**Пояснювальна записка  
до дипломного проєкту**

на тему: Веб-застосунок для відстеження цін на криптоактиви з оповіщенням про зміну ціни

КПІ.ІТз-9104.045440.02.81

Київ – 2023

Зміст

[Вступ 5](#_Toc133360134)

[1 АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 7](#_Toc133360135)

[1.1 Загальні положення 7](#_Toc133360136)

[1.2 Змістовний опис і аналіз предметної області 8](#_Toc133360137)

[1.3 Аналіз існуючих технологій та успішних IT-проєктів 9](#_Toc133360138)

[1.3.1 Аналіз відомих алгоритмічних та технічних рішень 9](#_Toc133360139)

[1.3.2 Аналіз допоміжних програмних засобів та засобів розробки 12](#_Toc133360140)

[1.3.3 Аналіз відомих програмних продуктів 16](#_Toc133360141)

[1.4 Аналіз вимог до програмного забезпечення 16](#_Toc133360142)

[1.4.1 Розроблення функціональних вимог 17](#_Toc133360143)

[1.4.2 Розроблення нефункціональних вимог 18](#_Toc133360144)

[1.5 Постановка задачі 19](#_Toc133360145)

[Висновки до розділу 19](#_Toc133360146)

[2 МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 20](#_Toc133360147)

[2.1 Моделювання та аналіз програмного забезпечення 20](#_Toc133360148)

[2.2 Архітектура програмного забезпечення 21](#_Toc133360149)

[2.3 Конструювання програмного забезпечення 21](#_Toc133360150)

[2.4 Аналіз безпеки даних 24](#_Toc133360151)

[Висновки до розділу 24](#_Toc133360152)

[3 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 25](#_Toc133360153)

[3.1 Аналіз якості ПЗ 25](#_Toc133360154)

[3.2 Опис процесів тестування 25](#_Toc133360155)

[3.3 Опис контрольного прикладу 26](#_Toc133360156)

[Висновки до розділу 26](#_Toc133360157)

[4 ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 27](#_Toc133360158)

[4.1 Розгортання програмного забезпечення 27](#_Toc133360159)

[4.2 Підтримка програмного забезпечення 27](#_Toc133360160)

[Висновки до розділу 28](#_Toc133360161)

[ВИСНОВКИ 29](#_Toc133360162)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 30](#_Toc133360163)

[ДОДАТКИ 31](#_Toc133360164)

Перелік умовних позначень

|  |  |  |
| --- | --- | --- |
| IDE | – | Integrated Development Environment – інтегроване середовище розробки. |
| API | – | Application programming interface, прикладний програмний Інтерфейс |
| SDK | – | Software development kit |
| IT | – | Інформаційні технології |
| ER | – | Entity-Relation diagram |
| OC | – | Операційна система. |
| БД | – | База даних. |

Вступ

Світова економіка зацікавлена в криптовалютах останній час, привертаючи інтерес компаній, трейдерів та інвесторів. Однак ринок криптоактивів, як відомо, є непередбачуваним та високо волатильним, що створює дилему при прийнятті інвестиційних рішень. Відсутність актуальної інформації про ринок може призвести до інвестиційних помилок та втрат активів.

Зростаюча популярність криптовалют і сектор фінансових технологій, що постійно розвивається, стимулює попит на доступні веб-додатки, які стежують за цінами на цифрові активи і сприяють прийняттю обґрунтованих рішень. Гучні імена у сфері фінтеху та криптовалют активно намагаються вирішити цю проблему, але єдиного рішення, яке б задовольнило потреби всіх користувачів, не існує.

Різні сектори можуть отримати користь від розробки веб-додатків, які відстежують ціни на криптоактиви та надсилають сповіщення про їх зміну. Запропонований проект має на меті створити такий додаток, який допоможе користувачам відстежувати ціни на різні криптоактиви. Це може бути дуже корисно для індивідуальних та корпоративних інвестицій, торгівлі криптовалютою та аналізу ринку.

Індивідуальні та корпоративні інвестиції, аналітичні агентства, регулятори, що регулюють ринок криптоактивів та торгівлю криптовалютою - можливі сфери застосування розроблених веб-додатків. Легка інтеграція таких інструментів у робочі процеси дозволяє значно підвищити ефективність інвестицій і знизити ризик втрат. В кінцевому підсумку, вони сприяють більш швидкому і якісному прийняттю рішень в сфері торгівлі криптовалютою.

Що стосується моніторингу та аналізу криптовалют, то створення веб-додатку, який забезпечує відстеження цін на криптовалюту та оповіщення про коливання цін, є високоефективним заходом. Ця ініціатива є ключовою для оптимізації процесу аналізу ринку та підвищення якості прийняття рішень

# АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Загальні положення

У сучасному світі криптовалютні активи стали важливим елементом фінансових операцій, що відкриває нові можливості для інвестицій та заробітку. Щоб використовувати криптовалюти не потрібна ідентифікація, що дає змогу використовувати їх анонімно. В Україні можливість отримувати платежі від анонімних користувачів допомогає волонтерам на денних засадах. (К., 2023)

Ринок криптовалют характеризується високою волатильністю, що робить його привабливим для інвесторів, але водночас ускладнює прийняття рішень щодо купівлі та продажу активів. Отже, актуальним є розробка інструментів, які дозволяють відстежувати зміни цін криптоактивів в режимі реального часу та сповіщати користувачів про ці зміни.

