НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки 6.170101 «Безпека інформаційних і комунікаційних систем»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ М.В.Грайворонський
(підпис)
«___»_____2019 р.

ЗАВДАННЯ
на дипломну роботу студенту

_____ Курті Даніїлу _____
(прізвище, ім'я, по батькові)

1. Тема роботи Захищена система документообігу на мобільній платформі_____ ,
науковий керівник роботи к. ф.-м.н. Грайворонський М.В. _____
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «____ » 2019 р. № _____

2. Термін подання студентом роботи 10 червня 2019 р.

3. Вихідні дані до роботи _____
_____
_____

4. Зміст роботи_____
_____
_____
_____
_____

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо) _____
_____
_____
_____

6. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів дипломної роботи	Примітка

Студент _____ _____
(підпис) (ініціали, прізвище)

Науковий керівник роботи _____ _____
(підпис) (ініціали, прізвище)

РЕФЕРАТ

Кваліфікаційна робота: 61 с., 21 рис., 4 додатки, 18 джерел.

Об’єкт дослідження: системи захищеного документообігу.

Предмет досліджень: захищеність системи документообігу на мобільній платформи.

Мета роботи: створення і тестування прототипу захищеної системи документообігу на мобільній платформі, позбавленої головних недоліків сучасних систем.

Методи дослідження: емпіричний, а саме порівняння та аналіз.

При написанні роботи був проведений теоретичний аналіз і узагальнення наукової літератури, практичний аналіз існуючих систем захищеного документообігу та подальшими висновками що до переваг та недоліків кожної з розглянутих реалізацій, проаналізовано наявність необхідних особливостей для зручного користувацького інтерфейсу.

Результатом роботи є створений прототип захищеної системи документообігу на мобільній платформі, яка позбавлена більшості недоліків аналогів.

Ключові слова: інформаційна безпека, системи документообігу, центри сертифікації, мобільна платформа, захист систем документообігу.

ABSTRACT

Explanatory note: 61 pages, 21 pictures, 0 tables, 4 additions, 18 sources.

Object of research: Secure document flow systems.

Subject of research: security of the document flow system on the mobile platform.

The purpose of the work: to model a secure document flow system on a mobile platform for testing the performance in the conditions of the needs of real users.

Research methods: empirical, namely, comparison and analysis.

During the research, such options were achieved: theoretical analysis and generalization of scientific literature, a practical analysis of existing systems of secure document flow and further conclusions about the advantages and disadvantages of each of the considered implementations, analyzed the availability of the necessary features for a convenient user interface.

The result of the work is created secure document flow system on the mobile platform, which is deprived of most of the shortcomings of analogues.

Key words: information security, secure document flow systems, certification centers, mobile platform, document flow systems security.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 7

Вступ 8

1 Аналіз захищених систем документообігу та загроз їх безпеці 11

1.1 Основні поняття інформаційної безпеки та захисту інформації 11

1.2 Системи документообігу 16

1.3 Загрози безпеці та методи захисту систем документообігу 19

1.4 Центри сертифікації 23

1.5 Електронно цифровий підпис 26

Висновки до першого розділу 30

2 Аналіз існуючих системи документообігу на мобільній платформі 31

2.1 PandaDoc 31

2.2 UKey 36

Висновки до другого розділу 40

3 Реалізація захищеної системи документообігу на мобільній платформі 41

3.1 Мобільний застосунок 42

3.2 Серверна частина 47

3.3 Подальший розвиток системи 48

3.4 Впровадження системи 49

Висновки до третього розділу 51

Висновки 52

Перелік джерел посилань 53

Додатки 55

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

- Месенджери – застосунки для обміну текстовими, файловими, відео повідомленнями.
- Фішинг – метод заволодіння інформацією приватного характеру обманним шляхом.
- Претекстинг – реалізація заздалегідь підготовленого сценарію, щоб спонукати жертву до розголошення інформації чи виконання дій.
- Токен – фізичний пристрій для забезпечення інформаційної безпеки та ідентифікації користувача.
- Брутфорс – метод рішення криптографічної задачі шляхом перебору всіх можливих варіантів ключа.
- Android – операційна система і платформа для мобільних телефонів та планшетів, створена компанією Google на базі ядра Linux.
- iOS – операційна система і платформа для мобільних телефонів та планшетів, створена компанією Apple.
- Play Market – онлайн магазин для розповсюдження мобільних застосунків на платформі Android.
- iCloud – хмарна файлова система для користувачів iOS систем.
- Heroku – хмарне середовище для розробки.
- Https – (HyperText Transfer Protocol Secure) розширення протоколу http з використанням сертифікатів для підвищення рівню захисту.
- АКЦС - Акредитований центр сертифікації ключів

ВСТУП

Поширення хмарних технологій в світі надає можливість отримати та обробити дані за допомогою пристрою з доступом до інтернету з будь-якої точки світу. Виходячи з цього, а також, зважаючи на розвиток та розповсюдженість мобільних пристроїв, можна зауважити що значна доля ринку надання послуг переходить в інформаційний простір акцентуючись на мобільних платформах.

Для того щоб відповідати потребам сучасних користувачів, сервісам які пов’язані з захистом інформації необхідно також змінюватись. На сам перед це стосується послуг перевірки цілісності відправленої інформації і верифікації відправника. Існуючі центри сертифікації мають у розпорядженні функціонал який доступний тільки для користувачів персональних комп’ютерів. А саме, розповсюдження ключів через фізичні носії, підключення до персональних комп’ютерів, надсилання через інтернет файлу який містить конфіденційну інформації. Для подальшої взаємодії, а також у зручному для користувача ключі, необхідні мобільні застосунки, які не втрачаючи позитивних якостей вже реалізованих систем, але будуть позбавлені недоліків і відповідають потребам учасників системі.

Існує безліч мобільних систем які позиціонують себе як застосунки для роботи з електронно цифровим підписом. Проте їх реалізація використовує фізичне написання пальцем підпису людини на екрані пристрою. Даний метод не гарантує правильність верифікації відправника документу. До того ж, похибка такого написання на екрані доволі велика і навіть реальний власник облікового запису може помилитись при написанні.

Приклади таких програм:

- PandaDoc[1]

- DocuSign[2]
- HelloSign[3]

На даний момент існує тільки один мобільний застосунок, від українських розробників, який використовує криптографічні методи захисту для підпису документів, проте він знаходиться в тестовому режимі. До його функціоналу входить верифікація відправника, передача інформації по захищеному каналу, імпорт файлу, який містить персональний ключ , для подальшого використання. Недоліком даного застосунку виступає не можливість використання без відповідних документів отриманих в банку з яким інтегрована система.

Звідси випливає висновок, що ніша застосунку для захищеного документообігу не зайнята навіть вже існуючими центрами сертифікації. Для вирішення цієї проблеми можливо розробити захищену систему документообігу на мобільній платформі в яку буде включено криптографічні методи захисту інформації, захищений канал передачі інформації, відсутність передачі приватного ключа користувача за межі його пристрою, перегляд історії транзакцій та документів користувача в системі. Посередником між транзакціями буде виступати окремий або існуючий сервер центру сертифікації. Враховуючи можливість інтеграції мобільної логіки до вже існуючих систем, даний застосунок націлений на розширення функціоналу центрів сертифікації, а також спрощенням процедури підпису.

Метою роботи є створення і тестування прототипу захищеної системи документообігу на мобільній платформі, позбавленої головних недоліків сучасних систем.

Для досягнення даної мети було поставлено такі завдання:

- Проаналізувати сучасні системи документообігу, насамперед ті, що орієнтовані на мобільні платформи.
- Виявити їх типові недоліки, знайти причину цих недоліків, запропонувати шлях(и) подолання цих недоліків, реалізувати прототип системи, що позбавлена цих недоліків.

- Провести тестування прототипу в умовах потреб реальних користувачів.

Під час розробки застосунку були використані такі **методи дослідження**, як емпіричний, а саме порівняння, для виявлення відмінностей від існуючих систем захищеного документообігу. А також, емпіричний аналіз для вимірювання показників окремо взятих складових частин застосунку. Таким чином отримуючи результати при роботі системи на реальних пристроях.

Наукова новизна роботи включає в себе вперше реалізовано систему з використанням криптографічних методів захисту інформації на мобільному пристрої за допомогою персонального ключа користувача який не покидає файлову систему. На додаток, реалізація включає в себе просту інтеграцію з існуючими центрами сертифікації.

Актуальність теми зумовлена широким використанням центрів сертифікації у сучасному світі, проте їх послуги не доступні користувачам мобільних платформ, що робить розробку захищеної системи документообігу на мобільній платформі першочерговим завданням

Практичне значення одержаних результатів - мобільний застосунок актуальної версії містить повністю реалізовані механізми захищеного каналу передачі інформації, криптографічні методи захисту інформації, користувацький інтерфейс, базу даних з тестовими користувачами системи. Виходячи з цього, можна зауважити що він готовий до використання на ринку надання послуг як в ролі окремого продукту так і інтегрованої частини в центри сертифікації. Реалізований застосунок не був впроваджений до організацій.

Об’єктом дослідження є системи захищеного документообігу.

Предмет досліджень - захищеність системи документообігу на мобільній платформи.

