**Пояснювальна записка  
до курсової роботи**

на тему: Тема курсової роботи

КПІ.ІХ-0XXX.045440.02.81

Київ – 2022

Зміст

[Перелік умовних позначень 4](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992741)

[Вступ 5](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992742)

[1 АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 6](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992743)

[1.1 Загальні положення 6](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992744)

[1.2 Змістовний опис і аналіз предметної області 6](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992745)

[1.3 Аналіз існуючих технологій та успішних IT-проєктів 6](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992746)

[1.3.1 Аналіз відомих алгоритмічних та технічних рішень 6](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992747)

[1.3.2 Аналіз допоміжних програмних засобів та засобів розробки 7](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992748)

[1.3.3 Аналіз відомих програмних продуктів 7](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992749)

[1.4 Аналіз вимог до програмного забезпечення 7](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992750)

[1.4.1 Розроблення функціональних вимог 8](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992751)

[1.4.2 Розроблення нефункціональних вимог 9](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992752)

[1.5 Постановка задачі 9](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992753)

[Висновки до розділу 10](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992754)

[2 МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 11](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992755)

[2.1 Моделювання та аналіз програмного забезпечення 11](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992756)

[2.2 Архітектура програмного забезпечення 11](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992757)

[2.3 Конструювання програмного забезпечення 11](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992758)

[2.4 Аналіз безпеки даних 13](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992759)

[Висновки до розділу 13](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992760)

[3 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 14](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992761)

[3.1 Аналіз якості ПЗ 14](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992762)

[3.2 Опис процесів тестування 14](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992763)

[3.3 Опис контрольного прикладу 15](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992764)

[Висновки до розділу 15](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992765)

[4 ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 16](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992766)

[4.1 Розгортання програмного забезпечення 16](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992767)

[4.2 Підтримка програмного забезпечення 16](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992768)

[Висновки до розділу 17](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992769)

[ВИСНОВКИ 18](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992770)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 19](file:///C:\Users\User\Downloads\Telegram%20Desktop\КР_шаблон_ч2_ПЗ_2023_.docx#_Toc115992771)

Перелік умовних позначень

|  |  |  |
| --- | --- | --- |
| ККІ | – | Колекційна карткова гра |
| ШІ | – | Штучний інтелект |

Вступ

Карткові ігри, беручи свій початок у глибині століть, завжди були популярними серед різних верств населення. В сучасному світі, з розвитком технологій, карткові ігри не лише не втратили своєї значимості, а й отримали можливість ще більше розкрити свій розважальний потенціал. В наш час існують безліч варіацій карткових ігор, серед яких кожен може знайти розвагу собі до душі. Особливий інтерес викликають колекційні карткові ігри, які поєднують в собі елементи стратегії, креативності, колекціонування та змагальності. Найчастіше карткові ігри призначені для гри у групі, це може становити собою велику проблему для людей, що не можуть знайти собі партнера для сумісної гри. Впровадження штучного інтелекту (ШІ), що імітує гравця, у такі ігри відкриває нові горизонти для гри наодинці.

Розвиток карткових ігор пройшов значний шлях від традиційних фізичних колод до сучасних цифрових форматів. Якщо раніше карткові ігри асоціювалися переважно з фізичними колодами, які гравці могли торкатися, мішати та розкладати на столі, то сьогодні все більша кількість цих ігор переходить у цифровий простір. Цей перехід відбувається на тлі загальної цифровізації розваг і відкриває нові можливості для інтерактивності, доступності та дозволяє розробникам частіше випускати оновлення для своїх ігор. Світова тенденція у сфері колекційних карткових ігор (ККІ) відзначається активною участю великих ігрових компаній, які створюють та підтримують свої ККІ. Це спостерігається у діяльності таких відомих компаній як Blizzard Entertainment з їхньою відомою грою "Hearthstone", Wizards of the Coast з легендарною "Magic: The Gathering", а також інших гравців на ринку, які розширюють уявлення про можливості та горизонти ККІ. Ці компанії, використовуючи свої ресурси та досвід, вносять значний вклад у розвиток жанру, постійно оновлюючи геймплей, вводячи нові механіки та елементи, що забезпечує зростання популярності ККІ серед широкої аудиторії гравців у всьому світі.

Ця курсова робота спрямована на створення колекційної карткової гри. Дослідження алгоритмів ігрових ШІ, створення ШІ, що імітуватиме дії суперника.

# АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Загальні положення

<У підрозділі викладають:

загальні положення по предметній області. Почати варто з загального огляду предметної області, загальних визначень по ній, напрямків розробок, тощо. Допускається навести загальну теорію. Обов’язково надати посилання на джерела інформації.[1]. Обсяг 3 стр., чи більше>

Колекційні карткові ігри (ККІ)[1] — різновид настільних (або картярських) ігор, для гри в які використовуються спеціально розроблені набори карток. Ці картки, крім гральної, мають також самостійну колекційну цінність: вони продаються й обмінюються серед збирачів, через що власне і виникла назва «Колекційні карткові ігри». Хоча колекційні картки відомі дуже давно, логічні ігри з використанням таких карток з'явилися лише на початку 1990-х років, і дуже швидко набули популярності. Однією з найперших та найуспішніших ККІ є гра «Magic: The Gathering», яка вважається родоначальницею всього цього класу ігор.

ККІ дуже близькі до настільних рольових ігор. Сюжети цих ігор базуються або на реальних спортивних змаганнях (автоперегони, баскетбол, бейсбол, футбол, хокей та інші популярні види спорту), або на уявних світах (це можуть бути будь-які сполучення фентезі, наукової фантастики, містики та інших пригодницьких жанрів). В останньому випадку сюжет зазвичай запозичується з відомих книг, коміксів, фільмів, а може й популярних комп'ютерних ігор (як-от «Pokemon»), та інколи це можуть бути і цілком оригінальні сюжети. Яскравим прикладом є гра «Magic: The Gathering», яка сама стала джерелом численних книжних новелізацій, супутніх товарів і різноманітних переспівів.

Кожна картка ККІ має власну назву, крім того, несе на собі оригінальну ілюстрацію та/або художній текст, а також може мати у грі відповідні спеціальні властивості. Усі ці елементи служать для підтримання атмосфери гри, що дасть гравцю змогу краще вжитися у свою роль, і повинні відповідати сюжету, на якому побудовано гру. Так картка, що має назву «Shivan Dragon», несе на собі зображення крилатого ящіркоподібного створіння, текст такого змісту: “В той час як більшість драконів просто жорстокі, Шиванський Дракон схоже отримує задоволення від страждань інших, часто розважаючись зі своєю жертвою, подібно тому як кіт грає з мишею, перш ніж завдати фінального удару” і має здібності польоту та вогняного дихання.

Колекційні карткові ігри — це досить складні стратегічні ігри. У грі беруть участь двоє (інколи і більше) гравців. На відміну від традиційних карткових ігор, таких як бридж, преферанс або покер, де всі гравці грають однією спільною колодою, кількісний та якісний склад якої обмежений і визначений заздалегідь, в ККГ кожен з гравців має власну колоду карток, склад якої він визначає сам, вибираючи на свій смак з доступного йому набору карток. Тому в грі майже ніколи не побачиш двох однакових колод. Складання колоди — вкрай важливий елемент підготовки до гри, чи не важливіший за сам процес гри. Це дозволяє гравцю практично нескінченно варіювати стратегію гри, використовуючи свої улюблені комбінації карток, способи їх взаємодії, та інші елементи гри.

Кожна система ККІ має основний набір правил, які описують цілі гравців, типи карток, що використовуються у грі, та базові принципи взаємодії цих карток. Кожна картка може мати додатковий текст, що визначає особливі ефекти від цієї картки у грі.

Практично всі ККІ побудовані на використанні якоїсь системи ресурсів. Ресурси можуть самі бути певного типу картками гри, або визначатися іншими засобами (такими як особливі фішки, лічильники якогось резерву ресурсів, відмітки на картках тощо). Використання ресурсів є загальним методом контролювання темпу гри. Ресурси також служать для встановлення балансу відносної сили карток. Потужніші картки або ефекти потребують більше ресурсів для їх розігрування. Також контроль над темпом може здійснюватися потоком карток, що входять та виходять з гри (приміром, щоб ввести в гру сильну картку, треба вивести з неї одну чи дві менш сильні).

Під час гри зазвичай гравці по черзі роблять ходи, розігруючи свої картки та виконуючи пов'язані з грою дії. Хід гравця можна розділити на окремі кроки, на кожному з яких виконуються свої специфічні види ігрових дій. Порядок і назви цих кроків відрізняються у різних системах ККІ, однак типовою є така послідовність:

* Відновлення *(Restore)* — усі картки в грі приводяться у стан готовності до наступного ходу (може бути як на початку, так і наприкінці ходу гравця).
* Підготовка *(Standby)* — виконуються всі ефекти, обумовлені картками, що на цей час знаходяться у грі.
* Взяття карток *(Draw cards)* — беруться картки з колоди, для забезпечення циркуляції карток у руці гравця.
* Розігрування карток *(Play cards)* — картки з руки гравця вводяться у гру, впливаючи на ігрову ситуацію (цей крок може виконуватися двічі, до і після фази конфлікту).
* Конфлікт *(Conflict)* — виконуються основні дії, направлені на досягнення перемоги у грі; найпоширенішим методом таких дій є бій *(combat)*.
* Скидання карток *(Discard cards)* — у більшості ККІ максимальна кількість карток, що можуть знаходитися в руці гравця, обмежена, або необхідно їх оновити для наступного ходу.