Криптовалюта – це цифровий або віртуальний актив, який використовує криптографічні методи для забезпечення безпеки транзакцій, контролю створення нових одиниць та верифікації передачі активів та зберігається з використанням технології блокчейн. Прикладом таких криптографічних методів виступають алгоритми «Proof Of Work» (Jakobsson, 1999) або «Proof of Stake» (Bentov, 2012).

Криптовалютний ринок – це організований торгівельний майданчик, що займається укладанням між учасниками ринку угод з куплі та продажу криптоактивів та інших цифрових активів, що можуть бути пов’язані з блокчейн технологіями. Прикладами таких криптоактивів є «Binance», «CoinBase» та інші.

Блокчейн (S., 2008) – це розподілена база даних, що дозволяє передавати дані у відкритому вигляді та зростає зберігаючи порядок блоків. Гарантує надійність даних використання хешу блоку у наступному блоку, що дозволяє перевірити цепочку записів на дотримання правил.

Предметною областю даного проекту є програмне забезпечення для моніторингу криптоактивів, спрямоване на спрощення процесу прийняття рішень щодо купівлі та продажу криптовалют. Основні аспекти цієї предметної області включають збір та аналіз даних про криптовалютні активи, розробку користувацьких інтерфейсів, реалізацію систем сповіщень та забезпечення безпеки та конфіденційності інформації. Розвиток даної предметної області відбувається під впливом технологічних нововведень, таких як блокчейн-технології.

## Змістовний опис і аналіз предметної області

Наразі існує ряд ІТ-рішень, які спрямовані на відстеження цін криптоактивів, таких як мобільні додатки, веб-сайти, біржі та аналітичні платформи. Проте, ці рішення мають певні недоліки, зокрема:

* Недостатня швидкість реагування на зміни цін, що призводить до пропущених можливостей та втрат інвестицій.
* Обмежена кількість криптоактивів, які можуть бути відстежені.
* Відсутність індивідуальних налаштувань сповіщень та аналізу ринку.
* Проблеми з безпекою користувацьких даних та конфіденційність інформації.

Для поліпшення ситуації з ІТ-розробками у предметній області можна розглянути наступні шляхи:

* Застосування передових алгоритмів та технологій для швидкого відстеження змін цін криптоактивів.
* Розширення списку доступних криптовалют для відстеження та аналізу.
* Розробка зручного користувацького інтерфейсу з можливістю індивідуальних налаштувань сповіщень та аналізу ринку.
* Забезпечення безпеки користувацьких даних та конфіденційності інформаці ї за допомогою сучасних методів шифрування та аутентифікації.

У рамках даного дипломного проекту обрано шлях розробки веб-додатку для відстеження цін на криптоактиви з оповіщенням про зміну ціни. Вибір обумовлений наступними причинами:

* Веб-додатки є доступними для широкого кола користувачів, незалежно від їх операційної системи чи пристрою.
* Швидке відстеження змін цін криптоактивів та надсилання сповіщень дозволить користувачам вчасно реагувати на ринкові зміни.
* Врахування індивідуальних налаштувань сповіщень та аналізу ринку забезпечить користувачам гнучкість вибору та контроль над отриманою інформацією.
* Забезпечення безпеки користувацьких даних та конфіденційності інформації є важливим аспектом розвитку програмного забезпечення та довіри користувачів.

Отже, розробка веб-додатку, який відповідає потребам і вимогам користувачів у відстеженні цін на криптоактиви та отриманні сповіщень про зміну цін, дозволить забезпечити швидке та якісне прийняття рішень в галузі криптовалютного трейдингу.

## Аналіз існуючих технологій та успішних IT-проєктів

Проаналізуємо відоме на сьогодні алгоритмічне забезпечення у даній області та технічні рішення, що допоможуть у реалізації веб-застосунку для відстеження цін на криптоактиви з оповіщенням про зміну ціни. Далі будуть розглянуті допоміжні програмні засоби, засоби розробки та готові програмні рішення.

### Аналіз відомих алгоритмічних та технічних рішень

При розробці веб-додатку для відстеження цін на криптоактиви з оповіщенням про зміну ціни, важливо розглянути наявні алгоритми та архітектурні рішення, які допоможуть вирішити проблеми та задачі, пов'язані з цією розробкою.

