НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

«До захисту допущено»

В.о. завідувача кафедри
М.В.Грайворонський
(ініціали, прізвище)

“_ _” _____2019 р.

Дипломна робота
освітньо-кваліфікаційного рівня « бакалавр »

з напряму підготовки (спеціальності) 6.170101 «Безпека інформаційних і комунікаційних систем»

на тему: «Застосування сервісів двофакторної автентифікації для побудови хмарних застосувань»

Виконав (-ла): студент (-ка) 4 курсу, групи ФБ-52
(шифр групи)

Самара Наталія Миколаївна
(прізвище, ім’я, по батькові)

Керівник к.т.н., доцент кафедри ІБ Родіонов А.М.
(посада, науковий ступінь, вчене звання, прізвище та ініціали)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент Самара Н.М.

Київ – 2019 року

Національний технічний університет України
«Київський політехнічний інститут»

Факультет (інститут) Фізико-технічний інститут
(повна назва)

Кафедра інформаційної безпеки
(повна назва)

Освітньо-кваліфікаційний рівень бакалавр
(назва ОКР)

Напрям підготовки 6.170101 «Безпека інформаційних і комунікаційних систем»
(код і назва)

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
М.В.Грайворонський
(ініціали, прізвище)

«___»_____2019 р.

ЗАВДАННЯ
на дипломну роботу студенту
Самарі Наталії Миколаївні
(прізвище, ім'я, по батькові)

1. Тема роботи «Застосування сервісів двофактрниї автентифіації для побудови хмарних застосувань»
керівник роботи Родіонов А.М., к.т.н., доцент кафедри ІБ,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «27» травня 2019 р. №1414-с

2. Строк подання студентом робот «10» червня 2019 р.

3. Вихідні дані до роботи Дані мережі інтернет щодо сервісів та застосунків двофакторної автентифікації

4. Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити) огляд роботи застосунків двофакторної автентифікації, аналіз різних факторів автентифікації, аналіз наявних застосунків двофакторної автентифікації, вибір та обгрунтування критеріїв вибору системи двофакторної автентифікації, впровадження обраної системи автентифікації.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Презентація (8 слайдів)

7. Дата видачі завдання 17.09.18

Календарний план

№ з/п	Назва етапів виконання дипломного роботи	Строк виконання етапів роботи	Примітка
1	Вивчення теоретичного матеріалу	18.09.18-20.12.18	Виконано
2	Написання літературного огляду	21.12.18-16.01.19	Виконано
3	Створення загального плану	16.01.19-21.01.19	Виконано
4	Написання першого розділу диплому	01.02.19-19.02.19	Виконано
5	Написання другого розділу диплому	20.02.19-04.04.19	Виконано
6	Проходження переддипломної практики	15.04.19-19.05.19	Виконано
7	Написання третього розділу диплому	12.05.19-22.05.19	Виконано
8	Написання висновків	22.05.19-30.05.19	Виконано

Студент Самара Н.М.
(ініціали, прізвище)

Керівник роботи Родіонов А.М.
(ініціали, прізвище)

РЕФЕРАТ

Тема роботи розкрита в пояснювальній записці обсягом 66 сторінок та складається з трьох розділів. Містить 13 ілюстрацій, 7 таблиць, 16 літературних посилань.

Основна мета даної роботи полягає у розробці методу вибору системи двофакторної автентифікації на основі смартфону. Для цього треба провести дослідження різних методів автентифікації, що використовуються застосунками двофакторної автентифікації, обрати критерії вибору системи двофакторної автентифікації

Об’єктом дослідження виступають сервіси двофакторної автентифікації.

У якості предмета дослідження розглядається метод вибору сервісу двофакторної автентифікації.

При написанні роботи було проведено теоретичний аналіз і узагальнення наукової літератури; запропоновано метод вибору сервісу двофакторної автентифікації.

Результатами роботи є запропонований та апробований метод вибору сервісу двофакторної автентифікації. Впровадження даного сервісу.

Значення результатів роботи полягає у використанні запропонованого методу для вибору кращого застосунку автентифікації для побудови надійних хмарних сервісів.

АВТЕНТИФІКАЦІЯ, ДВОФАКТОРНА АВТЕНТИФІКАЦІЯ, СЕРВІСИ, ДВОФАКТОРНА АВТЕНТИФІКАЦІЯ, СТАНДАРТИ АВТЕНТИФІКАЦІЇ, FIDO, OATH

ABSTRACT

The work theme is opened in the explanatory note in volume of 66 pages and consists of three sections. Contains: 13 drawings, 7 tables and 16 references.

The purpose of this work is develop a method for choosing a two-factor authentication system based on a smartphone. To do this, you need to hold research different authentication methods used by two-factor authentication applications, choose the criteria for choosing a two-factor authentication system.

The object of research is the two-factor authentication services.

As a subject of research deals with the method of choosing a two-factor authentication service

When writing the work theoretical analysis and synthesis of scientific literature was carried out; was proposed the method of choosing a two-factor authentication service.

The results of the work are proposed and tested method of choosing a two-factor authentication service. Implementation of this service.

The value of the work results is to use the proposed method to select the best authentication application for constructing reliable cloud services.

AUTHENTICATION, TWO-FACTOR AUTHENTICATION, SERVICE, TWO-FACTOR AUTHENTICATION, AUTHENTICATION STANDARDS, FIDO, OATH

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

FIDO – Fast IDentify Online

OATH – Initiative for Open Authentication – Ініціатива відкритої автентифікації

OTP – One-Time Password

HOTP – HMAC-Based One-Time Password Algorithm

TOTP – Time-based One-time Password Algorithm

OCRA – OATH Challenge-Response Algorithm

HMAC – Hash-based message authentication code

QR-код – Quick Response Code

API – application programming interface

2FA – Two-factor authentication

ВСТУП

Інтернет є невід’ємною частиною життя сучасної людини. Безперервний стрімкий розвиток даної мережі призводить до збільшення кількості різноманітних веб-застосунків, що використовуються для будь-яких сфер діяльності. Керований доступ до наданих сервісів та персоналізація контенту – в цьому полягає сучасна тенденція створення веб-застосунків.

Поруч з великою кількістю переваг їх застосування, існує і велика кількість недоліків. Вразливості веб-застосунків є одним з найбільш поширених шляхів проникнення в інформаційні системи та витоку особистої інформації користувачів. У зв’язку з цим підвищуються вимоги до автентифікації клієнтів веб-застосунків.

Як було сказано вище, кількість веб-застосунків зростає з кожним днем, і мати окремий логін та пароль для кожного сайту дуже важко для запам’ятовування. Це може призвести або до небезпечного зберігання такої «бази» на персональному комп’ютері користувача, або до використання для всіх сайтів однієї і тієї ж пари (логін, пароль). Існує багато методів підвищення безпеки автентифікації в наш час. Одним з найнадійніших є використання сервісів автентифікації, що розроблені згідно світових стандартів автентифікації.

Незважаючи на простоту у використанні, використання цих застосунків значно сприяє унеможливленню несанкціонованого входу до інформаційної системи.

Актуальність роботи зумовлюється тим, що в ній описуються програмні продукти, спроектовані згідно стандарту розробленого компанією FIDO Alliance, їх впровадження та застосування.

Метою даної роботи є метод вибору системи двофакторної автентифікації на основі смартфону

Для досягнення даної мети було поставлено такі завдання:

- огляд роботи різних методів автентифікації
- аналіз різних факторів автентифікації, та загроз, що можуть призвести до витоку секретної інформації користувачів чи порушення авторизації та автентифікації
- аналіз наявних застосунків двофакторної автентифікації
- вибір та обгрунтування критеріїв вибору системи двофакторної автентифікації
- впровадження обраної системи двофакторної автентифікації

Методами дослідження обрано: опрацювання літератури за даною темою, аналіз технічної документації.

Практичне значення результатів роботи випливає з можливості використання запропонованого методу вибору системи двофакторної автентифікації для побудови надійних веб-застосунків.

Таким чином **об’єктом дослідження** є сервіси двофакторної автентифікації, що розроблені згідно стандарту FIDO Alliance.

Предмет досліджень – застосування застосунків двофакторної автентифікації розроблених за стандартом FIDO Alliance.

1 АВТЕНТИФІКАЦІЯ

1.1. Автентифікація у правовому полі України

В Українському законодавстві існує 12 визначень терміну автентифікація. Найперший з них зустрічається в постанові Кабінету Міністрів України від 20 січня 1997 р. N 40 «Про затвердження Концепції створення Єдиної державної автоматизованої паспортної системи», втратила чинність 15.03.2006. Ця Концепція визначає шляхи, методи і засоби створення Єдиної державної автоматизованої паспортної системи (ЄДАПС) як найважливішої складової частини Державного реєстру населення. Всі інші визначення є більш сучасними і кожне певним чином стосується інформаційних систем, процедури розмежування доступу, а одже і інформаційної безпеки.

Про автентифікацію згадують в постановах, законах, рішеннях, що в свою чергу були схвалені розпорядженнями Кабінетом Міністрів України. Загалом прочитавши будь-яке з визначень зрозуміло що автентифікація - процес електронний, який дає змогу підтвердити належність певного ідентифікатора фізичній чи бридичній особі або інформаційній, телекомунікаційній чи інформаційно-телекомунікаційній системі.

Зважаючи на те що даний термін фігурує в документах, що стосуються Єдиного державного реєстру декларацій осіб, електронної системи охорони здоров’я, електронних довірчих послуг, тощо, та інформації щодо роботи з якою стосуються ці документи (особисті данні користувачів, медична таємния, та інше), можна зробити висновок про необхідність та важливість наявності автентифікації. Проте, дана процедура досі не є стандартизованою в Україні. Більшість європейських країн та США

знаходяться на шляху стандартизації автентифікації електронному середовищі.

1.2 Хмарні сервіси

Умовно всі види хмарних послуг можна поділити на три типи:

- Infrastructure as a Service (інфраструктура як послуга);
- Platform as a Service (платформа як послуга);
- Software as a Service (програмне забезпечення як послуга).

