МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютерна інженерія»

Освітньо-професійна програма: «Безпека комп'ютерних систем і мереж»

Група: 4КБ-02

Дипломний проект

здобувача освіти денної форми навчання
КБ.02.08.000.ДП

КОВАЛЬОВА
АНДРІЯ ОЛЕКСІЙОВИЧА

м. Одеса
2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: **123 «Комп'ютерна інженерія»**

Освітньо-професійна програма: «**Безпека комп'ютерних систем і мереж»**

Група: **4КБ-02**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту на тему:

Розробка системи контролю надійності паролів для корпоративного сайту

Проектний матеріал складається з пояснювальної записки на 72 сторінках та графічного (презентаційного) матеріалу на 14 аркушах (слайдах)

Дипломник _____ (Ковальов А.О.)

Керівник _____ (Стайкуца С.В.)

Консультанти:

з економічного розділу _____ (Канський М.Ю.)

з розділу охорони праці та техніки безпеки _____ (Чорновол Н.І.)

з нормоконтролю _____ (Петрашова В.І.)

старший консультант _____ (Кривченко Ю.В.)

До захисту допущений

Голова циклової комісії _____ (Кривченко Ю.В.)

Завідувач відділення _____ (Краснокутська К.Г.)

Захист «27» червня 2025 р. Протокол ЕК № 6

Оцінка ЕК 4 (добре) / 75,

Секретар ЕК _____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Відділення *комп'ютерних систем* Комісія *КТ та ПІ*

Спеціальність *123 «Комп'ютерна інженерія»*

Освітньо-професійна програма *«Безпека комп'ютерних систем і мереж»*

ЗАТВЕРДЖУЮ:
Заст. дир. з НВР *Беркань І.В.*
"19" 05 2025 р.

ЗАВДАННЯ

на дипломний проект

Здобувачеві освіти *Ковальова Андрія Олексійовича*
(прізвище, ім'я, по батькові)

1. Тема проекту *Розробка системи контролю надійності паролів для корпоративного сайту*

затверджена наказом по коледжу від "*14*" *листопада* 2024р. № *246*

2. Термін здачі закінченого проекту *16 червня, 2025 р.*

3. Вихідні данні до проекту *Розробка клієнтського модулю оцінки надійності паролів з урахуванням довжини, складу, шаблонів та вразливостей. Реалізування перевірки паролів на компрометацію через відкриті бази. Додавання генератору безпечних паролів. Використання React 18, TypeScript, Tailwind CSS та Framer Motion для створення UI. Здійснення повної оброби у паролів лише на клієнтській стороні. Побудування легкого серверу на Node.js + Express. Забезпечити розгортання через Vercel, розробку - у Replit.*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити)
Основи використання паролів у системах безпеки; Аналіз методів зламу паролів і їх класифікація; Методи оцінки надійності паролів та індикатори складності; Вимоги до паролів згідно з сучасними стандартами; Формування технічного завдання на створення системи; Обґрунтування вибору технологій для реалізації; Розробка клієнтської частини з візуальним інтерфейсом; Створення алгоритмів перевірки та генерації паролів; Перевірка паролів на компрометацію через зовнішні джерела; Розгортання та тестування системи, аналіз результатів

5. Перелік графічного (презентаційного) матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів)
Титул; Мета і завдання проєкту; Актуальність теми; Аналіз існуючих рішень; Архітектура рішення; Технології розробки; Функціонал сайту; Ключові елементи UI; Алгоритм оцінки пароля; Візуалізація результату перевірки; Безпека і обмеження; Тестування та результати; Хостинг та публікація; Висновки; Питання / Дякую за увагу

6. Консультанти по проекту, із зазначенням розділів проекту, що їх стосується

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Основний розділ	*Стайкуца С.В*		
Економічний розділ	*Канський М.Ю.*		
Розділ охорони праці	*Чорновол Н.І.*		
Нормоконтроль	*Петрашова В.І.*		
Старший консультант	*Кривченко Ю.В.*		

7. Дата видачі завдання 15.05.2025

Керівник *Стайкуца С.В.* _____
(підпис)

Завдання прийняв до виконання *Ковальов А.О.* _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/р	Назва етапів дипломного проекту	Термін виконання етапів дипломного проекту (роботи)	Відмітка про виконання
1	*Вступ. Формулювання мети та постановка задачі*	14.05.2025	Виконано
2	*Аналіз літературних джерел та нормативних*	15.05.2025	Виконано
	документів		Виконано
3	*Аналіз існуючих рішень і конкурентних сервісів*	17.05.2025	Виконано
4	*Вибір інструментів розробки*	18.05.2025	Виконано
5	*Проєктування архітектури системи*	19.05.2025	Виконано
6	*Розробка алгоритму перевірки надійності пароля*	22.06.2025	Виконано
7	*Створення генератора паролів*	28.05.2025	Виконано
8	*Розробка інтерфейсу користувача (UI/UX)*	1.06.2025	Виконано
9	*Інтеграція системи та деплой на платформу Vercel*	4.06.2025	Виконано
10	*Підготовка пояснювальної записки та технічної*	9.06.2025	Виконано
	документації		Виконано
11	*Випробування застосунку та аналіз результатів*	10.08.2025	Виконано
12	*Виконання економічних розрахунків*	11.05.2025	Виконано
13	*Розробка питань з охорони праці та техніки безпеки*	14.05.2025	Виконано
14	*Підготовка мультимедійної презентації проекту*	15.06.2025	Виконано

Дипломник _____
(підпис)

Керівник _____
(підпис)

Формат	Зона	Поз.	Позначення	Назва	Кіл.	Примітка
				Документація		
			КБ 02. 08 000. 00 ДП	Дипломний проект		
А4			КБ 02. 08 000. 00 ДП ПЗ	Пояснювальна записка	1	

Зм.	Арк.	№ докум.	Підпис	Дата	КБ 02. 08 000. 00 ДП		
Розробив		Ковальов А.О.		16.06	Розробка системи контролю надійності паролів для корпоративного сайту	Літ.	Аркуш 4 / Аркушів 72
Перевірив		Стайкуца С.В.		16.06		Н Д П	
Н. контр.		Петрашова В.І.		16.06.16		ВСП "ОТФК ОНТУ" гр.4КБ-02	
Затверд.		Кривченко Ю.В.		18.06			

ЗМІСТ

ВСТУП

У сучасному цифровому середовищі питання безпеки облікових даних набуває критичного значення. З розвитком інформаційних технологій дедалі більше уваги приділяється захисту персональної інформації користувачів, особливо в корпоративному секторі. Одним із найбільш вразливих елементів систем автентифікації залишаються паролі — найбільш розповсюджений, проте далеко не найнадійніший механізм захисту.

На жаль, через низький рівень обізнаності, зручність або звичку, багато користувачів створюють прості та легко передбачувані паролі, що значно підвищує ризик несанкціонованого доступу. Відповідно, виникає необхідність у наявності механізмів, які дозволяють не лише встановлювати вимоги до складності паролів, але й оцінювати їхню надійність на етапі створення.

У даній дипломній роботі розглядається комплексна проблема слабких паролів, методи їх злому, а також засоби захисту, що застосовуються сучасними системами безпеки. Значна увага приділяється методам оцінки надійності пароля, аналізу підходів, що використовуються різними вендорами, та формуванню власного алгоритму оцінки.

Основною метою цієї роботи є створення веб-додатку, що дозволяє користувачеві оцінити надійність пароля в режимі реального часу без передачі чи зберігання конфіденційних даних.

У першому розділі роботи надано теоретичний огляд поняття пароля, класифікацію методів його злому, а також розглянуто основні показники та метрики надійності. Також описано вимоги до паролів, що встановлюються постачальниками IT-рішень.

Практична частина присвячена безпосередньо розробці клієнтської системи перевірки надійності паролів для корпоративного веб-сайту. Наведено технічне завдання, обґрунтовано вибір засобів реалізації та представлено результати тестування розробленого рішення.

Результатом дипломного проєкту стала функціональна, зручна та безпечна система контролю надійності паролів, що відповідає сучасним вимогам до веб-безпеки.

1 ОСНОВНИЙ РОЗДІЛ

1.1 Фундаментальні відомості щодо використання паролів

Онлайн - безпека була головною проблемою з того часу, як Інтернет став необхідністю для суспільства - від бізнес-діяльності до повсякденного життя звичайних людей. Фундаментальним аспектом онлайн-безпеки є захист даних від несанкціонованого доступу. Найпоширенішим методом для цього є використання пароля як частини процесу онлайн-доступу. Пароль - це секретний рядок символів, який відомий лише користувачу, і його хешований код зберігається на сервері, що надає доступ до даних.

Коли користувач запитує доступ до даних, він вводить пароль разом з іншою ідентифікаційною інформацією, такою як ім’я користувача або електронна пошта. Хеш (message digest cipher) пароля обчислюється, і хеш-код передається на сервер, де він порівнюється з тим, що збережено.

Хеш-функції, як-от SHA-256, bcrypt або Argon2, забезпечують одностороннє шифрування - тобто навіть маючи хеш, відновити оригінальний пароль майже неможливо. Усі ці алгоритми призначені для ускладнення процесу підбору пароля шляхом перебору. Сучасні системи безпеки додатково використовують "сіль" (salt) - випадкові символи, що додаються до пароля перед хешуванням, - щоб захиститися від атак з попередньо обчисленими таблицями (rainbow tables).

Попри криптографічний захист, слабкий пароль залишається слабким місцем у системі. Простий пароль типу "123456" або "qwerty" буде підібрано за лічені секунди. Саме тому системи автентифікації дедалі частіше накладають жорсткі вимоги до складності пароля: мінімальна довжина, наявність великих і малих літер, цифр, спеціальних символів тощо.

У сучасних корпоративних системах автентифікації застосовуються додаткові шари захисту - наприклад, обмеження на кількість невдалих спроб входу, двофакторна автентифікація (2FA) або перевірка на використання пароля з відомих витоків. Останній підхід базується на зіставленні введеного пароля з базами зламаних облікових даних.

Веб-додатки нового покоління також пропонують динамічну оцінку надійності пароля в режимі реального часу - так звані password strength meters. Вони аналізують пароль ще до його надсилання на сервер і пропонують користувачу поради щодо посилення комбінації. Такі системи можуть враховувати не лише довжину й символи, але й контекст, наприклад, схожість з логіном, повторюваність, популярність шаблону.

З точки зору архітектури безпеки, пароль є лише одним із компонентів доступу, і його надійність безпосередньо залежить від трьох чинників: криптографічного захисту, надійності з боку користувача та логіки роботи самої системи. У комплексі ці складові визначають рівень ризику несанкціонованого доступу.

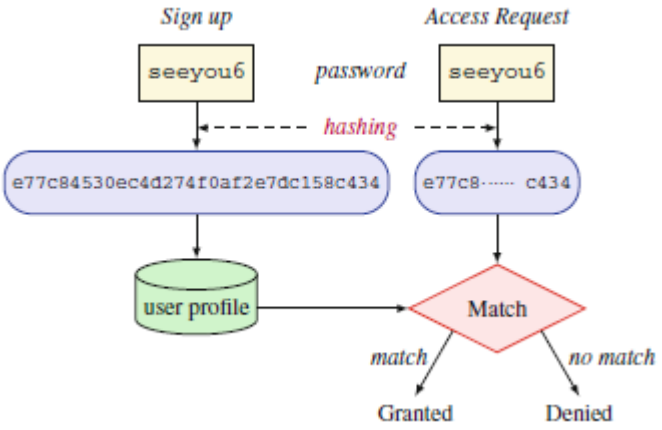


Рисунок 1.1. Механізм введення паролів для ресурсу
запрос на доступ

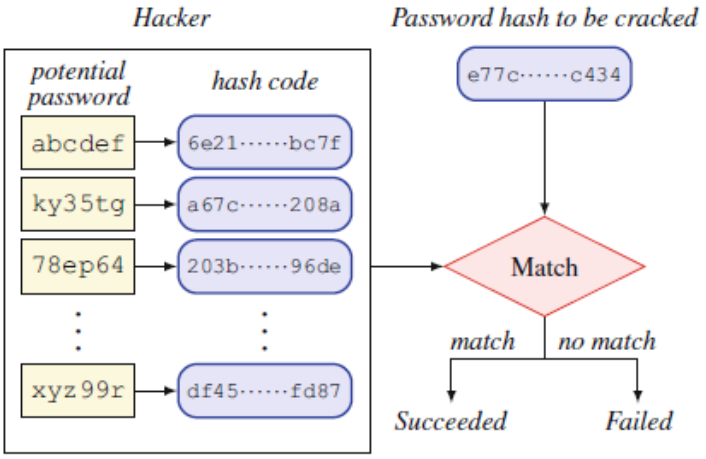


Рисунок 1.2. Взлом хеш-кода пароля
код пароля

Код, який порівнюється, вже збережений у базі даних сервера. Якщо знайдено збіг - вважається, що користувач дійсно той, за кого себе видає, і доступ надається; інакше - доступ забороняється.

Формально, хеш-функція *f* відображає пароль (рядок символів) *p* у хеш-код (шифр) *h*:

$$f(p) = h \qquad (1.1)$$

е *f* - необоротна функція, тобто обернена функція f^{-1} не існує.

Оскільки хеш-функція є необоротною, єдиний спосіб для сторонньої особи (наприклад, хакера) дізнатись пароль - це перебирати можливі рядки символів, щоб перевірити, чи збігається їхній хеш-код з хеш-кодом справжнього пароля. Якщо збігу немає - хакер пробує інший рядок і продовжує процес (це офлайн-атака) доти, доки не буде знайдено збіг (тобто пароль зламано), або поки хакер не здасться. Ідея зламу пароля зображена на рис. 2, де хакер пробує послідовність можливих паролів $p_1, p_2, \ldots, p_k$ для отримання хеш-кодів $h_1, h_2, \ldots, h_k$ і перевіряє, чи $h_k = h$, де *h* - це заданий хеш-код, який потрібно зламати. У разі онлайн-атаки правило «три спроби» може не дозволити хакеру здійснити більше трьох підходів.

1.2 Аналіз методів зламу паролів

Кожен пароль p хешується за допомогою функції шифрування f для генерації свого хеш-коду. h, як визначено в рівнянні (1). Оскільки в базі даних зберігається тільки h (а не p), ресурсу, завдання злому пароля полягає в тому, щоб використовувати метод C таким чином, щоб c(h) = p. Зауважте, що c не є f^{-1} , якої не існує. Тому хакеру потрібно з'ясувати, який метод c використовувати, щоб у нього було більше шансів на успішне знаходження p.

Найбільш часто використовувані методи злому паролів включають атаки грубої сили, атаки за словниками та деякі їх варіації, враховуючи компроміс між часом і простором.

Ці методи дозволяють зловмисникам систематично підбирати можливі комбінації або використовувати заздалегідь підготовлені списки.

1.2.1 Атаки типу Brute-Force

Нехай $p = a_1a_2\ldots a_l$, де $a_i \in \Sigma$, - це пароль довжини l, а Σ - алфавіт (множина допустимих символів), і $N = |\Sigma|$ - кількість символів у цьому алфавіті. Нехай задано

хеш-код h = f(p). Атака методом повного перебору (brute-force) полягає в тому, щоб перебирати кожен можливий рядок $s = x_1x_2\ldots x_l$, де $x_i \in \Sigma$, поки не буде знайдено такий, що f(s) = h.

Час t, необхідний для зламу пароля, пропорційний кількості можливих рядків:

$$t \in \Theta(N^l) \qquad (1.2)$$

де N^l - це кількість усіх можливих рядків довжини l, які можна скласти з алфавіту Σ. Тобто, часова складність атаки перебором є поліноміальною по N і експоненційною по l. Зазвичай алфавіт Σ - це один із базових наборів символів або їхня комбінація.

Більшість паролів використовують лише малі літери (L). У цьому випадку, для пароля довжиною 6 символів (мінімальна довжина, яку вимагає більшість сервісів), є $26^6 \approx 308.9 \times 10^6$ можливих комбінацій. Якщо функція хешування f може обробляти 100 000 запитів за секунду, то перебір усіх можливих комбінацій займе близько 3089 секунд, або 51.5 хвилини.

Якщо алфавіт більший, наприклад $D \cup L$ (цифри та малі літери), то перебір 6-символьного пароля займе понад 6 годин, 7-символьного - понад 9 днів, а 8-символьного - понад 11 місяців.

Практично неможливо зламати паролі довжиною 7 символів і більше методом перебору, якщо використовується розширений алфавіт Σ. Саме тому більшість онлайн-сервісів вимагають паролі не коротші за 8 символів, які містять хоча б одну велику літеру, цифру або спеціальний символ, окрім малих літер. Це дозволяє досягти N = (62 або 94) при $l \geq 8$.

1.2.2 Атаки за словником

Оскільки більшість людей використовують легкозапам’ятовувані паролі, які, ймовірно, є словами зі словників або варіаціями цих слів, хакер може перебирати кожне слово зі словника, а не випадкові рядки, як при повному переборі. При такому підході кожне слово $w_i \in D$ зі словника D перевіряється на відповідність умові $f(w_i) = h$, де h - це заданий хеш-код, який потрібно зламати. Отже, якщо пароль - це слово зі словника, його дуже легко зламати. На практиці всі хеш-коди

$f(w_i)$ заздалегідь обчислюються й зберігаються в базі даних, а не обчислюються під час виконання.

Цей підхід має дві проблеми. По-перше, словник має містити дуже велику кількість слів, щоб охопити більшість (якщо не всі) паролів, які, на вашу думку, можуть використовувати люди. По-друге, більшість користувачів знають про словникові атаки й уникають використання справжніх слів зі словника як паролів; натомість вони вносять невеликі зміни до слова, щоб його було легко запам'ятати. Наприклад, замість того, щоб безпосередньо використовувати *essay* як пароль, вони можуть використати *essay1*, *e55ay* або *3ssay*. Тому при словникових атаках часто застосовуються певні правила до «написання» слів, наприклад, заміна *l* на *1*, *0* на *0*, *e* на *3*, *s* на *5* тощо.

Нехай D - словник, що використовується, а m - кількість правил, тоді час t, необхідний для зламу пароля за допомогою словникової атаки, становить

$$t \in \mathcal{O}(|D| + m) \qquad (1.3)$$

Тобто, часова складність є лінійною відносно розміру словника та кількості правил, що є суттєвим покращенням порівняно з повним перебором. Однак цей метод може не знайти пароль, якщо словник не містить самого пароля або його варіацій після застосування правил.

1.2.3 Пошук за таблицею та атака за допомогою райдужних таблиць

Щоб пришвидшити час злому, атака з пошуком у таблиці зберігає попередньо обчислені хеші потенційних паролів (словників) у базі даних та зламує заданий хеш пароля, шукаючи в базі даних. Пошук набагато швидший, ніж оригінальна атака за словником, просто тому, що немає потреби обчислювати хеш для кожного вгадування під час виконання. Однак цей метод вимагає величезного обсягу місця для зберігання «всіх» можливих паролів та їх хешів.

Атака за райдужним столом [16] – це варіація атаки пошуку в таблиці. Замість попереднього обчислення хеш-кодів великої кількості потенційних паролів та їх зберігання в базі даних, підхід райдужної таблиці є компромісом між

часом і пам'яттю для зберігання набагато меншої кількості хеш-кодів, які все ще представляють величезну кількість паролів. Основна ідея полягає у створенні ланцюжка хеш-паролів довжиною *k* що охоплює *k* потенційні паролі та їхні хеш-коди.

Для кожного слова w ∈ D у словнику D ми створюємо ланцюжок c = (p1, h1,…, p_k, h_k), де p_1 = w, h_i = f (p_i) та p_i = r(h_{i-1}), f – це хеш-функція crypt, а r – функція reduce, яка «зменшує» код а до потенційного пароля. Для кожного ланцюжка в базі даних зберігається лише пара (p1, hk), тобто початковий пароль та кінцевий хеш-код. Отже, якщо k = 10 000, пара (p_1, h_n) представляє всі 10 000 паролів та їхні хеш-коди в ланцюжку, що забезпечує значну економію місця для зберігання. Райдужна таблиця T – це набір ланцюжків: T = {c_i, i = 1,…, n}, де ci – це ланцюжок для слова w_i в D.

Щоб зламати заданий хеш-код h пароля p, ми перевіряємо, чи h дорівнює кінцевому хешу h_k. Якщо так, то r(h_k) – це цільовий пароль p. В іншому випадку ми продовжуємо застосовувати r та f по черзі назад у ланцюжку, доки пароль не буде знайдено, або ми переміщуємо наступний ланцюжок у таблиці T. Якщо всі ланцюжки вичерпані, нам просто не вдалося знайти p.

