НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

«На правах рукопису»
УДК 004.93

До захисту допущено:
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2022 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою «Інформаційне забезпечення робототехнічних систем»
зі спеціальності 126 «Інформаційні системи та технології»
на тему: «Система виявлення та запобігання спуфінг-атакам під час біометричної ідентифікації за обличчям людини»

Виконав:
студент VI курсу, групи ІК-з11мп
Журавльов Дмитро Дмитрович _____

Керівник:
Старший викладач,
Польшакова Ольга Михайлівна _____

Рецензент:
Посада, науковий ступінь, вчене звання,
Прізвище, ім’я, по батькові _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2022 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – другий (магістерський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційне забезпечення робототехнічних систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2022 р.

ЗАВДАННЯ

на магістерську дисертацію студенту

Журавльову Дмитру Дмитровичу

1. Тема дисертації «Система виявлення та запобігання спуфінг-атакам під час біометричної ідентифікації за обличчям людини», науковий керівник дисертації Польшакова Ольга Михайлівна, старший викладач, затверджені наказом по університету від 08.11.2022 р. № 4097-с

2. Строк подання студентом дисертації 12.12.2022

3. Об'єкт дослідження: спуфінг-атаки на системи ідентифікаці за біометрією обличчя людини.

4. Вихідні дані: програма системи виявлення спуфінг-атак, технічна документація до розробленої системи, вісім додатків з графічним матеріалом.

5. Перелік завдань, які потрібно розробити: проаналізувати предметну область, існуючі рішення для виявлення спуфінг-атак, розробити моделі та обрати алгоритми, розробити архітектуру додатку, провести оцінку роботи системи, протестувати розроблену систему, провести маркетинговий аналіз стартап проекту.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: _____
_Структурна схема системи, функціональна схема системи, блок-схема ___
_алгоритму роботи додатку, діаграма послідовностей, діаграма прецедентів

7. Орієнтовний перелік публікацій _____
_____
_____

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання 05 вересня 2022 року

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз предметної області, огляд Літератури та існуючих рішень	15.09.2022	
2	Аналіз основних проблем під час збору та обробки даних з відеопотоків пристроїв ідентифікації	01.10.2022	
3	Проектування алгоритму роботи системи з виявлення спуфінг-атаки на зображенні	10.10.2022	
4	Вибір та обґрунтування оптимальності технічних рішень, реалізація системи	01.11.2022	
5	Тестування системи та аналіз отриманих результатів	10.11.2022	
6	Розробка стартап-проекту	20.11.2022	
7	Оформлення матеріалів дисертації	01.12.2022	

Студент _____ Дмитро ЖУРАВЛЬОВ
(підпис)

Науковий керівник _____ Ольга ПОЛЬШАКОВА
(підпис)

РЕФЕРАТ

Магістерська дисертація містить 122 сторінку, 58 рисунків, 34 таблиць, 8 додатків, 67 джерел.

Тема: Тема магістерської дисертації “ Система виявлення та запобігання спуфінг-атакам під час біометричної ідентифікації за обличчям людини”.

Актуальність: Актуальність магістерської дисертації полягає в тому, що сучасні системи захисту смартфонів та ноутбуків все частіше використовують біометричні дані користувача для ідентифікації та автентифікації, в тому числі обличчя людини. Такі світові корпорації як Apple, Samsung та Google використовують підсистеми ідентифікації користувача за обличчям у своїх пристроях, в тому числі смартфонах, що мають доступ до банківських даних людей, через системи Apple Pay та Google Pay, тому захист таких систем ідентифікації є вкрай важливим на сьогоднішній день.

Мета: Метою роботи є створення системи з виявлення та протидії спуфнінг-атакам, яка б надавала показник HTER менше 1%, а також могла працювати як незалежна система напряму з сенсором, так і в заємодіїї з іншими системами через відповідний інтерфейс.

Задачі: задачами роботи для досягнення мети є:

- Дослідження існуючих видів спуфінг-атак;
- Аналіз існучих алгоритмів виявлення спуфінг-атак, їх порівняння;
- Розробка власної систему виявлення спуфінг-атак;
- Тестування та порівння системи з існуючими рішеннями.

Об’єкт: Об’єктом дослідження є спуфінг-атаки на системи ідентифікації за біометрією обличчя.

Предмет дослідження: Предметом дослідження є системи виявлення та протидія спуфінг-атакам у системах ідентифікації за біометрією обличчя.

Ключові слова: анти-спуфінг, ідентифікація обличчя, машине навчання, ключові точки і ознаки, біометрична автентифікація та ідентифікація.

ABSTRACT

The master's thesis contains 122 pages, 58 figures, 34 tables, 8 appendices, 67 sources.

The topic of the master's thesis " System for detecting and preventing face spoofing attacks".

Relevance: The relevance of the master's thesis lies in the fact that modern smartphone and laptop protection systems increasingly use the user's biometric data for identification and authentication, including a person's face. Global corporations such as Apple, Samsung, and Google use subsystems of user identification based on their faces in their devices, including smartphones, which have access to people's banking data through the Apple Pay and Google Pay systems, so the protection of such identification systems is extremely important today.

Purpose: The purpose of the work is to create a system for detecting and countering spoofing attacks, which would provide an HTER rate of less than 1% and could also work as an independent system directly with a sensor, and also in interaction with other systems through the appropriate interface.

The tasks*:* to achieve the goal there are several tasks for the work*:*

- Research of existing types of spoofing attacks;
- Analysis of existing algorithms for detecting spoofing attacks, comparing them according to the HTER indicator and the number of types of attacks they can detect;
- Development of own spoofing attack detection system;
- Testing and comparing the system with existing solutions.

Object: The object of the study is spoofing attacks on facial biometric identification systems.

Subject: The subject of the study is detection systems and countermeasures against spoofing attacks in facial biometrics identification systems.

Keywords: anti-spoofing, facial recognition, machine learning, key points and features, biometric authentication and identification.

ЗМІСТ

ВСТУП 9

1 АНАЛІЗ ПРОБЛЕМ ВИЯВЛЕННЯ СПУФІНГ-АТАК ТА ОГЛЯД ІСНУЮЧИХ РІШЕНЬ 11

1.1 Види спуфінг-атак на системи лицьової біометрії 11

1.1.1 Загальний огляд атак на системи біометричної ідентифікації 11

1.1.2. Атаки з використанням масок 13

1.1.3. Атаки з використанням роздрукованих зображень 14

1.1.4. Атаки з використанням дисплеїв пристроїв 14

1.2 Визначення ефективності анти-спуфінг системи 16

1.3 Існуючі підходи для виявлення спуфінг-атак та їх порівняння 17

1.3.1 Динамічні підходи для виявлення спуфінг-атак 17

1.3.2. Статичні підходи для виявлення спуфінг-атак 18

1.3.3. Підходи з використанням сенсорів для виявлення спуфінг-атак 19

1.4 Огляд існуючих системи для боротьби зі спуфінг-атаками 22

1.5 Порівняльна характеристика існуючих анти-спуфінг систем 31

1.6 Формування функціональних вимог 33

1.7 Формування нефункціональних вимог 34

1.8 Висновки до розділу 35

2 ЗАПРОПОНОВАНЕ РІШЕННЯ ТА РОЗРОБКА СТРУКТУРНОЇ СХЕМИ 36

2.1 Запропоноване рішення для виявлення спуфінг-атак 36

2.2 Вибір алгоритму для виокремлення обличчя людини з відеопотоку 42

2.3 Розробка структурної схеми 44

2.4 Висновки до розділу 46

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ 47
3.1 Мова програмування та засоби розробки програмної системи 47
3.2 Вибір необхідних фреймворків та бібліотек 49
3.3 Опис архітектури програмної системи 53
3.4 Алгоритм роботи програми 54
3.5 Реалізація сервісів системи та діаграми класів 55
3.6 Проектування структури бази даних для застосунку 63
3.7 Тренування ETC класифікатору 69
3.8 Висновки до розділу 72
4 РОЗРОБКА ІНТЕРФЕЙСУ СИСТЕМИ 73
4.1 Опис графічного інтерфейсу користувача 73
4.2 Опис REST інтерфейсу 75
4.3 Висновки до розділу 77
5 ТЕСТУВАННЯ СИСТЕМИ 78
5.1 Тестування на датасетах CASIA FASD та Idiap REPLAY-ATTACK 78
5.2 Тестування на власних зразках зображень обличь 80
5.3 Тестування REST API 84
5.4 Висновки до розділу 88
6 ІНТСРУКЦІЯ КОРИСТУВАЧА ТА СЦЕНАРІЇ ВИКОРИСТАННЯ СИСТЕМИ ... 90
6.1 Сценарії використання системи 90
6.2 Інструкція з використання системи 93
6.3 Висновки до розділу 96
7 РОЗРОБКА СТАРТАП-ПРОЕКТУ 97
7.1 Опис ідеї проекту 97
7.2 Технологічний аудит ідеї проекту 98

7.3 Аналіз ринкових можливостей запуску стартап-проекту 100
7.4 Розроблення ринкової стратегії проекту 107
7.5 Розроблення маркетингової програми стартап-проекту 110
7.6 Висновки до розділу 114
ВИСНОВКИ 115
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 117

ВСТУП

Підсистеми захисту інформаційних систем, що використовують біометричні дані людини, в тому числі риси обличчя, вже не перший рік присутні на ринку інформаційних технологій. В той час коли звичайний текстовий парооль користувач може втратити або забути, так само як і можуть вкрасти електроні ключі доступу, то втратити унікальні біологічні особливості людини неможливо, а підробити їх вкрай важко. Одні з найпоширеніших біологічних ознак, що використовуються для ідентифікації – це голос людини, її візеруноку відбитків пальців, хода, візерунок судин ока, форма обличчя та інше. Так одними із розповсюджених систем біометричної ідентифікації та автентифікації є системи, що використвують обличчя людини, наприклад, системи компаній Apple, Samsung та Microsoft. На сьогоднішній день системи розпізнавання обличчя мають доволі велику точність за рахунок появих об’ємних датасетів зображень обличь та сучасних алгоритмів машиного навчання і моделей нейромереж. Так точність сучасних система вже досягає 0,000001 (одна помилка на мільйон), і дані системи мають достатньо висоску продуктивність з точки зору затрат оперативної пам’яті та CPU для застосування їх на мобільних платформах. Але головною проблемою у даних системах залішається їх безпека.

Проблеми безпеки технологій бометричної ідентифікації за обличчям людини, що швидко розвиваються, до зовнішніх атак і, зокрема, до підробки біометричніх даних почали вирішувати лише нещодавно. Спуфінг-атака, яку також у наукових роботах називають атакою на презентацію, є суто біометричною вразливістю, яка не є спільною з іншими рішеннями IT-безпеки. Суть атаки полягає у здатності обдурити біометричну систему, щоб вона розпізнала нелегітимного користувача як справжнього за допомогою представлення сенсору синтетичної підробленої версії оригінальної біометричної характеристики.

Великий інтерес до теми спуфінг-атак був привернутий після вдалої спроби взлому системи Face ID на смартфоні iPhone моделі “X”, з використанням спеціальної маски складної конструкції, яка була виготовлена з кам'яного порошку з собливими патчами біля очей, і при нагріві ці патчі відтворювали тепло справжнього обличчя людини за рахунок випромінювання інфрачервоного спектру [67]. Подібні

вразливості становлять велику загрозу для користувачів сучасних смартфонів, які сьогодні використовуються як засіб оплати з технологією NFC, що вже казати про системи ідентифікації та автентифікаціхї, які працюють в державному або банківському секторі.

Перші системи для боротьби зі спуфінг-атаками почали з’являтися у 2012 році, такі системи почали називати анти-спуфінг системами. Так дана робота містить аналіз існучих видів спуфінг-атак та алгоритмів для їх виявлення, щоб обрати найкращий алгоритм для розробки власної анти-спуфінг системи, який би забезпечив найкраще значення показника HTER та міг би виявляти більшу частину наявних видів спуфінг-атак. Розроблена система має можливість працювати як окрема самостійна система і напряму взаємодіяти із сенсором (веб-камерою у нашому випадку), а також може працювати як підсистема і взаємодіяти з іншими через REST API.

1 АНАЛІЗ ПРОБЛЕМ ВИЯВЛЕННЯ СПУФІНГ-АТАК ТА ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

1.1 Види спуфінг-атак на системи лицьової біометрії

1.1.1 Загальний огляд атак на системи біометричної ідентифікації

Атаки на біометричні системи ідентифікації за обличчям можна розділити на два типи: непрямі та прямі [1]. На рисунку 1 зображено блок-схему типової біометричної системи ідентифікації за обличчям людини з пронумерованими блоками, де можуть статися потенційні атаки. Непрямі атаки здійснюються зсередини системи, вимагаючи спочатку, щоб зловмисники отримали доступ до внутрішніх елементів такої системи. Потрапивши всередину, зловмисники можуть, наприклад, підробити екстрактори або компаратори функцій (блоки 3 і 5 на рис. 1), маніпулювати збереженими біометричними ознаками (блок 6) або використовувати можливі слабкі місця у каналах зв'язку (блоки 2, 4, 7 і 8). З непрямими атаками можна боротися, підвищивши безпеку каналів зв'язку та заблокувавши доступ до внутрішніх елементів систем розпізнавання, щоб кіберзлочинці не могли скористатися ними.

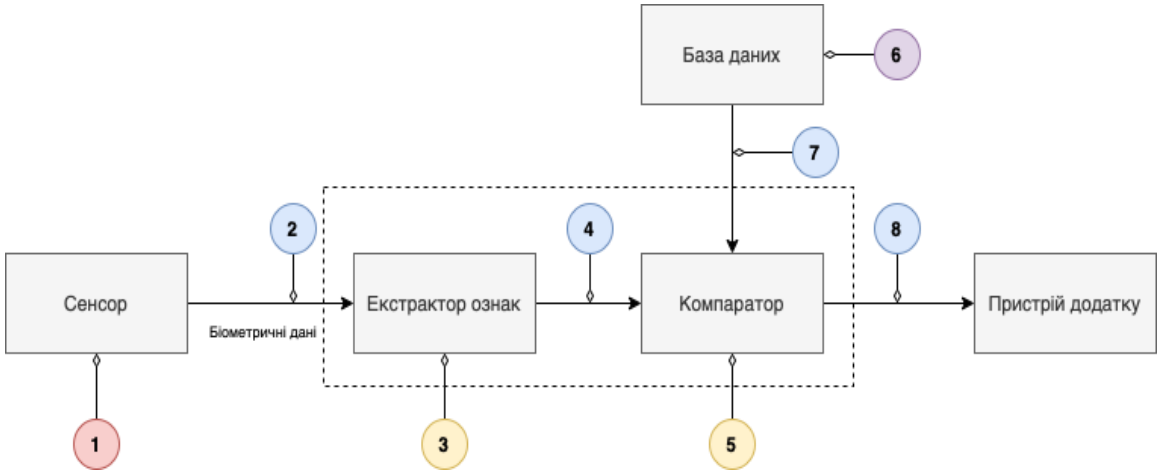


Рисунок 1.1 – Можливі точки атаки в типовій біометричній системі ідентифікації

Прямі або спуфінгові атаки [2] виконуються на рівні сенсора (відеокамери), що знаходиться поза контролем виробника біометричної системи. У таких випадках зловмисник намагається безпосередньо обдурити сенсор і, таким чином, не можна використати жоден з механізмів фізичного захисту. У прямій атаці, яка також

називається спуфінг-атакою, особа намагається маскуватися під іншу особу, фальсифікуючи свої біометричні характеристики та отримуючи таким чином несанкціонований доступ до системи.

Більшість систем розпізнавання обличчя використовують звичайні відеокамери як вхідні датчики. Ці пристрої можуть використовуватися для отримання однієї фотографії або відеорядів з обличчям людини, яка намагається заволодіти доступом до захищених ресурсів. Одним з важливих аспектів, на який треба звернути увагу під час розробки біометричної системи ідентифікації, є умови середовища під час збору даних. Добре відомий факт, що погані умови освітлення, поза та старіння серед інших варіацій можуть суттєво погіршити здатність розпізнавати обличчя людини [3]. Також користувачі можуть використовувати смартфон (рис. 2) для доступу до захищених ресурсів на самому пристрої або використовувати смартфон, як термінал для доступу до інших програм. У таких випадках умови навколишнього середовища можуть сильно відрізнятися при кожній спробі ідентифікації.

Переважну більшість спуфінг-атак можна класифікувати в одну з двох груп, залежно від того, чи застосовані засоби використовують 2D-простір (наприклад, роздруковане фото, відео на дисплеї), які успішно атакують двовимірні системи розпізнавання обличчя, тобто системи, де сенсором є відеокамера, або 3D-об’єми (наприклад, маски), які також можна використовувати для атаки на двовимірні системи розпізнавання обличчя. Для сучасних систем ідентифікації за обличчям людини можна виділити три основні типи атак, які розглянемо у наступних підрозділах.

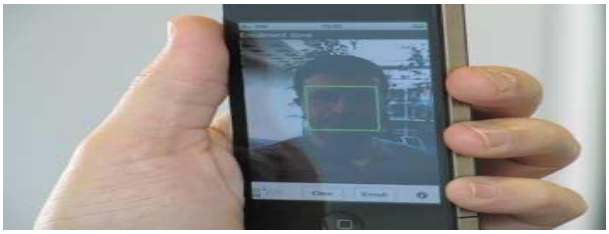


Рисунок 1.2 – Смартфон як датчик та система розпізнавання обличчя

1.1.2 Атаки з використанням масок

Найрозповсюдженішим методом спуфінг-атак на системи біометричної ідентифікації за обличчям людини є застосування масок, які відтворюють людські обличчя, також даний метод відомий як Mask attack. Доволі очевидний та простий у імплементації метод, який досить часто використовується професіональними злочинцями для обходу захисту біометричної системи ідентифікації (рис. 1.3).

Атаки з масками вимагають багато навичок, щоб їх добре виконати і доступ до додаткового матеріалу, оскільки потрібно створити приблизний 3D-прототип обличчя. Цей тип атаки, можливий для двовимірних систем розпізнавання обличчя, причому ймовірність успіху таких атак більша, оскільки контрзаходи можуть бути не в змозі досліджувати шаблони деформації.

Також треба зазначити, що готових рішень для протидії Mask attack доволі мало. Брак дослідницьких робіт, присвячених цій потенційній загрозі, можна пояснити технічними й економічними труднощами, пов’язаними зі створенням великих баз даних реалістичних масок. Однак ці перешкоди значно зменшилися з нещодавньою появою деяких компаній, де такі 3D-моделі обличчя можна отримати за розумну ціну [4]. Крім того, самостійне виготовлення маски для обличчя з кожним днем стає все більш здійсненним і легшим за допомогою нового покоління доступних апаратів 3D-зйомки, спеціального програмного забезпечення для сканування об’ємних об’єктів та зниження ціни на пристрої для 3D-друку [5].

Рисунок 1.3 – Приклад Mask attack

1.1.3 Атаки з використанням роздрукованих зображень

Інший вид атак який ми розглянемо використовує справжнє або підроблене фото обличчя особи, також має назву Printed attack. Ці спроби шахрайського доступу здійснюються з допомогою надання системі роздрукованої фотографії для розпізнавання справжнього користувача. Фотографія могла бути зроблена зловмисником за допомогою цифрової камери або навіть отримана з інтернету після того, як користувач сам завантажив її в одну з дуже популярних соціальних мереж онлайн. Зображення можна роздрукувати на папері (тобто атаки на друк, які були першими систематично досліджені в літературі) або відобразити на екрані цифрового пристрою (рис. 1.4), такого як мобільний телефон або планшет. Трохи більш просунутий тип Printed attack, який також вивчався, це використання фотомасок. Ці маски являють собою надруковані фотографії високої роздільної здатності, на яких вирізані очі та рот. Під час нападу самозванець розташовується позаду [6], щоб відтворити певні рухи обличчя, наприклад моргання очима.

Рисунок 1.4 – Приклад Printed attack

1.1.4 Атаки з використанням дисплеїв пристроїв

Інший тип атак дещо складніший у реалізаціїї та має назву Replay attack. Суть методу полягає у тому, що до сенсору (веб-камери у нашому випадку) підносять дисплей пристроя, наприклад, планшету на якому завчасно підготовлене відео з обличчям людини, шо рухається. Незважаючі на всю складність імплементації даного

методу він має доволі високу ефективність за рухонок того, що анти-спуфінг системи доволі часто використовують динимачіні підходи виявлення спуфінг-атак за рахунок аналізу часових послідовностей кадрів, наприклад, виявленння мікрорухів очей, дихання, міміки обличчя та звичайно моргання очей тощо. Всі ці ознаки живої людини легко відтворити на відеозапису. З появою загальнодоступних сайтів для обміну відео та відповідним зниженням цін на високоякісні камери, отримання зразків клієнтів стає дедалі легшим. Крім того, технологія, яка зазвичай використовується в анімаційному програмному забезпеченні для моделювання вигаданих персонажів, також може бути застосована для створення реалістичних підроблених біометричних зразків, які все ще демонструватимуть характеристики живої людини.

Два останні види спуфінг-атак (Replay attack та Printed attack) мають перелік певних ознак (рис. 1.5), за рахунок яких їх доволі просто виявити, і, таким чином, з’являється можливість відрізняти дисплей смартфона або роздруковане фото від реальної людини.

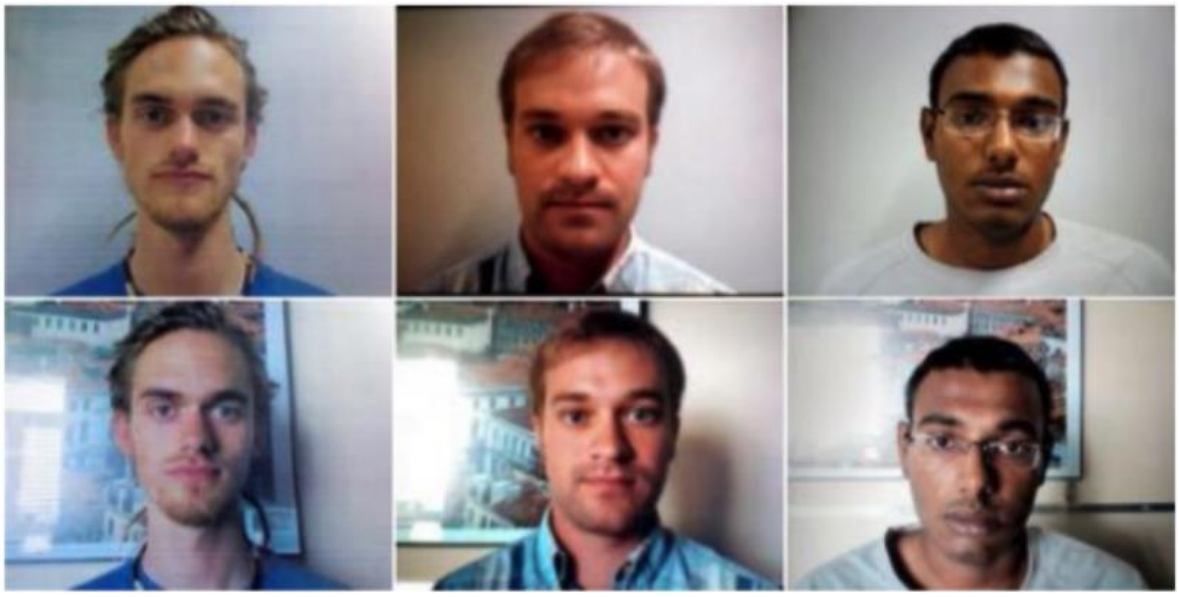


Рисунок 1.5 – Приклади Replay attack з наявним муаром зображення

Усі наявні ознаки, за рахунок яких можливо визначити Replay та Printed attack, були записані у таблицю 1.1.

Таблиця 1.1 – Ознаки Replay та Printed attack

Printed attack	Replay attack
Зниження якості текстури зображення під час друку	Муар
Артефакти передачі напівтонового зображення під час друку на принтері	Відображення (відблиски)
Механічні артефакти друку (горизонтальні лінії)	Плоский малюнок (відсутність глибини)
Відсутність локальних рухів (наприклад, моргання)	Можуть бути видні межі зображення
Можуть бути видні межі зображення	

1.2 Визначення ефективності анти-спуфінг системи

Для визначення ефективності роботи системи біометричного захисту найпоширенішою у використані є метрика HTER (формула 1.1), яка обчислюється у вигляді суми коефіцієнтів помилково дозволених ідентифікацій (англ. FAR – False Acceptance Rate) та помилково заборонених ідентифікацій (англ. FRR – False Rejection Rate), поділеної навпіл:

$$HTER = (FAR + FR\text{R})/2 \qquad (1.1)$$

Треба також звернути увагу на те, що в більшості системах з біометричним захистом в першу чергу приділюють увагу показнику FAR, так як першим пріоритетом ставлять безпеку системи і вкрай важливо недопустити жодного зловмисника, але є певний нюанс. При зменьшенні показника FAR настає така проблема, як невідворотне зростання FRR, тобто користувачі, що зареєстровані у системі, будуть все частіше помилково позначатися як несправжні особи, і у випадках з банківськими, державними, мілітарними та іншими структурами, що працюють з чутливими даними, цим можна пожертвувати, то комерційні мобільні технології, що

працюють з їх величезними масштабами та великою варіацією клієнтських пристроїв, є досить залежними від будь-яких факторів, що мають вплив на задоволення користувачів, наприклад, швидкість роботи системи, та можуть змусити їх - відмовитися від послуг, тому необхідно звернути особливу увагу при розробці системи на даний показник.

1.3 Існуючі підходи для виявлення спуфінг-атак та їх порівняння

1.3.1 Динамічні підходи для виявлення спуфінг-атак

Один із перших підходів до захисту систем двовимірного розпізнавання обличчя з’явився як протидія першим вивченим атакам, які використовували статичні роздруковані обличчя, тобто printed attack. Такі методи захисту від спуфінг-атак, які все ще залишаються досить популярними проти printed attack, покладаються на виявлення рухів обличчя на відео. Зокрема, вони засновані на аналізі траєкторії окремих сегментів обличчя. Типові ознаки які використовуються в цьому типі методів боротьби зі спуфінгом такі: кліпання очима [6]; жести обличчя та голови (наприклад, кивання, посмішка, погляд у різні боки), виявлення за допомогою відстеження погляду. Ці методи зазвичай дуже ефективні для виявлення printed attack, але втрачають точність проти спроб незаконного доступу, здійснених із відтворюваними відео, де підроблено не лише зовнішній вигляд обличчя, але й його рухи.

Щоб подолати цей недолік, були спеціально запропоновані деякі додаткові методи для виявлення атак на основі відео: використання 3D-структури обличчя за допомогою аналізу кількох 2D-зображень із різними позами голови [7]; використання аналізу на основі контексту, щоб отримати переваги нелицевої інформації, доступної з отриманих зразків, такої як характеристики руху зі сцени, наприклад, рух фону чи переднього плану; оцінка шуму, створюваного під час процесу повторного захоплення [8]. Динамічні системи захисту від спуфінгу зазвичай досягають дуже конкурентоспроможної продуктивності. Однак, як обмеження, їх не можна

використовувати в системах, де доступне лише одне зображення обличчя користувача (наприклад, програми пов’язані з зображенням паспорта), більше того, навіть у сценаріях, коли відеодані були записані (наприклад, програми спостереження), нерідко можна виявити, що лише кілька непослідовних кадрів придатні для аналізу обличчя, що також обмежує їх кінцеве використання та точність.

1.3.2 Статичні підходи для виявлення спуфінг-атак

Як згадувалося в попередньому підрозділі, для динамічних систем захисту від спуфінгу потрібна часова послідовність рухів частни обличчя достатньої тривалості для досягнення високої точності. Це обмеження спонукало до появи другої групи підходів для виявлення спроб спуфінг-атак на системи біометричної ідентифікації за обличчям, орієнтованих на аналіз одного статичного зображення, а не відеоданих. Ці методи загалом швидші, ніж їхні динамічні аналоги, і, отже, зручніші для користувача, за рахунок, у деяких випадках, певної втрати продуктивності.

Переважна більшість статичних методів на рівні ознак базується на аналізі текстури зображення обличчя за допомогою різних засобів обробки зображень, таких як: спектр Фур’є; численні фільтри різниці Гауса для вилучення інформації про конкретну частоту, яка також була поєднана з функціями, отриманими з моделі Ламберта, що підтверджує чудову продуктивність навіть за умов поганого освітлення [9]; один з останніх підходів, заснований на використанні локальних бінарних патернів (LBP) для виявлення printed attack, який був успішно поєднаний з іншими дескрипторами текстур, такими як вейвлети Габора, і з інформацією, пов’язаною з формою, отриманою за допомогою гістограми.

Ці статичні підходи проти спуфінг-атак також можуть бути застосовані для випадків, коли доступна відеопослідовність. У цьому сценарії аналіз виконується покадрово з використанням методів об’єднання (наприклад, голосування більшістю) на пізнішому етапі для об’єднання окремих оцінок, отриманих від кожного кадру, для створення унікального остаточного рішення. Хоча така стратегія і існує, загалом вона

менш ефективна, ніж системи, спеціально розроблені для роботи з відео, оскільки не використовується часова інформація.

Деякі з попередніх методів були успішно об’єднані, демонструючи підвищену точність порівняно з окремими параметрами [6]. Порівняльне дослідження кількох із цих динамічних і статичних підходів можна знайти в результатах змагань із протидії двовимірним спуфінг-атакам 2011 і 2013 років [10], де було показано, що поєднання обох типів технік (статичної та динамічної) на рівні функцій забезпечує найкращу продуктивність.

1.3.3 Підходи з використанням сенсорів для виявлення спуфінг-атак

Щодо методів захисту від спуфінгу з використанням сенсорів, тобто апаратного забезпечення, кількість внесків все ще не порівнянна з підходами, заснованими на програмному забезпеченні. Однак запропоновано деякі цікаві методи, засновані на технологіях за межами візуального спектру, наприклад: комплементарні інфрачервоні (ІЧ) або ближні інфрачервоні (NIR) зображення [11]; порівняння інформації про відбивну здатність світла справжніх обличь людей і матеріалів з яких роблять маски за допомогою спеціальної установки світлодіодів і фотодіодів на двох різних довжинах хвиль [12].

На додаток до попередніх робіт, існують інші технології, які також можуть бути використані як методи захисту від спуфінг-атак на рівні сенсора, хоча в більшості випадків ще не було проведено ретельних досліджень щодо їх ефективності за сценаріями виявлення живого обличчя людини, такі потенційно корисні механізми для запобігання спуфінг-атак включають: тепловізор, виявлення малюнка вен на обличчі, або отримання 3D моделі обличчя (рис. 1.6). Наприклад, 3D-сенсори можуть бути дуже надійними проти атак, які здійснюються з плоскими поверхнями, наприклад, ті ж самі printed attack, оскільки майже не виявляється різниця в глибині отриманої 3D моделі порівняно з 3D моделлю реального обличчя. З іншого боку, ефективність 3D-сенсорів під час маскових атак тільки почали досліджувати за допомогою аналізу текстур, натхненного двовимірними методами захисту на основі

LBP, а також аналізом компонентів відбиття світла, які можна обчислити з 3D сканів [10].

Подібним чином системи, засновані на виявленні термограми обличчя, в принципі, будуть дуже точними для виявлення всіх трьох типів основних атак підробки обличчя (тобто printed attack, replay та mask attack), оскільки не очікується теплової різниця у фальшивих обличчях. Деякі початкові спроби вивчити теплові зображення для виявлення справжнього обличчя вже були зроблені [13], включаючи придбання значно великої бази даних теплових зображень для стандартних і замаскованих спроб доступу, де були отримані доволі точні результати.

Рисунок 1.6 – Приклад сканер для отримання 3D моделі обличчя

Той факт, що дані методи, тобто 3D-розпізнавання та термічне розпізнавання обличчя, уже мають надійну та протестовану базу для особистої ідентифікації, може стати додатковою перевагою для їх розробки як альтернативи для підвищення безпеки, оскільки можна виконувати обидва завдання, тобто розпізнавання та захист від спуфінг-атак.

Зазвичай спостерігається, що методи боротьби зі спуфінг-атаками не працюють узгоджено на різних датасетах, і можуть мати значні втрати точності, коли їх тестують в умовах, відмінних від тих, для яких вони були розроблені. Таким чином, як правило, найкращі результати досягаються шляхом поєднання кількох

взаємодоповнюючих алгоритмів або функцій так, що слабкі сторони одних перекриваються сильними сторонами інших і навпаки. Недоліки та переваги представлених раніше трьох підходів виявлення спуфінг-атак зведені у таблицю 1.2.