1 АНАЛІЗ ЗАХИЩЕНИХ СИСТЕМ ДОКУМЕНТООБІГУ ТА ЗАГРОЗ ЇХ БЕЗПЕЦІ

Розділ включає в себе загальні відомості про основні властивості захисту інформації, структуру та принципи роботи систем документообігу і центрів сертифікації, а також описує існуючі механізми захисту таких систем з використанням різних підходів для підвищення рівня безпеки і комфортного використання.

1.1 Основні поняття інформаційної безпеки та захисту інформації

Захист інформації – заходи, спрямовані на збереження інформації від небажаних наслідків дій, що навмисно або випадково призводять до модифікації, розкриття чи руйнування даних[4]. На додаток вони забезпечують конфіденційність, цілісність, доступність, спостережність інформації яка зберігається, передається при можливості впливу загроз природного або штучного характеру, успішна реалізація яких призводить до наслідків різної тяжкості як власникам так і користувачам.

Конфіденційність – це властивість інформації, завдяки якій лише вповноважені користувачі мають змогу її отримати[5]. Доступ до якої може бути обмежено законом, від несанкціонованого доступу зі сторони зловмисників. На додаток, властивість повинна виконуватись як для інформації що передається так і зберігається. Порушення конфіденційності може бути спричинена рядом факторів таких як: помилка при розробці системи яка буде маніпулювати конфіденційною інформацією, знехтування правилами користування і зберігання ресурсами та носіями що містять таку інформацію, використання соціальної інженерії на людині яка має доступ до грифів таємності різного

ступеню. Конфіденційність даних забезпечується нормативно-правовими заходами , програмно-апаратними та технічними засобами.

Цілісність – це властивість інформації, яка дає можливість лише вповноваженим користувачам її модифікувати[6]. Тобто надання гарантії незмінності інформації впродовж усього циклу користування та взаємодії з нею користувачами. Порушення цілісності інформації може бути спричинена перехоплення інформації зловмисником під час передачі, викрадення зловмисником даних, їх модифікацією і з подальшою підміною оригіналу. Для перевірки цілісності даних використовуються хеш-функції, основний принцип яких полягає у перетворенні даних в унікальне значення фіксованої довжини. При зміні оригіналу змінюється і хеш-функція. Для забезпечення цілісності даних використовуються захищенні канали передачі інформації, криптографічні методи захисту, програмно-апаратні засоби.

Доступність – це властивість яка забезпечує користувача або процес, який володіє відповідними вповноваженнями, може використовувати інформацію згідно з правилами політики безпеки системи. Мається на увазі, що маючи необхідний доступ користувач отримує доступ до бажаної інформації в тому місці і часі які йому необхідні. Властивість доступності може бути порушена завдяки технічним порушенням, атакам зловмисників на різні частини системи, як програмно-апаратну так і технічну. Для забезпечення доступності розробляється зручний та інтуїтивно зрозумілий для користувача інтерфейс, який впроваджується разом з інструкцією за реалізованим функціоналом, а також проводиться період ознайомлення та тестування майбутніми користувачами системи.

Спостережність – це властивість системи яка забезпечує фіксування діяльності користувачів та сервісів в системі, встановлюючи однозначно дані про джерело тих чи інших дій, а також виконання заздалегідь встановлених дій у разі встановлення порушень правил політики безпеки системи і забезпечення відповідальності за певні дії. Для забезпечення спостережності системи використовуються журнали-аудиту.

Інформаційна безпека - це стан захищеності інформаційного середовища, захист інформації являє собою діяльність щодо запобігання витоку інформації, що захищається, несанкціонованих, навмисних та ненавмисних впливів на інформацію, що захищається, тобто процес, спрямований на досягнення цього стану. Метою реалізації інформаційної безпеки будь-якого об'єкта чи системи є побудова системи забезпечення інформаційної безпеки даного об'єкту.

Загроза безпеці являє собою сукупність факторів та умов, які виникли в процесі взаємодії об'єкта системи з іншими об'єктами, а також його компонентів між собою, які здатні спричинити негативні наслідки.

Між загрозою і небезпекою нанесення шкоди існують певні зв’язки які саме призводять до заподіяння конкретної шкоди або збитків, які визначаються зв'язок між явищами, при якій одне явище, а саме причинне, за існування певних умов неминуче спричиняє інше явище, а саме наслідкове. Загроза завжди породжує небезпеку. Небезпека може бути визначена як стан, в якому знаходиться об'єкт внаслідок виявлення загрози. Оскільки без виявлення не можливо однозначно визначити що об’єкт знаходить під загрозою в тому чи іншому випадку. Відмінність між ними полягає в тому, що небезпека є властивістю об'єкта

безпеки, а загроза - властивістю об'єкта взаємодії. Очікуваним результатом

загрози є шкода - наслідок зміни умов існування об'єкту в негативну сторону. Загрози інформаційній безпеці - це явища, які можуть призвести до порушень інформаційної безпеки системи. Види загроз інформаційної безпеки різноманітні і мають різні класифікації[7]:

- За характером порушення:
    1. Порушення конфіденційних даних
    2. Порушення працездатності системи
    3. Незаконне втручання в функціонування системи
- За тяжкістю порушення:
    1. Незначні помилки
    2. Дрібне хуліганство
    3. Серйозний злочин
- За мотивацією:
    1. Зловмисне
    2. Незловмисне
- За місцем виникнення:
    1. Зовнішні загрози
    2. Внутрішні загрози
- За об’єктом впливу:
    1. Загрози, націлені на інформаційну систему в цілому
    2. Загрози, націлені на окремі компоненти інформаційної системи
- За причиною виникнення:
    1. Загрози, що виникли в результаті недостатніх засобів технічного захисту

2. Загрози, що виникли через недостатність або слабку організованість організаційно-правових мір

- За каналом проникнення:
    1. Через слабкість програмного-апаратного комплексу системи
    2. Через прогалини в системі автентифікації, авторизації та недоліки системи зберігання даних користувачів

Згідно представленої класифікації загроз за видом об'єкта впливу вони поділяються на:

- загрози інформації,
- загрози персоналу об'єкта
- загрози власне забезпеченню інформаційної безпеки об'єкта.

При розробці необхідних, засобів, методів і заходів, що забезпечують захист інформації, необхідно враховувати велику кількість різних факторів.

Інформація, яка у даному випадку є предметом захисту, може бути представлена на різних технічних носіях, а також її носіями можуть бути користувачі і обслуговуючий персонал. Вона може піддаватися обробці в комп'ютерних системах, передаватися по каналах зв'язку і відтворюватись різними пристроями. Таким чином, для забезпечення інформаційної безпеки існують наступні принципи: системність, комплексність та безперервність захисту інформації; гнучкість управління і застосування; відкритість алгоритмів і механізмів захисту; простота застосування захисних засобів і заходів.

За способами здійснення всі заходи забезпечення безпеки комп'ютерних систем підрозділяють на:

- нормативно-правові(діючі закони, укази та нормативні акти);

- морально-етичні(норми поведінки);
- організаційно-адміністративні(регламентація процесів функціонування інформаційних систем - ІС);
- апаратно-програмні.

Особливу увагу необхідно приділити апаратно-програмним заходам захисту. До них відносяться електронні пристрої та спеціальні програми, які реалізуються самостійно або в комплексі з іншими. Можна виділити такі основні способи захисту:

- автентифікацію та авторизацію суб'єктів ІС;
- розмежування доступу до ресурсів ІС;
- контроль цілісності даних;
- забезпечення конфіденційності даних;
- аудит подій, що відбуваються в ІС;
- резервування ресурсів і компонентів ІС.

1.2 Системи документообігу

В сучасному світі системи документообігу займають невід’ємну частину життя інтернет користувачів. Навіть якщо не користуватись послугами таких систем на пряму то не можливо без них обійтись використовуючи месенджери і поштові сервіси.

Система файлообміну – це сукупна назва мереж для спільного використання файлів, система надає місце для файлу на власних серверах повернувши посилання на доступ до нього. Доступ до файлу можливий в залежності від встановлених налаштувань самої системи та побажань користувача який його завантажив. Особливості налаштувань політики доступу до файлу що був

завантажений притаманні платним сервісам. Існує два види надання послуг таких систем, а саме з використанням спеціалізованих комп'ютерних застосунків або з використанням інтернет ресурсів розміщених на віддалених чи хмарних серверах, які підтримуються організаціями по наданню відповідних послуг.

