КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій

_____

(факультет)

інформаційних технологій

_____

(кафедра)

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО КВАЛІФІКАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО РІВНЯ «БАКАЛАВР»

на тему: «Система захисту бази даних за допомогою стандарту шифрування AES»

____ ЖУРАКОВСЬКИЙ МАКСИМ ЕДУАРДОВИЧ_____

(прізвище, ім’я та по батькові студента повністю)

Київ 2024 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

автоматизації і інформаційних технологій

(факультет)
інформаційних технологій

(кафедра)

ЗАТВЕРДЖУЮ
Завідувач кафедри ІТ
к.т.н., доцент Гончаренко Т.А.

_____
„___” _____2024 року

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО КВАЛІФІКАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ НА ЗДОБУТТЯ ОСВІТНЬОГО РІВНЯ «БАКАЛАВР»

на тему: «Система захисту бази даних за допомогою стандарту шифрування AES»

Виконав: студент 4-го курсу, групи КН-20-2.

Спеціальності: 122 «Комп’ютерні науки»

Освітня програма: «Інформаційні і управляючі системи і технології»
(шифр і назва напряму підготовки, спеціальності)

Жураковський М.Е..
(прізвище та ініціали)

Керівник доцент кафедри кібербезпеки та комп’ютерної інженерії, к.т.н., доцент
Вишняков В.М.
(прізвище та ініціали)

Київ 2024 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет: автоматизації і інформаційних технологій.
Випускова кафедра: інформаційних технологій.
Освітній ступінь: «бакалавр» за ОП
Спеціальність: 122 «Комп`ютерні науки».
Освітня програма: Інформаційні управляючі системи і технології

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри ІТ
к.т.н., доцент Гончаренко Т.А.

_____
„___” _____2024 року

ЗАВДАННЯ
ДО ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО РІВНЯ «БАКАЛАВР»

Жураковський Максим Едуардович

1. Тема роботи: Система захисту бази даних за допомогою стандарту шифрування AES
   затверджена наказом ректора КНУБА №433/2 від « 29 » лютого 2024 р.
2. Керівник роботи: Вишняков Володимир Михайлович, доцент кафедри кібербезпеки та комп’ютерної інженерії, кандидат технічних наук, доцент.
3. Строк подання студентом роботи до захисту:
4. Зміст пояснювальної записки за розділами:

Р.1. Аналіз предметної області та постановка задачі.
Р.2. Алгоритмічне та математичне забезпечення.
Р.3. Розробка програмного забезпечення
Р.4. Ергономіка інформаційних технологій

5. Графічний матеріал:

С.1. Вступний слайд
С.2. Загальний опис баз даних
С.3. AES – Advanced encryption standard

С.4. Порівняння розмірів шифрування
С.5. DFD-діаграма
С.6. Алгоритм шифрування
С.7. Алгоритм обробки ключа
С.8. Створення бази даних
С.9. Діаграма класів
С.10. Тестовий приклад, вікно авторизації
С. 11. Тестовий приклад, вікно шифрування
С. 12. Висновки

6. Календарний план виконання атестаційної випускної роботи

Види робіт та їх зміст	Дата виконання
Розділ 1. Аналіз предметної області та постановка задачі	Березень 2024 р.
Тестовий приклад програми	Травень 2024 р.
Розділ 2. Алгоритмічне та математичне забезпечення	Травень 2024 р.
Розділ 3. Розробка програмного забезпечення	Травень 2024 р.
Розділ 4. Ергономіка інформаційних технологій	Травень 2024 р.
Остаточне оформлення роботи	Травень 2024 р.
Направлення роботи на рецензування	Червень 2024 р.
Попередній захист роботи на кафедрі	Червень 2024 р.

7. Дата видачі завдання: 12 січня 2024 р.

Зав. кафедри _____ Гончаренко Т.А.
(підпис) (прізвище та ініціали)

Керівник _____ Вишняков В.М.
(підпис) (прізвище та ініціали)

Виконавець _____ Жураковський М.Е.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Жураковський М.Е. Система захисту бази даних за допомогою стандарту шифрування AES.

Атестаційна випускна робота бакалавра за спеціальністю: 122 «Комп'ютерні науки», спеціалізація: «Інформаційні управляючі системи і технології». – Київський національний університет будівництва та архітектури. – Київ, 2024.

Робота присвячена можливості створення інформаційної системи захисту реляційних баз даних за допомогою стандарту шифрування Advanced Encryption Standard.

Ключові слова: бази даних, шифрування, AES, Rijndael.

SUMMARY

Zhurakovskiy M. E. Database protection system using the AES encryption standard.

Attestation graduation work of the bachelor in the specialty: 122 "Computer science", specialization: "Information management systems and technologies". – Kyiv National University of Construction and Architecture. – Kyiv, 2024.

The work is devoted to the possibility of creating an information system for the protection of relational databases using the Advanced Encryption Standard.

Keywords: databases, encryption, AES, Rijndael.

ЗМІСТ

Вступ

В сучасному інформаційному середовищі безпека даних стає все більшою проблемою. Запобігання несанкціонованому доступу до конфіденційної інформації, як правило, вимагає застосування надійних методів шифрування. У цьому контексті системи захисту баз даних набувають особливої важливості, оскільки вони забезпечують збереження конфіденційності, цілісності та доступності даних для авторизованих користувачів.

Також не слід забувати про широке використання баз даних у різних сферах діяльності, включаючи фінансові установи, медичні заклади, урядові та військові органи та багато інших. Забезпечення безпеки даних у цих системах стає критично важливим для захисту від фінансових втрат, порушення конфіденційності та збереження довіри споживачів і клієнтів.

Одним із найефективніших та широко використовуваних стандартів шифрування є Advanced Encryption Standard (AES). AES, розроблений замість застарілого стандарту DES, володіє високою стійкістю до криптоаналітичних атак і є рекомендованим для захисту конфіденційних даних в різноманітних областях, включаючи інформаційні технології, фінансові послуги, медицину та інші.

У даній роботі розглядається роль та ефективність систем захисту баз даних, які використовують стандарт шифрування AES. Досліджується принцип застосування AES для шифрування даних в базі даних, а також оцінюються переваги та обмеження такого підходу в контексті забезпечення безпеки інформації. Крім того, розглядаються практичні аспекти імплементації системи захисту баз даних з використанням AES, включаючи вибір ключових параметрів, управління ключами та управління доступом користувачів.

Метою даної роботи є створення системи захисту баз даних, якою можна буде користуватись та впроваджувати в ще більші системи, роблячи таким чином повноцінне працююче програмне забезпечення із зручним інтерфейсом.

Серед методів дослідження, які будуть використані у роботі, слід виділити наступні теоретичні методи: аналіз, співставлення та моделювання. Серед емпіричних методів використані: експеримент та вивчення теоретичного матеріалу.

Таким чином, потрібно буде розібратись, чому застосування систем захисту баз даних з використанням AES є необхідним елементом сучасної стратегії кібербезпеки, який сприяє забезпеченню конфіденційності, цілісності та доступності даних у вимогливому інформаційному середовищі.

Перелік умовних позначень

AES – Advanced Encryption Standard;

DES – Data Encryption Standard;

SQL – Structured Query Language;

БД – база даних;

WPF – Windows Presentation Foundation;

XAML – eXtensible Application Markup Language.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Опис предметної області

Предметною областю даної дипломної роботи є захист баз даних за допомогою стандарту шифрування AES. Кожна система БД має бути захищена від злому та викрадення інформації, яка зберігається. Для цього можна використовувати велику кількість підходів, наприклад: шифрування, хешування та різноманітні протоколи автентифікації.

А що таке база даних? Відсутнє якесь одне чітке визначення, але більшість з них мають спільні ознаки, такі як:

- Має зберігатися та оброблятися в обчислювальній системі, тобто ніякі реальні сховища даних не відносяться до БД;
- Всі елементи мають бути структурованими та пов’язаними один з одним для подальшого полегшення пошуку та обробки наявної інформації;
- Включає в себе схему або набір метаданих, за якими створюється та змінюється БД.

Бази даних розділяються на декілька типів: реляційні, нереляційні та змішані. Для кожного з них може підійти лише свій спосіб захисту інформації.

Реляційні БД – зберігають дані в таблицях зі стовпцями (атрибут) та строками (інстанс). Кожній таблиці присвоюється ідентифікатор, за яким в майбутньому можна встановити зв’язок між декількома таблицями. Доступ до даних отримується за допомогою SQL-запитів. (рис. 1.1)

Patient's ID	Full name	Policy	Medical histo	Photo	District's num
1	Грозовой Игорь Павлович	1	Текст1	NULL	1
2	Иегова Инна Петровна	1	Текст2	NULL	2
3	Афанасьев Антон Денисович	1	Текст3	NULL	2
4	Воробьёв Богдан Игоревич	1	Текст4	NULL	5
5	Тростков Леонид Геннадиевич	1	Текст5	NULL	5
6	Простакова Надежда Ивановна	0	Текст6	NULL	9
7	Рыльский Иван Богданович	1	Текст7	NULL	3
8	Фильский Сергей Антонович	1	Текст8	NULL	4
9	Глинко Андрей Андреевич	0	Текст9	NULL	6
10	Рабсинов Роман Игоревич	1	Текст10	NULL	8

Рис. 1.1 Приклад реляційної бази даних

Structured Query Language (SQL, мова структурованих запитів) – інформаційно-логічна мова програмування, за допомогою якої створюються, редагуються та керуються бази даних. Сучасний розвиток дозволяє не тільки робити ці базові функції, а й додавати тригери, представлення, різноманітні процедури, тобто SQL все більше перетворюється на класичну мову програмування. (рис. 1.2) [1]

```
DELIMITER//
ALTER TABLE districts ADD CONSTRAINT
ch2 CHECK(Number_of_morgues<Number_of_hospitals);
// DELIMITER

INSERT INTO districts(Districts_number, Number_of_ill_people, Number_of_hospitals, Number_of_morgues, Number_of_ambulance_
(11, 56, 5, 3, 4)
```

1 1_2 1_1 2

Запрос выполнен успешно, затронуто записей: 10 (31 мс)

Запрос выполнен успешно, затронуто записей: 1 (15 мс)

Рис. 1.2 Приклад виконання SQL-запиту

Прикладами основних реляційних БД є Microsoft SQL Server, Oracle, MySQL та Postgresql.

Нереляційні БД – зберігають інформацію за допомогою моделей, які зосереджені на якихось певних типах даних та оптимізовані для їх файного

збереження, тобто існує декілька видів NoSQL БД, наприклад, документові (рис. 1.4) або графові (рис. 1.3).

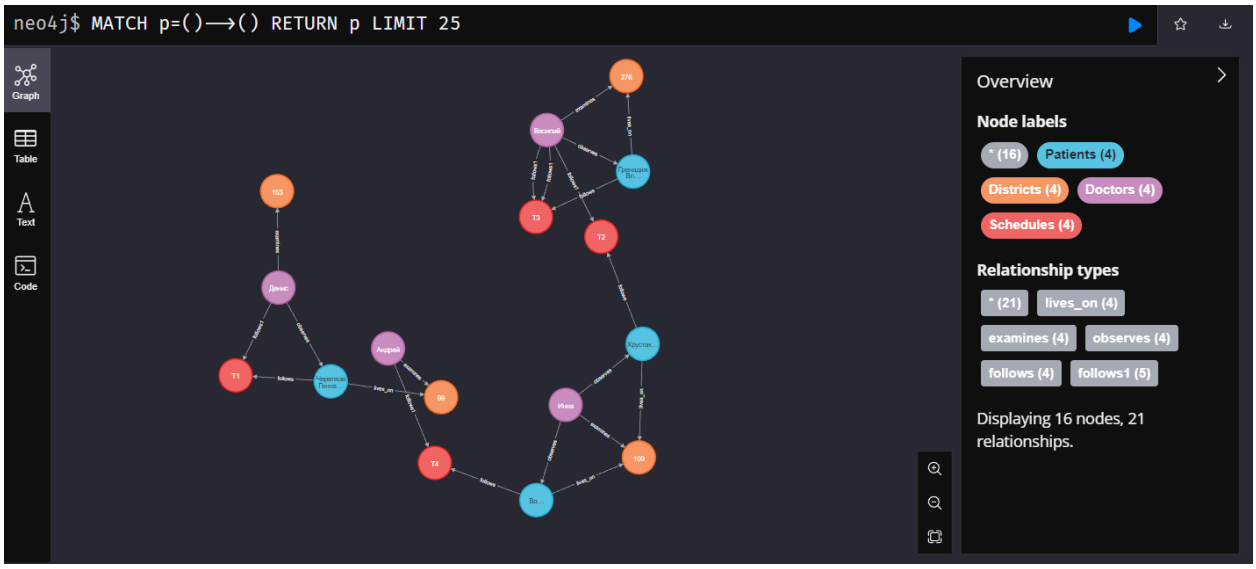


Рис. 1.3 Приклад нереляційної графової бази даних

Рис. 1.4 Приклад нереляційної документової бази даних

Поверхнево оглянувши типи БД, можна сказати, що для захисту за допомогою AES ліпше використати одну з реляційних баз даних.

Тепер, коли було надано визначення, що саме ми маємо захищати, можна зазначити, як шифрують дані для їх захисту.

При підготовці до створення системи шифрування даних слід звернути увагу на деякі аспекти, які мають великий вплив на подальшу роботу:

- Вибір системи шифрування;
- Захист ключів шифрування – вони мають бути добре збережені та керуватися за допомогою відповідних протоколів;
- Захист ключів доступу – на рівні із ключами шифрування, надання доступу до даних є не менш важливою операцією;
- Шифрування в покої – захист даних не тільки під час користування ними, а й при їх збережені та передачі;
- Аудит та моніторинг – спостереження за статусом захисту даних та запобігання можливих спроб злому;
- Підтримка безпеки на рівні додатків – захист даних не лише на рівні БД, а й в тій системі, де використовується шифрована інформація.

Розвиток захисту баз даних відбувався паралельно самому зросту баз даних. Одними з основних типів були базові методи автентифікації та верифікації – найпростіший вид, який дозволяв користувачам працювати лише в межах своїх прав доступу. Щодо саме шифрування, перші методи використовувались на рівні або колонок, або файлів.

Зараз використовуються не лише ці способи, більшість з них буде описана далі.

В колонкових шифрах ховались лише якісь вибіркові дані (ідентифікатори, паролі, особисті дані тощо), залишаючи можливість доступу до іншої інформації, яка була не такою основною.

Транзакційні журнали – це записи всіх змін даних, які були проведені під час роботи в базі. Вони є дуже суттєвими, коли треба відновити якусь інформацію, а їх шифрування дозволяє уникнути витоку та несанкціонованого доступу до даних.

При шифруванні з’єднання (наприклад, шифрування Secure Sockets Layer, SSL) виконується захист даних між сервером та браузером, тобто, все, що йде від клієнта до сервера захищається, що, в свою чергу, надає конфіденційності та зберігає цілісність даних.

Бази даних не можуть існувати без резервних копій для відновлення після можливих збоїв, які можуть знаходитись на хмарних сервісах або зовнішніх носіях та містити в собі конфіденційні дані. Саме ці дані теж потрібно шифрувати при їх створенні та загрузці.

Якщо шифрування йде на файловому рівні, то воно спрацьовує перед збереженням БД на диск, охороняючи всю записану інформацію, а потім дані повністю розшифровуються для відображення користувачеві. Найпопулярнішими зараз способами такого типу шифрування файлів, є симетричні алгоритми DES та AES.

Усі описані методи мають свої недоліки та переваги, наприклад, перший варіант є більш гнучким за доступом до даних, але менш захищеним, ніж останній. Для цієї роботи був обраний саме він, а саме, шифрування за допомогою AES.

1.2 Аналіз об’єкта дослідження

Захист інформації – сукупність методів і засобів, що забезпечують цілісність, конфіденційність і доступність інформації за умов впливу на неї загроз природного або штучного характеру, реалізація яких може призвести до завдання шкоди власникам і користувачам інформації. [2]

Захист інформації ведеться для підтримки таких властивостей інформації як:

- Конфіденційність – неавторизований (або без відповідних прав) користувач не може мати доступ до вже попередньо створеного набору даних.

- Цілісність – неавторизований (або без відповідних прав) користувач не може модифікувати вже створений набір даних.
- Доступність – авторизований користувач має право на доступ та редагування інформації в базі даних згідно встановленим правилам протягом заданого часу [2].

Якою є загроза, для запобігання якої, треба дотримуватись правил, описаних вище? Відповідно цим властивостям, це:

- Загрози конфіденційності:
  - несанкціонований доступ (НСД);
  - витік;
  - розголошення.
- Загрози цілісності:
  - знищення;
  - модифікація;
- Загрози доступності:
  - блокування;
  - знищення;

Для захисту від описаних загроз існують різні види захисту інформації:

- **Технічний** – забезпечує обмеження доступу до носія повідомлення апаратно-технічними засобами (антивіруси, фаєрволи, маршрутизатори, токіни, смарткарти тощо):
  - попередження витоку по технічним каналам;
  - запобігання блокуванню ;
- **Інженерний** – запобігає руйнуванню носія внаслідок навмисних дій або природного впливу інженерно-технічними засобами (до такого виду відносять обмежувальні конструкції, охоронно-пожежну сигналізацію).

- **Криптографічний** – попереджує доступ за допомогою математичних перетворень повідомлення:
  - попередження несанкціонованої модифікації ;
  - попередження НС розголошення.
- **Організаційний** – попередження доступу на об'єкт інформаційної діяльності сторонніх осіб за допомогою організаційних заходів (правила розмежування доступу). [2]

Для захисту інформації на рівні прикладного та системного програмного забезпечення зазвичай використовуються такі системи:

- розмежування доступу до інформації – коли користувачі або процеси намагаються одержати доступ до пасивних об'єктів, механізми, що реалізують керування доступом, на підставі політики безпеки і перевірки атрибутів доступу можуть «ухвалити рішення» про легальність запиту;
- ідентифікації та автентифікації – процедури встановлення належності користувачеві інформації в системі пред'явленого ним ідентифікатора;
- аудиту та моніторингу – процес безперервного або регулярного (періодичного) збору інформації про стан певних параметрів, які обираються для дослідження та належать певній робочий системі з навколишнім середовищем;
- антивірусного захисту – спеціалізований набір додатків для знаходження комп'ютерних вірусів, а також небажаних (шкідливих) програм загалом, відновлення заражених (модифікованих) такими програмами файлів, а також для профілактики – запобігання зараженню (модифікації) файлів шкідливим кодом. [2]

1.3 Опис предмету дослідження

Алгоритм Rijndael розроблений бельгійськими фахівцями Joan Daemen (Proton World International) та Vincent Rijmen (Katholieke Universiteit Leuven). Цей шифр переміг у проведеному Національним інститутом стандартів і техніки (NIST) США конкурсі на звання AES (Advanced Encryption Standard) і в 2001 році був прийнятий як новий американський стандарт.

Такий стандарт шифрування, як AES, відноситься до симетричних алгоритмів блочного шифрування. Він став наслідником не менш популярного та добре працюючого стандарту DES, який до сих пір не втрачає своєї значимості, але вже сприймається, як більш застарілий алгоритм.

В основі AES лежить шифр Rijndael, розмір блоку якого є 128 біт, тобто він може зашифрувати набір даних на цей розмір. Довжина ключа різниться – вона може бути 128, 192 та 256-бітною, що впливає на кількість раундів – 10, 12 або 14 відповідно. Кількість раундів, або ітерацій шифрування, впливає як на доступність злому захисту, так і на майбутню можливість розшифрування інформації. [3]

AES складається з лінійно-підстановних перетворень, які широко використовують табличні обчислення, а всі необхідні для цього таблиці задаються константою, тобто їм не потрібно залежати ні від ключа, ні від самих даних.

