**МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»**

**НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ**

**КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ В ЕНЕРГЕТИЦІ**

КУРСОВА РОБОТА

**з дисципліни:** ««Основи Веб-програмування»

**на тему:** «Додаток, який створює сильні та безпечні паролі»

|  |  |  |
| --- | --- | --- |
| **Керівник:**  Гагарін. О.О.  «Допущено до захисту»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (особистий підпис керівника)  «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 20\_\_ р. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Захищено з оцінкою  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оцінка) |  | **Виконав**:  Семенов Максим Михайлович  Студент 2 курсу  Групи ТВ-32  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (особистий підпис виконавця)  «\_\_\_» \_\_\_\_\_\_\_\_ 20\_\_ р. |

Посилання на **GitHub** репозиторій: <https://github.com/max-s-m/web-kursova>

**Київ – 2025**

**АНОТАЦІЯ**

Для виконання роботи обрано тему «Додаток, який створює сильні та безпечні паролі». Метою цього проєкту є розробка функціонального веб-застосунку, який дозволить як створювати сильні та складні паролі шляхом генерації, так їх і зберігати у надійному вигляді. Система поєднує у собі функції генерації паролів зі змінними параметрами під потреби користувача, а також зберігання паролів разом з логіном/ресурсом, до якого він є прив’язаний, у базі даних у зашифрованому базовим способом вигляді, що надасть більшу безпеку створених паролів.

Процес розробки та актуальність цієї роботи для користувачів будь-яких онлайн-сервісів, що потребують паролів, вказані у цій роботі у вступному розділі.

Пояснення деталей проєкту та особливостей реалізації вказані у розділах оновної частини. Там можна знайти опис можливостей програми, шляхів застосування, та процесу імплементації у схемах, діаграмах, та детальних описах сконструйованих систем.

Наступними розписані питання вибору технологій, що були застосовані у ході виконання, такі як база даних, способи запису та захисту паролів, які користувач згенерував, або додав самостійно.

Завершуючий розділ містить висновки, зроблені після завершення процесів планування, розробки, та оцінки результатів роботи продукту.

**ABSTRACT**

The topic chosen for the project was “An application that creates strong and secure passwords”. The purpose of this project is to develop a functional web application that will allow both creating strong and complex passwords by generating them and storing them in a secure manner. The system combines the functions of generating passwords with variable parameters according to the user's needs, as well as storing passwords along with the login/resource to which it is linked in a database in a basic encrypted form, which will provide greater security for the created passwords.

The development process and the relevance of this work for users of any online services that require passwords are described in the introductory section of this paper.

Explanations of the project details and implementation features are given in the sections of the update part. There, you can find a description of the program's capabilities, ways of using it, and the implementation process in diagrams, charts, and detailed descriptions of the designed systems.

The following sections describe the choice of technologies used in the course of implementation, such as the database, methods of recording and protecting passwords generated by the user or added by the user.

The final section contains the conclusions drawn after completing the planning, development, and evaluation of the product.

**ЗМІСТ**

АНОТАЦІЯ.................................................................................................................... 1

ABSTRACT.................................................................................................................... 2

ЗМІСТ............................................................................................................................. 3

ВСТУП .......................................................................................................................... 4

1. СТВОРЕННЯ ДІАГРАМ КОМПОНЕНТІВ, ВЗАЄМОДІЇ ТА КЛАСІВ ВЕБ-ДОДАТКУ...................................................................................................................... 5

2. ПРОЕКТУВАННЯ СТРУКТУРИ МЕНЮ З УРАХУВАННЯМ ПОТРЕБ КОРИСТУВАЧА............................................................................................................ 9

3. РОЗМІЩЕННЯ ТА СТИЛІЗАЦІЯ ЕЛЕМЕНТІВ ІНТЕРФЕЙСУ....................... 11

4. ПРОЕКТУВАННЯ МОДЕЛЕЙ ДАНИХ ДЛЯ ЗБЕРЕЖЕННЯ ІНФОРМАЦІЇ. 13

5. СТВОРЕННЯ ДІАГРАМ КОМПОНЕНТІВ, ВЗАЄМОДІЇ ТА КЛАСІВ ВЕБ-ДОДАТКУ.................................................................................................................... 14