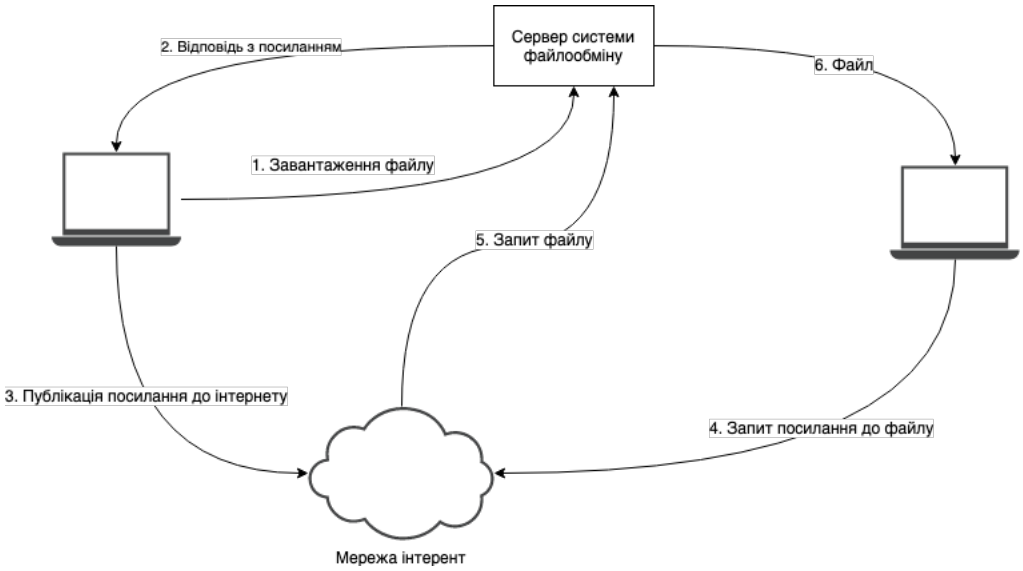

Рисунок 1.1 – Схема роботи системи файлообміну

Переваги систем з використання попередньо встановленого ПЗ:

- Стабільна робота
- Низьке споживання інтернет трафіку
- Швидкий доступ до інструментів системи

Недоліки систем з використання попередньо встановленого ПЗ:

- Обмежений спектр систем на яких можливе використання
- Потрібні налаштування конфігурацій при встановлені
- Система використовується тільки зареєстрованими користувачами
- Не можливе використання з мобільних построїв
- Вартість розробки та підтримки продукту

Переваги систем з використанням інтернет ресурсів:

- Доступ з будь якого пристрою з доступом в інтернет та файловою системою
- Можливе використання анонімними користувачами
- Дешевша розробка та підтримка продукту в порівняні з комп'ютерним застосунком

Недоліки систем з використанням інтернет ресурсів не знайдено.

Очевидно що користування системами файлообміну які використовують інтернет ресурси для надання послуг значно вигідніше як користувачам так і власникам системи.

Принципи роботи системи файлообміну:

- Підключення користувача до клієнтської частини
- Завантаження необхідного файлу в систему і його подальше завантаження на серверну частину
- Надання доступу до файлу іншим користувачам системи

Система документообігу – це вузькоспеціалізована система файлообміну основна ціль якої саме передача текстових файлів необхідного розширення, наприклад pdf, doc, txt. Також, основною відмінністю такої системи є жорсткіші вимоги до рівня захисту ніж в порівнянні зі звичайною системою файлообміну, оскільки інформація яка може міститься в документах буде відноситися до приватної, конфіденційної, державної таємниці.

1.3 Загрози безпеці та методи захисту систем документообігу

Підхід до захисту електронного документообігу повинен бути комплексним. Необхідно чітко оцінювати можливі загрози і ризики СЕД і можливі втрати від реалізованих загроз. Традиційний підхід до захисту інформації заснований на попередньому аналізі загроз і зіставленні їм сукупності механізмів захисту. Основні загрози для систем електронного документообігу можуть бути класифіковані таким чином [8]:

- загроза цілісності
- загроза конфіденційності
- загроза працездатності системи – це загроза, реалізація якої призводить до пошкоджень, припинення роботи системи. До таких загроз входять навмисні атаки, помилки користувачів, а також несправності в технічному обладнанні та програмному забезпеченні;
- неможливість доказу авторства – це загроза, основна суть якої полягає у тому, при відсутності використання електронно цифрового підпису в системі документообігу, то неможливо довести, що саме цей користувач створив даний документ. До того ж, неможливо зробити документообіг юридично значимим;
- загроза доступності.

Будь-яка система електронного документообігу повинна бути забезпечена захистом в різних проявах та реалізаціях від переліку загроз які були наведені вище. Відповідно, в комплексний захист такої системи повинні входити наступні заходи [9]:

- обмеження прав фізичного доступу до об’єктів системи документообігу;
- розмежування прав доступу до файлів і папок;
- підтвердження авторства електронного документу;
- контроль цілісності електронного документу;
- конфіденційність електронного документу;
- забезпечення юридичної сили електронного документу;
- забезпечення надійності функціонування технічних засобів;
- забезпечення резервування каналів зв’язку;
- резервне дублювання інформації;
- захист від вірусів;
- захист від "злому" мереж.

Для реалізації системи захисту даних і управління доступом відповідних файлів необхідне використання систем ідентифікації та автентифікації:

- ідентифікація користувача – процедура присвоєння користувачеві ідентифікатора в системі за рахунок наперед визначеного значення, тобто логін або електрона адреса.
- автентифікація - встановлення достовірності суб'єкта, що саме він є тим ким представляється.

Система ідентифікації і автентифікації користувачів системи є необхідним і важливим елементом будь-якої системи захищеного електронного документообігу.

На додаток, можна вважати, що загальними задачами для організації захищеного електронного документообігу з використанням систем ідентифікації і автентифікації є:

- Жорстка процедура ідентифікації і автентифікації користувачів для організації доступу до важливих ресурсів, що захищаються;
- Розмежування доступу до конфіденційної інформації і персональних даних;
- Блокування і зменшення шансу несанкціонованого доступу;
- Забезпечення доступності публічної інформації.

Також, важливо звернути увагу на методи ідентифікації і автентифікації користувачів мобільних систем. Найрозповсюдженіший – це парольний. Головна перевага парольної автентифікації - це простота реалізації та використання. Найбільша проблема, яка сильно понижує надійність даного способу - це людський чинник. Існує декілька факторів які роблять людину найслабшою ланкою в цьому ланцюзі безпеки. Перший з них – це бажання людей спростити пароль для простішого запам’ятовування. Більшість людей використовують ненадійні ключові слова, які легко підбираються звичайним брутфорсом або списком найрозповсюдженіших паролів. Цього можна запобігти завдяки впровадженню політики паролів, яка встановлює правила створення, а саме:

- довжину пароля
- використання спеціальних символів
- обмежений період використання пароля і примусова зміна при завершені строку дії попереднього
- неможливість повторного використання пароля впродовж заданого періоду часу.

Другий факт це можливість зловмисників отримати персональні дані користувачів завдяки соціальній інженерії, яка включає в себе фішинг, претекстинг, телефоний фішинг, дорожнє яблуко.

Апаратний принцип ідентифікації ґрунтується на ідентифікації особи користувача завдяки фізичному предмету, ключу, що знаходиться тільки в його персональному користуванні [10]. На сьогоднішній день, найбільше поширення одержали такі типи фізичних носії: різноманітні карти (проксіміті карти, смарт-карти, магнітні карти) та токени, які підключаються безпосередньо до одного з фізичних портів комп'ютера. Головною перевагою застосування апаратної ідентифікації є достатньо висока надійність. Проте, серйозною загрозою є можливість крадіжки зловмисниками фізичних носіїв токенів або навіть самих проксіміті карт. Також вони можуть бути втрачені, пошкоджені, передані іншій особі, дубльовані.

Найбільш надійний спосіб ідентифікації і автентифікації – це біометричний, при якому користувач ідентифікується за персональними біометричними даними, до переліку можливих варіантів входять: відбиток пальця, сканування сітківки ока, голос [11]. Проте, вартість системи з використанням таких технологій значно зростає, а сучасні біометричні технології ще не достатньо досконалі, щоб уникнути помилкових спрацьовувань або відмов.

Один з найважливіших факторів ідентифікації та автентифікації – це багаторівневість. Тобто, ці процеси можуть бути однофакторними або багатофакторними, коли для визначення особи та верифікації користувача застосовується відразу кілька параметрів [12]. Таке комбінування шарів можливе в різних варіація включаючи вже перечисленні методи, а також смс

повідомлення на контактний номер користувача, електронний лист на приватну електронну адресу. Найчастіша комбінація на сьогоднішній день вважається парольний метод і лист підтвердження на електрону пошту. Впровадження комбінованих систем збільшує кількість необхідних ідентифікаційних ознак і тим самим значно підвищує рівень безпеки і захисту систем електронного документообігу

1.4 Центри сертифікації

Для реалізації захищеної системи документообігу необхідна реалізація яка включає в себе гарантію, що відправник насправді являється тим від кого надійшов документ. Найбільш вдало з цією задачею справляється центр сертифікації, а саме інтеграція зі вже існуючими чи створення власного для відповідності вимогам.

Центр сертифікації – акредитований державою відділ чи організація яка займається послугами з надання електронно цифрових підписів, чия чесність вважається еталонною і не заперечною. В подальшому, підписи які будуть здійснені виданим ЕЦП, можуть бути підтвердженими або спростованими центром сертифікації підчас перевірки.

Реєстр – електронна база даних, в якій містяться відомості про самопідписані сертифікати електронної печатки ЦЗО, сертифікати ЦЗО для додавання електронної печатки до Довірчого списку та до даних у протоколі визначення статусу сертифіката у режимі реального часу, сертифікати кваліфікованих надавачів електронних довірчих послуг (далі – надавачі), сформовані з використанням самопідписаного сертифіката електронної печатки ЦЗО, статус

та обмеження у використанні таких сертифікатів, а також списки відкликаних сертифікатів ЦЗО [13].

