Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: **Удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока**

Виконав студент ІІ курсу, групи УБ – 20м
Спеціальність 125 – «Кібербезпека»
Освітня програма – «Управління інформаційною безпекою»
Шадюк Валентин Сергійович
Керівник к.ф.-м.н., доц. Шиян А.А.
«__»_____ 2021р.
Опонент: к.т.н., доц., каф. ОТ Войцеховська О.В.
«__»_____ 2021р.

Допущено до захисту
Голова секції УБ кафедри МБІС
д.т.н., проф. Яремчук Ю.Є.
“_____”_____2021 р.

Вінниця – 2021 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

АНОТАЦІЯ

В даній магістерській роботі проаналізовано та практично реалізовано програмний засіб для удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

В ході виконання роботи було проаналізовано теоретичний матеріал з обраної галузі, досліджено переваги та недоліки біометричної ідентифікації в системах, зокрема за сканером відбитка пальця, qr-кодом та райдужною оболонкою ока.

На основі отриманих даних та розроблено алгоритму роботи додатку, було реалізовано програмний засіб комплексної ідентифікації за біометричними даними осіб. Розробка здійснювалась на мові програмування C# у середовищі Visual Studio.

SUMMARY

In this master's thesis the software tool for Improving the method of complex identification of the user of remote access on the basis of QR-code, fingerprint and cornea is analyzed and practically implemented.

In the course of the work the theoretical material from the chosen field was analyzed, the advantages and disadvantages of biometric identification in systems were investigated, in particular by the fingerprint scanner, qr-code and iris.

Based on the obtained data and developed the algorithm of the application, the software of complex identification based on biometric data of persons was implemented. Development was carried out in the C # programming language in Visual Studio.

ЗМІСТ

ВСТУП 7
1 АНАЛІЗ МЕТОДІВ АВТЕНТИФІКАЦІЇ КОРИСТУВАЧІВ НА ОСНОВІ БІОМЕТРИЧНИХ ДАНИХ ТА QR-КОДУ 9
1.1 Поняття біометричної ідентифікації користувачів та її особливості 9
1.2 Біометрична ідентифікація користувача на основі відбитка пальця 13
1.3 Біометрична ідентифікація користувача на основі рогівки ока 16
1.4 Аналіз систем автентифікації на основі qr-кодів 19
1.5 Висновки та постановка задач 26
2 РОЗРОБКА АГОРИТМУ РОБОТИ ДОДАТКУ НА ОСНОВІ УДОСКОНАЛЕННЯ МЕТОДУ КОМПЛЕКСНОЇ ІДЕНТИФІКАЦІЇ КОРИСТУВАЧА 27
2.1 Удосконалення методу комплексної ідентифікації користувача та розробка алгоритму роботи додатку 27
2.2 Розробка алгоритму розпізнавання відбитку пальця 30
2.3 Розробка алгоритму зчитування рогівки ока 33
2.4 Розробка алгоритму формування та зчитування qr-коду 35
2.5 Обґрунтування вибору мови та засобу програмування 41
2.6 Висновки до розділу 45
3 ПРОГРАМНА РЕАЛІЗАЦІЯ ДОДАТКУ ДЛЯ УДОСКОНАЛЕННЯ МЕТОДУ КОМПЛЕКСНОЇ ІДЕНТИФІКАЦІЇ КОРИСТУВАЧА 46
3.1 Проектування графічного інтерфейсу користувача 46
3.2 Програмна реалізація додатку 49
3.3 Реалізація користувацького інтерфейсу 58
3.4 Тестування елементів розробленого програмного додатку 64
3.5 Висновки до розділу 66
4 ЕКОНОМІЧНА ЧАСТИНА 67

4.1 Оцінювання комерційного потенціалу розробки ПЗ на комплексної ідентифікації користувача 67
4.2 Прогнозування витрат на виконання наукової роботи та впровадження її результатів 72
4.3 Прогнозування комерційних ефектів від реалізації результатів розробки 76
4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності 78
4.5 Висновки до розділу 82
ВИСНОВОК 83
ПЕРЕЛІК ПОСИЛАНЬ 85
ДОДАТКИ 90
Додаток А. Технічне завдання 91
Додаток Б. Лістинг Registration.xaml 95
Додаток В. Лістинг Login.xaml 106
Додаток Г. Алгоритм роботи 118
Додаток Д. Інтерфейс користувача 119
Додаток Е. Ілюстративний матеріал 121
Додаток Ж. Протокол перевірки 127

ВСТУП

Актуальність. Важливим елементом забезпечення цілісності конфіденційної інформації є захист від несанкціонованого доступу до ресурсів інформаційних систем, що викликає необхідність створення надійних і зручних систем контролю доступу.

Кожний користувач сучасних інформаційно-комунікаційних систем декілька разів на день зустрічається з процедурами ідентифікації та автентифікації. Ці процедури виконуються кожний раз, коли користувач вводить пароль для доступу до інформаційної системи, мережі, бази даних або при запуску прикладної програми.

В результаті їх виконання користувач або отримує доступ до певних ресурсів інформаційної системи, або не отримує.

Біометричні технології розпізнавання (ідентифікації, верифікації) особистості широко зарекомендували себе при вирішенні різних завдань, пов'язаних із забезпеченням підвищеного рівня безпеки доступу до інформації та різним матеріальним об'єктам. В основі технологій лежить властивість унікальності біометричної характеристики людини (індивідуума), яку використовують як ідентифікатор. Одними з таких характеристик є зображення райдужної оболонки ока та відбиток пальця.

Така розробка комплексної системи біометричної ідентифікації з додаванням qr-коду надає можливості розробити якісну систему ідентифікації осіб, що матиме високу достовірність розпізнавання та, відповідну, покращить практичність роботи у застосуванні загалом.

Оскільки, з плином часу дослідження в даній галузі набирає обертів, серед найбільш відомих дослідницьких груп даної сфери варто відмітити: Cambridge University, Велика Британія (J. Daugman); Michigan State University, США (J. Anil, A. Ross) [1 – 3].

Проте, найбільшу увагу технологіям біометричного розпізнавання зараз

приділяється з боку комерційних компаній, що створюють цілі інститути і напрямки для їх реалізації і доведення до ринку.

Таким чином, нові сценарії використання технологій біометричного розпізнавання створюють нові завдання, вирішення яких дозволить істотно підвищити рівень безпеки і зручності при використанні різних сервісів і послуг.

Мета дослідження. Метою роботи є удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

Задачами дослідження є:

– здійснити аналіз методів автентифікації користувачів на основі біометричних даних та qr-коду;

– провести удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока та розробити алгоритм програмного засобу.

– здійснити програмну реалізацію додатку;

– обґрунтувати економічну доцільність розробки.

Об’єкт дослідження – процеси автентифікації користувачів.

Предмет дослідження – методи автентифікації користувачів.

Новизна роботи – полягає в удосконаленні комплексного методу віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

Практична цінність роботи полягає в розробці програмного засобу для біометричної ідентифікації користувача на основі удосконалення комплексного методу віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

1 АНАЛІЗ МЕТОДІВ АВТЕНТИФІКАЦІЇ КОРИСТУВАЧІВ НА ОСНОВІ БІОМЕТРИЧНИХ ДАНИХ ТА QR-КОДУ

1.1 Поняття біометричної ідентифікації користувачів та її особливості

За останні два десятиліття біометричні технології зробили великий крок уперед. Багато в чому сприяло поширення мікропроцесорних технологій. Ще у 80-ті роки систему контролю доступу, що використовує біометричні характеристики людини, можна було побачити лише у фантастичних фільмах. Сьогодні ж використання в системах контролю та управління доступом (СКУД) біометричних сканерів практично не ускладнює систему безпеки, і їхня вартість для деяких біометричних методів дуже низька. Більше того, близько третини ноутбуків виходить зараз із вбудованою системою зчитування відбитка пальців, а якщо в ноутбуці є відеокамера, на нього можна встановити систему розпізнавання людини [4].

Ідентифікація на основі біометричних даних – це засіб автоматичного розпізнавання особи на базі унікальних фізичних або поведінкових параметрів.

Вважається, що ця методика незабаром стане головним конкурентом цифровим сертифікатам та смарт-картам через такі переваги [5]:

1. Для біометричної ідентифікації достатньо фізичних параметрів людини і не потрібні жодні файли (які можна скопіювати) або паролі (які можна зламати). Тобто ідентифікація відбувається не за принципами «щось знаю» (пароль) та «чимось володію» (електронна картка), а за принципом «що я є» (фізичні параметри).

2. Унікальні людські ознаки хороші тим, що їх важко підробити, важко залишити фальшивий відбиток пальця за допомогою свого власного або зробити райдужну оболонку свого ока схожою на чиюсь іншу.

3. На відміну від паперових ідентифікаторів (паспорт, права водія, посвідчення особи), від пароля або персонального ідентифікаційного номера

(ПІН), біометричні характеристики не можуть бути забуті або втрачені, їх завжди легко «пред'явити».

Одним із головних показників розвитку біометрії є стандартизація даної області [16 – 7]. Вона йде за декількома напрямками: як для окремих областей, так і для всіх способів біометричної ідентифікації в цілому. Розробкою стандартів з біометрії займаються державні структури, міжнародні організації зі стандартизації та незалежні консорціуми, які є об'єднанням кількох виробників.

У біометричних системах ідентифікаційними є індивідуальні особливості людини, які в даному випадку називаються біометричними ознаками. Ідентифікація проводиться за рахунок порівняння отриманих біометричних параметрів і шаблонів, що зберігаються в основі. Залежно від характеристик, які використовуються, біометричні системи діляться на статичні та динамічні. Статична біометрія ґрунтується на даних (шаблонах), отриманих шляхом вимірювання анатомічних особливостей людини (відбитки пальців, візерунок райдужки ока тощо), а динамічна – на аналізі дій людини (голос, параметри підпису, її динаміка)).

В даний час активно використовуються такі біометричні ознаки [8]:

– відбитки пальців;

– геометрична форма кисті руки;

– форма та розміри особи;

– особливості голосу;

– візерунок рогівки та сітківки очей.

На рис. 1.1 наведено приклади таких біометричних даних: відбиток пальця, обличчя, Рогівкаока, розташування вен на лицьовій стороні долоні, спектрограма голосу, термограма особи, 3-мірне зображення обличчя, динаміка набору на клавіатурі, (і) – ДНК, відповідно.

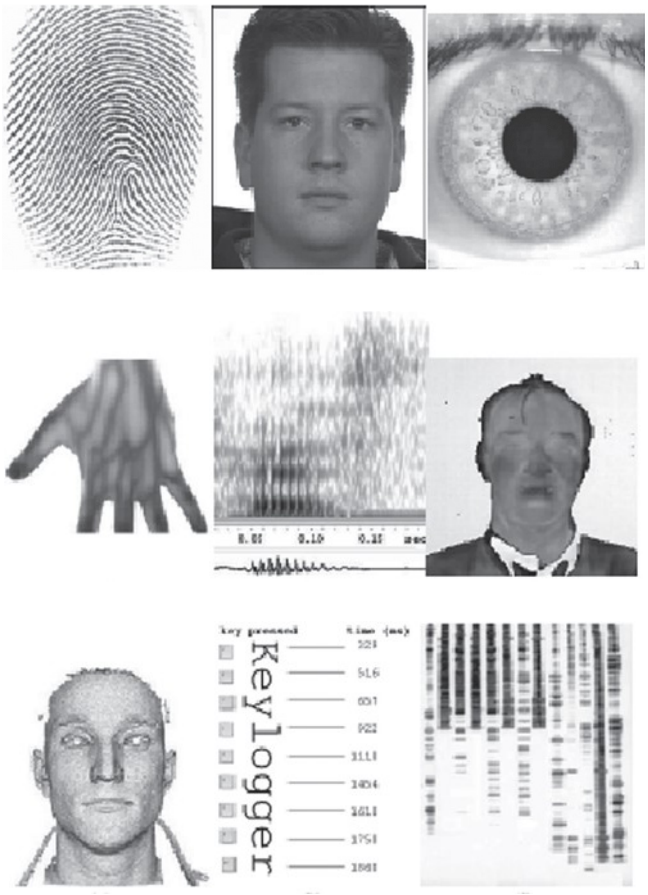


Рисунок 1.1 – Приклади різноманітних біометричних даних (за [9])

Основні проблеми та завдання в області біометричної автентифікації.

З недоліків біометричної автентифікації можна відзначити такі [10 – 11]:

– відсутність добре підготовленого професійного персоналу;

– висока вартість деяких пристроїв біометричної автентифікації;

– складність налаштування таких систем;

– протидія з боку співробітників, оскільки керівництво отримує можливість контролювати всі їх переміщення та фактично здійснювати контроль робочого часу;

– неможливо застосовувати для ідентифікації людей із деякими фізичними вадами;

– можливість користуватися муляжем (наприклад, відбитка пальця);

– нестандартне зростання (менше 1,55 м або більше 2,1 м). Сканування особи може стати скрутним.

Виходячи з аналізу недоліків біометричної ідентифікації, сформулюємо основні проблемні завдання, які необхідно розв'язати [12]:

– підготовка висококваліфікованого персоналу;

– мінімізування вартості біометричних пристроїв, створення бюджетної лінійки товарів;

– спрощення налаштування біометричних систем;

– робота з персоналом щодо популяризації даного напрямку;

– адаптація біометричних систем для ідентифікації людей із деякими фізичними вадами;

– удосконалення систем біометричної ідентифікації для виключення можливості скористатися муляжем.

Біометричні технології ідентифікації мають великі перспективи розвитку. При використанні систем на основі біометричних даних процедури доступу стають швидше, безпечніше та простіше. Але, незважаючи на величезну кількість переваг, біометричні технології мають низку складнощів та проблем. Так, необхідно вирішувати питання щодо використання пристроїв біометричної ідентифікації людьми з деякими фізичними вадами, щодо підготовки професійних кадрів, зменшення вартості пристроїв та ін. [13].

Впровадження біометричних паспортів є пріоритетним напрямком у розвитку міжнародних відносин. Використання біометричних технологій у паспортній системі дозволяє спростити процедуру ідентифікації, знизити ймовірність суб'єктивної помилки контролера, прискорити процес прикордонного контролю. В електронний паспорт може бути записано до 19 біометричних характеристик людини [14 – 15].

На підставі проведеного аналізу, можна зробити висновок про те, що найточнішим та найкращим методом біометричної ідентифікації для використання в електронній паспортній системі є ідентифікація за рогівкою ока. Як додатковий параметр можна використовувати відбиток пальця або геометричну форму кисті руки.

1.2 Біометрична ідентифікація користувача на основі відбитка пальця

Відбитки всіх пальців кожної людини унікальні на малюнку папілярних ліній і різняться навіть у близнюків. Відбитки пальців не змінюються протягом усього життя дорослої людини, вони легко і легко пред'являються при ідентифікації [16].

Якщо один з пальців пошкоджений, для ідентифікації можна скористатися «резервним» відбитком (відбитками), відомості про які зазвичай вносяться в біометричну систему при реєстрації користувача.

Обробка ідентифікаторів. Для отримання відомостей про відбитки пальців використовуються спеціалізовані сканери. Відомі три основні типи сканерів відбитків пальців: ємнісні, прокатні, оптичні.

Ємнісні сканери найдешевші, проте не відрізняються ні практичністю, ні довговічністю. Оскільки зображення відбитка у цих сканерах формується за рахунок різниці електричних потенціалів різних ділянок шкіри, ці сканери надзвичайно чутливі до залишкової статичної електрики. Вони виходять з ладу відразу після того, як їх торкнувся чоловік, чиї руки були наелектризовані, наприклад, через носіння одягу з вовняної або шовкової тканини. Крім того, якість зображення відбитків, що формується ємнісними сканерами, вкрай невелика [17 – 18].

Найдосконалішу технологію ідентифікації за відбитками пальців реалізують оптичні сканери. Вони трохи дорожчі за сканери інших типів, але позбавлені їх численних недоліків, довговічні і тому економічні, зручні і прості у використанні. Зображення відбитків характеризується високою якістю.

Прокатні сканери займають середнє положення. Вони зображення відбитка формується при «прокатуванні» відбитка через вузьке вікно сканера (звідси і назва), після чого цілісне зображення ідентифікатора «зшивається» з окремих кадрів, отриманих під час описаної процедури. Тому від користувача

такого сканера потрібно постійно дотримуватись одноманітності у швидкості та манері «прокатування» відбитків, що досить складно [19].

Нові дослідження довели, що оптичні сканери відбитків пальців абсолютно безпечні в антибактеріальному відношенні.

Пристрій, який зіставляє відбитки пальців з особистістю людини, є електронним пристроєм, званим датчиком відбитків пальців. Датчик відбитків пальців виконує сканування пальця людини, обробляє відбиток та зіставляє біометричні дані відбитка з інформацією, що міститься у ньому.

Ідентифікація за відбитком пальця часто використовується для безпеки. Наприклад, у захищених приміщеннях може використовуватися система безпеки, заснована на системі відбитків пальців, щоб забезпечити доступ лише певним людям. Є деякі сейфи, які покладаються на подібну технологію та відкриваються лише для людини із збереженим відбитком пальця [20].

Іноді використовується автентифікація по відбитку пальця, коли великій кількості людей потрібен доступ до одного і того ж входу протягом одного й того ж періоду часу. Деякі великі офісні будівлі використовують автентифікацію за відбитками пальців. Це дозволяє багатьом співробітникам входити в будинок у ключові моменти дня, наприклад, вранці та після обідньої перерви, без необхідності зупинятися, щоб подати ідентифікаційні матеріали працівникам служби безпеки.

Ідентифікація за відбитками пальців також є важливою для слідчих у кримінальних справах. Відбитки пальців, знайдені на місці злочину, можуть бути проаналізовані на наявність відбитків пальців, що зберігаються в базі даних поліції.

Позитивні збіги за відбитками пальців можуть допомогти детективам встановити особу особи, яка вчинила злочин, навіть якщо залишилося мало чи ні інших доказів [21].

У кожному відбитку пальця людини можна визначити два типи ознак – глобальні й локальні.

Глобальні ознаки (Рис.1.2) – ті, які людина можна побачити неозброєним оком [22]:

– папілярний візерунок;

– область візерунка – виділений фрагмент відбитка, в якому локалізовані всі глобальні ознаки;

– ядро або центр – точка, локалізована в середині відбитка або певної виділеної області;

– пункт «дельта» – початкова точка. Це місце, в якому відбувається;

– поділ або поєднання борозенок папілярних ліній, чи дуже коротка борозенка (може доходити до крапки).

Рисунок 1.2 – Глобальні ознаки відбитку пальця (за [22])

Тип лінії – дві найбільші лінії, що починаються як паралельні, а потім розходяться і огинають всю область образу [23].

Лічильник ліній – це число ліній на області образу, чи між ядром і пунктом «дельта».

Є наступні типи папілярних візерунків: візерунки типу «петля» (права, ліва, подвійна, центральна); візерунки типу «дельта» чи «дуга» (проста і гостра), візерунки типу «спіраль» (центральна і змішана).

Інший тип ознак – локальні ознаки. Їх називають мінуції (рис.1.3) (особливостями або особливими точками) – унікальні для кожного відбитку

пальця ознаки, які визначають пункти зміни структури папілярних ліній (роздвоєння, закінчення, розрив тощо), орієнтацію папілярних ліній і координати в цих пунктах. Кожен відбиток може містити до 70 і більше мінуцій.

Рисунок 1.3 – Локальні ознаки відбитку пальця (мінуції) (за [24])

Практика показує, що відбитки пальців у різних людей можуть мати окремі однакові глобальні ознаки, проте абсолютно неможливо наявність однакових мікровізерунків мінуцій. Тому глобальні ознаки використовують з метою поділу бази даних на класи і на етапі автентифікації. На другому етапі розпізнавання користуються вже локальними ознаками.

1.3 Біометрична ідентифікація користувача на основі рогівки ока

Сканування рогівки ока - це біометрична технологія на основі очей, яка використовує унікальні візерунки на кровоносних судинах сітківки У розпізнаванні рогівкиока використовуються технології відеокамер із тонким ближнім інфрачервоним світлом для отримання зображень складних та деталізованих структур райдужної оболонки, видимих зовні. Цифрові шаблони, закодовані на основі цих шаблонів за допомогою математичних і статистичних алгоритмів, дозволяють ідентифікувати людину або когось, хто вдає цю

людину. [25] Бази даних зареєстрованих шаблонів шукаються механізмами зіставлення зі швидкістю, що вимірюється в мільйонах шаблонів в секунду на (одноядерний) ЦП, і з низькою частотою помилкових збігів.

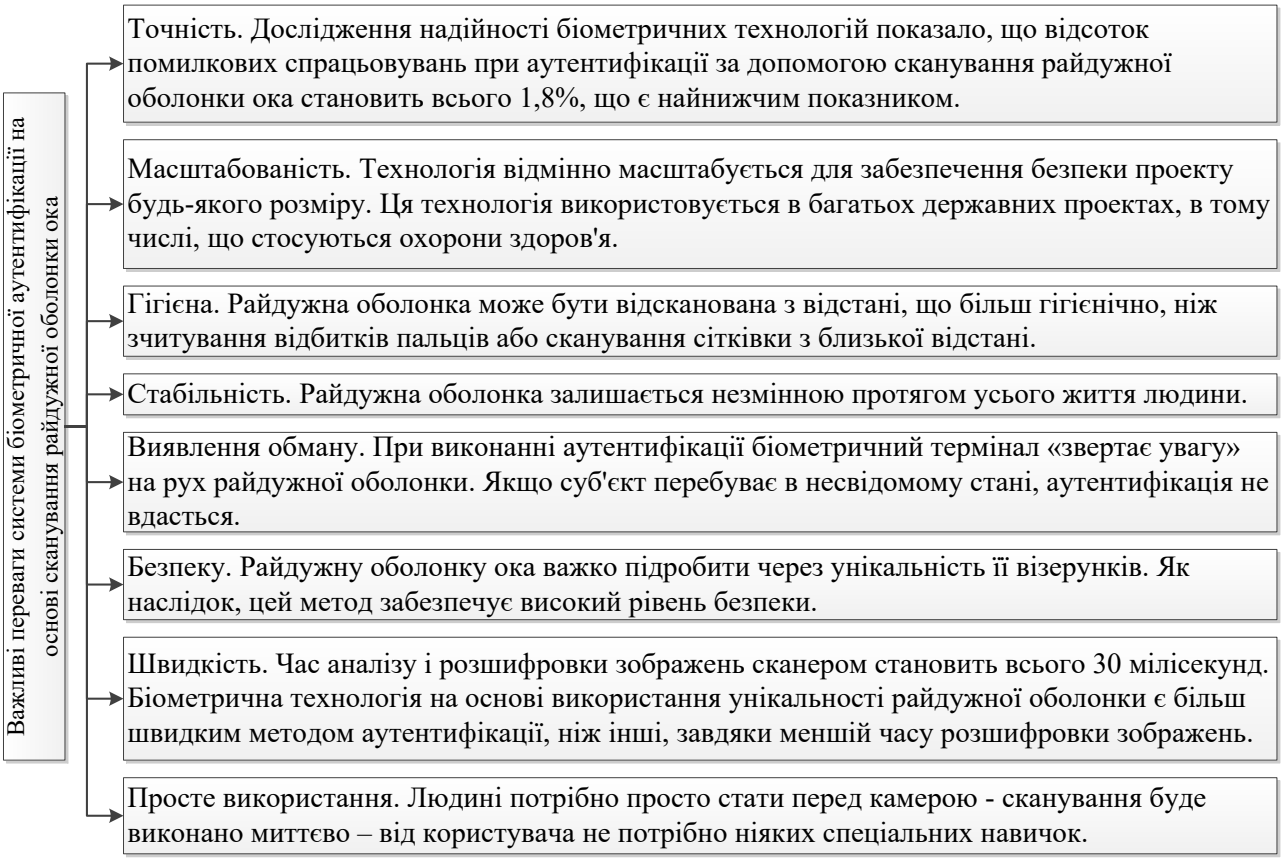


Рисунок 1.4 – Переваги системи біометричної ідентифікації на основі сканування рогівки ока (за [26])

Принцип роботи. Спочатку система повинна визначити внутрішню та зовнішню межі рогівки(зіниці та лімба) на зображенні ока. Подальші підпрограми виявляють та виключають повіки, вії та дзеркальні відображення, які часто закривають частини райдужної оболонки. Набір пікселів, що містить тільки райдужну оболонку, нормалізований за допомогою моделі гумового листа для компенсації розширення або звуження зіниці, потім аналізується для отримання бітового шаблону, що кодує інформацію, необхідну для порівняння двох зображень райдужної оболонки [27 – 28].

У разі алгоритмів Даугмана використовується вейвлет-перетворення Габора. Результатом є набір комплексних чисел, які несуть інформацію про

локальну амплітуду та фазу діафрагми. У алгоритмах Даугмана більшість інформації про амплітуді відкидається, і 2048 бітів, що становлять діаграму райдужної оболонки, складаються з інформації про фазу (складні знакові біти проекцій вейвлетів Габора). Відмова від інформації про амплітуду гарантує, що на шаблон значною мірою не впливають зміни освітленості або посилення камери, та сприяє довгостроковому використанню біометричного шаблону.

Для ідентифікації (порівняння шаблонів один-до-багатьом) або перевірки (порівняння шаблонів один-до-одному) [29] шаблон, створений шляхом візуалізації рогівкиока, порівнюється зі збереженими шаблонами в базі даних. Якщо відстань Хеммінга нижче порога прийняття рішення, позитивна ідентифікація була фактично проведена через статистичної крайньої неймовірності того, що дві різні людини могли б випадково погодитися («зіштовхнутися») у такій великій кількості біт, враховуючи високу ентропію шаблонів райдужної оболонки.

Переваги. Рогівкаока описується як ідеальна частина людського тіла для біометричної ідентифікації з кількох причин [30 – 31]:

Це внутрішній орган, який добре захищений від пошкоджень та зносу дуже прозорою та чутливою мембраною (рогівкою). Це відрізняє його від відбитків пальців, які важко розпізнати після багатьох років ручної праці. Рогівкав основному плоска, і її геометрична конфігурація контролюється лише двома додатковими м'язами (зіницями сфінктера та зіницями розширювача), які контролюють діаметр зіниці. Це робить форму рогівкинабагато більш передбачуваною, ніж, наприклад, обличчя.