Таблиця 1.2 – Підходи з виявлення спуфінг-атак

Тип підходу	Переваги	Недоліки
Динамічні підходи	— Використовують просторові та часові особливості відео — Висока точність — Дуже ефективні проти printed attacks	— Не можна використовувати в сценаріях тільки з одним фото (наприклад, фото паспорту) — Повільні — Втрата точності при replay attacks
Статичні підходи	— Можна використовувати з одним зображенням або відеопотоком — Швидкі — Повністю прозорий для користувача	— На основі лише просторової інформації зображення — Нижча точність
З використанням сенсорів	— Може бути ефективним проти printed,	— Дорогі — Зазвичай повільніші

	replay і mask attacks — Дуже висока точність	— Потрібен вищий рівень співпраці з боку користувачів

1.4 Огляд існуючих системи для боротьби зі спуфінг-атаками

Перші підходи були засновані на виявленні моргань очей людини шляхом аналізу зображення за маскою на початку 2007-2008 років [14]. Ідея заснована на побудові бінарного класифікатора, що дозволяє виділити зображення з відкритими та закритими очима у послідовності кадрів (рис. 1.7). Можливий аналіз відеопотоку за допомогою виділення частин обличчя (landmark detection), або використання, наприклад, нескладної нейронної мережі . Даний метод є доволі простим у реалізації тому часто у його можна побачити у самих простих системах захисту. На жаль, даний метод є досить простим для його обходу зловмисниками, тому, його використання у сучасних системах різко падає.

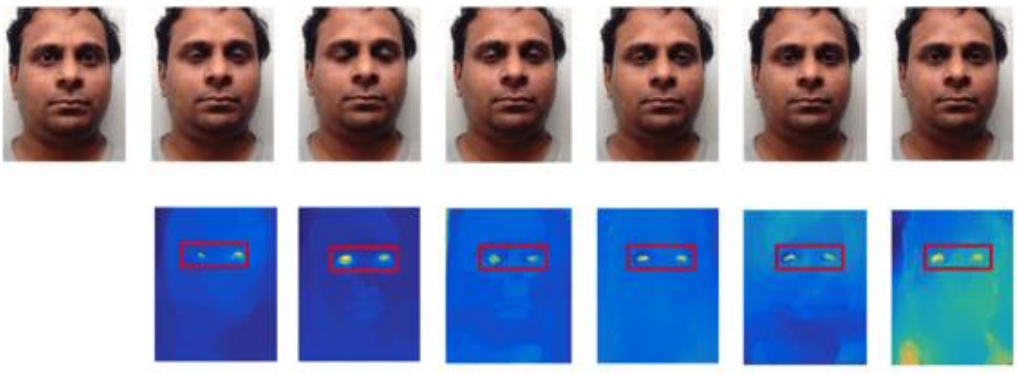


Рисунок 1.7 – Приклад роботи системи з виявлення моргань очей

Розповсюдженою стала система, що виявляє зменшення якості зображення після друку або відтворення на дисплеї електронного пристрою. Визначити погіршення якості зображення можна, наприклад, обчисливши локальні бінарні

патерни (local binary pattern) для різних ділянок обличчя після виділення їх з кадру [14]. Викладену систему вважають основою всього напряму алгоритмів face anti-spoofing з урахуванням аналізу зображення. Головна ідея полягає у тому, що ми послідовно беремо кожен піксель даного зображення, а також вісім сусідніх пікселів та порівнюємо їх яскравість і створюємо відповідну матрицю (рис. 1.8). Якщо яскравість більша, ніж на центральному пікселі, то у елмент матриці, що відповідає елементу матриці зображення записується одиниця, якщо менше – нуль. Так для кожного пікселя виходить матриця з дев'яти елементів, кожен елемент якої має значення 0 чи 1. За отриманими матрицями будується попіксельна гістограма, яка подається на вхід SVM-класифікатора [14].

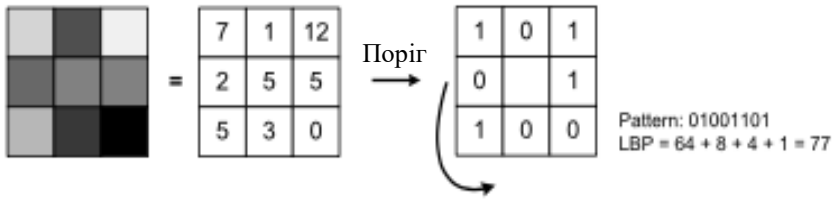


Рисунок 1.8 – Приклад обрахунку LBP

Для даної системи показник ефективності HTER складає приблизно 15%, і можна зробити висновок, що значна частина шахраїв долає захист без значних зусиль, хоча і слід визнати, що переважна кількість зловмисників не допускається системою. Алгоритм був протестований на наборі даних Replay-Attack від IDIAP, який містить у собі 1200 коротких відео 50-ти респондентів та трьох видів атак–printed attack, replay attack, high-definition attack.

Підходи з аналізу текстури зображення отримали продовження у 2015 році. Був розроблений алгоритм альтернативного розбиття зображення на канали, крім традиційного RGB був застосований канал YC_bC_r та HSV (рис. 1.9), для результатів яких знову підраховувалися локальні бінарні патерни, які за анологією з попереднім способом, були передані на вхід SVN класифікатора [15]. Показник HTER, розрахований на датасетах CASIA та Replay-Attack, склала доволі значні на той період 3%. Приблизно у цей же період (2015-2016 роки) з'являються анти-спуфінг

системи з виявлення муара. Було запропоноване рішення суть якого полягає у пошуку артефактів зображення у вигляді періодичного візерунка, спричинені накладенням двох розгорток (рис. 1.10) [16]. Запропоноване рішення виявилося доволі ефективним в порівняні з аналогами, показник HTER дорівнював приблизно 6% на наборах даних CASIA, IDIAP та RAFS. Запропонована система також була першою, ефективність роботи якої порівняли на різних наборах даних (датасетах).

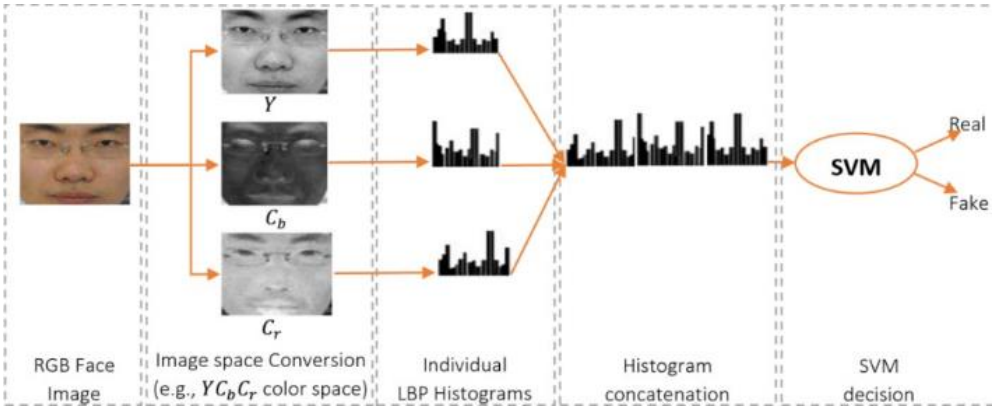


Рисунок 1.9 – Структура алгоритму запропонованого в дослідженні Букінафіта

Рисунок 1.10 – Періодичний візерунок муара на зображенні

Для детектування спроб printed attack, доволі очевидним рішенням було спробувати аналізувати не один взятий кадр, а декілька кадрів як послідовність, взяту з відео потоку. Була запропонована система яка виділяє ознаки з оптичного потоку на сусідніх парах кадрів та подає їх на вхід бінарного класифікатора і усереднює результати (рис. 1.11) [17]. Дана система виявилася досить ефективною, продемонструвавши HTER 1,52% на власному наборі даних, створеного Андре Анджесом.

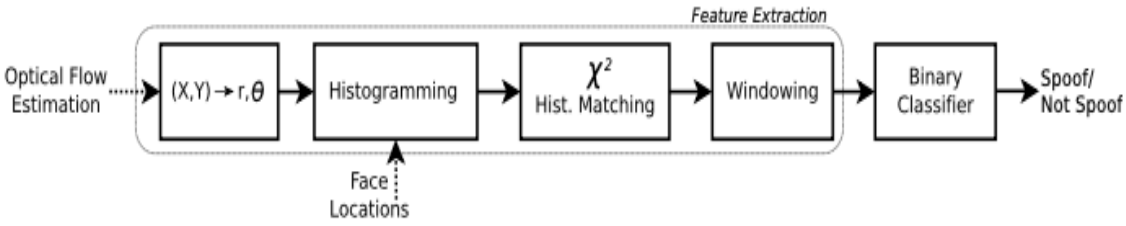


Рисунок 1.11 – Схема системи запропонованої Андре Анджесом

У 2013 була розроблена система, яка послідовно застосовувала складніші попередні перетворення відеопотоку, зокрема, відомий з робіт учених з MIT алгоритм ейлерівського посилення відео Eulerian video magnification [19], який успішно застосовувався для аналізу змін кольорів шкіряного покриву в залежності від пульсу [18], замінила LBP на HOOF (гістограми напрямів оптичного потоку), чітко зауваживши, що якщо ми хочемо відстежувати рухи, і ознаки нам потрібні мати відповідні вектори оптичного потоку, а не тільки аналіз текстур (рис. 1.12). Як класифікатор використовувався вже відомий SVM, традиційний на той момент. Алгоритм показав найкращі результати на той період часу на датасетах Print Attack 0% та Replay Attack 1,25%.

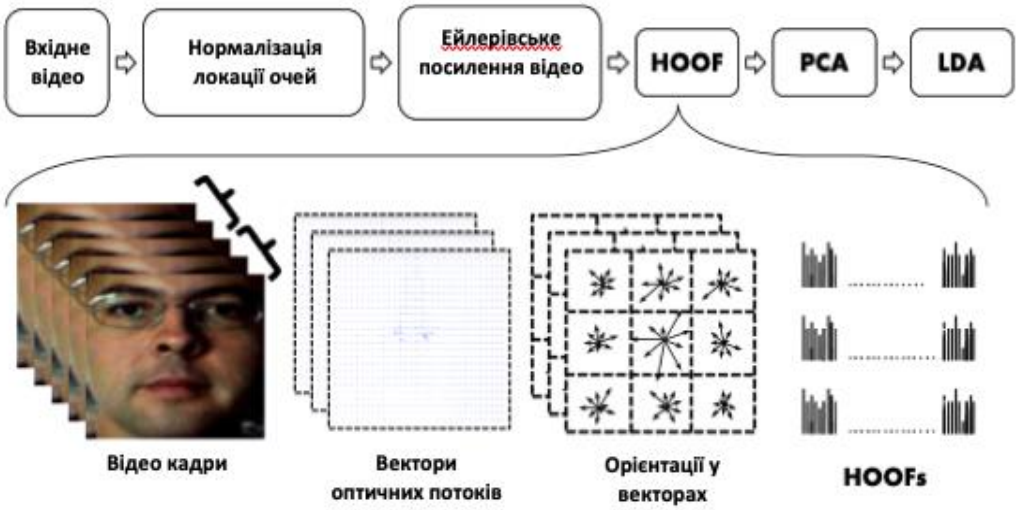


Рисунок 1.12 – Модель системи з використанням HOOF

З певного моменту розвитку анти-спуфінг систем почало з’являтися все більше досліджень заснованих на нейромережах з глибоким навчанням. Перша система, яка була розроблена, базувалася на методі аналізу карт глибини на окремих ділянках (патчах) зображення [20]. На перший погляд, карта глибини є дуже гарною ознакою визначення площини, в якій розташоване зображення, хоча б тому, що зображення на аркуші паперу «глибини» немає за визначенням (рис. 1.13). У дослідженнях проведених у 2017 року із зображення витягувалося достатньо велика кількість окремих невеликих ділянок (патчів), для них розраховувалися карти глибини, які потім зливали з картою глибини основного зображення [20]. Також вказувалося, що десять випадково отриманих патчів зображення обличчя достатньо для надійної протидії Printed Attack. Додатково автори зливали разом результати роботи двох згорткових нейромереж, перша з яких розраховувала карти глибини для патчів, а друга – для зображення загалом. Під час навчання на наборах даних із класом Printed Attack була визначена карта глибини, що має значення рівне нулю, а для тривимірної моделі обличчя – серія ділянок, що випадково відбираються. Якщо більш детально роздивитися систему, то сама по собі карта глибини була не така важлива, вона лише має певну індикаторну функцію, що допомагає характеризувати «глибину ділянки». Система показала значення HTER 3,78%. Для навчання були використані три відкриті набори даних – Replay-Attack, MSU-USSA та CASIA-MFSD.

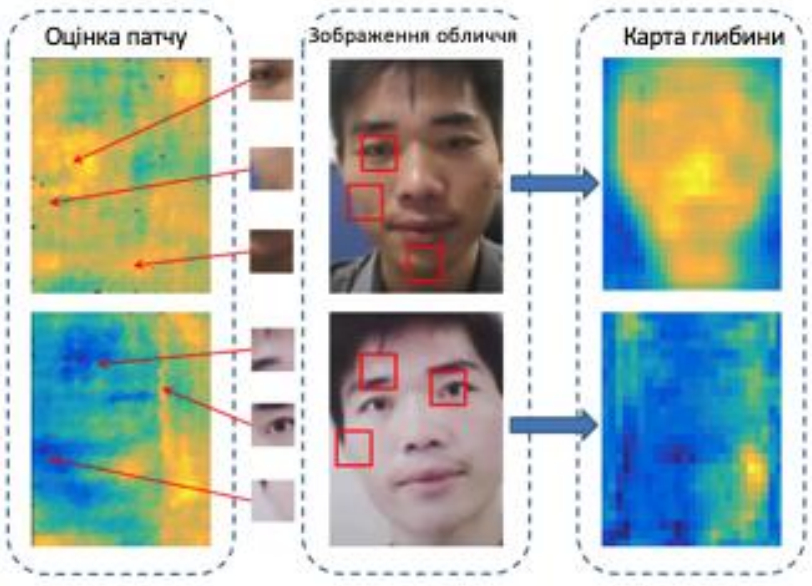


Рисунок 1.13 – Приклад аналізу зображення за картою глибини

На сьогоднішній день, доступність великої кількості якісних фреймворків для глибокого навчання нейромереж призвела до появи величезної кількості систем, які намагаються напряму вирішити задачу anti-spoofing в системах лицьової біометрії таким способом як комбінування нейромереж. Зазвичай це виглядає як стек карт ознак на виходах кількох нейромереж, преднавчиних на якомусь поширеному датасеті, який подається на бінарний класифікатор.

В цілому можна зробити висновок, що на сьогоднішній день опубліковано доволі багато робіт, які в переважній більшості демонструють гарні результати, але їх об'єднує один вагомий фактор, а саме: усі наявні результати продемонстровано в рамках одного конкретного датасету. Ситуація посилюється обмеженістю наявних наборів даних і, наприклад, на вже багаторазово згадоному Replay-Attack вже нікого не здивувати HTER 0%. Даний фактор призводить до появи доволі складних архітектур з застосуванням різних важких ознак, допоміжних алгоритмів, зібраних у стек, з кількома класифікаторами, результати яких усереднюються [21]. На виході автори отримують HTER ~ 0,04%. Це призводить до логічного висновку про те, що завдання face anti-spoofing у рамках конкретного датасету вирішено. В дослідженні Суаза Л. були зведені до таблиці різні сучасні методи на основі нейромереж [22]. Ми можемо самостійно побачити, що гарних результатів, тобто HTER <= 1%, вдалося досягти дуже різноманітними методами (рис. 1.14).

Автор	Ознаки	Класифікатор	HTER (%)
2012, Chingovska *et al.* [36]	LBP	SVM	15.16
2012, Freitas *et al.* [32]	LBP-TOP	SVM	7.60
2013, Komulainen *et al.* [31]	Motion Correlation + LBP	LLR + SVM + MLP	5.11
2013, Bharadwaj *et al.* [24]	HOOF + LBP	LDA	1.25
2013, CASIA [18]	**LBP + 1D-FFT + HMOF + Motion Correlation**	**SVM**	**0.00**
2013, IGD [18]	Motion	Adaboost	9.13
2013, MaskDown [18]	LBP + GLCM + LBP-TOP	LLR + LDA	2.50
2013, LNMIIT [18]	**LBP + GMM + 2D-FFT**	**SVM**	**0.00**
2013, Muvis [18]	LBP + Gabor Wavelets	PLS	1.25
2013, PRA Lab [18]	Color + Texture	SVM	1.25
2013, ATVS [18]	IQM	LDA	12.00
2013, UNICAMP [18]	2D-DFT + GLCM	SVM	15.62
2014, Galbally *et al.* [38]	IQA	LDA	15.20
2015, Menotti *et al.* [61]	DNN	CNN	0.75
2015, Tirunagari *et al.* [25]	DMD + LBP	SVM	3.75
2015, Wen *et al.* [67]	IDA	SVM	7.41
2015, Pinto *et al.* [58]	2D-DFT	PLS	14.27
2015, Boulkenafet *et al.* [41]	LBP + Color	SVM	2.90
2015, Arashloo *et al.* [52]	MLPQ-TOP + MBSIF-TOP	KDA	1.00
2015, Pinto *et al.* [65]	GMM + 2D-DFT	SVM	2.75
2016, Boulkenafet *et al.* [49]	LPQ + Color	SVM	3.30
2016, Feng et al. [12]	**HSC + Optical Flow**	**NN**	**0.00**
2016, Kim *et al.* [46]	MLBP + GLCM + IDA	SVM	5.50
2016, Phan *et al.* [54]	LDP-TOP	SVM	1.75
2017, Alotaibi and Mahmood [60]	AOS	CNN	10.00
2017, Lakshminarayana*et al.* [68]	Color	CNN	0.80

Рисунок 1.14 – Ефективність роботи анти-спуфінг систем на датасеті Print Attack

Один з головних виявлених недоліків анти-спуфінг систем заснованих на нейромережах, описаних раніше, полягає у тому, що під час спроб тестування нейромережі, навченої на одному наборі даних, на іншому датасеті, то результати виявилися набагато гіршими. В одному з досліджень 2015 року, був проведений аналіз використання різних анти-спуфінг систем на базі нейромереж з різними датасетами для навчання та тестування де для визначення справжності пред'явленого зображення використовувалася методика аналізу текстур зображення [23]. Результати дослідження зведенні то таблиці 1.3, де TPR – це відсоток правильно прийнятих та правильно відхилених спроб автентифікації (коефіцієнт істинно позитивних виявлень).

Таблиця 1.3 – порівняння анти-спуфінг систем на базі нейромереж

Метод	Train	Test	TPR FAR=0.1	TPR FAR=0.01
LBP+SVM	Idiap	Idiap	94.5	57.3
		MSU	14.1	0
	MSU	MSU	87.0	31.5
		Idiap	20.9	2.9
		Idiap	92.1	67.0

DoG-LBP +SVM	Idiap	MSU	19.5	0.2
	MSU	MSU	77.3	21.4
		Idiap	23.6	3.8
IDA+SVM	Idiap	Idiap	92.2	87.9
		MSU	75.5	29.8
	MSU	MSU	94.7	82.9
		Idiap	73.7	38.6

Як ми бачимо виходить наступна ситуація: алгоритм, натренований на даних Idiap, а застосований на датасеті MSU, дасть коефіцієнт істинно позитивних виявлень 14,1%, а, якщо зробити навпаки (навчити на MSU, а перевірити – на Idiap), то вдасться визначити вже 20,9 %. Для інших поєднань у дослідженні ситуація погіршується ще більше, наприклад, якщо натренувати алгоритм MSU, а перевірити – на CASIA, то TPR становитиме 10,8%. Це означає, що до атакуючих було помилково зараховано переважну кількість чесних користувачів, що призводить до небажаних наслідків. Ситуацію не змогло переламати навіть cross-database навчання, що начебто здається цілком розумним виходом із становища.

У ще одній статті опублікованій у 2016 року, були продемонстровані результати які свідчать, що навіть за досить складних комбінацій обробки та виділення таких надійних ознак, як моргання та текстур зображення, результати на незнайомих наборах даних можуть бути вкрай незадовільними, наприклад, показник HTER збільшувався у два рази на різних датасетах [24]. В кінцевому результаті стає цілком очевидно, що запропонованих способів відчайдушно не вистачає для узагальнення результатів анти-спуфінг систем на базі нейромереж.

Також були запропоновані рішення, що ґрунтуються на нестандартних підходах. Наприклад, було запропоновано скористатися методом дистанційної фотоплетизмографії (rPPG – remote photoplethysmography), що дозволяє виявити биття пульсу людини з відеозображення (рис. 1.15). Ідея полягає в тому, що при попаданні світла на живе обличчя людини частина світла відіб'ється, частина-розсіється, а частина – поглинається шкірою та тканинами обличчя [25]. При цьому картина буде різною залежно від ступеня заповнення тканин кров'ю. Таким чином, можна відстежити пульсацію крові в судинах обличчя та, відповідно, виявити пульс.

Звичайно, якщо закрити обличчя маскою або пред'явити екран телефону чи роздруковане зображення, ніякої пульсації виявити не вдасться. На цьому принципі Лю С. із співавторами запропонували розбивати зображення обличчя на ділянки, детектувати пульс методом дистанційної фотоплетизмографії, попарно порівнювати різні ділянки для підрахунку пульсу та будувати карти з метою виявлення наявності чи відсутності маски, а також порівняння пульсу на різних ділянках обличчя [25].

Результати роботи надали значення HTER ~10%, підтвердивши придатність методу. Є ще кілька робіт, що підтверджують перспективність цього підходу :

— J. H.-Ortega et al. Time Analysis of Pulsebased Face Anti-Spoofing in Visible and NIR – 2018 рік;
— X. Li. та ін. Generalized face anti-spoofing by detecting pulse from face videos – 2016 рік;
— H. E. Tasli et al. Remote PPG based on vital sign measurement using adaptive facial regions – 2014 рік.

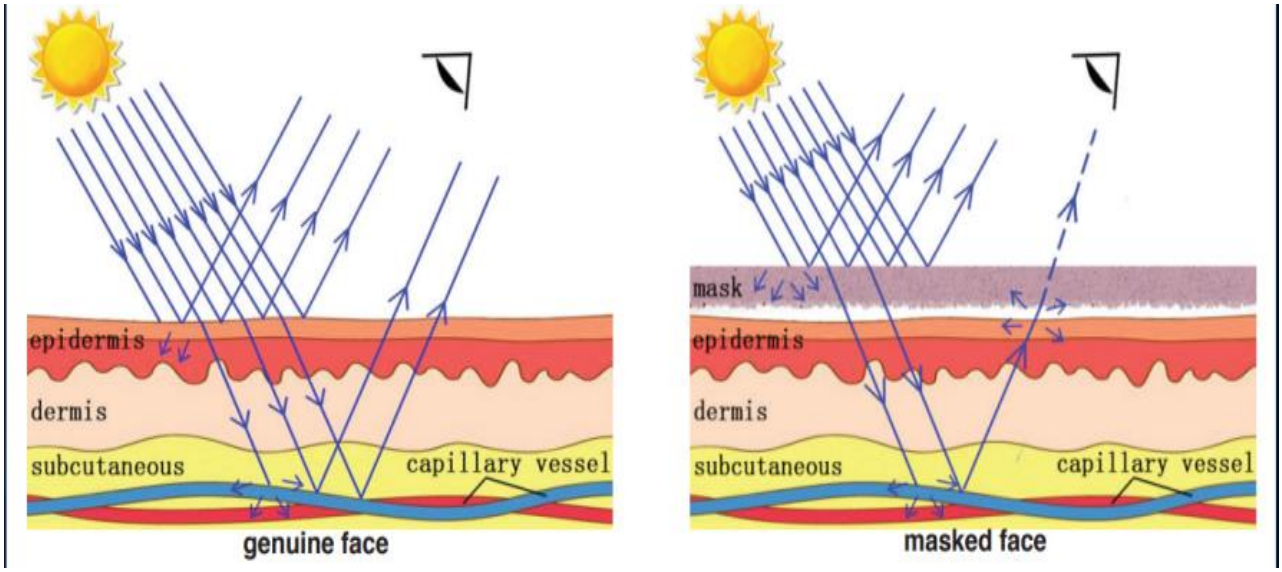


Рисунок 1.15 – Розсіювання світла для справжнього обличчя та маски

Треб зазначити що до 2017 року область face anti-spoofing містила незначну кількість робіт. Тільки з 2019 року почала з’являтися вагома кількість робіт з анти-спуфінг систем, що пов'язано з агресивним просуванням мобільних технологій ідентифікації по обличчю людини, насамперед компанією Samsung та Apple. Крім того, технологіями розпізнавання по обличчю зацікавилися банки. У галузь прийшло багато нових людей, що дає змогу сподіватися на швидкий прогрес. Але поки що, незважаючи на гарні результати публікацій, узагальнююча здатність існуючих алгоритмів виявляти спуфінг-атаки на системи ідентифікації та авторизації залишається дуже слабкою і не дозволяє говорити про будь-яку придатність до практичного використання за будь-яких умов.

1.5 Порівняльна характеристика існуючих анти-спуфінг систем

Враховуючі усі системи, які були наведені у попередньому підрозділі, була утворена таблиця 1.4 для порівняння усіх переваг та недоліків запропонованих рішень. Таблиця містить у собі дані про порівняння систем за можливістю протидіяти наявними видами спуфінг-атак, показнику ефективності систем HTER, можливості приймати як одне фото так і відеопоток, а також включає інформацію про додаткові обмаження чи переваги тої чи іншої системи.

Таблиця 1.4 – Порівняння існуючих анти-спуфінг систем

Характеристика системи / Назва системи	Виявляє Printed attack	Виявляє Replay attack	Виявляє Mask attack	Потребує додаткових дій користувача	Приймає відеопотік та одне зображення	HTER (%)	Додаткові недоліки
Виявлення моргань очима + LBP	Так	Ні	Ні	Так	Тільки відеопотік	15%	Повільна робота, так як іноді системі

							доводитьс я чекати на моргання користува ча
Аналіз кольоровог о простору зображення YC_bC_r	Так	Так	Так	Ні	Відеопотік та зображен ня	3 %	
Виявлення муара + LBP	Ні	Так	Ні	Ні	Відеопотік та зображен ня	6%	
Контекстно орієнтована система виявлення руху обличчя з рухом заднього фону зображення	Так	Так	Ні	Так	Тільки відеопотік	10%	
Ейлерівське посилення відео + HOOF гістограми напрямів оптичного потоку	Так	Так	Ні	Ні	Тільки відеопотік	до 1.25%	
Система аналізу карт глибини зображення	Так	Так	Ні	Ні	Відеопоті к та зображен ня	3,78%	Сильно залежить від фону та

							освітленості обличчя
rPPG система виявлення биття пульсу на обличчі	Так	Так	Так	Ні	Тільки відеопотік	10%	Сильно залежить від освітленості обличчя

1.6 Формування функціональних вимог

У даному підрозділі описані функції системи, які повинні бути реалізованими під час розробки рішення, щоб користувачі могли виконувати свої завдання. Так як 3D-спуфінг з атаками на 3D-сканери поки що не є великою проблемою, більш поширеним є 2D-спуфінг і тому саме на протидію йому буде спрямована система, в першу чергу це покладає обов'язок на виявлення та запобігання презентаційним атакам.

Враховуючі дані про характеристики існуючих систем з таблиці 1.4 та аналізуючи їх недоліки можна зазначити, що система повинна вміти боротися з двовимірними атаками, статичними або динамічними, використовуючи тільки одне зображення, або відеопотік. Система повинна намагатися досягти максимальної точності за мінімальний час, а також забезпечити зручність для користувача. Зазвичай для надійності моделі потрібна група дій. Ці дії можуть включати посмішку, вираження емоцій, таких як смуток, здивування або рухи головою. Ці взаємодії потребують значного часу та часто – незручні для користувачів. Таким чином, захист від підробки для певного виду атак повинен залучати якомога менше дій користувача, щоб досягти максимально позитивного досвіду користування системою. Усю вищевикладену інформацію зведемо до чітких вимог. Система повинна:

- Система самостійно повинна визначати обличчя людини з віодеопотоку;
- Система повинна виявляти тип атак Printed attack;
- Система повинна виявляти тип атак Replay attack;

- Система повинна виявляти тип атак Mask attack;
- Працювати без взаємодії з користувачем (без фізичних рухів користувача);
- Аналізувати як одне зображення так і відеопотік;
- Пристосовуватися до оточуючого середовища користувача;
- Бути кросс-платформеною та підтримувати роботи на операційних системах Windows, Linux, MacOS;
- Вміти взаємодіяти с веб-камерами ноутбуків та стаціонарних комп'ютерів;
- Система повинна сповіщати користувача про виникнення помилок під час своєї роботи.

1.7 Формування нефункціональних вимог

Як було зазначено у розділі 1 головними показниками ефективності систем проти спуфінг-атак є HTER , FAR і FRR. FAR і FRR, які є звичайними для біометричної перевірки, також використовуються для захисту від підробки обличчя. Специфіка завдання визначає показники, які використовуються для інтерпретації помилок. Якщо метою є відловлювати кожну атаку, FAR слід мінімізувати. Якщо зручність має вищий пріоритет, FRR стає важливішим. Якщо ми візуалізуємо ймовірності, ми побачимо, що 2 криві перетинаються в певній точці (рис. 1.16). Ця точка — рівна частота помилок (ERR – Equal error rate), і вона допомагає вибрати найкраще порогове значення для прийняття рішень. Залежно від вимог безпеки ми можемо перемістити порогове значення вліво або вправо, надаючи перевагу FAR або FRR. У нашому випадку зручність роботи користувача була більш важливою, тому ми відкоригували порогове значення, щоб мінімізувати FRR.

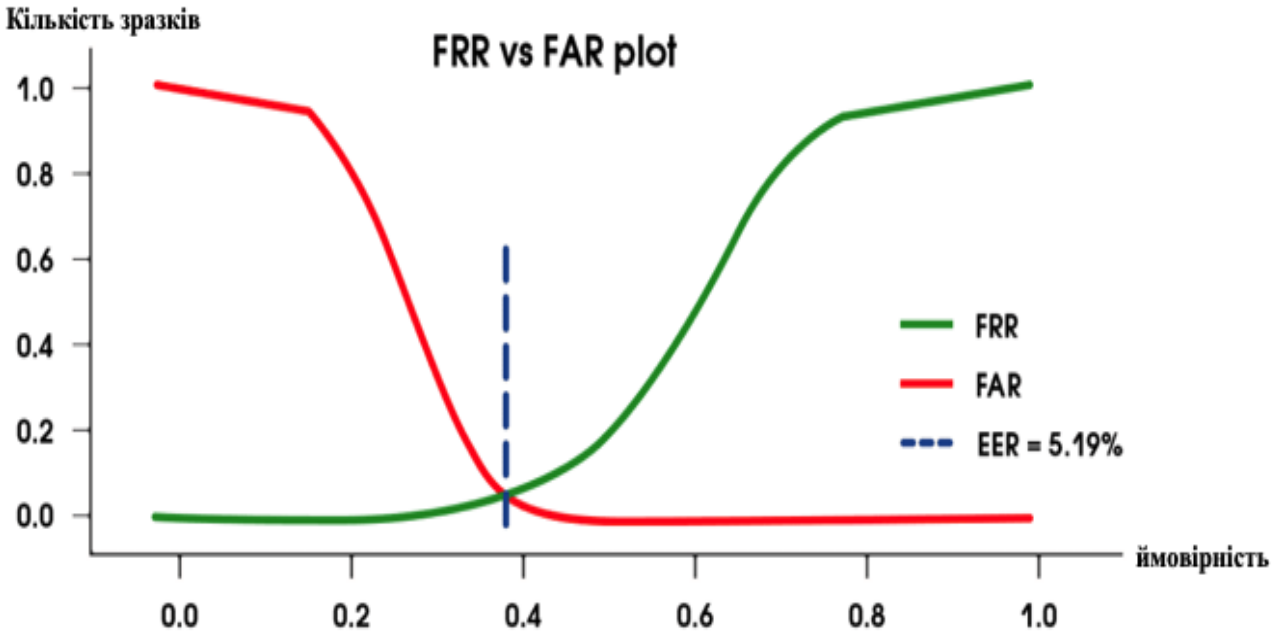


Рисунок 1.16 – Взаємозв’язок FAR та FRR

Пріоритетним залишається використання системи в різних інформаційних середовищах, тому система повинна мати універсальний API для взаємодії з іншими системами в неконтрольованому розгортанні. В результаті отримуємо наступні чіткі вимоги:

- Показник HTER < 1 % ;
- Система, яка відповідає раніше переліченим вимогам, повинна легко інтегруватися з існуючими системами розпізнавання обличчя за допомогою REST API;
- Для взаємодії системи з базою даних має підтримуватися хоча б одна ORM бібліотека;
- Зберігати вдалі та невдалі спроби автентифікації до бази даних.

