МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
УМАНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ САДІВНИЦТВА

Факультет економіки і підприємництва

Кафедра інформаційних технологій

Допущено до захисту
"_____" _____ 2023 р.
Зав. кафедри, к.т.н., доцент
_____ Ліщук Р.І.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

РОЗРОБКА ВЕБ-МОДУЛЮ ДЛЯ ГЕНЕРАЦІЇ УНІКАЛЬНИХ ПАРОЛІВ

Виконав – студент 4 курсу, групи 41-кн
освітньої програми "Комп’ютерні науки"
спеціальності 122 Комп’ютерні науки
Чередниченко Вячеслав Олексійович

Науковий керівник – к.е.н., доцент
Концеба Сергій Михайлович

Умань – 2023

ЗМІСТ

ВСТУП 4

РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 5

1.1 Актуальність досліджуваної задачі 5

1.2 Аналіз програмних продуктів для управління паролями 6

1.3 Постановка задачі 10

РОЗДІЛ 2 ПРОЕКТУВАННЯ ПРОГРАМНОГО ДОДАТКУ 12

2.1 Моделювання програмного додатка генерації та збереження паролів "Safe Password Storage" в нотації IDEF0 12

2.2 Модель аналізу предметної області програмного додатку генерації та збереження паролів "Safe Password Storage" 15

2.3 Модель проектування програмного додатку генерації та збереження паролів "Safe Password Storage" 18

2.4 Моделювання використання програмного додатку генерації та збереження паролів "Safe Password Storage" 20

2.5 Модель реалізації 22

2.6 Моделювання даних 22

РОЗДІЛ 3. РОЗРОБКА WEB-ДОДАТКУ 27

3.1 Створення проекту 27

3.2 Результат виконаної роботи 43

ВИСНОВКИ 59

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 60

ДОДАТКИ 61

ВСТУП

Паролі досить часто критикують як з точки зору безпеки, так і зручності, що робить їх менш ідеальним методом аутентифікації. Водночас паролі як ключі до онлайн-профілів вважаються першим і часто єдиним способом захисту цифрових даних користувачів. Більшість кібератак відбувається за для викрадення паролів з метою отримання конфіденційних даних.

Крадіжка даних може бути тісно пов’язана з фішингом чи іншими методами соціальної інженерії, наприклад, як спам, що являє собою масове розсилання небажаних листів найчастіше за допомогою електронної пошти з метою поширення шкідливих вкладень або посилань.

Для захисту особистої інформації користувачам доводиться використовувати складні паролі, що призводить до складності їх запам’ятовування при використанні на різних сайтах. Отже, виникає питання про способи зберігання паролів. Одним з найпростіших способів є запис паролю на папірець. Однак його легко втратити чи він може бути викраденим. У зв’язку з цим більш ефективним способом зберігання паролів є менеджер паролів.

Менеджер паролів – це програмне забезпечення, яке допоможе користувачу працювати з паролями та PIN-кодами. За допомогою вказаної програми є можливість генерувати унікальні, а головне надійні паролі, та зберігати їх в програмі.

Основною метою даного проекту є розробка програмного забезпечення для захисту особистих інтернет-даних користувачів шляхом генерації та збереження унікальних паролів. Пріорітетною задачею є розробка web-додатку, що має бути зручним та простим у використанні. Для користувачів буде реалізовано створення хешів, за допомогою яких будуть створюватися унікальні паролі, тому сам пароль запам’ятовувати не потрібно.

РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність досліджуваної задачі

З розвитком сучасних інформаційних технологій майже кожна людина користується комп’ютером чи ноутбуком. Все частіше люди спілкуються за допомогою соціальних мереж, черпають інформацію з інтернет-сайтів, проводять онлайн розрахунки та інше. У зв’язку з цим все частіше виникає питання про можливі способи збереження паролів, які є основним ключем до особистої інформації. Одним із варіантів розв'язання вказаної проблеми є створення програми для генерації та збереження паролів.

Використання різних менеджерів паролів допомагає користувачу створювати сотні унікальних і сильних паролів та не запам’ятовувати їх, бо інформація зберігається в базі даних. Але все частіше деяким користувачам хотілося б мати один пароль для всіх сервісів. Основним недоліком втрати цього пароля при зломі одного з сервісів є те, що зловмисник автоматично матиме доступ до всіх інших сервісів.

Отже, виникає необхідність у створенні програмного засобу, за допомогою якого користувач матиме змогу створювати та генерувати захищені паролі, дотримуючись таких правил:

– довжина паролю рекомендовано має складатися не менше як з восьми символів;

– створений пароль має складатися з латинських літер abcd…… та інших;

– пароль має містити латинські літери у великому регістрі;

– пароль має містити цифрові позначення;

– пароль має містити спеціальні символи такі, як @,%,#,! та інші.

Більшість менеджерів паролів у своїй роботі не дотримуються всіх цих правил, зазвичай немає спеціальних символів, цифр або змінного регістра [1].

Навіть якщо згенерувати стійкий пароль у вигляді ключового слова, користувачу важко або неможливо його запам’ятати. Унікальний пароль потрібно зберігати в базі даних, спочатку зашифрувавши, а це може призвести до небажаних наслідків таких, наприклад, як взлом. Ось чому виникла ідея створення програмного продукту, який дозволить зручно, а головне безпечно створювати паролі для різних інтернет сайтів.

1.2 Аналіз програмних продуктів для управління паролями

Під час проведення аналізу програмних продуктів-аналогів, які використовують користувачі під час своєї роботи з паролями, були розглянуті такі їх види, як LastPass, Dashlane та KeePass.

LastPass – це сервіс для керування паролями та їх безпечного зберігання.

Захист даних здійснюється на основі шифрування AES-256. Перевагами цієї програми є її багатоплатформність, вона має безплатну версію, привабливий зовнішній вигляд, інтуїтивно зрозумілий інтерфейс, а також можливість синхронізувати дані [2].

Недоліками програми є необхідність збереження своїх даних на віддаленому сервері. Це означає, що користувач не має контролю над сховищем паролів через що існує ризик можливого злому сервера чи облікового запису з витоком всіх збережених паролів [3].

На рис. 1.1 зображено вигляд вказаного програмного забезпечення.

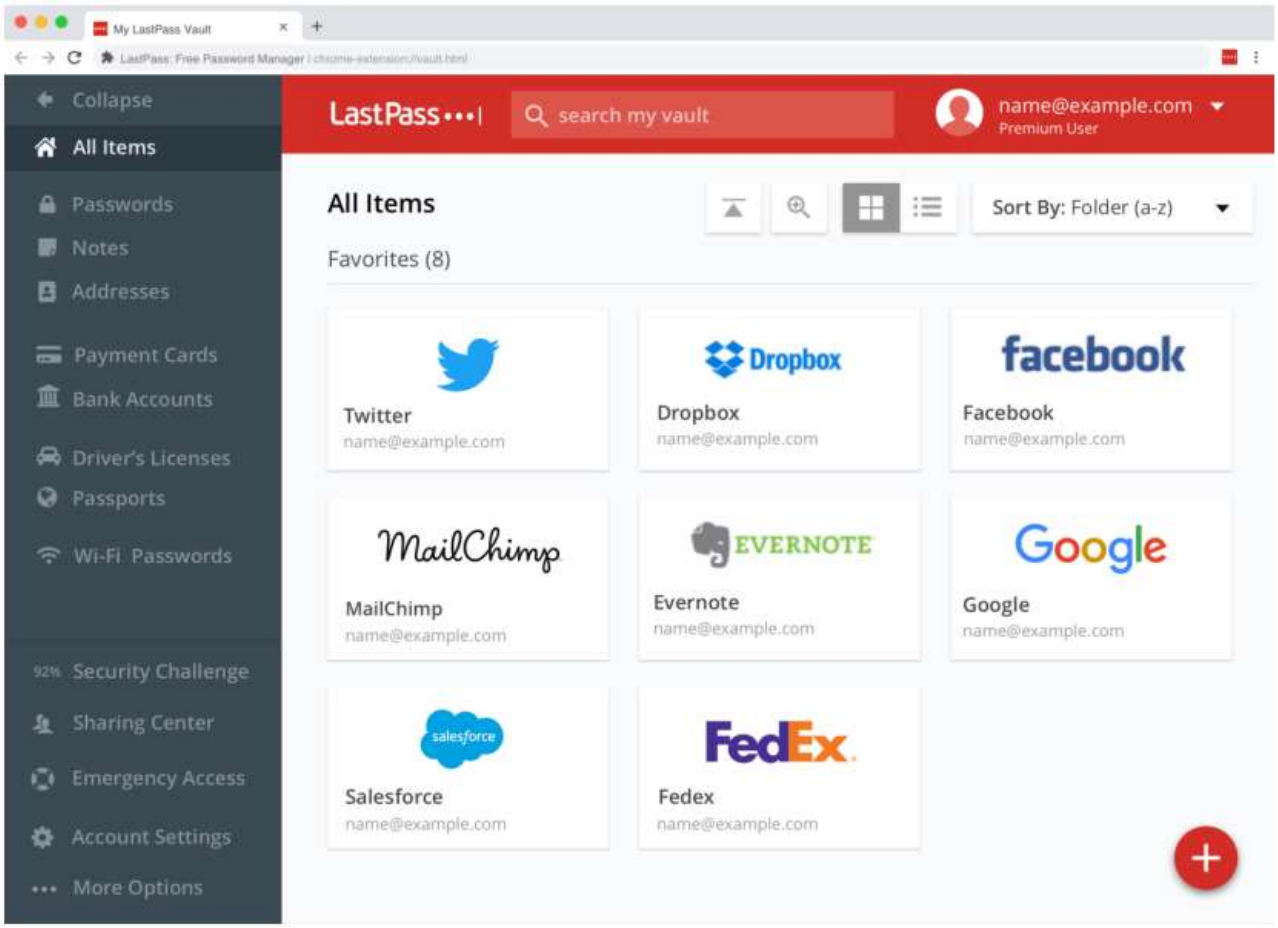


Рис. 1.1 Програма LastPass

Dashlane – це програма менеджер паролів, яка має як вебверсію, так і програму для ПК.

У програмі захист даних здійснюється на основі алгоритму шифрування AES-256. В порівнянні з LastPass ця програма має можливість зберігати свою базу даних на особистому пристрої, має двофакторну авторизацію за стандартом 2FA в безкоштовній версії, та U2F в платному тарифі [4].

Недоліками цієї програми є обмежений обсяг зберігання ключів для безкоштовної версії, а саме 50 паролів. Була знайдена вразливість через особливості роботи сервісу з PIN-кодами [3].

На рис. 1.2 зображено вигляд вказаного програмного забезпечення.

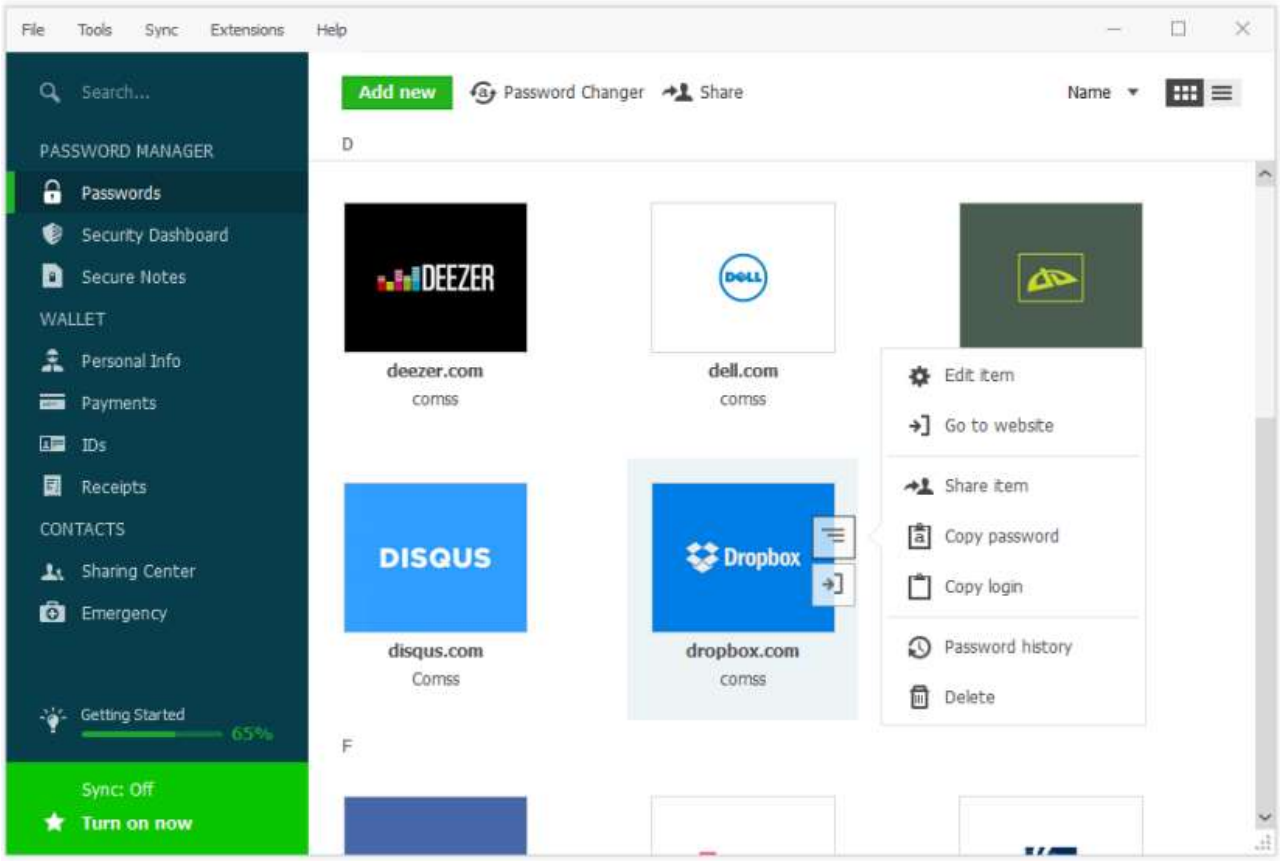


Рис. 1.2 Програма Dashlane

KeePass – це безкоштовний менеджер паролів з відкритим кодом.

Захист даних здійснюється на основі алгоритмів шифрування AES-256 і ChaCha20. Плюсом цієї програми є те, що ці алгоритми мають різний принцип роботи. Таким способом можна захистити користувача від спроб прямого дешифрування бази, оскільки хакер не буде точно впевнений в тому, який саме протокол потрібно застосувати для цього [5].

Недоліком цієї програми є MitM-атака це коли програма завантажує оновлення. Під час передачі незашифрованих пакетів є можливість підміни потрібного пакету даних на заражений. У разі успіху зловмисник може відправити запит на експорт всіх ключів, що зберігаються в активних базах [3].

На рис. 1.3 зображено вигляд вказаного програмного забезпечення.

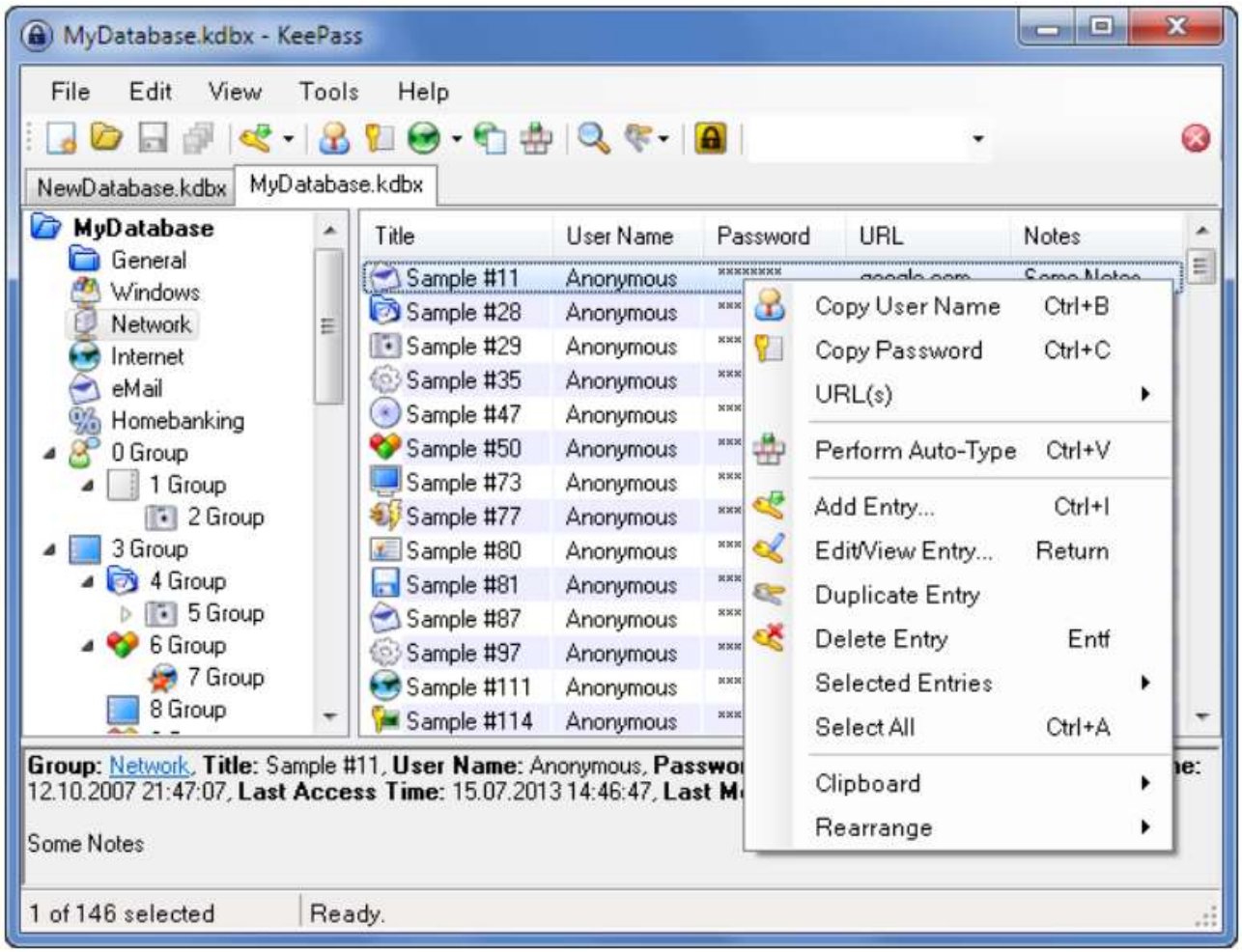


Рис. 1.3 Програма KeePass

Проаналізувавши усі аналоги, визначивши їхні переваги й недоліки в зберіганні паролів, було сформовано список задач для власного програмного забезпечення. А саме:

– використовувати клієнт-серверну архітектуру для багатоплатформності;

– на сервері програма повинна зберігати тільки хеш, а не паролі;

– можливість створення необмеженої кількості паролів;

– програма повинна мати декілька різних алгоритмів шифрування, що використовують різний за параметрами хеш, який буде зберігатися в базі даних, яка не потребує опису схеми таблиць;

– для захисту від MITM атак використовувати протокол HTTPS.

В результаті отримаємо продукт, що покриває недоліки наявних аналогів і забезпечує більш надійний захист від злому.

1.3 Постановка задачі

Метою кваліфікаційної роботи є створення програми у вигляді web-додатку, яка допоможе користувачу використовувати один пароль для генерації унікальних паролів для входу в усі сервіси. Під час створення паролю сервер буде отримувати назву сайту і “звичайний” пароль, який ввів користувач, потім сервер створить унікальний пароль, який буде використовуватися для входу на сайт. Програма не зберігає паролі, а тільки створює хеш для кожного сервісу, за допомогою якого генерується пароль для входу.

Для вирішення вказаної мети необхідно виконати наступні задачі:

– Створити два сервери за допомогою фреймворка Node.js, на яких буде реалізована вся логіка web-додатку.

– Створити бази даних для кожного Node.js сервера для зберігання інформації про користувача та хеш.

– Створити сервер за допомогою фреймворка AngularJS, що призначений тільки для зручності взаємодії користувача з web-додатком. На ньому відбувається валідація даних і обробка запитів з сервера Node.js.

– Розмістити проект на хостингу.

– Виконати тестування проекту.

Для реалізації продукту проекту потрібен хостинг. Найкраще підійде провайдер HostPro.ua так як він має найкращу технічну підтримку серед конкурентів. В якості середовища розробки буде використано Visual Studio, шаблоном проекту буде web-програма Node.js. За допомогою нього буде створено два сервери: перший для взаємодії з користувачем - клієнтський сервер з використанням бази даних MySQL; другий сервер - головний для збереження хешкодів. Для його роботи буде реалізовано базу даних mongoDB. MongoDB не вимагає опису схеми таблиць в порівнянні з реляційними БД. Дані зберігаються у вигляді колекцій, у яких не обов'язково повинна бути схожа структура даних, та документів. Оскільки фреймворк AngularJS використовує

концепцію Model-ViewController, його буде використано в якості web-додатку з можливістю оптимізації під різні платформи.

РОЗДІЛ 2 ПРОЕКТУВАННЯ ПРОГРАМНОГО ДОДАТКУ

2.1 Моделювання програмного додатка генерації та збереження паролів "Safe Password Storage" в нотації IDEF0

За допомогою методології IDEF0 розробляється декілька діаграм, з описом функції або процесу:

– контекстна діаграма;

– діаграма верхнього рівня;

– набір дочірніх діаграм, на яких зображено більш детальне уявлення про об'єкт моделювання.

Вхідними даними будуть дані про користувача та звичайний пароль що ввів користувач.

Обмеження складаються з:

– алгоритму створення унікального пароля;

– статистичних даних;

– алгоритму шифрування інформації.

Ресурси програми будуть складатися з двох серверів та двох баз даних для кожного сервера. Результатом роботи буде унікальний пароль для користувача. Контекстна діаграма процесу створення пароля зображено на рис. 2.1.

Декомпозиція першого рівня контекстної діаграми в нотації IDEF0 має за мету вивчення та аналізу взаємозв’язків організацій, а також змісту та структури.

На діаграмі можна побачити послідовність створення унікального пароля. Спочатку проводиться перевірка вхідних даних про користувача, потім ці дані зберігаються на сервері та робиться запит на головний сервер для створення паролю. На головному сервері відбувається генерація хешу. За допомогою нього і звичайного пароля створюється унікальний пароль з використанням унікального алгоритму [6].



Рис. 2.1 Контекстна діаграма процесу створення пароля

Функціональне моделювання програми генерації та збереження паролів "Safe Password Storage" зображено на рис. 2.2.