Часова складність атаки на райдужну таблицю становить

$$t \in \Theta(k\,D) \qquad (1.4)$$

що лінійно залежить від розміру словника, але з постійним коефіцієнтом k, який може бути досить великим (скажімо, 10 000 або більше). Це компроміс між часом і пам'яттю, що, використовуючи той самий простір для зберігання, він охоплює в k разів більше слів, використовуючи той самий словник, але приблизно в k разів повільніше, ніж атака за словником.

Одна з проблем підходу з райдужною таблицею полягає в колізії, коли два різні хеш-коди зводяться до одного пароля. Ще одна складність полягає в тому, як зробити функцію редукції r «добре поведеною». Тобто, r повинна відображати хеш-коди в розподілений (ймовірно, як вибраний користувачем, а не випадковий) набір паролів.

1.2.4 Атака з використанням марковської моделі

Стверджувалося, що користувачі віддають перевагу паролям, які легко запам'ятати. Більшість користувачів знають про атаки за словником, тому паролі, які легко запам'ятовуються людиною, здебільшого не містяться в словниках і не є випадковими (випадково згенеровані паролі важко запам'ятати). Одним із підходів до атаки на паролі, які легко запам'ятовуються людиною, є атака «розумного словника» з використанням словників, що містять паролі, які користувачі, ймовірно, генерують. Нараянан і Шматіков запропонували метод швидкої атаки за словником, заснований на ймовірності послідовності символів у паролях користувачів.

Метод використовує стандартну марковську модель для створення розумного словника, який набагато менший за ті, що використовуються в традиційній словниковій атаці. Основне спостереження полягає в тому, що «розподіл літер у паролях, які легко запам'ятовуються, ймовірно, буде подібним до розподілу літер у рідній мові користувачів».

Отже, словник Маркова може бути створений на основі ймовірності символів у послідовності.

Нехай v(x) - частота символу *x* в англійському тексті, а $v(x_{i+1}|x_i)$ - частота символу x_{i+1}, враховуючи, що раніше згенерований символ - це x_i. У марковській моделі нульового порядку ймовірність послідовності $\alpha = x_1x_2\ldots x_n$ дорівнює

$$p(\alpha) = \prod_{x \in \alpha} v(x) \tag{1.5}$$

де розподіл кожного символу не залежить від попереднього символу.

У марковській моделі першого порядку

$$p(x_1x_2 \ldots x_n) = v(x_1) \prod_{i=1}^{n} v(x_{i+1}|x_i) \tag{1.6}$$

Марковські словники створюються відповідно на двох рівнях.

Словник нульового порядку визначається як

$$D_{v,\theta} = \{\alpha : \prod_{x \in \alpha} v(x) \geq \theta\} \quad (1.7)$$

де theta - це поріг. Словник першого порядку – це

$$D_{v,\theta} = \{x_1 x_2 \ldots x_n : v(x_1) \prod_{i=1}^{n} v(x_{i+1}|x_i) \geq \theta\} \quad (1.8)$$

Великою перевагою цих моделей є те, що вони значно зменшують розмір простору пошуку, виключаючи зі словника більшість слів, які навряд чи є паролями, обраними користувачем. Якщо =1/7 (тобто створюється лише 14% послідовностей, тоді як 86% послідовностей ігноруються), словник нульового порядку все ще має 90% ймовірність покриття правдоподібних паролів. Словник, що містить 1/11 ключового простору, має 80% покриття, а 1/40 ключового простору має 50% покриття. Їхні експерименти з використанням словників малої частини пошуку space успішно відновив 67,6% паролів, що значно перевищує показники багатьох попередніх робіт.

1.2.5 Атака за допомогою ймовірнісної контекстно-вільної граматики

Атака за словником часто використовує правила спотворення слів, але вибір ефективних правил спотворення може бути складним. Один із підходів до вирішення цієї проблеми полягає у генеруванні припущень у порядку їхньої ймовірності бути паролями користувачів. Це збільшить ймовірність злому цільового пароля за обмежену кількість припущень. Основна ідея цього підходу полягає в оцінці ймовірності паролів користувачів з навчального набору, набору розкритих реальних паролів та створенні контекстно-вільної граматики, яка буде використовуватися для оцінки ймовірності формування рядка [21, 23].

Ймовірнісна контекстно-вільна граматика визначається як $G = (V, \Sigma, T, P)$, де V – скінченна множина нетерміналів (змінних), Σ – скінченна множина терміналів, T ∈ V – початкова змінна, а P – набір продукційних правил вигляду

$$\alpha \rightarrow \beta, (p) \quad (1.9)$$

де $\alpha \in V$ – змінна, $\beta \in V \times \boldsymbol{\Sigma}$ – рядок символів (змінних та терміналів), а p – ймовірність, пов'язана з правилом продукування таким чином, що $\Sigma_i p_i = 1$ для всіх продукцій i, які мають однакове α. У випадку паролів єдиними змінними (окрім початкового символу T) є L, D та S, що представляють літери, цифри та спеціальні символи. Позначення L_k, D_k та S_k представляють послідовні k літер, послідовні k цифр, послідовні k спеціальних символів відповідно. Ймовірність p_i кожного правила продукування i оцінюється за допомогою навчального набору. Ймовірність реченнявої форми (рядка, отриманого з *T*) – це добуток ймовірностей продукцій, що використовуються при виведенні. Прикладом виведення є

$$S \Rightarrow L_3D_1S_1 \Rightarrow L_34S_1 \Rightarrow L_34\# \qquad (1.10)$$

в яких використовуються правила продукування D1→ 4 та S1→ #

Претермінальні структури (речення-форми) впорядковані за зменшенням ймовірності, а для вгадування можна заповнити словникові слова та хеші.

1.3 Заходи щодо якості паролів

Аналіз надійності паролів тривалий час був активною областю досліджень та практики. Основна увага у цій роботі приділяється метрикам надійності паролів та оцінці цих метрик. Ми розглянемо кілька метрик якості паролів, включаючи складність, яку ми пропонуємо тут.

1.3.1 Індикатор якості пароля

Більшість користувачів знають про словникові атаки та уникають використання словникових слів для паролів. Однак користувачі хочуть, щоб паролі було легко запам'ятовувати, тому вони схильні використовувати одне слово та вносити до нього невеликі зміни. З огляду на це, методи словникової атаки також використовують різні правила спотворення слів, щоб зіставити пароль зі словами у словниках. Отже, надійність пароля враховує не лише те, чи є пароль у словниках, але й те, наскільки легко (чи важко) виправити «орфографічні помилки» в паролі, щоб він відповідав деяким словам у словниках. Це зазвичай вимірюється як лінгвістична відстань між паролем та словниковим словом.

Найпростішою лінгвістичною відстанню є відстань Левенштейна (або відстань редагування), яка є мінімальною кількістю операцій редагування (вставки, видалення та заміни), необхідних для перенесення одного слова в інше. Ця ідея була використана в метриці індексу якості пароля (PQI), запропонованій у [13] та уточненій у [14].

PQI пароля w - це пара λ = (D, L), де D - відстань Левенштейна w до базових слів словника, а L - ефективна довжина p, яка визначається як

$$L = m * \log_{10} N \qquad (1.11)$$

де m – довжина w, а N – розмір кодування, з якого взято символи w.

Ефективна довжина – це довжина, розрахована у «стандартизованому» кодуванні, наборі цифр D. Ідея полягає в тому, що пароль (наприклад, k38P довжини 4), взятий з кількох кодувань (D ∪ L ∪ U), так само важко зламати (або має приблизно таку ж кількість можливих кандидатів для злому), як і інший пароль (наприклад, 378902 довжини 6), взятий лише з набору цифр D.

Видно, що ефективна довжина пароля, заданого в (6), по суті така ж, як значення ентропії в (5), з постійним коефіцієнтом $\log^2$ 10. Обидва враховують довжину пароля, а також розмір кодування.

З мірою PQI, критерій якості, наведений у [13], стверджує, що пароль має хорошу якість, якщо D ≥ 3 та L ≥ 14.

1.3.2 Вимірювачі надійності паролів від постачальників послуг

Постачальники послуг використовують різні вимірювачі для оцінки надійності пароля з дещо відмінними алгоритмами та логікою. Хороший огляд та аналіз таких вимірювачів представлено в таблиці 1.1, де перераховано основні вимоги, що ставлять провайдери. Деякі з цих постачальників також враховують контекстну інформацію про користувача (наприклад, ім’я, прізвище, електронну пошту) під час класифікації якості пароля, що дозволяє виявити персоналізовані, а отже - менш безпечні комбінації. Більшість існуючих рішень базуються на традиційній моделі LUDS (використання великих і малих літер, цифр та

спеціальних символів), за винятком Dropbox, який застосовує більш сучасний підхід на основі оцінки ентропії та словникових атак.

Таблиця 1.1. Вимоги до паролів у різних постачальників

Сервіс	Шкала сили	Обмеження довжини		Потрібна кодування
		Хв	Макс	
Dropbox	Дуже слабкий, Слабкий, Так собі, Добре, чудово	6	72	Ø
Drupal	Слабкий, Справедливий, Добрий, Сильний	6	128	Ø
FedEx	Дуже слабкий, Слабкий, Середній, Сильний, Дуже сильний	8	35	1+нижчий,1+ верхній,1+цифра
Майкрософт	Слабкий, Середній, Сильний, Найкращий	1	–	Ø
Твіттер	Недійсний/Занадто короткий Очевидно, ні достатньо безпечно могло б бути більше безпечно, добре, Ідеальний	6	> 1000	Ø
Yahoo!	Слабкий, Сильний, Дуже сильний	6	32	Ø
eBay	Недійсний, Слабкий, Середній, Сильний	6	20	будь-які 2 набори кодування
Google	Слабкий, Справедливий, Добрий, Сильний	8	100	Ø
Скайп	Пор, Середній, Добре	6	20	2 кодування або тільки верхній
Apple	Слабкий, Помірний, Сильний	8	32	1+нижчий,1+ верхній,1+цифра
PayPal	Слабкий, Справедливий, Сильний	8	20	будь-які 2 набори кодування

PayPal вважає великі та малі літери одним набором символів.

Перевірник надійності паролів Dropbox під назвою zxcvbn використовує інший підхід для оцінки надійності пароля. Основна ідея полягає в перевірці, «наскільки поширений пароль згідно з кількома джерелами». Джерела включають поширені паролі з витікаючих наборів паролів, поширені назви з даних перепису населення та поширені слова у Вікіпедії. Алгоритм zxcvbn знаходить шаблони (підрядки) у паролі, які відповідають елементам у джерелах, і ці шаблони можуть перетинатися в паролі. Шаблони включають токен (logitech), обернений (DrowssaP), послідовність (jklm), повтор (ababab), клавіатуру (qAzxcde3), дату (781947) тощо. Потім він призначає оцінку спроби вгадування кожному збігу та, нарешті, шукає неперекриваючі суміжні збіги, які охоплюють пароль і мають мінімальну загальну кількість спроб вгадування.

Про надійність пароля: огляд та аналіз

Таблиця 1.2. Вивід багатоканальної перевірки паролів для password$1 [4]

Сервіс	Оцінка сили	
Apple	Помірний	2/3
Dropbox	Дуже слабкий	1/5
Drupal	Сильний	4/4
eBay	Середній	4/5
FedEx	Дуже слабкий	1/5
Google	Справедливий	3/5
Microsoft (версія 3)	Середній	2/4
PayPal	Слабкий	2/4
Скайп	Бідний	1/3
Твіттер	Ідеальний	6/6
Yahoo!	Дуже сильний	4/4

Алгоритми, що використовувалися цими постачальниками, призвели до значних розбіжностей в оцінці стійкості навіть для однакових комбінацій. Наприклад, пароль password$1 був оцінений як "дуже слабкий" у Dropbox та "дуже сильний" у Yahoo!, що добре ілюструє нестабільність та суб’єктивність існуючих підходів. Це свідчить про відсутність єдиного стандарту оцінки якості паролів і підкреслює потребу у створенні більш об’єктивного, гнучкого та прозорого механізму перевірки.

1.3.3 Метрика складності паролів

Ми пропонуємо метрику складності пароля, яка враховує як загальні вимоги LUDS, так і шаблони в паролі. Як і в багатьох інших мірах, кількість різних наборів символів, що використовуються в паролі, все ще відіграє важливу роль у цій метриці.

Крім того, ми враховуємо інші фактори, які можуть ускладнити вгадування пароля, такі як поєднання підрядків з одного набору символів, розташування спеціальних символів та підрядки в словнику.

Оскільки користувачі знають про використання різних наборів символів для створення паролів, вони частіше комбінують символи з одного набору, а не змішують їх. Наприклад, більш імовірним є пароль horse743, а не ho7r4se3. Останній вважається більш «складним», ніж перший, і його важче зламати. Ми знаходимо підрядки в паролі, які належать до одного набору символів, і підраховуємо кількість таких підрядків. Використовуючи той самий приклад, кількість підрядків у horse743 дорівнює 2 (horse і 743), тоді як кількість підрядків у ho7r4se3 дорівнює 6 (ho, 7, r, 4, se і 3). Однак ця кількість може бути більшою для довшого пароля, ніж для коротшого, тому ми беремо співвідношення цієї кількості до довжини пароля як фактор для нашої метрики.

Іншим фактором є розташування спеціальних символів. Багато користувачів використовують загальне слово, а потім додають спеціальний символ в кінці (або на початку). Наприклад, horse# може бути більш поширеним, ніж hor#se. Ми застосовуємо невелике покарання до цього шаблону, якщо пароль використовує тільки 2 набори символів (включаючи набір спеціальних символів).

Пароль, який збігається зі словом зі словника, є слабким, але пароль із підрядком, який збігається зі словом зі словника, може бути слабким або не слабким, залежно від довжини самого підрядка та його «ваги» в паролі. Багато паролів, обраних користувачами, містять підрядки, які є словами зі словника, але паролі можуть бути надійними. Наприклад, Pfan?6tk є досить сильним за більшістю критеріїв, які ми обговорювали, хоча він містить слово fan зі словника. Однак

довше слово зі словника (4 літери або більше) в паролі зробить його слабшим, особливо для відносно короткого пароля. Пароль Wfoot67 є набагато слабшим, ніж Wdfoot6237, хоча обидва містять слово foot зі словника. З огляду на ці міркування, наш показник складності пароля w обчислюється, як зазначено на формулі (1.12).

$$C(w) = \begin{cases} 0 & \text{if } l < 4 \\ n + (k/l) + s - p - (d/l) & \text{if } l \geq 4 \end{cases} \quad (1.12)$$

де n - кількість наборів символів у w, k - кількість підрядків з однаковим набором символів, l - довжина w, s - бонус, якщо w має особливі характеристики, p - штраф за положення спеціального символу, а d/l - штраф за наявність у словнику. Зокрема, s = 0,5, якщо w містить спеціальний символ, інакше - 0; p = 0,5, якщо спеціальний символ знаходиться на початку або кінці w і w має не більше 2 наборів символів, в іншому випадку 0; d - довжина підрядка, що є словом зі словника.

Ми можемо масштабувати метрику з діапазону 0–10 до 0–100 (у 10 разів), щоб її можна було обґрунтовано порівняти з іншими мірами, такими як ентропія NIST.

1.4 Аналіз екосистеми паролів в фокусі безпеки

Паролі використовувалися для надання доступу невідомим особам з давніх часів. Військові, шпигунські організації, організації підвищеної безпеки були свідками бурхливого використання паролів. І сьогодні вони використовуються не тільки для захисту комп'ютерів у традиційному розумінні, але і для контролю доступу до мобільних телефонів, будинків, банкоматів і багато чого іншого. Найчастіше паролі є єдиним засобом захисту додатків від несанкціонованого доступу, і, на жаль, багато користувачів не до кінця усвідомлюють важливість паролів. Як правило, вони встановлюють короткі, легко запам'ятовувані паролі, які дуже вразливі для атак. Мета даної статті - показати користувачам, наскільки вразливими можуть бути їхні особисті дані при використанні слабких паролів, продемонструвавши, наскільки легко або складно зламати паролі різної сили. Звичайно, при достатній обчислювальній потужності і швидкості жоден пароль не

буде захищений від атаки методом грубої сили. Однак, проявивши трохи пильності і знань, можна значно ускладнити завдання потенційним зловмисникам.

1.4.1 Основні аспекти безпеки паролів

Існує безліч аспектів безпеки паролів, які необхідно враховувати. До них відноситься спосіб зберігання паролів. Безпечне зберігання паролів має вирішальне значення для захисту паролів від зловмисних атак. Звичайний текст, хешування, солоне хешування, райдужні таблиці - все це різні методи зберігання паролів. Також необхідно враховувати, чи генеруються паролі людиною або комп'ютером. Паролі, згенеровані комп'ютером, зазвичай мають більш високий ступінь випадковості. Крадіжка паролів також є проблемою, яку слід враховувати. Пароль може бути вкрадений за допомогою соціальної інженерії, грубого форсування, кейлоггінгу і т. д. У наступних підрозділах розглядаються різні аспекти безпеки паролів.

А. Зберігання паролів

Пароль може складатися з символів, цифр і/або спеціальних знаків. Паролі в основному чутливі до регістру. Паролі можуть бути повністю цифровими. Вони називаються пасскодами і часто використовуються в якості PIN-кодів (персональних ідентифікаційних номерів) в банкоматах і інтернет-банкінгу. Паролі зберігаються в Інтернеті різними способами. Деякі з них набагато надійніші за інші, а деякі дуже вразливі для атак. У наступному розділі перераховано кілька найпопулярніших способів.

Паролі у вигляді звичайного тексту - це найпростіша форма зберігання пароля. Десь на сервері сайту є база даних, в якій зберігаються паролі та імена користувачів у вигляді звичайного тексту. Якщо пароль _PassText321', то в базі даних пароль зберігається як _PassText321'. Це найгірша форма зберігання паролів з точки зору безпеки. Якщо сайт зламають, а паролі зберігаються в зручному для читання вигляді, то всі паролі будуть негайно скомпрометовані. Хакер може прочитати всі паролі практично без додаткових зусиль.

Зашифровані паролі - Багато сайтів зберігають зашифровану форму пароля в базі даних на своєму сервері. При шифруванні використовується спеціальний ключ для перетворення пароля в випадковий рядок тексту. Перевага полягає в тому, що без ключа хакер не може отримати пароль. Все, що може бути отримано, - це випадковий зашифрований рядок. Недоліком є те, що ключ часто зберігається на тому ж сервері, де і паролі. Тому, якщо сервер зламаний і ключ знайдений, всі паролі можуть бути розшифровані та скомпрометовані. Сам факт того, що шифрування оборотне, тобто повідомлення може бути закодоване і розшифроване, становить загрозу безпеці.

Хешовані паролі - Хешування - це функція, яка перетворює пароль у випадковий довгий рядок букв і цифр. Перевага хешів перед шифруванням полягає в тому, що хеші є незворотними. Після того як пароль захешований, не існує алгоритму, що дозволяє змінити його назад на оригінальний. Хакерові доведеться хешувати кілька комбінацій одна за одною, щоб побачити, який хеш збігається з тим, що зберігається на сервері. Один зі способів зробити це - райдужні таблиці, які дуже швидко піддаються обчисленням. Хакери також можуть використовувати атаку методом грубої сили, коли всі можливі комбінації букв і цифр перебираються, хешуються і порівнюються з хешем, отриманим з бази даних. Цей метод може зайняти дуже багато часу і багато в чому залежить від потужності комп'ютера. Однак сьогодні комп'ютери стали дуже швидкими, і атаки грубої сили, такі як John The Ripper, дозволяють зламувати паролі досить ефективно. Існують різні типи алгоритмів хешування, такі як MD5, SHA-1, SHA-256 і SHA-512.

Солоні хеші - щоб зробити хеші більш безпечними, до них можна додати «сіль». Це означає, що перед хешуванням пароля до нього додається випадковий рядок символів або з префіксом, або з постфіксом. Кожен пароль має свою сіль. Навіть якщо солі зберігаються в базі даних, зламати паролі за допомогою райдужної таблиці буде дуже складно, оскільки солоні паролі довгі, складні та унікальні. Солоні хеші можна зламати методом грубої сили, але це займе набагато більше часу. Використання двох солей, однієї публічної та однієї приватної, також може захистити пароль від атак в автономному режимі.

