МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерних наук

ПРОКАЗА Андрій Іванович

Математичне та програмне забезпечення для керування паролями / Mathematical tools and software for password management

спеціальність: 121 - Інженерія програмного забезпечення
освітньо-професійна програма - Інженерія програмного забезпечення

Кваліфікаційна робота

Виконав студент групи ІПЗм-21
А. І. Проказа

_____

Науковий керівник:
к.т.н., доцент Р. П. Шевчук

_____

Кваліфікаційну роботу
допущено до захисту:

"___" _____ 20___ р.

Завідувач кафедри
_____ **А. В. Пукас**

ТЕРНОПІЛЬ - 2022

ЗМІСТ

ВСТУП 2
РОЗДІЛ І ОСОБЛИВОСТІ РОБОТИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ КЕРУВАННЯ ПАРОЛЯМИ 5
1.1. Особливості парольної ідентифікації 5
1.2. Аналіз програмного забезпечення для керування паролями 10
1.3. Постановка задачі дослідження 17
Висновки до розділу І 18
РОЗДІЛ ІІ АРХІТЕКТУРА ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ ПРОГРАМНИХ ПРОДУКТІВ ДЛЯ КЕРУВАННЯ ПАРОЛЯМИ 19
2.1. Архітектура програмного забезпечення для керування паролями 19
2.2. Розробка базових алгоритмів роботи програмного забезпечення 21
2.3. Прототипування 23
Висновки до розділу ІІ 27
РОЗДІЛ ІІІ РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ КЕРУВАННЯ ПАРОЛЯМИ 28
3.1. Обґрунтування вибору засобів розробки 28
3.2. Особливості програмної реалізації програмного забезпечення та бази даних 30
3.3. Тестування програмного забезпечення 36
Висновки до розділу ІІІ 45
ВИСНОВКИ ТА ПРОПОЗИЦІЇ 46
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 47
ДОДАТОК А ДЕКЛАРАЦІЯ ДОБРОЧЕСНОСТІ **Помилка! Закладку не визначено.**
ДОДАТОК Б ЛІСТИНГ ОСНОВНИХ ПРОГРАМНИХ МОДУЛІВ.. **Помилка! Закладку не визначено.**
ДОДАТОК В КОПІЯ ПУБЛІКАЦІЙ..... **Помилка! Закладку не визначено.**

ВСТУП

Актуальність теми. Одним з найбільш важливих кроків, які можна застосувати, щоб захистити себе – це використання унікального надійного паролю для кожного з облікових записів і додатків. На жаль, практично неможливо запам’ятати всі свої паролі. Крім того, щоб постійно вводити паролі на різних сайтах, генерувати нові паролі, відслідковувати відповіді на всі питання безпеки потрібно досить багато часу. А використовувати той самий пароль для всіх аккаунтів надзвичайно небезпечно. Однак існує рішення, що може спростити життя в безпечний спосіб – це менеджери паролів. З назви зрозуміло, що менеджер паролів - це програма, що дозволяє зберігати всі введені паролі (і не тільки паролі) і керувати ними [1]. Це її призначення. Такі додатки виступають в ролі сховища даних, часто прив'язаного до серверів однієї компанії, яка зобов'язується ці дані оберігати від зловмисників. Менеджер паролів допомагає швидше заповнювати форми для авторизації і автоматизує цей процес. Але що ще важливіше, він бере на себе процедуру створення пароля. Необережні користувачі вибирають собі паролі на кшталт «123456» або «пароль». Вони роблять це, щоб не запам'ятовувати щось більш складне, що підсумку ставить під загрозу особисті дані. У більшості браузерів є хоча б примітивне управління паролями. Це краще, ніж використовувати один і той же пароль для всіх сервісів, але можливості браузерів обмежені - все-таки вони створювалися для інших цілей. Більшість з них не згенерують надійний пароль. Експерти з безпеки рекомендують використовувати спеціалізовані сервіси. Вони мають одне призначення і рік за роком отримують нові корисні функції. Тому актуальним є розробка програмного забезпечення для керування паролями та їх шифруваня.

Зв’язок роботи з науковими програмами, планами, темами.

Напрямок виконаних досліджень безпосередньо пов'язаний з науково-дослідним напрямком кафедри комп’ютерних наук Західноукраїнського національного університету.

Об’єкт дослідження – процес керування паролями та їх шифрування.

Предмет дослідження є методи та програмні засоби для керування паролями та їх шифрування.

Методи дослідження: методи об’єктно-орієнтованого програмування та імітаційного моделювання.

Мета дослідження – підвищення ефективності керування паролями та їх шифрування за рахунок методу оцінювання складності паролю, що дає можливість підвищити безпеку системи збереження паролів.

Для досягнення поставленої мети потрібно вирішити наступні задачі:

- проаналізувати програмне забезпечення для керування паролями;
- розробити метод оцінювання складності паролю;
- розробити модель системи керування паролями та їх шифрування;
- розробити UX/UI дизайн для клієнтського web-додатку;
- розробити програмну реалізацію системи керування паролями та їх шифрування;

Наукова новизна.

- подальшого розвитку дістав метод зміни паролю, який, на відмінну від існуючих, динамічно змінює час використання паролю з урахуванням ймовірних загроз, що дозволяє підвищити надійність роботи системи.

Практичне значення.

У роботі реалізовано програмне забезпечення для підвищення ефективності керування паролями та їх шифрування за рахунок методу оцінювання складності паролю, що дає можливість підвищити безпеку системи збереження паролів.

Програмне забезпечення реалізовано із використанням Node.js, Docker, SSH), Java, React та Redux.

Особистий внесок здобувача.

Всі основні результати отримані автором особисто.

Апробація результатів.

Основні положення магістерського дослідження апробовані на III Міжнародній науково-практичній онлайн-конференції «Актуальні проблеми, пріоритетні напрямки та стратегії розвитку України» 13 жовтня у м. Київ.

Публікації.

Паляниця Т.В. Особливості розробки мобільного застосунку для попередження про надзвичайні ситуації / Т.В. Паляниця // Актуальні проблеми, пріоритетні напрямки та стратегії розвитку України: тези доповідей III Міжнародної науково-практичної онлайн-конференції, м. Київ, 13 жовтня 2021 року/ редкол. О.С. Волошкіна та ін. – К.: ІТТА, 2021. – с. 1366 - 1368

РОЗДІЛ І
ОСОБЛИВОСТІ РОБОТИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ КЕРУВАННЯ ПАРОЛЯМИ

1.1. Особливості парольної ідентифікації

В даний час безліч видів діяльності пов'язані з телекомунікаційними системами та мережами. При цьому особливе місце займає глобальна мережа Інтернет. Відомо, що Інтернет – це ефективний інструмент наукових досліджень, ведення бізнесу, впливу на людей без територіальних і національних кордонів.

Рівень інформаційної культури і знань в цій області необхідний всім співробітникам підприємств, які прагнуть вижити в умовах жорсткої конкуренції. Нові технології пов'язані з новими можливостями, прибутками і ринками. З ростом залученості важливих бізнес ресурсів в мережеві структури зростає і ризик роботи з ними.

При цьому більшість компаній вдаються до технології віддаленого доступу або загальному доступу декількох компаній до одного ресурсу через мережеві протоколи. Підключення віддалених користувачів до корпоративних ресурсів пов'язано зі значним ризиком і слабкою ланкою тут є парольний захист. Приблизно 30 років саме парольний захист стоїть на першому рубежі забезпечення безпеки телекомунікаційних систем і мереж. Серед основних переваг парольного захисту виділяють її звичність і простоту. Приклад проходження користувачем аутентифікації на основі пароля представлений на рис. 1.1 і зводиться до наступних основних операцій:

- користувач (Андрій) вводить свої ім'я (логін) і пароль (Qwe12Ty) на робочій станції;

- ім'я користувача та пароль передаються по мережі у відкритому вигляді;

- сервер аутентифікації знаходить обліковий запис Андрій в базі даних аутентифікації і порівнює введені дані з її змістом;

- при збігу даних – користувачеві призначаються певні права, в іншому випадку потрібно знову ввести логін і пароль.

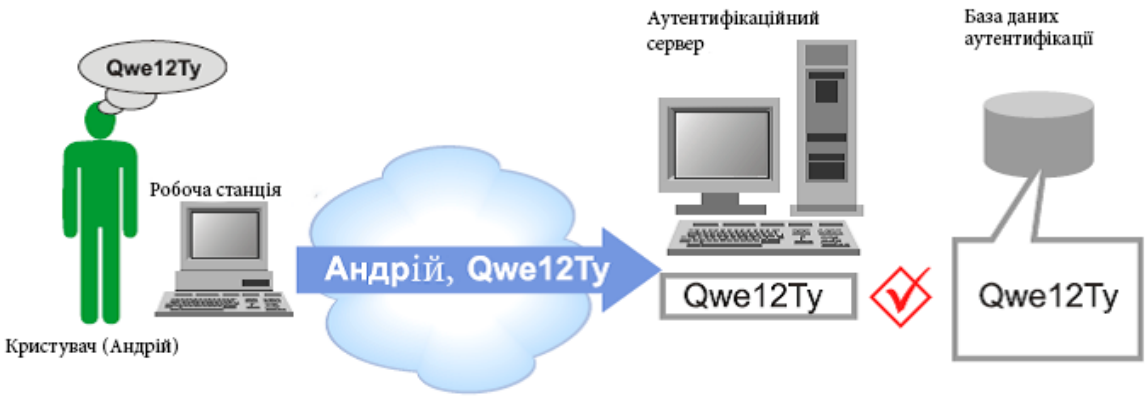

Рис. 1.1. Аутентифікація на основі пароля

Алгоритм виконання зазначених операцій представлений на рис. 1.2.

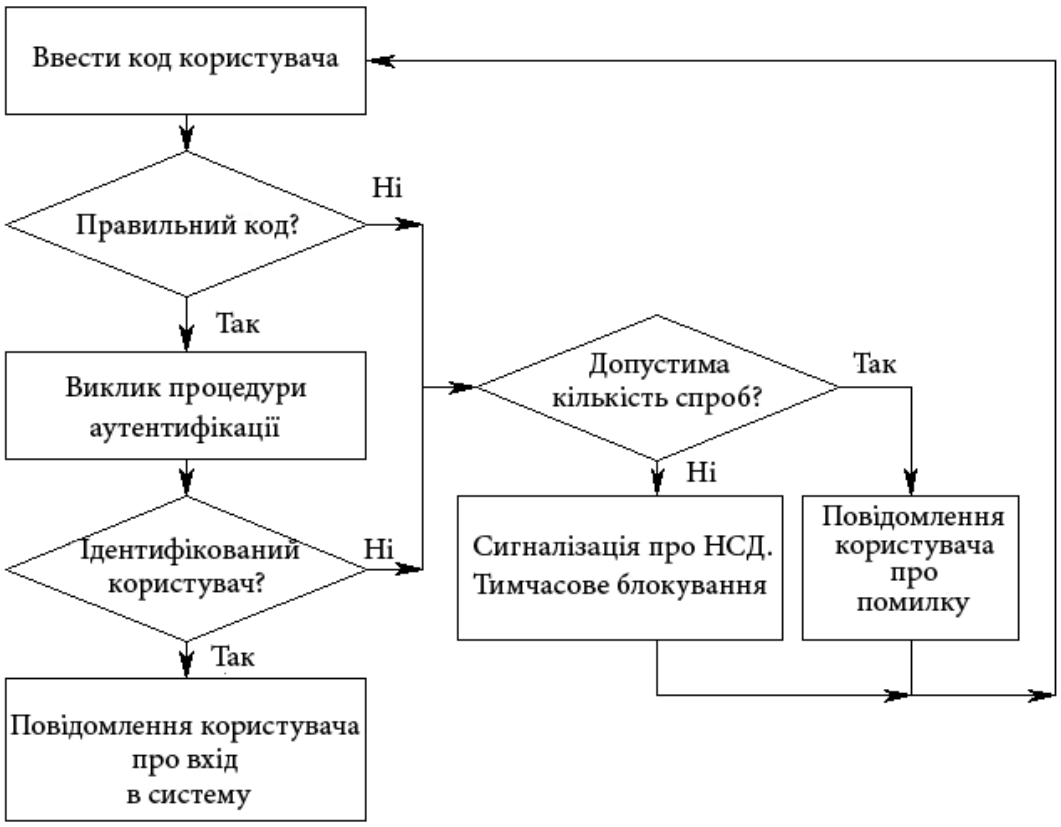

Рисунок 1.2 – Алгоритм виконання ідентифікації

Відомо, що 90% компаній використовують статичний пароль в якості єдиної кордону, який відділяє конфіденційну інформацію від «зовнішнього світу». Причому, розширення сфер використання парольного захисту, супроводжується великою винахідливістю методів крадіжки паролів.

За оцінками провідних фахівців, приблизно 80% інцидентів в сфері інформаційної безпеки трапляються внаслідок використання слабких паролів – до такого висновку прийшла компанія Trustwave за результатами власного дослідження, що охопило ряд компаній в 18 регіонах світу [6]. Аналітики досліджували вразливість елементів в системах інформаційної безпеки. При цьому були вивчені більше 300 інцидентів, що мали місце в 2011 році. Головний висновок, зроблений в результаті: слабкі паролі користувачів в інформаційних системах – найбільш вразливе місце, яке використовують зловмисники, як у великих, так і в невеликих компаніях.

Слабкий пароль – це погано, але зворотня сторона застосування складних паролів – труднощі утримання пароля в пам'яті людини. Як наслідок – недбалість зберігання пароля у вигляді робочих записів. Знаючи традицію поводження з парольною інформацією працівниками компаній, для зловмисника не важко отримати ці відомості.

Деякі зарубіжні компанії, що діють у сфері аналізу інцидентів в системах безпеки, роблять висновок: несанкціонований доступ до інформації про фінансову активності підприємства, договорах і графіках роботи здатний позначитися не те, що втратами, а й повним руйнуванням. Щорічні втрати від витоків інформації в США оцінюються мільярдами доларів [9]. Російський галузевий портал «Інформаційна Безпека Банків» в оцінці фінансового збитку від можливих зловживань співробітників посилається на дослідження Асоціації експертів з боротьби з шахрайством (ACFE, Association of Certified Fraud Examiners, США), яка бачить цю суму в розмірі 6% прибутку банку за рік. За спостереженнями асоціації, втрати при подібних інцидентах, в середньому, досягали 100 тис. доларів, а в 14,6% перевищили 1 млн. доларів.

Дослідницька компанія Javelin Strategy в своєму щорічному дослідженні, опублікованому в лютому 2021 року, оцінила світовий обсяг шахрайства і витоків даних з компаній і організацій за 2021 рік в 18 млрд. доларів.

Справа вже не в кількості паролів, що запам’ятовуються і диференціації символів самого пароля, який може бути з легкістю скомпрометований через недбалість користувачів; різні шкідливі програми і атаки комп'ютерної інженерії дозволяють отримати пароль незалежно від його складності. Крім цього, значні інтервали часу, протягом яких пароль і логін користувача залишаються незмінними, дозволяють застосувати різні методи їх перехоплення і підбору.

Існують методи кількісної оцінки стійкості парольних данних (формула Андерсона) [11]:

$$4.32 \cdot 10^4 \cdot (k) \frac{M}{P} \le A^l, \quad (1.1)$$

де k – кількість спроб підбору пароля в хвилину; M – час дії пароля в місцях; P – ймовірнусть вибору пароля; A^l – потужність простору паролів (A – потужність алфавіту паролів, l – довжина пароля).