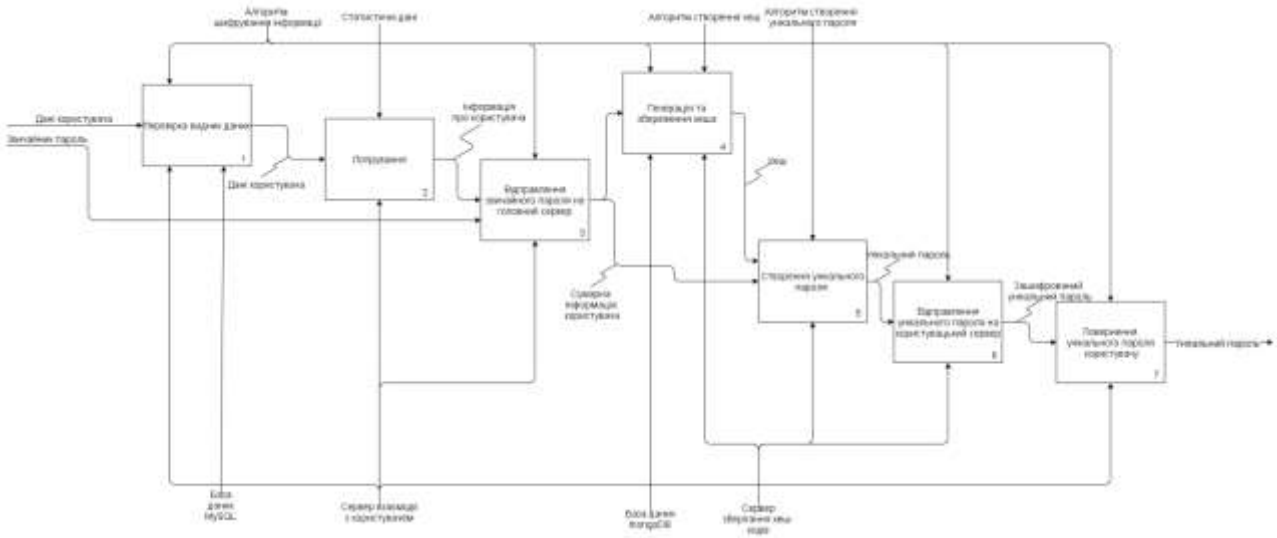


Рис. 2.2 Діаграма декомпозиції процесу створення пароля

Діаграма станів автомата необхідна для того, щоб показати поведінку об’єкта моделювання протягом його життєвого циклу. Тобто, це правила у вигляді ліній. за допомогою яких можна побачити контролер переходу з одного стану до іншого. Як вказано на схемі, що зображена нижче, після авторизації можливі такі дії: – отримання унікального пароля; – створення хешу, а потім унікального пароля; – можливість зміни пароля; Діаграма автоматів програмного додатку генерації та збереження паролів "Safe Password Storage" зображено на рис. 2.3.



Рис. 2.3 Діаграма станів автоматів програмного додатку генерації та збереження паролів "Safe Password Storage"

2.2 Модель аналізу предметної області програмного додатку генерації та збереження паролів "Safe Password Storage"

При моделюванні об’єктно-орієнтованих систем використовується діаграма класів аналізу. Діаграма класів являє собою сукупність елементів моделі, що відображають операції та відношення що їх з’єднують. Діаграма класів аналізу програмного додатку генерації та збереження паролів "Safe Password Storage" зображено на рис. 2.4.

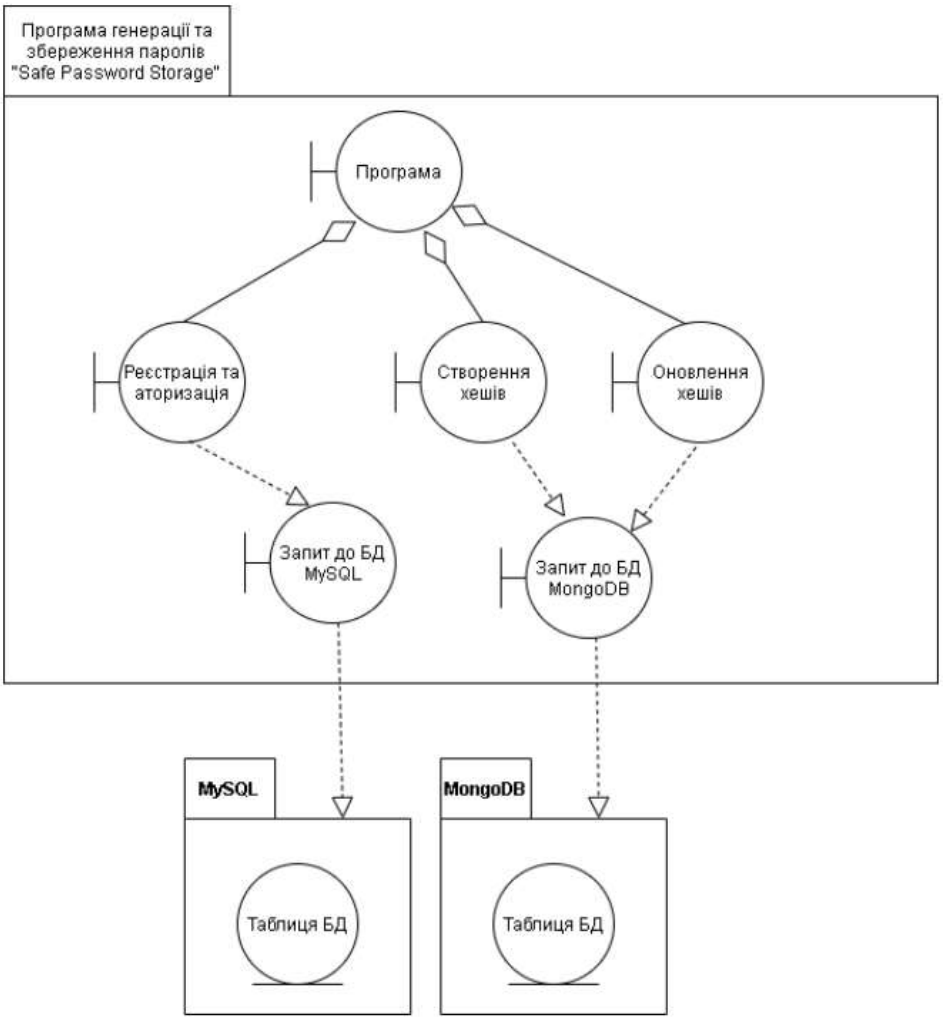


Рис. 2.4 Діаграма класів аналізу програмного додатку генерації та збереження паролів "Safe Password Storage"

Діаграми послідовності відображають взаємодію об’єктів моделювання за часом. Об’єкт моделювання представлений у вигляді прямокутника, що розташований зверху пунктирної лінії – лінії життя моделі об’єкту.

Діаграми послідовності програмного додатку генерації та збереження паролів "Safe Password Storage" зображено на рис. 2.5 – 2.6.

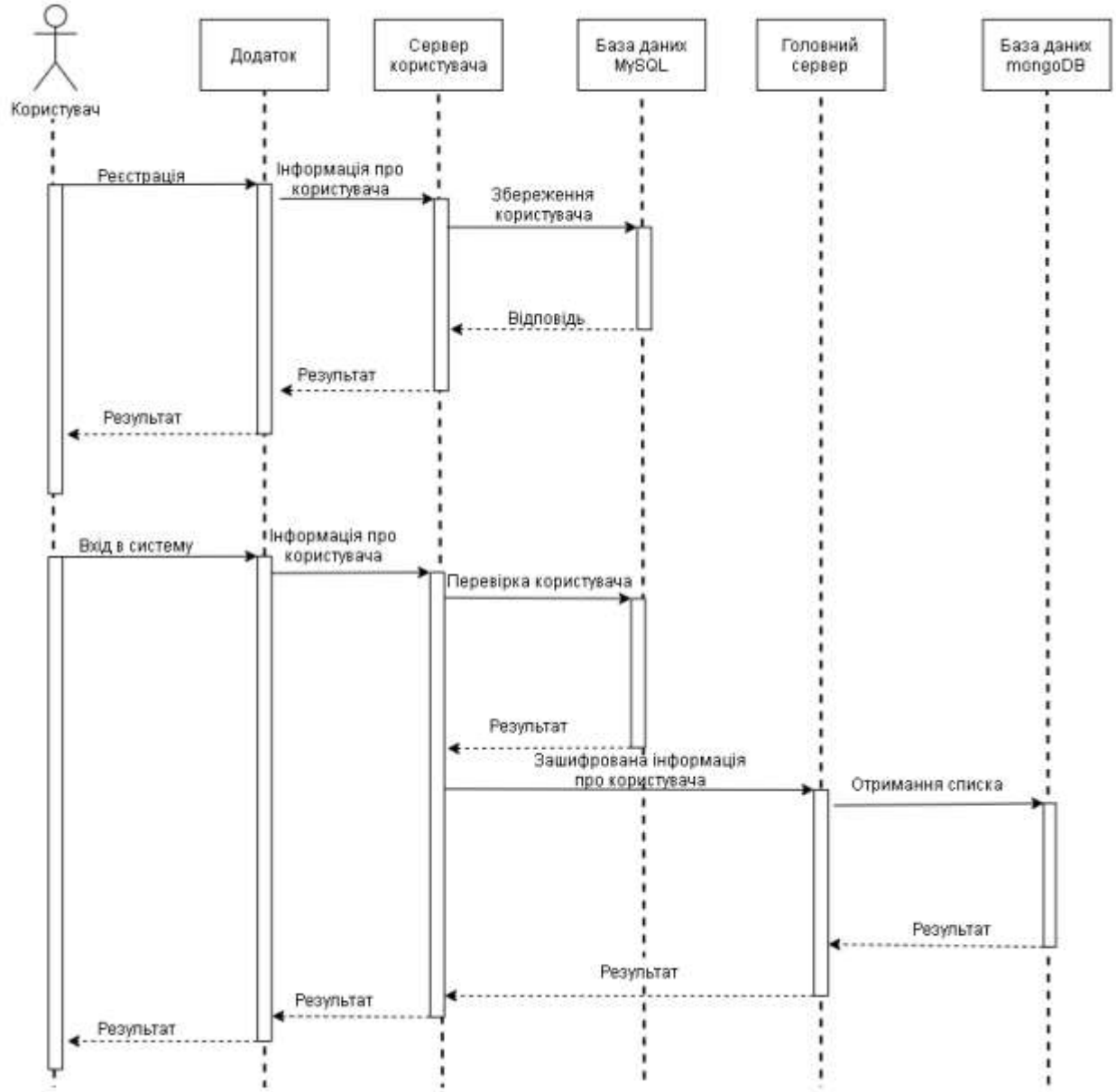


Рис. 2.5 Діаграма послідовності програмного додатку генерації та збереження паролів "Safe Password Storage"

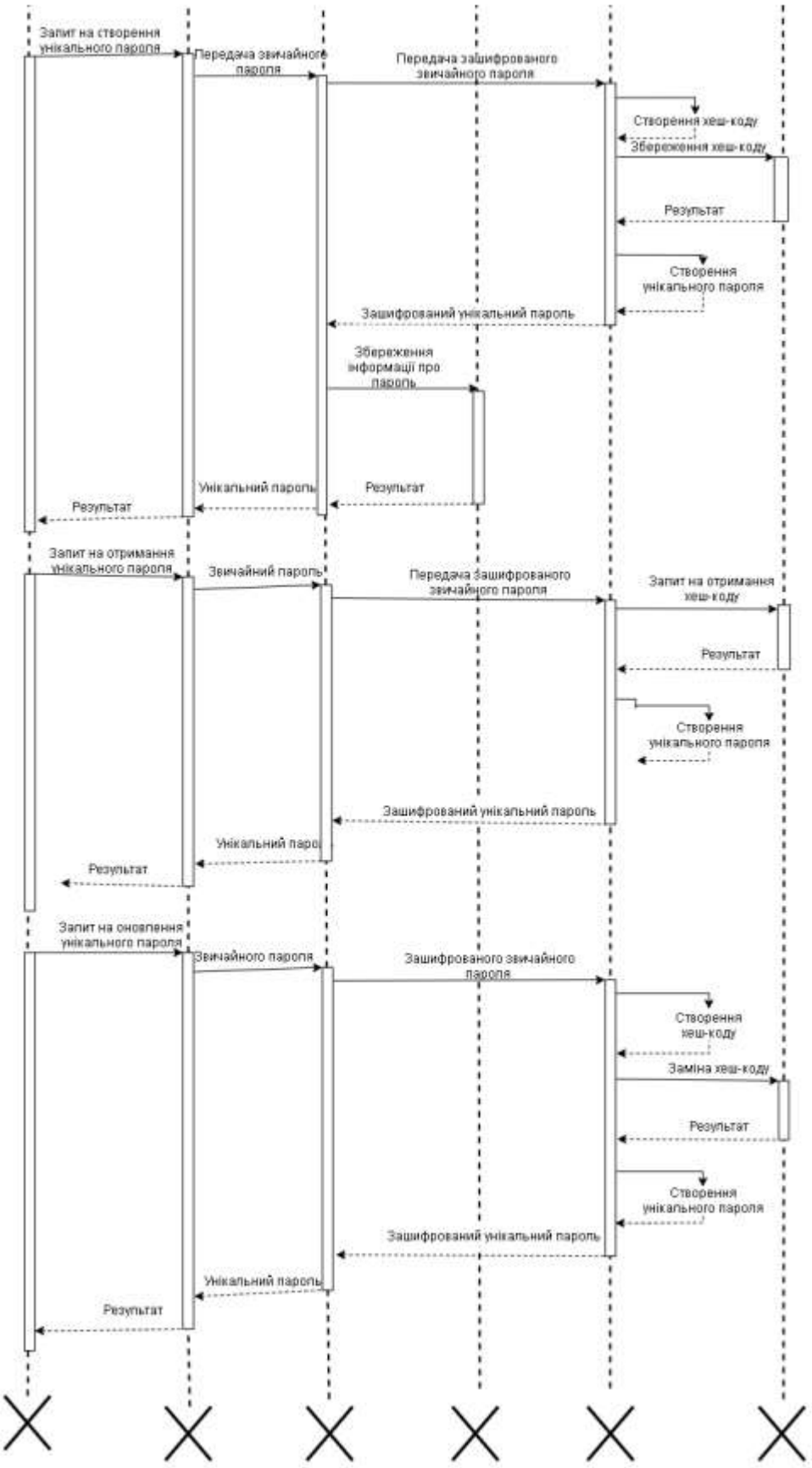


Рис. 2.6 Діаграма послідовності програмного додатку генерації та збереження паролів "Safe Password Storage"

Діаграми комунікації містить інформацію про шлях за допомогою якого відбувається взаємодія між модулями програмного продукту. Діаграму комунікації зображено на рис. 2.7



Рис. 2.7 Діаграма комунікації

2.3. Модель проектування програмного додатку генерації та збереження паролів "Safe Password Storage"

Діаграми класів являє собою структуру моделі, що включає такі елементи як: класи та типи даних і відображає їх зміст. Вказана діаграма показує статичні структури моделі об'єктно-орієнтованого програмування [7].

У діаграмі класів використання визначені наступні основні класи сторінок:

– Авторизація. Клас відповідає за авторизацію користувача, має такі атрибути як логін та пароль. Після завершення операції “авторизація” перенаправляє користувача на головну сторінку**.**

– Головна сторінка. Клас відповідає за відображення списків web-сайтів, які створив користувач, а також має можливість перенаправити користувача на сторінку для створення паролю.

– Створення пароля. Клас відповідає за створення унікального пароля. Необхідними атрибутами для цього є назва web-сайту та звичайний пароль.

– Web-сайти. Клас відповідає за відображення інформації про сайт, а також має можливість перенаправити користувача на редагування пароля та на сторінку унікального пароля.

– Редагування пароля. Клас відповідає за редагування пароля. Необхідними атрибутами для цього є назва web-сайту та звичайний пароль.

– Унікальний пароль. Клас відповідає за отримання унікального пароля. Необхідним атрибутом для цього є звичайний пароль.

Діаграми класів програмного додатку генерації та збереження паролів "Safe Password Storage" зображено на рис. 2.8.

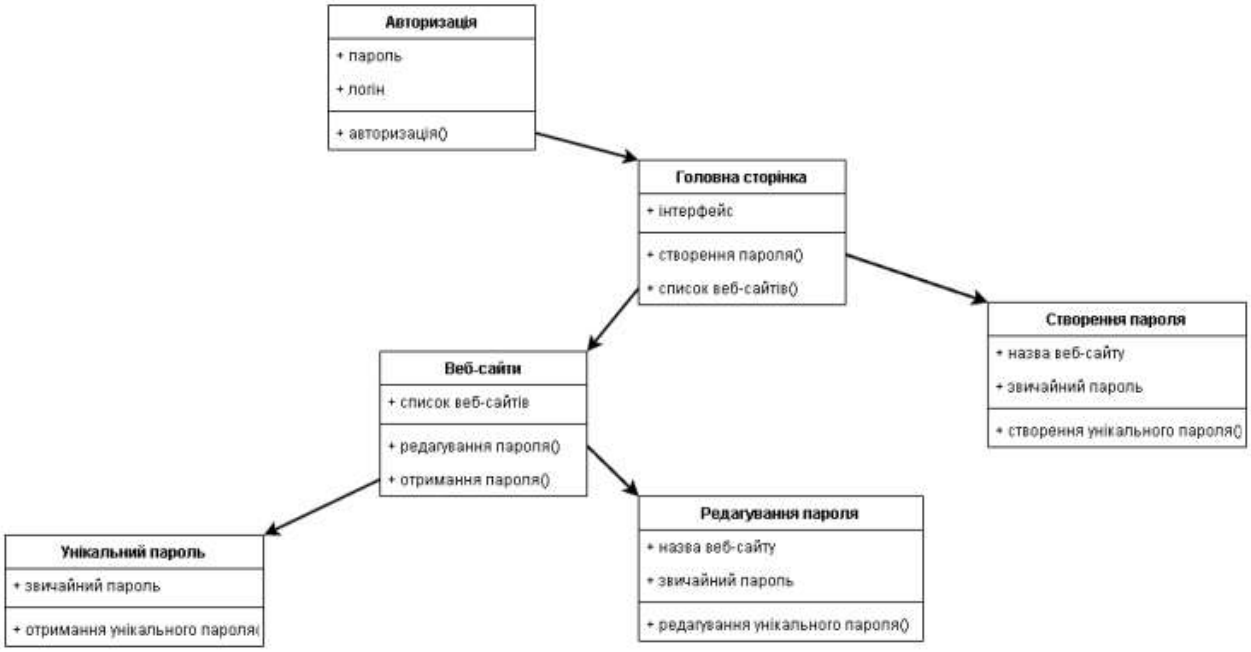


Рисунок 2.8 – Діаграма класів

Діаграма діяльності – це діаграма, за допомогою якої звертаємо увагу на послідовність виконання якихось дій. Діаграма має зображення у вигляді графа, вершини якого визначають стани видів діяльності чи певних дій, а дуги – переходи від одного стану до іншого. За допомогою діаграми діяльності будь-хто може вибирати порядок дій [8].

Діаграми діяльності програмного додатку генерації та збереження паролів "Safe Password Storage" зображено на рис. 2.9.

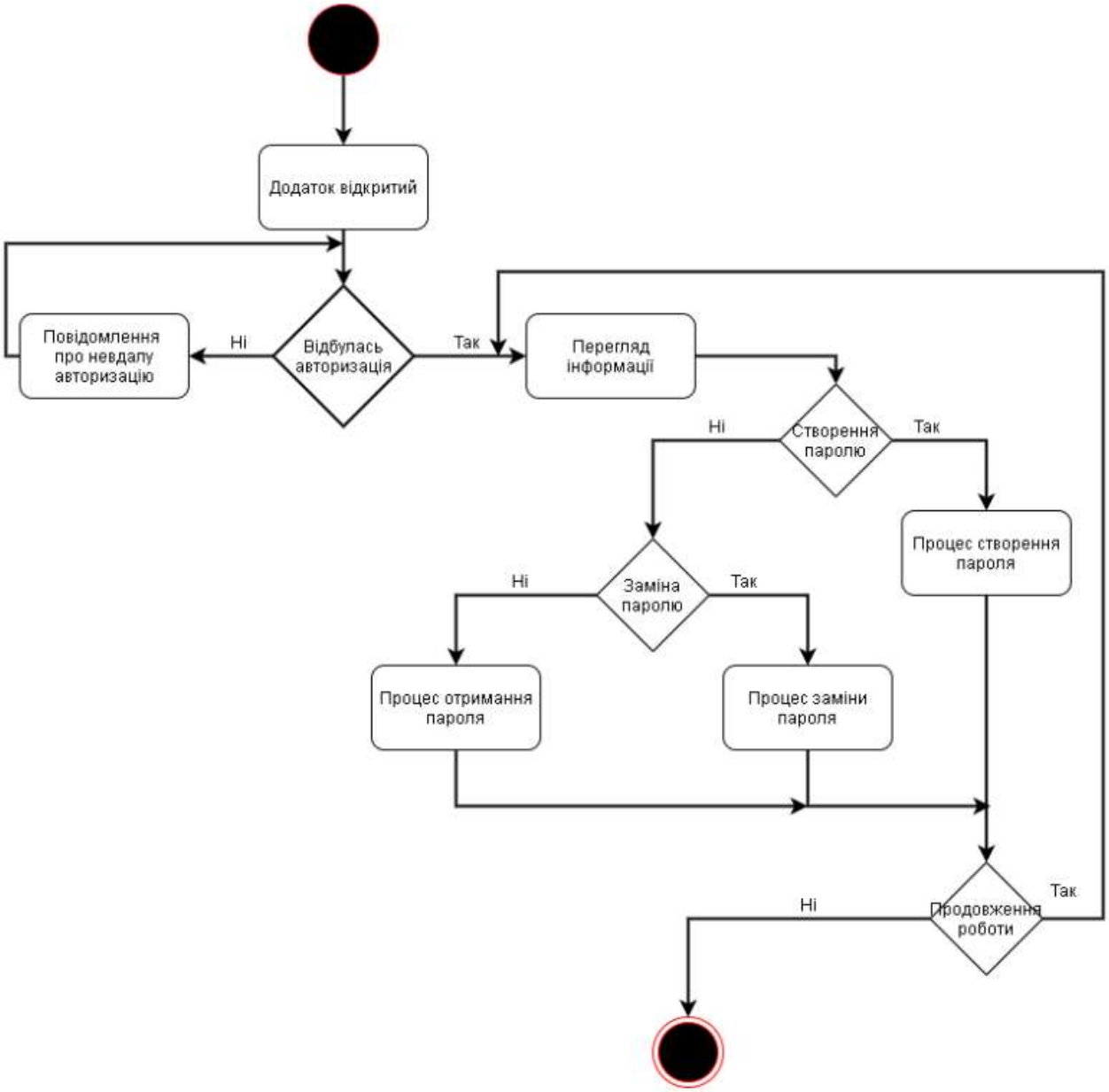


Рис. 2.9 Діаграма діяльності

2.4. Моделювання використання програмного додатку генерації та збереження паролів "Safe Password Storage"

Для зображення в системі можливих відношень між акторами та прецедентами, будують діаграму варіантів використання.