1.8 Висновки до розділу

У даному розділі був проведений аналіз загроз на системи біометричної ідеетифікації з використанням обличчя людини. Були визначенні наступні види атак, що використовуються шахраями на сьогоднішній день: Replay attack, Printed attack та

Mask Attack. Встановлено, що найтяжчим видом атак для виявлення є саме Replay attack, тому прирозгляді існуючих рішень в першу чергу зверталась увага на підходи, що вміють виявляти даний тип атак.

Були виокремлені основні методи виялення спуфінг, а саме: програмні статичні, програмні динамічні та апартані методи виявлення. Для кожного метода розглянуті переваги та недоліки, так найкращим метод визначено програмний статичний метод. Встановлені основні показники ефективності для анти-спуфінг систем, такі як: HTER, EER, FAR, FRR.

Проаналізовані існуючі анти-спуфінг системи, їх підходи до визначення спуфінг-атак, здатність протидіяти різним видам атак, показник HTER, набори датасетів на яких були проведенні тестування систем. Складена таблиця з порівняльною характеристикою існуючих систем, так найкращим методом, який був обраний за основу для реалізації власного рішення став метод аналізу кольорового простору зображення YC_rC_b. Також, на основі порівняльних даних існуючих систем, були складені основні вимоги до розроблювальної системи, так щоб система немала недоліків своїх конкурентів.

2 ЗАПРОПОНОВАНЕ РІШЕННЯ ТА РОЗРОБКА СТРУКТУРНОЇ СХЕМИ

2.1 Запропоноване рішення для виявлення спуфінг-атак

Так як система розроблюється для цивільних, а не військових потреб, то одним з головних пріоритетів при розробці є простота використання системи, і як було зазначено у першому розділі, захист від підробки для певного виду атак повинен залучати якомога менше дій користувача, щоб досягти максимально позитивного досвіду користування системою. Тому було прийнято рішення використовувати саме статичні методи виявлення спуфінг-атак, які не вимагають від користувача певних дій.

У статичних методах щоб ідентифікувати спуфінг-атаку, більша частина дослідників [27] використовували LBP – локальні бінарні шаблони – припускаючи, що текстура зображення 2D спуфінг-атаки відрізняється від реальної текстури

обличчя людини. Іншим підходом, менш дослідженим підходом, є використання різних колірних просторів/моделей [26]. Після аналізу характеристик існуючих систем у таблиці 1.4 та враховуючі усі вимоги поставленні до систем, у даній роботі пропонується підхід проти спуфінг-атак, заснований на перетвореннях колірного простору YC_rC_b і CIE L*u*v*. Даний підхід зосереджений на проблемі виявлення printed та replay attack але також підходить для виявлення mask attack. Підхід з використанням кольорових просторів YC_rC_b і CIE L*u*v* дозволяє аналізувати як одне зображення так і відеопотік, забезпечить HTER ~1% та є стійким до зміни фону зображення та освітлення обличчя.

Як згадувалося у першому розділі, кілька дослідників використовували підходи на основі кольорів для виявлення підробки обличчя [26]. Незважаючи на те, що колірний простір RGB є найбільш використовуваним у пристроях збору відео та фото, у роботі Лі Фенг [28], було зазначено, що це не найкращий колірний простір для виявлення спуфінг-атак через кореляцію між червоним, зеленим і синім, які перешкоджають розділенню інформації про яскравість і кольоровість.

Схема захисту від спуфінгу, запропонована в даній роботі, використовує переваги застосування двох різних просторів кольорів: YC_rC_b і CIE L*u*v*, дивіться Додаток А. Для кожного вхідного кадру або зображення в колірному просторі RGB виконується перетворення в YC_rC_b (блок 5 Додаток А) і CIE L*u*v* (блок 6 Додаток А). Потім обчислюється шість гістограм, по три гістограми на кожний із двох просторів (блок 7,8 Додаток А). Далі шість гістограм об’єднуються у вектор ознак (блок 9 Додаток А) FV = (Y, C_r, C_b, L, u, v) розміром 1536, шість нормалізованих гістограм у діапазоні 0–255, які служать вхідними даними для класифікатора додаткових дерев ETC (Extra Trees Classifier). На фінальному етапі роботи алгоритму, класифікатор вирішує, чи відповідає вектор вхідних ознак зображенню справжнього зразка чи це спуфінг-атака (блок 10 Додаток А).

Окремо треба виділити класифікатора додаткових дерев ETC. Extra Trees використовує весь набір даних для навчання дерев рішень. Таким чином, щоб забезпечити достатню різницю між окремими деревами рішень, він випадково

вибирає значення, за якими потрібно розділити функцію та створити дочірні вузли, на відміну від інших алгоритмів машинного навчання, з використанням дерев, які використовують алгоритм для жадібного пошуку та вибору значення, за яким потрібно розділити функцію. ExtraTrees мають цінність, особливо коли є проблема з обмеженою ефективністю апаратних ресурсів. Зокрема, коли будуються моделі, які мають значні етапи попереднього моделювання розробки/вибору функцій, і вартість обчислень є проблемою, ExtraTrees буде гарним вибором порівняно з іншими моделями на основі ансамблю дерев.

Запропонований підхід базується на очевидній ідеї, що оскільки два зображення одного об’єкта, одне — реального об’єкта, а інше — зображення атаки, мають різні візуальні характеристики, ця інформація може бути вилучена за допомогою відповідних кольорових гістограм. Інші роботи [29, 30, 31] продемонстрували ефективність комбінування різних колірних просторів для виявлення атаки підробки обличчя. У даній роботі тільки простори YC_rC_b і CIE L*u*v* використовуються для навчання класифікатора ETC та ідентифікації атаки підробки зображення. Концепція цього підходу виникла завдяки спостереженню за деякими регулярними відмінностями в кольорових гістограмах YC_rC_b і CIE L*u*v* між двома зображеннями [31] (зображення справжнього зразка обличчя та друкованого нападу) з одного фото обличчя людини, як показано на рисунку 2.1-2.2 та рисунку 2.2-2.3.

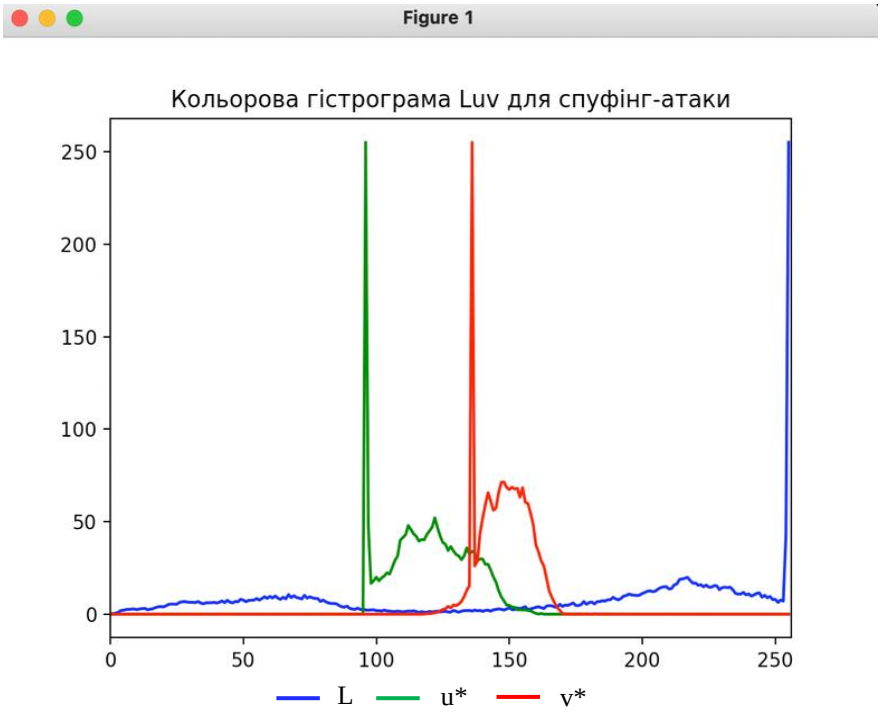


Рисунок 2.1 – Гістограма L*u*v* зображення спуфінг-атаки

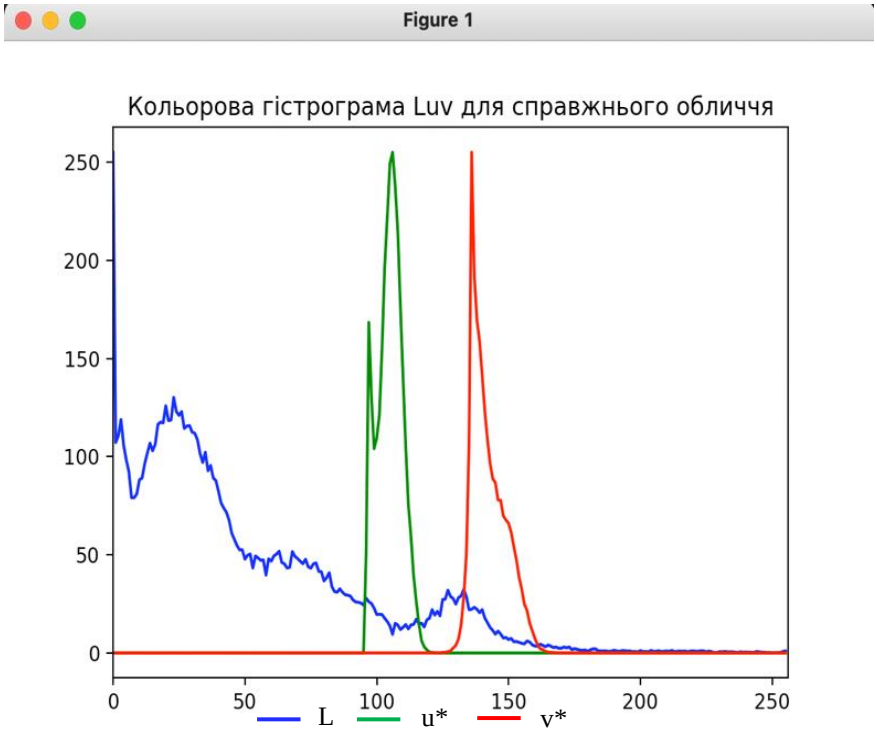


Рисунок 2.2 – Гістограма L*u*v* зображення справжнього обличчя

Можна побачити на рисунках 2.1-2.4, що інформація про яскравість зображень (канал L та Y) в обох колірних просторах має майже однакову форму на справжніх фотографіях чого не відбувається на зображеннях з printed attack.

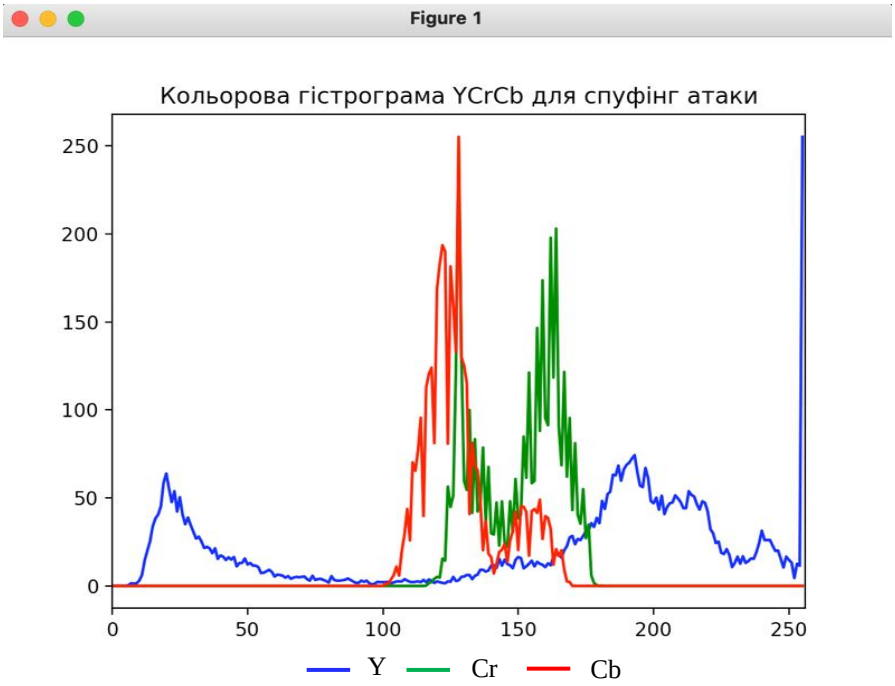


Рисунок 2.3 – Гістограма YC_rC_b зображення спуфінг-атаки

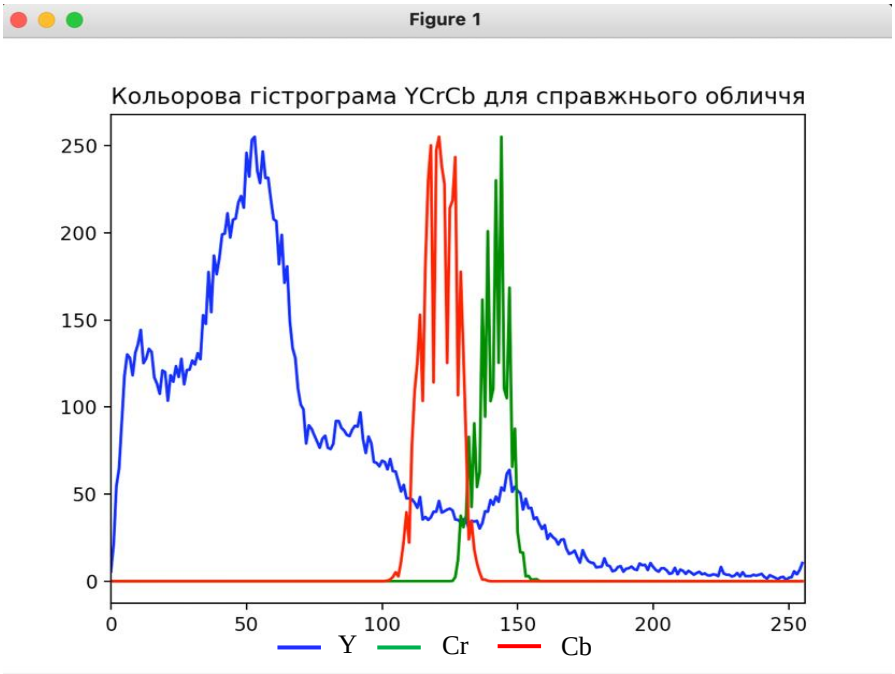


Рисунок 2.4 – Гістограма YC_rC_b зображення справжнього обличчя

Розглянемо більш детально простори кольорів, які застосовуються у даній роботі, а також конвертації зображення з RGB простору. Колірний простір CIE L*u*v* було введено Міжнародною комісією з освітлення (також відомою як CIE від її французької назви, Commission Internationale de lEclairage) у 1976 році [32]. Перетворення RGB в CIE L*u*v* описано в формулах (2.1), (2.2) і (2.3). Рівняння (2.4),

(2.5) і (2.6) є проміжними кроками та нормалізаціями, необхідними для цього перетворення простору кольорів.

$$L = \begin{cases} 116 * \sqrt[3]{Y} \text{ для } Y > 0.008856 \\ 903.3 * Y \text{ для } Y \leq 0.008856 \end{cases}; \quad (2.1)$$

$$u = 13 * L * \left(u` - u_n\right), \text{де } u_n = 0.19793943; \quad (2.2)$$

$$v = 13 * L * \left(v` - v_n\right), \text{де } v_n = 0.46831096; \quad (2.3)$$

$$v` = 9 * \frac{X}{X + 15 * Y + Z + 3}; \quad (2.4)$$

$$u` = 4 * \frac{X}{X + 15 * Y + Z + 3}; \quad (2.5)$$

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412 & 0.357 & 0.1804 \\ 0.212 & 0.715 & 0.0721 \\ 0.0193 & 0.119 & 0.9502 \end{bmatrix} * \begin{bmatrix} R \\ G \\ B \end{bmatrix}, \quad (2.6)$$

де R – це яскравість червоного кольору відповідного пікселя зображення, що вимірюється від 0 до 255, G – це яскравість зеленого кольору та B – це яскравість синього кольору. Щоб перетворити L, u і v у 8-бітове зображення, ці компоненти потрібно нормалізувати. Таким чином, L = 255 × L/100, u = 255/(354 × (u + 134)) і v = 255/(262 × (v + 140)).

Стосовно YC_rC_b , то ця модель в основному використовується для стиснення зображення для телетрансляцій і була визначена ITU – Міжнародним союзом електрозв'язку в стандарті ITU-R BT.601 [33]. Канал Y відповідає яскравості або компоненту яскравості, отриманому з RGB після гамма-корекції, Cr визначає, наскільки яскравість червоного кольору відрізняється від яскравості Y, а Cb позначає, наскільки яскравість синього кольору відрізняється від яскравості Y. Перетворення представлені у формулах 2.7- 2.9.

$$Y = 0.299 * R + 0.587 * G + 0.114 * B;$$

$$C_r = (R - Y) * 0.713 + \delta;$$

(3.9)

(2.8)

$$C_b = (B - Y) * 0.564 + \delta,$$ (2.9)

де для 8-бітного представлення зображення δ = 128.

2.2 Вибір алгоритму для виокремлення обличчя людини з відеопотоку

Система описана у даній роботі повинна не тільки виявляти спуфінг-атаки, а також і детектувати обличчя людини з відеопотоку. Були розглянуті різні алгоритми та рішення для виявлення обличчя людини на відео і був обраний алгоритм, що застосовує каскади Хаара. Рішення обумовлене тим, що імплементація даного підходу не потребує великої кількості додаткових бібліотек коду та є відносно простою, а також даний підхід має високу швидкість роботи [34]. Ще однією значимою перавагою методу у порівнянні з іншими є його стійкість до зміни освітлення.

Суть запропонованого підходу полягає у тому, що ми маємо каскади Хаара – це набір шаблонів Хаара, які створюються в процесі машинного навчання для деякого об’єкта на зображенні. Шаблони представляють собою розділення визначеної прямокутної області на набори різних прямокутних підобластей чорного чи білого кольору (рис. 2.2). Кожен шаблон є певним представленням певної частини об’єкта який ми намагаємось розпізнати (рис. 2.3). Наприклад, область під носом буде темніша, ніж область яка знаходиться над ним, так само як область рота буде більш темна ніж область лоба. Чим ми більше використовуємо різних шаблонів, тим точніше потім можна класифікувати об'єкт.

Ідея алгоритму виявлення обличчя полягає у тому, що ми послідовно накладаємо шаблони Хаара на площу визначеного зображення, а далі для кожного шаблона ми рахуємо значення за формулою (2.10), що являє собою різн (2.10) м яскравості пікселей.

$$F = X - Y,$$

де X – сумарне значення яскравості пікселей які перебувають у білій області примітиву, а Y – сумарне значення яскравості пікселей які перебувають у чорній області примітиву. При цьому кожен шаблон накладається кілька разів на одні і ті самі ділянки, але при цьому розмір шаблону кожен раз змінюється. Коли значення F перевищує завчасно встановлене лімітне значення для даного шаблону, то говорять, що шаблон є вагомим, тобто певна частина зображення співвідносить з даним шаблоном, і переходять до перевірки іншого примітиву із каскаду.

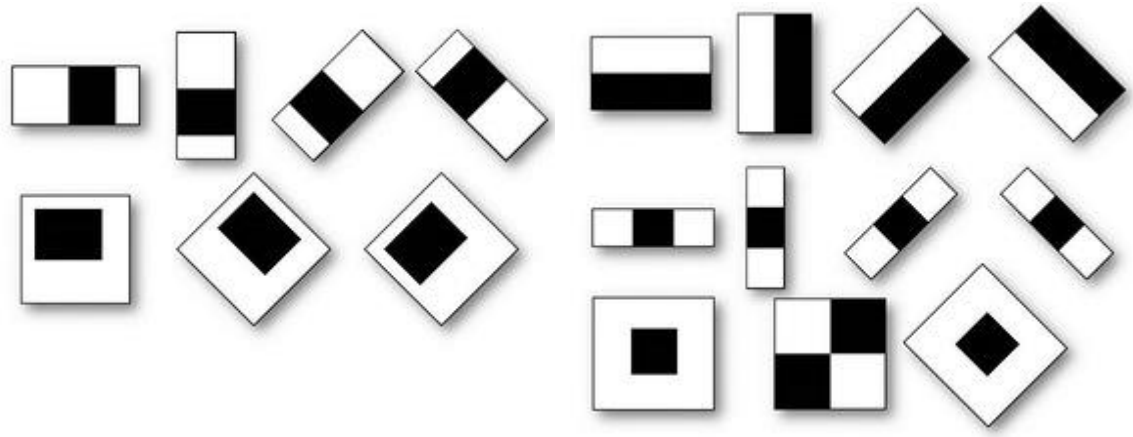


Рисунок 2.5 – Приклад шаблону Хаара

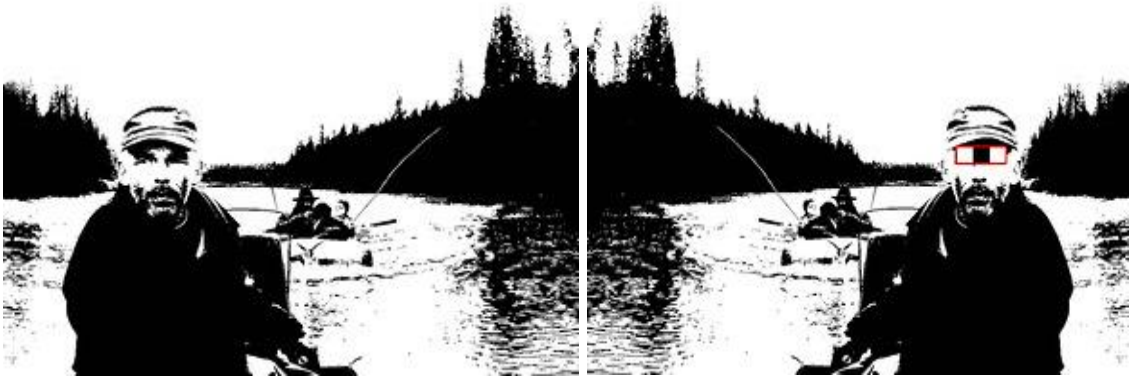


Рисунок. 2.6 – Приклад розпізнавання очей людини шаблоном Хаара

Як вже було зазначено, однією з головних переваг каскадів Хаара є те, що пошук заданого об’єкта здійснюється дуже швидко і в першу чергу це можливо за рахунок інтегрального представлення зображення [34]. Інтегральне зображення обчислюється за формулою (2.11) як:

(2.11)

$$M(x,y) = \sum_{i=0,j=0}^{i\leq x, j\leq y} I(i,j),$$

де x та y – це довжина та ширина зображення в пікселях відповідно, I – це матриця в якої кожен елемент містить значення яскравості відповідного пікселя зображення. Виходить що інтегральне значення для кожного окремого пікселя є сумою всіх значень яскравості пікселей, які знаходяться зверху і зліва від даного пікселя, і також до цієї суми додається особисте значення яскравості поточного пікселя. Даний підхід дає нам можливість підвищити швидкість підрахунку суми значень яскравості пікселей. Розглянемо приклад (рис. 2.7):

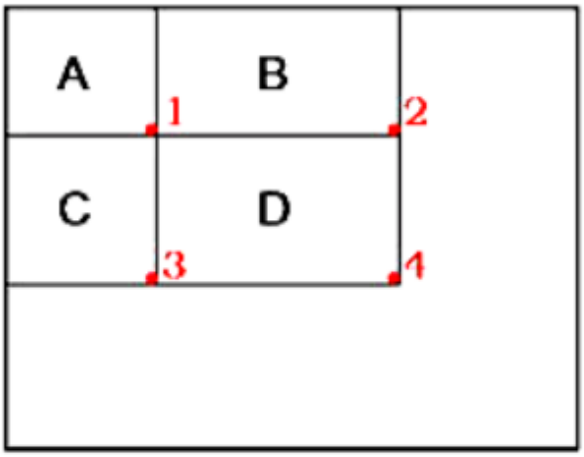


Рисунок 2.7 – Приклад для інтегрального зображення

Для того щоб вирахувати суму яскравості пікселей зображення на прямокутнику D на не потрібно за допомогою циклу перевіряти усі пікселі цієї ділянки, достатньо скористатися інтегральним зображенням і тоді значення яке ми шукаємо можна порахувати як значення у точці чотири додати значення у точці один і відняти значення у точках два і три, виходить: D = ABCD + A – CA – BA. На практиці ми отримаємо всього чотири звернення до пам’яті і три математичні операції.

2.3 Розробка структурної схеми

Враховуючі усі функціональні та нефункціональні вимоги до системи, та після визначення методу на основі якого буде реалізована система, була розроблена структурна схема системи виявлення спуфінг-атак, яка представлена у Додатку В. Так система у своєму складі має наступні блоки (додаток В) для задоволення усіх потреб архітектури:

1) Блок взаємодії з сенсором – даний блок необхідний для доступу до відеопотоку, який надається у нашому випадку веб-камерою комп'ютера, ноутбука. Також блок виокремлює окремий кадр з відеопотоку;
2) Конвертор вхідного зображення у чорно-біле – даний блок виконує перетворення вхідного зображення у чорно-білий кольровий простір, так як цього передбачає алгоритм виявлення обличчя з каскадами Хаара.
3) Конвертор інтегрального зображення – даний блок виконує перетворення чорно-білого зображення у інтегральне зображення (див. формула 2.11). Після конвертації зображення передається до блоку з каскажами Хаара;
4) Блок з каскадами Хаара –завдання даного блоку полягає у знаходженні позиції обличчя людини на вхідному зображенні;
5) Конвертор зображення у YCrCb простір – виконує пперетворення зображення обличчя з RGB простору у YCrCb кольровий простір;
6) Конвертор зображення у L*u*v* простір – виконує пперетворення зображення обличчя з RGB простору у L*u*v* кольровий простір;
7) Конкатенатор векторів ознак – даний блок утворює шість векторів для зображення, де кожен вектор відповідає за той чи інший канал двох кольорових просторів YC_rC_b і L*u*v*(по три канали на кожен простір), і кожен елемент вектора містить значення відповідного каналу для відповідного пікселя. Після утворення шести вектрорів усі вони об'єднуються у один вектор ознак, що потім подається на вхід ETC класифікатора;
8) ETC класифікатор – даний блок приймає на вхід об'єднаний вектор значень усіх каналів кольорових просторів YC_rC_b і CIE L*u*v*, подає його на вхід ETC класифікатора та потім надає вихідне значення у діапазоні від 0 до 1, де 1 – це 100% наявність спуфінг-атаки;

9) Аналізатор вихідних даних ETC класифікатора – блок отримає вихідне значення ETC класифікатора і порівнює його з пороговим значенням, що зберігається в БД, якщо значення класифікатору вище порогового значення, то дане зображення позначається, як спуфінг-атака;
10) Блок взаємодії з БД – блок зберігає дані про вдалі та невдалі спроби виявлення спуфінг-атак, а також передає порогове значення для аналізатор вихідних даних ETC класифікатора;
11) Обробник REST запитів – даний блок надає REST API інтерфейс для отримання зображення обличчя людини з стороннього ресурсу за допомогою HTTP протоколу;
12) Конветрор REST запиту – даний блок відповідальний за перетворення вмісту REST запиту до файлу зображення, що буде зберігатися у сховищі даних ЕОМ де працює система.

2.4 Висновки до розділу

В розділі визначено алгоритм за якими система буде виявляти обличчя людини з відеопотоку, а також безпосередньо алгоритм виявлення спуфінг-атаки на зображенні. Так запропоноване рішення виявлення спуфінг-атак відноситься до статичних методів та засноване на перетворені зображення з кольорового простору RGB до кольорових просторів YC_rC_b і CIE L*u*v* та аналізу ознак за допомогою ETC класифікатора. Вибір даного рішення обумовлений успішними результатами схожих анти-спуфінг систем з використанням кольорового простору YC_rC_b, також його спроможність до ефективної роботи підтверджена експериментальним шляхом під час розгляду гістограм зображень з наявними спуфінг-атаками та без них. Для виокремлення обличчя людини з відеопотоку обраний алгоритм з використанням примітивів Хаара, що забезпечує високу швидкість роботи та є стійким до зсіни освітлення обличчя людини.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

3.1 Мова програмування та засоби розробки програмної системи

Серед наявних мов програмування вибір був зупинений на мові програмування Python. Дане рішення обумовлене декількома факторами, в першу чергу тим, що дана мова програмування дозволяє задовільнити усі нефункціональні вимоги, які висуваються до системи, а саме: Python є кросплатформеною мовою програмування [34]; мова має значну кількість ORM фреймворків для взаємодії з БД (мінімум 6) [35]; наявна велика кількість WEB-фреймворків для розробки REST API, які також інтегруються з ORM фреймворками. Данна мова програмування має довгу історію розробки починаючі з 1991 року і також стабільну так активну підтримку, останнє оновлення відбулося 24 жовтня 2022 року [36]. Однією з найбільших сильних сторін Python є його обширна стандартна бібліотека, яка надає готовий набір модулів для більшості завдань програмування, включаючи роботу з файлами введення/виведення даних, систему взаємодії з інтернет мережею, розбір даних і маніпуляції, потоки, профілювання, модульне тестування, самоаналіз, обробка XML та JSON і багато іншого [37].

Однією з головних переваг мови Python для розробки запропонованої системи є те, що дана мова має значну кількість фреймворків та бібліотек для роботи з машинним навчанням, що в свою чергу допоможе нам при розробці SVM та ETC класифікатора. Жан Франсуа Пюже з відділу машинного навчання IBM висловив свою думку про те, що Python є найпопулярнішою мовою для штучного інтелекту та машинного навчання [38], і ґрунтувався на результатах пошуку тенденцій на веб-сайті indeed.com. Великий вибір бібліотек є однією з головних причин, чому Python є найпопулярнішою мовою програмування, яка використовується для машинного навчання. Бібліотека — це модуль або група модулів, опублікованих різними джерелами, як-от PyPi, які містять попередньо написаний фрагмент коду, який дозволяє користувачам отримувати певні функції або виконувати різні дії. ML вимагає безперервної обробки даних, а бібліотеки Python дозволяють отримувати

доступ, обробляти та перетворювати дані. Ось деякі з найпоширеніших бібліотек, які можна використовувати для ML та AI:

- Scikit-learn – для обробки базових алгоритмів машинного навчання, таких як кластеризація, лінійна та логістична регресії, регресія, класифікація та інші;
- Pandas – для високорівневих структур даних і аналізу. Бібліотека дозволяє об’єднувати та фільтрувати дані, а також збирати їх з інших зовнішніх джерел, наприклад, Excel;
- Keras – для глибокого навчання. Вона дозволяє швидко обчислювати та створювати прототипи, оскільки використовує GPU на додаток до CPU комп’ютера;
- TensorFlow – для роботи з глибоким навчанням шляхом налаштування, навчання та використання штучних нейронних мереж із масивними наборами даних;
- Matplotlib – для створення двовимірних графіків, гістограм, діаграм та інших форм візуалізації;
- NLTK – для роботи з комп’ютерною лінгвістикою, розпізнаванням і обробкою природної мови;
- Scikit – бібліотека з малим використанням пам’яті для обробки зображень;
- StatsModels – для статистичних алгоритмів і дослідження даних.

Також Python для машинного навчання — чудовий вибір, оскільки ця мова дуже гнучка:

- Він пропонує можливість вибрати використання ООП або скриптів;
- Також немає необхідності перекомпілювати вихідний код, розробники можуть внести будь-які зміни та швидко побачити результати.
- Програмісти можуть поєднувати Python та інші мови для досягнення своїх цілей.