Рогівкамає тонку текстуру, яка, як і відбитки пальців, визначається випадковим чином під час ембріональної вагітності. Як і відбиток пальця, дуже складно (якщо не неможливо) довести, що Рогівкаунікальна. Однак існує так багато факторів, що впливають на формування цих текстур (Рогівката відбиток пальця), що ймовірність помилкового збігу для будь-якої з них надзвичайно мала. Навіть генетично ідентичні особини (і ліве і праве очі однієї й тієї ж

людини) мають повністю незалежні текстури райдужної оболонки. Сканування рогівкиока схоже на фотографування та може виконуватись на відстані від 10 см до кількох метрів. Ідентифікованій особі немає необхідності торкатися будь-якого обладнання, до якого недавно доторкнувся незнайомець, тим самим усуваючи заперечення, яке було висунуто в деяких культурах проти сканерів відбитків пальців, де палець повинен торкатися поверхні, або сканування сітківки ока, де око потрібно піднести дуже близько до окуляру (як і мікроскоп).[32]

Використовується в комерційних цілях алгоритм розпізнавання рогівкиока, код IrisCode Джона Даугмана, має безпрецедентну частоту хибних збігів (краще, ніж 10 −11, якщо використовується поріг відстані Хеммінга 0,26, що означає, що дозволено до 26% бітів у двох кодах Iris. не погоджуватися через шум зображення, відбитків і т. д., але при цьому заявляти, що вони збігаються). [33] Незважаючи на те, що існують деякі медичні та хірургічні процедури, які можуть вплинути на колір та загальну форму райдужної оболонки, тонка текстура залишається напрочуд стабільною протягом багатьох десятиліть. Деяким визначенням рогівкивдалося досягти успіху у період близько 30 років.

Розпізнавання рогівкиока працює з прозорими контактними лінзами, окулярами та сонцезахисними окулярами без дзеркала.

1.4 Аналіз систем автентифікації на основі qr-кодів

Структура QR-коду є матрицею, що містить двійкову інформацію. QR-код закодовано багато різної інформації [34 – 35]. Крім даних, що закодував користувач у коді містяться:

– версія коду;

– код маски;

– рівень корекції помилок.

Також на QR-коді є необхідні поля, вони не несуть закодованої

інформації, а містять інформацію про версію коду, про тип його кодування та додаткову інформацію для його декодування [36]:

– пошукові візерунки;

– вирівнюючі візерунки;

– смуги синхронізації;

– код маски та рівня корекції;

– код версії (починаючи з 7 версії);

– тиха зона.

Рисунок 1.5 – Структура QR-коду (за [37 – 38])

Тиха зона є рамкою з порожніх модулів. Ширина такої рамки складає 4 модулі. Пошукові візерунки розташовані в кутах QR-коду, крім правого нижнього, і є три мішені, за якими визначається орієнтація і розташування QR-коду. Пошукові візерунки складаються з квадрата чорних модулів розміром 3x3 модуля навколо цього квадрата рамка з білих модулів, шириною 1 модуль і рамка з чорних модулів, так само шириною 1 модуль. Від іншого коду мета захищається половиною рамки з білих модулів шириною один модуль.

Вирівнюючі візерунки використовуються з другої версії коду для точного зчитування коду. Такі візерунки розміщуються на різних позиціях залежно від

версії коду. Вирівнюючі візерунки не можуть накладатись на пошукові візерунки. Вирівнюючі візерунки є рамкою з чорних модулів розміром 5х5 і шириною 1 модуль, в центрі візерунка знаходиться 1 чорний модуль, оточений рамкою з білих модулів [39].

Смуги синхронізації використовуються для визначення розміру модулів. Смуги розташовуються по горизонталі від верхнього лівого пошукового візерунка (від нижнього правого кута) до верхнього правого пошукового візерунка (до нижнього лівого кута) і вертикальна смуга йде від лівого верхнього пошукового візерунка (від нижнього правого кута) до лівого нижнього візерунка (до верхнього правого кута). Смуги йдуть безпосередньо від пошукового візерунка, обминаючи білу рамку. Смуги синхронізації не можуть накладатися на візерунки, що вирівнюють. Смуги є чергуванням білих і чорних модулів.

Код маски та рівня корекції розташований поруч із пошуковими візерунками [40]:

– під правим верхнім (8 модулів);

– праворуч від лівого нижнього (7 модулів);

– дублюються з боків лівого верхнього (8 модулів) з пробілом на 7 модулі (там, де проходять смуги синхронізації).

Код версії використовується пристроєм, що зчитує, для визначення версії коду [41]. Знаходиться ліворуч від верхнього правого пошукового візерунка та дублюється зверху від нижнього лівого пошукового візерунка. Дублювання відбувається за правилом: код версії у верхнього правого пошукового візерунка відображають дзеркально та повертають на 90 градусів проти годинникової стрілки.

На відміну від старого одномірного штрих-коду, який зчитується механічно, за допомогою маленького пучка світла, QR-код зчитується за допомогою двомірного цифрового датчика зображень (фотокамерою телефону, web-камерою), а потім у цифровій формі аналізується програмованим

процесором. Процесор знаходить три пошукові візерунки в кутах цифрового образу QR-коду і нормалізує зображення. Потім зчитуються модулі коду і перетворюються на двійкові числа.

У QR-код можна закодувати різну інформацію, і для цього од повинен підтримувати різні типи кодування, наприклад [42]: числове; символьно-числове; байтове; кандзі.

Числове кодування. Кодування в цьому режимі відбувається за допомогою розбиття всієї послідовності символів на рівні частини по 3 символи, після чого кожна група переводиться в двійкове число і записується в 10 біт інформації, після чого додається до загальної бітової послідовності [43]. Якщо кількість символів кодується не кратно 3, то якщо в кінці залишається 1 символ, то його кодують 4 бітами, а якщо залишилося 2 символи, то його кодують 7 бітами.

Таким чином, число «1506» перетворюється на рядок:

"00100101100000000110".

$1506 = 150 + 6$ $1506_{10} = 0010010110 + 00000110$

$150_{10} = 0010010110$ $1506_{10} = 001001011000000110$

$6_{10} = 00000110$

Рисунок 1.6 – Числове кодування (за [44])

Літерно-числове кодування. На відміну від числового кодування для символьного кодування послідовність символів розбивається по 2 символи і кодується в 11 біт інформації. У кожній групі кожен символ кодується згідно з таблицею. «Значення символів у буквенно-цифровому кодуванні» [45]. Добуток значення першого символу на 45 складається зі значенням другого символу, після чого записується в 11-бітове двійкове число і додається до загального рядка біта. Якщо в кінці рядка залишається один символ, його значення кодується 6 бітами. Таким чином, рядок «sourse» буде записано як «10100000100101011000010100010101 0»

sourse = so + ur + se
so = 28 * 45 + 24 = 1260 + 24 = 1284 = 10100000100
ur = 30 * 45 + 27= 1350 + 27= 1377= 10101100001
se = 28*45 + 14= 1260+ 14 = 1274= 10011111010
sourse = 10100000100 + 10101100001 + 10011111010
sourse = 10100000100101011000011001111101 0

Рисунок 1.7 – Літерно-числове кодування (за [46])

0	0	A	10	K	20	U	30	+	40
1	1	B	11	L	21	V	31	-	41
2	2	C	12	M	22	W	32	.	42
3	3	D	13	N	23	X	33	/	43
4	4	E	14	O	24	Y	34	:	44
5	5	F	15	P	25	Z	35		
6	6	G	16	Q	26	Пробел	36		
7	7	H	17	R	27	$	37		
8	8	I	18	S	28	%	38		
9	9	J	19	T	29	*	39		

Рисунок 1.8 – Значення символів у буквенно-цифровому кодуванні (за [47])

Байтове кодування. Байтове кодування – це універсальний тип кодування, яким можна закодувати будь-яку послідовність даних. Що стосується байтовим кодуванням, дані перетворюються на двійкову систему незмінному вигляді. При такому кодуванні рядок «УрДПУ» виглядатиме так: «1000010001110001000001000001001110000011111100001000011».

Кандзі кодування [48]. В основі кодування Кандзі, як і у випадку з буквено-числовим кодуванням, лежить візуально сприймається таблиця ієрогліфів з їх кодами. Для японської мови основне значення мають дві таблиці символів: JIS 0208:1997; JIS 0212:1990.

Версія коду. Коди поділяються на версії від 1 до 40. Кожна версія має особливості у конфігурації та кількості модулів, що становлять QR-код. Розмір 1 версії – 21×21, а версії 40 – 177×177. З кожною версією розмірність коду

збільшується на 4 модулі.

Для кожної версії коду визначено ємність з урахуванням рівня корекції помилок. Чим більше інформації необхідно закодувати і чим більший рівень корекції використовується, тим більша версія коду буде потрібна. Більшість сучасних QR-генераторів автоматично підбирають версію коду з урахуванням рівня корекції помилок.

Код маски. Маска використовується для підвищення надійності сканування QR-коду, за допомогою формування такої матриці коду, на якій було б якнайменше суміжних модулів однакового кольору. Для вибору маски формуються 8 матриць з однаковим вмістом. На кожній матриці окрема маска, яка підсумовує вихідну матрицю QR-коду за правилом «що виключає АБО» (xor). Отримані матриці піддають математичної обробки, де ведеться підрахунок "штрафного" коефіцієнта, після чого вибирають одну матрицю. Для вибору оптимальної матриці значення "штрафного" коефіцієнта має бути мінімальним [49].

Нижче наведено таблицю масок та відповідних математичних формул для підрахунку «штрафного» коефіцієнта (таб.1.1).

Таблиця1.1 – Математичні формули масок QR-коду (за [50])

Маска QR-коду	Математичні формули
000	(i + j) mod 2 = 0
001	i mod 2 = 0
010	j mod 3 = 0
011	(i + j) mod 3 = 0
100	((i div 2) + (j div 3)) mod 2 = 0
101	(i * j) mod 2 + (i j) mod 3 = 0
110	((i * j) mod 2 + (i j) mod 3) mod 2 = 0
111	((i + j) mod 2 + (i j) mod 3) mod 2 = 0

У математичній формулі маски використовуються номери рядків (j) та номери стовпців (i). У кожній матриці розглядається кожний інформаційний модуль. Якщо при застосуванні формули на модулі співвідношення правильне, модуль інвертується. У структурі QR-коду також є механізм підвищення надійності зберігання інформації, так званий рівень корекції помилок. Рівень

корекції помилок будується на основі алгоритму Ріда-Соломона [16, с.31]. Існує 4 види рівня корекції, які задаються під час створення QR-коду. Рівень корекції помилок сприяє збільшенню надійності зберігання інформації, що дозволяє при втраті частини коду відновити зашифровану інформацію, наприклад найвищий рівень корекції помилок (H), дозволяє прочитати код з втратою 30%. Однак рівень корекції призводить до збільшення розміру самого коду (Таб.3). Нижче наведено таблицю рівнів корекції помилок, на якій показано припустимі пошкодження для коректного зчитування QR-коду (таб.1.2).

Таблиця 1.2 – Допустимі пошкодження для коректного зчитування QR-коду (за [51])

Рівень корекції	Допустимий відсоток порушень
L	до 7%
M	до 15%
Q	до 25%
H	до 30%

У таблиці 3 наведено різні типи кодування, версії коду, рівні корекції помилок та розміри QR-коду, а також співвідношення у вигляді кількості символів, яку можна закодувати QR-код [11]. Наприклад, у QR-код версії 3 при буквенно-числовому кодуванні та рівнем корекції помилок «Q» можна закодувати 27 символів.

Таблиця 1.3 – Розміри QR-коду з урахуванням рівня корекції помилок та версії коду (за [52])

Версія QR-коду	Розмір QR-коду (модуль)	Рівень корекції помилок	Максимальна кількість символів із врахуванням корекції помилок			
			Числове кодування	Буквено-числове кодування	Байтове кодування	Кандзиі кодування
1	21x21	L	41	25	17	10
		M	34	20	14	8
		Q	27	16	11	7
		H	17	10	7	4

2	25x25	L	77	47	32	20
		M	63	38	26	16
		Q	48	29	20	12
		H	34	20	14	8
3	29x29	L	127	77	53	32
		M	101	61	42	26
		Q	77	47	32	20
		H	58	35	24	15
4	33x33	L	187	114	78	48
		M	149	90	62	38
		Q	111	67	46	28
		H	82	50	34	21

Таким чином, аналіз структури QR-коду показав, що QR-код – це зручний, ефективний засіб для зберігання та використання інформації різних видів, що має механізм підвищення надійності зберігання інформації.

1.5 Висновки та постановка задач

Отже, в даному розділі було проведено теоретичний огляд галузі, в якій проводиться розробка. У даній роботі представлено основні поняття систем автентифікації та способи такого захисту, також проведено аналіз застосування сучасних біометричних засобів ідентифікації, зокрема, для підвищення надійності та стійкості до атак в системах автентифікації користувачів.

В результаті проведеного аналізу теоретичного матеріалу та виходячи з мети і актуальності теми, були поставлені наступні задачі подальшої роботи:

– провести удосконалення та розробити алгоритм програмного засобу.

– здійснити програмну реалізацію додатку;

– обґрунтувати економічну доцільність розробки.

В результаті виконання поставлених завдань, планується досягти основної мети роботи, а саме розробки та реалізації методу комплексної ідентифікації користувача на основі QR-коду, відбитку пальця та рогівки ока

2 РОЗРОБКА АГОРИТМУ РОБОТИ ДОДАТКУ НА ОСНОВІ УДОСКОНАЛЕННЯ МЕТОДУ КОМПЛЕКСНОЇ ІДЕНТИФІКАЦІЇ КОРИСТУВАЧА

2.1 Удосконалення методу комплексної ідентифікації користувача та розробка алгоритму роботи додатку

Типова система автентифікації містить у собі такі складові ключові елементи:

– суб’єкт (користувач), за даними якого здійснюється процедура автентифікації;

– особлива відмінна риса даного суб’єкта – його певна біометрична характеристика;

– адміністратор системи автентифікації, що керує роботою системи та несе відповідальність за усі дії, що у ній відбуваються;

– алгоритм (принцип) роботи системи автентифікації та механізм управління системою автентифікації, тобто управління доступом.

Нині, дедалі більше використовується багатофакторна або комплексна автентифікація користувачів, що ґрунтується на спільному використанні певних факторів автентифікації, що значним чином підвищує рівень надійності та захищеності системи, оскільки при наявності лише одного ідентифікатора особи не можна цілковито бути впевненим у надійності його достовірності.

Для кожного випадку застосування біометричної ідентифікації, лише при застосуванні одного ідентифікатора існує ризик його підробки, примусового використання, втрати і т.п.

Саме тому, ефективним методом усунення даного недоліку є поєднання декількох біометричних ідентифікаторів у комплексі, що може значно підвищити рівень захищеності системи та надійності, власне, самого процесу ідентифікації користувача.

Отже, в даній роботі автором запропоновано удосконалення методу ідентифікації користувача саме на сонові використання таких ідентифікаторів у комплексі. Розглянемо більш детальніше алгоритм роботи такої системи.

Крок 1. Запуск виконуваного файлу.

Крок 2. Якщо користувач зареєстрований в системі – перехід до кроку 3. Якщо користувач не має облікового запису – перехід до кроку 9.

Крок 3. Здійснення авторизації користувача.

Введення унікального логіну користувача, введення зразка відбитку пальця та зчитування рогівки ока, формування qr-коду.

Крок 4. Підтвердження виконаних дій авторизації.

Крок 5. Якщо усі дані введенні коректні – перехід до кроку 6.

Якщо дані введені некоректно – повернення до кроку 3.

Крок 6. Перевірка достовірності введених даних, зв'язок з базою даних.

Крок 7. Якщо перевірка завершена вдало – перехід до кроку 8.

Якщо у доступі відмовлено – користувач може спробувати здійснити повторну авторизацію (крок 3).

Крок 8. Отримання доступу до захищуваної системи.

Крок 9. Здійснення процесу реєстрації в системі.

Введення унікального логіну користувача, введення зразка відбитку пальця та зчитування рогівки ока, формування qr-коду.

Крок 10. Підтвердження виконаних дій реєстрації.

Крок 11. Якщо усі дані введенні коректні – перехід до кроку 12.

Якщо дані введені некоректно – повернення до кроку 9.

Крок 12. Збереження даних нового користувача в системі.

Повідомлення про успішну реєстрацію та, відповідно, перехід до кроку 8.

Алгоритм представлений схематично на рисунку 2.1.

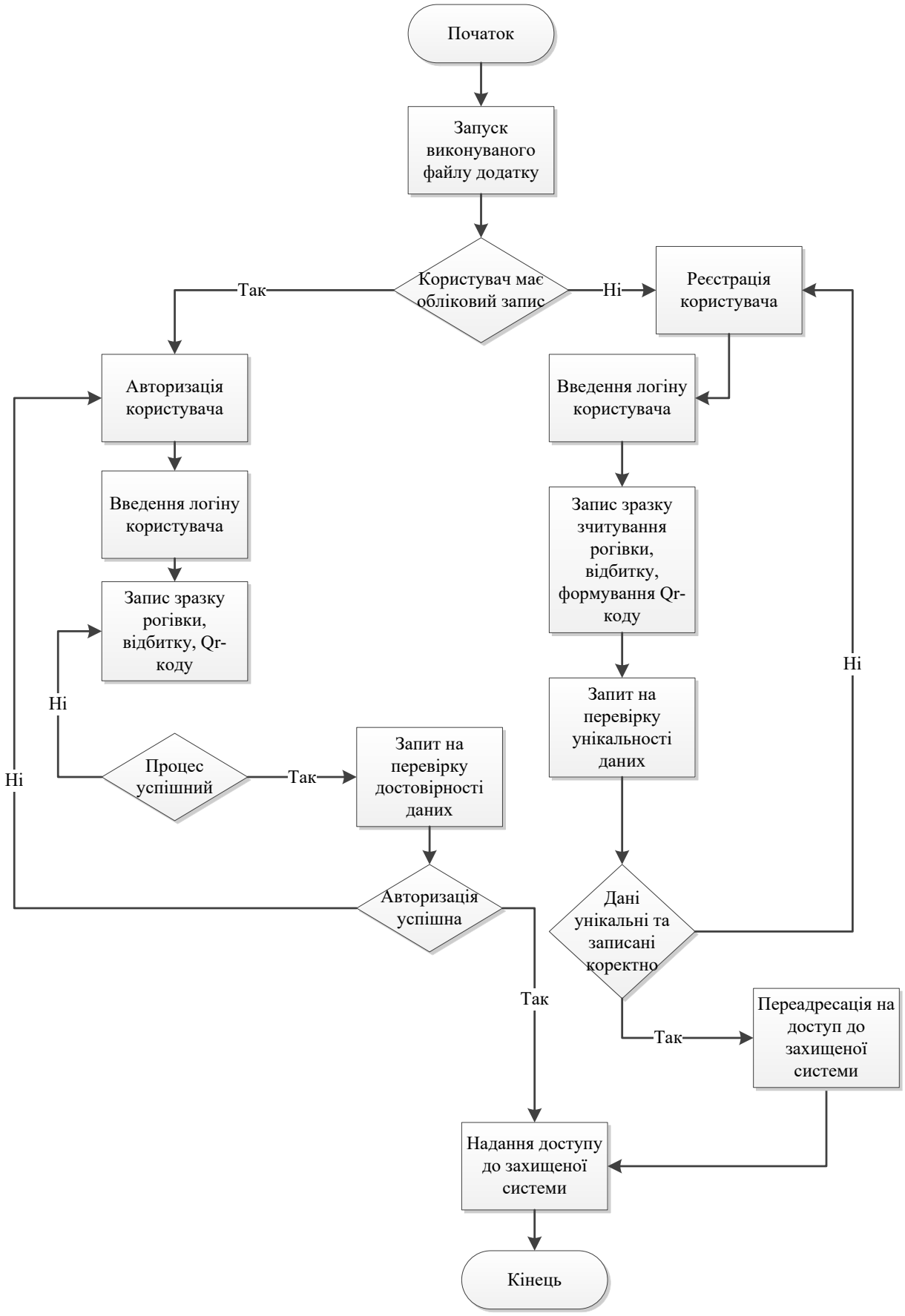


Рисунок 2.1 – Алгоритм роботи додатку

Таким чином, в ході написання даного підрозділу був обґрунтований та розроблений алгоритм додатку для комплексної автентифікації до системи захисту з використанням ідентифікації за рогівкою ока, сканом відбитка пальця та qr-кодом.

2.2 Розробка алгоритму розпізнавання відбитку пальця

Для розробки алгоритму роботи програмного засобу на основі використання скану відбитка пальця для реалізації процесу ідентифікації особи, розглянемо більш детальніше особливості структури такого відбитку.

В системах ідентифікації користувачів за відбитком пальця наявний модуль попередньої обробки сканованого зображення. Складність обробки залежить від якості отримуваного зображення зі сканера та ефективності методу виділення особливих точок. Основними етапами роботи із зображенням в процесі ідентифікації є: бінаризація, скелетизація, виділення особливих точок [6, 9].

Бінаризація – це перетворення зображення до чорно-білого вигляду. Кожен піксель може бути або абсолютно чорним, або абсолютно білим, що відповідає максимальному і мінімальному значенню інтенсивності пікселя. Ця процедура дозволяє позбавитись надлишкової інформації. Після цього лінії на бінаризованому зображенні стоншуються до товщини в один піксель, а з отриманого скелета відбитка пальця вилучають особливі точки [8 – 9]. Такими точками можуть служити точки закінчення папілярних ліній та їх розгалуження (рис. 2.2).

З отриманих на попередніх етапах точок формується масив об’єктів з наступними параметрами: координати точки; тип лінії; кут, утворений ними. Набір параметрів особливих точок, отриманий зі сканованого відбитку пальця, порівнюється з набором еталонних параметрів відбитків зареєстрованих користувачів АС.

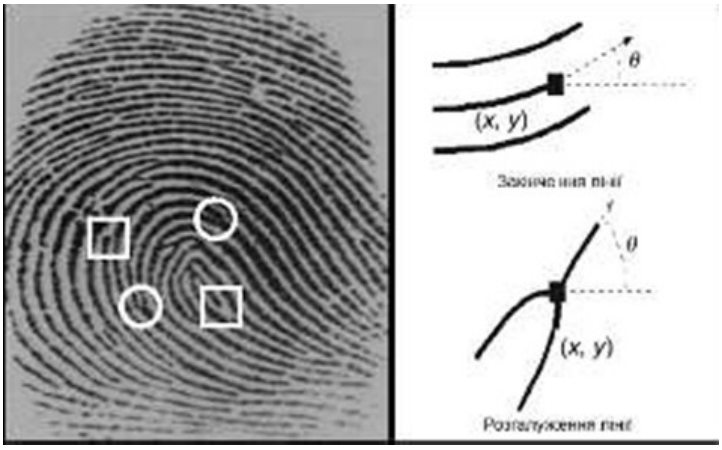


Рисунок 2.2 – Приклад особливих точок відбитка пальця (за [53])

На наступному етапі визначаються відхилення в значеннях цих параметрів. Великий поріг відхилення збільшить ймовірність хибного збігу біометричних характеристик двох користувачів – FAR (False Acceptance Rate). З іншого боку, мале значення допустимого відхилення є причиною збільшенню ймовірності відмови законному користувачу АС – FRR (False Rejection Rate) [6 – 7]. Проблема вибору порогу допуску пов'язана із деформацією та зміщенням пальця при скануванні, що призводить до отримання щоразу різних параметрів вилучених точок.

В результаті проведених досліджень було встановлено, що доцільно зберігати інформацію про комбінацію трьох особливих. Така структура – триплет, наведена на рис. 2.3

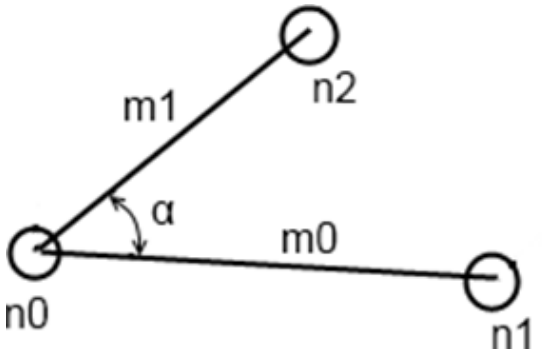


Рисунок 2.3 – Структура триплета (за [54])

Запропоновані параметри особливих точок є інваріантними до повороту та зміщення пальця, а також до деформації шкіри під час сканування.

Координати точок не зберігаються, адже, як було описано раніше, вони не надійні. Коли одна особлива точка асоціюється лише з однією парою особливих точок, така структура не є надійною і стій- кою до появи хибних точок чи пропуску точки (рис. 2.4).

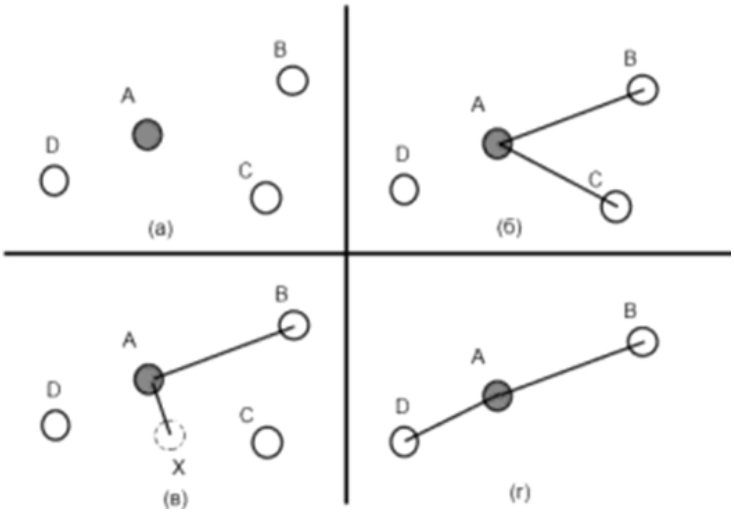


Рисунок 2.4 – Приклад генерації чотирьох особливих точок (за [55])

Отже, метод ідентифікації користувачів АС за особливими точками відбитків пальця полягає в наступному:

– типи точок, якими утворено триплет, повинні відповідати типам точок із ета-лонного зображення відбитка зареєстрованого користувача.

– відстань між центральною особливою точкою і її сусідніми точками має задовольняти умову:

$$\left|m_0 - m_{0e}\right| \le \Theta_m\left(m_0\right) \cap \left|m_1 - m_{1e}\right| \le \Theta_m\left(m_1\right)$$

– значення кута між m0 і m1 визначається наступною умовою:

$$\left|\alpha - \alpha_e\right| \le \Theta_\alpha\left(\alpha\right)$$

Наступним етапом є визначення відсотка відповідності наборів параметрів сканованого відбитку та збережених у системі ідентифікації еталонних значень для зареєстрованих користувачів.

2.3 Розробка алгоритму зчитування рогівки ока

Для розробки алгоритму роботи програмного засобу на основі використання зображення рогівки ока для реалізації процесу ідентифікації особи, розглянемо більш детальніше особливості структури розпізнавання рогівки.