Як показано на діаграмі, наведеній нижче, користувач має можливість на:

– авторизацію;

– реєстрацію;

– створення пароля;

– заміну пароля;

– перегляд паролів;

– отримання пароля;

– повідомлення про помилку на сервері.

В свою чергу адміністратор має можливість на:

– авторизацію;

– блокування користувача;

– розблокування користувача;

– підтримку роботи сервера.

Діаграма варіантів використання в UML представлена на рис. 2.10.

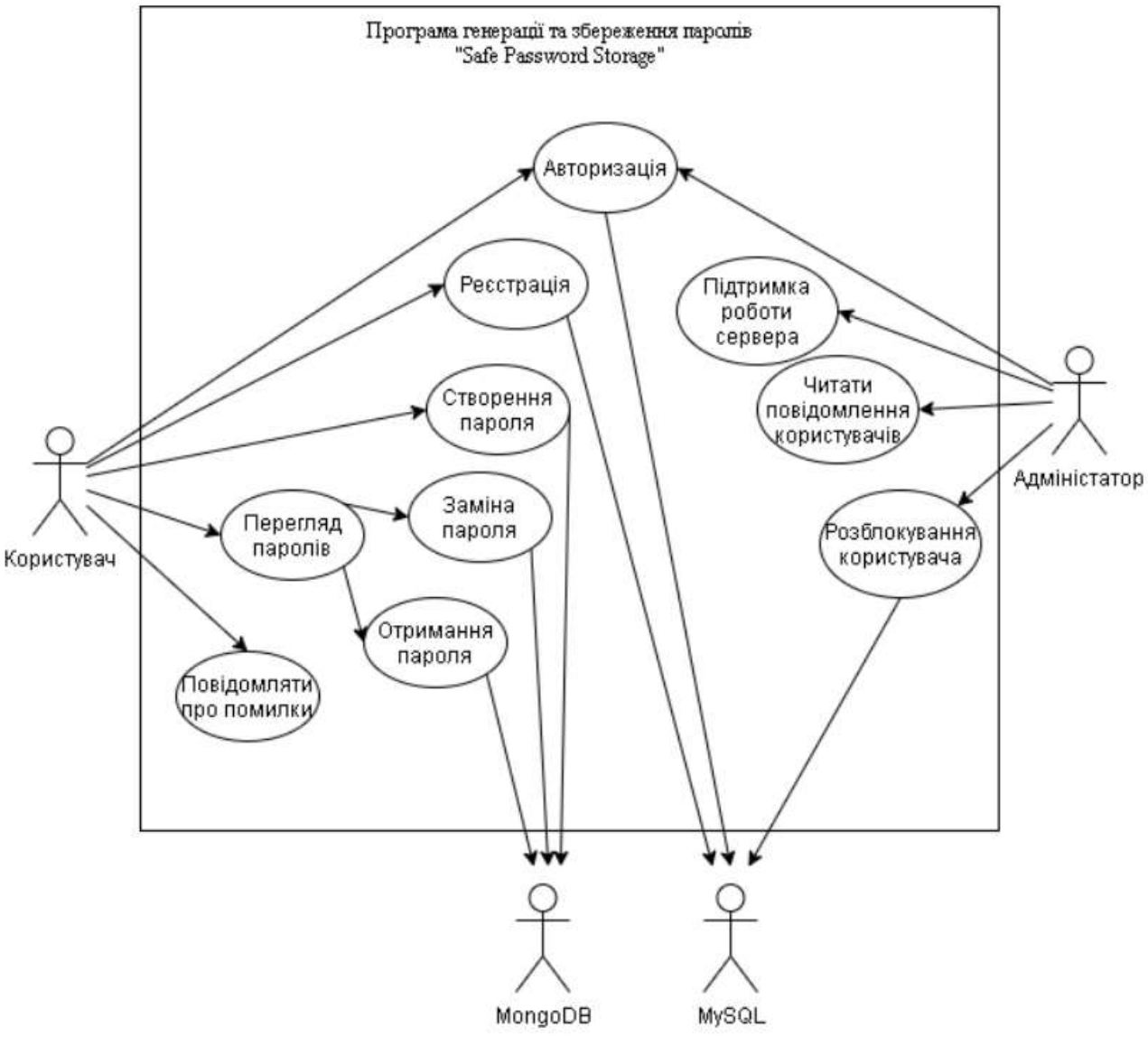


Рис. 2.10 Діаграма варіантів використання

2.5 Модель реалізації

Діаграма розгортання застосовується для розроблення конфігурації програмної системи у вигляді графічного зображення процесів, пристроїв і зав’язків між ними. Ця діаграма завершує процес об’єктного програмування і є останнім етапом специфікації моделювання [9].

Діаграми розгортання програмного додатку генерації та збереження паролів "Safe Password Storage" зображено на рис. 2.11.

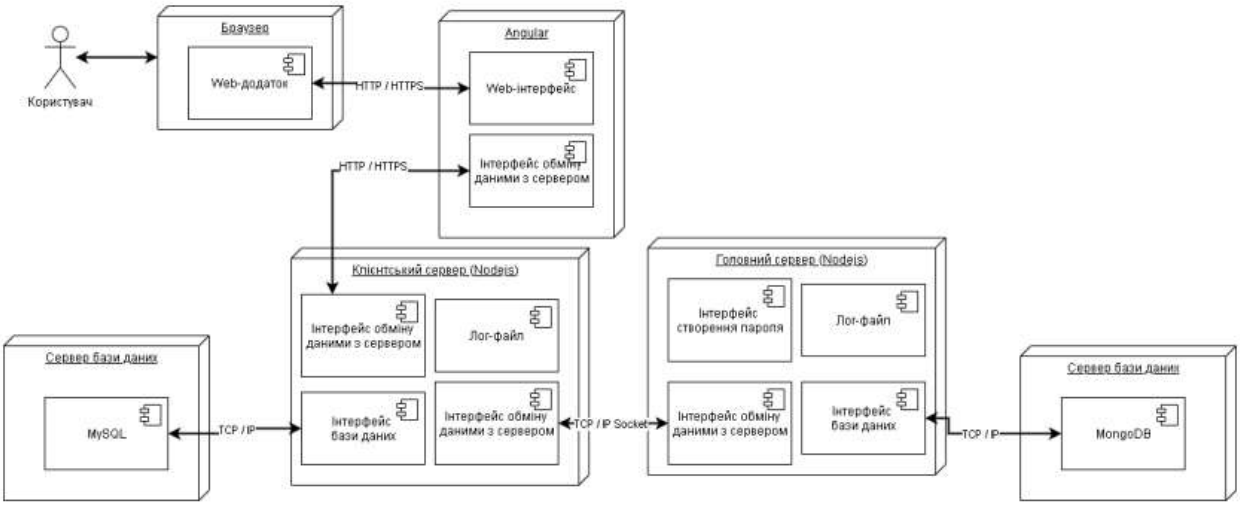


Рис. 2.11 Діаграма розгортання програмного додатку генерації та збереження паролів "Safe Password Storage"

2.6. Моделювання даних

Діаграма потоків даних використовують для проведення аналізу інформаційних систем з метою зменшення їх об’єму, виявлення схожої інформації або її дублювання. Вона представлена у вигляді ієрархії діаграм, за допомогою яких можна спостерігати процес перетворення інформації з початку введення в систему до кінцевої видачі готової моделі користувачу [10].

Діаграми потоків даних програмного додатку генерації та збереження паролів "Safe Password Storage" зображено на рис. 2.12.

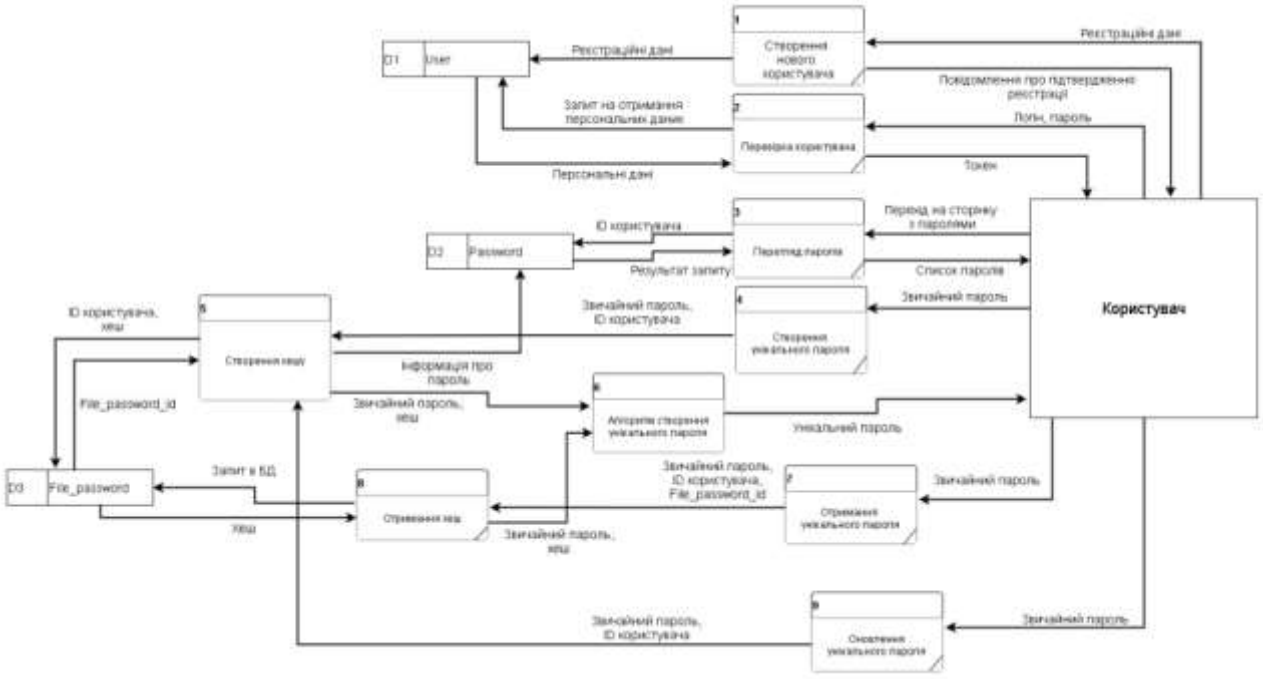


Рис. 2.12 Діаграма потоків даних

Логічна модель даних – це основне джерело інформації на етапі фізичного проектування. З її допомогою можливо проаналізувати будь-які внесені до бази даних зміни й оцінити їх вплив на прикладі розробленої програми [11].

Під час роботи сервера на базі Node.js будуть використовувати дві бази даних: MySQL та MongoDB. База даних MySQL потрібна для зберігання інформації про користувача, а MongoDB - для зберігання файлу, який містить всю необхідну інформацію для створення унікального пароля.

Відомості про типи даних та ключові атрибути наведені нижче у вигляді таблиць (табл. 2.1 – 2.7).

Таблиця 2.1

Список всіх сутностей бази даних

Users	Users
Blockedusers	Заблоковані користувачі
Passwords	Паролі
Messages	Повіломлення користувачів
Roles	Ролі користувачів
FilePassword	Файл для пароля

Таблиця 2.2

Користувачі (Users)

№	Поле	Атрибут	Тип
1	id	Унікальний ідентифікатор користувача	UUID(36)
2	Email	Персональне ім'я користувача	VARCHAR(255)
3	Password	Пароль на вхід в систему	VARCHAR(255)
4	Username	Ім’я користувача	VARCHAR(255)

Таблиця 2.3

Заблоковані користувачі (Blockedusers)

№	Поле	Атрибут	Тип
1	id	Унікальний ідентифікатор	UUID(36)
2	Description	Опис причини блокування	VARCHAR(250)
3	UserId	Унікальний ідентифікатор користувача	UUID(36)
4	Username	Ім’я користувача	VARCHAR(255)

Таблиця 2.4

Паролі (Passwords)

№	Поле	Атрибут	Тип
1	id	Унікальний ідентифікатор пароля	UUID(36)
2	Name	Ім’я пароля	VARCHAR(255)
3	File_password_Id	Унікальний ідентифікатор файлу пароля	VARCHAR(255)
4	Web_site_Id	Унікальний ідентифікатор сайту	VARCHAR(255)
5	User_Id	Унікальний ідентифікатор користувача	UUID(36)

Таблиця 2.5

Повіломлення користувачів (Messages)

№	Поле	Атрибут	Тип
1	Name	Ім’я повідомлення	VARCHAR(255)
2	UserId	Унікальний ідентифікатор користувача	UUID(36)
3	UserName	Ім’я користувача	VARCHAR(255)
4	Description	Опис причини повідомлення	VARCHAR(250)

Таблиця 2.6

Ролі користувачів (Roles)

№	Поле	Атрибут	Тип
1	id	Унікальний ідентифікатор пароля	UUID(36)
2	Name	Ім’я ролі	VARCHAR(255)

Таблиця 2.7

Файл для пароля (FilePassword)

№	Поле	Атрибут	Тип
1	_id	Унікальний ідентифікатор файлу пароля	OBJECTID
2	Id_user	Унікальний ідентифікатор користувача	STRING
3	webSites	Об’єкт, який зберігає всю необхідну інформацію для створення пароля	OBJECT
4	webSites._id	Унікальний ідентифікатор сайту	OBJECTID
5	webSites.hash	Унікальний хеш	STRING
6	webSites.test	Значення для тестування пароля на вірність	STRING
7	webSites.numbers	Масив значення для розміщення числових значень в паролі	[NUMBER]

Логічна модель даних програмного додатку генерації та збереження паролів "Safe Password Storage" зображено на рисунках 2.13 – 2.14.

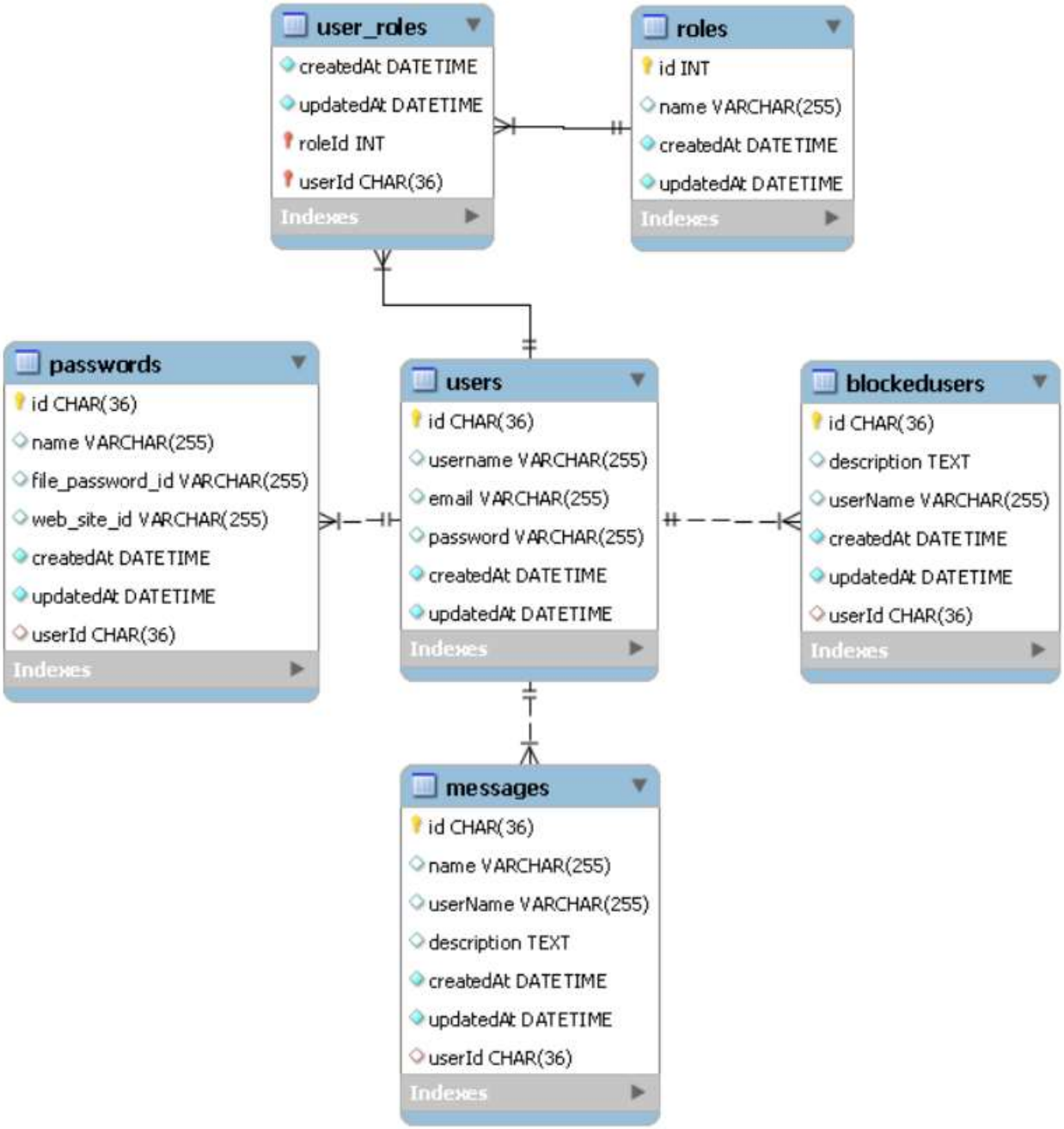


Рис. 2.13 Логічна модель даних MySQL

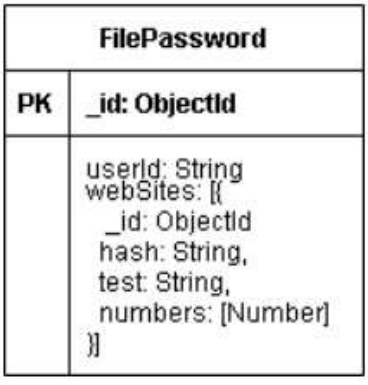


Рис. 2.14 Логічна модель даних MongoDB

РОЗДІЛ 3. РОЗРОБКА WEB-ДОДАТКУ

3.1 Створення проекту

Під час розробки програмного продукту генерації та збереження паролів "Safe Password Storage" використовувались програмний засіб Visual Studio Code (рис 3.1).

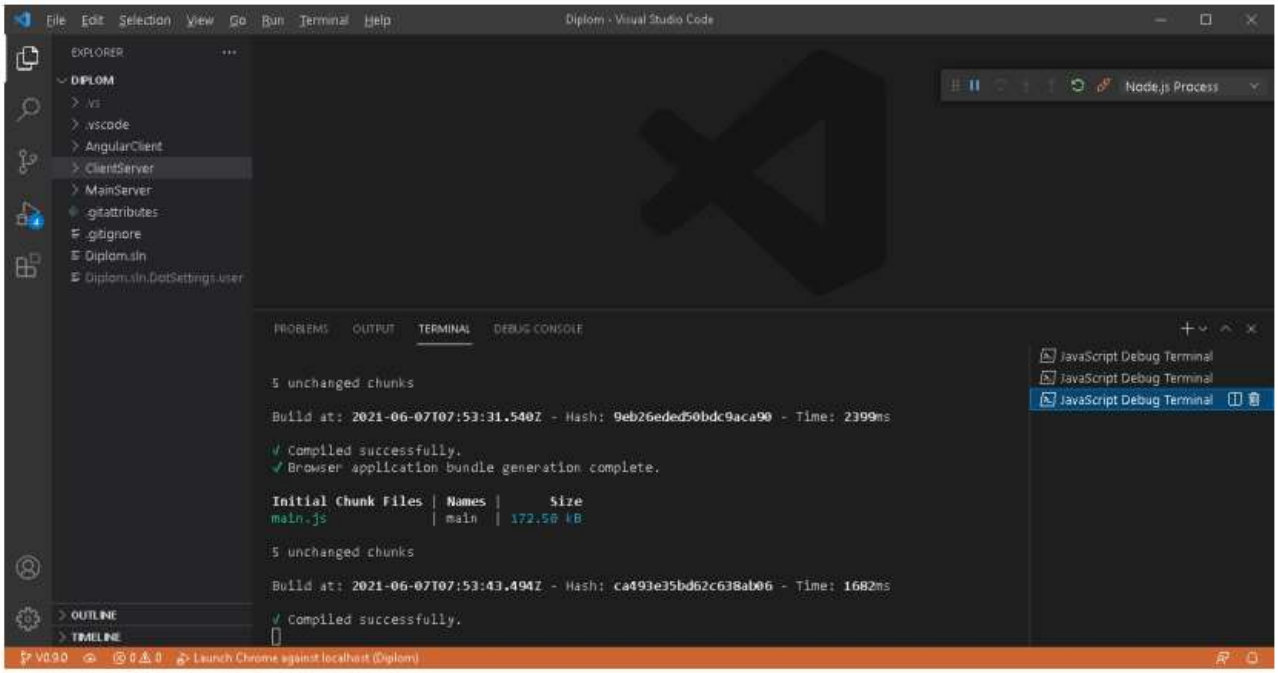


Рис. 3.1 Вікно програми Visual Studio Code

Для зручної налагодження програми були використані розширення в програмі Visual Studio Code (рис 3.2).