Потужність простору паролів (загальна кількість паролів), як відомо з комбінаторики, залежить від кількості знаків алфавіту (A) і довжини пароля (l). Час (T) за який пароль заданої довжини буде підібраний методом «грубої сили» визначається

$$T \le \frac{A^l}{V}, \quad (1.2)$$

де V – швидкість перебору.

Таблиця 1.1 обчислена за цією формулою, наочно демонструє залежність від зазначених величин. При цьому припускали $V = 1 \cdot 10^7$ паролей/с.

Таблиця 1.1 – Час повного перебору всіх паролів заданого алфавіту

Кількість знаків алфавіту	Довжина пароля			
	6 символов	8 символов	10 символів	12 символів
26 (латиниця всі маленькі або всі великі)	31 сек	5 годин 50 хв	163.5 діб	303 роки
52 (латиниця зі змінним регістром)	33 хв	62 доби	458 років	1,239,463 років
62 (латиниця різного регістра плюс цифри)	95 хв	252 доби 17 годин	2,661 рокі в	10,230,425 років
68 (латиниця різного регістра плюс цифри плюс знаки пунктуації.,;:!?)	2 годин 45 хв	529 діб	6703 років	30,995,621 років
80 (латиниця різного регістра плюс цифри плюс знаки пунктуації.,;:!? плюс дужки ()[]{} плюс #$%&*~)	7 годин 30 хв	5 років 4 місяці	34048 рокі в	217,908,031 років

Тому парольної або однофакторної аутентифікації для забезпечення надійної інформаційної безпеки недостатньо.

Для підвищення безпеки доступу до ресурсів і послуг на практиці використовують многофакторную аутентифікацію. Як додатковий фактор використовується річ, якою володіє суб'єкт. Наприклад, електронна або магнітна карта, флеш-пам'ять [11]. Зараз в якості таких засобів застосовуються персональні апаратні пристрої –- токени. По суті, токен – це

смарт-карта або USB-ключ. Токени дозволяють генерувати і зберігати ключі шифрування, забезпечуючи тим самим сувору аутентифікацію. На жаль, як і пароль, токен можна викрасти. Особливо використання такого підходу до аутентифікації володіє серйозною вразливістю при роботі з чужих комп'ютерів, наприклад в інтернет-кафе.

Сьогодні застосування рішень для багатофакторної аутентифікації є необхідною умовою забезпечення безпеки корпоративних ресурсів будь-якої організації. Переваги та недоліки двофакторної аутентифікації, в загальному, відомі [12]. До переваг можна віднести її здатність захистити інформацію, як від внутрішніх загроз, так і від зовнішніх вторгнень. Певною слабкістю можна вважати необхідність використання додаткових програмно-апаратних комплексів, пристроїв зберігання і зчитування даних. У той же час, зараз статистика зломів обчислювальних систем, які застосовують двухфакторну аутентифікацію, значно нижче, ніж у однофакторной аутентифікації.

Така аутентифікація вже сьогодні застосовується низкою компаній у сфері фінансів при створенні сервісів інтернет-банкінгу, мобільного банкінгу, файлообміну і інших рішень для кінцевих користувачів.

1.2. Аналіз програмного забезпечення для керування паролями

Паролі повинні бути складними і різними для кожного сервісу. Але щоб дотримуватися цього правила, необхідно використовувати менеджери паролів. Всі дані для авторизації на різних сервісах будуть зберігатися в ньому. Менеджери паролів допомагають створювати надійні унікальні паролі при реєстрації на веб-сайтах і зберігають їх на своїх серверах. Щоб зайти на ресурс або в додаток, можна скопіювати потрібний пароль з менеджера і вставити у відповідне поле. Часто ці програми дозволяють не тільки запам'ятовувати, але й автоматично вводити пароль на сайті. Перед початком варто звернути увагу на способи зберігання даних в менеджерах паролів. Інформація може зберігатися або віддалено в хмарі сервісу, або локально, на

комп'ютері користувача. У кожного із способів є плюси і мінуси. Хмарне зберігання і синхронізація зручні тим, що сервіси паролів доступні на всіх пристроях, де вони потрібні. Але, якщо хмара зламають, паролі можливо потраплять до зловмисників. Локальне місце зберігання більш надійне (хіба що у вас не вкрадуть комп'ютер або ноутбук), але менш зручний [3]. Припустимо, ви створили і пароль від Фейсбуку за допомогою менеджера паролів і зберегли інформацію на ПК. Але якщо зайти в Facebook зі смартфона, то новий пароль автоматично НЕ підтягнеться і його доведеться вводити вручну. Такий підхід А якщо ми говоримо про десятки акаунтів на різних сайтах, то подібна схема стає занадто незручною. Існують різні види подібного роду програм. Одні поширюються в вигляді розширень для популярних браузерів, інші - можна запускати в якості програмного забезпечення на комп'ютері. Також, більшість менеджерів паролів, 13 можна встановлювати на смартфонах і планшетах. Вибирайте відповідний варіант і зберігайте свої паролі і секретну інформацію в стовідсоткової безпеки.

Перед тим як приступити до розробки власного веб-додатку для управління паролями було проведено аналіз ринку. Нижче представлено порівняльний аналіз програм до управління паролями та їх шифрування. Спочатку було обрано параметри до порівняння. Серед головних критеріїв було визначено наступні: можливість 256-бітового AES-шифрування, наявність протоколів з нульовим розголошенням, забезпечення аутентифікації, а також наявний функціонал. Менеджери паролів пропонують безліч різних функцій: наприклад, автоматичне заповнення форм, генерація паролів, а деякі з них дозволяють навіть підключитися до VPN. Було зроблено огляд деяких з цих функцій, щоб з'ясувати, які з них є дійсно корисними, а які - просто приємними доповненнями. До уваги було брано теж такий критерій, як простота використання,адже менеджер паролів передусім повинен бути зручним. Також було проаналізовано ціну кожного з рішень. Першим рішенням до порівняння було обрано Google Chrome -

вбудований менеджер паролів, який пропонує можливість їх зберігання під обліковим записом Google (рисунок 1.3).

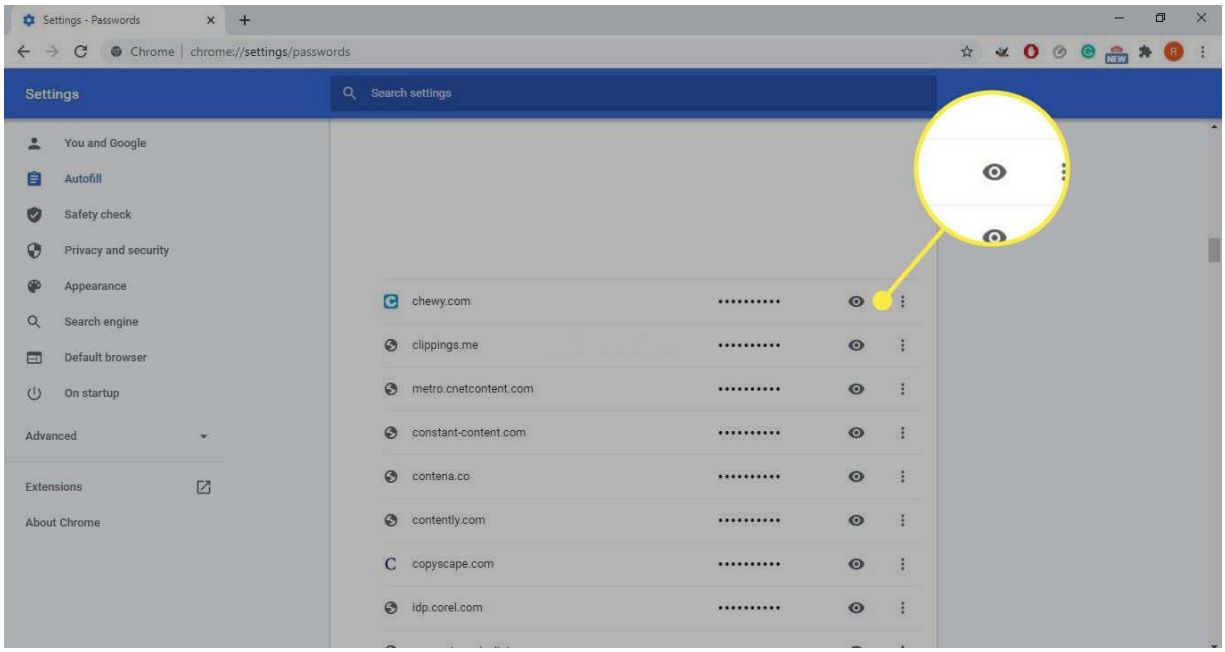

Рис.1.3. Інтерфейс менеджера паролів в Chrome

Він зручний, доступний і зрозумілий кожному користувачеві. Менеджер Chrome має можливість генерувати паролі, але варто відзначити, що ключі, які пропонує Chrome, не здаються такими вже й складними в порівнянні з комерційними аналогами [5]. Наприклад, відсутня можливість поставити більшу кількість знаків або використовувати спеціальні символи. В цілому це - доступний і звичний інструмент, проте деякі фахівці в області інформаційної безпеки вважають його не надійним, так як відсутній майстер-пароль і при компрометації облікового запису є ризик, що зловмисник зможе заволодіти всіма даними. Варто згадати і про те, що основний продукт компанії Google - дані користувачів, які використовуються для таргетування 14 реклами і іншого. Тому, можливо, не варто зберігати абсолютно всю інформацію в одному вбудованому менеджері, в тому числі - особливо значиму.

Менеджер паролів Dashlane [6], крім основних функцій надає можливість перевірити паролі в сховище на предмет стійкості. Головне вікно додатку зображено на рисунку 1.4 Також Dashlane підтримує Windows Hello -

можливість входу з використанням біометричних даних, в тому числі за допомогою сканування особи і відбитка пальця - і веде моніторинг даркнета, дозволяючи відслідковувати скомпрометовані адреси електронної пошти, паролі та фінансові відомості. Менеджер має безкоштовну версію, проте її можливості обмежені: в ній можна зберегти не більше 50 паролів.

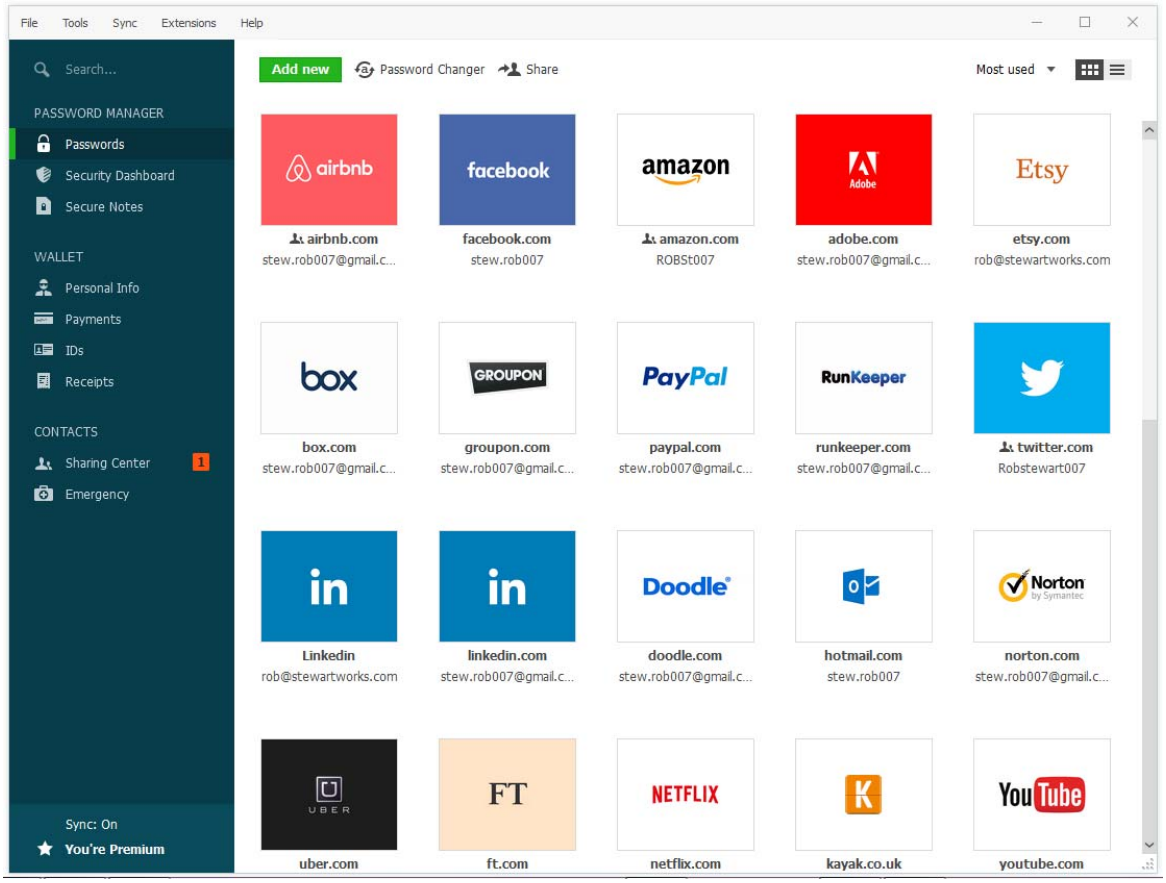

Рис.1.4. Інтерфейс менеджера паролів в Dashlane

Ще одним аналогом, який було взято до уваги є менеджер паролей Keeper. Це програма керування паролями та цифровий сейф, створений Keeper Security, який зберігає паролі веб-сайту, фінансову інформацію та інші конфіденційні документи, використовуючи 256-бітове шифрування AES, архітектуру нульових знань та двофакторну автентифікацію [7]. Сервіс пропонує лаконічний і зручний інтерфейс і надає безпечне сховище об'ємом в 10 ГБ. Так само як і Dashlane, він підтримує біометричну аутентифікацію з Windows Hello. Головне вікно додатку зображено на рисунку 1.5.