Біометрична технологія передбачає систему розпізнавання фізичної особи на основі фізіологічних і поведінкових характеристик об'єкта. Важливо нині. Серед можливих біометричних даних рогівка ока є однією з найбільш точних і надійних завдяки стабільності будови її візерунка, незважаючи на вік і стан здоров'я. Малюнок рогівки індивідуальний для кожної людини.

Навіть ліве і праве очі однієї людини є унікальними. Формування рогівки ока відбувається в перший рік життя людини і більше не змінюється протягом усього подальшого часу. Ці властивості рогівки роблять її чудовою з інших біометричними даними для автоматизованих систем ідентифікації.

Розглянемо кожен крок алгоритму розпізнавання рогівки ока опираючись на метод Дж. Даугмана [56], який планується надалі використати в роботі:

1. Реєстрація зображення ока здійснюється спеціалізованою камерою, яка працює в ближньому інфрачервоному діапазоні і оснащена смуговим фільтром та інфрачервоним освітлювачем. Смуговий фільтр гасить області спектру поза межами цієї смуги, а інфрачервоний освітлювач має вузький спектральний максимум у цій смузі, що в результаті забезпечує придушення відблисків, що перекривають малюнок райдужної оболонки [56].

$$max_{(r,x_0,y_0)} \left| G_\sigma(r) * \frac{\delta}{\delta r_{r,x_0,y_0}} \frac{l(x,y)}{2\pi r} ds \right|,$$

де $G_\sigma(r) -$ функція розмиття Гаусіаном з параметром σ; $* -$ поєднання двох функцій для згладжування вихідного зображення; $l(x,y)$ –яскравість зображення в деякій точці; $(x_0, y_0) -$ визначувані координати центру райдужки; $r \in [r_{min}, r_{max}] -$ можливі радіуси границя райдужної оболонки.

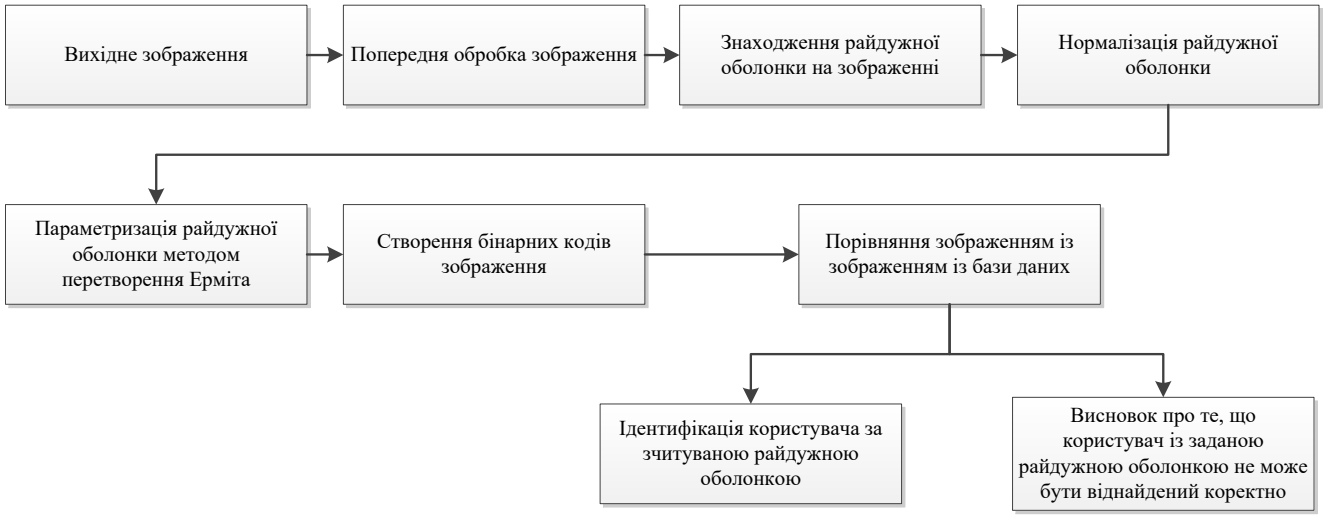


Рисунок 2.5 – Загальна схема розв'язання задачі ідентифікації рогівки ока (за [56])

При цьому зберігається безпечний для здоров'я рівень потужності ІЧ випромінювання освітлювача. Відразу після введення зображення попередньо обробляється для видалення деяких артефактів (просторової та яскравості дисторсії);

2. Оцінка якості зображення виключає зображення дуже низької якості з метою зниження ймовірності проведення помилкової сегментації;

3. Виділення на зображенні області інтересу (райдужної оболонки) полягає в знаходженні щодо темного об'єкта, близького за формою до кола і містить у собі концентричний більш темний об'єкт (зіниця). У багатьох системах виконується ще одна умова: наявність яскравого відблиску певної форми всередині зіниці (відблиск від освітлювача);

4. Оцінка якості сегментації полягає у визначенні придатності зображення та виділеної на ньому області рогівки для подальшої обробки.

$$G(r,\theta) = e^{-i\omega(\theta-\theta_0)} \cdot e^{\frac{-r(r-r_0)^2}{a^2}} \cdot e^{\frac{-(\theta-\theta_0)^2}{\beta^2}},$$

де $\omega -$ просторова частота; $\theta_0 -$ напрямок фільтра;

α і $\beta -$ коефіцієнти, обернено пропорційні частоті ω,

задають сімейство фільтрів Габора;

θ_0 і r_0 – задають просторові розташування фільтрів Габора.

5. Обчислення ознак і формування їх еталона райдужки складається з двох основних етапів, саме: нормування її зображення та обчислення інформативних ознак на нормованому зображенні;

6. Накопичення та вибір кращого еталон істотно покращує характеристики точності. Вибір еталона здійснюється на підставі показників якості, а також із використанням матриці крос-кореляції між накопиченими еталонами. Варто врахувати, що в деякихсистемах даний крок може бути відсутнім;

7. Створення запису для бази даних полягає у формуванні записи з власне зразка, часу реєстрації та інших даних, необхідні конкретного докладання.

8. Порівняння/пошук серед раніше зареєстрованих стандартів (розпізнаний, не розпізнаний). Біометрична система розпізнавання може виконувати такі завдання, як верифікація та ідентифікація. Верифікація здійснює перевірку на відповідність створеного зразка перевірка конкретній людині, раніше зареєстрованій у базі даних. А ідентифікація передбачає пошук тієї особи серед зареєстрованих, якій може належати стандарт.

Завдяки розгляду загального алгоритму розпізнавання, розуміння функціонування системи автоматичного розпізнавання райдужки очі, при подальшому її вивченні, стає гранично простим, що важливо задля подальшого детального вивчення існуючих методів.

2.4 Розробка алгоритму формування та зчитування qr-коду

Для розробки алгоритму роботи програмного засобу на основі використання QR-коду для реалізації процесу ідентифікації особи, розглянемо більш детальніше особливості його структури.

Дані QR-кодах закодовані за допомогою чорних і білих модулів, які надалі можуть бути розшифровані спеціальними автоматизованими засобами за дуже короткий час. При цьому використовуючи цю технологію можна

зашифрувати інформацію різних видів, починаючи зі звичайного тексту і закінчуючи зображеннями, адресами веб-сайтів, номерами телефону.

QR код підтримує кілька способів кодування даних, залежно від того, які символи використовуються: цифрове, літерно-цифрове, кандзі (китайсько-японські ієрогліфи) та побайтове кодування.

Цифрове кодування має на увазі використання лише цифр від 0 до 9, буквено цифрове — великі літери латинського алфавіту, цифри та символи $%*+-./: і пробіл, кандзі я розглядати не буду, а байти кодування не вимагають взагалі. Спочатку вам треба створити порожню послідовність біт, яка далі заповнюватиметься.

Процес генерації QR коду ділиться на кілька чітких кроків: кодування даних; додавання службової інформації та заповнення; поділ інформації на блоки; створення байтів корекції; об'єднання блоків; розміщення інформації на QR коді.

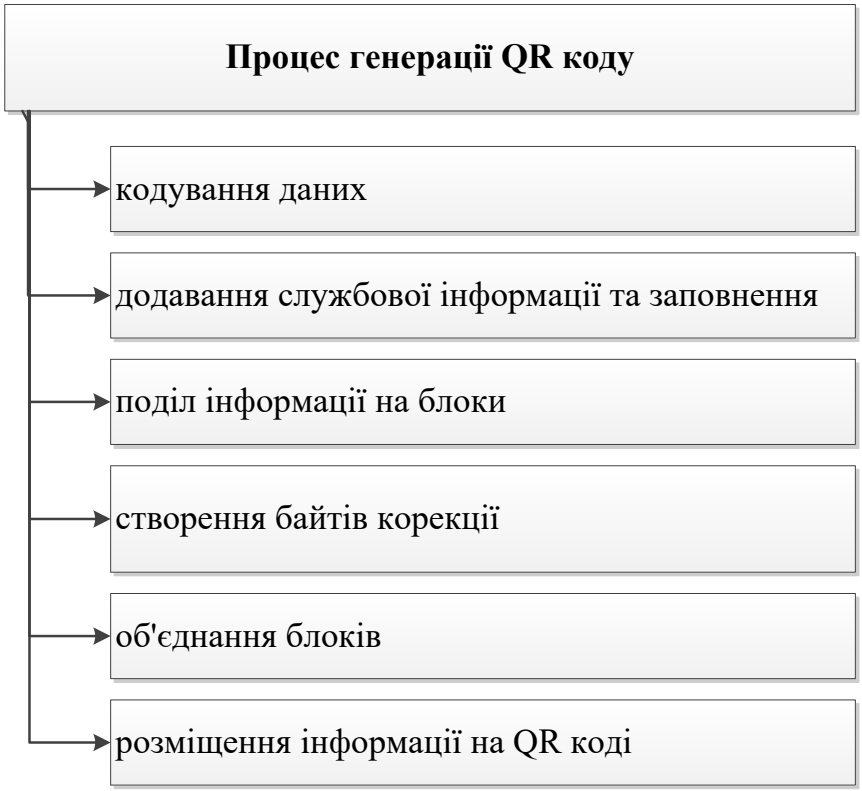


Рисунок 2.6 – Процес генерації qr-коду (за [56])

Дані QR-кодах закодовані за допомогою чорних і білих модулів, які надалі можуть бути розшифровані спеціальними автоматизованими засобами за дуже короткий час. При цьому використовуючи цю технологію можна зашифрувати інформацію різних видів, починаючи зі звичайного тексту і закінчуючи зображеннями, адресами веб-сайтів, номерами телефону.

QR код підтримує кілька способів кодування даних, залежно від того, які символи використовуються: цифрове, літерно-цифрове, кандзі (китайсько-японські ієрогліфи) та побайтове кодування. Цифрове кодування має на увазі використання лише цифр від 0 до 9, буквено цифрове — великі літери латинського алфавіту, цифри та символи $%*+-./: і пробіл, кандзі я розглядати не буду, а байти кодування не вимагають взагалі. Спочатку вам треба створити порожню послідовність біт, яка далі заповнюватиметься.

В роботі застосовується цифрове кодування. Цей тип кодування вимагає 10 бітів на 3 символи. Вся послідовність символів розбивається на групи по 3 цифри, і кожна група (тризначне число) перетворюється на 10-бітове двійкове число і додається до послідовності біт. Якщо загальна кількість символів не кратно 3, то якщо в кінці залишається 2 символи, отримане двозначне число кодується 7 бітами, а якщо 1 символ, то 4 бітами.

Наприклад, є рядок "12345678", який треба закодувати. Ми розбиваємо її на числа: 123, 456 і 78, потім переводимо кожне з них у двійковий вигляд: 0001111011, 0111001000 та 1001110, і об'єднуємо це в один потік: 000111101101100

Додавання службової інформації. На цьому етапі треба визначитися з рівнем корекції: що вище цей рівень, то вище допустимий рівень пошкодження зображення і тим менше інформації за рівного розміру. Усього є 4 рівні корекції: L (припустимо максимум 7% пошкоджень), M (15%), Q (25%) та H (30%). Найчастіше використовується рівень M. Якщо ви хочете додати QR код свій малюнок, то використовуйте рівень H.

Ще одна властивість QR коду - його версія (чим вона більша, тим більший розмір). Усього існує 40 версій. Номер версії залежить від кількості інформації, що кодується, і від рівня корекції. У таблиці 2 вказано максимальну кількість корисної інформації разом із службовою (у бітах), яку можна закодувати QR коді цієї версії. З цієї таблиці визначиться версія QR коду.

Додавання службових полів. До цього моменту вже має бути обраний рівень корекції та визначено версію. Тепер треба перед послідовністю біт, отриманої в попередньому пункті, додати на початку два поля: спосіб кодування та кількість даних. Спосіб кодування - поле довжиною 4 біти, яке має наступні значення: 0001 для цифрового кодування, 0010 для буквено-цифрового та 0100 для побайтового. Кількість даних - це кількість символів, що кодуються, а для побайтового - кількість байт (а не біт в отриманій послідовності), представлене у вигляді двійкового числа певної довжини [57].

Заповнення. На даному етапі у нас є послідовність біт даних, кількість біт в якій, напевно, не кратно 8. Треба доповнити її нулями так, щоб її довжина стала кратною 8. Тепер нашу послідовність біт можна розбити на групи по 8 біт і представити у вигляді послідовності байт (далі ми так і робитимемо). Якщо кількість біт у поточній послідовності байт менше, ніж потрібно для обраної версії, то її треба доповнити байтами, що чергуються 11101100 і 00010001. Таким чином, у нас вийшла послідовність байт, довжина якої відповідає обраній версії QR коду.

Приклад. Є послідовність: <послідовність біт, довжина якої кратна $8 > 10101011101$; доповнюємо її нулями, щоб її довжина стала кратною 8: <послідовність біт, довжина якої кратна $8 > 10101011101\ 00000$; тепер припустимо, що її довжина - 104 біти, а для обраної версії необхідно 128 біт, тоді для заповнення потрібно додати 24 «заповнюють» біти (3 байти): <послідовність біт, довжина якої кратна $8 > 10101011101\ 0000\ 1001$.

Таким чином, практична реалізація qr-коду здійснена за загальним алгоритмом наведеним вище. Усі деталі реалізації спираються на використання алгоритму кода Ріда-Соломона.

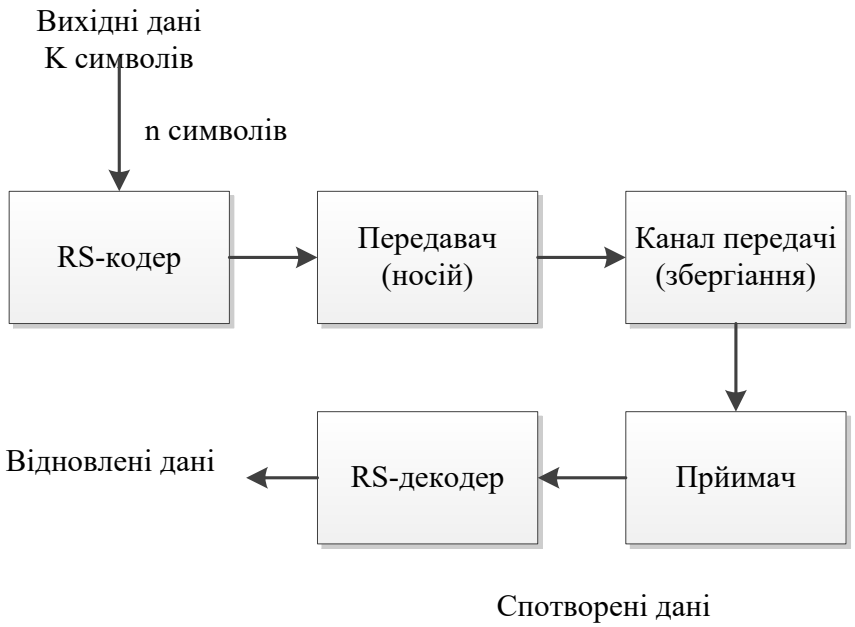


Рисунок 2.7 – Алгоритм формування qr-коду Ріда-Соломона (за [57])

Коротко наведемо особливості даного алгоритму.

При систематичному кодуванні до інформаційного блоку з символів k приписуються $2t$ перевірочних символів, при обчисленні кожного перевірочного символу використовуються всі k символів вихідного блоку. У цьому випадку немає витрат ресурсів при вилученні вихідного блоку, якщо інформаційне слово не містить помилок, але кодувальник/декодувальник повинен виконати $k(n-k)$ операцій складання та множення для генерації перевірочних символів. Крім того, оскільки всі операції проводяться в полі Галуа, самі операції кодування/декодування вимагають багато ресурсів і часу. Швидкий алгоритм декодування, що базується на швидкому перетворенні Фур'є, виконується за час порядку $O(ln^2 n)$.

Кодування. При операції кодування інформаційний поліном множиться на багаточлен, що породжує. Помноження вихідного слова S довжини k на поліном, що не наводиться, при систематичному кодуванні можна виконати таким чином:

До вихідного слова приписуються $2t$ нулів, виходить поліном $T = Sx\text{^}2t$.

Цей поліном ділиться на поліном, що породжує $G,$ знаходиться залишок $R, Sx\text{^}2t = QG + R$ де Q – приватне.

Цей залишок і буде коригуючим кодом Ріда Соломона, він приписується до вихідного блоку символів. Отримане кодове слово C=Sx^{2t}+R}\textstyle $C = Sx\text{^}2t + R$.

Кодувальник будується зі зсувних регістрів, суматорів та помножувачів. Зсувний регістр складається з осередків пам'яті, у кожному з яких є один елемент поля Галуа.

Декодування. Декодувальник, який працює за авторегресивним спектральним методом декодування, послідовно виконує наступні дії:

– обчислює синдром помилки;

– будує поліном помилки;

– знаходить коріння даного полінома;

– визначає характер помилки;

– виправляє помилки.

Обчислення синдрому помилки виконується синдромним декодером, який ділить кодове слово на багаточлен, що породжує. Якщо при розподілі з'являється залишок, то в слові є помилка. Залишок від поділу є синдромом помилки.

Побудова полінома помилки. Обчислений синдром помилки не вказує на положення помилок. Ступінь полінома синдрому дорівнює $2t,$ що набагато менше ступеня кодового слова n. Для отримання відповідності між помилкою та її становищем у повідомленні будується поліном помилок. Застосовується складний, але менш затратний алгоритм Берлекемпа - Мессі. Коефіцієнти

знайденого полінома безпосередньо відповідають коефіцієнтам помилкових символів кодовому слові.

Знаходження корення. На цьому етапі шукаються корені полінома помилки, що визначають положення спотворених символів кодового слова. Реалізується за допомогою процедури Ченя, що дорівнює повному перебору. У поліном помилок послідовно підставляються всі можливі значення, коли поліном звертається до нуля — коріння знайдено.

Таким чином, на основі стандартних методів генерації qr-коду та з використанням алгоритму Ріда-Соломона в роботі планується практично реалізувати генерацію qr-коду унікального для кожного користувача системи.

2.5 Обґрунтування вибору мови та засобу програмування

Виходячи з особливостей розробки та поставлених задач, обрано здійснювати практичну реалізацію веб-додатку на мові програмування C#, у середовищі Visual Studio та з використанням бази даних PostgreSQL.

C# – об'єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET. Розроблена Андерсом Гейлсбергом, Скотом Вілтамутом та Пітером Гольде під егідою Microsoft Research (при фірмі Microsoft) [58].

Зважаючи на зручний об'єктно-орієнтований дизайн, C# є гарним вибором для швидкого конструювання різних компонентів – від високорівневої бізнес логіки до системних додатків, що використовують низькорівневий код. Також слід зазначити, що C# є і Web орієнтованим – використовуючи прості вбудовані конструкції мови ваші компоненти можуть бути легко перетворені на Web сервіси, до яких можна буде звертатися з Internet за допомогою будь-якої мови на будь-якій операційній системі. Додаткові можливості і переваги перед іншими мовами приносить в C# використання передових Web технологій, таких як: XML (Extensible Markup Language) і SOAP (Simple Object Access Protocol). Середовище розробки Web сервісів дозволяє програмісту дивитися на існуючі

сьогодні Web додатки, як на рідні C# об'єкти, що дає можливість розробникам співвіднести Web сервіси, що є, з їх пізнаннями в об'єктно-орієнтованому програмуванні.

Мову програмування C# призначено для використання з програмною платформою. NET Framework, запропонованої компанією Microsoft в 2002 році. Платформа. NET Framework, або просто .NET, грає роль своєрідної операційної системи всередині операційної системи, дозволяючи створювати і звичайні програми, і веб-додатки на різних мовах програмування. В цілому нова платформа розроблялася, як альтернатива платформі Java, яка в той час швидко набирала популярність, основою якої також була віртуальна машина. Незважаючи на справедливу критику ряду особливостей платформи компанії Microsoft, до теперішнього моменту. NET Framework, а разом з нею і C# програмування, стали виключно популярні і затребувані в різних сферах.

Головною особливістю мови C# є її орієнтованість на платформу Microsoft .NET – творці C# поставили за мету розробки, надання розробникам засобів доступу до всіх можливостей платформи .NET. Це рішення можна вважати вимушеним, оскільки платформа .NET спочатку пропонувала значно більшу функціональність, ніж будь-яка інша мова. Розробку буде здійснено в середовищі розробки Visual Studio 2019. За замовчуванням Visual Studio форматує код по мірі його введення, автоматично вставляючи необхідні відступи та застосовує кольорове кодування для введення деяких елементів (наприклад, коментарів).

Microsoft Visual Studio – це інтегроване середовище розробки (IDE) від Microsoft. Середовище використовується для розробки комп'ютерних програм, а також веб-сайтів, веб-програм, веб-сервісів та мобільних додатків. Visual Studio використовує платформи для розробки програмного забезпечення Microsoft, такі як Windows API, Windows Forms, Windows Presentation Foundation, Windows Store і Microsoft Silverlight. Він може створювати як власний код, так і керований код. Приклад середовища наведений на рис. 2.8

[58].

Продукт, присвячений розробці настільних і серверних додатків Windows, інтегроване середовище розробки (IDE) Microsoft Visual Studio все більше нагадує швейцарський армійський ніж, здатний підтримувати безліч обчислювальних платформ, мов та середовищ виконання завдань.

Рисунок 2.8 — Середовище розробки Microsoft Visual Studio (за [58])

Visual Studio дає змогу розробнику написати єдину програму для роботи на кількох платформах Windows, таких як мобільний, настільний і навіть експериментальний середовище Microsoft HoloLens. Він також забезпечує спосіб створення додатків, які взагалі не працюють на комп'ютерах Windows, але замість цього вони працюють на пристроях iOS або у веб-додатках у хмарі.

Visual Studio включає в себе редактор коду, що підтримує IntelliSense (компонент завершення коду), а також рефакторинг коду. Вбудований відладчик працює як відладчик на рівні вихідного рівня, так і відладчик на рівні машини. Інші вбудовані інструменти включають програму для кодування, конструктор форм для побудови графічних інтерфейсів, веб-дизайнер, дизайнер класів та дизайнер схеми баз даних. Він приймає плагіни, які покращують функціональність практично на всіх рівнях, включаючи підтримку систем керування джерельними ресурсами (наприклад, Subversion та Git) та додавання

нових наборів інструментів, таких як редактори та візуальні розробники для мов або наборів інструментів для інших аспектів розробки програмного забезпечення життєвий цикл (наприклад, клієнт Team Foundation Server: Team Explorer).

Visual Studio включає в себе редактор коду, що підтримує IntelliSense (компонент завершення коду), а також рефакторинг коду. Вбудований відладчик працює як відладчик на рівні вихідного рівня, так і відладчик на рівні машини. Інші вбудовані інструменти включають програму для кодування, конструктор форм для побудови графічних інтерфейсів, веб-дизайнер, дизайнер класів та дизайнер схеми баз даних. Він приймає плагіни, які покращують функціональність практично на всіх рівнях, включаючи підтримку систем керування джерельними ресурсами (наприклад, Subversion та Git) та додавання нових наборів інструментів, таких як редактори та візуальні розробники для мов або наборів інструментів для інших аспектів розробки програмного забезпечення життєвий цикл (наприклад, клієнт Team Foundation Server: Team Explorer) [59].

Visual Studio підтримує 36 різних мов програмування і дозволяє редакторові коду та відладчику підтримувати (в тій чи іншій мірі) майже будь-яку мову програмування, якщо існує певна мова-служба.

Вбудовані мови включають C, C ++, C ++ / CLI, Visual Basic. NET, C #, F #, JavaScript, TypeScript, XML, XSLT, HTML та CSS. Підтримка інших мов, таких як Python, Ruby, Node.js та M, серед інших, доступна через плагіни. Java (і J #) були підтримані в минулому. А також містить в собі сильний інструмент NuGet для завантаження різного роду плагінів та додатків.

Таким чином, обрані засоби програмування, а саме мова C#, середовище Visual Studio та з використанням бази даних PostgreSQL, повністю задовільняють потребам розроблюваного додатку.

2.6 Висновки до розділу

Отже, в даному розділі була здійснена розробка алгоритму роботи програмного додатку для удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

В розділі описано процес удосконалення алгоритму розпізнавання відбитку пальця, розробку алгоритму зчитування рогівки ока, розробку алгоритму формування та зчитування qr-коду, розробка алгоритму роботи додатку та обґрунтування вибору мови та засобу програмування.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ДОДАТКУ ДЛЯ УДОСКОНАЛЕННЯ МЕТОДУ КОМПЛЕКСНОЇ ІДЕНТИФІКАЦІЇ КОРИСТУВАЧА

3.1 Проектування графічного інтерфейсу користувача

Інтерфейс користувача (User interface) являє собою сукупність засобів і методів, за допомогою яких користувач взаємодіє з різними пристроями і апаратурою. Іншими словами, це той набір кнопок, посилань, форм, діалогових вікон, іконок, піктограм, банерів, повзунків і стрічок прокрутки, за допомогою якого користувач управляє продуктом.

Інтерфейс – тільки половина у взаємодії з системою, інша половина - людина, користувач. Для хорошої роботи інтерфейсу потрібно точно знати, що саме в будь-який конкретний момент користувач сприймає в інтерфейсі, про що думає, чого хоче добитися.

Інтерфейси є основою взаємодії всіх сучасних інформаційних систем. Якщо інтерфейс будь-якого об'єкта (персонального комп'ютера, програми, функції) не змінюється (стабільний, стандартизований), це дає можливість модифікувати сам об'єкт, який не перебудовуючи принципи його взаємодії з іншими об'єктами.

Інтерфейс (ПІ) ділиться на компоненти. Для роботи розглянемо деякі з них, це графічну і біометричну компоненту.

