

Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра
на тему: «Проект підсистеми захисту бази даних Web-
орієнтованої інформаційної системи»

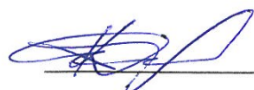
Виконала студентка 4-го курсу групи КІ-4
спеціальності 123 «Комп'ютерна інженерія»

 Карина БЛЮМБЕРГ

Керівник

 Ірина ПАНАРІНА

Рецензент

 Інна СУГОНЯК

Житомир – 2022

Державний університет «Житомирська політехніка»

Факультет інформаційно-комп'ютерних технологій

Кафедра комп'ютерної інженерії та кібербезпеки

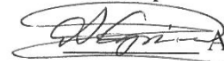
Спеціальність 123 «Комп'ютерна інженерія»

Освітня програма «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Єфіменко

1 березня 2022 р.

ЗАВДАННЯ

на випускню кваліфікаційну роботу бакалавра

Студент: Блюмберг Карина Олегівна

Тема роботи: «Проект підсистеми захисту бази даних Web-орієнтованої інформаційної системи»

затверджена Наказом університету від 1 березня 2022 р. № 127/с

Термін здачі студентом закінченої роботи "10" *06* 2022

Вихідні дані роботи (зазначається предмет і об'єкт дослідження):

Предмет досліджень – програмні засоби підсистеми захисту бази даних Web-орієнтованої інформаційної системи.

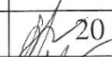
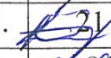
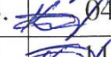
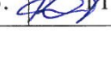
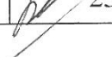
Об'єкт дослідження – процес захисту бази даних Web-орієнтованої інформаційної системи.

Обладнання та програмне забезпечення:

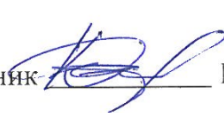
1. Операційна система Windows 10;

2. Програмне забезпечення Microsoft Visual Studio 2017 та MS SQL Server 2012
3. Мова програмування с#
4. Технологія програмування ASP.NET

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Кузьменко О. В.	 15 лютого 2022	 20 березня 2022
2	Кузьменко О. В.	 21 березня 2022	 02 квітня 2022
3	Кузьменко О. В.	 04 квітня 2022	 10 черня 2022
4	Кузьменко О. В.	 11 червня 2022	 25 черня 2022

Студент  Карина Блюмберг

Керівник  Ірина Панаріна

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	14 лютого 2022 – 20 лютого 2022	Виконано
2	Пошук та аналіз аналогічних сайтів	20 лютого 2022 - 1 березня 2022	Виконано
3	Вивчення матеріалу по заданій тематиці	2 березня 2022 – 12 березня 2022	Виконано
4	Планування структури програми	12 березня 2022 – 22 березня 2022	Виконано
5	Проектування та реалізація архітектури веб-системи	23 березня 2022 – 6 квітня 2022	Виконано
6	Написання програмного коду	7 квітня 2022 – 25 травня 2022	Виконано
7	Тестування роботи програми	26 травня 2022 – 28 травня 2022	Виконано
8	Аналіз отриманих результатів	29 травня 2022 – 10 червня 2022	Виконано
9	Оформлення пояснювальної записки	11 червня 2022 – 20 червня 2022	Виконано
10	Попередній захист	25 червня 2022	Виконано
11	Захист		

Студент  Карина Блюмберг

Керівник  Ірина Панаріна

АНОТАЦІЯ

Дипломна робота «Проект підсистеми захисту бази даних Web-орієнтованої інформаційної системи» на здобуття освітнього ступеня «Бакалавр» за напрямом підготовки «Комп'ютерна інженерія» виконана обсягом 54 сторінки машинного тексту, 16 рисунків, 3 таблиць, 1 додатку, 22 бібліографічних найменування за переліком посилань.

Актуальність. Найбільш поширеним методом створення надійної інформаційної системи – є створення в даній системі підсистеми захисту інформації. Для його здійснення існують безліч програмних засобів, що містять в собі різноманітний функціонал який забезпечує підтримку нормального стану захищеності інформаційної системи. Однією з найбільш поширених функцій – є безпосереднє шифрування цих даних у Web-орієнтованій системі та їх подальша передача захищеними каналами зв'язку. Таким чином, реалізування цієї функції програмним компонентом підсистеми захисту бази даних Web-орієнтованої інформаційної системи є актуальним.

Об'єкт дослідження – процес захисту бази даних Web-орієнтованої інформаційної системи.

Предмет досліджень – програмні засоби підсистеми захисту бази даних Web-орієнтованої інформаційної системи.

Мета дослідження – програмно реалізувати компонент підсистеми захисту бази даних Web-орієнтованої інформаційної системи.

Методи дослідження. Теоретичною основою досліджень є теорія моделювання процесів. Зокрема, теорія функціонального, процесного моделювання і моделювання потоків даних – для створення концептуальної моделі програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи; теорія об'єктно-орієнтованого моделювання – для створення структури програмного компоненту підсистеми захисту бази даних Web-орієнтованої

					КІК.КР.Б - 123 - 22 - ПЗ			
Зм.	Арк.	№ докум.	Підпис	Дата	«Програмний компонент підсистеми захисту бази даних Web-орієнтованої інформаційної системи»	Літ.	Арк.	Аркушів
Розроб.		Блюмберг К.О.		19.07				
Керівник		Панаріна І.В.		29.08				
Реценз.		Сусоняк І.І.		29.08				
Н. контр.		Головня О.С.		29.08				
Затверд.		Єфіменко А.А.		29.08		ФІКТ, Гр. КІ-4		

інформаційної системи.

Одержані результати. У дипломній роботі одержано такі результати:

- створено концептуальну модель підсистеми захисту бази даних Web-орієнтованої інформаційної системи, використання якої дозволяє формалізувати його застосування на рівні функцій, процесів і потоків даних, а також сформулювати та обґрунтувати варіанти використання програмного компоненту;
- створено структуру програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи на основі формалізування його роботи на рівні функцій, процесів і потоків даних, використання якої дозволяє забезпечити їх функціональну придатність;
- розроблено програмний компонент підсистеми захисту бази даних Web-орієнтованої інформаційної системи на основі його структури, використання якого дозволяє шифрувати інформаційні ресурси на передавати їх у відповідні БД незахищеними каналами зв'язку.

Ключові слова: ІНФОРМАЦІЙНА СИСТЕМА, КОМП'ЮТЕРНА СИСТЕМА, КРИПТОГРАФІЯ, СИМЕТРИЧНЕ ШИФРУВАННЯ, ПІДСИСТЕМА ЗАХИСТУ БАЗИ ДАНИХ, WEB-ОРІЄНТОВАНА ІНФОРМАЦІЙНА СИСТЕМА, КОНЦЕПТУАЛЬНА МОДЕЛЬ ПРОГРАМНОГО КОМПОНЕНТУ, ОБ'ЄКТНО-ОРІЄНТОВАНА МОДЕЛЬ ПРОГРАМНОГО КОМПОНЕНТУ, ПРОГРАМНИЙ КОМПОНЕНТ.

					КІК.КРБ – 123 – 22 – ПЗ			
					«Програмний компонент підсистеми			
Зм.	Арк.	№ докум.	Підпис	Дата	захисту бази даних Web-орієнтованої інформаційної системи»	Літ.	Арк.	Аркуші
Розроб.		Блюмберг К.О.						
Керівник		Панаріна І.В.						
Реценз.		Сугоняк І.І.						
Н. контр.		Головня О.С.						
Затверд.		Сфіменко А.А.				ФІКТ, Гр. КІ-4		

ANNOTATION

The thesis "Software component of the database protection subsystem of a Web-based information system" for the educational degree "Bachelor" in the field of study "Computer Engineering" was completed in the amount of 54 pages of machine text, 16 figures, 3 tables, 1 application, 22 bibliographic titles on list of links.

Relevance. The most common method of creating a reliable information system is to create an information protection subsystem in this system. For its implementation, there are many software tools that contain a variety of functionality that ensures the maintenance of a normal state of information system security. One of the most common functions is the direct encryption of data in a Web-based system and their subsequent transmission over insecure communication channels. Therefore the implementation of this function by the software component of the database protection subsystem of a Web-based information system is relevant.

The object of research is the process of protecting the database of a Web-oriented information system.

The subject of research is the software of the database protection subsystem of a Web-oriented information system.

The purpose of the study is to programmatically implement a component of the database protection subsystem of a Web-oriented information system.

research methods. The theoretical basis of research is the theory of process modeling. In particular, the theory of functional, process modeling and data flow modeling to create a conceptual model of the software component of the database protection subsystem of a Web-based information system; theory of object-oriented modeling - creation of the structure of the software component of the database protection subsystem of a Web-based information system.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Таваріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

Results have been received. The following results were obtained in the thesis work:

- a conceptual model of the database protection subsystem of a Web-oriented information system has been created, the use of which allows formalizing its application at the level of functions, processes and data flows, as well as forming and justifying options for using a software component;
- the structure of the software component of the database protection subsystem of the Web-oriented information system was created based on the formalization of its work at the level of functions, processes and data flows, the use of which allows to ensure their functional suitability;
- a software component of the database protection subsystem of a Web-based information system has been developed based on its structure, the use of which allows you to encrypt information resources for transferring them to the appropriate databases by insecure communication channels.

Keywords: information system, computer system, cryptography, symmetrical encryption, database protection subsystem, web-oriented information system, conceptual model of the software component, object-oriented model of the software component, software component.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	10
ВСТУП	11
РОЗДІЛ 1. АНАЛІЗ ПРОЦЕСУ ЗАХИСТУ ІНФОРМАЦІЙНОЇ СИСТЕМИ ТА ФОРМАЛІЗАЦІЯ ВИМОГ ДО ПРОГРАМНОГО КОМПОНЕНТУ	12
1.1 Аналізування процесу захисту інформаційної системи.	12
1.2 Формування вимог до програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи	16
Висновки за розділом 1	17
РОЗДІЛ 2. КОНЦЕПТУАЛЬНА МОДЕЛЬ ПРОГРАМНОГО КОМПОНЕНТУ ПІДСИСТЕМИ ЗАХИСТУ БАЗИ ДАНИХ WEB-ОРІЄНТОВАНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ	18
2.1 Формалізування діяльності підсистеми захисту підсистеми захисту бази даних Web-орієнтованої інформаційної системи	18
2.2 Встановлення зв'язків між етапами підсистеми захисту бази даних Web-орієнтованої інформаційної системи	22
2.3 Встановлення потоків даних між етапами підсистеми захисту бази даних Web-орієнтованої інформаційної системи	25
Висновки за розділом 2	28
РОЗДІЛ 3. ОБ'ЄКТНО-ОРІЄНТОВАНА МОДЕЛЬ ПРОГРАМНОГО КОМПОНЕНТУ ПІДСИСТЕМИ ЗАХИСТУ БАЗИ ДАНИХ WEB-ОРІЄНТОВАНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ	29
3.1 Специфікування вимог до програмного компоненту	29
3.2 Створювання логічної структури програмного компоненту	34
3.3 Створювання фізичної структури програмного компоненту	37
3.3.1 Створювання компонентів і зв'язків між ними.	38
3.3.2 Створювання вузлів і зв'язків між ними	40
Висновки за розділом 3	43

**РОЗДІЛ 4. ПРОГРАМНА РЕАЛІЗАЦІЯ КОМПОНЕНТУ ПІДСИСТЕМИ
ЗАХИСТУ БАЗИ ДАНИХ WEB-ОРІЄНТОВАНОЇ ІНФОРМАЦІЙНОЇ
СИСТЕМИ**

44

4.1 Реалізування програмного компоненту

44

4.2 Тестування програмного компоненту

47

4.2.1 Аналіз якості програмного забезпечення

47

4.2.2 Опис процесів тестування

51

4.2.3 Опис контрольного прикладу

52

4.3 Використання програмного компоненту

53

Висновки за розділом 4

59

ВИСНОВКИ

60

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

61

ДОДАТКИ

64

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змі	Арк.	№ докум	Підпис	Дата		

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

Основні умовні позначення та їх розшифрування:

IV – вектор ініціалізації

ОС – операційна система

ПЗ – програмне забезпечення

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змін	Арк.	№ докум	Підпис	Дата		

ВСТУП

З розвитком інформаційних технологій все більше організацій, державних та комерційних, мають змогу користуватися глобальною мережею Інтернет. Як наслідок, комп'ютерні системи і мережі зазнають тисяч різних атак як ззовні, так і зсередини. Для належного захисту інформації від атак, що здійснюють зловмисники, використовуються різні методи, способи та засоби, зокрема, й програмні. Це обумовлює необхідність захисту інформаційних ресурсів, що циркулюють як в межах звичайної локальної мережі, так і в межах глобальної мережі Інтернет. Одним із способів надійної передачі інформації незахищеними каналами зв'язку – є використання криптографічних засобів захисту інформації, тобто шифрування даних за допомогою симетричних та асиметричних алгоритмів шифрування. В даному випадку розглядається шифрування за допомогою симетричного блокового алгоритму AES-128 та надійна передача даних до БД Web-орієнтованої інформаційної системи.

Тому одним з основних завдань підсистеми захисту бази даних Web-орієнтованої інформаційної системи – створення та функціонування процесів надійного та якісного шифрування даних, а також безпечної їх передачі на відповідні БД Web-орієнтованої інформаційної системи. Таким чином, реалізування цієї функції програмним компонентом підсистеми захисту бази даних Web-орієнтованої інформаційної системи є актуальним.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

РОЗДІЛ 1. АНАЛІЗ ПРОЦЕСУ ЗАХИСТУ ІНФОРМАЦІЙНОЇ СИСТЕМИ ТА ФОРМАЛІЗАЦІЯ ВИМОГ ДО ПРОГРАМНОГО КОМПОНЕНТУ

1.1 Аналізування процесу захисту інформаційної системи

Інформаційна (автоматизована) система являє собою організаційно-технічну систему, що об'єднує обчислювальну (комп'ютерну) систему (апаратне та програмне забезпечення), фізичне середовище (зовнішнє середовище та контрольована зона), персонал (керівництво, адміністратори і користувачі) і оброблювальну інформацію. Тоді як однією зі складових її складових є інформаційна система [1 - 4].

Комп'ютерна система – інформаційно-технічний комплекс (сукупність апаратного та програмного забезпечення), який призначений для обробки, модифікації, вводу та виводу інформації [5 - 8].

У кожній інформаційній системі повинні бути забезпечені механізми (методи, засоби та способи) захисту інформації, що циркулює в системі, та процесів її обробки. Для здійснення належного захисту потрібно регулярно проводити пошук вразливостей, які можуть призвести до порушення конфіденційності, цілісності та доступності інформації. Цей пошук проводиться з метою постійного покращення та оновлення програмного забезпечення. Одним з основних методів захисту інформаційних ресурсів, для запобігання порушення конфіденційності, цілісності та доступності інформації – це використання криптографічних методів захисту та створених на їх основі програмного забезпечення [10].

Криптографія – мистецтво, наука та технологія забезпечення секретності інформації. Вивчає способи та методи захисту інформації від змін і неавторизованого втручання при передаванні, обробленні та зберіганні [10]. Тобто, основна ціль криптографії – передача захищеного повідомлення від

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змін	Арк.	№ докум	Підпис	Дата		

одного пункту до іншого, таким чином, щоб зміст повідомлення був відомий та зрозумілий лише відправнику та отримувачу повідомлення.

Криптографія, в основному, складається з двох головних процесів – шифрування та розшифрування. Шифрування — у системах обробки інформації – алгоритмічне (криптографічне) перетворення даних, яке виконується у посимвольній послідовності з метою одержання шифрованого тексту [11 - 12]. Розшифрування — процес санкціонованого перетворення зашифрованих даних у придатні для читання [13 - 14]. Тобто, шифрування призначене для приховування інформації, що передається, від втручання сторонніх осіб, яким вона не призначається для ознайомлення чи модифікації. А розшифрування призначене для отримання користувачем відкритого тексту з шифртексту.

Також є можливість отримання відкритого тексту з шифртексту користувачем, для якого цей текст був непризначений, тобто зловмисником. Процес несанкціонованого отримання інформації з зашифрованих даних називається дешифруванням [13 - 14]. При цьому ключ дешифрування зазвичай невідомий.

Дія шифру керується як алгоритмами, та, в кожному випадку, ключем. Ключ — це секретний параметр (в ідеалі, відомий лише двом сторонам) для окремого контексту під час передачі повідомлення. Ключі мають велику важливість, оскільки без змінних ключів алгоритми шифрування легко зламуються і непридатні для використання в більшості випадків. Історично склалось так, що шифри часто використовуються для шифрування та дешифрування, без виконання додаткових процедур, таких як аутентифікація або перевірка цілісності.

Дослідження характеристик мов, що мають будь-яке відношення до криптології, таких як частоти появи певних літер, комбінацій літер, загальні шаблони тощо, називається криптолінгвістикою.

Основними методами шифрування, на основі яких будуються алгоритми шифрування, є симетричне та асиметричне шифрування.

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Змі	Арк	№ докум	Підпис	Дата		

Симетричне шифрування – це метод шифрування при якому один і той же ключ (що зберігається в секреті) використовується як для шифрування, так і для розшифрування. На основі даного методу розроблено ефективні (швидкі й надійні) методи шифрування.

Переваги симетричного шифрування:

- Шифри із симетричним ключем спроектовані так, щоб мати велику пропускну здатність;
- Ключі для шифрів із симетричним ключем відносно короткі
- Шифри із симетричним ключем можна використати як примітиви для побудови різних криптографічних механізмів включно з псевдовипадковими генераторами чисел, хеш-функціями, обчислювально ефективних схем підпису та ін;
- Шифри із симетричним ключем можна комбінувати для отримання сильніших шифрів.

