

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
«КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА»
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
«ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ В ЕКОНОМІЦІ»**

Кафедра інформаційних систем в економіці

галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНИЙ БАКАЛАВРСЬКИЙ ПРОЕКТ

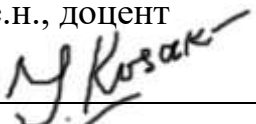
на тему

**ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ
«КРИПТОІНВЕСТИЦІЇ»**

студента Попова Дмитра Сергійовича _____

Науковий керівник:

к.е.н., доцент

_____ Козак І.А.

**Кваліфікаційний бакалаврський
проект допущений до захисту в
Екзаменаційній комісії з атестації
здобувачів вищої освіти**

в.о. завідувача кафедри

_____ Тішков Б.О.

Київ 2024

Міністерство освіти і науки України
Київський національний економічний університет імені Вадима Гетьмана
Навчально-науковий інститут «Інститут інформаційних технологій в економіці»

Кафедра інформаційних систем в економіці

ОСВІТНЬО_ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»

галузь знань 12 «Інформаційні технології»

спеціальність 122 «Комп'ютерні науки»

ПОГОДЖЕНО:

Керівник проектної групи(гарант)
освітньо-професійної програми

Іванченко Г.Ф.
“ ” 2024 р.

ЗАТВЕРДЖУЮ:

Завідувач кафедри
Тішков Б.О.

“ ” 2024 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувача вищої освіти *Попова Дмитра Сергійовича*

очної (денної) форми навчання

на підготовку кваліфікаційного бакалаврського проекту
на тему: «Проектування інформаційної системи «Криптоінвестиції»»

Тему затверджено наказом ректора Університету від «__» ____ 2024 р. № 332-ст.

Кваліфікаційний бакалаврський проект виконується на матеріалах дослідження предметної області та літературних джерел

План кваліфікаційного бакалаврського проекту

Розділ I Характеристика та аналіз предметної області

Розділ II Розробка вимог і моделювання інформаційної системи

Розділ III Проектування та реалізація компонентів системи

Об'єкт дослідження це процес або явище, що обрані для дослідження, зокрема методика і технологія створення ІС для криптоінвестицій, строго визначивши функції системи

Предмет дослідження частина об'єкту, що розглядаються в рамках даного проекту, зокрема інформаційні процеси, комп'ютерні технології з інвестицій у криптовалюту

Мета кваліфікаційного бакалаврського проекту провести дослідження і виконати проектні рішення з розроблення інформаційної системи криптоінвестицій

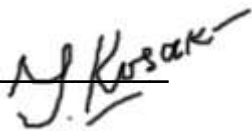
Конкретні завдання, які студент повинен виконати для досягнення поставленої мети:

У розділі I необхідно дати характеристику галузі – криптоінвестиції, визначити структуру об'єкта управління та функції її складових, особливу увагу слід приділити факторам, що впливають на організацію інформаційної системи, доцільно навести схему організаційної структури.

У розділі II провести аналіз і специфікацію вимог до інформаційної системи з криптоінвестицій. Дотримуючись вимог стандартів, виконати постановку та алгоритм розв'язання задач з автоматизації функцій криптоінвестування. Розробити інформаційну модель підтримки управління системою у відповідності до вимог, викладених методичних вказівках кафедри.

У розділі III виконати проектування таких компонентів системи: інформаційного, технічного та програмного забезпечення у відповідності з методичними рекомендаціями кафедри, вказати можливості і доцільності впровадження. У додатках обов'язково навести вихідні документи та запити чи програми.

Завдання підготував
науковий керівник _____



Козак Ірина Антонівна

“__13__” березня 2024 р.

Завдання одержав
студент _____

Попов Дмитро Сергійович

“_____” _____ 2024 р.

РЕЦЕНЗІЯ

на кваліфікаційний бакалаврський проєкт

здобувача вищої освіти

Попова Дмитра Сергійовича

(прізвище, ім'я, по батькові)

Тема Проектування інформаційної системи «криптоінвестиції»

Актуальність теми кваліфікаційного бакалаврського проєкту і доцільність його розроблення. В умовах цифрової економіки криптовалюти займають дедалі важливіше місце в інвестиційних портфелях як приватних осіб, так і інституційних інвесторів. Проектування інформаційної системи, яка б дозволяла автоматизувати процеси управління криптоінвестиціями, є своєчасним та важливим завданням.

Якість проведеного дослідження. Студент детально проаналізував сучасний стан ринку криптовалют, визначив ключові проблеми та потреби інвесторів. Проведено порівняння існуючих інформаційних систем, що використовуються у цій галузі, з виявленням їх сильних та слабких сторін. Розроблена концепція інформаційної системи є ґрунтовною та враховує всі ключові аспекти.

Позитивні риси кваліфікаційного бакалаврського проєкту. До позитивних рис кваліфікаційного бакалаврського проєкту слід віднести його чітку структуру та логічну послідовність викладу матеріалу. Автор вдало використав архітектурний шаблон MVC (Model-View-Controller) та реляційну базу даних MySQL для забезпечення надійності та масштабованості системи. Розроблений прототип системи в онлайн-сервісі Figma демонструє високий рівень опрацювання користувацького інтерфейсу та основних функцій.

Зауваження Серед зауважень можна виділити необхідність більш детального опису алгоритмів забезпечення безпеки даних та захисту від можливих кібератак. Також варто було б розглянути можливість використання альтернативних технологій та платформ, що могло б забезпечити більшу гнучкість та надійність системи.

Практична значимість висновків і рекомендацій. Розроблену систему можна впроваджувати в брокерські компанії та інвестиційні фонди, які спеціалізуються на криптовалютах. Розроблена система може суттєво підвищити ефективність інвестиційної діяльності, що підтверджується позитивними результатами тестів.

Місце роботи та посада рецензента КНП «ДКЛ №4 Солом'янського р. м. Києва»

Науковий ступінь, учене звання (за наявності) Каплун М. М.

(підпис, ПІБ)

Підпис засвідчую: в. о. директора

(посада, підпис)

Місце печатки організації, де працює рецензент

ВІДГУК

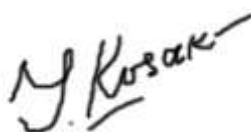
наукового керівника про кваліфікаційний бакалаврський проект
студента Навчально-наукового інституту
“Інститут інформаційних технологій в економіці”
галузі знань 12 “Інформаційні технології”
спеціальності 122 “Комп’ютерні науки”
Попова Дмитра Сергійовича

на тему: «Проектування інформаційної системи «Криптоінвестиції»»

- 1. Актуальність теми:** Підготовлений студентом кваліфікаційний бакалаврський проект присвячений проектуванню інформаційної системи, що спрямована на автоматизацію інвестиційної діяльності в криптовалютах, підвищення її ефективності та безпеки для кінцевих користувачів. Тема є актуальною.
- 2. Позитивні риси кваліфікаційного бакалаврського проекту:** Студентом самостійно здійснено проектування та розробку інформаційної системи «Криптоінвестиції».
- 3. Наявність самостійних розробок:** було запроектовано та розроблено інформаційну систему для кваліфікаційного проекту, студент проявив себе як сформований спеціаліст.
- 4. Наявність недоліків:** система може бути в подальшому вдосконалена за рахунок покращення та розширення функціональних можливостей.
- 5. Загальна оцінка кваліфікаційного бакалаврського проекту та його допущення до захисту перед Екзаменаційною комісією:** кваліфікаційний бакалаврський проект студента Попова Д.С. відповідає встановленим вимогам, рекомендується до захисту і заслуговує високої позитивної оцінки.

Науковий керівник

доцент, кандидат економічних наук



Козак І. А.

ЗГОДА

на обробку персональних даних

Я, Попов Дмитро Сергійович
(прізвище, ім'я, по батькові)

народився «17» листопада 2024 року, здобувач першого (бакалаврського) рівня вищої освіти за освітньою програмою «Комп'ютерні науки» кафедри інформаційних систем в економіці **денної** форми навчання ННІ «Інститут інформаційних технологій в економіці» Державного вищого навчального закладу «Київський національний економічний університет імені Вадима Гетьмана», відповідно до Закону України «Про захист персональних даних» від 1 червня 2010 року № 2297-УІ, шляхом підписання цього тексту, даю згоду ДВНЗ «Київський національний економічний університет імені Вадима Гетьмана» та Міністерству освіти і науки України на:

- обробку моїх персональних даних: прізвище, ім'я, по-батькові, дату народження, місце та форму навчання, з метою забезпечення реалізації адміністративно-правових відносин, відносин у сфері бухгалтерського і податкового обліку пов'язаних зі здійсненням заходів щодо виявлення та запобігання академічному плагіату;
- зберігання кваліфікаційної роботи в електронному архіві Університету (iRKNEU), розміщення її в бібліотечних репозитаріях та в інформаційних базах даних, що використовуються для перевірки робіт на наявність академічного плагіату.

Мої персональні дані, на обробку яких я даю цю згоду, можуть бути передані третім особам тільки у випадках, передбачених законодавством України.

«13» червня 2024 року  Попов Д. С.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

кваліфікаційного бакалаврського проекту студента 4 курсу

Навчально-наукового інституту

«Інститут інформаційних технологій в економіці»

Попова Дмитра Сергійовича, виконаного на тему:

«Проектування інформаційної системи «Криптоінвестиції»»

Київ: кафедра інформаційних систем в економіці, 2024 р.

Кваліфікаційний бакалаврський проект присвячений проектуванню інформаційної системи, що спрямована на автоматизацію інвестиційної діяльності в криптовалютах, підвищення її ефективності та безпеки для кінцевих користувачів. У роботі проведено аналіз сучасного стану ринку криптовалют, визначено ключові проблеми та потреби інвесторів.

Розроблена концепція інформаційної системи враховує потреби користувачів у зручному та безпечному доступі до інструментів аналізу ринку, управління інвестиційним портфелем та здійснення операцій з криптовалютними активами. Вона включає в себе систему реєстрації та аутентифікації користувачів, особистий кабінет інвестора, модуль звітності, реферальну програму та модуль зворотного зв'язку.

Запропоновано використання архітектурного шаблону MVC (Model-View-Controller) та реляційної бази даних MySQL для забезпечення надійності та масштабованості системи. Розроблено прототип системи в онлайн-сервісі Figma, що демонструє користувацький інтерфейс та основні функції.

Практичне значення проекту полягає у можливості впровадження розробленої системи в брокерські компанії та інвестиційні фонди, спеціалізовані на криптовалютах. Ефективність розробленої системи підтверджена позитивними результатами тестування та віртуальної експлуатації.

РЕФЕРАТ

Кваліфікаційний бакалаврський проект містить 133 сторінки, 9 таблиць, 65 рисунків, перелік джерел посилань з 33 найменувань, 4 додатки.

Проектування інформаційної системи «Криптоінвестиції»

Перелік ключових слів: *Комп'ютерна система, Інформаційна система, Binance Smart Chain, Операційна система, API, Система керування базами даних, UML, діаграма, Microsoft SQL Server Management Studio, прототипування, Figma, SySML, тест-кейси, функціональні вимоги, нефункціональні вимоги, бізнес-цілі...*

Об'єктом дослідження є процес управління криптоінвестиціями в умовах цифрової економіки.

Мета кваліфікаційного бакалаврського проекту полягає у створенні інформаційної системи, призначеної для автоматизації інвестиційної діяльності в криптовалютах, підвищення її ефективності та безпеки для кінцевих користувачів.

Методами розроблення кваліфікаційного бакалаврського проекту є створення інформаційної системи в середовищі розробки IntelliJ IDEA[2] та Microsoft Visual Code і використанні його зручного функціоналу в виді додаткових функцій.

Апаратура використана при розробленні кваліфікаційного бакалаврського проекту є персональним комп'ютером:

- Процесор Intel(R) Xeon(R) CPU E5-2660 V3 3,10GHz;
- Оперативна пам'ять – 16 Гб;
- Накопичувач – M2 SSD 128 Гб, SATA SSD 1Тб, HDD 1 Тб;
- Екран – IPS 25", IPS 27".

Отримані результати представляють новизну в розробці адаптивних модульних систем для криптоінвестицій.

Одержані результати можуть бути впроваджені у брокерські компанії та інвестиційні фонди, спеціалізовані на криптовалютах. Ефективність розробленої системи підтверджена позитивними результатами тестування та пілотної експлуатації.

Рік виконання кваліфікаційного бакалаврського проекту – 2024. Рік захисту кваліфікаційного бакалаврського проекту – 2024.

ЗМІСТ

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ	
РЕЦЕНЗІЯ	
ВІДГУК	
ЗГОДА	
АНОТАЦІЯ	
РЕФЕРАТ	
ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ	
ВСТУП	3
РОЗДІЛ 1	6
ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ	6
1.1 Характеристика предметної області та об'єкта дослідження	6
1.2 Аналіз існуючих рішень	7
1.2.1 Дослідження існуючих інформаційних систем для обраної предметної галузі	7
1.2.2 Порівняння програмних систем	11
1.2.3 Позитивні характеристики існуючих систем, що їх слід відтворити у проектних рішеннях	13
1.2.4 Недоліки існуючих систем та напрями їх удосконалення	13
1.3 Розробка концепції інформаційної системи	14
РОЗДІЛ 2	21
РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ..	21
2.1 Аналіз і специфікація вимог до інформаційної системи	21
2.1.1 Бізнес-вимоги до системи	22
2.1.2 Функціональні вимоги до системи	23
2.1.3 Нефункціональні вимоги до системи	26
2.2 Обґрунтування методології проектування та функціональна модель задачі	28
2.3 Моделювання предметної галузі	29
2.4 Характеристика задачі	33
2.4.1 Вихідна інформація	38
2.4.2 Вхідна інформація	38
2.5 Математичне забезпечення та алгоритм функціонування системи	40
2.5.1 Математичний опис	40
2.5.2 Алгоритм розв'язання задачі	42

2.6	Моделювання поведінки системи	46
2.6.1	Моделювання структури системи	54
2.6.2	Моделювання користувацького інтерфейсу.....	58
РОЗДІЛ 3		61
ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ		61
3.1	Інформаційне забезпечення.....	61
3.2	Організаційне забезпечення	67
3.3	Програмне забезпечення.....	68
3.4	Технічне забезпечення	72
3.5	Результати реалізації інформаційної системи	73
РОЗДІЛ 4		75
ТЕСТУВАННЯ І ВЕРИФІКАЦІЯ ПРОЕКТНИХ РІШЕНЬ		75
4.1	Розробка тест-кейсів.....	75
4.2	Трасування вимог до системи	76
ВИСНОВКИ		80
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ		83
ДОДАТКИ		
ДОДАТОК А		
	Тестова документація, підготовлена засобами Enterprise Architect.....	
ДОДАТОК Б		
	Фрагменти коду, що було створено для реалізації бекенду системи (контролери)	
	HTML код сторінки особистого кабінету:	
	Фрагмент CSS коду:.....	
	Фрагмент JS коду, що відповідає за анімації переходу, задній фон, слайдери та кнопку меню у мобільній версії:	
ДОДАТОК В		
	Сторінки прототипу системи зроблені у Figma[18]	
ДОДАТОК Г		
Схожість		

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД	База Даних
ІС	Інформаційна система
КС	Комп'ютерна система
BSC	Binance Smart Chain
ОС	Операційна система
API	Application Programming Interface
СКБД	Система керування базами даних
UML	Unified Modeling Language
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheets
SSMS	Microsoft SQL Server Management Studio
BDD	Block definition diagrams
IBD	Internal Block Diagram
MVC	Model View Controller
RAD	Rapid Application Development
SQL	Structured Query Language

ВСТУП

Сьогоднішній світ переживає стрімкий розвиток цифрових технологій, які охоплюють усі сфери людської діяльності, включаючи фінансову галузь. Особливо вражаючим є стрімке зростання ринку криптовалют, який за останні роки перетворився з локального явища на важливий сегмент світової економіки[9]. Криптовалюти пропонують нові можливості для інвестування, але, разом з тим, ставлять перед інвесторами нові виклики, пов'язані з волатильністю ринку, складністю аналізу та управління ризиками.

В умовах зростаючого інтересу до криптовалют та збільшення кількості інвесторів, виникає нагальна потреба в розробці ефективних інформаційних систем. Такі системи повинні надавати користувачам комплексні інструменти для аналізу ринку, управління інвестиціями та мінімізації ризиків, бути зручними, безпечними, надійними та адаптованими до динамічних змін криптовалютного ринку. Особливо актуальним є створення таких систем для українського ринку, який характеризується високим потенціалом розвитку в сфері криптовалют.

Метою даного бакалаврського проекту є проектування інформаційної системи для крипто інвестицій, яка б забезпечувала користувачам зручний та безпечний доступ до інструментів управління інвестиційним портфелем та здійснення операцій з криптовалютними активами. Для досягнення поставленої мети було визначено наступні завдання:

1. Провести аналіз предметної області, дослідити особливості криптовалютного ринку, його тенденції, виклики та потреби інвесторів.
2. Дослідити існуючі програмні рішення для криптоінвестицій, виявити їх переваги та недоліки, визначити кращі практики та напрямки для вдосконалення.
3. Розробити концепцію інформаційної системи, визначити її основні функції та можливості, враховуючи потреби інвесторів та сучасні тенденції розвитку криптовалютного ринку.

4. Визначити функціональні та нефункціональні вимоги до системи, детально описати очікувану поведінку системи, її характеристики та критерії якості.

5. Змодельовати поведінку, структуру та користувацький інтерфейс системи, використовуючи сучасні методи та інструменти моделювання програмного забезпечення.

6. Розробити прототип інформаційної системи, який би демонстрував основні функції та можливості системи, а також дозволяв проводити тестування та отримувати зворотний зв'язок від користувачів.

7. Обрати та обґрунтувати технології й інструменти для реалізації системи, враховуючи вимоги до продуктивності, безпеки, масштабованості та зручності використання.

8. Розробити тестову документацію та провести тестування системи, перевірити її функціональність, відповідність вимогам та надійність.

9. Забезпечити трасування вимог до системи, відстежити їх реалізацію на всіх етапах розробки та переконатися в їх повному виконанні.

Об'єктом дослідження є процес управління криптоінвестиціями в умовах цифрової економіки. Предметом дослідження є інформаційна система для крипто інвестицій, що забезпечує ефективне управління криптовалютними активами, аналіз ринку та мінімізацію ризиків.

В процесі виконання проекту було використано комплекс методів дослідження, включаючи аналіз літературних джерел та існуючих рішень, методи моделювання програмного забезпечення, прототипування, тестування та трасування вимог.

Теоретична значущість полягає в тому, що проект розширює знання про особливості розробки інформаційних систем для криптовалютного ринку та пропонує нові підходи до управління криптоінвестиціями. Методична значущість полягає в тому, що в процесі розробки були використані сучасні методи та інструменти моделювання, прототипування та тестування програмного забезпечення, які можуть бути застосовані для розробки інших

інформаційних систем. Практична значущість полягає в тому, що розроблена система може бути використана інвесторами для ефективного управління криптовалютними активами, аналізу ринку та мінімізації ризиків.

Результати дослідження, отримані в рамках даного проекту, були представлені на студентській науково-практичній конференції. Інформаційну базу дослідження склали праці вітчизняних та зарубіжних науковців в галузі криптовалют, інвестицій та розробки програмного забезпечення, статистичні дані офіційних сайтів, звіти рейтингових агентств, фінансова звітність компаній та інші джерела інформації.

РОЗДІЛ 1

ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Характеристика предметної області та об'єкта дослідження

Основна мета цієї інформаційної системи полягає в ефективному управлінні криптоінвестиціями, що включає в себе збір, аналіз, інтерпретацію та зберігання даних, пов'язаних з криптовалютними активами. Задача полягає у виявленні трендів ринку, визначенні потенційно прибуткових інвестиційних можливостей та управлінні ризиками. Розуміння принципів роботи криптовалюти допоможуть в цьому[1]. Це включає аналіз ринкових даних, таких як ціни, обсяги торгів, історія вартості криптовалют, а також вивчення новин і тенденцій, які можуть вплинути на ринок.

Задачі охоплюють[4]:

- Аналіз фінансових даних. Однією з основних задач оброблення інформації в системі криптоінвестицій є аналіз та обробка фінансових даних. Це може включати попереднє визначення показників успішності криптовалютних активів, ризиків, тенденцій ринку, дивідендної доходності тощо.
- Прогнозування та моделювання. Збирання, обробка та аналіз історичних даних для створення моделей прогнозування цінових та ринкових рухів криптовалют може бути важливою частиною системи. Це дозволить інвесторам приймати кращі рішення.
- Оцінка ризиків[5]. Обробка інформації для ідентифікації потенційних ризиків інвестування в криптовалюти, врахування факторів невизначеності та розробка стратегій управління ризиками.
- Оптимізація портфеля. Аналіз та оптимізація портфеля криптовалютних активів на основі отриманих даних для досягнення максимальної ефективності та мінімізації ризиків.

- Створення звітів та аналітика. Розробка звітів, аналітика результатів, що дозволяють інвесторам та менеджерам розуміти поточний стан ринку, результати своїх інвестицій та приймати відповідні рішення.

- Підтримка прийняття рішень. Забезпечення інвесторів необхідною інформацією та аналітикою для прийняття обґрунтованих рішень стосовно купівлі, продажу чи утримання криптовалютних активів.

Ця інформаційна система повинна надавати інвесторам різні плани та інвестиційні програми, дозволяючи їм приймати обґрунтовані рішення та оптимізувати свої портфелі з метою максимізації прибутку та мінімізації ризиків.

1.2 Аналіз існуючих рішень

Ринок криптовалют стає все більш динамічним та складним, що призводить до появи різноманітних програмних рішень, спрямованих на надання аналітичної, технічної та стратегічної підтримки для інвесторів та фахівців з фінансів[8]. Тому проаналізуємо характеристики чотирьох відомих програмних систем, що використовуються для аналізу та управління криптоінвестиціями.

1.2.1 Дослідження існуючих інформаційних систем для обраної предметної галузі

Binance – це одна з найбільших у світі криптовалютних бірж. Заснована в 2017 році Чанпенг Чжао та іншими співзасновниками, Binance швидко стала популярною серед трейдерів через свою широку лінійку криптовалютних пар, низькі комісії, високу швидкість обробки та інноваційні функції[11].

Платформа пропонує різноманітні послуги, такі як купівля/продаж криптовалют, ф'ючерси, опціони, стейкінг, позики, а також власна криптовалютна картка. Binance також відома своїм власним токеном Binance Coin (BNB), який використовується для знижок на комісії та інших переваг на платформі.

Крім торгівлі, Binance розвивається у напрямку блокчейн-технологій, запускаючи власну блокчейн-мережу Binance Smart Chain (BSC), яка надає можливість розробникам створювати децентралізовані додатки (DApps) та розвивати фінансові продукти.

Однак Binance також відома своєю регуляторною ситуацією, оскільки компанія періодично зіштовхується з регуляторами в різних країнах через питання ліцензування та дотримання правил фінансового регулювання. Головну сторінку Binance можна побачити на рисунку 1.1.

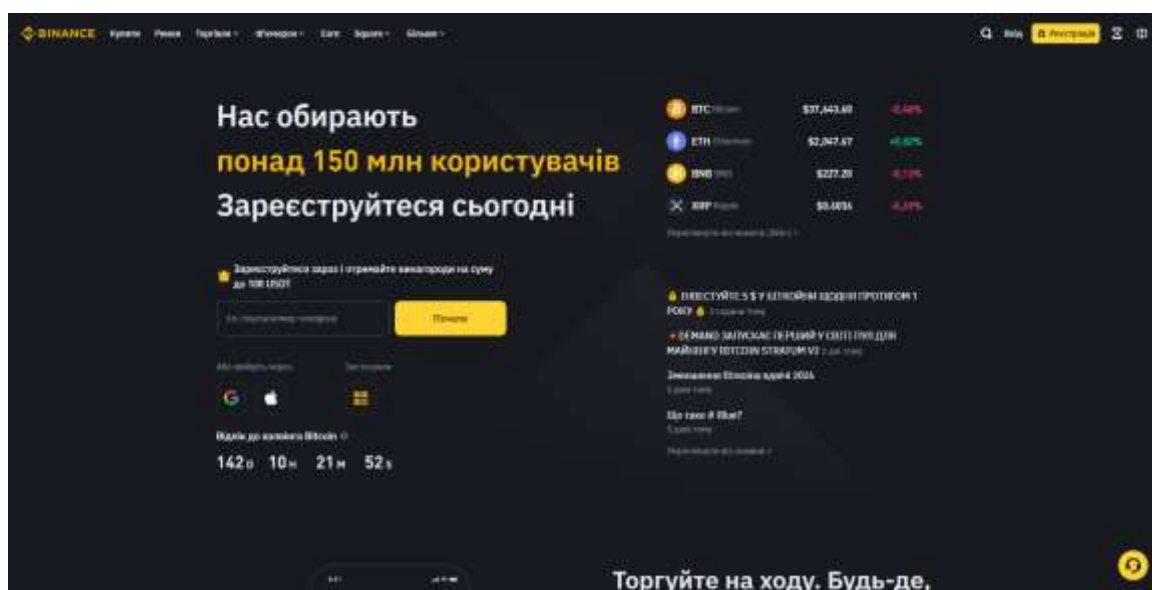


Рисунок 1.1 – Головна сторінка сайту «binance.com»

Coinbase – це одна з найвідоміших криптовалютних бірж [12]. Заснована в 2012 році в Сан-Франциско, ця компанія стала популярною серед користувачів, які бажають купувати, продавати та обмінювати криптовалюти, такі як Bitcoin, Ethereum, Litecoin та інші.

Coinbase пропонує простий інтерфейс для обміну криптовалют, що робить його популярним серед новачків у світі криптовалют (див. рисунок 1.2). Крім того, платформа надає додаткові послуги, такі як зберігання криптовалют у власному гаманці (wallet), створення торгових акаунтів для професійних трейдерів, а також освітні матеріали щодо криптовалют та їх використання[10].

Компанія активно розвивається і розширює свої послуги, включаючи впровадження нових криптовалют, функцій безпеки та партнерств з іншими фінансовими установами для полегшення обміну та торгівлі криптовалютами.

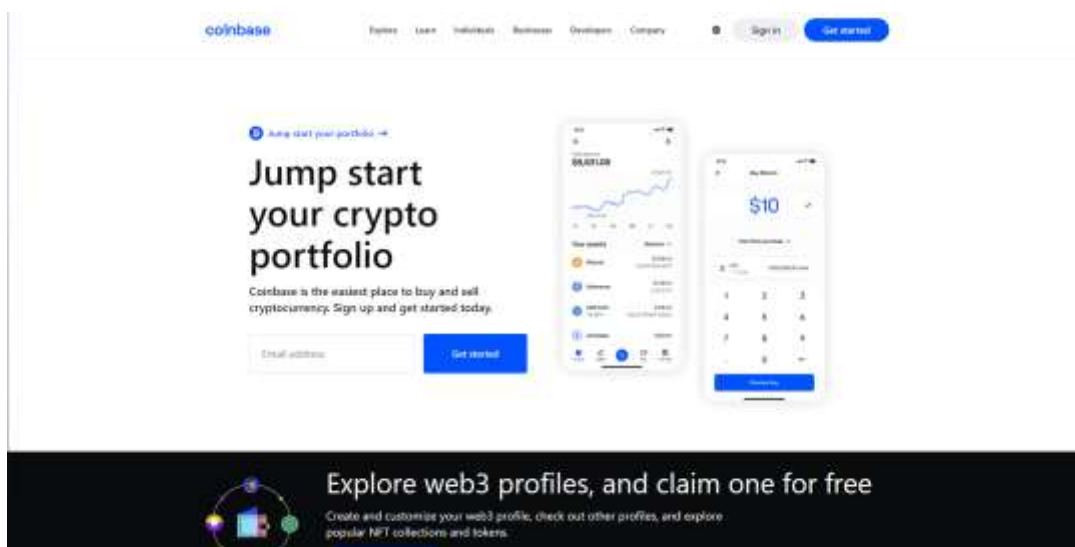


Рисунок 1.2 – Головна сторінка сайту «coinbase.com»

Нюбі – це провідна світова криптовалютна біржа, яка була запущена в 2013 році у Сінгапурі [13]. Вона пропонує платформу для обміну криптовалют, таких як Bitcoin, Ethereum, Litecoin, Ripple і багато інших.

Основною метою Нюбі є забезпечення швидкого та безпечного обміну криптовалют для своїх користувачів. Платформа також надає можливість торгувати ф'ючерсами на криптовалюти, а також відповідні інструменти для професійних трейдерів.

Крім цього, Нюбі активно розширює свої послуги, включаючи створення власних токенів (Нюбі Token), програму стимулювання для трейдерів, а також надання різноманітних фінансових послуг, таких як депозити, кредитування та інші інвестиційні продукти пов'язані з криптовалютами.

Нюбі активно працює над розвитком та поширенням своєї глобальної присутності, співпрацюючи з різними регуляторами та розширюючи набір послуг для своїх користувачів по всьому світу. Побачити її можна на рисунку 1.3.

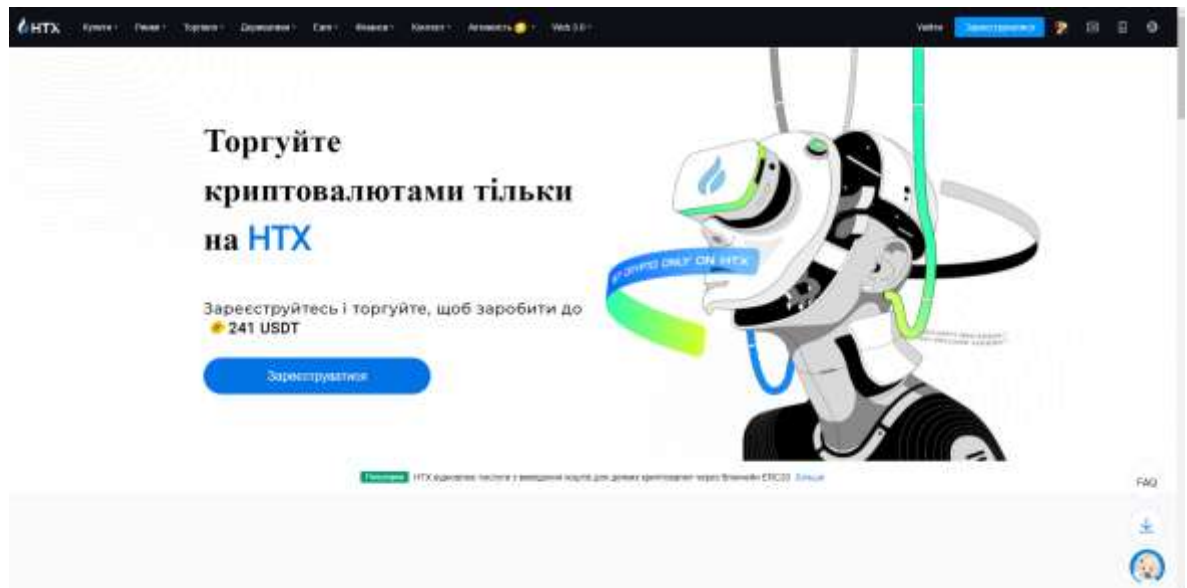


Рисунок 1.3 – Головна сторінка сайту «htx.com»

KuCoin – це міжнародна криптовалютна біржа, запущена в 2017 році зі штаб-квартирою в Гонконгу[14]. Вона стала відомою завдяки широкому спектру криптовалют, які доступні для торгівлі, та інноваційним фінансовим послугам (див. рисунок 1.4).

Платформа KuCoin пропонує можливість обмінювати більш ніж сотню різних криптовалют, включаючи популярні активи, такі як Bitcoin, Ethereum, Litecoin та інші альткойни. Крім торгівлі, KuCoin надає інші послуги, такі як програма стимулювання для трейдерів, створення власного токenu (KuCoin Shares), який надає додаткові переваги користувачам, і функції зберігання коштів у власному гаманці (wallet).

Однією з особливостей KuCoin є активна підтримка нових проектів та криптовалют, які набирають популярність на ринку. Біржа також намагається забезпечити високий рівень безпеки для своїх користувачів, використовуючи сучасні технології для захисту активів.

KuCoin постійно розвивається, розширює свою глобальну присутність та набір послуг, намагаючись стати однією з передових бірж у світі криптовалют.



Рисунок 1.4 – Головна сторінка сайту «kucoin.com»

1.2.2 Порівняння програмних систем

Результати порівняння програмних систем Binance, Coinbase, Huobi та KuCoin наведені в табл. 1.1.

Таблиця 1.1 – Порівняння систем

№	Характеристика	Coinbase	Huobi	KuCoin	Binance
Функціонал					
1	Купівля/продаж криптовалют	Так	Так	Так	Так
2	Ф'ючерси	Ні	Так	Ні	Так
3	Опціони	Ні	Ні	Ні	Так
4	Стейкінг	Ні	Так	Так	Так
5	Позики	Ні	Так	Ні	Так
6	Власний токен	Немає	Немає	Немає	Binance Coin (BNB)
7	Програма стимулювання	Ні	Так	Так	Ні
8	Криптовалютна картка	Ні	Ні	Ні	Так
9	Блокчейн-ініціативи	Не активно розвиває	Неактивно розвиває	Неактивно розвиває	Binance Smart Chain (BSC)
Зручність інтерфейсів					
1	Веб-сайт	Так	Так	Так	Так
2	Мобільний додаток	Так	Так	Так	Так
3	Інтерфейс користувача	Простий, зрозумілий	Зрозумілий, багатофункціональний	Зрозумілий, інтуїтивний	Зрозумілий, інтуїтивний
4	Доступність функцій	Базові функції, обмін, зберігання, графіки	Обмін, стейкінг, позики,	Обмін, стейкінг, програма	Обмін, ф'ючерси, опціони,

			торгівля ф'ючерсами	стимулювання	стейкінг, позики
5	Швидкість роботи	Швидкий	Швидкий	Швидкий	Швидкий
6	Можливість аналізу даних	Зручний аналіз графіків, статистика	Аналіз графіків, історія торгівлі, звіти	Аналіз графіків, статистика торгівлі	Аналіз графіків, статистика торгівлі
7	Функціонал для новачків	Доступний, легкий для використання	Доступний, можливості для початківців	Доступний, легкий для використання	Доступний, можливості для початківців
	Безпека	Висока безпека, двофакторна автентикація	Висока безпека, захист від вторгнень	Висока безпека, захист від вторгнень	Висока безпека, двофакторна автентикація
Зворотній зв'язок					
1	Підтримка клієнтів	Тільки пошта, терміни відповіді можуть варіюватись	Пошта, онлайн-чат, телефон, тікети	Пошта, онлайн-чат, тікети, спільнота	Пошта, онлайн-чат, тікети, спільнота
2	Час очікування на відповідь	Різноманітно, від кількох годин до кількох днів	Зазвичай від 24 годин до 72 годин	Різноманітно, від кількох годин до кількох днів	Різноманітно, від кількох годин до кількох днів
3	База знань	Так	Так	Так	Так
4	Соціальні медіа	Так	Так	Так	Так
5	Форми зворотного зв'язку	Електронна пошта, соціальні мережі, форуми	Електронна пошта, онлайн-чат, телефон	Електронна пошта, онлайн-чат, форуми	Електронна пошта, онлайн-чат, форуми
6	Мовна підтримка	Зазвичай англійська, іноді інші мови	Багатомовна, включаючи українську	Багатомовна, включаючи українську	Багатомовна, включаючи українську
7	Репутація компанії в плані підтримки	Різна, деякі відгуки про терміни відповіді	Добра, високо оцінена підтримка користувачів	Середня, різні відгуки про час очікування	Добра, високо оцінена підтримка користувачів

Binance видається найбільш розширеною з можливостями та інструментами для трейдерів, пропонуючи ф'ючерси, опціони, стейкінг, позики, різноманітні криптовалютні пари та Binance Smart Chain (BSC).

