

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук та технологій
(повне найменування інституту, факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка
до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ
АУТЕНТИФІКАЦІЇ НА ОСНОВІ РОЗПІЗНАВАННЯ ОБЛИЧ.
RESEARCH AND SOFTWARE IMPLEMENTATION OF
AUTHENTICATION METHODS BASED ON FACIAL RECOGNITION

Виконала: студентка ІІ курсу, групи КНТ-219м
спеціальності

122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Системи штучного інтелекту

Олефіренко А. О.
(прізвище та ініціали)

Керівник Колпакова Т.О.
(прізвище та ініціали)

Рецензент Казурова А.Є.
(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет Інститут інформатики та радіoeлектроніки,
факультет комп'ютерних наук і технологій
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 122 Комп'ютерні науки
(код і назва)
Освітня програма (спеціалізація) Системи штучного інтелекту
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ
Завідувач кафедри проф., д.т.н.
С. О. Субботін
“ ” _____ 2020 року

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТКИ

- Олефіренко Аліни Олегівни
(прізвище, ім'я, по батькові)
1. Тема проєкту (роботи) Дослідження та програмна реалізація методів аутентифікації на основі розпізнавання облич. Research and Software Implementation of Authentication Methods Based on Facial Recognition
- керівник проєкту (роботи) Колпакова Т.О., к.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
- затверджені наказом закладу вищої освіти від “3” листопада 2020 року №301
2. Строк подання студентом проєкту (роботи) 5 грудня 2020 року
3. Вихідні дані до проєкту (роботи) технічне завдання
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1 Аналіз проблеми та постановка завдань дослідження, 2 Матеріали та методи, 3 Розробка архітектури програми, 4 Основні рішення щодо реалізації компонентів системи, 5 Експлуатація, тестування та експериментальне дослідження програми, 6 Організаційно-економічна частина, 7 Охорона праці та безпека у надзвичайних ситуаціях, Додатки.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентацій

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
Основна частина	Колпакова Т.О., доцент		
Організаційно-економічна частина	Пожуєва Т.О., професор		
Охорона праці та цивільна безпека	Коробко О.В., ст. викладач		
Нормоконтролер	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання 2 жовтня 2020 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз предметної області	1 тиждень	
2	Розробка та затвердження технічного завдання	2 тиждень	Технічне завдання
3	Проектування структури системи	3-4 тиждень	Структурна схема
4	Розробка схеми функціонування системи та модулів програми	5-6 тиждень	Функціональна схема. Модулі
5	Створення програмного коду системи	7-8 тиждень	Текст програми
6	Тестування та відлагодження програми	9 тиждень	Працездатна система
7	Розробка розділів пояснювальної записки	10 тиждень	ПЗ
8	Розробка слайдів презентації	11-12 тиждень	Слайди презентації
9	Захист роботи	13 тиждень	

Студент(ка) _____ Олефіренко А.О.
(підпис) (прізвище, ініціали)

Керівник проєкту (роботи) _____ Колпакова Т.О.
(підпис) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи магістра: 122 с., 29 рис., 12 табл., 4 додатка, 42 джерел.

АУТЕНТИФІКАЦІЯ, ПРАВА КОРИСТУВАЧА, ДОСТУП ДО ДАНИХ, РОЗПІЗНАВАННЯ, КОРЕЛЯЦІЙНЕ ЗІСТАВЛЕННЯ, ЗАХИСТ ДАНИХ, ПРОГРАМНИЙ КОМПЛЕКС

Об'єкт дослідження – процес аутентифікації на основі біометричних даних.

Предмет дослідження – аутентифікація на основі біометричних даних користувача.

Мета магістерської роботи – дослідження та програмна реалізація методів аутентифікації за біометричними показниками користувача, а саме на основі розпізнавання облич.

Матеріали, методи та технічні засоби: для успішної реалізації поставленої мети у роботі використовуються поєднання сучасних методів та підходів, а саме методу кореляційного зіставлення зображень. Для роботи буде використовуватися мова програмування C++ та бібліотеку OpenCV. Ноутбук Lenovo G505 в наступній конфігурації: AMD A4-5000 (1.50 ГГц), RAM 8 ГБ, HDD 512 ГБ, nVidia AMD Radeon HD 8330, AMD Radeon HD 8570M, 6 ГБ, 15'' монітором, клавіатурою, маніпулятором типу «миша», що функціонує під управлінням операційної системи Microsoft Windows 10.

Результати: розроблено програмний комплекс, що реалізує запропонований підхід до аутентифікації за біометричними даними користувача, а саме використовує метод кореляційного зіставлення зображень для аутентифікації за обличчям користувача.

Висновки: було запропоновано новий метод аутентифікації на базі біометричних даних користувача, а саме зображення обличчя. Було розроблено програмний комплекс, що реалізує роботу запропонованого методу. Для цього було проаналізовано предметну область, розроблено технічне завдання для

реалізації програми, обрано середовище розробки, розроблено структурну схему програми, виконано конструювання програмного забезпечення для пошуку оптимального шляху, здійснено тестування розробленого програмного забезпечення.

Галузь використання: web- та мобільні-застосунки, що використовують особисту, банківську та державну інформацію користувачів.

ABSTRACT

Explanatory note of the final qualifying work of the master: 122 pages, 29 figures, 12 tables, 4 appendices, 42 sources.

AUTHENTICATION, USER RIGHTS, DATA ACCESS, RECOGNITION,
CORRELATION COMPARISON, DATA PROTECTION, SOFTWARE
PACKAGE

Object of study is the process of authentication based on biometric data.

Subject of study is authentication based on the user's biometric data.

The purposes of the work are research and software implementation of authentication methods based on user biometric indicators, namely on the basis of facial recognition.

Materials, methods and tools: for the successful implementation of this goal, the work uses a combination of modern methods and approaches, namely the method of correlation comparison of images. The C++ programming language and the OpenCV library will be used for this purpose. Lenovo G505 laptop in the following configuration: AMD A4-5000 (1.50 ГГц), RAM 8 ГБ, HDD 512 ГБ, nVidia AMD Radeon HD 8330, AMD Radeon HD 8570M, 6 GB, 15'' monitor, keyboard, mouse manipulator, running the Microsoft Windows 10 operating system.

Results. A software package has been developed that implements the proposed approach to authentication based on user biometric data, namely, it uses the image correlation method for authentication based on the user's face.

Conclusions. A new authentication method based on the user's biometric data, namely a face image, was proposed. A software package was also developed that implements the proposed method. To do this, the subject area was analyzed, a technical task for implementing the program was developed, a development environment was selected, a block diagram of the program was developed, software was designed to find the optimal path, and the developed software was tested.

The scope of use is web and mobile applications that use users' personal, banking, and government information.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів	10
Вступ.....	11
1 Аналіз проблеми та постановка завдань дослідження	13
1.1 Аутентифікація.....	13
1.2 Парольна аутентифікація.....	15
1.2.1 Метод парольної аутентифікації на основі відкритого пароля	15
1.2.2 Метод парольної аутентифікації заснований на хешуванні пароля	16
1.2.3 Аутентифікація на основі PIN-коду	17
1.2.4 Недоліки методів аутентифікації із запам'ятовуванням паролем	18
1.3 Аутентифікація за допомогою біометричних характеристик.....	19
1.3.1 Недоліки аутентифікації за допомогою біометричних характеристик	21
1.4 Метод аутентифікації за допомогою одноразових паролів	22
1.4.1 Недоліки методів аутентифікації за допомогою ОТР	23
1.5 Сучасні ускладнені методи аутентифікації	24
1.5.1 Багатофакторна аутентифікація.....	25
1.5.2 Однофакторная двоетапна аутентифікація	25
1.6 Постановка завдання.....	26
1.7 Висновки за розділом 1.....	26
2 Матеріали та методи	27
2.1 Біометричне розпізнавання в системах аутентифікації	27
2.2 Критерії біометричних систем	28
2.3 Переваги біометричної системи аутентифікації	29
2.4 Кореляційне зіставлення зображень	31
2.5 Висновки за розділом 2.....	34
3 Розробка архітектури програми	35

3.1	Опис основних вимог до програми	35
3.2	Проектування структури програмної системи	36
3.3	Вибір мови програмування.....	36
3.3.1	Мова програмування для взаємодії із web-сайтом та web-камерою....	36
3.3.2	Мова програмування для модуля розпізнавання	38
3.3.3	Бібліотека OpenCV	39
3.4	Вибір середовища розробки	41
3.5	Вимоги до технічних засобів.....	42
3.6	Висновки за розділом 3.....	43
4	Основні рішення щодо реалізації компонентів системи.....	44
4.1	Підключення OpenCV	44
4.2	Використовуваний метод розпізнавання і його реалізація	46
4.3	Реалізація клієнтської частини	47
4.4	Сервер	49
4.5	Висновки за розділом 4.....	50
5	Експлуатація, тестування та експериментальне дослідження програми	51
5.1	Опис застосування програми.....	51
5.2	Вибір порогу розпізнавання	52
5.3	Висновки за розділом 5.....	54
6	Організаційно-економічна частина	55
6.1	Планування розробки програмного продукту	55
6.2	Визначення витрат на розробку програми.....	56
6.3	Розрахунок економічної ефективності програмного продукту	63
7	Охорона праці та безпека у надзвичайних ситуаціях.....	65
7.1	Аналіз потенційних небезпек	65
7.2	Заходи щодо забезпечення техніки безпеки	66
7.3	Заходи щодо забезпечення виробничої санітарії та гігієни праці	69
7.4	Заходи безпеки у надзвичайних ситуаціях	75

7.4.1 Заходи пожежної безпеки	75
7.4.2 Заходи з цивільного захисту	77
Висновки.....	81
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	82
ДОДАТОК А Технічне завдання.....	87
A.1 Вступ	88
A.1.2 Призначення та область застосування.....	88
A.2 Підстави для розробки	88
A.3 Призначення розробки	88
A.4 Вимоги до програми	89
A.4.1 Вимоги до функціональних характеристик.....	89
A.4.2 Вимоги до надійності	89
A.4.3 Вимоги до технічних та програмних засобів	89
A.5 Вимоги до програмної документації	90
A.6 Стадії і етапи розробки.....	90
A.6.1 Стадії розробки	90
A.6.2 Етапи розробки	90
A.6.3 Зміст робіт по етапах	91
ДОДАТОК Б Опис програми.....	92
Б.1 Загальні свідчення.....	93
Б.2 Функціональне призначення	93
Б.3 Опис логічної структури.....	93
Б.4 Використані технічні засоби	95
ДОДАТОК В Текст програми.....	96
ДОДАТОК Г Слайди презентації.....	115

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

FAR – False Acceptance Rate;

FRR – False Rejection Rate;

OTP – One Time Password;

PIN-код – Personal Identification Number;

ЕОМ – електронна обчислювальна машина;

ЕЦП – електронний цифровий підпис;

ПЕОМ – персональна електронна обчислювальна машина;

ПЗ – персональний забезпечення;

ПП – програмний продукт;

СКУД – система контролю та управління доступом.

ВСТУП

В даний час для аутентифікації на web-сайтах використовується стандартна процедура введення логіна і пароля. З метою безпеки паролі для різних сайтів повинні відрізнятися. Користувач відвідує безліч різних web-ресурсів, де необхідно вводити логін і пароль для авторизації. Отже, необхідно якось зберігати всі дані для входу на відповідні сайти. Досить часто просто запам'ятати – не виходить, через високу ймовірність забути дані для входу. Інший спосіб – записувати паролі, або у форматі окремих файлів (наприклад, текстових), або використовувати Менеджери паролів. Збереження паролів в текстовому файлі не зручне та не безпечне. Використання менеджера паролів більш безпечніше та зручніше, але уразливості є і у них.

Менеджери паролів зазвичай використовують обраний користувачем основний пароль, або секретну фразу (passphrase), щоб сформувати ключ, який використовується для шифрування збережених паролів. Цей основний пароль повинен бути досить складним, щоб встояти при атаках злоумисників (наприклад, повний перебір).

Якщо основний пароль буде зламаний, то будуть розкриті всі збережені в базі даних програми паролі. Це демонструє зворотний зв'язок між зручністю використання і безпекою: єдиний пароль може бути більш зручний, але якщо він буде зламаний, то поставить під загрозу всі збережені паролі. Основний пароль може також бути атакований і виявлений при використанні кейлоггера або акустичного криптоаналізу. Деякі менеджери паролів включають генератор паролів. Згенеровані паролі можуть бути достатньо слабкими, якщо менеджер пароля не використовує криптографічний безпечний генератор випадкових чисел [1].

В браузері Google Chrome для того, щоб подивитися всі автоматично збережені паролі – досить знати один пароль. Після введення даного паролю можна подивитися всі автоматично збережені паролі.

В рамках магістерській роботі пропонується поліпшити наявні методи, що використовуються в менеджерах паролів. Замінити аутентифікацію за паролем на аутентифікацію за збереженим обличчям. В якості ключового ідентифікатору використовується зображення обличчя людини. Для додаткової надійності у кожного користувача буде пароль, але вводити його потрібно буде тільки у випадки невдалої повторної аутентифікації або при попаданні в діапазон «не свій, але і не чужий». Такий модуль авторизації буде зручний при використанні вдома і в офісі.

1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

Перший розділ буде присвячено детальному аналізу проблемної області пов'язаної із захистом та перевіркою користувацьких даних при доступі до ресурсів та сервісів веб-застосунків. Під час роботи буде проведено дослідження основних термінів таких, як аутентифікація, авторизація та ідентифікація. Буде виділено той етап перевірки користувачів над яким буде вестися подальша робота. Додатково буде проведено аналіз сучасного становища етапу для актуалізації завдання.

1.1 Аутентифікація

За термінами аутентифікація, авторизація та ідентифікація ховаються дії, що виконуються щодня більшістю користувачів машинально. При вході в онлайн-банкінг або електронну пошту система безпеки сайту просить ввести облікові дані: логін і пароль (або відбиток пальця). В обмін на вірну пару логін-пароль користувач отримує доступ до своїх фінансів або електронною поштою [1]-[6].

Насправді ніякого обміну не відбувалося. Відбулися по чергово три процеси: ідентифікація, аутентифікація та авторизація.

Ідентифікація, аутентифікація та авторизація – три процеси, що захищають користувацькі дані або грошові кошти від доступу сторонніх осіб.

Розуміння процесів прийде швидше, якщо дати їм визначення.

Ідентифікація – процес розпізнавання користувача за його ідентифікатором.

Аутентифікація – процедура перевірки автентичності, доказ що користувач саме той, за кого себе видає.

Авторизація – надання певних прав.

Тож аналізуючи ці визначення прийдемо до висновків, що саме аутентифікація механізм встановлення справжності користувача, що гарантує подальшу безпечність його даних.

За допомогою аутентифікації встановлюється справжність. Іншими словами дозволяється працювати в мережі тільки санкціонованим користувачам. Аутентифікація може проводитися при вході в мережу і під час роботи. При встановленні автентичності користувач повідомляє свій ідентифікатор. У процесі аутентифікації беруть участь дві сторони: користувач доводить свою справжність, а система перевіряє цей доказ і приймає рішення.

Незалежно від типу системи аутентифікації в ній завжди присутні п'ять елементів.

Перший елемент – конкретна людина або процес, який повинен проходити аутентифікацію, – суб'єкт доступу.

Другий елемент – розпізнавальний знак, ідентифікатор, який виділяє цю людину або цей процес серед інших.

Третій елемент – відмінна характеристика (аутентифікатор), що підтверджує приналежність ідентифікатора суб'єкту доступу.

Четвертий елемент – власник системи (адміністратор), який несе відповідальність за використання системи, і в розмежуванні авторизованих користувачів та інших покладається на механізм аутентифікації.

П'ятий елемент – механізм аутентифікації, який дозволяє перевірити присутність відмінної характеристики.

При успішному проходженні аутентифікації суб'єкту доступу повинні бути видані деякі права. Для цього служить механізм управління доступом. За допомогою цього ж механізму суб'єкт доступу позбавляється прав, якщо аутентифікація була неуспішною.

Фактори аутентифікації – певний вид інформації, що надається суб'єктом системи при його аутентифікації. Виділяють три фактори аутентифікації, що використовуються в різних комбінаціях: на основі знання

чого-небудь, володіння чим-небудь, на основі біометричних характеристик. В табл . 1.1 наведено приклади факторів аутентифікації [2]-[7].

Таблиця 1.1 – Фактори аутентифікації

Фактор аутентифікації	Приклад факторів аутентифікації
На основі знання чого-небудь	Пароль або парольна фраза PIN-код
На основі володіння чим-небудь	Фізичний ключ Карта з магнітною смугою ОТР-токен, що генерує одноразовий пароль
На основі біометричних характеристик	Відбиток пальця Малюнок сітківки ока Голос Параметри обличчя

1.2 Парольна аутентифікація

Для перевірки автентичності користувачів в інформаційних системах найбільш широко використовується аутентифікація по секретній інформації, яка невідома непосвяченим людям. У комп'ютерних системах-це пароль, що вводиться за допомогою клавіатури. Чим довше і складніше пароль, тим він більш стійкий. Парольна аутентифікація є найбільш простим методом аутентифікації з точки зору реалізації [9]-[10].

1.2.1 Метод парольної аутентифікації на основі відкритого пароля

Даний метод парольної аутентифікації є найпростішим і застарілим. Метод аутентифікації на основі відкритого пароля наведено на рис. 1.1.



Рисунок 1.1 – Аутентифікація на основі відкритого пароля

1.2.2 Метод парольної аутентифікації заснований на хешуванні пароля

В даний час програмного забезпечення застосовуються паролі не в чистому вигляді, а їх хеш-значення, одержувані за допомогою обчислення криптографічної хеш-функції.

Алгоритм аутентифікації заснованому на хешуванні пароля:

- користувач вводить свої ім'я користувача (логін) та пароль;
- комп'ютер обчислює хеш-значення від введеного пароля. Ім'я користувача і хеш-значення передаються по мережі серверу аутентифікації;
- сервер аутентифікації порівнює результат обчислення хеш-значення від введеного користувачем пароля з хеш-значенням, що зберігається в обліковому записі користувача;
- у разі збігу аутентифікація визнається успішною.

Використовуючи односпрямовану хеш-функцію стає неможливо відновити вихідну інформацію при володінні отриманої з неї хеш-значень. На рис. 1.2. зображена аутентифікація з використанням хеш-функцій.

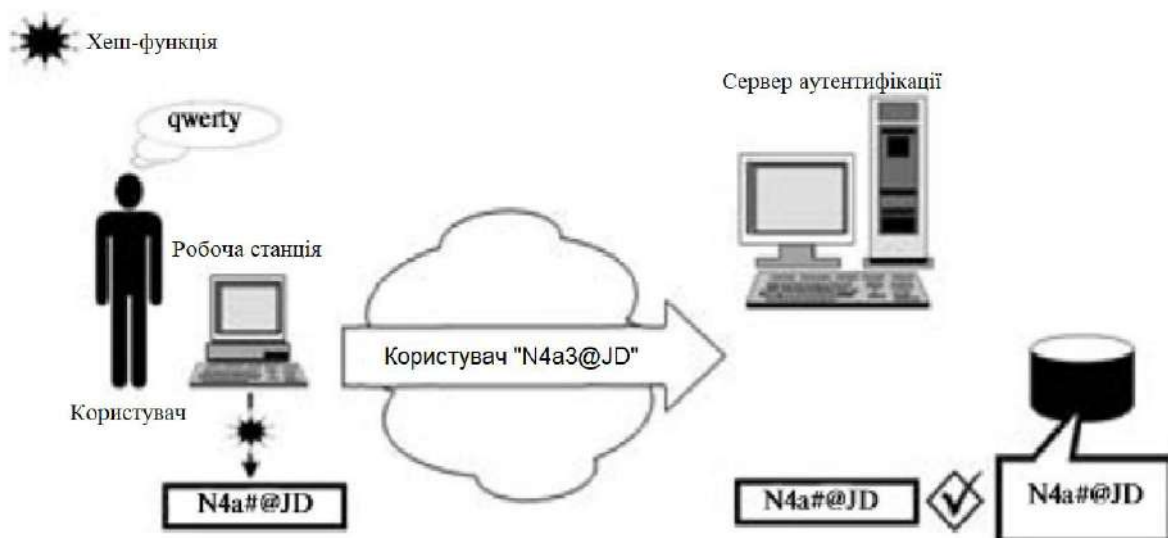


Рисунок 1.2 – Аутентифікація на основі хешування пароля

1.2.3 Аутентифікація на основі PIN-коду

Personal Identification Numbe (PIN-код) – аналог до паролю, але цей код складається тільки з цифр. Часто використовується там, де використання повноцінної клавіатури технічно ускладнене або недоречне [12]-[15]. Те, що PIN-код не має інших символів крім цифр, робить його простішим. Щоб ускладнити або й унеможливити процес його підбирання, обмежують кількість спроб вводу або збільшують інтервал очікування між хибними спробами.

PIN-код – це різновид пароля, зазвичай використовується для аутентифікації на локальному пристрої. Даний вид аутентифікації часто використовується з іншими видами пристроїв аутентифікації, наприклад, смарт-картами. Зазвичай PIN-код торгового автомата або банкомату складається з чотирьох цифр. Таким чином, один з кожних 10000 клієнтів мають один і той же PIN-код. Для PIN-коду характерно наступне:

- у локальному пристрої, в якому здійснюється аутентифікація за допомогою PIN-коду, є інтерфейс для користувача, а не для програм. Ніхто не може ввести PIN-код, не використовуючи клавіатуру даного пристрою [8];
- PIN-код не передається по мережі і не може бути перехоплений [2].

