Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

магістра
(назва освітньо-кваліфікаційного рівня)

студента *Гарбуза Вадима Олександровича*
(ПІБ)

академічної групи *122м-22-3*
(шифр)

спеціальності *122 Комп'ютерні науки*
(код і назва спеціальності)

освітньої програми *Комп'ютерні науки*
(назва освітньої програми)

на тему: *Розробка додатку EncryptPro та дослідження ефективності навчання криптографічним методам шифрування на його базі.*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	*Доц.Кабак Л.В.*			
розділів:				
спеціальний	*Доц.Кабак Л.В*			
економічний				
Рецензент	*проф. Корнієнко В.І.*			
Нормоконтролер	*проф. Лактіонов І.С.*			

Дніпро
2023

Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
Завідувач кафедри
Програмного забезпечення комп'ютерних систем
(повна назва)

_____ І.М. Удовик
(підпис) (прізвище, ініціали)

« » _____ 2023 Року

ЗАВДАННЯ
на виконання кваліфікаційної роботи магістра

спеціальності *122 Комп'ютерні науки*
(код і назва спеціальності)

122м-22-3 (група) *Гарбузу Вадиму Олександровичу* (прізвище та ініціали)

студенту

Тема дипломного проекту *Розробка додатку EncryptPro та дослідження ефективності навчання криптографічним методам шифрування на його базі.*

1 ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Наказ ректора НТУ «Дніпровська політехніка» від 9.10.2023 р. № 1227

2 МЕТА ТА ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Актуальність роботи зумовлена достатньо високою популярністю використання шифрування в наш час для захисту особистих даних.

Об'єктом дослідження є процес навчання користувачів криптографічним методам шифрування та перевірки їх знань за допомогою тестів.

Метою роботи є підвищення рівня знань студентів з теми шифрування та захисту інформації.

3 ОЧІКУВАНІ НАУКОВІ РЕЗУЛЬТАТИ

Практичним значенням є надання користувачам можливості навчання методам шифрування та перевірки знань за допомогою тестових завдань.

Наукова новизна роботи полягає в набутті подальшого розвитку методик візуалізації інноваційних криптографічних методів шифрування що сприяє підвищенню ефективності навчання. На базі цього методу було створено програмне забезпечення EncryptPro.

Практична цінність полягає в покращенні освоєння криптографії, підвищенні рівня кібербезпеки та стимулюванні інтересу до цієї галузі через новаторський підхід до навчання.

Обгрунтовано підставу для розробки програмного рішення, було здійснено деталізацію мети та задач проекту, визначено функціональні можливості програми для навчання шифруванню. Для розробки програми було обрано такі технологій та засоби як C#, в якості мови програмування, Visual Studio як середовище розробки. На етапі проектування було побудовано діаграми, які показую взаємодію користувача з додатком, а саме було побудовано діаграми дій Користувача, ER – діаграму, діаграму клієнтської частини та діаграму компонентів.

4 ЕТАПИ ВИКОНАННЯ РОБІТ

Найменування етапів робіт	**Строки виконання робіт(початок - кінець**
Аналіз існуючих рішень та постановка задачі роботи	12.09.2022- 30.09.2022
Аналіз засобів створення програмного забезпечення побудови діаграми процесу додатку для навчання шифруванню та перевірки знань студентів.	01.10.2022-31.10.2022
Розробка програмного забезпечення та дослідженняефективності запропонованих рішень.	01.11.2022-10.12.2022

Завдання видав _____ *Кабак Л.В.*
(підпис) (прізвище, ініціали)

Завдання прийняв до виконання _____ *Гарбуз В.О.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: _____

Термін подання дипломного проекту до ЕК _____

РЕФЕРАТ

Пояснювальна записка: 69 стор., 45 рис., 9 таблиці, 2 додатка, 19 джерел.

Об’єктом дослідження є перевірка якості знань студентів за допомогою розроблюваного додатку для навчання шифруванню.

Предметом дослідження є розробка програмного забезпечення, яке використовує криптографічні методи для шифрування та розшифрування інформації.

Наукова новизна роботи полягає в набутті подальшого розвитку методик візуалізації інноваційних криптографічних методів шифрування що сприяє підвищенню ефективності навчання. На базі цього методу було створено програмне забезпечення EncryptPro.

Метою роботи є підвищення рівня знань студентів з теми шифрування та захисту інформації.

Для досягнення мети потрібно вирішити такі задачі:

1) Аналіз криптографічних методів шифрування та їх ефективності.

2) Вибір оптимальних алгоритмів шифрування та середовища розробки.

3) Розробка програмного забезпечення EncryptPro для шифрування та розшифрування даних.

4) Тестування програмного забезпечення на ефективність та надійність шифрування.

5) Підготовка документації, що описує функціональність та принципи роботи EncryptPro.

Практична цінність полягає в покращенні освоєння криптографії, підвищенні рівня кібербезпеки та стимулюванні інтересу до цієї галузі через новаторський підхід до навчання.

Цей проект важливий для забезпечення захисту конфіденційної

інформації та надання користувачам засобів ефективного навчання за допомогою криптографічних методів.

СПИСОК КЛЮЧОВИХ СЛІВ: КРИПТОГРАФІЧНІ МЕТОДИ, ШИФРУВАННЯ ДАНИХ, АНАЛІЗ МЕТОДІВ ШИФРУВАННЯ, ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, ДОКУМЕНТАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, ШИФРУВАЛЬНІ АЛГОРИТМИ.

ABSTRACT

Explanatory note: 69 pages, 45 figures, 2 tables, 9 appendices, 19 sources.

The object of the research is to evaluate students' knowledge using a developed application for learning encryption.

The subject of the research is the development of software that utilizes cryptographic methods for encrypting and decrypting information.

The scientific novelty of the work lies in the acquisition of further development of visualization techniques for innovative cryptographic encryption methods, which contributes to enhancing the efficiency of learning. Based on this method, the software EncryptPro has been created.

The aim of the work is to enhance students' knowledge level on the topic of encryption and information security..

To achieve this goal, the following tasks need to be addressed:

1) Analysis of cryptographic encryption methods and their effectiveness.

2) Selection of optimal encryption algorithms and development environments.

3) Development of EncryptPro software for data encryption and decryption.

4) Testing the software for encryption effectiveness and reliability.

5) Preparation of documentation describing EncryptPro's functionality and principles of operation.

The practical value lies in enhancing understanding of cryptography, elevating cybersecurity levels, and fostering interest in the field through an innovative approach to education.

This project is crucial for safeguarding confidential information and providing users with effective learning tools using cryptographic methods.

KEYWORDS: CRYPTOGRAPHIC METHODS, DATA ENCRYPTION, ANALYSIS OF ENCRYPTHION METHODS, SOFTWARE TESTING,

SOFTWARE DOCUMENTATION, ENCRYPTION ALGORITHMS.

ЗМІСТ

ВСТУП

Сучасний інформаційний вік вимагає захисту конфіденційності та безпеки даних, в особливості при передачі їх через мережу. Криптографія, як наука про захист інформації шляхом шифрування та розшифрування, відіграє важливу роль у забезпеченні цієї безпеки. Зростання важливості цієї області привело до появи численних криптографічних методів та інструментів, призначених для забезпечення конфіденційності та цілісності даних [1].

У цьому контексті важливо не лише розробляти нові криптографічні методи, але і забезпечувати їх широке використання і розуміння серед користувачів. Навчання цих методів відіграє важливу роль у підготовці спеціалістів у сфері інформаційної безпеки.

Магістерська робота присвячена дослідженню ефективності навчання криптографічним методам шифрування за допомогою розробленого додатку для навчання та тестування, який я назвав "EncryptPro". Метою цього дослідження є визначення того, наскільки успішно цей додаток може навчати користувачів криптографічним навичкам і які ефективність та задоволеність користувачів можуть бути досягнуті за його допомогою.

У цьому дослідженні ми будемо досліджувати різні аспекти навчання криптографічним методам, такі як швидкість навчання, якість засвоєння матеріалу та загальне сприйняття користувачами. Ми також розглянемо можливості вдосконалення додатку "EncryptPro" на основі отриманих результатів дослідження.

Дана робота спрямована на підвищення ефективності навчання криптографічним методам шифрування та покращення якості підготовки фахівців у галузі інформаційної безпеки. Відповідно, результати цього дослідження можуть мати значущий внесок у практичну сферу криптографії та навчання інформаційної безпеки.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ РОБОТИ

1.1 Обгрунтування потреби у створенні EncryptPro

Сучасний світ зазнає стійкого зростання обсягу обміну електронною інформацією в різних сферах, починаючи від особистих даних та закінчуючи корпоративними та державними інформаційними системами. З цим зростанням обсягу інформації виникає нагальна потреба у її надійному захисті від несанкціонованого доступу. Криптографія є основним інструментом для забезпечення конфіденційності та цілісності цієї інформації.

Кібератака (також відома як кібернапад) - це навмисний інтернет-захоплюючий захід, спрямований на комп'ютерну систему, мережу, програмне забезпечення або дані з метою завдання шкоди, крадіжки інформації, розповсюдження вірусів, втручання в роботу системи або інших зловмисних дій. Кібератаки можуть бути виконані різними методами, включаючи відправку шкідливих програм, фішинг, деніал-оф-сервіс атаки, використання вразливостей в програмному забезпеченні тощо [2].

Метою кібератак може бути отримання конфіденційної інформації, викрадення грошей, руйнування робочих процесів, завдання політичної шкоди, або навіть терористичні цілі. Захист від кібератак включає в себе використання антивірусного програмного забезпечення, мережевих засобів безпеки, багатофакторної аутентифікації та регулярні оновлення програмного забезпечення для усунення вразливостей. Крім того, важливо навчати користувачів вибирати безпечні паролі і бути обережними в Інтернеті, щоб уникнути фішингових атак та інших шахрайських схем.

Рис. 1.1. Хакерські атаки у світі

Зі зростом кібератак по всьому світу з'явилась потреба в більш сильному захисті інформації. Для цього і була створена технологія шифрування даних. Нижче наведено основні види програм для захисту даних.

Найпоширенішими прикладами програмних засобів захисту інформації є такі:

1) система контролю і управління доступом;
2) антивірусні програми;
3) шифрувальне програмне забезпечення;
4) мережевий екран;
5) система виявлення вторгнень;
6) керування записами;
7) пісочниця;
8) система управління інформаційною безпекою
9) SIEM.

Однак криптографічні методи та принципи є вкрай складними і технічними, і для їх повного розуміння та використання необхідна відповідна підготовка. Завдяки росту інтернету та доступності технологій, ця область стала дедалі більш доступною, але одночасно і більш вразливою до зловмисних

атак і порушень [3].

Отже, створення додатку EncryptPro є насущною потребою з наступними обгрунтуваннями:

Сприяння інформаційній безпеці: Зростання кількості цифрової інформації створює більше можливостей для порушень безпеки. EncryptPro надає можливість здійснювати ефективне навчання криптографії, що сприяє покращенню рівня інформаційної безпеки.

Забезпечення доступності: Навчання криптографічним методам раніше вимагало значних зусиль і витрат, але EncryptPro робить цю область більш доступною для широкого загалу користувачів, не залежно від їхнього технічного рівня.

Підготовка фахівців: EncryptPro може служити як ефективний інструмент для підготовки фахівців з інформаційної безпеки, які відіграють ключову роль у захисті даних та мереж.

Зацікавленість користувачів: Зацікавленість користувачів у вивченні криптографії може бути підвищено завдяки інтерактивному та ефективному процесу навчання, що надається EncryptPro.

Аналіз та вдосконалення: Додаток EncryptPro також надає можливість збирати дані про взаємодію користувачів з матеріалом і аналізувати ефективність навчання, що може призвести до подальшого вдосконалення методів навчання криптографії.

Усі ці аспекти обгрунтовують необхідність створення додатку EncryptPro та дослідження ефективності навчання криптографічним методам шифрування на його базі, що є обґрунтованою відповіддю на виклики і вимоги сучасного цифрового світу.

1.2 Опис технологій розоробки програм для шифрування

Шифрування – оборотне перетворення даних, з метою приховання інформації, дешифрування – зворотній процес, що полягає у відновленні

первинних (до шифрування) даних. Проте, у сучасній літературі можна знайти інші визначення: під шифруванням розуміється синтез процесів зашифрування і розшифрування, а от дешифрування – це відновлення вхідного тексту без знання ключа (тобто, це процес злому шифру – криптоаналіз). Єдиної думки сьогодні не існує, можливу дану ситуацію виправить прийняття національного стандарту у галузі криптографії [4].

Розробка програм для шифрування є важливою галуззю інформаційної безпеки, і вона включає в себе використання різних технологій та підходів для забезпечення конфіденційності інформації. Ось опис основних технологій розробки програм шифрування:

1) Алгоритми шифрування:

а) Симетричне шифрування: Використовує один і той же ключ як для шифрування, так і розшифрування даних. Популярні алгоритми включають AES (Advanced Encryption Standard) і DES (Data Encryption Standard).

б) Асиметричне (публічний ключ) шифрування: Використовує пару ключів - публічний і приватний. Публічний ключ використовується для шифрування інформації, а приватний - для розшифрування. RSA і ECC (Elliptic Curve Cryptography) - це популярні алгоритми асиметричного шифрування.

2) Протоколи забезпечення з'єднання:

а) TLS/SSL: Використовується для шифрування даних між клієнтом і сервером під час передачі через мережу, таких як веб-сайти та електронна пошта. TLS (Transport Layer Security) і SSL (Secure Sockets Layer) забезпечують безпеку під час зв'язку.

3) Хеш-функції:

а) Хеш-функції використовуються для створення фіксованої довжини хеш-кодів із вхідних даних. Це важливо для перевірки цілісності даних. Популярні хеш-функції включають SHA-256 та MD5.

4) Захист від атак:

а) Захист від атак, таких як brute force або атаки посередників (man-in-the-middle), включає в себе заходи безпеки, які обмежують можливість

зламування шифрування. До таких заходів можуть входити обмеження повторних спроб введення пароля і використання публічних ключів для перевірки автентичності.

5) Інтеграція з іншими технологіями:

а) Розробка програм для шифрування може включати в себе інтеграцію з іншими технологіями, такими як біометричні дані, двофакторна аутентифікація і захищене сховище ключів.

6) Аудит і відстеження:

а)Для забезпечення безпеки програми шифрування важливо вести аудит та відстежувати дії користувачів, щоб виявляти можливі порушення інформаційної безпеки.

Розробка програм для шифрування вимагає глибокого розуміння криптографії і технічних аспектів забезпечення інформаційної безпеки. Враховуючи постійний розвиток технологій та загроз, ця галузь постійно еволюціонує для забезпечення максимального рівня безпеки і конфіденційності даних [5].

1.3 Аналіз існуючих програм шифрування та тестування знань

Аналіз конкурентів для додатку, який поєднує функції шифрування даних та навчання криптографії для студентів, може допомогти з'ясувати, які альтернативні рішення вже існують на ринку. Ось деякі програми-конкуренти, які можуть бути враховані:

Cryptool 2:

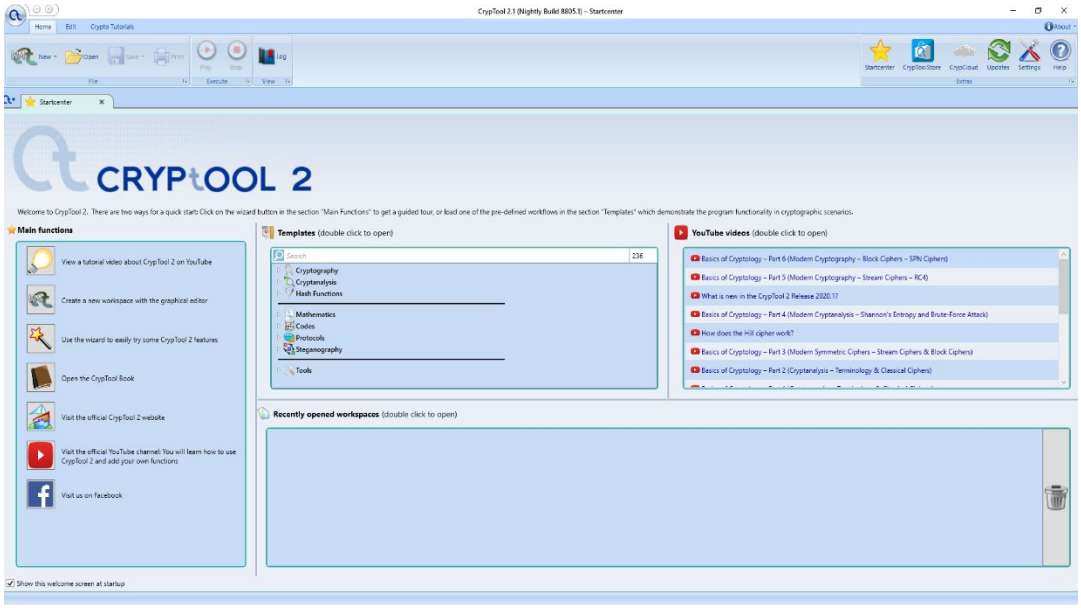


Рис. 1.2. Cryptool 2

Опис: Це безкоштовний відкритий додаток для навчання та практичного застосування криптографії.

Переваги:

Інтерактивні вправи та завдання для навчання шифрування.

Можливість власноруч створювати та тестувати криптографічні алгоритми.

Недоліки:

Вимагає завантаження та інсталяції.

Інтерфейс може бути складним для новачків [6].

CryptPad:

Рис. 1.3. CryptPad

Опис: Це онлайн-сервіс, який дозволяє користувачам шифрувати та спільно працювати з даними в хмаровому середовищі.

Переваги:

Легкість використання і доступ до даних з будь-якого пристрою.

Вбудована шифрування та захист приватності.

Недоліки:

Орієнтований на роботу з даними, а не на активне навчання криптографії [7].

Kahoot!:

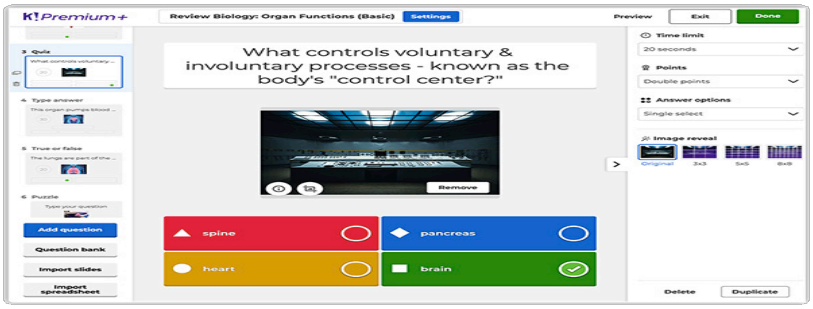


Рис. 1.4. Kahoot!

Опис: Комплексна платформа для створення та проведення онлайн-тестів та вікторин на різні теми, включаючи криптографію.

Переваги:

Легка інтеграція в освітній процес.

Забезпечує можливість створення власних криптографічних тестів.

Недоліки:

Орієнтований на тестування, а не на глибоке навчання [8].

CyberChef:

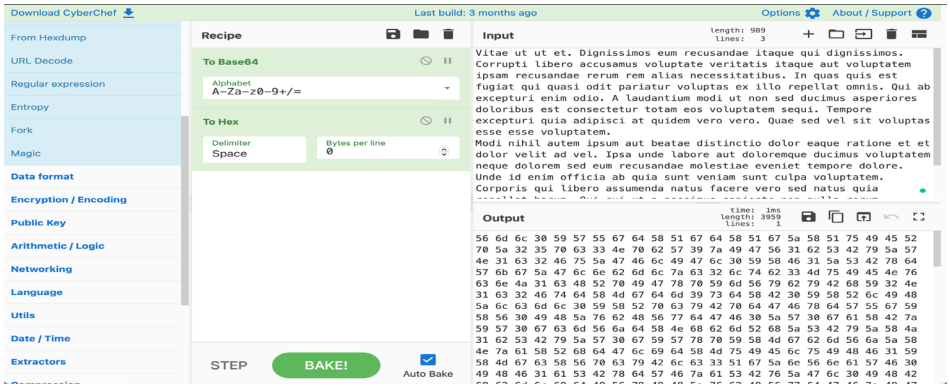


Рис. 1.5. CyberChef

Опис: Це відкритий онлайн-інструмент для аналізу, декодування та

шифрування даних.

Переваги:

Простий та інтуїтивно зрозумілий інтерфейс.

Можливість виконання різних криптографічних операцій.

Недоліки:

Не має інтегрованого навчання або навчальних матеріалів.

Враховуючи конкурентне оточення, EncryptPro може виділятися за рахунок комбінації інтерактивних навчальних матеріалів, можливості шифрування даних та створення практичних завдань, що спрямовані на глибоке засвоєння та застосування знань у сфері криптографії. Додаток також може надавати аналітику процесу навчання для подальшого вдосконалення навчального процесу [9].

1.4 Постановка задачі

Головною метою роботи є розробка додатку для криптографічного шифрування та дешифрування тексту з функцією перевірки знань за допомогою тестових завдань.

Для досягнення мети роботи були визначені такізадачі:

1. здійснити аналіз предметної області;
2. здійснити аналіз вимог;
3. проаналізувати технології розробки програм для шифрування;
4. обрати методи та засоби реалізації;
5. здійснити проектування програми;
6. розробка програми;
7. тестування програми;
8. розробка супровідної документації.

Після встановлення мети та задач проекту було визначено, що програма повиненна мати такі функціональні можливості:

a) мати команди для шифрування данних та їх дешифрування;

b) мати команди для переходу між сторінками програми;

c) мати команди для збереження інформації в файл;

d) мати команди для друку інформації;

e) мати команди для створення навчальних тестів;

f) мати команди для відображення статистики проходження тестів;

Після виконання аналізу було встановлено що, програма для шифрування та проведення тестів для узагальнення знань буде корисною для широкого кола користувачів які бажають навчитись криптографії.

РОЗДІЛ 2
МЕТОДИ ТА ТЕХНОЛОГІЇ ВИРІШЕННЯ ЗАДАЧІ

2.1 .NET Framework

Фреймворк – це основа програми, що складається з набору структурованих інструментів. Він визначає загальний каркас або скелет, на якому будується програма. Бібліотеки, у свою чергу, представляють собою частини коду, які вбудовуються у програму та виконують певні завдання відповідно до вимог розробників. Зазвичай, бібліотека містить набір функцій або класів, які можна використовувати у процесі розробки програм.

Разом вони дозволяють розробникам будувати функціонал програми на основі визначених основних засад. Фреймворк установлює основну структуру та підхід до організації програмного коду, визначаючи, яким чином потрібно писати програму.

Програми можуть бути розроблені на основі як фреймворку, так і на базі окремих бібліотек. Проте розробники частіше використовують фреймворки спільно з різноманітними бібліотеками для побудови повноцінного додатку.

Фреймворки дозволяють створювати унікальний набір функцій та ідеально підходять для розробки нетипових продуктів. JavaScript-фреймворки, наприклад, сприяють підвищенню продуктивності розробника, оскільки містять в собі певні правила, шаблони та набори функцій.

Вибір фреймворку або набору блоків для побудови нового проекту залежить від знань та досвіду розробника, а також від специфіки поставлених завдань. Сучасні фреймворки мають значний функціонал і дозволяють реалізувати різноманітні завдання, що робить їх популярними серед розробників.

.NET Framework - це платформа розробки програмного забезпечення, створена корпорацією Microsoft. Її історія розпочалась наприкінці 1990-х років, коли Microsoft вирішила створити нову технологію для розробки програм, яка б

дозволила розробникам створювати програмне забезпечення для різних платформ з використанням різних мов програмування.

Першою версією .NET Framework була версія 1.0, яка вийшла у 2002 році. Це була значна подія в світі розробки програмного забезпечення, оскільки .NET Framework пропонував зручний інтерфейс для створення програм, використовуючи різні мови програмування, такі як C#, Visual Basic .NET, та C++.

Основними принципами .NET були спрощення розробки програм шляхом використання переносимих бібліотек класів (CLR - Common Language Runtime) і мовної незалежності, що дозволяло розробникам використовувати мову, з якою вони були зручні, без втрати функціональності або продуктивності [10].

З часом .NET Framework вдосконалювався, отримав нові версії, покращення та додаткові функції. Наприклад, введення ASP.NET для веб-розробки, Windows Presentation Foundation (WPF) для розробки десктопних додатків, та інші інструменти.

Пізніше Microsoft розширила свою платформу і представила .NET Core - відкритий, крос-платформений варіант .NET Framework, який був зосереджений на підтримці різних операційних систем, таких як Windows, macOS та Linux. .NET Core став популярним серед розробників через свою гнучкість та можливості роботи на різних платформах.

У 2020 році .NET Core був об'єднаний з .NET Framework у нову платформу .NET 5, яка продовжує розвиватися та набирати популярність серед розробників з усього світу.

Якщо звернутися до статистики числа завантажень (Рис. 2.1), то .NET обирають частіше, ніж решту два фреймворки.

