Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра програмного забезпечення

Бакалаврська дипломна робота

на тему: «Розробка програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти»

Виконав: студент VI курсу
групи 2ПІ-20б
спеціальності
121 – Інженерія програмного забезпечення
(шифр і назва напряму підготовки, спеціальності)
Гладун І.М.
(прізвище та ініціали)

Керівник: к.т.н., доц. каф. ПЗ Ткаченко О.М.
(прізвище та ініціали)

Рецензент:. к.т.н., доц. каф. ОТ Богомолов С.В.
(прізвище та ініціали)

Допущено до захисту

Зав. кафедри ПЗ _____
«__10__» __червня_____ 2024 р.

ВНТУ – 2024

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення
Рівень вищої освіти перший (бакалаврський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 121 «Інженерія програмного забезпечення»
Освітньо-професійна програма «Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
О. Н. Романюк
«12» березня 2024 р.

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Гладуну Івану Миколайовичу

1. Тема роботи – розробка програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти.

Керівник роботи: Ткаченко Олександр Миколайович, к.т.н., доцент кафедри ПЗ, затверджені наказом вищого навчального закладу від 11 березня 2024 р. № 80.

2. Строк подання студентом роботи 3 червня 2024.

3. Вихідні дані до роботи: система управління базами даних – PostgreSQL; мова запитів – SQL; інструменти для розробки веб-інтерфейсу: HTML, CSS, JavaScript; мова програмування: Java (з використанням фреймворків Spring Boot, Hibernate, Thymeleaf); вихідні дані – наявний набір історичних даних про фінансові інструменти.

4. Зміст текстової частини: огляд технологій створення програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти та постановка задач дослідження; проектування структури та алгоритмів програмного продукту; розробка програмних засобів веб-платформи для оцінювання прибутковості інвестицій з використанням аналітичних інструментів та інтеграція готових шаблонів, включаючи варіантний аналіз і обґрунтування вибору засобів для реалізації програмного забезпечення, розробку функцій для завантаження фінансових даних та додання їх до бази даних, розрахунку основних фінансових показників.

5. Перелік ілюстративного матеріалу: сфери застосування програмного забезпечення для фінансового аналізу; основні етапи процесу оцінки інвестицій; методи розрахунку фінансових показників; новітні моделі прогнозування прибутковості; інтерфейс веб-додатку для фінансового аналізу.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата			
		Завдання видав		Завдання прийняв	
1-4	Ткаченко О.М. к.т.н., доцент каф ПЗ	13.03.24		03.06.24	

7. Дата видачі завдання _____ 13.03.24_____

КАЛЕНДАРНИЙ ПЛАН

Ч.ч.	Назва етапів бакалаврської дипломної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз розвитку мобільних систем для оцінювання прибутковості у фінансові інструменти	14.03.24 – 22.03.24	Вик.
2	Математичні моделі для оцінювання прибутковості ОВДП	25.03.24 – 19.04.24	Вик.
3	Розробка програмних модулів реалізації для оцінювання прибутковості у фінансові інструменти	22.04.24 – 10.05.24	Вик.
4	Впровадження, тестування та розробка документації	13.05.24 – 24.05.24	Вик.
5	Оформлення матеріалів до захисту БДР	30.05.24 – 03.06.24	Вик.

Студент _____ **І. М. Гладун**
(підпис) (ініціали та прізвище)

Керівник бакалаврської дипломної роботи _____ **О. М. Ткаченко**
(підпис) (ініціали та прізвище)

АНОТАЦІЯ

УДК 00492

Гладун І.М. Розробка програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти: бакалаврська дипломна робота зі спеціальності 121 – інженерія програмного забезпечення, освітня програма – інженерія програмного забезпечення. Вінниця: ВНТУ, 2024. 52 с. (до висновків)

На укр. мові. Бібліогр. : 25 назв ; рис. : 19; табл. 3.

У бакалаврській дипломній роботі проведено комплексний аналіз існуючих методів і технологій для оцінювання прибутковості інвестицій у фінансові інструменти. Визначено мету дослідження – спрощення процесу оцінювання прибутковості інвестицій за допомогою спеціалізованого програмного забезпечення.

Розроблено алгоритми та програмні засоби, що дозволяють користувачам без глибоких знань у фінансах легко аналізувати та прогнозувати прибутковість інвестицій. Запропонований підхід включає використання сучасних бібліотек для аналізу даних на **Java**, що підвищує ефективність та точність розрахунків.

Отримані результати можуть бути використані для створення інтуїтивно зрозумілих та зручних у використанні систем для фінансового аналізу, що сприятимуть оптимізації інвестиційних рішень у різних сферах діяльності.

Ключові слова: програмне забезпечення, оцінювання прибутковості, інвестиції, фінансові інструменти, аналіз даних.

ABSTRACT

UDC 00492

I.M. Gladun Development of software for evaluating the profitability of investments in financial instruments: bachelor's thesis on the specialty 121 – software engineering, educational program – software engineering. Vinnytsia: VNTU, 2024. 52 p. (to conclusions).

In Ukrainian speech Bibliogr. : 25 titles; Fig. : 19; table 3.

The bachelor's thesis provides a comprehensive analysis of existing methods and technologies for evaluating the profitability of investments in financial instruments. The aim of the study is to simplify the process of evaluating investment profitability using specialized software.

Algorithms and software tools have been developed that allow users without deep financial knowledge to easily analyze and forecast investment profitability. The proposed approach includes the use of modern data analysis libraries in **Java**, which enhances the efficiency and accuracy of calculations.

The results obtained can be used to create intuitive and user-friendly systems for financial analysis, promoting the optimization of investment decisions in various fields.

Keywords: software, profitability evaluation, investments, financial instruments, data analysis.

ЗМІСТ

ВСТУП

Обґрунтування вибору теми дослідження

Значна потреба в інвестиціях є однією з ключових складових економічного розвитку будь-якої країни, у тому числі і України [1]. Це охоплює як державні інвестиції, спрямовані на розвиток інфраструктури та соціальних сфер, так і приватні інвестиції, які стимулюють підприємництво та інновації.

Необхідність ефективного управління інвестиціями постає перед інвесторами як ніколи раніше. Інструменти для оцінювання ризику та прибутковості інвестицій дозволяють їм приймати обґрунтовані рішення та оптимізувати свій інвестиційний портфель.

Швидкий розвиток фінансових технологій, або фінтеху, створює нові можливості для розробки програмних рішень у сфері фінансів. Ці технології спрощують процеси оцінювання та управління інвестиціями, забезпечуючи інвесторів і компанії зручні та ефективні інструменти.

Зростаюча глобалізація фінансових ринків розширює доступ інвесторів до різноманітних фінансових інструментів та можливостей інвестування. Це вимагає від інвесторів більш складних методів оцінювання, які враховують глобальні ризики та можливості [2].

Розробка програмного забезпечення, що дозволяє інвесторам самостійно аналізувати та оцінювати свої інвестиції, важлива для підвищення фінансової грамотності та незалежності. Це сприяє розвитку ефективного інвестиційного середовища та сприяє зростанню фінансової стабільності.

Продовжуючи думку, розробка програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти спростить процес прийняття рішень інвесторами шляхом автоматизації складних фінансових аналізів. Це забезпечить швидкий доступ до потрібної інформації, покращить точність прогнозів і допоможе інвесторам уникнути непередбачених ризиків.

Додатково, програмне забезпечення включає розробку алгоритмів для прогнозування ринкових тенденцій та виявлення потенційних інвестиційних можливостей. Це важливо для забезпечення конкурентної переваги інвесторів на швидкозмінному фінансовому ринку.

Крім того, бакалаврська дипломна робота включає розширення функціоналу для адаптації до специфічних потреб різних типів інвесторів, включаючи як інституційних, так і приватних. Наприклад, можливість аналізувати ризики інвестицій для пенсійних фондів або інвестиційних портфелів фізичних осіб.

Новаторським аспектом цієї роботи може бути інтеграція штучного інтелекту та машинного навчання для автоматизованого прогнозування та оптимізації інвестиційних стратегій. Це дозволить програмі адаптуватися до змінних умов ринку та постійно вдосконалювати свої рекомендації [3].

Мета та завдання дослідження. *Мета роботи* полягає у підвищенні якості управління інвестиційними портфелями шляхом розробки та впровадження програмного забезпечення для оцінювання прибутковості фінансових інструментів.

Відповідно до поставленої мети в бакалаврській дипломній роботі потрібно вирішити такі задачі:

- аналіз методів та інструментів, що використовуються в управлінні інвестиціями, а також виявлення їхніх переваг і недоліків;
- створення програмного продукту, який забезпечить можливість оцінювання прибутковості різних фінансових інструментів, а також аналізу ризиків і вигод;
- проведення тестів програмного забезпечення для перевірки його ефективності, точності та надійності. Валідація здійснюється порівнянням результатів роботи програми з реальними ринковими даними та аналізом відповідності отриманих результатів вимогам інвесторів;

- впровадження розробленого програмного забезпечення на практиці, навчання користувачів його використанню та надання підтримки;
- аналіз результатів використання програмного забезпечення для визначення його впливу на якість управління інвестиціями та досягнення мети дослідження.

Ці завдання спрямовані на створення інструменту, який допоможе інвесторам приймати обґрунтовані рішення та оптимізувати свої інвестиційні стратегії з метою досягнення більш високої прибутковості та зниження ризиків.

Об'єкт дослідження – процес розробки програмного забезпечення для управління інвестиційними портфелями.

Предмет дослідження – методи та засоби розробки програмного забезпечення для оцінювання прибутковості різних фінансових інструментів, а також надання рекомендацій щодо прийняття інвестиційних рішень.

Методи досліджень: методи математичного моделювання для вибору моделі оцінювання прибутковості; методи проектування програмного забезпечення для розробки архітектури системи, алгоритмів та програмного забезпечення; методи математичної статистики для аналізу результатів тестування програмного продукту.

Новизна отриманих результатів

1. **Удосконалено** метод для прогнозування ринкових тенденцій, який покращує точність прогнозів та дозволяє інвесторам приймати більш обґрунтовані рішення. На візміну від існуючих методів, новий підхід враховує додаткові параметри, такі як аномальні зміни на ринку, що забезпечує більш адаптивний і точний аналіз.

2. **Отримав подальшого розвитку** метод оцінювання прибутковості інвестицій, який, на відміну від існуючих, враховує багатофакторний аналіз ризиків та можливостей, зокрема макроекономічні показники, галузеві тренди та специфічні характеристики фінансових інструментів, що дозволило підвищити точність оцінювання прибутковості фінансових інструментів.

Практичне значення отриманих результатів. Розроблено алгоритми та програмне забезпечення, яке надає користувачам зручний та ефективний засіб для оцінювання прибутковості фінансових інструментів, сприяючи прийняттю обґрунтованих рішень. Розроблене програмне забезпечення може бути впроваджене у практику фінансових установ, інвестиційних компаній та приватних інвесторів, підвищуючи ефективність їхньої діяльності та конкурентоспроможність.

Апробація. Результати роботи доповідалися на Міжнародній науково-практичній інтернет-конференції “Молодь в науці: дослідження, проблеми, перспективи” (Вінниця, 2024 р.).

Публікації. За тематикою дослідження опубліковано 1 наукову працю у тезах доповідей Міжнародної науково-практичної інтернет-конференції “Молодь в науці: дослідження, проблеми, перспективи” [4].

1 АНАЛІЗ РОЗВИТКУ МОБІЛЬНИХ СИСТЕМ ДЛЯ ОЦІНЮВАННЯ ПРИБУТКОВОСТІ У ФІНАНСОВІ ІНСТРУМЕНТИ

1.1 Аналіз стану мобільних систем у сфері оцінювання прибутковості інвестицій у фінансові інструменти

Мобільні додатки стають невід'ємною частиною сучасного фінансового ринку. Вони забезпечують доступ до важливої ринкової інформації, що дозволяє інвесторам приймати обґрунтовані рішення. Сучасні мобільні додатки не тільки надають інформацію, але і дозволяють здійснювати аналіз та управління портфелем інвестицій. Основні функції включають реальний час, технічний та фундаментальний аналіз, а також можливість для торгівлі.

Однією з актуальних тенденцій є зростання популярності мобільних додатків, які використовують штучний інтелект (ШІ) та машинне навчання для прогнозування ринкових тенденцій та рекомендацій з інвестування. Такі додатки аналізують великі обсяги даних та навчаються відповідати на зміни на ринку, що допомагає інвесторам приймати кращі рішення.

Однак існують і виклики, з якими стикаються розробники мобільних систем в цій сфері. Одним з них є забезпечення безпеки даних, оскільки мобільні додатки, що працюють з фінансовими даними, повинні дотримуватися високих стандартів захисту інформації.

Можливості для подальшої розробки програмного забезпечення включають розширення функціональності додатків, наприклад, можливість віртуального трейдингу, освітні матеріали для новачків у сфері інвестування, а також інтерактивні інструменти для аналізу портфеля та оптимізації стратегій інвестування.

Загалом, мобільні системи стали необхідним інструментом для інвесторів у сучасному світі, і подальший розвиток програмного забезпечення в цій області може значно полегшити процес прийняття інвестиційних рішень та підвищити ефективність управління фінансовими активами.

Основні функції мобільних додатків включають:

– Забезпечення доступу до актуальних цін на фінансові інструменти.

– Надання інвесторам доступу до різноманітних аналітичних інструментів, таких як графіки, індикатори технічного аналізу, новини та аналітичні звіти.

– Дозвіл користувачам відстежувати та керувати своїми інвестиціями.

– Надсилання сповіщень про зміни на ринку та досягнення встановлених цінових рівнів.

Мобільні системи володіють як своїми перевагами, так і недоліками, що важливо враховувати при їх використанні у сфері інвестицій.

До переваг такого виду мобільних систем можна віднести:

1. Мобільні додатки дозволяють інвесторам відстежувати та керувати інвестиціями в будь-який час та в будь-якому місці, що забезпечує більшу гнучкість та можливість оперативно реагувати на ринкові зміни.

2. Інтерфейси мобільних додатків розроблені з урахуванням потреб користувачів, що робить їх простими у використанні та дозволяє здійснювати різноманітні операції з інвестиціями без зайвих зусиль.

3. Інвестори можуть миттєво реагувати на ринкові зміни та отримувати актуальну інформацію, що дозволяє підтримувати високий рівень обізнаності та ефективно керувати своїм портфелем. На рисунку 1.1 наведено схему функціональності мобільних систем для інвестування.

Рисунок 1.1 – Схема функціональності мобільних систем для інвестування

У свою чергу такі системи мають і свої недоліки. Деякі мобільні додатки можуть мати обмежений функціонал у порівнянні з настільними версіями програм, що обмежує можливості користувачів у проведенні складних аналізів та операцій [5].

Також, використання мобільних додатків може створювати ризики безпеки, пов'язані з передачею фінансових даних через Інтернет, тому важливо обережно використовувати такі додатки та дотримуватися заходів безпеки.

Залежність від інтернет-з'єднання також може бути проблемним питанням. Для коректної роботи мобільних додатків потрібне стабільне Інтернет-з'єднання, що може створювати проблеми в умовах обмеженого доступу до мережі або низької швидкості Інтернету.

У сфері розвитку мобільних систем для оцінювання прибутковості інвестицій існують різні виклики та перспективи.

Безпека даних стає особливо важливою, оскільки зберігання та обробка персональних та фінансових даних користувачів потребує високого рівня захисту від кіберзлочинців та несанкціонованого доступу.

Інтеграція з іншими системами фінансових установ вимагає розробки додатків, які забезпечують безперервну інтеграцію з банками, брокерськими компаніями та іншими ключовими гравцями на фінансовому ринку.

Адаптивність до змін на ринку передбачає розвиток алгоритмів, які можуть швидко адаптуватися до змін у ринкових умовах. Це включає в себе постійне оновлення та вдосконалення програмного забезпечення для забезпечення відповідності актуальним тенденціям та потребам користувачів.

Мобільні системи відіграють важливу роль у сучасній інвестиційній діяльності, надаючи інвесторам інструменти для швидкого доступу до ринкової інформації, аналізу ризиків та прийняття обґрунтованих рішень.

Вони дозволяють інвесторам бути в курсі останніх ринкових тенденцій та негайно реагувати на зміни у фінансових ринках. Більш того, мобільні додатки стають все більш інтуїтивно зрозумілими та доступними, що сприяє підвищенню фінансової грамотності серед інвесторів. Завдяки мобільним системам інвестори можуть ефективніше управляти своїми портфелями та досягати своїх фінансових цілей.

1.2 Порівняльний аналіз аналогів системи

Аналіз аналогів дозволяє виявити сильні та слабкі сторони кожного аналогу. Це допомагає інвесторам чітко зрозуміти, що пропонує кожен сервіс і як вони відповідають їх потребам та вимогам. Завдяки порівняльному аналізу інвестор може зробити інформований вибір серед доступних альтернатив. Це допомагає уникнути неочікуваних проблем і втрат у майбутньому.

Також, порівняльний аналіз дозволяє вибрати сервіс, який надає найкращу комбінацію функціональності, вартості та якості обслуговування. Це

допомагає інвесторам максимізувати свій потенційний прибуток та оптимізувати витрати.

Аналізування спонукає компанії до постійного вдосконалення своїх продуктів та послуг, щоб привернути більше клієнтів і залишити їх задоволеними. Це сприяє здоровій конкуренції на ринку та стимулює інновації.

Отже, першим аналогом, що буде розглянуто у підрозділі буде «Robinhood»

Robinhood – це мобільний додаток, що дозволяє безкоштовно торгувати акціями, опціонами та криптовалютами. Він відомий своєю простотою та доступністю для початківців. Проте, має обмежений функціонал аналізу ринку та технічного аналізу [6]. У минулому виникали проблеми з безпекою даних. На рисунку 1.2 наведено приклад інтерфейсу цього аналогу.

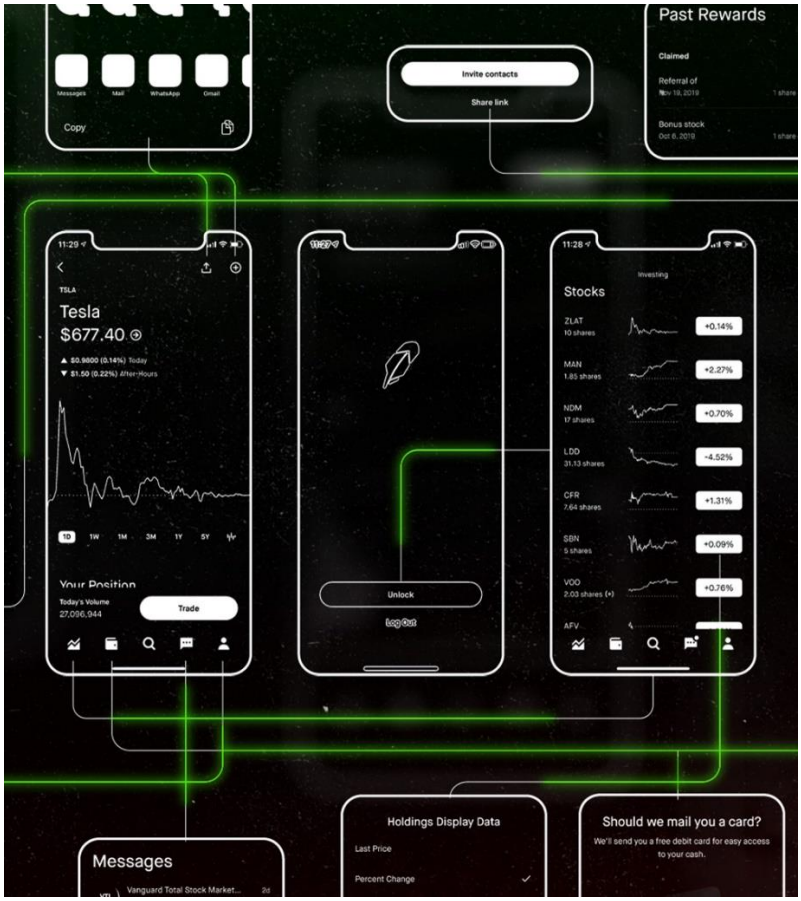


Рисунок 1.2 – Приклад інтерфейсу застосунку «Robinhood»

TD Ameritrade пропонує мобільний додаток для торгівлі акціями, опціонами та фондами [7]. Він володіє розширеним функціоналом для аналізу ринку та інвестиційних можливостей. Однак деякі користувачі відзначають нестабільну роботу додатка на деяких пристроях та відсутність деяких функцій у порівнянні з веб-версією. На рисунку 1.3 наведено приклад інтерфейсу цього аналогу.

