Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інтелектуальних інформаційних технологій та автоматизації
(повне найменування інституту. назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна технологія безконтактного обміну персональними даними»

Виконав: студент 2-го курсу, групи 1КН-21м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
_____ Каплунський О.А.
(прізвище та ініціали)

Керівник: завідувач кафедри КН, проф., д.т.н.
_____ Яровий А.А.
(прізвище та ініціали)
«15» 12 2022 р.

Опонент: д.т.н., професор каф. АІІТ
_____ Квєтний Р.Н.
(прізвище та ініціали)
«15» 12 2022 р.

Допущено до захисту
Завідувач кафедри КН
_____ д.т.н., проф. Яровий А.А.
(прізвище та ініціали)
«16» 12 2022 р.

Вінниця ВНТУ - 2022 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 122 «Комп'ютерні науки»
Освітньо-професійна програма – «Системи штучного інтелекту»

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
Д.т.н., проф. Яровий А.А.
14. 09. **2022 року**

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Каплунський Олександр Андрійович
(прізвище, ім'я, по батькові)

1. Тема роботи Інформаційна технологія безконтактного обміну персональними даними

керівник роботи завідувач кафедри КН, професор., д.т.н. Яровий А. А.
затверджені наказом вищого навчального закладу від "14" 09 2022 року №203
2. Строк подання студентом роботи 18 листопада 202_ року

3. Вихідні дані до роботи:
Вхідні дані – NFC-тег типу NTAG213 з обсягом пам'яті 180 байтів, додаток доступний на мобільних пристроях, тип інформації для запису на тег – текстова, мінімальний відсоток стиснення даних – 10, мова програмування – кросплатформена з можливістю написання мобільних додатків.
4. Зміст текстової частини:
Вступ, аналіз предметної області безконтактної передачі персональної інформації, розробка інформаційної технології безконтактного обміну персональної інформації, програмна реалізація інформаційної технології безконтактного обміну персональної інформації, економічна частина, висновки, перелік використаних джерел, додатки
5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень)
Схема взаємодії двох пристроїв при передачі даних за допомогою NFC, Схема режимів роботи NFC, Фрагмент схеми алгоритму пошуку режиму зв'язку NFC модуля, Характеристики стандартів NFC, Візуалізація ітерацій роботи алгоритму LZW, Приклад роботи дельта-кодування в алгоритмі LZMA, Загальна структурна схема інформаційної технології безконтактного обміну

персональними даними, Діаграма послідовності взаємодії мо
інформаційної технології безконтактного обміну персональними да
Схема системи клієнт-сервер, Схема алгоритму роботи інформа
технології безконтактного обміну персональними даними, Загальний в
роботи програми, Загальний вигляд стисненого посилання на пр
користувача, Діаграма залежності стиснення від об'єму інформації.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	виконан прийня
1-3	Яровий А.А., д.т.н., проф. завідувач каф. КН	14.09.2022	19.12.2
4	Нікіфорова Л. О., к. е. н., доц. каф. ЕПВМ	19.09.2022	12.12.202

7. Дата видачі завдання 14.09. 2022 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи
1	Аналіз сучасного стану розвитку технологій безконтактної передачі даних та алгоритмів стиснення даних	14.09.2022р – 31.10.2022р
2	Обґрунтування методу розв'язання задачі, обґрунтування вибору інструментів технічної реалізації	02.10.2022р – 16.10.2022р
3	Проектування інформаційної технології. Програмна реалізація інформаційної технології	17.10.2022р – 07.11.2022р
4	Підготовка економічної частини	08.11.2022р – 21.11.2022р
5	Апробація та/або впровадження результатів дослідження	23.11.2022р – 01.12.2022р
6	Оформлення пояснювальної записки, графічного матеріалу та презентації	02.12.2022р – 14.12.2022р

Студент _____ Каплунський О.А.

Керівник роботи _____ (підпис)

АНОТАЦІЯ

УДК 004.8

Каплунський О. А. Інформаційна технологія безконтактного обміну персональними даними. Магістерська кваліфікаційна робота зі спеціальності 122 – комп'ютерні науки, освітня програма - системи штучного інтелекту. Вінниця: ВНТУ, 2022. 136 с.

На укр. мові. Бібліогр.: 38 назв; рис.: 22; табл. 10.

Магістерська кваліфікаційна робота присвячена розробці інформаційної технології безконтактного обміну персональними даними. Технологія забезпечує програмні засоби, що реалізовані з метою підвищення ефективності передачі персональної інформації користувача, а саме збільшенню обсягу передачі, встановлення апаратного захисту даних та збільшенню швидкості зчитування інформації з пристрою.

У роботі досліджено предметну область безконтактної передачі інформації. Проаналізовано основні способи та засоби реалізації технології. Визначено основні складові модулі та побудовано оптимальну структурну схему технології для максимального забезпечення інформаційних потреб користувачів. Запропоновано інформаційну технологію безконтактної передачі персональних даних, що відрізняється від існуючих застосуванням алгоритму стиснення даних, дешевизною реалізації, гнучкістю системи. Розроблено мультикомпонентне програмне забезпечення для реалізації та проведення тестування. Програмні засоби реалізовано з використанням інтегрованого середовища розробки JetBrains WebStorm 2022, XCode, Android Studio, мови програмування JavaScript та фреймворків: express.js, React Native.

Ключові слова: алгоритм стиснення, NFC, безконтактна, передача даних.

ABSTRACT

Kaplunskyj O.A. Information technology for contactless personal data sharing. Master's thesis in the specialty 122 - computer sciences, educational program – artificial intelligence systems. Vinnytsia: VNTU, 2022. 136 p.

In Ukrainian language. Bibliographer: 38 titles; fig.: 22; table 10.

Master's thesis is devoted to the development of information technology of contactless exchange of personal data. The technology provides software tools implemented in order to increase the efficiency of the transfer of the user's personal information, namely, to increase the volume of transfer, establish hardware data protection and increase the speed of reading information from the device.

The subject area of non-contact information transmission is investigated in the work. The main methods and means of implementing the technology are analysed. The main component modules have been determined and an optimal structural diagram of the technology has been built to maximize the information needs of users. An information technology for contactless transfer of personal information is proposed, which differs from existing ones in the use of a data compression algorithm, low cost of implementation, and flexibility of the system. Multi-component software for implementation and testing has been developed. Software tools are implemented using the JetBrains WebStorm 2022 integrated development environment, XCode, Android Studio, JavaScript programming language and frameworks: express.js, React Native.

Keywords: compression algorithm, NFC, contactless, data transfer.

ЗМІСТ

ВСТУП

Актуальність досліджень. Починаючи з 2020 року, світ зазнав суттєвих змін. Значну роль у цьому зіграла пандемія вірусу. Це запровадило зміни у соціумі: мінімальні контакти з іншими людьми, соціальна дистанція та обережність. Буденні для людей речі зазнали змін або потрапили під обмеження.

Найбільшого удару зазнав бізнес, націлений на наданні послуг, які у основі містили контакт чи взаємодію між людьми. Проте нові реалії вимусили адаптувати принципи побудови роботи та процес взаємодії складових всередині бізнесу. Фінансові структури пропагують безконтактні системи оплати, державні заклади – онлайн запис на консультації, освітні заклади – дистанційне навчання, ресторанні бізнеси та кафе – безконтактні меню, та безготівкова оплата, транспорті компанії замінюють «живий» персонал на термінали або QR-коди.

Сучасна людина має смартфон який здатний розшифрувати штрих- або QR-коди та оснащений NFC-модулем, який, зазвичай, використовується для безготівкової оплати. Додаткові модулі дозволили впровадити нові можливості передачі інформації для користувачів. QR-коди найчастіше можна зустріти можна зустріти в закладах харчування, як альтернативу надрукованому меню, в транспорті та в цілому як джерело, яке містить посилання на ресурс або текстову інформацію, а закордоном їх використовують для створення анотацій або іншої допоміжної інформації. В Китаї більша частина інфраструктури зав’язана на використанні QR кодів. Мережа кодів в КНР дозволяє здійснювати платежі в магазинах, переводити кошти між людьми та використовується в інших аспектах життя жителів Китаю.

Зазвичай QR-код використовують як засіб зберігання посилання на веб-додаток або текстовий файл з інформацією. Також, QR-коди використовують для реклами та як персональні візитівки, універсальні ідентифікатори продуктів, працівників та будь-якої одиниці.

QR-код має низку недоліків, які негативно впливають на досвід роботи як користувачів, так і розробників сервісів, а також створює обмеження для розвитку продуктів, які базуються на даній технології. QR-код неможливо перезаписати: так як QR код містить в собі інформацію, яку зашифровано у вигляді візерунку, для зміни інформації записаної на коді, необхідно повторити процес генерації зображення з урахуванням внесених змін. Будь-хто може зчитати код та отримати інформацію зашифровану в ньому. Проте до уваги потрібно взяти проприєтарні коди, які базуються на QR-кодах, проте розшифрувати їх може лише пристрій з реалізованим алгоритмом дешифрування інформації. У випадку стандартних кодів, телефон має бути оснащеним якісною камерою, яка зможе розпізнати зображення в поганих умовах, зображення має бути якісним та великим, у випадку відсутності освітлення, камера не зможе розпізнати зображення. Пошкоджене зображення не буде правильно розшифровуватись, а тому дані можуть бути пошкоджені або втрачені. Також використання QR-коду для збереження динамічної інформації не є доцільним у випадку розміщення її у вигляді фізичної картинки. Звичайно, проектування QR-коду на екран пристрою, який буде оновлювати код щоразу, коли це необхідно, вирішує цю проблему. Також QR-коди мають обмежену кількість інформації для запису. У випадку додавання алгоритму стиснення даних, необхідно забезпечити метод дешифрації даних. NFC-теги позбавлені цих недоліків, а додаток для запису інформації на носій може виступати у ролі дешифратора інформації (цей процес може бути

вибірковим). Зважаючи на недоліки популярних методів передачі персональної інформації (малий об’єм пам’яті, метод взаємодії), розробка інформаційної технології безконтактного обміну персональними даними є актуальною.

Зв'язок роботи з науковими програмами, планами, темами.

Магістерська робота виконана відповідно до напрямку наукових досліджень кафедри комп’ютерних наук Вінницького національного технічного університету спеціальності 122 «Комп’ютерні науки» та плану наукової та навчально-методичної роботи кафедри.

Метою дослідження є розширення функціональних можливостей програмних засобів, що дозволить підвищити ефективність передачі персональних даних. На відміну від аналогів, які використовують QR-коди або пропрієтарні коди, в роботі пропонується застосувати NFC технології для підвищення швидкодії, безпеки та зручності використання, а також впровадити алгоритм стиснення даних для надання користувачу можливості зберігання більшого обсягу інформації на носії.

Основними задачами є:

1. Аналіз сучасного стану розвитку технологій безконтактної передачі персональної інформації;
2. Обґрунтування методу розв’язання задачі;
3. Проектування інформаційної технології безконтактної передачі персональних даних;
4. Програмна реалізація інформаційної технології безконтактної передачі персональних даних;
5. Тестування інформаційної технології безконтактної передачі персональних даних;
6. Оцінити та спрогнозувати економічний потенціал розробки.

Об’єктом дослідження є процес безконтактної передачі персональної інформації.

Предметом дослідження є програмні засоби для безконтактного передавання персональної інформації з можливістю стиснення інформації.

Методи дослідження. У процесі дослідження використовувались: дослідження сучасних методів передачі інформації, дослідження методів стиснення даних, дослідження методів масштабування та інтеграції програмного забезпечення, взаємодія з методами безконтактної передачі даних, методи та підходи до розробки веб-орієнтованих програмних додатків, методи об’єктно-орієнтованого програмування.

Наукова новизна одержаних результатів полягає в наступному:

Розроблено інформаційну технологію безконтактного обміну персональними даними, що відрізняється від існуючих застосуванням моделі поєднання методу стиснення даних LZW та NFC технології, що дозволило розширити функціональні можливості, а також забезпечити збільшення об’єму зберігання інформаційних даних на носії.

Практичне значення одержаних результатів. Практична цінність одержаних результатів полягає в тому, що на основі отриманих в магістерській кваліфікаційній роботі теоретичних положень реалізовано інформаційну технологію та розроблено програмні засоби для розширення функціональних можливостей безконтактної передачі персональної інформації, що досягається використанням більш вигідних для поставленої задачі технологій, з впровадженням додаткового функціоналу на програмному рівні.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується строгістю постановки задач, коректним застосуванням методів під час доведення наукових положень, порівнянням

результатів із відомими, та збіжністю результатів моделювання з результатами, що отримані під час впровадження розроблених програмних засобів.

Особистий внесок магістранта. Усі результати, наведені у магістерській кваліфікаційній роботі, отримані самостійно.

Апробація результатів роботи.

Основні результати даних досліджень пройшли апробацію на XII Міжнародній науково-практичній конференції «EURASIAN SCIENTIFIC DISCUSSIONS», 18-20.12.2022 Барселона, Іспанія.

Публікації.

За результатами магістерської кваліфікаційної роботи отримане авторське свідоцтво про реєстрацію авторського права на твір [1].

1 АНАЛІЗ СУЧАСНОГО СТАНУ РОЗВИТКУ ТЕХНОЛОГІЙ БЕЗКОНТАКТНОЇ ПЕРЕДАЧІ ДАНИХ ТА АЛГОРИТМІВ СТИСНЕННЯ ДАНИХ

1.1 Аналіз предметної області передачі даних

Технології бездротової передачі даних – підрозділ інформаційних технологій, які дозволяють передавати дані між двома та більшою кількістю точок на відстані без використання фізичного з’єднання. Для передачі даних використовуються радіохвилі, а раніше інфрачервоне, оптичне та лазерне випромінювання [2].

Інфрачервоні (ІЧ) бездротові технології – малопотужні, з малим радіусом дії. Передача даних здійснюється за допомогою світлодіодів, отримання − за допомогою фотодіодів. Інфрачервоні світлові сигнали працюють на низькій частоті світла і відстані передачі, що обмежена 1 м або менше. Інфрачервоне світло не може проникнути через стіни та інші перешкоди, що не пропускають світла [3].

Три найпоширеніших типи інфрачервоних мереж:

- мережі прямої видимості – передача сигналу здійснюється тільки в тому випадку, якщо пристрої знаходяться на прямій лінії видимості без перешкод;
- мережі на розсіяному випромінюванні – сигнал відбивається від стін і стелі;
- мережі на відбитому випромінюванні – сигнал передається на оптичний прийомо-передатчик, звідки направляється на приймач [3].

Оптичний бездротовий зв'язок – форма оптичного зв'язку, в якому видиме, інфрачервоне або ультрафіолетове світло використовується для передачі сигналу без використання дротових засобів зв'язку.

Системи бездротового оптичного зв'язку, що працюють у видимому спектрі світла зазвичай називають комунікацією видимим світлом. Такі системи використовують переваги світлодіодів, які можуть генерувати імпульси з дуже високою швидкістю без помітного впливу на вихідне світло і людське око. Системи бездротового оптичного зв’язку мають широкі можливості застосування, що включать, поміж іншими, локальні бездротові мережі, персональні бездротові мережі та транспортні мережі. Окрім того, наземні бездротові системи зв'язку, працюють на довжинах хвиль близького інфрачервоного спектра. Такі системи зазвичай послуговуються лазерним випромінюванням і забезпечують ефективний протокол – канал зв'язку з високою швидкістю передачі даних [4].

Передача даних за допомогою радіохвиль – вид бездротової передачі даних за допомогою використання радіохвиль різної довжини, відповідно до стандартів з якими працює той чи інший пристрій. Реалізація передачі даних через радіохвилі відбувається за допомогою використання радіопередавального та радіоприймального пристроїв. Радіопередавальний пристрій конвертує електричні сигнали в електромагнітне випромінювання. Це випромінювання сприймається радіоприймальним пристроєм і перетворюється в електричний сигнал. Радіохвилі дозволяють передавати дані в умовах коли пристрої передачі і зчитування сигналу не знаходяться в зоні прямої видимості, радіохвилі (залежно від довжини) можуть проходити через перешкоди та передаватись на велику відстань [5].

Бездротова передача даних у порівнянні з передаючою даних за допомогою фізичного з’єднання має низку недоліків, таких як: низька швидкість передачі даних, нижчий рівень безпеки та зниження якості сигналу через перешкоди. Проте, ці недоліки лише розшаровують сфери використання того чи іншого метода передачі даних. Для буденного використання, бездротова передача даних є незамінною з точки зору зручності, а недоліки не погіршують загальний досвід взаємодії з технологією. Окремої уваги потребує

питання безпеки передачі даних. Даний недолік усуває додаткова авторизація або автентифікація (у випадку якщо ресурс не є публічним).

Переважна більшість сучасних пристроїв мають вбудовані засоби для передачі даних бездротовим способом. Найпоширеніші з них – Bluetooth та Wi-Fi. За допомогою технології Wi-Fi виробники пристроїв створюють свої протоколи для безпровідної передачі даних (Air Drop, Huawei Share і тому подібні).

З розвитком технологій, Bluetooth втратив актуальність використання для передачі персональних даних через такі недоліки як: малий радіус дії, низька швидкість передачі даних, низький рівень безпеки. Саме тому Bluetooth використовують для бездротового підключення пристроїв вводу та виводу інформації до приладів, а Wi-Fi – для передачі персональної інформації.

Бездротові технології потребують як мінімум одного активного пристрою, який буде виконувати роль точки доступу, з якої буде проводитись передача даних, до якої буде виконано підключення інших пристроїв, приймачів.

Дані технології використовуються як для передачі великих обсягів інформації (фото-, відео-, аудіо-файли, програми, тощо), так і для відносно малих (здебільшого текстові файли, або повідомлення).

У випадку необхідності створення єдиної точки доступу для передачі невеликих повідомлень використання бездротових технологій передачі даних не є доцільними, через потребу створення постійного пристрою-точки доступу та ручного контролю для передачі даних. Для вирішення даної проблеми використовують технології безконтактної передачі даних.

Розвиток сучасних технологій дозволив створення дешевих носіїв інформації, які можуть розміщувати великі обсяги даних, залишаючись швидкими та компактними. Ще декілька десятиліть назад, користувачі могли собі дозволити носії з об’ємом в декілька кілобайтів. Технологічний прогрес дозволив значно покращити найголовніші характеристики носіїв інформації:

пришвидшити процес запису та зчитування інформації, збільшити обсяг інформації для запису, а також не тільки зберегти, а і зменшити розміри носія.

Зараз інформацію можна зберігати на твердотільних накопичувачах, на серверах хмарних сервісів, де використовуються ті самі твердотільні накопичувачі, але процес взаємодії користувача з технологією змінився: можна просто завантажити дані до серверу та мати до неї доступ з будь-якої точки світу де є можливість підключення до мережі інтернет, на чіпах які не потребують постійного живлення, а також просто зберігати зашифровану інформацію у вигляді згенерованого «ключа» (QR чи лінійний-код).

Питання зберігання великих об’ємів інформації наразі не є проблемою, проте для користувачів є актуальною можливість поширення статичної інформації, а іноді і динамічної інформації, без доступу до мережі. Бездротові носії наразі можуть зберігати невелику кількість інформації, тому для заощадження місця на пристрої використовують алгоритми стиснення даних. Носії зберігають інформацію у бінарному вигляді, або у якості ключа до алгоритму, який дозволить розшифрувати інформацію. Проте, можна забезпечити стиснення інформації на програмному рівні, щоб отримати додаткову перевагу і кількості інформації до запису [5].

Стиснення даних передбачає розробку компактного представлення інформації. Більшість представлень інформації містять велику кількість надлишковості. Надлишковість може існувати в різних формах. Вона може існувати у формі кореляції: просторово близькі пікселі на зображенні, як правило, також близькі за значенням. Надлишковість може бути пов’язана з контекстом: кількість можливостей для певної літери у фрагменті тексту різко зменшується, якщо попередньою літерою є q, у випадку використання англійської мови [6].

Це може бути ймовірнісним за своєю природою: літера е набагато частіше зустрічається у фрагменті англійського тексту, ніж літера q. Це може бути результатом того, як була сформована інформаційна послідовність:

озвучена мова має періодичну структуру. Або надмірність може бути функцією користувача інформації: дивлячись на зображення, ми не можемо бачити вище певної просторової частоти; отже, високочастотна інформація є надлишковою для цієї програми. Словник Merriam-Webster Dictionary визначає надмірність як «частину повідомлення, яку можна видалити без втрати важливої інформації». Таким чином, одним з аспектів стиснення даних є видалення надлишковості. Характеристика надлишковості передбачає певну форму моделювання. Отже, цей крок у процесі стиснення також відомий як моделювання. Цей процес називають декореляцією [6].

Після процесу видалення надлишковості, інформацію потрібно закодувати у двійкове представлення. На цьому етапі ми використовуємо той факт, що якщо інформація представлена за допомогою певного алфавіту, деякі літери можуть зустрічатися з більшою ймовірністю, ніж інші. На етапі кодування ми використовуємо коротші кодові слова для представлення літер, які зустрічаються частіше, таким чином зменшуючи середню кількість бітів, необхідних для представлення кожної літери [6].

Стиснення в усіх його формах використовує структуру або надлишковість даних для досягнення компактного представлення. Розробка алгоритму стиснення передбачає розуміння типів надлишковості, присутніх у даних, а потім розробку стратегій використання цих надлишковостей для отримання компактного представлення даних. Люди винайшли багато способів охарактеризувати та використовувати різні типи надлишковостей, присутні в різних видах технологій від телеграфу до стільникового телефону та цифрових фільмів [6].

Одним із способів класифікації схем стиснення є модель, яка використовується для характеристики надлишковостей. Однак більш поширені схеми стиснення поділяються на дві основні групи: стиснення без втрат і стиснення з втратами. Стиснення без втрат зберігає всю інформацію в стиснутих даних, а реконструкція ідентична вихідним даним. При стисненні з

втратами частина інформації, що міститься у вихідних даних, втрачається безповоротно. Втрата інформації в певному сенсі є платою за досягнення більш високого рівня стиснення. Ми починаємо наш аналіз схем стиснення даних, спочатку розглядаючи способи стиснення без втрат [5].

Стиснення без втрат передбачає пошук представлення який точно представлятиме вхідні дані. У даному процесі має бути реконструйована послідовність яка приведе до оригінального результату. Вимога про відсутність втрати інформації обмежує ступінь стиснення, який ми можемо отримати [6].

Для поставленої задачі найкраще підходять алгоритми стиснення інформації без втрати даних. Все залежить від типу персональної інформації яка буде розміщуватись на носієві. Виходячи зі сфери застосування інформаційної технології, а саме: збереження текстового посилання на ресурс, або текстової інформації, доцільним є використання алгоритмів стиснення даних без втрат. У іншому разі, доцільно було би розглядати також алгоритми, які мають втрату інформації у процесі стиснення. У випадку зберігання зображення чи текстової інформації, не чутливої до зміни допускається певна кількість інформації, якою можна знехтувати при конвертації інформації зі стисненого формату у початковий.

1.2 Аналіз існуючих рішень і аналогів для безконтактної передачі персональної інформації

Безконтактна передача інформації реалізується шляхом створення носія та пристрою-зчитувача. В минулому для передачі великих обсягів інформації використовувались магнітні диски, оптичні диски, та USB накопичувачі.

Дискета (магнітний диск) – це тип носія інформації, здатний зберігати електронні дані, такі як комп'ютерний файл. Дискету вперше було створено компанією IBM як альтернативу покупці жорстких дисків, які в той час були

надзвичайно дорогими. Ранні комп'ютери не мали приводів для читання компакт-дисків або USB, а дискети були єдиним способом завантаження нової інформації на пристрій або створення резервних копій. Дискети, в середньому, могли зберігати невеликий обсяг інформації – до 1.44 Мб [7].

Для роботи з дискетами комп’ютери того часу були оснащені спеціальними пристроями, які давали можливість зчитувати та записувати інформацію користувача на носій.

Оптичний диск – носій даних у вигляді пластикового чи алюмінієвого диска, призначений для запису даних за допомогою лазерного проміня. Інформація на дискових носіях зберігаються у вигляді дуже тонкої спіральної доріжки, нанесеної на спеціальний захищений шар диска, яка складається з мікроскопічних заглиблень і проміжків між ними. Залежно від типу диска, максимально можливий об’єм вміщуваної інформації міг досягати 17 Гб [8].

Для запису та зчитування даних з оптичних дисків використовують дисководи (деякі з них можуть лише зчитувати інформацію, інші – зчитувати і записувати).

Хоча оптичні диски виникли доволі давно, вони все ще використовуються для передачі інформації, проте все більше нових пристроїв не мають вбудованого дисковода, що показує зменшення актуальності використання даного типу носія.

USB накопичувач – носій інформації, що використовує флеш-пам’ять для для збереження даних. Накопичувачі можуть підключатись до пристроїв за допомогою різного виду USB (micro-USB, USB-C, USB-A) що дозволяє використання флеш накопичувача на різних типах пристроїв. USB накопичувачі зазвичай підтримують перезаписування. Флеш накопичувачі дозволяють зберігати великі об’єми даних: від 8 ГБ до 1 ТБ, залежно від типу накопичувача [9].

Зараз дискети втратили актуальність, а оптичні диски і USB накопичувачі починають втрачати актуальність використання. Це зумовлено

тим, що інтернет став поширенішим та значно дешевшим, що дозволяє відправляти велику кількість даних між користувачами без використання носіїв. Для збереження файлів з'явилась велика кількість «хмарних» сховищ даних, які можуть зберігати значно більшу кількість інформації та надають доступ до неї з будь-якого пристрою, підключеного до інтернету.

Актуальною стала проблема швидкої передачі малої кількості інформації без необхідності прямої взаємодії з носієм (підключення його у відповідний порт, мануального завантаження інформації).

У випадку створення локальної точки для поширення невеликих обсягів інформації є доцільним використання лінійних і двомірних кодів.

Лінійні коди (штрих-коди) використовуються для передачі невеликої кількості інформації. Вони здебільшого складаються з вертикальних штрихів і найчастіше використовуються в торгівлі для зберігання коду товару та друкуються на документах для збереження даних власника у закодованому вигляді [10].

Двовимірні коди можуть зберігати більший обсяг інформації ніж штрих-коди. Найбільш розповсюдженим з них є QR-код, який може зберігати в собі цифрову, текстову, бінарну інформацію. QR-код має вид квадрату з певним візерунком, в якому зашифрована певна інформація. З ростом кількості інформації, збільшується кількість деталей на візерунку. Максимальний об'єм даних, який можна записати в код – 2953 байт. Двовимірні коди мають широке використання в банківській сфері, соціальних мережах, торгівлі і зазвичай містять посилання на ресурс або діп-лінк у додаток [11].

QR-код мають низку ключових переваг у порівнянні з лінійними кодами:

- Дозволяє кодувати більшу кількість інформації;
- Легко розпізнається пристроєм для сканування;
- Може бути зчитаним навіть з незначними пошкодженнями візерунку.

Існує альтернатива лінійним і двовимірним кодам – NFC (near field communication) – технологія бездротового високочастотного зв'язку малого

радіусу дії. Ця технологія дає можливість обміну між пристроями (найчастіше смартфонами та безконтактними платіжними терміналами), що перебувають на відстані близько 10см. Також існують NFC-теги на яку можна записати інформацію розміром до 8кб (максимальний об'єм інформації залежить від типу чіпа, вбудованого в тег).

