Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА
на тему:
«МЕТОД ТА ЗАСІБ АВТЕНТИФІКАЦІЇ КОРИСТУВАЧІВ НА ОСНОВІ ТОКЕНІВ»

Виконав: студент 2–го курсу, групи 1БС–21м
спеціальності 125 – Кібербезпека
(шифр і назва напряму підготовки, спеціальності)
_____ А. М. Лоборчук
(прізвище та ініціали)
Керівник к. т. н., доц. каф. ЗІ
_____ Ю. В. Баришев
(прізвище та ініціали)
Опонент: к. т. н., доц. каф . ПЗ
_____ Н. П. Бабюк
(прізвище та ініціали)
«22» грудня 2022 р.

Допущено до захисту
Завідувач кафедри ЗІ
д. т. н., проф.
_____ В. А. Лужецький
«22» грудня 2022 р.

Вінниця ВНТУ – 2022 року

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань – 12 "Інформаційні технології"
Спеціальність – 125 "Кібербезпека"
Освітньо-професійна програма – Безпека інформаційних і комунікаційних систем

ЗАТВЕРДЖУЮ
Завідувач кафедри ЗІ,
д.т.н., проф.
В.А. Лужецький
«15» вересня 2022 року

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Лоборчуку Андрію Михайловичу

1. Тема роботи "Метод та засіб автентифікації користувачів на основі токенів" керівник роботи Баришев Юрій Володимирович, к. т. н., доц. каф. ЗІ затверджені наказом вищого навчального закладу від 14 вересня 2022 року № 203
2. Строк подання студентом роботи 19 грудня 2022 року
3. Вихідні дані до роботи:
– вид автентифікації - віддалена;
– забезпечення стійкості до перехоплення токена;
– створення блокових тестів до засобу.
4. Зміст текстової частини: Вступ. 1. Аналіз методів автентифікації користувачів. 2. Метод автентифікації віддалених користувачів. 3. Засіб автентифікації. 4. Тестування та експериментальне дослідження. 5. Економічна частина. Висновки. Перелік використаних джерел. Додатки.
5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень)

Порівняльний аналіз методів автентифікації (плакат, А4). Структура токена (плакат, А4). Протокол обміну токенами (плакат, А4). Алгоритм роботи методу автентифікації (плакат, А4). Алгоритм роботи програмної реалізації (плакат, А4). Результат блокового тестування (плакат, А4). Результати інтеграційного тестування (плакат, А4).

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	вико прий
1	Баришев Ю. В., к. т. н., доц. каф. ЗІ		
2	Баришев Ю. В., к. т. н., доц. каф. ЗІ		
3	Баришев Ю. В., к. т. н., доц. каф. ЗІ		
4	Баришев Ю. В., к. т. н., доц. каф. ЗІ		
5	Лесько О. Й. к.е.н., доц., професор каф. ЕПВМ		

7. Дата видачі завдання 1 вересня 2022 року

КАЛЕНДАРНИЙ ПЛАН

з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи
1	Аналіз завдання. Вступ	01.09.2022 – 04.09.2022
2	Аналіз інформаційних джерел за напрямком магістерської кваліфікаційної роботи	05.09.2022 – 04.10.2022
3	Аналіз та формування вимог до ПЗ	05.10.2022 – 16.10.2022
4	Розробка рішень, алгоритмів	17.10.2022 – 14.11.2022
5	Тестування розробленого ПЗ	15.11.2022 – 17.11.2022
6	Розробка розділу економічного обгрунтування доцільності розробки	18.11.2022 – 21.11.2022
7	Оформлення пояснювальної записки	22.11.2022 – 29.11.2022
8	Перевірка магістерської роботи на наявність плагіату	30.11.2022 – 06.12.2022
9	Попередній захист та доопрацювання МКР	07.12.2022 – 19.12.2022
10	Представлення МКР до захисту, рецензування	20.12.2022 – 21.12.2022
11	Захист МКР	22.12.2022 – 26.12.2022

Студент _____ (підпис) А. М. Лоборчук

Керівник роботи _____ (підпис) Ю. В. Баришев

АНОТАЦІЯ

УДК 004.056

Лоборчук А. М. Метод та засіб автентифікації користувачів на основі токенів. Магістерська кваліфікаційна робота зі спеціальності 125 – кібербезпека, освітня програма – Безпека інформаційних і комунікаційних систем. Вінниця: ВНТУ, 2022. 102 с.

Укр. мовою. Бібліогр.: 22 назв; рис.: 35; табл. 13.

Магістерська кваліфікаційна робота присвячена розробці методу та засобу автентифікації користувачів із використанням токенів. Здійснено аналіз сучасних методів автентифікації користувачів та синтезовано математичний опис процесу автентифікації. Розроблено метод автентифікації віддалених користувачів із використанням токенів. Спроектовано та реалізовано програмний засіб, що дозволяє здійснити автентифікацію користувача за розробленим методом. Виконано експериментальне дослідження методу та засобу автентифікації користувачів. Наведено економічне обґрунтування доцільності використання розробленого в роботі засобу.

Ключові слова: *автентифікація користувачів, мобільний застосунок, токен, віддалені користувачі, електронно–цифровий підпис, багатофакторна автентифікація*.

ABSTRACT

Loborchuk A. M. Method and tool for user authentication based on tokens. Master's thesis in specialty 125 – cybersecurity, educational program – Security of information and communication systems. Vinnytsia: VNTU, 2022. 102 p.

In Ukrainian. Bibliogr.: 22 titles; Fig. 35; Table 13.

Master's thesis is devoted to the development of a method and tool for user authentication using tokens. The analysis of modern methods of user authentication is carried out and mathematical description of user authentication process is synthesised. The method of remote user authentication using tokens is developed. A software tool that allows user authentication by the developed method is designed and implemented. Experimental research of the method and tool for user authentication is performed. The economical reasoning of developed tool usage expediency is performed.

Keywords: *user authentication, mobile application, token, remote users, digital signature, multi–factor authentication*.

ЗМІСТ

ВСТУП

Розв’язання задачі автентифікації користувачів стало невід’ємною частиною сучасного інформаційного простору. Більшість багатокористувацьких інформаційних систем, починаючи від засобів обміну миттєвими повідомленнями, закінчуючи веб–ресурсами та операційними системами, вимагають від користувачів використання облікових записів для отримання доступу до інформації ресурсів та системи, в цілому. Кожен користувач виконує автентифікацію перед початком роботи із обліковим записом, тому виконання процесу автентифікації є надзвичайно важливим з точки зору безпеки.

Водночас відомі методи стійкої автентифікації передбачають від користувача використання декількох факторів автентифікації, що породжує незручність. Особливо це набуває актуальності для багатокористувацьких комерційних систем, де стійка автентифікація віддалених користувачів зменшує комерційну привабливість програмного продукту. У зв’язку з цим виникає актуальна задача розробки методу автентифікації, який дозволить користувачам керувати налаштуваннями параметрів автентифікації відповідно до власних потреб.

Об’єктом дослідження є процес автентифікації віддалених користувачів.

Предметом є методи та засоби автентифікації віддалених користувачів.

Метою магістерської кваліфікаційної роботи є покращення рівня захищеності віддалених користувачів під час автентифікації за рахунок введення додаткового фактору автентифікації із можливістю керування його параметрами. Для досягнення мети необхідно:

- проаналізувати методи автентифікації;
- розробити метод автентифікації віддалених користувачів на основі токерів;
- розробити структуру та правила формування токенів;
- розробити засіб автентифікації;
- виконати тестування та експериментальне дослідження засобу;
- здійснити обґрунтування економічної доцільності розробки.

Наукова новизна магістерської роботи полягає в тому, що отримав подальший розвиток метод автентифікації віддалених користувачів, який відрізняється від відомих тим, що реєстрація користувача відбувається автоматично та користувач контролює генерацію токенів, що дозволяє збільшити стійкість автентифікації за рахунок використання додаткового фактору автентифікації, який покращить зручність користування для віддалених користувачів.

Практична цінність: одержаний метод і мобільний застосунок спрощують автентифікацію віддаленому користувачеві, надаючи можливість контролювати доступ до даних користувача.

Результати магістерської роботи доповідалися на таких конференціях:

– L науково–технічної конференції факультету інформаційних технологій та комп'ютерної інженерії, м. Вінниця, 2021[1];

– LI Науково–технічна конференція факультету інформаційних технологій та комп'ютерної інженерії, м. Вінниця, 2022 [2].

За результатами магістерської кваліфікаційної роботи опубліковано тези доповідей у збірниках матеріалів конференцій.

1 АНАЛІЗ МЕТОДІВ АВТЕНТИФІКАЦІЇ КОРИСТУВАЧІВ

1.1 Науково-технічне обґрунтування

Автентифікація – це процедура перевірки чи є суб’єкт тим, за кого себе видає, чи не використовує чужий ідентифікатор.

Автентифікацією користується якщо не увесь то майже весь. Перевірка користувача дозволяє захистити контроль доступу для систем, при цьому перевіряються дані що зберігають в базі даних. Зазвичай є ідентифікатором користувача та якийсь секрет, який знає користувач, за допомогою секрету і проводиться перевірка чи цей користувач той, за кого себе видає. Після автентифікації настає етап авторизації, або ж надання прав користувачу.

Досить часто терміни автентифікації та авторизації використовують як одне і теж саме, хоча вони такими не являються. Спочатку відбувається саме автентифікація, оскільки системі потрібно знати що далі робити із цим користувачем можна а що ні. Лише після авторизації користувач може повноцінно взаємодіяти із ресурсом.

Після того як було введено карантинні обмеження у 2020 році, компанії почали перехід на ввіддалену працю. Через що збільшилось віддалених взаємодій із ресурсами компаній, цим скористались кіберзлочинці.

1.2 Аналіз видів автентифікації користувачів

Наразі відомі такі типи автентифікацій: парольні, біометричні, апаратні, цифрові сертифікати [5 – 7]. Вони усі мають особливості, через які мають власні позитивні та негативні сторони, також через це може використовуватись двохфакторна автентифікація (в деяких випадках це комбінація декількох типів).

Автентифікація користувача залежить від одного або декількох параметрів, які повинен пред'явити користувач, для успішного проходження автентифікації. Ці параметри називаються факторами автентифікації.

Фактор автентифікації це дані або атрибут, що може використовуватися для автентифікації користувача.

Фактори автентифікації можуть бути такими :

– фактор знання передбачає наявність певної відомої тільки користувачу інформації;
– фактор володіння має на увазі, що користувач володіє деяким унікальним предметом, що містить необхідну характеристику (пластикові карти, USB токени тощо);
– фактор властивості передбачає використання деякої фізичної особливості користувача, наприклад, відбиток пальця, райдужна оболонка ока тощо;
– фактор розташування передбачає перевірку місцезнаходження користувача;
– фактор часу.

Залежно від кількості аутентифікуючих сторін аутентифікація може бути односторонньою (зазвичай клієнт доводить свою автентичність серверу) і двосторонньою (взаємною). Для його реалізації використовується спеціальний протокол шифрування.

Залежно від кількості факторів аутентифікація може бути однофакторною або багатофакторною. Однофакторна автентифікація використовує один із факторів знань, власності або властивостей. Факторів місця та часу недостатньо для однофакторної автентифікації, лише як додаткові фактори.

Використання багатофакторної автентифікації забезпечує підвищений рівень безпеки, оскільки неавторизовані користувачі не зможуть надати всі необхідні фактори. Якщо принаймні один фактор відсутній або неправильно вказано під час процесу автентифікації, система не може визначити автентичність користувача, який аутентифікується, і, отже, користувач не зможе отримати доступ до системи.

Найбільш поширеною формою аутентифікації є використання факторів знань. Використовуючи фактор знань під час автентифікації, користувач повинен підтвердити знання секрету. В якості секрету найчастіше використовується пароль - секретне слово або рядок, відоме тільки реальному користувачеві. Більшість механізмів багатофакторної автентифікації використовують пароль як один із факторів автентифікації. Варіанти PIN-кодів включають довші PIN-коди (що складаються з кількох слів) і коротші цифрові PIN-коди, які часто використовуються для доступу до автоматизованих касових апаратів.

Фактор власності передбачає автентифікацію конкретного об’єкта, яким володіє лише користувач. Основний принцип полягає в тому, що користувач може бути автентифікований лише за допомогою необхідних об’єктів, які йому належать. Прикладом фактору володіння є токени, що можуть бути декількох видів:

– Токени без підключення – фізичні пристрої, що не мають зв’язку з комп’ютерним пристроєм користувача. На таких пристроях зазвичай дані автентифікації відображаються на спеціалізованому екрані, а користувач самостійно копіює їх для виконання автентифікації;

– Токени з підключенням – фізичні пристрої, що підключаються до комп’ютерного пристрою користувача та автоматично передають дані автентифікації;

– Програмні токени – це тип пристрою двофакторної автентифікації, що може використовуватися для автентифікації при використання комп'ютерних сервісів. Програмні токени можуть зберігатися на будь-яких електронних пристроях та можуть бути дубльовані на декількох пристроях. Головним недоліком використання токенів є ймовірність втрати або крадіжка токену.

Фактор атрибута використовує фізичні характеристики суб’єкта, який підлягає автентифікації, для автентифікації. Для користувачів фізичними атрибутами можуть бути відбитки пальців або долонь, форма обличчя, голос, сітківка ока тощо. З точки зору принципала, використання факторів атрибутів

спрощує автентифікацію: немає необхідності запам'ятовувати паролі, зберігати додаткове програмне забезпечення або апаратні пристрої автентифікації. Однак використання атрибутивних факторів суб’єкта потребує наявності додаткового програмного та апаратного забезпечення, що дозволяє зчитувати та обробляти параметри фізичних характеристик суб’єкта. Відповідна система має бути достатньо чутливою та точною, щоб ідентифікувати відповідні атрибути різних об’єктів. Такі системи значно збільшують вартість процесу сертифікації.

Недоліком використання фактора атрибута є те, що зловмисник може підробити або вкрасти фізичний атрибут, і в деяких випадках фізичний атрибут суб’єкта неможливо змінити, наприклад, якщо зловмисник підробить відбиток пальця, який використовується як фактор автентифікації , то користувач не зможе змінити відбиток пальця, щоб змінити доступ до системи.

Фактор фізичного розташування використовує перевірку мережі, у якій перебуває користувач. Якщо користувач підключений до захищеної корпоративної мережі, йому буде надано доступ до системи, інакше в доступі буде відмовлено на основі факторів фізичного розташування. Для автентифікації одного лише фізичного місцезнаходження недостатньо, тому цей фактор часто використовується як додатковий фактор багатофакторної автентифікації.

Фактор часу також використовується як додатковий фактор при багатофакторній автентифікації та передбачає перевірку параметру часу проходження автентифікації та його відповідність умовам, що відповідають системі автентифікації.

Кожний із типів автентифікації має свою складність реалізації та захищеність, оцінку даних типів наведено в таблиці 1.1, де 1 мінімальна бал, 5 є максимальним балом.

Таблиця 1.1 – Оцінка складності та захищеності типів автентифікації

Параметр	парольна автентифікація	апаратна автентифікація	біометрична автентифікація	автентифікація за цифровими сертифікатами
Складність/ціна	1	4	3	2
Захищеність	2	4	3	3

Парольна автентифікація – користувач повинен пам’ятати або ж зберігати дані, зазвичай ім’я користувача та пароль. Пароль – секретна комбінація символів що відома лише власнику. Складність паролю впливає на захищеність, при цьому проблемою є sniffing, зловмисник може отримати доступ до паролю під час комунікації. Ключовим фактором є людський [5 – 7]:

- якщо пароль слабкий, його легко вгадати або підібрати;
- паролі можна вкрасти, якщо вони записані;
- користувачі можуть самі поділитися паролями;
- якщо пароль складний(сильний) його можна забути.

Апаратна автентифікація – це апарати(пристрої) що мають вбудований сертифікат, який використовується для ідентифікації власника. Зазвичай використовується із PIN–кодом для забезпечення багатофакторної аутентифікації. Одним із недоліків є те що апаратний пристрій можуть вкрасти або ж загубити.

Біометрична автентифікація – фактором автентифікації є користувач. Її поділяють на фізіологічну, поведінкову [5 – 7]. Фізіологічною біометрією є відбиток пальця, обличчя, око, геометрія руки. Поведінковою біометрією є розпізнавання голосу, хода, клавіатурне сканування та підпису. Серед недоліків відиляють неточність систем (проблема з помилками 1 та 2 типу, при налаштуванні зменшуватимуться кількість одного типу, та збільшуватиметься іншого), вразливість користувача (біометричний параметр може зазнати змін при нещасному випадку).

Автентифікація за цифровими сертифікатами – використовує публічний/приватний ключі. Дані шифруються що надає конфіденційності даним. Цифрові сертифікати зручні для користувачів, зазвичай працюють автоматично і вимагають мінімальних дій або участі будь–кого відправника або одержувача.

1.3 Методи автентифікації віддалених користувачів

Серед доступних методів можна виділити чотири найпопулярніших: HTTP Authentication Schemes, API Keys, OAuth, OpenID Connect [8].

HTTP протокол визначає схеми авторизації безпеки [9]:

- basic;
- bearer;
- digest;
- OAuth;
- та інші.

Серед них найпопулярнішими є такі: basic, bearer. Базову автентифікацію HTTP рідко рекомендують через її вразливість безпеки. Це найпростіший і найпряміший метод.

Ім’я користувача та пароль кодуються за допомогою Base64. Для цього методу не потрібні файли cookie, ідентифікатори сеансу, сторінки входу та інші подібні спеціальні рішення, а оскільки він використовує сам заголовок HTTP, немає потреби в рукостисканнях або інших складних системах відповіді.

Bearer – це схема автентифікації HTTP, яка включає токени безпеки. Токен що дозволяє отримати доступ до певного ресурсу чи URL–адреси, і, швидше за все, є зашифрованим рядком, який зазвичай генерується сервером у відповідь на запит входу. Клієнт повинен надіслати цей токен у заголовку авторизації під час надсилання запитів до захищених ресурсів.

Схема авторизації спочатку була створена як частина OAuth 2.0 у RFC–6750 [10], але іноді також використовується окремо. Подібно до базової автентифікації, автентифікацію слід використовувати лише через HTTPS (SSL).

Ключі API були створені як певне вирішення проблем ранньої автентифікації базової автентифікації HTTP та інших подібних систем. У цьому методі кожному користувачу вперше призначається унікальне згенероване значення, яке означає, що користувач відомий. Коли користувач намагається повторно увійти в систему, його унікальний ключ (іноді згенерований на основі комбінації апаратного забезпечення та IP–даних, а в інших випадках випадково згенерований сервером, який їх знає) використовується для підтвердження того, що це той самий користувач, що й раніше (рис. 1.1).

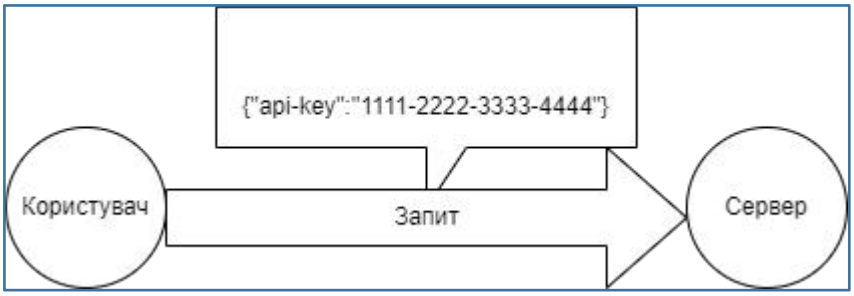


Рисунок 1.1 – Автентифікація користувача через запит із API key

Багато ключів API надсилаються в рядку запиту як частина URL–адреси, що полегшує виявлення для тих, хто не повинен мати до нього доступу.

Ключі API – прості. Використання єдиного ідентифікатора є простим, а для деяких випадків використання найкращим рішенням. Наприклад, якщо API обмежений у функціональності, де «читати» є єдиною можливою командою, ключ API може бути достатнім рішенням.

Однак проблема полягає в тому, що кожен, хто робить запит до служби, передає свій ключ, і теоретично цей ключ можна підібрати так само легко, як і будь–яку мережеву передачу, і якщо будь–яка точка у всій мережі є незахищеною, вся мережа розкрита.