Всі види хмар надаються за моделлю підписки, тобто їх використовують лише за необхідністю. Чудово пояснює суть хмарних послуг концепція Pizza-as-a-Service (Рисунок 1.1)

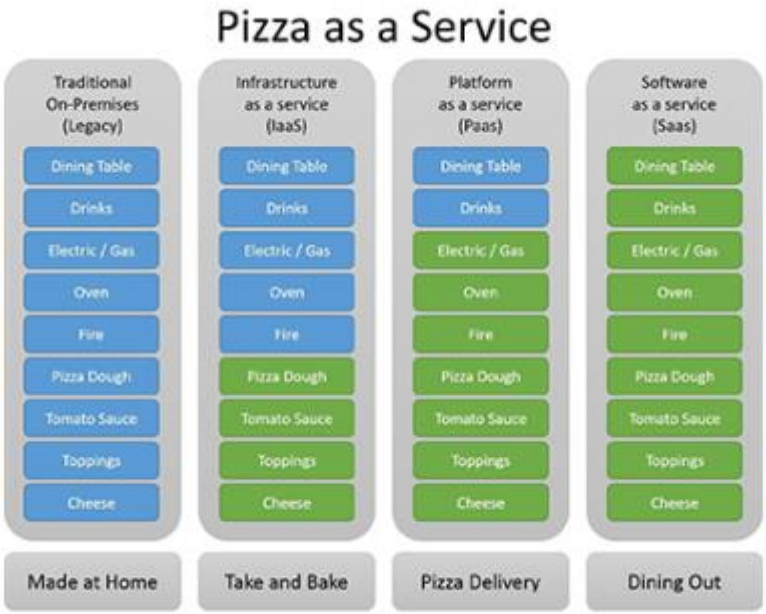

Рисунок 1.1 – Концепція Pizza-as-a-Service [2]

Отже, розглянемо детальніше всі види хмар. **Software as a Service –** перший вид хмарних технологій. Сервіси-представники даного виду хмар є

найбільш відомими та найбільш популярними. Даний вид передбачає надання готового рішення для клієнту з мінімальною необхідністю налаштування. Тобто теоретично, підписуючись на такий сервіс, керувати ним може будь-який користувач. Найвідоміші представники такого сервісу у корпоративному середовищі — це Office 365. Якщо говорити про SMB, то тут варто згадати такі хмарні сервіси як Dropbox, Evernote, Trello та ін.

Послуги типу **Platform as a Service** розраховані в першу чергу на розробників. Вони являють собою набори готових компонентів для створення застосунків, а також фреймворки для керування платформою. В даному випадку компонентами будуть сервіси даних, репозиторії, інструменти автоматизованого деплою, середовища тестування і тому подібні сервіси. Приклади PaaS-сервисів — Google AppEngine, VMWare Pivotal Cloud Foundry, Red Hat's OpenShift, Heroku та ін.

Останній вид сервісу — це **Infrastructure as a Service**. Інфраструктура як послуга за своїми об'єктами та характеристиками найбільш наближена до володіння власним сервером. У випадку з IaaS ви отримуєте у своє розпорядження хмарні процесори, пам'ять, диски та мережі, з яких згодом ви створюєте сервери-маршрутизатори та налаштовуєте мережеву топологію так, як вам необхідно.

Світовий досвід показав, що використання хмарних сервісів надає багато переваг та є дуже гнучним. Завдяки великій потужності користується великим попитом серед розробникві.

1.3FIDO

Приймати загальновизнані світові стандати в якості національних стандартів є світовою практикою. Наприклад серія стандартів ISO, що розроблялися міжнародною неурядовою організацією ISO, діє в Україні в якості серії національних стандартів ДСТУ ISO. Впровадження і використання даних стандартів дають можливість підприємствам України виходити на міжнародний ринок. Автентифікація на основі вільних і відкритих стандартів Альянсу FIDO дозволяє замінювати входи, що містять лише паролі, безпечним і швидким способом входу у веб-сайти та програми. До складу альянсу входять компанії-гіганти, такі як Google та Microsoft.

Альянс FIDO опублікував три набори специфікацій для більш простої, сильної аутентифікації:

- Універсальний другий фактор FIDO (FIDO U2F),
- Універсальна структура автентифікації FIDO (FIDO UAF)
- Протоколи клієнта до аутентификатора (CTAP). CTAP доповнює специфікацію веб-автентифікації W3C (WebAuthn).

Всі протоколи FIDO базуються на криптографії відкритого ключа. Вони забезпечують широкий спектр випадків використання та сценарії розгортання.

1.3.1 FIDO2

FIDO2 складається з специфікації W3C Web Authentication і відповідних протоколів Client-to-Authenticator (CTAP) від FIDO Alliance. FIDO2 підтримує безпаролевий, вторинний і багатофакторний користувальницький досвід роботи з вбудованими (або пов'язаними)

аутентифікаторами (такими як біометричні або PIN-коди) або зовнішніми аутентифікаторами (такими як FIDO Security Keys, мобільні пристрої, тощо). (Рисунок 1.2)

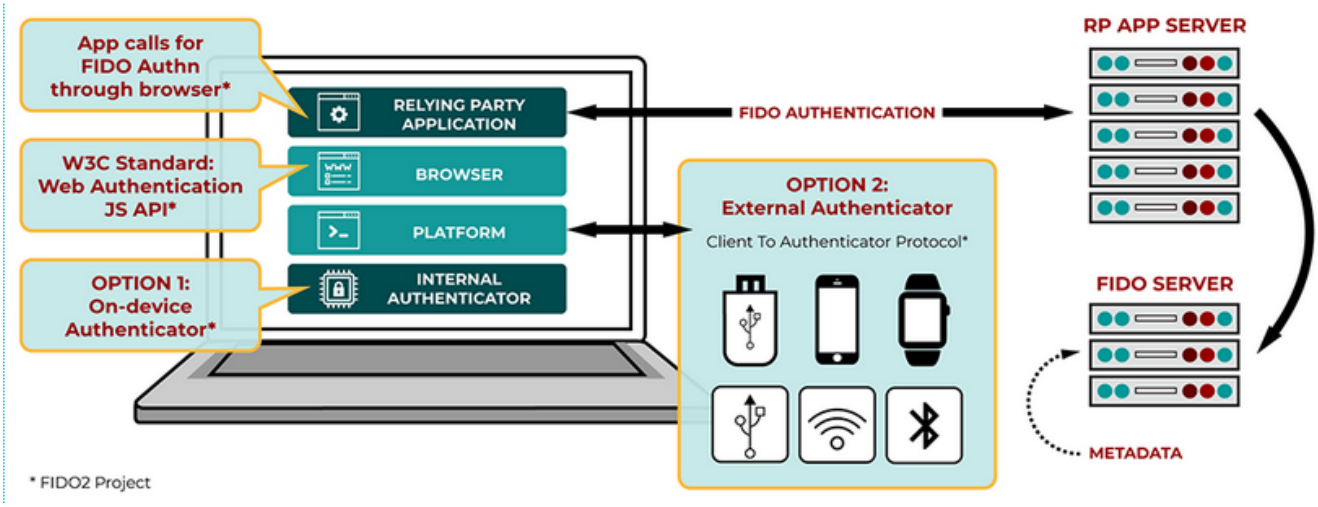

Рисунок 1.2 – Схема автентифікації за FIDO2 [3]

W3C WebAuthn. WebAuthn визначає стандартний веб-API, який вбудовується в браузери і платформи, щоб активувати підтримку автентифікації FIDO.

CTAP2. дозволяє використовувати зовнішні аутентифікатори (FIDO Security Keys, мобільні пристрої) для аутентифікації у браузерах з підтримкою FIDO2 і операційних системах через USB або NFC для безпаролевої чи багатофакторної аутентифікації.

CTAP1. Нова назва FIDO U2F, CTAP1 дозволяє використовувати існуючі пристрої FIDO U2F (наприклад, FIDO Security Keys) для аутентифікації у браузерах з підтримкою FIDO2 і операційних систем з використанням USB або NFC як другий фактор.

1.3.2 FIDO UAF

FIDO UAF підтримує автентифікацію без пароля. FIDO UAF передбачає наявність пристрою з встановленим стеком FIDO UAF у користувача. Необхідно зареєструвати свій пристрій в онлайн-службі, вибравши місцевий механізм аутентифікації, наприклад: провести пальцем, подивитись у камеру, говорити у мікрофон, ввести PIN-код і т.д. Протокол FIDO UAF дозволяє службі вибрати механізми представлені на пристрої користувача. (Рисунок 1.3)

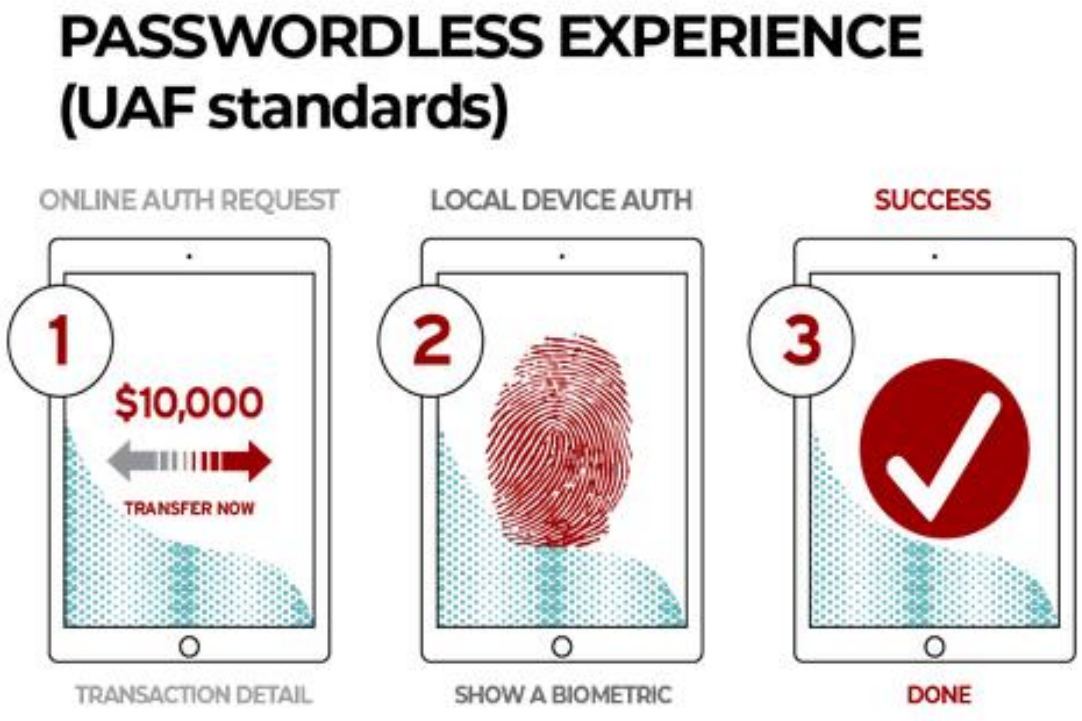

Рисунок 1.3 – FIDO UAF (автентифікація без паролів) [3]

Після реєстрації користувач просто повторює локальну дію аутентифікації, коли потрібно аутентифікувати службу. Користувачеві більше не потрібно вводити пароль під час аутентифікації з цього

пристрою. FIDO UAF також дозволяє використовувати одразу декілька механізмів аутентифікації, таких як відбиток пальця + PIN.

