

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)
на тему:

«Локальний менеджер паролів»

Виконав студент групи ІПЗк-20-1
БИЛКОВ Владислав Сергійович

Керівник роботи:
КОНЦЕДАЙЛО Валерій Валерійович

Консультант:
ЧИЖМОРТЯ Олексій Володимирович

Рецензент:
КРАВЧЕНКО Світлана Миколаївна

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

В.о. зав. кафедри інженерії
програмного забезпечення

Андрій МОРОЗОВ

«14» лютого 2023 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: Билков Владислав Сергійович

Керівник роботи: Чижмотря Олексій Володимирович

Тема роботи: **«Локальний менеджер паролів»**,

затверджена Наказом закладу вищої освіти від **«14» лютого 2023 р., №70с**

Вихідні дані для роботи: Методи та засоби розробки десктопної системи для управління паролями. Моделі, технології, структури, засоби та методи розробки та взаємодії десктопних систем управління паролями.

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Чижмотря О.В.	16.02.2023р.	16.02.2023р.
2	Чижмотря О.В.	16.02.2023р.	16.02.2023р.
3	Чижмотря О.В.	16.02.2023р.	16.02.2023р.

Календарний план

№ з/п	Назва етапів випускної кваліфікаційної роботи	Термін виконання роботи	Примітка
1	Аналіз вимог до програмного продукту	16 лютого 2023 – 25 лютого 2023	Виконано
2	Пошук, огляд та аналіз аналогічних продуктів	26 лютого 2023 – 7 березня 2023	Виконано
3	Вивчення матеріалу по конкретній тематиці	8 березня 2023 – 21 березня 2023	Виконано
4	Планування структури програми	22 березня 2023 – 1 квітня 2023	Виконано
5	Проектування та реалізація архітектури локального сховища паролів	2 квітня 2023 – 12 квітня 2023	Виконано
6	Написання програмного коду	13 квітня 2023 – 25 травня 2023	Виконано
7	Тестування роботи програми	26 травня 2023 – 28 травня 2023	Виконано
8	Аналіз одержаних результатів	28 травня 2023 – 1 червня 2023	Виконано
9	Оформлення пояснювальної записки	1 червня 2023 – 15 червня 2023	Виконано
10	Попередній захист	18 травня 2023	Виконано

Реферат

Випускна робота бакалавра складається з програмного комплексу десктопної програми для локального сховища паролів, а також пояснювальної записки. Пояснювальна записка містить текстову частину, яка викладена на 60 сторінках друкованого тексту, а також має 22 сторінки додатків. Список використаних джерел містить 15 найменувань і займає 2 сторінки. В роботі наведено 31 рисунок. Загальний обсяг роботи становить 94 сторінки.

Метою роботи є розробка десктопної програми для локального сховища паролів з використанням мови програмування C та графічного інтерфейсу GTK4. Програма забезпечує зручне та безпечне зберігання паролів на локальному комп'ютері користувача. Шифрування паролів виконується за допомогою алгоритму AES (Advanced Encryption Standard), а самі паролі зберігаються у бінарних файлах.

У роботі було визначено вимоги користувача та обґрунтовано створення концептуальної моделі. Було проаналізовано аналоги програмних продуктів та обрано використання мови програмування C та графічного інтерфейсу GTK4. Проведено розробку структури сховища у вигляді бінарних файлів, а також розроблено алгоритм роботи програми, обрано архітектуру та розроблено об'єктно-орієнтовану структуру системи. Описано програмну реалізацію десктопної програми для локального сховища паролів.

КЛЮЧОВІ СЛОВА: ЛОКАЛЬНЕ СХОВИЩЕ, ПАРОЛІ, ДЕСКТОПНА ПРОГРАМА, ШИФРУВАННЯ, AES, БІНАРНІ ФАЙЛИ.

					ІПЗ.КР.Б – 121 – 23 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата	Локальний менеджер паролів Пояснювальна записка	Літ.	Арк.	Аркушів
Розроб.		В.С. Билков					4	94
Керівник		В.В. Концедайло				Житомирська політехніка, група ІПЗК-20-1		
Консульт		О.В. Чижмотря						
Рецензент		С.М. Кравченко						
Зав. каф.		А.В. Морозов						

ABSTRACT

The Bachelor's thesis consists of a software complex of a desktop application for a local password manager, as well as an explanatory note. The explanatory note includes a textual part, which is presented in 60 pages of printed text, and also contains 22 pages of appendices. The list of references includes 15 sources and occupies 2 pages. The thesis includes 31 figures. The total volume of the thesis is 94 pages.

The aim of the thesis is to develop a desktop application for local password storage using the C programming language and the GTK4 graphical interface. The program ensures convenient and secure storage of passwords on the user's local computer. Password encryption is performed using the AES (Advanced Encryption Standard) algorithm, and the passwords themselves are stored in binary files.

The requirements of the user were determined and the creation of a conceptual model was justified in the thesis. Analogues of software products were analyzed, and the use of the C programming language and the GTK4 graphical interface was chosen. The development of the storage structure in the form of binary files was conducted, as well as the design of the program's algorithm, selection of the architecture, and development of an object-oriented system structure. The software implementation of the desktop application for local password storage is described.

KEYWORDS: LOCAL STORAGE, PASSWORDS, DESKTOP APPLICATION, ENCRYPTION, AES, BINARY FILES.

		<i>В.С. Билков</i>			<i>ІІЗ.КР.Б – 121 – 23 – ІІЗ</i>	<i>Арк.</i>
						<i>5</i>
<i>Змн.</i>	<i>Арк.</i>	<i>№ докум.</i>	<i>Підпис</i>	<i>Дата</i>		

Зміст

Вступ.....	9
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ЛОКАЛЬНОГО СХОВИЩА ПАРОЛІВ..	11
1.1. Опис проекту та постановка задачі.....	11
1.2. Мета і завдання проекту.....	12
1.3. Огляд існуючих програмних продуктів.....	14
1.4. Обґрунтування та вибір технологій для розробки	20
Висновки до першого розділу.....	24
Розділ 2. ПРОЕКТУВАННЯ ЛОКАЛЬНОГО МЕНЕДЖЕРА ПАРОЛІВ	26
2.1. Визначення варіантів використання та структури системи	26
2.2. Проектування та реалізація алгоритмів роботи системи.....	33
2.3. Реалізація додатку локальний менеджер паролів	42
Висновки до другого розділу	51
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ЛОКАЛЬНИМ МЕНЕДЖЕРОМ ПАРОЛІВ	53
3.1. Порядок встановлення та налаштування параметрів системи.	53
3.2. Структура інтерфейсу та порядок роботи з системою.....	55
3.3. Тестування системи	62
Висновки до третього розділу.....	68
ВИСНОВКИ.....	69
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	70
ДОДАТКИ.....	72

Перелік умовних скорочень

- iOS: iOS є мобільною операційною системою, розробленою компанією Apple для їх пристроїв, таких як iPhone, iPad та iPod Touch. Вона включає широкий спектр функцій та можливостей для розробки додатків та роботи з пристроями Apple.

- Gtk4: Gtk4 (GTK+ 4) - це бібліотека графічного інтерфейсу користувача, що використовується для розробки додатків з графічним інтерфейсом у середовищі Linux та інших платформах. Вона надає набір інструментів та компонентів для створення модерних інтерактивних додатків з використанням GTK API.

- C: C - це загальноприйнята мова програмування, яка широко використовується для розробки системного програмного забезпечення, драйверів, вбудованого програмування та багатьох інших застосувань. Вона відома своєю ефективністю, портативністю і можливістю прямого доступу до рівня апаратного забезпечення.

- AES: AES (Advanced Encryption Standard) - це симетричний алгоритм шифрування, який використовується для захисту конфіденційної інформації. Він є одним з найпоширеніших алгоритмів шифрування і використовується у багатьох застосуваннях, включаючи захист даних в мережах, зберігання файлів та шифрування комунікації.

- Біт: Біт (binary digit) - це найменша одиниця інформації в комп'ютерній науці та цифровій технології. Він може приймати два можливих значення: 0 або 1. Біти використовуються для збереження та обробки інформації в цифрових пристроях.

- Bin: Bin (binary) - скорочення від "binary" або "binary file" - означає двійковий файл, який містить машинний код або даний у двійковому форматі. Це формат файлу, який використовується для збереження виконуваних програм або даних у вигляді двійкових чисел.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						7
Змн.	Арк.	№ докум.	Підпис	Дата		

- Src: Src (source) - скорочення від "source" - означає вихідний код програми або файл, що містить текстовий код програми, який можна змінювати або компілювати в виконуваний файл. Вихідний код зазвичай написаний програмістами у відповідних мовах програмування.

- OpenSSL: OpenSSL - це бібліотека та набір інструментів для реалізації протоколів шифрування та безпеки в програмах. Вона надає функціональність для шифрування, дешифрування, генерації цифрових підписів, створення сертифікатів безпеки та інші операції забезпечення безпеки даних.

- Pacman: Pacman - це пакетний менеджер для дистрибутиву Linux, використовуваний в Arch Linux та інших деривативах Arch Linux. Він дозволяє користувачам легко встановлювати, оновлювати та керувати пакетами програмного забезпечення з офіційних репозиторіїв.

- AUR: AUR (Arch User Repository) - це спільнотний репозиторій пакетів для дистрибутиву Arch Linux. Він містить пакети, які не включені до офіційних репозиторіїв Arch Linux, і їх можна встановити за допомогою спеціального клієнта AUR, наприклад, yaourt або yaу.

- Git: Git - це розподілена система керування версіями, яка використовується для відстеження змін у файловій системі під час розробки програмного забезпечення. Вона дозволяє розробникам працювати з однією й тією ж версією проекту, спільно працювати над змінами, керувати гілками розробки та злиттями.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						8
Змн.	Арк.	№ докум.	Підпис	Дата		

Вступ

У наші дні зростає кількість інтернет-сервісів та онлайн-акаунтів, що вимагають реєстрації та авторизації за допомогою унікальних паролів. Забезпечення безпеки інформації у такому контексті є важливим завданням, яке вимагає від користувачів навичок створення й зберігання надійних паролів. Одним з ефективних рішень для вирішення цієї проблеми є використання локального зберігача паролів.

Основна мета цієї кваліфікаційної роботи полягає у розробці та реалізації локального зберігача паролів, який забезпечуватиме безпеку та зручність зберігання та керування паролями користувачів. Для досягнення цієї мети планується визначити вимоги до програмного продукту, проаналізувати існуючі рішення, розробити архітектуру програмного продукту, реалізувати його програмний код, провести тестування та підготувати документацію для використання продукту.

Завдання даної кваліфікаційної роботи включають наступні етапи:

1. Визначення вимог до локального зберігача паролів з урахуванням потреб користувачів і стандартів безпеки даних.
2. Аналіз існуючих рішень для зберігання паролів з метою оцінки їх переваг і недоліків.
3. Проектування та розробка архітектури програмного продукту, включаючи вибір необхідних технологій та інструментів розробки.
4. Розробка та тестування програмного коду, враховуючи встановлені вимоги та архітектурні рішення.
5. Документування та підготовка програмного продукту до використання, забезпечення зрозумілості та легкості використання для користувачів.
6. Оцінка ефективності розробленого програмного продукту з точки зору безпеки, функціональності та продуктивності.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						9
Змн.	Арк.	№ докум.	Підпис	Дата		

Предметом даної кваліфікаційної роботи є розробка локального зберігача паролів, зосереджена на технічних аспектах розробки програмного продукту та його використанні для зберігання паролів.

Об'єктом дослідження є процес проектування та розробки програмного продукту, який включає в себе розробку архітектури, вибір технологій та інструментів розробки, програмування, тестування та впровадження програмного продукту.

Отже, у наші дні все більше інтернет-сервісів та онлайн-акаунтів, що потребують надійних паролів. У зв'язку з цим важливим стає розробка локального зберігача паролів для забезпечення безпеки та зручності користувачів. Ця кваліфікаційна робота має на меті визначити вимоги, проаналізувати існуючі рішення, розробити архітектуру, реалізувати програмний код та підготувати документацію. Завдання включають визначення вимог, аналіз рішень, проектування, розробку, тестування та оцінку ефективності програмного продукту. Об'єктом дослідження є процес проектування та розробки програмного продукту для зберігання паролів.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		10

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ЛОКАЛЬНОГО СХОВИЩА ПАРОЛІВ

1.1. Опис проекту та постановка задачі

Даний проект є програмною розробкою, метою якої є створення локального сховища паролів для безпечного та зручного зберігання та керування ними. Програмний продукт має дружній та інтуїтивний інтерфейс, забезпечуючи захист від несанкціонованого доступу до конфіденційної інформації. Основна функція полягає в зберіганні паролів у захищеному файлі, створенні безпечних паролів та використанні сучасних методів шифрування для забезпечення безпеки даних. Загальна характеристика проекту - розробка зручного та надійного інструменту для зберігання та управління паролями.

Однією з основних функцій програмного продукту є зберігання паролів у захищеному від доступу файлі, що дозволить користувачам зручно зберігати свої паролі та зберігати їх у безпечному місці. Крім того, програма надає можливість створювати та зберігати безпечні паролі, що зменшує ризик їх перехоплення та зламу.

Оскільки програмний продукт зберігає важливу та конфіденційну інформацію користувачів, велика увага приділяється його безпеці та стійкості. Для цього будуть використовуватися сучасні методи шифрування, а також регулярні оновлення для забезпечення безпеки користувачів та захисту від нових загроз [13].

Загальна характеристика проекту полягає в тому, що це розробка програмного продукту, призначеного для зберігання та управління паролями користувачів. Цей продукт є локальним зберігачем паролів, що означає, що всі дані зберігаються на комп'ютері користувача та не потребує підключення до Інтернету. Продукт має дозволяти користувачам легко додавати, змінювати та видаляти свої паролі, а також керувати доступом до них. Програмний продукт

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						11
Змн.	Арк.	№ докум.	Підпис	Дата		

забезпечуватиме безпеку даних за допомогою різних методів шифрування та захисту від несанкціонованого доступу.

Загальна характеристика проекту передбачає створення надійного та зручного засобу для зберігання паролів, що забезпечить безпеку та зручність використання для користувачів.

Підсумовуючи: основна функція цього інструменту - зберігання паролів в файлі захищеному від несанкційованого доступу. Забезпечення безпеки даних досягається за допомогою сучасних методів шифрування. Крім того, інструмент має мати дружній інтерфейс користувача, що дозволяє легко і швидко використовувати його функціонал.

1.2. Мета і завдання проекту

Мета даної кваліфікаційної роботи полягає в розробці та впровадженні локального сховища паролів, яке забезпечить зручне та безпечне зберігання та управління паролями користувачів. Для досягнення цієї мети необхідно визначити вимоги до програмного продукту, проаналізувати існуючі рішення на ринку, спроектувати та реалізувати архітектуру програмного продукту, розробити та протестувати програмний код, документувати та підготувати програмний продукт до використання.

Окрім цього, мета дослідження полягає в забезпеченні високого рівня безпеки даних, що зберігаються у сховищі паролів, за допомогою сучасних методів шифрування. Розроблений програмний продукт повинен відповідати вимогам користувачів щодо зручності та надійності зберігання та управління паролями, а також бути готовим до подальшого розвитку та розширення.

Для досягнення цієї мети необхідно розробити та реалізувати програмний продукт, який задовольнятиме вимоги користувачів щодо зручності та надійності зберігання паролів, а також має високий рівень безпеки даних. Для досягнення цих цілей передбачено низку завдань, які необхідно виконати.

		<i>В.С. Билков</i>			<i>ІПЗ.КР.Б – 121 – 23 – ПЗ</i>	<i>Арк.</i>
						<i>12</i>
<i>Змн.</i>	<i>Арк.</i>	<i>№ докум.</i>	<i>Підпис</i>	<i>Дата</i>		

До них входять:

1. Визначення вимог до локального зберігача паролів. Для цього необхідно провести аналіз потреб користувачів та вимог до безпеки збереження паролів, визначити функціональні та нефункціональні вимоги до програмного продукту.
2. Аналіз існуючих рішень. Необхідно провести огляд існуючих рішень на ринку та визначити їх переваги та недоліки. На основі цього аналізу потрібно визначити найбільш ефективні підходи до розробки локального зберігача паролів.
3. Проектування та розробка архітектури програмного продукту. На основі вимог до програмного продукту необхідно розробити архітектуру програми, вибрати підходящі технології та засоби розробки. Для забезпечення високої безпеки даних необхідно визначити методи шифрування та автентифікації користувачів.
4. Розробка та тестування програмного коду. Після проектування архітектури програмного продукту необхідно розробити та протестувати програмний код. Під час тестування необхідно перевірити правильність роботи програмного продукту, а також його безпеку та стійкість до злому.
5. Документування та підготовка програмного продукту до використання. Після розробки та тестування програмного продукту необхідно підготувати документацію, яка описуватиме його функціональність та процеси взаємодії з користувачем. Також потрібно підготувати програму до використання та встановлення на різних пристроях.
6. Оцінка ефективності програмного продукту. Оцінка ефективності програмного продукту буде включати декілька кроків, таких як тестування, збір зворотного зв'язку від користувачів.

1.3. Огляд існуючих програмних продуктів

LastPass є одним з найбільш відомих і популярних зберігачів паролів. Це хмарний сервіс, що дозволяє зберігати паролі та інші конфіденційні дані на серверах компанії. Користувачі можуть отримувати доступ до своїх даних з будь-якого пристрою за допомогою спеціального розширення для веб-браузера або мобільної програми.

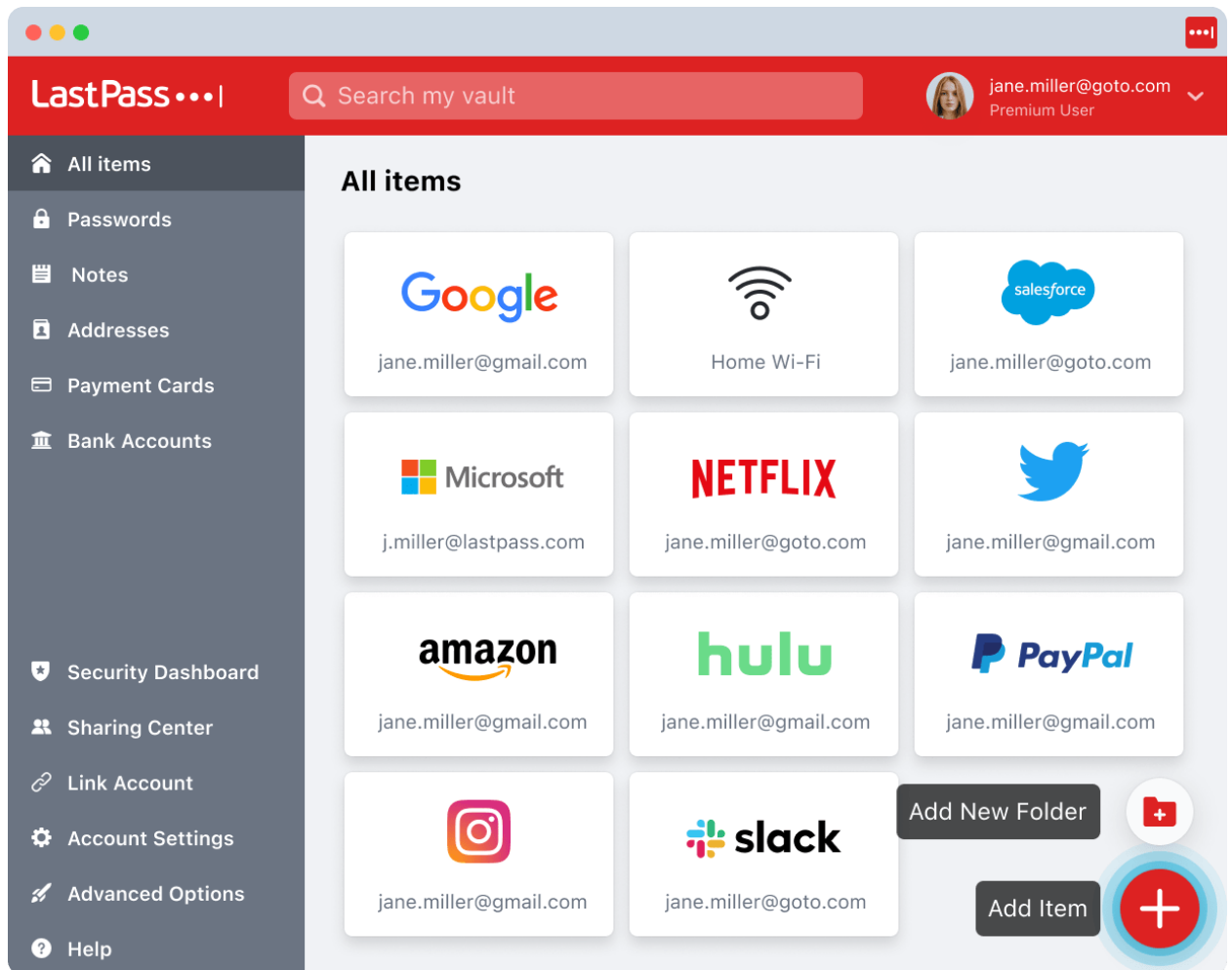


Рис. 1.1 - Головне меню LastPass

Переваги:

- Забезпечення надійної і безпечної збереження паролів і конфіденційної інформації;
- Швидкий і зручний доступ до паролів на різних пристроях;
- Автоматичне заповнення паролів на веб-сайтах;
- Генерація складних паролів;

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						14
Змн.	Арк.	№ докум.	Підпис	Дата		

- Можливість зберігати не тільки паролі, а й іншу конфіденційну інформацію, наприклад, номери кредитних карт;
- Можливість додавати додаткові шари захисту, наприклад, двох факторну автентифікацію.

Недоліки:

- Потребує інтернет-підключення для доступу до зберігача паролів;
- Оплата за розширені функції та можливості;
- Питання щодо безпеки при збереженні даних на серверах компанії;
- Не всі функції доступні в безкоштовній версії;
- Можливість злому акаунту зловмисником, якщо пароль до зберігача паролів стає відомим зловмиснику.

Отже LastPass пропонує безкоштовну та платну версії, які відрізняються за кількістю функцій та можливостей. Сервіс дозволяє автоматично заповнювати форми, генерувати складні паролі, синхронізувати дані між різними пристроями та додавати додаткову захист за допомогою двофакторної аутентифікації.

Однією з найбільших переваг LastPass є його відкритість для інтеграції з різними сервісами та програмами, що дозволяє значно спростити роботу з паролями. Проте, оскільки сервіс є хмарним, його безпека може становити певну загрозу, особливо в разі порушення безпеки на серверах компанії.

