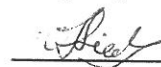


Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

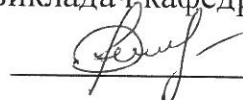
ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра
на тему: «Проект підсистеми захищеного обміну
документами інформаційної системи документообороту
підприємства»

Виконав студент 4-го курсу групи КБ-2
спеціальності 125 «Кібербезпека»

 Андрій Сірик

Керівник Старший викладач кафедри КІ та КБ

 Єлизавета Байлюк

Рецензент Старший викладач кафедри КІ та КБ

 Олександра Покотило

Житомир – 2022

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Єфіменко

1 березня 2022 р.

ЗАВДАННЯ

на випускню кваліфікаційну роботу бакалавра

Студент: Сірик Андрій Юрійович

Тема роботи: «Проект підсистеми захищеного обміну документами інформаційної системи документообороту підприємства»

затверджена Наказом університету від 1 березня 2022 р. № 128/с

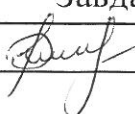
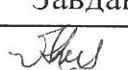
Термін здачі студентом закінченої роботи 10 червня 2022 р.

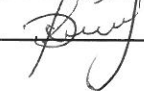
Вихідні дані роботи (зазначається предмет і об'єкт дослідження): Проект безпечного обміну документами;

Обладнання та програмне забезпечення:

1. Персональний комп'ютер Intel Core i5 8th;
2. ОС Windows 10;
3. Microsoft Visual Studio 2022;
4. Microsoft SQL Server Management Studio 18;
5. Microsoft Office Excel.

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

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Байлюк Є. М.	 01.03.22	 01.03.2022

Студент  Андрій Сірик
Керівник  Єлизавета Байлюк

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	01.03.22-02.03.22	
2	Аналіз завдання	02.03.2022-03.03.22	
3	Формування технічного завдання	03.03.2022	
4	Встановлення потрібних додатків	03.03.22-04.03.22	
5	Розробка аутентифікації	04.03.22-15.03.22	
6	Розробка формування документів	15.03.22-30.03.22	
7	Розробка функцій адміністрування	30.03.22-10.04.22	
8	Розробка вивантаження на сервер	10.04.22-20.04.22	
9	Розробка методу шифрування	20.04.22-25.04.22	
10	Тестування	25.04.22-01.05.22	
11	Написання пояснювальної записки	01.05.22-01.06.22	

Студент Андрій Сірик

Керівник Єлизавета Байлюк

АНОТАЦІЯ

Пояснювальна записка до дипломної роботи «Підсистема захищеного обміну документами інформаційної системи документообороту підприємства». Складається з вступу, трьох розділів, висновків, списку використаної літератури.

У дипломній роботі було досліджено особливості розробки та використання захищеного електронного документообігу.

У першому розділі було досліджено використання електронного документообігу в бізнес-процесах, проаналізовано користь та безпечність даного методу в порівнянні з традиційним. У порівнянні, обрано тип шифрування для проекту. Проаналізовано використання документообігу в Україні та порівняно найпопулярніші сервіси. А також створено технічне завдання для проекту.

У другому розділі було проаналізовано мову програмування C#, а також платформу .net framework 6.0. Обрано найвідоміший алгоритм відкритого ключа RSA та досліджено принцип роботи методу. Розглянуто протокол безпечної передачі файлів SFTP.

У третьому розділі було розроблено підсистему захищеного документообігу у між підприємствами в інформаційній системі, протестовано проект та описано основні принципи роботи.

Текстова частина викладена на 59 сторінках друкованого тексту. Список використаних джерел містить 20 найменувань і займає 2 сторінки. В роботі наведено 37 рисунків. Загальний обсяг роботи – 67 сторінок.

Ключові слова: C#, SFTP, WPF, RSA, СЕД.

					КІК.КР.Б – 125 – 22 – ПЗ				
Змн.	Арк.	№ докум.	Підпис	Дата	Проект підсистеми захищеного обміну документами інформаційної системи документообороту підприємства	Літ.		Арк.	Аркушів
Розроб.		Сірик А. Ю.		10.06				4	67
Керівник		Байлюк Є. М.		28.06					
Реценз.		Покотило О. А.		28.06					
Н. контр.		Байлюк Є. М.							
Затверд.		Єфіменко А.А.		28.06		ФІКТ, Гр. КБ-2			

ANNOTATION

Explanatory note to the thesis "Subsystem of secure exchange of documents of the information system of document management of the enterprise". It consists of an introduction, three chapters, conclusions, a list of references.

In the thesis the peculiarities of development and use of secure electronic document management were investigated.

The first section examines the use of electronic document management in business processes, analyzes the benefits and safety of this method compared to traditional. In comparison, the encryption type for the project is selected. The use of document management in Ukraine and comparatively the most popular services are analyzed. And also the technical task for the project is created.

The second section analyzed the C # programming language, as well as the .net framework 6.0. The most well-known RSA public key algorithm is chosen and the principle of operation of the method is investigated. The protocol of safe transfer of SFTP files is considered.

In the third section the subsystem of the protected document circulation in between the enterprises in information system was developed, the project was tested and the basic principles of work were described.

The text is set out on 58 pages of printed text. The list of used sources contains 20 names and occupies 2 pages. The paper presents 37 figures. The total volume of the work is 67 pages.

Keywords: C #, SFTP, WPF, RSA, SED.

		<i>Сірик А. Ю.</i>			<i>КІК.КР.Б – 125 – 22 – ПЗ</i>	<i>Арк.</i>
		<i>Байлюк Є. М.</i>				
<i>Змн.</i>	<i>Арк.</i>	<i>№ докум.</i>	<i>Підпис</i>	<i>Дата</i>		<i>5</i>

ЗМІСТ

ВСТУП	8
РОЗДІЛ 1 ОБҐРУНТУВАННЯ НЕБХІДНОСТІ РОЗРОБКИ ПІДСИСТЕМИ ЗАХИЩЕНОГО ОБМІНУ ДОКУМЕНТАМИ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДОКУМЕНТООБОРОТУ	9
1.1 Автоматизація бізнес-процесу, документообіг	9
1.2 Безпечна передача інформації.....	15
1.3 Використання документообігу в Україні	24
1.4 Постановка технічного завдання	30
РОЗДІЛ 2 АНАЛІЗ ТЕХНОЛОГІЙ ПРОЄКТУВАННЯ ПІДСИСТЕМИ ЗАХИЩЕНОГО ОБМІНУ ДОКУМЕНТАМИ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДОКУМЕНТООБОРОТУ	32
2.1 Мова програмування C#	32
2.2 Платформа .net framework	36
2.3 Метод асиметричного шифрування RSA.....	39
2.4 Протокол передачі файлів SFTP	44
РОЗДІЛ 3 ПРОЕКТНА РЕАЛІЗАЦІЯ ПІДСИСТЕМИ ЗАХИЩЕНОГО ДОКУМЕНТООБІГУ «DOC FLOW».....	47
3.1 Опис додатку, тестування.....	47
ВИСНОВКИ.....	56
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	57
ДОДАТКИ.....	59

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

СЕД – Система електронного документообігу

PKI – Public Key Infrastructure

TSL – Transport Layer Security

SFTP – SSH File Transfer Protocol

WPF – Windows Presentation Foundation

GUI – Graphical user interface

XML – Extensible Markup Language

CLR – Common Language Runtime Developers

MSIL – Microsoft Intermediate Language

RSA – Криптографічний алгоритм

C# – Мова програмування

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				7
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. Системи електронного документообігу, а також відмова від паперів, на сьогоднішній день стрімко впроваджуються повсюдно. Вони мають незаперечну низку переваг, таких, наприклад, як електронний архів, у порівнянні з обробкою паперових документів.

Автоматизація документообігу стає ознакою успішної компанії, зацікавленою у підвищенні ефективності своїх бізнес-процесів, а отже, і конкурентоспроможності загалом.

Актуальність електронного документообігу визначається об'єктивними вимогами часу, такими як зростання інформаційного потоку та динамічність інформаційного середовища.

Система електронного документообігу дозволяє як упорядкувати діловодство, заощадити час і витратні матеріали, так й сприяє чіткішій організації бізнес-процесів компанії, часто значно підвищуючи її ефективність.

Метою кваліфікаційної роботи є розробка проєкту для захищеного обміну документами в інформаційній системі

Визначена мета обумовлює наступні завдання:

- 1) Розробка додатку з базовою аутентифікацією;
- 2) Налаштування шаблону документів;
- 3) Використання методу асиметричного шифрування;
- 4) Розробка вивантаження документів на спільний сервер та завантаження з нього.

Об'єктом дослідження є технології роботи документообігу

Предметом дослідження є підсистема захищеного обміну документами

У процесі виконання кваліфікаційної роботи було розроблено проєкт безпечного документообігу між організаціями, який обумовлює використання асиметричного шифрування при передачі документів.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				8
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1 ОБҐРУНТУВАННЯ НЕБХІДНОСТІ РОЗРОБКИ ПІДСИСТЕМИ ЗАХИЩЕНОГО ОБМІНУ ДОКУМЕНТАМИ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДОКУМЕНТООБОРОТУ

1.1 Автоматизація бізнес-процесу, документообіг

Завдяки швидкому розвитку технологій та їх широкому впровадженню електронний документообіг та архівна справа посідають найвищі позиції порівняно з паперовим аналогом.

Кількість та обсяг документів, які використовуються в сучасному бізнес-просторі, збільшується. Причому частка електронних і паперових документів з часом змінюється на користь цифрових технологій. Зараз, за статистикою, обсяг корпоративної електронної текстової інформації зростає у геометричній прогресії.

Характерною рисою суспільства на сучасному етапі є його глибока комп'ютеризація. Рівень комп'ютеризації визначається рівнем використаних інформаційних технологій. Одним з основних напрямків розвитку ІТ є системи управління, які використовуються для автоматизації таких завдань, як виробництво, облік, збут, управління персоналом, фінансами та інформаційними ресурсами.

Можна сміливо сказати, що відмовитися повністю від паперових документів ніколи не вдасться, однак, як стверджують досвідчені фахівці в цій галузі, використання електронних варіантів все одно буде домінувати [20].

Система електронного документообігу (далі – СЕД) служить основою для впровадження систем управління бізнесом, систем управління збутом та інших важливих для сучасної організації ІТ-систем, а отже, і для функціонування СЕД всіх ІТ-систем організації [20].

Збільшення обсягів інформації та впровадження комп'ютерних технологій призвело до підвищення вимог до організації інформаційно-документаційного обслуговування. За цих умов виникає необхідність раціоналізації перевірених методів пошуку, обробки та зберігання інформації (документів) шляхом розробки

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

нових прийомів, режимів і методів оцінки, аналізу та оптимізації документообігу підприємства з використанням сучасних інформаційних технологій.

Перш за все, для офісного робітника великою перевагою буде обмеження рутини та зручності роботи з документами, можливість не витратити час на пошук необхідної інформації. Важливим фактором є простота використання СЕД, «дружність» її інтерфейсу. Система повинна бути зрозумілою і зрозумілою звичайному працівнику, якому доводиться працювати з нею щодня.

За даними Forrester Research, 38% компаній зі списку Fortune 500 вважають, що придбання сучасної СЕД має вирішальне значення для успіху їхнього бізнесу. На думку галузевих аналітиків (є багато таких думок, які в деяких моментах відрізняються одна від одної), переваги для корпоративних користувачів від впровадження СЕД досить різноманітні [6]. Наприклад, за даними Siemens Business Services, при використанні СЕД:

1. Продуктивність персоналу зростає на 20-25%;
2. Вартість архівування електронних документів на 80% нижча порівняно з вартістю зберігання паперових архівів.



Рисунок 1.1 – Порівняння традиційного і електронного документообігу

Розглянемо привілеї СЕД. Основними перевагами електронного документообігу є:

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

1. Підвищення прозорості та керованості бізнесу завдяки покращеному доступу до інформації та чіткішому контролю процесу. З метою підвищення прозорості бізнес-процесів для керівництва та зниження ймовірності шахрайства.
2. Надійне зберігання та запобігання втраті документів, в тому числі в результаті форс-мажорних обставин, що означає обмеження правових та економічних ризиків.
3. Зниження витрат. Впровадження електронного документообігу дозволяє знизити витрати на зберігання документації, а також витратних матеріалів, які використовуються при управлінні паперовими документами.
4. Розширення доступу до документів. Кожен користувач системи бачить лише ті документи, до яких має доступ.
5. Прискорення процесу узгодження та затвердження документів.

Однак радикальні, революційні підходи до автоматизації документів небезпечні, оскільки можуть призвести до серйозних проблем управління і навіть дезорганізації операцій. склад в організації.

Отже, електронний документообіг – це сукупність технологій, які не тільки суттєво оптимізують, але й істотно змінюють роботу будь-якої організації. І очевидно, що система такого масштабу не може бути впроваджена «з нуля», оскільки це може спричинити тривалий параліч діяльності організації. Причому збої відбуватимуться не тільки в автоматизованому внутрішньому середовищі організації, але як наслідок - у підпорядкованих їй організаціях. Тому перед впровадженням системи електронного документообігу та діловодства слід ретельно оцінити готовність усіх осіб, які приймають рішення (організації) до нових якісних технологій.

Системи часто бувають модульними, являючи собою набір апаратних та програмних засобів, які керують створенням, твердженням, поширенням та іншими фазами життєвого циклу документа. Управління документами в електронному вигляді не означає використання лише електронних документів.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		11

Управління документами можна визначити як процес нагляду за офіційними діловими операціями підприємства, записами про прийняття рішень та тимчасовими документами, що мають важливе значення, які представлені у форматі документа [10].

Дана технологія використовується для управління завданнями, затвердження та розподілу у життєвому циклі документа. Робочий процес ґрунтується на правилах, які передають в електронному вигляді завдання з документом або кількома документами на робочий стіл спеціалістів. Вони зазвичай поділяють робочу діяльність на чітко визначені завдання, ролі, правила та процедури, які регулюють більшу частину роботи у виробництві та офісі. Реалізація робочого процесу може включати активацію, відстеження, контроль стану, обмін повідомленнями, передачу черг та маршрутизацію документів. Стає можливим контролювати процес управління, відстежувати та маршрутизувати пов'язані з ним документи.

Управління робочими процесами включає: моделювання процесів, для чого необхідні моделі робочих процесів і методи захоплення та опису процесу; реінжиніринг процесів, навіщо необхідні методи оптимізації процесу; впровадження та автоматизація робочих процесів, для чого необхідні методології та технології використання інформаційних систем та авторів для реалізації, планування, виконання та контролю завдань робочого процесу, описаних у специфікації робочого процесу [7].

Системи електронного документообігу - це набір технологій, пов'язаних задля досягнення поставленої мети. У СЕД, як правило, вбудовані такі технології:

- 1) технологія перетворення паперових документів на цифрові за допомогою сканерів;
- 2) технологія пошуку документів за словами у них;
- 3) технологія інновації виробничих процесів шляхом їхнього реінжинірингу, що дозволяє контролювати маршрут руху документів у компанії;

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		12

СЕД зазвичай базується на двох основних системах: Document Management, яка керує динамічними документами, такими як файл текстового процесора, та Document Imaging, яка має справу зі статичними документами, такими як зображення документів, отримані за допомогою сканера. Система також може містити бази даних, які допомагають організувати паперові документи. Це можуть бути факси, скановані зображення або документи, креслення, текстові документи, електронні таблиці, звіти баз даних, листи, специфікації та, по суті, будь-які документи.

Ринок систем безпеки електронного документообігу є однією з найбільш розвиваючих галузей ІТ-індустрії за останні роки.

Оскільки системи електронного документообігу можуть піддаватися різним хакерським атакам, різні компанії пропонують власні рішення для їх захисту.

Розробка базується на технології PKI (Public Key Infrastructure). Ця технологія заснована на використанні асиметричних криптографічних алгоритмів. Такий захист передбачає наявність двох різних ключів – публічного та приватного. Таким чином, безпека робочого процесу досягається за допомогою шифрування документів, а надійність – за допомогою електронного цифрового підпису (ЕЦП). Електронний цифровий підпис повинен, перш за все, забезпечувати достовірність, автентичність та юридичну відповідність електронним документам. Застосований у вищезазначеній системі рівень безпеки дозволяє обмінюватися електронними документами, що містять конфіденційну інформацію.

Крім того, для захисту електронних документів використовуються сертифіковані засоби захисту криптографічної інформації. Ці зміни забезпечують доступ до конфіденційних документів лише автору документа. Інші користувачі системи не мають цього права [7].

Доступ до документів контролюється системою, а правила системи встановлює адміністратор безпеки.

Системи workflow - це системи, які регулюють правильну послідовність виконання завдань з документами. Модифікацією цих систем є Business Process Management (BPM): вони дозволяють керувати процесами в динаміці – збираючи

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		13

та аналізуючи статистику канцелярських процесів. При аналізі життєвого циклу електронного документа слід чітко дотримуватися основних вимог до систем електронного документообігу:

- 1) вводити дані користувача виходячи з організаційно-кадрової структури організації;
- 2) забезпечити єдину реєстрацію всієї вхідної кореспонденції, в тому числі листів і скарг громадян, з подальшим направленням документів на розгляд керівництву організації або її структурним підрозділам;
- 3) забезпечувати доступ до документів на рівні окремих користувачів, на рольовій основі та на основі ієрархічної структури організації;
- 4) ведення журналів реєстрації та контролю виконання; - контролювати строки виконання документів, інформувати підрядника та діловода про наближення контрольних строків, про невиконання документів у строк;
- 5) забезпечити надійне зберігання документів та їх опис;
- 6) підтримка шаблонів документів, реквізитів документів, проектів документів, зв'язків між документами - відстеження документів поза системою, вилучення документів із системи;
- 7) підтримання ієрархії документів на етапі проектування, у тому числі колективної розробки;
- 8) супроводжувати затвердження, затвердження, затвердження документа;
- 9) документообіг - документообіг, обробка всіх видів маршрутів, автоматична відправка повідомлень, обмін повідомленнями та замовленнями в системі, створення записів відправлень іншим організаціям;

Шифрування файлів у цій системі запобігає несанкціонованому доступу до інформації. Необхідність захисту електронного документообігу важлива не тільки при роботі з документами, а й при їх відправленні. Дуже важливо, щоб документ не читався і не змінювався під час доставки. Це гарантує, що лише призначені

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк С. М.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

користувачі зможуть прочитати листування, якщо у них є ключі електронного підпису. Шифрування даних та встановлення електронного підпису здійснюється автоматично. Запровадження такого захисту дозволяє не тільки знизити ймовірність втрати паперових документів, але й підвищити впевненість у досягненні повної конфіденційності інформації.