1.2.3 Позитивні характеристики існуючих систем, що їх слід відтворити у проектних рішеннях

Huobi вирізняється розширеним спектром каналів зв'язку, таких як пошта, онлайн-чат, телефон та тікети, що може полегшити взаємодію з підтримкою.

Binance веде у розвитку блокчейн-технологій, запускаючи Binance Smart Chain (BSC), що дає можливість розробникам створювати децентралізовані додатки та розвивати фінансові продукти.

Варто враховувати, що кожна з цих бірж періодично зіштовхується з регуляторами у різних країнах через питання ліцензування та дотримання правил фінансового регулювання.

Coinbase славиться своїм простим інтерфейсом, що робить його добрим вибором для новачків у світі криптовалют. Binance також має інтуїтивний інтерфейс для трейдерів.

Нові системи повинні мати розширений набір інструментів для трейдерів, подібно до Binance, що має ф'ючерси, опціони, стейкінг та інші можливості. Простий та зрозумілий інтерфейс, як у Coinbase чи Binance, сприяє простоті використання для користувачів будьякого рівня досвіду.

Наявність різних каналів зворотного зв'язку, як у Huobi, забезпечує оперативну допомогу та задоволення потреб клієнтів. Регулярне оновлення та вдосконалення платформи є ключовим для залишання системи актуальною й конкурентоздатною. Важливо враховувати сучасні методи захисту даних та фінансових активів користувачів, подібно до заходів безпеки, які використовує Binance.

1.2.4 Недоліки існуючих систем та напрями їх удосконалення

Один з найбільших недоліків полягає у складнощах, з якими користувачі можуть стикатися при взаємодії з підтримкою клієнтів. Час очікування на відповідь чи обробку питань може бути значною проблемою. Удосконалення цього аспекту передбачає впровадження ефективних систем тікетів або

автоматизованих процесів, що сприятимуть оперативнішій та якісній підтримці користувачів.

Також, хоча безпека є пріоритетом для цих платформ, проте іноді вони стикаються з проблемами безпеки, що призводить до витоку даних або кібератак. Подальше удосконалення систем безпеки через впровадження передових методів шифрування, двофакторної аутентифікації та посиленням заходів захисту може підвищити рівень впевненості користувачів у захисті їхніх активів.

Деякі з цих платформ також мають проблеми з взаємодією з регуляторами у різних країнах через питання ліцензування та дотримання правил фінансового регулювання. Удосконалення в цьому напрямі може включати посилення співпраці з регуляторами та удосконалення політики дотримання правил.

1.3 Розробка концепції інформаційної системи

На основі проведеного аналізу предметної області, система повинна забезпечувати інтеграцію з актуальними ринковими даними, включаючи ціни на криптовалюти, обсяги торгів, історичні дані тощо. Це дозволить користувачам приймати обґрунтовані рішення, базуючись на найсвіжішій інформації. Враховуючи високі ризики безпеки у сфері криптовалют, система повинна включати розширені механізми безпеки, такі як двофакторна аутентифікація, шифрування даних, захист від несанкціонованого доступу та шахрайства.

Інтерфейс має бути інтуїтивно зрозумілим та зручним для користувачів різного рівня досвіду. Це включає наявність ясної навігації, зрозумілого представлення даних, інформативних панелей інструментів та ефективних засобів візуалізації.

Система повинна надавати інструменти для управління інвестиційним портфелем, включаючи відстеження прибутку/збитку, історію транзакцій та аналітику портфеля. До того ж, вона має бути гнучкою, щоб адаптуватися до

швидкозмінних умов ринку та зростаючих потреб користувачів. Також важливим елементом є масштабованість, для ефективного оброблення та збільшення обсягу транзакцій за великої кількості користувачів.

Потрібно передбачити можливість інтеграції з іншими фінансовими системами, платіжними шлюзами та аналітичними інструментами для забезпечення широкого спектру функціональності.

В проекті застосовується архітектурний шаблон MVC (Model-View-Controller), який є визнаним стандартом у сфері розробки програмного забезпечення. MVC дозволяє структурувати програму на три ключові компоненти: модель, представлення та контролер, що сприяє ефективному розділенню відповідальностей.

1. Модель (Model) відповідає за представлення бізнес-даних та бізнес-логіку програми. Це може бути окремий клас або набір класів, які управляють даними, забезпечують доступ до бази даних, реалізують операції з даними, здійснюють валідацію та обробляють бізнес-правила.

2. Представлення (View). Компонент представлення відповідає за візуалізацію даних для користувача. Це може бути виглядом HTML-сторінки, шаблоном веб-інтерфейсу чи графічним інтерфейсом користувача. Представлення отримує дані з моделі та візуалізує їх для користувача.

3. Контролер (Controller). Контролер служить як медіатор між моделлю та представленням, обробляючи запити від користувача, взаємодіючи з моделлю для виконання необхідних операцій, та вибираючи відповідне представлення для відображення результатів. Контролер також передає вхідні дані від користувача до моделі для обробки та вибору відповідного візуального представлення.

Основною перевагою використання MVC є чітке розділення відповідальностей між компонентами, що робить систему більш організованою, підтримуваною та гнучкою для розвитку. Такий підхід сприяє легкій інтеграції змін у бізнес-логіці без впливу на візуальну частину та навпаки, забезпечуючи більш ефективне управління проектом.

Для реалізації front-end частини використовувалися HTML, CSS та JavaScript. Поєднання цих трьох технологій дозволяє створювати повноцінні, привабливі та функціональні веб-додатки та веб-сайти. HTML відповідає за структуру та контент, CSS – за візуальне представлення, а JavaScript забезпечує інтерактивність та динамічну поведінку.

Для демонстрації користувацьких інтерфейсів було створено прототип інформаційної системи у онлайн-сервісі Figma[17]. На рисунку 1.5 представлено повну картину створеного клікабельного прототипу.

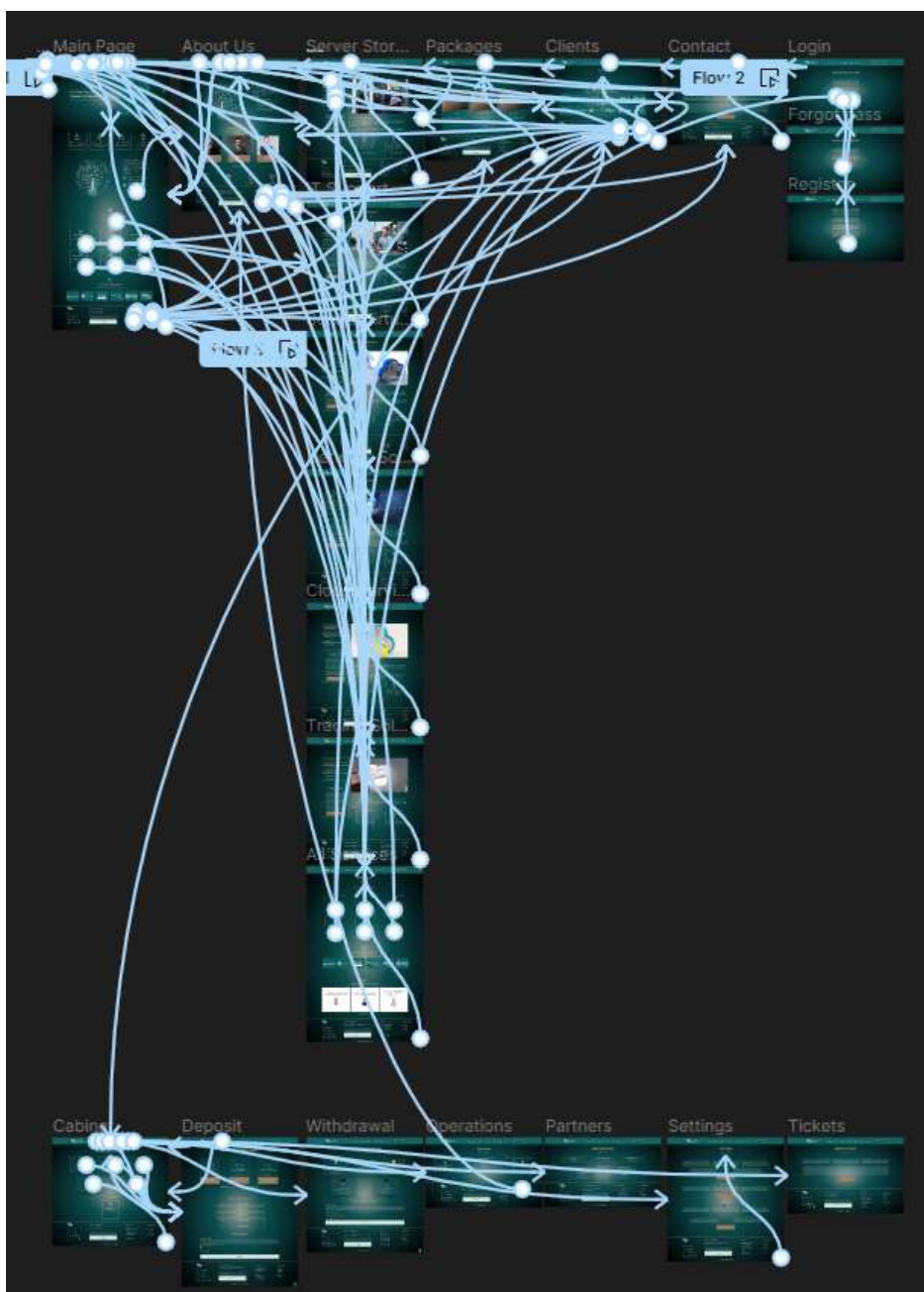


Рисунок 1.5 – Клікабельний прототип інформаційної системи
«Криптоінвестиції» розроблений у Figma

Для реалізації клікабельності необхідно додати налаштування для потрібних елементів (див. рисунки 1.6-1.10).

Для елемента меню «Services» необхідно створити додаткове під-меню:

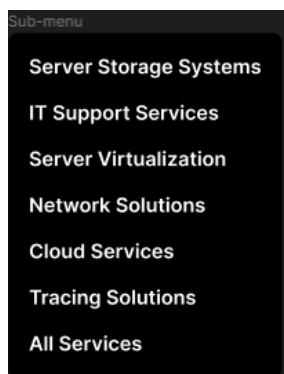


Рисунок 1.6 – Створене вікно

Після цього треба додати налаштування для елемента меню «Services» та самого вікна

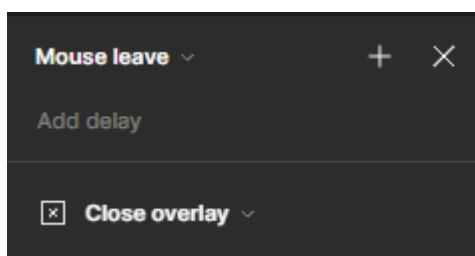
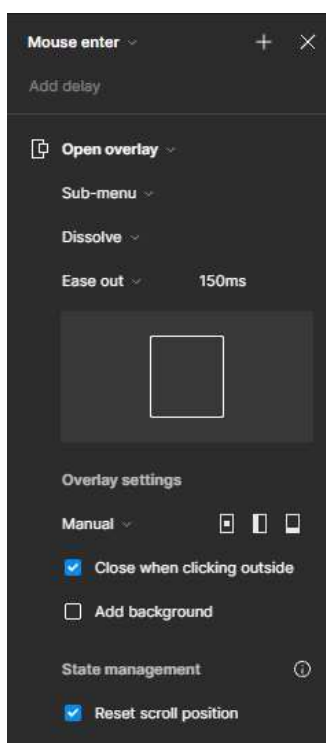


Рисунок 1.7-1.8 – Налаштування прототипів

Таким самим чином створюємо інтерактивність для інших кнопок

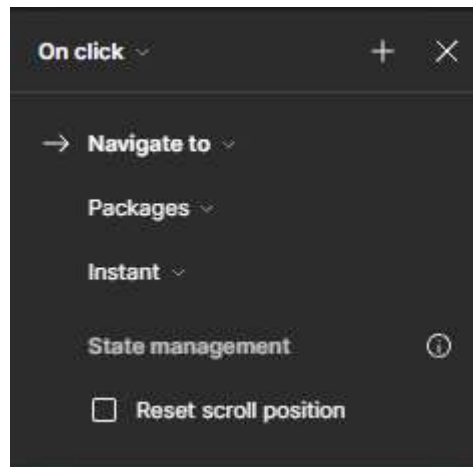


Рисунок 1.9 – Прототип переміщення на іншу сторінку

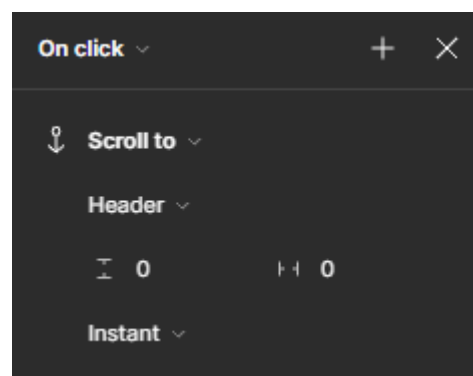


Рисунок 1.10 – Прототип для кнопки підняття вгору

Сторінки прототипу системи наведено у «Додатку В».

Перший крок полягав у розробці привабливої та інформативної головної сторінки, що відображає основні аспекти та послуги сайту. Результатом став дизайн головної сторінки з легким доступом до ключових розділів та передачею основної ідеї проекту.

Далі створювалася сторінка "About Us" для ознайомлення з компанією, її місією, цінностями та командою. Дизайн цієї сторінки ефективно демонструє історію та цілі компанії.

Наступний розділ зосереджений на наданні інформації про безпеку сервісу та захист даних користувачів. Для цього були створені окремі сторінки для кожного сервісу з описами та особливостями рівнів безпеки та додаткової підтримки.

Також було створено інтерактивну сторінку вибору пакету підтримки з варіативними опціями та чітким описом кожного пакету, для підвищення

візуальної привабливості сайту використані інтегровані анімаційні ефекти на різних сторінках та елементах меню.

Крім того, розроблено функціональний особистий кабінет з можливостями управління коштами, перегляду історії операцій, участі в реферальній програмі та налаштування профілю. Створено додаткові сторінки для зворотного зв'язку та повідомлення про помилки або побажання.

Також, був проведений аналіз функціонала системи та узгодження з вимогами (див. табл. 1.2).

Таблиця 1.2 – Аналіз функціонала та узгодження з вимогами

Priority	User Story	Результат
1	Клієнт хоче бачити привабливу та інформативну головну сторінку, яка відображає основні аспекти та послуги сайту.	Дизайн головної сторінки, який забезпечує легкий доступ до ключових розділів сайту і передає основну ідею проекту.
2	Клієнт хоче дізнатися детальніше про компанію, її місію, цінності та команду.	Дизайн сторінки "About Us", який ефективно демонструє історію та цілі компанії.
3	Для клієнта важливо дізнатися про безпеку сервісу та його даних	Створено сторінки для кожного сервісу, включаючи описи та особливості рівнів безпеки та додаткової підтримки.
4	Дозволити користувачам легко вибирати та налаштовувати пакети підтримки.	Інтерактивна сторінка вибору пакету підтримки з варіативними опціями та чітким описом кожного пакету.
5	Підвищити візуальну привабливість сайту за допомогою анімації.	Інтеграція анімаційних ефектів на різних сторінках та елементах меню для поліпшення користувацького досвіду.
6	Надати користувачам зручний спосіб управління їхніми інвестиціями та особистими даними.	Функціональний особистий кабінет з можливостями управління коштами, перегляду історії операцій, участі в реферальній програмі та налаштування профілю.
7	Клієнт бажає залишити відгук, побажання чи повідомити про помилки	Створено додаткові сторінки для зворотного зв'язку та бажаним способом (гаряча лінія телефону чи електронна пошта).

Під час розробки проекту для реалізації збереження даних, було використано реляційну БД MySQL. Для побудови схеми БД було використано СКБД MySQL Workbench[19].

Клієнт із сервером зв'язуються за допомогою HTTP-запитів[29]. Серед них є:

- запит на реєстрацію нового користувача;
- запит на авторизацію (отримання ключа авторизації);
- запит на отримання персональних даних, таких як баланс або додані гаманці;
- запит на депозит;
- запит на виведення коштів;
- запит на перегляд усіх операцій;
- запит на налаштування або зміну персональних даних;
- запит на виведення реферальних коштів;
- запит на зворотний зв'язок;

Фрагменти програмної реалізації цього прототипу можна побачити у «Додатку Б».

РОЗДІЛ 2

РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Аналіз і специфікація вимог до інформаційної системи

Процес аналізу та специфікації вимог забезпечує чітке розуміння потреб користувачів, бізнес-цілей та функціональних можливостей, необхідних для досягнення цих цілей.

Збір вимог проводиться шляхом інтерв'ю та опитувань ключових зацікавлених сторін, аналізу конкурентного середовища та існуючих рішень на ринку криптоінвестицій, а також визначення основних бізнес-процесів та сценаріїв використання системи.

Аналіз вимог включає класифікацію зібраних вимог на функціональні та нефункціональні, перевірку їх повноти, узгодженості та реалістичності, виявлення та вирішення конфліктів між вимогами, а також пріоритезацію відповідно до важливості та впливу на бізнес-цілі.

Специфікація вимог передбачає детальний опис функціональних вимог до системи, визначення нефункціональних вимог щодо продуктивності, безпеки, масштабованості, зручності використання та відповідності нормативним вимогам, створення діаграм та інших артефактів для візуалізації вимог, а також документування їх у зрозумілій та структурованій формі. Узгодження та затвердження вимог із зацікавленими сторонами забезпечує відповідність їхнім очікуванням та потребам, а також офіційне затвердження вимог ключовими зацікавленими сторонами. При необхідності можливе управління змінами та контроль над ними протягом життєвого циклу розробки системи, аналіз впливу змін на існуючі вимоги, архітектуру системи та план розробки, а також забезпечення належної комунікації та узгодження змін із зацікавленими сторонами.

2.1.1 Бізнес-вимоги до системи

Чітко визначені бізнес-цілі дозволяють зрозуміти основну мету створення системи та забезпечити її відповідність потребам користувачів і ринку. Наприклад, метою може бути надання зручної та безпечної платформи для інвестицій у криптовалюту, залучення нових інвесторів або підвищення ефективності торгових операцій.

На основі бізнес-цілей формуються вимоги до системи, які визначають її функціональність, продуктивність, безпеку, зручність використання та інші важливі характеристики. Вимоги забезпечують, що система відповідає очікуванням користувачів, дотримується нормативних вимог та забезпечує конкурентні переваги на ринку.

Детально прописані вимоги до функціоналу, інтерфейсу, процесів, зберігання даних та безпеки допомагають розробникам створити якісний продукт, який задовольнить потреби інвесторів та адміністраторів. Чіткі вимоги до швидкодії, масштабованості та надійності гарантують належну роботу системи навіть за великих навантажень.

Крім того, ретельно продумані вимоги до аналітичних інструментів, візуалізації даних та інтеграції з зовнішніми джерелами інформації забезпечать точність та актуальність інвестиційної аналітики.

Далі наведено діаграму бізнес-вимог та бізнес-цілей кожної особи яка взаємодіє з системою (див. рисунок 2.1). Діаграму було розроблено в онлайн-застосунку Draw.IO[7]. Основними особами, що взаємодіють з системою є інвестори, розробники та технічний персонал, фінансові аналітики та партнери.



Рисунок 2.1 – Діаграма визначення бізне-цілей та бізнес-вимог

2.1.2 Функціональні вимоги до системи

Функціональні вимоги визначають основні можливості та функції, які система повинна надавати для задоволення потреб користувачів та досягнення бізнес-цілей проекту.

Чітко сформульовані функціональні вимоги забезпечують, що система матиме необхідний функціонал для управління обліковими записами користувачів, створення та керування крипто-гаманцями, здійснення торгових операцій з купівлі/продажу криптовалют, доступу до аналітичних інструментів та візуалізації даних. Вони також гарантують наявність важливих функцій безпеки, таких як двофакторна автентифікація, шифрування даних та захист від шахрайства.

Крім того, функціональні вимоги забезпечують, що система матиме необхідні адміністративні функції для моніторингу активності користувачів, налаштування параметрів системи та генерації звітів. Це допомагає підтримувати ефективну роботу системи та приймати обґрунтовані рішення щодо її розвитку та вдосконалення.

Ретельно проаналізовані та задокументовані функціональні вимоги є основою для розробки якісного програмного забезпечення, яке буде відповідати потребам цільової аудиторії та забезпечувати конкурентні переваги на ринку. Вони допомагають уникнути непорозумінь та помилок на етапах проектування, розробки та тестування системи.

До основних функціональних вимог можна віднести:

1. Реєстрація та аутентифікація користувачів.

Користувач повинен мати можливість зареєструватися в системі за допомогою електронної пошти. В рамках реєстрації користувач вказує ПІБ, email, номер телефону та пароль[30]. Система здійснює перевірку вказаних при реєстрації даних, підтвердження email, захист від ботів. Користувач, зареєстрований в системі, може аутентифікуватися за допомогою email та пароля. Система має передбачати функцію відновлення пароля на випадок його втрати або несанкціонованого доступу в аккаунт

2. Робота з рахунками та транзакціями.

Після входу користувачу має надаватись інформація щодо балансів на його рахунках, історії транзакцій, портфелю активів. Користувач повинен мати можливість інвестувати свої кошти за вибраним планом та мати можливість виведення криптовалюти на зовнішні гаманці

3. Аналітика та інвестиції портфелю.

Користувач може відслідковувати зміну балансу, прибутковість портфеля і окремих активів у часі. Система формує звіти про стан портфеля, історію транзакцій користувача в розрізі вибраного проміжку часу. Також користувач може обрати новий план інвестицій, або достроково завершити попередній.

Детальніше переглянути функціональні вимоги можна на діаграмі, що зображена на рисунку 2.2 та на рисунках 2.3-2.4 переглянути їх специфікацію[3].

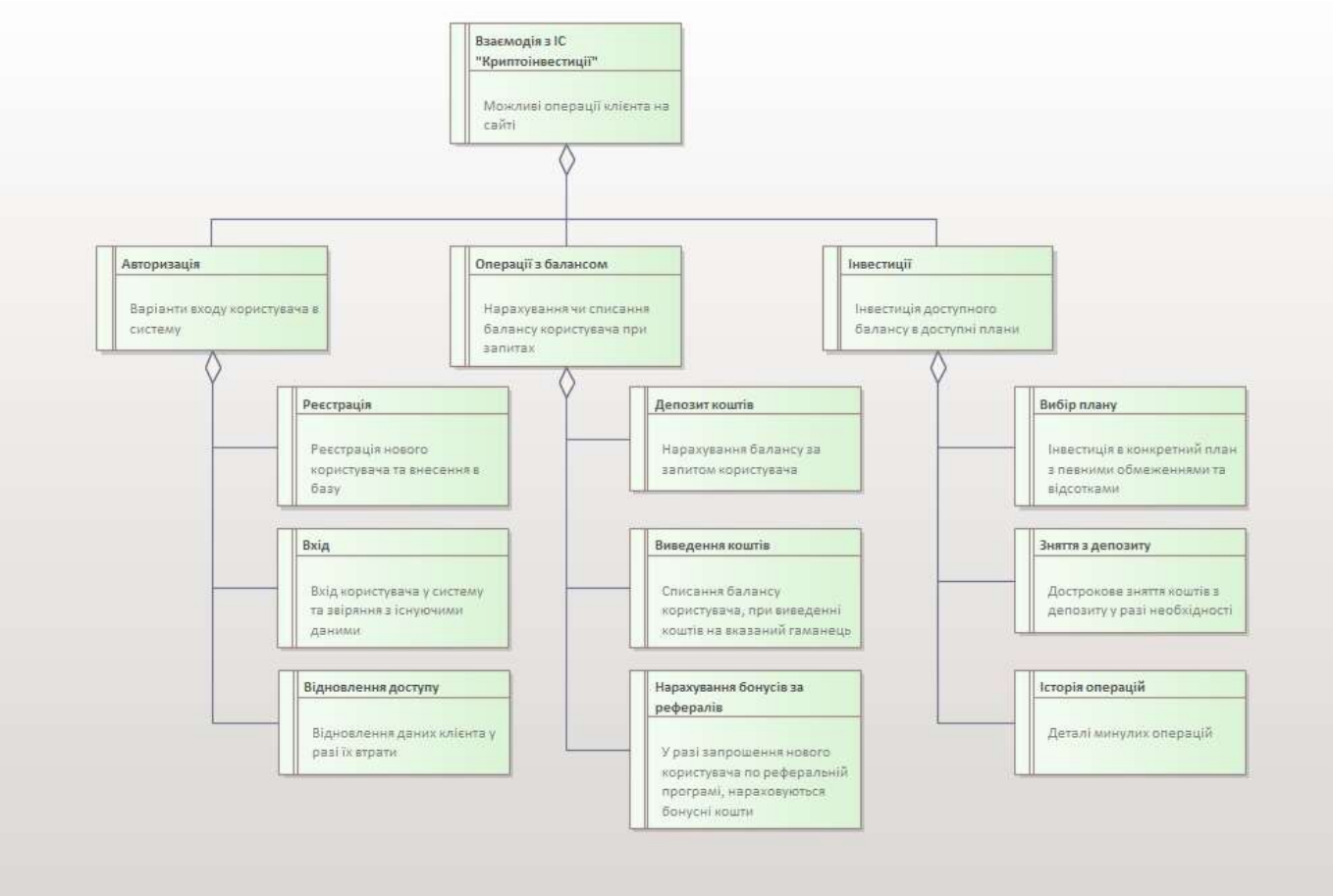


Рисунок 2.2 – Діаграма функціональних вимог

Item	Stereotype	Status	Difficulty	Priority
☒ Взаємодія з ІС "Криптоінвестиції" Можливі операції клієнта на сайті	Functional	implemented	Low	High
☒ Авторизація Варіанти входу користувача в систему	Functional	implemented	High	High
☒ Реєстрація Реєстрація нового користувача та внесення в базу	Functional	implemented	Medium	Medium
☒ Відновлення доступу Відновлення даних клієнта у разі їх втрати	Functional	implemented	High	Medium
☒ Вхід Вхід користувача у систему та звіряння з існуючими даними	Functional	implemented	Medium	Medium
☒ Операції з балансом Нарахування чи списання балансу користувача при запитах	Functional	implemented	Medium	High
☒ Виведення коштів Списання балансу користувача, при виведенні коштів на вказаний гаманець	Functional	implemented	Medium	High
☒ Депозит коштів Нарахування балансу за запитом користувача	Functional	implemented	Medium	High
☒ Нарахування бонусів за рефералів У разі запрошення нового користувача по реферальній програмі, нараховуються бонусні кошти	Functional	Mandatory	High	Low

Рисунок 2.3 – Специфікація вимог «Авторизація» та «Операції з балансом»

☒ Інвестиції Інвестиція доступного балансу в доступні плани	Functional	Implemented	Medium	High
☒ Вибір плану Інвестиція в конкретний план з певними обмеженнями та відсотками	Functional	Implemented	High	Medium
☒ Зняття з депозиту Дострокове зняття коштів з депозиту у разі необхідності	Functional	Implemented	Medium	Medium
☒ Історія операцій Деталі минулих операцій	Functional	Implemented	Medium	Low

Рисунок 2.4 – Специфікація вимог «Інвестиції»

2.1.3 Нефункціональні вимоги до системи

Нефункціональні вимоги визначають критерії функціонування системи, які не стосуються безпосередньо її основних функцій, але є критично важливими для задоволення потреб користувачів та досягнення бізнес-цілей проекту.

Одним з ключових аспектів нефункціональних вимог є забезпечення належної продуктивності та масштабованості системи. Ці вимоги гарантують, що система зможе ефективно обробляти велику кількість запитів та операцій користувачів, особливо в періоди пікових навантажень, забезпечуючи швидкий відгук та стабільну роботу.

Нефункціональні вимоги можуть охоплювати питання безпеки та конфіденційності даних, що є надзвичайно важливим для фінансової системи, пов'язаної з криптовалютами. Вони визначають стандарти захисту від несанкціонованого доступу, шифрування даних, контролю доступу та інших заходів безпеки, які гарантують захист конфіденційної інформації користувачів та забезпечують довіру до системи.

Зручність використання та дружній інтерфейс забезпечує, що система буде інтуїтивно зрозумілою та легкою у використанні для різних категорій користувачів, незалежно від їхнього рівня технічних знань та досвіду роботи з криптовалютами.

Крім того, нефункціональні вимоги можуть включати критерії сумісності та інтеграції з іншими системами та сервісами, вимоги до

підтримки та оновлення системи, а також вимоги до надійності та відмовостійкості для забезпечення безперебійної роботи.

Загальні нефункціональні вимоги можна оглянути на рисунку 2.5, їх специфікації на рисунках 2.6-2.9 відповідно.

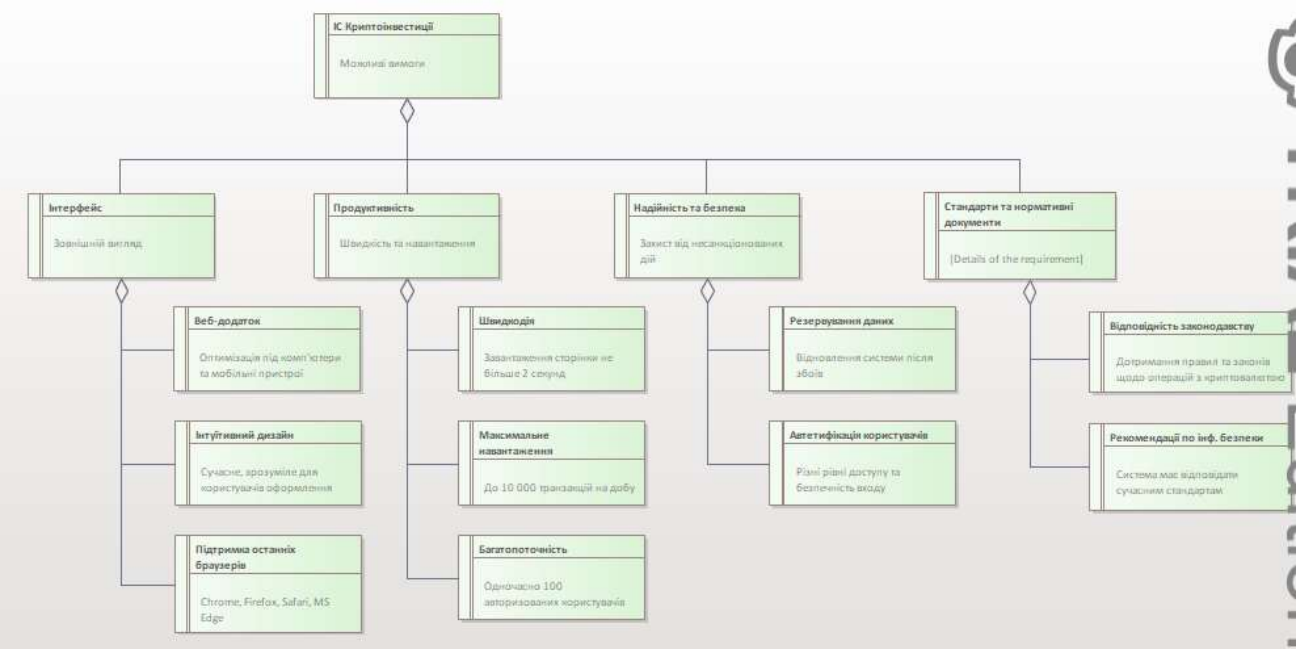


Рисунок 2.5 – Діаграма функціональних вимог.

☒ ІС Криптоінвестицій Можливі вимоги	Functional	Implemented	Medium	Medium
☒ Інтерфейс Зовнішній вигляд	NonFunctional	Approved	High	High
☒ Веб-додаток Оптимізація під комп'ютери та мобільні пристрої	NonFunctional	Implemented	High	Medium
☒ Інтуїтивний дизайн Сучасне, зрозуміле для користувачів оформлення	NonFunctional	Implemented	Low	Medium
☒ Підтримка останніх браузерів Chrome, Firefox, Safari, MS Edge	NonFunctional	Implemented	Low	Medium

Рисунок 2.6 – Специфікація нефункціональних вимог «Інтерфейс»

☒ Надійність та безпека Захист від несанкціонованих дій	NonFunctional	Approved	High	High
☒ Автентифікація користувачів Різні рівні доступу та безпечність входу	NonFunctional	Implemented	Medium	High
☒ Резервування даних Відновлення системи після збоїв	NonFunctional	Implemented	Medium	High

Рисунок 2.7 – Специфікація нефункціональних вимог «Надійність та безпека»

☒ Продуктивність Швидкість та навантаження	NonFunctional	Validated	High	Medium
☒ Багатопоточність Одночасно 100 авторизованих користувачів	NonFunctional	Validated	Medium	Medium
☒ Максимальне навантаження До 10 000 транзакцій на добу	NonFunctional	Validated	Medium	Low
☒ Швидкодія Завантаження сторінки не більше 2 секунд	NonFunctional	Validated	Medium	Medium

Рисунок 2.8 – Специфікація нефункціональних вимог «Продуктивність»

☒ Стандарти та нормативні документи Відповідність до міжнародних законів та норм	NonFunctional	Proposed	Medium	High
☒ Відповідність законодавству Дотримання правил та законів щодо операцій з криптовалютою	NonFunctional	Implemented	Medium	High
☒ Рекомендації по інф. безпеки Система має відповідати сучасним стандартам	NonFunctional	Implemented	Medium	High

Рисунок 2.9 – Специфікація нефункціональних вимог «Стандарти та нормативні документи»

2.2 Обґрунтування методології проектування та функціональна модель задачі

Вибір об'єктно-орієнтованого програмування, як методології для проектування інформаційної системи для криптоінвестицій, має такі причини:

– ООП сприяє створенню модульних та взаємозамінних компонентів системи. Це означає, що різні елементи системи, такі як обробка транзакцій, управління користувацькими акаунтами, аналітика портфеля, можуть бути розроблені як окремі об'єкти, що спрощує процес розробки та тестування. Також це дозволяє легко перевикористовувати код в інших частинах проекту або навіть у майбутніх проектах.