1.2.4 Недоліки методів аутентифікації із запам'ятовуванням паролем

Методи аутентифікації із запам'ятовуванням паролем володіють багатьма недоліками. Нижче будуть перераховані атаки на системи, в яких використовується аутентифікація на основі пароля:

- крадіжка парольного файлу. Зловмисник може прочитати паролі користувача з парольного файлу або резервної копії;
- підбір пароля. Виходячи з особистих даних користувача, зловмисник намагається увійти в систему;
- примус. Для того щоб змусити користувача відкрити свій пароль, зловмисник використовує загрози або фізичний примус;
- соціотехніка. Зловмисник представляється адміністратором і змушує користувача або відкрити свій пароль, або змінити його на вказаний ним пароль;
- підглядання з-за спини. Розташований поруч зловмисник або відеокамера стежить за тим, як користувач вводить пароль;
- троянський кінь. Зловмисник приховано встановлює програмне забезпечення, імітує звичайний механізм аутентифікації, але збирає імена користувачів і паролі при спробах користувачів увійти в систему;
- аналіз мережевого трафіку. Зловмисник аналізує мережевий трафік, що передається від клієнта до сервера, для відновлення з нього імен користувачів і паролів;
- трасування пам'яті. Зловмисник використовує програму для копіювання пароля користувача з буфера клавіатури;
- апаратний сніффер клавіатури. Зловмисник приховано встановлює в комп'ютер користувача апаратний засіб, що збирає інформацію, яку вводить користувач при вході в систему, наприклад, KeyCarbon, KeyDevil для дротових клавіатур.

1.3 Аутентифікація за допомогою біометричних характеристик

Ключовим ідентифікатором в такому вигляді аутентифікацій є біометричні характеристики. До біометричних характеристик можна віднести [2]-[16]:

- райдужну оболонку ока;
- відбиток пальця;
- особа;
- кисть;
- сітківка.

Фізіологічні біометричні характеристики зазвичай незмінні протягом життя людини. Оскільки біометричний параметр унікальний для даної людини, його можна використовувати для однофакторної аутентифікації користувача. Біометрична аутентифікація зазвичай є одним з найбільш простих методів для користувача, які повинні проходити аутентифікацію. Добре спроектована система просто знімає показання з людини і правильно виконує аутентифікацію.

При біометричній аутентифікації Користувач надає зразок-розпізнаване, необроблене зображення або запис фізіологічної, або поведінкової характеристики. За допомогою реєструючого пристрою, наприклад, сканера або камери, цей біометричний зразок обробляється для отримання інформації про відмітні знаки, в результаті чого виходить контрольний шаблон. Шаблон являє собою досить великі числові послідовності, сам зразок неможливо відновити з шаблону. Контрольний шаблон і є «пароль» користувача. На рис. 1.3. представлена система для біометричної аутентифікації користувача.

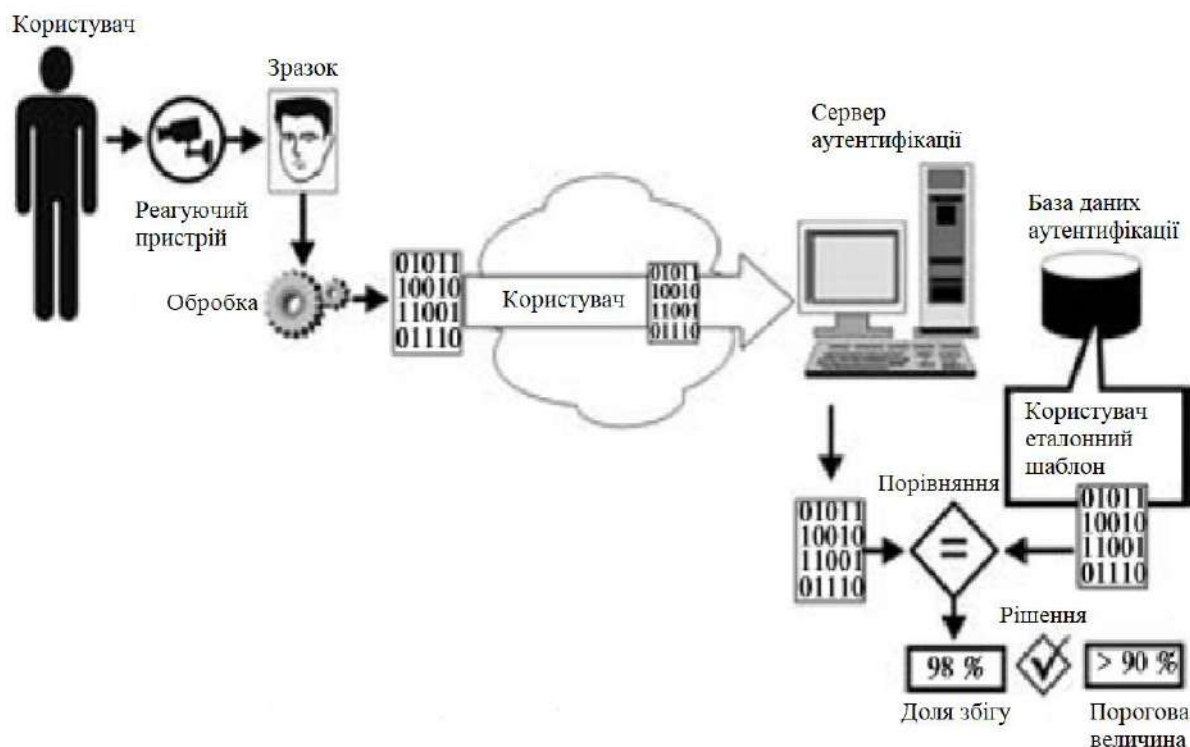


Рисунок 1.3 – Схема роботи біометричної системи

Контрольний шаблон порівнюється з еталонним шаблоном, створеним на основі декількох зразків певної фізіологічної або поведінкової характеристики користувача, взятих при його реєстрації в біометричній системі. Оскільки ці два параметри (контрольний і еталонний шаблон) повністю ніколи не збігаються, то біометричній системі доводиться приймати рішення про те, «досить» вони збігаються. Ступінь збігу повинна перевищувати певну порогову величину.

Контрольний шаблон може бути помилково визнаний:

- відповідним еталонним шаблоном іншої особи;
- коефіцієнтом невірних збігів, що відносяться до помилок 1 роду або ймовірність помилкового допуску;
- коефіцієнтом невірних розбіжностей, що відносяться до помилок 2 роду або ймовірність помилкового відмови в доступі.

Обидва коефіцієнти відображають здатність системи надавати обмежений вхід авторизованим користувачам. Простота реєстрації і якість «шаблонів» – важливий фактор загальної ефективності біометричної системи.

Неякісний «шаблон» може ускладнити роботу користувача, змушуючи його вдатися до повторної реєстрації в біометричній системі [2],[8],[13].

1.3.1 Недоліки аутентифікації за допомогою біометричних характеристик

До недоліків аутентифікації за допомогою біометричних характеристик можна наступні: втручання в приватне життя; особисті, культурні та релігійні аспекти. У табл. 1.2. надано опис недоліків біометричного методу аутентифікації.

Таблиця 1.2 – Недоліки біометричного методу аутентифікації

Опис атаки	Захист від даної атаки
Підробка відмінної риси. Зловмисник виготовляє копію фізичної відмінної риси законного користувача і пред'являє цю копію біометричному датчику.	Зняття показників з високим рівнем деталізації. При виготовленні еталонного шаблону з законного користувача знімають Додаткові біометричні показники.
Відтворення поведінки користувача. Зловмисник записує поведінкову відмінну рису користувача і відтворює на біометричному датчику.	Змінювана поведінка. При кожній спробі аутентифікації система вимагає від користувача різного прояву його поведінкової біометричної характеристики.
Перехоплення біометричних показників. Зловмисник перехоплює біометричні показники законного користувача в момент їх передачі між пристроями.	Шифрування біометричних даних. Біометричні дані шифруються відразу після їх отримання від користувача пристроєм зчитування, їх передача між пристроями здійснюється тільки в шифрованому вигляді.
Відтворення біометричного «підпису». Зловмисник відтворює показник біометричного датчику – «підпис», яка далі обробляється системою так, немов була отримана від реальної людини.	Аутентифікація біометричного «підпису». Заходи аутентифікації вживаються щодо біометричних даних, чим гарантується їх надходження тільки з надійних джерел. Використання електронного цифрового підпису (ЕЦП) для забезпечення цілісності біометричного «підпису».

1.4 Метод аутентифікації за допомогою одноразових паролів

В основі методу аутентифікації за допомогою одноразового пароля лежить гіпотеза про те, що тільки законний користувач може пройти перевірку, так як тільки він знає свій пароль. Одним з варіантів захисту від різних атак на аутентифікацію по паролю є аутентифікація з використанням одноразових паролів.

Одноразові паролі-динамічна аутентифікаційна інформація, що генерується для одиничного використання за допомогою аутентифікаційних пристроїв. Даний вид паролів невразливий для атаки методом аналізу мережевого трафіку, що є значною перевагою перед запам'ятовуються паролями. Незважаючи на те, що зломисник може перехопити пароль методом аналізу мережевого трафіку, оскільки пароль дійсний лише один раз і протягом обмеженого проміжку часу, у зломисника в кращому випадку є досить обмежена можливість представитися Користувачем за допомогою перехопленої інформації [3].

Для генерації одноразових паролів зазвичай використовуються ОТР-токени.

ОТР-токени – мобільний персональний пристрій, який належить певному користувачеві і генерує одноразові паролі, використовувані для аутентифікації даного користувача.

Перевагою застосування аутентифікаційних пристроїв є те, що багато з них вимагають від користувача введення PIN-коду:

- для активації ОТР-токена;
- в якості додаткової інформації при генерації ОТР;
- для пред'явлення серверу аутентифікації разом з ОТР.

Даний метод відноситься до двофакторної аутентифікації, тому що використовуються два фактори аутентифікації. Частково метод аутентифікації за одноразовим паролем зустрічається в системах «Інтернет-банк».

Для генерації одноразових паролів ОТР-токени використовують хеш-функції або криптографічні алгоритми:

- симетрична криптографія (криптографія з одним ключем) - в цьому випадку користувач і сервер аутентифікації використовують один і той же секретний ключ;
- асиметрична криптографія (криптографія з відкритим ключем) - в цьому випадку в пристрої зберігається закритий ключ, а сервер аутентифікації використовує відповідний відкритий ключ.

Методи, використовувані ОТР-токенами, можна розділити на дві групи, що працюють: в асинхронному режимі («запит-відповідь»); в синхронному режимі («тільки відповідь»). Метод «запит-відповідь» показаний на рис. 1.4.



Рисунок 1.4 – Метод «запит-відповідь»

1.4.1 Недоліки методів аутентифікації за допомогою ОТР

Нижче наведені відомі атаки на системи, що використовують аутентифікацію за допомогою ОТР-токена.

- атака «людина посередині». Зловмисник перехоплює одноразовий пароль, посланий законним користувачем при аутентифікації, блокує законного користувача і використовує перехоплений пароль для входу в систему;
- крадіжка аутентифікаційного токена. Зловмисник викрадає аутентифікаційний токен законного користувача і використовує його для входу в систему;
- підбір PIN-коду аутентифікаційного токена. Зловмисник вручну здійснює перебір всіх можливих значень PIN-коду, викраденого ним аутентифікаційного токена законного користувач [5];
- витяг значення секретного ключа з програмного аутентифікаційного токена. Зловмисник копіює токен, намагаючись знайти в ньому зберігається секретний ключ [2].

1.5 Сучасні ускладнені методи аутентифікації

Аутентифікація, що здійснюється з використанням двох чи більше факторів називається багатофакторною.

У відповідності до вимог Європейської директиви PSD2 в платіжних системах використовується так звана посилена аутентифікація, коли для аутентифікації використовуються принаймні два різних типи факторів. Типами факторів є:

- властивість, якою володіє суб'єкт;
- знання — інформація, яку знає суб'єкт;
- володіння — річ, якою володіє суб'єкт.

Суворі аутентифікація – аутентифікація, під час якої використовується інформація без розкриття цієї інформації. Як правило, реалізується за допомогою асиметричних криптографічних алгоритмів.

1.5.1 Багатофакторна аутентифікація

Багатофакторна аутентифікація являє собою метод, при якому користувачеві для доступу до облікового запису або підтвердження операції з грошовими коштами необхідно двома різними факторами довести, що саме він власник облікового запису або що саме він здійснює вхід.

Серед видів багатофакторної аутентифікації найбільш поширена двофакторна аутентифікація (2FA – 2-factor authentication) – метод, при якому користувачеві для отримання доступу необхідно надати два різних типи аутентифікаційних даних, наприклад, щось відоме тільки користувачеві (пароль) і щось притаманне тільки користувачеві (відбиток пальця).

Доступ до ресурсів через введення логіна і пароля, є однофакторною аутентифікацією, оскільки для входу використовується тільки один тип аутентифікаційних даних-відомий користувачеві пароль.

1.5.2 Однофакторная двоетапна аутентифікація

Завдяки тому, що смартфони стали невід'ємною частиною нашого життя, саме вони стали одним із способів підтвердження особи користувача. Вони є токенами для доступу до різних ресурсів. В цьому випадку одноразовий пароль генерується або за допомогою спеціального додатку, або приходить по SMS – це максимально простий для користувача метод.

Аутентифікація відбувається наступним чином:

Користувач вводить логін і пароль, зазначені при реєстрації. Якщо дана пара коректна (логін є в базі і відповідає паролю) система висилає одноразовий пароль, який має обмежений час дії.

Користувач вводить одноразовий пароль і, якщо він збігається з тим, що відправила система, то користувач отримує доступ до свого облікового запису, грошових коштів або підтверджує грошовий переказ.

Навіть якщо зловмисник отримає логін і пароль для облікового запису (за допомогою шкідливої програми, крадіжки записної книжки з паролями або методами соціальної інженерії та фішингу), то після введення цих даних система відправить на прив'язаний мобільний телефон користувача одноразовий код з обмеженим часом дії. Без одноразового коду шахрай не зможе викрасти грошові кошти.

1.6 Постановка завдання

Головна мета магістерської роботи полягає у дослідженні, розробці та програмній реалізації методу аутентифікації на основі біометричних даних користувача, а саме на базі розпізнавання за біометричним ідентифікатором – двомірним зображення обличчя. Для подальшої роботи буде використовуватися метод швидкої нормалізованої взаємної кореляції та бібліотека функцій алгоритмів комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення – OpenCV.

1.7 Висновки за розділом 1

Розглянуто проблемну область використання сучасних методів аутентифікації для захисту користувацьких даних. В результаті проведеного аналізу було з'ясовано, що одним з найбільш актуальних та безпечних методів аутентифікації є аутентифікація за біометричними даними. Використання методу швидкої нормалізованої взаємної кореляції забезпечить високу точність порівняння еталонних та отриманих зображень.

2 МАТЕРІАЛИ ТА МЕТОДИ

В першому розділі було проаналізовано сучасне становище у питанні ідентифікації, аутентифікації та авторизації. Попередній аналіз довів, що використання біометричних даних користувача є одним з найбільш перспективних методів аутентифікації.

Тому актуальним є дослідження та реалізація моделей аутентифікації заснованих біометричних даних про користувача. Проте для цього слід визначити, які саме дані слід використовувати та як вони можуть бути отримані.

Другий розділ присвячено детальному ознайомленню та дослідженню процесу аутентифікації за біометричними даними.

2.1 Біометричне розпізнавання в системах аутентифікації

При аутентифікації біометрична система перевіряє дані, що надходять від суб'єкта аутентифікації звіряючи їх з контрольним шаблоном. Деякі системи аутентифікації мають дуже обмежений пошук серед великого числа зареєстрованих записів. Наприклад, користувач з трьома еталонними шаблонами відбитків пальців може мати можливість надати для перевірки будь-який з трьох пальців, і система зробить пошук збігів 1:1 серед еталонних шаблонів даного користувача [2]-[8].

Біометричне розпізнавання – це процес визначення особистості користувача за біометричними характеристиками. У режимі розпізнавання система визначає особистість користувача, здійснюючи порівняння контрольного шаблону з багатьма біометричними еталонними шаблонами 1:N (або порівняння один до багатьох). У разі знаходження збігу одночасно визначається і засвідчується особа користувача.

Біометрична аутентифікація широко поширена в таких областях, як судова медицина і діяльність правоохоронних органів. Системи біометричної

аутентифікації часто вбудовуються в ноутбуки і смартфони. Біометрія в 95% випадків використовує статистичні методи. Математична статистика відноситься до числа точних наук. Для біометричних систем характерні помилки першого та другого роду. З двох біометричних систем краще та, у якій помилка другого роду менше при однаковому значенні помилок першого роду. Важливими характеристиками біометричних систем є їх швидкість роботи і вартість. Біометричних методів безліч. До розряду часто використовуваних методів можна віднести методи ідентифікації за відбитком пальця, сітківки ока, геометрії руки, райдужної оболонки, геометрії особи [4].

2.2 Критерії біометричних систем

Біометричні системи в якості ключового ідентифікатора можуть використовувати наступні біометричні характеристики: райдужну оболонку ока, відбиток пальця, сітківку, кисть, обличчя. У таблиці 2.1 необхідні критерії для реалізації біометричних систем аутентифікації.

На рис. 2.1. представлена діаграма розподілу біометричних систем на ринку. Дані актуальні на 2019 [5]-[10].

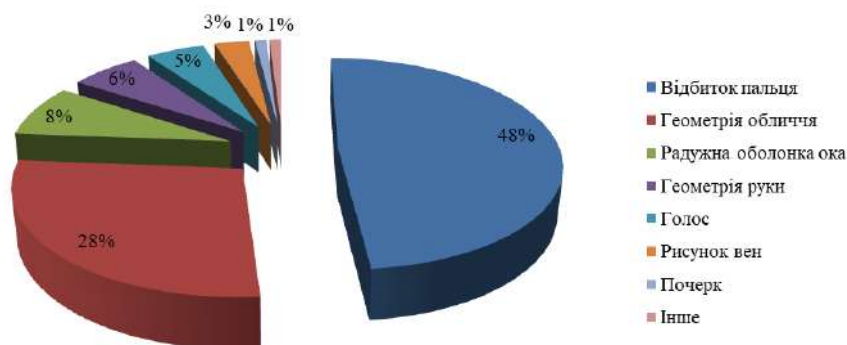


Рисунок 2.1 – Діаграма розподілу біометричних систем на ринку

З кожним роком стає все більше систем, заснованих на біометричному методі ідентифікації користувача по зображенню обличчя. На пересічному ринку це пояснюється масовим введенням технологій на кшталт Face ID.

Таблиця 2.1 – Критерії для реалізації біометричної системи аутентифікації

Біометрична характеристика	Реєструючий пристрій	Зразок	Досліджувані риси
Лице	Відеокамера або web-камера, стереокамера	Зображення обличчя (двовимірне або тривимірне)	Форма черепа, відносне розташування і форма носа, розташування брів, розташування вилиць
Відбиток пальця	Периферійний пристрій настільного комп'ютера, зчитувач, вбудований в клавіатуру	Зображення відбитку пальця	Розташування і напрямок гребінців виступів і розгалужень на відбитку пальців
Кисть руки	Настінний пристрій-зчитувач	Тривимірне зображення верху і боків кисті	Висота і ширина кісток і суглобів кисті і пальців
Райдужна оболонка	Інфрачервона відеокамера	Чорно-біле зображення райдужної оболонки ока	Смужки і борозенки в райдужній оболонці ока
Сітківка	Настінний або настільний зчитувальний пристрій	Зображення сітківки	Розташування кровоносних судин на сітківці

2.3 Переваги біометричної системи аутентифікації

Для реалізації модуля авторизації на web-сайті була обрана система біометричної аутентифікації. Біометричні системи аутентифікації мають ряд переваг перед іншими система контролю та управління доступом (СКУД). Біометрична аутентифікація проводиться за унікальним ідентифікатором, а не за асоційованим з людиною матеріальним носієм або паролем. Аутентифікація

з використанням біометричних характеристик практично виключає несанкціонований доступ.

З існуючих методів біометричної ідентифікації був обраний метод аутентифікації з геометрії особи. Існує багато методів заснованих на розпізнаванні обличчя. Дані методи спираються на унікальні фізіологічні особливості форми черепа. Існує кілька різновидів даного методу: 2D розпізнавання, 3D розпізнавання.

Вибрано було 2D розпізнавання. Для цілей, переслідуваних в роботі даний вид розпізнавання підходить найкраще. Переваги 2D розпізнавання:

- метод не вимагає дорогого ідентифікуючого обладнання;
- для реалізації алгоритму підійде web-камера, яка зазвичай знаходиться в кожному ноутбукі;
- при наявності необхідної кількості еталонних зображень дає незначний відсоток помилок 1 і 2 роду;
- швидкість ідентифікації швидше, ніж у методів, що використовують 3D зображення;
- володіє меншим розміром збереженої шаблонної інформації. 3D модель несе в собі більше інформації і має в рази більший розмір.

Не один метод не обходиться без недоліків.

Основні недоліки даного методу і шляхи їх нівелювання в роботі будуть представлені нижче в таблиці 2.2.

Таблиця 2.2. – Недоліки 2D методу аутентифікації і способи їх нівелювання

Недоліки	Спосіб усунення під час роботи
1	2
Низька статистична достовірність	Даний модуль аутентифікації буде встановлюватися на локальний ПК і призначений для роботи обмеженого кола осіб. стабільність роботи гарантована при числі користувачів, що не перевищують 30.

Продовження таблиці 2.2

1	2
Пред'являються вимоги до освітлення	Система буде використовуватися в закритому приміщенні. Освітленість протягом дня не буде сильно змінюватися.
Зовнішні перешкоди (наприклад, сонцезахисні окуляри)	У приміщення немає потреби ходити в сонцезахисних окулярах. Даний атрибут повинен бути відсутнім на обличчі людини.
Відхилення голови неприпустимі	В якості шаблонних зображень будуть задаватися кілька положень голови щодо камери, що дозволить знизити помилку 2 роду

2.4 Кореляційне зіставлення зображень

Традиційна техніка порівняння поточного зображення з еталоном ґрунтується на розгляді зображень як двовимірних функцій яскравості (дискретних двовимірних матриць інтенсивності). При цьому вимірюється або відстань між зображеннями, або міра їх близькості [17]-[19].