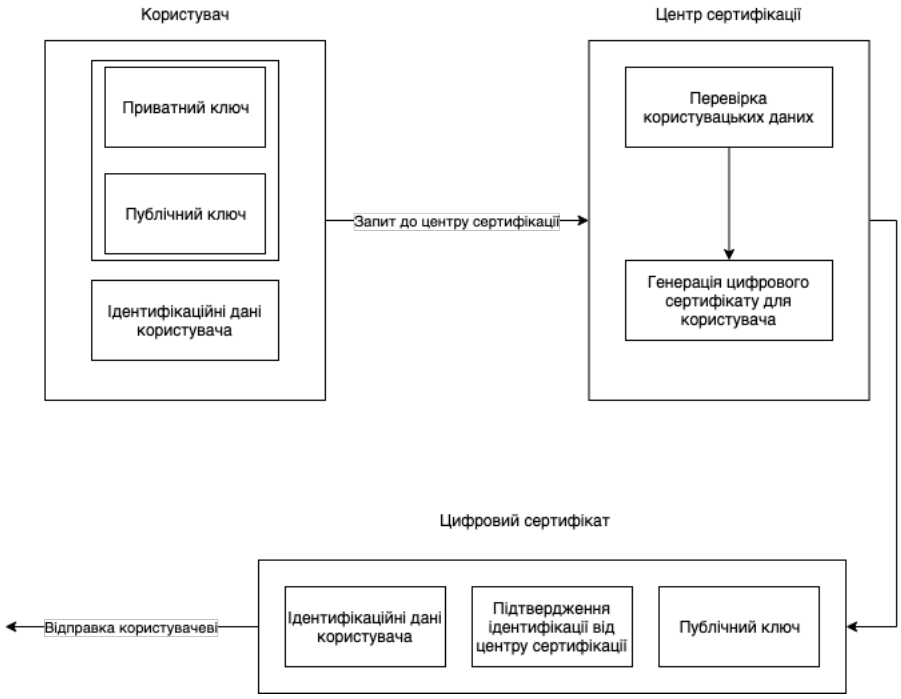

Рисунок 1.2 – Схема роботи центру сертифікації

1.4.1 Про центральний засвідчувальний орган

Відповідно до Закону України «Про електронні довірчі послуги» до складу електронних довірчих послуг входять:

- створення, перевірка та підтвердження удосконаленого електронного підпису чи печатки;
- формування, перевірка та підтвердження чинності сертифіката електронного підпису чи печатки;
- формування, перевірка та підтвердження чинності сертифіката автентифікації веб-сайту;
- формування, перевірка та підтвердження електронної позначки часу;
- реєстрована електронна доставка;

- зберігання удосконалених електронних підписів, печаток, електронних позначок часу та сертифікатів, пов’язаних з цими послугами.

Суб’єктами відносин у сфері електронних довірчих послуг є:

- користувачі електронних довірчих послуг;
- надавачі електронних довірчих послуг;
- органи з оцінки відповідності;
- засвідчувальний центр;
- центральний засвідчувальний орган;
- контролюючий орган.

Електронні довірчі послуги надаються, як правило, на договірних засадах надавачами електронних довірчих послуг. Кожна послуга, що входить до складу електронних довірчих послуг, може надаватися як окремо, так і в сукупності.

Кваліфікований надавач електронних довірчих послуг - юридична особа незалежно від організаційно-правової форми та форми власності, фізична особа - підприємець, яка надає одну або більше електронних довірчих послуг, діяльність якої відповідає вимогам Закону України «Про електронні довірчі послуги» та відомості про яку внесені до Довірчого списку.

Міністерство юстиції України як головний орган у системі центральних органів виконавчої влади, що забезпечує формування та реалізує державну політику у сфері електронних довірчих послуг, щодо юридичних осіб, фізичних осіб - підприємців, які мають намір надавати електронні довірчі послуги, виконує функції центрального засвідчувального органу шляхом внесення відомостей про них до Довірчого списку відповідно до вимог цього Закону.

Повноваження центрального засвідчувального органу визначено у Статті 7 Закону України «Про електронні довірчі послуги». До повноважень

центрального засвідчувального органу, по відношенню до кваліфікованих надавачів електронних довірчих послуг належить надання таких послуг:

- адміністративна послуга внесення юридичних осіб та фізичних осіб - підприємців, які мають намір надавати електронні довірчі послуги, до Довірчого списку;
- надання кваліфікованих електронних довірчих послуг надавачам електронних довірчих послуг з використанням самопідписаного сертифіката електронної печатки центрального засвідчувального органу, що призначений для надання таких послуг;
- надання послуги постачання передачі сигналів точного часу, синхронізованого з Державним еталоном одиниць часу і частоти.

Технічне та технологічне забезпечення виконання функцій центрального засвідчувального органу здійснюється адміністратором інформаційно-телекомунікаційної системи центрального засвідчувального органу - державним підприємством, яке належить до сфери управління головного органу у системі центральних органів виконавчої влади, що забезпечує формування та реалізує державну політику у сфері електронних довірчих послуг (державним підприємством «Національні інформаційні системи») [14].

1.5 Електронно цифровий підпис

У цьому розділі перелічені головні риси та принципи побудови систем електронного цифрового підпису.

Електронний цифровий підпис (ЕЦП) – це блок інформації, який додається до даних автором тобто підписувачем та захищає файл від несанкціонованої модифікації і вказує на власника підпису.

Для функціонування ЕЦП використовуються 2 ключі захисту:

- Таємний ключ, який зберігається у підписувача на фізичному носії
- Відкритий ключ, який, публікується у загальнодоступному доступі і є відкритою інформацією.

Для накладання ЕЦП використовується приватний ключ власника, а для його перевірки – відкритий, загальнодоступний ключ.

Принцип такої дії алгоритму зумовлений важкооборотними криптографічними перетвореннями. Тобто завдяки публічному ключу не вийде підписатися, а приватним не можливо перевірити.

Доцільно зазначити, що приватний ключ підписувача не дублюється, не зберігається ніде окрім фізичного носію власника і є повною особистою власністю підписувача і не надається будь-яким іншім особам, включаючи навіть центр сертифікації ключів. Будь-хто може перевірити цифровий підпис, використовуючи тільки відкритий ключ.

Накладання електронного цифрового підпису - це операція, яка здійснюється відправником, тобто власником, документу із використанням його приватного ключа. Під час виконанні даної операції на вхід подаються дані, які необхідно підписати, та приватний ключ підписувача. Система конвертує дані за допомогою приватного ключа унікальний блок даних фіксованого розміру, який може бути істинним тільки для цього персонального ключа та саме для цих вхідних даних, тобто на виході отримується ЕЦП.

У подальшому ЕЦП, передається у парі з вхідним документом або надсилається окремо і тоді така комбінація даних утворює захищений електронний документ.

Перевірка електронного цифрового підпису – це операція, яка виконується отримувачем захищеного електронного документу з використанням відкритого ключа підписувача, тобто відправника. Для виконання цієї операції необхідно мати в розпорядженні відкритий ключ відправника, отримавши його разом з документом чи з відритого доступу і захищений документ. Відповідна частина системи перевіряє, чи дійсно цифровий підпис відповідає документу та відкритому ключу. Якщо документ або відкритий ключ були модифіковані, перевірка закінчиться із негативним результатом і буде встановлено не відповідність.

Для повноцінного функціонування систем ЕЦП необхідно забезпечити отримувача достовірною інформацію що до відкритого ключа відправника та можливість впевнитися, що ця копія відкритого ключа належить саме цьому відправнику. Для виконання таких умов, існують спеціальні захищені довідники ключів, які ведуться спеціальними установами – центрами сертифікації ключів.

Центри сертифікації ключів перевіряють дані власника відкритого ключа та видають захищені електронні документи спеціального зразка – сертифікати відкритих ключів. В них містяться відкритий ключ та перевірена центром сертифікації інформація про власника ключа. Сертифікат відкритого ключа підписується електронним цифровим підписом центру сертифікації ключів. Тоді достатньо отримати захищеним та достовірним каналом лише сертифікат самого центру сертифікації ключів, щоб мати можливість перевірити достовірність будь-якого сертифікату, який був виданий цим центром.

Генерація приватного та відкритого ключів ЕЦП - початкова процедура, яка виконується користувачем до виконання процедури сертифікації відкритого

ключа. Генерацію виконує спеціалізоване програмне забезпечення – генератор ключів, який надається центром сертифікації ключів [15].

Висновки до першого розділу

Проведено аналіз структури систем документообігу, центрів сертифікації та принципів їх взаємодії між собою і користувачем. Також проаналізовано можливі загрози безпеці таких систем і методи, які забезпечують зменшення ризику нанесення шкоди у разі успішної реалізації загроз. Визначено доцільність розроблення захищеної системи документообігу, яка буде складатись з незалежного центру сертифікації та мобільного застосунку для взаємодії з користувачами системи.