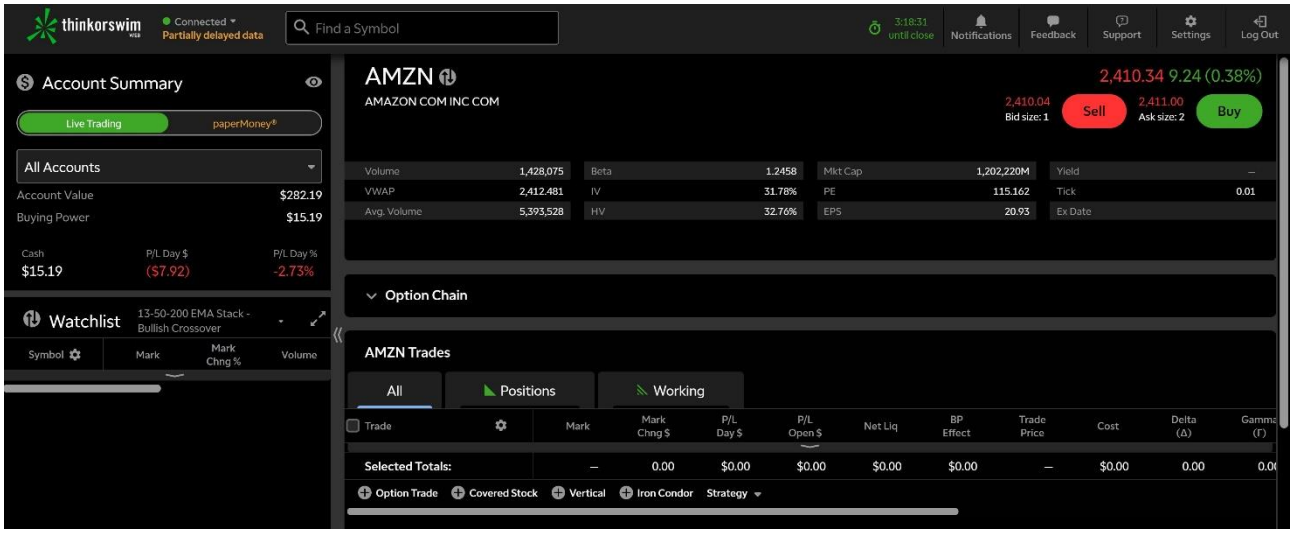


Рисунок 1.3 – Приклад інтерфейсу «TD Ameritrade»

Acorns – це інноваційний мобільний додаток, який спрощує інвестування для користувачів, пропонуючи автоматичне інвестування їх заощаджень у різні інвестиційні фонди [8]. Основний принцип Acorns полягає в тому, що він автоматично "округлює" ваші покупки до найближчого долара і інвестує різницю в інвестиційні фонди. Цей метод дозволяє навіть тим, хто має обмежений бюджет, розпочати інвестування без значного внеску.

Однак, не дивлячись на переваги, які надає Acorns, деякі недоліки варто врахувати. Перш за все, високі витрати на управління портфелем можуть стати проблемою для тих, хто має невеликі суми грошей на інвестування. Особливо для користувачів з невеликими обсягами активів ці витрати можуть значно зменшити загальний дохід від інвестування.

Крім того, обмежений вибір інвестиційних продуктів може обмежити інвестиційні можливості користувачів. Така обмеженість може збентежити деяких інвесторів, які шукають ширший спектр інвестиційних можливостей для диверсифікації свого портфеля та зниження ризику. На рисунку 1.3 наведено приклад інтерфейсу цього аналогу.

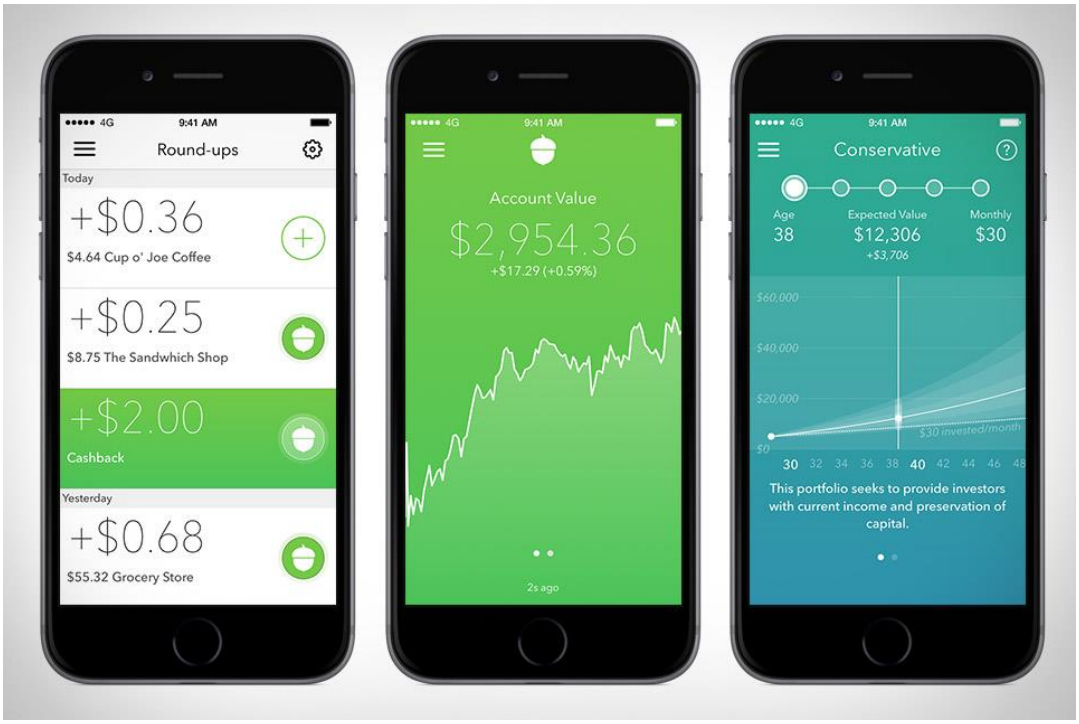


Рисунок 1.4 – Інтерфейс додатку Acorns

Розроблюване програмне забезпечення призначене для покращення функціоналу та обслуговування мобільних додатків, які використовуються для оцінювання прибутковості інвестицій. Це програмне забезпечення має розширений набір інструментів для аналізу ринку, що допоможе інвесторам отримувати більш детальну та зрозумілу інформацію про поточні тенденції та можливості на ринку.

Одним з основних покращень є збільшення рівня безпеки даних. Крім того, було впроваджено покращення, які спрямовані на поліпшення користувацького досвіду та зручності використання додатків. Це удосконалений інтерфейс користувача, додаткові функції аналізу та

інструменти для прийняття обґрунтованих інвестиційних рішень, а також розширені можливості інтеграції з іншими фінансовими платформами та сервісами.

Загалом, розроблюване програмне забезпечення має на меті підвищити якість та ефективність мобільних додатків для інвесторів, забезпечуючи їх розширеними інструментами аналізу, підвищеною безпекою даних та поліпшеним користувацьким досвідом.

Таблиця 1.1 – Порівняльна характеристика аналогів

Функціонал	Robinhood	TD Ameritrade	Acorns	Розроблюване ПЗ
Безкоштовна торгівля	+	–	–	+
Аналітика ринку	–	+	–	+
Автоматизовані інвестиції	–	+	+	+
Вартість управління портфелем	+	+	+	–
Безпека даних	–	+	–	+

Отже, у цьому підрозділі розглянуто важливість порівняльного аналізу мобільних систем для оцінювання прибутковості інвестицій, а також описано розроблюване програмне забезпечення, спрямоване на поліпшення функціоналу та обслуговування таких додатків.

Порівняльний аналіз дозволяє інвесторам зробити обґрунтований вибір серед наявних аналогів, визначивши переваги та недоліки кожного з них.

Розроблюване програмне забезпечення має на меті покращення функціоналу та сервісу мобільних додатків для оцінювання прибутковості

інвестицій. Воно планується мати розширений набір інструментів для аналізу ринку, покращену безпеку даних та інші покращення, що сприятимуть поліпшенню якості та ефективності інвестиційного процесу для користувачів.

1.3 Постановка задач бакалаврської дипломної роботи

Постановка задач бакалаврської дипломної роботи є критичним етапом у процесі підготовки роботи. Враховуючи широкий спектр методів та технологій, що можуть бути використані для розв'язання задачі розробки програмного забезпечення для оцінювання прибутковості інвестицій, постановка задачі визначає напрямок дослідження та конкретизує завдання, які потрібно вирішити.

Основною метою постановки задачі є уточнення мети та області застосування роботи [9]. Це допомагає зрозуміти, які саме проблеми потрібно вирішити та які конкретні завдання дослідження повинні бути вирішені. Наприклад, важливо уточнити, які саме фінансові інструменти будуть аналізуватися та які показники прибутковості будуть враховуватися.

Постановка задачі також допомагає визначити методи, які будуть використовуватися для аналізу фінансових даних та розробки програмного забезпечення. Вона дозволяє уточнити, чи будуть використовуватися фінансові моделі, статистичні методи, методи машинного навчання чи інші технології.

Крім того, постановка задачі визначає критерії успіху дослідження. Це допомагає зрозуміти, як буде оцінюватися результат роботи та які конкретні показники потрібно буде досягти.

Ураховуючи вищезгадані аспекти, постановка задачі в бакалаврській дипломній роботі стає ключовим етапом, що визначає успішність та релевантність дослідження. Її проведення допомагає уникнути невизначеності та розпливчатості цілей дослідження, а також забезпечує чітку орієнтацію в процесі його виконання.

Маємо наступні задачі з розробки програмного забезпечення для оцінювання прибутковості інвестицій:

1. Розробка модуля для збору та обробки фінансових даних.
2. Створення алгоритму для аналізу ризиків та прибутковості інвестицій.
3. Впровадження блокчейн технологій для забезпечення безпеки та достовірності даних.

Ці задачі було обрано, оскільки вони безпосередньо відображають основні вимоги та функціональні можливості програмного забезпечення, необхідного для розв'язання поставленої проблеми.

Розробка модуля для збору та обробки фінансових даних є ключовою, оскільки правильне збирання та обробка даних є основою для подальшого аналізу та прийняття рішень.

Створення алгоритму для аналізу ризиків та прибутковості інвестицій визначає ефективність та точність роботи програми. Цей алгоритм повинен бути здатний враховувати різні фактори, які впливають на прибутковість інвестицій, та надавати користувачам об'єктивну інформацію.

Впровадження блокчейн технологій допомагає забезпечити безпеку та надійність даних, що є важливим аспектом в фінансовій сфері. Ця технологія дозволяє зберігати та обмінюватися фінансовою інформацією безпечно та ефективно.

Постановка задач з розробки програмного забезпечення для оцінювання прибутковості інвестицій важлива для чіткого визначення цілей та напрямку роботи. Обрані задачі відображають основні функціональні вимоги до програми і гарантують її ефективність та надійність.

1.4 Аналіз методів розв’язання задачі

Аналіз методів розв'язання задачі розробки програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти є важливим етапом в процесі створення такого програмного продукту [10]. Розглянутий

аналіз спрямований на визначення найбільш ефективних та надійних методів, які дозволять користувачам отримати об'єктивну та достовірну інформацію про прибутковість їхніх інвестицій.

Один із можливих підходів - використання фінансових моделей. Це включає в себе математичні алгоритми та моделі, які допомагають прогнозувати поведінку фінансових інструментів на ринку. Наприклад, модель Чорного-Шоулза використовується для оцінювання цін опціонів, а модель CAPM (Capital Asset Pricing Model) допомагає визначити ризик та прибутковість інвестиційного портфеля [11].

Додатково, статистичні методи грають важливу роль у аналізі фінансових даних. Вони дозволяють виявляти зв'язки між різними факторами та прибутковістю інвестицій, а також аналізувати часові ряди для прогнозування майбутніх тенденцій на ринку.

Зокрема, методи машинного навчання стали дуже популярними в останні роки. Вони дозволяють автоматично аналізувати величезні обсяги даних та виявляти складні зв'язки, що допомагає приймати кращі інвестиційні рішення.

Оптимізаційні методи також використовуються для вибору оптимального портфеля інвестицій, щоб максимізувати прибуток при обмежених ресурсах та ризиках. Ці методи можуть враховувати різноманітні фактори, такі як ризик, дохідність та ліквідність активів.

Не менш важливим є економічний аналіз, що враховує економічні фактори та ринкові умови при оцінці прибутковості інвестицій. Це допомагає зрозуміти вплив зовнішніх чинників на фінансові ринки та приймати обґрунтовані інвестиційні рішення.

Крім вищезгаданих методів, також можна розглянути використання симуляційних моделей [12]. Цей підхід дозволяє моделювати різні сценарії розвитку ринку та оцінювати їх вплив на прибутковість інвестицій. Симуляційні моделі дають можливість аналізувати ризики та приймати рішення на основі прогнозування можливих результатів у різних умовах. Цей метод

дозволяє інвесторам приймати обґрунтовані рішення на основі реалістичних прогнозів та уникати непередбачених ризиків.

Крім того, важливо враховувати роль технологій блокчейну в оцінюванні прибутковості інвестицій. Технологія блокчейну може забезпечити безпеку та надійність обміну фінансовою інформацією, що важливо для інвесторів та фінансових установ. Вона може також допомогти в автоматизації процесів, зменшенні витрат та покращенні доступності фінансових послуг.

У кінцевому підсумку, аналіз методів розв'язання задачі програмного забезпечення для оцінювання прибутковості інвестицій є складним та многогранним процесом, який вимагає інтеграції різноманітних методів та підходів для досягнення оптимальних результатів.

1.5 Висновок

Отже, у розділі було поставлено передусім важливу задачу – визначення напрямку та обсягу бакалаврської дипломної роботи. Ця постановка задачі є ключовим етапом у підготовці дослідження, оскільки вона визначає загальний контекст і мету роботи.

В ході аналізу були розглянуті аналоги програмного забезпечення для оцінювання прибутковості інвестицій. Цей аналіз дозволив виявити існуючі рішення та їхні переваги та недоліки. Ретельне вивчення аналогів дозволило визначити прогалини та можливості для подальшого вдосконалення власного програмного продукту.

Проведено аналіз стану мобільних систем у сфері оцінювання прибутковості інвестицій у фінансові інструменти. Цей аналіз дозволив виявити сучасні тенденції та визначити можливості для впровадження мобільних технологій у дану сферу.

2 МАТЕМАТИЧНІ МОДЕЛІ ДЛЯ ОЦІНЮВАННЯ ПРИБУТКОВОСТІ ОВДП

2.1 Основні фактори, що впливають на прибутковість ОВДП

При оцінюванні прибутковості ОВДП (облігацій внутрішньої державної позики) слід враховувати низку ключових факторів, кожен з яких має значний вплив на загальний дохід інвестора. Нижче наведено детальний опис основних факторів, що впливають на прибутковість ОВДП:

Номінальна вартість облігації. Номінальна вартість облігації, або її «номінал», є основною сумою, яку емітент обіцяє виплатити інвестору при погашенні облігації. Для ОВДП номінальна вартість зазвичай є стандартною і встановлюється урядом. Номінальна вартість важлива, оскільки вона використовується для розрахунку купонних виплат та визначення загальної суми, яку інвестор отримає при погашенні облігації[13]. На рисунку 2.1 наведено схему номінальної вартості облігацій.

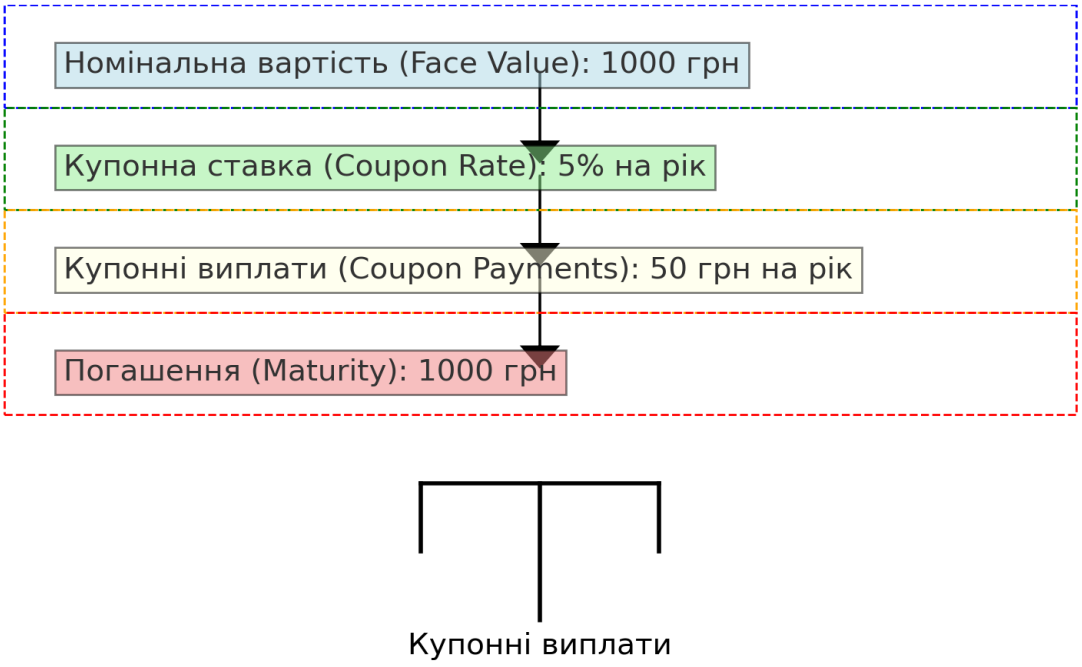


Рисунок 2.1 – Схема номінальної вартості облігацій

Купонна ставка. Купонна ставка – це відсоткова ставка, за якою здійснюються періодичні виплати інвестору. Виплати здійснюються у формі купонних платежів, які можуть бути щорічними, щоквартальними або в іншому періодичному режимі, залежно від умов облігації. Купонна ставка визначається під час емісії облігації і залишається фіксованою протягом усього терміну її обігу. Вона є основним джерелом доходу для інвесторів і відіграє важливу роль у визначенні привабливості облігації на ринку [14]. На рисунку 2.2 наведено схему купонних виплат облігацій.

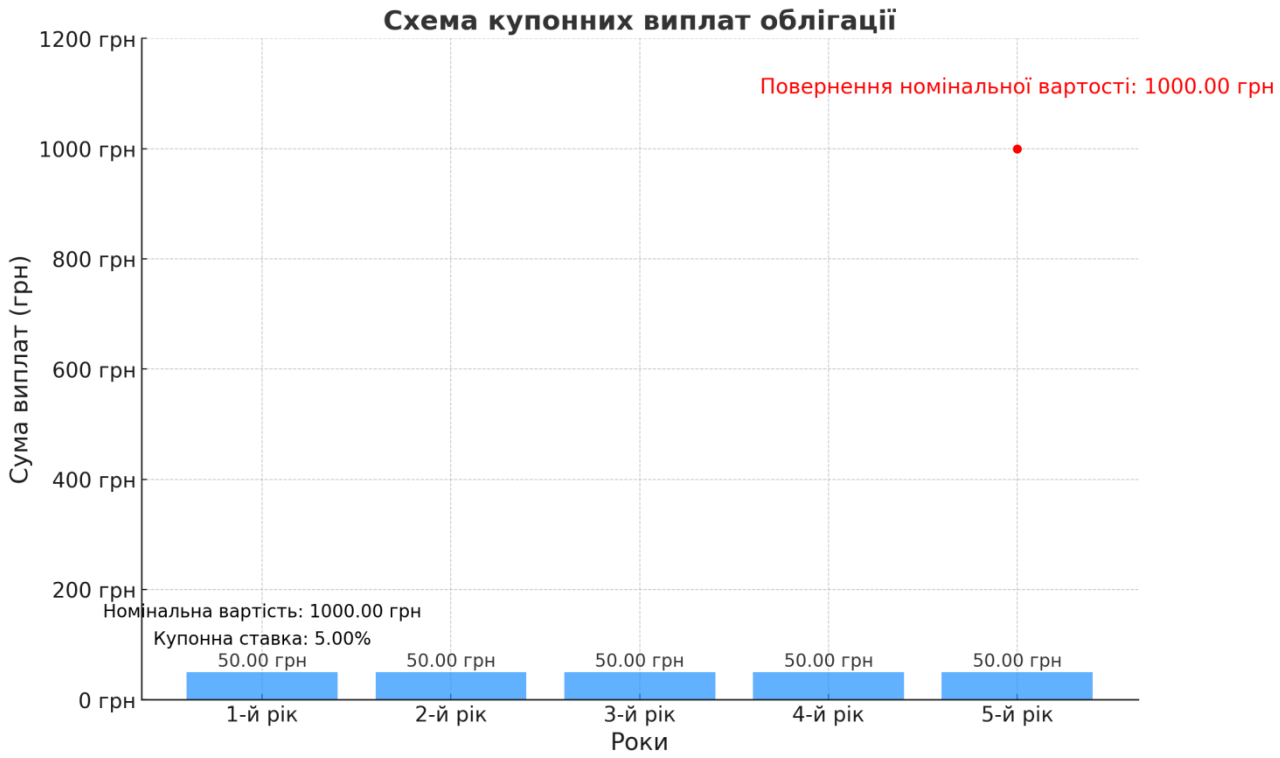


Рисунок 2.2 – Схема купонних виплат облігацій

Термін до погашення. Термін до погашення облігації визначає період, протягом якого облігація залишається в обігу до моменту її погашення емітентом. Термін може варіюватися від декількох місяців до кількох десятиліть. Термін до погашення є критичним фактором для оцінювання ризику і прибутковості облігації. Зазвичай, довгострокові облігації мають вищу купонну ставку для компенсації інвесторам за додатковий ризик, пов'язаний із тривалішим періодом утримання [15].