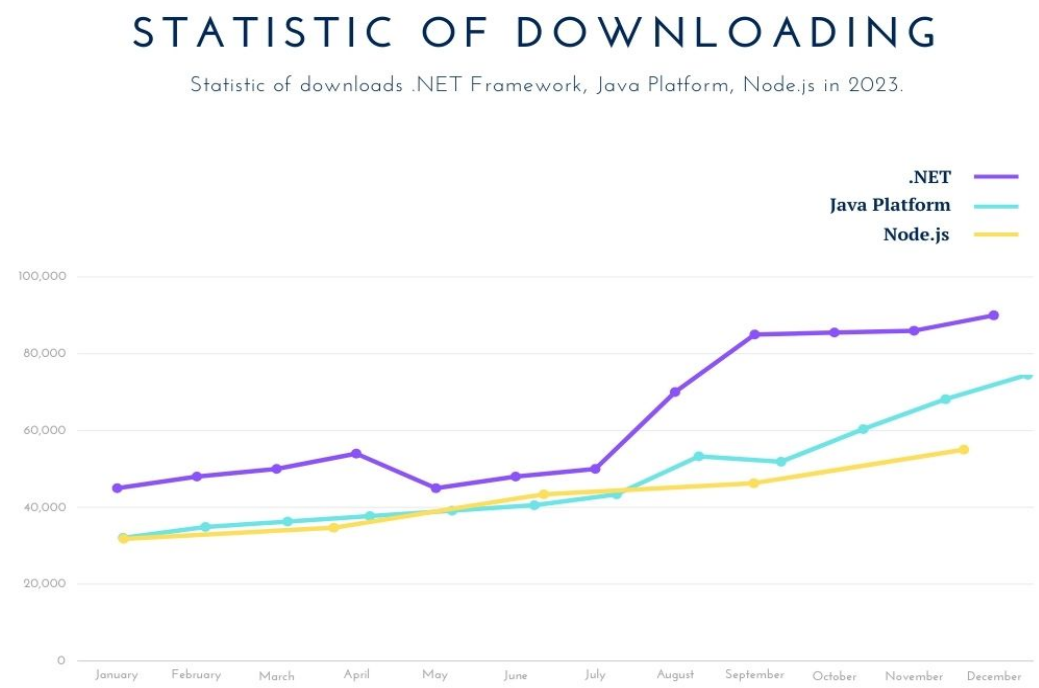


Рис. 2.1. Статистика числа завантажень конкурентних фреймворків

.NET Framework та його наступники .NET Core і .NET 5/6 мають кілька конкурентів у світі розробки програмного забезпечення. Ось деякі з них:

Java: Java вважається одним з основних конкурентів .NET. Вона має велику спільноту розробників і широке застосування у великих корпораціях. Java також пропонує переносимість між різними платформами завдяки своєму віртуальному середовищу виконання.

Python: Python - це інша популярна мова програмування, яка конкурує з .NET, зокрема в галузі швидкості розробки та простоти використання. Python має велику кількість бібліотек і фреймворків для широкого спектру завдань.

JavaScript (і його фреймворки): У світі веб-розробки, JavaScript, а також фреймворки та бібліотеки, такі як React, Angular та Vue.js, конкурують з .NET, особливо у плані розробки клієнтських додатків та веб-сайтів [11].

PHP: PHP є дуже популярним у веб-розробці, зокрема для створення веб-сайтів та веб-додатків. Він конкурує з .NET у сфері серверного веб-програмування.

Ruby (і Ruby on Rails): Ruby має свою платформу для веб-розробки, відому як Ruby on Rails, яка конкурує з .NET у веб-розробці, особливо в сфері швидкості розробки та простоти використання.

Ці конкуренти пропонують свої унікальні переваги та можливості, і вибір між ними зазвичай залежить від конкретних потреб проекту, вподобань розробника та характеристик самого завдання.

.NET має кілька переваг порівняно зі своїми конкурентами:

Широкий спектр мов програмування: .NET підтримує різні мови програмування, такі як C#, F#, Visual Basic .NET, та інші. Це дає розробникам можливість вибору мови, яка найбільше підходить для конкретного завдання.

Інтегрована середовище розробки (IDE): Visual Studio, основне середовище розробки для .NET, є потужним та функціональним інструментом з великою кількістю утиліт для розробки, налагодження та відлагодження програм [12].

Велика бібліотека класів: .NET має обширну бібліотеку класів, яка забезпечує широкий спектр функціональності для розробників. Це дозволяє розробникам швидко будувати програми, використовуючи готові рішення.

Крос-платформеність: З появою .NET Core та .NET 5/6, .NET став крос-платформеним, що означає, що програми, розроблені на .NET, можуть працювати на різних операційних системах, таких як Windows, macOS та Linux.

Висока продуктивність: .NET відомий своєю продуктивністю та ефективністю в роботі, особливо у великих корпоративних системах [13].

Підтримка корпорацією Microsoft: Як продукт Microsoft, .NET має сильну підтримку, регулярні оновлення та активну спільноту розробників.

Хоча існують інші конкуренти зі своїми унікальними перевагами, .NET продовжує залишатися конкурентоспроможним завдяки своїм функціям, широкому спектру можливостей та підтримці від великої IT-корпорації [14].

2.2 Visual Studio

Visual Studio є одним з найпопулярніших та потужних інтегрованих середовищ розробки (IDE) для програмістів у світі програмування. Він розроблений компанією Microsoft і має широкий набір функціоналу для різних типів програмних продуктів. Ось більш докладний огляд його функціоналу [15]:

Мови програмування та платформи:

Visual Studio підтримує багато мов програмування, включаючи, але не обмежуючись:

C#: Основна мова для розробки додатків на платформі .NET.

C++: Для системного програмування, розробки додатків та ігор.

Python, JavaScript, TypeScript, інші: Розширюється до багатьох інших мов програмування.

Редактор коду:

Синтаксичне виділення та автодоповнення: Допомагає зробити код більш читабельним та швидше вводити його. Синтаксичне виділення та автодоповнення відображене на (Рис.2.2).

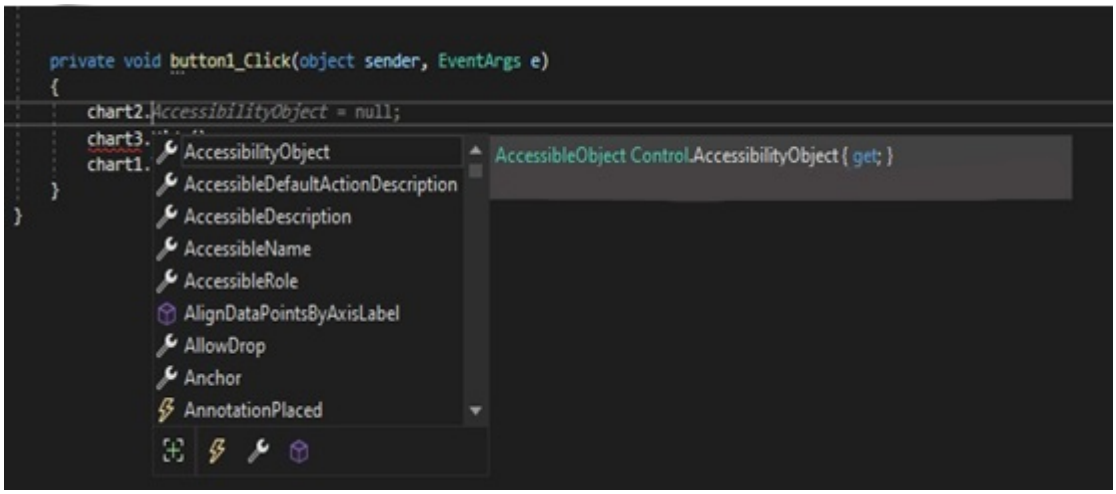


Рис. 2.2. Синтаксичне виділення та автодоповнення

Перевірка помилок: Негайне виявлення синтаксичних та логічних

помилок у коді відображене на (Рис. 2.3).

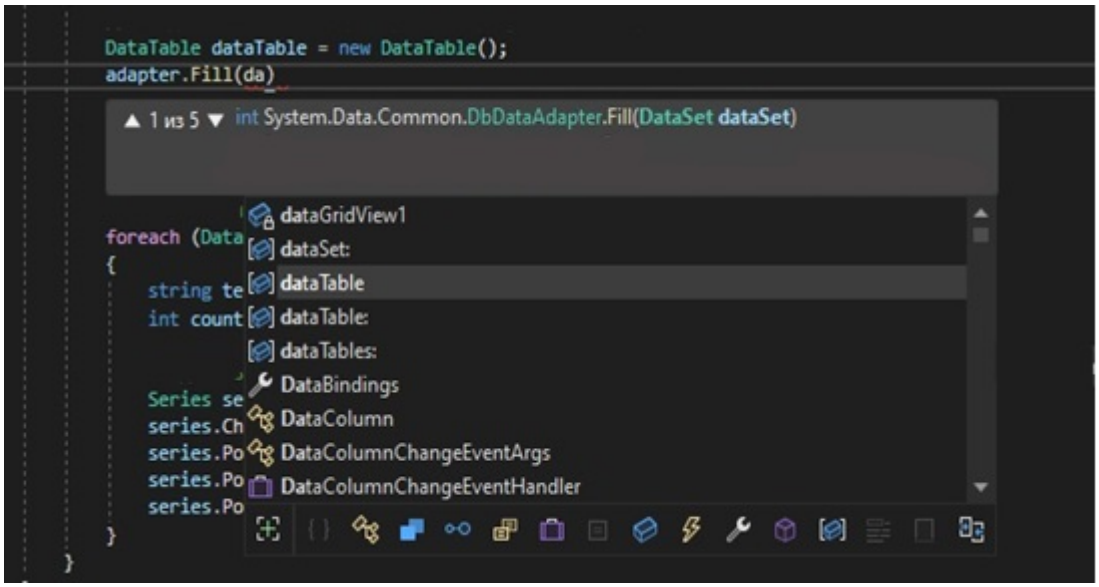


Рис. 2.3. Перевірка помилок

Форматування коду: Автоматичне форматування для підтримки внутрішнього стилю коду.

Відлагодження та тестування:

Точки зупинки: Можливість встановлювати точки зупинки в коді для відлагодження відображено на (Рис. 2.4).

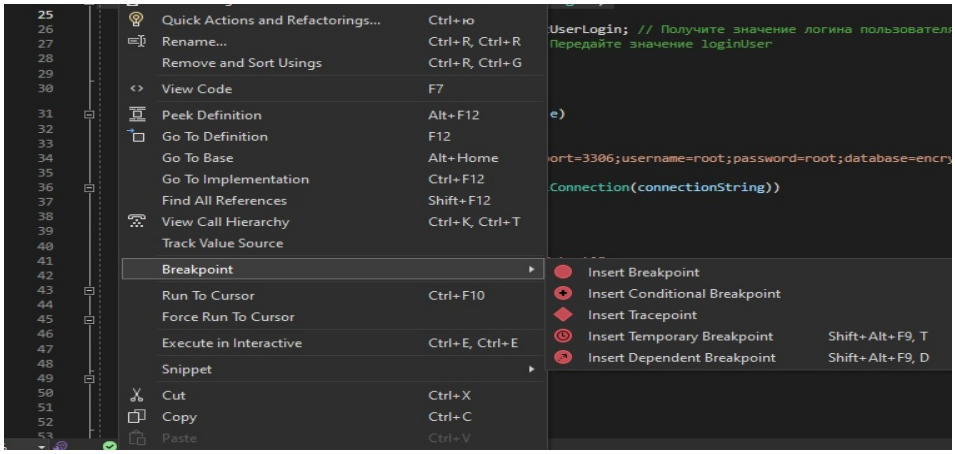


Рис 2.4. Точки зупинки

Крок-за-кроком виконання: Дозволяє виконувати код по одному кроці,

спостерігаючи за змінними та виконанням коду, це відображено на (Рис. 2.5).

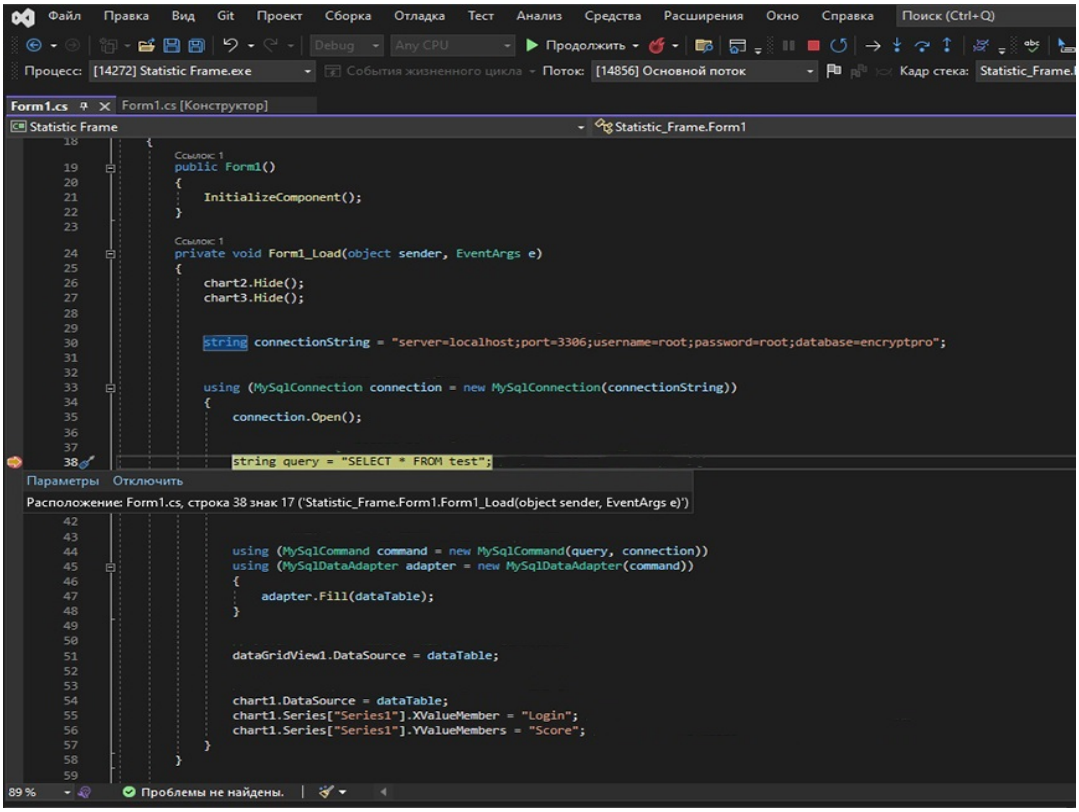


Рис. 2.5. Крок-за-кроком виконання

Юніт-тести: Інтеграція з фреймворками тестування для автоматичного тестування програмного забезпечення.

Управління проектами та версіями:

Git, TFS (Team Foundation Server): Інтегрована підтримка систем контролю версій для спільної роботи над проектами.

Керування завданнями: Інструменти для організації завдань та керування проектами.

Розробка мобільних та веб-додатків:

Xamarin: Для розробки крос-платформових мобільних додатків.

ASP.NET: Для створення веб-додатків та служб.

Розширені можливості:

Широкий вибір плагінів та розширень: Дозволяє розширити функціонал

за потребою.

Інтеграція з Azure: Для розгортання та керування хмарними послугами.

Visual Studio має кілька версій, таких як Community, Professional та Enterprise, кожна з яких має свої функції та призначення відповідно до потреб користувача.

Це потужний інструмент для програмістів, який надає широкий спектр можливостей для ефективної розробки програмного забезпечення на різних платформах та мовах програмування.

Отже, вибір Visual Studio може бути доцільним, якщо вам потрібно багатофункціональне та потужне середовище розробки з широким спектром можливостей для роботи з різними мовами програмування та платформами. Однак, в усіх випадках варто враховувати специфіку проекту та індивідуальні потреби розробника при виборі інструменту.

2.3 Інференційний аналіз

Інференційний аналіз - це метод вивчення та розуміння даних, спрямований на винесення висновків про популяцію на основі зразкових даних. Цей підхід застосовується в багатьох галузях, включаючи статистику, машинне навчання, науку про дані та дослідницькі роботи.

У статистиці, інференційний аналіз допомагає робити припущення про всю популяцію на основі обмеженого обсягу даних (вибірки). Це може включати оцінку середнього значення, розмаху, варіації або взаємозв'язків між різними змінними.

Типовий процес інференційного аналізу включає такі етапи:

Формулювання гіпотез: Ставиться певне припущення про параметри популяції на основі зразкових даних.

Збір даних: Отримання інформації, яка є предметом аналізу.

Аналіз даних: Використання статистичних методів для оцінки параметрів популяції.

Висновок: Зроблення висновків щодо гіпотези, враховуючи ступінь достовірності результатів.

Точність висновків у інференційному аналізі залежить від правильного вибору представницької вибірки, коректної статистичної методології та акуратного урахування можливих помилок.

У сфері машинного навчання, інференційний аналіз використовується для передбачення результатів на основі аналізу вхідних даних. Наприклад, він використовується для класифікації об'єктів за їх характеристиками, розпізнавання образів, аналізу тексту і т.д.

Формула для інференційного аналізу середнього значення оцінки за пройдені тести зазвичай ґрунтується на розрахунку довірчого інтервалу для середнього значення популяції, використовуючи вибіркові дані.

Довірчий інтервал для середнього можна обчислити через формулу інтервалу довіри із стандартною помилкою середнього. Ця формула зазвичай використовується для побудови довірчих інтервалів на основі середнього значення у вибірці по формулі (2.1):

$$\overline{x} \pm z \times \frac{s}{\sqrt{n}} \tag{2.1}$$

де:

$\overline{x}$ - середнє значення вибірки.

z - критичне значення для обраного рівня довіри (наприклад, для довіри на рівні 95% z буде 1,96, якщо розподіл є нормальним).

s - стандартне відхилення вибірки.

n - розмір вибірки.

Ця математична формула дозволяє оцінити, наскільки середнє значення у вибірці може відрізнятися від реального середнього значення у всій групі або популяції. Вона допомагає припускати, яке середнє значення може існувати у всій групі на основі даних, які ми маємо. Таке розуміння дозволяє робити висновки про всю групу на основі обмеженої інформації, отриманої з вибірки. [16].

В цілому, інференційний аналіз є потужним інструментом, що дозволяє здійснювати узагальнення та робити висновки про популяції на основі аналізу обмежених даних.

2.4 MAMP

MAMP (Macintosh, Apache, MySQL, PHP/Python/Perl) - це програмне забезпечення для створення локального сервера на операційній системі macOS. Це дозволяє розробникам створювати та тестувати веб-сайти на своєму комп'ютері перед їх публікацією в Інтернеті. Ось деякі плюси MAMP у порівнянні з програмами-конкурентами:

Простота використання:

Легкість налаштування: MAMP пропонує простий та швидкий спосіб налаштування веб-сервера, бази даних та інтерпретатора мов програмування.

Інтуїтивний інтерфейс: Графічний інтерфейс користувача робить його дуже зрозумілим для новачків, загальний вид програми відображено на (Рис. 2.6).

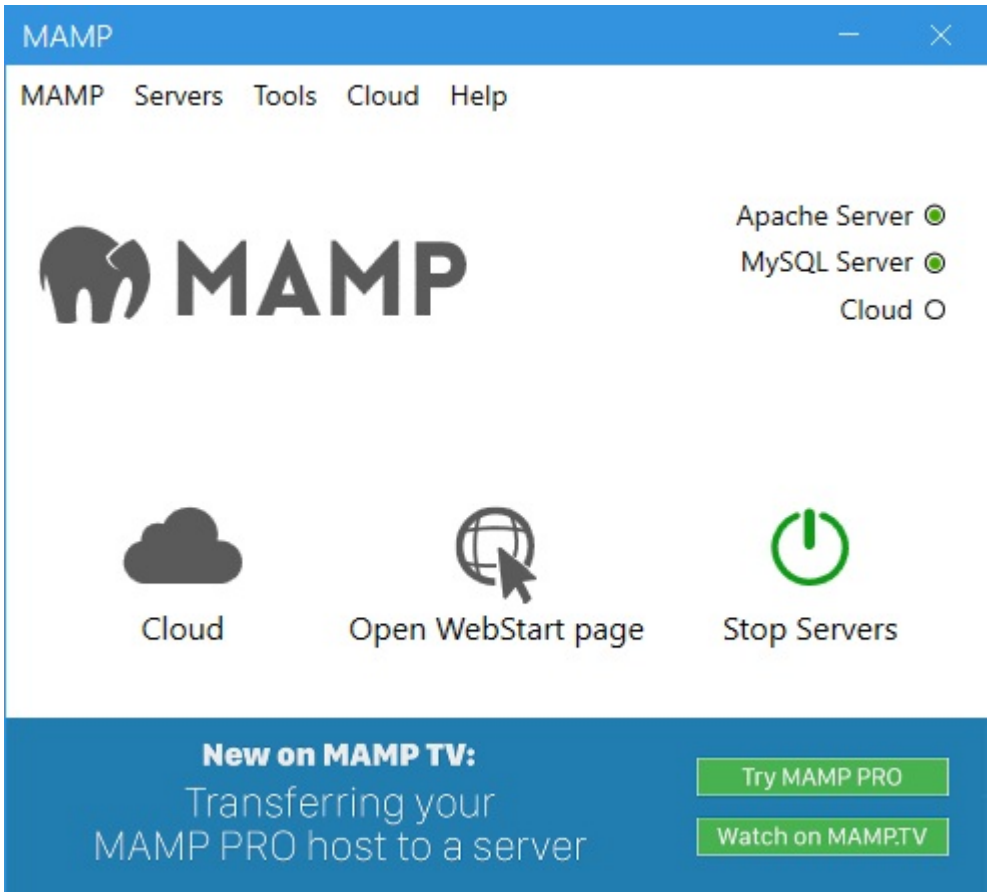


Рис. 2.6. Загальний вид програми

Повнота рішення:

Все-в-одному: MAMP включає в себе все необхідне для створення локального сервера, включаючи Apache в якості веб-сервера, MySQL або MariaDB як базу даних та PHP, Python, Perl як інтерпретатори мов програмування.

Широкий функціонал: На відміну від окремих інсталяцій окремих компонентів, MAMP надає зручний пакет зі всіма потрібними компонентами.

Гнучкість та розширюваність:

Підтримка інших мов програмування: Окрім PHP, MAMP також підтримує Python, Perl тощо, що робить його більш гнучким для програмістів з різними потребами.

Розширення за допомогою плагінів та розширень: Можливість розширення функціоналу за допомогою додаткових плагінів та компонентів.

Сумісність з macOS:

Оптимізація для macOS: MAMP розроблено спеціально для macOS, тому воно оптимізоване для цієї платформи, що може забезпечити кращу сумісність та продуктивність.

Налаштування і технічна підтримка:

Спрощена конфігурація: У MAMP більше уваги приділяється спрощеній настройці, що робить його привабливим для тих, хто не хоче глибоко розбиратися у налаштуваннях сервера [17].

MAMP має свої унікальні переваги в порівнянні з іншими програмами для створення локального сервера, особливо для користувачів macOS, завдяки своїй простоті використання та повноті функцій, які воно надає.

2.5 PHP MyAdmin

PHPMyAdmin - це веб-інтерфейс для керування базами даних MySQL або MariaDB. В контексті MAMP, PHPMyAdmin зазвичай використовується як інструмент для управління базами даних, які входять до складу локального сервера, загальний вигляд відображено на рис.2.7.

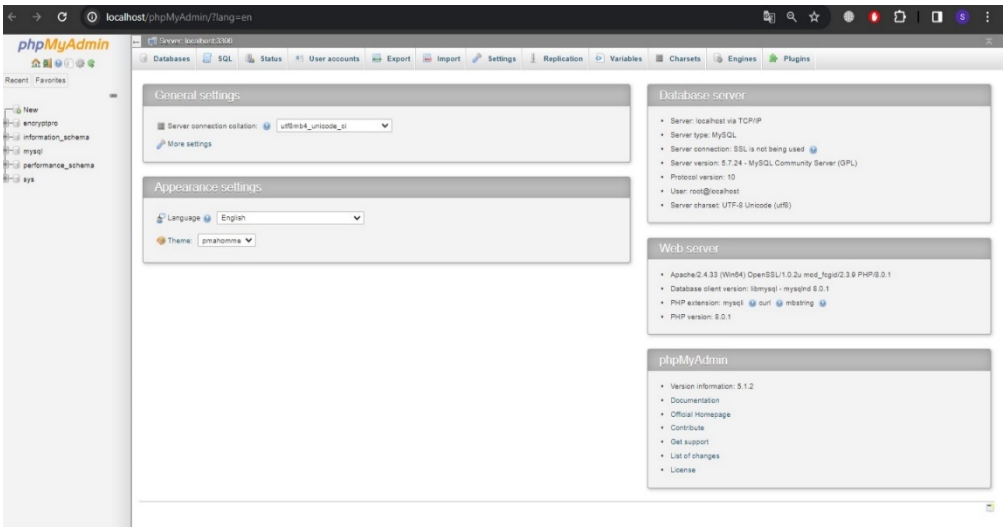


Рис. 2.7. Загальний вигляд PHP MyAdmin

Ось деякі переваги та особливості використання PHPMyAdmin у MAMP:

Графічний інтерфейс для MySQL/MariaDB:

Зручний інтерфейс: PHPMyAdmin надає інтуїтивно зрозумілий веб-інтерфейс для керування базами даних, де користувачі можуть виконувати запити, створювати, видаляти та редагувати таблиці та дані.