Крім того, гнучкість Python дозволяє розробникам вибирати стилі програмування, які їм цілком зручні, або навіть поєднувати ці стилі для вирішення різних типів проблем найефективнішим способом. Наприклад, імперативний стиль складається з команд,

які описують, як комп’ютер має виконувати ці команди. За допомогою цього стилю ви визначаєте послідовність обчислень, які відбуваються як зміна стану програми. Функціональний стиль також називають декларативним, оскільки він визначає, які операції слід виконувати. Він не враховує стан програми, порівняно з імперативним стилем, він оголошує твердження у формі математичних рівнянь. Об'єктно-орієнтований стиль базується на двох концепціях: клас і об'єкт, де подібні об'єкти утворюють класи. Цей стиль не повністю підтримується Python, оскільки він не може повністю виконати інкапсуляцію, але розробники все одно можуть використовувати цей стиль до певної міри. Процедурний стиль є найпоширенішим серед початківців, оскільки він виконує завдання у форматі крок за кроком. Фактор гнучкості зменшує ймовірність помилок, оскільки програмісти мають можливість взяти ситуацію під контроль і працювати в комфортних умовах.

3.2 Вибір необхідних фреймворків та бібліотек

Одне з головних завдань, яке повинна виконувати система, це – виявлення обличчя людини з відеопотоку або на одному зображенні. Як було зазначено у другому розділі для вирішення даного завдання запропоновано використовувати алгоритм на основі каскадів Хаара, тому одна із бібліотек, що буде застосовуватися при розробці системи – це OpenCV. OpenCV є найпопулярнішою бібліотекою комп'ютерного зору [39]. Спочатку OpenCV написана мовою C/C++, тепер вона забезпечує прив’язки для Python (рис. 3.1). OpenCV використовує алгоритми машинного навчання для пошуку обличь на зображенні. Оскільки обличчя дуже складні для пошуку, немає жодного простого тесту, який скаже вам, знайдено обличчя чи ні. Натомість є тисячі дрібних шаблонів і особливостей, які повинні бути узгоджені, в результаті у вас можуть бути мільйони обчислень, які призведуть до зупинки комп’ютера. Щоб обійти це, в OpenCV вбудовані методи для роботи з каскадами Хаара. OpenCV розбиває проблему виявлення обличь на кілька етапів. Для кожного блоку виконується дуже приблизний і швидкий тест. Якщо тест проходить, виконується дещо детальніший тест і так далі. Алгоритм може мати від 30 до 50 таких

етапів або каскадів, і він виявить обличчя, лише якщо всі етапи пройдено. Перевагою OpenCV є те, що більша частина зображення повертатиме негативний результат розпізнавання на перших кількох етапах, а це означає, що алгоритм не витрачатиме час на перевірку всіх функцій на ньому. Замість того, щоб займати години, розпізнавання обличчя тепер можна виконувати в режимі реального часу.

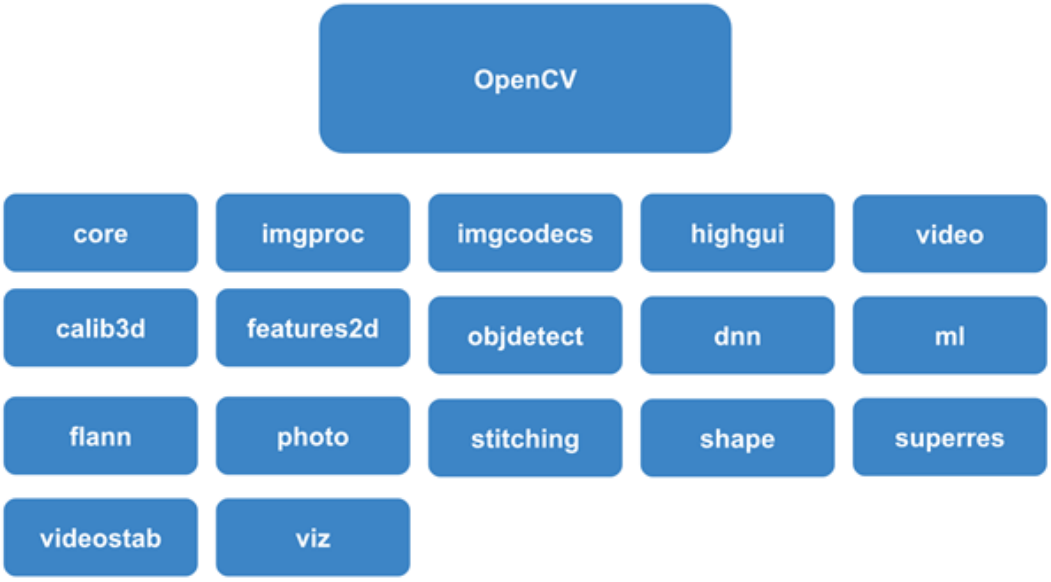


Рисунок 3.1 – Модулі OpenCV

Ще одна з причин вибору OpenCV полягає у тому, що дана бібліотека має функціонал для перетворення зображення із кольорового простору RGB в простори YCrCb та L*u*v*, а ще бібліотека надає можливість побудови векторів гістограм для заданих зображень, що значно пришвидшить процес розробки системи. Також OpenCV має функціонал для роботи з веб-камерою комп'ютера, що допоможе задовільнити одну з функціональних вимог до системи. OpenCV захоплює кожен кадр із веб-камери, а потім ви можете розпізнавати обличчя, обробляючи кожен кадр.

Далі розглянемо фреймворк який допоможе нам реалізувати REST API. Так я REST API необхідний нам тільки для того, щоб прийняти одне зображення, а потім, проаналізувавши його, відправити відповідь у форматі true/false, тобто містить зображення спуфінг-атаку чи не містить, то необхідності у фреймворку з великим набором функціоналу немає. Пропонується використовувати веб-фреймворк Flask

для мови Python. Flask є легким та гнучким у використанні фреймворком, він не покладається на велику кількість розширень, щоб функціонувати [40]. Дизайн Flask дає веб-розробникам повний контроль над архітектурою застосунку. Flask також підтримує модульне програмування, де його функціональні можливості можна розділити на кілька взаємозамінних модулів. Кожен модуль діє як незалежний будівельний блок, який може виконувати одну частину функціональності. Разом це означає, що всі складові частини конструкції є гнучкими, рухливими та тестуються незалежно одна від одної.

Для розробки та тренування ETC класифікатора, що саме і буде визначати чи є зображення спуфінг-атакою чи ні, була обрана бібліотека Scikit-learn. Scikit learn — це бібліотека, яка використовується в машинному навчанні та зосереджена на моделюванні даних. Бібліотека зосереджена на моделюванні, а не на завантаженні та маніпулюванні даними, тому займає не велику кількість пам’яті ЕОМ. Бібліотека машинного навчання scikit-learn у Python містить багато функцій (рис. 3.2) для спрощення машинного навчання, виокремимо деякі з них:

- Алгоритми керованого навчання: будь-який алгоритм машинного навчання, про який ви можливо чули, має дуже високу ймовірність належати до бібліотеки scikit-learn. Набір інструментів scikit-learn має такі алгоритми керованого навчання, який включає – узагальнені лінійні моделі, такі як лінійна регресія, дерева рішень, опорні векторні машини та байєсовські методи;
- Алгоритми неконтрольованого навчання: ця колекція алгоритмів включає факторизацію, кластерний аналіз, аналіз головних компонентів і неконтрольовані нейронні мережі;
- Вилучення ознак: за допомогою scikit-learn ви можете витягувати певні ознаки з тексту та зображень;
- Перехресна перевірка: точність і валідність контрольованих моделей на невидимих даних можна перевірити за допомогою scikit-learn;
- Зменшення розмірності: за допомогою цієї функції можна зменшити кількість атрибутів у даних для подальшої візуалізації, узагальнення та вибору ознак.
- Кластеризація: ця функція дозволяє групувати дані без міток;

— Методи ансамблювання: За допомогою цієї функції можна об’єднати прогнози кількох контрольованих моделей.

Також з метою аналізу даних під час тестів необхідно буде візуалізувати отримані результати і у цьому нам допоможе бібліотека Matplotlib.

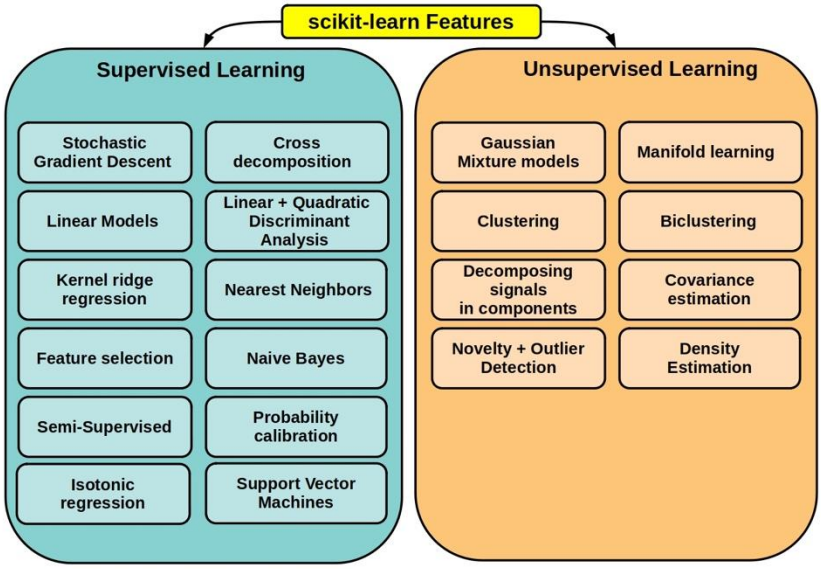


Рисунок 3.2 – Компоненти бібліотеки scikit-learn

Matplotlib — це бібліотека з відкритим кодом, яка допомагає будувати графіки. Бібліотека має кілька переваг, нижче наведено деякі з них:

— Проста у використанні і легка для розуміння початківцям;

— Простіша у використанні для людей, які вже мали досвід роботи з Matlab або іншими інструментами для побудови графіків;

— Бібліотека забезпечує високоякісні зображення та сюжети в різних форматах, таких як png, pdf, pgf тощо;

— Забезпечує керування різними елементами фігури, такими як DPI, колір фігури, розмір фігури.

Отже система буде реалізована за допомогою мови програмування Python. Для детектування зображення обличчя та перетворення кольорового простору зображення, побудови векторів гістограм зображень буде використана бібліотека OpenCV. Для аналізу векторів гістограм зображень буде застосована бібліотека scikit-

learn та Matplotlib. А реалізувати REST API та доступ до бази даних допоможе фреймворк Flask та бібліотека SqlAlchemy.

3.3 Опис архітектури програмної системи

Система має архітектуру з чотирма рівнями PSCD (рис. 3.1), яка притаманна для веб-застосунків, а саме: Presentation Layer, Business Layer, Persistence Layer, Database Layer. Кожен рівень має наступні функціональні вимоги:

— Presentation Layer – презентаційний рівень обробляє HTTP-запити, перетворює параметри JSON в об’єкт, а також автентифікує запит і передає його на бізнес-рівень. Він складається з переглядів, тобто інтерфейсної частини для взаємодії через REST API;

— Business Layer – бізнес-рівень обробляє всю бізнес-логіку. Він складається з класів обслуговування, сервісів та використовує послуги, що надаються рівнями доступу до даних. Він також виконує авторизацію та валідацію даних;

— Persistence Layer – даний рівень відповідальний за взаємодію з базою даних. Містить усю логіку зберігання даних та перекладає дані з бізнес-об’єкти до рядків бази даних;

— Database Layer – на рівні бази даних виконуються операції CRUD (створення, отримання, оновлення, видалення).

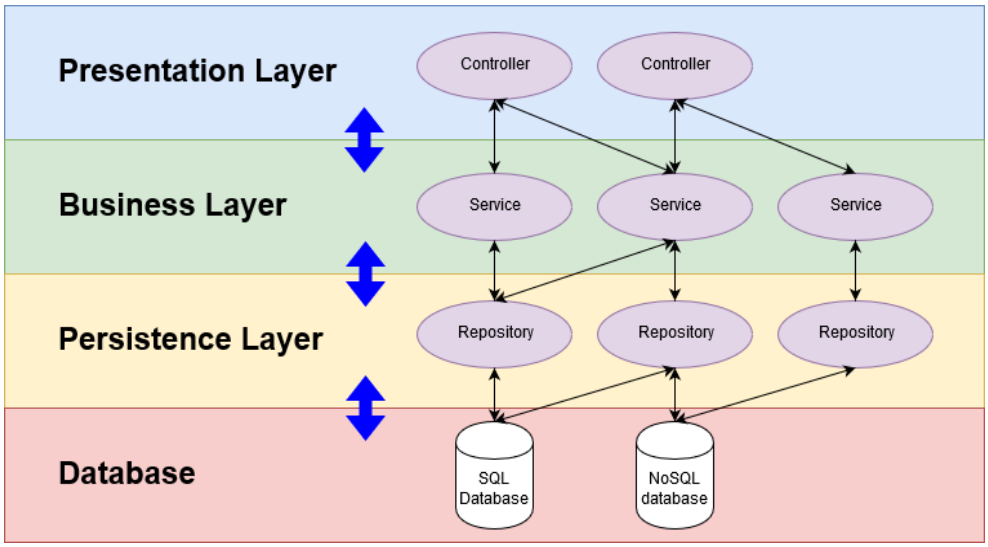


Рисунок – 3.1. Базова архітектура додатку

Треба зазначити, що для реалізації Persistence Layer буде використовуватися бібліотека SqlAlchemy. SQLAlchemy можна використовувати для автоматичного завантаження таблиць із бази даних за допомогою рефлексії без написання SQL скриптів. Рефлексія в SqlAlchemy — це процес читання бази даних і створення об'єктів на основі отриманої інформації. Підхід який пропонує бібліотека SqlAlchemy дозволяє використовувати нам усі наявні переваги об'єктно-орієнтовного програмування та зберігати дані з двовимірних таблиць у об'єктах відповідних класів, що дозволяє більше зосередитися на розробці бізнес-вимог, а не на підтримці технічних рішень системи.

Також до системи входять кеш-даних та база даних. Задача кешу мінімізувати навантаження системи на базу даних та покращити продуктивність системи в цілому, за рахунок того, що під час кожної перевірки зображення на наявність спуфінг-атаки системи не буде одразу вносити інформацію до бази даних, а спочатку буде зберігати її до кешу, і коли в кеші буде зберігатися достатня кількість даних, то вони будуть записані до бази і збереженні.

Архітектура системи передбачає роботу програмного забезпечення як на персональному комп'ютері з наявною веб-камерою, так і на веб-сервері у якості веб-сервісу, коли зображення обличчя передається через REST API, що реалізовано за допомогою фреймворка FLASK.

3.4 Алгоритм роботи програми

Програмне забезпечення має шість послідовних етапів роботи. Далі представлений опис кожного етапу:

1) Перший етап – це отримання зображення обличчя людини з відеопотоку веб-камери або через REST API додатку;
2) Другий етап передбачає нормалізацію зображення за розміром, конвертацію його у чорно-білий кольоровий простір та застосування каскадів Хаара для виявлення обличчя людини;

3) На третьому етапі послідовно виконується конвертація зображення з кольорового простору RGB до кольорових просторів YC_rC_b і CIE L*u*v*, як це зазначено у формулах 2.1 – 2.9. Відповідно ми маємо два зображення, одне у просторі YC_rC_b та інше у просторі CIE L*u*v*. Далі для кожного зображення ми отримаємо три вектори, кожен вектор відповідає одному каналу кольорового простору (YC_rC_b чи L*u*v*) і кожен елемент вектора відповідає пікселю зображення відповідно. Кінцевою фазою даного етапу є об'єднання усіх шести векторів в один вектор (вектор ознак).
4) Четвертий етап передбачає застосування ETC класифікатори на вхід якого подається об'єднаний вектор отриманий під час третього етапу. Результатом роботи класифікатора є числове значення ймовірності того, що зображення містить спуфінг-атаку від 0 до 1.
5) На п'ятому етапі інформація про виявлення спуфінг-атаки виводиться на екран користувачу, або відправляється REST response у форматі true/false, якщо був використаний REST API.
6) Шостий етап – це збереження даних про спробу ідентифікації до кешу застосунку або до бази даних у випадку, коли розмір кешу досягає 100 записів або останній запис був внесений більше шести хвилин тому.

Для кращого розуміння роботи системи була розроблена блок-схема роботи застосунку, що представлена у Додатку В.

3.5 Реалізація сервісів системи та діаграми класів

У даному підрозділі послідовно розглянемо реалізацію кожного сервісу, що реалізують функціонал блоків із структурної схеми, яка була описана у підрозділі 2.4. У підрозділі визначена структура класів та технічні рішення, що були використанні для реалізації того чи іншого сервісу.

Сервіс отримання відеозображення реалізований у вигляді класу VideoCaptureService (рис 3.2). Клас інкапсулює у собі інформацію про максимальні розмір вікна відеозображенням та номер/назву порту до якого під'єднана веб-камера

комп'ютера. Метод класу captureVideo перевіряє доступність веб-камери комп'ютера та активує функцію запису відео з веб-камери. Метод класу getOneFrame бере один поточний кадр з відеопотоку та надає його у вигляді масиву байтів. Доступ до веб-камери здійснюється за допомогою бібліотеки OpenCV, клас VideoCapture.

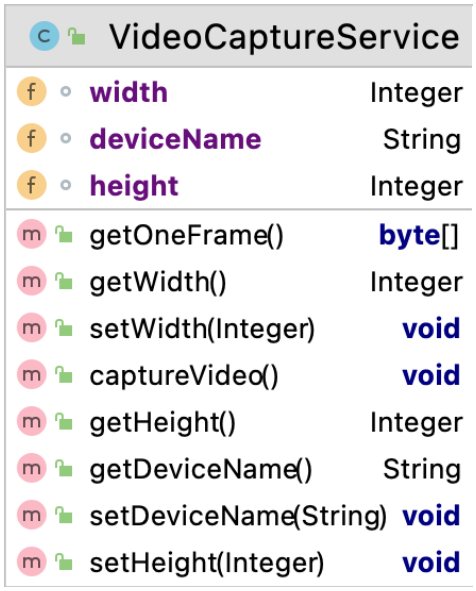


Рисунок 3.2 – UML діаграма класу сервісу отримання відеозображень

Клас FaceDetectorService відповідальний за реалізацію сервісу детекції обличчя людини з каскадами Хаара (рис 3.3). До класу інкапсулювані два зміні класів OpenCV: faceCascade класу Cascade та converter класу CvtColorConverter. Зміна converter дозволяє виконувати перетворення вхідного зображення у чорно-білий кольоровий простір, а зміна faceCascade зберігає самі каскади Хаара. Зміна img зберігає вхідне зображення на якому буде визначатися обличчя людини. Методи detectFaces та detectFirstFace виконують перетворення вхідного зображення у чорно-білий кольоровий простір та шукають частину зображення з обличчям людинни, використовуючі класс CascadeClassifier та метод detectMultiScale бібліотеки OpenCV. Класс може визначати усі зображень обличь людей на одному кадрі.

Для реалізації сервісу утворення гістограм зображень для двох кольорових просторів розроблений клас HistogramBuilderService (Рисунок 3.4). Клас інкапсулює у собі зміну img, яка зберігає зображення, для якого будуть утворюватися три

вектори, де кожен елемент вектору відповідає значенню яскравості одного з трьох каналів YC_rC_b чи CIE L*u*v* простору, для конкретного пікселя зображення.

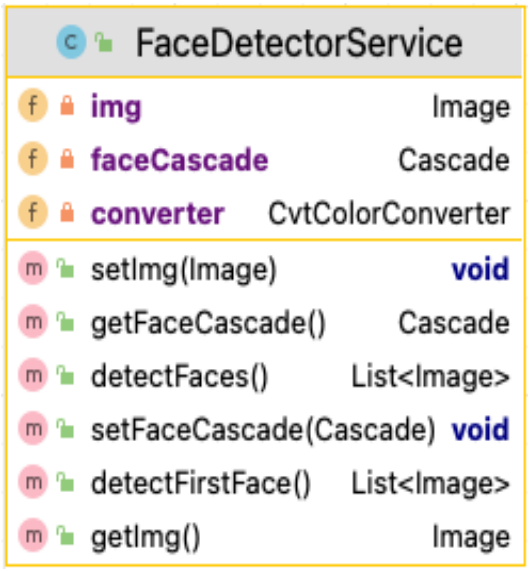


Рисунок 3.3 – UML діаграма класу сервісу отримання відеозображень

Для того щоб отримати три вектори для зображення розроблений метод calculateHist. Перед тим як застосувати цей метод зображення конвертується з RGB простору в YC_rC_b чи CIE L*u*v* за допомогою класу OpenCV CvtColorConverter. З метою тестування системи у клас HistogramBuilderService також був доданий метод plotHist, який дозволяє побудувати гістограми для заданого зображення, щоб можна було оцінити певні ознаки зображень зі спуфінг-атаками та без них. На гістограмі можна побачити значення для усіх трьох каналів відповідного кольрового простору. Метод plotHist реалізований з використанням бібліотеки Matplotlib та її класу Pyplot, що дозволяє не тільки будувати гістограми, а й ще масштабувати їх та зберігати зображення у окремий файл гістограм для кожного каналу кольорових просторів.

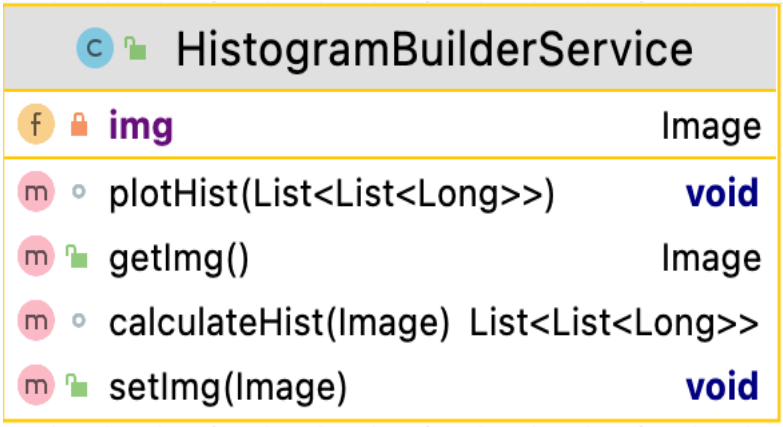


Рисунок 3.4 – UML діаграма класу сервісу утворення гістограм зображень

Сервіс визначення спуфінг-атаки з ETC реалізований у вигляді двох класів: Classifier та дочірнього класу EtcClassifier (рис. 3.5). Два класи створені для зручного розширення функціоналу у майбутньому. Так клас Classifier містить базовий функціонал для роботи з класифікаторами бібліотеки sklearn та інкапсулює у собі зміну pathToFile, яка зберігає шлях до файлу формату «.pkl», який в свою чергу зберігає у собі значення коефіцієнтів натренованої моделі класифікатора, також клас Classifier містить метод loadModel, який завантажує модель класифікатора з файлу формата «.pkl» за допомгою класу Joblib бібліотеки sklearn, метод load. Клас EtcClassifier вже реалізує метод predict. Метод predict у якості вхідного параметру приймає об'єднаний вектор, який складається з шести векторів (по три вектори на кожен простір YC_rC_b і CIE L*u*v*), та за допомогою завантаженої моделі класифікатора і класу Classifier та методу predict_proba цього класу, який надає бібліотека sklearn, на виході видає числове значення в діапазоні від 0 до 1, яке відповідає значенню ймовірності спуфінг-атаки для вхідного зображення. Окремо треба зазначити, що також були реалізовані класи для python скрипта, який виконує тренування ExtraTreesClassifier класифікатора на завчасно підготовлених даних, з заданими вхідними параметрами, а також можливістю дізнатися похибку та час навчання класифікатора. Інформація про усі класи додатку системи представлені у UML діаграмі класів у Додатку Г.

Наступний сервіс – це сервіс збереження даних про спроби виявлення спуфінг-атак, який реалізований за допомогою бібліотеки SqlAlchemy за принципом CRUD

репозиторія. Сервіс розроблений у вигляді одного класа AuthenticationAttemptRepository (рис. 3.6) який реалізує усі чотири операції CRUD: створення запису, оновлення запису, видалення та отримання інформації про запис у БД. Засоби бібліотеки SqlAlchemy надають нам можливість виконувати усі операцій по взаємодії з базою даних без написання SQL скриптів, що значно пришвидшує розробку фунціоналу програми і дає можливість більше сконцентруватися на розробці бізнес-логіки.

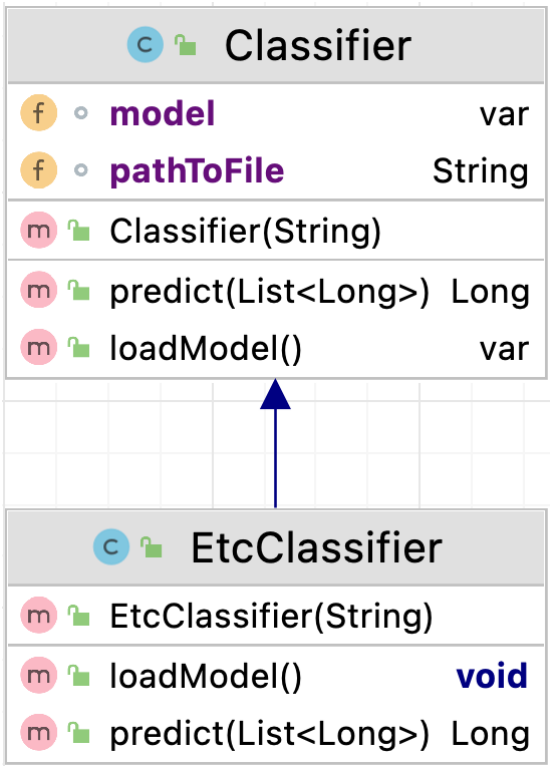


Рисунок 3.5 – UML діаграма класу сервісу визначення спуфінг-атаки з ETC

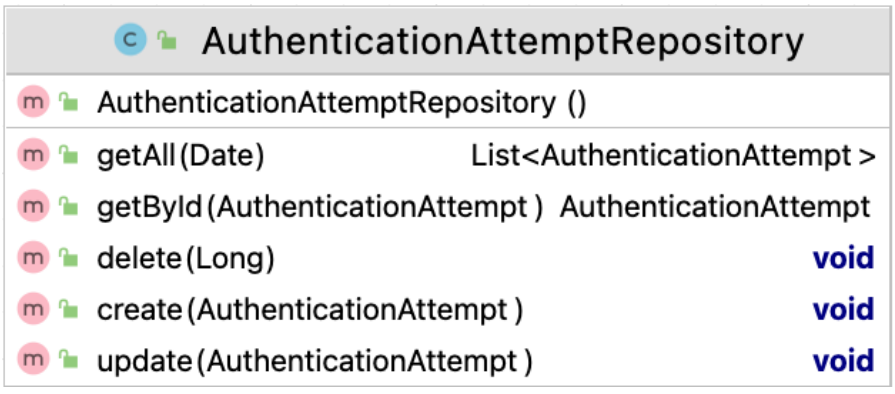


Рисунок 3.6 – UML діаграма класу сервісу збереження даних до БД

REST сервіс реалізований за допомогою класу AuthenticationAttemptController (рис. 3.7). Основний функціонал класу реалізований за допомогою веб-фреймворку Flask. Клас має головний метод, який забезпечують доступ до бізнес-логіки через REST API: checkForSpoofing. За допомогою спеціального декоратора Flask: “@app.route('/checkForSpoof/', methods=['POST'])”, задається URL для доступу до веб-ресурсу, а також HTTP метод POST. Структура даних, які подаються на вхід REST сервіса, задана за допомогою класу AuthenticationDto (дивись Додаток Г), поля якого автоматично заповнюються Flask під час десеріалізації вхідного JSON, де JSON розшифровується як JavaScript Object Notation. JSON — це текстовий формат для зберігання та транспортування даних, в тому числі і через HTTP протокол. Клас AuthenticationDto містить у собі об’єкт типу MultipartFile бібліотеки Flask, що дозволяє передавати файл зображення, яке буде перевірятися на наявність спуфінг-атаки. Перед початком обробки REST запиту клас AuthenticationAttemptController виконує його валідацію за допомогою метода validateRequest, який перевіряє щоб усі поля класу AuthenticationDto були заповнені даними відповідного типу.

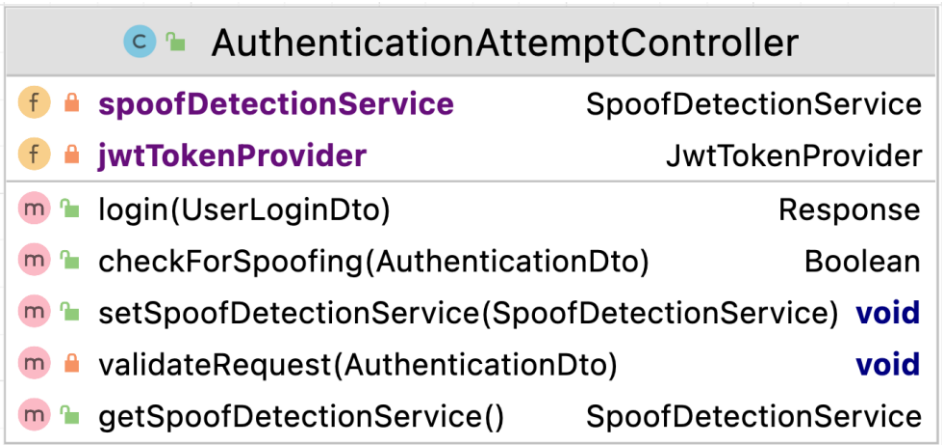


Рисунок 3.7 – UML діаграма класу REST сервісу

Також для реалізації функціоналу REST сервіса був створений клас SpoofDetectionService (рис. 3.8), що об’єднує у собі усю бізнес логіку класів FaceDetectorService, HistogramBuilderService та EtcClassifier. Таке розбиття на класи дозволяє притримуватися принципів розробки ПО SOLID. Так метод hasImageSpoofing класу SpoofDetectionService має у якості вхідного параметру

зображення, а на виході повертає булеве значення (формату true/flase), що сповіщає систему про наявність спуфінг-атаки.

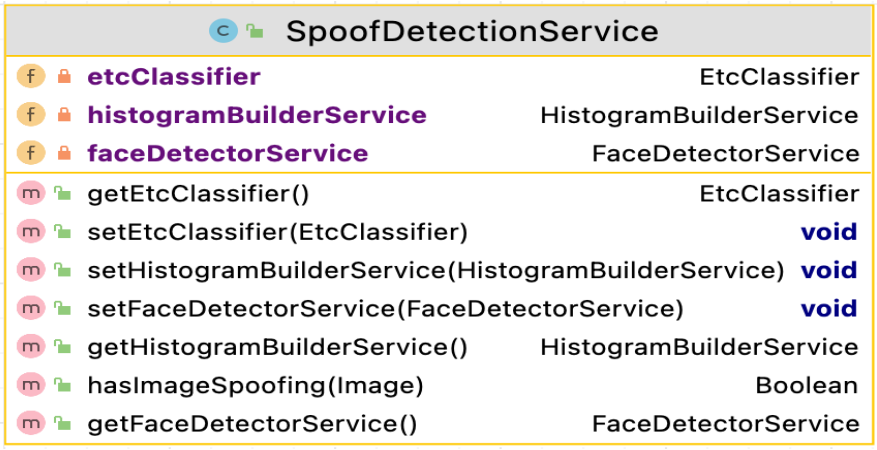


Рисунок 3.8 – UML діаграма класу SpoofDetectionService REST сервісу

Окремо треба зупинитися на заходах безпеки при реалізації REST API. В архітектурі REST наскрізна обробка даних передбачає послідовність потенційно вразливих операцій:

- Під час мапінгу з/на HTTP-повідомлення та URL-адресу ресурсу (відображення контролера);
- Коли створюється екземпляр об’єкта, що представляє цільовий ресурс, і виконується запитана операція (виклик сервісів із контролера);
- Під час створення представлення стану для цільових ресурсів (специфічної функції, сервісу);
- Під час доступу/зміненні даних у серверних системах, які містять стан ресурсу (збереження в БД або у кеш).