– Криптоінвестиційні системи часто мають високий рівень складності з багатьма взаємодіючими компонентами. ООП дозволяє легше

управляти цією складністю за рахунок інкапсуляції та абстракції, приховуючи складні деталі від кінцевого користувача.

- ООП забезпечує систему гнучкістю, необхідною для адаптації до змінних вимог та умов. Об'єкти можуть бути легко змінені, розширені або замінені, не порушуючи іншу частину системи, що особливо важливо для динамічного ринку криптовалют.

- ООП дозволяє розподілити роботу серед команди розробників більш ефективно. Кожен член команди може працювати над окремими класами або модулями, зменшуючи залежності та конфлікти в коді.

- Цей метод підтримує принципи, такі як наслідування та поліморфізм, які дозволяють створювати більш гнучкий та надійний код. Це зменшує ймовірність помилок та сприяє створенню якісного програмного продукту.

- Більшість сучасних програмних мов та фреймворків підтримують ООП, що забезпечує високу сумісність з іншими технологіями, широкий вибір інструментів та легку інтеграцію з зовнішніми бібліотеками та сервісами.

Враховуючи ці переваги, ООП є ідеальним вибором для розробки складної, гнучкої та масштабованої інформаційної системи, яка зможе ефективно адаптуватися до постійно змінюваних вимог та умов ринку.

2.3 Моделювання предметної галузі

Основна мета цієї інформаційної системи полягає в ефективному управлінні криптоінвестиціями, що включає в себе збір, аналіз, інтерпретацію та зберігання даних, пов'язаних з криптовалютними активами. Задача полягає у виявленні трендів ринку, визначенні потенційно прибуткових інвестиційних можливостей та управлінні ризиками. Це включає аналіз ринкових даних, таких як ціни, обсяги торгів, історія вартості криптовалют, а також вивчення новин і тенденцій, які можуть вплинути на ринок.

Ключовими процесами в системі є реєстрація користувачів, створення та управління крипто-гаманцями, здійснення операцій з купівлі/продажу криптовалют, відстеження курсів та аналіз ринку. Система повинна надавати зручний інтерфейс для інвесторів, надійні механізми зберігання та передачі криптовалют, а також актуальну аналітику для прийняття інвестиційних рішень.

Загалом, інформаційна система "Криптоінвестиції" має на меті створити комплексне рішення для роботи з криптовалютами, поєднуючи функції гаманця, обмінника та аналітичної платформи в одному місці. Ключовими вимогами є безпека, зручність використання та своєчасна і точна інформація для успішних інвестицій.

Ця інформаційна система повинна надавати інвесторам різні плани та інвестиційні програми, дозволяючи їм приймати обґрунтовані інвестиційні рішення та оптимізувати свої портфелі з метою максимізації прибутку та мінімізації ризиків[5].

Модель предметної області подано ієрархією об'єктно-процесних діаграм мовою моделювання ОРМ (див. рисунки 2.10-2.16).

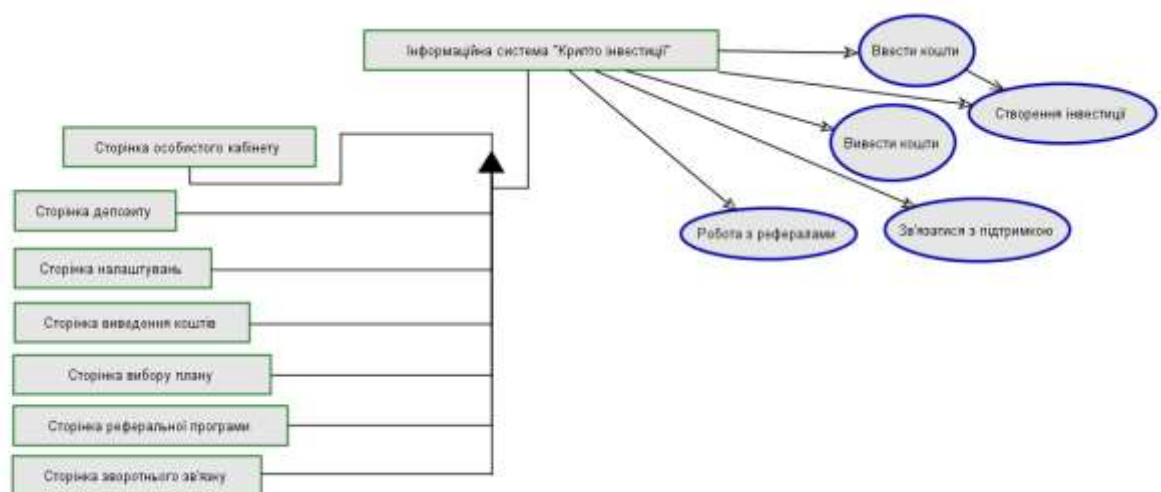


Рисунок 2.10 – Головна діаграма

На діаграмах наступного рівня детально подається функціонал через механізм збільшення. Перший підпроцес – «Робота з рефералами», показує дії, які користувач робить для обробки, запрошення людей та виведення коштів.



Рисунок 2.11 – Діаграма «Робота з рефералами», механізм збільшення

Другий підпроцес – «Створення інвестиції», відображає алгоритм вибору інвестиційного плану в особистому кабінеті, враховуючи внесену суму.

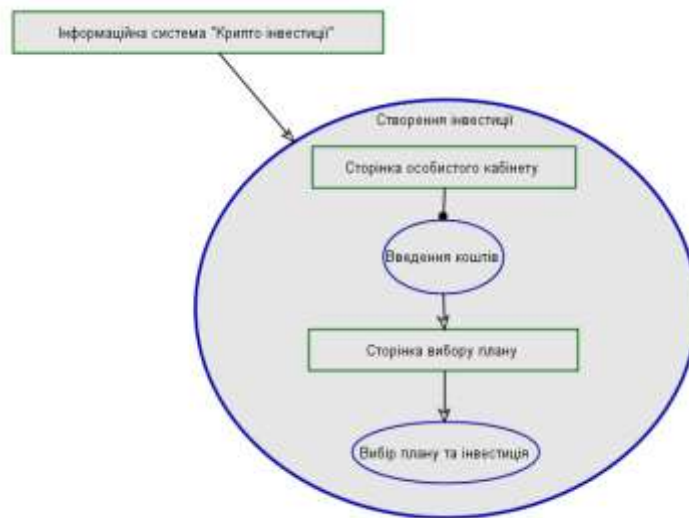


Рисунок 2.12 – Діаграма «Створення інвестиції»

Третій підпроцес – «Зв'язок з підтримкою», на ньому продемонстровано необхідні дії для зворотного зв'язку зі службою підтримки.

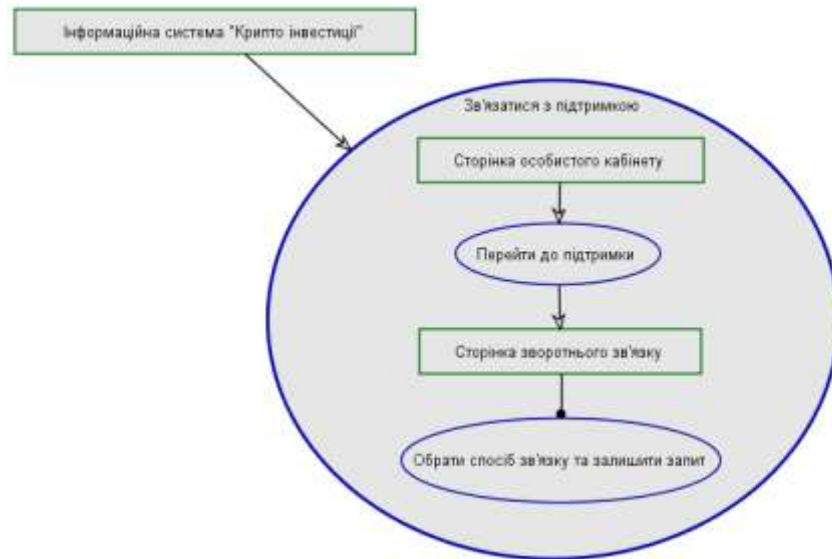


Рисунок 2.13 – Діаграма «Зв'язок з підтримкою»

Четвертий підпроцес – «Вивести кошти», показує алгоритм дій клієнта при необхідності виведення конкретних коштів чи валюти на вказаний гаманець.

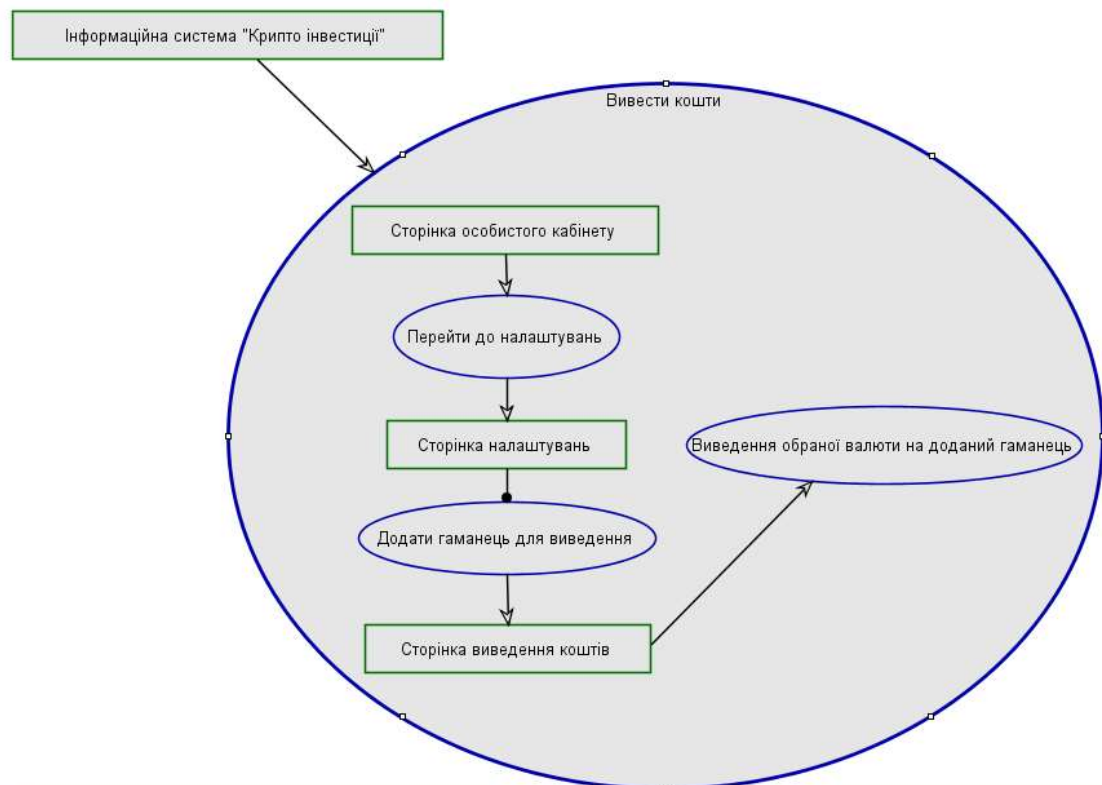


Рисунок 2.14 – Діаграма «Вивести кошти»

На останньому підпроцесі зображено алгоритм депозиту коштів на аккаунт клієнта з конкретної сторінки.



Рисунок 2.15 – Діаграма «Ввести кошти»

Розгортання OPD діаграм відбувається за допомогою ієрархічного підходу. Це означає, що головна діаграма розбивається на піддіаграми, які деталізують окремі процеси або об'єкти.



Рисунок 2.16 – Механізм розгортання

За допомогою мови OPM (Object-Process Methodology) було створено ієрархію діаграм, які візуалізували ключові процеси, об'єкти та їх взаємозв'язки.

2.4 Характеристика задачі

Метою даного проекту є створення інформаційної системи для інвестування в криптовалюту, яка надасть користувачам можливості для придбання, зберігання та продажу різних криптоактивів (Bitcoin, Ethereum, USDT тощо[28]) з метою отримання прибутку від торгівлі та довгострокового інвестування.

Система дозволить користувачам поповнювати особистий рахунок фіатними грошима та криптовалютами, а також виводити кошти назад на банківські рахунки та особисті гаманці.

Обсяг проекту включає створення веб-орієнтованої платформи з наступними основними компонентами:

- Підсистема реєстрації та аутентифікації користувачів, що забезпечить створення облікових записів, авторизацію та аутентифікацію.
- Особистий кабінет інвестора з можливостями перегляду залишків, історії транзакцій, поточного стану інвестицій та здійснення операцій (поповнення, зняття, торгівля криптоактивами).
- Інтеграція з банківськими платіжними системами та сервісами обміну криптовалют для забезпечення можливості вводу/виводу коштів.
- Модуль звітності, що формуватиме аналітичну інформацію щодо операцій та інвестиційної ефективності за запитом користувача.

Система реалізовуватиметься із застосуванням сучасних веб-технологій та хмарних сервісів.

Основною задачею системи є надання повного спектру сервісів для комфортного інвестування в криптовалюту будь-якому зареєстрованому користувачу. Також можна виділити наступні задачі:

- Забезпечення можливості реєстрації користувачів (інвесторів) в системі та подальшої аутентифікації зареєстрованих користувачів для роботи з особистим кабінетом.
- Надання актуальної інформації про курси криптовалют, їх капіталізацію, обсяги торгів на ринках для прийняття обґрунтованих інвестиційних рішень.
- Забезпечення можливості поповнення рахунку фіатними грошима (з банківських карт, електронних гаманців) та криптовалютами з власних гаманців користувача.

– Надання інвесторам різних планів та інвестиційних програм, дозволяючи їм приймати обґрунтовані інвестиційні рішення та оптимізувати свої портфелі з метою максимізації прибутку та мінімізації ризиків[5].

– Забезпечення максимальної безпеки зберігання коштів користувачів в системі, конфіденційності та цілісності даних.

Для забезпечення більш глибокого розуміння вимог і очікувань зацікавлених сторін, збалансування потреб усіх учасників проекту, при його реалізації, необхідно враховувати наступні пункти:

Потреби інвесторів

- Зручний доступ до торгівлі криптовалютою
- Просте поповнення рахунку та виведення коштів
- Інформація про стан рахунку та історію операцій
- Аналіз ринку та підтримка в прийнятті інвестицій
- Безпека збереження коштів

Потреби власників (бізнесу)

- Залучення нових клієнтів та зростання капіталізації
- Збільшення кількості та обсягів транзакцій
- Позитивний імідж та лояльність клієнтів
- Мінімізація витрат на підтримку
- Дотримання правових норм

Потреби розробників

- Чіткі вимоги до системи
- Нормований обсяг робіт
- Вчасне тестування та верифікація
- Зворотний зв'язок від зацікавлених осіб

Виконання цих потреб дозволить розробити систему, яка не лише вирішує поточні завдання, але й має потенціал для адаптації до майбутніх викликів та розвитку готельного ринку.

Функціональні вимоги до інформаційної системи криптоінвестування:

1. Реєстрація та аутентифікація користувачів.

Користувач повинен мати можливість зареєструватися в системі за допомогою електронної пошти. В рамках реєстрації користувач вказує ПІБ, email, номер телефону та пароль. Система здійснює перевірку вказаних при реєстрації даних, підтвердження email. Користувач, зареєстрований в системі, може аутентифікуватися за допомогою email та пароля. Система має передбачати функцію відновлення пароля на випадок його втрати або несанкціонованого доступу в аккаунт[30].

2. Робота з рахунками та транзакціями.

Після входу користувачу має надаватись інформація щодо балансів на його рахунках, історії транзакцій, портфелю активів. Користувач повинен мати можливість інвестувати свої кошти за вибраним планом та мати можливість виведення криптовалюти на зовнішні гаманці

3. Аналітика та звітність щодо портфелю.

Користувач може відслідковувати зміну балансу, прибутковість портфеля і окремих активів у часі. Система формує звіти про стан портфеля, історію транзакцій користувача в розрізі вибраного проміжку часу.

4. Безпека та конфіденційність.

Система повинна забезпечувати безпечне збереження та неможливість несанкціонованого доступу до приватних ключів і коштів користувачів. Всі операції в системі мають проходити строгу верифікацію та підтвердження зі сторони користувача. Конфіденційні дані клієнтів мають бути недоступні стороннім особам

5. Підтримка клієнтів та комунікація

Реалізувати модуль техпідтримки із системи тікетів та бази знань. Система має відстежувати статуси вирішення поставлених клієнтами питань. Передбачити різні канали зворотного зв'язку (email, телефон)

Можливі нефункціональні вимоги до інформаційної системи криптоінвестицій:

1. Вимоги до інтерфейсу.

Інтерфейс користувача має бути реалізований у вигляді веб-додатку, оптимізованого для використання на настільних комп'ютерах та мобільних пристроях. Дизайн інтерфейсу повинен бути сучасним, інтуїтивно зрозумілим для користувачів. Час завантаження сторінок системи – не більше 2 секунд для швидкісного інтернету. Система має коректно відображатися і функціонувати в останніх версіях браузерів: Chrome, Firefox, Safari, MS Edge

2. Вимоги до продуктивності.

Система повинна забезпечувати одночасну роботу до 100 авторизованих користувачів. Максимальне навантаження – до 10 000 транзакцій на добу

3. Вимоги до надійності та безпеки.

Резервування даних має забезпечувати можливість відновлення системи після збоїв. Доступ до даних користувачів мають лише авторизовані співробітники

4. Вимоги до відповідності стандартам і нормативним документам.

Система має відповідати сучасним стандартам і рекомендаціям щодо інформаційної безпеки. Функціонування системи повинне відповідати законодавству щодо операцій з криптовалютами

Комбінація цих функціональних і нефункціональних вимог (див. рисунок 2.17) дозволить створити ефективну, надійну та користувацьки-орієнтовану інформаційну систему, яка відповідатиме сучасним вимогам та тенденціям світу криптовалют.

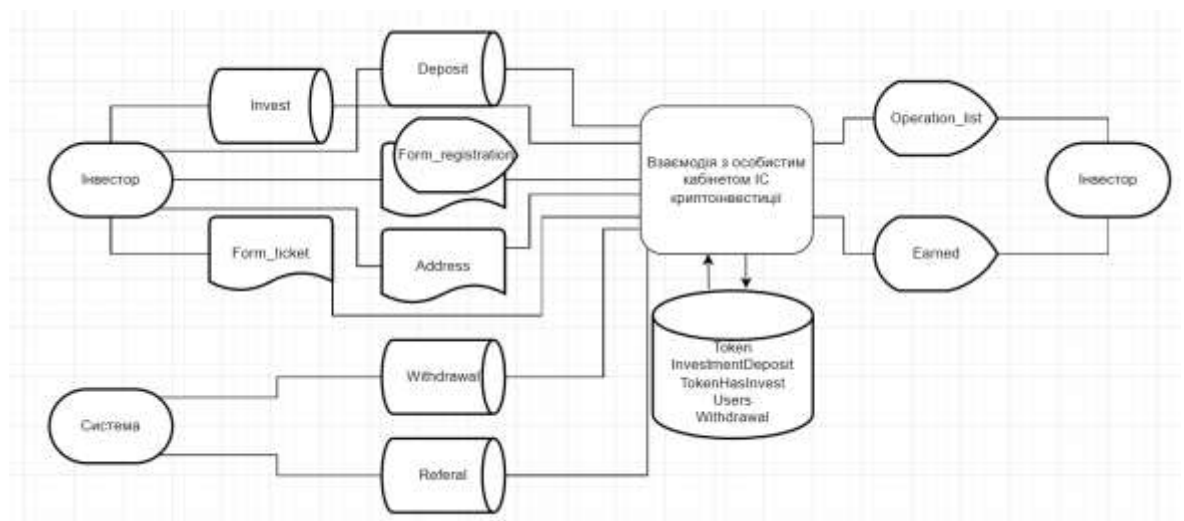


Рисунок 2.17 – Інформаційна модель задачі

2.4.1 Вихідна інформація

Вихідна інформація представляє результати обробки вхідних даних і взаємодії з користувачами. Також вона забезпечує користувачам зрозуміле та актуальне розуміння стану їхніх інвестицій, дозволяє відстежувати ефективність їхніх інвестиційних стратегій та гарантує, що вони своєчасно інформовані про будь-які важливі події або потреби в діях з їхнього боку. Перелік і опис вихідних повідомлень наведено у табл. 2.1.

Таблиця 2.1 – Перелік та опис вихідних повідомлень

Назва вихідного повідомлення	Ідентифікатор	Форма подання і вимоги до неї	Періодичність видання	Термін видання і допустимий час затримки	Користувач інформації
Список операцій	Operation_list	Екранна форма	По мірі надходження запитів на відображення	Одразу після запиту (затримка до 5 сек)	Інвестор
Заробіток з інвестицій	Earned	Екранна форма	По мірі надходження запитів на відображення	Одразу після запиту (затримка до 5 сек)	Інвестор

2.4.2 Вхідна інформація

Вхідними повідомленнями для здійснення контролю за основними операціями ІС для криптоінвестицій є форма реєстрації, депозит токенів, виведення коштів, винагорода за рефералів, форма зворотного зв'язку, інвестиція та додавання крипто-гаманців. Ці дані використовуються для обробки запитів на транзакції, таких як інвестиції, виведення коштів, а також для оновлення інформації про ринкові ціни та управління профілем користувача[28]. Без цієї вхідної інформації система не зможе адекватно

реагувати на потреби користувачів та забезпечувати актуальність та ефективність своїх сервісів. Перелік і опис вхідних повідомлень наведено у табл. 2.2.

Таблиця 2.2 – Перелік та опис вхідних повідомлень

Назва вхідного повідомлення	Ідентифікатор	Форма подання	Термін і частота надходження	Джерело
Форма реєстрації клієнтів	<i>Form_registration</i>	Документ, Екранна форма	По мірі реєстрації студентів	Дані від інвестора
Депозит токенів	<i>Deposit</i>	Файл	По мірі здійснення депозиту	Дані від інвестора
Виведення коштів	<i>Withdrawal</i>	Файл	По мірі надсилання запиту на виведення коштів	Дані від сайту
Винагорода за рефералів	<i>Referral</i>	Файл	По мірі реєстрації рефералів	Дані від сайту
Форма зворотного зв'язку	<i>Form_ticket</i>	Документ	По мірі створення нових тікетів	Дані від інвестора
Інвестиція	<i>Invest</i>	Файл	По мірі запитів на інвестування	Дані від інвестора
Додавання крипто-гаманців	<i>Address</i>	Документ	Один раз, перед виведенням коштів	Дані від інвестора

2.5 Математичне забезпечення та алгоритм функціонування системи

2.5.1 Математичний опис

Економіко-математична модель ІС для криптоінвестицій може бути розроблена для розрахунку суми виплати реферальної програми, відсотку користувачів, що зробили інвестицію, середнього часу інвестицій, загальну суму інвестицій та відсоток закритих тікетів.

Визначення суми виплати коштів за реферала:

- Сума внесених і-тих коштів ref-им користувачем – L_{ir}
- Відсоток R нарахувань від L-ої суми депозиту – 15
- Сума виплати і-тих коштів за ref-ого користувача – S_{ir}

$$S_{ir} = \frac{L_{ir}}{100} * 15$$

Відсоток користувачів, які зробили інвестицію:

- Загальна кількість і-тих користувачів N_i
- Кількість і-тих користувачів, що інвестували D_i
- Відсоток і-тих користувачів, які інвестували у проект P_i

$$P_i = \left(\frac{D_i}{N_i} \right) * 100$$

Середній час інвестиційних планів:

- Термін інвестиції в і-тий інвестиційний план – T_i
- Кількість інвестиційних планів – N
- Середній час і-тих інвестиційних планів – V_i

$$V_i = \frac{\sum T_i}{N}$$

Середня сума інвестицій:

- Інвестиція одного n-го користувача – I_n
- Кількість I-інвестицій n-го користувача – C_n
- Середня сума I інвестиції n-го користувача – A_n

$$A_n = \frac{\sum I_n}{C_n}$$

Відсоток закритих тікетів:

- Закриті тікети m-ими менеджерами від i-их клієнтів – C_{mi}
- Всього тікетів від i-их клієнтів – T_i
- Відсоток закритих тікетів m-ими менеджерами від i-их клієнтів –

P_{mi}

$$P_{mi} = \frac{C_{mi}}{T_i} * 100$$

Залежно від мети та специфіки задачі, можуть бути використані інші параметри та формули. Економіко-математична модель допоможе оцінити які інвестиції є найпопулярнішими, загальну популярність сервісу та інвестицій в цілому.

2.5.2 Алгоритм розв'язання задачі

Алгоритм розв'язання задачі розділено на різні кроки та подано на рисунках 2.18-2.23.

Форми користувацького інтерфейсу (наприклад, Form_registration, Form_login) використовуються для введення даних користувачем під час реєстрації чи входу в систему.

Логічні переходи представляють точки прийняття рішень, де в залежності від вхідних даних або дій користувача вибирається подальший напрямок алгоритму.

Процеси описують виконувані дії, такі як "Вивести кошти", "Змінити пароль", "Налаштувати акаунт".

Бази даних символізують місця зберігання даних, таких як інформація про користувачів, транзакції чи логи.

Дії користувача представляють введення або взаємодію, яку користувач має виконати, наприклад, подати заявку або здійснити платіж.

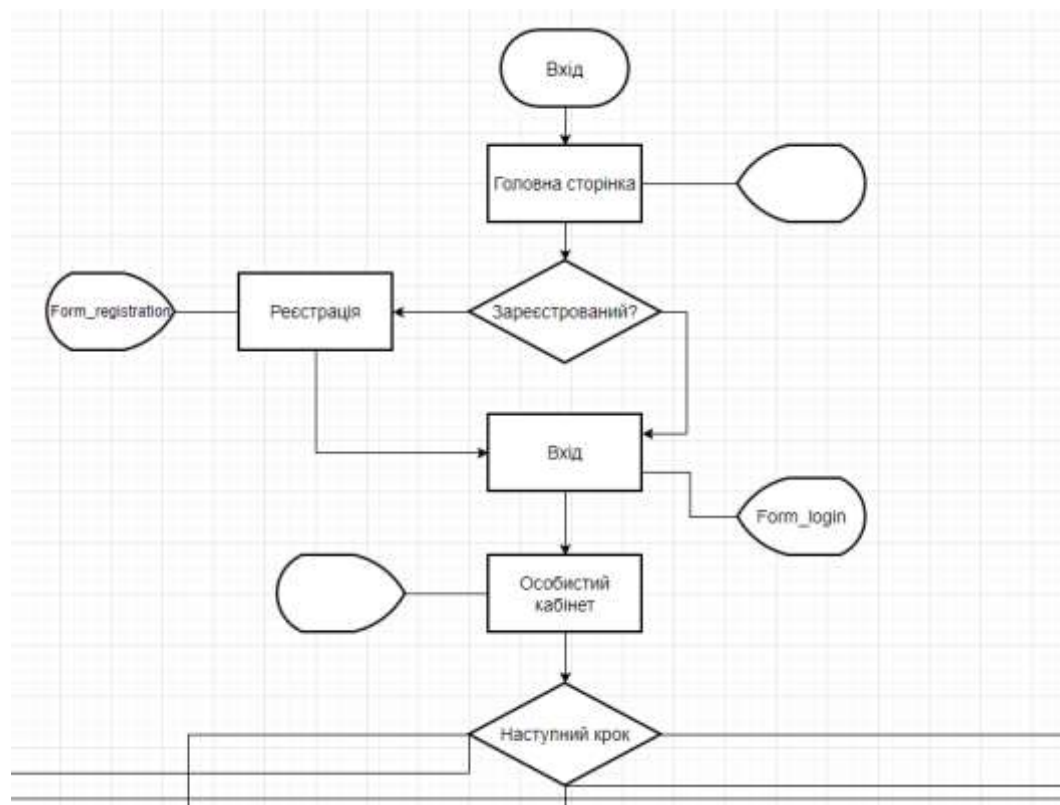


Рисунок 2.18 – Етап входу в акаунт в алгоритмі розв'язання задачі операцій особистого кабінету

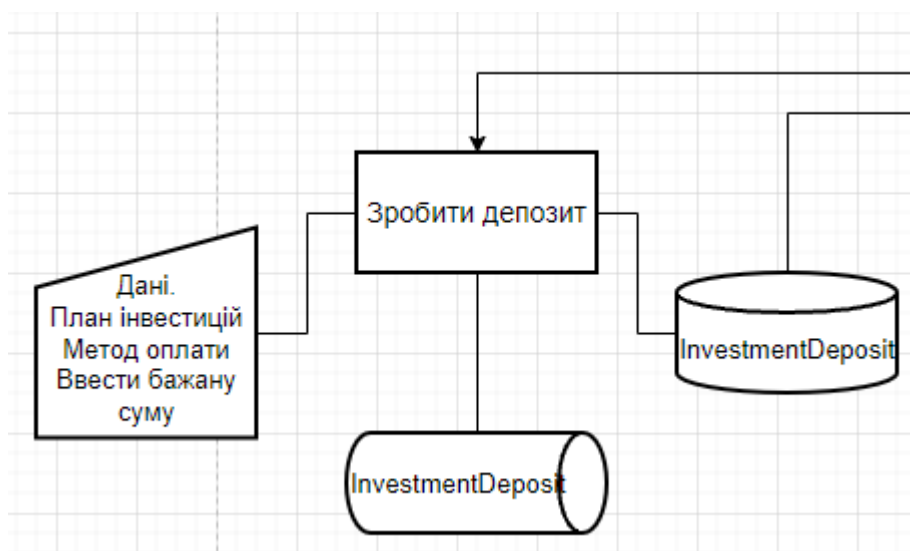


Рисунок 2.19 – Крок з депозитом коштів в алгоритмі розв’язання задачі

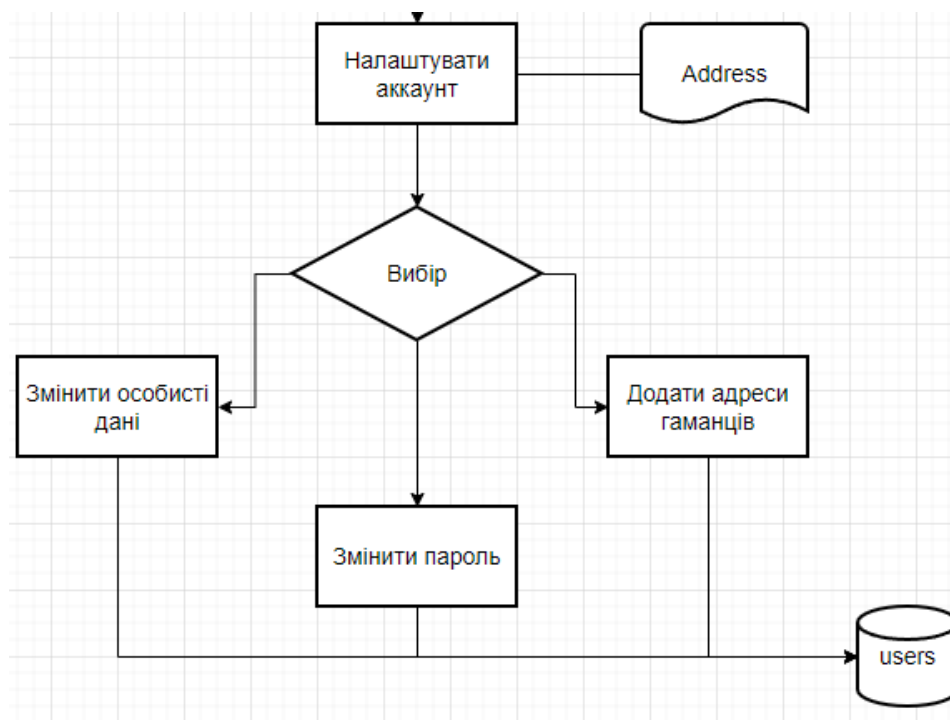


Рисунок 2.20 – Крок з налаштуваннями аккаунту в алгоритмі розв’язання задачі

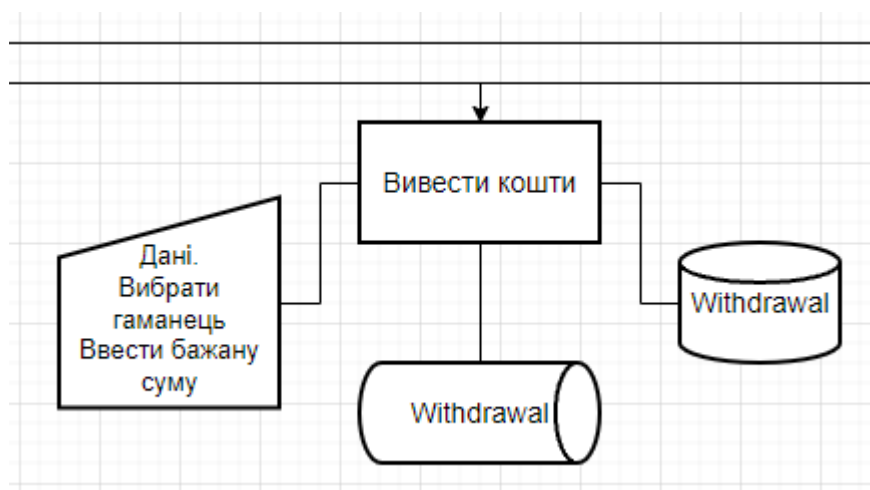


Рисунок 2.21 – Крок з виведенням коштів в алгоритмі розв’язання задачі

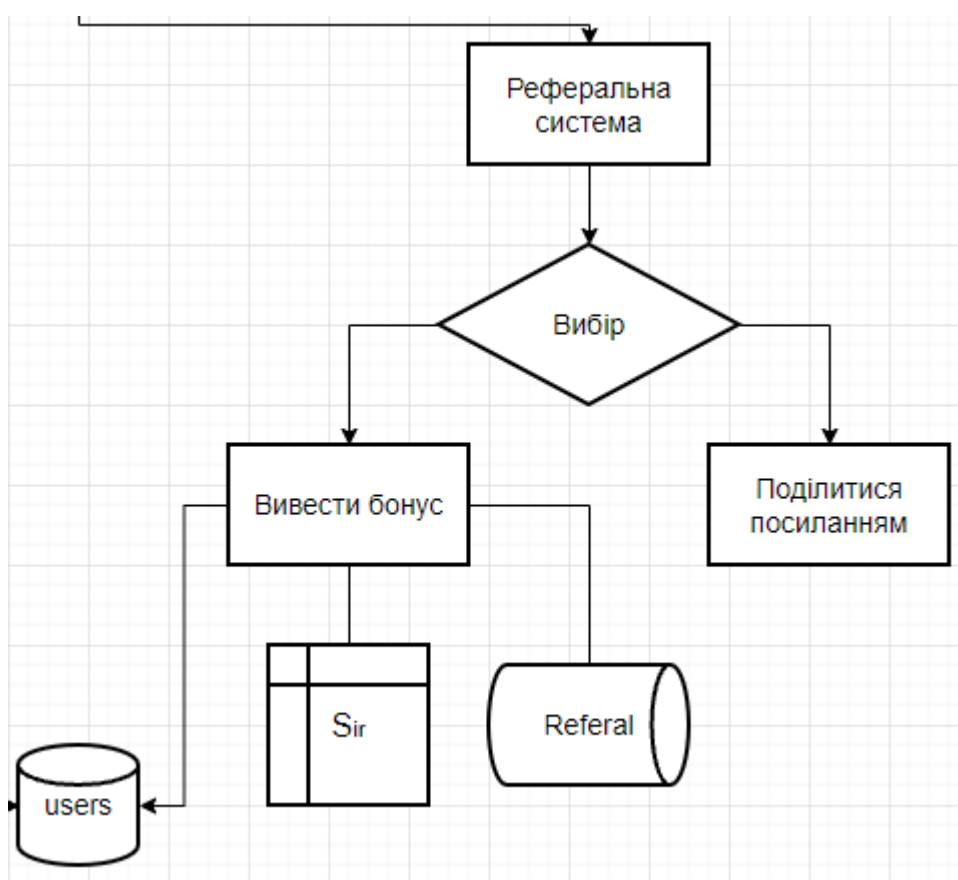


Рисунок 2.22 – Крок з операціями реферальної системи в алгоритмі розв’язання задачі