Функціональні можливості:

Управління базами даних: PHPMyAdmin дозволяє керувати всіма аспектами бази даних, включаючи створення, видалення та редагування таблиць, керування користувачами та їх дозволами, виконання SQL-запитів тощо.

Імпорт/експорт даних: Можливість імпорту та експорту даних у різних форматах, що робить легше переміщення та резервне копіювання інформації. Даний процес відображено на (Рис. 2.8).

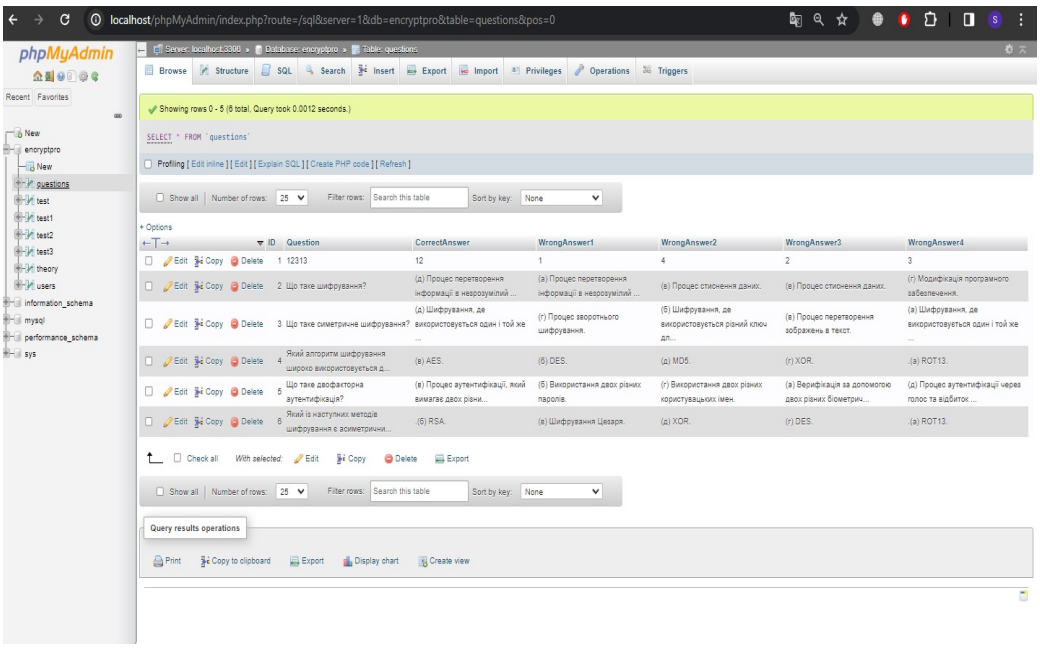


Рис. 2.8. Можливість імпорту та експорту даних

Інтеграція з MAMP:

Легка установка та доступ: Встановлення MAMP зазвичай включає в себе PHPMyAdmin, що забезпечує зручний доступ до управління базами даних, які працюють під управлінням локального сервера.

Налаштування та безпека:

Керування налаштуваннями: PHPMyAdmin дозволяє настроювати параметри баз даних, забезпечуючи більше контролю над їх роботою.

Безпека: Є можливості для забезпечення безпеки, такі як управління доступом до баз даних та введенням прав користувачів.

PHPMyAdmin у складі MAMP надає зручний спосіб керувати та адмініструвати бази даних, що використовуються локально на вашому комп'ютері. Це робить процес розробки та тестування веб-додатків більш ефективним та зручним для користувачів [18].

РОЗДІЛ 3

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ДОСЛІДЖЕННЯЕФЕКТИВНОСТІ ЙОГО ЗАСТОСУВАННЯ

3.1. Функціональне призначення системи

Розроблена програма призначена для вивчення та перевірки знань у сфері шифрування. Інтерфейс програми розділений на дві частини: у верхній є навігаційна панель, за допомогою якої користувач може обирати потрібну форму, а весь решта простір вікна займає контент, який змінюється залежно від вибраної опції на панелі навігації.

При запуску програми відкривається вікно авторизації, де користувач може увійти або створити новий обліковий запис. У програмі існує два рівні доступу: студент та викладач.

Головне меню облікового запису типу «Студент» має такі пункти:

- головна;
- теорія;
- тестування;
- методи шифрування.

Меню для користувача з роллю "Викладача" відрізняється від звичайного меню. Воно має додаткову вкладку "Налаштування". Користувач з роллю "Викладача" матиме змогу редагувати теоретичний матеріал, відкривши відповідну форму.

3.2. Опис застосованих математичних методів

Особливості області задачі передбачають використання математичних методів шифрування та кодування. При розробці підсистеми маршрутизації

запитів були застосовані наступні математичні методи:

- Шифр Цезаря;
- Шифр Тритеміуса;
- Шифр Гамування;
- DES-шифрування;
- Triple DES-шифрування;
- AES.

3.3 Проектування моделі бази даних

ER-діаграма — це вид схеми, що відображає взаємозв'язки між різними сутностями всередині системи. Зазвичай такі діаграми використовуються для розробки та оптимізації реляційних баз даних у сферах освіти, наукових досліджень, програмного забезпечення та бізнес-інформаційних систем. ER-діаграми використовують стандартний набір символів, таких як прямокутники, ромби, овали та лінії, для відображення сутностей, їх атрибутів та зв'язків. Ці діаграми відповідають певним граматичним структурам [19].

Після цього було проведено проектування бази даних і побудовано ER-діаграму для відображення цієї бази даних яка зображена на (Рис. 3.1).

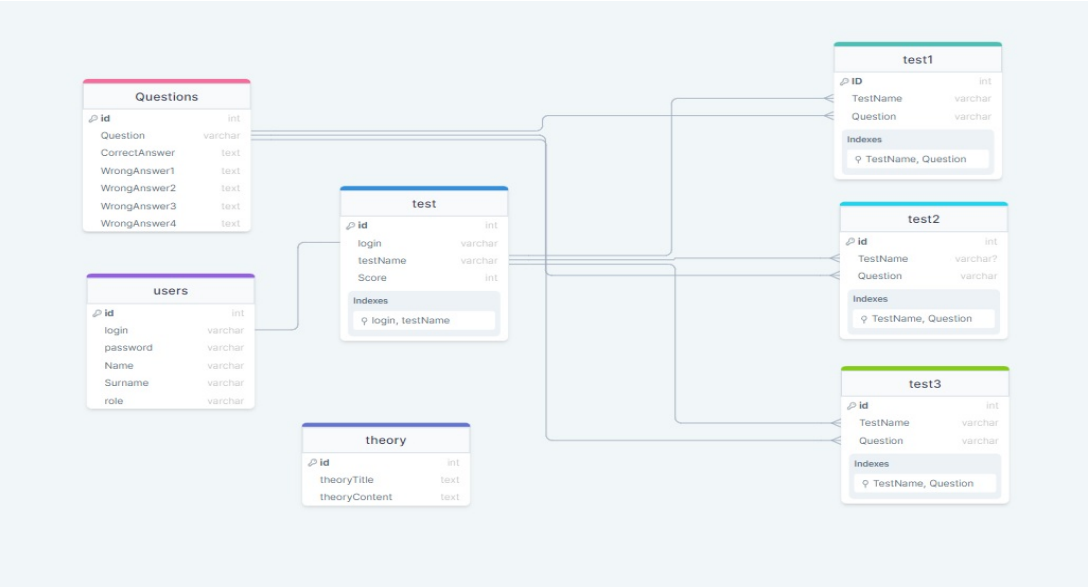


Рис. 3.1. ER-діаграма

Дана база даних містить 6 таблиць, що мають певні атрибути:

1. users – таблиця, що містить дані про користувачів програми. Таблиця містить такі атрибути: id (ідентифікатор користувача), password (пароль користувача), name (ім’я користувача), surname (прізвище користувача) та role (роль користувача);

2. Questions – таблиця, що містить дані про питання для тестів. Таблиця містить такі атрибути: id (ідентифікатор заходу), Question (запитання), CorrectAnswer (правильна відповідь), WrongAnswer1 (неправильна відповідь 1), WrongAnswer2 (неправильна відповідь 2), WrongAnswer3 (неправильна відповідь 3) WrongAnswer4 (неправильна відповідь 4);

3. test – таблиця, що містить в собі результати тестування. Таблиця містить такі атрибути: id (ідентифікатор категорії), login (логін користувача), testName (назва тесту), Score (оцінка);

4. theory – таблиця, що містить в собі теорію. Таблиця містить такі атрибути: id (ідентифікатор нагадування), theory_Title (Заголовок теорії), theory_content (Вміст самої теорії);

5. test – таблиця, що містить в собі дані про тести. Таблиця містить такі атрибути: id (ідентифікатор), TestName (назва тесту), Question (питання).

3.4. Опис структури системи та алгоритмів її функціонування

У процесі створення програми, яка взаємодіє зі студентами та викладачами, була використана база даних для зберігання необхідної інформації. Файлова структура була розділена на логічні частини, які відповідають за різні аспекти програми. Для сторінок WinForm, пов'язаних із методами шифрування, було створено окремий каталог "ECryptons", а для основних екранів — каталог "Main". Подібно було організовано інші частини файлової структури. Це дозволило зробити файлову структуру більш зрозумілою та легше відстежувати зв'язки між її частинами. У Visual Studio 2022 було створено проект з такою структурою, що має наступний вигляд (див. Рис. 3.2).

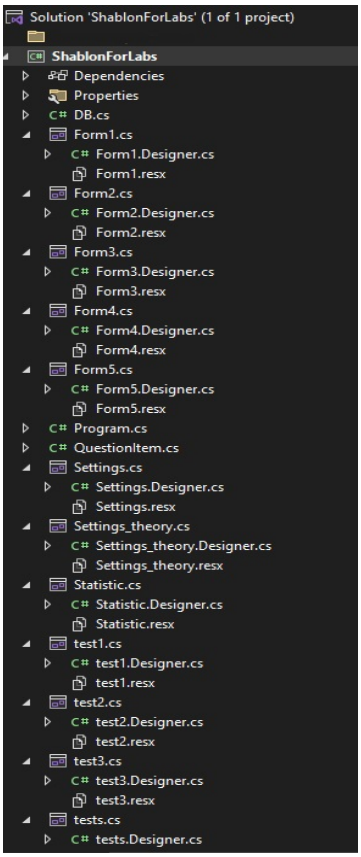


Рис. 3.2. Файлова структура програмного продукту

Вона складатиметься з таких каталогів як:

– «Data» містить файли зображень та допоміжні матеріали;

– «ECryptsons» містить сторінки WinForm, які управляють відображенням та функціонуванням методів шифрування;

– «Dialog» містить інтерфейси для відображення діалогових вікон з певною інформацією;

– «Helpboard» містить сторінки WinForms з інформацією про використання програмного застосунку;

– «Main» зберігає основні сторінки проєкту, до яких можна перейти через головне меню програми.

Аналіз та проєктування різних областей є першими кроками у створенні програмного рішення. Одним із результатів цього етапу є діаграма "Випадок використання", що описує основні групи користувачів системи та їхні можливості використання системи.

Цілі аналізу вимог та моделювання включають:

– досягнення згоди між розробниками, клієнтами та користувачами щодо функціоналу системи;

– покращення розуміння розробниками того, як система повинна працювати та які функції вона повинна виконувати;

– обмеження функціоналу системи для чіткого визначення її меж та завдань;

– створення базового плану для наступних кроків у проєкті;

– визначення інтерфейсу користувача для забезпечення зручного та зрозумілого взаємодії з системою.

Use-case діаграми використовуються для моделювання різних видів діяльності, які виконує організація, а також для уявлення функціональних вимог до програмного забезпечення під час його проєктування та розробки. Ці діаграми доповнюються текстовим описом, що відображає ієрархічну структуру вимог за допомогою сценаріїв використання [8]. У звіті зображено

діаграми компонентів, використання та ER-діаграму на (Рис. 3.3. – 3.5).

Рис. 3.3. Діаграма використання клієнтської частини

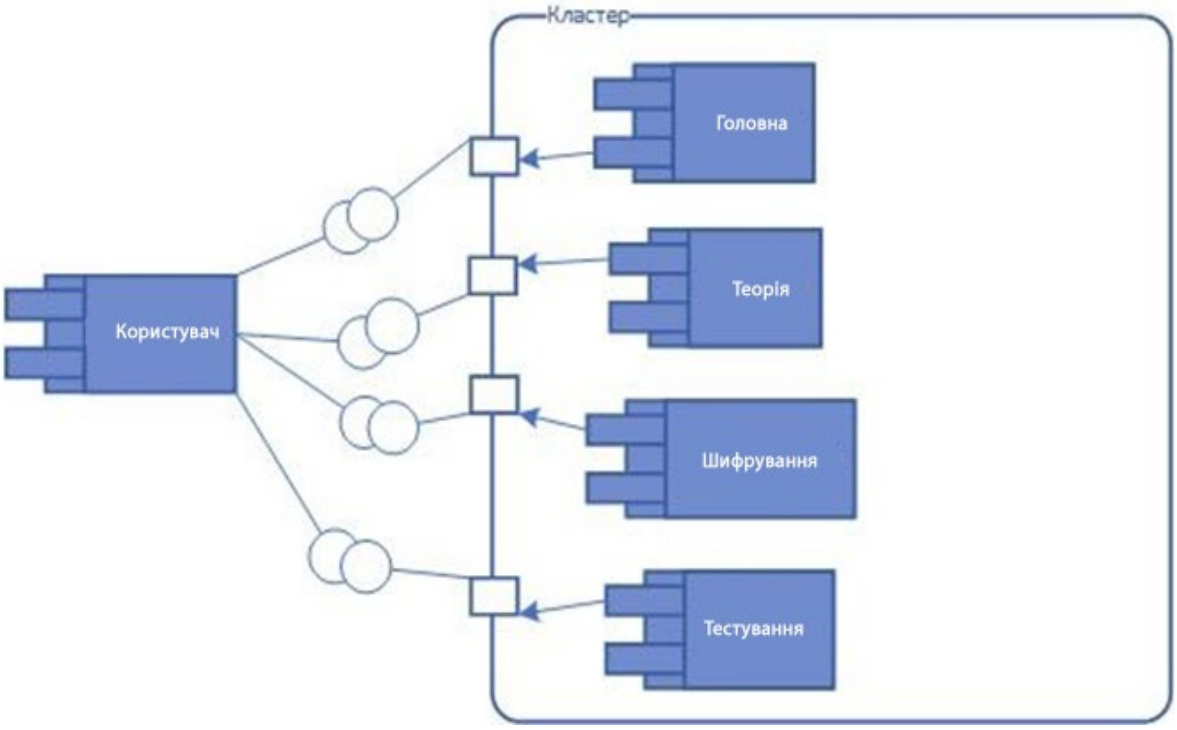


Рис. 3.4. Діаграма компонентів

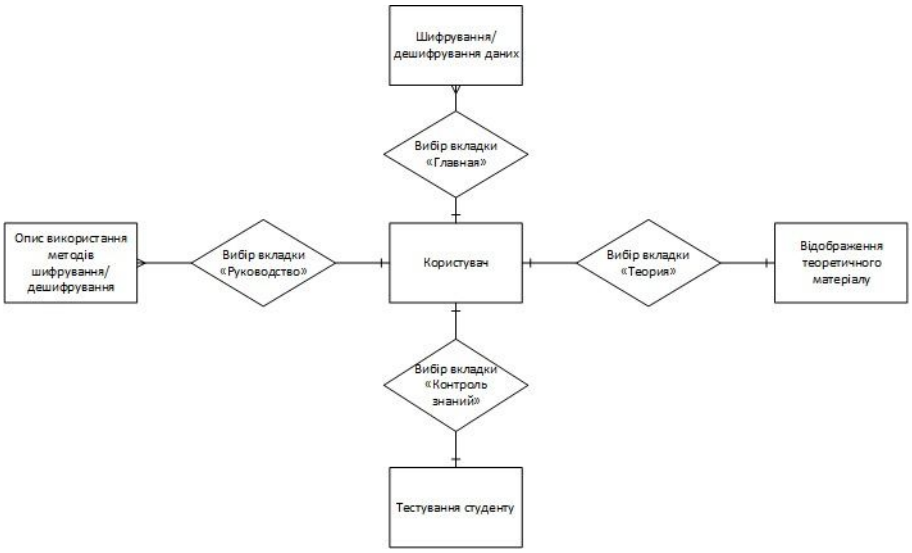


Рис. 3.5. ER-діаграма

Для спроєктованого додатку було розроблено діаграмму використання (див. Рис. 3.6).

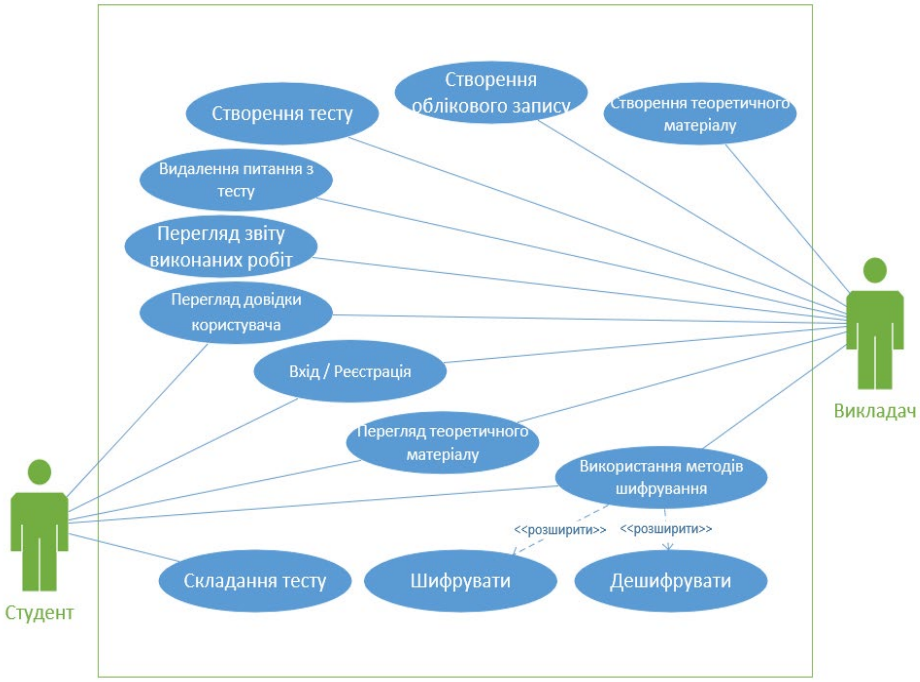


Рис. 3.6. Діаграма дій користувача (використання)

Програма, що була спроєктована, включає два види вікон: вікно для авторизації та основне вікно програми. Вони обидва мають панель навігації у верхній частині. Ця програма побудована так, що для відображення будь-якої

потрібної сторінки використовується тільки одне вікно. Перемикання між вкладками панелі навігації не змінює основне вікно, лише вміст відображення відповідного модулю. Це дозволяє швидше завантажувати обрані сторінки. Макет основного вікна має такий вигляд, як показано на (Рис. 3.7).

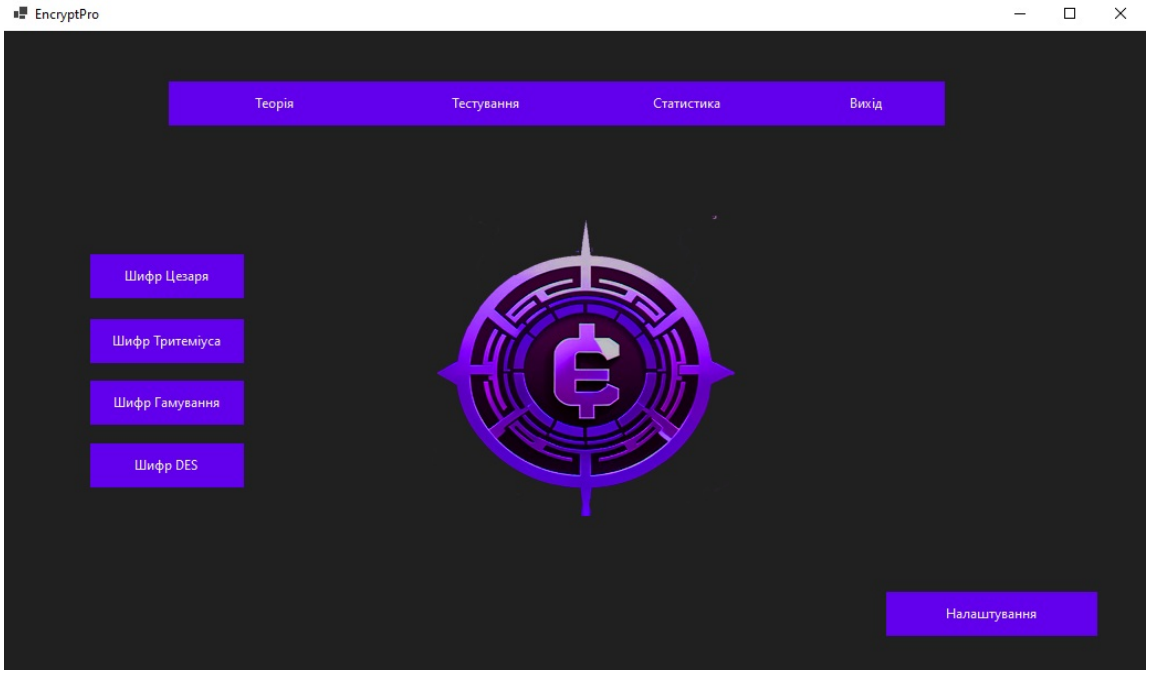


Рис. 3.7. Макет головного вікна додатку

У верхній частині головного вікна розташована панель навігації. Для облікових записів рівня "Студент" на панелі навігації є вкладки: "Головна", "Теорія", "Тест", "Довідка користувача" (див. рис. 3.8). Якщо обліковий запис належить до типу "Викладач", то панель навігації містить вкладки: "Головна", "Теорія", "Панель адміністратора", "Довідка користувача", "Звіт" (див. Рис. 3.9).

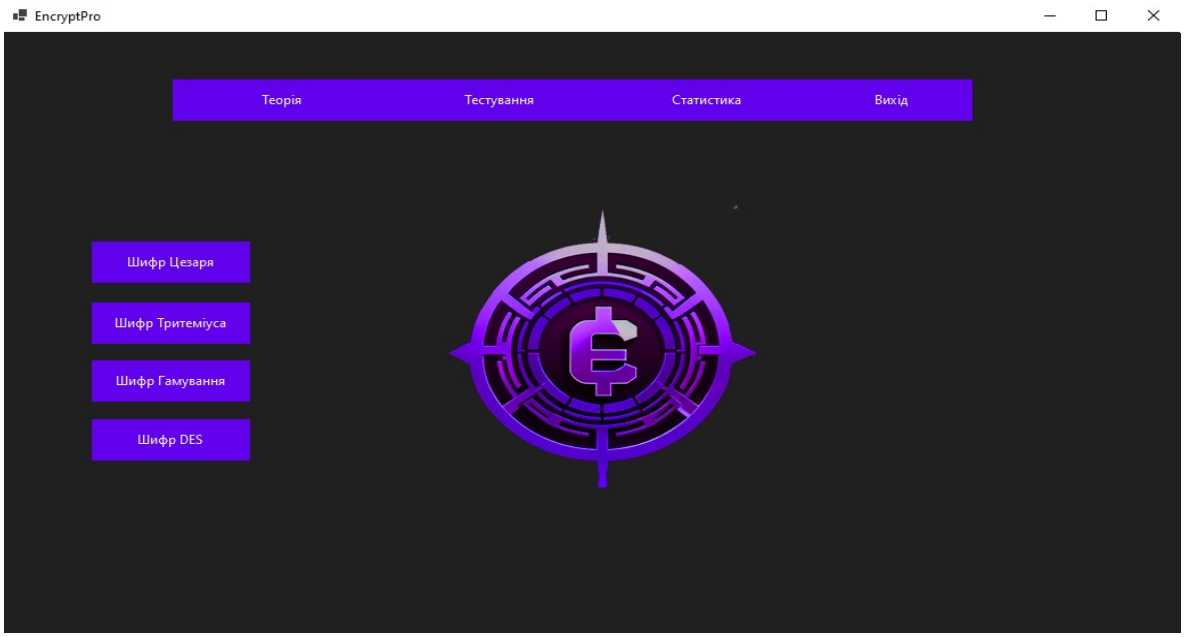


Рис. 3.8. Макет форми авторизації рівня «Студент»

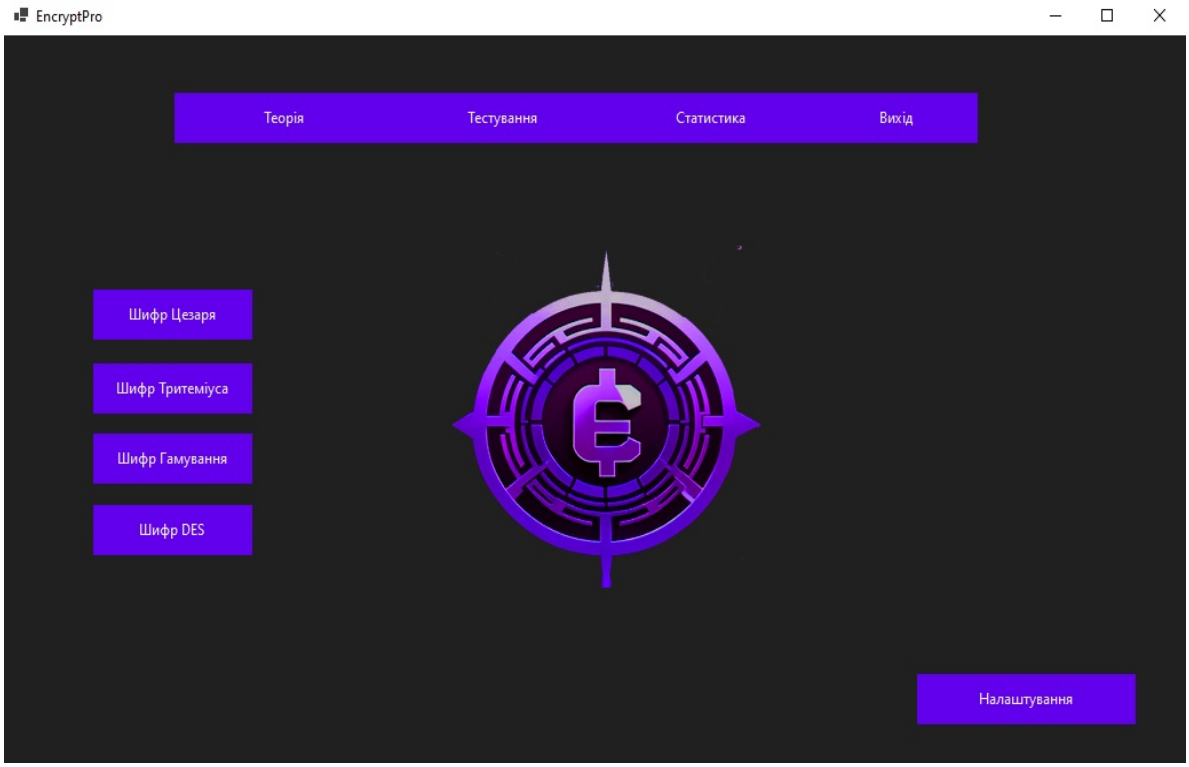


Рис. 3.9. Макет форми авторизації рівня «Викладач»

Після запуску програми відображається сторінка авторизації, яка надає можливість користувачеві увійти до свого облікового запису або створити новий. Макет цієї сторінки містить поля для введення особистих даних, необхідних для входу, а також відповідні кнопки (див. Рис. 3.10).