Ринкова ціна облігації. Ринкова ціна облігації – це поточна ціна, за якою облігація продається або купується на вторинному ринку. Вона може відрізнятися від номінальної вартості залежно від ринкових умов, таких як зміни процентних ставок, інфляція, кредитний рейтинг емітента та інші макроекономічні фактори. Ринкова ціна облігації є важливим параметром, оскільки вона впливає на поточну дохідність інвестора і використовується при розрахунку показників прибутковості, таких як прибутковість до погашення (YTM) та поточна дохідність (Current Yield) [16].

Інфляція. Інфляція має значний вплив на реальну прибутковість облігацій. Висока інфляція може знизити реальну вартість майбутніх купонних виплат та номінальної вартості, яку інвестор отримає при погашенні облігації. Інвестори повинні враховувати очікуваний рівень інфляції при оцінюванні реальної прибутковості облігацій [17].

Процентні ставки. Зміни у процентних ставках центрального банку впливають на ринкові ціни облігацій. Коли процентні ставки зростають, ринкова ціна облігацій зазвичай знижується, оскільки нові облігації випускаються з вищими купонними ставками. Навпаки, коли процентні ставки знижуються, ринкова ціна облігацій зростає. Інвестори повинні враховувати прогнозовані зміни процентних ставок при оцінюванні прибутковості своїх інвестицій [18]. На рисунку 2.3 наведено схему впливу процентних ставок на ринкову ціну облігацій.

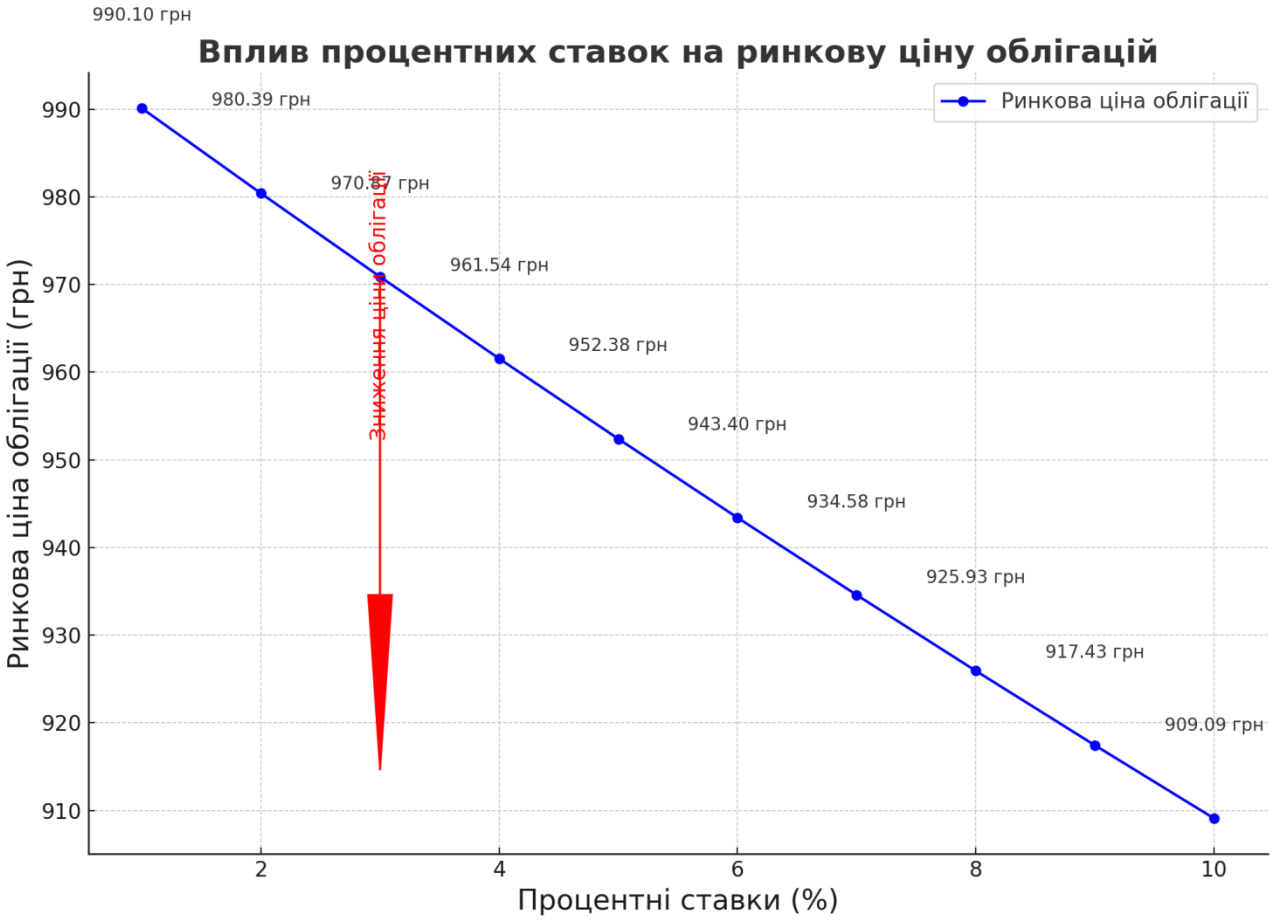


Рисунок 2.3 – Схема впливу процентних ставок на ринкову ціну облігацій

Кредитний рейтинг емітента. Кредитний рейтинг емітента облігацій відображає його фінансову стабільність та здатність виконати свої боргові зобов'язання. Вищий кредитний рейтинг знижує ризик дефолту і, як правило, призводить до нижчої купонної ставки. Інвестори повинні враховувати кредитний рейтинг емітента при оцінюванні ризиків і потенційної прибутковості інвестицій в ОВДП [19].

2.2 Модель оцінювання прибутковості до погашення (Yield to Maturity, YTM)

Yield to Maturity (YTM) – це показник, який дозволяє оцінити загальну прибутковість облігаці, якщо буде утримуватися до погашення. Він враховує всі майбутні купонні виплати, а також різницю між поточною ринковою ціною

облігації та її номінальною вартістю. Це один з найважливіших показників для інвесторів, оскільки він дає змогу порівняти прибутковість різних облігацій з урахуванням усіх фінансових потоків.

Розрахунок YTM проводиться за наступною формулою:

$$YTM = \frac{C + \frac{F \cdot P}{n}}{\frac{F + P}{2}}$$

де:

- C – купонний платіж;
- F – номінальна вартість облігації;
- P – поточна ринкова ціна облігації;
- n – кількість років до погашення.

Ця формула враховує всі майбутні доходи від облігації, а також можливий прибуток або збиток від її продажу на момент погашення. Оскільки YTM враховує всі аспекти дохідності, його можна вважати внутрішньою нормою доходності облігації. Інвестори часто використовують цей показник для прийняття рішень про купівлю або продаж облігацій.

Для прогнозування прибутковості облігацій можна також використати метод лінійної регресії, який дозволяє враховувати декілька факторів одночасно:

$$\gamma = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

де:

γ – передбачене значення прибутковості;

β_0 – вільний член;

$\beta_1, \beta_2, \cdots, \beta_n$ – коефіцієнти;

$x_1, x_2, \cdots, x_n$ – змінні, що описують різні фактори, які можуть впливати на прибутковість, наприклад, ринкові ставки, інфляції та кредитний рейтинг.

2.3 Математичне моделювання оцінки ризиків і дохідності ОВДП

Розробка математичної моделі для оцінки прибутковості та ризиків ОВДП включає врахування різних змінних та використання відповідних математичних формул. Основні змінні, що враховуються, включають номінальну вартість, купонні ставки, поточні ринкові ціни та макроекономічні показники, такі як процентні ставки та рівень інфляції.

Розрахунок поточної вартості облігації

Поточна ринкова ціна облігації визначається як сума приведених вартостей майбутніх купонних платежів і номінальної вартості:

$$P = \sum_{t=1}^{n} \frac{C}{(1+r)^t} + \frac{F}{(1+r)^n}$$

де:

- P – поточна ринкова ціна облігації;
- C – купонний платіж;
- r – ринкова процентна ставка;
- F – номінальна вартість облігації;
- n – кількість років до погашення.

Оцінка прибутковості з урахуванням інфляції

Прибутковість з урахуванням інфляції розраховується за формулою:

$$Y_{\text{реальна}} = \frac{1 + Y_{\text{номінальна}}}{1 + \text{Інфляція}} - 1$$

де:

- $Y_{\text{реальна}}$ – реальна прибутковість;
- $Y_{\text{номінальна}}$ – номінальна прибутковість;
- Інфляція – рівень інфляції.

Ця формула дозволяє оцінити реальний дохід інвестора з урахуванням знецінення грошей через інфляцію.

Розрахунок дюрації облігації

Дюрація облігації – це показник чутливості ціни облігації до змін процентної ставки. Вона вимірює середньозважений час до отримання всіх платежів по облігації. Розрахунок дюрації проводиться за формулою:

$$D = \frac{\sum_{t=1}^{n} \frac{t \cdot C}{(t+r)^t} + \frac{n \cdot F}{(1+r)^n}}{P}$$

де:

- D – дюрація облігації;
- C – купонний платіж;
- r – ринкова процентна ставка;
- F – номінальна вартість облігації;
- n – кількість років до погашення;
- P – поточна ринкова ціна облігації.

Для моделювання ймовірності погашення облігації можна застосувати логістичну регресію:

$$\gamma = \frac{1}{1 + e - (\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n)}$$

де:

γ – ймовірність погашення облігації;

β_0 – вільний член;

$\beta_1, \beta_2, \cdots, \beta_n$ – коефіцієнти;

$x_1, x_2, \cdots, x_n$ – змінні, які можуть включати різні економічні та ринкові фактори.

2.4 Алгоритмічне забезпечення для розрахунку дохідності ОВДП

Розробка алгоритмів для розрахунку прибутковості ОВДП включає використання математичних моделей і формул, описаних у попередніх розділах. Алгоритми повинні забезпечити точність і ефективність обчислень.

Алгоритм розрахунку купонних виплат

Купонні виплати розраховуються за формулою:

$$C = F \times \frac{S}{100}$$

де:

- C – купонний платіж;
- F – номінальна вартість облігації;
- S – купонна ставка.

Цей алгоритм дозволяє визначити регулярні виплати інвестору протягом терміну облігації.

Алгоритм визначення ринкової вартості облігації

Для визначення поточної ринкової вартості облігації використовується приведене значення майбутніх платежів:

$$P = \sum_{t=1}^{n} \frac{C}{(1+r)^t} + \frac{F}{(1+r)^n}$$

Цей алгоритм враховує всі майбутні грошові потоки, що генеруються облігацією, і приводить їх до поточної вартості.

Алгоритм оцінювання прибутковості до погашення (YTM)

Розрахунок YTM включає наступні кроки:

1. Ініціалізація значень купонних платежів (C), номінальної вартості (F), поточної ціни (P) і кількості років до погашення (n).
2. Використання формули YTM для ітеративного обчислення досягнення точного значення.

Алгоритм врахування ризиків

Для врахування ризиків, пов’язаних з кредитним рейтингом емітента, застосовується корекція розрахованих значень прибутковості:

$$Y_{\text{скоригована}} = YTM \times (1 - R_{\text{ризик}})$$

де:

- $Y_{\text{скоригована}}$ –скоригована прибутковість;

- $R_{\text{ризик}}$ – коефіцієнт ризику, визначений кредитним рейтингом.

Аналіз і прогнозування

На основі отриманих даних система може виконати аналіз і прогнозування майбутньої прибутковості облігацій. Це дозволяє інвесторам приймати обґрунтовані рішення щодо купівлі чи продажу облігацій. Алгоритми прогнозування можуть включати методи регресійного аналізу, моделі часових рядів та інші статистичні методи для оцінки майбутніх змін ринку.

Регресія - спосіб вибрати з сімейства функцій ту, що мінімізує функцію втрат. Остання характеризує, наскільки сильно пробна функція відхиляється від значень у заданих точках. В даному випадку точками є значення ціни облігації на деяку дату. Оскільки точки отримані експериментально, вони неминуче містять помилку, шум, тому розумніше вимагати, щоб функція передавала загальну тенденцію, а не точно проходила всі точки. Регресію можна вважати «інтерполюючою апроксимацієб»: необхідно провести криву якомога ближче до точок і зберегти її максимально простою, щоб вловити загальну тенденцію. За баланс між цими суперечливими вимогами відповідає функція втрат (в англомовних джерелах «loss function» або «cost function»).

У випадку лінійної регресії сімейство функцій, з яких відбувається вибір, є лінійною комбінацією наперед заданих базисних функцій f_i

$$f = \sum_i w_i f_i$$

Лінійну регресію називають лінійною саме через лінійну комбінацію базисних функцій f_i — це не пов'язано з самими базовими функціями (вони можуть бути лінійними чи ні). Мета регресії — знайти коефіцієнти цієї лінійної комбінації, і цим визначити регресійну функцію f (яку також називають моделлю).

Часовий ряд є набором спостережень, отриманих шляхом регулярного вимірювання однієї змінної протягом деякого періоду часу. Так, у ряді даних облігацій представлено щоденні значення за кілька місяців. У часовому ряді деяка змінна спостерігається регулярно із заданим інтервалом протягом

певного проміжку часу. Таким чином, форма даних для типового часового ряду – це одна послідовність або список спостережень, виконаних через рівні проміжки часу.

2.5 Висновки

Розроблені моделі та алгоритми дозволять створити ефективну систему для оцінювання прибутковості ОВДП, яка враховуватиме основні фактори, що впливають на дохідність інвестора. Це забезпечить можливість об'єктивного аналізу та прийняття обґрунтованих інвестиційних рішень.

3 РОЗРОБКА ПРОГРАМНИХ МОДУЛІВ РЕАЛІЗАЦІЇ ДЛЯ ОЦІНЮВАННЯ ПРИБУТКОВОСТІ У ФІНАНСОВІ ІНСТРУМЕНТИ

3.1 Варіантний аналіз і обґрунтування вибору засобів для реалізації програмного забезпечення

В сучасному фінансовому світі, де конкуренція постійно зростає, важливо мати ефективні інструменти для оцінки прибутковості інвестицій у фінансові інструменти. Розробка програмного забезпечення для таких цілей стає ключовим завданням для фінансових аналітиків, інвестиційних менеджерів та інших учасників ринку. Це допомагає їм приймати обгрунтовані рішення на основі об'єктивних даних і аналізу.

Враховуючи складність фінансових ринків та постійні зміни умов інвестування, важливо мати програмне забезпечення, яке може забезпечити не лише точність та швидкість обробки даних, але й аналітичні можливості для прогнозування та моделювання різних сценаріїв.

Одним із ключових аспектів вибору програмних засобів для розробки такого ПЗ є їхні функціональні можливості, зокрема, здатність ефективно обробляти фінансові дані, розраховувати показники прибутковості та ризику, а також проводити складні фінансові аналізи.

Додатково, важливо враховувати питання безпеки даних, оскільки фінансова інформація є дуже чутливою та піддається ризику кібератак. Тому обрані програмні засоби повинні мати високий рівень захисту даних та відповідати вимогам регулювальних органів у сфері фінансів.

У цьому розділі буде проведено варіантний аналіз різних програмних засобів для реалізації програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти. Аналіз буде базуватися на їхніх функціональних можливостях, безпеці даних, а також враховуватиме вартість впровадження та підтримки програмного забезпечення.

Розглянемо основні характеристики кожної мови програмування.

C – це мова програмування загального призначення, відома своєю ефективністю і швидкістю виконання. Вона часто використовується для системного програмування та розробки вбудованих систем через свою низькорівневу природу та прямий доступ до апаратного забезпечення [20].

R – це мова програмування та середовище для статистичних обчислень і графічної візуалізації даних. Вона широко використовується в аналізі даних і наукових дослідженнях завдяки великому набору пакетів для статистичного аналізу [21].

C++ – це мова програмування загального призначення, яка поєднує в собі можливості низькорівневого програмування, характерного для C, та об'єктно-орієнтованого підходу, що дозволяє створювати складні програмні системи [22].

І, нарешті, Java – це мова програмування, яка відома своєю крос-платформенністю, безпекою та простотою використання. Вона широко використовується в розробці мобільних застосунків, веб-сервісів та корпоративних систем [23].

Порівнюючи ці мови програмування за зазначеними критеріями, можна зробити висновок. По-перше, щодо функціональних можливостей, Python і R можуть мати перевагу у відношенні до C і C++, оскільки вони мають потужні бібліотеки для обробки фінансових даних та аналізу.

Проте, щодо безпеки даних, Java виявляється більш підходящою завдяки своїй вбудованій системі безпеки та контролю доступу до даних. Щодо вартості впровадження та підтримки, мови програмування, такі як Python та JavaScript, можуть бути більш економічно вигідними, оскільки вони мають безкоштовні інструменти та велику спільноту розробників.

У таблиці 3.1 наведено порівняльну характеристику кожної з названих мов програмування для визначення найбільш точної для виконання визначених задач.

Таблиця 3.1 – Порівняльна таблиця мов програмування

Критерій	C	R	C++	Java	Python	JavaScript
Функціональність	+	–	+	+	+	+
Безпека даних	+/–	–	+/–	+	–	–
Вартість впровадження та підтримки	+	+	+	+/–	+	+
Крос-платформ. сумісність	–	–	–	+	–	–
Швидкодія	+	-	+	+	–	+
Простота використання	–	+	–	+	+	+
Розширюваність	+	+	+	+	+	+
Спільнота розробників	+	+	+	+	+	+

Отже, враховуючи, що тема передбачає розробку мобільного застосунку, Java, з її крос-платформенністю та широким використанням у розробці Android-додатків, виявляється найбільш оптимальним вибором. Вона поєднує в собі надійність, швидкість та безпеку, що важливо для мобільних застосунків, і має широкий набір інструментів для розробки. Таким чином, Java може бути найкращим варіантом для реалізації вашого програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти в мобільному форматі.

3.2 Обгрунтування середовища розробки

Вибір інтегрованого середовища розробки (Integrated Development Environment – IDE) для розробки програмного забезпечення на Java є важливим етапом у процесі розробки. IDE повинно забезпечити зручність, продуктивність та ефективність для розробника. У нашому випадку, оскільки ми розглядаємо

розробку мобільного застосунку для оцінювання прибутковості інвестицій у фінансові інструменти на Java, оптимальним вибором буде IDE, яка надає розширену підтримку розробки Android-додатків.

IntelliJ IDEA – це інтегроване середовище розробки (IDE) для програмістів, розроблене компанією JetBrains [24]. Вона підтримує багато мов програмування, включаючи Java, Kotlin, Groovy, Scala та інші. На рисунку 3.1 навдено приклад інтерфейсу цієї IDE.