#### Аналіз архітектури додатка

Серед технічних рішень ми розглянемо наступні:

1. Мікросервісна архітектура: ця архітектура передбачає розбиття системи на невеликі незалежні компоненти, які відповідають за окремі функції. Мікросервіси спрощують розробку, тестування та розгортання програмного забезпечення, а також дозволяють легко масштабувати систему.
2. Монолітна архітектура - це підхід до розробки програмного забезпечення, при якому всі компоненти системи об'єднуються в одному кодовому репозиторії та розгортаються як єдиний застосунок. У монолітній архітектурі різні частини системи, такі як фронтенд, бекенд, база даних та інші сервіси, взаємодіють безпосередньо один з одним. Це забезпечує простоту розробки, але може призвести до збільшення складності та меншої гнучкості при розширенні та масштабуванні системи.

#### Аналіз математичних алгоритмічних рішень

Наш додаток буде агрегувати дані з багатьох джерел даних, що додасть необхідність в обчисленні середніх значень. Серед доступних варіантів є наступні алгоритми:

1. Розрахунок середнього значення: цей алгоритм полягає в сумуванні всіх значень цін на криптоактив і поділі суми на кількість значень. Це найпростіший метод, але може бути вразливим до винятково високих або низьких значень, які можуть з'явитися через проблеми на окремих джерелах.
2. Розрахунок медіани: медіана - це середнє значення, яке розділяє відсортований набір даних на дві рівні частини. Цей алгоритм менш чутливий до винятково високих або низьких значень і надає більш стабільний результат.
3. Розрахунок квартилів: квартилі - це значення, які розділяють відсортований набір даних на чотири рівні частини. Вони використовуються для визначення розподілу даних та можуть допомогти виявити відхилення або аномалії в даних. Зазвичай використовують 0.25 та 0.75 квартиль, а також 0.5 квартиль, що і є медіаною.

#### Аналіз сортувальних алгоритмічних рішень

Сортувальні алгоритми відіграють важливу роль в обробці та аналізі даних, особливо в контексті нашого веб-додатку для відстеження цін криптовалют. Одним з прикладом використання сортування є рахування квартиля:

1. Швидке сортування (Quicksort (Knuth, 1998)) - це ефективний алгоритм, який використовує стратегію "розділяй та владарюй" для сортування даних. Він має середню часову складність O(n log n) та добре працює для великих наборів даних. Однак, в найгіршому випадку його складність може досягти O(n^2).
2. Порозрядне сортування (Radix sort (Knuth, 1998)) - це алгоритм, що працює з числовими значеннями і сортує дані за кожним розрядом числа починаючи з найменшого розряду. Цей алгоритм має лінійну складність O(nk) де n - кількість елементів, а k - кількість розрядів.. Порозрядне сортування є стабільним алгоритмом, але його ефективність залежить від розрядності чисел.
3. Злиттєве сортування (Merge sort (Knuth, 1998)) - це інший алгоритм, який використовує стратегію "розділяй та владарюй". Він рекурсивно розбиває набір даних на дві половини, сортує їх окремо та зливає відсортовані половини разом. Злиттєве сортування має стабільну часову складність O(n log n), але вимагає додаткового простору пам'яті.

#### Підсумок

Після порівняння алгоритмів та архітектур, ми вирішили вибрати мікросервісну архітектуру, оскільки вона спрощує розробку на бекенді і дозволяє швидку адаптацію до змін та масштабування системи.

З алгоритмічної точки зору, було вибрано квартилі та медіани, а не середнє значення, тому що ці алгоритми менш чутливі до проблем в окремих джерелах даних, що можуть вплинути на середнє значення.

Щодо алгоритму сортування, було вирішено використати порозрядне сортування. Цей алгоритм є стабільним та ефективним для роботи з числами, які і будуть основним набором даних.

Отже, було обрано мікросервісну архітектуру для спрощення розробки бекенду та алгоритми квартилів та медіан для точного відображення даних про ціни криптовалют. Порозрядне сортування використовується як оптимальний сортувальний алгоритм для обробки числових дан х в нашому веб-додатку.

### Аналіз допоміжних програмних засобів та засобів розробки

Проаналізуємо відоме на сьогодні допоміжні програмні засоби та засоби розробки, що допоможуть у реалізації веб-застосунку для відстеження цін на криптоактиви з оповіщенням про зміну ціни.

#### Аналіз допоміжних програмних засобів для Back-end частини

Для розкобки мікросервісів Back-End частини, необхідно визначити мову програмування та пакетний менеджер для швидкої та безпечної розробки. Ми розглянемо наступні популярні мови:

1. Rust – мова програмування відома своєю безпекою та ефективною роботою у високонагруженій середі. Rust пропонує систему контролю доступу до пам'яті, яка запобігає небезпечним ситуаціям, таким як зчитування неправильних даних та змагання за ресурси. Проте розробка на ній та поріг входу є вище чим у більш високорівневих аналогів
2. Go - мова програмування, розроблена Google, що відрізняється швидкістю роботи та простотою синтаксису. Є дуже популярною технологією за свою легкість розробки та швидку компіляцію.
3. Node.js - середовище виконання JavaScript, що дозволяє розробляти серверні додатки. Цей фреймворк дозволяє використовувати JavaScript технологію як для Front-End так і для Back-End.
4. PHP - популярна мова програмування для розробки веб-додатків. Останнім часом втрачає свою популярність, проте все ще пропонує зручну та легку взаємодію.