Блоки даних, що обробляється з використанням Rijndael/AES, поділяються на масиви байтів, і кожна операція шифрування є байт-орієнтованою. Кожен раунд складається з трьох різних перетворень, які можна обернути. Їх називають шарами:

1. **Нелінійний шар**, в якому виконується заміна байтів. Шар реалізований за допомогою S-блоків (рис. 1.5), що мають оптимальну нелінійність, і запобігає можливості використання лінійного, диференціального та інших сучасних методів

криптоаналізу. Для кожного з трьох n-бітних ключів S-блок є сталим.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Рис. 1.5 S-блок для 128-бітного ключа

Для вхідних бітів 4 і B вихідним значенням буде B3.

2. **Лінійний шар, що перемішує**, гарантує високий ступінь взаємопроникнення символів блоку для маскування статистичних зв'язків. На цьому шарі прямокутному масиві байтів виконується зсув рядків масиву і перестановка стовпців.
3. **Шар додавання за модулем 2 з підключем** безпосередньо виконує шифрування.

Шифр починається і закінчується додаванням ключа. Це дозволяє закрити вхід першого раунду при атаці на вже відомий текст, і зробити криптографічно значущим результат останнього раунду.

Для розшифрування даних, всі попередньо описані етапи треба пройти в зворотному порядку.

AES, як учасник конкурсу, проходила перевірку на стійкість. Після цього, у 2003 році, Агентством національної безпеки США було постановлено, що шифр є достатньо надійним для захисту державних даних рівнів Secret та Top Secret. [4]

До та після цього проводились спроби провести успішну атаку, яка б лишила Reijndael його статусу. Однією з найвідоміших була **XSL-атака**.

XSL-атака – вид кібератаки, що використовує мову XSLT (Extensible Stylesheet Language Transformations) для виконання шкідливого коду на вразливих системах. Ця мова призначена для перетворення XML-документів на різні формати, такі як HTML, текст або змінений XML.

Основними механізмами XSL-атаки є:

- Впровадження шкідливого XSLT.

Атакуючий може впровадити шкідливий XSLT в систему через вразливість, пов'язану з обробкою XML та XSLT-документів. Наприклад, якщо веб-додаток дозволяє користувачам завантажувати XSLT-файли для обробки XML, зловмисник може завантажити спеціально створений XSLT-файл, що містить шкідливий код.

- Використання розширень XSLT.

XSLT підтримує розширення, які дозволяють виконувати складні операції, включаючи виконання скриптів та команд на сервері. Зловмисник може використовувати ці розширення для виконання довільного коду на сервері, отримавши цим доступ до даних або контроль над системою.

Відмінності AES від більшості інших шифрів полягають у його простому математичному описі. Ця особливість становила причину занепокоєння для Нільса Фергюсона, відомого нідерландського криптографа та консультанта компанії Microsoft. У своїх дослідженнях він вказав, що безпека шифру

ґрунтується на новому, ще не перевіреному припущенні про складність розв'язання певних видів рівнянь. Також Брюс Шнайєр, який є автором спільної з Нільсом книги, висловив своє стурбоване ставлення до AES, зауваживши:

«Ми маємо одне критичне зауваження до AES: ми не зовсім довіряємо його безпеці. Що турбує нас найбільше в AES, так це його проста алгебраїчна структура… Жоден інший блоковий шифр не має такого простого представлення алгебри. Ми гадки не маємо, веде це до атаки чи ні, але незнання цього є достатньою причиною, щоб скептично ставитися до використання AES.» [5]

У 2002 році Ніколя Куртуа та Йозеф Пепшик опублікували статтю, у якій вони описали теоретичну атаку, відому як XSL-атака. Ця атака мала потенціал розкрити шифри AES та Serpent. Проте результати їхньої роботи не були сприйняті всіма оптимістично:

«Я вважаю, що у роботі Куртуа-Пепшика є помилка. Вони переоцінили кількість лінійно-незалежних рівнянь. В результаті вони не мають достатньої кількості лінійних рівнянь для вирішення системи, і [вказаний] метод не може зламати Rijndael. Він має певні переваги і заслуговує на вивчення, але не зламує Rijndael у його нинішньому вигляді.» [6]

На сторінці, присвяченій обговоренню конкурсу NESSIE, наприкінці 2002 року один із авторів шифру, Вінсент Реймен, висловив думку, що XSL-атака є лише мрією (це було пізніше підтверджено в 2004 році на 4-й конференції AES у Бонні). На це Ніколя Куртуа відповів, наголосивши, що ця "мрія" може стати для автора AES жахом. [7]

У 2003 році Шон Мерфі та Метт Робшоу опублікували роботу, в якій вони обґрунтували можливість атаки на алгоритм AES, що скорочує кількість операцій для злому з 2128 до 2100. Однак на 4-й конференції AES Ілля Толі та Альберто Дзаноні показали, що робота Мерфі та Робшоу невірна. Пізніше, у 2007 році, Чу-Ві Лім та Хунгмінг Ху також показали, що ця атака не може працювати у тому вигляді, як вона була описана. [8]

Другою спробою стала так звана **атака по стороннім каналам**.

На відміну від XSL-атак, атаки сторонніми каналами не пов'язані з математичними особливостями шифру, але використовують певні особливості в практичній реалізації систем, які використовують дані шифри, з метою розкрити секретні дані, в тому числі ключі. Декілька подібних атак було проведено на системи, які користувались AES.

У квітні 2005 року Деніел Бернштейн опублікував роботу, в якій описав атаку, що використовує інформацію про час виконання кожної операції шифрування для злому. Ця атака потребувала понад 200 мільйонів обраних шифротекстів для визначення ключа.

У жовтні 2005 року Даг Арне Освік, Аді Шамір та Еран Трумер представили роботу, в якій описали кілька атак, що використовують час виконання операцій для знаходження ключа. Одна з цих атак здобула ключ після 800 операцій шифрування. Для успішної реалізації атаки криптоаналітик повинен мати можливість запускати програми на системі, де виконується шифрування.

У грудні 2009 року була опублікована робота, в якій використання диференціального аналізу помилок, штучно створених у матриці стану на 8-му раунді шифрування, дозволило відновити ключ за 2^{32} операцій.

1.4 Аналіз актуальності

У сучасному цифровому світі практично кожен додаток вимагає створення бази даних для зберігання, обробки та управління інформацією. З урахуванням великої кількості даних, які проходять через сервери та взаємодіють з користувачами, виникає необхідність у забезпеченні безпеки цієї інформації від будь-яких можливих загроз.

Одним із найефективніших методів захисту є використання вже існуючих стандартів шифрування, таких як AES (Advanced Encryption Standard). Ці

стандарти шифрування пройшли великий відбір та тестування, що довело їх надійність, швидкість та гнучкість. Застосування стандартів шифрування забезпечує високий рівень захисту даних під час їх передачі, зберігання та обробки, що дозволяє зберігати конфіденційність і цілісність інформації.

Такий підхід до захисту баз даних стає важливим елементом безпеки в сучасному ІТ-просторі, де кіберзагрози стають все більш складними та розповсюдженими. Отже, використання стандартів шифрування, таких як AES, є важливим етапом в забезпеченні безпеки та конфіденційності даних у всіх сферах інформаційних технологій.

Доречі, з розвитком інформаційних технологій спостерігається зростання рівня кіберзлочинності, що ставить перед великими та маленькими компаніями завдання забезпечення надійного захисту баз даних. Це вимагає наявності спеціалізованих фахівців, які розбираються у принципах та методах захисту інформації, зокрема у стандартизованих підходах до шифрування даних.

Багато країн встановлюють строгі вимоги до захисту будь-якої інформації, що може бути пов’язана з їхніми громадянами або іншими сутностями. Ці вимоги охоплюють різноманітні сфери, такі як медичні заклади, державні структури, банківська сфера, платіжні системи та інші. Наявність кваліфікованих спеціалістів, які розуміють стандарти та методи захисту даних, стає важливим фактором для дотримання цих вимог та забезпечення безпеки інформації.

Поступовий перехід до хмарних сервісів, що є популярним серед як простих користувачів, так і лідерів великих корпорацій, неминуче вимагає високого рівня захисту та шифрування всіх даних, що передаються через сервери. Цей факт підкреслює необхідність надійності та безпеки, яка виникає як серед простих користувачів, так і серед фахівців та експертів у галузі кібербезпеки.

Шифрування даних є одним із ключових аспектів забезпечення безпеки в хмарних сервісах. Безперечним вибором для багатьох стає алгоритм AES, який відомий своєю надійністю та високою ефективністю. Цей алгоритм є предметом довіри не лише серед фахівців у галузі інформаційної безпеки, а й серед широкого кола користувачів, оскільки його ефективність була доведена практикою та численними дослідженнями.

Таким чином, використання алгоритму AES для захисту даних в хмарних сервісах відображає загальну тенденцію до застосування надійних та перевірених рішень в сфері кібербезпеки, що забезпечує високий рівень захисту конфіденційної інформації.

1.5 Стан вже існуючих рішень

Як дуже популярний та затребуваний спосіб захисту даних, алгоритм шифрування AES використовується багатьма компаніями, а вони, в свою чергу, додають цей метод до своїх додатків.

Серед найвідоміших є:

- Популярні месенджери та додатки для обміну повідомленнями: WhatsApp, Signal, Telegram. В них можна захищати дані мобільних телефонів, записи текстів та багато іншого.
- Електронні поштові сервіси: Gmail, Outlook, ProtonMail. В них можна захищати реєстраційні дані та листи.
- Онлайн-банкінг та фінансові сервіси: PayPal, Revolut, TransferWise. В них можна захищати реєстраційні дані, дані карток, фінансові записи транзакцій тощо.
- Хмарні сховища та файлообмінні сервіси: Dropbox, Google Drive, Microsoft OneDrive. В них можна захищати реєстраційні дані, файли.

- VPN-сервіси: NordVPN, ExpressVPN, CyberGhost. В них можна захищати реєстраційні дані, історію з’єднань, налаштування серверів.
- Онлайн-магазини та платіжні системи: Amazon, eBay, Visa, Mastercard. В них можна захищати реєстраційні дані клієнтів, дані про наявні та відсутні продукти, грошові транзакції, історію запитів та ін.
- Системи для відеоконференцій та віддаленого робочого місця: Zoom, Microsoft Teams, Slack. В них можна захищати реєстраційні дані клієнтів, записи дзвінків, їх дати, тривалість тощо.
- Сервіси для зберігання та обробки медичної інформації: Epic, Cerner, HealthVault. В них можна захищати реєстраційні дані пацієнтів, дані докторів, інформацію про медичні послуги та ін.

На основі AES було створено декілька інших способів шифрування – серед них Anubis та Grand Cru.

Anubis – це створений автором Reijndael, Вінсентом Рейменом, ще один симетричний блочний алгоритм. В ньому за замовчуванням використовується 128-бітний ключ і 128-бітний блок, але розмір може зростати до 320-бітного за допомогою застосування на кожні нові 32 біти ще одного раунду шифрування.

Варіативність полягає в тому, що деякі процедури, а саме: операції над S-блоками та матрицями змішування байтів кожного стовбця, в алгоритмі є інволюційними (оберненими самім собі). Таке рішення дозволяє використовувати пункти роботи алгоритму як для шифрування, так і для дешифрування. Це дозволяє Anubis реалізовувати шифр на обладнанні низької цінової категорії.

Існує дві версії цього алгоритму – із випадковим S-блоком та фіксованим, відповідно класичний варіант та tweaked.

Найоптимальнішим зломом Anubis є повний перебір всіх ключів, тобто кількістю в 2^{127} варіантів, що не представляється можливим на сучасному обладнанні.

Grand Cru – є також симетричним блочним алгоритмом (що є прямим наслідком), який розробив Йохан Борст. Це шифрування стало посиленою та глибоко модифікованою версією свого попередника. Для нього використовується 128-бітний ключ та 128-бітний блок.

В алгоритмі Rijndael з чотирьох перетворень даних у раунді лише одна операція – накладання підключення операцією XOR – залежить від ключа. У алгоритмі Grand Cru збільшення кількості ключових перетворень у раунді зміцнює криптостійкість при тій же кількості раундів. Раунд Grand Cru відрізняється від раунду Rijndael додаванням двох ключових операцій замість однієї безключової.

Як заявляє автор алгоритму, його безпечність дорівнює безпечності Rijndael, а, так як для нього не існує ніяких працездатних атак, то й це шифрування неможливо зламати. Окрім цього, через те, що його математична структура більш складна, які-небудь XSL-атаки стають ще складнішими у реалізації.

Не зважаючи на його дуже добрий захист даних та високу надійність, швидкість шифрування Grand Cru не є достатньо високою для використання у великих системах, які обробляють багато наборів даних.

1.6 Визначення цілей дослідження та постановка задачі

Метою дослідження цієї роботи є створення системи захисту баз даних за допомогою існуючого стандарту шифрування AES для детального ознайомлення із його можливостями та загального навчання використання Rijndael.

Рис. 1.6 Дерево цілей системи захисту та шифрування

Для відтворення у дійсність мети треба дотримуватись виконання поставлених цілей (рис. 1.6):

- Обрати систему управління базами даних, мову програмування та спосіб шифрування даних для подальшої розробки майбутнього додатку;
- Створити базовий інтерфейс додатку та його основні функції;
- Створити та додати до програми декілька баз даних, перевірити, що користувач може ними управляти;
- Реалізувати генерацію ключей та запрограмувати шифрування даних для БД;
- Об’єднати всі розробки в єдину систему захисту баз даних.

Об’єктом дослідження роботи є системи захисту інформації.

Предметом дослідження роботи є створення систем шифрування інформації в базі даних за допомогою стандартизованих алгоритмів (зокрема виділяється AES).

Система є базою даних.

Зовнішнім середовищем є набір впорядкованих та об’єднаних за обраним принципом даних, захищений алгоритмом шифрування.

Тема захисту даних за допомогою AES є дуже розвинутою, але, як завжди потребується мати збережену інформацію, так саме є потреба в дослідженні алгоритмів шифрування та засобів для відповідного використання.

2. АЛГОРИТМИЧНЕ ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Математичне забезпечення системи

В основі всього обраного процесу шифрування лежить вже існуючий алгоритм. Ми вже доторкалися до його мінімального опису в першому розділі, але тепер буде докладно описано все, що зв’язано з AES.

Для виконання алгоритму, треба мати розуміння відповідних понять (табл. 2.1) та допоміжних процедур (табл. 2.2). [9]

Таблиця 2.1 Поняття

Назва	Опис
Block	Послідовність біт, з яких складається input, output, State та Round Key. Також в якості Block може бути послідовність байтів.
Cipher Key	Секретний криптографічний ключ, який використовується процедурою Key Expansion, щоб зробити набір раундових ключів (Round Keys); може бути представлений як прямокутний масив байтів, що має чотири рядки і Nk колонок.
Ciphertext	Вихідні дані алгоритму шифрування.
Key Expansion	Процедура генерації Round Keys із Cipher Key.
Round Key	Round Keys отримуються з Cipher Key за допомогою використання процедури Key Expansion. Вони

	застосовуються до State при шифруванні та дешифруванні.

Продовження таблиці 2.1 Поняття

Назва	Опис
State	Проміжний результат шифрування, який може бути представлений прямокутним масивом байтів, що має 4 рядки і Nb колонок.
S-box	Нелінійна таблиця замін, що використовується в кількох трансформаціях заміни байтів та у процедурі Key Expansion для взаємнооднозначної заміни значення байту.
Nb	Число стовбців (32-бітних слів), які складають State. Для AES це значення дорівнює 4.
Nk	Число 32-бітних слів, які складають ключ шифру. Для AES це значення дорівнює 4, 6, або 8.
Nr	Число раундів, яке є функцією *Nk* та *Nb*. Для AES це значення дорівнює 10, 12 або 14.
Rcon[]	Масив, який складається з бітів 32-разрядного слова та є постійним для даного раунду.

Rcon (Round Constants) – це масив констант, що використовуються в процесі розширення ключа в алгоритмі шифрування AES (Advanced Encryption Standard). Цей масив містить значення, які додаються до раундових ключів на кожному раунді шифрування для забезпечення унікальності та непередбачуваності ключів.

AES ключ розширюється за допомогою процесу, який називається розширенням ключа (Key Expansion). Він починається з основного ключа, який користувач надає алгоритму.

Rcon[] є масивом, що містить заздалегідь певні значення, що використовуються в процесі розширення ключа. Він має розмірність 4x10 для AES-128, 4x8 для AES-192 та 4x7 для AES-256. Кожне значення в масиві Rcon[] розраховується за такою формулою:

$$R_{con}[i]=(RC[i],0,0,0), \quad (2.1)$$

де RC[i] – деяке значення з наперед визначеної таблиці, яке залежить від поточного індексу і. Ці значення підбираються таким чином, щоб забезпечити різноманітність та непередбачуваність ключових матеріалів на кожному раунді розширення ключа.

Використання Rcon[] у розширенні ключа допомагає посилити безпеку алгоритму, роблячи розклад ключів, що використовуються на кожному раунді шифрування менш схильним до атак. Замість простого повторювання або залежності тільки від основного ключа, кожен раундовий ключ отримує унікальне доповнення у вигляді значень з масиву Rcon[], що забезпечує додатковий рівень складності та надійності.

Таблиця 2.2 Допоміжні процедури

Назва	Опис
AddRoundKey()	Трансформація при шифровані та оберненому шифровані, під час якої Round Key складається

	за модулем два зі State. Довжина RoundKey дорівнює розміру State (тобто, якщо Nb = 4, то довжина RoundKey дорівнює 128 біт або 16 байт).

Продовження таблиці 2.2 Допоміжні процедури

Назва	Опис
InvMixColumns()	Трансформація при розшифруванні, яка є зворотною до MixColumns().
InvShiftRows()	Трансформація при розшифруванні, яка є зворотною до ShiftRows().
InvSubBytes()	Трансформація при розшифруванні, яка є зворотною до SubBytes().
MixColumns()	Трансформація при шифровані, яка бере усі стовбці State та змішує їх дані (незалежно один від одного), щоб отримати нові стовбці.
RotWord()	Функція, яка використовується в процедурі Key Expansion; бере 4-байтове слово та проводить над ним циклічну перестановку.

ShiftRows()	Трансформації при шифровані, які обробляють State, циклічно змішуючи останні три строки State на різні величини.
SubBytes()	Трансформації при шифровані, які обробляють State, використовуючи нелінійну таблицю заміщення байтів (S-box), застосовуючи її незалежно до кожного байту State.

Продовження таблиці 2.2 Допоміжні процедури

Назва	Опис
SubWord()	Функція, яка використовується в процедурі Key Expansion; вона бере на вході 4-байтове слово та, застосовуючи S-box до кожного з чотирьох байтів, видає вихідне слово.

2.2 Опис проектних рішень з математичного забезпечення

Від розміру ключа в стандарті шифрування AES залежить, наскільки добре буде захищена інформація. Усього є три існуючих для цього варіанти: 128, 192 та 256 бітів або 16, 24 та 32 байтів відповідно (табл. 2.3).

Також розмір впливає на число 32-бітних слів, число раундів та Rcon[] (табл. 2.1). Занадто великі значення може лише погіршити стан використання програми але полегшити можливість злому бази даних.

Табл. 2.3 Порівняння шифрування різних розмірів

Назва	Довжина	Рівень безпеки	Швидкість та продуктивність
AES-128	128 бітів	Забезпечує стандартний рівень безпеки. Цього зазвичай достатньо більшості додатків, особливо якщо ключі добре захищені і використовуються правильні методи криптографічної реалізації.	Працює швидше, ніж 192-бітне та 256-бітне шифрування, оскільки операції шифрування та розшифрування вимагають менше обчислювальних ресурсів.
AES-192	192 біти	Забезпечує підвищений рівень безпеки, ніж 128-бітове шифрування. Це може бути корисним у більш критичних областях, де потрібна додаткова стійкість до атак.	Працює повільно через додатковий обсяг обчислень, необхідних для обробки більш довгих ключів.