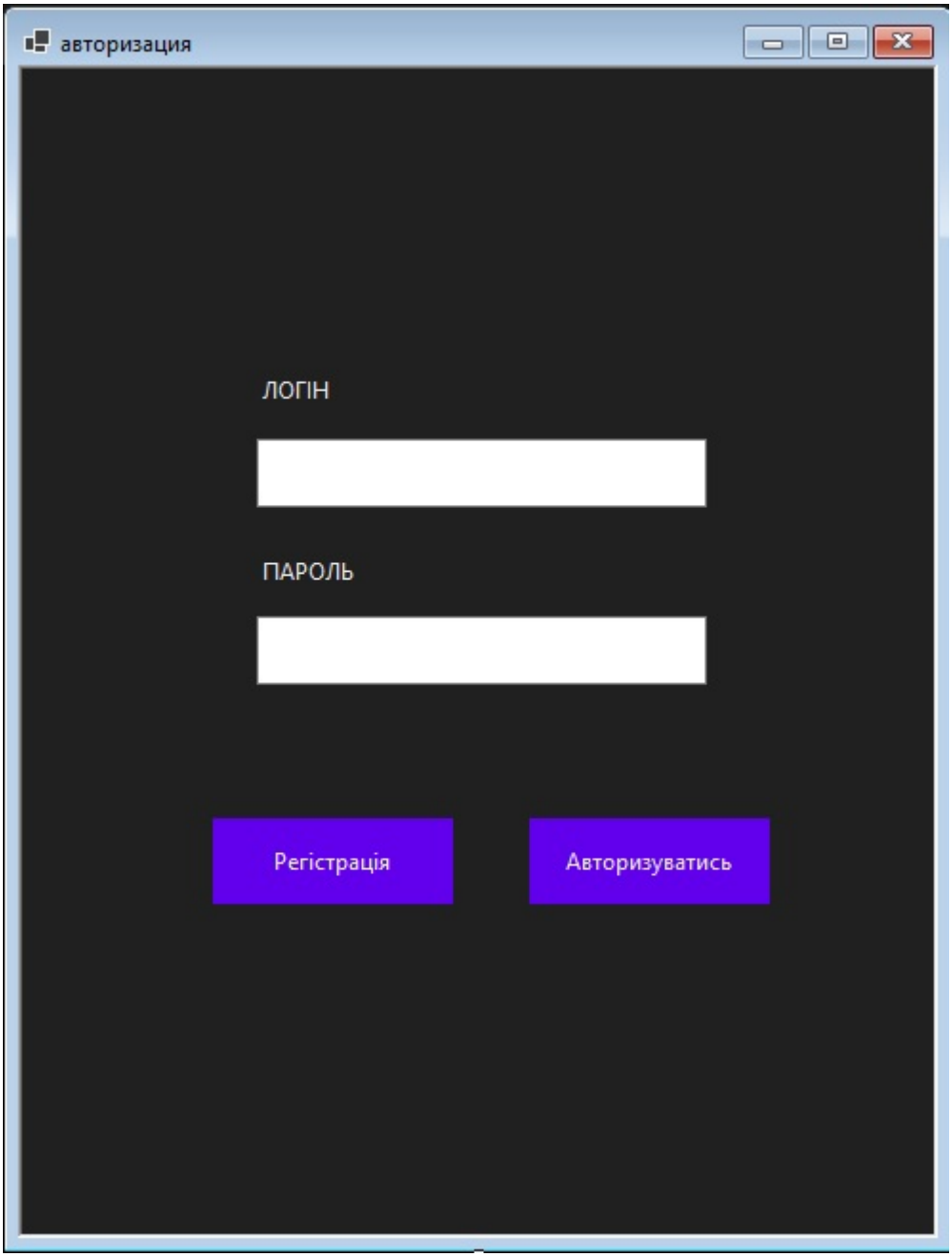


Рис. 3.10. Макет форми авторизації

Розроблюваний проєкт має такі форми:

– Form2.cs – відображає головне вікно, та обраний вміст вікна.
– Авторизація.cs – відображає вікно авторизації з обраним вмістом.
– Form1.cs – сторінка що відображає шифрування методом Цезаря.
– Form3.cs – сторінка що відображає шифрування методом Тритеміуса.
– Form4.cs – сторінка що відображає шифрування методом Гамування.

– Form5.cs – сторінка що відображає шифрування методом DES, Triple DES, AES.

– Регістрація.cs – відображає сторінку введення даних необхідних для реєстрації.

– Statistics.cs – відображає сторінку з даними.

– Theory.cs – відображає сторінку з теоретичним матеріалом.

– Tests.cs – відображає вікно де знаходяться всі тести.

– Test№.cs – відображає сторінку з тестом для контролю знанн.

– Settings.cs – відображає сторінку що містить поля налаштування даних облікового запису.

Головне вікно програми відкривається одразу після успішної авторизації, та має наступний вигляд (див. Рис. 3.11). Призначення компонентів та методів головного вікна описані в таблицях 3.1 – 3.2.

Таблиця 3.1

Компоненти головного вікна

Тип компоненту	Назва компоненту	Призначення
Button	button1	Виходить з облікового запису повертаючись до вікна Авторизації
Button	button2	Завершує роботу программи
Button	button3	Мінімізує головне вікно
Button	button1	Відкриває форму зі статистичними даними
Button	button2	Завантажує форму з шифруванням методом Цезаря.
Button	button3	Завантажує форму з шифруванням методом Тритеміуса.
Button	button4	Завантажує форму з шифруванням методом Гамування.
Button	button5	Завантажує форму з шифруванням методом DES.

Продовження таблиці 3.1

Button	button6	Відкриває форму зі тестами для перевірки знань
Button	button7	Закриває додаток
Button	button8	Відкриває форму з теорією
PictureBox	PictureBox1	Відображає логотип додатку

Таблиця 3.2

Методи головного вікна

Назва методу	Призначення
MoveForm(object sender, MouseButtonEventArgs e)	Функція змінення положення форми
void LogOut()	Припинити використання поточного облікового запису і повернутися до екрану авторизації.
MinWindow()	Виконати реєстрацію або авторизацію облікового запису.
void OpenTheory()	Змінити вміст вікна на сторінку Theory
void OpenSettings()	Змінити вміст вікна на сторінку Settings
void OpenTest()	Змінити вміст вікна на сторінку Tests
void OpenStat()	Змінити вміст вікна на сторінку Statistics
void OpenCrypt()	Змінити вміст вікна на сторінку ECryptons
Button_Click(object sender, RoutedEventArgs e)	Завершення роботи додатку

Зміст головного вікна змінюється залежно від обраної сторінки. Якщо вибрано вкладку "Головна" на панелі навігації, у вікні буде показано методи шифрування, які можна спробувати у програмі, як зображено на (Рис. 3.11).

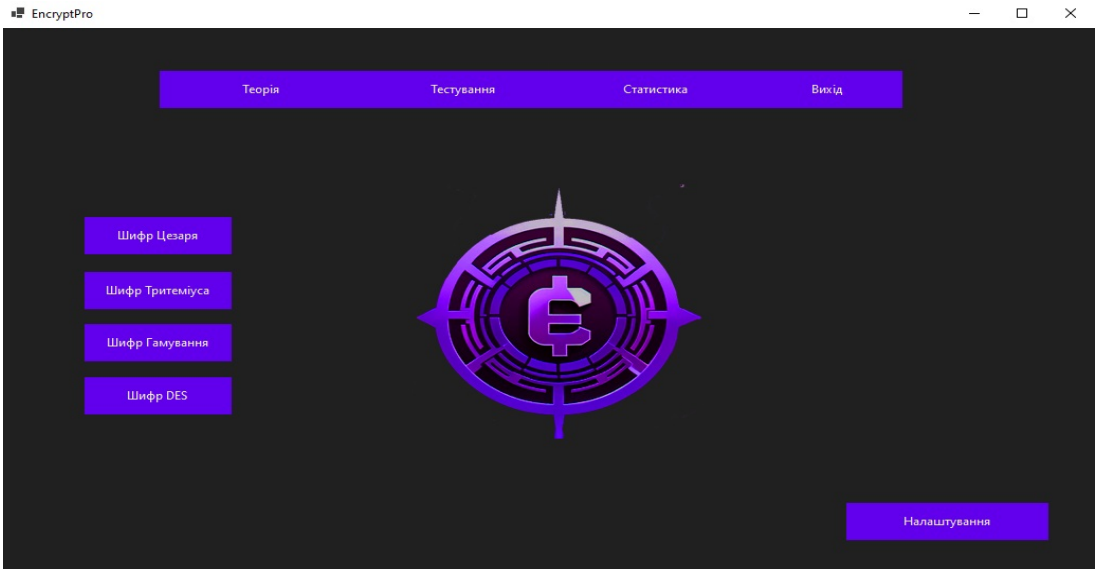


Рис. 3.11. Відображення методів шифрування на вкладці “Головна”.

У лівій частині сторінки, знаходяться кнопки керування, за допомогою яких користувач може обрати необхідний метод шифрування. Натиснувши на одну з кнопок, буде відображено вміст обраного методу. Методи описані в таблиці 3.3.

Таблиця 3.3

Методи які використовуються в шифруванні

Назва методу	Призначення
MoveForm(object sender, MouseButtonEventArgs e)	Функція змінення положення форми
Button_Click(object sender, RoutedEventArgs e)	Завершення роботи додатку
MinWindow()	Виконати реєстрацію або авторизацію облікового запису.
void OpenCrypt()	Змінити вміст вікна на сторінку ECryptons
void OpenTheory()	Змінити вміст вікна на сторінку Theory
void OpenTest()	Змінити вміст вікна на сторінку Tests
void OpenStat()	Змінити вміст вікна на сторінку Statistics
void LogOut()	Припинити використання поточного облікового запису і повернутися до екрану авторизації.

Для перегляду теоретичного матеріалу використовується форма «theory». Макет форми з теоретичним матеріалом зображено на (Рис.3.12).

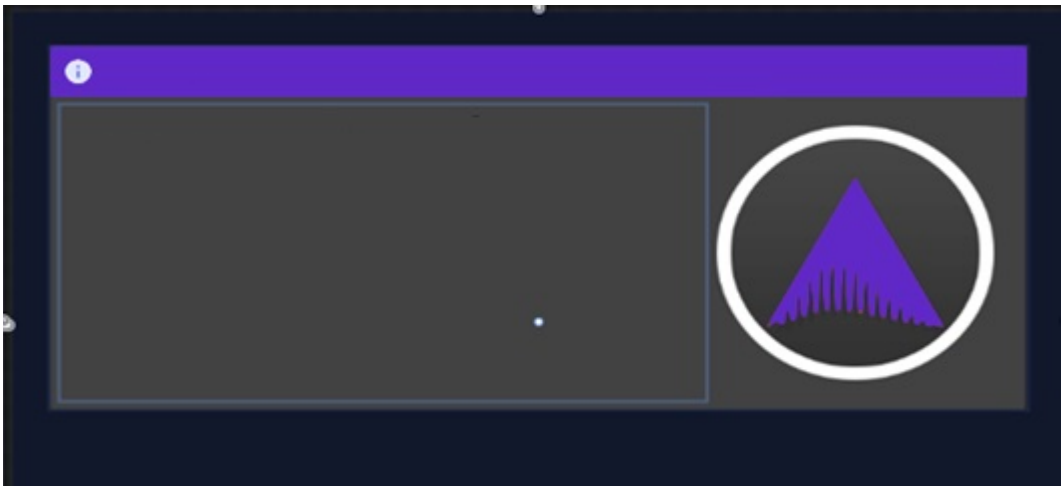


Рис. 3.12. Макет форми Theory

Сторінка «Theory» містить лише один компонент – StackPanel, до якого під час виконанання програми завантажуються дані з бази даних. За допомогою створенної функції, дані що надходять у текстовому вигляді перетворюються у View-об'єкт, який потім додається до стек панелі.

Обравши на панелі навігації вкладку «Налаштування» до головного вікна завантажується вміст форми «Settings». Вкладка поділена на блоки, де кожен блок відповідає за певну функціональну можливість. Макет сторінки «Settings» зображено на (Рис. 3.13), компоненти та методи сторінки описані в таблицях 3.4 – 3.5.

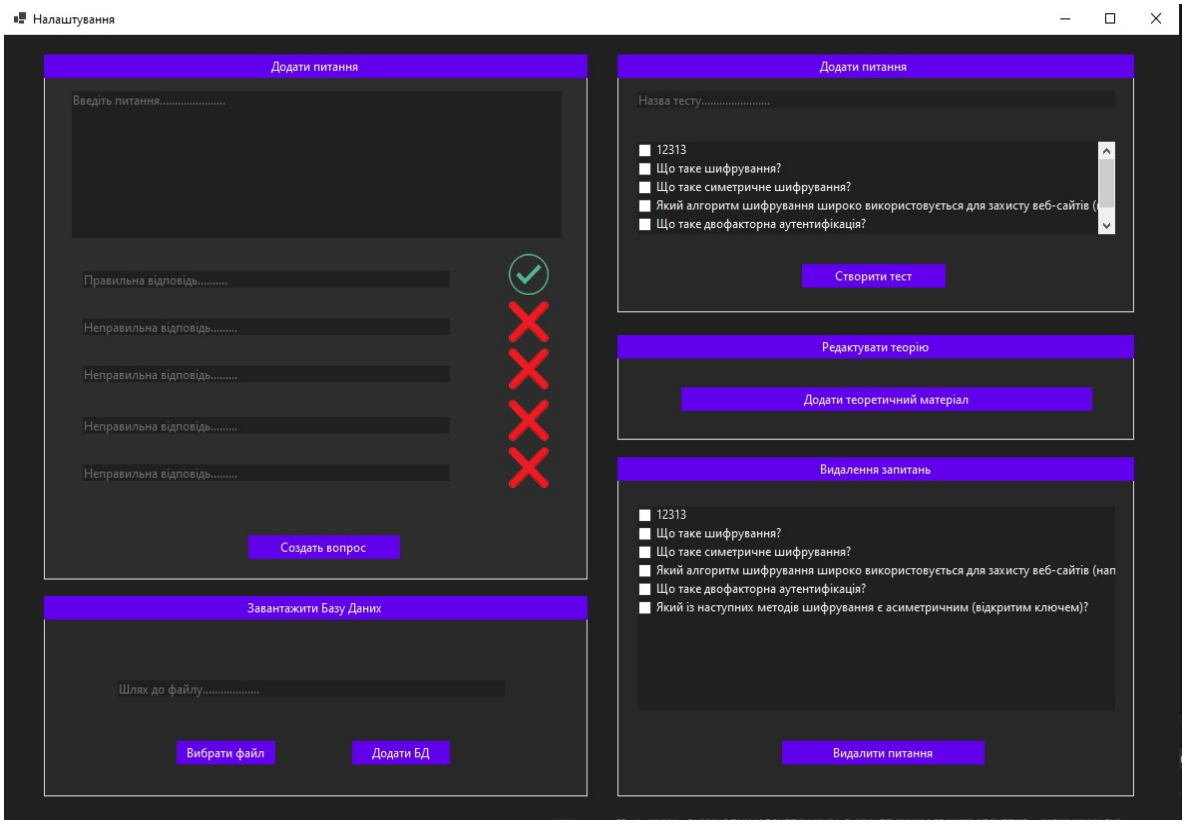


Рис. 3.13. Макет форми Settings

Таблиця 3.4

Компоненти сторінки «Settings»

Тип компоненту	Назва компоненту	Призначення
GroupBox	groupBox1	Містить в собі компоненти для додавання питань для тестів
GroupBox	groupBox2	Містить в собі компоненти для завантаження БД
GroupBox	groupBox3	Містить в собі компоненти вибору питань для тесту
GroupBox	groupBox4	Містить в собі компоненти для видалення питань з БД
TextBox	txtQuestion	Поле вооду питання
TextBox	txtCorrectAnswer	Поле вводу правильної відповіді
TextBox	txtWrongAnswer1	Поле вводу не правильної відповіді 1
TextBox	txtWrongAnswer2	Поле вводу не правильної відповіді 2
TextBox	txtWrongAnswer3	Поле вводу не правильної відповіді 3
TextBox	txtWrongAnswer4	Поле вводу не правильної відповіді 4

Продовження таблиці 3.4

TextBox	textBox2	Поле вводу місцезнаходження бази даних
TextBox	textBox1	Поле вводу назви тесту
CheckedListBox	CheckedListBox2	Поле вибору питань які будуть в тесті
CheckedListBox	CheckedListBox1	Поле вибору питань які будуть видалені з БД
Button	Button1	Зберігає питання до БД
Button	Button5	Обирає файл БД з ПК
Button	Button4	Додає файл БД до БД додатку
Button	ButtonCreateTest	Створює тест та таблицю БД з вибраними питаннями
Button	Button2	Відкриває форму додання теоретичного матеріалу
Button	Button3	Видалення обраних питаннь з БД

Таблиця 3.5

Методи сторінки «Setting»

Назва методу	Призначення
string SelectQuests ()	Виділяє всі пункти у списку запитань.
void ClearQuests()	Очищає поля для введення даних.
void Select(string tag)	Вибирає правильну відповідь на питання.
void SaveQuestions(object sender, RoutedEventArgs e)	Записує питання до бази даних.
void OpenDB(object sender, RoutedEventArgs e)	Відкриває вікно вибору місця зберігання бази даних.
void SaveFolderDatabase(object sender, RoutedEventArgs e)	Модифікує налаштування місця зберігання файлу бази даних.
void DelQuest(object sender, RoutedEventArgs e)	Видаляє питання із бази даних.
void loadQuestions()	Завантажує список питань із бази даних.
void createQuest (string i)	Додає питання до таблиці.
void OpenEditTheory()	Відкриває форму для редагування теорії.

Для кожного методу шифрування створено окремі форми, які завантажуються при специфічних подіях. Існують різні типи форм для методів шифрування: шифр Цезаря, шифр Гамування, шифр Тритеміуса та шифр DES.

Компоненти та методи сторінок звичайного шифрування перераховані у таблицях 3.6 – 3.7, а макет зображено на (Рис. 3.14).

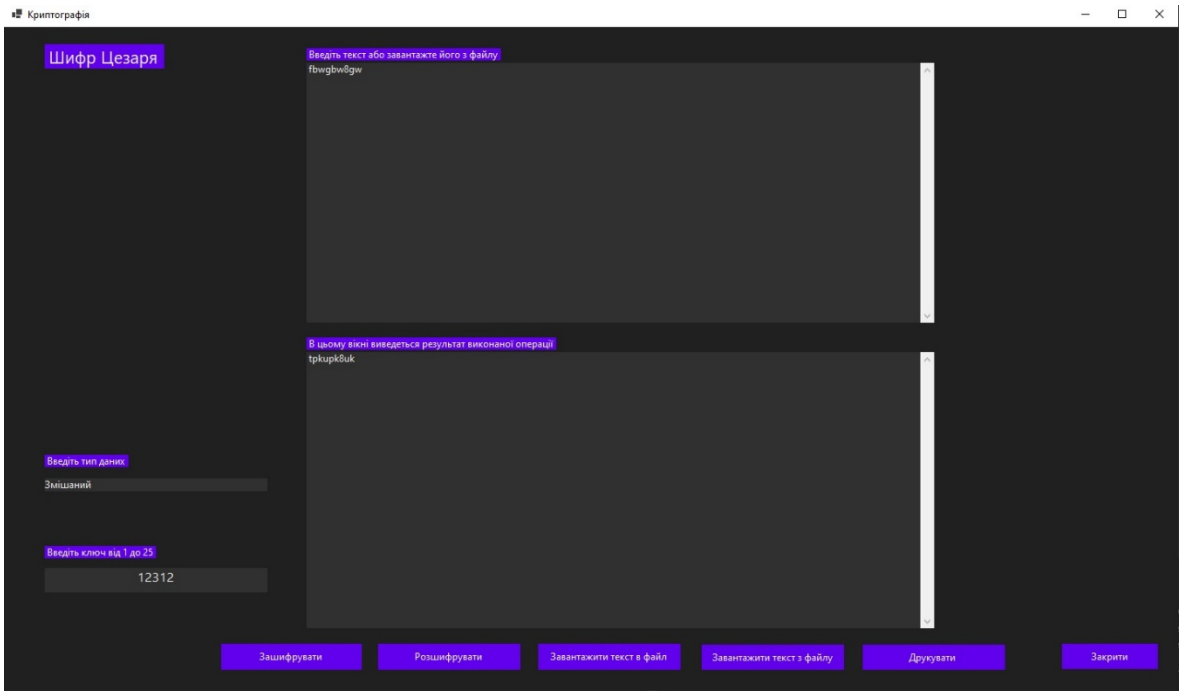


Рис. 3.14. Макет форми шифру Цезаря

Таблиця 3.6

Компоненти сторінки шифру Цезаря

Тип компоненту	Назва компоненту	Призначення
TextBox	textBox1	Поле вводу тексту
TextBox	textBox2	Поле виведення результату роботи методу
TextBox	textBox3	Поле вводу ключа шифрування
TextBox	textBox4	Поле вводу типу даних
Button	button1	Шифрує вхідні дані
Button	button2	Дешифрує вхідні дані
Button	button3	Зберігає резльтат виконання до файлу

Продовження таблиці 3.6

Button	button4	Завантажує з файлу вхідні дані
Button	button5	Закриває форму
Button	button6	Відправляє результат на друкування

Таблиця 3.7

Методи сторінки звичайного шифрування

Назва методу	Призначення
void ButtonClick1 (object sender, RoutedEventArgs e)	Шифрування даних
void ButtonClick2 (object sender, RoutedEventArgs e)	Дешифрування даних
void ButtonClick3 (object sender, RoutedEventArgs e)	Зберігає резльтат виконання до файлу
void ButtonClick4 (object sender, RoutedEventArgs e)	Завантажує з файлу вхідні дані
void ButtonClick5 (object sender, RoutedEventArgs e)	Закриває форму
void ButtonClick6 (object sender, RoutedEventArgs e)	Відправляє результат на друкування

Для огляду виконаних завдань викладачем, був розроблений шаблон сторінки під назвою "Statistics" (див. Рис. 3.15). Деталі та способи функціонування елементів сторінки " Statistics " наведено у таблицях 3.8 - 3.9.