Здебільшого мітка перебуває в пасивному стані і не споживає енергії. При піднесенні пристрою з NFC-зчитувачем, тег активується від електромагнітного випромінювання і передає збережену інформацію.

Для запису даних на NFC-тег пристрій передає ключ-пароль та дані для запису. Ключ зберігається в пристрої, що гарантує захист від перезапису інформації [12].

Перевагами NFC-тегу в порівнянні з лінійними і двовимірними кодами є можливість запису великої кількості інформації, перезапис даних на існуючому тезі, маленькі розміри носія, швидкість взаємодії та захист зчитуваною інформації шляхом встановлення захисного коду-пароля.

Недоліками є обов’язкова наявність NFC-зчитувача, який може бути відсутнім в бюджетних моделях смартфонів та ціна тегу.

Безконтактний носій інформації – лінійний (також двовимірний) код або ж пристрій з вбудованим модулем для запису даних, який містить в собі деяку інформацію. Інформація може бути збережена у різних виглядах: посилання, адреса, діп-лікн у додаток, пароль до точки доступу Wi-Fi або ж файл.

Безконтактна передача персональної інформації стала актуальною з приходом карантинних обмежень та є найбезпечнішим варіантом обміну інформацією між людьми. Вона замінила рахунок в ресторані, оплату транспорту у кондуктора, паперові візитівки, меню та рекламні буклети. Зараз ці речі поширюються у вигляді роздруківок з QR-кодами або NFC-тегів з записаною на них інформацією, яку хочуть передати користувачу.

За допомогою смартфона з камерою та NFC-модулем на актуальній версії операційної системи (Android або IOS), яка офіційно підтримується виробником, користувач може зчитати носій даних.

Проте, існують випадки, коли необхідно розміщення великого обсягу інформації на носіях. Це можуть бути JSON файли, які описують товар, або ж запит у форматі curl з великою кількістю параметрів, картинки чи аудіо файли. Розміри такої інформації можуть сягати граничних можливостей носія який використовується для її запису. Як правило, додатки не підтримують можливість стиснення даних для заощадження місця на пристрої, а тому і створюють обмеження для користувачів з такими специфічними задачами. Це обмеження є актуальним як і для для великих підприємств зі своєї інфраструктурою автентифікації на різні рівні доступу, так і для звичайних користувачів метою яких є збереження великих обсягів даних.

Процедура стиснення даних вимагає присутності пристрою – дешифратора (або додатку), який здатний працювати з застосованим алгоритмом стиснення. Для сучасних пристроїв проведення таких маніпуляцій з даними не є тяжким завданням, яке вимагає високої обчислювальної здібності. Додаток який створює запис та стиснення даних може працювати і у зворотному напрямку: проводити розшифрування інформації та конвертувати її у оригінальний вигляд. Такий функціонал може бути опціональним, задля покриття більшої кількості аудиторії додатку.

Виходячи з вищевказаних аспектів функціонування та використання безконтактних носіїв, можна сформувати перелік вимог до інформаційної технології безконтактної передачі персональних даних:

- Використання інформаційної технології має бути умовно безкоштовним;
- Інформаційна технологія не повинна мати обмеження у використанні чіпів;

- Інформаційна технологія повинна бути орієнтована як на європейський, так і на американський ринок;
- Інформаційна технологія має бути простою для використання;
- Інформаційна технологія повинна бути сумісною з сучасними смартфонами, які перебувають на офіційній підтримці виробника;
- Інформація на носії повинна бути захищена від несанкціонованого перезапису;
- Інформаційна технологія повинна мати функціонал стиснення та розшифрування даних, який повинен бути опціональним, задля збереження гнучкості використання;

1.3 Аналіз відомих програмних рішень для передачі персональної інформації

З моменту широкого використання безконтактної передачі інформації, багато систем додали можливість поширення даних за допомогою безконтактних методів. Популярні мобільні додатки реалізували даний функціонал за допомогою пропрєітарних методів.

Спочатку розглянемо недоліки використання пропрєітарних (тобто які можуть не бути розпізнані будь-яким пристроєм, без спеціального додатку) методів передачі інформації.

У якості прикладу функціонування даного процесу є функціонал поширення персонального профіля у додатку Instagram, розробленого компанією Meta. Для передачі персонального посилання на профіль користувача розробники додатку використали модифікований QR-код (рис. 1.1). В QR записано посилання на відкриття додатку Instagram з завантаженням профілю, посилання на який було записано в код. За допомогою камери, вбудованої в додаток, інший користувач може просканувати згенерований код, після чого його буде перенаправлено на

профіль (рис. 1.2). Лише мобільний додаток Instagram може розпізнати специфічний код та отримати з нього інформацію. Звичайний додаток камери смартфону не в змозі розшифрувати інформацію, адже QR-код побудовано за власним алгоритмом додатку Instagram, який невідомий для стандартних додатків операційної системи [13].

Існує ще один вид обмеження базових можливостей безконтактної передачі даних за допомогою QR-коду. Існують додатки, які не модифікують QR-код, а використовують у якості інформації унікальній ідентифікатор тої чи іншої операції, дані про яку зберігаються на сервері. Отримати запитувані дані може лише додаток, в якому використовується пропрєітарний код. У додатку вбудована камера, яка зчитує інформацію з коду, та за допомогою запиту на сервер, з використання секретного ключа, дані розшифровуються та використовуються в додатку, або показуються користувачу. У деяких ситуаціях використання такого підходу є виправдане необхідністю захисту даних, як в банківських додатках для оплати, проте такі додатки не працюють при відсутності інтернету і потребують стабільного підключення.

Рисунок 1.2 – Загальний вигляд пропрієтарного QR-код додатку «Instagram»

Рисунок 1.2 – Загальний вигляд пропрієтарного сканеру QR-коду додатку «Instagram»

Недоліком даного методу передачі персональної інформації є примусове використання додатку для розпізнавання коду.

Існують програмні продукти які мають функціонал безконтактної передачі даних шляхом запису інформації на NFC-носій. Будь-який пристрій оснащений NFC модулем може отримати інформацію з NFC-тегу, звичайно, у випадку того якщо носій не захищено паролем. Цей факт можна сприймати як признак пропрієтарності, проте це є базовою можливістю використання NFC-тегу, використання якого залежить від розробників, або від пристрою який записує дані на носій. Для порівняння було обрано додатки: Popl, PhoneTag.

Popl – мобільний додаток, що надає функціонал для створення персонального NFC-тегу з можливістю персоналізації. Додаток має перелік соціальних мереж та сервісів які можна інтегрувати з системою. Користувач може скопіювати посилання на профіль Instagram, Facebook, LinkedIn, TikTok, тощо та занести його в систему Popl. Програма формує посилання та надає

інструмент для запису посилання на NFC-тег. Носій інформації можна закріпити в публічному місці, на роботі, на задній стороні телефону, де будь-хто зможе зчитати інформацію з тегу та перейти на сайт з посиланнями на соціальні мережі користувача [12].

Загальний вигляд інтерфейсу додатку зображено на рисунку 1.3.

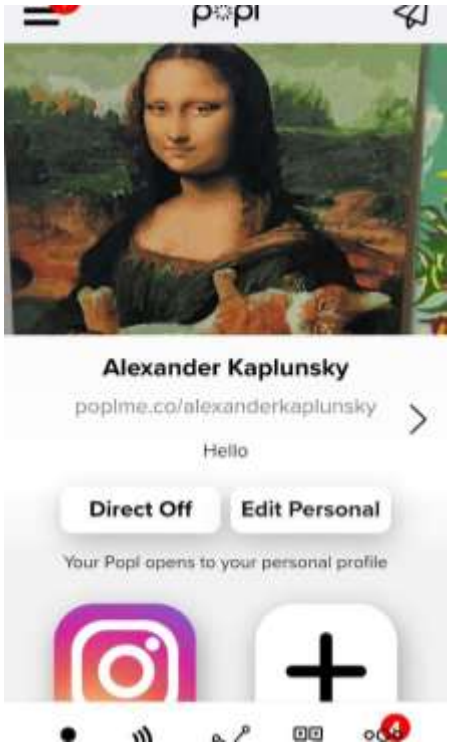


Рисунок 1.3 – Загальний вигляд інтерфейсу додатку Popl

Недоліками додатку Popl є висока ціна тегу, відсутність можливості записувати власні дані на носій та увімкнути режим стиснення даних для об’ємної інформації, проте сервіс має.

Перевагами додатку Popl є можливість створення персонального варіанту оформлення NFC-тегу, оптимізація додатку під старі пристрої та велика аудиторія користувачів, яка допомагає підтримувати актуальність та популярність додатку.

PhoneTag – додаток з ідентичним функціоналом до Popl, але з більшою кількістю соціальних мереж, також можливістю переказу коштів за допомогою сервісу PayPal та Venmo [13].

Додаток почав набирати популярність у кінці 2021 року, розроблений у відповідь на шалену популярність додатку Popl. Розробник зробив додаток меншим за обсягом від додатку конкурента, проте оптимізація додатку PhoneTag піж різні стартфони виконана з великою кількістю багів, що, зрештою, впливає на досвід взаємодії користувача. Загальний вигляд інтерфейсу додатку PhoneTag зображено на рисунку 1.4.

Рисунок 1.4 – Загальний вигляд інтерфейсу додатку PhoneTag

Перевагами додатку є можливість створення персонального дизайну NFC-носія інформації, велика кількість інтегрованих соціальних мереж, що є привабливим для користувача, а також можливість прямого перенаправлення у соціальну мережу за допомогою технології діп-лінкігу, яка працює зі смартфонами.

Недоліком додатку є нестабільна робота, високе використання ресурсів смартфону. Додаток надає можливість записати власний формат інформації на пристрій, проте містить обмеження: дозволяється записати лише посилання на ресурс, або невеликий текст. Відсутність стиснення інформації накладає обмеження на кількість даних, яку користувач може зберегти на носієві та поширити серед інших людей.

Отже розглянуті програмні продукти орієнтовані здебільшого на зберігання посилань на різні ресурси, або ж на зберігання відносно невеликої кількості інформації. Тим самим вони накладають обмеження на використання NFC-тегів для користувача і втрачають гнучкість.

Проаналізувавши усі аналоги, визначено їхні можливості та недоліки, які враховувались при створенні власного програмного забезпечення.(табл. 1.1). Для зручності, введемо назву розроблюваного програмного засобу – «Tappin».

Таблиця 1.1 – Порівняльні характеристики програмних продуктів

Критерій	Popl	PhoneTag	Tappin
Пропрієтарність	0	0	1
Можливість використання у випадку відсутності NFC-тегу	0	0	1
Можливість створення бібліотеки з посилань	1	1	1
Можливість стиснення даних при записі на носій	0	0	1
Зручний та зрозумілий інтерфейс	1	0	0
Можливість створення унікального дизайну NFC-тегу	1	1	0
Всього	3	2	4

Таблиця порівняльних характеристик показала, що розробка програмного продукту є доцільною. Продукт буде покривати недоліки наявних рішень та надавати широкий набір функцій, які необхідні у використанні інформаційної технології безконтактної передачі персональних даних.

1.4 Висновок до розділу 1

В даному розділі було розглянуто технології передачі інформації; проаналізовано існуючі способи передачі інформації, виділено актуальність безконтактної передачі даних; проведено аналіз методів стиснення, зокрема методи стиснення інформації з втратами та без втрат; проведено порівняння існуючих методів безконтактної передачі інформації, зокрема з фізичними носіями інформації та методи передачі без носіїв, сфери застосування методів безконтактної передачі, виділено їх переваги та недоліки, а саме можливість створення персонального дизайну носія інформації, стабільність та оптимізація роботи додатку на різних моделях смартфонів, функціональні можливості, пропрієтарність носіїв інформації, об’єм накопичувача, можливість стиснення даних; визначено задачі та вимоги до інформаційної технології безконтактного обміну персональними даними, розглянуто модулі-аналоги та проведено порівняльний аналіз між ними, що дозволяє перейти до етапу дослідження технології для безконтактної передачі персональних даних.

2 МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ БЕЗКОНТАКТНОГО ОБМІНУ ПЕРСОНАЛЬНИМИ ДАНИМИ

2.1 Аналіз принципів функціонування технології NFC

NFC (near field communication, ближній безконтактний зв'язок) – технологія безконтактної передачі даних малого радіусу дії, яка забезпечує обмін даними між пристроями, що знаходяться на відстані близько 10 сантиметрів один від одного [16].

Технологія NFC розроблена для обміну різними типами інформації, такими як номера телефонів, зображення, файли формату MP3 або даними цифрової авторизації між двома пристроями з підтримкою NFC, наприклад, мобільними телефонами, або між NFC-телефонами і сумісними RFID чіп картами або зчитувальними пристроями, розташованими в безпосередній близькості один від одного. Технологія NFC може бути використана в якості ключа доступу до контенту і для таких сервісів, як оплата за безготівковим розрахунку, оплата квитків і контроль доступу [16].

Технологія сумісна з широко використовуваним стандартом Smart Card на основі стандартів ISO / IEC 14443 A (наприклад, Mifare) і ISO / IEC 14443 B, а також JIS X 6319-4 (FeliCa). Для обміну між двома пристроями розроблені новий протокол ECMA-340 і стандарт ISO / IEC 18092 [16].

Таким чином, технологія NFC сумісна з уже існуючою інфраструктурою безконтактних карт, використовуваної в громадському транспорті і платіжних системах, але націлена перш за все на застосування в мобільних пристроях.

Стандарт ISO 14443 регламентує організацію зв'язку в межах суспільно доступних і неліцензованому радіочастот діапазону ISM (industrial, scientific and medical – промислові, наукові та медичні радіочастоти). Так само як і в технологіях, що відповідають цьому стандарту, в NFC зв'язок підтримується

за допомогою електромагнітної індукції, коли дві рамкові антени розташовуються в межах ближнього поля один одного. Через індуктивного зв'язку рамкових антен пристрій з якого ініціюється запит, потрапляє у поле пристрою приймача. Зміни кола приймального пристрою викликають амплітудні або фазові зміни в напрузі антени пристрою приймача. Даний метод модуляції називається навантажувальною модуляцією [16].

Робоча частота технології зв'язку NFC становить 13,56 МГц, швидкість передачі даних максимум 424 кбіт/с при відстані між пристроями до 4 см, при збільшенні відстані швидкість знижується до 212 кбіт/с і, на граничному для цієї технології відстані 20 см – до 106 кбіт/с. Сигнал піддається амплітудній маніпуляції з глибиною 100% (OOK, on-off keying, включено-виключено) або 10% (ASK, amplitude shift keying - амплітудна маніпуляція) і фазової маніпуляції BPSK (binary phase shift keying, двійкова фазова маніпуляція). На відміну від звичайних безконтактних технологій в даному частотному діапазоні, обмін даними між NFC-пристроями може бути як активно-активним, так і активно-пасивним, тому в стандарті NFC простежуються тісні зв'язки зі світом радіочастотної ідентифікації (RFID) [16].

При передачі інформації пасивного пристрою використовується амплітудна маніпуляція ASK. При обміні з активним пристроєм обидва пристрої рівноправні і виступають в якості поллінгових [17].

Полінг – варіант опитування готовності пристрою. Перший пристрій надсилає сигнал та чекає на відповідь від другого. Якщо є відповідь, або зміна в електромагнітному полі – підключення відбулось і пристрій готовий до роботи.

Кожен пристрій має власне джерело живлення, тому сигнал несучої відключається відразу після закінчення передачі. Активна фаза передачі поживає більшу кількість енергетичний ресурсів пристрою, але за рахунок швидкої передачі даних за допомогою NFC, процес великого споживання енергії є недовгим [17].

Схема взаємодії пристроїв при передачі інформації за допомогою NFC наведена на рисунку 2.1.

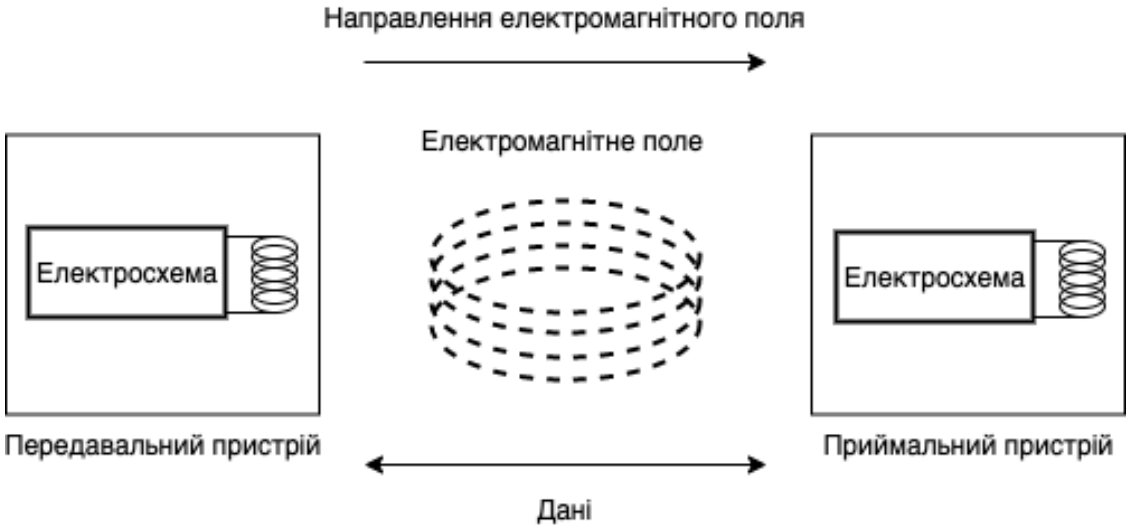


Рисунок 2.1 – Схема взаємодії двох пристроїв при передачі даних за допомогою NFC

За рахунок індуктивного зв'язку між опитуваних і прослуховуючих пристроїв пасивний пристрій впливає на активний. Зміна електромагнітного поля прослуховувального пристрою викликає зміну амплітуди або фази напруги на антені опитувального пристрою, який потрапляє в поле впливу. Цей механізм називається модуляцією навантаження. Вона виконується в режимі прослуховування із застосуванням допоміжної несучої 848 кГц. Залежно від стандарту застосовується амплітудна (ASK для 14443 A) або фазова маніпуляція (BPSK для 14443 B). Ще один пасивний режим, сумісний з FeliCa, здійснюється без допоміжної піднесе з маніпуляцією ASK на частоті 13,56 МГц [18].

В технології NFC підтримуються три основні режими роботи (рис 2.2):

– Режим емуляції карти (пасивний режим): NFC-пристрій працює як звичайна безконтактна картка відповідно до одним з сумісних стандартів;

– Ad-hoc режим: обмін інформацією між двома NFC-пристроями. Ініціатор (опитуваних пристрій) споживає менше енергії по порівняно з режимом читання чи запису, так як в даному випадку адресат (Приймальний пристрій) використовує свій власний джерело живлення;
– Режим читання або запису (активний режим): NFC-пристрій є активним і виробляє читання або запис в пасивну сумісну RFID-мітку [18].

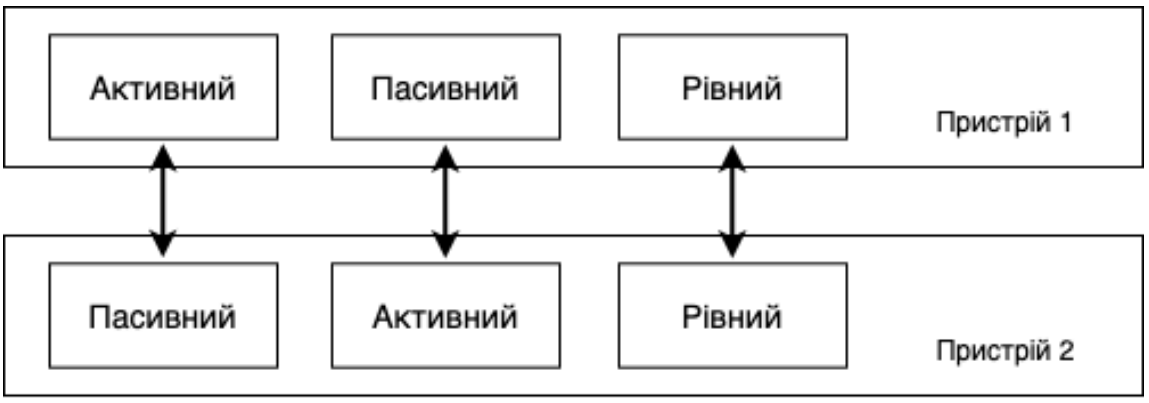


Рисунок 2.2 – Схема режимів роботи NFC

Кожен режим роботи (емуляція карти, спеціальний робочий, режим читання або запису) можна об'єднати з однією з наступних технологій передачі [17]:

– NFC-A (назад сумісна з ISO / IEC 14443 A);
– NFC-B (назад сумісна з ISO / IEC 14443 B);
– NFC-F (назад сумісна з JIS X 6319-4).

Щоб забезпечити підтримку різних технологій, NFC-пристрій в режимі опитування спочатку посилає відповідний сигнал запиту і чекає відповідь від NFC-A, NFC-B і NFC-F міток. Після отримання відповіді від сумісного пристрою NFC пристрій встановлює відповідний режим зв'язку (режим NFC-A, NFC-B або NFC-F). Алгоритм роботи наведено на рисунку 2.3 [18].

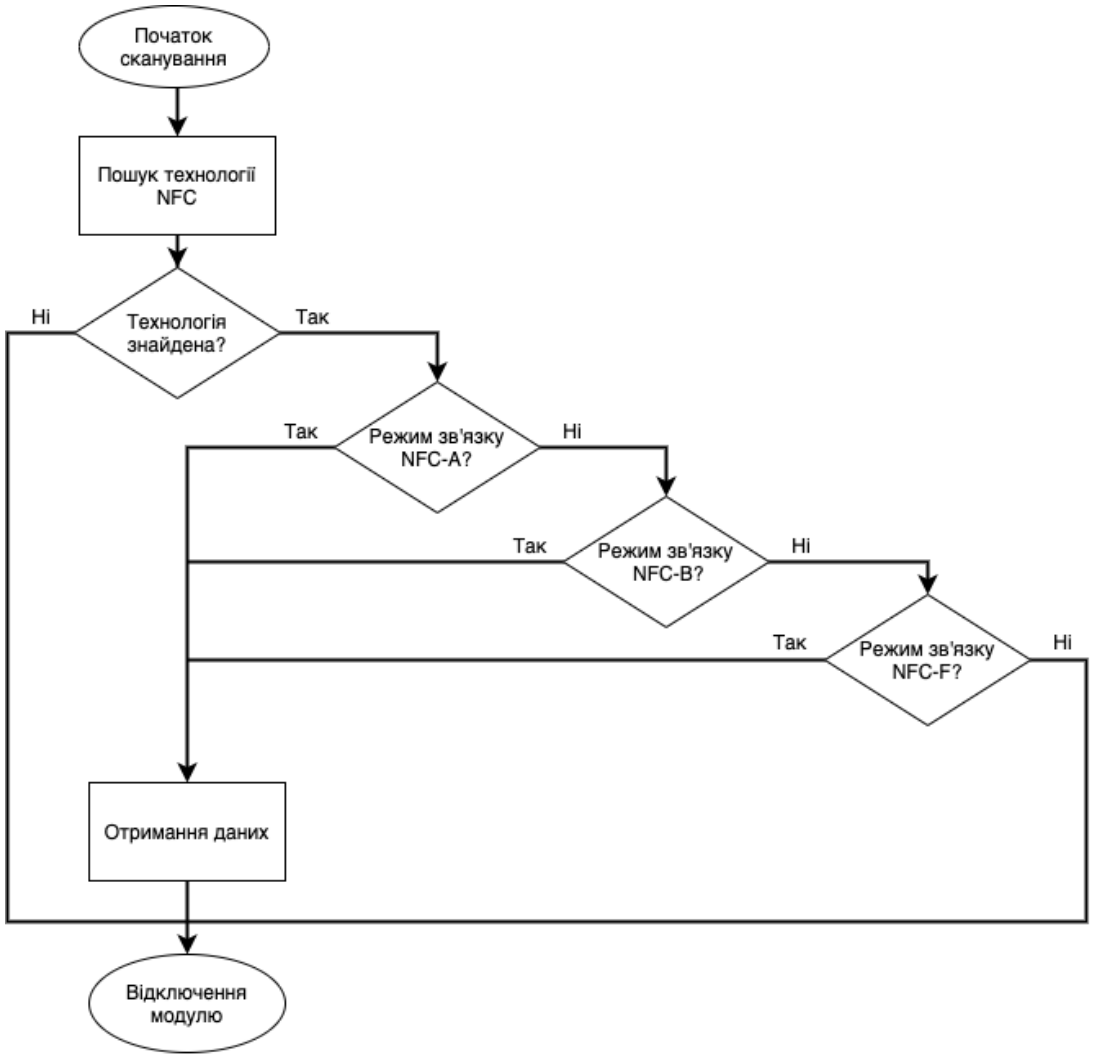


Рисунок 2.3 – Фрагмент схеми алгоритму пошуку режиму зв’язку NFC модуля

Схеми кодування і модуляції змінюються в залежності від активного або пасивного режиму зв'язку, радіо інтерфейсу NFC-A, -B, -F і бітової швидкості передачі даних. Для кодування використовується модифікований код Міллера, манчестерський код та NRZ-L код [19].

У таблиці 2.1 показані схеми кодування, модуляції і швидкість передачі даних для режимів зв'язку NFC-A, -B або -F [19].

Таблиця 2.1 – Характеристики стандартів NFC

Специфікація технічних стандартів радіоінтерфейсів NFC					
Стандарти асоціації NFC-Forum	Прийом(П)/ Сканування (С)	Кодування	Модуляція	Швидкість передачі даних	Частота
NFC-A	С	Модифікований код Міллера	Амплітудна маніпуляція (ASK) 100%	106 кбит/с	13,56 МГц
	П	Манчестерський код	Навантажувальна модуляція (ASK)	106 кбит/с	13,56 МГц +- 848 кГц
NFC-B	С	NRZ-L	Амплітудна маніпуляція (ASK) 10%	106 кбит/с	13,56 МГц
	П	NRZ-L	Навантажувальна модуляція (BPSK)	106 кбит/с	13,56 МГц +- 848 кГц
NFC-F	С	Манчестерський код	Амплітудна маніпуляція (ASK) 10%	212 / 424 кбит/с	13,56 МГц
	П	Манчестерський код	Навантажувальна маніпуляція (ASK)	212 / 424 кбит/с	13,56 МГц

NFC технологія використовує радіохвилі для комунікації та передачі даних між пристроями. Технологія Wi-Fi та Bluetooth також використовує радіохвилі для поширення зони доступу до джерела інформації, але структура приймача інформації відрізняється від приймача NFC. В основі NFC технології лежить електромагнітна індукція.

Явище електромагнітної індукції полягає у виникненні електричного струму в замкнутому електропровідному контурі при зміні магнітного потоку через площу цього контуру. Такий електричний струм називають індукційним. Індукційний струм у котушці з металевого дроту виникає також під час зміни сили струму в другій котушці, магнітне поле якої пронизує першу котушку. Індукційний струм виникає також під час руху котушки відносно нерухомого постійного магніту. Якщо з'єднана з гальванометром котушка рухається повільно в однорідному полі, то індукційний струм не виникає, бо кількість силових ліній, що перетинають котушку, весь час залишається незмінною [20-21].