1.3.3 FIDO U2F

FIDO U2F підтримує двофакторну автентифікацію. FIDO U2F дозволяє онлайн-службам підвищувати безпеку існуючої інфраструктури паролів, додаючи другий фактор для входу користувача. Користувач входить до системи з іменем користувача та паролем, як і раніше. Служба також може спонукати користувача надати другий фактор у будь-який час. Сильний другий фактор дозволяє службі спростити свої паролі (наприклад, 4-значний PIN-код) без шкоди для безпеки. (Рисунок 1.4)

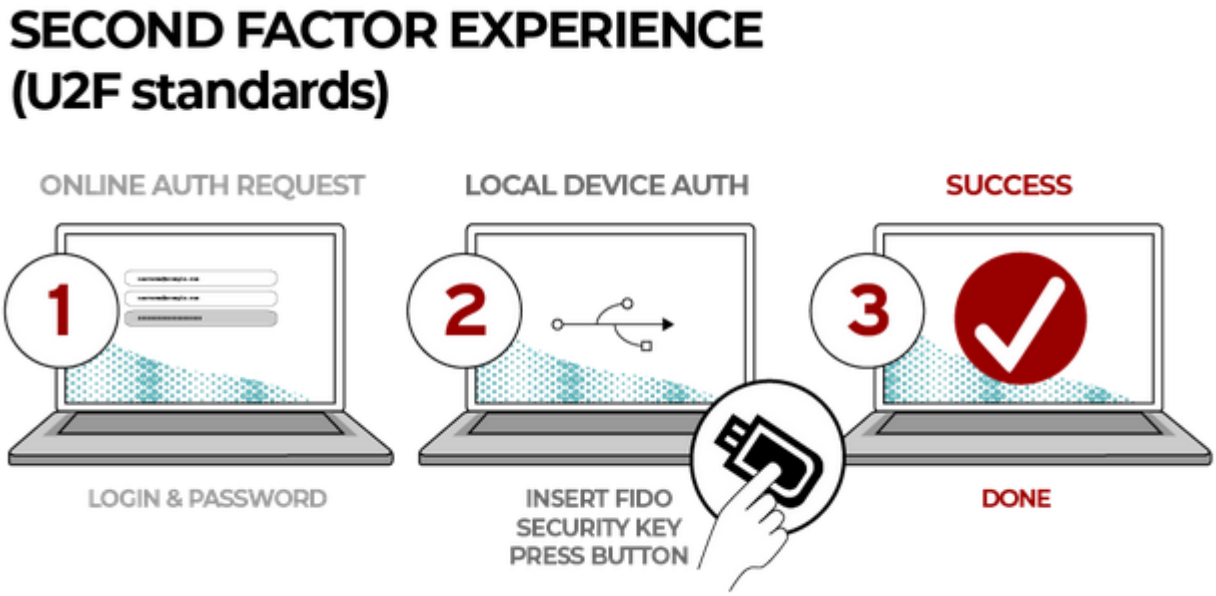

Рисунок 1.4 – FIDO U2F (використання другого фактора) [3]

Під час реєстрації та аутентифікації користувач представляє другий фактор, просто натиснувши кнопку на USB-пристрої. Користувач може використовувати свій пристрій FIDO U2F у всіх онлайнових службах, які підтримують протокол, що використовує вбудовану підтримку в веб-браузерах.

1.4 OATH

OATH – ініціатива відкритої аутентифікації, що надає сертифікацію та рекомендації щодо автентифікації. Не зважаючи на те що сертифікація не є бузкоштовною, стандарти котрі були розроблені організацією є відкритими. Зокрема вони стосуються визначення, побудови та реалізації процедури ватентифікації заснованої на криптографічних алгоритмах, що вважаються стійкими.[4]

На даний момент розроблено три алголгоритми з використанням одноразового пароля (OTP): HOTP, TOTP, OCRA.

HOTP – алгоритм, що генерує одноразові паролі, заснований на HMAC. HMAC – механізм перевірки цілісності інформації, що використовує криптографічні хеш-функції з секретним ключем, що має бути відомий двом сторонам. Даний алгоритм є алгоритмом строгої односторонньої автентифікації. Використовує хеш-функцію SHA-1. Використовує лічильник для синхронізації клієнта та сервера автентифікації. Має генерувати неменше 6 значних паролів для достатньої безпеки. Вхідні параметри алгоритму: значення лічильника, секретний ключ.[5]

TOTP – алгоритм також генерує одноразові паролі, є поліпшенням HOTP. Як і попередній алгоритм є алгоритмом строгої односторонньої автентифікації. Алгоритм генерує паролі на основі часу, поточного інтервалу з заздалегіть встановленими межами, що використовується для синхронізації з сервером. Може бути побудований на основі будь-якої хеш-функції, з секретним ключем, що має бути відомим і клієнту, і серверу. Вхідні параметри алгоритму: інтервал часу, хеш-функція, секретний ключ.[6]

OCRA – алгоритм, що можна вважати розширенням TOTP. Об’єднує в собі можливості односторонньої та взаємної автентифікації. В якості вхідних параметрів додатково використовуються пареметри сесії, сервера, варіант автентифікації, одностороння чи взаємна, та інше.[7]

1.5 Веб-ватентифікація

Веб автентифікація – процедура встановлення належності користувачеві інформації в системі пред'явленого ним ідентифікатора. Вона є частиною процедури надання користувачу доступу до ресурсу. (Рисунок 1.5) В свою чергу процедуру надання доступу поділяють на три етапа:

1) ідентифікація – процедура розпізнавання користувача в системі;

2) автентифікація;

3) авторизація – керування рівнями та засобами доступу до певного захищеного ресурсу.

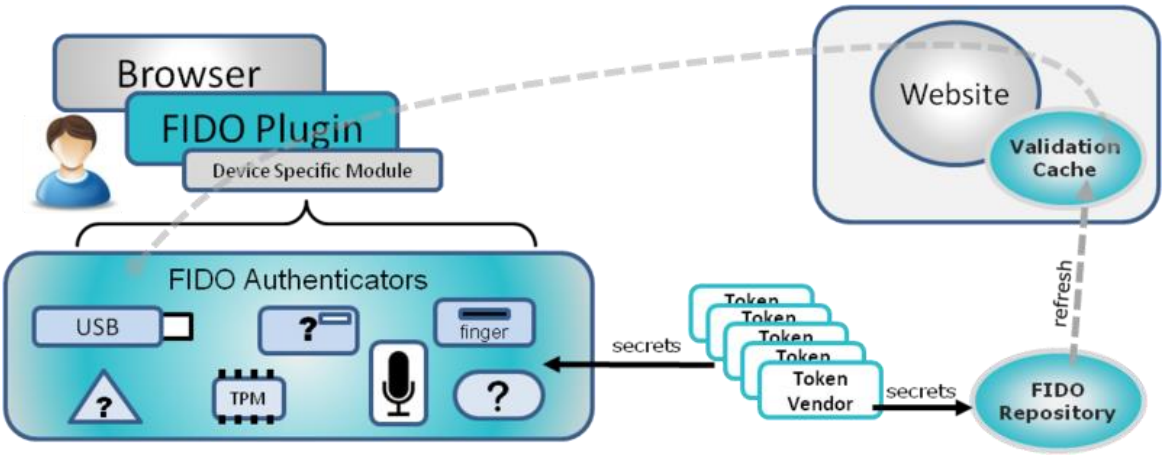

Рисунок 1.5 – Схема процедури автентифікації. [3]

На даний момент існує велика кількість методів веб-автентифікації. Нажаль найзручніший, простий у реалізації та використанні, метод з використанням пароля є водночас найбільш ненадійним. Близько третини інтернет-користувачів вважають таку автентифікацію достатнім захистом особистих данних у веб-застосунках. Розвинуті технології підбору паролю, як єдиного фактору автентифікації, методи соціальної інженерії, певна безпечність поведінки користувачів, з огляду на недостатню інформованість, не залишають жодного шансу для того щоб вважати надійним даний метод автентифікації надійним. Також досить часто люди використовують однакову пару логін та пароль для всіх, чи більшості, застосунків. І тому підвищують ризик злому особистих сторінок одразу у всіх, чи декількох, застосунках.

1.5.1 Класифікації автентифікації

Метди автентифікації діляться на слабку (однофакторну), сильну (багатофакторну) та строгу автентифікацію.

Слабкою називають однофакторну автентифікацію за допомогою пароля. На стійкість даного методу автентифікації значною мірою впливає людський чинник.[8] Контроль складності пароля (проста перевірка паролю на відповідність реегулярному виразу, наприклад «'^(?=.*[0-9].*)(?=.*[a-z].*)(?=.*[A-Z].*)[0-9a-zA-Z]{8,}$'» регулярний вираз при використанні якого пароль має містити хоча б одну цифру, хоча б одну літеру латинського алфавіту нижнього та верхнього регістрів та бути довшим за 8 символів) – спроба «удосконалити» вищеприведений метод автентифікації. З одного боку користувач не зможе поставити дуже легкий для злому пароль, наприклад «1234», тому є більш захищеним від атаки за доомогою пвдбору, з іншого, не надто підвищує надійність методу, адже замість «1234» з’явиться «123456Aa».