Продовження таблиці 2.3 Порівняння шифрування різних розмірів

Назва	Довжина	Рівень безпеки	Швидкість та продуктивність
AES-256	256 бітів	Забезпечує максимальний рівень безпеки серед	Працює повільно через додатковий

		трьох варіантів. Цей рівень безпеки може бути корисним у вкрай чутливих областях, таких як урядові або військові системи, де потрібна максимальна стійкість до криптоаналітичних атак.	обсяг обчислень, необхідних для обробки більш довгих ключів.

```
protected Aes()
{
   LegalBlockSizesValue = s_legalBlockSizes.CloneKeySizesArray();
   LegalKeySizesValue = s_legalKeySizes.CloneKeySizesArray();

   BlockSizeValue = 128;
   FeedbackSizeValue = 8;
   KeySizeValue = 256;
   ModeValue = CipherMode.CBC;
} [10]
```

Вбудований клас AES має розмір ключа, відомий як keysize, рівний 256 бітам. Незважаючи на це, при безпосередньому програмному шифруванні дозволяється використовувати ключі меншого розміру, що дає можливість більшої гнучкості та адаптації при виборі. Такий підхід не обмежується конкретним програмним забезпеченням або обмеженнями, які можуть існувати в системі.

Розглянувши всі переваги та недоліки зазначених варіантів, було вирішено обрати ключ шифрування довжиною 256 бітів, бо, при використанні шифрування для захисту баз даних перевагу слід віддати рівню її захищеності.

При кожній передачі даних має створюватися новий ключ. Для цього треба обрати метод його генерації. [11]

- **Випадкова генерація.**

Ключі генеруються за допомогою випадкового процесу або генератора випадкових чисел. Цей метод може бути використаний для створення випадкових ключів з великим ступенем непередбачуваності.

- **Парольне посилення.**

Використання паролів або парольних фраз, введених користувачем, для генерації ключів шляхом застосування алгоритмів хешування, наприклад, PBKDF2, bcrypt або scrypt.

- **Генерація заснована на хешуванні.**

Використання значення або хеша певних даних або фрази, які можуть бути введені користувачем, для створення ключа. Наприклад, можна використовувати HMAC-SHA256 для генерації ключа з вхідного паролю.

- **Ключовий обмін.**

Використання протоколів ключового обміну, таких як Діффі-Хеллмана, для безпечного обміну ключами між двома або більше сторонами. Ключі генеруються на основі обмінених секретних даних.

- **Ключовий генератор.**

Використання спеціалізованих алгоритмів генерації ключів, таких як генератори випадкових чисел або алгоритми, які базуються на фізичних властивостях пристрою, для створення ключів.

Весь алгоритм AES заснований на випадковій генерації різноманітних величин та їх перетворенні, тому було обрано саме цей спосіб.

2.3 Проектування системи

Одним із найліпших інструментів при проектуванні є UML-діаграми, завдяки яким розробники можуть віддати більше уваги саме проектуванню та архітектурі, не відволікаючись на генералізацію позначень.

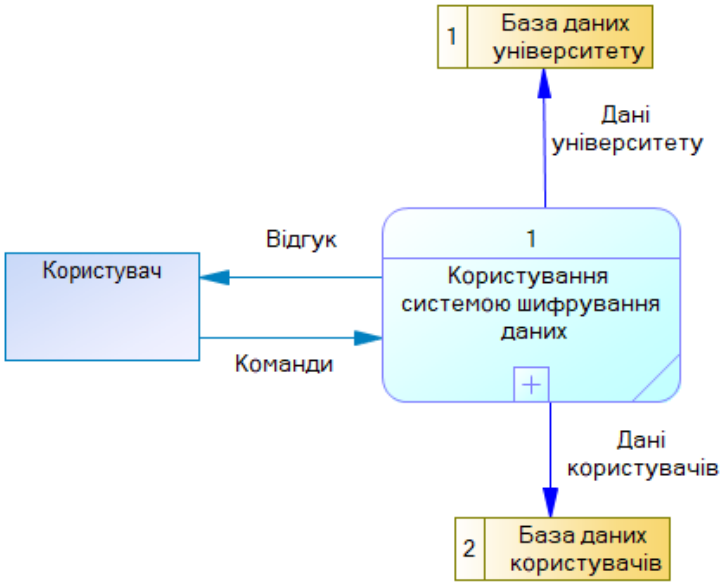


Рис. 2.1 Нульовий рівень DFD-діаграми

При проектуванні системи однією з головних задач є відобразити роботу програми та зв'язок із зовнішніми джерелами, адресатами, потоками та сховищами даних. Для цього використовується DFD-діаграма (діаграма потоків даних).

На концептуальному (нульовому рівні) діаграми (рис. 2.1) відображені основні зв’язки між користувачами, програмним забезпеченням та базами даних.

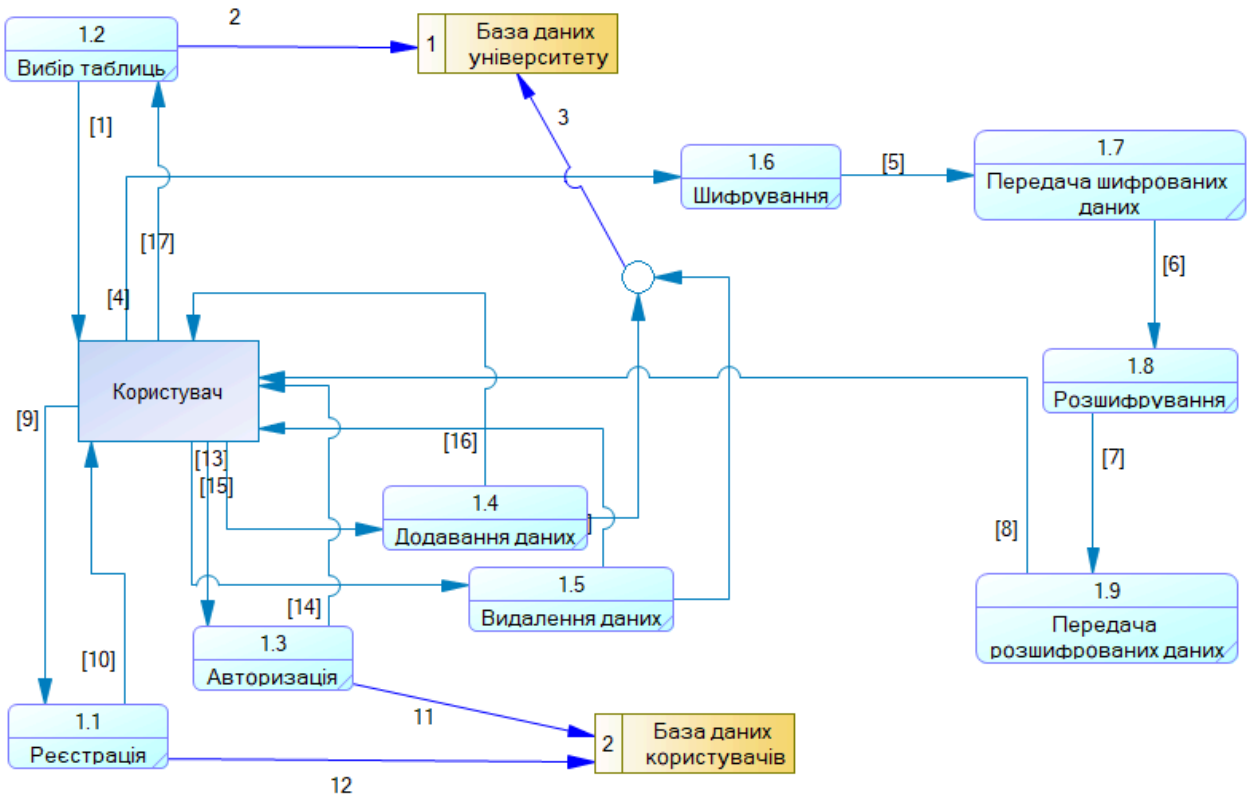


Рис. 2.2 Перший рівень DFD-діаграми

Умовні позначення (рис. 2.2):

1 – Одна з таблиць з даними університету;

2 – Дані університету;

3 – Нові дані університету;

4 – Дані для шифрування;

5 – Зашифровані дані;

6 – Дані для розшифрування;

7 – Розшифровані дані;

8 – Таблиця із розшифрованими даними;

9 – Дані для реєстрації;

10 – Відгук про стан реєстрації;

11 – Дані користувача;

12 – Дані користувача;

13 – Дані для авторизації;

14 – Відгук про стан авторизації;

15 – Нові дані;

16 – Оновлена таблиця;

17 – Вибір таблиці.

На першому рівні DFD-діаграми відображено, як користувач може користуватися програмою, а також як саме дані передаються з БД до таблиць та з таблиць до БД.

Діаграма послідовностей є важливим інструментом в об'єктно-орієнтованому аналізі та проектуванні, що використовується для моделювання взаємодій між об'єктами в системі. Вона фокусується на тимчасовому порядку повідомлень, що передаються між об'єктами для виконання певного функціоналу. Діаграма показує, як об'єкти взаємодіють один з одним у процесі виконання одного або кількох сценаріїв використання.

На створеній діаграмі послідовностей (рис. 2.3) відображено життєвий цикл програми, який включає в себе життєві цикли об’єктів класів, описаних на рис. 3.9.

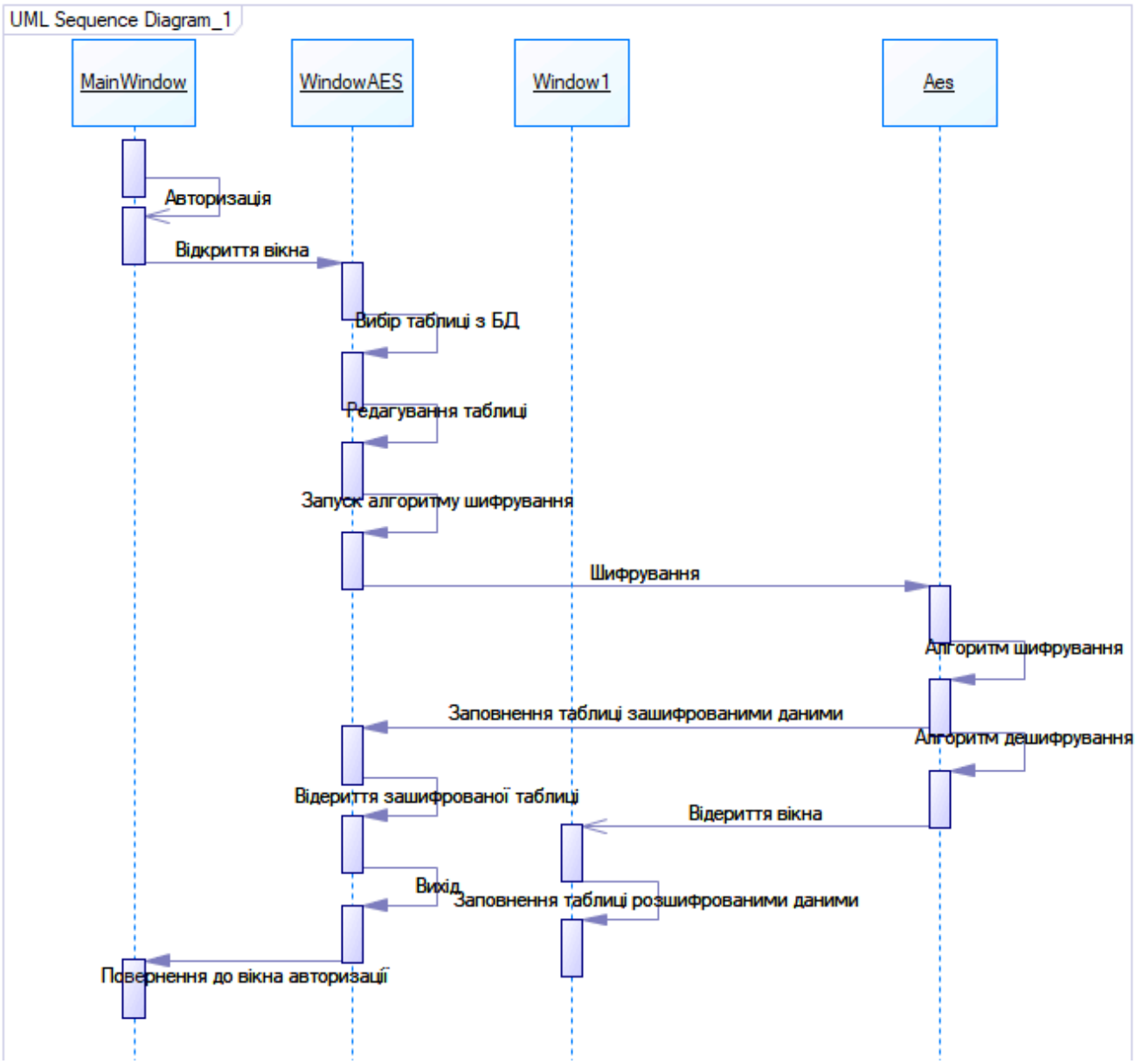


Рис. 2.3 Діаграма послідовностей

Від саме назв класів йдуть пунктирні лінії, які є «лініями життя» обраного об’єкту – течією часу при роботі з програмою. На них показуються прямокутники, які відображають діяльність об’єктів або виконання їх функцій. Сигнали та повідомлення зображені стрілками.

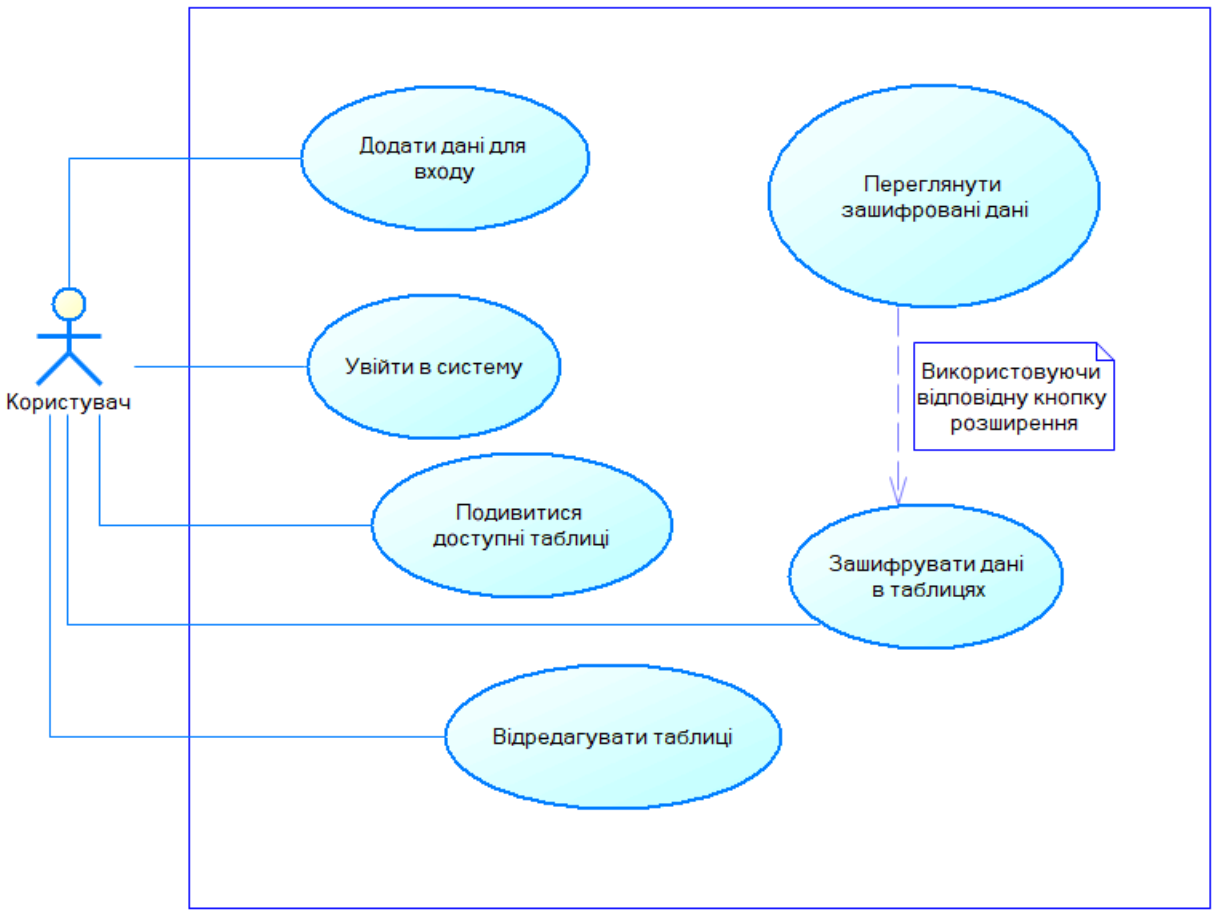


Рис. 2.4 Діаграма прецедентів

Діаграма прецедентів є важливим інструментом у процесі розробки програмного забезпечення. Вона допомагає візуалізувати взаємодію між користувачами (акторами) та системою, показуючи, які функції та дії їм доступні. Діаграма дозволяє чітко визначити вимоги до системи, що полегшує спілкування між розробниками, аналітиками та зацікавленими сторонами. Завдяки такій діаграмі можна краще зрозуміти, які функції та сценарії використання системи є найбільш важливими та затребуваними, а також виявити можливі недоліки або упущення в проектуванні системи.

На use-case діаграмі (рис. 2.4) зображена робота користувача з системою – його можливості використання програми. В лиці актора виступає користувач.

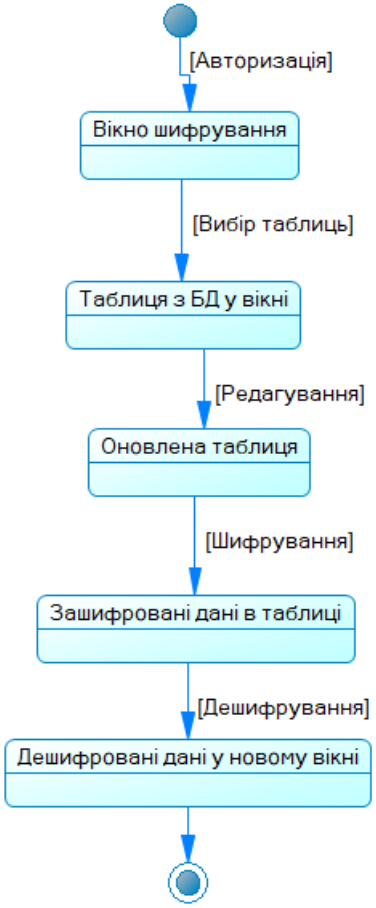


Рис. 2.5 Діаграма станів

Діаграма станів є важливим інструментом при проектуванні залежних від стану систем, що використовується для моделювання її динамічної поведінки. Діаграма відображає різні стани об'єкта та переходи між цими станами у відповідь на зовнішні події; допомагає зрозуміти та візуалізувати, як об'єкт змінює свій стан протягом свого життєвого циклу, надаючи чітку картину можливих переходів та умов, що їх викликають.

На рис. 2.5 зображена така діаграма станів, на якій є вся путь станів при використанні користувачем системи від його авторизації та відкриття основного вікна програми до шифрування/дешифрування даних та їх виведення на відповідні об’єкти інтерфейсу.