B. Паролі, згенеровані людиною, і паролі, згенеровані випадковим чином

Паролі можуть бути як згенерованими людиною, так і випадковими. Генератор випадкових чисел генерує випадковий рядок чисел із символами із заздалегідь визначеного набору символів. Кожен символ із цього набору має однакову ймовірність бути обраним. Генератор псевдовипадкових чисел (ГПСЧ) генерує випадкову послідовність і знаходить застосування в криптографії. Числа ГПСЧ не є по-справжньому випадковими, оскільки генеруються з невеликого набору початкових значень. Цей набір називається станом ГПСЧ, і до нього входить істинно випадкове зерно.

Паролі, згенеровані людиною, ніколи не бувають по-справжньому випадковими. Згенеровані людиною паролі зазвичай легко запам'ятати. Люди вибирають паролі, які зазвичай схожі на якісь елементи їхнього життя. Наприклад, адреси, дати народження, імена родичів або слова, які часто використовуються в повсякденному житті. Також часто використовуються паролі типу «abcdefg» або «1 2 3 4 5 6».

Оскільки у людей є кілька облікових записів, важко запам'ятати стільки різних паролів. Тому більшість воліє використовувати короткі, легко запам'ятовувані паролі. Це робить згенеровані людиною паролі більш вразливими й легко вгадуваними. Також було відзначено, що веб-користувачі схильні до повторного використання своїх паролів. Якщо один пароль стане відомим, то під загрозою опиняться відразу кілька облікових записів. Оскільки більшість паролів є згенерованими людиною, кожен користувач повинен сам стежити за тим, щоб паролі були надійними та безпечними.

C. Крадіжка паролів

Витік паролів може відбутися декількома способами. Зловмисник може зламати базу даних сайту, де зберігаються облікові дані користувачів, і дізнатися величезну кількість паролів. Крадіжка може відбутися і на особистому рівні. Користувач може записати пароль десь, і він потрапить до рук зловмисників. Або ж користувач може встановити дуже простий і очевидний пароль, який легко вгадати. Соціальна інженерія, фішинг або кейлоггери також можуть

скомпрометувати паролі. Дуже часто паролі можуть бути розкриті за допомогою грубої сили або автономних атак за словником.

1.4.2 Щодо сили паролів

При атаці методом грубої сили перебираються всі можливі комбінації в заданому наборі символів і намагаються зіставити їх з оригінальним паролем. Чим більше число можливих комбінацій, тим більше часу потрібно алгоритму для генерації припущень. В середньому, перш ніж знайти правильну комбінацію, перебирається майже половина від загальної кількості комбінацій. Чим більше часу потрібно для злому пароля, тим він надійніший. Тому логічно зробити висновок, що чим більша довжина пароля, тим краще він протистоїть атаці грубої сили. Нехай довжина пароля, який необхідно зламати, дорівнює N. Нехай пароль складається тільки з літер нижнього регістру. Це утворює набір символів. Можливих кандидатів на кожен символ пароля - 26. Для більш загального випадку нехай набір символів складається з k символів. Тоді кількість можливих паролів може бути N^k. Таким чином, довжина пароля може збільшуватися або за рахунок збільшення N, або за рахунок збільшення k.

Якщо пароль має довжину 6 і складається тільки з малих літер, то кількість можливих паролів дорівнює 26^6, що становить 308915776. Якби він складався з символів верхнього та нижнього регістрів, то розмір набору символів становив би 52, а кількість можливих варіантів – 52^6, що становить 1,9770 x 10^{10}. Якщо розмір пароля 7, то можливості будуть 26^7 і 52^7.

Щоб довести, що довгий пароль дійсно складніше зламати, ніж короткий, введені користувачем паролі були хешовані, а потім перебрані. Спочатку паролі хешувалися за допомогою хеш-функції MD5. Після хешування пароля створюються комбінації фіксованої довжини. Кожна комбінація хешується за допомогою тієї ж хеш-функції MD5 і порівнюється з хешем оригінального пароля. Якщо знайдено збіг, функція завершується. Слово, хеш якого збігся з хешем оригіналу, є правильним паролем. У найгіршому випадку код перевірить всі комбінації, перш ніж знайде збіг. Час, витрачений на злом кожного пароля, підраховується і виводиться в таблицю.

A. Числові тести

Перші тести проводилися для паролів з 5 букв. Було підраховано час, витрачений на злом одного пароля, і тест був повторений для ста різних паролів, що складаються тільки з малих літер від a до z. Наступний набір тестів був проведений для паролів з 6 букв. Знову обчислювався час, необхідний для злому одного пароля, і тест повторювався для ста різних паролів з набору символів, що складався з малих літер a-z. У таблиці наведено 20 результатів тесту. Як видно з таблиці 1.3, час, необхідний для злому пароля з шести літер, значно вищий, ніж для пароля з п'яти літер. Також з таблиці видно, що збільшення часу відбувається більш-менш рівномірно. Як видно з графіка, середнє збільшення часу становить 26.

B. Буквено-цифрові тести

Буквено-цифрові тести Наступний набір тестів був проведений для розрахунку часу злому 6-буквених буквено-цифрових паролів. Для цього було протестовано двадцять паролів. Літерно-цифрові паролі порівнювалися з двадцятьма випадково обраними літерними паролями з 6 літер і обчислювалися їх графіки, які показують, наскільки посилюється пароль при додаванні до нього набору символів. Для буквено-цифрових паролів набір символів дорівнює 36. Отже, для 6-буквеного буквено-цифрового пароля число можливостей дорівнює 36^6= 2176782336, а для 6-буквеного буквеного пароля число можливостей дорівнює 26(^{6}) =308915776.

C. Тести для декількох випадків

Наступний набір тестів був проведений для розрахунку часу перебору паролів, що складаються з літер як верхнього, так і нижнього регістру. Набір символів для паролів з декількома регістрами становить 52. Було протестовано двадцять випадкових паролів з 6 літер кожен. Вони порівнювалися з двадцятьма паролями в нижньому регістрі і обчислювалися їх графіки. Для кожного багатобуквеного пароля з 6 букв число можливостей дорівнює 52^6=19770609664, а для буквеного пароля з 6 букв число можливостей дорівнює 26(^{6}) =308915776. Графічні результати підтверджують той факт, що збільшення набору символів значно посилює пароль.

Таблиця 1.3. Розрахунковий час злому 5-літерних, 6-літерних, літерно-цифрових паролів і паролів у подвійному регістрі.

Ст./№	5-літерний пароль	Час на злом	Пароль з 6 букв	Час на злом	Буквено-цифровий пароль	Час на злом	Пароль у подвій-ному регістрі	Час на злом
1	Bales	57795.2	abases	1599703	abas34	9944469	Acajou	72887164
2	Candy	58503.7	ballad	1621442	a346be	11524403	Blunts	75462234
3	Delta	52585.6	bennis	1457532	aes3er	10989201	Chough	73235678
4	Egads	56186.9	chinos	1763321	45alze	11031055	Diesel	69984567
5	Feign	55397	daddle	1705889	bes567	10134510	Ethoxy	77567893
6	Garum	47403.85	doting	1514065	045kat	10139948	Flabby	74221345
7	Hoary	68526.15	elects	1557074	bute90	9567085	Gnawed	80556784
8	Igapo	61641.75	fabled	1394745	blips2	10044859	Hector	79556788
9	Lobby	49092.6	glades	1737407	cat101	11071539	Imagos	77564856
10	Maims	60824.15	hacker	1659651	cupola	11116028	Jovial	76554345
11	Nutsy	62828.85	incite	1768656	citco5	11043269	Keener	77908456
12	Peare	60157.1	jinxed	1393465	celt67	11191393	Legmen	72345677
13	Rearm	66847.9	khazen	1613898	delta4	11272714	Macaco	71236578
14	Rough	66346.05	legmen	1398087	5doggy	11167292	Nankin	78665432
15	Skids	67386.45	milady	1623292	death8	11417336	Oafish	69783321
16	Taboo	67245.85	nibble	1642988	dupe33	11597704	Pablum	70112345
17	Thyme	66887.05	odours	1636991	epm4t6	11591159	Quiche	71864579
18	Users	42237.35	phenom	1554071	epm4t6	10855216	Rabato	74556789
19	Xylem	56794.2	quaked	1592733	34egg7	9858146	Sebums	73455675
20	Zonal	61287.9	stomps	1651824	etoph4	10832474	Valued	75338904
21	Brisk	58221.7	honcho	1605678	foggy3	10455933	Yonder	78012256
22	Crimp	61043.8	piston	1667832	grab43	10978211	Zither	79100843

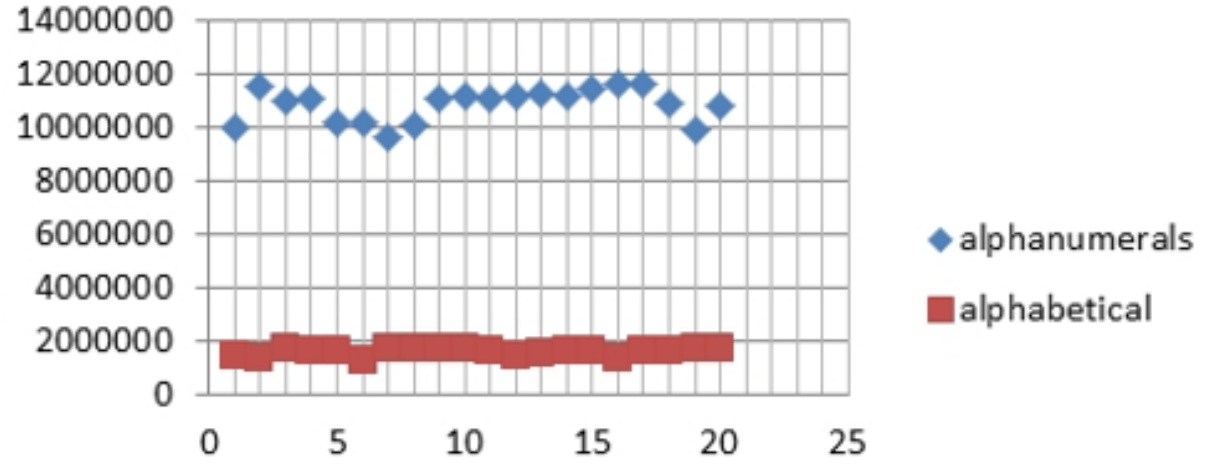


Рисунок 1.3. Порівняння часу витраченого на злом 6-літерного та літерно-цифрового паролів

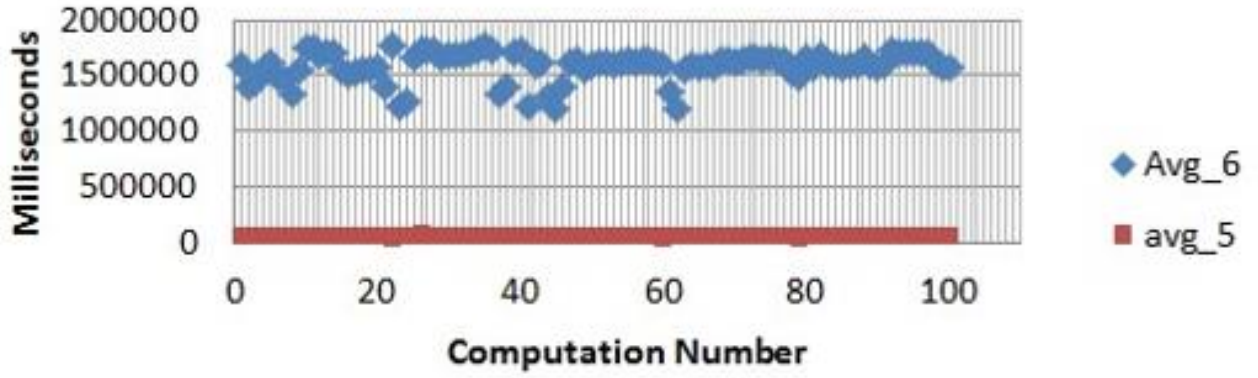


Рисунок 1.4. Час зламу пароля залежно від довжини, витрачений на злом паролів з 5 і 6 букв

1.4.3 Ентропія

А. Ентропія як інформаційний зміст

Ентропія визначається в контексті імовірнісної моделі. Код, що генерує рядок - BBBBB...‖, матиме ентропію, рівну нулю, оскільки немає ніякої невизначеності щодо наступного символу. Відомо, що наступним символом має бути _B'. Якщо 256-бітний ключ згенерований випадковим чином, то він має 256 біт ентропії. Але якщо кожна цифра не має рівної ймовірності, то ентропія не буде відображати справжню непередбачуваність. Якщо ключ «криптографічний» в 50% випадків і дійсно випадковий 256-бітний ключ, то ентропія дорівнює приблизно 128 бітам, але кількість припущень, необхідних для перебору, може бути не 2^{128-1}, а 2^{256-1},

оскільки в половині випадків пароль вдається зламати з першої спроби, а в інших випадках його доводиться вгадувати.

B. Сильні сторони пароля з точки зору ентропії

Коли мова йде про паролі, ентропія використовується для визначення сили пароля з точки зору його інформативності, що вимірюється в бітах. Для пароля довжиною m біт буде потрібно 2^m спроб, щоб перебрати всі можливості при атаці грубою силою. Очевидно, що чим вище ентропія, тим вище стійкість пароля.

Ентропія визначається наступним чином:

$$H = L * log_2 N \tag{1.13}$$

де L - довжина пароля, а N - розмір символу.

Нехай пароль - _Ast34beta1, який вибирається з набору символів розміром 62. Тоді ентропія дорівнює $H = 10 * \log_{10}62/\log_{(10)\ 2}$; що становить H= 59,541 біт. Таким чином, ентропія пароля залежить як від довжини, так і від кількості всіх можливих символів. Що більше збільшує ентропію на біт - довжина або розмір набору символів? З рівняння ясно, що довжина пароля має більше значення.

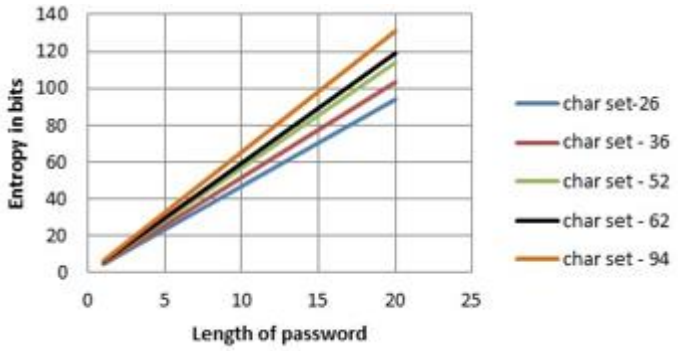


Рисунок 1.5. Залежність ентропії від довжини

Таблиця 1.4. Ентропія паролів зростаючої довжини і фіксованого набору символів

Sr. no	Charset-26	Charset-36	Charset-52	Charset-62	Charset-94
1	4.700439718	5.169925001	5.700439718	5.95419631	6.554588852
2	9.400879436	10.33985	11.40087944	11.90839262	13.1091777
3	14.10131915	15.509775	17.10131915	17.86258893	19.66376656
4	18.80175887	20.67970001	22.80175887	23.81678524	26.21835541

5	23.50219859	25.84962501	28.50219859	29.77098155	32.77294426
6	28.20263831	31.019550	34.20263831	35.72517786	39.32753311
7	32.90307803	36.18947501	39.90307803	41.67937417	45.88212196
8	37.60351775	41.35940001	45.60351775	47.63357048	52.43671081
9	42.30395746	46.52932501	51.30395746	53.58776679	58.99129967
10	47.00439718	51.69925001	57.00439718	59.5419631	65.54588852
11	51.7048369	56.86917502	62.7048369	65.49615941	72.10047737
12	56.40527662	62.03910002	68.40527662	71.45035572	78.65506622
13	61.10571634	67.20902502	74.10571634	77.40455204	85.20965507
14	65.80615605	72.37895002	79.80615605	83.35874835	91.76424392
15	70.50659577	77.54887502	85.50659577	89.31294466	98.31883278
16	75.20703549	82.71880002	91.20703549	95.26714097	104.8734216
17	79.90747521	87.88872502	96.90747521	101.2213373	111.4280105
18	84.60791493	93.05865003	102.6079149	107.1755336	117.9825993
19	89.30835464	98.22857503	108.3083546	113.1297299	124.5371882
20	94.00879436	103.3985	114.0087944	119.0839262	131.091777

1.4.4 Деякі вимоги до надійності паролів від вендорів

Значення надійних паролів було досить докладно розглянуто в попередніх розділах. Очевидно, що крім заходів безпеки, які організація вживає для захисту даних користувачів, відповідальність за надійність паролів лежить і на користувачах. Користувачів можна змусити вводити певний ступінь складності своїх паролів, встановивши деякі необхідні правила. Користувач повинен дотримуватися цих правил при виборі нового пароля під час реєстрації. Дослідження, проведене в [6], доводить, що люди вибирають слабші паролі для сайтів, на яких діють несуворі правила, а реєстрація нового облікового запису досить захищена. Переглядаючи сторінки входу/реєстрації/реєстрації наступних веб-гігантів, вдалося отримати достатньо даних, щоб зрозуміти, яких правил вони вимагають від своїх клієнтів дотримуватися при створенні нового облікового запису.

Ebay.com - ebay.com дотримується наступних правил для паролів.

- Мінімум шість символів і максимум 20 символів
- Принаймні одна цифра та/або спеціальний символ
- Має бути чутливим до регістру. Тобто в них повинні бути присутніми як великі, так і малі літери.
- Паролі класифікуються як «слабкі», «середні» або «сильний».

Користувач отримує повідомлення, якщо пароль недійсний або занадто короткий. Пароль класифікується як «середній» або «слабкий», якщо в ньому не використовуються літери, цифри та спеціальні символи. Щоб класифікувати пароль як «сильний», пароль повинен складатися не тільки з літер (як верхніх, так і нижніх), цифр і спеціальних символів, але і мати довжину більше 6. Пароль довжиною шість з усіма комбінаціями символів оцінюється як «середній», а довжиною сім і більше - як «сильний».

Amazon.com - amazon.com дотримується наступних правил щодо пароля при реєстрації облікового запису користувача

- Повинен складатися мінімум з 6 букв
- Повинен складатися з верхнього та нижнього регістру та/або комбінації літер і цифр.

Flipkart.com - flipkart.com дотримується наступних правил введення паролів при реєстрації

- Він повинен складатися мінімум з чотирьох символів.

Facebook.com - facebook.com дотримується наступних правил та умов щодо паролів при реєстрації нового користувача.

- Довжина пароля повинна складати не менше 6 символів.

Adobe.com - adobe.com дотримується наступних умов для паролів при реєстрації нового користувача.

- Довжина пароля повинна становити не менше 6 символів.

Hotmail.com - hotmail.com застосовує наступні умови до паролів при реєстрації нового користувача.

- Довжина пароля повинна становити не менше 8 символів.

- Повинен містити будь-які два з наступних символів - великі літери/малі літери/цифри/спеціальні символи.

Після короткого аналізу правил, яких дотримуються вищезазначені сайти, можна прийти до висновку, що найменшою безпекою паролів володіє flipkart.com. Найсуворіших правил дотримується ebay.com, за яким слідує hotmail.com. Їх обмеження змушують користувачів встановлювати паролі, які, природно, важко піддаються перебору.

1.5 Розробка системи контролю налійності пародів для корпоративного сайту

У сучасному цифровому середовищі питання безпеки інформаційних систем набуває дедалі більшого значення. Одним із ключових елементів захисту даних є система автентифікації, яка у більшості випадків базується на паролях. Незважаючи на розвиток багатофакторної автентифікації та інших засобів захисту, паролі залишаються основним методом ідентифікації користувачів у корпоративних системах.

Недостатня надійність паролів часто призводить до витоків конфіденційної інформації, фінансових втрат, а також до порушення репутації компанії. Статистика показує, що значна частина користувачів створює слабкі паролі, які легко піддаються атакам перебором або соціальній інженерії. Це особливо критично для корпоративних веб-ресурсів, які містять чутливу інформацію та мають підвищені вимоги до кіберзахисту.

Метою даної роботи є розробка системи контролю надійності паролів для корпоративного сайту, яка дозволятиме виявляти слабкі паролі, надавати користувачам рекомендації щодо їх посилення та інтегруватися в існуючу інфраструктуру автентифікації. Такий підхід сприятиме підвищенню загального рівня інформаційної безпеки в компанії та зменшенню ризику несанкціонованого доступу до внутрішніх ресурсів. Впровадження подібної системи також забезпечить більш ефективний контроль за політикою паролів та підвищить усвідомленість користувачів щодо важливості безпечних облікових даних.

1.5.1 Складання технічного завдання

Створення веб-системи, яка аналізує надійність паролів користувачів під час їх створення або зміни, виявляє слабкі паролі, надає рекомендації щодо їх посилення та інтегрується у процес автентифікації корпоративного сайту. Система має підвищити рівень кібербезпеки організації шляхом запобігання використанню ненадійних паролів.

