конфігурацій таблиць у базі даних. В результаті цей метод надає повний набір CRUD-операцій (створення, читання, оновлення та видалення), необхідних для роботи на рівні сервісів, що дозволяє мінімізувати кількість ручного коду для взаємодії з базою даних.

ImportRepository виконує функції зберігання, редагування та керування конфігураціями імпорту файлів, які налаштовуються користувачем. Модель Import інкапсулює всі параметри, необхідні для імпорту файлів. До них належать:

– джерело даних – можливість імпортувати дані як з локальних файлів, так і з зовнішніх ресурсів в інтернеті;

– тип файлу – підтримка різних форматів файлів для зручності імпорту;

– шлях до файлів – можливість зазначати шлях до каталогу з файлами або застосовувати маску для мультиімпорту, що забезпечує одночасну обробку кількох файлів;

– шлях до архіву файлів – дозволяє архівувати файли після імпорту для подальшого використання чи зберігання;

– назва таблиці – вказує на відповідну таблицю у базі даних, куди будуть імпортовані дані;

– внутрішня інформація системи – зберігає технічні параметри, такі як дата, час імпорту та id користувача, необхідні для функцій системного рівня;

– додаткові параметри: специфічні конфігурації, які можуть знадобитися залежно від конкретних вимог для кожного типу імпорту. Ці дані трансформуються у формат JSON-B для зберігання у базі даних.

Джерела даних можуть бути отримані за допомогою раніше згаданого інтерфейсу SourceRepository. Для зручного зберігання і використання метаданих про ці джерела у моделі Source передбачено декілька ключових полів:

– повна назва – офіційна назва джерела даних, яка використовується для ідентифікації та забезпечення зрозумілості записів у базі даних;

– скорочена назва – коротка версія назви, яка зручна для відображення в інтерфейсі або при побудові звітів;

– абревіатура – ще більш компактний формат представлення назви, який може використовуватися у візуальних елементах, звітах або налаштуваннях інтерфейсу;

– код країни з довідника K040 – спеціальний код з довідника Національного банку України, що відповідає країні джерела і дозволяє легко ідентифікувати, звідки походять дані, та допомагає в інтеграції з іншими системами або довідниками;

– внутрішня інформація системи.

Репозиторій FileInsertRepository розроблений для створення та наповнення даними таблиць у базі даних з файлів, які імпортуються користувачами. Цей клас забезпечує автоматичне створення таблиць, встановлення структури колонок, а також наповнення їх даними у пакетному режимі.

Ключові можливості FileInsertRepository включають:

– автоматичне створення таблиць на основі даних, що надходять;

– визначення типів даних для колонок;

– гнучкість при обробці різноманітних даних;

– пакетна обробка для підвищення продуктивності;

– гнучке управління значеннями колонок.

Репозиторій дозволяє створювати таблиці в базі даних, орієнтуючись на структуру першого рядка даних. Для кожної колонки автоматично визначається її назва та тип, залежно від вмісту. Якщо назва колонки виявляється порожньою, їй автоматично присвоюється ім’я, що містить префікс і порядковий номер. Створення таблиці враховує наявність ключових полів, таких як первинний ключ bri_id та додаткові колонки для службових даних bri_time, bri_uid і bri_f_time, що дозволяє відстежувати інформацію про час імпорту, користувача та інші метадані.

Для кожної колонки визначається SQL-тип, виходячи з типу значень у колонці. Для цього перевіряються типи всіх значень, щоб визначити оптимальний SQL-тип, уникаючи конфліктів у разі наявності змішаних типів. Наприклад, якщо виявлено числа з плаваючою комою та цілі числа, для колонки автоматично встановлюється тип DOUBLE PRECISION.

Репозиторій підтримує широкий набір типів даних (рядки, числа, дати, JSON), що дозволяє працювати з файлами, які містять неоднорідну інформацію. У разі виявлення різних типів значень у колонці, вона конвертується у формат VARCHAR, щоб забезпечити сумісність даних.

Для завантаження великих обсягів даних реалізована пакетна обробка (batch processing), що дозволяє групувати кілька записів в одну транзакцію. Це значно оптимізує швидкість завантаження, особливо при роботі з великими файлами, адже пакетний режим знижує навантаження на базу даних.

Додатковий метод налаштування значень у підготовленому запиті забезпечує коректне налаштування значень кожного типу, використовуючи розширені методи для налаштування специфічних SQL-типів.

Таким чином, FileInsertRepository надає можливості завантаження та обробки великих масивів даних з файлів у базу даних, оптимізує витрати ресурсів на завантаження і дозволяє гнучко налаштовувати структуру таблиць, що критично важливо для інтерактивного аналізу в системі підтримки кредитних портфельних ризиків.

6.1.2 Сервіси

У системі присутні 3 сервіси:

– ImportService;

– SourceService;

– WarehouseService.

ImportService та SourceService реалізують доступ до CRUD-функцій відповідних репозиторіїв (ImportRepository та SourceRepository), водночас

забезпечуючи мінімально необхідну бізнес-логіку для коректної роботи з конфігураціями імпорту та джерелами даних. Ці сервіси виступають як додатковий шар, що дозволяє безпечно й ефективно оперувати базовими функціями створення, читання, редагування та видалення записів у базі даних.

WarehouseService надає розширений набір методів для інтеграції файлів різних форматів у базу даних. Кожен підтримуваний системою тип файлу (JSON, CSV, XML, Excel, TXT) має окремий метод обробки, налаштований відповідно до специфіки структури та формату даних. Ці методи містять індивідуальну бізнес-логіку попередньої підготовки даних, що гарантує ефективне та точне завантаження інформації.

Для кожного типу файлу забезпечується спеціальна логіка трансформації вхідних параметрів. Наприклад, якщо для Excel-файлу не вказано межі таблиці (останній стовпчик), або зазначено некоректне значення, воно автоматично трансформується в числове значення -1, що підтримується методом парсингу Excel-файлів.

Для TXT-файлів передбачено можливість використання індексів розбиття стовпців. Ці індекси вказуються користувачем у вигляді текстового списку, сервіс, у свою чергу, конвертує його у список числових значень (List<Integer>), що дозволяє легко розпізнавати та розбивати стовпці за вказаними індексами під час завантаження даних.

WarehouseService також користується послугами класів-утиліт для читання та парсингу файлів у необхідний формат і далі передає готові дані до репозиторію на обробку.

6.1.3 Файловий сервер та класи-утиліти

Через обмеження, пов’язані з роботою веб-додатків із локальними файловими системами користувачів, було обрано оптимальне рішення для забезпечення ефективного управління даними в системі підтримки інтерактивного аналізу портфельних ризиків кредитного портфеля банку. Найкращим варіантом

стала організація централізованого файлового сервера, який забезпечує зручний доступ як для кінцевих користувачів, так і для веб-сервера системи.

Для ефективного функціонування файлового сервера важливо врахувати:

– сумісність із ОС клієнтів – сервер повинен підтримувати протоколи, які працюють на різних операційних системах користувачів (Windows, Linux);

– архівування файлів – налаштування архівування даних, щоб розподілити оброблені дані від необроблених та мінімізувати ризики втрати інформації.

Організація файлового сервера дає змогу користувачам завантажувати вихідні дані, такі як звіти чи файли обліку, безпосередньо на сервер. З іншого боку, веб-додаток автоматично отримує доступ до цих файлів, проводить парсинг і зберігає структуровану інформацію у базі даних для подальшого аналізу.

Для інтеграції роботи з файловим сервером та первинної обробки файлів різних форматів у системі реалізовані 3 класи-утиліти – це FileReadr, FileParser та FileArchiver.

Клас-утиліта FileReadr забезпечує весь функціонал зчитування файлів як з файлової системи, так і з веб-ресурсів, конвертуючи вміст файлів у текстовий формат (String). Завдяки цьому класу, система може легко опрацьовувати дані з різних джерел, зокрема з URL-адрес та локальних шляхів.

Для обробки шляхів розташування файлів компонент FileReadr забезпечує автоматичне перетворення шляхів до файлів у формат, сумісний із локальним файловим середовищем веб-сервера. Цей механізм враховує особливості побудови шляхів різними операційними системами і дозволяє уникнути проблем із несумісністю форматів. У випадку Windows, наприклад, шляхи часто мають зворотні слеші (\) і можуть містити літери дисків. При налаштуванні доступу до файлового серверу за допомогою мережевого диску, типовий шлях розташування файлу може виглядати так: R:\In\report.xlsx.

Для зчитаних файлів автоматично визначається кодування за допомогою бібліотеки Apache Tika, яка використовує технології штучного інтелекту для аналізу байтової структури файлу. Це дозволяє обробляти файли з різними кодуваннями, такими як UTF-8, Windows-1251, ISO-8859-1 та інші, і робить

процес визначення кодування більш адаптивним і точним, особливо для текстових файлів без явно зазначеного кодування.

Особливий підхід застосовується для файлів типу XML, які можуть містити декларацію кодування у спеціальному тегу (<?xml version=”1.0” encoding=”ENCODING”?>). FileReadr спочатку сканує XML-файл на наявність такого тегу, оскільки це є надійнішим способом визначення кодування. Застосування цього підходу особливо важливе для XML-файлів, що можуть містити підпис КЕП (кваліфікований електронний підпис), оскільки цей підпис може викликати помилки у штучному інтелекті при спробі визначити кодування виключно за байтами. Завдяки цьому клас забезпечує більш точне та стабільне зчитування таких файлів.

Клас-утиліта FileParser відповідає за перетворення вмісту зчитаних файлів у формат, придатний для обробки в системі, а саме – у тип List<Map<String, Object>>. Цей тип було обрано для зберігання табличних даних, оскільки він ідеально підходить для зберігання структурованої інформації, де кожен Map<String, Object> відображає один рядок таблиці, де ключ – це назва стовпчика, а значення – це вміст комірки відповідного типу.