1.2 Безпечна передача інформації

Шифрування - це єдиний спосіб забезпечити цілісність інформації при її зберіганні та передачі. Конфіденційність є ключовим пріоритетом при використанні криптографії. Вона означає, що тільки люди з відповідним дозволом можуть отримати доступ до інформації, що передається, і що ця інформація захищена від несанкціонованого доступу на всіх етапах.

Конфіденційність необхідна для збереження приватності тих, чия особиста інформація зберігається у корпоративних системах.

Навіть якщо зашифрована інформація надходить до зловмисника, вона практично непотрібна для неавторизованих осіб, які не мають відповідних ключів для розшифрування.

Якщо система має цілісність, це означає, що дані в системі переміщуються та обробляються передбачуваним чином. Навіть коли дані обробляються, вони не змінюються.

Криптографія забезпечує цілісність даних за допомогою алгоритмів хешування та методів шифрування. Надаючи коди та цифрові ключі, щоб переконатися, що отримане є справжнім і походить від передбачуваного відправника, одержувач може бути впевнений, що отримані дані не були підроблені під час передачі.

Криптографія також допомагає переконатися, що особистість відправника та одержувача, а також джерело або місце призначення інформації вірні, причому останнє є найважливішим.

Аутентифікація можлива лише через обмін спеціальними ключами, які відправник використовує для підтвердження своєї особи. Зазвичай це включає ім'я

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		15

користувача та пароль, але також може включати інші методи, такі як смарт-карта, сканування сітківки ока, розпізнавання голосу або сканування відбитків пальців.

У цьому контексті невідомість (властивість захищеності комп'ютерної системи) відноситься до підтвердження переданого повідомлення, яке або надіслано, або отримано. Цей принцип гарантує, що відправник не може заперечувати те, що він/вона надіслав дані. Він використовує цифрові підписи, щоб відправник не міг заперечувати походження даних [19].

Інформаційна безпека є однією з найбільших проблем для підприємств, що працюють на конкурентній основі у сучасному бізнес-середовищі. При використанні правильних стратегій криптографія допомагає захистити інтелектуальну власність користувача, не даючи їй стати жертвою кіберзагроз та зловмисників [15].

Алгоритми в одній системі шифрування можуть бути реалізовані двома способами: програмно або апаратно. Основними перевагами програмної реалізації алгоритму шифрування є: портативність, гнучкість та простота використання та модернізації. Це означає, що алгоритм, написаний мовою програмування (наприклад C/C++, C#, JAVA), може бути виконаний з невеликими змінами і без великих додаткових витрат на будь-якому комп'ютері. Також немає істотних обмежень при перенесенні коду з одного додатка до іншого.

Проте АНБ (Агентство національної безпеки США) дозволяє лише апаратне шифрування. Крім того, для військових та серйозних комерційних додатків як основний тип реалізації алгоритмів шифрування, як і раніше, використовується апаратне забезпечення. На це є кілька причин [19].

Перша причина – швидкість. Кожен алгоритм шифрування складається з багатьох складних бітових операцій. Оскільки ці операції не вбудовані в жодний універсальний комп'ютер, програмні реалізації криптографічних алгоритмів працюватимуть неефективно. Крім того, шифрування часто є мультиобчислювальним процесом, і покладання такого завдання на базовий процесор також є неефективним. Апаратне шифрування має на увазі шифрування даних на спеціальному чіпі всередині пристрою, що значно збільшує швидкість.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		16

Хоча деякі дослідники роблять спроби оптимізувати час виконання програмних рішень, результати завжди гірші, ніж у спеціалізованого обладнання.

Друга причина – безпека. Алгоритм шифрування, що працює на звичайному комп'ютері, не має фізичного захисту, що перешкоджає проникненню зловмисника в систему та внесення певних змін до алгоритму. Навпаки, пристрої апаратного шифрування можуть бути додатково фізично захищені - від способу зберігання таких пристроїв (зміщення, відеоспостереження і т. д.) до способу реалізації фізичного захисту на самому пристрої (наприклад, захист від несанкціонованого доступу). Мікросхеми HBIC спеціального призначення можуть бути покриті такою речовиною, що будь-яка спроба доступу до їх механізмів призведе до руйнування логіки мікросхеми (такими були мікросхеми американських урядових Clipper та Capstone). Таким чином, обладнання для захисту від вторгнень також може запобігти модифікації апаратного шифрування [15].

Третя причина апаратної переваги – простота установки. А саме, шифрування зв'язку між пристроями, що не є комп'ютерами загального призначення (телефон, факс, модем і т. д.), набагато простіше і дешевше реалізувати шляхом встановлення апаратного забезпечення для шифрування, ніж встановлення програмного забезпечення і певного мікропроцесора.

З огляду на це цілком зрозуміло, що для реалізації у високопродуктивних системах буде використовуватися апаратна реалізація криптоалгоритмів, а не програмна. Це щодня підтверджується на практиці багатьма провідними організаціями, які використовують такий підхід до захисту інформації у своїх продуктах (криптовлаштуваннях).

Завдяки ряду переваг застосування програмованих мікросхем є однією з найвдячніших технологій апаратної реалізації криптологічних алгоритмів.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		17

Symmetric Encryption

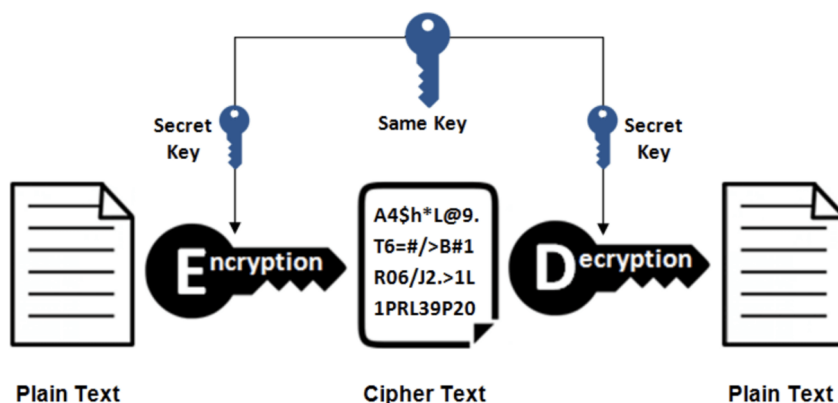


Рисунок 1.2 – Демонстрація роботи базового шифрування

Відмінною особливістю симетричної криптографії є використання одного і того ж ключа (тому симетричного) для шифрування та дешифрування. Таким чином, ключ являє собою спільну таємницю між двома (або більше) сторонами, які бажають спілкуватися.

Симетричні схеми зазвичай називають шифрами. Перший відомий шифр датується римлянами: є свідчення того, що Юлій Цезар використовував цей метод для спілкування зі своїми полководцями, тому схему зазвичай називають «шифром Цезаря». Він полягає в простому зміщенні букв у повідомленні на певну кількість позицій.

Caesar cipher ("key" = 3)

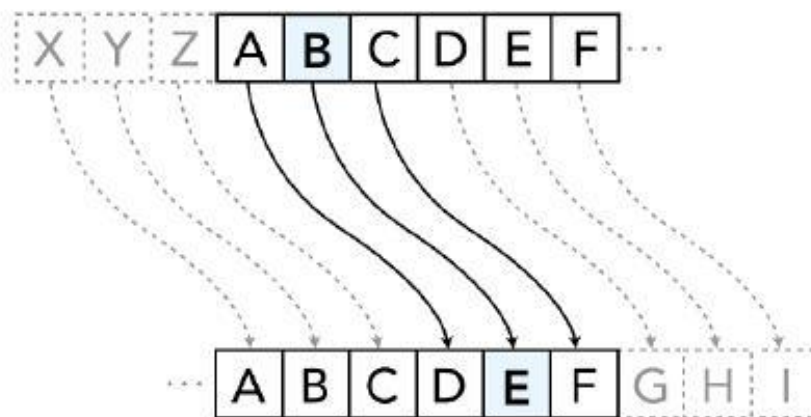


Рисунок 1.3 – Приклад роботи шифру Цезаря

		Сірик А. Ю.		
		Байлюк Є. М.		
Змн.	Арк.	№ докум.	Підпис	Дата

КІК.КР.Б – 125 – 22 – ПЗ

Арк.

18

Сучасні шифри поділяються на два сімейства: потокові шифри і блокові шифри. Схеми в першому сімействі шифрують біти повідомлення по одному, тоді як блочні шифри, як випливає з назви, беруть певну кількість бітів і шифрують їх як єдине ціле, доповнюючи відкритий текст так, щоб він був кратним розмір блоку. Прикладом є дуже відомий AES, який використовує блоки розміру 128, 192 або 256. Блочні шифри працюють у різних режимах, залежно від того, чи використовується алгоритм шифрування «як є» (EBC), використовуючи «вектор ініціалізації» (CBC і CBCC) або випадкова «початкова точка» (CTR і CTRC) [19].

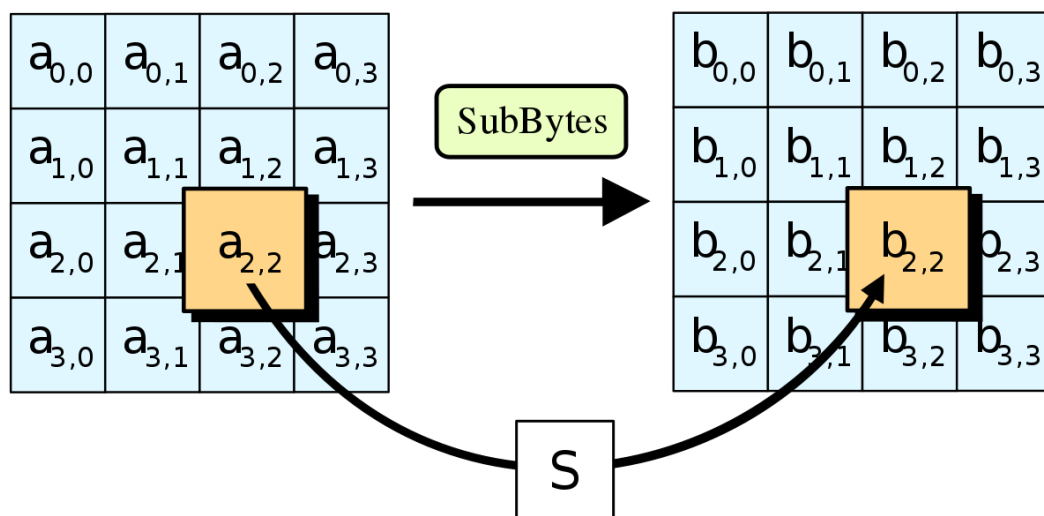


Рисунок 1.3 – Приклад роботи методу шифрування AES

Слово, пов'язане з таємницею, «криптографія» має напрочуд чітку етимологію. Слово походить від грецьких коренів криптос (грец. $\kappa\rho\upsilon\pi\tau\acute{o}\varsigma$), що означає «прихований», і graphein (грец. $\Gamma\rho\acute{\alpha}\mu\mu\alpha$), що означає «писати». Криптографія як наука має на меті вирішення двох основних завдань, це забезпечення конфіденційності та цілісності інформації. Ці проблеми вирішуються за допомогою різноманітних методів, призначених для трансформації даних, щоб вони не мали цінності для зломисників. В даний час в якості основних можна виділити наступні криптографічні методи:

- стенографії;
- кодування;
- стиснення;
- шифрування.

В Інтернеті шифрування є важливим інструментом для захисту інформації. Шифрування досягає наступної мети: надавати інформацію в хаотичному вигляді, нечитабельний набір символів, букв і цифр. Однак цей процес має протилежний ефект, якщо в оператора є: алгоритм перетворення, ключ дешифрування та вихідні дані. Аббревіатура дещо відрізняється від інших методів криптографії, оскільки дозволяє зберегти в таємниці не тільки значення, а й сам факт передачі та зберігання секретної інформації. Хеш - це метод маскування приватних файлів даних серед відкритих файлів. У результаті приховування секретних даних і створення реалістичних даних стає неможливо відрізнити їх. При цьому прихований файл також можна зашифрувати, і якщо хтось його знайде, то сприйме це як збій системи. Спільне використання цих методів: шифрування + скорочення, значно знижує ризик виявлення та «читання» секретної інформації. Інший метод, кодування, перетворює дані в код. У цьому випадку код може бути представлений цифрами, символами або комбінаціями букв. При використанні цього методу криптографії знадобляться спеціальні словники і таблиці. У наш час інформаційні мережі також використовують кодування, перетворюючи початкове повідомлення за допомогою програмного та апаратного забезпечення, в процесі чого підвищується достовірність інформації, яку ми хочемо передати. Кодування та шифрування — це два різні методи, хоча можна не помітити цього на перший погляд і часто плутаєте. Відмінність полягає в тому, що якщо необхідно відновити інформацію, яка була зашифрована, оператор повинен мати алгоритм заміни. А при використанні шифрування нам потрібно буде знати правила шифру і самого ключа. Третій метод перерахований не у всіх джерелах, і, можливо, його додавання є спірним, але все ще потребує розгляду, оскільки він часто використовується разом із шифруванням. Мова йде про стиснення інформації, метою якого є зменшення кількості інформації (даних), яка зайнята. У цьому випадку доступ до інформації можна отримати лише шляхом зворотного перетворення.

Нині стиснення та зворотне перетворення надзвичайно доступні, навіть якщо алгоритм є секретним. Вони мають досить високу ймовірність виявити себе методом відбору та на основі зібраної статистики. Тому наступним кроком у

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		20

стисканні конфіденційної інформації є її шифрування. Стиснення в цьому випадку економить наш час при передачі даних. Захист інформації завжди був важливою метою, про що свідчать багато джерел, деякі з них. 4 тисячі років. У стародавні часи криптографія використовувалася для збереження повідомлень у таємниці після того, як з'явилися листи. Хоча сама писемність винайдена, вона певною мірою представляла криптографію - через малу кількість освічених людей. Але з часом це коло почало розширюватися. У стародавньому світі існували досить примітивні методи, які зараз можна опосередковано віднести до криптографії.

Наприклад, у Стародавньому Єгипті 4 тисячі років тому використовувалися спеціальні ієрогліфи (це перша згадка в історії). Але перед єгиптянами тоді стояло завдання не приховувати інформацію чи ускладнювати читання, а змагатися в красномовстві та винахідливості. Це був спосіб привернути увагу до текстів. Ще одна згадка — винахід пристрою під назвою Scitalla. Один із найстаріших криптографічних механізмів, відомих у світі (близько 400 р. до н.е.), він також є «стародавнім шифром Спарти». Це була палиця (паличка), на яку треба було намотати пергамент у вигляді стрічки. Таким чином, згорнувши стрічку в спіраль, вони отримали місце для письма. Текст був написаний по всій довжині палиці, після чого пергамент знімали і вручали одержувачу. Текст розгорнутої стрічки був нечитабельним. Щоб прочитати повідомлення, знадобився стрижень, який відповідав би оригінальному діаметру.



Рисунок 1.4 – Пристрій Scitalla

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

Пізніше Аристотель відкрив шифр Скаллала, взявши за основу конус, намотавши пергамент з текстом, змінивши висоту. Таким чином підбираємо необхідний діаметр. Після цього повідомлення стало доступним для читання. Мабуть, найяскравішим прикладом криптографії стародавнього світу є шифр Цезаря. Це досягнення римської криптографії спиралося на певний метод заміни, при якому кожен символ у тексті замінюється символом на постійній відстані від нього. І вправо, і вліво.

Етап сучасної криптографії припав на промислову революцію, її сміливо можна назвати епохою шифрувальних машин. Криптографію не обійшла автоматизація; для шифрування використовувалися ротаційні криптосистеми. Одним з найвизначніших винаходів сучасності (близько 1790 р.) була механічна машина Томаса Джефферсона, також відома як циліндр Джефферсона. А в 1917 році Едвард Хеберн створив Enigma, яка пізніше була доопрацьована Артуром Кірхом. Роторні системи активно використовувалися під час Другої світової війни, що дозволило створити дуже міцні шифри.

Винахід комп'ютера та створення доступного Інтернету зробили криптографію більш доступною. Наприклад, це включає стільниковий зв'язок, захист від спаму в електронній поштовій скриньці, банківські операції, цифрове телебачення та багато іншого. Його почали використовувати фізичні особи в електронних комерційних і господарських операціях. В результаті ми маємо валюту під назвою біткойн, ця валюта не контролюється державою.

Блокчейн — це ланцюжок блоків

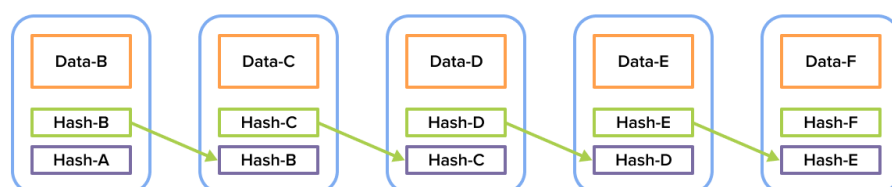


Рисунок 1.4 – Базовий принцип роботи блокчейну

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

Розроблено технології криптовалютних транзакцій – блокчейн. Можна сказати, що одним з новітніх застосувань криптографії є коди Data Matrix. Так борються з контрафактною продукцією шляхом маркування товарів. Ці коди складаються з ідентифікаційного коду та криптовалюти. Отже, ми маємо систему тегів, яка генерує код. Він унікальний, містить дані про товар, визначає позицію продукту в системі та єдиному каталозі, а криптохвости додатково шифрують кожен код у виробництві. Оскільки код не повністю збережений в системі, його неможливо отримати. Крім того, система не буде поновлювати його протягом щонайменше 5 років з дати видачі титула.