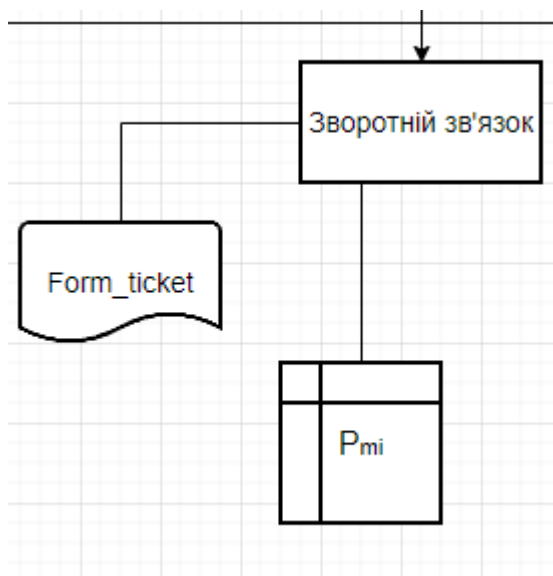


Рисунок 2.23 – Крок зі зв'язком зі службою підтримки в алгоритмі розв'язання задачі

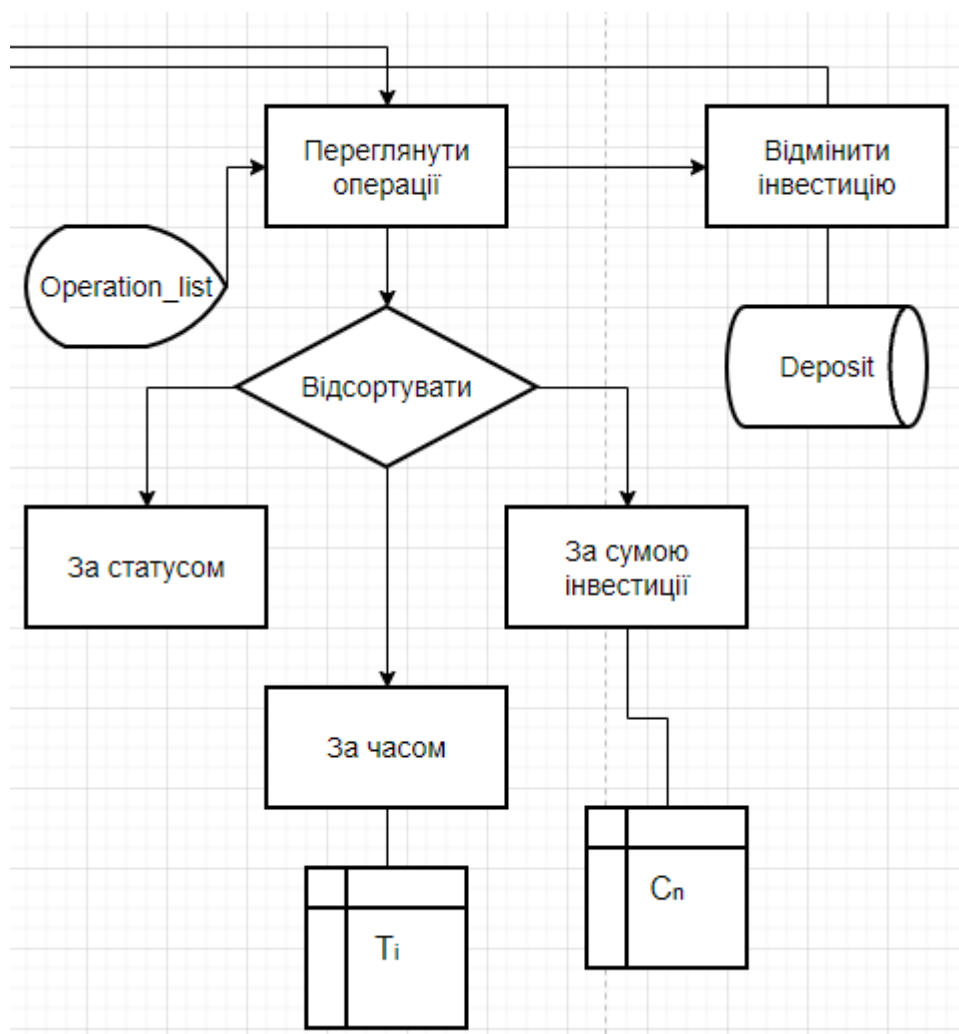


Рисунок 2.23 – Крок з переглядом операцій над інвестиціями в алгоритмі розв'язання задачі

2.6 Моделювання поведінки системи

Моделювання поведінки допомагає візуалізувати та зрозуміти взаємодію між різними компонентами системи, а також процеси, які відбуваються в ній.

Для моделювання поведінки системи можна використовувати різні підходи та методології, такі як діаграми діяльності UML, діаграми станів, діаграми послідовності та інші. Ці інструменти дозволяють графічно представити потік даних, послідовність дій, переходи між станами та взаємодію між об'єктами системи.

Наприклад, діаграми діяльності можна використовувати для моделювання процесів, таких як реєстрація нового користувача, створення крипто-гаманця, купівля або продаж криптовалют тощо. Вони візуалізують послідовність кроків, умови переходів, розгалуження та з'єднання потоків, що полегшує розуміння та аналіз цих процесів.

Діаграми станів, у свою чергу, можуть бути використані для моделювання життєвого циклу об'єктів системи, таких як стани гаманця (створений, активний, заблокований) або стани транзакції (ініційована, в процесі, завершена). Ці діаграми допомагають визначити можливі переходи між станами, тригери, що викликають ці переходи, та дії, які повинні виконуватися в кожному стані.

Діаграми послідовності дозволяють змодельовати взаємодію між об'єктами системи в часі, показуючи обмін повідомленнями та порядок виконання операцій. Це може бути корисним для візуалізації процесів, таких як авторизація користувача, здійснення транзакції або відображення аналітичних даних.

Моделювання поведінки системи "Криптоінвестиції" допомагає розробникам краще зрозуміти вимоги до системи, виявити потенційні проблеми та недоліки на ранніх етапах розробки, а також забезпечити



Рисунки 2.25-2.26 – Матриця відповідності вимог і прецедентів

Наступним кроком перейдемо до створення текстових сценаріїв. Текстові сценарії є ефективним способом детального опису варіантів використання інформаційної системи. Вони допомагають зрозуміти, як система повинна працювати в різних ситуаціях, і забезпечують підґрунтя для розробки, тестування та документації.

Сценарії варіантів використання зазвичай складаються у вигляді структурованих текстових документів, які описують послідовність кроків, що виконуються системою та користувачами для досягнення певної мети.

Детальні текстові сценарії допомагають усім зацікавленим сторонам (замовникам, аналітикам, розробникам, тестувальникам) отримати спільне розуміння очікуваної поведінки системи. Вони є основою для розробки функціональних специфікацій, тест-кейсів і користувацької документації. Крім того, сценарії полегшують виявлення можливих проблем та непередбачених ситуацій на ранніх етапах, що дозволяє своєчасно внести корективи.

На рисунках 2.27-2.29 продемонстровано приклад текстового сценарію для прецеденту «Депозит». З цього сценарію можна побачити як проходить процес інвестицій.

Type:

Scenario:

Basic Path

Basic Path

</

Рисунок 2.27 – Базовий сценарій інвестицій при успіху

Type:	Scenario:	
Alternate	Alternate Path.	
		
Step	Action	
	1	<u>Система відкриває сторінку вибору платіжної системи.</u>
	2	<u>Користувач обирає "Скасувати" замість введення платіжних даних.</u>
	3	<u>Система скасовує операцію і повертає користувача до сторінки з інвестиційними планами.</u>

Рисунок 2.28 – Альтернативний сценарій при негативному результаті

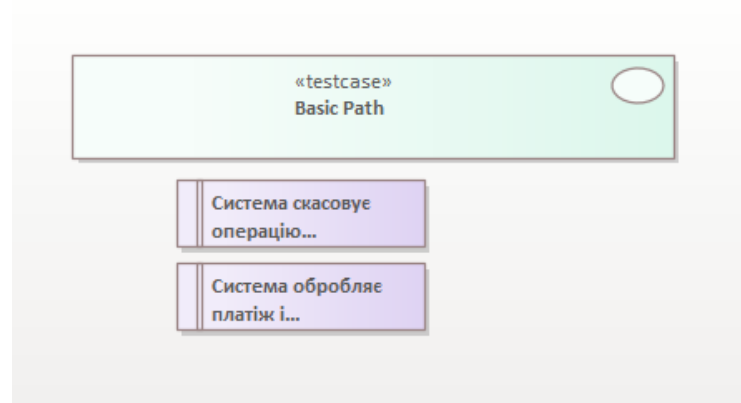


Рисунок 2.29 – Готовий сценарій з позитивним та негативним результатом

Далі наведено приклади поведінки системи, залежних від дій користувача на діаграмах послідовностей.

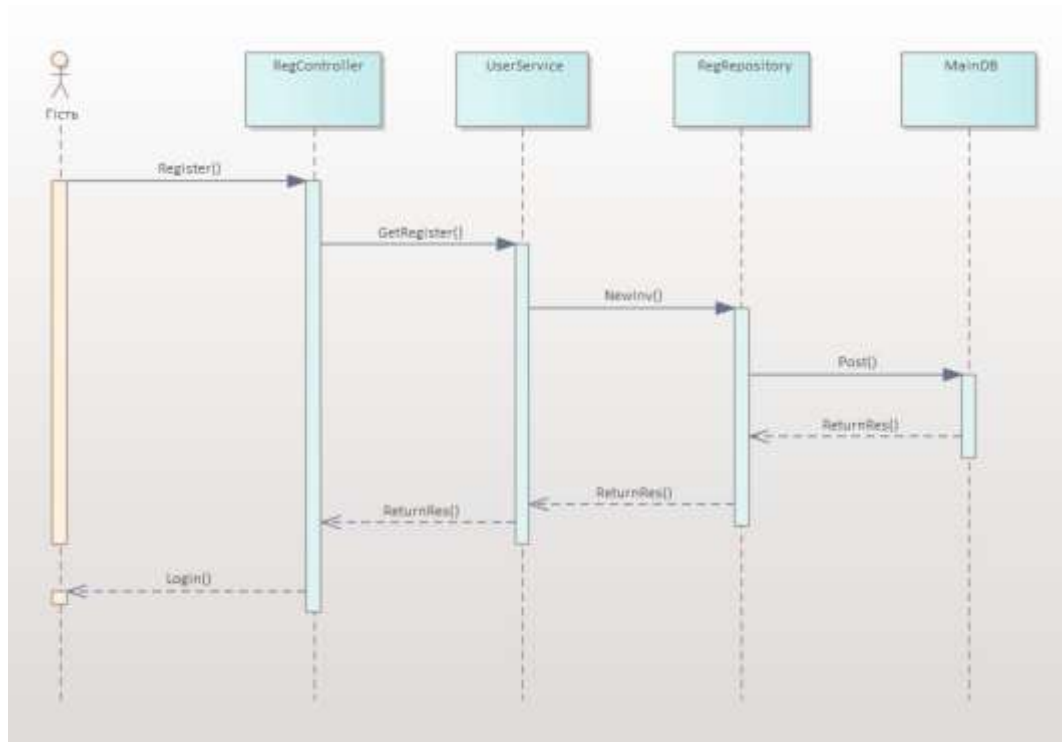


Рисунок 2.30 – Послідовність дій при реєстрації користувача

Ця діаграма послідовності показує процес реєстрації (див. рисунок 2.30). Користувач вводить свої дані для реєстрації і натискає кнопку “Зареєструватися”. Система отримує дані користувача і перевіряє їх на валідність. Якщо дані невалідні, система повертає відповідне повідомлення про помилку. Якщо користувач не заблокований і не в чорному списку, система генерує код підтвердження і відправляє його на електронну пошту користувача. Система також зберігає дані користувача в базі даних із статусом “не підтверджений”[30].

Наступна діаграма демонструє нам процес отримання даних токенів, що доступні на сайті, а саме їх актуальної ціни. Для цього відправляється запит, на парсинг даних зі стороннього ресурсу, наприклад, криптовалютної біржі (див. рисунок 2.31).

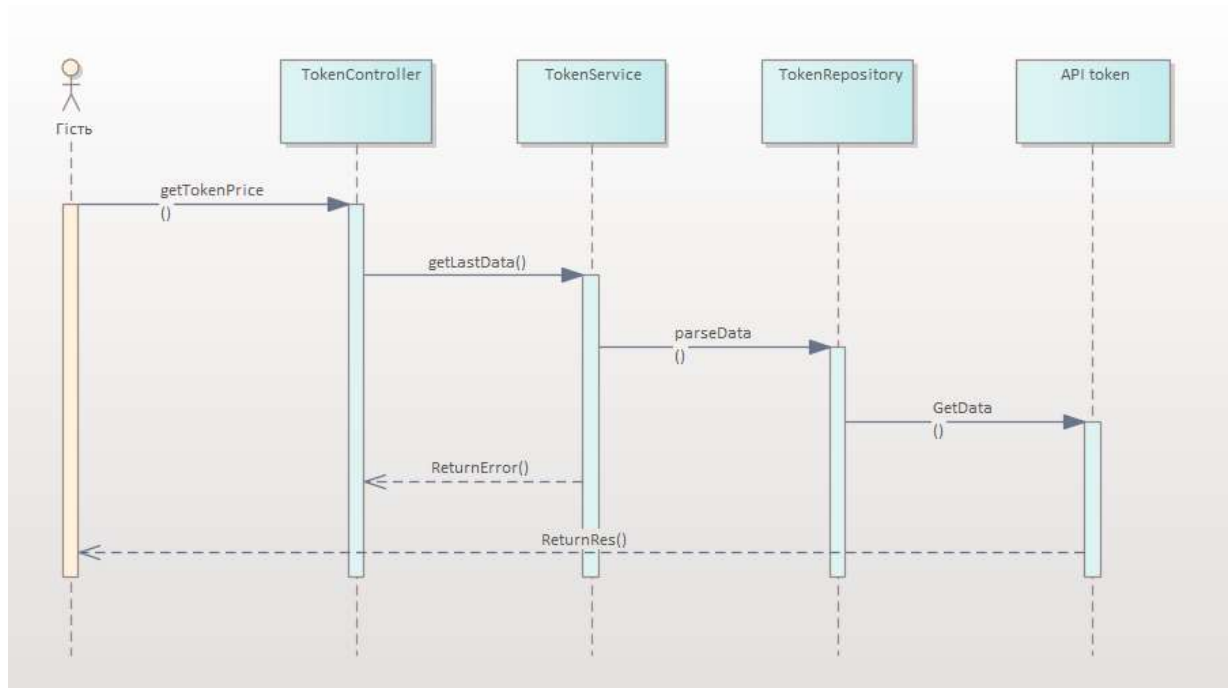


Рисунок 2.31 – Послідовність дій при запиті на отримання даних токенів

Для того, щоб вивести кошти з сайту, відправляється запит на перевірку поточного балансу користувача, після чого запускається процес виводу токенів, будь який результат зберігається у базу даних та виводиться користувачу (див. рисунок 2.32).

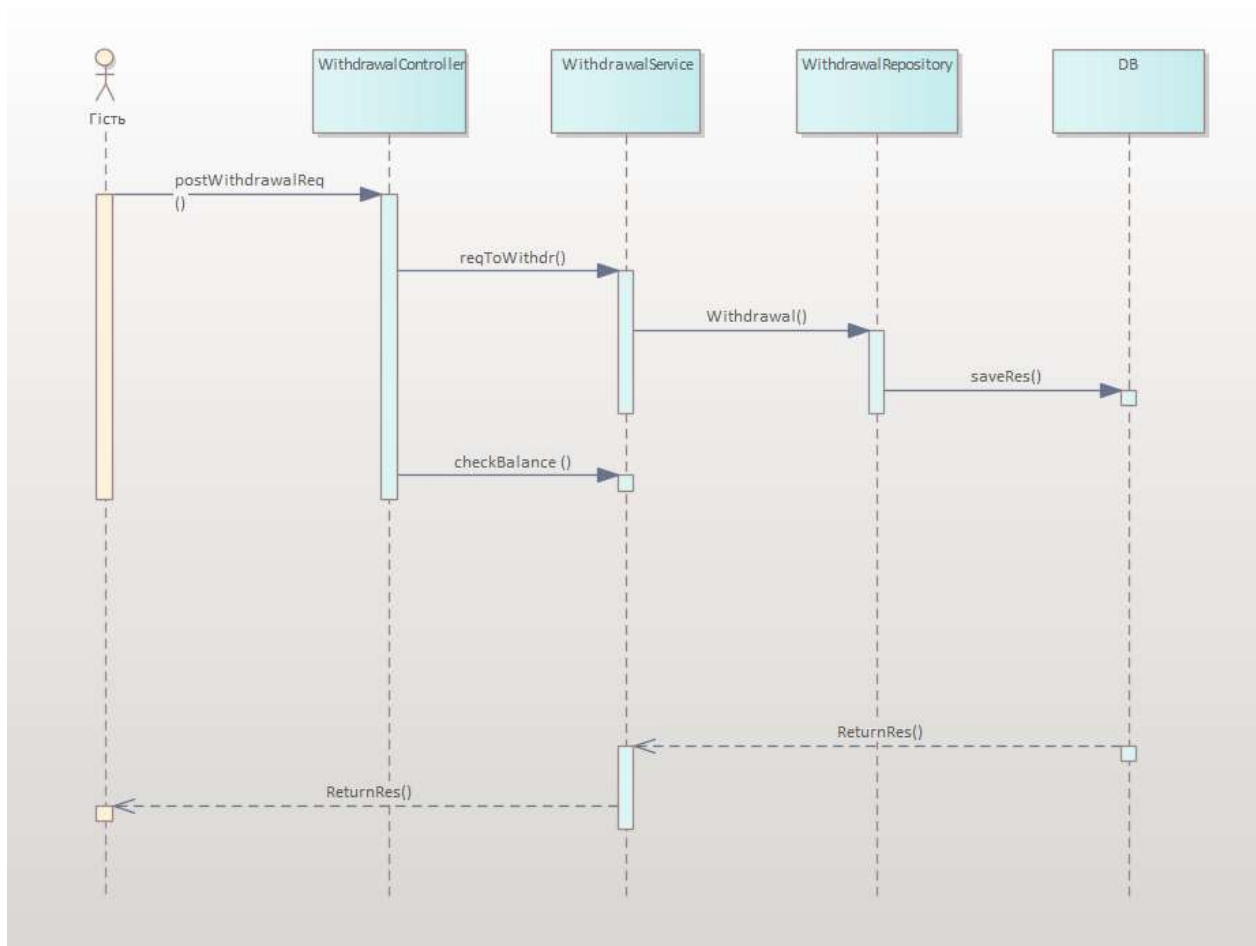


Рисунок 2.32 – Послідовність процесів при виведенні коштів

Для депозиту токенів відправляється запит на сторонній API сервіс, для отримання поточного балансу гаманця, якщо він відмінний від нуля та більший або дорівнює мінімально необхідній сумі – починається процес депозиту, а саме підтвердження та відправка запитів. Результат повертається користувачу (див. рисунок 2.33).

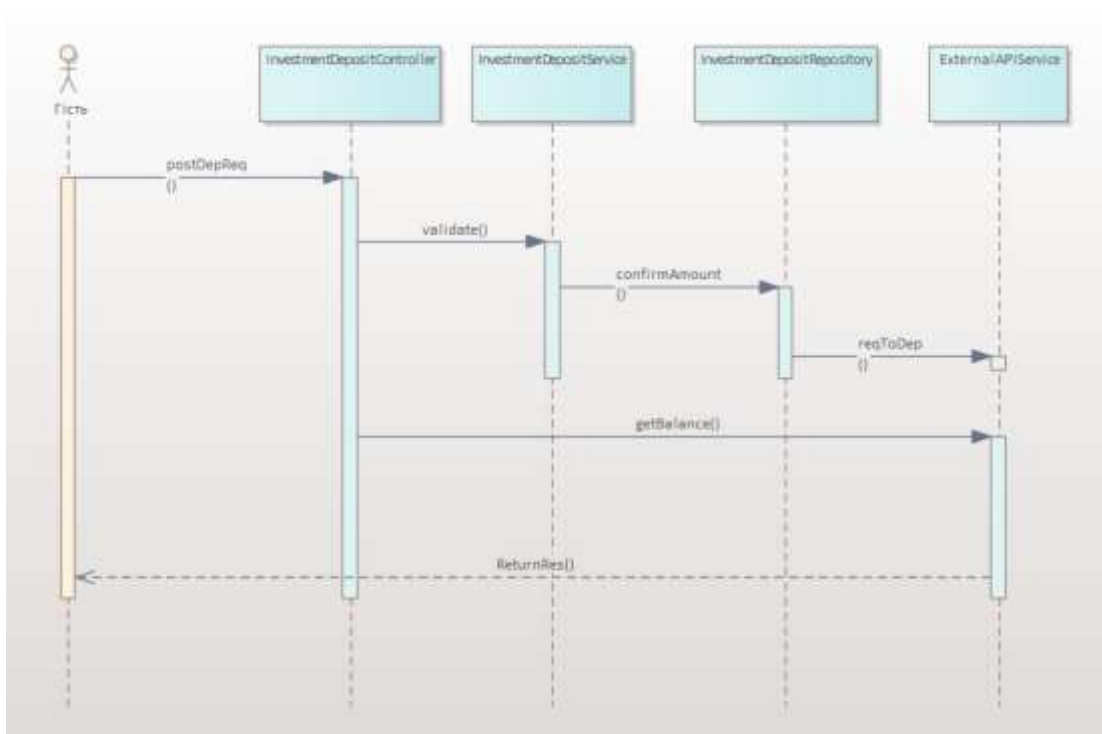


Рисунок 2.33 – Послідовність процесів при депозиті коштів

Для нарахування бонусного балансу користувачу, за реєстрацію нового клієнта, перевіряється його реферальна адреса, якщо реєстрація підтверджується та новий клієнт вносить депозит, користувач отримує відсоток від депозиту та може його вивести (див. рисунок 2.34).

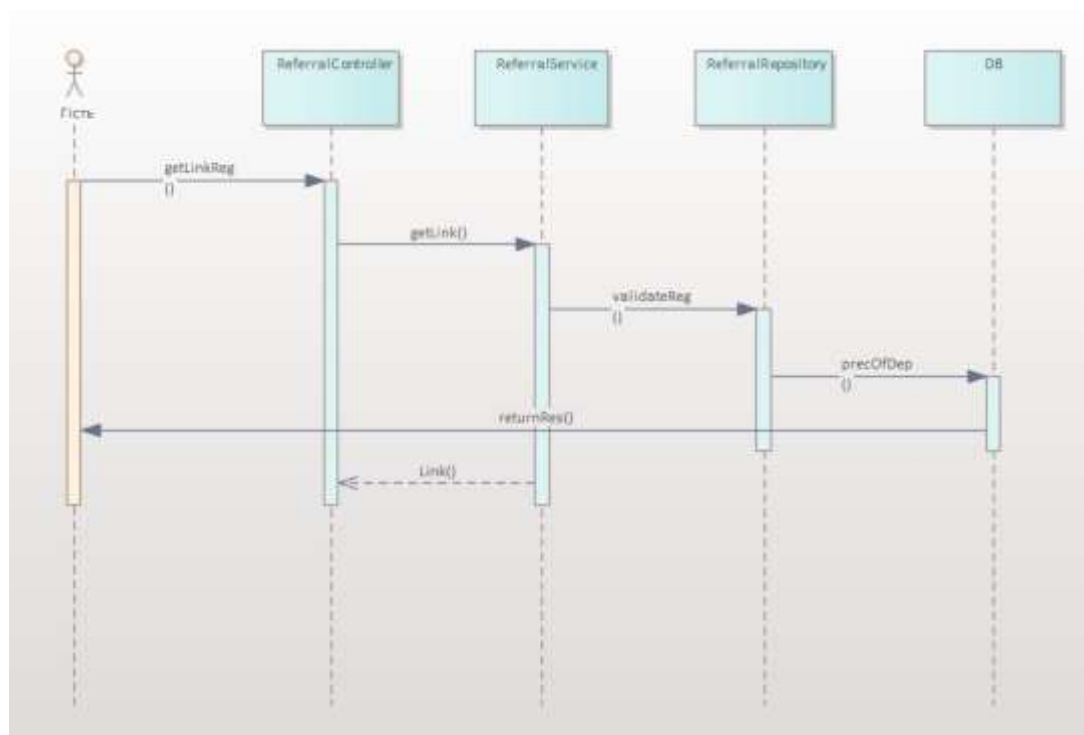


Рисунок 2.34 – Послідовність процесів при взаємодії з рефералами

2.6.1 Моделювання структури системи

В проєкті застосовується архітектурний шаблон MVC (Model-View-Controller), який є визнаним стандартом у сфері розробки програмного забезпечення[15]. MVC дозволяє структурувати програму на три ключові компоненти: модель, представлення та контролер, що сприяє ефективному розділенню відповідальностей.

1. Модель (Model). Модель відповідає за представлення бізнес-даних та бізнес-логіку програми. Це може бути окремий клас або набір класів, які управляють даними, забезпечують доступ до бази даних, реалізують операції з даними, здійснюють валідацію та обробляють бізнес-правила.

2. Представлення (View). Компонент представлення відповідає за візуалізацію даних для користувача. Це може бути виглядом HTML-сторінки, шаблоном веб-інтерфейсу чи графічним інтерфейсом користувача. Представлення отримує дані з моделі та візуалізує їх для користувача.

3. Контролер (Controller). Контролер служить як медіатор між моделлю та представленням, обробляючи запити від користувача, взаємодіючи з моделлю для виконання необхідних операцій, та вибираючи відповідне представлення для відображення результатів. Контролер також передає вхідні дані від користувача до моделі для обробки та вибору відповідного візуального представлення.

Основною перевагою використання MVC є чітке розділення відповідальностей між компонентами, що робить систему більш організованою, підтримуваною та гнучкою для розвитку[16]. Такий підхід сприяє легкій інтеграції змін у бізнес-логіці без впливу на візуальну частину та навпаки, забезпечуючи більш ефективне управління проєктом.

На рисунку 2.35 продемонстровано діаграму класів, яка відображає взаємозв'язки та залежності між різними елементами системи, розроблену в Enterprise Architect[3].

Фрагменти коду, що було створено для реалізації бекенду системи та фронтенду, наведені у «Додатку Б».

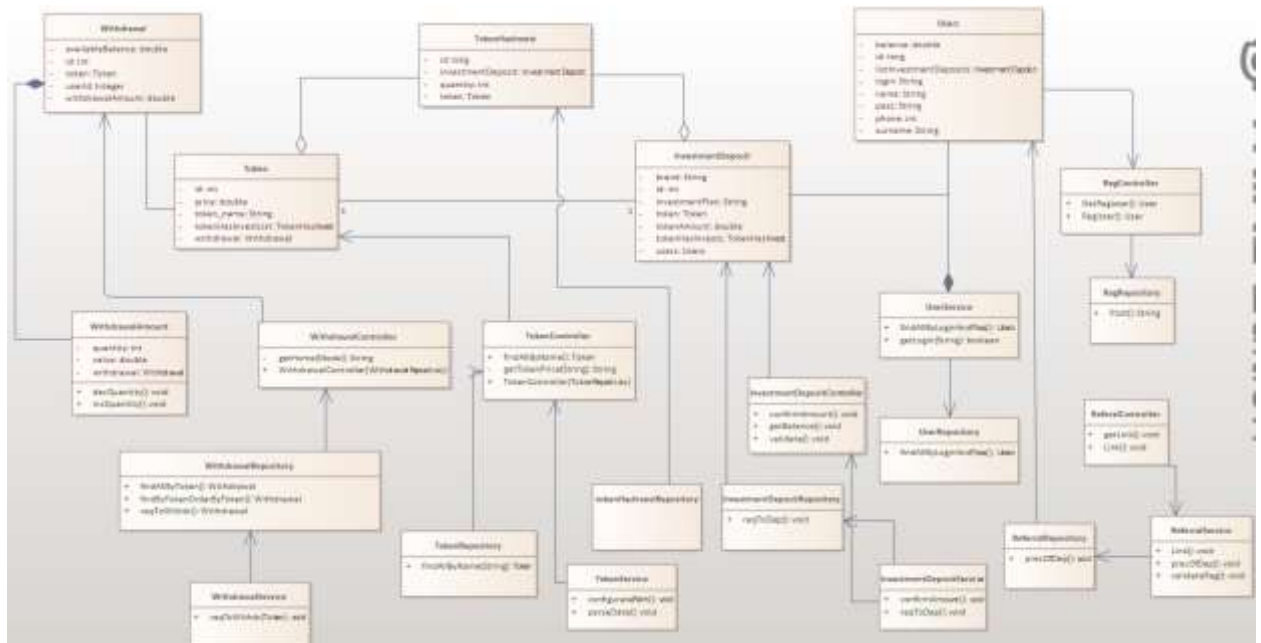


Рисунок 2.35 – Діаграма класів системи

Аналізуючи класи за архітектурним підходом, можна виділити:

CyTHoSti (Model):

Users. Цей клас представляє користувачів системи. Він містить атрибути, такі як id, login, name, surname, pass, phone, availableBalance, deposits, token.

Token. Цей клас представляє токени, які використовуються в системі. Він містить атрибути, такі як id, name, price, investmentPlan, tokenAmount, tokenHasInvests, users.

InvestmentDeposit. Цей клас представляє інвестиційні депозити, які роблять користувачі. Він містить атрибути, такі як id, quantity, value, token, user, withdrawal.

Withdrawal. Цей клас представляє зняття коштів, які роблять користувачі. Він містить атрибути, такі як id, quantity, value, withdrawalAmount, token, user.

TokenHasInvests. Цей клас представляє зв'язок між токенами та інвестиційними депозитами. Він містить атрибути, такі як id, token, investmentDeposit.

Відносини:

Користувач володіє токенами. Цей зв'язок представляє зв'язок один до багатьох між класами Users та Token. Користувач може володіти одним або кількома токенами.

Токен має інвестиційні депозити. Цей зв'язок представляє зв'язок один до багатьох між класами Token та InvestmentDeposit. Токен може мати один або кілька інвестиційних депозитів.

Інвестиційний депозит належить користувачеві. Цей зв'язок представляє зв'язок один до одного між класами InvestmentDeposit та Users. Інвестиційний депозит може належати лише одному користувачеві.

Інвестиційний депозит використовує токен. Цей зв'язок представляє зв'язок один до одного між класами InvestmentDeposit та Token. Інвестиційний депозит може використовувати лише один токен.

Зняття коштів належить користувачеві. Цей зв'язок представляє зв'язок один до одного між класами Withdrawal та Users. Зняття коштів може належати лише одному користувачеві.

Зняття коштів використовує токен. Цей зв'язок представляє зв'язок один до одного між класами Withdrawal та Token. Зняття коштів може використовувати лише один токен.

Функціональні можливості:

- Система дозволяє користувачам реєструватися та входити в систему.
- Користувачі можуть переглядати список доступних токенів.
- Користувачі можуть купувати токени.
- Користувачі можуть робити інвестиційні депозити в токени.
- Користувачі можуть знімати кошти з інвестиційних депозитів.
- Система зберігає історію транзакцій користувачів.

Діаграми визначення блоків надають систематизований підхід до аналізу та проектування системи, сприяючи ідентифікації компонентів, їх функцій та взаємозв'язків. Структурування функціоналу дозволяє зобразити функціональні блоки системи та їх взаємозв'язки. У контексті криптоінвестицій

це може означати виділення модулів для обробки та аналізу даних криптовалют, блоків для здійснення торгівельних операцій, систем для управління портфелем, інвестицій і т.д. (див. рисунок 2.36).