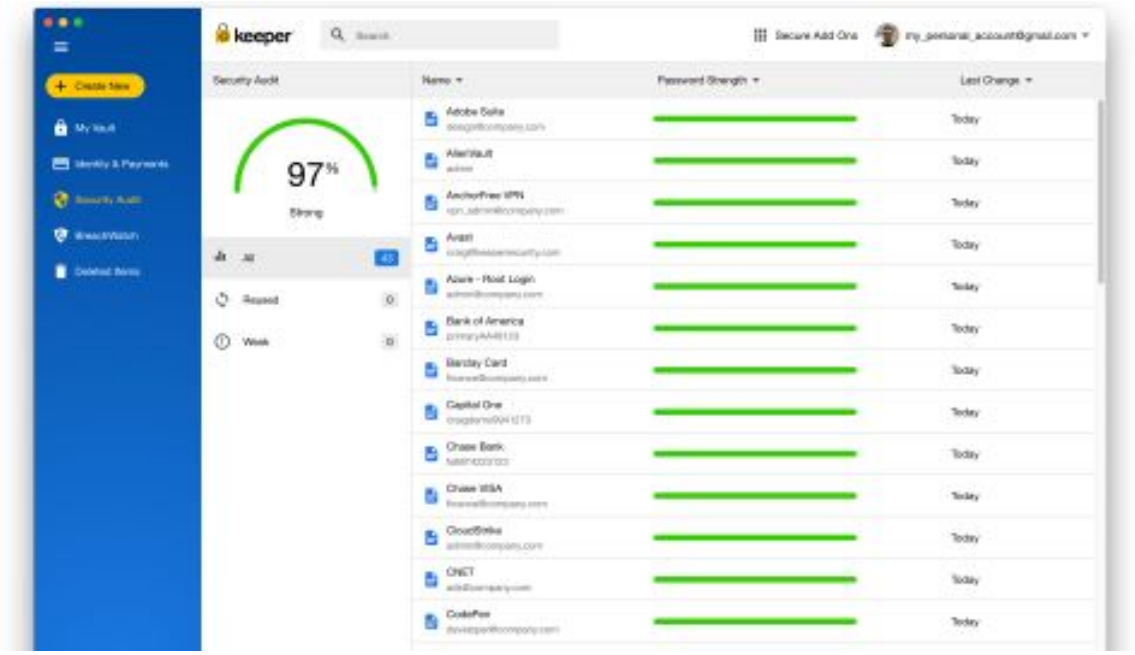

Рис.1.5. Інтерфейс менеджера паролів в Keeper

Як і попередні менеджери, 1Password сумісний з Windows Hello і сканує даркнет на предмет витоків інформації. Зашифроване сховище розраховане на зберігання 1 ГБ даних [8]. Головне вікно додатку зображено на рисунку 1.6.

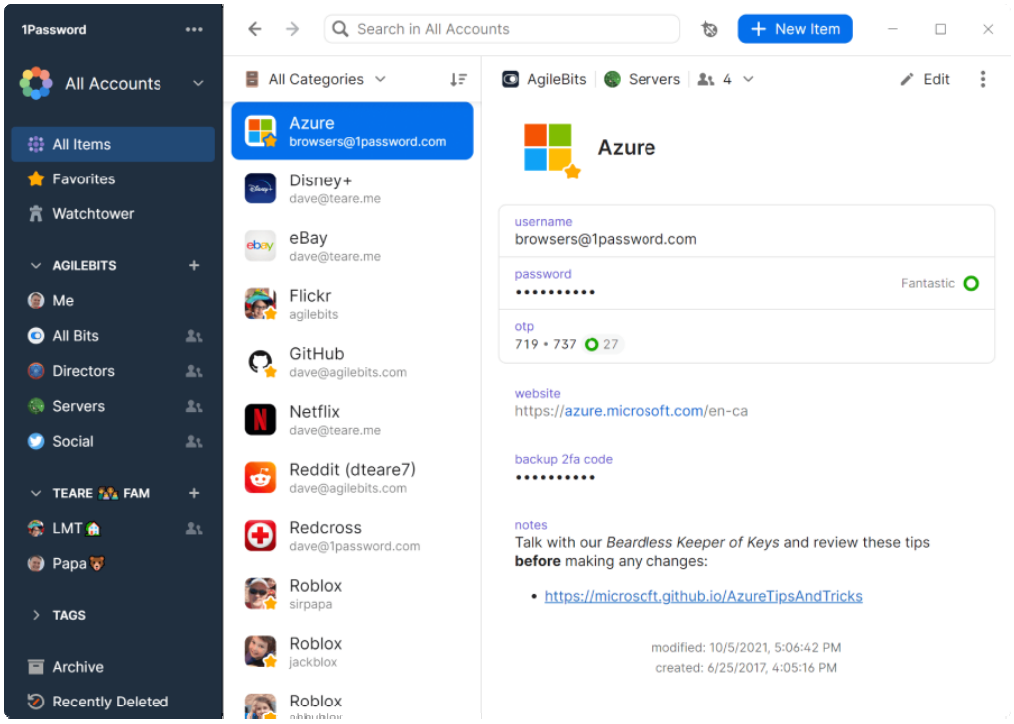

Рис.1.6. Інтерфейс менеджера паролів в 1Password

З унікальних функцій менеджера необхідно виділити сімейний тариф, який дозволяє приєднати одночасно до п'яти користувачів з необмеженою

кількістю пристроїв, і вбудовану функцію батьківського контролю, яка дозволяє простежити за тим, щоб діти не змогли змінити паролі від важливих ресурсів.

Безкоштовна версія LastPass [9] пропонує найширший спектр можливостей з усіх комерційних менеджерів паролів, існуючих на ринку: вона дає можливість збереження необмеженої кількості паролів на необмеженій кількості пристроїв з додатковою можливістю надання доступу до них одному користувачеві (рисунок 1.7). Преміум-версія дозволяє давати доступ кільком користувачам, а також забезпечує біометричну ідентифікацію, 1 ГБ простору в сховище і цілодобову підтримку користувачів по електронній пошті.

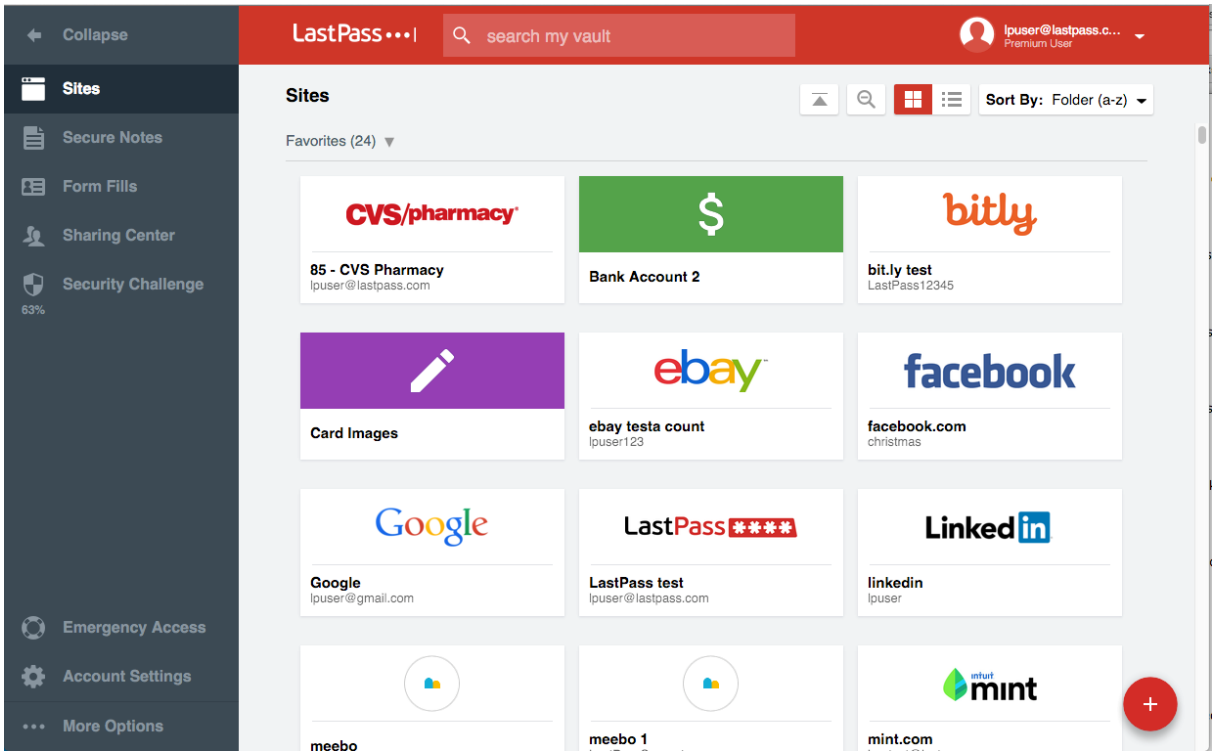

Рис.1.7. Інтерфейс менеджера паролів в LassPass

Підсумувавши все вищесказане, можна відзначити, що використання менеджерів паролів дозволяє значно полегшити роботу з веб-сервісами та убезпечити облікові записи. Даний спосіб зберігання паролів набагато надійніше традиційних, таких як використання одного і того ж пароля або комбінації символів з мінімальним розходженням, і можна відзначити, що все 18 більше користувачів починають освоювати менеджери. Однак при

виборі менеджера варто враховувати його особливості та призначення. Наприклад, вбудовані в браузер менеджери зручні, зрозумілі більшості користувачів, але поки все ще програють комерційним аналогам по здатності генерувати складні паролі, немає можливості перемикатися між браузерами або експертувати паролі до файлу. Через це більшість експертів схиляються до того, щоб розглядати такі менеджери швидше як розширення браузерів і не зберігати в подібного роду додатках важливі дані, наприклад для банківських сервісів. Комерційні менеджери володіють великим спектром можливостей, вони більш комплексні і працюють у вигляді незалежних додатків. Підсумовуюча таблиця подана нижче (таблиця 1.2).

Таблиця 1.2 – Порівняльна таблиця аналогів

	Passportal	1Password	dashlane	LastPass
Price	$	$$	$$$	$$
Organisation of passwords	★★★★☆	★★★☆☆	★★★☆☆	★★★★★
Customer support	★★★★★	★★★☆☆	★★★☆☆	★★☆☆☆
Autofill in browser	✓	✓	✓	✓
Remote access on other devices	✓	✓	✓	✓
2-factor authentication support	✓	✓	✓	✓
Automatic password generator	✓	✓	✓	✓
Autofill payment information		✓	✓	✓
Password strength report	✓	✓	✓	✓
Unlimited password storage	✓	✓	✓	✓
Mobile app	✓	✓	✓	✓
Security Audits	✓	✓	✓	✓
Data Storage	✓	✓	✓	✓
Batch change passwords			✓	
Unlimited custom fields		✓		
Automatic password changer			✓	
Travel mode		✓		
Overall Score	8	9	7	8

Серед розглянутих аналогів, було відзначено наступні недоліки: блокування акаунту в цілях безпеки, блокування ір адреси, щоб запобігти спробам зламу. Деякі з представлених рішень не передбачають наявність мастер-пароля. Натомість, пропоноване рішення в цій магістерській кваліфікаційній роботі передбачає використання мастер пароля до шифрування всіх інших паролей, що підвищує безпеку такого додатку. Кожна наступна зміна мастер-пароля передбачає перешифрування всіх паролів. Також, варто зазначити, що не всі з представлених рішень передбачають можливість спільного доступу до пароля, а також експорту паролів до файлу. Саме тому актуальним є розробка власного рішення, яке поєднає в собі реалізацію недоліків та відсутнього функціоналу в кожному з розглянутих аналогів.

1.3. Постановка задачі дослідження

Після порівняльного аналізу аналогів до системи у магістерській кваліфікаційній роботі було вирішено скласти список завдань, необхідних для створення програмного продукту. Головною задачею роботи є створення системи, що дозволить спростити процес управління паролями та їх шифрування.

Виконання цієї задачі передбачає:

– проведення обґрунтування доцільності розробки;

– провести аналіз інформаційного забезпечення розробки;

– розробка методу визначення надійності паролю;

– розробка методу оцінювання необхідності змінювати пароль;

– розробка моделі системи управління паролями та їх шифрування;

– розробка повного UX/UI дизайну для клієнського web-додатку;

– програмна реалізація системи керування паролями та їх шифрування;

– проектування архітектури бази даних веб-системи;

– налаштування серверного оточення для розміщення системи керування паролями та їх шифрування;

– розробка серверної частини та програмних засобів системи, що містить API для роботи з клієнтським додатком.

Висновки до розділу I

Розглянуто особливості задачі парольної ідентифікації та роль менеджерів паролів у сучасному світі та визнано їх важливість. Досліджено види систем для управління паролями та розглянуто їх головні функції. Обрано різновид такої системи, а також розглянуто аналоги. В ході перегляду аналогів системи було визначено їх переваги, які можна покращити: замість функціоналу коментування та вподобання публікацій, було вирішено розробити власні методи оцінювання складності паролів.

Виявлено недоліки аналогів, які врахуються у магістерській кваліфікаційній роботі: можливість відновлення мастер-пароля, автоматичне перешифрування паролів з встановленою періодичністю, використання тимчасового паролю для входу, а також двофакторної автентифікації, забезпечення історії змін паролів і можливість відкату.

В результаті дослідження аналогів було сформульовано основні завдання, які необхідно виконати для розробки програмного забезпечення.

РОЗДІЛ ІІ
МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ ТА АРХІТЕКТУРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ КЕРУВАННЯ ПАРОЛЯМИ

2.1. Архітектура програмного забезпечення для керування паролями

Для реалізації системи необхідно обрати технології, які в повній мірі зможуть задовольнити технічні та функціональні вимоги модулів. Оскільки веб-додаток має мікросервісну архітектуру, то варто розглянути окрему частини бекенд та фронтенд. Для бекенду основним фреймворком було обрано Spring Boot. Використовуючи Spring Boot можна досягти таких самих результатів, що і на чистій Java, але набагато меншими зусиллями і меншою кількістю коду. Мета даного фреймворку - щоб програміст зосередився на вирішенні бізнес завдання замість того, щоб витрачати час на налаштування коду. Всі налаштування за додатком можна помістити в файл налаштувань замість java класів. Як правило, файл - application.properties лежить в папці resources. Але і це можна змінити і винести настройки на віддалений сервер наприклад або в інший файл. Spring Boot при запуску завантажує файл залежностей бібліотек. Якщо це Maven то pom.xml. Spring Boot також включає в себе вбудований контейнер сервлетів (по замовчуванні Tomcat) що дозволяє запускати веб додатки як звичайні java програми: використовуючи jar файл. Для розробки фронтенду основним фреймворком було обрано React. React - це бібліотека JavaScript, яка використовується для створення призначеного для користувача інтерфейсу. Відмінною рисою бібліотеки є концентрація на компонентах - ми можемо створити окремі компоненти і потім їх легко переносити з проекту в проект [4]. Ще одна особливість React - використання JSX. JSX представляє комбінацію коду JavaScript і XML і надає простий і інтуїтивно зрозумілий спосіб для визначення коду візуального інтерфейс.

Загальна архітектура програмного забезпечення зображена на рисунку 2.1.