Попередні версії цієї специфікації, OAuth 1.0 і 1.0a, були набагато складнішими, ніж OAuth 2.0 [11]. Найпоширеніші реалізації OAuth замість цього використовують один або обидва токени:

access token – надсилається як ключ API, він дозволяє застосунку отримувати доступ до даних користувача, за бажанням, токени доступу можуть закінчуватись;

refresh token – необов'язково частина потоку OAuth, refresh tokens отримує новий токен доступу, якщо термін їх дії минув. OAuth2 поєднує автентифікацію та авторизацію, щоб забезпечити більш складний контроль обсягу та дійсності.

У OAuth 2.0 користувач входить у систему. Згодом система запитує автентифікацію, як правило, у формі токена. Після чого користувач перешле цей запит на сервер автентифікації, який або відхилить, або дозволить цю автентифікацію. Токен надається користувачеві, а потім запитувачу. Такий токен може бути перевірений у будь–який час незалежно від користувача запитувачем для перевірки та може використовуватися з часом із суворо обмеженим обсягом і терміном дії.

Це принципово набагато безпечніша та потужніша система, ніж інші підходи, головним чином тому, що вона дозволяє встановлювати області, які можуть надавати доступ до різних частин служби API, і оскільки маркер відкликається через певний час, це значно ускладнює роботу для повторного використання зловмисниками.

OAuth 2.0 надає кілька популярних варіантів, які підходять для різних типів клієнтів API [11]:

- авторизація;
- парольна автентифікація;
- надсилання облікових даних клієнта;
- OpenID Connect.

Авторизації – найпоширеніший процес, який переважно використовується для серверних і мобільних веб–застосунків.

Пароль власника ресурсу – вимагає входу за допомогою імені користувача та пароля. Оскільки в такому випадку облікові дані будуть частиною запиту, цей варіант підходить лише для довірених клієнтів (наприклад, офіційних програм, випущених постачальником API).

Облікові дані клієнта – цей варіант призначений для автентифікації між серверами, описує підхід, коли клієнтська програма діє від свого власного імені, а не від імені окремого користувача. У більшості сценаріїв цей потік забезпечує засоби, які дозволяють користувачам вказувати свої облікові дані в клієнтській програмі, щоб вона могла отримати доступ до ресурсів під контролем клієнта.

OpenID Connect – базується на основі протоколу OAuth 2.0, який дозволяє комп'ютерним клієнтам перевіряти особу користувача на основі автентифікації, виконаної сервером авторизації, а також отримувати основну інформацію про профіль кінцевого користувача. користувач сумісним способом, подібним до REST [12].

З технічної точки зору OpenID Connect визначає RESTful API[13], використовуючи JSON як формат даних.

OpenID Connect дозволяє низці клієнтів, включаючи веб–, мобільні та JavaScript–клієнти, запитувати та отримувати інформацію про сеанси та кінцевих користувачів. Набір специфікацій є розширюваним і підтримує додаткові функції, такі як шифрування ідентифікаційних даних, виявлення постачальників OpenID і керування сеансами.

OpenID Connect визначає процес входу, який дозволяє клієнтській програмі автентифікувати користувача та отримувати інформацію (або «заяви») про цього користувача, наприклад ім'я користувача, електронну адресу тощо. Ідентифікаційна інформації користувача закодована в захищеному веб–токені JSON (JWT), який називається ID–токен.

OpenID Connect визначає механізм виявлення під назвою OpenID Connect Discovery, де сервер OpenID публікує свої метадані за відомою URL–адресою, як правило

URL–адреса повертає перелік JSON кінцевих точок OpenID/OAuth, підтримуваних областей і претензій, відкритих ключів, які використовуються для підпису маркерів, та інших деталей. Клієнти можуть використовувати цю інформацію для створення запиту до сервера OpenID. Імена та значення полів визначені в специфікації OpenID Connect Discovery.

Методи для надання віддаленого доступу що не базуються на токенах та базуються на них мають власні проблеми.

Методи що використовують токени володіють такими недоліками:

– зберігання великої кількості даних у токені робить його великим, що сповільнює запити;

– токени не можна використовувати для автентифікації користувача у фоновому режимі на сервері, оскільки в базі даних не існує сеансу;

– зберігання маркерів на стороні клієнта є проблематичним. Токени можна зберігати у сховищі сеансу, але він очищається, коли закривається браузер. У локальному сховищі токени будуть прив’язані до певного домену;

– токен на стороні клієнта може бути викрадений зловмисником, що робить його вразливим до атак міжсайтового сценарію (XSS);

– можлива атака man–in–the–middle;

– можливе використання токену навіть якщо запис користувача видалено.

Методи що не використовують токени володіють такими недоліками:

– потрібно зберігати дані сеансу в базі даних або зберігати їх у пам’яті на сервері. Це робить його менш масштабованим;

– клієнт повинен надсилати дані на кожен запит, навіть якщо URL–адреси не потребують автентифікації для доступу;

– потрібно встановлювати з’єднання із кожним пристроєм;

– вразливість до credential stuffing/brute force.

Отже, тема є актуальною. Метод повинен автоматично оновлювати токени, бути захищеним від атаки man–in–the–middle.

1.4 Аналіз токенів авторизації

Авторизація на основі токену – це протокол, який генерує зашифровані токени безпеки. Цей токен надає користувачам доступ до захищених сторінок і ресурсів протягом обмеженого періоду часу без необхідності повторного введення імені користувача та пароля [14].

Автентифікація на основі токенів працює через процес із 5 етапів [14]:

Етап 1. Запит: користувач входить у службу, використовуючи свої облікові дані для входу, що надсилає запит на доступ до сервера або захищеного ресурсу.

Етап 2. Перевірка: сервер перевіряє інформацію для входу, щоб визначити, чи повинен користувач мати доступ. Це передбачає перевірку введеного пароля на відповідність наданому імені користувача.

Етап 3. Надсилання токена: сервер генерує безпечний підписаний токен автентифікації для користувача протягом певного періоду часу.

Етап 4. Зберігання: токен передається назад у браузер користувача, який зберігає його для доступу до майбутніх відвідувань веб–сайту. Коли користувач переходить на новий веб–сайт, токен автентифікації декодується та перевіряється. Якщо знайдеться збіг, користувачеві буде дозволено продовжити.

Етап 5. Термін дії: токен залишатиметься активним, доки користувач не вийде з системи або не закриє сервер.

Ключові переваги токенів автентифікації [15]:

1. Токени не мають стану: токени автентифікації створюються службою автентифікації та містять інформацію, яка дає змогу користувачеві підтвердити свою особу без введення облікових даних для входу.

2. Термін дії токенів закінчується: коли користувач завершує сеанс перегляду та виходить із служби, наданий йому токен знищується. Це гарантує, що облікові записи користувачів захищені та не піддаються ризику кібератак.

3. Токени шифруються та генеруються машиною. Кожен токен є унікальним для сеансу користувача та захищений алгоритмом, який гарантує,

що сервери можуть ідентифікувати токен, який був підроблений, і заблокувати його.

4. Токени спрощують процес входу: токени автентифікації гарантують, що користувачам не потрібно повторно вводити свої облікові дані кожного разу, коли вони відвідують веб–сайт. Це робить процес швидшим і зручнішим для користувачів, що довше утримує людей на веб–сайтах і спонукає їх відвідувати їх знову в майбутньому.

5. Токени додають бар'єр для запобігання хакерам: бар'єр 2FA для запобігання доступу хакерів до даних користувача та корпоративних ресурсів.

Серед недоліків можна виділити [15]:

– використовується лише один ключ, при неналежному використанні є можливість скомпрометувати конфіденційну інформацію;

– час життя. Чим більший час життя тим більша ймовірність що зловмисник отримає доступ до нього, при малому часі життя потрібно буде проводити автентифікацію частіше що дратуватиме користувачів та збільшить можливість перехоплення вхідних даних.

Використання лише паролів полегшує хакерам перехоплення облікових записів користувачів, але за допомогою токенів користувачі можуть підтверджувати свою особу за допомогою фізичних токенів і додатків для смартфонів. Це додає додатковий рівень безпеки, не даючи хакерам отримати доступ до облікового запису, навіть якщо їм вдасться викрасти облікові дані користувача.

JSON Web Token (JWT) – це відкритий стандарт (RFC 7519 [16]), який визначає компактний і самодостатній спосіб безпечної передачі інформації між сторонами у JSON форматі. Цю інформацію можна перевірити та довіряти їй, оскільки вона має цифровий підпис. JWT можна підписувати за допомогою секрету (з алгоритмом HMAC) або пара публічний/приватний ключ за допомогою RSA або ECDSA [17].

Підписані токени можуть підтвердити цілісність частин, що містяться в ньому, тоді як зашифровані токени приховують ці частини від інших сторін.

Коли токени підписуються за допомогою пар публічних/приватних ключів, підпис також засвідчує, що лише сторона, яка володіє закритим ключем є тією, яка підписала його.

Авторизація – це найпоширеніший варіант використання JWT. Після входу користувача кожний наступний запит включатиме JWT, дозволяючи користувачеві отримувати доступ до маршрутів, послуг і ресурсів, які дозволені цим маркером. Single Sign On (єдиний вхід) – це функція, яка сьогодні широко використовує JWT через невеликі накладні витрати та можливість легкого використання в різних доменах.

Веб–токени JSON є способом безпечної передачі інформації між сторонами. Оскільки JWT можна підписувати, наприклад, за допомогою пари публічний/приватний ключів – сервери можуть бути впевнені, що відправники є тими, за кого себе видають. Крім того, оскільки підпис обчислюється за допомогою header–а та payload, є також можливість перевірити чи вміст не було змінено.

У своїй формі веб–токени JSON складаються з трьох частин, розділених крапками [17]:

- header;
- payload;
- signature.

Таким чином, JWT зазвичай виглядає так : xxxxx.yyyyy.zzzzz.

Header зазвичай складається з двох частин: типу маркера, яким є JWT, і використовуваного алгоритму підпису, наприклад HMAC SHA256 або RSA [17]. Тоді цей JSON кодується Base64Url, щоб сформувати першу частину JWT (рис. 1.2).

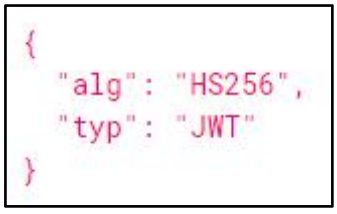


Рисунок 1.2 – Приклад header

Друга частина токена – це payload, який містить дані. Зазвичай це дані про сутність (користувача) і додаткові дані. Існує три види даних: зареєстровані, публічні та приватні [17].

Зареєстровані – це набір попередньо визначених даних, які не є обов'язковими, але рекомендовані для надання набору корисної, сумісної інформації. Деякі з них: iss (емітент), exp (термін дії), sub (тема), aud (аудиторія) та інші.

Публічні можуть бути визначені за бажанням тими, хто використовує JWT. Але щоб уникнути колізії, їх слід визначити в реєстрі веб–токенів IANA JSON або визначити як URI, що містить простір імен, стійкий до зіткнень.

Приватні – це спеціальні дані, створені для обміну інформацією між сторонами, які домовилися про їх використання, і не є ні зареєстрованими, ні публічними.

Payload потім кодується Base64Url для формування другої частини веб–токена JSON (рис. 1.3).

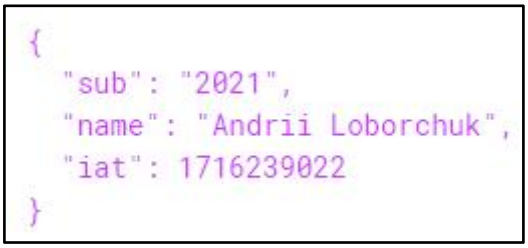


Рисунок 1.3 – Приклад payload

Підпис використовується для перевірки того, що повідомлення не було змінено на шляху, і у випадку токенів, підписаних приватним ключем, він також може підтвердити, що відправник JWT є тим, за кого він себе видає.

Щоб створити частину підпису, потрібно взяти закодований заголовок, закодовану корисну інформацію, секрет, алгоритм, зазначений у заголовку, і підписати це [17]. Наприклад, на рисунку 1.4 зображено як буде створено підпис при використанні алгоритму HMAC SHA256.

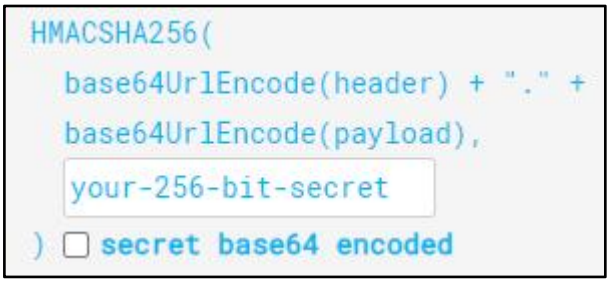


Рисунок 1.4 – Приклад створення signature

Результатом є три рядки Base64–URL, розділені крапками, які можна легко передати в середовищах HTML і HTTP, але при цьому є більш компактними порівняно зі стандартами на основі XML, такими як SAML[18].

На рисунку 1.5 зображено JWT, який має попередній header і payload, закодовані та підписані секретом.

Рисунок 1.5 – Приклад JWT

Для перевірки JWT або ж генерації можливо використати сайт jwt.io [17].

Термін дії JWT закінчується через певні проміжки часу. Коли JWT створюється, йому надається певний момент закінчення терміну дії. Термін життя JWT є остаточним, і рекомендується, щоб він був дещо коротким (хвилини, а не години). Традиційні сеанси та JWT відрізняються. Традиційні сеанси – це завжди певна тривалість від останньої взаємодії з користувачем. Тобто якщо користувач натискає кнопку, його сеанс подовжується. Натомість JWT програмно замінюються шляхом створення нового JWT для користувача. Щоб вирішити цю проблему використовують маркери оновлення. Маркери

оновлення – це непрозорі маркери, які використовуються для створення нових JWT. Термін дії маркерів оновлення також повинен закінчитися в якийсь момент, але вони можуть бути більш гнучкими в цьому механізмі, оскільки вони зберігаються в постачальнику ідентифікаційної інформації.

Також відомі випадки, коли у JWT було виявлені експлойти [19]. Стандарт JWT приймає багато різних типів алгоритмів для створення підпису:

- RSA;
- HMAC;
- еліптична крива
- none(жодного).

Алгоритм none визначає, що маркер не підписаний. Якщо цей алгоритм дозволений, є можливим обійти перевірку підпису, змінивши існуючий алгоритм на none та видаливши підпис. Заголовок JWT може містити параметр Kid ID ключа. Його часто використовують для отримання ключа з бази даних або файлової системи. Програма перевіряє підпис за допомогою ключа, отриманого через параметр kid. Якщо можна зробити ін’єкцію параметра, він може відкрити шлях до обходу підпису або атак RCE, SQLi та LFI [19]. У заголовку JWT розробники також можуть використовувати параметр jku, щоб указати URL–адресу набору веб–ключів JSON. Цей параметр вказує, де програма може знайти веб–ключ JSON (JWK), який використовується для перевірки підпису. Зловмисник може змінити значення параметра jku, щоб вказати на його власний JWK замість дійсного. Це дозволить зловмиснику підписувати шкідливі маркери за допомогою власного закритого ключа. Після надсилання шкідливого токена програма отримає JWK зловмисника та використає його для перевірки підпису.

1.5 Постановка задачі досліджень

Отже безпека при автентифікації користувачів є важливою та актуальною задачею. Люди користуються все більшою кількістю веб-ресурсів,

через що збільшуються можливості кіберзлочинців. Існуючі засоби автентифікації не завжди дозволяють забезпечити високий рівень безпеки, оскільки є вразливими до атак різного роду та мають свої недоліки. Тому виконання досліджень у цій сфері залишиться актуальним ще довгий час.

Для того, щоб підвищити рівень захищеності під час автентифікації користувачів, необхідно при розробці нової моделі протоколу автентифікації потрібно опиратися на ефективні сучасні криптографічні протоколи та алгоритми.

Із розглянутих методів автентифікації користувачів можна виділити протоколи автентифікації із використанням токена.

На основі обраної моделі необхідно розробити модель процесу автентифікації із використанням токена для користувачів. Відповідну модель представити у вигляді протоколу. Здійснити розробку алгоритмів сформованим протоколом автентифікації. Розроблені алгоритми реалізувати у вигляді програмного засобу, що відповідатиме таким вимогам:

– програмний засіб повинен бути представленим у вигляді мобільного застосунку;
– забезпечити захищену автентифікацію користувача;
– забезпечити можливість роботи з більше ніж одним ресурсом.

1.6 Висновки з розділу

Тому процес аутентифікації дуже важливий. Це допомагає організаціям підтримувати безпеку мережі, гарантуючи, що лише авторизовані користувачі мають доступ до захищених ресурсів, включаючи комп’ютерні системи, мережі, бази даних, веб-сайти та інші мережеві програми та служби. У цьому розділі розглядаються сучасні методи автентифікації користувачів і обговорюються ключові поняття. Розглянемо види аутентифікації та їх функції.

Використання токенів при автентифікації має низку переваг щодо безпеки:

– унікальність – токени є специфічними і можуть створюватися для використання або для певного пристрою;

– можливість відкликання – токени можна відкликати окремо в будь–який час без необхідності оновлення облікових даних, які не змінюються;

– обмеженість – токени можуть мати обмежені набори прав, щоб дозволити лише доступ, необхідний для випадку використання;

– випадковість – токени генеруються на основі генераторів псевдовипадкових чисел та зберігають секретні дані у зашифрованому вигляді.

Окрім доступу до даних, токени можуть використовуватись у захисті від піратства, що теж є важливим. При піратстві порушуються авторські права авторів [2].

На основі отриманих у першому розділі результатів досліджень інформаційних джерел можна перейти до моделювання протоколу автентифікації віддалених користувачів із використанням токенів та його подальшого дослідження.

2 МЕТОД АВТЕНТИФІКАЦІЇ ВІДДАЛЕНИХ КОРИСТУВАЧІВ

2.1 Математичний опис процесу автентифікації

Для того щоб формалізовано описати системи використано теоретико–множинний підхід, оскільки він дозволяє описувати множину усіх елементів, які входять до неї. Дослідження процесу автентифікації користувачів дозволяє представити систему автентифікації, як сукупність множин вхідних та вихідних даних, постійних та змінних параметрів системи:

$$SA = (X, Y, B, A, O), \text{ де} \tag{2.1}$$

SA – система автентифікації;

X – множина вхідних даних;

Y – множина вихідних даних;

B – множина постійних параметрів системи;

A – множина змінних параметрів системи;

O– множина виконуваних операцій.

У процесі односторонньої автентифікації користувача приймають участь сторона J (користувач), що проходить автентифікацію, сторона G (сервер, сервіс), що відповідає за перевірку автентичності J. Тоді для сторін J та G множини складових системи будуть відрізнятися та належатимуть підмножинами відповідних вказаних вище множин.

$$\begin{gathered} X^J; X^G \subset X, \\ Y^J; Y^G \subset Y, \\ P^J; P^G \subset P, \\ N^J; N^G \subset N, \\ O^J; O^G \subset O \end{gathered} \tag{2.2}$$

Процес автентифікації на основі асиметричного шифрування використовує відкритий та секретний ключі. Математичний опис виглядатиме таким чином для:

–сторони J:

$$X^J = \{I^G; kS; h(r); P^{kB}(r; I^G)\}, Y^J = \{r^*\}, B^J = \{P, h; I^G; kS\}, A^J = \{r^*; h(r^*); I^{G*}\},$$
$$O^J = \{O_1, O_2, \dots, O_n\} \quad (2.3)$$

–сторони G:

$$X^G = \{I^G; kB\}, Y^G = \{h(r); P^{kB}(r; I^G); R\}, B^G = \{P, h; I^G\},$$
$$A^G = \{r; P^{kB}(r; I^G); h(r); r^*; I^{G*}; R\}, O^G = \{O_1^*, O_2^*, \dots, O_n^*\} \quad (2.4)$$

I – ідентифікатор користувача;

kS – секретний ключ;

kB – відкритий ключ;

r – певне псевдовипадкове значення;

P – алгоритми асиметричного криптографічного шифрування;

$P^{kB}(r; I^G)$ – значення що зашифроване алгоритмом P з використанням відкритого ключа kB ;

$r^*; I^{G*}$ – значення розшифроване алгоритмом P з використанням секретного ключа kS ;

h – алгоритм гешування;

$h(r)$; $h(r^*)$ – геш псевдовипадкових значень;

Oi – арифметична операція;

R – результат автентифікації.

Отже було розглянуто математичний опис односторонньої автентифікації із використанням асиметричного шифрування.

2.2 Узагальнений опис метода автентифікації

Мобільні телефони є у більшості людей, а мобільні застосунки використовуються все частіше, тому було обрано мобільний застосунок як один із елементів методу автентифікації.