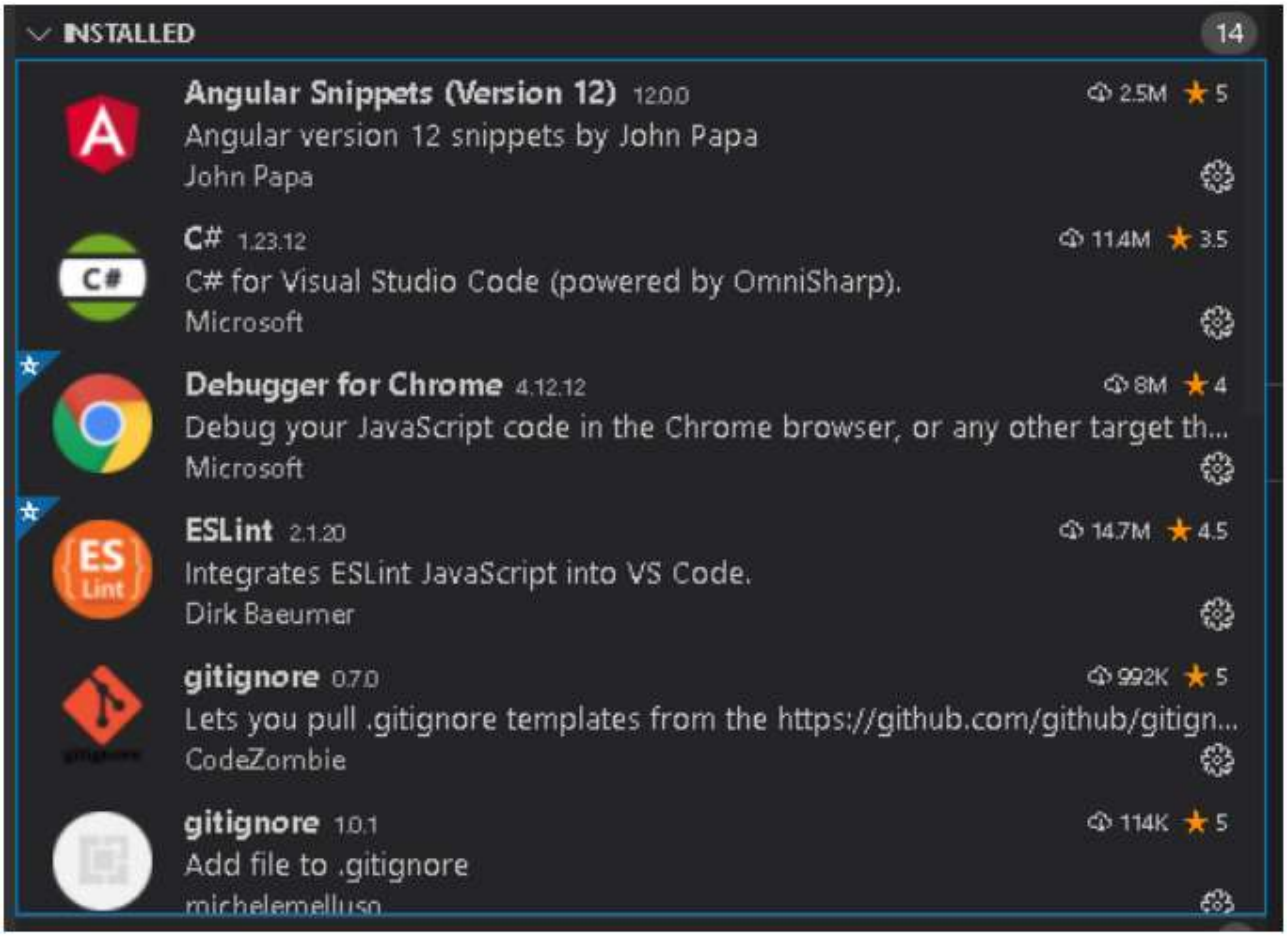


Рис. 3.2 Перелік встановлених розширень

Після цього в файлі проекту створено такі теки:

– AngularClient, що містить фреймворк Angular для взаємодії з клієнтом.

– ClientServer, який містить фреймворк Node.js, що відповідає за підключення до сервера MySQL, в якому зберігається інформація про користувача.

– MainServer також використовує фреймворк Node.js, що відповідає за підключення до сервера MongoDB, в якому зберігається інформація про хеші паролів.

Інформація про створені компоненти та сервіси наведені нижче у вигляді таблиці (табл. 3.1).

Таблиця 3.1

Структура проекту AngularClient

№	Назва класу	Опис	Тип
1	AuthInterceptor	Відповідає за перетворення HTTP запитів перед їх відправленням на сервер.	Interceptor
2	HttpErrorInterceptor	Відповідає за перевірку отриманого результату з сервера.	Interceptor
3	AuthService	Ця послуга надсилає запити на реєстрацію.	Service
4	TokenStorageService	Відповідає за управління інформацією про користувача. Інформацію зберігає в браузері.	Service
5	UserService	Ця послуга надає методи доступу до загальнодоступних та захищених ресурсів.	Service
6	BlockedComponent	Цей компонент відображає список заблокованих користувачів.	Component
7	BoardUserComponent	Цей компонент відображає список паролів, які створив користувач. В ньому користувач має змогу створити, отримати та відредагувати унікальний пароль.	Component
8	BugReportSystemComponent	Цей компонент відображає список повідомлень про помилки користувачів.	Component
9	BugReportUsersComponent	Цей компонент відображає список повідомлень про помилки, що виникли на сервері.	Component
10	HomeComponent	Цей компонент відображає стартову сторінку проекту.	Component
11	LoginComponent	Цей компонент відображає форму для авторизації.	Component
12	ProfileComponent	Цей компонент відображає основну інформацію про користувача.	Component
13	RegisterComponent	Цей компонент відображає форму для реєстрації.	Component
14	ReportComponent	Цей компонент відображає форму для створення повідомлення адміністратора про помилки.	Component
15	AppRoutingModule	Відповідає за маршрутизацію для web додатку.	Module
16	AppComponent	Цей компонент є кореневим компонентом фреймворку Angular.	Component
17	AppModule	Це кореневий модуль фреймворку Angular.	Module

Структура проекту AngularClient зображена на рисунку 3.3.

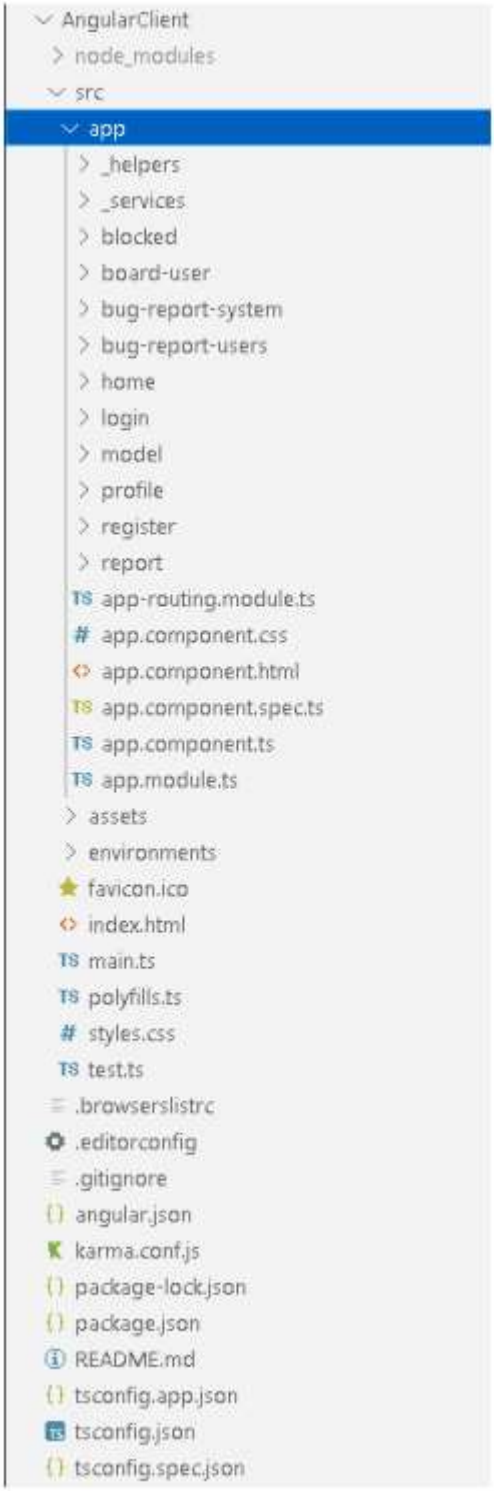


Рис. 3.3 Структура проекту AngularClient

Структура каталогів для проекту ClientServer зображена на рисунку 3.4.



Рис. 3.4 Структура проекту ClientServer

Інформацію про створені файли проекту ClientServer наведено нижче у вигляді таблиці (табл. 3.2).

Таблиця 3.2

Структура проекту ClientServer

№	Назва файлу	Опис
1	auth.config.js.	Містить секретний ключ для побудови JWT аутентифікації
2	Client.js	Шаблон класу для зберігання інформації користувача під час підключення через socket.
3	db.config.js	Містить параметри для підключення до бази даних MySQL та Sequelize.
4	auth.controller.js	Виконує обробку дії реєстрації та входу.
5	user.controller.js	Контролер для обробки запитів користувача.
6	rsa.js	Клас для створення закритих та відкритих ключів, шифрування та дешифрування інформації.
7	authJwt.js	Виконує перевірку користувача по ролі у базі даних.
8	socket.js	Містить функції для відправки запитів на головний сервер за допомогою бібліотеки socket.io-client.js.
9	verifySignUp.js	Виконує перевірки на повторюваність імені чи електронної пошти у базі даних.
10	models/index.js	За допомогою бібліотеки sequelize.js робить підключення до бази даних MySQL, використовуючи моделі файлів, які розташовані в такі models та створює таблиці з взаємозв’язками.
11	auth.routes.js	Містить запроси для реєстрації та авторизації.
12	user.routes.js	Містить основні запроси користувача та адміністратора.
13	babel.config.js	Містить налаштування для використання бібліотеки babel фреймворком Node.js.
14	server.js	Виконує підключення всіх необхідних модулів та маршрутів.

Структура каталогів для проекту MainServer зображена на рисунку 3.5.



Рис. 3.4 Структура проекту MainServer

Інформація про створені файли проекту MainServer наведені нижче у вигляді таблиці (табл. 3.3).

Таблиця 3.3

Структура проекту MainServer

№	Назва файлу	Опис
1	Algoritm.js	Містить метод створення унікального пароля.
2	Client.js	Шаблон класу для зберігання інформації користувача під час підключення через socket.
3	config.js	Містить параметри для підключення до бази даних MongoDB та mongoose.
4	db.js	Містить необхідні методи для взаємодії з базою даних MongoDB.
5	rsa.js	Клас для створення закритих та відкритих ключів, шифрування та дешифрування інформації.
6	FilePassword.js	Містить схему таблиці FilePassword.
7	user.json	Зберігає інформацію про кількість невірно введених паролів користувача.
8	babel.config.js	Містить налаштування для використання бібліотеки babel фреймворком Node.js.
9	server.js	Виконує підключення всіх необхідних модулів та маршрутів.

Для проведення аутентифікації в проекті використовується бібліотека jsonwebtoken.js, яка потрібна для верифікації тверджень. Авторизований користувач має токен, який зберігається в sessionStorage та перевіряється при кожному запиті. Коли час дії токена вийшов, клієнту потрібно знову авторизуватися для продовження роботи в системі.

Для формування ключа та безпечного зберігання паролів користувача в базі даних було використано бібліотеку bcryptjs.js. Для того щоб дізнатися пароль злоумисник використовує райдужні таблиці, тобто бази даних, в яких за пароль відповаідає обчислений хеш. Знайшли хеш – знайшли пароль. Для захисту від цих атак bcrypt використовує необоротні хеш-функції, які включають salt. Дана функція збільшує довжину та складність пароля, що робить попередні обчислення просто безглуздими а отже атаку неможливою.

Підключення та передача даних виконується за допомогою бібліотеки Socket.IO. Socket.IO – це JavaScript-бібліотека для web-застосунків і обміну даними в реальному часі. Складається з двох частин: клієнтської і серверної. Було створено клас RSA, призначений для шифрування та дешифрування

даних. За допомогою нього відбувається створення відкритого і закритого ключа. Послідовність підключення та передача даних між серверами наведено на рисунках 3.5-3.6.

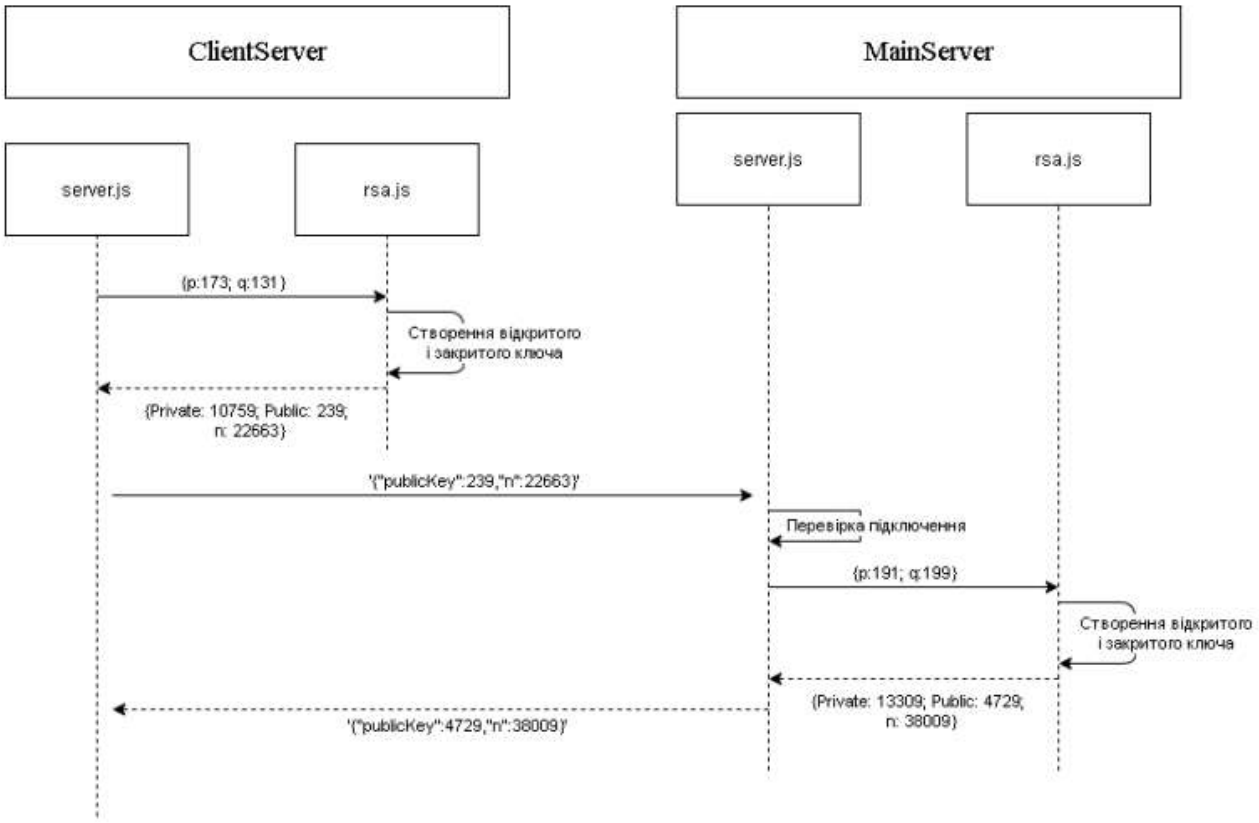


Рис. 3.5 Встановлення зв'язку між серверами

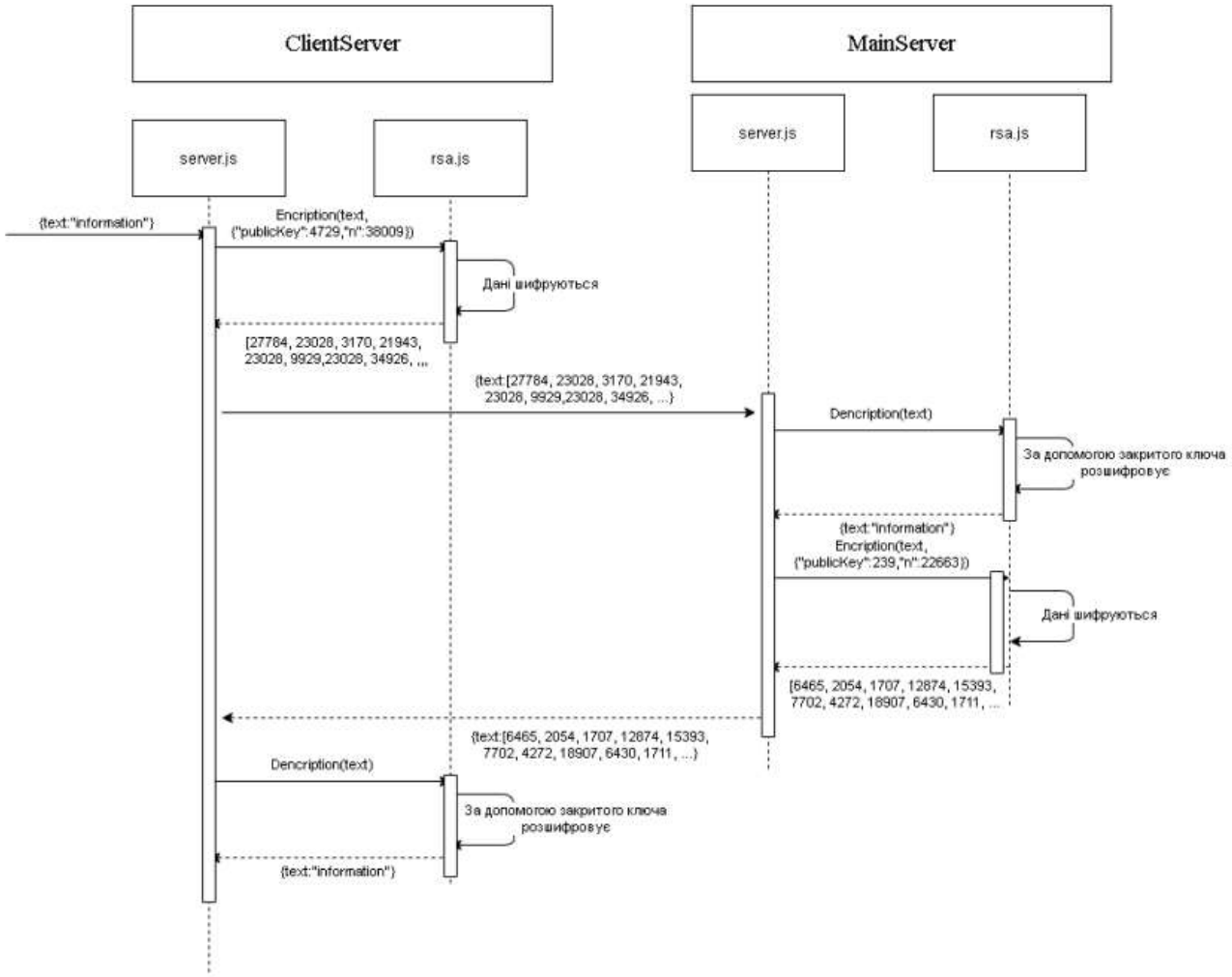


Рис. 3.6 Передача даних між серверами

Алгоритм шифрування паролів був розроблений автором проекту, детальний опис роботи якого наведений нижче.

На початку роботи формується хеш та масив чисел для врахування позиції додаткової інформації в паролі. Далі виконується сумування кожного символу як код символу. Отриманий результат перемножується на кожний символ. Результат розбивається на символи та множиться з хешем. Після цього відбувається ділення результату на частини та проходить перевірка на кількість елементів в масиві. Якщо умова перевірки невірна то зменшується або додається додаткова інформація до тих пір, поки умова не буде дійсною. Отриманий результат передається у функцію CreatePassword. В цій функції відбувається створення структури майбутнього пароля та за допомогою неї будуємо унікальний пароль.

Унікальність даного алгоритму формування пароля полягає в тому, що немає ніякої можливості відновити звичайний пароль. Блок-схеми кожного етапу формування пароля відображено на рисунках 3.7-3.13.