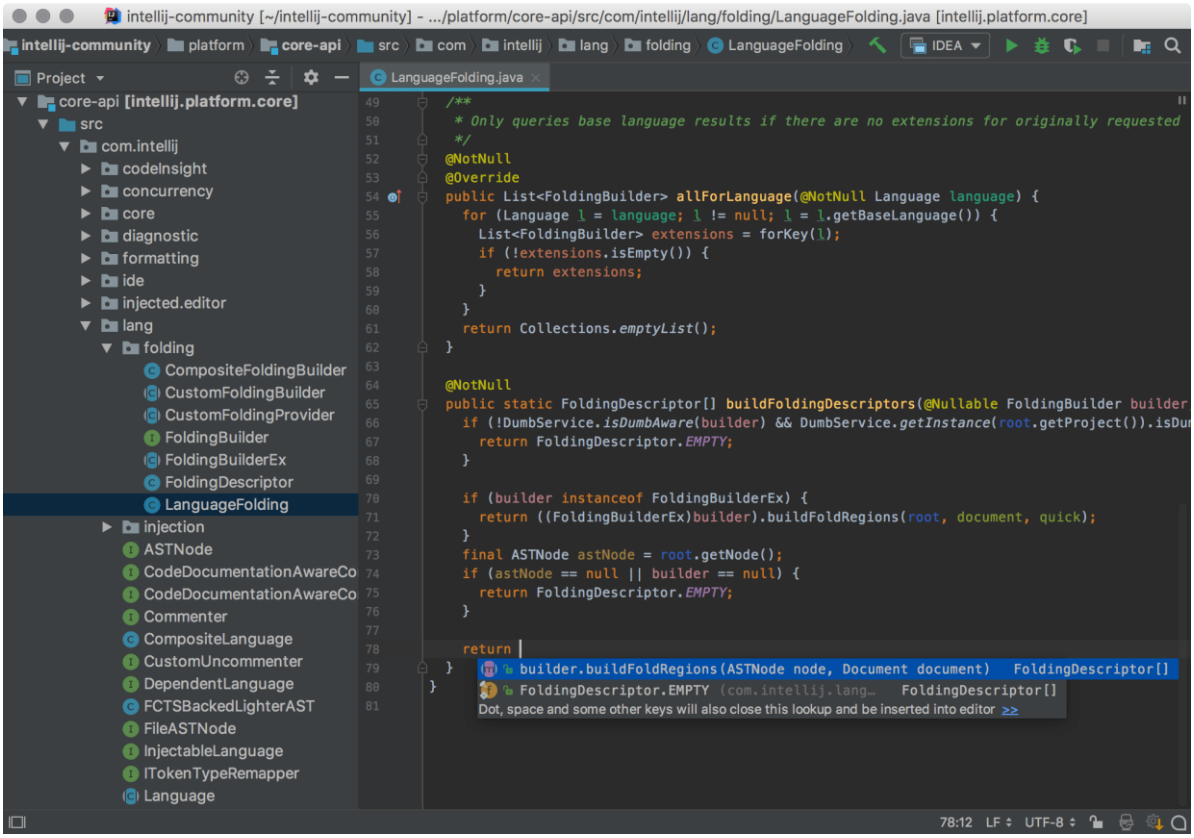


Рисунок 3.1 – Приклад інтерфейсу IntelliJ IDEA

Висока продуктивність, розширені можливості рефакторингу коду, широкий набір інструментів для розробки та велика кількість розширень є його перевагами. Проте, платна версія має вартість, і велика кількість функцій може бути переважною для початківців.

Eclipse – це інтегроване середовище розробки (IDE), що створене для роботи з різними мовами програмування, включаючи Java, C/C++, PHP, Python та інші [25]. На рисунку 3.2 наведено приклад інтерфейсу цієї IDE.

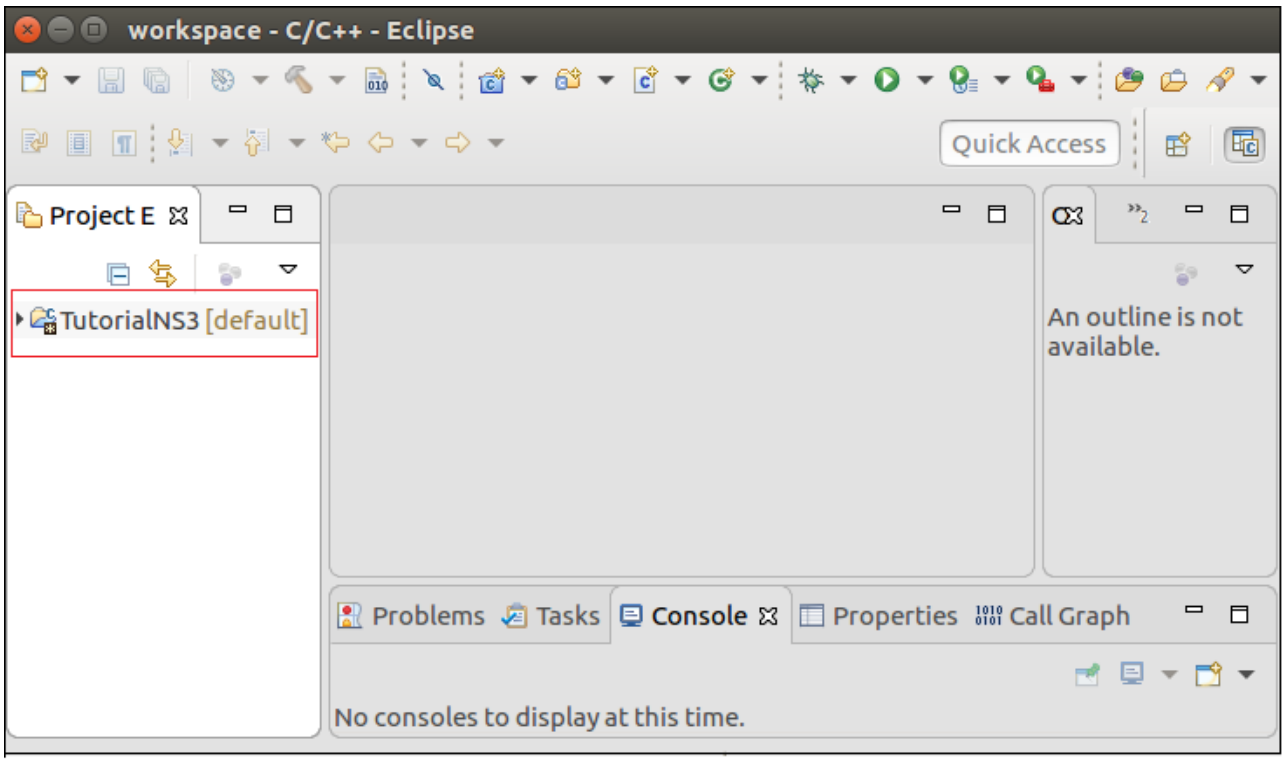


Рисунок 3.2 – Інтерфейс IDE Eclipse

Його перевагами є відкрите джерело, широкий вибір плагінів та активна спільнота користувачів. Проте, воно менш продуктивне порівняно з іншими IDE, і має меншу кількість розширених функцій.

Android Studio – це офіційне інтегроване середовище розробки (IDE) для платформи Android, розроблене компанією Google. Воно оптимізоване для розробки Android-додатків і має інтегровані емулятори, візуальний редактор інтерфейсів, підтримку Kotlin, широкий набір інструментів для тестування та підтримку Google Services. Проте, його обмеження полягає в тому, що воно спеціалізоване лише на розробці для Android. На рисунку 3.3 наведено приклад інтерфейсу.

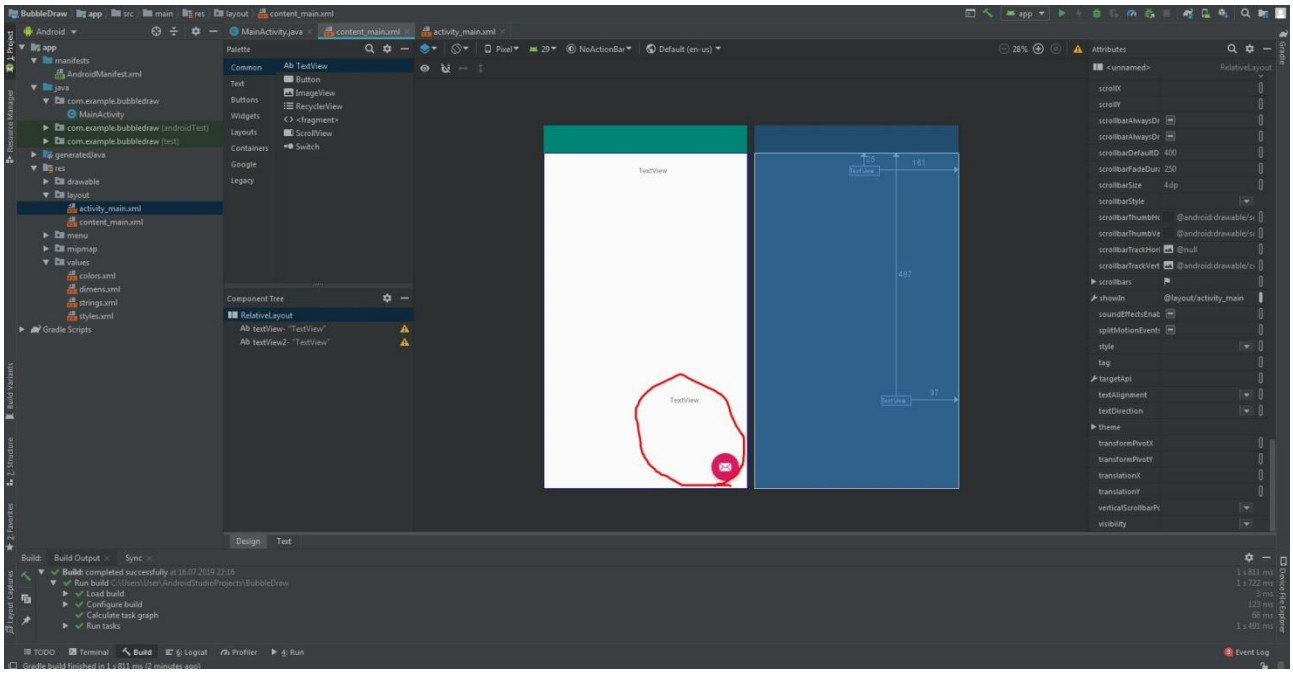


Рисунок 3.3 – Інтерфейс Android Studio

Щодо IntelliJ IDEA, вона відома своєю високою продуктивністю та розширеними можливостями рефакторингу коду. Вона є потужним інструментом для роботи з різними мовами програмування, але для новачків може здатися складною через велику кількість функцій у платній версії. Обговоренням є те, що для професіоналів це чудовий вибір, оскільки вони можуть використовувати всі розширені можливості для підвищення продуктивності та якості коду.

З щодо Eclipse, воно пропонує великий вибір плагінів та активну спільноту користувачів, що робить його важливим інструментом для роботи з різними мовами програмування. Проте, його менша продуктивність та обмежена кількість розширених функцій може затримувати досвідчених розробників.

Нарешті, Android Studio є спеціалізованим інструментом для розробки Android-додатків. Його інтегрованість з Android SDK та широкий набір інструментів для тестування та підтримки Google Services роблять його найкращим вибором для розробки для платформи Android. Однак, він

обмежений лише розробкою для Android, що може бути недоліком у випадку, коли розробка розширюється на інші платформи.

Таким чином, найкращим варіантом для розробки мобільного застосунку на Java є Android Studio. Це інтегроване середовище розробки, спеціалізоване на розробці Android-додатків, що базуються на Java. Android Studio надає широкий набір інструментів та функціональних можливостей, включаючи емулятори, візуальний редактор інтерфейсів, інтеграцію зі засобами Google та багато іншого. Крім того, Android Studio має активну спільноту розробників та регулярно оновлюється з урахуванням останніх тенденцій у розробці мобільних додатків.

3.3 Розробка модуля авторизації користувача

У світі фінансів та інвестицій розробка програмного забезпечення стає все більш важливою та актуальною задачею. Сучасні технології дозволяють створювати ефективні інструменти для оцінки ризику та прибутковості інвестицій у різноманітні фінансові продукти.

Один із ключових аспектів в цьому процесі - розробка модуля авторизації користувача, який забезпечує безпеку та конфіденційність фінансових даних. Цей модуль виступає основною ланкою у забезпеченні доступу до програмного забезпечення, гарантуючи, що лише авторизовані користувачі матимуть доступ до фінансової інформації та інвестиційних інструментів.

У даному розділі буде представлений процес розробки та реалізації модуля авторизації, зокрема, архітектура, методи аутентифікації та захист від несанкціонованого доступу.

Вирішення цих завдань є критичним для забезпечення ефективності та надійності фінансового програмного забезпечення, що сприяє успішній роботі інвестиційних компаній та фінансових установ.

Інтерфейс входу повинен бути зручним та інтуїтивно зрозумілим для користувачів, типові елементи такого інтерфейсу включають поля для введення

логіну та паролю, кнопку для підтвердження входу, а також можливість відновлення забутого пароля. Важливо також забезпечити можливість двофакторної аунтефікації для підвищення рівня безпеки. При розробці цього інтерфесу слід звертати увагу на адаптивність дизайну для забезпечення зручного доступу з різних пристроїв, таких як комп'ютери, планшети та смартфони.

Модуль авторизації користувача є ключовою частиною будь-якої системи, оскільки забезпечує безпеку даних і визначає права доступу користувачів. У нашому програмному забезпеченні для оцінювання прибутковості інвестицій у фінансові інструменти, модуль авторизації відповідає за перевірку облікових даних користувача перед наданням доступу до основного функціоналу.

Модуль авторизації реалізований у вигляді активності (Activity) в застосунку. Він включає в себе наступні елементи:

1. UI для введення облікових даних. Користувач має можливість ввести свій логін та пароль у відповідних текстових полях.

2. Перевірка введених даних. Перед тим, як надати доступ, validateInput(String username, String password) перевіряє коректність введених даних. У випадку, якщо поля логіну та паролю порожні, виводиться відповідне повідомлення.

3. Аутентифікація. Після успішної перевірки введених даних authenticate(String username, String password) перевіряє автентичність користувача. У нашому випадку використовується дамі аутентифікація, де логін "user" та пароль "password" є валідними.

4. Перехід на головний екран. Після успішної аутентифікації користувача startMainActivity() перенаправляє його на головний екран додатку, де він може використовувати основний функціонал.

5. Закриття поточної активності. Для забезпечення безпеки та уникнення повернення користувача до екрану авторизації після успішного входу, closeActivity() закриває поточну активність авторизації (Рисунок 3.4).

Рисунок 3.4 – Модуль авторизації користувача

Модуль авторизації забезпечує безпеку та конфіденційність даних користувачів у нашому додатку для оцінювання прибутковості інвестицій. Це важлива частина функціоналу, яка дозволяє користувачам впевнено використовувати програму, знаючи, що їхні дані захищені.

3.4 Розробка модуля додавання інвестиції

Модуль додавання інвестиції є ключовою складовою частиною нашого програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти. Він відповідає за можливість користувача додавати нові інвестиції до свого портфеля. Ця функціональність істотно розширює можливості додатку, роблячи його більш гнучким та привабливим для користувачів. Додаток надає зручний та ефективний інструмент для керування інвестиціями, дозволяючи користувачам з легкістю контролювати свій фінансовий портфель і приймати обмірковані рішення щодо їхніх інвестиційних стратегій.

В цьому модулі користувачам надається можливість ввести дані нової інвестиції, такі як назва інвестиції та її сума. Після введення даних користувач може зберегти інвестицію в свій портфель, що дозволяє відстежувати її в додатку та враховувати при аналізі прибутковості інвестиційного портфеля.

Ця можливість додає значну вартість для користувачів, дозволяючи їм зберігати всі свої інвестиції в одному місці та зручно керувати ними. Такий підхід робить програму більш привабливою для інвесторів та сприяє підвищенню їхньої задоволеності від користування застосунком.

Для забезпечення інтеграції з іншими частинами системи, дані про нові інвестиції повинні бути коректно збережені в базі даних і бути доступними для подальшого аналізу або обробки. Це дозволяє користувачам з легкістю переглядати додані інвестиції, аналізувати їхню ефективність та проводити необхідні фінансові операції.

Модуль додавання інвестиції реалізований у вигляді окремої активності (Activity). Він включає в себе наступні елементи:

UI для введення даних про інвестицію: Користувач має можливість ввести назву інвестиції та її суму відповідно до поля editTextInvestmentName та editTextInvestmentAmount.

Логіка додавання інвестиції: Після натискання кнопки "Add Investment" (buttonAddInvestment) здійснюється спроба додати інвестицію. Ця логіка здійснюється в методі addInvestment(String name, String amount). У випадку успішного додавання виводиться повідомлення "Investment Added", в іншому випадку – "Error Adding Investment".

Повернення до списку інвестицій: Після успішного додавання інвестиції можливо повернутися до списку інвестицій. Це можна реалізувати шляхом навігації назад до відповідної активності, або викликавши метод навігації через інтент (Рисунок 3.5).

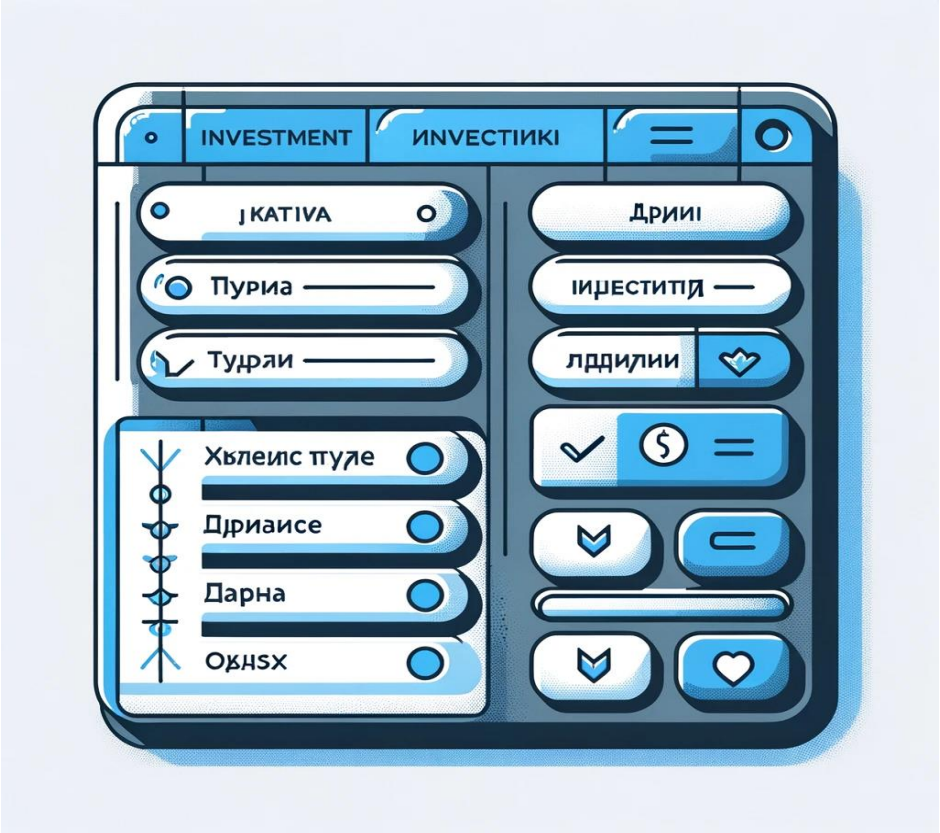


Рисунок 3.5 – Модуль додавання інвестиції

Модуль додавання інвестиції дозволяє користувачам зручно та ефективно керувати своїми інвестиціями в програмі. Він є важливою складовою частиною системи, що дозволяє розширити функціональні можливості додатку та полегшити процес управління інвестиціями.

3.5 Висновки

У розділі було обгрунтовано вибір мови програмування Java для виконання поставлених завдань БДР. Android Studio є спеціалізованим інструментом для розробки Android-додатків. Його інтегрованість з Android SDK та широкий набір інструментів для тестування та підтримки Google Services роблять його найкращим вибором для розробки для платформи Android. Також у розділі було описано роботу двох модулів розробленої системи, а саме модуль авторизації користувача, а також створення інвестиції.

4 ВПРОВАДЖЕННЯ, ТЕСТУВАННЯ ТА РОЗРОБКА ДОКУМЕНТАЦІЇ

4.1 Впровадження програмного продукту

Впровадження програмного забезпечення є важливим етапом в процесі його життєвого циклу, який включає інтеграцію, налаштування, тестування, а також навчання користувачів і оцінку результатів. Цей процес є критичним для забезпечення безперебійної роботи системи та її відповідність очікуванням користувачів.

Впровадження починається з ретельної підготовки, яка включає детальний аналіз існуючого середовища, оцінку сумісності нового ПЗ з поточними системами та розробку відповідної технічної документації. На цьому етапі важливо врахувати всі особливості апаратного та програмного забезпечення, яке буде використано для розгортання системи, щоб уникнути технічних проблем у майбутньому.