KeePass - це безкоштовний open-source зберігач паролів, який дозволяє користувачам зберігати свої паролі у безпечному місці та керувати ними. Програма доступна для встановлення на різних платформах, включаючи Windows, macOS, Linux та Android.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						15
Змн.	Арк.	№ докум.	Підпис	Дата		

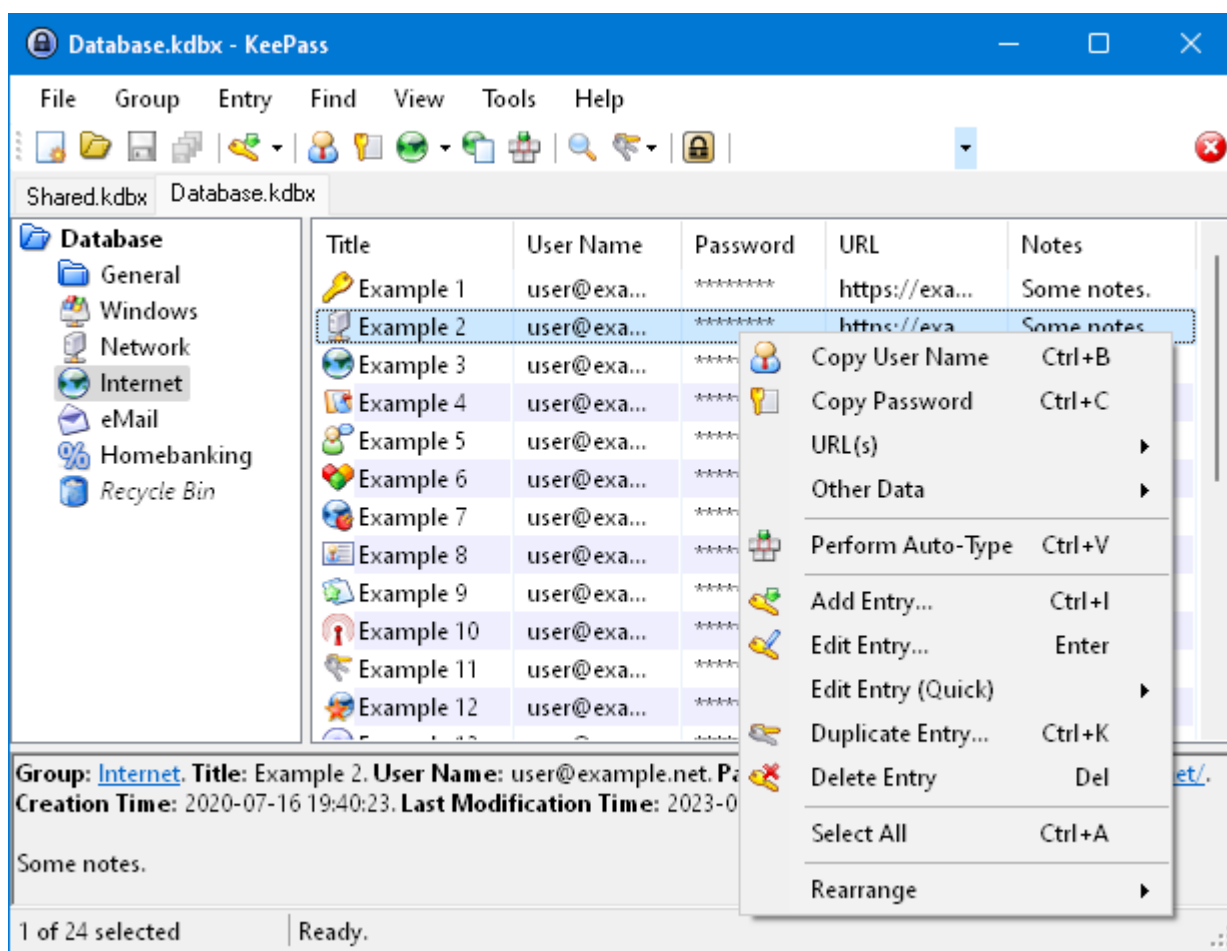


Рис. 1.2 - Головне меню KeePass

Основні переваги KeePass:

- Безкоштовність та відкритий код, що дозволяє користувачам перевіряти безпеку програми та редагувати її код;
- Високий рівень безпеки, який забезпечується за допомогою різних методів шифрування та захисту даних;
- Відсутність потреби у підключенні до інтернету, що дозволяє користувачам зберігати свої паролі офлайн;
- Можливість використання різних баз даних та перенесення їх між різними пристроями.

Однак, також є деякі недоліки:

- Недостатня зручність використання в порівнянні з деякими іншими зберігачами паролів, такими як LastPass;

- Обмеження функцій, таких як автоматичне заповнення паролів та моніторинг за порушенням безпеки даних.

Keerpass має багато переваг, включаючи відкритий вихідний код, можливість використання на різних платформах та можливість керування паролями з допомогою майстра паролів. Крім того, він пропонує високий рівень безпеки та конфіденційності, оскільки паролі зберігаються в зашифрованому файлі, який можна захистити паролем або ключовим файлом.

Однак, як і будь-який інший зберігач паролів, Кеерасс має свої недоліки. Його інтерфейс може бути складним для новачків та не дуже зручним у використанні. Крім того, для синхронізації даних між різними пристроями потрібно використовувати сторонні сервіси або інші засоби, що може бути складним для деяких користувачів.

Dashlane - це ще один зберігач паролів, який пропонує широкий спектр функцій для зберігання та управління паролями. Додаток працює на різних платформах, включаючи Windows, macOS, Android та iOS, і має розширення для браузерів, щоб зручно зберігати та заповнювати паролі на веб-сайтах.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						17
Змн.	Арк.	№ докум.	Підпис	Дата		

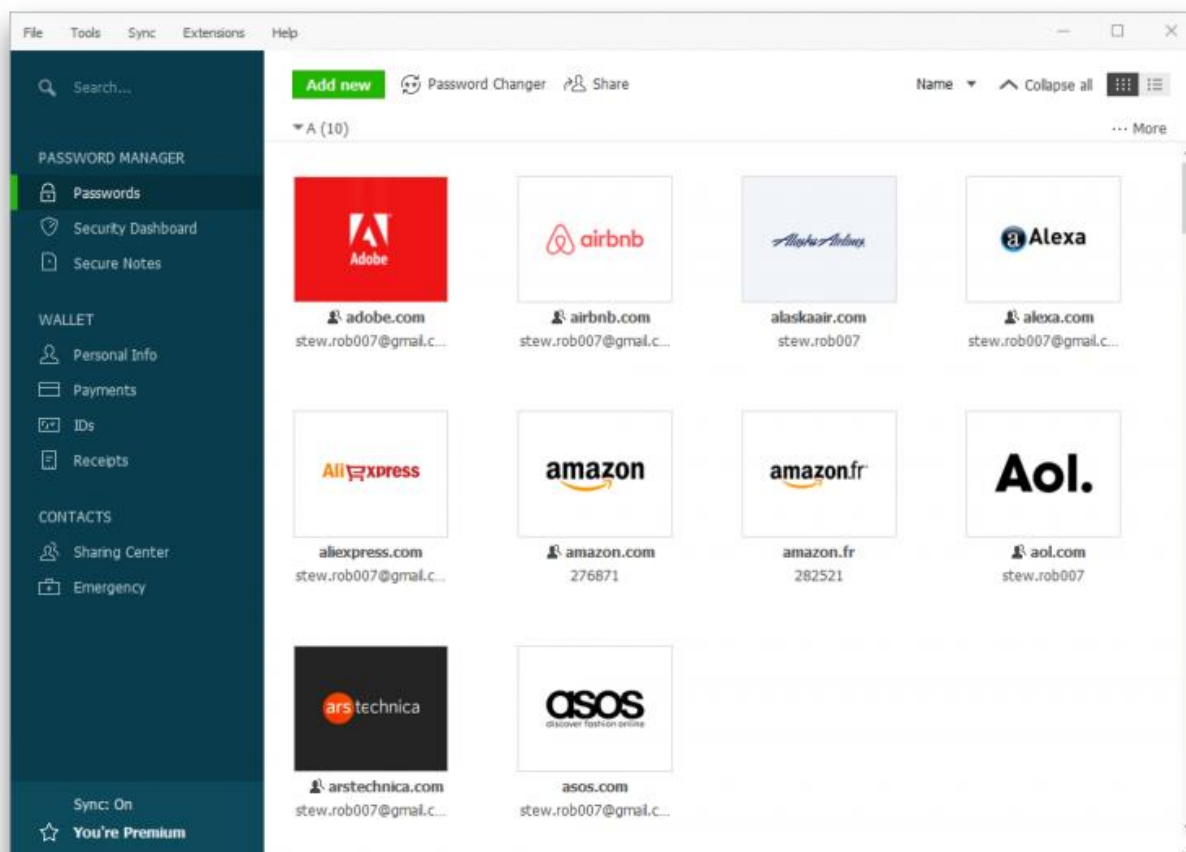


Рис. 1.3 - Головне меню Dashlane.

Переваги Dashlane:

- Легка інтеграція з браузерами та мобільними пристроями: Dashlane працює на більшості популярних браузерів, таких як Chrome, Firefox, Safari та Edge, а також має мобільні додатки для Android та iOS.
- Автоматична зміна паролів: Dashlane може автоматично генерувати нові, складні паролі та змінювати старі на всіх сайтах, на яких ви маєте обліковий запис.
- Автозаповнення форм: Dashlane може автоматично заповнювати форми для реєстрації та входу на сайти.
- Шифрування на стороні клієнта: Dashlane шифрує всі ваші дані на вашому пристрої, що забезпечує більш високий рівень безпеки.
- Синхронізація між пристроями: Dashlane синхронізує ваші дані між усіма пристроями, на яких ви встановили додаток.

Недоліки Dashlane:

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						18
Змн.	Арк.	№ докум.	Підпис	Дата		

- Обмежена безкоштовна версія: Для отримання повного функціоналу вам потрібно придбати платну версію.
- Висока ціна: Платна версія Dashlane коштує досить дорого порівняно з іншими аналогами.
- Залежність від інтернету: Для використання деяких функцій, таких як синхронізація між пристроями, потрібне підключення до Інтернету.
- Обмежена підтримка браузерів: Dashlane не підтримує всі браузери, що може бути проблемою для деяких користувачів.

Одна з основних переваг Dashlane полягає в його простоті використання та інтуїтивно зрозумілому інтерфейсі, що дозволяє легко знаходити та організовувати паролі. Також є функції автоматичного заповнення форм та створення складних паролів.

Dashlane також пропонує додаткові функції, такі як моніторинг порушень безпеки, який повідомляє користувача про потенційні порушення безпеки даних та поради щодо зміни пароля. Крім того, є можливість зберігання інших конфіденційних даних, таких як номери кредитних карток та адреси доставки.

Однак, як і більшість програм для зберігання паролів, Dashlane має певні недоліки, зокрема, обмежений безкоштовний план, що не дозволяє користувачам зберігати більше ніж 50 паролів. Крім того, деякі користувачі вказують на проблеми з автоматичним заповненням форм на деяких веб-сайтах, що може бути незручним.

Підсумовуючи LastPass, KeePass та Dashlane є популярними програмами для зберігання паролів. Їхня основна мета полягає у тому, щоб забезпечити безпечне та зручне зберігання паролів користувачів.

KeePass - безкоштовна програма з відкритим кодом, що забезпечує безпеку та конфіденційність даних. LastPass - платна програма з багатим функціоналом та можливістю синхронізації з різними пристроями, але може

мати проблеми з безпекою. Dashlane - ще одна платна програма з багатою функціональністю та моніторингом потоків даних, але може бути витратнішою. Вибір програми для зберігання паролів залежить від ваших особистих потреб та вимог до безпеки та функціоналу.

З огляду на розглянуті аналоги зберігачів паролів можна зробити висновок, що існуючі рішення, які пропонуються на ринку, мають свої переваги та недоліки, які варто враховувати при виборі зберігача паролів. Однак, недоліки таких рішень, як низький рівень безпеки та обмежені можливості для налаштування, можуть стати критичними для користувачів, які мають велику кількість облікових записів та повинні зберігати багато паролів.

Тому, написання кваліфікаційного проекту на тему сховище паролів може бути дуже корисним для тих, хто має потребу в надійному та безпечному зберіганні паролів. Такий проект може допомогти розробити зберігач паролів, який має більш високий рівень безпеки, можливості для налаштування та інтеграції з іншими програмними продуктами.

1.4. Обґрунтування та вибір технологій для розробки

Для даного проекту вимоги до інтерфейсу полягають у тому, щоб він був легким та простим у використанні для користувача. Оскільки цільова операційна система - Linux, необхідно вибрати технологію, яка буде легко програмуватися та зможе бути легко перенесена на інші операційні системи Linux без перепрограмування. Крім того, важливим аспектом є можливість збереження паролів та їх безпека через шифрування [12].

В рамках даного проекту, було вирішено використовувати технологію gtk 4 для написання графічного інтерфейсу. Одним з головних критеріїв при виборі технології була легкість програмування та можливість легкого переносу на інші операційні системи сімейства linux без перепрограмування під них.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						20
Змн.	Арк.	№ докум.	Підпис	Дата		

Gtk 4 - це оновлена версія відомої бібліотеки gtk для створення графічних інтерфейсів на різних платформах. Вона підтримує багато різних елементів інтерфейсу, які дозволяють легко створювати гарні інтерфейси з різноманітними ефектами та анімацією. Крім того, gtk 4 має хорошу документацію та підтримується великою спільнотою розробників [1].

Використання gtk 4 в даний час є одним з найбільш оптимальних виборів для створення графічного інтерфейсу на платформі linux. Ця технологія дозволяє створювати гарні інтерфейси з мінімальними зусиллями, що зменшує час розробки та збільшує ефективність програміста. Крім того, gtk 4 підтримується на багатьох різних операційних системах, що дозволяє з легкістю переносити програму на інші платформи без необхідності перепрограмування.

Таким чином, використання gtk 4 є логічним та оптимальним вибором для даного проекту, оскільки ця технологія дозволить створити графічний інтерфейс з привабливим дизайном та функціональністю, зберігаючи при цьому легкість програмування та можливість легкого переносу програми на інші операційні системи.

Використання бінарного файлу як бази даних має свої переваги та недоліки. Однією з переваг є швидкий доступ до даних, оскільки бінарні файли зазвичай займають менше місця та мають менші обсяги, ніж бази даних на основі реляційних систем управління базами даних. Крім того, бінарні файли можуть бути шифровані, що забезпечить додатковий рівень безпеки та конфіденційності.

Проте, бінарний файл не має деяких переваг, що мають інші формати баз даних. Наприклад, бази даних на основі реляційних систем можуть бути більш масштабованими та забезпечувати більші можливості для збереження даних у вигляді таблиць та взаємозв'язків між ними. Крім того, бінарний файл може бути менш зручним для редагування та пошуку даних порівняно з іншими форматами баз даних.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						21
Змн.	Арк.	№ докум.	Підпис	Дата		

В даному проєкті використання бінарного файлу як бази даних має свої переваги. Оскільки програма для збереження паролів не потребує масштабування та складних операцій з базами даних, бінарний файл є більш простим та ефективним рішенням. Крім того, шифрування бінарного файлу забезпечить додатковий рівень безпеки даних, що є дуже важливим у програмі для збереження паролів.

Вибір мови програмування залежить від багатьох факторів, таких як тип проєкту, його складність, час та бюджет. У даному проєкті ми обрали мову програмування C, і ось чому.

По-перше, C є компільованою мовою, що дозволяє досягти високої швидкодії та ефективності виконання програми. Це особливо важливо для програм, що працюють з конфіденційними даними, такими як паролі, де швидкодія та ефективність є критичними факторами [6].

По-друге, мова програмування C має багато ресурсів та документації, які роблять її дуже популярною для системних програм та програм з низьким рівнем доступу до апаратного забезпечення. Це дозволяє нам швидко знайти рішення на будь-які проблеми, що виникають в процесі розробки [7].

По-третє, C має багатий набір бібліотек, які дозволяють легко реалізувати різноманітні функції. Наприклад, для реалізації шифрування паролів ми можемо використовувати бібліотеку OpenSSL, яка має широкий набір криптографічних алгоритмів [2].

Іншими можливими варіантами мов програмування для даного проєкту можуть бути Python, Java або C++. Python має простий синтаксис та широку спільноту, але не забезпечує такої ж високої швидкодії, як C. Java також має широку спільноту та безпеку, але він має більш високі вимоги до ресурсів та може бути повільним. C++ має високу швидкодію та можливості, але він більш складний у використанні та програмуванні [8].

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						22
Змн.	Арк.	№ докум.	Підпис	Дата		

Вибір методу шифрування AES (Advanced Encryption Standard) для реалізації локального зберігача паролів можна обґрунтувати наступними факторами: [4].

1. Стандартизація і широке використання: AES є прийнятим і рекомендованим стандартом шифрування, який використовується в багатьох сферах, включаючи інформаційну безпеку, фінансові послуги та комунікації. Його використання дозволяє забезпечити сумісність з іншими системами і продуктами, що використовують AES для шифрування.
2. Безпека: AES вважається надійним і стійким алгоритмом шифрування. Він був підданий широкому криптоаналізу та оцінці безпеки, і його стійкість була виявлена. AES використовує симетричне шифрування, що забезпечує швидкість і ефективність операцій шифрування та дешифрування безпосередньо на клієнтському пристрої.
3. Швидкодія: AES має ефективну реалізацію та високу швидкодію. Це особливо важливо для локального зберігача паролів, оскільки шифрування та розшифрування паролів повинні відбуватися миттєво для забезпечення зручності користувача.
4. Розповсюджені бібліотеки та підтримка: Існує широкий вибір криптографічних бібліотек та фреймворків, які надають реалізацію AES, що дозволяє легко використовувати його в програмному продукті. Багато мов програмування та платформ надають підтримку AES, що спрощує його інтеграцію та використання.
5. Розмір ключа: AES підтримує ключі різної довжини, включаючи 128, 192 та 256 біт. Це дозволяє вибрати достатньо довгий ключ для забезпечення високого рівня безпеки.

Загалом, вибір AES для локального зберігача паролів є обґрунтованим, оскільки цей алгоритм є стандартом безпеки, що забезпечує надійне

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		23

шифрування, швидкодію та широку підтримку. Враховуючи чутливу природу паролів, важливо використовувати надійний метод шифрування, такий як AES, для забезпечення безпеки та захисту конфіденційності паролів користувачів [11].

Висновки до першого розділу

У даному розділі було проведено аналіз напрямків використання інформаційних технологій для реалізації локального сховища паролів. Було описано проект і поставлено задачу, що полягає у розробці та реалізації локального зберігача паролів забезпеченого безпекою та зручністю використання.

Метою проекту є розробка та реалізація локального зберігача паролів, який забезпечуватиме безпеку та зручність зберігання та керування паролями користувачів. Для досягнення цієї мети були визначені наступні завдання:

1. Визначення вимог до локального зберігача паролів, які враховують потреби користувачів і стандарти безпеки даних.
2. Аналіз існуючих рішень, які використовуються для зберігання паролів, з метою визначення їх переваг і недоліків.
3. Проектування та розробка архітектури програмного продукту, включаючи вибір необхідних технологій та засобів розробки.
4. Розробка та тестування програмного коду з врахуванням встановлених вимог та архітектурних рішень.
5. Документування та підготовка програмного продукту до використання, забезпечення зрозумілості та легкості використання для користувачів.
6. Оцінка ефективності розробленого програмного продукту з точки зору безпеки, функціональності та продуктивності.

Під час огляду існуючих програмних продуктів було виявлено різноманітні рішення для зберігання паролів, проте не всі з них відповідають вимогам безпеки та зручності використання.

Також було проведено обґрунтування та вибір технологій для розробки локального зберігача паролів. Було враховано фактори, такі як підтримка технологій шифрування, широкий вибір бібліотек та фреймворків, а також розповсюдженість та швидкодія.

Отже, проведений аналіз дозволяє зробити висновок, що розробка локального зберігача паролів з використанням методу шифрування AES та відповідних технологій є доцільною і дозволить забезпечити безпеку та зручність зберігання паролів користувачів. У наступних розділах будуть детальніше розглянуті архітектура програмного продукту, розробка та тестування програмного коду, а також оцінка ефективності розробленого програмного продукту.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		25

Розділ 2. ПРОЕКТУВАННЯ ЛОКАЛЬНОГО МЕНЕДЖЕРА ПАРОЛІВ

2.1. Визначення варіантів використання та структури системи

Для забезпечення безперервного виконання всіх завдань і функцій, які система має виконувати, і для відповідності вимогам і стандартам, необхідно описати детальніше високо рівневі вимоги до локального менеджера паролів.

1. Безперервна функціональність: Система повинна працювати надійно, забезпечуючи доступ до паролів у будь-який момент часу. Немає прийнятних перерв у роботі системи, оскільки це може вплинути на продуктивність та безпеку користувачів.
2. Стандарти і специфікації: Система повинна відповідати всім встановленим стандартам і специфікаціям, що стосуються безпеки та захисту паролів. Це включає в себе криптографічні протоколи, використання безпечних алгоритмів шифрування, аутентифікацію користувачів та інші важливі вимоги.
3. Умови для користувачів: Користувачі повинні дотримуватись певних умов при використанні системи керування паролями. Це може включати в себе встановлення складних паролів, регулярну зміну паролів.
4. Основні ділові переваги: Система повинна мати основні переваги, які сприяють ефективному керуванню паролями. Це має включати в себе централізоване управління паролями, автоматичне створення складних паролів.
5. Можливості системи: Система повинна мати широкий спектр функцій і можливостей, які полегшують керування паролями. Це може включати в себе зберігання і захист паролів.

Варіанти використання локального менеджера паролів:

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		26

1. Збереження та керування паролями: Основна функція локального менеджера паролів полягає у збереженні та безпечному керуванні паролями для різноманітних онлайн-сервісів, акаунтів та програм. Користувачі можуть створювати нові записи паролів, зберігати вже існуючі паролі, а також виконувати пошук, редагування та видалення паролів.
2. Генерація сильних паролів: Менеджер паролів забезпечує можливість генерації випадкових, сильних та надійних паролів для нових акаунтів або зміни існуючих паролів. Користувачі можуть налаштовувати параметри генерації, такі як довжина паролю та використовувані символи (цифри, літери, спеціальні символи), для створення унікальних і складних паролів.
3. Захист паролів: Локальний менеджер паролів забезпечує високий рівень захисту збережених паролів. Він шифрується із використанням AES алгоритму шифрування, що забезпечує безпеку та конфіденційність інформації.

Нижче наведена діаграма варіантів використання.

Локальний менеджер паролів надає різні можливості для неавторизованого та зареєстрованого користувача:

Неавторизований користувач:

1. Створення акаунту: Неавторизований користувач може створити новий акаунт у локальному менеджері паролів. Для цього він повинен ввести пароль. Після успішної реєстрації, користувач стає зареєстрованим та отримує доступ до додаткових функцій.