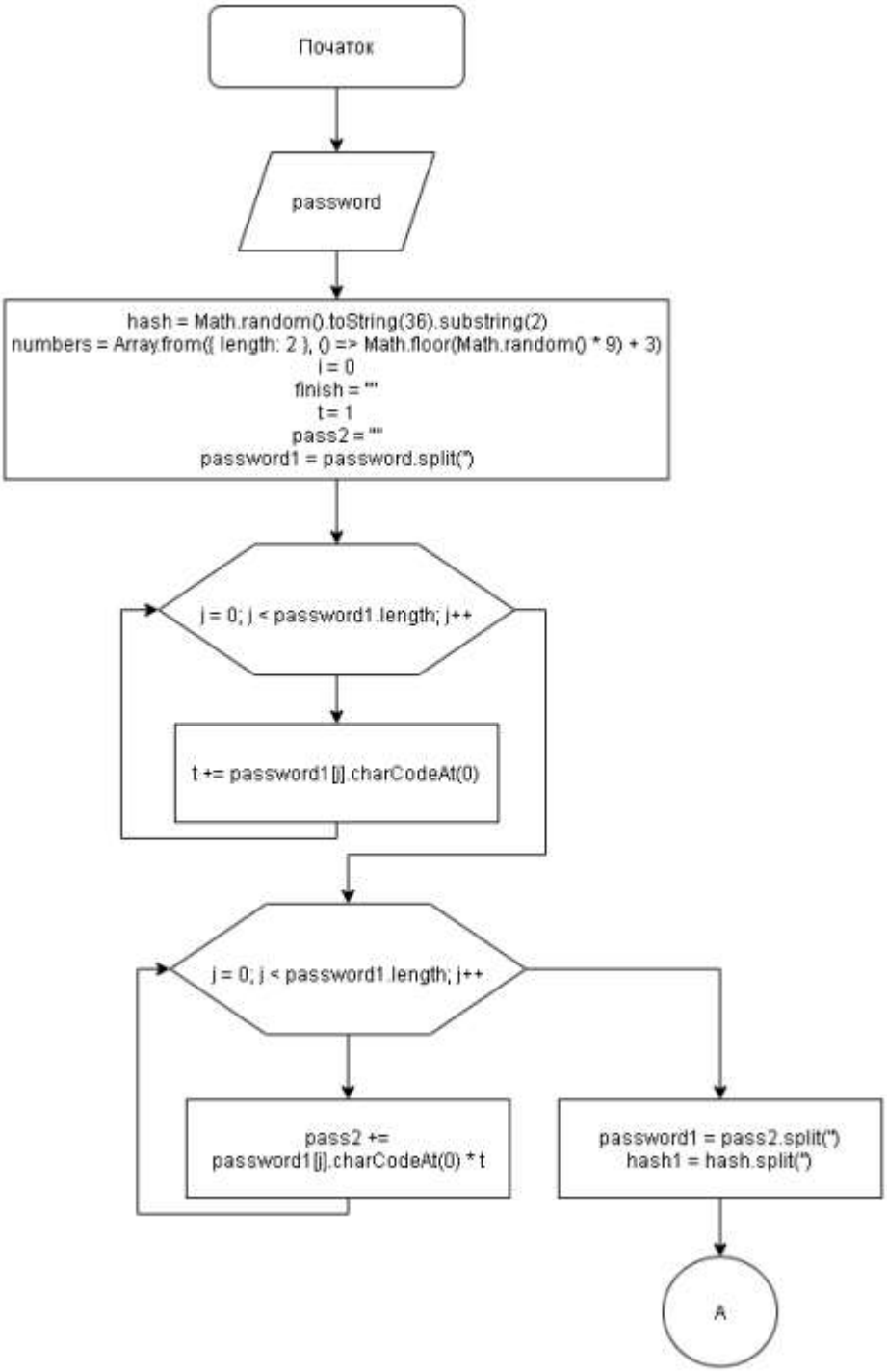


Рис. 3.7 Блок-схема стартового алгоритму

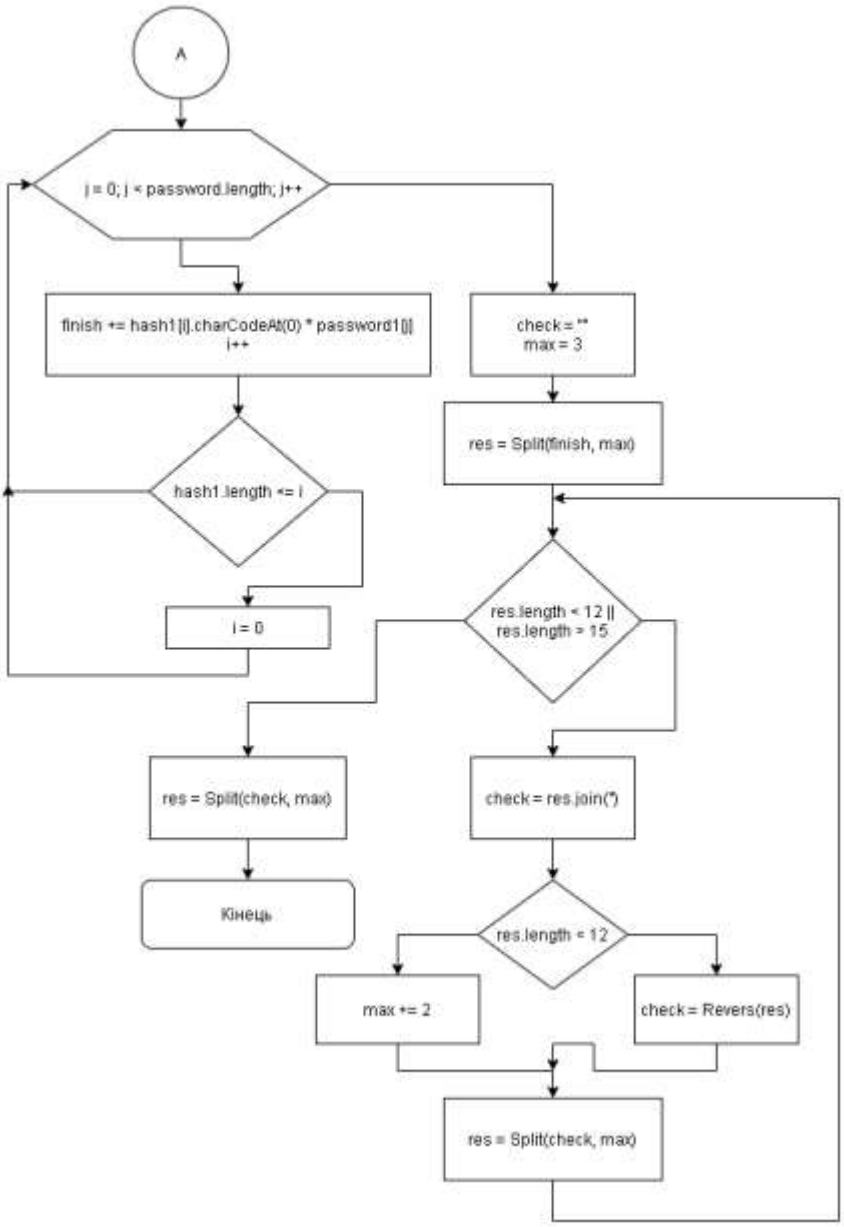


Рис. 3.8 Продовження блок-схеми стартового алгоритму

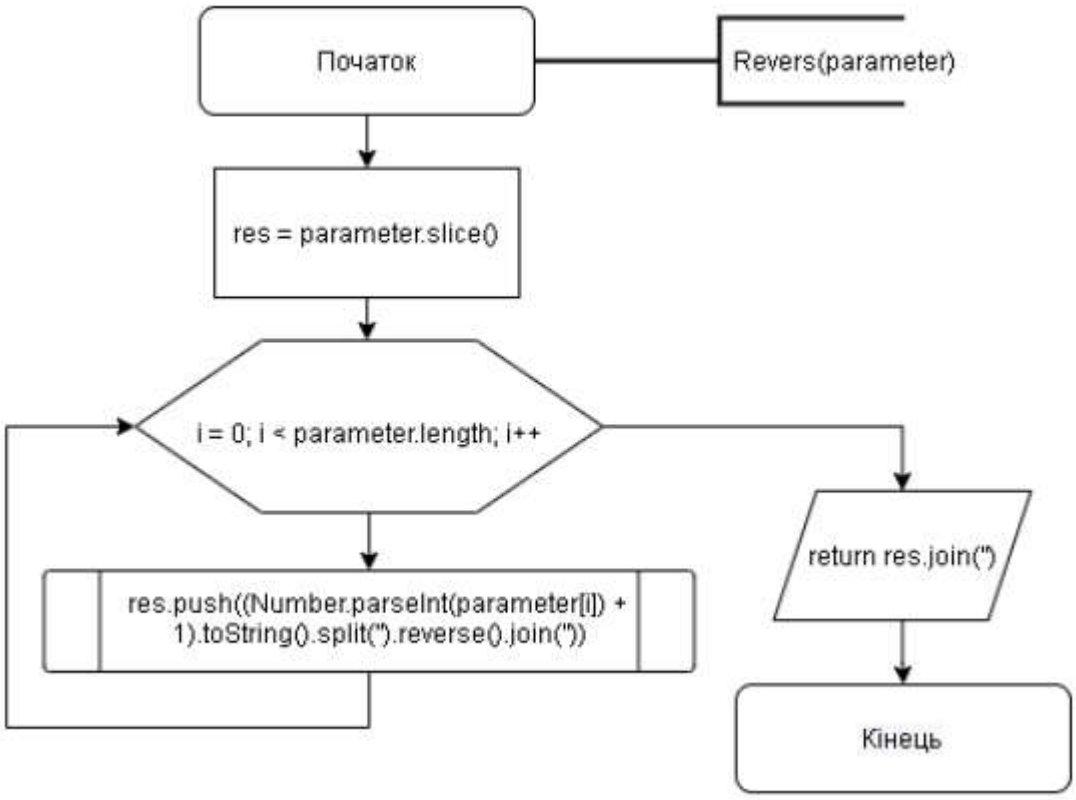


Рис. 3.9 Блок-схема функції Revers

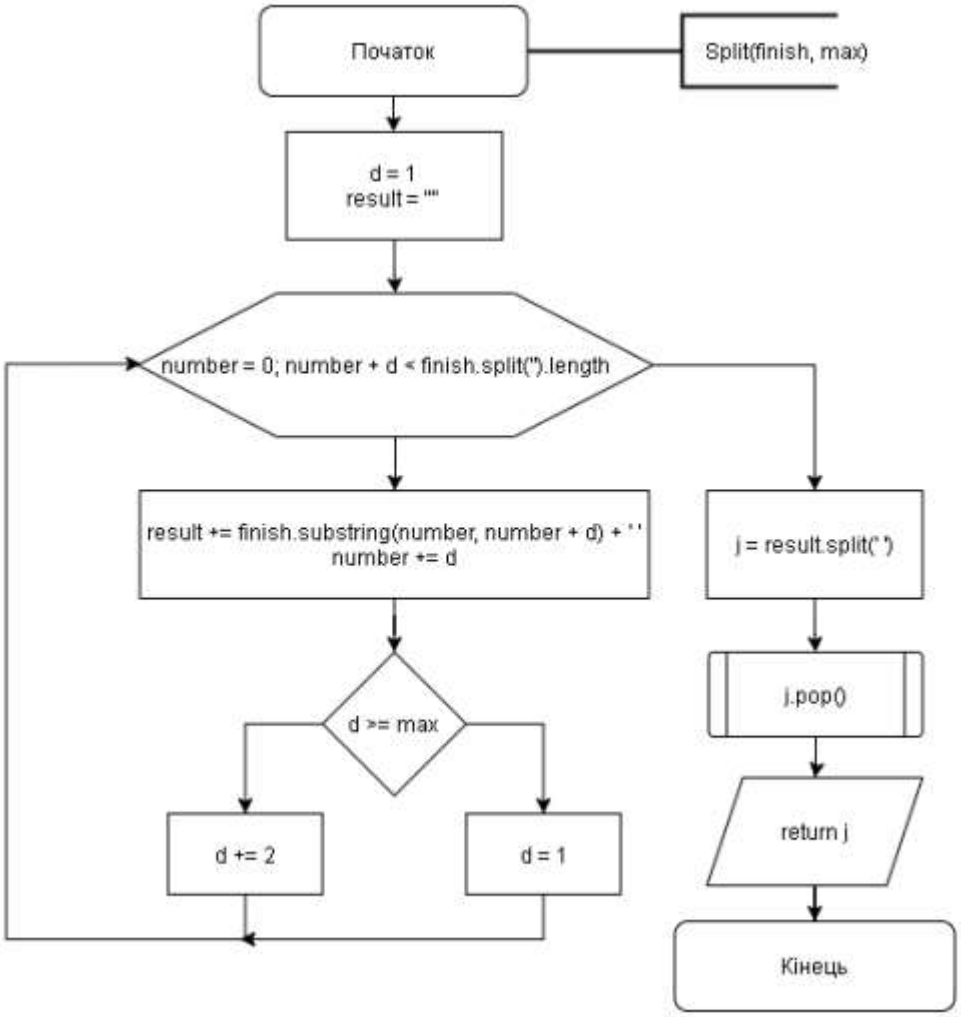


Рисунок 3.10 Блок-схема функції Split

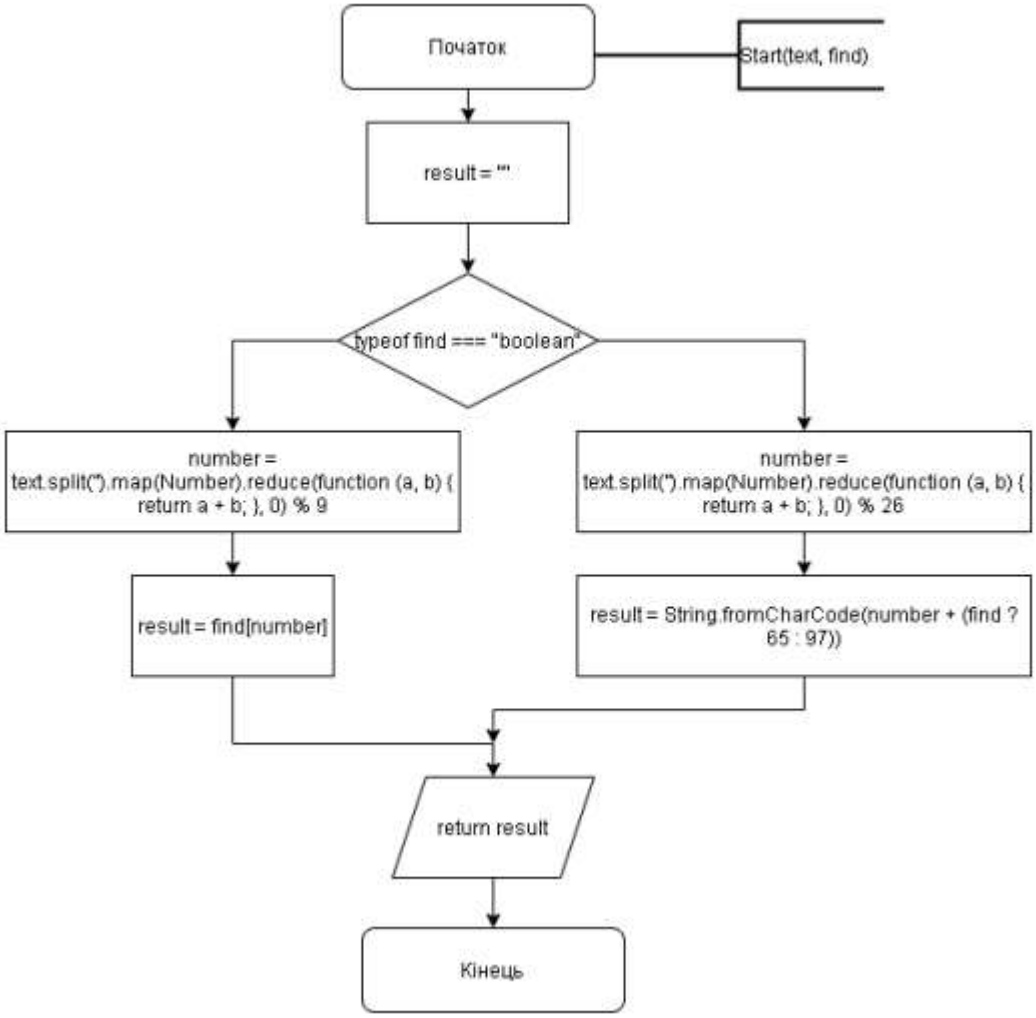


Рис. 3.11 Блок-схема функції Start

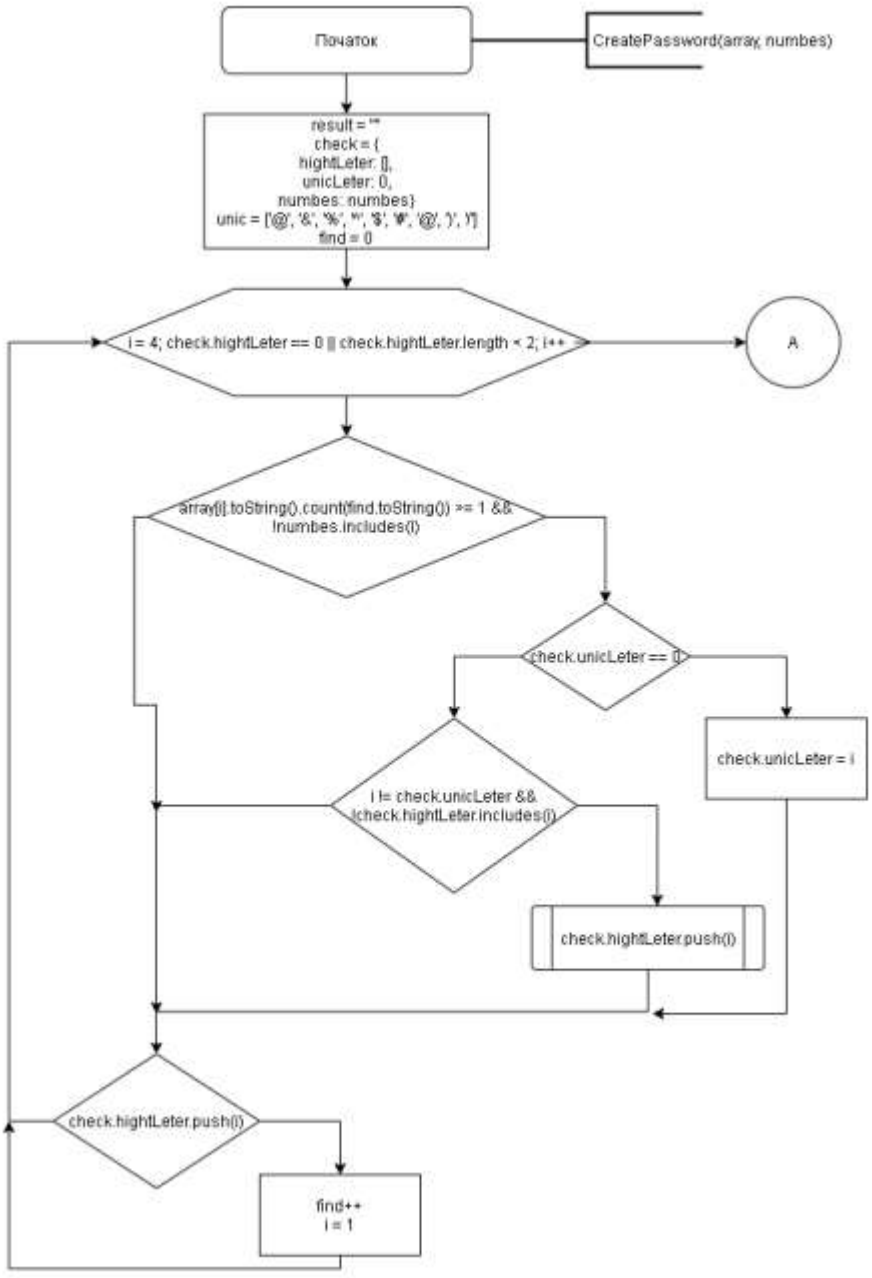


Рис. 3.12 Блок-схема функції CreatePassword

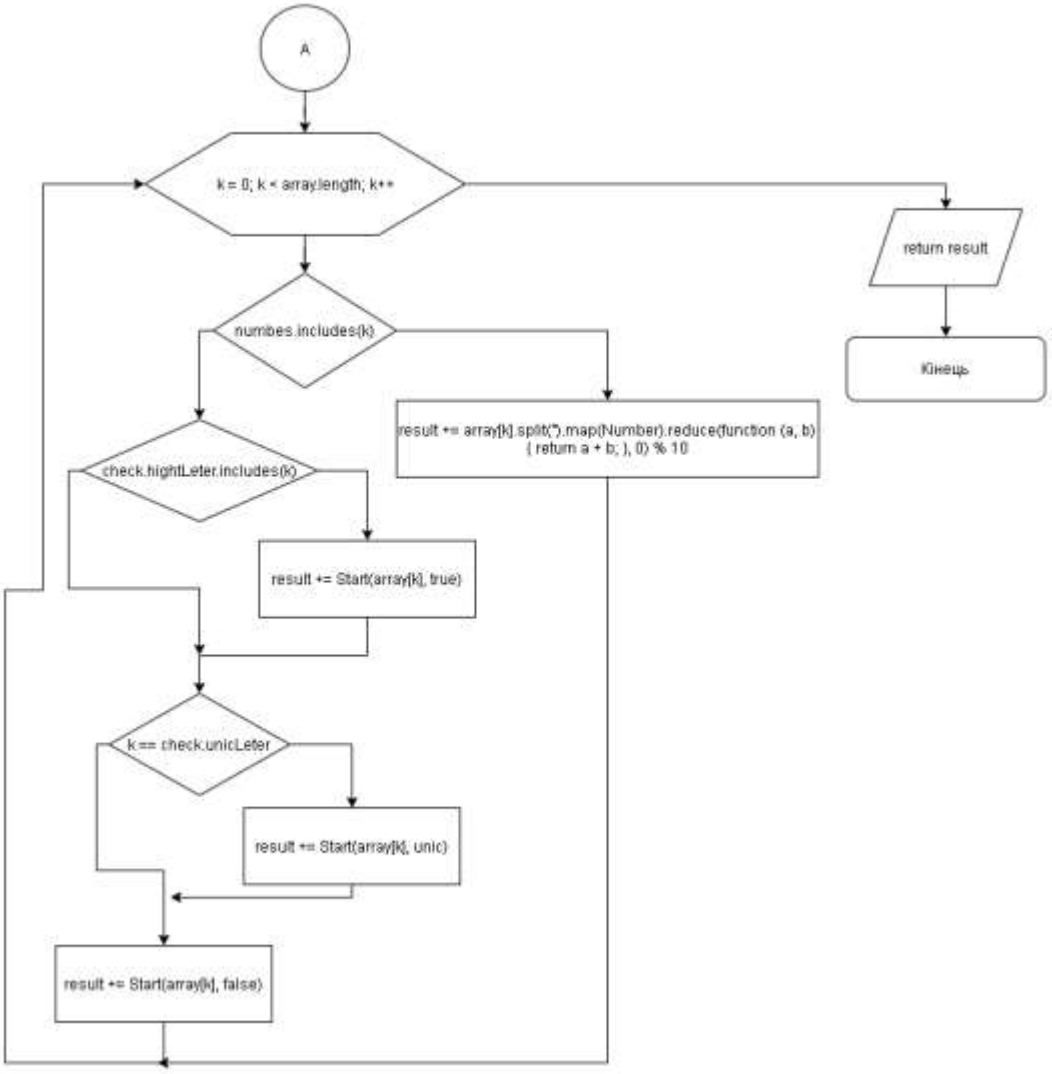


Рис. 3.13 Продовження блок-схеми функції CreatePassword

3.2 Результат виконаної роботи

У результаті виконання роботи було досягнуто мету проекту–створення програми генерації та збереження паролів “Safe Password Storage”. Створено основний функціонал:

– головна сторінка сайту;

– механізм авторизації та реєстрації користувачів;

– можливість створення та редагування пароля;

– можливість створення повідомлення про помилки;

– адміністративна частина.

Головна сторінка сайту зображено на рисунку 3.14.



Рис. 3.14 Скріншот головної сторінки

Дане програмне забезпечення дає можливість неавторизованому користувачеві зробити тестову спробу перевірки алгоритму шифрування паролів шляхом введення будь якого пароля та в результаті отримати зразок нового унікального пароля (рис. 3.15).

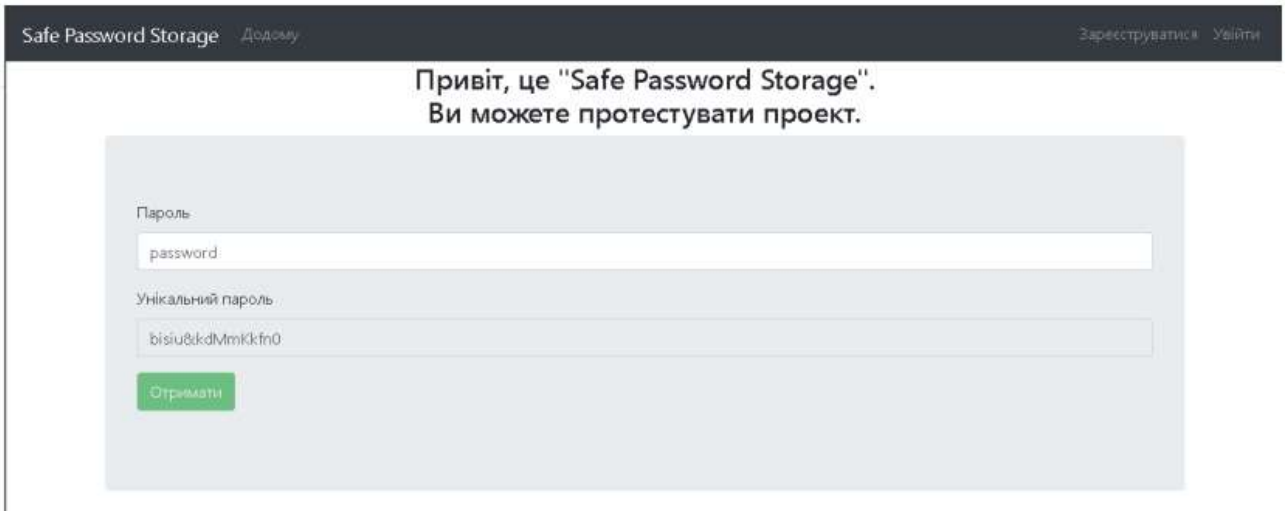


Рис. 3.15 Скріншот роботи алгоритму

Першим етапом роботи користувача в даному програмному забезпеченні є сторінки реєстрації та авторизації (рис. 3.15 – 3.16).