Найпростіша атака на авторизацію з таким методом автентифікації - Brute force. Brute force – метод злому облікових записів шляхом підбору пароля шляхом повного перебору. В мережі є вдосталь інформації щоб, навіть не маючи конкретних знань мережевих технологій та програмування, відтворити таку атаку на протязі години. Dictionary attack – Brute force атака, при котрій в якості ймовірних паролів використовують слова зі словника. Окрім того ісують програми аналізатори трафіка, за допомогою яких можна перехопити пакети відправлені з певного хосту, і таким чином пароль, або хеш паролю, може бути перехоплений зловмисником.

Сильною або багатофакторою називають атентифікацію, що використовує декілька факторів автентифікації. Існує всього три типи факторів:

- Фактор знання – певна секретна завчасно визначена інформація (наприклад пароль чи пін-код, або відповідь на секретне питання, тощо)

- Фактор володіння – річ якою ти володієш і можеш це підтвердити (наприклад USB-токен, смарт-картка чи телефон, тощо)
- Фактор властивості – ознака якою ти володієш, власне біометричний фактор (наприклад відбиток пальцю, геометрія руки, райдужка або сітківка ока, тощо) [8]

З точки зору безпеки, найкраще використовувати разом всі три фактори, але це робить процедуру автентифікації незручною та тривалою в часі. Зазвичай використовують лише два фактора автентифікації, тому багатофакторну ще називають двофакторною. Розглянемо атаку на популярну нині дфвофакторну автентифікацію, що використовує пароль як перший фактор та одноразовий пароль, що надзодить користувачу в SMS-повідомленні. Таку автентифікацію використовують інтепнет-банкінги. Для успішного проведення атаки необхідно знати пароль та володіти вказаним при реєстрації номером телефону (СІМ-карткою), чи власне телефоном жертви. Вище було описано яким чином підбираються та перехоплюються паролі. Для отримання одноразового паролю використовують методи соціальної інженерії, наприклад фішинг.

Строга автентифікація є разновидом багатофакторної автентифікації. Ідея строгої аутентифікації, що реалізується в криптографічних протоколах, полягає в наступному. Користувач доводить свою автентичність інформаційній системі демонструючи знання будь-якого секрету, який, наприклад, може бути попередньо розподілений безпечним способом між сторонами автентифікаційного обміну. На даний момент строга автентифікація є найбільш стійкою та незвадаючи на це вона не є стійкою до атак що аправлені на мережу чи власне апаратне забезпечення.[8]

Криптографічні протоколи автентифікації передбачають наявність двох сторін: сторона що перевіряє автентичність, та сторона що доводить свою

автентичність. Протоколи в свою чергу поділяються на односторонні та двосторонні. *Протокол односторонньої автентифікації* (one-way authentication) передбачає що одна сторона перевіряє автентичність іншої. При використанні *протоколу двосторонньої (взаємної) автентифікації* (mutual authentication) кожна сторона одночасно перевіряє автентичність іншої і доводить свою автентичність.

Висновки до розділу 1

В наш час автентифікація є важливою процедурою для забезпечення конфіденційності цілісності та доступності інформації користувачів будь-яких систем та ресурсів. Вона використовується повсякчас та всюди для обмеження та здійснення контролю доступу, як фізичного доступу до обєкту, так і доступу до інформації в системі. Стандартизація систем автентифікації значно спрощує побудову застосунків, що використовують сильну та строгу автентифікацію і допомагають розробникам, за допомогою мінімальної кількості часу, зробити систему більш захищеною.

2 ОГЛЯД ЗАСТОСУНКІВ ДВОФАКТОРНОЇ АВТЕНТИФІКАЦІЇ

2.1 Введення

Як було сказано вище найбільш надійною вважається строга автентифікація. Ідея строгої аутентифікації, що реалізується в криптографічних протоколах, полягає в наступному. Користувач доводить свою автентичність інформаційній системі демонструючи знання будь-якого секрету, який може бути попередньо розподілений безпечним способом між сторонами автентифікаційного обміну. Для впровадження строгої автентифікації використовують сервіси автентифікації. Реалізації представлені компаніями-гігантами, наприклад, як Google та Microsoft надають впевненість в їх коректності та стійкості обраних ними факторів автентифікації. Далі розглянемо чотири найпопулярніших сервіса.

2.2 Google Authenticator

Google Authenticator – мобільний застосунок, що використовується для виконання двофакторної автентифікації, в облікових записах Google та сторонніх сервісах. Реалізований для декількох мобільних платформ, не має можливості ініціалізації на декількох пристроях. Секретний ключ можна зісканувати в застосунок як QR-код, або ввести вручну. Налаштування в застосунку представлені лише синхронізацією часу з серверами Google, на випадок якщо ви не можете увійти в обліковий запис з кодом із застосунку.

Автентифікатор генерує 6-ти або 8-мизначний одноразовий пароль, з використанням відкритих стандартів алгоритмів HOTP та TOTP,

розроблених компанією OATH. Дані паролі використовуються в якості другого фактору автентифікації і вводяться після коректного введення логіну та паролю. Пароль дійсний протягом 30 секунд, що запобігає використанню кілька разів.

Останні версії застосунку є приватною власністю Google, попередні версії програми були доступні з відкритим вихідним кодом на GitHub.

Для того щоб вбудувати в застосунок використовується API. Спочатку необхідно отримати ключ зареєструвавши свій застосунок в Google Developers Console. Потім викликати gapi.client.init(), передавши певні параметри, на сторінці авторизації. Для проведення автентифікації застосунку необхідне з'єднання з серверами автентифікації Google. [9-10]

Загальні дані стосовно даного застосунку приведено у Таблиці 2.1.

Таблиця 2.1 – Дані про застосунок Google Authenticator

Алгоритми генерації одноразових паролів	TOTP та HOTP
Сертифікація алгоритмів компанією OATH	Ні
Принцип роботи	6-ти або 8-мизначний одноразовий пароль в якості другого фактора
Безкоштовний	Так
Розробник	Google
Open source	до певної версії

Продовження таблиці 2.1

Операційні системи	Android, BlackBerry, iOS, J2ME.
Шифровані резервні копії	Ні
Версія для ПК	Ні
Використання на декількох дивайсах	Ні
Розмір застосунку	4 Мб
Оцінка на Google Play	4.2
Кількість завантажень на Google Play	10 000 000+
Вибір мови інтерфейсу	Так
Захист від скріншотів	Ні
Налаштування часового інтервалу дійсності пароля	Ні

2.3 Microsoft Authenticator

Microsoft Authenticator – мобільний застосунок допомагає входити в облікові записи, виконуючи двофакторну автентифікацію. Працює з будь-

яким обліковим записом, який використовує двофакторну автентифікацію та підтримує одноразові паролі (TOTP).

Microsoft Authenticator можна використовувати кількома способами, включаючи:

- Після входу з іменем користувача та паролем в застосунок надходить запит на автентифікацію.
- Вхід без введення пароля, використовуючи ім’я користувача, застосунок автентифікації та мобільний пристрій, використовуючи відбиток пальця, обличчя або PIN-код.
- Як генератор коду для будь-яких облікових записів, які підтримують програми автентифікації.

В якості генератора кодів Microsoft Authenticator генерує шестизначний пароль, який відображається під кожним доданим обліковим записом. Пароль дійсний протягом 30 секунд, що запобігає використанню коду кілька разів. Ініціалізація облікового запису проходить шляхом сканування QR-коду, або введення коду вручну. Не має можливості ініціалізувати один обліковий запис в декількох застосунках на різних пристроях одночасно.[11]

Загальні дані стосовно даного застосунку приведено у Таблиці 2.2.

Таблиця 2.2 – Дані про застосунок Microsoft Authenticator

Алгоритми генерації одноразових паролів	TOTP
Сертифікація алгоритмів компанією OATH	Ні

Продовження таблиці 2.2

Принцип роботи	1 - розпізнавання облич, сканер відбитку або введення пін, замість паролю 2 - 6-ти або 8-мизначний одноразовий пароль в якості другого фактора 3 - підтвердити в застосункув якості другого фактора
Безкоштовний	Так
Розробник	Microsoft
Open source	Hi
Операційні системи	Android і iOS.
Шифровані резервні копії	Так
Версія для ПК	Hi
Використання на декількох дивайсах	Hi
Розмір застосунку	6.7 Мб
Оцінка на Google Play	4.5
Кількість завантажень на Google Play	10 000 000+
Вибір мови інтерфейсу	Так

Продовження таблиці 2.2

Захист від скріншотів	Ні
Налаштування часового інтервалу дійсності пароля	Ні

2.4 Authy 2FA

Authy 2-Factor Authentication – мобільний застосунок допомагає входити в облікові записи, виконуючи двофакторну автентифікацію. Працює з будь-яким обліковим записом, який використовує двофакторну автентифікацію.

Існують наступні типи аутентифікації:

- OTP (Одноразовий пароль) забезпечує захист SMS або голосового дзвінка 2FA. Забезпечує кращий захист, ніж лише ім'я користувача та пароль.
- Генерація кодів на основі алгоритму TOTP, немає необхідності в підключені до стільникової мережі або мережі передачі даних.

В якості генератора однорвзових паролів застосунок від Authy генерує шестизначний пароль, який використовується як другий фактор автентифікації, після введення імені користувача та пароля. Він дійсний протягом 30 секунд. Ініціалізація облікового запису проходить шляхом сканування QR-коду, або введення коду вручну

Застосунок потребує авторизації на сервері Authy за номером телефону, для створення облікового запису, в разі його відсутності. Це дає змогу використовувати застосуног одночасно на декількох пристроях, що

будуть синхронізовані. В разі втрати пристрій деавторизувати з будь-якого іншого авторизованого пристрою.