Графічний інтерфейс користувача – це представлення програмних функцій графічними елементами екрану. Характерною особливістю цього виду інтерфейсу є те, що діалог з користувачем ведеться не за допомогою команд, а за допомогою графічних образів – меню, вікон, інших елементів. Хоча і в цьому інтерфейсі подаються команди машині, але це робиться «опосередковано», через графічні образи.

Біометрична технологія. Тут людина постає як сукупність ознак поведінки. Певні біометричні дані користувача (у випадку даної розробки – це

клавіатурний почерк), зчитуються з клавіатури, а потім за допомогою обраних методів виділяються команди.

Перевагами хорошого користувацького інтерфейсу можна вважати такі:

– підвищення конкурентоспроможності;

– зниження вартості розробки;

– збільшення аудиторії продукту;

– зменшення витрат на навчання і підтримку користувачів;

– зменшення втрат продуктивності працівників при впровадженні системи і більш швидке відновлення втраченої продуктивності;

– доступність функціональності системи для максимальної кількості користувачів;

– зниження ризику помилок.

Відмінними рисами якісного інтерфейсу є:

1. Стильова гнучкість – можливість використовувати різні інтерфейси з одним і тим же додатком, на практиці реалізується у вигляді набору «skins», для web-інтерфейсів – за допомогою таблиці стилів, в тому числі можливість у виборі користувачем власних установок ПІ (колір, ікони , підказки та ін.).

2. Спільне нарощування функціональності – можливість розвивати додаток без руйнування існуючого інтерфейсу.

3. Масштабованість – можливість легко налаштовувати і розширювати як інтерфейс, так і сам додаток при збільшенні числа користувачів, робочих місць, обсягу і характеристик даних.

4. Адаптивність до дій користувача – додаток має допускати можливість введення даних і команд безліччю різних способів (клавіатура, миша, інші пристрої). Крім цього програма повинна враховувати можливість переходу і повернення від вікна до вікна, від режиму до режиму, і правильно обробляти такі ситуації.

5. Незалежність в ресурсах – для створення призначеного для користувача інтерфейсу повинні надаватися окремі ресурси, спрямовані на

зберігання і обробку даних, необхідних для підтримки користувача.

6. Кросплатформеність – при переході на іншу апаратну (програмну) платформу повинно здійснюватися автоматичне перенесення і призначеного для користувача інтерфейсу, і кінцевого додатку.

7. Мультимедійність – сукупність всіх видів інформації (графічної, звукової, відео).

Далі розглянемо особливості розробки інтерфейсу діалогових вікон для розроблюваного додатку. У головному діалоговому вікні отримання доступу до роботи з системою, користувачеві надаватиметься можливість обрати функцію автентифікації або реєстрації, залежно від того чи має він обліковий запис в системі, чи ні. Після обрання відповідного процесу для подальшого проходження, відкривається наступне вікно, що містить основні компоненти розроблюваного програмного засобу (рис. 3.1).

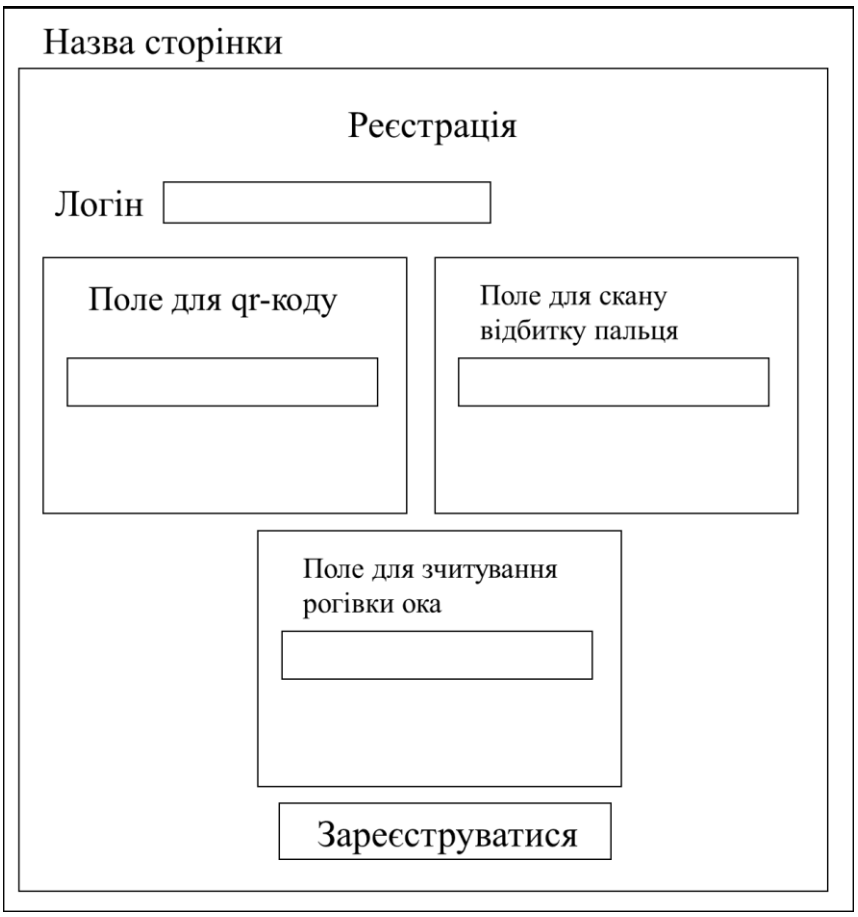


Рисунок 3.1 – Проектування діалогового вікна додатку

Отже, у діалоговому вікні для здійснення процесу реєстрації/авторизації користувача розташоване поле для вводу логіну, прикріплення (відображення сформованого) qr-коду, поле для прикріплення скану відбитка пальця, поле зчитування рогівки ока веб-камерою.

У нижній частині вікна розташована кнопка «Зареєструватися» (або для процесу авторизації «Увійти», відповідно), що підтверджує користувачем запуск відповідного процесу.

Таке діалогове вікно має просту структуру, легке для розуміння користувачем, не перенасичене зайвою інформацією, а отже якомога зручне та доступне для швидкого практичного застосування.

3.2 Програмна реалізація додатку

Враховуючи, поставлені задачі для розробки програмного засобу для удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока, було реалізовано наступну кодову послідовність.

Бібліотеки використані при програмній реалізації додатку:

```
using Emgu.CV;
using Emgu.CV.CvEnum;
using Emgu.CV.Face;
using Emgu.CV.Structure;
using Emgu.CV.Util;
using FaceDetectionAndRecognition;
using MessagingToolkit.QRCode.Codec;
using Microsoft.Win32;
using Newtonsoft.Json;
using QrEyeFinger.Context;
using SpeechEyeRecognize.Models;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Drawing.Imaging;
using System.IO;
using System.Runtime.CompilerServices;
using System.Timers;
```

```
using System.Windows;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Threading;
```

Вікно реєстрації нового користувача:

```
public partial class Registration : Window, INotifyPropertyChanged
  {
     public Registration()
     {
        InitializeComponent();
        captureTimer = new Timer()
        {
           Interval = Config.TimerResponseValue
        };
        captureTimer.Elapsed += CaptureTimer_Elapsed;
     }
```

Звернення до веб-камери ПК:

```
private Bitmap cameraCapture;
     public Bitmap CameraCapture
     {
        get { return cameraCapture; }
        set
        {
           cameraCapture = value;
           imgCamera.Dispatcher.Invoke(DispatcherPriority.Normal,
              new Action(() => { imgCamera.Source = BitmapToImageSource(cameraCapture);
}));
           NotifyPropertyChanged();
        }
     }
```

Фіксування рогівки ока:

```
public void OnLoad()
     {
        GetFacesList();
        if (videoCapture == null)
        {
           videoCapture = new VideoCapture(Config.ActiveCameraIndex);
           videoCapture.SetCaptureProperty(CapProp.Fps, 30);
           videoCapture.SetCaptureProperty(CapProp.FrameHeight, 450);
           videoCapture.SetCaptureProperty(CapProp.FrameWidth, 370);
           captureTimer.Start();
        }
        Init();
     }
```

Обробка зафіксованого зразка рогівки:

```
private void Init()
    {
      var resourceFile = $"{AppDomain.CurrentDomain.BaseDirectory}/ResourceFile.json";
      bool exists = System.IO.File.Exists(resourceFile);

      if (!exists)
        System.IO.File.Create(resourceFile);

      bool existsResourceImage = System.IO.Directory.Exists(ResourceImage);

      if (!existsResourceImage)
        System.IO.Directory.CreateDirectory(ResourceImage);
    }
```

Запис зразку рогівки до бази даних:

```
public void GetFacesList()
    {
      //haar cascade classifier
      if (!File.Exists(Config.HaarCascadePath))
      {
        string text = "Cannot find Haar cascade data file:\n\n";
        text += Config.HaarCascadePath;
        MessageBoxResult result = MessageBox.Show(text, "Error",
          MessageBoxButton.OK, MessageBoxImage.Error);
      }
      haarCascade = new CascadeClassifier(Config.HaarCascadePath);
      faceList.Clear();
      string line;
      FaceData faceInstance = null;
```

Перевірка зчитаного зображення та процесу фіксування рогівки:

```
private void ProcessFrame()
    {
      if (active)
      {
        bgrFrame = videoCapture.QueryFrame().ToImage<Bgr, Byte>();
        if (bgrFrame != null)
        {
          try
          {
            //for emgu cv bug
            Image<Gray, byte> grayframe = bgrFrame.Convert<Gray, byte>();

            Rectangle[] faces = haarCascade.DetectMultiScale(grayframe, 1.2, 10,
              new System.Drawing.Size(50, 50), new System.Drawing.Size(200, 200));
```

```
                //detect face
                FaceName = "No face detected";
                foreach (var face in faces)
                {
                    bgrFrame.Draw(face, new Bgr(255, 255, 0), 2);
                    detectedFace = bgrFrame.Copy(face).Convert<Gray, byte>();
                    FaceRecognition();
                    break;
                }
                CameraCapture = bgrFrame.ToBitmap();
            }
            catch (Exception ex)
            {
                throw new ArgumentException("Error: " + ex);
                //todo log
            }

        }
    }
}
```

Обробка зображення в базі даних:

```
private BitmapImage BitmapToImageSource(Bitmap bitmap)
    {
        using (MemoryStream memory = new MemoryStream())
        {
            bitmap.Save(memory, System.Drawing.Imaging.ImageFormat.Bmp);
            memory.Position = 0;
            BitmapImage bitmapimage = new BitmapImage();
            bitmapimage.BeginInit();
            bitmapimage.StreamSource = memory;
            bitmapimage.CacheOption = BitmapCacheOption.OnLoad;
            bitmapimage.EndInit();

            return bitmapimage;
        }
    }
```

Присвоєння логіну зчитаному біометричному параметру:

```
private void RegistrationVoid()
    {
        try
        {
            if (!VerifyData())
            {
                MessageBox.Show("Невдала спроба реєстрації");
                return;
```

```
            }
            RegistrFace(UserName.Text);
            var fileName = RegisterFingerPrint(UserName.Text);
            var user = new User()
            {
               UserName = UserName.Text,
               FingerPrintName = fileName,
               EncoderText = encoderText
            };

            var users = GetUsers();

            if (users == null)
            {
               users = new List<User>() { user };
            }
            else
            {
               users.Add(user);
            }
```

Реєстрація відбитку пальця:

```
private string RegisterFingerPrint(string personName)
      {
         var extension = Path.GetExtension(pathToFile.Text);
         var newFileName = $"{Guid.NewGuid()}{extension}";
         var resPath = $"{ResourceImage}\\{newFileName}";
         var encoder = new PngBitmapEncoder();
        encoder.Frames.Add(BitmapFrame.Create((BitmapSource)fingerPrintImage.Source));
         using (FileStream stream = new FileStream(resPath, FileMode.Create))
            encoder.Save(stream);
         return newFileName;
      }
```

Верифікація даних відбитку пальця:

```
private bool VerifyData()
      {
         if (String.IsNullOrEmpty(UserName.Text))
         {
            MessageBox.Show("Введіть логін");
            return false;
         }
         if (imgCamera.Source == null)
         {
            MessageBox.Show("Сталася помилка. Не вдалося зчитати райдужку ока");
            return false;
         }
```

```
        return true;
    }
```

Процес запуску генерування qr-коду:

```
private void Button_Click_4(object sender, RoutedEventArgs e)
    {
        OpenFileDialog dlg = new OpenFileDialog();
        //   dlg.InitialDirectory = "c:\\";
        //   dlg.Filter = "Image files (*.jpg)|*.jpg|All Files (*.*)|*.*";
        dlg.RestoreDirectory = true;
        if (dlg.ShowDialog() == true)
        {
            string selectedFileName = dlg.FileName;
            BitmapImage bitmap = new BitmapImage();
            bitmap.BeginInit();
            bitmap.UriSource = new Uri(selectedFileName);
            bitmap.EndInit();
            fingerPrintImage.Source = bitmap;
        }
    }
```

Верифікація qr-коду в системі:

```
private void qrGen_Click(object sender, RoutedEventArgs e)
    {
        SaveFileDialog sfd = new SaveFileDialog();
        sfd.Filter = @"JPEG file|*.jpg;*.jpeg;";
        sfd.ValidateNames = true;
        if (sfd.ShowDialog() == true)
        {
            QRCodeEncoder encoder = new QRCodeEncoder();
            encoder.QRCodeScale = 8;
            encoderText = Guid.NewGuid().ToString();
            Bitmap bmp = encoder.Encode(encoderText);
            bmp.Save(sfd.FileName, ImageFormat.Jpeg);
            //qrGen.IsCancel = true;
        }
        string selectedFileName = sfd.FileName;
        BitmapImage bitmap = new BitmapImage();
        bitmap.BeginInit();
        bitmap.UriSource = new Uri(selectedFileName);
        bitmap.EndInit();
        qrPrintImage.Source = bitmap;
        qrGen.Visibility = Visibility.Hidden;
    }
```

Процес авторизації зареєстрованого користувача:

```
public partial class Login : Window, INotifyPropertyChanged
  {
     public Login()
     {
        InitializeComponent();
        captureTimer = new Timer()
        {
           Interval = Config.TimerResponseValue
        };
        captureTimer.Elapsed += CaptureTimer_Elapsed;
     }
```

Запит до веб-камери для зчитування рогівки ока:

```
private Bitmap cameraCapture;
     public Bitmap CameraCapture
     {
        get { return cameraCapture; }
        set
        {
           cameraCapture = value;
           imgCamera.Dispatcher.Invoke(DispatcherPriority.Normal,
              new Action(() => { imgCamera.Source = BitmapToImageSource(cameraCapture);
}));
           NotifyPropertyChanged();
        }
     }
```

Обробка отриманого зразку рогівки ока з камери:

```
public void OnLoad()
     {
        qwImage.Visibility = Visibility.Visible;
        GetFacesList();
        if (videoCapture == null)
        {
           videoCapture = new VideoCapture(Config.ActiveCameraIndex);
           videoCapture.SetCaptureProperty(CapProp.Fps, 30);
           videoCapture.SetCaptureProperty(CapProp.FrameHeight, 450);
           videoCapture.SetCaptureProperty(CapProp.FrameWidth, 370);
           captureTimer.Start();
        }
     }
```

Запит на зчитування відбитку пальця користувача:

```
{
        Change_ResolutionByPath(Path.Combine(ResourceImage, user.FingerPrintName));
        var tempFile = SaveToTempDir();
        Change_ResolutionByPath(tempFile);
        var fingerprintImg1 = ImageLoader.LoadImage(Path.Combine(ResourceImage,
user.FingerPrintName));
        var fingerprintImg2 = ImageLoader.LoadImage(tempFile);
        //// Building feature extractor and extracting features
        var featExtractor = new PNFeatureExtractor() { MtiaExtractor = new
Ratha1995MinutiaeExtractor() };
        var features1 = featExtractor.ExtractFeatures(fingerprintImg1);
        var features2 = featExtractor.ExtractFeatures(fingerprintImg2);
        // Building matcher and matching
        var matcher = new PN();
        double similarity = matcher.Match(features1, features2);
        var score = similarity.ToString("0.000");
        similarity.ToString("0.000");

        return score;
    }
```

Запит на обробку qr-коду користувача:

```
private string DecodeQr()
    {
        MessagingToolkit.QRCode.Codec.QRCodeDecoder decode = new
MessagingToolkit.QRCode.Codec.QRCodeDecoder();
        var path = SaveQRToTempDir();
        var text = decode.Decode(new QRCodeBitmapImage((Bitmap)
Image.FromFile(path)));

        return text;
    }
```

Обробка qr-коду користувача:

```
private string SaveQRToTempDir()
    {
        var path = Path.GetTempFileName();
        var encoder = new PngBitmapEncoder();
        encoder.Frames.Add(BitmapFrame.Create((BitmapSource)qrPrintImage.Source));
        using (FileStream stream = new FileStream(path, FileMode.Create))
          encoder.Save(stream);

        return path;
    }
```

Якщо некоректно заповнене поле для введення логіну:

```
private bool VerifyData()
      {
         if (String.IsNullOrEmpty(UserName.Text))
         {
            MessageBox.Show("Введіть логін");
            return false;
         }
```

Якщо веб-камерою ПК не зчитана, а системою не зафіксована рогівка ока:

```
if (imgCamera.Source == null)
         {
            MessageBox.Show("Сталася помилка. Не вдалося зчитати райдужку ока");
            return false;
         }
         return true;
      }
```

Відмова або підтвердження успішної авторизації:

```
{
         List<User> users = new List<User>();
         var path = $"{
Path.GetDirectoryName(AppDomain.CurrentDomain.BaseDirectory)}/ResourceFile.json";
         bool exists = System.IO.File.Exists(path);
         if (!exists)
            System.IO.File.Create(path);

         using (StreamReader r = new StreamReader(path))
         {
            string json = r.ReadToEnd();
            users = JsonConvert.DeserializeObject<List<User>>(json);
         }
         return users;
      }
```

Таким чином, можна вважати, що обрані засоби програмування дозволяють у повній мірі реалізувати поставлені задачі роботи та досягти створення програмного засобу для комплексної ідентифікації користувача з використанням унікальних біометричних даних людини (рогівка ока та відбиток пальця) та qr-кодом.

3.3 Реалізація користувацького інтерфейсу

Під час реалізації користувацького інтерфейсу використовувались спроектовані на початку розділу, зразки діалогових вікон. Розглянемо покроково діалогові вікна програми під час роботи з додатком.

Після запуску виконуваного exe-файлу додатку відкривається головне вікно програми. У такому вікні наведена загальна інформація про систему, у верхній частині вікна вказано, що немає жодного авторизованого користувача та представлені кнопки «Вхід» та «Реєстрація» для того, щоб користувачі могли розпочати роботу із системою.

Після натиснення кнопки «Реєстрація» користувачеві відкривається відповідне вікно, де потрібно представити відповідні дані, а саме: логін, зчитати рогівку ока, прикріпити скан відбитку пальця та сформувати qr-код і, відповідно, зберегти його для подальшого використання.

У випадку, якщо дані надані користувачем введені вірно, тобто, введений унікальний для даної системи логін користувача, коректно зчитана рогівка ока, доданий скан відбитка пальця у відповідній якості і сформований qr-код, для користувача стає активна кнопка «Зареєструватися».

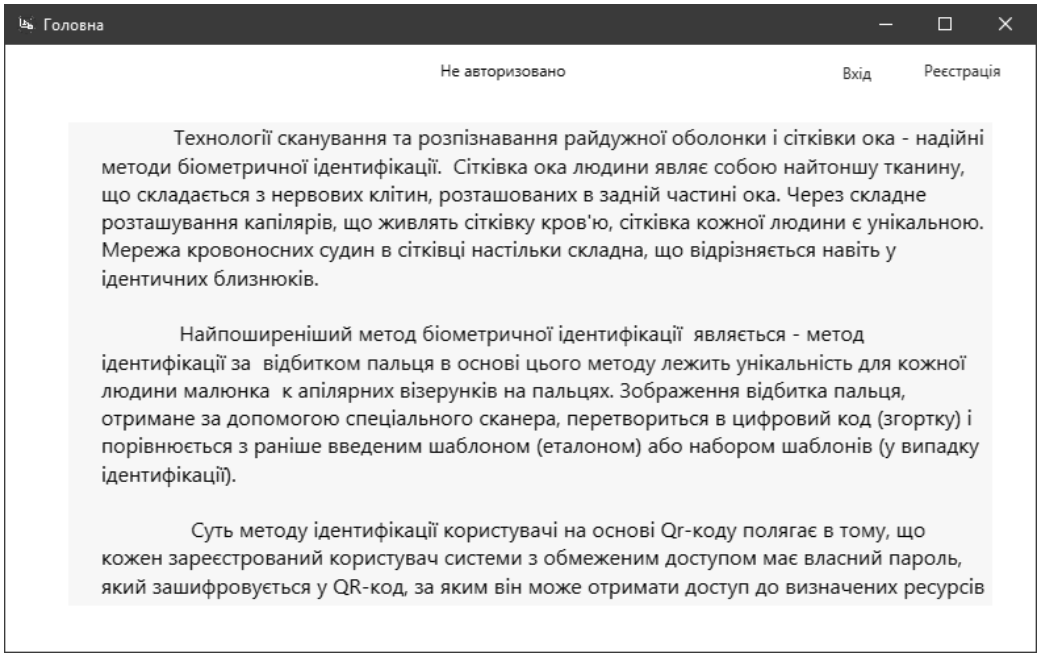


Рисунок 3.2 – Вигляд головного вікна додатку

На рисунку 3.3. наведено вигляд діалогового вікна для здійснення процесу реєстрації користувача із відповідними до заповнення полями.

На рисунку 3.4 наведений приклад заповнених полів даних для здійснення авторизації:

– прямокутником відображається зчитана веб-камерою рогівка ока,

– сформований qr-код для користувача

– доданий відбиток пальця.

У випадку, якщо усі дані коректні (введений унікальний для даної системи логін користувача, коректно зчитана рогівка ока, доданий скан відбитка пальця у відповідній якості і сформований qr-код), користувач отримує повідомлення про успішний процес реєстрації.

Після входу до системи, користувачеві відкривається головне вікно додатку.

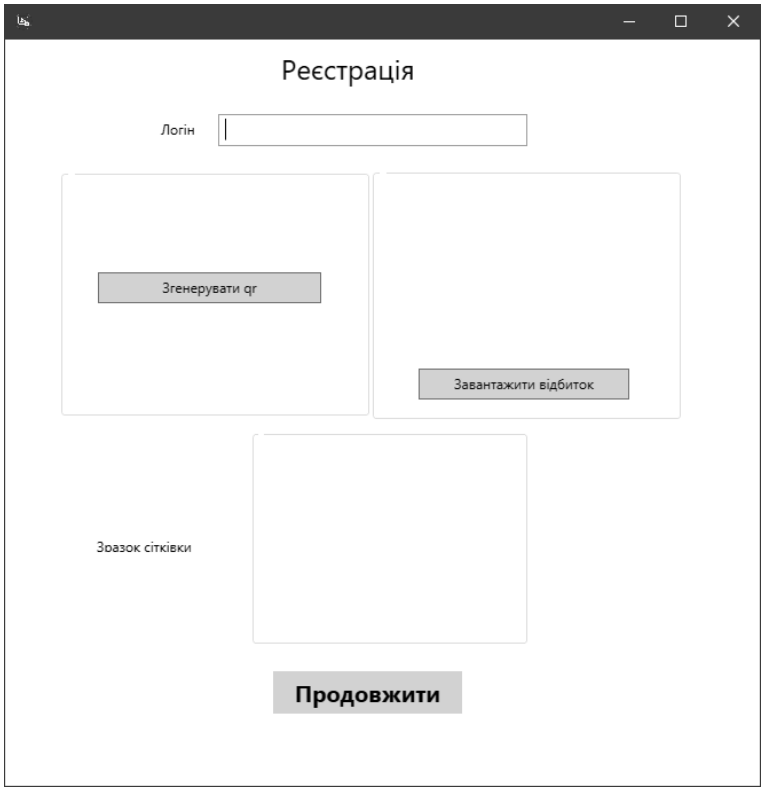


Рисунок 3.3 – Вигляд вікна реєстрації користувача

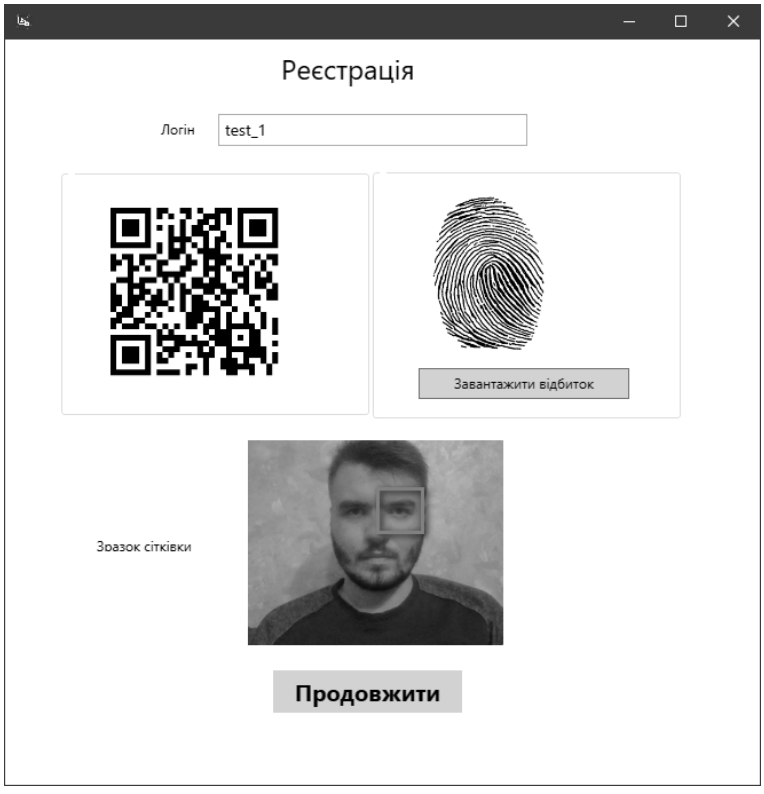


Рисунок 3.4 – Вигляд вікна реєстрації користувача із заповненими полями

Рисунок 3.5 – Зразок сформованого qr-коду для користувача

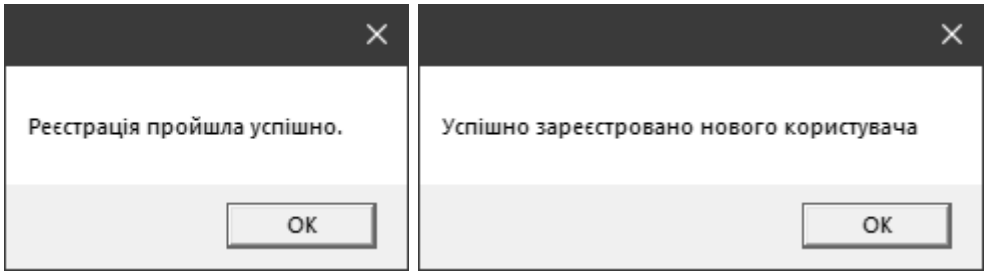


Рисунок 3.6 – Повідомлення про успішну реєстрацію користувача

Як зазначалось вище, після входу до системи, користувачеві відкривається головне вікно додатку, у верхній частині якого відображено його логін, що був використаний для входу.