6. РОЗРОБКА БАЗИ ДАНИХ ДЛЯ ЗБЕРЕЖЕННЯ ТА ОТРИМАННЯ ІНФОРМАЦІЇ.............................................................................................................. 17

7. ІНТЕГРАЦІЯ СТОРОННІХ СЕРВІСІВ В СИСТЕМУ......................................... 18

8. СТВОРЕННЯ СЕРВЕРНОЇ АРХІТЕКТУРИ ДОДАТКУ..................................... 18

9. РЕАЛІЗАЦІЯ ОБРОБКИ ЗАПИТІВ ТА ВЗАЄМОДІЇ З БАЗОЮ ДАНИХ НА СЕРВЕРІ...................................................................................................................... 20

10. ТЕСТУВАННЯ ТА НАЛАШТУВАННЯ ВЕБ-ДОДАТКУ НА СЕРВЕРІ.........23

ВИСНОВКИ................................................................................................................ 27

ВИКОРИСТАНІ ДЖЕРЕЛА...................................................................................... 28

**ВСТУП**

На сьогоднішній день використання технологій розвинуто до рівню, на якому майже кожний користувач комп’ютерних технологй у світі неодноразово зіткався з необхідністю реєстрації з використанням паролю. Застосування паролів є поширеним та найпростішим методом для сайтів зберігати конфіденційнсть користувачів, проте слабкі паролі є вразливістю. Через значну кількість ресурсів, до яких людина може мати пароль та/або бажати надійніший, доречним є використання застосунку для швидкої генерації надійних паролів та зберігання їх. Такі умови являють собою привід до написання цієї роботи, а також описують її актуальність, оскільки застосунок проєкту пропонує функціонал генерації, видалення, перегляду, та зберігання паролів у безпечному, зашифрованому форматі.

В процесі розробки системи було спроектовано застосунок, розглянуто методи шифрації, способи генерації, та обрано метод зберігання. Для проєкту імплементовано базу даних, яка відповідає потребам кінцевого продукту. Користувацькі дані в цьому випадку є персональними та важливими, тому реалізовано шифрацію та дешифрацію даних, за допомогою унікального ключа доступа до всіх згенерованих/доданих вручну паролів.

Програму протестовано та проаналізовано результати роботи на коректність та базову зрозумілість і повноту інтерфейсу.

**1. Створення діаграм компонентів, взаємодії та класів веб-додатку.**

A black screen with white ovals

AI-generated content may be incorrect.

Рисунок 1.1 - Діаграма варіантів використання

A screenshot of a computer

AI-generated content may be incorrect.

Рисунок 1.2 – Діаграма послідовності генерації паролю

A diagram of a company

AI-generated content may be incorrect.

Рисунок 1.3 – Діаграма компонентів

A screenshot of a computer

AI-generated content may be incorrect.

Рисунок 1.4 – Діаграма класів

1. Користувачеві буде можливий доступ до сховища паролів:
   1. Екран для введення майстер-ключа:
      1. При першому використанні з новим ключем – створення сховища.
      2. При наступних – розшифрування даних для доступу.
   2. Перевірка коректності майстер-ключа через розшифрування спеціального маркера.
   3. Відображення повідомлень про помилку введення ключа або статус операції.
   4. Функція блокування сховища (вимагає повторного введення ключа).
2. Користувач зможе генерувати паролі, для чого буде реалізовано:
   1. Форма налаштування параметрів генерації:
      1. Введення довжини пароля.
      2. Введення кількості символів латиниці, кирилиці, спеціальних символів.
      3. Введення списку дозволених спеціальних символів.
      4. Автоматичне додавання цифрових символів для досягнення заданої довжини.
   2. Алгоритм генерації пароля на основі введених параметрів.
   3. Відображення згенерованого пароля.
3. Можна буде редагувати записи паролів:
   1. Додавання нового запису (логін + пароль):
      1. Форма для введення власного логіна та пароля.
      2. Форма для введення логіна та збереження раніше згенерованого пароля.
      3. Шифрування логіна та пароля на стороні клієнта майстер-ключем перед збереженням.
   2. Перегляд збережених паролів:
      1. Відображення списку пар логін/пароль після введення майстер-ключа.
      2. Розшифрування даних на стороні клієнта перед відображенням.
   3. Видалення запису пароля:
      1. Можливість видалення запису за введеним логіном.
      2. Шифрування введеного логіна майстер-ключем для пошуку відповідного запису.
   4. Відображення повідомлень про статус операцій (додавання, видалення).
4. Частиною функціоналу буде зберігання та шифрування даних:
   1. Шифрування/дешифрування логінів та паролів на стороні клієнта за допомогою майстер-ключа. Майстер-ключ не передається на сервер у відкритому вигляді.
   2. Алгоритм шифрування на основі динамічної мапи заміни символів, залежної від майстер-ключа.
   3. Зберігання зашифрованих версій логінів та паролів у базі даних MySQL.
   4. Зберігання на сервері спеціального маркера, зашифрованого майстер-ключем, для валідації ключа.
5. Користувач матиме простий інтерфейс, який включатиме:
   1. Поділ функціоналу на екрани: введення ключа, головне меню, перегляд, редагування, генерація.
   2. Візуальне оформлення з можливістю встановлення фонового зображення та відображенням логотипу.
   3. Адаптивність інтерфейсу для різних розмірів екранів.