FileParser містить методи, налаштовані для парсингу різних форматів файлів, таких як JSON, CSV, XML, Excel, і TXT, які можуть бути зчитані у текстовий формат (String) за допомогою класу FileReadr. Основне завдання FileParser – забезпечити конвертацію цих текстових даних у стандартний формат List<Map<String, Object>>, незалежно від вихідного типу файлу.

Метод parseJson виконує парсинг JSON файлів у формат List<Map<String, Object>>, незалежно від складності структури JSON. Це дозволяє забезпечити уніфіковане представлення даних у вигляді табличної структури, що значно спрощує подальшу обробку інформації.

У випадку, коли JSON являє собою простий масив елементів, метод зчитує кожен об’єкт у масиві та додає його до List<Map<String, Object>>. Такі JSON файли найчастіше використовуються для зберігання рядів даних, де кожен об’єкт описує набір атрибутів і значень для одного запису. Коли такий масив

розпізнається, дані просто зберігаються у вихідному форматі без додаткової обробки.

Часто деякі JSON файли, надані НБУ, представлені у вигляді двовимірного масиву об’єктів. Кожен рядок масиву містить списки значень, де кожен об’єкт представляє окреме значення та додаткову інформацію, як-от DATE_B та DATE_E, які описують періоди актуальності даних. Наявність цих полів дозволяє точно відслідковувати зміни в даних, що може бути корисно для аналізу історичних даних.

У процесі обробки двовимірного масиву зчитуються значення DATE_B та DATE_E для кожного запису. На основі цих дат створюються поля D_OPEN, D_MODI, D_CLOSE, що відображають періоди активності запису: D_OPEN – дата початку дії, D_MODI – дата модифікації запису, D_CLOSE – дата, коли запис втратив актуальність. Це стандарт, поширений у форматі файлів НБУ, зокрема для Excel, що забезпечує кращу сумісність при імпорті файлів різних типів.

Метод parseXML в класі FileParser відповідає за парсинг XML-файлів у структуру List<Map<String, Object>>. Основна задача цього методу – обробити дані з XML-файлів й уніфікувати їх для подальшого використання системою.

Метод спершу використовує допоміжний метод, який очищує XML від зайвих або некоректних фрагментів, що можуть порушувати парсинг. Для цього визначаються стартові та кінцеві теги основного контенту (у банківських системах використовуються теги <DECLAR>…</DECLAR> або <NBUSTATREPORT>…</NBUSTATREPORT>). У разі наявності зайвих даних (наприклад підпису КЕП), метод видаляє все, що передує основному контенту.

Після очищення XML, метод створює структуру для парсингу, використовуючи об’єкти DocumentBuilder та Document. Основні елементи, такі як HEAD чи DECLARHEAD (дані заголовку файлу) та DATA чи DECLARBODY (дані основної таблиці), знаходяться й обробляються окремо. Це дає можливість зберігати заголовкову інформацію разом з табличними даними, роблячи їх повністю зрозумілими і придатними для подальшого аналізу.

Кожен елемент DATA чи DECLARBODY обробляється як окремий рядок даних. Метод addXmlRow заповнює рядок відповідними атрибутами та значеннями, зчитуючи дані з дочірніх вузлів XML-елементів. Додатково в рядок додається інформація з заголовка (HEAD), що може містити важливі метадані про файл.

Таким чином, метод забезпечує зчитування як заголовкових, так і табличних даних з XML у єдиний формат.

Парсинг Excel-файлів у методі parseExcel виконано за допомогою бібліотеки Apache POI, яка надає інструментарій для роботи з Excel-документами. Основна мета методу – конвертувати дані з Excel в єдину структуру, зокрема у список карт (Map), де кожен елемент містить рядок таблиці з відповідними значеннями клітинок.

Метод починається з перевірки та ініціалізації необхідних параметрів, таких як встановлення максимального розміру для файлів. Потім, використовуючи бібліотеку Apache POI, відкривається потік до файлу та створюється об’єкт Workbook (робоча книга), з якого отримується конкретний аркуш (Sheet) за заданим в параметрах методу індексом (sheetNum).

Для коректного оброблення таблиці визначаються межі даних.

Визначаються межі рядків за допомогою параметрів fromRow і toRow. Якщо toRow не вказано, метод використовує останній рядок аркуша;

За допомогою параметрів fromCell і toCell встановлюються межі для стовпців. Якщо стовпці з порожніми назвами зустрічаються більше певної кількості, то наступні стовпці ігноруються, так як вважається, що було задане завелике граничне значення, щоб не записувати зайві порожні дані до таблиці.

На основі першого рядка таблиці (зазвичай це заголовок) створюється список назв стовпців. Якщо клітинка порожня або є частиною об’єднаних клітинок, то метод намагається з’ясувати значення, перевіряючи об’єднані ділянки.

Крім того, ім’я стовпця очищується (видаляються зайві пробіли, символи нового рядка тощо) і у разі необхідності форматується, щоб відповідати певним вимогам (наприклад, обмеженням на довжину).

Після визначення назв стовпців метод переходить до парсингу самих рядків. Для кожного рядка:

Перевіряється кожна клітинка в межах зазначених стовпців.

Якщо клітинка має значення (не є порожньою), це значення додається до запису (Map), де ключем є назва стовпця, а значенням – вміст клітинки.

Якщо рядок містить лише порожні клітинки, він пропускається.

Якщо клітинка містить формулу, вона обробляється окремо за допомогою об’єкта FormulaEvaluator, який дозволяє обчислювати результат формули, і відправляється користувачеві у вигляді готового значення.

Для числових значень перевіряється, чи це дата, і у разі необхідності, дата повертається у відповідному форматі.

В разі виникнення помилок при роботі з файлом (наприклад, якщо файл зашифровано або виникла проблема з форматом) метод ловить виключення і виводить відповідну інформацію для налагодження.

Результатом роботи методу є List<Map<String, Object>>, де кожна Map представляє один рядок даних з Excel, а значення клітинок цього рядка зберігаються під ключами – назвами стовпців.

Таким чином, метод дає змогу ефективно обробляти Excel-файли та конвертувати їх в універсальний формат для подальшої обробки.

Метод parseCSV виконує парсинг CSV-файлів, перетворюючи вміст у список мап, де кожен рядок CSV відповідає одному елементу в списку. Метод приймає чотири параметри: вміст файлу як рядок, роздільник для колонок, назву файлу та формат дати.

Основні етапи роботи методу:

– якщо вміст файлу порожній або не визначений, метод повертає порожній список;

– весь текст розбивається на окремі рядки за допомогою символу нового рядка;

– перший рядок використовується як заголовок, кожен елемент якого стає ключем у мапі для відповідних значень з наступних рядків;

– рядки, починаючи з другого, розділяються на окремі значення згідно з вказаним роздільником;

– якщо кількість значень у рядку не збігається з кількістю заголовків, рядок пропускається;

– кожен рядок перетворюється на мапу (Map), де ключами є назви колонок, а значеннями – відповідні дані з цього рядка.

Метод parseTxt у класі FileParser призначений для обробки текстових файлів із табличною структурою даних. Цей метод створений для роботи з текстовими файлами, зокрема звітами або файлами, наданими банками та іншими фінансовими установами. Його основне завдання – зчитати дані з файлу, виділити колонки та уніфікувати структуру, щоб дані можна було зручно обробляти далі в системі.

Метод parseTxt гнучкий у налаштуванні: він може обробляти дані двома способами, залежно від того, чи наданий список індексів колонок. Якщо індекси відомі, метод використовує їх для розбиття кожного рядка тексту на колонки. У разі, якщо індекси не надані, метод автоматично визначає позиції колонок на основі спеціальних символів-роздільників (найчастіше – вертикальних рисок |). Це дозволяє налаштувати обробку як для форматів з чітко визначеними межами колонок, так і для тих, де інформація організована за допомогою символів.

На першому етапі метод розбиває вміст текстового файлу на окремі рядки, щоб можна було працювати з ними окремо. Після цього виконується аналіз рядків для визначення:

– рядка із заголовками колонок, який зазвичай розташований на початку табличних даних;

– рядка з першими значеннями даних, що дозволяє знайти позиції роздільників і виділити межі колонок.

Метод ідентифікує назви колонок з рядка заголовка. У разі повторення назв метод застосовує унікальні позначення, додаючи індекси до однакових назв, щоб уникнути конфліктів. Це забезпечує зручність у подальшій роботі з даними, гарантуючи унікальність кожного ключа.

Після визначення структури колонок метод починає обробляти кожен рядок даних. Кожен рядок перетворюється на мапу (Map), де кожне значення прив’язане до своєї колонки. Порожні значення або прогалини замінюються на null, що дозволяє зберегти однакову структуру таблиці навіть при відсутності деяких значень. Крім того, метод пропускає службові рядки, що складаються лише з роздільників (наприклад, _ або -), щоб не додавати зайвих елементів до таблиці.

Усі методи парсингу також здатні визначити дату звіту на основі імені файлу, використовуючи шаблон дати, який може надати користувач. Це дозволяє додати поле з датою звіту для кожного рядка даних, що полегшує відстеження періодичності та актуальності інформації.

Третій клас-утиліта FileArchiver є ключовим компонентом системи, що відповідає за всі операції, пов’язані з архівуванням файлів на файловому сервері. Його основна мета – забезпечити структуроване зберігання використаних файлів, зберегти історію імпортів і запобігти повторному завантаженню або обробці одних і тих самих даних. Це досягається шляхом організації файлів у систематизованих каталогах із чіткою структурою, що базується на поточній даті.