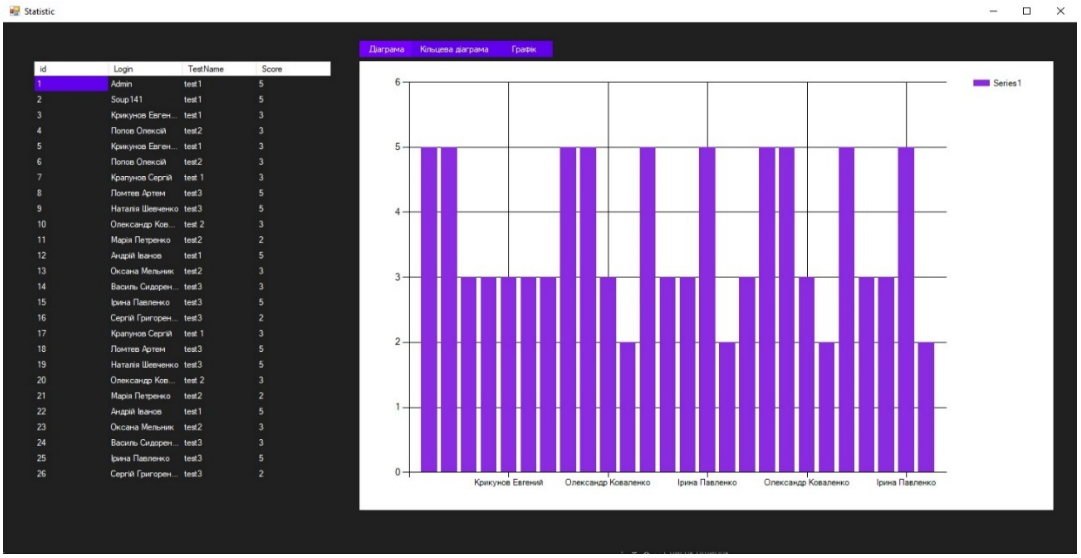


Рис. 3.15. Макет форми «Statistics»

Таблиця 3.8

Компоненти сторінки «Statistics»

Тип компоненту	Назва компоненту	Призначення
DataGrid	dataGridView1	Поле відображення інформації
Chart	chart1	Поле відображення графіків
Button	button1	Відображає діаграму
Button	button2	Відображає кільцеву діаграму
Button	button3	Відображає графік
TextBox	userNum	Поле введення фільтру за прізвищем студента
TextBox	userLogin	Поле введення фільтру за логіном студента
TextBox	userScore	Поле введення фільтру за оцінкою студента

Таблиця 3.9

Методи сторінки «Statistics»

Назва методу	Призначення
void LoadChartData()	Завантажує дані про студента та його роботи з БД
void Update()	Застосовує фільтрування до вмісту таблиці.

Продовження таблиці 3.9

void Button_Click1()	Відкриває “Графік”
void Button_Click2()	Відкриває “Кільцеву діаграму”
void Button_Click3()	Відкриває “Діаграму”
void Changing()	Оновлює дані таблиці

Для активної взаємодії з користувачем створено спеціальні вікна, які відповідають стилю програми і замінюють стандартні вікна. Кожне з цих вікон має власне зображення, відповідне до типу діалогу, кнопку для підтвердження операції та текстове поле, де виводиться потрібне повідомлення.

Усі вікна мають однакову структуру, відрізняючись лише зображенням. Макет цих вікон зображено на (Рис. 3.16).

Рис. 3.16. Діалогове вікно помилки

3.5. Обгрунтування та організація вхідних та вихідних даних програми

Програма отримує вхідні дані, включаючи інформацію про користувачів, зокрема студентів, а також текстові повідомлення у зашифрованому або незашифрованому вигляді. Результатом роботи програми є список студентів, які успішно або не успішно склали тест, а також зашифровані або розшифровані текстові повідомлення, залежно від обраної операції.

3.6. Опис роботи розробленої системи

3.6.1. Використані технічні засоби

Для коректної роботи програмного застосунку достатньо мати наступну конфігурацію комп`ютера:

- материнська плата – ASUS H110M-CS;
- процесор – Intel Core i5-6600k;
- оперативна пам’ять – HyperX Fury DDR4-3222 MHz 16gb;
- жорсткий диск – WD 1 TB;
- відеокарта Nvidia Geforce 1060 6GB .

3.6.2. Використані програмні засоби

Програму створено в середовищі програмування Microsoft Visual Studio 2022 мовою C# з використанням технології WinForms.

3.6.3. Виклик та завантаження програми

Перед вікриттям програмного додатку, його необхідно встановити на ПК, запустивши файл інсталяції «EncryptPro.exe». Після встановлення додатку, на робочому столі з’явитсься ярлик, натиснувши на який відкривається программа «EncryptPro».

3.6.4. Опис інтерфейсу користувача

При запуску програмного застосунку з'являється вікно авторизації, де користувач може увійти до свого акаунту, заповнивши поля (див. Рис. 3.17). Потрібно ввести логін та пароль, а потім натиснути кнопку "Авторизуватись". Після цього програма перевіряє введений логін. Якщо такий логін існує у базі

даних, вона порівнює введений пароль з паролем, збереженим для цього облікового запису. Якщо паролі співпадають, користувач отримує доступ до головного вікна. У протилежному випадку з'являється діалогове вікно з повідомленням про некоректність введених даних.

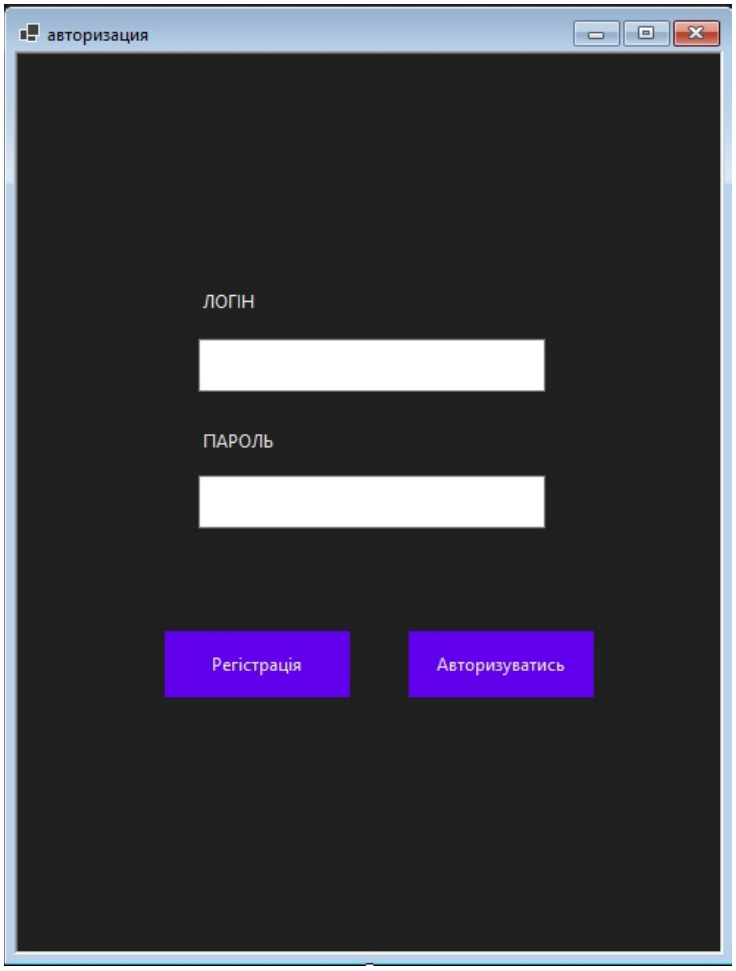


Рис. 3.17. Вікно авторизації користувача

При відсутності облікового запису користувача надається можливість створити новий, перейшовши до вікна реєстрації після натискання кнопки "Зареєструватись". Для створення акаунту потрібно заповнити всі поля вводу й натиснути кнопку "Реєстрація".

Під час реєстрації програма перевіряє можливість додавання запису до бази даних, спробуючи переконатися, що обраний логін ще не використовується. Якщо такий запис вже існує у базі даних, користувач буде попереджений про помилку (див. Рис. 3.18). Приклад заповненого вікна

реєстрації показано на (Рис. 3.19).

Рис. 3.18. Діалогове вікно попередження

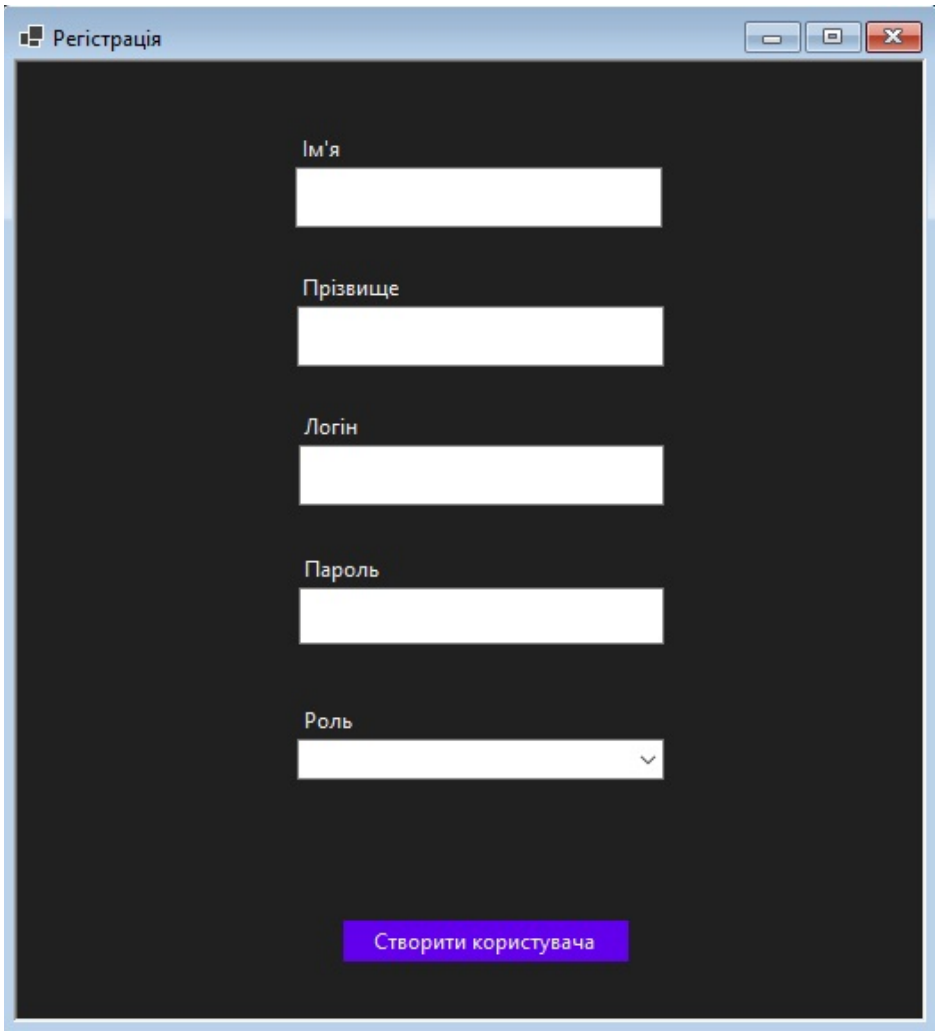


Рис. 3.19. Вікно реєстрації нового акаунту

Після створення нового облікового запису активізується вікно авторизації, де користувач може увійти за допомогою щойно створеного акаунту. У випадку ненавмисного натискання кнопки "Реєстрація" і потреби

повернутися до попереднього вікна, користувач може натиснути кнопку "Авторизація", розташовану у верхній частині вікна.

Після успішної авторизації користувача з'являється головне вікно програми, у верхній частині якого розміщена панель навігації (див. Рис. 3.20).

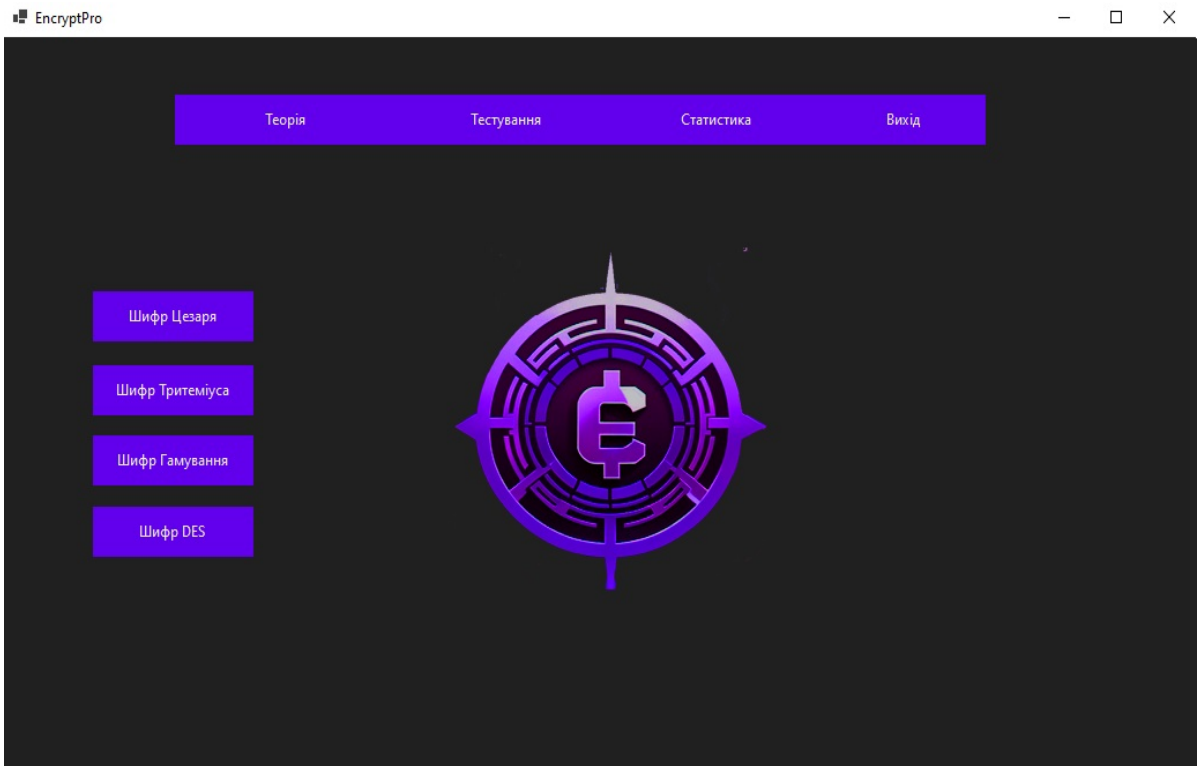


Рис. 3.20. Головне вікно програми

В навігаційній панелі є такі вкладки: "Головна", "Теорія", "Тестування".

На вкладці "Головна" знаходяться різні методи шифрування. Користувач може вибрати необхідний метод та виконати відповідну операцію, заповнивши відповідні поля (див. Рис. 3.21).

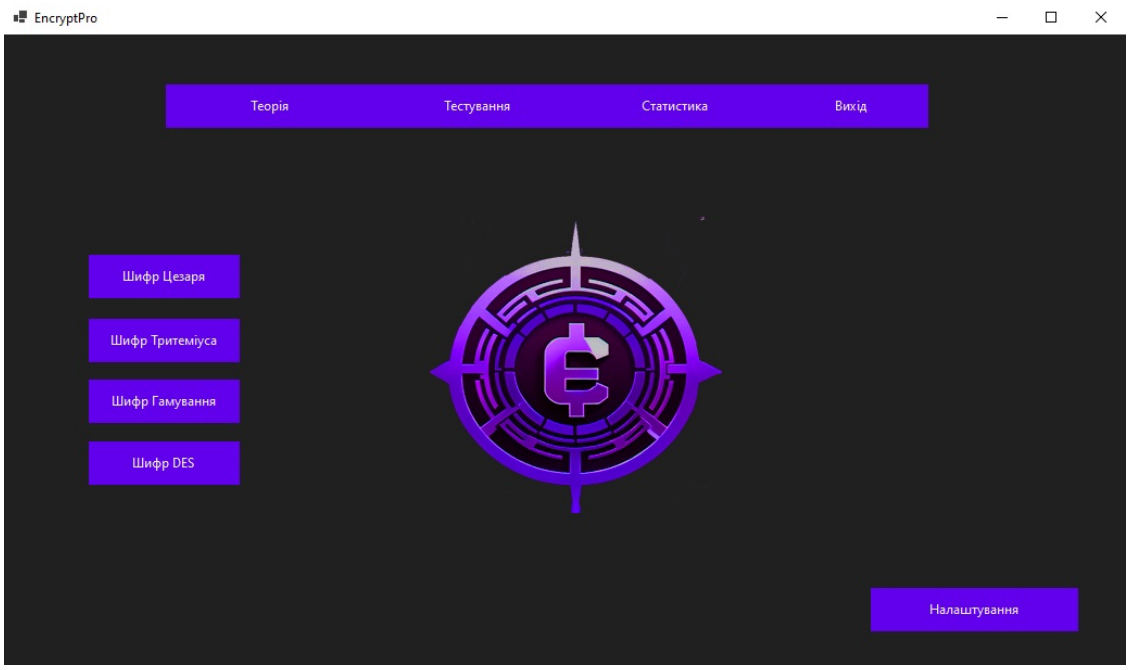


Рис. 3.21. Вкладка «Головна»

У програмі передбачена можливість як шифрувати, так і дешифрувати введений текст за допомогою відповідних кнопок. Користувач повинен ввести вхідні дані у відповідне поле (див. Рис. 3.22).

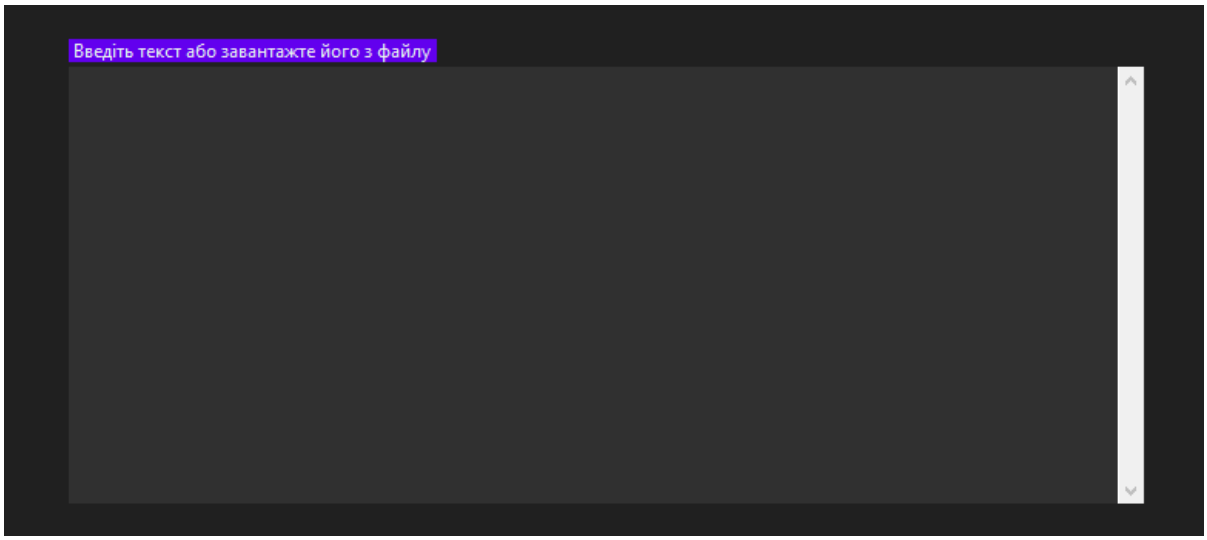


Рис. 3.22. Поле введеня текстових даних

У певних методах шифрування необхідно вказати ключ у відповідному полі. Якщо ключ не було введено, у цьому полі відображається підказка (див. Рис. 3.23)

Рис. 3.23. Вкладка, що має поле воду ключа

Коли всі необхідні поля заповнені, користувач може зашифрувати дані, натиснувши кнопку "Зашифрувати". Для дешифрування даних слід натиснути кнопку "Розшифрувати". Крім того, якщо у користувача є інформація, яку треба зашифрувати або розшифрувати у файлі, кнопка "Завантажити з файлу" надає можливість завантажити дані з файлу й отримати бажаний результат (див. Рис. 3.24).

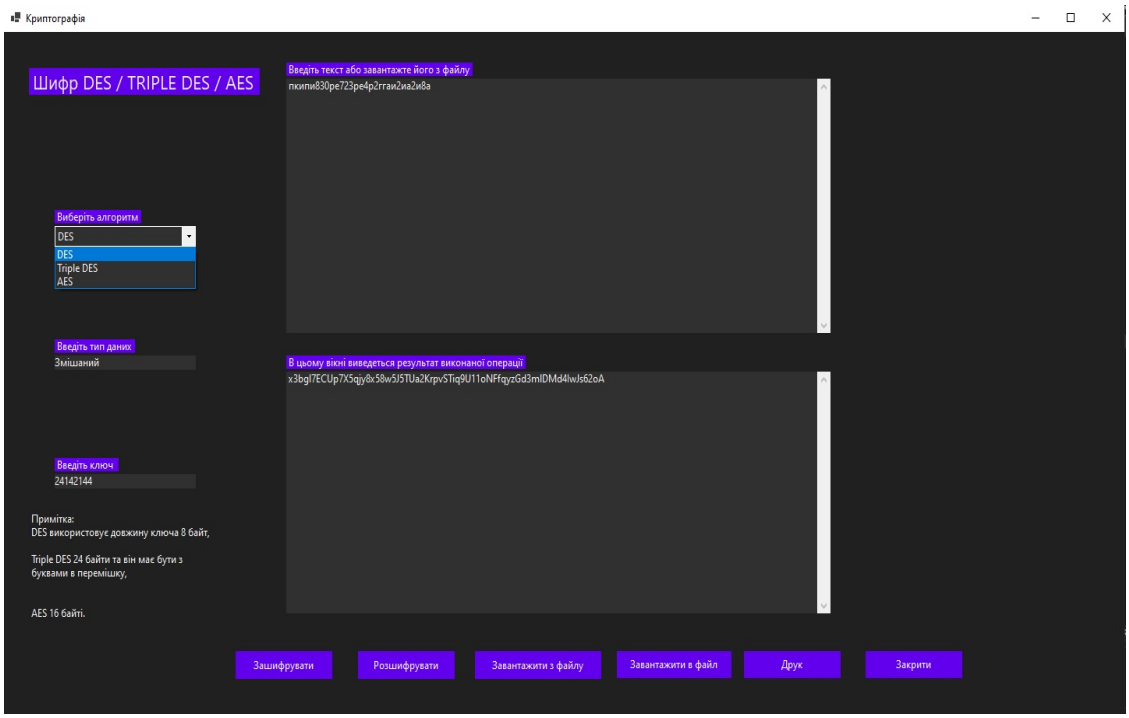


Рис. 3.24. Приклад завантаження даних з файлу

Якщо користувачу потрібно зберегти отриману інформацію з програмного додатку, він може це зробити за допомогою кнопки "Зберегти в файл". Результат виконання методу буде збережено у файлі (див. Рис. 2.25)

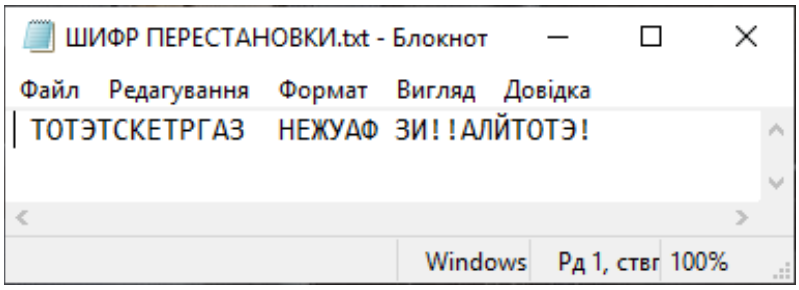


Рис. 3.25. Файл, що містить збережений результат

Вкладка "Теорія" містить теоретичний матеріал, який викладач може додавати або редагувати залежно від потреб (див. Рис. 3.26-3.27).

Рис. 3.26. Вкладка «Теорія»

Вкладка "Тестування" містить тест, створений викладачем, де студент може обрати одну із запропонованих відповідей (див. Рис. 3.28).