Зареєстрований користувач:

1. Перегляд паролів: Зареєстрований користувач має можливість переглядати всі збережені паролі у своєму обліковому записі. Він може бачити інформацію про кожен пароль, таку як назва акаунту або сервісу, пароль.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						27
Змн.	Арк.	№ докум.	Підпис	Дата		

2. Видалення паролів: Зареєстрований користувач може видаляти непотрібні або застарілі паролі зі свого облікового запису. Видалення паролів забезпечує оновлення та підтримку актуального списку паролів.
3. Створення нових паролів: Зареєстрований користувач може створювати нові паролі для різних акаунтів або сервісів. Він може вибрати параметри генерації паролів, такі як довжина, використовувані символи (цифри, літери, спеціальні символи) або рівень складності. Це дозволяє зареєстрованому користувачу створювати міцні та унікальні паролі для забезпечення безпеки

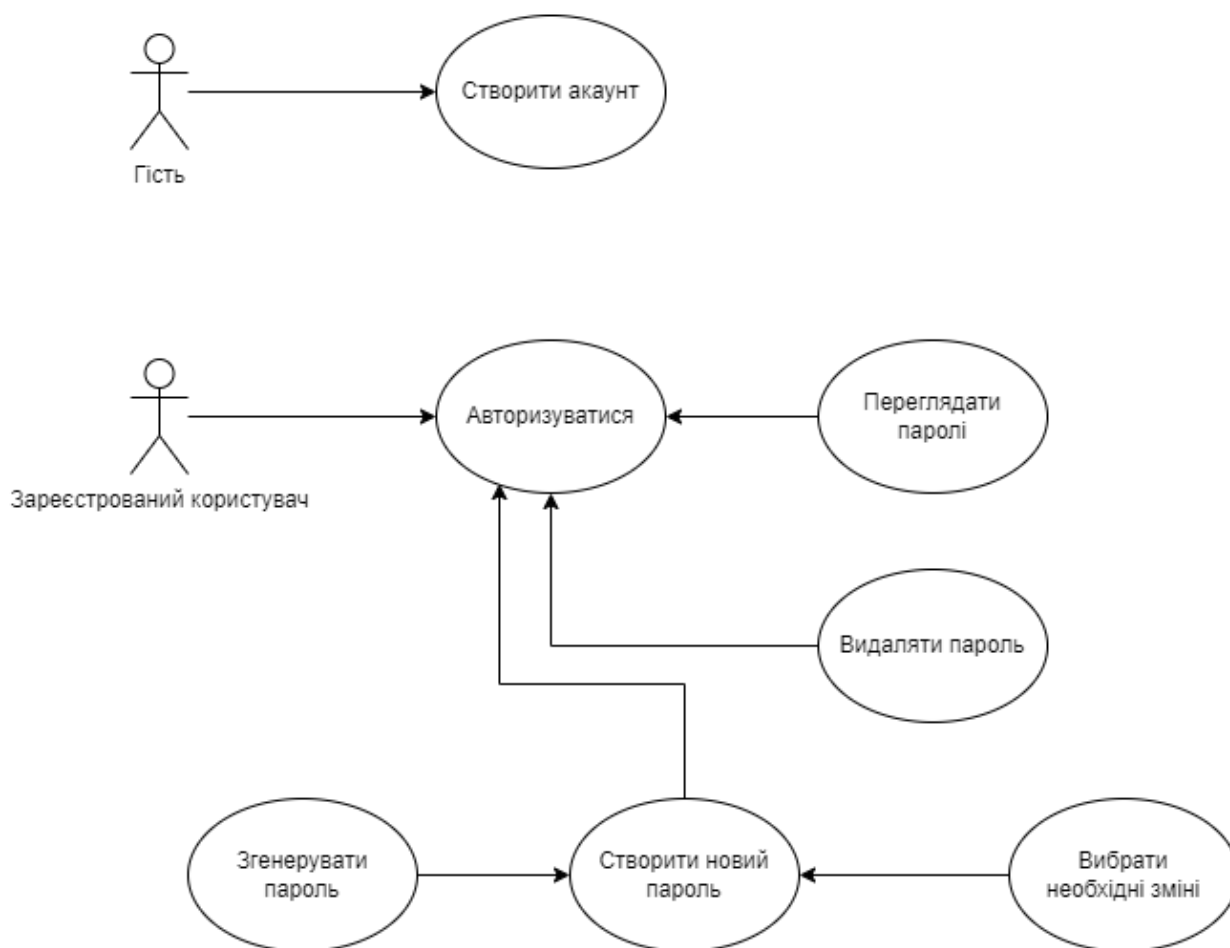


Рис. 2.1 - Діаграма варіантів використання

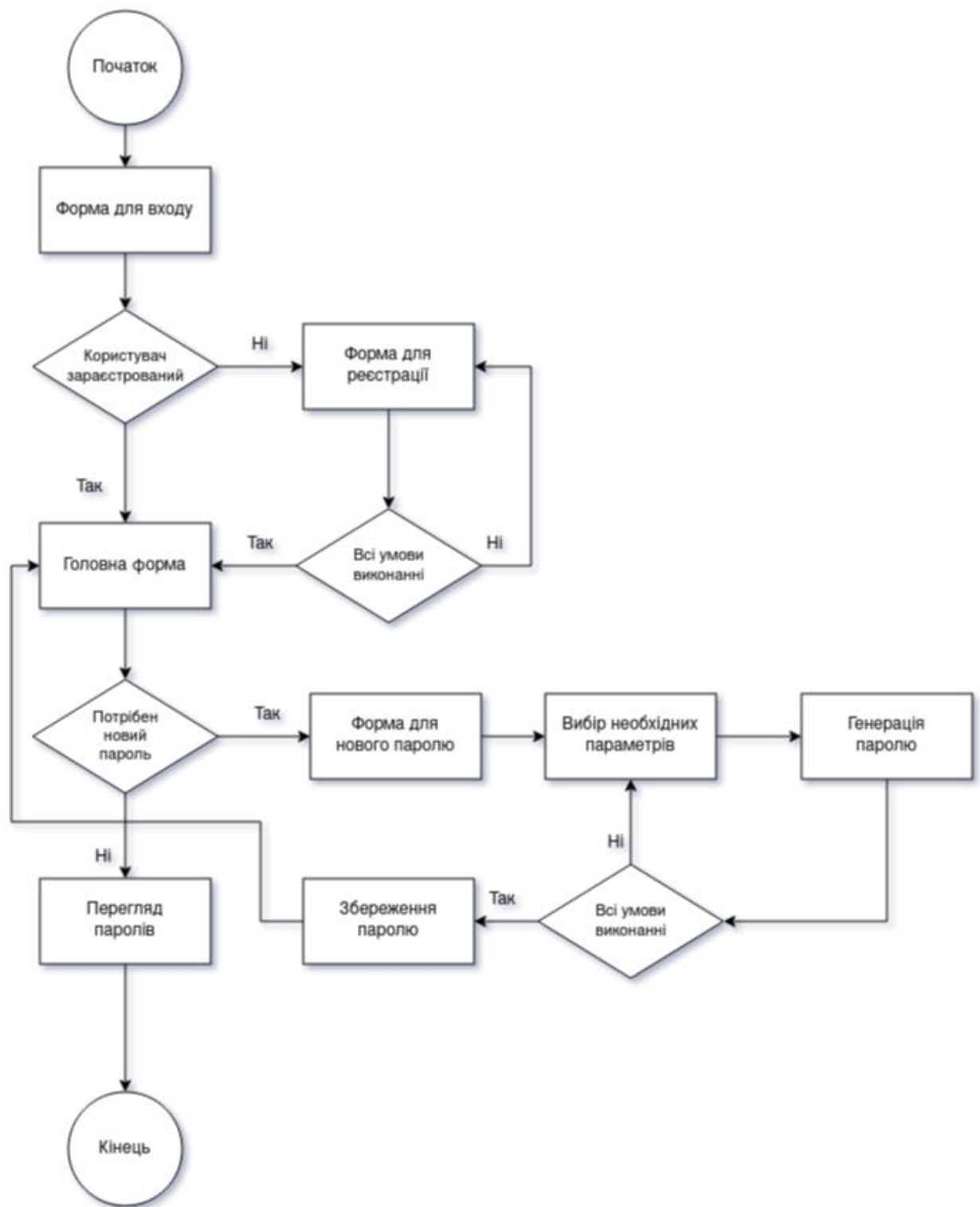


Рис. 2.2 - Загальний алгоритм програми

Загальний алгоритм програми локального менеджера паролів можна описати наступним чином:

1. Запуск програми: Користувач запускає десктопний локальний менеджер паролів на своєму пристрої.

2. Авторизація: Користувач має можливість авторизуватись у програмі, вводячи свій пароль. Якщо користувач вже зареєстрований, він може увійти до свого облікового запису. В іншому випадку, користувач може створити новий обліковий запис.
3. Головне меню: Після авторизації або створення акаунту користувач потрапляє в головне меню програми. Тут він може вибрати бажану опцію, яка дозволить йому виконувати певні дії з паролями.
4. Перегляд паролів: Користувач може переглядати список збережених паролів, який відображається у вигляді списку. Кожен пароль має свої атрибути, такі як назва акаунту, пов'язані дані та інструкції.
5. Видалення паролів: Користувач може вибрати пароль, який бажає видалити зі свого облікового запису. Видалення паролю забезпечує оновлення та підтримку актуального списку паролів.
6. Створення нових паролів: Користувач має можливість створювати нові паролі для різних акаунтів або сервісів. Він може встановити параметри генерації паролю, такі як довжина, використовувані символи (цифри, літери верхнього/нижнього регістру, спецсимволи) та інші вимоги.
7. Збереження паролів: Кожен створений або відредагований пароль зберігається у системі. Збережені паролі зашифровуються для забезпечення безпеки. Вони можуть бути доступні тільки авторизованому користувачу.
8. Завершення роботи: Користувач має можливість вийти з програми або завершити її роботу. При виході з програми, всі зміни та збережені паролі залишаються в системі і будуть доступні під час наступного запуску програми.

Структура проекту:

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		30

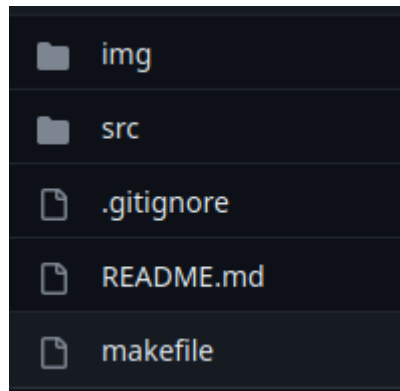


Рис. 2.3 - Загальна структура проекту

Пояснення структури:

Коренева папка проекту:

- README.md: Файл, який містить опис проекту, інструкції щодо його встановлення та використання.
- Makefile: Файл, який містить правила збірки та компіляції програмного коду.

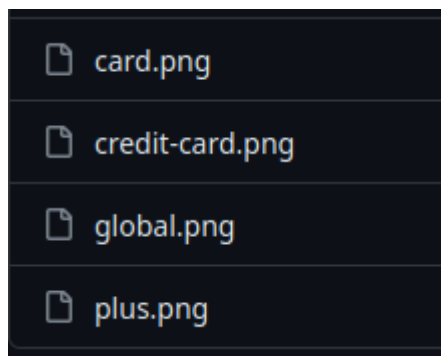


Рис. 2.4 - Вміст папки img

Папка img:

Містить іконки необхідні для програми.

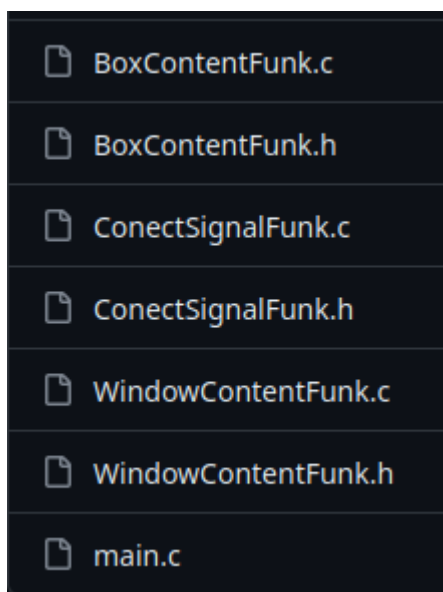


Рис. 2.5 - Вміст папки src

Папка src:

- BoxContentFunk.c: Файл містить функції, які використовуються для заповнення боксів.
- BoxContentFunk.h: Файл містить оголошення функцій, які використовуються для заповнення боксів.
- ConnectSignalFunk.c: Файл містить функції, які використовуються для реалізації сигналів натиснення кнопок та інших подій.
- ConnectSignalFunk.h: Файл містить оголошення функцій, необхідних для реалізації сигналів натиснення кнопок та інших подій.
- WindowContentFunk.c: Файл містить функції, які використовуються для заповнення вікон.
- WindowContentFunk.h: Файл містить оголошення функцій, які використовуються для заповнення вікон.
- main.c: Головний файл програми, де знаходиться точка входу.

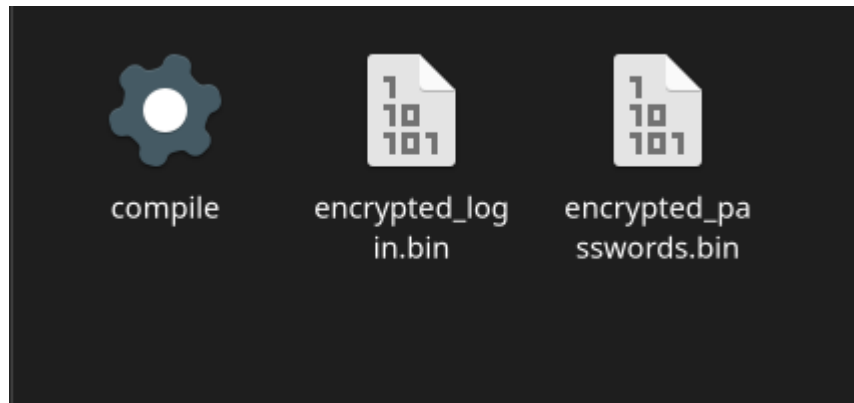


Рис. 2.6 - Вміст папки bin

Папка bin:

Хоча ці файли не згадані в загальній структурі проекту, вони є не менш важливими для проекту.

- **compile:** Цей файл є готовою скомпільованою програмою.
- **encrypted_login.bin:** Цей файл є важливим для забезпечення безпеки та захисту вхідних даних. Він містить зашифровану інформацію про вхідні дані.
- **encrypted_passwords.bin:** Цей файл також є важливим для безпеки даних. Він містить зашифровану інформацію про паролі користувачів.

2.2. Проектування та реалізація алгоритмів роботи системи

Гарний алгоритм генерації паролів є важливим аспектом безпеки і захисту інформації в сучасному цифровому світі. Ось декілька причин, чому важливо мати ефективний алгоритм генерації паролів:

1. **Захист від перебору:** Гарний алгоритм генерації паролів ускладнює завдання атакувачам, які намагаються зламати паролі методом перебору. Якщо паролі генеруються випадковим чином і містять комбінацію символів, цифр і спеціальних знаків, то їх важко відгадати навіть за допомогою потужних обчислювальних ресурсів [11].

2. **Захист від словникових атак:** Ефективний алгоритм генерації паролів уникне використання очевидних комбінацій, таких як слова зі словників або поширені паролі, що легко здогадатися. Це запобігає атакам, які використовують словникові методи.
3. **Унікальність паролів:** Хороший алгоритм генерації паролів забезпечує унікальність паролів для кожного користувача. Це важливо, оскільки у разі, якщо один пароль розкривається, інші облікові записи залишаються безпечними.
4. **Зручність для користувача:** Хоча гарний алгоритм генерації паролів забезпечує складність та безпеку, він також може бути зручним для користувача. Забезпечення випадковості та автоматичної генерації паролів полегшує завдання користувачам, які не мають сильних паролів або уявлення про стійкість.
5. **Забезпечення довгострокової безпеки:** Хороший алгоритм генерації паролів враховує сучасні стандарти безпеки і захисту даних. Він використовує сильну криптографію та відповідає рекомендаціям щодо довжини, складності та періодичності зміни паролів.

Всі ці аспекти важливі для забезпечення безпеки облікових записів, захисту особистих даних та запобігання несанкціонованому доступу до інформації. Тому нижче наведений створений алгоритм генерації паролів.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		34

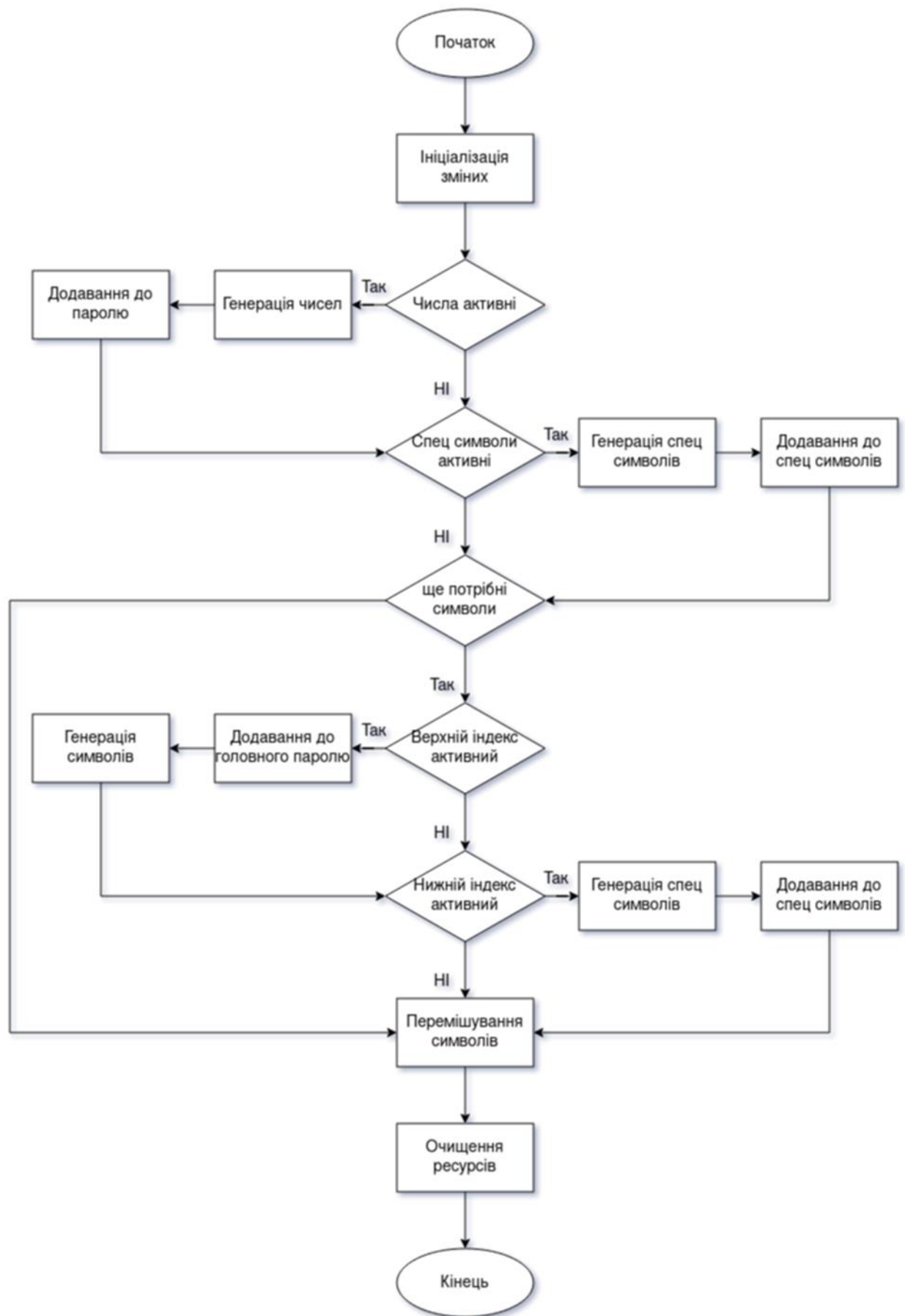


Рис. 2.7 - Загальний алгоритм роботи генерації паролів

Нижче наведено фрагмент коду, який виконує описаний алгоритм:

```
GString *password = g_string_new("");

if (is_number_active) {
    for (int i = 0; i < min_numb; i++) {
        gchar digit = g_random_int_range('0', '9' + 1);
        g_string_append_c(password, digit);
    }
}

if (is_special_active) {
    const gchar *special_chars = "!@#$$%^&*()";
    int special_chars_count = strlen(special_chars);

    for (int i = 0; i < min_special; i++) {
        int random_index = g_random_int_range(0, special_chars_count);
        gchar special_char = special_chars[random_index];
        g_string_append_c(password, special_char);
    }
}

int remaining_length = length - password->len;

if (remaining_length > 0) {
    gchar *character_set = "";

    if (is_upper_active)
        character_set = g_strconcat(character_set,
"ABCDEFGHIJKLMNOPQRSTUVWXYZ", NULL);

    if (is_lower_active)
        character_set = g_strconcat(character_set,
"abcdefghijklmnopqrstuvwxyz", NULL);

    if (is_number_active)
        character_set = g_strconcat(character_set, "0123456789", NULL);

    if (is_special_active)
        character_set = g_strconcat(character_set, "!@#$$%^&*()", NULL);

    int character_set_length = strlen(character_set);

    for (int i = 0; i < remaining_length; i++) {
        int random_index = g_random_int_range(0, character_set_length);
        gchar random_char = character_set[random_index];
        g_string_append_c(password, random_char);
    }

    g_free(character_set);
}

gchar *result = g_string_free(password, FALSE);
return result;
```

Цей алгоритм генерації паролів працює наступним чином:

1. Створюється новий порожній об'єкт рядка `password` за допомогою функції `g\_string\_new()`.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		36

2. Якщо прапорець `'is_number_active'` встановлений, то цикл виконується `'min_numb'` разів. Кожного разу генерується випадкове число у діапазоні від `'0'` до `'9'`, яке представляє цифру, і ця цифра додається до рядка `'password'`.
3. Якщо прапорець `'is_special_active'` встановлений, то цикл виконується `'min_special'` разів. Кожного разу генерується випадковий індекс в межах довжини рядка `'special_chars'`, і символ з цього індексу додається до рядка `'password'`.
4. Обчислюється залишкова довжина, яка обчислюється шляхом віднімання поточної довжини `'password'` від загальної довжини `'length'`.
5. Якщо залишкова довжина більша за 0, створюється порожній рядок `'character_set'`, який буде використовуватись для вибору додаткових символів для паролю.
6. Залежно від встановлених прапорців (`'is_upper_active'`, `'is_lower_active'`, `'is_number_active'`, `'is_special_active'`), до `'character_set'` додаються відповідні набори символів, які визначаються за допомогою функції `'g_strconcat()'`.
7. Обчислюється довжина `'character_set'`.
8. Цикл виконується `'remaining_length'` разів. На кожній ітерації генерується випадковий індекс в межах довжини `'character_set'`, і символ з цього індексу додається до рядка `'password'`.
9. Після завершення циклу `'character_set'` вивільняється з пам'яті за допомогою функції `'g_free()'`.
10. Рядок `'password'` конвертується в звичайний C-стрічковий масив за допомогою функції `'g_string_free()'`. Параметр `'FALSE'` вказує, що пам'ять, виділена для рядка, не потребує вивільнення.
11. Згенерований пароль повертається як результат функції.

Цей алгоритм дозволяє генерувати паролі з комбінації чисел, спеціальних символів, великих і малих літер. Він забезпечує випадковість і різноманітність паролів, що допомагає підвищити їхню безпеку. Крім того, він може легко адаптуватись до зміни вимог щодо паролів шляхом встановлення відповідних параметрів.

Наступним кроком є вибір алгоритму шифрування. Вибір гарного алгоритму генерації паролів є важливою складовою безпеки та захисту інформації в сучасному цифровому світі. На сьогоднішній день велика кількість користувачів використовують паролі для доступу до своїх особистих акаунтів, електронних платіжних систем, соціальних мереж та багатьох інших онлайн-сервісів. Завдяки широкому використанню паролів, їх безпека стає надзвичайно важливою.

Однак, багато користувачів використовують слабкі, прості або прогнозовані паролі, які легко піддаються атакам і злому. Це створює ризик витоку конфіденційної інформації, зловживання акаунтами, крадіжки особистих даних та інших кіберзагроз. Часто хакери використовують різні методи атак, такі як перебор паролів, словникові атаки, використання підкорених паролів та інші техніки для злому паролів, особливо якщо вони є слабкими.

Тому вибір гарного алгоритму генерації паролів є важливим аспектом безпеки. Ефективний алгоритм генерації паролів повинен забезпечувати достатню складність та унікальність паролів. Він повинен створювати паролі, які складаються з комбінації великих і малих літер, цифр, спеціальних символів, а також бути достатньо довгими для ускладнення їх перебору.

Використання гарного алгоритму генерації паролів допомагає збільшити рівень безпеки та унеможливити легкий злом паролів. Це робить паролі більш стійкими до атак і забезпечує захист конфіденційної інформації та особистих даних. Важливо також враховувати, що регулярна зміна паролів та

унікальність паролів для різних акаунтів також є важливими практиками безпеки.

Враховуючи широке використання паролів та постійно зростаючі кіберзагрози, розробка та використання гарного алгоритму генерації паролів стає необхідним завданням для забезпечення безпеки інформації. Це допомагає захистити особисті дані, запобігти злому акаунтів та зберегти важливу інформацію в безпеці. Тому важливо приділяти належну увагу вибору та використанню гарного алгоритму генерації паролів для забезпечення надійності та безпеки цифрових активів.

Алгоритм AES (Advanced Encryption Standard) був вибраний через свою ефективність, надійність і широке використання в криптографічних застосуваннях. AES є симетричним алгоритмом шифрування, що означає, що для шифрування та розшифрування використовується один і той же ключ. Він забезпечує сильний рівень безпеки і використовується у багатьох сферах, таких як захищений обмін даними, зберігання конфіденційної інформації та інші застосування, де безпека даних є пріоритетом.

Алгоритм AES базується на заміщенні і перестановці байтів у вхідному блоку даних за допомогою ключа шифрування. Він використовує операції побітового XOR, зсувів, заміщення та лінійних перетворень для створення криптостійкого шифротексту. AES має різні режими роботи, такі як ECB (Electronic Codebook), CBC (Cipher Block Chaining), CTR (Counter) та інші, які дозволяють використовувати його для різних типів даних і застосувань [9].