Для двоетапної перевірки використовується найчастіше одноразовий пароля на основі часу (TOTP) і одноразового пароля на основі HMAC (HOTP) для автентифікації користувачів.

TOTP – це алгоритм, який обчислює одноразовий пароль із загального секретного ключа та поточного часу.

HTOP – це алгоритм, який використовує алгоритм hmac для генерації одноразового пароля.

В методі братимуть участь 3 сторони рис. 2.1: клієнт, сервер, пристрій для автентифікації.

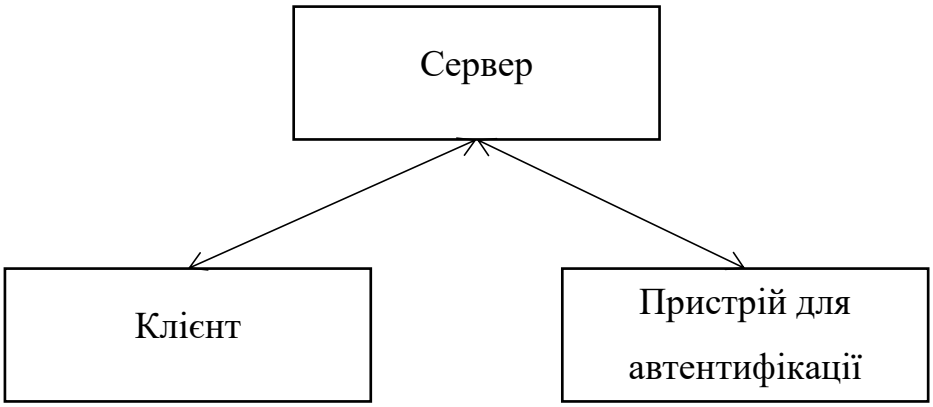


Рисунок 2.1 – Структура методу автентифікації

Клієнт – пристрій який бажає отримати токен. Сторона “Клієнт” це пристрій який відображає інформацію ресурсу. Сервер – ресурс на якому клієнт бажає автентифікуватись та отримати інформацію. Пристрій для автентифікації

– мобільний пристрій із розробленим програмним застосунком для автентифікації.

Користувачу для взаємодії потрібно буде встановити програмний застосунок на мобільний телефон та зв’язати його із даними користувача на сервері.

Для швидкої передачі інформації використовують штрих–коди. Всі штрих–коди поділяються на 1D штрих–коди і 2D штрих–коди. Існує багато версій, найчастіше використовується EAN–13, GS1 DataBar, Code–128, QR Code, ISBN. 1D коди – це коди, що складаються із чорних і білих смуг (наприклад, EAN–13). 2D штрих–коди включають і інші геометричні фігури. Це дозволяє їм зберігати більше інформації та займати менше місця, ніж 1D–коди. Залежно від конкретного типу штрих–коду, 1D штрих–коди можуть мати від 20 до 25 символів, а 2D–коди – до 2000 символів. На вигляд штрих–коду впливає кількість закладеної в ньому інформації, мова введення, наявність цифр, великих літер. Найчастіше обирають EAN–13 або QR–код. У коді EAN–13 містяться тільки цифри.

Код EAN–13 має таку структуру:

- перші 2–3 цифри – це код країни, де зареєстрований даний номер;
- наступні 4–5 цифр – реєстраційний номер підприємства;
- наступні 3–5 цифр – порядковий номер продукції усередині підприємства;
- остання 13–а цифра – контрольне число.

На рисунку 2.2 зображено приклад EAN–13 коду.

Рисунок 2.2 – Приклад EAN–13 коду

QR–код містить і літери, і цифри, тому він більш універсальний. На рисунку 2.3 зображено тестовий QR–код, а на рисунку 2.4 текст зо в нього записаний.

Рисунок 2.3 – QR–код із тестовим текстом

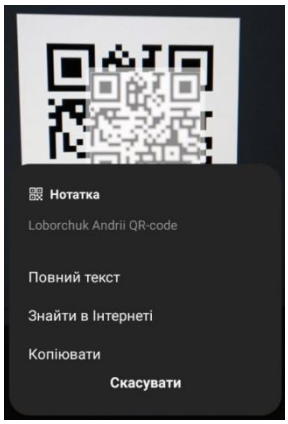


Рисунок 2.4 – Зчитування тексту із тестового QR–код

Отже QR–код – це швидкий та доступний метод передання інформації. Оскільки користувачу потрібно отримувати дані із екрану персонального комп’ютера такий код легко передає будь-яку потрібну інформацію у вигляді тексту, який можливо перетворювати. Одним із найкращих варіантів перевання потрібної інформації є JSON формат перетворити у рядок та передати через QR–код. Після зчитування інформації рядок перетворюється назад у JSON формат.

При встановленні зв’язку програмний застосунок отримує дані про сервер та записує ці дані у пам’ять телефону. Наступним кроком є надсилання публічного ключа серверу. Після чого користувач може отримувати запити на отримання токена. Якщо користувач виконав усі ці кроки, у нього є можливість отримувати список запитів на отримання токенів. Узагальнений алгоритм методу автентифікації зображено на рисунку 2.5.

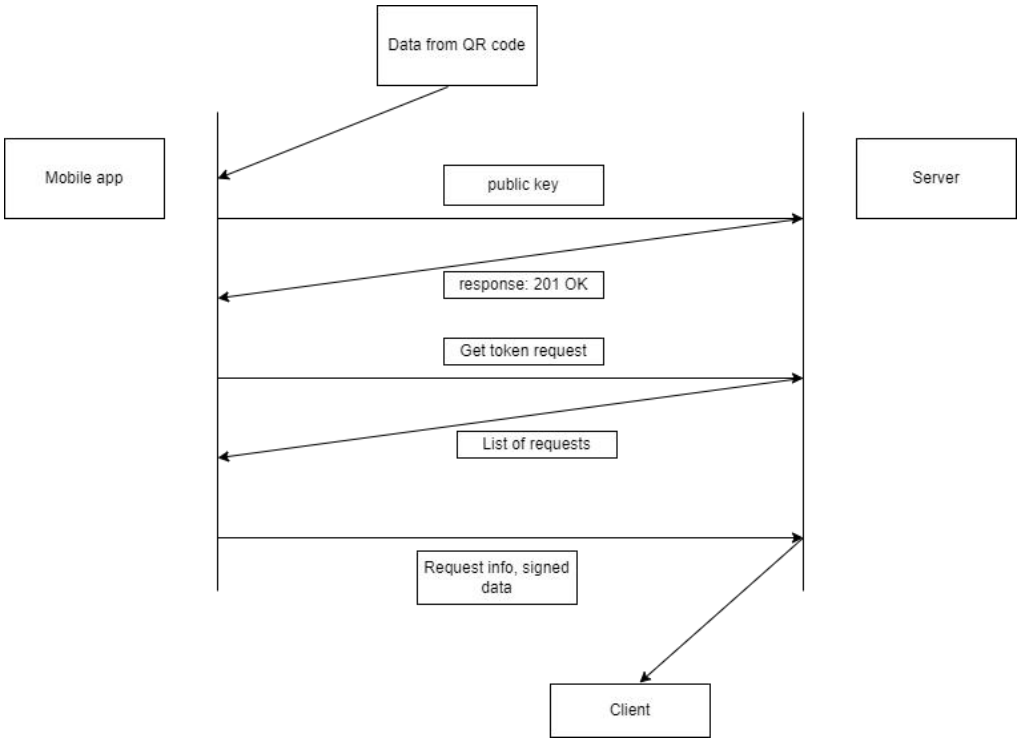


Рисунок 2.5 – Узагальнений алгоритм роботи методу

Для підпису та верифікації використовуватиметься алгортм RSA. Алгоритм RSA складається з 4 етапів: генерації ключів, підпису, верифікації та розповсюдження ключів.

Безпека алгоритму RSA побудована на принципі складності факторизації цілих чисел. Алгоритм використовує два ключі — відкритий (public) і секретний (private), разом відкритий і відповідний йому секретний ключі утворюють пари ключів (keypair). На рисунку 2.6 зображено схему підпису RSA.

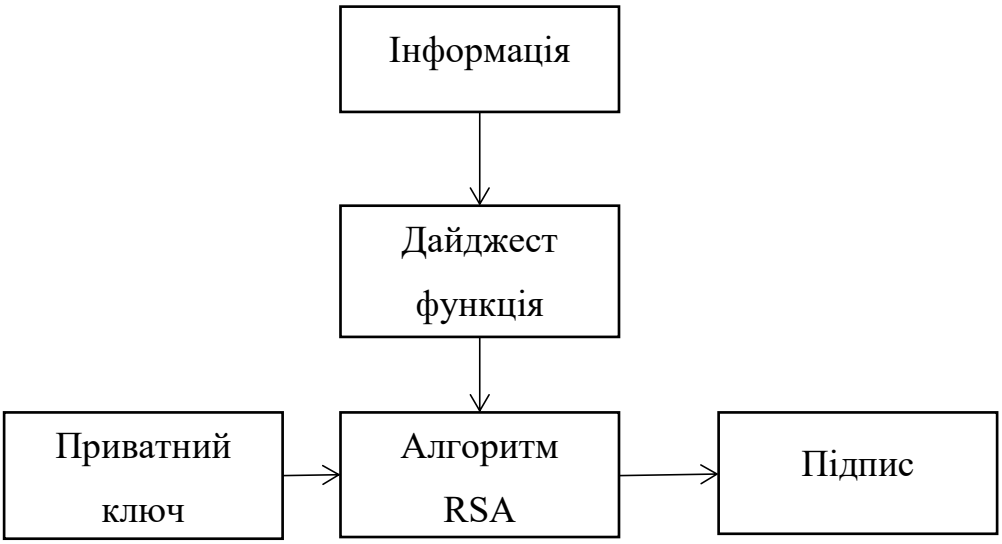


Рисунок 2.6 – Схема створення підпису RSA

Для того щоб перевірити підпис, потрібно мати декілька вхідних даних: публічний ключ, інформацію. Публічний ключ надається саме для верифікації, у випадку розроблюваного метода, публічний ключ передається при встановленні зв'язку між мобільним застосунком та сервером. Інформація – дані що було підписано. Оскільки саме сервер надає ці дані, передавати їх йому не потрібно. При отриманні підпису, сервер має усе що потрібно для верифікації даних. Після того як сервер підтвердить що підпис дійсний, він вносить в базу даних. На рисунку 2.7 зображено схема перевірки підпису RSA.

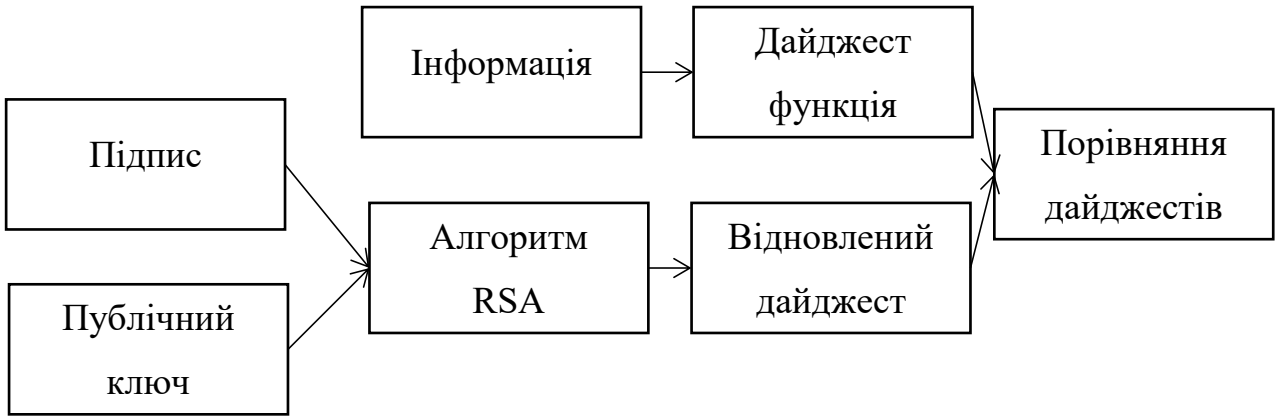


Рисунок 2.7 – Схема перевірка підпису RSA

До одного мобільного застосунку є можливість прив’язати велику кількість ресурсів. В разі отримання запиту на токен, користувач в мобільному застосунку обирає чи надавати його, при цьому користувач бачить IP–адресу яка зробила запит (аби перевірити чи ця адреса відповідає адресі з якої робить запит користувач).

2.3 Структура токена

За генерацію токенів відповідає мобільний застосунок. Токен складається із двох частин, публічної та приватної. Структура токена зображена на рисунку 2.8.

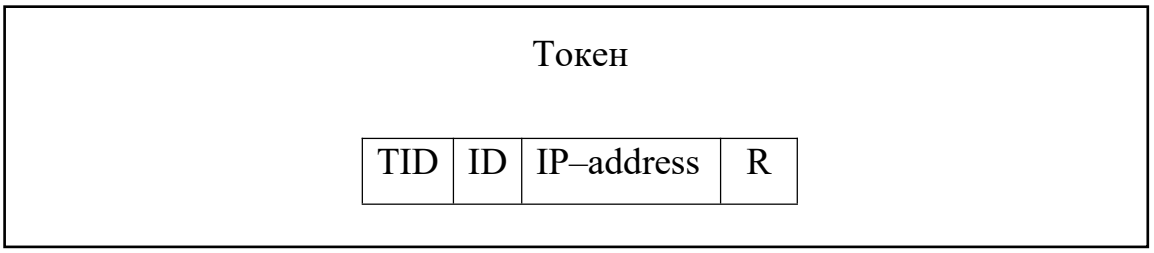


Рисунок 2.8 – Структура токена

Токен складається із:

- TID – ідентифікатор токена;
- ID – ідентифікатор користувача (логін);
- IP – address – адреса з якої здійснений запит;
- R – права доступу;

Перед генеруванням токену, мобільний застосунок має отримати запит із даними. Запит дійсний лише 60 секунд, якщо пройшло 60 секунд користувач уже не зможе підтвердити запит та створити токен. Значення ідентифікаторів TID є цілим числом, TID унікальним для кожного токену, ID користувача.

TID також є важливим, оскільки комбінації інших параметрів може повторятися, що призводитиме до того що підписи будуть повторюватися. Із

кожним новим токеном TID буде збільшуватись на 1, за приклад було взяти параметр nonce що використовується у транзакціях в блокчейні [1].

У токені користувача не зберігаються змінні параметри, однак термін придатності для токенів обмежений та контролюється серверною частиною. Сервер зберігає додаткові поля для всіх токенів у базі даних. Додатковим полем є expiration time. Після певного проміжку часу вказаному в параметрі токен стане не дійсним, якщо цей параметр встановлено.

IP–адреса слугує для перевірки того хто використовує токен, тобто токен можливо використовувати лише з певною адресою, що дає можливість контролювати IP– адресу на яку приймає запит на отримання токена та покращує захищеність.

Дані підписуватимуться за допомогою приватного ключа який було створено при встановленні зв’язку, а публічний ключ відправляється на сервер. За допомогою публічного ключа сервер зможе перевіряти чи був токен підписаний власником пристрою.

Токен автентифікації не лише автентифіковує користувача. Токен ще й містить додаткові параметри, які вказують деталі даного токена. Для того щоб збільшити рівня безпеки доцільно додати й можливість розмежування прав доступу, щодозволить ще більше контролювати можливості токена.

Будь–який користувач може сформувати токени для автентифікації через мобільний застосунок. Для одного користувача може бути створено декілька токенів, з різними правами доступу. Завдяки використанню токенів з обмеженими правами на доступ користувачі можуть підвищити рівень безпеки. Наприклад, використання токенів в яких встановлено права доступу лише читання довзолить обмежити доступ до змін даних.

Для створення нового токена користувач правила доступу генеруватимуться автоматично. Вибір правил доступу здійснюється з множини усіх можливих наборів правил доступу.

2.4 Висновки до розділу

У цьому розділі виконано математичний опис процесу односторонньої автентифікації користувачів із асиметричним шифруванням. Для формалізованого математичного опису системи використано теоретико–множинний підхід.

Розроблено метод автентифікації користувачів із використанням токенів. Описано процес автентифікації за розробленим методом. Детально описано особливості використання токену при автентифікації користувачів за розробленим методом.

З’ясовано, що необхідна умова для використання розробленого методу автентифікації виконується. Для розробленого методу автентифікації здійснено розмежування прав доступу на основі токенів користувачів. Перейдемо до опису алгоритмів роботи засобу автентифікації користувачів із токенами.

3 ЗАСІБ АВТЕНТИФІКАЦІЇ КОРИСТУВАЧІВ НА ОСНОВІ ТОКЕНІВ

3.1 Обґрунтування вибору засобів розробки

Перед тим як розпочати розробку потрібно обрати мову програмування, середовище розробки. Розробити потрібно сервер та мобільний застосунок. Серед мов програмування для розробки сервера було виділено 3: Java, JavaScript, Python. JavaScript не пропонує багато пакетів для аналізу даних і вбудованих функцій в порівнянні з Python. Популярність Python зобов'язана великій кількості різноманітних бібліотек з відкритим вихідним кодом, що дозволяє знайти використання у будь–якому напрямку. Також Python має різні інтерпретатори (наприклад, Jython дозволяє написання програм використовуючи віртуальну машину Java). Для розробки було обрано мову програмування Python.

При встановленні Python встановлюється IDLE та Shell. Вбудоване середовище IDLE підходить лише для простих програм, порівняно з іншими середовищами розробки, вбудоване середовище немає підказок, не зберігає файли автоматично, немає інтеграції з Git. Чудовими середовищами розробки Python є Visual Studio та Pycharm. Pycharm порівняно з Visual Studio є середовищем розробленим для Python, в той час Visual Studio середовище що лише підтримує Python і він чудово підійде коли вже був встановлений для інших потреб. Pycharm надає доступ до безлічі плагінів як від користувачів так і від компаній, при будь–якій зміні файлів одразу зберігає їх, має інтеграцію з Git, дозволяє для проектів створювати віртуальні середовища, щоб не навантажувати одне середовище безліччю бібліотек, також має чудовий засіб налагодження (debug) коду. Середовищем розробки було обрано PyCharm.

Для розробки мобільних застосунків є досить великий вибір, зокрема: Java, Kotlin, Swift. При розробці на цих мовах програмування потрібно розробляти розробляти для різних систем окремо Java, Kotlin(Android), Swift

(iOS), тобто розробка не володіє кросплатформеністю. За цим критерієм можна виділити Flutter, React–Native. Розробником Flutter є Google, він швидко розвивається та розширю свої межі застосування. У Flutter окрім мобільних застосунків є можливість розробляти веб–додатки, застосунки для Windows, Linux, macOS. Тепер можна перейти до планування розробки.

3.2 Структура засобу автентифікації

Засіб автентифікації користувачів повинен виконувати автентифікацію користувача розробленим методом автентифікації. Засіб автентифікації користувачів представлений у вигляді серверної частини та мобільного застосунку.

Серверна частина засобу складається з декількох частин: блоку керування, блок перевірки токенів, бази даних. Архітектура розробленого засобу автентифікації для серверної частини зображена на рисунку 3.1

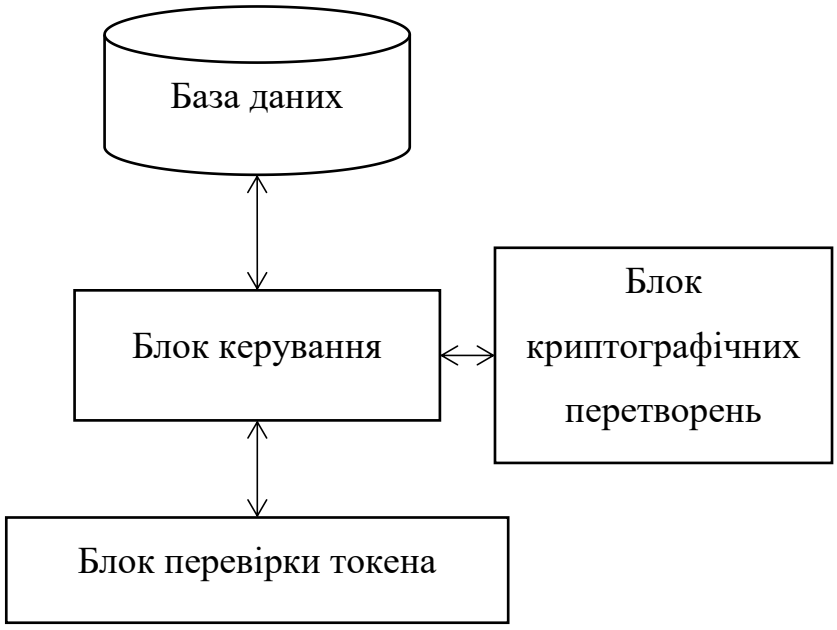


Рисунок 3.1 – Структура серверної частини засобу

Блок керування відповідає за обробку даних. Блок керування – центральний блок серверної частини, що взаємодіє з усіма іншими блоками.