Недоліки симетричного шифрування:

- При зв'язку між двома особами, ключ потрібно тримати в секреті на обох кінцях;
- У великій мережі потрібно опікуватись багатьма ключами;
- У зв'язку між двома особами криптографічна практика вимагає частої зміни ключів.

Проблемою симетричного шифрування є необхідність передачі ключа, для розшифрування інформації, таким чином ключ може бути перехоплений кимось іншим.

Будь хто, знаючи секретний ключ, може розшифрувати інформацію. Тоді як в асиметричному шифруванні є два пов'язаних ключа — пара ключів. Відкритий ключ (англ. public key) — публічний, до нього повинні мати доступ всі ті, хто матиме потребу зашифрувати інформацію. Тоді як закритий ключ —

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

приватний ключ (англ. private key), повинен бути доступним лише тому хто має право розшифрувати інформацію, за своїм розміром він значно більший від секретного ключа симетричного шифрування [13 - 14].

Будь-яку інформацію, зашифровану за допомогою відкритого ключа можна розшифрувати лише застосовуючи той самий алгоритм, але з використанням відповідного приватного ключа. Також всю інформацію, зашифровану за допомогою приватного ключа, можна розшифрувати лише за допомогою відповідного відкритого ключа. Це означає, що немає необхідності хвилюватись за передачу ключа, відкритий ключ повинен бути публічним. Але асиметричне шифрування є значно повільнішим від симетричного. Також потребує значно більше обчислювальної потужності як для шифрування, так і для розшифрування інформації.

Через вади в швидкодії асиметричного методу цей метод доводиться використовувати разом з симетричним (асиметричні методи на 3 — 4 порядки повільніші). Так, для розв'язання задачі ефективного шифрування з передаванням секретного ключа, використаного відправником, інформація спочатку симетрично зашифровується випадковим ключем, потім цей ключ зашифровують відкритим асиметричним ключем одержувача, після чого повідомлення і ключ відправляються по мережі.

Програмний компонент підсистеми захисту бази даних Web-орієнтованої ін-формаційної системи буде невеликим некомерційним проектом, що працювати в межах локальної мережі та буде здійснювати обмін інформацією в межах цієї мережі. Для даної задачі буде достатньо використання симетричного методу шифрування, на основі одного ключа шифрування та розшифрування.

Симетричні алгоритми шифрування можна розділити на потокові та блочні алгоритми шифрування. Поточкові алгоритми шифрування послідовно обробляють текст повідомлення. Блочні алгоритми працюють з блоками фіксованого розміру. Найбільш поширеними алгоритмами симетричного шифрування – є потокові з довжиною ключа 128 та 256 біт. Одним із таких алгоритмів є Advanced Encryption Standard (AES) або ж коротко алгоритм AES.

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

Даний алгоритм буде використовуватися в процесі розробки програмного компоненту.

1.2 Формування вимог до програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи

Програмний компонент використовуватиметься для захисту бази даних Web-орієнтованої інформаційної системи, а саме шифрування інформаційних ресурсів на стороні клієнта та їх подальшу передачу до відповідних БД.

З огляду на це до програмного компоненту повинні пред'являтися ряд вимог до його структури та функціонування. Такими вимогами є:

1) програмний компонент повинен бути класифікований та побудована за наступними вимогами:

- за способом організації – основі архітектури клієнт-сервер;
- за сферою діяльності – програмний продукт захисту бази даних Web-орієнтованої інформаційної системи.

2) програмний компонент повинен бути побудований мовою програмування високого рівня на основі об'єктно-орієнтованого підходу;

3) програмний компонент повинен мати можливість роботи як в межах локальної мережі, так і підтримувати вихід в глобальну мережу Інтернет;

4) програмний компонент повинен мати зручний графічний інтерфейс для користувачів;

5) програмний компонент повинен підтримувати «локалізацію»;

6) програмний компонент повинен підтримуватися найбільш популярними операційними системи персональних комп'ютерів. Зокрема:

- Microsoft Windows (Windows XP, Windows Vista, Windows 7, Windows 8, Windows 10);
- Mac OS X;

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Змін	Арк.	№ докум.	Підпис	Дата		

- Unix-подібні операційні системи (Linux, FreeBSD).

Висновки за розділом 1

Проаналізовано процес захисту інформаційних систем за допомогою криптографічних методів. Серед них було вибрано алгоритм симетричного блокового шифрування AES-128, на основі якого буде створюватися програмний компонент.

За результатами аналізу розроблено вимоги до програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи. Зокрема, сформовано вимоги до його структури та функціонування.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

РОЗДІЛ 2. КОНЦЕПТУАЛЬНА МОДЕЛЬ ПРОГРАМНОГО КОМПОНЕНТУ ПІДСИСТЕМИ ЗАХИСТУ БАЗИ ДАНИХ WEB-ОРІЄНТОВАНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Формалізування діяльності підсистеми захисту підсистеми захисту бази даних Web-орієнтованої інформаційної системи

Для формалізування діяльності підсистеми захисту бази даних Web-орієнтованої інформаційної системи обрано методологію IDEF0 (Integrated DEFinition Methods). IDEF0 (Integration Definition for Function Modeling) - нотація опису бізнес-процесів. Заснована на методології SADT. SADT (Structured Analysis and Design Technique, технологія структурного аналізу і проектування) – графічні позначення і підхід до опису систем. Тобто, IDEF0 використовується для створення функціональної моделі, що відображає структуру і функції системи, а також потоки інформації і матеріальних об'єктів, що зв'язують ці функції [15].

Методологія IDEF0 заснована на принципі того, що модель – це штучний об'єкт, що представляє собою відображення (образ) системи і її компонентів[15]. Причому, кожний компонент пояснює певний функціонал системи, що пред'являється до системи та порядок його використання та взаємодії з іншими компонентами цієї ж системи. Для більш кращого розуміння всіх процесів, що відбуваються в системі, над кожним компонентом проводиться декомпозиція (розбивання на декілька менших функціональних частин). Вона проводиться до тих пір, поки ця процедура є доцільною, з точки зору розробника системи.

Компонент системи, в методології IDEF0, представляється у вигляді набору взаємодіючих і взаємопов'язаних блоків, що відображають процеси, операції, дії, що відбуваються в системі, що вивчається [15]. Компоненти системи, або як їх ще називають – функціями, представляються за допомогою прямокутників. Взаємодіями між функціями відбуваються за допомогою інтерфейсів, які

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

представляються за допомогою стрілок, що входять в блок або такими, що виходять з нього. Причому, вхідні стрілки – це початкова (вхідна) інформація, що перетворюється функцією і на її виході, за допомогою вихідних стрілок, отримується результат перетворення (вихідна інформація). Далі, цей результат стає початковою інформацією, для наступної функції, що взаємодіє з попередньою. В методології IDEF існує чотири види стрілок [15]:

- зліва вхідна стрілка – вхід процесу – інформація, яка буде перетворена в ході виконання функції;
- справа вихідна стрілка – вихід процесу – перетворена інформація;
- зверху вхідна стрілка – управління процесу – інформація або документ, який визначає як повинен виконуватися функція;
- знизу вхідна стрілка – механізм процесу – те, що перетворює вхід в вихід: співробітники або техніка.

Кожна модель IDEF0 являє собою покрокову, ітеративну процедуру, в якій існує ряд строгих формальних правил, що забезпечують переваги методології щодо однозначності, точності і цілісності складних багаторівневих моделей [15].

Сформулюємо мету моделювання [15]: створення підсистеми захисту бази даних Web-орієнтованої інформаційної системи шляхом шифрування вибраних користувачем файлів та їх передача у відповідні БД в зашифрованому вигляді. А також, можливість розшифрування файлів та отримання вихідних (відкритих файлів).

Розпочнемо з побудови функціональної моделі (див. рис. 2.1). Відповідно до опису системи основною діяльністю є підсистема захисту, що складається з підсистеми шифрування та підсистеми розшифрування файлів, що присутні в комп'ютерній системі.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змін	Арк.	№ докум.	Підпис	Дата		

Для належного розуміння даного процесу потрібно визначити вхідні та вихідні дані, а також механізми та управління [15]:

- вхідними даними для підсистеми захисту (підсистеми шифрування та підсистеми розшифрування) є відкритий файл, що потрібно зашифрувати та розшифрувати. Даний файл вибирається користувачем системи за допомогою файлового менеджера ОС.
- вихідними даними для підсистеми захисту є зашифровані та розшифровані файли пентесту;
- управління в даному проекті є період та час за який підсистема захисту виконує виконує свої функції;
- механізмами проведення захисту виступають користувач системи.

З огляду на це функціональна модель програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи матиме такий вигляд як на рис. 2.1.

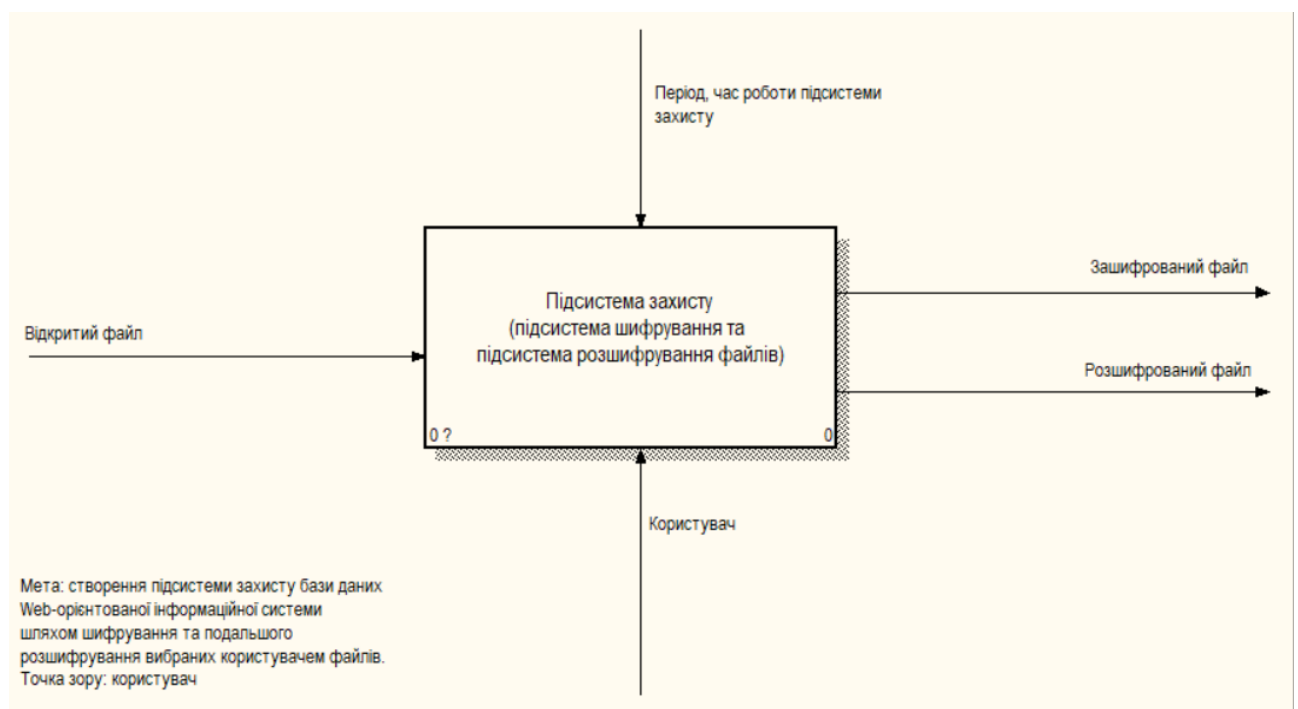


Рисунок 2.1 – Функціональна модель програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи

Проведемо декомпозицію функціональної моделі (див. рис. 2.2), описавши послідовність функцій даної підсистеми захисту:

- шифрування файлу;
- розшифрування файлу.

Таким чином, програмний компонент представляється двома підсистемами, кожна з яких реалізує окрему функцію. Отримані підсистеми можна розбивати на менші частини, до тих пір поки не досягнуть належної точності, яку вимагають від даних декомпозицій. Так як даний компонент не складний, то проводити подальшу декомпозицію немає потреби.

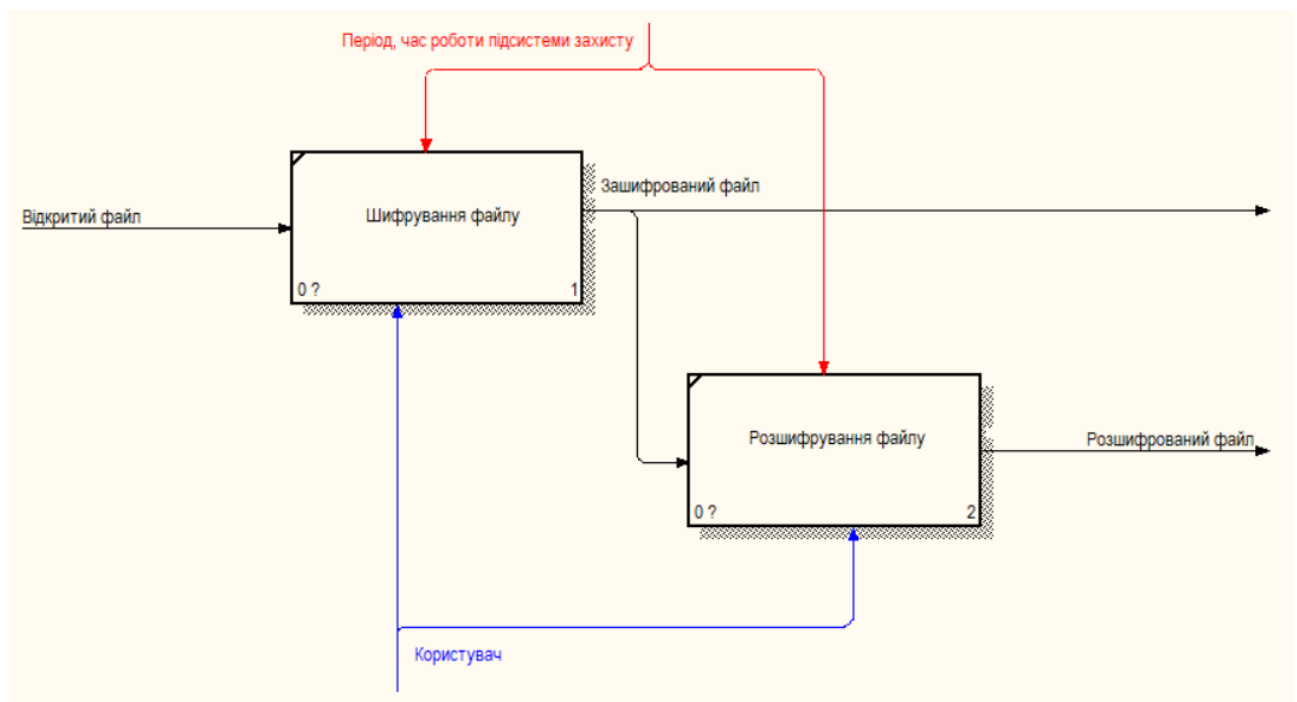


Рисунок 2.2 – Декомпозиція функціональної моделі програмного компоненту

Робота програмного компоненту починається з підсистеми «Шифрування файлу». В даній підсистемі на вхід подається відкритий файл. В результаті

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змін	Арк.	№ докум.	Підпис	Дата		

роботи даної підсистеми отримуємо зашифрований файл. Після цього робота підсистеми «Шифрування файлу» завершується та починається виконання функції «Розшифрування файлу».

У підсистему «Розшифрування файлу» на вхід подається зашифрований користувачем файл, який вибирається з відповідного списку раніше зашифрованих файлів. В результаті, на виході, отримуємо розшифрований файл. Після цього робота підсистеми «Розшифрування файлу» завершується.

Підсистеми «Шифрування файлу» та «Розшифрування файлу» працюють за допомогою симетричного блокового алгоритму AES-128

2.2 Встановлення зв'язків між етапами підсистеми захисту бази даних Web-орієнтованої інформаційної системи

Для встановлення зв'язків між етапами підсистеми захисту бази даних Web-орієнтованої інформаційної системи обрано методологію IDEF3 як способу опису процесів з використанням структурованого методу, що дозволяє експерту з предметної області уявити стан речей як упорядковану послідовність подій з одночасним описом об'єктів, що мають безпосереднє відношення до процесу[16]. Тобто, даний тип діаграм описує послідовність певних дій що виконуються в системі. На відміну від більшості технологій моделювання процесів, IDEF3 не має жорстких синтаксичних або семантичних обмежень, які роблять незручним опис неповних або нецілісність систем [14]. Також, IDEF3-моделювання органічно доповнює традиційне моделювання з використанням стандарту методології IDEF0 шляхом задання послідовності одиниць робіт означеної діяльності.

Однією з головних елементарних одиниць нотації IDEF3 – є робота. Робота (Unit of Work, activity) – елемент моделі типу IDEF3, що відображає певну дію, яка комбінується з іншими діями в деяку послідовність [16]. Дані дії відображаються за допомогою прямокутника. Кожній дії присвоюють унікальний ідентифікаційний номер та ім'я, виражене віддієсловним іменником,

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

що позначає процес дії. Одночасно в одну роботу може входити та виходити з неї тільки по одній стрілці.