Багаторівнева послідовність перетворень у рамках REST означає, що одна слабка ланка в ланцюжку може зробити вашу програму вразливою. Так найрозповсюдженішими атаками є: Injection Attacks, DoS Attacks, Broken Access Control, Parameter Tampering та MITM attack.

Враховуючі усі фактори небезпеки для REST API частини системи, для її захисту був реалізований функціонал роботи з JWT токеном. JWT – це JSON веб-токен, є відкритим стандартом (RFC 7519) [41], який визначає компактний і

самодостатній спосіб безпечної передачі інформації між сторонами інтернет мережі як об’єкт JSON. Через відносно невеликий розмір JWT можна надіслати через URL-адресу, через параметр POST або всередині заголовка HTTP, і він швидко передається. JWT містить всю необхідну інформацію про сутність відправника REST запиту, щоб уникнути повторного звернення до бази даних. Одержувачу JWT також не потрібно викликати сервер для перевірки токена. Використання JWT має переваги порівняно з простими веб-токенами (SWT) і токенами мови розмітки безпеки (SAML):

- Більш компактний: JSON є менш детальним, ніж XML, тому, коли він закодований, JWT менший, ніж токен SAML. Це робить JWT хорошим вибором для передачі в середовищах HTML і HTTP;
- Більш безпечний: JWT можуть використовувати для підпису пару відкритих/приватних ключів у формі сертифіката X.509. JWT також може бути симетрично підписаний спільним секретним ключем за допомогою алгоритму HMAC. І хоча токени SAML можуть використовувати пари відкритих/приватних ключів, як-от JWT, підписати XML за допомогою цифрового підпису без введення незрозумілих прогалин у безпеці дуже складно порівняно з простотою підпису JSON;
- Більш поширений: аналізатори JSON поширені в більшості мов програмування, в тому числі і Python, оскільки вони відображаються безпосередньо на об’єкти. І навпаки, XML не має природного відображення документа в об’єкт. Це полегшує роботу з JWT;
- Легший в обробці: JWT використовується в масштабі Інтернету. Це означає, що його легше обробляти на пристроях користувача, особливо мобільних.

Так у системі для роботи із JWT токенами для автентифікації користувача були розроблені класи: JwtTokenFilter та JwtTokenProvider (дивись додаток Г). Клас rest-контролера AuthenticationAttemptController інкапсулює у собі об’єкт класу JwtTokenProvider (рис. 3.7), що дозволяє створювати (у методі login) та валідувати токен клієнта, що звертається до системи через REST API. Клас JwtTokenFilter перехоплює кожен REST запрос до системи, та перевіряє наявність токена і його

валідність для кожного окремого клієнта і при успішній валідації перенаправляє запрос на клас rest-контролера AuthenticationAttemptController.

Також програмна реалізація включає низку класів:

— Для роботи з обліковими записами користувачів;

— Класи для відображення та створення нотифікацій для користувача;

— Класи для роботи з налаштуваннями додатку.

Усі перелічені класи програмного продукту представлені в додатку Г.

3.6 Проектування структури бази даних для застосунку

В якості основної СУБД для роботи системи було обрано PostgreSQL. Рішення обумовлено тим, що PostgreSQL є безкоштовним рішенням та не передбачає отримання ліцензії для комерційної розробки, також СУБД працює на всіх основних операційних системах, включаючи Linux, Mac OS X, UNIX (Gentoo, Debian, Mint, Zorin, CutefishOS) і Windows [42]. СУБД підтримує зберігання тексту, зображення, звуків та відео, а також включає інтерфейси програмування для C/C++, Java, Perl, Python, Ruby, Tcl і Open Database Connectivity (ODBC). PostgreSQL підтримує значну частину стандарту SQL і пропонує багато сучасних функцій, включаючи такі:

— Керування багатоверсійним паралелізмом (MVCC);

— Потокова реплікація (станом на версії 9.0);

— Hot Standby (станом на версії 9.0).

Також авторами PostgreSQL розроблений зручний та безкоштовний клієнт для роботи з СУБД – PgAdmin, який має простий у використанні інтерфейс, що дозволяє створювати таблиці та інші об’єкти БД без написання SQL скриптів (рис. 3.9). Ще однією перевагою PostgreSQL є те, що PostgreSQL підтримує чотири стандартні процедурні мови, що дозволяє користувачам писати власний код будь-якою з мов, і його можна виконувати на сервері бази даних PostgreSQL. Ці процедурні мови: PL/pgSQL, PL/Tcl, PL/Perl і особливо важлива для нашої системи **PL/Python** . Крім

того, підтримуються інші нестандартні процедурні мови, такі як PL/PHP, PL/V8, PL/Ruby, PL/Java тощо.

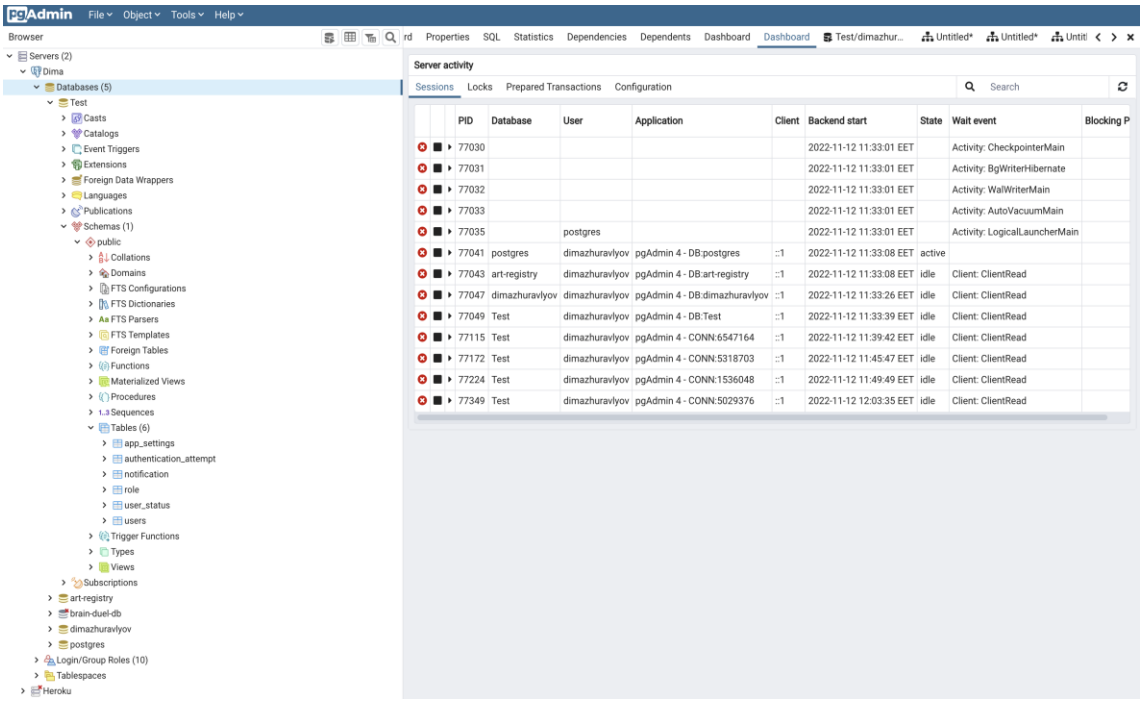


Рисунок 3.9 – Інтерфейс клієнту PgAdmin для роботи з СУБД

Також система передбачає наявність кешу, що було описано у підрозділі 3.3, для підвищення швидкості роботи системи. Для реалізації кешу була застосована система Redis — розподілене сховище даних, яке зберігається в оперативній пам'яті. Redis — це розширене сховище типу словника, тобто зберігаються ключі і значень за цими ключами, із відкритим вихідним кодом і це ідеальне рішення для створення високопродуктивних масштабованих веб-додатків [43]. Redis має три основні особливості, які відрізняють його від інших сховищ:

— Redis повністю зберігає свою базу даних у оперативній пам’яті, використовуючи диск лише для примусового збереження;
— Redis має відносно багатий набір типів даних у порівнянні з багатьма сховищами даних “ключ-значення” [43];
— Redis може копіювати дані на будь-яку кількість підлеглих пристроїв.

Нижче наведено певні переваги Redis для застосунків:

— Надзвичайно швидкий – Redis дуже швидкий і може виконувати близько 110 000 операцій запису за секунду і приблизно 81 000 операцій читання даних за секунду;
— Підтримує розширені типи даних − Redis нативно підтримує більшість типів даних, які вже відомі розробникам, наприклад список, set, відсортований set і хеш таблиці. Це полегшує вирішення різноманітних проблем, оскільки ми знаємо, яку проблему можна краще вирішити за допомогою якого типу даних;
— Операції є атомарними – усі операції Redis є атомарними, що гарантує, що якщо два клієнти одночасно отримають доступ, сервер Redis отримає останнє оновлене значення;
— Багатофункціональний інструмент – Redis є багатофункціональним інструментом, який можна використовувати в кількох випадках, наприклад кешування, черги обміну повідомленнями (Redis вбудовано підтримує публікацію/підписку), будь-які короткочасні дані у вашій програмі, як-от веб-сесії програми, підрахунок звернень до веб-сторінки тощо.

Після того, як ми визначилися з технічними аспектами створення та керування БД можемо перейти до опису структури бази. Для того щоб описати структуру бази даних нам необхідно визначити, яку інформацію ми збираємося зберігати. Запропонована структура бази даних повинна вирішувати питання зберігання:

— Даних користувачів системи та її адміністраторів;
— Вдалих та невдалих спроб перевірки зображення на спуфінг-атаку;
— Налаштувань системи;
— Системних нотифікацій системи.

Враховуючі інформацію, що була викладена вище, база даних PostgreSQL буде складатися з шести таблиць (дивись Додаток Ґ). Далі розглянемо основні таблиці додатку.

USERS – таблиця зберігає інформацію про користувачів системи. В таблиці зберігаються в тому числі дані адміністраторів системи та дані користувачів REST API (рис. 3.10). Таблиця має наступні колонки:

- Id – унікальний ідентифікатор;
- Firstname – ім’я користувача;
- Lastname – прізвище користувач;
- Username – ім’я та прізвище користувача або коретке ім’я користувача;
- Password – пароль до системи користувача;
- Email – електрона поштова адреса користувача;
- registr_date – дата реєстрації користувача;
- status_id – статус користувача в системі;
- role_id – роль користувача в системі (звичайний користувач, адміністратор, користувач REST API);
- image – зображення обличчя користувача, представлене як масив байтів.

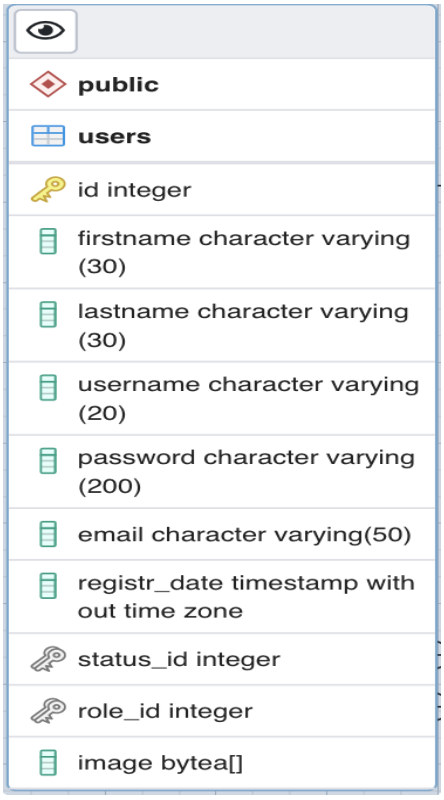


Рисунок 3.10 – Структура таблиці USERS

APP_SETTINGS – таблиця зберігає інформацію про налаштування системи, які можуть змінюватися під час роботи додатку (рис. 3.11). Наприклад, це шлях до файлу з моделлю ETC класифікатора, дані про характеристики серверу, порогове значення ймовірності виявлення спуфінг-атаки. Таблиця має наступні колонки:

- id – унікальний ідентифікатор;
- name – назва налаштування додатку;
- value – значення налаштування для системи;
- user_id – зовнішній ключ, що посилається на запис про користувача;
- description – опис налаштування.

AUTHENTICATION_ATTEMPT – таблиця містить дані про кожну спробу виявлення спуфінг-атаки для конкретного користувача системи (рис. 3.12). Таблиця має наступні колонки:

- id – унікальний ідентифікатор запису;
- user_id – зовнішній ключ, що посилається на запис про користувача;
- date – дата проведення перевірки;
- success – результат перевірки формату true/false. Значення True означає, що спуфінг-атаку виявлено;
- comment – коментар до запису;
- photo – фото обличчя людини, представлене як масив байтів.

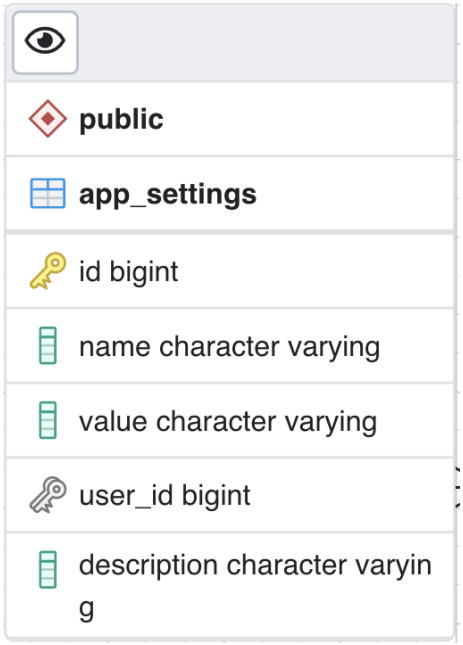


Рисунок 3.11 – Структура таблиці APP_SETTINGS

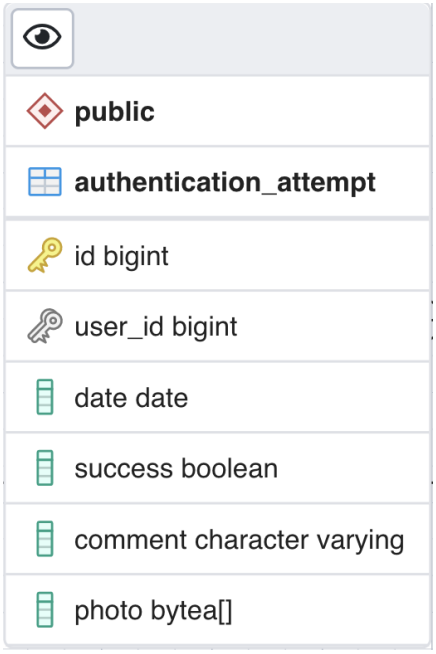


Рисунок 3.12 – Структура таблиці AUTHENTICATION_ATTEMPT

NOTIFICATION – таблиця містить дані про нотифікації для користувачів, які можуть демонструватися користувачам у випадках планових робіт по модифікації системи або у аварійних випадках. Таблиця має наступні колонки:

– id– унікальний ідентифікатор запису;
– text – текстове повідомлення для користувача;

- seen – відмітка про те, чи була показана нотифікація користувачу;
- date – дата створення нотифікації;
- user_id – зовнішній ключ, що посилається на запис про користувача;

Також структура бази даних включає додаткові таблиці для зберігання інформації про статус та роль користувача системи.

3.7 Тренування ETC класифікатору

Для тренування ETC класифікатору, що саме і визначає спуфінг-атаки, треба визначитися з доступними датасетами зображень, що містять спуфінг-атаки. Щоб оцінити ефективність алгоритмів виявлення спуфінг-атак, багато опублікованих робіт розробили та протестували свої алгоритми на власних базах даних підробок, однак лише кілька авторів зробили загальнодоступними свої бази даних для виявлення спуфінг-атак [44-46]. У цьому підрозділі надано короткий опис двох загальнодоступних баз даних підробки обличь, що застосовані для навчання ETC класифікатора у даній роботі: база даних Idiap REPLAY-ATTACK і база даних CASIA Face Anti- Spoofing. Існує кілька інших загальнодоступних баз даних для виявлення підробок обличчя, наприклад, база даних VidTIMIT Audio-Video і база даних DaFEx (8 суб'єктів) також використовувалися для виявлення підробки обличчя, але їх обмежений розмір і різноманітність підробок робить їх менш привабливими. для використання в експериментальних дослідженнях.

База даних Idiap REPLAY-ATTACK [47], випущена в 2012 році, складається з 1300 відеозаписів спроб реального доступу та атак 50 різних суб'єктів. У тих самих умовах отримання (контрольоване та несприятливе освітлення) атаки підробки обличчя було згенеровано шляхом підробки спроб перевірки в реальному часі тих самих суб'єктів за допомогою друкованих фотографій, відображених фотографій/відео на екрані мобільного телефону та відображених фотографій/відео на HD екрані.

База даних CASIA Face Anti-Spoofing Database (FASD) [46], випущена в 2012 році, складається з 600 відеозаписів справжніх обличь людей і спроб атак 50 різних осіб. Хоча розмір бази даних CASIA дещо менший, ніж база даних Idiap, вона містить більш різноманітні зразки з точки зору пристроїв збору (камера Sony NEX-5 з високою роздільною здатністю та низькоякісна веб-камера), варіації обличчя (варіації пози та виразу) , а також спроби атаки (деформація фото, вирізане фото та HD-відео).

У таблиці 3.1 наведена зведена інформація двох вищезгаданих баз даних з точки зору розміру вибірки, пристрою збору даних, типу атаки, а також розподілу суб'єктів за віком, статтю та расою.

Таблиця 3.1 – Характеристики датасетів CASIA та Idiap

Датасет	Рік випуску	Кількість різних обличь людей	Кількість відеозаписів	Пристрій збору даних	Типи атак	Раса людини	Гендер людей	Вік людей
Idiap REPLAY-ATTACK	2012	50	200 з реальними обличчями; 1000 з атаками	MacBook 13__camera (320 × 240)	Printed attack; Replay attack	Європейська 76%; Азійська 22%; Афро-американська 2%;	Чоловіки % 86; Жінки %14	Від 20 до 40 років
CASIA FASD	2012	50	150 з реальними обличчями; 450 з атаками	Камера низької якості зйомки (640 × 480) Камера середньої якості	Printed attack; Replay attack	Азійська 100%	Чоловіки % 86; Жінки %14	Від 20 до 35 років

				зйомки (480 × 640) Sony NEX-5 camera (1280 × 720)				

Для тренування ETC класифікатору та створення робочої моделі застосовується бібліотека sklearn та її клас ExtraTreesClassifier. Були визначенні наступні параметри для ETC класифікатору під час тренування:

- n_estimators – Кількість дерев у лісі. У нашому випадку значення дорівнює 20;
- criterion – Функція вимірювання якості розбиття. У нашому випадку Gini impurity;
- max_depth – Максимальна глибина дерева. Якщо не встановлена, то вузли розгортаються до тих пір, поки усі листки чисті або поки всі листки не містять менше ніж min_samples_split вибірки;
- min_samples_split – Мінімальна кількість зразків, необхідних для розбиття внутрішнього вузла. У нашому випадку значення дорівнює 2;
- min_samples_leaf – Мінімальна кількість зразків, необхідна для листкового вузла. Точка розколу на будь-якій глибині розглядатиметься лише в тому випадку, якщо вона хоча б виходить за min_samples_leaf у кожній з лівої та правої гілок. Це може мати ефект згладжування моделі, особливо в регресії. У нашому випадку значення дорівнює 10;
- n_jobs – Кількість завдань, які потрібно виконати паралельно. У нашому випадку значення дорівнює 8;
- min_impurity_split – Поріг ранньої зупинки росту дерева. Вузол розділиться якщо його домішка перевищує поріг, інакше це лист.

Навчання класифікатору зайняло 20 хвилин. У результаті маємо дві натреновані моделі для двох різних датасетів CASIA FASD та Idiap REPLAY-ATTACK.

3.8 Висновки до розділу

В даному розділі були визначені основні програмні зособи реалізації системи, а саме: мова програмування, бібліотеки, фреймворки. Так обраною мовою програмування є Python, фреймворком для розробки REST API є Flask, допоміжні бібліотека для реалізації: openCV, sklearn, numPy, matplotlib.

Була визначена архітектура системи відповідно до якої бувли розробленні необхідні класи сервісів, репозиторіїв та контролерів. Розроблена та описана система захисту REST API інтерфейсу на основні JWT токену. Був описаний функціонал кожного класу програми, а також була створена UML діаграма класів додатку.

Визначена СУБД додатку, а саме PostgreSQL та ORM бібліотека для взаємодії з БД sqlAlchemy. Розроблена структура бази даних, яка представлена у вигляді ER діаграми. Наданий опис кожної таблиці бази даних та її структура.

Проведений опис навчання ETC класифікатору. Визначені наявні у відкритому доступі датасети зображень зі спуфінг-атаками, а саме CASIA та Idiap для навчання класифікатору. Визначені основні параметри налаштувань ETC класифікатору та описані результати навчання моделі.

4 РОЗРОБКА ІНТЕРФЕЙСУ СИСТЕМИ

4.1 Опис графічного інтерфейсу користувача

Інтерфейс система маж дві складові: графічний інтерфейс, що надається користувача під час застосування веб-камери комп’ютера та REST API для відправки зображення до системи на перевірку через мережу інтернет.

Графічний інтерфейс реалізований засобами бібліотеки OpenCV, що дозволяють створювати вікна настільних комп’ютерних додатків. Як було зазначено у підрозділі 4.1, функціонал захвату зображення з веб-камери та його виведення на дисплей реалізує клас VideoCaptureDervice, що використовує для цього клас VideoCapture бібліотеки OpenCV. Графічний інтерфейс представляє собою вікно розміром 320x240 пікселів на яке виводиться зображення з веб-камери. Як тільки програма виявляє обличчя людини, навколо обличчя утворюється синій прямокутник. Якщо перед веб-камерою знаходиться справжня людина, то поряд з зображенням обличчя виводиться напис «True» зеленого кольору (рис. 4.1). Якщо до веб-камери представлено фото чи дисплей з відеозаписом, то колір рамки змінюється на червоний та поряд з рамкою виводиться напис «False» (рис. 4.2). Розміри вікна можуть змінюватися рухами миші комп’ютера.

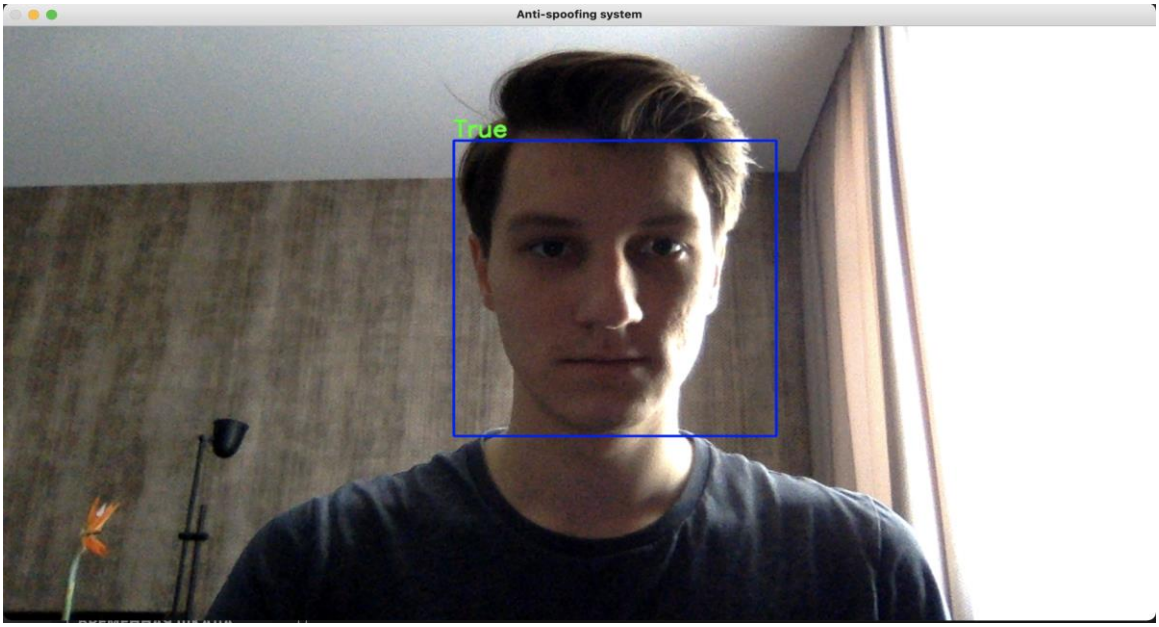


Рисунок 4.1– Інтерфейс додатку під час виявлення справжнього обличчя людини

Також додаток включає можливість відображення нотифікацій для користувачів системи у певних випадках (рис. 4.3). Так система може попереджувати про: відсутність з’єднання з базою даних; наявність певних помилок під час своєї роботи; планові технічні роботи та їх час проведення.

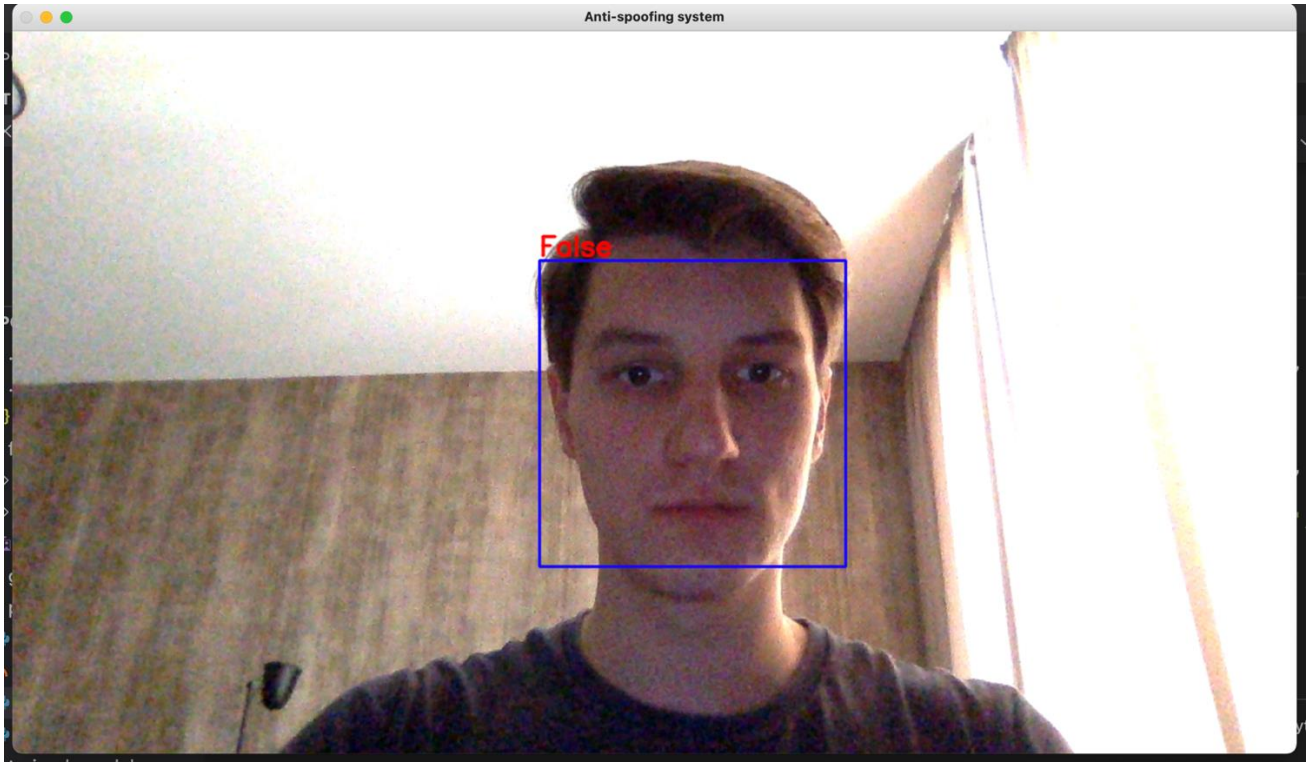


Рисунок 4.2 – Інтерфейс додатку під час виявлення спуфінг-атаки

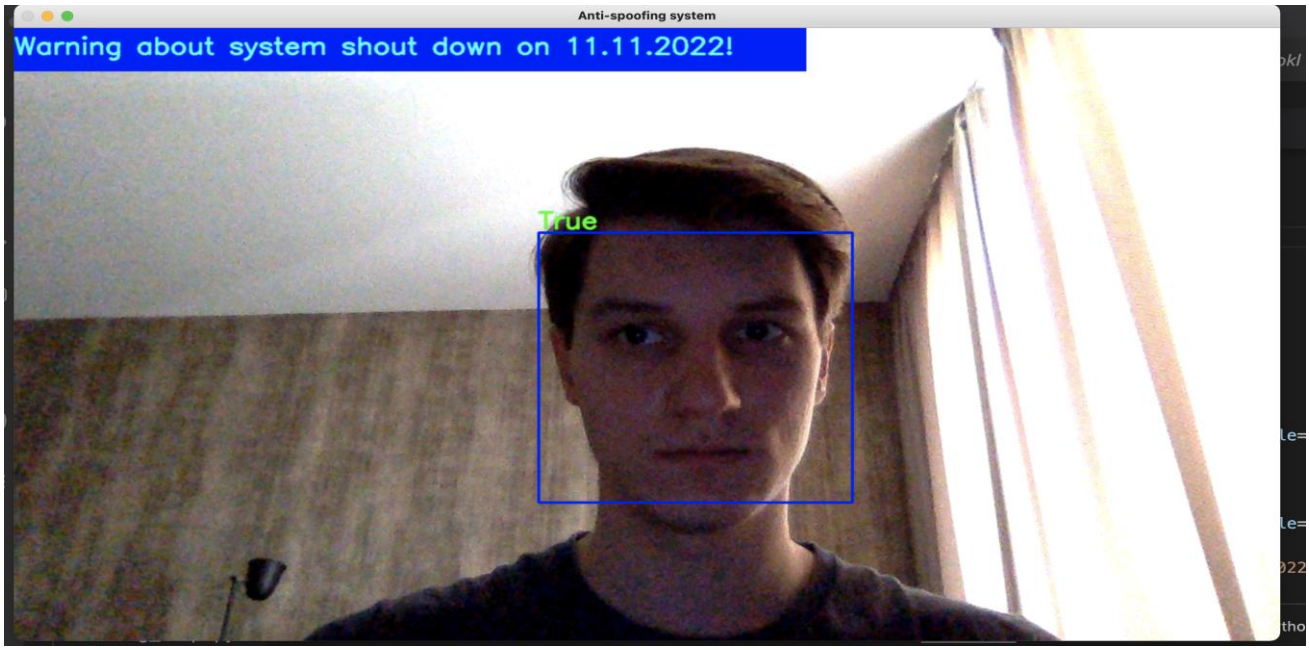


Рисунок 4.3 – Інтерфейс додатку під демонстрації нотифікації

4.2 Опис REST інтерфейсу

Інша частин інтерфейсу користувача – це REST API. REST інтерфейс складається з кількох ендпоїнтів перелік яких представлений у таблиці 4.1.

Таблиця 4.1 – Опис ендпойнтів REST API

URL адреса	HTTP метод	Вхідні параметри	Відповідь REST	Опис

{server_url}/api/spoofApp/checkForSpoofing	POST	AuthenticationDto {
		MultipartFile photoToCheck;
		String userName;
		Long userId;
		}
			True/false	Перевірка зображення на наявність спуфінг-атаки. Якщо відповідь True, то спуфінг-атаки немає.

{server_url}/api/spoofApp/login	POST	UserLoginDto {
		String username;
		String password;
		}
			Response {
			id;
			firstName;
			lastName;
			email;
			status;
			role;
			username;
			token;
			}
				API для отримання jwt токену.

{server_url}/api/spoofApp/user/createUser	POST	UserCreateDto { String firstName; String lastName; String username; String email; String password; String image; Role role; }	Response { id; firstName; lastName; email; status; role; username; }	API для створення облікового запису користувача.
{server_url}/api/spoofApp/user/update	POST	UserUpdateDto { Long id; String firstName; String lastName; String username; String email; String password; String image; Role role; }	Response { id; firstName; lastName; email; status; role; username; }	API для оновлення даних облікового запису користувача.