Вибір зупинився на Rust через його безпекові гарантії та продуктивність, що забезпечує вищу надійність розробленого веб-додатку, що є критично у сфері фінансів. Для комунікації між бекенд-сервісами буде використовуватися протокол gRPC (Wang, et al., 1993). Для встановлення додаткових залежностей буде використовувати пакетний менеджер cargo.

#### Аналіз допоміжних програмних засобів для Front-End частини

Для Front-End частини, необхідно також розглянути можливі варіанти. На жаль, на даний момент вибір є тільки серед JavaScript фреймворків, так як WebASM технології все ще не користуються великою користувацькою підтримкою, тому пропоную розглянути саме фреймворки. Серед який є наступні:

1. React - популярний фреймворк, що дозволяє створювати інтерактивні веб-додатки з легкими компонентами. Нажаль, має досить посередню швидкодію в порівнянні з конкурентами.
2. Angular - потужний фреймворк, розроблений Google, який підтримує структурований підхід до розробки. Останніми роками втрачає свою популярність на ринку.
3. Svelte - інноваційний фреймворк, який компілює код компонентів безпосередньо в оптимізований JavaScript, що покращує швидкість роботу додатку.

Було вибрано JavaScript з фреймворком React для розробки фронтенду через його широку підтримку, гнучкість та легкість інтеграції з різними бекенд-технологіями. Також буде використовуватися TypeScript для покращення безпекових якостей динамічної мови програмування JavaScript. Для передачі даних у режимі реального часу між бекендом та фронтендом будуть використовуватися веб-сокети та REST кінцеві точки. Для встановлення залежностей буде використовуватися npm.

#### Аналіз бази даних

В проекті буде використовуватись база даних, тому необхідно проаналізувати та обрати базу даних, що буде мати високу швидкодію та безпекові гарантії. Для зберігання та обробки даних ми розглядаємо такі системи баз даних, як PostgreSQL, Microsoft SQL Server та MongoDB:

1. PostgreSQL - відкрита система об'єктно-реляційної бази даних з високою продуктивністю та надійністю, підтримкою розширеного набору вбудованих функцій та активною спільнотою.
2. Microsoft SQL Server - комерційна система реляційної бази даних з відмінною продуктивністю, безпекою та інтеграцією з іншими продуктами Microsoft, але з обмеженнями щодо платформи та ліцензійних умов.
3. MongoDB - відкрита система баз даних NoSQL з орієнтацією на документи, що пропонує гнучкість та масштабованість, але може бути менш підходящою для структурованих даних та реляційних запитів.

Було обрано PostgreSQL як систему баз даних для нашого проекту через її високу продуктивність, надійність, підтримку розширеного набору вбудованих функцій та активну спільноту. Також ця система не має обмежень щодо ліцензування та не потребує використання Windows системи, що дає можливість гнучкої розробки. Ці фактори забезпечують оптимальне рішення для обробки та зберігання даних про криптоактиви та оповіщення про зміну ціни.

#### Аналіз засобів розробки

Для успішної розброки та тестування програмного забеспечення виникає потреба в використанні інтегрованих середовищ розробки. Ми розглянемо такі середовища програмування, як Visual Studio Code, Sublime Text та Jetbrains IDE.

1. Visual Studio Code (VSCode) - легкий, але потужний текстовий редактор від Microsoft з широким набором розширень для різних мов програмування та технологій.
2. Sublime Text - швидкий текстовий редактор з гнучкими налаштуваннями та підтримкою багатьох мов програмування.
3. Jetbrains IDE - потужні комерційні середовища розробки від JetBrains, що має вбудовану підтримку для різних мов програмування та фреймворків, але може бути дещо важким для системи.

Ми вибрали VSCode через його легкість, широку підтримку мов програмування та гнучкість налаштувань. Також важливим фактором є безкоштовність програмного забезпечення. Для керування залежностями буде використовуватися система Nix.

#### Підсумок

У підсумку, для реалізації нашого веб-додатку ми обрали наступні технології та інструменти:

Мова програмування Rust для бекенду через її безпечність, продуктивність та надійність.

gRPC для комунікації між мікросервісами, що забезпечує високу продуктивність та сумісність.

Websocket для передачі потокових даних на клієнтську сторону.

JavaScript та React для розробки фронтенду, оскільки вони дозволяють створювати швидкі та гнучкі інтерфейси користувача.

Nix для керування залежностями, що дозволяє просте та послідовне використання пакетів і середовищ.

PostgreSQL як систему баз даних через її високу продуктивність, надійність та підтримку розширеного набору вбудованих функцій.

Visual Studio Code як редактор коду, завдяки його легкості, широкій підтримці мов програмування та розширюваності.

Вибір цих технологій базується на їх перевагах, а також на порівняльному аналізі з альтернативними рішеннями. Таким чином, ми забезпечуємо розвиток ефективного, безпечного та легко супроводжуваного веб-додатку для відстеження цін на криптоактиви з оповіщеннями про зміну ціни.