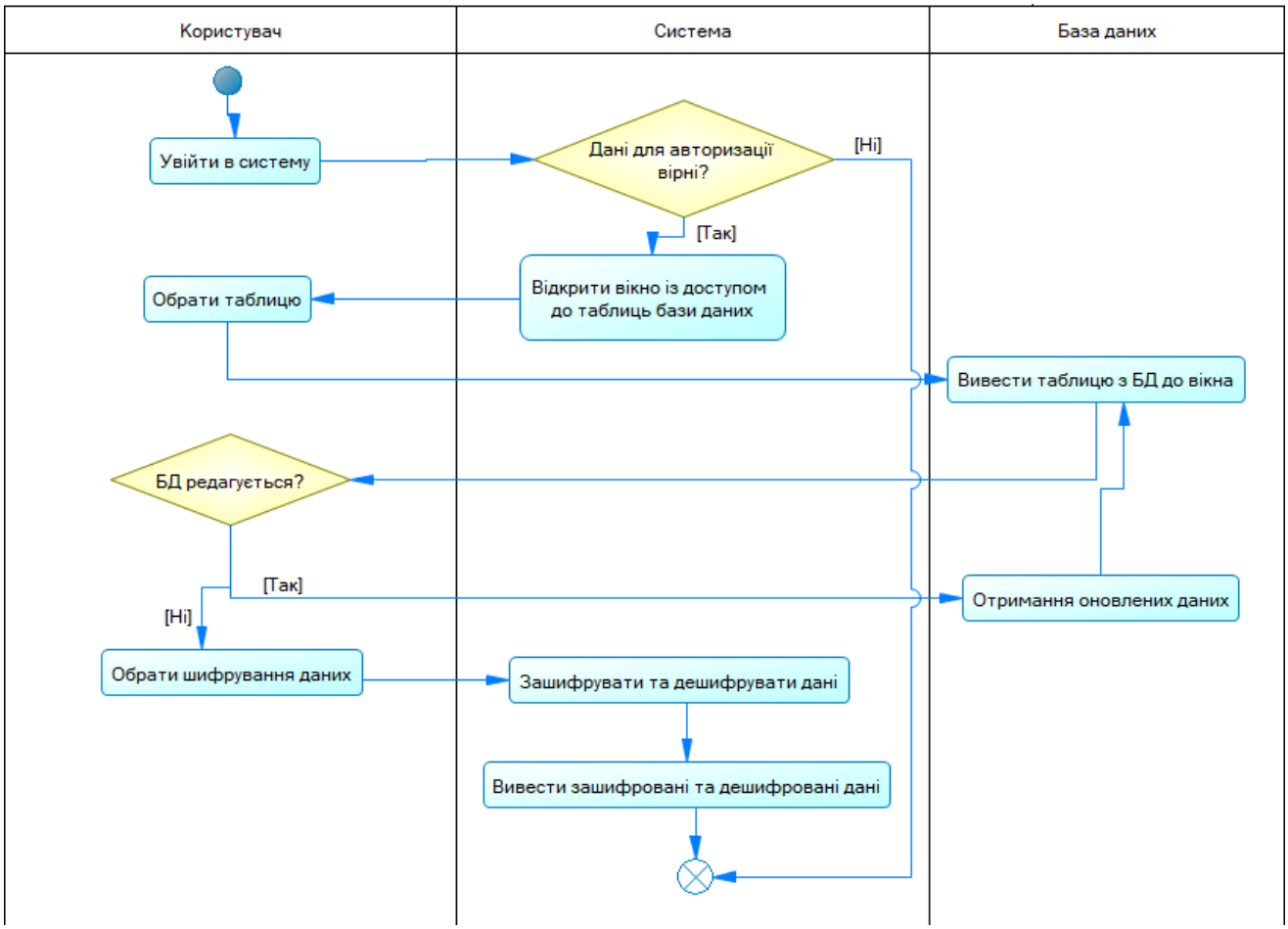


Рис. 2.6 Діаграма діяльності

Діаграма діяльності є одним із інструментів, що використовуються для моделювання робочих процесів у системі. Вона показує послідовність дій та потік управління між ними, допомагаючи візуалізувати процес виконання завдань та рішень. Діаграма корисна для опису як складних процесів лише на рівні системи, так і окремих операцій усередині класів чи об'єктів.

Діаграми діяльності також потрібні для аналізу та оптимізації процесів, проектування робочих потоків та моделювання алгоритмів. Вони дозволяють виявити вузькі місця та неефективності в роботі системи, полегшуючи їхню автоматизацію.

На створеній діаграмі діяльності (рис. 2.6) зображені дії станів, описаних на рис. 2.5. Взаємодія йде між трьома об’єктами – користувачем (виконує дії

через інтерфейс), самою системою (оброблює дії і виконує задані функції) та базою даних (зберігає потрібні дані).

2.4 Алгоритмічне забезпечення програми

Для того, щоб використати алгоритм AES потребуються дві вхідні величини – текст, який треба зашифрувати, та ключ шифрування (рис. 2.7). Після проведення необхідних операцій, а саме процес шифрування для тексту й алгоритм обробки для ключа шифрування (рис. 2.8), на виході користувач отримує зашифрований набір даних.

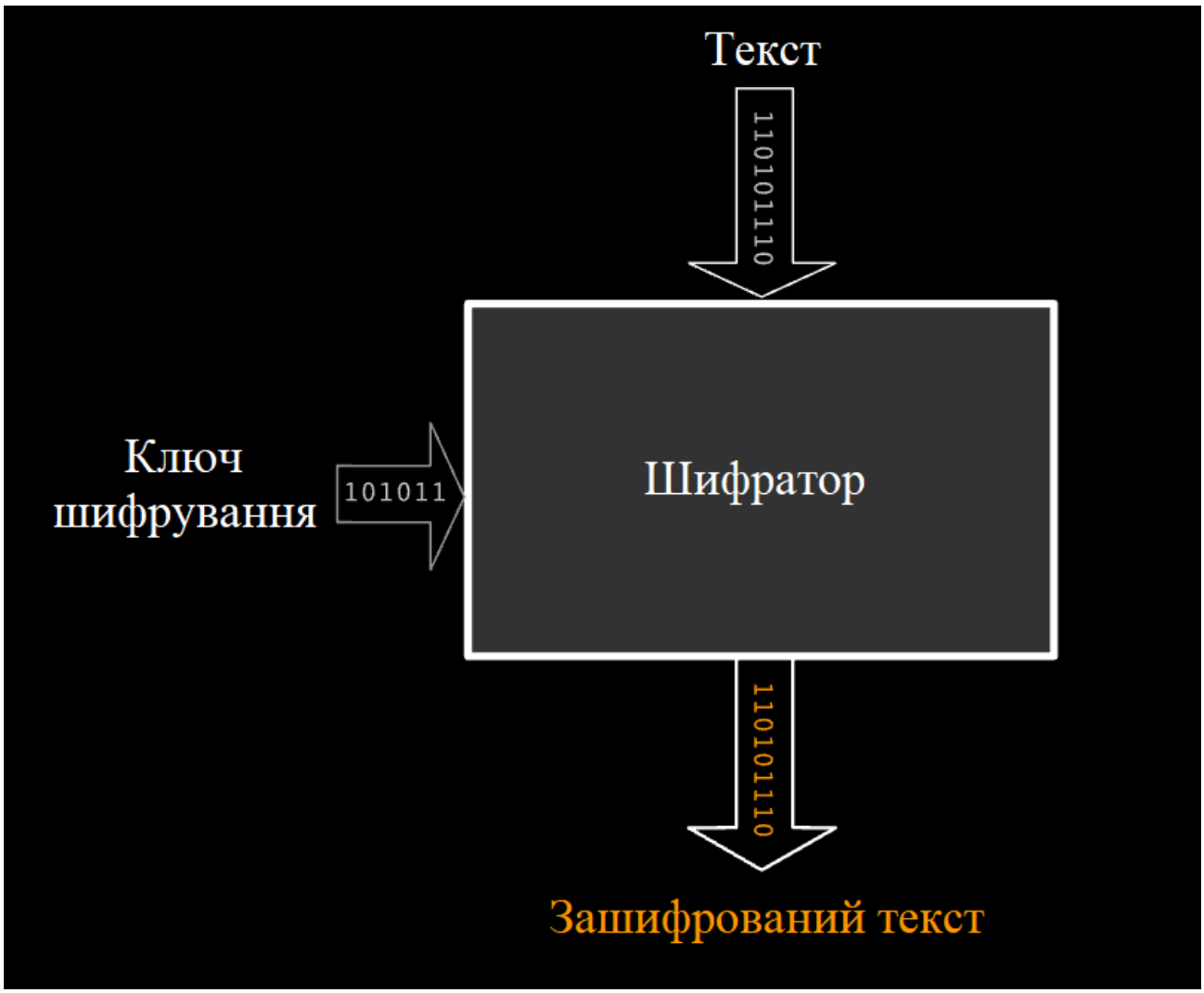


Рис. 2.7 Загальна робота алгоритму

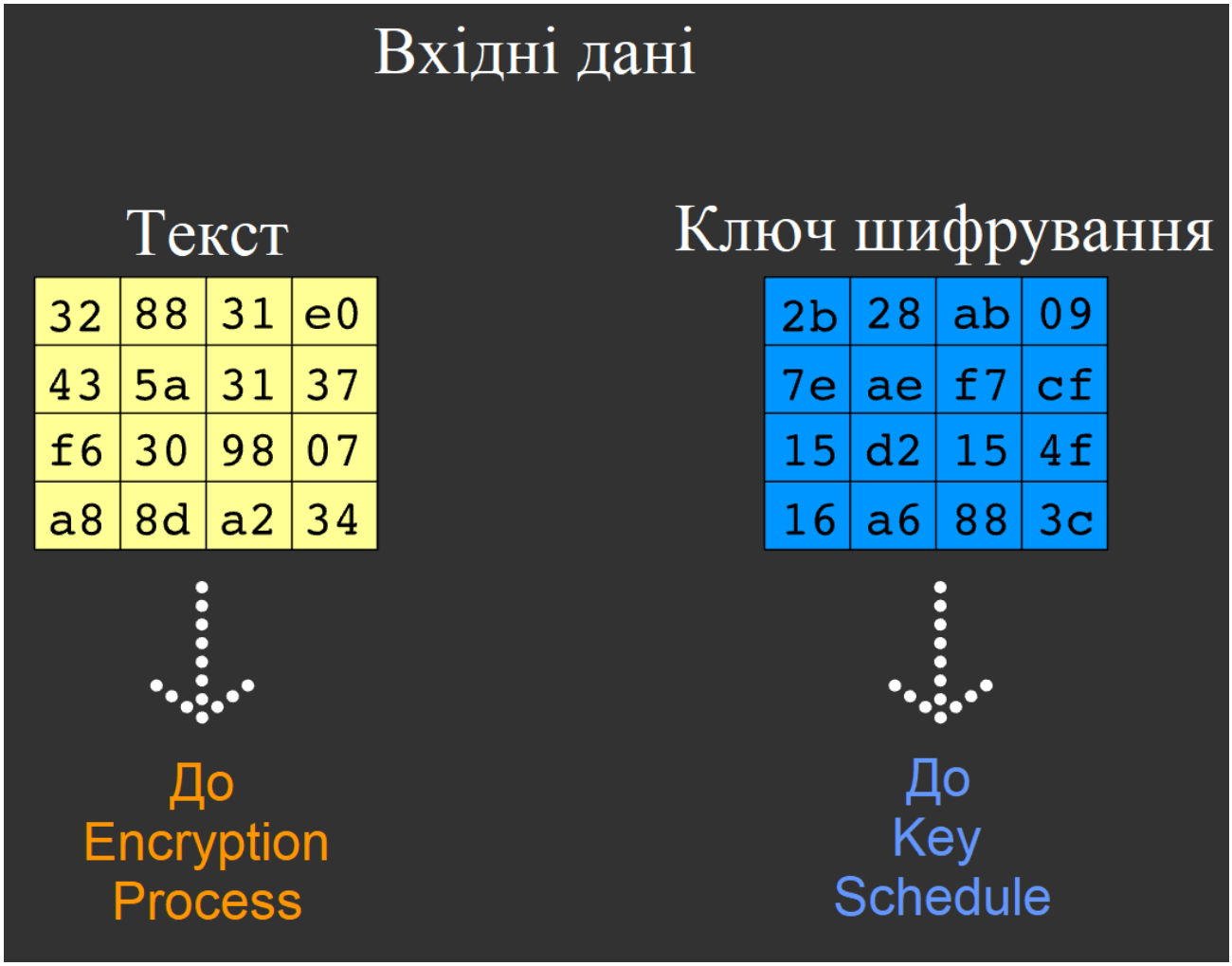


Рис. 2.8 Розподілення вхідних даних

На початку шифрування вхідні дані (наприклад, текст) копіюються в масив State за формулою:

$$state[r,c] = input[r+4c], \quad (2.2)$$

де $0 \le r \le 4$, $0 \le c \le Nb$.

Після цього до State за модулем два додається ключ шифрування на першій ітерації процедури AddRoundKey, а на всіх послідуючих – раундовий ключ. В кінці дані виводяться за формулою:

$$state[r,c] = output[r+4c] \quad (2.3)$$

Описаний алгоритм зображен на рис. 2.9.

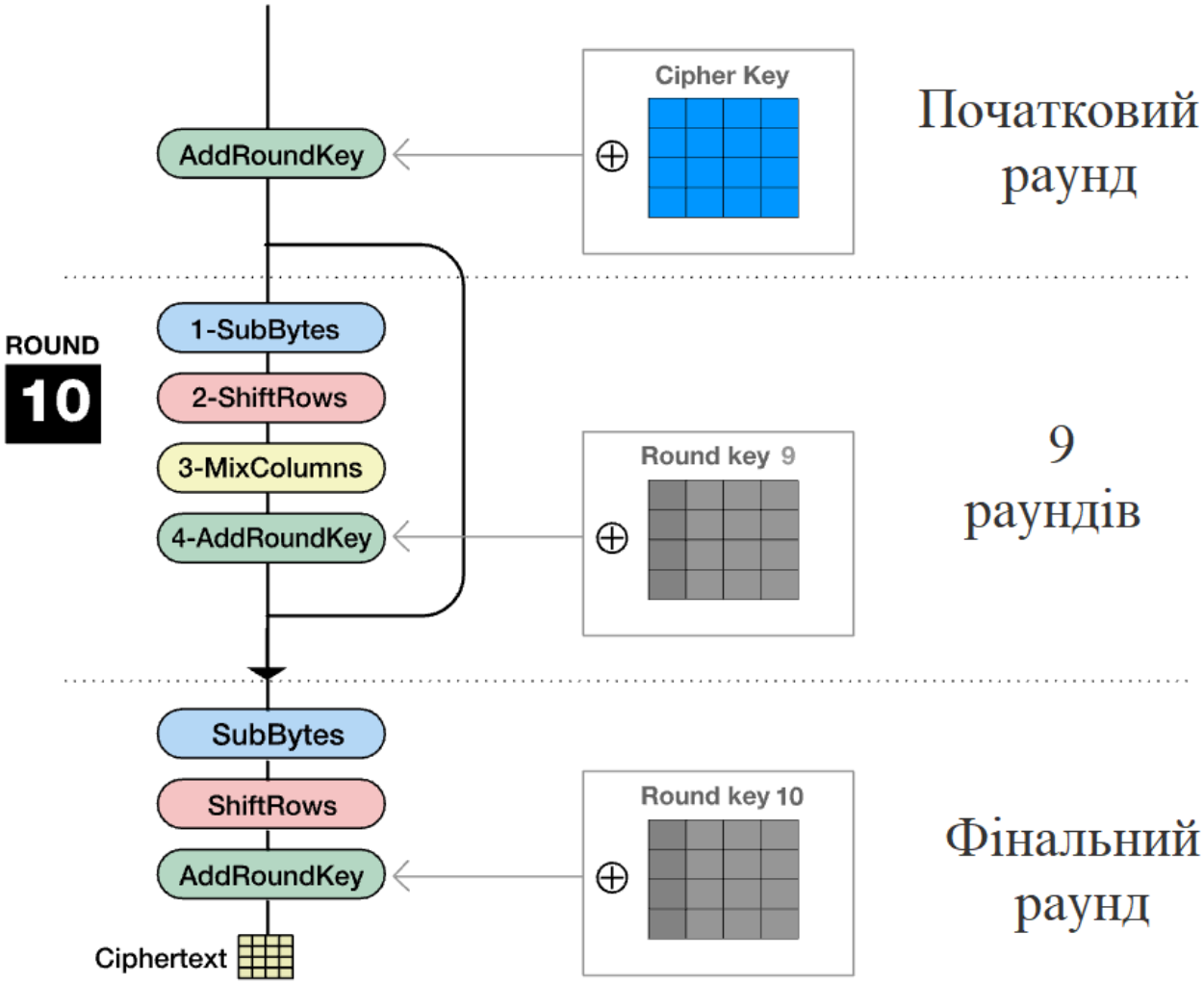


Рис. 2.9 Алгоритм шифрування

Розглянемо детальніше процедури, які виконуються при шифруванні, а саме SubBytes, ShiftRows, MixColumns, AddRoumdKey та алгоритм обробки ключа шифрування для створення раундового ключа.

Процедура SubBytes (рис. 2.10) обробляє кожний байт, з якого складається State, щоб незалежно провести нелінійну заміну байтів, використовуючи таблицю замін (S-box(рис. 1.5)). Така операція забезпечує непередбаченість алгоритму шифрування.

Формула заміни:

$$b_{i,j} = S(a_{i,j}), \tag{2.4}$$

де $b_{i,j}$ – нова матриця, $a_{i,j}$ – стара матриця, $S(a_{i,j})$ – відповідне попередньому значенню [i,j] значення в матриці замін.

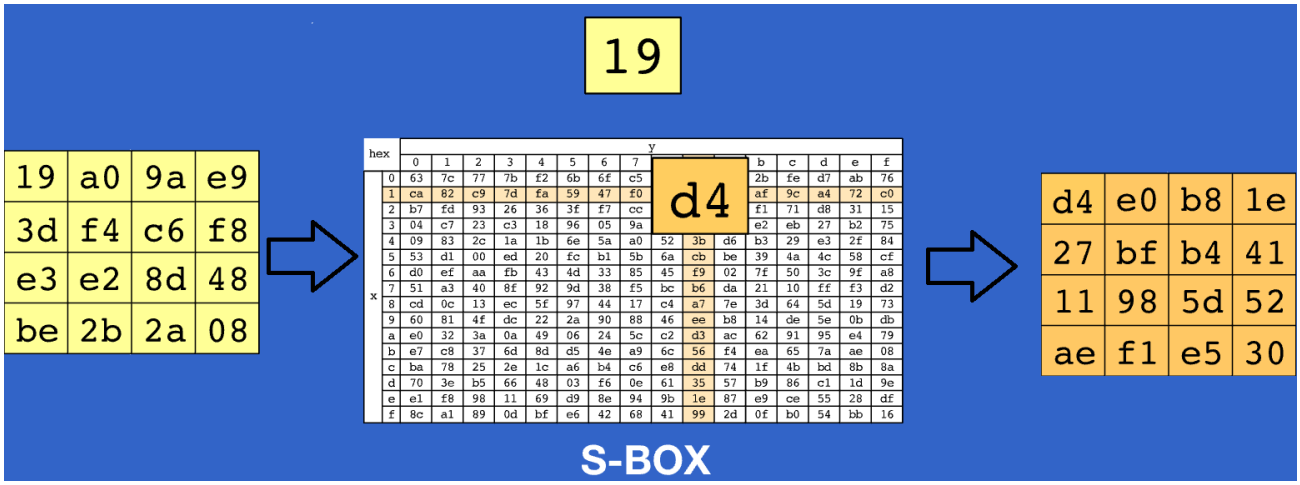


Рис. 2.10 Робота процедури SubBytes

Процедура ShiftRows (рис. 2.11) працює з строками отриманого після попередньої процедури State. При виконанні цієї трансформації кожна із строк циклічно зміщуються на кількість байтів по горизонталі, яка залежить від номера даної строки (рахунок починається з 0). Таким чином, кожна нова колонка має в собі по одному значенню з колонок попередньої матриці.