Інтеграція нового ПЗ з існуючою інфраструктурою є наступним кроком. Вона включає розгортання серверів, налаштування бази даних, що буде використовуватися для зберігання та обробки даних, а також імпорт наявних даних до нової системи. Імпорт даних може потребувати їх попереднього очищення та конвертації для забезпечення сумісності з новим ПЗ. На цьому етапі важливо забезпечити правильну конфігурацію системи, щоб вона могла ефективно взаємодіяти з іншими компонентами інфраструктури.

Після інтеграції починається тестування системи. Спочатку проводиться технічне тестування, яке включає перевірку функціональності, продуктивності та безпеки ПЗ. Це тестування допомагає виявити потенційні проблеми та недоліки в роботі системи. Далі, пілотне тестування проводиться у тестовому середовищі з обмеденою кількістю користувачів для виявлення можливих проблем в реальних умовах експлуатації. На завершення, функціональне

тестування дозволяє перевірити, чи відповідає система вимогам користувачів, і оцінити її готовність до повноцінного впровадження (Рисунок 4.1).

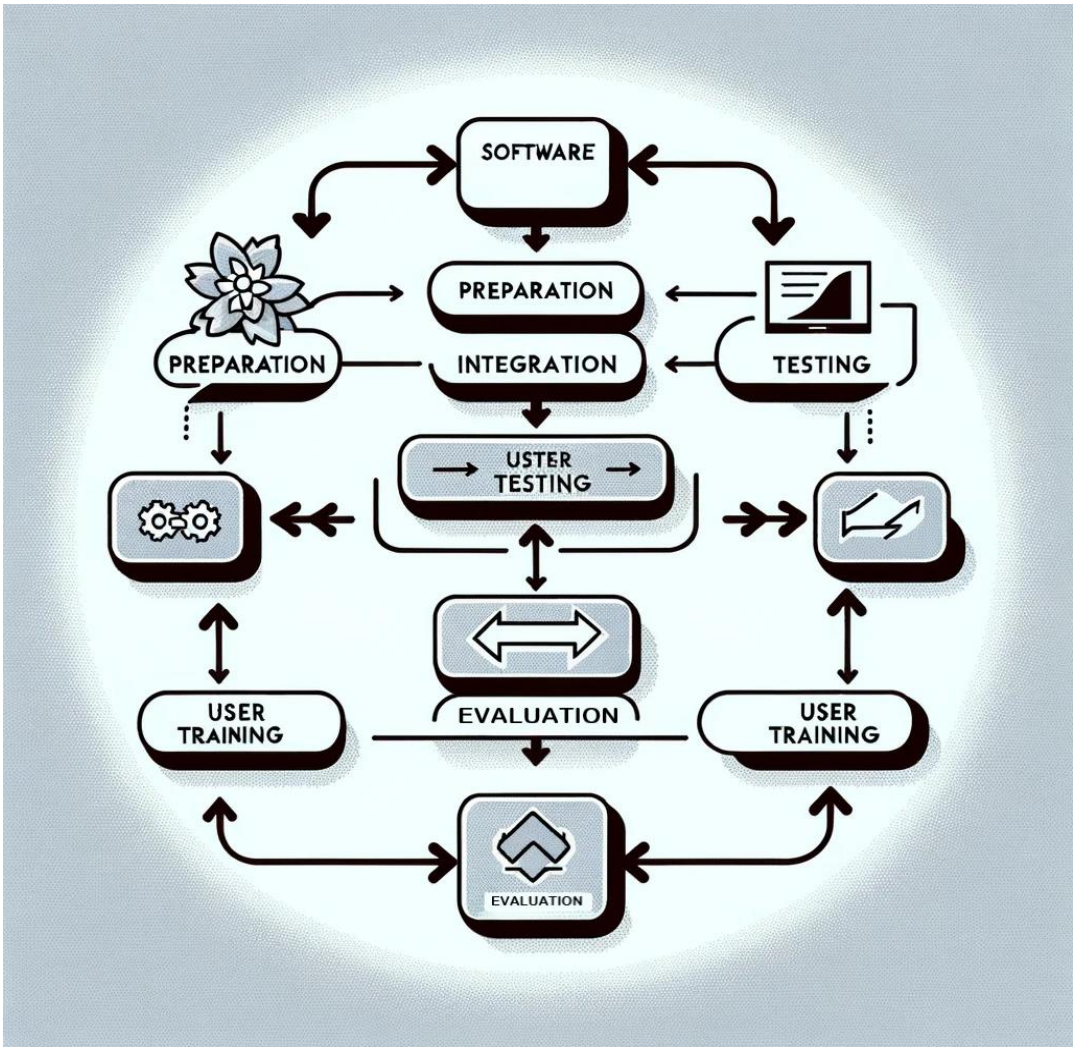


Рисунок 4.1 – Схема процесу впровадження ПЗ

Навчання користувачів є ключовим компонентом процесу впровадження, оскільки від їхнього розуміння та вміння користуватися новим ПЗ залежить успішність його інтеграції у бізнес-процеси. Проводяться тренінги, де користувачі отримують інструкції щодо роботи з новим ПЗ, а також забезпечується технічна підтримка та консультація протягом періоду адаптації. Навчання повинно бути структурованим та орієнтованим на різні рівні користувачів, забезпечуючи як базові, так і розширені знання про систему.

Оцінка впровадження включає збір зворотного зв’язку від користувачів, аналіз їх задоволеності та виявлення можливих проблем у функціонування системи. Після цього проводиться оцінка ефективності ПЗ шляхом аналізу показників продуктивності і порівняння їх з початковими очікуваннями. Це дозволяє визначити, наскільки успішно нова система відповідає поставленим цілям та чи потрібно внести додаткові корективи для її вдосконалення. Якщо виявляються недоліки або потреби в додаткових функціях, вносяться необхідні коригування, що сприяє подальшому покращенню системи.

Під час впровадження можуть виникти різні виклики, зокрема технічні проблеми, пов’язані з інтеграцією ПЗ в існуючу IT-інфраструктуру, невідповідність очікувань користувачів функціональним можливостям системи, труднощі з міграцією даних або опір з боку користувачів до змін у бізнес-процесах. Важливо активно працювати над вирішенням цих проблем, забезпечуючи підтримку користувачів та постійний моніторинг роботи системи (Рисунок 4.2).

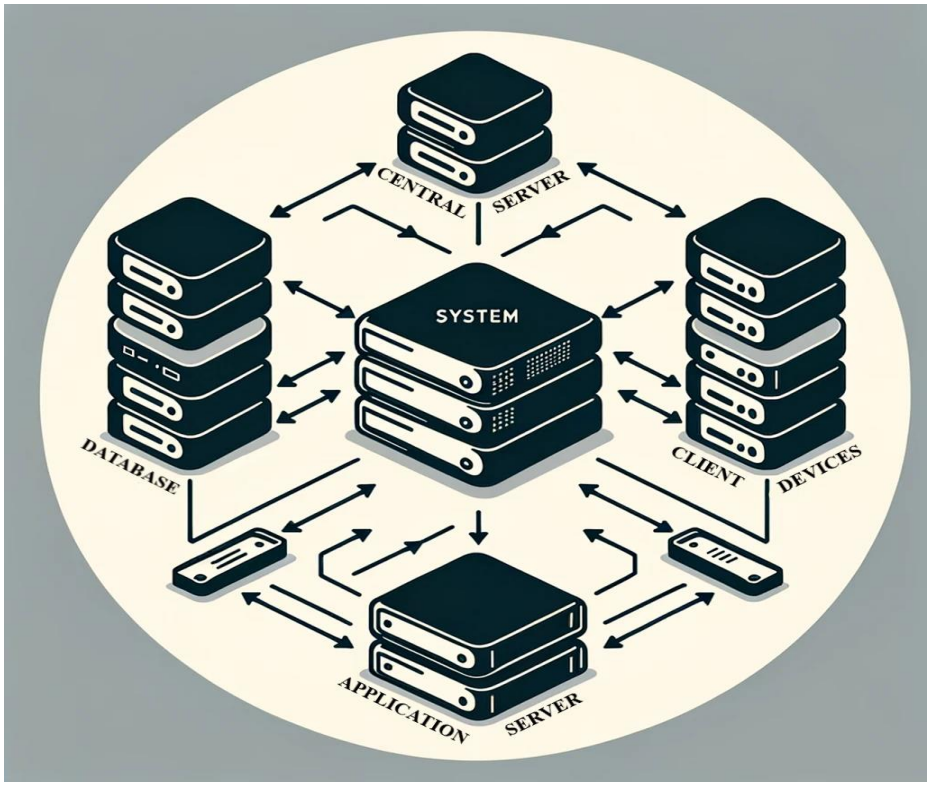


Рисунок 4.2 – Приклад інтеграції системи з існуючою IT-інфраструктурою

Таким чином, впровадження є комплексним процесом, який включає кілька важливих етапів. Це забезпечує успішну інтеграцію нового програмного забезпечення, його відповідність потребам користувачів, а також гарантує, що система працює ефективно і надійно в реальних умовах експлуатації.

4.2 Тестування програмного забезпечення

Тестування є ключовим етапом розробки програмного забезпечення. Включає в себе функціональне тестування, тестування продуктивності, а також тестування безпеки. Функціональне тестування перевіряє, чи відповідає програмне забезпечення вимогам і специфікаціям. Тестування продуктивності оцінює швидкість та ефективність роботи додатка під різними умовами. Тестування безпеки перевіряє захищеність системи від потенційних загроз і несанкціонованого доступу.

Функціональне тестування

Функціональне тестування було зосереджено на перевірці основних функцій системи, зокрема авторизації користувача, додавання нових інвестицій, перегляду інвестицій та налаштування системи сповіщень.

Для модуля авторизації користувача перевірено різні сценарії, включаючи успішний вхід до системи з валідними обліковими даними та обробку помилок при введенні невалідних даних. Система повинна була відмовляти у доступі при введені неправильного логіна або пароля, і це включало стуації, коли введені дані не відповідають жодному обліковому запису в базі даних. Перевірядися випадки з введенням некоректних паролів для існуючих логінів, а також сценарії, коли логін зовсів не зареєстрований у системі. Блок-схема алгоритму авторизації представлена на рисунку 4.3.

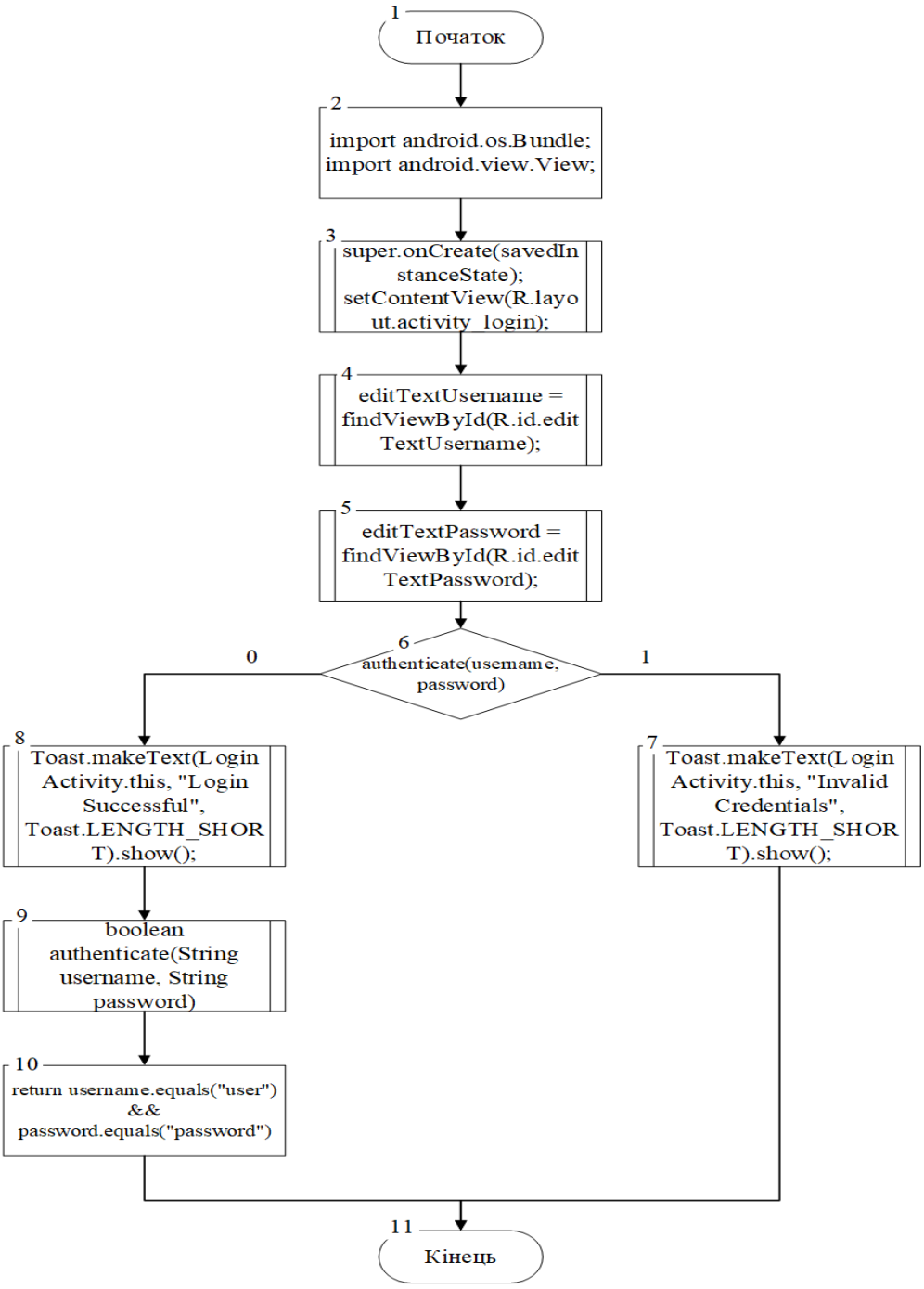


Рисунок 4.3 – Схема алгоритму авторизації

Для модуля додавання нових інвестицій тестування включало ретельну перевірку всіх аспектів цього процесу, щоб забезпечити коректність роботи і стабільність функціонування. Спершу було здійснено тестування введених даних про нові інвестиції, що включало як ручний, так і автоматизований підхід для перевірки різноманітних сценаріїв введення. Було перевірено, як система обробляє різні типи даних, включаючи текстові і числові значення, дати і

спеціальні символи. Особлива увага приділялася перевірці того, що система коректно відображає помилки введення даних, наприклад, коли користувач вводить невалідні або неповні дані. Це ключало тестування на відсутність обов'язкових полів, неправильні формати введення, та наявність аномалій у введених даних. Блок-схема додавання нових інвестицій представлена на рисунку 4.4.

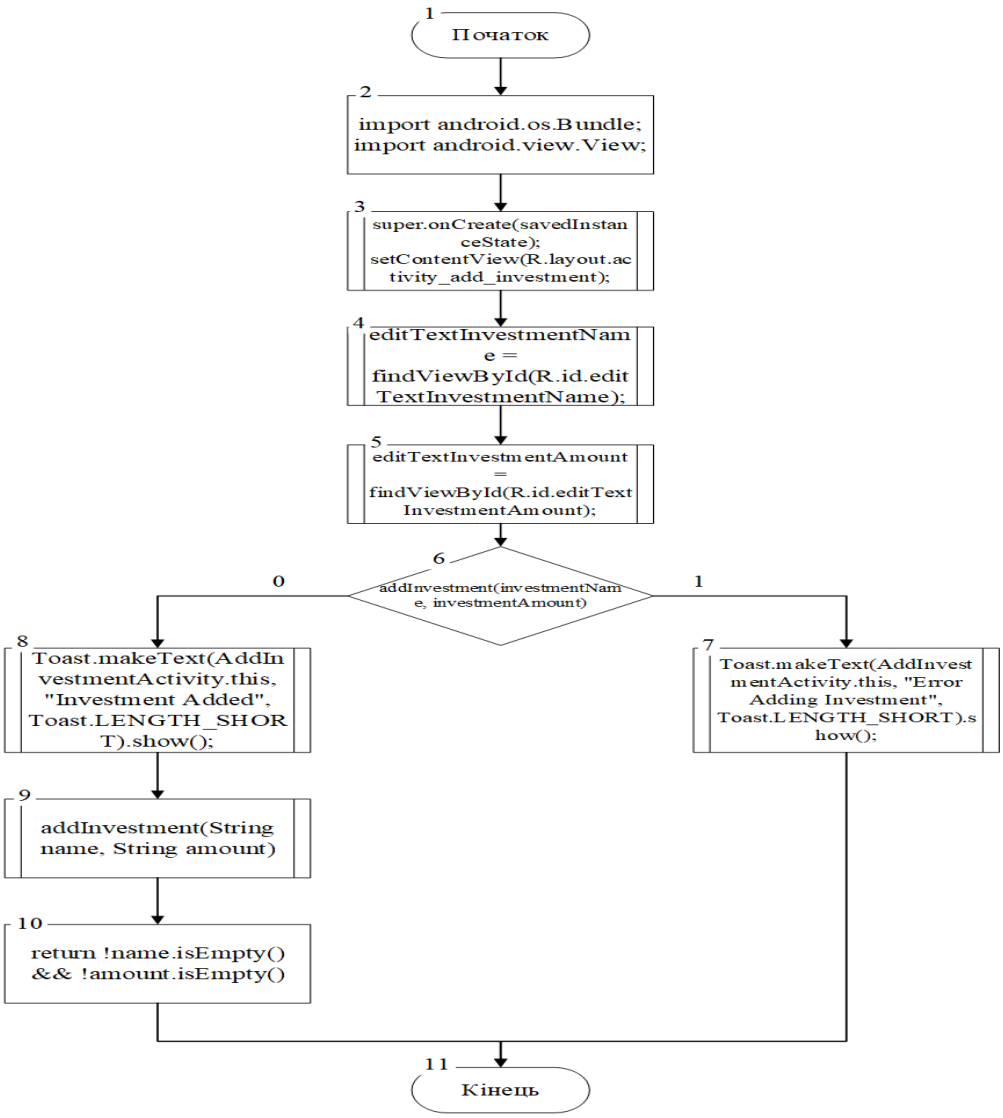


Рисунок 4.4 – Схема додавання інвестиції

Під час функціонального тестування було виявлено кілька критичних і незначних проблем. Однією з основних проблем було неправильне оброблення

невалідних даних під час авторизації, що вимагало додаткових перевірок та верифікації даних. Ще однією проблемо юбула оптимізація швидкості обробки даних у випадках з великими обсягами інвестиційних записів, що було вирішено шляхом покращення запитів до бази даних і оптимізації алгоритмів.

Нефункціональне тестування

Нефункціональне тестування включало оцінку продуктивності системи, її безпеки та зручності використанн. Продуктивність перевірялася шляхом моделювання великих обсягів даних і одночасних запитів користувачів для виявлення можливих вузьких місць у виконанні системи. Це тестування виявило необхідність оптимізації запитів до бази даних для підвищення швидкості обробки запитів.

Безпека системи перевірялася через симуляцію атак, таких як SQL-ін’єкції, злом паролів та інші можливі загрози. Це тестування допомогло забезпечити надійність захисту даних користувачів і запобігти несанкціонованому доступу до системи. Зокрема, було перевірено механізм шифрування та інших кофідеційних даних.

Зручність використання оцінювалася за допомогою юзабіліті-тестів, що включали збирання зворотного зв’язку від користувачів щодо інтерфейсу системи. Користувачі надавали відгуки про легкість навігації по системі, логічність розміщення елементів керування і зручність введення даних. Це дозволило виявити можливі покращення для підвищення зручності використання.

Інтеграційне тестування

Інтеграційне тестування було направлене на перевірку взаємодії між різними модулями системи. Це тестування гарантувало, що передача даних між компонентами відбувається без помилок, і всі модулі працюють разом назалежним чином.

Наприклад, інтеграційне тестування перевіряли, як модуль додавання нових інвестицій взаємодіє з базою даних і з модулем перегляду інвестицій.

Було забезпечено, що дані про нові інвестиції зберігаються правильно і коректно відображаються у списку інвестицій для користувача.

4.3 Приклади застовування математичних моделей і формул

У цьому підрозділі розглядаються конкретні приклади застосування математичних моделей і формул на реальних даних, отриманих з фінансових операцій. Це включає обробку операцій покупки і продажу, а також обчислення доходу на основі введених даниих. Використання реальних даних дозволяє наочно продемонструвати ефективність та правильність застосування моделей, що розроблені у програмному забезпеченні для оцінювання прибутковості інвестицій.

Для оцінювання прибутковості інвестицій використовувалися математичні моделі та формули, описані у другому розділі. Нижче наведено приклади їх застосування у програмному забезпеченні.