Блок керування виконує послідовність команд для отримання результату (відповіді користувачу), після чого відправляє його, в залежності від запиту це може бути як HTML–сторінка, так і просто відповідь із кодом що повідомить про результат взаємодії. Блок керування також забезпечує взаємодію засобу із базою даних та формує відповідні SQL–запита до бази даних.

Блок перевірки токенів з блоку керування отримує токен користувача, який необхідно перевірити. Блок перевірки токенів через блок керування взаємодіє з базою даних, де виконує пошук токену, що відповідає наданому користувачем. Для токену виконується перевірка на можливість його подальшого використання. Результати перевірки надаються до блоку керування.

Блок криптографічних перетворень проводить перевірку підпису за допомогою публічного кдюча.

Мобільний застосунок складається із таких частин: блоку керування, користувацький інтерфейс, блок для роботи із даними на пристрої, блок криптографічних перетворень, блок зчитування QR коду. Архітектура розробленого засобу автентифікації для мобільного застосунку зображена на рисунку 3.2

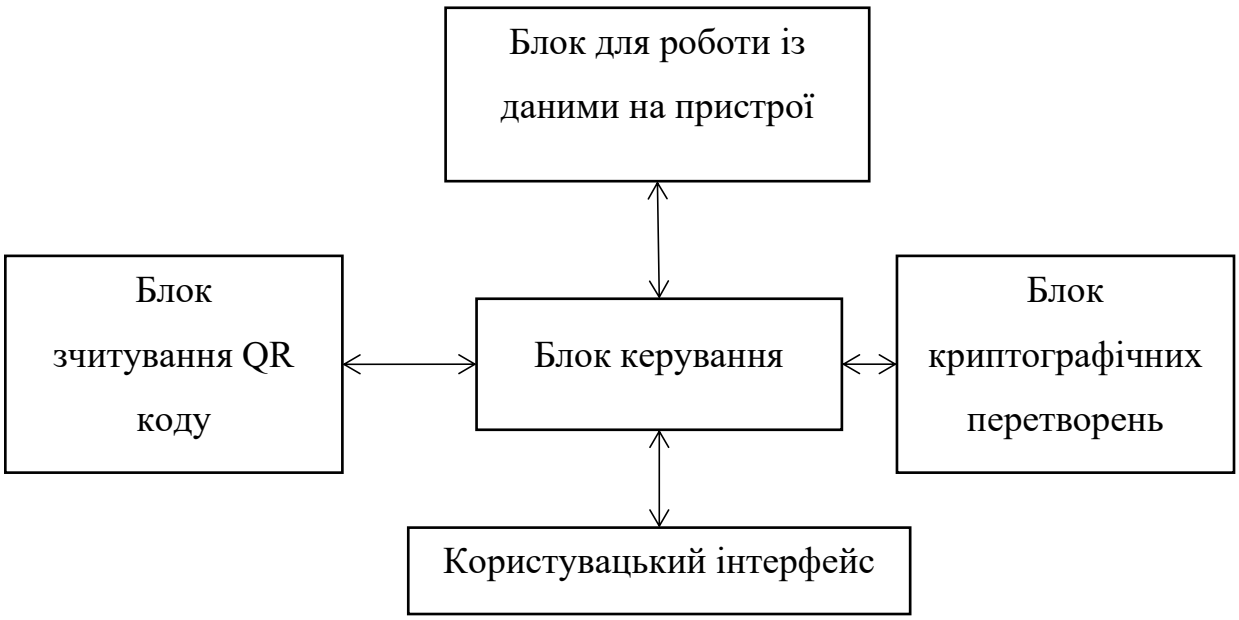


Рисунок 3.2 – Структура мобільного застосунку

Блок криптографічних перетворень у мобільному застосунку створює публічний та приватний ключі, підписує дані для подальшої перевірки на сервері. Блок керування відповідає за обробку даних, виконання послідовності команд. Користувацький інтерфейс дозволяє користувачу вказати необхідні дані для автентифікації та відображає усі потрібні дані. В нього входять усі візуальні частини мобільного застосунку (widget–и). Користувацький інтерфейс складається із декількох головних віджетів: головна сторінка, сторінка із даними користувача, сканування QR–коду, сторінка із запитами на отримання токена. Взаємозв'язок сторінок зображено на рисунку 3.3.

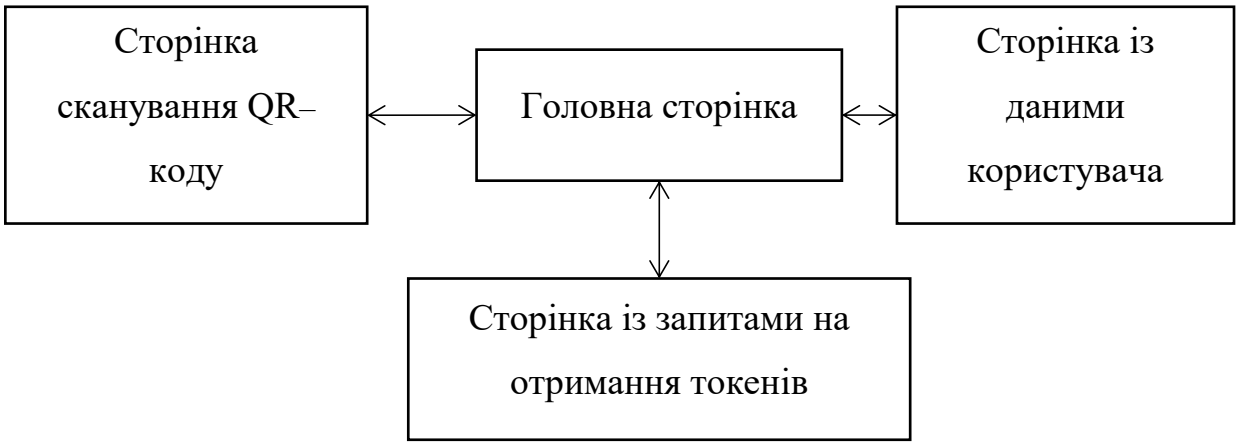


Рисунок 3.3 – Взаємодія між сторінками мобільного застосунку

Блок зчитування QR–коду взаємодії із камерою. Перед тим як користуватись камерою надсилається запит користувачу на отримування доступу до камери. Після отримання доступу починається процес зчитування даних із камери та пошук QR–коду. Якщо код було знайдено та він має формат JSON, дані відправляються в блок керування, який виконує спробу встановлення зв'язку із сервером.

3.3 Алгоритм засобу автентифікації

При запуску мобільного застосунку іде перевірка чи вказано параметри користувача. Якщо користувач вже вказував до цього дані то відкривається

головна сторінка, в іншому випадку відкривається сторінка для вказання даних користувача. Після того як користувач вказав дані відкривається головна сторінка. На головній сторінці користувач користувач має можливість 3 взаємодій: відкрити сторінку зміни даних користувачів, сканування QR–коду та встановлення зв'язку, обрати ресурс запити на доступ до якого користувач бажає переглянути. Узагальнений алгоритм мобільного застосунку зображено на рисунку 3.4.

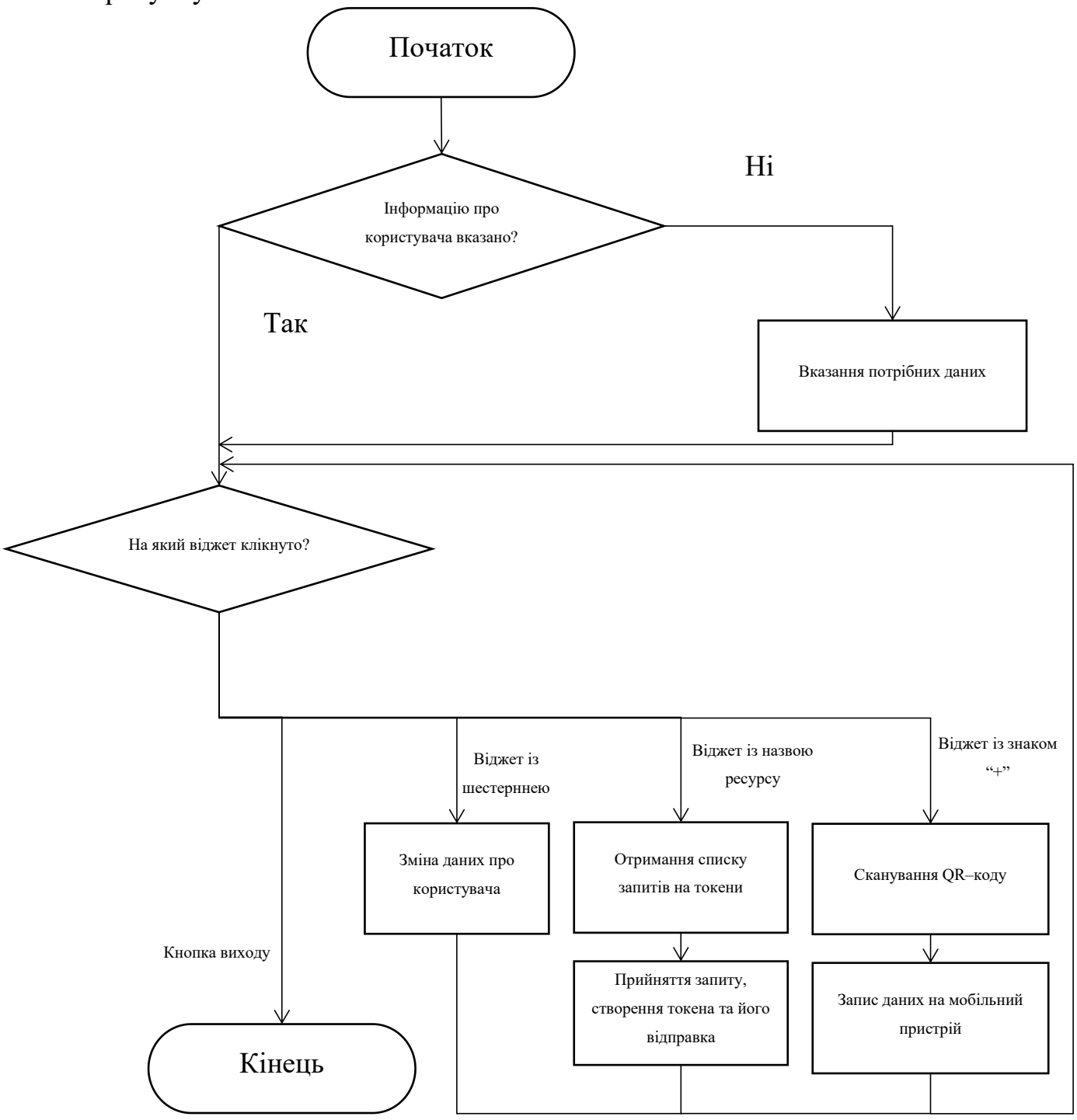


Рисунок 3.4 – Узагальнений алгоритм роботи мобільного застосунку

Якщо користувач хоче змінити дані що були вказані ним йому потрібно нажати на кнопку із шестернею. Алгоритм роботи при нажатті на шестерню зображено на рисунку 3.5.

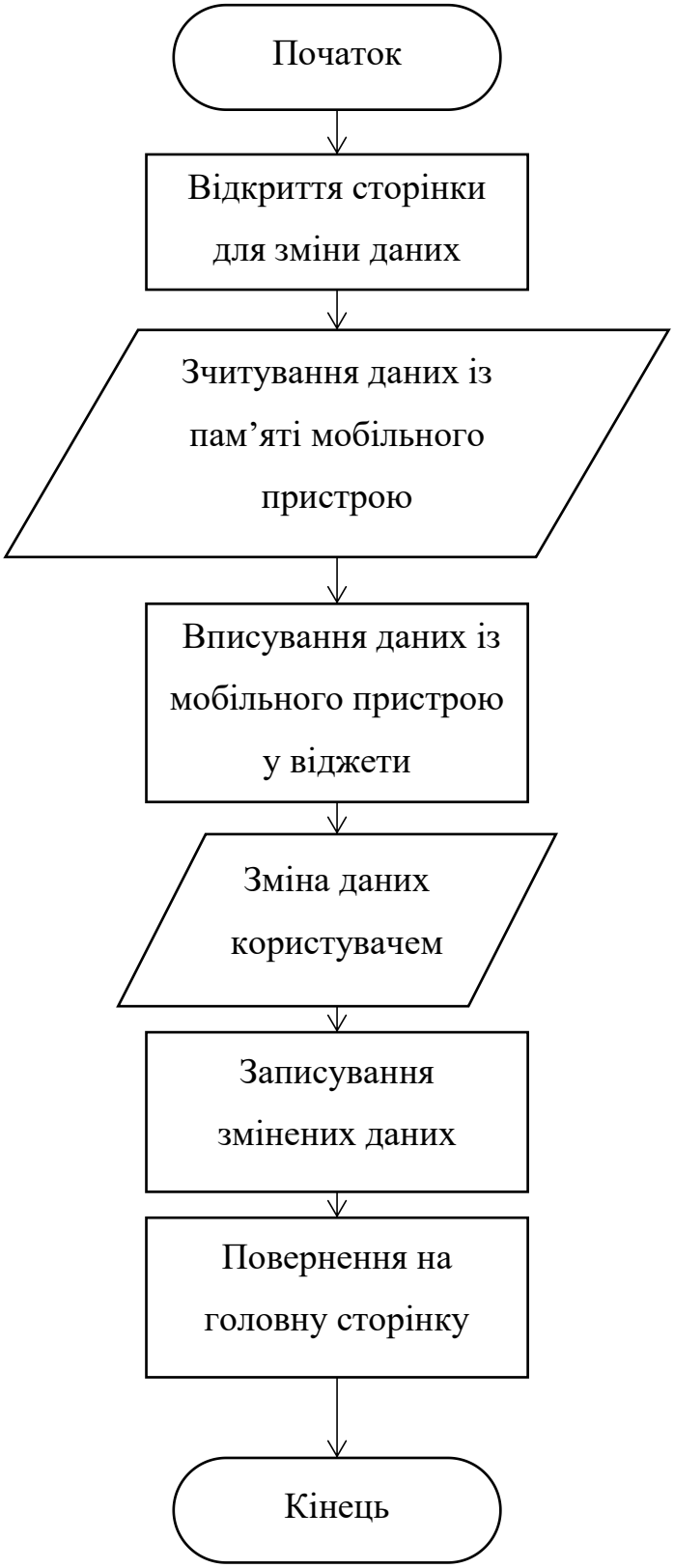


Рисунок 3.5 – Алгоритм зміни даних користувача в мобільному застосунку

Якщо користувач не використовував мобільний застосунок раніше то ніяких зв’язаних ресурсів у ньому не буде. Для того щоб зв’язати ресурс із мобільним застосунком потрібно на головній сторінці нажати на віджет із знаком “+”. Алгоритм додавання ресурсу зображено на рисунку 3.6.

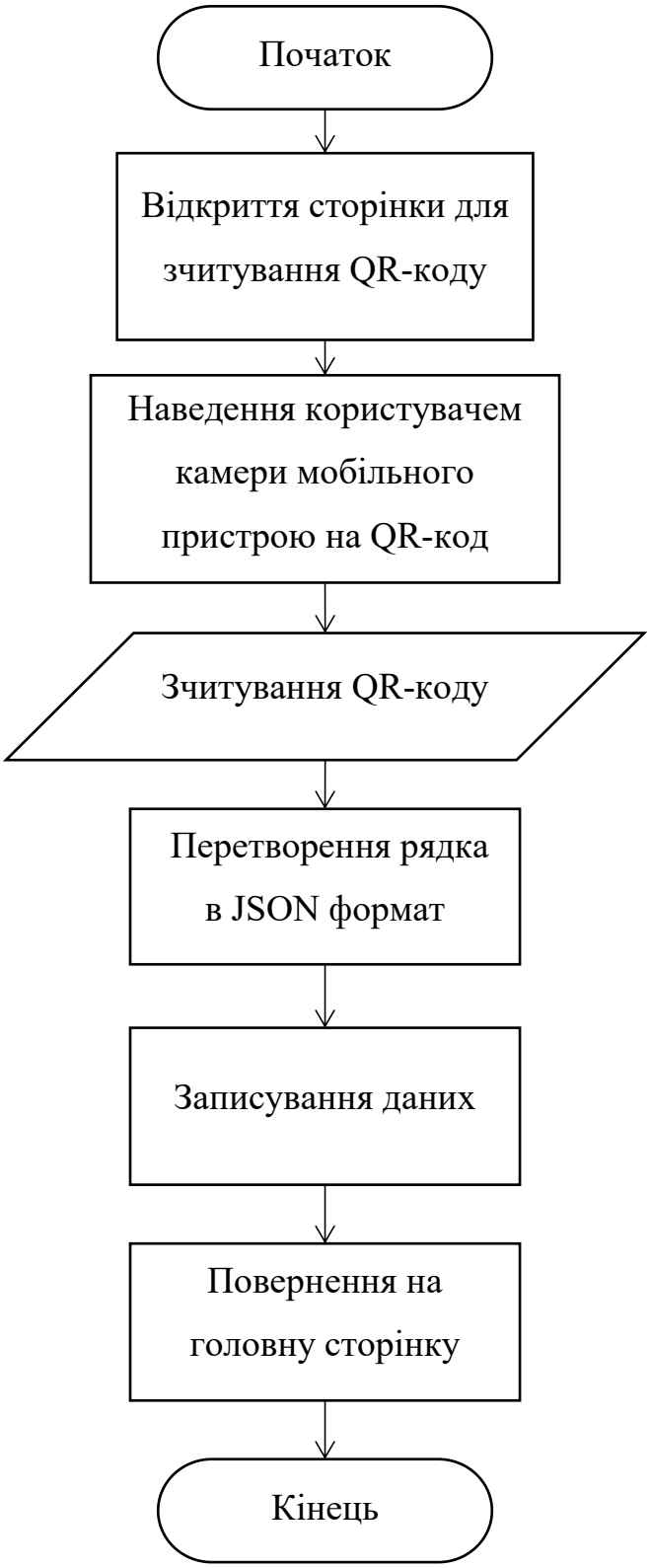


Рисунок 3.6 – Алгоритм додавання ресурсу в мобільному застосунку

Якщо користувач уже зв’язував мобільний застосунок із ресурсами то на головному екрані буде відображатись кнопки із назвами ресурсів. Алгоритм взаємодії із запитами зображено на рисунку 3.7.

Рисунок 3.7 – Алгоритм прийняття запиту на отримання токена

Серверна частина знаходиться в очікувані, до поки не отримає запит, після чого в залежності від умов обробляє його та дає відповідь. Основними посиланнями для обробки використовується: ‘/’, ‘/login’. ‘/login’ – потрібен для реєстрації або щоб залогінитись, містить поля для таких даних: login, password. У випадку не існування користувача із вказаним login-ом, проводиться реєстрація користувача із введеними даними. ‘/’ потрібна для відображення інформації, тестування роботи токена. На рисунку 3.8 зображено алгоритм роботи сервера.