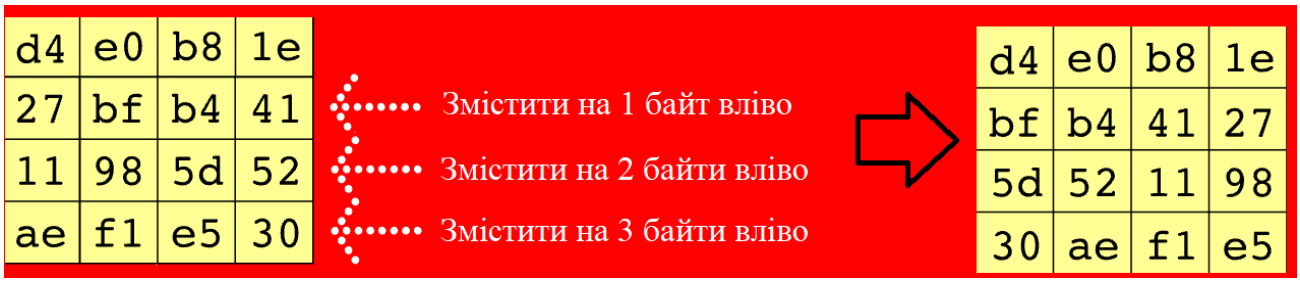


Рис. 2.11 Робота процедури ShiftRows

Процедура MixColumns (рис. 2.12), як слідує з назви, змішує чотири байти кожної колонки State, використовуючи для цього обратиму лінійну трансформацію. Кожен із стовбців трактується, як поліном третього ступеню. Цей поліном множиться за модулем $x^4 + 1$ на фіксований многочлен c(x) =

$3x^3 + x^2 + x + 2$ (перший стовбець матриці в квадратних скобках на рис. 2.12). Ця процедура додає нелінійності та заплутаності разом із ShiftRows.

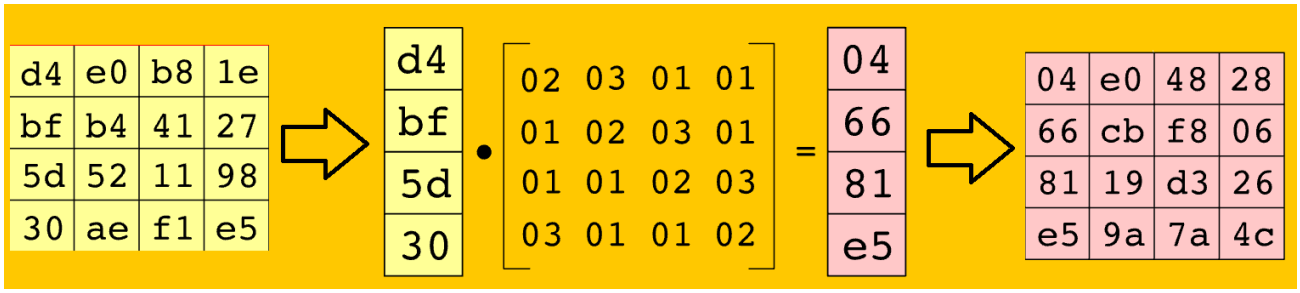


Рис. 2.12 Робота процедури MixColumns

Процедура AddRoumdKey об'єднуються RoumdKey та State кожної ітерації, як вже було зазначено вище.

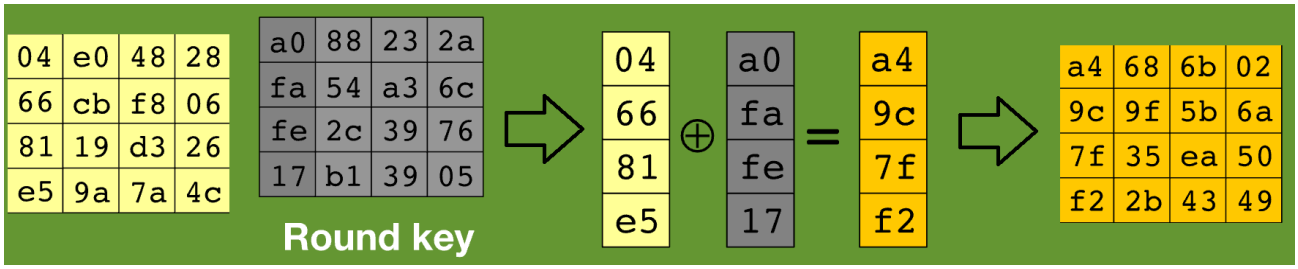


Рис. 2.13 Робота процедури AddRoumdKey

Текст проходить описані операції 10 раз, але не використовує MixColumn в останній ітерації.

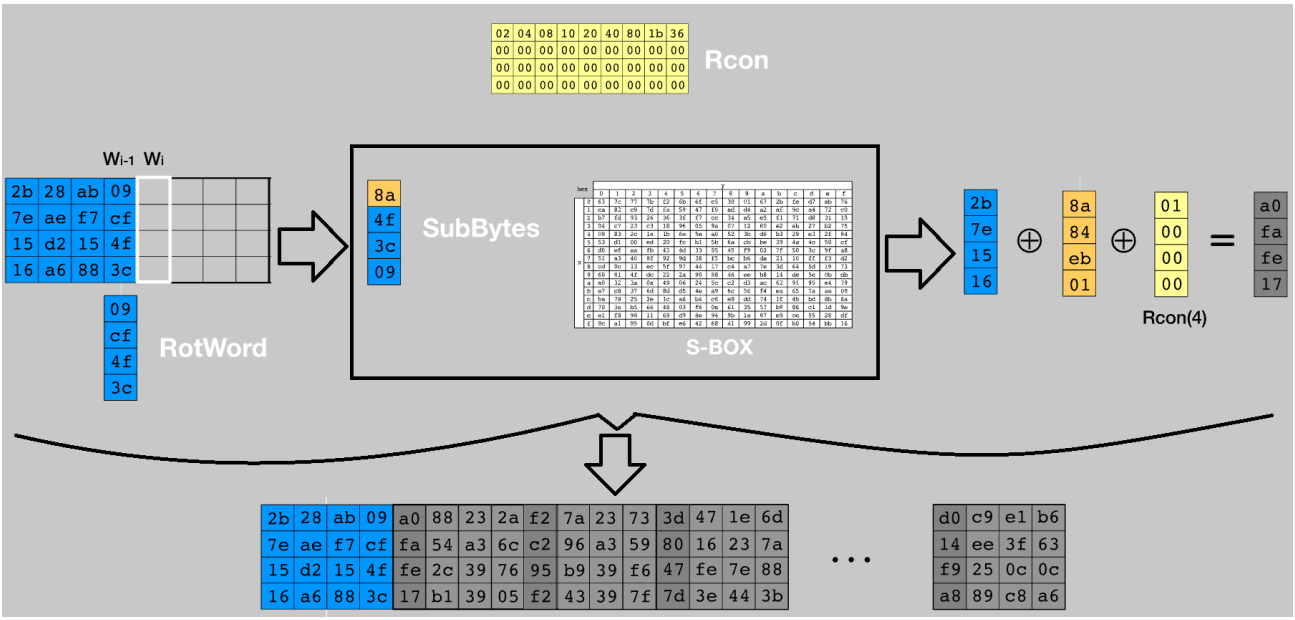


Рис. 2.14 Алгоритм обробки ключа

При використанні алгоритму обробки ключа (рис. 2.14) спочатку проводиться процедура RotWord для потрібного стовбця. Під час цієї процедури всі байти зміщуються на один вверх. Отриманий набір байтів проходить через вже описану раніше процедуру SubBytes, використовуючи той самий S-box. Наступним кроком алгоритму стає сума за модулем 2 іншої колонки, отриманого після матриці змін значення байтів та байтів з Rcon[i], де i – номер колонки. Отриманий стовбець стає першим в новому раундовому ключі.

Алгоритм повторюється для кожного нового раундового ключа, але замість ключа шифрування вже використовується RoundKey попередньої ітерації.

3. ПРОЕКТНІ РІШЕННЯ. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Опис проектних рішень програмного забезпечення

WPF – аналог WinForms, система для побудови клієнтських програм Windows з візуально привабливими можливостями взаємодії з користувачем, графічна (презентаційна) підсистема у складі .NET Framework (починаючи з версії 3.0), що використовує мову XAML. [14]

В основі WPF лежить векторна система візуалізації, незалежна від дозволу пристрою на виведення та оптимізована для сучасного графічного обладнання. WPF надає засоби для створення візуального інтерфейсу, включаючи мову розмітки XAML (eXtensible Application Markup Language), елементи керування, прив'язку даних, макети, двовимірну та тривимірну графіку, анімацію, стилі, шаблони, документи, текст, мультимедіа та оформлення.

XAML – це мова, призначена для декларативного опису інтерфейсу на основі XML. Вона також забезпечує модель поділу коду та дизайну, яка сприяє співпраці між програмістом та дизайнером. Крім цього, XAML включає вбудовану підтримку стилів елементів, що дозволяє легко розділяти елементи на рівні керування. Ці рівні можна розбити на векторні фігури та властивості/дії. Це полегшує завдання стилю будь-якого елемента, наприклад, кнопок або написів. [15]

Графічною технологією, яка використовується у WPF, є DirectX, на відміну від більш старого Windows Forms, де застосовується GDI/GDI+. Завдяки використанню апаратного прискорення графіки через DirectX продуктивність WPF вища, ніж у GDI+.

Для роботи з WPF можна використовувати будь-яку .NET-сумісну мову програмування. Серед таких мов є C#, F#, VB.NET, C++, Ruby, Python, Delphi (Prism), Lua тощо. Для повноцінної розробки доступні як Visual Studio, так і Expression Blend. Орієнтацією першої є програмування, а другої – дизайн, що

дозволяє виконувати багато завдань без необхідності ручного редагування XAML. Для створення програми в цій роботі було використано саме перший варіант.

WPF дозволяє використовувати різні можливості створення інтерактивних настільних додатків:

- **Прив'язка даних** – цей механізм дозволяє гнучко зв'язувати різні дані в розмітці XAML, починаючи від значень властивостей елементів керування і закінчуючи загальнодоступними властивостями, що відображають поля бази даних через Entity Framework. Прив'язка даних здійснюється за допомогою класу Binding, який успадковується від MarkupExtension, що дозволяє застосовувати прив'язки як коду, так і розмітки.
- **Стилі** дозволяють створювати оформлення елементів та зазвичай використовуються виключно в розмітці. Вони дозволяють змінювати зовнішній вигляд елементів.
- **Шаблони елементів управління**, які представлені класом ControlTemplate, дозволяють змінювати як зовнішній вигляд, так і структуру елемента. Це відрізняється від стилів, дозволяючи вносити зміни у графічне уявлення елемента, а й у його структуру. При цьому шаблон елемента управління визначається властивістю Template.
- **Шаблони даних**, на відміну шаблонів елементів управління, використовуються для конкретного контексту даних. У блокових елементах управління контекст даних встановлюється через властивість DataContext, а спискових – через ItemsSource. Сам шаблон даних представлений класом DataTemplate. Для вказівки типу даних, до якого застосовується шаблон, використовується властивість DataType.

DataGrid – представляє елемент керування, що відображає дані в сітці, яка налаштовується. Використовується для виводу даних із створеної БД.

Елемент управління DataGrid надає гнучкий спосіб відображення колекції даних у вигляді стовпців (з основними ознаками) та рядків (елементів, які відносяться до ознак). Він включає вбудовані типи стовпців DataGrid і можливість використання стовпця-шаблону для вставки користувачем даних. Вбудований тип рядка включає розділ, який може використовуватися для відображення додаткового контенту під значеннями комірок. [16]

Microsoft SQL Server Management Studio – це інтегроване середовище розробки, яке використовується для управління будь-якими компонентами SQL Server, а також для створення, зміни та управління базами даних SQL Server. [17]

Основними характеристиками та можливостями студії є:

- Управління базами даних, що дозволяє адміністраторам баз даних і розробникам керувати всіма аспектами баз даних SQL Server, включаючи створення, видалення, зміну баз даних, таблиць, уявлень, процедур, функцій і багато іншого.
- За допомогою SSMS можна писати, редагувати та виконувати T-SQL запити безпосередньо у середовищі розробки. Це зручно для аналізу даних, налагодження запитів та створення нових запитів.
- Середовище надає можливість управління правами доступу до баз даних та об'єктів, а також управління користувачами та ролями безпеки.
- За допомогою студії можна моніторити роботу SQL Server, а також налаштовувати параметри сервера для оптимізації продуктивності та забезпечення надійності.

- Management Studio дозволяє імпортувати та експортувати дані між різними джерелами даних, включаючи інші бази даних, текстові файли тощо.
- SSMS забезпечує можливість розробки звітів та аналітичних запитів за допомогою інструментів, таких як SQL Server Reporting Services (SSRS) та SQL Server Analysis Services (SSAS).
- Середовище інтегрується з іншими інструментами та платформами розробки, такими як Visual Studio та Azure Data Studio, що полегшує роботу з даними та додатками.

SQL Management Studio є одним із основних інструментів для роботи з SQL Server і надає широкий набір функціональних можливостей для розробників, адміністраторів баз даних та аналітиків даних. [17]

3.2 Розробка компонентів програмного забезпечення

3.2.1 Підготовка баз даних

Для перевірки працездатності програми шифрування даних за допомогою Microsoft SQL Server Management Studio було створено тестовий приклад бази даних університету. Також БД використовується для збереження вхідних даних користувача, а саме його логін та пароль.

Перед створенням основної бази даних треба зробити її концептуальну, логічну та фізичну моделі, в яких показати зв’язки між елементами, їх основні ознаки та залежності. Загалом ці моделі можна об’єднати в одне поняття – ER-модель.

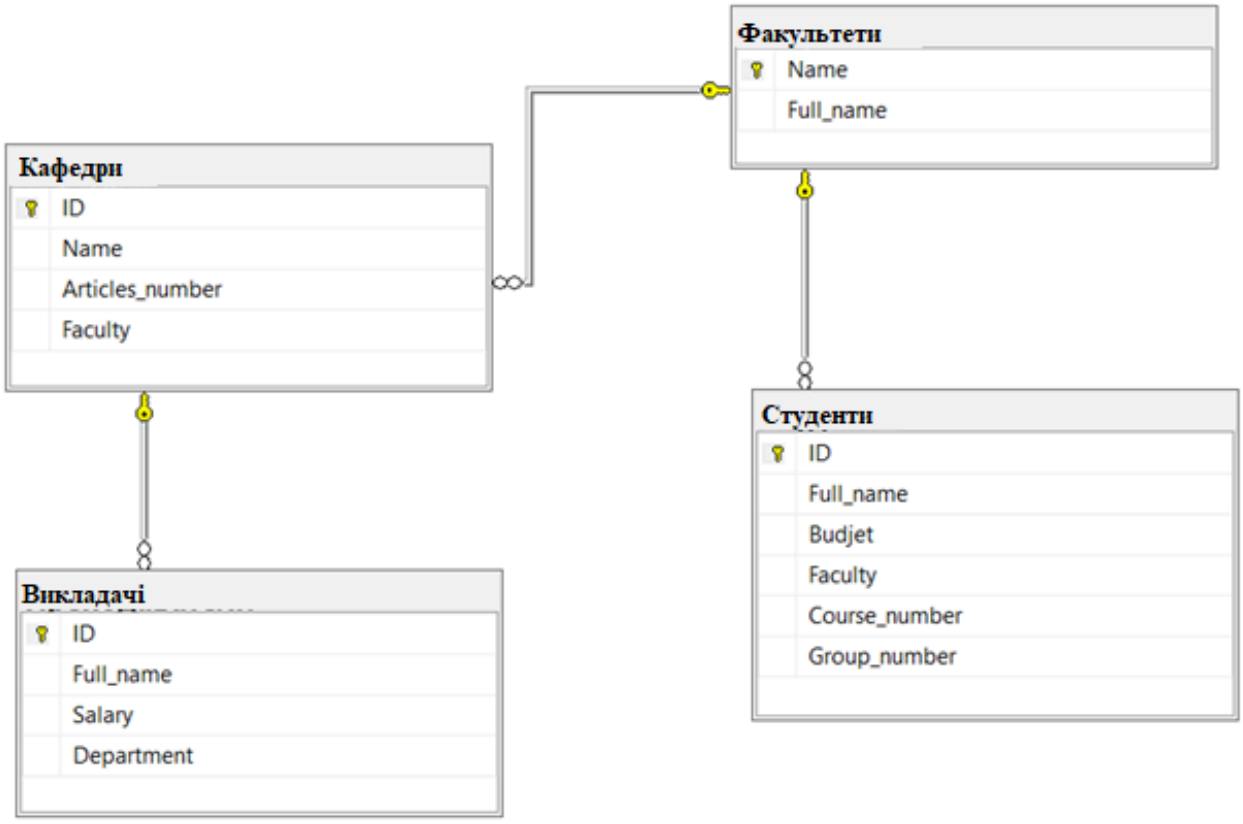


Рис. 3.1 Концептуальна модель бази даних університету

На концептуальній моделі (рис. 3.1) відображена абстракція, яка визначає структуру системи та зв’язки між її складовими – Викладачами, Кафедрами, Факультетами та Студентами. Кожній сутності наданий свій головний ключ та ознаки.

На логічній моделі (рис. 3.2) відображається більше деталей для ознак: є заданими типи даних і зовнішні ключі, та зв’язків: з’явились їх назви та тип, один до багатьох. Також усі сутності, від яких виходить зв’язок «багато», є залежними від тих, в кого лише «один» (наприклад, декілька викладачів працює на кафедрі, але один викладач не може бути на декількох кафедрах разом; викладачів без зв’язком із кафедрою немає).

Логічна модель не залежить від технології, в якій вона буде застосовуватися, тому неповний збіг між типами даних в SQL Server Management Studio та на рис. 3.2 дозволяється.