У лівій частині вікна присутня кнопка виходу, якщо користувачеві необхідно завершити роботу з системою.

У основній частині вікна може відображатись будь-яка інформація, що необхідна у кожному конкретному випадку (текстові, графічні дані, схеми, мультимедіа і т.д.).

Приклад даного вікна наведений на рисунку 3.7.

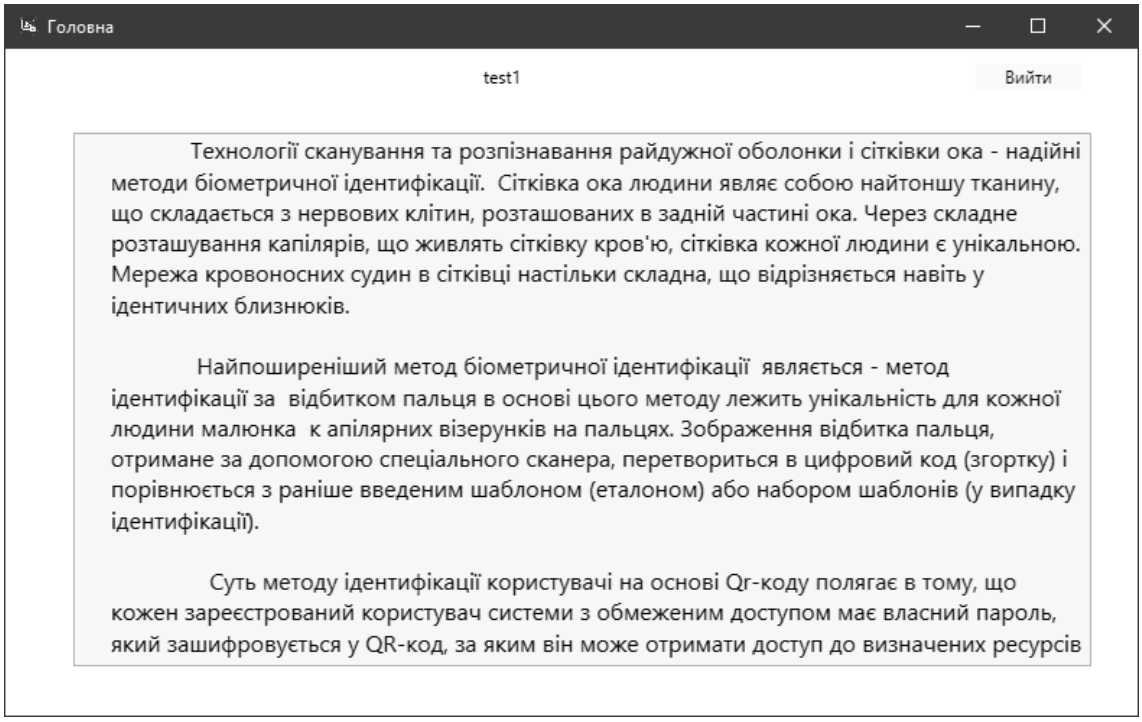


Рисунок 3.7 – Вигляд вікна авторизованого в системі користувача

Якщо користувач, був вже раніше зареєстрований в систем, для початку роботи в ній, йому необхідно здійснити процес авторизації.

Після натиснення кнопки «Увійти» користувачеві відкривається відповідне вікно, де потрібно представити відповідні дані, а саме: логін, зчитати рогівку ока, прикріпити скан відбитку пальця та додати раніше сформований qr-код.

У випадку, якщо дані надані користувачем введені вірно, тобто, введений унікальний для даної системи логін користувача, коректно зчитана рогівка ока, доданий скан відбитка пальця у відповідній якості і сформований qr-код, для користувача стає активна кнопка «Увійти».

На рисунку 3.8. наведено вигляд діалогового вікна для здійснення процесу авторизації користувача із відповідними до заповнення полями.

На рисунку 3.9 наведений приклад заповнених полів даних для здійснення авторизації (прямокутником відображається зчитана веб-камерою рогівка ока, сформований qr-код для користувача, доданий відбиток пальця).

Рисунок 3.8 – Вигляд вікна реєстрації користувача

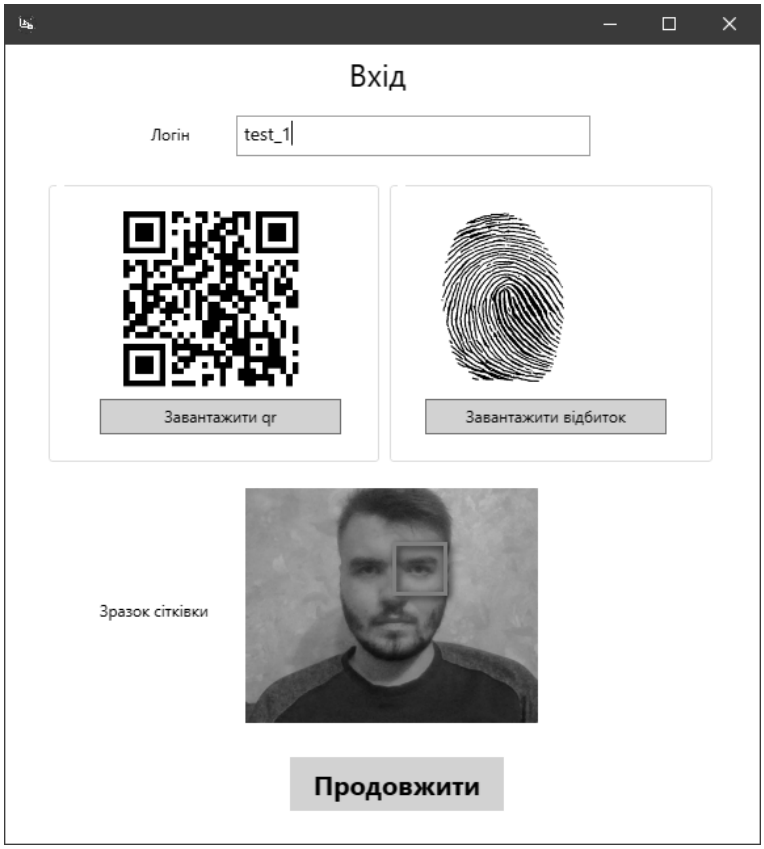


Рисунок 3.10 – Вигляд вікна реєстрації користувача із заповненими полями

У випадку, якщо усі дані коректні (логін користувача вірний, коректно зчитана рогівка ока, доданий скан відбитка пальця у відповідній якості і доданий qr-код), користувач отримує повідомлення про успішний процес авторизації в системі.

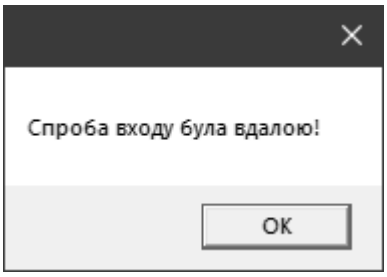


Рисунок 3.11 – Повідомлення про успішну реєстрацію користувача

У випадку, якщо користувач ввів не вірний логін, або зчитана рогівка ока чи відбиток пальця не відповідають логін, або ж логін, відбиток пальця та рогівка ока наведені коректно, але qr-код не відповідає вказаним даним, користувачеві у доступі відмовляється.

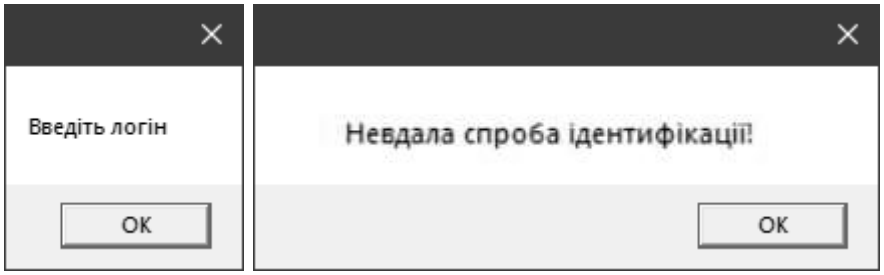


Рисунок 3.12 – Повідомлення про відмову у авторизації користувача

Таким чином, в даному підрозділі було описано та покроково пояснено алгоритм роботи користувача із системою комплексної ідентифікації до з використанням зчитування рогівки ока, скану відбитку пальця, розпізнання логіну та qr-коду користувача.

Даний користувацький інтерфейс розроблявся згідно поставлених вимог до візуального представлення та функціональної задачі додатку.

3.4 Тестування елементів розробленого програмного додатку

Рівень довіри до особи, що проходить процедуру ідентифікації, визначається з виконання умови $R \geq \delta$, де δ – порогове значення визнання ідентичності відбитків користувача системи.

Змінюючи значення параметра δ контролюється чутливість системи ідентифікації до спотворення сканованого зображення відбитка.

Однак, у випадку недостатньо великого значення порогового параметра системи існує можливість хибного визнання користувачем системи сторонньої особи.

В результаті експериментальних досліджень встановлено, що бажане значення цього параметра для системи ідентифікації користувачів системи становить $\delta = 90\%$.

Для демонстрації проведених досліджень було використано програмний додаток.

Результати роботи якого наведені на рис. 3.13.

Рисунок 3.13 – Результати роботи програмного комплексу із достовірності розпізнавання відбитку пальц

Отже, реалізований метод ідентифікації користувачів системи на основі особливих точок за відбитками пальця дозволив підвищити ефективність

процесу розпізнавання шляхом зменшення чутливості алгоритму до повороту та зміщення пальця при скануванні.

При цьому зберігаються такі переваги даного класу алгоритмів ідентифікації, як швидкодія та невисока обчислювальна складність.

Застосування методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока дозволяє підвищити достовірність ідентифікації особи на 11,2% у порівнянні із двофакторною ідентифікацією. Отже, використання розробленого методу в системі дозволить спростити процедуру ідентифікації користувачів та знизити вимоги до апаратних засобів системи.

Окрім того, застосування удосконаленого методу розпізнавання відбитків пальців у комплексі із ідентифікацією за рогівкою ока та qr-кодом суттєво підвищує достовірність розпізнання користувачів в системі та надійність її застосування на практиці в цілому.

3.5 Висновки до розділу

Таким чином, в даному розділі було здійснено практичну реалізацію програми комплексної автентифікації до системи захисту з використанням ідентифікації за рогівкою ока, сканером відбитку пальця та qr-кодом. В розділі описано проектування графічного користувацького інтерфейсу, особливості програмної реалізації розробки, реалізацію користувацького інтерфейсу додатку та тестування ефективності та достовірності методів використаних в роботі.

Реалізований метод ідентифікації користувачів системи на основі особливих точок за відбитками пальця дозволив підвищити ефективність процесу розпізнавання шляхом зменшення чутливості алгоритму до повороту та зміщення пальця при скануванні.

При цьому зберігаються такі переваги даного класу алгоритмів ідентифікації, як швидкодія та невисока обчислювальна складність.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Оцінювання комерційного потенціалу розробки ПЗ на комплексної ідентифікації користувача

Метою проведення технологічного аудиту є оцінювання комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності [60].

Результатом магістерської кваліфікаційної роботи є розробка програмного засобу для. Удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока. Для проведення технологічного аудиту залучено трьох незалежних експертів. У нашому випадку такими експертами є: Шиян А. А. (к.ф.-м.н., доцент каф. МБІС ВНТУ), Карпінець В. В.(к.т.н., доцент каф. МБІС ВНТУ) та Присяжний Д. О. (викладач каф. МБІС ВНТУ). Оцінювання комерційного потенціалу було здійснене за критеріями, що наведені в таблиці 4.1

Таблиця 4.1 – Критерії оцінювання комерційного потенціалу розробки бальна оцінка

Критерії оцінювання та 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 роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів

Продовження таблиці 4.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-тер.	0	1	2	3	4
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	Експлуатацій ні витрати значно вищі, ніж в аналогів	Експлуатацій ні витрати дещо вищі, ніж в аналогів	Експлуатацій ні витрати на рівні експл. витрат аналогів	Експлуатацій ні 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
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навч. наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фін. ресурси, які відсутні. Джерела фін. ідеї відсутні	Потрібні незначні фін.ресурси. Джерела фін. 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 використовую ться у військово-промисловом у комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві

Продовження таблиці 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	Необхідно отримання великої к-ті дозвільних документів на вир-во та реалізацію продукту, що вимагає значних коштів	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів	Необхідно тільки пові-домлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання комерційного потенціалу експертами розробки зведено в таблицю 4.2.

Таблиця 4.2 - Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта		
	1 – Шиян А.А	2 – Карпінець В.В.	3 – Присяжний Д.П.
1	4	3	3
Ринкові переваги (недоліки):			
2	4	3	3
3	3	3	3
4	3	3	4
5	4	4	3
Ринкові перспективи			
6	4	3	3
7	3	4	3
Практична здійсненність			
8	4	3	4
9	3	4	3
10	3	3	3
11	4	3	3
12	3	4	4
Сума балів	$СБ_1 = 42$	$СБ_1 = 40$	$СБ_1 = 39$
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = 40$		

За даними таблиці 4.2 можна зробити висновок, щодо рівня комерційного потенціалу розробки. Зважимо на результат й порівняємо його з рівнями комерційного потенціалу розробки, що представлено в таблиці 4.3.

Таблиця 4.3 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів $\overline{СБ}$, 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

Рівень комерційного потенціалу розробки, становить 40 балів, що відповідає рівню «вище середнього».

Проаналізуємо суть технічної проблеми та розглянемо аналоги.

Ідентифікація на основі біометричних даних – це засіб автоматичного розпізнавання особи на базі унікальних фізичних або поведінкових параметрів.

У біометричних системах ідентифікаційними є індивідуальні особливості людини, які в даному випадку називаються біометричними ознаками. Ідентифікація проводиться за рахунок порівняння отриманих біометричних параметрів і шаблонів, що зберігаються в основі. Залежно від характеристик, які використовуються, біометричні системи діляться на статичні та динамічні. Статична біометрія ґрунтується на даних (шаблонах), отриманих шляхом вимірювання анатомічних особливостей людини (відбитки пальців, візерунок райдужки ока тощо), а динамічна – на аналізі дій людини (голос, параметри підпису, її динаміка)).

Враховуючи такі переваги розробленого методу, можемо порівняти його з існуючими ресурсами, що спрямовані на виконання подібних функцій.

У таблиці 4.4 наведені основні технічні показники близького по змісту ресурсу і нового програмного продукту.

Таблиця 4.4 – Основні технічні показники близького по змісту ресурсу і нового програмного продукту

Показники, %	Інший додаток	Нова розробка	Відношення параметрів нової розробки до параметрів аналога
Функціональність	30	100	3.3
Надійність	70	100	1.4
Сумісність	100	100	1
Супровід	40	100	2,5
Економія ресурсів і часу	80	40	2.0
Простота використання	80	100	1.25

Отже, аналізуючи розроблений проект, можна стверджувати, що пошук здійснюється за всіма трьома параметрами.

На підставі вищевикладеного можна стверджувати, що нове технічне рішення, що пропонується для розробки, буде мати кращі показники ніж у аналога та більшою мірою задовольнить потреби споживачів. Тому його розробка та впровадження є актуальним та доцільним.

Даний програмний засіб створений для використання аналітичними структурами для пошуку удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

Це можуть бути державні органи безпеки, SEO-спеціалісти. Програмний засіб на сьогодні має перспективу та користь як для пересічних користувачів так і для спецслужб. Продукт, який пропонується є модифікацією продуктів що вже існують. Готовий програмний продукт буде реалізовуватись на ринку програмних засобів шляхом щомісячної підписки за певну плату.

Під час встановлення ціни та попиту на новий програмний продукт основна увага повинна акцентуватися на унікальності об’єкта купівлі-продажу,

цінах продуктів конкурентів, перевагах порівняно з аналогами, витратах, які зазнає покупець у разі заміни старого продукту новим, ступені терміновості та гостроті потреби.

Програмний засіб готовий для використання. Фахівці відповідної кваліфікації наявні, трудові та фінансові ресурси теж, обслуговування програми може відбуватись в режимі он-лайн, з будь-якої точки світу, тому що програмний додаток підключений до серверу і потребує доступу в Інтернет.

Комерціалізація розробки знаходиться на початковому етапі. Ведуться пошуки інвесторів та партнерів. Наявні зацікавлені особи, що готові першими випробувати програмний засіб в обмін на акт впровадження та подальшу рекламу від їх імені. Просування на ринок планується шляхом реалізації та продажу через спеціалізовані магазини ПЗ.

4.2 Прогнозування витрат на виконання наукової роботи та впровадження її результатів

Прогнозування витрат на виконання науково-дослідної, дослідно-конструкторської та конструкторсько-технологічної роботи складається з таких етапів:

1-й етап: розрахунок витрат, які безпосередньо стосуються виконавців даного розділу роботи;

2-й етап: розрахунок загальних витрат на виконання даної роботи;

3-й етап: прогнозування загальних витрат на виконання та впровадження результатів даної роботи.

Виконаємо розрахунок витрат, які безпосередньо стосуються виконавців даного розділу роботи, за такими статтями та формулами, приймаючи до уваги те, що для розробки інформаційної технології було залучено одного розробника програмного забезпечення.

1. Основна заробітна розробника-дослідника $З_о$:

$$З_о = \frac{М}{Т_р} \cdot t, \text{грн.} \quad (4.1)$$

де М – місячний посадовий оклад – 9 500 грн;

$Т_р$ – число робочих днів в місяці; приблизно $Т_р = 22$ дні;

t – число робочих днів роботи розробника-дослідника - 70.

Таким чином:

$$З_о = \frac{9500}{22} \cdot 70 = 30227,273 \text{ (грн.)}$$

2. Додаткова заробітна плата $З_д$ розробника розраховується як 12% від основної заробітної плати:

$$З_д = 0,12 \cdot 30227,273 = 3\,627,273 \text{ (грн.)}$$

3. Нарахування на заробітну плату $Н_{зп}$ розробника становить:

$$Н_{зп} = (З_о + З_д) \cdot \frac{\beta}{100} \quad (4.2)$$

де $З_о$ – основна заробітна плата розробника;

$З_д$ – додаткова заробітна плата розробника;

β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування – 22%.

$$Н_{зп} = (30227,273 + 3\,627,273) \cdot 0,22 = 7\,448 \text{ (грн.)}$$

4. Амортизація обладнання, комп'ютерів та приміщень, які використовувались під час виконання даного етапу роботи розраховуємо за формулою:

$$А = \frac{Ц \cdot Т}{12 \cdot Т_В} \quad (4.3)$$

де Ц – загальна балансова вартість обладнання, приміщення тощо, грн.;

Т – фактична тривалість використання, міс;

$Т_В$ – термін використання обладнання, приміщень тощо, роки.

Розробка програмного забезпечення ведеться 3 місяці.

Розрахунки зведено до таблиці 4.5:

Таблиця 4.5 – Амортизаційні відрахування

Найменування	Балансова вартість (грн.)	Термін використання (років)	Фактична тривалість використання, (міс.)	Величина амортизаційних відрахувань, (грн..)
Офісне приміщення	70 000	20	3	875
Комп'ютер	21 400	5	3	1 070
Монітор	3200	5	3	160
Всього				2 105

5. Витрати на силову електроенергію $В_е$ розраховуються за формулою:

$$В_е = В \cdot П \cdot Ф \cdot К_п \text{ (грн.)} \quad (4.4)$$

Де В – вартість 1 кВт-год.;

П – установлена потужність обладнання – 0,8 кВт;

Ф – фактична кількість годин роботи обладнання – 528 годин,

$К_п$ – коефіцієнт використання потужності.

$$В_е = 2{,}44 \cdot 0{,}8 \cdot 528 \cdot 0{,}14 = 144{,}29 \text{ (грн).}$$

6. Інші витрати $В_{ін}$ охоплюють:

– витрати на управління організацією;

– оплату службових відряджень;

– витрати на утримання, ремонт та експлуатацію основних засобів;

– витрати на опалення, освітлення, водопостачання, охорону праці тощо.

Інші витрати $В_{ін}$ можна прийняти як 50% від суми основної заробітної плати розробника:

$$В_{ін} = 30227,273 \cdot 0,5 = 15\,114 \text{ (грн)}$$

Послуги Інтернету – 150 грн., папір – 100 грн., канцтовари – 80 грн. Загальна вартість становить:

$$150 + 100 + 80 = 330 \text{ (грн.)}$$

7. Сума всіх попередніх статей витрат дає витрати на виконання даної частини роботи – В.

$$В = 30227,273 + 3\,627,273 + 7\,448 + 2\,105 + 144,29 + 15\,114 + 330 = 58\,995,84 \text{ (грн.)}$$

8. Проведемо прогнозування загальних витрат ЗВ на виконання та впровадження результатів виконаної наукової роботи. Прогнозування здійснюється за формулою:

$$ЗВ = \frac{В_{заг}}{\beta}, \text{ грн.} \quad (4.5)$$

де β – коефіцієнт, який характеризує етап (стадію) виконання даної роботи.

Так, якщо розробка знаходиться:

– на стадії науково-дослідних робіт, то $\beta \approx 0,1$;

– на стадії технічного проектування, то $\beta \approx 0,2$;

– на стадії розробки конструкторської документації, то $\beta \approx 0,3$;

– на стадії розробки технологій, то $\beta \approx 0{,}4$;

– на стадії розробки дослідного зразка, то $\beta \approx 0{,}5$;

– на стадії розробки промислового зразка, $\beta \approx 0{,}7$;

– на стадії впровадження, то $\beta \approx 0{,}9$.

$В_{заг}$ – загальна вартість всієї наукової роботи.

$$В = 58\,995{,}84\ (грн.)$$

$$ЗВ = \frac{58\,995{,}84}{0{,}7} = 84\,280\ (грн.)$$

Отже, прогноз загальних витрат ЗВ на виконання та впровадження результатів виконаної наукової роботи складає $84\,280\ (грн.)$

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

У даному підрозділі проведемо кількісне прогнозування, яку вигоду, зиск можна отримати у майбутньому від впровадження результатів виконаної наукової роботи. В умовах ринку узагальнюючим позитивним результатом, що його отримує підприємство від впровадження результатів тієї чи іншої розробки, є збільшення чистого прибутку підприємства. Зростання чистого прибутку можна оцінити у теперішній вартості грошей.

Зростання чистого прибутку забезпечить інвестору надходження додаткових коштів, які дозволять покращити фінансові результати діяльності.

Виконання даної наукової роботи та впровадження її результатів складає приблизно 1 рік.

Позитивні результати від впровадження розробки очікуються вже в перші місяці після впровадження.

Проведемо детальне прогнозування позитивних результатів та кількісне їх оцінювання по роках.

Обчислимо збільшення чистого прибутку підприємства $\Delta П_і$ для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, розраховується за формулою:

$$\Delta П_i = \sum_{1}^{n} (\Delta П_я \cdot N + П_я \cdot \Delta N)_i \qquad (4.6)$$

де $\Delta П_я$ – покращення основного якісного показника від впровадження результатів розробки у даному році;

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки;

$П_я$ – основний якісний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки.

Припустимо, що внаслідок впровадження результатів наукової розробки чистий прибуток підприємства збільшиться на 100 грн., а кількість одиниць реалізованої послуги збільшиться:

– протягом першого року – на 1000 од.,

– протягом другого року – ще на 5000 од.,

– протягом третього року – ще на 7000 од.

Орієнтовно: реалізація продукції до впровадження результатів наукової розробки складала 1 шт., а прибуток, що його отримувало підприємство на одиницю продукції до впровадження результатів наукової розробки – 40 грн.

Потрібно спрогнозувати збільшення чистого прибутку підприємства від впровадження результатів наукової розробки у кожному році відносно базового.

Збільшення чистого прибутку підприємства $\Delta П_1$ протягом першого року складе:

$$\Delta П_1 = 40 \cdot 1 + (40 + 100) \cdot 1000 = 180\,000 \text{ (грн.)}$$

Обчислимо збільшення чистого прибутку підприємства $\Delta П_2$ протягом другого року:

$$\Delta П_2 = 40 \cdot 1 = (40 + 100) \cdot (1000 + 5000) = 1\,080\,000 \text{ (грн.)}$$

Збільшення чистого прибутку підприємства $\Delta П_3$ протягом третього року становитиме:

$$\Delta П_3 = 40 \cdot 1 + (40 + 100) \cdot (1000 + 5000 + 7000) = 2\,340\,000 \text{ (грн.)}$$

Отже, розрахунки показують, що відповідно прогнозуванню комерційний ефект від впровадження розробки виражається у значному збільшенні чистого прибутку підприємства.

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Основними показниками, які визначають доцільність фінансування наукової розробки певним інвестором, є абсолютна і відносна ефективність вкладених інвестицій та термін їх окупності.

Розрахунок ефективності вкладених інвестицій передбачає:

1-й крок. Розрахунок теперішньої вартості інвестицій PV, що вкладаються в наукову розробку. Такою вартістю ми можемо вважати прогнозовану величину загальних витрат ЗВ на виконання та впровадження результатів НДДКР, тобто $ЗВ = PV = 84\,280 \text{ (грн.)}$

2-й крок. Розрахуємо очікуване збільшення прибутку$\Delta П_i$, що його отримає підприємство (організація) від впровадження результатів наукової розробки, для кожного із років, починаючи з першого року впровадження.

3-й крок. Будуємо вісь часу, на якій відображаємо всі платежі (інвестиції та прибутки), що мають місце під час виконання науково-дослідної роботи та впровадження її результатів.

Рисунок 4.1 характеризує рух платежів (інвестицій та додаткових прибутків).