**2. Проектування структури меню з урахуванням потреб користувача**

Основна взаємодія користувача з програмою "Генератор Паролів" відбувається через послідовність екранів та меню, що відображаються залежно від поточного стану програми (наприклад, чи введено майстер-ключ) та обраних дій.

Після успішного введення майстер-ключа, користувачу буде доступне Головне Меню, яке включатиме наступні пункти:

* Переглянути паролі
* Додати/видалити пароль
* Заблокувати сховище (Вихід)

A diagram of a circle and a circle

AI-generated content may be incorrect.

Рисунок 2.1 - Схематичне зображення головного меню програми

Опис пунктів Головного Меню та пов'язаних з ними екранів/дій:

1. Переглянути паролі:
   1. Цей пункт меню переведе користувача на екран відображення списку збережених логінів та паролів.
   2. Дані будуть розшифровані поточним майстер-ключем та представлені у зручному для читання форматі.
   3. З цього екрану буде доступна кнопка для повернення до головного меню.
2. Додати/видалити пароль:
   1. Цей пункт меню переведе користувача на екран "Редагування паролів", який є підменю з наступними опціями:
      1. Додати пароль (переводить на екран "Додавання паролю", де користувач може обрати додавання або генерування паролю).

A screenshot of a phone

AI-generated content may be incorrect.

Рисунок 2.2 – Схематичне зображення меню генерації

* + 1. Видалити пароль (переводить на екран "Видалення паролю", де користувачеві буде запропоновано ввести логін/ресурс, запис якого необхідно видалити.
    2. Повернутися до Головного Меню: Закриває екран "Редагування паролів" та повертає користувача до Головного Меню.

1. Заблокувати сховище (Вихід)
   1. Цей пункт меню здійснює вихід з поточного сеансу роботи зі сховищем.
   2. Майстер-ключ скидається, і для подальшого доступу до даних його потрібно буде ввести знову (користувач перенаправляється на екран введення майстер-ключа).

**3. Розміщення та стилізація елементів інтерфейсу**

Інтерфейс програми "Генератор Паролів" складається з кількох екранів:

1. Екран введення Майстер-ключа:
   1. На екрані присутні:  
      - Поле для введення майстер-ключа.  
      - Кнопка "Розблокувати / Створити сховище".  
      - Область для відображення повідомлень (про помилку або успіх).

A screenshot of a computer screen

AI-generated content may be incorrect.

Рисунок 3.1 - Схема розміщення елементів інтерфейсу екрана введення майстер-ключа

1. Головне Меню (після розблокування):
   1. На екрані вертикально розміщені кнопки:  
      - "Переглянути паролі".  
      - "Додати/видалити пароль".  
      - "Заблокувати сховище (Вихід)".
2. Екран "Редагування паролів":
   1. На екрані вертикально розміщені кнопки:  
      - "Додати пароль".  
      - "Видалити пароль".  
      - "Повернутися до Головного Меню".
3. Екран "Додавання паролю":
   1. Кнопки вибору режиму:  
      - "Ввести свій пароль".  
      - "Згенерувати пароль".
   2. При виборі "Ввести свій пароль":  
      - Поле для введення логіна/ресурсу.  
      - Поле для введення пароля.  
      - Кнопка "Зберегти".
   3. При виборі "Згенерувати пароль":  
      - Поля для параметрів генерації (довжина, типи символів).  
      - Кнопка "Згенерувати".  
      - Область відображення згенерованого пароля.  
      - Поле для введення логіна/ресурсу (для згенерованого пароля).  
      - Кнопка "Зберегти згенерований пароль".
   4. Кнопка "Назад до Редагування".
   5. Область для повідомлень.