2 АНАЛІЗ ІСНУЮЧИХ СИСТЕМИ ДОКУМЕНТООБІГУ НА МОБІЛЬНІЙ ПЛАТФОРМІ

На сьогоднішній день існує декілька мобільних застосунків які позиціонують себе як системи документообігу які використовують електронний підпис для верифікації користувачів, а також інтерфейс для контролю отриманих та відправлених документів у системі. Весь аналіз функціоналу застосунків буде проводить на мобільній операційній системі Android. Проте важливо зазначити, що перевірка існування застосунку на мобільній операційній системі iOS є також необхідною, оскільки відмова від обслуговування значної кількості потенційних користувачів не може позитивно відзначитись.

2.1 PandaDoc

Даний застосунок створений більше для роботи з документами в цілому, проте невід'ємною його частиною є верифікація відправника за його електронним підписом, оскільки розробники застосунку роблять акцент саме на документах купівлі та продажу.

Після встановлення мобільного застосунку з Play Market, користувачів зустрічає форма входу в систему, проте можливості зареєструватись не можливо, оскільки цей функціонал винесений до десктопної версії застосунку. В контексті користувача, таких інтерфейс вважається не зручним, оскільки потрібні додаткові дії, які можливо виконати і на мобільній платформі, оскільки сама форма не містить особливих полів для заповнення чи специфічного функціоналу.

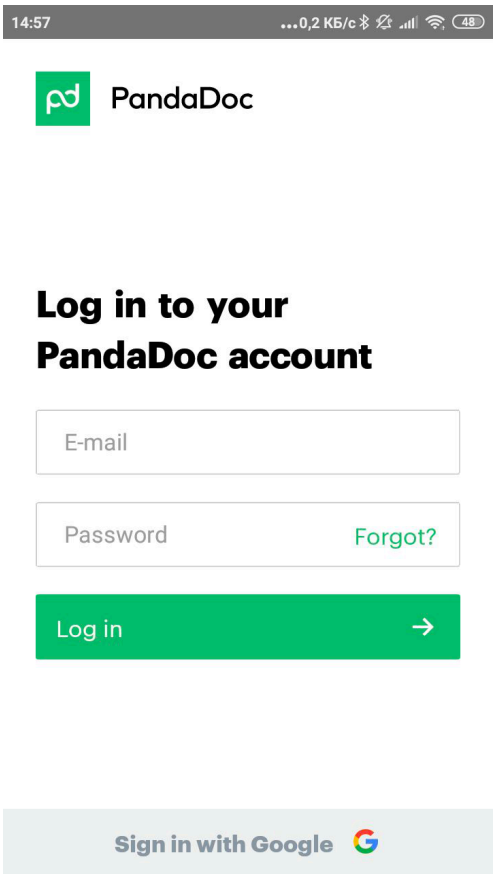

Рисунок 2.1 – Форма входу в систему

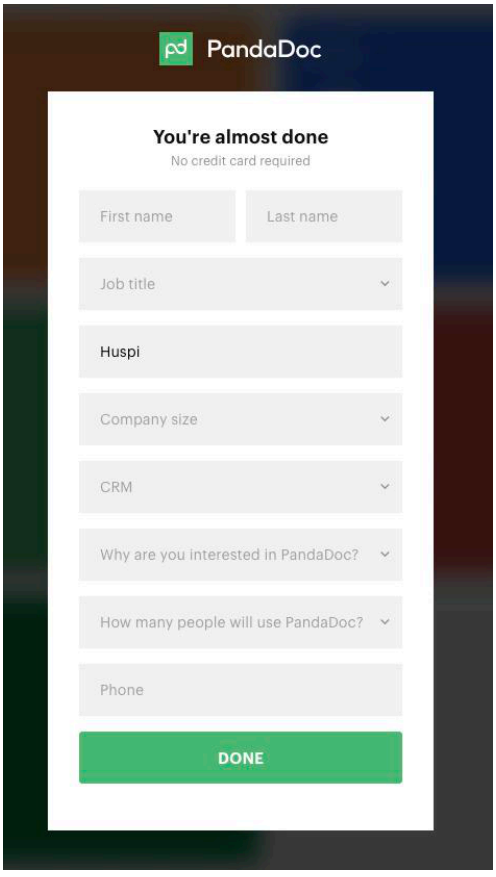

Рисунок 2.2 – Форма реєстрації в десктопному застосунку

Після входу в систему користувач знаходиться на головній сторінці, яка наповнена звичним інтерфейсом управління мобільного застосунку. А саме: нижнє меню навігації і головний екран на якому відображається функціонал вибраної вкладки з нижнього меню.

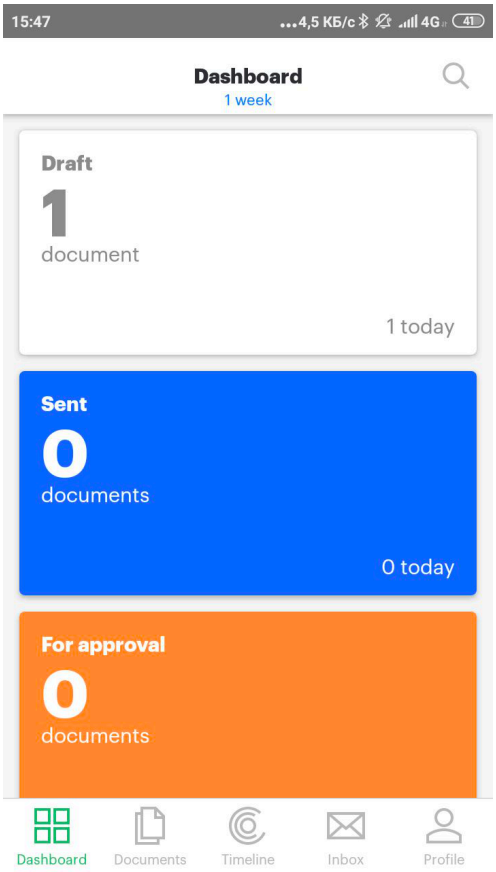

Рисунок 2.3 – Головне меню мобільного застосунку

Проте на відміну від заявленого функціоналу, реалізованим є тільки частина. На сам перед, система не дозволяє підписувати і відправляти документи які вже знаходяться у файловій системі пристрою. Операції в застосунку можливо виконувати тільки зі створеними, за його допомогою, документами. Проте навіть тут є обмеження. Документ повинен бути створений з шаблону, який був затверджений користувачем раніше. На момент аналізу даного застосунку, не вдалося створити документ з шаблону через несправності роботи системи, так як список шаблонів був пустим.

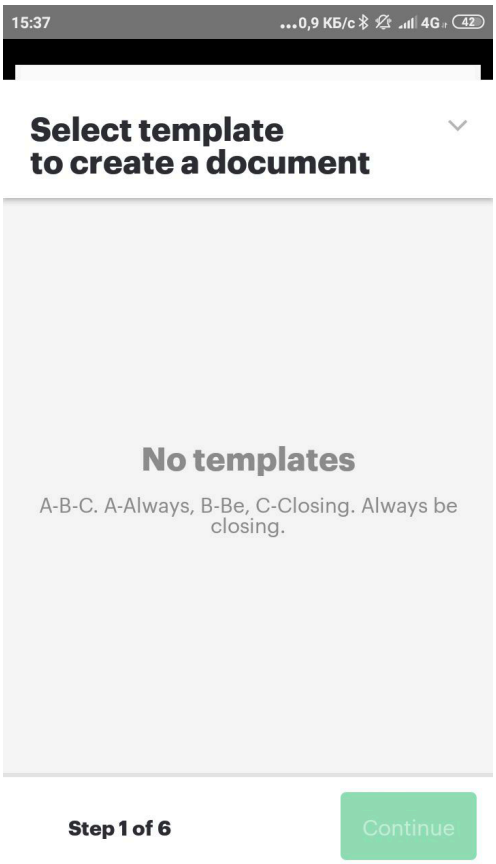

Рисунок 2.4 – Неможливість створити документ

Але на відміну від мобільного застосунку, відправити документ вдалося, навіть з електронним підписом, хоча це не достовірний метод підпису. А саме, персональний підпис людини за допомогою пальця на екрані. Далі підпис розміщується на поля документу і тоді він вважається «підписаним». Хоча інструменти для редагування документів складаються з багатьох необхідних інструментів, проте це не повинно бути поставлене вище ніж безпека їх передачі.

Застосунок містить зручну історію взаємодії з документами. Вона відображає назву, автора, час та дату створення, час та дату подальшого редагування, час та дату відправлення і статус самого документу, такі як: чернетка, надіслано, редагується, очікує підтвердження, переглянуто, завершено, прострочено, чекає оплати, оплата підтверджена, відхилено.