Нижче наведено загальний алгоритм роботи AES:

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		39

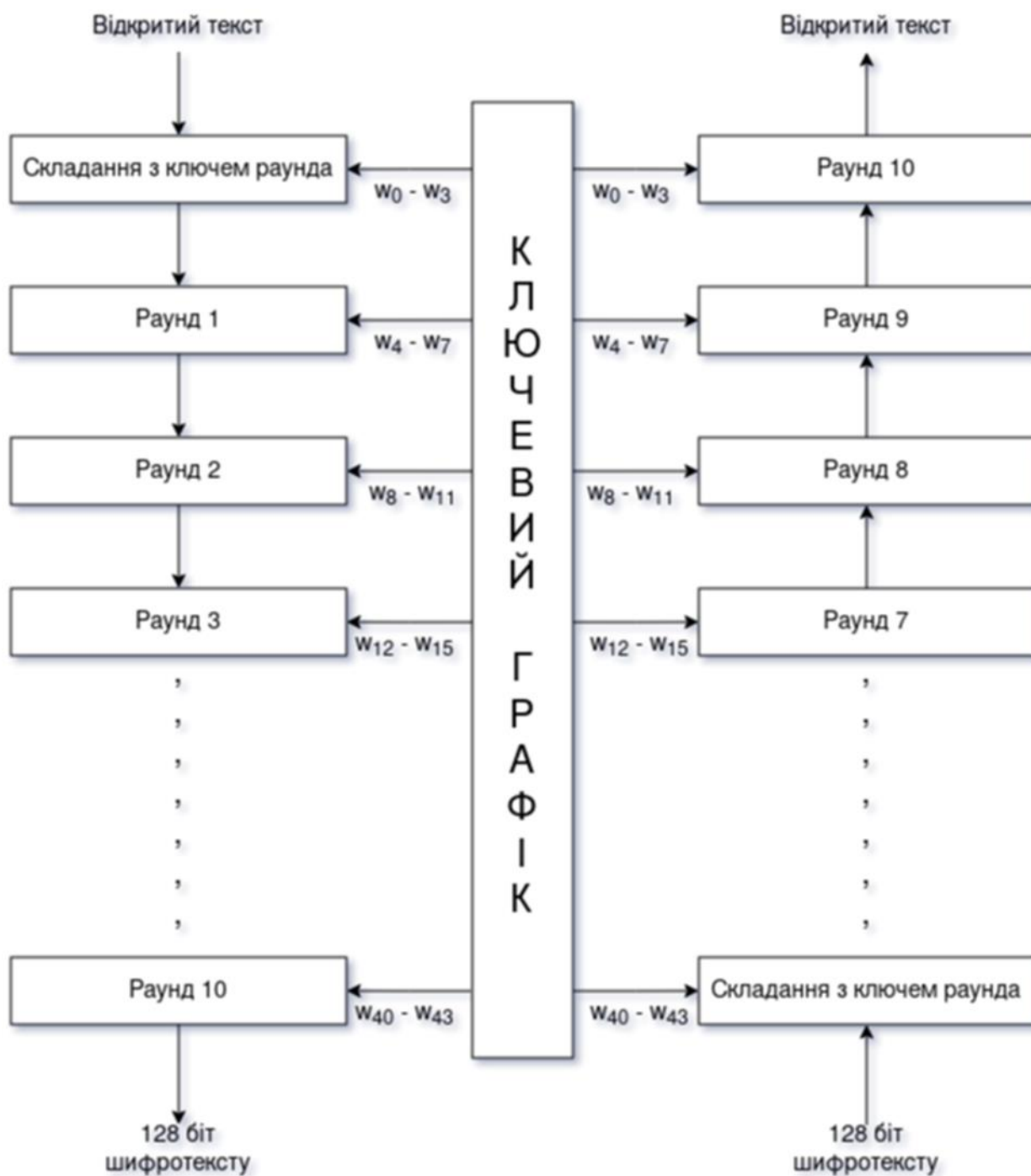


Рис. 2.8 - Блок схема алгоритму AES

Загальний алгоритм роботи AES:

1. Розбивка вхідного тексту на блоки однакового розміру.
2. Ініціалізація ключа шифрування.
3. Поблочне шифрування кожного блоку даних з використанням ключа.

4. Повторення процесу шифрування для кожного блоку, якщо є багато блоків.
5. Формування шифротексту з зашифрованих блоків.
6. Дешифрування шифротексту за допомогою ключа розшифрування та обернених операцій шифрування для кожного блоку даних.
7. Повторення процесу розшифрування для кожного блоку, якщо є багато блоків.
8. Формування вихідного тексту з розшифрованих блоків.
9. Таким чином, AES забезпечує безпечне шифрування та розшифрування даних з використанням сильного ключа, що робить його одним з найбільш популярних алгоритмів шифрування в сучасній криптографії.

Використання готової бібліотеки для шифрування AES в мові програмування C має декілька важливих переваг.

1. Надійність і безпека: Бібліотека AES реалізує стандартний алгоритм шифрування, що є широко використовуваним і визнаним надійним у галузі криптографії. Вона пройшла численні перевірки безпеки і тестування, що забезпечує стійкість та надійність застосування.
2. Швидкодія: Готова бібліотека містить оптимізований код, який забезпечує високу швидкодію шифрування і дешифрування. Це важливо, особливо якщо потрібно обробляти великі обсяги даних або виконувати шифрування в реальному часі.
3. Стандартизація: Алгоритм AES є стандартизованим і використовується в різних сферах, включаючи комунікації, захист даних та інформаційну безпеку. Використання готової бібліотеки сприяє забезпеченню взаємодії та сумісності з іншими системами та протоколами, які також використовують AES.
4. Зменшення ризику помилок: Розробка власного алгоритму шифрування може бути складною та підлягати ризику помилок.

Використання готової бібліотеки забезпечує високий рівень якості і надійності, оскільки код вже пройшов випробування та перевірки безпеки.

Функції ``aes_encrypt`` і ``aes_decrypt`` надають зручний і простий спосіб використовувати алгоритм AES у вашій програмі. Функція ``aes_encrypt`` приймає вхідний текст ``input`` і ключ шифрування ``key``, а потім використовує ці дані для шифрування. Результат шифрування зберігається у вихідному масиві ``output``. Функція ``aes_decrypt`` працює на протилежних принципах і використовує ключ розшифрування ``key`` та зашифрований текст ``input`` для відновлення оригінальних даних, які зберігаються у вихідному масиві ``output``.

Код цих функцій:

```
// Функція шифрування AES
void aes_encrypt(const char *input, const char *key, char *output)
{
    AES_KEY aes_key;
    AES_set_encrypt_key((const unsigned char *)key, 128, &aes_key);
    AES_encrypt((const unsigned char *)input, (unsigned char *)output,
&aes_key);
}

// Функція дешифрування AES
void aes_decrypt(const char *input, const char *key, char *output)
{
    AES_KEY aes_key;
    AES_set_decrypt_key((const unsigned char *)key, 128, &aes_key);
    AES_decrypt((const unsigned char *)input, (unsigned char *)output,
&aes_key);
}
```

Такий підхід дозволяє зосередитись на основних завданнях програми, а не на власноручній реалізації алгоритму шифрування. Використання готової бібліотеки допомагає зекономити час і зусилля, а також забезпечує надійність та ефективність шифрування даних.

2.3. Реалізація додатку локальний менеджер паролів

Для початку опишемо файл ``main.c``. Він є головним файлом додатку "Локальний менеджер паролів". Він містить функцію ``main``, яка є вхідною точкою програми. У цій функції створюється об'єкт ``GtkApplication`` з

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						42
Змн.	Арк.	№ докум.	Підпис	Дата		

ідентифікатором "org.gtk.example" і реєструється зв'язок з подією `activate`. Після цього викликається функція `g\_application\_run`, яка починає цикл обробки подій GTK. В кінці роботи додатку пам'ять, виділена для об'єкта `GtkApplication`, звільняється.

Наступним є файл Файл `WindowContentFunk.h`. Він оголошує функції, які відповідають за вміст вікон додатку "Локальний менеджер паролів". Включення `BoxContentFunk.h` в файл забезпечує доступ до функцій, пов'язаних з вмістом боксів.

Оголошені функції включають:

- `content\_login\_window`: Функція, яка відповідає за вміст вікна входу в систему. Приймає об'єкт `GtkApplication` в якості аргументу.
- `content\_main\_window`: Функція, яка відповідає за вміст головного вікна програми. Приймає об'єкт `GtkApplication` в якості аргументу.

Директива `#pragma once` вказує препроцесору уникати повторного включення цього файлу. Вона забезпечує, що файл буде включений тільки один раз, незалежно від кількості його включень у проєкті.

Файл `WindowContentFunk.c` містить реалізацію функцій, які відповідають за вміст вікон додатку "Локальний менеджер паролів".

Функція `content\_main\_window` відповідає за вміст головного вікна програми. Вона створює вікно, встановлює його заголовок та розмір, а також створює контейнер `container` для розміщення інших елементів. Далі вона створює та додає до контейнера розділову лінію, контейнер для вмісту та підвал. Вміст вікна розташовується в вертикальному напрямку, а контейнер для вмісту отримує прокрутку, якщо його розмір перевищує доступний простір. Результат виконання цього коду можна побачити на Рис. 2.9.

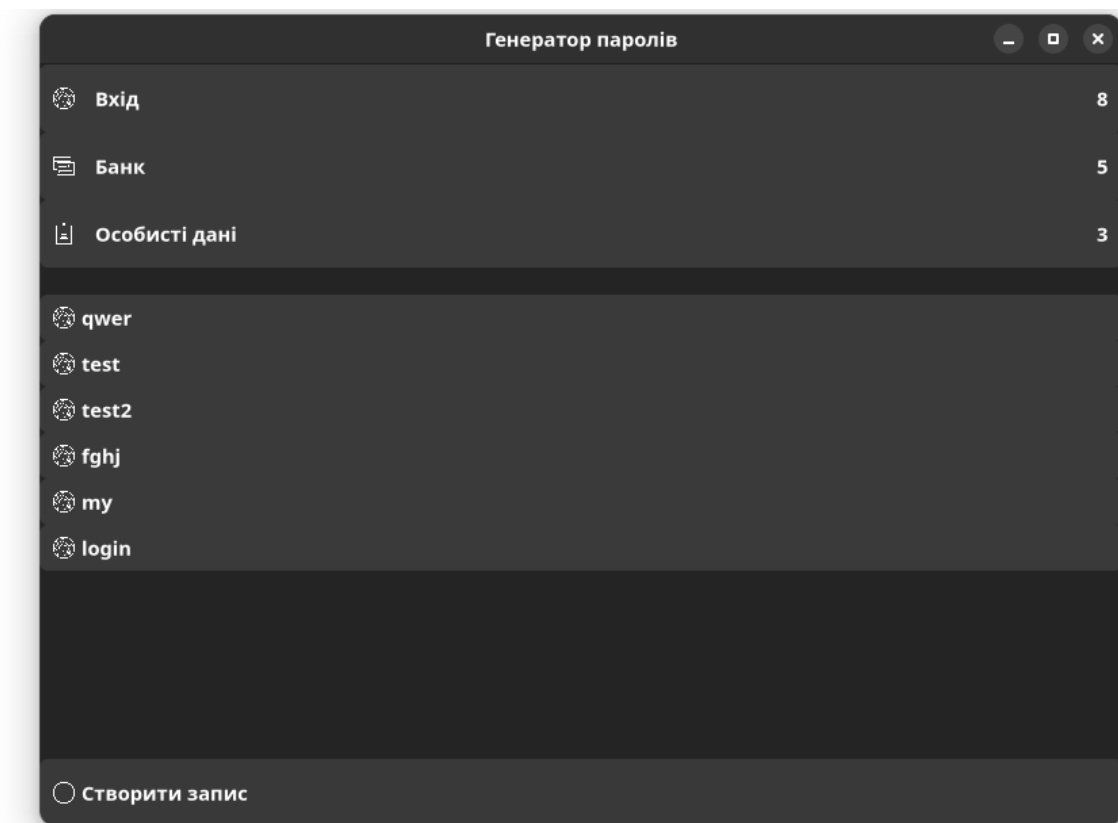


Рис. 2.9 - Головне вікно програми

Функція `'content_login_window'` відповідає за вміст вікна входу в систему. Вона створює вікно, встановлює його заголовок та розмір, а також створює контейнер `'container'` для розміщення інших елементів. Функція `'content_login_container_box'` відповідає за створення вмісту контейнера для входу в систему. Після створення вмісту, вікно відображається.

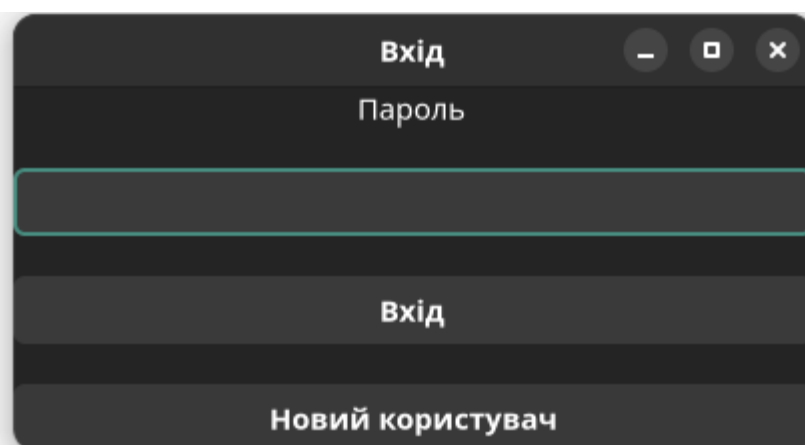


Рис. 2.10 - Вікно для входу

Обидві функції використовують функції та віджети GTK для створення і розміщення вмісту вікон додатку.

Файл `BoxContentFunk.h` містить оголошення функцій, які відповідають за розміщення вмісту вікон додатку "Локальний менеджер паролів" і підключаються до відповідного файлу `ConectSignalFunk.h`.

Функція `content\_login\_container\_box` приймає вказівник на контейнер `container` і відповідає за створення вмісту контейнера для входу в систему.

Функція `content\_header\_box` приймає вказівник на віджет `header` і відповідає за створення вмісту заголовку вікна.

Функція `content\_center\_box` приймає вказівник на віджет `content` і відповідає за створення вмісту основної частини вікна.

Функція `content\_footer\_box` приймає вказівник на віджет `footer` і відповідає за створення вмісту підвалу вікна.

Функція `content\_add\_pass\_header\_box` приймає вказівник на віджет `header\_box` і відповідає за створення вмісту заголовку вікна додавання паролю.

Функція `content\_add\_pass\_properties\_pass\_box` приймає вказівник на віджет `propertiespass\_box` і відповідає за створення вмісту контейнера з властивостями паролю.

Ці функції використовують функції та віджети GTK для створення вмісту вікон додатку і їх розміщення в відповідних контейнерах.

Файл `BoxContentFunk.c` містить реалізацію функцій, які відповідають за вміст вікон додатку "Локальний менеджер паролів":

1. `content\_login\_container\_box`: Ця функція створює контейнер для форми входу. Вона створює текстовий напис, поле введення пароля, кнопку відправки та кнопку нового користувача. Вона з'єднує сигнали кнопок з відповідними обробниками подій.
2. `content\_header\_box`: Ця функція створює блок заголовка з трьома кнопками: `entry\_button`, `card\_button` та `user\_data\_button`. Кожна кнопка має відповідну мітку, лічильник та зображення. Функція встановлює властивості розміру та вирівнювання віджетів.

	Вхід	8
	Банк	5
	Особисті дані	3

Рис. 2.11 - Результат виконання функції `content\_header\_box`

3. `content\_center\_box`: Ця функція зчитує зашифровані паролі з файлу та створює кнопки для кожного запису пароля. Вона також створює мітки для паролів (спочатку приховані). Функція з'єднує сигнали кнопок з обробником подій.

	qwer	
	test	Пароль: IWKU
	test2	
	fghj	
	my	Пароль: 7DB0J9QB8
	login	

Рис. 2.12 - Результат виконання функції `content\_center\_box`

4. `content\_footer\_box`: Ця функція створює блок нижнього колонтитулу з кнопкою "Додати". Вона встановлює розмір та додає зображення та мітку до кнопки.
5. `content\_add\_pass\_header\_box`: Ця функція створює блок заголовка для додавання нового пароля. Вона включає поля для вибору типу, введення назви та генерації пароля. Вона також створює комбінований список для вибору типу пароля.

Тип	Ім'я	Пароль	
			Генерувати

Рис. 2.13 - Результат виконання функції `content\_add\_pass\_header\_box`

6. `content\_add\_pass\_properties\_pass\_box`: Ця функція створює блок для визначення властивостей пароля, таких як довжина, мінімальна кількість цифр та мінімальна кількість спеціальних символів. Вона

включає елементи керування, такі як повзунки, прапорці та кнопки для налаштування значень.

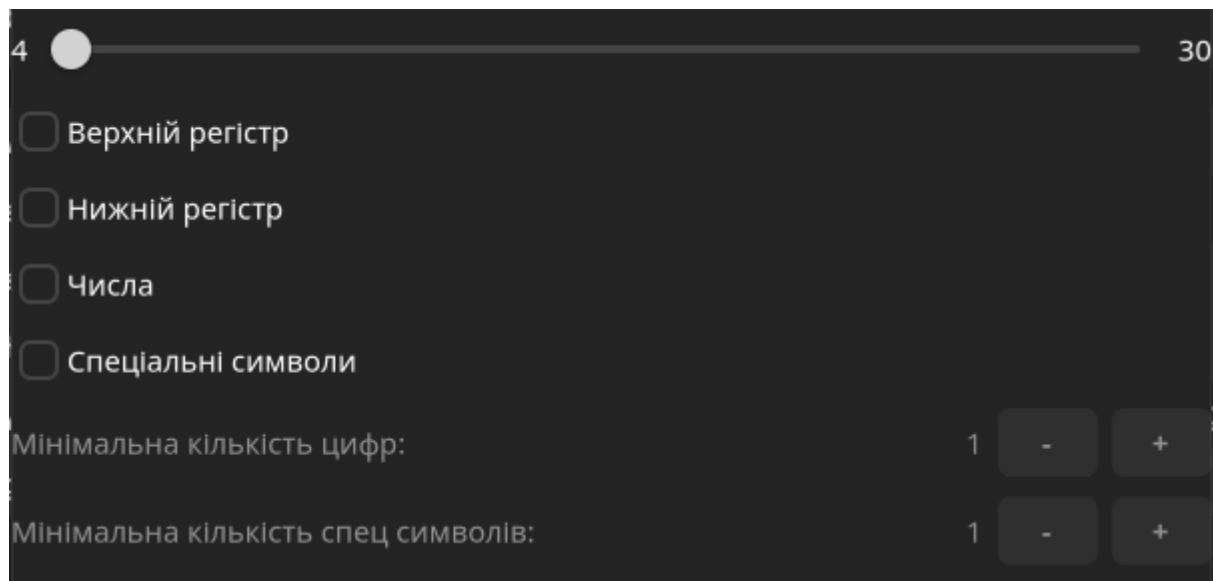


Рис. 2.14 - Результат виконання функції

``content_add_pass_properties_pass_box``

Загалом, ці функції допомагають створювати різні компоненти GUI, такі як форми входу, заголовки, блоки контенту, нижні колонтитули та форми для введення паролів. Код призначений для роботи з бібліотекою GTK у середовищі C.

Файл "ConectSignalFunk.h" містить наступні функції:

1. ``login_button_clicked``: Функція, яка викликається при натисканні кнопки для входу. Приймає два аргументи: об'єкт ``GtkButton`` та віджет ``passentry``.
2. ``add_button_clicked``: Функція, яка викликається при натисканні кнопки для додавання. Приймає два аргументи: об'єкт ``GtkButton`` та даних користувача (``user_data``).
3. ``generate_pass``: Функція, яка генерує випадковий пароль і повертає його як рядок (``gchar *``).
4. ``on_scale_value_changed``: Функція, яка викликається при зміні значення на діапазоні (``GtkRange``). Приймає два аргументи: об'єкт ``GtkRange`` та віджет ``length_scale_label``.

5. ``on_number_checkbutton_press_event``: Функція, яка викликається при натисканні на прапорець (``GtkWidget``) для включення або виключення використання цифр у паролі. Приймає два аргументи: віджет ``GtkWidget`` та ``box`` (віджет, який містить прапорець).
6. ``on_special_checkbutton_press_event``: Функція, яка викликається при натисканні на прапорець (``GtkWidget``) для включення або виключення використання спеціальних символів у паролі. Приймає два аргументи: віджет ``GtkWidget`` та ``box`` (віджет, який містить прапорець).
7. ``on_lower_checkbutton_press_event``: Функція, яка викликається при натисканні на прапорець (``GtkWidget``) для включення або виключення використання строчних літер у паролі. Приймає два аргументи: віджет ``GtkWidget`` та ``user_data`` (додаткові дані користувача).
8. ``on_upper_checkbutton_press_event``: Функція, яка викликається при натисканні на прапорець (``GtkWidget``) для включення або виключення використання прописних літер у паролі. Приймає два аргументи: віджет ``GtkWidget`` та ``user_data`` (додаткові дані користувача).
9. ``on_plus_min_number_button_clicked``: Функція, яка викликається при натисканні кнопки "+" для збільшення значення мінімальної кількості цифр у паролі. Приймає три аргументи: об'єкт ``GtkButton`` та два віджети ``label``.
10. ``on_minus_min_number_button_clicked``: Функція, яка викликається при натисканні кнопки "-" для зменшення значення мінімальної кількості цифр у паролі. Приймає три аргументи: об'єкт ``GtkButton`` та два віджети ``label``.
11. ``on_plus_special_button_clicked``: Функція, яка викликається при натисканні кнопки "+" для збільшення значення мінімальної

кількості спеціальних символів у паролі. Приймає три аргументи: об'єкт `GtkButton` та два віджети `label`.

12. `on\_minus\_special\_button\_clicked`: Функція, яка викликається при натисканні кнопки "-" для зменшення значення мінімальної кількості спеціальних символів у паролі. Приймає три аргументи: об'єкт `GtkButton` та два віджети `label`.

13. `on\_add\_pass\_button\_clicked`: Функція, яка викликається при натисканні кнопки для додавання пароля. Приймає два аргументи: об'єкт `GtkButton` та даних користувача (`user\_data`).

14. `button\_item\_clicked`: Функція, яка викликається при натисканні кнопки елемента. Приймає два аргументи: об'єкт `GtkButton` та даних користувача (`user\_data`).

Ці функції відповідають за обробку подій та взаємодію з різними віджетами GTK для реалізації певної функціональності в програмі.

Файл ConectSignalFunk.c:

- Код починається з включення необхідних заголовочних файлів та оголошення деяких глобальних змінних та зовнішніх віджетів GTK+.
- Функція "generate_pass" генерує пароль на основі заданих параметрів (кількість цифр, спеціальних символів, довжина тощо).
- Функція "on_scale_value_changed" викликається при зміні значення елемента діапазону (слайдера). Вона оновлює мітку, що відображає поточну довжину пароля, та генерує новий пароль.
- Функції "on_number_checkbutton_press_event" і "on_special_checkbutton_press_event" є обробниками подій для прапорців вибору. Вони оновлюють стан відповідних параметрів та генерують новий пароль.
- Функції "on_lower_checkbutton_press_event" і "on_upper_checkbutton_press_event" обробляють події, коли

натискаються прапорці для нижніх та верхніх символів відповідно. Вони оновлюють стан відповідних параметрів та генерують новий пароль.

- Функція "generate_button_clicked" викликається при натисканні кнопки "Generate" (Генерувати). Вона генерує новий пароль та оновлює відповідний елемент введення.
- Функції "on_plus_min_number_button_clicked" і "on_minus_min_number_button_clicked" обробляють події, коли натискаються кнопки "плюс" і "мінус" для мінімальної кількості цифр відповідно. Вони оновлюють мінімальну кількість цифр та генерують новий пароль.
- Функції "on_plus_special_button_clicked" і "on_minus_special_button_clicked" обробляють події, коли натискаються кнопки "плюс" і "мінус" для мінімальної кількості спеціальних символів відповідно. Вони оновлюють мінімальну кількість спеціальних символів та генерують новий пароль.
- Функція "login_button_clicked" викликається при натисканні кнопки "Login" (Увійти). Вона порівнює введений пароль з розшифрованим паролем, збереженим у файлі, і виконує певні дії на основі результату.
- Функція "add_button_clicked" викликається при натисканні кнопки "Add" (Додати). Вона відкриває нове вікно для додавання нового пароля.
- Функція "add_new_user_label_clicked" викликається при натисканні мітки для додавання нового користувача. Вона відкриває нове вікно для створення нового облікового запису.
- Функція "check_password_length" викликається при натисканні кнопки "Create" (Створити) у вікні створення нового облікового

запису. Вона перевіряє довжину введеного пароля і виконує шифрування, якщо довжина знаходиться в заданому діапазоні.

- Функції "aes_encrypt" і "aes_decrypt" використовуються для шифрування та розшифрування паролів за алгоритмом AES.

Висновки до другого розділу

Розділ 2 "Проектування локального менеджера паролів" є ключовим етапом у створенні системи, оскільки він зосереджується на розробці та реалізації самого менеджера паролів. Цей розділ надає детальну інформацію про всі аспекти проекту, починаючи з визначення варіантів використання та структури системи, а закінчуючи реалізацією конкретного додатку.