A screenshot of a phone

AI-generated content may be incorrect.

Рисунок 3.2 - Схема розміщення елементів інтерфейсу екрана додавання паролю

1. Екран "Видалення паролю":
   1. На екрані присутні:  
      - Поле для введення логіна/ресурсу для видалення.  
      - Кнопка "Видалити".  
      - Кнопка "Назад до Редагування".  
      - Область для повідомлень.
2. Екран "Список Паролів":
   1. Область для динамічного відображення списку розшифрованих пар логін/пароль.
   2. Кнопка "Назад до Головного Меню".

Загальні елементи стилізації:

* Використання фонового зображення для всієї сторінки.
* Відображення логотипу над основним робочим вікном програми.
* Центральне розміщення основного вікна та його елементів.
* Кнопки та поля вводу стилізовані для чіткості та зручності.

**4. Проектування моделей даних для збереження інформації**

В системі передбачено наступні основні сутності даних для зберігання інформації в базі даних MySQL:

* Запис Пароля (Password Entry)
* Маркер Валідації (Validation Marker)

A black background with a black square

AI-generated content may be incorrect.

Рисунок 4 - Діаграма моделей даних для збереження інформації

Модель даних "Запис Пароля" зберігає інформацію про кожен окремий логін та пов'язаний з ним пароль у зашифрованому вигляді.

Атрибути:

* id (первинний ключ, автоінкремент)
* encrypted\_login (зашифрований логін/ресурс)
* encrypted\_password (зашифрований пароль)

Модель даних "Маркер Валідації" зберігає спеціальний зашифрований рядок, який використовується для перевірки коректності майстер-ключа без необхідності дешифрування всіх записів паролів.

Атрибути:

* id (первинний ключ, автоінкремент)
* marker\_key (унікальний текстовий ідентифікатор маркера)
* encrypted\_value (зашифроване значення маркера)

**5. Створення діаграм компонентів, взаємодії та класів веб-додатку.**

Взаємодія клієнтської частини "Генератора Паролів" із сервером відбувається через HTTP API для виконання операцій зі сховищем паролів.