Authy доступно для мобільних пристроїв Android і iOS, Windows, Apple Watch. Захист 2FA Authy також доступний як розширення для веб-переглядача. На відміну від попередніх, сервіс від Authy дозволяє зберігати зашифровані резервні копії у хмарі. [12-13]

Загальні дані стосовно даного застосунку приведено у Таблиці 2.3.

Таблиця 2.3 – Дані про застосунок Authy 2FA

Алгоритми генерації одноразових паролів	TOTP
Сертифікація алгоритмів компанією OATH	Ні
Принцип роботи	1 - розпізнавання облич, сканер відбитку або введення пін, замість паролю 2 - 6-ти або 8-мизначний одноразовий пароль в якості другого фактора 3 - підтвердити в застосункув якості другого фактора
Безкоштовний	Так
Розробник	Authy
Open source	Ні
Операційні системи	iOS, Android, MacOS, Windows

Продовження таблиці 2.3

Шифровані резервні копії	Так
Версія для ПК	Так
Використання на декількох дивайсах	Так
Розмір застосунку	6.6 Мб
Оцінка на Google Play	4.2
Кількість завантажень на Google Play	1 000 000+
Вибір мови інтерфейсу	Так
Захист від скріншотві	Так
Налаштування часового інтервалу дійсності пароля	Ні

2.5 FreeOTP

FreeOTP - це застосунок для двофакторної аутентифікації в системах, що використовують OTP протоколи. Не потребує додаткової реєстрації, навідміну від застосунку Authy. Ініціалізація облікового запису проходить шляхом сканування QR-коду, або введення коду вручну.

FreeOTP реалізує відкриті стандарти HOTP і TOTP для генерації одноразових паролів. Існує можливість вказати інтервал дії одноразового пароля, наприклад, 30 секунд або дві години, а також кількість невдалих спроб введення пароля.

Для впровадження двофакторної автентифікації необхідно використати будь-який серверний компонент, який реалізує ці стандарти. Застосунок поширюється з відкритим вихідним кодом, та може бути використаний для будь-якого сервісу.

Загальні дані стосовно даного застосунку приведено у Таблиці 2.4.

Таблиця 2.4 – Дані про застосунок FreeOTP

Алгоритми генерації одноразових паролів	TOTP та HOTP
Сертифікація алгоритмів компанією OATH	Ні
Принцип роботи	6-ти або 8-мизначний одноразовий пароль в якості другого фактора
Безкоштовний	Так
Розробник	Red Hat
Open source	Так
Операційні системи	Android і iOS
Шифровані резервні копії	Ні

Продовження таблиці 2.4

Версія для ПК	Ні
Використання на декількох дивайсах	Ні
Розмір застосунку	418 Кб
Оцінка на Google Play	4.5
Кількість завантажень на Google Play	500 000+
Вибір мови інтерфейсу	Ні (лише англійська)
Захист від скріншотві	Так
Налаштування часового інтервалу дійсності пароля	Ні

Висновок до розділу 2

В Таблиці 2.5 приведені зведені результати аналізу застосунків автентифікації. Як ми бачимо частина розглянутих застосунків надає змогу використовувати не лише строгу автентифікацію, і пропонують в якості другого фактора використовувати фактор властивості (відбиток пальця, тощо). Усі сервіси мають можливість використання застосункі в якості генераторів одноразових паролів, що побудовані на основі таких криптографічних протоколів як HOTP та TOTP, що є необхідним для побудови строгої автентифікації. Жоден застосунк не був сертифікований

компанією OATH. Не зважаючи на це протоколи автентфікації HOTP та TOTP побудовані згідно стандартів компанії. На мою думку найдоцільнішим було б використання застосункі FreeOTP чи Google Authenticator оскільки в них наявний лише один метод двофакторної автентифікації, що забезпечує строгу автентифікацію користувача.

Назва застосунку	Google Authenticator	Microsoft Authenticator	Authy 2FA	FreeOTP
Алгоритми генерації одноразових паролів	TOTP та HOTP	TOTP	TOTP	TOTP та HOTP

Таблиця 2.5 – Загальна таблиця даних про застосунки

Сертифікація алгоритмів компанією OATH	Ні	Ні 1 - розпізнавання облич, сканер відбитку або введення пін, замість паролю 2 - 6-ти або 8-мизначний одноразовий пароль в якості другого фактора 3 - підтвердити в застосункув якості другого фактора	Ні 1 - розпізнавання облич, сканер відбитку або введення пін, замість паролю 2 - 6-ти або 8-мизначний одноразовий пароль в якості другого фактора 3 - підтвердити в застосункув якості другого фактора	Ні
Принцип роботи	6-ти або 8-мизначний одноразовий пароль в якості другого фактора			6-ти або 8-мизначний одноразовий пароль в якості другого фактора

Безкоштовний	Так	Так	Так	Так
Шифровані резервні копії в хмарі	Ні	Так	Так	Ні
Open source	до певної версії	Ні	Ні	Так
Розробник	Google	Microsoft	Authy	Red Hat
Застосунки під операційні системи	Android, BlackBerry, iOS, J2ME.	Ні	iOS, Android, MacOS, Windows	Ні
Версія для ПК	Ні	Ні	Так	Ні
		Android і iOS		Android і iOS

Продовження таблиці 2.5

Використання на декількох дивайсах	Ні	Ні	Так	Ні
Налаштування часового інтервалу	Ні	Ні	Ні	Так
Розмір застосунку	4 Мб	6.7 Мб	6.6 Мб	418 Кб
Вибір мови інтерфейсу	Так	Так	Так	Ні
Оцінка на Google Play	4,2	4,5	4,2	4,5
Кількість завантажень на Google Play	10 000 000+	10 000 000+	1 000 000+	500 000+
Захист від скріншотві	Ні	Ні	Так	Так

Продовження таблиці 2.5

Продовження таблиці 2.5

3 РОЗРОБКА ТА РЕАЛІЗАЦІЯ ВЕБ-ЗАСТОСУНКУ

3.1 Введення

У цьому розділі ми скористаємось отриманими з попереднього розділу даними. Сформулюємо критерії вибору сервісу двофакторної автентифікації. Згідно запропонованого методу оберемо сервіс двофакторної автентифікації та розробимо застосунок що використовує обраний сервіс двофакторної автентифікації.

3.2 Критерії вибору сервісу автентифікації

Найпершим та найважливішим критерієм є *надійність методу двофакторної автентифікації, що використовується сервісом*. Оскільки найбільш стійкою вважається строга двофакторна автентифікація, вибір падає на метод двофакторної автенифікації, що використовує за основу відкриті стандарти криптографічних протоколів автентифікації.

Другим критерієм є *наявність сертифікату відповідності стандарту*. Оскільки на даний момент введено сертифікацію лише для відкритих стандартів криптографічних протоколів від OATH, та сертифікація не є безкоштовною, даний критерій може бути використаним лише для одного з представлених у розглянутих сервісах методу автентифікації.

У випадку вибору сервісу, що має можливість використання різних методів автентифікації, що не є однаково стійкими, ми не можемо гарантувати однакову захищеність облікового запису всім користувачам. Обираючи той чи інший метод автентифікації, користувач натомість отримує різні рівні захисту своїх облікових записів. Тож наступним критерієм є *наявність можливості «нав'язати» користувачу обраний метод автентифікації*.

Зі сторони користувача для оцінки застосунку найбільш вірним є *кількість завантажень застосунку* на певній платформі, наприклад Google Play. Мінімальні та максимальні оцінки до критеріїв приведені в Таблиці 3.1.

Таблиця 3.1 – Мінімальні та максимальні оцінки до критеріїв

Надійність методу двофакторної автентифікації, що використовується сервісом.	6 - усі наявні методи підходять для строгої автентифікації; 3 - один чи декілька наявних методів підходять для строгої автентифікації; 0 - жоден з методів не підходить для строгої автентифікації;
Наявність сертифікату відповідності стандарту	2 - так; 0 - ні;
Можливості «нав’язати» користувачу обраний метод автентифікації	3 - так; 0 - ні;
Обрати конкретний метод автентифікації	3 - так; 0 - ні;
Кількість завантажень застосунку	3 - 10 000 000+; 2 - 1 000 000+; 1 - 500 000+; 0 - <500 000+

За даним методом оберемо кращий застосунок автентифікації. В Таблиці 3.2 приведені оцінки всіх застосунків.

Таблиця 3.2 – Оцінки розглянутих застосунків

Назва застосунку	Google Authenticator	Microsoft Authentificator	Authy (2FA)	FreeOTP
Надійність методу двофакторної автентифікації, що використовується сервісом.	6	3	3	6
Наявність сертифікату відповідності стандарту	0	0	0	0
Можливості «нав’язати» користувачу обраний метод автентифікації	3	0	0	3
Обрати конкретний метод автентифікації	3	0	0	3
Кількість завантажень застосунку	3	3	2	1
Загальна оцінка	15	6	5	13

Як бачимо за даними критеріями найкращим застосунком двофакторної автентифікації є Google Authenticator. Тож для впровадження ми обираємо даний застосунок.

3.3 Опис веб-застосунку

Використані технології: php, html, бази даних MySQL.

В якості серверного компоненту, що реалізує стандарт TOTP, було взято php клас з відкритого проекту на GitHub, Google Authenticator. [14-15]

Структура проекту:

- config.php – конфігурація бази даних. Підключається у всіх файлах, що взаємодіють з базою даних. Додаток Б.
- index.php – форми реєстрації та входу. Обробка даних після відправлення. У випадку коректно введених даних перенаправляє на сторінку device.php. Головна сторінка проекту. Додаток В.
- device.php – QR-код з секретним ключем, випадку виконання входу не відображається, поле вводу одноразового пароля. Виконується первинна перевірка корректності введеного одноразового пароля. У випадку коректно введених даних перенаправляє на сторінку home.php. Додаток Г.
- home.php – виконується перевірка відповідності введеного одноразового пароля, та згенерованого за допомогою взятого компоненту. Відображається результат входу: «Success» - у випадку коректного пароля, «Failed» - у випадку введення не коректного одноразового пароля. Додаток Д.