Іншим елементом моделей нотації IDEF3 – є зв'язки [16]. Зв'язки створюються для встановлення логічного зв'язку між діями. Головною їх особливістю є те, що вони односпрямовані і, зазвичай, організовується зліва направо. У діаграмах нотації IDEF3 виділяють три види зв'язків [16]:

- зв'язок типу *«тимчасове передування»*. Вихідна дія повинна повністю завершитися, перш ніж почнеться виконання кінцевої дії. У багатьох випадках завершення одного дії ініціює початок виконання іншої;
- зв'язок типу *«об'єктний потік»*. Деякий об'єкт, який є результатом виконання вихідної дії, необхідний для виконання кінцевого дії;
- зв'язок типу *«нечітке відношення»*. Виділення відносин між діями, які неможливо описати з використанням попередніх або об'єктних зв'язків. Одне із застосувань нечітких відношень – відображення взаємовідносин між паралельно виконуються діями.

З'єднання [16]. З'єднання розбивають або з'єднують внутрішні потоки і використовуються для опису розгалуження процесу.

Об'єкт посилання [16]. Об'єкт посилання слугує для вираження ідеї, концепції або даних, які не можна пов'язати зі стрілкою, з'єднанням або роботою.

Далі, згідно з наступним етапом концептуального моделювання, в нотації IDEF3 [16], побудуємо процесну модель програмного компоненту (див. рис. 2.3). Початковим процесом функціонування програмного компоненту, що моделюється, є його запуск Користувачем та увімкнення функції «Шифрування файлу». Як було сказано раніше, за допомогою цієї функції відбувається шифрування вибраного файлу за допомогою алгоритму блокового шифрування

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змн	Арк	№ докум	Підпис	Дата		

AES-128. Після закінчення процедури шифрування – відбувається запис ID зашифрованого файлу, назви файлу, зашифрованих байтів (зашифрованого файлу) та дати шифрування в БД «Files». А в БД «Encrypt» - записується ID зашифрованого файлу, псевдорандомний пароль, ключ шифрування та вектор ініціалізації IV (більш детально про ці дані описано в четвертому розділі).

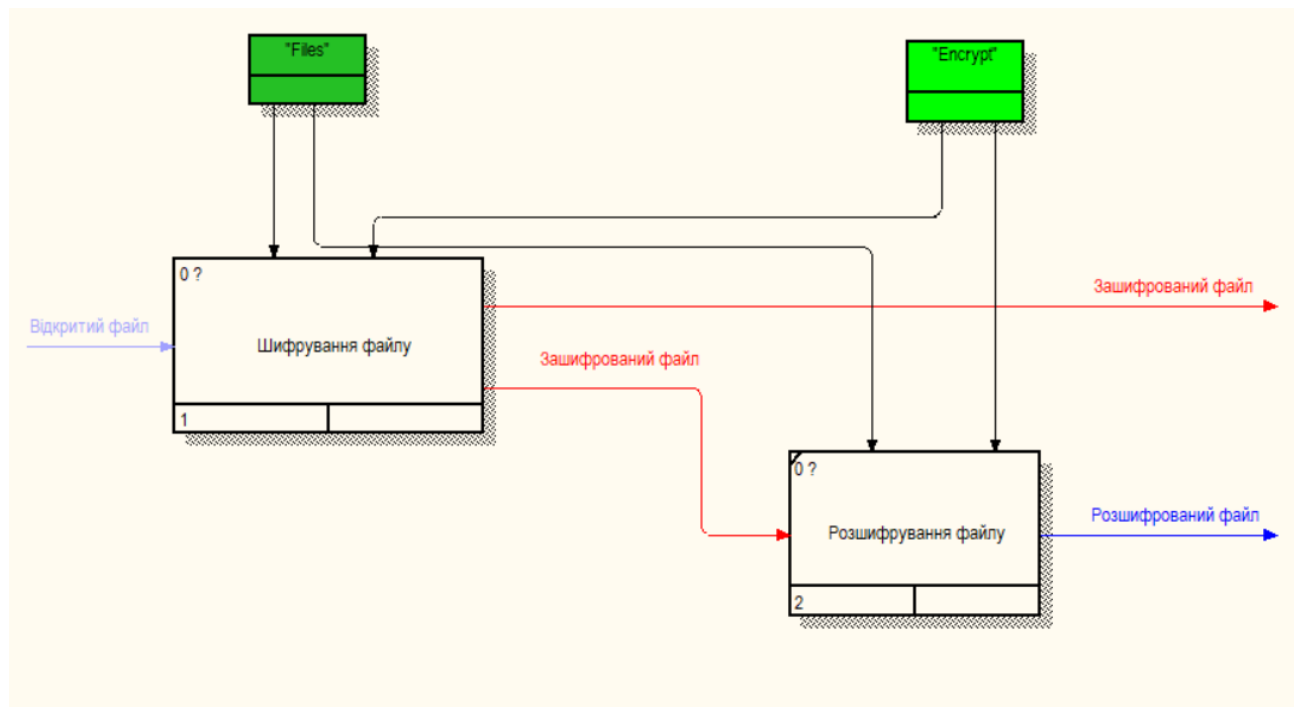


Рисунок 2.3 – Процесна модель програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи

Наступний процес функціонування програмного компоненту, що моделюється, – це увімкнення функції «Розшифрування файлу». Вона відбувається за рахунок вибору одного із зашифрованих файлів, зі списку доступних. В результаті виконання даного процесу – отримуємо розшифрований файл. Для розшифрування файлу потрібно використовувати дані, що записані в БД «Files» та «Encrypt».

Як результат, Користувач має змогу захистити свій файл шляхом його шифрування та передачі останнього у відповідні БД. А також, отримати вихідний файл з даних БД шляхом розшифрування.

2.3 Встановлення потоків даних між етапами підсистеми захисту бази даних Web-орієнтованої інформаційної системи

Для встановлення потоків даних між етапами підсистеми захисту бази даних Web-орієнтованої інформаційної системи обрано методологію DFD (Data Flow Diagram). Діаграма потоків даних (англ. Data Flow Diagram або DFD) – модель проектування, графічне представлення «потоків» даних в інформаційній системі. Діаграма потоків даних також може використовуватись для візуалізації процесів обробки даних (структурне проектування) [17]. Тобто, методологія описує зовнішні по відношенню до системи джерела і адресати даних, логічні функції, потоки даних і сховища даних, до яких здійснюється доступ.

Дана нотація складається з чотирьох елементів [17]:

- процес (англ. Process) – це функція або послідовність дій, які потрібно зробити, щоб дані були оброблені. Тут немає суворої системи синтаксичних вимог;
- потік даних (англ. Data flow). В нотації відображається у вигляді стрілок, які показують, яка інформація входить, а яка виходить з того чи іншого блоку на діаграмі;
- сховище даних (англ. Data store). Внутрішнє сховище даних для процесів в системі. Надійшли дані перед обробкою і результат після обробки, а також проміжні значення повинні десь зберігатися;

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

- зовнішні сутності (англ. External Entity). Це будь-які об'єкти, які не входять до самої системи, але є для неї джерелом інформації або одержувачами будь-якої інформації з системи після обробки даних.

Нотація DFD призначена для розуміння процесів руху потоків даних – їх появу, обробку і напрямки надсилання результатів [17].

Відповідно до нотації DFD, основним процесом є «Підсистема захисту» [17]. На даній діаграмі зображено один процес та одну зовнішню сутність (див. рис. 2.4). Зовнішньою сутністю виступає Користувач, що використовує програмний компонент. Дана сутність передає дані на обробку процесом «Підсистема захисту» та отримує оброблені результати, які в свою чергу стають новими вхідними даними для заданого процесу. Ці дії повторюють до тих пір поки не буде досягнуто кінцевої цілі – отримання зашифрованого та розшифрованих файлів, а також передачі даних шифрування у відповідні БД. Процес «Підсистема захисту» виконує такі ж дії з точністю навпаки – спочатку отримує вхідні дані від зовнішньої сутності, які обробляє, та видає результат.

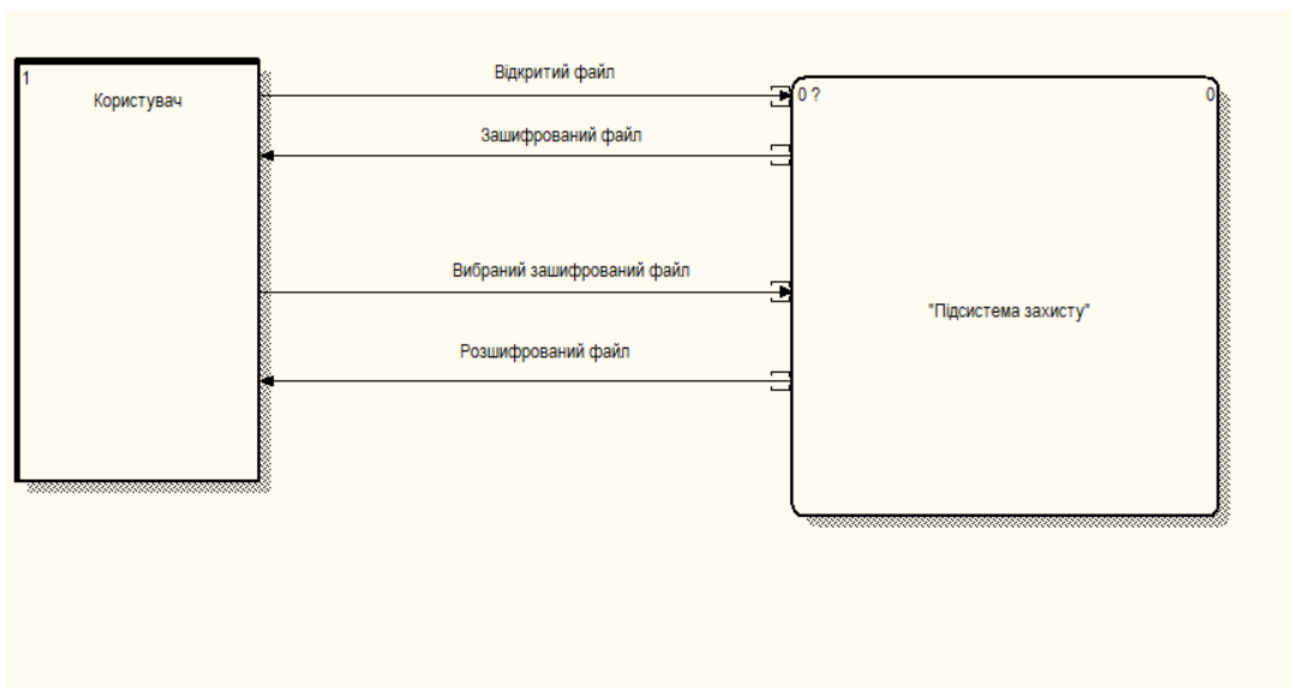


Рисунок 2.4 – Модель потоків даних програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи

Розглянемо діаграму декомпозиції даної процесної моделі програмного компоненту. Елементи даної діаграми пов'язані послідовністю потоків даних – їх отримання, обробкою та передачею. Тобто, головними частинами діаграми є сховища даних.

Для коректної роботи даної системи необхідно всього дві БД, а точніше файли – це «Files» і «Encrypt». В БД «Files» містяться ID зашифрованого файлу, назва файлу, зашифровані байтів файлу (зашифрований файл) та дата зашифрування. В БД «Encrypt» – ID зашифрованого файлу, псевдорандомний пароль, ключ шифрування та вектор ініціалізації IV (більш детально про ці дані описано в четвертому розділі). Дані БД потрібні для коректної роботи програмного компоненту (див. рис. 2.5).

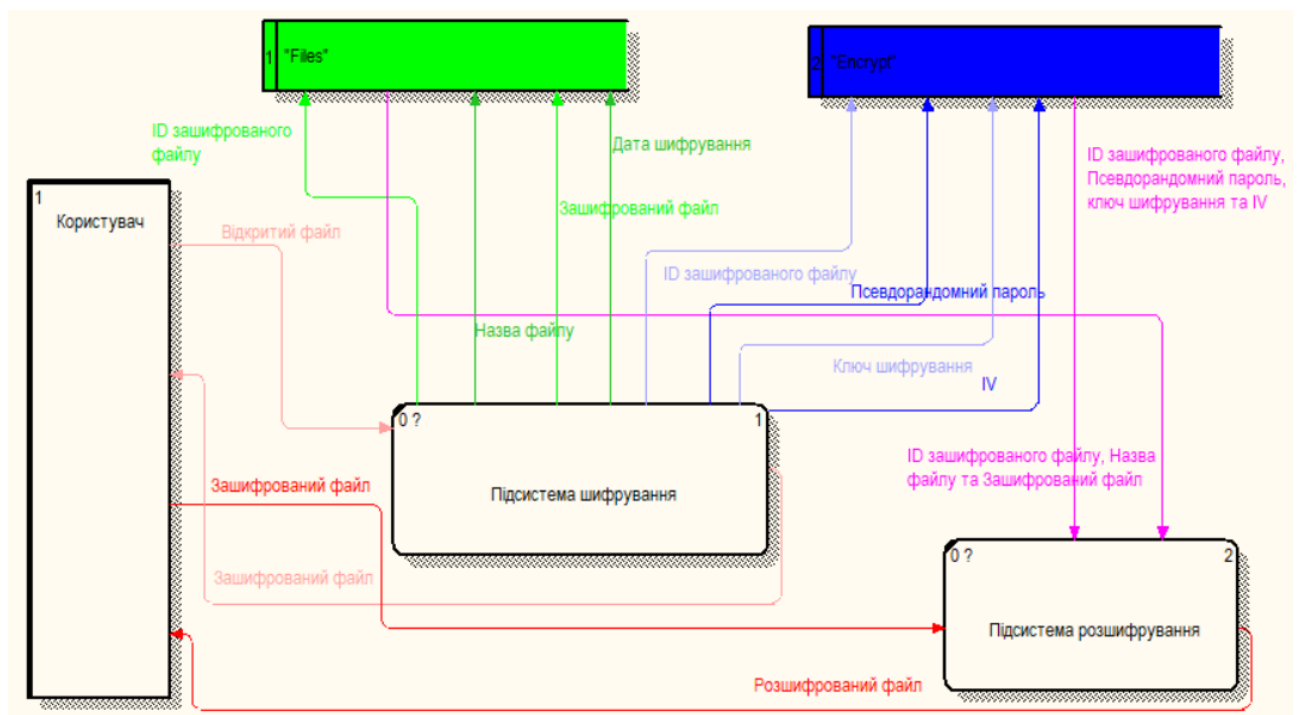


Рисунок 2.5 – Декомпозиція моделі потоків даних

Потік даних в програмному компоненті, що моделюється за допомогою діаграми нотації DFD (див. рис. 2.5), починається з передачі відкритого файлу від користувача до «Підсистеми шифрування». Дана підсистема створює псевдорандомний пароль для шифрування файлу. На основі даного паролю

створюється ключ шифрування та вектор ініціалізації IV. Завдяки пароллю, ключу та IV відбувається процедура шифрування файлу (на основі симетричного блокового алгоритму AES-128). В результаті роботи даної системи отримуємо відкритий файл, який передається користувачу, а також наступні дані записуються до БД:

- ID зашифрованого файлу, назва файлу, зашифровані байтів файлу (зашифрований файл) та дата зашифрування записуються в БД «Files»;
- ID зашифрованого файлу, псевдорандомний пароль, ключ шифрування та вектор ініціалізації IV записуються в БД «Encrypt».

Далі, відбувається робота «Підсистеми розшифрування». В дану підсистему на вхід подаються вибраний користувачем зашифрований файл (зі списку зашифрованих), а також наступні дані з відповідних БД:

- ID зашифрованого файлу, назва файлу, зашифровані байтів файлу (зашифрований файл) з БД «Files»;
- ID зашифрованого файлу, псевдорандомний пароль, ключ шифрування та вектор ініціалізації IV з БД «Encrypt».

В результаті роботи «Підсистеми розшифрування» - отримуємо розшифрований (відкритий) файл, який система передає користувачу на ознайомлення.

Висновки за розділом 2

Створено концептуальну модель програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи, використання якої дозволяє формалізувати його застосування на рівні функцій, процесів і потоків

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

даних, а також сформулювати та обґрунтувати варіанти використання програмного компоненту.

Процес моделювання розпочато з визначення контексту, тобто найбільш абстрактного рівня опису програмного компоненту в цілому, у контекст якого входили визначення мети та точки зору на концептуальну модель. На функціональному рівні концептуальної моделі відображено програмний компонент тестування на проникнення в комп'ютерну систему. Для кожної з функцій сформовано вхідні та вихідні дані. Тоді як на процесному рівні концептуальної моделі відображено послідовність одиниць роботи при роботі підсистеми захисту. На рівні потоків даних концептуальної моделі відображено потоки даних між визначеними на функціональному рівні функціями з урахуванням інформаційних і зовнішніх об'єктів.

РОЗДІЛ 3. ОБ'ЄКТНО-ОРІЄНТОВАНА МОДЕЛЬ ПРОГРАМНОГО КОМПОНЕНТУ ПІДСИСТЕМИ ЗАХИСТУ БАЗИ ДАНИХ WEB-ОРІЄНТОВАНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Специфікування вимог до програмного компоненту

Створення, опис та деталізація об'єктно-орієнтованої моделі будь-якого програмного застосунку починається зі специфікування вимог до нього. Тобто, на початку даного моделювання потрібно в'яснити сферу застосування програмного застосунку, його функціональне призначення та функціональні вимоги. Дану проблему дозволяє вирішити використання моделей в нотації мови UML, а саме – діаграми варіантів використання [18].