У підрозділі 2.1 "Визначення варіантів використання та структури системи" проводиться аналіз функціональних вимог та визначення основних компонентів системи. Цей етап дозволяє визначити, як система буде використовуватись користувачами, а також встановити її загальну структуру. Варіанти використання враховують різні сценарії, включаючи створення, збереження, редагування та використання паролів. В результаті цього етапу структура системи стає чіткою, що сприяє ефективному проектуванню та реалізації.

Підрозділ 2.2 "Проектування та реалізація алгоритмів роботи системи" є одним з найважливіших етапів проектування, оскільки він включає розробку алгоритмів, необхідних для правильної роботи системи. У цьому підрозділі здійснюється проектування алгоритмів генерації паролів, шифрування, розшифрування та взаємодії з базою даних. Належна реалізація цих алгоритмів гарантує безпеку паролів та ефективну роботу системи в цілому. Крім того, розробка алгоритмів забезпечує зручний та інтуїтивно зрозумілий інтерфейс користувача, що є важливим аспектом успішного менеджера паролів.

Підрозділ 2.3 "Реалізація додатку локального менеджера паролів" концентрується на конкретній реалізації менеджера паролів. Цей підрозділ

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		51

може включати детальний опис коду, використаних бібліотек або інших технологій для створення додатку. Реалізація включає створення інтерфейсу користувача, імплементацію алгоритмів, роботу з базою даних та взаємодію з користувачем. У цьому підрозділі можуть описуватись ключові функціональні можливості додатку, його архітектура та особливості реалізації.

Загалом, розділ 2 "Проектування локального менеджера паролів" виконує важливу роль у розробці системи. Він забезпечує чітке визначення структури та функціональності системи, а також детальну реалізацію необхідних алгоритмів та компонентів. Цей розділ є фундаментом для подальшої розробки та тестування системи, а також забезпечує надійність та безпеку управління паролями для користувачів.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						52
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ЛОКАЛЬНИМ МЕНЕДЖЕРОМ ПАРОЛІВ

3.1. Порядок встановлення та налаштування параметрів системи.

Вимоги до системи для десктопного додатку локального менеджера паролів (з підтримкою GTK4) на базі Arch Linux:

Для використання десктопного додатку локального менеджера паролів на базі Arch Linux з підтримкою GTK4, потрібен комп'ютер або ноутбук, що відповідає наступним мінімальним вимогам:

1. Операційна система: Arch Linux або похідні дистрибутиви. Arch Linux Цей додаток розроблено для роботи на Arch Linux, одному з популярних дистрибутивів Linux. Його можна встановити на систему, яка базується на Arch Linux, таку як Manjaro, Antergos або інші [3].
2. Процесор: з тактовою частотою від 1,3 до 3,8 ГГц. Додаток працює на комп'ютерах з різними процесорами, які мають тактову частоту від 1,3 ГГц до 3,8 ГГц. Це означає, що його можна використовувати на більшості сучасних комп'ютерів та ноутбуків.
3. Оперативна пам'ять: мінімум 2 ГБ. Для плавної роботи додатку рекомендується мати мінімум 2 ГБ оперативної пам'яті. Більший обсяг оперативної пам'яті дозволить ефективніше виконувати різні завдання та запускати більш об'ємні програми одночасно.
4. Наявність встановленого GTK4. Додаток використовує бібліотеку GTK4 для створення графічного інтерфейсу користувача. Перед використанням додатку переконайтеся, що встановлено GTK4. Ви можете встановити GTK4, використовуючи менеджер пакетів Pacman або AUR.
5. Наявність встановленого пакету OpenSSL. Додаток використовує пакет OpenSSL для шифрування паролів та забезпечення безпеки.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						53
Змн.	Арк.	№ докум.	Підпис	Дата		

Перед використанням додатку переконайтеся, що встановлено пакет OpenSSL. Ви можете встановити його, використовуючи менеджер пакетів Pacman або AUR.

Ці вимоги вважаються мінімальними і гарантують здатність використовувати додаток на вашій системі. Однак, для кращої продуктивності та зручності рекомендується мати комп'ютер з більш потужним процесором та більшим обсягом оперативної пам'яті. Таке обладнання забезпечить більш швидку та плавну роботу додатку, що позитивно вплине на досвід користувачів.

Процес встановлення десктопного додатку локального менеджера паролів з використанням Makefile та розповсюдженням через Git описується наступним чином:

1. Перед встановленням переконайтеся, що у вас встановлені необхідні залежності: GTK4 та OpenSSL. Для цього переконайтеся, що у вас встановлені відповідні пакети GTK4 та OpenSSL або виконайте їх встановлення з допомогою менеджера пакетів вашої операційної системи.
2. Завантажте вихідний код додатку з Git-репозиторію. Виконайте команду ``git clone <URL репозиторію>`` для клонування репозиторію на ваш комп'ютер [15].
3. Перейдіть до каталогу з вихідним кодом додатку. Відкрийте командний рядок або термінал та перейдіть до каталогу, де було здійснене клонування репозиторію.
4. Виконайте команду ``make compile``. Ця команда збирає вихідний код, компілює його та створює виконуваний файл додатку. Виконання цієї команди запустить процес компіляції та зв'язування, використовуючи задані змінні середовища, такі як CXX, CXXFLAGS та LIBS з файлу Makefile [5].

5. Після успішного виконання команди `make compile` створиться каталог `bin`, в якому буде знаходитися виконуваний файл додатку.
6. Ви можете запустити додаток, виконавши команду `./bin/назва\_додатку`, де `назва\_додатку` - це ім'я виконуваного файлу, яке було вказано в Makefile [14].
7. Якщо ви бажаєте очистити створені об'єктні файли та виконуваний файл, виконайте команду `make clean`. Вона видалить файли з розширенням `.o` з каталогу `src` та виконуваний файл з каталогу `bin` [10].

Таким чином, ви успішно встановили десктопний додаток локального менеджера паролів за допомогою Makefile та Git.

3.2. Структура інтерфейсу та порядок роботи з системою

Структура інтерфейсу в додатку локального менеджера паролів є ключовим аспектом, оскільки вона визначає спосіб взаємодії користувача з програмою. Інтерфейс повинен бути зручним, інтуїтивно зрозумілим та ефективним для забезпечення зручного використання системи.

Вікно для входу в систему є першим екраном, з якого користувач починає взаємодію з додатком локального менеджера паролів. Воно має просту і зрозумілу структуру, яка дозволяє користувачеві ввести пароль для отримання доступу до системи.

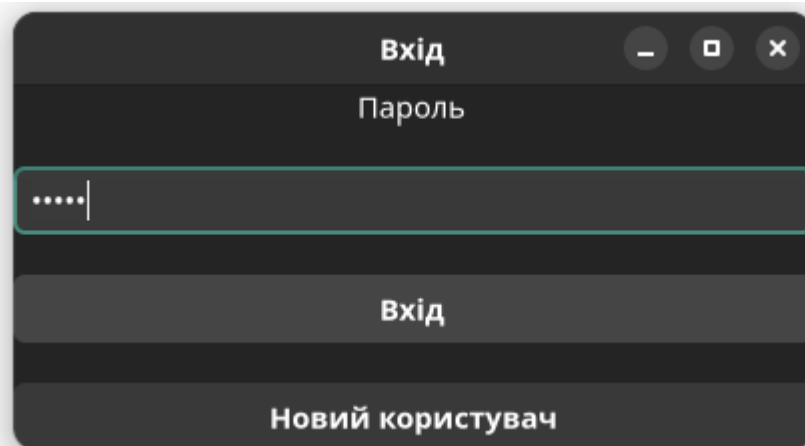


Рис. 3.1 - Форма для входу

Структура вікна для входу в систему:

- Заголовок вікна: Містить назву форми а також тест який каже що нам потрібно ввести.
- Поле для введення паролю: Це текстове поле, де користувач може ввести свій - пароль. Поле приховане (замасковане символами,) для забезпечення безпеки пароля.
- Кнопка "Вхід" : Ця кнопка викликає процес перевірки введеного пароля та авторизації користувача в системі. При натисканні кнопки програма перевіряє правильність введеного пароля та виконує відповідні дії, якщо авторизація успішна.
- Кнопка "Новий користувач": Ця кнопка надає можливість створити новий обліковий запис користувача. При натисканні кнопки відкривається окрема форма для створення користувача.

Вікно створення нового користувача також є важливою частиною додатку де необхідно створити новий обліковий запис для користувача.

Рис. 3.2 - Форма реєстрації

Структура вікна створення нового користувача:

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						56
Змн.	Арк.	№ докум.	Підпис	Дата		

- Заголовок вікна: Містить назву додатку або описову назву сторінки, наприклад "Створення нового користувача".
- Поле для введення паролю: Це текстове поле, де користувач може ввести пароль для нового облікового запису.
- Кнопка "Підтвердити" (або "Створити"): Ця кнопка викликає процес створення нового користувача з введеним паролем. При натисканні кнопки програма перевіряє правильність введеного пароля та виконує відповідні дії, якщо процес створення успішний.

Структура головного вікна включає три блоки, кожен з яких виконує свою функцію та надає користувачеві зручний спосіб взаємодії з додатком. Давайте розглянемо ці блоки більш детально:

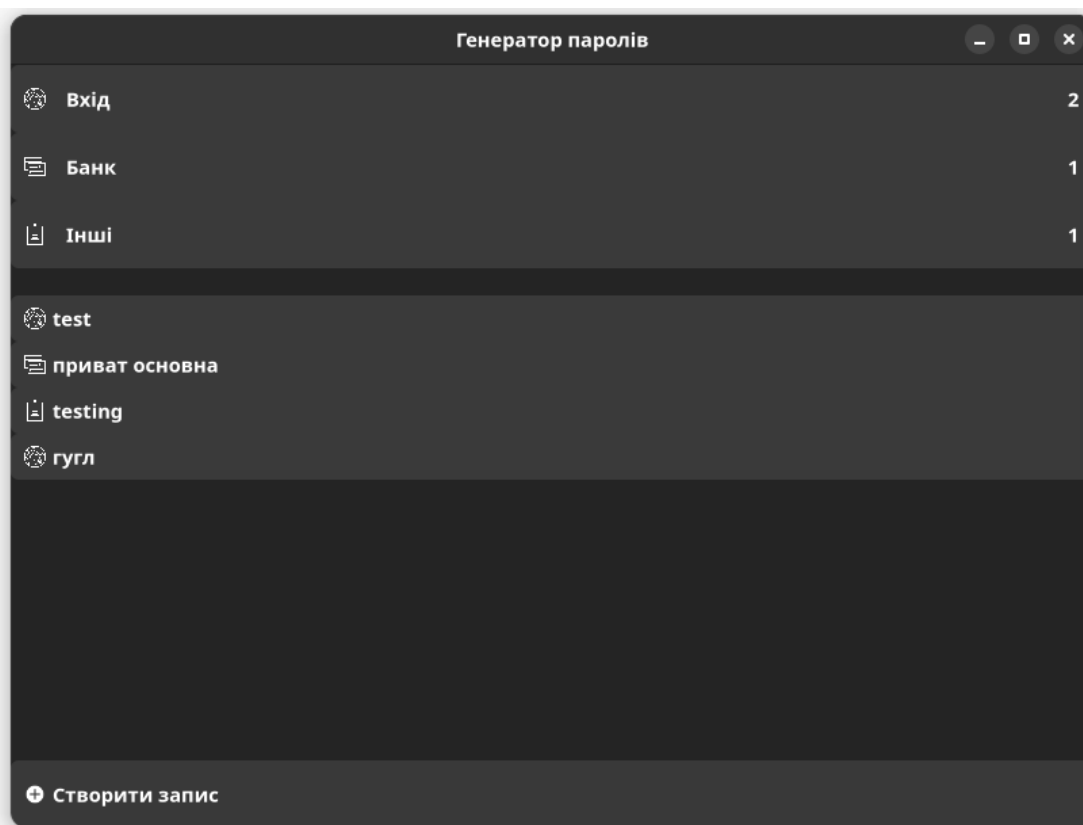


Рис. 3.3 - Головна форма програми

Блок кнопок вибору типу паролю: У цьому блоку розташовані кнопки, що відображають типи паролів, наприклад, "Вхід", "Банк", "Особисті дані" або будь-які інші типи, які відповідають функціональності додатку. Користувач

може обрати потрібний тип паролю, щоб переглянути або змінити відповідний пароль.

Блок кнопок збережених паролів: У цьому блоку розташовані кнопки, які відображають імена збережених паролів. Кожна кнопка містить ім'я паролю, наприклад, ім'я сервісу або акаунту, до якого пароль належить. При натисканні на кнопку, відповідний пароль відображається для перегляду користувачем.

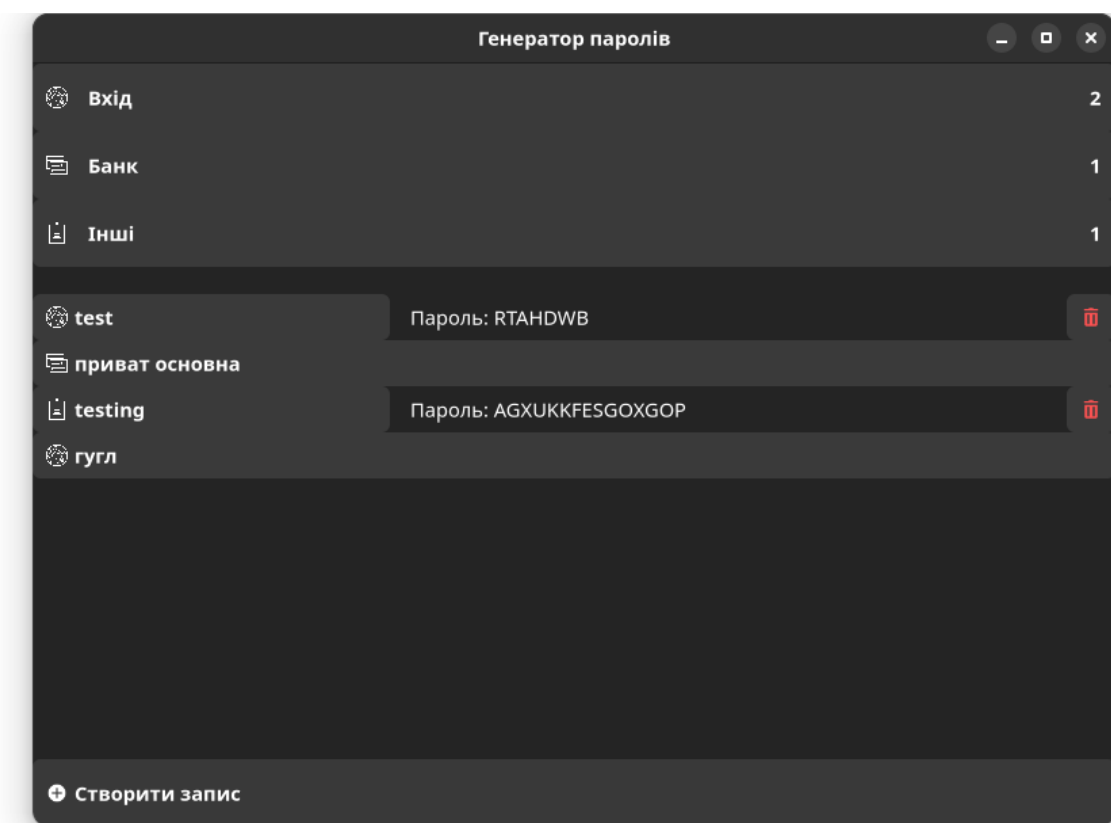


Рис. 3.4 - Показ паролю

Блок кнопки "Створити новий пароль": У цьому блоку розташована кнопка "Створити новий пароль". Ця кнопка надає можливість користувачеві створити новий пароль для додатку або сервісу. При натисканні на цю кнопку відкривається вікно створення нового паролю, де користувач може ввести необхідну інформацію та зберегти новий пароль.

Форма реєстрації складається з трьох блоків, які надають користувачеві зручність і гнучкість при створенні нового паролю. Розглянемо ці блоки детальніше:

Рис. 3.5 - Форма для створення паролю.

Блок вибору типу та введення імені паролю: У цьому блоку розташований список типів паролів, які включають в себе різні категорії, наприклад, "Вхід", "Банк", "Особисті дані". Користувач може обрати потрібний тип паролю зі списку. Крім того, блок містить поле для введення імені паролю, яке дозволяє користувачу надати ім'я або опис паролю для подальшого розпізнавання.

Блок налаштування паролю: У цьому блоку знаходяться налаштування паролю, що дозволяють користувачу контролювати його характеристики. Тут можна встановити довжину паролю та вибрати потрібні опції, які включають верхній і нижній регістр, цифри та символи. Крім того, в блоку розміщуються кнопки, які дозволяють користувачу налаштувати кількість цифр і символів у паролі, змінюючи їх кількість за бажанням.

Блок збереження паролю: У цьому блоку розташована кнопка "Зберегти", яка дозволяє користувачу зберегти введений пароль після встановлення всіх налаштувань. Після натискання на кнопку "Зберегти" новий пароль буде зареєстровано і збережено для майбутнього використання.

Порядок роботи з системою локального менеджера паролів є наступним:

1. Вхід до системи:

1.1. Користувач запускає програму і бачить вікно входу.

1.2. Він вводить свій пароль.

1.3. Після натискання кнопки "Вхід" система перевіряє введені дані і автентифікує користувача.

1.4. Якщо дані введено правильно, користувачу відкривається головне вікно програми. Якщо дані невірні, він отримує повідомлення про помилку і може спробувати знову.

2. Створення нового користувача:

2.1. Користувач, якщо не має облікового запису, може натиснути кнопку "Створити користувача" у вікні входу.

2.2. Він бачить вікно створення нового користувача, де має заповнити необхідні дані.

2.3. Користувач вводить пароль .

2.4. Після натискання кнопки "Підтвердити" система перевіряє введені дані.

2.5. Якщо дані введено коректно, обліковий запис користувача створюється, і користувач може використовувати його для входу до системи.

3. Використання головного вікна:

3.1. Після входу користувач бачить головне вікно програми з трьома блоками.

3.2. В першому блоку він бачить кнопки, що відображають типи паролів (наприклад, "Вхід", "Банк", "Особисті дані"). Користувач може натиснути на кнопку з вибраним типом, щоб переглянути відповідні збережені паролі.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		60

3.3.В другому блоку користувач бачить кнопки, на яких вказані імена збережених паролів. При натисканні на кнопку відповідного паролю відображається сам пароль.

3.4.У третьому блоку розташована кнопка "Створити новий пароль". Користувач може натиснути на неї, щоб створити новий пароль згідно зі своїми налаштуваннями.

3.5.При необхідності користувач може натиснути кнопку виходу або закрити програму, щоб завершити роботу з системою.

4. Створення нового паролю:

У третьому блоку розташована кнопка "Створити новий пароль". Користувач може натиснути на неї, щоб створити новий пароль згідно зі своїми налаштуваннями.

При натисканні кнопки "Створити новий пароль" користувач бачить форму реєстрації нового паролю з трьома блоками.

У першому блоку знаходиться список типів паролів, поле для введення імені, поле для введення паролю та кнопка "Перегенерувати". Користувач може обрати тип паролю зі списку, ввести ім'я і ввести або згенерувати пароль.

У другому блоку розміщені налаштування паролю, такі як довжина паролю та чекбокси для верхнього і нижнього регістру, цифр та символів. Користувач може налаштувати параметри паролю згідно своїх потреб.

У третьому блоку розташована кнопка "Зберегти". Користувач може натиснути на неї, щоб зберегти новий пароль.

Після натискання кнопки "Зберегти" система перевіряє введені дані, створює новий пароль з урахуванням налаштувань та зберігає його в системі.

Це загальний порядок роботи з системою локального менеджера паролів, який дозволяє користувачу входити до системи, створювати нових користувачів, переглядати та керувати збереженими паролями.

3.3. Тестування системи

Одна з основних функцій системи - генерація паролів, тому для тестування цієї функції потрібно виділити найбільше часу.

Тестування алгоритму генерації паролю включало наступні кроки:

1. Перевірка правильності генерації кількості чисел (`min_numb`), коли включена опція використання чисел (`is_number_active`).

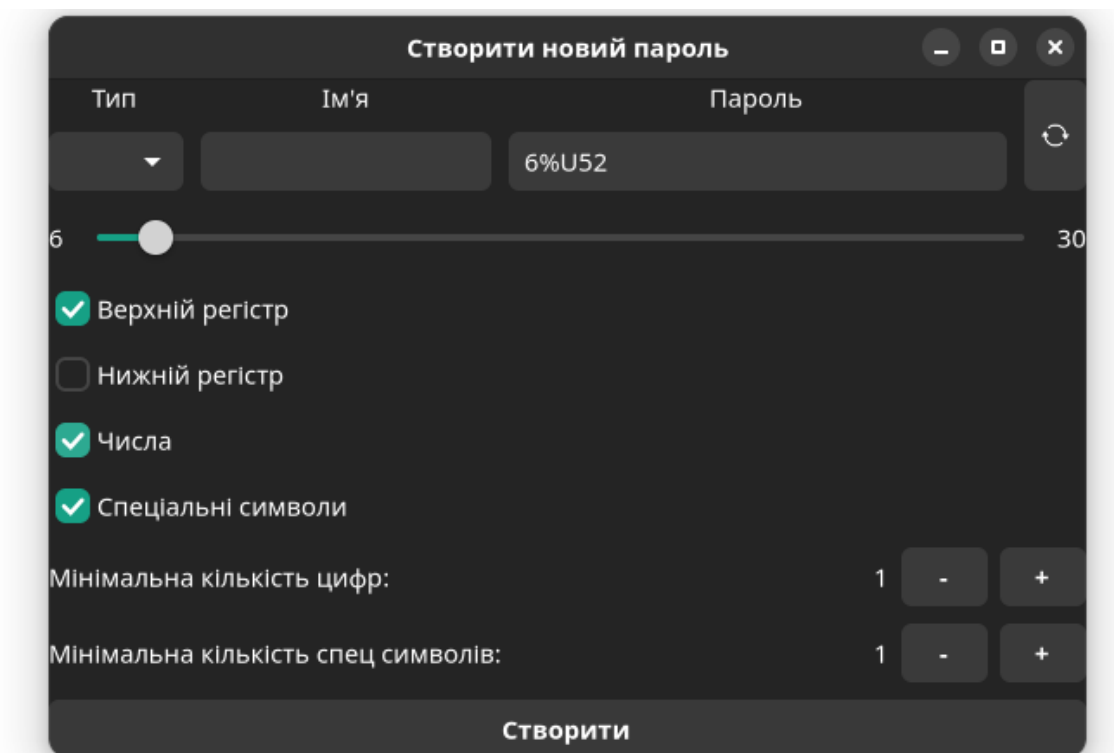


Рис. 3.6 - Тестування генерації паролю

2. Перевірка правильності генерації кількості спеціальних символів (`min_special`), коли включена опція використання спеціальних символів (`is_special_active`).