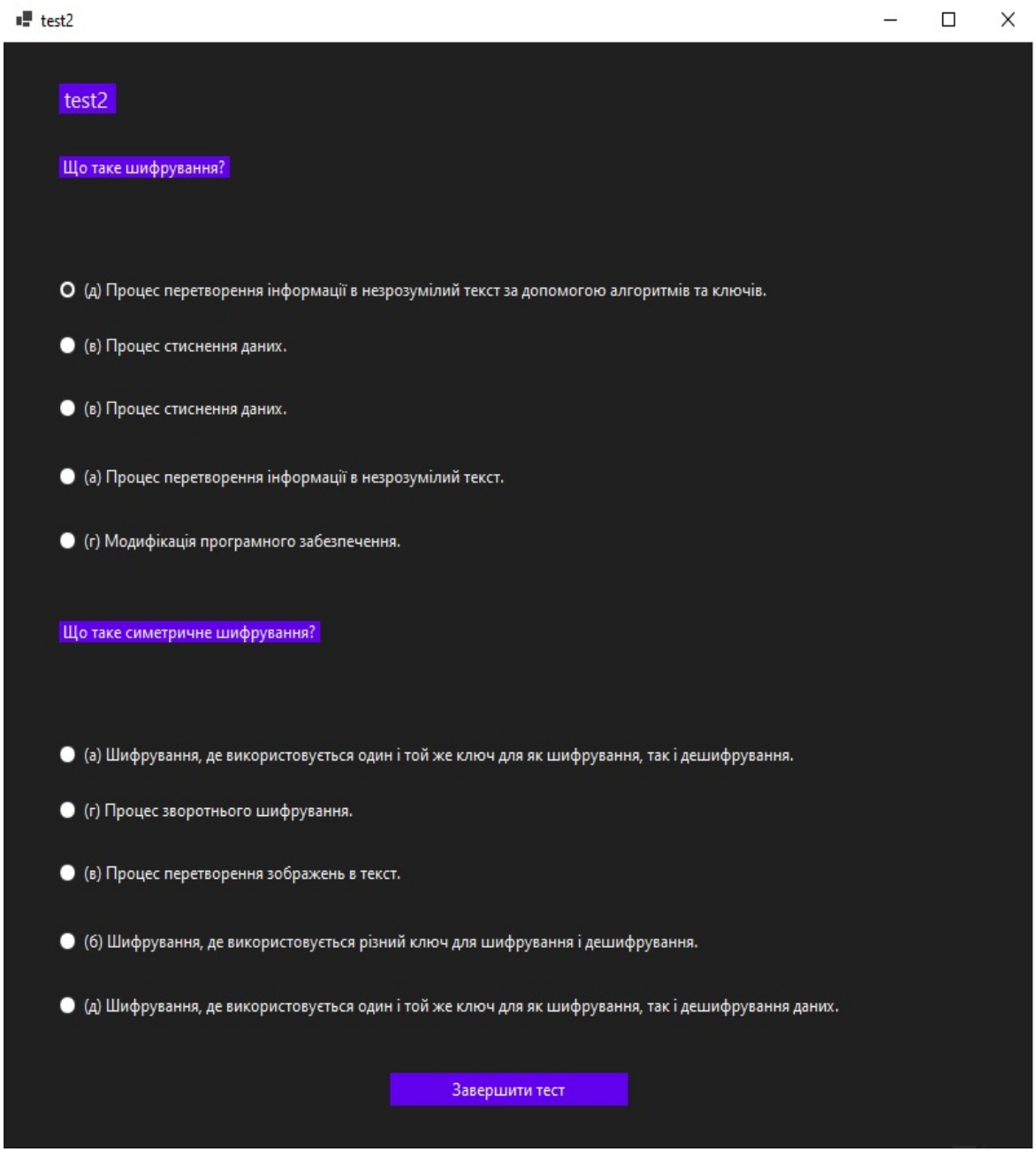


Рис. 3.28. Вкладка «Контроль знань»

Після завершення тесту користувач отримує відповідне повідомлення, де зазначена кількість правильних і неправильних відповідей, а також оцінка за пройдене тестування (див. Рис. 3.29).

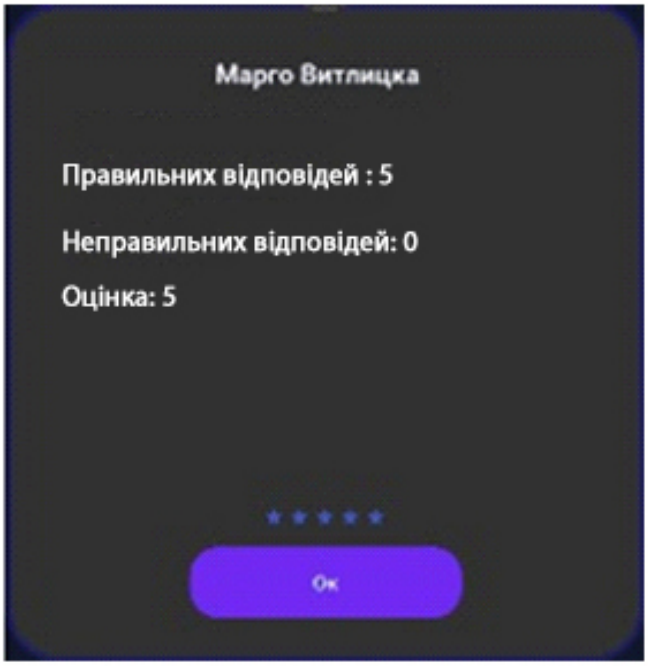


Рис. 3.29. Повідомлення про завершення тесту

Для викладача доступний новий пункт у навігаційній панелі - "Налаштування". Тут викладач може додавати питання до тесту, обирати базу даних, видаляти конкретні питання, створювати нові облікові записи для викладачів, а також редагувати теоретичний матеріал (див. Рис. 3.30).

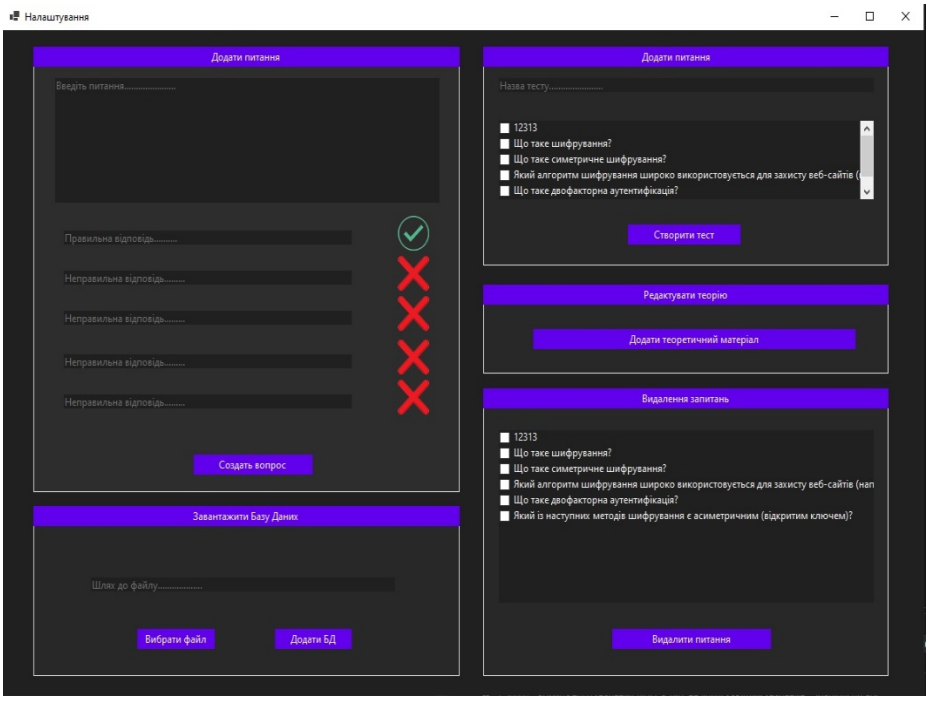


Рис. 3.30. Налаштування

Після натискання кнопки "Додавання теоретичних матеріалів" відкривається нове вікно, де викладач може ввести назву теми, матеріал з цієї теми та за бажанням додати зображення (див. Рис. 3.31).

Крім того, у викладача і студентів з'являється вкладка "Статистика", де вони можуть переглянути перелік студентів, які виконали тест, та за потреби відфільтрувати список студентів за такими критеріями, як прізвище та ім'я, група та отримана оцінка (див. Рис. 3.32).

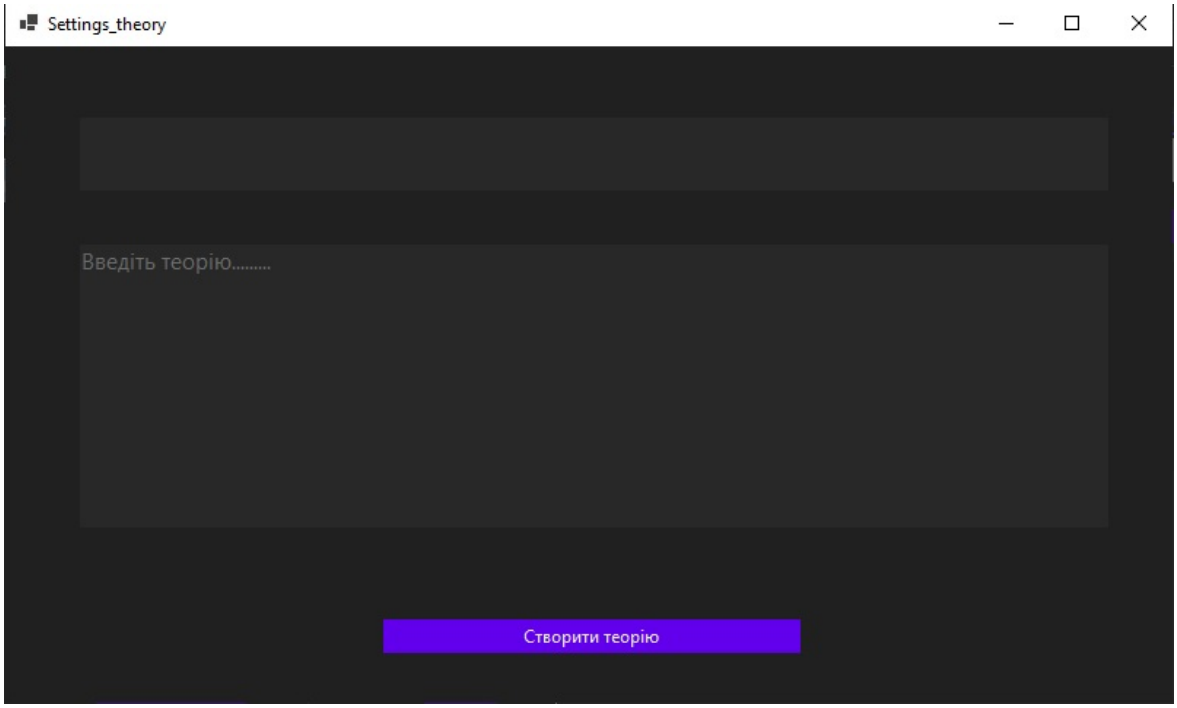
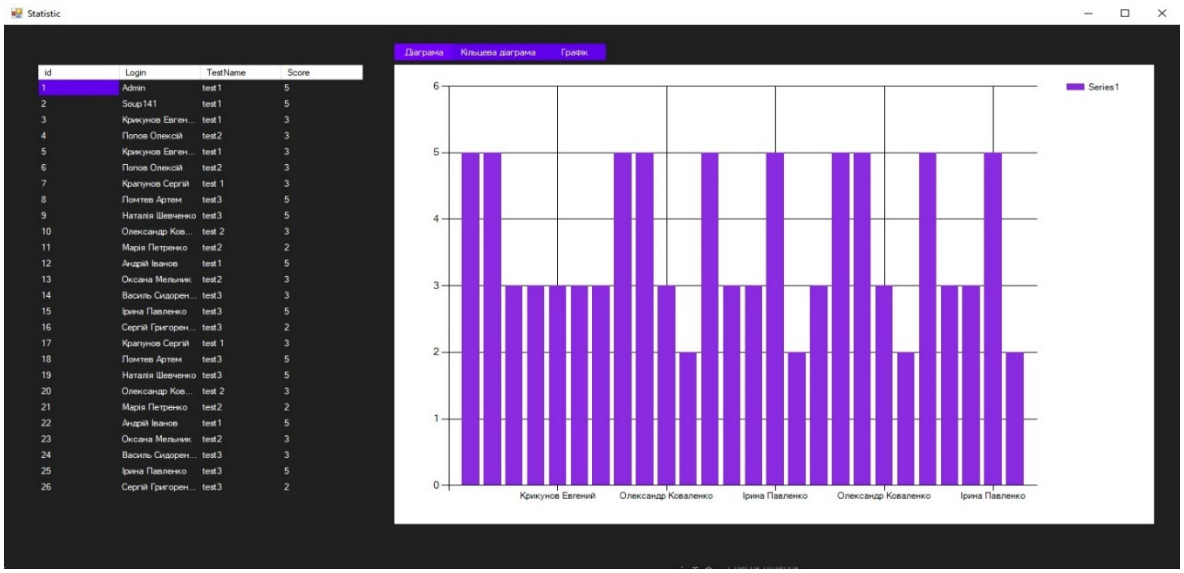


Рис. 3.31. Додання теоретичних відомостей



id	Login	TestName	Score
1	Admin	test1	5
2	Soup141	test1	5
3	Крикунов Евген...	test1	3
4	Попов Олексій	test2	3
5	Крикунов Евген...	test1	3
6	Попов Олексій	test2	3
7	Крапунов Сергій	test 1	3
8	Помтев Артем	test3	5
9	Наталія Шевченко	test3	5
10	Олександр Ков...	test 2	3
11	Марія Петренко	test2	2
12	Андрій Іванов	test1	5
13	Оксана Мельник	test2	3
14	Василь Сидорен...	test3	3
15	Ірина Павленко	test3	5
16	Сергій Григорен...	test3	2
17	Крапунов Сергій	test 1	3
18	Помтев Артем	test3	5
19	Наталія Шевченко	test3	5
20	Олександр Ков...	test 2	3
21	Марія Петренко	test2	2
22	Андрій Іванов	test1	5
23	Оксана Мельник	test2	3
24	Василь Сидорен...	test3	3
25	Ірина Павленко	test3	5
26	Сергій Григорен...	test3	2

Рис. 3.32. Вкладка «Статистика»

ВИСНОВКИ

У кваліфікаційній роботі було створено програму звану "EncryptPro", яка призначена для шифрування та розшифрування текстових повідомлень. Ця програма має зручний інтерфейс та різні рівні доступу: "Студент" і "Викладач". Залежно від рівня доступу, користувачі можуть складати тести з шифрування, використовувати методи шифрування та переглядати теоретичний матеріал. "Викладач" має більше функцій, включаючи створення тестів, додавання теоретичного матеріалу та перегляд звітів від студентів. Ця програма може бути корисною у навчальних закладах, оскільки поєднує методи шифрування з контролем знань, дозволяючи використовувати одне програмне забезпечення для навчання.

Основною особливістю "EncryptPro" є поєднання декількох функцій - шифрування та контролю знань, що полегшує роботу викладачам та навчальним закладам в яких може використовуватись даний додаток.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. “Конфіденційність в Інтернеті” [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki (остання дата звернення 30.10.2023).

2. Гнатюк С.О., Кінзерявий В.М., Охріменко А.О. Особливості криптографічного захисту державних інформаційних ресурсів Науково-практичний журнал “Безпека інформації” 2012. №1, С. 69-79.

3. “Хакерські атаки у світі” [Електронний ресурс]. – Режим доступу: https://www.slovoidilo.ua/2021/10/22/infografika/svit/krayiny-zhertvy-ta-krayiny-ahresory-xakerskyx-vijnax (остання дата звернення 1.11.2023).

4. Курсант М.Б. Назар; проф. Ю.І. Грицюк, д-р техн. наук – Львівський ДУ БЖД, Шифрування інформації методом простої заміни, Науковий вісник НЛТУ України. – 2011. − Вип. 21.5. – С. 323-325.

5. Кулібаба С. О. Курченко О. А. Криптографічний метод шифрування даних, Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 3(15), С. 216–223.

6. “Cryptool 2” [Електронний ресурс]. – Режим доступу: https://www.cryptool.org/en/ct2 (остання дата звернення 15.11.2023).

7. “Cryptopad” [Електронний ресурс]. – Режим доступу: https://docs.cryptpad.org/ru/user_guide/apps/general.html (остання дата звернення 15.11.2023).

8. “Kahoot!” [Електронний ресурс]. – Режим доступу: https://osvita.ua/vnz/high_school/73080 (остання дата звернення 15.11.2023).

9. “Cyberchef” [Електронний ресурс]. – Режим доступу: https://webcatalog.io/uk/apps/cyberchef (остання дата звернення 15.11.2023).

10. “.NET Framework” [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/.NET_Framework (остання дата звернення 20.11.2023).

11. Устілкін В.В., Люта М.В., Розломій І.О. Дослідження мов програмування java та c# для серверних платформ та робочих станцій: журнал

науковий огляд № 9, Київ, 2016. - Вип. 30. – С. 15–20.

12. Ішкова Е. А. Самовчитель С #. Початок програмування/Е.А. Ішкова. – К.: Наука та техніка, 2013. – 496 с.

13. Бішоп Дж. С # у короткому викладі / Дж. Бішоп, Н. Хорспул. - К: Лабораторія знань, 2013. – 472 с.

14. Мак-Дональд Метью. Silverlight 5 з прикладами на C# для професіоналів / Метью МакДональд. - Х: Альянс, 2020. - 848 с.

15. Загальні відомості про Visual Studio. URL: https://docs.microsoft.com /visual-studio-ide?view=vs-2019.

16. “Інференційний аналіз” [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki (остання дата звернення 25.11.2023).

17. “MAMP” [Електронний ресурс]. – Режим доступу: https://sovety.pp.ua/index.php/ua/onlajn/vebmajstru/3560-lokalnij-veb-server-mamp (остання дата звернення 30.11.2023).

18. “PHP MyAdmin” [Електронний ресурс]. – Режим доступу: wikipedia.org/wiki/PhpMyAdmin (остання дата звернення 30.11.2023).

19. Дейт К. Дж. Введення в системи баз даних/К.Дж. Дейт. – К.: Діалектика; Видання 6-е, 2019. – 784 с.

20. Комп’ютерні науки та інформаційні технології: матеріали XVIII Міжнародної конференції з проблем використання інформаційних технологій в освіті, науці та промисловості, 8 грудн. 2023 р., Дніпро, Україна / Гарбуз, В. О., Кабак, Л.В., Мороз, Б. І. " Сучасна актуальність використання шифрування інформації".

ДОДАТОК А

ЛІСТИНГ ПРОГРАМИ

Файл «MainFrame.cs»

```
using MySqlConnector;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ShablonForLabs
{
   internal class DB
   {
      MySqlConnection connection = new
MySqlConnection("server=localhost;port=3306;username=root;password=root;data
base=encryptpro");


      public void openConnection()
      {
      if(connection.State==System.Data.ConnectionState.Closed)
               connection.Open();

      }
      public void closeConnection()
      {
         if (connection.State == System.Data.ConnectionState.Open)
            connection.Close();

      }

      public MySqlConnection getConnection()
      {
         return connection;
      }

   }
}
```

Файл «Авторизація.cs »

```
using MySqlConnector;
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics.Eventing.Reader;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ShablonForLabs
{
   public partial class авторизація : Form
   {

      public string LoginUser
      {
         get { return textBoxLogin.Text; }
         set { textBoxLogin.Text = value; }
      }

      public string PassUser
      {
         get { return textBox2.Text; }
         set { textBox2.Text = value; }
      }

      public void ClearLoginAndPass()
      {
         textBoxLogin.Clear();
         textBox2.Clear();
      }
      public авторизация()
      {
         InitializeComponent();
      }

      public void button2_Click(object sender, EventArgs e)
      {
         String loginUser = textBoxLogin.Text;
         test1 Test = new test1(loginUser);

         String passUser = textBox2.Text;

         DB db = new DB();
         DataTable table = new DataTable();

         MySqlDataAdapter adapter = new MySqlDataAdapter();

         MySqlCommand command = new MySqlCommand("SELECT * FROM
`users` WHERE `login` = @login AND `password` = @password",
db.getConnection());
```

```
            command.Parameters.Add("@login", MySqlDbType.VarChar).Value =
loginUser;
            command.Parameters.Add("@password", MySqlDbType.VarChar).Value =
passUser;

            adapter.SelectCommand = command;
            adapter.Fill(table);
            CurrentUserLogin = textBoxLogin.Text;
            if (table.Rows.Count > 0)
            {
                textBox3.Text = ("Авторизація успішна");
                Form2 form2 = new Form2(this);
                form2.Show();
                this.Hide();
            }
            else
            {
                textBox3.Text = ("Не правильний логін або пароль");
            }




        }
        public string CurrentUserLogin { get; private set; }

        private void авторизация_Load(object sender, EventArgs e)
        {

        }

        private void textBox2_TextChanged(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {

            Регістрація регістрація = new Регістрація();
            регістрація.Show();
        }
    }
}
```

Регістрація.cs

```
using MySqlConnector;
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace ShablonForLabs
{
    public partial class Регістрація : Form
    {
        public Регістрація()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {

            if (textBox1.Text == "")
            {
                MessageBox.Show("Введіть логін ");
                return;
            }

            if (textBox2.Text == "")
            {
                MessageBox.Show("Введіть пароль");
                return;
            }

            if (textBox3.Text == "")
            {
                MessageBox.Show("Введіть ім'я");
                return;
            }

            if (textBox4.Text == "")
            {
                MessageBox.Show("Введіть прізвище");
                return;
            }


            if (isUserExist())
                return;

            DB db = new DB();
```

```
        MySqlCommand command = new MySqlCommand("INSERT INTO `users`
(`login`, `password`, `Name`, `Surname`, `role`) VALUES (@login,@password,
@name , @surname , @role)", db.getConnection());

        command.Parameters.Add("@login", MySqlDbType.VarChar).Value =
textBox1.Text;
        command.Parameters.Add("@password", MySqlDbType.VarChar).Value =
textBox2.Text;
        command.Parameters.Add("@name", MySqlDbType.VarChar).Value =
textBox4.Text;
        command.Parameters.Add("@surname", MySqlDbType.VarChar).Value =
textBox3.Text;
        command.Parameters.Add("@role", MySqlDbType.VarChar).Value =
comboBox1.Text;

        db.openConnection();

        if (command.ExecuteNonQuery() == 1)
           MessageBox.Show("Аккаунт було створено");
        else
           MessageBox.Show("Аккаунт  не було створено");

        db.closeConnection();

    }
    public Boolean isUserExist()

    {
        DB db = new DB();
        DataTable table = new DataTable();

        MySqlDataAdapter adapter = new MySqlDataAdapter();

        MySqlCommand command = new MySqlCommand("SELECT * FROM
`users` WHERE `login` = @uL ", db.getConnection());

        command.Parameters.Add("@uL", MySqlDbType.VarChar).Value =
textBox1.Text;

        adapter.SelectCommand = command;
        adapter.Fill(table);

        if (table.Rows.Count > 0)
        {
           MessageBox.Show("Користувач з таким логіном вже існує");
           return true;
        }

        else

           return false;
```

```
    }
```

Form 2.cs

```
using MySqlConnector;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics.Eventing.Reader;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.DataFormats;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace ShablonForLabs
{
   public partial class Form2 : Form
   {
      public string theoryTitle { get; set; }
      public string theoryContent { get; set; }
      public Dictionary<string, string> GetAllUserRoles()
      {
         string connectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";
         string query = "SELECT login, role FROM users";
         var userRoles = new Dictionary<string, string>();

         using (MySqlConnection connection = new
MySqlConnection(connectionString))
         using (MySqlCommand command = new MySqlCommand(query,
connection))
         {
            connection.Open();
            using (MySqlDataReader reader = command.ExecuteReader())
            {
               while (reader.Read())
               {
                  string login = reader["login"].ToString();
                  string role = reader["role"].ToString();
                  userRoles.Add(login, role);
               }
            }
         }

         return userRoles;
      }
```

```
    public авторизація Авторизація;
    public Form2(авторизація авторизація)
    {
        InitializeComponent();

        this.Авторизація = авторизація;
    }



    public void Form2_Load(object sender, EventArgs e)
    {

        string connectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";
        string login = Авторизація.CurrentUserLogin;
        using (MySqlConnection connection = new
MySqlConnection(connectionString))
        {
            connection.Open();

            string query = "SELECT role FROM users WHERE login=@login";

            using (MySqlCommand cmd = new MySqlCommand(query, connection))
            {
                cmd.Parameters.AddWithValue("@login", login);
                string role = cmd.ExecuteScalar() as string;

                if (role != null)
                {
                    if (role == "Admin" || role == "Teacher")
                    {
                        button9.Visible = true;
                    }
                    else
                    {
                        button9.Visible = false;
                    }
                }
            }
        }
    }

    private void button2_Click(object sender, EventArgs e)
    {
        Form1 form1 = new Form1();
        form1.Show();
    }

    private void button1_Click(object sender, EventArgs e)
    {
```

```
        Statistic statistic = new Statistic();
        statistic.Show();

    }

    private void button6_Click(object sender, EventArgs e)
    {
        tests Tests = new tests(Авторизація);
        Tests.Show();


    }

    private void button7_Click(object sender, EventArgs e)
    {

        Авторизация.ClearLoginAndPass();
        this.Close();

        Авторизация.Show();
    }

    private void button3_Click(object sender, EventArgs e)
    {
        Form3 form3 = new Form3();
        form3.Show();
    }

    private void button4_Click(object sender, EventArgs e)
    {
        Form4 form4 = new Form4();
        form4.Show();
    }

    private void button5_Click(object sender, EventArgs e)
    {
        Form5 form5 = new Form5();
        form5.Show();
    }

    private void button9_Click(object sender, EventArgs e)
    {
        Settings Set = new Settings();
        Set.Show();
    }

    private void button8_Click(object sender, EventArgs e)
    {
        theory Teorya = new theory();
        Teorya.Show();
    }
} }
```