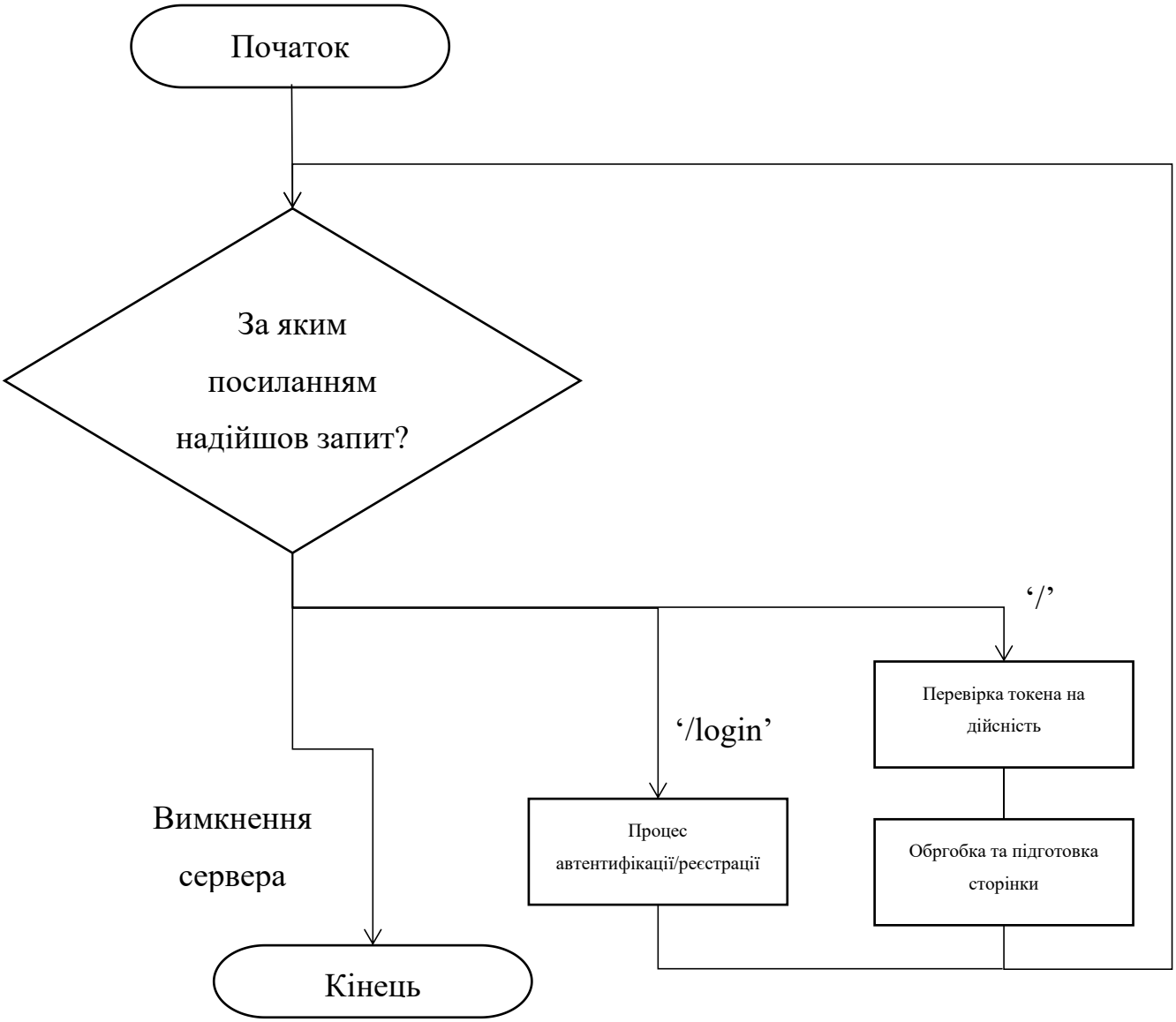


Рисунок 3.8 – Узагальнений алгоритм роботи серверу

В процесі автентифікації, якщо користувач ввів правильні дані, створюється запит на отримання токена, який згодом може бути підтверджений в мобільному застосунку. Після підтвердження токен отримає сервер. При реєстрації створюється QR-код для встановлення зв’язку між мобільним застосунком та сервером. На рисунку 3.9 зображено алгоритм автентифікації/реєстрації.

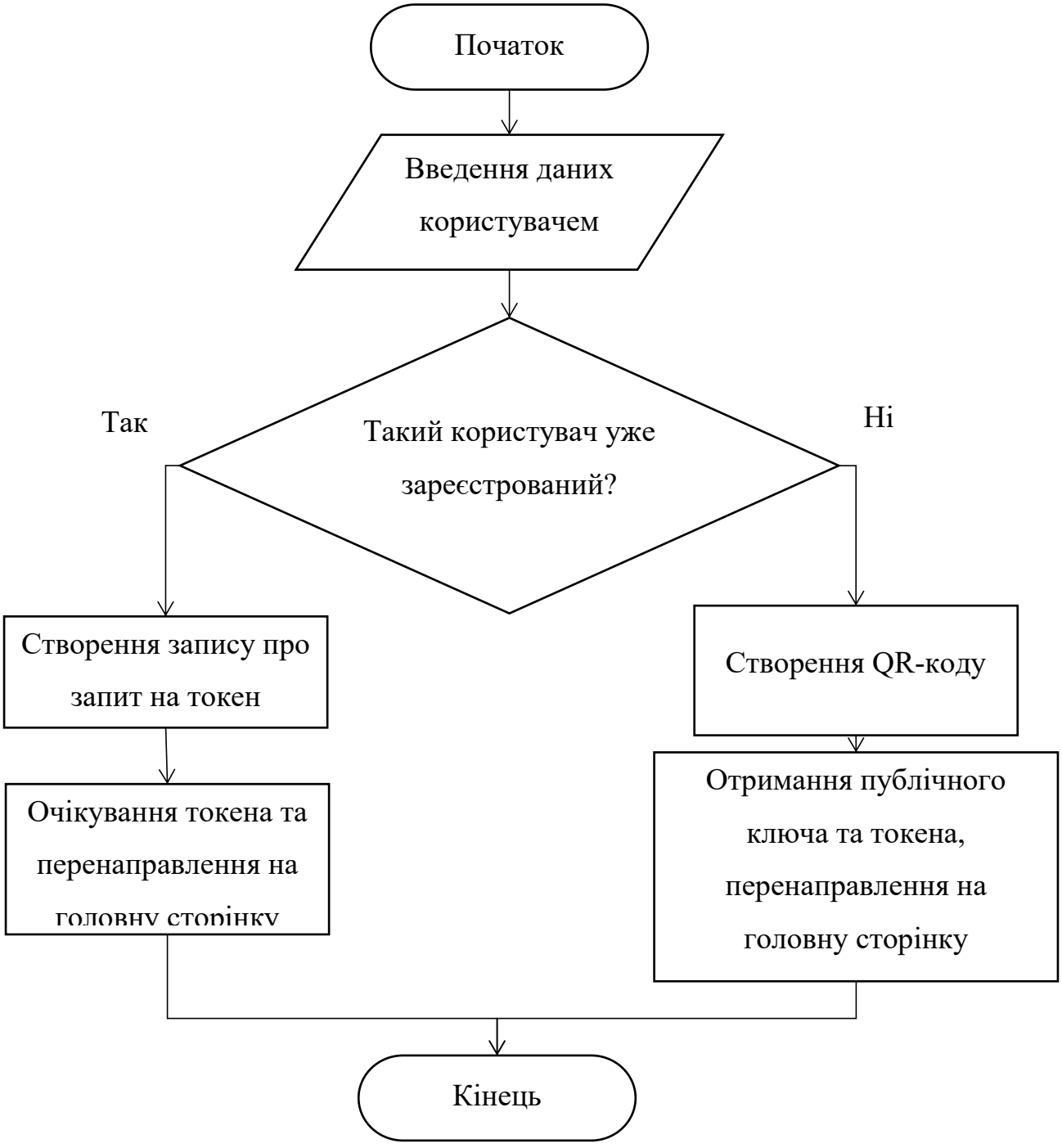


Рисунок 3.9 – Узагальнений алгоритм автентифікації/реєстрації

При спробі перейти на головну сторінку перевіряються дані що були передані серверу (токен). Сервер зчитує дані із бази даних та перевіряє підпис. На рисунку 3.10 зображено алгоритм роботи сервера при запиті на головну сторінку.

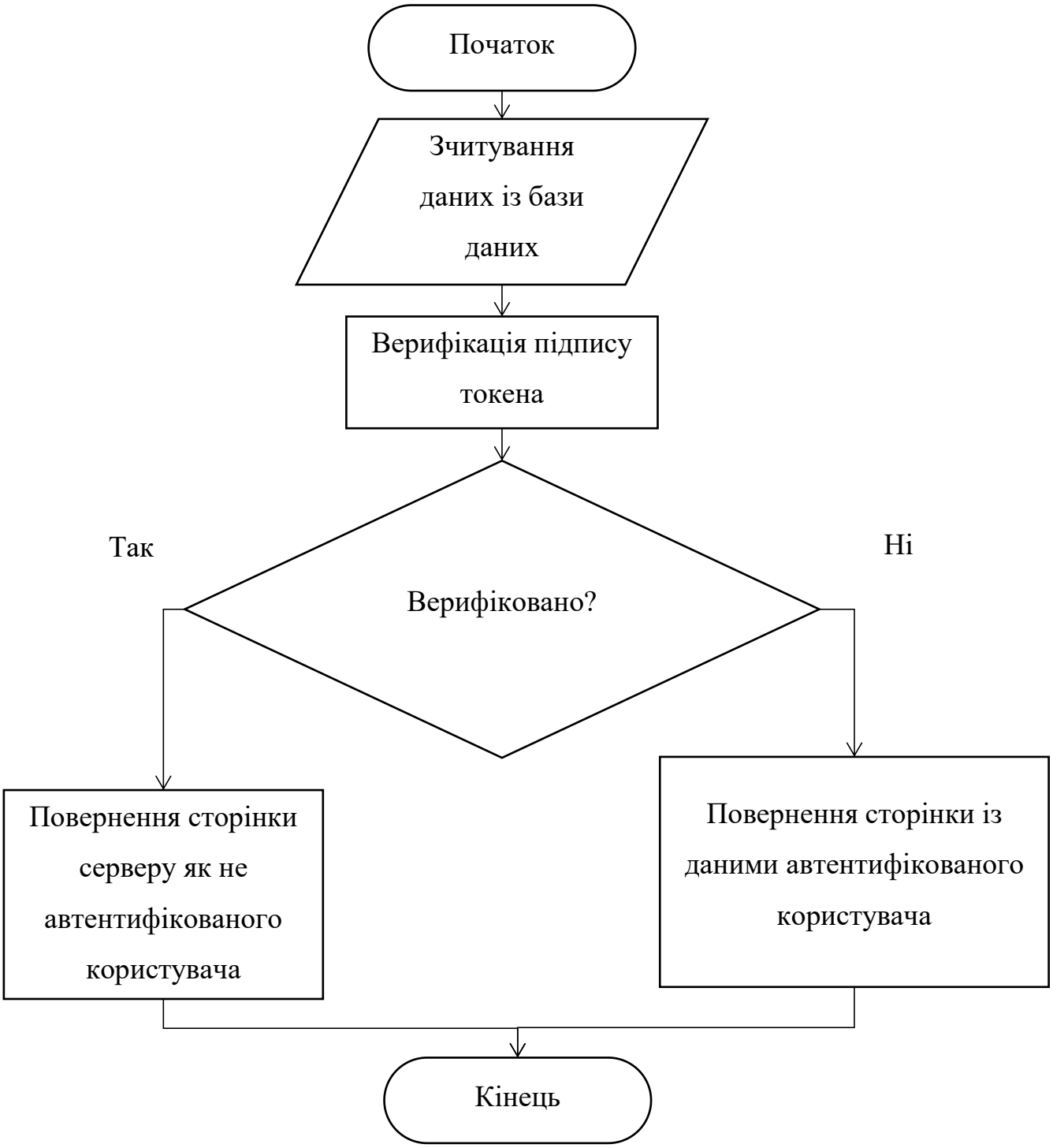


Рисунок 3.10 – Узагальнений алгоритм сервера при запиті на головну сторінку

Одним із важливих елементів сервера є база даних. База даних складатиметься із 3 таблиць.

Основною є таблиця із login-ом та паролем, вона заповняється при реєстрації користувача. Login є початковим ідентифікатором користувача, тому він дуже важливий.

Наступною є таблиця в якій записується публічний ключ користувача, вона зв’язана із 1 таблицею один до одного, оскільки користувач може мати лише один мобільний застосунок (тобто 1 публічний ключ) прив’язаний до користувацького аккаунту.

3 таблицею є список токенів та їхні дані, вона також зв’язана із 1 таблицею багато до одного, оскільки користувач може мати багато токенів для доступу до користувацьких даних.

На рисунку 3.11 зображено взаємозв’язок таблиць.

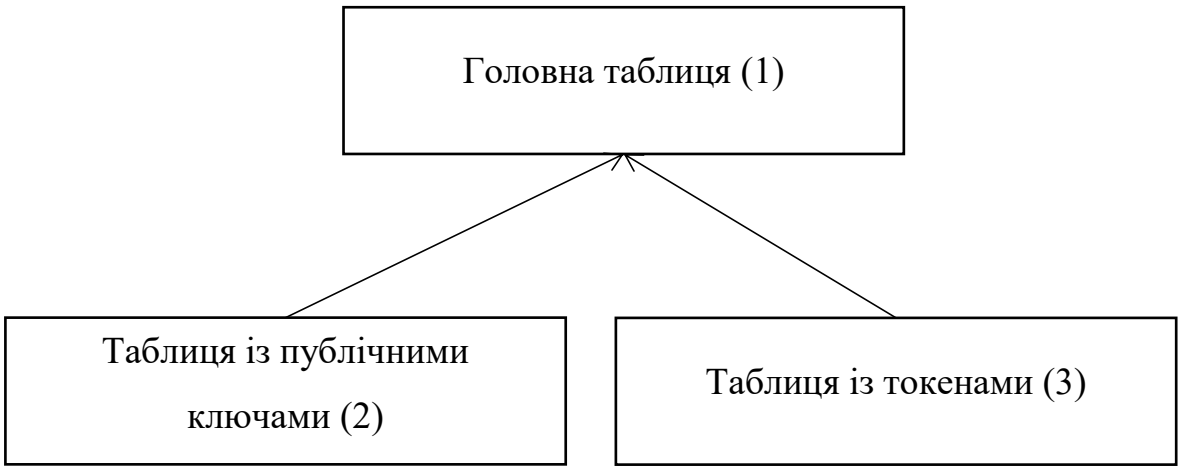


Рисунок 3.11 – Взаємозв’язок таблиць

Взаємодія із базою даних відбувається повістю через сервер

Отже тепер можна перейти до етапу тестування мобільного застосунку та його взаємодії із серверною частиною.

3.4 Висновок до розділу

Отже, у цьому розділі розроблено узагальнену архітектуру засобу автентифікації користувачів із токенами, що передбачає клієнт–серверну реалізацію засобу. На основі архітектури, сформовано узагальнений алгоритм роботи засобу автентифікації. Для засобу автентифікації розроблено алгоритми генерації та перевірки токенів користувачів. Розроблено алгоритми автентифікації користувача для серверної та клієнтської частин.

Розроблені алгоритми дозволяють перейти до їхньої програмної реалізації, що складатимуть засіб автентифікації користувачів.

4 ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ РОЗРОБКИ

Чим більше функцій має застосунок, то важче його перевірити вручну. Автоматичні тести допомагають переконатися, що програма працює правильно, перш, ніж публікувати його, зберігаючи при цьому функції та швидкість виправлення помилок.

У Flutter доступно три типи тестів [20]: unit test, widget test, integration test. Юніт тест (unit test) – перевіряє одну функцію, метод або клас. Віджет тест (widget test) – перевіряє один віджет. Інтеграційний тест (integration test) – перевіряє програму в цілому або велику частину програми. Тестування віджетів входить в автоматичне інтеграційне тестування.

Тестування відбуватиметься в два етапи: перший етап – модульне тестування, другий етап – інтеграційне тестування. Інтеграційне тестування відбуватиметься вручну, в той час як блокове – автоматизовано.

4.1 Блокове тестування

Для створення юніт–тестів використовується вбудовані можливості Flutter[20]. Тести створюються за допомогою функції test, яка приймає два аргументи, першим аргументом є рядок що використовуватиметься як назва тесту, другим аргументом є функція що й виконуватиметься для тестування. Для об’єднання тестів використовується використовується функція group (аргументи ті самі що й у функції test). Для перевірки даних використовується функція expect, вона приймає два аргументи: очікуваний результат, отриманий результат [20].

Для перевірки в модульному тестуванні було обрано криптографічні функції та функції пов’язані із взаємодією з даними. Серед криптографічних функцій підпис повідомлень, верифікація повідомлень. Також перевірці підлягають функції що записують та зчитують дані.

Щоб перевірити підписання та верифікацю даних нам спочатку потрібно створити ключі (приватний, публічний), після чого створити повідомлення яке буде підписано. У функцію підпису передати: повідомлення, хеш-функцію яку використовувати алгоритму, приватний ключ. Функція верифікації вимагає підписані дані, повідомлення, хеш-функція, публічний ключ. Верифікаційна функція повертає бульову зміну, яка вкаже при пройдено верифікацію. Узагальнений алгоритм тесту підписування та верифікації зображено на 4.1.

Рисунок 4.1 – Узагальнений алгоритм тесту підписання/верифікації

Тест із читання та запису інформації проводить запис усіх можливих типів даних. Після того як дані були записані проводиться зчитування даних та порівняння їх із вхідними даними.

Для тестування функціональних можливостей (основних) було створено 2 тести. Результат тестування зображено на рисунку 4.2.

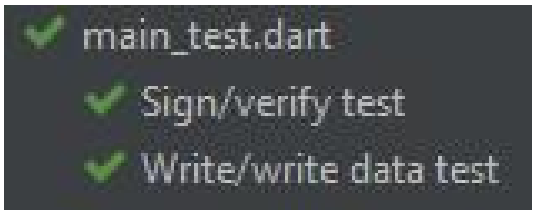


Рисунок 4.2 – Результат блокового тестування функцій

Наступним етапом є тестування віджетів. Для тестування віджетів використовується функція testWidgets, який так само як test приймає 2 аргументи, назву теста, функцію для тестування. Різниця в тому, що функція в testWidgets приймає 1 аргумент класу WidgetTester, за допомогою якого відбувається тестування.

Перед тим як почати взаємодіяти із віджетом потрібно його побудувати, для цього використовується функція tester.pumpWidget, в яку передаємо клас віджета. Оскільки взаємодіє із інтерфейсом не людина а код він не відображається на екрані, ійому потрібно знати із чим взаємодіяти. Для цього використовується клас find. У даному класі є різні методі для пошуку елементів інтерфейсу. Серед них функція text, яка дозволяє знаходити віджети що містять певний текст. Але є такі елементи інтерфейсу що не мають тексту а мають лише іконки, для їхнього пошуку було використаноскористатися функцією byIcon. При пошуку функції можуть повертати декілька віджетів, тому потрібно бути обережним із вибором функцій для пошуку.

Для тестування віджетів потрібно слідувати алгоритму зображеному на рисунку 4.3.

Рисунок 4.3 – Алгоритм тестування віджета

На рисунку 4.4 зображено результати тестування віджетів 2 сторінок.

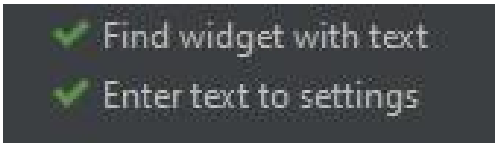


Рисунок 4.4 – Результат блокового тестування віджетів

Повне автоматичне інтеграційне тестування неможливе, оскільки мобільному застосунку потрібно зчитувати QR–код, що неможливо реалізувати.

На це є декілька причин: для зчитування QR–коду потрібен доступ до камери, потрібно щоб камера побачила QR–код. Отже інтеграційне тестування проводитеметься вручну.

В результаті модульного тестування усі тести пройшли успішно. Наступним кроком є інтеграційне тестування.

4.2 Інтеграційне тестування

Було виконано інтеграційне тестування, що дозволило перевірити коректність роботи автентифікації та взаємодії між мобільним застосунком та серверною частинами.

Хоча Flutter має можливість автоматичного інтеграційного тестування, та воно не підходить для тестування розробленого мобільного застосунку. Оскільки автоматичне інтеграційного тестування надає можливість перевірити чи коректно працюють віджети, та не можуть допомогти із функціоналом що пов’язаний із зовні. У випадку розробленого мобільного застосунку це: сканування QR-коду, прийняття запиту на токен. Оскільки для сканування QR-коду потрібен рух мобільним пристроєм, а для прийняття запиту на токен потрібно щоб він існував та був дійсним.

Головна сторінка відображає список зв’язаних ресурсів (якщо вони є), та має дві кнопки справа внизу та список ресурсів з якими користувач уже зв’язував засіб. При цьому перехід на інші сторінки відбувається лише із головної сторінки, з інших сторінок можливо перейти лише на головну сторінку. Справа вверху знаходиться кнопка для оновлень головної сторінки. При першому відкритті застосунку із головної сторінки користувача переправить на сторінку вказання даних користувача, користувач за бажання може не вказувати їх.