Швидкий розвиток технологій у сфері комп'ютерів призвів до того, що криптографія перейшла до складних алгоритмів шифрування. А з появою мобільного зв'язку, пристроїв та Інтернету перенесла це в цивільне поле. Інша сторона криптографії, яку я хотів би розглянути, — це юридична, а саме державне регулювання. З точки зору регулювання, є дві зацікавлені сторони – та сама держава, яка хоче зберегти таємницю, і борці за громадянські права, комерційні структури. Правозахисники спираються на 12-ту Загальну декларацію прав людини (ООН 1948): «Ніхто не може бути підданий свавільному втручанню в його приватне та сімейне життя, свавільним нападам на недоторканність його житла, таємницю його листування чи його честь і репутація. Кожен має право на захист від такого втручання або нападів». Наразі саме шифрування залишається основним гарантом права громадян на приватність. З боку бізнесу — зацікавленість у шифруванні онлайн-сфери, розвитку електронної комерції, а отже — шанс на успішний бізнес, не боячись, що дані постачальників будуть у відкритому доступі або потраплять у чужі руки [19].

На закінчення можна з упевненістю сказати, що раніше криптографія використовувалася політиками, військовими та дипломатами, створена вченими, які вручну засекречували дані протягом багатьох століть. Так, звичайно, в наш час існували шифрувальні машини, але все ж сфера криптографії була обмежена найближчим колом.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

1.3 Використання документообігу в Україні

Серед платних послуг в Україні найбільш популярними є сервіси електронного документообігу: М.Е.Doc, ВЧАСНО, Document.Online, Art-Office, Comarch EDI, FossDoc та ще близько 10 продуктів.

Як правило, всі послуги доступні цілодобово без вихідних, тобто в будь-який час. Вони також підтримують достатній рівень безпеки щодо конфіденційності та збереження даних, а також можуть надавати кваліфіковану технічну підтримку. Деякі пропонують можливість розширити функціонал сервісу відповідно до індивідуальних потреб клієнта.

Ринок програмного забезпечення в Україні пропонує компаніям різноманітні програмні рішення для впровадження або вдосконалення електронного документообігу – як внутрішнього, так і зовнішнього. Створені безкоштовні сервіси, які враховують усі вимоги українського законодавства.

До них відноситься Paperless - сервіс для компаній і підприємців, що дозволяє обмінюватися документами один з одним в електронному вигляді. Розпочати роботу легко: достатньо створити особистий кабінет та електронний цифровий підпис. Користуючись сервісом «Безпаперовий», юридична особа може обмінюватися бухгалтерськими та юридичними документами з контрагентами, а саме: договорами, рахунками, виконаними роботами, рахунками-фактурами тощо.

Стаття 9 Закону України «Про електронні документи та електронний документообіг» містить таке визначення електронного документообігу (електронного документообігу) - сукупності процесів створення, обробки, відправлення, відправлення, отримання, зберігання, використання та знищення електронних документів, які виконуються із застосуванням перевірки цілісності та, за необхідності, підтвердження отримання таких документів» [1].

Основним суб'єктом електронного документообігу є електронний документ – «документ, в якому зберігається інформація у вигляді електронних даних, у тому числі обов'язкові реквізити документа» [1].

Основними цілями впровадження електронного документообігу в організації є:

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

- 1) підвищення ефективності управління бізнесом;
- 2) прискорення документообігу в організації;
- 3) зниження трудомісткості обробки документів.

Електронний документообіг дозволяє створити в організації єдиний інформаційний простір, інтегруючи всі системи документації в інформаційний вузол. Інтеграція відбувається без втрати якості роботи з документами, зберігаючи традиції українського діловодства. Основою такої інтеграції є надійне зберігання документів та взаємодіючі системи документообігу. Усі опрацьовані документи зберігаються в одному сховищі, що дозволяє здійснювати оптимальний пошук та відбір інформації при підготовці матеріалів [2].

Обмін електронними документами між контрагентами та партнерами можливий лише за умови технічної відповідності електронних документів, тобто за умови використання однієї служби документообігу.

Для вирішення цієї проблеми була створена спеціальна стандартизована платформа електронного документообміну «ПТАХ». Ця платформа відкрита для всіх програмістів та користувачів і дозволяє інтегрувати передані дані з електронними системами компаній за умови підключення та використання платформи.

Однак для більшої зручності перед запровадженням електронного документообігу краще запитати у контрагентів, чим вони користуються, і вибрати серед них найпопулярніший сервіс. Для решти можна використовувати "ПТАХ"

З технічної точки зору обмінюватися документами з державою набагато простіше. Багато сервісів надають можливість відправити звіт до податкової.

Document. Online service.

Один з популярний сучасний сервіс електронного документообігу. Сервіс підтримує співпрацю з усіма акредитованими центрами сертифікації ключів України та позиціонується як єдина платформа внутрішнього та зовнішнього документообігу. Можна використовувати його з будь-якого пристрою - стаціонарних і мобільних пристроїв з усього світу. Сервіс Document.Online

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		25

сумісний з будь-якою операційною системою, встановленою на пристрої користувача.



Рисунок 1.5 – Логотип Document.Online

Система дозволяє завантажувати та створювати електронні документи з готових шаблонів. Крім того, є можливість попереднього перегляду завантажених, створених та підписаних документів з інформацією про підписанта.

Додатково є функціонал обміну документами та встановлення маршрутів документації, що дозволяє реалізувати бізнес-процес приймання та підписання документа як всередині компанії, так і з контрагентами, а також повідомляти інших співробітників про погодження та підписання.

Document.Online також пропонує систему контролю електронного документообігу: зміна статусу документа після дії, повідомлення про завершення маршруту документа тощо.

Передбачена інтеграція з BAS (1С для України) та іншими популярними системами обліку.

Захист документів забезпечений Microsoft. Зберігаються документи на серверах Microsoft Azure.

Між серверами Microsoft Azure автоматично підтримується зв'язок. Завдяки багаторазовому копіюванню даних ймовірність втрати даних в разі виходу з ладу одного з серверів Microsoft незначна. Навіть в разі фізичного зруйнування носія дані будуть відновлені протягом години.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

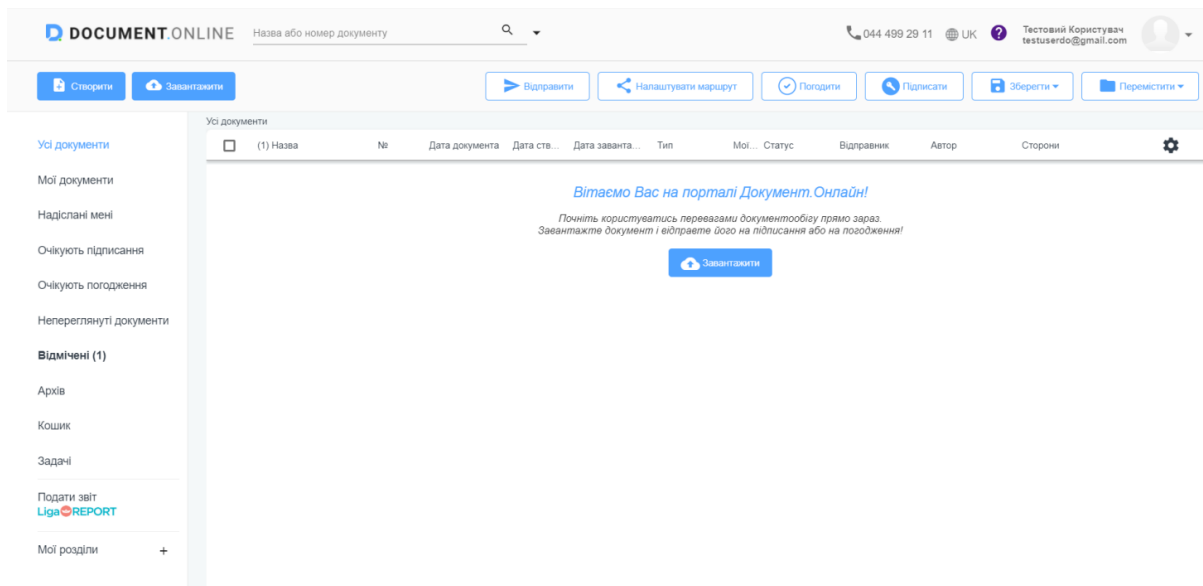


Рисунок 1.6 – Інтерфейс додатку Document.Online

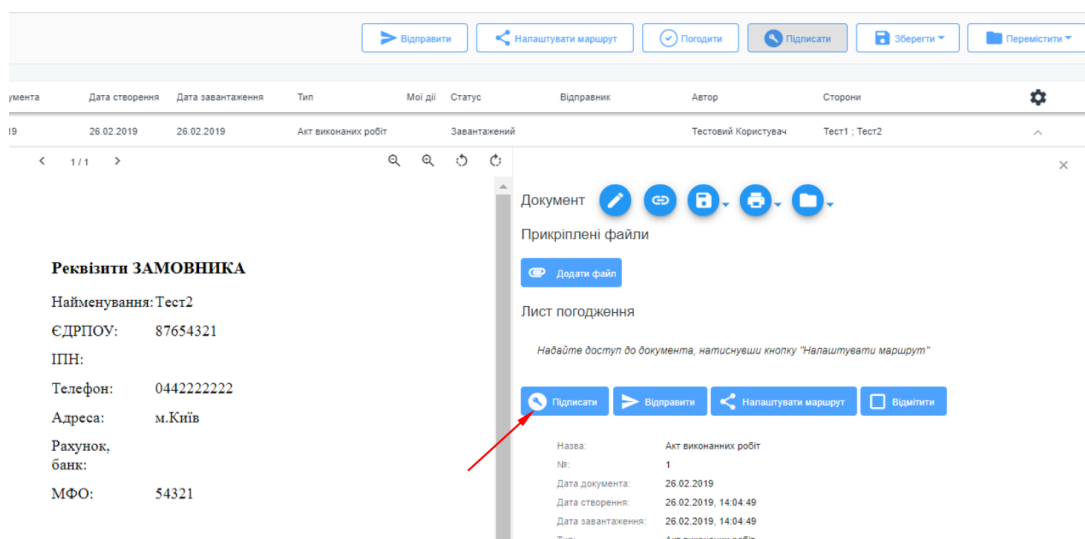


Рисунок 1.7 – Приклад підписання документу у Document.Online

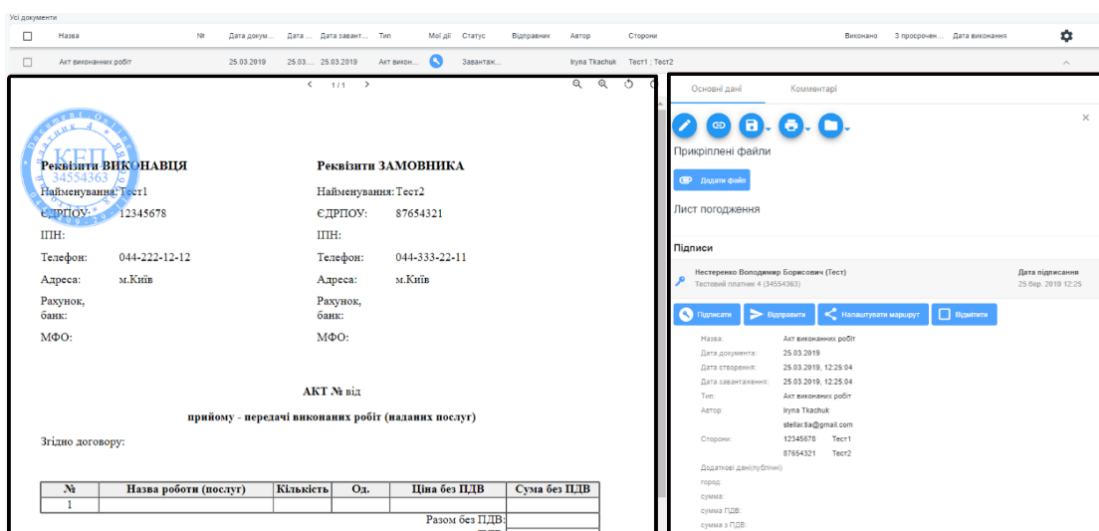


Рисунок 1.8 – Перегляд документу у Document.Online

		Сірик А. Ю.			KIK.KP.B – 125 – 22 – ПЗ	Арк.
		Байлюк С. М.				27
Змн.	Арк.	№ докум.	Підпис	Дата		

My Electronic Document.

М.Е.Дос (My Electronic Document) - програмне забезпечення для бухгалтерів та підприємців, яке робить процес надсилання звітності до державних органів та обміну електронними документами з партнерами простим, безпечним та швидким. Даний додаток є найпопулярнішим в Україні для роботи з електронними документами. 81% компаній в Україні вже обрали М.Е.Дос. Оперативно реагуючи на кожен новий крок в законодавстві, розробник надав зручний і простий механізм подання звітності, а пізніше – і реєстрації податкових накладних в електронному вигляді. Завдяки М.Е.Дос можна складати, редагувати і зберігати будь-яку необхідну документацію. Звітність та ведення документообігу з клієнтами реалізовано в одній програмі.



Рисунок 1.9 – Логотип М.Е.Дос

М.Е.Дос - провідний український програмний продукт, в якому реалізовано можливості не лише для створення документів та підписання засобами електронного підпису, але й для передачі документів із можливістю погодження або відхилення. Отже, М.Е.Дос - програма електронного документообігу повного циклу. Дана система документообігу є унікальною і універсальною. Це найкращий помічник у роботі з усіма типами документів в електронному вигляді: податкові накладні, акти, рахунки і звіти. Це програмний продукт для підприємств різних сфер діяльності та систем оподаткування.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				28
Змн.	Арк.	№ докум.	Підпис	Дата		

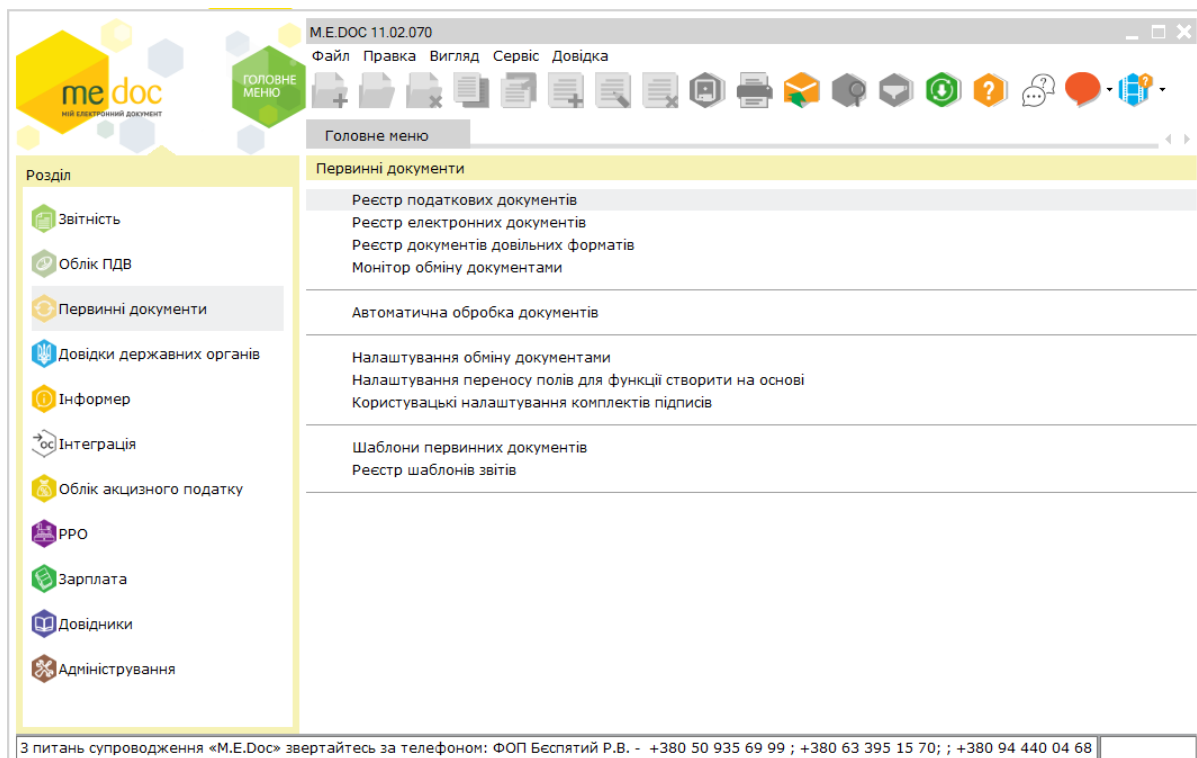


Рисунок 1.10 – Інтерфейс додатку М.Е.Doc

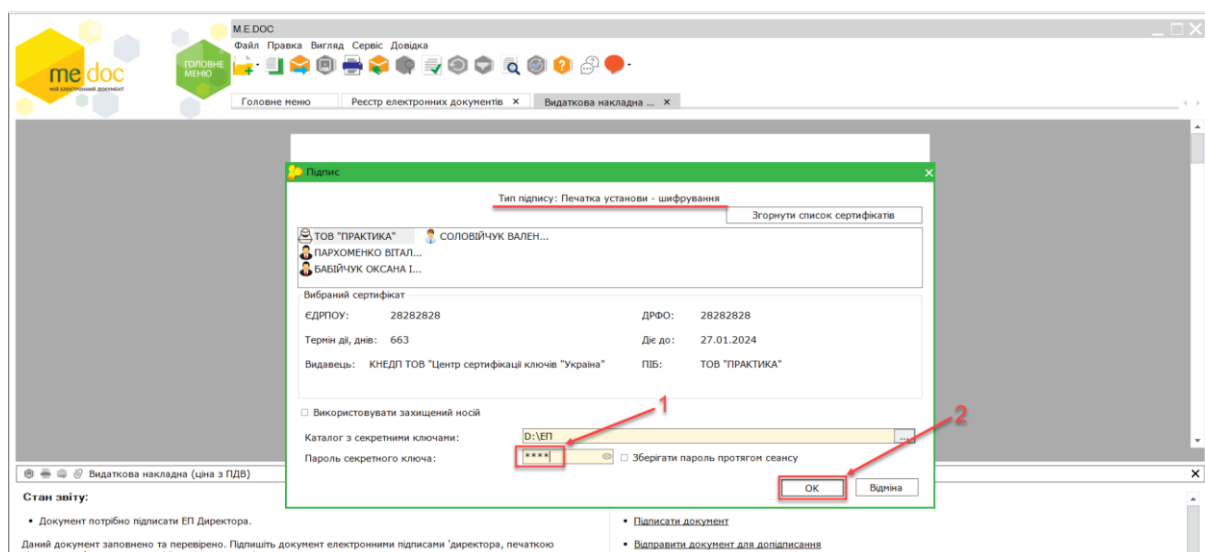


Рисунок 1.11 – Приклад підписання документу у М.Е.Doc

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		29

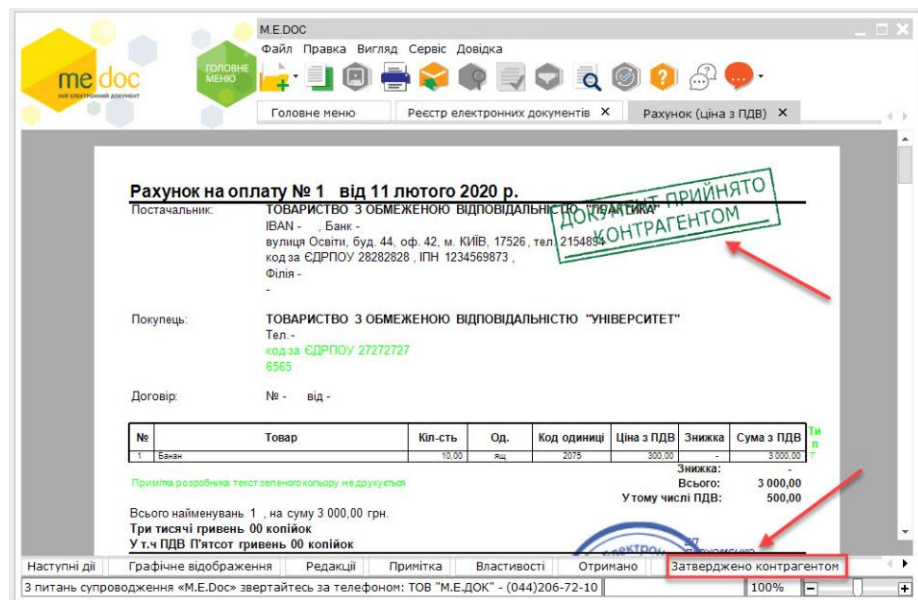


Рисунок 1.12 – Перегляд документу у М.Е.Док

1.4 Постановка технічного завдання

Ціллю проекту – є розробка програмного забезпечення для безпечної передачі документів між організаціями.