Діаграма варіантів використання – діаграма, на якій зображуються варіанти використання проектованої системи, укладені в кордон (границю) суб'єкта, і зовнішні актори, а також взаємовідношення між акторами і варіантами використання [18]. Тобто, дана діаграма описує функціональне призначення системи в найзагальнішому вигляді з точки зору всіх її користувачів і

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлова І.В.				
Змі	Арк	№ докум	Підпис	Дата		

зацікавлених осіб. Модель, що описується за допомогою діаграми варіантів використання, пояснює те, що повинна робити система (або окремий програмний застосунок) в процесі свого функціонування, не зачіпаючи питання про те, як вона повинна це робити. Дану діаграму можна уявити у вигляді системи типу «чорна скриня» – в неї подаються вхідні дані/параметри, які система обробляє та перетворює в певний результат (вихідні дані/параметри). Причому, її не цікавить технічна та фізична сторона реалізації виконання варіантів використання і взаємодія акторів з системою.

Мета створення діаграми варіантів використання полягає в досягненні наступних цілей [18]:

- визначення спільних кордонів (границь) функціональності проектованої системи в контексті модельованої предметної області;
- уточнення вимог до функціональної поведінки проектованої системи в формі варіантів використання;
- розроблення вихідної концептуальної моделі системи для її подальшої деталізації у формі логічних і фізичних моделей;
- підготовлення вихідної документації для взаємодії розробників системи з її замовниками і користувачами.

Дані цілі досягаються за допомогою взаємодії екторів і варіантів використання [18].

Ектор (actor) являє собою будь-яку зовнішню по відношенню до проектованої системи сутність, яка взаємодіє з системою і використовує її функціональні можливості для досягнення певної мети або вирішення приватних завдань. Тобто, ектори використовуються для моделювання зовнішніх по відношенню до проектованої системи сутностей, які взаємодіють з системою і використовують її в якості окремих користувачів. В якості екторів можуть виступати будь-які об'єкти, суб'єкти або системи, які ззовні

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

взаємодіють з модельованою системою. Як приклад, це може бути людина, програмний застосунок, технічний пристрій чи інша система або підсистема. Ектора в системі визначає розробник даної системи.

Загальноприйнятим графічним позначенням ектора на діаграмах в нотації мови UML є фігурка «дротяного чоловічка», під яким записується обов'язкове ім'я ектора.

Варіант використання (use case) являє собою загальну специфікацію сукупності виконуваних системою дій з метою надання деякого спостережуваного результату, який має значення для одного або декількох акторів. Іншими словами, варіант використання являє собою закінчений фрагмент поведінки системи з точки зору тих чи інших зацікавлених осіб, а також описує сервіси (їх функціонал), які система надає в користування актору. Тобто, кожен варіант використання – це певна послідовність дій, які повинні бути виконані системою або в системі, з метою надання відповідного результату або сервісу певному актору. Сукупність усіх варіантів використання, що розглядаються в контексті поведінки проектованої системи, описує кордон (границю) модельованої системи або утворює її суб'єкт. А суб'єкт (subject), в свою чергу, – це елемент моделі, який володіє функціональною поведінкою.

Загальноприйнятим графічним зображенням варіанту використання в нотації мови UML є еліпс, всередині якого або нижче його записується обов'язкове ім'я (name) в формі рядка тексту. Ім'я використовується для ідентифікації окремого випадку використання, як елементу моделі. кожен

Взаємодія між варіантами використання та акторами здійснюється за допомогою так званих «відношень». Під «відношенням» (relationship) в мові UML розуміється довільний семантичний взаємозв'язок між окремими елементами моделі. Тобто, на діаграмі варіантів використання, відношення пов'язують акторів з варіантами використання, а також акторів і варіанти використання між собою. Згідно з стандартами мови UML, є чотири види відношень, які зображуються на даному типі діаграм [18]:

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Змін	Арк.	№ докум.	Підпис	Дата		

1) відношення асоціації (association) – відношення, що призначене для позначення взаємодії актора з варіантом використання. Іншими словами, на діаграмі варіантів використання асоціація завжди є бінарною і пояснює семантичні особливості окремої взаємодії актора і варіанту використання. Графічно відношення асоціації позначається за допомогою простої стрілки в формі літери «V».

2) відношення узагальнення (generalization) – відношення, що призначене для пояснення того факту, що один елемент моделі є спеціалізацією або окремим випадком (нащадком) іншого елемента моделі, який в свою чергу є прашуром для попереднього. Дане відношення може пов'язувати лише елементи одного типу. Графічно відношення узагальнення позначається суцільною лінією зі стрілкою в формі не закрашеного трикутника, яка вказує на загальний елемент моделі.

3) відношення включення (include) – відношення, що призначене для пояснення того факту, що деякий варіант використання містить поведінку, що визначена в іншому варіанті використання. Ставлення включення може пов'язувати в моделі тільки два варіанти використання. Графічно дане відношення позначається як відношення залежності в формі пунктирною лінії з «V»-подібною стрілкою, спрямованою від залежного варіанту використання до незалежного варіанту використання.

4) відношення розширення (extend) – відношення, що показує взаємодію одного варіанти використання з деяким іншим варіантом використання, функціональність або поведінка якого задіється першим не завжди, а тільки при виконанні деяких додаткових умов. Розширення є спрямованим бінарним відношенням, оскільки два варіанти використання в цьому відношенні завжди є впорядкованими. Графічно дане відношення позначається як відношення залежності в формі пунктирною лінії з «V»-подібної стрілкою, спрямованої від залежного варіанту використання до незалежного варіанту використання і з'єднаної з ним в так званій точці розширення (extension point).

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

Вимоги до програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи специфіковано діаграмою варіантів використання (див. рис. 3.1) [18]. На даній діаграмі відбувається взаємодія між ектором системи та варіантами використання.

Ектором в даній системі виступає Користувач. Користувач – це будь-яка фізична (людина) чи юридична (компанія) особа, що використовує програмний компонент з метою захисту файлу шляхом його шифрування та подальшою відправкою зашифрованих даних у відповідні БД, що знаходять на локальному сервері.

Головними варіантами використання виступають «Проведення шифрування» та «Проведення розшифрування», робота яких починається безпосередньо після запуску програми. «Проведення шифрування» включає в себе процедуру «Шифрування файлу» – здійснює шифрування вибраного користувачем файлу. Шифрування здійснюється за допомогою симетричного блокового алгоритму AES-128. Проте, дана процедура шифрування можлива лише за умови «Вибір файлу» - процедура, за допомогою якої відбувається попередній вибір файлу для шифрування (за допомогою файлового менеджера ОС). Після здійснення шифрування, відбувається процедура «Запис даних до відповідних БД», яка є складовою процедури «Проведення шифрування». Процедура запису даних складається з двох частин:

- У першій частині відбувається запис ID зашифрованого файлу, назву файлу, зашифровані байти (зашифрований файл) та дату зашифрування в БД «Files»;
- У другій частині – запис ID зашифрованого файлу, псевдорандомний пароль, ключ шифрування та IV до БД «Encrypt»;

Після закінчення процесу шифрування Користувач може отримати результати шифрування у вигляді списку імен всіх зашифрованих ним файлів, а також здійснити процедуру «Проведення розшифрування», для отримання вихідного (відкритого) файлу. Для цього він використовує дану процедуру, що дозволяє розшифрувати вибраний файл, а також вивести його на екран. Проте,

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змін	Арк.	№ докум.	Підпис	Дата		

для виведення цих результатів, потрібно вибрати зашифрований файл, який потрібно розшифрувати. Це здійснюється за допомогою процедури «Вибір зашифрованого файлу». Остання процедура можлива при умові наявності відповідних даних у БД «Files» та «Encrypt».

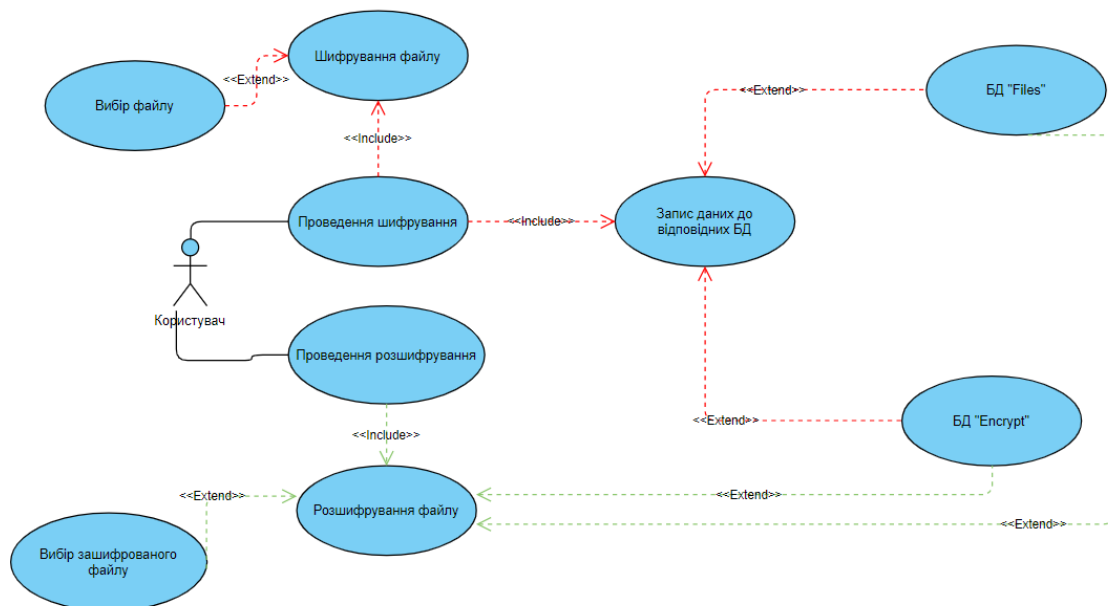


Рисунок 3.1 – Специфікування вимог до програмного компоненту діаграмою варіантів використання

3.2. Створювання логічної структури програмного компоненту

Особливість логічного уявлення полягає в тому, що воно оперує поняттями, які мають в деякому сенсі віртуальний характер. Іншими словами, різні елементи логічного представлення, такі як класи, асоціації, стану та шляхів сполучення, не є матеріальними сутностями. Вони лише відображають розуміння структури або архітектури тієї чи іншої системи або динамічні аспекти її поведінки. Основне призначення логічного представлення полягає в аналізі структурних і функціональних відносин між елементами моделі системи [18].

У загальному випадку поведінка об'єктів, елементів систем або підсистем може розглядатися в кількох аспектах, наслідком чого є наявність декількох типів діаграм для моделювання поведінки в нотації мови UML. Якщо поведінка

розглядається з точки зору моделювання діяльності в формі послідовності дій, які виконуються різними елементами, що входять до складу даної системи – то використовуються діаграми діяльності [18].

Діаграми діяльності (activity diagram) – це візуальне представлення графу діяльності, що надають графічні уявлення про робочі потоки поетапних дій та дій з підтримкою вибору, ітерації та паралелізму. Тобто, головним аспектом моделювання діяльності в нотації мови UML є послідовність дій і умов їх виконання, а також потік об'єктів, які є необхідними для виконання окремих дій або є результатом їх виконання.

Процес функціонування діаграми діяльності часто описується за допомогою таких ключових понять, як «діяльність» та «дія» [18].

Діяльність (activity) є специфікацією параметризованої поведінки в формі координованої послідовності підлеглих одиниць, індивідуальними елементами яких є дії. Тобто, на діаграмі діяльності поведінка діяльності описується за допомогою моделі потоку управління і потоку даних. В свою чергу потік управління моделюється у вигляді вузлів діяльності, з'єднаними дугами діяльності. Також, окрім вузлів та дуг діяльності використовуються специфічні конструкції для управління, такі як синхронізація, рішення і управління паралельними процесами. Цікавим фактом є те, що діяльності можуть утворювати ієрархії, в яких одні діяльності викликають інші. Елементами кожної діяльності є дії.

Дія (action) являє собою елементарну одиницю специфікації поведінки, над якою не може бути проведена подальша декомпозиція в формі діяльності. Тобто, дія – це іменований елемент, якому відповідав би один крок або етап при виконанні деякої діяльності. Процес виконання будь-якої дії в діаграмі діяльності полягає в перетворенні та/або обробці даних, що циркулюють в системі, що моделюється. Це відбувається за рахунок прийому початкових даних з входу системи та видача результату на виході системи. Також, в результаті виконання дії може змінитися стан системи. При цьому кожна дія може виконуватися будь-яку кількість разів.

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змін	Арк.	№ докум.	Підпис	Дата		

Концепція діяльності в нотації мови UML може бути застосована до моделювання процесів і потоків робіт. Діяльності можуть також використовуватися при моделюванні інформаційних систем для специфікації процесів системного рівня. Стрілки ведуть від старту до кінця і позначають порядок в якому відбуваються активності.

Основними графічними елементами діаграми діяльності є [18]:

- округлені прямокутники, що позначають певну дію;
- ромби, що позначають рішення (умову вибору);
- чорні прямокутники, що позначають (розподіл) чи кінець (об'єднання) паралельних активностей;
- чорний кружок позначає старт (початковий стан) процесу;
- чорний кружок в колі позначає кінець (кінцевий стан).

Таким чином, логічна структура програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи систему відображена діаграмою діяльності (див. рис. 3.2) [18]. На даній діаграмі початок діяльності починається з Користувача, який своїми діями здійснює процес «Запуск програми» – запуск програмного компоненту підсистеми захисту. Далі, Користувач, здійснює процес «Вибір файлу шифрування» – процедура вибору користувачем файлу, який потрібно зашифрувати та передати до відповідних БД. Вибір файлу здійснюється за допомогою файлового менеджера ОС.

сканування IP-адресів локальної комп'ютерної мережі, з метою отримання IP-адресу кожної комп'ютерної системи, які є елементами даної мережі. Далі, користувач стикається з розгалуженням, а саме – з умовою вибору – для продовження роботи програмного компоненту потрібно вибрати файл. Якщо дана умова не виконана, то робота програмний компонент призупиняється. При виборі файлу – відбувається виконання процесу «Шифрування файлу» - шифрування файлу за допомогою симетричного блокового алгоритму AES-128. Інформація про успішне шифрування передається користувачу у вигляді списку усіх зашифрованих ним файлів, а дані шифрування передаються до БД «Files»

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Змін	Арк.	№ докум.	Підпис	Дата		

та «Encrypt». За ці дії відповідають процеси «Отримання зашифрованого файлу» та «Передача даних шифрування в БД «Files» та «Encrypt» відповідно.

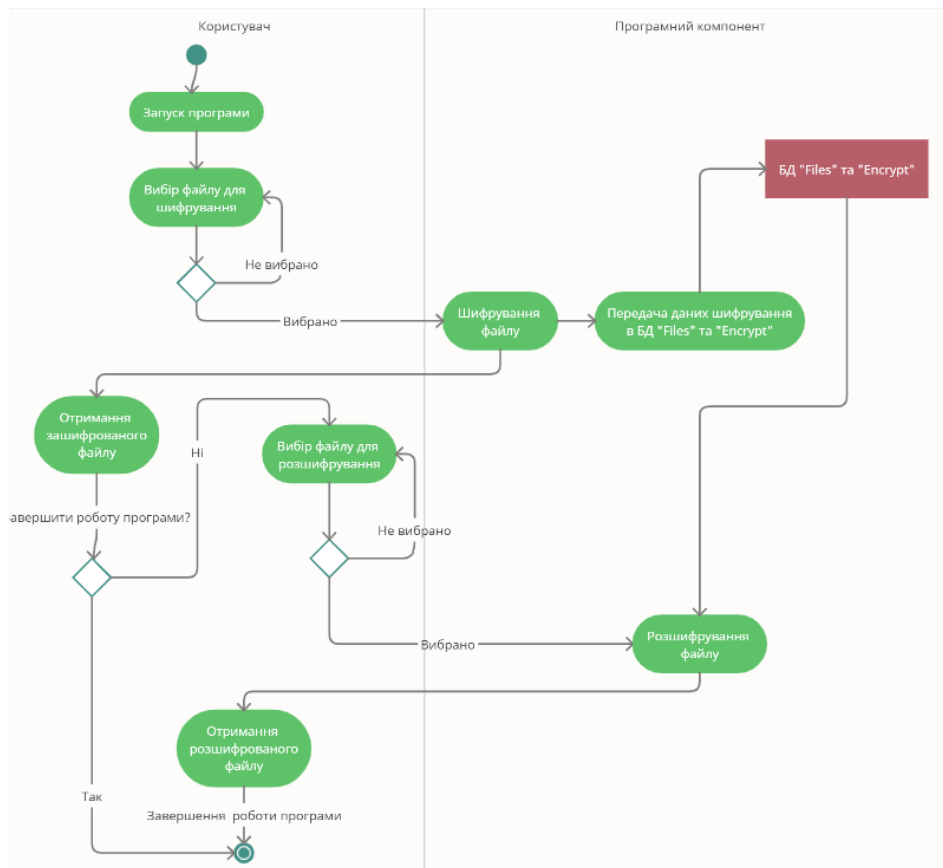


Рисунок 3.2 – Логічна структура програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи

Далі, користувач знову стикається з розгалуженням. Дане розгалуження відповідає за можливість дострокового завершення програмного компоненту.

Наступними діями користувача, при продовженні роботи програми, є – вибір файлу для розшифрування. Дані дії здійснюються за допомогою процесу «Вибір IP-адреси для сканування». Користувач знову стикається з розгалуженням, а саме – з умовою вибору – для продовження роботи програмного компоненту потрібно вибрати файл. Якщо дана умова не виконана, то робота програмний компонент призупиняється.

Далі, на стороні програмного компоненту, здійснюється процедура «Розшифрування файлу» - здійснення розшифрування файлу за допомогою симетричного блокового алгоритму AES-128. Розшифрування здійснюється лише за умови наявності зашифрованого файлу, а також інших даних

шифрування, у БД «Files» та «Encrypt». Як результат, розшифрований файл передається користувачу, в процедуру «Отримання розшифрованого файлу». За допомогою останньої процедури відбувається відкриття розшифрованого файлу відповідним програмним забезпеченням.