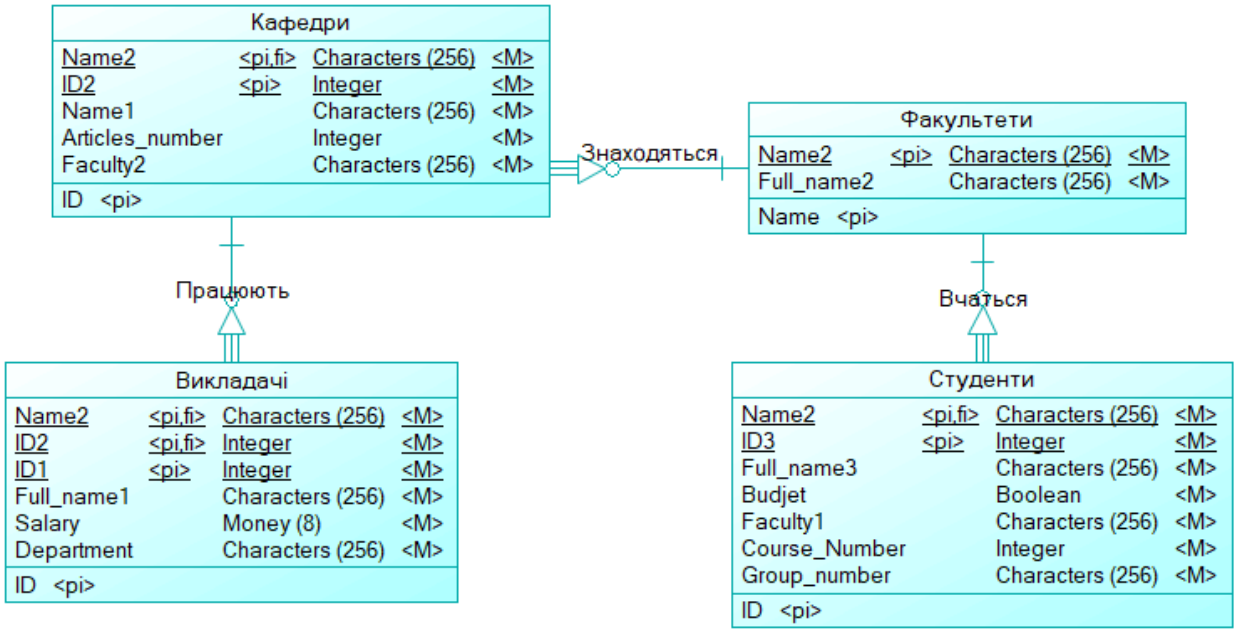


Рис. 3.2 Логічна модель бази даних університету

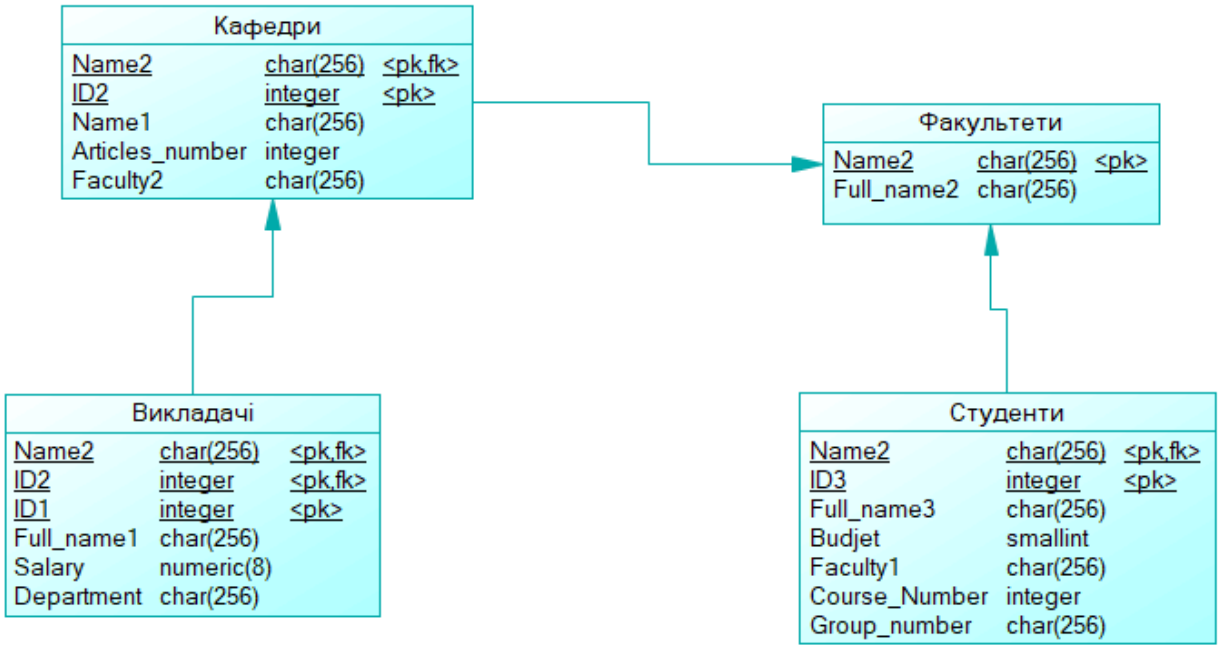


Рис. 3.3 Фізична модель бази даних університету

Фізична модель (рис 3.3) відображає все, що потрібно знати про базу даних для її створення.

База даних, яка розробляється, не є занадто наповненою, тому моделі з рис. 3.2 та 3.3 практично повністю збігаються, залишаючи невеликі відмінності, які не є дуже важливими для їх опису.

Друга база даних, в якій будуть зберігатися логіни та паролі для входу до програми, є невеликою, тому для неї існує лише логічна (рис. 3.4) та фізична (рис. 3.5) моделі.

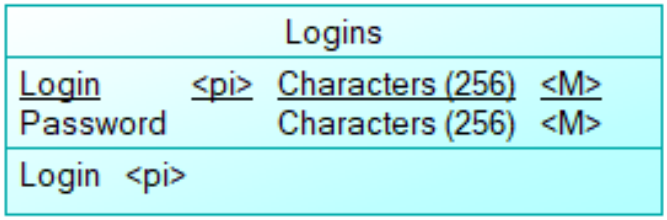


Рис. 3.4 Логічна модель бази даних входу

Рис. 3.5 Фізична модель бази даних входу

Закінчивши підготовку, можна почати створювати БД. Обидві бази даних будуть знаходитись на одному сервері \SQLEXPRESS. За допомогою утиліт SQL-менеджеру додаємо всі необхідні таблиці, ознаки, ключі та зв’язки.

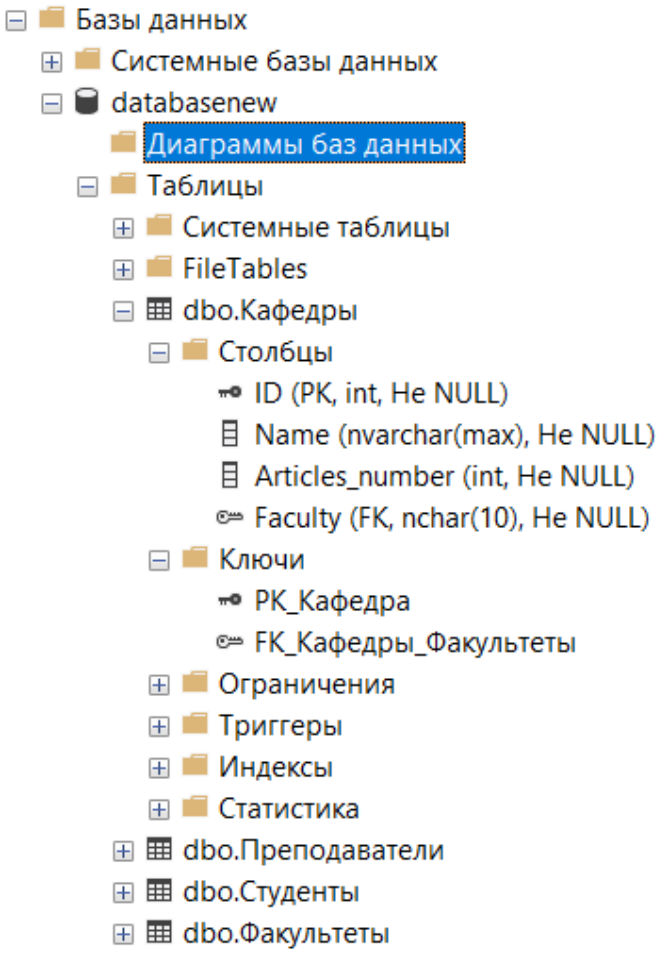


Рис. 3.6 Загальний вигляд оформленої бази даних

	Имя столбца	Тип данных	Разрешить знач...
🔑	ID	int	☐
	Full_name	nvarchar(MAX)	☐
	Salary	money	☐
	Department	int	☐

Рис. 3.7 Вигляд заданих стовбців таблиці даних викладачів

Перед тим, як завершити підготовку головної бази даних, треба перевірити, чи проходить вона перевірку на «нормальність». Для цього БД має відповідати умовам хоча би трьох нормальних форм.

Нормальна форма – це спосіб організації даних у базі даних таким чином, щоб уникнути зайвої інформації, яка може призвести до помилок під час роботи з ними. В основному, це набір правил, які говорять, яким чином дані мають бути організовані, щоб усе працювало як слід.

Перша нормальна форма відповідає за те, щоб у таблицях БД не було декількох значень в одній клітинці. На рис. 3.8 видно, що цю форму база проходить.

ID	Full_name	Salary	Department
101	Добрін Сте...	23000,0000	43
102	Хіщина Вал...	25000,0000	47
103	Морозко О...	25000,0000	52
104	Дулін Вале...	28000,0000	43
105	Ячин Мико...	24000,0000	50
106	Скворцов ...	24000,0000	50
107	Лагода Кир...	26000,0000	45
108	Фолькіна Ір...	25000,0000	45
109	Солод Мар...	20000,0000	48
110	Чап Ганна ...	23000,0000	43

Рис. 3.8 Заповнена таблиця викладачів

Друга нормальна форма відповідає за те, щоб у таблицях був ключ, а також, що всі неключові ознаки є залежними лише від нього, що теж можна побачити на рис. 3.8.

Третя нормальна форма відповідає за те, щоб між ознаками не було транзитивних залежностей, тобто, вони не можуть залежати один від одного через інші ознаки (ключі тощо) (рис. 3.8).

3.2.2 Опис класів

При розробці програми було створено два робочих вікна, в яких користувач може проводити потрібні операції. Також класи для цих вікон та їх методи стали однією з головних частин розробки.

Третє вікно, як і третій клас, є лише зручним відображенням роботи алгоритму шифрування та передачі даних між двома вікнами, але також має бути відображено в роботі програми та на діаграмі класів.

Останній клас – це вбудований в мову програмування C# клас Aes. Він представляє абстрактний базовий клас, від якого наслідуються всі подальші реалізації.

Таблиця 3.1 Методи класу MainWindow

Тип	Назва	Параметри	Опис
void	InitializeDatabase	Немає	Створює та відкриває зв’язок між класом та створеним сервером баз даних для подальшого використання однієї з них.
bool	ValidateUser	string login, string password	Створює SQL-запит для перевірки наявності логіна та пароля в базі даних. Повертає 1, якщо такий набір даних існує, 0 – якщо ні.

Продовженя таблиці 3.1 Методи класу MainWindow

Тип	Назва	Параметри	Опис
void	LoginButton_Click	object sender, RoutedEventArgs e	Якщо ValidateUser повертає 1, відкриває вікно WindowAES та закриває вікно входу, або виводить попередження.
Void	AddUserButton_Click	object sender, RoutedEventArgs e	Перевіряє, чи існує вже дані для логіна в БД. Створює SQL-запит для додання нового користувача та додає нові логін з паролем. При наявності логіна виводить попередження.

Таблиця 3.2 Методи класу WindowAES

Тип	Назва	Параметри	Опис
void	InitializeDatabase	Немає	Створює та відкриває зв’язок між класом та створеним сервером баз даних для подальшого

			використання однієї з них.

Продовження таблиці 3.2 Методи класу WindowAES

Тип	Назва	Параметри	Опис
void	LoadData	Немає	Завантажує дані з бази даних у datatable, щоб потім додати їх до потрібного datagrid.
void	FillComboBox	Немає	Робить запит на отримання та заповнює обраний combobox назвами таблиць з бази даних.
void	EncryptData	string key	Створює копію першої datagrid, змінює типи даних всіх стовбців на string та поклітинково шифрує дані за допомогою AES, пвсля чтого відображає зашифровані дані і іншому datagrid.

Продовження таблиці 3.2 Методи класу WindowAES

Тип	Назва	Параметри	Опис
string	GenerateRandomKey	Немає	Генерує випадковий ключ довжиною шістнадцять символів. Повертає цей ключ.
byte[]	EncryptStringToBytes_Aes	string plainText, byte[] Key, byte[] IV	Зашифровує текст у байти. Повертає масив отриманих байтів.
void	DecryptData	string key	Отримує зашифровані дані з відповідного datagrid та проходить по кожній стрічці й кожному стовбцю, щоб розшифрувати дані. Передає ці дані в datagrid нового вікна та відкриває його.
string	DecryptStringFromBytes_Aes	byte[] cipherText, byte[] Key, byte[] IV	Розшифровує текст з отриманого масиву байтів. Повертає отриманий текст.

Продовження таблиці 3.2 Методи класу WindowAES

Тип	Назва	Параметри	Опис
void	Window_Closing	object sender, System.ComponentModel.CancelEventArgs e	Завершує підключення до бази даних після закриття вікна.
void	comboBox_SelectionChanged_1	object sender, SelectionChangedEventArgs e	Викликає метод LoadData при зміні бажаної таблиці в combobox.
void	EncryptButton_Click	object sender, RoutedEventArgs e	Генерує випадковий ключ шифрування за допомогою GenerateRandomKey. Шифрує та дешифрує дані за допомогою відповідних методів. При помилці виводить попередження.
void	ShowButton_Click	object sender, RoutedEventArgs e	Виводить у вікні таблицю із зашифрованими даними.
void	ExitButton_Click	object sender, RoutedEventArgs e	Закриває вікно шифрування та відкриває вікно входу.

Таблиця 3.3 Поля класу Aes [18]

Тип	Назва	Опис
int	BlockSize	Розмір блока шифрування, який використовується в криптографічній операції.
byte[]	Key	Ключ, який використовується для шифрування та дешифрування заданого набора даних.
byte[]	IV	Випадково згенерований ініціалізуючий вектор.
int	KeySize	Розмір ключа для шифрування та дешифрування.

Ініціалізуючий вектор (IV, Initialization Vector) – невеликий випадково згенерований блок даних, який використовується у криптографії, особливо у режимах шифрування блоків.

Таблиця 3.4 Методи класу Aes [18]

Тип	Назва	Параметри	Опис
void	Clear	Немає	Звільняє всі ресурси, які використовуються класом алгоритму.
void	Create		Створює криптографічний об’єкт, який використовується для виконання алгоритму.

Продовження таблиці 3.4 Методи класу Aes

Тип	Назва	Параметри	Опис
void	CreateDecryptor	Немає	Створює симетричний об’єкт дешифратора з ключем та ініціалізуючим вектором (табл. 3.3).
void	CreateEncryptor		Створює симетричний об’єкт шифратора з ключем та ініціалізуючим вектором (табл. 3.3).
string	ToString		Повертає значення string, яке відповідає обраному об’єкту.

Разом із класом Aes в програмі використовується ще декілька класів: MemoryStream, CryptoStream та StreamReader.

Єдина дія, яка проводиться з останнім класом Window1 – це ініціалізація відповідного йому вікна – InitializeComponent().

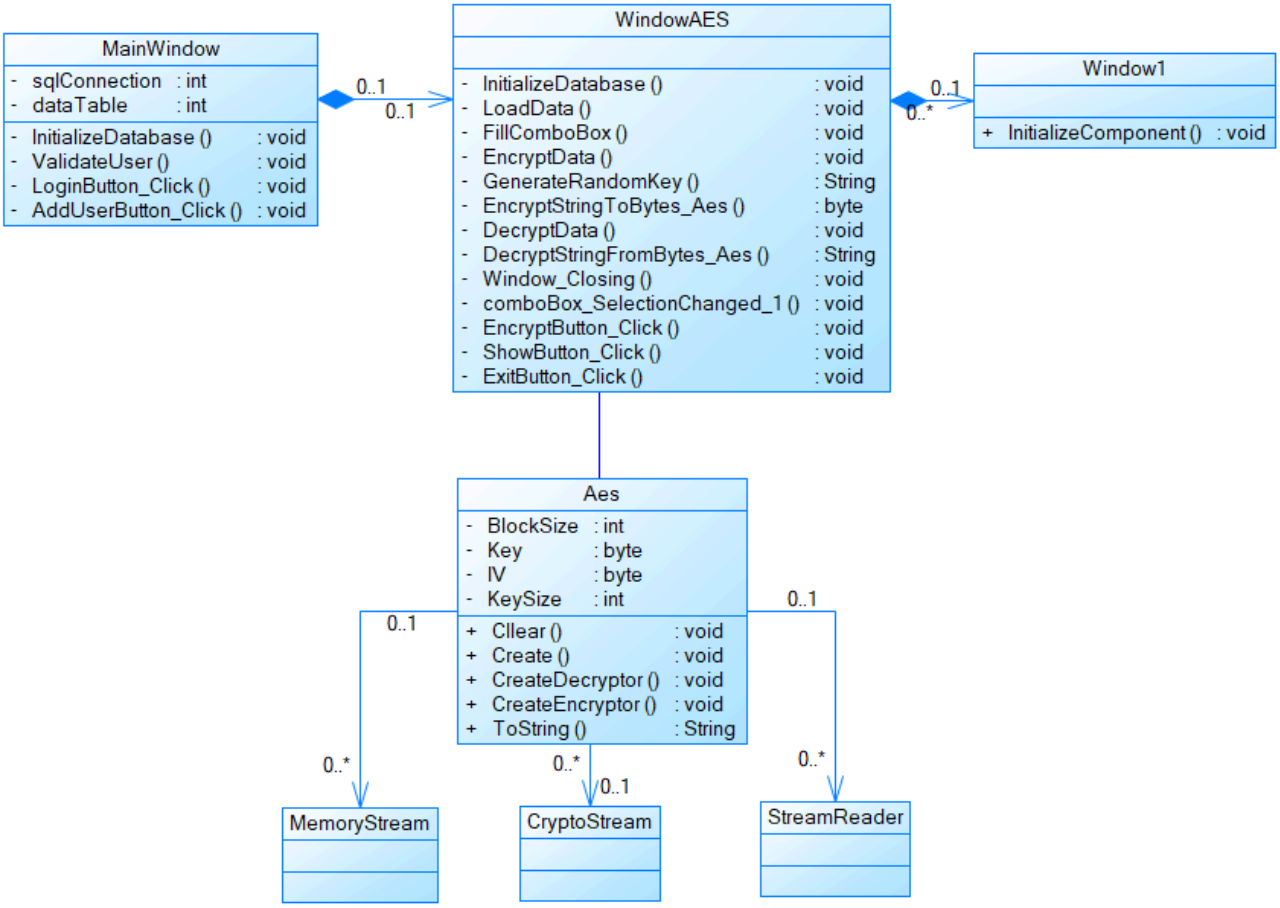


Рис. 3.9 Діаграма класів

На рис. 3.9 зазначені всі класи, описані вище, та зв’язки між ними. Таким чином, WindowAes є залежним від існування MainWindow, а Window1 – від WindowAes. Між WindowAes та Aes простий зв’язок використання. Також показано додаткові класи, пов’язані з Aes.

3.3 Опис тестового прикладу

Після запуску програми користувач попадає на вікно входу (рис. 3.9), де йому потрібно ввести логін та пароль для доступу до бази даних. При невірному вводі, на екран виводиться попередження «Your login or password are incorrect.».

Рис. 3.10 Вікно входу

Для того, щоб користувач міг зайти в програму, йому потрібно спочатку зареєструватися. Для цього достатньо ввести бажані логін та пароль та натиснути відповідну кнопку. Якщо користувач введе вже існуючий логін, виведеться попередження «You can not use this login.». На рис. 3.11, 3.12 та 3.13 показано, як працює додавання користувачі до бази даних.