Після обробки файлів, таких як звіти або дані для аналізу, FileArchiver переміщує їх у спеціально створені директорії. Це дозволяє уникнути плутанини між новими та обробленими файлами. Каталоги організовуються з урахуванням поточної дати, що полегшує пошук архівованих даних у майбутньому. Наприклад, файли можуть зберігатися за шляхом виду /archive/2024/20241108/, що дозволяє швидко знайти дані за конкретний день.

Для файлів, отриманих із зовнішніх джерел (наприклад, через API або з веб-сайтів), FileArchiver також завантажує їх локально.

6.1.4 Контролери

У системі наявні три контролери: ImportController, SourceController та WarehouseController.

Контролери ImportController та SourceController є важливими складовими серверної частини системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфеля банку. Вони реалізовані на основі REST-архітектури, що забезпечує уніфікований підхід до обробки запитів, полегшуючи інтеграцію з іншими компонентами системи та сторонніми сервісами [15]. Завдяки використанню сучасних технологій і підходів до розробки, обидва контролери гарантують зручність, ефективність і надійність у виконанні своїх функцій.

Контролери працюють із запитами з фрон-енду, дозволяючи системі отримувати, обробляти та повертати дані про конфігурації імпорту та джерела у форматі JSON. Завдяки анотаціям Spring, таким як @RestController та @RequestMapping, забезпечується автоматична маршрутизація запитів до відповідних методів, що спрощує розробку та підтримку коду. Анотація @RestController об’єднує можливості @Controller і @ResponseBody, автоматично серіалізуючи відповіді у формат JSON, що робить процес обробки запитів і надання даних фронт-енду максимально ефективним.

ImportController відповідає за роботу з інформацією про імпорти. Зокрема, він дозволяє отримувати список усіх записів імпорту, додавати нові дані, оновлювати існуючі або видаляти записи за ідентифікатором. Для реалізації цих функцій використовуються HTTP-запити різних типів: GET, POST та DELETE. Усі методи забезпечують повернення відповідних HTTP-статусів, що є стандартом для REST-сервісів і дозволяє легко обробляти результати взаємодії.

SourceController виконує аналогічні функції, але для роботи із джерелами даних. Як і у випадку з ImportController, SourceController дозволяє додавати, оновлювати або видаляти записи про джерела. Таке розмежування

відповідальності між контролерами сприяє модульності системи, спрощуючи її подальшу підтримку та масштабування.

Додатково використання залежностей, які інжектуються за допомогою анотації @Autowired, дозволяє контролерам легко взаємодіяти з сервісними класами, такими як ImportServiceImpl, що реалізує бізнес-логіку. Це забезпечує чіткий розподіл між рівнями обробки даних, зберігання логіки та взаємодії із зовнішнім середовищем.

DatabaseSetupController є головним контролером для роботи з веб формою, у якій користувач заповнює і викликає конфігурації імпорту файлів Основне завдання цього контролера – забезпечити зручний інтерфейс для конфігурації джерел даних, управління імпортом файлів і інтеграції інформації з різних форматів у єдине сховище. Контролер приймає дані по викликаному імпорту та викликає необхідні методи сервісу WarehouseService та класів-утиліт.

Контролер використовує анотацію @Controller, що вказує на його роль у взаємодії з веб-інтерфейсом. Основний метод home() відповідає за початкове завантаження сторінки конфігурації.

Однією з ключових функцій контролера є обробка даних, які передаються через HTTP-запити, зокрема методом insertTable(). Цей метод приймає об’єкт DTO (Data Transfer Object) із параметрами, заданими користувачем у формі. DTO включає інформацію про тип джерела, формат даних, шлях до файлу, роздільники колонок, параметри обробки аркушів Excel тощо. Дані проходять валідацію, а у разі виявлення помилок користувачу повертається повідомлення з описом проблеми.

Обробка імпорту реалізована в методі updateDatabase(). Він забезпечує підтримку різноманітних джерел даних, таких як локальні файли та веб-ресурси. Для кожного формату файлу передбачено окремий сценарій обробки.

Файли JSON, наприклад, можуть бути завантажені з локального сховища або безпосередньо з веб-ресурсів. Дані додаються в базу через методи сервісу WarehouseServiceImpl, що інтегрується з контролером через анотацію @Autowired.

Для обробки електронних таблиць Excel передбачена можливість роботи з декількома аркушами одночасно. Користувач може задавати діапазони рядків і колонок, а також специфічні формати дат.

Контролер також інтегрується з класами-утилітами, такими як FileReadr і FileArchiver, що відповідають за пошук файлів, відповідних заданим маскам, і їх архівування після обробки. Це дозволяє зберігати історію імпортованих даних і запобігати повторній обробці тих самих файлів.

6.2 Фронт-енд частина

Фронт-енд частина відповідає за табличний інтерфейс системи та інтеграцію з BI-системою Metabase.

Фронт-енд системи виконує ключову роль у забезпеченні зручної взаємодії користувача з інтерфейсом. Реалізація заснована на використанні бібліотеки Tabulator, яка забезпечує побудову гнучких та функціональних таблиць для обробки даних. Інтерфейс системи побудований так, щоб користувач міг швидко вносити дані, виконувати операції імпорту та керувати параметрами, необхідними для подальшої обробки на бек-енді.

Таблиця, побудована на основі Tabulator, дозволяє здійснювати редагування полів, використовуючи різноманітні елементи управління, такі як випадаючі списки, текстові поля та чекбокси. Для інтеграції з бекендом використовується обмін даними за допомогою HTTP-запитів (GET та POST). При завантаженні таблиці система отримує дані з бек-енду, зокрема список конфігурацій імпорту, який оновлюється для кожного запису.

Особливістю фронт-енду є можливість роботи з багатоступеневим процесом імпорту. Перед початком імпорту користувач заповнює форму, в якій зазначає всі необхідні параметри. Якщо потрібно виконати кілька імпортів, система автоматично формує цикл POST-запитів, який забезпечує обробку даних послідовно для кожного імпорту.

Для зручності користувачів передбачено форматування рядків таблиці, яке динамічно змінюється залежно від вибраних умов. Наприклад, якщо користувач обирає формат даних CSV, то у відповідних полях відображаються тільки ті параметри, які характерні для цього формату, такі як роздільник. Якщо ж обрано формат Excel, то з’являються додаткові параметри для вказання номера листа, першого та останнього рядка чи стовпця. Ця динамічна логіка забезпечується обробкою JSON-параметрів, які зберігаються у базі даних у вигляді серіалізованих об’єктів. Під час обробки даних на фронт-енді ці параметри автоматично розпаковуються, надаючи користувачеві змогу вносити зміни, після чого дані знову упаковуються у формат JSON і надсилаються на сервер.

Фронт-енд також підтримує функціонал додавання та видалення. Для додавання нового запису користувач викликає відповідну дію, після чого система автоматично створює новий рядок із базовими параметрами, зберігаючи його на сервері. Видалення записів відбувається за допомогою відповідних запитів до API, після чого рядок видаляється як з бази даних, так і з інтерфейсу.

Таким чином, фронт-енд забезпечує високий рівень інтерактивності, що дозволяє ефективно працювати з даними та налаштовувати параметри обробки відповідно до потреб користувачів.

Також було модифіковано код BI-системи Metabase для безшовної інтеграції з розробленим фронт-ендом.

У бічне меню Metabase було додано посилання на сторінку імпорту файлів, що дозволяє заповнювати сховище даних інформацією прямо в одному веб-застосунку.

6.3 Сховище даних

6.3.1 Схема source_data

Після завершення процесу обробки на бек-енді, дані завантажуються до бази даних і зберігаються у схемі source_data. Ця схема виконує роль проміжного етапу, де дані зберігаються у своєму первинному вигляді після імпорту, без

застосування складної структури чи додаткової обробки. Таблиці, що створюються в межах цієї схеми, є незалежними одна від одної: вони не мають жодних взаємозв'язків, таких як зовнішні ключі, а також не використовують складних типів даних, таких як масиви чи структуровані об'єкти. Це забезпечує простоту у роботі з даними на цьому етапі, проте створює виклики для їхнього подальшого використання у аналізі даних.

Через необхідність забезпечення усієї необхідної інформації в одному робочому файлі таблиці схеми source_data часто містять значну кількість задубльованої інформації. Наприклад, однакові записи можуть міститися у різних файлах. Це призводить до надлишковості, яка може негативно впливати на ефективність зберігання та аналізу, збільшуючи час виконання запитів і споживання ресурсів бази даних.

Окрім дублювання, ще однією проблемою є наявність великої кількості порожніх значень у полях таблиць. Порожні поля можуть виникати через відсутність інформації в джерелах даних або через нерівномірну структуру різних джерел. Такі прогалини у даних ускладнюють їхню подальшу обробку, адже потребують додаткових перевірок та очищення під час виконання аналітичних запитів. Це також може викликати некоректне формування результатів у випадках, коли аналітичні алгоритми не враховують наявність пропущених значень.

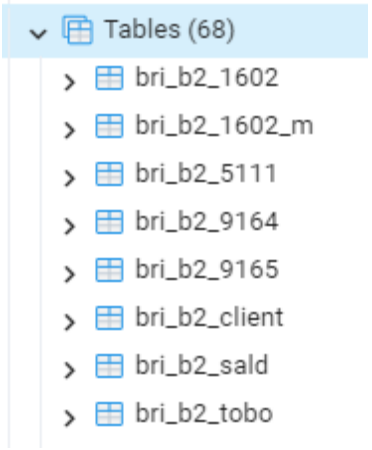

Рисунок 5.1 – Таблиці у схемі source_data

Окрім вищезазначеного, часто зустрічається ситуація, коли окремі дані з таблиць схеми source_data варто було б зберігати у виділених таблицях, щоб досягти кращої структурованості. Наприклад, інформація, яка повторюється через різні рядки таблиці, така як назви продуктів, категорії або статуси, могла б бути перенесена у окремі довідкові таблиці. Це дозволило б значно оптимізувати структуру бази даних, забезпечити кращу керованість даними та зменшити їх надлишковість. Більш того, це сприяло б нормалізації даних, яка є стандартом для побудови ефективних і масштабованих баз даних.