Основні завдання системи полягають у: реалізація механізму перевірки складності пароля відповідно до встановлених критеріїв (довжина, наявність цифр, символів, регістру тощо); виявленні використання поширених або скомпрометованих паролів (через базу даних зламаних паролів); надання зворотного зв'язку користувачу у вигляді порад для покращення пароля; логування спроб введення слабких паролів для подальшого аналізу; інтеграція з існуючим механізмом автентифікації корпоративного веб-додатку.

До функціоналу входить веб-інтерфейс для перевірки пароля в режимі реального часу; панель адміністратора для налаштування критеріїв надійності паролів; сумісність із типовими фреймворками автентифікації; можливість підключення до зовнішніх API або локальних баз скомпрометованих паролів; захист від атак типу brute-force та ін’єкцій.

Також слід звернути увагу на обмеження такі як: система не повинна зберігати введені паролі в незашифрованому вигляді; не допускається використання застарілих криптографічних алгоритмів; забороняється порушення політик безпеки, прийнятих у компанії.

1.5.2 Вибір методів та інструментів реалізації

У процесі розробки системи контролю надійності паролів для корпоративного сайту було прийнято рішення використати сучасний стек технологій, орієнтований на безпечну клієнтську обробку, високу продуктивність та зручність розробки.

Клієнтська частина (FrontEnd) реалізована як односторінковий застосунок (SPA) з використанням наступних інструментів:

Мова програмування:

TypeScript - типізована надбудова над JavaScript, що дозволяє уникати великої кількості помилок на етапі компіляції. Застосовується як на клієнтській, так і на серверній частині додатку.

Фреймворк та збирач:

Vite - сучасний інструмент для збирання фронтенду, який забезпечує швидке оновлення модулів і зручну розробку SPA-додатків.

React 18 - основний фреймворк для побудови інтерфейсу користувача, який забезпечує компонентний підхід і ефективну роботу з віртуальним DOM.

Маршрутизатор:

Wouter - легковажний маршрутизатор для клієнтської навігації.

Бібліотеки:

Lucide React - бібліотека іконок з підтримкою кастомізації.

Framer Motion - бібліотека анімацій для покращення користувацького досвіду.

Стили та UI:

Tailwind CSS - утилітарна CSS-бібліотека, що дозволяє швидко стилізувати компоненти, не виходячи з HTML-структури.

Radix UI - набір headless-компонентів інтерфейсу, що спрощує створення доступних та гнучких елементів взаємодії.

Серверна частина (BackEnd):

Серверна частина виконує допоміжну роль, переважно для обслуговування файлів додатку та базової маршрутизації:

Node.js - серверне середовище виконання JavaScript.

Express.js - мінімалістичний веб-фреймворк для створення REST API та обробки HTTP-запитів.

TypeScript - застосовується і на сервері для типобезпеки.

TSX та ESBuild - використовуються для розробки та збирання серверної частини з високою продуктивністю.

Drizzle ORM - налаштований для потенційної взаємодії з базою даних (PostgreSQL), хоча в поточній реалізації використовується зберігання в пам’яті.

Середовище розгортання:

Replit - середовище для розробки, яке забезпечує онлайн-перегляд, гарячу перезагрузку та зручну співпрацю.

Vercel - платформа для продакшн-розгортання, яка дозволяє швидко публікувати застосунок, підтримує статичні сайти та serverless-функції.

Перевірка паролів:

Локальна реалізація функції перевірки складності паролів (із урахуванням довжини, символів, регістру тощо).

Можливе підключення до сторонніх API для перевірки наявності пароля в базах злитих облікових даних.

Додаткові інструменти:

Git для контролю версій.

ESLint / Prettier для підтримки єдиного стилю коду.

tsconfig.json для налаштування середовища розробки на TypeScript.

Також для покращення продуктивності було застосовано збирання з Vite, що зменшує час запуску; оптимізацію об'єму коду за допомогою tree shaking; відкладене завантаження (lazy loading) окремих компонентів.

1.5.3 Опис можливостей та етапів реалізації продукту

Розроблений програмний продукт представляє собою клієнт-серверний вебзастосунок, основною метою якого є забезпечення надійної, зручної та безпечної перевірки паролів на їхню стійкість до зламу. Система надає користувачеві можливість у реальному часі перевіряти введений пароль, отримуючи оцінку його сили, а також формувати надійні паролі з використанням криптографічно безпечного генератора. Особливістю реалізації є те, що всі операції з обробки та оцінки пароля виконуються виключно на стороні клієнта, що гарантує конфіденційність і виключає ризик перехоплення чутливих даних. Інтерфейс реалізовано з використанням сучасних технологій - React, TypeScript та Tailwind

CSS, що забезпечує адаптивність, інтерактивність та зручність взаємодії для кінцевого користувача.

Розробка продукту здійснювалась поетапно. На початковому етапі було сформульовано вимоги до функціоналу системи, визначено архітектуру рішення, обрано відповідні інструменти та програмні засоби. Далі реалізовано клієнтську частину, включаючи логіку аналізу пароля, генератор, а також візуальні елементи інтерфейсу, що забезпечують зворотний зв'язок у зручній формі. Паралельно було створено серверну оболонку на базі Node.js і Express, яка відповідає за розгортання застосунку і його базову маршрутизацію. Після інтеграції компонентів здійснено тестування системи з метою перевірки її стабільності, правильності обробки вхідних даних та відповідності очікуваним результатам. Завершальним етапом стало розгортання вебзастосунку на хмарній платформі Vercel з налаштуванням середовища для публічного доступу.

Таким чином, продукт відповідає заявленим функціональним та технічним вимогам, забезпечуючи швидку, безпечну та ефективну перевірку паролів, що може використовуватись як частина корпоративного або навчального середовища для підвищення рівня цифрової гігієни користувачів.

1.5.4 Аналіз результатів розробки

У рамках дипломного проєкту було створено систему контролю надійності паролів для корпоративного сайту, яка забезпечує високий рівень безпеки користувачів за рахунок інноваційного підходу до оцінки паролів і генерації безпечних паролів.

Для реалізації клієнтської частини було застосовано сучасні фреймворки і бібліотеки: React 18 як основний UI-фреймворк, Vite для швидкої збірки і розробки, TypeScript для типізації та підвищення якості коду. Для маршрутизації використано Wouter, а стилізація інтерфейсу виконана за допомогою Tailwind CSS і компонентів Radix UI. Візуальні ефекти реалізовано через Framer Motion, що покращує користувацький досвід.

Основний функціонал - персоналізований алгоритм аналізу сили пароля, розроблений на JavaScript, який оцінює надійність пароля в режимі реального часу

без передачі даних на сервер. Також реалізовано криптографічно безпечний генератор паролів.

На серверній стороні застосовано Node.js з Express.js і TypeScript, що забезпечує надійність і масштабованість бекенду. Сервер використовується для обслуговування файлів додатку, при цьому вся обробка паролів виконується на клієнтській стороні для підвищення конфіденційності.

Для розробки використовувалась платформа Replit, яка дозволила організувати швидкий цикл розробки з гарячою перезагрузкою. Для виробничого розгортання застосовано Vercel, що забезпечує швидке і безвідмовне завантаження сайту.

Проведене тестування показало коректну роботу всіх ключових функцій: оцінка сили пароля відбувається миттєво, без затримок, а генератор створює надійні паролі. Час завантаження сайту на хостингу не перевищує 2 секунд, що відповідає сучасним вимогам.

Інтерфейс користувача відзначається інтуїтивністю та адаптивністю, що забезпечує зручність використання на різних пристроях. Безпека реалізована на високому рівні - паролі ніколи не передаються на сервер і не зберігаються.

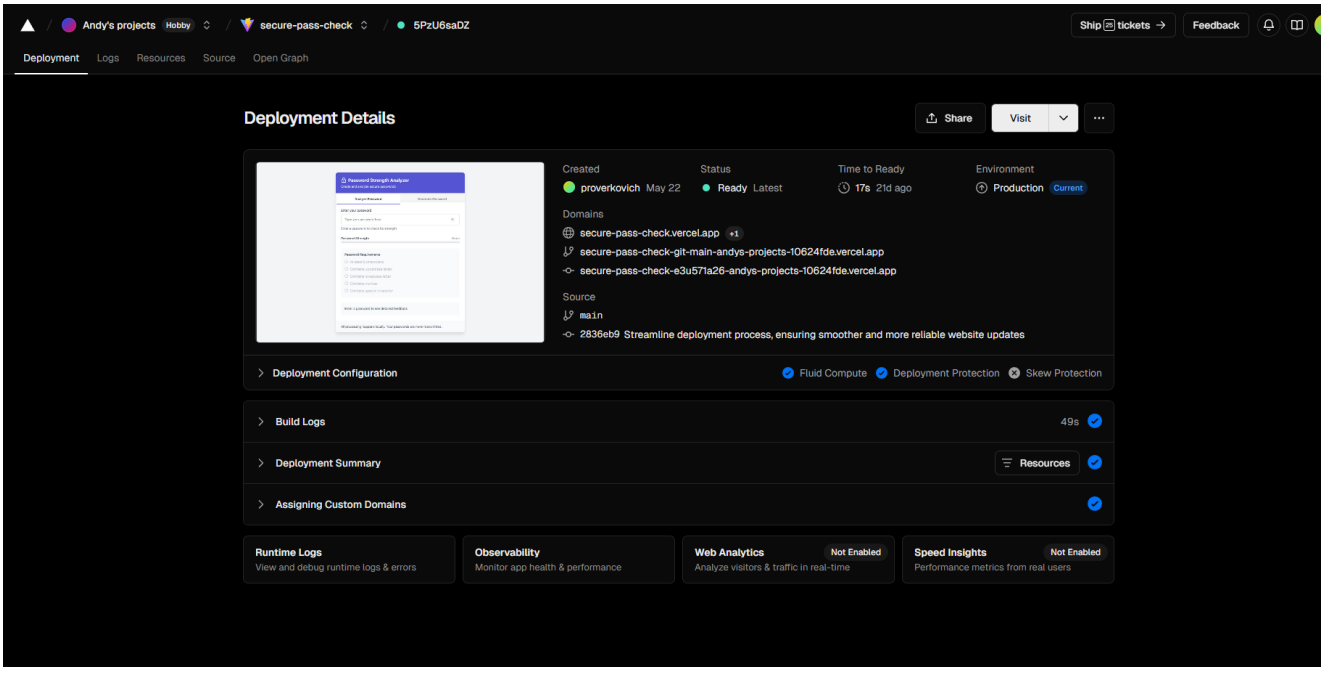


Рисунок 1.6. Хостингова сторінка

На цьому рисунку показано інтерфейс платформи хостингу Vercel, де розміщено розроблений веб-сайт системи контролю надійності паролів. Відображається статус розгортання проєкту, що підтверджує успішне завантаження та безперебійну роботу сайту в продуктивному середовищі. Платформа забезпечує швидке оновлення версій, автоматичне масштабування і надійність, що дозволяє підтримувати високу продуктивність та доступність ресурсу для користувачів у будь-який час.

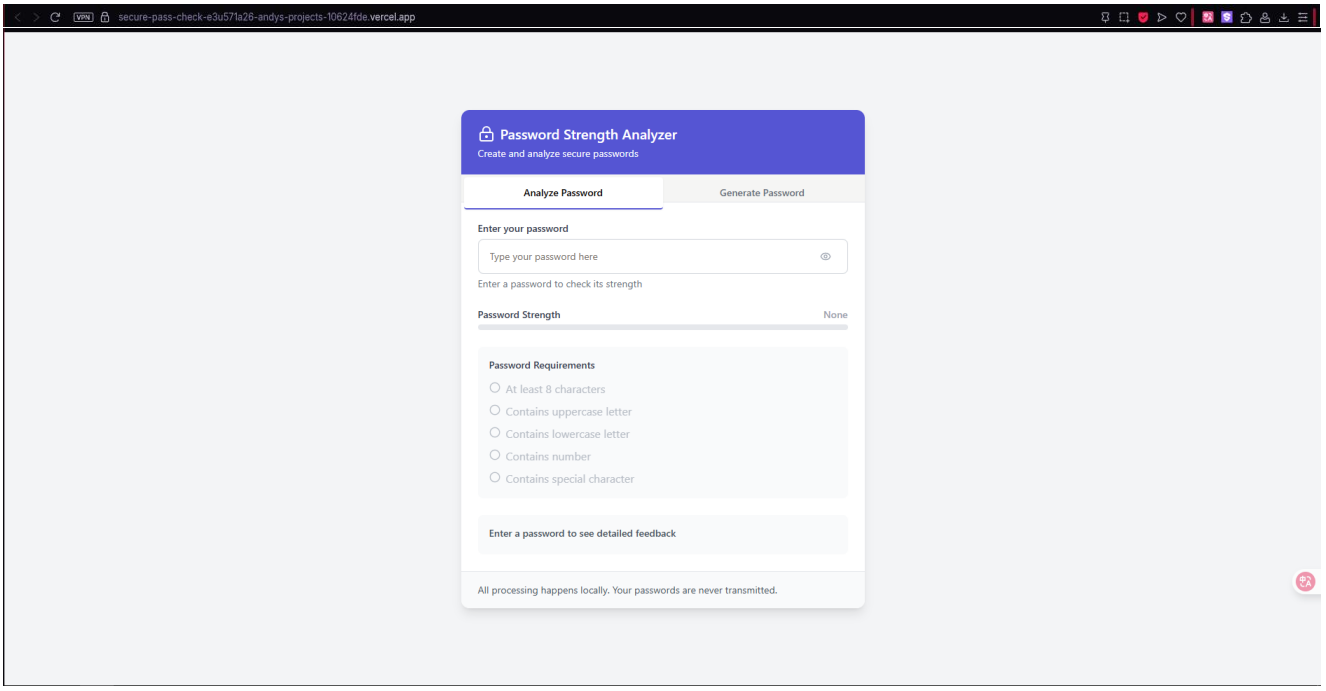


Рисунок 1.7. Головна сторінка веб-сайту з формою введення пароля

На цьому рисунку зображено головний інтерфейс системи контролю надійності паролів. Користувач бачить поле для введення пароля та індикатор його сили, який динамічно змінюється залежно від введених символів. Інтерфейс виконаний у сучасному стилі з використанням Tailwind CSS, що забезпечує чіткий і зрозумілий дизайн.

Компоненти інтерфейсу адаптовані до різних розмірів екранів, що гарантує зручність користування як на десктопних пристроях, так і на мобільних.

Додатково передбачено перемикач, який дозволяє миттєво перейти до генератора паролів - це дає змогу користувачам створити безпечну комбінацію символів за заданими критеріями без необхідності залишати сторінку.

Доступ до додатку не вимагає авторизації: достатньо відкрити посилання, після чого одразу відкривається інтерфейс перевірки пароля. Перехід між вкладками “Аналіз пароля” та “Генератор” реалізовано за допомогою зручного інтерфейсного перемикача, що дозволяє швидко змінювати функціональність без перезавантаження сторінки чи втрати введених даних

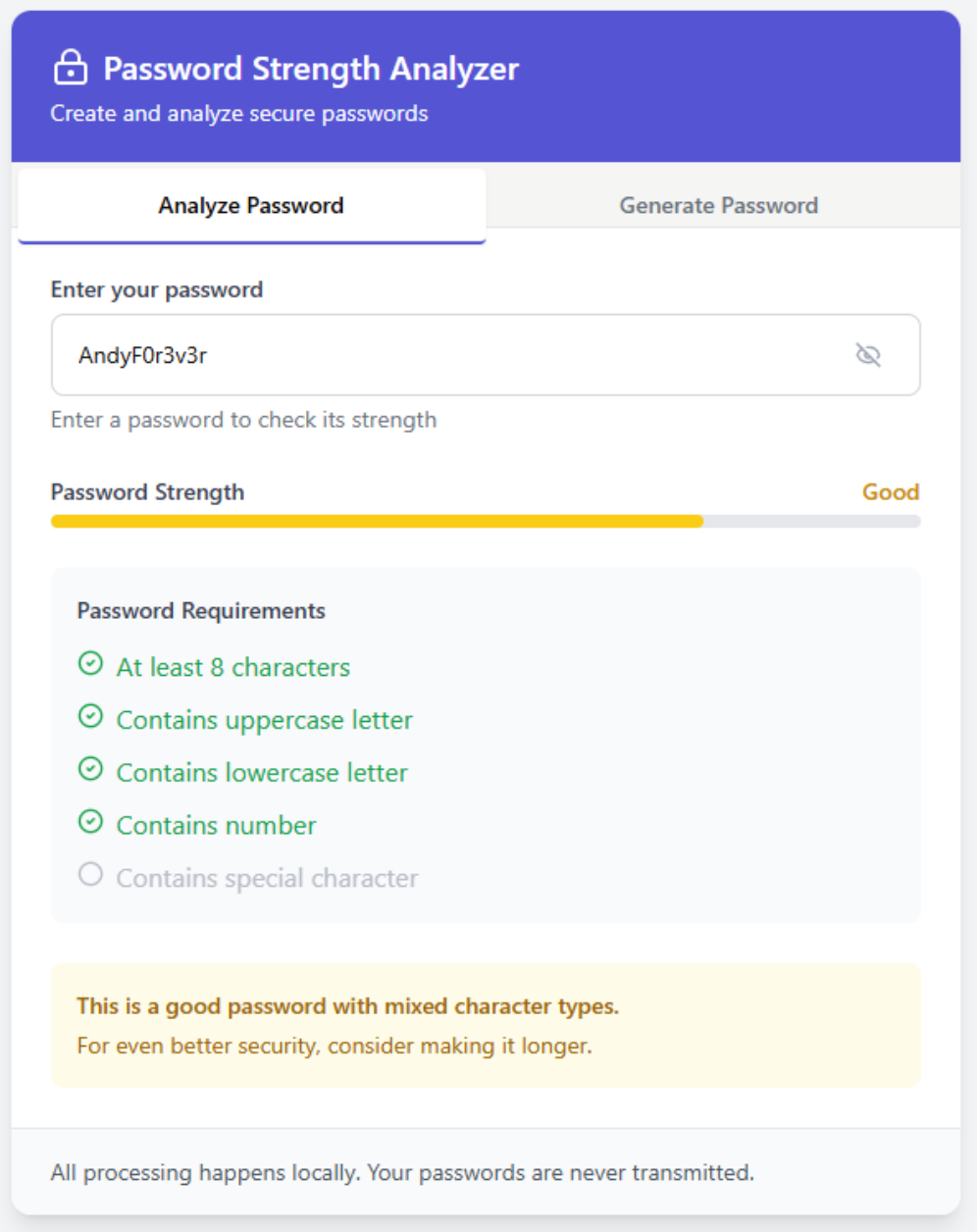


Рисунок 1.8. Аналізатор паролю

На цьому рисунку зображено головний інтерфейс системи контролю надійності паролів. Користувач бачить поле для введення пароля та індикатор його сили, який динамічно змінюється залежно від введених символів. Інтерфейс виконаний у

сучасному стилі з використанням Tailwind CSS, що забезпечує чіткий і зрозумілий дизайн.

Окрім цього, система надає текстові підказки в реальному часі, що дозволяє миттєво реагувати на виявлені слабкі сторони введеного пароля та покращувати його ще до завершення введення.