Основна форма повинна містити: перемикач (відправити/одержати документ), поля для повної адресації (організація та користувач), поле вибору файлу для відправки або теки для збереження отриманих файлів.

Передача документів має бути захищеною протоколом TLS (наприклад: використання протоколу SFTP). Сам документ повинен бути зашифрованим асиметричним шифруванням.

У додатку має бути базова аутентифікація, зв'язана з базою даних. Паролі користувачів мають зберігатись у зашифрованому вигляді, тобто після хешування.

При реєстрації мають бути згенеровані ключі для шифрування та дешифрування. Публічний ключ занесений до бази даних, а приватний ключ збережений на комп'ютері.

Програма повинна мати можливість адміністрування, з розділенням на ролі (користувач та адміністратор). Адміністратор повинен мати можливість видаляти або додавати користувачів, змінювати паролі, переглядати електронну бібліотеку по всім користувачам. Звичайний користувач може тільки змінювати свій пароль та переглядати історію відправки, отримання своїх документів.

		Сірик А. Ю.			KIK.KP.B – 125 – 22 – ПЗ	Арк.
		Байлюк С. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		30

Необхідна функція при отриманні документів, щоб була можливість отримати всі документи або з певним відбором.

Також потрібно розробити шаблон для документу в .xls форматі. А отримання документу у .pdf форматі.

Висновок до розділу 1.

Отже у розділі було розглянуто важливість використання електронного документообігу в бізнес-процесах, проаналізовано користь та безпечність даного методу в порівнянні з традиційним.

Описано види безпечного транспортування даних у інформаційних мережах, порівняно типи шифрування.

Продемонстровано використання документообігу в Україні та порівняно найпопулярніші сервіси.

А також створено технічне завдання для майбутнього проекту.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		31

РОЗДІЛ 2 АНАЛІЗ ТЕХНОЛОГІЙ ПРОЄКТУВАННЯ ПІДСИСТЕМИ ЗАХИЩЕНОГО ОБМІНУ ДОКУМЕНТАМИ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДОКУМЕНТООБОРОТУ

2.1 Мова програмування C#

Мова програмування – це штучна мова, призначена для вираження обчислень, які може виконувати машина, зокрема комп'ютер. Мови програмування розвивалися дуже швидко з початку 1950-х років, що призвело до винайдення понад сотень різних мов програмування. Завдяки швидкому технологічному розвитку апаратного забезпечення з швидшими процесорами, все більш потужні мови програмування можуть використовуватися для задоволення потреб розробки більш ефективних програм для різних додатків [8].

Перше покоління мов програмування – це машинні мови, які з'явилися на початку 1950-х років. Він був записаний у двійковій формі, серії нулів і одиниць. Бінарний код важко зрозуміти людині і він дуже схильний до помилок. Основною проблемою машинних мов є машинна залежність, оскільки машинні мови створювалися по-різному для різних для кожного ЦП [13].

Друге покоління мов програмування – це символічні мови програмування асемблера, які були «написані в більш спрощеній формі і на більш високому рівні абстрактних машинних мов, замість ряду нулів і одиниць були символи (відсоток, долар і частини слова). і комбінації чисел, які використовуються для створення команд. Але символічні мови програмування на асамблеї все ще мають проблеми високої апаратної залежності та недостатньої переносимості, що означає, що коди асемблера не можуть бути реалізовані на різних машинах [8].

Період з початку 1960-х до 1980-х років приніс нам мови програмування третього покоління. Третє покоління мов програмування називають проблемно-орієнтованими мовами і також вважаються мовами високого рівня. Мови програмування третього покоління були перетворені з англійської на машинні мови; для перетворення цих інструкцій використовувалися компілятори. C, C++, C# і Java – це приклади мов програмування третього покоління. Більшість третіх

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

поколінь мов програмування мають компілятори або інтерпретатори. Перевага цього полягає в тому, що програми можуть працювати дуже швидко після компіляції. Проблема «машинної залежності», з якою стикалися перше і друге покоління мов програмування, більше не стосується третього покоління мов програмування. Основна проблема мов програмування третього покоління полягає в тому, що для різних типів процесорів потрібні різні вихідні коди мов програмування третього покоління, а мови програмування третього покоління досить важко розробити [3].

Мови програмування четвертого покоління більше орієнтовані на вирішення проблем. Основна відмінність між іншими мовами програмування покоління полягає в тому, що вони більше піклуються про те, що потрібно зробити, ніж про те, як. Особливості, очевидні в мовах четвертого покоління, полягають у тому, що він повинен бути зручним для користувача, портативним і незалежним від операційних систем, використовуватися не програмістами, мати інтелектуальні параметри за замовчуванням щодо того, що хоче користувач і дозволяти користувачеві швидко отримувати результати, використовуючи код мінімальних вимог. Найбільш відомими прикладами мов запитів четвертого покоління є SQL, MYSQL [16]. Мова програмування п'ятого покоління (5GL) — це мова програмування, заснована на розв'язанні проблем із застосуванням обмежень, заданих програмі, а не на основі алгоритму, написаного програмістом. Використовуючи мови програмування п'ятого покоління, комп'ютери можуть мати власну здатність мислити, а їхні власні висновки можна робити, використовуючи запрограмовану інформацію у великих базах даних. Мрія про роботу зі штучним інтелектом і нечіткою логікою здійснилася завдяки використанню мов програмування цього покоління [5].

C – це мова програмування, яка народилася в «AT & T's Bell Laboratories» у США в 1972 році. Її написав Денніс Річі. Ця мова була створена для конкретної мети: для розробки операційної системи UNIX (яка використовується на багатьох комп'ютерах). З самого початку C мав на меті бути корисним — дозволяти зайнятим програмістам виконувати завдання. Після цього C почали

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

використовувати все більше й більше людей за межами Bell Laboratories, оскільки він ефективніший за інші мови програмування того часу. Наприкінці 70-х років С зайняв домінуючу позицію серед мов програмування. Комітет, сформований Американським національним інститутом стандартів (ANSI), у 1989 році затвердив версію С, відому як ANSIC. За деякими винятками, кожен сучасний компілятор С має можливість дотримуватися цього стандарту. ANSI С був затверджений Міжнародною організацією стандартів (ISO) у 1990 році. У назві С є щось цікаве. Його назвали С, тому що його попередник називався В, який також був розроблений Кеном Томпсоном з Bell Labs [17].

Основними архітекторами С# були Пітер Голд, Ерік Ганнерсон, Андерс Хейлсберг, Пітер Соллічі та Скотт Вільтамут. Звичайно, головним розробником мови С# був Андерс Хейлсберг, провідний архітектор Microsoft. С# був розроблений як чиста об'єктно-орієнтована мова програмування. С# дебютував у 2000 році на конференції професійних розробників (PDC), де основним доповідачем був засновник Microsoft Білл Гейтс. Тоді ж було анонсовано Visual Studio .NET [13].

С# – це проста, сучасна й об'єктно-орієнтована мова програмування. Метою С# була розробка мови програмування, яку не тільки легко вивчати, але й підтримувати сучасну функціональність для всіх видів розробки програмного забезпечення.

Якщо подивитись на історію мов програмування та їх характеристики, кожна мова програмування була розроблена з певною метою, щоб вирішити конкретну потребу того часу.

Проте С# був розроблений з урахуванням потреб бізнесу та підприємств. С# розроблено для компаній, які розробляють усі види програмного забезпечення з використанням єдиної мови програмування.

С# надає функціональні можливості для підтримки розробки сучасного програмного забезпечення. С# відповідає потребам розробки веб-, мобільних пристроїв і додатків. Деякі з функцій сучасної мови програмування, які підтримує С#, включають генерики, типи var, автоматичну ініціалізацію типів і колекцій,

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		34

динамічне програмування, асинхронне програмування, кортежи, зіставлення шаблонів, розширене налагодження та обробку винятків тощо [16].

На синтаксис C# впливають C++, Java, Pascal та кілька інших мов, які легко прийняти. C# також уникає складності та неструктурованих мовних функцій.

C# є типобезпечною мовою. C# не дозволяє перетворення типів, які можуть призвести до втрати даних або інших проблем. C# дозволяє розробникам писати безпечний код. C# також фокусується на написанні ефективного коду [6].

Хоча більшість мов програмування були розроблені для певної мети, C# був розроблений для C#. Ми можемо використовувати C# для створення сучасних програмних додатків. C# можна використовувати для розробки всіх видів додатків, включаючи клієнтські програми Windows, компоненти та бібліотеки, служби та API, веб-додатки, мобільні додатки, хмарні програми та відеоігри.

Керування пам'яттю вручну вимагає від розробників керувати виділенням і зняттям блоків пам'яті. Керування пам'яттю вручну займає багато часу і складно. C# забезпечує автоматичне керування пам'яттю, щоб розробники звільнялися від цього обтяжливого завдання. У переважній більшості випадків таке автоматичне керування пам'яттю підвищує якість коду та підвищує продуктивність розробника без негативного впливу ні на виразність, ні на продуктивність [6].

WPF, що розшифровується як Windows Presentation Foundation, є останнім підходом Microsoft до фреймворку GUI, який використовується разом із платформою .NET.

GUI розшифровується як графічний інтерфейс користувача. Windows має графічний інтерфейс для роботи з комп'ютером, а браузер має графічний інтерфейс, який дозволяє переглядати веб-сторінки.

Фреймворк графічного інтерфейсу дозволяє створювати програму з широким набором елементів графічного інтерфейсу, як-от мітки, текстові поля та інші добре відомі елементи. Без рамки графічного інтерфейсу довелося б малювати ці елементи вручну та обробляти всі сценарії взаємодії з користувачем, як-от текст і введення мишею. Це багато роботи, тому замість цього більшість розробників

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк С. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		35

використовуватимуть фреймворк GUI, який виконає всю основну роботу і дозволить розробникам зосередитися на створенні чудових програм.

Існує багато фреймворків GUI, але для розробників .NET найцікавішими на даний момент є WinForms і WPF. WPF є найновішим, але Microsoft все ще підтримує і підтримує WinForms.

Однією з важливих особливостей є використання розмітки XAML на основі декларативного XML: ви можете створити розширений графічний інтерфейс, використовуючи декларативні або керовані оголошення коду C#, VB.NET і F# або їх комбінацію.

2.2 Платформа .net framework

.NET Framework – це середовище розробки та виконання, яке дає змогу різноманітним мовам програмування та бібліотекам безперешкодно працювати разом для створення програм на базі Windows, які легше створювати, керувати, розгортати та інтегрувати з іншими мережевими системами. Це основа, яка підтримує вимоги до виконання всіх програм .Net. .NET Framework складається з двох основних компонентів:

1. Середовище виконання загальної мови (CLR) CLR – це незалежний від мови середовище розробки та виконання, що надає послуги, що допомагають «керувати» виконанням програми. CLR надає основні послуги, такі як управління пам'яттю та потоками. Підтримує код, написаний для .Net, і забезпечує безпеку типів.

2. Бібліотеки класів Framework (FCL) FCL – це узгоджена об'єктно-орієнтована бібліотека, що містить готові пакети функцій. Ця бібліотека функцій надається платформою .Net для розробки програм CLR. Бібліотека функцій складається з об'єктно-орієнтованих класів, які спрощують завдання програмування та прискорюють час програмування. Бібліотека класів підтримує програми розробника, які запускаються як програми командного рядка та програми на базі Windows.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

Керування кодом є ядром CLR, а сама CLR є ядром .Net Framework. Кожна мова програмування, призначена для середовища CLR, керується кодом. Модель безпеки та розробки, що використовується платформою .NET, означає, що багато програм, які раніше потрібно було встановити локально на комп'ютері, тепер можна розгортати через Інтернет. Програма може запускати компоненти від різних постачальників і з різних веб-сайтів, та кожен компонент матиме власну безпеку та дозволи на локальному комп'ютері. В результаті більше не доведеться чекати нескінченні місяці, поки постачальник надасть пакет оновлень або оновлення програми для відновлення або покращення існуючих програм. Це пояснюється тим, що зі зрозумілих причин компоненти на веб-сайтах можна оновлювати швидше, а впровадження буде значно спрощено. Середовище .Net дозволяє розробляти багато програм, наприклад консольні програми. Такі програми мають перевагу в тому, що вони є утилітами командного рядка, які можуть отримати доступ до графічного інтерфейсу користувача (GUI) Windows. Середовище надає набір повторно використовуваних об'єктів графічного інтерфейсу в класах Windows Forms. Він також надає подібний набір об'єктів, які входять до класів веб-форм Windows для створення веб-програм [14].

CLR - Common Language Runtime Developers.

В минулому поєднували свій код з різними бібліотеками і використовували зовнішні компоненти для своїх програм. На додаток до бібліотеки часу виконання доступно багато спеціалізованих бібліотек коду. Бібліотеки динамічних посилань (DLL) надають ще одне джерело коду під час виконання. Наприклад, код інтерфейсу прикладного програмування Windows (API) міститься в кількох DLL, які програма може викликати за потреби. Однак для мови програмування C# платформа .Net визначає понад 1000 класів, до яких розробник може отримати прямий доступ або використовувати як шаблон для створення нових класів, щоб розширити функціональність, надану цими класами. CLR — це середовище, яке використовується фреймворком .Net. Керує кодом під час виконання та надає послуги, які допомагають створювати програми. Код, написаний для CLR, називається «керованим» кодом. За словами Microsoft, процес виконання коду в

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		37

CLR схожий на виконання «угоди про співпрацю» з середовищем виконання. Керований код проходить процес "перевірки", перш ніж CLR дозволить йому запуснитися на комп'ютері. CLR забезпечує безпечне середовище, де програма не викликає помилку пам'яті, яка намагається отримати доступ до пам'яті, яка їй не належить. Це поширена проблема для програмістів C++, які іноді можуть забути ініціалізувати покажчик перед спробою отримати доступ до пам'яті. C# також певним чином використовує покажчики, але приховує їх від програміста як змінні типу посилання. Програма просто не дозволить використовувати змінну-індикатор або будь-яку звичайну змінну без присвоєного значення. Однак процес кодування C# не є ідеальним і все ще може викликати помилки під час виконання. CLR спостерігає за кодом і фіксує ці екземпляри. Коли це відбувається, програма викидає виняток через CLR. Програміст може вирішити включити користувацький код у програму, щоб стежити за цими «винятками» та ловити їх у самій програмі. Якщо не оброблено належним чином або не оброблено взагалі, CLR завершить програму. Розробники визнають, що CLR набагато гнучкіший з точки зору взаємодії коду мови програмування. Це ще одна дуже хороша особливість фреймворку .Net і мови програмування C# [14].

MSIL – Microsoft Intermediate Language.

C# перетворює файли вихідного коду в IL (проміжна мова), який теоретично можна використовувати для запуску однієї програми в різних операційних системах і на різних обчислювальних платформах. Коли програма виконується, середовище виконання надсилає цю інформацію, яка є IL, або в даному випадку MSIL, компілятору «точно вчасно», який, у свою чергу, перетворює код у код, властивий комп'ютеру, на якому виконується програма. Корпорація Майкрософт використовує функцію «точно вчасно», щоб вказати, що дії вживаються лише за необхідності.

Справжня об'єктно-орієнтована природа додатків .Net Основною зміною порівняно з попередніми програмами ASP є те, що .Net пропонує повністю об'єктно-орієнтоване середовище розробки та розробки додатків. Усі програми .Net повністю об'єктно-орієнтовані. Весь код знаходиться в якомусь класі. Усе є

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк С. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

об'єктом, навіть прості типи даних. Повна підтримка статичних властивостей і властивостей екземплярів, статичних методів і методів екземпляра, подій, віртуальних функцій, абстрактних класів, поліморфізму, приховування та успадкування даних [14].