На рисунку 4.5 можна побачити головну сторінку мобільного застосунку.

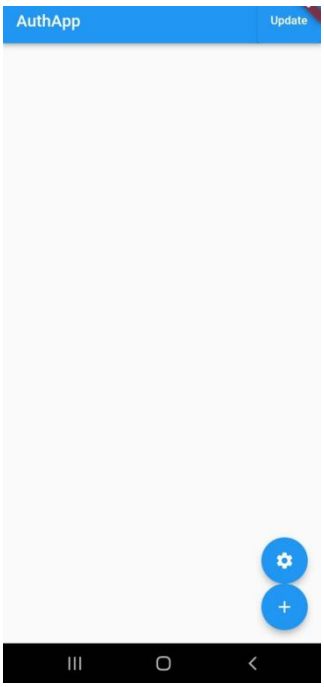


Рисунок 4.5 – Головна сторінка мобільного застосунку

Кнопка із шестернею відкриває налаштування, в яких користувач може налаштувати прізвище, ім’я та електронну пошту (рис. 4.6).

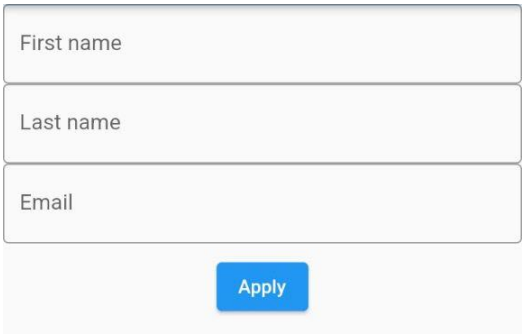


Рисунок 4.6 – Сторінка зміни/встановлення даних користувача

Наступним кроком користувач відкриває сторінку для того щоб пройти початкову автентифікацію(парольна). Після того як користува введе дані, йому потрібно нажати на кнопку “Get QR”. Якщо користувач був зареєстрований тоді відправляється запит на отримання токена, в іншому випадку створюється QR– код для встановлення зв’язку. На рисунку 4.7 зображено сторінку парольної автентифікації.

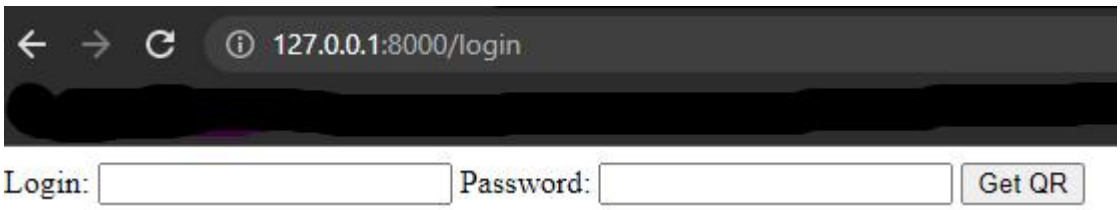


Рисунок 4.7 – Вигляд сторінки парольної автентифікації

Кнопка із знаком “+” запитує надання доступу до камери та починає сканування на QR–код. Якщо QR–код буде знайдено, дані зчитаються та камера закриється, почнеться процес встановлення зв’язку. Сканування проводиться з невеликим періодом та у визначеній зоні. Зону сканування видно за декількома ознаками: зона сканування світліша, кути зони сканування обведені червоним. Сканування QR–коду відбувається дійсно швидко та легко. На рисунку 4.8 зображено вигляд при спробі сканування тестового QR–коду.

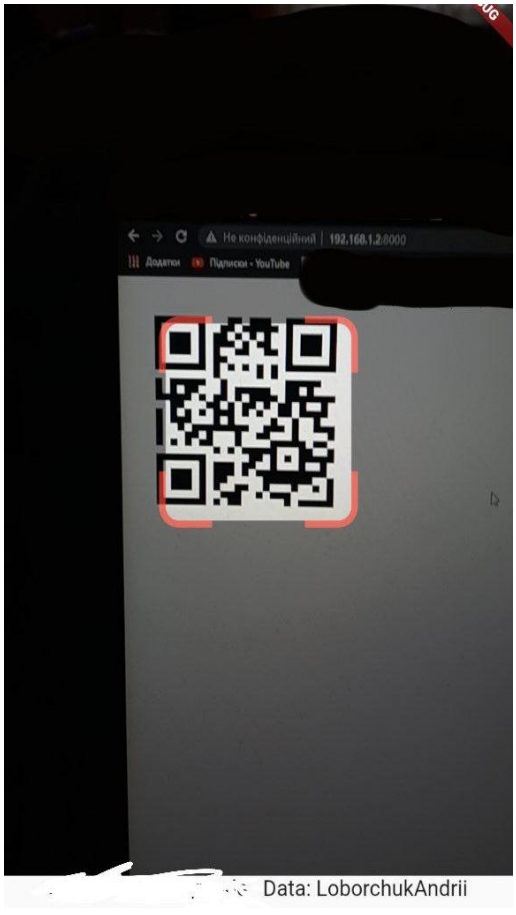


Рисунок 4.8 – Вигляд застосунку при спробі сканування QR–коду

Після того як було встановлено зв’язок на головну сторінку добавляється віджет із даним ресурсом(див. рис. 4.9).

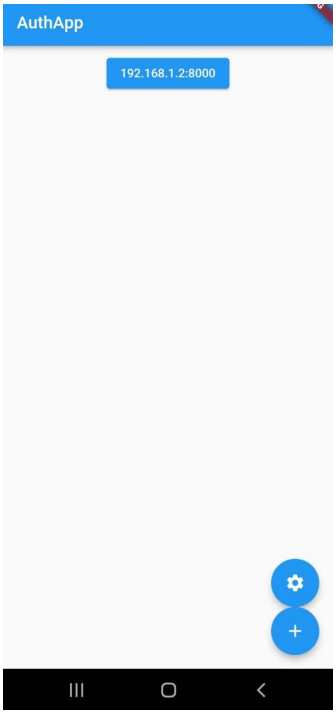


Рисунок 4.9 – Головна сторінка мобільного із одним прив’язаним ресурсом

Після того як користувач відправив запит на отримання токена, на головній сторінці користувач натискає на кнопку із назвою ресурсу, токен якого користувач хоче прийняти та надати його (рис. 4.10).

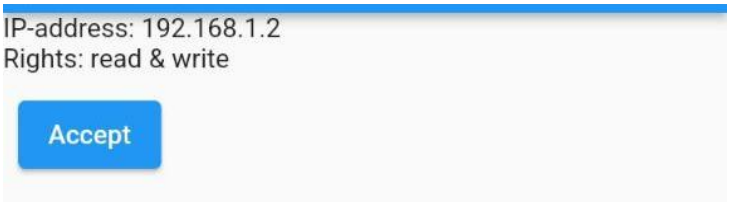


Рисунок 4.10 – Приклад запиту на отримання токена та кнопки прийняття

Після чого користувача переправляє на головну сторінку. На рисунку 4.11 зображено сторінку.

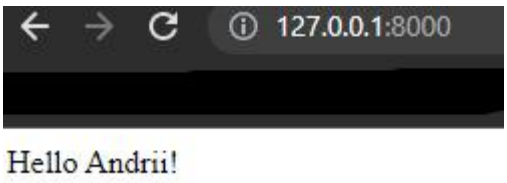


Рисунок 4.11 – Вигляд сторінки коли користувача автентифіковано

В результаті тестування помилок не виникло, мобільний застосунок використовує метод автентифікації та справно автентифіковує користувача.

4.3 Висновки до розділу

Виконано розробку клієнт–серверного програмного застосунку, що ввиконує автентифікацію користувача. Здійснено блокове тестування та інтеграційне тестування застосунку. Результати тестування підтвердили коректність роботи модуля автентифікації. В результаті тестування перевірено коректність реалізації методу на основі токенів.

5 ЕКОНОМІЧНА ЧАСТИНА

Науково–технічна розробка має право на існування та впровадження, якщо вона відповідає вимогам часу, як в напрямку науково–технічного прогресу та і в плані економіки. Тому для науково–дослідної роботи необхідно оцінювати економічну ефективність результатів виконаної роботи.

Магістерська кваліфікаційна робота «Метод та засіб автентифікації користувачів на основі токенів» відноситься до науково–технічних робіт, які орієнтовані на виведення на ринок (або рішення про виведення науково–технічної розробки на ринок може бути прийнято у процесі проведення самої роботи), тобто коли відбувається так звана комерціалізація науково–технічної розробки. Цей напрямок є пріоритетним, оскільки результатами розробки можуть користуватися інші споживачі, отримуючи при цьому певний економічний ефект. Але для цього потрібно знайти потенційного інвестора, який би взявся за реалізацію цього проекту і переконати його в економічній доцільності такого кроку.

Для наведеного випадку нами мають бути виконані такі етапи робіт:

1) проведено комерційний аудит науково–технічної розробки, тобто встановлення її науково–технічного рівня та комерційного потенціалу;

2) розраховано витрати на здійснення науково–технічної розробки;

3) розрахована економічна ефективність науково–технічної розробки у випадку її впровадження і комерціалізації потенційним інвестором і проведено обґрунтування економічної доцільності комерціалізації потенційним інвестором.

5.1 Проведення комерційного та технологічного аудиту науково–технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Метод та засіб автентифікації користувачів на основі токенів» є

оцінювання науково–технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково–технічної діяльності.

Оцінювання науково–технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням 5–ти бальної системи оцінювання за 12–ма критеріями, наведеними в табл. 5.1 [21].

Таблиця 5.1 – Рекомендовані критерії оцінювання науково–технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5–ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою

Продовження таблиці 5.1

7	Активна конкуренція великих	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї	Потрібні незначні фінансові ресурси. Джерела фінансування	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуютьс
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій	Термін реалізації ідеї від 3–х до 5–ти років. Термін окупності інвестицій більше	Термін реалізації ідеї менше 3–х років. Термін окупності інвестицій від 3–	Термін реалізації ідеї менше 3–х років. Термін окупності інвестицій менше
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь–які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково–технічного рівня та комерційного потенціалу науково–технічної розробки потрібно звести до таблиці.

Таблиця 5.2 – Результати оцінювання науково–технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	4	4	4
2. Ринкові переваги (наявність аналогів)	2	2	3
3. Ринкові переваги (ціна продукту)	2	2	3
4. Ринкові переваги (технічні властивості)	1	2	2
5. Ринкові переваги (експлуатаційні витрати)	2	2	2
6. Ринкові перспективи (розмір ринку)	2	2	2
7. Ринкові перспективи (конкуренція)	3	2	3
8. Практична здійсненність (наявність фахівців)	5	5	5
9. Практична здійсненність (наявність фінансів)	3	4	3
10. Практична здійсненність (необхідність нових матеріалів)	4	5	5
11. Практична здійсненність (термін реалізації)	4	4	4
12. Практична здійсненність (розробка документів)	4	3	4
Сума балів	36	37	40
Середньоарифметична сума балів $СБ_c$	37,7		

За результатами розрахунків, наведених в таблиці 5.2, зробимо висновок щодо науково–технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 5.3 [21].

Таблиця 5.3 – Науково–технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ ,розрахована на основі висновків експертів	Науково–технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Метод та засіб автентифікації користувачів на основі токенів» становить 37,7 бала, що, відповідно до таблиці 5.3, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

5.2 Розрахунок узагальненого коефіцієнта якості розробки

Окрім комерційного аудиту розробки доцільно також розглянути технічний рівень якості розробки, розглянувши її основні технічні показники. Ці показники по–різному впливають на загальну якість проектної розробки.

Узагальнений коефіцієнт якості ($B_н$) для нового технічного рішення розрахуємо за формулою [22]:

$$B_н = \sum_{i=1}^{k} \alpha_i \cdot \beta_i \quad , \tag{5.1}$$

де k – кількість найбільш важливих технічних показників, які впливають на якість нового технічного рішення;

αi – коефіцієнт, який враховує питому вагу і–го технічного показника в загальній якості розробки. Коефіцієнт αi визначається експертним шляхом і при цьому має виконуватись умова $\sum_{i=1}^{k} \alpha_i = 1$;

βi – відносне значення і–го технічного показника якості нової розробки.

Відносні значення β_i для різних випадків розраховуємо за такими формулами:

– для показників, зростання яких вказує на підвищення в лінійній залежності якості нової розробки:

$$\beta_i = \frac{I_{нi}}{I_{ai}} \quad , \tag{5.2}$$

де $I_{нi}$ та $I_{на}$ – чисельні значення конкретного і–го технічного показника якості відповідно для нової розробки та аналога;

– для показників, зростання яких вказує на погіршення в лінійній залежності якості нової розробки:

$$\beta_i = \frac{I_{ai}}{I_{ні}}; \quad (5.3)$$

Використовуючи наведені залежності можемо проаналізувати та порівняти техніко–економічні характеристики аналогу та розробки на основі отриманих наявних та проектних показників, а результати порівняння зведемо до таблиці 5.4.

Таблиця 5.4 – Порівняння основних параметрів розробки та аналога.

Показники (параметри)	Одиниця вимірювання	Аналог	Проектований пристрій	Відношення параметрів нової розробки до аналога	Питома вага показника
Рівень унікальності	бал	7	9	1,29	0,15
Швидкість автентифікації	с	0,1	0,05	2	0,25
Захищеність системи	%	87	92	1,06	0,1
Точність ідентифікації об'єкта	%	78	85	1,09	0,3
Зручність інтерфейсу користувача	бал	6,2	8,5	1,37	0,2

Узагальнений коефіцієнт якості (Вн) для нового технічного рішення складе:

$$B_н = \sum_{i=1}^{k} \alpha_i \cdot \beta_i = 1{,}29 \cdot 0{,}15 + 2 \cdot 0{,}25 + 1{,}06 \cdot 0{,}1 + 1{,}09 \cdot 0{,}3 + 1{,}37 \cdot 0{,}2 = 1{,}40.$$

Отже за технічними параметрами, згідно узагальненого коефіцієнту якості розробки, науково–технічна розробка переважає існуючі аналоги приблизно в 1,40 рази.

5.3 Розрахунок витрат на проведення науково–дослідної роботи

Витрати, пов'язані з проведенням науково–дослідної роботи на тему «Метод та засіб автентифікації користувачів на основі токенів», під час планування, обліку і калькулювання собівартості науково–дослідної роботи групуємо за відповідними статтями.

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно–технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [21]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \qquad (5.4)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

T_p – середнє число робочих днів в місяці, T_p=22 дні.

$$З_о = 17050{,}00 \cdot 40 / 22 = 31000{,}00 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.5 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту з дослідження засобу автентифікації користувачів	17050,00	775,00	40	31000,00
Інженер–розробник програмного забезпечення автентифікації	16550,00	752,27	36	27081,82
Технік	7280,00	330,91	12	3970,91
Всього				62052,73

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників (Зр) за відповідними найменуваннями робіт НДР на тему «Метод та засіб автентифікації користувачів на основі токенів» розраховуємо за формулою:

$$З_p = \sum_{i=1}^{n} C_i \cdot t_i \quad , \tag{5.5}$$

де Сі – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год, ti – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду Сі можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}} \quad , \tag{5.6}$$

де M_M – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо $M_м$=6700,00 грн, K_i – коефіцієнт міжкваліфікаційного співвідношення для

встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б) [21];

$К_с$ – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

$Т_р$ – середнє число робочих днів в місяці, приблизно $Т_р$ = 22 дн;

$t_{зм}$ – тривалість зміни, год.

$С_1 = 6700{,}00 \cdot 1{,}10 \cdot 1{,}65 / (22 \cdot 8) = 69{,}09$ *грн.*

$З_{р1} = 69{,}09 \cdot 6{,}25 = 431{,}84$ *грн.*

Таблиця 5.6 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
Підготовка робочого місця інженера–розробника програмного забезпечення	6,25	2	1,10	69,09	431,84
Інсталяція програмного забезпечення середовища розробки	5,00	3	1,35	84,80	423,98
Формування кодів програмних блоків автентифікації	9,00	5	1,70	106,78	961,03
Формування бази даних дослідження засобу автентифікації користувачів	15,00	2	1,10	69,09	1036,41

Продовження таблиці 5.6

Контроль проходження програмних експериментів автентифікації	7,50	4	1,50	94,22	706,64
Всього					3559,90

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (5.7)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 11%.

$З_{дод}$ = (62052,73 + 3559,90) · 11 / 100% = 7217,39 грн.

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (5.8)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

Зн = (62052,73 + 3559,90 + 7217,39) · 22 / 100% = 16022,60 грн.

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою «Метод та засіб автентифікації користувачів на основі токенів».

Витрати на матеріали (*М*), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$M = \sum_{j=1}^{n} H_j \cdot Ц_j \cdot K_j - \sum_{j=1}^{n} B_j \cdot Ц_{в\,j}, \quad (5.9)$$

де H_j – норма витрат матеріалу j–го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j–го найменування, грн/кг;

K_j – коефіцієнт транспортних витрат, (K_j = *1,1 … 1,15*);

B_j – маса відходів j–го найменування, кг;

$Ц_{вj}$ – вартість відходів j–го найменування, грн/кг.

$M_1 = 3,0 \cdot 219,00 \cdot 1,11 – 0 \cdot 0 = 729,27$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 5.7 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, грн	Норма витрат, кг	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Папір офісний UPM–Kymmene A4 80 г/м NEW Future Laser білий	219,00	3,0	0	0	729,27
Папір Crystal A5 80 г/м PRO білий	62,00	4,0	0	0	275,28
Набір настільний Transformers BumbleBee Movie TF19–205 KITE	120,00	3,0	0	0	399,60

Продовження таблиці 5.7

Набір канцелярський Milan Silver (08737)	449,00	3,0	0	0	1495,17
Тонер IPM HP LJ P1005/1006/1505/CANON LBP–3010/3020 (TSH87B) black	518,00	1,0	0	0	574,98
Диск оптичний CD–RW	22,00	3,0	0	0	73,26
USB флеш накопичувач Transcend 64Gb JetFlash 700 (TS64GJF700)	279,00	1,0	0	0	309,69
Всього					3857,25

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Метод та засіб автентифікації користувачів на основі токенів», розраховуємо, згідно з їхньою номенклатурою, за формулою:

$$К_в = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j \qquad (5.10)$$

де $Н_j$ – кількість комплектуючих j–го виду, шт.;

$Ц_j$ – покупна ціна комплектуючих j–го виду, грн;

$К_j$ – коефіцієнт транспортних витрат, ($К_j = 1,1 \ldots 1,15$).

$К_в = 1 \cdot 4260,00 \cdot 1,11 = 4728,60$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 5.8 – Витрати на комплектуючі

Найменування комплектуючих	Кількість, шт.	Ціна за штуку, грн	Сума, грн
Графічний адаптер: – тип відеокарти Дискретна – відеокарта nVidia RTX 3050 – об'єм відеопам'яті 4 ГБ	1	4260,00	4728,60
Концентратор Speedlink SNAPPY SLIM USB Hub, 4–Port, USB 2.0, Passive, Black	1	250,00	277,50
Кабель для передачі даних SATA III 1.0m Cablexpert (CC–SATAM–DATA–XL)	1	50,00	55,50
Всього			5061,60

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення.

Балансову вартість спецустаткування розраховуємо за формулою:

$$B_{спец} = \sum_{i=1}^{k} Ц_i \cdot C_{пр.i} \cdot K_i \quad , \qquad (5.11)$$

де $Ц_i$ – ціна придбання одиниці спецустаткування даного виду, марки, грн;

$C_{пр.i}$ –кількість одиниць устаткування відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує доставку, монтаж, налагодження устаткування тощо, ($K_i = 1,10…1,12$);

k – кількість найменувань устаткування.

$B_{спец} = 3329,00 \cdot 1 \cdot 1,11 = 3695,19$ *грн.*

Отримані результати зведемо до таблиці:

Таблиця 5.9 – Витрати на придбання спецустаткування по кожному виду

Найменування устаткування	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Зовнішній жорсткий диск 2.5" 2TB Game Drive for Xbox Seagate (STKX2000400)	1	3329,00	3695,19
Всього			3695,19

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \quad , \qquad (5.12)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

Кі – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, (Кі = 1,10…1,12);

k – кількість найменувань програмних засобів.

Впрг = 125,00 · 1 · 1,11 = 138,75 грн.