### Аналіз відомих програмних продуктів

<У пункті викладають:

опис готових програмних продуктів за вашою предметною областю, які частково чи повністю реалізують функціонал описаний у технічному завданні. Обов’язково має бути порівняльна таблиця функціоналу та особливостей з вашою розробкою.>

Для порівняння курсової роботи з аналогом можна скористатись таблицею 1.3.

Таблиця 1.3 – Порівняння з аналогом

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Функціонал | Дипломний проєкт(назва) | Аналог для порівняння 1 | Аналог для порівняння n | Пояснення |
|  |  |  |  |  |

## Аналіз вимог до програмного забезпечення

<У підрозділі викладають:

опис головного(-х) функціоналу розробки, діаграму варіантів використання та опис варіантів використання.>

Головною функцією програмного забезпечення є …, більше функцій можна побачити на рисунку 1.3.

Рисунок 1.3 – Діаграма варіантів використання

В таблицях 1.2 - 1.17 наведені варіанти використання програмного забезпечення.

Таблиця 1.2 - Варіант використання UC-1

|  |  |
| --- | --- |
| Use case name | Реєстрація користувача |
| Use case ID | UC-01 |
| Goals | Реєстрація нового користувача в системі |
| Actors | Гість (незареєстрований користувач) |
| Trigger | Користувач бажає зареєструватися |
| Pre-conditions | - |
| Flow of Events | Користувач переходить на сторінку реєстрації. В поля для реєстрації вводяться відповідні дані: пошта користувача, пароль в системі, та його повтор для підтвердження, а також чек бокс для підтвердження умов сервісу. Після заповнення даних користувача натискає кнопку реєстрації. Після цього з’являється повідомлення про успішну реєстрацію, і користувач перенаправляється на сторінку входу. |
| Extension | В випадку введення не коректних даних, кнопка реєстрації стає неактивною. Якщо якесь конкретне поле введено некоректно, то воно підсвічується червоним надписом. |
| Post-Condition | Створення сторінки користувача, перехід на сторінку входу |

### Розроблення функціональних вимог

<У пункті викладають:

модель вимог, їх опис та матрицю трасування.>

Програмне забезпечення розділене на модулі. Кожен модуль має свій певний набір функцій. На рисунку 1.4 наведено загальну модель вимог, а в таблицях 1.18 – 1.27 наведений опис функціональних вимог до програмного забезпечення. Матрицю трасування вимог можна побачити на рисунку 1.5.

Рисунок 1.4 – Модель вимог у загальному вигляді

Таблиця 1.16 – Функціональна вимога FR-1

|  |  |
| --- | --- |
| Назва | Реєстрація користувача |
| Опис | Система повинна надавати можливість реєстрації користувачеві шляхом введення пошти, паролю, підтвердження паролю. |

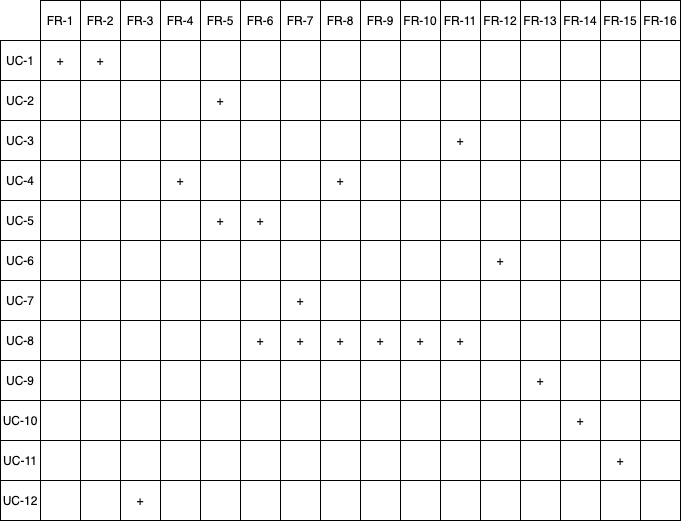


Рисунок 1.11 – Матриця трасування вимог

### Розроблення нефункціональних вимог

<У пункті викладають:

опис нефункціональних вимог.>

## Постановка задачі

<У підрозділі викладають:

що саме ви збираєтесь реалізувати. Це не зовсім вимоги і варіанти використання. Це достатньо узагальнене поняття з метою, цілями та задачами, що підлягають розв’язанню у результаті розробки програмного забезпечення на дипломний проєкт.>

## Висновки до розділу

< Необхідно стисло описати усе, що було виконано у даному розділі. Обсяг 0,75-1 сторінка>

# МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Моделювання та аналіз програмного забезпечення

<У підрозділі викладають:

опис бізнес процесів вашої розробки та їх наочне представлення за допомогою засобів моделювання (BPMN та інші).>

Для опису бізнес процесу програмного забезпечення використовується BPMN модель (рисунок 2.1).