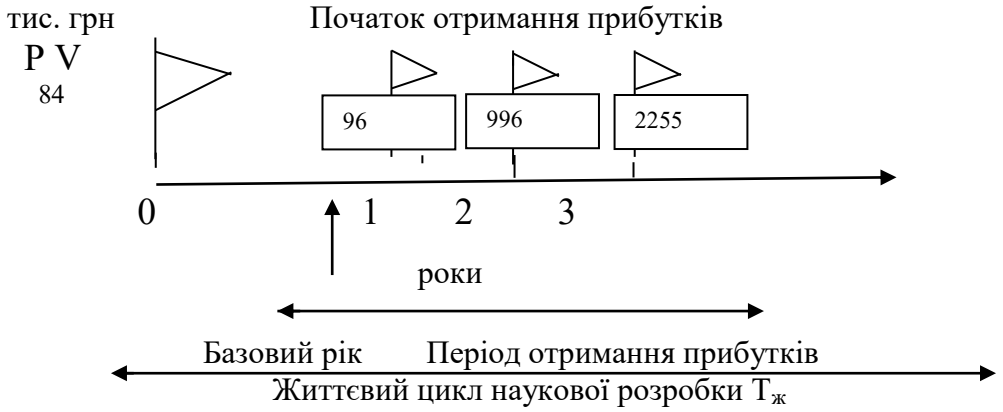


Рисунок 4.1 – Вісь часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів НДДКР

4-й крок. Розрахуємо абсолютну ефективність вкладених інвестицій $Е_{абс}$ за формулою:

$$Е_{абс} = (ПП - PV), (грн.) \quad (4.7)$$

де ПП – приведена вартість всіх чистих прибутків, що їх отримає підприємство (організація) від реалізації результатів наукової розробки, грн.;

PV – теперішня вартість інвестицій $PV = ЗВ, грн.$

Приведена вартість всіх чистих прибутків ПП розраховується за формулою:

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, (грн) \quad (4.8)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн.;

Т – період часу, протягом якого виявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні - 0,1;

t – період часу (в роках) від моменту отримання чистого прибутку до точки „0”;

$$ПП = \frac{96\,720}{(1+0{,}1)^1} + \frac{995\,720}{(1+0{,}1)^2} + \frac{2255720}{(1+0{,}1)^3} = 2\,605\,591(грн.)$$

$$Е_{абс} = 829937{,}26 - 67664{,}17 = 762273{,}08\ (грн.)$$

Оскільки $Е_{абс} > 0$, результат від проведення наукових досліджень щодо розробки програмного продукту та їх впровадження принесе прибуток, тобто є доцільним, проте це ще не свідчить про те, що інвестор буде зацікавлений у фінансуванні даної програми.

5-й крок. Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $Е_в$ за формулою:

$$Е_в = \sqrt[Т_ж]{1 + \frac{Е_{абс}}{PV}} - 1 \quad (4.9)$$

де $Е_{абс}$ – абсолютна ефективність вкладених інвестицій, грн.;

PV – теперішня вартість інвестицій $PV = ЗВ$, грн.

$Т_ж$ – життєвий цикл наукової розробки, роки.

$$E_{\text{в}} = \sqrt[3]{1 + \frac{762273{,}08}{84\ 280}} - 1 = \sqrt[3]{10{,}044} - 1 = 1{,}16 \text{ або } 116\%$$

Порівняємо $E_{\text{в}}$ з мінімальною (бар'єрною) ставкою дисконтування τ_{min}, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть.

Спрогнозуємо величину τ_{min}. У загальному вигляді мінімальна (бар'єрна) ставка дисконтування τ_{min}визначається за формулою:

$$\tau_{min} = d + f \tag{4.10}$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; d = 0,2;

f – показник, що характеризує ризикованість вкладень; величина f = 0,5.

$$\tau_{min} = 0{,}2 + 0{,}5 = 0{,}7$$

Оскільки

$$E_{\text{в}} = 116\% > \tau_{min} = 70\%,$$

то у інвестора є потенційна зацікавленість у фінансуванні даної наукової розробки.

6-й крок. Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій $\text{Т}_{\text{ок}}$ за формулою:

$$\text{Т}_{\text{ок}} = \frac{1}{\text{Е}_{\text{в}}}, \text{рік} \tag{4.11}$$

$$\text{Т}_{\text{ок}} = \frac{1}{1{,}16} = 0{,}9 \text{ (року)}$$

Оскільки термін окупності вкладених у реалізацію наукового проекту інвестицій менше трьох років ($\text{Т}_{\text{ок}} < 3$ років), то фінансування нової розробки є доцільним.

4.5 Висновки до розділу

В даному розділі було виконано оцінювання комерційного потенціалу розробки програмного засобу для удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

Проведено технологічний аудит з залученням трьох незалежних експертів. Визначено, що рівень комерційного потенціалу розробки вище середнього. Згідно з проведеним оцінюванням нова розробка є якісною та конкурентоспроможною.

Згідно із розрахунками всіх статей витрат на виконання науково-дослідної, дослідно-конструкторської та конструкторсько-технологічної роботи загальні витрати на розробку складають 84 280 (грн.). Розрахована абсолютна ефективність вкладених інвестицій в сумі 762273,08 (грн.)свідчить про отримання прибутку інвестором від комерціалізації програмного продукту.

Щорічна ефективність вкладених в наукову розробку інвестицій складає 116%, що вище за мінімальну бар'єрну ставку дисконтування, яка складає 70%. Це означає потенційну зацікавленість інвесторів у фінансуванні розробки.

Термін окупності вкладених у реалізацію проекту інвестицій становить 0,9 (року), що також свідчить про доцільність фінансування нової розробки.

Отже, в ході економічного аналізу представленої розробки, було з’ясовано, що проект має високу доцільність та є економічно ефективним.

ВИСНОВОК

В магістерській дипломній роботі здійснено едосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока

Для реалізації контролю доступу до ресурсів захищуваної системи необхідно використовувати систему ідентифікації користувачів. Біометрія є одним із найперспективніших напрямків для використання в АС. В якості унікальних ознак користувача системи використовуються:

– статичні характеристики – папілярний візерунок на пальцях, райдужна оболонка ока, геометрія обличчя, геометрія рук, сітківка ока;

– динамічні характеристики – голос, динаміка рукописного почерку тощо.

Нині, дедалі більше використовується багатофакторна або комплексна автентифікація користувачів, що ґрунтується на спільному використанні певних факторів автентифікації, що значним чином підвищує рівень надійності та захищеності системи, оскільки при наявності лише одного ідентифікатора особи не можна цілковито бути впевненим у надійності його достовірності.

Для кожного випадку застосування біометричної ідентифікації, лише при застосуванні одного ідентифікатора існує ризик його підробки, примусового використання, втрати і т.п.

Саме тому, ефективним методом усунення даного недоліку є поєднання декількох біометричних ідентифікаторів у комплексі, що може значно підвищити рівень захищеності системи та надійності, власне, самого процесу ідентифікації користувача.

Таким чином, в першому розділі роботи було проведено теоретичний огляд галузі, в якій проводиться розробка, представлено основні поняття систем автентифікації та способи такого захисту, також проведено аналіз застосування сучасних біометричних засобів ідентифікації, зокрема, для

підвищення надійності та стійкості до атак в системах автентифікації користувачів.

У другому розділі була здійснена розробка алгоритму роботи програмного додатку для удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

В розділі описано процес удосконалення алгоритму розпізнавання відбитку пальця, розробку алгоритму зчитування рогівки ока, розробку алгоритму формування та зчитування qr-коду, розробка алгоритму роботи додатку та обґрунтування вибору мови та засобу програмування.

У третьому розділі було здійснено практичну реалізацію програми комплексної автентифікації до системи захисту з використанням ідентифікації за рогівкою ока, сканером відбитку пальця та qr-кодом. В розділі описано проектування графічного користувацького інтерфейсу, особливості програмної реалізації розробки, реалізацію користувацького інтерфейсу додатку та тестування ефективності та достовірності методів використаних в роботі. Реалізований метод ідентифікації користувачів в системі дозволив підвищити ефективність процесу розпізнавання користувачів, шляхом поєднання декількох ідентифікаторів. При цьому зберігаються такі переваги даного класу алгоритмів ідентифікації, як швидкодія та невисока обчислювальна складність.

В четвертому розділі роботи, проаналізувавши отримані економічні показники, можна вважати, що запропонована розробка програмного засобу має високий комерційний потенціал, а тому є доцільною для подальшого впровадження.

Таким чином, аналізуючи отримані результати практично реалізованої роботи, можна вважати, що в результаті дослідження було удосконалено метод комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока та розроблено програмний засіб для реалізації удосконаленого методу.

ПЕРЕЛІК ПОСИЛАНЬ

1. Кухарев Г. А. Биометрические системы: Методы и средства идентификации личности че- ловека / Г.А. Кухарев. – СПб.: Политехника, 2001. –240 с.

2. Давлетханов М. Способы идентификации по отпечаткам пальцев [Электронный ресурс] / М. Давлетханов. – 2004. – Режим доступа: http://www.infobez.ru/article.

3. Домарев В.В. Безопасность информационных технологий. Системный подход / В.В. Домарев. – К.: ООО ТИД Диа Софт, 2004. – 992 с.

4. Евангели А. Технологии биоидентификации и биометрический рынок / А. Евангели // PC Week/RE/ 2003. №7 – С. 24-25.

5. Зиятдинов А.И. Принципы построения систем биометрической аутентификиции / А.И. Зиятдинов // МФТИ. 2005. – 8 с.

6. Сеньор Э.У. Руководство по биометрии / Э. У. Сеньор, Н. К. Ратха, Ш. Панканти, Дж. Х. Коннел, Р. М. Болл. – М.: Техносфера, 2007. – 368 с.

7. Задорожный В. Идентификация по отпечаткам пальцев. Ч. 1 [Электронный ресурс] / В. Задорожный // PC Magazine/Russian Edition. – 2004. – №1. – Режим доступа: http://bre.ru/security/20994.html.

8. Chaohong Wu Advanced feature extraction algorithms for automatic fingerprint recognition systems / Chaohong Wu. 2007. – 122 p.

9. Chirag Dadlani Fingerprint Recognition Using Minutiae-Based Features / Chirag Dadlani. 2007. – 26 p.

10. Різник О. «Біокон» – система біометричної ідентифікації користувача комп’ютерної мережі / О. Різник, Д. Дзюба, А. Чернодуб // Системи підтримки прийняття рішень. Теорія і практика: зб. доп. наук.-прак. конф. з міжнар. участю. «СППР 2009». – Київ, 2009. – С. 189 – 193.

11. Старовойтов В.В., Монич Ю.И. Распознавание человека по изображению радужной оболочки глаза // 2011. N.3. P.278–284.

12. A. Bertillon, “a couleur de liris.” Rev. Sci., vol. 36, no. 3, pp. 6573, 1885

13. J.G Daugman. “Iris recognition.” American Scientist, July-Aug. 2001

14. L. Flom and A. Safir, “Iris recognition system.” U.S. Patent 4 641 349, 1987.

15. J.G Daugman, “High Confidence Visual Recognition of Persons by a Test ou Statistical Independence.” IEEE Transactions On Pattern Analysis and Machine Intelligence, Vol. 15, No.11, pp.1148-1161,Nov. 1993.

16. R. Wildes, J.C. Asmuth, G.L. Green, S.C. Hsu, R.J. Kolczynski, J.R. Matey and S.E. McBride, “A system for automated iris recognition.” In Proceedings ou the IEEE Workshop on Applications of Computer Vision, pp. 121-128, 1994

17. W.W. Boles, “A wavelet transform based technique for the recognition of the human iris.” In Proceedings of the International Symposium on Signal Processing and its Application, ISSPA, Gold Coast, Australia, pp. 25-30, August, 1996.

18. S. Lobregt and M. A. Viergever, “A Discrete dynamic contour model.” IEEE Trans-actions on Medival Imaging, 14(1), 1995, 12-24.

19. Y. Zhu, T. Tan and Y. Wang “Biometric personal identification based on iris pattern.” Proceeding of. 15th International Conference on Pattern Recognition, vol. 2, pp. 801804, 2000.

20. S. Lim K. Lee O. Byeon and T. Kim, “Efficient Iris Recognition through Improvement of Feature Vector and Classifier.” ETRI J., vol. 23, no. 2, pp. 61-70, 2001.

21. S. Noh, K. Pae, C. Lee, J. Kim. “Multiresolution independent component analysis for iris identification.”, Computers and Communications, Phuket, Thailand, 2002.

22. Форсайт Д.А., Понс Ж. Компьютерное зрение. Современный подход.: Пер. с англ. – М.: Издательский дом "Вильямс" 2004. – 928 с.

23. Синицын И.Н., Новиков С.О., Ушмаев О.С. Развитие технологий интеграции биометрической информации // Системы и средства информатики. 2004. N.14. P.5.

24. Методы компьютерной обработки изображений / Под. ред. В.А.Сойфера. – 2-е изд., испр. – М.: ФИЗМАТЛИТ, 2003. – 784 с.

25. Розенфельд А. Распознавание и обработка изображений с помощью вычислительных машин. М.:Мир, 1972. – 230 с.

26. J. Zuo, N. Kalka, N. Schmid, A robust IRIS segmentation procedure for unconstrained subject presentation, in Aug 2006, pp. 1–6

27. S. Lim K. Lee O. Byeon and T. Kim, “Efficient Iris Recognition through Improvement of Feature Vector and Classifier.” ETRI J., vol. 23, no. 2, pp. 61-70, 2001.

28. W.W. Boles and B. Boashash, “A human identification technique using images of the iris and wavelet transform.” IEEE Transactions on Signal Processing, vol. 46, no. 4, pp. 1185-1188, 1998.

29. B. J. Joung, J. O. Kim, C. H. Chung, Key Seo Lee, Wha Young Yim, Sang Hyo Lee, “On Improvement for Normalizing Iris Region for a Ubiquitous Computing.” 2005, Singapore, May 9-12, 2005.

30. Canny J. A computational approach to edge detection // IEEE TPAMI. 1986. V.8. N.6. P.679–698

31. Kozlov A., Kudashev O., Matveev Y., Pekhovsky T., Simonchik K., Shulipa A. SVID speaker recognition system for the NIST SRE 2012

32. Продан А.И., Таланов А.О. Использование набора слуховых характеристик речи при идентификации по голосу // Казань, 2011. С. 338–344.

33. Коваль С.Л., Хитров М.В. Идентификация дикторов при анализе разноязычных фонограмм на основе сравнения формантных спектров URL: http://zhenilo.narod.ru/new_main/ips/2003_speech.pdf

34. Koval S. Formants matching as a robust method for forensic speaker identification // Proc. 11th Int. Conf. on Speech and Computer. St. Petersburg, 2006. P. 125–128.

35. Ladefoged P. Preliminaries to Linguistic Phonetics. Chicago: University of Chicago Press, 1971. 122 p.

36. Tomashenko N., Khokhlov Y. Fast algorithm for automatic alignment of speech and imperfect text data // Lecture Notes in Computer Science. 2013. V. 8113 LNAI. P. 146–153. doi: 10.1007/978-3-319-01931-4_20

37. Young S., Kershaw D., Odel J., Ollason D., Valtchev V., Woodland P. The HTK Book. Cambridge University Engineering Department, 2002. 271 p.

38. Schwarz P. Phoneme Recognition Based on Long Temporal Context. Ph.D. thesis. Brno University of Technology, 2008. 75 p.

39 BioAPI Consortium (Биометрический консорциум) URL: http://www.bioapi.org.

40 BioLink: защита информации, учет рабочего времени и контроль доступа средствами идентификации, сканерами отпечатков пальцев, голоса и лица, СКД, СКУД URL: http://www.biolink.ru.

41. Біометрія як універсальний спосіб ідентифікації людини URL: http://bablyukh.clan.su/publ/1-1-0-4.

42. Попов М., Задорожный В. Биометрические системы безопасности: URL: www.BRE.ru

43. Лакин Г.Ф. Биометрия. Учебное пособие. – М.: Высшая школа. – 1990. – 223 с.

44. Абламейко С.В. Обробка зображень: технології, методи, застосування / С.В. Абламейко. – М.: Амалфея, 2000. – 304 с.

45. Анісімов Б.В. Розпізнавання і цифрова обробка зображень / Б.В. Анісімов, В.Д. Курганов, В.К. Злобін. – М.: Висш.шк., 1983. – 295 с.

46. Pentland, A. & Turk, M. (1991). Eigenfaces for Recognition. Journal of Cognitive Neuroscience 3(1): 71-86.

47. Бастрикін А. І. Дактилоскопія. Знаки руки. / Бастрикін А. І., 2004.

48. Шаров В. Біометричні методи комп'ютерної безпеки/ Владислав Шаров // ByteMag. – 2005. URL: https://www.bytemag.ru/articles/detail.php

49.Задорожний В. Ідентифікація за відбитками пальців. Частина 1 / Задорожний В. // PC Magazine. – 2004. URL: https://bms.ucoz.ru/statii/

50. Егорова О. В., Клыкова Е. И., Пантелеев В. Г. Компьютерная микроскопия. – М.: Техносфера, 2005. – 304 с.

51. Гонсалес Р., Вудс Р.. Цифровая обработка зображений. – М.: Техносфера, 2005. – 1072 с.

52. Байгарова Н.С., Бухштаб Ю.А., Евтеева Н.Н., Корягин Д.А. Некоторые подходы к организации содержательного поиска изображений и видеоинформации.– Москва: Институт прикладной математики им. Келдыша РАН, 2002.– 24 с.

53. Гончаров А., Мельниченко А. Pseudometric Approach to Content Based Image Retrieval and Near Duplicates Detection.– РОМИП, 2008.– 34 с.

54. Хагхигат М. Дискриминантний кореляційний аналіз: Fusion в режимі реального часу для мультимодального біометричного розпізнавання / М. Хагхигат, М. Абдель-Мотталеб, В. Аналлабі., 2016.

55. N. K. Ratha. Enhancing security and privacy in biometrics-based authentication systems, IBM systems Journal / N. K. Ratha, J. H. Connell, R. M. Bolle. – №40.

56. Біометричні системи безпеки [Електронний ресурс] // Режим доступу: http://uadoc.zavantag.com

57. Біометрична автентифікація [Електронний ресурс] // Режим доступу: http://ukrdoc.com.ua/

58. Начало работы с Visual Studio URL: https://visualstudio.microsoft.com/ru/vs/getting-started/

59. Visual Studio URL: https://docs.microsoft.com/ru-ru/visualstudio/get-started/visual-studio-ide?view=vs-2019

60. Методичні вказівки до виконання студентами-магістрантами економічної частини магістерських кваліфікаційних робіт / Уклад. В. О. Козловський – Вінниця: ВНТУ, 2012. – 22 с.

ДОДАТКИ

Додаток А. Технічне завдання

Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

ЗАТВЕРДЖУЮ
Голова секції «Управління інформаційною
безпекою» кафедри МБІС
д.т.н., професор
_____ Ю. Є. Яремчук

«24» вересня 2021 р.

ТЕХНІЧНЕ ЗАВДАННЯ

до магістерської кваліфікаційної роботи на тему:

Удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока
08-42.МКР.009.00.00 ТЗ

Керівник магістерської кваліфікаційної роботи
Керівник к.ф.-м.н., доц. Шиян А.А.
_____

Вінниця – 2021 р.

1 Найменування та область застосування

Програмний засіб для удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

Область застосування: ідентифікація користувачів на основі біометричних даних та qr-коду.

2 Підстави для розробки

Розробка виконується на основі наказу ректора ВНТУ 24 вересня 2021 року №277.

3 Мета та призначення розробки

3.1 Мета розробки

Метою роботи є розроблення та реалізація програмного засобу для удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

3.2 Призначення

Розроблений програмний продукт забезпечує можливість підвищення достовірності ідентифікації користувачів на основі вдосконаленого методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

4 Джерела розробки

1. Хорошко В.А. Методы и средства защиты информации / В.А. Хорошко, А.А. Чекатков. – К.: Изд-во Юниор, 2003. – 504 с.

2. Задонский А. Ю. Вопросы аутентификации в информационных системах. – [Електронний ресурс] // Режим доступу: http://www.compserv.ru

3. Чала Л. Е. Методи динамічної ідентифікації користувачів розподілених інформаційних систем. / Л. Е. Чала. – Харків, 2006. – 20 с.

4. Дудатьєв А.В., Каплун В.А., Семеренко В.П. Д 81 Захист програмного забезпечення. Частина 1. Навчальний посібник. – Вінниця: ВНТУ, 2005. – 140 с.

5 Вимоги до програми

5.1 Вимоги до функціональних характеристик

5.1.1 Програмний додаток призначений для ідентифікації користувачів на основі біометричних даних та qr-коду.

5.1.2 Реалізація методу не повинна вимагати спеціальних ліцензійних програмних додатків

5.1.3 Програмний додаток повинен мати зручний, легкий у розумінні користувача інтерфейс.

5.2 Вимоги до надійності:

5.2.1 Програмний додаток повинен бути працездатним продуктом, функціонувати без помилок.

5.2.2 Програмний додаток повинен працювати без помилок, у випадку виникнення критичних ситуацій необхідно передбачити виведення відповідних повідомлень.

5.4 Вимоги до складу і параметрів технічних засобів:

– оперативна пам'ять – не менше 512 Mb.

5.5 Вимоги до інформаційної та програмної сумісності – будь-яка операційна система.

6 Вимоги до програмної документації

6.1 Обов’язкова поетапна інструкція для майбутніх користувачів, наведена у пункті 3.3

7 Вимоги до технічного захисту інформації

6.1 Необхідно забезпечити контроль доступу користувачів до системи на основі біометричних даних та qr-коду.

8 Техніко-економічні показники

8.1 Програмний додаток має бути простим у використанні, легко змінюваним, мати можливість швидкого введення змін.

8.2 Витрати на програмні продукти, що використовуються в ході розробки мають бути мінімальними.

9 Стадії та етапи розробки

№	Назва етапів МКР	Початок	Закінчення
1	Визначення напрямку магістерської роботи, формулювання та затвердження теми	01.09.2021	26.09.2021
2	Аналіз предметної області обраної теми	27.09.2021	05.10.2021
3	Апробація отриманих результатів	06.10.2021	15.10.2021
4	Розробка алгоритму роботи	16.10.2021	31.10.2021
5	Написання магістерської роботи на основі розробленої теми	01.11.2021	14.11.2021
6	Розробка економічної частини	15.11.2020	21.11.2021
7	Передзахист магістерської кваліфікаційної роботи	22.11.2021	25.11.2021
8	Виправлення, уточнення, корегування магістерської кваліфікаційної роботи	26.11.2021	19.12.2021
9	Захист магістерської кваліфікаційної роботи	20.12.2021	23.12.2021

10 Порядок контролю та прийому

10.1 До приймання магістерської кваліфікаційної роботи надається:

– ПЗ до магістерської кваліфікаційної роботи;

– демонстрація результату магістерської кваліфікаційної роботи;

– презентація;

– відзив керівника роботи;

– відзив рецензента.

Технічне завдання до виконання прийняв _____ Шадюк В.С.