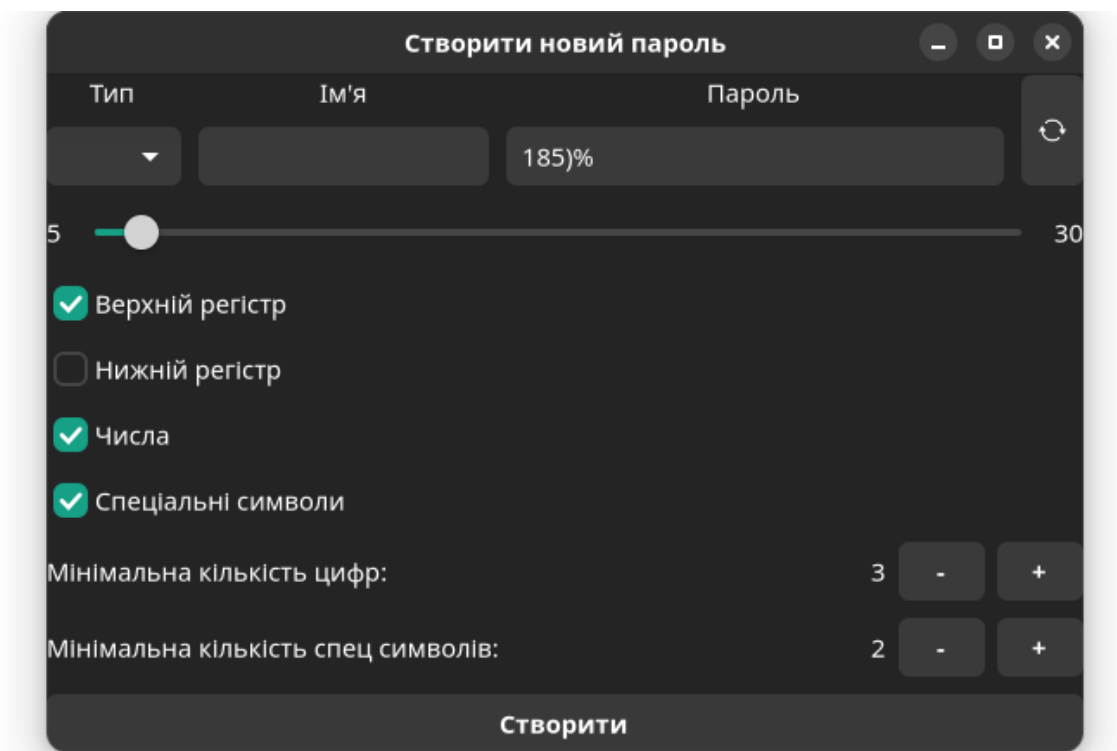


Рис. 3.7 - Тестування кількості символів

3. Перевірка правильної загальної довжини паролю (length) і відповідності вимогам щодо комбінації символів.
4. Тестування зміни довжини паролю:
5. Перевірка, як зміна значення налаштування довжини паролю (length_scale) впливає на відображення значення на етикетці (length_scale_label).
6. Перевірка генерації паролю з відповідною довжиною після зміни значення.
7. Тестування відображення та збереження згенерованого паролю:
8. Перевірка правильного відображення згенерованого паролю у відповідному полі введення (entries[2]).
9. Перевірка коректного збереження паролю та можливості його отримання з буфера обміну.
10. Тестування граничних значень:

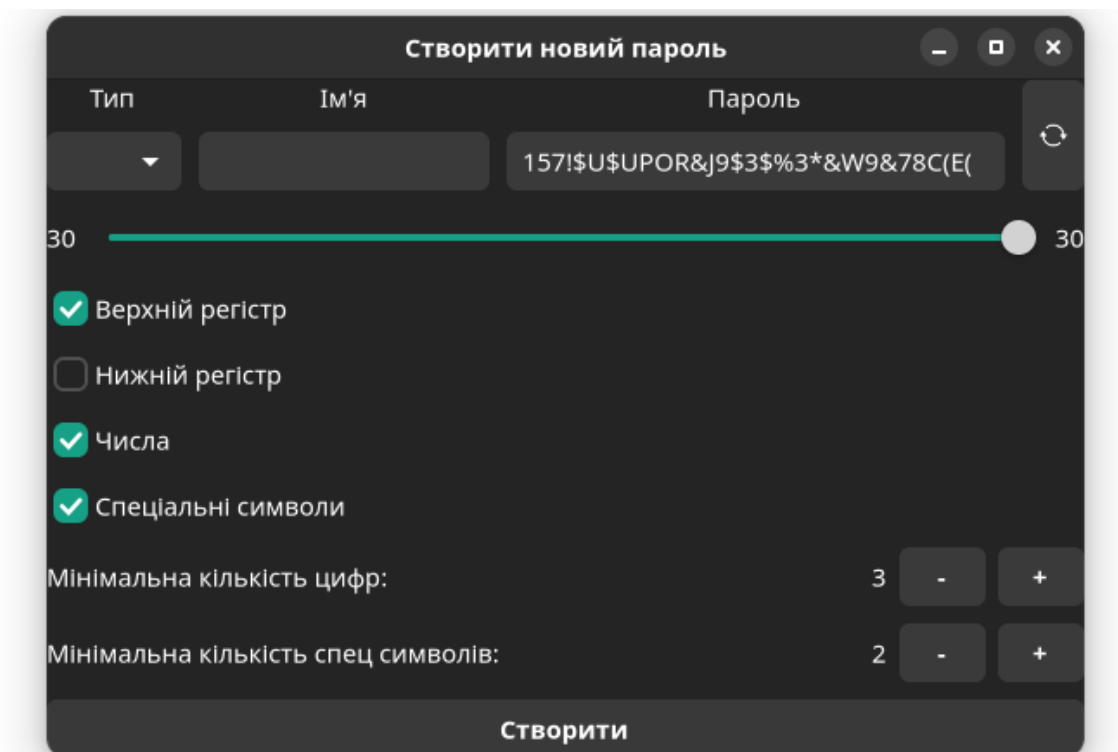


Рис. 3.8 - Тестування максимальних величин і правильності генерації

11. Перевірка поведінки коду при нульових або негативних значеннях min_numb, min_special, length та відповідних налаштуваннях активності.

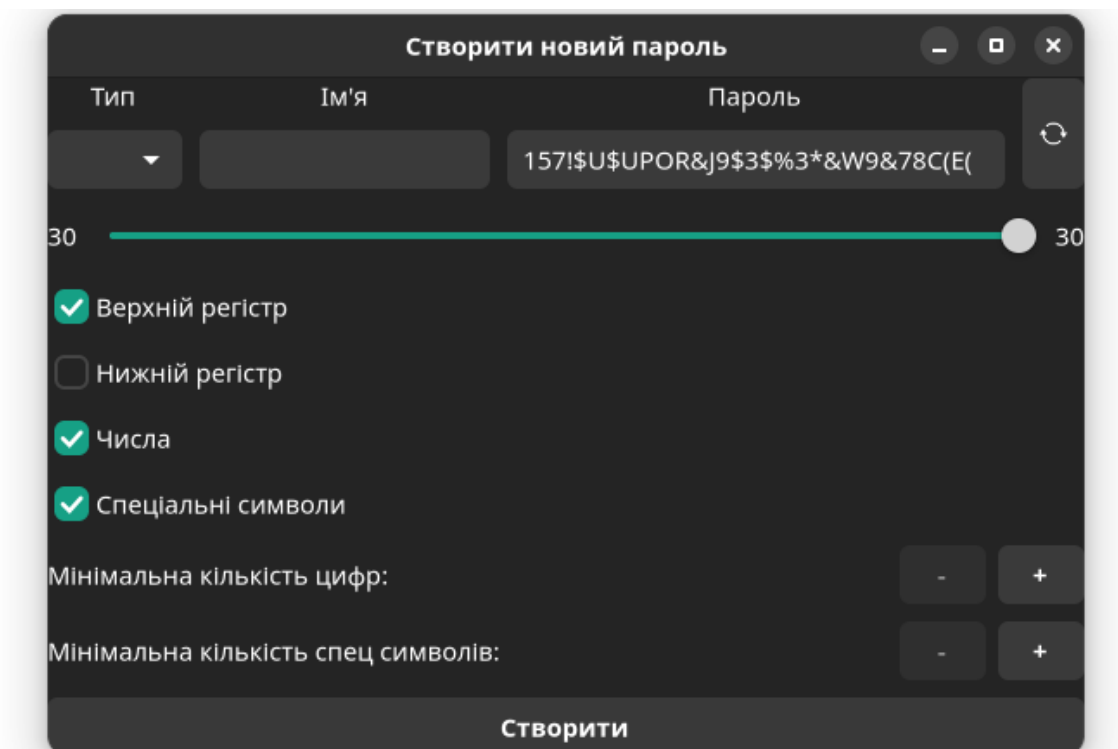


Рис. 3.9 - Показ блокування кнопок при зменшенні кількості символів, цифр

Перед початком тестування всієї програми було проведено тестування алгоритму генерації паролів і зміни налаштувань. Одна з найголовніших функцій системи - генерація паролів, і саме на цю функцію було виділено найбільше часу для тестування. Крім того, було перевірено правильність відображення та збереження згенерованого паролю, а також обробку граничних значень і зміну довжини паролю.

Після успішного тестування алгоритму генерації паролів, перейшли до тестування всієї програми.

1. Перевірка відображень різних імен і паролів в системі.

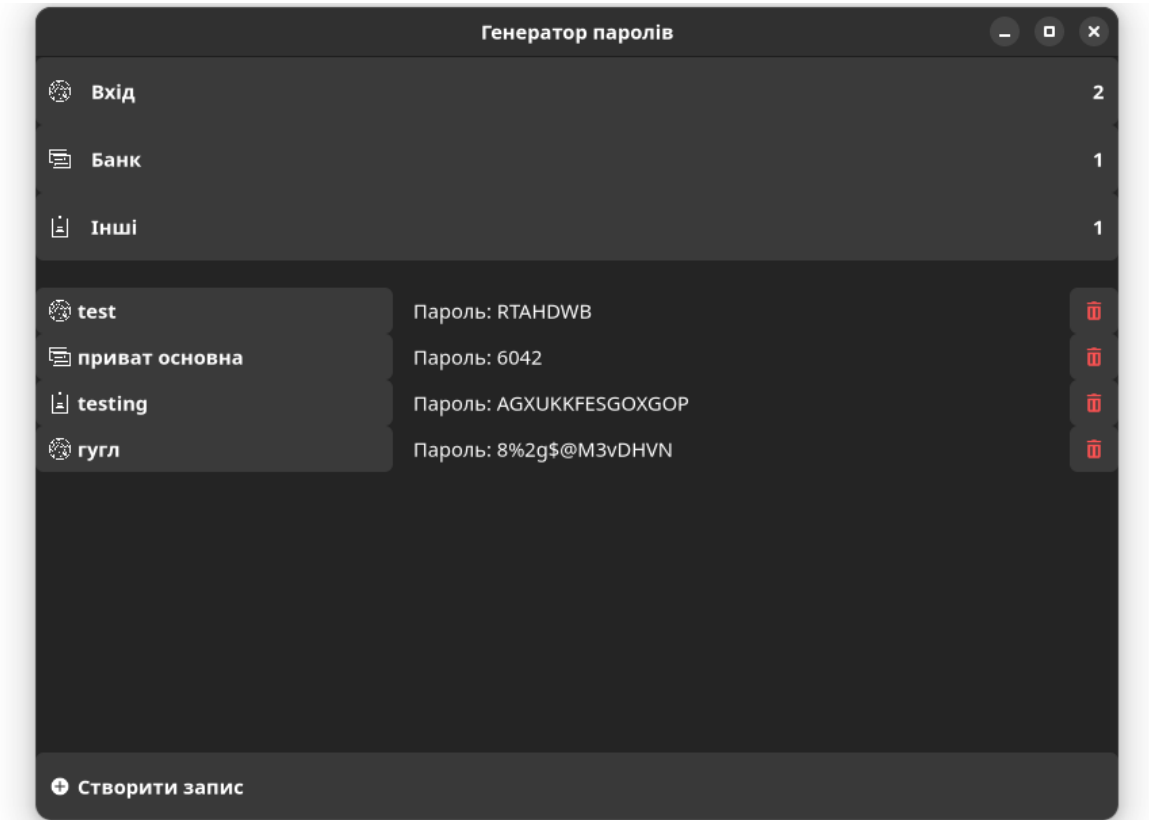


Рис. 3.10 - Перевірка відображень різних кількості символів в паролі а також роботу з різнимим мовами.

2. Тестування різних розмірів головного вікна

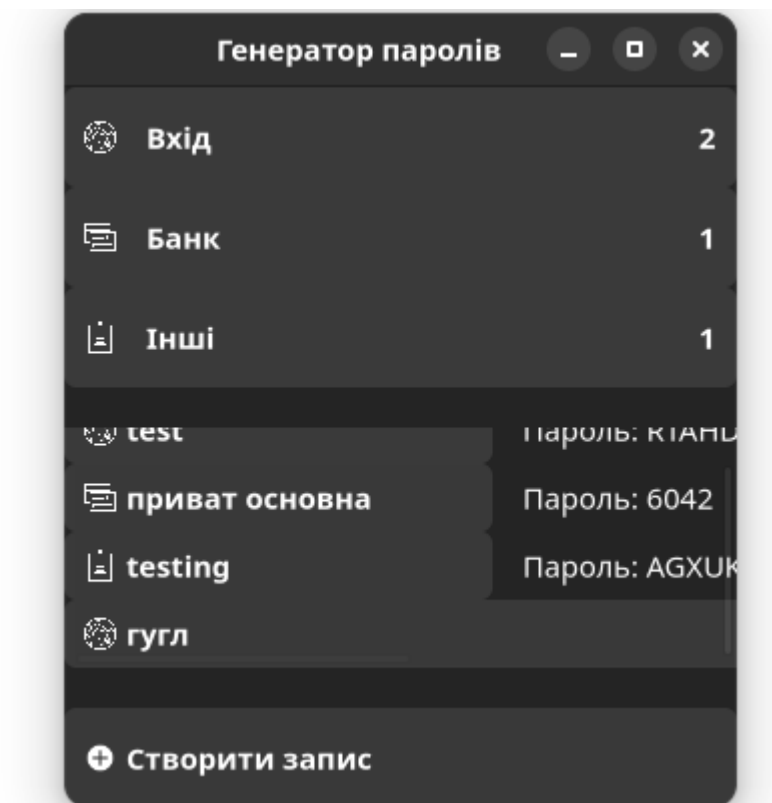


Рис. 3.11 - Показ мінімальних розмірів головного вікна

3. Тестування повноекранного режиму.

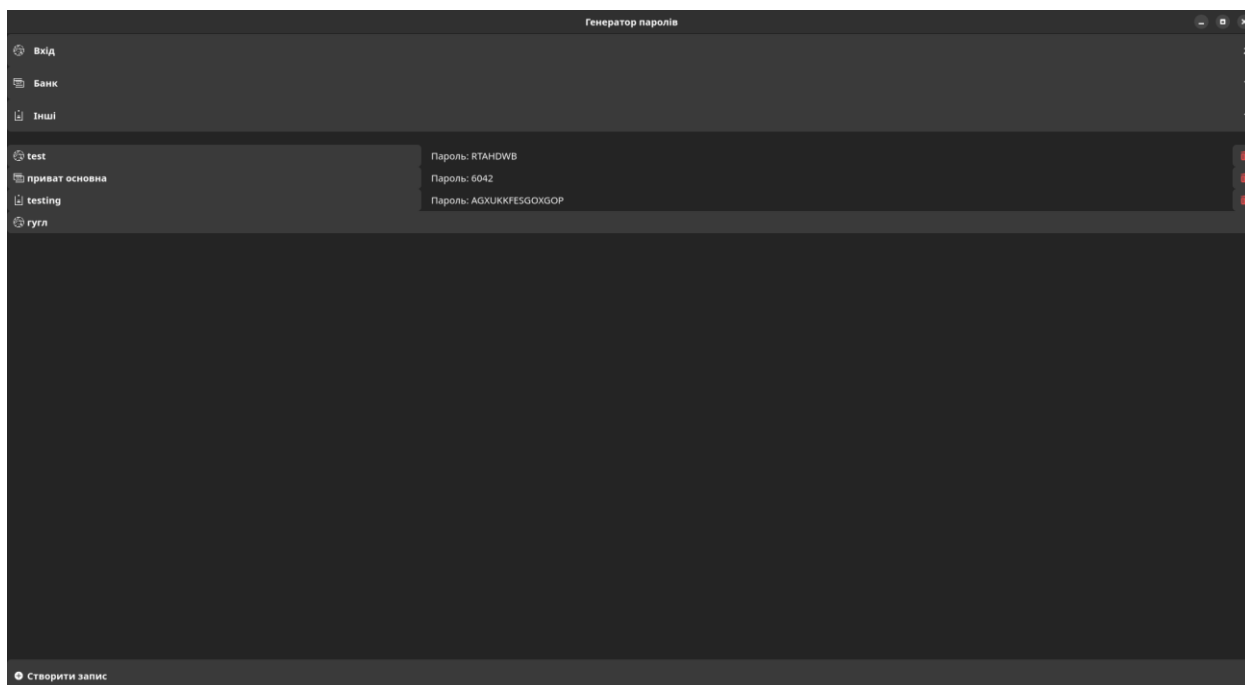


Рис. 3.12 - Показ максимальних розмірів головного вікна

4. Тестування розмірів вікна для генерації паролів

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						66
Змн.	Арк.	№ докум.	Підпис	Дата		

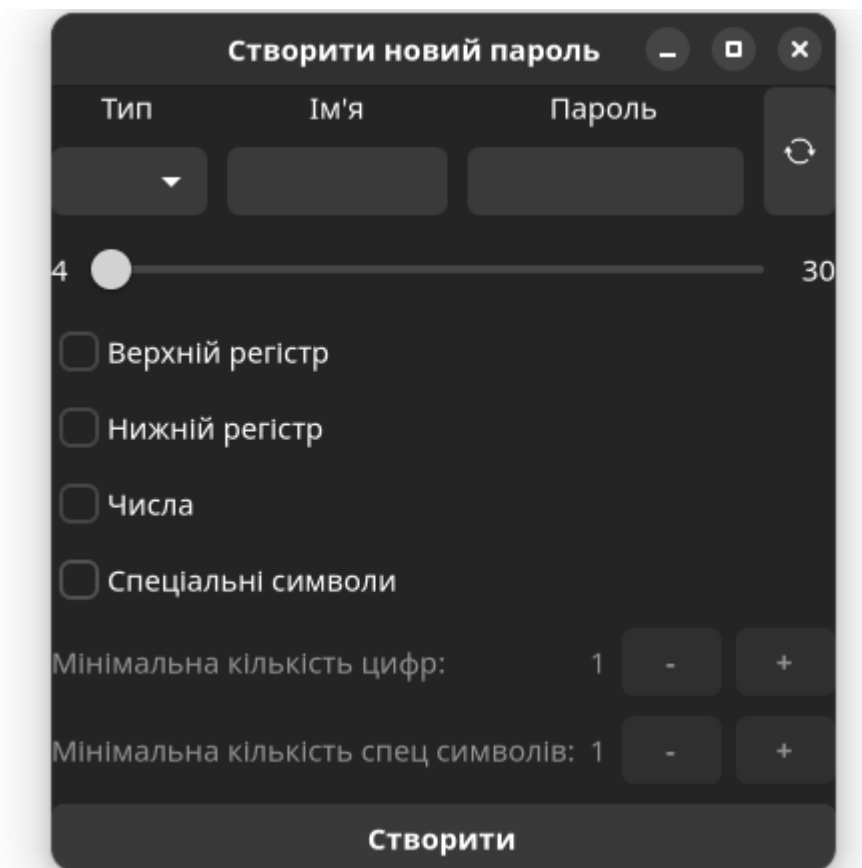


Рис. 3.13 - Показ мінімальних розмірів вікна для генерації паролів

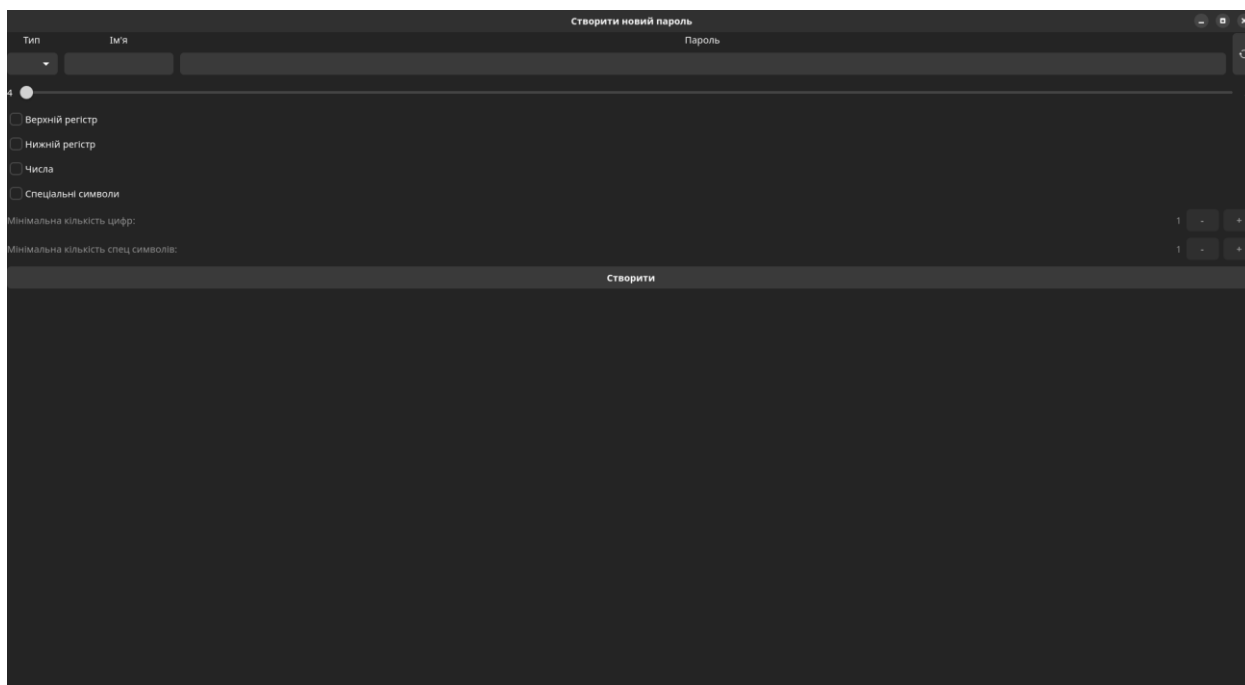


Рис. 3.14 - Показ максимальних розмірів вікна для генерації паролів

У цьому тестуванні було перевірено відображення різних імен і паролів в системі. Також була проведена перевірка роботи програми з різними розмірами головного вікна, включаючи мінімальні та максимальні розміри.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						67
Змн.	Арк.	№ докум.	Підпис	Дата		

Крім того, було протестовано режим повноекранного вікна та розміри вікна для генерації паролів.

Ці тести були проведені з метою забезпечення коректної роботи програми в різних умовах та переконання в правильності її функціональності.

Висновки до третього розділу

У розділі було представлено детальне описання порядку встановлення та налаштування параметрів системи, структури інтерфейсу та порядку роботи з системою, а також результати тестування системи.

Було розглянуто процес встановлення та налаштування параметрів системи, що дозволяє користувачам налаштовувати різні параметри генерації паролів. Цей процес було описано докладно з вказівкою кроків для зручного встановлення параметрів.

Також було розглянуто структуру інтерфейсу системи та порядок роботи з нею. Користувачам було пояснено, як використовувати систему для генерації паролів, відображення та збереження згенерованих паролів. Детально описано процеси зміни довжини паролю, роботу з різними розмірами головного вікна та використання повноекранного режиму.

Окрім того, проведено тестування системи з метою перевірки її функціональності та коректності роботи. Були проведені тести на генерацію правильної кількості символів, обробку граничних значень, відображення різних імен і паролів в системі, а також роботу з різними розмірами вікна. Результати тестування свідчать про правильну роботу системи та виконання вимог щодо генерації паролів та їх відображення.

В цілому, розділ надає чітку і зрозумілу інформацію про інтерфейс та порядок роботи з системою. Користувачам надається необхідна інструкція для ефективного встановлення параметрів, використання функцій системи та тестування її роботи. Розділ 3 допомагає користувачам зрозуміти і працювати з системою локального менеджера паролів з комфортом та надійністю.

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		68

ВИСНОВКИ

У даній роботі було проведено аналіз напрямків використання інформаційних технологій для реалізації локального сховища паролів. Розділ 1 розпочався з опису проекту та постановки задачі, де були визначені основні цілі та вимоги до системи. Далі були проаналізовані існуючі програмні продукти в цій області, що дозволило визначити їх переваги та недоліки. Згодом було обгрунтовано вибір технологій для розробки системи локального менеджера паролів.

Розділ 2 присвячений проектуванню локального менеджера паролів. Було проведено визначення варіантів використання та структури системи, що дало змогу визначити основні компоненти та функції системи. Далі були розроблені та реалізовані алгоритми роботи системи, включаючи генерацію та збереження паролів. На заключному етапі розділу було розроблено додаток локального менеджера паролів.

У розділі 3 було описано інтерфейс та порядок роботи з локальним менеджером паролів. Було пояснено порядок встановлення та налаштування параметрів системи, включаючи вибір параметрів генерації паролів. Також була описана структура інтерфейсу та пояснено порядок роботи з системою, включаючи генерацію, відображення та збереження паролів. На завершення розділу було проведено тестування системи з метою перевірки її функціональності та коректності роботи.