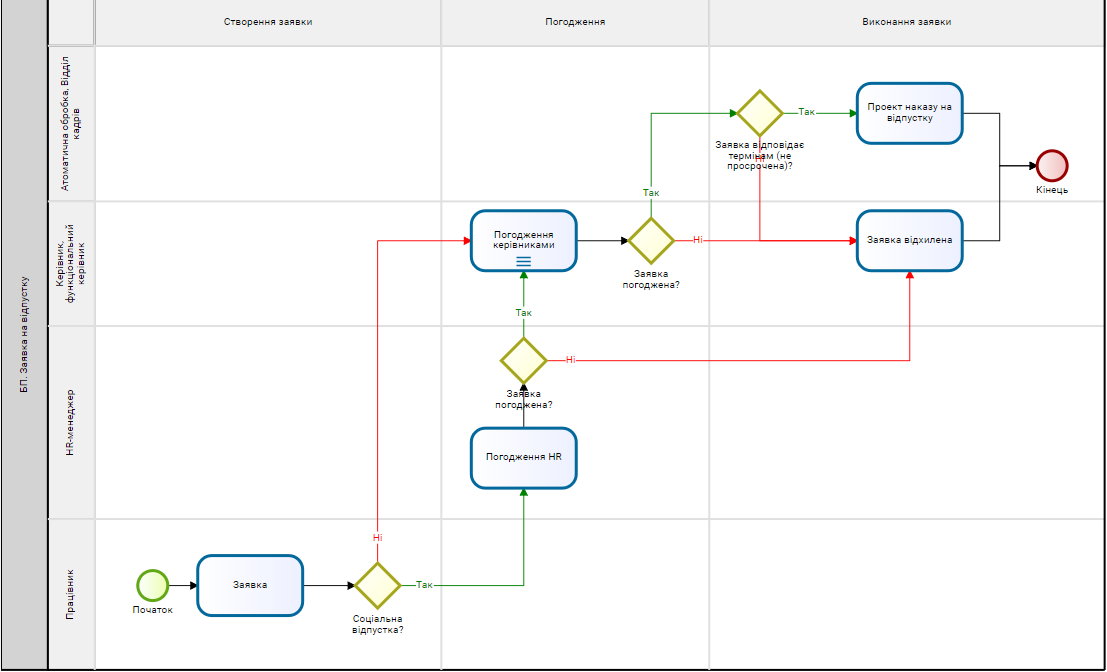


Рисунок 2.1 – Схема бізнес-процесу

Опис послідовності створення облікового запису користувача:

* користувач переходить на сторінку реєстрації;
* користувач заповнює поля реєстрації;
* якщо введені поля, не відповідають шаблону заповнення на клієнтській стороні, відповідні поля підсвічуються помилкою;

## Архітектура програмного забезпечення

<У підрозділі викладають:

втілення архітектурного паттерну для вашого ПЗ та деталізація його компонентів за необхідності. Необхідно виконати повний опис архітектури (схеми, таблиці та ін.).>

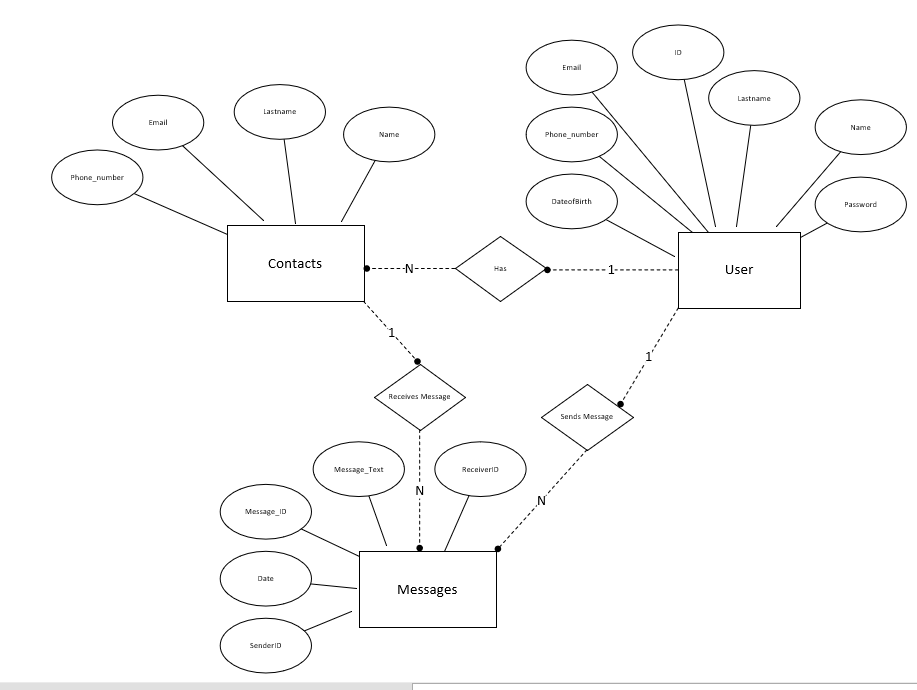


Рисунок 2.2 – ER діаграма сутностей User, Messages, Contacts

## Конструювання програмного забезпечення

<У підрозділі викладають:

опис оригінальних алгоритмів чи модифікацій існуючих. Опис структур даних, програмних структур та ін. Опис бази даних з представленням концептуальної, логічної чи фізичної моделі та з описом сутностей чи таблиць. Опис утиліт, бібліотек та іншого стороннього програмного забезпечення, що використовується у розробці. Аналіз системних вимог.>

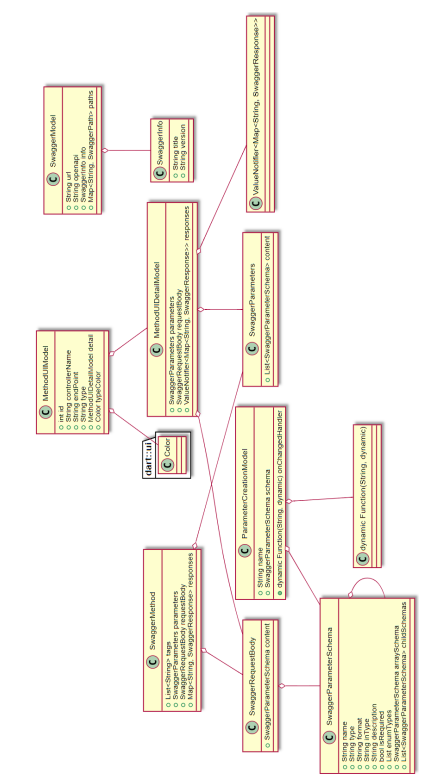


Рисунок 2.3 – Схема структурна класів

Бібліотека xpring, яка виконує запити в мережу блокчейн – однопоточна, тобто не може оброблювати паралельні запити. Через таку специфіку бібліотеки, довелось розробляти рішення яке б не блокувало запити з різних потоків, або інших частин коду. За основу ідеї алгоритму була взята багатопоточність самої мови програмування java. Java для вирішення проблем з потоками представляє ділянки коду які можна синхронізувати, що унеможливлює одночасний доступ до нього з різних потоків. Ця синхронізація проходить за допомогою передачі управління об’єкта-монітора. Алгоритм вирішення проблеми з доступом до однопоточної бібліотеки наведено на рисунку 2.4.

В якості системи управління базами даних використовується Postgres. База даних серверу призначена для зберігання користувачів, а також даних про їх …. Опис таблиць бази даних наведено у таблицях 2.11 - 2.14. Модель бази даних наведена на рисунку 2.12.

Таблиця 2.11 – Опис таблиці user

|  |  |  |  |
| --- | --- | --- | --- |
| Таблиця | Назва поля | Тип даних | Опис |
| user | id | serial | ідентифікаційний номер користувача |
| email | varchar | електронна пошта користувача |
| password\_id | int | посилання на запис у таблиці password, де зберігається пароль користувача |

Опис утиліт, бібліотек та іншого стороннього програмного забезпечення, що використовується у розробці наведено в таблиці 2.22.

Таблиця 2.22 – Опис утиліт

|  |  |  |
| --- | --- | --- |
| №  п/п | Назва утиліти | Опис застосування |
| 1 | IntelliJ IDEA | Головне середовище розробки програмного забезпечення серверної частини курсової роботи. |
| 2 | Postman | Програмне забезпечення необхідне для тестування rest запитів. Використовувалось для тестування API інтерфейсів, та клієнтських запитів. |
| 3 | MySQL Workbench | Програмне забезпечення яке надає легкий графічний інтерфейс для доступу до бази даних. |

## Аналіз безпеки даних

<У підрозділі викладають:

аналіз вразливостей ПЗ та будь-які питання пов’язані з безпекою даних.>

## Висновки до розділу

< Необхідно стисло описати усе, що було виконано у даному розділі. Обсяг 0,75-1 сторінка>

# АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Аналіз якості ПЗ

<У підрозділі викладають:

аналіз якості ПЗ за певними метриками.>

## Опис процесів тестування

<У підрозділі викладають:

опис процесів тестування та приклади тестів.>

Було виконане мануальне тестування програмного забезпечення, опис відповідних тестів наведено у таблицях 3.3 – 3.30.

Таблиця 3.3 – Тест 1.1

|  |  |
| --- | --- |
| Тест | Реєстрація користувача |
| Модуль | Реєстрація користувача |
| Номер тесту | 1.1 |
| Початковий стан системи | Користувач знаходиться на сторінці реєстрації |
| Вхідні данні | Електронна пошта, пароль, підтвердження паролю |
| Опис проведення тесту | У відповідні поля вводяться: коректна електронна пошта, яка до цього не була зареєстрована в системі, пароль від 10 до 64 символів, який містить хоча б з одну англійську літеру, одне число і один спеціальний символ, і який не входить у топ 10000 найпопулярніших паролей, підтвердження паролю, яке співпадає з раніше введеним паролем. Після цього натискається кнопка підтвердження реєстрації. |