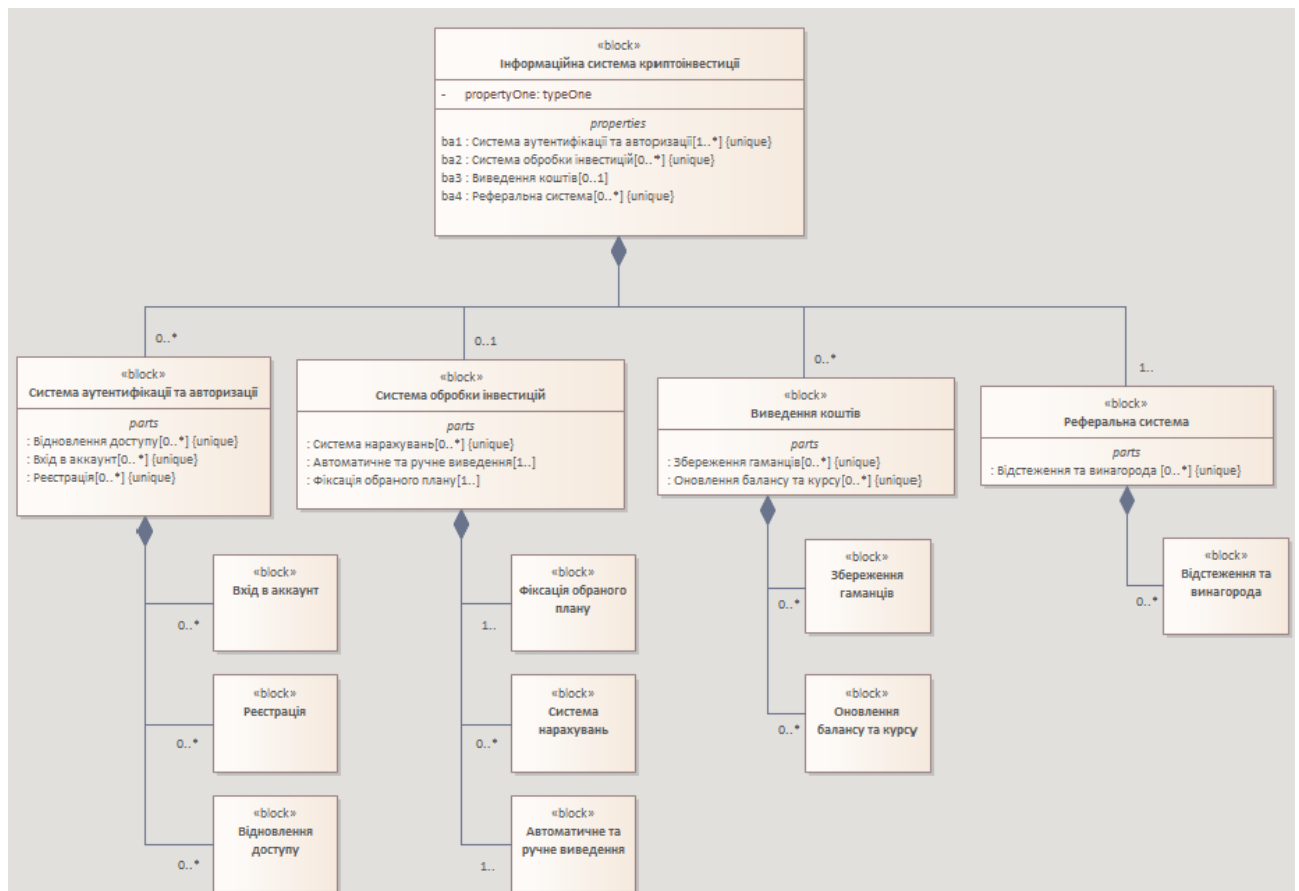


Рисунок 2.36 – Діаграма визначення блоків для подання внутрішньої структури системи (BDD)

На рисунку 2.37 можна побачити приклад виведення коштів. Діаграма починається з блоку GetBalance User, який отримує баланс користувача. Цей баланс потім надсилається до блоку Wallet, який зберігає його. Якщо користувач хоче вивести кошти, він вибирає токен та суму, яку хоче вивести. Ця інформація потім надсилається до блоку Withdrawal.

Блок Withdrawal спочатку перевіряє, чи має користувач достатньо коштів на виведення. Якщо так, він надсилає запит на виведення коштів на сервер за допомогою блоку PostReq.

Сервер перевіряє запит і, якщо він дійсний, надсилає токени користувачеві за допомогою блоку SendTokens. Після того, як токени надіслано користувачеві, блок SaveRes зберігає результати транзакції.

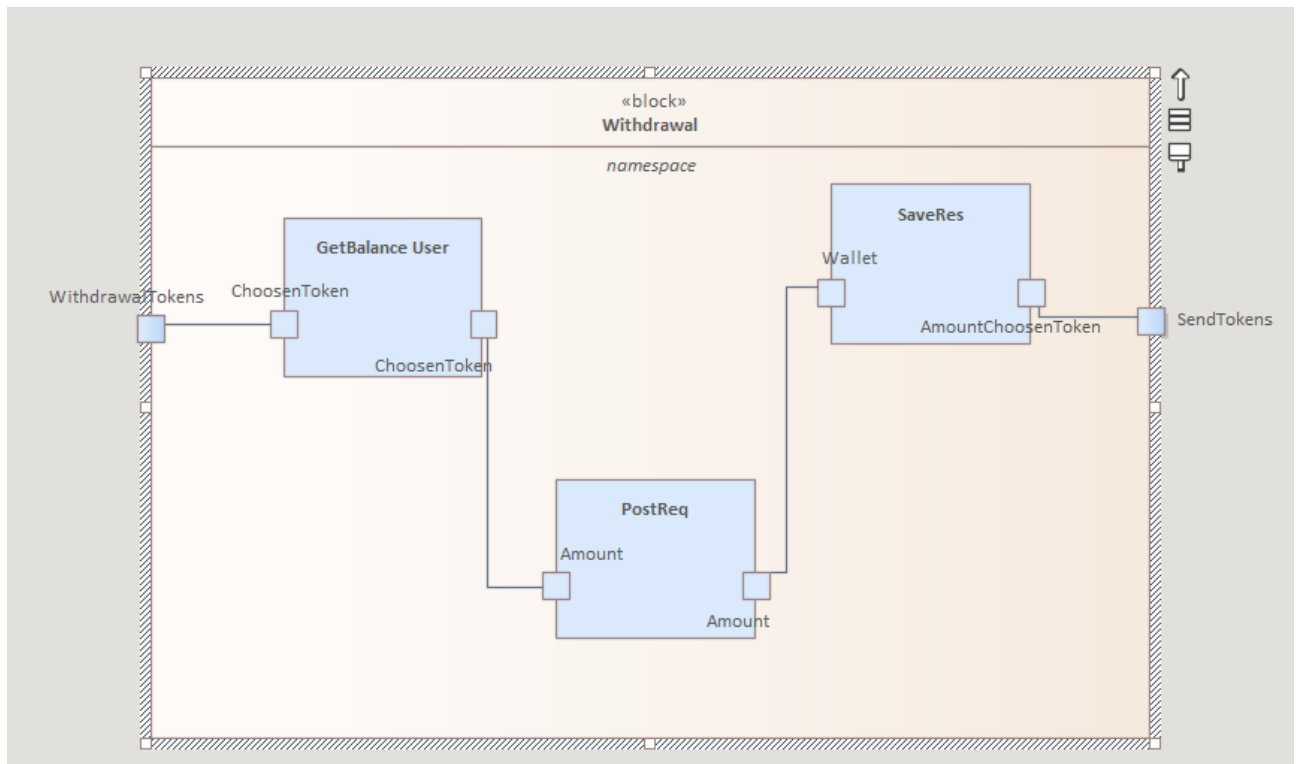


Рисунок 2.37 – Процес виведення коштів(IBD)

2.6.2 Моделювання користувацького інтерфейсу

На рисунках 2.38-2.40 зображено приклади користувацького інтерфейсу ІС, створені в Enterprise Architect[3]. Наведено можливості реалізації сторінки депозиту, реферальної системи та виведення коштів. Ці приклади ілюструють різні аспекти використання системи.

На рисунку 2.38 зображено інтерфейс сторінки депозиту, який призначений для вибору плану інвестицій та, безпосередньо, вибору бажаної валюти і введенню коштів на свій аккаунт.

Deposit

Reflective Investments

Tab 1 Tab 2 Tab 3

Select investment plan

Plan1
0.4%-0.6%

Plan2
0.7%-0.9%

Plan3
1%-1.2%

Select payment system

ETH

BTC

LTC

USDT

XRP

Make a deposit

Enter the amount
10

☐ From the payment system

Deposit

Рисунок 2.38 – Інтерфейс сторінки депозиту

На рисунку 2.39 продемонстровано інтерфейс сторінки реферальної системи. На ній відображається статистика запрошень, зароблені з цього кошти та саме посилання для запрошення.

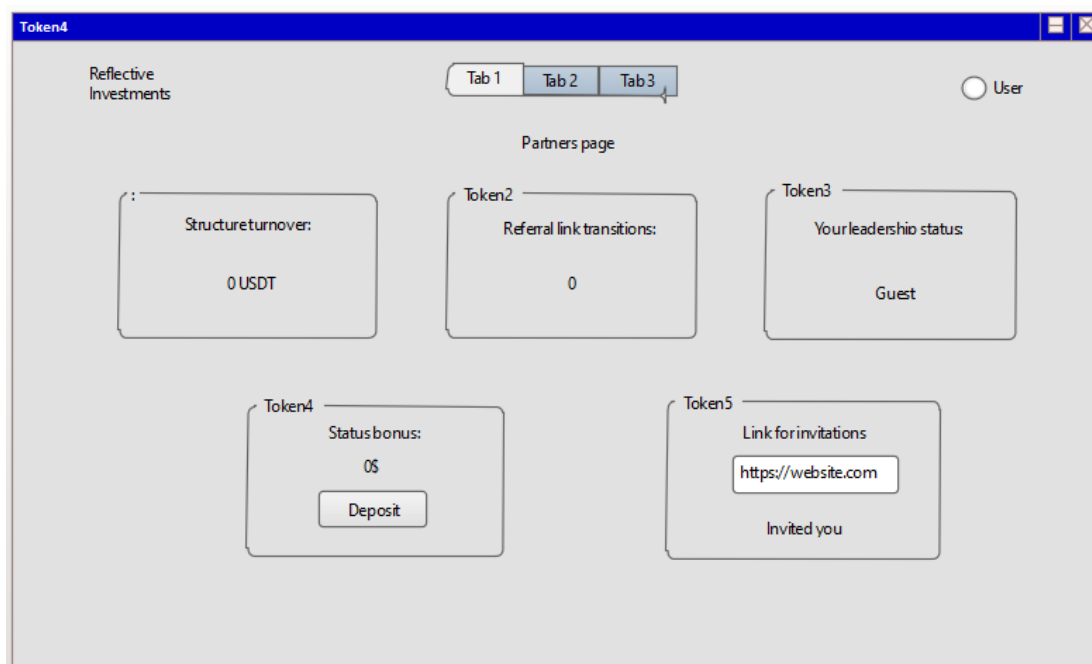


Рисунок 2.39 – Інтерфейс сторінки реферальної системи

На рисунку 2.40 продемонстровано сторінку виведення зароблених коштів. Для цього необхідно обрати систему оплати, з якої буде виконано виведення, та необхідну для цього суму.

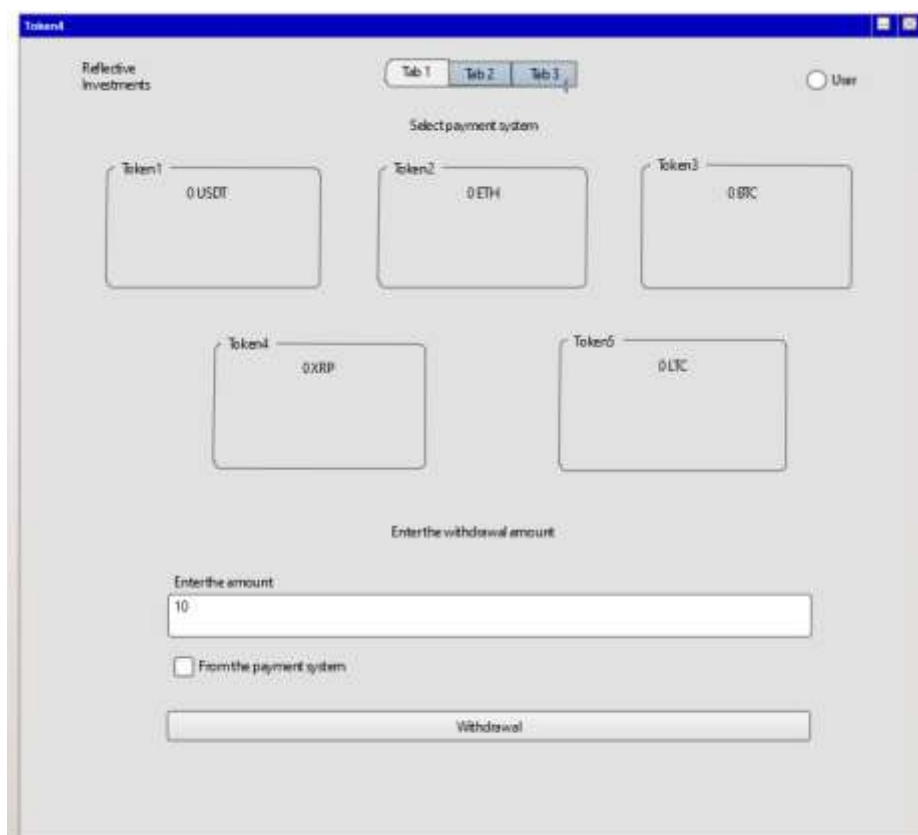


Рисунок 2.40 – Інтерфейс сторінки виведення коштів

РОЗДІЛ 3

ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1 Інформаційне забезпечення

Microsoft SQL Server Management Studio (SSMS) – інтегроване середовище розробки та управління базами даних Microsoft SQL Server[20]. Це безкоштовне програмне забезпечення, яке дозволяє розробникам та адміністраторам баз даних здійснювати широкий спектр операцій, таких як:

- Створення та зміна баз даних, таблиць, схем, індексів та інших об'єктів баз даних;
- Виконання SQL-запитів та зберігання їх у вигляді скриптів;
- Управління безпекою та авторизацією, включаючи налаштування прав доступу до об'єктів баз даних та аутентифікацію користувачів;
- Моніторинг та профілювання роботи баз даних, включаючи аналіз виконання запитів, процедур та інших операцій;
- Резервне копіювання та відновлення баз даних;
- Розгортання баз даних на веб-серверах та інших серверах, що підтримують SQL Server.

SSMS має різні інтерфейси користувача, включаючи графічний інтерфейс користувача (GUI) та редактор коду SQL. Інтерфейс користувача дозволяє користувачам працювати з базами даних, використовуючи графічний інтерфейс. Редактор коду SQL дозволяє розробникам створювати скрипти SQL та інші кодові об'єкти.

Усі функції SSMS можуть бути виконані через графічний інтерфейс або за допомогою команд SQL, що дозволяє розробникам та адміністраторам баз даних вибрати той спосіб роботи з програмою, який їм зручний.

Ось декілька обґрунтувань, чому варто вибрати Microsoft SQL Server Management Studio:

- Microsoft SQL Server Management Studio забезпечує високий рівень надійності та стабільності під час роботи з базами даних. Вона дозволяє

ефективно виконувати завдання з керування та підтримки бази даних, забезпечуючи надійний захист даних.

- SSMS має велику кількість функцій та інструментів для керування базами даних, які можуть ефективно працювати з даними. Наприклад, вона має різні інструменти огляду для створення та зміни таблиці, процедури зберігання та функції, інструменти для створення запитів, забезпечення безпеки баз даних тощо.

- SSMS має простий та зрозумілий інтерфейс, який дозволяє легко працювати з базами даних. Це дозволяє користувачам швидко освоювати функції та інструменти, що скорочує час на навантаження та управління базами даних.

- Microsoft SQL Server Management Studio має широкую спільну кількість користувачів і забезпечує високий рівень підтримки. Багато користувачів можуть знайти відповіді на свої запитання через різні джерела, такі як документація, форуми та спільноти.

Порівняємо між собою Microsoft SQL Server Management Studio та MySQL Workbench.

Microsoft SQL Server Management Studio (SSMS) та MySQL - це дві різні системи керування базами даних (СКБД). Обидві системи можуть використовуватися для зберігання та управління даними, але вони мають свої особливості та відмінності.

Основні відмінності між SSMS та MySQL наступні:

- SSMS розробляється та підтримується компанією Microsoft, тоді як MySQL розробляється та підтримується компанією Oracle. Крім того, MySQL є відкритим програмним забезпеченням, тоді як SSMS вимагає ліцензію для використання.

- SSMS підтримує тільки бази даних SQL Server, тоді як MySQL може працювати з різними типами баз даних, такими як MySQL, MariaDB, Percona та інші.

– SSMS надає широкі можливості для роботи з хмарними службами, такими як Azure SQL Database, Azure Data Lake Storage, Azure Stream Analytics тощо. MySQL також може працювати в хмарі, але не має таких широких можливостей для інтеграції з хмарними службами.

– SSMS може працювати з дуже великими базами даних та має більш продуману систему масштабування, ніж MySQL.

– SSMS має більш розвинений та зручний інтерфейс користувача, ніж MySQL, що дозволяє більш ефективно працювати з базами даних.

MySQL може працювати швидше, ніж SSMS, зокрема на великих обсягах даних, завдяки своїй оптимізації та широкому спектру можливостей налаштування.

Після визначення з базою даних необхідно було створити модель бази даних яка продемонстрована на рисунку 3.1.

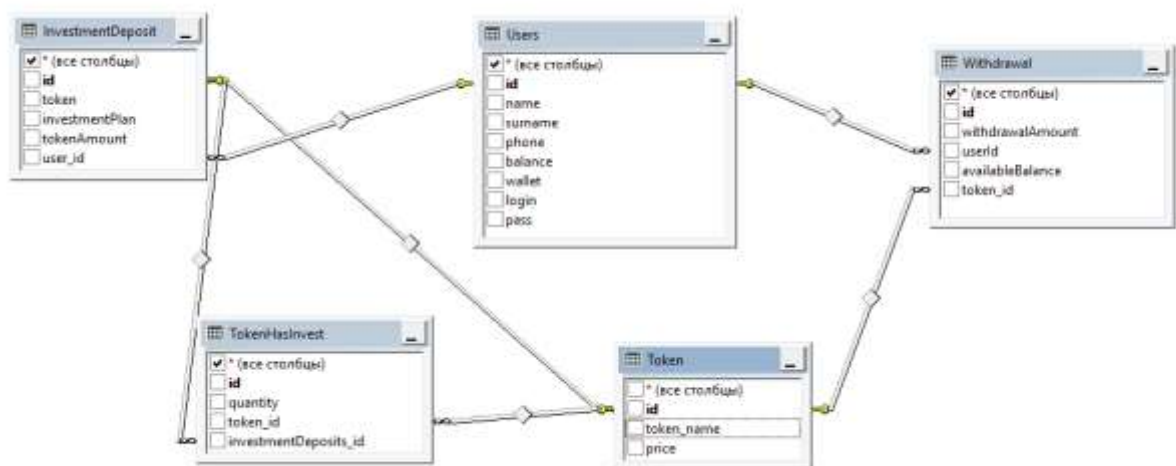


Рисунок 3.1 – Схема бази даних

На основі цієї схеми було створено базу даних в програмі SSMS. Тут є усі необхідні колекції. Деякі з наведених колекцій є вкладеними тобто є частиною інших. Отриману базу даних наведено на рисунку 3.2.

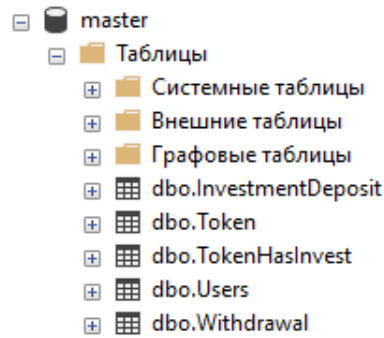


Рисунок 3.2 – База даних побудована в SSMS.

Для кодування даних в базі даних використовуються криптографічні ключі[31].

Результат перетворення ER-моделі в конкретну схему бази даних на основі СУБД MSSQL та вміст файлу згенерованої моделі даних представлено у «Додатку Г».

Далі будуть описані інформаційні масиви даних, вони представлені у таблицях 3.1-3.5.

Таблиця 3.1 – Опис інформаційного об'єкту «InvestmentDeposit»

Назва атрибута	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індекс не поле
id	I	так	Має право змінювати лише власник	ПК	так	—	ІНД
token	V(255)	так			так	—	ІДД
investment Plan	V(255)	так			так	—	ІДД
tokenAmount	F	так			так	—	ІДД
user_id	I	так		ВК	—	—	ІНД

Таблиця 3.2 – Опис інформаційного об'єкту «Token»

Назва атрибута	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індекс не поле
id	I	так	Має право змінювати адміністратор	ПК	—	—	ІНД
token_name	V(255)	так			так	—	ІДД
Address	V(255)	так			так	—	ІДД
price	F	так			так	—	ІДД

Таблиця 3.3 – Опис інформаційного об'єкту «TokenHasInvests»

Назва атрибута	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індексне поле
id	I	так	Має право змінювати лише власник	ПК	так	—	ІНД
quantity	i	так			так	—	ІДД
token_id	I	так		ВК	—	—	ІНД
investmentDeposit_id	I	так		ВК	—	—	ІНД

Таблиця 3.4 – Опис інформаційного об'єкту «Users»

Назва атрибута	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індексне поле
id	I	так	Має право змінювати адміністратор	ПК	—	—	ІНД
name	V(255)	так			так	—	ІДД
surname	V(255)	так			так	—	ІДД
phone	i(10)	так			так	—	ІДД
balance	i	так			так	—	ІДД
wallet	V(255)	так			так	—	ІДД
login	V(255)	так			так	—	ІДД
pass	V(255)	так			так	—	ІДД

Таблиця 3.5 – Опис інформаційного об'єкту «Withdrawal»

Назва атрибута	Формат	Обов'язковий (необов'язковий)	Обмеження на право звертання	Первинний (вторинний) ключ	Виводимість значень	Дублювання значень, %	Індексне поле
id	I	так	Має право змінювати лише власник	ПК	так	—	ІНД
withdrawal Amount	F	так			так	—	ІДД
availableBalance	F	так			так	—	ІДД
user_id	I	так		ВК	—	—	ІНД
token_id	I	так		ВК	—	—	ІНД

Специфікація даних інформаційної системи «Криптоінвестиції» представлена на рисунках 3.3 – 3.7.

	Имя столбца	Тип данных	Разрешить ...
🔑	id	bigint	☐
	token	nvarchar(255)	☒
	investmentPlan	nvarchar(255)	☒
	tokenAmount	float	☒
	user_id	bigint	☒
			☐

Рисунок 3.3 – Специфікація таблиці «InvestmentDeposit»

	Имя столбца	Тип данных	Разрешить ...
🔑	id	bigint	☐
	token_name	nvarchar(255)	☒
	price	float	☒
			☐

Рисунок 3.4 – Специфікація таблиці «Token»

	Имя столбца	Тип данных	Разрешить ...
🔑	id	bigint	☐
	quantity	int	☒
	token_id	bigint	☒
	investmentDeposits_id	bigint	☒
			☐

Рисунок 3.5 – Специфікація таблиці «TokenHasInvests»

	Имя столбца	Тип данных	Разрешить ...
🔑	id	bigint	☐
	name	nvarchar(255)	☒
	surname	nvarchar(255)	☒
	phone	int	☒
	balance	int	☒
	wallet	bit	☒
	login	nvarchar(255)	☒
	pass	nvarchar(255)	☒
			☐

Рисунок 3.6 – Специфікація таблиці «Users»

	Имя столбца	Тип данных	Разрешить ...
🔑	id	bigint	☐
	withdrawalAmount	float	☒
	userId	bigint	☒
	availableBalance	float	☒
	token_id	bigint	☒
			☐

Рисунок 3.7 – Специфікація таблиці «Withdrawal»

3.2 Організаційне забезпечення

Далі продемонстрована діаграма станів. На цій діаграмі вказана поведінка інформаційної системи у випадках зареєстрованого користувача та гостя. Якщо клієнт не зареєструвався, йому не буде доступна функція депозиту і стейкінгу токенів, ці функції доступні лише з особистого кабінету. Діаграму станів можна побачити на рисунку 3.8.

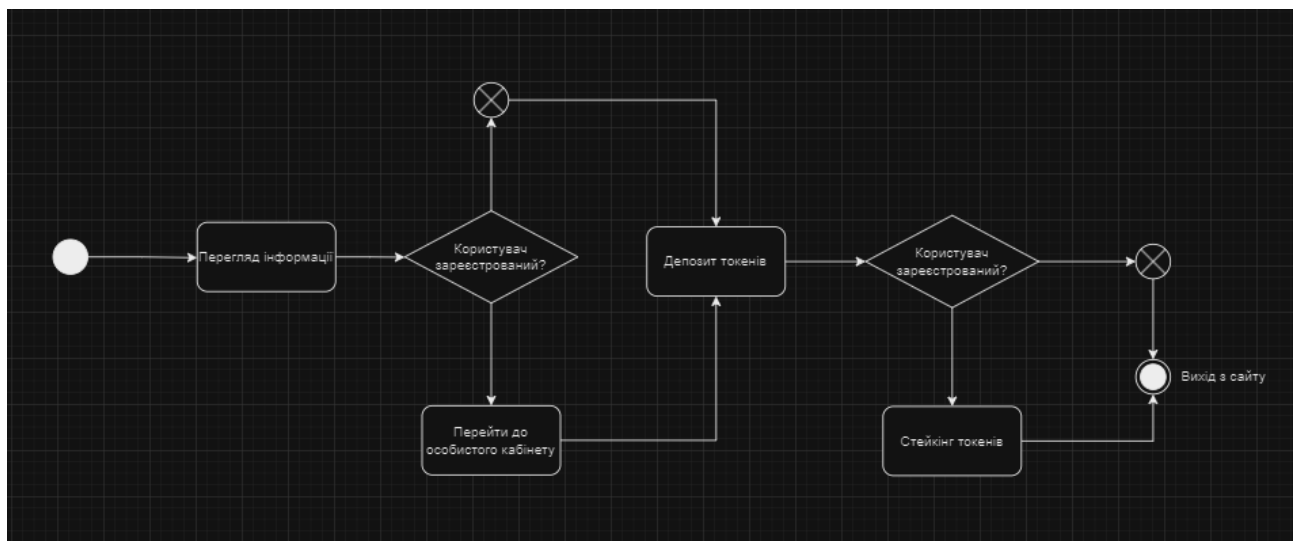


Рисунок 3.8 – Діаграма станів

На рисунку 3.9 відображається процес відправки заявки на розгляд адміністрації – скарги, побажання та запитання, а саме діаграма дій. Якщо користувач авторизований у системі, він може перейти на сторінку «Tickets» з особистого кабінету, в іншому випадку необхідно це зробити з головної сторінки. Після цього користувач може продовжити займатися своїми справами чи вийти з сайту, паралельно з тим, як адміністрація обробляє запит.

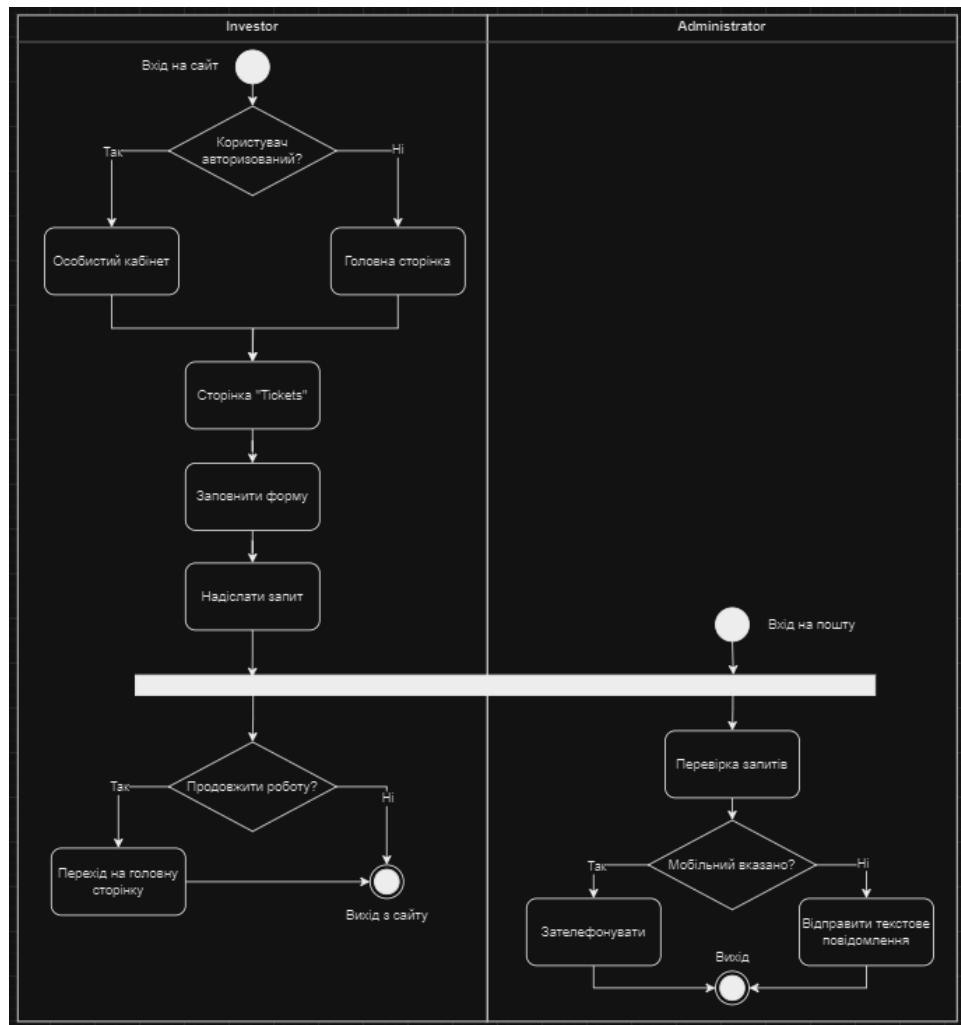


Рисунок 3.9 – Діаграма дії

3.3 Програмне забезпечення

Далі буде розглянуто програмне забезпечення, яке було використано для розробки системи, включаючи середовища розробки, бібліотеки, фреймворки та інші інструменти.

Основне середовище розробки для створення коду системи це IntelliJ IDEA, воно було вибрано через його підтримку широкого спектра мов програмування, потужні інструменти для рефакторингу, інтеграцію з системами контролю версій, такими як Git, а також зручний і інтуїтивно зрозумілий інтерфейс. Використання IntelliJ IDEA забезпечує високу продуктивність розробки завдяки автодоповненню коду, інтегрованим інструментам для тестування та налагодження. Visual Studio Code – середовище розробки, яке використовувалося для створення фронтенд-

частини системи. Він підтримує розширення для роботи з HTML, CSS, JavaScript та іншими веб-технологіями, що робить його ідеальним для розробки інтерфейсу користувача. Структуру розроблених елементів front-end частини можна побачити на рисунку 3.10.

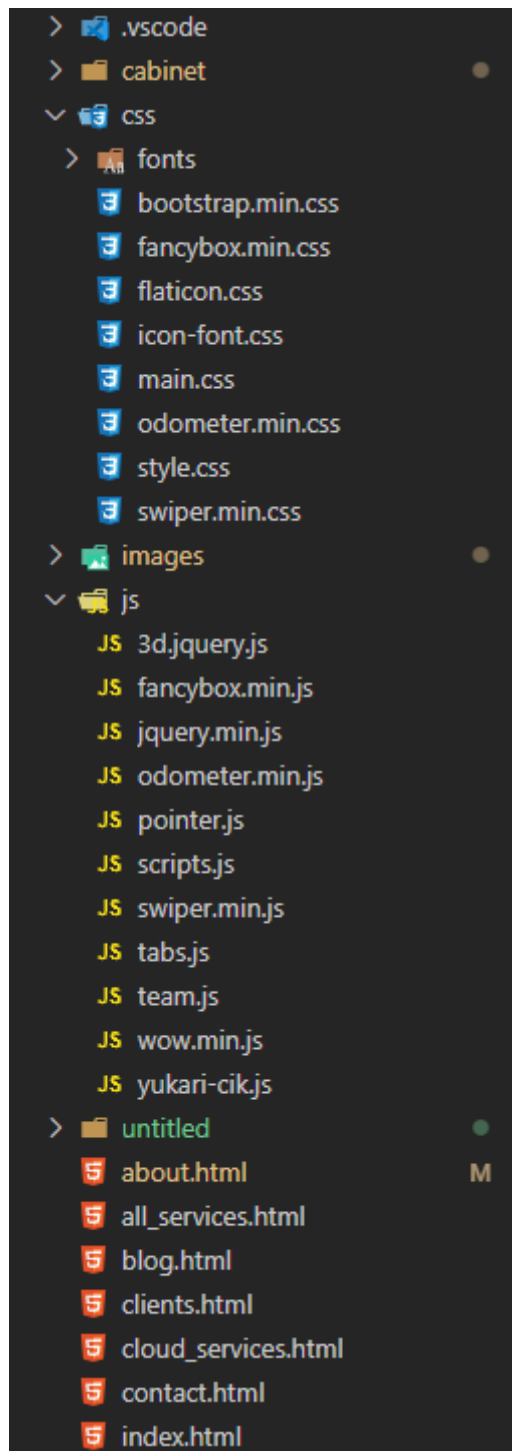


Рисунок 3.10 – Створені файли для front-end частини

До основних фреймворків можна віднести Spring, він використовувався для розробки бекенду системи. Він забезпечує потужні інструменти для

розробки веб-додатків, управління транзакціями, інтеграції з базами даних, а також підтримку безпеки. Використання Spring дозволяє швидко розробляти, тестувати та розгортати додатки, забезпечуючи високу продуктивність та масштабованість системи. Hibernate – ORM (Object-Relational Mapping) фреймворк, який використовується для роботи з базою даних. Hibernate спрощує процес збереження, оновлення та видалення об'єктів в базі даних, забезпечуючи гнучкість та ефективність управління даними.

Microsoft SQL Server – основна система управління базами даних (СУБД), використана в проєкті. SQL Server забезпечує надійне та ефективне управління даними, підтримку транзакцій, забезпечення безпеки та масштабованість. Використання SQL Server дозволяє зберігати великі обсяги даних та забезпечувати високу продуктивність системи.