	Версія .NET Framework												
ОС	1.0	1.1	2.0	3.0	3.5	4.0	4.5	4.5.1	4.5.2	4.6	4.6.1	4.6.2	
Windows 98	+												
Windows NT	+												
Windows Me	+												
Windows 2000	+	+	+										
Windows XP	+	+	+	+	+	+							
Windows Server 2003		+	+	+	+								
Windows Server 2008			+	+	+	+	+	+	+	+			
Windows Vista			+	+	+	+	+	+	+	+			
Windows 7			+	+	+	+	+	+	+	+	+	+	
Windows Server 2008 R2			+	+	+	+	+	+	+	+	+	+	
Windows Server 2012			+	+	+	+	+	+	+	+	+	+	
Windows 8			+	+	+	+	+	+	+	+	+	+	
Windows 8.1			+	+	+	+		+	+	+	+	+	
Windows Server 2012 R2			+	+	+	+	+	+	+	+	+	+	
Windows 10			+	+	+	+	+	+	+	+	+	+	

Рисунок 2.1 – Відповідність підтримуваних версій .NET Framework у Windows

2.3 Метод асиметричного шифрування RSA

Криптографія вважається як розділом математики, так і розділом теоретичних обчислень. Шифрування та цифровий підпис — це криптографічні методи, які використовуються для реалізації служб безпеки. Основний елемент, який використовується, називається системою шифрування або алгоритмом шифрування. Кожна система шифрування включає в себе пару перетворень даних, які називаються шифруванням і дешифруванням. В асиметричних алгоритмах, тобто алгоритмах відкритого ключа, ключ шифрування відрізняється від ключа дешифрування. Крім того, ключ дешифрування не може бути розрахований (протягом розумного часу) на основі ключа шифрування. Ключ шифрування називається «відкритим ключем», і тому будь-хто може надіслати зашифроване повідомлення одержувачу, але тільки одержувач може його розшифрувати. Ключ

дешифрування називається «секретним ключем». Детермінована схема шифрування — це криптосистема, яка завжди створює один і той самий зашифрований текст для заданого базового тексту і попередньо визначеного ключа, навіть на додаток до незалежного виконання алгоритму шифрування. Таким чином, детерміноване шифрування може надати інформацію зломиснику, який може розпізнати її з раніше відомого зашифрованого тексту. Імовірнісне шифрування використовує випадковість в алгоритмах шифрування, тому виходить інше зашифроване повідомлення, коли одне й те саме основне повідомлення шифрується кілька разів [4]. Для того, щоб бути семантично захищеним, тобто, щоб приховати часткову інформацію про основне повідомлення, алгоритм шифрування повинен бути імовірнісним. Імовірнісне шифрування відіграє дуже важливу роль у шифруванні з відкритим ключем. Припустимо, зломисник спостерігає за зашифрованим текстом і припускає, що основним повідомленням є ТАК або НІ. Коли використовується детермінований алгоритм шифрування, зломисник може просто спробувати зашифрувати кожне зі своїх припущень за допомогою відкритого ключа і порівняти результат із раніше спостережуваним зашифрованим текстом. Для захисту від такого типу атаки схеми шифрування з відкритим ключем повинні використовувати випадкові елементи, щоб гарантувати, що одне основне повідомлення буде зашифровано у велику кількість можливих зашифрованих повідомлень. Детермінована схема шифрування може бути перетворена в імовірнісну шляхом додавання випадкового рядка до основного повідомлення перед шифруванням за допомогою детермінованого алгоритму. У цьому випадку дешифрування вимагає застосування детермінованого алгоритму та ігнорування випадкового рядка, який був доданий. Раніше схеми, які застосовували цей підхід, були розкриті на основі обмежень у детермінованих схемах шифрування. ОАЕР інтегрує випадкову вставку [12].

Криптографія з відкритим ключем була вперше представлена в 1970-х роках завдяки роботам Діффі та Хеллмана і є великим проривом у криптографічному світі. Найвідомішою схемою з відкритим ключем, ймовірно, є RSA, представлена в 1978 році Рівестом, Шаміром і Адлеманом і досі широко використовується в даний

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		40

час. Ключовою особливістю є ідея асиметричного ключа, на відміну від симетричних схем, описаних вище; тобто кожен ключ складається з пари ключів[4]. Один із ключів є відкритим, використовується для шифрування та розповсюджується по каналу, тоді як інший є приватним і володіє лише авторизованим користувачем (користувачами), щоб дозволити розшифровку. Через цю особливу природу для реалізації протоколу необхідна функція, яку легко обчислити, але важко інвертувати. У криптографії вони називаються односторонніми функціями [12].



Рисунок 2.2 – Автори алгоритму RSA

Більшість односторонніх функцій, які використовуються в криптографії, засновані на деяких складних проблемах, що впливають з теорії чисел, наприклад, на прикладі розкладання на множники для RSA або дискретного логарифму для схеми Діффі-Хеллмана. Звичайно, оскільки ключ шифрування є відкритим, зловмисник, який намагається розшифрувати зашифрований текст, може спробувати зашифрувати кожне можливе повідомлення і в кінцевому підсумку отримати потрібний відкритий текст. Очевидно, це відбувається лише в теорії; однак говорити про повну таємність більше немає сенсу. Натомість схеми з відкритими ключами розроблені таким чином, щоб забезпечити безпеку обчислень.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		41

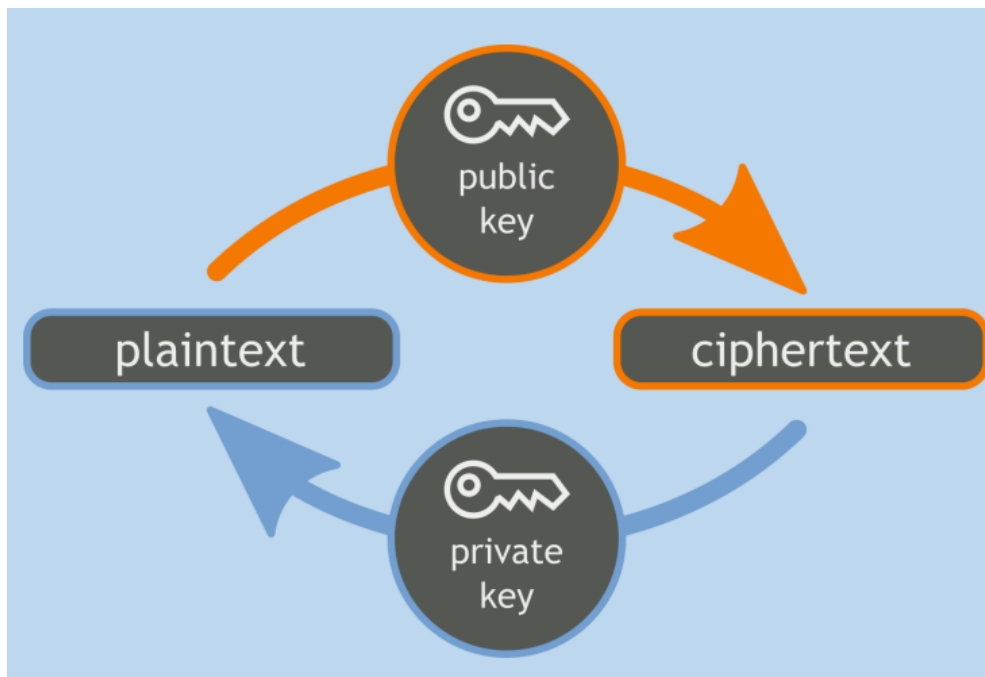


Рисунок 2.3 – Принцип роботи асиметричного шифрування

RSA є найвідомішим алгоритмом відкритого ключа. Його можна використовувати як для шифрування, так і для підписання. Алгоритм вважається надійним, коли ключі досить довгі (512 біт вважаються недостатніми, 768 біт - середньої надійності, 1024 біти - хорошими). Безпека RSA заснована на великій проблемі цілочисельної факторізації. Значний прогрес у тому, як можна розглядати великі числа, може зробити метод RSA вразливим. RSA на даний момент є найважливішим алгоритмом відкритого ключа [4].

RSA використовує операцію піднесення за модулем для шифрування та розшифровки повідомлення, яке виходить шляхом перетворення тексту в цифрову форму.

Генерація ключів RSA:

1. Вибираються два великих випадкових простих числа p та q .
2. Обчислюється модуль системи – добуток: $n = p \cdot q$.
3. Обчислюється функція Ейлера: $\varphi(n) = \varphi(p \cdot q) = (p - 1)(q - 1)$.
4. Випадковим чином вибирається число e (ключ шифрування), таке що $1 < e < \varphi(n)$ та взаємно просте з $\varphi(n)$.
5. За допомогою розширеного алгоритму Евкліда знаходиться число d (ключ дешифрування), таке що $ed = 1 \pmod{\varphi(n)}$.

6. (e, n) публікується у якості відкритого ключа.

7. (d, n) виконує роль закритого ключа і тримається таємниці.

Шифрування: повідомлення m розбивається на цифрові блоки, менші n ; кожен блок повідомлення m_i зашифровують за формулою, як показано на рис 2.4.

$$c_i = m_i^e \bmod n$$

Рисунок 2.4 – Формула розшифровки

Дешифрування: для кожного зашифрованого блоку c_i обчислюють за формулою, як показано на рис 2.5.

$$m_i = c_i^d \bmod n$$

Рисунок 2.5 – Формула дешифрування

У більш огляді захисту, запропонованої ключами RSA різної довжини, захист аналізується на основі методів розкладання на множники (факторингу), що існували в 1995 та перспективах їх розвитку, а також розглядає можливість залучення великих обчислювальних ресурсів по інформаційних мережах. Проведена в 1997 році оцінка показала, що 512-бітний ключ RSA може бути розкритий (факторингом) за \$ 1,000,000 та вісім місяців. У 1999 році 512-бітовий ключ був розкритий за сім місяців і це означає, що 512-бітові ключі вже не забезпечують достатню безпеку за винятком дуже короткострокових завдань безпеки.

В даний час Лабораторія RSA рекомендує для звичайних завдань ключі розміром 1024 біти, а для особливо важливих завдань – 2048 бітів (наприклад, для головного Майстра Сертифікатів).

Деякі нещодавно введені стандарти встановлюють для загальних завдань мінімальний розмір ключа 1024 біт. Менш цінна інформація може бути надійно зашифрована ключем 768-бітної довжини, оскільки такий ключ все ще недосяжний для всіх відомих алгоритмів злому. Для оцінки рівнів безпеки різних розмірів ключів можна використовувати модель, що пропонує Lenstra і Verheul.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

Зазвичай, ключ індивідуального користувача має певний термін життя, який закінчується через деякий час, наприклад, через рік. Це дає можливість регулярно замінювати ключі та забезпечувати необхідний рівень безпеки. Після закінчення терміну життя ключа користувач повинен створити новий ключ, попередньо переконавшись, що параметри криптосистеми залишилися колишніми, зокрема, що система використовує ключі тієї ж довжини. Звичайно, заміна ключа не захищає від нападу на повідомлення, зашифровані колишнім ключем, але для цього розмір ключа повинен підбиратися згідно з очікуваним часом актуальності даних. Можливість заміни ключів дозволяє підтримувати криптографічну систему відповідно до поточних рекомендацій щодо розмірів ключів, які регулярно публікує Лабораторія RSA.

2.4 Протокол передачі файлів SFTP

Secure Shell Protocol (SSH) — це протокол, який використовується для встановлення безпечного з'єднання між віддаленим сервером і комп'ютером. Secure Shell використовує шифрування з відкритим ключем для забезпечення надійної аутентифікації користувачів і безпечного зашифрованого зв'язку через Інтернет. Остання версія SSH2. Він був розроблений та прийнятий у 2006 році Інженерною групою з розробки Інтернету (IETF) для заміни старих незахищених протоколів оболонки. SSH File Transfer Protocol (SFTP) розроблено як розширення протоколу SSH2 для забезпечення безпечної передачі файлів. І командні з'єднання, і з'єднання даних шифруються між клієнтом і FTP-сервером, що дозволяє безпечно передавати паролі та іншу конфіденційну інформацію через мережу [18].

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

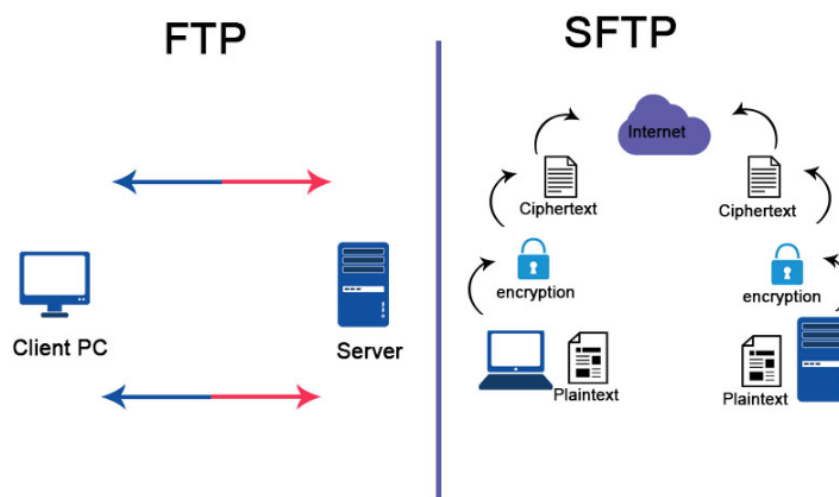


Рисунок 2.6 – Порівняння схеми роботи протоколу FTP та SFTP

Підключення до серверів SFTP встановлюються за допомогою аутентифікації пароля або за допомогою аутентифікації відкритих/приватних ключів. Завдяки автентифікації паролем, яка безпечно передається, користувачеві потрібні лише ім'я користувача та пароль для входу на сервер SFTP. Іншим методом є використання пар ключів. Генерується пара відкритих і закритих ключів, і відкритий ключ зберігається на сервері SFTP. Клієнт із закритим ключем аутентифікується на сервері під час входу, та якщо ключі збігаються, клієнт SFTP отримає доступ до системи. Для додаткової безпеки до приватного ключа також можна додати парольну фразу [11].

SFTP безпечно передає файли. Даний протокол використовує SSH та зашифровані FTP-команди, щоб уникнути перехоплення паролів та розкриття конфіденційної інформації у вигляді простого тексту. Оскільки клієнт повинен бути автентифікованим сервером, SFTP також захищає від атак "людина посередині".

Переваги використання SFTP:

Обмін даними – SFTP форматує команди та дані в спеціальні пакети та надсилає їх через одне з'єднання. Він не використовує окремі канали, як FTP. Це усуває незручності для адміністраторів серверів, яким в іншому випадку потрібно відкрити ряд портів для звичайних FTP - з'єднань.

Безпека – за допомогою FTP обидва канали зазвичай надсилаються незашифрованими, і ці дані можуть бути перехоплені та прочитані іншими

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		45

сторонами. Деякі FTP-сервери забезпечують безпеку, підключаючись через неявні SSL і явні з'єднання SSL. Однак вони повністю відрізняються від SFTP, який базується на протоколі SSH2 [18].

Брандмауер – SFTP використовує порт 22 для вхідних з'єднань сервера та вихідних підключень клієнта. Це робить брандмауер SFTP зручним, оскільки йому потрібен лише один порт, щоб відкрити його через брандмауер.

Метадані – за допомогою SFTP користувачі можуть отримати точний доступ до метаданих, таких як дата, час, розмір, дозволи та іншу інформацію, в той час як деякі FTP-сервери, особливо старіші, можуть мати проблеми з правильним отриманням цієї інформації, а інші не надають їх зовсім. Це може призвести до розбіжності резервного копіювання даних, оскільки метадані локальних файлів не збігаються з тими на сервері [11].

Висновок до розділу 2.

Отже у розділі розглянули просту, сучасна й об'єктно-орієнтована мову програмування C#. Метою якої була розробка мови програмування, яка може легко підтримувати сучасну функціональність для всіх видів розробки програмного забезпечення.

Середовище розробки .NET Framework, яке дає змогу різноманітним мовам програмування та бібліотекам безперешкодно працювати разом для створення програм на базі Windows.

Найвідоміший алгоритм відкритого ключа RSA. Який можна використовувати як для шифрування, так і для електронного підписання.

Застосовування SFTP у надійній політиці безпеки, так як безпека та цілісність даних є вирішальним пріоритетом для корпорацій та державних органів. Оскільки SFTP за своєю природою більш безпечний ніж звичайний FTP.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		46

РОЗДІЛ 3 ПРОЕКТНА РЕАЛІЗАЦІЯ ПІДСИСТЕМИ ЗАХИЩЕНОГО ДОКУМЕНТООБІГУ «DOC FLOW»

3.1 Опис додатку, тестування

Додаток Doc Flow створений для автоматизації бізнес-процесів та безпечної передачі документів між організаціями. Написаний на мові програмування C#, на платформі .net framework 6.0. Використовується ядро СУБД SQL Server.

На початку роботи програми відкривається вікно з аутентифікацією. Для входу необхідно ввести код ЄДРПОУ організації, ім'я користувача та пароль користувача.



Рисунок 3.1 – Вікно входу

Щоб створити нову організацію потрібно натиснути на кнопку «Log in» та ввести дані організації. Ім'я та код повинні бути унікальними. Код складається з 8 цифр, пароль має бути не менш ніж 8 символів.

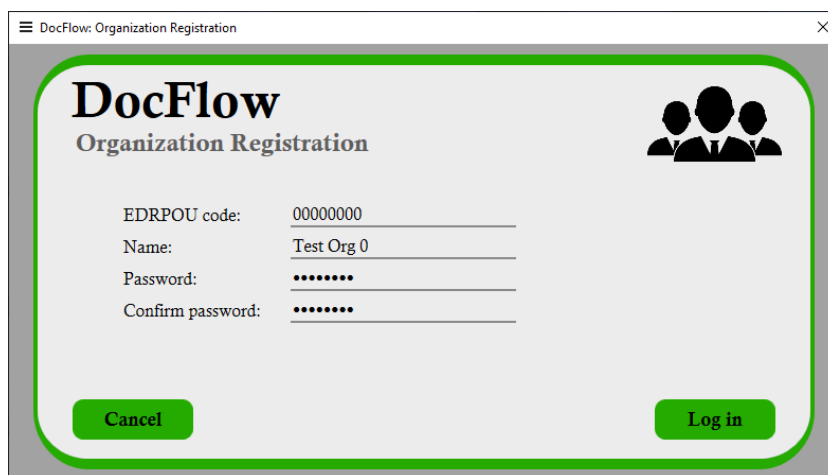


Рисунок 3.2 – Реєстрація організації

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		47

Після введення даних про організацію необхідно створити першого користувача. Пароль також повинен бути не менш ніж 8 символів.