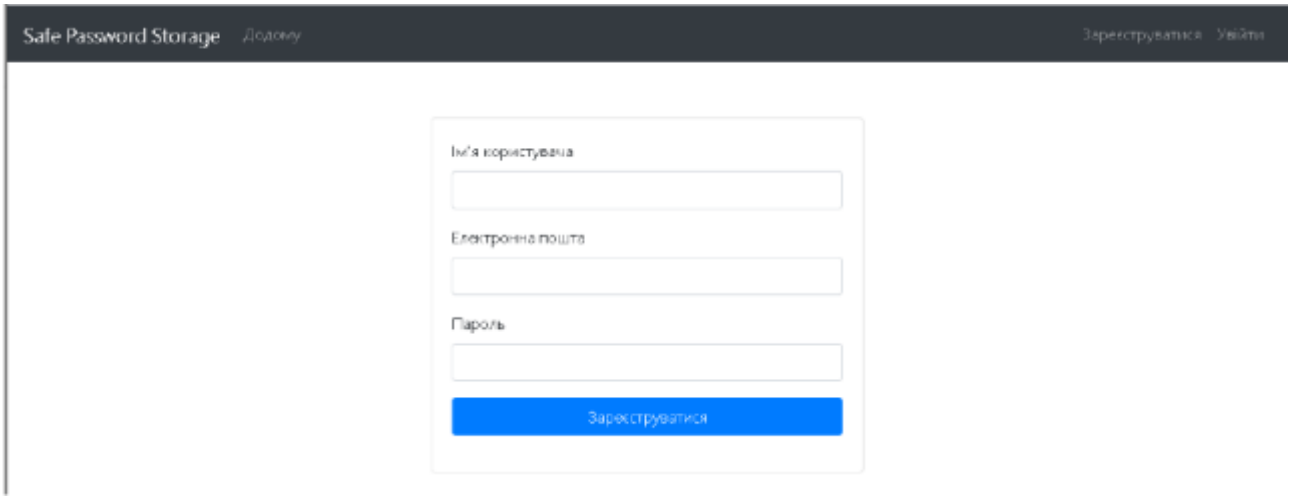


Рис.3.15 Сторінка реєстрації



Рис.3.16 Сторінка авторизації

Одним з функціоналів роботи сайту є перевірка на валідність введених даних клієнта (рис. 3.17).

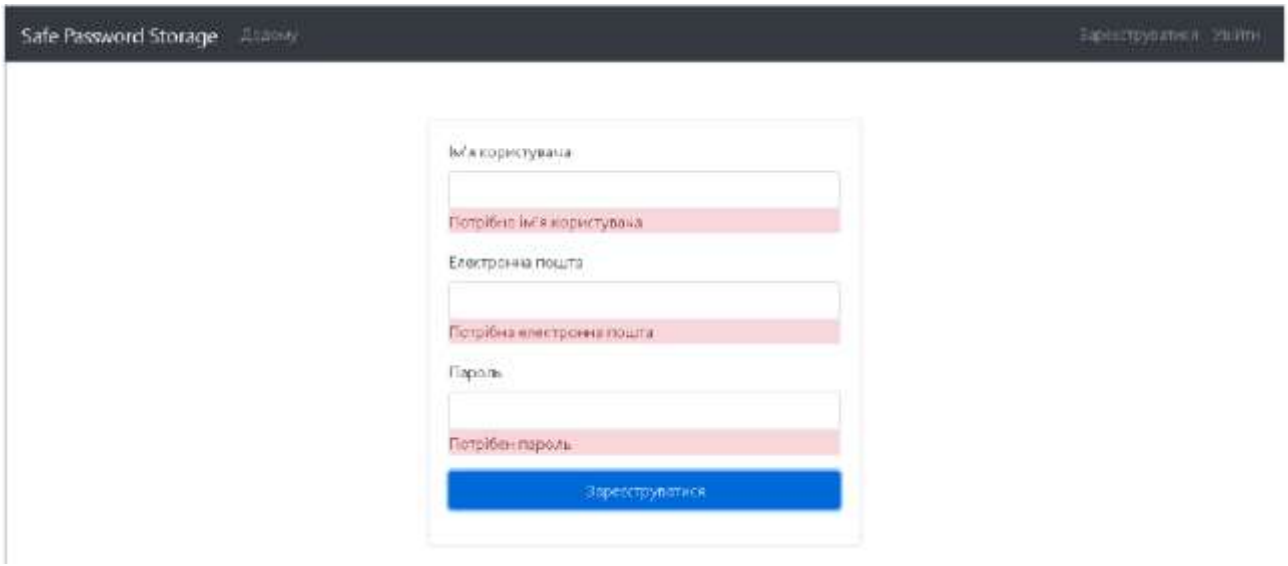


Рис. 3.17 Скріншот роботи перевірки на валідність

Ім’я та логін клієнта під час реєстрації проходять перевірку на сумісність в базі даних для виявлення подібності (рис. 3.18).

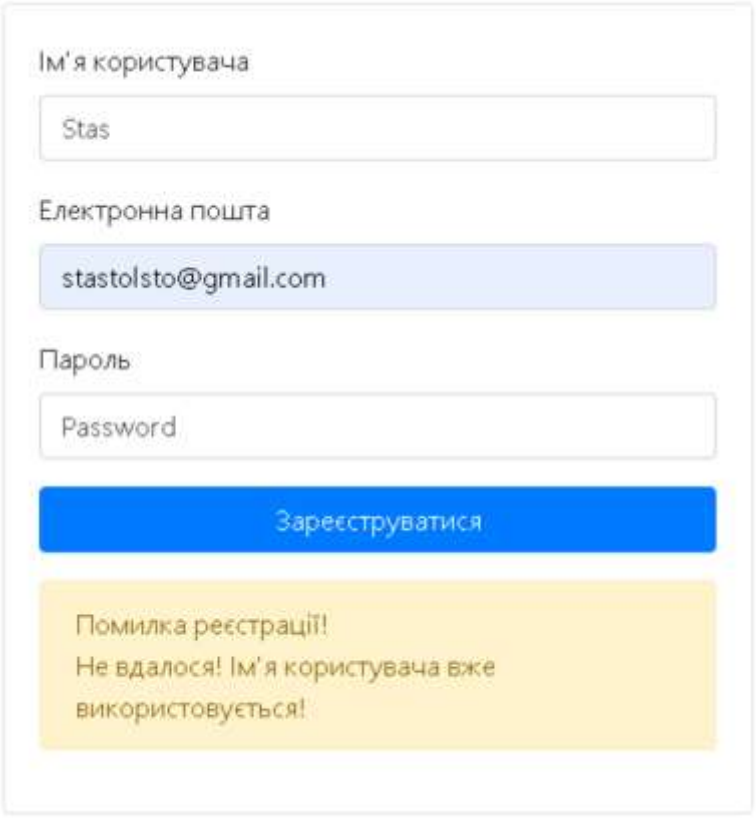


Рис. 3.18 Скріншот повідомлення про те, що ім’я вже існує

Після реєстрації користувач може здійснити вхід через сторінку авторизації (рис. 3.19 – 3.20).

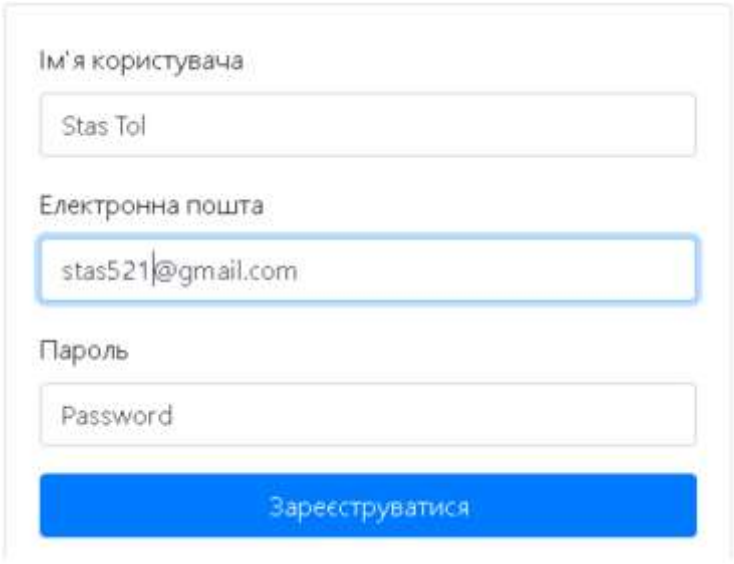


Рис.3.19 Скріншот реєстрації користувача

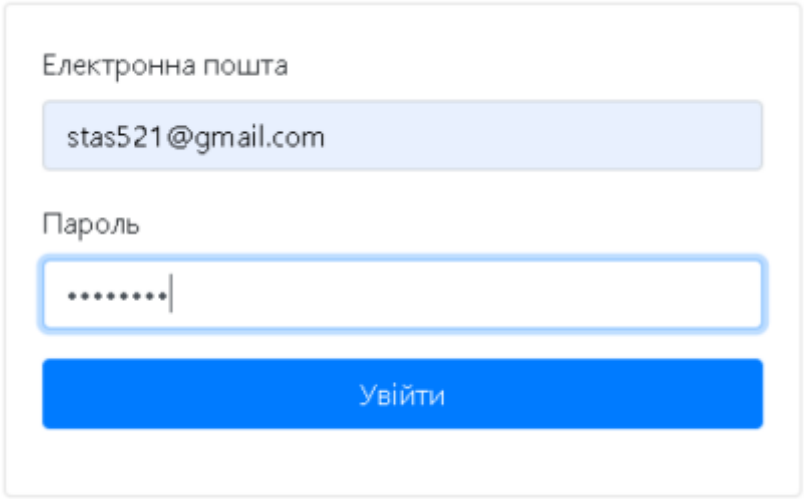


Рис. 3.20 Скріншот авторизації користувача

Після успішної авторизації користувач матиме змогу перейти на такі сторінки як створення паролю та звіту про помилку (рис. 3.21 – 3.22).

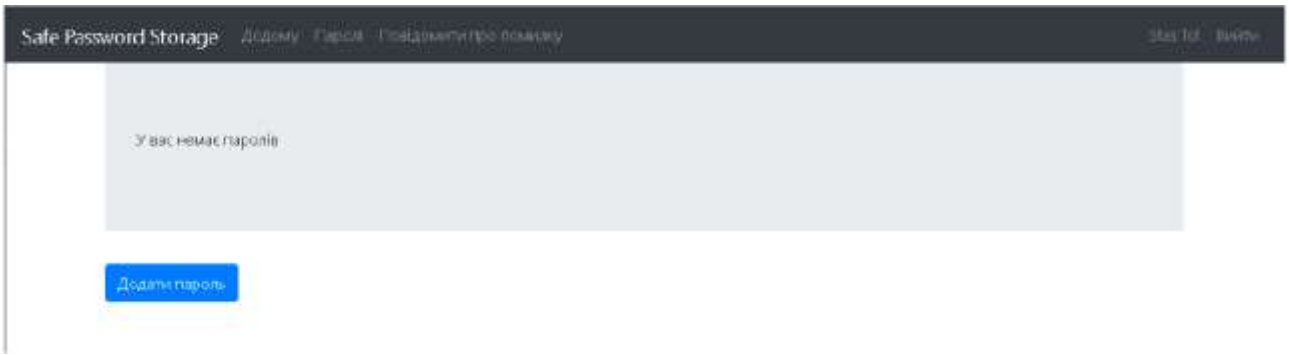


Рис.3.21 Сторінка взаємодії з паролем



Рис. 3.22 Сторінка створення повідомлення про помилку

Користувач, який був більше ніж 15 хвилин на сайті, отримає повідомлення про те, що час роботи токену завершився (рис. 3.23).

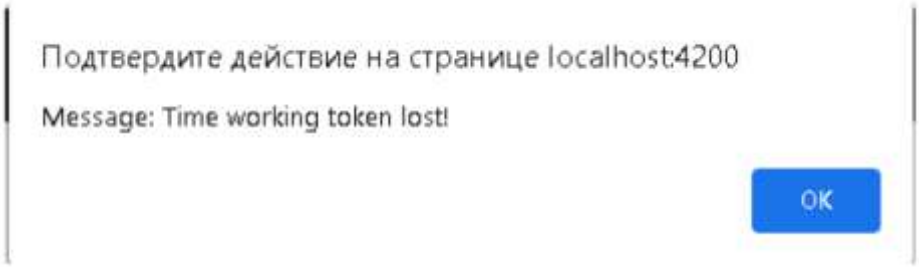


Рис. 3.23 Повідомлення про завершення роботи токена

На сторінці взаємодії з паролем користувачу спочатку потрібно створити пароль (рис. 3.24). Після підтвердження введених даних користувач отримає унікальний пароль (рис. 3.25).

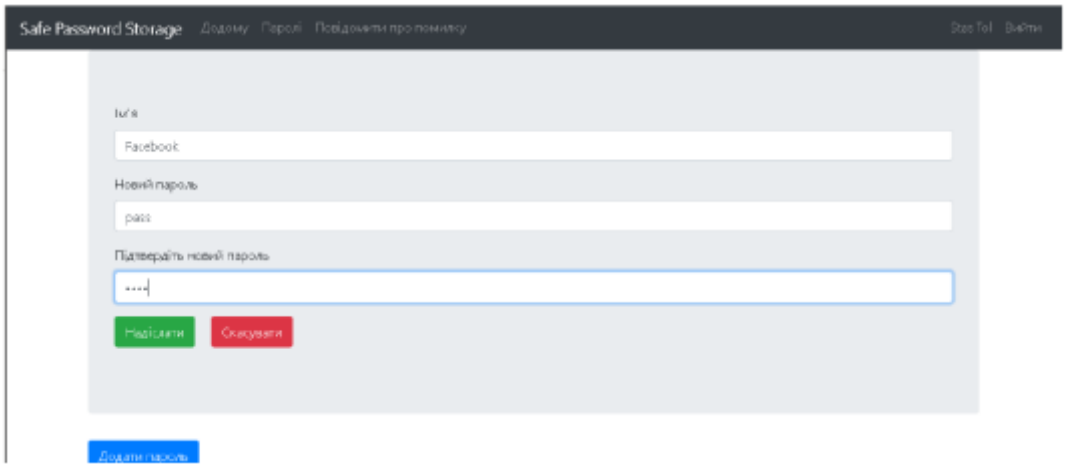


Рис.3.24 Форма створення пароля

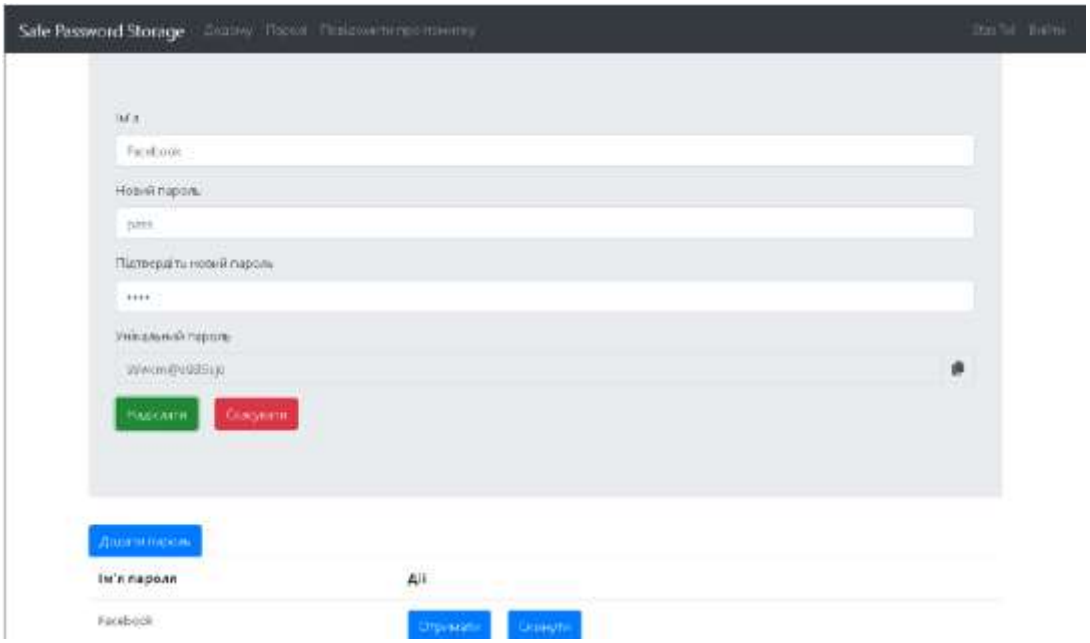


Рис. 3.25 Результат створення пароля

Після повторного завантаження сторінки з створеними паролями (рис. 3.26) користувач матиме змогу знову отримати унікальний пароль якщо введе вірно дані звичайного пароля (рис. 3.27).

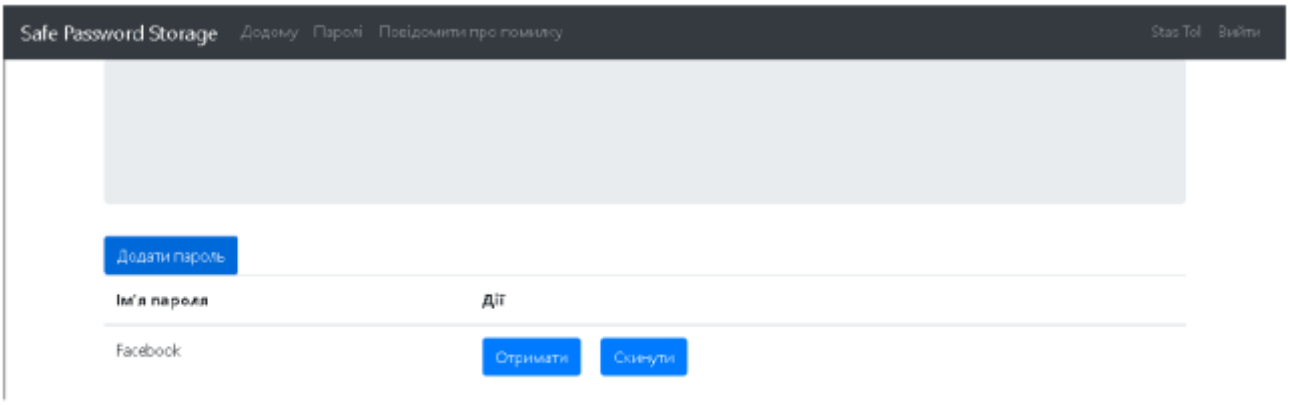


Рис. 3.26 Оновлена сторінка паролів

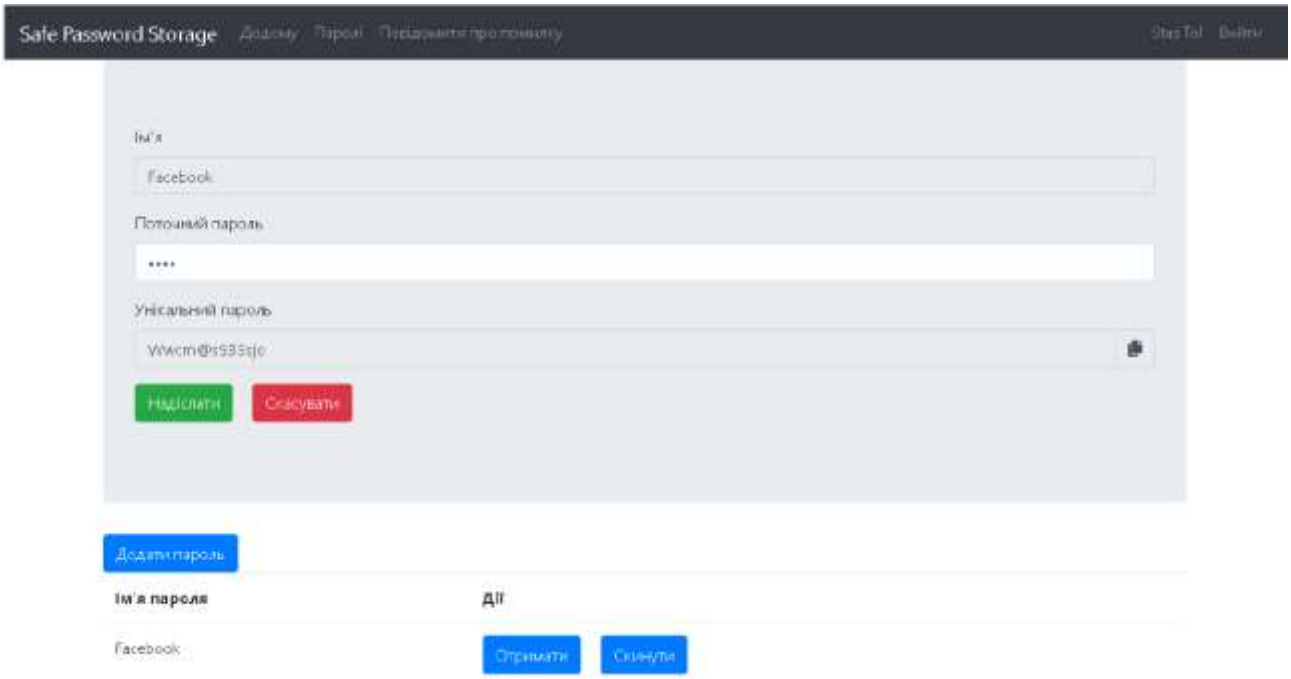


Рис. 3.27 Результат отримання пароля

Користувач має змогу змінити існуючий пароль (рис. 3.28 – 3.29).