Як правило, для обчислення відстаней між зображеннями використовується співвідношення:

$$\rho(f, g) = \left[\sum_{\langle x, y \rangle \in X} |f(x, y) - g(x, y)|^\alpha \right]^{1/\alpha}, \quad (2.1)$$

де $f(x, y)$ та $g(x, y)$ – функції інтенсивності;

X – поле зору;

величина $\alpha \in [1, \infty)$ визначає характеристики використовуваної метрики в (2.1).

Очевидно, що при $\alpha = 2$ цей вираз описує звичайну евклідову відстань між зображеннями, що розглядаються як вектори, що належать простору $L^2(x, y)$ на полі зору X функцій інтенсивностей з інтегрованим квадратом [18].

Нехай дано n еталонних зображень $\{f_i\}, i=1,2,\dots,n$, кожне з яких відповідає i -му класу. Віднесення знову пред'являється фрагмента зображення g до деякого класу j може здійснюватися, наприклад, за методом мінімальної відстані до відповідного еталону:

$$j = \arg \min_i \rho(g, f_i). \quad (2.2)$$

Цей найпростіший метод має два основних недоліки:

- критерій виявлення залежить від лінійних розмірів еталона і зображення;
- критерій виявлення не інваріантний навіть до найпростіших фотографічних перетворень яскравості виду $f' = af + b$.

Більш прийнятним тому є використання кореляційної метрики, а саме, нормованого коефіцієнта кореляції,

$$K(f, g) = \frac{\sum_{\langle x, y \rangle \in X} (f(x, y) - f^0)(g(x, y) - g^0)}{\sqrt{\sum_{\langle x, y \rangle \in X} (f(x, y) - f^0)^2} \sqrt{\sum_{\langle x, y \rangle \in X} (g(x, y) - g^0)^2}}, \quad (2.3)$$

де f^0, g^0 – середні значення інтенсивності для зображень f та g відповідно.

Нормований коефіцієнт кореляції володіє наступними добре відомими властивостями:

- $-1 \leq K(f, g) \leq 1, \forall f, g$;
- $(K(f, g) = 1) \Leftrightarrow (g = af + b, a > 0, \forall b)$;
- $(K(f, g) = -1) \Leftrightarrow (g = af + b, a < 0, \forall b)$.

Останню властивість, як правило, називають «зворотним контрастом».

Нехай, як і раніше, дано n еталонних зображень $\{f_i\}, i=1,2,\dots,n$, кожне з яких відповідає i -му класу. Виявлення фрагмента зображення g за методом максимального кореляційного зв'язку здійснюється тоді за правилом (2.2) отримаємо:

$$j = \arg \min_i K(g, f_i). \quad (2.3)$$

Після цього на підставі отриманого значення максимальної кореляції може перевірятися достовірність детектування. Якщо $K(g, f_i) \geq K_{\min}$, то виявлення визнається достовірним. В іншому випадку об'єкт вважається нерозпізнаним. Теоретичним обґрунтуванням застосування кореляційного методу виявлення є його суворе оптимальність для виявлення детермінованого сигналу в білому шумі з гаусовим розподілом яскравостей [18],[19].

Важливі недоліки кореляційних методів виявлення проявляються в присутності радіометричних (яскравісних) і особливо геометричних спотворень поточного зображення в порівнянні з еталонним. Зокрема, спостерігається швидке зменшення кореляційного зв'язку при так званих ракурсних спотвореннях, наприклад, при поворотах зображень. Присутність спотворень типу «warping» зазвичай вже не дозволяє використовувати кореляційні методи виявлення. У той же час, для цього класу алгоритмів запропоновано велику кількість процедур, що дозволяють або підвищити їх працездатність, або значно прискорити процес пошуку. На цьому шляху були розроблені ієрархічні кореляційні алгоритми, значно зберегли актуальність і на поточний момент. Конструктивна ідея зменшення часу пошуку лежить в основі методу амплітудного ранжирування. Згідно з цим методом слід аналізувати швидкість зростання кореляції в міру обробки поля зору і, якщо ця швидкість недостатня, припиняти обробку поточного фрагмента, переходячи до наступного. Для того щоб домогтися інваріантності кореляційних алгоритмів хоча б до афінної групи

перетворень, були випробувані різні перетворення зображень, наприклад перетворення Мелліна. На жаль, в силу недостатньої стійкості кореляційних алгоритмів до можливих спотворень, вони не знаходять широкого застосування при конструюванні алгоритмів виявлення складно структурованих об'єктів.

Окремий напрямок, що використовує кореляційні методи виявлення, це створення когерентних пристроїв обробки-оптичних кореляторів. Дослідження в цьому напрямку активно тривають, так як навіть незважаючи на значні конструктивні труднощі боротьби з мінливістю еталонів, реалізована тут фантастична швидкість обробки (швидкість світла) привертає до себе пильну увагу. Проблематика побудови оптичних кореляційних пристроїв не є предметом розгляду даної глави. Проте, зазначимо, що практика використання оптичних кореляторів виявила несподівано високий рівень помилок аномального виявлення об'єктів.

2.5 Висновки за розділом 2

У другому розділі проаналізовано процес аутентифікації за користувацькими біометричними даними.

У якості біометричних даних будемо використовувати зображення обличчя. Запропоновано використання кореляційного зіставлення зображень для порівняння еталонних даних про користувача із даними за якими відбувається аутентифікація.

3 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

3.1 Опис основних вимог до програми

Об'єктом розробки є модуль аутентифікації для web-сайтів, заснований на розпізнаванні обличчя. Мета роботи – створити систему аутентифікації, що використовує біометричний метод розпізнавання, який в якості ідентифікатор використовує 2D зображення обличчя.

Необхідно реалізувати 3-основні модуля для системи авторизації. Перший модуль представляє клієнтську частину, де від користувача приймаються вхідні дані для ідентифікації. Другий модуль представляє сервер для обробки даних від клієнта і модуль розпізнавання. У модулі розпізнавання здійснюється зіставлення зображення, прийнятого від сервера з еталонним зображенням, що зберігаються в базі.

Тож серед функціональних вимог можна виділити:

- фіксування поточного зображення;
- завантаження зображення до сервера зіставлення даних;
- зберігання еталонних зображень;
- порівняння зображень для аутентифікації;
- виведення результатів;
- надання та забезпечення графічного інтерфейсу користувача.

Для забезпечення надійності програми, вона має базуватися на комп'ютері. На протязі усього сеансу роботи додаток не повинен збоїти або зависати. Після оновлення інформаційної бази або функціоналу, додаток треба перезапустити.

Більш того до інформаційної бази, яку використовує додаток, висовуються наступні умови:

- вносити зміни до бази може тільки адміністратор;
- оновлення бази слід виконувати дотримуючись правил зберігання.

3.2 Проектування структури програмної системи

При проектуванні структури майбутньої програми було виділено 4 основних програмних модуля, які представлено на рисунку 3.1.

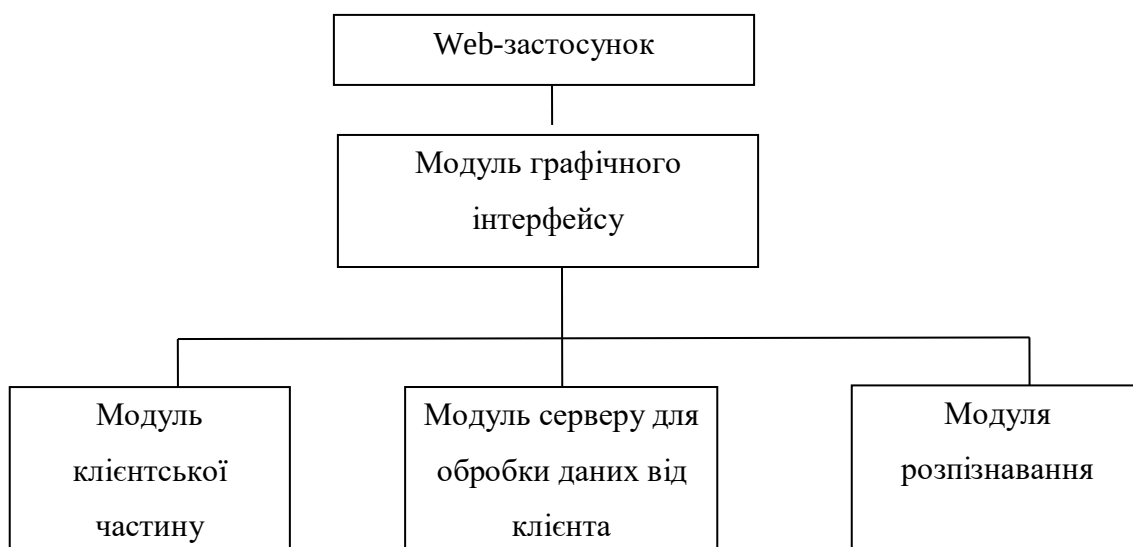


Рисунок 3.1 – Основні програмні модулі

Розробку представлених модулів можна аргументувати тим, що саме розділення основного функціоналу програмного комплексу за таким принципом гарантує формування функціонально завершених блоків програми.

3.3 Вибір мови програмування

3.3.1 Мова програмування для взаємодії із web-сайтом та web-камерою

JavaScript – динамічна, об'єктно-орієнтована прототипна мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується для створення сценаріїв веб-сторінок, що надає можливість на боці клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд web-сторінки [20],[21].

JavaScript класифікують як прототипну (підмножина об'єктно-орієнтованої), скриптову мову програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми програмування (імперативну та частково функціональну) і деякі відповідні архітектурні властивості, зокрема: динамічна та слабка типізація, автоматичне керування пам'яттю, прототипне наслідування, функції як об'єкти першого класу.

Мова JavaScript використовується для:

- написання сценаріїв веб-сторінок для надання їм інтерактивності;
- створення односторінкових веб-застосунків (React, AngularJS, Vue.js);
- програмування на боці сервера (Node.js(Express.js));
- стаціонарних застосунків (Electron, NW.js);
- мобільних застосунків (React Native, Cordova);
- сценаріїв в прикладних програмах (наприклад, в програмах зі складу Adobe Creative Suite чи Apache JMeter);
- всередині PDF-документів тощо.

Незважаючи на схожість назв, мови Java та JavaScript є двома різними мовами, що мають відмінну семантику, хоча й мають схожі риси в стандартних бібліотеках та правилах іменування. Синтаксис обох мов отриманий «у спадок» від мови C, але семантика та дизайн JavaScript є результатом впливу мов Self та Scheme [22].

JavaScript має C-подібний синтаксис, але в порівнянні з мовою C має такі корінні відмінності [23]:

- об'єкти, з можливістю інтроспекції і динамічної зміни типу через механізм прототипів;
- функції як об'єкти першого класу;
- обробка винятків;
- автоматичне приведення типів;
- автоматичне збирання сміття;
- анонімні функції.

3.3.2 Мова програмування для модуля розпізнавання

C++ – мова програмування високого рівня з підтримкою кількох парадигм програмування: об'єктно-орієнтованої, узагальненої та процедурної. Розроблена Б'ярном Страуструпом (англ. Bjarne Stroustrup) в AT&T Bell Laboratories (Мюррей-Хілл, Нью-Джерсі) 1979 року та початково отримала назву «Сі з класами». Згодом Страуструп перейменував мову на C++ у 1983 р. Базується на мові програмування С. Вперше описана стандартом ISO/IEC 14882:1998, найбільш актуальним же є стандарт ISO/IEC 14882:2014 [24]-[25].

У 1990-х роках C++ стала однією з найуживаніших мов програмування загального призначення. Мову використовують для системного програмування, розробки програмного забезпечення, написання драйверів, потужних серверних та клієнтських програм, а також для розробки розважальних програм, наприклад, відеоігор. C++ суттєво вплинула на інші популярні сьогодні мови програмування: C# та Java [24].

При створенні C++ прагнули зберегти сумісність з мовою С. Більшість програм на С справно працюватимуть і з компілятором C++. C++ має синтаксис, заснований на синтаксисі С (див. список операторів мов С та C++).

Нововведеннями C++ порівняно з С є [24, 25]:

- підтримка об'єктно-орієнтованого програмування через класи;
- підтримка узагальненого програмування через шаблони;
- доповнення до стандартної бібліотеки;
- додаткові типи даних;
- обробка винятків;
- простори імен;
- вбудовані функції;
- перевантаження операторів;
- перевантаження імен функцій;
- посилання і оператори управління вільно розподіленою пам'яттю.

У 1998 році ратифіковано міжнародний стандарт мови C++: ISO/IEC 14882 «Standard for the C++ Programming Language». Поточна версія цього стандарту — ISO/IEC 14882:2011.

У таблиці 3.1 наведено порівняння зв'язки C++ та OpenCV з іншими мовами програмування та бібліотеками для роботи із задачами розпізнавання зображень.

Таблиця 3.1 – Порівняння мов програмування

Критерії порівняння	Мови програмування		
	C++ та OpenCV	Python та TensorFlow	MATLAB
Наявність літератури та простота вивчення	++	++	++
Підтримка парадигми модульного програмування	±	+	–
Продуктивність роботи	++	+	–
Паралельна обробка запитів та реалізація	++	++	±
Простота розроблення графічного інтерфейсу	+	±	±

Як видно з таблиці зв'язка C++ та OpenCV добре себе зарекомендувала. Такий підхід може дуже доречно використовуватися для низки задач не пов'язаних із детальним розпізнаванням зображень для подальшої класифікації або діагностування.

3.3.3 Бібліотека OpenCV

OpenCV – бібліотека функцій та алгоритмів комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення з відкритим кодом. Бібліотека надає засоби для обробки і аналізу вмісту зображень, у тому числі розпізнавання об'єктів на фотографіях (наприклад, осіб і фігур людей, тексту тощо), відстежування руху об'єктів, перетворення зображень,

застосування методів машинного навчання і виявлення загальних елементів на різних зображеннях [26],[27].

Бібліотека розроблена Intel і нині підтримується Willow Garage та Itseez. Сирцевий код бібліотеки написаний мовою C++ і поширюється під ліцензією BSD. Біндинги підготовлені для різних мов програмування, таких як Python, Java, Ruby, Matlab, Lua та інших. Може вільно використовуватися в академічних та комерційних цілях.

Бібліотека містить понад 2500 оптимізованих алгоритмів, серед яких повний набір як класичних так і практичних алгоритмів машинного навчання і комп'ютерного зору. Алгоритми OpenCV застосовують у таких сферах [26],[27]:

- аналіз та обробка зображень;
- системи з розпізнавання обличчя;
- ідентифікації об'єктів;
- розпізнавання жестів на відео;
- відстежування переміщення камери;
- побудова 3D моделей об'єктів;
- створення 3D хмар точок зі стерео камер;
- склеювання зображень між собою, для створення зображень всієї сцени з високою роздільною здатністю;
- система взаємодії людини з комп'ютером;
- пошуку схожих зображень із бази даних;
- усунення ефекту червоних очей при фотозйомці зі спалахом;
- стеження за рухом очей;
- аналіз руху;
- ідентифікація об'єктів;
- сегментація зображення;
- трекінг відео;
- розпізнавання елементів сцени і додавання маркерів для створення доповненої реальності та інші.

OpenCV написана на C++ і її основний інтерфейс також реалізовано на C++, але бібліотека і досі представляє старіший C інтерфейс. Наразі реалізовано інтерфейс мовами Python, Java і MATLAB / OCTAVE (починаючи з версії 2.5). API для цих інтерфейсів можна знайти в онлайн документації. Оболонки для інших мов, таких як C#, C#, Ruby були розроблені з метою охоплення ширшої аудиторії.

Всі нові розробки та алгоритми OpenCV наразі розробляються у C++ інтерфейсі.

3.4 Вибір середовища розробки

Microsoft Visual Studio — серія продуктів фірми Майкрософт, які включають інтегроване середовище розробки програмного забезпечення та низку інших інструментальних засобів [28]. Ці продукти дозволяють розробляти як консольні програми, так і програми з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також веб-сайти, веб-застосунки, веб-служби як в рідному, так і в керованому кодах для всіх платформ, що підтримуються Microsoft Windows, Windows Mobile, Windows Phone, Windows CE, .NET Framework, .NET Compact Framework та Microsoft Silverlight.

Visual Studio включає один або декілька з наступних компонентів:

- Visual Basic .NET, а до його появи — Visual Basic;
- Visual C++;
- Visual C#;
- Visual F# (входить до складу Visual Studio 2010);
- Visual Studio Debugger.

Багато варіантів постачання також включають:

- Microsoft SQL Server;
- MSDE Visual Source Safe — файл-серверна система управління версіями.

Версія Visual Studio 2015 стала першою офіційно безкоштовною – була представлена 20 червня 2015 року. Суттєвою зміною стала підтримка багатьох цільових платформ: окрім базової Windows з'явилась можливість будувати проекти для IOS та Android. Для розробників комп'ютерних ігор була додана підтримка фреймворку Unity. Був оновлений механізм аутентифікації: користувач під час запуску Visual Studio синхронізується з єдиним аккаунтом Microsoft. Версія має в собі .NET Framework 4.6 та підтримку універсальної платформи Windows 10. Розробників на мові C++ потішили новим функціоналом стандарту C++14, та навіть деякими поліпшеннями з C++17. Останнім оновленням на цей момент є Update 2 від 30 березня 2016-го року, у якому багато уваги приділено стабільності, та продовжено роботу у напрямку підтримки нових стандартів мови C++.

Версія Visual Studio 2017 побачила світ 30 березня 2016 року. Основними змінами стали інтерфейс інсталятора та численні незначні покращення у різних компонентах середовища розробки. Очікується підтримка мови програмування Solidity. Версія остаточно побачила світ під назвою Visual Studio 2017 7-го березня 2017-го року.

Версія Visual Studio 2019 була випущена у грудні 2018 року. Вона мала 5 етапів. У березні був вже визначений RC (Release Candidate). Тому через місяць, а саме 2-го квітня була випущена стабільна версія Visual Studio 2019 на Windows та Visual Studio on MAC. Як і у минулих версіях є три типи (Community, Professional, Enterprise).

3.5 Вимоги до технічних засобів

Для нормальної розробки програми, повинні виконуватися наступні умови до апаратної частини:

- наявність IBM-сумісної персональної електронної обчислювальної машини (ПЕОМ), на якій буде працювати додаток;
- процесор із тактовою частотою 1,5 GHz та більше;

- оперативна пам'ять 1,5 ГБ та більше;
- вільний простір на жорсткому диску не менше, ніж 2 ГБ (для збереження зображень).

Вимоги до програмної частини:

- завантажена бібліотека OpenCV.

3.6 Висновки за розділом 3

Під час роботи над розділом було спроектовано архітектуру майбутнього ПП. Були спроектовані модулі майбутньої програми. Було проаналізовано найбільш популярні мови програмування та після ретельного порівняння перевага була віддана JavaScript та зв'язці C++ і OpenCV.

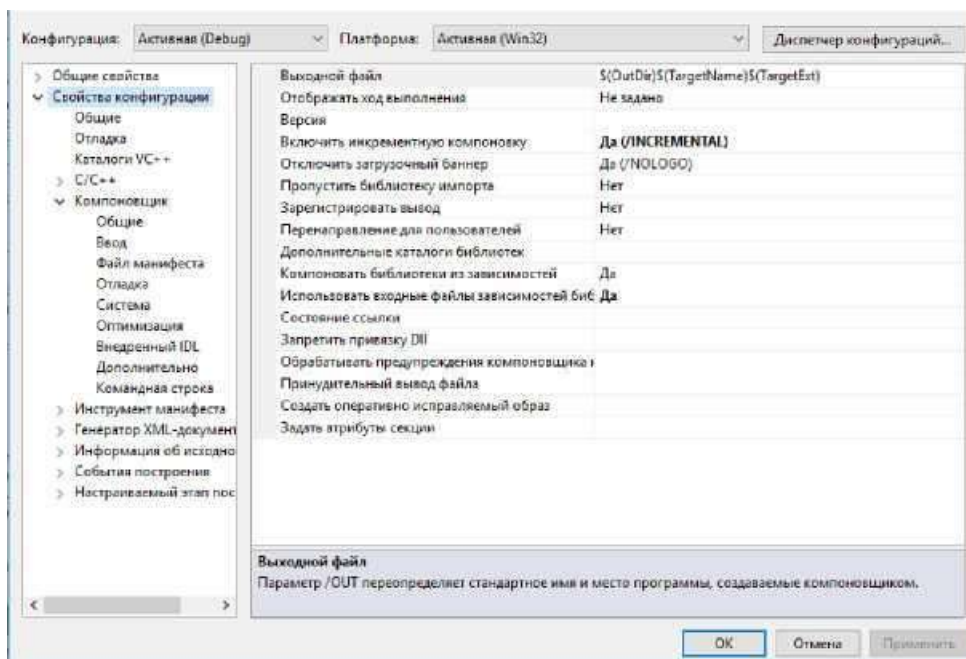


Рисунок 4.2 – Використання вхідних файлів залежностей бібліотека

У загальних властивостях компоувальника необхідно включити інкрементну компоновку і вказати, що необхідно використовувати вхідні файли залежностей бібліотек.

Для debug і release конфігурацій необхідно підключити потрібні бібліотеки, які будуть використовуватися в проекті. Ліба для debug конфігурації має в назві додаткову букву d на рисунку 4.3 здійснюється процес підключення бібліотек для різних конфігурацій.