В результаті дослідження було розроблено і реалізовано локальний менеджер паролів, який забезпечує зручне та безпечне зберігання та генерацію паролів. Використання такої системи дозволяє користувачам ефективно керувати своїми паролями та забезпечити високий рівень безпеки їх застосування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. GTK4 - Використана офіційна документація GTK4. Доступно на: <https://docs.gtk.org/>
2. OpenSSL - Використана офіційна документація OpenSSL. Доступно на: <https://www.openssl.org/docs/>
3. Arch Linux - Використані документація та ресурси спільноти Arch Linux для отримання інформації про конфігурацію та встановлення пакетів. Доступно на: <https://wiki.archlinux.org/>
4. AES (Advanced Encryption Standard) - Використана офіційна документація Національного інституту стандартів і технологій (NIST) США щодо AES. Доступно на: <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf>
5. GNU Privacy Guard (GPG) - Офіційна документація GNU Privacy Guard, включаючи довідкові матеріали та документацію з командного рядка. Доступно на: <https://gnupg.org/documentation/>
6. C Library - Офіційна документація для стандартної бібліотеки мови C. Доступно на: <https://www.gnu.org/software/libc/manual/>
7. The C Programming Language (Книга) - Книга "The C Programming Language" авторів Браяна Кернігана та Деніса Річі. ISBN: 978-0131103627
8. Secure Coding in C and C++ (Книга) - Книга "Secure Coding in C and C++" авторів Роберта Сі. Сіверса та Джеймса С. Куленанда. ISBN: 978-0-321-33572-4
9. Practical Cryptography for Developers (Книга) - Книга "Practical Cryptography for Developers" автора Сушіл Джаярай. ISBN: 978-1484248997

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						70
Змн.	Арк.	№ докум.	Підпис	Дата		

10. GNU Core Utilities - Офіційна документація для основних утиліт GNU (наприклад, ls, grep, rm тощо) в Unix-подібних операційних системах. Доступно на: <https://www.gnu.org/software/coreutils/>
11. Understanding Cryptography (Книга) - Книга "Understanding Cryptography" авторів Крістофера Паоула та Йона Шнайдера. ISBN: 978-3642041006
12. Linux Security - Розділ "Security" в документації Linux, який містить інформацію про різні аспекти безпеки в Linux. Доступно на: <https://www.kernel.org/doc/html/latest/security/index.html>
13. Practical Cryptography - Книга "Practical Cryptography" авторів Нільса Фергюсона та Брюса Шнайера. ISBN: 978-0471223573
14. The Linux Command Line - Книга "The Linux Command Line" автора Вільяма Шотта. ISBN: 978-1593273897
15. Version Control with Git - Книга "Version Control with Git" автора Джон Лоусона. ISBN: 978-1449345037

ДОДАТКИ

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		72

Програма

Файл main.c

```

#include <gtk/gtk.h>

#include "WindowContentFunk.h"

//ініціалізація
static void activate(GtkApplication* app, gpointer user_data)
{
    content_login_window(app);
}

int main(int argc, char *argv[])
{
    GtkApplication *app;
    int status;

    app = gtk_application_new ("org.gtk.example",
G_APPLICATION_DEFAULT_FLAGS);
    g_signal_connect (app, "activate", G_CALLBACK (activate), NULL);
    status = g_application_run (G_APPLICATION (app), argc, argv);
    g_object_unref (app);

    return status;
}

```

Файл WindowsContentFunk.h

```

#pragma once

#include "BoxContentFunk.h"

#include <gtk/gtk.h>

void content_login_window(GtkApplication* app);

void content_main_window(GtkApplication* app);

```

Файл WindowsContentFunk.c

```

#include "WindowContentFunk.h"

void content_main_window(GtkApplication* app)
{
    GtkWidget *window, *container;
    GtkWidget *header, *scroll, *content, *footer;

    window = gtk_application_window_new(app);

    gtk_window_set_title(GTK_WINDOW(window), "Генератор паролів");
    gtk_window_set_default_size(GTK_WINDOW(window), 800, 600);
    container = gtk_box_new(GTK_ORIENTATION_VERTICAL, 20);

    gtk_window_set_child(GTK_WINDOW(window), container);
}

```

```

//create header box and apend to container
header = gtk_box_new(GTK_ORIENTATION_VERTICAL, 0);
content_header_box(header);
gtk_box_append(GTK_BOX(container), header);

//create and append content box make him scroller
scroll = gtk_scrolled_window_new();
content = gtk_box_new(GTK_ORIENTATION_VERTICAL, 0);

//займає все доступне місце
gtk_widget_set_hexpand(scroll, TRUE);
gtk_widget_set_vexpand(scroll, TRUE);

content_center_box(content);
gtk_box_append(GTK_BOX(container), scroll);
gtk_scrolled_window_set_child(GTK_SCROLLED_WINDOW(scroll), content);

//create and append container footer
footer = gtk_box_new(GTK_ORIENTATION_VERTICAL, 5);
content_footer_box(footer);
gtk_box_append(GTK_BOX(container), footer);

gtk_window_present(GTK_WINDOW(window));
}

void content_login_window(GtkApplication* app)
{
    GtkWidget *window, *container;

    //create main window and set child
    window = gtk_application_window_new(app);
    gtk_window_set_title(GTK_WINDOW(window), "Вхід");
    gtk_window_set_default_size(GTK_WINDOW(window), 400, 200);

    container = gtk_box_new(GTK_ORIENTATION_VERTICAL, 20);

    content_login_container_box(container);

    gtk_window_set_child(GTK_WINDOW(window), container);

    // show window and run main cycle GTK
    gtk_window_present(GTK_WINDOW(window));
}

```

Файл BoxContentFunk.h

```

#pragma once

#include "ConectSignalFunk.h"

#include <gtk/gtk.h>

void content_login_container_box(GtkWidget *container);
void content_header_box(GtkWidget *header);
void content_center_box(GtkWidget *content);
void content_footer_box(GtkWidget *footer);
void content_add_pass_header_box(GtkWidget *header_box);
void content_add_pass_properties_pass_box(GtkWidget
*propertiespass_box);

```

		В.С. Билков			ІІЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						74
Змн.	Арк.	№ докум.	Підпис	Дата		

Файл BoxContentFunk.c

```
#include "BoxContentFunk.h"

GtkWidget *entries[3];
GtkWidget *container_box;
GtkWidget *counters[3];

void content_login_container_box(GtkWidget *container)
{
    GtkWidget *textlabel, *submitbutton, *passentry, *newuserbutton;

    textlabel = gtk_label_new("Пароль");

    passentry = gtk_entry_new();
    gtk_entry_set_visibility(GTK_ENTRY(passentry), FALSE);

    submitbutton = gtk_button_new_with_label("Вхід");
    g_signal_connect(submitbutton, "clicked",
G_CALLBACK(login_button_clicked), passentry);

    newuserbutton = gtk_button_new_with_label("Новий користувач");
    g_signal_connect(newuserbutton, "clicked",
G_CALLBACK(add_new_user_label_clicked), NULL);

    gtk_box_append(GTK_BOX(container), textlabel);
    gtk_box_append(GTK_BOX(container), passentry);
    gtk_box_append(GTK_BOX(container), submitbutton);
    gtk_box_append(GTK_BOX(container), newuserbutton);
}

void content_header_box(GtkWidget *header)
{
    GtkWidget *entry_button, *card_button, *user_data_button;
    GtkWidget *entry_box, *card_box, *user_data_box;
    GtkWidget *entry_label, *card_label, *user_data_label;
    GtkWidget *entry_counter, *card_counter, *user_data_counter;
    GtkWidget *entry_image, *card_image, *user_data_image;

    entry_button = gtk_button_new();
    card_button = gtk_button_new();
    user_data_button = gtk_button_new();

    gtk_widget_set_size_request(entry_button, -1, 50);
    gtk_widget_set_size_request(card_button, -1, 50);
    gtk_widget_set_size_request(user_data_button, -1, 50);

    entry_box = gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 15);
    card_box = gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 15);
    user_data_box = gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 15);

    entry_label = gtk_label_new("Вхід");
    card_label = gtk_label_new("Банк");
    user_data_label = gtk_label_new("Інші");
    //fill all and position label in the button
    gtk_widget_set_hexpand(entry_label, TRUE);
    gtk_widget_set_hexpand(card_label, TRUE);
    gtk_widget_set_hexpand(user_data_label, TRUE);
    gtk_widget_set_halign(entry_label, GTK_ALIGN_START);
    gtk_widget_set_halign(card_label, GTK_ALIGN_START);
```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						75
Змн.	Арк.	№ докум.	Підпис	Дата		

```

gtk_widget_set_halign(user_data_label, GTK_ALIGN_START);

entry_counter = gtk_label_new("0");
card_counter = gtk_label_new("0");
user_data_counter = gtk_label_new("0");

counters[0] = entry_counter;
counters[1] = card_counter;
counters[2] = user_data_counter;

entry_image = gtk_image_new_from_file("img/global.png");
card_image = gtk_image_new_from_file("img/credit-card.png");
user_data_image = gtk_image_new_from_file("img/card.png");

gtk_box_append(GTK_BOX(entry_box), entry_image);
gtk_box_append(GTK_BOX(entry_box), entry_label);
gtk_box_append(GTK_BOX(entry_box), entry_counter);

gtk_box_append(GTK_BOX(card_box), card_image);
gtk_box_append(GTK_BOX(card_box), card_label);
gtk_box_append(GTK_BOX(card_box), card_counter);

gtk_box_append(GTK_BOX(user_data_box), user_data_image);
gtk_box_append(GTK_BOX(user_data_box), user_data_label);
gtk_box_append(GTK_BOX(user_data_box), user_data_counter);

gtk_button_set_child(GTK_BUTTON(entry_button), entry_box);
gtk_button_set_child(GTK_BUTTON(card_button), card_box);
gtk_button_set_child(GTK_BUTTON(user_data_button), user_data_box);

gtk_box_append(GTK_BOX(header), entry_button);
gtk_box_append(GTK_BOX(header), card_button);
gtk_box_append(GTK_BOX(header), user_data_button);

}

void content_center_box(GtkWidget *content)
{
    FILE* file = fopen("bin/encrypted_passwords.bin", "rb");
    if (file != NULL)
    {
        int num_passwords = 0;

        while (1) {
            size_t type_len, name_len, pass_len;

            if (fread(&type_len, sizeof(size_t), 1, file) != 1 ||
                fread(&name_len, sizeof(size_t), 1, file) != 1 ||
                fread(&pass_len, sizeof(size_t), 1, file) != 1) {
                break; // Помилка при зчитуванні довжини блоків
            }

            if (type_len == 0 || name_len == 0 || pass_len == 0) {
                break; // Недопустима довжина блоку
            }

            // Пропустити дані блоку
            fseek(file, type_len + name_len + pass_len, SEEK_CUR);

            num_passwords++;
        }
    }
}

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						76
Змн.	Арк.	№ докум.	Підпис	Дата		

```

// Використання отриманої кількості записів
printf("Кількість записів: %d\n", num_passwords);

// Повернутися до початку файлу
rewind(file);

//size_t max_text_length = AES_BLOCK_SIZE; // Максимальна довжина
тексту
char** encrypted_type = malloc(num_passwords * sizeof(char*));
char** encrypted_name = malloc(num_passwords * sizeof(char*));
char** encrypted_password = malloc(num_passwords *
sizeof(char*));

for (int i = 0; i < num_passwords; i++) {
    size_t type_len, name_len, pass_len;

    fread(&type_len, sizeof(size_t), 1, file);
    fread(&name_len, sizeof(size_t), 1, file);
    fread(&pass_len, sizeof(size_t), 1, file);

    encrypted_type[i] = malloc((type_len + 1) * sizeof(char));
    encrypted_name[i] = malloc((name_len + 1) * sizeof(char));
    encrypted_password[i] = malloc((pass_len + 1) *
sizeof(char));

    fread(encrypted_type[i], sizeof(char), type_len, file);
    fread(encrypted_name[i], sizeof(char), name_len, file);
    fread(encrypted_password[i], sizeof(char), pass_len, file);

    encrypted_type[i][type_len] = '\0'; // Додати завершуючий
символ '\0'
    encrypted_name[i][name_len] = '\0';
    encrypted_password[i][pass_len] = '\0';
}
fclose(file);

int counter_type_login = 0;
int counter_type_bank = 0;
int counter_type_another = 0;

// Додавання кнопок для кожного елементу
for (int i = 0; i < num_passwords; i++)
{
    // Створення горизонтального контейнера для кожного елементу
    GtkWidget *item_box =
gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 5);
    gtk_box_set_homogeneous(GTK_BOX(item_box), TRUE);
    gtk_box_append(GTK_BOX(content), item_box);

    // Створення кнопки з іменем
    GtkWidget *button = gtk_button_new();
    GtkWidget *box = gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 5);

    // Визначення типу та вибір відповідної картинки
    GtkWidget *image = NULL;

    if (strcmp(encrypted_type[i], "Вхід") == 0) {
        image = gtk_image_new_from_file("img/global.png");
        counter_type_login++;
    } else if (strcmp(encrypted_type[i], "Банк") == 0) {

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						77
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        image = gtk_image_new_from_file("img/credit-card.png");
        counter_type_bank++;
    } else if(strcmp(encrypted_type[i], "Інші") == 0) {
        image = gtk_image_new_from_file("img/card.png");
        counter_type_another++;
    }

    gtk_box_append(GTK_BOX(box), image);

    // Створення етикетки з іменем
    GtkWidget *name_label = gtk_label_new(encrypted_name[i]);
    gtk_box_append(GTK_BOX(box), name_label);

    // Додавання вертикального контейнера до кнопки
    gtk_button_set_child(GTK_BUTTON(button), box);

    g_signal_connect(button, "clicked",
G_CALLBACK(button_item_clicked), item_box);
    gtk_box_append(GTK_BOX(item_box), button);

    // Створення етикетки для паролю (початково прихована)

    char password_text[256];
    snprintf(password_text, sizeof(password_text), "Пароль: %s",
encrypted_password[i]);

    GtkWidget *password_label = gtk_label_new(password_text);
    gtk_widget_set_visible(password_label, FALSE);
    gtk_widget_set_hexpand(password_label, TRUE);
    gtk_label_set_xalign(GTK_LABEL(password_label), 0.0);
    gtk_widget_set_margin_start(password_label, 10);
    gtk_box_append(GTK_BOX(item_box), password_label);

    // Створення кнопки видалення
    GtkWidget *delete_button = gtk_button_new();
    GtkWidget *delete_image =
    gtk_image_new_from_file("img/trash.png");
    gtk_widget_set_hexpand(delete_button, FALSE);
    gtk_widget_set_halign(delete_button, GTK_ALIGN_END);
    gtk_button_set_child(GTK_BUTTON(delete_button),
delete_image);

    // Визначення видимості кнопки видалення
    gtk_widget_set_visible(delete_button, FALSE);

    // Додавання кнопки видалення до контейнера
    gtk_box_append(GTK_BOX(item_box), delete_button);

    // З'єднання обробника події для кнопки видалення
    //g_signal_connect(delete_button, "clicked",
G_CALLBACK(delete_button_clicked), item_box);
    }

    char text[20];
    sprintf(text, "%d", counter_type_login);
    gtk_label_set_text(GTK_LABEL(counters[0]), text);

    sprintf(text, "%d", counter_type_bank);
    gtk_label_set_text(GTK_LABEL(counters[1]), text);

    sprintf(text, "%d", counter_type_another);
    gtk_label_set_text(GTK_LABEL(counters[2]), text);

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						78
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        for (int i = 0; i < num_passwords; i++) {
            free(encrypted_type[i]);
            free(encrypted_name[i]);
            free(encrypted_password[i]);
        }
        free(encrypted_type);
        free(encrypted_name);
        free(encrypted_password);
    }
    else
    {
        printf("Помилка відкриття файлу.\n");
    }
    container_box = content;

    /*char decrypted_type[AES_BLOCK_SIZE];
    char decrypted_name[AES_BLOCK_SIZE];
    char decrypted_password[AES_BLOCK_SIZE];

    if (file != NULL)
    {
        fread(encrypted_type, sizeof(char), AES_BLOCK_SIZE, file);
        fread(encrypted_name, sizeof(char), AES_BLOCK_SIZE, file);
        fread(encrypted_password, sizeof(char), AES_BLOCK_SIZE, file);
        fclose(file);

        g_print("File text-----\n");
        g_print("Type: %s\n", encrypted_type);
        g_print("Name: %s\n", encrypted_name);
        g_print("Password: %s\n", encrypted_password);

        const char *key = "my_secret_key";

        aes_decrypt(encrypted_type, key, decrypted_type);
        aes_decrypt(encrypted_name, key, decrypted_name);
        aes_decrypt(encrypted_password, key, decrypted_password);

        g_print("Decriped text-----\n");
        g_print("Decrypted Type: %s\n", decrypted_type);
        g_print("Decrypted Name: %s\n", decrypted_name);
        g_print("Decrypted Password: %s\n", decrypted_password);
    }*/
}

void content_footer_box(GtkWidget *footer)
{
    GtkWidget *add_button, *add_box, *add_label, *add_image;
    add_button = gtk_button_new();
    g_signal_connect(add_button, "clicked",
G_CALLBACK(add_button_clicked), NULL);
    gtk_widget_set_size_request(add_button, -1, 50);

    add_box = gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 5);
    add_label = gtk_label_new("Створити запис");
    add_image = gtk_image_new_from_file("img/plus.png");

    gtk_box_append(GTK_BOX(add_box), add_image);
    gtk_box_append(GTK_BOX(add_box), add_label);
    gtk_button_set_child(GTK_BUTTON(add_button), add_box);
    gtk_box_append(GTK_BOX(footer), add_button);

```

```

}

void content_add_pass_header_box(GtkWidget *header_box)
{
    GtkWidget *pass_box, *type_box, *name_box;
    GtkWidget *pass_label, *type_label, *name_label;
    GtkWidget *pass_entry, *name_entry;
    GtkWidget *generate_button;
    GtkWidget *type_combo_box;

    // Створюємо блоки для пароля, типу та імені
    pass_box = gtk_box_new(GTK_ORIENTATION_VERTICAL, 10);
    type_box = gtk_box_new(GTK_ORIENTATION_VERTICAL, 10);
    name_box = gtk_box_new(GTK_ORIENTATION_VERTICAL, 10);

    // Створюємо мітки для полів пароля, типу та імені
    type_label = gtk_label_new("Тип");
    name_label = gtk_label_new("Ім'я");
    pass_label = gtk_label_new("Пароль");

    // Створюємо поля вводу для пароля, типу та імені
    //type_entry = gtk_entry_new();
    //entries[0] = type_entry;
    name_entry = gtk_entry_new();
    entries[1] = name_entry;
    pass_entry = gtk_entry_new();
    gtk_widget_set_hexpand(pass_entry, TRUE);
    entries[2] = pass_entry;

    // Створення моделі для випадаючого списку
    GtkListStore *list_store = gtk_list_store_new(1, G_TYPE_STRING);

    // Додавання елементів до моделі
    GtkTreeIter iter;
    gtk_list_store_append(list_store, &iter);
    gtk_list_store_set(list_store, &iter, 0, "Вхід", -1);
    gtk_list_store_append(list_store, &iter);
    gtk_list_store_set(list_store, &iter, 0, "Банк", -1);
    gtk_list_store_append(list_store, &iter);
    gtk_list_store_set(list_store, &iter, 0, "Інші", -1);

    // Створення випадаючого списку
    type_combo_box =
    gtk_combo_box_new_with_model(GTK_TREE_MODEL(list_store));
    GtkCellRenderer *renderer = gtk_cell_renderer_text_new();
    gtk_cell_layout_pack_start(GTK_CELL_LAYOUT(type_combo_box),
    renderer, TRUE);
    gtk_cell_layout_set_attributes(GTK_CELL_LAYOUT(type_combo_box),
    renderer, "text", 0, NULL);
    entries[0] = type_combo_box;

    // Створюємо кнопку "Генерувати"
    generate_button = gtk_button_new();
    GtkWidget *image = gtk_image_new_from_file("img/refresh.png");

    // Встановлюємо картинку як дочірній віджет для кнопки
    gtk_button_set_child(GTK_BUTTON(generate_button), image);

    g_signal_connect(generate_button, "clicked",
    G_CALLBACK(generate_button_clicked), pass_entry);

    // Додаємо мітки та поля вводу в блоки

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						80
Змн.	Арк.	№ докум.	Підпис	Дата		

```

gtk_box_append(GTK_BOX(type_box), type_label);
gtk_box_append(GTK_BOX(type_box), type_combo_box);

gtk_box_append(GTK_BOX(name_box), name_label);
gtk_box_append(GTK_BOX(name_box), name_entry);

gtk_box_append(GTK_BOX(pass_box), pass_label);
gtk_box_append(GTK_BOX(pass_box), pass_entry);

// Додаємо блоки до заголовкового блоку
gtk_box_append(GTK_BOX(header_box), type_box);
gtk_box_append(GTK_BOX(header_box), name_box);
gtk_box_append(GTK_BOX(header_box), pass_box);
gtk_box_append(GTK_BOX(header_box), generate_button);
}

// Функція для додавання блоку властивостей пароля
void content_add_pass_properties_pass_box(GtkWidget *propertiespass_box)
{
    GtkWidget *scale_length_box, *min_number_box, *min_special_box;
    GtkWidget *length_scale_label, *max_length_scale_label,
    *name_min_label, *name_special_label, *counter_min_number_label,
    *counter_special_label;
    GtkWidget *scale_length_pass;
    GtkWidget *uppercase_checkbutton, *lowercase_checkbutton,
    *number_checkbutton, *specialchar_checkbutton;
    GtkWidget *plus_min_number_button, *minus_min_number_button,
    *plus_special_button, *minus_special_button;

    // Створюємо блок для шкали довжини пароля
    scale_length_box = gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 7);
    min_number_box = gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 7);
    min_special_box = gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 7);
    gtk_widget_set_sensitive(min_number_box, FALSE);
    gtk_widget_set_sensitive(min_special_box, FALSE);

    // Створюємо мітки для мінімальної та максимальної довжини пароля
    length_scale_label = gtk_label_new("4");
    max_length_scale_label = gtk_label_new("30");

    name_min_label = gtk_label_new("Мінімальна кількість цифр:");
    name_special_label = gtk_label_new("Мінімальна кількість спец
символів:");
    counter_min_number_label = gtk_label_new("1");
    gtk_widget_set_hexpand(counter_min_number_label, TRUE);
    gtk_label_set_xalign(GTK_LABEL(counter_min_number_label), 1.0);
    counter_special_label = gtk_label_new("1");
    gtk_widget_set_hexpand(counter_special_label, TRUE);
    gtk_label_set_xalign(GTK_LABEL(counter_special_label), 1.0);

    // Створюємо шкалу для вибору довжини пароля
    scale_length_pass =
    gtk_scale_new_with_range(GTK_ORIENTATION_HORIZONTAL, 4, 30, 1);
    gtk_widget_set_hexpand(scale_length_pass, TRUE);
    g_signal_connect(scale_length_pass, "value-changed",
    G_CALLBACK(on_scale_value_changed), length_scale_label);

    // Створюємо прапорці для вибору властивостей пароля
    uppercase_checkbutton = gtk_check_button_new_with_label("Верхній
перістр");
    g_signal_connect(uppercase_checkbutton, "toggled",
    G_CALLBACK(on_upper_checkbutton_press_event), NULL);

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						81
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        lowercase_checkbutton      =      gtk_check_button_new_with_label("Нижній
pericтp");
        g_signal_connect(lowercase_checkbutton,                        "toggled",
G_CALLBACK(on_lower_checkbutton_press_event), NULL);
        number_checkbutton = gtk_check_button_new_with_label("Числа");
        g_signal_connect(number_checkbutton,                        "toggled",
G_CALLBACK(on_number_checkbutton_press_event), min_number_box);
        specialchar_checkbutton
=
gtk_check_button_new_with_label("Спеціальні символи");
        g_signal_connect(specialchar_checkbutton,                        "toggled",
G_CALLBACK(on_special_checkbutton_press_event), min_special_box);

        plus_min_number_button = gtk_button_new_with_label("+");
        minus_min_number_button = gtk_button_new_with_label("-");
        plus_special_button = gtk_button_new_with_label("+");
        minus_special_button = gtk_button_new_with_label("-");

        g_signal_connect(plus_min_number_button,                        "clicked",
G_CALLBACK(on_plus_min_number_button_clicked), counter_min_number_label);
        g_signal_connect(minus_min_number_button,                        "clicked",
G_CALLBACK(on_minus_min_number_button_clicked), counter_min_number_label);
        g_signal_connect(plus_special_button,                        "clicked",
G_CALLBACK(on_plus_special_button_clicked), counter_special_label);
        g_signal_connect(minus_special_button,                        "clicked",
G_CALLBACK(on_minus_special_button_clicked), counter_special_label);

        // Додаємо мітки, шкалу та прапорці до блоку властивостей пароля
        gtk_box_append(GTK_BOX(scale_length_box), length_scale_label);
        gtk_box_append(GTK_BOX(scale_length_box), scale_length_pass);
        gtk_box_append(GTK_BOX(scale_length_box), max_length_scale_label);

        gtk_box_append(GTK_BOX(min_number_box), name_min_label);
        gtk_box_append(GTK_BOX(min_number_box), counter_min_number_label);
        gtk_box_append(GTK_BOX(min_number_box), minus_min_number_button);
        gtk_box_append(GTK_BOX(min_number_box), plus_min_number_button);
        gtk_box_append(GTK_BOX(min_special_box), name_special_label);
        gtk_box_append(GTK_BOX(min_special_box), counter_special_label);
        gtk_box_append(GTK_BOX(min_special_box), minus_special_button);
        gtk_box_append(GTK_BOX(min_special_box), plus_special_button);

        gtk_box_append(GTK_BOX(propertiespass_box), scale_length_box);
        gtk_box_append(GTK_BOX(propertiespass_box), uppercase_checkbutton);
        gtk_box_append(GTK_BOX(propertiespass_box), lowercase_checkbutton);
        gtk_box_append(GTK_BOX(propertiespass_box), number_checkbutton);
        gtk_box_append(GTK_BOX(propertiespass_box),
specialchar_checkbutton);
        gtk_box_append(GTK_BOX(propertiespass_box), min_number_box);
        gtk_box_append(GTK_BOX(propertiespass_box), min_special_box);
    }