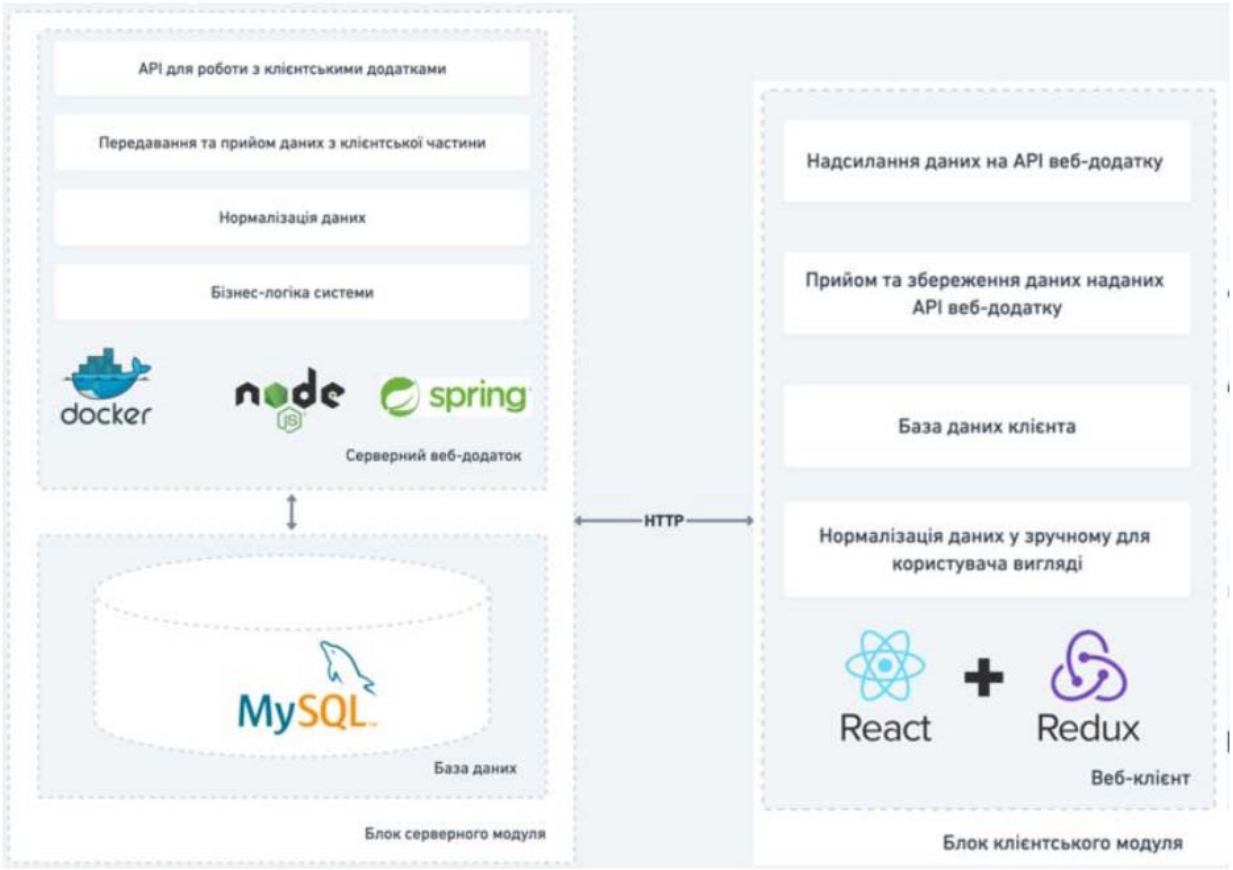

Рис.2.1. Архітектура програмного забезпечення

Програмне забезпечення керування паролями представляє собою багатомодульну програму, що має серверний та клієнтський модулі. Система даними і записує зміни до бази даних по стороні серверної частини. Для роботи клієнтського модуля необхідно зареєструватись та увійти до системи попередньо авторизувавшись. Серверна частина містить базу даних та всю необхідну логіку веб-системи. База даних призначена для зберігання, зміни і обробки необхідної інформації. Веб-додаток, як частина системи містить у собі бізнес логіку 23 системи, нормалізацію даних, що будуть записані до бази, REST API, побудоване на контролерах для роботи з клієнтськими модулями. В цілому, серверна частина являється основною та забезпечує роботу клієнтської сторони, виконуючи функцію передавання та прийому даних системи в цілому. Вся структура веб-сторінки може бути представлена

за допомогою DOM (Document Object Model) - організація елементів html, якими ми можемо маніпулювати, змінювати, видаляти або додавати нові [4]. Для взаємодії з DOM застосовується мова JavaScript. Сценарії користувачів визначає бізнес-логіка. Бізнес-сценарії – це набір кроків, які може виконати звиклий користувач. Призначенням цього блоку є захист структури предметної області, контроль або вплив на його операції.

2.2. Розробка базових алгоритмів роботи програмного забезпечення

Для підвищення безпеки і надійності паролів користувачів було реалізовано власний метод визначення надійності паролю. Метод було застосовано як і при ручному вводі символів так і при генерації паролю. А також було розглянуто автоматичне встановлення нагадувань про зміну старого паролбю на новий. Для цього було зреалізовано метод оцінювання необхідності змінювати пароль на новий

2.2.1 Алгоритм оцінювання необхідності зміни пароля

Алгоритм оцінювання формує оцінку щодо необхідності змінювати пароль на новий. Розглянемо на прикладі як працює алгоритм. Для кожного пароля до уваги береться набір критеріїв за якими система виставляє оцінки (по 10ти бальній шкалі). Буде застосовано 5 критеріїв оцінки. У кожного критерія є своя вага, відповідно якій одні критерії оцінки є більш вагомі ніж інші. Відповідно критеріям надаються коефіцієнти для обрахувань. Сума коефіцієнтів дорівнює 1. Ці коефіцієнти встановлює користувач.

У таблиці 2.1 наявні прикладові дані для обрахування необхідності змінювати пароль на новий.

Таблиця 2.1 - Початкові дані для обрахування

Критерій	Коефіцієнт	Прикладова оцінка системи
Складність паролю	0,25	7.5
Довжина паролю	0,15	8
Час останньої заміни	0,25	5.5
Час останнього використання	0,22	6ю5
Загальна кількість використань	0,15	9

Оцінка виставляється в межах від 1 до 10. Обрахування необхідності змінювати пароль на новий обчислюється за формулою 2.1.

$$R = \sum_{i=0}^{n} A_i * K_i \quad (2.1)$$

де

A_i – оцінка системи;

K_i – коефіцієнт критерія;

Таким чином отримуємо результат у балах від 1 до 10.

$R = 7{,}5 * 0{,}25 + 0{,}15 * 8 + 5{,}5 * 0{,}23 + 6{,}5 * 0{,}22 + 0{,}15 * 9 = 7.12.$

З результату дістаємо оцінку необхідності змінювати пароль на новий, яка дорівнює 7,2.

2.2.2 Алгоритм оцінювання складності введеного пароля

Було розроблено алгоритм для оцінювання рівня складності введеного пароля. Рівень складності пароля обчислюється в залежності від обчислюваної «бітової стійкості» [0;4], а також коефіцієнту подібності :

bits ≥ 128 – 4;

128 < bits ≥ 64 – 3;

64 < bits ≥ 56 – 2;

bits < 56 – 1;

порожній пароль – 0.

Оцінка складності, розраховується за формулою 2.2: «бітова стійкість» помножена на коефіцієнт подібності:

$$P = \log(charset)*(\frac{length}{\log(2)})*(1 - k) \quad (2.2)$$

де:

- P – складність;

- log – натуральний логарифм;

- engths – довжина пароля;

- charset – розмір для типу множини.

Існуючі можливі типи подано нижче. Кожен з них береться до уваги, в залежності від того чи вони зустрічаються в рядку:

- малі англійські букви [abcdefghijklmnopqrstuvwxyz];

- заголовні англійські букви [ABCDEFGHIJKLMNOPQRSTUVWXYZ];

- спеціальні символи [~ `! @ # $% ^ & * () -_ + ="];

- цифри [1234567890];

- k – коефіцієнт подібності (розраховується за даними, для користувача) в діапазоні від 0 до 1;.

Коефіцієнт подібності розраховується як підрахунок входження підрядків згенерованої бази фраз у введеному користувачем паролі.

2.3. Прототипування

Інтерфейс – інтеракція,що існує між двома функціональними об'єктами, вимоги до якої визначаються стандартом; сукупність засобів, методів і правил взаємодії між елементами системи [6]. Іншими словами, користувацький інтерфейс, скорочено UI (user interface) – це спосіб, яким користувач виконуєте будь-яку задачу за допомогою будь-якого продукту, а

саме дії, що ним здійснюються і те, що він отримуєте у відповідь [7]. Перше вікно з яким має справу незареєстрований користувач, це вікно реєстрації. Форма реєстрації містить наступні поля: - Login – текстове поле, для введення логіну; - Email address – адреса електронної пошти, поле з валідацією на наявність символу @; - Password/Confirm password – поле головного мастер-пароля ; - Password Level1/level2 – тимчасові паролі на два рівні доступу; - Login Time – кількість невдалих логувань; - Type of encoding – метод шифрування; На рисунку 2.2 зображено протопи вікна реєстрації, з усіма наявними полями та кнопками.

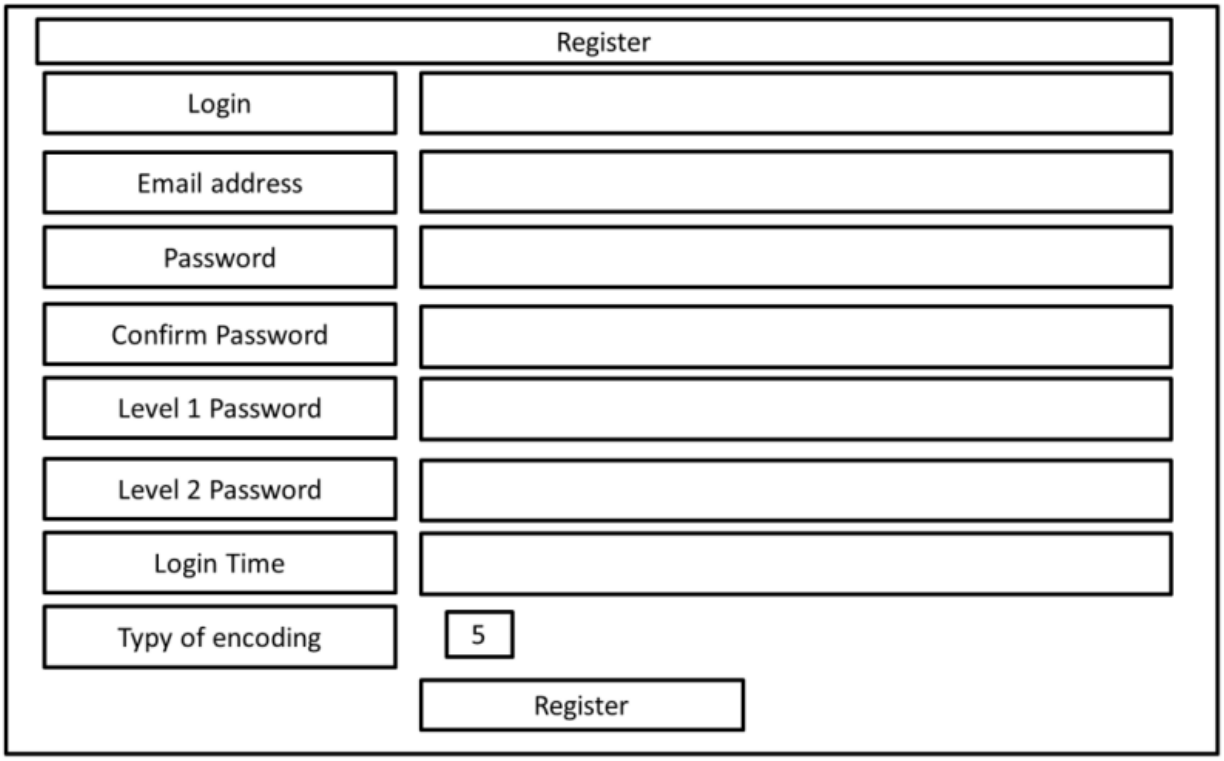

Рис.2.2. Прототип екранної форми «Реєстрація користувача»

Після реєстрації, користувач повинен увійти до системи, щоб мати змогу користуватися сервісом. Процес входу має автентифікацію через пошту. Первинне вікно для входу (рисунок 2.3) виглядає наступним чином:

- login - основна інформація про користувача;
- password - пароль користувача;
- remevber me – чекбокс, що відповідає за збереження сесії;

- forgot password – посилання для відновлення мастер-пароля.

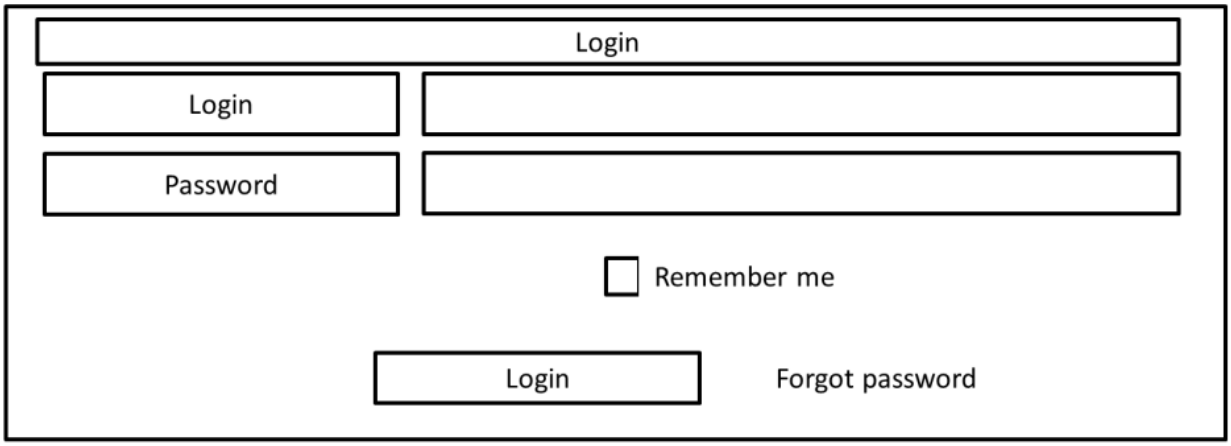

Рис.2.3. Прототип екранної форми «Домашня сторінка авторизації»

Після успішного залогування користувач потрапляє на головне вікно додатку. Прототип вікна зображено на рисунка 2.4.

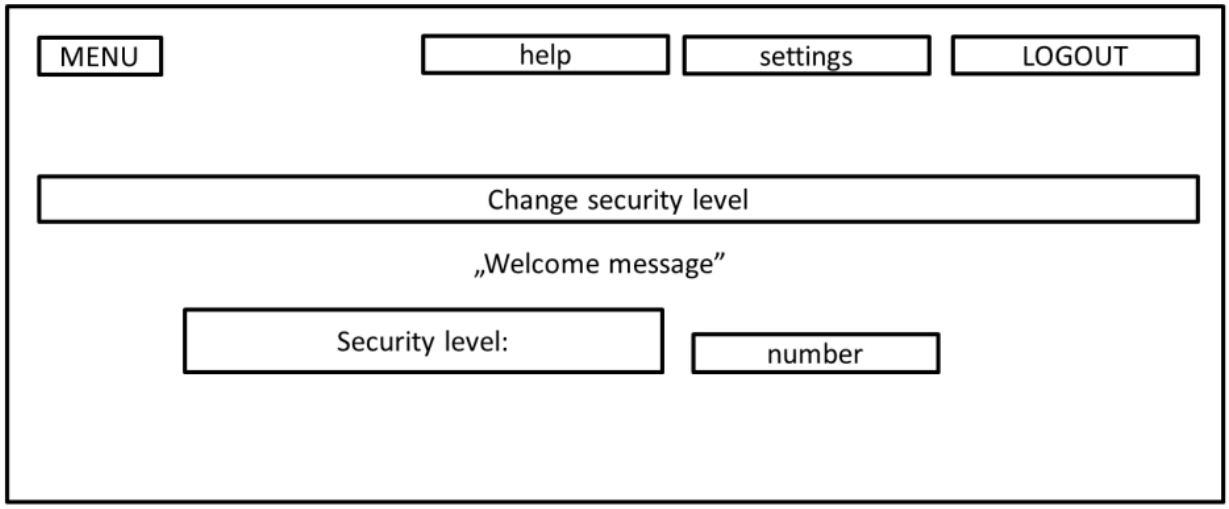

Рис.2.4. Прототип екранної форми «Головна сторінка»

На рисунку 2.5 відображений процес зміни рівня безпеки. Для цього користувачеві знадобиться пароль, що відповідає тому рівню, на який хоче перейти користувач.