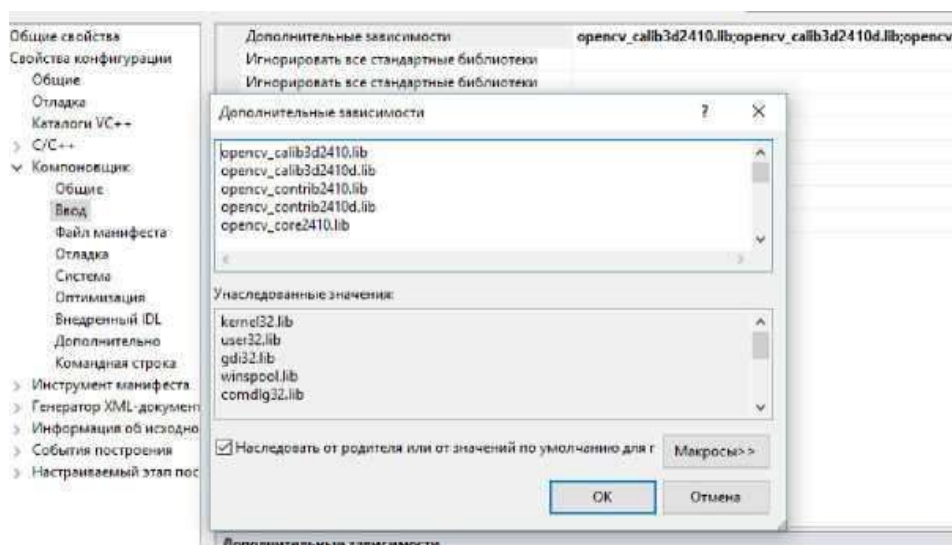


Рисунок 4.3 – Підключення необхідних бібліотек

4.2 Використовуваний метод розпізнавання і його реалізація

У якості методу порівняння двох зображень використовується метод швидкої нормалізованої взаємної кореляції. Кореляція між двома зображеннями є одним з підходів для виявлення ознак подібності двох зображень. Критерієм подібності зображення з одним з еталонних зображень є коефіцієнт взаємної кореляції. Ідентичними будуть ті зображення, у яких коефіцієнт кореляції буде 1.

За допомогою функції `cvmatchtemplate` відбувається порівняння зображення, отриманого від користувача з шаблонами, що зберігаються в базі.

Слід розглянути функцію: `cvMatchTemplate (image, templ, result, CV_TM_CCORR_NORMED)`.

Тут, `image` – зображення, отримане від користувача; `templ` – шаблон, що зберігається в базі; `result` – результат роботи функції; `CV_TM_CCORR_NORMED` – метод нормалізованої кросскореляції

Шаблон зображення повинен бути менше, ніж вихідне зображення.

Функція ковзає по зображенню, порівнюючи перекриті ділянки розміром $m \times n$ від шаблону, використовуючи метод нормалізованої кросскореляції. Ковзання відбувається зліва направо, зверху вниз. Результат порівняння записується в змінну `result`. Для функції `CV_TM_CCORR_NORMED` формула (2.3) буде мати вигляд:

$$R(x, y) = \frac{\sum_{\langle x', y' \rangle \in X} (T'(x', y') \cdot I'(x + x', y + y'))}{\sqrt{\sum_{\langle x', y' \rangle \in X} (T'(x', y'))^2} \sqrt{\sum_{\langle x, y \rangle \in X} (I'(x + x', y + y'))^2}}, \quad (4.1)$$

де $x' = 0 \dots m - 1$ та $y' = 0 \dots n - 1$;

T – Template – шаблон, який шукаємо на зображенні;

I – Image – зображення, в якому очікуємо знайти відповідності.

Мета полягає в тому, щоб виявити найвищу область відповідності. Для кожного розташування T над I записується метрика в матрицю результатів R . Кожне місце (x, y) в R містить метрику відповідності.

Після порівняння з масиву результатів необхідно обчислити мінімальне і максимальне значення. Для обчислення даних значень використовується функція `cvMinMaxLoc`.

Слід розглянути функцію: `cvMinMaxLoc(result, &min, &max, *rmin, *rmax)`.

Тут `result` – це результат функції `cvMatchTemplate`, `min` – мінімальне значення, `max` – відповідно максимальне значення, `rmin` – покажчик на точку розташування мінімуму, а `rmax` – покажчик на точку розташування максимуму.

У програмі в якості шаблонів використовуються 8 зображень в розмірі 180x220. Кожен шаблон являє собою певне положення щодо web-камери. Використання вибірки з n -кількості зображень дозволяє знизити ймовірність появи помилки 2 роду. Дослідним шляхом було виявлено, що використовувати 8 шаблонів оптимально для завдання, поставленого в роботі.

4.3 Реалізація клієнтської частини

Захоплення зображення для подальшого розпізнавання проводиться на клієнті. Користувачеві для аутентифікації необхідно натиснути кнопку для захоплення зображення. Потім зображення потрапляє на сервер для подальшої обробки.

В якості захоплюючого пристрою виступає web-камера ноутбука. На формі знаходяться такі елементи, як:

- «Video» – в нього буде відтворюється потокове відео отримане з web-камери;
- «Canvas» – в нього будуть поміщені кадри для збереження;
- кнопка «start» – кнопка для захоплення зображення;
- «Value» – поле в яке виводиться відповідь отриманий від сервера.

Для зручності користування користувачеві будуть виводиться підказки. Дозвіл на використання відеопотоку користувача зображено на рисунку 4.4. На рисунку 4.5 представлений інтерфейс користувача.

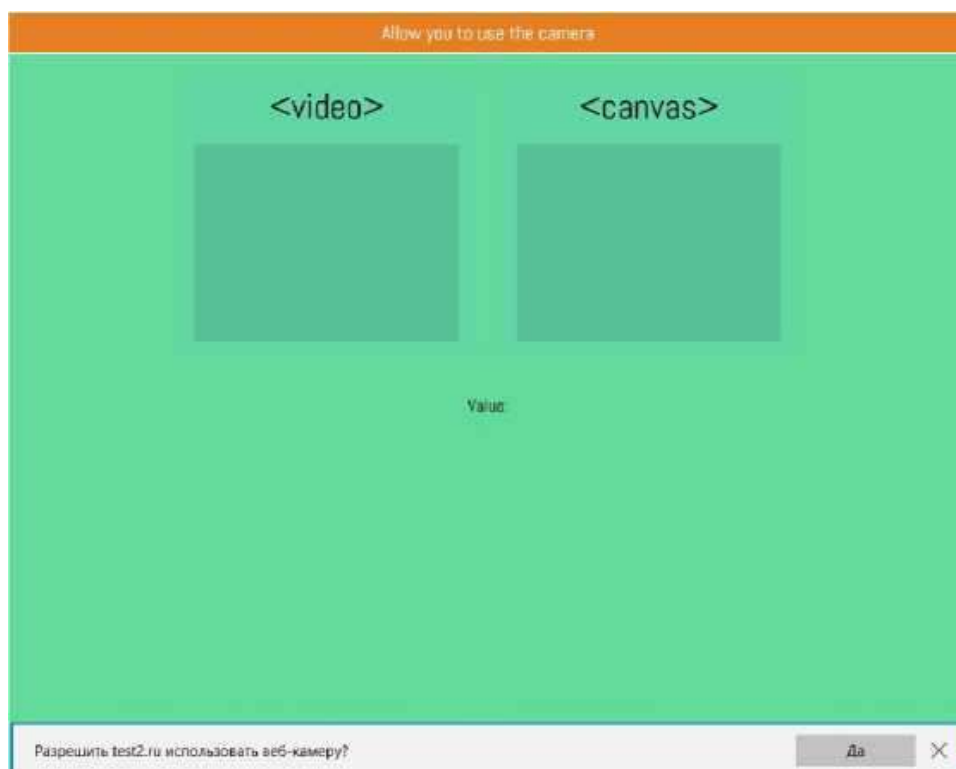


Рисунок 4.4 – Дозвіл використання web-камери

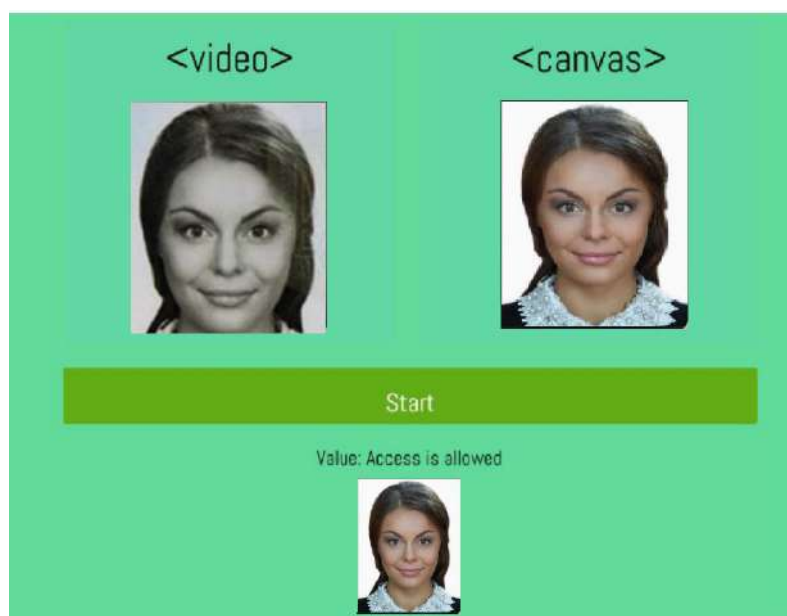


Рисунок 4.5 – Web-клієнт

Для запиту дозволу на використання відеопотоку користувача застосовується наступна конструкція:

```
{
  // разрешение от пользователя получено // приховуємо підказку
allow.style.display = "none";
  // получаем url поточного видео
  videoStreamUrl = window.URL.createObjectURL(stream); // встановлюємо,
як джерело для video video.src = videoStreamUrl; button.style.display = 'block';
}, function () {
  console.log (' щось не так з відеострімом');
});
```

У разі успіху в callback повернеться об'єкт типу stream, по якому можна отримати url відеопотоку. Для цього існує конструкція window.URL.createObjectURL(stream).

Далі передаємо даний URL об'єкту video, при натисканні кнопки захоплюємо поточний кадр video в canvas. Беремо data: url зображення з canvas і передаємо його на сервер. Частина коду реалізації представлений нижче. А інший код буде представлений в Додатку.

4.4 Сервер

Сполучною ланкою між клієнтською частиною і модулем розпізнавання є сервер. Після отримання зображення від клієнта в закодованому вигляді відбувається декодування. Отримане зображення зберігається в базу зображень. Викликається модуль розпізнавання, який бере отримане зображення з бази і порівнює з шаблонами. Результат порівняння надходить на сервер, де відбувається аналіз. За результатами формується відповідь, яка відображається на формі у клієнта.

Введемо позначення False Acceptance Rate (FAR) – процентний поріг, що визначає ймовірність того, що одна людина може бути прийнятий за іншого

(коефіцієнт помилкового доступу – за природою схожий до помилки 2 роду). Величина $1 - FAR$ називається специфічність.

False Rejection Rate (FRR) – ймовірність того, що людина може бути не розпізнаний системою (коефіцієнт помилкової відмови в доступі – за природою схожий до помилки 1 роду). Величина $1 - FRR$ називається чутливість.

Результат розпізнавання порівнюється з кількома пороговими значеннями, які були обрані на підставі перетину графіків FAR і FRR. Даний графік буде наведено в главі 4. Було обрано три основні пороги:

- 0,84 – аутентифікація пройдена успішно, доступ дозволений;
- 0,81-0,84 – повторна аутентифікація;
- $< 0,81$ – аутентифікація пройдена невдало, доступ закритий.

Якщо в результаті повторної аутентифікації коефіцієнт схожості потрапляє в діапазон від (0,81; 0,84), то порогом доступу буде точка EER. Дана точка знаходиться на перетині графіків FAR і FRR. У нашому випадку значення даної точки дорівнює 0,83. Доступ буде дозволений, якщо значення коефіцієнта схожості буде більше 0,83, і заборонений в протилежному випадку. За даними аутентифікації буде сформована відповідь. Відповідь надсилається в клієнтську частину, де Користувачеві виводиться результат аутентифікації.

4.5 Висновки за розділом 4

Розроблено програмну для реалізації аутентифікації за біометричними даними користувача, яка повністю реалізує поставлену задачу. Обрано механізм реалізації відповідних методів та алгоритмів, а також елементи керування, з якими користувачу буде зручно використовувати розроблений програмний продукт (ПП).

5 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

Цей розділ буде присвячено опису розробленого програмного комплексу. Буде означено сферу застосування, та наведено алгоритм користування. Також у цьому розділі буде проведено тестування та оцінка отриманих результатів.

5.1 Опис застосування програми

Для тестування системи використовувалися біометричні характеристики трьох користувачів. Два не зареєстрованих і один зареєстрований. Для двох типів користувачів були зроблені вибірки, що складаються з 100 фотографій. Для зареєстрованого користувача було зроблено 8 шаблонів. Кожен шаблон характеризується положенням голови щодо камери. Шаблон являє собою обличчя людини без сторонніх об'єктів. Приклад шаблону зображений на рисунку 5.1.



Рисунок 5.1 – Шаблон

Для того, щоб визначити точний поріг доступу необхідно провести тестування системи на двох групах користувачів (зареєстрованих і незареєстрованих). На рисунку 5.2 подано графік залежності кількості аутентифікацій від значення коефіцієнта кореляції.

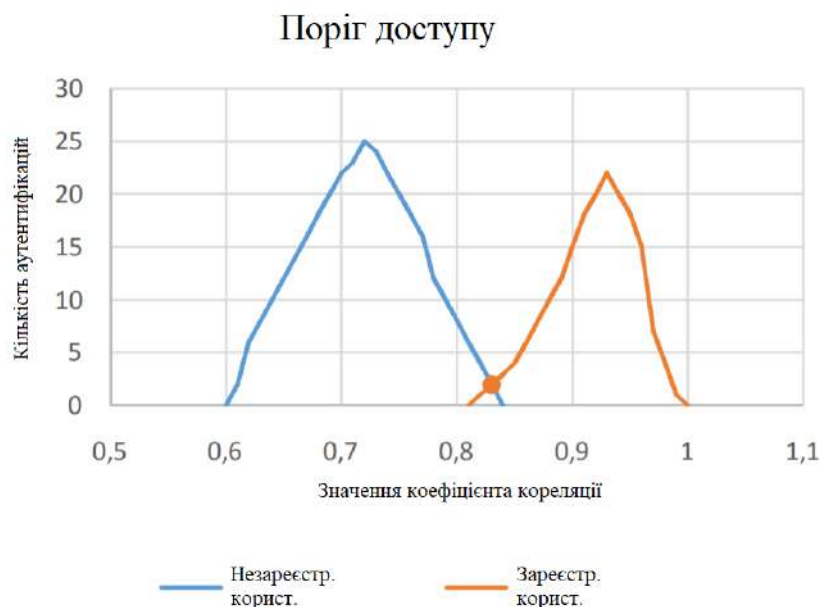


Рисунок 5.2 – Графік порогового доступу

На графіку добре видно два піки значень кореляції. Значення кореляції, яке відповідає максимальній кількості аутентифікацій незареєстрованих користувачів і зареєстрованих. При проведенні тестування системи на незареєстрованих користувачів коефіцієнт коливався в діапазоні (0,6, 0,84), максимальна кількість аутентифікацій проходила з коефіцієнтом 0,73. Коефіцієнт кореляції для зареєстрованих користувачів знаходився в діапазоні (0,81, 1), максимальна кількість аутентифікацій проходило з коефіцієнтом 0,93.

У точці зі значенням 0,83 два графіка перетинаються. Значення в даній точці буде пороговим для системи. При значенні порога розпізнавання 0,83 помилки першого і другого роду для даної системи будуть мінімальні. Початковим порогом розпізнавання було значення рівне 0,9. Число помилок FRR було більше. Імовірність помилки $FRR \approx 0,08$.

5.2 Вибір порогу розпізнавання

При експериментальній перевірці FRR необхідно послідовно n раз пред'являти біометричні характеристики користувача, який успішно пройшов реєстрацію. Підраховується відношення числа невдалих спроб до загального

числа всіх спроб. Отриманий результат дає оцінку ймовірності помилки FRR. При значенні $n \geq 1/FRR$ вважається, що оцінка є достовірною.

Щоб перевірити характеристику FAR системі пред'являють m раз біометричну характеристику користувачів, які не пройшли реєстрацію. Підраховується кількість n^2 вдалих спроб (позитивна аутентифікація) до загального числа всіх спроб m . отримане відношення показує ймовірність помилки FAR. При значенні $m \geq 1/FAR$ оцінка вважається достовірною.

Для підрахунку FAR і FRR було проведено 100 експериментів, як для FAR, так і для FRR. Результати будуть представлені нижче в таблиці 5.1.

Таблиця 5.1 – Результати FAR і FRR

	FAR	FRR
Кількість експериментів	$m=100$	$n=100$
Кількість (вдалих / невдалих) результатів	1	3
Імовірність помилки	0.01	0.03
Достовірність результатів	$m \geq 1/FAR$ $m=100$ достовірний	$n \geq 1/FAR$ $n>33.3$ достовірний

Порахуємо кількість користувачів для стабільної роботи системи при поточному значенні FAR. Для розрахунків використовується формула:

$$P \approx \sqrt{1/FAR}. \quad (5.1)$$

де P – кількість користувачів;

FAR – ймовірність помилки першого роду.

При $FAR = 0,01$ число користувачів стабільно використовують систему буде рівним 10. Це досить, тому що кількість користувачів на одному персональному комп'ютері для даної системи не перевищить даного значення.

5.3 Висновки за розділом 5

Таким чином, в ході виконання роботи було вирішено поставлене завдання. Була створена система аутентифікації на web-сайті. Система складається з трьох модулів: розпізнавання, клієнтської частини і сервера. Дані модулі були повністю реалізовані у відповідності з усіма поставленими вимогами.

Система була протестована на групі користувачів, як зареєстрованих в системі, так і не зареєстрованих. Тестові дані вважаються достовірними.

Для системи були обчислені ймовірності помилок першого і другого роду. При такій ймовірності помилок система працює стабільно.

Надалі планується модернізувати модуль розпізнавання системи, тому що розробляється більш досконалий модуль, заснований на порівнянні 3D моделей особи. Дане модернізовані дозволить ширше використовувати дану систему.

6 ОРГАНІЗАЦІЙНО-ЕКОНОМІЧНА ЧАСТИНА

В даному розділі приводиться економічне обґрунтування створення програмного забезпечення (ПЗ), оцінка затрат на його створення, розрахунок економічної ефективності та терміну окупності [29].

У дипломній роботі магістра досліджено та реалізовано систему аутентифікації на основі біометричних даних користувача.

Очікується, що впровадження в експлуатацію розробленого ПЗ у банківських або державних підприємствах підвищить рівень безпеки та дозволить скоротити витрати на системи безпеки, зокрема 14000 гривень на місяць.

У розробці беруть участь програміст із заробітною платою 14500 грн. в місяць і консультант з заробітною платою 9500 грн. у місяць. На розробку проєкту було витрачено 2 місяці. Розробка програми розпочалася першого вересня і повинна бути виконана до кінця жовтня 2020 року.

6.1 Планування розробки програмного продукту

Весь комплекс розробки програмного продукту можна розділити на етапи [24]. Для кожного етапу вказуються трудомісткість, кількість виконавців і тривалість робіт. Інформація з характеристики робіт по розробці програмного забезпечення зведена у таблицю 6.1.

Таблиця 6.1 – Характеристика робіт по розробці програми

Найменування робіт	Трудомісткість		Виконавці	Тривалість, днів
	люд-дн	%		
1	2	3	4	5
Аналіз ПЗ	4	6	Програміст	2
			Консультант	2
Визначення вимог до ПП	8	11,9	Програміст	4
			Консультант	4

Продовження таблиці 6.1

1	2	3	4	5
Проектування структури програми	32	47,8	Програміст Консультант	16 16
Розробка схеми функціонування програми	3	4,5	Програміст	3
Створення програмного коду	9	13,4	Програміст	9
Тестування і налагодження програмного продукту	7	10,4	Програміст	7
Складання програмної документації	4	6	Програміст	4
Разом	67	100	Програміст Консультант	45 22

6.2 Визначення витрат на розробку програми

Для визначення витрат на розробку програми складають калькуляцію кошторисної вартості робіт, що включає статті:

- основна заробітна плата;
- додаткова заробітна плата;
- відрахування на єдиний соціальний внесок (ЄСВ);
- матеріали та комплектуючі вироби;
- витрати на спеціальне обладнання (на оплату машинного часу ЕОМ для написання і налагодження програмних засобів);
- накладні витрати.

Витрати за статтею «Основна заробітна плата» складаються з планового фонду зарплати всіх категорій працівників, зайнятих в розробці програми. Розрахунок зарплати на підставі даних про трудомісткість, представлений в таблиці 6.2.

Таблиця 6.2 – Основна заробітна плата

Посада виконавця	Чисельність, чол	Місячний оклад	Кількість місяців роботи	Сума ЗП, грн
Програміст	1	14500	2	29000
Консультант	1	9500	1	9500
Разом	2			38500

Додаткову заробітну плату приймають рівною 10% від основної зарплати. Вона розраховується за формулою (6.1):

$$ЗП_{\text{дод}} = ЗП_{\text{осн}} * 10\%. \quad (6.1)$$

Підставивши величину основної заробітної плати в дану формулу отримаємо: $ЗП_{\text{дод}} = 39500 * 0.1 = 3850$ грн.

Відрахування на єдиний соціальний внесок (ЄСВ) визначають у відсотковому відношенні від сум основної та додаткової зарплати з урахуванням премій і доплат. Відрахування вираховуються за формулою (6.2):

$$\begin{aligned} OT &= (ЗП_{\text{осн}} + ЗП_{\text{дод}}) * 0,22. \\ OT &= (38500 + 3850) * 0,22 = 9317 \text{ грн.} \end{aligned} \quad (6.2)$$

Використовується чотири найменування матеріалів: картридж – 700 грн, захищений флеш-носій – 500 грн., зовнішній SDD – 1500 грн, папір – 90 грн. Витрати на матеріали розраховують за формулою (6.3):

$$M = \sum_{i=1}^n (Ц_i * N_i * (1 + K_{\text{м.з.}}) - Ц_{io} * N_{io}), \quad (6.3)$$

де М – витрати на матеріали, покупні напівфабрикати і комплектуючі вироби, грн .;

C_i – ціна і-го найменування матеріалу, напівфабрикату і комплектуючого, грн .;

N_i – потреба в і-му матеріалі, напівфабрикаті і комплектуючому;

$K_{m.3}$ – коефіцієнт, що враховує транспортно-заготівельні витрати;

C_{io} – ціна зворотних відходів і-го найменування матеріалу, грн;

N_{io} – кількість зворотних відходів і-го найменування;

n – кількість найменувань матеріалів, напівфабрикатів і комплектуючих.

$$C_{io} = 0; N_{io} = 0; K_{m.3} = 0,05;$$

$$M = (1 + 0,05) * (700 + 500 + 1500 + 90) = 2929,5 \text{ грн.}$$

Таким чином, разом витрати на матеріали та комплектуючі вироби становлять 2929,5 гривень.