Поява електричного струму в замкненому контурі під час зміни магнітного поля, яке його пронизує, свідчить про дію в контурі сторонніх сил не електростатичної природи або про виникнення ЕРС індукції. Явища електромагнітної індукції кількісно описують на основі встановлення зв'язку між ЕРС індукції і магнітним потоком. Ця величина залежить від значень вектора не в одній точці, а в усіх точках поверхні, обмеженої плоским замкненим контуром [20-21].

Саме завдяки електромагнітній індукції NFC дозволяє взаємодію з носіями, які не під'єднані до джерела живлення, наприклад, при оплаті банківською картою, оснащеною NFC чіпом. Банківська карта не має джерела живлення, тому NFC чіп перебуває у знеструмленому стані, хоча має в собі деяку кількість інформації, достатню для проведення оплати.

Банківський термінал оснащений NFC модулем з котушкою. При активації модуля, він переходить у режим зчитування інформації та утворює

електромагнітне поле з радіусом до 5 сантиметрів. При потраплянні банківської карти у межі дії електромагнітного поля NFC модуля, за допомогою електромагнітної індукції на чіп подається живлення, яке запускає виконання запрограмованих інструкцій. Перебуваючи в зоні дії електромагнітної індукції, NFC чіп на карті починає генерувати власне електромагнітне поле, виконуючи роль пристрою передавача. Після синхронізації сигналів пристроїв, термінал отримує дані про банківську карту. Дані зберігаються у вигляді двійкового коду, записаного на чіп. Пристрій приймач розшифровує двійкові дані та має інструкції для роботи з форматом інформації, отриманого з носія.

Цей приклад є найбільш простим для розуміння і використовується людьми щодня. Такий алгоритм роботи з NFC справедливий не лише для процесу оплати, а й для будь-яких сфер використання NFC чіпів (міток).

Запис даних на чіп схожий з процесом зчитування. NFC модуль має декілька варіантів роботи з чіпами: зчитування та запис.

Процес зчитування був розглянутий в прикладі з оплатою банківською картою та його алгоритм відомий. У випадку запису даних на NFC чіп, процес є дещо складнішим і залежить не лише від специфікацій NFC модуля, а й від специфікацій NFC тегу.

NFC теги можуть містити пароль, який встановлено на рівні мікросхеми. При спробі запису даних на мікросхему, NFC модуль забов’язаний також передати пароль, інакше тег відмовить у записі. Також, пароль може бути не встановлений в чіп. NFC пристрій може працювати в режимі без пароля. У випадку передачі запиту на запис з паролем, чіп запам’ятає його та потім буде працювати у режимі з паролем.

За допомогою пароля, NFC тег отримує захист від несанкціонованого запису, зчитування та перезапису інформації.

2.2 Моделі та алгоритми стиснення даних без втрат

Інформаційна технологія буде використовуватись для збереження посилання або текстових даних, чутливих до змін, тому запропоновано розглянути лише алгоритми стиснення інформації без втрати даних. З усіх відомих алгоритмів було обрано три, як самі популярні та найбільш підходящі до використання у створених умовах розробки: LZW, LZ77, LZMA.

LZW (Алгоритм Лемпеля - Зіва – Велча) – це універсальний алгоритм стиснення без втрати даних. Попередником алгоритму LZW є алгоритм LZ78, оприлюднений Абрахамом Лемпелем і Якобом Зівом. Цей алгоритм сприймався як математична абстракція до певного часу, допоки Тері Уелч не опублікував свою роботу з модифікованим алгоритмом, який отримав назву LZW [22].

Алгоритм стиснення реалізований таким чином, що послідовно зчитуються символи вхідного потоку даних, після чого відбувається перевірка, чи існує у створеній таблиці рядків початковий рядок. Якщо він існує, тоді відбувається зчитування першого символу, інакше, в стрім відправляється код для попереднього знайденого рядка, рядок записується в таблицю і відбувається наступна ітерація пошуку. Наприклад, у випадку стиснення тексту, в таблицю потрапить 256 записів, адже текст – байтові дані від 0 до 255 байтів. Якщо буде використовуватись 10-ти бітний код, тоді під коди для записів залишаються значення від 256 до 1023. Нові рядки формують таблицю послідовно, щоб можна було вважати порядковий номер запису (індекс) його кодом. Алгоритму розгортання (декодування) у точці входу потрібен лише закодований текст, адже алгоритм може відтворити таблицю шифрування по закодованому тексту. Алгоритм генерує однозначний код за рахунок того, що кожного разу, коли генерується новий код, новий рядок додається в таблицю

рядків. LZW постійно перевіряє, чи є рядок вже відомим, і якщо так, виводить існуючий код без генерації нового. Таким чином, кожен рядок зберігатиметься в єдиному екземплярі і матиме свій унікальний номер. Отже, при дешифруванні генерується новий рядок, а при отриманні вже відомого, рядок вилучається зі словника [22].

Наведемо приклад роботи алгоритму за допомогою псевдокоду:

1. Ініціалізація словника всіма можливими односимвольними фразами.
2. Ініціалізація вхідної фрази X першим символом повідомлення.
3. Зчитати черговий символ K з кодованого повідомлення.
4. Якщо знайдений кінець повідомлення, то видати код для X, інакше:
5. Якщо фраза X(K) вже є у словнику, присвоїти вхідній фразі значення X(K) і перейти до Кроку 2, інакше видати код X, додати X(K) у словник, присвоїти вхідній фразі значення K та перейти до Кроку 2 [22].

Псевдокоду недостатньо для того щоб зрозуміти роботу алгоритму, тому розглянемо приклад стиснення та декодування тексту. Спочатку створимо початковий словник окремих символів. У стандартному кодуванні ASCII є 256 різних символів, тому, щоб усі вони були правильно закодовані нам невідомо, які символи будуть присутні у вихідному тексті, а які — ні. Початковий розмір коду дорівнюватиме 8 біт. Якщо нам заздалегідь відомо, що у вихідному тексті буде менше різних символів, то цілком розумно зменшити кількість біт. Щоб створити таблицю, ми встановимо відповідність коду 0 відповідного символу з кодом біт 00000000, тоді 1 відповідає символу з кодом 00000001 і так до коду 255. Насправді ми вказали, що кожен код від 0 до 255 є головним.
Більше в таблиці не буде інших кодів, які мають цю властивість [22].

У міру зростання словника, розмір груп повинен зростати, щоб врахувати нові елементи. 8-бітові групи дають 256 можливих комбінації біт, тому, коли в словнику з'явиться 256 рядок, алгоритм повинен перейти до 9-

бітних груп. З появою 512-го рядка відбудеться перехід до 10-бітових груп, що дає можливість запам'ятовувати вже 1024 слова і так далі по прогресії [22].

У нашому прикладі алгоритму заздалегідь відомо про те, що використовуватиметься всього 5 різних символів, отже, їх зберігання буде використовуватися мінімальна кількість біт, що дозволяє їх запам'ятати, тобто 3 (8 різних комбінацій) [22].

Нехай у якості вхідних даних ми маємо текст, який складається з послідовність символів «abacabadabacabae», у такому випадку наведемо кроки реалізації стиснення. Візуалізація роботи алгоритму наведена у таблиці 2.2.

1. Згідно з викладеним вище алгоритмом, запишемо у порожній рядок «a» і перевіримо, чи є рядок «a» у таблиці. Оскільки ми під час ініціалізації занесли до таблиці всі рядки з одного символу, то рядок «a» є у таблиці.
2. Зчитуємо наступний символ «b» із вхідного потоку і перевіряємо, чи є рядок «ab» у таблиці. Такого рядка в таблиці поки що немає.
3. Додаємо в таблицю індекс 5 зі значенням «ab». У потік: «0».
4. Рядок «ba» не існує, тому записуємо у таблицю індекс 6 зі значенням «ba». У потік: <1>.
5. Рядок «ac» не існує, тому записуємо у таблицю індекс 7 зі значенням «ac». У потік: <0>.
6. Рядок «ca» не існує, тому записуємо у таблицю індекс 8 зі значенням «ac». У потік: <2>.
7. Рядок «ab» існує в таблиці, але «aba» - ні, тому записуємо у таблицю індекс 9 зі значенням «aba». У потік: <5>.
8. Рядок «ab» не існує в таблиці, тому записуємо у таблицю індекс 10 зі значенням «ab». У потік: <0>.
9. Рядок «ab» не існує в таблиці, тому записуємо у таблицю індекс 11 зі значенням «da». У потік: <3>.

10. Рядок «aba» існує в таблиці, але «abac» - ні, тому записуємо у таблицю індекс 12 зі значенням «abac». У потік: <9>.
11. Рядок «ca» існує в таблиці, але «cab» - ні, тому записуємо у таблицю індекс 13 зі значенням «cab». У потік: <8>.
12. Рядок «ba» існує в таблиці, але «bae» - ні, тому записуємо у таблицю індекс 14 зі значенням «bae». У потік: <6>.
13. Останній рядок «e», за ним йде кінець повідомлення, тому ми просто виводимо в потік <4>.

Таблиця 2.2 – Візуалізація ітерацій роботи алгоритму LZW

Поточний рядок	Поточний символ	Наступний символ	Вивід		Словник
			Індекс	Біти	
AB	A	B	0	000	5: AB
BA	B	A	1	001	6: BA
AC	A	C	0	000	7: AC
CA	C	A	2	010	8: CA
AB	A	B	-	-	-
ABA	B	A	5	101	9: ABA
AD	A	D	0	000	10: AD
DA	D	A	3	011	11: DA
AB	A	B	-	-	-
ABA	B	A	-	-	-
ABAC	A	C	9	1001	12: ABAC
CA	C	A	-	-	-
CAB	A	B	8	1000	13: CAB
BA	B	A	-	-	-
BAE	A	E	6	0110	15: BAE
E	E	-	4	0100	-

В результаті отримаємо закодоване повідомлення «0 1 0 2 5 0 3 9 8 6 4», яке на 11 біт коротше від вхідних даних: «abacabadabacabae».

Особливість LZW полягає в тому, що для розшифрування даних нам не потрібно зберігати таблицю рядків. Алгоритм побудований таким чином, що ми можемо відновити таблицю рядків, користуючись лише потоком кодів [22].

У випадку отримання закодованого повідомлення для дешифрації, необхідно мати лише початковий запис, а наступні записи можна отримати за допомогою алгоритму, адже вони є конкатенацією попередніх записів [22].

Перевагами використання алгоритму LZW є те що він не вимагає обчислення ймовірностей символів або кодів, для декомпресії не потрібно зберігати таблицю рядків для розпакування; алгоритм побудований таким чином, що ми можемо відновити таблицю рядків, користуючись лише потоком кодів, даний тип стиснення не вносить спотворень у вихідний.

Недолікам алгоритму є відсутність аналізу вхідних даних, від чого зменшується оптимальність.

LZ77 – це алгоритм стиснення без втрат, опублікований в статтях Абрахама Лемпеля та Якоба Зіва у 1977 та 1978 роках. Цей алгоритми найбільш відомі в сімействі LZ*, яке включає також LZW, LZSS, LZMA та інші алгоритми. Обидва алгоритми належать до алгоритмів зі словниковим підходом. LZ77 використовує так зване «ковзаюче вікно», що еквівалентно неявному використанню словникового підходу, вперше запропонованого в LZ78 [23].

Основна ідея алгоритму - заміна повторного входження рядка посиланням на одну з попередніх позицій входження. Для цього використовують метод ковзаючого вікна. Ковзаюче вікно можна представити у вигляді динамічної структури даних, яка організована так, щоб запам'ятовувати раніше сказану інформацію і надавати до неї доступ. Таким чином, сам процес стискаючого кодування згідно LZ77 нагадує написання

програми, команди якої дозволяють звертатися до елементів ковзаючого вікна, і замість значень послідовності, що стискається, вставляти посилання на ці значення в ковзаючому вікні. У стандартному алгоритмі LZ77 збіги рядка кодуються парою: довжина збігу-зміщення [23].

Кодуюча пара трактується саме як команда копіювання символів з ковзаючого вікна з певної позиції. Для виконання алгоритму потрібно повернутися у словнику на значення зміщення символів та скопіювати значення довжини символів, починаючи з поточної позиції. Особливість даного алгоритму стиснення полягає в тому, що використання пари довжина-зміщення є не тільки прийнятним, але і ефективним у тих випадках, коли значення довжини перевищує значення зміщення [23].

LZ77 використовує принцип ковзаючого вікна, що ковзає за повідомленням. Нехай, на поточній ітерації вікно зафіксовано. З правого боку вікна нарощуємо рядок нижчого ранга, поки воно є в рядку <ковзаюче вікно + нарощуємий рядок> і починається в ковзаючому вікні. Назвемо рядок, що нарощується, буфером. Після нарощування алгоритм видає код, що складається з трьох елементів:

- зміщення у вікні;
- довжина буфера;
- останній символ буфера [22-23].

Наприкінці ітерації алгоритм зсуває вікно на довжину, що дорівнює довжині буфера + 1 [22].

Нехай у якості вхідних даних ми маємо текст, який складається з послідовність символів «abcabckkkkkk», у такому випадку кодування буде відбуватись наступним чином: у тексті двічі зустрічається поєднання «abc», тому його можна записати в словник, в початковому тексті залишимо посилання на словник. В такому разі початковий текст можна інтерпретувати як «11kkkkkk» і запам'ятати, що індексу 1 присвоєне значення «abc».

У випадку якщо текст міститиме символ еквівалентний індексу, наприклад «abc1abckkk», пропонується кодування з урахуванням бітів. Вважатимемо, що текст складається з впорядкованого набору бітів. Один символ дорівнює значенню у 8 біт. Послідовно зчитуємо текст, формуючи вихідний файл. У випадку якщо символ зустрівся один раз, або знаходиться на останньому місті в тексті, на вихід надсилається 1 біт та 8 бітів символу. Ця дія також справедлива у випадку якщо символ не новий, але в парі з наступним ще не зустрічався. Таким чином при дешифруванні, виконавець буде знати, що отримавши біт 1, необхідно вивести наступні 8 бітів, адже вони ідентифікують символ. Якщо символ вже зустрічався, то парі можна присвоїти два біта – 01 і ідентифікатор існуючої пари. Повертаючись до вхідних даних, при використанні бітового алгоритму на виході отримаємо значення 001 11, а від букви k буде записано 9 бітів. Залишок тексту зашифровано сімома бітами: 00001 01 [22-23].

Перша група нулів, яка закінчується одиницею означає що раніше в тексті була зустрінута певна послідовність символів. Друга група вказує на якій відстані потрібно шукати прототип [23].

Для виконання декодування за алгоритмом LZ77 проводиться зчитування серії бітів до першого одиночного. Нехай x – кількість бітів. Якщо в черзі на зчитування знаходиться порядок 00100.., тоді зчитується 3 біти (x=3), але якщо перший біт – 1, тоді зчитується лише він (x=1). Нехай k – кількість бітів, які будуть взяті з розшифрованого тексту. Тоді для алгоритму розшифрування будуть справедливі наступні кроки [23]:

1. Якщо x >= 17, то зчитуються чергові 8 біт і записуються в молодші біти k.
2. Якщо x > 17, тоді зчитуємі 8 біт записуються у старші біти k.
3. Якщо k = 1, тоді зчитуються наступні 8 біт і надсилаються на вихід алгоритму у якості декодованого символу тексту.

Перевагою використання алгоритму LZ77 є те що він дозволяють кодувати послідовності символів різної довжини.

Недоліки використання алгоритму LZ77: неможливість кодування підрядків, що віддаляються один від одного на відстані, більшій за довжину словника, довжина підрядка обмежена розміром буфера, низька ефективність при кодуванні незначного обсягу даних.

LZMA – це алгоритм словникового стиснення даних, вихідні дані якого закодовані інтервальним кодуванням, що використовує складну модель обчислення ймовірності появи кожного біта. Система стиснення знаходить відповідності, використовуючи словникову структуру даних, і створює потік символів і посилань фраз, які знаходяться у словнику, закодованим 1 бітом інтервальним кодувальником [24].

Головною інновацією LZMA було те, що замість загальної байтової моделі, модель LZMA використовувала бітові поля, що залежать від контексту, в кожному поданні літер або фраз. Ця модель є простою, як і бітова модель, але дає кращий коефіцієнт стиснення, тому що уникає змішування незв'язних бітів разом у тому самому контексті [24].

Алгоритм кодування LZMA базується на використанні алгоритму LZ77 з використанням дельта-фільтру та інтервального кодування [24].

На вході в алгоритм, дані пропускаються через дельта-фільтр, де вони перетворюються та модифікуються для подальшого кодування. Отримана послідовність бітів піддається словниковому стисненню за алгоритмом LZ77. На виході з алгоритму з «ковзаючим вінком», отримуємо код, який для досягнення кращого стиснення піддається інтервальному кодуванню. В результаті отримується інтервал цілих чисел, який і буде відповідати вихідній послідовності [24].

Дельта фільтр перебудовує вхідні дані для ефективного стиснення за алгоритмом LZ77. Перший байт на виході збігається з першим байтом на вході, наступні байти представлені як різниця між поточним і попереднім

байтом (таб. 2.3). Дельта-кодування оптимізує набір даних для більш ефективного стиснення за алгоритмом LZ77 [22, 24].

Таблиця 2.3 – Приклад роботи дельта-кодування в алгоритмі LZMA

Вхідна послідовність:	2,3,4,6,7,9,8,7,5,3,4
Закодована послідовність:	2,1,1,2,1,2,−1,−1,−2,−2,1
Кількість різних символів на вході:	8
Кількість різних символів на виході:	4

Інтервальне кодування. При інтервальному кодуванні всі символи повідомлення кодуються як одне число, щоб досягти найкращого коефіцієнта стиснення. Це працює ефективно з ймовірностями появи символу який не дорівнює степеню двійки. Інтервальне кодування працює так:

1. Виділяється діапазон цілих чисел і оцінка ймовірності входження для символів.
2. Вихідний діапазон чисел ділиться на піддіапазони, розмір яких пропорційний ймовірності входження символу, яким вони відповідають.
3. Кожен символ повідомлення кодується, після чого діапазон скорочується до розміру діапазону щойно закодованого символу і знову ділиться за ймовірністю входження [24].

Перевагою використання алгоритму LZMA є те, що він має більш високий коефіцієнт стиснення, динамічний розмір словника та потребує менше пам’яті для проведення декодування.

Недоліком використання алгоритму LZMA є те що він не працює однаково ефективно зі всіма типами даних.

Розглянувши алгоритми LZW, LZ77 та LZMA, провівши їх аналіз, виходячи з поставленої задачі та з переваг та недоліків кожного, для виконання роботи було обрано алгоритм стиснення LZW, через його простоту

застосування, меншу кількість ключової інформації для проведення дешифрування, швидку роботу з текстовим форматом даних.

2.3 Розробка загальної структурної схеми функціонування інформаційної технології безконтактного обміну персональними даними

Розробимо загальну структурну схему інформаційної технології безконтактної передачі персональних даних. Модуль складається з 6 основних компонентів:

- мобільний додаток;
- модуль стиснення даних;
- програмний інтерфейс (API);
- NFC API;
- NFC тег;
- веб-сторінка профілю;
- база даних.

Опишемо кожен із компонентів.

Мобільний додаток представляє собою сторінку профілю користувача, зі списком соціальних мереж, посилань на сайт, мобільним контактом. Користувач має можливість обрати соціальні мережі та ввести посилання на них або записати будь-яку текстову інформацію на носій з можливістю увімкнути режим стиснення даних. Дані про користувача будуть зберігатись на сервері. Після внесення інформації, користувач матиме змогу записати посилання на персональну сторінку з переліком введених соціальних мереж, контактів на NFC-тег, або згенерувати QR-код у випадку відсутності NFC-модуля на смартфоні.

Модуль стиснення даних відповідає за обробку текстової інформації, містить в собі реалізацію алгоритму LZW з можливістю стиснення даних, та подальшої дешифрації. Модуль є опціональним для користувача і

використовується при створенні нового запису на тег, а також при скануванні самостійно перевіряє чи інформація у стисненому вигляді та залежно від цього проводить дешифрацію, підготовлюючи інформацію до опрацювання основним функціоналом мобільного додатку.

Програмний інтерфейс (API) представляє собою серверну частину, використовується для збереження даних користувача в базу даних, для авторизації та ідентифікації користувача, а також для валідації введених даних.

NFC API використовується для запису персональної інформації на NFC-тег. NFC-тег використовується у якості безконтактного носія інформації. Мобільний додаток за допомогою NFC API буде записувати посилання на профіль користувача на тег. Мітка буде зберігати інформацію та передавати її у тому випадку якщо в полі дії з’явиться пристрій з активованим зчитувачем NFC.

Веб-сторінка профілю використовується для відображення обраних соціальних мереж користувача та для переадресації користувача на соціальну мережу за обраним посиланням.

База даних зберігає дані профілю та інформацію між сесіями в додатку, медіа-файлів (фотографія профілю користувача).

Загальна структурна схема роботи системи показана на рисунку 2.5.

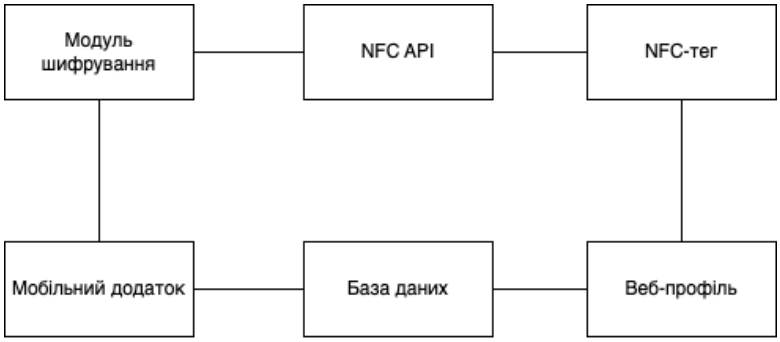


Рисунок 2.4 – Загальна структурна схема інформаційної технології безконтактного обміну персональними даними

2.4 Моделювання інформаційної технології безконтактного обміну персональними даними із використанням мови UML

Модель – це абстракція, яка створюється з метою осмислення чого завгодно перед тим, як його створювати [25].

Абстрагування – це вибіркове вивчення деяких аспектів проблеми. Основна мета абстрагування полягає в тому, щоб ізолювати аспекти, важливі для деякої цілі, і відкинути всі інші [25].

UML (англ. Unified Modeling Language) — уніфікована мова моделювання. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, яка називається UML-моделлю [25].

Для опису основного циклу роботи програми можна використати різні діаграми UML, зокрема, діаграму активності та діаграму послідовностей.

Діаграма активності є аналогом звичної схеми алгоритму програми, тобто відображає, як поток управління переходить з одної діяльності в іншу.

Діаграма послідовностей також показує життєвий цикл певної сутності, але при цьому зображає їх на єдиній часовій осі, взаємодію деяких сутностей (акторів) та відносну тривалість процесів. Тому використання діаграми послідовностей дає більш повне представлення про цикл роботи програмного забезпечення [25].

До об’єктів діаграми послідовності можемо віднести зазначені у попередньому розділі компоненти інформаційної технології, а також додамо два об’єкт – користувача для відображення взаємодії. Таким чином, отримаємо 5 об’єктів: користувач, веб-клієнт, NFC тег, сервер, база даних. Розроблена діаграма послідовностей зображена на рисунку 2.6.

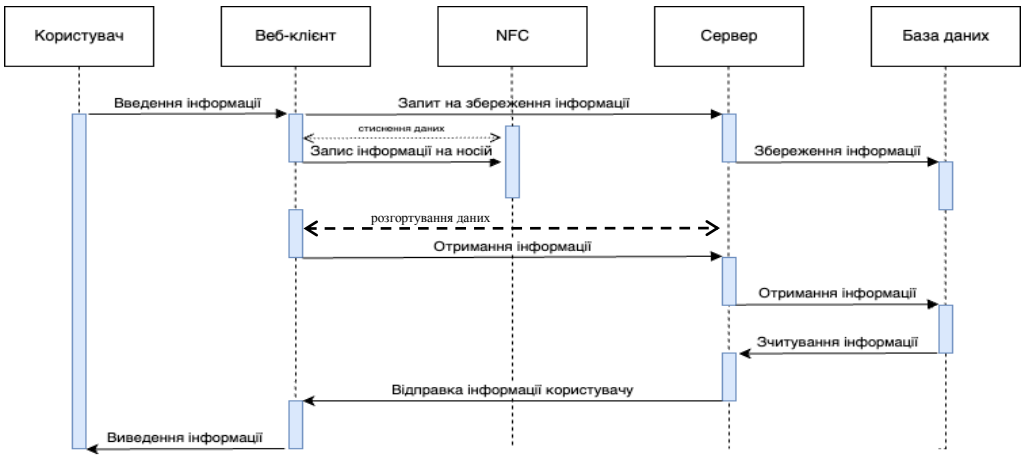


Рисунок 2.5 – Діаграма послідовності взаємодії модулів інформаційної технології безконтактного обміну персональними даними

2.5 Висновок до розділу 2

В даному розділі було розглянуто специфіку функціонування NFC технології; розглянуто процес взаємодії пристроїв з одночасно увімкненими NFC пристроями; проведено аналіз режимів роботи та наведено алгоритм взаємодії пристрою NFC з сумісними пристроями; Наведено сценарії використання NFC модуля в пристроях; розглянуто принцип функціонування технології з технічної сторони; описано принцип явища електромагнітної індукції. Проаналізовано алгоритми стиснення даних, наведено приклади їх роботи та обрано алгоритм LZW. Розроблено структуру інформаційної технології безконтактного обміну персональними даними та наведено схему взаємодії складових модуля. Виходячи з структури модуля, спроектовано основні класи інформаційної технології та наведено діаграму класів. На діаграмі класів відображено основні набори даних, які необхідні для роботи класу, а також функції (методи) які використовує клас. Спроектовано UML діаграму Отримані результати дослідження дозволяють перейти до проектування усього модуля.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ БЕЗКОНТАКТНОГО ОБМІНУ ПЕРСОНАЛЬНИМИ ДАНИМИ

3.1 Обґрунтування вибору архітектури інформаційної технології безконтактного обміну персональними даними

Архітектура програмного забезпечення – це структура програми, системи або модуля, яка відображає взаємодію між елементами системи, їх властивості та логічні частини за які відповідає той чи інший елемент. Побудування архітектури є першим етапом розробки ПЗ. При проектуванні модулю, орієнтуючись на поставлене технічне завдання, необхідно розробити найбільш оптимальну архітектуру, враховуючи швидкість роботи майбутнього проекту, зв'язки основних елементів системи, а також закласти можливість масштабування архітектури. Архітектура програмного забезпечення створює обмеження для написання функціонала, декларує єдиний стиль коду та закріплює ключові кроки, створені на ранніх етапах розробки [26].

Інформаційна технологія безконтактного обміну персональними даними має клієнт-серверну архітектуру.

Сервер – потужна обчислювальна машина, призначена для збереження та обробки інформації та її передачі між мобільним додатком та веб-профілем.

Клієнт – приймає та обробляє введену користувачем інформацію (мобільний додаток), а також відображає існуючі дані користувача (веб-профіль).