Оцінка прибутковості до погашення (Yield to Maturity, YTM)

Оцінка прибутковості до погашення є важливим показником для інвесторів, які вкладають кошти в облігації. Формула для розрахунку YTM виглядає наступним чином:

$$YTM = \frac{C + \frac{F - P}{n}}{\frac{F + P}{2}}$$

Приклад застосування:

Розглянемо облігацію з номінальною вартістю F = 1000 грн, поточною ринковою ціною P = 950 грн, річним купонним платежем C = 50 грн і терміном до погашення n = 5 років.

Підставляючи значення у формулу, отримуємо:

$$YTM = \frac{50 + \frac{1000 - 950}{5}}{\frac{1000 + 950}{2}} = \frac{50 + 10}{975} = \frac{60}{975} \approx 0.0615 \text{ або } 6.15\%$$

Цей показник використовується для аналізу привабливості облігацій порівняно з іншими інвестиційними інструментами.

Розразунок чистої приведеної вартості (Net Present Vakue, NPV)

Чиста приведена вартість є показником для оцінки доцільності інвестиційних проєктів. Вона розраховується за формулою:

$$NPV = \sum_{t=1}^{T} \frac{C_t}{(1+r)^t} - I_0$$

Приклад застосування:

Розглянемо інвестиційний проєкт з початковими інвестиціями $I_0 = 10000$ грн і оцікуваними грошовими протоками $C_1 = 2000$ грн, $C_2 = 3000$ грн, $C_3 = 4000$ грн, $C_4 = 5000$ грн, $C_5 = 6000$ грн. Ставка дисконту становить r = 0.10.

Підставляючи значення у формулу, отримуємо:

$$NPV = \frac{2000}{(1+0.10)^1} + \frac{3000}{(1+0.10)^2} + \frac{4000}{(1+0.10)^3} + \frac{5000}{(1+0.10)^4} + \frac{6000}{(1+0.10)^5} - 10000$$

Після розрахунків отримуємо:

$$NPV \approx 1818.18 + 2479.34 + 3005.41 + 3415.07 + 3725.47 - 10000 \approx$$

$$\approx 14443.47 - 10000 = 4443.47 \text{ грн}$$

Цей позитивний результат свідчить про те, що інвестиція є вигідною і проєкт варто реалізовувати.

Для ілюстрації застосування математичних моделей і формул на реальних даних було проведено тестування з використанням наступних даних про операції покупки і продажу (Таблиця 4.1).

Таблиця 4.1 – Фрагмент таблиці для тестування обробки даних

	Продаж	Покупка	Продаж	Покупка	Дохід	
Пн	964,18	963,89	966,08	963,89	17,65	
Вт	964,56	964,08	966,25	964,08	17,59	
Ср	964,64	964,49	966,63	964,49	17,58	
Чт	966,08	964,89	967,02	964,89	17,58	
Пт	966,25	965,30	967,40	965,30	17,57	
Сб			967,79	965,70	17,57	
Нд			968,17	966,11	17,56	
Пн	967,02	966,53	968,56	966,53	17,56	
Вт	967,4	966,94	968,94	966,94	17,55	
Ср	968,56	967,35	969,33	967,35	17,55	
Чт	968,94	967,76	969,71	967,76	17,54	
Пт	969,33	968,17	970,10	968,17	17,54	
Сб			970,48	968,58	17,54	
Нд			970,87	968,99	17,53	

4.4 Розробка документації

Розробка документації є критично важливим етапом у процесі створення програмного забезпечення, оскільки вона забезпечує як розробників, так і користувачів необхідними інструментами для розуміння, використання та підтримки продукту. Документація слугує ключовим засобом комунікації між всіма учасниками процесу розробки, а також надає інструкції для кінцевих користувачів щодо встановлення, налаштування і експлуатації програмного забезпечення.

Документація охоплює декілька основних аспектів, такі як:

1. Користувацька документація включає посібник користувача, що надає покрокові інструкції щодо використання основних функцій системи. У посібнику описані процеси авторизації, перегляду інвестицій, додавання нових інвестицій і налаштування сповіщень. Також документ містить розділ часто запитуваних питань (FAQ), де користувачі можуть знайти відповіді на типові

проблеми і питання. На рисунку 4.5 представлено приклад користувацької документації з інструкцією по встановленню програми.

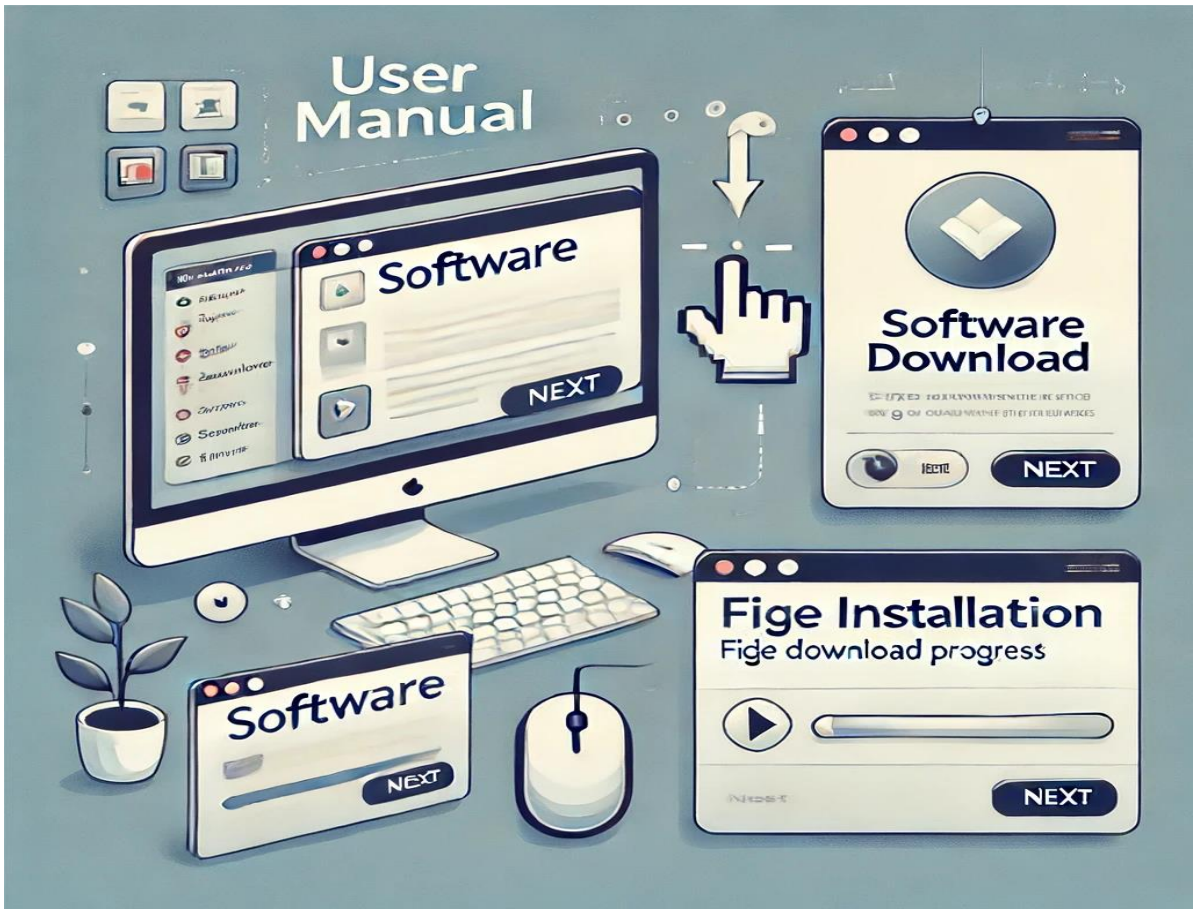


Рисунок 4.5 – Приклад користувацької документації з інструкцією по встановленню програми

2. Документація для адміністратора включає інструкції з налаштування і підтримки системи. Це включає процедури встановлення програмного забезпечення, налаштування бази даних, управління користувачами і забезпечення безпеки системи. Інструкції охоплюють кроки з управління обліковими записами користувачів, включаючи їх створення, видалення та модифікацію.

3. Технічна документація охоплює архітектуру системи і інструкції для розробників. Вона містить UML-діаграми, що візуалізують основні компоненти системи, їх взаємодію і функціональність. Ця документація

допомагає розробникам зрозуміти структуру коду, використані бібліотеки і методи розширення або інтеграції з іншими системами. UML-діаграма діяльності розробленої системи представлена на рисунку 4.5.

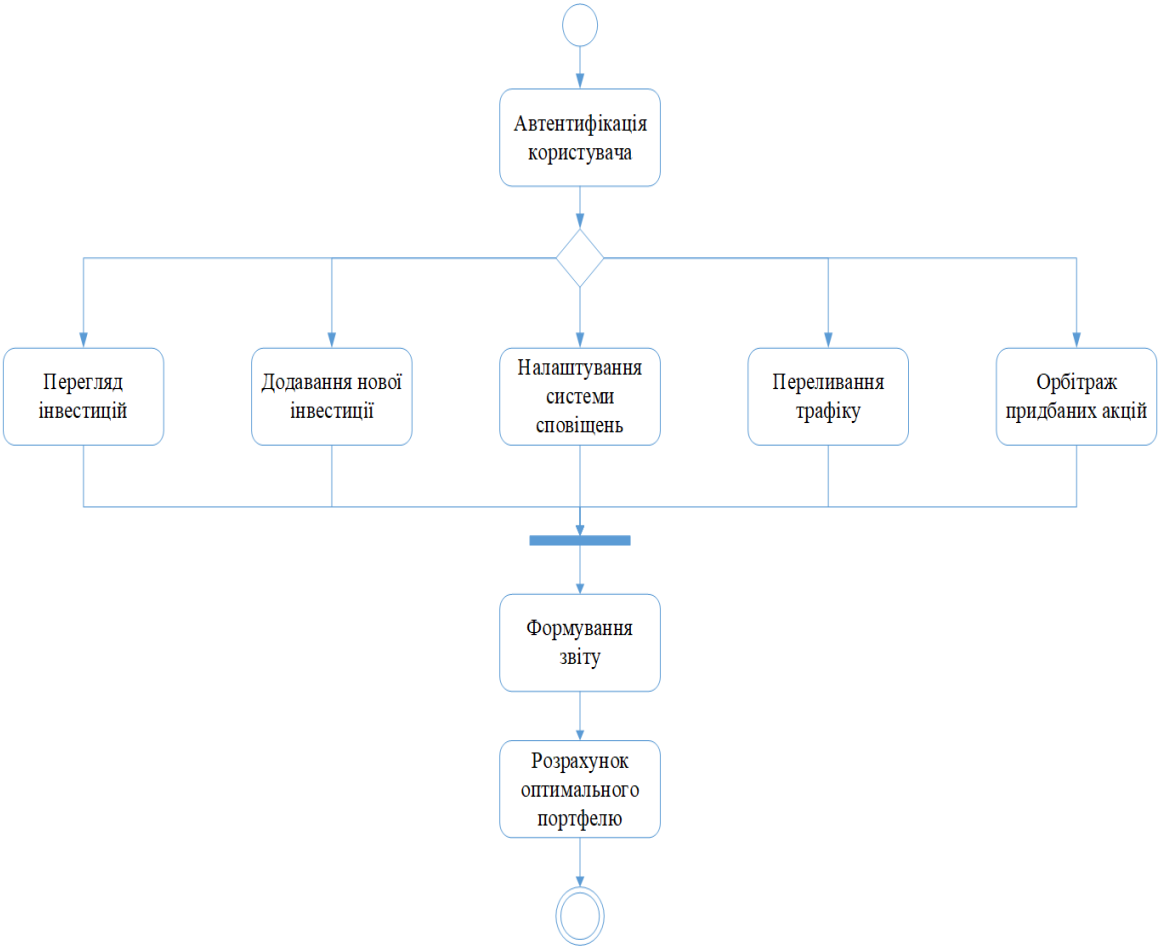


Рисунок 4.5 – UML-діаграма діяльності розробленої системи

UML-діаграма показує структуру і основні потоки операцій у системі, включаючи процеси авторизації користувача, перегляд інвестицій, додавання нових інвестицій та налаштування сповіщень. Це забезпечує розуміння архітектури системи і полегшує її підтримку та розвиток.

4. Графічна документація також є важливим елементом, оскільки вона надає візуальні засоби для розуміння архітектури системи і процесів її роботи. Наприклад, архітектурні діаграми описують загальну структуру програми, включаючи основні компоненти і їх взаємодію. Такі діаграми допомагають

швидко зрозуміти, як функціонує система і як її частини взаємодіють між собою. На рисунку 4.6 наведено приклад архітектурної діаграми.

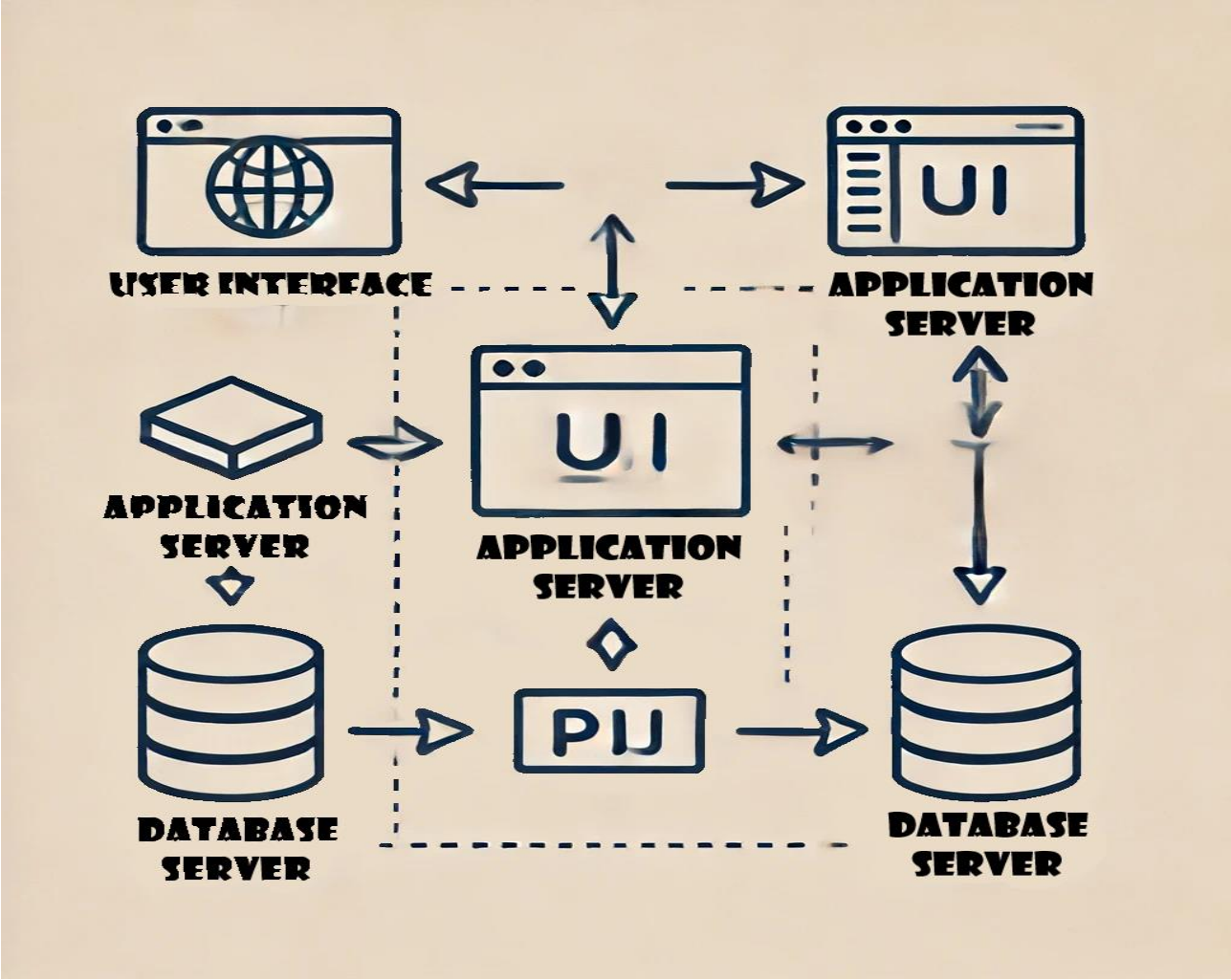


Рисунок 4.6 – Приклад архітектурної діаграми

На завершення, підтримка актуальності документації є важливим завданням. Регулярне оновлення документів відповідно до змін у програмному забезпеченні, додавання нових функцій і виправлення застарілої інформації є необхідним для забезпечення її точності і корисності. Використання сучасних систем контролю версій дозволяє легко відслідковувати всі зміни і повертатися до попередніх версій документації в разі потреби.

4.5 Висновки

Проведене тестування та розробка документації підтвердили високу якість програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти. Тестування дозволило виявити і виправити критичні проблеми, забезпечити функціональність, надійність і безпеку системи. Розроблена документація надає всі необхідні інструкції для користувачів та адміністраторів, забезпечуючи легкість впровадження і експлуатації системи.

Впроваджені функції дозволяють користувачам ефективно управляти своїми інвестиціями, надаючи їм інструменти для авторизації, перегляду інвестицій, додавання нових активів та налаштування сповіщень. Подальший розвиток системи може включати вдосконалення алгоритмів аналізу, інтеграцію з іншими фінансовими інструментами та розширення функціональності для покращення підтримки прийняття інвестиційних рішень.

ВИСНОВКИ

Бакалаврська диломна робота присвячена розробці програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти. У ході дослідження було досягнуто мети створення ефективного інструменту, що дозволяє користувачам без глибоких знань у сфері фінансів легко аналізувати та прогнозувати прибутковість інвестицій.

Основним завданням роботи було спрощення процесу оцінювання прибутковості інвестицій за допомогою спеціалізованого програмного забезпечення, яке забезпечує зручний доступ до фінансових даних та аналітичних інструментів. У результаті розроблено програму, що дозволяє користувачам завантажувати та аналізувати фінансові дані, проводити розрахунки ключових показників та отримувати зручні графічні інтерпретації результатів.

Програмне забезпечення було реалізовано з використанням сучасних технологій та бібліотек, що забезпечують високу точність та ефективність обробки даних. Використання Python та відповідних бібліотек для аналізу даних, а також Flask для створення веб-інтерфейсу, дозволило забезпечити гнучкість та масштабованість розробленого рішення.

Тестування програмного продукту підтвердило його ефективність і точність, а також відповідність вимогам користувачів, що дозволяє рекомендувати його для практичного використання у сфері фінансового аналіуз. Отримані результати можуть бути застосовані дял створення системи, що дозволяють оптимізувати інвестиційні рішення в різних сферах діяльності.

Розроблене програмне забезпечення сприятиме підвищенню фінансової грамотності та незалежності користувачів, надаючи їм інтуїтивно зрозумілі інструменти для аналізу прибутковості інвестицій та прийняття обґрунтованих рішень. Це також допоможе інвесторам мінімізувати ризики та максимілізувати прибутковість своїх інвестиційних портфелів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Аналіз потреби в інвестиціях [Електронний ресурс] – Режим доступу до ресурсу: https://osvita.ua/vnz/reports/econom_pidpr/19054/.

2. Особливості фінансової глобалізації [Електронний ресурс] – Режим доступу до ресурсу: https://economyandsociety.in.ua/journals/19_ukr/11.pdf.

3. Методичні вказівки до виконання бакалаврських кваліфікаційних робіт для студентів студентів спеціальності 121 "Інженерія програмного забезпечення" / Уклад. О. Н. Романюк, Г. О. Черноволик – Вінниця: ВНТУ, 2022. – 52с.

4. Гладун І.М. Ткаченко О. М. Оцінювання прибутковості інвестицій у фінансові інструменти. Молодь в науці: дослідження, проблеми, перспективи (МН-2024) конференції підрозділів Вінницького національного технічного університету, Вінниця, 2024 [Електронний ресурс] – Режим доступу до ресурсу: https://conferences.vntu.edu.ua/index.php/mn/mn2024/paper/view/21640.

5. Недоліки мобільних систем [Електронний ресурс] – Режим доступу до ресурсу: https://hmilnyk-licey.org.ua/news/10-09-11-18-01-2023/.

6. Robinhood Crypto [Електронний ресурс] – Режим доступу до ресурсу: https://robinhood.com/eu/en/about/crypto/.

7. TD Ameritrade [Електронний ресурс] – Режим доступу до ресурсу: https://www.schwab.com/td-ameritrade

8. Acorns [Електронний ресурс] – Режим доступу до ресурсу: https://www.acorns.com/.

9. Постановки задач бакалаврської дипломної роботи [Електронний ресурс] – Режим доступу до ресурсу: https://studfile.net/preview/14501401/.