Отримані результати зведемо до таблиці:

Таблиця 5.10 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Середовище розробки: – Android Studio	1	125,00	138,75

Продовження таблиці 5.10

– PyCharm	1	143,00	158,73
Всього			297,48

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{\text{обл}} = \frac{Ц_б}{T_в} \cdot \frac{t_{\text{вик}}}{12}, \quad (5.13)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

tвик – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

Тв – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

Аобл = (40111,00 · 2) / (2 · 12) = 3342,58 грн.

Проведені розрахунки зведемо до таблиці.

Таблиця 5.11 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Персональний комп'ютер проведення розробки Vinga Wolverine A5003 (I5M16G3060T.A5003)	40111,00	2	2	3342,58

Продовження таблиці 5.11

Робоче місце інженера–розробника ПЗ	8560,00	5	2	285,33
Пристрої передачі даних комутатор мережевий TP–Link LS1005G	520,00	2	2	43,33
Пристрій виводу інформації	6840,00	5	2	228,00
Оргтехніка	7250,00	4	2	302,08
Приміщення лабораторії	620400,00	20	2	5170,00
ОС Windows 10	8370,00	2	2	697,50
Прикладний пакет Microsoft Office 2016	7825,00	2	2	652,08
Мобільний телефон Samsung A12	9999,00	3	2	555,50
Всього				11276,42

Витрати на силову електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{\text{уі}} \cdot t_i \cdot Ц_e \cdot K_{\text{впі}}}{\eta_i}, \qquad (5.14)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

ti – тривалість роботи обладнання на етапі дослідження, год;

Це – вартість 1 кВт–години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо Це = 6,20 грн;

Квпі – коефіцієнт, що враховує використання потужності, Квпі <1;

і – коефіцієнт корисної дії обладнання, і<1.

$B_e = 0,32 \cdot 220,0 \cdot 6,20 \cdot 0,95 / 0,97 = 436,48$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 5.12 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Персональний комп'ютер проведення розробки Vinga Wolverine A5003 (I5M16G3060T.A5003)	0,32	220,0	436,48
Робоче місце інженера–розробника ПЗ	0,15	220,0	204,60
Пристрої передачі даних комутатор мережевий TP–Link LS1005G	0,01	440,0	27,28
Пристрій виводу інформації	0,50	25,0	77,50
Оргтехніка	0,62	4,5	17,30
Ноутбук Asus Zenbook 14 Процесор: – AMD Ryzen 7 5800H (4.2 ГГц) – кількість ядер: 8 ядер Оперативна пам'ять: – оперативна пам'ять 16 ГБ – тип пам'яті DDR4	0,05	200,0	62,00
Всього			825,16

До статті «Службові відрядження» дослідної роботи на тему «Метод та засіб автентифікації користувачів на основі токенів» належать витрати на відрядження штатних працівників, працівників організацій, які працюють за договорами цивільно–правового характеру, аспірантів, зайнятих розробленням

досліджень, відрядження, пов'язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з'їзди, конференції, наради, пов'язані з виконанням конкретних досліджень.

Витрати за статтею «Службові відрядження» розраховуємо як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{св} = (З_о + З_р) \cdot \frac{H_{св}}{100\%}, \quad (5.15)$$

де $H_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св}$ = 20%.

$B_{св} = (62052,73 + 3559,90) \cdot 20 / 100\% = 13122,53$ *грн.*

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» розраховуємо як 30…45% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{сп} = (З_о + З_р) \cdot \frac{H_{сп}}{100\%}, \quad (5.16)$$

де $Н_{сп}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 30%.

$B_{сп} = (62052,73 + 3559,90) \cdot 30 / 100\% = 19683,79$ *грн.*

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{H_{ів}}{100\%}, \quad (5.17)$$

де $H_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 50%.

$I_в = (62052,73 + 3559,90) \cdot 50 / 100\% = 32806,31$ *грн.*

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та

раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково–технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (5.18)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 115%.

$В_{нзв}$ = (62052,73 + 3559,90) · 115 / 100% = 75454,52 грн.

Витрати на проведення науково–дослідної роботи на тему «Метод та засіб автентифікації користувачів на основі токенів» розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}. \quad (5.19)$$

$В_{заг}$= 62052,73 +3559,90 +7217,39 +16022,60 +3857,25 +5061,60 +3695,19 + 297,48 + 11276,42 +825,16 +13122,53 +19683,79 +32806,31 +75454,52 = 254932,86 грн.

Загальні витрати *ЗВ* на завершення науково–дослідної (науково–технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (5.20)$$

де η – коефіцієнт, який характеризує етап (стадію) виконання науково–дослідної роботи, приймемо η=0,9.

ЗВ = 254932,86 / 0,9 = 283258,73 грн.

5.4 Розрахунок економічної ефективності науково–технічної розробки при її можливій комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково–технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження проведені за темою «Метод та засіб автентифікації користувачів на основі токенів» передбачають комерціалізацію протягом 4–х років реалізації на ринку.

В цьому випадку основу майбутнього економічного ефекту будуть формувати:

ΔN – збільшення кількості споживачів яким надається відповідна інформаційна послуга у періоди часу, що аналізуються;

Показник	1–й рік	2–й рік	3–й рік	4–й рік
Збільшення кількості споживачів, осіб	1200	3600	4300	4000

N – кількість споживачів яким надавалась відповідна інформаційна послуга у році до впровадження результатів нової науково–технічної розробки, приймемо 8000 осіб;

$Ц_б$ – вартість послуги у році до впровадження інформаційної системи, приймемо 4680,00 грн;

$\pm\Delta Ц_o$ – зміна вартості послуги від впровадження результатів, приймемо 174,34 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із 4–х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково–технічної розробки, розраховуємо за формулою [21]:

$$\Delta П_i = (\pm \Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \quad (5.21)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2022 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0,8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту). Приймемо ρ =40%;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2022 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1–го року:

$\Delta П_1 =$ (174,34·8000,00+4854,34·1200)·0,83·0,4·(1–0,18/100%)=1965553,20 грн.

Збільшення чистого прибутку 2–го року:

$\Delta П_2 =$ (174,34·8000,00+4854,34·4800)·0,83·0,4·(1–0,18/100%)=6723117,08 грн.

Збільшення чистого прибутку 3–го року:

$\Delta П_3 =$ (174,34·8000,00+4854,34·9100)·0,83·0,4·(1–0,18/100%)= =12405762,82 грн.

Збільшення чистого прибутку 4–го року:

$\Delta П_4 =$ (174,34·8000+4854,34·13100)·0,83·0,4·(1–0,18/100%)=17691944,91 грн.

Приведена вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково–технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (5.22)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково–технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково–технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ=0,24;

t – період часу (в роках) від моменту початку впровадження науково–технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$ПП$ = 1965553,20/(1+0,24)[1]+6723117,08/(1+0,24)[2]+12405762,82/(1+0,24)[3]+ +17691944,91/(1+0,24)[4]=1585123,55+4372474,69+6506664,56+7483225,59=19947 488,39 грн.

Величина початкових інвестицій PV , які потенційний інвестор має вкласти для впровадження і комерціалізації науково–технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \tag{5.23}$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково–технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=2,4;

$ЗВ$ – загальні витрати на проведення науково–технічної розробки та оформлення її результатів, приймаємо 283258,73 грн.

$PV = k_{інв} \cdot ЗВ$ = 2,4 · 283258,73 = 679820,95 грн.

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково–технічної розробки становитиме:

$$E_{абс} = ПП - PV \tag{5.24}$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково–технічної розробки, 19947488,39 грн;

PV – теперішня вартість початкових інвестицій, 679820,95 грн.

$E_{абс} = ПП - PV$ = 19947488,39 – 679820,95 = 19267667,43 грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково–технічної розробки:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (5.25)$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, 19267667,43 грн;

PV – теперішня вартість початкових інвестицій, 679820,95 грн;

$T_ж$ – життєвий цикл науково–технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 4 роки.

$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1$ = (1+19267667,43/679820,95)$^{1/4}$= 1,33.

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \quad (5.26)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2022 році в Україні d =0,1;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,26.

$\tau_{мін}$ = 0,1+0,26 = 0,36 < 1,33 свідчить про те, що внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково–технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково–дослідну роботу за темою «Метод та засіб автентифікації користувачів на основі токенів» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково–технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \tag{5.27}$$

де $E_в$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок}$ = 1 / 1,33 = 0,75 р.

$T_{ок} <$ 3–х років, що свідчить про комерційну привабливість науково–технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

5.5 Висновки з розділу

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Метод та засіб автентифікації користувачів на основі токенів» становить 37,7 бала, що, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

При оцінюванні за технічними параметрами, згідно узагальненого коефіцієнту якості розробки, науково–технічна розробка переважає існуючі аналоги приблизно в 1,40 рази.

Також термін окупності становить 0,75 р., що менше 3–х років, що свідчить про комерційну привабливість науково–технічної розробки і може

спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Отже можна зробити висновок про доцільність проведення науково–дослідної роботи за темою «Метод та засіб автентифікації користувачів на основі токенів».

ВИСНОВКИ

Аналіз задач кібербезпеки при перевірці автентичності користувача перед отриманням доступу до роботи з інформаційними ресурсами показав необхідність покращення засобів автентифікації шляхом надання можливості користувачеві керувати параметрами факторів автентифікації. Завдяки подальшому аналізу факторів автентифікації обґрунтовано використання токенів як таких факторів автентифікації.

Під час дослідження методів автентифікації виконано математичний опис процесу автентифікації користувачів. Для формалізованого математичного опису системи використано теоретико–множинний підхід. В даній магістерській кваліфікаційній роботі для підвищення рівня захищеності запропоновано метод автентифікації з використанням токена, що використовує мобільний пристрій із мобільним застосунком у процесі автентифікації користувача. Розроблено структуру та метод формування токену, а також правила його формування, що дозволяють надати контроль щодо створення токенів користувачеві.

На основі проведених досліджень сформовано загальну архітектуру засобу автентифікації користувачів, який представлено у вигляді клієнт–серверного модуля. Розроблено узагальнений алгоритм модуля автентифікації, а також алгоритми використання токенів користувачів та алгоритми процесу автентифікації користувача для серверної та клієнтської сторін.

Метод автентифікації реалізовано мовами програмування Dart (мобільний застосунок) та Python (сервер). Для тестування та дослідження коректності роботи методу автентифікації розроблено мобільний застосунок, що використовує модуль автентифікації для забезпечення безпеки при автентифікації користувачів на основі токенів. Тестування розроблених засобів показало коректність роботи окремих складових розробки та коректність роботи загалом під час інтеграції у клієнт–серверний застосунок при різних

ситуаціях та будь–яких вхідних даних. Це дозволило обґрунтувати коректність прийнятих технічних рішень.

Шляхом обчислення показників економічної доцільності було визначено собівартість, очікувана рентабельність, що дозволило обґрунтувати перспективність реалізації засобу та його комерційну привабливість.

Розроблені метод та засіб можуть бути використані як для однофакторної, так і для багатофакторної автентифікації. При цьому можливість конфігурування токенів та розроблений мобільний застосунок дозволять підвищити захищеність багатофакторної автентифікації без суттєвого зростання складності користування для користувача.

Перспективи подальшого розвитку даного дослідження полягають у гранулюванні прав доступу користувача та можливості розмежування прав у багатокористувацьких системах таких, як корпоративні мережі, що дозволить підвищити їх загальний стан кібербезпеки.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Лоборчук А. М. Телеграм–бот для моніторингу подій смарт–контракту: матеріали L науково–технічної конференції факультету інформаційних технологій та комп'ютерної інженерії, м. Вінниця, 2021 р. URL: https://conferences.vntu.edu.ua/index.php/all–fitki/all–fitki–2021/paper/view/11994/10004 (дата звернення: 20.10.2022)

2. Лоборчук А. М. ІНТЕРНЕТ ПІРАТСТВО: матеріали LI Науково–технічна конференція факультету інформаційних технологій та комп'ютерної інженерії, м. Вінниця, 2022 р. URL: https://d.conf.vntu.edu.ua/index.php/all–fitki/all–fitki–2022/paper/view/15676/13490 (дата звернення: 20.10.2022)

3. Мартинова, Л. Є., Умніцин М. Ю., Назарова К. Є. Дослідження та порівняльний аналіз методів аутентифікації. Молодий вчений. 2016. С. 90-93.

4. Шибанов, С. В., Карпушин Д. А. Порівняльний аналіз сучасних методів аутентифікації користувача. Математичне та програмне забезпечення систем у промисловій та соціальній сферах. 2015р. №1. С. 33-37.

5. Nilesh A. Lal, Salendra Prasad, Mohammed Farik, A Review Of Authentication Methods. international journal of scientific & technology research volume 5, issue 11, 2016 p. 246 – 249.

6. Syed Zulkarnain Syed Idrus, Estelle Cherrier, Christophe Rosenberger, Jean–Jacques Schwartzmann. A Review on Authentication Methods. Australian Journal of Basic and Applied Sciences, 2013, 7 (5), p. 95–107.

7. Mohammad A. Alia, Abdelfatah Aref Tamimi, and Omaima N. A. AL–Allaf, Cryptography Based Authentication Methods. Proceedings of the World Congress on Engineering and Computer Science 2014 Vol I, 2014, San Francisco, p. 199 – 204.

8. Guy Levin. 4 Most Used REST API Authentication Methods, July 26 2019. URL: https://blog.restcase.com/4–most–used–rest–api–authentication–methods/ (accessed: 09.09.2022).

9. Hypertext Transfer Protocol (HTTP) Authentication Scheme Registry, 2012 (upd. 2022). URL: https://www.iana.org/assignments/http–authschemes/http–authschemes.xhtml (accessed: 13.09.2022).

10. RFC 6750:2012. The OAuth 2.0 Authorization Framework: Bearer Token Usage, RFC 6750, p. 18, October 2012. URL: https://www.rfc–editor.org/rfc/rfc6750.

11. OAuth 2.0. URL: https://oauth.net/2/ (accessed: 15.09.2022).

12. OpenID Connect. URL: https://openid.net/connect/ (acessed: 16.09.2022).

13. Leonard Richardson, Mike Amundsen. RESTful Web APIs: Services for a Changing World, Newton. O'Reilly Media, 2013, p. 406.

14. Token–based authorization of Connection Oriented Network resources, 2004. URL: https://www.researchgate.net/profile/Leon–Gommans/publication/252434943_Token–based_authorization_of_Connection_Oriented_Network_resources/links/552e156a0cf29b22c9c5187c/Token–based–authorization–of–Connection–Oriented–Network–resources.pdf (accessed: 19.09.2022).

15. Authentication Token. URL: https://www.fortinet.com/resources/cyberglossary/authentication–token (accessed: 23.09.2022).

16. RFC 7519:2015, "JSON Web Token (JWT)", May 2015, p. 30. URL: https://www.rfc–editor.org/rfc/rfc7519 (accessed: 26.09.2022).

17. JSON Web Tokens. URL: https://jwt.io/ (accessed: 26.09.2022).

18. Holly Guevara. How SAML Authentication Works, 2021. URL: https://auth0.com/blog/how–saml–authentication–works/ (accessed: 28.09.2022).

19. Timur Guvenkaya. JSON Web Token attacks and vulnerabilities, July 2 2021. URL: https://www.invicti.com/blog/web–security/json–web–token–jwt–attacks–vulnerabilities/ (accessed: 30.09.2022).

20. Testing Flutter apps. URL: https://docs.flutter.dev/testing (accessed: 15.11.2022).

21. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. Вінниця, ВНТУ, 2021. 42 с.

22. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа. Вінниця, ВНТУ, 2016. 113 с.

ДОДАТКИ

Додаток А

Код мобільного застосунку

Main.dart:import 'dart:developer';
import 'dart:io';

import 'package:flutter/material.dart';
import 'package:authapp/user_info.dart';
import 'package:authapp/QR.dart';
import
'package:shared_preferences/shared_preferences.dart';
import 'package:fast_rsa/fast_rsa.dart';
import 'package:authapp/token_page.dart';

void main() {
 runApp(const MyApp());
}

class MyApp extends StatelessWidget {
 const MyApp({super.key});

 // This widget is the root of your application.
 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   title: 'Flutter Demo',
   theme: ThemeData(
    // This is the theme of your application.
    //
    // Try running your application with "flutter run". You'll see the
    // application has a blue toolbar. Then, without quitting the app, try
    // changing the primarySwatch below to Colors.green and then invoke
    // "hot reload" (press "r" in the console where you ran "flutter run",
    // or simply save your changes to "hot reload" in a Flutter IDE).
    // Notice that the counter didn't reset back to zero; the application
    // is not restarted.
    primarySwatch: Colors.blue,
   ),
   home: MyHomePage(title: 'AuthApp'),
  );
 }
}

class MyHomePage extends StatefulWidget {
 const MyHomePage({super.key, required this.title});

 // This widget is the home page of your application. It is stateful, meaning
 // that it has a State object (defined below) that contains fields that affect
 // how it looks.

 // This class is the configuration for the state. It holds the values (in this
 // case the title) provided by the parent (in this case the App widget) and
 // used by the build method of the State. Fields in a Widget subclass are
 // always marked "final".

 final String title;

 @override
 State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
 final Future<SharedPreferences> _prefs = SharedPreferences.getInstance();
 List<String> Sites = [];
 var Widgets = <Widget>[];

```
  @override
  void initState() {
    super.initState();
    update();
    _prefs.then((SharedPreferences prefs) {
      if (prefs.getString("Private") == null) {
        // RSA.generate(512).then((keys) {
        //   prefs.setString("Private", keys.privateKey);
        //   prefs.setString("Public", keys.publicKey);
        // });
      }
      if (prefs.getString("Email") == null) {
        Navigator.push(context, MaterialPageRoute(builder:
(context) => const UserInfoPage()));
      }
    });
  }

  void update() {
    _prefs.then((SharedPreferences prefs) {
      if (prefs.getStringList('Sites') != null) {
        Sites = prefs.getStringList('Sites')!;
      } else {
        Sites = [];
      }
      setState(() {
        for (var name in Sites) {
          Widgets.add(Padding(
            padding: const EdgeInsets.all(8.0),
            child: ElevatedButton(
              onPressed: () {
                Navigator.push(context,
MaterialPageRoute(builder: (context) => const
TokenPage()));
              },
              child: Text(name),
            ),
          ));
        }
      });
    });
  }

  @override
  Widget build(BuildContext context) {
    // This method is rerun every time setState is called,
for instance as done
    // by the _incrementCounter method above.
    //
    // The Flutter framework has been optimized to make
rerunning build methods
    // fast, so that you can just rebuild anything that needs
updating rather
    // than having to individually change instances of
widgets.
    return Scaffold(
      appBar: AppBar(
        // Here we take the value from the MyHomePage
object that was created by
        // the App.build method, and use it to set our appbar
title.
        title: Text(widget.title),
        actions: [
          ElevatedButton(
            onPressed: update,
            child: Text("Update"),
          )
        ],
      ),
      body: Center(
        // Center is a layout widget. It takes a single child
and positions it
        // in the middle of the parent.
        child: Column(
          // Column is also a layout widget. It takes a list of
children and
          // arranges them vertically. By default, it sizes
itself to fit its
          // children horizontally, and tries to be as tall as its
parent.
          //
          // Invoke "debug painting" (press "p" in the
console, choose the
```

```
        // "Toggle Debug Paint" action from the Flutter Inspector in Android
        // Studio, or the "Toggle Debug Paint" command in Visual Studio Code)
        // to see the wireframe for each widget.
        //
        // Column has various properties to control how it sizes itself and
        // how it positions its children. Here we use mainAxisAlignment to
        // center the children vertically; the main axis here is the vertical
        // axis because Columns are vertical (the cross axis would be
        // horizontal).
        mainAxisAlignment: MainAxisAlignment.start,
        children: Widgets,
      ),
    ),
    floatingActionButton:
    Column(mainAxisAlignment:
MainAxisAlignment.end, children: <Widget>[
      FloatingActionButton(
        onPressed: () {
          Navigator.of(context).push(MaterialPageRoute(
            builder: (context) => const UserInfoPage(),
          ));
        },
        child: const Icon(Icons.settings),
        heroTag: "Settings",
      ),
      FloatingActionButton(
        onPressed: () {
          Navigator.of(context).push(MaterialPageRoute(
            builder: (context) => const QRViewExample(),
          ));
        },
        child: const Icon(Icons.add),
        heroTag: "Add",
      ),
    ]), // This trailing comma makes auto-formatting nicer for build methods.
  );
 }
}
```