- logout.php – файл що «вбиває» сесію та очищує дані входу. Перенаправляє на сторінку index.php. Додаток Е.
- userClass.php – клас, що містить функції реєстрації, входу та функцію отримання даних користувача. Напряму працює з базою даних. Додаток Є.

Форма входу має два поля вводу: поле для вводу імені користувача, поле для вводу пароля користува. Форма реєстрації має чотири поля для вводу: поле для вводу імені, електронної пошти, імені користувача та пароля. Саме і данні певним чином хешуються та використовуються згодом як секретний ключ. Форми реєстрації та входу виглядають наступним чином (Рисунок 3.1).

Дані введені при реєстрації зберігаються вподальшому в таблиці бази даних, структура якої приведена в Додатку А.

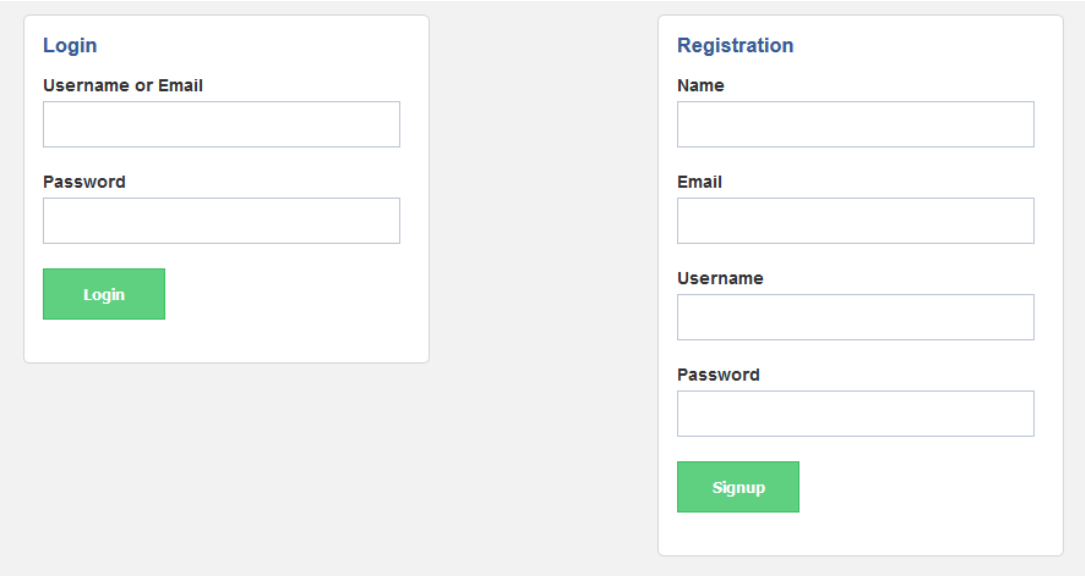

Рисунок 3.1 – Зовнішній вигляд форм реєстрації та входу

3.4 Робота веб-застосунку

Реєстрація:

- Необходно зареєструватися, для цього вводимо корректну інформацію у відповідну форму (Рисунок 3.2)
- За допомоггою застосунку на смартфоні зісканувати QR-код що з'явиться на екрані (Рисунок 3.3) та ввести одноразовий пароль що з'явився в застосунку (Рисунок 3.4).
- Потрапляємо на сторінку з результатами входу (Рисунок 3.6)
- У випадку введення некоректного одноразового пароля ми побачимо повідомлення відповідне до Рисунку 3.7