6.3.2 Схема analytics

Щоб вирішити виділені проблеми, у системі передбачено етап нормалізації даних, який запускається автоматично за допомогою тригерів після завантаження інформації у схему source_data. Під час цього етапу виконується реструктуризація даних, що включає перерозподілення інформації між таблицями схеми analytics сховища даних, пов’язання інформації між окремими таблицями з використанням зовнішніх ключів, а також перетворення даних у вигляд, придатний для аналітики [16]. Таблиці формуються у структурі, що мінімізує необхідність в полях з порожніми значеннями.

ER-діаграму схеми analytics представлено у додатку Л.

Довідник банків представлений таблицею banks, яка зберігає всю необхідну інформацію про банки, включаючи короткі, довгі та повні назви, адреси, а також всі можливі коди та ідентифікаційні номери (наприклад, МФО код). Ця таблиця забезпечує централізоване управління даними про банки та слугує ключовою ланкою для роботи з іншими елементами системи.

Для визначення країни банку використовується зв’язок із таблицею kl_k040, що є довідником Національного банку України (НБУ) з кодів країн. Таким чином, завдяки цьому зв’язку, система має можливість отримувати розширену інформацію про країну банку, зокрема її код або резидентний статус. Таблиця

kl_k040, у свою чергу, використовує зв’язок із таблицею kl_k030, яка зберігає коди резидентності.

Унікальні клієнтські коди в системі зберігаються в таблиці bri_client, яка може містити як унікальний ідентифікатор клієнта системи, так і поле id_abs для зберігання ідентифікаторів клієнтів із зовнішніх автоматизованих банківських систем (АБС). Це дозволяє інтегрувати дані клієнтів з інших джерел, забезпечуючи уніфікованість і зручність обробки інформації.

Індикатори в системі зберігаються в таблиці bri_indicator, де кожен індикатор має унікальний ідентифікатор. Таблиця ref_indicators забезпечує альтернативні назви цих індикаторів, а також класифікує їх за допомогою поля gr, що дозволяє структуризувати індикатори за певними групами або категоріями. Значення індикаторів на конкретні дати зберігаються в таблиці indicators, яка є основним джерелом для аналізу змін цих показників у часі.

Джерела даних, з яких завантажуються індикатори та інші пов’язані дані, представлені в таблиці data_source. Ця таблиця використовується для інтеграції з таблицями ref_indicators, indicators, import, а також із фронт-ендом системи. Вона є критично важливою для забезпечення цілісності та повноти даних. Таблиця import зберігає конфігурації імпорту, які вводяться користувачами через інтерфейс, забезпечуючи гнучкість у налаштуванні та автоматизацію процесу обробки даних.

Додаткові довідники НБУ, такі як таблиця kl_k110, зберігають інформацію про види економічної діяльності (з довідника К110). Вона взаємодіє з таблицями kl_k111, kl_k112 та kl_k115, які забезпечують деталізацію цих видів діяльності відповідно до розділів, секцій та груп. Така структура забезпечує глибоку деталізацію і можливість класифікації економічної діяльності банків чи інших суб'єктів аналізу.

Додатково, для різних індикаторів, які зберігаються в таблиці indicators, можуть використовуватися інші довідники в залежності від їх типу та призначення. Ці зв’язки реалізуються через поле param, яке має формат JSON-B, що забезпечує гнучкість у зберіганні та обробці додаткових параметрів.

Таким чином, вся структура довідників та їх зв’язків забезпечує комплексну та багаторівневу інтеграцію даних, що дозволяє ефективно обробляти, класифікувати та аналізувати інформацію. Це є основою для підтримки роботи системи управління ризиками та забезпечує можливість автоматизації складних процесів аналітики.

6.4 Розгортання системи

Було побудовано діаграму розгортання системи, яку можна побачити у додатку А.

Система передбачає використання трьох основних компонентів: сервера додатків, сервера бази даних та файлового сервера. Система підтримує як роздільне розгортання серверів, так і поєднання їх на одному серверному пристрої. Під час тестування системи було обрано розгортання усіх серверів на одному фізичному пристрої під управлінням операційної системи Linux Ubuntu Server. Такий підхід спрощує налаштування взаємодії між сервісами, зменшує час на конфігурацію та забезпечує мінімальні затрати ресурсів.

У разі їхнього окремого розміщення необхідно додатково організувати доступ до файлового сервера з вебсервера. Це може бути реалізовано шляхом монтування директорії файлового сервера у вигляді мережевого диска через такий протокол, як SMB (Server Message Block). Він є найбільш придатним для забезпечення кросплатформної сумісності з клієнтськими пристроями.

Користувачам системи рекомендується підключити файловий сервер як мережевий диск до своїх пристроїв. Це дозволяє працювати з даними у зручному для них середовищі без необхідності переходу до специфічного інтерфейсу чи використання стороннього ПЗ. У разі відсутності користувача у тій самій локальній мережі, що й файловий сервер, можливе використання таких технологій:

– VPN (Virtual Private Network) – створення безпечного тунелю між клієнтським пристроєм і корпоративною мережею.

– Засоби віддаленого доступу – наприклад, організація доступу через вебінтерфейси, які є частиною корпоративних платформ, таких як Citrix або Microsoft Remote Desktop Services.

Ці методи є стандартними для банківських установ України, де безпека доступу до даних та їх передача є одними з ключових пріоритетів.

Для забезпечення сумісності з різними операційними системами клієнтських пристроїв було обрано використання пакету Samba, який реалізує підтримку протоколу SMB. Цей підхід дозволяє організувати безшовний доступ до даних для таких ОС:

– Windows – шляхом стандартного підключення мережевого диска через графічний інтерфейс або командний рядок.

– macOS – за допомогою вбудованих засобів підключення мережевих ресурсів.

– UNIX-подібні системи (включаючи Linux) – через пряме монтування директорій за допомогою CLI-утиліт.

Доступ до бази даних здійснюється шляхом використання інтерфейсу JDBC. Необхідні налаштування задаються у змінні середовища системи.

Прямий доступ до серверів має тільки системний адміністратор. Це підвищує безпеку системи та є стабільною практикою в більшості корпоративних установ.

Висновки до розділу 6

У розділі детально розглянуто процес реалізації системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфеля банку. Система складається з бек-енд і фронт-енд частин, а також інтеграції з BI-системою та організації сховища даних. Ключовими аспектами реалізації є використання сучасних технологій і підходів для забезпечення гнучкості, масштабованості та ефективності системи.

Для бек-енд частини було використано мову програмування Java з фреймворком Spring Framework, що дозволяє побудувати модульну архітектуру за моделлю MVC. Це забезпечило чіткий розподіл логіки на рівні даних, сервісів та контролерів.

Реалізовано три репозиторії, які спрощують CRUD-операції з базою даних, та три сервіси, що інтегрують функціонал для роботи з різними типами файлів.

Особлива увага приділена парсингу файлів різних форматів (JSON, XML, Excel, CSV, TXT) через спеціалізовані класи-утиліти, які адаптують вхідні дані до стандартного формату для подальшого аналізу.

Запроваджено файловий сервер, що забезпечує централізоване управління файлами, архівування та інтеграцію з веб-додатком.

Для фронт-енд частини створено інтерфейс на основі JavaScript із використанням бібліотеки Tabulator, яка забезпечує гнучкість і функціональність таблиць.

Забезпечено інтерактивність інтерфейсу, включаючи динамічну зміну параметрів залежно від формату даних, підтримку циклічного імпорту та редагування записів безпосередньо в таблицях.

Реалізовано інтеграцію з бек-ендом через API, що дозволяє автоматизувати процес імпорту та налаштування параметрів обробки.

В рамках інтеграції з BI-системою було вибрано Metabase завдяки простоті інтеграції та широкому функціоналу. Для зручності користувачів додано можливість імпорту даних прямо через веб-інтерфейс Metabase.

Модифікація BI-системи дозволила забезпечити її безшовну інтеграцію з розробленим інтерфейсом, що підвищило зручність роботи та зменшило часові витрати на перемикання між системами.

Сховище даних поділяється на дві ключові схеми: source_data для проміжного зберігання необроблених даних та analytics для нормалізованих і структурованих даних. Така архітектура дозволяє мінімізувати дублювання та порожні значення, забезпечуючи оптимальне середовище для аналітики.

Довідники, зв’язки між таблицями та автоматизовані механізми обробки даних створюють надійну базу для інтеграції з зовнішніми джерелами.

Комплексна структура системи сприяє автоматизації складних процесів обробки та аналізу даних, що є важливим для управління кредитними ризиками банків. Побудовані діаграми, інтеграція довідників і гнучкі налаштування імпорту підкреслюють її орієнтацію на кінцевого користувача, дозволяючи адаптувати систему до специфічних потреб банківської галузі.

Система є гнучкою, так як підтримує можливість як роздільного розміщення серверів, що вимагає додаткового налаштування взаємодії через мережеві протоколи, такий як SMB, так і розміщення серверів на одному серверному пристрої.

Використання пакету Samba забезпечує сумісність із різними операційними системами клієнтських пристроїв, спрощуючи доступ до даних і підвищуючи зручність для користувачів. Також розглянуто можливу потребу у віддаленому доступі до системи, запропоновано технології VPN та корпоративні вебінтерфейси.

7 ПРОВЕДЕННЯ АНАЛІЗУ ПОКАЗНИКІВ

Виконання аналізу кредитних портфельних ризиків є головним процесом, який система прагне оптимізувати.