QR.dart:

```
import 'package:qr_code_scanner/qr_code_scanner.dart';
import 'package:flutter/foundation.dart';
import 'package:flutter/material.dart';
import 'dart:io';
import 'dart:convert';
import
'package:shared_preferences/shared_preferences.dart';
import 'package:http/http.dart' as http;


class QRViewExample extends StatefulWidget {
 const QRViewExample({Key? key}) : super(key: key);

 @override
 State<StatefulWidget> createState() =>
_QRViewExampleState();
}

class _QRViewExampleState extends
State<QRViewExample> {
 Barcode? result;
 QRViewController? controller;
 final GlobalKey qrKey = GlobalKey(debugLabel: 'QR');
 final Future<SharedPreferences> _prefs =
SharedPreferences.getInstance();
 // In order to get hot reload to work we need to pause the camera if the platform
 // is android, or resume the camera if the platform is iOS.
 @override
 void reassemble() {
  super.reassemble();
  if (Platform.isAndroid) {
   controller!.pauseCamera();
  }
  controller!.resumeCamera();
 }
```

```
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   body: Column(
    children: <Widget>[
     Expanded(flex: 4, child: _buildQrView(context)),
     Expanded(
      flex: 1,
      child: FittedBox(
       fit: BoxFit.contain,
       child: Column(
        mainAxisAlignment:
MainAxisAlignment.spaceEvenly,
        children: <Widget>[
         if (result != null)
          Text(
           'Barcode                    Type:
${describeEnum(result!.format)}   Data: ${result!.code}')
         else
          const Text('Scan a code'),
         Row(
          mainAxisAlignment:
MainAxisAlignment.center,
          crossAxisAlignment:
CrossAxisAlignment.center,
          children: <Widget>[
           Container(
            margin: const EdgeInsets.all(8),
            child: ElevatedButton(
             onPressed: () async {
              await controller?.toggleFlash();
              setState(() {});
             },
             child: FutureBuilder(
              future: controller?.getFlashStatus(),
              builder: (context, snapshot) {
               return          Text('Flash:
${snapshot.data}');
              },
             )),
           ),
           Container(
            margin: const EdgeInsets.all(8),
            child: ElevatedButton(
             onPressed: () async {
              await controller?.flipCamera();
              setState(() {});
             },
             child: FutureBuilder(
              future: controller?.getCameraInfo(),
              builder: (context, snapshot) {
               if (snapshot.data != null) {
                return Text(
                 'Camera          facing
${describeEnum(snapshot.data!)}');
               } else {
                return const Text('loading');
               }
              },
             )),
           )
          ],
         ),
        ],
       ),
      ),
     ],
    ),
   );
  }

  Widget _buildQrView(BuildContext context) {
   // For this example we check how width or tall the
device is and change the scanArea and overlay
accordingly.
   var scanArea = (MediaQuery.of(context).size.width <
400 ||
     MediaQuery.of(context).size.height < 400)
     ? 250.0
     : 300.0;
   // To ensure the Scanner view is properly sizes after
rotation
```

```
  // we need to listen for Flutter SizeChanged notification and update controller
  return QRView(
   key: qrKey,
   onQRViewCreated: _onQRViewCreated,
   overlay: QrScannerOverlayShape(
     borderColor: Colors.red,
     borderRadius: 10,
     borderLength: 30,
     borderWidth: 10,
     cutOutSize: scanArea),
   onPermissionSet: (ctrl, p) => _onPermissionSet(context, ctrl, p),
  );
 }

 void _onQRViewCreated(QRViewController controller) {
  setState(() {
   this.controller = controller;
  });
  controller.resumeCamera();
  controller.scannedDataStream.listen((scanData) {
   setState(() {
    result = scanData;
    if (result != null) {
     Map<String, dynamic> Data = jsonDecode(result!.code!);
     _prefs.then((SharedPreferences prefs) async {
      var SiteList = prefs.getStringList('Sites');
      if ( SiteList != null) {
       SiteList.add(Data["Name"]);
       prefs.setStringList('Sites', SiteList);
       prefs.setString(Data["Name"], result!.code!);
      } else {
       List<String> l_list = [Data["Name"]];
       prefs.setStringList('Sites', l_list);
       prefs.setString(Data["Name"], result!.code!);
      }
      await http.post(Data["key"], body: {'public': prefs.getString("Public")});
      controller!.pauseCamera();
      Navigator.pop(context);
     });
    }
   });
  });
 }

 void _onPermissionSet(BuildContext context, QRViewController ctrl, bool p) {
  if (!p) {
   ScaffoldMessenger.of(context).showSnackBar(
    const SnackBar(content: Text('no Permission')),
   );
  }
 }

 @override
 void dispose() {
  controller?.dispose();
  super.dispose();
 }
}
```

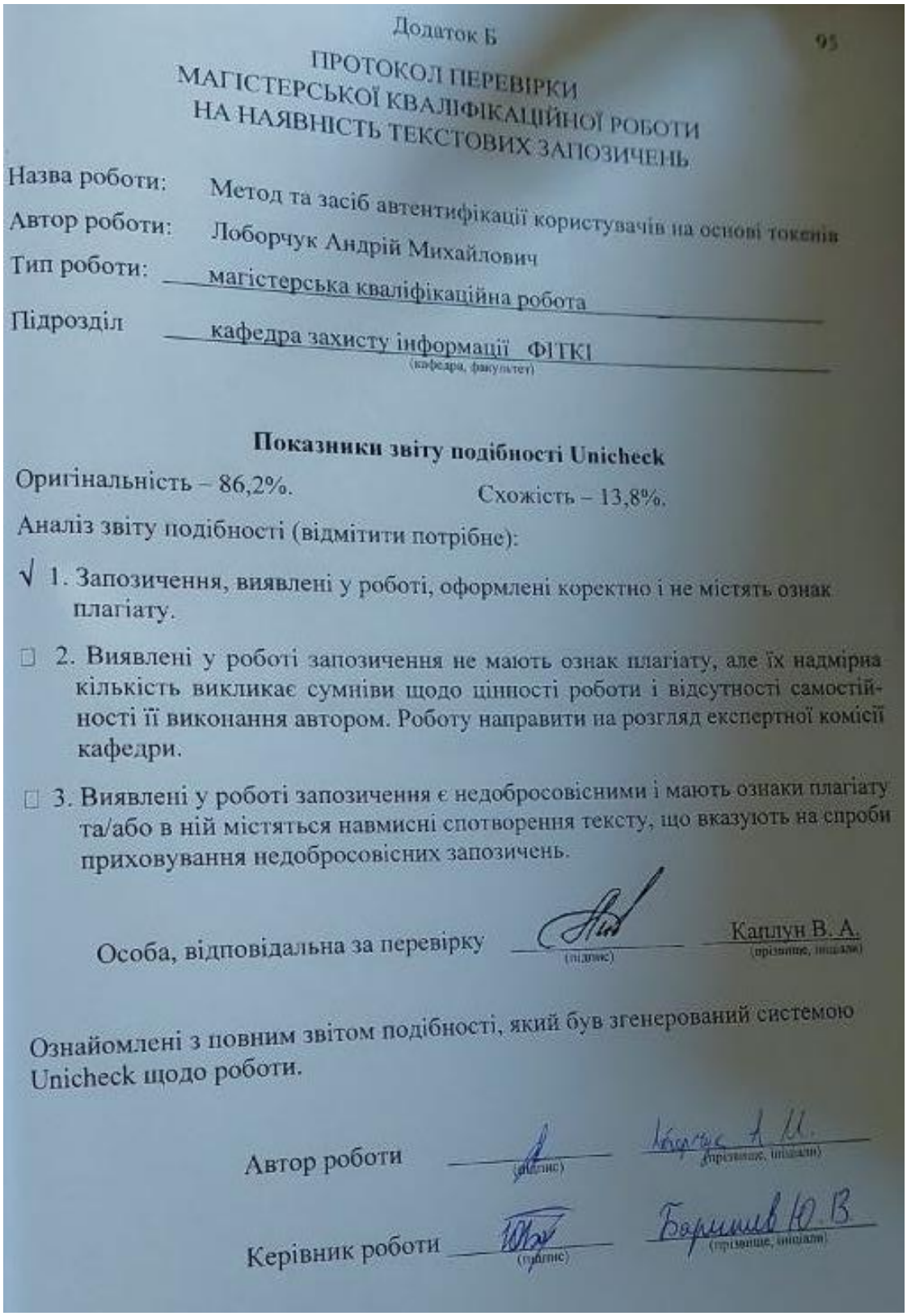
User_info.dart:

```
import 'dart:developer';
import 'dart:io';

import 'package:flutter/foundation.dart';
import 'package:flutter/material.dart';
import 'package:shared_preferences/shared_preferences.dart';


class UserInfoPage extends StatefulWidget {
 const UserInfoPage({Key? key}) : super(key: key);


 @override
 State<UserInfoPage> createState() => _UserInfoPageState();
}
```

```
class _UserInfoPageState extends State<UserInfoPage>
{

 final Future<SharedPreferences> _prefs =
SharedPreferences.getInstance();

 final FirstName = TextEditingController();
 final LastName = TextEditingController();
 final Email = TextEditingController();



 Future<void> SetInfo() async {
  final SharedPreferences prefs = await _prefs;
  prefs.setString('FirstName', FirstName.text);
  prefs.setString('LastName', LastName.text);
  prefs.setString('Email', Email.text);
  Navigator.pop(context);
 }

 @override
 void initState() {
  super.initState();
    _prefs.then((SharedPreferences prefs) {
      FirstName.text = prefs.getString('FirstName') ?? '';
      LastName.text = prefs.getString('LastName') ?? '';
      Email.text = prefs.getString('Email') ?? '';
  });
 }

 @override
 Widget build(BuildContext context) {

  return Scaffold(
    appBar: AppBar(

      title: Text("Settings"),
    ),
    body: Center(

      child: Column(
mainAxisAlignment: MainAxisAlignment.start,
children: <Widget>[

 TextField(
  decoration: InputDecoration(
   border: OutlineInputBorder(),
   hintText: 'First name'
  ),
   controller: FirstName
 ),
 TextField(
  decoration: InputDecoration(
   border: OutlineInputBorder(),
   hintText: 'Last name'
  ),
   controller: LastName
 ),
 TextField(
  decoration: InputDecoration(
   border: OutlineInputBorder(),
   hintText: 'Email'
  ),
   controller: Email
 ),
 Padding(
  padding: const EdgeInsets.all(8.0),
  child: ElevatedButton(
   onPressed: SetInfo,
   child: Text('Apply'),
  )
 )
],
),
),
);
}
}
```

Додаток Б

ПРОТОКОЛ ПЕРЕВІРКИ
МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ



Додаток Б 95

ПРОТОКОЛ ПЕРЕВІРКИ
МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Метод та засіб автентифікації користувачів на основі токенів
Автор роботи: Лоборчук Андрій Михайлович
Тип роботи: магістерська кваліфікаційна робота
Підрозділ кафедра захисту інформації ФІТКІ
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність – 86,2%. Схожість – 13,8%.

Аналіз звіту подібності (відмітити потрібне):

√ 1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

☐ 2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

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

Особа, відповідальна за перевірку (підпис) Каплун В. А. (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи (підпис) Лоборчук А. А. (прізвище, ініціали)

Керівник роботи (підпис) Бармин Ю. В. (прізвище, ініціали)

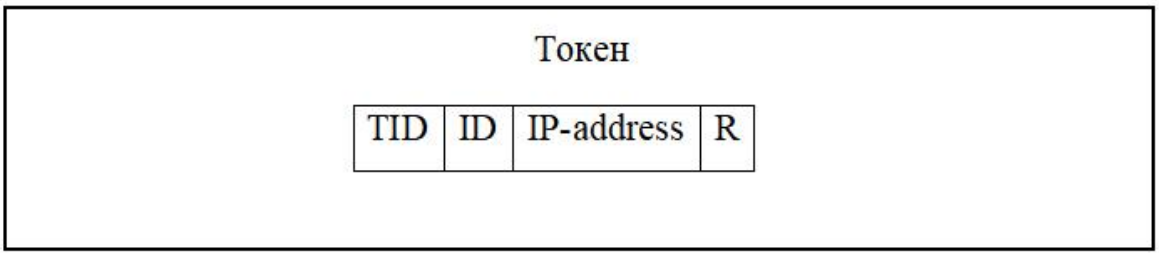
ІЛЮСТРАТИВНА ЧАСТИНА

МЕТОД ТА ЗАСІБ АВТЕНТИФІКАЦІЇ КОРИСТУВАЧІВ НА ОСНОВІ ТОКЕНІВ

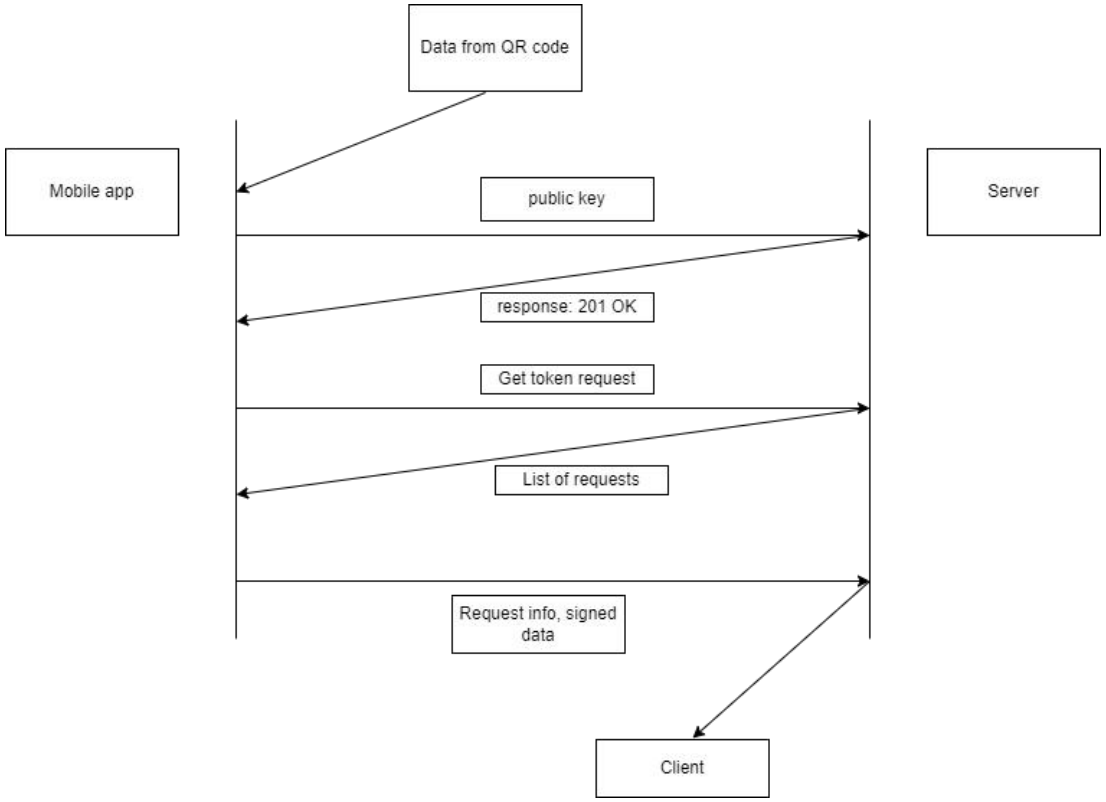
ПОРІВНЯЛЬНИЙ АНАЛІЗ МЕТОДІВ АВТЕНТИФІКАЦІЇ

Параметр	парольна автентифікація	апаратна автентифікація	біометрична автентифікація	автентифікація за цифровими сертифікатами
Складність/ціна	1	4	3	2
Захищеність	2	4	3	3

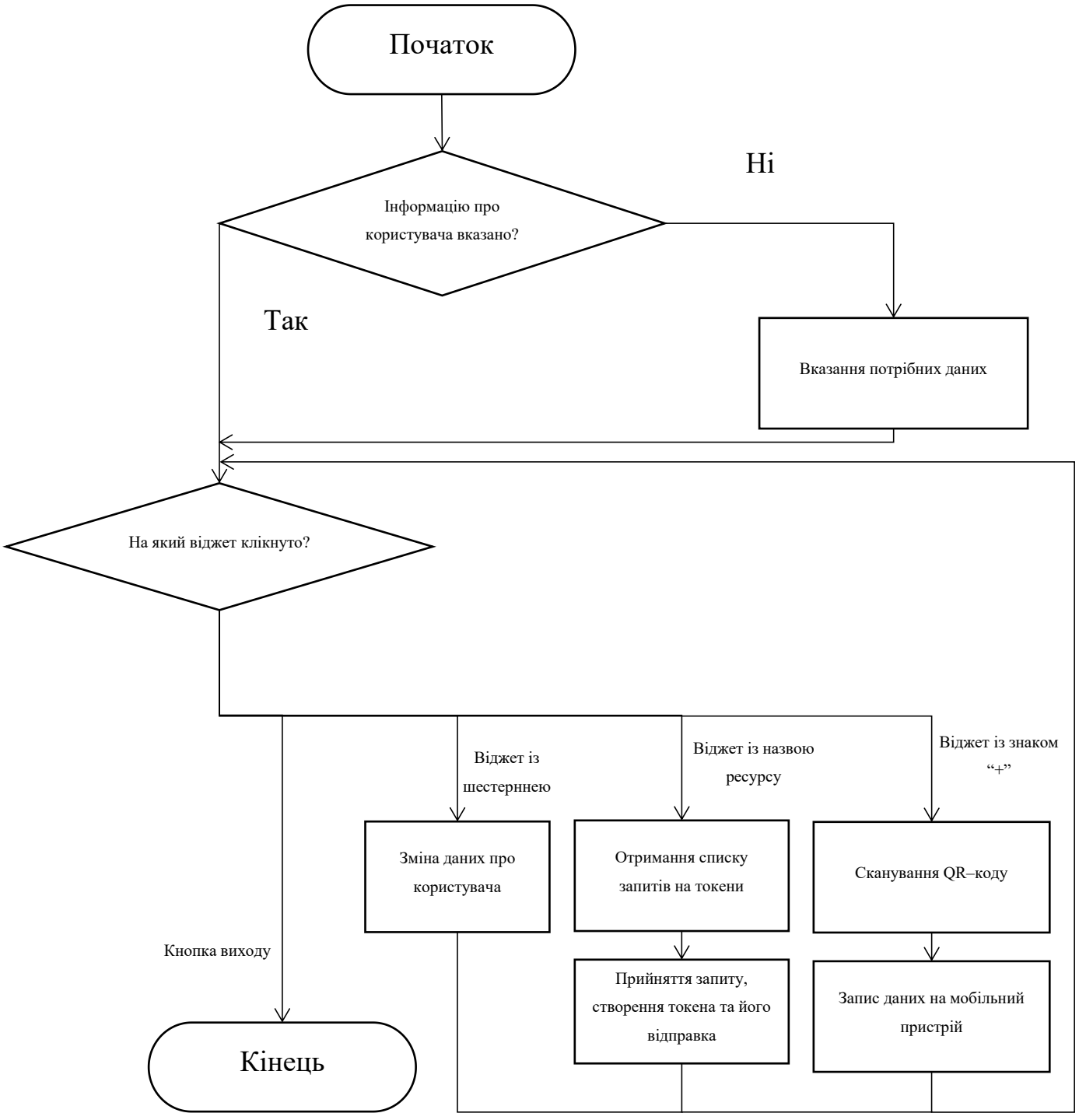
СТРУКТУРА ТОКЕНА



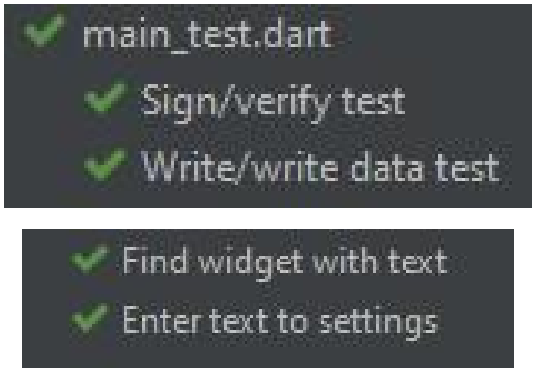
ПРОТОКОЛ ОБМІНУ ТОКЕНАМИ



АЛГОРИТМ РОБОТИ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ



РЕЗУЛЬТАТ БЛОКОВОГО ТЕСТУВАННЯ



РЕЗУЛЬТАТИ ІНТЕГРАЦІЙНОГО ТЕСТУВАННЯ