Рисунок 2.5 – Електронний підпис застосунку

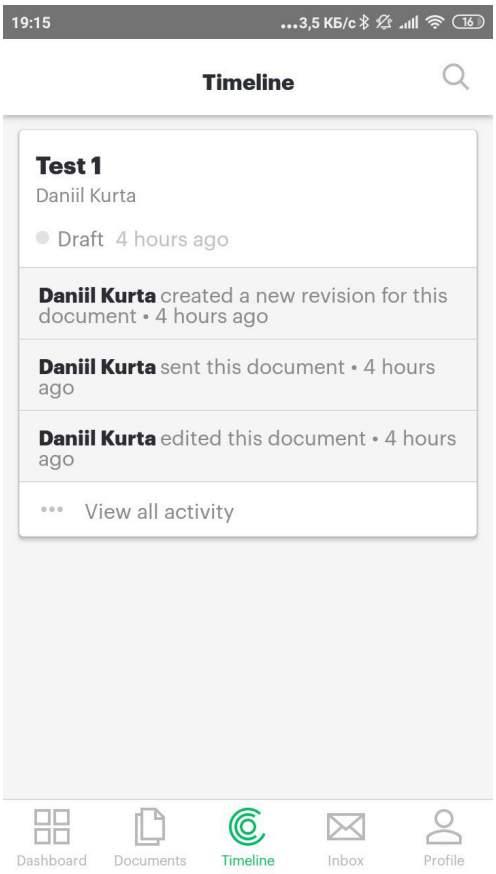

Рисунок 2.6 – Історія взаємодії з документами

В меню налаштувань профілю можливо змінити підпис, встановлення паролю безпеки застосунку, зміна облікового запису та форма для взаємодії з командою підтримки застосунку.

2.2 UKey [16]

Даний застосунок є реалізацією української групи розробників і являє собою дуже зручний і продуманий продукт позбавлений явних недоліків в безпеці. Насамперед реєстрація в системі можлива одразу через застосунок, що в порівнянні з попереднім великий плюс.

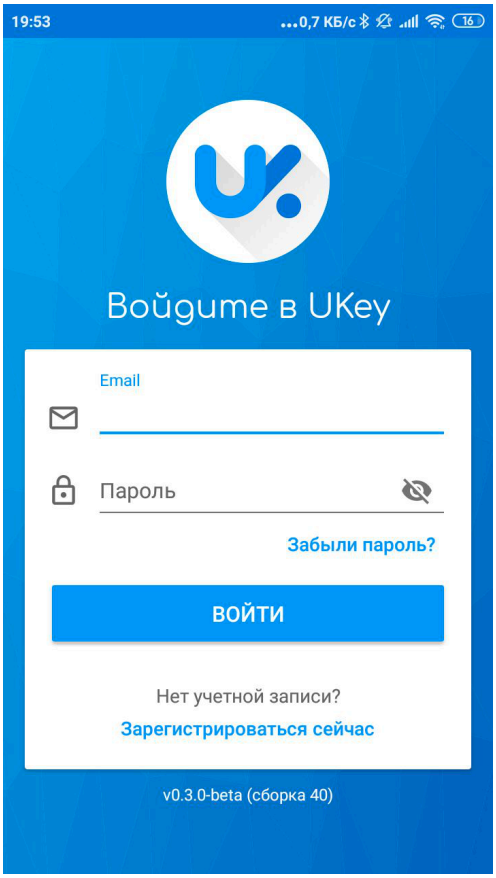

Рисунок 2.7 – Форма входу в систему

Після реєстрації і входу в систему для оцінки можливостей застосунку необхідно завантажити принаймні один файловий ключ розширення .dat, .jks, .p12, .pfx, .zs2. Для цього необхідне принаймні одне підключення до комп'ютеру для передачі файлу ключа, оскільки пряме завантаження у файлову систему застосунку неможливе. Функціонал взаємодії з такою кількістю ключів вражає.

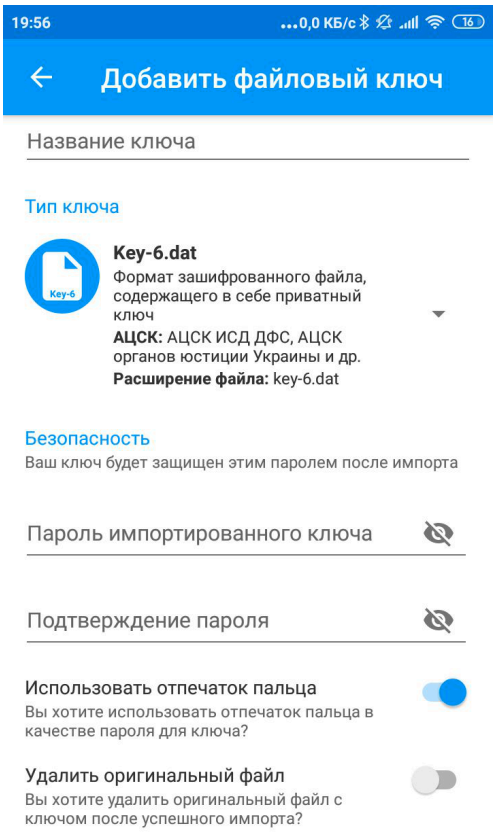

Рисунок 2.8 – Форма налаштування створення повідомлення

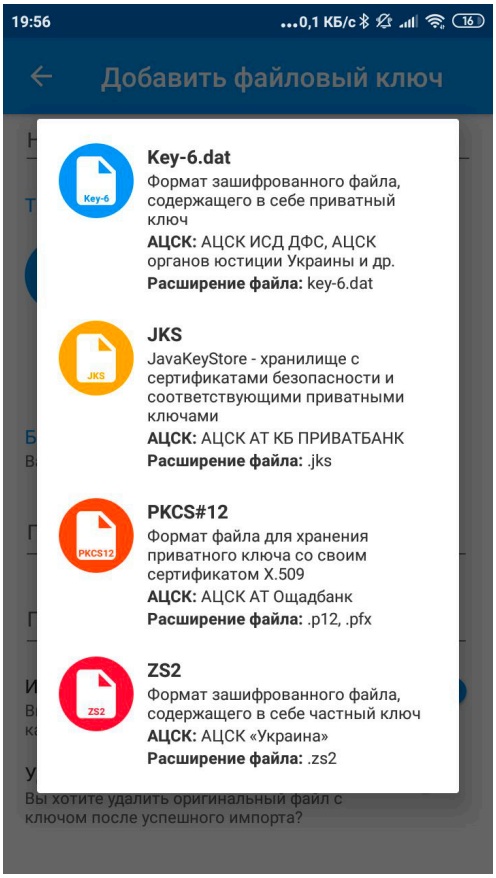

Рисунок 2.9 – Вибір одного з запропонованих розширень ключів

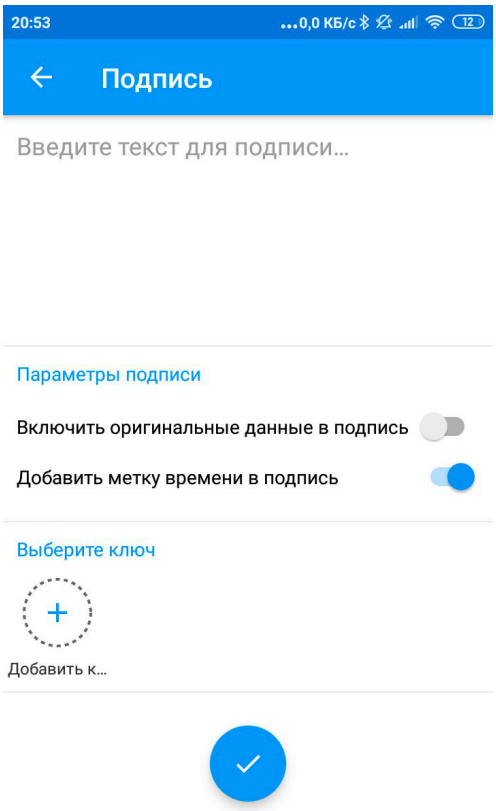

Рисунок 2.10 – Форма створення повідомлення

Повідомлення може бути як просто текстове, так і файл з локальної файлової системи мобільного пристрою. Після створення запису про ключ в застосунку, а саме імпорт необхідного файлу, створення та підтвердження пароля для можливості користуватись цим записом, необхідно ввести пароль від сертифікату, інакше його використання буде не можливе. При закінченні процесу підпису інформації, користувач побачить всю інформацію про результат цього процесу. На екрані буде відображено сам підпис, інформацію яка була підписана і посилання для поширення серед інших користувачів. Поширення включає в себе підпис і текст повідомлення, тому будь який бажаючий зможе перевірити підпис завдяки публічному ключу який можна знайти у відкритих списках акредитованого центру сертифікації ключів.