Принцип роботи такої архітектури полягає в тому, що клієнт відповідає за пряму взаємодію з користувачем за допомогою створеного інтерфейсу. Клієнт збирає необхідну інформацію від користувача додатку, за необхідності опрацьовує та модифікує, а потім відправляє на сервер за допомогою HTTP

або TCP протоколів передачі даних (найпопулярніші протоколи, які задовільнять потребу комунікації клієнта та сервера). TCP протокол відрізняється від HTTP протоколу тим, що дозволяє створити постійне підключення між двома елементами системи (у нашому випадку клієнт та сервер), саме тому зміни можуть динамічно передаватись у випадку зміни даних у будь-якому елементі системи. Сервер відповідає за обробку та збереження надісланої інформації з клієнта. Сервер може обробляти декілька клієнтів, або система може містити декілька серверів, які виконують різні функції, залежно від побудованої архітектури, та зв’язуються між собою за допомогою HTTP або TCP протоколів. Сервер має доступ до бази даних системи, та відповідає за безпеку доступу до неї, а також за операції з інформацією. При побудові архітектури, можна вказати пріоритетність запитів до серверу, таким чином сервер може в першу чергу обробляти більш важливі для роботи систему функції [26].

Структурна схема взаємодії клієнт-серверної архітектури зображено на рисунку 3.1. Схема відображає взаємодію між сервером, базою даних, та двома клієнтами інформаційної технології безконтактного обміну персональними даними.

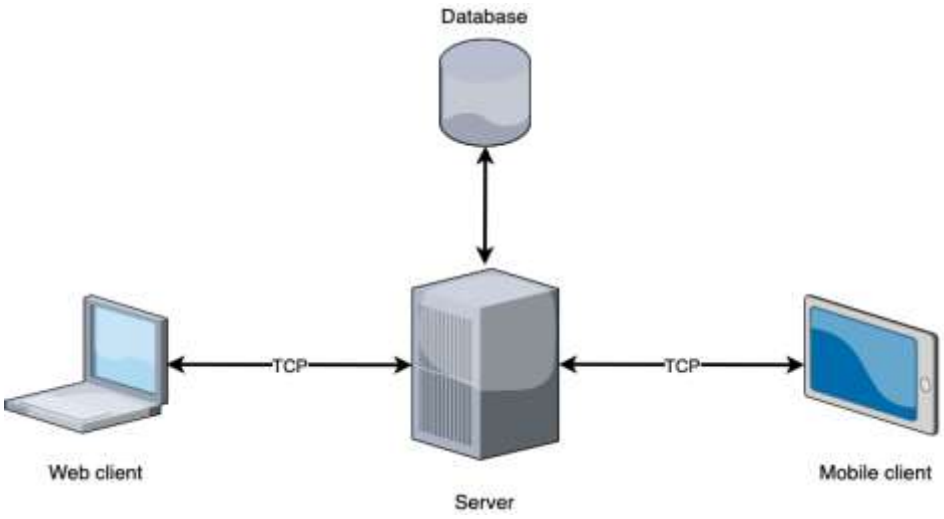


Рисунок 3.1 – Схема системи клієнт-сервер

Функції серверу:

- збереження інформації;
- забезпечення безпеки збереження інформації;
- обробка інформації з клієнту;
- авторизація, автентифікація запиту;
- відправка результату обробки інформації на клієнт.
- Функції, які реалізуються на стороні клієнта:
- відображення інтерфейсу;
- стиснення інформації;
- створення запитів на сервер;
- отримання даних користувача.

3.2 Обґрунтування вибору мови програмування

Для розробки інформаційної технології безконтактного обміну персональними даними необхідно розробити клієнтську та серверну частини. Клієнтська частина міститиме в собі два клієнта: мобільний додаток та веб додаток.

Так як дана модель складається з трьох компонентів, підтримка та розширення функціоналу може створити проблему з пошуком кадрів та дороговизною роботи. Найбільш раціональним вибором мови програмування є обрання популярної мови з можливістю створення кросплатформених додатків.

На сьогоднішній день найпопулярнішими мовами для створення кросплатформених додатків є Dart, C# та JavaScript.

Розглянемо мову програмування Dart.

Dart розроблена компанією Google, призначена, перш за все, для розробки настільних, веб-, мобільних та серверних додатків. Це означає, що програму можна компілювати під різні платформи – Windows, Android, IOS,

Browser. Dart – клас-орєнтована мова, в основу якої увійшли такі мови як Java, JavaScript, Smalltak. Основними принципами даної мови програмування є простота, ефективність та масштабування. Dart об’єднує в собі рішення з більшлю потужністю зі знайомим синтаксисом, високою читаємістю та зрозумілістю коду [27].

Основні можливості Dart:

- Класи та інтерфейси являють собою простий та зрозумілий механізм для створення API. Конструкції додають інкапсуляцію та повторне використання компонентів;
- Використання опціональних типів. Мова програмування надає можливість використання статичних типів даних, але не обмежує використання нетипізованих змінних. Це дозволяє реалізувати міграцію з простого нетипізованого додатку, до комплексного модульного додатка зі строгою типізацією;
- Інструменти для розробки: Dart має велику кількість середовищ для розробки, та широкий асортимент бібліотек для розробки та підтримки коду.
- Продуктивність коду. Швидкість роботи програм на Dart є високою через оптимізацію та використання практик, зарекомендувавших себе при створенні старших мов програмування.
- Масштабованість. Програми написані на Dart легко піддають масштабованості через використання класово-орієнтованої архітектури.
- Можливість створення кросплатформених додатків. Dart використовується для написання серверних додатків, а фреймворк Flatter дозволяє створювати мобільні та веб-додатки [27].

Dart не є універсальним рішенням проблем. Він не зможе дати таку саму продуктивність коду при створенні ігор як C, він не оптимізований для обробки великої кількості даних як Python. Dart – це інструмент зі своєю

областю застосування. JavaScript може виконувати ті самі функції, що і Dart, та має перевагу у кількості бібліотек. При застосуванні стандартних бібліотек Dart має перевагу перед JavaScript. Більшість API враховує різницю двигунів браузерів і виправляє неочікувану поведінку коду в різних середовищах виконання. Функціонал стандартних бібліотек Dart дозволяє вирішити велику кількість проблем, для вирішення яких на JavaScript потрібно використовувати сторонні бібліотеки, які можуть конфліктувати між собою або мати вплив один на одного, через що поведінка коду може бути неочікуваною [27].

У Dart робота з конфліктами реалізована краще. Мова програмування має вбудовані методи обходу конфліктів та мала кількість бібліотек не створює надлишкового навантаження коду. Менша кількість бібліотек зменшує варіативність вибору та концентрує увагу користувачів на підтримку та розвиток бібліотеки [27].

Ключовим фактором при обранні мови програмування є кількість активних програмістів. JavaScript та C# мають велику кількість користувачів, інтернет ресурси переповнені курсами, форумами, готовими рішеннями питань, які виникають в процесі розробки. Громаду Dart формують інженери, для яких Dart не є першою мовою програмування, тому типові приклади та проблеми не виносяться на обговорення, а пошук рішення є складним.

C# – об’єктно орієнтована мова програмування з C-подібним синтаксисом розроблена компанією Microsoft. Програміст, знайомий з такими мовами як Java та C++ чи C, зможе швидко зрозуміти за вивчити особливості написання коду на C# та почати писати код. Синтаксис C# позбавлений проблем C++ та надає такі потужні можливості, як обнуляємі значення типів, перерахування, делегати, прямий доступ до пам’яті. C# підтримує універсальні методи та типи, які підвищують безпеку написання та потужність коду. Мова програмування підтримує основні принципи ООП – інкапсуляцію, наслідування та поліморфізм. Клас може наслідуватись напряму від

батьківського класу, але може реалізовувати в собі велику кількість інтерфейсів [28].

C# є кросплатформеною мовою програмування. .NET Core фреймворк дозволяє створювати додатки для Windows, Linux, Mac OS. ASP.NET Core використовується для написання серверних та веб-додатків. Фреймворк має велику кількість шаблонів та готових рішень для створення програмних продуктів. Xamarin використовується для створення Android та IOS додатків з використанням мови програмування C#. Також C# використовують для машинного навчання та створення інтелектуальних, аналітичних систем [28].

C# має велику кількість базових бібліотек, структур та інструментів для написання програм різного роду. Вирішення стандартних проблем не займає велику кількість часу, так як розробники платформи .NET додали функціонал для вирішення різних типів задач [28].

Мова є сучасною, тому актуальний функціонал реалізований або перебуває в розробці. При компіляції код конвертується в проміжну мову, яка виконується на клієнтській машині. Даний принцип конпіляції не дозволяє писати такий швидкий код, як на C++ чи Golang. На противагу потужності, C# має велику кількість інструментів та можливостей для оптимізації [28].

Загалом, C# має велику перевагу при створенні об’ємних проектів з великою кількістю логіки. .NET надає архітектурні переваги, які покращують масштабування, підтримку та надійність коду [28].

Перевагами мови C# є:

– Масштабованість. Архітектурні обмеження .NET спонукають до написання надійного модульного коду з можливістю перевикористання та перевизначення калсів програми;
– Простота та гнучкість. C# має велику кількість стандартних інструментів та бібліотек для розробки, які дозволяють уникнути типових проблем протягом розробки проекту;

– Підтримка спільноти. C# широко використовується для створення додатків на різних платформах і має велику спільноту, що спрощує пошук рішень задач різного рівня складності [28].

Основним недоліком використання C# є орієнтованість на створення великих проектів, тому швидкість розробки значно більша ніж у JavaScript та Dart. У випадку розробки малооб'ємних проектів використання C# не є доцільним. Також, платформа Xamarin не займає провідних позицій у створені кросплатформених додатків, тому пошук кваліфікованих розробників є проблематичним, що ускладнює підтримку програмного продукту [28].

JavaScript – це обєктно–орієнтована мова, яка використовується для розробки веб-, мобільних, настільних та серверних доадтків. Також мова має підтримку функціонального програмування [29].

JavaScript не компілюється, на відміну від Dart та C#, а інтерпретується інтерпретатором в реальному часі. Ця мова програмування широко використовується для створення кросплатформених додатків. JavaScript має велику кількість бібліотек для написання серверних, мобільних, настільних та веб-додатків [28].

Бібліотеки для створення кросплатформених додатків:

– Мобільні додатки: React Native, Expo, Apache Cordova;
– Серверні додатки: NestJS, NodeJS, ExpressJS;
– Веб-додатки: React JS, Vue JS, Angular JS, Next JS;
– Настільні додатки: React Native, Electon JS.

JavaScript у порівнянні з C-подібними мовами має корінні відмінності: об'єкти, з можливістю інтроспекції; функції як об'єкти першого класу; автоматичне приведення типів; автоматичне прибирання сміття; анонімні функції [28].

JavaScript має велику підтримку спільноти. Існує велика кількість ресурсів з прикладами коду, вирішенням типових проблем, а також форумів,

де можна знайти відповідь для рішення поставленої задачі. Низький поріг входу та популярність мови створює великий відсоток JavaScript розробників на ринку з надлишковою кількістю кадрів. Це формує умови для легкого пошуку кваліфікованих та легкозамінних працівників [29].

Зараз JavaScript популярний та стрімко розвивається. Розробники бібліотек та фреймворків стабільно випускають оновлення та реалізовують новий функціонал, що спрощує написання проектів [29].

Недоліками JavaScript є:

– Відкрита архітектура. Відсутність архітектурних обмежень не створюють єдиного стилю написання коду, що провокує написання немасштабованого та тяжкопідтримуємого коду. Цей недолік справедливий для використання JavaScript без фреймворків. Більшість фреймворків (наприклад Angular та NestJS) мають архітектурні обмеження.
– Нестрога типізація. Відсутність строгої типізації провокує написання низькоякісного коду з великою кількістю помилок та тяжкою підтримкою. Даний недолік є справедливим лише для JavaScript. TypeScript має строгу типізацію, що вирішує описані проблеми, але дотримання правил в TypeScript є опціональним.
– Низька швидкість роботи. Інтерпретація коду в реальному часі створює низьку потужність коду. Тому при написанні великих проектів необхідно робити оптимізацію коду [29].

Незважаючи на недоліки JavaScript, дана мова є універсальним інструментом для написання кропслатформених додатків будь-яких розмірів з можливістю безпроблемної підтримки. Швидкість і результативність розробки мовою JavaScript є кращою ніж у розглянутих аналогів.

У результаті аналізу мов програмування мова програмування JavaScript є найкращою для розробки інформаційної технології безконтактного обміну персональними даними. JavaScript є найбільш пристосованою для розробки

кросплатформених додатків з невеликим обсягом функціоналу. Програми написані мовою програмування JavaScript легко підтримувати та розвивати.

3.3 Обґрунтування вибору фреймворків та бібліотек для розробки

Так як JavaScript є не тільки однією з найпопулярніших мов на ринку, а й найпопулярнішою мовою для написання веб-додатків, для нього розроблено велику кількість бібліотек та фреймворків. На ринку існують як нові продукти, які постійно оновлюються та мають високу підтримку розробниками – React JS, Vue.js, Next JS, Angular JS так і старі та неактуальні – Jquery, YUI, які використовуються лише для підтримки старих проектів.

Також можна використовувати «чистий» JavaScript, але це значно знижує швидкість розробки та ускладнює підтримку проекту, проте великий відсоток програмних продуктів зараз працюють на «чистому» JavaScript, але це зумовлено швидким ростом індустрії та оновленням технологій, застосування яких є дорогим процесом для малих компаній.

Важливим критерієм при обранні фреймворків та бібліотек для розгляду є підтримка SPA. SPA – (з англ. single page application) односторінковий додаток. Раніше дані на веб-сторінці відображались статично та оновлення інформації потребувало перезавантаження сторінки для проведення запиту на отримання нових даних. Пізніше, сторінки стали більш інтерактивними, але навігація по сайту супроводжувалась завантаженням браузером нової сторінки. Сучасні додатки мають підтримку SPA, це дозволяє одноразово завантажити візуальну частину для роботи додатку та не викликати завантаження нової сторінки при навігації по сайту, а потім лише динамічно завантажувати або оновлювати дані для відображення. Це дозволяє покращити швидкодію роботи додатку та дати користувачу новий досвід роботи з веб-додатками.

Порівняємо між собою фреймворки Angular JS, Vue.js та бібліотеку React для обрання у якості інструменту для розробки веб-клієнту інформаційної технології безконтактного обміну персональними даними.

Angular JS – фреймворк для мови JavaScript з модульною системою. Даний фреймворк містить в собі архітектурні обмеження та строгу типізацію, яка реалізовується за допомогою використання мови програмування TypeScript (JavaScript з ООП надбудовою). Можна вважати, що Angular JS пропагує розробку в ООП стилі, , але також можлива розробка в функціональному стилі [30].

Angular JS за стилем написання коду та за використанням підходу до архітектури та обмеженнями схожий на Java [30].

Angular JS – потужний інструмент для розробки, якій містить в собі більшість інструментів та утитіл для розробки. Додаткове встановлення сторонніх пакетів потрібне лише у випадку реалізації специфічної задачі. Фреймворк розроблений для створення SPA додатків, що підвищує його актуальність [30].

Переваги використання Angular JS:

– Компонентний підхід. Цей підхід дозволяє розробку додатку у якості великої кількості структурних одиниць – компонентів. Використання компонентного підходу дозволяє створити додатковий рівень абстракції. З практичної сторони, компонентний підхід дозволяє створити універсальні компоненти, які будуть перевикористовуватись в різних місцях додатку, що пришвидшує розробку та впровадження нового функціоналу;
– Імпорт залежностей. Додаток розбивається на функціональні одиниці – модулі. Модулі можна імпортувати між собою для перевикористання методів. Модульна система створює додатковий рівень абстракції та накладає архітектурні обмеження при створення додатку;

- Висока швидкість роботи при великій кількості функціоналу. Використання Angular JS дозволяє створювати комплексні рішення з великою кількістю функціоналу та має високу потужність при роботі під навантаженнями;
- Строга типізація. Використання TypeScript дозволяє розробнику краще описувати код та покращує підтримку і безпеку написаного коду [29].

Недоліки Angular JS:

- Висока завантаженість. Angular JS містить велику кількість інструментів, які рідко використовуються на практиці, або які мають кращі аналоги. Непотрібні модулі навантажують систему. При створенні невеликих проектів Angular JS має низьку потужність роботи;
- Низька оптимізація SPA. При роботі з SPA, Angular JS напряму оновлює інтерфейс, що негативно відображається на швидкості роботи веб-додатку. Аналоги Angular JS використовують технології для оптимізації оновлення інтерфейсу, що значно покращує швидкодію додатку;
- Низька швидкість розробки. Модульна структура та використання TypeScript – це інструмент для розробки об’ємних та комплексних проектів з великою кількістю функціоналу. Вона гарантує надійність роботи, але значно сповільнює швидкість розробки додатку через необхідність написання великої кількості коду [30].

Vue.js – фреймворк для створення веб-додатків. На відміну від інших популярних бібліотек та фреймворків, Vue.js не був розроблений великою компанією. Розробка починалась з етапу стартапу та переросла у самостійний проект. Vue.js дозволяє створення SPA та може використовуватись як у якості бібліотеки (для розширення існуючого функціоналу), так і у якості фреймворку в якому буде проводитись розробка [31].

Переваги використання Vue.js:

- Можливість поступової інтеграції. Vue.js можна поступово використовувати при розробці проекту, підключаючи бібліотеку для окремих сторінок. У випадку коли потрібно написати новий проект, існує Vue CLI, який містить в собі усі необхідні інструменти для старту розробки;
- Низький поріг входу. Vue.js доволі легко вивчити без досвіду з іншими бібліотеками. Фреймворк має детальну документацію та велику кількість уроків;
- Мутабельність. Мутабельність – стиль роботи з референсним типом даних при якому для введення змін до об’єкту не створюється ідентичний об’єкт з тим самим набором полів але з новими даними, а змінюється існуючий оновлюючи дані. Мутабельність Vue.js дозволяє створювати підписки на один об’єкт та модифікувати його з будь-якого місця додатку, а зміни динамічно будуть відображатись у всіх місцях;
- Підтримка SPA. Vue.js дозволяє створювати SPA, що підвищує актуальність використання;
- Велика кількість інструментів. Vue CLI містить велику кількість інструментів для розробки додатків. Необхідний функціонал вже реалізований і не використання сторонніх бібліотек не необхідним. Також Vue.js має широкий функціонал для полегшення розробки візуальної частини додатку [31].

Недоліки використання:

- Маленька спільнота. Спільнота Vue.js ще формується так як фреймворк новий. Нехватка спеціалістів та досвідчених програмістів зменшує шанс отримання кваліфікованої допомоги та ускладнює пошук кадрів;

– Ризик завеликої гнучкості. У Vue.js можуть з’являтись проблеми з інтеграцією у великі проекти, а прикладів комбінування декількох бібліотек або фреймворків мало [31].

React JS – бібліотека для створення веб-додатків, розроблена компанією Facebook. Бібліотека широко використовується для розробки додатків різної величини та має широку підтримку зі сторони розробників та спільноти. Бібліотека дозволяє створювати SPA та має прогресивну технологію оптимізації даного виду додатків [32].

Переваги використання React JS:

– Компонентна структура. Аналогічно до Angular JS, дана бібліотека підтримує компонентну структуру написання коду;
– Оптимізація SPA. React JS зберігає декілька копій візуального інтерфейсу: поточну версію та майбутню версію зі змінами які будуть оновлені та відображені для користувача. Для оновлення бібліотека порівнює між собою версії інтерфейсу та змінює лише ті чистини які містять зміни і таким чином показує високі результати швидкодії;
– Підтримка спільноти. React JS – стрімко ростуча бібліотека з великою підтримкою зі сторони розробників та спільноти. Для React JS створено велику кількість бібліотек які спрощують роботу. Публічні ресурси містять багато матеріалів для вивчення та професійного розвитку з використанням React JS;
– JSX. JSX – синтаксис бібліотеки React, який дозволяє писати код, використовуючи в одному файлі JavaScript, HTML та CSS. Це зменшує обсяг коду та спрощує розробку;
– Масштабування. React JS підходить для написання проектів різного обсягу та при створенні правильної структури може масштабуватись залежно від потреб проекту [32].

Недоліки React JS:

- Відсутність архітектурних обмежень. React JS не накладає на розробника архітектурних обмежень при створенні додатку, що може призвести до створення немасштабованої, незручної структури, при написанні коду недосвідченим розробником;
- Велика кількість бібліотек. Для React JS створена велика кількість бібліотек та утиліт, що ускладнює пошук якісного рішення задачі. Більшість бібліотек не мають підтримки та існують лише завдяки людям, які їх використовують та створюють додатковий функціонал чи виправляють помилки [32].

Порівнявши Angular JS, React JS та Vue.js, для створення веб-клієнта інформаційної технології безконтактного обміну персональними даними було обрано бібліотеку React JS через те що дана бібліотека має оптимізацію для створення SPA, має широку підтримку спільноти та підходить для створення проектів з невеликою кількістю функціоналу.

Мова програмування JavaScript дозволяє створювати не лише веб- чи серверні додатки, а й мобільні додатки за допомогою використання фреймворків: React Native, Expo та Apache Cordova. Дані фреймворки значною мірою наповнюють мобільний ринок.

Загалом, розробляючи мобільний додаток на крлосплатформі JavaScript, потрібно розуміти, що це не «нативна» мова розробки операційних систем IOS чи Android, тому швидкість роботи додатків написаних на JavaScript буде нижча ніж на Java або Swift та буде стрімко знижуватись при збільшенні функціоналу додатку або зі збільшенням процесів, що потребують тяжких обчислень.

Використання JavaScript для розробки мобільних додатків є доцільним якщо додаток буде відповідати наступним критеріям:

- Маленька кількість функціоналу;
- Тривіальність поставленої задачі;
- Ненативна поведінка;
- Відсутність тяжких обчислень.

JavaScript крослпатформа дозволяє за швидкий період часу створити додаток на дві найпопулярніші платформи (IOS, Android) з використанням мінімуму грошових та людських ресурсів.

Apache Cordova – одна з найперших комерційних реалізацій мобільної кросплатформи з використанням JavaScript. Даний фреймворк використовує WebView для написання коду на JavaScript та HTML. WebView – технологія для перегляду HTML сторінок в мобільних додатках. Apache Cordova містить низку методів за допомогою якої JavaScript може керувати базовим функціоналом смартфону [33].

В наш час Apache Cordova є технологічним архаїзмом через те що WebView обмежує створення функціоналу додатку. Сучасні мобільні програми потребують більшого контролю над мобільним пристроєм, а альтернативою Apache Cordova стали PWA – (з англ. progressive web applications) – прогресивні веб-додатки, які представляють собою веб-сайти які можуть кешувати дані та автономно працювати без інтернету. Використання PWA не потребує нативної частини і запускається вбудованим браузером пристрою [33].

React Native – мобільна кросплатформа створена за концепцією React. Фундаментальною відмінністю між Apache Cordova та Expo є те, що останній тісно пов’язаний з нативним кодом. React Native містить інструмент синхронізації JavaScript коду з Java або Objective-C, Swift який дозволяє передавати дані та викликати нативні функції з JavaScript і навпаки. Даний зв’язок реалізується за допомогою технології Bridge [34].

За допомогою Bridge, в React Native можна інтегрувати будь-яку нативну бібліотеку та керувати нею за допомогою JavaScript. Саме тому React Native має широкий базовий функціонал, а рішення для специфічних задач вже реалізовані спільнотою. У випадку якщо рішення поки не існує, можна створити власний модуль взаємодії нативної бібліотеки та JavaScript коду [34].

Expo – надбудова над React Native, яка дозволяє пришвидшити створення додатку та містить власну бібліотеку компонентів для розробки. Також, перевагою Expo є те, що створення збірки додатку виконується на серверах з віртуальною машиною, що дозволяє не маючи пристрою з Mac OS, створити додаток для IOS [35].

Незважаючи на зручність використання Expo, даний фреймворк має обмежений набір бібліотек та інструментів для розробки, що обмежує можливість розробки функціоналу у додатку.

Expo категорично не підходить для розробки програмних додатків з використанням NFC-модуля по причині відсутності бібліотек для роботи з даним модулем пристрою.

Для створення мобільної клієнтської частини інформаційної технології безконтактного обміну персональними даними обрання технології розробки є безальтернативним по причині високої специфічності поставленої задачі – робота з NFC модулем пристрою.

Працювати з даним модулем має змогу лише React Native по причині існування бібліотек для роботи з NFC та можливістю інтеграції нативної платформи додатку з JavaScript частиною. Також React Native має підтримку зі сторони розробників та є актуальним фреймворком на ринку кросплатформених додатків.

JavaScript широко використовується для написання серверних додатків. Це можуть бути як прості додатки які мають малу кількість запитів і функціоналу, так і великі комплексні рішення з великою кількістю обчислень.

Для розробки серверних додатків за допомогою JavaScript використовується платформа Node.js на основі якої базується велика кількість бібліотек та фреймворків. Для вибору технології для розробки серверної частини розглянемо фреймворки Nest.js, Express.js та сервіс Firebase з можливістю написання коду на Node.js.

Nest.js – веб-фреймворк для написання серверної частини за допомогою JavaScript. В даний фреймворк закладені принципи схожі до Angular або Spring фреймворку для Java [36].

Nest.js має модульну структуру та архітектурні обмеження. Даний підхід дозволяє створити легко масштабуємий каркас додатку з легкою підтримкою. Next.js використовує TypeScript для забезпечення безпеки написаного коду. Бібліотека фреймворку містить інструменти для полегшення написання програмної логіки та рішення задач будь-якого рівня складності [36].

Next.js підходить для створення великих проектів з великою кількістю обтиснень та складною логікою. Архітектура дозволяє постійно розвивати проект та додавати до нього нову логіку. Модульний підхід надає можливість перевикористовувати написаний код і повідомляє розробника коли з’являється циклічна залежність модулів [36].

Недоліками Next.js є повільна розробка проекту через необхідність збереження єдиного стилю, який закладений в архітектуру додатку, а також велика вага проекта.

Express.js – веб-фреймворк який базується на платформі Node.js. Фреймворк не накладає архітектурних обмежень при написанні проекту та містить базові функції, необхідні для розробки. Фреймворк є легким та містить багато бібліотек для розробки [37].

Express.js доцільно використовувати для швидкого написання невеликих серверних додатків. Відкрита архітектура негативно впливає на стиль коду та структуру проекту, через що при масштабуванні можуть з’являтись конфлікти архітектури.

Firebase – мобільна платформа розроблена компанією Google яка представляє собою комплексне рішення для розробки мобільних, веб- та серверних додатків. Система містить в собі сервіси для виконання різнорідних задач [38].

Для розробки серверної частини інформаційної технології безконтактного обміну персональними даними необхідно буде використовувати сервіси Firebase Firestore Database, Firebase Storage, Firebase Cloud Functions та Firebase Hosting [38].

Firebase Firestore Database – система яка дозволяє зберігати на сервері дані у форматі JSON. Firestore працює за допомогою TCP протоколу, що забезпечує двосторонній зв’язок між клієнтською та серверною частиною і оновлення даних в реальному часі без необхідності робити додаткові запити до бази [38].

Firebase Storage – система для розміщення файлів на сервері. За допомогою сервісу можна завантажити на сервер будь-який файл та отримати посилання на нього для завантаження або перегляду в браузері [38].