```

Файл ConectSignalFunk.h

```

#pragma once

#include <gtk/gtk.h>
#include <openssl/aes.h>
#include <stdio.h>
#include "WindowContentFunk.h"

void login_button_clicked(GtkButton *button, GtkWidget *passentry);

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		82

```

void add_button_clicked(GtkButton *button, gpointer user_data);

gchar *generate_pass();
void on_scale_value_changed(GtkRange *range, GtkWidget
*length_scale_label);
gboolean on_number_checkbutton_press_event(GtkWidget *widget, GtkWidget
*box);
gboolean on_special_checkbutton_press_event(GtkWidget *widget, GtkWidget
*box);
gboolean on_lower_checkbutton_press_event(GtkWidget *widget, gpointer
user_data);
gboolean on_upper_checkbutton_press_event(GtkWidget *widget, gpointer
user_data);
void on_plus_min_number_button_clicked(GtkButton *button, GtkWidget
*label);
void on_minus_min_number_button_clicked(GtkButton *button, GtkWidget
*label);
void on_plus_special_button_clicked(GtkButton *button, GtkWidget *label);
void on_minus_special_button_clicked(GtkButton *button, GtkWidget
*label);
void login_button_clicked(GtkButton *button, GtkWidget *passentry);
void generate_button_clicked(GtkButton *button, gpointer user_data);
void add_button_clicked(GtkButton *button, gpointer user_data);
void content_main_window(GtkApplication *app);

void add_new_user_label_clicked(GtkLabel *label, gpointer user_data);

void check_password_length(GtkWidget *button, GtkWidget *passentry);

void aes_encrypt(const char *input, const char *key, char *output);

void aes_decrypt(const char *input, const char *key, char *output);

void on_add_pass_button_clicked(GtkButton *button, gpointer user_data);

void button_item_clicked(GtkButton *button, gpointer user_data);

void password_visibility_changed(GtkLabel *label, GParamSpec *pspec,
gpointer user_data);

```

Файл ConbectSignalFunk.c

```

#include "ConectSignalFunk.h"

static gboolean is_number_active = FALSE;
static gboolean is_special_active = FALSE;
static gboolean is_upper_active = FALSE;
static gboolean is_lower_active = FALSE;

static int min_numb = 1;
static int min_special = 1;
static int length = 4;

extern GtkWidget *entries[3];
extern GtkWidget *container_box;
extern GtkWidget *counters[3];

gchar *generate_pass() {
    GString *password = g_string_new("");

    if (is_number_active) {

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		83

```

        for (int i = 0; i < min_numb; i++) {
            gchar digit = g_random_int_range('0', '9' + 1);
            g_string_append_c(password, digit);
        }

    if (is_special_active) {
        const gchar *special_chars = "!@#$%^&*()";
        int special_chars_count = strlen(special_chars);

        for (int i = 0; i < min_special; i++) {
            int random_index = g_random_int_range(0,
special_chars_count);
            gchar special_char = special_chars[random_index];
            g_string_append_c(password, special_char);
        }
    }

    int remaining_length = length - password->len;

    if (remaining_length > 0) {
        gchar *character_set = "";

        if (is_upper_active)
            character_set = g_strconcat(character_set,
"ABCDEFGHIJKLMNOPQRSTUVWXYZ", NULL);

        if (is_lower_active)
            character_set = g_strconcat(character_set,
"abcdefghijklmnopqrstuvwxyz", NULL);

        if (is_number_active)
            character_set = g_strconcat(character_set, "0123456789",
NULL);

        if (is_special_active)
            character_set = g_strconcat(character_set, "!@#$%^&*()",
NULL);

        int character_set_length = strlen(character_set);

        for (int i = 0; i < remaining_length; i++) {
            int random_index = g_random_int_range(0,
character_set_length);
            gchar random_char = character_set[random_index];
            g_string_append_c(password, random_char);
        }

        g_free(character_set);
    }

    gchar *result = g_string_free(password, FALSE);
    return result;
}

void on_scale_value_changed(GtkRange *range, GtkWidget
*length_scale_label)
{
    double value = gtk_range_get_value(range);
    char buffer[10];
    sprintf(buffer, "%.0f", value);
    gtk_label_set_text(GTK_LABEL(length_scale_label), buffer);
}

```

```

length = value;

const gchar *text = generate_pass();

GtkEntryBuffer *buffer_pass =
gtk_entry_get_buffer(GTK_ENTRY(entries[2]));
gtk_entry_buffer_set_text(buffer_pass, text, -1);
}

gboolean on_number_checkbutton_press_event(GtkWidget *widget, GtkWidget
*box)
{
    gboolean is_active =
gtk_check_button_get_active(GTK_CHECK_BUTTON(widget));
    if(is_active == FALSE)
    {
        gtk_widget_set_sensitive(box, FALSE); // Встановлюємо контейнер
як неактивний
    }
    else
    {
        gtk_widget_set_sensitive(box, TRUE); // Встановлюємо контейнер як
активний
    }

    is_number_active = is_active;

    const gchar *text = generate_pass();

    GtkEntryBuffer *buffer =
gtk_entry_get_buffer(GTK_ENTRY(entries[2]));
    gtk_entry_buffer_set_text(buffer, text, -1);

    return TRUE; // Повертаємо TRUE, щоб зупинити подальшу обробку події
}

gboolean on_special_checkbutton_press_event(GtkWidget *widget, GtkWidget
*box)
{
    gboolean is_active =
gtk_check_button_get_active(GTK_CHECK_BUTTON(widget));
    if(is_active == FALSE)
    {
        gtk_widget_set_sensitive(box, FALSE); // Встановлюємо контейнер
як неактивний
    }
    else
    {
        gtk_widget_set_sensitive(box, TRUE); // Встановлюємо контейнер як
активний
    }

    is_special_active = is_active;

    const gchar *text = generate_pass();

    GtkEntryBuffer *buffer =
gtk_entry_get_buffer(GTK_ENTRY(entries[2]));
    gtk_entry_buffer_set_text(buffer, text, -1);

    return TRUE; // Повертаємо TRUE, щоб зупинити подальшу обробку події
}

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						85
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        gboolean on_lower_checkbutton_press_event(GtkWidget *widget, gpointer
user_data)
        {
            is_lower_active =
gtk_check_button_get_active(GTK_CHECK_BUTTON(widget));

            const gchar *text = generate_pass();

            GtkEntryBuffer *buffer =
gtk_entry_get_buffer(GTK_ENTRY(entries[2]));
            gtk_entry_buffer_set_text(buffer, text, -1);

            return TRUE;
        }

        gboolean on_upper_checkbutton_press_event(GtkWidget *widget, gpointer
user_data)
        {
            is_upper_active =
gtk_check_button_get_active(GTK_CHECK_BUTTON(widget));

            const gchar *text = generate_pass();

            GtkEntryBuffer *buffer =
gtk_entry_get_buffer(GTK_ENTRY(entries[2]));
            gtk_entry_buffer_set_text(buffer, text, -1);

            return true;
        }

        void generate_button_clicked(GtkButton *button, gpointer user_data)
        {
            const gchar *text = generate_pass();

            GtkEntryBuffer *buffer =
gtk_entry_get_buffer(GTK_ENTRY(entries[2]));
            gtk_entry_buffer_set_text(buffer, text, -1);
        }

        void on_plus_min_number_button_clicked(GtkButton *button, GtkWidget
*label)
        {
            if(is_special_active == TRUE)
            {
                if(min_numb + min_special > length)
                {
                    gtk_widget_set_sensitive(GTK_WIDGET(button), FALSE);
                    return;
                }
            }
            else if(min_numb+1 > length)
            {
                gtk_widget_set_sensitive(GTK_WIDGET(button), FALSE);
                return;
            }

            min_numb++;

            char buffer[10];
            sprintf(buffer, "%.0d", min_numb);
            gtk_label_set_text(GTK_LABEL(label), buffer);

```

		В.С. Билков			ІІЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						86
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }

    void on_minus_min_number_button_clicked(GtkButton *button, GtkWidget
*label)
    {
        if(min_numb - 1 < 0)
        {
            gtk_widget_set_sensitive(GTK_WIDGET(button), FALSE);
            return;
        }

        min_numb--;

        char buffer[10];
        sprintf(buffer, "%.0d", min_numb);
        gtk_label_set_text(GTK_LABEL(label), buffer);
    }

    void on_plus_special_button_clicked(GtkButton *button, GtkWidget *label)
    {
        if(is_number_active == TRUE)
        {
            if(min_numb + min_special > length)
            {
                gtk_widget_set_sensitive(GTK_WIDGET(button), FALSE);
                return;
            }
        }
        else if(min_special+1 > length)
        {
            gtk_widget_set_sensitive(GTK_WIDGET(button), FALSE);
            return;
        }

        min_special++;

        char buffer[10];
        sprintf(buffer, "%.0d", min_special);
        gtk_label_set_text(GTK_LABEL(label), buffer);
    }

    void on_minus_special_button_clicked(GtkButton *button, GtkWidget *label)
    {
        if(min_special - 1 < 0)
        {
            gtk_widget_set_sensitive(GTK_WIDGET(button), FALSE);
            return;
        }

        min_special--;

        char buffer[10];
        sprintf(buffer, "%.0d", min_special);
        gtk_label_set_text(GTK_LABEL(label), buffer);
    }

    void login_button_clicked(GtkButton *button, GtkWidget *passentry)
    {
        GtkEntryBuffer *passbufer =
        gtk_entry_get_buffer(GTK_ENTRY(passentry));
        const char *password = gtk_entry_buffer_get_text(passbufer);
    }

```

```

// Зчитування зашифрованого паролю з файлу
FILE *file = fopen("bin/encrypted_login.bin", "rb");
char encrypted_password[AES_BLOCK_SIZE];
if (file != NULL) {
    fread(encrypted_password, sizeof(char), AES_BLOCK_SIZE, file);
    fclose(file);
}
// Розшифрування зашифрованого паролю
char decrypted_password[AES_BLOCK_SIZE];
const char *key = "my_secret_key";
aes_decrypt(encrypted_password, key, decrypted_password);

// Порівняння введеного паролю з розшифрованим паролем
//if (strcmp(password, decrypted_password) == 0)
if(true){
    //todo in the help ui
    GtkWidget *container = gtk_widget_get_parent(passentry);
    GtkWidget *window = gtk_widget_get_parent(container);
    GtkApplication *app =
gtk_window_get_application(GTK_WINDOW(window));
    gtk_window_destroy(GTK_WINDOW(window));

    content_main_window(app);
}

}

void add_button_clicked(GtkButton *button, gpointer user_data)
{
    //todo inthe help ui
    GtkWidget *window;
    GtkWidget *container, *header_box, *propertiespass_box;
    GtkWidget *submit_button;

    window = gtk_window_new();
    gtk_window_set_title(GTK_WINDOW(window), "Створити новий пароль");
    gtk_window_set_default_size(GTK_WINDOW(window), 600, 400);

    // Створюємо контейнер та блоки для заголовка та властивостей пароля
    container = gtk_box_new(GTK_ORIENTATION_VERTICAL, 10);
    header_box = gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 10);
    propertiespass_box = gtk_box_new(GTK_ORIENTATION_VERTICAL, 10);

    // Додаємо блоки до контейнера
    gtk_box_append(GTK_BOX(container), header_box);
    gtk_box_append(GTK_BOX(container), propertiespass_box);

    // Додаємо блоки заголовка та властивостей пароля
    content_add_pass_header_box(header_box);
    content_add_pass_properties_pass_box(propertiespass_box);

    // Створюємо кнопку "Створити"
    submit_button = gtk_button_new_with_label("Створити");
    g_signal_connect(submit_button, "clicked",
G_CALLBACK(on_add_pass_button_clicked), NULL);

    gtk_box_append(GTK_BOX(container), submit_button);

    // Встановлюємо контейнер як дочірній віджет для вікна та відображаємо
    вікно
    gtk_window_set_child(GTK_WINDOW(window), container);
    gtk_window_present(GTK_WINDOW(window));

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		88

```

    }

    void add_new_user_label_clicked(GtkLabel *label, gpointer user_data)
    {
        GtkWidget *window;
        GtkWidget *container;
        GtkWidget *pass_label;
        GtkWidget *pass_entry;
        GtkWidget *create_button;

        window = gtk_window_new();
        gtk_window_set_title(GTK_WINDOW(window), "Створення нового
акаунту");
        gtk_window_set_default_size(GTK_WINDOW(window), 600, 400);

        container = gtk_box_new(GTK_ORIENTATION_VERTICAL, 10);

        pass_label = gtk_label_new("Придумайте пароль для вашого акаунту від
5 до 20 символів\n(Постарайтесь його не забути, тому що ви можете втратити ваші
дані)");

        pass_entry = gtk_entry_new();

        create_button = gtk_button_new_with_label("Створити акаунт");
        g_signal_connect(create_button, "clicked",
G_CALLBACK(check_password_length), pass_entry);

        gtk_box_append(GTK_BOX(container), pass_label);
        gtk_box_append(GTK_BOX(container), pass_entry);
        gtk_box_append(GTK_BOX(container), create_button);

        gtk_window_set_child(GTK_WINDOW(window), container);
        gtk_window_present(GTK_WINDOW(window));
    }

    void check_password_length(GtkWidget *button, GtkWidget *passentry)
    {
        GtkEntryBuffer *passbufer =
gtk_entry_get_buffer(GTK_ENTRY(passentry));
        const char *password = gtk_entry_buffer_get_text(passbufer);
        int length = strlen(password);

        if (length >= 5 && length <= 20) {
            // Шифрування паролю, якщо він відповідає вимогам довжини
            char encrypted_password[AES_BLOCK_SIZE];
            const char *key = "my_secret_key";
            aes_encrypt(password, key, encrypted_password);

            FILE *file = fopen("bin/encrypted_login.bin", "wb");
            if (file != NULL) {
                fwrite(encrypted_password, sizeof(char), AES_BLOCK_SIZE,
file);
                fclose(file);
            }
        } else {
            g_print("Пароль повинен мати від 5 до 20 символів.\n");
        }
    }

    // Функція шифрування AES
    void aes_encrypt(const char *input, const char *key, char *output)
    {

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		89

```

        AES_KEY aes_key;
        AES_set_encrypt_key((const unsigned char *)key, 128, &aes_key);
        AES_encrypt((const unsigned char *)input, (unsigned char *)output,
&aes_key);
    }

    // Функція дешифрування AES
    void aes_decrypt(const char *input, const char *key, char *output)
    {
        AES_KEY aes_key;
        AES_set_decrypt_key((const unsigned char *)key, 128, &aes_key);
        AES_decrypt((const unsigned char *)input, (unsigned char *)output,
&aes_key);
    }

    void on_add_pass_button_clicked(GtkButton *button, gpointer user_data)
    {
        // Отримання тексту з полів вводу
        const char *type_text;
        GtkTreeIter iter;
        GtkTreeModel *model = gtk_combo_box_get_model(entries[0]);
        if (gtk_combo_box_get_active_iter(entries[0], &iter)) {
            gtk_tree_model_get(model, &iter, 0, &type_text, -1);
        }

        //const          char          *type_text          =
        gtk_editable_get_text(GTK_EDITABLE(entries[0]));
        const          char          *name_text          =
        gtk_editable_get_text(GTK_EDITABLE(entries[1]));
        const          char          *pass_text          =
        gtk_editable_get_text(GTK_EDITABLE(entries[2]));

        /*const char *key = "my_secret_key";

        // Шифрування тексту
        char encrypted_type[AES_BLOCK_SIZE];
        char encrypted_name[AES_BLOCK_SIZE];
        char encrypted_password[AES_BLOCK_SIZE];

        aes_encrypt(type_text, key, encrypted_type);
        aes_encrypt(name_text, key, encrypted_name);
        aes_encrypt(pass_text, key, encrypted_password);

        // Запис зашифрованих даних у файл
        FILE *file = fopen("bin/encrypted_passwords.bin", "wb");
        if (file != NULL) {
            fwrite(encrypted_type, sizeof(char), AES_BLOCK_SIZE, file);
            fwrite(encrypted_name, sizeof(char), AES_BLOCK_SIZE, file);
            fwrite(encrypted_password, sizeof(char), AES_BLOCK_SIZE, file);
            fclose(file);
        }
        g_print("Just text\n");
        g_print("Type: %s\n", type_text);
        g_print("Name: %s\n", name_text);
        g_print("Password: %s\n", pass_text);

        g_print("Crypted text----\n");
        g_print("Crypted Type: %s\n", encrypted_type);
        g_print("Crypted Name: %s\n", encrypted_name);
        g_print("Crypted Password: %s\n", encrypted_password);*/

        FILE *file = fopen("bin/encrypted_passwords.bin", "ab");

```

		В.С. Билков			ІІЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						90
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        if (file != NULL) {
            size_t type_len = strlen(type_text) + 1;    // +1 для нуль-
термінатора
            size_t name_len = strlen(name_text) + 1;
            size_t pass_len = strlen(pass_text) + 1;

            fwrite(&type_len, sizeof(size_t), 1, file);
            fwrite(&name_len, sizeof(size_t), 1, file);
            fwrite(&pass_len, sizeof(size_t), 1, file);
            fwrite(type_text, sizeof(char), type_len, file);
            fwrite(name_text, sizeof(char), name_len, file);
            fwrite(pass_text, sizeof(char), pass_len, file);

            fclose(file);
        }

        // Створення горизонтального контейнера для кожного елементу
        GtkWidget *item_box = gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 5);
        gtk_box_set_homogeneous(GTK_BOX(item_box), TRUE);
        gtk_box_append(GTK_BOX(container_box), item_box);

        // Створення кнопки з іменем
        GtkWidget *button_pass = gtk_button_new();
        GtkWidget *box = gtk_box_new(GTK_ORIENTATION_HORIZONTAL, 5);

        // Визначення типу та вибір відповідної картинки
        GtkWidget *image = NULL;

        if (strcmp(type_text, "Вхід") == 0) {
            image = gtk_image_new_from_file("img/global.png");
            // Отримання тексту з мітки
            char *text = gtk_label_get_text(GTK_LABEL(counters[0]));

            // Конвертування тексту в ціле число
            int value = atoi(text);

            sprintf(text, "%d", value + 1);
            gtk_label_set_text(GTK_LABEL(counters[0]), text);
        } else if (strcmp(type_text, "Банк") == 0) {
            image = gtk_image_new_from_file("img/credit-card.png");
            // Отримання тексту з мітки
            char *text = gtk_label_get_text(GTK_LABEL(counters[1]));

            // Конвертування тексту в ціле число
            int value = atoi(text);

            sprintf(text, "%d", value + 1);
            gtk_label_set_text(GTK_LABEL(counters[1]), text);
        } else if (strcmp(type_text, "Інші") == 0) {
            image = gtk_image_new_from_file("img/card.png");
            // Отримання тексту з мітки
            char *text = gtk_label_get_text(GTK_LABEL(counters[2]));

            // Конвертування тексту в ціле число
            int value = atoi(text);

            sprintf(text, "%d", value + 1);
            gtk_label_set_text(GTK_LABEL(counters[2]), text);
        }

        gtk_box_append(GTK_BOX(box), image);
    
```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						91
Змн.	Арк.	№ докум.	Підпис	Дата		

```

// Створення етикетки з іменем
GtkWidget *name_label = gtk_label_new(name_text);
gtk_box_append(GTK_BOX(box), name_label);

// Додавання вертикального контейнера до кнопки
gtk_button_set_child(GTK_BUTTON(button_pass), box);

g_signal_connect(button_pass, "clicked",
G_CALLBACK(button_item_clicked), item_box);
gtk_box_append(GTK_BOX(item_box), button_pass);

// Створення етикетки для паролю (початково прихована)

char password_text[256];
snprintf(password_text, sizeof(password_text), "Пароль: %s",
pass_text);

GtkWidget *password_label = gtk_label_new(password_text);
gtk_widget_set_visible(password_label, FALSE);
gtk_widget_set_hexpand(password_label, TRUE);
gtk_label_set_xalign(GTK_LABEL(password_label), 0.0);
gtk_widget_set_margin_start(password_label, 10);
gtk_box_append(GTK_BOX(item_box), password_label);

// Створення кнопки видалення
GtkWidget *delete_button = gtk_button_new();
GtkWidget *delete_image = gtk_image_new_from_file("img/trash.png");
gtk_widget_set_hexpand(delete_button, FALSE);
gtk_widget_set_halign(delete_button, GTK_ALIGN_END);
gtk_button_set_child(GTK_BUTTON(delete_button), delete_image);

// Визначення видимості кнопки видалення
gtk_widget_set_visible(delete_button, FALSE);

// Додавання кнопки видалення до контейнера
gtk_box_append(GTK_BOX(item_box), delete_button);

// З'єднання обробника події для кнопки видалення
//g_signal_connect(delete_button, "clicked",
G_CALLBACK(delete_button_clicked), item_box);

GtkWidget *toplevel = gtk_widget_get_ancestor(button,
GTK_TYPE_WINDOW);

if (GTK_IS_WINDOW(toplevel)) {
    gtk_window_close(GTK_WINDOW(toplevel));
}

}

void button_item_clicked(GtkButton *button, gpointer user_data) {
    GtkWidget *box = GTK_WIDGET(user_data);

    // Отримання першого дочірнього елемента контейнера
    GtkWidget *child = gtk_widget_get_last_child(GTK_WIDGET(box));

    // Отримання передостаннього дочірнього елемента контейнера
    GtkWidget *prev_sibling = gtk_widget_get_prev_sibling(child);

    // Перевірка стану видимості другого дочірнього елемента
    gboolean visible = gtk_widget_get_visible(child);

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
						92
Змн.	Арк.	№ докум.	Підпис	Дата		

```

if (visible) {
    // Приховуємо другий дочірній елемент
    gtk_widget_hide(child);
    gtk_widget_hide(prev_sibling);
} else {
    // Показуємо другий дочірній елемент
    gtk_widget_show(child);
    gtk_widget_show(prev_sibling);
}
}

```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		93

Збірка

Makefile

```
CXX = gcc
CXXFLAGS = $(shell pkg-config --cflags gtk4) -Wextra -
I/path/to/openssl/include

LIBS = $(shell pkg-config --libs gtk4) -L/path/to/openssl/lib -lssl -lcrypto

SRC := $(wildcard src/*.c)
OBJ := $(patsubst src/%.c, src/%.o, $(SRC))

src/%.o: src/%.c
    $(CXX) -c $< $(CXXFLAGS) -o $@

compile: $(OBJ)
    mkdir -p bin
    $(CXX) $(LIBS) $(CXXFLAGS) $^ -o bin/$@

clean:
    rm -rf src/*.o bin/
```

		В.С. Билков			ІПЗ.КР.Б – 121 – 23 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		94