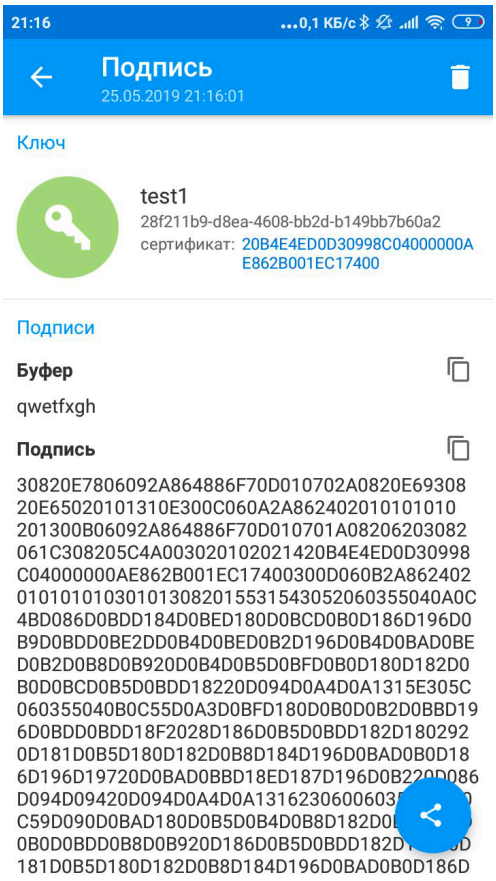

Рисунок 2.11 – Форма створення повідомлення

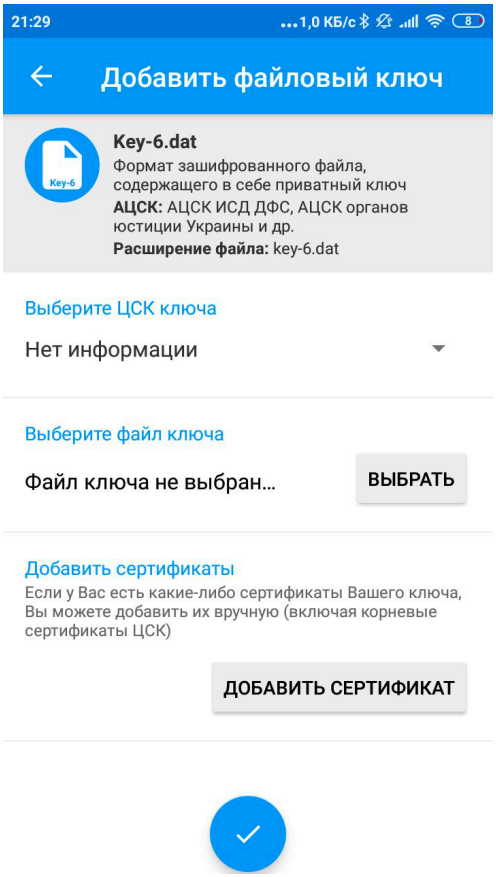

Рисунок 2.12 – Форма імпорту сертифікату ключа

Висновки до другого розділу

Проведено аналіз захищеності існуючих систем документообігу на мобільній платформі, насамперед систем, що використовують платформу Android. Більшість програм, які розміщені в Play Market, описують свій функціонал як достатній для того, щоби гарантувати безпеку системи і верифікацію відправника. Однак реалізація підпису у цих системах полягає в розміщенні візуального підпису безпосередньо на самому документі, що не відповідає вимогам захисту електронних документів. Єдиний прототип системи, що позбавлений цього недоліку і використовує справжній електронний цифровий підпис на основі ключів, які видає АКЦС, розроблено в Україні. На поточний момент застосунок знаходиться ще в режимі тестування. Недоліком є необхідність завантаження ключа сертифікату через сторонні канали, що може призвести до втрати або модифікації ключа зловмисниками.

3 РЕАЛІЗАЦІЯ ЗАХИЩЕНОЇ СИСТЕМИ ДОКУМЕНТООБІГУ НА МОБІЛЬНІЙ ПЛАТФОРМІ

Розглянуті системи захищеного документообігу містять як переваги так і недоліки. Для розробки більш захищеної системи, необхідно звернути увагу на методи реєстрації користувачів, а також зберігання і передачу їх персональних даних включно з приватним і публічним ключем.

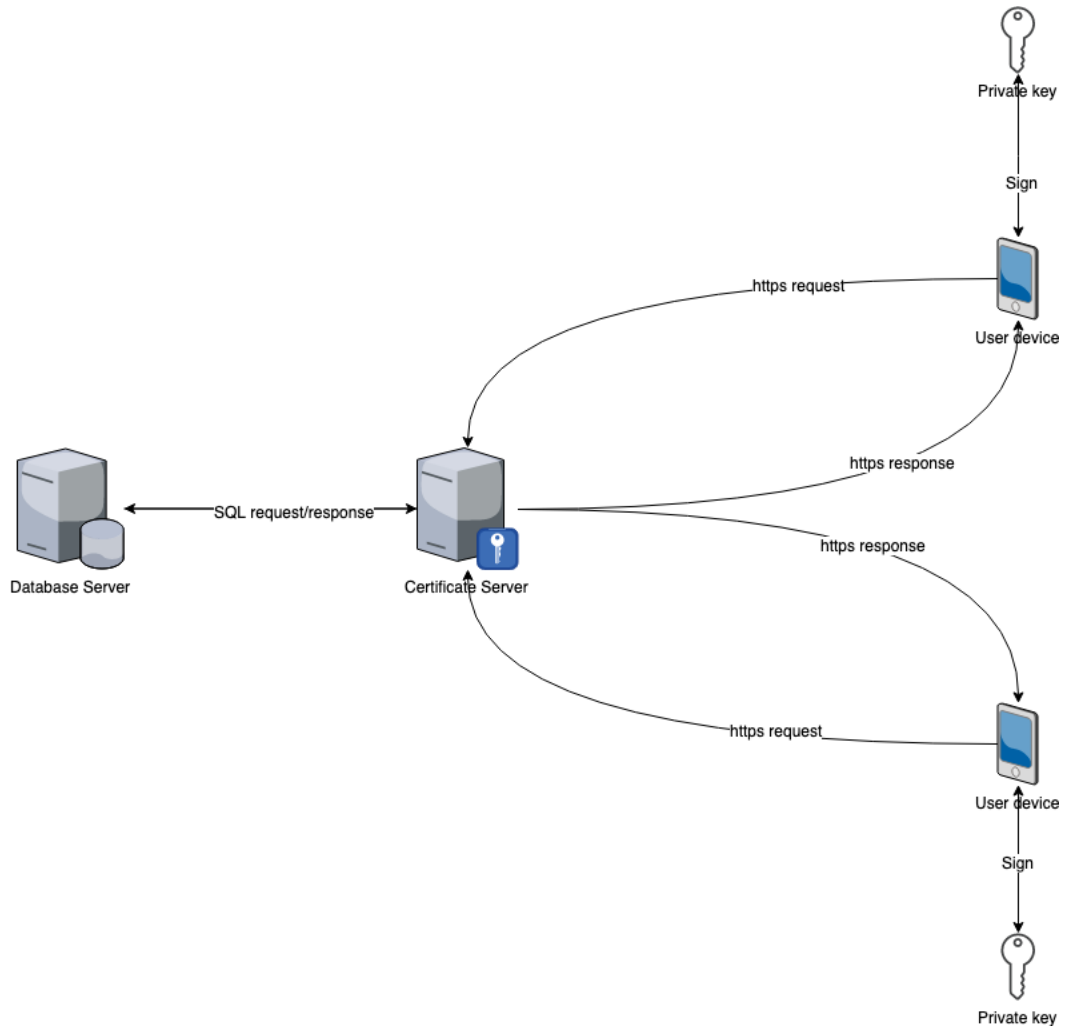

Рисунок 3.1 – Схема роботи системи

3.1 Мобільний застосунок

Мобільний застосунок був розроблений з використання бібліотеки React Native[17] для розробки одного коду під дві різні платформи, що значно знижує витрати на розробку та подальшу підтримку продукту під час його життєвого циклу.

Під час реєстрації в системі необхідно вказати логін, електронну адресу користувача для підтвердження облікового запису, інакше вхід в систему під цим логіном буде не можливий, ім’я, фамілію, пароль, підтвердження паролю.

Форма логіну містить форму для введення комбінації логіну і паролю користувача, а також можливість зареєструватись у системі у разі відсутності облікового запису.

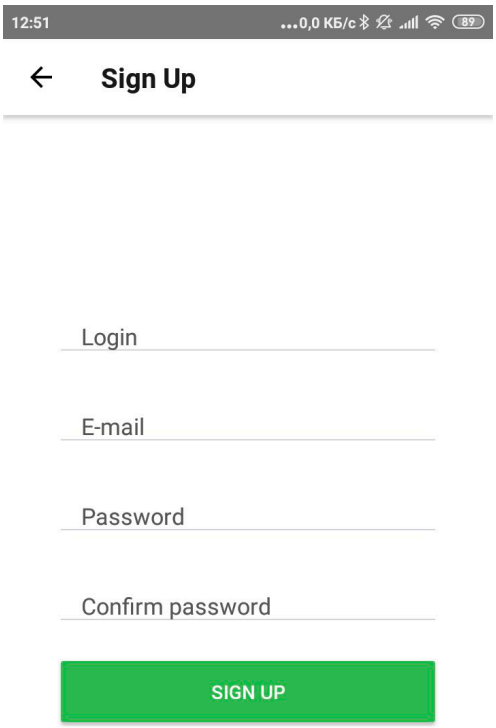

Рисунок 3.2 – Форма реєстрації в системі

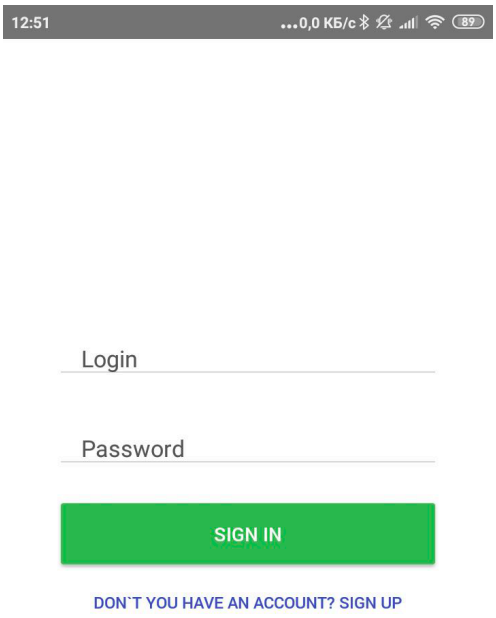

Рисунок 3.3 – Форма входу в систему

Після автентифікації та авторизації, користувач отримує можливість повністю користуватись функціоналом застосунку.