Задля прикладної демонстрації аналітичних можливостей системи було проведено регресійний аналіз показника частки непрацюючих кредитів у кредитному портфелі (далі NPL). Цей показник є одним з ключових показників оцінки якості кредитного портфелю. Національний Банк України оприлюднює значення цього показника по банківській системі України щомісячно на своєму вебпорталі [17], що підкреслює його важливість в оцінці кредитних ризиків.

7.1 Попередня підготовка даних

За допомогою системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку було здійснено завантаження показників [18]:

– розподіл кредитів, наданих фізичним та юридичним особам у національній та іноземній валютах, та розміру кредитного ризику за класами боржника відповідно до Положення № 351 (у розрізі банків);

– розподіл кредитів, наданих суб’єктам господарювання в національній та іноземній валютах за видами економічної діяльності, що класифікуються за розділами, з них непрацюючих у визначенні, наведеному в Положенні № 351 (у розрізі банків);

– розподіл кредитів, наданих суб'єктам господарювання в національній та іноземній валютах за розміром суб'єкта господарювання та видами економічної діяльності, що класифікуються за розділами, з них непрацюючих у визначенні, наведеному в Положенні № 351;

– розподіл кредитів, наданих фізичним особам та суб’єктам господарювання в національній та іноземній валютах, та розміру сформованих резервів за стадіями

знецінення за міжнародним стандартом фінансової звітності 9 "Фінансові інструменти" (у розрізі банків);

– розподіл кредитів, наданих суб'єктам господарювання в національній та іноземній валютах, та розміру сформованих резервів за видами економічної діяльності та стадіями знецінення за міжнародним стандартом фінансової звітності 9 "Фінансові інструменти" (у розрізі банків);

– розподіл вкладів фізичних осіб та сума можливого відшкодування Фондом гарантування вкладів фізичних осіб (у розрізі банків).

Також для забезпечення нормалізації даних було імпортовано довідники Національного Банку, які використовуються для формування показників статистичної звітності [19]:

– k030 – код резидентності;

– k040 – код країни (відповідно до Національного стандарту України ДСТУ ISO 3166-1:2009 "Коди назв країн світу", затвердженого наказом Державного комітету України з питань технічного регулювання та споживчої політики від 23 грудня 2009 року № 471);

– k042 – код ознаки країни за засобами платежу (офшорні зони);

– k110 – вид економічної діяльності [відповідно до національного класифікатора України ДК 009:2010 "Класифікація видів економічної діяльності", затвердженого наказом Державного комітету України з питань технічного регулювання та споживчої політики від 11 жовтня 2010 року № 457 (зі змінами)];

– k111 – коди розділів видів економічної діяльності (узагальнені);

– k112 – коди секцій видів економічної діяльності (узагальнені);

– k115 – код групи видів економічної діяльності;

– r030 – код валюти або банківського металу [відповідно до Класифікатора іноземних валют та банківських металів, затвердженого постановою Правління Національного банку України від 04 лютого 1998 року № 34 (у редакції постанови Правління Національного банку України від 19 квітня 2016 року № 269) (зі змінами)];

– f082 – код типу боржника;

– fst – код зміни стадії знецінення/моделі очікуваних кредитних збитків, визнаної за Міжнародним стандартом фінансової звітності 9 “Фінансові інструменти”;

– s080 – клас боржника/контрагента.

Після виконання імпорту дані було автоматично нормалізовано за допомогою тригерів та процедур СКБД PostgreSQL. Дані з довідників було трансформовано у необхідну структуру відповідних таблиць схеми analytics сховища даних, а показники було структуровано у таблицю indicators за допомогою таблиць ref_indicators та bri_indicator.

Для побудови статистичної вибірки для проведення регресійного аналізу показника NPL було використано імпортовані дані з таблиці «розподіл кредитів, наданих фізичним та юридичним особам у національній та іноземній валютах, та розміру кредитного ризику за класами боржника відповідно до Положення № 351 (у розрізі банків)». Їх було реформатовано у представлення даних (view), дані з якого потім було вивантажено у Excel за допомогою відповідного GET-запиту.

У Excel були прораховані значення показників NPL по датам у розрізі банків та національної/ іноземної валюти. Показники було за формулою 6.1.

$$NPL = \frac{CP_I + CP_C}{CP}, \tag{6.1}$$

де CP_I – це кредитний портфель фізичних осіб 5-го класу (найнижчий рейтинг кредитоспроможності);

CP_C – це кредитний портфель юридичних осіб 10-го класу (найнижчий рейтинг кредитоспроможності);

CP – це обсяг кредитного портфелю.

7.2 Регресійний аналіз

Далі було здійснено лінійний однофакторний регресійний аналіз показника NPL по банківській системі за кредитним портфелем в національній валюті в його залежності від курсу долара США встановленого НБУ на відповідну дату з довірчим інтервалом 95%. Для цього було задіяно стандартний інструмент Excel проведення регресійного аналізу.

Отримані результати представлено на рисунку 7.1.

ВИВЕДЕННЯ ПІДСУМКІВ

Регресійна статистика	
Множинний R	0,411792671
R-квадрат	0,169573204
Нормований R-квадрат	0,159321022
Стандартна помилка	0,05914456
Спостереження	83

Дисперсійний аналіз

	df	*SS*	*MS*	*F*	*Значимість F*
Регресія	1	0,057858944	0,057858944	16,54020512	0,000109822
Залишок	81	0,283344399	0,003498079		
Разом	82	0,341203343			

	Коефіцієнти	*Стандартна помилка*	*t-статистика*	*P-Значення*	*Нижні 95%*	*Верхні 95%*
Y-перетин	0,569954551	0,039587547	14,39731909	4,86349E-24	0,491187749	0,648721352
USD	-0,005221015	0,001283762	-4,0669651	0,000109822	-0,007775299	-0,002666732

Рисунок 7.1 – Результати регресійного аналізу показника NPL по банківській системі за кредитним портфелем в національній валюті

Значимість F становила 0,0001, що менше за рівень значимості, що вказує на статистичну значущість досліджуваного фактору.

Проте, значення коефіцієнту детермінації R-квадрат вийшло досить низьким. Однією з причин цього стало встановлення Національним Банком України фіксованого курсу долара США протягом достатньо тривалого періоду.

В зв’язку з цим було прибрано записи для цього періоду з вибірки та повторно проведено регресійний аналіз, результати якого наведено на рисунку 7.2.

ВИВЕДЕННЯ ПІДСУМКІВ						
Регресійна статистика						
Множинний R	0,476355598					
R-квадрат	0,226914656					
Нормований R-квадрат	0,214445537					
Стандартна помилка	0,05827745					
Спостереження	64					
Дисперсійний аналіз						
	df	*SS*	*MS*	*F*	*Значимість F*	
Регресія	1	0,061805607	0,061805607	18,19813135	6,92483E-05	
Залишок	62	0,210568192	0,003396261			
Разом	63	0,272373799				
	Коефіцієнти	*Стандартна помилка*	*t-статистика*	*P-Значення*	*Нижні 95%*	*Верхні 95%*
Y-перетин	0,612211093	0,045315874	13,50985949	3,91E-20	0,521625951	0,702796235
USD	-0,006566977	0,001539402	-4,265926787	6,92483E-05	-0,009644198	-0,003489756

Рисунок 7.2 – Результати регресійного аналізу показника NPL по банківській системі за кредитним портфелем в національній валюті за відкоригованою вибіркою

В результаті виконаних змін, значення R-квадрат збільшилось до 0,227, що покращило точність формули регресії.

На рисунку 7.3. можна побачити графік залежності показника NPL від курсу долара США по історичним даним та прогнозованим даним за формулою регресії.

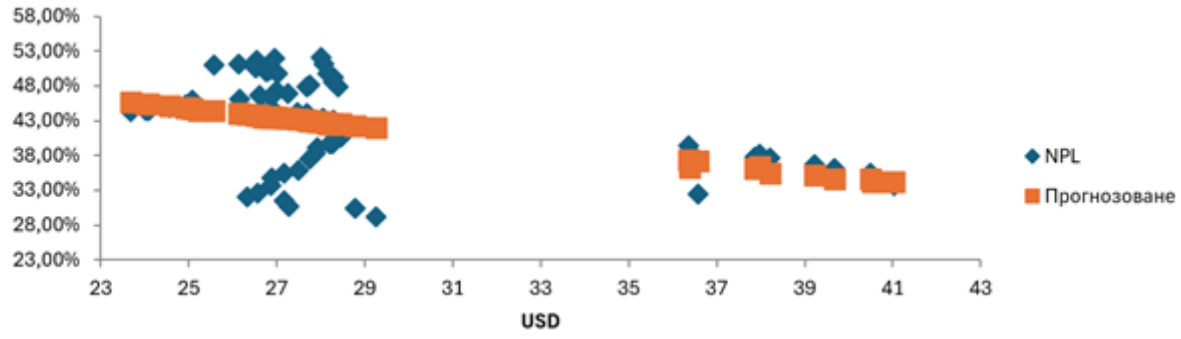

Рисунок 7.3 – Графік залежності показника NPL від курсу долара США по історичним даним та прогнозованим даним за формулою регресії

З графіку можна побачити, що зі зростанням курсу долара США прогнозований NPL зменшується.

Також, можна виділити, що регресія більш точно прогнозує дані в зоні високих значень курсу.

Слід зазначити, що у бюджеті на 2025 рік Міністерство фінансів України заклало курс долара США в розмірі 45 грн [20]. Враховуючи вищезазначене, для прогнозу значень NPL на 2025 рік можна застосувати формулу регресії 7.2:

$$NPL = 0{,}6122 - 0{,}0066 * CURS_{USD}, \quad (7.2)$$

де $CURS_{USD}$ – це значення курсу долара США на відповідну дату.

Далі було проведено регресійний аналіз кредитного портфелю в національній валюті по АТ "СЕНС БАНК".