Form 1.cs

```
using System;
using System.Drawing.Printing;
using System.Text;
using System.Windows.Forms;
using static System.Net.Mime.MediaTypeNames;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace ShablonForLabs

{
   public partial class Form1 : Form
   {


      public Form1()
      {
         InitializeComponent();
      }



      public class CaesarCipher
      {
         private const string EnglishAlphabet =
"ABCDEFGHIJKLMNOPQRSTUVWXYZ";
         private const string UkrainianAlphabet =
"АБВГҐДЕЄЖЗИІЇЙКЛМНОПРСТУФХЦЧШЩЬЮЯ";
         private const int EnglishAlphabetLength = 26;
         private const int UkrainianAlphabetLength = 33;

         public static string Encrypt(string input, int key)
         {
            StringBuilder encryptedText = new StringBuilder();

            foreach (char c in input)
            {
               if (char.IsLetter(c))
               {
                  string alphabet = GetAlphabet(c);
                  int alphabetLength = alphabet.Length;
                  int index = alphabet.IndexOf(char.ToUpper(c));
                  int encryptedIndex = (index + key) % alphabetLength;
                  char encryptedChar = char.IsUpper(c) ? alphabet[encryptedIndex] :
char.ToLower(alphabet[encryptedIndex]);
                  encryptedText.Append(encryptedChar);
               }
               else
               {
                  encryptedText.Append(c);
```

```
            }
        }

        return encryptedText.ToString();
    }
    public static string Decrypt(string input, int key)
    {
        StringBuilder decryptedText = new StringBuilder();

        foreach (char c in input)
        {
            if (char.IsLetter(c))
            {
                string alphabet = GetAlphabet(c);
                int alphabetLength = alphabet.Length;
                int index = alphabet.IndexOf(char.ToUpper(c));
                int decryptedIndex = (index - key + alphabetLength) %
alphabetLength;
                char decryptedChar = char.IsUpper(c) ? alphabet[decryptedIndex] :
char.ToLower(alphabet[decryptedIndex]);
                decryptedText.Append(decryptedChar);
            }
            else
            {
                decryptedText.Append(c);
            }
        }

        return decryptedText.ToString();
    }

    private static string GetAlphabet(char c)
    {
        // Визначаємо алфавіт на основі символу
        if (EnglishAlphabet.Contains(char.ToUpper(c)))
        {
            return EnglishAlphabet;
        }
        else if (UkrainianAlphabet.Contains(char.ToUpper(c)))
        {
            return UkrainianAlphabet;
        }
        else
        {
            // За замовчуванням використовуємо англійський алфавіт
            return EnglishAlphabet;
        }
    }
}
```

```
private void Зашифрувати_Click(object sender, EventArgs e)
{
   // Отримуємо введений текст і ключ
   string inputText = textBox1.Text;
   int key = int.Parse(textBox3.Text);

   // Шифруємо текст
   string encryptedText = CaesarCipher.Encrypt(inputText, key);

   // Виводимо зашифрований результат
   textBox2.Text = encryptedText;

   // Шифруємо дані в іншому форматі (приклад для текстового формату)
   string format = textBox4.Text;
   if (format == "Текстовий")
   {
      string encryptedData = CaesarCipher.Encrypt(inputText, key);

      // Використовуйте отримані зашифровані дані за необхідності
      // Наприклад, збереження у файл або передача через мережу
   }
   else if (format == "Інший формат")
   {
      // Шифрування даних в іншому форматі
   }
   else
   {
      // Обробка невідомого формату даних
   }
}



private void label1_Click(object sender, EventArgs e)
{

}

private void button2_Click(object sender, EventArgs e)
{
   {
      string inputText = textBox1.Text;
      int key = int.Parse(textBox3.Text);

      string decryptedText = CaesarCipher.Decrypt(inputText, key);

      textBox2.Text = decryptedText;
   }
}
```

```
        private void button6_Click(object sender, EventArgs e)
        {

            this.Close();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            SaveFileDialog saveFileDialog = new SaveFileDialog();
            saveFileDialog.Filter = "Текстові файли (*.txt)|*.txt|Всі файли (*.*)|*.*";
            saveFileDialog.Title = "Зберегти файл";

            if (saveFileDialog.ShowDialog() == DialogResult.OK)
            {
                string filePath = saveFileDialog.FileName;
                string content = textBox2.Text;

                try
                {
                    File.WriteAllText(filePath, content);
                    MessageBox.Show("Файл успішно збережено.");
                }
                catch (Exception ex)
                {
                    MessageBox.Show($"Сталася помилка при збереженні файлу:
{ex.Message}");
                }
            }
        }

        private void button3_Click(object sender, EventArgs e)
        {
            OpenFileDialog openFileDialog = new OpenFileDialog();
            openFileDialog.Filter = "Текстові файли (*.txt)|*.txt|Всі файли (*.*)|*.*";
            openFileDialog.Title = "Відкрити файл";

            if (openFileDialog.ShowDialog() == DialogResult.OK)
            {
                string filePath = openFileDialog.FileName;

                try
                {
                    string content = File.ReadAllText(filePath);
                    textBox1.Text = content;
                    MessageBox.Show("Файл успішно завантажено.");
                }
                catch (Exception ex)
                {
                    MessageBox.Show($"Сталася помилка при завантаженні файлу:
{ex.Message}");
```

```
            }
        }
    }

    private void button5_Click(object sender, EventArgs e)
    {
        {
            PrintDocument printDocument = new PrintDocument();
            printDocument.PrintPage += new
PrintPageEventHandler(PrintPageHandler);

            try
            {
                PrintDialog printDialog = new PrintDialog();
                printDialog.Document = printDocument;

                if (printDialog.ShowDialog() == DialogResult.OK)
                {
                    printDocument.Print();
                    MessageBox.Show("Вміст успішно надіслано на друк.");
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Сталася помилка при надсиланні на друк:
{ex.Message}");
            }
        }
    }

    private void PrintPageHandler(object sender, PrintPageEventArgs e)
    {
        string content = textBox2.Text;
        Font font = new Font("Arial", 12);

        e.Graphics.DrawString(content, font, Brushes.Black, new PointF(10, 10));
    }
```

Form 3.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Drawing.Printing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
using System.Windows.Forms;
using static System.Net.Mime.MediaTypeNames;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace ShablonForLabs
{
   public partial class Form3 : Form
   {
      public Form3()
      {
         InitializeComponent();
      }
      private int[] lastKey = new int[3];
      public class TrithemiusCipher
      {


         public static string DecryptWithMatchingLength(string input, int[] key)
         {
            StringBuilder decryptedText = new StringBuilder();
            int keyIndex = 0;

            foreach (char c in input)
            {
               if (char.IsLetter(c))
               {
                  string alphabet = GetAlphabet(c);
                  int alphabetLength = alphabet.Length;
                  int index = alphabet.IndexOf(char.ToUpper(c));
                  int decryptedIndex = (index - key[keyIndex] + alphabetLength) %
alphabetLength;
                  char decryptedChar = char.IsUpper(c) ? alphabet[decryptedIndex] :
char.ToLower(alphabet[decryptedIndex]);
                  decryptedText.Append(decryptedChar);

                  // Оновлення ключа
                  keyIndex++;
                  if (keyIndex >= key.Length)
                  {
                     keyIndex = 0;
                  }
               }
               else
               {
                  decryptedText.Append(c);
               }
            }

            return decryptedText.ToString();
         }
```

```
        private const string EnglishAlphabet =
"ABCDEFGHIJKLMNOPQRSTUVWXYZ";
        private const string UkrainianAlphabet =
"АБВГҐДЕЄЖЗИІЇЙКЛМНОПРСТУФХЦЧШЩЬЮЯ";
        private const int EnglishAlphabetLength = 26;
        private const int UkrainianAlphabetLength = 33;

        public static string Encrypt(string input, int[] key)
        {
            StringBuilder encryptedText = new StringBuilder();

            foreach (char c in input)
            {
                if (char.IsLetter(c))
                {
                    string alphabet = GetAlphabet(c);
                    int alphabetLength = alphabet.Length;
                    int index = alphabet.IndexOf(char.ToUpper(c));
                    int encryptedIndex = (index + key[0]) % alphabetLength;
                    char encryptedChar = char.IsUpper(c) ? alphabet[encryptedIndex] :
char.ToLower(alphabet[encryptedIndex]);
                    encryptedText.Append(encryptedChar);
                    key[0] = (key[0] * key[1] + key[2]) % alphabetLength; // оновлення
ключа
                }
                else
                {
                    encryptedText.Append(c);
                }
            }

            return encryptedText.ToString();
        }

        public static string Decrypt(string input, int[] key)
        {
            StringBuilder decryptedText = new StringBuilder();

            foreach (char c in input)
            {
                if (char.IsLetter(c))
                {
                    string alphabet = GetAlphabet(c);
                    int alphabetLength = alphabet.Length;
                    int index = alphabet.IndexOf(char.ToUpper(c));
                    int decryptedIndex = (index - key[0] + alphabetLength) %
alphabetLength;
                    char decryptedChar = char.IsUpper(c) ? alphabet[decryptedIndex] :
char.ToLower(alphabet[decryptedIndex]);
                    decryptedText.Append(decryptedChar);
                    key[0] = (key[0] * key[1] + key[2]) % alphabetLength; // оновлення
ключа
```

```
            }
            else
            {
                decryptedText.Append(c);
            }
        }

        return decryptedText.ToString();
    }

    private static string GetAlphabet(char c)
    {
        if (EnglishAlphabet.Contains(char.ToUpper(c)))
        {
            return EnglishAlphabet;
        }
        else if (UkrainianAlphabet.Contains(char.ToUpper(c)))
        {
            return UkrainianAlphabet;
        }
        else
        {
            return EnglishAlphabet;
        }
    }
}

private void button1_Click(object sender, EventArgs e)
{
    string inputText = textBox1.Text;
    int[] key = new int[3];
    key[0] = int.Parse(textBox3.Text);
    key[1] = int.Parse(textBox5.Text);
    key[2] = int.Parse(textBox6.Text);
    lastKey[0] = key[0];
    lastKey[1] = key[1];
    lastKey[2] = key[2];
    string encryptedText = TrithemiusCipher.Encrypt(inputText, key);

    textBox2.Text = encryptedText;

    string format = textBox4.Text;
    if (format == "Текстовий")
    {
        string encryptedData = TrithemiusCipher.Encrypt(inputText, key);
        // Шифрування даних в текстовому форматі
    }
    else if (format == "Інший формат")
    {
        // Шифрування даних в іншому форматі
    }
    else
```

```
            {
                // Обробка невідомого формату даних
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            string inputText = textBox1.Text;
            int[] key = new int[3];
            key[0] = int.Parse(textBox3.Text);
            key[1] = int.Parse(textBox5.Text);
            key[2] = int.Parse(textBox6.Text);

            // Отримання шифрованого тексту з поля textBox2
            string encryptedText = textBox2.Text;

            // Розшифрування даних з такою ж довжиною, як і шифрування
            string decryptedText =
TrithemiusCipher.DecryptWithMatchingLength(encryptedText, key);

            textBox2.Text = decryptedText;
        }


        private void Button6_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void Button4_Click(object sender, EventArgs e)
        {
            SaveFileDialog saveFileDialog = new SaveFileDialog();
            saveFileDialog.Filter = "Текстові файли (*.txt)|*.txt|Всі файли (*.*)|*.*";
            saveFileDialog.Title = "Зберегти файл";

            if (saveFileDialog.ShowDialog() == DialogResult.OK)
            {
                string filePath = saveFileDialog.FileName;
                string content = textBox2.Text;

                try
                {
                    System.IO.File.WriteAllText(filePath, content);
                    MessageBox.Show("Файл успішно збережено.");
                }
                catch (Exception ex)
                {
                    MessageBox.Show($"Сталася помилка при збереженні файлу:
{ex.Message}");
                }
            }
        }
```

```
        private void Button3_Click(object sender, EventArgs e)
        {
            OpenFileDialog openFileDialog = new OpenFileDialog();
            openFileDialog.Filter = "Текстові файли (*.txt)|*.txt|Всі файли (*.*)|*.*";
            openFileDialog.Title = "Відкрити файл";

            if (openFileDialog.ShowDialog() == DialogResult.OK)
            {
                string filePath = openFileDialog.FileName;

                try
                {
                    string content = System.IO.File.ReadAllText(filePath);
                    textBox1.Text = content;
                    MessageBox.Show("Файл успішно завантажено.");
                }
                catch (Exception ex)
                {
                    MessageBox.Show($"Сталася помилка при завантаженні файлу:
{ex.Message}");
                }
            }
        }

        private void Button5_Click(object sender, EventArgs e)
        {
            PrintDocument printDocument = new PrintDocument();
            printDocument.PrintPage += new
PrintPageEventHandler(PrintPageHandler);

            try
            {
                PrintDialog printDialog = new PrintDialog();
                printDialog.Document = printDocument;

                if (printDialog.ShowDialog() == DialogResult.OK)
                {
                    printDocument.Print();
                    MessageBox.Show("Вміст успішно надіслано на друк.");
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Сталася помилка при надсиланні на друк:
{ex.Message}");
            }
        }


        private void PrintPageHandler(object sender, PrintPageEventArgs e)
        {
```

```
            string content = textBox2.Text;
            Font font = new Font("Arial", 12);

            e.Graphics.DrawString(content, font, Brushes.Black, new PointF(10, 10));
        }

        private void button7_Click(object sender, EventArgs e)
        {
            textBox7.Text = string.Join(", ", lastKey);
        }
```

Form 4.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Drawing.Printing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace ShablonForLabs
{
    public partial class Form4 : Form
    {
        private string encryptionKey;
        public Form4()
        {
            InitializeComponent();
        }
        public class GammaGenerator
        {
            public static string GenerateGamma(int length)
            {
                StringBuilder gamma = new StringBuilder();
                Random random = new Random();

                for (int i = 0; i < length; i++)
                {
                    char randomChar = (char)random.Next(32, 127); // Вибираємо
випадковий символ з діапазону ASCII
                    gamma.Append(randomChar);
                }

                return gamma.ToString();
            }
```

```
    }
    public class TextEncryptor
    {
        public static string EncryptText(string plainText, string gamma)
        {
            StringBuilder encryptedText = new StringBuilder();

            for (int i = 0; i < plainText.Length; i++)
            {
                char gammaChar = gamma[i];
                char plainChar = plainText[i];

                // Використовуємо побітове "XOR" для шифрування
                char encryptedChar = (char)(plainChar ^ gammaChar);

                encryptedText.Append(encryptedChar);
            }

            return encryptedText.ToString();
        }

        public static string DecryptText(string encryptedText, string gamma)
        {
            StringBuilder decryptedText = new StringBuilder();

            for (int i = 0; i < encryptedText.Length; i++)
            {
                char gammaChar = gamma[i];
                char encryptedChar = encryptedText[i];

                // Використовуємо побітове "XOR" для розшифрування
                char decryptedChar = (char)(encryptedChar ^ gammaChar);

                decryptedText.Append(decryptedChar);
            }

            return decryptedText.ToString();
        }
    }
    private string GenerateGamma(int length)
    {
        StringBuilder gamma = new StringBuilder();
        Random random = new Random();

        for (int i = 0; i < length; i++)
        {
            char randomChar = (char)random.Next(32, 127); // Вибираємо
випадковий символ з діапазону ASCII
            gamma.Append(randomChar);
        }

        return gamma.ToString();
```

```
    }
    private void button1_Click(object sender, EventArgs e)
    {
        string plainText = textBox1.Text;

        // Генерація гами
        string gamma = GenerateGamma(plainText.Length);

        // Шифрування тексту
        string encryptedText = EncryptText(plainText, gamma);

        textBox2.Text = encryptedText;


        // Збереження ключа шифрування
        encryptionKey = gamma;
    }



    private string EncryptText(string plainText, string gamma)
    {
        StringBuilder encryptedText = new StringBuilder();

        for (int i = 0; i < plainText.Length; i++)
        {
            char gammaChar = gamma[i];
            char plainChar = plainText[i];

            // Використовуємо побітове "XOR" для шифрування
            char encryptedChar = (char)(plainChar ^ gammaChar);

            encryptedText.Append(encryptedChar);
        }

        return encryptedText.ToString();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        string encryptedText = textBox1.Text;

        if (!string.IsNullOrEmpty(encryptionKey))
        {
            // Розшифрування тексту
            string decryptedText = DecryptText(encryptedText, encryptionKey);
            textBox2.Text = decryptedText;
        }
        else
        {
            MessageBox.Show("Немає ключа шифрування. Спочатку виконайте
шифрування тексту.");
```

```
        }
    }

    private string DecryptText(string encryptedText, string gamma)
    {
        StringBuilder decryptedText = new StringBuilder();

        for (int i = 0; i < encryptedText.Length; i++)
        {
            char gammaChar = gamma[i];
            char encryptedChar = encryptedText[i];

            // Використовуємо побітове "XOR" для розшифрування
            char decryptedChar = (char)(encryptedChar ^ gammaChar);

            decryptedText.Append(decryptedChar);
        }

        return decryptedText.ToString();
    }

    private void button3_Click(object sender, EventArgs e)
    {

        OpenFileDialog openFileDialog = new OpenFileDialog();
        openFileDialog.Filter = "Текстові файли (*.txt)|*.txt|Всі файли (*.*)|*.*";
        openFileDialog.Title = "Відкрити файл";

        if (openFileDialog.ShowDialog() == DialogResult.OK)
        {
            string filePath = openFileDialog.FileName;

            try
            {
                string content = System.IO.File.ReadAllText(filePath);
                textBox1.Text = content;
                MessageBox.Show("Файл успішно завантажено.");
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Сталася помилка при завантаженні файлу:
{ex.Message}");
            }
        }

    }

    private void button5_Click(object sender, EventArgs e)
    {

        PrintDocument printDocument = new PrintDocument();
```

```
        printDocument.PrintPage += new
PrintPageEventHandler(PrintPageHandler);

        try
        {
            PrintDialog printDialog = new PrintDialog();
            printDialog.Document = printDocument;

            if (printDialog.ShowDialog() == DialogResult.OK)
            {
                printDocument.Print();
                MessageBox.Show("Вміст успішно надіслано на друк.");
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show($"Сталася помилка при надсиланні на друк:
{ex.Message}");

        }
        void PrintPageHandler(object sender, PrintPageEventArgs e)
        {
            string content = textBox2.Text;
            Font font = new Font("Arial", 12);

            e.Graphics.DrawString(content, font, Brushes.Black, new PointF(10, 10));
        }
    }

    private void button4_Click(object sender, EventArgs e)
    {

        SaveFileDialog saveFileDialog = new SaveFileDialog();
        saveFileDialog.Filter = "Текстові файли (*.txt)|*.txt|Всі файли (*.*)|*.*";
        saveFileDialog.Title = "Зберегти файл";

        if (saveFileDialog.ShowDialog() == DialogResult.OK)
        {
            string filePath = saveFileDialog.FileName;
            string content = textBox2.Text;

            try
            {
                System.IO.File.WriteAllText(filePath, content);
                MessageBox.Show("Файл успішно збережено.");
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Сталася помилка при збереженні файлу:
{ex.Message}");
            }
        }
```

```
    }

    private void button6_Click(object sender, EventArgs e)
    {
        {
            this.Close();
        }
    }
```

Form 5.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Drawing.Printing;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace ShablonForLabs
{
    public partial class Form5 : Form
    {
        public Form5()
        {
            InitializeComponent();
        }




        private void button1_Click(object sender, EventArgs e)
        {
            string plaintext = textBox1.Text;
            string encryptionAlgorithm = comboBox1.SelectedItem.ToString();
            string encryptionKey = textBox3.Text;
            string encryptedText = "";

            if (encryptionAlgorithm == "DES")
            {
                encryptedText = EncryptTextUsingDES(plaintext, encryptionKey);
            }
            else if (encryptionAlgorithm == "Triple DES")
```

```
            {
                encryptedText = EncryptTextUsingTripleDES(plaintext, encryptionKey);
            }
            else if (encryptionAlgorithm == "AES")
            {
                encryptedText = EncryptTextUsingAES(plaintext, encryptionKey);
            }
            else
            {
                throw new ArgumentException("Invalid encryption algorithm selected.");
            }

            textBox2.Text = encryptedText;
        }

        private void button2_Click(object sender, EventArgs e)
        {
            string encryptedText = textBox1.Text;
            string encryptionAlgorithm = comboBox1.SelectedItem.ToString();
            string encryptionKey = textBox3.Text;
            string decryptedText = "";

            if (encryptionAlgorithm == "DES")
            {
                decryptedText = DecryptTextUsingDES(encryptedText, encryptionKey);
            }
            else if (encryptionAlgorithm == "Triple DES")
            {
                decryptedText = DecryptTextUsingTripleDES(encryptedText,
encryptionKey);
            }
            else if (encryptionAlgorithm == "AES")
            {
                decryptedText = DecryptTextUsingAES(encryptedText, encryptionKey);
            }
            else
            {
                throw new ArgumentException("Invalid encryption algorithm selected.");
            }

            textBox2.Text = decryptedText;
        }

        private string EncryptTextUsingDES(string plaintext, string encryptionKey)
        {
            DESCryptoServiceProvider cryptic = new DESCryptoServiceProvider();
            cryptic.Key = ASCIIEncoding.ASCII.GetBytes(encryptionKey);
            cryptic.IV = ASCIIEncoding.ASCII.GetBytes(encryptionKey.Substring(0,
8)); // Використовуємо перші 8 байт ключа для IV
            cryptic.Mode = CipherMode.CBC;

            using (MemoryStream ms = new MemoryStream())
```

```
        {
            using (CryptoStream crStream = new CryptoStream(ms,
cryptic.CreateEncryptor(), CryptoStreamMode.Write))
            {
                byte[] data = Encoding.UTF8.GetBytes(plaintext);
                crStream.Write(data, 0, data.Length);
            }

            byte[] encryptedData = ms.ToArray();
            string encryptedText = Convert.ToBase64String(encryptedData);
            return encryptedText;
        }
    }

    private string DecryptTextUsingDES(string encryptedText, string
encryptionKey)
    {
        DESCryptoServiceProvider cryptic = new DESCryptoServiceProvider();
        cryptic.Key = ASCIIEncoding.ASCII.GetBytes(encryptionKey);
        cryptic.IV = ASCIIEncoding.ASCII.GetBytes(encryptionKey.Substring(0,
8)); // Використовуємо перші 8 байт ключа для IV
        cryptic.Mode = CipherMode.CBC;

        byte[] encryptedData = Convert.FromBase64String(encryptedText);

        using (MemoryStream ms = new MemoryStream(encryptedData))
        {
            using (CryptoStream crStream = new CryptoStream(ms,
cryptic.CreateDecryptor(), CryptoStreamMode.Read))
            {
                using (StreamReader reader = new StreamReader(crStream))
                {
                    string decryptedText = reader.ReadToEnd();
                    return decryptedText;
                }
            }
        }
    }

    private string EncryptTextUsingTripleDES(string plaintext, string
encryptionKey)
    {
        TripleDESCryptoServiceProvider cryptic = new
TripleDESCryptoServiceProvider();
        cryptic.Key = ASCIIEncoding.ASCII.GetBytes(encryptionKey);
        cryptic.IV = ASCIIEncoding.ASCII.GetBytes(encryptionKey.Substring(0,
8)); // Використовуємо перші 8 байт ключа для IV
        cryptic.Mode = CipherMode.CBC;

        using (MemoryStream ms = new MemoryStream())
        {
            using (CryptoStream crStream = new CryptoStream(ms,
```

```
cryptic.CreateEncryptor(), CryptoStreamMode.Write))
            {
                byte[] data = Encoding.UTF8.GetBytes(plaintext);
                crStream.Write(data, 0, data.Length);
            }

            byte[] encryptedData = ms.ToArray();
            string encryptedText = Convert.ToBase64String(encryptedData);
            return encryptedText;
        }
    }

    private string DecryptTextUsingTripleDES(string encryptedText, string
encryptionKey)
    {
        TripleDESCryptoServiceProvider cryptic = new
TripleDESCryptoServiceProvider();
        cryptic.Key = ASCIIEncoding.ASCII.GetBytes(encryptionKey);
        cryptic.IV = ASCIIEncoding.ASCII.GetBytes(encryptionKey.Substring(0,
8)); // Використовуємо перші 8 байт ключа для IV
        cryptic.Mode = CipherMode.CBC;

        byte[] encryptedData = Convert.FromBase64String(encryptedText);

        using (MemoryStream ms = new MemoryStream(encryptedData))
        {
            using (CryptoStream crStream = new CryptoStream(ms,
cryptic.CreateDecryptor(), CryptoStreamMode.Read))
            {
                using (StreamReader reader = new StreamReader(crStream))
                {
                    string decryptedText = reader.ReadToEnd();
                    return decryptedText;
                }
            }
        }
    }

    private string EncryptTextUsingAES(string plaintext, string encryptionKey)
    {
        AesCryptoServiceProvider cryptic = new AesCryptoServiceProvider();
        cryptic.Key = ASCIIEncoding.ASCII.GetBytes(encryptionKey);
        cryptic.IV = ASCIIEncoding.ASCII.GetBytes(encryptionKey.Substring(0,
16)); // Використовуємо перші 16 байт ключа для IV
        cryptic.Mode = CipherMode.CBC;

        using (MemoryStream ms = new MemoryStream())
        {
            using (CryptoStream crStream = new CryptoStream(ms,
cryptic.CreateEncryptor(), CryptoStreamMode.Write))
            {
                byte[] data = Encoding.UTF8.GetBytes(plaintext);
```