Firebase Cloud Functions – платформа для створення веб-додатків на платформі Node.js. Сервіс дозволяє написати повноцінний веб-додаток за завантажити його на сервер. Зазвичай використовується для компенсації функцій, які не реалізовані у Firebase або для рішення специфічних задач [38].

Firebase Hosting – сервіс для розміщення веб- або серверного додатку на потужностях Firebase. Використовується для легкого налаштування доступу до ресурсу [38].

Недоліками Firebase є ціна використання, яка зростає в залежності від кількості запитів на сервер та відсутність готової реалізації деякого функціоналу.

Серверна частина інформаційної технології безконтактного обміну персональними даними відповідає за створення та оновлення профілю користувача, збереження даних та отримання інформації про користувача. Це

означає, що серверна частина матиме простий функціонал з низьким навантаженням зі сторони користувачів. Виходячи з поставленої задачі серверної частини, доцільним є використання платформи Firebase.

3.4 Розробка схеми алгоритму роботи програми

Для відображення циклу роботи програмного забезпечення використаємо текстовий та графічний опис (рисунок 3.2). Цей алгоритм відповідає за функціонування основних можливостей додатку.

Кроки алгоритму:

1. Автентифікація користувач.
2. Якщо користувач зареєстрований в системі, перейти на крок 5.
3. Створення облікового запису.
4. Встановлення фотографії профіля.
5. Отримання створених посилань користувача.
6. Обрання дії користувачем. Якщо додавання нового посилання або редагування створеного, то перейти на крок 7, у випадку видалення посилання – 8.
7. Обрання типу посилання та додавання тексту посилання.
8. Обрання посилання та видалення посилання.
9. Занесення змін до бази даних.
10. Обрання запису даних на NFC-мітку, якщо обрано запис – перехід на крок 11, якщо ні – 15.
11. Обрання режиму стиснення даних, якщо обрано стиснення – перехід на крок 12, якщо ні – 13.
12. Стиснення посилання на профіль за допомогою алгоритму LZW.
13. Активація запису даних за допомогою NFC.
14. Піднесення NFC-тега до пристрою.
15. Завершення роботи з мобільним додатком.

16. Сканування пристроєм NFC-тега, якщо тег відсканований, то перейти на крок 15, інакше – завершення роботи з додатком.
17. Отримання інформації про профіль посилання на який записаного в тег.
18. Якщо увімкнено режим стиснення – перейти на крок 19, якщо ні – крок 20.
19. Розгортування інформації за допомогою зворотного алгоритму LZW.
20. Якщо встановлено режим прямої навігації – перейти на крок 21, якщо ні – крок 22.
21. Обрання посилання для переходу.
22. Навігація користувача за посиланням.
23. У випадку повторного сканування тегу перейти на крок 16, інакше – завершення роботи з модулем.

Створений алгоритм роботи інформаційної технології безконтактної передачі персональних даних об’єднує функціонал усіх сервісів додатку:

- Веб-клієнт;
- Сервер;
- Мобільний додаток;
- NFC-API;
- Модуль стиснення даних.

Алгоритм роботи показує процес функціонування інформаційної технології: від моменту реєстрації користувача в системі, до запису інформації на персональний носій користувача. Алгоритм є простий та зрозумілий для користувача. Алгоритм роботи інформаційної технології є однаковим для платформ під які розроблявся додаток (IOS та Android).

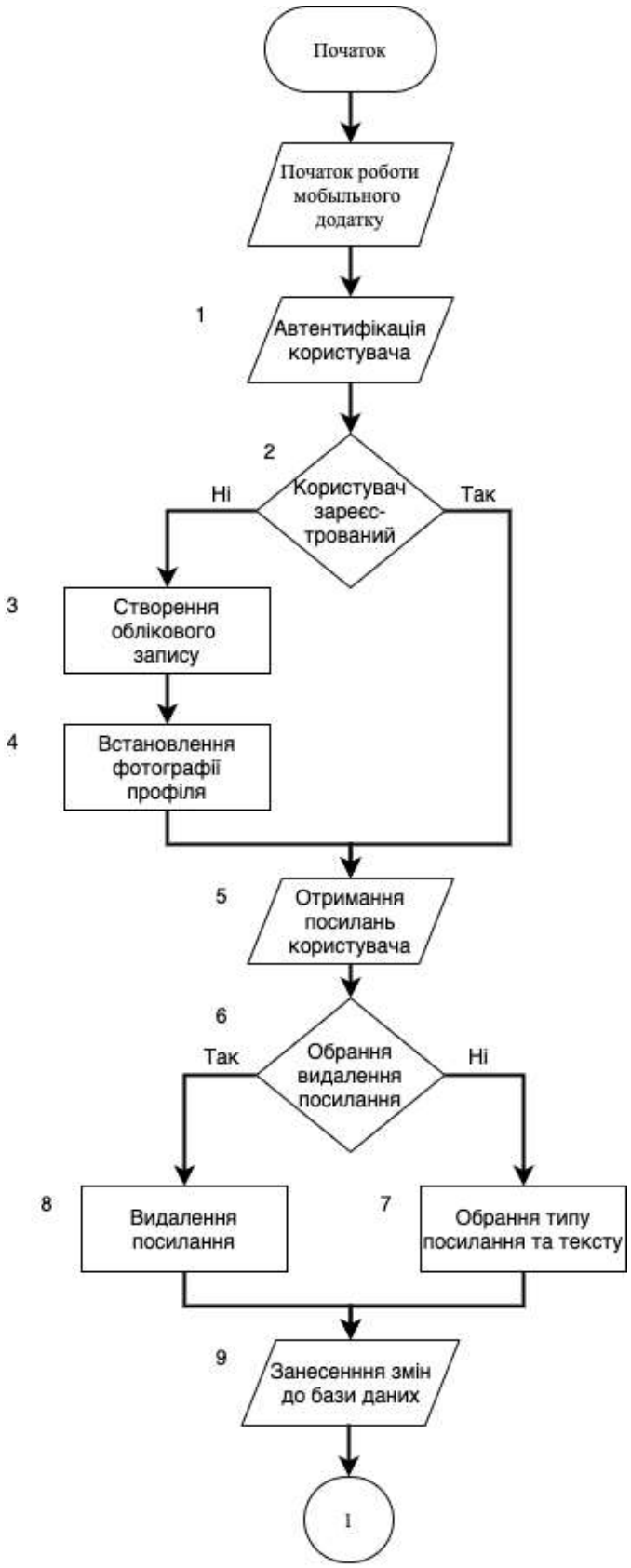


Рисунок 3.2 – Схема алгоритму роботи інформаційної технології безконтактного обміну персональними даними

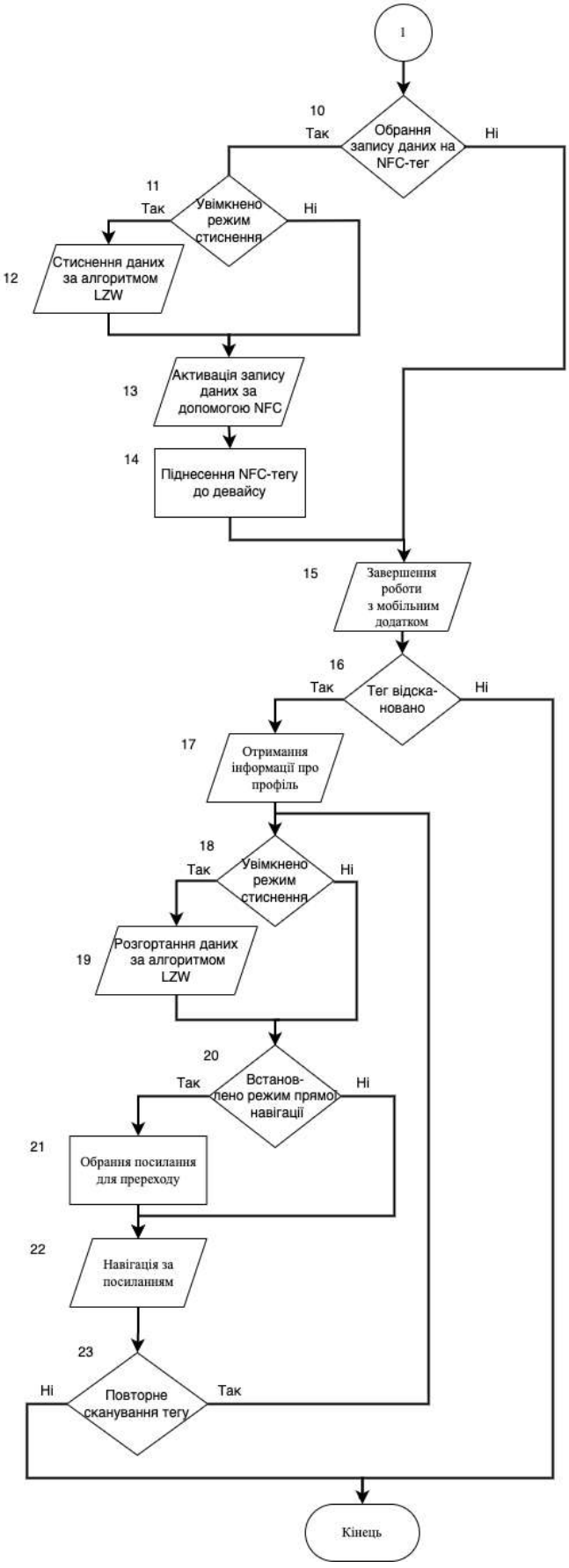


Рисунок 3.2, аркуш 2

Таким чином було розроблено алгоритм роботи інформаційної технології безконтактного обміну персональними даними, розписано основні кроки.

3.5 Тестування та аналіз результатів роботи програмного забезпечення

Тестування функціональності буде проводитись за такими критеріями:

- Сторінки інтерфейсу користувача виконують розроблений функціонал;
- Додаток має записувати дані на NFC-тег;
- NFC-тег має зчитуватись іншими пристроями, які підтримують роботу з NFC технологією;
- Користувач має змогу увімкнути режим стиснення даних при записі та при зчитуванні інформації за допомогою додатку;
- Середній степінь стиснення інформації має бути у межах 10%;
- Посилання, записане на NFC-тег має відкриватись та направляти користувача на профіль користувача;
- Якщо увімкнена функція прямої переадресації, то при скануванні NFC-тегу, користувача має бути переведено за посиланням;
- Користувач має змогу перезаписати інформацію на NFC-тегу.

Після встановлення додатку на смартфон (для тестів використовується Android пристрій з версією операційної системи 10.0 та підтримкою технології NFC), користувач може відкрити його, після чого опиниться на сторінці реєстрації або входу (рисунок 3.3).

По замовчуванню користувач не має облікового запису в додатку, тому необхідно створити новий профіль користувача. У якості ідентифікатора профіля використовується комбінація з електронної пошти користувача та паролю. Також, необхідно вказати ім’я користувача, яке буде відображатись в мобільному додатку та в веб-профілі (рис. 3.4).

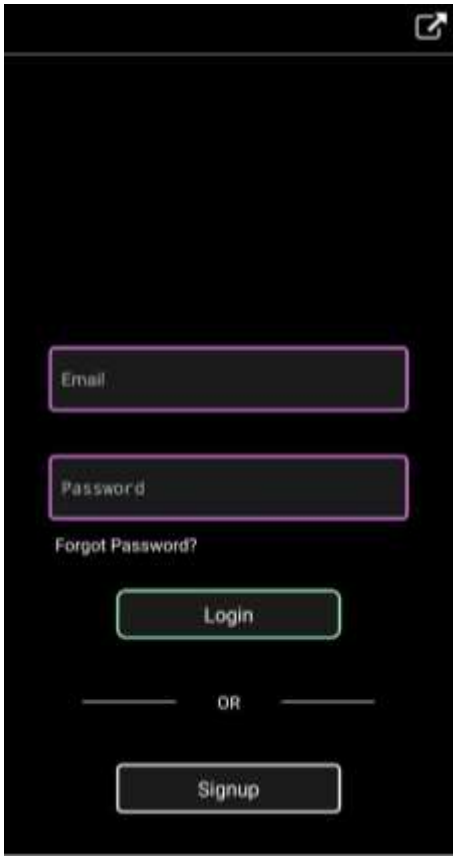


Рисунок 3.3 – Загальний вигляд початкової сторінки інформаційної технології

Для початку реєстрації необхідно натиснути на кнопку «Signup», після чого користувача буде перенаправлено на сторінку введення інформації для реєстрації нового облікового запису (рис. 3.4).

На сторінці реєстрації присутня валідація введених даних. У випадку якщо користувач не введе обов’язкові поля для реєстрації (в даному випадку усі поля обов’язкові), або допустить помилку в форматі адреси електронної пошти, додаток сповістить користувача про помилки та не дасть продовжити реєстрації до усунення проблеми (рис. 3.5).

Рисунок 3.4 – Загальний вигляд сторінки введення інформації для реєстрації нового облікового запису

Рисунок 3.5 – Загальний вигляд прикладу роботи валідації даних на етапі реєстрації нового облікового запису

Після успішної реєстрації, користувача буде перенаправлено в основну частину додатку, де користувач матиме змогу відредагувати фотографію

профіля, змінити опис профіля та додати посилання на соціальні мережі (рис.3.6).

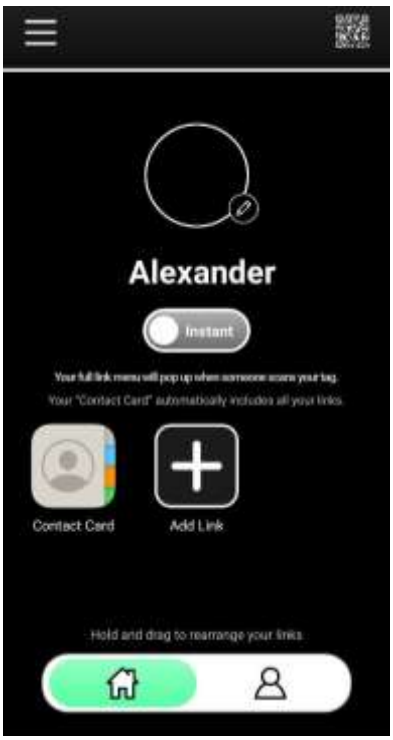


Рисунок 3.6 – Загальний вигляд основної функціональної частини додатку, де відображається інформація про користувача та його посилання

Для створення нового посилання необхідно натиснути на кнопку «Add Link» та обрати соціальну мережу або за допомогою жесту «вліво», перейти на список можливих посилань та обравши необхідну соціальну мережу, встановити посилання на неї, після чого натиснути на кнопку «Done». Якщо посилання було створено, то воно з’явиться в списку посилань на головній сторінці додатку. Для тесту було внесено посилання на соціальну мережу Instagram (рис. 3.7 – 3.9).

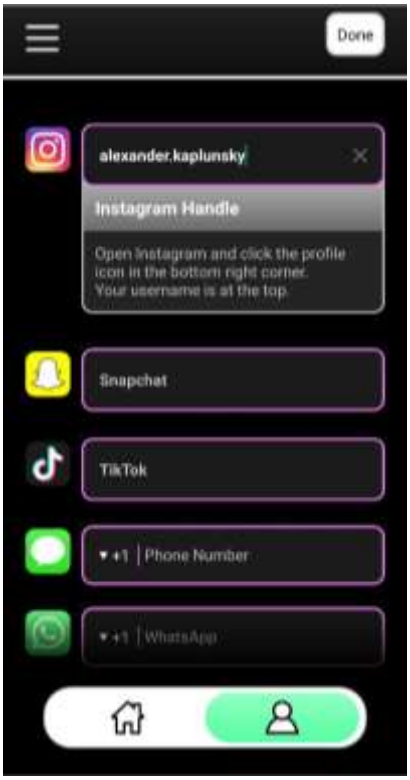


Рисунок 3.7 – Загальний вигляд інтерфейсу додавання нового посилання за допомогою переходу в меню по жесту «Вліво»

Рисунок 3.8 – Загальний вигляд меню додавання посилання з головної сторінки додатку

Рисунок 3.9 – Загальний вигляд доданого посилання на соціальну мережу Instagram

Створене посилання на профіль користувача матиме вигляд: https%3A%2F%2Ftappin-nfc.web.app%2Foleksandr.kaplunskyj%3Fnfclabel%3Dtappin%26compression%3Dtrue%26links%3Dinstagram%26links%3Dcontact%26links%3Dfacebook%26links%3Dpaypal.

Дане посилання створене без режиму стиснення зберігає у собі метадані про користувача та про створені ним посилання. Довжина такого повідомлення – 170 байт. Ліміт запису на NFC-тег – 180 байт. Проте, можна використовувати теги з більшим об’ємом пам’яті (наприклад, 560 байт), проте тестування проводилось на носієві з лімітом 180 байт, та цифри отримані при виконання тестування є справедливими для такого об’єму інформації

Для запису посилання на профіль користувача на безконтактний носій інформації, необхідно відкрити бокове меню та обрати пункт «Activate Phone Tag». Після цього буде активовано NFC модуль пристрою, який буде очікувати коли в зоні контакту опиниться сумісний NFC-тег, придатний до запису інформації (рис. 3.10 – 3.11). Користувач отримає інструкцію для взаємодії з NFC-тегом та можливість увімкнути режим стиснення даних.

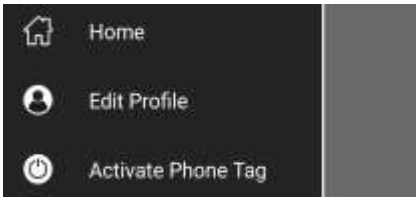


Рисунок 3.10 – Загальний вигляд кнопки активації запису даних на NFC-тег

Рисунок 3.11 – Загальний вигляд анотації до роботи з NFC-тегом

Після створення безконтактного носія, в пам'ять NFC-тегу буде записано посилання на персональний профіль користувача. У випадку увімкнення режиму стиснення даних, посилання буде зашифроване таким чином, що розшифрувати його можна були лише за використання мобільного додатка.

Зашифроване повідомлення буде переставлено у вигляді набору байтів Unit8Array, текстовий вигляд якого зображено на рисунку 3.12.

Рисунок 3.12 – Загальний вигляд стисненого посилання на профіль користувача

Довжина стисненого посилання – 153 байти, що є меншим від оригінального посилання на 10%.

Для зчитування даних необхідно піднести NFC-тег до зони дії модуля пристрою, після чого користувача буде перенаправлено на веб-сторінку профілю, посилання на який зберігається на носії (рис. 3.13), у випадку якщо увімкнено режим стиснення даних, сканування необхідно проводити за допомогою мобільного додатку, адже він містить алгоритм розгортування стисненого повідомлення. Проте, можна використовувати і інший додаток з можливістю роботи з алгоритмом стиснення LZW, який міститиме функціонал розгортування інформації, проте це не є раціональним, адже, створений у ході проектування інформаційної технології, додаток протестовано та результат розгортування гарантовано відпрацьовує відповідно до поставленим перед ним технічним завданням.

Більшість пристроїв вже вміють працювати з NFC-тегами, тому пристрій самостійно встановить контакт з носієм та отримає інформацію з нього. Деякі види пристроїв, залежно від версії операційної системи та встановленого програмного забезпечення, можуть не сканувати оточення смартфону для пошуку в районі взаємодії активних пристроїв, які сумісні з NFC-модулем, тому для зчитування інформації необхідно буде самостійно увімкнути NFC-модуль за допомогою вбудованих в пристрій функцій.

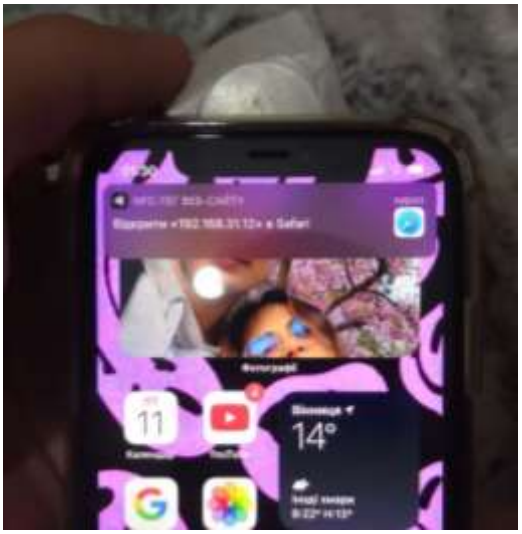


Рисунок 3.13 – Загальний вигляд взаємодії пристрою з NFC-модулем та NFC-тегу з записаною на нього інформацією

Після переходу за посиланням, на інтерфейсі можна обрати додане посилання на соціальну мережу та знавігуватись за ним (рис. 3.14). Якщо додаток, на який навігується користувач, підтримує технологію діп-лінкінгу, замість веб-версії додатку, буде відкрито мобільний додаток.

Рисунок 3.14 – Загальний вигляд інтерфейсу профілю користувача, посилання на який зберігається NFC носієм

Отже, було розроблено алгоритм тестування основного функціоналу інформаційної технології передачі персональної інформації. Для забезпечення правильного функціонування необхідно провести повторне проходження алгоритму тестування.

У таблиці 3.1 наведено результати роботи модулю стиснення даних, при проходженні 10 ітерацій тесту, за умови використання максимального розміру інформації у 180 байтів.

Таблиця 3.1 – Результати роботи модуля стиснення даних

№	Оригінальне повідомлення	Оригінальна довжина	Довжина після стиснення	% стиснення
1	https%3A%2F%2Ftappin-nfc.web.app%2Foleksandr.kaplunskyj%3Fnfclabel%3Dtappin%26compression%3Dtrue%26links%3Dinstagram%26links%3Dcontact%26links%3Dfacebook%26links%3Dpaypal	170	151	11
2	https%3A%2F%2Ftappin-nfc.web.app%2Foleksandr.kaplunskyj%3Fnfclabel%3Dtappin%26compression%3Dtrue%26links%3Dinstagram%26links%3Dtelegrm%26links%3Dy2k%26links%3Damazon%26links%3Dpost	180	158	12
3	https%3A%2F%2Ftappin-nfc.web.app%2Foleksandr.kaplunskyj%3Fnfclabel%3Dtappin%26compression%3Dtrue%26links%3Dyoutube%26links%3Dtelegrm%26links%3Dy2k%26links%3Damazon%26links%3Dpost	178	151	15
4	https%3A%2F%2Ftappin-nfc.web.app%2Foleksandr.kaplunskyj%3Fnfclabel%3Dtappin%26compression%3Dtrue%26links%3Dyoutube%26links%3Debay%26links%3Dy2k%26links%3Damazon%26links%3Dpost	175	147	16
5	https%3A%2F%2Ftappin-nfc.web.app%2Foleksandr.kaplunskyj%3Fnfclabel%3Dtappin%26compression%3Dtrue%26links%3Dyoutube%26links%3Debay%26links%3Dy2k%26links%3Dspotify%26links%3Dpost	176	154	12
6	https%3A%2F%2Ftappin-nfc.web.app%2Foleksandr.kaplunskyj%3Fnfclabel%3Dtappin%26compression%3Dtrue%26links%3Dsoundcloud%26links%3Debay%26links%3Dy2k%26links%3Dspotify%26links%3Dpost	179	157	12
7	https%3A%2F%2Ftappin-nfc.web.app%2Foleksandr.kaplunskyj%3Fnfclabel%3Dtappin%26compression%3Dtrue%26links%3Dsoundcloud%26links%3Debay%26links%3Dy2k%26links%3Dfacebook%26links%3Dpost	180	160	11

Таблиця 3.1 – Продовження

№	Оригінальне повідомлення	Оригінальна довжина	Довжина після стиснення	% стиснення
8	https%3A%2F%2Ftappin-nfc.web.app%2Foleksandr.kaplunskyj%3Fnfclabel%3Dtappin%26compression%3Dtrue%26links%3Dsoundcloud%26links%3Debay%26links%3Dy2k%26links%3Dcontact%26links%3Dpost	179	153	14
9	https%3A%2F%2Ftappin-nfc.web.app%2Foleksandr.kaplunskyj%3Fnfclabel%3Dtappin%26compression%3Dtrue%26links%3Dgmail%26links%3Dyoutube%26links%3Dy2k%26links%3Dnetflix%26links%3Dpost	177	150	15
10	https%3A%2F%2Ftappin-nfc.web.app%2Foleksandr.kaplunskyj%3Fnfclabel%3Dtappin%26compression%3Dtrue%26links%3Dinstagram%26links%3Dyoutube%26links%3Dp2p%26links%3Dcb%26links%3Dpost	176	156	11

Результати наведені у таблиці 3.1 наглядно показують роботу модуля шифрування. Середній відсоток стиснення при 10 ітераціях дорівнює 13%, при застосуванні тексту розміром близького до граничних (180 байтів)

Також було проведене тестування модулю стиснення даних у сценарії використання NFC-мітки більшого об’єму даних – 560 байт, на основі чого побудовано діаграму залежності стиснення від об’єму інформації (рис 3.15).

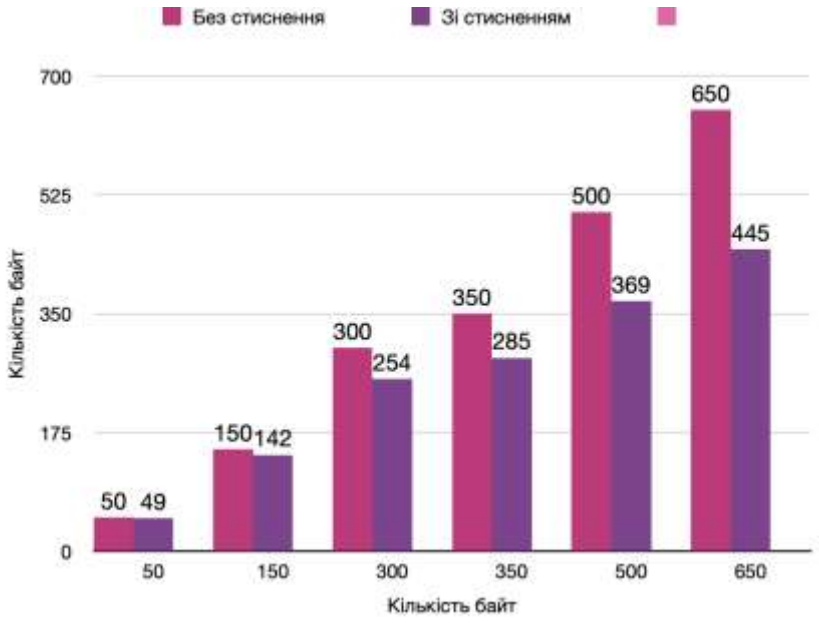


Рисунок 3.15 – Діаграма залежності стиснення від об’єму інформації

Таким чином, бачимо, що при збільшенні об’єму інформації для стиснення, степінь стиснення значно збільшується, адже зі збільшенням об’єму інформації, пропорційно збільшується кількість дубльованих комбінацій, що призводить до більш ефективного її стиснення.

Після проходження 50 циклів тестування програмного продукту не було виявлено помилок в роботі мобільного, серверного та веб-додатку. При різних варіаціях посилань на профіль користувача, середній степінь стиснення посилання перебував у межах 13%, залежно від набору символів, що задовольняє поставлені перед інформаційною технологією задачі.