У статтю «Витрати на спеціальне обладнання» входять витрати на придбання, транспортування, монтаж і налагодження нестандартного обладнання. Практично, в даному випадку, в цій статті враховуються витрати на оплату машинного часу ЕОМ для написання і налагодження програмних засобів. Для цього необхідно скласти кошторис «витрат на утримання і експлуатацію устаткування», виходячи з якого визначиться вартість одного машино-години роботи ПК, після множення якої на машинний час пішло на написання і налагодження програми отримаємо витрати на оплату машинного часу.

Амортизаційні відрахування визначають за формулою (6.4):

$$A = BB * \frac{H_a}{100}, \quad (6.4)$$

де BB – балансова вартість обчислювальної техніки, грн;

H_a – норма амортизаційних відрахувань на повне оновлення обчислювальної техніки,%. Приймаємо, що техніка повинна працювати 4 роки, тому $H_a = 25\%$.

Балансова вартість обчислювальної техніки становить 38000 грн. Тому амортизаційні відрахування становлять:

$$A = 48000 * 0,25 = 12000 \text{ грн}$$

Статтю "Експлуатація обладнання" розраховують підсумовуванням витрат на силову електроенергію і допоміжні матеріали. Витрати на електроенергію визначають за формулою (6.5):

$$C_{\text{э}} = N_{\text{н}} * \Phi_{\text{эф}} * K_{\text{зв}} * K_{\text{зм}} * C_{\text{э}}, \quad (6.5)$$

де $N_{\text{н}}$ – номінальна потужність ЕОМ, кВт;

$\Phi_{\text{эф}}$ – річний ефективний фонд часу роботи ЕОМ, машино–год;

$K_{\text{зв}}$ – середній коефіцієнт завантаження за часом;

$K_{\text{зм}}$ – коефіцієнт завантаження по потужності;

$C_{\text{э}}$ – ціна одного кВт * год електроенергії, грн ./ (кВт * год).

Номінальна Потужність ЕОМ – 0,2 кВт. Річний ефективний фонд часу роботи ЕОМ ставити 1800 годин. Середні коефіцієнти завантаження за часом та за потужністю рівні відповідно 0,9 та 0,6. Ціна однієї кіловат–години електроенергії ставить 2,11 грн. Отримуємо:

$$C_{\text{е}} = 0,2 * 1800 * 0,9 * 0,6 * 2,11 = 410,18 \text{ грн.}$$

Приймаємо, що $C_e = 410$ грн.

Витрати за статтею "Заробітна плата обслуговуючих робітників та відрахування на соціальне страхування в інші фонди" визначають за формулою (6.6):

$$ЗП_{\text{обсл}} = ФЗП_{\Gamma} * (1 + K_{\text{відрах}}) * \frac{t_{\text{обсл}}}{\Phi_{\text{еф.обсл}}}, \quad (6.6)$$

де $ФЗП_{\Gamma}$ – річний фонд заробітної плати (основної та додаткової) обслуговуючих робітників, грн.;

$K_{\text{відрах}}$ – коефіцієнт, що враховує відрахування на соціальне страхування і в інші фонди;

$t_{\text{обсл}}$ – час протягом року, необхідне технічне обслуговування ЕОМ, годин / рік. Приймаємо $t_{\text{обсл}} = 30$ год.;

$\Phi_{\text{еф.обсл}}$ – річний ефективний фонд часу обслуговуючого персоналу, годин / рік. Приймаємо, що річний ефективний фонд часу обслуговуючого персоналу становить 1800 годин

Заробітна плата обслуговуючого персоналу = 10800 грн/місяць.

$$ФЗП_{\Gamma} = 10800 * 12 = 129600 \text{ грн.}$$

$$ЗП_{\text{обсл}} = 129600 * (1 + 0,22) * 30/1800 = 2635,2 \text{ грн.}$$

Суму витрат за статтею "Поточний ремонт обладнання" обчислимо як 3% від балансової вартості обладнання, оскільки неможливо заздалегідь визначити, скільки коштів необхідно витратити на придбання запасних частин наявного обладнання. Балансова вартість обчислювальної техніки становить 48000 грн. Тому сума витрат за статтею "Поточний ремонт обладнання" становить: $48000 * 0,03 = 1440$ грн.

Інших витрат за проектом не передбачено, тому сума витрат за цією статтею дорівнює нуль гривень.

Кошторис витрат на утримання і експлуатацію устаткування наведений в таблиці 6.3.

Таблиця 6.3 – Кошторис витрат на утримання і експлуатацію устаткування

Найменування статей витрат	Сума, грн.
Амортизація обладнання	12000
Заробітна плата основна і додаткова обслуговуючих робітників з відрахуваннями на соціальні заходи	2635,2
Експлуатація обладнання (крім витрат на поточний ремонт)	410
Поточний ремонт обладнання	1440
Інші витрати	0
Разом	16485,2

Витрати на оплату машинного часу ЕОМ для написання і налагодження програмних засобів визначаються за формулою (6.7):

$$C_{mo} = P_{екс} * t_{мо}, \quad (6.7)$$

де C_{mo} – витрати на оплату машинного часу, грн;

$P_{екс}$ – експлуатаційні витрати на одну годину машинного часу цієї цифрової ЕОМ, грн. / машино–год.;

$t_{мо}$ – машинний час цифрової ЕОМ для написання і налагодження даного програмного продукту, машино–год.

$$P_{екс} = 16485,2 / 1800 = 9,16 \text{ грн./ машино– год.}$$

Для написання і налагодження даного програмного продукту ЕОМ експлуатується протягом 45 днів в одну зміну по 8 годин, що складає в сумі 360 годин. Таким чином отримуємо суму за статтею «Витрати на спеціальне обладнання»:

$$C_{\text{мо}} = 9,16 * 360 = 3297,6 \text{ грн.}$$

До накладних витрат відносяться витрати на загальне управління і загальногосподарські потреби (заробітна плата апарату управління, канцелярські витрати і т.д.), утримання та експлуатацію будівель. Накладні витрати включаються до вартості розробки програми непрямым шляхом – у відсотках до основної заробітної плати розробників. Для обчислення цієї статті видатків у дипломному проєкті приймемо, що накладні видатки становлять 40% від основної заробітної плати розробників, що складає 38500грн. Тому сума накладних видатків становить: $38500 * 0,40 = 15400$ грн.

Калькуляція кошторисної вартості робіт з розробки програми наведена в таблиці 6.4.

Таблиця 6.4 – Калькуляція кошторисної вартості робіт з розробки програми

Найменування статей витрат	Сума, грн.
Основна заробітна плата	38500
Додаткова заробітна плата	3850
ЄСВ	9317
Матеріали і комплектуючі	2929,5
Витрати на спеціальне обладнання	3297,6
Накладні витрати	15400
Разом	73294,1

6.3 Розрахунок економічної ефективності програмного продукту

Річну економію від впровадження програми, що дозволяє знизити витрати машинного часу ЕОМ для рішення певного завдання в порівнянні із програмою, що застосовувалася раніше, визначають за формулою:

$$E = (T_{M_1} - T_{M_2}) \cdot B_{\text{екс.КТЗ}} + E_{\text{н}}^a \cdot \frac{\Phi_{\text{б.КТЗ}} \cdot (T_{M_1} - T_{M_2})}{\Phi_{\text{еф.КТЗ}}} - \frac{S_{\text{рп}}}{T_{\text{с}}} + E_{\text{зп}}, \quad (6.8)$$

де T_{M_1}, T_{M_2} – машинний час, необхідний для розв’язання поставлених завдань, відповідно в старому та у новому варіантах, машино–год./рік;

$B_{\text{екс}}$ – експлуатаційні витрати, що доводяться на 1 год машинного часу ЕОМ, грн/ машино–год.;

$E_{\text{н}}^a$ – нормативний коефіцієнт ефективності капітальних вкладень у засоби автоматизації;

$\Phi_{\text{б.КТЗ}}$ – балансова вартість ЕОМ (КТЗ), грн.;

$\Phi_{\text{еф.КТЗ}}$ – річний ефективний фонд часу роботи ЕОМ, машино–год.;

$S_{\text{рп}}$ – сумарні витрати на розробку програми, грн.;

$T_{\text{с}}$ – термін служби впроваджуваної програми до її морального зношування, років. Приймаємо цей термін 5 років;

$E_{\text{зп}}$ – економія на заробітній платі.

До впровадження даного ПЗ дані функції виконувались співробітником, що відповідає за обробку рукописних символів, 1800 годин на рік, а після 0 годин на рік (економія складає 1,0 ставки одного співробітника, що відповідає за логістику, зокрема за складання оптимальних шляхів розвезення вантажів, з заробітною платою 11000 гривень на місяць).

$$E = (1800 - 0) * 7,41 + 0,2 * (48000 \cdot (1800 - 0)) / 1800 - 73294,1/5 + \\ + 14000 * 1,0(1 + 0,22) * 12 = 13338 + 9600 - 14658,82 + 204960 = 213239,18 \text{ грн.}$$

Коефіцієнт ефективності капітальних вкладень E и строк їхньої окупності $T_{\text{ок}}$ розраховуються за формулами (6.9) і (6.10):

$$E\phi = \frac{E}{S_{\text{рп}}}, \quad (6.9)$$

$$T_{\text{ок}} = \frac{S_{\text{рп}}}{E}, \quad (6.10)$$

$$E\phi = 213239,18 / 73294,1 = 2,9,$$

$$T_{\text{ок}} = 73294,1 / 213239,18 = 0,34 \text{ року.}$$

Зведемо отримані техніко–економічні показники в таблицю 6.5.

Таблиця 6.5 – Техніко–економічні показники

Показники	Сума	Одиниця виміру
Витрати на розробку програми	73294,1	грн.
Річна економія	213239,18	грн.
Строк окупності програми	0,34	рік

Таким чином, впровадження в експлуатацію розробленого ПЗ на підприємстві дозволить скоротити витрати на обслуговування системи безпеки на 14000 гривень на місяць. Економічні розрахунки показали, що строк окупності програми 0,34 року, що значно менше часу її нормативного строку окупності (5 років). Крім того, річна економія при використанні програми становить 213239,18 грн. Виходячи з виконаних розрахунків, можна зробити висновок, що розробка даного програмного продукту є економічно доцільною.

7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ

7.1 Аналіз потенційних небезпек

Під час проведення дослідження та програмної реалізації методів аутентифікації на основі розпізнавання обличчя, основні роботи по обробці отриманої інформації виконуються за допомогою ПК в обчислювальному центрі. Потенційні небезпечні і шкідливі фактори, що виникають при роботі за комп'ютером, бувають:

- фізичні;
- психофізичні;
- санітарно-гігієнічні;
- пов'язані з порушеннями правил пожежної безпеки;
- пов'язані з проявом наслідків надзвичайних ситуацій.

До фізичних відносяться: вплив електромагнітного поля, випромінюваного робочим комп'ютером на людину; погіршення зору, пов'язане з постійною напругою очей, з поганою освітленістю робочої зони; механічне травмування робітників, в зв'язку з невиконанням правил охорони праці, наприклад, робота з несправним або зламаним обладнанням; ураження струмом, струмопровідними елементами або несправними джерелами живлення, при порушенні цілісності електропроводки.

До психофізичних відносяться: незадовільна організація робочого місця, пов'язана з недостатнім урахуванням вимог ергономіки (відсутність необхідного обсягу робочого місця, неправильне розташування елементів комп'ютера, оснащення робочого місця не відповідним кріслом); негативний вплив постійної роботи за комп'ютером на хребет і скелет людини в цілому, пов'язане зі статичним становищем робітника на робочому місці; дратівливість, в зв'язку з відсутністю або малою кількістю технологічних перерв для роботи за комп'ютером; вплив шумів, що виникають при роботі комп'ютерів.

До санітарно-гігієнічних відносяться: підвищена запиленість, викликана роботою з паперами, з постійним рухом обертових елементів комп'ютера, з

відсутністю достатньої вентиляції приміщення. Може привести до збільшення частоти загальних захворювань.

Небезпечні і шкідливі фактори пов'язані з порушеннями правил пожежної безпеки: коротке замикання, несправність систем охолодження різного устаткування, куріння в недозволеному місці.

Небезпеки пов'язані з проявом наслідків надзвичайних ситуацій: невідповідність персоналу в умовах надзвичайних ситуацій.

7.2 Заходи щодо забезпечення техніки безпеки

З метою зменшення негативного впливу ПК на організм людини враховані вимоги при організації роботи за НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями» [30].

Згідно ПУЕ, щодо безпеки ураження людей електричним струмом, приміщення ОЦ, є приміщенням без підвищеної небезпеки (сухе, добре опалювальне, з струмонепровідною підлогою, з температурою всередині приміщення 18-30°C, з вологістю 40-50%) [31].

Щоб уникнути ураження електричним струмом все електрообладнання на робочому місці заземлено відповідно до ДСТУ 7237:2011 «Система стандартів безпеки праці. Електробезпека. Загальні вимоги та номенклатура видів захисту» [32]. Захисне заземлення, опір якого складає не більше 4 Ом, забезпечує захист людей від ураження електричним струмом при дотику до металевих неструмопровідних частин, які можуть опинитися під напругою в результаті пошкодження ізоляції.

Захисне заземлення виконано навмисним електричним з'єднанням металевих частин електроустановок з «землею» або її еквівалентом.

Захисному заземленню підлягають металеві частини електрообладнання, доступні для дотику людини і не мають інших видів захисту, що забезпечують електробезпеку.

Захисне заземлення електроустановок слід виконувати при:

- при номінальній напрузі 380 В і вище змінного струму та номінальній напрузі 440 В і вище постійного струму - у всіх випадках;
- при номінальній напрузі від 42 В до 380 В змінного струму та від 110 В до 440 В постійного струму при роботах в умовах з підвищеною небезпекою і особливо небезпечних за НПАОП 40.1-1.01-97.

Заземлюючі пристрої, що використовуються для заземлення електроустановок відповідають всім вимогам, що пред'являються до заземлення цих електроустановок.

Матеріал, конструкція і розміри заземлювачів забезпечують стійкість до механічних, хімічних і термічних дій на весь період експлуатації.

Для вирівнювання потенціалів металеві будівельні та виробничі конструкції приєднані до мережі заземлення.

Згідно з НПАОП 0.00-7.15-18 [30] площа на одне робоче місце є не менше ніж 6,0 м², а обсяг - ніж 20,0 м³.

Згідно НПАОП 0.00-7.15-18 неприпустимо розташування приміщень, призначених для роботи з ВДТ в підвалах і цокольних поверхах. Також забороняється розміщення, згідно НАПБ Б.03.002-2007 «Норми визначення категорій приміщень, будинків та зовнішніх установок по вибухопожежної та пожежної безпеки», вибухонебезпечних приміщень категорії А і Б та виробництв з мокрими технологічними процесами поряд з приміщенням, де розташовуються ЕОМ, а також над такими приміщеннями, або під ними [33].

Планування розміщення комп'ютеризованих робочих місць в приміщенні проводимо з урахуванням наступних вимог:

- робочі місця з ВДТ розміщуються на відстані не менше 1 м від стіни зі світловими прорізами;
- відстань між бічними поверхнями ВДТ має бути не менше 1,2 м;
- прохід між рядами робочих місць має бути не менше 1м.

Враховано розміри меблів на комп'ютеризованих робочих місцях, зокрема робочого столу. Відповідно до НПАОП 0.00-7.15-18 рекомендовані

розміри столу для робочого місця з ВДТ представляють: висота - 725 мм, ширина - 600-1400 мм, глибина - 800-1000 мм.

Приймаємо, що робочий стіл має такі розміри: ширина - 1200 мм, глибина - 800 мм.

Комп'ютеризовані робочі місця розміщені рядами уздовж стіни з вікнами. Це дає можливість зробити неможливим дзеркальне відбиття на екрані ВДТ джерел природного світла (вікон) і попадання їх у поле зору операторів, що погіршує умови їх зорової роботи. Конструкція робочого місця і взаємне розташування всіх його елементів (сидіння, органи управління, засоби відображення інформації тощо) відповідає антропометричним, фізіологічним і психологічним вимогам, а також характеру роботи [34].

Робоче місце з ПК обладнано відповідно до вимог ДСТУ ISO 13406-2:2006. «Ергономічні вимоги до роботи з відеотерміналами з плоским екраном. Частина 2. Ергономічні вимоги до дисплеїв з плоским екраном» [35].

Робоча поверхня встановлена з урахуванням характеру виконуваної роботи і має прямокутну форму. Висота сидіння і підставки для ніг (при нерегульованій висоті робочої поверхні). В цьому випадку висоту робочої поверхні встановлюють по номограмі залежності висоти робочої поверхні для різних видів робіт, простору для ніг і висоти робочого сидіння від росту людини (рисунок 1) для робочого зі зростом 1800 мм.

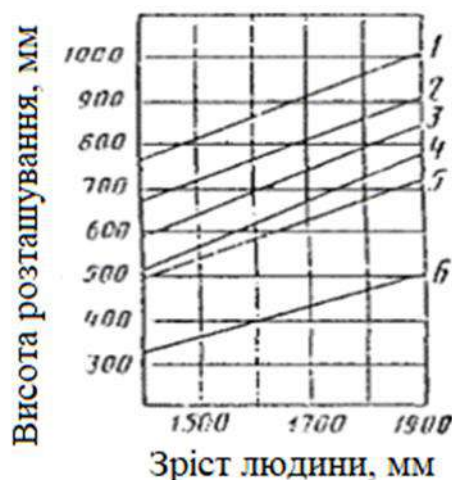


Рисунок 7.1 - Номограма

Оптимальна робоча поза для робітників більш низького зросту досягається за рахунок збільшення висоти робочого сидіння і підставки для ніг на величину, рівну різниці між висотою робочої поверхні для робітника зі зростом 1800 мм і висотою робочої поверхні, оптимальної для зросту даного робітника [34].

7.3 Заходи щодо забезпечення виробничої санітарії та гігієни праці

Освітленість робочих місць відповідає вимогам ДБН В.2.5-28-2018 «Природне та штучне освітлення» [36].

Приміщення для роботи з ВДТ мають природне і штучне освітлення відповідно до ДБН В.2.5-28-2018.

Природне освітлення здійснюється через світлові прорізи, орієнтовані переважно на північ / північний схід і забезпечують коефіцієнт природного освітлення (КПО) вище 1,5%.

Штучне освітлення здійснюється системою загального рівномірного освітлення з додатковим використанням світильників місцевого освітлення.

Як джерела світла для штучного освітлення використовуються переважно люмінесцентні лампи типу ЛБ і світильники типу ЛВП. У світильниках місцевого освітлення використовуються лампи розжарювання.

Віконні прольоти приміщень для роботи з ВДТ обладнані регульованими жалюзі і фіранками.

Розрахуємо систему загального рівномірного штучного освітлення в офісному приміщенні [37].

Приймаємо розміри приміщення довжина $A = 15$ м, ширина $B = 9$ м, висота $H = 4$ м. Висота робочої поверхні складає $h = 0,7$ м. Коефіцієнти відбиття поверхонь приміщення: $\rho_c = 70\%$; $\rho_{ст} = 50\%$; $\rho_{п} = 30\%$.

Згідно з ДБН В.2.5-28:2018. рівень нормованого загального освітлення в приміщенні нормується як $E_n = 300$ лк.

Обираємо світильник типу ЛВП, призначений для приміщень висотою до 4,5 м. IP=54 – ступінь захисту світильника від пилу та бризок води, $L/h = 1,3$ – числове значення коефіцієнта світильника.

Коефіцієнт запасу k_z враховує зниження рівня освітленості з часом в результаті забруднення та старіння ламп, світильників і поверхонь приміщення, приймається в залежності від виробничих умов. Приймаємо $k_z = 1,4$.

Коефіцієнт нерівномірності (мінімального) освітлення z (відношення середньої освітленості до мінімальної освітленості), для люмінесцентних ламп низького тиску, як правило дорівнює $z = 1,1$.

Розрахуємо кількість рядів світильників у приміщенні:

$$N_p = \frac{B}{(H-h_p) \cdot [L/h]} = \frac{9}{(4-0,7) \cdot 1,3} = 2,1 \approx 3 \text{ ряди світильників.} \quad (7.1)$$

Визначимо максимально припустиму відстань між рядами світильників:

$$L_{max} = \frac{B}{N_p}, \quad (7.2)$$

де B – ширина приміщення;

N_p – кількість рядів.

Розрахуємо:

$$L_{max} = \frac{9}{3} = 3 \text{ м.}$$

Розраховуємо висоту підвісу світильника над робочою поверхнею:

$$h = \frac{L_{max}}{[L/h]}. \quad (7.3)$$

Розрахуємо:

$$h = \frac{3}{1,3} = 2,3 \text{ м.}$$

Знайдемо висоту звисання світильника від стелі:

$$h_3 = H - h_p - h, \quad (7.4)$$

де H – висота приміщення;

h_p – висота робочої поверхні;

h – висота підвісу світильника над робочою поверхнею.

Розрахуємо:

$$h_3 = 4 - 0,7 - 2,3 = 1 \text{ м.}$$

Чисельне значення індексу приміщення визначаємо за рівнянням:

$$i = \frac{AB}{h(A+B)}, \quad (7.5)$$

де A – довжина приміщення, м;

B – ширина приміщення, м;

h – висота розміщення світильників над робочою поверхнею, м.