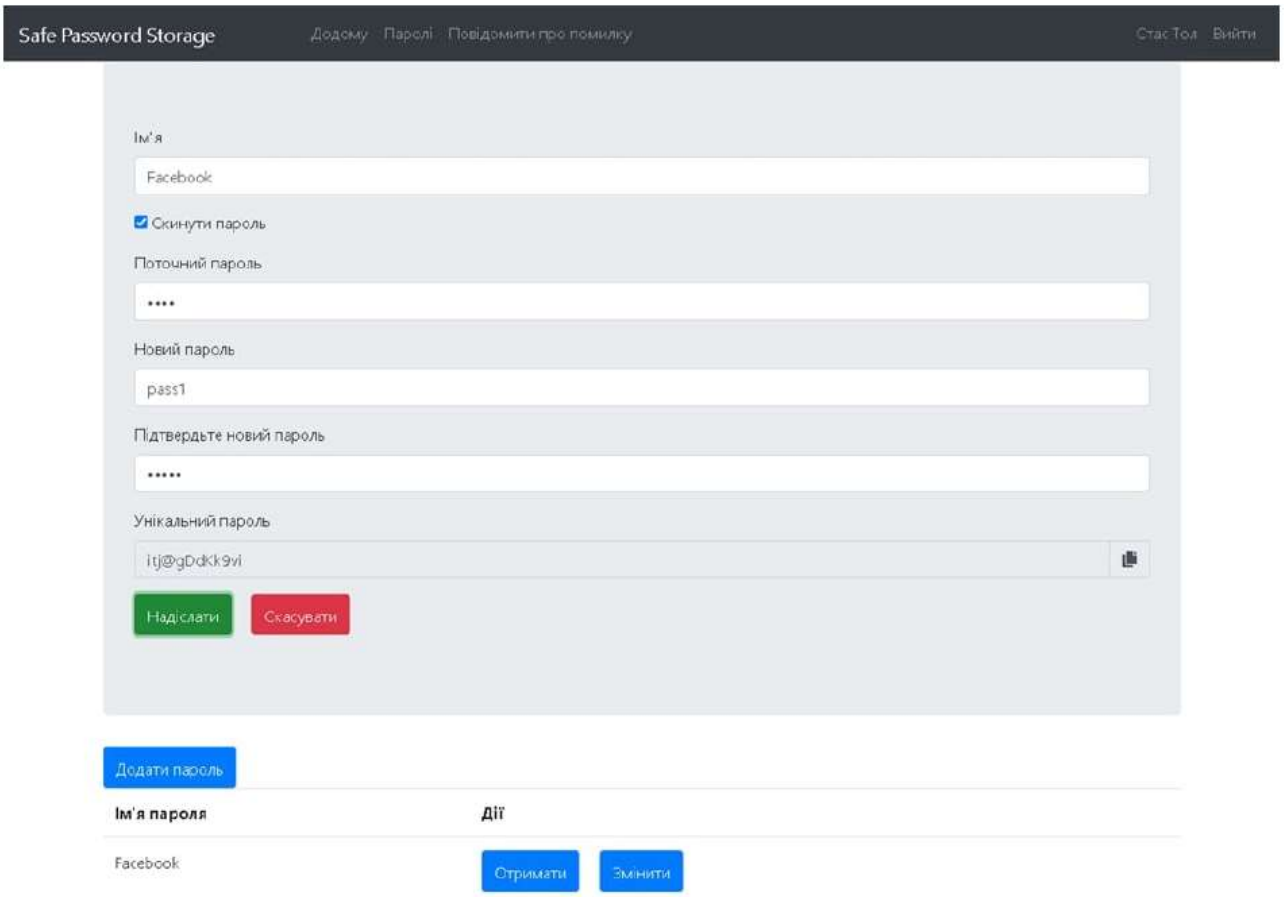


Рис. 3.28 Форма зміни існуючого пароля

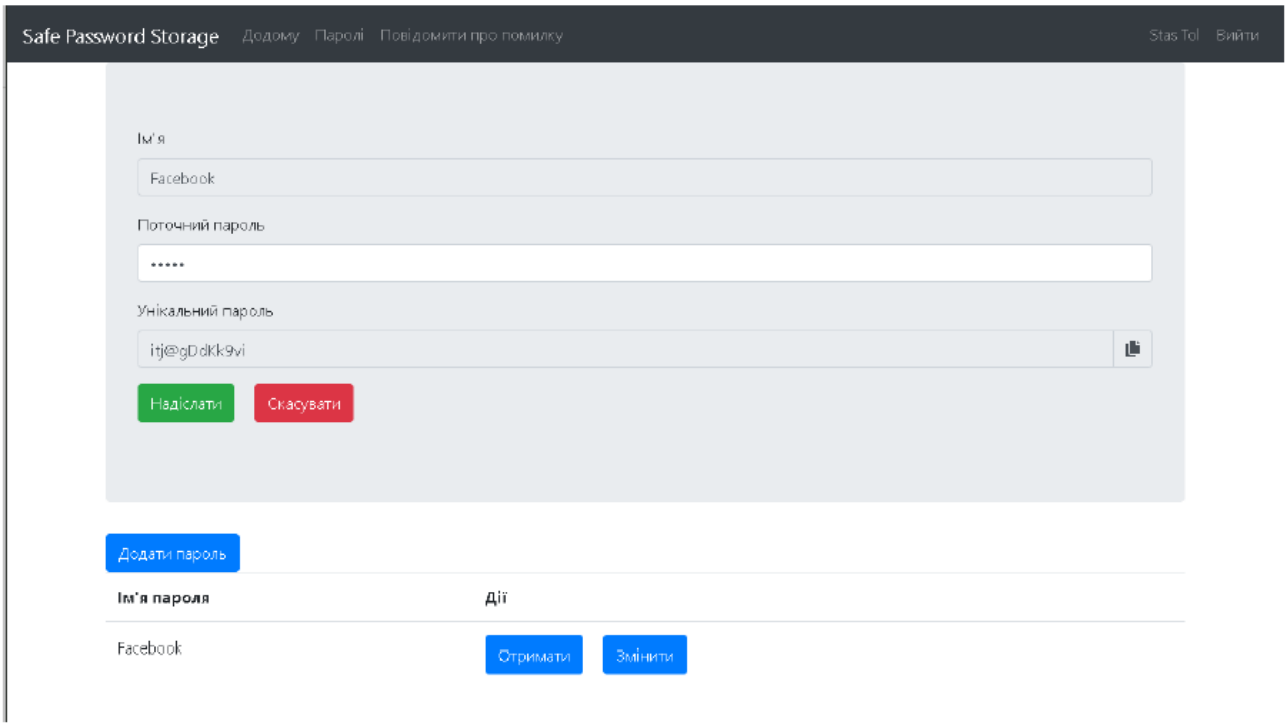


Рис. 3.29 Перевірка пароля після зміни

Якщо користувач введе звичайний пароль 5 раз невірно то система автоматично його заблокує (рис. 3.30).

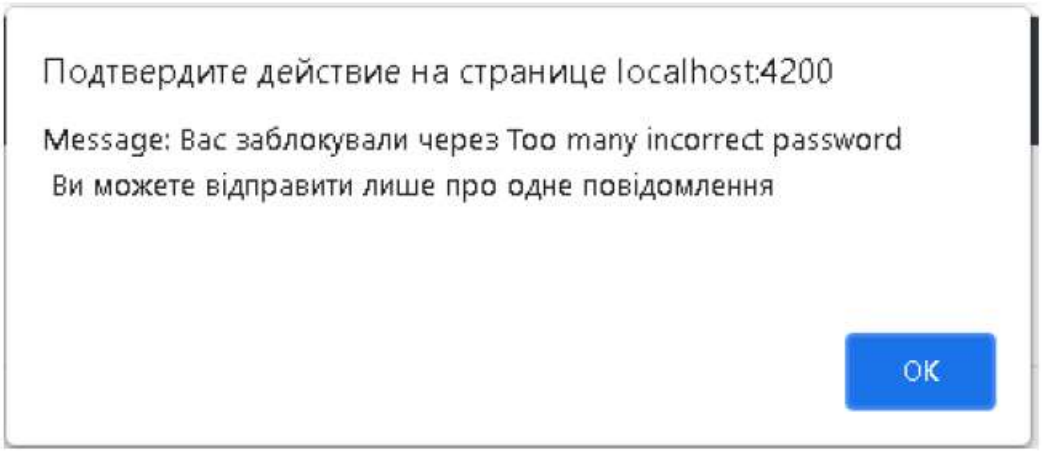


Рис. 3.30 Повідомлення про блокування користувача

Заблокований користувач має змогу відправити лише одне повідомлення про блокування адміністратору (рис. 3.31 – 3.34).

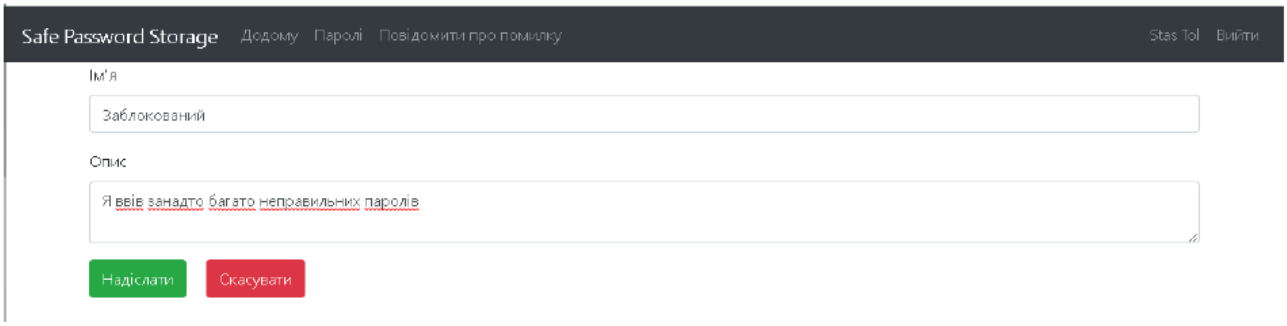


Рис.3.31 Форма зворотного зв’язку

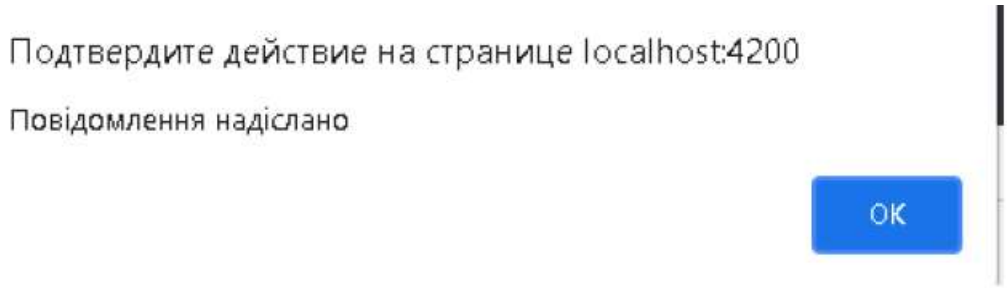


Рис.3.32 Повідомлення про відправку повідомлення

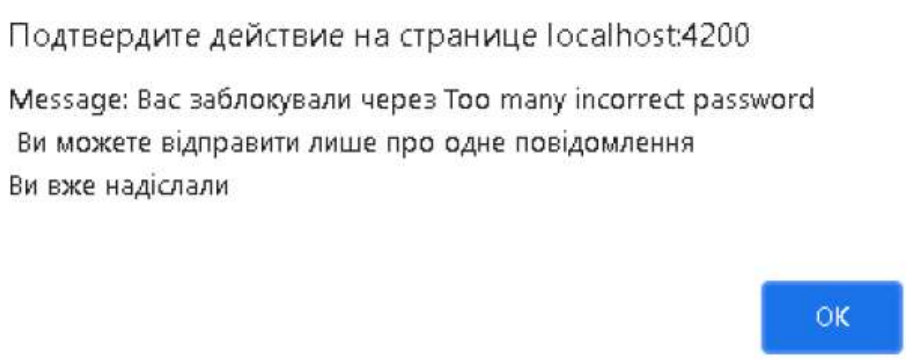


Рис.3.33 Повідомлення про відмову на повторне відправлення

В меню сторінки адміністратора (рис. 3.34) відображаються переходи на сторінки зворотного зв’язку з користувачем (рис. 3.35) або системою (рис. 3.36) та список заблокованих користувачів (рис. 3.37).



Рис.3.34 Меню сторінки адміністратора

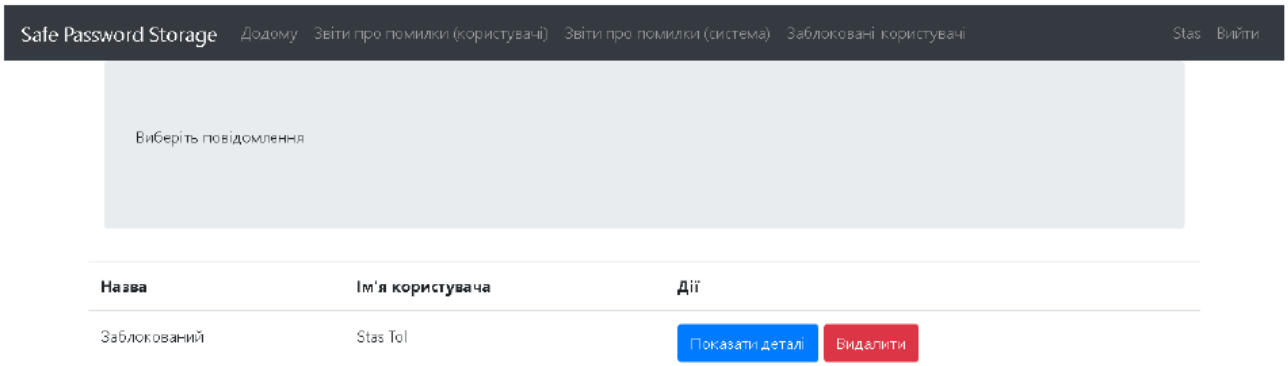


Рис.3.35 Сторінка зворотного зв’язку з користувачем

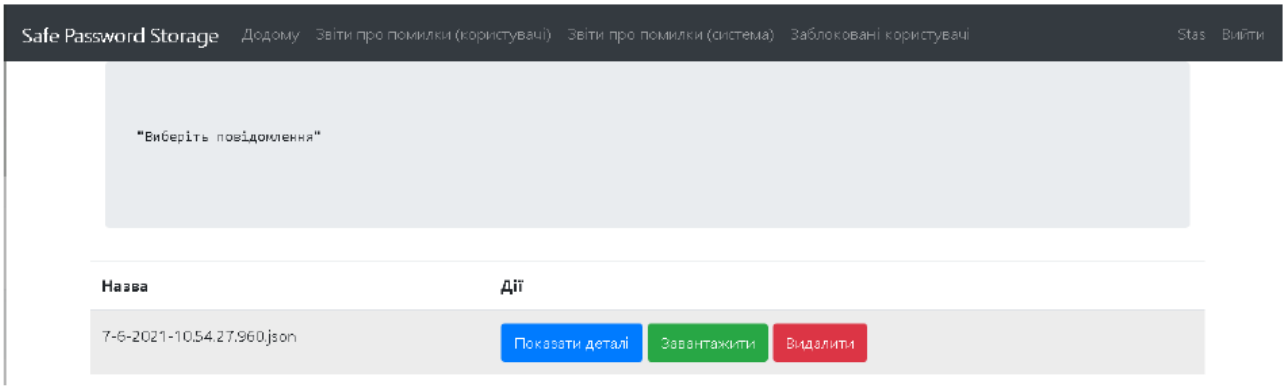


Рис.3.36 Сторінка зворотного зв’язку з системою

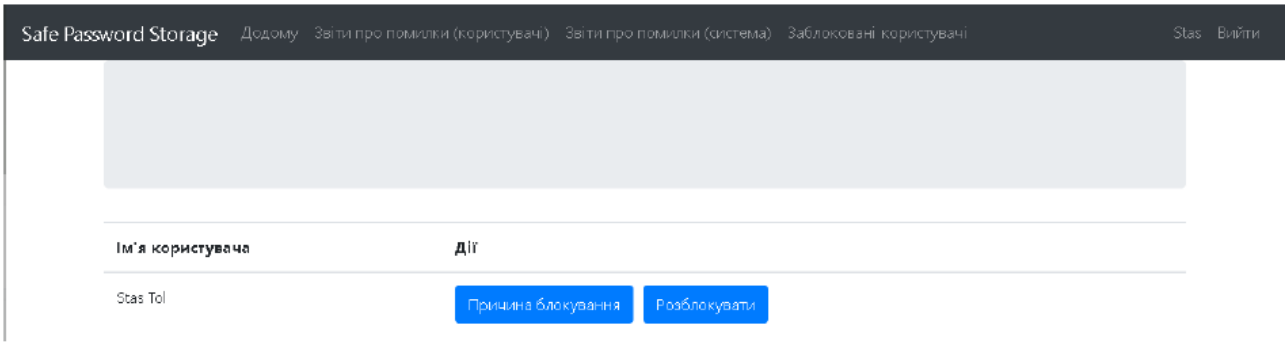


Рис. 3.37 Сторінка списку заблокованих користувачів

На сторінці зворотного зв’язку з користувачем адміністратор має змогу переглянути детальну інформацію повідомлення від користувача (рис. 3.38) та видалити його (рис. 3.39).

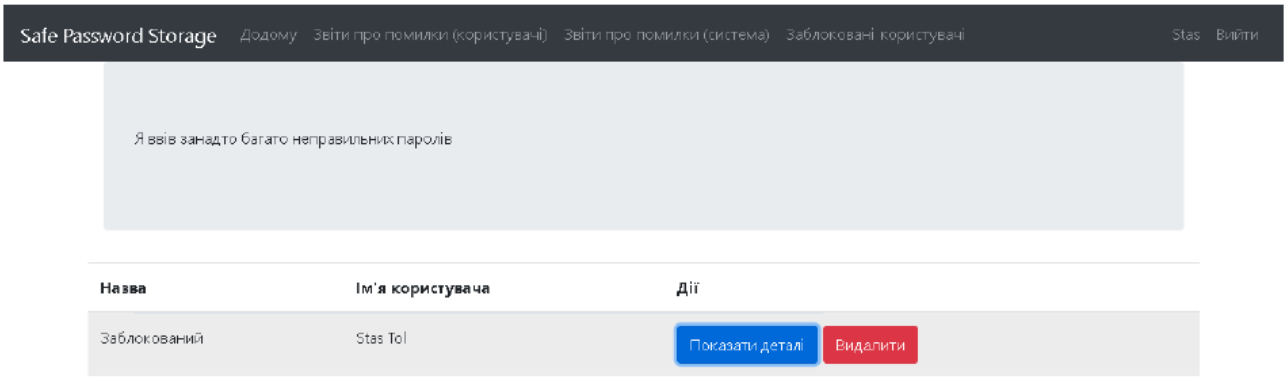


Рис. 3.38 Детальна інформація повідомлення

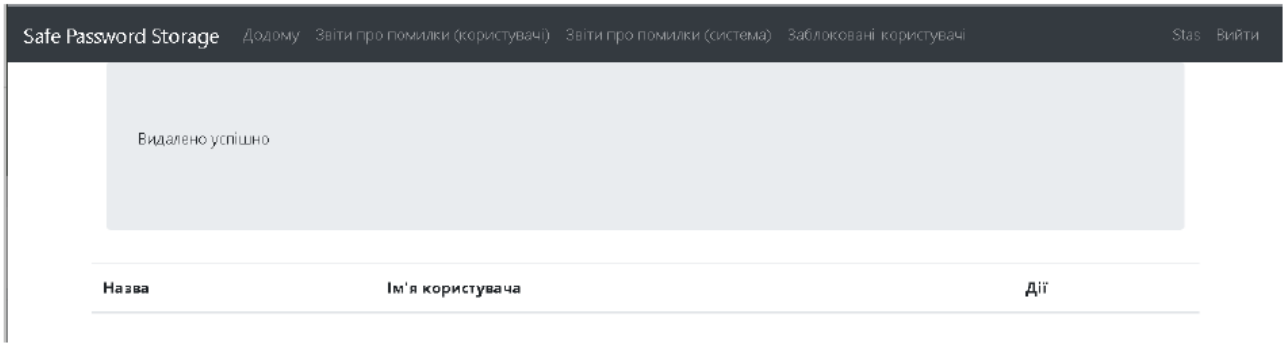


Рис. 3.39 Результат після видалення повідомлення

Сторінка зворотного зв'язку з системою дає змогу адміністратору переглянути детальну інформацію про помилку (рис. 3.40), завантажити цей файл (рис. 3.41) та видалити його (рис. 3.42).

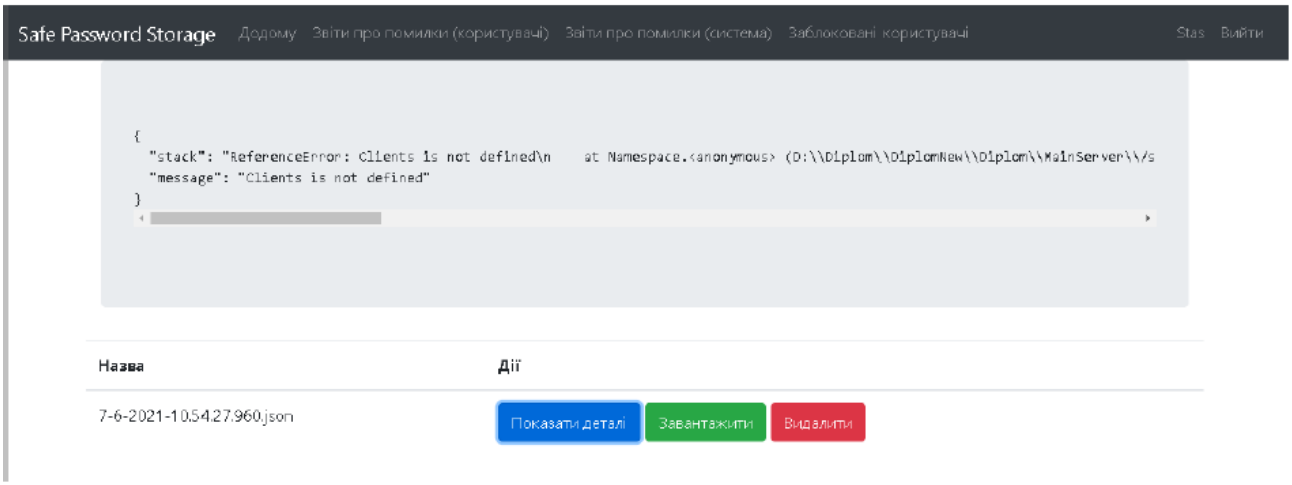


Рис. 3.40 Детальна інформація повідомлення про помилки в системі

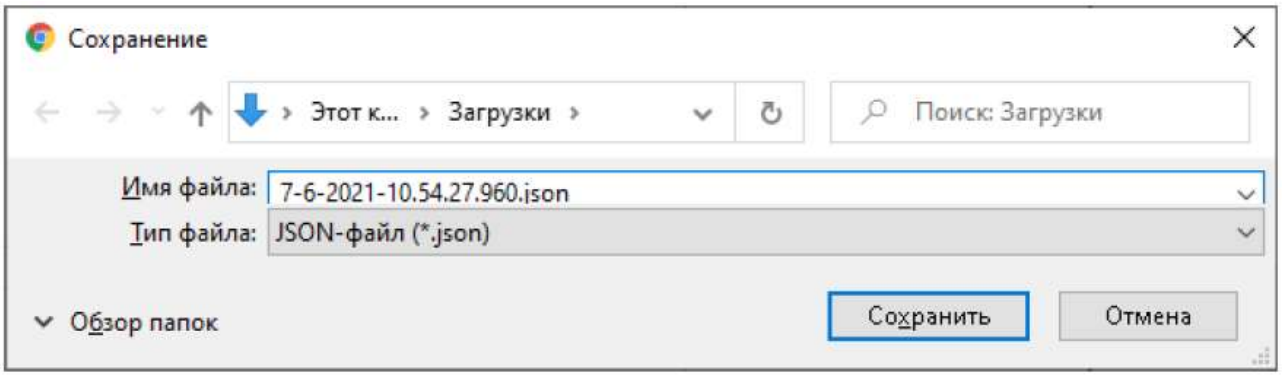


Рис. 3.41 Завантаження файлу

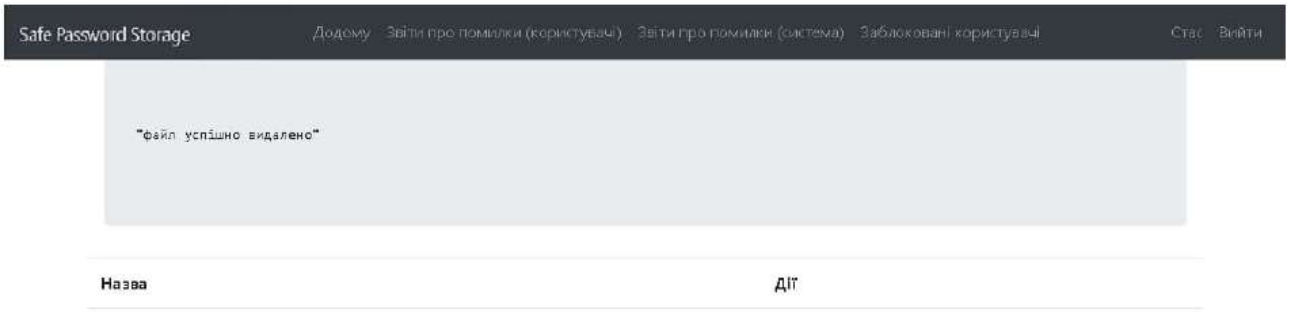


Рис. 3.42 Результат після видалення файлу

На сторінці з відображенням списку заблокованих користувачів адміністратор має змогу переглянути детальну інформацію про причину блокування користувача (рис. 3.43), а також зняти обмеження блокування з користувача (рис. 3.44-3.45).