1. Валідація майстер-ключа / Розблокування сховища:
   1. Після введення користувачем майстер-ключа та натискання кнопки "Розблокувати / Створити сховище", клієнт виконує POST-запит на сервер за шляхом /api/validate-key.
   2. У тілі запиту передається об'єкт { masterKey: "введений\_ключ" }.
   3. Обробка відповіді:
      1. Статус 200 OK: Майстер-ключ валідний (або нове сховище створено). Клієнт зберігає майстер-ключ локально для поточної сесії, відображає повідомлення про успіх та переходить до Головного Меню.
      2. Статус 401 Unauthorized: Неправильний майстер-ключ. Клієнт відображає повідомлення про помилку, отримане з тіла відповіді сервера (поле message).
      3. Статус 400 Bad Request: Не надано майстер-ключ. Клієнт відображає відповідне повідомлення.
      4. Статус 5xx Server Error: Клієнт відображає стандартне повідомлення "Серверна помилка, спробуйте ще раз".
2. Отримання списку збережених паролів:
   1. Після вибору пункту "Переглянути паролі" в Головному Меню, клієнт виконує GET-запит на сервер за шляхом /api/passwords.
   2. Обробка відповіді:
      1. Статус 200 OK: Сервер повертає масив об'єктів { id, encrypted\_login, encrypted\_password }. Клієнт дешифрує encrypted\_login та encrypted\_password за допомогою збереженого майстер-ключа та відображає список користувачеві.
      2. Статус 5xx Server Error: Клієнт відображає повідомлення "Помилка завантаження паролів".
3. Додавання нового запису пароля:
   1. Після заповнення користувачем даних для нового пароля (логін, пароль) та підтвердження, клієнт спочатку шифрує логін та пароль локально за допомогою майстер-ключа.
   2. Потім виконується POST-запит на сервер за шляхом /api/passwords.
   3. У тілі запиту передається об'єкт { encryptedLogin: "зашифрований\_логін", encryptedPassword: "зашифрований\_пароль" }.
   4. Обробка відповіді:
      1. Статус 201 Created: Пароль успішно додано. Клієнт відображає повідомлення про успіх.
      2. Статус 400 Bad Request: Некоректні вхідні дані. Клієнт відображає повідомлення про помилку з тіла відповіді.
      3. Статус 409 Conflict: Запис з таким (зашифрованим) логіном вже існує. Клієнт відображає відповідне повідомлення.
      4. Статус 5xx Server Error: Клієнт відображає стандартне повідомлення про помилку.
4. Видалення запису пароля:
   1. Після введення користувачем логіна для видалення та підтвердження, клієнт спочатку шифрує цей логін локально за допомогою майстер-ключа.
   2. Потім виконується DELETE-запит на сервер за шляхом /api/passwords.
   3. У тілі запиту передається об'єкт { encryptedLogin: "зашифрований\_логін\_для\_видалення" }.
   4. Обробка відповіді:
      1. Статус 200 OK (або 204 No Content): Пароль успішно видалено. Клієнт відображає повідомлення про успіх.
      2. Статус 400 Bad Request: Некоректні вхідні дані. Клієнт відображає повідомлення про помилку.
      3. Статус 404 Not Found: Пароль з таким (зашифрованим) логіном не знайдено. Клієнт відображає відповідне повідомлення.
      4. Статус 5xx Server Error: Клієнт відображає стандартне повідомлення про помилку.

**6. Розробка бази даних для збереження та отримання інформації**

1. Обґрунтування вибору СКБД: Було обрано реляційну систему керування базами даних (СКБД) MySQL. Вибір обґрунтований її надійністю, широкою розповсюдженістю, доброю підтримкою спільнотою та достатньою продуктивністю для потреб "Генератора Паролів". Структура даних програми є простою та чітко визначеною, що добре узгоджується з реляційною моделлю.
2. Схема бази даних складається з двох таблиць: passwords та validation\_marker.

Таблиця passwords:

* id (INT, PK, AI)
* encrypted\_login (TEXT, NN, UQ)
* encrypted\_password (TEXT, NN)

Таблиця validation\_marker:

* id (INT, PK, AI)
* marker\_key (VARCHAR(50), UQ, NN)
* encrypted\_value (TEXT, NN)

(Де PK - Primary Key, AI - Auto Increment, NN - Not Null, UQ - Unique)

A black and white image of a person

AI-generated content may be incorrect.

Рисунок 6 - Схема бази даних

(На рисунку зображено дві таблиці без прямих зв'язків зовнішніми ключами між ними).

1. Створені індекси:
   1. Таблиця passwords, індекс за полем encrypted\_login (тип UNIQUE INDEX). Використовується для забезпечення унікальності зашифрованих логінів та для прискорення операцій пошуку записів (наприклад, при видаленні або перевірці на дублікати перед вставкою).
   2. Таблиця validation\_marker, індекс за полем marker\_key (тип UNIQUE INDEX). Використовується для швидкого доступу до єдиного запису маркера валідації та забезпечення його унікальності.

**7. Інтеграція сторонніх сервісів в систему**

У програмі "Генератор Паролів" не передбачено інтеграції зі сторонніми сервісам (система не розгортається, шифрування оригінальне, а база даних локальна), через що, як вказано у інструкції, ця частина не імплементовувалася.

**8. Створення серверної архітектури додатку**

Для розробки серверної частини програми було обрано монолітну архітектуру. Такий підхід був обраний через відносно невеликий та чітко визначений набір функціональних вимог, що дозволяє спростити розробку серверного додатку. Всі серверні компоненти працюють в межах одного процесу Node.js.

Серверний додаток реалізований з використанням фреймворку Express.js і відповідає за наступні основні функції:

1. Обробка HTTP-запитів від клієнтської частини.
2. Валідація майстер-ключа через взаємодію з маркером у базі даних.
3. Збереження, отримання та видалення зашифрованих записів паролів з бази даних MySQL.