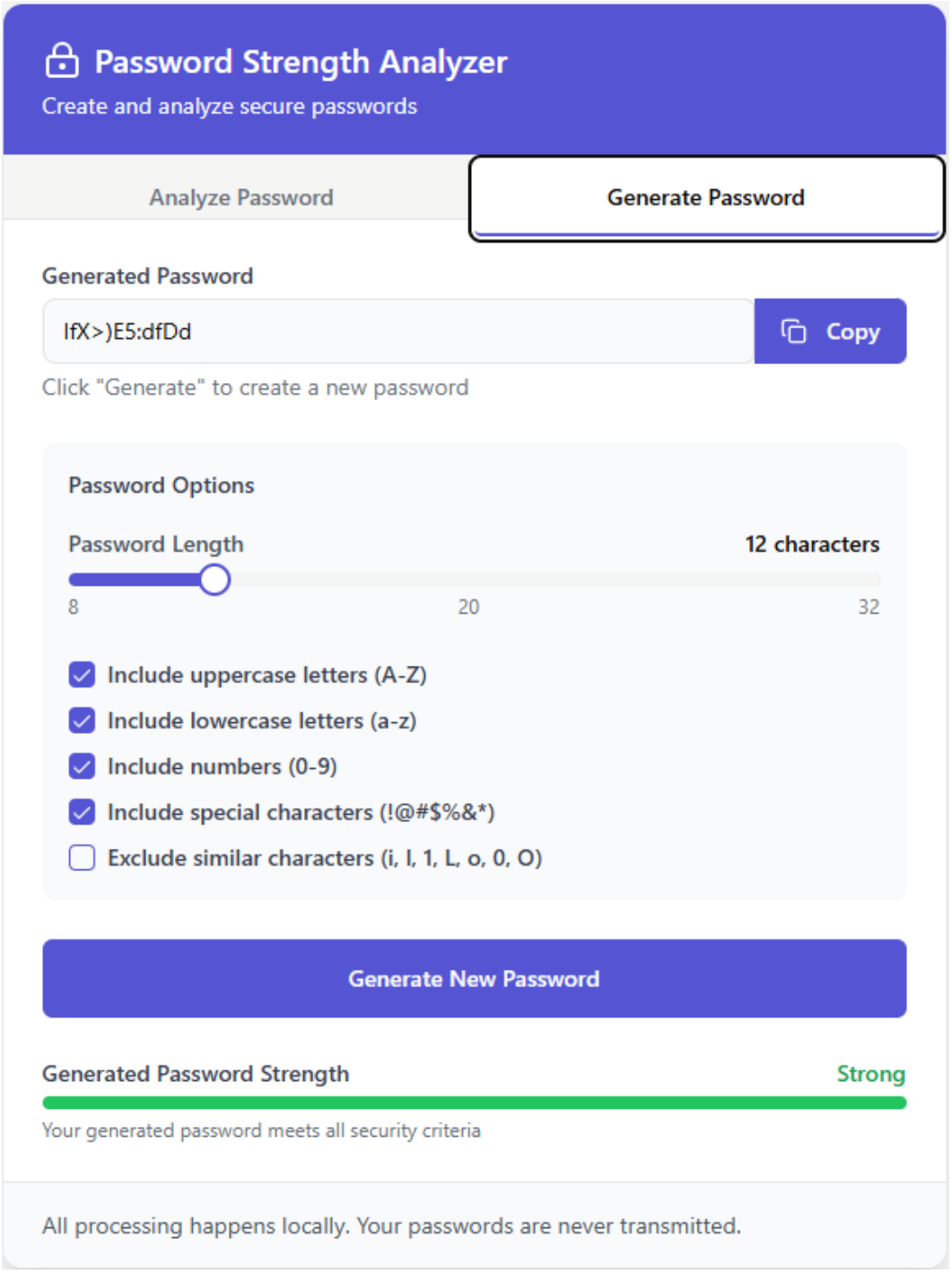


Рисунок 1.9. Генератор паролю

На цьому рисунок представлений інструмент для автоматичного створення надійних паролів. Користувач може згенерувати пароль заданої довжини з різними

параметрами (великі та малі літери, цифри, символи). Інтерфейс простий і зрозумілий, що дозволяє швидко отримати криптографічно безпечний пароль. Згенерований пароль відразу можна скопіювати для подальшого використання, що підвищує загальний рівень безпеки корпоративного сайту.

1.5.5 Загальна різниця між веб-сайтом та веб-додатком:

Веб-сайт - це електронна вітрина з інформацією.

Приклади: Блоги, новинні сайти, корпоративні сайти, лендінги.

Особливості:

- Переважно статичний контент.
- Менше інтерактивних елементів.
- Користувач є спостерігачем, а не активним учасником.
- Найчастіше, він не вимагає авторизації.

Веб-додаток - це онлайн-програма, яка виконує задачі.

Приклади: Gmail, Google Docs, онлайн-банкінг, системи реєстрації, редактори.

Особливості:

- Динамічний контент, який змінюється у відповідь на дії користувача.
- Багато інтерактивності: форми, обробка запитів, збереження даних.
- Часто потребує авторизації.
- Має клієнтську та серверну логіку.

Таблиця 1.5. Порівняння

Характеристика	Веб-сайт	Веб-додаток
Призначення	Інформаційне	Функціональне
Взаємодія з користувачем	Мінімальна або одностороння	Активна, двостороння
Технології	HTML, CSS, трохи JS	HTML, CSS, JS + логіка (React, Node.js, API)
Збереження даних	Не обов’язково	Бувають
Авторизація	Рідко	Часто

Значення онлайн-безпеки стало першочерговим із широкою інтеграцією Інтернету в різні сфери суспільства, що охоплює як бізнес-операції, так і повсякденну діяльність людей. Невід'ємним компонентом онлайн-безпеки є захист даних від несанкціонованого доступу. Переважним методом, що використовується

для цієї мети, є використання паролів у процесі онлайн-автентифікації. Пароль, що являє собою конфіденційний рядок символів, відомий виключно користувачеві, служить ключовим елементом у механізмі доступу. Хешований код цього пароля надійно зберігається на сервері, відповідальному за надання доступу до пов'язаних даних.

Коли користувач шукає доступ до даних, він вводить пароль разом з додатковою ідентифікаційною інформацією, такою як ім'я користувача або електронна пошта. Згодом обчислюється шифр дайджесту повідомлення (хеш) пароля, і отриманий хешований код передається на сервер. Потім сервер порівнює цей хеш-код з раніше збереженим хеш-кодом у своїй базі даних. Якщо збіг виявлено, користувач перевіряється як законний заявник своєї особи, що призводить до надання доступу. Натомість, якщо збігу не знайдено, доступ негайно відмовляється.

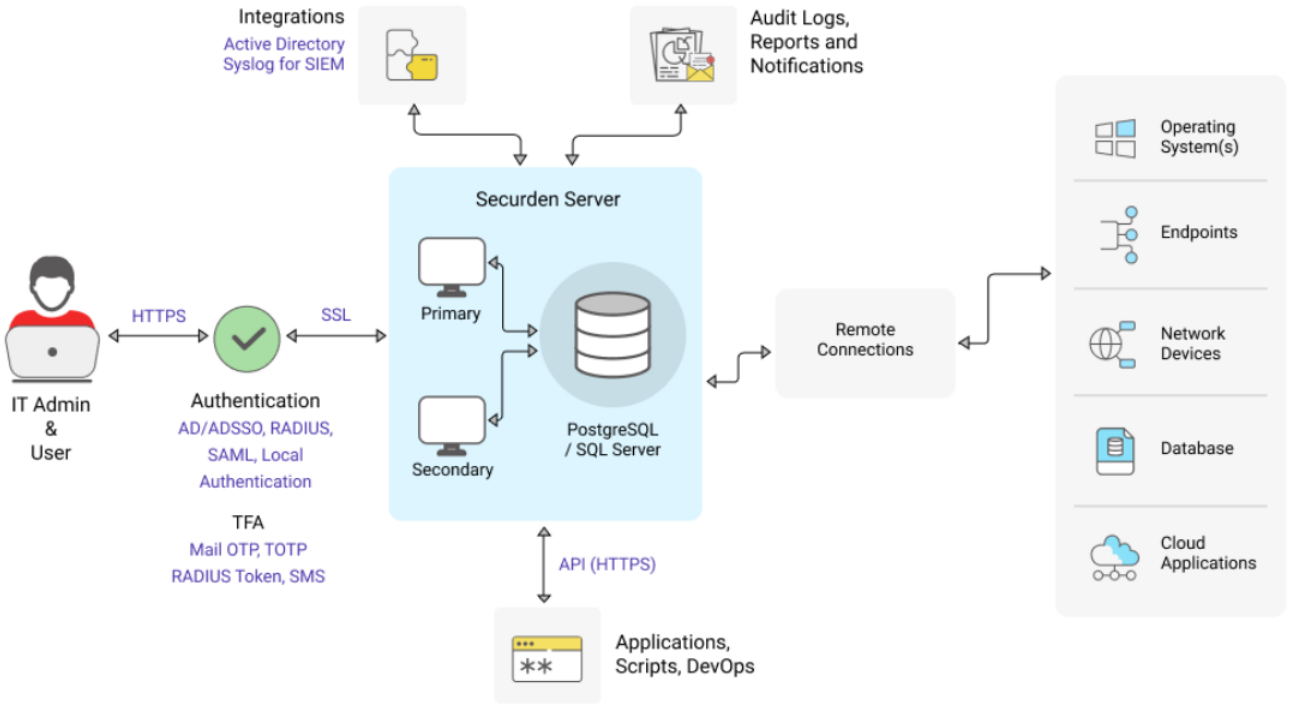


Рисунок 1.10. Архітектура системи

Ідентифікація (від латинського identifico - ототожнювати): Надання суб’єктам та об’єктам ідентифікатора та/або зіставлення цього ідентифікатора з переліком вже наявних. Як приклад, звернення до людини за іменем та по-батькові - це ідентифікація.

Автентифікація(від грецького: αυθεντικός; справжній або дійсний):

Підтвердження справжності чогось чи когось. Наприклад, показ паспорта - це підтвердження автентичності імені та по-батькові, яке було зазначено.

Методи автентифікації, у веб-додатках, особливо корпоративного призначення, надійна автентифікація користувачів є базовою умовою інформаційної безпеки. Зважаючи на це, системи контролю надійності паролів мають безпосередній вплив на загальну ефективність автентифікації.

Авторизація є функцією, яка визначає права доступу до ресурсів та контролює цей доступ.

Схема нижче демонструє типовий ланцюжок взаємодії між користувачем та веб-додатком під час автентифікації, із включенням етапу оцінки надійності пароля на клієнтській стороні:

Рисунок 1.11. Архітектура процесу автентифікації

Авторизація - це не те саме, що ідентифікація та автентифікація: ідентифікація–це спосіб, яким людина представляється системі; автентифікація - це встановлення відповідності особи, яка пред'явила себе системі, заявленому нею ідентифікатору; а авторизація – це надання цій особі можливостей відповідно до наданих права бо ж перевірка наявності прав під час спроби здійснення будь-якої дії. Прикладом авторизації можуть слугувати ліцензії на певну діяльність.

У межах даного проєкту, що реалізує клієнтську перевірку паролів без збереження облікових даних на сервері, критично важливо розуміти, як побудовано стандартні протоколи автентифікації та які їх слабкі місця. Це дає змогу ефективніше інтегрувати механізм попереднього оцінювання надійності паролів на етапі їх створення.

Схема нижче демонструє типовий ланцюжок взаємодії між користувачем та веб-додатком під час автентифікації, із включенням етапу оцінки надійності пароля на клієнтській стороні:

Рисунок 1.12. Етапи доступу користувача

На рисунку зображено послідовність етапів доступу користувача до захищених ресурсів: ідентифікація, автентифікація та авторизація. Першим кроком є ідентифікація - користувач повідомляє систему, ким він є, наприклад, вводячи логін або адресу електронної пошти. Наступний етап - автентифікація, під час якої система перевіряє достовірність введених облікових даних, зазвичай за допомогою пароля або іншого фактора. Завершальним етапом є авторизація, коли система

визначає рівень доступу користувача до функцій або ресурсів відповідно до його ролі чи прав. Ця структура забезпечує контрольований і безпечний процес взаємодії з системою.

Роль пароля в автентифікації у традиційній схемі автентифікації основним засобом підтвердження особи користувача є пароль, що поєднується з логіном або електронною адресою. Незважаючи на розвиток альтернативних методів (одноразові коди, біометрія, WebAuthn), паролі лишаються стандартом де-факто в більшості систем.

Проте використання слабких, повторно застосованих або зламаних паролів є головною причиною зламів акаунтів. Саме тому доцільно впроваджувати механізми перевірки якості пароля до його надсилання на сервер — на стороні клієнта, як у даному проєкті.

Таблиця 1.6. Порівняння основних механізмів автентифікації

Механізм	Принцип роботи	Переваги	Недоліки
HTML-форма	POST-запит з логіном і паролем	Гнучкість, контроль UI	Вимагає захищеного каналу
HTTP Basic / Digest	Пароль передається в заголовку	Простота, підтримка в браузерах	Вразливість без HTTPS, неможливість logout
Token-based (JWT)	Токен у cookie або заголовку	Безсерверна перевірка доступу	Вимагає захисту токена
OAuth / OpenID	Делегована автентифікація	Безпечний доступ через провайдерів	Складність впровадження

Форму автентифікації з перевіркою пароля на клієнті та сесійну авторизацію варто застосовувати, щоб зменшити навантаження на сервер і підвищити UX.

Уразливості при реалізації автентифікації:

- Використання слабких паролів, що не проходять перевірку на складність;
- Використання вкрадених паролів, наявних у відкритих зливах (тут важлива інтеграція з базами типу HaveIBeenPwned);

- CSRF та XSS-атаки, що перехоплюють токени або форми;
- Відсутність багатофакторної автентифікації;
- Використання HTTP замість HTTPS, що дозволяє перехоплювати паролі;
- Зберігання паролів у відкритому вигляді на сервері.

Саме тому одним з основних напрямів захисту є на етапі створення пароля попередити користувача про те, що пароль є слабким або скомпрометованим.

Двофакторна автентифікація - це спосіб забезпечити додатковий рівень безпеки при доступі до облікових записів. Для цього потрібно не тільки ввести пароль для входу, але й код, який має тільки авторизований користувач. Наприклад: якщо неавторизований користувач спробує увійти, використовуючи ваше ім'я користувача та пароль, йому знадобиться додатковий код, який є тільки у вас. Цей код може бути надісланий на ваш номер телефону, або це може бути одноразовий пароль у додатку або на окремому пристрої, який змінює код через короткі проміжки часу. Цей метод входу значно зменшує ризик доступу до будь-якого облікового запису з невідомого пристрою або користувачем із зловмисними намірами.

Значення клієнтської перевірки надійності паролів. У рамках реалізації дипломного проєкту була створена клієнтська система контролю надійності пароля, яка:

- Перевіряє довжину, складність, відсутність шаблонів (типу qwerty123);
- Аналізує на предмет збігів зі словниками та витоками;
- Надає зворотний зв'язок і рекомендації;
- Вбудована у форму реєстрації/зміни пароля;
- Працює повністю на клієнтській стороні — без передачі пароля на сервер до відправки.

Це дозволяє:

- знизити кількість слабких паролів у системі;
- зменшити ризик компрометації облікових записів;
- підвищити загальний рівень інформаційної гігієни користувачів.

OWASP Top 10 - визнана світовою методологією оцінки вразливостей веб-додатків у всьому світі і відображає сучасні тренди безпеки веб-додатків.

Проєкт OWASP (Open Worldwide Application Security Project) щорічно публікує рейтинг найбільш критичних вразливостей у веб-додатках. У контексті систем автентифікації та паролів особливо актуальні кілька позицій з OWASP Top 10.

Таблиця 1.7. Рейтинг

OWASP Top 10 (релевантне)	
A01: Broken Access Control	Критична
A02: Cryptographic Failures	Висока
A03: Injection	Середня
A04: Insecure Design	Висока
A05: Security Misconfig	Середня
A06: Vulnerable Components	Середня
A07: Identification & Auth	Критична
A08: Software Integrity	Середня
A09: Logging & Monitoring	Висока
A10: SSRF	Середня

A07: Broken Identification and Authentication - найважливіший пункт у контексті паролів. Основні вразливості:

- Прийняття занадто простих паролів (на кшталт 123456, password);
- Відсутність контролю кількості спроб входу (можливість brute-force атак);
- Невикористання двофакторної автентифікації;
- Недостатній захист токенів автентифікації;
- Неправильне управління життєвим циклом сесій (наприклад, сесії не завершуються автоматично).

Моя система допомагає знизити ризики OWASP A07:

- Перевірка пароля на складність і наявність у базах злитих паролів (зменшує ризик компрометації);
- Клієнтська валідація дозволяє не передавати слабкий пароль на сервер — зменшення ризику атаки через реєстрацію;
- Поради користувачу щодо довжини, спеціальних символів і унікальності пароля;
- Можливість інтеграції генератора надійних паролів.

2 ЕКОНОМІЧНИЙ РОЗДІЛ

В дипломному проекті створена система контролю надійності паролів для корпоративного сайту

У межах дипломного проєкту було розроблено вебзастосунок «SecurePassCheck», призначений для оцінки надійності паролів, що вводяться користувачами корпоративного сайту. Система забезпечує перевірку паролів на відповідність критеріям безпеки та надає рекомендації щодо їх покращення. Додатково реалізовано функціонал безпечної генерації нових паролів.

Застосунок є одночасно інформативним, інтерактивним та конфіденційним, що є ключовою особливістю цього типу продукту. На відміну від типових рішень, які передають дані на сервер для обробки, у запропонованій системі всі операції з паролями здійснюються виключно на стороні клієнта, без збереження або передачі введеної інформації, що гарантує високий рівень приватності та відповідність сучасним вимогам інформаційної безпеки (зокрема - принципу «zero knowledge»). Сайт складається з двох основних логічних частин, клієнтська частина до якої входить JavaScript-фреймворку React (версія 18) у зв'язці з TypeScript, Tailwind CSS для стилізації, Framer Motion для анімацій та Wouter для маршрутизації та серверна частина побудована на базі Node.js + Express із підтримкою TypeScript. Реалізовані функції перевірки, генерації, візуалізації сили паролю. Основне призначення сервера — обслуговування клієнтських запитів, деплой статичних ресурсів та забезпечення майбутнього масштабування

При оцінці ефективності створюваного веб-додатку виходимо з того, що він є не комерційним продуктом, тому розраховуємо загальні витрати на розробку.

Загальні витрати ($В_з$) на створення веб-додатку складаються з декількох параметрів:

$$В_з = В_р + В_в + В_е \quad (2.1)$$

де $В_р$ – витрати на розробку сайту;

$В_в$ – витрати на впровадження сайту;

$В_е$ – витрати на експлуатацію сайту;

Витрати на розробку сайту ($В_р$) є одноразовими та складаються з вартості наступних видів робіт зі створення сайту :

1. Постановка задачі проектування;
2. Аналіз технічного завдання та пошук літератури;
3. Дослідження методів зламу паролів та існуючих підходів до оцінки їх надійності;
4. Аналіз вимог безпеки до паролів відповідно до сучасних стандартів;
5. Обґрунтування вибору методів, алгоритмів та технологій реалізації;
6. Розробка структури клієнтської частини та архітектури проєкту;
7. Реалізація алгоритму аналізу та генерації паролів;
8. Інтеграція з API для перевірки компрометації паролів;
9. Розробка користувацького інтерфейсу;
10. Реалізація серверної частини (Express.js) для деплою та обслуговування фронтенду;
11. Тестування системи та аналіз результатів.

Для визначення витрат на розробку веб-додатку ($В_р$) розраховуємо оплату праці виконавців, безпосередньо притягнених до її виконання. Для реалізації проекту Web-системи використовуються наступні спеціалісти: Розробник ПЗ; Системний архітектор; Frontend-розробник; Backend-розробник; QA-інженер / Розробник; Full-stack розробник.

Для визначення трудомісткості розробки веб-додатку ($В_р$) складено план-графік по розробці веб-додатку і тривалості виконання робіт. Розподіл робіт по етапах і видах виконавців наведено в таблиці 2.1.

Таблиця 2.1. План-графік по розробці Web- додатку

№	Назва етапу	Час виконання (годин)	Посада виконавця
1	Аналіз вимог і складання технічного завдання	10	Розробник ПЗ
2	Проєктування архітектури та вибір технологій	8	Системний архітектор
3	Реалізація клієнтської частини (UI, логіка)	25	Frontend-розробник

Продовження таблиці 2.1

4	Реалізація серверної частини (API, обробка даних)	15	Backend-розробник
5	Тестування, налагодження, усунення помилок	10	QA-інженер / Розробник
6	Деплой, публікація, підготовка документації	7	Full-stack розробник
ВСЬОГО:		75 годин	

Розрахунок трудомісткості здійснений в наступній послідовності:

Складений перелік всіх етапів і видів робіт, які необхідно виконати в ході даної розробки. Після узгодження з керівником проекту допущено виключення, доповнення, об'єднання окремих етапів і видів робіт;

2. По кожному виду робіт визначений кваліфікаційний рівень виконавців. В разі виконання однієї роботи виконавцями різної кваліфікації, робота розподілена на ряд паралельних конкретних робіт для кожної категорії виконавця.

В умовах відсутності нормативної бази тривалість виконання окремих робіт розраховуємо на основі вірогідності на основі вірогідних оцінок робіт, що задаються виконавцями.

Розмір заробітної плати розраховуємо виходячи з чисельності різних категорій виконавців, трудомісткості, що витрачається ними на виконання різних видів робіт, а також їх середньої заробітної плати (ставки) за годину (або один робочий день).

При визначенні вартості виконуваних робіт орієнтуємося на ціни, представлені на сайтах фірм, що спеціалізуються в сфері створення та модернізації web-ресурсів .