3.6 Висновок до розділу 3

У даному розділі було проведено порівняння мов програмування JavaScript, Dart та Java, у ході якого було виділено їх недоліки та переваги. Орієнтуючись на критерії обрання мови програмування для розробки інформаційної технології безконтактного персональними даними, було обрано мову JavaScript. Було роглянуто популярні бібліотеки та фреймворки для написання мобільної клієнтської, серверної та веб-клієнтської частини модуля. У ході аналізу переваг та недоліків технологій для реалізації, було обрано React JS для реалізації веб-клієнтської частини, Firebase – для серверної частини, React Native – для мобільної клієнтської частини. Обрана архітектура для створення програмного модуля.

Розроблено та описано схему алгоритму роботи програми. Реалізовано алгоритм LZW мовою JavaScript. Наведено алгоритм тестування основного функціонала інформаційної технології безконтактного обміну персональними даними, виведено середній степінь стиснення інформації – 13%.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Комерційний та технологічний аудит науково-технічної розробки

Метою даного розділу є проведення технологічного аудиту, в даному випадку нового програмного продукту інформаційна технологія безконтактного обміну персональними даними. Розробка надає можливість користувачу записати персональну інформацію на пристрій-накопичувач (NFC-тег) та надати можливість іншому користувачу зчитати інформацію з тегу. Система має прийти на заміну QR-кодам, Wi-Fi та Bluetooth. Особливістю програми є те, що дана інформаційна технологія міститиме в собі алгоритм стиснення, на відміну від аналогів, які не містять алгоритму стиснення даних, тому обмежена кількість інформації до запису. Аналогів є досить багато, і їх ціна залежить від аксесуару з NFC. Найдешевший екземпляр – 187грн. Середня ціна на найпопулярніші аксесуари – 800 грн.

Для проведення комерційного та технологічного аудиту залучають не менше 3-х незалежних експертів. Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням п’ятибальної системи оцінювання за 12-ма критеріями, у відповідності із табл. 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздат-ність продук-кту в реаль-них умовах

Продовження табл. 4.1

Ринкові переваги					
2	Багато аналогів на малому ринку	Ринкові п 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 аналогів	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
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практик на здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так із комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	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-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві

Продовження табл. 4.1

11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Усі дані по кожному параметру занесено в таблиці 4.2

Таблиця 4.2 – Результати оцінювання комерційного потенціалу розробки

Критерії оцінювання	ПІБ експертів		
	Експерт 1	Експерт 2	Експерт 3
	Бали		
Технічна здійсненність концепції	4	4	4
Наявність аналогів на ринку	4	4	4
Цінова політика	4	4	4
Технічні та споживчі властивості виробу	4	3	4
Експлуатаційні витрати	4	4	4
Ринок збуту	4	4	4
Конкурентоспроможність	3	4	4
Фахівці з технічної і комерційної реалізації	4	4	4
Фінансування	3	4	3
Матеріально-технічна база	3	3	4
Термін реалізації ідеї	3	3	4
Супровідна документація	3	4	4
Сума	43	45	47
Середньоарифметична сума балів	(43+45+47) / 3 = 45		

За даними таблиці 4.2 можна зробити висновок щодо рівня комерційного потенціалу даної розробки. Для цього доцільно скористатись рекомендаціями, наведеними в таблиці 4.3.

Таблиця 4.3 - Рівні комерційного потенціалу розробки

Середньоарифметична сума балів СБ , розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-48	Високий

Як видно з таблиці, рівень комерційного потенціалу розроблюваного нового програмного продукту є високим, що досягається за рахунок того, що дана технологія є актуальною у сучасних реаліях, адже після пандемії бізнес зрозумів доцільність використання безконтактних методів передачі персональної інформації. Спосіб взаємодії через NFC є популярним на поточний момент. Аналоги не містять алгоритму стиснення даних, тому обмежена кількість інформації до запису. Інформаційна технологія міститиме в собі алгоритм стиснення. Дана технологія прийде на заміну QR-кодам, Wi-Fi та Bluetooth.

4.2 Прогнозування витрат на виконання науково-дослідної (дослідно-конструкторської) роботи

4.2.1 Основна заробітна плата розробників, яка розраховується за формулою:

$$З_о = \frac{M}{T_р} \cdot t \text{ ,} \qquad (4.1)$$

де М – місячний посадовий оклад конкретного розробника (дослідника), грн.;

$Т_р$ – число робочих днів в місяці, 23 днів;

t – число днів роботи розробника (дослідника).

Результати розрахунків зведемо до таблиці 4.1.

Таблиця 4.1 – Основна заробітна плата розробників

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник проекту	33000	1434,78	42	60260,870
Інженер	31500	1369,57	42	57521,739
Всього				117782,61

Так як в даному випадку розробляється програмний продукт, то розробник виступає одночасно і основним робітником, і тестувальником розроблюваного програмного продукту.

4.2.2 Додаткова заробітна плата розробників, які приймали участь в розробці обладнання.

Додаткова заробітна плата прийнято розраховувати як 13,5 % від основної заробітної плати розробників та робітників:

$$З_д = З_о \cdot 13,5\ \% / 100\ \% \qquad (4.2)$$

$З_д = (117782,61 \cdot 13,5\ \% / 100\ \%) = 15900,65$ (грн.)

4.2.3 Нарахування на заробітну плату розробників.

Згідно діючого законодавства нарахування на заробітну плату складають 22 % від суми основної та додаткової заробітної плати.

$$Н_з = (З_о + З_д) \cdot 22\ \% / 100\% \qquad (4.3)$$

$$Н_з = (117782{,}61 + 15900{,}65) \cdot 22\ \% / 100\ \% = 29410{,}32 \text{ (грн.)}$$

4.2.4. Оскільки для розроблювального пристрою не потрібно витрачати матеріали та комплектуючи, то витрати на матеріали і комплектуючі дорівнюють нулю.

4.2.5 Амортизація обладнання, яке використовувалось для проведення розробки.

Амортизація обладнання, що використовувалось для розробки в спрощеному вигляді амортизація обладнання, що використовувалась для розробки розраховується за формулою:

$$A = \frac{Ц}{Тв} \cdot \frac{t_{вик}}{12} \text{ [грн.].} \qquad (4.4)$$

де Ц – балансова вартість обладнання, грн.;

Т – термін корисного використання обладнання згідно податкового законодавства, років

$t_{вик}$ – термін використання під час розробки, місяців

Розрахуємо, для прикладу, амортизаційні витрати на комп’ютер балансова вартість якого становить 50040 грн., термін його корисного використання згідно податкового законодавства – 2 роки, а термін його фактичного використання – 1,83 міс.

$$A_{обл} = \frac{50040}{2} \times \frac{1{,}83}{12} = 3807{,}391 \text{ грн.}$$

Аналогічно визначаємо амортизаційні витрати на інше обладнання та приміщення. Розрахунки заносимо до таблиці 4.2. Для розрахунку амортизації нематеріальних ресурсів використовується формула:

$$A_{н.р.} = Ц_{н.р.} * Н_а * \frac{t_{вик.}}{12} \quad (4.5)$$

Але, так як вартість ліцензійної ОС та спеціалізованих ліцензійних нематеріальних ресурсів менше 20000 грн, то даний нематеріальний актив не амортизується, а його вартість включається у вартість розробки повністю, $B_{нем.ак.}$ = 10000 грн.

Таблиця 4.2 – Амортизаційні відрахування матеріальних і нематеріальних ресурсів для розробників

Найменування обладнання	Балансова вартість, грн.	Строк корисного використання, років	Термін використання обладнання, місяців	Амортиза-ційні відрахуван-ня, грн.
Комп'ютер та комп'ютерна периферія	50040	2	1,83	3807,391
Офісне обладнання	24700	4	1,83	939,674
Приміщення	985000	20	1,83	7494,565
Всього				12241,63

5.2.6 Тарифи на електроенергію для непобутових споживачів (промислових підприємств) відрізняються від тарифів на електроенергію для населення. При цьому тарифи на розподіл електроенергії у різних постачальників (енергорозподільних компаній), будуть різними. Крім того, розмір тарифу залежить від класу напруги (1-й або 2-й клас). Тарифи на розподіл електроенергії для всіх енергорозподільних компаній встановлює

Національна комісія з регулювання енергетики і комунальних послуг (НКРЕКП). Витрати на силову електроенергію розраховуються за формулою:

$$В_е = В \cdot П \cdot Ф \cdot К_п, \quad (4.6)$$

де В – вартість 1 кВт-години електроенергії для 1 класу підприємства, В = 6,2 грн./кВт;

П – встановлена потужність обладнання, кВт. П = 0,4 кВт;

Ф – фактична кількість годин роботи обладнання, годин.

$К_п$ – коефіцієнт використання потужності, $К_п$ = 0,9.

$В_е = 0,9 \cdot 0,4 \cdot 8 \cdot 42 \cdot 6,2 = 749,952$ (грн.)

5.2.7 Інші витрати та загальновиробничі витрати.

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» розраховуються як 100…150% від суми основної заробітної плати дослідників:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.7)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати».

$I_в = 117782,61 * 123\% / 100\% = 144872,6$ (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» розраховуються як 100…150% від суми основної заробітної плати дослідників:

$$H_{нзв} = (З_о + З_р) \cdot \frac{H_{нзв}}{100\%}, \quad (4.8)$$

де $H_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати».

$H_{нзв}$ = 117782,61 * 133 % / 100 % = 156651 (грн.)

5.2.9 Витрати на проведення науково-дослідної роботи.

Сума всіх попередніх статей витрат дає загальні витрати на проведення науково-дослідної роботи:

$B_{заг}$ = 117782,61+15900,65+29410,32+12241,63+10000+749,95+144872,6+
+156651 = 487608,64 грн.

5.2.11 Розрахунок загальних витрат на науково-дослідну (науково-технічну) роботу та оформлення її результатів.

Загальні витрати на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховуються *ЗВ*, визначається за формулою:

$$ЗВ = \frac{B_{заг}}{\eta} \text{ (грн)}, \quad (5.9)$$

де η – коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи.

Так, якщо науково-технічна розробка знаходиться на стадії: науково-дослідних робіт, то η=0,1; технічного проектування, то η=0,2; розробки конструкторської документації, то η=0,3; розробки технологій, то η=0,4; розробки дослідного зразка, то η=0,5; розробки промислового зразка, то η=0,7; впровадження, то η=0,9. Оберемо η = 0,5, так як розробка, на даний момент, знаходиться на стадії дослідного зразка:

$ЗВ$ = 487608,64 / 0,5 = 975217 грн.

4.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

В ринкових умовах узагальнювальним 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. Саме зростання чистого прибутку забезпечить потенційному інвестору надходження додаткових коштів, дозволить покращити фінансові результати його діяльності, підвищить конкурентоспроможність та може позитивно вплинути на ухвалення рішення щодо комерціалізації цієї розробки.

Для того, щоб розрахувати можливе зростання чистого прибутку у потенційного інвестора від можливого впровадження науково-технічної розробки необхідно:

а) вказати, з якого часу можуть бути впроваджені результати науково-технічної розробки;

б) зазначити, протягом скількох років після впровадження цієї науково-технічної розробки очікуються основні позитивні результати для потенційного інвестора (наприклад, протягом 3-х років після її впровадження);

в) кількісно оцінити величину існуючого та майбутнього попиту на цю або аналогічні чи подібні науково-технічні розробки та назвати основних суб'єктів (зацікавлених осіб) цього попиту;

г) визначити ціну реалізації на ринку науково-технічних розробок з аналогічними чи подібними функціями.

При розрахунку економічної ефективності потрібно обов'язково враховувати зміну вартості грошей у часі, оскільки від вкладення інвестицій до отримання прибутку минає чимало часу. При оцінюванні ефективності інноваційних проектів передбачається розрахунок таких важливих показників:

- абсолютного економічного ефекту (чистого дисконтованого доходу);
- внутрішньої економічної дохідності (внутрішньої норми дохідності);
- терміну окупності (дисконтованого терміну окупності).

Аналізуючи напрямки проведення науково-технічних розробок, розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором можна об'єднати, враховуючи визначені ситуації з відповідними умовами.

4.3.1 Розробка чи суттєве вдосконалення програмного засобу (програмного забезпечення, програмного продукту) для використання масовим споживачем.

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

$$\Delta П_i = (\pm \Delta Ц_0 \cdot N + Ц_0 \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \qquad (4.10)$$

де $\pm \Delta Ц_o$ – зміна вартості програмного продукту (зростання чи зниження) від впровадження результатів науково-технічної розробки в аналізовані періоди часу;

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки;

$Ц_о$ – основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки, $Ц_о = Ц_б \pm \Delta Ц_о$;

Цб – вартість програмного продукту у році до впровадження результатів розробки;

ΔN – збільшення кількості споживачів продукту, в аналізовані періоди часу, від покращення його певних характеристик;

λ – коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт $\lambda = 0{,}8333$.

p – коефіцієнт, який враховує рентабельність продукту;

ϑ – ставка податку на прибуток, у 2022 році ϑ =18%.

Припустимо, що при прогнозованій ціні 100 грн. за одиницю виробу, термін збільшення прибутку складе 3 роки. Після завершення розробки і її вдосконалення, можна буде підняти її ціну на *20* грн. Кількість одиниць реалізованої продукції також збільшиться: протягом першого року – на 75000 шт., протягом другого року – на 100000 шт., протягом третього року на 125000 шт. До моменту впровадження результатів наукової розробки реалізації продукту не було:

$\Delta П_1 = (0*20 + (100 + 20)*75000)* 0{,}8333* 0{,}5) * (1 - 0{,}18) = 2562499{,}898$ *грн.*

$\Delta П_2 = (0*20 + (100 + 20)*(75000+100000)* 0{,}8333* 0{,}5) * (1 - 0{,}18) = 7174999{,}713$ *грн.*

$\Delta П_3 = (0*20 + (100 + 20)*(75000+100000+125000)* 0{,}8333* 0{,}5) * (1 - 0{,}18) = 12299999{,}508$ *грн.*

Отже, комерційний ефект від реалізації результатів розробки за три роки складе 22037499,12 грн.

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності.

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

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (5.11)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої науково-дослідної (науково-технічної) роботи, грн;

Т – період часу, протягом якою виявляються результати впровадженої науково-дослідної (науково-технічної) роботи, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, $\tau = 0,05...0,15$;

t – період часу (в роках).

Збільшення прибутку ми отримаємо починаючи з першого року:

ПП= (2562499,898/(1+0,1)[1])+(7174999,713/(1+0,1)[2])+(12299999,508/(1+0,1)[3]) = 2329545,36 + 5929751,829 + 9241171,681 = 17500468,87 грн.

Далі розраховують величину початкових інвестицій *PV*, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{інв} * ЗВ, \qquad (4.12)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай $k_{інв}=2...5$, але може бути і більшим;

ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

$PV = 2 * 975217 = 1950434{,}56$ *грн.*

Тоді абсолютний економічний ефект $E_{абс}$ або чистий приведений дохід *(NPV, Net Present Value)* для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV, \qquad (4.13)$$

$E_{абс} = 17500468{,}87 - 1950434{,}56 = 15550034{,}32$ грн.

Оскільки $E_{абс} > 0$ то вкладання коштів на виконання та впровадження результатів даної науково-дослідної (науково-технічної) роботи може бути доцільним.

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність або показник внутрішньої но-рми дохідності (IRR, Internal Rate of Return) вкладених інвестицій та порівняти її з так званою бар'єрною ставкою дисконтування, яка визначає ту мінімальну внутрішню економічну дохідність, нижче якої інвестиції в будь-яку науково-технічну розробку вкладати буде економічно недоцільно.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього використаємо формулу:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1, \tag{4.14}$$

$T_ж$ – життєвий цикл наукової розробки, роки.

$E_в = \sqrt[3]{(1 + 15550034{,}32/1950434{,}56} - 1 = 1{,}078$

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді визначається за формулою:

$$\tau = d + f, \tag{4.15}$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2022 році в Україні d = (0,09…0,14);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,5).

$$\tau_{\min} = 0{,}14 + 0{,}05 = 0{,}19.$$

Так як $E_в > \tau_{min}$, то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій за формулою:

$$T_{ок} = \frac{1}{E_в}, \tag{4.16}$$

$T_{ок}$ = 1 / 1,078 = 0,93 р.

Оскільки $T_{ок}$ < 3-х років, а саме термін окупності рівний 0,93 роки, то фінансування даної наукової розробки є доцільним.

4.5 Висновки до розділу 4

Економічна частина даної роботи містить розрахунок витрат на розробку нового програмного продукту, сума яких складає 975217 гривень. Було спрогнозовано орієнтовану величину витрат по кожній з статей витрат. Також розраховано чистий прибуток, який може отримати виробник від реалізації нового технічного рішення, розраховано період окупності витрат для інвестора та економічний ефект при використанні даної розробки. В результаті аналізу розрахунків можна зробити висновок, що розроблений програмний продукт за ціною дешевший за аналог і є висококонкурентоспроможним. Період окупності складе близько 0,93 роки.

ВИСНОВКИ

У ході виконання магістерської кваліфікаційної роботи проаналізовано сучасний стан розвитку технологій безконтактної передачі персональних даних, доведено актуальність розробки інформаційної технології, проведено аналіз та порівняння методів стиснення, розглянуто системи-аналоги та проведено їх порівняльний аналіз та на основі їх порівняння встановлено вимоги та завдання для власної розробки.

За результатами дослідження технологій безконтактної передачі даних та методів алгоритмів стиснення даних, обрано технологію NFC (для передачі інформації), та алгоритм LZW (для стиснення даних). За результатами дослідження обґрунтовано метод розв’язання задачі.

Виконано моделювання інформаційної технології безконтактної передачі інформації, в процесі якого визначено, що інформаційна технологія буде представлена у формі кросплатформеного мобільного додатку з серверною частиною та базою даних. Створено UML діаграму послідовності взаємодії класів інформаційної технології.

На основі проведеного моделювання обґрунтовано застосування мови програмування JavaScript та фреймворків для мобільного додатку, веб-додатку, серверної частини: React Native, React, Firebase. В результаті розроблено усі частини інформаційної технології, доведено їх працездатність та ефективність.

Проведено 50 циклів тестування інформаційної технології, в результаті якого показники стиснення персональної інформації є в межах 13%, що є більшим за значення поставлене у задачі магістерської кваліфікаційної роботи.

Проведено розрахунок витрат на розробку нового програмного продукту, спрогнозовано орієнтовану величину витрат, розраховано чистий прибуток. Розроблена інформаційна технологія є висококонкурентоспроможною та період окупності складе близько 0,93 роки.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Yarovyi A.A., Kaplunskyi O.A., Yarova O.A. Information technology for contactless exchange of personal data // Eurasian scientific discussions. Proceedings of the 12th International scientific and practical conference. Barca Academy Publishing. Barcelona, Spain. 2022. Pp. 212-218. [Електронний ресурс] – Режим доступу: https://sci-conf.com.ua/wp-content/uploads/2022/12/EURASIAN-SCIENTIFIC-DISCUSSIONS-18-20.12.22.pdf
2. Бездротові технології [Електронний ресурс] – Режим доступу: https://web.archive.org/web/20090423000634/http://www.wireless-e.ru/articles/technologies.php – Назва з екрану.
3. Технології бездротового зв’язку в ноутбуках технології. Функції і настройка ОС [Електронний ресурс] – Режим доступу: https://static-course-assets.s3.amazonaws.com/ITE50UK/course/module7/7.4.1.2/7.4.1.2.html– Назва з екрану.
4. Оптичний бездротовий зв’язок [Електронний ресурс] – Режим доступу: https://artsandculture.google.com/entity/m010vvkf0?hl=uk – Назва з екрану.
5. Енциклопедія техніки [Електронний ресурс] – Режим доступу: https://dic.academic.ru/dic.nsf/enc_tech/990/%D1%80%D0%B0%D0%B4%D0%B8%D0%BE%D0%BA%D0%B0%D0%BD%D0%B0%D0%BB – Назва з екрану.
6. Що таке дискета? [Електронний ресурс] – Режим доступу: https://uk.zaptech.net/what-is-floppy-disk – Назва з екрану.
7. Стиснення даних, Халід Сайуд [Електронний ресурс] – Режим доступу: https://www.csd.uoc.gr/~hy438/lectures/Sayood-DataCompression.pdf – Назва з екрану.
8. Типи оптичних дисків [Електронний ресурс] – Режим доступу: https://www.chaynikam.info/ukr/stat_cd.html – Назва з екрану.

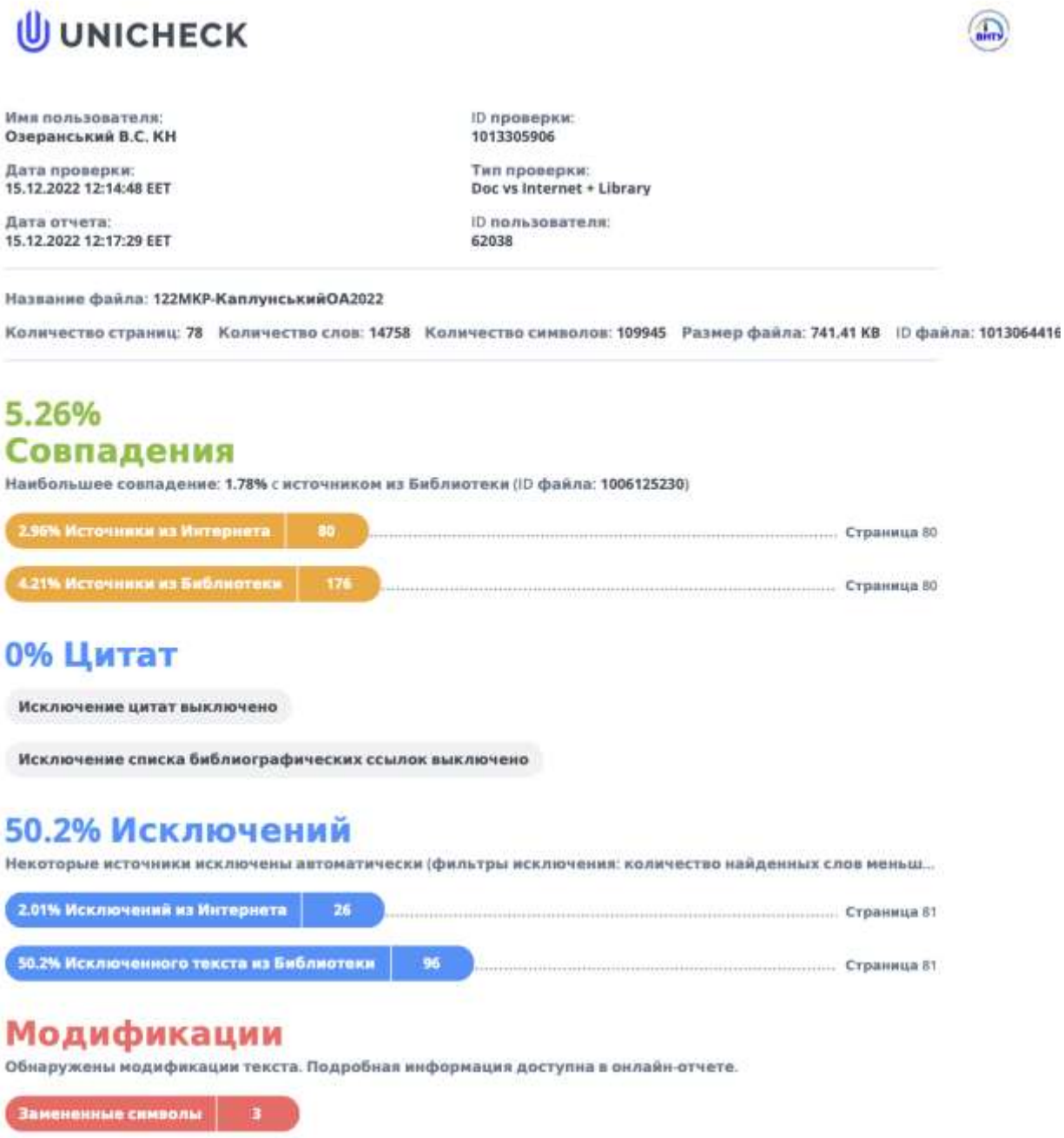
9. USB-флеш-накопичувач [Електронний ресурс] – Режим доступу: https://www.wikiwand.com/uk/USB-флеш-накопичувач – Назва з екрану.
10. Види штрих-кодів [Електронний ресурс] – Режим доступу: http://www.neo.ua/ua/zakony/vidy-shtrix-kodov – Назва з екрану.
11. QR-коди [Електронний ресурс] – Режим доступу: https://en.wikipedia.org/wiki/QR_code#Encoding – Назва з екрану.
12. NFC: розгляд технології Near Field Communication [Електронний ресурс] – Режим доступу: https://droider.ru/post/nfc-razbor-tehnologii-near-field-communication-27-05-2020/ – Назва з екрану.
13. About Instagram [Електронний ресурс] – https://about.instagram.com/ – Назва з екрану.
14. Popl – Your digital business card [Електронний ресурс] – https://popl.co/ – Назва з екрану.
15. PhoneTag [Електронний ресурс] https://phonetag.co/– Назва з екрану.
16. NFC – ближній безконтактний зв’язок [Електронний ресурс] – Режим доступу: https://www.it.ua/knowledge-base/technology-innovation/nfc – Назва з екрану.
17. Near Field Communication [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Near_Field_Communication#%D0%9C%D1%96%D1%82%D0%BA%D0%B8 – Назва з екрану.
18. Klaus Finkenzeller, «RFID Handbuch», Hanser Verlag.
19. НФС форум [Електронний ресурс] – Режим доступу: https://nfc-forum.org/ – Назва з екрану.
20. Електромагнітна індукція. Закон електромагнітної індукції [Електронний ресурс] – Режим доступу: https://uahistory.co/pidruchniki/sirotyk-physics-and-astronomy-11-class-2019-standard-level/21.php – Назва з екрану.
21. Електромагнітна індукція: формула, як це працює, приклади [Електронний ресурс] – Режим доступу: https://uk.warbletoncouncil.org/induccion-electromagnetica-10040 – Назва з екрану.

22. Алгоритми LZW, LZ77 та LZ78 [Електронний ресурс] – Режим доступу: https://habr.com/post/132683/ – Назва з екрану.
23. How data compression works: exploring LZ77 [Електронний ресурс] – Режим доступу: https://towardsdatascience.com/how-data-compression-works-exploring-lz77-3a2c2e06c097 – Назва з екрану.
24. Hardware Implementation of LZMA Data Compression Algorithm [Електронний ресурс] – Режим доступу: https://research.ijais.org/volume5/number4/ijais12-450900.pdf – Назва з екрану.
25. Діаграми UML для моделювання процесів і архітектури проекту [Електронний ресурс] – Режим доступу: https://evergreens.com.ua/ua/articles/uml-diagrams.html – Назва з екрану.
26. Грамотна клієнт-серверна архітектура: як правильно проектувати і розробляти web API [Електронний ресурс] – Режим доступу: https://echo.lviv.ua/dev/6455 – Назва з екрану.
27. Dart | Документація [Електронний ресурс] – Режим доступу до ресурсу: https://metanit.com/dart/tutorial – Назва з екрану.
28. Чому програмісти C# скоро будуть нарозхват [Електронний ресурс] – Режим доступу: https://proglib.io/p/c-sharp-popularity – Назва з екрану.
29. Що таке JavaScript, та як функціонує JavaScript [Електронний ресурс] – Режим доступу: http://ruszura.in.ua/uncategorized/scho-take-javascript-yak-funktsionuyu-javascript.html – Назва з екрану.
30. Angular 5 [Електронний ресурс] – Режим доступу: https://habr.com/ru/post/341688/#s1_2 – Назва з екрану.
31. Vue.js вступ [Електронний ресурс] – Режим доступу: https://vuejs.org/v2/guide/ – Назва з екрану.
32. Вступ до React JS [Електронний ресурс] – Режим доступу: https://reactjs.org/tutorial/tutorial.html – Назва з екрану.