3.3 Створювання фізичної структури програмного компоненту

Фізичне моделювання в нотації мови UML призначене для деталізації програмних компонентів і взаємозв'язків між ними, а також артефактів, які їх реалізують, і вузлів, на яких вони розгортаються і виконуються. Для моделювання фізичної структури зазвичай відповідаються діаграми компонентів та діаграми розгортання.

3.3.1 Створювання компонентів і зв'язків між ними

Одним з основних понять фізичного моделювання є компонент, під яким розуміється модульна одиниця програми з можливими інтерфейсами. Для графічного представлення компонентів і взаємозв'язків між ними призначена діаграма компонентів.

Діаграма компонентів (Component diagram) відображає залежності між компонентами програмного забезпечення, включаючи компоненти вихідних кодів, бінарні компоненти, та компоненти, що можуть виконуватись [18]. Модуль програмного забезпечення може бути представлено як компоненту. Одні компоненти можуть існувати тільки на етапі компіляції програмного коду, деякі – під час компонування, а інші – на етапі його виконання.

Мета створення діаграми компонентів полягає в досягненні наступних цілей [18]:

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змн	Арк	№ докум	Підпис	Дата		

- специфікація варіанту програмної системи, що буде виконуватися;
- візуалізація загальної структури вихідного коду програмної системи;
- забезпечення багаторазового використання окремих фрагментів програмного коду;
- представлення концептуальної і фізичної схем баз даних.

Діаграма компонентів забезпечує узгоджений перехід від логічного представлення до конкретної реалізації проекту у формі програмного коду. Діаграма компонентів відображає загальні залежності між компонентами, розглядаючи останні як класифікатори.

Основним елементом діаграми компонентів виступає компонент. Компонент (component) – елемент моделі, що представляє деяку модульну частина системи з інкапсульованим вмістом, специфікація якого є взаємозаміною в його оточенні [18]. У нотації мови UML в якості компонента може розглядатися будь-який автономний елемент системи або підсистеми.

Головною особливістю компонентів в моделі програмної системи – можливість їх повторного використання в різних проектах, замінюючи і пов'язуючи їх разом за допомогою наданих і необхідних інтерфейсів. Як результат – можливість використання гнучкості при створенні реальних проектів та скорочення термінів їх розробки. А артефакти, які реалізують компоненти, призначені для початкового і повторного розгортання при їх реалізації з використанням найбільш сучасних програмних технологій незалежно, наприклад, від модифікації існуючої системи.

Основними графічними елементами діаграми компонентів, крім компонентів, виступають [18]:

- інтерфейси, що з'єднують між собою компоненти;
- порти, які входять до складу компонента;
- відношення між ними.

За допомогою діаграми компонентів, що моделює фізичну структуру компоненту, описано послідовність компонентів, що взаємодіють між собою в програмному засобі, що проектується (див. рис. 3.3) [18].

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

На даній діаграмі бачимо персональний комп'ютер користувача. З даного пристрою відбувається запуск програмного компоненту. Спочатку, користувач взаємодіє з графічним інтерфейсом програми, за який відповідають файли «myForm.aspx» та «myForm.aspx.cs». Всі дії, що здійснює користувач в даній програмі, виконуються за допомогою функціоналу, який описаний в наступних файлах:

- «EncryptWorker» – основний файл (клас), що відповідає за шифрування та розшифрування файлів;
- «EncryptedFile» – файл (клас), що відповідає за запис (збереження) зашифрованих байтів файлу, псевдорандомного паролю, ключа шифрування та вектору ініціалізації IV, а також їх подальшу передачу в БД «Files» та «Encrypt»;
- «DataManager» – файл (клас), що відповідає за взаємодію з БД «Files» та «Encrypt».

Тобто, файли що відповідають за графічний інтерфейс, а саме – «myForm.aspx» та «myForm.aspx.cs» – взаємодіють з файлами «EncryptWorker», «EncryptedFile» та «DataManager». В свою чергу, останній файл забезпечує взаємодію з БД «Files» та «Encrypt».

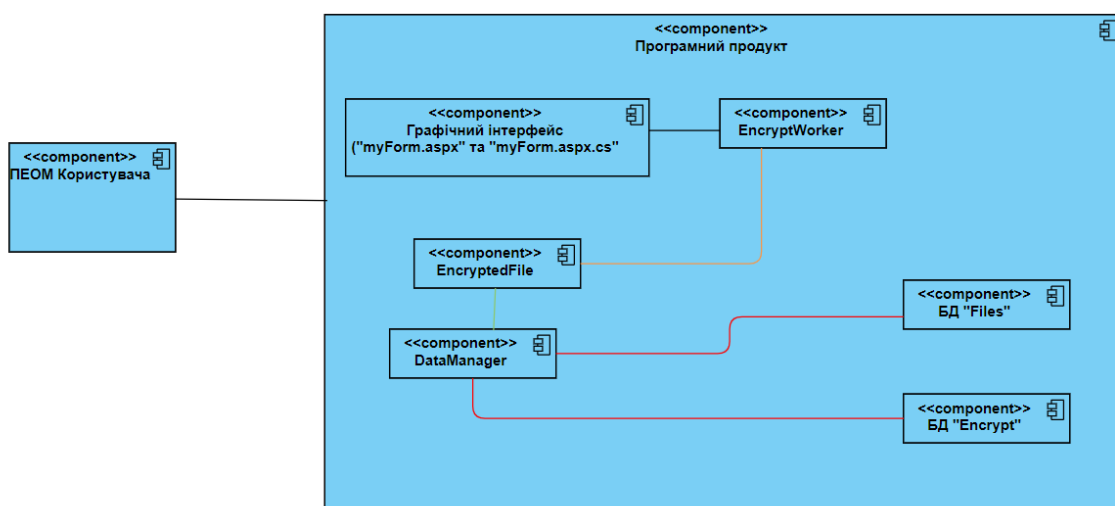


Рисунок 3.3 – Відображення діаграма компонентів програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи

3.3.2 Створювання вузлів і зв'язків між ними

Наступним основним поняттям фізичного моделювання є вузол, на якому розгортається один або кілька компонентів. Для графічного представлення вузлів і взаємозв'язків між ними призначена діаграма розгортання (deployment diagram) [18].

Діаграма розгортання (deployment diagram) призначена для представлення загальної конфігурації або топології розподіленої програмної системи і містить зображення розміщення різних артефактів по окремих вузлах системи [18]. Також, діаграма розгортання показує наявність фізичних з'єднань або маршрутів для передачі інформації між апаратними та програмними пристроями, які забезпечують функціонування системи в різних режимах.

Мета створення діаграми компонентів полягає в досягненні наступних цілей [18]:

- уточнення фізичних вузлів, необхідних для розміщення на них компонентів програмної системи, що виконують в процесі функціонування даної системи;
- наочне зображення фізичних зв'язків між вузлами реалізації системи на етапі її виконання;
- виявлення вузьких місць системи і реконфігурація її топології для досягнення необхідної продуктивності.

Діаграма розгортання здійснює візуалізацію тільки тих елементів фізичного представлення моделі, які існують під час виконання програмної системи. Такими елементами можуть бути конкретні артефакти, такі як виконані файли і бібліотеки динамічного компонування.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Змін	Арк.	№ докум	Підпис	Дата		

Діаграма розгортання може містити графічні зображення процесорів, периферійних пристроїв, артефактів і різних відносин між ними. На відміну від діаграм логічного представлення, діаграма розгортання є, як правило, єдиною для розроблюваної системи, оскільки має цілком відбивати особливості її топології та реалізації. У нотації мови UML створена спрощена модель розгортання, яка виявляється достатньою для більшості практичних застосувань.

Основними графічними елементами діаграми розгортання є [18]:

1. Вузол (node) – це елементом моделі, який представляє певний обчислювальний ресурс для розгортання на ньому різних артефактів. В свою чергу вузол складається з середовища виконання та пристрою:

1.1. Середовище виконання (execution environment) являє собою вузол, який володіє функціональністю, необхідної для практичного виконання розгорнутих на ньому здійснених артефактів. Воно моделює програмно-апаратну платформу, яка надає набір стандартних послуг, які можуть використовуватися здійсненими компонентами додатків.

1.2. Пристрій (device) являє собою вузол, який володіє деяким загальним обчислювальним ресурсом зі здатністю обробляти розгорнуті на ньому артефакти. В якості обчислювального ресурсу вузла розглядається наявність одного або декількох процесорів і деякого об'єму оперативної пам'яті.

2. Артефакт (artifact) являє собою елемент моделі, який специфікує деяку фізично існуючу частину інформації, яка використовується або вироблену в ході розробки програмного забезпечення або в процесі розгортання і функціонування системи.

3. Специфікація розгортання (deployment specification) описує безліч властивостей, які визначають параметри виконання артефакту компонента, що розгортається на деякому вузлі.

4. Відношення на діаграмі розгортання.

Фізичну структуру програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи, що відображена діаграмою

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Змі	Арк	№ докум	Підпис	Дата		

розгортання, представлено на рис. 3.4 [18]. Дана діаграма пояснює фізичну та логічну взаємодію між артефактами (кодом) даного компоненту.

Весь графічний інтерфейс відбувається в двох файлах – «myForm.aspx» та «myForm.aspx.cs». Також є файли (класи) «EncryptWorker», «EncryptedFile» та «DataManager». В них знаходиться програмний код, що відповідає за взаємодію функціональної частини програмного компоненту

Також на даній діаграмі відображені дві БД:

- «Files» – в дану БД записуються ID зашифрованого файлу, назва файлу, зашифровані байти (зашифрований файл) та дата зашифрування;
- «Encrypt» – в дану БД записуються ID зашифрованого файлу, псевдорандомний пароль, ключ шифрування та вектор ініціалізації IV.

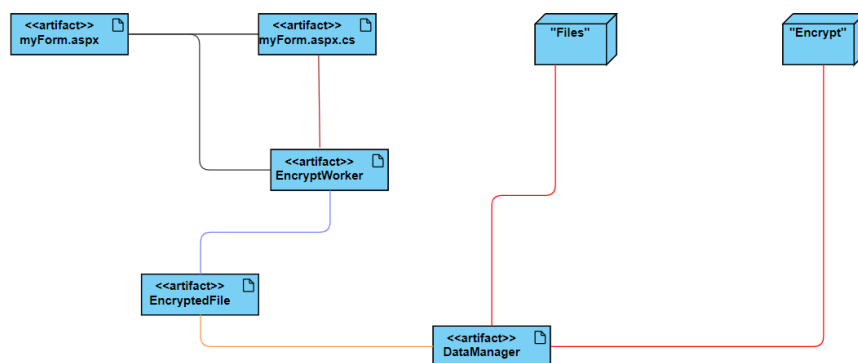


Рисунок 3.4 – Відображення фізичної структури програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи

Висновки за розділом 3

Створено структуру програмного компоненту підсистеми захисту бази даних Web-орієнтованої інформаційної системи на основі формалізування його роботи на рівні функцій, процесів і потоків даних, використання якої дозволяє забезпечити його функціональну придатність.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Змін	Арк.	№ докум.	Підпис	Дата		

Зокрема, специфіковано вимоги до програмного продукту, які описані та промодельовані за допомогою діаграми варіантів використання. Також було створено логічну та фізичні структури програмного компоненту. Логічна структура описана та промодельована за допомогою діаграми діяльності. Тоді як фізична структура враховує:

- компоненти і зв'язки між ними. Описано та промодельовано за допомогою діаграми компонентів.
- вузли і зв'язки між ними. Описано та промодельовано за допомогою діаграми розгортання.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змін	Арк.	№ докум	Підпис	Дата		

РОЗДІЛ 4. ПРОГРАМНА РЕАЛІЗАЦІЯ КОМПОНЕНТУ ПІДСИСТЕМИ ЗАХИСТУ БАЗИ ДАНИХ WEB-ОРІЄНТОВАНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

4.1 Реалізування програмного компоненту

Для створення програмного компоненту підсистеми захисту бази даних web-орієнтованої інформаційної системи було вибрано мову програмування об'єктно-орієнтованого типу C#. Дану мову програмування було вибрано виходячи з умови поставленого завдання та функціональних вимог до програмного компоненту [3, 18, 19].

C# (вимовляється Сі-шарп) — об'єктно-орієнтована мова програмування високого рівня з безпечною системою типізації для платформи .NET [19]. На вибір даної мови програмування також вплинуло те, що «мова C#» – це ключ до сучасного об'єктно-орієнтованого програмування. C# - це легка та зрозуміла мова, що створена для розробки високопродуктивного програмного забезпечення і надзвичайно популярна серед програмістів [19]. Головна перевага даної мови програмування в тому, що вона кросплатформна – тобто за її допомогою можливо створювати програмні продукти не тільки під платформу Windows, а й під Linux, Mac OS, Android, iOS. Дана мова забезпечує концептуальний фундамент, на який спираються інші мови програмування і багато сучасних засобів обробки даних. Мову використовують розробки програмного забезпечення для бізнесу, написання потужних серверних та клієнтських програм, а також для розробки відеоігор та мобільних програм.

Також однією з головних причин вибору даної мови програмування вплинуло те, що вона містить засоби розробки програм для широкого спектру задач, від низькорівневих утиліт і драйверів до вельми складних програмних комплексів.

Перевагами, що вплинули на вибір мови C#, є [19]:

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Змі	Арк	№ докум	Підпис	Дата		

1. Доступність. Для C # існує величезна кількість навчальної літератури, що перекладена на різні мови світу. Мова має низький поріг входження та з легкістю підійде новачкам, які тільки починають опановувати процес розробки програмного забезпечення.

2. Простота в використанні. C# – це об'єктно-орієнтована, проста і водночас потужна мова програмування, яка дозволяє розробникам створювати багатофункціональні програми.

3. Сучасна мова програмування. C# – це мова компілюваного типу, тому вона має всі переваги таких мов. Також ця мова поєднує найкращі ідеї сучасних мов програмування – таких як Java, C++, Visual Basic.

4. Швидкість та якість розробки програмного забезпечення. Через велику різноманітність синтаксичних конструкцій та можливості працювати з платформою .Net, C# дозволяє швидше, ніж на будь-якій іншій мові, розробляти програмні рішення.

5. Кросплатформеність. Мову C# можливо використовувати для розробки програмного забезпечення для платформ Windows, Linux, Mac OS, Android, iOS.

6. Обчислювальна продуктивність. Мова спроектована таким чином, щоб дати програмісту максимальний контроль над усіма аспектами структури і порядку виконання програми.

7. Надійність та елегантність. Елегантність C# досягається за рахунок великої різноманітності синтаксичних конструкцій. А велику надійності було досягнуто через роботу CLR машини, адже на відміну від інших компілятор CLR запускає розроблений додаток на віртуальному процесорі. Тому у разі виникнення будь-яких помилок, це ніяк не вплине на роботу інших програм у системі. Проте це також означає, що для запуску програми потрібен додатковий час. Відповідно програми написані мовою програмування C# надійніші, але менш швидкі (ніж такі ж програми написані на C++).

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

Також, окрім мови програмування, здійснювався вибір середовища розробки програми, операційної системи на базі якої вона розроблялася та технологій її створення. Також, при розробці даного програмного компоненту, потрібно було вибрати надійну та практичну СУБД.

Операційна система. Програмний компонент підсистеми захисту бази даних web-орієнтованої інформаційної системи створювався на базі ОС сімейства Windows, а саме – ОС Windows 10. Даний вибір пов'язаний з тим, що дане сімейство ОС частково створене за допомогою мови програмування C#, а також виходячи з того, що більшість середовищ створення програмних продуктів мовою C# розроблені лише для даного сімейства ОС.

Середовище розробки. Середовищем для розробки програмного компоненту тестування на проникнення в комп'ютерну систему було вибрано Microsoft Visual Studio 2017.

Технології програмування. Технологією програмування, що використовувалася в процесі створення програмного компоненту, була ASP.NET, що надає можливість розробляти вебзастосунки і вебсервіси для Microsoft Windows на мові C#.

СУБД. При розробці даного програмного компоненту було використано Microsoft SQL Server 2012 (коротко – MS SQL Server 2012) – потужна та надійна безкоштовна система управління даними, що забезпечує функціональне та надійне сховище даних для веб-сайтів та настільних додатків [20].

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

4.2 Тестування програмного компоненту

4.2.1 Аналіз якості програмного забезпечення

Для аналізу якості будь-якого програмного застосунку, а також створеного програмного компоненту підсистеми захисту бази даних web-орієнтованої інформаційної системи, необхідно визначити характеристики та підхарактеристики, яким даний програмний продукт повинен відповідати [21].

Існують чотири основні характеристики та їх підхарактеристики:

1. Функціональність – властивості, визначають спроможність ПЗ виконувати в заданому середовищі упорядковану послідовність дій для задоволення споживчих властивостей, замовлених користувачем, відповідно до вимог обробки і загальносистемних засобів. Атрибути функціональності програмного забезпечення (ПЗ):

1.1. Функціональна повнота – атрибут, який показує ступінь достатності основних функцій для вирішення завдань відповідно до призначення програмного забезпечення.

1.2. Правильність – атрибут, який показує, як забезпечується досягнення правильних та погоджених результатів.

1.3. Захищеність – атрибути, які вказують на можливість запобігати несанкціонованому доступу до програм і даних.