10. Аналіз розв'язання оптимізаційних задач [Електронний ресурс] – Режим доступу до ресурсу: https://csc.knu.ua/media/filer_public/c7/83/c783a9ff-5591-4b2c-8ebf-2dce9bd4cb28/analiz_optim_240521_tereshchenko.pdf.

11. Capital Asset Pricing Model [Електронний ресурс] – Режим доступу до ресурсу: https://www.investopedia.com/terms/c/capm.asp.

12. Використання симуляційних моделей у фінансових системах [Електронний ресурс] – Режим доступу до ресурсу: http://194.44.12.92:8080/xmlui/handle/123456789/4840.

13. Основні фактори ОВДП [Електронний ресурс] – Режим доступу до ресурсу: https://index.minfin.com.ua/ua/finance/bonds/.

14. Купонна ставка, як впливає на фінансові системи де вона врахована? [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%9A%D1%83%D0%BF%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BE%D0%B1%D0%BB%D1%96%D0%B3%D0%B0%D1%86%D1%96%D1%8F

15. Термін до погашення [Електронний ресурс] – Режим доступу до ресурсу: https://cbonds.ua/glossary/maturity-date/.

16. Ринкова ціна облігації [Електронний ресурс] – Режим доступу до ресурсу: https://eaf.nmu.org.ua/ua/Students/Metodichka/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%A4%D0%B8%D0%BD%D0%B0%D0%BD%D1%81%D0%BE%D0%B2%D0%B8%D0%B9_%D1%80%D0%B8%D0%BD%D0%BE%D0%BA.pdf.

17. Інфляція [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%84%D0%BB%D1%8F%D1%86%D1%96%D1%8F.

18. Процентні ставки [Електронний ресурс] – Режим доступу до ресурсу: https://minfin.com.ua/ua/deposits/rates/.

19. Кредитний рейтинг емітента [Електронний ресурс] – Режим доступу до ресурсу: https://cbonds.ua/glossary/credit-rating/.

20. C, як мова програмування [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/C_(%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F).

21. R? Що це за мова програмування? [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/R_(%D0%BC%D0%BE%D0%B

2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F).

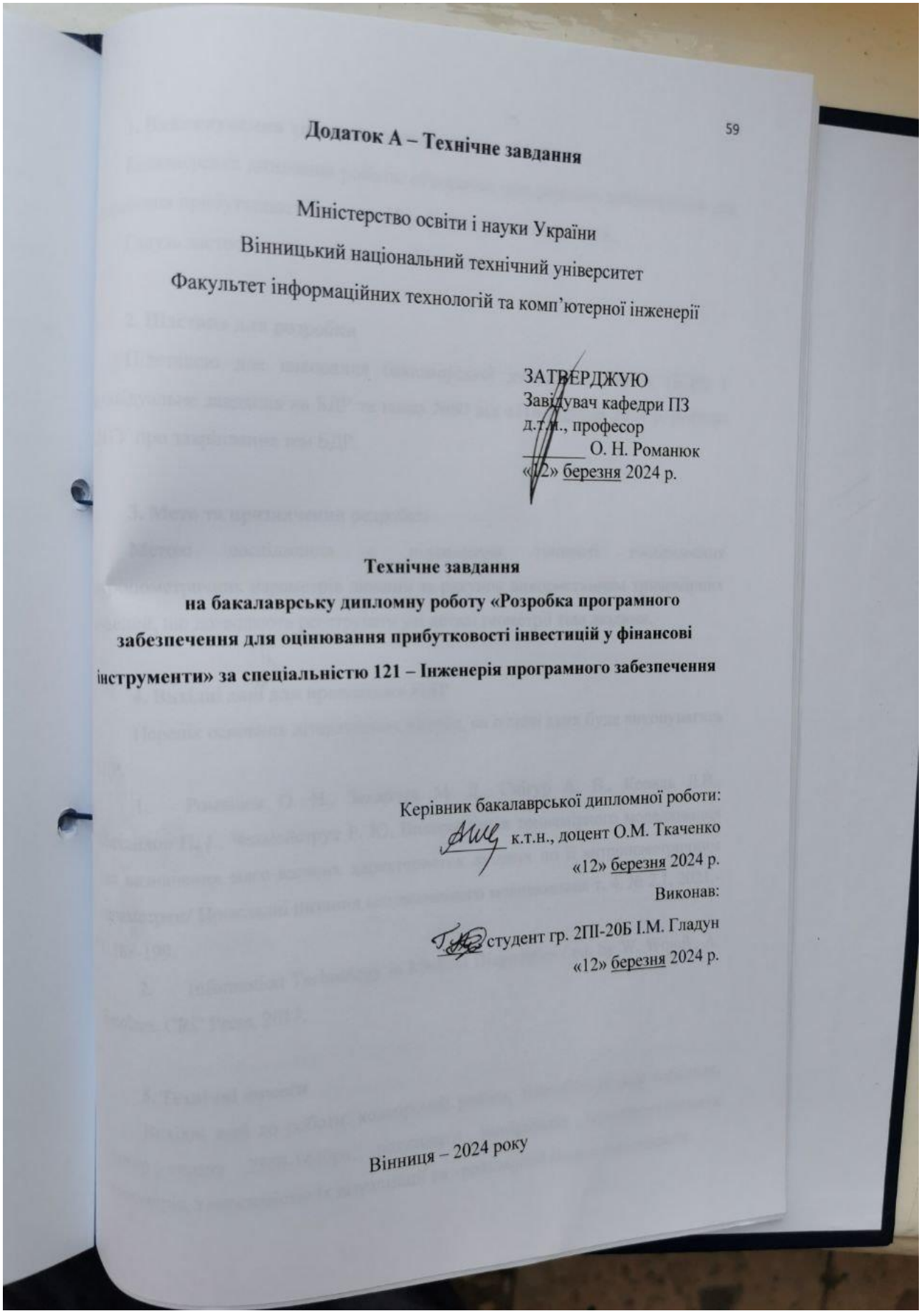
22. C++, унікальність мови [Електронний ресурс] – Режим доступу до ресурсу: https://dou.ua/forums/topic/42888/.

23. Java незамінна мова програмування [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Java.

24. IntelliJ IDEA – Середовище розробки [Електронний ресурс] – Режим доступу до ресурсу: https://javarush.com/ua/groups/posts/uk.2311.stvorennja-proektu-v-intellij-idea.

25. Eclipse у чому переваги? [Електронний ресурс] – Режим доступу до ресурсу: https://foxminded.ua/eclipse-ide/.

ДОДАТКИ



Додаток А – Технічне завдання

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
д.т.н., професор
_____ О. Н. Романюк
«12» березня 2024 р.

Технічне завдання
на бакалаврську дипломну роботу «Розробка програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти» за спеціальністю 121 – Інженерія програмного забезпечення

Керівник бакалаврської дипломної роботи:
к.т.н., доцент О.М. Ткаченко
«12» березня 2024 р.
Виконав:
студент гр. 2ПІ-20Б І.М. Гладун
«12» березня 2024 р.

Вінниця – 2024 року

1. Найменування та галузь застосування

Бакалаврська дипломна робота: «Розробка програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти».

Галузь застосування – медична діагностика.

2. Підстава для розробки

Підставою для виконання бакалаврської дипломної роботи (БДР) є індивідуальне завдання на БДР та наказ №80 від «11» березня 2024 р. ректора ВНТУ про закріплення тем БДР.

3. Мета та призначення розробки

Метою дослідження є підвищення точності вимірювання антропометричних параметрів людини за рахунок використанням тривимірних моделей, що дозволяють реєструвати усі деталі геометрії тіла людини.

4. Вихідні дані для проведення НДР

Перелік основних літературних джерел, на основі яких буде виконуватись БДР.

1. Романюк О. Н., Захарчук М. Д., Снігур А. В., Коваль Л.В., Михайлов П. І., Чехмейструк Р. Ю. Використання тривимірного моделювання для визначення масо-вагових характеристик людини по її антропометричним параметрам/ Прикладні питання математичного моделювання т. 4, № 2.1, 2021.- С.188-199.

2. Information Technology in Medical Diagnostics / ed. by W. Wójcik, A. Smolarz. CRC Press, 2017.

5. Технічні вимоги

Вихідні дані до роботи: кольоровий режим, Truecolor, розмір кординат, розмір екрану 2560-1440px, результати вимірювань антропометричних параметрів, з можливістю їх візуалізації на тривимірній моделі тіла людини.

6. Конструктивні вимоги

Графічна та текстова документація повинна відповідати діючим стандартам України.

7. Перелік технічної документації, що пред'являється по закінченню робіт:

1. Пояснювальна записка до БДР.
2. Технічне завдання.
3. Лістинги програми

8. Вимоги до рівня уніфікації та стандартизації

При розробці програмних засобів слід дотримуватися уніфікації і ДСТУ

9. Стадії та етапи розробки:

№ п\п	Назва етапів бакалаврської дипломної роботи	Строк виконання етапів роботи
1	Аналіз розвитку мобільних систем для оцінювання прибутковості у фінансові інструменти	14.03.24 –22.03.24
2	Математичні моделі для оцінювання прибутковості ОВДП	25.03.24 –19.04.24
3	Розробка програмних модулів реалізації для оцінювання прибутковості у фінансові інструменти	22.04.24 –10.05.24
4	Впровадження, тестування та розробка документації	13.05.24 –24.05.24
5	Оформлення матеріалів до захисту БДР	30.05.24 – 03.06.24

10. Порядок контролю та прийняття

Виконання етапів бакалаврської дипломної роботи контролюється керівником згідно з графіком виконання роботи. Прийняття бакалаврської дипломної роботи здійснюється ДЕК, затвердженою зав. кафедрою згідно з графіком.

62

Додаток Б – Протокол перевірки на плагіат

ПРОТОКОЛ
ПЕРЕВІРКИ БАКАЛАВРСЬКОЇ ДИПЛОМНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Розробка програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти.

Тип роботи: БДР

Підрозділ: кафедра програмного забезпечення, ФІТКІ

Науковий керівник: Ткаченко О.М.

Оригінальність	96,3 %
Схожість	3,7 %

Аналіз звіту подібності

▪ **Запозичення, виявлені у роботі, оформленні коректно і не містять ознак плагіату.**

☐ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цілісності роботи і відсутності самостійності її автора. Роботу направити на доопрацювання.

☐ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховання недобросовісних запозичень.

Особа, відповідальна за перевірку Черноволик Г.О.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck

Автор роботи Гладун І.М.

Керівник роботи Ткаченко О.М.

Додаток В – Лістинг

```
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;

public class MainActivity extends AppCompatActivity {

   private static final String PREFS_NAME = "MyPrefsFile";
   private static final String FIRST_RUN_KEY = "firstRun";

   MediaPlayer mediaPlayerNext, mediaPlayerOption, mediaPlayerTheory, mediaPlayerTrue_2;
   private String selectedTopic = "";

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
```

```
final LinearLayout DMD = findViewById(R.id.DMDLayout);
final LinearLayout PDR = findViewById(R.id.PDRLayout);
final LinearLayout theoryMenu = findViewById(R.id.theoryLayoutMenu);
final Button startQuiz = findViewById(R.id.startQiz);

mediaPlayerNext = MediaPlayer.create(this, R.raw.sounds_1);
mediaPlayerOption = MediaPlayer.create(this, R.raw.sounds_2);
mediaPlayerTheory = MediaPlayer.create(this, R.raw.theory);
mediaPlayerTrue_2 = MediaPlayer.create(this, R.raw.sounds_4);

LinearLayout informationLayout = findViewById(R.id.information);
informationLayout.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      showAlertDialog();
   }
});

SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
boolean firstRun = settings.getBoolean(FIRST_RUN_KEY, true);

if (firstRun) {
   SharedPreferences.Editor editor = settings.edit();
   editor.putBoolean(FIRST_RUN_KEY, false);
   editor.apply();
}

DMD.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
```

```
        mediaPlayerOption.start();
        selectedTopic = "DMD";
        DMD.setBackgroundResource(R.drawable.round_back_stroke10);
        PDR.setBackgroundResource(R.drawable.round_backa_white10);
        theoryMenu.setBackgroundResource(R.drawable.round_backa_white10);
    }
});

PDR.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mediaPlayerOption.start();
        selectedTopic = "PDR";
        PDR.setBackgroundResource(R.drawable.round_back_stroke10);
        DMD.setBackgroundResource(R.drawable.round_backa_white10);
        theoryMenu.setBackgroundResource(R.drawable.round_backa_white10);
    }
});

theoryMenu.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mediaPlayerTheory.start();
        selectedTopic = "theory";
        theoryMenu.setBackgroundResource(R.drawable.round_back_stroke10);
        DMD.setBackgroundResource(R.drawable.round_backa_white10);
        PDR.setBackgroundResource(R.drawable.round_backa_white10);
    }
});
```

```
startQuiz.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if (selectedTopic.isEmpty()) {
            mediaPlayerNext.start();
            Toast.makeText(MainActivity.this,! ", Toast.LENGTH_SHORT).show();
        } else {
            Intent intent;
            if (selectedTopic.equals("DMD")) {
                intent = new Intent(MainActivity.this, DmdActivityMenu.class);
                mediaPlayerTrue_2.start();
            } else if (selectedTopic.equals("PDR")) {
                mediaPlayerTrue_2.start();
                intent = new Intent(MainActivity.this, PdrActivityMenu.class);
            } else if (selectedTopic.equals("theory")) {
                mediaPlayerTrue_2.start();
                intent = new Intent(MainActivity.this, TheoryActivity.class);
            } else {
                Toast.makeText(MainActivity.this, Toast.LENGTH_SHORT).show();
                return;
            }
            startActivity(intent);
            finish();
        }
    }
});


LinearLayout game1Layout = findViewById(R.id.Game_1);
game1Layout.setOnClickListener(new View.OnClickListener() {
```

```
        @Override
        public void onClick(View v) {
            if (getTotalResultFromFile() >= 30) {
                Intent intent = new Intent(MainActivity.this, MenuGameActivity.class);
                startActivity(intent);
            } else {
                Toast.makeText(MainActivity.this, "Для відкриття гри потрібно набрати більше 30 балів!", Toast.LENGTH_SHORT).show();
            }
        }
    });
}

private void showAlertDialog() {
    AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
    builder.setMessage("Виконав Гладун І.М.\nЗастосунок
            .setPositiveButton("OK", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    dialog.dismiss();
                }
            });
    AlertDialog dialog = builder.create();
    dialog.show();
}

private int getTotalResultFromFile() {
    int totalResultFromFile = 0;
    try {
        File file = new File(getFilesDir(), "saving_result.txt");
        if (file.exists()) {
            BufferedReader br = new BufferedReader(new FileReader(file));
```

```
            String line;
            while ((line = br.readLine()) != null) {
                int result = Integer.parseInt(line);
                if (result >= 101) {
                    result = 100;
                } else if (result <= -1) {
                    result = 0;
                }
                totalResultFromFile += result;
            }
            br.close();
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
    return totalResultFromFile;
  }
} package com.example.pdr_ukraine_2024;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.graphics.Color;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;
```

```
import androidx.appcompat.widget.AppCompatButton;

import java.util.List;
import java.util.Timer;
import java.util.TimerTask;

public class QuizActivity extends AppCompatActivity {

    private TextView questions;
    private TextView question;

    private AppCompatButton option1,option2,option3,option4;
    private AppCompatButton nextBtn;

    private Timer quizTimer;
    private boolean quizFinished = false;

    private int seconds = 0;
    private  int totalTimeInMins = 1;

    private List<QuestionsList> questionsList;

    private int currentQuestionPosition = 0;
    private String selectedQptionByUser = "";

    MediaPlayer mediaPlayerNext, mediaPlayerFail, mediaPlayerOption,
mediaPlayerTrue_2,mediaPlayerTrue,mediaPlayerFalse;

    @SuppressLint("SetTextI18n")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_quiz);

final ImageView backBtn = findViewById(R.id.backBtn);
final TextView timer = findViewById(R.id.timer);
final TextView selectedTopicName = findViewById(R.id.selectedTopicName);

mediaPlayerNext = MediaPlayer.create(this,R.raw.sounds_1);
mediaPlayerOption = MediaPlayer.create(this,R.raw.sounds_2);;
mediaPlayerFail = MediaPlayer.create(this,R.raw.sounds_3);
mediaPlayerTrue_2 = MediaPlayer.create(this,R.raw.sounds_4);
mediaPlayerFalse = MediaPlayer.create(this,R.raw.sound_false);
mediaPlayerTrue = MediaPlayer.create(this,R.raw.sound_true);

questions = findViewById(R.id.questions);
question = findViewById(R.id.question);

option1 = findViewById(R.id.option1);
option2 = findViewById(R.id.option2);
option3 = findViewById(R.id.option3);
option4 = findViewById(R.id.option4);

nextBtn = findViewById(R.id.nextBtn);

final String getSelectedTopic = getIntent().getStringExtra("selectedTopic");

selectedTopicName.setText(getSelectedTopic);

questionsList = QuestionsBank.getQuestions(getSelectedTopic);
```

```
        startTimer(timer);

        questions.setText((currentQuestionPosition + 1) + "/" + questionsList.size());
        question.setText(questionsList.get(0).getQuestion());
        option1.setText(questionsList.get(0).getOption1());
        option2.setText(questionsList.get(0).getOption2());
        option3.setText(questionsList.get(0).getOption3());
        option4.setText(questionsList.get(0).getOption4());

        //Зміна зображення у питаннях
        ImageView questionImageView = findViewById(R.id.questionImage);
        int questionImageResource = questionsList.get(currentQuestionPosition).getQuestionImage();
        int imageResource = getResources().getIdentifier(String.valueOf(questionImageResource),
"drawable", getPackageName());
        questionImageView.setImageResource(imageResource);



        backBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                mediaPlayerNext.start();
                quizTimer.purge();
                quizTimer.cancel();

                startActivity(new Intent(QuizActivity.this, MainActivity.class));
                finish();
            }

        });
```

```
        option1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                if (selectedQptionByUser.isEmpty()) {

                    selectedQptionByUser = option1.getText().toString();
                    option1.setBackgroundResource(R.drawable.round_back_rad10);
                    option1.setTextColor(Color.WHITE);

                    revealAswer();

questionsList.get(currentQuestionPosition).setUserSelectedAnswer(selectedQptionByUser);

                    soudForAnswerOption(selectedQptionByUser);
                }

            }
        });


        option2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                if (selectedQptionByUser.isEmpty()) {

                    selectedQptionByUser = option2.getText().toString();
                    option2.setBackgroundResource(R.drawable.round_back_rad10);
                    option2.setTextColor(Color.WHITE);
```

```
                revealAswer();

questionsList.get(currentQuestionPosition).setUserSelectedAnswer(selectedQptionByUser);
                soudForAnswerOption(selectedQptionByUser);
            }
        }
    });

    option3.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {

            if (selectedQptionByUser.isEmpty()) {

                selectedQptionByUser = option3.getText().toString();
                option3.setBackgroundResource(R.drawable.round_back_rad10);
                option3.setTextColor(Color.WHITE);

                revealAswer();

questionsList.get(currentQuestionPosition).setUserSelectedAnswer(selectedQptionByUser);
                soudForAnswerOption(selectedQptionByUser);
            }

        }
    });

    option4.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
```

```
            if (selectedQptionByUser.isEmpty()) {

                selectedQptionByUser = option4.getText().toString();
                option4.setBackgroundResource(R.drawable.round_back_rad10);
                option4.setTextColor(Color.WHITE);

                revealAswer();

questionsList.get(currentQuestionPosition).setUserSelectedAnswer(selectedQptionByUser);
                soudForAnswerOption(selectedQptionByUser);
            }

        }
    });


    nextBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {

            if (selectedQptionByUser.isEmpty()){
                mediaPlayerNext.start();
                Toast.makeText(QuizActivity.this, "Оберіть варіант відповіді",
Toast.LENGTH_SHORT).show();
            }else{
                mediaPlayerTrue_2.start();
                changeNextQuestion();
            }
        }

    });
```

```
}

private void startTimer(final TextView timerTextView) {
    quizTimer = new Timer();
    quizTimer.scheduleAtFixedRate(new TimerTask() {
        @Override
        public void run() {
            if (totalTimeInMins == 0 && seconds == 0) {
                // Перевірка чи тест вже не завершено
                if (!quizFinished) {
                    quizFinished = true; // Встановлюємо прапорець, що тест завершено
                    stopTimer(); // Зупиняємо таймер
                    // Перехід до MainActivity
                    Intent intent = new Intent(QuizActivity.this, MainActivity.class);
                    startActivity(intent);
                    finish();
                }
            } else {
                if (seconds == 0) {
                    totalTimeInMins--;
                    seconds = 59;
                } else {
                    seconds--;
                }

                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {

                        String finalMinutes = String.valueOf(totalTimeInMins);
```

```
                String finalSeconds = String.valueOf(seconds);

                if (finalMinutes.length() == 1) {
                    finalMinutes = "0" + finalMinutes;
                }
                if (finalSeconds.length() == 1) {
                    finalSeconds = "0" + finalSeconds;
                }

                timerTextView.setText(finalMinutes + ":" + finalSeconds);

            }
          });
        }
      }
    }, 1000, 1000);
}

private void stopTimer() {
    if (quizTimer != null) {
        quizTimer.purge();
        quizTimer.cancel();
    }
}

private int getCorrectAnswers(){

    int correctAnswers = 0;

    for (int i = 0; i < questionsList.size() ;i++){
```

```
        final String getUserSelectedAnswer = questionsList.get(i).getUserSelectedAnswer();
        final String getAnswer = questionsList.get(i).getAnswer();

        if(getUserSelectedAnswer.equals(getAnswer)){
            correctAnswers++ ;
        }
    }

    return correctAnswers;
}
private int getInCorrectAnswers(){

    int correctAnswers = 0;

    for (int i = 0; i < questionsList.size() ;i++){
        final String getUserSelectedAnswer = questionsList.get(i).getUserSelectedAnswer();
        final String getAnswer = questionsList.get(i).getAnswer();

        if(!getUserSelectedAnswer.equals(getAnswer)){
            correctAnswers++ ;
        }
    }

    return correctAnswers;
}
@Override
public void onBackPressed() {
    quizTimer.purge();
    quizTimer.cancel();
```

```
    startActivity(new Intent(QuizActivity.this,MainActivity.class));
    finish();
}
private void revealAswer(){

    final String getAnsver = questionsList.get(currentQuestionPosition).getAnswer();

    if (option1.getText().toString().equals(getAnsver)){
        option1.setBackgroundResource(R.drawable.round_back_green10);
        option1.setTextColor(Color.WHITE);
    }
    else if (option2.getText().toString().equals(getAnsver)){
        option2.setBackgroundResource(R.drawable.round_back_green10);
        option2.setTextColor(Color.WHITE);
    }
    else if (option3.getText().toString().equals(getAnsver)){
        option3.setBackgroundResource(R.drawable.round_back_green10);
        option3.setTextColor(Color.WHITE);
    }
    else if (option4.getText().toString().equals(getAnsver)){
        option4.setBackgroundResource(R.drawable.round_back_green10);
        option4.setTextColor(Color.WHITE);
    }

}
private void changeNextQuestion(){
    currentQuestionPosition ++;
    if ((currentQuestionPosition+1 )== questionsList.size()) {
        nextBtn.setText("Готово");
    }
```

```
    if (currentQuestionPosition < questionsList.size()) {
        selectedQptionByUser = "";

        option1.setBackgroundResource(R.drawable.round_back_white_stroke2_10);
        option1.setTextColor(Color.parseColor("#1F6BB8"));

        option2.setBackgroundResource(R.drawable.round_back_white_stroke2_10);
        option2.setTextColor(Color.parseColor("#1F6BB8"));

        option3.setBackgroundResource(R.drawable.round_back_white_stroke2_10);
        option3.setTextColor(Color.parseColor("#1F6BB8"));

        option4.setBackgroundResource(R.drawable.round_back_white_stroke2_10);
        option4.setTextColor(Color.parseColor("#1F6BB8"));

        questions.setText((currentQuestionPosition + 1) + "/" + questionsList.size());
        question.setText(questionsList.get(currentQuestionPosition).getQuestion());
        option1.setText(questionsList.get(currentQuestionPosition).getOption1());
        option2.setText(questionsList.get(currentQuestionPosition).getOption2());
        option3.setText(questionsList.get(currentQuestionPosition).getOption3());
        option4.setText(questionsList.get(currentQuestionPosition).getOption4());


        ImageView questionImageView = findViewById(R.id.questionImage);
        int questionImageResource =
questionsList.get(currentQuestionPosition).getQuestionImage();
        int imageResource = getResources().getIdentifier(String.valueOf(questionImageResource),
"drawable", getPackageName());
        questionImageView.setImageResource(imageResource);
    } else {
        Intent intent = new Intent(QuizActivity.this, QuizResults.class);
```

```
            intent.putExtra("correct", getCorrectAnswers());
            intent.putExtra("incorrect", getInCorrectAnswers());

            startActivity(intent);
            finish();
        }
    }
    private void soudForAnswerOption(String selectedQptionByUser){


        final String getAnswer = questionsList.get(currentQuestionPosition).getAnswer();

        if(selectedQptionByUser.equals(getAnswer)) {
            mediaPlayerTrue.start();
        }else{
            mediaPlayerFalse.start();
        }
    }
}
```