Таблиця 2.2. Витрати на заробітну плату

№	Персонал	Етапи розробки	Кількість робочих годин (або днів)	Погодинна ставка (або денна ставка), грн.	Заробітна плата, грн.
1	UI/UX-дизайнер	Розробка макетів, кольорової схеми, типографіки	12	220	2 640

Продовження таблиці 2.2

2	Frontend-розробник	Розробка інтерфейсу та логіки клієнта	25	200	5 000
3	Backend-розробник	Створення серверної частини	15	230	3 450
4	Системний архітектор	Вибір технологій, проєктування архітектури	8	250	2 000
				ВСЬОГО:	$В_{зп}$ = 13 090

До складу витрат на оплату праці також включаються податки, збори і інші обов'язкові платежі, встановлені системою оподаткування що діє. Розмір єдиного соціального внеску складає 22% від заробітної плати, розраховується за наступною формулою:

$$В_{єсв} = В_{зп} \text{ x } 0,22 \quad (2.2)$$

$$13\ 090 \text{ x } 0,22 = 2\ 879,8 \text{ грн}$$

Загальні витрати ($В_р$) на розробку веб-додатку розраховуємо як сума витрат на заробітну плату праці персоналу ($В_{зп}$) та єдиного соціального внеску ($В_{єсв}$):

$$В_р = В_{зп} + В_{єсв} \quad (2.3)$$

$$13\ 090 + 2\ 879,8 = 15\ 969,8 \text{ грн}$$

Витрати на впровадження веб-додатку ($В_в$) складаються з двох складових :

- витрати на реєстрацію доменного імені на 1 рік ($В_{в1}$) - .app(620,59 грн)
- витрати на реєстрацію в пошукових системах ($В_{в2}$), Google використовую бескоштовно.

$$В_в = В_{в1} + В_{в2} = 620,59 \quad (2.4)$$

Витрати на експлуатацію веб-додатку (Ве) включають вартість робіт з підтримки додатку в робочому стані і вартість послуг по продовженню доменного імені на 1 рік.

Підтримка додатку в робочому стані може здійснювати сама організація. Для певних робіт з цього переліку може використовуватися обслуговуючий персонал

(адміністратор web- додатку). У таблиці 2.3 визначаються постійні витрати як сума витрат на впровадження та експлуатацію додатку протягом року.

Таблиця 2.3. Постійні витрати

№	Стаття витрат	Вартість за рік, грн.
1	Оплата хостингу (Vercel, Replit тощо)	2 400
2	Підписка на дизайн-інструменти (Figma Pro)	1 200
3	Підтримка та обслуговування додатку	3 000
	Всього:	$В_{пост}$ = 6 600 грн

Загальні витрати ($В_з$) на розробку, впровадження та експлуатацію веб-додатку розраховуємо за наступною формулою:

$$В_з = В_р + (В_в + В_е) \quad (2.5)$$

$$15969,8 + (620,59 + 6600) = 23190,39 \text{ грн}$$

З техніко-економічної точки зору, додаток має низку переваг:

- Висока масштабованість: завдяки використанню сучасних вебтехнологій (React, Vite, Express) система легко адаптується до корпоративного середовища різного рівня.
- Низькі витрати на інфраструктуру: за рахунок відсутності потреби в постійному серверному обчисленні та збереженні даних, підтримка додатку в продакшн-середовищі має мінімальну вартість.
- Безпечність реалізації: відсутність передачі паролів на сервер повністю виключає ризики витоку конфіденційної інформації.
- Швидка розробка і впровадження: використання модульного підходу, Vite-середовища для швидкої збірки і хостингу через Vercel/Replit значно скорочує час розгортання.

Таким чином, розроблений застосунок поєднує функціональність, безпечність, швидкість та доступність реалізації, що робить його ефективним рішенням для інтеграції в корпоративну інфраструктуру без значних економічних витрат.

3 РОЗДІЛ ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

Розробка програмного забезпечення, незважаючи на її інтелектуальний характер і відсутність безпосередньої фізичної небезпеки, потребує серйозної уваги до питань охорони праці. Тривале перебування за комп'ютером, використання електронного обладнання, психоемоційне навантаження, високий рівень зорового й статичного навантаження - усе це створює певні ризики для здоров'я фахівця. Саме тому при виконанні дипломного проєкту необхідно враховувати чинні нормативно-правові акти у сфері охорони праці, техніки безпеки, ергономіки, електробезпеки та пожежної безпеки.

3.1 Санітарно-гігієнічні умови

Основними параметрами комфортного та безпечного середовища є виробниче приміщення, параметри мікроклімату, освітлення, шуму, електро- та пожежобезпеки . Згідно з вимогами санітарних норм, робоче місце має забезпечувати оптимальні умови для психофізіологічного стану працівника.

3.1.1 Освітлення

Воно має бути м'яким, без мерехтіння й відблисків. Рекомендується встановлення фільтрів або матових плівок на монітор, встановлення світлодіодне освітлення нейтрального спектра, що мінімізує втому очей. Використовувати антиблікові екрани або матове покриття монітора.

Використовується природне та штучне освітлення, Необхідно розміщувати джерела штучного світла збоку або ззаду користувача для зменшення навантаження на очі.

Освітленість робочої зони: не менше 300 лк при штучному освітленні.

3.1.2 Шум

Під час виконання завдань, що вимагають зосередженості, рівень шуму в приміщенні не повинен перевищувати 50 дБ. Щоб зменшити вплив шуму та вібрацій, обладнання встановлюють на спеціальні амортизуючі підкладки. У випадках, коли джерелом шуму є стіни, їх обробляють звукоізоляційними

матеріалами.

3.1.3 Мікроклімат

Основними параметрами комфортного та безпечного повітряного середовища є:

- Температурний режим: 20–24 °C;
- Рівень відносної вологості повітря: 40–60 %;
- Швидкість руху повітря: не більше 0,1 м/с у холодний період;
- Рівень шуму: не повинен перевищувати 50 дБ.

Робоче приміщення обладнано природною та штучною вентиляцією з регулярним провітрюванням.

3.1.4 Організація робочого місця програміста

Робоче місце програміста, як правило, розташовується в офісному або домашньому середовищі. Умови праці повинні відповідати вимогам нормативних документів, зокрема ДСанПіН 3.3.2.007-98 та ДСТУ EN 29241-3.

Робоче місце оснащене ергономічним офісним кріслом із регульованою висотою та спинкою, що підтримує хребет у фізіологічно правильному положенні; стіл підібрано відповідно до зросту користувача, з урахуванням глибини посадки клавіатури та екрану; монітор розташовано на оптимальній висоті — верхня третина екрана на рівні очей або трохи нижче, на відстані 50–70 см від очей користувача; установлено підставку для ніг, щоб зменшити навантаження на ноги та хребет**.** Дисплей повинен бути нахилений під кутом 10–20°, а центр екрану розміщений на 15–20 см нижче рівня очей.

З метою профілактики порушень опорно-рухового апарату, зору та нервової системи, розробнику рекомендовано дотримуватися режиму праці з перервами: 5–10 хвилин відпочинку через кожні 50 хвилин роботи за комп’ютером. Під час цих перерв доцільно виконувати гімнастику для очей, легкі фізичні вправи та змінювати позу.

Організація режиму праці та відпочинку:

- Дотримано режиму праці за ПК згідно з рекомендаціями МОЗ: перерва 10 хвилин щогодини.

- У перервах виконуються вправи для очей, шиї та хребта, що попереджають перевтому та остеохондроз.
- Використовується таймер або спеціальне ПЗ, яке автоматично нагадує про необхідність зробити перерву.

Уся організація робочого місця має відповідати ергономічним стандартам, з урахуванням взаємного розташування обладнання для забезпечення комфорту та продуктивності працівника. На рис. 3.1 представлено робоче місце і робоча поза користувача комп'ютера.

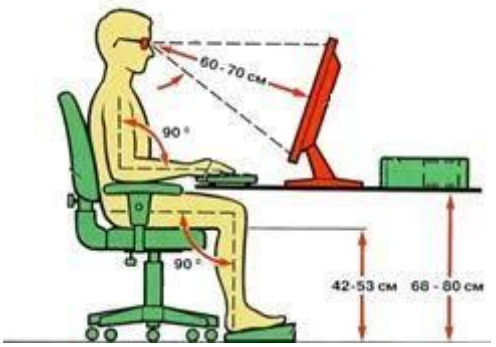


Рисунок 3.1. Робоче місце і робоча поза користувача комп'ютера

3.1.5 Електробезпека

Комп’ютерна техніка належить до електроустановок напругою до 1000 В і відноситься до II класу електобезпеки. Робоче місце програміста має бути обладнане справною електромережею з обов’язковим заземленням. Кабелі, розетки та електричні з’єднання мають бути надійно ізольовані. Усі прилади повинні мати сертифікати відповідності й технічну документацію.

Категорично забороняється самостійний ремонт електрообладнання, використання пошкоджених дротів, подовжувачів без заземлення або підключення пристроїв з видимими дефектами. Рекомендується використання автоматичних вимикачів та пристроїв захисного відключення (УЗО), які унеможливлюють ураження електричним струмом у разі замикання або перенапруги.

Особливу увагу слід приділити використанню багатопортових подовжувачів: їхнє перевантаження може стати причиною короткого замикання або перегріву.

За для дотримання безпеки необхідно, щоб усі пристрої підключено через перевірені мережеві фільтри з захистом від перенапруги. Комп’ютерна техніка

заземлена, електромережі відповідають нормам ПУЕ (Правил улаштування електроустановок). Заборонено самостійне втручання в електромережу — всі несправності усуваються тільки кваліфікованими спеціалістами. Пристрої регулярно проходять візуальний контроль на предмет пошкоджень кабелів, корпусів, перегріву.

3.2 Психоемоційна безпека

Робота програміста часто пов’язана з інтенсивним інтелектуальним навантаженням, дедлайнами та тривалим перебуванням у монотонному середовищі. Для запобігання емоційному вигоранню та збереження психічного здоров’я необхідно дотримуватись гігієни сну, уникати багатогодинної безперервної роботи та забезпечити зміну діяльності впродовж дня.

Крім того, важливо враховувати режим харчування, тривалість робочого дня (не більше 8 годин) та уникнення нічної праці, що призводить до порушень біоритмів і зниження продуктивності.

Для забезпечення психоемоційної стабільністі необхідно уникникати перевтоми використовуються методи тайм-менеджменту (Pomodoro, Work-Break Cycle); дотримуватись чіткого графіку роботи, передбачено час для фізичної активності, відпочинку, повноцінного сну; використовувати програми для зниження стресу (музика, нейрошум, цифровий детокс).

3.3 Пожежна безпека

Робота з комп’ютерною технікою передбачає потенційні ризики займання внаслідок перегріву, короткого замикання або несправності обладнання. Тому робоче місце має відповідати вимогам пожежної безпеки, зокрема:

- повинні використовуватись тільки сертифіковані джерела живлення;
- забороняється накривати вентиляційні отвори в системному блоці чи ноутбуці;
- не допускається розміщення легкозаймистих матеріалів поблизу електропристроїв;

- наявність первинних засобів пожежогасіння (вогнегасника типу ВВК-1, ВП-2) у доступному місці є обов’язковою.

Також слід дотримуватися чіткого плану евакуації у разі виникнення пожежі, знати місце розташування аварійних виходів і протипожежного інвентарю.

- На робочому місці розміщено вогнегасник порошкового типу (ВП-2), призначений для гасіння електрообладнання.
- Створено схему евакуації, визначено шляхи виходу з приміщення у разі надзвичайної ситуації.
- Проводиться перевірка стану електричних з’єднань та справності обладнання не рідше одного разу на місяць.
- Заборонено використання несправних розеток, кустарних подовжувачів, зберігання легкозаймистих матеріалів біля техніки.

Рисунок 3.2. Первинні засоби пожежогасіння

Варто пам’ятати, що інформаційна та цифрова безпека теж не мало важливі, використання захищенного середовища розробки (Replit, Vercel), забезпечують контрольований доступ і резервне копіювання. Застосування двофакторної автентифікації для доступу до облікових записів хостингу та середовища розробки. Впровадження шифрування даних на рівні локального збереження проєкту.

ВИСНОВКИ

У результаті виконання дипломної роботи було досягнуто поставлену мету - створено безпечний клієнтський веб-додаток для перевірки надійності паролів у реальному часі без передачі або збереження персональних даних. Розроблена система орієнтована на корпоративне використання та відповідає сучасним вимогам до захисту облікових даних.

У ході дослідження було проаналізовано типові загрози, пов’язані зі слабкими паролями, визначено помилки, яких найчастіше припускаються користувачі, а також вивчено вимоги стандартів безпеки, зокрема OWASP. Це дало змогу сформулювати чіткі критерії оцінки надійності паролів.

Технічна реалізація проєкту включала створення клієнтської архітектури з повною обробкою введених даних у браузері без участі серверної логіки, що забезпечує максимальну конфіденційність. Було розроблено кастомний алгоритм оцінки складності пароля з урахуванням довжини, різноманітності символів, поширених шаблонів і потенційних вразливостей. Крім того, реалізовано генератор безпечних паролів із можливістю налаштування параметрів.

Інтерфейс додатку створено з використанням технологій React, TypeScript, Tailwind CSS і Framer Motion, що забезпечило сучасний вигляд, інтерактивність та адаптивність. Серверна частина, реалізована на Node.js з Express, використовується виключно для хостингу, не взаємодіючи з паролями користувача. Сайт було розгорнуто на платформі Vercel для забезпечення стабільного доступу.

Проведене тестування продемонструвало стабільну роботу застосунку на різних пристроях і підтвердило відповідність функціональним і безпековим вимогам. Результати показують, що застосунок є ефективним інструментом підвищення цифрової гігієни та може бути інтегрований у внутрішню політику кібербезпеки компаній.

Таким чином, запропонована система демонструє практичну цінність і відповідає актуальним викликам у сфері інформаційної безпеки.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Крамаренко, С. В.; Ковальчук, Ю. П. Основи безпеки інформаційних систем. – Київ: КНУ імені Тараса Шевченка, 2020. – 248 с.
2. ДСТУ ISO/IEC 27002:2015. Інформаційні технології. Методи захисту. Практичні правила управління інформаційною безпекою. – Київ: ДП «УкрНДНЦ», 2015. – 96 с.
3. Краснокутський, Р. Р. Програмний модуль аналізу паролів для захисту від соціотехнічних атак. – Київ: НАУ, 2020. – 66 с.
4. Dell'Amico, M., Michiardi, P. ; Password strength: An empirical analysis, 2010. – 11 p.
5. Chanda, K.; Password Security: An Analysis of Password Strengths and Vulnerabilities, 2016. – 25 p.
6. Hu, G. On Password Strength: A Survey and Analysis, 2010. – 166 p.
7. Houshmand, S.; Yazdi, S. Analyzing Password Strength & Efficient Password Cracking. – International Journal of Computer Applications, 2011. – 14 p.
8. Password Storage - OWASP Cheat Sheet Series – 2023. [Веб-сайт]. URL: https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html – Дата звернення: 18.05.2025.
9. NIST. Спеціальна публікація 800-63B. – 2023. [Веб-сайт]. URL: https://pages.nist.gov/800-63-3/sp800-63b.html – Дата звернення: 18.05.2025.
10. Have I Been Pwned: API Документація – 2023. [Веб-сайт]. URL: https://haveibeenpwned.com/api/v3 – Дата звернення: 18.05.2025.
11. Crum & Forster Insurance. Two-factor authentication. – 2021. [Веб-сайт]. URL: https://www.cfins.com/wp-content/uploads/2021/05/2FA-Instructions.pdf – Дата звернення: 18.05.2025.
12. Що таке хороші паролі? (OWASP) – 2025. [Веб-сайт]. URL: https://owasp.org/www-community/password-special-characters.
13. OWASP Foundation. OWASP Authentication Cheat Sheet. [Веб-сайт]. URL: https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html – Дата звернення: 25.05.2025.

ДОДАТОК А

Вміст файлів з кодом мовою TypeScript проекту веб-додатку

Головний клієнтський код (frontend): client/src/App.tsx + main.tsx

client/src/App.tsx

```
import { Switch, Route } from "wouter";
import { queryClient } from "./lib/queryClient";
import { QueryClientProvider } from "@tanstack/react-query";
import { Toaster } from "@/components/ui/toaster";
import { TooltipProvider } from "@/components/ui/tooltip";
import NotFound from "@/pages/not-found";
import Home from "@/pages/home";

function Router() {
  return (
    <Switch>
      <Route path="/" component={Home} />
      <Route component={NotFound} />
    </Switch>
  );
}

function App() {
  return (
    <QueryClientProvider client={queryClient}>
      <TooltipProvider>
        <Toaster />
        <Router />
      </TooltipProvider>
    </QueryClientProvider>
  );
}

export default App;
```

client/src/ main.tsx

```
import { createRoot } from "react-dom/client";
import App from "./App";
import "./index.css";

createRoot(document.getElementById("root")!).render(<App />);
```

Основний алгоритм перевірки паролів: client/src/libs/passwordUtils.ts

```
// Password strength analysis function
export function analyzePassword(password: string) {
  // Check criteria
  const hasLength = password.length >= 8;
  const hasUppercase = /[A-Z]/.test(password);
  const hasLowercase = /[a-z]/.test(password);
  const hasNumber = /[0-9]/.test(password);
  const hasSpecial = /[^A-Za-z0-9]/.test(password);

  // Calculate score
  let score = 0;
  if (password.length > 0) {
```

```
  // Base score for length
  score += Math.min(2, Math.floor(password.length / 4));

  // Add points for variety
  if (hasUppercase) score += 1;
  if (hasLowercase) score += 1;
  if (hasNumber) score += 1;
  if (hasSpecial) score += 1;

  // Bonus for length
  if (password.length >= 12) score += 1;
  if (password.length >= 16) score += 1;
}

// Determine strength text and color
let strengthText, colorClass, feedbackClass, feedback, feedbackDetail;

if (password.length === 0) {
  strengthText = "None";
  colorClass = "strength-none";
  feedbackClass = "bg-gray-50 text-gray-600";
  feedback = "Enter a password to see detailed feedback";
  feedbackDetail = "";
} else if (score < 3) {
  strengthText = "Weak";
  colorClass = "strength-weak";
  feedbackClass = "bg-red-50 text-red-700";
  feedback = "This password is weak and easily guessable.";
  feedbackDetail = hasLength ?
    "Try adding more variety with symbols and numbers." :
    "Make your password at least 8 characters long.";
} else if (score < 5) {
  strengthText = "Fair";
  colorClass = "strength-fair";
  feedbackClass = "bg-amber-50 text-amber-700";
  feedback = "This password provides some security but could be stronger.";

  if (!hasUppercase || !hasLowercase) {
    feedbackDetail = "Add both uppercase and lowercase letters.";
  } else if (!hasNumber) {
    feedbackDetail = "Include at least one number.";
  } else {
    feedbackDetail = "Add a special character for extra security.";
  }
} else if (score < 7) {
  strengthText = "Good";
  colorClass = "strength-good";
  feedbackClass = "bg-yellow-50 text-yellow-700";
  feedback = "This is a good password with mixed character types.";
  feedbackDetail = "For even better security, consider making it longer.";
} else {
  strengthText = "Strong";
  colorClass = "strength-strong";
  feedbackClass = "bg-green-50 text-green-700";
  feedback = "Excellent! This is a strong, secure password.";
  feedbackDetail = "Remember to use unique passwords for different accounts.";
}

return {
  score,
  strengthText,
  hasLength,
```

```
    hasUppercase,
    hasLowercase,
    hasNumber,
    hasSpecial,
    colorClass,
    feedbackClass,
    feedback,
    feedbackDetail
  };
}

// Password generation function
interface PasswordOptions {
  length: number;
  useUppercase: boolean;
  useLowercase: boolean;
  useNumbers: boolean;
  useSymbols: boolean;
  excludeSimilar: boolean;
}

export function generatePassword(options: PasswordOptions): string {
  // Character sets
  const lowerChars = options.excludeSimilar ? 'abcdefghjkmnpqrstuvwxyz' :
'abcdefghijklmnopqrstuvwxyz';
  const upperChars = options.excludeSimilar ? 'ABCDEFGHJKMNPQRSTUVWXYZ' :
'ABCDEFGHIJKLMNOPQRSTUVWXYZ';
  const numberChars = options.excludeSimilar ? '23456789' : '0123456789';
  const symbolChars = '!@#$%^&*()_+~`|}{[]:;?><,./-=';

  // Build character pool based on options
  let chars = '';
  if (options.useLowercase) chars += lowerChars;
  if (options.useUppercase) chars += upperChars;
  if (options.useNumbers) chars += numberChars;
  if (options.useSymbols) chars += symbolChars;

  // Ensure we have at least one character type
  if (chars.length === 0) {
    chars = lowerChars;
  }

  // Generate initial password
  let password = '';

  // Ensure at least one character from each selected set
  if (options.useLowercase) {
    password += lowerChars.charAt(Math.floor(Math.random() * lowerChars.length));
  }
  if (options.useUppercase) {
    password += upperChars.charAt(Math.floor(Math.random() * upperChars.length));
  }
  if (options.useNumbers) {
    password += numberChars.charAt(Math.floor(Math.random() * numberChars.length));
  }
  if (options.useSymbols) {
    password += symbolChars.charAt(Math.floor(Math.random() * symbolChars.length));
  }

  // Fill remaining length with random characters
  while (password.length < options.length) {
    const randomChar = chars.charAt(Math.floor(Math.random() * chars.length));
```

```
    password += randomChar;
  }

  // Shuffle the password to ensure randomness
  return shuffleString(password);
}

// Utility function to randomly shuffle a string
function shuffleString(str: string): string {
  const array = str.split('');
  for (let i = array.length - 1; i > 0; i--) {
    const j = Math.floor(Math.random() * (i + 1));
    [array[i], array[j]] = [array[j], array[i]];
  }
  return array.join('');
}
```