Додаток Б. Лістинг Registration.xaml

```
using Emgu.CV;
using Emgu.CV.CvEnum;
using Emgu.CV.Face;
using Emgu.CV.Structure;
using Emgu.CV.Util;
using FaceDetectionAndRecognition;
using MessagingToolkit.QRCode.Codec;
using Microsoft.Win32;
using Newtonsoft.Json;
using QrEyeFinger.Context;
using SpeechEyeRecognize.Models;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Drawing.Imaging;
using System.IO;
using System.Runtime.CompilerServices;
using System.Timers;
using System.Windows;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Threading;

namespace QrEyeFinger
{
   public partial class Registration : Window, INotifyPropertyChanged
   {
      public Registration()
      {
         InitializeComponent();
         captureTimer = new Timer()
         {
            Interval = Config.TimerResponseValue

         };
         captureTimer.Elapsed += CaptureTimer_Elapsed;
      }


      private RecognizeContext _context = new RecognizeContext();
      private string encoderText = "";

      #region Properties
```

```
        public event PropertyChangedEventHandler PropertyChanged;
        private VideoCapture videoCapture;
        private CascadeClassifier haarCascade;
        private Image<Bgr, Byte> bgrFrame = null;
        private Image<Gray, Byte> detectedFace = null;
        private List<FaceData> faceList = new List<FaceData>();
        private VectorOfMat imageList = new VectorOfMat();
        private List<string> nameList = new List<string>();
        private VectorOfInt labelList = new VectorOfInt();
        private bool active = true;
        private bool enableAutoNavigate = true;
        private string ResourceImage = $"{
Path.GetDirectoryName(AppDomain.CurrentDomain.BaseDirectory)}/ResourceImage";

        private EigenFaceRecognizer recognizer;
        private Timer captureTimer;
        private Type type;
        private bool show = true;

        #region FaceName

        private string faceName;

        public string FaceName
        {
          get { return faceName; }
          set
          {
            faceName = value.ToUpper();
            lblFaceName.Dispatcher.Invoke(DispatcherPriority.Normal, new Action(() => {
lblFaceName.Content = faceName; }));
            NotifyPropertyChanged();
          }
        }

        #endregion


        private Bitmap cameraCapture;

        public Bitmap CameraCapture
        {
          get { return cameraCapture; }
          set
          {
            cameraCapture = value;
            imgCamera.Dispatcher.Invoke(DispatcherPriority.Normal,
              new Action(() => { imgCamera.Source = BitmapToImageSource(cameraCapture);
```

```
}));
            NotifyPropertyChanged();
        }
    }

    #endregion


    #region CameraCaptureFaceImage

    private Bitmap cameraCaptureFace;

    public Bitmap CameraCaptureFace
    {
        get { return cameraCaptureFace; }
        set
        {
            cameraCaptureFace = value;
            NotifyPropertyChanged();
        }
    }

    public void OnLoad()
    {

        GetFacesList();
        if (videoCapture == null)
        {
            videoCapture = new VideoCapture(Config.ActiveCameraIndex);
            videoCapture.SetCaptureProperty(CapProp.Fps, 30);
            videoCapture.SetCaptureProperty(CapProp.FrameHeight, 450);
            videoCapture.SetCaptureProperty(CapProp.FrameWidth, 370);
            captureTimer.Start();
        }

        Init();
    }

    private void Init()
    {
        var resourceFile = $"{AppDomain.CurrentDomain.BaseDirectory}/ResourceFile.json";
        bool exists = System.IO.File.Exists(resourceFile);

        if (!exists)
            System.IO.File.Create(resourceFile);

        bool existsResourceImage = System.IO.Directory.Exists(ResourceImage);
```

```
            if (!existsResourceImage)
                System.IO.Directory.CreateDirectory(ResourceImage);
        }

        protected virtual void NotifyPropertyChanged([CallerMemberName] String
propertyName = "")
        {
            var handler = PropertyChanged;
            if (handler != null) handler(this, new PropertyChangedEventArgs(propertyName));
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            OnLoad();
        }
        private void CaptureTimer_Elapsed(object sender, ElapsedEventArgs e)
        {
            ProcessFrame();
        }
        #endregion

        #region Method

        public enum Type
        {
            Login,
            Registration
        }

        public void GetFacesList()
        {
            //haar cascade classifier
            if (!File.Exists(Config.HaarCascadePath))
            {
                string text = "Cannot find Haar cascade data file:\n\n";
                text += Config.HaarCascadePath;
                MessageBoxResult result = MessageBox.Show(text, "Error",
                    MessageBoxButton.OK, MessageBoxImage.Error);
            }

            haarCascade = new CascadeClassifier(Config.HaarCascadePath);
            faceList.Clear();
            string line;
            FaceData faceInstance = null;

            // Create empty directory / file for face data if it doesn't exist
            if (!Directory.Exists(Config.FacePhotosPath))
            {
```

```
            Directory.CreateDirectory(Config.FacePhotosPath);
        }

        if (!File.Exists(Config.FaceListTextFile))
        {
            string text = "Cannot find face data file:\n\n";
            text += Config.FaceListTextFile + "\n\n";
            text += "If this is your first time running the app, an empty file will be created for
you.";
            MessageBoxResult result = MessageBox.Show(text, "Warning",
                MessageBoxButton.OK, MessageBoxImage.Warning);
            switch (result)
            {
                case MessageBoxResult.OK:
                    String dirName = Path.GetDirectoryName(Config.FaceListTextFile);
                    Directory.CreateDirectory(dirName);
                    File.Create(Config.FaceListTextFile).Close();
                    break;
            }
        }

        StreamReader reader = new StreamReader(Config.FaceListTextFile);
        int i = 0;
        while ((line = reader.ReadLine()) != null)
        {
            string[] lineParts = line.Split(':');
            faceInstance = new FaceData();
            faceInstance.FaceImage =
                new Image<Gray, byte>(Config.FacePhotosPath + lineParts[0] +
Config.ImageFileExtension);
            faceInstance.PersonName = lineParts[1];
            faceList.Add(faceInstance);
        }

        foreach (var face in faceList)
        {
            imageList.Push(face.FaceImage.Mat);
            nameList.Add(face.PersonName);
            labelList.Push(new[] { i++ });
        }

        reader.Close();

        // Train recogniser
        if (imageList.Size > 0)
        {
            recognizer = new EigenFaceRecognizer(imageList.Size);
            recognizer.Train(imageList, labelList);
```

```
    }
}

private void ProcessFrame()
{
    if (active)
    {
        bgrFrame = videoCapture.QueryFrame().ToImage<Bgr, Byte>();

        if (bgrFrame != null)
        {
            try
            {
                //for emgu cv bug
                Image<Gray, byte> grayframe = bgrFrame.Convert<Gray, byte>();

                Rectangle[] faces = haarCascade.DetectMultiScale(grayframe, 1.2, 10,
                    new System.Drawing.Size(50, 50), new System.Drawing.Size(200, 200));

                //detect face
                FaceName = "No face detected";
                foreach (var face in faces)
                {
                    bgrFrame.Draw(face, new Bgr(255, 255, 0), 2);
                    detectedFace = bgrFrame.Copy(face).Convert<Gray, byte>();
                    FaceRecognition();
                    break;
                }

                CameraCapture = bgrFrame.ToBitmap();
            }
            catch (Exception ex)
            {
                throw new ArgumentException("Error: " + ex);
                //todo log
            }

        }
    }
}


private void FaceRecognition()
{
    if (imageList.Size != 0)
    {
        //Eigen Face Algorithm
        FaceRecognizer.PredictionResult result =
```

```
recognizer.Predict(detectedFace.Resize(100, 100, Inter.Cubic));
                    FaceName = nameList[result.Label];
                    CameraCaptureFace = detectedFace.ToBitmap();
                }
                else
                {
                    FaceName = "Please Add Face";
                }
            }

            private BitmapImage BitmapToImageSource(Bitmap bitmap)
            {
                using (MemoryStream memory = new MemoryStream())
                {
                    bitmap.Save(memory, System.Drawing.Imaging.ImageFormat.Bmp);
                    memory.Position = 0;
                    BitmapImage bitmapimage = new BitmapImage();
                    bitmapimage.BeginInit();
                    bitmapimage.StreamSource = memory;
                    bitmapimage.CacheOption = BitmapCacheOption.OnLoad;
                    bitmapimage.EndInit();

                    return bitmapimage;
                }
            }

            #endregion
            private void RegistrFace(string personName)
            {
                if (detectedFace == null)
                {
                    MessageBox.Show("No face detected.");
                    return;
                }

                //Save detected face
                detectedFace = detectedFace.Resize(100, 100, Inter.Cubic);
                detectedFace.Save(Config.FacePhotosPath + "face" + (faceList.Count + 1) +
Config.ImageFileExtension);
                StreamWriter writer = new StreamWriter(Config.FaceListTextFile, true);
                // string personName = Microsoft.VisualBasic.Interaction.InputBox("Your Name");
                writer.WriteLine(String.Format("face{0}:{1}", (faceList.Count + 1), personName));
                writer.Close();
                GetFacesList();
                MessageBox.Show("Реєстрація пройшла успішно.");
            }

            private void RegistrationVoid()
```

```
        {
            try
            {
                if (!VerifyData())
                {
                    MessageBox.Show("Невдала спроба реєстрації");
                    return;
                }

                RegistrFace(UserName.Text);

                var fileName = RegisterFingerPrint(UserName.Text);


                var user = new User()
                {
                    UserName = UserName.Text,
                    FingerPrintName = fileName,
                    EncoderText = encoderText
                };

                var users = GetUsers();

                if (users == null)
                {
                    users = new List<User>() { user };
                }
                else
                {
                    users.Add(user);
                }
                string json = JsonConvert.SerializeObject(users);
                File.WriteAllText($"{
Path.GetDirectoryName(System.AppDomain.CurrentDomain.BaseDirectory)}/ResourceFile.json",
json);

                var w = Application.Current.Windows[0];
                w.Hide();
                captureTimer.Stop();
                videoCapture.Dispose();

                CommonPage commonpage = new CommonPage();
                commonpage.AuthUserName.Content = user.UserName;
                commonpage.Login.Visibility = Visibility.Hidden;
                commonpage.Regist.Visibility = Visibility.Hidden;
                commonpage.Exit.Visibility = Visibility.Visible;

                commonpage.ShowDialog();
```

```
                MessageBox.Show("Успішно зареєстровано нового користувача");

                w.Close();
            }
            catch (Exception exception)
            {
                Console.WriteLine(exception);
                MessageBox.Show("Невдала спроба реєстрації");
            }
        }

        private string RegisterFingerPrint(string personName)
        {

            var extension = Path.GetExtension(pathToFile.Text);
            var newFileName = $"{Guid.NewGuid()}{extension}";
            var resPath = $"{ResourceImage}\\{newFileName}";

            var encoder = new PngBitmapEncoder();

encoder.Frames.Add(BitmapFrame.Create((BitmapSource)fingerPrintImage.Source));
            using (FileStream stream = new FileStream(resPath, FileMode.Create))
                encoder.Save(stream);

            return newFileName;
        }

        private bool VerifyData()
        {
            if (String.IsNullOrEmpty(UserName.Text))
            {
                MessageBox.Show("Введіть логін");
                return false;
            }


            if (imgCamera.Source == null)
            {
                MessageBox.Show("Сталася помилка. Не вдалося зчитати райдужку ока");
                return false;
            }

            return true;
        }

        private void Auth_OnClosed(object sender, EventArgs e)
```

```
        {
            captureTimer.Stop();
            videoCapture.Dispose();
        }

        private void Button_Click_2(object sender, RoutedEventArgs e)
        {
            RegButton.Background = new SolidColorBrush(Colors.LightGray);

            qrGen.Visibility = Visibility.Visible;

            RegistrationVoid();
        }

        private List<User> GetUsers()
        {
            List<User> users = new List<User>();

            var path = $"{
Path.GetDirectoryName(AppDomain.CurrentDomain.BaseDirectory)}/ResourceFile.json";

            //bool exists = System.IO.File.Exists(path);

            //if (!exists)
            //    System.IO.File.Create(path);

            using (StreamReader r = new StreamReader(path))
            {
                string json = r.ReadToEnd();
                users = JsonConvert.DeserializeObject<List<User>>(json);
            }

            return users;
        }

        private void Button_Click_4(object sender, RoutedEventArgs e)
        {

            OpenFileDialog dlg = new OpenFileDialog();
            //   dlg.InitialDirectory = "c:\\";
            //   dlg.Filter = "Image files (*.jpg)|*.jpg|All Files (*.*)|*.*";
            dlg.RestoreDirectory = true;

            if (dlg.ShowDialog() == true)
            {
                string selectedFileName = dlg.FileName;
                BitmapImage bitmap = new BitmapImage();
                bitmap.BeginInit();
```

```
            bitmap.UriSource = new Uri(selectedFileName);
            bitmap.EndInit();
            fingerPrintImage.Source = bitmap;
        }


    }

    private void qrGen_Click(object sender, RoutedEventArgs e)
    {
        SaveFileDialog sfd = new SaveFileDialog();
        sfd.Filter = @"JPEG file|*.jpg;*.jpeg;";
        sfd.ValidateNames = true;


        if (sfd.ShowDialog() == true)
        {
            QRCodeEncoder encoder = new QRCodeEncoder();
            encoder.QRCodeScale = 8;
            encoderText = Guid.NewGuid().ToString();
            Bitmap bmp = encoder.Encode(encoderText);
            bmp.Save(sfd.FileName, ImageFormat.Jpeg);

            //qrGen.IsCancel = true;
        }

        string selectedFileName = sfd.FileName;
        BitmapImage bitmap = new BitmapImage();
        bitmap.BeginInit();
        bitmap.UriSource = new Uri(selectedFileName);
        bitmap.EndInit();
        qrPrintImage.Source = bitmap;

        qrGen.Visibility = Visibility.Hidden;
    }
  }
}
```

Додаток В. Лістинг Login.xaml

```
using Emgu.CV;
using Emgu.CV.CvEnum;
using Emgu.CV.Face;
using Emgu.CV.Structure;
using Emgu.CV.Util;
using FaceDetectionAndRecognition;
using Microsoft.Win32;
using Newtonsoft.Json;
using PatternRecognition.FingerprintRecognition.Core;
using PatternRecognition.FingerprintRecognition.FeatureExtractors;
using PatternRecognition.FingerprintRecognition.Matchers;
using QrEyeFinger.Context;
using SpeechEyeRecognize.Models;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Runtime.CompilerServices;
using System.Runtime.InteropServices;
using System.Timers;
using System.Windows;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Threading;
using MessagingToolkit.QRCode.Codec.Data;

namespace QrEyeFinger
{
   public partial class Login : Window, INotifyPropertyChanged
   {
      public Login()
      {
         InitializeComponent();
         captureTimer = new Timer()
         {
            Interval = Config.TimerResponseValue

         };
         captureTimer.Elapsed += CaptureTimer_Elapsed;
      }


      private RecognizeContext _context = new RecognizeContext();
```

```
        #region Properties
        private string ResourceImage = $"{
Path.GetDirectoryName(AppDomain.CurrentDomain.BaseDirectory)}/ResourceImage";


        public event PropertyChangedEventHandler PropertyChanged;
        private VideoCapture videoCapture;
        private CascadeClassifier haarCascade;
        private Image<Bgr, Byte> bgrFrame = null;
        private Image<Gray, Byte> detectedFace = null;
        private List<FaceData> faceList = new List<FaceData>();
        private VectorOfMat imageList = new VectorOfMat();
        private List<string> nameList = new List<string>();
        private VectorOfInt labelList = new VectorOfInt();
        private bool active = true;
        private bool enableAutoNavigate = true;

        private EigenFaceRecognizer recognizer;
        private Timer captureTimer;
        private Type type;
        private bool show = true;

        #region FaceName

        private string faceName;

        public string FaceName
        {
          get { return faceName; }
          set
          {
            faceName = value.ToUpper();
            lblFaceName.Dispatcher.Invoke(DispatcherPriority.Normal, new Action(() => {
lblFaceName.Content = faceName; }));
            NotifyPropertyChanged();
          }
        }

        #endregion


        private Bitmap cameraCapture;

        public Bitmap CameraCapture
        {
          get { return cameraCapture; }
          set
```

```
            {
                cameraCapture = value;
                imgCamera.Dispatcher.Invoke(DispatcherPriority.Normal,
                    new Action(() => { imgCamera.Source = BitmapToImageSource(cameraCapture);
}));
                NotifyPropertyChanged();
            }
        }

        #endregion

        #region CameraCaptureFaceImage

        private Bitmap cameraCaptureFace;

        public Bitmap CameraCaptureFace
        {
            get { return cameraCaptureFace; }
            set
            {
                cameraCaptureFace = value;
                NotifyPropertyChanged();
            }
        }

        public void OnLoad()
        {
            qwImage.Visibility = Visibility.Visible;
            GetFacesList();
            if (videoCapture == null)
            {
                videoCapture = new VideoCapture(Config.ActiveCameraIndex);
                videoCapture.SetCaptureProperty(CapProp.Fps, 30);
                videoCapture.SetCaptureProperty(CapProp.FrameHeight, 450);
                videoCapture.SetCaptureProperty(CapProp.FrameWidth, 370);
                captureTimer.Start();
            }
        }

        protected virtual void NotifyPropertyChanged([CallerMemberName] String
propertyName = "")
        {
            var handler = PropertyChanged;
            if (handler != null) handler(this, new PropertyChangedEventArgs(propertyName));
        }

        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
```

```
    OnLoad();
}
private void CaptureTimer_Elapsed(object sender, ElapsedEventArgs e)
{
    ProcessFrame();
}

private void OpenVideoFile_Click(object sender, RoutedEventArgs e)
{
    OpenFileDialog openDialog = new OpenFileDialog();
    if (openDialog.ShowDialog().Value == true)
    {
        captureTimer.Stop();
        videoCapture.Dispose();

        videoCapture = new VideoCapture(openDialog.FileName);
        captureTimer.Start();
        this.Title = openDialog.FileName;
        return;
    }
}

#endregion

#region Method

public void GetFacesList()
{
    //haar cascade classifier
    if (!File.Exists(Config.HaarCascadePath))
    {
        string text = "Cannot find Haar cascade data file:\n\n";
        text += Config.HaarCascadePath;
        MessageBoxResult result = MessageBox.Show(text, "Error",
            MessageBoxButton.OK, MessageBoxImage.Error);
    }

    haarCascade = new CascadeClassifier(Config.HaarCascadePath);
    faceList.Clear();
    string line;
    FaceData faceInstance = null;

    // Create empty directory / file for face data if it doesn't exist
    if (!Directory.Exists(Config.FacePhotosPath))
    {
        Directory.CreateDirectory(Config.FacePhotosPath);
    }
```

```
            if (!File.Exists(Config.FaceListTextFile))
            {
               string text = "Cannot find face data file:\n\n";
               text += Config.FaceListTextFile + "\n\n";
               text += "If this is your first time running the app, an empty file will be created for
you.";
               MessageBoxResult result = MessageBox.Show(text, "Warning",
                  MessageBoxButton.OK, MessageBoxImage.Warning);
               switch (result)
               {
                  case MessageBoxResult.OK:
                     String dirName = Path.GetDirectoryName(Config.FaceListTextFile);
                     Directory.CreateDirectory(dirName);
                     File.Create(Config.FaceListTextFile).Close();
                     break;
               }
            }

            StreamReader reader = new StreamReader(Config.FaceListTextFile);
            int i = 0;
            while ((line = reader.ReadLine()) != null)
            {
               string[] lineParts = line.Split(':');
               faceInstance = new FaceData();
               faceInstance.FaceImage =
                  new Image<Gray, byte>(Config.FacePhotosPath + lineParts[0] +
Config.ImageFileExtension);
               faceInstance.PersonName = lineParts[1];
               faceList.Add(faceInstance);
            }

            foreach (var face in faceList)
            {
               imageList.Push(face.FaceImage.Mat);
               nameList.Add(face.PersonName);
               labelList.Push(new[] { i++ });
            }

            reader.Close();

            // Train recogniser
            if (imageList.Size > 0)
            {
               recognizer = new EigenFaceRecognizer(imageList.Size);
               recognizer.Train(imageList, labelList);
            }
         }
```

```
        private void ProcessFrame()
        {
            if (active)
            {
                bgrFrame = videoCapture.QueryFrame().ToImage<Bgr, Byte>();

                if (bgrFrame != null)
                {
                    try
                    {
                        //for emgu cv bug
                        Image<Gray, byte> grayframe = bgrFrame.Convert<Gray, byte>();

                        Rectangle[] faces = haarCascade.DetectMultiScale(grayframe, 1.2, 10,
                            new System.Drawing.Size(50, 50), new System.Drawing.Size(200, 200));

                        //detect face
                        FaceName = "No face detected";
                        foreach (var face in faces)
                        {
                            bgrFrame.Draw(face, new Bgr(255, 255, 0), 2);
                            detectedFace = bgrFrame.Copy(face).Convert<Gray, byte>();
                            FaceRecognition();
                            break;
                        }

                        CameraCapture = bgrFrame.ToBitmap();
                    }
                    catch (Exception ex)
                    {
                        throw new ArgumentException("Error: " + ex);
                        //todo log
                    }

                }
            }
        }


        private void FaceRecognition()
        {
            if (imageList.Size != 0)
            {
                //Eigen Face Algorithm
                FaceRecognizer.PredictionResult result =
recognizer.Predict(detectedFace.Resize(100, 100, Inter.Cubic));
                FaceName = nameList[result.Label];
                CameraCaptureFace = detectedFace.ToBitmap();
```

```
            }
            else
            {
                FaceName = "Please Add Face";
            }
        }

        private BitmapImage BitmapToImageSource(Bitmap bitmap)
        {
            using (MemoryStream memory = new MemoryStream())
            {
                bitmap.Save(memory, System.Drawing.Imaging.ImageFormat.Bmp);
                memory.Position = 0;
                BitmapImage bitmapimage = new BitmapImage();
                bitmapimage.BeginInit();
                bitmapimage.StreamSource = memory;
                bitmapimage.CacheOption = BitmapCacheOption.OnLoad;
                bitmapimage.EndInit();

                return bitmapimage;
            }
        }

        #endregion
        private void RegistrFace(string personName)
        {
            if (detectedFace == null)
            {
                MessageBox.Show("No face detected.");
                return;
            }

            //Save detected face
            detectedFace = detectedFace.Resize(100, 100, Inter.Cubic);
            detectedFace.Save(Config.FacePhotosPath + "face" + (faceList.Count + 1) +
Config.ImageFileExtension);
            StreamWriter writer = new StreamWriter(Config.FaceListTextFile, true);
            // string personName = Microsoft.VisualBasic.Interaction.InputBox("Your Name");
            writer.WriteLine(String.Format("face{0}:{1}", (faceList.Count + 1), personName));
            writer.Close();
            GetFacesList();
            MessageBox.Show("Реєстрація пройшла успішно.");
        }


        private void LoginVoid()
        {
```

```
            if (!VerifyData())
               return;

            var userList = GetUsers();

            if (!userList.Any())
               MessageBox.Show("Кориистувача не знайдено");

            var user = userList.FirstOrDefault(x => x.UserName == UserName.Text);

            var faceIdentification = false;

            if (!String.IsNullOrEmpty(lblFaceName.Content.ToString().ToLower()) &&
lblFaceName.Content.ToString() != "No face detected") ;
            faceIdentification = true;

            if (faceIdentification)
            {
               var score = Convert.ToDecimal(CompareFingerPrint(user));
               if (score > 60)
               {
                  if (DecodeQr() != user.EncoderText)
                  {
                     MessageBox.Show("Кориистувача не знайдено");
                     return;
                  }

                  var w = Application.Current.Windows[0];
                  w.Hide();
                  captureTimer.Stop();
                  videoCapture.Dispose();

                  MessageBox.Show("Спроба входу була вдалою!");


                  CommonPage commonpage = new CommonPage();
                  commonpage.AuthUserName.Content = user.UserName;

                  commonpage.Login.Visibility = Visibility.Hidden;
                  commonpage.Regist.Visibility = Visibility.Hidden;
                  commonpage.Exit.Visibility = Visibility.Visible;

                  commonpage.ShowDialog();

                  captureTimer = null;
                  active = false;
                  enableAutoNavigate = false;
```

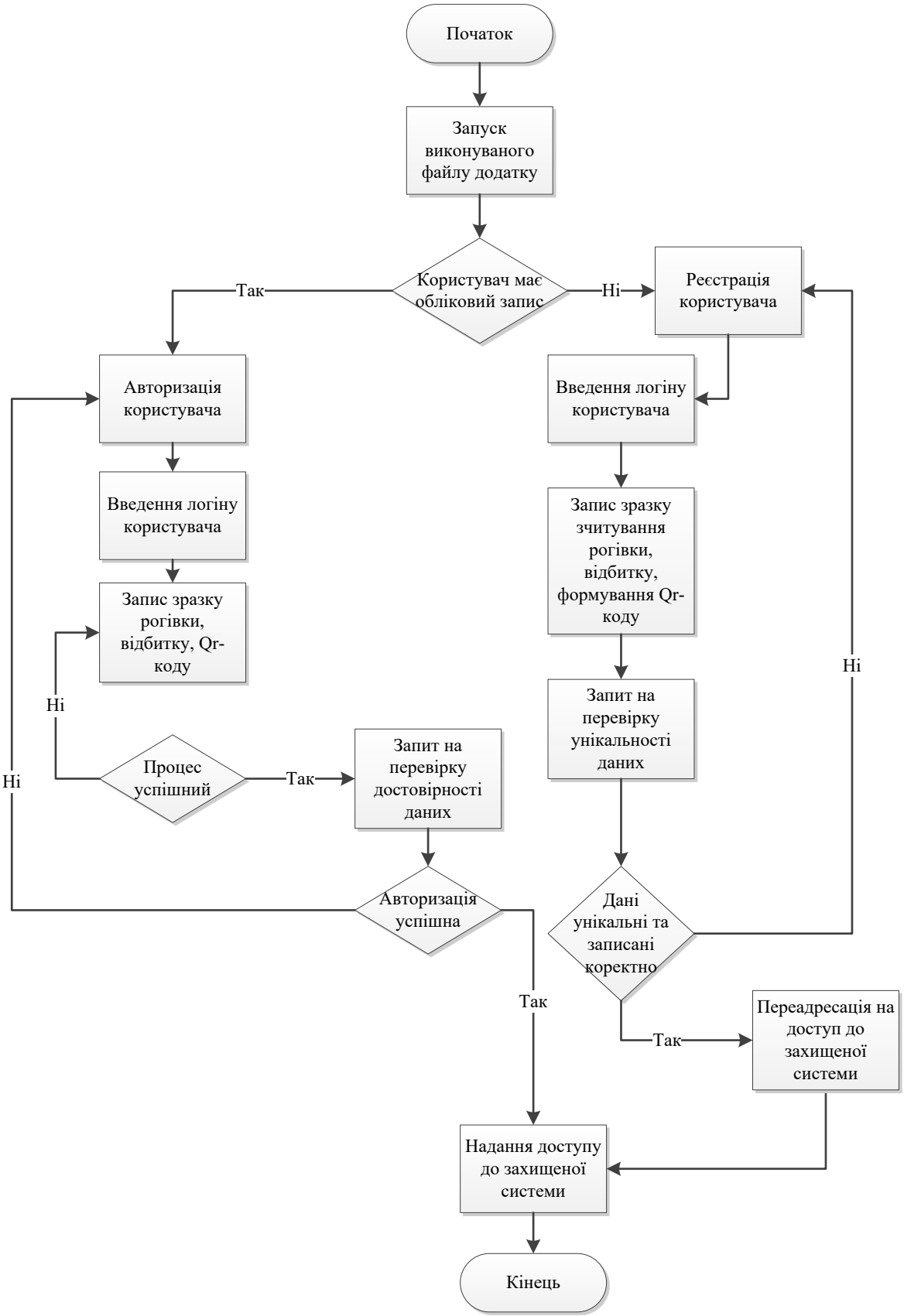
```
                w.Close();

            }
            else
            {
                MessageBox.Show($"Невдала спроба ідентифікації!");
            }
        }
        else
        {
            MessageBox.Show("Невдала спроба ідентифікації");
        }
    }
    private string CompareFingerPrint(User user)
    {
        Change_ResolutionByPath(Path.Combine(ResourceImage, user.FingerPrintName));
        var tempFile = SaveToTempDir();
        Change_ResolutionByPath(tempFile);

        var fingerprintImg1 = ImageLoader.LoadImage(Path.Combine(ResourceImage,
user.FingerPrintName));

        var fingerprintImg2 = ImageLoader.LoadImage(tempFile);

        //// Building feature extractor and extracting features
        var featExtractor = new PNFeatureExtractor() { MtiaExtractor = new
Ratha1995MinutiaeExtractor() };
        var features1 = featExtractor.ExtractFeatures(fingerprintImg1);
        var features2 = featExtractor.ExtractFeatures(fingerprintImg2);

        // Building matcher and matching
        var matcher = new PN();
        double similarity = matcher.Match(features1, features2);
        var score = similarity.ToString("0.000");
        similarity.ToString("0.000");

        return score;
    }

    private string DecodeQr()
    {
        MessagingToolkit.QRCode.Codec.QRCodeDecoder decode = new
MessagingToolkit.QRCode.Codec.QRCodeDecoder();

        var path = SaveQRToTempDir();
        var text = decode.Decode(new QRCodeBitmapImage((Bitmap)
Image.FromFile(path)));
```

```
            return text;
        }

        private string SaveQRToTempDir()
        {
            var path = Path.GetTempFileName();
            var encoder = new PngBitmapEncoder();
            encoder.Frames.Add(BitmapFrame.Create((BitmapSource)qrPrintImage.Source));
            using (FileStream stream = new FileStream(path, FileMode.Create))
                encoder.Save(stream);

            return path;
        }


        private string SaveToTempDir()
        {
            var path = Path.GetTempFileName();
            var encoder = new PngBitmapEncoder();

encoder.Frames.Add(BitmapFrame.Create((BitmapSource)fingerPrintImage.Source));
            using (FileStream stream = new FileStream(path, FileMode.Create))
                encoder.Save(stream);

            return path;
        }

        private Bitmap Change_ResolutionByPath(string file)
        {
            using (Bitmap bitmap = (Bitmap)Image.FromFile(file))
            {
                using (Bitmap newBitmap = new Bitmap(bitmap))
                {
                    newBitmap.SetResolution(500, 500);
                    return newBitmap;
                }
            }
        }


        private bool VerifyData()
        {
            if (String.IsNullOrEmpty(UserName.Text))
            {
                MessageBox.Show("Введіть логін");
                return false;
            }
```