Розрахуємо:

$$i = \frac{15 * 9}{2,3 * (15 + 9)} = 2,44$$

Значення коефіцієнта використання світлового потоку η вибирається в залежності від виду джерела світла, типу обраного світильника, коефіцієнтів відбиття поверхонь приміщення (ρ_c , $\rho_{ст}$, $\rho_{п}$) та індексу приміщення. В нашому випадку $\eta = 58 \%$.

Визначаємо сумарний світловий потік освітлювальної установки у даному виробничому приміщенні:

$$\Phi_{\Sigma} = \frac{E_n * S * k_z * z}{\eta}, \quad (7.6)$$

де Φ_{Σ} – розрахункове значення сумарного світлового потоку у приміщенні, лк;

E_n – нормоване значення освітленості, лк;

S – площа освітлюваної поверхні, м²;

k_z – коефіцієнт запасу;

z – коефіцієнт нерівномірності (мінімальної) освітленості;

η – коефіцієнт використання світлового потоку.

Розрахуємо:

$$\Phi_{\Sigma} = \frac{300 * 15 * 9 * 1,4 * 1,1}{0,58} = 107535 \text{ лм},$$

Визначаємо умовну загальну кількість світильників у приміщенні:

$$N^* = \frac{AB}{L_{max}^2}. \quad (7.7)$$

Розрахуємо:

$$N^* = \frac{15 * 9}{3 * 3} = 15 \approx 15 \text{ шт.}$$

Кратне числу рядів $N_p = 3$.

Розрахуємо світловий потік умовного джерела світла:

$$\Phi_{\text{л}}^* = \frac{\Phi_{\Sigma}}{N_{\text{л}}}, \text{ лм}, \quad (7.8)$$

де Φ_{Σ} – розрахункове значення сумарного світлового потоку у приміщенні, лм;

$N_{\text{л}}$ – загальна кількість ламп у приміщенні,

Розрахуємо:

$$N_{\text{л}} = N * n, \quad (7.9)$$

де n – кількість ламп у світильнику (в ЛВП світильнику знаходиться 2 лампи).

Розрахуємо:

$$N_{\text{л}} = 15 * 2 = 30 \text{ шт.},$$

$$\Phi_{\text{л}} = \frac{107535}{30} = 3585 \text{ лм.}$$

Обираємо тип стандартної лампи з найближчим значенням фактичного світлового потоку лампи $\Phi_{\text{л}}$ і знайдемо коефіцієнт m (співвідношення між розрахунковим світловим потоком лампи $\Phi_{\text{л}}^*$ та фактичним світловим потоком вибраної стандартної лампи $\Phi_{\text{л}}$):

Обираємо лампу ЛДЦ із $P_{\text{л}}=80$ Вт, $\Phi_{\text{л}} = 3800$ лм

$$m = \frac{\Phi_{\text{л}}^*}{\Phi_{\text{л}}}. \quad (7.10)$$

Розрахуємо:

$$m = \frac{3585}{3800} = 0,94$$

Визначаємо оптимальну (фактичну) кількість світильників у приміщенні:

$$N = N * m. \quad (7.11)$$

Розрахуємо:

$$N = 15 * 0,94 \approx 15 \text{ шт.}$$

Фактична кількість ламп у приміщенні:

$$N_{\phi\lambda} = N_{\phi} \cdot n. \quad (7.12)$$

Розрахуємо:

$$N_{\phi\lambda} = 15 * 2 = 30 \text{ шт.}$$

Визначаємо загальну розрахункову освітленість E_p у приміщенні, що створюється при застосуванні стандартних ламп:

$$E_p = \frac{\Phi_{\lambda} N_{\phi\lambda} \eta}{Sk_3 z} \text{ лк.} \quad (7.13)$$

Розрахуємо:

$$E_p = \frac{3800 * 30 * 0,5}{15 * 9 * 1,4 * 1,1} = 274 \text{ лк.}$$

Тобто виконується умова $E_p = (-10\% \dots +20\%) \cdot E_n, \text{ лк.}$

Метеорологічні умови в виробничих приміщеннях – температура повітря, відносна вологість повітря й швидкість його переміщення відповідають вимогам Державних санітарних норм 3.3.6.042-99 «Державні санітарні норми мікроклімату виробничих приміщень» [39].

Роботи в офісному приміщенні, належать до категорії Іб - легка робота, тому передбачені наступні оптимальні значення параметрів мікроклімату:

– у холодний період року: температура 22-24°C; відносна вологість: 40-60%; швидкість переміщення повітря: 0,1 м/с

– у теплий період року: температура 23-25°C; відносна вологість: 40-60%; швидкість переміщення повітря: 0,2 м/с.

Режим праці і відпочинку при роботі на ЕОМ з ВТД відповідає НПАОП 0.00-7.15-18. Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями для операторів ЕОМ за характером роботи, становить при 8 годинній робочій зміні– 15 хвилинні перерви для відпочинку кожні 2 години, при 12 годинній робочій зміні – протягом останніх 4 годин, кожен годину перерви протягом 15 хвилин. Задля уникнення надмірної втоми виконується активний відпочинок – виконання гімнастичних вправ, що сприяють м'язовому розслабленню і зняття нервової напруги [30].

7.4 Заходи безпеки у надзвичайних ситуаціях

7.4.1 Заходи пожежної безпеки

Пожежі в обчислювальному центрі становлять особливу небезпеку, тому що пов'язані з великими матеріальними втратами. Як відомо, пожежа може виникнути при взаємодії горючих речовин, окислення і джерел запалювання. У приміщеннях ВЦ присутні всі три основні чинники, необхідні для виникнення пожежі.

Для відводу теплоти від ПК у обчислювальному центрі постійно діє потужна система кондиціонування. Тому кисень, як окислювач процесів горіння, є в будь-якій точці обчислювального центру.

Обчислювальний центр, згідно ДСТУ Б В.1.1-36:2016 «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою» [40] відноситься до категорії «П-Па», а клас можливої пожежі, згідно ДБН В.1.1-7:2016 «Пожежна безпека об'єктів будівництва. Загальні вимоги» [41], визначається, як «А».

Оскільки приміщення (дослідницької лабораторії, конструкторського бюро, тощо) що обладнане ПК з ВДТ має площу 135 м², тому відповідно до вимог п. 3.8 розділу «Типові норми належності вогнегасників» ДСТУ 4297:2004 «Пожежна техніка. Технічне обслуговування вогнегасників. Загальні технічні вимоги» [34] для гасіння електроустановок, що знаходяться під напругою, передбачені вуглекислотні вогнегасники типу ВВК-3,5 у кількості 7 штук (з розрахунку один вогнегасник с величиною заряду вогнегасної речовини 3,5 кг і більше, на 20 м² площі приміщення).

Усі працівники обчислювального центру під час прийняття на роботу і в процесі праці повинні проходити протипожежний інструктаж, перевірку знань з питань пожежної безпеки. Евакуаційні виходи і проходи утримуються постійно вільними та нічим не захаращуються.

Відповідальний за протипожежний стан приміщень обчислювального центру повинен періодично (не рідше одного разу на місяць) перевіряти працездатність пристрою, який забезпечує автоматичне вимкнення системи вентиляції у разі пожежі, а також вогнедимозатримувальних пристроїв. Під час ремонту слід не допускати руйнування або порушення цілісності негорючих діафрагм у фальш-підлоговому просторі.

Фільтри припливно-витяжної вентиляції слід чистити згідно з затвердженим графіком, але не рідше двох разів на рік. Роботи з ремонту вузлів (блоків) ЕОМ, як правило, проводити не дозволяється. Їх необхідно проводити в окремому приміщенні (майстерні). У разі необхідності проведення ремонту або технічного обслуговування ЕОМ безпосередньо в залі допускається мати не більше 0,5 л легкозаймистих рідин у небиткій, щільно закритій тарі. Для промивання деталей необхідно застосовувати, як правило, негорючі миючі препарати. Чарунки та інші знімні пристрої допускається промивати горючими рідинами тільки в спеціальному приміщенні, обладнаному припливно-витяжною вентиляцією.

Відповідальний за пожежну безпеку приміщень обчислювального центру зобов'язаний негайно сповістити технічну службу у разі виявлення

незадовільного технічного стану димових та теплових сповіщувачів автоматичної пожежної сигналізації, а також про несправності автоматичної установки об'ємного (газового) пожежогасіння [34].

Усі працівники обчислювального центру повинні вміти привести в дію (в разі потреби) первинні засоби пожежогасіння. Регулярно, але не рідше одного разу на квартал, слід очищати від пилу ЕОМ, контрольновимірну апаратуру, кабельні траншеї та фальшпідлоговий простір за допомогою пиловсмоктувача.

7.4.2 Заходи з цивільного захисту

Стійкість роботи об'єкта - здатність виконувати задані функції не тільки в нормальних, а й в надзвичайних ситуаціях, попереджати виникнення аварій і катастроф на об'єкті. Зокрема, об'єкти виробничої сфери повинні випускати продукцію в необхідному обсязі, номенклатурі, заданої якості та вартості, що забезпечує конкурентоспроможність на ринку. Стійка робота об'єкта неможлива без урахування стійкості самого об'єкта.

Під стійкістю об'єкта розуміють здатність його інженерно-технічного комплексу (будівель, споруд, обладнання, інженерних, комунально-енергетичних, транспортних та інших систем і комунікацій) протистояти руйнівній дії джерел надзвичайних ситуацій (НС) .

До комунально - енергетичних систем відносяться:

- водопостачання;
- теплопостачання;
- електропостачання;
- газопостачання;
- каналізація.

Усі вище перераховані системи інженерно-технічного комплексу не тільки повинні бути стійкі при впливі на них природних і техногенних НС, але і не повинні стати джерелами НС.

Забезпечення стійкості роботи систем водопостачання промислового об'єкта досягається:

- виконанням норм інженерно-технічних заходів при будівництві;
- проведенням заходів щодо підвищення фізичної стійкості, перш за все трансформаторних підстанцій, насосних станцій, очисних споруд і трубопроводів;
- своєчасним ремонтом і ревізією окремих ділянок систем водопостачання;
- використанням декількох незалежних джерел води;
- можливістю подачі води з одного водопроводу в інший;
- наявністю резервних джерел води;
- застосуванням автоматичних систем сигналізації при аваріях і автоматичних засувках;
- кільцюванням водопровідної мережі в межах міста для обходу пошкоджених ділянок водопроводу;
- високоефективної і надійної системою очисних споруд.

Забезпечення стійкості роботи системи теплопостачання промислового об'єкта досягається:

- виконанням норм інженерно-технічних заходів при будівництві;
- проведенням заходів щодо підвищення фізичної стійкості, перш за все котелень, насосних станцій, теплотрас і теплопроводів;
- прокладкою під землею газопроводів теплотрас;
- надійною роботою систем аварійного відключення ділянок теплотрас при аварійних ситуаціях;
- підвищенням надійності теплопроводів за рахунок кільцювання в межах міст і районів, де знаходяться об'єкти споживання;
- проведенням своєчасного профілактичного ремонту від корозії і засмічення.

Забезпечення стійкості роботи системи електропостачання промислового об'єкта досягається:

- виконанням норм інженерно-технічних заходів при будівництві;
- наявністю систем автоматики, аварійної сигналізації та захисту;
- проведення додаткових заходів по підвищенню стійкості наземних споруд (станції, підстанції, розподільчі пункти, трансформаторні станції тощо) та повітряних ліній електропередач;
- кільцюванням розподільчої мережі;
- дублюванням або резервування мереж електропостачання для об'єктів, аварії на яких в випадках відключення електропостачання можуть викликати надзвичайні ситуації;
- прокладкою електричних кабелів під землею в межах міст;
- заборона земляних робіт в містах без дозволу енергонагляду;
- своєчасним профілактичним ремонтом обладнання та його модернізацією;
- високим рівнем професійної підготовки обслуговуючого персоналу, його здатністю запобігати и діяти в питання надзвичайних ситуацій [41]

Забезпечення стійкості роботи систем газопостачання промислового об'єкта досягається:

- виконанням норм інженерно-технічних заходів при будівництві;
- проведенням додаткових заходів щодо підвищення фізичної стійкості будівель газорозподільних станцій і пунктів;
- прокладкою під землею газопроводів високого та середнього тиску;
- надійною роботою систем аварійного відключення ділянок газопроводу при аварійних ситуаціях;
- підвищенням надійності газопостачання об'єктів за рахунок кільцювання газопроводів в межах міст;
- подачею газу споживачеві не менше ніж через дві газорозподільні станції;
- впровадженням в диспетчерське управління і обслуговування газового господарства телемеханічних пристроїв і автоматики;

– високим рівнем професійної підготовки обслуговуючого персоналу, здатного запобігати або локалізувати аварійні ситуації.

Забезпечення стійкості роботи системи каналізації промислового об'єкта досягається:

- виконанням норм інженерно-технічних заходів при будівництві;
 - своєчасним очищенням колекторів та інших ділянок при закупорці;
 - роздільної системою каналізації за умови, що колектори обох частин системи з'єднані між собою, що дає можливість відключати окремі пошкоджені ділянки;
 - узгодженням з органами санітарного нагляду місць скидання стічних вод;
 - забезпеченням надійної роботи станцій перекачування;
- проведенням своєчасного профілактичного ремонту [42].

ВИСНОВКИ

Під час виконання магістерської роботи, було запропоновано новий метод аутентифікації на базі біометричних даних користувача, а саме зображення обличчя. Також було розроблено програмний комплекс, що реалізує роботу запропонованого методу.

Для цього були поставлені і вирішені такі задачі:

- ознайомлення та аналіз предметної області;
- розгляд мови програмування C++ та бібліотеки OpenCV;
- розроблення програмного комплексу для реалізації та тестування запропонованого методу.

Розроблений програмний комплекс, реалізує основні функції, а саме:

- фіксування поточного зображення;
- завантаження зображення до сервера зіставлення даних;
- зберігання еталонних зображень;
- порівняння зображень для аутентифікації;
- виведення результатів;
- надання та забезпечення графічного інтерфейсу користувача.

Запропонований метод значно прискорює час роботи.

Розроблений програмний комплекс має практичну спрямованість та готовий до використання.

Під час розробки програми та аналізу розробленого програмного комплексу з'явилися ідеї з удосконалення, такі як: використання математичного апарату штучних нейронних мереж для розпізнавання зображень та більш точного використання.

Нові можливості зможуть розширити функціонал програмного комплексу та підвищити його продуктивність.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Petrenko A. Designing security of personal data in distributed health care platform [Text] / A. Petrenko, R. Kyslyi, I. Pysmennyi // Technology audit and production reserves. – 2018. – № 4(2). – С. 10-15.
2. Kosar N. Problems of protection consumers personal data by domestic enterprises: aspects of social and ethical marketing [Text] / N. Kosar, N. Kuzo, A. Kyrylenko // Маркетинг і цифрові технології. – 2019. – Т. 3, № 1. – С. 78-91.
3. Jikia M. Correlation between the principles of publicity of Constitutional Court decisions and personal data protection (case study of Georgian Constitutional Court) [Text] / M. Jikia, S. Demetrashvili // Проблеми законності. – 2019. – Вип. 147. – С. 244-251.
4. Lavrenchuk S. Modern trends and methodology of personal data protection by Raspberry PI means [Text] / S. Lavrenchuk, S. Kostiuchko, A. Vozniak, A. Bulik // Комп'ютерно-інтегровані технології: освіта, наука, виробництво. – 2019. – № 35. – С. 141-145.
5. Khlaponin D. Yu. Mechanisms of public management in the personal data protection in smart cities [Text] / D. Yu. Khlaponin // Transfer of innovative technologies. – 2020. – Vol. 3, № 1. – С. 93-95.
6. Bahniuk N. Web service authorization for local network users [Text] / N. Bahniuk, P. Mykhailutsa, A. Khrystinets // Комп'ютерно-інтегровані технології: освіта, наука, виробництво. – 2018. – № 30-31. – С. 5-9.
7. Surkov S. S. Comparison of Authorization Protocols for Large Requests in the Operation Queue Environment [Text] / S. S. Surkov // Herald of advanced information technology. – 2020. – Vol. 3, no 3. – С. 163–173.
8. Matselyukh I. Legitimation through authorization and moral evaluation in discourse analysis of RT news [Text] / I. Matselyukh // Молодий вчений. – 2019. – № 10(1). – С. 176-182.

9. Polozhaenko S. A. Methods of mathematical design and machine authentication of anomalous diffusive processes [Text] / S. A. Polozhaenko // *Электротехнические и компьютерные системы*. – 2013. – Вип. 11. – С. 150- 156.
10. Juszczuk M. Various methods for employees' authentication at SMEs [Text] / M. Juszczuk, M. Milosz, E. Milosz // *Актуальні проблеми економіки*. – 2015. – № 2. – С. 474-481.
11. Narimanova O. V. Development of anti-counterfeit consumer product authentication system [Text] / O. V. Narimanova // *Праці Одеського політехнічного університету*. – 2015. – Вип. 2. – С. 124-127.
12. Evseev S. P. International legislation on personal data protection [Text] / S. P. Evseev // *Системи обробки інформації*. – 2012. – Вип. 9. – С. 140-144.
13. Rizak M. Rights and Obligations of Third Parties in Legal Relationships Regarding Circulation and Processing of Personal Data in Ukraine [Text] / M. Rizak // *Віче*. – 2014. – № 14. – С. 24-27.
14. Лохман Н. В. Аутентифікація понятійно-категоріального апарату системи інноваційного потенціалу [Текст] / Н. В. Лохман // *Науковий вісник Ужгородського національного університету. Серія : Міжнародні економічні відносини та світове господарство*. – 2018. – Вип. 21(1). – С. 137-141.
15. Брижко В. Захист персональних даних в Україні [Текст] / В. Брижко // *Вісник Академії управління МВС*. – 2007. – № 1. – С. 85-97.
16. Єнгаличев С. О. Біометрична аутентифікація на основі аналізу клавіатурного почерку [Текст] / С. О. Єнгаличев, С. Г. Семенов // *Прикладная радиоэлектроника*. – 2012. – Т. 11, № 2. – С. 309-311.
17. Mukhina M. Unified Template of Georeferenced Images for Visual Correlation-Extreme Navigation System [Text] / M. Mukhina // *Proceedings of the National Aviation University*. – 2015. – № 4. – С. 42-48.
18. Kotvytskiy A. T. Correlation of the number of images of an N-point gravitational lens and the number of solutions of its system [Text] / A. T. Kotvytskiy, S. D. Bronza, V. Yu. Shablenko // *Odessa astronomical publications*. – 2017. – Vol. 30. – С. 35-37.

19. Yeriomina N. Formation of reference images and decision function in radiometric correlation-extremal navigation systems [Text] / N. Yeriomina, S. Petrov, A. Tantsiura, M. Iasechko, V. Larin // Восточно-Европейский журнал передовых технологий. – 2018. – № 4(9). – С. 27-35.
20. Базурін В. М. Особливості навчання веб-програмування мовою JavaScript студентів-математиків [Текст] / В. М. Базурін // Вісник Житомирського державного університету імені Івана Франка . – 2014. – Вип. 1. – С. 79-83.
21. Ragimova I. JavaScript security using cryptographic hash functions [Text] / I. Ragimova, F. Gubadova, B. Asker-zade, S. Pohasii // Сучасні інформаційні системи. – 2019. – Т. 3, № 4. – С. 105-108.
22. Щербакова М. Є. Функціональні особливості JavaScript-додатків [Текст] / М. Є. Щербакова, Є. В. Щербаков // Вісник Східноукраїнського національного університету імені Володимира Даля. – 2014. – № 10. – С. 142-146.
23. Здоренко В. Г. Использование javascript-сценариев при создании автоматизированных информационных систем управления [Текст] / В. Г. Здоренко, Л. П. Голубев. // Технології та дизайн. – 2016. – № 2.
24. Stroustrup B. Programming: Principles and Practice Using C++ (2nd Edition) [Text] / B. Stroustrup. – С.: Addison-Wesley Professional, 2015. – 1312 p.
25. Lippman S.B., Lajoie J., Moo B.E. C++ Primer (5th Edition) [Text] / S.B. Lippman, J. Lajoie, B.E. Moo. – С.: Addison-Wesley Professional, 2012. – 976 p.
26. Ших Н. В. Аналіз роботи фронтальних класифікаторів бібліотеки OpenCV / Н. В. Ших, І. О. Шаклеїна // Комп'ютерно-інтегровані технології: освіта, наука, виробництво. – 2019. – № 35. – С. 107-111.
27. Системи адаптації роботів і технологія OpenCV [Текст] : навч. посіб. / О. М. Цимбал, А. І. Бронніков; Харків. нац. ун-т радіоелектроніки. – Харків : ХНУРЕ, 2019. – 144 с.
28. Microsoft Visual Studio [Electronic resource]. – Access mode: <https://visualstudio.microsoft.com/>.

29. Методичні вказівки по економічному обґрунтуванню дипломних проектів для студентів спеціальностей 7.080403 «Програмне забезпечення автоматизованих систем», 7.080402 «Інформаційні технології проектування» і 7.091501 «Комп'ютерні та інтелектуальні системи і мережі» всіх форм навчання для усіх форм навчання / Укл.: Остапенко В.В., Касьян Е.М., Сердюк Є.М. – Запоріжжя: ЗНТУ, 2009. – 22 с.

30. НПАОП 0.00-7.15-18. Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями. Державні санітарні правила і норми. – Харків: Форт, 2018 – 48 с.

31. Правила улаштування електроустановок [Текст] : ПУЕ-2017. – На заміну ПУЕ-86 ; чинний з 2017-08-21. – К. : Міненерговугілля України, 2017. – 617 с.

32. ДСТУ 7237:2011 «Система стандартів безпеки праці. Електробезпека. Загальні вимоги та номенклатура видів захисту». – Київ: Держспоживстандарт України, 2011. – 10 с.

33. ДСТУ Б В.1.1-36:2016 «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою». – Київ : Мінрегіонбуд України, 2016. – 66 с.