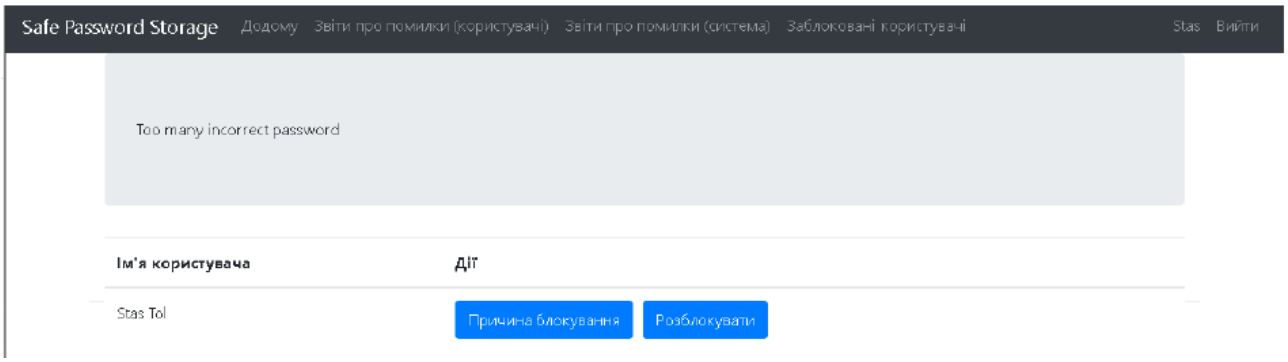


Рис.3.43 Детальна інформація про заблокованого користувача

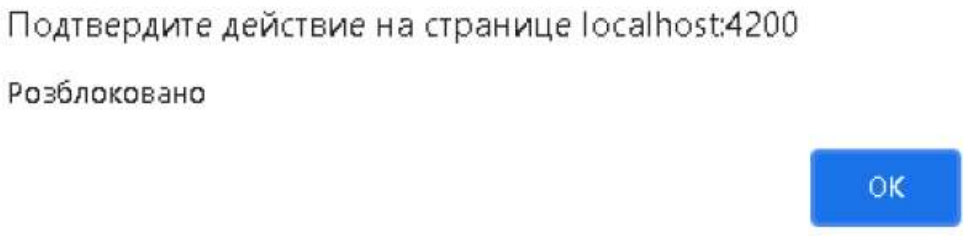


Рис.3.44 Повідомлення про зняття обмеження з користувача

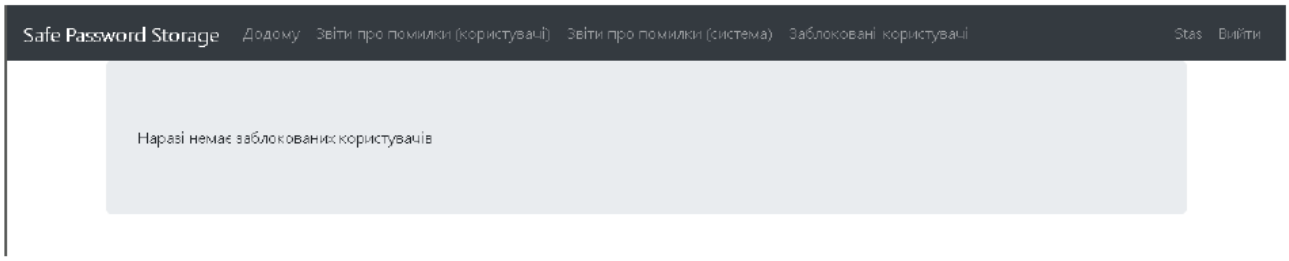


Рис. 3.45 Результат

На даний момент даний проект реалізовано у вигляді web-додатку. Він оптимізований під мобільні пристрої (рис. 3.46).

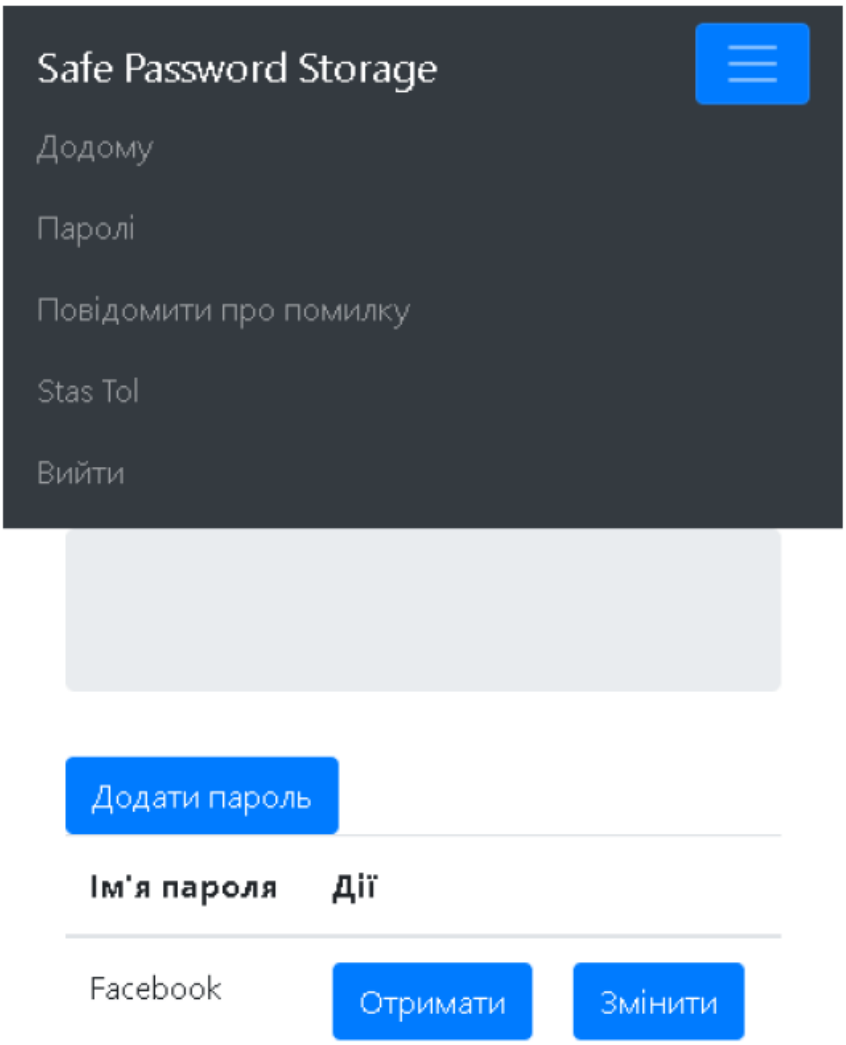


Рис. 3.46 Вигляд web-сторінки на мобільному пристрою

Всі запити на сервер даного проекту були протестовані за допомогою Postman (рис. 3.47), приклад тесту авторизації зображено на рисунку 3.48. Він дозволив перевірити вірність обробки сервером значень, що були відправлені під час запитів.

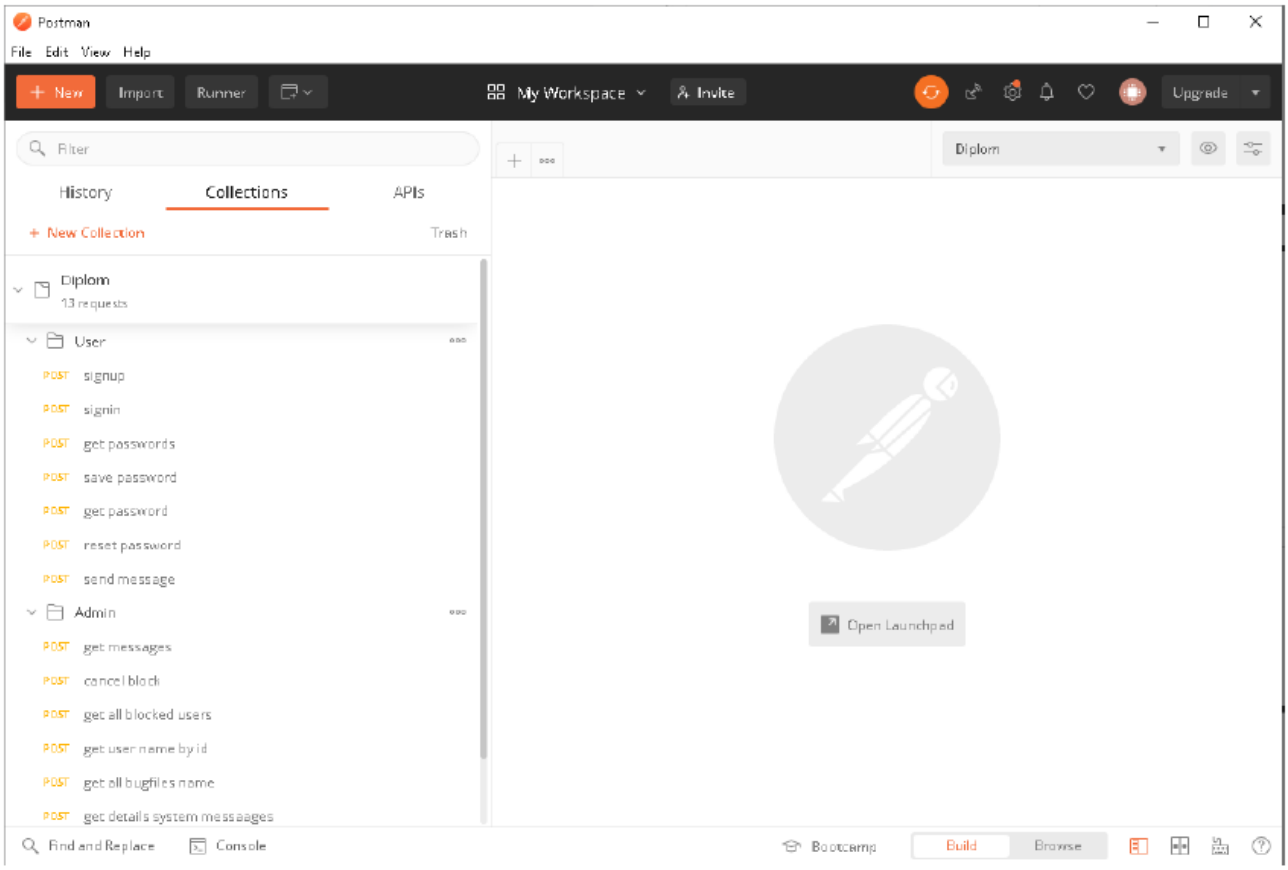


Рис. 3.47 Програма Postman

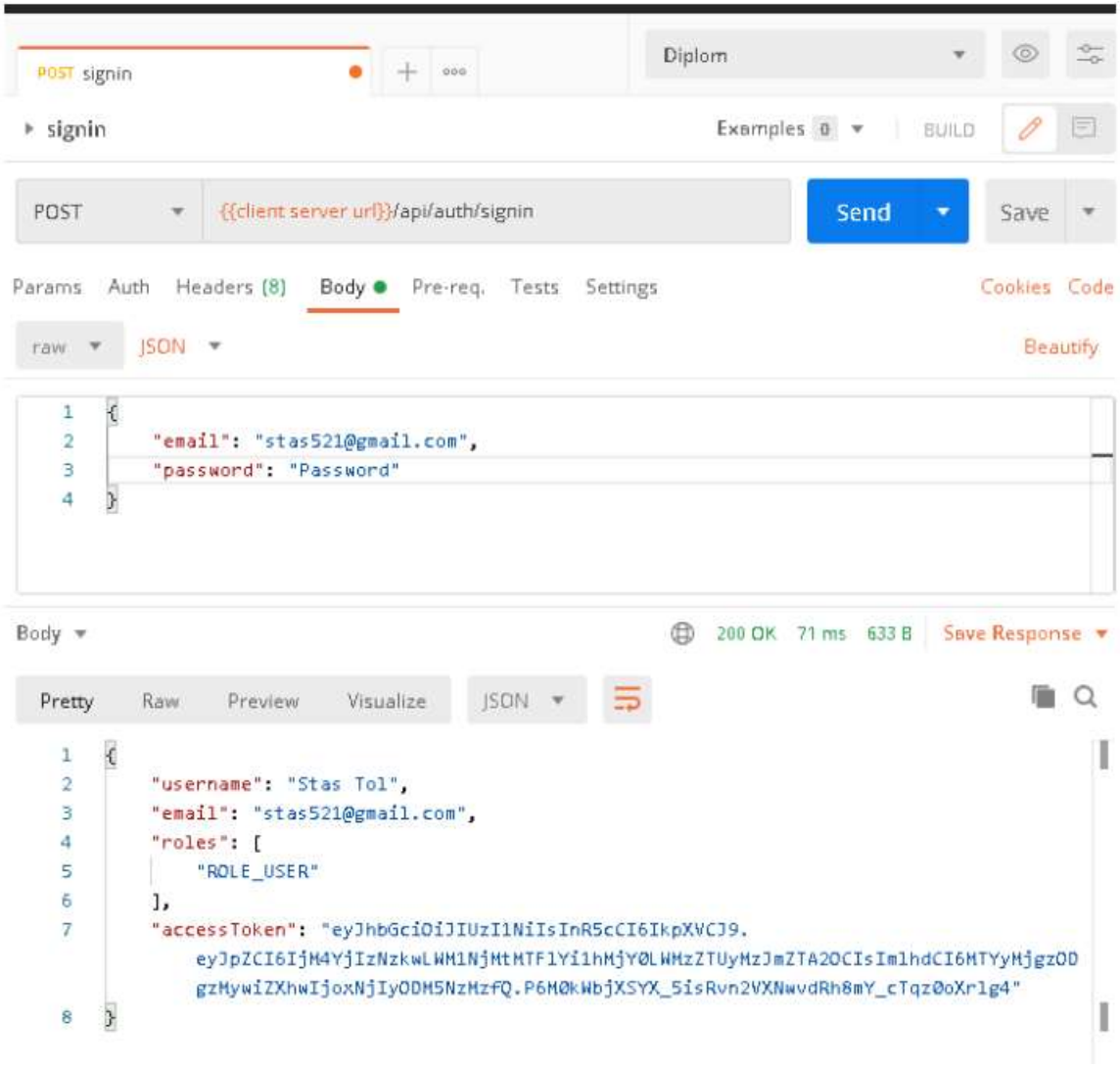


Рисунок 3.48 Приклад тестування запиту на авторизацію

Декілька паролів були перевірені на складність взлому на сайті https://www.my1login.com/resources/password-strength-test/ і отримали високу оцінку надійності (рис. 3.49).

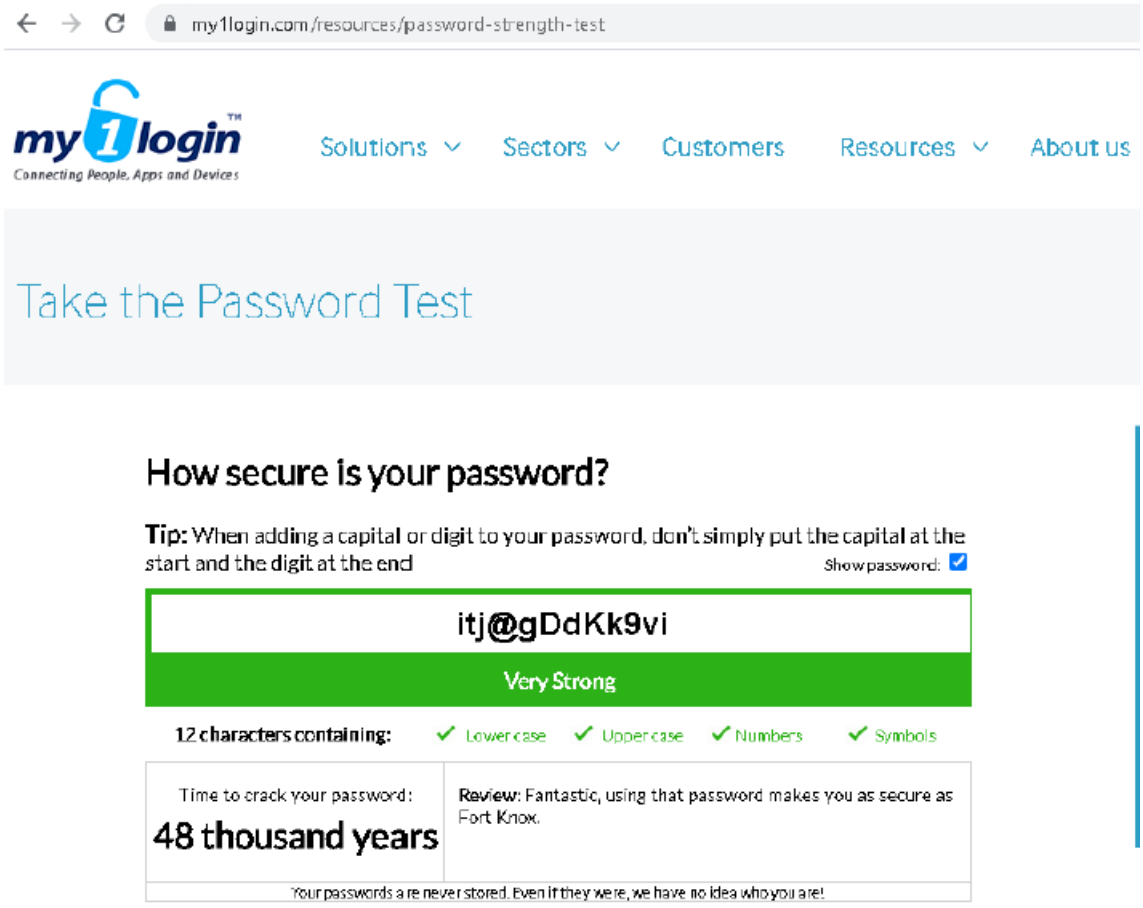


Рис. 3.49 Приклад тестування паролю

ВИСНОВОК

Під час проектування кваліфікаційної роботи бакалавра було проведено аналіз наявних аналогічних рішень на ринку, розглянуті їх сильні та слабкі сторони. Майже відсутність менеджерів для унікального збереження паролів в базі даних змусила розробити власну web-додаток.

Для реалізації продукту проекту з функцією не збереження паролів в базі даних на стороні клієнта була обраний фреймворк Angular, а серверна частина складається з двох серверів, що реалізовані за допомогою фреймворк Node.js. Для вирішення даної задачі з використанням баз даних було створено два сервери:

– перший – клієнтський сервер з використанням бази даних MySQL;

– другий – головний для збереження хеш-кодів.

Для роботи головного сервера було використано базу даних mongoDB. MongoDB не вимагає опису схеми таблиць в порівнянні з реляційними базами даних. Особливістю його роботи є те, що дані зберігаються у вигляді колекцій де структура даних та документів не обов'язково повинна бути схожа.

Роботу серверів реалізовано у вигляді REST API, та забезпечено синхронізацію з базою даних. Для авторизації користувача було використано токен і додано адміністративний інтерфейс, що дало змогу створювати функціонал для будь-якої системи. В процесі роботи також було використано протокол HTTPS для захисту від MITM атак.

Одним з головних етапів створення проекту “Програма генерації та збереження паролів “Safe Password Storage” є його моделювання за допомогою мови моделювання UML. Розроблення та побудова різних видів діаграм допомогло створити етапи розробки програми з отриманням уявлення про кінцевий результат додатку.

Створення програми генерації та збереження паролів “Safe Password Storage” на сьогодні має важливе значення. Захист конфіденційної інформації користувачів – один з пріоритетів розвитку IT-технології сучасного світу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. A. Gordo, J. Almazan, J. Revaud, and D. Larlus. Deep image ´ retrieval: Learning global representations for image search. In European Conference on Computer Vision, 2016.

2. Найважливіші недоліки безпеки, знайдені в LastPass у Chrome URL: https://uk.allsoftfree.com/criticalsecurity-flaws-found-lastpass-chrome-33913

3. Revisiting Security Vulnerabilities in Commercial Password Managers URL: http://eprints.whiterose.ac.uk/158056/8/Revisiting_Security_Vulnerabilities_in_Com me rcial_Password_Managers_2.pdf

4. Плюси і мінуси Dashlane URL: https://sravni.cc/reviews/plyusy-i-minusy-programmy-dashlane/

5. Плюси і мінуси KeePass URL: https://sravni.cc/reviews/plyusy-i-minusy-keepass-kipas/

6. МЕТОДОЛОГІЯ IDEF0 URL: https://stud.com.ua/87184/ekonomika/metodologiya_idef0 .

7. Діаграма класів URL: https://uk.photo-555.com/7917237-class-diagram .

8. Діаграма діяльності UML URL: http://pingwin.lg.ua/post/diagrama/uk/deatelnost-diagrama-dialnosti-umlplanerka-onlajn-skola-kreativnosti.htm

9. UML URL: https://www.ibm.com/docs/en/rational-soft-arch/9.7.0?topic=diagrams-uml-models

10. Діаграми потоків даних- DFD URL: https://thelib.info/tehnologii/3215921-diagrami-potokiv-danih-dfd/

11. Етапи проектування та моделювання БД URL: http://ua.textreferat.com/referat-23369-3.html