Основні модулі/компоненти серверної частини:

1. Головний файл сервера (server.js): Ініціалізація Express-додатку, налаштування проміжного програмного забезпечення (middleware) для обробки запитів (наприклад, cors, express.json), визначення маршрутів API, запуск HTTP-сервера.
2. Модуль маршрутизації API (вбудований в server.js): Визначення ендпоінтів (/api/validate-key, /api/passwords) та прив'язка їх до відповідних функцій-обробників.
3. Функції-обробники API запитів (в server.js): Обробка логіки для кожного ендпоінта. Включає отримання даних із запиту, взаємодію з модулем доступу до БД та модулем криптографії (для маркера), формування та відправлення HTTP-відповіді клієнту.
4. Модуль доступу до бази даних (server/db.js): Налаштування з'єднання з MySQL (створення пулу з'єднань), надання функцій для виконання SQL-запитів (INSERT, SELECT, DELETE), ініціалізація схеми бази даних (створення таблиць, якщо вони не існують).
5. Модуль криптографії (server/cryptoService.js): На сервері використовується для шифрування/дешифрування маркера валідації майстер-ключа. Логіка ідентична клієнтському крипто-сервісу.

Схема основних компонентів серверної частини:

A group of white rectangular labels

AI-generated content may be incorrect.

Рисунок 8 - Діаграма компонентів серверної частини

**9. Реалізація обробки запитів та взаємодії з базою даних на сервері**

Серверна частина "Генератора Паролів" надає API для взаємодії з клієнтським додатком.

1. POST /api/validate-key
   1. Даний запит відповідає за перевірку наданого майстер-ключа або за створення нового сховища, якщо воно ще не існує.
   2. Тіло запиту: { "masterKey": "рядок\_ключа" } (JSON).
   3. Дії з базою даних:
      1. Виконується запит SELECT encrypted\_value FROM validation\_marker WHERE marker\_key = 'master\_validation\_marker' для отримання існуючого зашифрованого маркера.
      2. Якщо маркер знайдено, його значення дешифрується на сервері за допомогою наданого masterKey та порівнюється з очікуваним значенням.
      3. Якщо маркер не знайдено, на сервері генерується новий зашифрований маркер (текст "PasswordGeneratorValidationMarkerOK", зашифрований наданим masterKey) і записується в базу даних: INSERT INTO validation\_marker (marker\_key, encrypted\_value) VALUES ('master\_validation\_marker', 'новий\_зашифрований\_маркер').
   4. Використання індексів: Для отримання маркера використовується унікальний індекс за полем marker\_key таблиці validation\_marker.
   5. Відповідь:
      1. Успіх (статус 200): { "success": true, "message": "Ключ валідний" / "Нове сховище створено..." } (JSON).
      2. Помилка (статус 401/400/5xx): { "success": false, "message": "Текст помилки" } (JSON).
2. GET /api/passwords
   1. Даний запит відповідає за отримання всіх збережених зашифрованих записів паролів.
   2. Дії з базою даних: Виконується запит SELECT id, encrypted\_login, encrypted\_password FROM passwords.
   3. Використання індексів: Запит може використовувати первинний ключ id для впорядкування, але основне сканування таблиці не оптимізується специфічними індексами для цього запиту, оскільки повертаються всі записи.
   4. Відповідь:
      1. Успіх (статус 200): { "success": true, "data": [{ "id": ..., "encrypted\_login": "...", "encrypted\_password": "..." }, ...] } (JSON).
      2. Помилка (статус 5xx): { "success": false, "message": "Текст помилки" } (JSON).
3. POST /api/passwords
   1. Даний запит використовується для створення нового запису зашифрованого логіна та пароля.
   2. Тіло запиту: { "encryptedLogin": "зашифр\_логін", "encryptedPassword": "зашифр\_пароль" } (JSON).
   3. Дії з базою даних: Виконується запит INSERT INTO passwords (encrypted\_login, encrypted\_password) VALUES ($1, $2), де $1 - encryptedLogin, $2 - encryptedPassword. Перед вставкою може виконуватися перевірка на унікальність encrypted\_login (або це забезпечується обмеженням БД).
   4. Використання індексів: При вставці та перевірці на унікальність використовується унікальний індекс за полем encrypted\_login таблиці passwords.
   5. Відповідь:
      1. Успіх (статус 201): { "success": true, "message": "Пароль успішно додано" } (JSON).
      2. Помилка (статус 400/409/5xx): { "success": false, "message": "Текст помилки" } (JSON).
4. DELETE /api/passwords
   1. Даний запит використовується для видалення запису пароля за його зашифрованим логіном.
   2. Тіло запиту: { "encryptedLogin": "зашифр\_логін\_для\_видалення" } (JSON).
   3. Дії з базою даних: Виконується запит DELETE FROM passwords WHERE encrypted\_login = $1, де $1 - encryptedLogin.
   4. Використання індексів: Для пошуку запису для видалення використовується унікальний індекс за полем encrypted\_login таблиці passwords.
   5. Відповідь:
      1. Успіх (статус 200/204): { "success": true, "message": "Пароль успішно видалено" } (JSON).
      2. Помилка (статус 400/404/5xx): { "success": false, "message": "Текст помилки" } (JSON).