```
            crStream.Write(data, 0, data.Length);
        }

        byte[] encryptedData = ms.ToArray();
        string encryptedText = Convert.ToBase64String(encryptedData);
        return encryptedText;
      }
    }

    private string DecryptTextUsingAES(string encryptedText, string
encryptionKey)
    {
      AesCryptoServiceProvider cryptic = new AesCryptoServiceProvider();
      cryptic.Key = ASCIIEncoding.ASCII.GetBytes(encryptionKey);
      cryptic.IV = ASCIIEncoding.ASCII.GetBytes(encryptionKey.Substring(0,
16)); // Використовуємо перші 16 байт ключа для IV
      cryptic.Mode = CipherMode.CBC;

      byte[] encryptedData = Convert.FromBase64String(encryptedText);

      using (MemoryStream ms = new MemoryStream(encryptedData))
      {
        using (CryptoStream crStream = new CryptoStream(ms,
cryptic.CreateDecryptor(), CryptoStreamMode.Read))
        {
          using (StreamReader reader = new StreamReader(crStream))
          {
            string decryptedText = reader.ReadToEnd();
            return decryptedText;
          }
        }
      }
    }
    private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
    {

    }
    private void Form_Load(object sender, EventArgs e)
    {
      label6.AutoSize = false;
      label6.Text = "Примітка: DES використовує довжину ключа 8 байт,\r\n
Triple DES 24 байти, \r\nAES 16 байтів";
      label6.Height = 100;
    }

    private void button3_Click(object sender, EventArgs e)
    {

      OpenFileDialog openFileDialog = new OpenFileDialog();
      openFileDialog.Filter = "Текстові файли (*.txt)|*.txt|Всі файли (*.*)|*.*";
      openFileDialog.Title = "Відкрити файл";
```

```
        if (openFileDialog.ShowDialog() == DialogResult.OK)
        {
            string filePath = openFileDialog.FileName;

            try
            {
                string content = System.IO.File.ReadAllText(filePath);
                textBox1.Text = content;
                MessageBox.Show("Файл успішно завантажено.");
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Сталася помилка при завантаженні файлу:
{ex.Message}");
            }
        }

    }

    private void button4_Click(object sender, EventArgs e)
    {

        SaveFileDialog saveFileDialog = new SaveFileDialog();
        saveFileDialog.Filter = "Текстові файли (*.txt)|*.txt|Всі файли (*.*)|*.*";
        saveFileDialog.Title = "Зберегти файл";

        if (saveFileDialog.ShowDialog() == DialogResult.OK)
        {
            string filePath = saveFileDialog.FileName;
            string content = textBox2.Text;

            try
            {
                System.IO.File.WriteAllText(filePath, content);
                MessageBox.Show("Файл успішно збережено.");
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Сталася помилка при збереженні файлу:
{ex.Message}");
            }
        }
    }

    private void button5_Click(object sender, EventArgs e)
    {

        PrintDocument printDocument = new PrintDocument();
        printDocument.PrintPage += new
PrintPageEventHandler(PrintPageHandler);

        try
```

```
        {
            PrintDialog printDialog = new PrintDialog();
            printDialog.Document = printDocument;

            if (printDialog.ShowDialog() == DialogResult.OK)
            {
                printDocument.Print();
                MessageBox.Show("Вміст успішно надіслано на друк.");
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show($"Сталася помилка при надсиланні на друк:
{ex.Message}");

        }
        void PrintPageHandler(object sender, PrintPageEventArgs e)
        {
            string content = textBox2.Text;
            Font font = new Font("Arial", 12);

            e.Graphics.DrawString(content, font, Brushes.Black, new PointF(10, 10));
        }
    }

    private void button6_Click(object sender, EventArgs e)
    {


        this.Close();


    }
```

Settings.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.Common;
using MySqlConnector;

namespace ShablonForLabs
{
    public partial class Settings : Form
```

```
    {

        private string connectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";
        private MySqlConnection dbConnection;
        private int tableCounter = 1;
        public Settings()
        {
            InitializeComponent();
            dbConnection = new MySqlConnection(connectionString);
            dbConnection.Open();
            LoadQuestions();
            this.button3.Click += new EventHandler(DeleteSelectedQuestions);
            this.ButtonCreateTest.Click += new EventHandler(CreateTestTable);

        }

        private void CreateTestTable(object sender, EventArgs e)
        {
            string connectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";
            MySqlConnection connection = new MySqlConnection(connectionString);

            try
            {
                connection.Open();


                string tableName = "Test" + tableCounter;
                string createTableQuery = $"CREATE TABLE IF NOT EXISTS
{tableName} (ID INT AUTO_INCREMENT PRIMARY KEY, TestName
VARCHAR(255), Question VARCHAR(255))";
                MySqlCommand createTableCommand = new
MySqlCommand(createTableQuery, connection);
                createTableCommand.ExecuteNonQuery();

                string testName = textBox1.Text;

                        for (int i = 0; i < checkedListBox2.Items.Count; i++)
                {
                    if (checkedListBox2.GetItemChecked(i))
                    {
                        string question = checkedListBox2.Items[i].ToString();
                        string insertDataQuery = $"INSERT INTO {tableName} (TestName,
Question) VALUES (@TestName, @Question)";
                        MySqlCommand insertDataCommand = new
MySqlCommand(insertDataQuery, connection);
                        insertDataCommand.Parameters.AddWithValue("@TestName",
testName);
                        insertDataCommand.Parameters.AddWithValue("@Question",
question);
                        insertDataCommand.ExecuteNonQuery();
```

```
            }
        }

        MessageBox.Show($"Таблица {tableName} створена і данні додані в таблицю та БД.");
        tableCounter++;
      }
      catch (Exception ex)
      {
        MessageBox.Show("Ошибка: " + ex.Message);
      }
      finally
      {
        connection.Close();
      }
    }


    private void Settings_Load(object sender, EventArgs e)
    {

    }

    public class QuestionItem
    {
      public int ID { get; set; }
      public string QuestionText { get; set; }

      public QuestionItem(int questionID, string questionText)
      {
        ID = questionID;
        QuestionText = questionText;
      }

      public override string ToString()
      {
        return QuestionText;
      }
    }

    private void label1_Click(object sender, EventArgs e)
    {

    }

    private void flowLayoutPanel1_Paint(object sender, PaintEventArgs e)
    {

    }

    private void textBox6_TextChanged(object sender, EventArgs e)
    {
```

```
        }

        private void button1_Click(object sender, EventArgs e)
        {

            string question = txtQuestion.Text;
            string correctAnswer = txtCorrectAnswer.Text;
            string[] wrongAnswers = new string[4];

            wrongAnswers[0] = txtWrongAnswer1.Text;
            wrongAnswers[1] = txtWrongAnswer2.Text;
            wrongAnswers[2] = txtWrongAnswer3.Text;
            wrongAnswers[3] = txtWrongAnswer4.Text;


            Random random = new Random();
            wrongAnswers = wrongAnswers.OrderBy(x => random.Next()).ToArray();

            using (MySqlCommand cmd = new MySqlCommand("INSERT INTO
Questions (Question, CorrectAnswer, WrongAnswer1, WrongAnswer2,
WrongAnswer3, WrongAnswer4) VALUES (@Question, @CorrectAnswer,
@WrongAnswer1, @WrongAnswer2, @WrongAnswer3, @WrongAnswer4)",
dbConnection))
            {
                cmd.Parameters.AddWithValue("@Question", question);
                cmd.Parameters.AddWithValue("@CorrectAnswer", correctAnswer);
                cmd.Parameters.AddWithValue("@WrongAnswer1", wrongAnswers[0]);
                cmd.Parameters.AddWithValue("@WrongAnswer2", wrongAnswers[1]);
                cmd.Parameters.AddWithValue("@WrongAnswer3", wrongAnswers[2]);
                cmd.Parameters.AddWithValue("@WrongAnswer4", wrongAnswers[3]);
                cmd.ExecuteNonQuery();
            }

            MessageBox.Show("Питання збережено до БД.", "Інформація",
MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
        private void LoadQuestions()
        {
            string connectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";
            MySqlConnection connection = new MySqlConnection(connectionString);

            try
            {
                connection.Open();
                string query = "SELECT Question FROM Questions";
                MySqlCommand command = new MySqlCommand(query, connection);
                MySqlDataReader reader = command.ExecuteReader();

                while (reader.Read())
                {
```

```
                checkedListBox2.Items.Add(reader["Question"].ToString());
                checkedListBox3.Items.Add(reader["Question"].ToString());
            }

            reader.Close();
        }
        catch (Exception ex)
        {
            MessageBox.Show("Ошибка: " + ex.Message);
        }
        finally
        {
            connection.Close();
        }
    }
    private void ButtonCreateTest_Click(object sender, EventArgs e)
    {




    }
    private void SaveSelectedQuestionsToTestTable(string testName,
List<QuestionItem> selectedQuestions)
    {

        using (MySqlConnection connection = new
MySqlConnection(connectionString))
        {
            connection.Open();

            string clearTestQuery = "DELETE FROM test WHERE TestName =
@TestName";
            using (MySqlCommand clearTestCommand = new
MySqlCommand(clearTestQuery, connection))
            {
                clearTestCommand.Parameters.AddWithValue("@TestName",
testName);
                clearTestCommand.ExecuteNonQuery();
            }

            foreach (var question in selectedQuestions)
            {
                string insertTestQuery = "INSERT INTO test (TestName, QuestionID)
VALUES (@TestName, @QuestionID)";
                using (MySqlCommand insertTestCommand = new
MySqlCommand(insertTestQuery, connection))
                {
                    insertTestCommand.Parameters.AddWithValue("@TestName",
testName);
                    insertTestCommand.Parameters.AddWithValue("@QuestionID",
question.ID);
```

```
                insertTestCommand.ExecuteNonQuery();
            }
        }

        connection.Close();
    }
}
private void DeleteSelectedQuestions(object sender, EventArgs e)
{
    string connectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";
    MySqlConnection connection = new MySqlConnection(connectionString);

    try
    {
        connection.Open();

        for (int i = 0; i < checkedListBox3.Items.Count; i++)

        {
            if (checkedListBox3.GetItemChecked(i))
            {
                string questionToDelete = checkedListBox3.Items[i].ToString();
                string query = "DELETE FROM Questions WHERE Question =
@Question";
                MySqlCommand command = new MySqlCommand(query,
connection);
                command.Parameters.AddWithValue("@Question",
questionToDelete);
                command.ExecuteNonQuery();
                MessageBox.Show("Питання видалено");
            }
        }

        checkedListBox3.Items.Clear();
        LoadQuestions();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Ошибка: " + ex.Message);
    }
    finally
    {
        connection.Close();
    }
}

private void Settings_Load_1(object sender, EventArgs e)
{

}
```

```
        private void button2_Click(object sender, EventArgs e)
        {
            Settings_theory setteor = new Settings_theory();
            setteor.Show();

        }

        private void button5_Click(object sender, EventArgs e)
        {

            OpenFileDialog openFileDialog = new OpenFileDialog();
            openFileDialog.Filter = "Все файлы (*.*)|*.*";

            if (openFileDialog.ShowDialog() == DialogResult.OK)
            {
                textBox2.Text = openFileDialog.FileName;
            }
        }

        private void button4_Click(object sender, EventArgs e)
        {
            string filePath = textBox2.Text;

            if (File.Exists(filePath))
            {
                AddFileToDatabase(filePath);
            }
            else
            {
                MessageBox.Show("Обраного файлу не існує.");
            }
        }
        private void AddFileToDatabase(string filePath)
        {
            try
            {
                using (MySqlConnection connection = new
MySqlConnection(connectionString))
                {
                    connection.Open();

                    byte[] fileData = File.ReadAllBytes(filePath);

                    string insertQuery = "INSERT INTO Files (FileName, FileData)
VALUES (@FileName, @FileData)";

                    using (MySqlCommand cmd = new MySqlCommand(insertQuery,
connection))
                    {
                        cmd.Parameters.AddWithValue("@FileName",
Path.GetFileName(filePath));
                        cmd.Parameters.AddWithValue("@FileData", fileData);
```

```
                cmd.ExecuteNonQuery();
            }

            connection.Close();
        }

        MessageBox.Show("Файл успішно додано до БД.");
      }
      catch (Exception ex)
      {
        MessageBox.Show("Ошибка: " + ex.Message);
      }
    }
  }
}
```

Settings_theory.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySqlConnector;



namespace ShablonForLabs
{
  public partial class Settings_theory : Form
  {
    private string connectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";

    public Settings_theory()
    {
      InitializeComponent();
    }

    private void button1_Click(object sender, EventArgs e)
    {
      string theoryTitle = txtTheoryTitle.Text;
      string theoryContent = txtTheoryContent.Text;

      AddTheoryToDatabase(theoryTitle, theoryContent);

      txtTheoryTitle.Clear();
```

```
            txtTheoryContent.Clear();
        }

        private void AddTheoryToDatabase(string title, string content)
        {
            using (MySqlConnection connection = new
MySqlConnection(connectionString))
            {
                connection.Open();

                string insertQuery = "INSERT INTO Theory (theoryTitle, theoryContent)
VALUES (@Title, @Content)";
                using (MySqlCommand cmd = new MySqlCommand(insertQuery,
connection))
                {
                    cmd.Parameters.AddWithValue("@Title", title);
                    cmd.Parameters.AddWithValue("@Content", content);
                    cmd.ExecuteNonQuery();
                }

                connection.Close();
            }
        }
```

Tests.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using MySqlConnector;
using System.Windows.Forms;

namespace ShablonForLabs
{
    public partial class tests : Form
    {
        private авторизація авторизація;

        public tests(авторизація авторизація)
        {
            InitializeComponent();
            this.авторизация = авторизація;

        }

        private void textBox1_TextChanged(object sender, EventArgs e)
        {
        }

        private void TestStart_Click(object sender, EventArgs e)
        {
```

```
            string CurrentUserLogin = авторизація.CurrentUserLogin;
            test1 Test = new test1(CurrentUserLogin);
            Test.Show();
        }

        private void tests_Load(object sender, EventArgs e)
        {
            string connectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";

            using (MySqlConnection connection = new
MySqlConnection(connectionString))
            {
                connection.Open();

                string queryTest1 = "SELECT TestName FROM test1";
                using (MySqlCommand commandTest1 = new
MySqlCommand(queryTest1, connection))
                using (MySqlDataReader readerTest1 = commandTest1.ExecuteReader())
                {
                    if (readerTest1.Read())
                    {
                        label2.Text = readerTest1["TestName"].ToString();
                    }
                    else
                    {
                        label2.Text = "Немає даних";
                    }
                }

                string queryTest2 = "SELECT TestName FROM test2";
                using (MySqlCommand commandTest2 = new
MySqlCommand(queryTest2, connection))
                using (MySqlDataReader readerTest2 = commandTest2.ExecuteReader())
                {
                    if (readerTest2.Read())
                    {
                        label3.Text = readerTest2["TestName"].ToString();
                    }
                    else
                    {
                        label3.Text = "Немає даних";
                    }
                }

                string queryTest3 = "SELECT TestName FROM test3";
                using (MySqlCommand commandTest3 = new
MySqlCommand(queryTest3, connection))
                using (MySqlDataReader readerTest3 = commandTest3.ExecuteReader())
                {
                    if (readerTest3.Read())
                    {
```

```
                label5.Text = readerTest3["TestName"].ToString();
            }
            else
            {
                label5.Text = "Немає даних";
            }
        }

    }

    private void button1_Click(object sender, EventArgs e)
    {
        string CurrentUserLogin = авторизація.CurrentUserLogin
        test2 Test2 = new test2(CurrentUserLogin);
        Test2.Show();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        string CurrentUserLogin = авторизація.CurrentUserLogin;
  test3 Test3 = new test3(CurrentUserLogin);
        Test3.Show();
    }
  }
}
```

Test.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.Entity;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Windows.Forms.DataVisualization.Charting;
using MySqlConnector;

namespace ShablonForLabs
{
   public partial class test1 : Form
   {
      private string CurrentUserLogin;


      public test1(string loginUser)
```

```
    {
        InitializeComponent();
        this.CurrentUserLogin = loginUser;
    }



    private void test1_Load(object sender, EventArgs e)
    {
                string connectionStr =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";

        using (MySqlConnection connection = new
MySqlConnection(connectionStr))
        {
            connection.Open();
            string testName = label1.Text;
            string test1Query = $"SELECT TestName, Question FROM test1";

            string questionsQuery = $"SELECT WrongAnswer1, WrongAnswer2,
WrongAnswer3, CorrectAnswer, WrongAnswer4 FROM questions ";

            string test1ConnectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";
string questionsConnectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";

            using (MySqlConnection test1Connection = new
MySqlConnection(test1ConnectionString))
            {


                using (MySqlConnection questionsConnection = new
MySqlConnection(questionsConnectionString))
                {
                    test1Connection.Open();
                    questionsConnection.Open();

                    using (MySqlCommand test1Command = new
MySqlCommand(test1Query, test1Connection))
                    using (MySqlCommand questionsCommand = new
MySqlCommand(questionsQuery, questionsConnection))
                    {
                        using (MySqlDataReader test1Reader =
test1Command.ExecuteReader())
                        using (MySqlDataReader questionsReader =
questionsCommand.ExecuteReader())
                        {
                            if (test1Reader.Read() && questionsReader.Read())
                            {
                                label1.Text = test1Reader["TestName"].ToString();
                                label2.Text = test1Reader["Question"].ToString();
```

```
                        radioButton1.Text =
questionsReader["WrongAnswer1"].ToString();
                        radioButton2.Text =
questionsReader["WrongAnswer2"].ToString();
                        radioButton3.Text =
questionsReader["WrongAnswer3"].ToString();
                        radioButton4.Text =
questionsReader["CorrectAnswer"].ToString();
                        radioButton5.Text =
questionsReader["WrongAnswer4"].ToString();
                    }
                }
            }
        }
    }
    }

    private void button1_Click(object sender, EventArgs e)
    {
        int score = 0;
        string username = CurrentUserLogin;
        if (radioButton4.Checked)          {
            score = 5;
        }
                else
        {
            score = 0;
        }
        string testConnectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";
        string testUpdateQuery = $"UPDATE test SET Score = {score}, Login =
'{username}' WHERE TestName = '{label1.Text}'";


        using (MySqlConnection testConnection = new
MySqlConnection(testConnectionString))
        {
            testConnection.Open();

            using (MySqlCommand testCommand = new
MySqlCommand(testUpdateQuery, testConnection))
            {
                int rowsAffected = testCommand.ExecuteNonQuery();

                if (rowsAffected > 0)
                {
                    MessageBox.Show($"Оценка {score} сохранена в базе данных.");

                }
                else
                {
```

```
                MessageBox.Show("Не удалось сохранить оценку.");

            }
        }
    }

  }

}
```

Theory.cs

```
using MySqlConnector;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Reflection.Emit;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ShablonForLabs
{
  public partial class theory : Form
  {
    public theory()
    {
      InitializeComponent();
    }
    private void theory_Load(object sender, EventArgs e)
    {
      LoadTheoryDataFromDatabase();
    }
    public void LoadTheoryDataFromDatabase()
    {
      {
        MySqlConnection connection = new
MySqlConnection("server=localhost;port=3306;username=root;password=root;data
base=encryptpro");

        try
        {
          connection.Open();

          for (int i = 1; i <= 2; i++) // Проходим по ID 1 и 2
          {
```

```
                string selectQuery = "SELECT theoryTitle, theoryContent FROM
Theory WHERE ID = @ID";

                MySqlCommand command = new MySqlCommand(selectQuery,
connection);
                command.Parameters.AddWithValue("@ID", i);

                using (MySqlDataReader reader = command.ExecuteReader())
                {
                    if (reader.Read())
                    {
                        string theoryTitle = reader["theoryTitle"].ToString();
                        string theoryContent = reader["theoryContent"].ToString();

                        if (i == 1)
                        {
                            existingLabel.Text = theoryTitle;

                            existingTextBox.Text = theoryContent;
                            existingTextBox.ReadOnly = true;
                        }
                        else if (i == 2)
                        {
                            label1.Text = theoryTitle;

                            textBox1.Text = theoryContent;
                            textBox1.ReadOnly = true;
                        }
                    }
                }
            }
            catch (MySqlException ex)
            {
                MessageBox.Show("Помилка завантаження даних з БД: " +
ex.Message);
            }
            finally
            {
                connection.Close();
            }
        }
    }
```

Statistics.cs

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.ComponentModel;
```

```
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Windows.Forms.DataVisualization.Charting;
using MySql.Data.MySqlClient;

namespace Statistic_Frame
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void Form1_Load(object sender, EventArgs e)
      {
         chart2.Hide();
         chart3.Hide();

         string connectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";

         using (MySqlConnection connection = new
MySqlConnection(connectionString))
         {
            connection.Open();

            string query = "SELECT * FROM test"; // Замените на ваш запрос SQL

            DataTable dataTable = new DataTable();

            using (MySqlCommand command = new MySqlCommand(query,
connection))
            using (MySqlDataAdapter adapter = new MySqlDataAdapter(command))
            {
               adapter.Fill(dataTable);
            }

            dataGridView1.DataSource = dataTable;

                        chart1.DataSource = dataTable;
            chart1.Series["Series1"].XValueMember = "Login";
chart1.Series["Series1"].YValueMembers = "Score
      }

      private void chart2_Click(object sender, EventArgs e)
      {
```

```
    }

    private void button3_Click(object sender, EventArgs e)
    {
       LoadChartData1();
       chart1.Hide();
       chart2.Hide();
       chart3.Visible = true;
    }

    private void button2_Click(object sender, EventArgs e)
    {
       LoadChartData();
       chart2.Visible = true;
       chart1.Hide();
       chart3.Hide() ;


    }
    private void LoadChartData()
    {
       string connectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";

       using (MySqlConnection connection = new
MySqlConnection(connectionString))
       {
          connection.Open();


          string query = "SELECT TestName, COUNT(*) AS Count " +
                "FROM test " +
                "WHERE Score IN (3, 2, 5) " +
                "GROUP BY TestName";

          using (MySqlCommand command = new MySqlCommand(query,
connection))
          using (MySqlDataAdapter adapter = new MySqlDataAdapter(command))
          {
             DataTable dataTable = new DataTable();
             adapter.Fill(dataTable);

             chart2.Series.Clear();

             foreach (DataRow row in dataTable.Rows)
             {
                string testName = row["TestName"].ToString();
                int count = Convert.ToInt32(row["Count"]);

                Series series = chart2.Series.Add(testName);
                series.ChartType = SeriesChartType.Pie;
```

```
                series.Points.AddXY("5", count);
                series.Points.AddXY("4", count);
                series.Points.AddXY("3", count);
            }
        }
    }


}
private void LoadChartData1()
{
    string connectionString =
"server=localhost;port=3306;username=root;password=root;database=encryptpro";

    using (MySqlConnection connection = new
MySqlConnection(connectionString))
    {
        connection.Open();

        string query = "SELECT * FROM test";

        using (MySqlCommand command = new MySqlCommand(query,
connection))
        using (MySqlDataAdapter adapter = new MySqlDataAdapter(command))
        {
            DataTable dataTable = new DataTable();
            adapter.Fill(dataTable);

            chart3.Series.Clear();

            // Додавання нової серії для відображення даних
            Series series = new Series();
            series.ChartType = SeriesChartType.Line; // Змінено на Line
            series.Name = "Score";
            series.Color = Color.Purple;
            // Додавання точок даних з DataTable до серії
            foreach (DataRow row in dataTable.Rows)
            {
                string login = row["Login"].ToString();
                int score = Convert.ToInt32(row["Score"]);

                DataPoint point = new DataPoint();
                point.SetValueXY(login, score);
                series.Points.Add(point);
            }

            // Додавання серії до Chart
            chart3.Series.Add(series);

            // Налаштування відображення на осі X
            chart3.ChartAreas[0].AxisX.Title = "Login";
            chart3.ChartAreas[0].AxisX.Interval = 1; // Інтервал між мітками
```

```
                // Налаштування відображення на осі Y
                chart3.ChartAreas[0].AxisY.Title = "Score";
            }
        }
    }

    private void button1_Click(object sender, EventArgs e)
    {
        chart2.Hide();
        chart3.Hide();
        chart1.Visible = true;
    }
  }
}
```

Додаток Б

ПЕРЕЛІК ДОКУМЕНТІВ НА ОПТИЧНОМУ НОСІЇ

Ім'я файла	**Опис**
Пояснювальні документи	
Кваліфікаційна робота.doc	Пояснювальна записка. Документ Word.
Кваліфікаційна робота.pdf	Пояснювальна в форматі PDF
Программа	
Program.rar	Архів. Містить коди програми і откомпільовану програму
Презентація	
Презентація.ppt	Презентація