33. Apache cordova [Електронний ресурс] – Режим доступу: https://cordova.apache.org/ – Назва з екрану.
34. React Native – вступ [Електронний ресурс] – Режим доступу: https://reactnative.dev/docs/getting-started – Назва з екрану.
35. Вступ в Expo вступ [Електронний ресурс] – Режим доступу: https://docs.expo.io/ – Назва з екрану.
36. NestJS – прогресивний Node.js фреймворк [Електронний ресурс] – Режим доступу: https://docs.nestjs.com/ – Назва з екрану.
37. Express – фреймворк для веб-додатків Node.js [Електронний ресурс]. – Режим доступу: https://expressjs.com/ – Назва з екрану.
38. Firebase документація [Електронний ресурс] – Режим доступу: https://firebase.google.com/docs – Назва з екрану.

ДОДАТКИ

Додаток А (обов'язковий)
Протокол перевірки МКР на наявність текстових запозичень



UNICHECK

Имя пользователя:
Озеранський В.С. КН

ID проверки:
1013305906

Дата проверки:
15.12.2022 12:14:48 EET

Тип проверки:
Doc vs Internet + Library

Дата отчета:
15.12.2022 12:17:29 EET

ID пользователя:
62038

Название файла: 122МКР-КаплунськийОА2022

Количество страниц: 78 Количество слов: 14758 Количество символов: 109945 Размер файла: 741.41 KB ID файла: 1013064416

5.26%
Совпадения

Наибольшее совпадение: 1.78% с источником из Библиотеки (ID файла: 1006125230)

2.96% Источники из Интернета 80 ..... Страница 80

4.21% Источники из Библиотеки 176 ..... Страница 80

0% Цитат

Исключение цитат выключено

Исключение списка библиографических ссылок выключено

50.2% Исключений

Некоторые источники исключены автоматически (фильтры исключения: количество найденных слов меньш...

2.01% Исключений из Интернета 26 ..... Страница 81

50.2% Исключенного текста из Библиотеки 96 ..... Страница 81

Модификации

Обнаружены модификации текста. Подробная информация доступна в онлайн-отчете.

Замененные символы 3

Додаток Б (обов'язковий)

Лістинг програми

```
// модуль запису інформації на носій
const ActivateLink = (props) => {
  const {navigationToActivate, changeActivate, navigation, profileName} = props;
  const [isCompression, setIsCompression] = useState(false);

  useEffect(() => {
    writeLinkToNfc();
    return () => {
      // NfcManager.unregisterTagEvent().catch(() => 0);
    };
  }, []);

  const _cleanUp = () => {
    // NfcManager.unregisterTagEvent().catch(() => 0);
    NfcManager.cancelTechnologyRequest().catch(() => 0);
  };

  const writeLinkToNfc = async () => {
    const userProfileLink = generateProfileLink(profileName);
    try {
      let resp = await NfcManager.requestTechnology(NfcTech.Ndef, {
        alertMessage:
          'Hold your Phone Tag to the top back of your phone to activate it.!',
      });
     // Запис даних на NFC-носій
      const payload = isCompression ? LZVEncode(resp) : resp;
      let ndefRecord = Ndef.uriRecord(payload);
      console.log('ndefRecord', ndefRecord);
      let bytes = Ndef.encodeMessage([ndefRecord]);
      console.log('bytes', bytes);
      await NfcManager.writeNdefMessage(bytes);
      console.log('successfully write ndef');
      await NfcManager.setAlertMessageIOS('I got your tag!');
      _cleanUp();
      navigation.navigate('ActivateLinkNext');
    } catch (ex) {
      await NfcManager.setAlertMessageIOS(
        'Failure to activate NFC tag. It might be not empty!',
      );
      console.log('ex', ex);
      _cleanUp();
    }
  };

  return (
    <View style={s.container}>
```

```
      <Image
        style={s.phone}
        resizeMode={'contain'}
        source={
          Platform.OS === 'ios'
            ? require('../../assets/activate_ios_phone.png')
            : require('../../assets/activate_android_phone.png')
        }
      />
      <Text style={s.labelText}>Your can enable data compression</Text>
      <Switch
        toggleSwitch={() => setIsCompression(!isCompression)}
        switchInitial={isCompression}
        disable={false}
      />
      <Text style={s.labelText}>
        Your Tappin will be activated with the current profile you are logged in
        with
      </Text>
      <TouchableOpacity
        style={button.buttonGreen}
        onPress={() => writeLinkToNfc()}>
        <Text style={button.buttonText}>Activate</Text>
      </TouchableOpacity>
    </View>
  );
};

// підключення до Redux
const connectToStore = connect(
  (state) => {
    const {profileName} = state.user.userInfo;
    const {navigationToActivate} = state.uiState;
    return {profileName, navigationToActivate};
  },
  (dispatch) => {
    return {
      changeActivate: (status) => {
        dispatch(changeNavToActivate(status));
      },
    };
  },
);

export default connectToStore(ActivateLink);

// функція стиснення за алгоритмом LZV
export function LZVEncode(s) {
  const dict = {};
  const data = (s + '').split('');
  const out = [];
```

```
  let currChar;
  let phrase = data[0];
  let code = 256;
  for (let i = 1; i < data.length; i++) {
    currChar = data[i];
    if (dict[phrase + currChar] != null) {
      phrase += currChar;
    } else {
      out.push(phrase.length > 1 ? dict[phrase] : phrase.charCodeAt(0));
      dict[phrase + currChar] = code;
      code++;
      phrase = currChar;
    }
  }
  out.push(phrase.length > 1 ? dict[phrase] : phrase.charCodeAt(0));
  for (let i = 0; i < out.length; i++) {
    out[i] = String.fromCharCode(out[i]);
  }
  return out.join('');
}

// Функція розгортування за алгоритмом LZW
export function LZWDecode(s) {
  const dict = {};
  const data = (s + '').split('');
  let currChar = data[0];
  let oldPhrase = currChar;
  const out = [currChar];
  let code = 256;
  let phrase;
  for (let i = 1; i < data.length; i++) {
    const currCode = data[i].charCodeAt(0);
    if (currCode < 256) {
      phrase = data[i];
    } else {
      phrase = dict[currCode] ? dict[currCode] : oldPhrase + currChar;
    }
    out.push(phrase);
    currChar = phrase.charAt(0);
    dict[code] = oldPhrase + currChar;
    code++;
    oldPhrase = phrase;
  }
  return out.join('');
}


// модуль навігації додатку

const hiddenRoutes = [
  'EditProfile',
```

```
  'People',
  'PersonalQr',
  'ActivateLinkNext',
  'FirstRegIntermediate',
  'EditDrawerProfile',
  'Tutorial',
  'ActivatePhoneTag',
];

export default function DrawerCustomContent(props) {
  const dispatch = useDispatch();
  const navigation = useNavigation();

  const [logoutDialog, setLogoutDialog] = useState(false);

  const logOutOpen = () => {
    setLogoutDialog(true);
  };
  const logOut = () => {
    dispatch(performFirebaseLogout());
    dispatch(makeEmptyeStore());
    dispatch(makeUserDataEmpty());
    dispatch(setIsAuthenticated(false));

    // navigation.replace('Welcome');
  };

  const toDrawerEditProfile = () => {
    navigation.navigate('EditProfile', {showSaveButton: true});
  };

  const toHome = () => {
    dispatch(makeFirstPageActive(true));
    navigation.navigate('Home');
  };

// функція активації процесу запису даних на носій
  const toActivatePhoneTag = () => {
    dispatch(changeNavToActivate(true));
    navigation.navigate('ActivatePhoneTag');
  };

  const toTutorial = () => {
    navigation.navigate('Tutorial');
  };

  const scanNfc = async () => {
    try {
      await NfcManager.registerTagEvent();
    } catch (ex) {
      console.warn('ex', ex);
```

```
    NfcManager.unregisterTagEvent().catch(() => 0);
  }
};
const toHelp = async () => {
  try {
    await Linking.openURL('https://www.phonetag.co/faq');
  } catch (error) {
    console.log('error ', error);
  }
};
const toBuyPage = async () => {
  try {
    await Linking.openURL('https://www.phonetag.co/shop');
  } catch (error) {
    console.log('error ', error);
  }
};

const renderDrawer = () => {
  if (logoutDialog === true) {
    return (
      <>
        <View style={s.overlayHiddenMenuLogout}>
          <Image
            resizeMode={'contain'}
            style={s.logoImage}
            source={require('../assets/phone_tag_logo.png')}
          />
        </View>
        <View style={s.logoutMessage}>
          <Image
            resizeMode={'stretch'}
            style={s.logoutIcon}
            source={require('../assets/logout-icon.png')}
          />
          <Text style={s.logoutMessageLabel}>
            Are you sure you want to logout?
          </Text>
        </View>
        <View
          style={{
            alignItems: 'center',
            width: '100%',
            justifyContent: 'center',
          }}>
          <TouchableOpacity
            onPress={() => logOut()}
            style={[s.logoutButton, {borderColor: button.buttonBorderGreen}]}>
            <Text style={s.logoutText}>Yes</Text>
          </TouchableOpacity>
          <TouchableOpacity
```

```
            onPress={() => setLogoutDialog(false)}
            style={[
              s.logoutButton,
              {borderColor: button.buttonBorderPurple},
            ]}>
            <Text style={s.logoutText}>No</Text>
          </TouchableOpacity>
        </View>
      </>
    );
  } else {
    const filteredProps = {
      ...props,
      state: {
        ...props.state,
        routeNames: props.state.routeNames.filter(
          (routeName) => !hiddenRoutes.includes(routeName),
        ),
        routes: props.state.routes.filter(
          (route) => !hiddenRoutes.includes(route.name),
        ),
      },
    };

// візуальна частина навігації
    return (
      <>
        <View style={s.overlayHiddenMenu}>
          <Image
            resizeMode={'contain'}
            style={s.logoImage}
            source={require('../assets/phone_tag_logo.png')}
          />
        </View>
        <DrawerItem
          labelStyle={{fontSize: 18, color: common.commonTextColor}}
          icon={() => (
            <Image
              source={require('../assets/drawer_home_icon.png')}
              style={s.drawerIcon}
            />
          )}
          label="Home"
          style={{
            width: '100%',
            margin: 0,
            padding: 0,
          }}
          onPress={() => toHome()}
        />
        <DrawerItem
```

```
  labelStyle={{fontSize: 18, color: common.commonTextColor}}
  icon={() => (
    <Image
      source={require('../assets/drawer_edit_profile_icon.png')}
      style={s.drawerIcon}
    />
  )}
  label="Edit Profile"
  onPress={() => toDrawerEditProfile()}
/>
<DrawerItem
  style={{marginRight: 0}}
  labelStyle={{fontSize: 18, color: common.commonTextColor}}
  icon={() => (
    <Image
      source={require('../assets/drawer_activate_tag_icon.png')}
      style={s.drawerIcon}
    />
  )}
  label="Activate Phone Tag"
  onPress={() => toActivatePhoneTag()}
/>
<DrawerItem
  labelStyle={{fontSize: 18, color: common.commonTextColor}}
  icon={() => (
    <Image
      source={require('../assets/drawer_info_icon.png')}
      style={s.drawerIcon}
    />
  )}
  label="Tutorial"
  onPress={() => toTutorial()}
/>
<DrawerItem
  labelStyle={{fontSize: 18, color: common.commonTextColor}}
  icon={() => (
    <Image
      source={require('../assets/drawer_buy_tag_icon.png')}
      style={s.drawerIcon}
    />
  )}
  label="Buy Phone Tag"
  onPress={() => toBuyPage()}
/>
<DrawerItem
  labelStyle={{fontSize: 18, color: common.commonTextColor}}
  icon={() => (
    <MaterialIcon
      name={'cellphone-nfc'}
      size={25}
      color={common.commonTextColor}
```

```
            />
          )}
          label="Read Phone Tag"
          onPress={() => scanNfc()}
        />
        <DrawerItem
          labelStyle={{fontSize: 18, color: common.commonTextColor}}
          icon={() => (
            <Image
              source={require('../assets/drawer_help_icon.png')}
              style={s.drawerIcon}
            />
          )}
          label="Help"
          onPress={() => toHelp()}
        />
        <DrawerItem
          labelStyle={{fontSize: 18, color: common.commonTextColor}}
          icon={() => (
            <Image
              source={require('../assets/logout.png')}
              style={s.drawerIcon}
            />
          )}
          label="Logout"
          onPress={() => logOutOpen()}
        />
      </>
    );
  }
 };

 return (
  <DrawerContentScrollView {...props}>
    {renderDrawer()}
  </DrawerContentScrollView>
 );
}

// стилі навігації
const s = StyleSheet.create({
 title: {
  fontSize: 24,
  fontWeight: 'bold',
  marginTop: -10,
  textAlign: 'center',
 },
 logoutButton: {
  alignSelf: 'center',
  borderWidth: 2,
  width: 150,
```

```
  paddingVertical: 5,
  paddingHorizontal: 20,
  borderRadius: 8,
  textAlign: 'center',
  marginBottom: 30,
},
logoutText: {
  fontSize: 18,
  color: common.commonTextColor,
  textAlign: 'center',
},
titleContainer: {
  justifyContent: 'center',
  alignItems: 'center',
},
drawerIcon: {
  width: 25,
  height: 25,
},
logoImage: {
  height: 50,
  marginBottom: 40,
  marginTop: 40,
  width: '85%',
  alignSelf: 'center',
},
logoutIcon: {
  height: 45,
  width: 45,
  marginBottom: 20,
  marginLeft: 20,
  alignSelf: 'center',
},
logoContainer: {
  height: 80,
  width: '100%',
},
overlayHiddenMenu: {
  zIndex: 100,
  borderWidth: 1,
  backgroundColor: 'transparent',
  borderColor: 'transparent',
  width: '100%',
  height: '30%',
  paddingTop: 0,
  // position: 'absolute',
  justifyContent: 'flex-end',
  alignItems: 'center',
},
overlayHiddenMenuLogout: {
  zIndex: 100,
```

```
    borderWidth: 1,
    borderColor: 'transparent',
    width: '100%',
    marginTop: 41,
    // position: 'absolute',
    justifyContent: 'center',
    alignItems: 'center',
  },
  logoutMessage: {
    width: '100%',
    marginTop: 20,
    marginBottom: 40,
    alignItems: 'center',
    justifyContent: 'center',
  },
  logoutMessageLabel: {
    fontSize: 17,
    color: '#fff',
  },
});

// функція створення посилання
export default (userName) => {
  if (!userName) {
    throw new Error("Generate user link is failed. username can't be empty");
  }
  return config.baseTappinWeb + userName + '?nfclabel=tappin';
};

const Stack = createStackNavigator();

// модуль основної навігації додатку
const Navigation = React.forwardRef(({onStateChange}, ref) => {
  const {isAuthenticated} = useSelector((state) => state.auth);
  const dispatch = useDispatch();

  useEffect(() => {
    dispatch(checkIsAuth());
  }, []);

// рівні доступу залежно від того чи автентифікований користувач
  if (isAuthenticated === null) {
    return (
      <View
        style={{
          width: '100%',
          height: '100%',
          justifyContent: 'center',
          alignItems: 'center',
        }}>
        <View style={{height: 70, width: 70}}>
```

```
          <Progress.CircleSnail size={70} />
        </View>
      </View>
    );
  }
  return (
    <NavigationContainer
      onStateChange={onStateChange}
      ref={ref}
      // initialState={isPersistState()}
    >
      <Stack.Navigator
        screenOptions={{
          gestureEnabled: Platform.OS === 'ios',
          ...TransitionPresets.SlideFromRightIOS,
        }}
        headerMode={'none'}>
        {isAuthenticated ? (
          <Stack.Screen
            name={'DrawerNavigation'}
            component={DrawerNavigation}
          />
        ) : (
          <Stack.Screen name={'Registration'} component={Registration} />
        )}
      </Stack.Navigator>
    </NavigationContainer>
  );
});
export default Navigation;

// точка входу в програму, основний модуль
const App = (props) => {
  let navigationRef = useRef(null);
  useEffect(() => {
    urlSet(props.initialURL);
    Linking.addEventListener('url', urlSet);
// ініціалізація NFC менеджера
    NfcManager.start();
    NfcManager.setEventListener(NfcEvents.DiscoverTag, (tag) => {
      NfcManager.setAlertMessageIOS('I got your tag!');
      navigationRef.current.dispatch(DrawerActions.closeDrawer());
      const tagInfo = decodeTad(tag);
      navigateToLink(tagInfo[0][1]);
      NfcManager.unregisterTagEvent().catch(() => 0);
    });
    return () => {
      Linking.removeEventListener('url', urlSet);
      NfcManager.setEventListener(NfcEvents.DiscoverTag, null);
      NfcManager.unregisterTagEvent().catch(() => 0);
    };
```

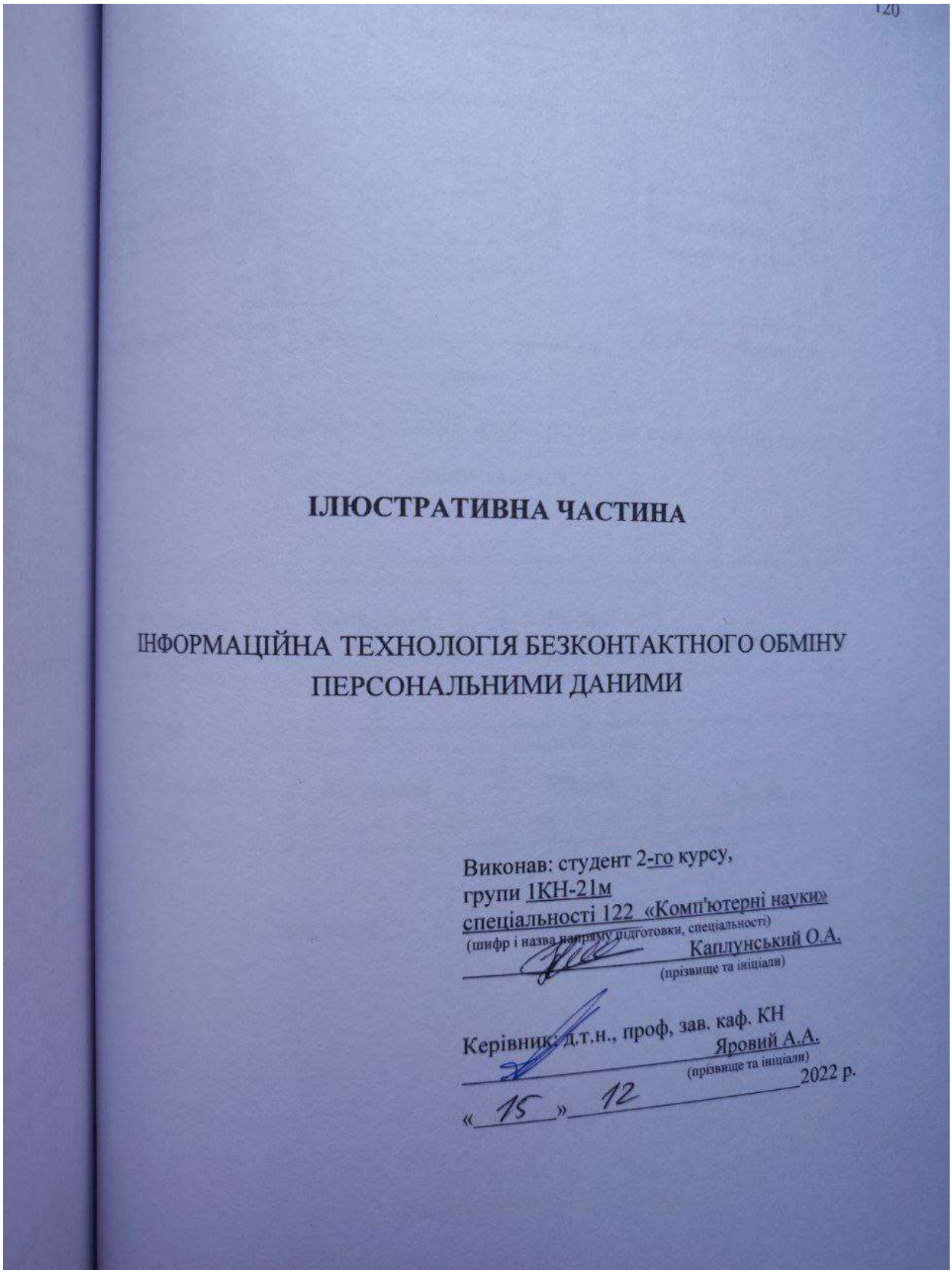
```
  }, []);
  const navigateToLink = (tagDecoded) => {
    if (tagDecoded) {
      Linking.canOpenURL(tagDecoded).then((supported) => {
        if (supported) {
          Linking.openURL(tagDecoded);
        } else {
          console.warn(' cant opened link ');
        }
      });
    }
  };
  const decodeTad = (tag) => {
    let parsed = null;
    if (tag.ndefMessage && tag.ndefMessage.length > 0) {
      // according to its TNF & type
      const ndefRecords = tag.ndefMessage;

      function decodeNdefRecord(record) {
        if (Ndef.isType(record, Ndef.TNF_WELL_KNOWN, Ndef.RTD_TEXT)) {
          return ['text', Ndef.text.decodePayload(record.payload)];
        } else if (Ndef.isType(record, Ndef.TNF_WELL_KNOWN, Ndef.RTD_URI)) {
          return ['uri', Ndef.uri.decodePayload(record.payload)];
        }

        return ['unknown', '---'];
      }

      parsed = ndefRecords.map(decodeNdefRecord);
    }
    return parsed;
  };
  const urlSet = async (url) => {
    const username = url.url.replace('tappinapp://', '');
    if (username) {
      await saveDeepLinkUser(username);
    }
  };
  const handleNavigatorRef = (ref) => {
    navigationRef.current = ref;
    NavigationService.setTopLevelNavigator(ref);
  };
  return (
    <>
      <StatusBar barStyle="light-content" />
      <Provider store={store}>
        <Navigation ref={handleNavigatorRef} />
      </Provider>
    </>
  );
};
```

Додаток В (обов'язковий)



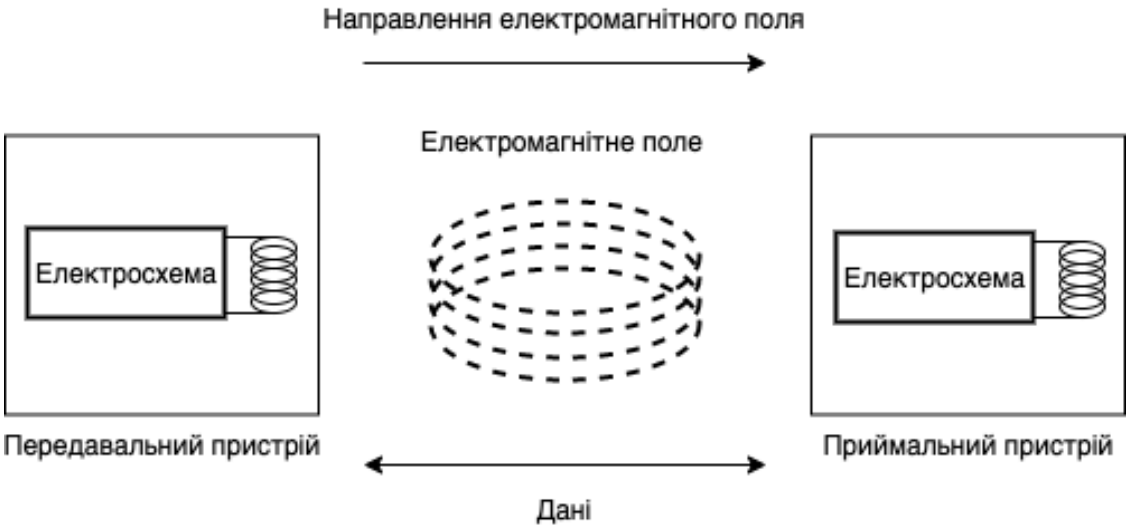


Рисунок В.1 – Схема взаємодії двох пристроїв при передачі даних за допомогою NFC

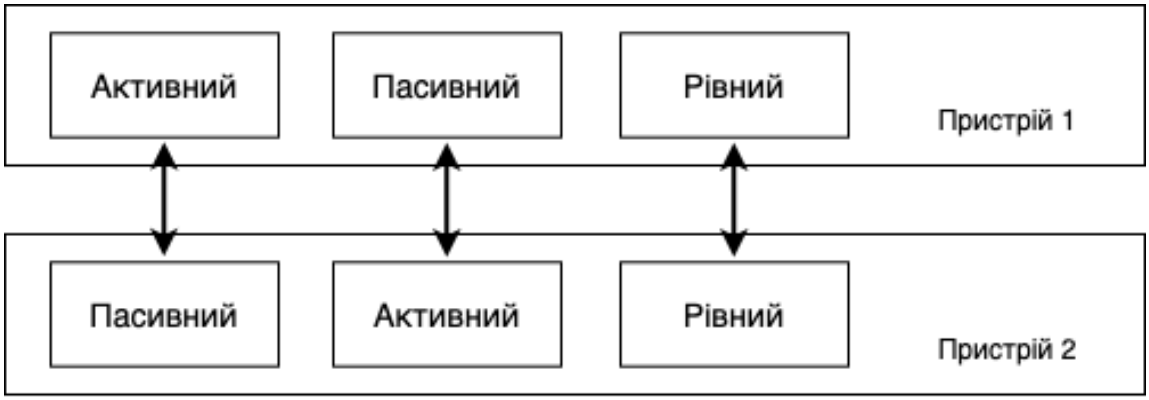


Рисунок В.2 – Схема режимів роботи NFC

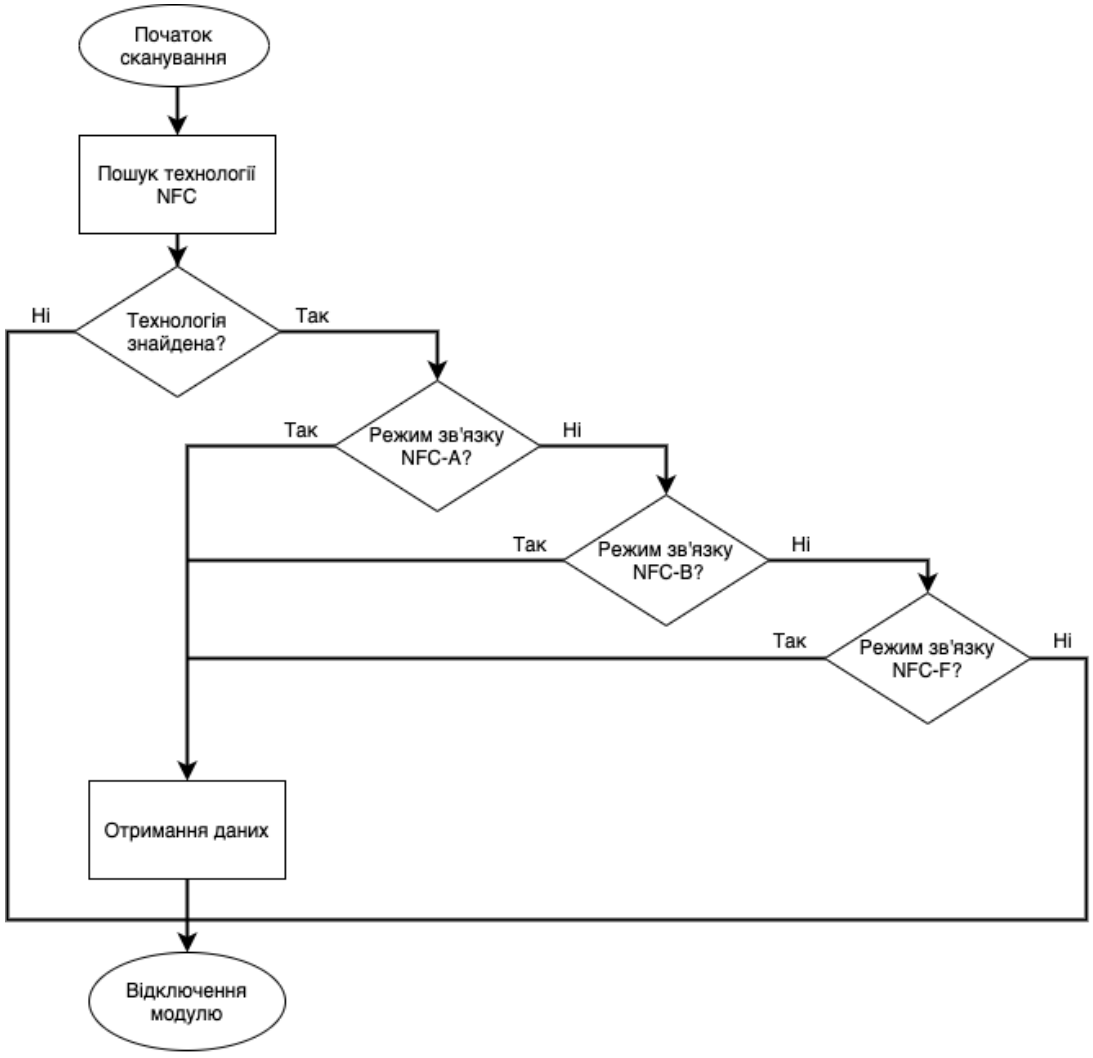


Рисунок В.3 – Фрагмент схеми алгоритму пошуку режиму зв’язку NFC модуля

Таблиця В.1 – Характеристики стандартів NFC

Специфікація технічних стандартів радіоінтерфейсів NFC					
Стандарти асоціації NFC-Forum	Прийом(П)/ Сканування (С)	Кодування	Модуляція	Швидкість передачі даних	Частота
NFC-A	С	Модифікований код Міллера	Амплітудна маніпуляція (ASK) 100%	106 кбит/с	13,56 МГц
	П	Манчестерський код	Навантажувальна модуляція (ASK)	106 кбит/с	13,56 МГц +- 848 кГц
NFC-B	С	NRZ-L	Амплітудна маніпуляція (ASK) 10%	106 кбит/с	13,56 МГц
	П	NRZ-L	Навантажувальна модуляція (BPSK)	106 кбит/с	13,56 МГц +- 848 кГц
NFC-F	С	Манчестерський код	Амплітудна маніпуляція (ASK) 10%	212 / 424 кбит/с	13,56 МГц
	П	Манчестерський код	Навантажувальна маніпуляція (ASK)	212 / 424 кбит/с	13,56 МГц