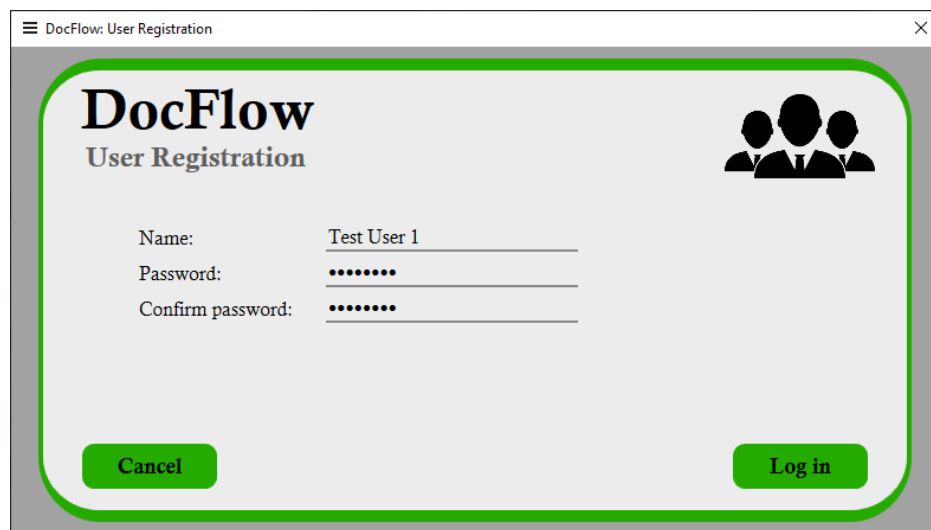


Рисунок 3.3 – Реєстрація користувача

При натисненні кнопки «Log in» проходить перевірка полів, при успішній реєстрації, обов’язково потрібно зберегти приватний ключ користувача, програма не дасть можливості відхилити збереження ключа. Тому що відновлення ключа не можливе.

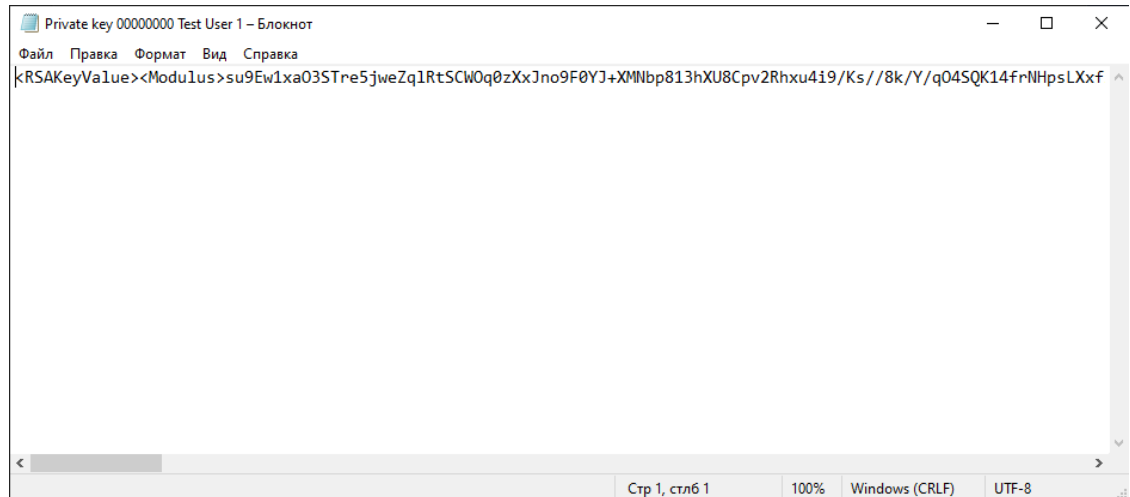


Рисунок 3.4 – Приватний ключ нового користувача

Після успішної реєстрації або входу відкривається основне вікно додатку. На даному етапі відкриваються можливості відправки та отримання документів.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		48

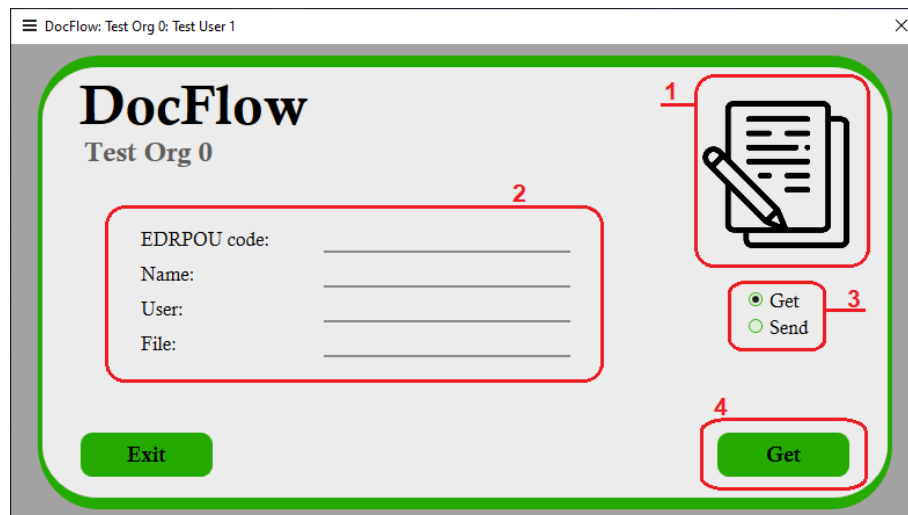


Рисунок 3.5 – Головне вікно додатку. 1 – Налаштування, 2 – реквізити для відправки або отримання документів, 3 – перемикач одержати/відправити, 4 – кнопка одержати/відправити документ

При натисненні на рисунок документу відкривається вікно налаштувань. На першій сторінці знаходиться електронна бібліотека, тобто історія відправлення або отримання документів. Також є можливість завантажити порожній шаблон документу. Це важливо, так як відправити документ можна лише по шаблону.

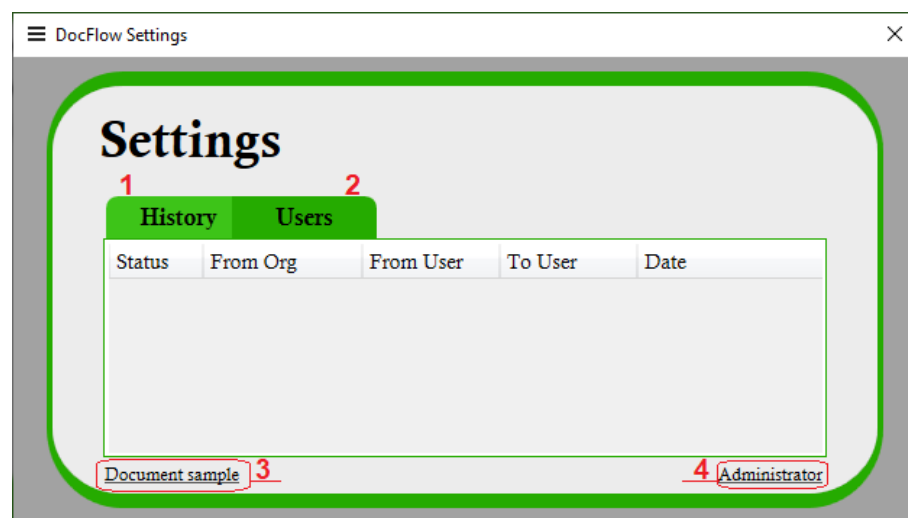


Рисунок 3.6 – Вікно налаштувань. 1– Сторінка електронної бібліотеки, 2 – сторінка зі списком користувачів, 3 – кнопка завантаження шаблону документа, 4 – кнопка для входу в режим адміністратора

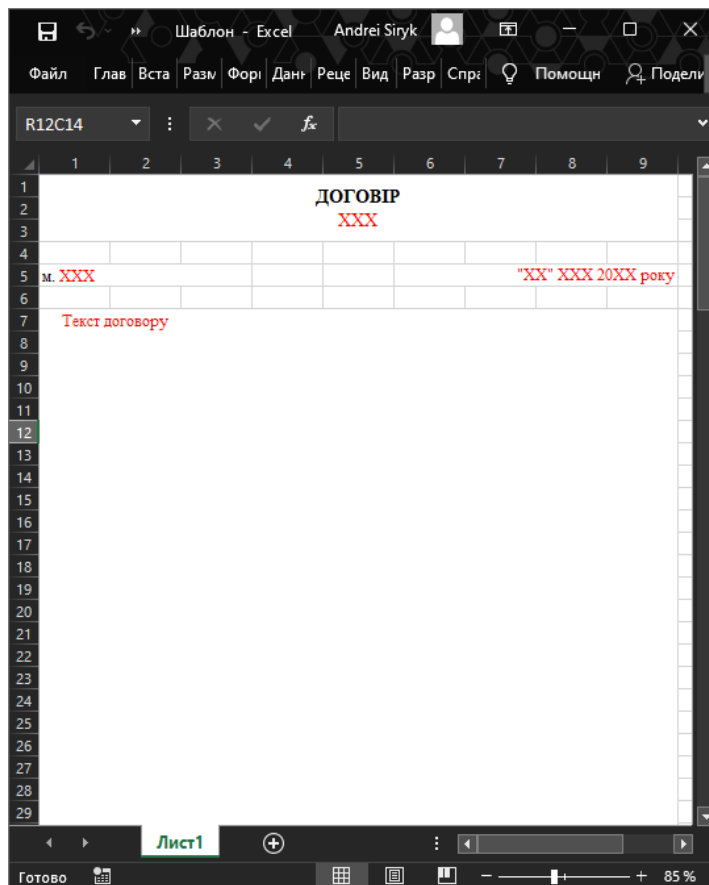


Рисунок 3.7 – Порожній шаблон документу

При натисненні на кнопку «Administrator» відкривається вікно входу у режим адміністратора, необхідно ввести пароль від організації. Адміністратор має можливість видаляти або додавати користувачів, змінювати паролі, переглядати електронну бібліотеку по всім користувачам. Звичайний користувач може тільки змінювати свій пароль та переглядати історію відправки, отримання своїх документів.

Рисунок 3.8 – Вхід у режим адміністратора

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		50

У режимі адміністратора з'являються кнопки додавання, видалення користувачів, а також перезаповнюються табличні частини.

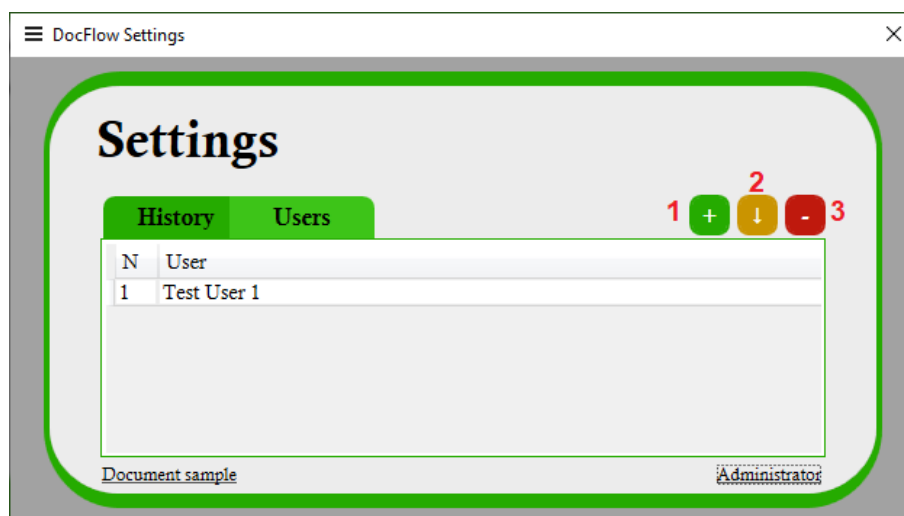


Рисунок 3.9 – Сторінка «Користувачі». 1 – Кнопка створення нового користувача, 2 – кнопка зміни пароля, 3 – кнопка видалення користувача

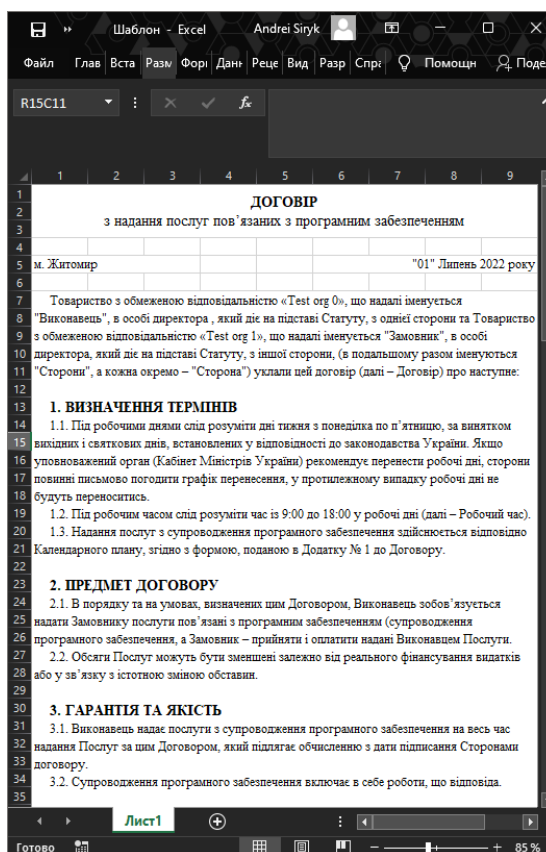


Рисунок 3.10 – Заповнений шаблон документу для відправлення

Після підготовки документу необхідно заповнити реквізити для відправки документу та обрати файл відправки. Після введення коду організації ім'я та список користувачі заповнюється автоматично.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк С. М.				51
Змн.	Арк.	№ докум.	Підпис	Дата		

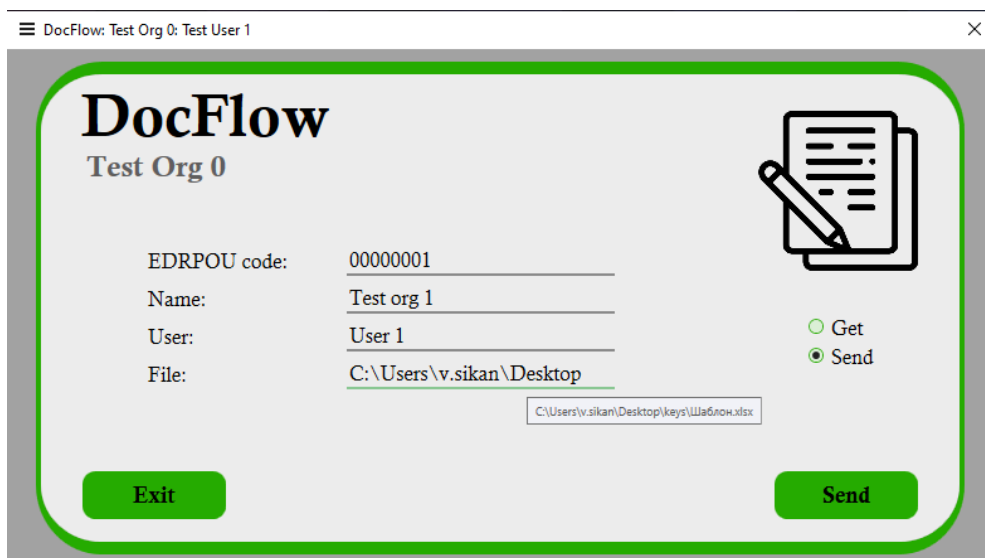


Рисунок 3.11 – Форма відправлення документа

Після натиснення на кнопку відправки виконується перевірка коректності заповнення реквізитів. При успішному відправленні з'явиться повідомлення.

При відправленні, документ розбивається на текстові частини, з'єднується, та конвертується у байтову систему. Після чого масив з байтами по блочно шифрується відритим ключем контрагента. Розмір одного блока становить 64 байт.

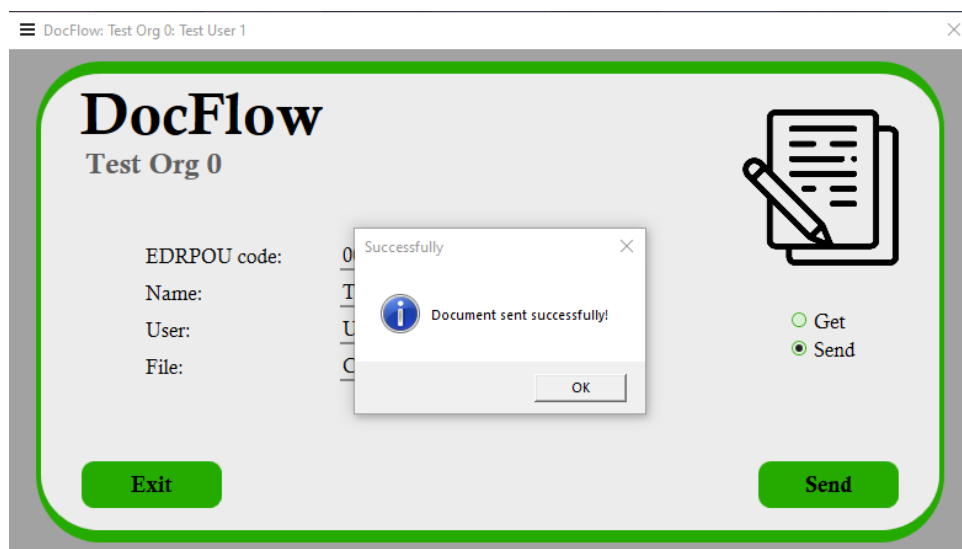


Рисунок 3.12 – Повідомлення про успішну відправку

У історії з'явився документ зі статусом «Sent», після отримання документу статус зміниться.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

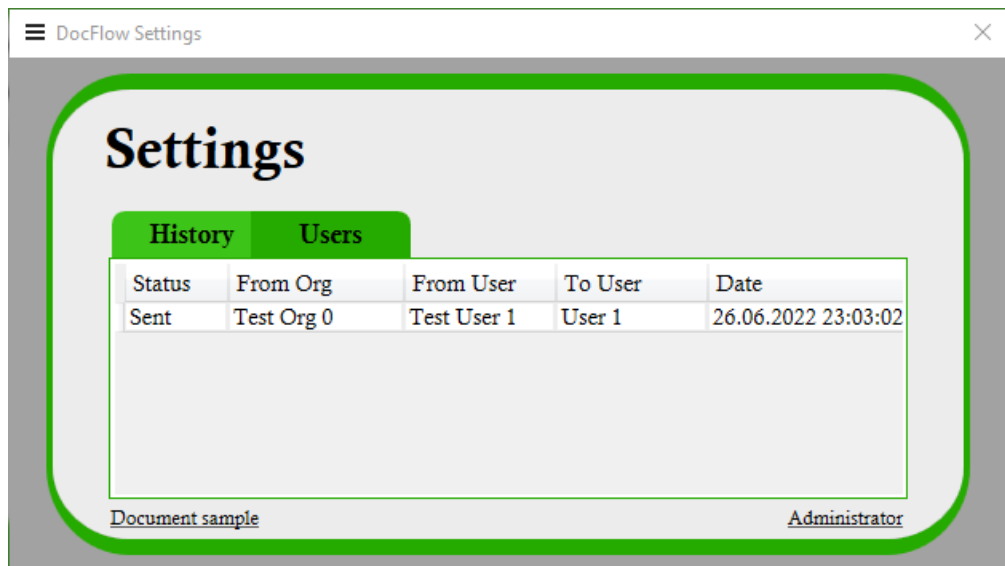


Рисунок 3.13 – Відображення відправленого документа у електронній бібліотеці

Підключимось до серверу SFTP за допомогою програми WinSCP та перевіримо відправку файлу.