{server_url}/api/spoofApp/ user/delete/{id}	DELETE	id – унікальний ідентифікатор користувача	HTTP code 200	API для видалення даних облікового запису користувача.
{server_url}/api/spoofApp/ user/search/{name}/{first}/{last}	GET	name – унікальне ім’я користувача (нікнейм); first – ім’я користувача; last – прізвище користувача.	Response { id; firstName; lastName; email; status; role; username; }	API для пошуку користувача за нікнеймом або ім’ям або прізвищем.
{server_url}/api/spoofApp/ user user/image/{id}	GET	id – унікальний ідентифікатор користувача	Multipart File photo	API для отримання фото обличчя зареєстрованого користувача

4.3 Висновки до розділу

Уданому розділі був представлений графічний інтерфейс користувача для роботи додатку з веб-камерою комп’ютера (під час роботи у якості настільного додатку), а також описаний інтерфейс для взаємодії з додатком через REST API. Для REST API були описані усі наявні ендпоїнти, їх функціонал, URL адреси, HTTP методи, формат вхідного запиту (формат JSON), а також формат відповіді до клієнта.

5 ТЕСТУВАННЯ СИСТЕМИ

5.1 Тестування на датасетах CASIA FASD та Idiap REPLAY-ATTACK

Для тестування системи і визначення її показника ефективності HTER використаємо датасети, що були описані у підпункті 4.4. Усього буде використано 1800 зразків для тестування. Результати роботи системи наведені у таблиці 5.1.

Таблиця 5.1 – результати тестування системи

Метод	Датасет	EER(%)	HTER(%)
Запропоноване рішення: YC_rC_b+Luv+ETC	CASIA FASD	0.074	0.7
	Idiap REPLAY-ATTACK	0.074	0.58

З даних у таблиці 5.1 можна зробити висновок, що розроблена система виконує поставлену до неї вимогу у підрозділі 1.6 і показник HTER < 1%.

Для того щоб можна було більш змістовно оцінити якість роботи системи була складена порівняльна таблиця 5.2 показника HTER з іншими підходами. Треба зазначити, що результати HTER у таблиці 5.2 стосуються лише статичних методів (тобто методів, які використовують лише одне зображення для виявлення спуфінг-атаки). У порівнянні з методами, заснованими на виявленні живості та/або використанні часових проміжків (кліпання очима, наприклад), є деякі методи, які дають кращі результати, ніж запропонований підхід, а деякі роботи досягають гірших результатів HTER, наприклад у [48-49].

Таблиця 5.2 – Порівняння роботи алгоритмів на датасеті Replay-Attack

Метод	EER(%)	HTER(%)
Radiometric Transforms [52]	-	0.8

DEND-CLUSTERING-Ensemble [50]	-	5.0
MAXDIST-Ensemble [50]	-	5.0
CTMF [51]	-	4.4
Unicamp [53]	9.83	15.62
ATVS [53]	0.83	12.00
MUVIS [53]	0.00	1.25
PRA Lab [53]	0.00	1.25
Client Specific MsLBP [54]	-	1.45
Client Specific HOG [54]	-	3.58
Client Specific LBP-TOP+SVM [55]	-	3.95
LBP+SVM [56]	-	13.87
HSV-Y C_bC_r+C-SURF+PCA [15]	0.1	2.2
CNN [57]	-	10
Kernel Fusion (MBSIF-TOP+MLPQ-TOP) [58]	1.67	1.00
LBP+DoG+HOG+IQA [59]	1.6	1.0
Multiscale (HSV+Y C_bC_r)+SVM [15]	-	3.1
Color texture CNN + SVM [60]	0.1	0.9
CTMF [61]	4.0	4.4

FASNet [62]	-	1.2
ResNet-50[63]	1.16	1.28
GIF + IQA [64]	1.02	1.31
Запропоноване рішення: YC_rC_b+Luv+ETC	0.074	0.58

Що стосується бази даних Replay-Attack, як ми бачимо у таблиці 5.2 то результати демонструють, що комбінація кольорових просторів YCrCb і CIE L*u*v* здатна протистояти сучасним методам спеціально розробленим для виявлення підробки обличчя за якістю роботи (показником HTER). Треба зазначит, що для підвищення продуктивності після перетворення простору кольорів можна використовувати дескриптор текстури. Деякі автори виявляють переваги використання дескриптора текстури, як у [65, 15]. Тобто є сенс розглянути поєднання підходів з аналізом текстур, кольорових просторів та заднього фону зображення.

5.2 Тестування на власних зразках зображень обличь

Під час тестування системи на власних зразках зображень до веб-камери підносилися зображення зі спуфінг-атаками типу replay attack та printed attack на різних поверхнях друку. Так до веб-камери підносилися зображення обличчя роздруковані на фото-папері, зображення надруковані на пластикових документах та зображення на екрані смартфону. Окремо треба зазначити, що тестування усіх типів атак проходили при різній освітленості кімнати.

Для початку протестуємо який буде результат роботи системи при представлені справжнього обличчя людини, результат на рисунку 5.1. Як бачимо на рисунку 5.1 обличчя успішно виявлено та перевірка на наявність спуфінг-атаки виконана без помилок (напис True означає що обличчя справжнє).

Далі протестуємо роботу системи при спробі спуфінг-атаки типу printed attack. Для початку представимо роздруковане зображення обличчя на фото-папері. Результат можна побачити на рисунку 5.2 і як видно обличчя виявлено успішно так само як і спуфінг-атака.

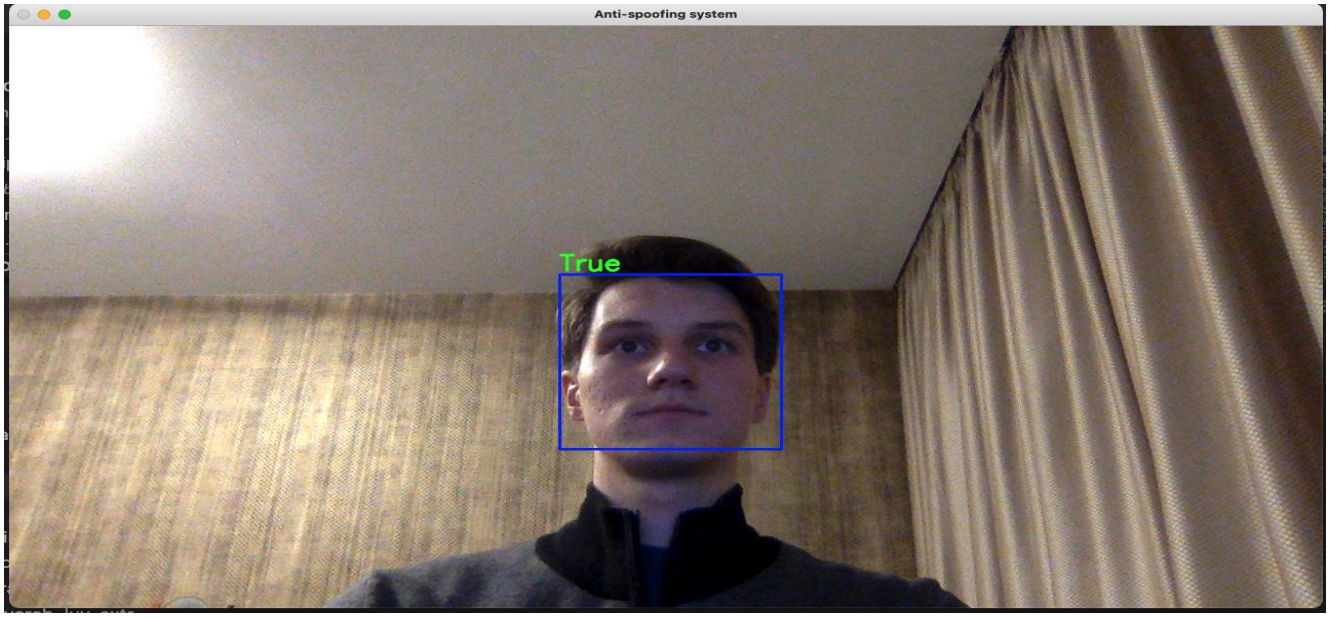


Рисунок 5.1 – Результат роботи системи при представленні справжнього обличчя

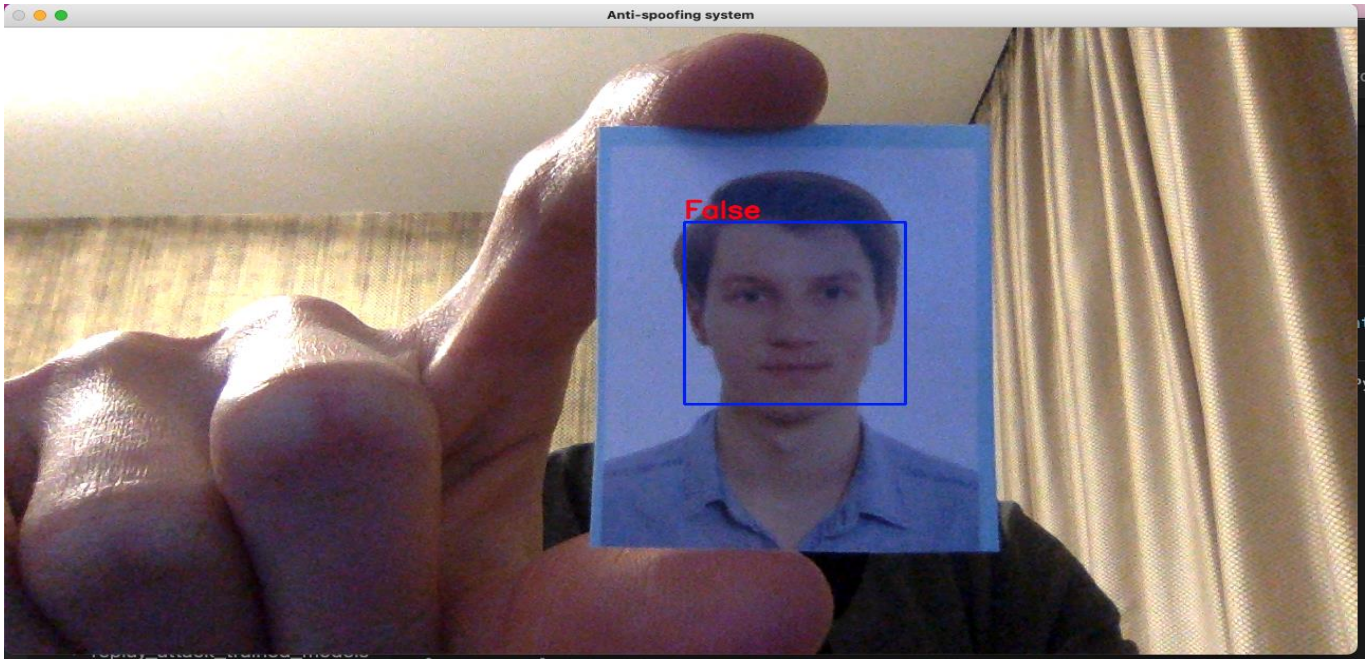


Рисунок 5.2 – Результат роботи системи при printed attack

Також було протестована спроба виявлення printed attack з зображенням роздрукованим на іншому матеріалі, а саме пластику (рис. 5.3).

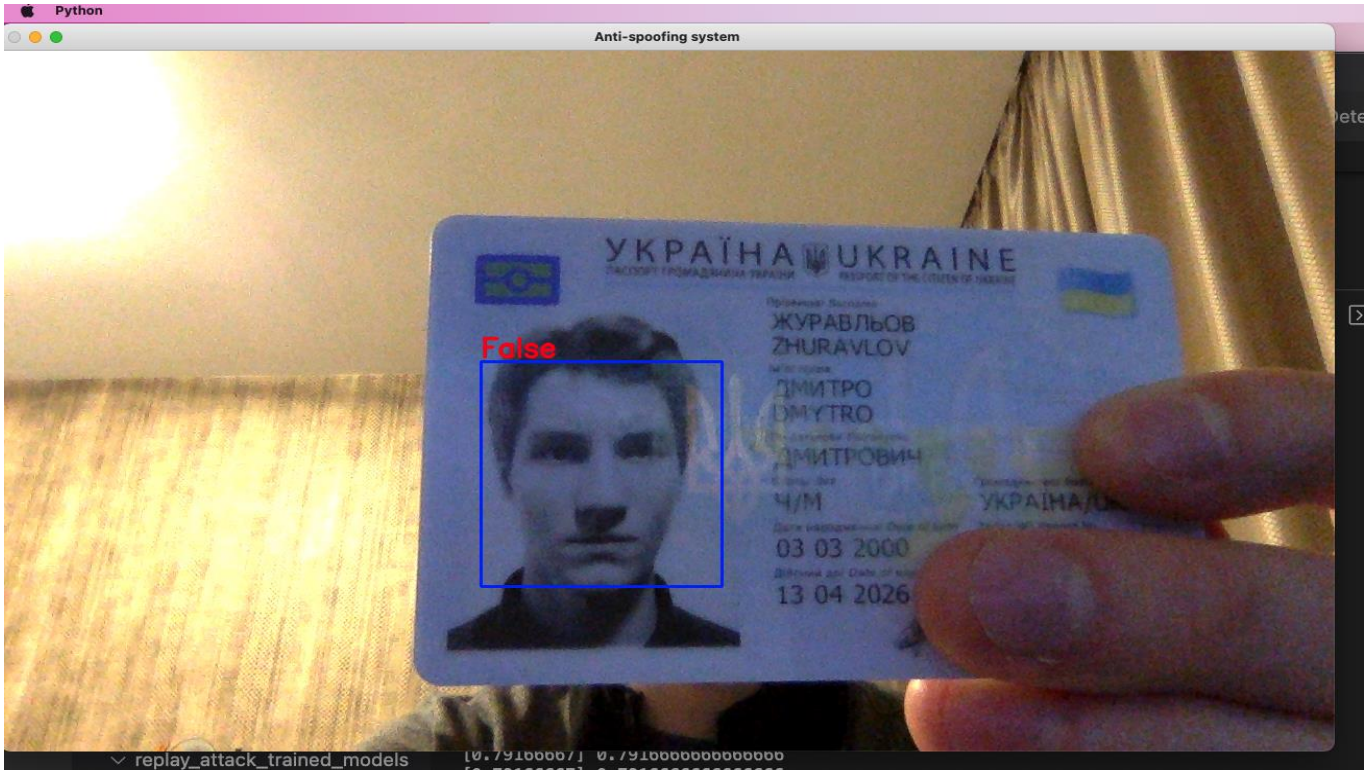


Рисунок 5.3 – Результат роботи системи при printed attack 2

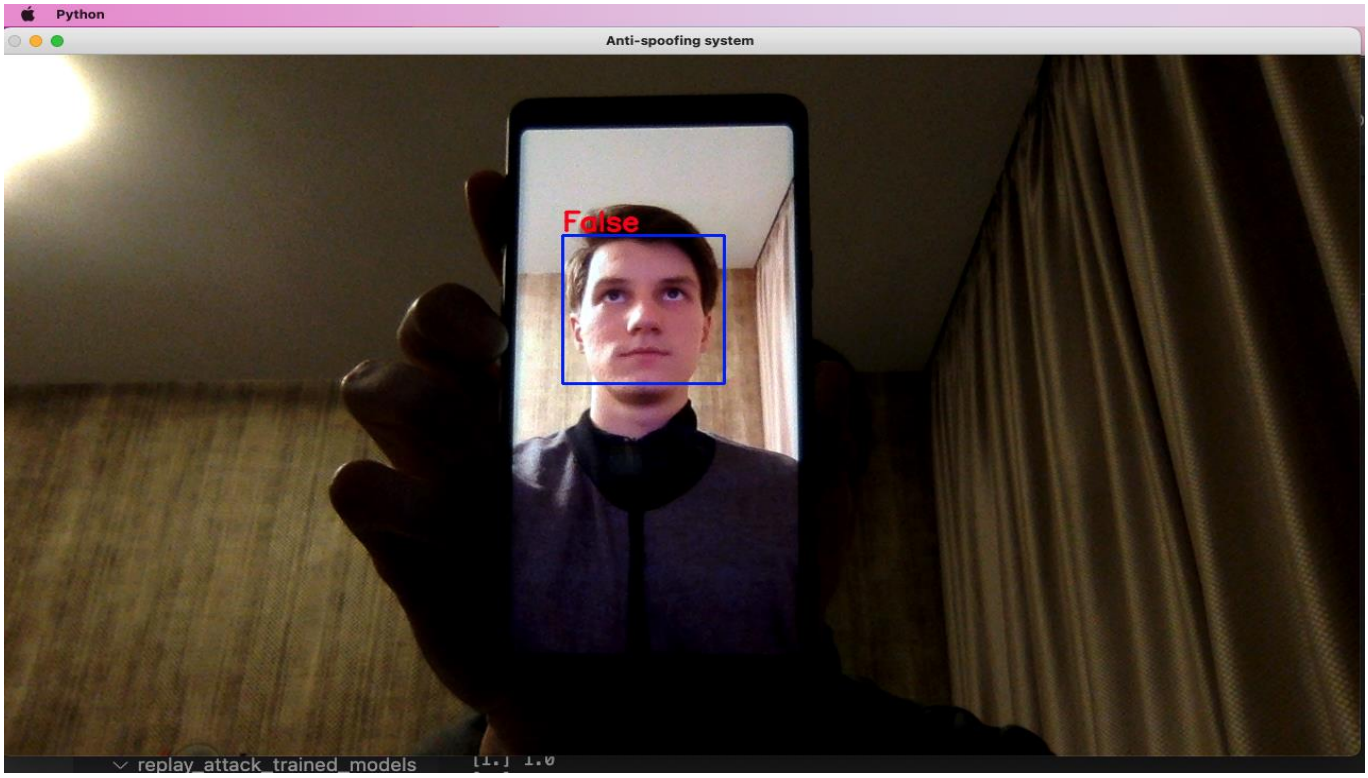


Рисунок 5.4 – Результат роботи системи при replay attack

Далі робота системи була протестована на зразку replay attack. До веб-камери було представлено зображення обличчя людини на смартфоні (рис. 5.4), як можна

побачити обличчя було успішно виявлено так само як і спуфінг-атака типу replay attack.

На рисунках 5.5-5.6 представлені вихідні значення ймовірності виявлення спуфінг-атаки ETC класифікатора. Так на рисунку 5.6 можна побачити, що для типу атаки replay attack значення ETC класифікатора майже завжди дорівнює 1.

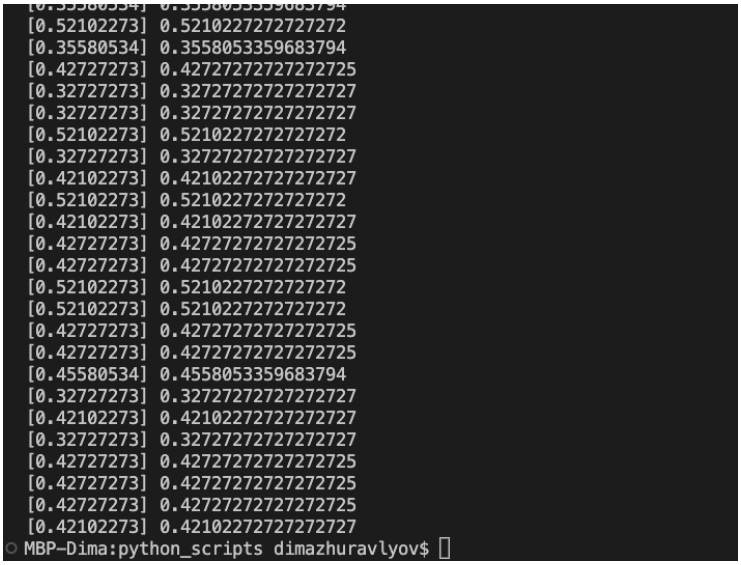


Рисунок 5.5 – Результати ETC класифікатора для справжнього обличчя

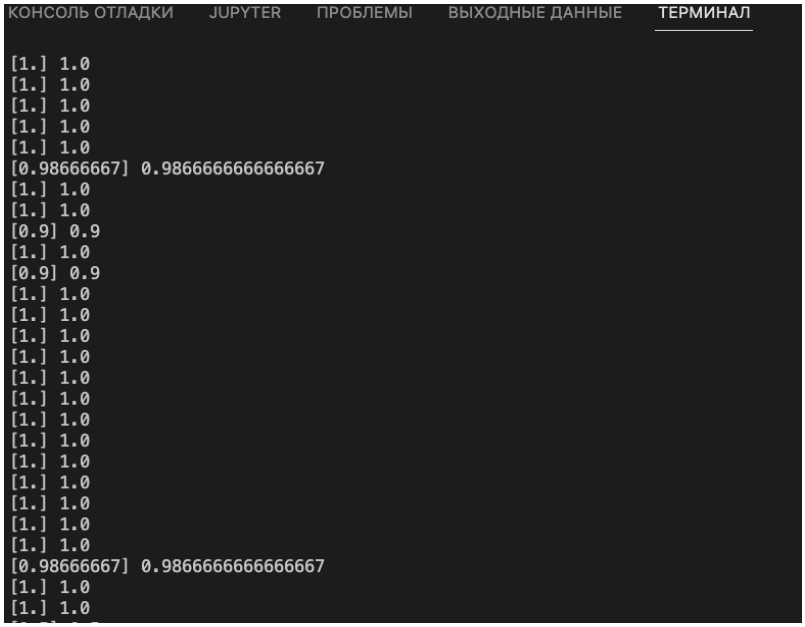


Рисунок 5.6 – Результати ETC класифікатора для replay attack

5.3 Тестування REST API

Для тестування REST API був застосований додаток POSTMAN. Це HTTP-клієнт, який тестує HTTP-запити, використовуючи графічний інтерфейс користувача, за допомогою якого ми отримуємо різні типи відповідей, які згодом можемо перевірити.

Для початку перевіримо, що система захисту REST API з JWT токеном працює коректно і сторонній користувач немає змоги отримати доступ до ресурсів. Відправимо запит на перевірку спуфінг-атаки: POST {server_url}/api/spoofApp/checkForSpoofing, не вказавши JWT токен у хедері запиту. Результат на рисунку 5.7.

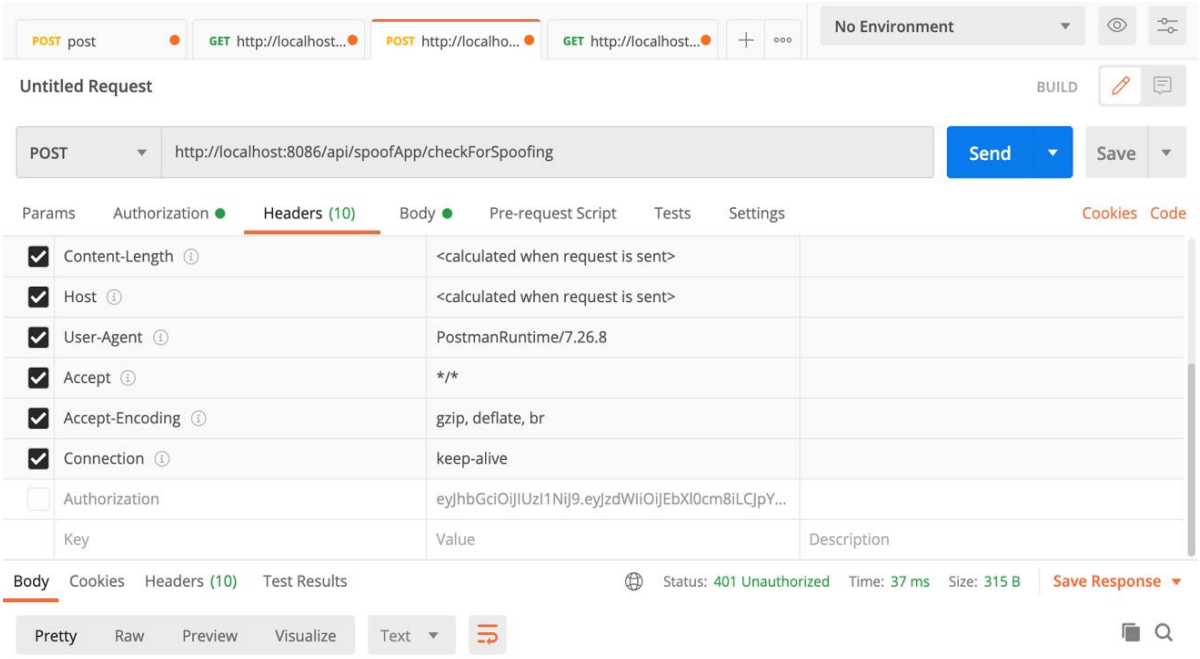


Рисунок 5.7 – Відповідь системи на запит без токену

Як бачимо на рисунку 5.7 при спробі запиту без токену або не з правильним токеном у відповідь надійде HTTP код 401, що свідчить про невдалу авторизацію. Далі для тестування виконаємо авторизацію за допомогою API: {server_url}/api/spoofApp/login метод POST та отримаємо JWT токен. Як видно на рисунку 5.8 API для авторизації працює коректно та повертає необхідні дані визначені для даного ендпоінту, в тому числі і сам токен. Далі повторно виконаємо запит checkForSpoofing для перевірки

зображення на спуфінг та підставимо токен у хедер запиту, під назвою Authorization, і перевіримо, що запит виконається успішно і ми отримаємо відповідь у форматі true/false. Результати представлені на рисунку 5.9, як бачимо API спрацювало успішно.

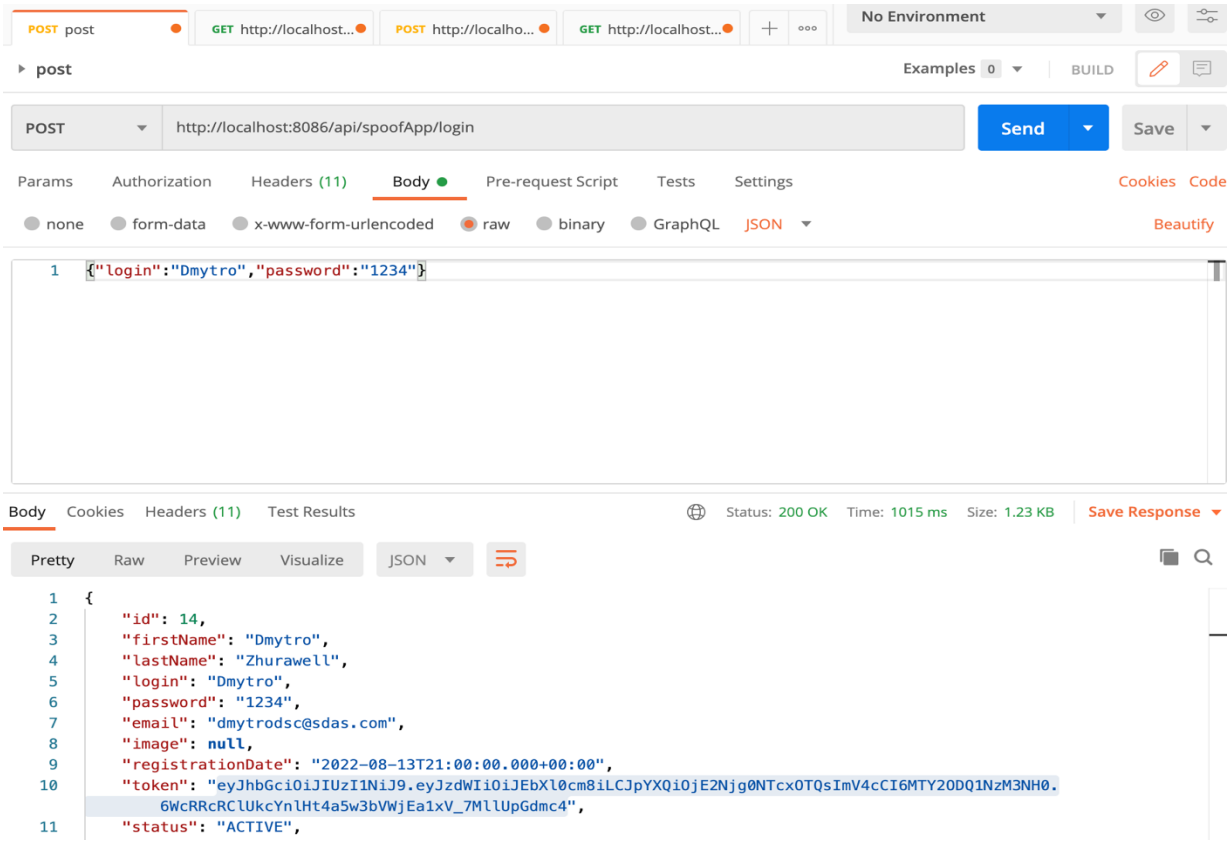


Рисунок 5.8 – Тестування API для авторизації

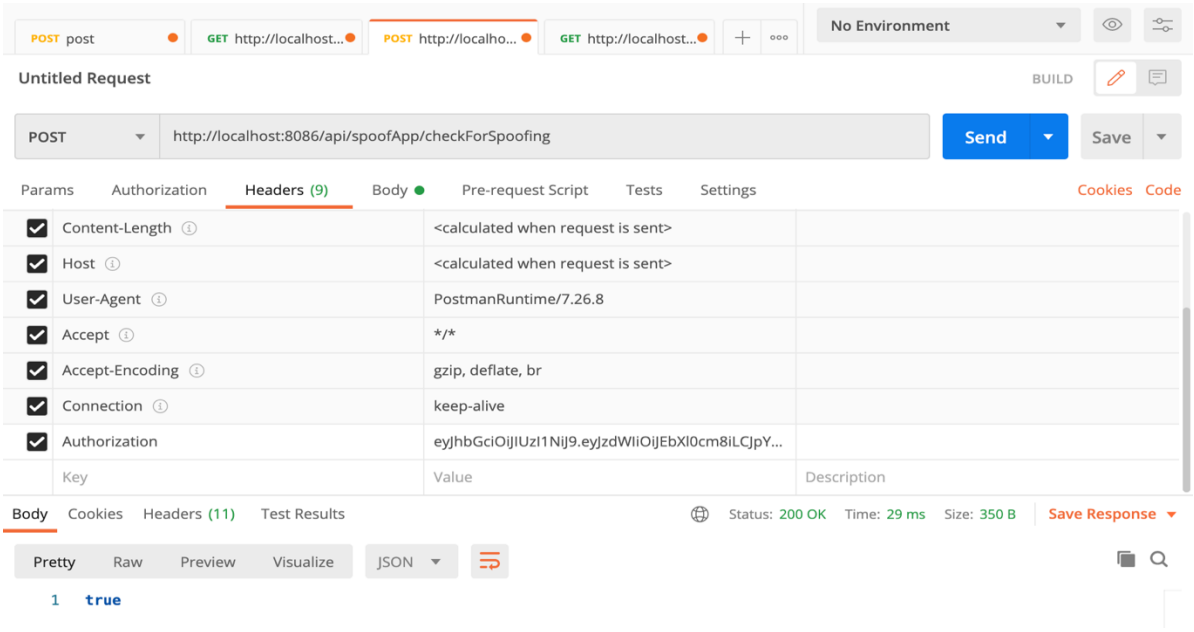


Рисунок 5.9 – Тестування API для перевірки спуфінгу з наявним JWT токеном

Далі перевіримо API для операції пов’язаних з обліковим записом користувача. Для початку перевіримо API для створення нового користувача у системі: {server_url}/api/spoofApp/ user/createUser. Результат на рисунку 5.10, як бачимо все працює відповідно до опису в підрозділі 4.3.