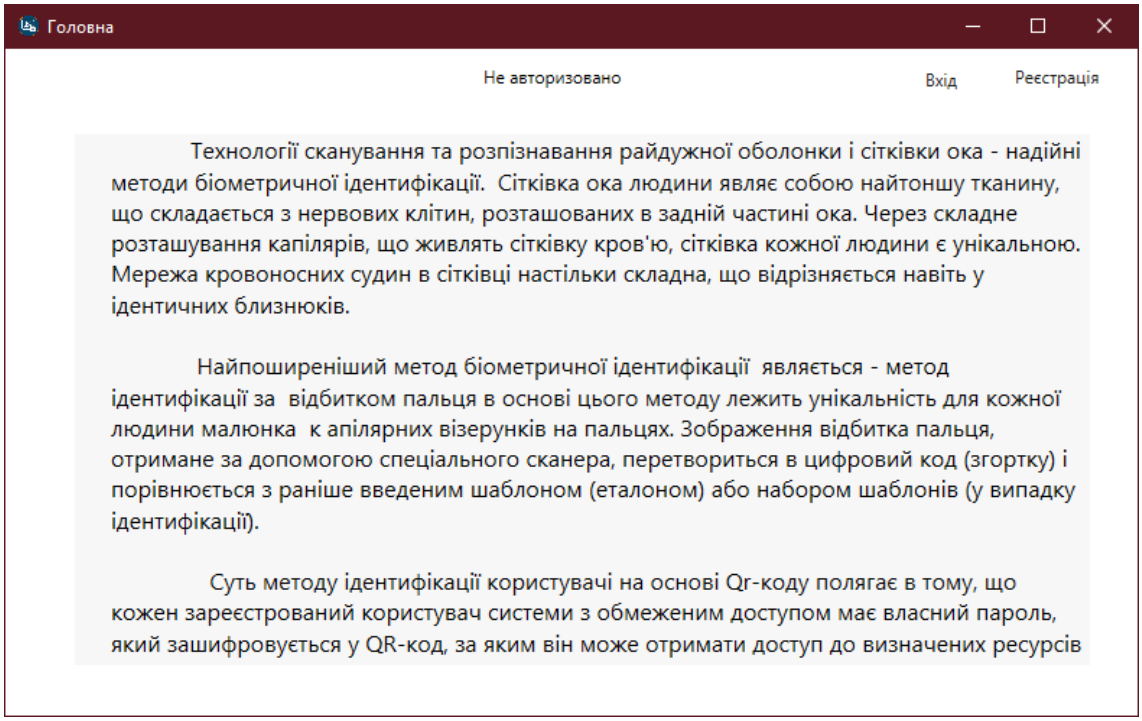
```
            if (imgCamera.Source == null)
            {
                MessageBox.Show("Сталася помилка. Не вдалося зчитати райдужку ока");
                return false;
            }

            return true;
        }

        private void Auth_OnClosed(object sender, EventArgs e)
        {
            captureTimer.Stop();
            videoCapture.Dispose();
        }

        private void Button_Click(object sender, RoutedEventArgs e)
        {
            LoginBtn.Background = new SolidColorBrush(Colors.LightGray);

            //qrButonImage.Visibility = Visibility.Visible;
            //qrPrintImage.Visibility = Visibility.Visible;

            LoginVoid();
        }

        private List<User> GetUsers()
        {
            List<User> users = new List<User>();

            var path = $"{
Path.GetDirectoryName(AppDomain.CurrentDomain.BaseDirectory)}/ResourceFile.json";

            bool exists = System.IO.File.Exists(path);

            if (!exists)
                System.IO.File.Create(path);

            using (StreamReader r = new StreamReader(path))
            {
                string json = r.ReadToEnd();
                users = JsonConvert.DeserializeObject<List<User>>(json);
            }

            return users;
        }

        private void Button_Click_3(object sender, RoutedEventArgs e)
```

```
        {
            OpenFileDialog dlg = new OpenFileDialog();
            //   dlg.InitialDirectory = "c:\\";
            dlg.Filter = @"JPEG file|*.jpg;*.jpeg;";
            dlg.RestoreDirectory = true;

            if (dlg.ShowDialog() == true)
            {
                string selectedFileName = dlg.FileName;
                BitmapImage bitmap = new BitmapImage();
                bitmap.BeginInit();
                bitmap.UriSource = new Uri(selectedFileName);
                bitmap.EndInit();
                qrPrintImage.Source = bitmap;
            }
        }

        private void Button_Click_4(object sender, RoutedEventArgs e)
        {

            OpenFileDialog dlg = new OpenFileDialog();
            //   dlg.InitialDirectory = "c:\\";
            //   dlg.Filter = "Image files (*.jpg)|*.jpg|All Files (*.*)|*.*";
            dlg.RestoreDirectory = true;

            if (dlg.ShowDialog() == true)
            {
                string selectedFileName = dlg.FileName;
                BitmapImage bitmap = new BitmapImage();
                bitmap.BeginInit();
                bitmap.UriSource = new Uri(selectedFileName);
                bitmap.EndInit();
                fingerPrintImage.Source = bitmap;
            }


        }

    }
}
```

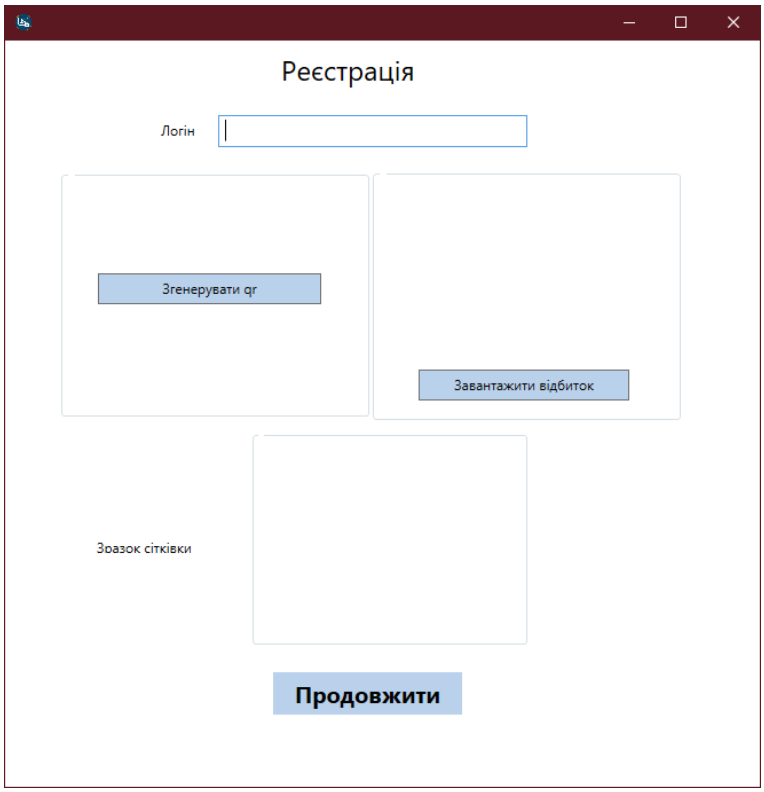
Додаток Г. Алгоритм роботи



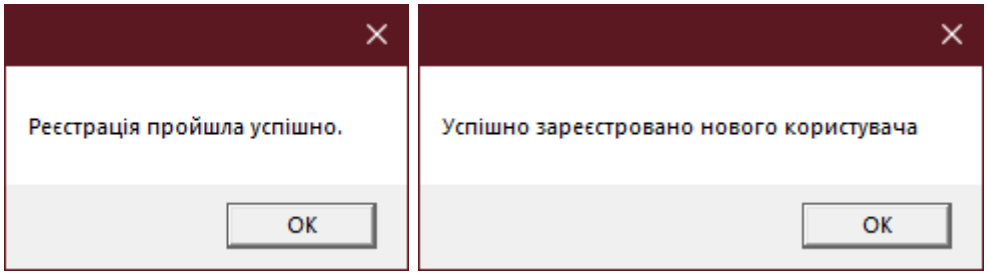
Додаток Д. Інтерфейс користувача



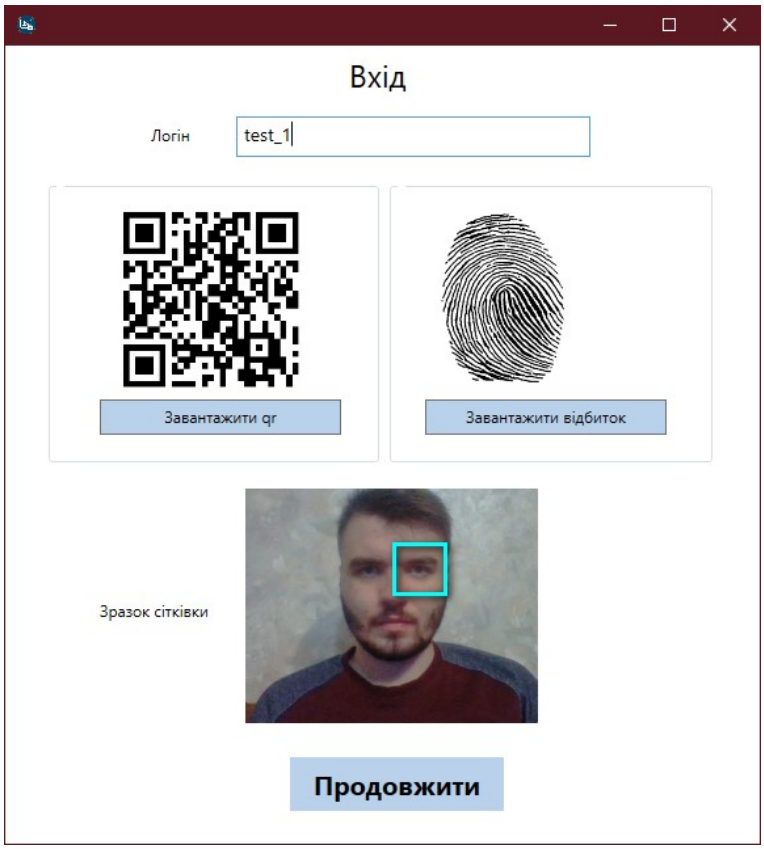
Вигляд головного вікна додатку



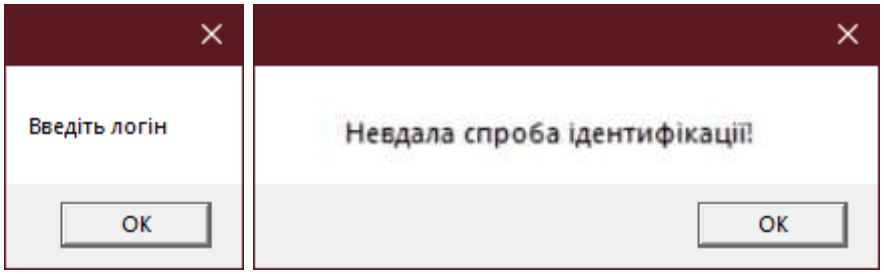
Вигляд вікна реєстрації користувача



Повідомлення про успішну реєстрацію користувача



Вигляд вікна реєстрації користувача із заповненими полями



Повідомлення про відмову у авторизації користувача

Додаток Е. Ілюстративний матеріал

Удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока

Виконав: ст. гр. УБ – 20м Шадюк В.С.

Керівник: к.ф.-м.н., проф. Шиян А.А.

Актуальність:

Актуальність обраної теми обумовлена наступними факторами:

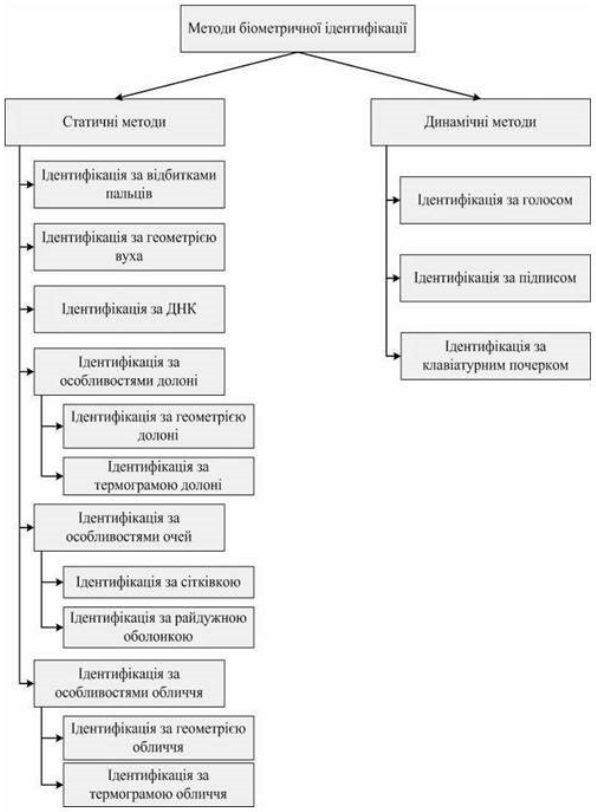
- інтенсивністю розвиток інформаційних систем;
- необхідністю реалізації контролю доступу в інформаційних системах;
- зростанням залежності організації в цілому та людини в частості від правильного функціонування інформаційних систем.

Мета роботи:

- Метою роботи є розробка програми комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока

Біометрична ідентифікація

- Біометрична ідентифікація – це процес порівняння і визначення подібності між даними людини і його біометричних «шаблоном».



Переваги біометричної ідентифікації порівняно з парольною

Технологія	Плюси	Мінуси
Пароль, PIN - код, ключові слова і так далі	1. Низька вартість 2. Немає необхідності носити аутентифікатор 3. Висока швидкість розпізнавання при роботі з великими базами	1. Користувачі часто застосовують короткі легко підбирані паролі 2. Існують можливості перехоплення паролів 3. Необхідність регулярно міняти паролі 4. Складно запам'ятати велику кількість паролів
Відбиток пальця, особливості веселкової оболонки ока, форми кисті рук і т.д.	1. Унеможливлюється несанкціонованого використання аутентифікатора; 2. Забезпечується висока міра захисту від імітації; 3. Відпадає необхідність обов'язкового носіння аутентифікатора; 4. Виключається втрата або псування аутентифікатора, забудькуватість, передача третім особам і т.п.); 5. Відсутні витрати на виготовлення, купівлю	1. Неможливо отримати статичний ключ 2. Можливість помилок першого і другого роду (пропуск або помилкова тривога)

Переваги системи біометричної аутентифікації на основі сканування рогівки ока

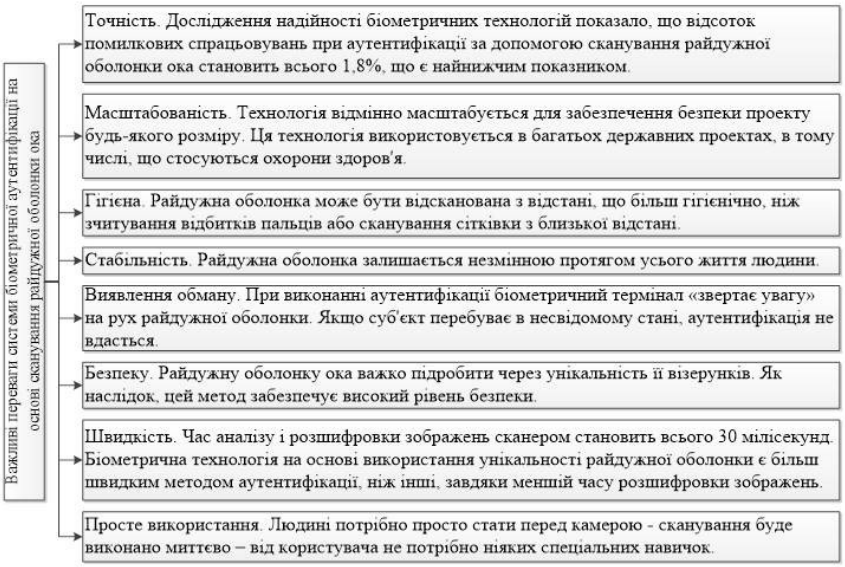
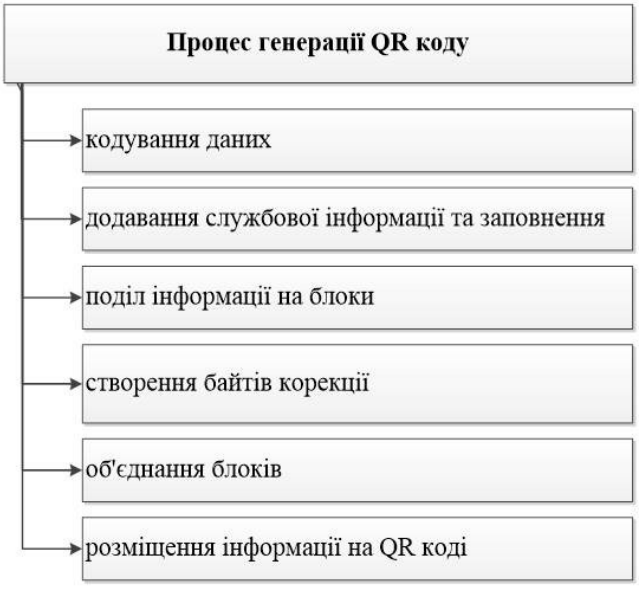
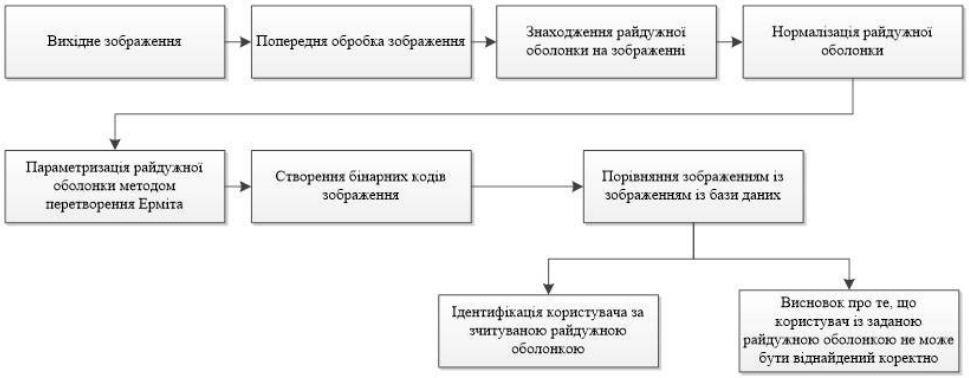


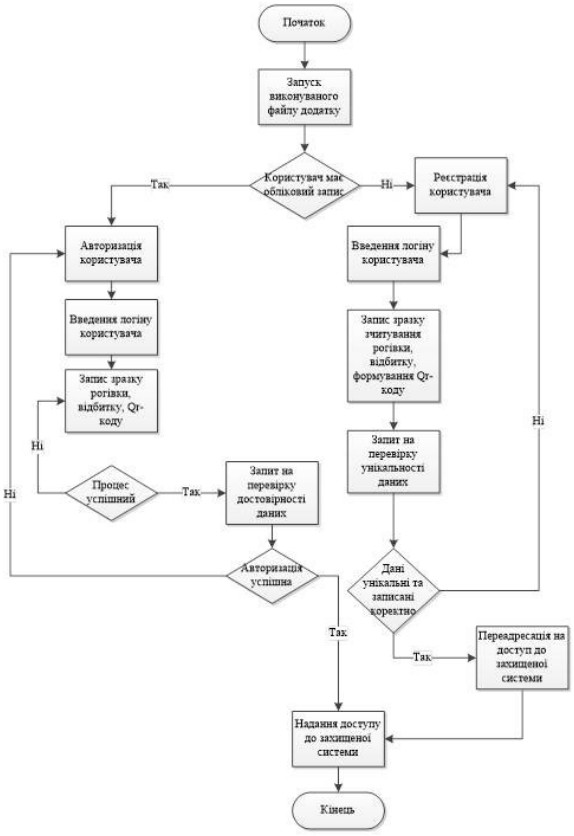
Схема процесу генерації qr-коду



Загальна схема розв'язання задачі ідентифікації за рогівкою ока



Алгоритм роботи комплексної системи ідентифікації

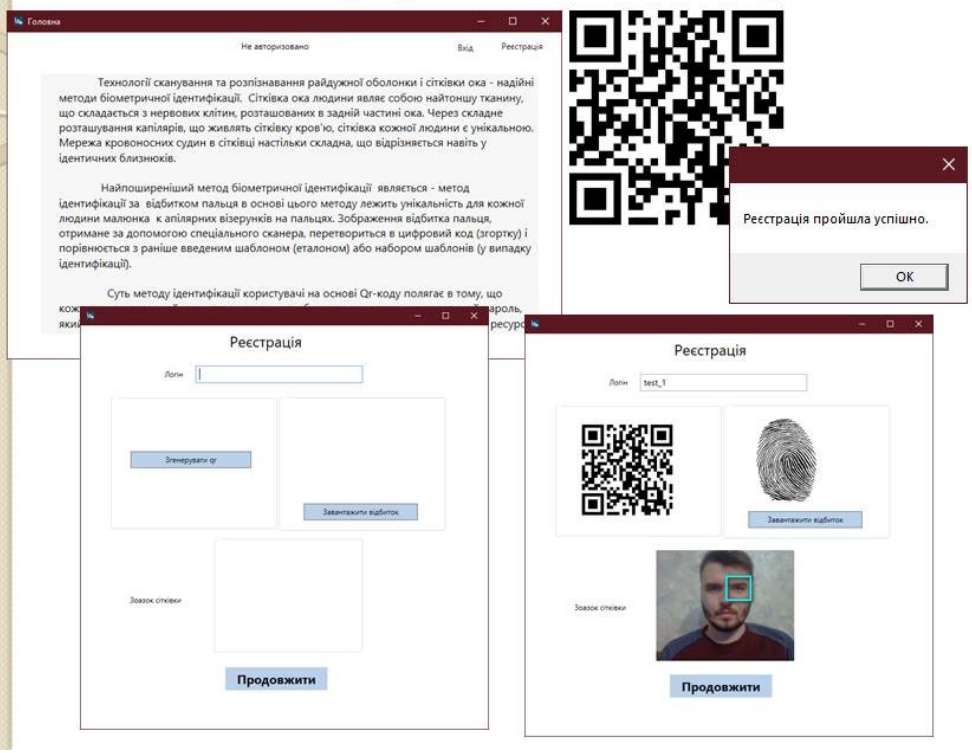


Засоби програмної реалізації

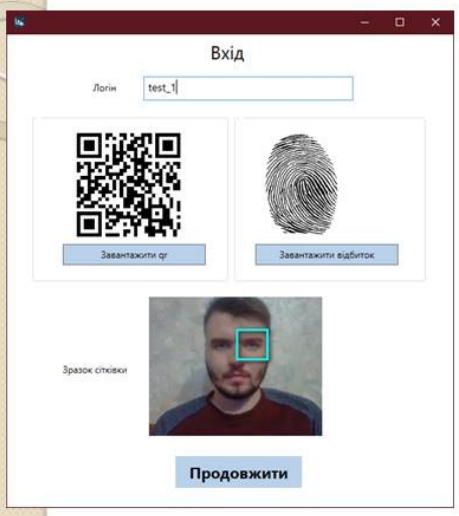
Для програмної реалізації розроблюваної системи ідетифікації користувачів використовують такі засоби:

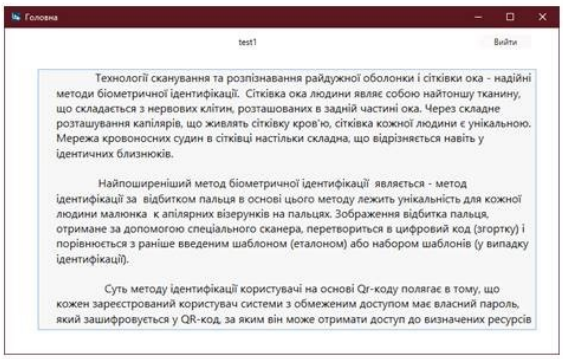
- мова об’єктно-орієнтованого програмування C#;
- середовище Visual Studio;
- бібліотека комп'ютерного зору OpenCV.

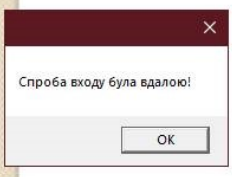
Інтерфейс додатку

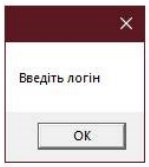


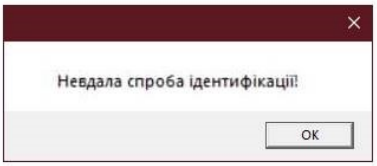
Інтерфейс додатку











Висновки

Отже, в магістерській дипломній роботі здійснювалась розробка програми комплексної автентифікації до системи захисту з використанням ідентифікації на основі QR-коду, відбитку пальця та рогівки ока

Варто відзначити такі переваги біометричної системи, що пояснюють актуальність та доцільність її розробки:

- надійність і швидкість аутентифікації;
- високий рівень безпеки: біометричні ознаки людини неповторні, що зводить до мінімуму кількість помилок при впізнанні;
- дані біометричних характеристик неможливо втратити або забути;
- пристрої біометричної автентифікації зручні в користуванні і бюджетних в експлуатації.

Таким чином, в результаті здійсненої роботи, було досягнуто поставленої мети, а саме розроблено програму комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока.

Дякую за увагу!

Додаток Ж. Протокол перевірки

ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ

Назва роботи: Удосконалення методу комплексної ідентифікації користувача віддаленого доступу на основі QR-коду, відбитку пальця та рогівки ока
Тип роботи: Магістерська кваліфікаційна робота
Підрозділ: Факультет МІБ, кафедра менеджменту та безпеки інформаційних систем,
гр. УБ-20м
Науковий керівник Шиян А.А., доцент кафедри МБІС, к.ф.-м.н.

Показники звіту подібності

Plagiat.pl (StrikePlagiarism)		Unicheck	
КП1			
КП2		Оригінальність	84 %
Тривога/Білі знаки	/	Схожість	16 %

Аналіз звіту подібності (відмінити подібне)

- ☐ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.
- ☐ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора. Роботу направити на доопрацювання.
- ☐ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Заявляю, що ознайомлений(-на) з повним звітом подібності, який був згенерований Системою щодо роботи (додається)

Автор _____ Шадюк В.С.
(підпис) (прізвище, ініціали)

Опис прийнятого рішення

Ступінь оригінальності роботи відповідає вимогам, що висуваються до МКР
Особа, відповідальна за перевірку_____ Коваль Н.П.
(підпис) (прізвище, ініціали)

Експерт _____ _____
(за потреби) (підпис) (прізвище, ініціали)