На рисунку 3.11 зображена діаграма компонентів системи для процесу виводу коштів зі свого аккаунту. Сервер приймає запит на виведення коштів та відправляє новий запит на передачу токенів. Після усіх підтверджень, токени відправляються користувачеві та в базу даних записується результат., враховуючи взаємодію між компонентами та їхнє вбудоване розташування на тлі Spring Boot[24], може бути описана наступним чином:

- User. Ініціатор процесу, який хоче вивести токени.
- Token. Представляє конкретну криптовалюту, яку користувач вибирає для виведення.
- WithdrawalTokens. Це початкова точка сценарію взаємодії, яка може бути активована користувачем для початка процесу виведення.
- ChooseToken. Етап, на якому користувач вибирає специфічний токен для виведення.
- GetBalance. Компонент системи, який відповідає за отримання інформації про баланс токенів на рахунку користувача.
- Withdrawal. Компонент, який обробляє сам процес виведення, включаючи перевірку можливості виведення та виконання транзакції.

- PostReq. Обробляє запити від користувача на виведення коштів та комунікацію з внутрішніми або зовнішніми службами для реалізації цього запиту.
- SaveRes. Компонент, призначений для збереження результатів операції у базі даних системи.
- Amount. Величина, яку користувач хоче вивести, відіграє ключову роль у процесі виведення.
- SendTokens. Кінцева дія у сценарії, де токени відправляються користувачу або на зовнішній гаманець.

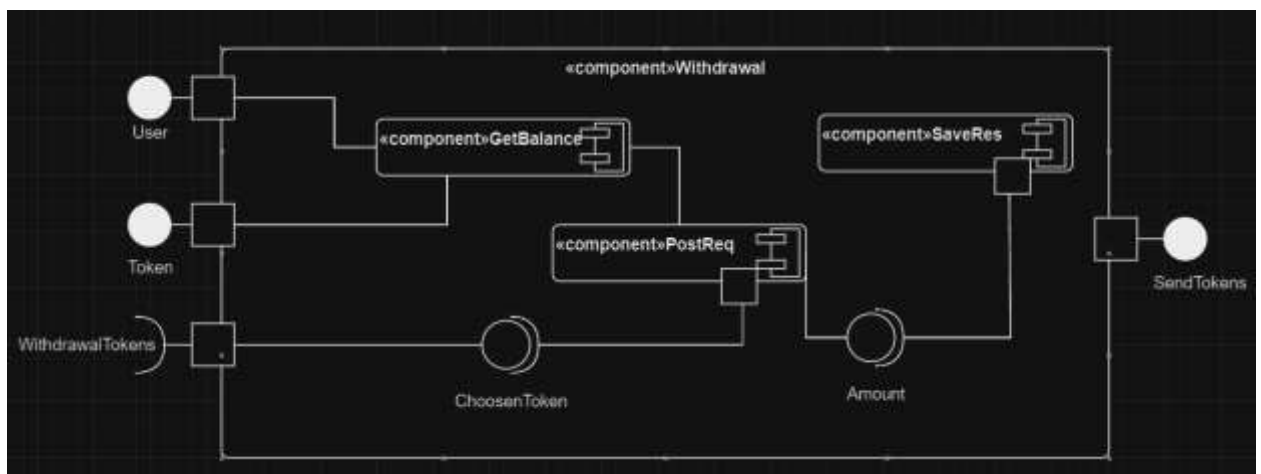


Рисунок 3.11 – Діаграма компонентів, що використовуються при вході в акаунт

3.4 Технічне забезпечення

Зображена діаграма розгортання на рисунку 3.12, ілюструє архітектуру інформаційної системи, розділену на фронтенд, бекенд та серверні компоненти з базою даних.

Фронтенд вказує на клієнтську частину системи, яка взаємодіє безпосередньо з користувачем. Зазвичай включає HTML[21], CSS[22] та JavaScript[23], що забезпечують структуру, стилізацію та інтерактивність веб-інтерфейсу. Цей шар відповідальний за представлення інформації та збір вхідних даних від користувачів.

Бекенд містить логіку програми, яка виконується на сервері, включаючи такі компоненти як "functional.param", що відповідальна за обробку параметрів функцій, та "price.parser", який, займається аналізом цінових даних, отриманих з зовнішніх джерел.

Для розміщення бекенду та бази даних системи використовується серверне обладнання, яке забезпечує високу продуктивність та надійність. Сервер підтримують багатоядерні процесори, значні обсяги оперативної пам'яті та високошвидкісні накопичувачі. Він відповідає за обробку запитів від фронтенду, виконання логіки бекенду та взаємодію з базою даних.

База Даних – централізоване сховище даних, де зберігаються всі дані, необхідні для роботи системи. База даних приймає запити на збереження та видачу даних від сервера, забезпечуючи постійність та цілісність інформації.

Для розробки використовувався робочий комп'ютер, який забезпечує швидкодію завдяки багатоядерному процесору та достатньому обсягу оперативної пам'яті. Система має такі характеристики:

- Процесор Intel(R) Xeon(R) CPU E5-2660 V3 3,10GHz;
- Оперативна пам'ять – 16 Гб;
- Накопичувач – M2 SSD 128 Гб, SATA SSD 1Тб, HDD 1 Тб;
- Екрани – IPS 25", IPS 27".
- Відеокарта – RTX 2060 Super.

Мережеве обладнання забезпечує швидкісне з'єднання між серверами, системами зберігання даних та комп'ютером. Це обладнання гарантує високу пропускну здатність та безпеку мережі.

Лінії з пунктиром на діаграмі показують взаємодію між різними компонентами системи. Фронтенд відправляє запити до бекенду, який, у свою чергу, взаємодіє з сервером та базою даних для виконання цих запитів та повернення результатів.

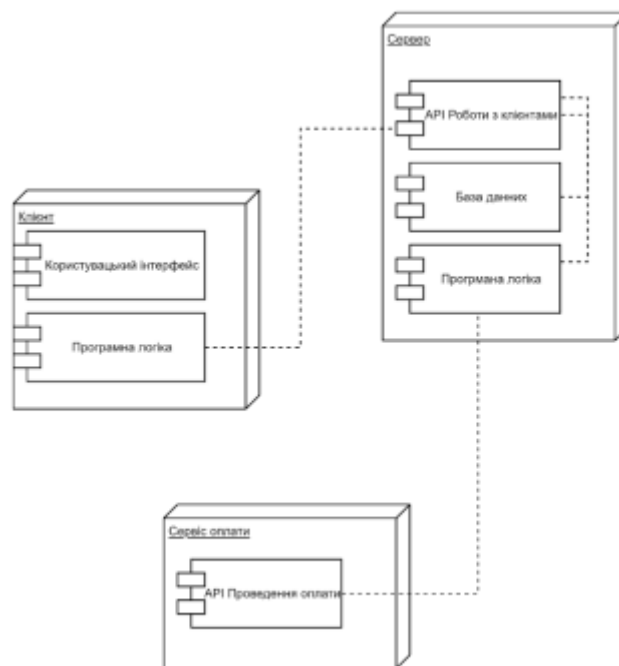


Рисунок 3.12 – Діаграма розгортання

3.5 Результати реалізації інформаційної системи

Функціональний та зручний веб-інтерфейс реалізовано з інтуїтивно зрозумілим дизайном, чітким розподілом на сторінки і розділи (головна, про нас, послуги, клієнти, контакти). Використано адаптивну верстку для коректного відображення на різних пристроях та сучасні підходи до веб-дизайну та юзабіліті.

Система користувачів та особистий кабінет передбачає можливість реєстрації нових користувачів, входу в особистий кабінет, відновлення доступу через скидання пароля. В особистому кабінеті користувачі можуть переглядати статистику доходів, історію операцій, керувати налаштуваннями та додавати адреси гаманців.

Інвестиційний функціонал дозволяє вибирати один з запропонованих інвестиційних планів, встановлювати розмір інвестиційного внеску, виводити кошти на додані гаманці з відображенням історії операцій.

Реалізована реферальна програма для залучення нових інвесторів.

Модуль зворотного зв'язку дає можливість надсилати скарги, відгуки та пропозиції від користувачів.

На сайті представлена інформація про компанію, перелік послуг, клієнтська база.

Була створена база даних, під'єднана до системи, написано бекенд з відповідними класами, а фронтенд частину реалізовано з використанням HTML, CSS та JavaScript.

Використано сучасні технології та підходи до розробки для забезпечення високої продуктивності, безпеки та масштабованості системи[32].

РОЗДІЛ 4

ТЕСТУВАННЯ І ВЕРИФІКАЦІЯ ПРОЕКТНИХ РІШЕНЬ

4.1 Розробка тест-кейсів

Документація, яка супроводжує процес тестування системи, містить в собі ретельні описи різноманітних сценаріїв, які охоплюють різні аспекти її функціональності, продуктивності, безпеки та інших параметрів. Це допомагає забезпечити повне покриття системи тестами та виявити потенційні проблеми або недоліки ще на етапі її проектування[25].

Характеристика кожного виду тест-кейсів та їх використання у контексті проектування інформаційної системи для криптоінвестицій:

1. Модульні тест-кейси призначені для тестування окремих модулів або компонентів системи на рівні коду. Вони допомагають виявити дефекти та помилки в реалізації функцій, перевіряючи їх поведінку в ізольованому середовищі. Модульні тест-кейси забезпечують покриття коду тестами та полегшують виявлення та усунення помилок на ранніх етапах розробки.

2. Інтеграційні тест-кейси використовуються для перевірки взаємодії між різними модулями та компонентами системи. Вони допомагають виявити проблеми інтеграції, некоректну передачу даних між компонентами та порушення бізнес-логіки в комплексних сценаріях використання. Інтеграційні тест-кейси забезпечують, що різні частини системи працюють разом належним чином.

3. Приймальні тест-кейси імітують реальні сценарії використання системи користувачами та призначені для перевірки відповідності системи встановленим вимогам. Вони охоплюють основні функції та можливості системи в тестовому середовищі, максимально наближеному до продуктивного. Приймальні тест-кейси допомагають виявити функціональні недоліки, проблеми з інтерфейсом або невідповідності вимогам перед випуском системи в продуктивне середовище.

4. Тест-кейси сценаріїв призначені для перевірки повних потоків роботи системи, що охоплюють кілька функцій або модулів. Вони імітують реальний шлях користувача в системі, перевіряючи взаємодію між різними компонентами та цілісність потоку даних. Тест-кейси сценаріїв допомагають виявити порушення бізнес-логіки та проблеми, пов'язані з комплексною взаємодією різних частин системи.

5. Негативні тест-кейси спеціально розроблені для перевірки поведінки системи в виняткових ситуаціях або при введенні некоректних даних. Вони допомагають переконатися, що система правильно обробляє помилки, відхиляє невалідні вхідні дані та захищена від небажаних або шкідливих дій користувачів. Негативні тест-кейси сприяють підвищенню стійкості та безпеки системи.

Використання різних видів тест-кейсів забезпечує всебічне тестування інформаційної системи "Криптоінвестиції" на різних рівнях та в різних сценаріях використання. Це допомагає виявити та усунути дефекти, перевірити відповідність системи вимогам, забезпечити її надійність, безпеку та зручність використання для користувачів[26].

У «Додатку А» наведено тестова документація, підготовлена засобами Enterprise Architect.

4.2 Трасування вимог до системи

Трасування вимог забезпечує зв'язок між вимогами до системи та її компонентами, тестами та іншими артефактами на кожному етапі життєвого циклу розробки програмного забезпечення[6]. Реалізовану діаграму можна побачити на рисунку 4.1. Відповідно матриці трасування відображені на рисунках 4.2-4.3.

Трасування дозволяє відстежувати походження кожної вимоги, її реалізацію в коді, тестові сценарії та випадки використання. Це допомагає гарантувати, що всі вимоги були врахованими і правильно реалізованими. Відсутність належного трасування може призвести до пропущених або

неправильно реалізованих вимог, що, в свою чергу, може спричинити дефекти в системі.

Крім того, трасування вимог полегшує внесення змін та управління вимогами в процесі розробки та супроводу системи. Воно дозволяє визначити, які компоненти та тести можуть бути залучені при зміні певної вимоги, що допомагає зменшити ризики та спростити процес вдосконалення системи.

Трасування також є корисним для забезпечення відповідності нормативним вимогам та стандартам якості у сфері фінансових систем та криптовалют. Воно надає докази того, що всі необхідні вимоги були враховані та реалізовані належним чином.

Для ефективного трасування вимог можуть використовуватися спеціальні інструменти управління вимогами та інші програмні рішення. Вони допомагають організувати, відстежувати та візуалізувати зв'язки між вимогами, компонентами системи та іншими артефактами протягом усього життєвого циклу розробки.

На діаграмі трасування вимог для інформаційної системи використовуються три основні зв'язки.

Зв'язок Trace забезпечує прослідковуваність між вимогами та іншими артефактами системи, такими як компоненти, тестові випадки та випадки використання. Він дозволяє відстежувати походження кожної вимоги та її реалізацію в системі.

Зв'язок Abstraction встановлюється між вимогами різних рівнів абстракції, коли одна вимога є деталізацією або конкретизацією іншої, більш загальної вимоги. Він допомагає організувати вимоги в ієрархічну структуру та забезпечує їх узгодженість.

Зв'язок Realization встановлюється між вимогою та елементами системи (компонентами, модулями, класами тощо), які безпосередньо реалізують цю вимогу. Він показує, які частини системи відповідають за задоволення певних вимог.

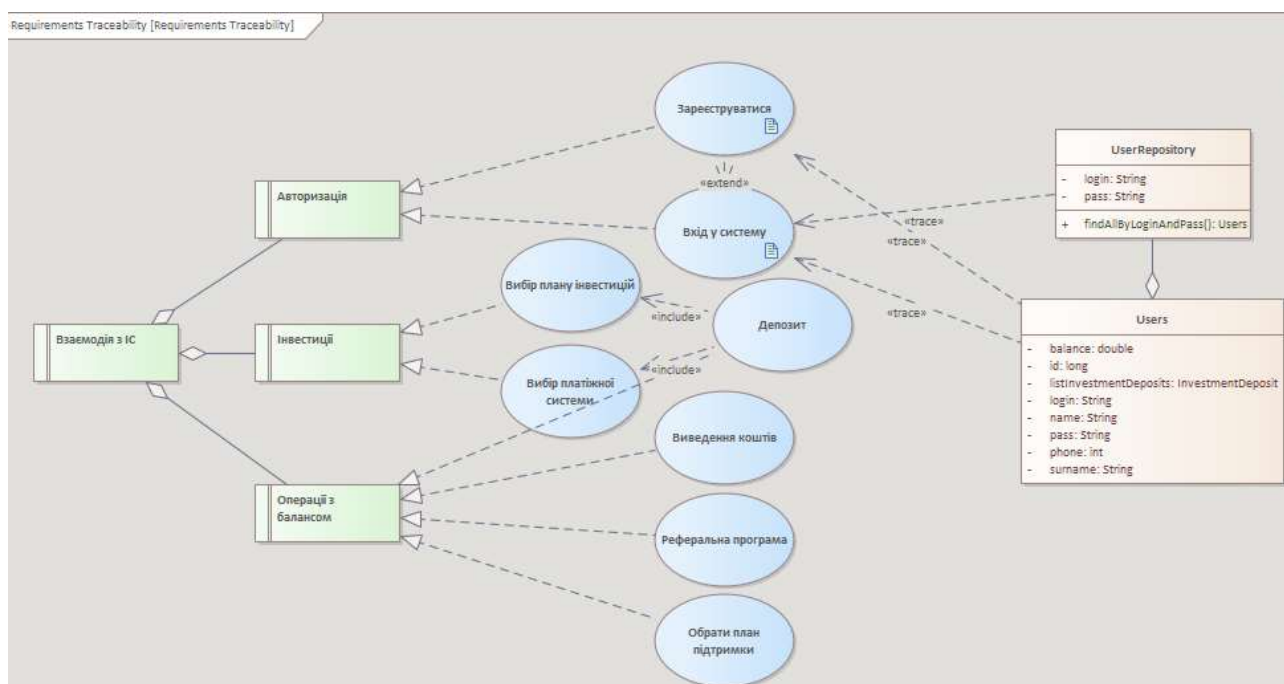


Рисунок 4.1 – Діаграма трасування в системі

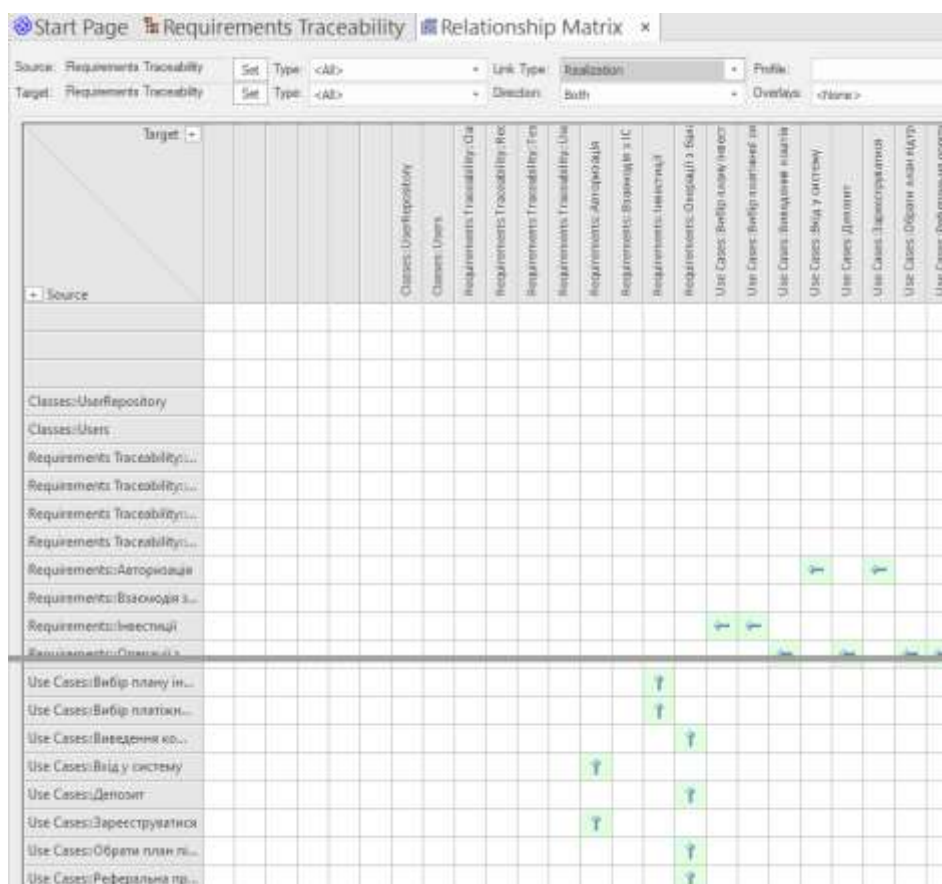


Рисунок 4.2 – Матриця зв'язків «Realization»

ВИСНОВКИ

Даний бакалаврський проект був присвячений комплексному дослідженню та розробці інформаційної системи для криптоінвестицій, що відповідає потребам сучасного ринку криптовалют. В рамках проекту було здійснено глибокий аналіз предметної області, виявлено потреби інвесторів, а також проведено детальне дослідження існуючих рішень на ринку.

На першому етапі проекту було проведено аналіз предметної області, що включав в себе дослідження особливостей криптовалютного ринку, його тенденцій та викликів.

З метою визначення оптимальних рішень для проектованої системи було проведено детальне дослідження існуючих програмних продуктів для криптоінвестицій, таких як Binance, Coinbase, Huobi та KuCoin. Аналіз їх функціональних можливостей, переваг та недоліків дозволив виявити кращі рішення для проектування та вдосконалення.

На основі проведеного аналізу було розроблено інформаційну систему, яка поєднує в собі ключові функції, необхідні для ефективного управління криптоінвестиціями. Вона включає в себе:

- Систему реєстрації та аутентифікації користувачів, що забезпечує безпечний доступ до особистих кабінетів інвесторів.
- Особистий кабінет інвестора з можливістю перегляду балансів, історії транзакцій, поточного стану інвестицій та здійснення операцій з криптовалютними активами.
- Інтеграцію з сервісами обміну криптовалют, що забезпечує зручне поповнення рахунків та виведення коштів.
- Реферальну програму, що стимулює залучення нових користувачів.
- Модуль зворотного зв'язку, що дозволяє користувачам надсилати скарги, відгуки та пропозиції щодо роботи системи.

Для забезпечення ефективної розробки системи було визначено функціональні та нефункціональні вимоги. Функціональні вимоги описують основні можливості системи, тоді як нефункціональні вимоги визначають критерії її функціонування, такі як продуктивність, безпека, зручність використання та масштабованість.

Для візуалізації ключових процесів, об'єктів та їх взаємозв'язків в системі було використано мову моделювання OPM (Object-Process Methodology). З її допомогою було створено ієрархію діаграм, які описують функціонал системи та взаємодію її компонентів. Також було розроблено алгоритм розв'язання задачі, що описує послідовність дій для виконання основних операцій в системі.

Було обрано архітектурний шаблон MVC (Model-View-Controller), який забезпечує чітке розділення відповідальностей між компонентами системи. Model відповідає за дані та бізнес-логіку, View – за візуалізацію даних для користувача, а Controller – за обробку запитів та взаємодію між моделлю та представленням.

Для збереження даних було обрано реляційну базу даних Microsoft SQL Server, яка є однією з найпопулярніших та надійних СУБД. Для побудови схеми бази даних було використано СКБД MySQL Workbench, що дозволило візуалізувати структуру даних та їх взаємозв'язки.

Щоб продемонструвати користувацькі інтерфейси було розроблено прототип інформаційної системи в онлайн-сервісі Figma. Прототип включає в себе дизайн головної сторінки, особистого кабінету інвестора, сторінок реєстрації, авторизації, поповнення рахунку, виведення коштів, історії транзакцій, реферальної програми та інших розділів.

Для реалізації функціоналу системи було використано сучасні технології та підходи до розробки веб-додатків, такі як Spring Boot, HTML, CSS та JavaScript. Використання цих технологій забезпечило високу продуктивність, безпеку та масштабованість системи.

З метою забезпечення якості та надійності системи було розроблено комплекс тест-кейсів, які охоплюють всі основні функціональні можливості системи. Тестування проводилося на різних етапах розробки, що дозволило виявити та усунути потенційні дефекти та помилки.

Трасування вимог до системи забезпечило їх повну реалізацію та відповідність потребам користувачів. В результаті проведених робіт було створено функціональну та зручну інформаційну систему для криптоінвестицій, яка відповідає сучасним вимогам безпеки, масштабованості та зручності використання.

У подальшому розроблену систему можна вдосконалювати шляхом розширення її функціональних можливостей, інтеграції з іншими платформами та сервісами, а також впровадження нових інвестиційних інструментів та стратегій[33].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

- 1) Накамото С. "Bitcoin: A Peer-to-Peer Electronic Cash System." 2008. [Електронний ресурс]. URL: <https://bitcoin.org/bitcoin.pdf> (дата звернення: 03.04.2024).
- 2) IntelliJ IDEA [Електронний ресурс] – URL: https://uk.wikipedia.org/wiki/IntelliJ_IDEA (дата звернення: 17.05.2024).
- 3) Enterprise Architect User Guide [Електронний ресурс] – URL: <https://sparxsystems.com/bin/EAUserGuide.pdf> (дата звернення: 17.05.2024).
- 4) Євген Брітвін. "Криптовалюти: як це працює?" Харків: Фоліо, 2019. – 320 с.
- 5) "Криптовалюти: ризики та можливості для фінансового ринку України." Аналітичний звіт Національного банку України. 2021. [Електронний ресурс]. URL: <https://bank.gov.ua> (дата звернення: 17.05.2024).
- 6) Traceability Diagram [Електронний ресурс] – URL: <https://sparxsystems.com/resources/gallery/diagrams/software/sw-traceability-diagram.html> (дата звернення: 17.05.2024).
- 7) Онлайн-програмне забезпечення для створення блок-схем [Електронний ресурс] / Draw.IO – URL: <https://app.diagrams.net/> (дата звернення: 17.05.2024).
- 8) Antonopoulos, A. M. Mastering Bitcoin: Programming the Open Blockchain / A. M. Antonopoulos. – [Б. м.]: O'Reilly Media, Inc., 2017.
- 9) Schwab, K. The Fourth Industrial Revolution / K. Schwab. – Нью Йорк: Crown Business, 2016.
- 10) Tapscott, D. Blockchain Revolution: How the Technology Behind Bitcoin Is Changing Money, Business, and the World / D. Tapscott, A. Tapscott. – Нью Йорк: Penguin, 2016.
- 11) Binance Academy. What is Binance? [Електронний ресурс] / Binance Academy. – URL: <https://academy.binance.com/en/articles/what-is-binance> (дата звернення: 17.05.2024).

- 12) Coinbase. About Coinbase [Электронный ресурс] / Coinbase. – URL: <https://www.coinbase.com/about> (дата звернения: 17.05.2024).
- 13) Huobi Global. About Huobi Global [Электронный ресурс] / Huobi Global. – URL: <https://www.huobi.com/en-us/about/> (дата звернения: 17.05.2024).
- 14) KuCoin. About KuCoin [Электронный ресурс] / KuCoin. – URL: <https://www.kucoin.com/about> (дата звернения: 17.05.2024).
- 15) Gamma, E. Design Patterns: Elements of Reusable Object-Oriented Software / E. Gamma. – [Б. м.]: Addison-Wesley Professional, 2019.
- 16) Fowler, M. Patterns of Enterprise Application Architecture / M. Fowler. – [Б. м.]: Addison-Wesley Professional, 2003.
- 17) Freeman, E. Head First Design Patterns / E. Freeman, E. Robson, B. Bates, K. Sierra. – [Б. м.]: O'Reilly Media, Inc., 2004.
- 18) Figma. Figma: the collaborative interface design tool [Электронный ресурс] / Figma. – URL: <https://www.figma.com/> (дата звернения: 17.05.2024).
- 19) MySQL. MySQL: The world's most popular open source database [Электронный ресурс] / MySQL. – URL: <https://www.mysql.com/> (дата звернения: 17.05.2024).
- 20) Microsoft. Download SQL Server Management Studio (SSMS) [Электронный ресурс] / Microsoft. – URL: <https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16> (дата звернения: 17.05.2024).
- 21) W3Schools. HTML Tutorial [Электронный ресурс] / W3Schools. – URL: <https://www.w3schools.com/html/> (дата звернения: 17.05.2024).
- 22) W3Schools. CSS Tutorial [Электронный ресурс] / W3Schools. – URL: <https://www.w3schools.com/css/> (дата звернения: 17.05.2024).
- 23) Mozilla Developer Network. JavaScript [Электронный ресурс] / Mozilla Developer Network. – URL: <https://developer.mozilla.org/en-US/docs/Web/JavaScript> (дата звернения: 17.05.2024).
- 24) Spring. Spring Framework [Электронный ресурс] / Spring. – URL: <https://spring.io/> (дата звернения: 17.05.2024).

25) Guru99. Software Testing Tutorial: Free QA Course [Электронный ресурс] / Guru99. – URL: <https://www.guru99.com/software-testing.html> (дата звернения: 17.05.2024).

26) Software Testing Help. Software Testing Tutorial - Complete Guide [Электронный ресурс] / Software Testing Help. – URL: <https://www.softwaretestinghelp.com/software-testing-tutorial/> (дата звернения: 17.05.2024).

27) Investopedia. Cryptocurrency [Электронный ресурс] / Investopedia. – URL: <https://www.investopedia.com/terms/c/cryptocurrency.asp> (дата звернения: 17.05.2024).

28) TradingView. Financial Charts, Data & Analytics [Электронный ресурс] / TradingView. – URL: <https://www.tradingview.com/> (дата звернения: 17.05.2024).

29) DZone. What is a REST API? [Электронный ресурс] / DZone. – URL: <https://dzone.com/articles/what-is-a-rest-api> (дата звернения: 17.05.2024).

30) TechTarget. What is Two-Factor Authentication (2FA)? [Электронный ресурс] / TechTarget. – URL: <https://www.techtarget.com/searchsecurity/definition/two-factor-authentication> (дата звернения: 17.05.2024).

31) OWASP Foundation. OWASP Top Ten [Электронный ресурс] / OWASP Foundation. – URL: <https://owasp.org/www-project-top-ten/> (дата звернения: 17.05.2024).

32) National Institute of Standards and Technology. Cybersecurity Framework [Электронный ресурс] / National Institute of Standards and Technology. – URL: <https://www.nist.gov/cyberframework> (дата звернения: 17.05.2024).

33) International Organization for Standardization. ISO/IEC 27001 - Information security management [Электронный ресурс] / International Organization for Standardization. – URL: <https://www.iso.org/standard/54534.html> (дата звернения: 17.05.2024).

ДОДАТКИ

ДОДАТОК А

Тестова документація, підготовлена засобами Enterprise Architect

У склад тестової документації включаються плани тестування, спрямовані на оцінку продуктивності та надійності системи. Це дозволяє оцінити, наскільки ефективно система функціонуватиме у реальних умовах експлуатації щодо швидкодії та стабільності.

Раннє виявлення помилок або недоліків завдяки тестуванню допомагає уникнути серйозних проблем на пізніших етапах розробки. Це сприяє зниженню ризиків та витрат на виправлення дефектів, що може значно вплинути на успішність проекту в цілому.

A.1 Системні тести

Розроблені тестові звіти (Test Report) зображені на рисунках A.1.1-A.5.2:

Test Report

1 April, 2024

Зареєструватися

UseCase in package 'Use Cases'

Зареєструватися

Version 1.0 Phase 1.0 Proposed

|snox.jz created on 29.03.2024. Last modified 01.04.2024

TESTING

Standard System test. Реєстрація нового користувача

Description

1. Відкрити сторінку реєстрації.

2. Ввести коректні дані в усі необхідні поля: ім'я, електронну адресу, пароль.

3. Натиснути кнопку "Зареєструватися".

Input

Використовувати дозволено тільки латиницю

Acceptance Criteria

Results

Користувач успішно зареєстрований у системі і переадресований на головну сторінку.

У системі з'являється новий запис про користувача.

01.04.2024 - Pass

Run by: snox.jz

Checked by: snox.jz

[Last run Pass at 01.04.2024 by snox.jz and checked by snox.jz.]

Рисунок A.1.1 – Системний тест-кейс «Реєстрація нового користувача»

TESTING
Standard System test. Перегляд списку доступних криптовалютних активів Description 1. Увійти у систему за допомогою свого облікового запису. 2. Відкрити розділ "Криптовалютні активи". Input Користувач авторизований у системі Acceptance Criteria Results Список доступних криптовалютних активів відображається коректно. Для кожного активу вказана інформація про назву, ціну, кількість доступних токенів. 01.04.2024 - Pass Run by: snox.jz Checked by: snox.jz
[Last run Pass at 01.04.2024 by snox.jz and checked by snox.jz.]

Рисунок А.1.2 – Системний тест-кейс «Перегляд списку доступних криптовалютних активів»

Test Report

1 April, 2024

Вхід у систему

UseCase in package 'Use Cases'

Вхід у систему

Version 1.0 Phase 1.0 Proposed

snox.jz created on 29.03.2024. Last modified 01.04.2024

TESTING
Standard System test. Спроба ввійти з некоректними обліковими даними Description 1. Відкрити сторінку входу в систему. 2. Ввести некоректні дані в поля: електронна адреса та пароль. 3. Натиснути кнопку "Увійти". Input Користувач не авторизований у системі Acceptance Criteria Results Система повідомляє про помилку вводу облікових даних. Користувач не авторизується у системі. 01.04.2024 - Fail Run by: snox.jz Checked by: snox.jz
[Last run Fail at 01.04.2024 by snox.jz and checked by snox.jz.]

Рисунок А.1.3 – Системний тест-кейс «Вхід в систему з некоректними обліковими даними»

A.2 Модульні тести

Розроблені тестові звіти (Test Report):

Test Report	15 April, 2024
Додавання гаманців UseCase in package 'Use Cases' Додавання гаманців Version 1.0 Phase 1.0 Proposed snxx.jz created on 29.03.2024. Last modified 29.03.2024	
TESTING Standard Unit test. Тестування функції додавання крипто-гаманця Description 1. Перейти в налаштування особистого кабінету 2. Виклик функції createWallet() з валідними даними для створення гаманця Bitcoin. 3. Перевірка, що функція повертає успішний результат з доданим об'єктом гаманця. 4. Перевірка, що створений гаманець зберігається в системі та відображається в інтерфейсі користувача. Input Валідні дані для додавання гаманця Bitcoin (назва гаманця, тип криптовалюти). Acceptance Criteria Results При виклику функції з валідними даними гаманець успішно створюється та зберігається в системі [Last run Pass at Invalid DateTime by and checked by .]	

Рисунок А.2.1 – Модульний тест-кейс «Тестування функції створення крипто-гаманця»

A.3 Інтеграційні тести

Розроблені тестові звіти (Test Report):

Test Report	15 April, 2024
Депозит <i>UseCase in package 'Use Cases'</i>	
Version 1.0 Phase 1.0 Proposed snox.jz created on 29.03.2024. Last modified 14.04.2024	
Депозит	
snox.jz created on 29.03.2024. Last modified 14.04.2024	
TESTING	
Standard Integration test. Тестування процесу купівлі криптовалюти Description 1. Ініціювати процес купівлі криптовалюти з валідними даними. 2. Перевірити, що система успішно отримує курс обміну з інтегрованого обмінника. 3. Перевірити, що система правильно розраховує вартість покупки у фіатній валюті. 4. Перевірити, що система успішно передає запит на оплату до платіжного шлюзу. 5. Симулювати успішну оплату через платіжний шлюз. 6. Перевірити, що система успішно поповнила баланс гаманця відповідною кількістю криптовалюти. 7. Перевірити, що система правильно оновила баланс фіатних коштів користувача. Input Валідні дані для купівлі криптовалюти (тип криптовалюти, кількість, платіжні дані). Acceptance Criteria Results При виклику з валідними даними процес купівлі криптовалюти проходить успішно, гаманець поповнюється, а баланс оновлюється. [Last run Not Run at Invalid DateTime by and checked by .]	