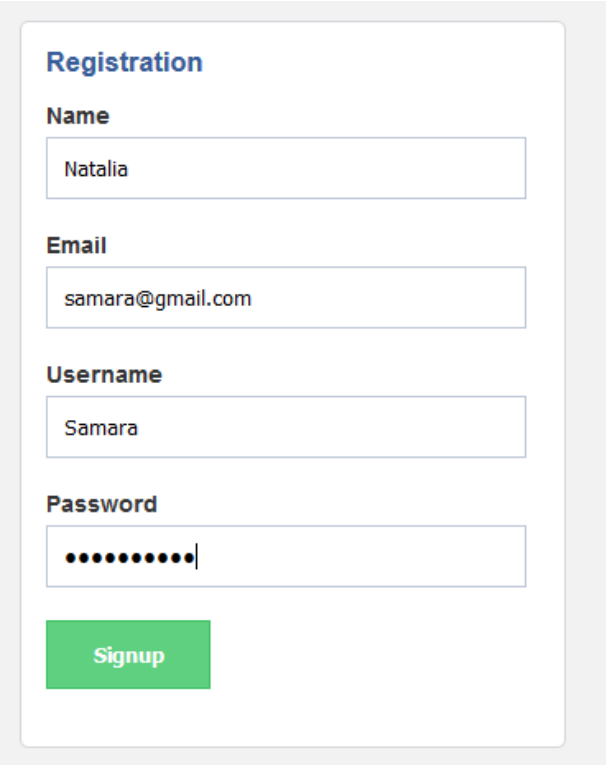

Рисунок 3.2 – Введення даних в форму реєстрації

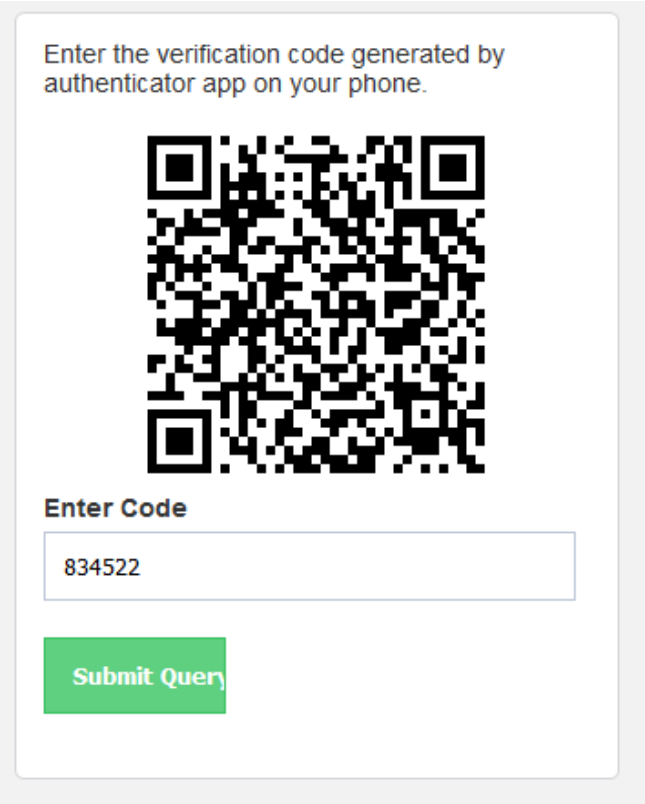

Рисунок 3.3 – QR-код для сканування, поле для введення одноразового пароля

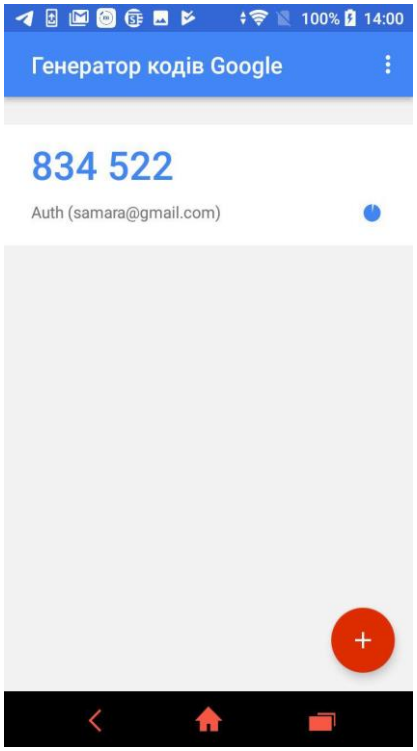

Рисунок 3.4 – Одноразовий пароль в застосунку Google Authenticator

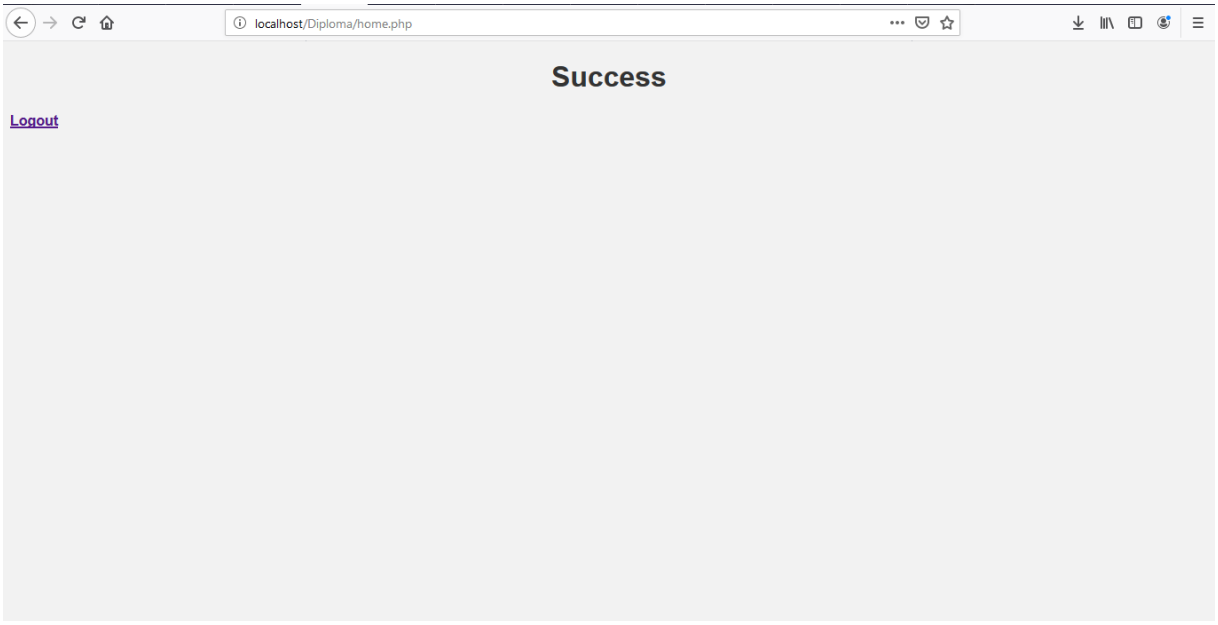

Рисунок 3.5 – Позитивний результат авторизації

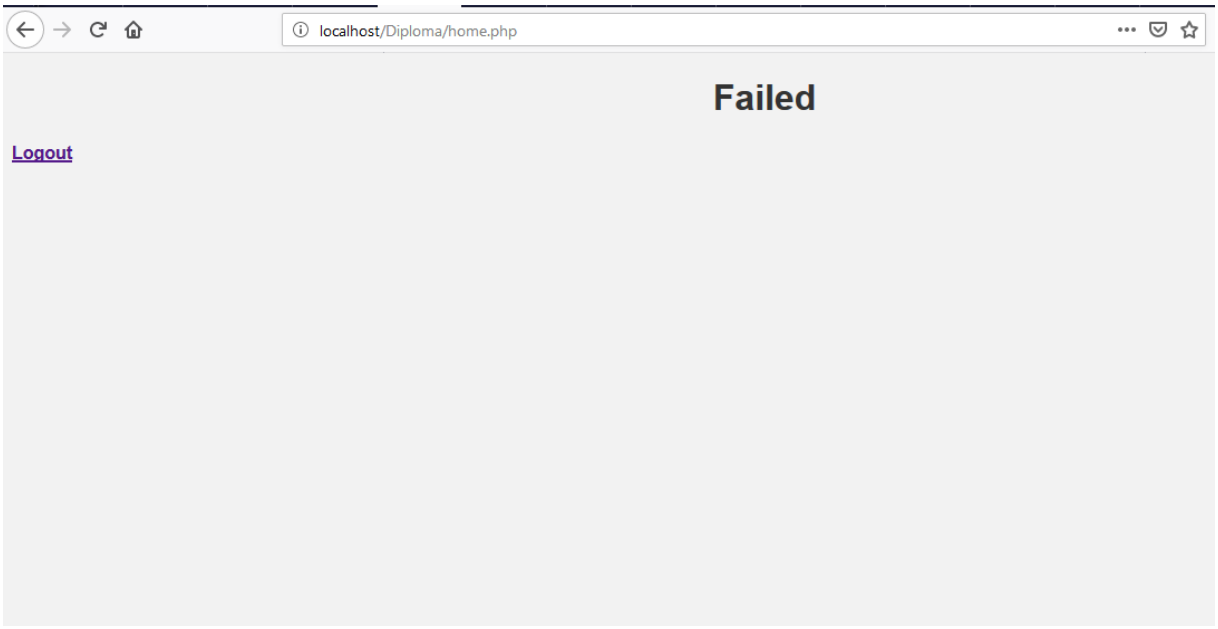

Рисунок 3.6 – Негативний результат авторизації

Вхід виконується іднетично до реєстрації, з різницею в заповнення форми входу (Рисунок 3.7) та вводу одноразового паролю без сканування QR-коду (Рисунок 3.8).

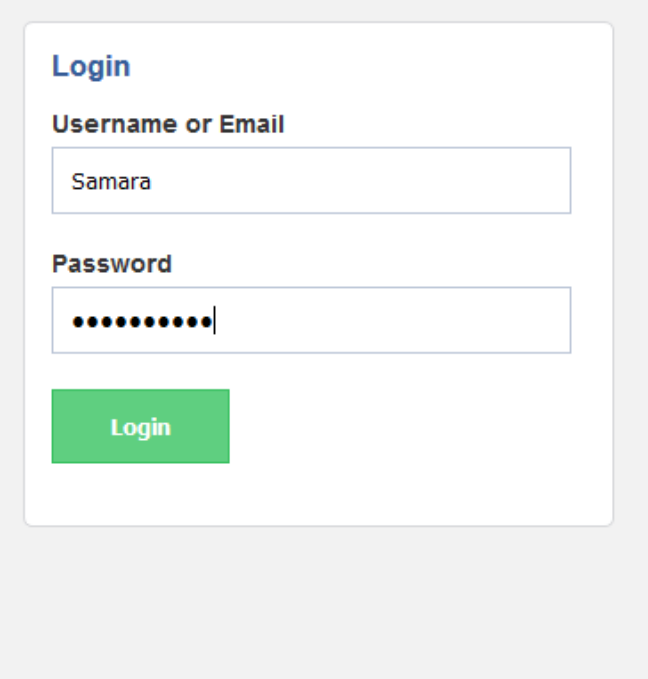

Рисунок 3.7 – Введення даних в форму входу

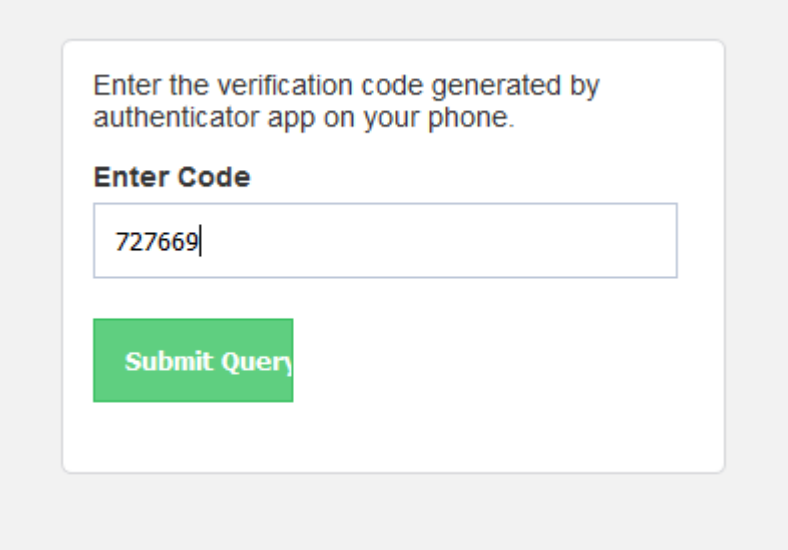

Рисунок 3.8 – Введення одноразового пароля у випадку входу

Висновки до розділу 3

В даному розділі було приведено та апробовано метод вибору застосунку двофакторної автентифікації. Було обрано та обгрунтовано вибір критеріїв оцінки застосунків двофакторної автентифікації, що були розглянуті в попередньому розділі. В результаті для впровадження було обрано застосунок Google Authenticator, що отримав вищий бал серед розглянутих застосунків. Було розроблено та перевірено веб-застосунок, що виконує двофакторну автентифікацію з Google Authenticator. Метод автентифікації, що використовує даний застосунок розроблений згідно відкритого стандарту від компанії OATH криптографічного протоколу автентифікації TOTP.

ВИСНОВКИ

В даній роботі були розглянуті методи двофакторної автентифікації описані у відкритих стандартах автентифікації від компаній FIDO та OATH.

Було проведено дослідження застосунків, що забезпечують виконання двофакторної автентифікації. Проаналізовано запропоновані методи автентифікації. На основі отриманих даних запропоновано ряд критеріїв вибору кращого застосунку двофакторної автентифікації.

Відповідно до запропонованих критеріїв було обрано кращим застосунок Google Authenticator, що забезпечує використання стійкого методу двофакторної автентифікації, розробленого згідно відкритих стандартів. Застосунок Google Authenticator є генератором одноразових паролів, розроблений згідно відкритих стандартів компанії OATH, протоколів автентифікації TOTP та HOTP.

Окрім того, було розроблено власний веб-застосунок, що використовує одноразові паролі за протоколом автентифікації TOTP та HOTP в якості другого фактора автентифікації. Даний веб-застосунок розроблено задля демонстрації використання обраного у третьому розділі застосунку двофакторної автентифікації від Google.

В ході виконання дипломної роботи було запропоновано метод вибору системи двофакторної автентифікації на основі смартфону. Обрані та аргументовані критерії вибору застосунку двофакторної автентифікації та методу автентифікації, що ним використовується. Розроблений веб-застосунок, що ілюструє впровадження та використання системи двофакторної автентифікації, обраної запропонованим методом.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Термін «Автентифікація» [Електронний ресурс] // КАБІНЕТ МІНІСТРІВ УКРАЇНИ. – 1997. – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/40-97-%D0%BF/ed20060315/find?text=%C0%E2%F2%E5%ED%F2%E8%F4%B3%EA%E0%F6%B3%FF.
2. Березніков А. Cloud services [Електронний ресурс] / Андрій Березніков // denovo. – 2018. – Режим доступу до ресурсу: https://www.de-novo.biz/blog/vidi-hmarnih-servisiv-yakij-obrati-ta-oglyad-hmarnih-provajd-8.
3. Specifications Overview [Електронний ресурс] // FIDO Alliance. – 2018. – Режим доступу до ресурсу: https://fidoalliance.org/specifications/.
4. OATH Certification [Електронний ресурс] // OATH Authentication. – 2019. – Режим доступу до ресурсу: https://openauthentication.org/oath-certification/.
5. HOTP: An HMAC-Based One-Time Password Algorithm [Електронний ресурс] // Network Working Group. – 2005. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc4226.
6. TOTP: Time-Based One-Time Password Algorithm [Електронний ресурс] // Internet Engineering Task Force. – 2008. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc6238.
7. OCRA: OATH Challenge-Response Algorithm [Електронний ресурс] // Internet Engineering Task Force. – 2011. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc6287.
8. The State of Strong Authentication [Електронний ресурс] // Javelin Strategy & Research. – 2019. – Режим доступу до ресурсу:

https://1nmqmp2u9dgf3jo9centu6rq-wpengine.netdna-ssl.com/wp-content/uploads/2019/01/The-State-of-Strong-Authentication-2019-Report.pdf.