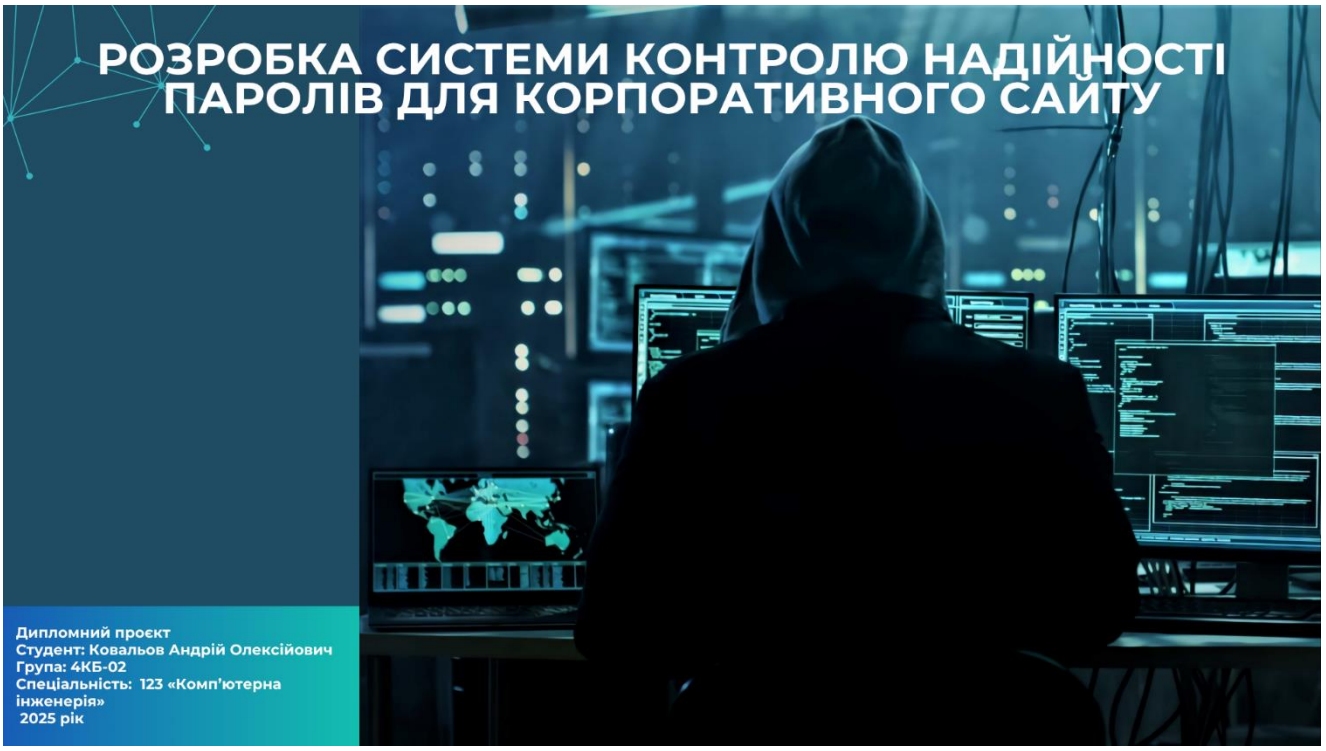
Головний серверний код (backend): server/index.ts

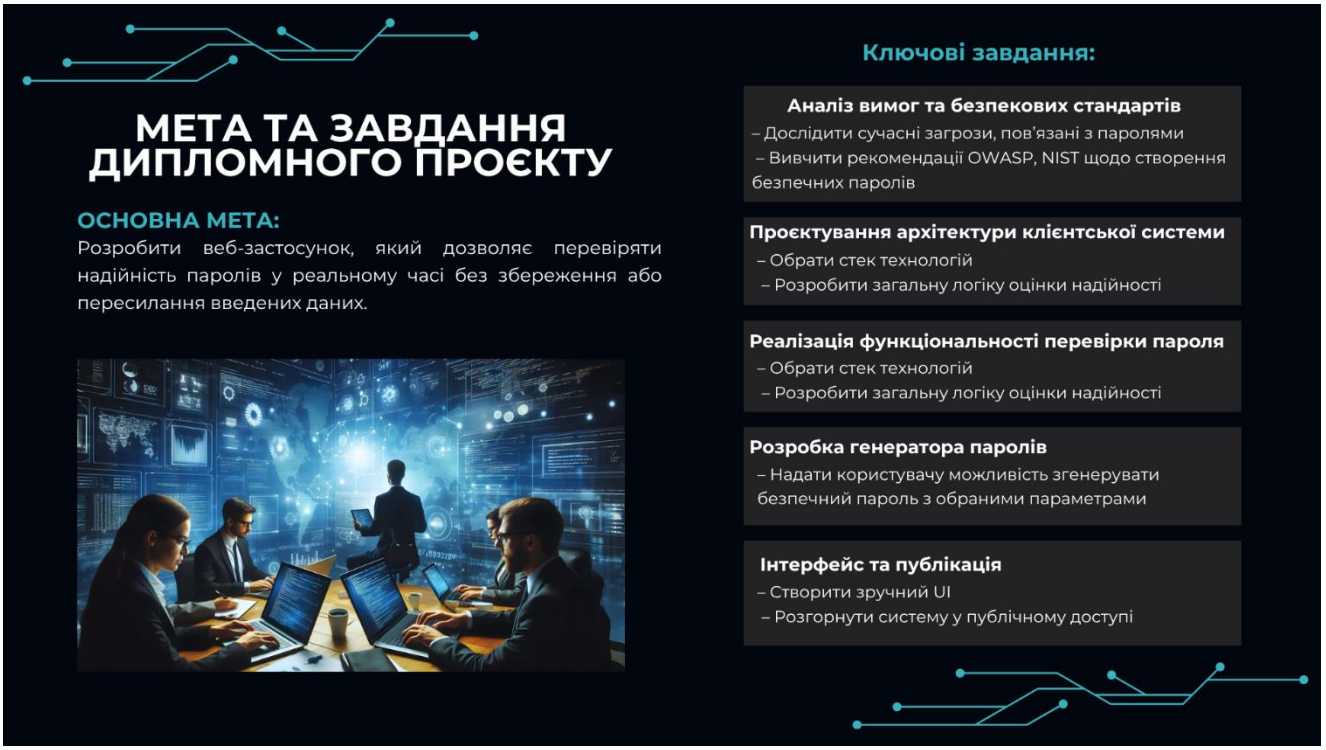
```
import express, { type Request, Response, NextFunction } from "express";
import { registerRoutes } from "./routes";
import { setupVite, serveStatic, log } from "./vite";

const app = express();
app.use(express.json());
app.use(express.urlencoded({ extended: false }));

app.use((req, res, next) => {
  const start = Date.now();
  const path = req.path;
  let capturedJsonResponse: Record<string, any> | undefined = undefined;

  const originalResJson = res.json;
  res.json = function (bodyJson, ...args) {
    capturedJsonResponse = bodyJson;
    return originalResJson.apply(res, [bodyJson, ...args]);
  };

  res.on("finish", () => {
    const duration = Date.now() - start;
    if (path.startsWith("/api")) {
      let logLine = `${req.method} ${path} ${res.statusCode} in ${duration}ms`;
      if (capturedJsonResponse) {
        logLine += ` :: ${JSON.stringify(capturedJsonResponse)}`;
      }

      if (logLine.length > 80) {
        logLine = logLine.slice(0, 79) + "…";
```

```
      }

      log(logLine);
    }
  });

  next();
});

(async () => {
  const server = await registerRoutes(app);

  app.use((err: any, _req: Request, res: Response, _next: NextFunction) => {
    const status = err.status || err.statusCode || 500;
    const message = err.message || "Internal Server Error";

    res.status(status).json({ message });
    throw err;
  });

  // importantly only setup vite in development and after
  // setting up all the other routes so the catch-all route
  // doesn't interfere with the other routes
  if (app.get("env") === "development") {
    await setupVite(app, server);
  } else {
    serveStatic(app);
  }

  // ALWAYS serve the app on port 5000
  // this serves both the API and the client.
  // It is the only port that is not firewalled.
  const port = 5000;
  server.listen({
    port,
    host: "0.0.0.0",
    reusePort: true,
  }, () => {
    log(`serving on port ${port}`);
  });
})();
```

ДОДАТОК Б

Слайди мультимедійної презентації





ЧОМУ ЦЯ ТЕМА Є НАДЗВИЧАЙНО АКТУАЛЬНОЮ?

КЛЮЧОВІ ФАКТИ:

Паролі залишаються основним методом аутентифікації

Більшість систем все ще покладаються на введення паролів.

Слабкі паролі — причина більшості атак

Компрометація облікових записів — один із найпоширеніших векторів атак.

Повторне використання паролів

Користувачі часто використовують однакові або схожі паролі на різних ресурсах.

Недостатня обізнаність користувачів

Не всі розуміють, як створювати складні паролі, або чому це важливо.

Існуючі рішення не завжди зручні або доступні

Більшість перевірок відбуваються на сторонніх серверах або потребують реєстрації.

Надійні паролі — основа цифрової безпеки.
Проста та безпечна перевірка пароля прямо у браузері — актуальне рішення для сучасного користувача.

ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

ПРИКЛАДИ ПОПУЛЯРНИХ СЕРВІСІВ:

LastPass / 1Password / Bitwarden

➤ Менеджери паролів з можливістю зберігання, генерації та автозаповнення
➤ Часто вимагають акаунта, синхронізації, іноді платні

Have I Been Pwned

➤ Онлайн-перевірка, чи потрапив пароль у злиті бази даних
➤ Працює через API, але не дає оцінки сили пароля

Генератори паролів (вбудовані в браузери)

➤ Генерують випадкові паролі, але не пояснюють, наскільки вони безпечні
➤ Без інтерфейсу оцінки або зворотного зв'язку

НЕДОЛІКИ ІСНУЮЧИХ РІШЕНЬ:

- Потребують реєстрації або збереження облікових даних
- Часто працюють на сторонньому сервері (менше приватності)
- Не завжди дають миттєвий візуальний фідбек
- Не пояснюють, чому пароль слабкий

ПІДХІД ПРОЄКТУ:

✓ Без акаунтів
✓ Без передачі пароля на сервер
✓ Оцінка і пояснення на клієнті, у реальному часі

АРХІТЕКТУРА КЛІЄНТСЬКОЇ СИСТЕМИ ПЕРЕВІРКИ ПАРОЛІВ

КОРОТКИЙ ОПИС:

Система працює повністю на клієнтській стороні.
Усі перевірки виконуються без пересилання чи збереження введених паролів.
Сервер використовується лише для розміщення сайту (деплой).

[Користувач]
↓
[Введення пароля]
↓
[Перевірка в браузері]
↓
[Вивід результату]
Сервер тільки для хостингу

ЛОГІКА ВЗАЄМОДІЇ:

1. Користувач відкриває вебсторінку
   ↳ Сайт завантажується з хостингу (Vercel)
2. Введення пароля у форму
   ↳ Запускається JavaScript-алгоритм прямо у браузері
3. Клієнтська перевірка
   ↳ Оцінка складності (довжина, символи, шаблони)
   ↳ Рекомендації та візуальний індикатор
4. Результат показується миттєво
   ↳ Пароль нікуди не зберігається й не передається

ВИКОРИСТАНІ ТЕХНОЛОГІЇ ТА ІНСТРУМЕНТИ

КЛІЄНТСЬКА ЧАСТИНА (FRONTEND)

React 18
➤ JavaScript-бібліотека для побудови інтерфейсу користувача
➤ Дозволяє створювати реактивні, модульні компоненти
➤ Підтримує оновлення без перезавантаження сторінки

TypeScript
➤ Надбудова над JavaScript з підтримкою статичної типізації
➤ Зменшує кількість помилок під час розробки
➤ Підвищує масштабованість і читаємість коду

Tailwind CSS
➤ Утилітарний CSS-фреймворк для швидкого стилювання
➤ Дозволяє писати адаптивний і охайний інтерфейс без перевантаження HTML
➤ Ідеально підходить для кастомного дизайну

Framer Motion
➤ Бібліотека для створення анімацій у React
➤ Додає плавність взаємодії та покращує UX
➤ Використовується для реакції на введення, індикаторів тощо

СЕРВЕРНА ЧАСТИНА (BACKEND ДЛЯ ХОСТИНГУ)

Node.js + Express (мінімальне використання)
➤ Node.js дозволяє створювати легкі веб-сервери на JavaScript
➤ Express — фреймворк для швидкого створення роутів
➤ Використовується лише для базового розгортання (не обробляє паролі!)

ІНФРАСТРУКТУРА

Vercel
➤ Хмарна платформа для хостингу frontend-додатків
➤ Підтримує CI/CD (автоматичне оновлення після змін)
➤ Гарантує швидкий доступ з будь-якої точки світу через HTTPS

Replit (середовище розробки)
➤ Онлайн-редактор коду з підтримкою реального часу
➤ Дозволяє зручно розробляти, тестувати та демонструвати проєкт
➤ Застосовувався на етапі dev (локального тестування)

ВИКОРИСТАНІ ТЕХНОЛОГІЇ ТА ІНСТРУМЕНТИ

Аналіз введеного пароля

Користувач вводить пароль у спеціальне поле, і система автоматично аналізує його надійність у реальному часі.

✓ Оцінка довжини
✓ Типи символів (великі, малі, цифри, спецсимволи)
✓ Виявлення повторюваних шаблонів

Візуальний індикатор надійності

Кольорова шкала показує, наскільки сильний пароль:
Слабкий | Середній | Складний | Надійний

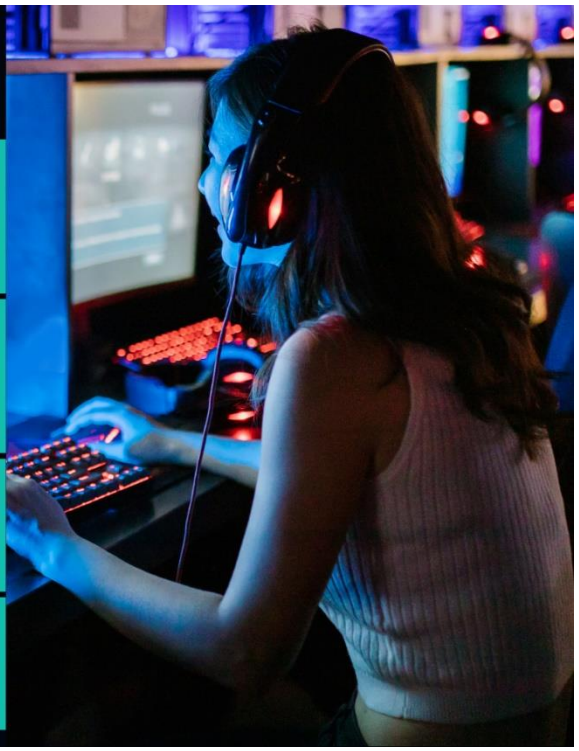
✓ Миттєва реакція
✓ Зворотний зв'язок без перезавантаження

Генератор безпечних паролів

Дає змогу автоматично створити надійний пароль із заданими параметрами:
✓ Обирається довжина
✓ Можна включити/виключити символи
✓ Копіювання одним кліком

Поради для покращення пароля

Після перевірки слабкого пароля користувач отримує конкретні поради:
✓ Додайте спецсимволи
✓ Збільште довжину
✓ Уникайте шаблонів типу "12345"



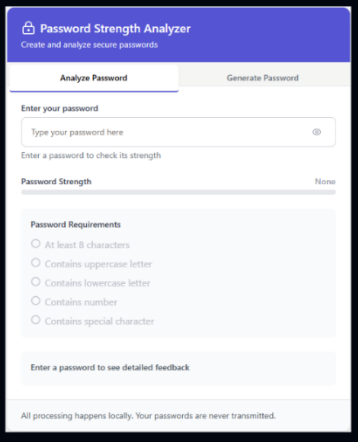
ІНТЕРФЕЙС КОРИСТУВАЧА: ПРОСТОТА ТА ЗРУЧНІСТЬ

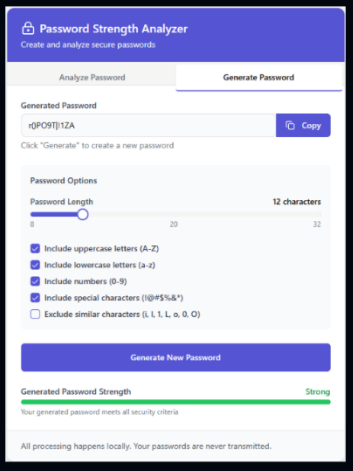
🔑 Поле введення пароля
➤ Миттєва реакція на кожен введений символ
➤ Немає кнопки "перевірити" — все динамічно

📊 Індикатор надійності
➤ Кольорова шкала (від червоного до зеленого)
➤ Текстове повідомлення: "Слабкий", "Середній", "Сильний" "Надійний"

🔄 Генератор паролів
➤ Налаштування: довжина, включення символів
➤ Кнопка "Згенерувати" і "Копіювати"

💡 Поради для покращення
➤ Поява підказок, якщо пароль слабкий
➤ Формат: "Додайте спецсимвол", "Пароль занадто короткий"





ЯК СИСТЕМА ВИЗНАЧАЄ НАДІЙНІСТЬ ПАРОЛЯ?