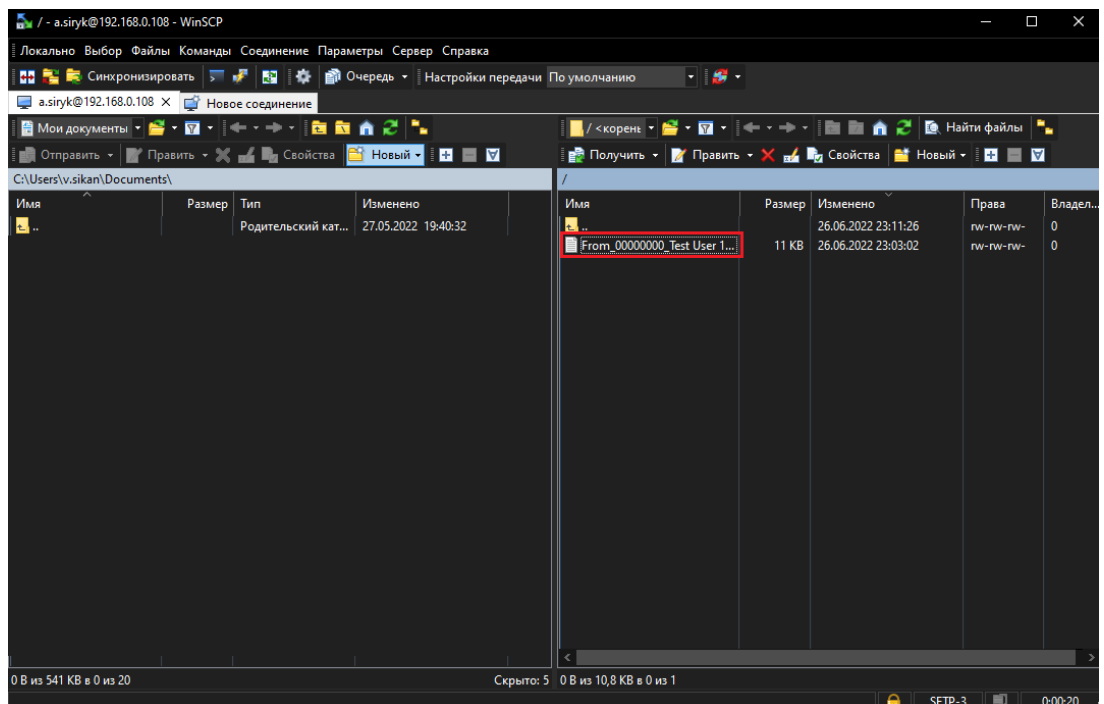


Рисунок 3.14 – Створення зашифрованого документа на SFTP сервері

Відкривши документ, можна побачити, що документ насправді зашифровано.



Рисунок 3.15 – Перегляд зашифрованого документу

Перезайдемо у програму під організацією контрагента та отримаємо документ. При отриманні є можливість вказати точну організацію та користувача, від якого потрібно отримати документи або не заповнювати та отримати всі документи. При натисненні на кнопку «Get» необхідно обрати приватний ключ, при виборі хибного ключа буде помилка з описом проблеми.

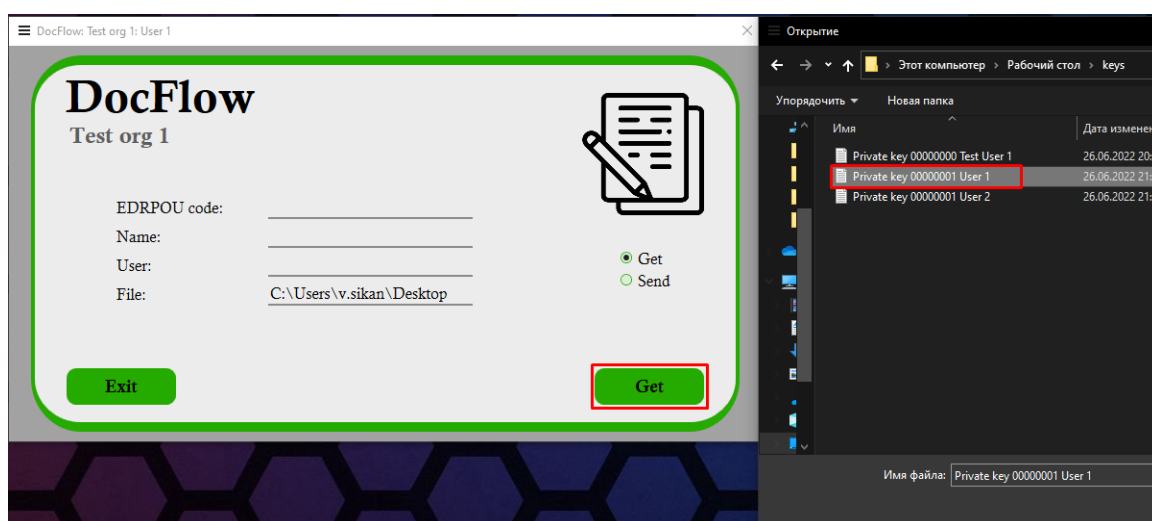


Рисунок 3.16 – Отримання документу, вибір приватного ключа

При успішному отриманні документу, він буде збережений в обрану теку в форматі .pdf.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк С. М.				54
Змн.	Арк.	№ докум.	Підпис	Дата		

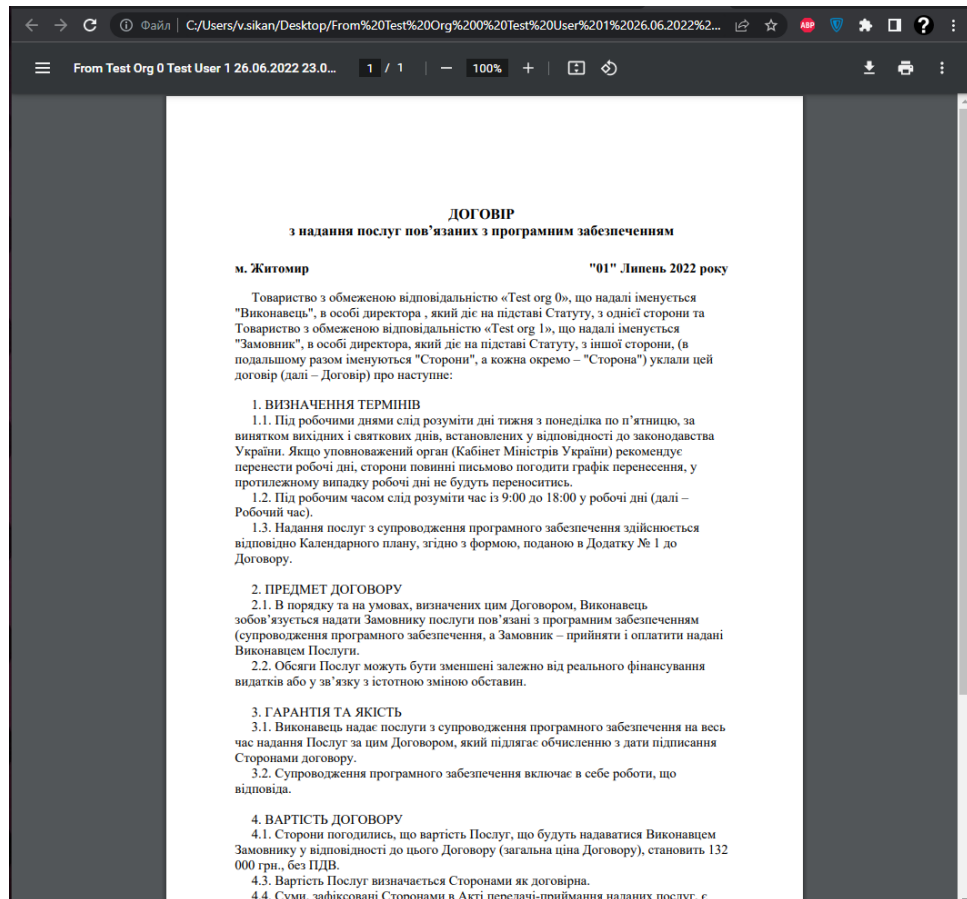


Рисунок 3.17 – Отриманий документ у форматі .pdf

Висновок до розділу 3.

У результаті завершення проєкту, досліджено роботу захищеного документообігу. Розглянуто етапи роботи з документами. Використано метод шифрування RSA на практиці.

Проект повністю відповідає вимогам технічного завдання.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				55
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

У дипломній роботі було досліджено особливості розробки та використання захищеного електронного документообігу. Пояснювальна записка складається з анотацій, вступу, трьох розділів, висновків та списку використаної літератури.

У першому розділі було досліджено використання електронного документообігу в бізнес-процесах, проаналізовано користь та безпечність даного методу в порівнянні з традиційним. У порівнянні, обрано тип шифрування для проекту. Проаналізовано використання документообігу в Україні та порівняно найпопулярніші сервіси. А також створено технічне завдання для проекту.

У другому розділі було проаналізовано мову програмування C#, а також платформу .net framework 6.0. Обрано найвідоміший алгоритм відкритого ключа RSA та досліджено принцип роботи методу. Розглянуто протокол безпечної передачі файлів SFTP.

У третьому розділі було розроблено підсистему захищеного документообігу у між підприємствами в інформаційній системі, протестовано проект та описано основні принципи роботи.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		56

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Верховна Рада України. Про електронні документи та електронний документообіг: закон України від 22 травня 2003 року № 851-IV. 2003.
2. Електронний документообіг та діловодство. Рішення Microsoft в сфері документообігу для українських органів державної влади та місцевого самоврядування. URL: <https://www.microsoft.com/>.
3. Abelson J., Sussman A. Structure and interpretation of computer programs. 2009.
4. Burnett S. RSA security's official guide to cryptography. 2001.
5. Grigonis R. Fifth-generation computers. Oxford University Press, 1987.
6. Hajjar D., AbouRizk S. Integrating document management with project and company data. Journal of Computing in Civil Engineering, 2000.
7. Hejlsberg A., Wiltamuth S. The c# programming language.
8. Jielin D., Dong A. Network dictionary. Saratoga, calif. Javvin Technologies, 2007.
9. Joia L. Large-scale Reengineering in project documentation and workflow at engineering consultancy companies. International Journal of Information Management, 1998.
10. Lati M., Hameri A., Eleoranta E. Improved project management through improved document management. 2001.
11. Mark K. Simple file transfer protocol. 1984.
12. Menezes A., Orschot P. Handbook of applied cryptography. 1996.
13. Richter J. Clr via c#. 2006.
14. Robert J. Introduction to C# using .NET. New Jersey : Upper Saddle River, 2002.
15. Schneier B. Applied cryptography. Protocols, algorithms and source code. 2017.
16. Stelman A. Head first c#. 2007.
17. Stewart B. History of the C programming language. 2000.
18. Tanenbaum E., Weatherall D. Computer networks. 2008.

		Сірик А. Ю.			<i>KIK.KP.B – 125 – 22 – ПЗ</i>	Арк.
		Байлюк Є. М.				57
Змн.	Арк.	№ докум.	Підпис	Дата		

19. The EM side-channel(s). Cryptographic hardware and embedded systems / D. Agrawal et al.

20. Tyagi A., Chua D. Process-parameter-interface model for lean design management. Singapore : Proc. 9th Annual Conference International Group for Lean Construction, 2001.

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		58

ДОДАТКИ

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				59
Змн.	Арк.	№ докум.	Підпис	Дата		

Літинг коду проекту

Файл FlowWindow.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using System.Drawing;
using System.Security.Cryptography;
using System.Data.SqlClient;
using System.Data;
using System.IO;
using Microsoft.Win32;
using System.Windows.Media.Effects;
using Excel = Microsoft.Office.Interop.Excel;
using SautinSoft;
using Renci.SshNet;
using Renci.SshNet.Sftp;

namespace DocFlow
{
    public class User
    {
        public string UserName { get; set; }
        public int num { get; set; }
    }

    public class HistoryClass
    {
        public string Status { get; set; }
        public string FromOrg { get; set; }
        public string FromUser { get; set; }
        public string ToUser { get; set; }
        public string Date { get; set; }
    }

    public partial class FlowWindow : Window
    {
        string PrivatKey = String.Empty;
        DataBase dataBase = new DataBase();
        public FlowWindow()
        {
            InitializeComponent();

            ToolTip toolTip = new ToolTip();
            StackPanel toolTipPanel = new StackPanel();
            toolTipPanel.Children.Add(new TextBlock { Text = "Select a folder to save documents" });

            toolTip.Content = toolTipPanel;
            File_.ToolTip = toolTip;

        }

        private void Window_MouseDown(object sender, MouseButtonEventArgs e)
        {
            Keyboard.ClearFocus();
            ListBoxUsers.Visibility = Visibility.Collapsed;
        }

        private void Exit_Click(object sender, RoutedEventArgs e)
        {
            WindowSignIn taskWindow = new WindowSignIn();
            this.Close();
            taskWindow.Show();
        }

        private void RB_Send_Checked(object sender, RoutedEventArgs e)
        {
            Get.Visibility = Visibility.Collapsed;
            Send.Visibility = Visibility.Visible;
            if (EDRPOU != null)
            {
                EDRPOU.Text = string.Empty;
                Name_.Text = string.Empty;
                File_.Text = string.Empty;
                ListBoxUsers.Items.Clear();
                User.Text = string.Empty;
                ToolTip toolTip = new ToolTip();
                StackPanel toolTipPanel = new StackPanel();
                toolTipPanel.Children.Add(new TextBlock { Text = "Select a document to send" });
                toolTip.Content = toolTipPanel;
                File_.ToolTip = toolTip;
            }
        }

        private void Get_Click(object sender, RoutedEventArgs e)
        {
            if (File_.Text == "")
            {
                MessageBox.Show("Save path not selected!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
            }
        }
    }
}
```

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк С. М.				60
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        return;
    }

    string key = String.Empty;
    if (PrivatKey == String.Empty)
    {
        MessageBox.Show("Choose a private key", "Info", MessageBoxButtons.OK, MessageBoxIcon.Information);
        OpenFileDialog openFileDialog = new OpenFileDialog();
        openFileDialog.Filter = "txt files (*.txt)|*.txt|All files (*.*)|*.*";
        if (openFileDialog.ShowDialog() == true)
        {
            key = File.ReadAllText(openFileDialog.FileName);
            bool TrueKey = key.Contains("<RSAKeyValue><Modulus>") & key.Contains("</Modulus><Exponent>") & key.Contains("<Exponent><P>") & key.Contains("</D></RSAKeyValue>") &
key.Contains("</DQ><InverseQ>");
            if (TrueKey == false)
            {
                MessageBox.Show("This file does not match the key pattern!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }
            PrivatKey = openFileDialog.FileName;
        }
        else
        {
            MessageBox.Show("Documents received: 0 \nPrivate key not selected!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
            return;
        }
    }

    }

    SqlDataAdapter adapter = new SqlDataAdapter();
    DataTable Table = new DataTable();
    string Request;
    if (EDRPOU.Text != String.Empty & User.Text == String.Empty)
    {
        Request = $"select * from History where ToOrg = '{InfoOrg.Content.ToString()}' and ToUser = '{InfoUser.Content.ToString()}' and FromOrg = '{Name_.Text}' and Status = 'Sent'";
    }
    else if (User.Text != String.Empty)
    {
        Request = $"select * from History where ToOrg = '{InfoOrg.Content.ToString()}' and ToUser = '{InfoUser.Content.ToString()}' and FromOrg = '{Name_.Text}' and FromUser = '{User.Text}' and
Status = 'Sent'";
    }
    else
    {
        Request = $"select * from History where ToOrg = '{InfoOrg.Content.ToString()}' and ToUser = '{InfoUser.Content.ToString()}' and Status = 'Sent'";
    }

    SqlCommand command = new SqlCommand(Request, dataBase.GetConnection());
    adapter.SelectCommand = command;

    try
    {
        Mouse.OverrideCursor = Cursors.AppStarting;
        adapter.Fill(Table);
        Mouse.OverrideCursor = null;
    }
    catch
    {
        Mouse.OverrideCursor = null;
        MessageBox.Show("No database connection!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        return;
    }
    if (Table.Rows.Count > 0)
    {
        Mouse.OverrideCursor = Cursors.AppStarting;
        foreach (DataRow row in Table.Rows)
        {
            try
            {
                GetFileFromSftp(@"\" + row[6].ToString());
                string FileWay = @"D:\Діплом\DocFlow\Temp\" + row[6].ToString();
                CryptoClass CryptoClass = new CryptoClass();
                List<byte> massArray = new List<byte>();
                massArray = CryptoClass.ReturnDecrypt(File.ReadAllText(PrivatKey), FileWay);
                string CurrentDate = row[7].ToString();
                CurrentDate = CurrentDate.Replace(':', '.');

                EditExcelFile(File_.Text, Encoding.UTF8.GetString(massArray.ToArray()), "From " + row[0].ToString()! + " " + row[1].ToString()! + " " + CurrentDate);

                string CommandChangePass = $"UPDATE History SET Status = 'Received' WHERE Path = '{row[6].ToString()}'";
                SqlCommand commandUpdate = new SqlCommand(CommandChangePass, dataBase.GetConnection());
                dataBase.openConnection();
                if (commandUpdate.ExecuteNonQuery() == 1)
                {
                    dataBase.CloseConnection();
                }
            }
            catch
            {
                Mouse.OverrideCursor = null;
                PrivatKey = String.Empty;
                MessageBox.Show("Wrong key!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }
        }
        Mouse.OverrideCursor = null;
        MessageBox.Show("Documents received!", "Successfully", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
    }
    else
    {
        MessageBox.Show("There are no incoming documents!", "Info", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}