Результати наведено на рисунку 7.4.

ВИВЕДЕННЯ ПІДСУМКІВ						
Регресійна статистика						
Множинний R	0,824393274					
R-квадрат	0,67962427					
Нормований R-квадрат	0,67445692					
Стандартна помилка	0,058338494					
Спостереження	64					
Дисперсійний аналіз						
	df	*SS*	*MS*	*F*	*Значимість F*	
Регресія	1	0,447621965	0,447621965	131,5227743	5,76E-17	
Залишок	62	0,211009553	0,00340338			
Разом	63	0,658631519				
	Коефіцієнти	*Стандартна помилка*	*t-статистика*	*P-Значення*	*Нижні 95%*	*Верхні 95%*
Y-перетин	-0,355117441	0,045363342	-7,828291075	7,86E-11	-0,445797469	-0,264437414
USD	0,017672878	0,001541015	11,4683379	5,76E-17	0,014592433	0,020753322

Рисунок 7.4 – Результати регресійного аналізу показника NPL по АТ "СЕНС БАНК" за кредитним портфелем в національній валюті за відкоригованою вибіркою

Коефіцієнт детермінації R-квадрат у цьому випадку склав майже 68%, що вказує на високу статистичну значущість фактору курсу долара США для прогнозу показника NPL по банку.

На рисунку 7.5. зображено графік залежності показника NPL від курсу долара США для АТ "СЕНС БАНК".

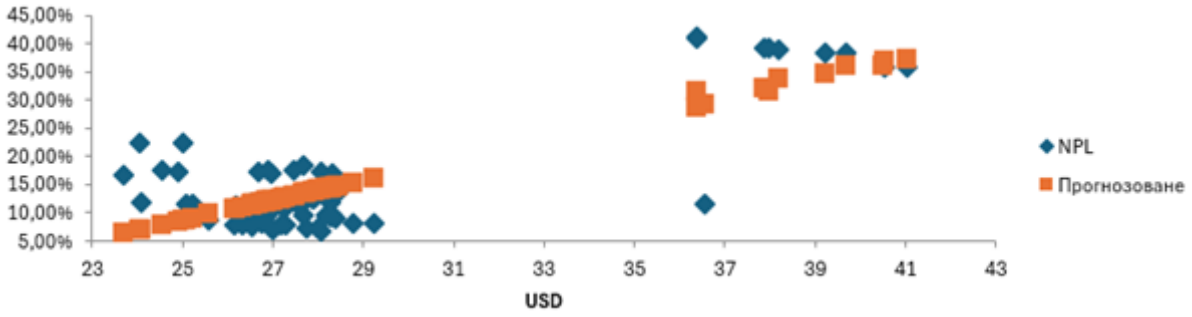

Рисунок 7.5 – Графік залежності показника NPL від курсу долара США по історичним даним та прогнозованим даним за формулою регресії для АТ "СЕНС БАНК"

На графіку можна побачити, що історичні значення NPL досить наближені до розрахованих по формулі регресії. На відміну від показника по банківській системі, значення NPL у АТ "СЕНС БАНК" зростають зі зростанням курсу долара США, що вказує на необхідність внесення змін до політики роботи по управлінню непрацюючими кредитами.

Формула 6.3 регресії для АТ "СЕНС БАНК" виглядає наступним чином:

$$NPL = -0{,}3551 + 0{,}0177 * CURS_{USD}, \quad (6.3)$$

де $CURS_{USD}$ – це значення курсу долара США на відповідну дату.

Висновки до розділу 7

У дослідженні було продемонстровано можливості системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку на прикладі регресійного аналізу показника частки непрацюючих кредитів (NPL). Цей показник є важливим інструментом оцінки якості кредитного портфелю та використовується Національним Банком України для оцінювання кредитних ризиків банківської системи.

Проведення регресійного аналізу на основі залежності NPL від курсу долара США дозволило виявити статистично значущі взаємозв'язки між цими

показниками, хоча низький коефіцієнт детермінації на початковому етапі дослідження вказав на необхідність уточнення вибірки. Після вилучення періодів із фіксованим курсом долара США точність моделей покращилась.

Для банківської системи було отримано формулу регресії, яка показує зменшення NPL зі зростанням курсу долара США. Це може бути обумовлено структурними особливостями портфелю та впливом економічних факторів.

Для АТ "СЕНС БАНК" було виявлено протилежну тенденцію: NPL зростає зі збільшенням курсу долара США.

8 СТАРТАП-ПРОЄКТ

Цей розділ присвячений ринковому дослідженню реалізації проєкту як стартапу та його конкурентоспроможності з наявними конкурентами.

8.1 Опис ідеї проєкту

По-перше, сформульована ідея стартап-проєкту та вигоди, які користувач зможе отримати з системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку.

Таблиця 8.1 – Опис ідеї стартап-проєкту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Розробка системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфелю банку	Завантаження інформації з робочих файлів в базу даних	Систематизоване, зручне збереження інформації в централізованому місці, звідки далі можна її аналізувати різними інструментами
	Автоматична нормалізація даних для поширених форматів звітності	Нормалізація даних виконується системою автоматично для великої кількості шаблонів файлів, без потреби у залученні спеціалістів дата-аналітиків
	Виконання аналізу кредитних портфельних ризиків у BI-системі Metabase	Широкий інструментарій, для використання якого у системі вже присутні усі налаштування бази даних

Проєкт має мати вирішувати реальні проблеми, чимось виділятися посеред наявних рішень на ринку. Для аналізу переваг було проведено порівняння слабких та нейтральних характеристик ідеї проєкту.

Таблиця 8.2 – Порівняльний аналіз сильних, слабких та нейтральних характеристик ідеї проєкту

№ п/п	Техніко-економічні характеристики ідеї	Ринкові конкуренти				W	N	S
		Мій проєкт	Без-коштовні BI-системи	Power BI	Рішення на замовлення			
1.	Надання джерела упорядкованих даних	+	–	–	+		+	
2.	Конфіденційність та локальне збереження даних	+	– (деякі)	+	+			+
3.	Необхідний інструментарій для аналізу	+	+	+	+			+
4.	Переваги у непов’язаних з аналітикою портфельних кредитних ризиків сферах	– (однакові з Metabase Open Source)	–	+	– (за додаткову оплату)	+		
5.	Низька вартість	+ –	+	+ –	–			+

8.2 Технологічний аудит ідеї проєкту

Для того, щоб максимізувати можливість виходу стартап-проєкту на ринок, він має мати в доступності усі технології, що забезпечать виконання поставлених вимог. Для цього було проведено аналіз технологічної здійсненності проєкту.

Таблиця 8.3 – Аналіз технологічної здійсненності проєкту.

№ п/п	Ідея проєкту	Технології її реалізації	Наявність технологій	Доступність технологій
1.	Структурований інтерфейс імпорту робочих файлів	JavaScript + Tabulator	Наявні	Доступні безкоштовно
2.	Сховище даних з реалізацією трансформування і нормалізації даних	PostgreSQL	Наявна	Доступна безкоштовно
3.	Обмін даними тонкого клієнта з сервером	Spring Framework + Thymeleaf	Наявні	Доступні безкоштовно
4.	Розширені можливості роботи з базою даних	Spring Framework + JDBC	Наявні	Доступні безкоштовно
5.	Визначення кодування файлів засобами штучного інтелекту	Apache Tika	Наявна	Доступна безкоштовно
6.	Файловий сервер, доступний з будь-якої оперативної системи	Linux Ubuntu Server + Samba	Наявні	Доступні безкоштовно

Усі необхідні технології для реалізації є доступними без оплати. Це позитивно вплинуло на процес розробки та відкрило можливість проводити більш гнучку цінову політику.

8.3 Аналіз ринкових можливостей запуску стартап-проекту

Ринок і присутні конкуренти часто диктують умови для успішності виходу стартапу в продаж. Виконаємо аналіз ринкових можливостей запуску стартап-проєкту

Таблиця 8.4 – Попередня характеристика потенційного ринку стартап-проєкту

№ п/п	Показники стану ринку	Характеристика
1	Кількість головних гравців, од	5
2	Загальний обсяг продаж, грн/ум.од	~$15 млн.
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Невелика кількість потенційних клієнтів, висока конкуренція
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні
6	Середня норма рентабельності в галузі (або по ринку), %	47%

Конкуренція не має стати серйозною проблемою для стартапу.

Таблиця 8.5 – Характеристика потенційних клієнтів стартап-проєкту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
	Потреба в зручній та швидкій системі нормалізації та відображення даних для аналізу ризиків кредитного портфелю	Комерційні банки	– кількість клієнтів; – фінансові можливості.	– зручність та швидкість застосунку; – підтримка різних форматів даних;

Клієнтами стартап-проєкту мають стати банки України різних масштабів та фінансових можливостей. Важливо задовольнити усім можливим вимогам клієнтів, щоб система могла стати в нагоді користувачам.

8.4. Факторний аналіз

Наступним кроком є аналіз ринкового середовища, включно із аналізом загроз та можливостей. Результат аналізу загроз наведено у табл. 8.6, а можливостей – у табл. 8.7

Таблиця 8.6 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Недостача обчислювальних потужностей	При спробах введення завеликої кількості даних система може не впоратись з їх обробкою через недостатню потужність.	Встановлення потужннішого серверного обладнання, що буде відповідати потребі клієнта в його запитах.
2	Хакерська атака	При хакерській атаці дані, що використовує банк можуть бути скомпрометовані.	Для захисту локальної мережі банку в якій працює застосунок необхідно використовувати файрвол та інші засоби захисту від віддаленого доступу, щоб запобігти можливі витоки даних.