Продовження таблиці 3.4

|  |  |
| --- | --- |
| Очікуваний результат | Реєстрація проходить успішно, користувач додається у систему і перенаправляється на сторінку авторизації. |
| Фактичний результат | Реєстрація проходить успішно, користувач додається у систему і перенаправляється на сторінку авторизації. |

## Опис контрольного прикладу

<У підрозділі викладають:

повний опис контрольного прикладу з усіма можливими розгалуженнями та особливостями. Кроки доповнюють ілюстраціями.>

## Висновки до розділу

< Необхідно стисло описати усе, що було виконано у даному розділі. Обсяг 0,75-1 сторінка>

# ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Розгортання програмного забезпечення

<У підрозділі викладають:

повний опис покрокового розгортання ПЗ. Кроки доповнюють ілюстраціями.>

Клієнтську і серверну частини програмного забезпечення було вирішено розгорнути на платформі Heroku. Для розгортання було використано сервіс GitHub Actions, який надає можливості для постійної інтеграції і розгортання [13].

Розгортання починається коли новий код застосунку доставляється у репозиторій у гілку main. Тоді у середовищі GitHub Actions створюється Docker image за допомогою Dockerfile, що знаходиться у проекті. Цей image розгортається у Heroku за допомогою пакету heroku-deploy. Інформацію про розгортання клієнтської і серверної частини проекту можна побачити на рисунках 4.1 і 4.2.

Рисунок 4.1 - Інформація про розгортання клієнту

## Підтримка програмного забезпечення

<У підрозділі викладають:

опис того, як буде виконуватись підтримка програмного забезпечення. Ілюстрації.>

Користувачі повинні мати можливість отримати нову версію консольного застосунку з кожною версією. До того ж кожна нова версія консольного застосунку повинна бути опублікована в npm. Для автоматизації цього процесу був використаний сервіс GitHub Actions.

Створення нового випуску починається, коли нова версія консольного застосунку доставляється у репозиторій у гілку main, тобто коли commit має tag формату “v\*.\*.\*.”, де замість “\*” знаходиться число. Тоді у середовищі GitHub Actions встановлюється NodeJS. Після цього для проекту встановлюються залежності і проект збирається. Bash скрипт за допомогою бібліотеки pkg генерує виконувані файли (executables) для Linux і для Windows, та пакує файл для Linux у .deb пакет. Після цього .deb пакет і файл для Windows архівуються,

## Висновки до розділу

< Необхідно стисло описати усе, що було виконано у даному розділі. Обсяг 0,75-1 сторінка>

# ВИСНОВКИ

У висновках викладають найважливіші наукові й практичні результати роботи та наводять:

* оцінку одержаних результатів і їх відповідність сучасному рівню наукових і технічних знань;
* ступінь впровадження та можливі галузі або сфери використання результатів роботи;
* наукову, науково-технічну, соціально-економічну значущість роботи;
* доцільність продовження досліджень за відповідною тематикою тощо.

Також у висновках необхідно відобразити стан вирішення усіх поставлених в дипломному проєктуванні задач.

В результаті виконання дипломного проєкту було спроєктовано …

В якості середовища розробки обрано …

У якості БД використано …

Після реалізації застосунку він був протестований на пристроях з різними версіями Android, з різними розмірами екранів щоб переконатися, що додаток акуратно відображається на різних пристроях.

Наукова новизна роботи (якщо вона є) полягає в наступному (достатньо вказати щось одне).

Вперше:

* реалізовано можливість запитів від пацієнта до лікаря;
* використано те-то, що дозволило те-то.

Модифіковано:

* те-то, що дозволило те-то.

Набуло подальший розвиток:

* те-то, що дозволило те-то.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

**Bentov I., Gabizon, A., Mizrahi, A.** Cryptocurrencies Without Proof of Work [Книга]. - 2012.

**Jakobsson M., Juels, A.** Proofs of Work and Bread Pudding Protocols [Книга]. - 1999.

**Knuth Donald** The art of computer programming: Volume 3: Sorting and Searching [Книга]. - 1998. - сс. 168-179.

**Knuth Donald** The art of computer programming: Volume 3: Sorting and Searching [Книга]. - 1998. - сс. 113-122.

**Knuth Donald** The art of computer programming: Volume 3: Sorting and Searching [Книга]. - 1998. - сс. 158-168.

**S. Nakamoto** Bitcoin: A peer-to-peer electronic cash system [Книга]. - 2008.

**Wang Xingwei, Hong Zhao та Jiakeng Zhu** GRPC: A communication cooperation mechanism in distributed systems [Книга]. - 1993.

**К. Хомʼяков** Криптовалюта під час війни: миттєві донати та блокчейн-версія "Плану Маршалла" [Онлайновий] // liga.net. - 13 04 2023 p.. - 23 04 2023 p.. - https://www.liga.net/ua/economics/opinion/kriptovalyuta-vo-vremya-voyny-mgnovennye-donaty-i-blokcheyn-versiya-plana-marshala.

# ДОДАТКИ