Рисунок А.3.1 – Інтеграційний тест-кейс «Тестування процесу купівлі криптовалюти»

A.4 Приймальні тести

Розроблені тестові звіти (Test Report):

Test Report	15 April, 2024
Вибір плану інвестицій	
UseCase in package 'Use Cases'	
Вибір плану інвестицій Version 1.0 Phase 1.0 Proposed snok.jz created on 29.03.2024. Last modified 29.03.2024	
TESTING	
Standard Acceptance test. Тестування процесу інвестування в криптовалюту Description 1. Авторизуватися в системі із тестовим обліковим записом. 2. Перейти до розділу "Deposit" та ознайомитися з доступними планами. 3. Вибрати один з інвестиційних планів та ввести суму для інвестування. 4. Підтвердити інвестицію та оплатити відповідну суму з балансу фіатних коштів. 5. Перевірити, що баланс інвестиційного рахунку оновився відповідно до внесеної суми. 6. Перевірити, що система відображає деталі поточного інвестиційного плану (період, очікуваний прибуток тощо). 7. Почекаати певний період часу (наприклад, 1 тиждень) та перевірити, що система правильно нараховує прибуток від інвестицій. 8. Вивести частину або всю суму інвестиції на фіатний рахунок. 9. Перевірити, що баланси рахунків оновлені коректно. Input 1. Дані для вибору інвестиційного плану (період інвестування, очікуваний прибуток). 2. Сума для інвестування. Acceptance Criteria Results Процес інвестування проходить без збоїв та помилок. Система правильно відображає всю необхідну інформацію на кожному етапі процесу. Нарахування прибутку відбувається згідно з обраним інвестиційним планом. Виведення коштів з інвестиційного рахунку виконується вчасно та коректно.	
[Last run Not Run at Invalid DateTime by and checked by .]	

Рисунок А.4.1 – Приймальний тест-кейс «Тестування процесу інвестування в криптовалюту»

A.5 Тести-сценарії

Розроблені тестові звіти (Test Report):

Test Report

15 April, 2024

Виведення коштів

UseCase in package 'Use Cases'

Виведення коштів

Version 1.0 Phase 1.0 Proposed

snox.jz created on 29.03.2024. Last modified 29.03.2024

TESTING

Basic Path Scenario test. Виведення криптовалюти

Description

1. Перейти до розділу "Withdrawal".
2. Вибрати гаманець з криптовалютою, яку потрібно вивести (наприклад, Bitcoin).
3. У розділі "Enter the amount" ввести кількість криптовалюти для виведення.
4. Система має відобразити поточний курс обміну та суму у фіатній валюті, яку буде отримано після виведення.
5. Підтвердити операцію виведення.
6. Ввести дані банківського рахунку для отримання фіатних коштів.
7. Підтвердити операцію виведення.
8. Система має відобразити повідомлення про успішне виведення.
9. Перевірити, що баланс гаманця криптовалюти оновився відповідно до виведеної кількості.
10. Перевірити, що баланс фіатних коштів поповнився відповідно до суми виведення.

Input

Кількість криптовалюти для виведення.
Дані рахунку для отримання фіатних коштів.

Acceptance Criteria

Results

Процес виведення криптовалюти проходить без збоїв та помилок у позитивному сценарії.
Баланси гаманця криптовалюти та фіатних коштів оновлюються коректно після успішного виведення.

[Last run Not Run at Invalid DateTime by and checked by .]

Рисунок А.5.1 – Тест-сценарій «Виведення криптовалюти»

Test Report

15 April, 2024

Виведення коштів

UseCase in package 'Use Cases'

Виведення коштів

Version 1.0 Phase 1.0 Proposed

snox.jz created on 29.03.2024. Last modified 29.03.2024

TESTING

Basic Path Scenario test. Alternate

Description

...
11. Спробувати вивести більшу кількість криптовалюти, ніж є на балансі гаманця.
12. Система має відобразити повідомлення про помилку та не дозволити виведення.
13. Перевірити, що баланси гаманця криптовалюти та фіатних коштів не змінилися.

Input

Кількість криптовалюти для виведення.
Дані банківського рахунку для отримання фіатних коштів.

Acceptance Criteria

Results

Система правильно відхиляє спробу виведення та не змінює баланси.

[Last run Not Run at Invalid DateTime by and checked by .]

Рисунок А.5.2 – Тест-сценарій (Альтернативний сценарій) «Виведення криптовалюти (негативний випадок)»

ДОДАТОК Б

Фрагменти коду, що було створено для реалізації бекенду системи

(контролери)

InvestmentDeposit:

```
package reflectiveInv.entity;  
import lombok.AllArgsConstructor;  
import lombok.Getter;  
import lombok.NoArgsConstructor;  
import lombok.Setter;  
import reflectiveInv.repository.InvestmentDepositRepository;  
import javax.persistence.*;  
import java.util.List;
```

@Getter

@Setter

@NoArgsConstructor

@AllArgsConstructor

@Entity

@Table(name = "investmentDeposits")

public abstract class InvestmentDeposit implements InvestmentDepositRepository {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private long id;

private String token;

private String investmentPlan;

private double tokenAmount;

@ManyToOne

@JoinColumn(name = "user_id")

private Users users;

@OneToMany(mappedBy = "investmentDeposits")

private List<TokenHasInvest> tokenHasInvests;

}

Token:

```
package reflectiveInv.entity;  
import lombok.AllArgsConstructor;  
import lombok.Getter;  
import lombok.NoArgsConstructor;  
import lombok.Setter;  
import javax.persistence.*;  
import java.util.List;
```

```

@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor

@Entity
@Table(name = "token")
public class Token {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;

    public void setToken_name(String token_name) {
        this.token_name = token_name;
    }

    public void setPrice(double price) {
        this.price = price;
    }

    @Getter
    private String token_name;
    @Getter
    private double price;

    @OneToMany(mappedBy = "token")
    private List<Withdrawal> withdrawal;

    @OneToMany(mappedBy = "token")
    private List<TokenHasInvest> tokenHasInvestList;
}

```

TokenHasInvest:

```

package reflectiveInv.entity;
import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;
import reflectiveInv.repository.TokenHasInvestRepository;
import javax.persistence.*;

@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor

@Entity
@Table(name = "token_has_invest")
public abstract class TokenHasInvest implements TokenHasInvestRepository {

```

```

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private long id;

private int quantity;

@ManyToOne
@JoinColumn(name = "token_id")
private Token token;

@ManyToOne
@JoinColumn(name = "investmentDeposits_id")
private InvestmentDeposit investmentDeposit;

public TokenHasInvest(int quantity, Withdrawal withdrawal, InvestmentDeposit
investmentDeposit) {
    this.quantity = quantity;
    this.token = token;
    this.investmentDeposit = investmentDeposit;
}
}

```

Users:

```

package reflectiveInv.entity;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import javax.persistence.*;
import javax.validation.constraints.Size;
import java.util.List;

@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor

@Entity
@Table(name = "users")
public class Users {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;
    private String name;
    private String surname;
    private int phone;
    private int balance;
}

```

```

private boolean wallet;

@Size(min=4, message = "Введіть 4 символи")
private String login;

@Size(min=4, message = "Введіть 4 символи")
private String pass;

@OneToMany(mappedBy = "users")
private List<InvestmentDeposit> listInvestmentDeposits;
}

```

Withdrawal:

```

package reflectiveInv.entity;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import javax.persistence.*;

@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor

@Entity
@Table(name = "withdrawal")
public class Withdrawal {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;

    public void setWithdrawalAmount(double withdrawalAmount) {
        this.withdrawalAmount = withdrawalAmount;
    }

    @Getter
    private double withdrawalAmount;
    private int userId;
    private double availableBalance;

    @ManyToOne
    @JoinColumn(name = "token_id")
    private Token token;
}

```

HTML код сторінки особистого кабінету:

```
<nav class="navbar navbar-expand-md navbar-light bg-light">
  <div class="container">
    <div class="logo">
      <a href="../index.html">
        
      </a>
    </div>
    <div class="site-menu">
      <div class="menu">
        <div class="collapse navbar-collapse">
          <ul class="nav navbar-nav" itemscope
itemtype="http://www.schema.org/SiteNavigationElement">
            <li class="nav-item">
              <a itemprop="url" href="cabinet-user.html" class="nav-link">
                <span itemprop="name">Cabinet <span class="border border-
top"></span>
                <span class="border border-right"></span>
                <span class="border border-bottom"></span>
                <span class="border border-left"></span>
              </span>
            </a>
          </li>
          <li class="nav-item">
            <a itemprop="url" href="deposit.html" class="nav-link" aria-
current="page">
              <span itemprop="name">Deposit <span class="border border-
top"></span>
              <span class="border border-right"></span>
              <span class="border border-bottom"></span>
              <span class="border border-left"></span>
            </span>
          </a>
        </li>
        <li class="nav-item">
```

```

<a href="operation-cach.html" data-toggle="dropdown" aria-expanded="false"
    class="dropdown-toggle nav-link">
    <span itemprop="name">Withdrawal<span class="border border-
top"></span>

    <span class="border border-right"></span>
    <span class="border border-bottom"></span>
    <span class="border border-left"></span>
    </span>
</a>
</li>
<li class="nav-item">
    <a itemprop="url" href="operations.html" class="nav-link">
    <span itemprop="name">Operations <span class="border border-
top"></span>

    <span class="border border-right"></span>
    <span class="border border-bottom"></span>
    <span class="border border-left"></span>
    </span>
</a>
</li>
<li class="nav-item">
    <a itemprop="url" href="partners.html" class="nav-link">
    <span itemprop="name">Partners <span class="border border-
top"></span>

    <span class="border border-right"></span>
    <span class="border border-bottom"></span>
    <span class="border border-left"></span>
    </span>
</a>
</li>
<li class="nav-item">
    <a itemprop="url" href="settings.html" class="nav-link">
    <span itemprop="name">Settings <span class="border border-
top"></span>

    <span class="border border-right"></span>

```

```

        <span class="border border-bottom"></span>
        <span class="border border-left"></span>
    </span>
</a>
</li>
<li class=" nav-item">
    <a itemprop="url" href="tickets.html" class="nav-link">
        <span itemprop="name">Tickets <span class="border border-
top"></span>

        <span class="border border-right"></span>
        <span class="border border-bottom"></span>
        <span class="border border-left"></span>
    </span>
    </a>
</li>
</ul>
</div>
</div>
</div>
<div class="hamburger-menu">
    <span></span>
    <span></span>
    <span></span>
</div>
<div class="navbar-button">
    <div class="telh">
        &nbsp;&nbsp;&nbsp;timwer
    </div>
</div>
</div>
</nav>

```

```

<header class="page-header" data-background="">
  <div class="container">
    <h2>PERSONAL STATISTICS
  </h2>
</div>
</header>
<main>
  <div class="tablo">
    <div class="tablo--1-ve-3 wow fadeInLeft cabinet" data-wow-delay="0.5s"
      style="visibility: visible; animation-delay: 0.5s; animation-name: fadeInLeft;">
      <div class="ozellik-kutu-iletisim" onclick="location.href='#';"
        style="cursor: pointer; will-change: transform; transform: perspective(300px)
rotateX(0deg) rotateY(0deg);"
        data-tilt="">
        
        <h3 class="baslik-4 h-yazi-margin-kucuk-2">0 USDT</h3>
        <a href="#" class="custom-button wow fadeInUp cabinet" data-wow-delay="0.6s"
          style="visibility: visible; animation-delay: 0.6s; animation-name:
fadeInUp;">Deposit →</a>
        <p class="ozellik-kutu-iletisim--yazi">Referral: 0 USDT</p>
        <p class="ozellik-kutu-iletisim--yazi">Withdrawal: 0 USDT</p>
        <p class="ozellik-kutu-iletisim--yazi">Profit: 0 USDT</p>
      </div>
    </div>
    <div class="tablo--1-ve-3 wow fadeInLeft cabinet" data-wow-delay="0.5s"
      style="visibility: visible; animation-delay: 0.5s; animation-name: fadeInLeft;">
      <div class="ozellik-kutu-iletisim" onclick="location.href='#';"
        style="cursor: pointer; will-change: transform; transform: perspective(300px)
rotateX(0deg) rotateY(0deg);"
        data-tilt="">
        
        <h3 class="baslik-4 h-yazi-margin-kucuk-2">0 ETH</h3>
        <a href="#" class="custom-button wow fadeInUp cabinet" data-wow-delay="0.6s"

```

```

        style="visibility: visible; animation-delay: 0.6s; animation-name:
fadeInUp;">Deposit →</a>
        <p class="ozellik-kutu-iletisim--yazi">Referral: 0 ETH</p>
        <p class="ozellik-kutu-iletisim--yazi">Withdrawal: 0 ETH</p>
        <p class="ozellik-kutu-iletisim--yazi">Profit: 0 ETH</p>
    </div>
</div>
<div class="tablo--1-ve-3 wow fadeInLeft cabinet" data-wow-delay="0.5s"
style="visibility: visible; animation-delay: 0.5s; animation-name: fadeInLeft;">
    <div class="ozellik-kutu-iletisim" onclick="location.href='#";"
        style="cursor: pointer; will-change: transform; transform: perspective(300px)
rotateX(0deg) rotateY(0deg);"
        data-tilt="">
        
        <h3 class="baslik-4 h-yazi-margin-kucuk-2">0 BTC</h3>
        <a href="#" class="custom-button wow fadeInUp cabinet" data-wow-delay="0.6s"
            style="visibility: visible; animation-delay: 0.6s; animation-name:
fadeInUp;">Deposit →</a>
        <p class="ozellik-kutu-iletisim--yazi">Referral: 0 BTC</p>
        <p class="ozellik-kutu-iletisim--yazi">Withdrawal: 0 BTC</p>
        <p class="ozellik-kutu-iletisim--yazi">Profit: 0 BTC</p>
    </div>
</div>
<div class="tablo--1-ve-3 wow fadeInLeft cabinet" data-wow-delay="0.5s"
style="visibility: visible; animation-delay: 0.5s; animation-name: fadeInLeft;">
    <div class="ozellik-kutu-iletisim" onclick="location.href='#";"
        style="cursor: pointer; will-change: transform; transform: perspective(300px)
rotateX(0deg) rotateY(0deg);"
        data-tilt="">
        
        <h3 class="baslik-4 h-yazi-margin-kucuk-2">0 XRP</h3>
        <a href="#" class="custom-button wow fadeInUp cabinet" data-wow-delay="0.6s"
            style="visibility: visible; animation-delay: 0.6s; animation-name:
fadeInUp;">Deposit →</a>

```

```
<p class="ozellik-kutu-iletisim--yazi">Referral: 0 XRP</p>
    <p class="ozellik-kutu-iletisim--yazi">Withdrawal: 0 XRP</p>
    <p class="ozellik-kutu-iletisim--yazi">Profit: 0 XRP</p>
</div>
</div>
<div class="tablo--1-ve-3 wow fadeInLeft cabinet" data-wow-delay="0.5s"
    style="visibility: visible; animation-delay: 0.5s; animation-name: fadeInLeft;">
    <div class="ozellik-kutu-iletisim" onclick="location.href='#';"
        style="cursor: pointer; will-change: transform; transform: perspective(300px)
        rotateX(0deg) rotateY(0deg);"
        data-tilt="">
        
        <h3 class="baslik-4 h-yazi-margin-kucuk-2">0 LTC</h3>
        <a href="#" class="custom-button wow fadeInUp cabinet" data-wow-delay="0.6s"
            style="visibility: visible; animation-delay: 0.6s; animation-name:
            fadeInUp;">Deposit →</a>
        <p class="ozellik-kutu-iletisim--yazi">Referral: 0 LTC</p>
        <p class="ozellik-kutu-iletisim--yazi">Withdrawal: 0 LTC</p>
        <p class="ozellik-kutu-iletisim--yazi">Profit: 0 LTC</p>
    </div>
</div>
</div>
<div>
<section class="lastOperations">
    <h2 class="h2-baslik-hizmetler-2 wow fadeInUp">LAST TRANSACTIONS</h2>
    <div class="col-lg usttablo">
        <div class="box wow flipInY" data-wow-delay="0.7s"
            style="visibility: visible; animation-delay: 0.7s; animation-name: flipInY;">
            <span></span>
            <span></span>
            <span></span>
            <span></span>
            <div class="content1">
                <div class="paketler__pr-kutu">
```

```
        <div class="iconw">
            <i class="flaticon-badge"></i>
        </div>

        <h3 class="baslik-3 h-yazi-margin-kucuk">You have not made any
transactions</h3>

    </div>

</div>

</div>

</div>

</div>

</section>
```

Фрагмент CSS коду:

```
.inputLabel {
    color: #D9B08C;
    margin: 0;
    font-weight: 600;
    font-size: 24px;
}

.radioLabel {
    position: relative;
    font-size: 18px;
    line-height: 24px;
    color: #D9B08C;
    font-weight: 400;
    letter-spacing: .05em;
    cursor: pointer;
}

.cabMoneyMoveResult,
.cabMoveDataItem {
    padding-bottom: 10px;
```

```
}
```

```
#add_Sum {  
    margin-bottom: 10px;  
}
```

```
.iconn.icon-profile {  
    font-size: 24px;  
}
```

```
.tablo-1-ve-3.wow.cabinet {  
    padding-bottom: 20px !important;  
}
```

```
.custom-button.wow.fadeInUp.cabinet {  
    display: inline !important;  
    padding: 10px 40px;  
}
```

```
.crypto {  
    padding-bottom: 10px;  
    width: 48px;  
}
```

```
.lastOperations {  
    margin-bottom: 40px;  
    margin-top: 40px;  
}
```

```
.withd {  
    font-size: 24px;  
    color: #D9B08C;
```

```
}
```

```
.payItem {  
  display: flex;  
}
```

```
.filters {  
  display: flex;  
}
```

```
.operationItems {  
  display: flex;  
  flex-wrap: wrap;  
  margin: 0;  
  margin-bottom: 100px !important;  
}
```

```
.partnersHeader {  
  color: #D1E8E2;  
  font-weight: 500;  
}
```

```
.invited {  
  font-size: 20px;  
  color: #D1E8E2;  
  margin-top: 20px;  
}
```

```
.iletisim-form-alani.settings {  
  /* padding-top: 10rem; */  
}
```

```
.settingsHeader {  
  font-size: 24px;  
  color: #D9B08C;  
  text-align: center;  
  font-weight: 600;  
}
```

```
.settingsHeader.sec {  
  padding-top: 60px;  
}
```

```
.payIcon.settings {  
  margin-bottom: 10px;  
}
```

```
.walletName {  
  color: #2c3531;  
  margin-bottom: 10px;  
}
```

```
.logItems {  
  text-align: center;  
  display: flex;  
  gap: 30px;  
  justify-content: center;  
}
```

```
.logItem {  
  text-align: center;  
  color: #D9B08C;  
  font-size: 18px;  
  margin: 0;
```

```
}
```

```
.logItem:hover {  
  color: #2c3531;  
}
```

```
@media (max-width: 901px) {  
  .iletisim-form-alani.register {  
    margin-top: -19vh !important;  
  }  
}
```

Фрагмент JS коду, що відповідає за анімації переходу, задній фон, слайдери та кнопку меню у мобільній версії:

```
// HAMBURGER MENU
```

```
var main = function () {  
  $('hamburger-menu').click(function () {  
    $('.side-widget').animate({  
      left: "0px"  
    }, 200, function () {  
  
      $(document).on("click.menu", function (event) {  
        var target = $(event.target);  
        console.log(target);  
        if (!target.closest(".side-widget").length || target.closest(".closed").length) {  
          closeMenu(function () {  
            $(document).off("click.menu");  
          });  
        }  
      })  
    });  
  });  
});
```

```
// DATA BACKGROUND IMAGE
```

```
var pageSection = $("*");
pageSection.each(function (indx) {
  if ($(this).attr("data-background")) {
    $(this).css("background", "url(" + $(this).data("background") + ")");
  }
});
```

```
// DATA BACKGROUND COLOR
```

```
var pageSection = $("*");
pageSection.each(function (indx) {
  if ($(this).attr("data-background")) {
    $(this).css("background", $(this).data("background"));
  }
});
```

```
// WOW ANIMATION
```

```
wow = new WOW({
  animateClass: 'animated',
  offset: 0
});
wow.init();
```

```
// CAROUSEL CLASSES SLIDER
```

```
var swiper = new Swiper('.carousel-classes', {
  slidesPerView: '3',
  spaceBetween: 10,
  loop: 'true',
  pagination: {
    el: '.swiper-pagination',
    clickable: true,
  },
});
```

```
breakpoints: {  
  640: {  
    slidesPerView: 1,  
    spaceBetween: 15,  
  },  
  768: {  
    slidesPerView: 2,  
    spaceBetween: 30,  
  },  
  1024: {  
    slidesPerView: 3,  
    spaceBetween: 30,  
  },  
}  
});  
  
// MAIN SLIDER  
var swiper = new Swiper('.main-slider', {  
  slidesPerView: '1',  
  speed: 900,  
  loop: true,  
  effect: 'fade',  
  opacity: 1,  
  autoplay: {  
    delay: 15000,  
    disableOnInteraction: false,  
  },  
  navigation: {  
    prevEl: '.button-prev',  
    nextEl: '.button-next',  
  },  
  pagination: {
```

```
    el: '.swiper-pagination',  
    clickable: true,  
  },  
});
```

```
// PRELOADER  
$(window).load(function () {  
  $("body").addClass("page-loaded");  
});
```

ДОДАТОК В

Сторінки прототипу системи зроблені у Figma[18]

Розглянемо прототипи сторінок інформаційної системи «Криптоінвестиції» що представлені на рисунках В.1 – В.29.



Рисунок В.1 – Ієрархія створених сторінок



Рисунок В.3 – Сторінка «About Us»

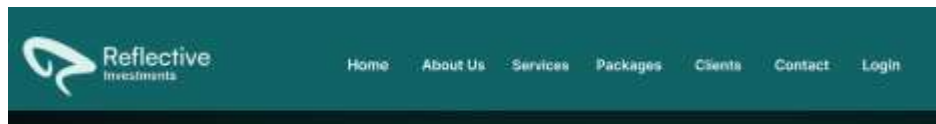


Рисунок В.4-5 – Випадаючий список «Services»



Рисунок В.6 – Всі сервіси для підтримки інвесторів



Рисунок В.7 – Сервіс «Server Storage Systems»

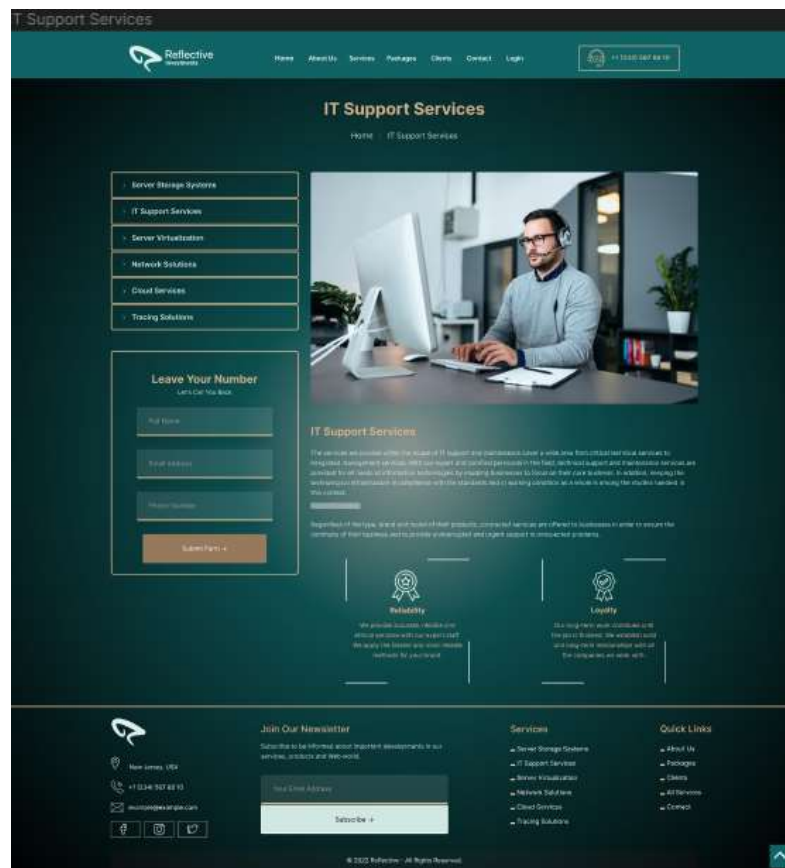


Рисунок В.8 – Сервіс «IT Support Services»

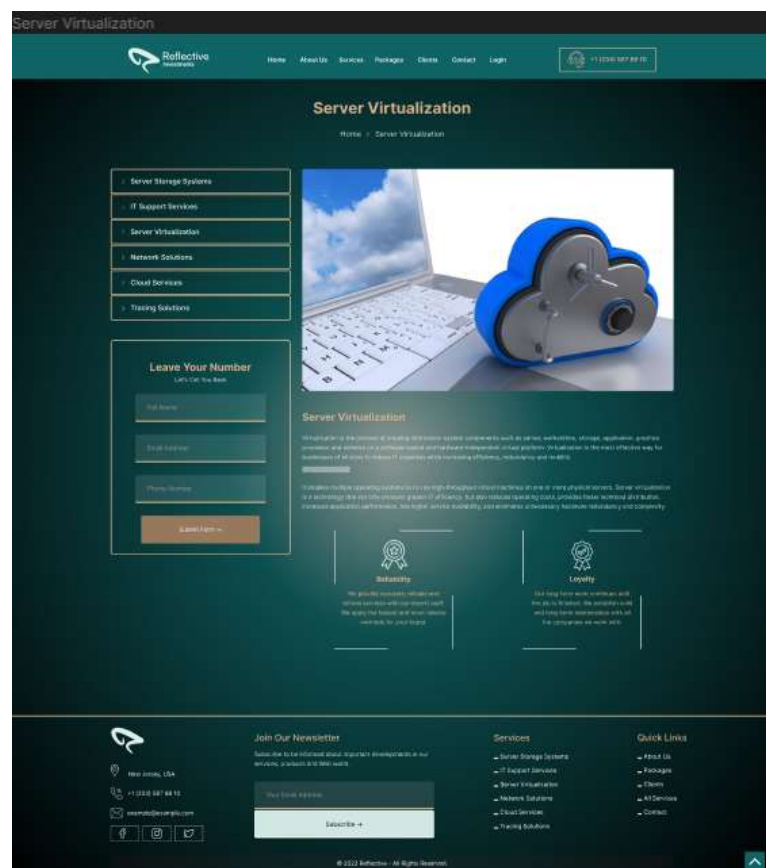


Рисунок В.9 – Сервіс «Server Virtualization»

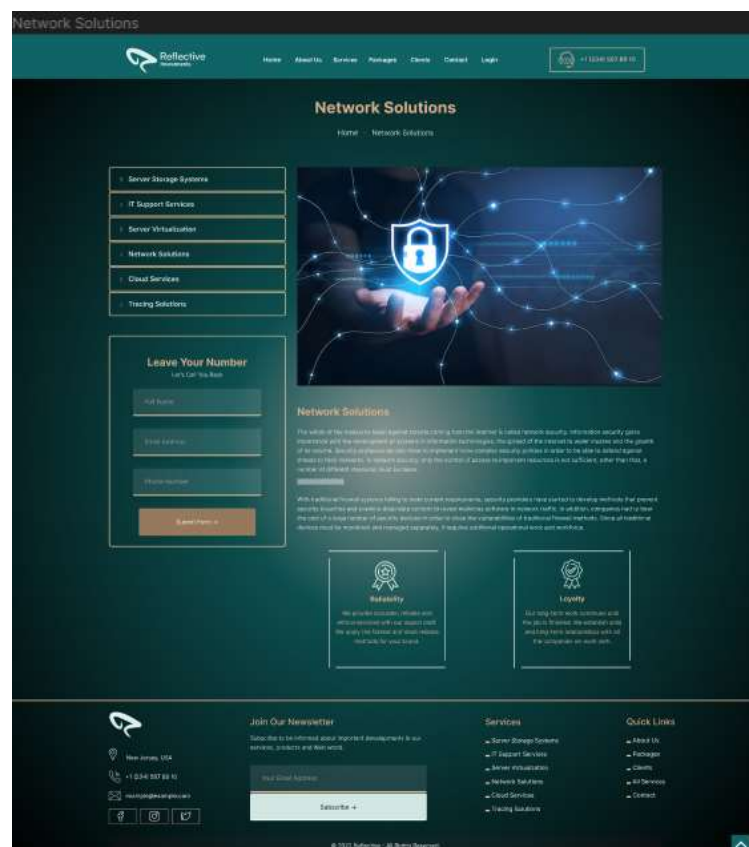


Рисунок В.10 – Сервіс «Network Solutions»

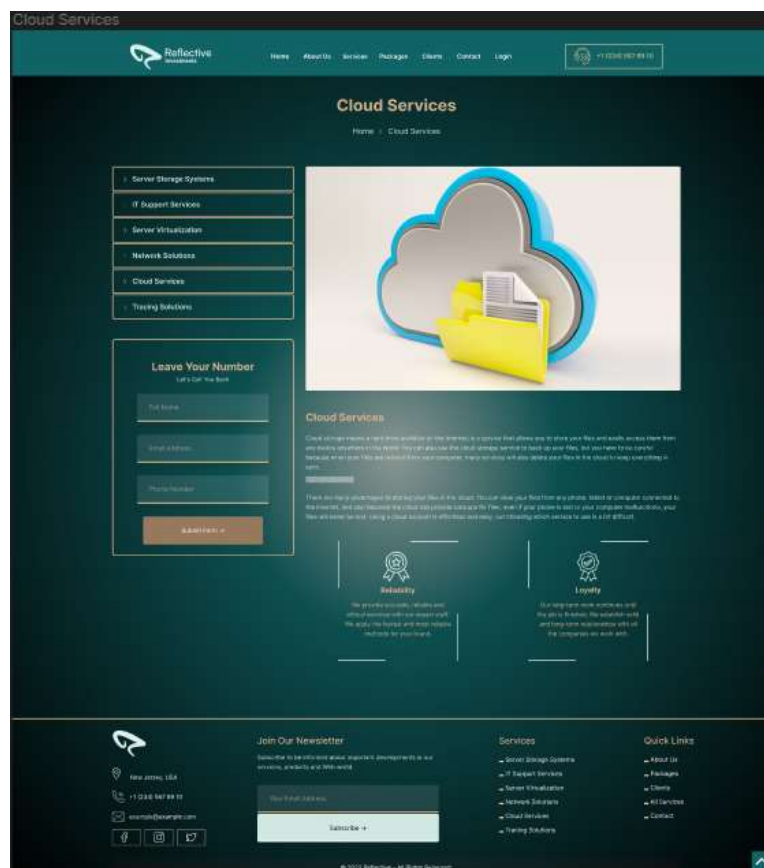


Рисунок В.11 – Сервіс «Cloud Services»

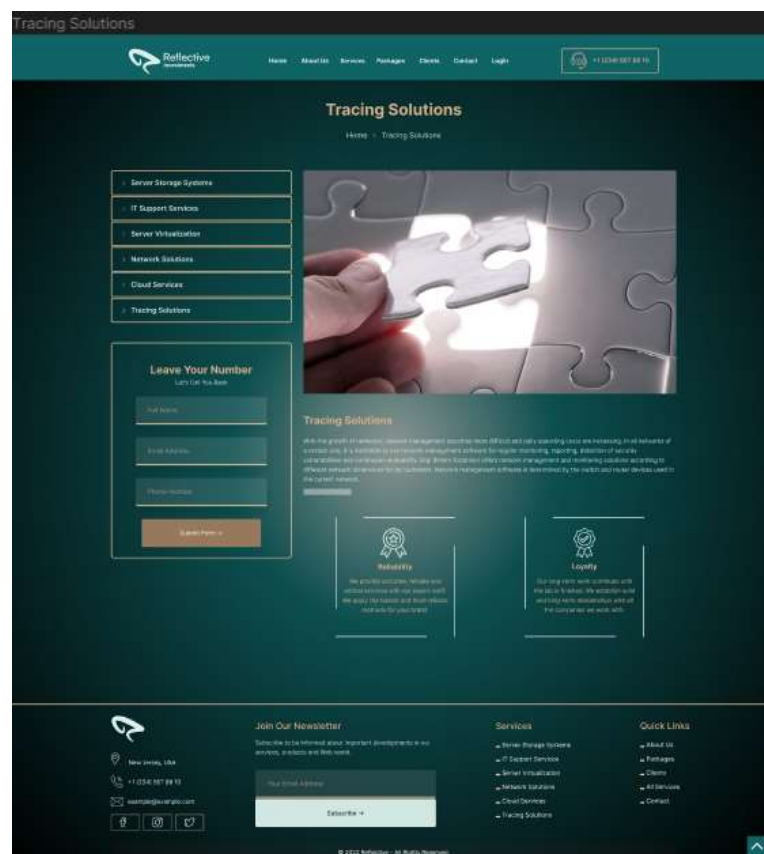


Рисунок В.12 – Сервіс «Tracing Solutions»

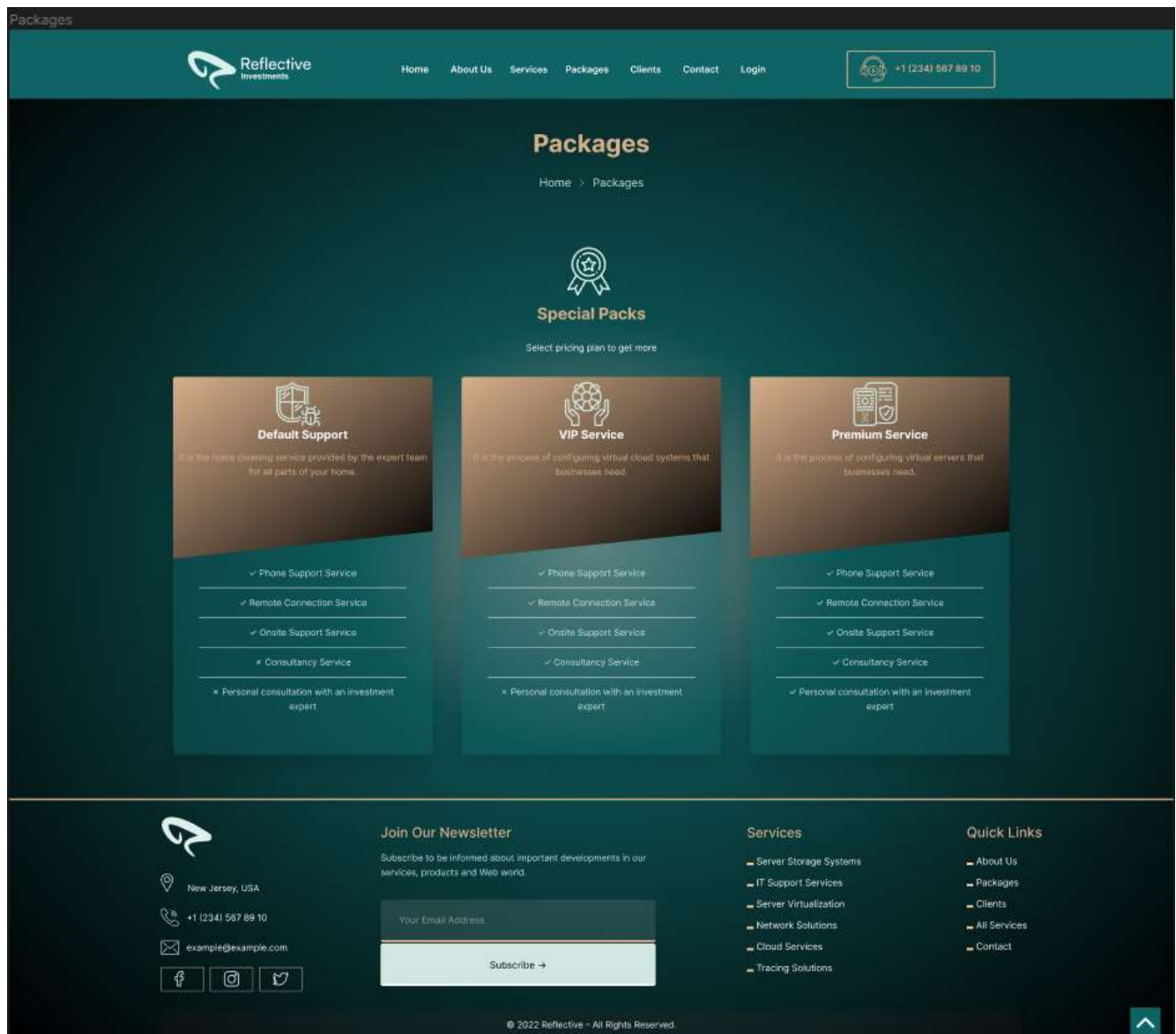


Рисунок В.13 – Сторінка вибору пакету підтримки інвесторів з додатковими сервісами