Таблиця В.2 – Візуалізація ітерацій роботи алгоритму LZW

Поточний рядок	Поточний символ	Наступний символ	Вивід		Словник
			Індекс	Біти	
AB	A	B	0	000	5: AB
BA	B	A	1	001	6: BA
AC	A	C	0	000	7: AC
CA	C	A	2	010	8: CA
AB	A	B	-	-	-
ABA	B	A	5	101	9: ABA
AD	A	D	0	000	10: AD
DA	D	A	3	011	11: DA
AB	A	B	-	-	-
ABA	B	A	-	-	-
ABAC	A	C	9	1001	12: ABAC
CA	C	A	-	-	-
CAB	A	B	8	1000	13: CAB
BA	B	A	-	-	-
BAE	A	E	6	0110	15: BAE
E	E	-	4	0100	-

Таблиця В.3 – Приклад роботи дельта-кодування в алгоритмі LZMA

Вхідна послідовність:	2,3,4,6,7,9,8,7,5,3,4
Закодована послідовність:	2,1,1,2,1,2,−1,−1,−2,−2,1
Кількість різних символів на вході:	8
Кількість різних символів на виході:	4

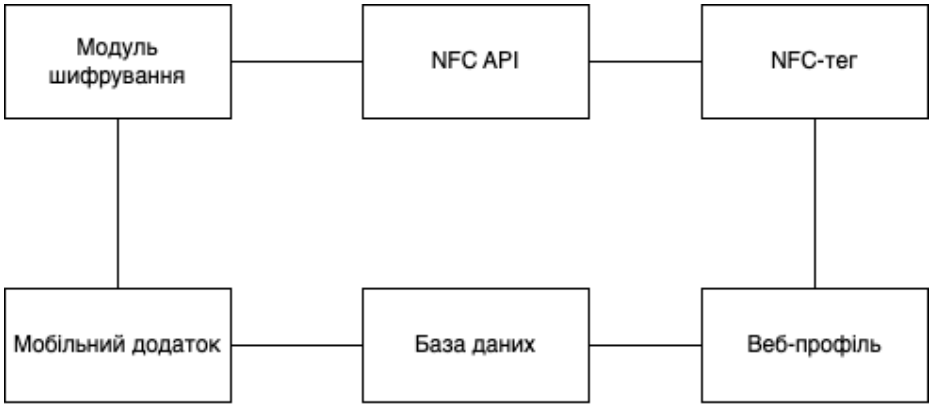


Рисунок В.4 – Загальна структурна схема інформаційної технології безконтактного обміну персональними даними

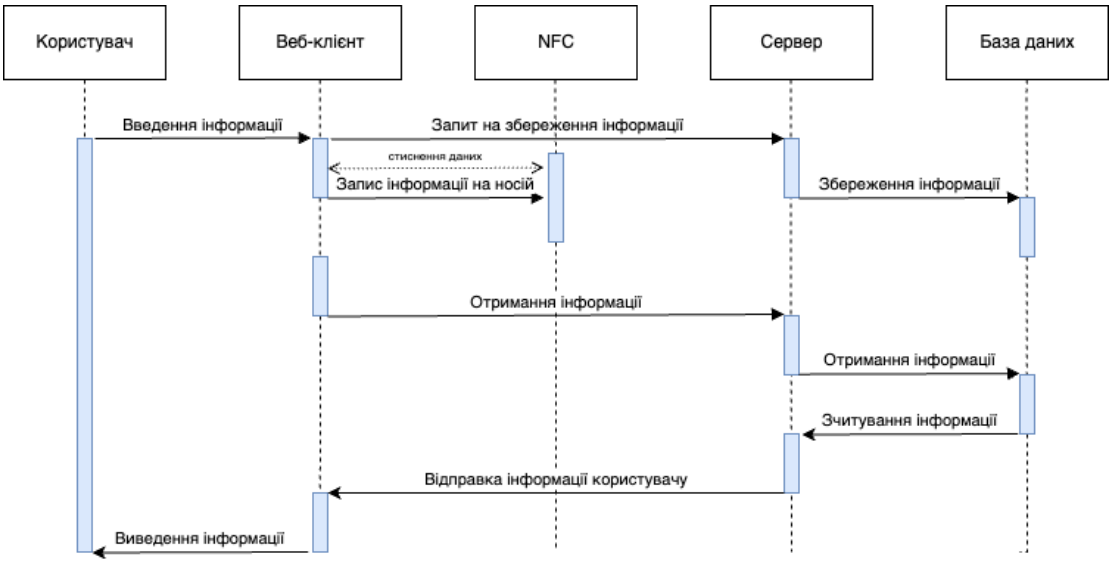


Рисунок В.5 – Діаграма послідовності взаємодії модулів інформаційної технології безконтактного обміну персональними даними

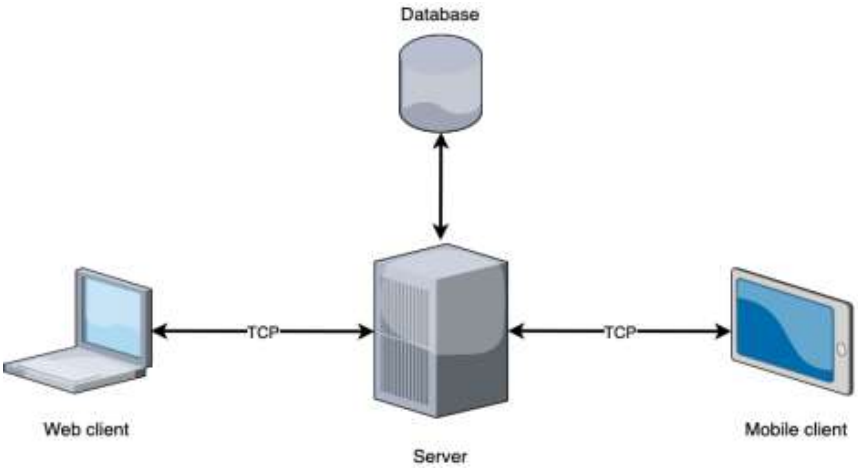


Рисунок В.6 – Схема системи клієнт-сервер

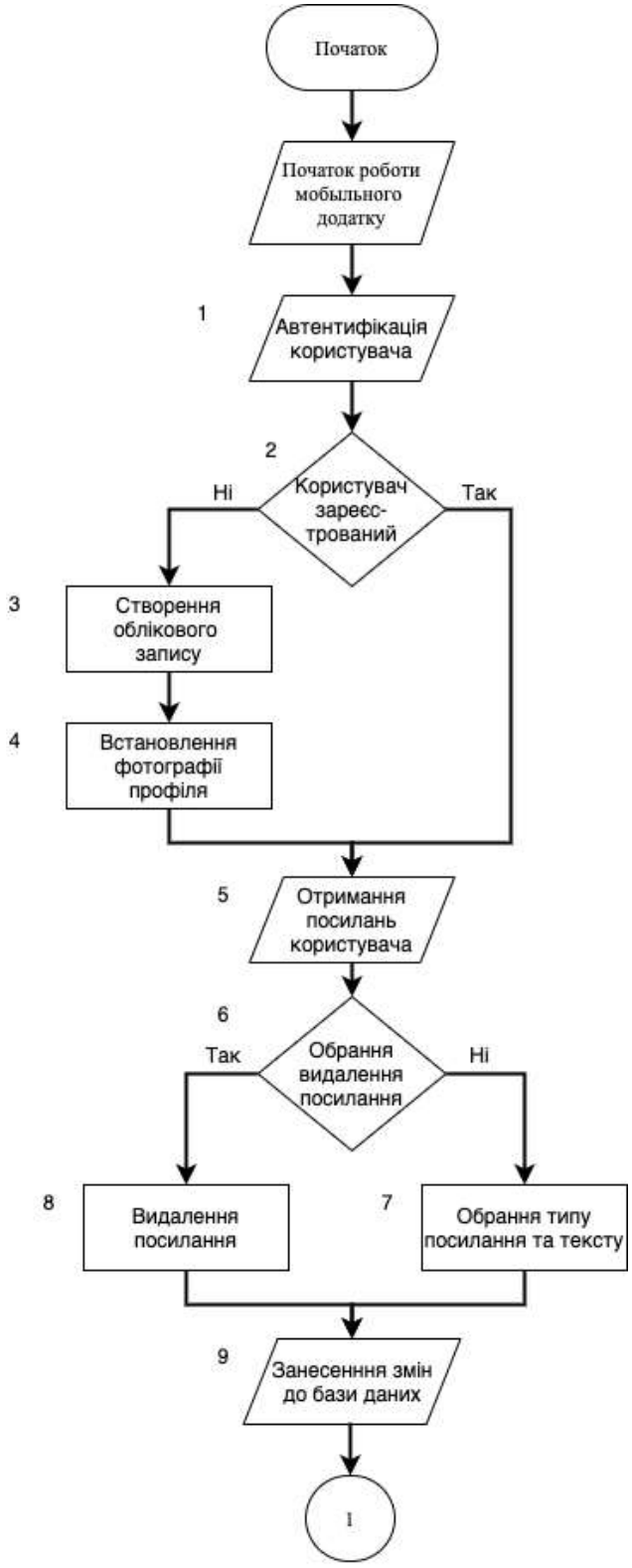


Рисунок В.7 – Схема алгоритму роботи інформаційної технології безконтактного обміну персональними даними

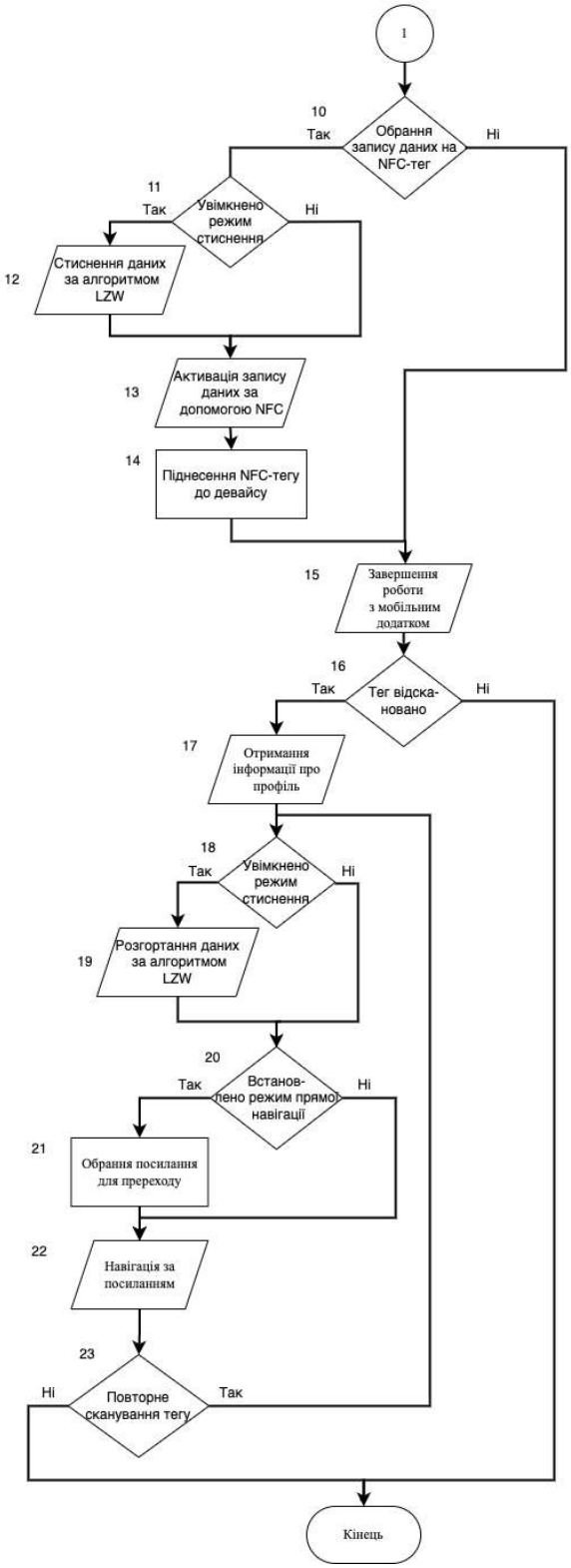


Рисунок В.7 – Схема алгоритму роботи інформаційної технології безконтактного обміну персональними даними, аркуш 2

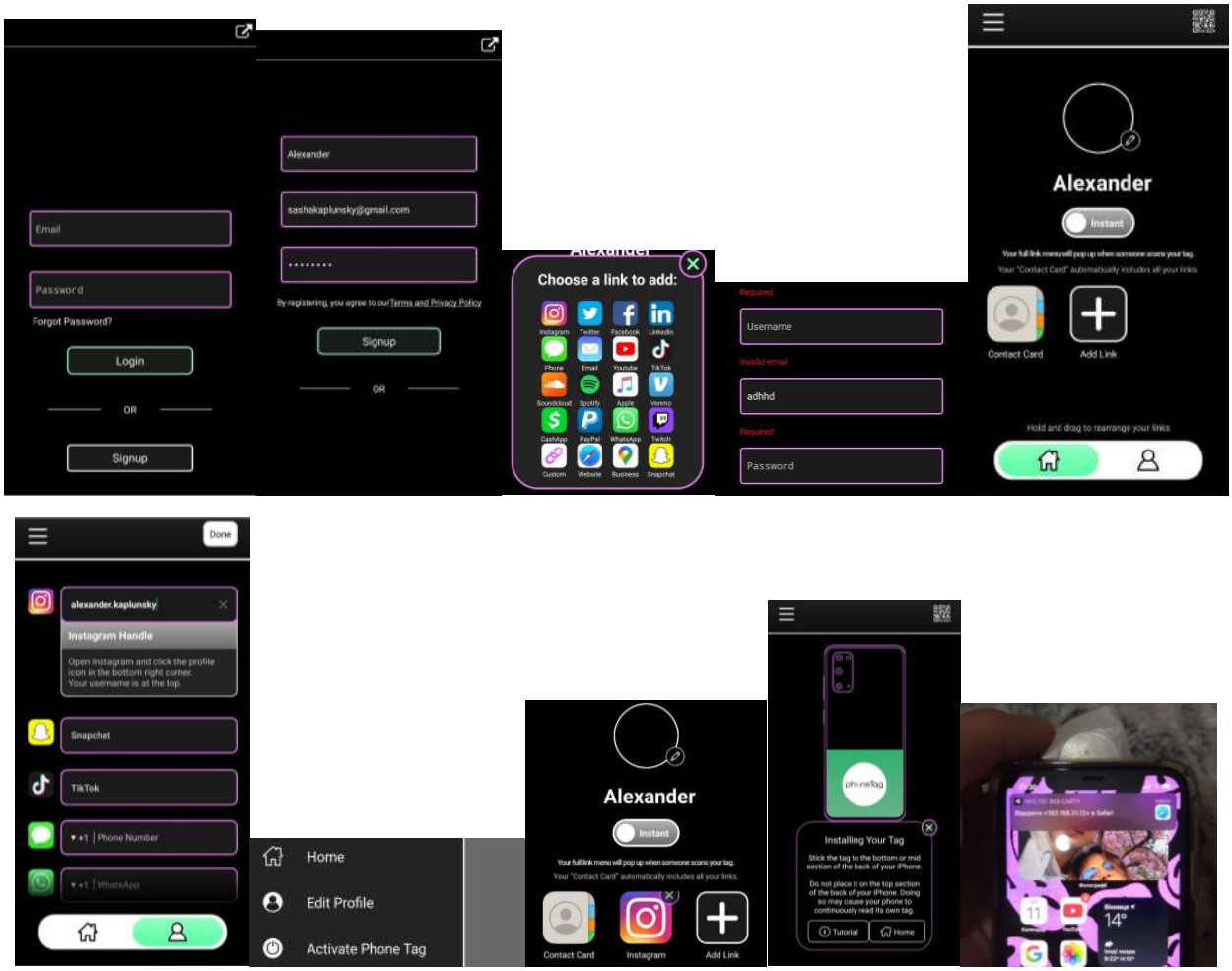


Рисунок В.8 – Загальний вигляд роботи програми

Рисунок В.9 – Загальний вигляд стисненого посилання на профіль користувача

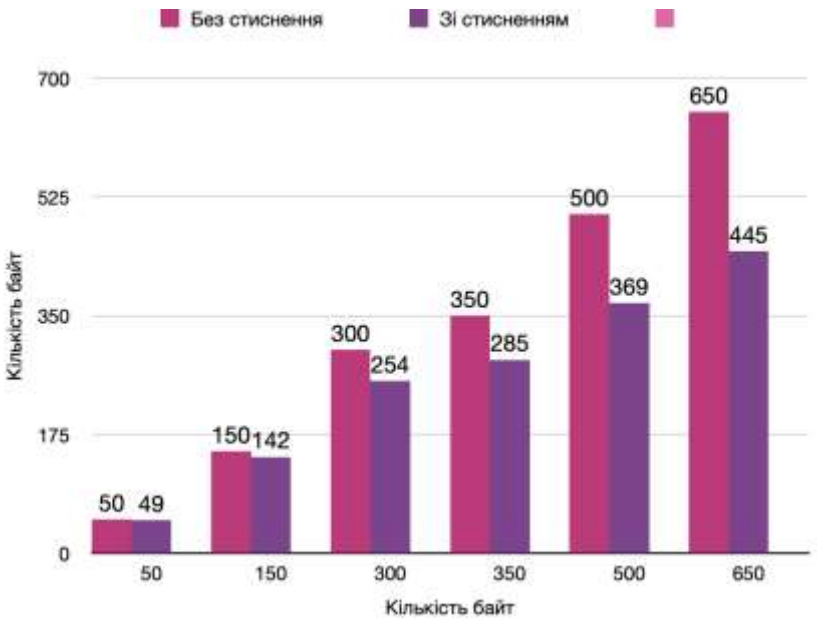


Рисунок В.10 – Діаграма залежності стиснення від об’єму інформації

Додаток Г (довідниковий)
Інструкція користувача

Для початку роботи з програмним модулем необхідна наявність смартфону або іншого пристрою з операційною системою Android або IOS, які офіційно підтримуються вендором та вбудованим NFC-модулем, а також NFC мітка без встановленого паролю, сумісна з NFC-модулем смартфону.

Інструкція користувача:

1. Встановіть додаток на смартфон та відкрийте його.
2. У випадку наявності акаунту введіть комбінацію з емейлу та паролю і натисніть на кнопку «Login».

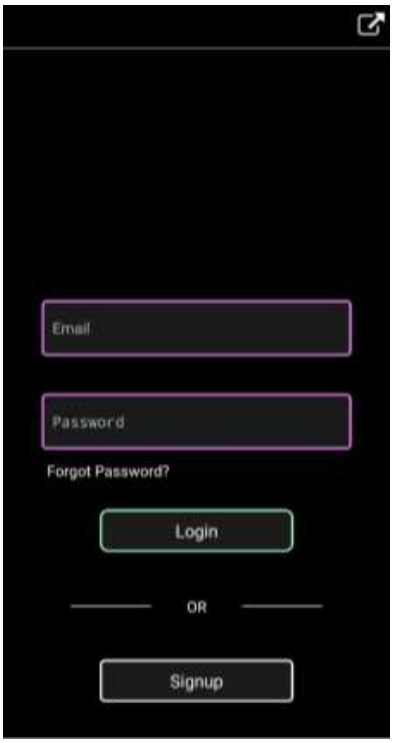


Рисунок Г.1 – Загальний вигляд початкової сторінки додатку

3. Якщо немає створеного профілю користувача, необхідно натиснути на кнопку «Signup» та ввести ім’я користувача, активну електронну пошту та вигадати пароль.

Рисунок Г.2 – Загальний вигляд сторінки введення інформації для реєстрації нового облікового запису

4. Після створення нового облікового запису, користувач опиниться на головній сторінці додатку.
5. Для додавання нового посилання на соціальну мережу, необхідно натиснути на кнопку «Add link» або зробити жест «вліво» та обрати соціальну мережу на яку необхідно додати та завантажити посилання на неї.

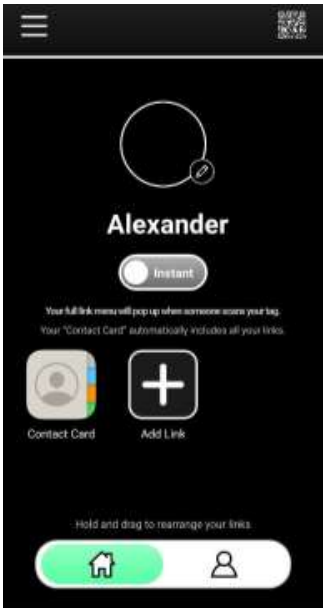


Рисунок Г.3 – Загальний вигляд основної функціональної частини додатку, де відображається інформація про користувача та його посилання

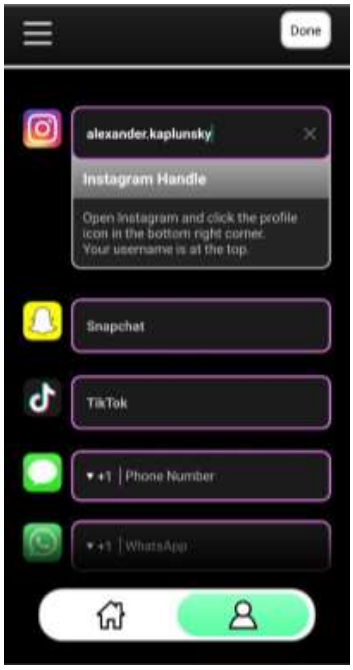


Рисунок Г.4 – Загальний вигляд інтерфейсу додавання нового посилання за допомогою переходу в меню по жесту «Вліво»

Рисунок Г.5 – Загальний вигляд меню додавання посилання з головної сторінки додатку

6. Після створення нового посилання, в головному меню з’явиться доданий елемент який можна буде редагувати.

Рисунок Г.6 – Загальний вигляд вигляд доданого посилання на соціальну мережу Instagram

7. Для запису даних на NFC-мітку необхідно відкрити бокове меню та обрати пункт «Activate Phone Tag».

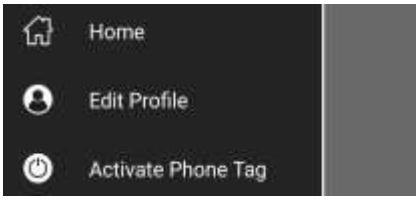


Рисунок А.7 – Загальний вигляд кнопки активації запису даних на NFC-тег

8. Піднесіть NFC-тег до пристрою та очікуйте звукового сигналу, який буде сигналізувати про успішний процес запису.
9. Після успішного запису на NFC-тег буде записано посилання на веб-профіль користувача. Для того щоб переглянути профіль необхідно активувати функцію NFC на пристрої та піднести тег в активну зону зчитування.

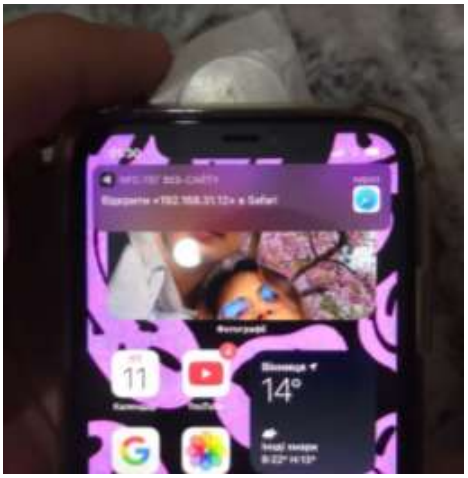


Рисунок А.8 – Загальний вигляд взаємодії пристрою з NFC-модулем та NFC-тегу з записаною на нього інформацією

10. Як тільки пристрій зчитає інформацію з носія, буде запропоновано відкрити посилання в браузері пристрою. За посиланням буде відображено профіль користувача.

Рисунок А.9 – Загальний вигляд інтерфейсу профілю користувача, посилання на який зберігається на NFC носію