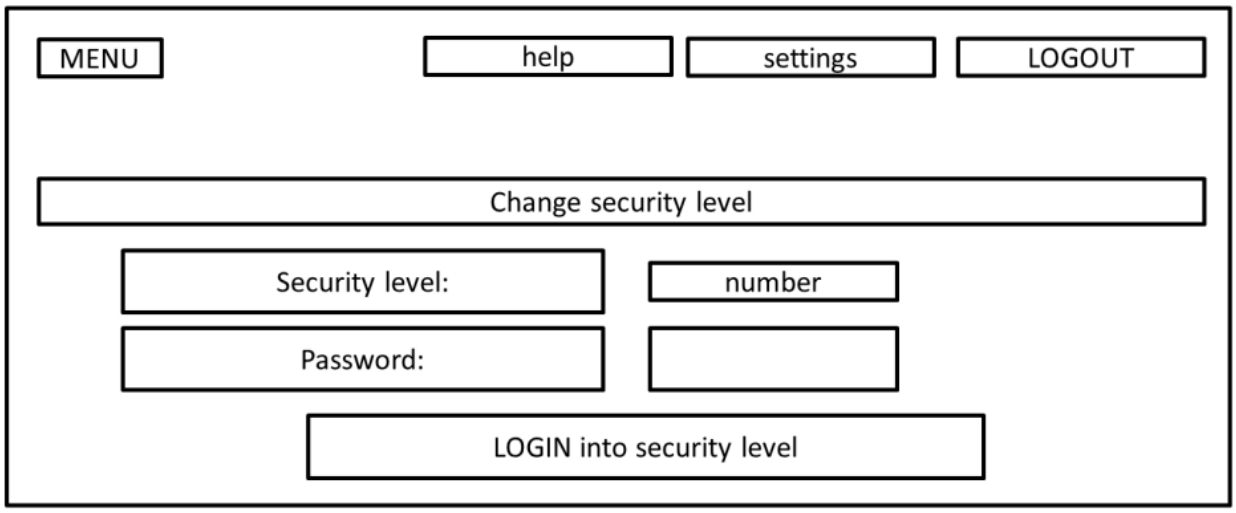

Рис.2.5. Прототип екранної форми «Логування на другий рівень безпеки»»

На другому рівні безпеки користувач може переглядати або редагувати вже додані паролі, а також додавати нові. На рисунку 2.6 відображено прототип вікна додавання/редагування паролю.

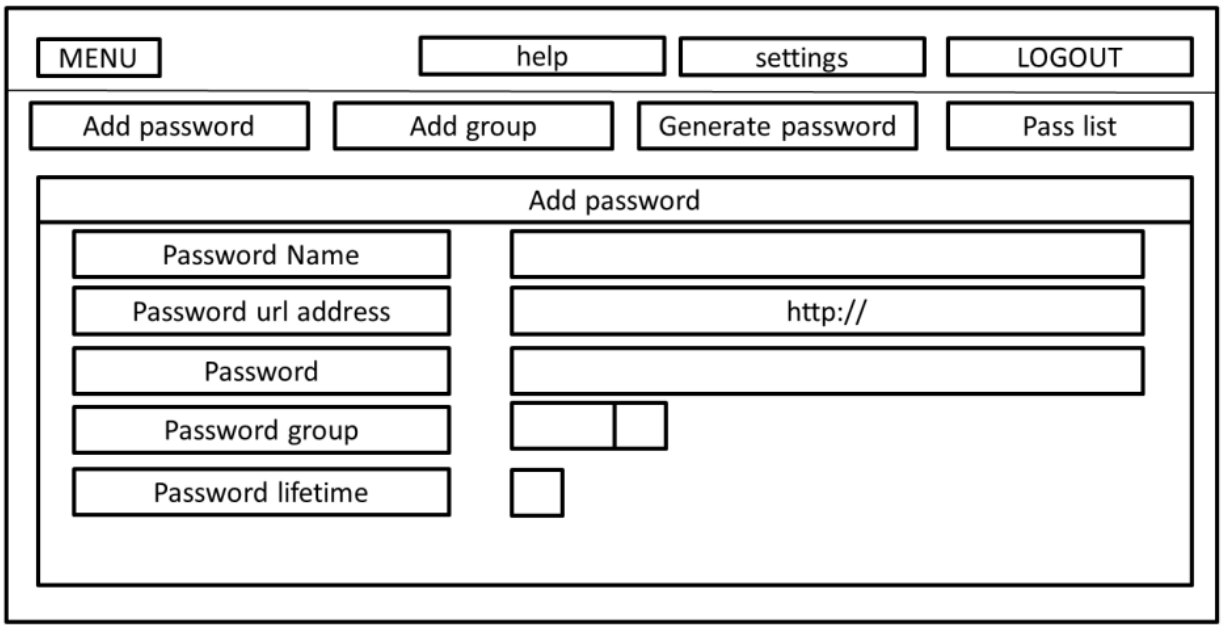

Рис.2.6. Прототип екранної форми «Додавання/редагування паролю»

Як тільки користувач залогується до другого рівня безпеки, йому доступноою стає можливість переглядати історію своїх змін, та повернутися до якоїсь конкретної точки, щоб відмінити всі нові зроблені зміни. Прототип вікна історії зображено на рисунку 2.7.

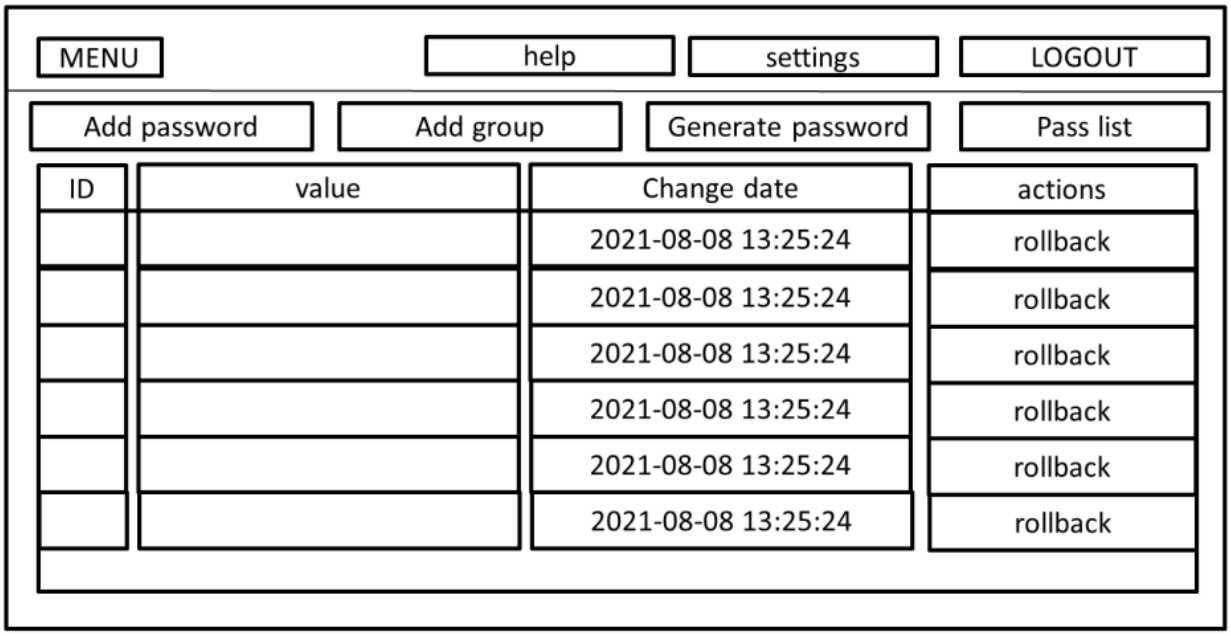

Рис.2.7. Прототип екранної форми «Історія змін»

Під час відкату до попередньої версії паролю, до історія додається новий запис з заголовком про відновлення.

Висновки до розділу II

В цьому розділі було проаналізовано інформаційне забезпечення системи та розглянуто початкову версію інтерфейсу. Розглянуто загальну функціональну структурну модель, а також розроблено власні методи оцінювання складності паролю, а також алгоритм сповіщення користувачів для зміни старого паролю на новий. Також було розглянуто принципи побудови прототипів інтерфейсу, що побудовані на засадах UI/UX, що дозволяє зробити інтерфейс інтуїтивно-зрозумілим для користувача та пришвидшить його роботу з додатком [8].

РОЗДІЛ ІІІ
РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ КЕРУВАННЯ ПАРОЛЯМИ

3.1. Обґрунтування вибору засобів розробки

Для реалізації системи необхідно обрати технології, які в повній мірі зможуть задовольнити технічні та функціональні вимоги модулів. Оскільки веб-додаток має мікросервісну архітектуру, то варто розглянути окрему частини бекенд та фронтенд. Для бекенду основним фреймворком було обрано Spring Boot. Використовуючи Spring Boot можна досягти таких самих результатів, що і на чистій Java, але набагато меншими зусиллями і меншою кількістю коду.

Мета даного фреймворку - щоб програміст зосередився на вирішенні бізнес завдання замість того, щоб витрачати час на налаштування коду. Серед основних переваг фреймворку можна відзначити: - легкість запуску: пакет проектів має вбудований сервер та інші необхідні компоненти, необхідні для запуску програми; - автоматична конфігурація: для запуску основної програми не потрібна додаткова конфігурація; - швидкість: спрощення створення додатків за допомогою Spring Boot, що призводить до швидшого та простішого процесу розробки.

Всі налаштування за додатком можна помістити в файл налаштувань замість java класів. Як правило, файл - application.properties лежить в папці resources. Але і це можна поміняти і винести настройки на віддалений сервер наприклад або в інший файл [9]. Spring Boot при запуску завантажує файл залежностей бібліотек. Якщо це Maven то pom.xml. Він просканує всі залежності, які підключені і після використовує файл проперти, щоб підставити настройки в необхідні залежності. Візьмемо для прикладу підключення бази даних.

Спочатку вказуються в файлі pom.xml залежності, що будуть використовуватись наприклад spring-data-jpa. Spring «знає», що дана бібліотека заснована на роботі з базою даних. При запуску він буде шукати у файлі application.properties налаштування до бази (логін, пароль). Якщо він їх не знайде - то не запуститься і видасть відповідне повідомлення. Spring Boot також включає в себе вбудований контейнер сервлетів (по замовчуванні Tomcat) що дозволяє запускати веб додатки як звичайні java програми: використовуючи jar файл. Або якщо ще простіше кажучи: просто натискаючи на зелену стрілку в IDE як Ви це робили з “Hello world” додатками.

Фактично Spring Boot має автоматичну настройку майже для всіх Spring модулів, які ми зараз розберемо. Для розробки фронтенду основним фреймворком було обрано React. React - це бібліотека JavaScript, яка використовується для створення призначеного для користувача інтерфейсу. React був створений компанією Facebook, а перший реліз бібліотеки побачив світ у березні 2013 року. Поточної версій на даний момент (жовтень 2020 року) є версія React v17.0.

Спочатку React призначався для вебу, для створення веб-сайтів, проте пізніше з'явилася платформа React Native, яка вже призначалася для мобільних пристроїв. React представляється ідеальний інструмент для створення масштабованих веб-додатків (в даному випадку мова йде про фронтенді), особливо в тих ситуаціях, коли додаток являє SPA (односторінкове додаток). React відносно простий в освоєнні, має зрозумілий та лаконічний синтаксис.

Відмінною рисою бібліотеки є концентрація на компонентах - ми можемо створити окремі компоненти і потім їх легко переносити з проекту в проект. Ще одна особливість React - використання JSX. JSX представляє комбінацію коду JavaScript і XML і надає простий і інтуїтивно зрозумілий спосіб для визначення коду візуального інтерфейсу. Вся структура веб-сторінки може бути представлена за допомогою DOM (Document Object Model) - організація елементів html, якими ми можемо 32 маніпулювати,

змінювати, видаляти або додавати нові. Для взаємодії з DOM застосовується мова JavaScript.

3.2. Особливості програмної реалізації програмного забезпечення та бази даних

3.2.1. Розробка алгоритмів шифрування паролів

SHA512 - це хеш-функція, яка обчислює хеш-значення для набору даних. Хеш обчислюється за спеціальним алгоритмом без додаткових даних. Це робить можливими деякі атаки - особливо при використанні таблиць. Через це під час перемішування додають сіль. Сіль - це послідовність байтів / символів, які додаються до паролю перед хешуванням. Важливо змінювати сіль під час зміни пароля [10].

Сіль зберігається в базі даних. Для кращого захисту паролів ви можете додати перець. Це також випадкова послідовність байтів / символів - але вона зберігається всередині програми, а не в базі даних. Ви можете зберігати перець у коді вашої програми або у файлі конфігурації. Фрагмент коду функції представло на лістингу 3.1.

Повний лістинг класу, що відповідає за шифрування SHA512 подано в додатках. HMAC - це код автентифікації повідомлень на основі хешу. Він обчислює хеш даних, але з використанням секретного ключа. Це означає, що тільки власник ключа може обчислити хеш [11]. HMAC має багато переваг, таких як велика стійкість проти атак.

Схема HMAC представлена на рисунку 3.1.

Лістинг класу, що відповідає за шифрування методом HMAC подано в додатках.

```
private static String calculateSHA512(String text) {
    try {
        MessageDigest md = MessageDigest.getInstance("SHA-512");
        byte[] messageDigest = md.digest(text.getBytes());
        BigInteger no = new BigInteger(1, messageDigest);
        String hashtext = no.toString(16);
        while (hashtext.length() < 32) {
            hashtext = "0" + hashtext;
        }
        return hashtext;
    }
    catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(e);
    }
}
```

Лістинг 3.1. Фрагмент коду реалізації алгоритму SHA512

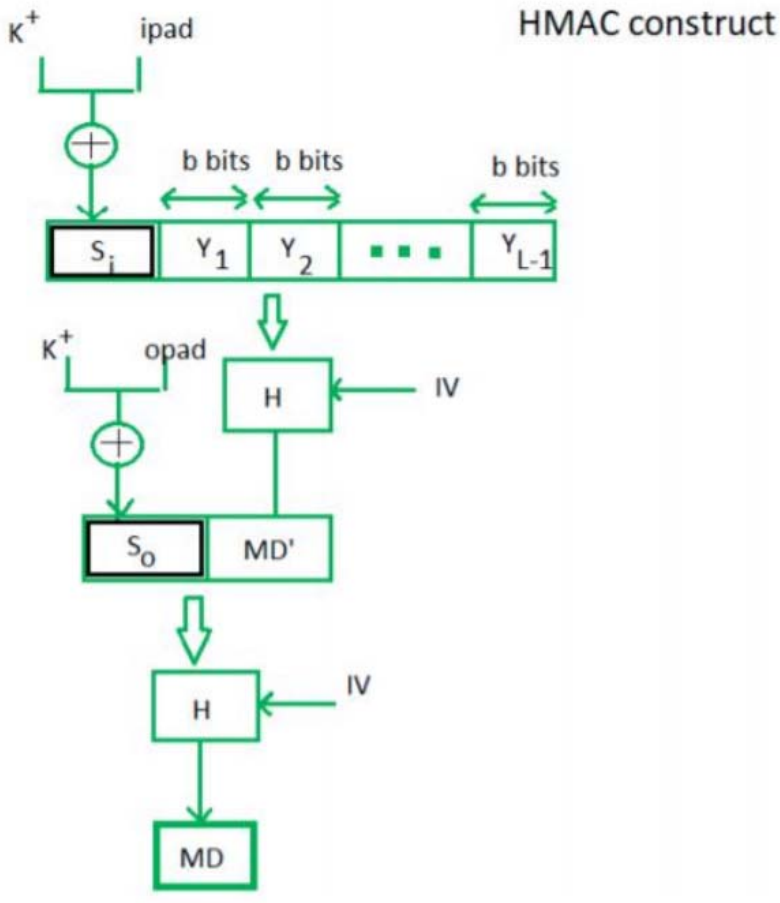

Рис. 3.1. Схема HMAC

Серед переваг цього алгоритму можна зазначити наступні. HMAC ідеально підходять для високопродуктивних систем, таких як маршрутизатори, завдяки використанню хеш-функцій, які швидко обчислюються та перевіряються на відміну від систем із відкритим ключем. Цифрові підписи більші за HMAC, проте HMAC забезпечують порівняно вищий рівень безпеки. HMAC використовуються в адміністраціях, де системи відкритих ключів заборонені. Якщо говорити про недоліки, то HMAC використовує спільний ключ. Якщо ключ відправника або одержувача пошкоджено, зловмисникам буде легко створювати несанкціоновані повідомлення.