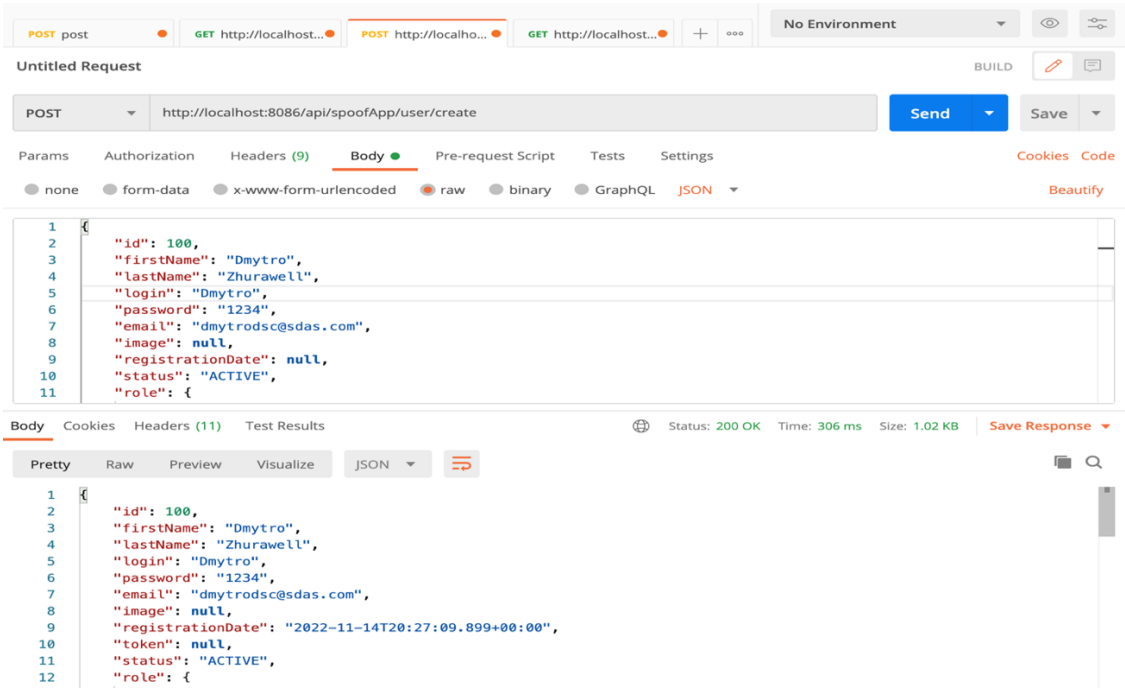


Рисунок 5.10 – Тестування API для створення нового користувача

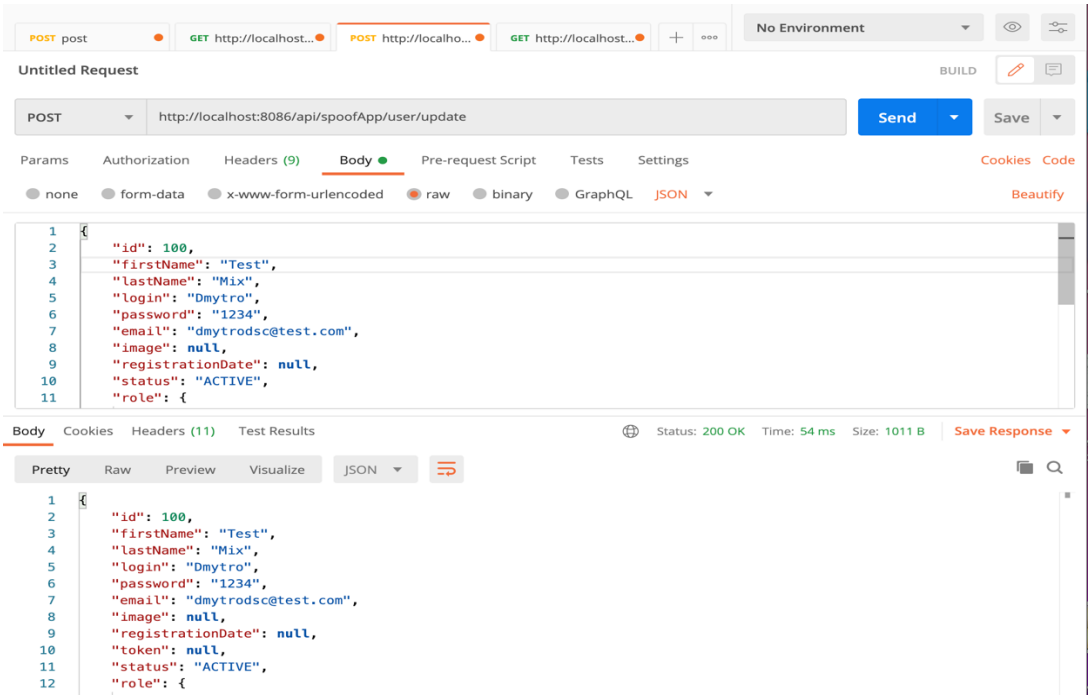


Рисунок 5.11 – Тестування API для оновлення даних користувача

Наступне API яке було протестоване призначене для оновлення даних користувача системи: {server_url}/api/spoofApp/user/update. Як бачимо на рисунку 5.11 API працює без помилок. Далі перевіримо роботу API для пошуку користувача: {server_url}/api/spoofApp/ user/search/{name}/{first}/{last}. Результат на рисунку 5.12.

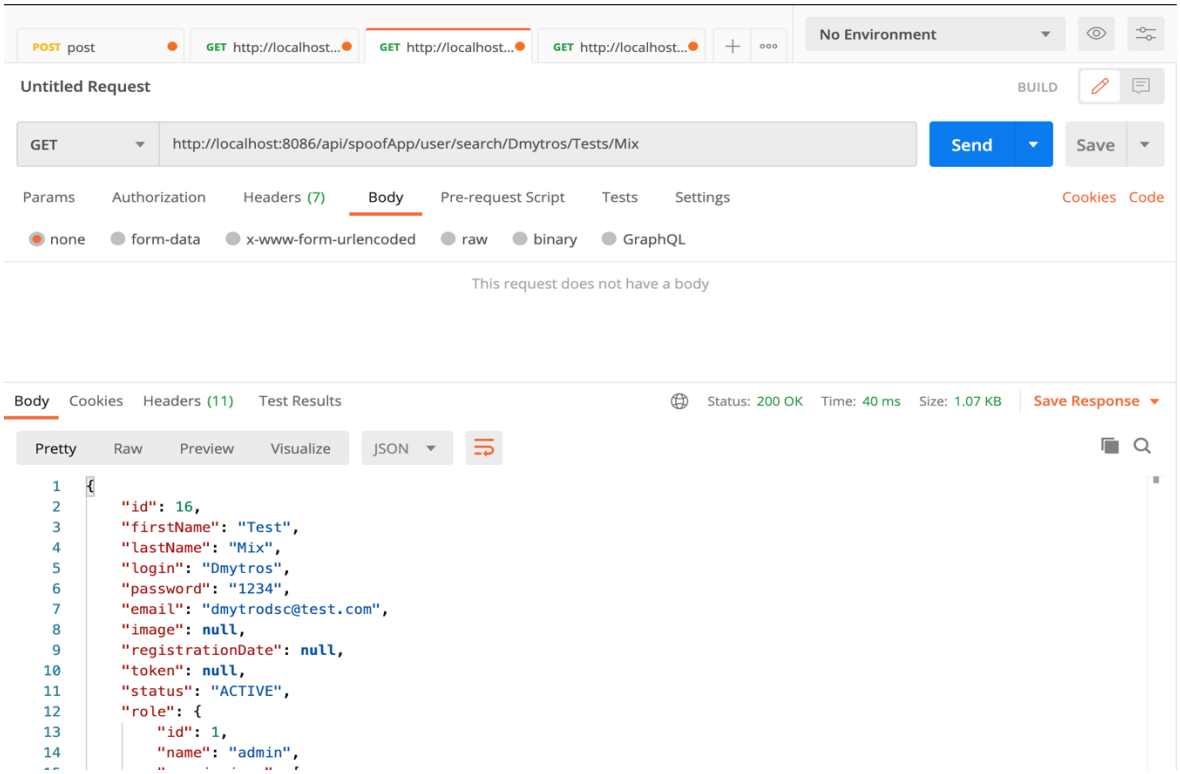


Рисунок 5.12 – Тестування API для пошуку даних користувача

Так само було перевірено API для видалення користувача у системі (рис. 5.13).

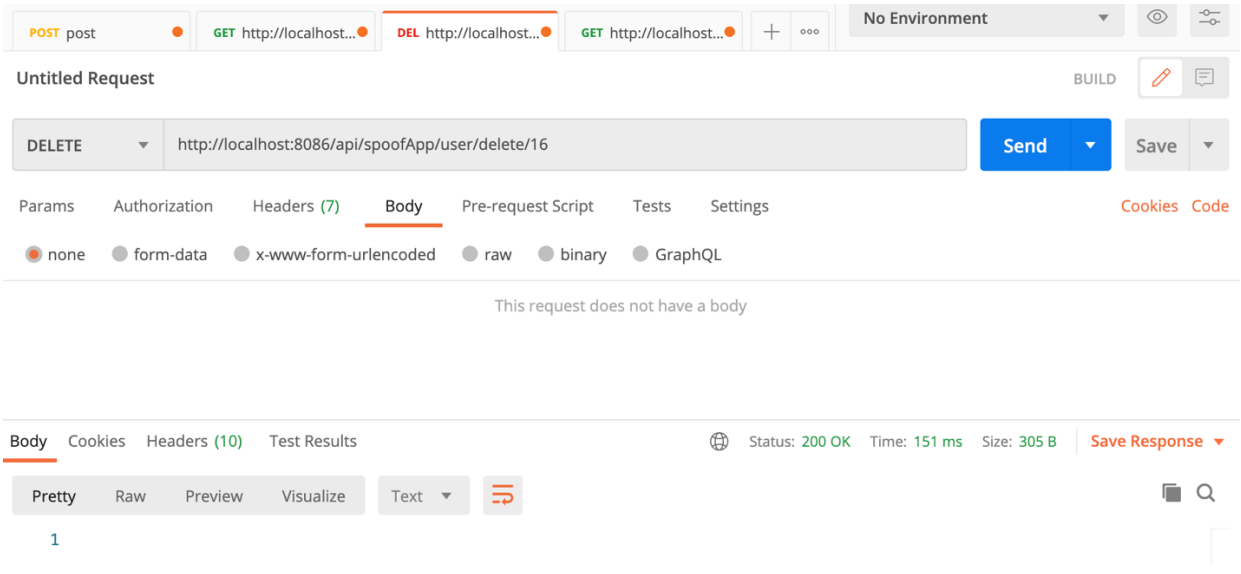
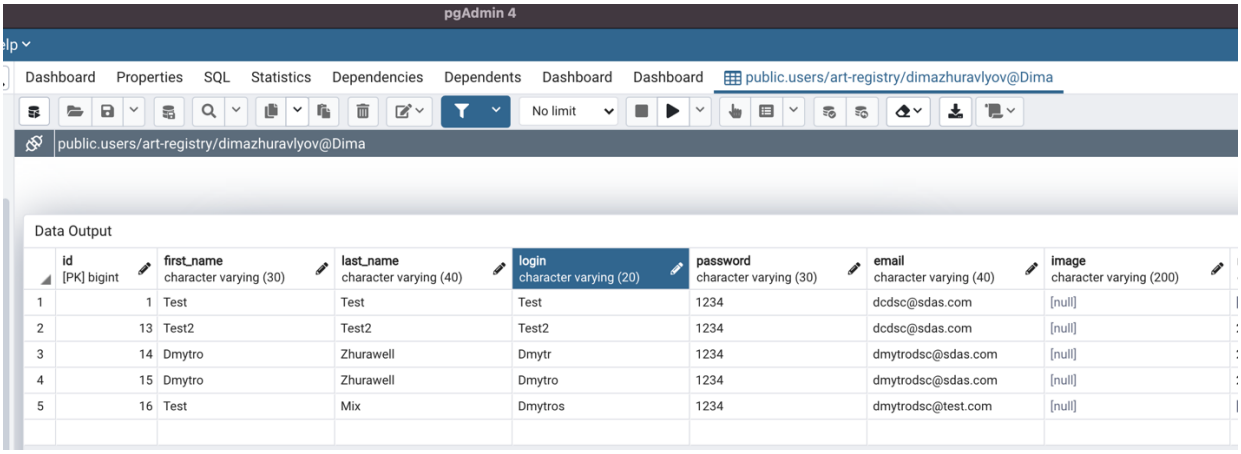


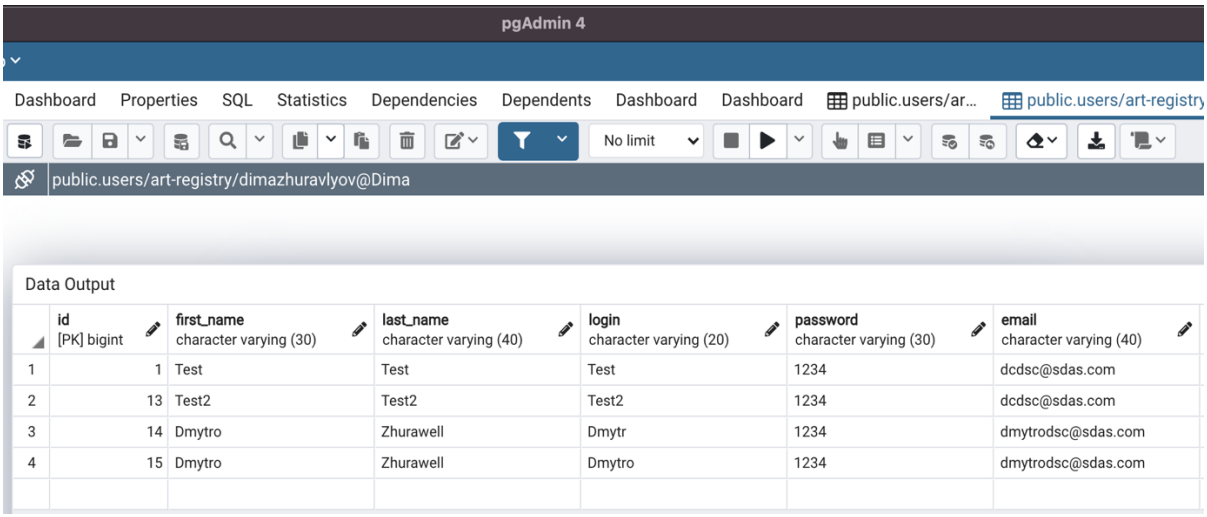
Рисунок 5.13 – Тестування API для видалення даних облікового запису

Як бачимо на рисунку 5.13 API для видалення облікових даних користувача – {server_url}/api/spoofApp/user/delete/{id} працює успішно. Також перевіримо наявність даних у БД. На рисунку 5.14 можна побачити усіх наявних в БД користувачів, що були додані через REST API, а на рисунку 5.15 бачимо відсутність видаленого користувача після тестування API – {server_url}/api/spoofApp/user/delete/{id}.



	id [PK] bigint	first_name character varying (30)	last_name character varying (40)	login character varying (20)	password character varying (30)	email character varying (40)	image character varying (200)
1	1	Test	Test	Test	1234	dcdsc@sdas.com	[null]
2	13	Test2	Test2	Test2	1234	dcdsc@sdas.com	[null]
3	14	Dmytro	Zhurawell	Dmytr	1234	dmytrodsc@sdas.com	[null]
4	15	Dmytro	Zhurawell	Dmytro	1234	dmytrodsc@sdas.com	[null]
5	16	Test	Mix	Dmytros	1234	dmytrodsc@test.com	[null]

Рисунок 5.14 – Вміст БД після тестування REST API



	id [PK] bigint	first_name character varying (30)	last_name character varying (40)	login character varying (20)	password character varying (30)	email character varying (40)
1	1	Test	Test	Test	1234	dcdsc@sdas.com
2	13	Test2	Test2	Test2	1234	dcdsc@sdas.com
3	14	Dmytro	Zhurawell	Dmytr	1234	dmytrodsc@sdas.com
4	15	Dmytro	Zhurawell	Dmytro	1234	dmytrodsc@sdas.com

Рисунок 5.15 – Вміст БД після тестування API видалення користувача

5.4 Висновки до розділу

В розділі описані результати тестування системи на відкритих датасетах спуфінг-атак CASIA FASD та IDIAP, що містять набори Replay та Printed спуфінг-атак, а також результати тестування через веб-камеру комп’ютера на власних даних. Так на датасеті CASIA система надала показник HTER = 0.7%, а на датасеті IDIAP показник HTER = 0.58%. З цих даних можна зробити висновок, що система виконує поставлену до неї вимогу мати показник HTER < 1% та також перевершує за цим показником 90% існуючих систем, що були знайдені автором даної роботи та протестовані на датасеті IDIAP, порівняльні дані були надані у таблиці 5.2. При цьому показник EER(%) дорівнював 0.074 для розробленої системи.

Під час тестування на власних даних через веб-камеру комп’ютера була перевірена здатність системи протистояти до Replay та Printed атак і виявляти справжнє обличчя. Усі тести були пройдені успішно: роздруковане фото, фото на пластику та фото на дисплеї смартфону були помічені як спуфінг-атаки.

Також було проведено тестування REST API під час якого було перевірено функціонування усіх ендпойнтів, в тому числі роботу захисту з JWT токеном. При запиті без токену був отриманий HTTP код 401 та дані не були надані клієнту. Після тестування наявність або відсутність потрібної інформації у БД була перевірена через pgAdmin, усі потрібні записи були зроблені, видалені та модифіковані.

6 ІНТСРУКЦІЯ КОРИСТУВАЧА ТА СЦЕНАРІЇ ВИКОРИСТАННЯ СИСТЕМИ

6.1 Сценарії використання системи

У системи є два актори які використовують функціонал додатку, це – простий користувач та адміністратор. Для початку розглянемо сценарії які може виконувати простий користувач системи. Так простий користувач може виконувати: перевірку зображення на наявність спуфінг-атаки через веб-камеру комп'ютеру; перевірку зображення через REST API; авторизацію для роботи з REST API (таблиця 6.1).

Таблиця 6.1 – Сценарій використання системи користувачем

№	Назва сценарію	Опис	Актори	Тригери	Передумови	Основний розвиток
1	Перевірка на спуфінг-атаку з веб-камерою	Система перевіряє надане через веб-камеру зображення на наявність спуфінг-атаки	Користувач	Користувач підносить зображення обличчя до веб-камери	Наявне зображення обличчя людини	Користувач отримає повідомлення на екрані після перевірки на спуфінг-атаку зображення
2	Перевірка на спуфінг з REST API	Система перевіряє надане зображення на наявність спуфінг-	Користувач	Користувач надсилає REST запит типу POST	Авторизований користувач	Користувач отримає REST response формату true/false і система

		атаки для конкретного користувача				зберігає спробу перевірки.
3	Авторизацію для роботи з REST API	Система перевіряє наявність відповідного користувача за паролем та логіном	Користувач	Користувач надсилає REST запит типу POST	Користувач з створеним обліковим записом у системі	Користувач отримає REST response з JWT токеном для подальшої роботи

Далі розглянемо сценарії для адміністратора системи. Так адміністратор системи виконує наступні сценарії: створення облікового запису користувача; змінення облікового запису користувача; видалення запису користувача; змінення налаштувань системи; авторизацію для роботи з REST API (таблиця 6.2).

Таблиця 6.2 – Сценарій використання системи адміністратором

№	Назва сценарію	Опис	Актори	Тригери	Передумови	Основний розвиток
1	Створення облікового запису користувача	Адміністратор надсилає REST запит для створення облікового	Адміністратор	Адміністратор надсилає REST запит типу POST	Авторизований адміністратор	Дані про обліковий запис користувача збережені у БД

		запису користувача у системі				
2	Оновлення облікового запису користувача	Адміністратор надсилає REST запит для оновлення облікового запису користувача у системі	Адміністратор	Адміністратор надсилає REST запит типу POST	Авторизований адміністратор	Дані про обліковий запис користувача оновлені у БД
3	Видалення облікового запису користувача	Адміністратор надсилає REST запит для оновлення облікового запису користувача у системі	Адміністратор	Адміністратор надсилає REST запит типу DELETE	Авторизований адміністратор	Дані про обліковий запис користувача видаленні з БД
4	Змінення налаштувань системи	Адміністратор вносить зміни до таблиці app_settings	Адміністратор	Застосування відповідного інтерфейсу pgAdmin	Авторизований адміністратор	Дані у таблиці бази даних app_settings змінені

		через pgAdmin				
5	Авторизацію для роботи з REST API	Система перевіряє наявність відповідного користувача за паролем та логіном	Адміністратор	Користувач надсилає REST запит типу POST	Адміністратор з створеним обліковим записом у системі	Користувач отримає REST response з JWT токеном для подальшої роботи

Також для кращого розуміння варіантів використання системи була розроблена діаграма прецедентів, що представлена у додатку Д. Діаграма має двох акторів, а саме: користувач та адміністратор.

6.2 Інструкція з використання системи

Для початку визначимо мінімальні технічні вимоги для ЕОМ, а також вимоги до наявності певного програмного забезпечення (бібліотек, інтерпретаторів, тощо) щоб мати змогу запустити додаток системи. Усі вимоги зведені до таблиці 6.3.

Таблиця 6.3 – Технічні вимоги до ЕОМ та програмного оточення системи

Назва	Вимоги
Операційна система	Windows 7 та вища; MacOS Monteryy версія 12.1 та вища; Linux Ubuntu версія 19 та вища
Оперативна пам’ять	4 гігабайт або більше
Вільного місяця на HDD/SSD	10 ГБ

CPU	Intel Core 2 Duo 2.4 GHz, AMD Athlon X2 2.8 GHz або краще
Відеокарта	512 MB AMD Radeon HD 3870 / NVIDIA GeForce 8800 GT or better або краще
Канали зв’зку	Доступ до мережі інтернет
Інтерпретатор Python	Версія 3.5 або вища
СУБД PostgreSql	Версія 12 або вища
Бібліотека matplotlib для python	Версія 3.1.1 або вища
Бібліотека numpy для python	Версія 1.19.5 або вища
Бібліотека opencv-python для python	Версія 4.4.0.44 або вища
Бібліотека opencv-contrib-python для python	Версія 4.5.1.48 або вища
Бібліотека scikit-learn для python	Версія 0.21.3 або вища
Фреймворк Flask	Версія 2.2.2 або вища
Бібліотека sqlAlchemy	Версія 1.4.44 або вища

Далі розглянемо попередні налаштування які необхідно виконати перед запуском додатку. Перед запуском додатку у таблицю app_settings бази даних необхідно додати наступні обов’язкові записи налаштувань (рис. 6.1), які представлені у таблиці 6.4.

Таблиця 6.4 – Обов’язковий вміст таблиці app_settings

id	name	value	user_id	description
Будь-яке унікальне число в межах даної колонки	threshold	Ціле число від 1 до 100, яке встановлює відсоток ймовірності при якій зображення	null	Будь-який опис

		позначається як спуфінг-атака		
Будь-яке унікальне число в межах даної колонки	Secret_key	Будь-яка строчка від 8 до 50 символів без пробілів, яка буде використовуватися для кодування JWT токена	null	Будь-який опис

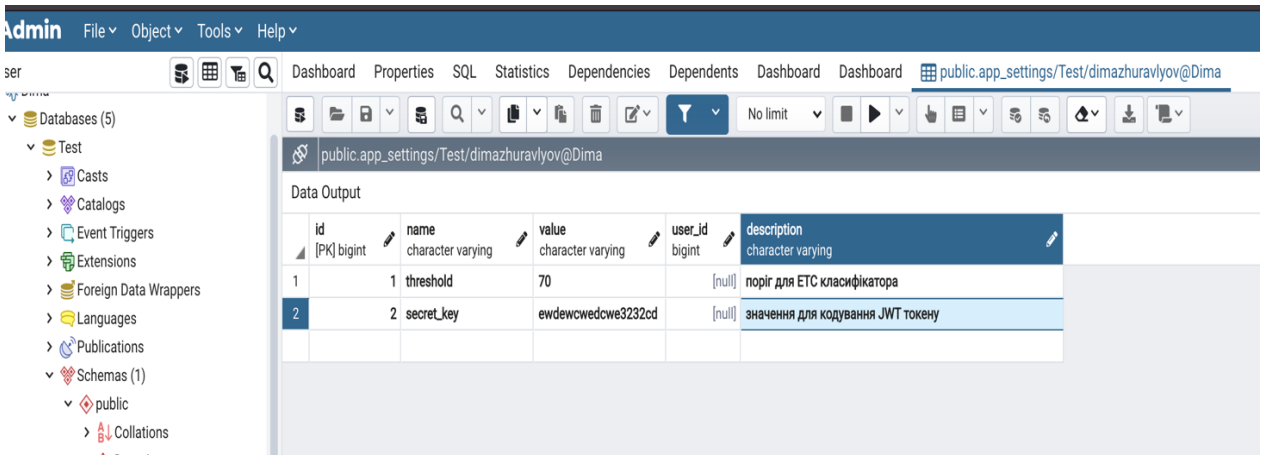


Рисунок 6.1 – Обов’язкові записи у таблиці app_settings

Після того як інформація про налаштування внесена до БД треба переконатися у наявності інтернет з’єднання і можна запускати додаток. Для цього треба розпакувати архів з додатком та відкривши консоль/термінал операційної системи перейти до папки python_scripts, що буде у папці розпакованого архіву spoof_detection_app. Далі за допомогою інтерпретатора python треба запустити головний скрипт main.py з певними обов’язковими ключами (параметрами), а саме:

- Ключ «-n» для вказання шляху до файлу натренованої моделі ETC класифікатора;
- Ключ «-d» для вказання номеру порту до якого підключена веб-камера. Для того щоб обрати веб-камеру за замовчуванням треб а вказати цифру 0.

Якщо всі попередні умови виконані правильно перед користувачем з’явиться вікно у якому буде виведено зображення з веб-камери комп’ютера з відповідною рамкою на обличчі. Для того щоб завершити роботу додатку треба в консолі/терміналі, де був запущений додаток, натиснути клавішу «Esc».

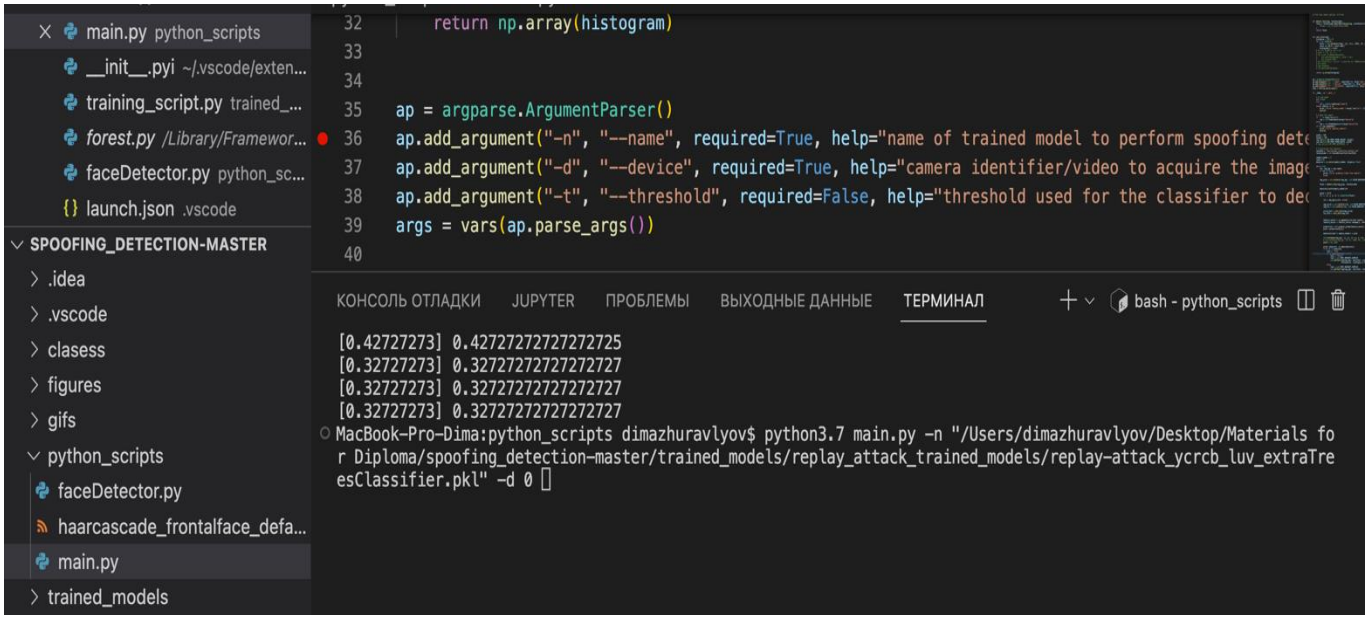


Рисунок 6.2 – Приклад запуску додатку через термінал

6.3 Висновки до розділу

В розділі були визначені усі користувачі системи та наявний для них функціонал з детальним описом усіх дій, що можуть бути виконані. Так система має простого користувача та адміністратора.

Встановлені мінімальні вимоги до апаратного та програмного забезпечення для роботи програмної системи, включно з версіями ОС, інтерпретатора Python та програмних бібліотек Python. Вказані обов’язкові налаштування які необхідно додати до базиданих у таблицю app_settings та параметри додатку які необхідно вказувати при його запуску через термінал/консоль.

7 РОЗРОБКА СТАРТАП-ПРОЕКТУ

7.1 Опис ідеї проекту

Перспективність створення системи з виявлення та протидії спуфінг-атакам у системах ідентифікації за обличчям полягає у тому, що на 2020 рік кількість власників смартфонів у світі досягнула 3,5 млрд людей і з кожним роком це число тільки стрімко зростає [66], і сьогодні головним засобом ідентифікації та автентифікації в смартфонах світових виробників таких як: APPLE, Samsung, Huawei, є саме біометричний захист за допомогою обличчя користувача.

Для кращого розуміння ідеї проекту та переваг, які проект може надати користувачам була розроблена таблиця 7.1.

Таблиця 7.1 – Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Система що здатна визначати чи піднесено до камери справжнє обличчя людини чи підробка і при цьому система можемо працювати як з веб-камерою так і приймати зображення для аналізу через REST API	1. У системах захисту смартфонім	Підвищення надійності захисту особистих даних
	2. У системах що потребують реєстрації користувача з фото	Зменшення часу обробки особистої анкети користувача, так як валідація фото буде проходити без участі людини

Безумовно перед реалізацією стартап-проекту необхідно визначити недоліки та переваги конкурентів, що вже присутні на ринку або оголосили про свою розробку. Сьогодні на ринку є декілька конкурентів, що надають схожий функціонал користувачам, що потребує мати систему виявлення спуфіг-атак. Головним недоліком конкурентів є вартість системи, а також відсутність застосування системи

виявлення спуфінг атак окремо від системи автентифікації. Усі дані стосовно характеристик запропонованої системи та характеристик систем конкурентів наведені у таблиці 7.2.

Таблиця 7.2 – Визначення та порівняння характеристик проекту

№ п/п	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів				W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проект	Apple	Samsung	Microsoft			
1.	Вартість	Низька	Висока	Висока	Висока	-	-	+
2.	Можливість інтеграції з іншими сервісами	Так	Ні	Ні	Так	-	-	+
3.	Можливість працювати на різних платформах	Так	Ні	Ні	Так	-	-	+
4.	Необхідність тільки веб-камери	Так	Ні	Так	Так	-	+	-
5.	Наявність клієнтської бази	Ні	Так	Так	Так	+	-	-

7.2 Технологічний аудит ідеї проекту

Проект у своєму складі передбачає наявність апаратної складової у вигляді веб-камери та програмної. Проект складається з наступних програмних модулів, що

потребують окремих технології: модуль отримання відеопотоку, модуль пошуку обличчя на зображенні, модуль виявлення спуфінг атаки, модуль збереження даних до БД, СУБД. Для оцінку можливості реалізації проекту було проведено технологічний аудит результати якого представлені у таблиці 7.3.

Таблиця 7.3 – Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Сервіс отримання відеопотоку з веб-камери	Мова Python, Бібліотека OpenCV	Наявні, дороблювати не потрібно.	Доступні. Безкоштовний доступ.
2	Сервіс виокремлення обличчя людини на зображенні	Бібліотека OpenCV, каскади Хаара	Наявна, дороблювати не потрібно.	Доступні. Безкоштовний доступ.
3	Сервіс виявлення спуфінг атаки на зображенні	Бібліотеки scikit-learn та numpy	Наявна, дороблювати не потрібно.	Доступні. Безкоштовний доступ.
4	Сервіс збереження даних до БД	Мова Python, бібліотека sqlAlchemy	Наявні, дороблювати не потрібно.	Доступні. Безкоштовний доступ.
5	СУБД	PostgreSql	Наявна, дороблювати не потрібно.	Доступні. Безкоштовний доступ.

Обрана технологія реалізації ідеї проекту: мова програмування Python; бібліотеки: OpenCV, matplotlib, scikit-learn, sqlAlchemy; фреймворк Flask; СУБД postgreSql.				

За результатами аналізу таблиці 7.3 можна зробити висновок, що усі необхідні технології наявні та доступні на ринку, тому технологічна реалізація проекту повністю можлива.