БАЗОВІ КРИТЕРІЇ ОЦІНКИ:

Довжина пароля
- ➤ Чим більше символів — тим краще
- ➤ Мінімум рекомендовано: 8–12 символів

Різноманітність символів
- ➤ Велика літера (A–Z)
- ➤ Мала літера (a–z)
- ➤ Цифри (0–9)
- ➤ Спецсимволи (!, @, #, %, тощо)

Уникнення шаблонів і повторів
- ➤ Повтори типу: "111", "aaaa"
- ➤ Поширені комбінації: "123456", "qwerty"

Заборона на словникові слова (опційно)
- ➤ Наприклад: "password", "admin", "welcome"

Власна система балів
- ➤ Кожен критерій дає певну кількість балів
- ➤ Порогові значення визначають рівень безпеки

Рівень	Бали	Кольорова зона
Слабкий	0-25%	Червона
Середній	26-50%	Помаранчевий
Складний	51-75%	Жовта
Надійний	76-100%	Зелена

Оцінка = Базові бали + Бонуси за символи – Штрафи за повторення

ВИКОРИСТАНІ ТЕХНОЛОГІЇ ТА ІНСТРУМЕНТИ

Аналіз введеного пароля

Користувач вводить пароль у спеціальне поле, і система автоматично аналізує його надійність у реальному часі.
- ✓ Оцінка довжини
- ✓ Типи символів (великі, малі, цифри, спецсимволи)
- ✓ Виявлення повторюваних шаблонів

Візуальний індикатор надійності

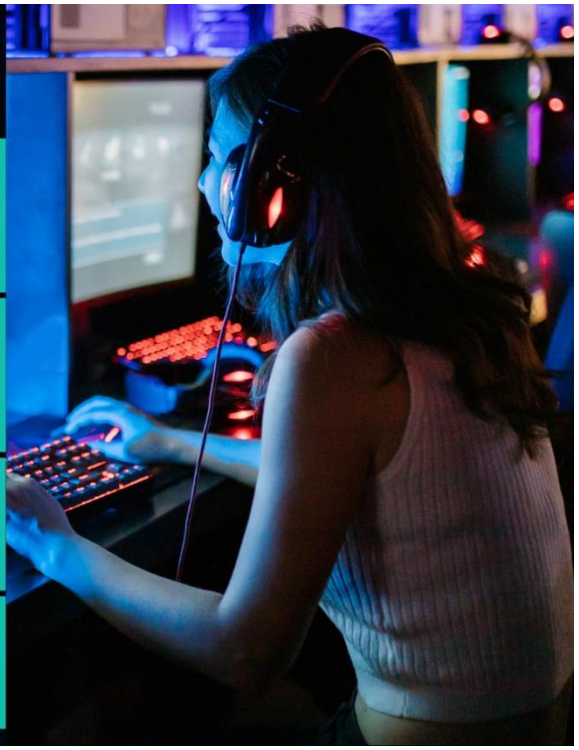
Кольорова шкала показує, наскільки сильний пароль:
Слабкий | Середній | Складний | Надійний
- ✓ Миттєва реакція
- ✓ Зворотний зв'язок без перезавантаження

Генератор безпечних паролів

Дає змогу автоматично створити надійний пароль із заданими параметрами:
- ✓ Обирається довжина
- ✓ Можна включити/виключити символи
- ✓ Копіювання одним кліком

Поради для покращення пароля

Після перевірки слабкого пароля користувач отримує конкретні поради:
- ✓ Додайте спецсимволи
- ✓ Збільште довжину
- ✓ Уникайте шаблонів типу "12345"



ЗАХИСТ ДАНИХ ТА ПРИНЦИПИ БЕЗПЕКИ

КЛЮЧОВІ ПРИНЦИПИ:

Перевірка відбувається повністю на клієнті
➤ Жодна інформація не передається на сервер

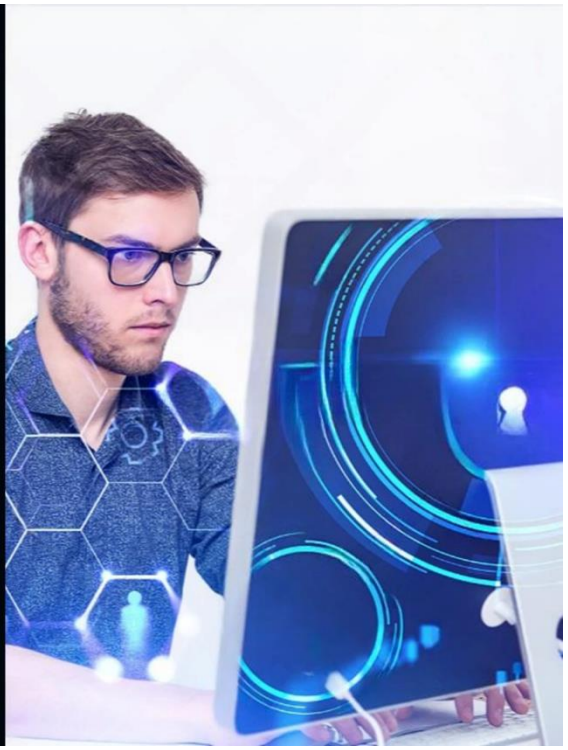
Паролі не зберігаються
➤ Введене значення існує лише в оперативній пам'яті браузера

Відсутність підключення до баз даних
➤ Немає централізованого зберігання або реєстрації

Відкрита логіка оцінки
➤ Користувач розуміє, за якими критеріями формується результат

ОБМЕЖЕННЯ:

- Не перевіряє пароль за базами злитих (offline only)
- Результати можуть відрізнятись від політик конкретної організації



ЯК ПЕРЕВІРЯЛАСЬ СИСТЕМА?

МЕТОДИ ТЕСТУВАННЯ:

Ручне функціональне тестування
➤ Кожен елемент перевірено на працездатність

Кросбраузерна перевірка
➤ Chrome, Firefox, Edge — повна підтримка

Тестування адаптивності
➤ Перевірено на телефонах та планшетах

UX-фідбек від користувачів
➤ Система інтуїтивно зрозуміла без інструкцій

ДЕ І ЯК РОЗМІЩЕНО САЙТ?

ІНФРАСТРУКТУРА:

Хостинг: Vercel
➤ HTTPS-захист
➤ Швидке завантаження з CDN

Dev-середовище: Replit
➤ Онлайн-редактор коду та тестування

Деплой:
➤ Зміни зберігаються й автоматично оновлюють live-версію

Публічне посилання:
https://secure-pass-check.vercel.app

ЩО БУЛО ДОСЯГНУТО?

ОСНОВНІ РЕЗУЛЬТАТИ:

- Створено просту та зручну систему перевірки паролів
- Перевірка виконується безпечним клієнтським методом
- Реалізовано генератор, індикатор та рекомендації
- Проєкт готовий до впровадження в корпоративне середовище

МОЖЛИВОСТІ ПОДАЛЬШОГО РОЗВИТКУ:

- Інтеграція з API злитих паролів (HavelBeenPwned)
- Розширення логіки оцінки з урахуванням поведінки користувача
- Додавання темної/світлої теми, підтримка інших мов

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

РЕЦЕНЗІЯ

на дипломний проект здобувача (здобувачки) освіти
відділення комп'ютерних систем

Ковальова Андрія Олексійовича

(прізвище, ім`я та по батькові)

Спеціальність *123 «Комп'ютерна інженерія»*

Освітня програма *«Безпека комп'ютерних систем і мереж»*

Керівник дипломного проекту (роботи) *Стайкуца Сергій Володимирович*

(прізвище, ім`я та по батькові)

Тема дипломного проекту (роботи) *Розробка системи контролю надійності паролів для корпоративного сайту*

Обсяг розрахунково-пояснювальної записки *70* сторінок

Обсяг графічної (презентаційної) частини *13* аркушів (слайдів)

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) заключення про ступінь відповідності виконаного дипломного проекту завданню

Представлений на рецензію дипломний проект відповідає затвердженій темі та виконаний відповідно технічному завданню. Дипломний проект присвячений темі розробки рішення для контролю надійності паролів корпоративного сайту, складається з пояснювальної записки та мультимедійної презентації.

б) характеристика виконання кожного розділу дипломного проекту

Пояснювальна записка складається з основного розділу (базова інформація, виявлення проблематики, складання технічного завдання, аналіз технологій, розробка рішень), економічного розділу, розділу охорони праці та додатків. Перелічені розділи поетапно охоплюють розробку, виконані докладно та обґрунтовано. Розділ охорони праці містить загальну інформацію та рекомендації щодо організації робочого місця. Економічний розділ проекту містить обчислення вартості розробки програмного рішення.

в) оцінка якості виконання пояснювальної записки та графічної частини дипломного проекту

Графічна частина складається з 15 слайдів мультимедійної презентації, виконаної у програмному продукті MS PowerPoint, які містять ілюстративні схеми, алгоритми, рішення та розрахунки, передбачені технічним завданням. Пояснювальна записка виконана акуратно та у відповідності до норм. Якість виконання графічної частини проекту та пояснювальної записки добра, завдання виконане у повному обсязі.

г) перелік позитивних якостей дипломного проекту *Запропоновано комплексне рішення з оцінки надійності паролів корпоративного сайту.*

Через реалізацію функції перевірки паролів на складність рішення дозволяє суттєво знизити ризики, виявлені в рамках OWASP A07

д) основні недоліки дипломного проекту

Було б доцільним провести аналіз конкурентних рішень, вже представлених на ринку та оцінити поточні рішення в фокусі проблематики. Є деякі порушення при оформленні пояснювальної записки.

Оцінка розрахункової частини *Відмінно*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Відмінно*

Прізвище, ім'я, по батькові рецензента *к.т.н. Рудніченко Микола Дмитрович*

Місце роботи і посада рецензента *Національний університет «Одеська політехніка», доцент кафедри інформаційних технологій*

Підпис: _____

«23» _____ 2025 р.

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

ВІДГУК

керівника на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Ковальова Андрія Олексійовича

(прізвище, ім`я та по батькові)

Спеціальність: *123 "Комп'ютерна інженерія"*

Освітня програма: *«Безпека комп'ютерних систем і мереж»*

Тема дипломного проекту: Розробка системи контролю надійності паролів для корпоративного сайту

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ

а) обсяг і якість виконання проекту (графічного матеріалу і розрахунково-пояснювальної записки) *Дипломний проект виконано відповідно технічному завданню. Пояснювальна записка містить 70 сторінки. У пояснювальній записці розглянуто щодо систем контролю надійності паролів для корпоративних сайтів. Проведено аналіз відкритих джерел інформації, сформульовано проблему та запропоновано рішення. Графічна частина складається з слайдів мультимедійної презентації, які також містять креслення, передбачені технічним завданням. Якість виконання пояснювальної записки та графічної частини добра, розробку виконано в повному обсязі.*

б) самостійність роботи над проектом: *Протягом всього строку дипломного проектування та переддипломної практики здобувач освіти Аратовський В.В. виконував всі етапи розробки проекту, без порушення термінів. Роботу студент виконував в більшій мірі самостійно, з оглядом на рекомендації керівника та отримуючи зворотній зв'язок.*

в) теоретична підготовка випускника (випускниці): *Здобувач освіти Ковальов А.О. під час роботи над дипломним проектом проаналізував достатню кількість літературних джерел та матеріалів за даною тематикою.*

Вважаю, що теоретична підготовка дипломника якісна і він готовий до захисту дипломного проекту.

г) вміння розв'язувати виробничі та конструкторські питання

Під час дипломного проектування здобувач освіти Ковальов А.О. приймав рішення щодо вибору обладнання, аналізував вимоги на етапах проектування, розробляв проектні рішення, обґрунтовував вибір платформи розробки на основі інформації з відкритих джерел, аналізував мови програмування та алгоритми реалізації.

Оцінка розрахункової частини *Добре*

Оцінка графічної частини *Відмінно*

Загальна оцінка *Добре*

Прізвище, ім'я, по батькові керівника дипломного проекту

Стайкуца Сергій Володимирович

Місце роботи і посада керівника дипломного проекту

"Державний університет інтелектуальних технологій і зв'язку", доцент кафедри кібербезпеки та технічного захисту інформації, помічник декана факультету інформаційних технологій та кібербезпеки

Підпис

«20» червня 2025 р.

ДОЗВІЛ
НА РОЗМІЩЕННЯ
ВИПУСКНОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
(ДИПЛОМНОГО ПРОЕКТУ)
В ЕЛЕКТРОННОМУ РЕПОЗИТАРІЇ ВСП «ОТФК ОНТУ»

Ми, що нижче підписалися,

Ковальов Андрій Олексійович
здобувач освіти гр. 4КБ-02, та

Стайкуца Сергій Володимирович,
керівник дипломного проекту,

не заперечуємо щодо розміщення електронного варіанту пояснювальної записки до дипломного проекту фахового молодшого бакалавра на тему:

«Розробка системи контролю надійності паролів для корпоративного сайту» (автор роботи – Ковальов А.С., керівник роботи – Стайкуца С.В.)

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2025 році, у повному обсязі в електронному репозитарії ВСП «ОТФК ОНТУ» для вільного доступу через мережу Інтернет.

Несемо відповідальність за ідентичність електронного та друкованого варіантів випускної кваліфікаційної роботи і даємо згоду на обробку персональних даних.

Виконавець _____ / Ковальов А.С. /

Керівник _____ / Стайкуца С.В. /

«18» червня 2025 р.

ДОВІДКА

циклової комісії КТ та ПІ
про допуск до захисту дипломного проєкту
здобувача (здобувачки) освіти IV курсу
відділення комп'ютерних систем групи 4КБ-02

Ковальова Андрія Олексійовича

на тему *Розробка системи контролю надійності паролів для корпоративного сайту*

Висновок відповідальної особи за проведення нормоконтролю:

пояснювальна записка до дипломного проєкту виконана з некритичними порушеннями ДСТУ та оформлена відповідно до вимог Положення про дипломне проєктування

(підпис) *18.06.2025* (дата) *Петрашова В.І.* (П.І.Б.)

Висновок відповідальної особи за перевірку роботи на наявність академічного плагіату *згідно звіту про перевірку від 17.06.2025 р. значення коефіцієнту подібності в роботі становить 15,36%, коефіцієнт цитування – 2,01%.*

(підпис) *18.06.2025* (дата) *Краснокутська К.Г.* (П.І.Б.)

Попередня експертиза (малий захист) дипломного проєкту

здобувача (здобувачки) освіти *Ковальова А.О.* (П.І.Б.)

проведена «*18*» *червня* 2025 р.

Висновки *Пояснювальна записка до дипломного проєкту виконана у повному обсязі. Випускна кваліфікаційна робота (дипломний проєкт) відповідає вимогам Положення про дипломне проєктування та рекомендована до захисту.*

Голова ЦК КТ та ПІ (підпис) *Кривченко Ю.В.* (П.І.Б.)

StrikePlagiarism.com

Дата звіту 6/17/2025
Дата редагування 6/17/2025

Документ прийнятий

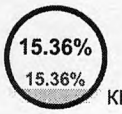
Звіт подібності

метадані

Назва організації
Odesa Technical Professional College of Odesa National University of Technology
Заголовок
Розробка системи контролю надійності паролів для корпоративного сайту
Автор Науковий керівник / Експерт
Ковальов Андрій ОлексійовичСтайкуца Сергій Володимирович
підрозділ
Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету"

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1

КЦ

25
Довжина фрази для коефіцієнта подібності 2

13473
Кількість слів

105960
Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		1
Інтервали	A→	0
Мікропробіли		0
Білі знаки		0
Парафрази (SmartMarks)	a	44

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://www.mecs-press.org/ijcnis/ijcnis-v8-n7/IJCNIS-V8-N7-4.pdf	206 1.53 %
2	https://www.mecs-press.org/ijcnis/ijcnis-v8-n7/IJCNIS-V8-N7-4.pdf	114 0.85 %
3	https://card-file.ontu.edu.ua/bitstreams/12d5c0ab-e979-48f2-a8ec-d5fc31f71fd5/download	62 0.46 %
4	https://card-file.ontu.edu.ua/bitstreams/538ada8a-2c79-4b1e-b7d2-b0c97f68bc1c/download	61 0.45 %
5	https://card-file.ontu.edu.ua/bitstreams/55e2b8f2-7d3c-4235-99fc-2be51199b96d/download	46 0.34 %

6	https://card-file.ontu.edu.ua/bitstreams/6cf43324-8f08-4031-ba42-f80b18efbbc8/download	43 0.32 %
7	https://card-file.ontu.edu.ua/bitstreams/6cf43324-8f08-4031-ba42-f80b18efbbc8/download	41 0.30 %
8	Кваліфікаційна Робота Платонов Кирило ФІТ 4.4 [redacted] 6/6/2025 State University of Trade and Economics (Кафедра інженерії програмного забезпечення та кібербезпеки)	39 0.29 %
9	https://card-file.ontu.edu.ua/bitstreams/82a6d375-2b69-4233-b80f-fbfd149b7747/download	37 0.27 %
10	https://card-file.ontu.edu.ua/bitstreams/538ada8a-2c79-4b1e-b7d2-b0c97f68bc1c/download	34 0.25 %

з домашньої бази даних (0.12 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Розробка web-застосунку для генерації повідомлень із використанням технологій штучного інтелекту [redacted] 6/14/2025 Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	11 (2) 0.08 %
2	Розробка системи авторизації користувача на web-сервері за допомогою nrf-модулю [redacted] 6/15/2025 Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	5 (1) 0.04 %

з програми обміну базами даних (0.80 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Кваліфікаційна Робота Платонов Кирило ФІТ 4.4 [redacted] 6/6/2025 **State University of Trade and Economics (Кафедра інженерії програмного забезпечення та кібербезпеки)**	**50 (3) 0.37 %**
2	Торгівельна web платформа для продажу 3d моделей [redacted] 3/16/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	15 (2) 0.11 %
3	МойсеєнкоВВ_Розробка автоматизованої системи керування паролями з дослідженням мультиваріантної оцінки їхньої надійності на базі машинного навчання.docx [redacted] 12/8/2023 Kryvyi Rih National University (Кафедра моделювання і програмного забезпечення)	15 (2) 0.11 %
4	«Розробка веб-сайту для пошуку вакансій» [redacted] 6/6/2025 Zhytomyr Agricultural Technical Professional College (Zhytomyr Agricultural Technical Professional College)	13 (2) 0.10 %
5	Balchikbayeva D., Anuarov B. CS-2120.pdf [redacted] 6/3/2024 Astana IT University (Astana IT University)	10 (1) 0.07 %
6	Система створення інтерактивних карт [redacted] 3/15/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	5 (1) 0.04 %

з Інтернету (14.44 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	**https://card-file.ontu.edu.ua/server/api/core/bitstreams/44c16132-5f53-48e2-b6c0-61e9a2f0fd75/content**	**465 (38) 3.45 %**
2	**https://www.mecs-press.org/ijcnis/ijcnis-v8-n7/IJCNIS-V8-N7-4.pdf**	**349 (4) 2.59 %**
3	**https://card-file.ontu.edu.ua/bitstreams/538ada8a-2c79-4b1e-b7d2-b0c97f68bc1c/download**	**290 (16) 2.15 %**
4	https://card-file.ontu.edu.ua/bitstreams/aed610a6-43ef-47e0-9066-e85c89456f3e/download	165 (14) 1.22 %
5	**https://card-file.ontu.edu.ua/bitstreams/6cf43324-8f08-4031-ba42-f80b18efbbc8/download**	**84 (2) 0.62 %**
6	https://card-file.ontu.edu.ua/bitstreams/62baa43e-b968-4993-bb54-8cf8761a89b2/download	71 (6) 0.53 %
7	**https://card-file.ontu.edu.ua/bitstreams/12d5c0ab-e979-48f2-a8ec-d5fc31f71fd5/download**	**62 (1) 0.46 %**
8	**https://card-file.ontu.edu.ua/bitstreams/82a6d375-2b69-4233-b80f-fbfd149b7747/download**	**62 (3) 0.46 %**
9	**https://card-file.ontu.edu.ua/bitstreams/55e2b8f2-7d3c-4235-99fc-2be51199b96d/download**	**59 (2) 0.44 %**
10	https://card-file.ontu.edu.ua/bitstreams/549ee9fe-7574-4ae5-b500-9fe2711f33e6/download	58 (6) 0.43 %
11	https://card-file.ontu.edu.ua/bitstreams/bbaf3f38-16a8-4070-bead-5562769b7c71/download	49 (4) 0.36 %
12	https://card-file.ontu.edu.ua/bitstreams/21173711-5b67-4b87-b17f-6302c25e7a31/download	46 (3) 0.34 %
13	https://card-file.ontu.edu.ua/bitstreams/035f6436-20b4-4ee6-8e99-bede670e308b/download	35 (4) 0.26 %
14	**https://card-file.ontu.edu.ua/bitstreams/53ed22ad-8700-4162-b97a-082a1ad472d6/download**	**34 (1) 0.25 %**
15	https://card-file.ontu.edu.ua/bitstreams/9908b7a9-6b3e-46f5-a46e-84d83787cfd4/download	29 (3) 0.22 %
16	https://dev.to/franciscomendes10866/schema-validation-with-zod-and-expressjs-111p	18 (2) 0.13 %
17	https://tux.org.ua/identifikatsiya-ta-autentifikatsiya-koristuvachiv/	14 (1) 0.10 %
18	https://www.cnblogs.com/woider/p/6835466.html	11 (1) 0.08 %
19	https://card-file.ontu.edu.ua/bitstreams/72fa1396-889f-4082-af7d-898b6ac28dd4/download	11 (1) 0.08 %
20	https://card-file.ontu.edu.ua/server/api/core/bitstreams/a141b658-5fa7-4f90-b0bd-7f0ccaed21e5/content	10 (1) 0.07 %
21	https://card-file.ontu.edu.ua/bitstreams/34a6756b-592f-4b77-a805-183aa03a6a26/download	9 (1) 0.07 %
22	http://rep.knlu.edu.ua/xmlui/bitstream/handle/787878787/1969/%D0%90%D0%BD%D1%82%D0%BE%D0%BD%D0%BE%D0%B2%D0%B0.pdf?sequence=1	8 (1) 0.06 %
23	https://card-file.ontu.edu.ua/server/api/core/bitstreams/8da72e29-656f-4ee4-9b22-716dedf53ff5/content	7 (1) 0.05 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютерна інженерія»
Освітньо-професійна програма: **«Безпека комп'ютерних систем і мереж» Група:** 4КБ- 02

Дипломний **проект здобувача освіти денної форми навчання КБ.** 02.08.000.ДП