	Login	Password
1	Login	0000

Рис. 3.11 Дані користувачів для входу

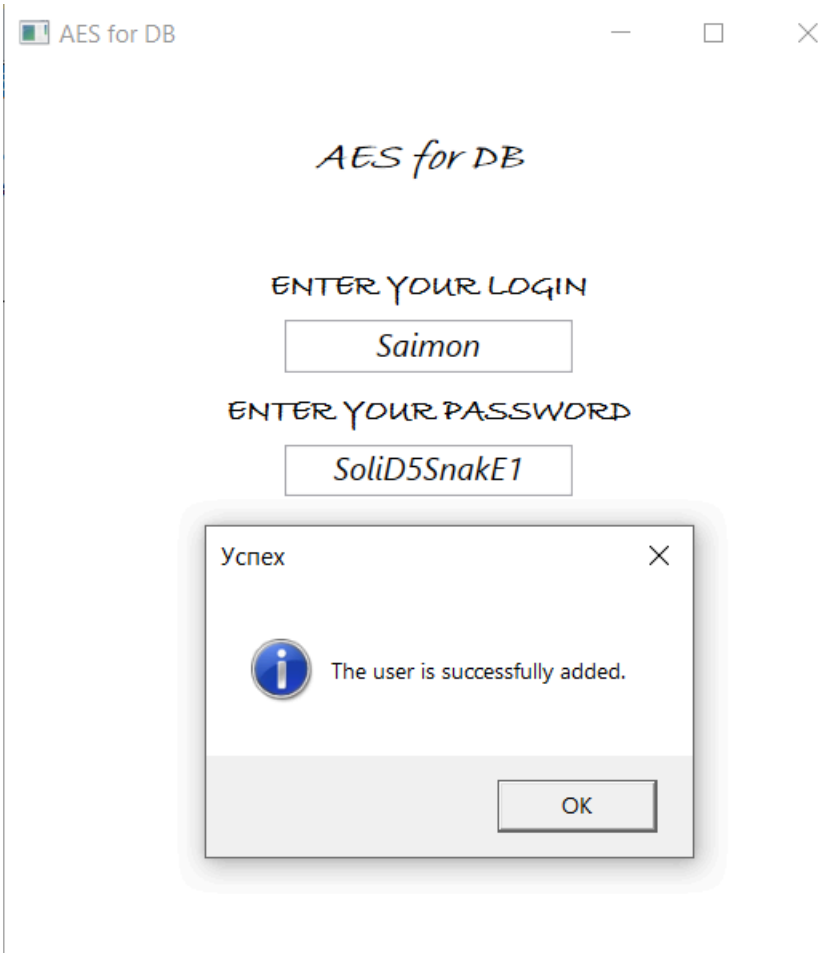


Рис. 3.12 Додавання користувача до бази даних

	Login	Password
	Login	0000
	Saimon	SoliD5SnakE1
	Stopper	FiNd0My9PpRoPeRt...

Рис. 3.13 Дані користувачів для входу після оновлення

Після входу користувач нарешті переходить до головного робочого вікна (рис. 3.13), в якому він може переглянути всі таблиці з бази даних університету, додати нові пункти та видалити вже існуючи. Шифрування проходить при

натисканні кнопки «Enter», після чого користувач може переглянути зашифровані дані (рис. 3.14).

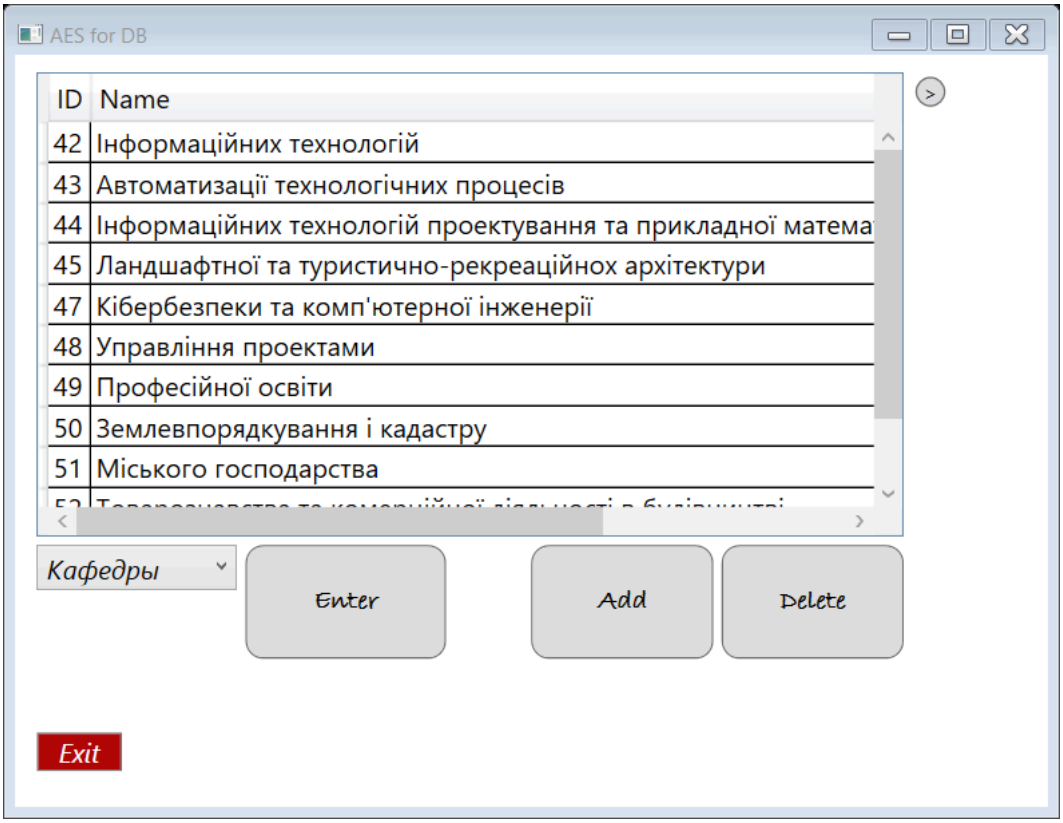


Рис. 3.14 Вікно для перегляду, передачі та шифрування бази даних

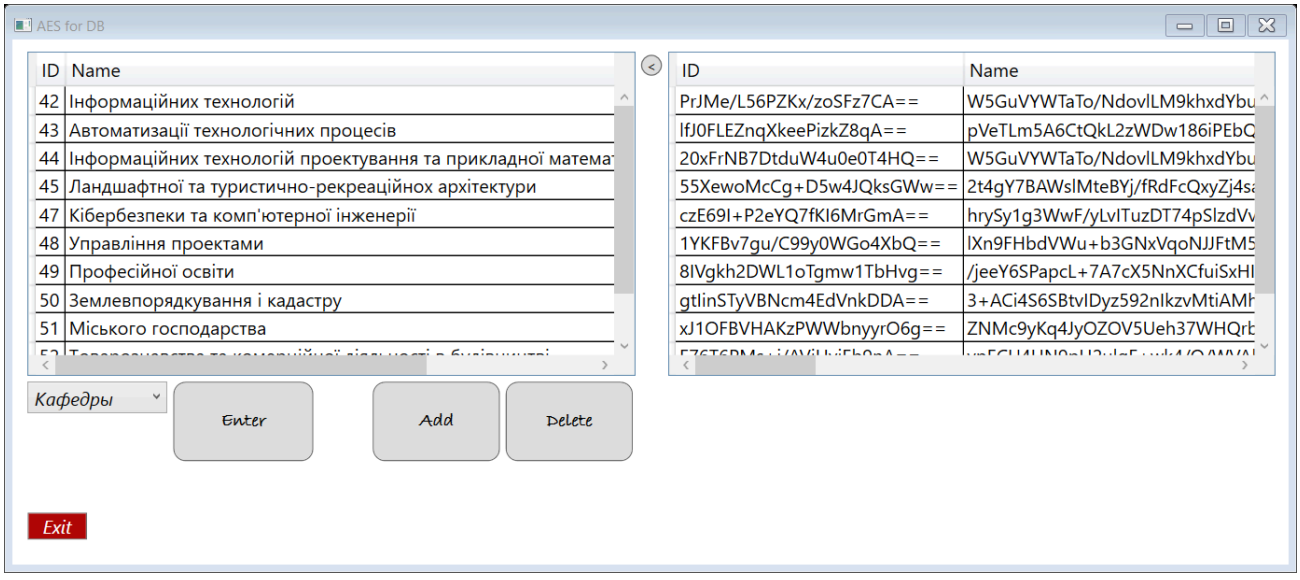


Рис. 3.15 Вікно з базою даних після шифрування та відкритим datagrid із зашифрованими даними

Після передачі зашифрованих даних, вони розшифровуються та передаються до нового вікна (рис. 3.15).

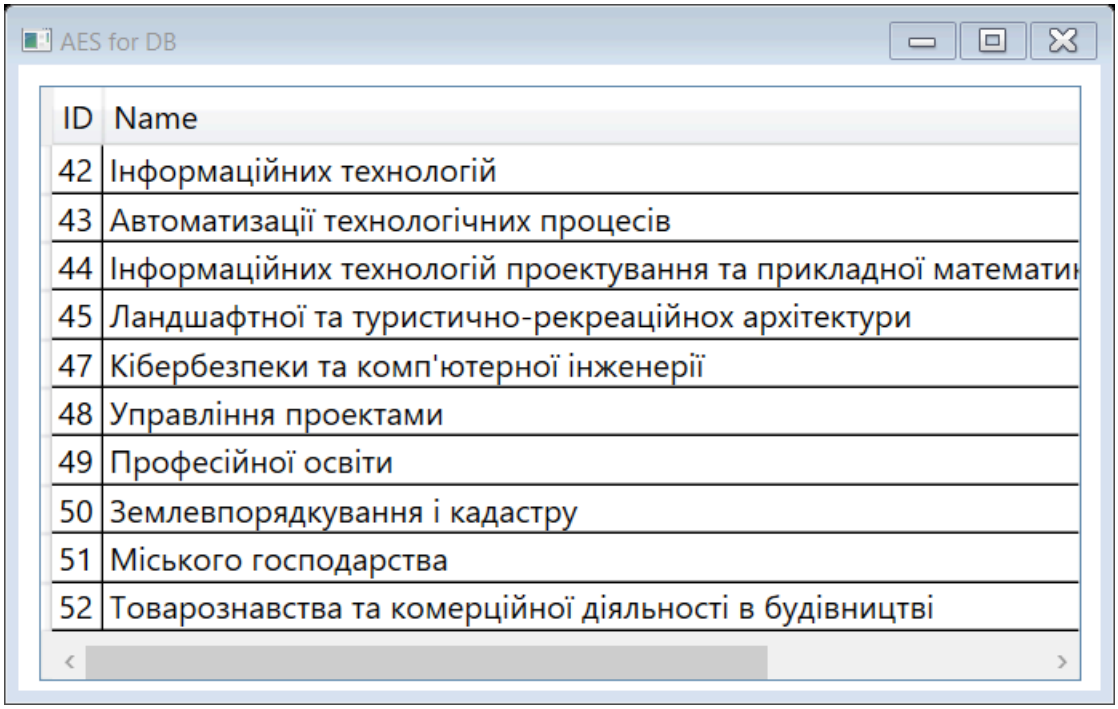


Рис. 3.16 Передана та розшифрована до іншого вікна база даних

4. ЕРГОНОМІКА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Всі ергономічні показники розділяються на декілька категорій [21]:

- **Гігієнічні** пов'язані з забезпеченням здоров'я та гігієни під час використання продукту або праці на робочому місці. Наприклад, це може включати правильне розташування екрану комп'ютера, забезпечення вентиляції, контроль за шумом та інші аспекти, що впливають на здоров'я користувача.
- **Антропометричні** визначають фізичні розміри людини, такі як розмір рук, ноги, зріст, ширина плечей тощо. Знання антропометрії дозволяє розробникам створювати продукти, які оптимально відповідають розмірам і формі тіла користувачів.
- **Фізіологічні** пов'язані з функціонуванням фізіологічних систем тіла людини під час використання продукту або перебування на робочому місці. Наприклад, це може включати ергономічне розташування клавіатури та миші для зменшення напруги на руках та спині.
- **Психофізіологічні** є взаємозв’язками між фізіологічними процесами та психічними станами людини. Вони враховуються при розробці продуктів або організації робочих місць для забезпечення комфорту та ефективності користувачів. Наприклад, яскраве освітлення або некоректно вибраний кольоровий дизайн можуть впливати на психологічний стан людини.
- **Психологічні** пов'язані з психологічними потребами, уподобаннями, ставленням та сприйняттям користувачів. Розуміння психологічних аспектів допомагає створювати продукти та робочі місця, які задовольняють потреби користувачів і сприяють їхньому комфорту та задоволенню.

Як розробникам системи шифрування, серед потрібних нам показників є останні два – психофізіологічні та психологічні. Ще одним впливаючим показником є гігієнічний, але він займає лише друге місце.

Зовнішній вигляд та інтерфейс системи повинні задовольняти наступним вимогам візуальної ергономіки:

- зручність інтерфейсу, у тому числі, використання меню з урахуванням віку графічного представлення об'єктів меню, що вчиться з максимальним застосуванням, використання піктографічних систем позначень, що сформувалися;
- зручність контекстно-залежної допомоги і спливаючих підказок;
- відповідність колірних, текстових, звукових рішень, інформативній насиченості і гармонійності екранів ергономічним вимогам до електронних видань і вікових психологічних особливостей користувачів.

В якості ергономічних властивостей призначеного для користувача інтерфейсу розглядаються: яскравість фону на екрані монітора, яскравість зображення на екрані, контрастність зображення, наявність поліекранних режимів відображення.

Серед вимог гігієнічних можна зазначити наявність не більше семи відкритих вікон на екрані. Або кількість інформаційних елементів у вікні, на розташування найбільш важливих часто використовуваних вікон та інформаційних елементів і так далі. Наприклад, найбільш комфортним для візуального сприйняття є розташування полів введення/виведення інформації в нижній частині екрану.

При розробці інтерфейсу потрібно дотримуватись єдиного стильового оформлення, до якого входять шрифти, кольори та рамки, для підвищення

естетичних показників, таких як цілісність композиції. Це дозволить отримати гармонічність між частинами та цілим. [20]

Відомий дизайнер Якуб Нільсен запропонував використовувати десять принципів проектування взаємодії між користувачем та спроектованою програмою. У табл. 4.1 було проведено дослід на відповідність створеної системи цим принципам.

Табл. 4.1 Відповідність принципам проектування

Назва	Опис	Відповідність створеної системи
Видимість статусу системи	Користувач завжди повинен бути в курсі того, що відбувається, отримуючи своєчасний і відповідний зворотний зв'язок.	Інтерфейс завжди знаходиться на потрібних вікнах та показує те, що відбувається в системі.
Відповідність між системою та реальним світом	Система має «розмовляти мовою користувача», використовуючи зрозумілі йому терміни і концепції, а не системно-орієнтовану мову.	Усі написи зрозумілі користувачам та не є системно-орієнтованими.

Продовження Таблиці 4.1 Відповідність принципам проектування

Назва	Опис	Відповідність створеної системи
Керованість та свобода для користувача	Користувач часто помилково вибирає системні функції і повинен мати видимий аварійний вихід з небажаного стану системи без складних діалогів. Слід підтримувати функції скасування (undo) та повтору (redo).	Функції видалення та додавання даних справляються з функціями скасування, але повтору в системі немає.
Узгодженість та стандарти	Користувачі не повинні гадати, чи означають одне й те саме різні слова, ситуації чи операції.	В інтерфейсі немає повторювань термінів.
Запобігання помилкам	Краще спроектувати систему так, щоб проблеми не виникали, ніж покладатися на добрі повідомлення про помилки. Слід усувати умови, що призводять до помилок, або попереджати користувача про можливі проблеми.	При використанні системи помилки не відображаються, але є повідомлення, якщо користувач виконує неприпустиму дію.

Продовження Таблиці 4.1 Відповідність принципам проектування

Назва	Опис	Відповідність створеної системи
Розпізнавати краще, ніж згадувати	Навантаження на пам'ять користувача має бути мінімізованим, явно показуючи йому об'єкти, дії та варіанти вибору. Користувач не повинен запам'ятовувати інформацію з однієї частини діалогу для використання в іншій.	Система є невеликою, навантаження на пам’ять практично немає.
Гнучкість та ефективність використання	Акселератори (засоби швидкого виконання команд), які новий користувач може помітити, часто прискорюють роботу досвідчених користувачів.	Система є невеликою, тому акселераторів немає.
Естетичний та мінімалістичний дизайн	В інтерфейсі не повинно бути непотрібної інформації або інформації, необхідної лише в окремих випадках.	Дизайн є мінімалістичним, без зайвої інформації.
Допомогти користувачеві поняття та виправити помилку	Повідомлення про помилки мають бути написані простою мовою, без кодів, чітко формулювати проблему та пропонувати конструктивне рішення.	Повідомлення є чіткими та зрозумілими.

Продовження Таблиці 4.1 Відповідність принципам проектування

Довідка та документація	Хоча ідеальним варіантом є система, яка не вимагає документації, необхідно надавати довідку та документацію. Інформація повинна бути легкою для пошуку, відповідати завданню користувача, описувати конкретні дії та не бути надто об'ємною.	Документації в прямому доступі немає.

Розглядаючи власний інтерфейс, можна виділити відповідність більшості описаних в цьому розділі аспектів, особливо зручності з’явлення вікон, відповідність єдиному стильовому оформленні, правильному розташуванні робочих елементів інтерфейсу та його можливостям не залежати від масштабу, який обирається користувачем.

ВИСНОВКИ

Визначено, що метод шифрування AES-256 забезпечує високу стійкість до різноманітних атак, хоч він поступається у швидкості іншим варіантам стандарту, але він має значні переваги у досконалості захисту конфіденційної інформації.

Доведено, що впровадження шифрування AES у системах управління базами даних забезпечує високий рівень захисту від несанкціонованого доступу, перехоплення та викрадення даних.

Для досягнення мети даної роботи щодо забезпечення захисту баз даних, за умов можливості використання отриманих результатів у системах значно більшого масштабу, розроблено спеціалізоване програмне забезпечення для захисту баз даних, ядром якого обрано стандарт шифрування AES-256.

Список використаних джерел

1. SQL в Access: основні поняття, глосарій і синтаксис [Електронний ресурс] // Microsoft: [сайт]. URL: https://support.microsoft.com/uk-ua/topic/sql-в-access-основні-поняття-глосарій-і-синтаксис-444d0303-cde1-424e-9a74-e8dc3e460671 (дата звернення: 02.06.2024).
2. Лекція 31. Захист інформації в комп'ютерних мережах [Електронний ресурс] // Інформаційний портал Технічного фахового коледжу: [сайт]. URL: https://e-tk.lntu.edu.ua/mod/page/view.php?id=3587 (дата звернення: 02.06.2024).
3. AES Advanced Encryption Standard (або Rijndael) [Електронний ресурс] // Технології мереж: [сайт]. URL: https://nettech.ua/news/aes-advanced-encryption-standard-ili-rijndael (дата звернення: 02.06.2024).
4. National Policy on the Use of the Advanced Encryption Standard (AES) to Protect National Security Systems and National Security Information [Електронний ресурс] // Committee on National Security Systems: [сайт]. URL: https://cryptome.org/aes-natsec.htm (дата звернення: 02.06.2024).
5. Niels Ferguson, Bruce Schneier. Practical Cryptography, 2003.
6. Comments from Readers [Електронний ресурс] // Schneier on Security: [сайт]. URL: https://www.schneier.com/crypto-gram/archives/2002/1015.html#8 (дата звернення: 02.06.2024).
7. Rijndael and other block ciphers [Електронний ресурс] // NESSIE: [сайт]. URL: https://web.archive.org/web/20031108050830/http://www.cosic.esat.kuleuven.ac.be/nessie/forum/read.php?f=1&i=82&t=82 (дата звернення: 02.06.2024).

8. Chu-wee Lim, Khoongming Khoo. An Analysis of XSL Applied to BES (англ.) // Fast Software Encryption. – Heidelberg: Springer Berlin / Heidelberg, 2007. – Vol. 4593. – P. 242-253.
9. Самойлов В.В. ОПИС АЛГОРИТМУ ШИФРУВАННЯ RIJNDAEL. ПРОСТА РЕАЛІЗАЦІЯ АЛГОРИТМУ НА МОВІ ПРОГРАМУВАННЯ C#. [Електронний ресурс] // Інтернет конференція: [сайт]. URL: http://www.konferenciaonline.org.ua/ua/article/id-72/ (дата звернення: 02.06.2024).
10. Aes.cs [Електронний ресурс] // Github: [сайт]. URL: https://github.com/dotnet/runtime/blob/5535e31a712343a63f5d7d796cd874e563e5ac14/src/libraries/System.Security.Cryptography/src/System/Security/Cryptography/Aes.cs (дата звернення: 02.06.2024).
11. Generate keys for encryption and decryption [Електронний ресурс] // Microsoft: [сайт]. URL: https://learn.microsoft.com/en-us/dotnet/standard/security/generating-keys-for-encryption-and-decryption (дата звернення: 02.06.2024).
12. Діаграми потоків даних (Data Flow Diagrams) [Електронний ресурс] // Maxym Zosym: [сайт]. URL: https://www.maxzosim.com/data-flow-diagrams/ (дата звернення: 02.06.2024).
13. Діаграма послідовності (Sequence Diagrams) [Електронний ресурс] // Maxym Zosym: [сайт]. URL: https://www.maxzosim.com/sequence-diagrams/ (дата звернення: 02.06.2024).
14. Лабораторна робота № 1 [Електронний ресурс] // Державний університет "Житомирська політехніка" - Освітній портал: [сайт]. URL: https://learn.ztu.edu.ua/pluginfile.php/14285/mod_resource/content/4/lmi_lr1_2017.docx.pdf (дата звернення: 02.06.2024).
15. XAML overview (WPF .NET) [Електронний ресурс] // Microsoft: [сайт]. URL:

https://learn.microsoft.com/en-us/dotnet/desktop/wpf/xaml/?view=netdesktop-8.0 (дата звернення: 02.06.2024).