Додаток Г – Графічна частина

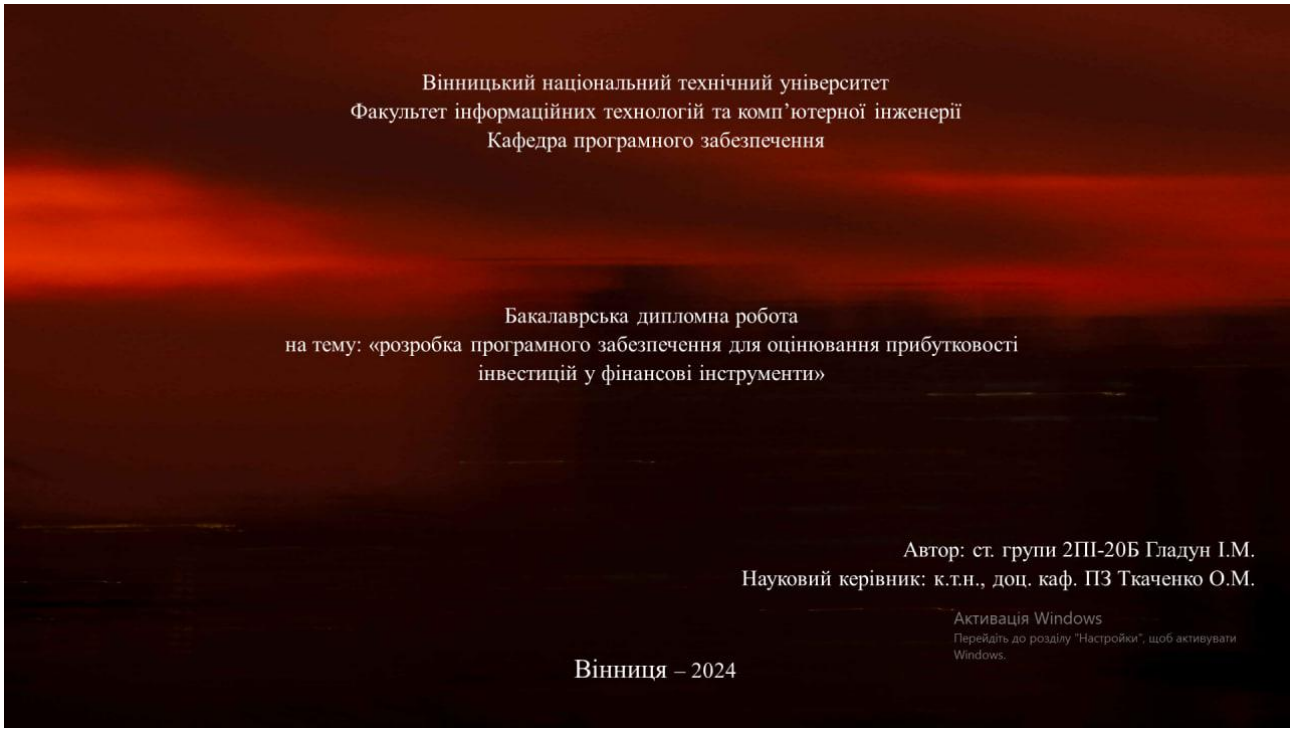


Рисунок Г.1 – Назва роботи

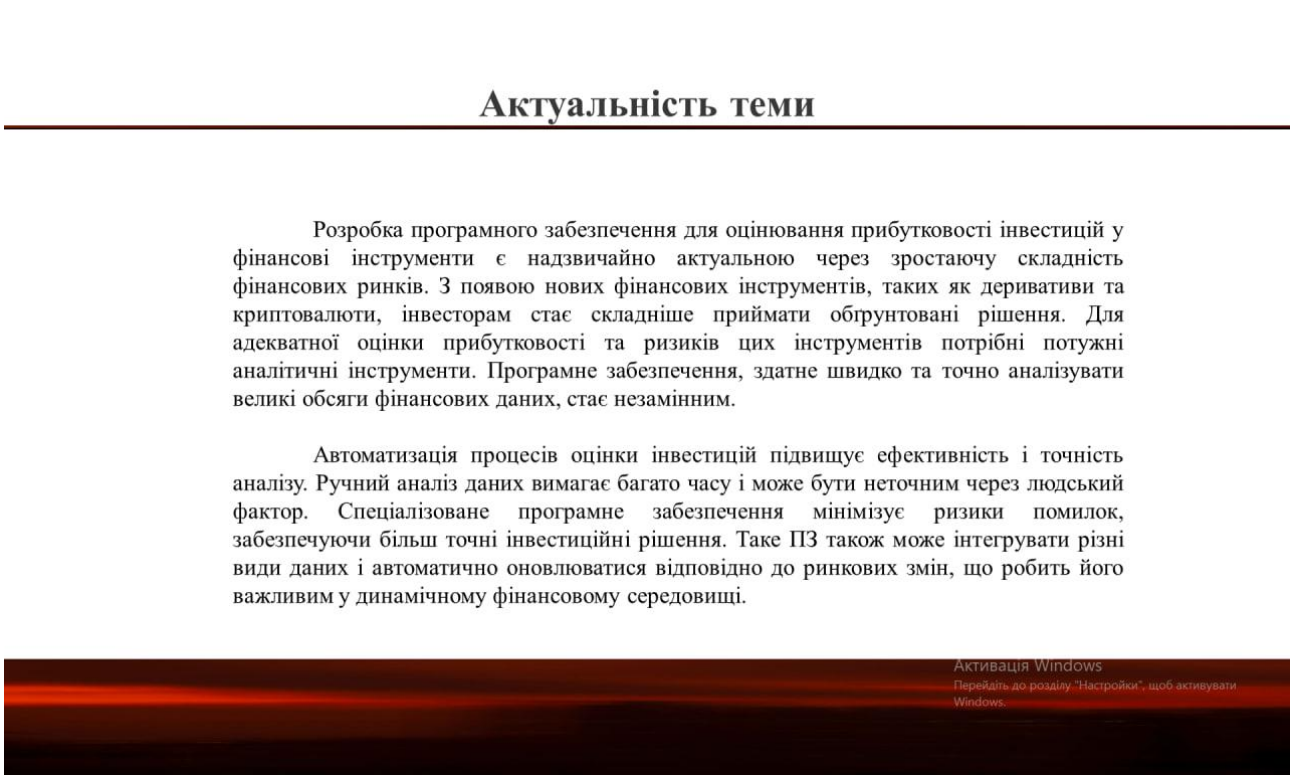


Рисунок Г.2 – Актуальність теми

Мета, об'єкт та предмет дослідження

1. **Мета роботи** полягає у підвищенні якості управління інвестиційними портфелями шляхом розробки та впровадження програмного забезпечення для оцінювання прибутковості фінансових інструментів.

2. **Об'єктом дослідження** є процес управління інвестиційними портфелями та методи оцінювання фінансових інструментів.

3. **Предметом дослідження** є розробка програмного забезпечення, яке дозволяє автоматизувати оцінювання прибутковості різних фінансових інструментів, аналізувати ризики та вигоди, а також надавати рекомендації для оптимізації інвестиційних рішень.

Рисунок Г.3 – Мета, об’єкт та предмет дослідження

Завдання на бакалаврську дипломну роботу

1. Створення програмного продукту, який забезпечить можливість оцінювання прибутковості різних фінансових інструментів, а також аналізу ризиків і вигод.
2. Проведення тестів програмного забезпечення для перевірки його ефективності, точності та надійності. Валідація здійснюється порівнянням результатів роботи програми з реальними ринковими даними та аналізом відповідності отриманих результатів вимогам інвесторів.
3. Впровадження розробленого програмного забезпечення на практиці, навчання користувачів його використанню та надання підтримки.
4. Аналіз результатів використання програмного забезпечення для визначення його впливу на якість управління інвестиціями та досягнення мети дослідження.

Рисунок Г.4 – Завдання на бакалаврську дипломну роботу

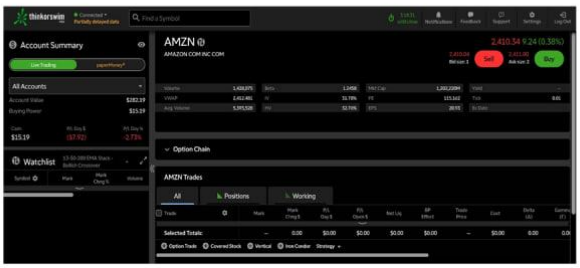
Наукова новизна отриманих результатів та практична цінність

1. **Удосконалено** метод для прогнозування ринкових тенденцій, який покращує точність прогнозів та дозволяє інвесторам приймати більш обґрунтовані рішення. На відміну від існуючих методів, новий підхід враховує додаткові параметри, такі як аномальні зміни на ринку, що забезпечує більш адаптивний і точний аналіз.
2. **Отримав подальшого розвитку** метод оцінювання прибутковості інвестцій, який, на відміну від існуючих, враховує багатофакторний аналіз ризиків та можливостей, зокрема макроекономічні показники, галузеві тренди та специфічні характеристики фінансових інструментів, що дозволило підвищити точність оцінювання прибутковості фінансових інструментів.

Рисунок Г.5 – Наукова новизна отриманих результатів та практична цінність

Порівняльний аналіз аналогів

«Robinhood»



TD Ameritrade

Acorns

Рисунок Г.6 – Порівняльний аналіз аналогів

Порівняльний аналіз аналогів

Функціонал	Robinhood	TD Ameritrade	Acorns	Розроблюване ПЗ
Безкоштовна торгівля	+	–	–	+
Аналітика ринку	–	+	–	+
Автоматизовані інвестиції	–	+	+	+
Вартість управління портфелем	+	+	+	–
Безпека даних	–	+	–	+

Рисунок Г.7 – Порівняльний аналіз аналогів

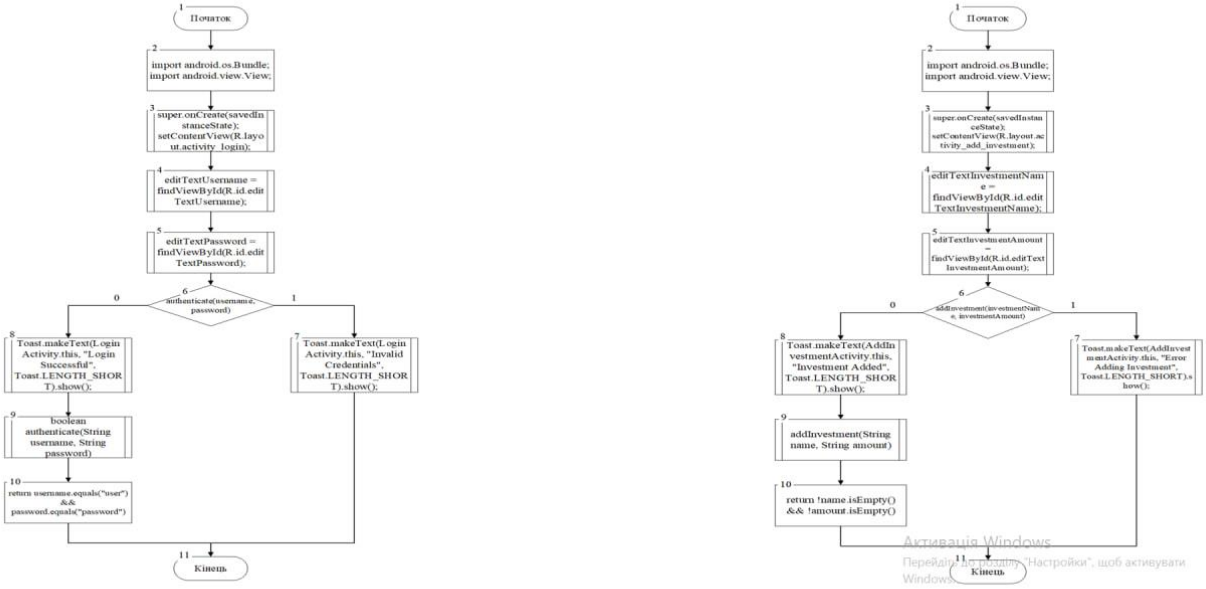
Засоби для реалізації завдань

Android Studio

Java

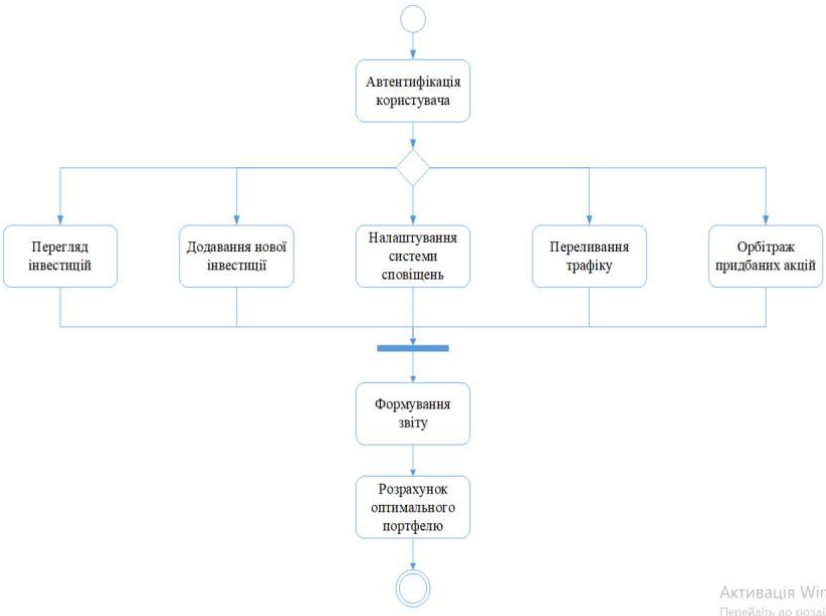
Gradle

Рисунок Г.8 – Засоби для реалізації завдань



Блок-схема алгоритму авторизації **Блок-схема додавання інвестиції**

Рисунок Г.9 – Блок-схеми алгоритму авторизації та додавання інвестиції



UML-діаграма діяльності розробленої системи

Рисунок Г.10 – UML-діаграма

Висновки

Бакалаврська диломна робота присвячена розробці програмного забезпечення для оцінювання прибутковості інвестицій у фінансові інструменти. У ході дослідження було досягнуто мети створення ефективного інструменту, що дозволяє користувачам без глибоких знань у сфері фінансів легко аналізувати та прогнозувати прибутковість інвестицій.

Програмне забезпечення було реалізовано з використанням сучасних технологій та бібліотек, що забезпечують високу точність та ефективність обробки даних. Використання Python та відповідних бібліотек для аналізу даних, а також Flask для створення веб-інтерфейсу, дозволило забезпечити гнучкість та масштабованість розробленого рішення.

Розроблене програмне забезпечення сприятиме підвищенню фінансової грамотності та незалежності користувачів, надаючи їм інтуїтивно зрозумілі інструменти для аналізу прибутковості інвестицій та прийняття обґрунтованих рішень. Це також допоможе інвесторам мінімізувати ризики та максимілізувати прибутковість своїх інвестиційних портфелів.

Рисунок Г.11 – Висновки

Апробація та публікація результатів бакалаврської дипломної роботи

Апробація та публікація результатів бакалаврської дипломної роботи. Результати бакалаврської дипломної роботи доповідалися на конференції Молодь в науці: дослідження, проблеми, перспективи (МН-2024) і опубліковані в тезах доповіді цієї конференції.

Публікації.

Гладун І.М. Оцінювання прибутковості інвестицій у фінансові інструменти/ І. М. Гладун, О. М. Ткаченко // Молодь в науці: дослідження, проблеми, перспективи (МН-2024) конференції підрозділів Вінницького національного технічного університету, Вінниця, 2024.

Рисунок Г.12 – Апробація та публікація результатів бакалаврської дипломної роботи