34. Основи професійної безпеки та здоров'я людини : підручник / В. В. Березуцький [та ін.] ; під ред. проф. В. В. Березуцького. – Харків : НТУ “ХПІ”, 2018. – 553 с

35. ДСТУ ISO 13406-2:2006. Ергономічні вимоги до роботи з відеотерміналами з плоским екраном. Частина 2. Ергономічні вимоги до дисплеїв з плоским екраном / Нац. Стандарт України. Вид. офіц. [Уведено вперше; чинний від 2007-10-01]. – Київ : ДП «УкрНДНЦ», 2007. 117 с.

36. ДБН В.2.5-28:2018. Природне та штучне освітлення. Посібник: проблеми природного і штучного освітлення. – Київ: Етін, 2019. – 180 с.

37. Методичні вказівки до лабораторного заняття «Розрахунок загального рівномірного штучного освітлення виробничих приміщень» з дисципліни «Цивільний захист і охорона праці в галузі»: для студентів усіх

спеціальностей та форм навчання /Укл. : В.І. Шмирко, О.В. Коробко, Ю.І. Троян. – Запоріжжя: каф. ОПіНС. НУ «Запорізька політехніка», 2020. – 36с.

38. ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку. – Київ: Міністерство охорони здоров'я України головне санітарно-епідеміологічне управління, 1999 – 24 с.

39. ДСН 3.3.6.042-99 Санітарні норми мікроклімату виробничих приміщень. – Харків: Форт, 2000 – 24 с.

40. ДСТУ Б В.1.1-36:2016 «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою». – Київ: Мінрегіонбуд України, 2016. – 66 с.

41. ДБН В.1.1-7:2016 «Пожежна безпека об'єктів будівництва. Загальні вимоги». – Київ: Міністерство регіонального розвитку, будівництва та житлово-комунального господарства України, 2017. – 39 с.

42. Шоботов В. М. Цивільна оборона [Текст] : Навчальний посібник / В. М. Шоботов. – Вид. 2-ге, перероб. – К. : Центр навчальної літератури, 2006. – 438 с.

ДОДАТОК А
Технічне завдання

A.1 Вступ

A.1.2 Призначення та область застосування

Розроблений програмний комплекс може для реалізації аутентифікації користувачів за їх біометричними даними. Така система може використовуватися у державних та банківських установах для підвищення рівня безпеки та захищеності особистих даних

Програма представляє інтерфейс для зручної роботи реалізованими функціями.

A.2 Підстави для розробки

Підставою для розробки є завдання на магістерську роботу.

Сьогодні в світі спостерігається зростаючий інтерес до проблеми машинного розпізнавання осіб. Системи розпізнавання осіб впроваджуються в сучасні смартфони, беруться на озброєння владою деяких держав для упіймання злочинців.

Використання таких систем значно підвищує рівень безпеки систем.

A.3 Призначення розробки

Розроблений програмний комплекс може використовуватися для розпізнавання та розширення доступу до ресурсів web-застосунку авторизованим користувачам.

Розроблений програмний комплекс, має реалізовувати основні функції, а саме:

- фіксування поточного зображення;
- завантаження зображення до сервера зіставлення даних;
- зберігання еталонних зображень;
- порівняння зображень для аутентифікації;

- виведення результатів;
- надання та забезпечення графічного інтерфейсу користувача.

A.4 Вимоги до програми

A.4.1 Вимоги до функціональних характеристик

ПЗ повинне забезпечувати виконання нижче перерахованих функцій:

- фіксування поточного зображення;
- завантаження зображення до сервера зіставлення даних;
- зберігання еталонних зображень;
- порівняння зображень для аутентифікації;
- виведення результатів;
- надання та забезпечення графічного інтерфейсу користувача.

A.4.2 Вимоги до надійності

Для забезпечення надійності програми, вона має базуватися на комп'ютері. На протязі усього сеансу роботи додаток не повинен збоїти або зависати. Після оновлення інформаційної бази або функціоналу, додаток треба перезапустити.

A.4.3 Вимоги до технічних та програмних засобів

Для нормальної розробки програми, повинні виконуватися наступні умови до апаратної частини:

- наявність IBM-сумісної персональної електронної обчислювальної машини (ПЕОМ), на якій буде працювати додаток;
- процесор із тактовою частотою 1,5 GHz та більше;
- оперативна пам'ять 1,5 ГБ та більше;

– вільний простір на жорсткому диску не менше, ніж 2 ГБ (для збереження зображень).

Вимоги до програмної частини:

– завантажена бібліотека OpenCV.

A.5 Вимоги до програмної документації

Склад програмної документації повинен включати в себе:

- технічне завдання;
- опис проектування;
- опис тестування;
- опис програми.

A.6 Стадії і етапи розробки

A.6.1 Стадії розробки

Розробка повинна бути проведена в три стадії:

- розробка технічного завдання;
- робоче проектування;
- впровадження.

A.6.2 Етапи розробки

На стадії розробки технічного завдання повинен бути виконаний етап розробки, погодження та затвердження справжнього технічного завдання.

На стадії робочого проектування повинні бути виконані перераховані нижче етапи робіт:

- розробка програми;
- розробка програмної документації;
- випробування програми.

На стадії впровадження повинен бути виконаний етап розробки підготовка і передача програми.

A.6.3 Зміст робіт по етапах

На етапі розробки технічного завдання повинні бути виконані перераховані нижче роботи:

- постановка задачі;
- визначення і уточнення вимог до технічних засобів;
- визначення вимог до програми;
- визначення стадій, етапів і термінів розробки програми та документації на неї;
- узгодження і затвердження технічного завдання.

На етапі розробки програми повинна бути виконана робота з програмування (кодування) і налагодженні програми.

На етапі розробки програмної документації повинна бути виконана розробка програмних документів відповідно до вимог до складу документації.

На етапі випробувань програми повинні бути виконані перераховані нижче види робіт:

- розробка, узгодження та затвердження та методики випробувань;
- проведення приймально-здавальних випробувань;
- коректування програми і програмної документації за результатами випробувань.

ДОДАТОК Б
Опис програми

Б.1 Загальні свідчення

ПЗ розроблений за допомогою мови програмування C++ та бібліотеки OpenCV. Тобто реалізації бізнес логіки програми використовується мова програмування C++.

Б.2 Функціональне призначення

Розроблений програмний комплекс може використовуватися для розпізнавання та розширення доступу до ресурсів web-застосунку авторизованим користувачам.

Розроблений програмний комплекс, реалізує основні функції, а саме:

- фіксування поточного зображення;
- завантаження зображення до сервера зіставлення даних;
- зберігання еталонних зображень;
- порівняння зображень для аутентифікації;
- виведення результатів;
- надання та забезпечення графічного інтерфейсу користувача.

Б.3 Опис логічної структури

На структурній схемі (рис. Б.1) основні програмні модулі, що були виділені при проектуванні системи.

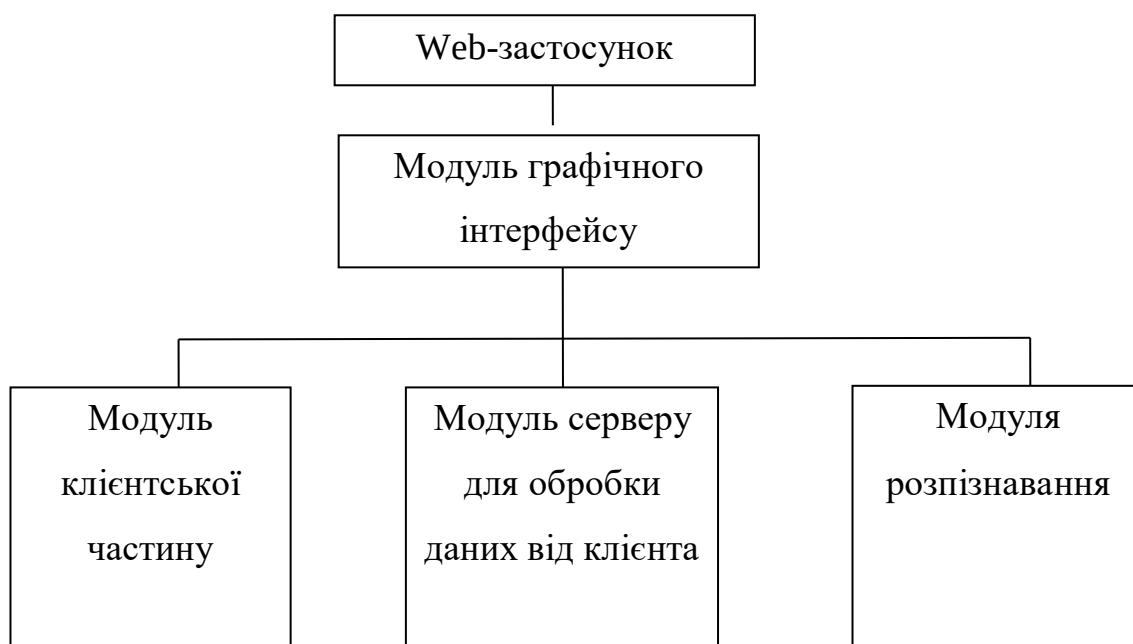


Рисунок Б.1 – Структурна схема проекту

Розробку представлених модулів можна аргументувати тим, що саме розділення основного функціоналу програмного комплексу за таким принципом гарантує формування функціонально завершених блоків програми.

Для того щоб почати розробку модуля розпізнавання необхідно підключити бібліотеку OpenCV. У даній бібліотеці містяться алгоритми і методи для обробки зображень.

Першим кроком необхідно викачати дистрибутив бібліотеки з сайту opencv.org. далі необхідно витягти вміст бібліотеки в зазначену директорію. Створюємо системну змінну, де вказуємо шлях до папки з opencv. Запускаємо visual studio і створюємо порожній C++ проект. Відкриваємо властивості проекту. Знаходимо вкладку Каталоги VC++. У рядку «Додаткові каталоги включаються файлів» вказуємо на «opencv\build\include». А в рядку «Каталоги бібліотек» вказуємо на самі ліби.

У загальних властивостях компоувальника необхідно включити інкрементну компоновку і вказати, що необхідно використовувати вхідні файли залежностей бібліотек.

Для debug і release конфігурацій необхідно підключити потрібні бібліотеки, які будуть використовуватися в проекті. Ліба для debug конфігурації має в назві додаткову букву d здійснюється процес підключення бібліотек для різних конфігурацій.

Б.4 Використані технічні засоби

Для нормального функціонування програми, повинні виконуватися наступні умови до апаратної частини:

- наявність IBM-сумісної персональної електронної обчислювальної машини (ПЕОМ), на якій буде працювати додаток;
- процесор із тактовою частотою 1,5 GHz та більше;
- оперативна пам'ять 1,5 ГБ та більше;
- вільний простір на жорсткому диску не менше, ніж 2 ГБ (для збереження зображень).

Вимоги до програмної частини:

- завантажена бібліотека OpenCV.

ДОДАТОК В
Текст програми

```

package.json
{
  "name": "fr-mediator",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "build": "tsc",
    "start": "nodemon lib/index.js",
    "test": "echo \"Error: no test specified\" && exit 1",
    "watch": "npm run build -- -w"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "body-parser": "^1.18.3",
    "dotenv": "^6.1.0",
    "express": "^4.16.3",
    "mongodb": "^3.1.9",
    "mongoose": "^5.3.13",
    "morgan": "^1.9.1",
    "multer": "^1.4.1",
    "nodemon": "^1.18.4",
    "request": "^2.88.0",
    "request-promise-native": "^1.0.5",
    "serialport": "^7.0.2"
  },
  "devDependencies": {
    "@types/node": "^10.12.5",
    "typescript": "^3.1.6"
  }
}
tsconfig.json
{
  "compilerOptions": {
    "target": "es6",
    "module": "commonjs",
    "outDir": "lib",
    "strictNullChecks": true
  },
  "files": [
    "./node_modules/@types/node/index.d.ts"
  ],
  "include": [
    "src/**/*.ts"
  ],
  "exclude": [
    "node_modules"
  ]
}
src/api.ts
import { Face, Encoding, Name, FacesStorage } from './core';
import { FaceModel } from './db';
const request = require('request');
const requestPromise = require('request-promise-native');

const { RECOGNIZER_URL } = process.env;

export function registerFace(face: Face): Promise<void> {
  return FaceModel.create(face);
}

```

```

export function encode(file, callback) {
  console.log(`${RECOGNIZER_URL}/encode`);
  const options = {
    url: `${RECOGNIZER_URL}/encode`,
    method: 'POST',
    headers: {
      'Content-Type': 'multipart/form-data',
      accept: 'application/json',
    },
  };

  /* body: encoding */
  const r = request.post(options, callback);

  const form = r.form();
  form.append('file', file.buffer,
    { contentType: 'image/png', filename: file.originalname });
}

export function compare(encoding: Encoding, storage: FacesStorage, tolerance: number):
Promise<any> {
  const options = {
    url: `${RECOGNIZER_URL}/recognize`,
    method: 'POST',
    headers: {
      accept: 'application/json',
    },
    body: JSON.stringify({
      encoding,
      storage,
      tolerance,
    }),
    json: true,
  };

  return requestPromise.post(options);
}

export async function recognize(encoding: Encoding, tolerance: number = 0.1):
Promise<Name | null> {
  try {
    const storage: FacesStorage = await FaceModel.find({}).exec();
    const storageLength = storage.length;

    if (storageLength === 0) {
      return null;
    }

    return compare(encoding, storage, tolerance);
  } catch (e) {
    console.error('ERROR', e);
    throw (e);
  }
}

src/core.ts
export type Name = string;
export type Encoding = number[];
export type Face = {
  name: Name;
  encoding: Encoding;
};

```

```

export type FacesStorage = Face[];

export enum ApiResponse {
  NoFileUploaded = 'No file uploaded',
  NoFacesDetected = 'No faces detected',
  FaceNotRecognized = 'Face is not recognized',
  FaceRecognized = 'Face is recognized',
  InvalidFaceReceived = 'Invalid face received',
  FaceRegistered = 'Face is registered',
}
src/db.ts
import * as Mongoose from 'mongoose';
import { Face } from './core';

const {
  DB_USER,
  DB_PASS,
  DB_URL,
} = process.env;

Mongoose.connect(
  `mongodb://${DB_USER}:${DB_PASS}@${DB_URL}`,
  { useNewUrlParser: true },
  (err) => {
    if (err) {
      console.error('Error while connecting to the DB');
      console.error(err);
      return;
    }

    console.log('Connected to the DB');
  });

Mongoose.set('debug', (collectionName, method, query, doc) => {
  console.log(`${collectionName}.${method}`, JSON.stringify(query), doc);
});

const FaceSchema = {
  name: String,
  encoding: [Number],
}

export const FaceModel: Mongoose.Model<Face> = Mongoose.model('Face', FaceSchema);
src/index.ts
const express = require('express');
const morgan = require('morgan');
const bodyParser = require('body-parser');
require('dotenv').config();

import router from './router';

const PORT = 8081;
const HOST = '0.0.0.0';

const app = express();

app.use(morgan('combined'));
app.use(bodyParser.json());
app.use(router);

app.listen(PORT, HOST);
console.log(`Running on http://${HOST}:${PORT}`);

```

```

src/router.ts
const express = require('express');
const multer = require('multer');

import { ApiResponse, Encoding, Face } from './core';
import { recognize, registerFace, encode } from './api';
import { listSerialPorts } from './serialport';

const router = express.Router();
const upload = multer();

/* formData */
router.post('/recognize', upload.any(), function (req, res) {
  /**
   * Proxies request to a recognizer
   */
  if (!req.files) {
    res.send({ message: ApiResponse.NoFileUploaded });
  }

  const file = req.files[0];

  if (!file) {
    res.send({ message: ApiResponse.NoFileUploaded });
  }

  if (!isFileAllowed(file.originalname)) {
    res.send({ message: ApiResponse.NoFileUploaded });
  }

  encode(file, async (error, proxyRes, body) => {
    if (error) {
      console.log(error);
      console.error(ApiResponse.NoFacesDetected);
      res.send({ message: ApiResponse.NoFacesDetected });

      return;
    }

    if (body === ApiResponse.NoFacesDetected) {
      console.error(ApiResponse.NoFacesDetected);
      res.send({ message: ApiResponse.NoFacesDetected });

      return;
    }

    try {
      const encoding: Encoding = JSON.parse(body);
      const name = await recognize(encoding);

      if (name) {
        res.send({
          name,
          message: ApiResponse.FaceRecognized,
        });
      } else {
        console.error(ApiResponse.FaceNotRecognized);
        res.send({
          encoding,
          message: ApiResponse.FaceNotRecognized,
        });
      }
    }
  });
});

```

```

    } catch (e) {
      console.log(body);
      console.error(e);
      res.status(500).send(e, body);
    }
  });
});

/* { name, encoding } */
router.post('/register', function (req, res) {
  const face: Face = req.body;
  if (face.name == null || !Array.isArray(face.encoding)) {
    res.status(400).send({ message: ApiResponse.InvalidFaceReceived });
    return;
  }
  registerFace(face);

  res.send({ message: ApiResponse.FaceRegistered });
});

router.get('/serialports', async function (req, res) {
  const ports = await listSerialPorts();
  res.send(ports);
})

function isFileAllowed(fileName) {
  const allowedExtensions = ['png', 'jpg', 'jpeg', 'gif'];
  return (
    fileName.includes('.')
    &&
    allowedExtensions.indexOf(fileName.split('.')[1].toLowerCase()) !== -1
  );
}

export default router;
src/serialport.ts
const SerialPort = require('serialport');

export const listSerialPorts = () => new Promise((resolve, reject) => {
  SerialPort.list(function (err, ports) {
    if (err) {
      reject(err);
    }

    resolve(ports);
  });
});
package.json
{
  "name": "fr-client",
  "version": "0.1.0",
  "private": true,
  "proxy": "http://localhost:8081/",
  "dependencies": {
    "antd": "^3.10.8",
    "axios": "^0.17.1",
    "classnames": "^2.2.6",
    "prop-types": "^15.6.2",
    "react": "^16.6.3",
    "react-dom": "^16.6.3",
    "react-json-view": "^1.19.1",
    "react-jss": "^8.6.1",

```

```

    "react-router-dom": "^4.3.1",
    "react-scripts": "1.1.0"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test --env=jsdom",
    "eject": "react-scripts eject"
  }
}
src/index.js
import React from 'react';
import ReactDOM from 'react-dom';
import { App } from './components';
import registerServiceWorker from './registerServiceWorker';
import 'antd/dist/antd.css';

ReactDOM.render(<App />, document.getElementById('root'));
registerServiceWorker();
src/components/index.js
export { default as App } from './App';
src/components/App.js
import React, { Component } from 'react';
import injectSheet from 'react-jss';
import { BrowserRouter as Router, Route, Switch } from 'react-router-dom';
import Authorize from '../modules/Authentication/components/Authorize';
import Home from './Home';
import { AuthenticationProvider } from '../modules/Authentication';
import { ProtectedRoute } from '../modules/Authentication';

function App() {
  return (
    <AuthenticationProvider>
      <Router>
        <Switch>
          <ProtectedRoute path="/" exact component={Home} />
          <Route path="/authenticate/" component={Authorize} />
        </Switch>
      </Router>
    </AuthenticationProvider>
  );
};

export default App;
src/components/Home.js
import { Layout, Menu } from 'antd';
import React from 'react';
import injectSheet from 'react-jss';

import {
  withAuthentication,
  withAuthenticationPropTypes,
} from '../modules/Authentication';
import logo50 from '../images/logo50.png';
import { compose } from '../modules/util';
import { constants } from '../config';
import { SerialPortsList } from './SerialPortsList';

const { Header, Content, Footer } = Layout;

const styles = {
  img: {

```

```

    height: '50px',
    width: '50px',
    margin: '6px',
  },
  header: {
    display: 'flex',
    justifyContent: 'space-between',
  },
  menu: {
    lineHeight: '64px',
  },
  content: {
    padding: '50px',
    backgroundColor: '#fff',
  }
};

function Home(props) {
  const { classes } = props;

  const onMenuClick = ({ key }) => {
    if (key === 'logout') {
      props.authentication.setName(undefined);
    }
  }

  return (
    <Layout>
      <Header className={classes.header}>
        <img src={logo50} className={classes.img} alt="logo" />
        <Menu
          theme="dark"
          mode="horizontal"
          className={classes.menu}
          onClick={onMenuClick}
        >
          <Menu.Item key="logout">{constants.LOG_OUT}</Menu.Item>
        </Menu>
      </Header>
      <Content className={classes.content}>
        <SerialPortsList />
      </Content>
      <Footer />
    </Layout>
  );
}

Home.propTypes = {
  ...withAuthenticationPropTypes,
}

export default compose(
  withAuthentication,
  injectSheet(styles),
)(Home);
src/components/SerialPortsList.js
import React from 'react';
import ReactJson from 'react-json-view';

import { api, notification } from '../services';
import { constants } from '../config';

```

```

export class SerialPortsList extends React.Component {
  state = {
    list: [],
  };

  async componentDidMount() {
    try {
      const listResponse = await api.getSerialPortsList();
      this.setState({ list: listResponse.data })
    } catch (e) {
      console.error(e);
      notification.error(constants.FAILED_FETCHING_SERIAL_PORTS);
    }
  }

  render() {
    const { list } = this.state;

    return (
      <React.Fragment>
        <h1>{constants.SERIAL_PORTS}</h1>
        <ReactJson src={list} />
      </React.Fragment>
    )
  }
}
src/config/index.js
import * as constants from './constants';

export { constants };
src/config/constants.js
export const FACE_NOT_RECOGNIZED_RESPONSE = 'Face is not recognized';
export const FACE_RECOGNIZED_RESPONSE = 'Face is recognized';

export const nameIsRegistered = (name) => `${name} is registered`;
export const CAPTURE = 'Capture';
export const TRY_AGAIN = 'Try Again';
export const REGISTER_FACE = 'Register Face';
export const PLEASE_INPUT_YOUR_NAME = 'Please input your name!';
export const NAME = 'Name!';

export const FAILED_FETCHING_SERIAL_PORTS = 'Failed fetching serial ports';
export const SERIAL_PORTS = 'Serial Ports';
export const LOG_OUT = 'Log out';
src/services/index.js
import * as api from './api';
import * as notification from './notification';

export { api, storage, notification };
src/services/api.js
import axios from 'axios';

export const recognize = (data /* formData */) => axios({
  data,
  method: 'post',
  url: '/recognize',
  config: { headers: { 'Content-Type': 'multipart/form-data' } } },
});

export const register = (data /* { name, encoding } */) => axios({
  data,
  method: 'post',