2. Зручність застосування – це множина атрибутів, що характеризують умови взаємодії користувача з ПЗ. Атрибути зручності застосування ПЗ:

2.1. Зрозумілість – визначається, наскільки зрозумілі для розпізнавання логічні концепції ПЗ та умов їх застосування;

2.2. Легкість навчання – визначається, наскільки доступні (легкі) для вивчення умови використання.

3. Ефективність – характеризується ступенем відповідності використовуваних ресурсів середовища до функціонування рівня якості.

4. Час реакції – час відгуку, опрацювання, виконання функцій.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змн	Арк	№ докум	Підпис	Дата		

У табл. 4.1 наведено конфліктуючі між собою підхарактеристики з визначених. Позначка «+» – означає, що підхарактеристики конфліктують між собою, позначка «-» - підхарактеристики не конфліктують між собою.

Таблиця 4.1 – Результат попарного зіставлення підхарактеристик, за якими тестується програмний компонент

Характеристика		Функціональність			Зручність застосування		Ефективність
		Функціональна повнота	Правильність	Захищеність	Зрозумілість	Легкість навчання	Час реакції
Функціональність	Функціональна повнота	-	-	-	-	-	0,3
	Правильність	+	-	-	-	-	0,5
	Захищеність	+	-	-	+	+	0,5
Зручність	Зрозумілість	+	+	+	-	-	0,9
	Легкість навчання	+	-	+	-	-	0,8
Ефективність	Час реакції	0,3	0,5	0,5	0,9	0,8	0,7

У таблиці 4.2. наведено метрики та вагові коефіцієнти для означених характеристик.

Таблиця 4.2 – Метрики та вагові коефіцієнти характеристик

Характеристика	Підхарактеристика	Метрика	Вага
Функціональність	Функціональна повнота	Число реалізованих функцій від загального числа заявлених функцій	0.8
	Правильність	Число результатів від загального числа очікуваних результатів	0.8
	Захищеність	Число захищеності програмного забезпечення	0.5
Зручність застосування	Зрозумілість	Число зрозумілості використання програмного забезпечення	1
	Легкість навчання	Число легкості навчання програмного забезпечення	0.9
Ефективність	Час реакції	Число часу відгуку програми на дії користувача	0,7

Також було визначено наступні характеристики експлуатаційної якості:

- **результативність** – використовується, тому що є важливим оцінка якості виконання програмою своїх безпосередніх функцій та отримання результату, що за своєю точністю задовольнить користувача;

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змн	Арк	№ докум	Підпис	Дата		

- **продуктивність** – ступінь, в якій витрачаються ресурси на досягнення користувачем заданої ефективності вирішення завдань у встановленому контексті використання програмного забезпечення;
- **задоволеність** – визначає ступінь, в якій користувач задоволений програмним забезпеченням в межах його використання.

Детальніше взаємозв'язок характеристик зовнішньої та експлуатаційної якості наведено на рис. 4.1

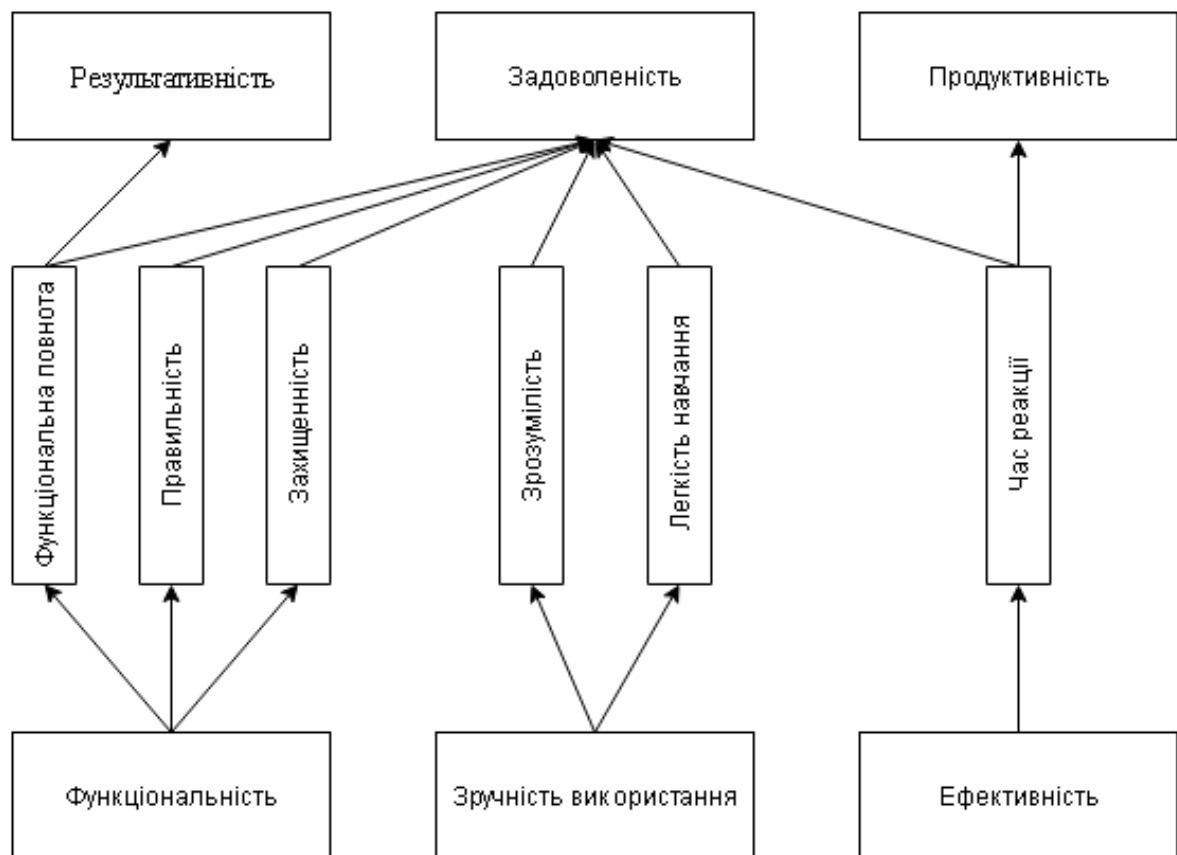


Рисунок 4.1 – Взаємозв'язок зовнішньої та експлуатаційної якості

4.2.2. Опис процесів тестування

Тестування програмного забезпечення – процес перевірки відповідності заявлених до продукту вимог і реально реалізованої функціональності, здійснений шляхом спостереження за його роботою в штучно створених ситуаціях і на обмеженому наборі тестів. Таким чином тестування є своєрідним методом контролю якості тих характеристик ПЗ, що проявляються в процесі його роботи [21]. Також тестування надає наочну оцінку системи для того, щоб знайти відмінності між тим, яким програмне забезпечення повинно бути і якою воно є. Тестування поділяється на велику кількість різних видів. Для перевірки невідповідності між реальною поведінкою функцій, що були реалізовані в рамках ПЗ, і очікуваною поведінкою, визначеною вимогами до ПЗ використовується функціональне тестування.

Під час тестування програмного компоненту системи захисту інформації бази даних Web-орієнтованої інформаційної системи необхідно перевірити функціональну відповідність наступних функцій [21]:

- належну та правильну процедуру роботи підсистеми захисту бази даних web-орієнтованої інформаційної системи, а саме реалізація системи для шифрування та дешифрування;
- перевірка правильної роботи функції попереднього вибору файлу для шифрування та розшифрування;
- збереження інформації про результати шифрування файлу в БД «Files», де зберігаються байти зашифрованого файлу, а також запис пароля до зашифрованого файлу, ключа шифрування та вектора ініціалізації (IV) до БД «Encrypt».

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змн	Арк	№ докум	Підпис	Дата		

Застосування буде вважатися таким, що пройшло тестування, якщо:

- результати виконання функцій будуть відповідати очікуваним;
- дані у сформованих файлах коректні та коректно записані у відповідні БД.

4.2.3. Опис контрольного прикладу

При тестуванні програмного компоненту підсистеми захисту бази даних web-орієнтованої інформаційної системи було перевірено його функціональність. В табл. 4.3 наведено інформацію про тестування основних варіантів використання та їх результати зображені на рис. 4.3.

Таблиця 4.3 – Перевірка запису інформації

Мета тесту	Перевірка можливості належного шифрування та розшифрування файлів та передача даних шифрування у відповідні БД
Початковий стан	Відкрита програма підсистеми захисту бази даних web-орієнтованої інформаційної системи
Вхідні дані	Відкритий файл, над яким буде здійснюватися процедура шифрування, а також подальшого розшифрування
Схема проведення тесту	Запустити функцію шифрування. Після отримання зашифрованого файлу – запустити функцію розшифрування
Очікуваний результат	Отримання зашифрованого файлу, а також, в процесі розшифрування, отримання відкритого (розшифрованого) файл
Стан програмного продукту після проведення випробувань	Список файлів, що над якими здійснювався процес шифрування, успішно отримано. Над будь-яким файлом, з даного списку, можливо здійснити операцію розшифрування з метою отримання відкритого тексту

4.3. Використання програмного компоненту

Створений програмний компонент підсистеми захисту бази даних web-орієнтованої інформаційної системи – це веб-застосунок (у вигляді веб-сайту) в якому клієнтом виступає веб-браузер, а сервером – локальний сервер IIS Express, що вбудований в Microsoft Visual Studio). Даний програмний компонент написаний мовою програмування C#, за допомогою технології ASP.NET, з використання середовища розробки Microsoft Visual Studio 2017 та СУБД Microsoft SQL Server 2012 на базі ОС Windows 10. Графічний вигляд даного додатку зображено на рис. 4.2.

Як видно з рис. 4.2 розроблений компонент складається з двох функціональних частин. Перша функціональна частина – це вибір файлу з файлової системи ОС Windows, шифрування даного файлу та передача назви (імені) зашифрованого файлу в поле зі списком зашифрованих файлів.

Друга функціональна частина – це вибір зашифрованого файлу, з поля зі списком зашифрованих файлів, та його розшифрування з метою отримання вихідного відкритого тексту.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Пашаріна І.В.				
Змін	Арк.	№ докум.	Підпис	Дата		

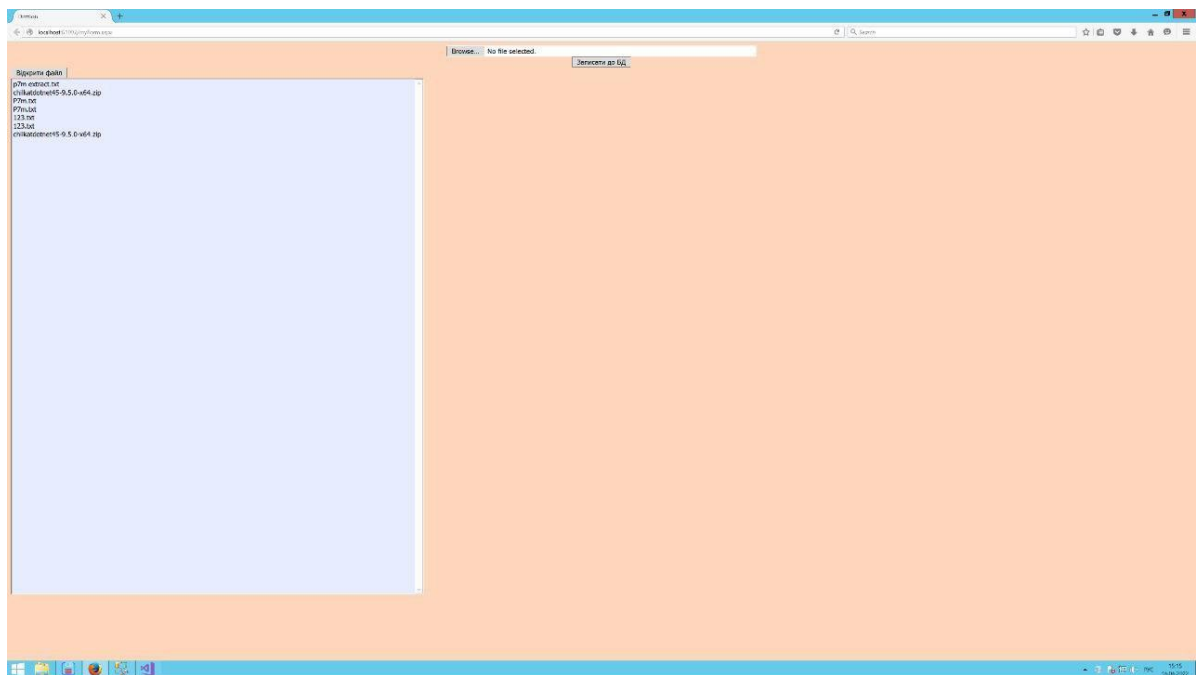


Рисунок 4.2 – Графічний вигляд програмного компоненту

Робота даного програмного компоненту починається з натискання клавіші «Browse...», що розташована зверху посередині під пошуковим полем (поле з URL-адресою). Після натискання даної клавіші відбувається відкриття файлового менеджера. Даний процес відображено на рис. 4.3.

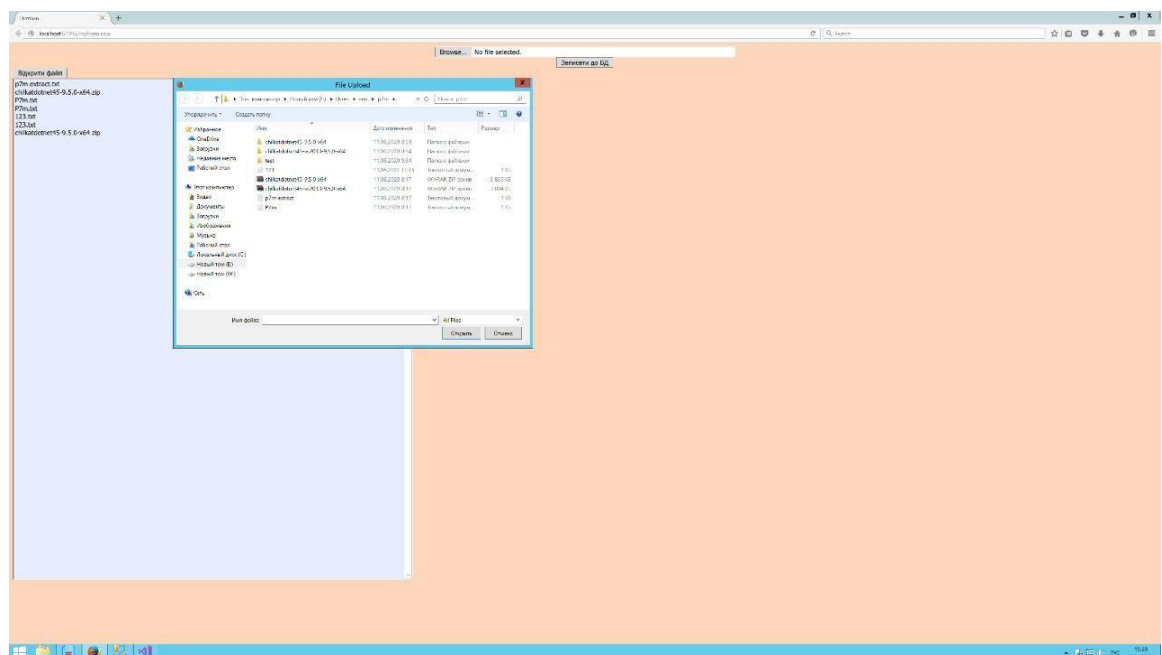


Рисунок 4.3 – Вибір файлу для подальшого шифрування

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Панаріна І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		

В файловому менеджері потрібно знайти потрібний користувачу файл (для подальшого шифрування) та натиснути клавішу «Відкрити», що знаходиться у файловому менеджері. Після даної процедури в текстовому полі, що знаходиться праворуч від клавіші «Browse...», буде відображено вибраний користувачем файл. Даний процес відображено на рис. 4.4.