Таблиця 8.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Збільшення кількості банків в Україні	Так як кількість клієнтів обмежена, кожний новий потенційний клієнт є важливим. Мій проєкт має стати в нагоді новому банку, у якого ще не налагоджені процеси аналітики ризиків та потоки даних і який не має великої кількості коштів.	Індивідуальна пропозиція новому банку, яка може включати скоректований план прайсингу, що спростить рішення о інвестиції в проєкт для банку
2	Зміни в вимогах	Ускладнення процесів аналізу кредитних портфельних ризиків	Нова хвиля маркетингової компанії, яка

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
	розрахунків показників НБУ	робить проблему, яку прагнне вирішити мій стартап-проєкт, ще більш нагальною. Це може збільшити кількість зацікавлених потенційних клієнтів	сфокусується на тому наскільки легко можна дотрматись нових вимог використовуючи мій проєкт

Виділені загрози є важливими і мають бути враховані у реалізації стартап-проєкту. Можливості виходу на ринок створюються завдяки потенційному збільшенню кількості банків в Україні та змінам в вимогах розрахунків показників НБУ.

8.5 Аналіз конкуренції

Таблиця 8.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Вказати тип конкуренції – чиста конкуренція	Компанії встановлюють власні ціни на свої послуги й не можуть контролювати ціни на усьому ринку.	Вибір оптимальної вартості для свого продукту.
2. За рівнем конкурентної боротьби – національний	Ринок систем для аналізу ризиків присутній лише на національному рівні через особливості української регуляторної	Необхідно цілитись лише на локальний ринок та приділяти більше уваги особливостям роботи з місцевою фінансовою системою.

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
	фінансової системи та через занадто високу вартість послуг від міжнародних компаній.	
3. За галузевою ознакою – внутрішньогалузева	Конкуренція не виходить за рамки систем для аналізу кредитних ризиків для банків.	Концентрація на вузькій направленості надає можливість розробки якісного спеціалізованого рішення.
4. Конкуренція за видами товарів – товарно-видова	Існують інші компанії-конкуренти зі схожими продуктами, які виконують ті самі функції.	Для того, щоб виділятися поміж конкурентів необхідно надавати зручніше, швидше та якісніше рішення за розумну ціну.
5. За характером конкурентних переваг – нецінова	Здебільшого, вибір систем аналізу ризиків відбувається за якістю послуг, швидкістю роботи, зручнішим інтерфейсом тощо.	Впровадження нових технологій та пошук нестандартних рішень
6. За інтенсивністю – не марочна	На ринку немає відомих компаній з гучним іменем, тому вибір системи відбувається за фактичні технічні переваги та функціонал продукту	Необхідно вигідно виділятися саме якістю функціоналу та унікальними можливостями продукту

Конкуренція є дуже обмеженою для такої системи, проте і кількість можливих клієнтів також обмежена.

Таблиця 8.9 – Аналіз конкуренції в галузі за М. Портером

	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
Складові аналізу	Прямих конкурентів немає.	– CS; – UNITY-BARS; – Lime Systems.	Немає постачальників.	– швидкість роботи системи; – підтримка різних форматів даних; – своєчасні оновлення системи при змінах законодавства.	– краще співвідношення ціни та якості; – звичність у використанні старих систем;
Висновки:	Ні.	Потенційними конкурентами є компанії, що надають подібні рішення на замовлення	Ні.	Так, від вимог клієнта функціонал системи може змінюватися та адаптуватися.	Клієнти можуть надавати перевагу товарам-замінникам через їх більшу звичність та співвідношення ціни-якості

Відсутність прямих конкурентів спрощує вихід на ринок і дозволяє сфокусуватись на непрямих конкурентах.

Таблиця 8.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	Впровадження функціоналу за потребою користувача	При впровадженні додаткового функціоналу на основі аналізу потреб користувачів та їх прямого запиту надає перевагу на уніфікованими продуктами із фіксованими можливостями
2	Простота адаптації до використання	Можливість швидко встановити та налаштувати систему на серверах клієнта допомагає зекономити час та ресурси, а універсальність продукту для різних типів даних дозволяє оперувати великою кількістю різноманітної інформації

Виділені фактори конкурентоспроможності можуть покращити позицію стартап проєкту на ринку.

8.6 Сильні та слабкі сторони проєкту

Таблиця 8.11 – Порівняльний аналіз сильних та слабких сторін проєкту.

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з … (назва підприємства)						
			–3	–2	–1	0	+1	+2	+3
1	Впровадження функціоналу за	13						+	

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з … (назва підприємства)						
	потребою користувача								
2	Простота адаптації до використання	16				+			

Впровадження функціоналу за потребою користувача може надати суттєвих конкурентних переваг для стартап проєкту.

Таблиця 8.12 – SWOT-аналіз стартап-проєкту

Сильні сторони: – швидке встановлення та налаштування системи; – універсальність в роботі з різними даними; – робота системи в локальній мережі клієнтів; – регулярне оновлення системи при зміні законодавства.	Слабкі сторони: – використання системи потребує додаткового навчання співробітників
Можливості: – можливість впровадження додаткового функціоналу;	Загрози: – хакерські атаки.

Зазначені сильні сторони мають вирізнити стартап проєкт серед наявних на ринку рішень

8.7 Впровадження стартап-проєкту

Таблиця 8.13 – Альтернативи ринкового впровадження стартап-проєкту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Індивідуальні пропозиції компаніям	Середня	3 місяці
2	Презентація проєкту на тематичній конференції	Середня	2 місяці

Виділені альтернативи допоможуть привернути увагу потенційних клієнтів до стартап проєкту.

Таблиця 8.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Комерційні банки України	Висока	Високий	Низька	Складна
Які цільові групи обрано: комерційні банки України.					

Проєкт розроблявся до продажу комерційним банкам України.

Таблиця 8.15 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Створення базової версії продукту та її тестування для обмеженого	Акцентуалізація на зворотному зв’язку від користувачів та адаптація	Широкі можливості роботи з даними, локальне збереження необхідної інформації та швидкість й зручність	Стратегія спеціалізації

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
	кола користувачів	доопрацювання системи перед запуском	роботи з системою	

Спеціалізація має стати найкращою стратегією розвитку стартап проєкту.

Таблиця 8.16 – Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1	Так	Проєкт націлений на нових споживачів	Ні	Стратегія лідера

Ідея проєкту є новою для українського FinTech ринку.

Таблиця 8.17 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	Швидкість та зручність роботи з системою	Стратегія спеціалізації	Система працює порівняно швидко та надає користувачам зручні варіанти налаштування при роботі з системою	Комфорт
2	Робота з різними видами даних	Стратегія спеціалізації	Система працює з основними шаблонами даних та типами файлів, що дозволяє обробляти більшість можливих джерел інформації	Універсальність
3	Локальне збереження необхідних даних та налаштувань	Стратегія спеціалізації	Оброблені дані зберігаються на локальному сервері клієнта, що дозволяє йому зручно ними оперувати та надійно їх захищати	Локальність

Система має бути комфортною та універсальною для локального ринку банків України. Вона має задовольняти усім можливим вимогам та побажанням користувачів.

Таблиця 7.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1	Потреба у швидкому надані доступу до системи	Систему можна встановити та почати використовувати досить швидко і його опанування не займає багато часу	Простота
2	Потреба у зручності та швидкості роботи з системою	Система має зручний інтерфейс, який можна налаштовувати під власні потреби. При роботі з системою значно зростає швидкість обробки даних та їх компонування	Зручність
3	Потреба у безпечності даних користувачів та документообігу	Контроль доступу до функціоналу системи, повністю локалізована структура системи, система не зберігає та не передає жодної інформації в інтернет	Безпека даних

У специфіці аналітики кредитних портфельних ризиків можна побачити велику кількість проблем, що породжують виділені проблеми.

Таблиця 8.19 – Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>I. Товар за задумом</td><td colspan="3">Базова потреба: оптимізація аналізу кредитних ризиків та підвищення ефективності роботи аналітиків банків.
Основна вигода: автоматизація обробки даних, структуроване сховище даних, інтегровані засоби візуалізації</td></tr>
<tr><td rowspan="6">II. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх /Тл/Е/Ор</td></tr>
<tr><td>Складність використання системи</td><td>Нм</td><td>Ор</td></tr>
<tr><td>Швидкість обробки даних</td><td>М</td><td>Тх</td></tr>
<tr><td colspan="3">Якість: відповідає вимогам банківської галузі щодо безпеки, точності та надійності.</td></tr>
<tr><td colspan="3">Пакування: цифрова інфраструктура (установка на сервер, документація, навчальні матеріали).</td></tr>
<tr><td colspan="3">Марка: “BRI” – Bank Risk Intelligence</td></tr>
<tr><td rowspan="2">III. Товар із підкріпленням</td><td colspan="3">До продажу: готова система</td></tr>
<tr><td colspan="3">Після продажу: технічна підтримка, оновлення програмного забезпечення, адаптація під специфічні вимоги замовників.</td></tr>
</table>

У таблиці 8.19 було описано три рівні моделі товару.

Таблиця 8.20 – Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	до 65$ на користувача в місяць	300000$ за систему + 100000$ за рік технічної підтримки	4 – 150 млн. $ в рік	Від 5000$ за систему + 5000$ за рік технічної підтримки до 30000$ за систему +15000$ за рік технічної підтримки

Ціни конкурентів дуже сильно різняться, тому було виділено такі верхню та нижню межу цін на придбання системи. Такий діапазон цін має забезпечити велику кількість користувачів та гарний прибуток.

Таблиця 8.21 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Пошук нових клієнтів	Пошук нових клієнтів шляхом контактування зацікавлених осіб; формування попиту; дослідження економічного стану галузі для корекції цін на послуги.	0 … n, де n – це кількість комерційних банків України	Особисто запропонувати систему потенційному клієнту без посередників

Кількість потенційних клієнтів обмежена кількістю комерційних банків України. Це відносно мала кількість зацікавлених клієнтів, тому оптимальним маркетинговим варіантом є особисті пропозиції для банків.