```

```

    url: '/register',
  });

export const getSerialPortsList = () => axios('/serialports');
src/services/notification.js
import { message } from 'antd';

export const error = (text) => message.error(text);
export const success = (text) => message.success(text);
modules/util/compose.js
// https://github.com/reduxjs/redux/blob/master/src/compose.js

/**
 * Composes single-argument functions from right to left. The rightmost
 * function can take multiple arguments as it provides the signature for
 * the resulting composite function.
 *
 * @param {...Function} funcs The functions to compose.
 * @returns {Function} A function obtained by composing the argument functions
 * from right to left. For example, compose(f, g, h) is identical to doing
 * (...args) => f(g(h(...args))).
 */

export default function compose(...funcs) {
  if (funcs.length === 0) {
    return arg => arg
  }

  if (funcs.length === 1) {
    return funcs[0]
  }

  return funcs.reduce((a, b) => (...args) => a(b(...args)))
}
module/util/index.js
export { default as compose } from './compose';
module/Authentication/index.js
export {
  AuthenticationProvider,
  withAuthentication,
  withAuthenticationPropTypes,
} from './AuthenticationProvider';
export { default as ProtectedRoute } from './ProtectedRoute';
module/Authentication/AuthenticationProvider.js
import React from 'react';
import PropTypes from 'prop-types';

const AuthenticationContext = React.createContext();

export class AuthenticationProvider extends React.Component {
  constructor(props) {
    super(props);

    const localStorageName = localStorage.getItem('name');
    const localStorageNameIsDefined = localStorageName && localStorageName !==
'undefined';

    const name = localStorageNameIsDefined ? localStorageName : undefined;

    this.state = {
      name,
    };
  }

```

```

    }

    setName = (name) => {
      this.setState({ name });
      localStorage.setItem('name', name);
    }

    render() {
      const providerValue = {
        name: this.state.name,
        setName: this.setName,
      };

      return (
        <AuthenticationContext.Provider value={providerValue}>
          {this.props.children}
        </AuthenticationContext.Provider>
      );
    }
  }
}

export const withAuthentication = (Component) => {
  function WithAuthenticationHoc(props) {
    return (
      <AuthenticationContext.Consumer>
        {(context) => <Component {...props} authentication={context} />}
      </AuthenticationContext.Consumer>
    );
  }

  WithAuthenticationHoc.displayName = `WithAuthentication(${Component.displayName || Component.name})`;

  return WithAuthenticationHoc;
};

export const withAuthenticationPropTypes = {
  authentication: PropTypes.shape({
    name: PropTypes.string,
    setName: PropTypes.func.isRequired,
  }).isRequired,
};

module/Authentication/ProtectedRoute.js
import React from 'react';
import { Redirect, Route } from 'react-router-dom';
import { withAuthentication } from './AuthenticationProvider';

function PrivateRoute(
  {
    component: Component,
    authentication,
    ...rest
  }) {
  return (
    <Route
      {...rest}
      render={(props) =>
        authentication.name ? (
          <Component {...props} />
        ) : (
          <Redirect
            to={{

```

```

        pathname: "/authenticate",
        state: { from: props.location }
      }}
    />
  )
}
/>
);
}

export default withAuthentication(PrivateRoute);
module/Authentication/components/Authorize.js
import React, { Component } from 'react';
import injectSheet from 'react-jss';
import { Redirect } from 'react-router-dom';
import PropTypes from 'prop-types';

import Video from './Video';
import RegisterFaceForm from './RegisterFaceForm';

import {
  withAuthentication,
  withAuthenticationPropTypes,
} from '..';

import { compose } from '../../util';

const styles = {
  result: {
    width: '640px',
    margin: '8px auto',
    textAlign: 'center',
  },
};

class Authorize extends Component {
  static propTypes = {
    ...withAuthenticationPropTypes,
    classes: PropTypes.objectOf(PropTypes.string).isRequired,
  };

  state = {
    result: '',
    unrecognizedFaceEncoding: [],
    modalVisible: false,
    confirmLoading: false,
  };

  openRegisterModal = (unrecognizedFaceEncoding) => this.setState({
    unrecognizedFaceEncoding,
    modalVisible: true,
  });

  closeRegisterModal = () => this.setState({
    unrecognizedFaceEncoding: [],
    modalVisible: false,
  });

  onVideoResult = (result) => this.setState({ result });

  render() {
    const { classes } = this.props;

```

```

const {
  modalVisible,
  confirmLoading,
} = this.state;

if (this.props.authentication.name) {
  return (
    <Redirect to="/" />
  );
}

return (
  <React.Fragment>
    <Video
      onRegisterClick={this.openRegisterModal}
      onResult={this.onVideoResult}
    />

    <div className={classes.result}>
      {this.state.result}
    </div>

    <RegisterFaceForm
      modalVisible={modalVisible}
      confirmLoading={confirmLoading}
      closeModal={this.closeRegisterModal}
      unrecognizedFaceEncoding={this.state.unrecognizedFaceEncoding}
    />
  </React.Fragment>
);
}
}

export default compose(
  injectSheet(styles),
  withAuthentication,
)(Authorize);
module/Authentication/components/RegisterFaceForm.js
import React, { Component } from 'react';
import PropTypes from 'prop-types';
import { Form, Input, Modal } from 'antd';

import { constants } from '../../../config';
import { api, notification } from '../../../services';
import { compose } from '../../../util';
import {
  withAuthentication,
  withAuthenticationPropTypes
} from
'../AuthenticationProvider';

const FormItem = Form.Item;

class RegisterFaceForm extends Component {
  static propTypes = {
    modalVisible: PropTypes.bool.isRequired,
    confirmLoading: PropTypes.bool.isRequired,
    closeModal: PropTypes.func.isRequired,
    unrecognizedFaceEncoding: PropTypes.arrayOf(PropTypes.number).isRequired,
    ...withAuthenticationPropTypes,
  };

  setFormRef = (element) => {
    this.form = element;
  };

```

```

handleRegisterOk = (event, values) => {
  this.props.form.validateFields(async (err, values) => {
    if (err) {
      return;
    }

    this.setState({
      confirmLoading: true,
    });

    const { name } = values;
    const data = {
      name,
      encoding: this.props.unrecognizedFaceEncoding,
    };

    try {
      await api.register(data);

      notification.success(constants.nameIsRegistered(name));

      this.props.authentication.setName(name);
    } catch (e) {
      console.error(e.response);
      notification.error(e.message);

      this.props.closeModal();
    }
  });
}

render() {
  const { getFieldDecorator } = this.props.form;
  const { modalVisible, confirmLoading, closeModal } = this.props;
  return (
    <Modal title={constants.REGISTER_FACE}
      visible={modalVisible}
      onOk={this.handleRegisterOk}
      confirmLoading={confirmLoading}
      onCancel={closeModal}
    >
      <Form>
        <FormItem>
          {getFieldDecorator('name', {
            rules: [{ required: true, message: constants.PLEASE_INPUT_YOUR_NAME }],
          })(
            <Input placeholder={constants.NAME} />
          )}
        </FormItem>
      </Form>
    </Modal>
  );
}

export default compose(
  Form.create(),
  withAuthentication,
)(RegisterFaceForm);
module/Authentication/components/Video.js
import React, { Component } from 'react';

```

```

import injectSheet from 'react-jss';
import classNames from 'classnames';
import PropTypes from 'prop-types';
import { constants } from '../../config';
import { api, notification } from '../../services';
import {
  withAuthentication,
  withAuthenticationPropTypes,
} from '../AuthenticationProvider';
import { compose } from '../../util';

const styles = {
  image: {
    width: '640px',
    height: '480px',
    margin: '8px auto',
  },
  video: {
    height: '100%',
    width: '100%',
  },
  canvas: {
    height: '100%',
    width: '100%',
  },
  hidden: {
    display: 'none',
  },
};

class Video extends Component {
  static propTypes = {
    ...withAuthenticationPropTypes,
    classes: PropTypes.objectOf(PropTypes.string).isRequired,
    onRegisterClick: PropTypes.func.isRequired,
    onResult: PropTypes.func.isRequired,
  };

  state = {
    result: '',
    unrecognizedFaceEncoding: [],
    videoHidden: false,
    canvasHidden: true,
    captureHidden: false,
    againHidden: true,
    registerHidden: true,
    spinHidden: true,
    confirmLoading: false,
  };

  async componentDidMount() {
    try {
      this.video.srcObject = await navigator.mediaDevices.getUserMedia({ video: true
    });
    } catch (e) {
      console.error(e);
    }
  }

  setVideoRef = (element) => {
    this.video = element;
  };

```

```

setCanvasRef = (element) => {
  this.canvas = element;
};

buildVideoClassNames = () => classNames({
  [this.props.classes.video]: true,
  [this.props.classes.hidden]: this.state.videoHidden,
});

buildCanvasClassNames = () => classNames({
  [this.props.classes.canvas]: true,
  [this.props.classes.hidden]: this.state.canvasHidden,
});

processRecognitionResponse = (response) => {
  const faceNotRecognized = response.data.message ===
constants.FACE_NOT_RECOGNIZED_RESPONSE;
  const faceRecognized = response.data.message ===
constants.FACE_RECOGNIZED_RESPONSE;

  const registerHidden = !faceNotRecognized;
  const result = faceRecognized ? undefined : response.data.message;
  const unrecognizedFaceEncoding = faceNotRecognized ? response.data.encoding : [];

  if (faceRecognized) {
    this.props.authentication.setName(response.data.name);
    return;
  }

  return { registerHidden, result, unrecognizedFaceEncoding };
}

handleCaptureClick = () => {
  this.setState({
    videoHidden: true,
    canvasHidden: false,
    captureHidden: true,
    spinHidden: false,
  });

  // Draw the video frame to the canvas.
  const context = this.canvas.getContext('2d');
  context.drawImage(this.video, 0, 0, this.canvas.width, this.canvas.height);

  this.canvas.toBlob(async (blob) => {
    const formData = new FormData();
    formData.append('file', blob, 'selfie.jpg');

    try {
      const response = await api.recognize(formData)
      const processedResponse = this.processRecognitionResponse(response);

      if (processedResponse) {
        this.props.onResult(processedResponse.result);
        this.setState({
          ...processedResponse,
          spinHidden: true,
          againHidden: false,
        });
      }
    } catch (e) {

```

```

        console.error(e);
        notification.error(e.message);

        this.setState({
            spinHidden: true,
            againHidden: false,
        });
    }
    });
};

handleTryAgainClick = () => {
    this.setState({
        result: '',
        videoHidden: false,
        canvasHidden: true,
        captureHidden: false,
        againHidden: true,
        registerHidden: true,
        unrecognizedFaceEncoding: [],
    });
};

handleRegisterClick = () => {
    this.props.onRegisterClick(this.state.unrecognizedFaceEncoding);
};

render() {
    const { classes } = this.props;
    const {
        captureHidden,
        againHidden,
        registerHidden,
        spinHidden
    } = this.state;

    return (
        <React.Fragment>
            <div className={classes.image}>
                <video
                    className={this.buildVideoClassNames()}
                    ref={this.setVideoRef}
                    autoPlay
                />
                <canvas
                    className={this.buildCanvasClassNames()}
                    ref={this.setCanvasRef}
                />
            </div>

            <VideoControls
                handleCaptureClick={this.handleCaptureClick}
                handleTryAgainClick={this.handleTryAgainClick}
                handleRegisterClick={this.handleRegisterClick}
                captureHidden={captureHidden}
                againHidden={againHidden}
                registerHidden={registerHidden}
                spinHidden={spinHidden}
            />
        </React.Fragment>
    );
}

```

```

};

export default compose(
  injectSheet(styles),
  withAuthentication,
)(Video);
module/Authentication/components/VideoControls.js
import React from 'react';
import { Button, Spin } from 'antd';
import classNames from 'classnames';
import injectSheet from 'react-jss';

import { constants } from '../../../config';

const styles = {
  actionButton: {
    margin: '0 4px',
  },
  hidden: {
    display: 'none',
  },
  actions: {
    width: '640px',
    margin: '8px auto',
    textAlign: 'center',
  },
}
const VideoControls = (
  {
    classes,
    handleCaptureClick,
    handleTryAgainClick,
    handleRegisterClick,
    captureHidden,
    againHidden,
    registerHidden,
    spinHidden,
  }
) => {
  const buildCaptureButtonClassNames = () => classNames({
    [classes.actionButton]: true,
    [classes.hidden]: captureHidden,
  });
  const buildTryAgainButtonClassNames = () => classNames({
    [classes.actionButton]: true,
    [classes.hidden]: againHidden,
  });

  const buildRegisterButtonClassNames = () => classNames({
    [classes.actionButton]: true,
    [classes.hidden]: registerHidden,
  });

  const buildSpinClassNames = () => classNames({
    [classes.hidden]: spinHidden,
  });

  return (
    <div className={classes.actions}>
      <Button
        type="primary"
        icon="search"

```

```

        className={buildCaptureButtonClassNames()}
        onClick={handleCaptureClick}
      >
        {constants.CAPTURE}
      </Button>
      <Button
        type="primary"
        icon="reload"
        className={buildTryAgainButtonClassNames()}
        onClick={handleTryAgainClick}
      >
        {constants.TRY_AGAIN}
      </Button>
      <Button
        type="primary"
        icon="cloud-upload-o"
        className={buildRegisterButtonClassNames()}
        onClick={handleRegisterClick}
      >
        {constants.REGISTER_FACE}
      </Button>
      <Spin className={buildSpinClassNames()} />
    </div>
  );
};
export default injectSheet(styles)(VideoControls);
public/index.html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <meta name="theme-color" content="#000000">
    <!--
      manifest.json provides metadata used when your web app is added to the
      homescreen on Android. See https://developers.google.com/web/fundamentals/engage-
      and-retain/web-app-manifest/
    -->
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json">
    <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.ico">
    <!--
      Notice the use of %PUBLIC_URL% in the tags above.
      It will be replaced with the URL of the `public` folder during the build.
      Only files inside the `public` folder can be referenced from the HTML.

      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
      work correctly both with client-side routing and a non-root public URL.
      Learn how to configure a non-root public URL by running `npm run build`.
    -->
    <title>React App</title>
  </head>
  <body>
    <noscript>
      You need to enable JavaScript to run this app.
    </noscript>
    <div id="root"></div>
  </body>

```

ДОДАТОК Г
Слайди презентації



Рисунок Г.1 – Слайд №1

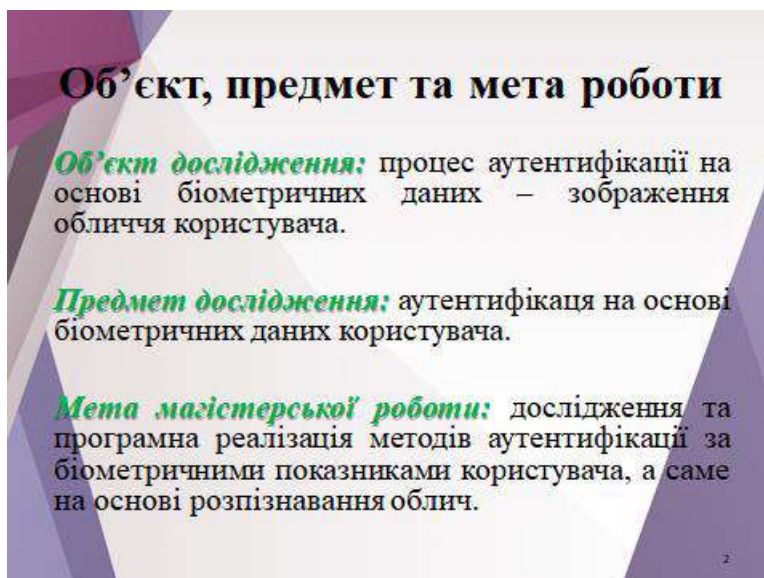


Рисунок Г.2 – Слайд №2

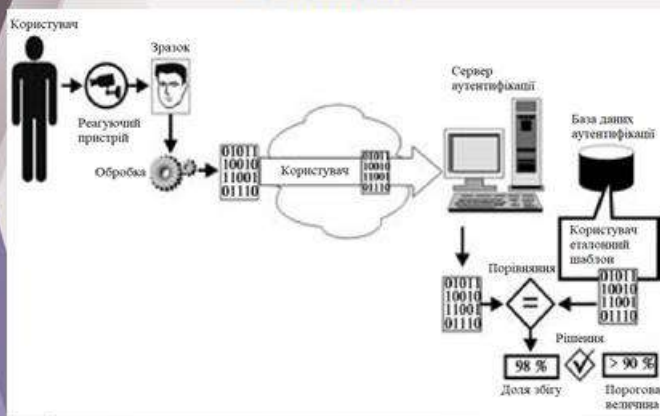
Постановка завдання

Головна мета магістерської роботи полягає у дослідженні, розробці та програмній реалізації методу аутентифікації на основі біометричних даних користувача, а саме на базі розпізнавання за біометричним ідентифікатором – двовірним зображення обличчя. Для роботи буде використовуватися метод швидкої нормалізованої взаємної кореляції та бібліотека функцій та алгоритмів комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення – OpenCV.

3

Рисунок Г.3 – Слайд №3

Схема роботи біометричної системи



4

Рисунок Г.4 – Слайд №4



Рисунок Г.5 – Слайд №5

Недоліки 2D методу аутентифікації і способи їх нівелювання

Недоліки	Спосіб усунення під час роботи
Низька статистична достовірність	Даний модуль аутентифікації буде встановлюватися на локальний ПК і призначений для роботи обмеженого кола осіб. Стабільність роботи гарантована при числі користувачів, що не перевищують 30.
Пред'являються вимоги до освітлення	Система буде використовуватися в закритому приміщенні. Освітленість протягом дня не буде сильно змінюватися.
Зовнішні перешкоди (наприклад, сонцезахисні окуляри)	У приміщенні немає потреби ходити в сонцезахисних окулярах. Даний атрибут повинен бути відсутнім на обличчі людини.
Відхилення голови неприпустимі	В якості шаблонних зображень будуть задаватися кілька положень голови щодо камери, що дозволить знизити помилку 2 роду.

Рисунок Г.6 – Слайд №6

Кореляційне зіставлення зображень

$$R(x, y) = \frac{\sum_{(x', y') \in X} (T'(x', y') \cdot I'(x + x', y + y'))}{\sqrt{\sum_{(x', y') \in X} (T'(x', y'))^2} \sqrt{\sum_{(x', y') \in X} (I'(x + x', y + y'))^2}}$$

де T – Template – шаблон, який шукаємо на зображенні;
 I – Image – зображення, в якому очікуємо знайти відповідності.

Мета полягає в тому, щоб виявити найвищу область відповідності. Для кожного розташування T над I записується метрика в матрицю результатів R . Кожне місце (x, y) в R містить метрику відповідності.

Рисунок Г.7 – Слайд №7

Порівняння мов програмування

Критерії порівняння	Мови програмування		
	C++ та OpenCV	Python та TensorFlow	MATLAB
Наявність літератури та простота вивчення	++	++	++
Підтримка парадигми модульного програмування	±	+	–
Продуктивність роботи	++	+	–
Паралельна обробка запитів та реалізація	++	++	±
Простота розроблення графічного інтерфейсу	+	±	±

Рисунок Г.8 – Слайд №8

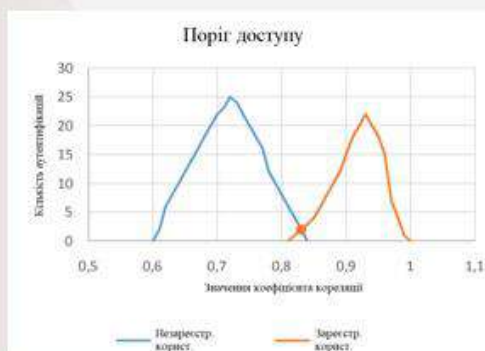


Рисунок Г.9 – Слайд №9



Рисунок Г.10 – Слайд №10

Визначення порогового доступу



11

Рисунок Г.11 – Слайд №11

Результати тестувань

	FAR	FRR
Кількість експериментів	m=100	n=100
Кількість (вдалих / невдалих) результатів	1	3
Імовірність помилки	0.01	0.03
Достовірність результатів	$m \geq 1/FAR$ m=100 достовірний	$n \geq 1/FRR$ n>33.3 достовірний

FAR – False Acceptance Rate – коефіцієнт помилкового доступу – за природою схожий до помилки 2 роду. Величина називається специфічність.

FRR – False Rejection Rate – коефіцієнт помилкової відмови в доступі – за природою схожий до помилки 1 роду. Величина називається чутливість.

12

Рисунок Г.12 – Слайд №12

Висновки

Під час виконання магістерської роботи, було запропоновано метод аутентифікації на базі біометричних даних користувача, а саме зображення обличчя з використанням кореляційного зіставлення зображень. Також було розроблено програмний комплекс, що реалізує роботу запропонованого методу.

Запропонований метод підвищує рівень безпеки систем, що використовують розширену систему доступу до інформації/функціоналу.

Під час розробки програми та аналізу розробленого програмного комплексу, а саме використання математичного апарату штучних нейронних мереж для розпізнавання зображень та більш точного використання.

Рисунок Г.13 – Слайд №13

Дякую за увагу

[1]



[2]



[1] – <http://www.zntu.edu.ua/>
 [2] – <http://pazntu.net/>

Рисунок Г.14 – Слайд №14