9. Getting Started [Електронний ресурс] – Режим доступу до ресурсу: https://developers.google.com/api-client-library/javascript/start/start-js.
10. Authentication using the Google APIs Client [Електронний ресурс] – Режим доступу до ресурсу: https://developers.google.com/api-client-library/javascript/start/start-js.
11. What is the Microsoft Authenticator app? [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/azure/active-directory/user-help/user-help-auth-app-overview.
12. Features [Електронний ресурс] // authy. – 2019. – Режим доступу до ресурсу: https://authy.com/features/.
13. TouchID, PIN, Password, Encryption [Електронний ресурс] // Authy. – 2019. – Режим доступу до ресурсу: https://authy.com/features/secure/.
14. FreeOTP [Електронний ресурс] – Режим доступу до ресурсу: https://freeotp.github.io/.
15. Getting Started [Електронний ресурс] – Режим доступу до ресурсу: https://developers.google.com/api-client-library/javascript/start/start-js.
16. Open source version of Google Authenticator [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/google/google-authenticator.

ДОДАТКИ

Додаток А

Створення таблиці бази даних :

```
CREATE TABLE IF NOT EXISTS `users` (
  `uid` int(11) NOT NULL primary key AUTO_INCREMENT,
  `username` varchar(45) DEFAULT NULL,
  `email` varchar(120) DEFAULT NULL,
  `password` varchar(200) DEFAULT NULL,
  `name` varchar(100) DEFAULT NULL,
  `auth_code` varchar(16) DEFAULT NULL,
)
```

Додаток Б

config.php

```
<?php
session_start();
define('DB_SERVER', 'localhost');
define('DB_USERNAME', 'root');
define('DB_PASSWORD', '1234');
```

```
define('DB_DATABASE', 'local');


define("BASE_URL", "http://localhost/Diploma/");


function getDB()
{
    $dbhost = DB_SERVER;
    $dbuser = DB_USERNAME;
    $dbpass = DB_PASSWORD;
    $dbname = DB_DATABASE;
    try {
        $dbConnection = new PDO("mysql:host=$dbhost;dbname=$dbname",
$dbuser, $dbpass);
        $dbConnection->exec("set names utf8");
        $dbConnection->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
        return $dbConnection;
    } catch (PDOException $e) {
        echo 'Connection failed: ' . $e->getMessage();
    }
}
```

Додаток В

index.php

```
<?php
include("config.php");
if (!empty($_SESSION['uid'])) {
    header("Location: device.php");
}
include('class/userClass.php');
$user = new userClass();

require_once 'googleLib/GoogleAuthenticator.php';
$auth = new GoogleAuthenticator();
$secret = $ auth ->createSecret();

$errorMsgReg = '';
$errorMsgLogin = '';
if (!empty($_POST['login'])) {
    $usernameEmail = $_POST['usernameEmail'];
    $password = $_POST['password'];
    if (strlen(trim($usernameEmail)) > 1 && strlen(trim($password)) > 1) {
```

```
        $uid = $user->userLogin($usernameEmail, $password, $secret);
        if (!$uid) {
            $errorMsgLogin = "Please check details.";
        } else {
            $url = BASE_URL . 'device.php';
            header("Location: $url");
        }
    }
}

if (!empty($_POST['signup'])) {

    $username = $_POST['username'];
    $email = $_POST['email'];
    $password = $_POST['password'];
    $name = $_POST['name'];
    $username_check = preg_match('~^[A-Za-z0-9_]{3,20}$~i', $username);
    $email_check = preg_match('~^[a-zA-Z0-9._-]+@[a-zA-Z0-9._-]+\.([a-zA-
Z]{2,4})$~i', $email);
    $password_check = preg_match('~^[A-Za-z0-9!@#$%^&*()_]{6,20}$~i',
$password);
```

```
    if (!$username_check || !$email_check || !$password_check ||
strlen(trim($name)) <= 0) {
        $errorMsgReg = "Enter valid details.";
    } else {
        $uid = $user->userRegistration($username, $password, $email, $name,
$secret);

        if (!$uid) {
            $errorMsgReg = "Username or Email already exits.";
        } else {
            $url = BASE_URL . 'device.php';
            header("Location: $url");
        }

    }
}

?>
<html>
<head>
```

```
    <title>2-Step Verification</title>
    <link rel="stylesheet" type="text/css" href="style.css" charset="utf-8"/>
</head>
<body>
<div id="container">
    <h1>2-Step Verification using Google Authenticator</h1>

    <div id="login">
        <h3>Login</h3>
        <form method="post" action="" name="login">
            <label>Username or Email</label>
            <input type="text" name="usernameEmail"/>
            <label>Password</label>
            <input type="password" name="password"/>
            <div class="errorMsg"><?php echo $errorMsgLogin; ?></div>
            <input type="submit" class="button" name="login" value="Login">
        </form>
    </div>

    <div id="signup">
        <h3>Registration</h3>
```

```
        <form method="post" action="" name="registration">
            <label>Name</label>
            <input type="text" name="name"/>
            <label>Email</label>
            <input type="text" name="email"/>
            <label>Username</label>
            <input type="text" name="username"/>
            <label>Password</label>
            <input type="password" name="password"/>
            <div class="errorMsg"><?php echo $errorMsgReg; ?></div>
            <input type="submit" class="button" name="signup" value="Signup">
        </form>
    </div>

</div>


</body>
</html>
```

Додаток Г

device.php

```
<?php
include('config.php');
if (empty($_SESSION['uid'])) {
    header("Location: index.php");
}

require_once 'googleLib/GoogleAuthenticator.php';
include('class/userClass.php');

$user = new userClass();
$userData = $user->userData($_SESSION['uid']);
$secret = $userData->auth_code;
$email = $userData->email;

$ auth = new GoogleAuthenticator();
$qrCodeUrl = $ auth ->getQRCodeGoogleUrl($email, $secret, 'Auth');

?>
<html>
<head>
    <title>2-Step Verification</title>
```

```
    <link rel="stylesheet" type="text/css" href="style.css" charset="utf-8"/>
</head>
<body>
<div id="container">
    <h1>2-Step Verification</h1>
    <div id='device'>

        <p>Enter the verification code generated by authenticator app on your phone.</p>
        <div id="img">
            <img src='<?php echo $qrCodeUrl; ?>'/>
        </div>

        <form method="post" action="home.php">
            <label>Enter Code</label>
            <input type="text" name="code"/>
            <input type="submit" class="button"/>
        </form>
    </div>
</div>
</body>
```

```
</html>
```

Додаток Д

home.php

```
<?php
include('config.php');


include('class/userClass.php');
$user = new userClass();
$userData = $user->userData($_SESSION['uid']);


if ($_POST['code']) {
    $code = $_POST['code'];
    $secret = $userData->auth_code;
    require_once 'googleLib/GoogleAuthenticator.php';
    $auth = new GoogleAuthenticator();
    $check = $auth ->verifyCode($secret, $code, 2);


    if (!$check) {
        ?>
        <html>
        <head>
```

```
        <title>Failed</title>
        <link rel="stylesheet" type="text/css" href="style.css" charset="utf-8"/>
    </head>
    <body>
    <h1>Failed</h1>
    <h4><a href="<?php echo BASE_URL; ?>logout.php">Logout</a></h4>
    </body>
    </html>
    <?php
} else {
    $userData = $user->userData($_SESSION['uid']);
    ?>
    <html>
<head>
    <title>Success</title>
    <link rel="stylesheet" type="text/css" href="style.css" charset="utf-8"/>
</head>
<body>
<h1>Success</h1>
<h4><a href="<?php echo BASE_URL; ?>logout.php">Logout</a></h4>
</body>
```

```
        <?
    }
}
```

Додаток Е

logout.php

```
<?php
include('config.php');
session_destroy();
if (empty($_SESSION['uid'])) {
    $url = BASE_URL . 'index.php';
    header("Location: $url");
}
```

Додаток Є

userClass.php

```
<?php
class userClass
{
    /**
     * @param $usernameEmail
```

```
     * @param $password
     * @param $google_auth_code
     * @return bool
     */
    public function userLogin($usernameEmail, $password, $google_auth_code)
    {

        $db = getDB();
        $hash_password = hash('sha256', $password);
        $stmt = $db->prepare("SELECT uid FROM users WHERE
(username=:usernameEmail or email=:usernameEmail) AND
password=:hash_password");
        $stmt->bindParam("usernameEmail", $usernameEmail,
PDO::PARAM_STR);
        $stmt->bindParam("hash_password", $hash_password,
PDO::PARAM_STR);
        $stmt->execute();
        $count = $stmt->rowCount();
        $data = $stmt->fetch(PDO::FETCH_OBJ);
        $db = null;
        if (!$count) {
            return false;
```

```
        } else {
            $_SESSION['uid'] = $data->uid;
            $_SESSION['google_auth_code'] = $google_auth_code;
            return true;
        }
    }

    /**
     * @param $username
     * @param $password
     * @param $email
     * @param $name
     * @param $secret
     * @return string
     */
    public function userRegistration($username, $password, $email, $name,
$secret)
    {
        try {
            $db = getDB();
```

```
            $stmt = $db->prepare("INSERT INTO
users(username,password,email,name,google_auth_code) VALUES
(:username,:hash_password,:email,:name,:google_auth_code)");
            $stmt->bindParam("username", $username);
            $hash = hash('sha256', $password);
            $stmt->bindParam("hash_password", $hash);
            $stmt->bindParam("email", $email);
            $stmt->bindParam("name", $name);
            $stmt->bindParam("google_auth_code", $secret);
            $stmt->execute();
            $uid = $db->lastInsertId();
            $db = null;
            $_SESSION['uid'] = $uid;
            return $uid;

        } catch (PDOException $e) {
            echo '{"error":{"text":' . $e->getMessage() . '}}';
        }
    }

    /**
```

```
     * @param $uid
     * @return mixed
     */
    public function userData($uid)
    {
        try {
            $db = getDB();
            $stmt = $db->prepare("SELECT
email,username,name,google_auth_code FROM users WHERE uid=:uid");
            $stmt->bindParam("uid", $uid, PDO::PARAM_INT);
            $stmt->execute();
            $data = $stmt->fetch(PDO::FETCH_OBJ);
            return $data;
        } catch (PDOException $e) {
            echo '{"error":{"text":' . $e->getMessage() . '}}';
        }
    }
}
```