<У підрозділі викладають:

процес використання знань предметної області у програмному забезпеченні на поточний момент розвитку ІТ-технологій. Фактично описують процес імплементації предметної області у ІТ as-is. Описують недоліки поточного стану речей з предметною областю у сфері ІТ. Описати можливі шляхи покращення ситуації з розробками у сфері ІТ за вашою предметною областю. Вказати який шлях обрали ви в рамках курсової роботи. Обсяг 1 стр., чи більше >

## Аналіз існуючих технологій та успішних IT-проєктів

Проаналізуємо відоме на сьогодні алгоритмічне забезпечення у даній області та технічні рішення, що допоможуть у реалізації <вашої розробки, вказати назву>. Далі будуть розглянуті допоміжні програмні засоби, засоби розробки та готові програмні рішення. <Обсяг 5 стр., чи більше>

### Аналіз відомих алгоритмічних та технічних рішень

<У пункті викладають:

відомі алгоритми для розв’язання задач чи підзадач у вашій розробці, в кінці робиться порівняльний аналіз алгоритмів та обирається той, який ви будете використовувати у розробці. Або робиться висновок про необхідність розробки оригінальних алгоритмів чи модифікацію існуючих. Розглядаються відомі технічні рішення, які допоможуть у реалізації розробки: архітектурні паттерни, платформи, тощо. Для технічних рішень також робиться порівняльний аналіз та обираються ті, що будуть використані у розробці. Можливо зробити висновок про розробку оригінального технічного рішення, чи модифікацію існуючого.>

### Аналіз допоміжних програмних засобів та засобів розробки

<У пункті викладають:

опис допоміжних програмних засобів та їх порівняльний аналіз, якщо існує кілька аналогів (мова йде про сторонні бібліотеки, пакети, фреймворки, тощо). Розглядаються засоби розробки за допомогою яких можна виконати вашу розробку, робиться їх порівняльний аналіз та на його основі обираються ті, що вам підходять (маються на увазі мови програмування та IDE).>

### 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. Обсяг 5 стр., чи більше >

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. |

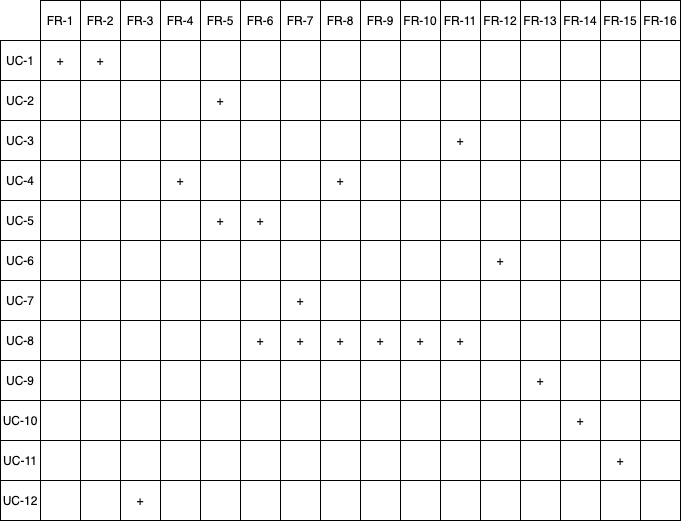


Рисунок 1.11 – Матриця трасування вимог

### 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 курсову роботу. Обсяг 1 сторінка, чи більше. >

## 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). Обсяг 2 сторінки, чи більше. >

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

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.). Обсяг 2 сторінки, чи більше.>

## 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. Обсяг 4 сторінки, чи більше.>

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 2.3.

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. Обсяг 2 сторінки, чи більше.>

## Опис процесів тестування

<У підрозділі викладають:

опис процесів тестування та приклади тестів. Обсяг 4 сторінки, чи більше.>

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 … |
| 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. Не обов’язковий розділ.>

## 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. Обсяг 2 сторінки, чи більше.>

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. Обсяг 1 сторінка, чи більше.>

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 курсовій роботі задач.

В результаті виконання курсової роботи 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

1. [Оформлення бібліографії ДСТУ 7.1:2006](https://drive.google.com/file/d/1VUr7fwKHOk8t1u8sh-Sv2dlEc_6OBoIS/view?usp=sharing)
2. [Оформлення бібліографії ДСТУ 8302-2015](https://drive.google.com/file/d/17RYEYVnSXolL0S8D1k85VfReYd2yC6WO/view?usp=sharing) з прикладами