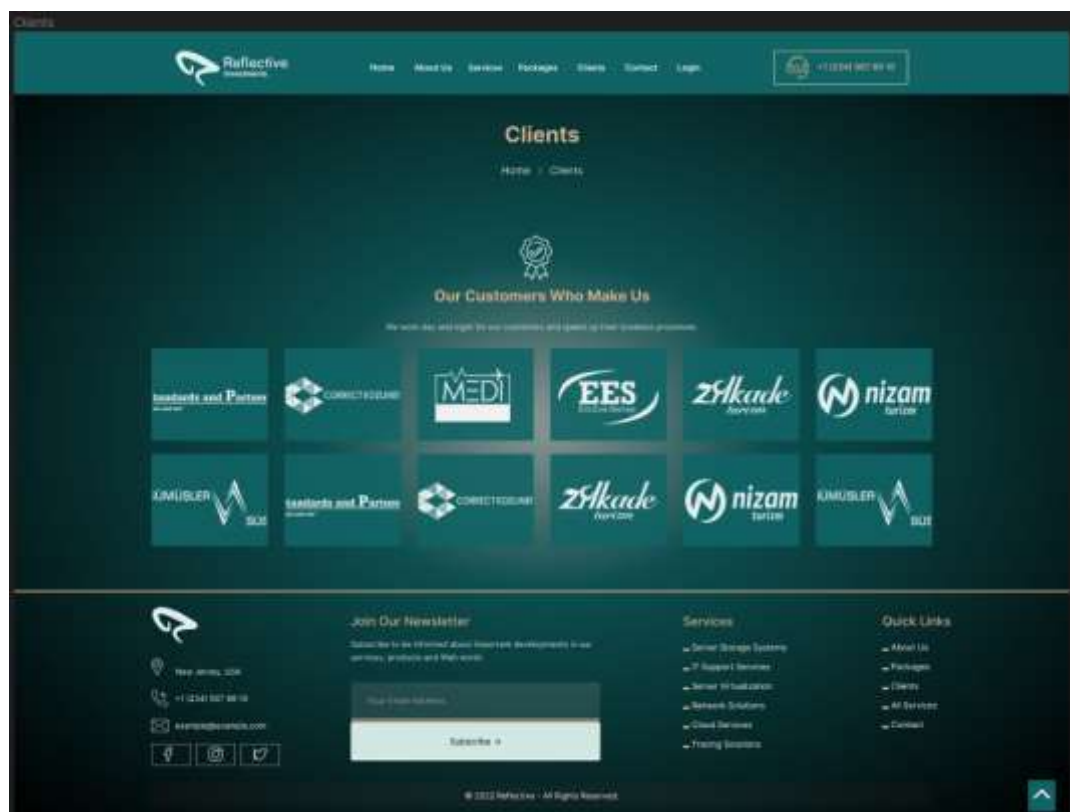


Рисунок В.14 – Сторінка клієнтів-інвесторів

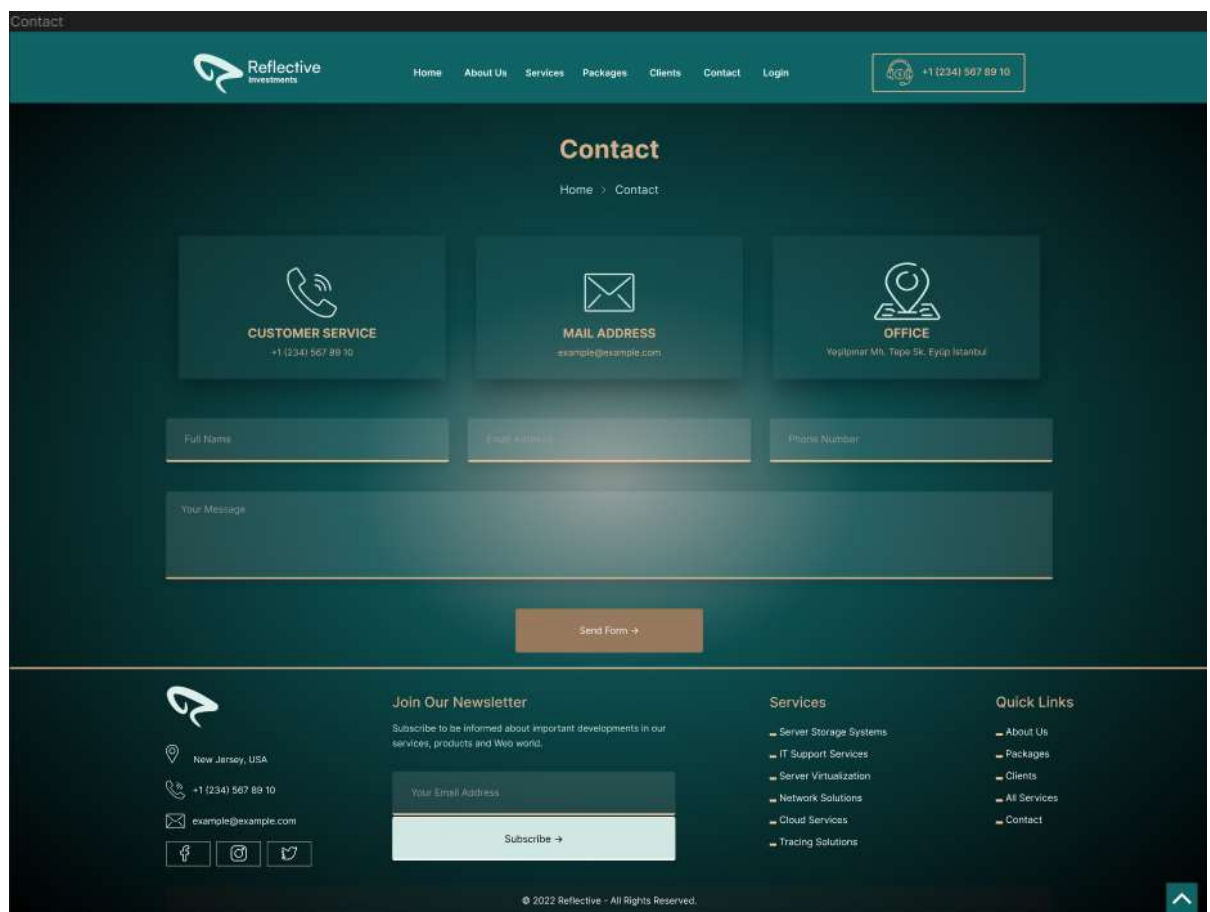


Рисунок В.15 – Сторінка зворотного зв'язку

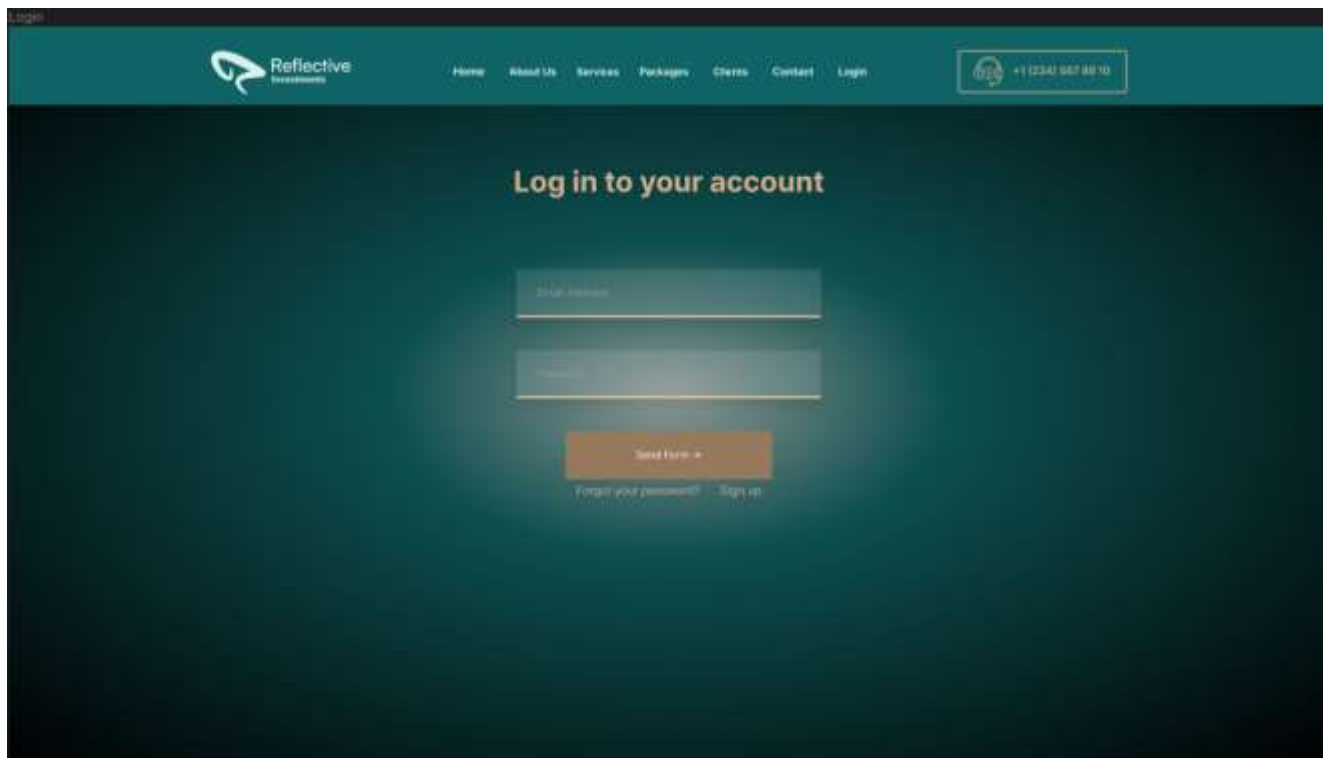


Рисунок В.16 – Сторінка входу в аккаунт

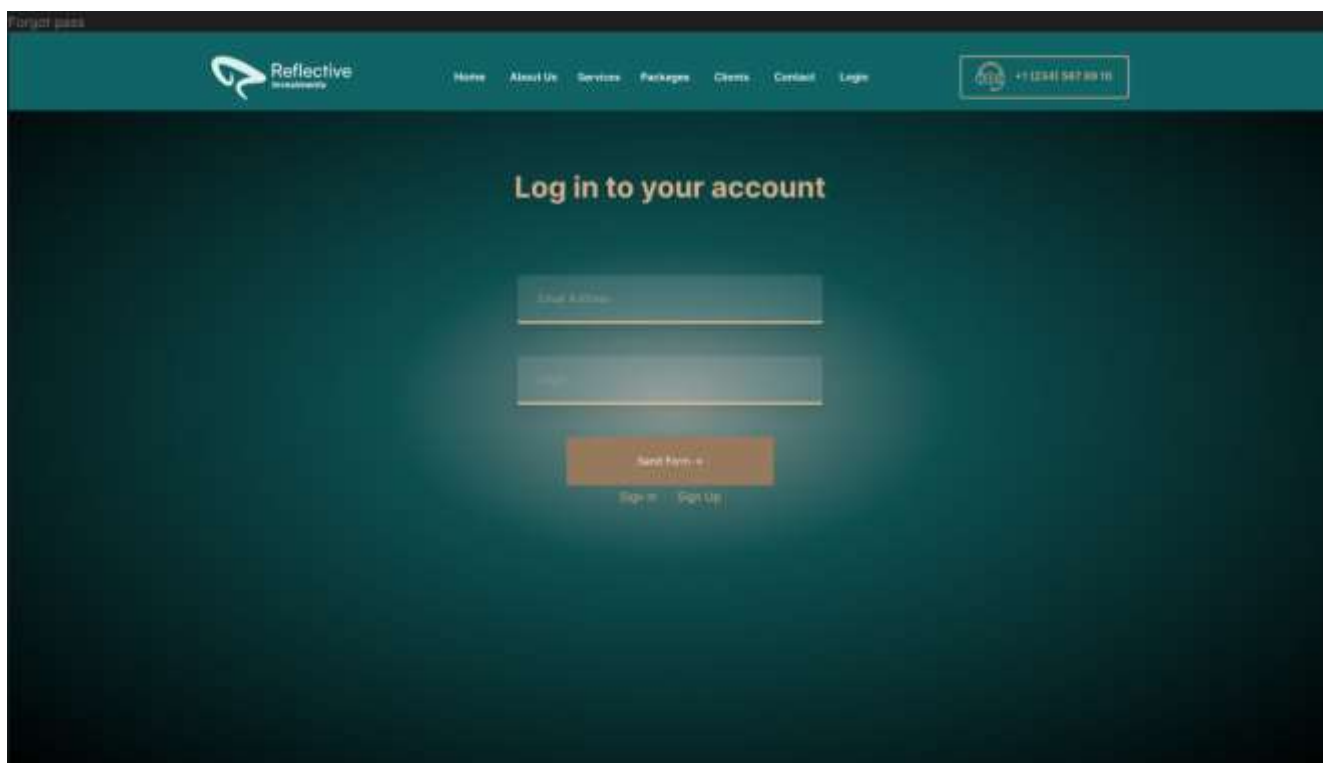


Рисунок В.17 – Сторінка відновлення доступу к аккаунту

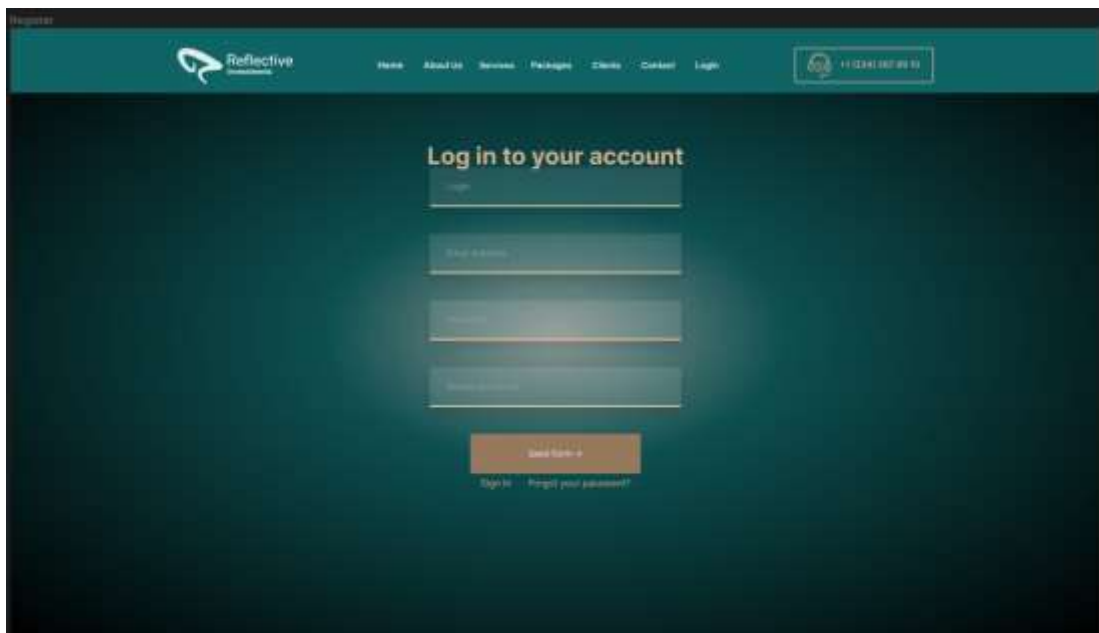


Рисунок В.18 – Сторінка реєстрації

Дизайн особистого кабінету.

Після входу в аккаунт, або реєстрації нового, нам стає доступним особистий кабінет та подальші операції введення та виведення коштів.



Рисунок В.19 – Сторінка особистого кабінету зі статистикою доходу

Deposit

Reflective
Blockchain

Cabinet
Deposit
Withdrawal
Operations
Partners
Settings
Tools

SELECT INVESTMENT PLAN

Cabinet / Deposit

RETURN

0.4%-0.6%

10 days

Amount of deposit:
50 USDT - 300.01 USDT

ACCUMULS EVERY DAY

Profit per day: 0.08 USDT
Total profit: 0.88 USDT

RETURN

0.7%-0.9%

25 days

Amount of deposit:
2000 usdt - 4999.01 usdt

ACCUMULS EVERY DAY

Profit per day: 0.38 USDT
Total profit: 9.25 USDT

RETURN

1%-1.2%

55 days

Amount of deposit:
2000 usdt - 4999.01 usdt

ACCUMULS EVERY DAY

Profit per day: 0.12 USDT
Total profit: 6.60 USDT

SELECT PAYMENT SYSTEM

ETH
Available: 0 ETH

BTC
Available: 0 BTC

LTC
Available: 0 LTC

USDT
Available: 0 USDT

XRP
Available: 0 XRP

MAKE A DEPOSIT

Enter the amount:

10

Min/Max deposit amount: 50 USDT - 300.01 USDT

☒ From the payment system

☐ From the balance

Plan: 0.6% 10 days

Amount: 10 USDT

Deposit

New Jersey, USA
+1 (234) 567 89 10
example@example.com

Join Our Newsletter

Subscribe to be informed about important developments in our services, products and this world.

Subscribe

Services

- Server Storage Systems
- IT Support Services
- Server Virtualization
- Network Solutions
- Cloud Services
- Tracing Solutions

Quick Links

- About Us
- Packages
- Clients
- All Services
- Contact

© 2022 Reflective - All Rights Reserved

Рисунок В.20 – Сторінка вибору плану інвестицій та бажаної суми депозиту

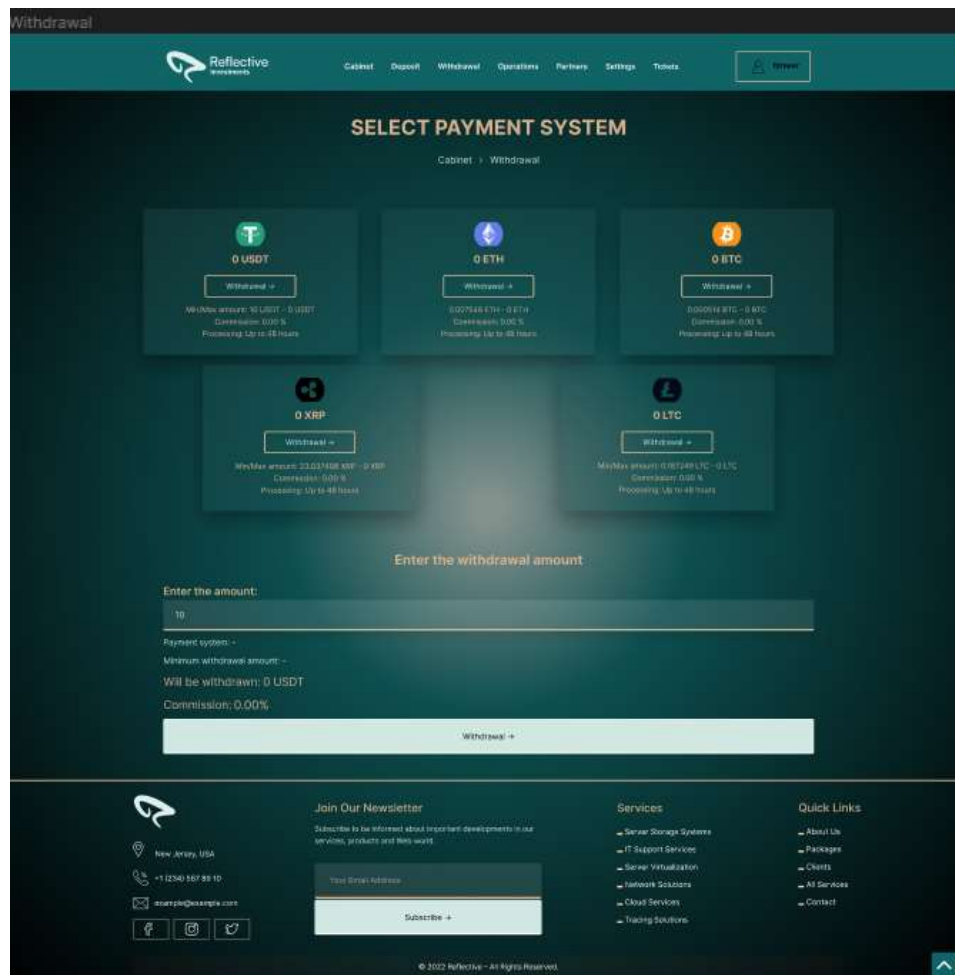


Рисунок В.21 – Сторінка виводу коштів

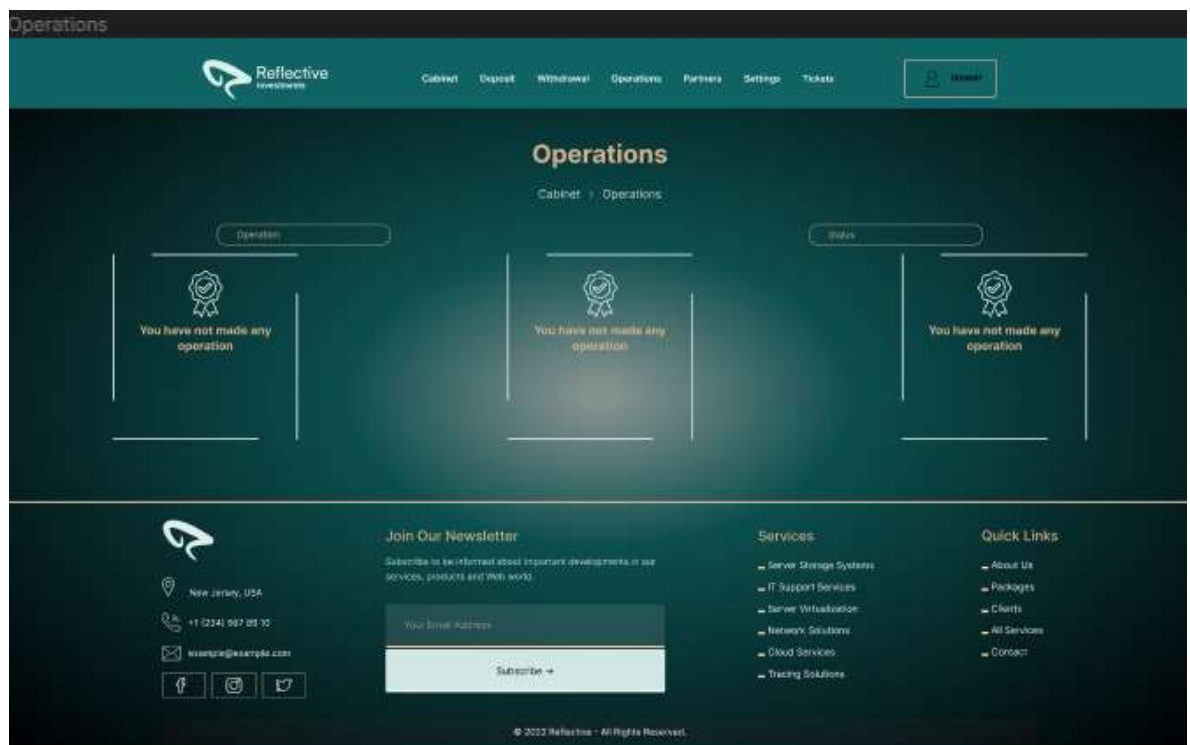


Рисунок В.22 – Сторінка з історією операцій

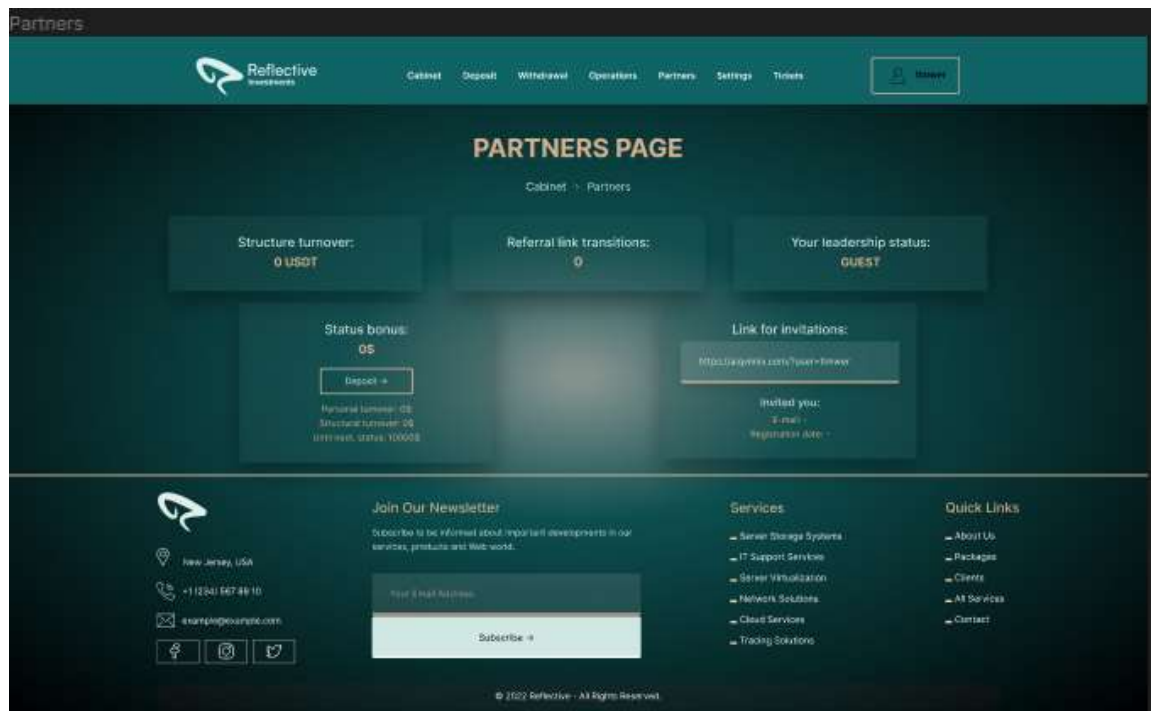


Рисунок В.23 – Сторінка реферальної програми

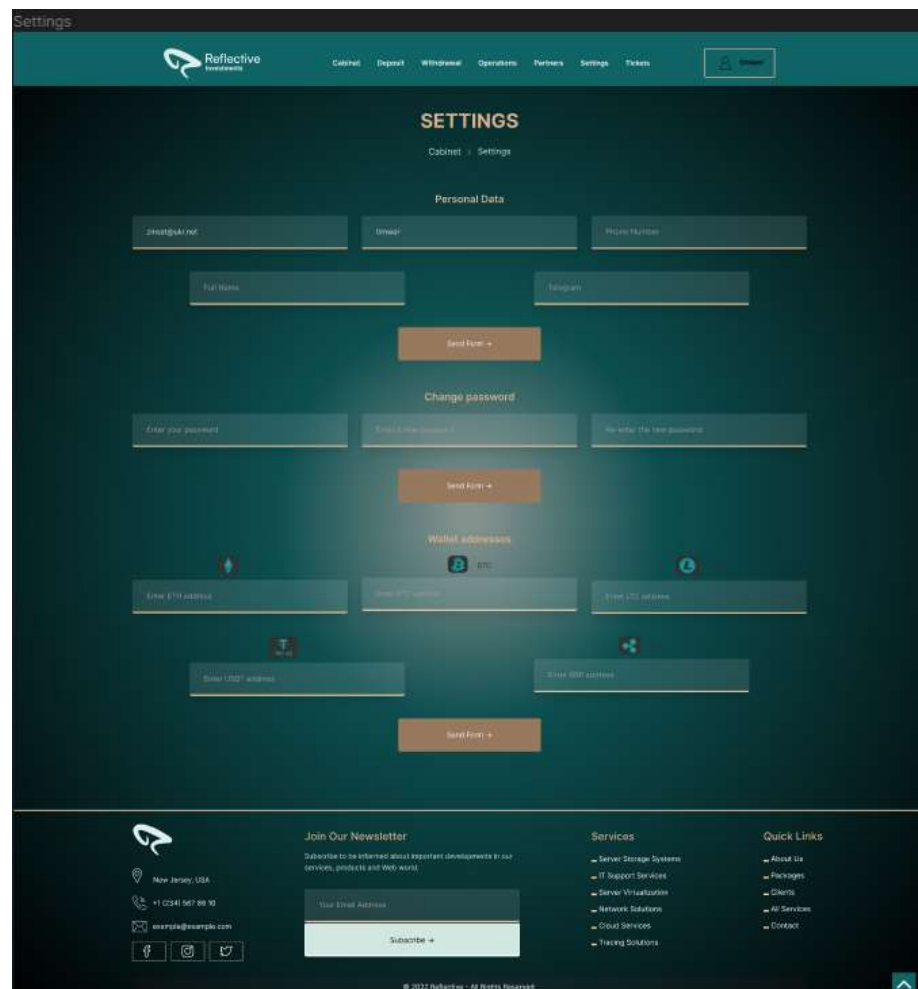


Рисунок В.24 – Сторінка налаштувань та додавання адрес гаманців

Tickets

 Reflective
INCUBATORS

[Cabinet](#) [Desktop](#) [Withdrawal](#) [Operations](#) [Partners](#) [Settings](#) [Tickets](#)

 [Logout](#)

CREATE A TICKET

[Cabinet](#) > [Tickets](#)

[Send Form >](#)



 New Jersey, USA

 +1 (234) 567 89 10

 support@reflective.com

Join Our Newsletter

Subscribe to be informed about important developments in our services, products and this world.

[Subscribe >](#)

Services

- Server Storage Systems
- IT Support Services
- Server Virtualization
- Network Solutions
- Cloud Services
- Training Solutions

Quick Links

- About Us
- Packages
- Clients
- All Services
- Contact

© 2022 Reflective - All Rights Reserved.

Рисунок В.25 – Сторінка скарг та побажань

ДОДАТОК Г

Результат перетворення ER-моделі в модель даних СУБД MySQL. Згенерований код:

```
CREATE TABLE Users (  
    id BIGINT PRIMARY KEY,  
    name NVARCHAR(255),  
    surname NVARCHAR(255),  
    phone INT,  
    balance INT,  
    wallet BIT,  
    login NVARCHAR(255) CHECK (LEN(login) >= 4),  
    pass NVARCHAR(255) CHECK (LEN(pass) >= 4)  
);  
  
CREATE TABLE Token (  
    id BIGINT PRIMARY KEY,  
    token_name NVARCHAR(255),  
    price FLOAT  
);  
  
CREATE TABLE InvestmentDeposit (  
    id BIGINT PRIMARY KEY,  
    token NVARCHAR(255),  
    investmentPlan NVARCHAR(255),  
    tokenAmount FLOAT,  
    user_id BIGINT,  
    FOREIGN KEY (user_id) REFERENCES Users(id)  
);  
  
CREATE TABLE TokenHasInvest (  
    id BIGINT PRIMARY KEY,  
    quantity INT,  
    token_id BIGINT,  
    investmentDeposits_id BIGINT,  
    FOREIGN KEY (token_id) REFERENCES Token(id),  
    FOREIGN KEY (investmentDeposits_id) REFERENCES InvestmentDeposit(id)  
);  
  
CREATE TABLE Withdrawal (  
    id BIGINT PRIMARY KEY,  
    withdrawalAmount FLOAT,  
    userId BIGINT,  
    availableBalance FLOAT,  
    token_id BIGINT,  
    FOREIGN KEY (userId) REFERENCES Users(id),  
    FOREIGN KEY (token_id) REFERENCES Token(id)  
);
```

Ім'я користувача:

Інформаційних систем в економіці Шкуратовська Те...

Дата перевірки:

04.06.2024 00:41:43 EEST

Дата звіту:

04.06.2024 07:09:44 EEST

ID перевірки:

1016317169

Тип перевірки:

Doc vs Internet + Library

ID користувача:

100005745

Назва документа: Попов_Д_ІН-404

Кількість сторінок: **91** Кількість слів: **11997** Кількість символів: **101639** Розмір файлу: **3.25 MB** ID файлу: **1016114611**

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

9.54%

Схожість

Найбільша схожість: **3.63%** з джерелом з Бібліотеки (ID файлу: **1016112527**)**2.35%** Джерела з Інтернету

365

Сторінка 93

9.34% Джерела з Бібліотеки

570

Сторінка 95

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%

Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи

13

Підозріле форматування

20
сторінок

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
«КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ
УНІВЕРСИТЕТ ІМЕНІ ВАДИМА ГЕТЬМАНА»
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
«ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ В
ЕКОНОМІЦІ»

Кафедра інформаційних систем в економіці

галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНИЙ БАКАЛАВРСЬКИЙ ПРОЕКТ

на тему

ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ
КРИПТОІНВЕСТИЦІЙ
студента Попова Дмитра Сергійовича

Науковий керівник:
к.е.н.,

Козак І.А.

Кваліфікаційний
бакалаврський проект
допущений до захисту в
Екзаменаційній комісії з
атестації здобувачів
вищої освіти

в.о. завідувача кафедри

Тішков Б.О.

Київ 2024