3.2.2. Розробка структури бази даних

На першому етапі було запроектовано базу даних, яка буде доречною для збереження даних для входу користувачів та їх паролі. Структура бази даних показана на показаній схемі EER на рисунку 3.2 [12].

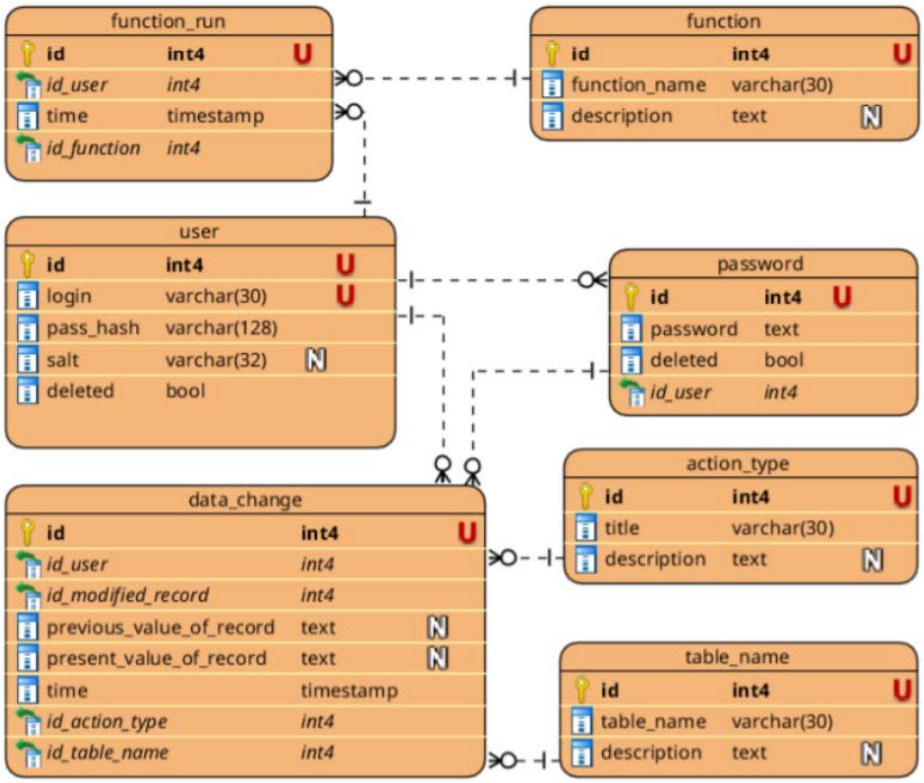

Рис. 3.2. Схема структури бази даних

Подана структура бази даних, створена командами мови SQL [13-15]. Код подано в додатках. Всі паролі в базі даних зберігаються у зашифрованому вигляді. Це означає, що користувач створює мастер-пароль для шифрування паролів, що зберігаються в базі даних. За допомогою такого головного пароля будуть шифруватися та розшифровуватися паролі користувачів. Мастер-пароль вводиться користувачем лише один раз після входу і зберігається у змінній під час роботи програми.

3.2.2. Розробка функціональних модулів програмного забезпечення

Проектування моделі веб-додатку є однією з найважливіших частин процесу розробки програмного забезпечення. Правильно розроблений додаток є швидшим та легшим у реалізації, що означає скорочення робочого часу призначеного для імплементації. Ретельне проектування також дозволяє виявити багато помилок перед тим, як почати писати програму, і дозволяє їх легко виправити. Дизайн веб-сайту включає визначення нефункціональних функціональних вимог, точне визначення структури бази даних та розробку відповідних діаграм для сприяння у впровадженні.

Першим етапом проектування веб-додатку було визначення функціональних вимог [16], яким він буде відповідати. Завдяки цьому вдалося точно визначити завдання веб-сайту, що створюється. Його основна функціональність включає:

- вибір способу шифрування, під час реєстрації нового користувача: SHA512 або HMAC;
- зміна пароля для входу;
- функціонал додавання паролів користувачів;
- функціонал перегляду збережених паролів;
- розшифрування пароля, якщо користувач просить його відобразити;
- реєстрація спроб логування користувачами (запис таких даних, як час входу, результат входу (успішно, не вдалося), IP-адресу користувача);

- представлення користувачеві даних про останній успішний і невдалих вхід;
- збереження інформації про кількість наступних неправильних пробних входів користувачів;
- у випадку, якщо користувач не зміг увійти принаймні два рази, час очікування входу користувача збільшується до 5 секунд;
- у випадку, якщо користувач не зміг увійти щонайменше тричі встановлюється продовжиення часу перевірки входу користувача до 10 секунд;
- у випадку, якщо користувачу не вдався увійти як мінімум чотири рази, обліковий запис користувача блокується на 2 хвилини;
- у випадку успішного входу в систему число неправильних входів скидається до нуля;
- функціонал перевірки IP-адреси, якою користується користувач – підрахунок правильних та неправильних спроб входу з вибраної IP-адреси;
- у випадку, якщо користувачі не змогли увійти щонайменше чотири рази з однієї і тої самої адреси - IP адреса блокується назавжди;
- у випадку успішного входу з конкретної IP-адреси, скиньте кількість наступних неправильні логіни з цієї адреси;
- можливість зняти блокування IP-адреси.

Таким чином, одним з перших кроків реалізації вищеописаних вимог було створення моделей всіх сутностей в додатку: «Пароль», «Користувач», «Лог», «Операція зміни даних», «Виконана функція». Кожна така сутність позначена анотацією Entity. Клас типу Entity вказує на клас, який на абстрактному рівні співвідноситься з таблицею в базі даних. Кожен об'єкт, інстанційований цим класом, вказує модель самої таблиці, що містить інформацію останнього. Поля класу позначаються анотаціями @Column(), що відповідає назвам стовпців у таблиці. Лістинг кожної з поданих моделей подано в додатках. Всі процеси, що виконуються веб-сайтом, можуть бути

описані за допомогою графічних позначень, які називаються діаграмами BPMN (Bussiness Process Model and Notation). Вони однозначні, зрозумілі та гнучкі, що полегшує розуміння того, як працюють певні функціональні можливості [17-19]. На рисунку 3.3 показаний процес додавання паролів.

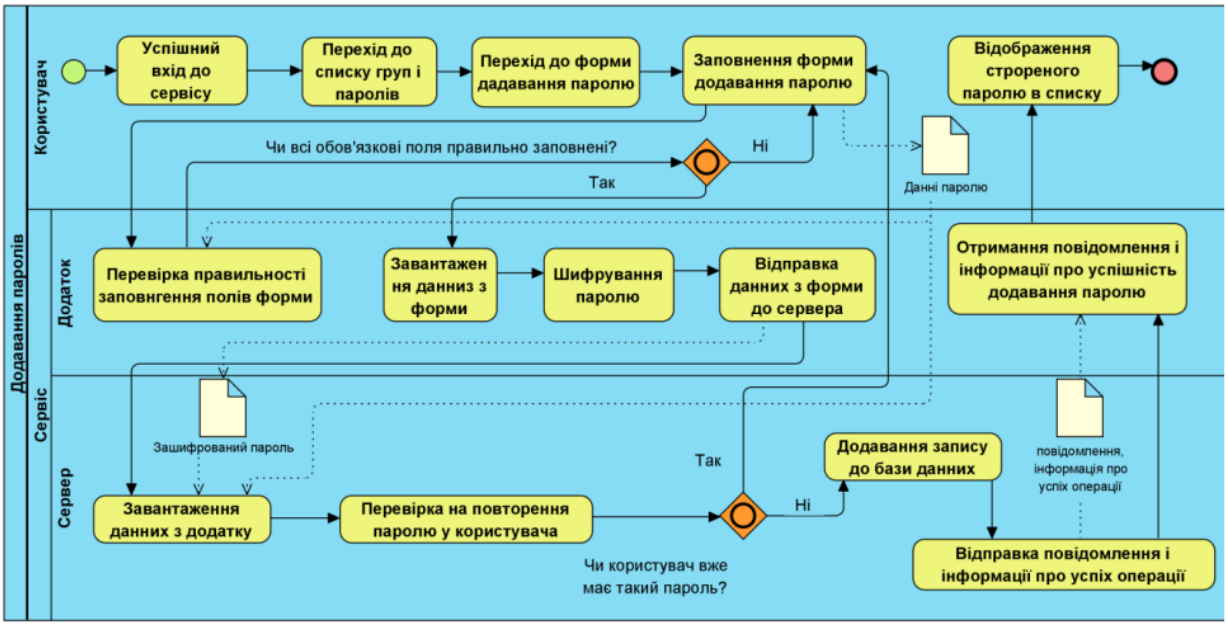

Рис.3.3. Схема алгориму додавання паролів до бази даних

Наступним кроком було визначення нефункціональних вимог [20], тобто таких, які не стосуються функціональності веб-додатку, але визначають спосіб його роботи:

• веб-сайт під’єднується до бази даних SQL;

• для користування веб-сайтом потрібне з’єднання з Інтернетом;

• веб-додаток доступний у браузері,

• паролі в базі даних зберігаються у зашифрованому вигляді,

• пароль для входу та паролі доступу для кожного рівня безпеки зберігаються у хеш-формі, ці паролі є ключами шифрування на індивідуальному рівні безпеки;

• веб-сайт використовує алгоритм SHA-512 як хеш-функцію;

• веб-сайт використовує алгоритм AES-256 для шифрування паролів.

У аплікації реалізація вищеописаних вимог виглядає наступним чином. Завдяки контролерам, серверна частина аплікації комунікує з клієнтською

[21]. Відповідно існує чотири контролери: UserController, PasswordController, FunctionRunController та DataChangeController. Контролер позначається в коді анотацією @Controller з вказанням методу передачі даних, наприклад: @PostMapping, @GetMapping тощо. Повний код кожного з контроллерів подано у додатку Д. Головна логіка програми реалізована у сервісних класах, які позначені анотацією @Service. Сервісний клас, що реалізує логіку роботи з паролем і пов'язаний з контролером «PasswordController» подано у додатках.

3.3. Тестування програмного забезпечення

З метою перевірки правильної роботи розробленого веб-сайту були проведені тести. Вони дають змогу виявляти помилки та виправляти їх ще до виходу остаточної версії. Тестування також дозволяє помітити будь-які незручності та труднощі, пов’язані з використанням веб-сайту. Зазвичай тести проводяться з точки зору конкретної особи (часто тієї, для кого створюється веб-сайт) і стосуються конкретного модуля. Ще однією перевагою є той факт, що частину веб-сайту можна протестувати до закінчення всього вихідного коду. Black-box тестування - це функціональний і нефункціональне тестування без доступу до внутрішньої структурі компонентів системи. Метод тестування «чорного ящика» - процедура отримання та вибору тестових випадків на основі аналізу специфікації (функціональної або нефункціональної), компонентів або системи без посилання на їх внутрішній устрій [18].

За допомогою Black Box Testing можна протестувати будь-який додаток, просто зосередившись на входах і виходах, не знаючи його внутрішньої реалізації коду.

Існує багато видів тестування чорного ящика, але найбільш важливими є наступні.

Функціональне тестування - цей тип тестування чорного ящика пов'язаний з функціональними вимогами системи; це роблять тестери програмного забезпечення.

Нефункціональне тестування. Цей тип тестування чорного ящика пов'язаний не з тестуванням конкретної функціональності, а з нефункціональними вимогами, такими як продуктивність, масштабованість, зручність використання.

Регресійне тестування - Регресійне тестування проводиться після того, як виправлення коду, оновлення або будь-яке інше обслуговування системи для 40 перевірки того, що новий код не торкнувся існуючий код. Веб-додаток було протестовано в браузері Google Chrome, версія 63.0.3239.84, 64-розрядна версія, в системі Windows 10 Education.

Правильність роботи функції, що дозволяє реєстрацію користувача, була перевірена в тесті, проведеному за сценарієм, представленим у таблиці 3.1.

Таблиця 3.1 – Тестовий сценарій реєстрації

Користувач	Новий користувач
Ціль тесту	Створення нового облікового запису
Дії користувача	1. Запуск програми в браузері. 2. Перехід до вікна реєстрації. 3. Заповнення реєстраційної форми. 4. Натискання кнопки реєстрації.
Очікуваний результат	Створення облікового запису
Альтернативний результат	Помилка реєстрації через неправильні дані

Хід тесту виглядає наступним чином. Після запуску програми та переходу до вікна реєстрації відображалася реєстраційна форма, показана на рисунку 3.4

На рисунку 3.5 показано процес реєстрації, який успішно завершено, і користувач одночасно входить на веб-сайт, тоді як на рисунку 3.6 показано ефект введення неправильних даних у форму реєстрації.