**Розділ 10. Тестування та налаштування веб-додатку на сервері**

1. Тестування серверної частини:

Серверний код, зокрема модуль криптографії (ServerCryptoService), був частково покритий unit-тестами для перевірки коректності його основних функцій. Тестування проводилося з використанням фреймворку Jest.

Основні сценарії, покриті unit-тестами:

* Коректність шифрування та дешифрування: Перевірка того, що текст, зашифрований та потім розшифрований тим самим ключем, повертається до свого оригінального вигляду.
* Стабільність шифрування: Перевірка, що повторне шифрування одного й того ж тексту тим самим ключем завжди дає однаковий зашифрований результат.
* Реакція на невірний ключ дешифрування: Перевірка, що при спробі дешифрування тексту ключем, відмінним від ключа шифрування, оригінальний текст не відновлюється.
* Обробка граничних випадків: Перевірка коректності шифрування та дешифрування порожнього рядка.

Результати тестування:

Всі написані unit-тести для модуля ServerCryptoService пройшли успішно. Це вказує на те, що базові операції шифрування та дешифрування в межах реалізованого алгоритму працюють відповідно до очікувань для перевірених сценаріїв.

1. Розгортання серверної частини:
   1. Для розгортання серверної частини (Node.js додаток) пропонується використовувати віртуальну машину (наприклад, Amazon EC2, DigitalOcean Droplet) на базі операційної системи Linux (наприклад, Ubuntu).
   2. Серверний додаток буде завантажуватися на сервер шляхом клонування з Git-репозиторію або у вигляді zip-архіву.
   3. Для управління процесом Node.js буде використовуватися менеджер процесів, такий як PM2, для забезпечення автоматичного перезапуску у разі збоїв та моніторингу.
   4. База даних MySQL буде розгорнута на тій же віртуальній машині або як окремий керований сервіс бази даних (наприклад, Amazon RDS, Google Cloud SQL).
2. Розгортання клієнтської частини:
   1. Клієнтська частина "Генератора Паролів" (HTML, CSS, JavaScript файли) є набором статичних файлів.
   2. Ці файли сервіруються безпосередньо серверним додатком Node.js/Express за допомогою middleware express.static('public'). Таким чином, окремого розгортання для клієнтської частини не потрібно, вона доставляється разом із серверним додатком.
3. Сценарії тестування (тест-кейси):
   1. Назва групи: Доступ до сховища

Назва сценарію: Успішне розблокування існуючого сховища  
Необхідні дії: Користувач вводить правильний майстер-ключ у відповідне поле та натискає кнопку "Розблокувати / Створити сховище".  
Очікуваний результат: Відображається повідомлення "Ключ валідний", користувач перенаправляється на Головне Меню.

* 1. Назва групи: Доступ до сховища

Назва сценарію: Спроба розблокування з невірним майстер-ключем  
Необхідні дії: Користувач вводить неправильний майстер-ключ та натискає кнопку "Розблокувати / Створити сховище".

Очікуваний результат: Відображається повідомлення про помилку "Неправильний майстер-ключ", користувач залишається на екрані введення ключа.

* 1. Назва групи: Доступ до сховища

Назва сценарію: Створення нового сховища

Необхідні дії: Користувач (при першому запуску або після очищення БД) вводить новий майстер-ключ та натискає кнопку "Розблокувати / Створити сховище".