7.3 Аналіз ринкових можливостей запуску стартап-проекту

У даному підрозділі визначені ринкові можливості для кращого розуміння рентабельності проекту та сенсу його впровадження на даний момент часу. Аналіз попиту, розвитку та динаміки ринку представлений у таблиці 7.4.

Таблиця 7.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	Прямих конкурентів немає. Гравців які надають схожі технології на ринку 4.
2	Загальний обсяг продаж, грн/ум.од	394,3 млрд ум.од
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Без обмежень
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі (або по ринку), %	Вкрай висока від 30 до 40%

За даними у таблиці 7.4 можна зробити висновок, що ситуація на ринку є сприятливою для подальшого розвитку проекту.

Далі визначимо потенційні групи клієнтів та їх особливості для того, щоб можна було покращити характеристики з метою задоволення потреб усіх груп хоча б частково. Характеристики для кожної групи клієнтів наведені у таблиці 7.5.

Таблиця 7.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Захист персональних даних на пристроях з біометричною ідентифікацією, в першу чергу на смартфонах та ноутбуках.	Користувачі смартфонів провідних компаній. Користувачі персональних ноутбуків з функцією розблокування за обличчям.	Користувачам смартфонів більш важлива робота системи за різних оточуючих умов (освітленість, нестабільне (розмите) зображення при триманні смартфону в руках.	Швидкість роботи; Простий інтерфейс; Можливість працювати у різних оточуючих умовах.
2	Пришвидшення процесу реєстрації облікових записів користувачів на онлайн ресурсах, що потребують наявності справжнього фото користувача	Онлайн сервіси, що потребують валідації фотографії обличчя користувача.	Для частини клієнтів важлива підтримка DOCKER для розгортання додатку у хмарних сервісах.	Наявність REST API інтерфейсу; Наявність захисту REST API; Можливість перегляду статистики перевірок.

Далі проаналізуємо можливі фактори, що будуть сприяти впровадженню проекту та фактори, що потенційно можуть нашкодити даному процесу. Для аналізу факторів загроз була складена таблиця 7.6, а для факторів, що сприяють розвитку проекту складена таблиця можливостей 7.7.

Таблиця 7.6 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Розвиток конкурента	Створення конкурентом аналогічної незалежної системи виявлення-спуфінг-атак	Пришвидшення власного виходу на ринок; Зниження ціни; Розробка додаткового функціоналу; Зменшення витрат на технічну інфраструктуру за рахунок хмарних сервісів.
2	Фактор кібербезпеки	Атаки на систему через REST API	Покращення захисту за рахунок авторизації з JWT токеном

Таблиця 7.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Вільний ринок	На ринку немає рішення щоб надавало перевірку зображень на спуфінг через REST API	Співпраця з онлайн сервісами, що перевіряють фото користувачів перед створенням облікового запису
2	Збільшення попиту на смартфони/ноутбуки	Зацікавленість компаній у підвищенні надійності систем біометричної	Співпраця з компаніями виробниками смартфонів

	з біометричним захистом	ідентифікації та авторизації у смартфонах/ноутбуках	

Наступним кроком визначимо риси конкуренції на ринку (таблиця 7.8).

Таблиця 7.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Тип конкуренції: Чиста	Узгодження попиту і пропозиції через ціновий механізм.	Формувати ціну нижче ніж у конкурентів
2. За рівнем конкурентної боротьби: Міжнародний	Основні конкуренти це міжнародні компанії із США, Кореї так Китаю.	Розширювати клієнтську базу по всьому світу
3. За галузевою ознакою: Внутрішньогалузева	Конкуренція тільки у межах галузі безпеки інформаційних технологій	Використання останніх технологій розробки РО
4. За характером конкурентних переваг: Цінова та нецінова	Ціна конкурентів більша; У конкурентів немає функціоналу взаємодії з іншими системами через REST API	Розширення інтеграції з онлайн сервісами, що потребують верифікації фото користувачів
5. За інтенсивністю: не марочна	Йде конкуренція саме товарів, а не марок.	Збільшувати функціонал продукту

Далі проведемо більш детальний аналіз умов конкуренції. Для цього була розроблена таблиця 7.9 аналіз конкуренції в галузі за М. Портером.

Таблиця 7.9 – Аналіз конкуренції в галузі за М. Портером

	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
Складові аналізу	В Україні немає. У світі частково Apple, Samsung та Microsoft	Потенційний конкурент Huawei	Постачальники відсутні	Платоспроможність	Товари-замінники Відсутні
Висновки:	Конкуренція несформована	Є потенційні конкуренти з можливістю виходу на ринок. Строки невідомі.	Постачальники відсутні	Клієнти диктують цінову політику	Товари-замінники Відсутні

За даними у таблиці 7.9 можна зробити висновок, що конкурентна присутність проекту на ринку цілком можлива. Для конкурентоспроможності необхідно мати: можливість гнучкої взаємодії з сторонніми системами; також набір засобів для підтримки роботи з хмарними технологіями; гнучку системи цін для різних розмірів підприємст.

На основі попередніх даних з таблиць 7.2, 7.5-7.7 та 7.9 визначимо та обґрунтуємо перелік факторів конкурентоспроможності (таблиця 7.10).

Таблиця 7.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	Ціна	Порівняно з конкурентами ціна впровадження та використання системи є меншою, що безумовно сприяє появі попиту на товар у покупців
2	Простота інтеграції з іншими системами	Наявність REST API значно пришвидшує інтеграцію роботи з іншими система та є унікальною складовою продукту, що відсутня у конкурентів.
3	Масштабованість	За рахунок використанні у системі засобу контейнеризації DOCKER розгортання системи можливе у хмарному середовищі без будь-яких проблем, що робить системи легкомасштабованою.
4	Універсальність	Система є кросплатформеною та може бути розгорнута майже на будь-якому пристрої, що підтримує інтерпретатор Python
5	Швидкість впровадження	За рахунок REST API та підтримці DOCKER система швидко впроваджується

За визначеними факторами конкурентоспроможності був проведений аналіз сильних та слабких сторін проекту. Аналіз представлений у таблиці 7.11.

Таблиця 7.11 – Порівняльний аналіз сильних та слабких сторін проекту

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні						
			–3	–2	–1	0	+1	+2	+3
1	Ціна	5		+					
2	Простота інтеграції з іншими системами	1	+						

3	Масштабованість	15				+			
4	Універсальність	2	+						
5	Швидкість впровадження	15				+			

Останім заходом ринкового аналізу можливостей впровадження проекту буде складання SWOT-аналізу на основі проаналізованих ринкових загроз та можливостей, та сильних і слабких сторін (табл. 7.11). У таблиці 7.12 визначимо сильні та слабкі сторони проекту, а також можливості для розвитку проекту та потенційні загрози.

Таблиця 7.12 – SWOT- аналіз стартап-проекту

Сильні сторони: – Можливість інтеграції з іншими системами; – Універсальність використання на різних апаратних та програмних платформах (операційних системах)	Слабкі сторони: – Малий досвід; – Мала база тестових зразків
Можливості: – Відсутність прямих аналогів проекту, несформована конкуренція; – Великий попит на апарати з системами біометричного захисту	Загрози: – Поява нових конкурентів; – Поява нових технічних засобів для спуфінг-атак

За результатами SWOT- аналізу визначимо альтернативи ринкової поведінки для впровадження стартап-проекту на ринок та можливий оптимальний час їх реалізації, враховуючі можливості потенційних конкурентів, що можуть вивести своє рішення на ринок раніше. Перелік альтернатив ринкової поведінки представлений у таблиці 7.13. Проаналізовавши час та ймовірність отримання ресурсів, для реалізації заходів, ми зможемо обрати для себе найкращу альтернативу.

Таблиця 7.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Залучення реклами у соціальних мережах та на веб-сайтах . Онлайн продажі продукту напряму споживачам.	Висока	1 рік
2	Колаборація з виробником смартфонів/ноутбуків	Висока	2-2.5 рік
3	Представлення продукту на технічних виставках	Середня	1-1.5 роки

На основі даних з таблиці 7.13 можна зробити висновок, що залучення реклами у соціальних мережах і на веб-сайтах та онлайн продажі продукту напряму споживачам є найкращою альтернативою з представлених.

7.4 Розроблення ринкової стратегії проекту

Одна з головних задача, яку потрібно вирішити під час розроблення ринкової стратегії – це визначити цільові групи потенційних користувачів. Для цього була створена таблиця 7.14.

Таблиця 7.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Великі компанії виробники смартфонів та ноутбуків, що застосовують	Готові	Зростаючий попит	Середня	Середня складність. Необхідно багато часу для

	біометричний захист у своїх виробах				тестування системи.
2	Середні та малі інтернет-сервіси, що потребують верифікації фото облікового запису користувача.	Готові	Зростаючий попит	Низька	Низький бар'єр входу
3	Державні установи, що мають системи біометричного захисту	Готові	Низький попит	Середня	Високий бар'єр входу. Жорсткі вимоги до тестування та безпеки даних і документації.
Які цільові групи обрано: Обрано середні та малі інтернет-сервіси, а також великі компанії виробники смартфонів та ноутбуків.					

Далі сформуємо базову стратегію розвитку для обраних сегментів ринку (див. табл. 7.15).

Таблиця 7.15 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку*
1	Колаборація з великими компаніями виробниками смартфонів та ноутбуків, а також надання	Концентрований маркетинг	Наявність зручного функціоналу по взаємодії зі сторонніми системами, а також для роботи з	Стратегія спеціалізації

	окремого сервісу малим та середнім інтернет-компаніям.		хмарними технологіями.	

Наступним кроком визначимо базову стратегію конкурентної поведінки, що представлена у таблиці 7.16.

Таблиця 7.16 – Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1	Так	Шукати нових та забирати існуючих	Не буде копіювати	Стратегія заняття конкурентної ніші

Враховуючі вимоги користувачів з обраних сегментів та обрану базову стратегію розвитку проекту, була розроблена стратегія позиціонування. Стратегія передбачає формування ринкової позиції так, щоб споживачі мали можливість ідентифікувати проект. Стратегія позиціонування сформована у таблиці 7.17.

Таблиця 7.17 – Визначення базової стратегії конкурентної поведінки

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	1. Простий інтерфейс; 2. Безпека даних;	Стратегія спеціалізації	1. Ціна 2. Можливість інтеграції з іншими сервісами	1. Швидкість послуг 2. Простота використання 3. Легкість впровадження

	3. Швидкість роботи; 4. Масштабованість; 5. Ціна		3. Універсальність використання на різних платформах	

7.5 Розроблення маркетингової програми стартап-проекту

На початку розроблення маркетингової програми проекту треба визначити маркетингову концепцію товару. Для цього на основі даних про конкурентоспроможність товару була розроблена таблиця 7.18

Таблиця 7.18 – Визначення базової стратегії конкурентної поведінки

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1	Система захисту від спуфінг-атак у мобільних пристроях	Виявлення спуфінг-атак типу replay attack, printed attack та атака з масками	Кросплатформеність системи; Ціна.
2	Система аналізу зображень на наявність спуфінг-атаки, яка може інтегруватися	Наявність захищеного REST API	Можливість продукту працювати у хмарному середовищі

	з існуючими системами		

Далі розробимо трьохрівневу маркетингову модель продукту для того, щоб потенційні інвестори мали краще уявлення про функціонал та характеристики продукту. Опис трьох рівнів моделі представлений у таблиці 5.19.

Таблиця 7.19 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
I. Товар за задумом	Продукт виконує дві функції: – Як підсистема виявлення та запобігання усіх основних типів спуфінг-атак на системи захисту з використанням біометричних даних обличчя – Як незалежна система-сервіс що дозволяє використовувати свій функціонал з аналізу зображень н а наявність спуфінг-атак через надсилання даних по REST API		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	1. Зручний користувацький інтерфейс 2. Зменшення ціни на обслуговування та експлуатацію 3. Розмір програми на жорсткому диску 4. Швидкість обробки REST запиту 5. Малий показник HTER та EER	-/+	+/+/+/+/+

	Якість: продукт пройшов тестування на двох відкритих наборах даних спуфінг-атак IDIAP та CASIA
	Цифровий файл, zip архів
	Марка: Анти-спуфінг система
ІІІ. Товар із підкріпленням	До продажу: Програмний продукт
	Після продажу: Програмний продукт, технічна підтримка та підписка на збільшений функціонал системи
За рахунок чого потенційний товар буде захищено від копіювання: захищення інтелектуальної власності шляхом патентування	

На наступному етапі визначимо цінові межі для товару. В першу чергу будемо безумовно опиратися на собівартість товару, а далі ціна залежатиме від цін конкурентів, а також платоспроможності цільової групи споживачів (табл. 7.20).

Таблиця 7.20 – Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	-	800 у. о.	Від 1 млн. у. о. на рік	Як REST сервіс: Від 1000 до 10000 грн на місяць. Як підсистема захисту у смартфоні ноутбуку: Від 1000 до 2000 грн за один пристрій.

Далі визначимо систему збуту нашого товару в залежності від специфіки ринку та клієнтських побажань, а також доцільності залучення посередників. Дані представлені у таблиці 7.21.

Таблиця 7.21 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Специфіка передбачає онлайн закупівлі продукту з можливістю підписки на сервіс.	Опис та реклама можливостей товару; Пошук нових клієнтів; Надання пропозицій додаткового функціоналу.	Канал одного рівня	Селективна з використанням комбінованого каналу збуту

Останнім етапом для розробки маркетингової програми є створення концепції маркетингових комунікацій. Враховуючі попередньо обрану основу позиціонування проекту та специфіку поведінки клієнтів була складена таблиця 7.22.

Таблиця 7.22 – Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Використання переважно соціальних мереж та месенджерів для обміну даними та отримання інформації	Мобільні месенджери повідомлень: Webex, Slack, Skype, Telegram, тощо;	Комунікація через соціальні мережі; Вільний доступ до інформації про роботу та розвиток компанії та	Зацікавити компанії у розгляді рекламної презентації та перетворити їх на нових клієнтів.	Вказання переваг простоти та швидкості використання продукту, а також його здатності протистояти

		Офіційна електрона пошта.	впровадження нових технологій.		будь-яким спуфінг-загрозам за рахунок інноваційних технологій

7.6 Висновки до розділу

У висновку можна зазначити, що впровадження даного стартап-проекту на ринок є повністю реалістичним та обґрунтованим з економічної точки зору. На даний час ринок має достатній попит за рахунок малого та середнього бізнесу, що надають інтернет-сервіси, які потребують верифікації фотографії користувача, а також за рахунок великих підприємств, що виробляють смартфони та ноутбуки з функцією біометричного захисту з використанням обличчя людини. Також як було зазначено у підпункті 7.1, попит на такі технології захисту з кожним роком зростає, тому ми можемо казати про позитивну динаміку розвитку ринку у даній сфері.

На сьогоднішній день у запропонованого продукту не існує повноцінних аналогів, які б мали можливість інтеграції з іншими системами через REST-API, що значно посилює конкурентоспроможність проекту. Є схожі «вбудовані» системи виявлення спуфінг-атак у світових корпорацій виробників смартфонів та ноутбуків але їх розробки не є самостійними продуктами, тому конкуренція на ринку є слабкою та сприяє впровадженню проекту.

Найкраща альтернатива впровадження проекту – це колаборація з великими компаніями виробниками смартфонів та ноутбуків, які мають великі бюджети для підтримки та розвитку нових систем, наприклад, Apple, Samsung, Microsoft, Huawei.

Подальша імплементація проекту є доцільна та перспективна з комерційної та технічної точки зору.

ВИСНОВКИ

У роботі була проаналізована актуальність проблеми виявлення спуфінг-атак, яка полягає у тому, що сучасні виробники смартфонів та ноутбуків все частіше використовують біометричні системи захисту з використанням обличчя людини і попит на такі пристрої щороку збільшується. Були визначені усі відомі методи та засоби спуфінг-атак на системи ідентифікації з використанням обличчя людини, а саме: replay attack, printed attack та mask attack. Для кожного виду спуфінг-атаки були визначенні особливості, що дозволяють виокремити цю атаку під час аналізу зображення.

Було розглянуту існуючі підходи виявлення спуфінг-атак, такі як: динамічні, статичні та апаратні. Для кожного підходу у роботі були визначені переваги та недоліки, які були зведені у відповідну таблицю (табл. 1.2). Так само були розглянуті існуючі системи та їх характеристики, такі як: FAR, FRR, HTER, здатність приймати відеопотік та зображення, а також типи атак які система може визначати. Була сформована таблиця переваг та недоліків для кожної системи, що вже існують (розділ 2). Спираючись на аналіз існуючих рішень був обраний статичний підхід виявлення спуфінг-атак із застосуванням перетворення кольорового простору зображення у кольорові простори YC_rC_b і CIE L*u*v* та використанні ETC класифікатору для аналізу об’єднаного вектору гістограм зображень, що саме і визначає наявність спуфінг атаки.

Розроблена система виявлення спуфінг-атак може самостійно виявляти обличчя людини на зображенні завдяки застосуванню каскадів Хаара, що входять до функціоналу бібліотеки OpenCV, яка була використана при розробці. Система передбачає застосування як і веб-камери для отримання зображення обличчя людини з відеопотоку так і REST API для інтеграції зі сторонніми сервісами. В кінцевому результаті розроблена система може працювати як з відеопотоком так і з одним зображенням, виявляти replay attack та printed attack. Усі спроби виявлення атак через REST API зберігаються у базу даних, яка була побудована з використанням СУБД PostgresSql.

Навчання ETC класифікатора та тестування системи було проведено на двох відкритих наборах даних спуфіг-атак CASIA FASD та Idiap REPLAY-ATTACK. Результати тестування надали показник HTER = 0.58 (при EER 0.074). Дані результати були порівнянні з показниками інших систем, інформація про порівняння зведена до таблиці 5.2. За даними у таблиці 5.2 та у розділі 5 був зроблений висновок, що система відповідає усім функціональним та нефункціональним вимогам, що були до неї поставленні та вирішує недоліки, що були у аналогічних рішеннях.

Для системи був проведений аналіз розробки потенційного стартап-проекту. Можна зазначити, що система є актуальною та затребуваною для сьогоднішніх умов технологічного ринку та на неї існує істотний попит з розвитком технологій біометрії у смартфонах та ноутбуках [66]. Були визначені потенційні конкуренти серед технологічних компанії та проведена оцінка конкурентоспроможності проекту. Оцінка є задовільною для ведення конкурентної боротьби на ринку. Визначенні цільові клієнти, а саме: крупні технологічні корпорації, які розробляють смартфону та ноутбуки з засобами біометричного захисту, а також малі та середні компанії, що володіють інтернет-сервісам у яких передбачена верифікація фотографії користувача.

На останок треба зазначити перспективу розвитку системи. Для покращення надійності захисту інформаційних систем та таких показників як HTER, перспективним є шлях об’єднання кількох біометричних показників для ідентифікації та автентифікації користувача, наприклад, поєднання зображення обличчя з відбитками пальців або голосом людини.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ratha, N.K., Connell, J.H., Bolle, R.M.: An analysis of minutiae matching strength. In: Pro-ceedings of the International Conference on Audio- and Video-Based Biometric Person Au- thentication (AVBPA),. Springer-Verlag, 2001.
2. Jain, A.K., Flynn, P., Ross, A.A. (eds.): Handbook of Biometrics. Springer-Verlag, 2008.
3. Li, S.Z., Jain, A.K. (eds.): Handbook of Face Recognition. Springer-Verlag, 2011.
4. Online 3D Printing Service. 2020. URL: www.sculpteo.com/main (дата звернення: 12.11.2022).
5. Development of 3D printers with high precision, reliability and ease of use. 2018. URL: https://www.sharebot.it/en/our-history/ (дата звернення: 12.11.2022).
6. K. Kollreider, H. Fronthaler, and J. Bigun, ‘‘Evaluating liveness by face images and the structure tensor,’’ in *Proc. IEEE Workshop Autom. Identificat. Adv. Technol. (AutoID)*, 2005. 75–80 с.
7. M. de Marsico, M. Nappi, D. Riccio, and J. Dugelay, ‘‘Moving face spoofing detection via 3D projective invariants,’’ in *Proc. IEEE Int. Conf Biometrics (ICB)*, 2012. 73–78 с.
8. A. da Silva Pinto, H. Pedrini, W. Schwartz, and A. Rocha, ‘‘Video- based face spoofing detection through visual rhythm analysis,’’ in *Proc. 25th Conf. Graph., Patterns Images (SIBGRAPI),* 2012. 221–228 с.
9. B. Peixoto, C. Michelassi, and A. Rocha, ‘‘Face liveness detection under bad illumination conditions,’’ in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, 2011. 3557–3560 с.
10. I. Chingovska *et al.*, ‘‘The 2nd competition on counter measures to 2D face spoofing attacks,’’ in *Proc. IAPR Int. Conf. Biometrics (ICB)*, 2013. 1–6 с.
11. I. Pavlidis and P. Symosek, ‘‘The imaging issue in an automatic face/disguise detection system,’’ in Proc. IEEE Workshop Comput. Vis. Beyond Vis. Spectr., Methods Appl, 2000. 15–24 с.

12. Y. Kim, J. Na, S. Yoon, and J. Yi, ‘‘Masked fake face detection using radiance measurements,’’ *J. Opt. Soc. Amer.*, 2009. 760–766 с.
13. L. Sun, W. Huang, and M. Wu, ‘‘TIR/VIS correlation for liveness detection in face recognition,’’ in *Proc. 14th Int. Conf. Comput. Anal. Images Pattern (CAIP),* 2011. 114–121 с.
14. Chingovska I. On the Effectiveness of Local Binary Patterns in Face Anti-spoofing / I. Chingovska, A. Anjos. – Marcel, 2019.
15. Boulkenafet Z. Face anti-spoofing based on color texture analysis / Boulkenafet Zinelabidine – University of Oulu, Finland, 2015.
16. Keyurkumar P. Live Face Video vs. Spoof Face Video / Keyurkumar Patel – Michigan State University, USA, 2020.
17. Anjos A. Motion-based counter-measures to photo attacks in face recognition / Anjos André – Paris, France, 2014.
18. Image analysis and biometrics. 2020. URL: http://iab-rubric.org/papers/PID2777141.pdf (дата звернення: 13.11.2022).
19. Hao-Yu W. Eulerian Video Magnification for Revealing Subtle Changes in the World / W. Hao-Yu. 2012. URL: https://people.csail.mit.edu/mrub/evm/ (дата звернення: 13.11.2022).
20. Atoum Y. Face Anti-Spoofing Using Patch and Depth-Based CNNs / Yousef Atoum / Michigan State University, East Lansing. 2020. URL: http://cvlab.cse.msu.edu/pdfs/FaceAntiSpoofingUsingPatchandDepthBasedCNNs.pdf (дата звернення: 13.11.2022).
21. Song X. Discriminative Representation Combinations for Accurate Face Spoofing Detection / Xiao Song, 2018. URL: https://arxiv.org/abs/1808.08802 (дата звернення: 12.11.2022).
22. Souza L. How far did we get in face spoofing detection? / Luiz Souza / Engineering Applications of Artificial Intelligence. 2018. URL: https://arxiv.org/pdf/1710.09868v2.pdf (дата звернення: 12.11.2022).

23. Keyurkumar P. Cross-Database Face Antispoofing with Robust Feature Representation. 2016. – URL: http://biometrics.cse.msu.edu/Publications/Face/PatelHanJain_FaceAntispoofing_CCBR2016.pdf (дата звернення: 12.11.2022).
24. Si-Qi L. Remote Photoplethysmography Correspondence Feature for 3D Mask Face Presentation Attack Detection. 2018. URL: https://openaccess.thecvf.com/content_ECCV_2018/papers/Siqi_Liu_Remote_Photoplethysmography_Correspondence_ECCV_2018_paper.pdf (дата звернення: 12.11.2022).
25. Inhan Kim, Juhyun Ahn, Daijin Kim: Face spoofing detection with highlight removal effect and distortions. In: 2016 IEEE International Conference on Sys- tems, Man, and Cybernetics (SMC), 2016. 4299-4304 c.
26. Li, L., Feng, X., Boulkenafet, Z., Xia, Z., Li, M., Hadid, A.: An original face anti-spoofing approach using partial convolutional neural network. In: 2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA), 2016. 1– 6 c.
27. Li, L., Correia, P.L., Hadid, A.: Face recognition under spoofing attacks: countermeasures and research directions. IET Biometrics, 2018.
28. Maatta, J., Hadid, A., Pietikainen, M.: Face spoofing detection from single images using texture and local shape analysis. IET Biometrics, 2012.
29. de Freitas Pereira, T., Komulainen, J., Anjos, A., De Martino, J.M., Hadid, A., Pietika¨inen, M., Marcel, S.: Face liveness detection using dynamic tex- ture. EURASIP Journal on Image and Video Processing, 2014.
30. Valter Costa: Image-based Object Spoofing Detection –, Armando Sousa, , and Ana Reis, FEUP – Faculty of Engineering of the University of Porto, Rua Dr. Roberto Frias s/n Porto, Portugal, 2019.
31. Billmeyer, F.W.: Color Science: Concepts and Methods, Quantitative Data and Formulae, 2nd ed., by Gunter Wyszecki and W. S. Stiles, John Wiley and Sons, New York, 1982. 950 c.

32. ITU: ITU-R Recommendation BT.601-5: Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios. Tech. rep., ITU, Geneva, Switzerland, 1995.
33. Великий Я. О. Анализ принципа распознавания объектов на изображении методом Виолы–Джонса / Открытые информационные и компьютерные интегрированные технологии, 2015. 162 – 166 с. URL: http://nbuv.gov.ua/UJRN/vikt_2015_68_22 (дата звернення: 12.11.2022).
34. Access to underlying platform’s identifying data. 2022. URL: https://docs.python.org/3/library/platform.html?highlight=android (дата звернення: 12.11.2022).
35. Object-relational Mappers. 2022. URL: https://www.fullstackpython.com/object-relational-mappers-orms.html (дата звернення: 12.11.2022).
36. Released With Big Performance Improvements, Task Groups For Async I/O. 2022. URL: https://www.phoronix.com/news/Python-3.11-Released (дата звернення: 12.11.2022).
37. Build a Rapid Web Development Environment for Python Server Pages and Oracle, 2006. URL: https://web.archive.org/web/20190402124435/https://www.oracle.com/technetwork/articles/piotrowski-pythoncore-084049.html (дата звернення: 12.11.2022).
38. Python statistics. 2019. URL: https://community.ibm.com/community/user/legacy?lang=en (дата звернення: 12.11.2022).
39. Face Recognition with Python, in Under 25 Lines of Code . 2014. URL: https://realpython.com/face-recognition-with-python/#opencv (дата звернення: 12.11.2022).
40. Flask docs. 2022. URL: https://flask.palletsprojects.com/en/2.2.x/ (дата звернення: 12.11.2022).
41. JSON Web Token. 2015. URL: https://www.rfc-editor.org/rfc/rfc7519 (дата звернення: 12.11.2022).

42. PostgreSQL – Overview. 2019. URL: https://www.tutorialspoint.com/postgresql/postgresql_overview.htm (дата звернення: 12.11.2022).
43. Redis – Overview. 2020. URL: https://www.tutorialspoint.com/redis/redis_overview.htm (дата звернення: 12.11.2022).
44. A. Anjos and S. Marcel, “Counter-measures to photo attacks in face recognition: A public database and a baseline,” in *Proc. IJCB*, 2011. 1–7 с.
45. X. Tan, Y. Li, J. Liu, and L. Jiang, “Face liveness detection from a single image with sparse low rank bilinear discriminative model,” in *Proc. ECCV*, 2010. 504–517 с.
46. Z. Zhang, J. Yan, S. Liu, Z. Lei, D. Yi, and S. Z. Li, “A face antispoofing database with diverse attacks,” in *Proc. ICB*, 2012. 26–31 с.
47. I. Chingovska, A. Anjos, and S. Marcel, “On the effectiveness of local binary patterns in face anti-spoofing,” in *Proc. IEEE BIOSIG*, 2012. 1–7 с.
48. Komulainen, J., Hadid, A., Pietikainen, M., Anjos, A., Marcel, S.: Complementary countermeasures for detecting scenic face spoofing attacks. In: 2013 International Conference on Biometrics (ICB), 2013. 1–7 с.
49. Benlamoudi, A., Aiadi, K.E., Ouafi, A., Samai, D., Oussalah, M.: Face antispoofing based on frame difference and multilevel representation. Journal of Electronic Imaging, 2017.
50. Akhtar, Z., Foresti, G.L.: Face Spoof Attack Recognition Using Discriminative Image Patches. Journal of Electrical and Computer Engineering, 2016. 1–14 с.
51. Zhang, L.B., Peng, F., Qin, L., Long, M.: Face spoofing detection based on color texture Markov feature and support vector machine recursive feature elimination. Journal of Visual Communication and Image Representation, 2018. 51, 56–69 с.
52. Edmunds, T., Caplier, A.: Face spoofing detection based on colour distortions. IET Biometrics, 2018. 27–38 с.
53. Chingovska, I., Yang, J., Lei, Z., Yi, D., Li, S.Z., Kahm, O., Glaser, C., Darner, N., Kuijper, A., Nouak, A., Komulainen, J., Pereira, T., Gupta, S., Khandel Wa, S.,

Bansal, S., Rai, A., Krishna, T., Goyal, D., Waris, M.A., Zhang, H., Ahmad, I., Kiranyaz, S., Gabbouj, M., Tronci, R., Pili, M., Sirena, N., Roli, F., Galbally, J., Ficrrcz, J., Pinto, A., Pedrini, H., Schwartz, W.S., Rocha, A., Anjos, A., Marcel, S.: The 2nd competition on counter measures to 2D face spoofing attacks. In: 2013 International Conference on Biometrics (ICB), 2013. 1– 6 c.

54. Jianwei Yang, Zhen Lei, Dong Yi, Li, S.Z.: Person-Specific Face Antispoofing With Subject Domain Adaptation. IEEE Transactions on Information Forensics and Security, 2015. 797–809 c.
55. Chingovska, I., Rabello dos Anjos, A.: On the Use of Client Identity Information for Face Antispoofing. IEEE Transactions on Information Forensics and Security, 2015. 787-796 c.
56. Maatta, J., Hadid, A., Pietikainen, M.: Face spoofing detection from single images using micro-texture analysis. In: 2011 International Joint Conference on Biometrics, 2011. 1-7 c.
57. Alotaibi, A., Mahmood, A.: Enhancing computer vision to detect face spoofing attack utilizing a single frame from a replay video attack using deep learning. In: 2016 International Conference on Optoelectronics and Image Processing, 2016. 1–5 c.
58. Arashloo, S.R., Kittler, J., Christmas, W.: Face Spoofing Detection Based on Multiple Descriptor Fusion Using Multiscale Dynamic Binarized Statistical Image Features. IEEE Transactions on Information Forensics and Security, 2015. 2396-2407 c.
59. Farmanbar, M., Toygar, O: Spoof detection on face and palmprint biometrics. Signal, Image and Video Processing, 2017. 1253–1260 c.
60. Zhao, X., Lin, Y., Heikkila, J.: Dynamic Texture Recognition Using Volume Local Binary Count Patterns With an Application to 2D Face Spoofing Detection. IEEE Transactions on Multimedia, 2018. 552–566 c.
61. Zhang, L.B., Peng, F., Qin, L., Long, M.: Face spoofing detection based on color texture Markov feature and support vector machine recursive feature elimination. Journal of Visual Communication and Image Representation, 2018. 51, 56–69 c.

62. Karray, F., Campilho, A., Cheriet, F. (eds.): Image Analysis and Recognition, Lecture Notes in Computer Science, 2017.
63. Tu, X., Fang, Y.: Ultra-deep Neural Network for Face Anti-spoofing. In: Neural Information Processing, 2017. 686-695 c.
64. Peng, F., Qin, L., Long, M.: POSTER: Non-intrusive Face Spoofing Detection Based on Guided Filtering and Image Quality Analysis. In: Security and Privacy in Communication Networks, 2017. 774-777 c.
65. Li, L., Correia, P.L., Hadid, A.: Face recognition under spoofing attacks: counter-measures and research directions, 2018. 3-14 c.
66. Number of smartphone subscriptions worldwide. 2022. URL: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ (дата звернення: 12.11.2022).
67. Face id in businesstransactions. 2017. URL: http://www.bkav.com/d/top-news/-/view_content/content/103968/bkav%92s-new-mask-beats-face-id-in-twin-way-severity-level-raised-do-not-use-face-id-in-business-transactions (дата звернення: 12.11.2022).