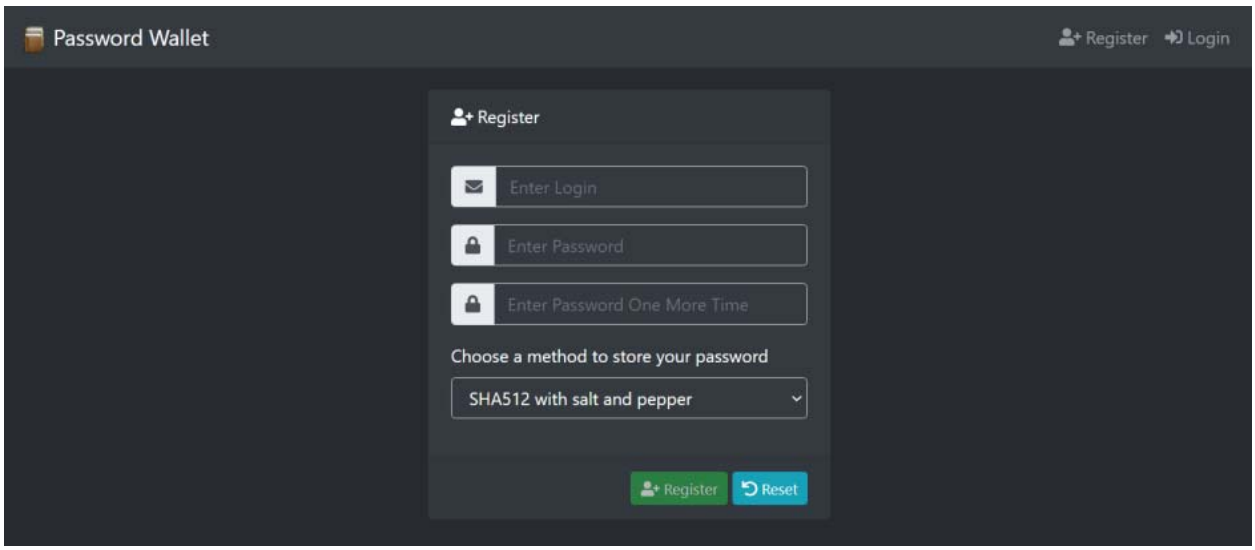

Рис.3.4. Реєстрація користувача

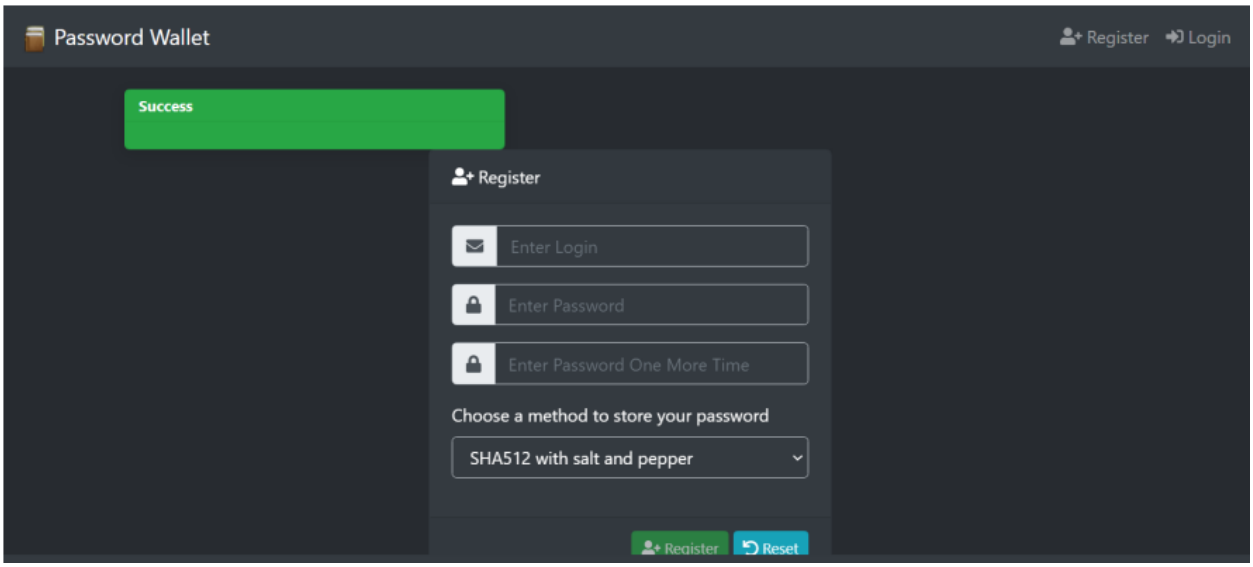

Рис.3.5. Реєстрація користувача (правильні дані)

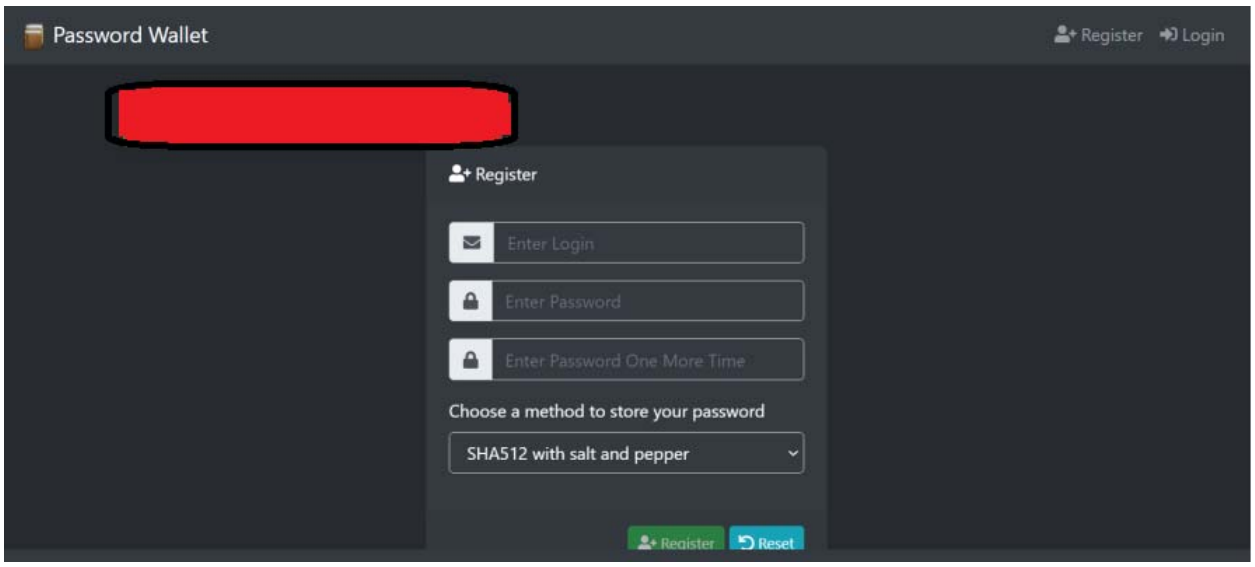

Рис.3.6. Реєстрація користувача (неправильні дані)

Робота функції, що дозволяє користувачеві увійти на веб-сайт, була перевірена в тесті, проведеному за сценарієм, представленим у таблиці 3.2.

Таблиця 3.2 – Тестовий сценарій логування

Користувач	Користувач, що має акаунт в системі
Ціль тесту	Вхід до аплікації
Дії користувача	1. Запуск програми в браузері. 2. Перехід до вікна входу. 2. Заповнення форми входу. 3. Натиснення кнопки входу
Очікуваний результат	Успішний вхід на сайт
Альтернативний результат	Відображається повідомлення про надання неправдивих даних

Після запуску програми та переходу до вікна входу в систему відображалася форма входу, показана на рисунку 3.7.

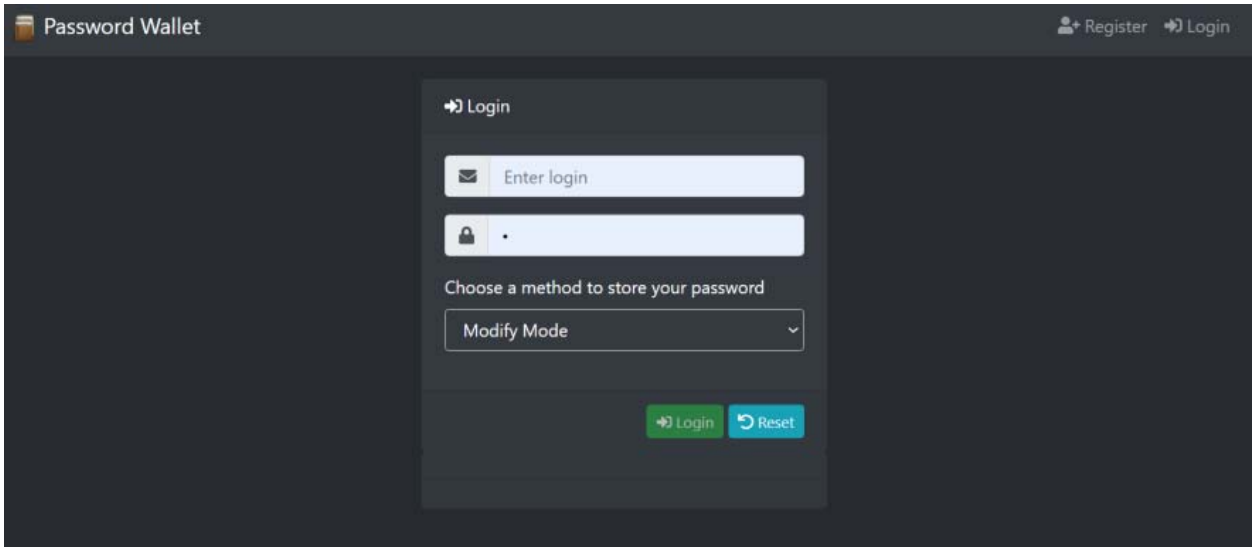

Рис.3.7. Форма логування

Якщо будуть введені правильні дані, користувач увійде до системи та перейде у вікно головного меню, як показано на рисунку 3.8.

Введення неправильних даних призведе до помилки входу та відображення відповідного повідомлення, яке було представлено на рисунку 3.9.

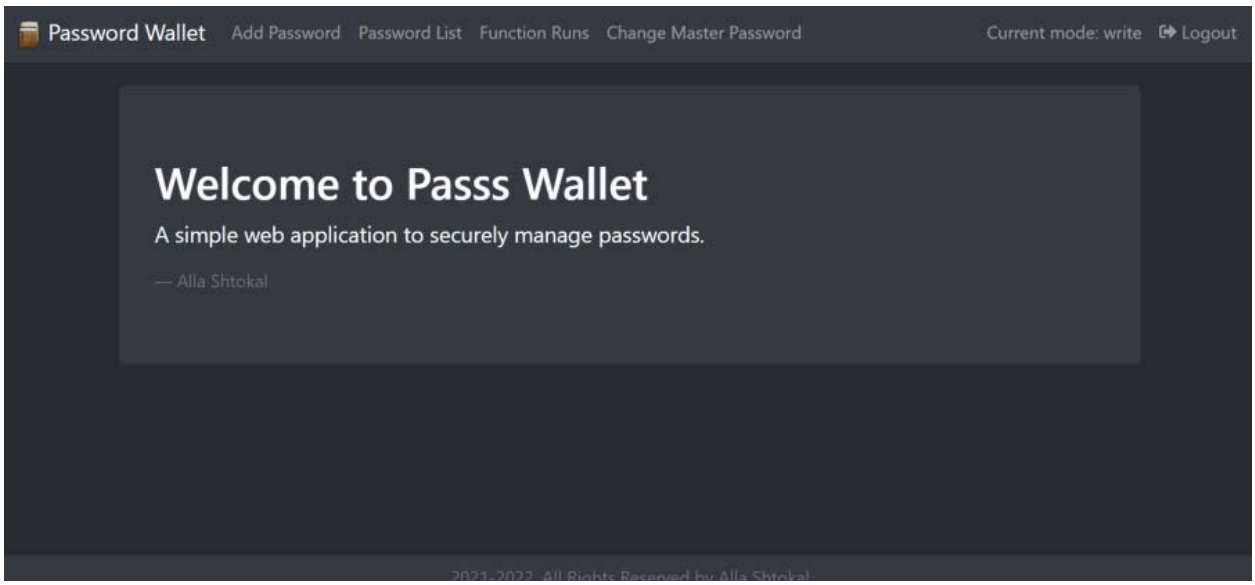

Рис.3.8. Успішний вхід

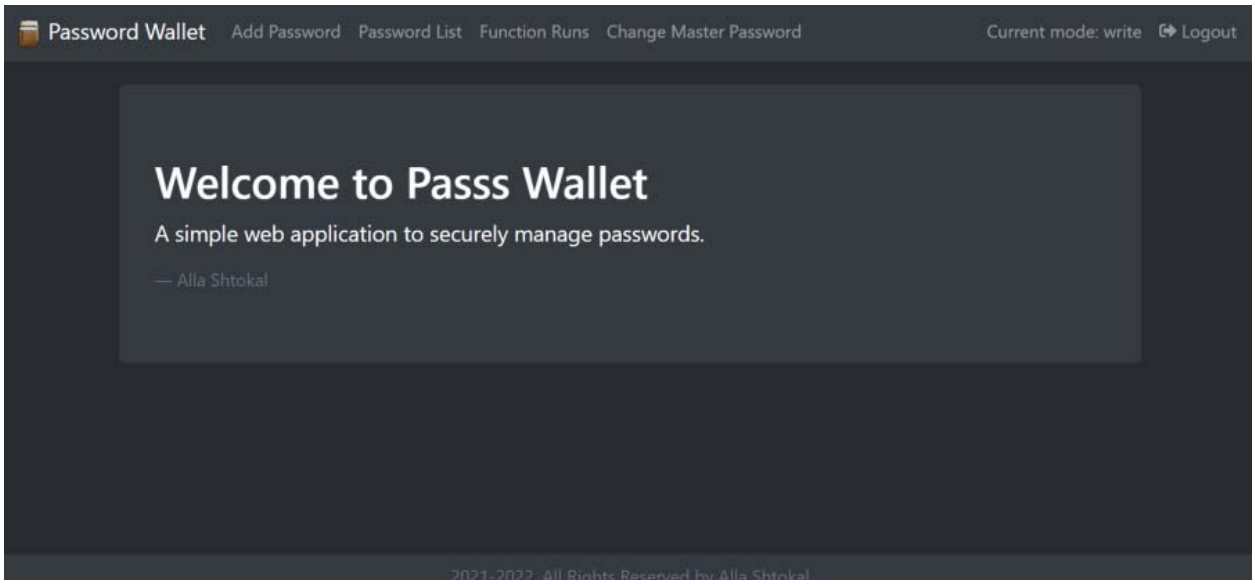

Рис.3.9. Невдалий вхід

Наступною перевіреною функціональністю був механізм відображення історії змін паролів, а також можливість відкату до попереднього кроку. Це дозволяє у випадку небажаних змін повернутися до попереднього стану. Ця функціональність була перевірена на основі сценарію, описаного в таблиці 4.3.

Таблиця 3.3 – Тестовий сценарій тестування історії зміни паролів

Користувач	Користувач, що увійшов до системи
Ціль тесту	Відображення історії змін та повернення до попередніх станів
Дії користувача	1. Перехід у вікно зі списокм паролів 2. Натиснення кнопки з переглядом історії для конкретного вибраного паролю 4. У новому вікні з історією натиснення іконки переходу до попереднього стану
Очікуваний результат	Успішний перехід до попереднього стану
Альтернативний результат	Відображення нового запису в історії змін про повернення всіх змін до вибраного стану

Залогований користувач повинен перейти у вікно зі списком усіх паролів, та обрати той, для якого хоче переглянути історію змін. Натиснувши на іконку поруч із відповідним рядком, рисунок 3.10.