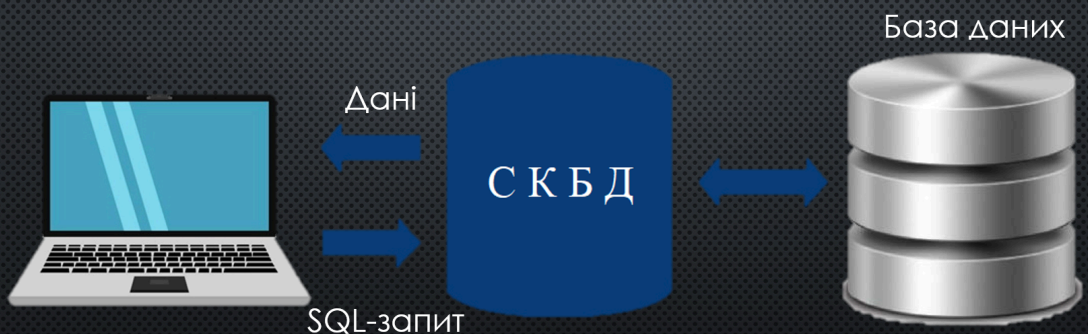
16. DataGrid Class [Електронний ресурс] // Microsoft: [сайт]. URL: https://learn.microsoft.com/en-us/dotnet/api/system.windows.controls.datagrid?view=windowsdesktop-8.0 (дата звернення: 02.06.2024).
17. What is SQL Server Management Studio (SSMS)? [Електронний ресурс] // Microsoft: [сайт]. URL: https://learn.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16 (дата звернення: 02.06.2024).
18. Aes class [Електронний ресурс] // Microsoft: [сайт]. URL: https://learn.microsoft.com/en-us/dotnet/api/system.security.cryptography.aes?view=net-8.0 (дата звернення: 02.06.2024).
19. rijndael_ingles2004.swf [Електронний ресурс] // WaybackMachine: [сайт]. URL: https://web.archive.org/web/20140529233954/http://www.cs.bc.edu/~straubin/cs381-05/blockciphers/rijndael_ingles2004.swf (дата звернення: 02.06.2024).
20. Ергономіка інформаційних технологій: конспект лекцій / уклад. О.О. Терентьєв. – К.: КНУБА, 2011. – 40 с.
21. Ергономіка інформаційних технологій: конспект лекцій / О.В. Горда. – К.: КНУБА, 2020. – 83с.

Додаток А
Інформаційні слайди

СИСТЕМА ЗАХИСТУ БАЗИ ДАНИХ ЗА ДОПОМОГОЮ СТАНДАРТУ ШИФРУВАННЯ AES

Керівник: Вишняков В.М.
Виконавець: Жураковський М.Е., КН-20-2
Кафедра Інформаційних Технологій
2024 р.

ЗАГАЛЬНИЙ ОПИС БАЗ ДАНИХ



Продовження додатку А
Інформаційні слайди

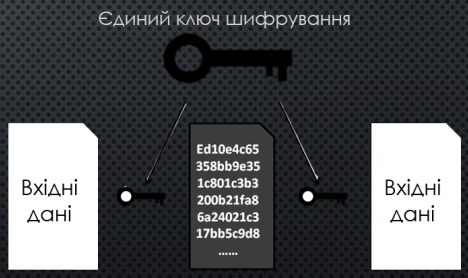
AES – ADVANCED ENCRYPTION STANDARD

- AES – СИМЕТРИЧНИЙ АЛГОРИТМ БЛОКОВОГО ШИФРУВАННЯ.

Розмір блоку: 128 біт

Ключ шифрування: 128/192/256 біт

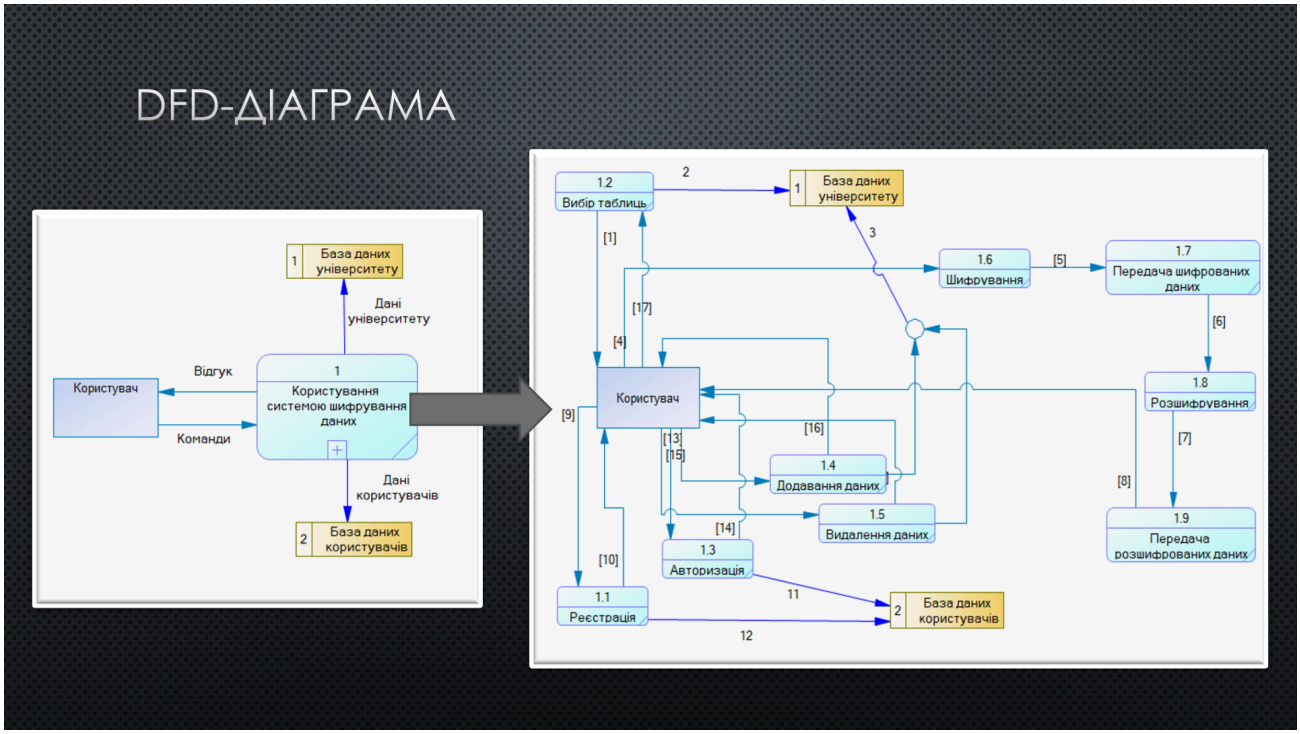
Число раундів: 10/12/14



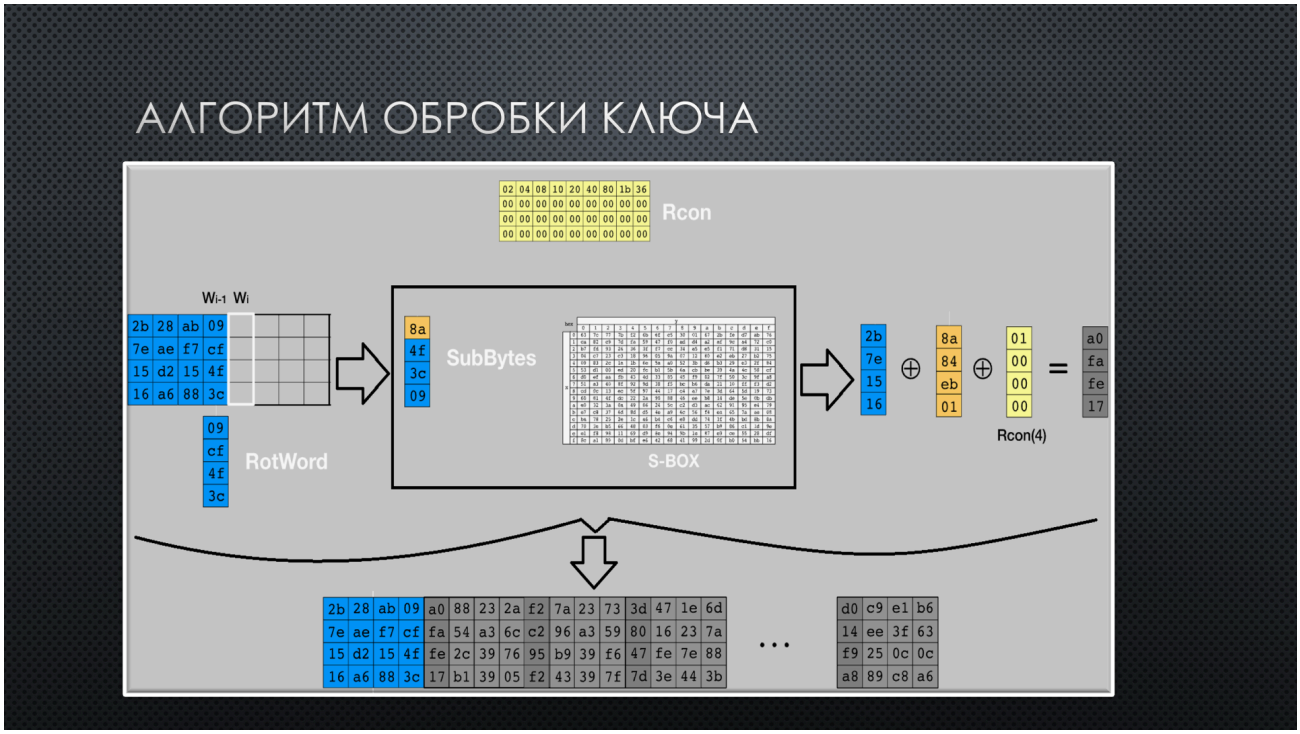
ПОРІВНЯННЯ РОЗМІРІВ ШИФРУВАННЯ

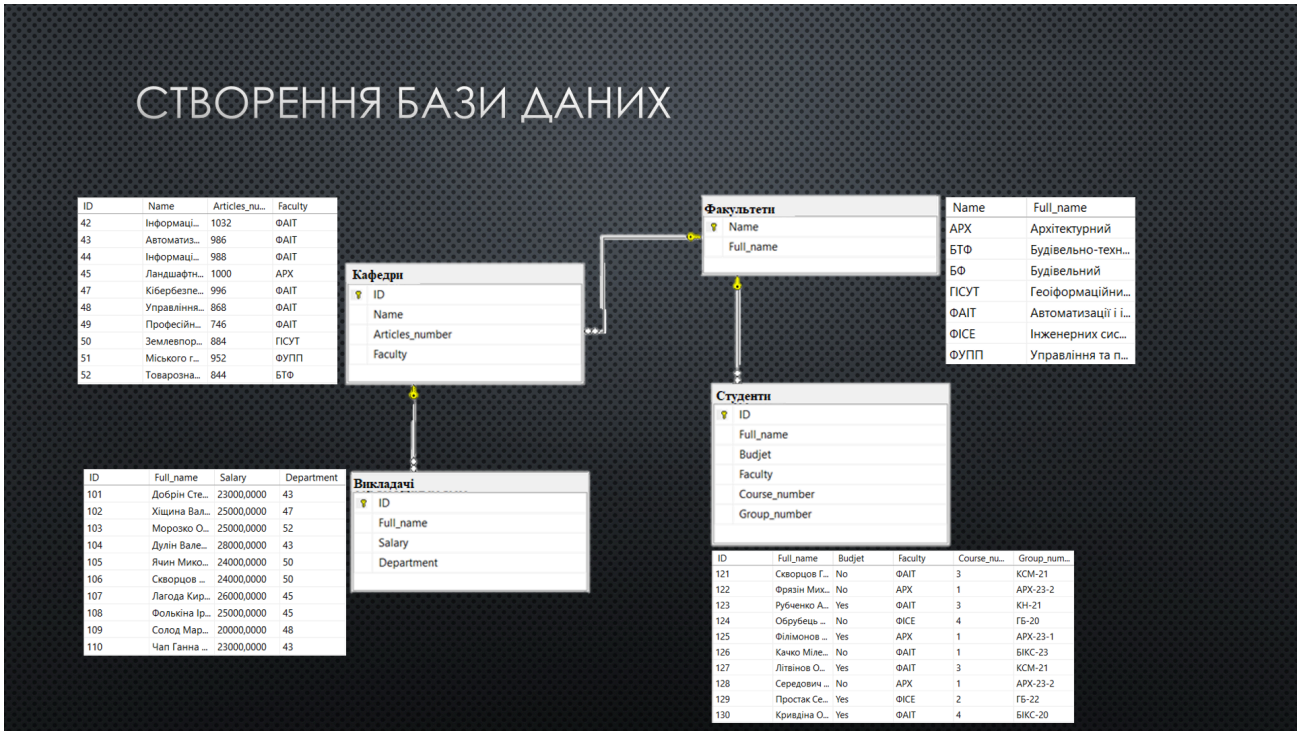
Назва	Довжина	Рівень безпеки	Швидкість та продуктивність
AES-128	128 бітів	Стандартний рівень безпеки.	Швидше, ніж 192-бітне та 256-бітне шифрування.
AES-192	192 біти	Підвищений рівень безпеки, ніж 128-бітове шифрування.	Повільне шифрування, довший ключ.
AES-256	256 бітів	Максимальний рівень безпеки.	Повільне шифрування, найдовший ключ.

Продовження додатку А
Інформаційні слайди



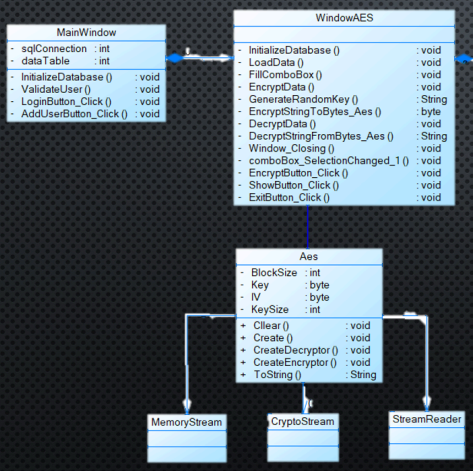
Продовження додатку А
Інформаційні слайди





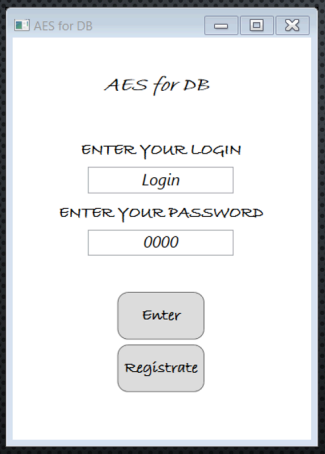
Продовження додатку А
Інформаційні слайди

ДІАГРАМА КЛАСІВ

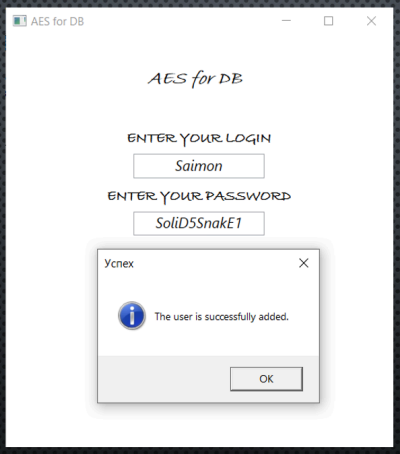


- WindowAes є залежним від існування MainWindow, а Window1 – від WindowAes. Між WindowAes та Aes простий зв'язок використання. Також показано додаткові класи, пов'язані з Aes.

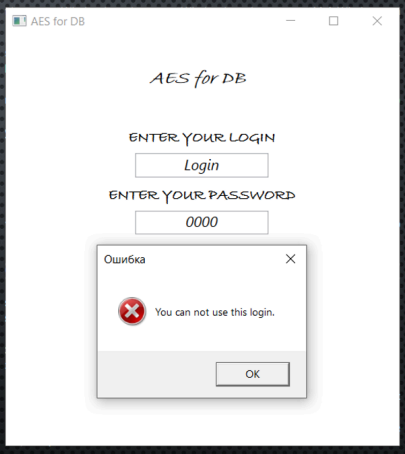
ТЕСТОВИЙ ПРИКЛАД, ВІКНО АВТОРИЗАЦІЇ



Вікно авторизації



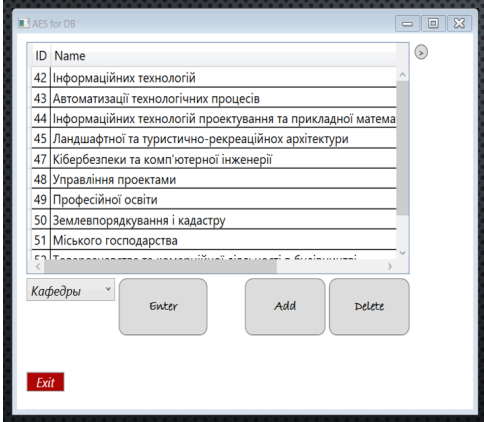
Вікно авторизації після регістрації



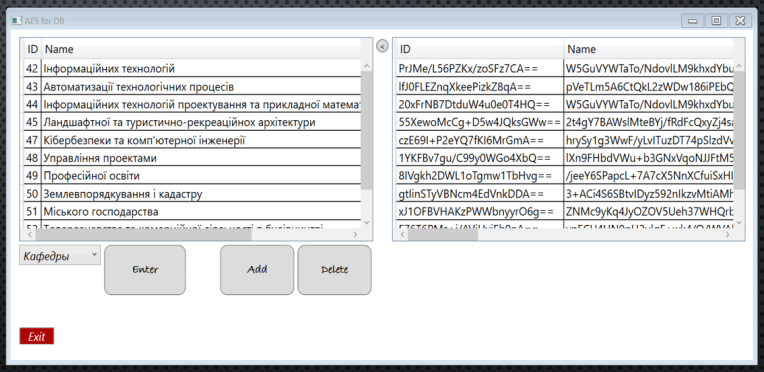
Вікно авторизації при використанні існуючого логіна для регістрації

Продовження додатку А
Інформаційні слайди

ТЕСТОВИЙ ПРИКЛАД, ВІКНО ШИФРУВАННЯ



Головне вікно



Головне вікно після перегляду зашифрованих даних

ВИСНОВКИ

- Метод шифрування AES-256 забезпечує високу стійкість до різноманітних атак; поступається у швидкості іншим варіантам стандарту; має значні переваги у досконалості захисту конфіденційної інформації.
- Впровадження шифрування AES у системах управління базами даних забезпечує високий рівень захисту.
- Розроблено спеціалізоване програмне забезпечення для захисту баз даних, ядром якого обрано стандарт шифрування AES-256.