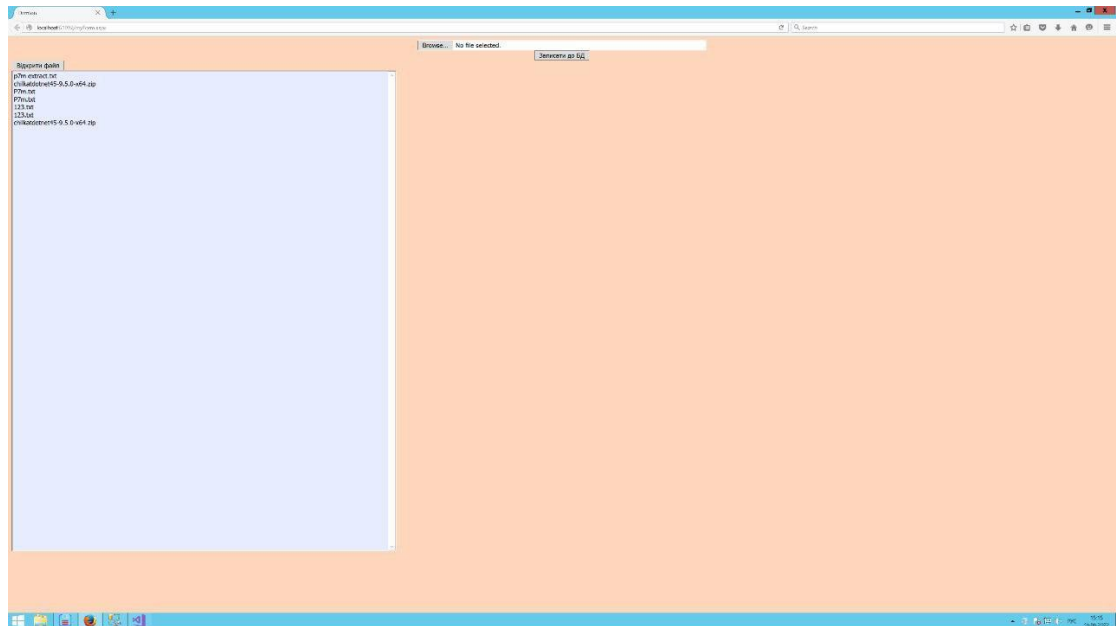


Рисунок 4.4 – Відображення вибраного користувачем файлу

Далі потрібно натиснути клавішу «Записати до БД» і при натисканні даної клавіші спрацює процедура шифрування файла. Тобто, буде виконано наступне:

- 1) Програмний компонент, за допомогою класу «EncryptWorker», генерує псевдорандомний пароль довжиною 14 символів;
- 2) Далі, з даного пароля формується ключ шифрування та вектор ініціалізації IV, за допомогою класу «EncryptWorker». Ініціалізаційний вектор (англ. initialization vector, далі – IV) — це блок біт, який використовують декілька режимів для у випадковлення шифрування і, таким чином, видають різні шифротексти навіть на одному відкритому тексті зашифрованому декілька разів, без потреби в повільнішому процесі отримання нового ключа [22];

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змн	Арк	№ докум	Підпис	Дата		

- 3) Для подальшого шифрування використовується ключ шифрування та вектор ініціалізації IV. Тобто, за допомогою класу «EncryptWorker», відбувається шифрування – байти відкритого файлу (байтовий потік) буде зашифровано за допомогою алгоритму AES-128. Це симетричний алгоритм блокового шифрування з розміром блоку для шифрування 128 біт та ключом шифрування 128 біт. Для цього з даного пароля формується ключ шифрування. Якщо коротко – то байти тексту (байтовий потік) розбивається на блоки розміром 128 біт та шифруються за допомогою ключа шифрування та IV.
- 4) Далі відбувається запис (збереження) зашифрованих байтів файлу, псевдорандомного паролю, ключа шифрування та IV в класі екземплярі класу «EncryptedFile».
- 5) Після цього, за допомогою класу «DataManager», відбувається робота с БД. Тобто, клас робить два SQL-запити до класу «EncryptedFile» з метою запису відповідних даних до відповідних БД. Перший запит – записує ID зашифрованого файлу, назву файлу, зашифровані байти (зашифрований файл) та дату зашифрування в БД «Files». Другий запит – записує ID зашифрованого файлу, псевдорандомний пароль, ключ шифрування та IV до БД «Encrypt»;
- 6) Як результат – вибраний файл зашифрований, дані шифрування записані в відповідні БД. А також відбувається відображення зашифрованого файлу в інтерфейсі програмного компоненту (в полі відображення списку зашифрованих файлів). За всі візуальні відображення відповідає клас «myForm». Даний процес, що описаний в пунктах 1-6, відображено на рис. 4.5.

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змн	Арк	№ докум	Підпис	Дата		

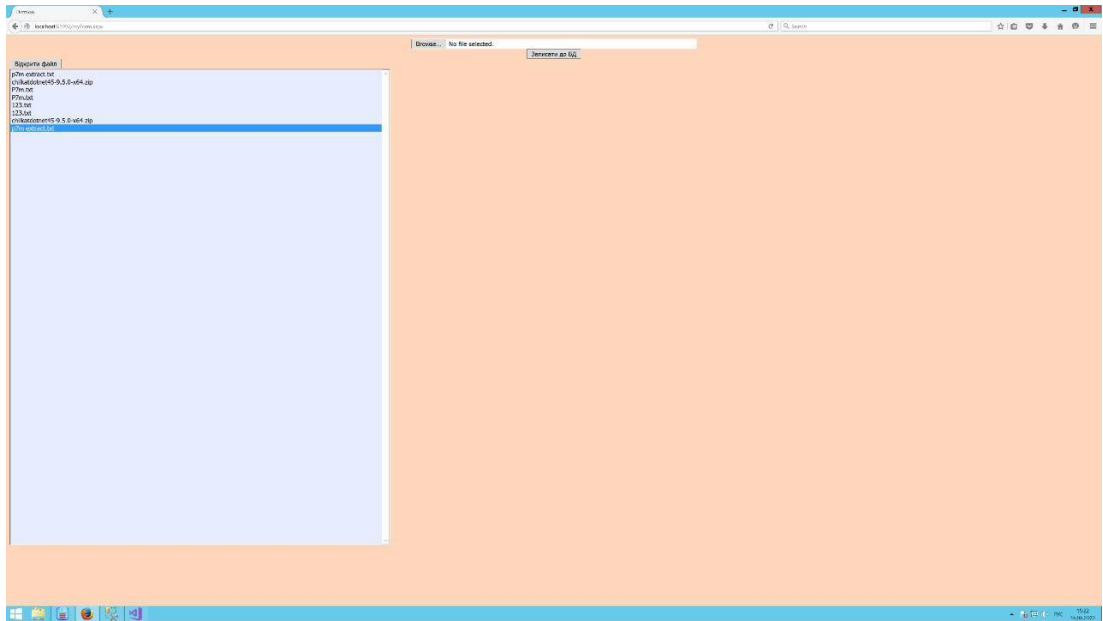


Рисунок 4.5 – Відображення зашифрованого файлу в полі відображення списку зашифрованих файлів

Далі розглянемо другу функціональну частину програмного компоненту – це вибір зашифрованого файлу, з поля зі списком зашифрованих файлів, та його розшифрування з метою отримання вихідного відкритого тексту. Для здійснення цієї процедури вибираємо зашифрований файл (в полі відображення списку зашифрованих файлів), який бажаємо розшифрувати та натискаємо клавішу «Відкрити файл». При натисканні даної клавіші спрацює процедура розшифрування файла. Тобто, буде виконано наступне:

- 1) За допомогою класу «DataManager» здійснюється запит до двох БД. Перший запит – до БД «Files». За допомогою цього запиту здійснюється пошук файлу по його назві (імені) в даній БД. Далі, відбувається співставлення даної назви (імені) з зашифрованими байтами файлу і після чого зашифровані байти, у вигляді SQL-запиту, передаються в екземпляр класу «EncryptedFile». Другий запит – до БД «Encrypt». За допомогою цього SQL-запиту, по ID файлу, здійснюється пошук псевдорандомного паролю, ключа шифрування та IV в даній БД. Отримані дані записуються в екземпляр класу «EncryptedFile».

- 2) Далі, отримані та записані дані, з екземпляру класу «EncryptedFile» передаються в клас «EncryptWorker», а вже в останньому класі відбувається процедура розшифрування файлу (за допомогою псевдорандомного паролю, ключа шифрування та IV);
- 3) Розшифрований файл відкривається відповідним програмним забезпеченням і відображається користувачу. Наприклад, текстовим редактором «Блокнот», якщо це файл з розширенням .txt. За всі візуальні відображення відповідає клас «myForm». Даний процес, що описаний в пунктах 1-3, відображено на рис. 4.6.

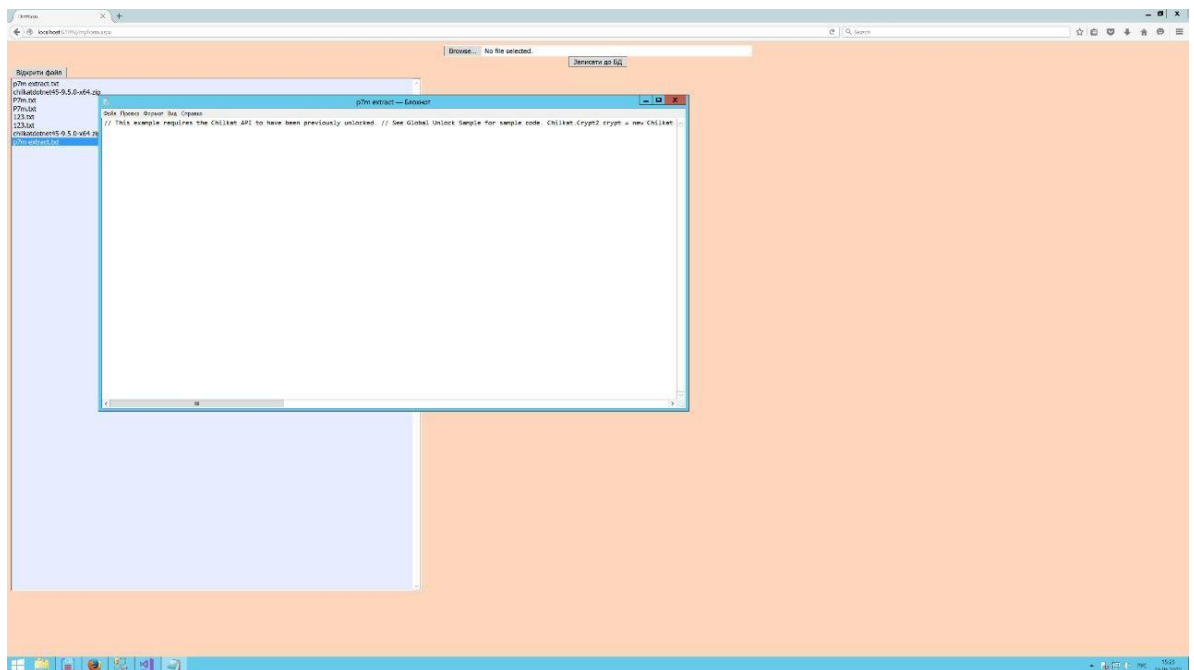


Рисунок 4.6 – Відображення та відкриття розшифрованого (відкритого вихідного) файлу

Завершення роботи програми здійснюється при закритті браузеру або даної вкладки з програмним компонентом (див. рис. 4.7).

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Змін	Арк.	№ докум	Підпис	Дата		

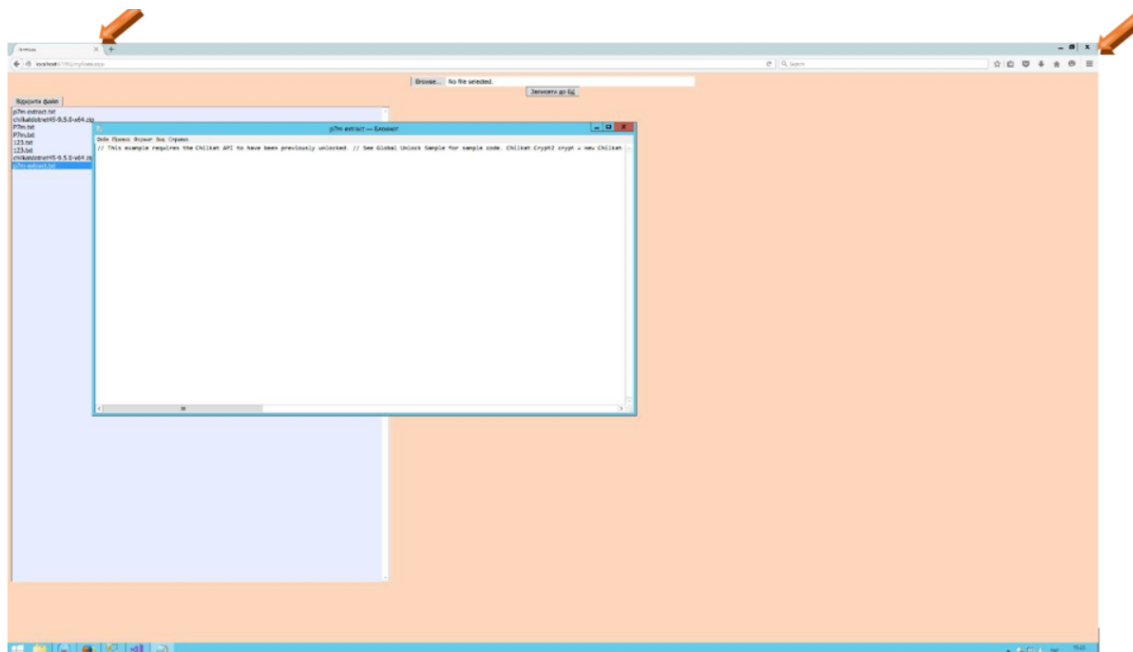


Рисунок 4.7 – Завершення роботи програмного компоненту

Висновки за розділом 4

Розроблено програмний компонент підсистеми захисту бази даних web-орієнтованої інформаційної системи, використання якого дозволяє зашифровувати файли та надійно їх зберігати, а також надає змогу отримувати до даних файлів шляхом їх розшифрування.

Зокрема, описано процес реалізування програмного компоненту, що складався з вибору мови програмування, середовища створення програмного компоненту та СУБД, операційної системи, на базі якої створювався компоненту, а також технології його створення. Також в даному розділі проведено тестування програмного компоненту, а також описано використані при цьому методики.

На завершальному етапі описано використання програмного компоненту, що супроводжувався графічними прикладами у вигляді ілюстративного матеріалу виконання програми.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змін	Арк.	№ докум	Підпис	Дата		

ВИСНОВКИ

Вирішено актуальне завдання моделювання програмного компоненту підсистеми захисту бази даних web-орієнтованої інформаційної системи для його програмної реалізації. Це стало можливим завдяки отриманню таких окремих результатів:

1. Проаналізовано процес захисту на основі використання процедур шифрування на основі симетричного блокового алгоритму AES-128. З огляду на це розроблено вимоги до програмного компоненту підсистеми захисту бази даних web-орієнтованої інформаційної системи.

2. Створено концептуальну модель програмного компоненту підсистеми захисту бази даних web-орієнтованої інформаційної системи, використання якої дозволяє формалізувати його застосування на рівні функцій, процесів і потоків даних, а також сформулювати та обґрунтувати варіанти використання програмного компоненту.

3. Створено структуру програмного компоненту підсистеми захисту бази даних web-орієнтованої інформаційної системи на основі формалізування

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змі	Арк.	№ докум	Підпис	Дата		

його роботи на рівні функцій, процесів і потоків даних, використання якої дозволяє забезпечити їх функціональну придатність.

4. Розроблено програмний компонент підсистеми захисту бази даних web-орієнтованої інформаційної системи на основі його структури, використання якого дозволяє захистити файли при передачі їх до відповідних БД шляхом шифрування, а також отримання відкритих файлів за допомогою процедури розшифрування.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Методи та засоби мультимедійних інформаційних систем: навч. посіб. / Т. М. Басюк, П. І. Жежнич; Нац. ун-т «Львів. політехніка». — Львів: Вид-во Львів. політехніки, 2015. — 426 с. — Бібліогр.: с. 413—416.
2. Грицунов О. В. Інформаційні системи та технології. Навчальний посібник. — Х.: ХНАМГ, 2010. — 222 с.
3. Гайдамакин Н. А. Автоматизированные системы, базы и банки данных. Вводный курс: Учебное пособие. — М.: Гелиос АРВ, 2002. — 368 с.
4. Інформатика: Комп'ютерна техніка. Комп'ютерні технології. — Київ : Академія, 2002. — 704 с.
5. Леонтьев В.П. Новейшая энциклопедия персонального компьютера. — ОЛМА Медиа Групп, ОЛМА-ПРЕСС Образование, 2006. — С. 896.
6. Руденко В.Д., Макачук О.М., Патланжоглу М.О. Практичний курс інформатики. — Київ : Фенікс, 1996. — С. 418.
7. Ярмуш О.В., Редько М.М. Інформатика і комп'ютерна техніка. — Київ : Вища освіта, 2006. — С. 359.
8. Гуржій А.М., Поворознюк Н.І., Самсонов В.В. Інформатика та інформаційні технології. — Харків : ООО «Компанія СМІТ», 2003. — С. 352.
9. Інформатика. Комп'ютерна техніка. Комп'ютерні технології. — Київ : Каравела, 2011. — С. 592.
10. Криптографія. [Електронний ресурс]. — Режим доступу: https://esu.com.ua/search_articles.php?id=1576. — Дата доступу: травень 2022. — Назва з екрану.
11. Шнайер Б. Прикладная криптография. — М. : Триумф, 2002. — 816 с. (рос.).
12. Вербіцький О. В. Вступ до криптології. — Л. : ВНТЛ, 1998. — 248 с.
13. Мао В. Современная криптография. — М. : Диалектика, 2005. — 768 с. (рос.)
14. Грайворонський, М. В. Безпека інформаційно-комунікаційних систем. підручник / М. В. Грайворонський, О. М. Новіков. — Київ : Видавнича група BHV, 2009. — 698 с.

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Зм.	Арк.	№ докум.	Підпис	Дата		

15. Методологія функціонального моделювання IDEF0 [Електронний ресурс]. – Режим доступу : <https://nsu.ru/smk/files/idef.pdf>. – Дата доступу: травень 2022. – Назва з екрану.
16. Методологія IDEF3. [Електронний ресурс]. – Режим доступу : <https://itteach.ru/bpwin/metodologiya-idef3>. – Дата доступу: травень 2022. – Назва з екрану.
17. DFD – Вікіпедія. [Електронний ресурс]. – Режим доступу : <https://ru.wikipedia.org/wiki/DFD>. – Дата доступу: травень 2022. – Назва з екрану.
18. Леоненков А. UML 2: самоучитель. / А. Леоненков. – СПб : БХВ – Петербург, 2007. – 246 с.
19. А. Хейлсберг, М. Торгерсен, С. Вилтамут, П. Голд. Язык программирования C#. Классика Computers Science. 4-е издание = C# Programming Language (Covering C# 4.0), 4th Ed. — СПб.: «Питер», 2012. — 784 с.
20. Microsoft.com. [Електронний ресурс]. – Режим доступу: <https://www.microsoft.com/ru-ru/download/details.aspx?id=35579> – Дата доступу: травень 2022. – Назва з екрану.
21. Канер С. Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений / С. Канер, Д. Фолк, Е. Нгуен. – К. : Издательство «ДиаСофт», 2010. – 544 с.
22. Ініціалізаційний вектор – Вікіпедія. [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Ініціалізаційний_вектор – Дата доступу: травень 2022. – Назва з екрану.