Таблиця 8.22 – Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Фокус на покращенні рішення	Збір відгуків клієнтів, канал комунікації пропозицій, скарг та побажань	Швидкість та простота в обробці даних, автоматична структуризація сховища даних, інтеграція з сучасною BI-системою для візуалізації	Зацікавити потенційних клієнтів шляхом рекламування ідеї, переваг та можливостей проєкту.	Практичне специфіковане звернення з демонстрацією переваг проєкту та можливостей вирішення реальних практичних проблем

Система буде мати регулярні оновлення, що будуть задовольняти новим вимогам та зможуть привабити нових клієнтів.

Висновки до розділу 8

У розділі було проведено комплексний аналіз можливостей реалізації проєкту системи підтримки інтерактивного аналізу портфельних ризиків

кредитного портфеля банку як стартапу. Досліджено технічну здійсненність, ринкові умови та конкурентоспроможність продукту.

Ідея проєкту є інноваційною для національного ринку, оскільки об'єднує автоматизацію нормалізації даних, інтеграцію з BI-системами та локальне збереження інформації, що відповідає потребам українських банків.

Технологічна база проєкту сформована на основі відкритих технологій (JavaScript, PostgreSQL, Spring Framework, Apache Tika), які забезпечують його доступність та гнучкість у розробці, що дозволяє мінімізувати витрати.

Ринкові можливості свідчать про перспективність запуску продукту, оскільки ринок систем для аналізу кредитних ризиків зростає, а конкуренція має національний характер із переважно локальними гравцями.

SWOT-аналіз виявив ключові сильні сторони проєкту, такі як зручність роботи, локальне збереження даних та універсальність. Водночас до основних ризиків належать потенційні хакерські атаки та потреба в навчанні користувачів.

Конкурентна стратегія орієнтована на створення високоякісного спеціалізованого рішення для українських банків із фокусом на швидкість, зручність і безпеку системи.

Запропонований стартап-проєкт має значний потенціал для виходу на ринок і здатен задовольнити актуальні потреби комерційних банків, зокрема в умовах посилення регуляторних вимог та зростання обсягів оброблюваних даних. Обрані стратегічні підходи сприятимуть формуванню конкурентних переваг та залученню клієнтів.

ВИСНОВКИ

У межах цієї роботи було розглянуто процес створення системи підтримки інтерактивного аналізу портфельних ризиків кредитного портфеля банку. Сформовано концепцію, розроблено програмну реалізацію та виконано аналіз можливостей запуску системи як стартапу. Ключові результати, досягнуті під час дослідження, викладено нижче.

У вступі обґрунтовано актуальність дослідження, яка пов’язана зі зростанням обсягів кредитних портфелів банків і необхідністю покращення управління ризиками у банківській сфері. Зазначено, що створення інноваційної системи підтримки аналізу є важливим для підвищення ефективності діяльності банківських установ України.

Перший розділ було присвячено вивченню актуальних проблем, пов’язаних із аналізом портфельних кредитних ризиків. Зокрема, визначено складнощі в інтеграції різнорідних даних та обмежені можливості існуючих рішень для візуалізації. Запропоновано основні принципи, які повинні забезпечувати ефективність аналітики, такі як точність, безпеку і зручність використання.

У другому розділі детально описано ключові задачі розробки системи, включно з нормалізацією даних, створенням багаторівневої структури сховища даних, забезпеченням інтеграції з BI-системами та розробкою зручного інтерфейсу для користувачів. Обґрунтовано структуру системи, яка базується на трирівневій архітектурі, що сприяє підвищенню масштабованості та продуктивності.

Проведено аналіз доступних на ринку рішень, включаючи безкоштовні BI-системи, такі як Metabase і Knowage, платформи Power BI та кастомні рішення від українських компаній. Оцінено переваги та недоліки кожного підходу, що дозволило визначити оптимальний шлях для реалізації системи.

У четвертому розділі було виконано проєктування системи шляхом побудови діаграми сценаріїв використання, схеми документообігу, структурної схеми, діаграми послідовності та блок-схеми імпорту файлів.

У п’ятому розділі було вибрано стек технологій для розробки. Бек-енд реалізовано за допомогою Java та Spring Framework, що забезпечило надійність і масштабованість. Для фронт-енд частини використано JavaScript та бібліотеку Tabulator, яка дозволяє ефективно працювати з великими наборами даних. Як базу даних обрано PostgreSQL, що підтримує складну аналітику та інтеграцію з іншими компонентами.

Розділ 6 описує етапи програмної розробки, включаючи створення репозиторіїв, сервісів, контролерів та файлового сервера, логіку імпорту даних та виконання аналізу ризиків, що дозволяє автоматизувати процеси аналітики.

Розглянуто процес розгортання системи у середовищі користувача, включаючи налаштування серверної інфраструктури, інтеграцію з BI-системою та базою даних. Описано можливості розгортання системи залежно від серверних пристроїв та методів доступу до локальної мережі.

Виконано демонстрацію роботи системи на тестових даних. Продемонстровано можливості системи щодо аналізу кредитних портфельних ризиків, зокрема оцінку показника NPL для банківської системи та окремого банку.

У фінальному розділі проаналізовано можливості запуску проєкту як стартапу. Проведено технологічний аудит і SWOT-аналіз, які підтвердили конкурентоспроможність ідеї. Розглянуто ринкові умови, що свідчать про зростаючий попит на системи аналізу ризиків. Сформульовано рекомендації щодо розвитку проєкту та стратегій залучення клієнтів.

Результатом виконаної роботи стала повноцінна система підтримки аналізу портфельних ризиків кредитного портфеля банку, яка відповідає актуальним потребам українських банківських установ. Запропоноване рішення забезпечує:

– автоматизацію процесів обробки та нормалізації даних;

– інтеграцію з BI-системами для візуалізації аналітичної інформації;

– зручність використання завдяки адаптивному інтерфейсу та функціональності.

Створення системи є економічно виправданим і перспективним у контексті її комерціалізації як стартапу. Застосування відкритих технологій забезпечило мінімізацію витрат на розробку, а використання сучасних підходів до аналізу ризиків сприяє підвищенню точності й ефективності роботи банківських ризик-аналітиків.

СПИСОК ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. ІНСТРУКЦІЯ про порядок регулювання діяльності банків в Україні затверджена Постановою правління НБУ № 638 від 28.08.2001 (зі змінами та доповненнями). [Електронний ресурс] – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/z0841-01#Text.

2. Мінін П., Букасов М. Автоматизація аналізу портфельних кредитних ризиків банку. *V Міжнародна науково-практична конференція молодих вчених та студентів «Інженерія програмного забезпечення і передові інформаційні технології SoftTech-2023»*. [Електронний ресурс] – Режим доступу до ресурсу: https://drive.google.com/file/d/1racc22TBKkFFNzBRSzOrePKpFfbnGDJ1/view.

3. About Looker Studio Pro [Електронний ресурс] – Режим доступу до ресурсу: https://support.google.com/looker-studio/answer/13715508?hl=en.

4. Офіційний сайт Tableau Public [Електронний ресурс] – Режим доступу до ресурсу: https://www.tableau.com/products/public.

5. Закупівля Технічна підтримка АБС "БАРС Millennium"? - 72261000-2 від Національний Банк України [Електронний ресурс] – Режим доступу до ресурсу: https://tender.uub.com.ua/tender/UA-2023-12-29-004244-a/.

6. What is Java technology and why do I need it? [Електронний ресурс] – Режим доступу до ресурсу: https://www.java.com/en/download/help/whatis_java.html.

7. Spring Framework Overview [Електронний ресурс] – Режим доступу до ресурсу: https://docs.spring.io/spring-framework/reference/overview.html.

8. Spring in Action, Sixth Edition – Крейг Уоллс, 2022 р.

9. Офіційний сайт Metabase [Електронний ресурс] – Режим доступу до ресурсу: https://www.metabase.com/Head First.

10. Офіційний сайт BIRT [Електронний ресурс] – Режим доступу до ресурсу: https://eclipse-birt.github.io/birt-website/

11. Офіційний сайт KNOWAGE [Електронний ресурс] – Режим доступу до ресурсу: https://www.knowage-suite.com/site/

12. Web MVC framework [Електронний ресурс] – Режим доступу до ресурсу: https://docs.spring.io/spring-framework/docs/3.2.x/spring-framework-reference/html/mvc.html

13. Spring Data JPA. URL: https://spring.io/projects/spring-data-jpa

14. Патерни проєктування / Е.Фрімен, Е. Робсон, Б. Берт, С. Кеті., 2020. – 672 с.

15. Declarative REST Client Електронний ресурс] – Режим доступу до ресурсу: https://cloud.spring.io/spring-cloud-netflix/multi/multi_spring-cloud-feign.html

16. Database table relationships. [Електронний ресурс] – Режим доступу до ресурсу: https://www.metabase.com/learn/databases/table-relationships

17. Обсяги активних операцій та частка непрацюючих активів в цілому по системі [Електронний ресурс] – Режим доступу до ресурсу: https://bank.gov.ua/files/stat/NPL_AO_2024-10-01.xlsx

18. Перелік довідників, які використовуються для формування показників статистичної звітності [Електронний ресурс] – Режим доступу до ресурсу: https://bank.gov.ua/ua/statistic/nbureport/registers

19. Наглядова статистика [Електронний ресурс] – Режим доступу до ресурсу: https://bank.gov.ua/ua/statistic/supervision-statist

20. Який курс долара прогнозують на 2025 рік — проєкт бюджету [Електронний ресурс] – Режим доступу до ресурсу: https://suspilne.media/835985-akij-kurs-dolara-prognozuut-na-2025-rik-proekt-budzetu/