Очікуваний результат: Відображається повідомлення "Нове сховище створено, ключ валідний", користувач перенаправляється на Головне Меню.

* 1. Назва групи: Управління паролями

Назва сценарію: Успішне додавання власного пароля

Необхідні дії: Після розблокування сховища, користувач переходить до "Додати/видалити пароль" -> "Додати пароль" -> "Ввести свій пароль", заповнює поля "Логін/Ресурс" та "Пароль", натискає "Зберегти".  
Очікуваний результат: Відображається повідомлення "Пароль успішно додано". При наступному перегляді паролів новий запис присутній у списку.

* 1. Назва групи: Управління паролями

Назва сценарію: Успішна генерація та збереження пароля

Необхідні дії: Користувач переходить до генерації пароля, задає параметри, генерує пароль, вводить "Логін/Ресурс" для згенерованого пароля та натискає "Зберегти згенерований пароль".

Очікуваний результат: Відображається повідомлення "Згенерований пароль успішно додано". При наступному перегляді паролів новий запис присутній у списку.

* 1. Назва групи: Управління паролями

Назва сценарію: Успішне видалення пароля

Необхідні дії: Користувач переходить до "Додати/видалити пароль" -> "Видалити пароль", вводить існуючий "Логін/Ресурс" та натискає "Видалити".  
Очікуваний результат: Відображається повідомлення "Пароль успішно видалено". При наступному перегляді паролів цей запис відсутній.

* 1. Назва групи: Управління паролями

Назва сценарію: Спроба додати пароль з існуючим логіном  
Необхідні дії: Користувач намагається додати новий пароль, використовуючи "Логін/Ресурс", який вже існує в сховищі.  
Очікуваний результат: Відображається повідомлення про помилку, що запис з таким логіном вже існує.

**ВИСНОВКИ**

Протягом виконання даної курсової роботи, метою якої було зазначено створення програми для генерації надійних паролів, реалізовано веб-додаток, що здатний у зрозумілому, простому, та одночас безпечному форматі виконувати функції як генерації, так і зберігання важливих даних.

Ключовим аспектом програми є механізм шифрування даних: логіни та паролі шифруються безпосередньо у браузері користувача за допомогою унікального майстер-ключа перед їх відправкою на сервер. Це забезпечує, що майстер-ключ та не зашифровані дані ніколи не покидають клієнтську сторону, підвищуючи рівень конфіденційності. Для зберігання зашифрованої інформації та маркера валідації майстер-ключа було обрано систему керування базами даних MySQL, яка забезпечує надійне збереження даних.

В результаті розробки було створено веб-додаток, що надає користувачам можливість генерувати паролі за заданими параметрами (довжина, наявність різних типів символів), зберігати власні або згенеровані паролі разом з логінами у зашифрованому вигляді, переглядати та видаляти збережені записи. Система валідації майстер-ключа та структурований інтерфейс спрямовані на забезпечення функціональності та керованості процесу управління паролями.

**ВИКОРИСТАНІ ДЖЕРЕЛА**

1. Node.js Tutorial. URL: <https://www.w3schools.com/nodejs/> (date of access: 25.05.2025).
2. MySQL: Understanding What It Is and How It’s Used.  URL: <https://www.oracle.com/mysql/what-is-mysql/> (date of access:  25.05.2025).
3. Для чого потрібен менеджер пакетів npm. URL: <https://foxminded.ua/npm-tse/> (дата звернення: 25.05.2025).
4. Модульне тестування JavaScript з Jest.  URL: <https://programmingmentor.com.ua/jest-intro/> (дата звернення:  25.05.2025).
5. Коротко про АРІ та його тестування.  URL: <https://qagroup.com.ua/publications/korotko-pro-ari-ta-jogo-testuvannia/> (дата звернення: 25.05.2025).
6. Microservices vs. monolithic architecture. URL: <https://www.atlassian.com/microservices/microservices-architecture/microservices-vs-monolith> (date of access: 25.05.2025).
7. Hash function - MDN Web Docs Glossary - Mozilla.  URL: <https://developer.mozilla.org/en-US/docs/Glossary/Hash_function> (date of access:  25.05.2025).