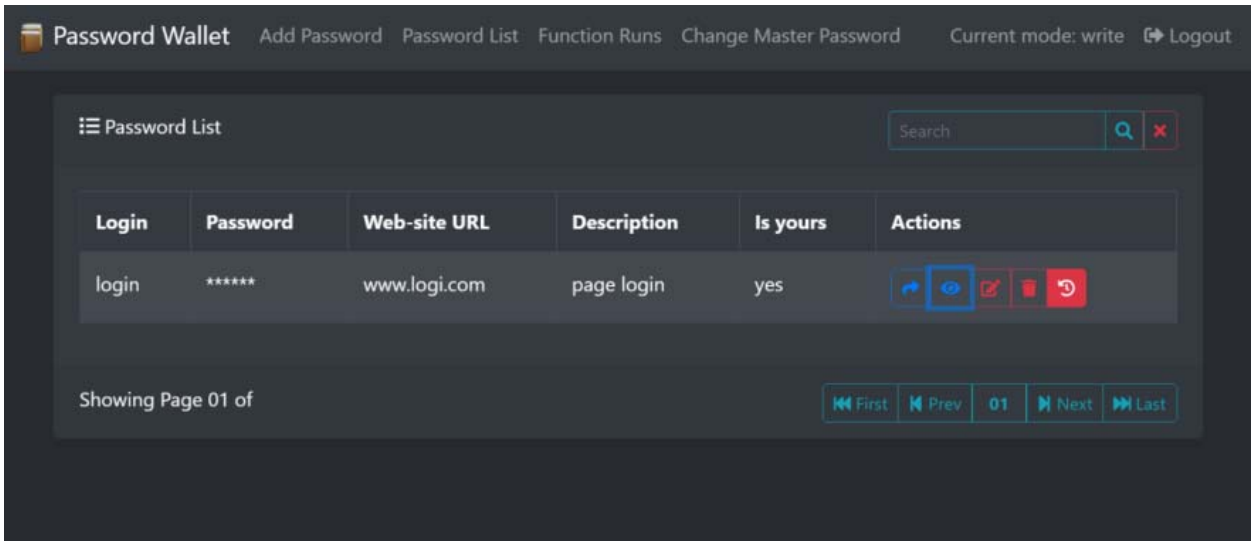

Рис.3.10. Екран зі списком паролів

Після натиснення кнопки з іконкою «Історія» користувач потрапляє у нове вікно з історією. На рисунку 3.11 показано історію змін паролю.

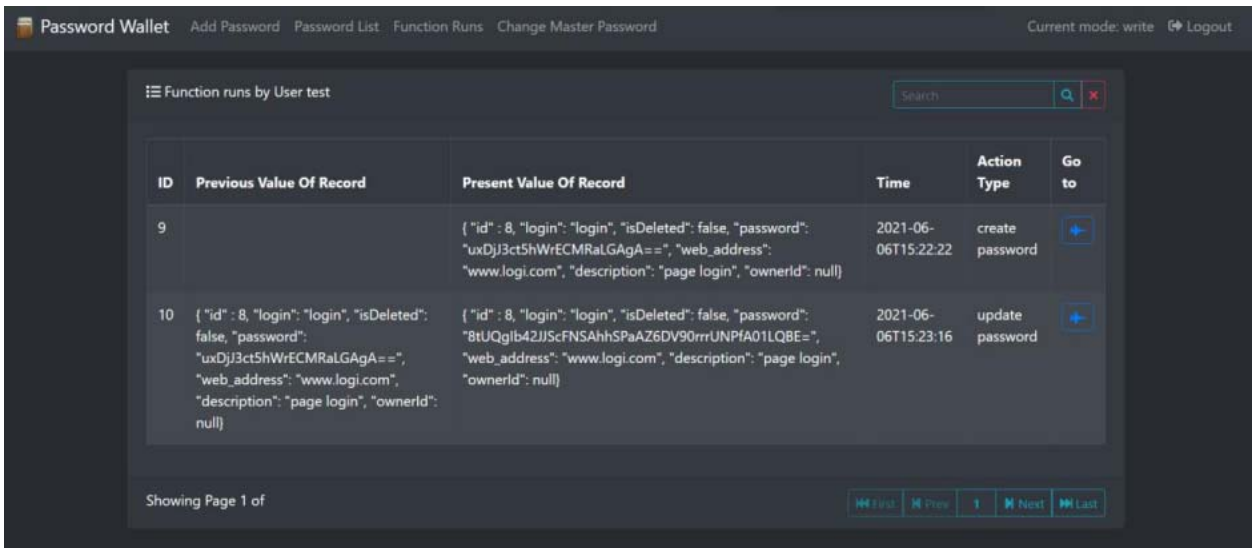

Рис.3.11. Екран зі відображенням змін паролю

Щоб повернутися до вибраного попереднього стану паролю потрібно натиснути на іконку в правому стовпці.

На рисунку 3.12 показано ефект вдалої спроби повернення до попередньої версії паролю.

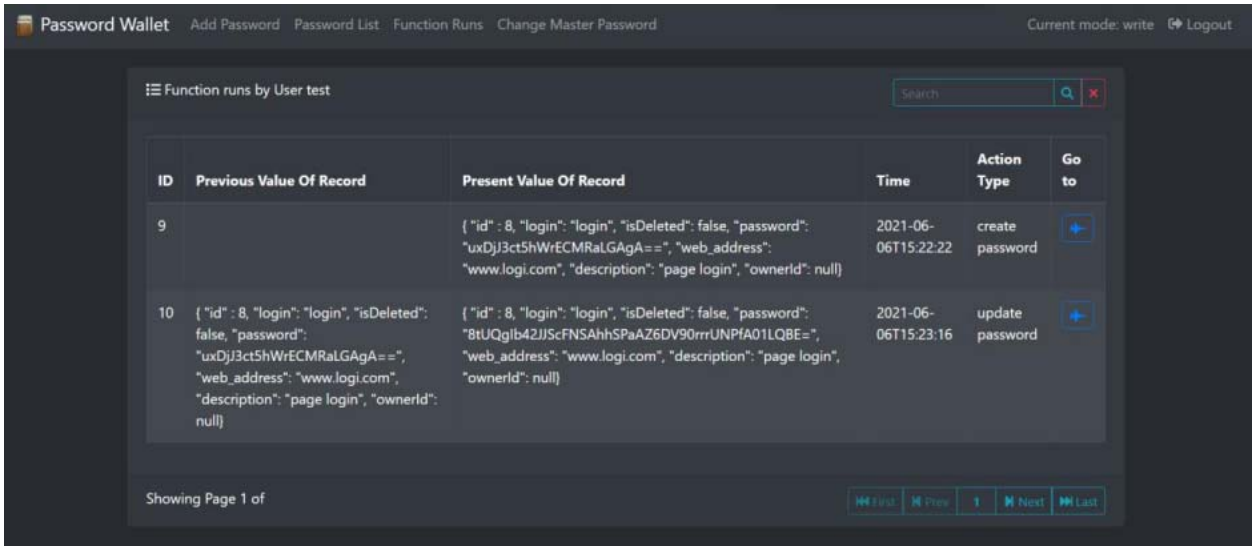

Рис.3.12. Вдала спроба повернення для попереднього паролю

Для того, щоб перевірити правильність функціональних можливостей додавання паролів, було проведено тест. Сценарій тесту подано у таблиці 3.4.

Таблиця 3.4 – Тестовий сценарій додавання паролів

Користувач	Користувач, що увійшов до системи
Ціль тесту	Додавання паролю
Дії користувача	1. Перехід до вікна зі списком паролів та груп. 2. Перехід до вікна додавання пароля. 3. Заповнення відповідних полів. 4. Натиснення кнопки, що додає пароль.
Очікуваний результат	Успішне додання паролю
Альтернативний результат	Відображення повідомлення про неправильні дані

Для того, щоб створити пароль, потрібно перейти у вікно, що містить список паролів, а потім вибирати опцію додавання пароля у верхній панелі. В результаті користувач перенаправляється у вікно, представлене на рисунку 3.13.

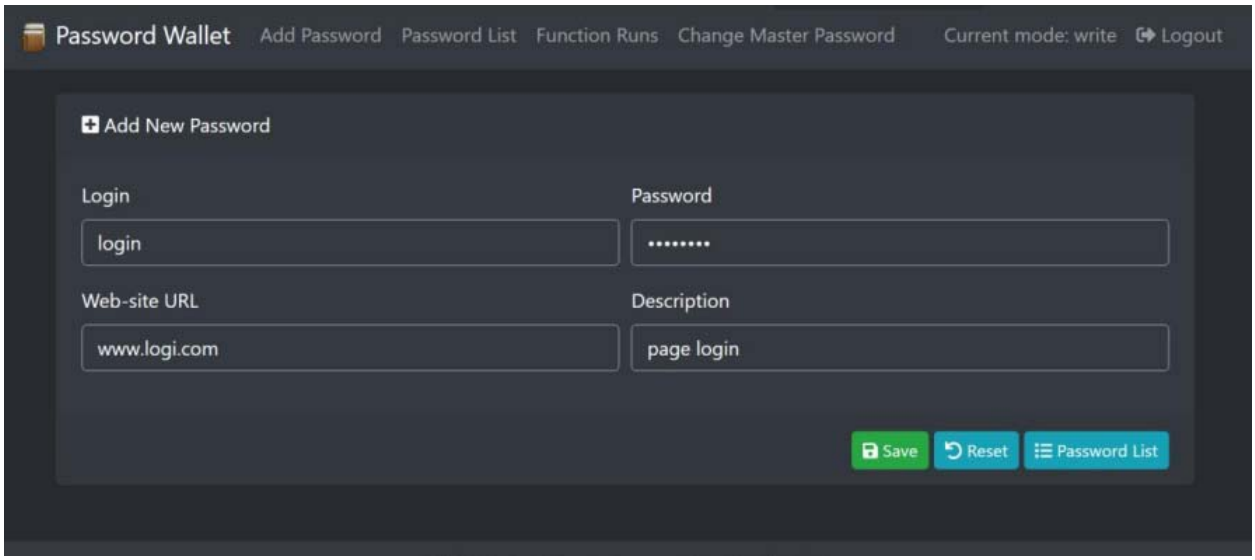

Рис.3.13. Вдала спроба повернення для попереднього паролю

Додаючи пароль, необхідно вказати його зміст та назву веб-сайту, якому він призначений. Адреса веб-сайту не є обов’язковою.

Інструкція користувача призначена для швидкого та лаконічного опису роботи з програмним продуктом. Інструкція дозволяє скоротити час

ознайомлення користувача з системою, а також у разі потреби може допомогти вирішити конкретну проблему.

Користуватися системою можуть лише зареєстровані та авторизовані користувачі. На головному екрані належи створити новий акаунт або ж увійти вже з існуючого на відповідний рівень безпеки Для того, щоб почати управління своїми паролями користувач може:

1. Перейти з домашньої сторінки на сторінку своїх паролів.

2. Додати новий пароль натиснувши кнопку «Додати пароль»

3. Після успішного додавання паролю його можна відредагувати, переглянути або видалити. Усі дії проведені над паролями логуються.

4. Щоб переглянути історію змін потрібно перейти у головне меню і натиснути "Історія"

5. У вікні з історією можна повернутися до конкретного запису в історї записів.

6. Користувач, що не увійшов до другого рівня безпеки може лише переглядати паролі, але не може їх редагувати чи видаляти.

7. На першому рівні безпеки доступна можливість поділитися паролем з іншим користувачем.

Якщо якась дія доступна на першому рівні безпеки, то це означає, що вона так само доступна і на другому рівні безпеки. Паролі можна додавати в групи. Можна переглянути список всіх паролів або паролів, що належать до конкретної групи. Для пошуку потрібно на головній сторінці біля пошукової стрічки вибрати групу, за якою буде виконуватись пошук і ввести ключове слово.

Якщо користувач вже має додані паролі та виконував над нимиоперації усування, вносив зміни, то у правому верхньому куті буде відображатися кнопка «Історія». Після натискання на кнопку користувач потрапляє до вікна з відображеною історією змін, де може зробити відкат до вибраної ним зміни. Таким чином, вся базова інформація буде надана користувачеві перед початком роботи з додатком.

Висновки до розділу III

У третьому розділі було обрано технологію реалізації компонентів системи та розроблено алгоритми шифрування паролів. Оскільки для додатку було обрано мікросервісну архітектуру, то бекенд та фронтенд розроблялися окремо. Для бекенду було обрано фреймворк написаний мовою Java – Spring Boot,а для фронтенду – React. У рамках розробки алгоритму робота програмного додатку, передбачено функціонал додавання паролів, їх редагування, поширення, а також перегляд історії змін паролів і вразі потреби відкат до конкретного моменту часу.Розроблено файл конфігурації програмного додатку, що регулює налаштування програми та бібліотеки, обрано необхідні поля конфігурації, що забезпечать найбільш зручне та водночас гнучке використання програмного застосунку. Також було розроблено структуру бази даних веб-додатку. Для потреб веб додатку було прийнято рішення використовувати реляційну базу даних MySQL. Було визначено основні функціональна та нефункціональні вимоги веб-додатку, а також представлено діаграму процесу додавання пар

Розглянуто види та методики тестування, обрано методику тестування. Було проведено опис роботи головних сценаріїв програмного забезпечення. Додаток для кожного з розглянутих сценаріїв працює правильно. Розроблено інструкцію користувача

ВИСНОВКИ ТА ПРОПОЗИЦІЇ

У результаті виконання кваліфікаційної роботи було розроблено функціонал програмного забезпечення для керування паролями та їх шифрування. Детальний аналіз предметної області показав, що проблема потребує вирішення. Було виконано аналіз відомих аналогів проекту. Було виявлено недоліки, які врахуються у кваліфікаційній роботі: можливість відновлення мастер-пароля, автоматичне перешифрування паролів з встановленою періодичністю, використання тимчасового паролю для входу, а також двофакторної автентифікації, забезпечення історії змін паролів і можливість відкату.

Створено перелік завдань та план дій для розробки програмного забезпечення. Розроблено загальну функціональну структурну модель, яка описує основні процеси системи. Під час розробки було реалізовано:

– розроблено метод формування оцінки терміну щодо необхідності змінювати пароль на новий;

– розроблено метод визначення рівня складності паролю;

– розроблено модель системи управління паролями та їх шифрування;

– здійснено програмну реалізацію системи управління паролями та їх шифрування;

– спроектовано архітектуру бази даних веб-системи;

– налаштовано серверне оточення для розміщення системи управління паролями та їх шифрування;

– розроблено серверну частину та програмні засоби системи, що містить API для роботи з клієнським додатком для веб.

Розроблене програмне забезпечення було протестоване за допомогою технології «чорної скриньки». Тестування показало коректність відображення інформації та довело працездатність системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Менеджер паролів. URL: https://uk.wikipedia.org/wiki/Менеджер_паролів
2. Штокал А.С., Войтко В.В., Хошаба О.М. Тези доповідей науково-практичної конференції «Молодь в науці: дослідження, проблеми, перспективи (МН-2021)». URL: https://conferences.vntu.edu.ua/index.php/mn/mn2021/paper/view/12975
3. Ю.А. Тарнавський Технології захисту інформації, 2018. 161 с.
4. Node.js. URL: https://www.youtube.com/watch?v=XdtbRkN97go
5. Google passwords: https://passwords.google.com/
6. Dashlane. URL: https://www.dashlane.com/
7. Keeper. URL: https://www.keepersecurity.com/
8. 1Password. URL: https://1password.com/ru/
9. LastPass. URL: https://www.lastpass.com/
10. Niels Provos, David Mazières A Future-Adaptable Password Scheme. Paper - 1999 USENIX Annual Technical Conference, 1999.