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змн	Арк	№ докум	Підпис	Дата		

ДОДАТКИ

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Пашаріна І.В.				
Змі	Арк.	№ докум	Підпис	Дата		

Лістинг коду myForm.aspx

```

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="myForm.aspx.cs"
Inherits="Demion.myForm" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Demion</title>
</head>
<body id="MasterBody" runat="server" style="background-color: #FED6BC;">
    <form id="mainForm" runat="server">
        <div style="text-align: center;">
            <asp:FileUpload ID="fileUploader" runat="server" BackColor="White"
BorderColor="Black" Width="600px" />
        </div>
        <div style="text-align: center;">
            <asp:Button ID="btnSend"
runat="server"
Text="Записати до БД" OnClick="btnSend_Click" ToolTip="Записати обраний
файл до БД, зашифрувавши його" />
        </div>
        <div style="text-align: center;">
        </div>
        <div>
            <asp:Button ID="btnOpen" runat="server" Text="Відкрити файл"
OnClick="btnOpen_Click" />
        </div>
        <div>
            <asp:ListBox ID="list" runat="server" BackColor="#E7ECFF" Height="100px"
Width="800px" OnSelectedIndexChanged="list_SelectedIndexChanged"></asp:ListBox>
        </div>
    </form>
</body>
</html>

```

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змі	Арк.	№ докум	Підпис	Дата		

Лістинг коду myForm.aspx.cs

```

using System;
using System.Collections.Generic;
using System.Data;
using System.IO;
using System.Linq;
using System.Text;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Demion
{
    public partial class myForm : System.Web.UI.Page
    {
        /// <summary>
        /// Об'єкт для роботи з БД
        /// </summary>
        DataManager dm = new DataManager();

        /// <summary>
        /// Завантаження сторінки
        /// </summary>
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                ShowFilesInFileList();
            }
        }

        /// <summary>
        /// Індекс обраного файлу у списку
        /// </summary>
        int selectedIndex = 0;

        /// <summary>
        /// Відобразити вміст БД у списку файлів
        /// </summary>
        private void ShowFilesInFileList()
        {
            try { selectedIndex = list.SelectedIndex + 1; }
            catch { }
            DataTable dt = dm.SelectFromMainTable();

            list.Items.Clear();
            foreach (DataRow row in dt.Rows)
            {
                string filename = row["filename"].ToString();
                list.Items.Add(filename);
            }
        }

        /// <summary>
        /// Записати обраний файл до БД, зашифрувавши його
        /// </summary>
        protected void btnSend_Click(object sender, EventArgs e)
        {
            if (!fileUploader.HasFile) // якщо файл не обрано - сповістити про це
            {

```

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Змн	Арк	№ докум	Підпис	Дата		

```

        ClientScript.RegisterStartupScript(this.GetType(), "myalert",
"alert('Оберіть файл!');", true);
        return;
    }
    string password = GeneratePassword(14); // пароль шифрування
    byte[] key = EncryptWorker.CreateKey(password);
    byte[] IV = EncryptWorker.CreateIV(password);

    byte[] encrypted = EncryptWorker.Encrypt(fileUploader.FileBytes, key, IV);
    if (!dm.InsertValues(fileUploader.FileName, encrypted, password, key, IV))
        ClientScript.RegisterStartupScript(this.GetType(), "myalert", "alert('Не
вдалося записати файл!');", true);
    ShowFilesInFileList();
}

/// <summary>
/// Згенерувати пароль випадковим чином
/// </summary>
/// <param name="len">Довжина пароля</param>
private string GeneratePassword(int len)
{
    string alphabet =
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890";
    StringBuilder res = new StringBuilder();
    Random rnd = new Random();
    while (0 < len--)
        res.Append(alphabet[rnd.Next(alphabet.Length)]);
    return res.ToString();
}

/// <summary>
/// Кнопка "Відкрити файл"
/// </summary>
protected void btnOpen_Click(object sender, EventArgs e)
{
    string path = HttpContext.Current.Server.MapPath("~/"); // шлях до кореня
    path += list.Items[selectedIndex];
    if (dm.SaveDecryptedFile(path, selectedIndex + 1))
    {
        // якщо вийшло розшифрувати - відкрити збережений файл
        System.Diagnostics.Process process = new System.Diagnostics.Process();
        process.StartInfo.FileName = path;
        process.Start();
    }
}

/// <summary>
/// Запам'ятовувати обраний індекс у списку
/// </summary>
protected void list_SelectedIndexChanged(object sender, EventArgs e)
{
    selectedIndex = list.SelectedIndex;
}
}
}

```

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Зм.	Арк.	№ докум.	Підпис	Дата		

Лістинг коду myForm.aspx.designer.cs

```
//-----
// <автоматически создаваемое>
//     Этот код создан программой.
//
//     Изменения в этом файле могут привести к неправильной работе и будут потеряны в случае
//     повторной генерации кода.
// </автоматически создаваемое>
//-----

namespace Demion {

    public partial class myForm {

        /// <summary>
        /// MasterBody элемент управления.
        /// </summary>
        /// <remarks>
        /// Автоматически создаваемое поле.
        /// Для изменения переместите объявление поля из файла конструктора в файл кода
        программной части.
        /// </remarks>
        protected global::System.Web.UI.HtmlControls.HtmlGenericControl MasterBody;

        /// <summary>
        /// mainForm элемент управления.
        /// </summary>
        /// <remarks>
        /// Автоматически создаваемое поле.
        /// Для изменения переместите объявление поля из файла конструктора в файл кода
        программной части.
        /// </remarks>
        protected global::System.Web.UI.HtmlControls.HtmlForm mainForm;

        /// <summary>
        /// fileUploader элемент управления.
        /// </summary>
        /// <remarks>
        /// Автоматически создаваемое поле.
        /// Для изменения переместите объявление поля из файла конструктора в файл кода
        программной части.
        /// </remarks>
        protected global::System.Web.UI.WebControls.FileUpload fileUploader;

        /// <summary>
        /// btnSend элемент управления.
        /// </summary>
        /// <remarks>
        /// Автоматически создаваемое поле.
        /// Для изменения переместите объявление поля из файла конструктора в файл кода
        программной части.
        /// </remarks>
        protected global::System.Web.UI.WebControls.Button btnSend;

        /// <summary>
        /// btnOpen элемент управления.
        /// </summary>
        /// <remarks>
        /// Автоматически создаваемое поле.
```

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Павлюк І.В.				
Зм.	Арк.	№ докум.	Підпис	Дата		

```

    /// Для изменения переместите объявление поля из файла конструктора в файл кода
программной части.
    /// </remarks>
    protected global::System.Web.UI.WebControls.Button btnOpen;

    /// <summary>
    /// list элемент управления.
    /// </summary>
    /// <remarks>
    /// Автоматически создаваемое поле.
    /// Для изменения переместите объявление поля из файла конструктора в файл кода
программной части.
    /// </remarks>
    protected global::System.Web.UI.WebControls.ListBox list;
}
}

```

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змі	Арк.	№ докум	Підпис	Дата		

Лістинг коду EncryptWorker.cs

```

using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Web;

namespace Demion
{
    public static class EncryptWorker
    {
        /// <summary>
        /// Зашифрувати
        /// </summary>
        /// <param name="data">Масив даних</param>
        /// <param name="key">Ключ</param>
        /// <param name="IV">IV</param>
        public static byte[] Encrypt(byte[] data, byte[] key, byte[] IV) // string password
        {
            //byte[] key = CreateKey(password);
            //byte[] IV = CreateIV(password);

            byte[] encrypted;
            using (Aes aes = Aes.Create())
            {
                aes.Mode = CipherMode.CBC;
                aes.KeySize = 128;
                aes.BlockSize = 128;
                aes.FeedbackSize = 128;
                aes.Padding = PaddingMode.Zeros;
                aes.Key = key;
                aes.IV = IV;
                ICryptoTransform encryptor = aes.CreateEncryptor(aes.Key, aes.IV);
                using (MemoryStream msEncrypt = new MemoryStream())
                {
                    using (CryptoStream csEncrypt = new CryptoStream(msEncrypt, encryptor,
CryptoStreamMode.Write))
                    {
                        csEncrypt.Write(data, 0, data.Length);
                        csEncrypt.FlushFinalBlock();
                        encrypted = msEncrypt.ToArray();
                    }
                }
            }
            return encrypted;
        }

        /// <summary>
        /// Розшифрувати
        /// </summary>
        /// <param name="data">Масив зашифрованих даних</param>
        /// <param name="key">Ключ</param>
        /// <param name="IV">IV</param>
        public static byte[] Decrypt(byte[] data, byte[] key, byte[] IV)
        {
            byte[] decrypted;
            using (Aes aes = Aes.Create())
            {
                aes.Mode = CipherMode.CBC;

```

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арх.
		Павлюк І.В.				
Зм.	Арх.	№ докум.	Підпис	Дата		

```

        aes.KeySize = 128;
        aes.BlockSize = 128;
        aes.FeedbackSize = 128;
        aes.Padding = PaddingMode.Zeros;
        aes.Key = key;
        aes.IV = IV;
        var decryptor = aes.CreateDecryptor(aes.Key, aes.IV);
        using (MemoryStream msEncrypt = new MemoryStream())
        {
            using (CryptoStream csEncrypt = new CryptoStream(msEncrypt, decryptor,
CryptoStreamMode.Write))
            {
                csEncrypt.Write(data, 0, data.Length);
                csEncrypt.FlushFinalBlock();
                decrypted = msEncrypt.ToArray();
            }
        }
    }
    return decrypted;
}

/// <summary>
/// Сгенерувати IV
/// </summary>
/// <param name="password">Пароль, на основі якого проведеться генерування</param>
public static byte[] CreateIV(string password)
{
    var salt = GetSalt(9);

    const int Iterations = 325;
    using (var rfc2898DeriveBytes = new Rfc2898DeriveBytes(password, salt,
Iterations))
        return rfc2898DeriveBytes.GetBytes(16);
}

/// <summary>
/// Сгенерувати ключ
/// </summary>
/// <param name="password">Пароль, на основі якого проведеться генерування</param>
public static byte[] CreateKey(string password)
{
    var salt = GetSalt(10);

    int iterationCount = 20000; // Nowadays you should use at least 10.000
iterations
    using (var rfc2898DeriveBytes = new Rfc2898DeriveBytes(password, salt,
iterationCount))
        return rfc2898DeriveBytes.GetBytes(16);
}

private static int saltLengthLimit = 32;
private static byte[] GetSalt(int maximumSaltLength)
{
    var salt = new byte[maximumSaltLength];
    using (var random = new RNGCryptoServiceProvider())
    {
        random.GetNonZeroBytes(salt);
    }

    return salt;
}
}
}

```

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арх.
		Паваріна І.В.				
Змн	Арх	№ докум	Підпис	Дата		

Лістинг коду EncryptedFile.cs

```

using System;

namespace Demion
{
    public class EncryptedFile
    {
        /// <summary>
        /// Пароль
        /// </summary>
        public string Password { get; set; }

        /// <summary>
        /// Ключ
        /// </summary>
        public byte[] Key { get; set; }

        /// <summary>
        /// IV
        /// </summary>
        public byte[] IV { get; set; }

        public EncryptedFile(string password, byte[] key, byte[] iv)
        {
            Password = password;
            Key = key;
            IV = iv;
        }
    }
}

```

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змн	Арк.	№ докум	Підпис	Дата		

Лістинг коду DataManager.cs

```

using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.IO;
using System.Linq;
using System.Text;
using System.Web;

namespace Demion
{
    /// <summary>
    /// Клас для роботи з БД
    /// </summary>
    public class DataManager
    {
        /// <summary>
        /// Рядок підключення до основної БД
        /// </summary>
        private string conStrMain = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename="
+ AppDomain.CurrentDomain.BaseDirectory + "App_Data\\Files.mdf;Integrated Security=True";

        /// <summary>
        /// Рядок підключення до БД з паролями
        /// </summary>
        private string conStrPass = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename="
+ AppDomain.CurrentDomain.BaseDirectory + "App_Data\\Encrypt.mdf;Integrated Security=True";

        /// <summary>
        /// Команда, через яку проходить робота з основною БД
        /// </summary>
        private SqlCommand cmdMain;

        /// <summary>
        /// Команда, через яку проходить робота з БД з паролями
        /// </summary>
        private SqlCommand cmdPass;

        /// <summary>
        /// Конструктор класу
        /// </summary>
        public DataManager()
        {
            cmdMain = new SqlCommand(); // ініціалізувати команду
            cmdMain.Connection = new SqlConnection(conStrMain); // ініціалізувати з'єднання з
            БД

            cmdPass = new SqlCommand(); // ініціалізувати команду
            cmdPass.Connection = new SqlConnection(conStrPass); // ініціалізувати з'єднання з
            БД

            Connect();
        }

        /// <summary>
        /// Підключитися до БД
        /// </summary>
        public void Connect()
        {
            if (cmdMain.Connection.State != ConnectionState.Open)

```

		Блумберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Татаріна І.В.				
Зм.	Арк.	№ докум.	Підпис	Дата		

```

        cmdMain.Connection.Open();
        if (cmdPass.Connection.State != ConnectionState.Open)
            cmdPass.Connection.Open();
    }

    /// <summary>
    /// Закрити підключення до БД
    /// </summary>
    public void Disconnect()
    {
        cmdMain.Connection.Close();
        cmdPass.Connection.Close();
    }

    /// <summary>
    /// Метод відбору SELECT з БД
    /// </summary>
    /// <param name="query">Строка запиту</param>
    /// <param name="cmd">Команда</param>
    /// <returns>Повертає вибірку з БД</returns>
    public DataTable Select(string query, SqlCommand cmd)
    {
        try
        {
            DataTable ds = new DataTable();
            SqlDataAdapter adapter = new SqlDataAdapter(query, cmd.Connection);
            adapter.Fill(ds);
            return ds;
        }
        catch (Exception ex)
        {
            return null;
        }
    }

    /// <summary>
    /// Метод відбору SELECT з основної БД
    /// </summary>
    /// <param name="query">Строка запиту</param>
    /// <returns>Повертає вибірку з БД</returns>
    public DataTable SelectFromMainTable()
    {
        string query = "select * from [dbo].[Main_Table]";
        return Select(query, cmdMain);
    }

    /// <summary>
    /// Зберегти файл, попередньо розшифрувавши його
    /// </summary>
    /// <param name="idFile">id_file у БД</param>
    public bool SaveDecryptedFile(string path, int idFile)
    {
        string query = $"select * from [dbo].[Main_Table] where id_file={idFile}";
        DataTable dt = Select(query, cmdMain);
        byte[] bytes = (byte[])dt.Rows[0]["file_bytes"];

        query = $"select * from [dbo].[Passwords] where id_file={idFile}";
        dt = Select(query, cmdPass);
        EncryptedFile encFile = new EncryptedFile(dt.Rows[0]["password"].ToString(),
            (byte[])dt.Rows[0]["aes_key"],
            (byte[])dt.Rows[0]["aes_iv"]);

        byte[] result = EncryptWorker.Decrypt(bytes, encFile.Key, encFile.IV);
    }

```

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        File.WriteAllBytes(path, result);
        return true;
    }

    /// <summary>
    /// Команда Insert до БД
    /// </summary>
    /// <param name="command">Команда для запису</param>
    /// <returns>true - запис виконано, false - помилка запису до БД</returns>
    public int Insert(string command, SqlCommand cmd)
    {
        object result;
        cmd.CommandText = command;
        try { result = cmd.ExecuteScalar(); }
        catch (Exception ex)
        {
            return -1;
        }
        if (result == null)
            return -1;
        return (int)result;
    }

    /// <summary>
    /// Записати значення до двох БД
    /// </summary>
    /// <param name="filename">Ім'я файлу</param>
    /// <param name="encrypted">Масив шифрованих байтів</param>
    /// <param name="password">Пароль</param>
    /// <param name="key">Ключ на основі пароля</param>
    /// <param name="IV">IV на основі пароля</param>
    public bool InsertValues(string filename, byte[] encrypted, string password, byte[]
key, byte[] IV)
    {
        string command = String.Format(@"INSERT INTO [dbo].[Main_Table]
            ([filename], [file_bytes], [file_date])
            output inserted.id_file
            VALUES
            ({0}', @bytes, GETDATE())",
            filename);

        try
        {
            cmdMain.Parameters.Add("@bytes", SqlDbType.VarBinary).Value = encrypted;
            int result = Insert(command, cmdMain);
            cmdMain.Parameters.Clear();

            if (result < 1)
                return false;

            command = $"INSERT INTO [dbo].[Passwords]
                ([id_file], [password], [aes_key], [aes_iv])
                VALUES
                ({result}, '{password}', @bytesKey, @bytesIV)";
            cmdPass.Parameters.Add("@bytesKey", SqlDbType.VarBinary).Value = key;
            cmdPass.Parameters.Add("@bytesIV", SqlDbType.VarBinary).Value = IV;
            Insert(command, cmdPass);
            cmdPass.Parameters.Clear();
        }
        catch
        {
            cmdMain.Parameters.Clear();
            cmdPass.Parameters.Clear();
        }
        return true;
    }

```

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змі	Арк	№ докум	Підпис	Дата		

}} }

		Блюмберг К.О.			КІК.КР.Б – 123 – 22 – ПЗ	Арк.
		Паваріна І.В.				
Змі	Арк.	№ докум	Підпис	Дата		