```

		Сірук А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				61
Змн.	Арк.	№ докум.	Підпис	Дата		

```
private void Send_Click(object sender, RoutedEventArgs e)
{
    if (EDRPOU.Text == "" | User.Text == "")
    {
        MessageBox.Show("Organization or user not filled!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
        return;
    }
    if (File_.Text == "")
    {
        MessageBox.Show("File not selected!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
        return;
    }
    SqlDataAdapter adapter = new SqlDataAdapter();
    DataTable Table = new DataTable();
    string Request = $"select OpenKey from Users where EDRPOU = '{EDRPOU.Text}' and Name_ = '{User.Text}'";

    SqlCommand command = new SqlCommand(Request, dataBase.GetConnection());
    adapter.SelectCommand = command;

    try
    {
        Mouse.OverrideCursor = Cursors.AppStarting;
        adapter.Fill(Table);
        Mouse.OverrideCursor = null;
    }
    catch
    {
        Mouse.OverrideCursor = null;
        MessageBox.Show("No database connection!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);

        return;
    }
    if (Table.Rows.Count > 0)
    {
        try
        {
            string OpenKey = Table.Rows[0].ItemArray[0].ToString();
            CryptoClass CryptoClass = new CryptoClass();
            List<byte> massArray = new List<byte>();
            massArray = CryptoClass.ReturnEncrypt(OpenKey, File_.Text);
            string CurrentDate_ = DateTime.Now.ToString();
            string CurrentDate = CurrentDate_.Replace(':', '');
            string pathFile = "From_" + InfoEDRPOU.Content.ToString() + "_" + InfoUser.Content.ToString() + "_" + EDRPOU.Text + "_" + User.Text + "_" + CurrentDate + ".txt";
            string filename = @"D:\Ділознав\DocFlow\Temp\" + pathFile;

            StreamWriter file = new StreamWriter(filename);
            file.Write(Convert.ToBase64String(massArray.ToArray()));
            file.Close();
            SetFileToSftp(filename);
            File.Delete(filename);

            string CommandWriteUser = $"insert into History(FromOrg, FromUser, ToOrg, ToUser, Status, Path, Date) values('{InfoOrg.Content.ToString()}','{InfoUser.Content.ToString()}','{Name_}','{User.Text}','Sent','{pathFile}','{CurrentDate}')";

            SqlCommand commandUser = new SqlCommand(CommandWriteUser, dataBase.GetConnection());

            dataBase.openConnection();
            if (commandUser.ExecuteNonQuery() == 1)
            {
                MessageBox.Show("Document sent successfully!", "Successfully", MessageBoxButton.OK, MessageBoxImage.Asterisk);

            }
            else
            {
                MessageBox.Show("Document not sent!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
            }
            dataBase.CloseConnection();

        }
        catch
        {
            MessageBox.Show("Document not sent!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
            return;
        }
    }
    else
    {
        MessageBox.Show("Organization or user does not exist!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
        return;
    }
}

private void SetFileToSftp(string PathFrom)
{
    var host = "192.168.0.108";
    var port = 22;
    var username = "a.siryk";
    var password = "Qwer1234";

    var client = new SftpClient(host, port, username, password);

    client.Connect();
    if (client.IsConnected)
    {
        using (var fileStream = new FileStream(PathFrom, FileMode.Open))
        {
            client.BufferSize = 4 * 1024; // bypass Payload error large files
            client.UploadFile(fileStream, System.IO.Path.GetFileName(PathFrom));
        }
    }
}
```

		Сірук А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				62
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }
}
else
{
    MessageBox.Show("No connection to server!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
}
}

private void GetFileFromSftp(string PathFrom)
{
    var host = "192.168.0.108";
    var port = 22;
    var username = "a.siryk";
    var password = "Qwer1234";
    var client = new SftpClient(host, port, username, password);
    client.Connect();
    if (client.IsConnected)
    {
        using (Stream file1 = File.Create(@"D:\Диплом\DocFlow\Temp" + PathFrom))
        {
            client.BufferSize = 4 * 1024; // bypass Payload error large files
            client.DownloadFile(PathFrom, file1);
        }
    }
    else
    {
        MessageBox.Show("No connection to server!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
    }
}

private void RB_Get_Checked(object sender, RoutedEventArgs e)
{
    Get.Visibility = Visibility.Visible;
    Send.Visibility = Visibility.Collapsed;
    if (EDRPOU != null)
    {
        EDRPOU.Text = string.Empty;
        Name_.Text = string.Empty;
        File_.Text = string.Empty;
        ListBoxUsers.Items.Clear();
        User.Text = string.Empty;

        ToolTip toolTip = new ToolTip();
        StackPanel toolTipPanel = new StackPanel();
        toolTipPanel.Children.Add(new TextBlock { Text = "Select a folder to save documents" });

        toolTip.Content = toolTipPanel;
        File_.ToolTip = toolTip;
    }
}

private void EDRPOU_LostKeyboardFocus(object sender, KeyboardFocusChangedEventArgs e)
{
    if (EDRPOU.Text == "")
    {
        Name_.Text = "";
        User.Text = "";
        ListBoxUsers.Items.Clear();
        return;
    }

    SqlDataAdapter adapter = new SqlDataAdapter();
    DataTable Table = new DataTable();
    string Request = $"select Users.Name_, Users.EDRPOU, Organization.Name_ as OrgName from Users Inner join Organization on Users.EDRPOU = Organization.EDRPOU where Users.EDRPOU = '{EDRPOU.Text}'";

    SqlCommand command = new SqlCommand(Request, dataBase.GetConnection());
    adapter.SelectCommand = command;
    ListBoxUsers.Items.Clear();
    Name_.Text = "";
    User.Text = "";
    try
    {
        Mouse.OverrideCursor = Cursors.AppStarting;
        adapter.Fill(Table);
        Mouse.OverrideCursor = null;
    }
    catch
    {
        Mouse.OverrideCursor = null;
        MessageBox.Show("No database connection!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
        Keyboard.ClearFocus();
        return;
    }
    if (Table.Rows.Count > 0)
    {
        if (Table.Rows.Count > 3)
        {
            ListBoxUsers.Height = 84;
        }
        else if (Table.Rows.Count == 3)
        {
            ListBoxUsers.Height = 64;
        }
        else if (Table.Rows.Count == 2)
        {
            ListBoxUsers.Height = 44;
        }
        else if (Table.Rows.Count <= 1)
    }
}

```

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				63
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        {
            ListBoxUsers.Height = 24;
        }
        Name_.Text = Table.Rows[0].ItemArray[2]!.ToString();

        for (int i = 0; i < Table.Rows.Count; i++)
        {
            ListBoxUsers.Items.Add(Table.Rows[i].ItemArray[0]!.ToString());
        }
    }
    else
    {
        Keyboard.ClearFocus();
        Name_.Text = "";
        MessageBox.Show("The Organization does not exist!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
    }
}

private void OnPreviewMouseDown(object sender, EventArgs e)
{
    ListBoxUsers.Visibility = Visibility.Visible;
}

private void User_LostFocus(object sender, RoutedEventArgs e)
{
    ListBoxUsers.Visibility = Visibility.Collapsed;
}

private void ListBoxUsers_SelectionChanged(object sender, SelectionChangedEventArgs e)
{
    if (ListBoxUsers.SelectedItem != null)
    {
        User.Text = ListBoxUsers.SelectedItem.ToString();
    }
}

private void File_PreviewMouseDown(object sender, MouseButtonEventArgs e)
{
    if (Send.Visibility == Visibility.Visible)
    {
        OpenFileDialog openFileDialog = new OpenFileDialog();
        openFileDialog.Filter = "Excel Files|*.xls;*.xlsx;*.xlsm";
        if (openFileDialog.ShowDialog() == true)
        {
            File_.Text = openFileDialog.FileName;
        }
    }
    else
    {
        System.Windows.Forms.FolderBrowserDialog dialog = new System.Windows.Forms.FolderBrowserDialog();
        if (dialog.ShowDialog() == System.Windows.Forms.DialogResult.OK)
        {
            File_.Text = dialog.SelectedPath;
        }
    }
    ToolTip toolTip = new ToolTip();
    StackPanel toolTipPanel = new StackPanel();
    toolTipPanel.Children.Add(new TextBlock { Text = File_.Text, FontSize = 9 });

    toolTip.Content = toolTipPanel;
    File_.ToolTip = toolTip;
}

private void Images_PreviewMouseDown(object sender, MouseButtonEventArgs e)
{
    Settings Settings = new Settings();
    this.IsHitTestVisible = false;
    BlurEffect myBlurEffect = new BlurEffect();
    myBlurEffect.Radius = 8;
    this.Effect = myBlurEffect;
    Settings.Show();
    Settings.InfoOrg.Content = InfoOrg.Content;
    Settings.InfoEDRPOU.Content = InfoEDRPOU.Content;

    Settings.InfoUser.Content = InfoUser.Content;
    FillUsers(Settings.TableUsers);
    FillHistory(false, Settings.TableHistory);
}

private void FillHistory(bool admin, DataGrid TableHistory)
{
    SqlDataAdapter adapter = new SqlDataAdapter();
    DataTable Table = new DataTable();
    string Request;
    if (admin)
    {
        Request = $"select * from History where ToOrg = '{InfoOrg.Content.ToString()}' or FromOrg = '{InfoOrg.Content.ToString()}'";
    }
    else
    {
        Request = $"select * from History where (ToOrg = '{InfoOrg.Content.ToString()}' and ToUser = '{InfoUser.Content.ToString()}' or (FromOrg = '{InfoOrg.Content.ToString()}' and FromUser = '{InfoUser.Content.ToString()}')";
    }
    SqlCommand command = new SqlCommand(Request, dataBase.GetConnection());
    adapter.SelectCommand = command;

    try

```

		Сірук А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		64

```

{
    Mouse.OverrideCursor = Cursors.AppStarting;
    adapter.Fill(Table);
    Mouse.OverrideCursor = null;
}
catch
{
    Mouse.OverrideCursor = null;
    MessageBox.Show("No database connection!", "Error", MessageBoxButton.OK, MessageBoxImage.Error);
    return;
}
if (Table.Rows.Count > 0)
{
    int n = 1;
    foreach (DataRow row in Table.Rows)
    {
        HistoryClass HistoryClass = new HistoryClass();
        HistoryClass.Status = row[5].ToString();
        HistoryClass.Date = row[7].ToString();
        HistoryClass.ToUser = row[3].ToString();
        HistoryClass.FromUser = row[1].ToString();
        HistoryClass.FromOrg = row[0].ToString();

        TableHistory.Items.Add(HistoryClass);

        n++;
    }
}
}

private void FillUsers(DataGrid TableUsers)
{
    User User = new User();
    User.UserName = InfoUser.Content.ToString();
    User.num = 1;
    TableUsers.Items.Add(User);
}

private void FlowWindow__Closing(object sender, System.ComponentModel.CancelEventArgs e)
{
    foreach (Window window in Application.Current.Windows)
    {
        if (window.GetType() == typeof(Settings) )
        {
            (window as Settings)!.Close();
        }
        if (window.GetType() == typeof(WindowSignInOrg))
        {
            (window as WindowSignInOrg)!.Close();
        }
        if (window.GetType() == typeof(ChangePass))
        {
            (window as ChangePass)!.Close();
        }
        if (window.GetType() == typeof(AddUser))
        {
            (window as AddUser)!.Close();
        }
    }
}

private void EditExcelFile(string FilePath, string text, string namePDF)
{
    string[] ArrayText = text.Split('\n');

    File.Copy(@"D:\Диплом\DocFlow\Шаблон\ДляПолучения.xlsx", @"D:\Диплом\DocFlow\Temp\Полученный Файл.xlsx", true);
    var excelapp = new Excel.Application();
    excelapp.Visible = false;

    var excelappworkbooks = excelapp.Workbooks;

    var excelappworkbook = excelapp.Workbooks.Open(@"D:\Диплом\DocFlow\Temp\Полученный Файл.xlsx",
        Type.Missing, Type.Missing, Type.Missing,
        "WWWWW", "WWWWW", Type.Missing, Type.Missing, Type.Missing,
        Type.Missing, Type.Missing, Type.Missing, Type.Missing,
        Type.Missing, Type.Missing);

    var excelsheets = excelappworkbook.Worksheets;

    var excelworksheet = (Excel.Worksheet)excelsheets.get_Item(1);

    var excelcells = excelworksheet.get_Range("A1", "A1");
    excelcells.Value2 = ArrayText[0];

    excelcells = excelworksheet.get_Range("A5", "A5");
    excelcells.Value2 = ArrayText[1];

    excelcells = excelworksheet.get_Range("F5", "F5");
    excelcells.Value2 = ArrayText[2];

    excelcells = excelworksheet.get_Range("A7", "A7");
    excelcells.Value2 = ArrayText[3];
    excelappworkbooks = excelapp.Workbooks;
    excelappworkbook = excelappworkbooks[1];
    excelappworkbook.Save();
    excelapp.Quit();

    SautinSoft.ExcelToPdf x = new SautinSoft.ExcelToPdf();

```

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

```
bool check = false;
do
{
    try
    {
        x.ConvertFile(@"D:\Диплом\DocFlow\Temp\Полученный Файл.xlsx", FilePath + @"\" + namePDF + ".pdf");
        File.Delete(@"D:\Диплом\DocFlow\Temp\Полученный Файл.xlsx");
        check = true;
    }
    catch
    {
        check = false;
    }
}
while (!check);
}
}
```

Файл CryptoClass.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Security.Cryptography;
using System.IO;
using ExcelDataReader;
using System.Data;

namespace DocFlow
{
    class CryptoClass
    {
        public string HashPassword(string Pass)
        {
            //хеширование пароля
            using (var md5Hash = MD5.Create())
            {
                // Представление массива байтов исходной строки
                var sourceBytes = Encoding.UTF8.GetBytes(Pass);

                // Сгенерировать хэш-значение (массив байтов) для входных данных
                var hashBytes = md5Hash.ComputeHash(sourceBytes);

                // Преобразование массива хэш-байтов в строку
                var hash = BitConverter.ToString(hashBytes).Replace("-", string.Empty);
                return hash;
            }
        }

        public string[] GenerateRSAKeys()
        {
            string[] Keys = new string[2];

            const int KEY_SIZE = 1024;
            using (var rsa = new RSACryptoServiceProvider(KEY_SIZE))
            {
                Keys[0] = rsa.ToXmlString(false); // Public Key
                Keys[1] = rsa.ToXmlString(true); // Private Key
            }

            return Keys;
        }

        public List<byte> ReturnEncrypt(string OpenKey, string File_)
        {
            RSACryptoServiceProvider rsa = new RSACryptoServiceProvider();
            rsa.FromXmlString(OpenKey);

            int Length_ = 64;
            string FullText = ReturnStringFromFile(File_);
            byte[] text = Encoding.UTF8.GetBytes(FullText);
            int part = text.Length / Length_;
            List<byte> massArray = new List<byte>();

            for (int i = 0; i <= part; i++)
            {
                byte[] temp = ReturnSubByte(text, i * Length_, Length_);

                massArray.AddRange(rsa.Encrypt(temp, true));
            }
            return massArray;
        }

        private byte[] ReturnSubByte(byte[] text, int i, int Length)
        {
            byte[] temp = new byte[Length];
            int Index_ = 0;
            for (int j = i; j < i + Length; j++)
            {
                if (text.Length <= j) continue;
                temp[Index_] = text[j];
                Index_++;
            }
        }
    }
}
```

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк С. М.				66
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        return temp;
    }

    public List<byte> ReturnDecrypt(string PrivateKey, string File_)
    {
        int Length_ = 128;
        RSACryptoServiceProvider rsa = new RSACryptoServiceProvider();
        rsa.FromXmlString(PrivateKey);

        string FullText = File.ReadAllText(File_);
        byte[] text = Convert.FromBase64String(FullText);
        int part = text.Length / Length_;
        int RemainderRart = text.Length % Length_;
        List<byte> massArray = new List<byte>();
        part = RemainderRart == 0 ? part - 1 : part;

        for (int i = 0; i <= part; i++)
        {
            if(RemainderRart != 0 & i == part)
            {
                Length_ = RemainderRart;
            }

            byte[] temp = ReturnSubByte(text, i * Length_, Length_);

            massArray.AddRange(rsa.Decrypt(temp, true));

        }
        return massArray;
    }

    private string ReturnStringFromFile(string FileWay)
    {
        string TextFile = "";
        FileStream stream = File.Open(FileWay, FileMode.Open, FileAccess.Read);
        IExcelDataReader excelReader = ExcelReaderFactory.CreateOpenXmlReader(stream);
        DataSet result = excelReader.AsDataSet();

        DataTable dt = result.Tables[0];
        string text = dt.Rows[0][0].ToString();
        TextFile = TextFile + text + "↓";
        text = dt.Rows[5][0].ToString();
        TextFile = TextFile + text + "↓";
        text = dt.Rows[5][6].ToString();
        TextFile = TextFile + text + "↓";
        text = dt.Rows[7][0].ToString();
        TextFile = TextFile + text;
        stream.Close();
        return TextFile;
    }
}

```

Повний проект дипломної роботи можна завантажити за посиланням:
https://gitlab.com/kb2_sayu/docflow.git

		Сірик А. Ю.			КІК.КР.Б – 125 – 22 – ПЗ	Арк.
		Байлюк Є. М.				
Змн.	Арк.	№ докум.	Підпис	Дата		67