Насамперед функціонал відправки повідомлень відбувається через вибір отримувача зі списку, вибір файлу розширення .pdf з файлової системи пристрою. Після цього відбувається підпис документу і відправлення відповідному користувачеві.

Необхідно зазначити, що у зв’язку з закритою файловою системою операційної системи iOS неможливо отримати перелік файлів на пристрої. Застосунок повинен отримати підтвердження від користувача на використання облікового запису iCloud і взаємодіяти з його файлами. На системі Android

застосунок також зобов’язаний отримати підтвердження від користувача, проте після цього всі файли необхідного розширення будуть відображені в системі.

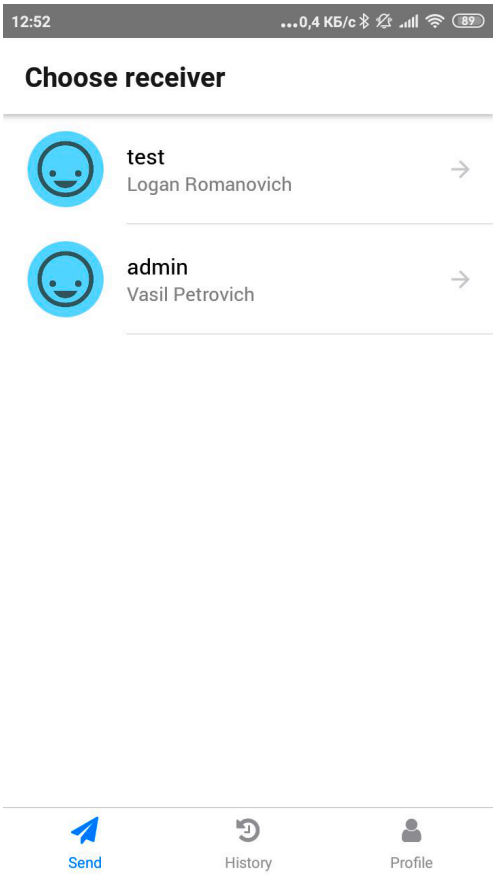

Рисунок 3.4 – Форма вибору отримувача

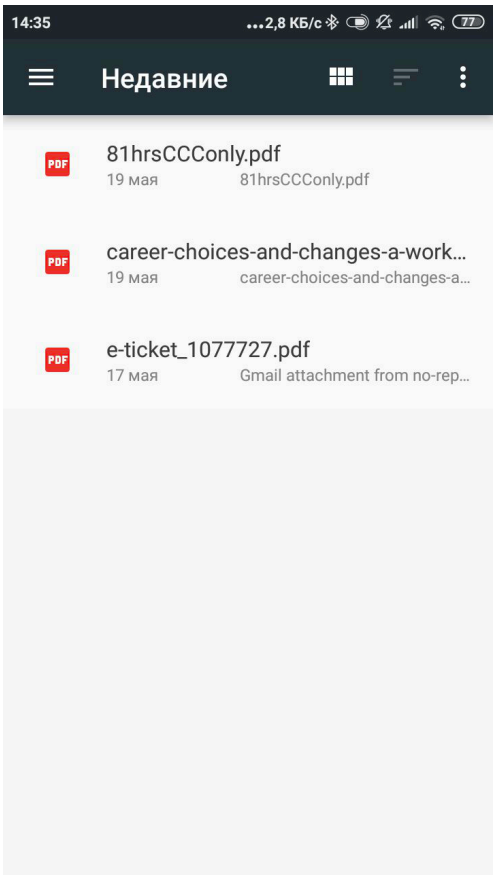

Рисунок 3.5 – Форма вибору документу

Після успішного надсилання, як відправник так і отримувач з легкістю зможуть переглянути історію надісланих документів, їх вміст а також верифікацію підпису яка виражається в усміхнених або сумних обличчях в правій колонці.

Потрібно зауважити, що для підпису в поточній реалізації використовується алгоритм RSA з довжиною ключа 2048 біт, для демонстрації роботи системи в цілому без інтеграції з існуючими центрами сертифікації. Такий вибір алгоритму зумовлений його швидкістю перевірки підпису, навіть тоді коли швидкість його генерації доволі низька, але це не критично оскільки генерація відбувається тільки один раз, в порівнянні з перевіркою, яка відбувається на порядок частіше.

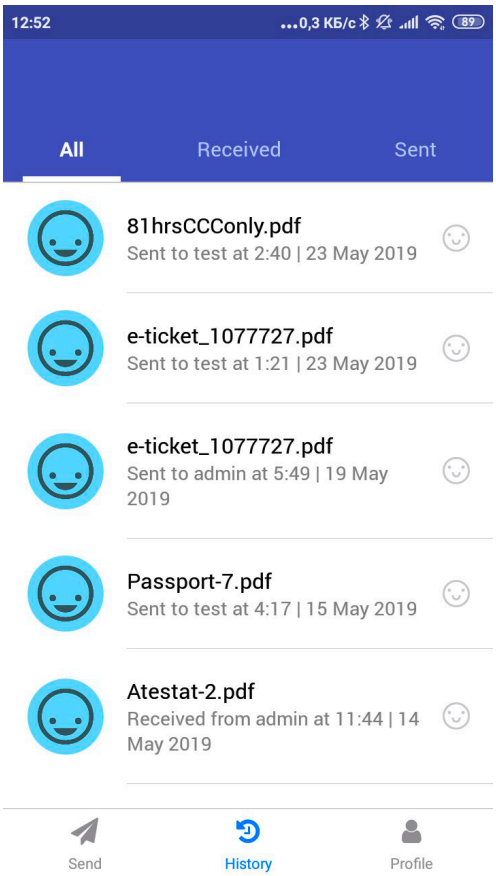

Рисунок 3.6 – Форма історії надсилань

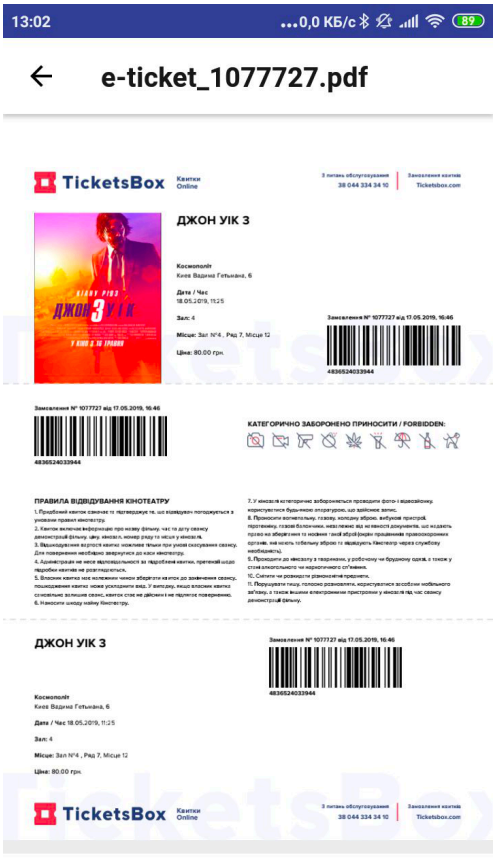

Рисунок 3.7 – Перегляд надісланого повідомлення

Основна перевага мобільного застосунку в порівнянні з аналогами – це відсутність передачі приватного ключа користувача сторонніми каналами для імпорту його у файлову систему застосунку. Реалізація такого функціоналу була досягнута завдяки тому, що при реєстрації нового користувача пара приватного і публічного ключу генерується виключно на пристрої. При видаленні застосунку з пристрою приватний ключ користувача також видаляється, оскільки це єдина копія яка взагалі існує. Проте така реалізація не забороняє створення нового користувача на цьому ж пристрої, але з обмеженням що до логіну, який не може дублюватись зі вже існуючим, так як логін виступає первинним ідентифікатором користувача.

3.2 Серверна частина

Серверна частина була реалізована завдяки бібліотеці Node.js[18] для зменшення витрат та часу на розробку, підтримку і впровадження логіки до хмарного середовища. Також дана бібліотека значно полегшить інтеграцію з центрами сертифікації.

В поточній реалізації системи, логіка серверної частини знаходиться на хмарному сервісі Heroku для імітації роботи центру сертифікації. Основна задача – це перевірка підпису відправника, оскільки це основна функція центру сертифікації. Також до задач входять:

- Збереження всіх файлів системи
- Збереження інформації про користувачів
- Верифікація токену користувача для взаємодії з сервером
- Розмежування доступу до файлів та інформації різних користувачів

- Методи для реєстрації користувачів
- Методи для зміни інформації користувачів, на приклад паролю

3.3 Подальший розвиток системи

Подальший розвиток застосунку і системи в цілому можливо розділити на декілька етапів в залежності від їх впливу на процес використання системи.

Перший етап розвитку – це інтеграція в систему існуючого центру сертифікації який встановить необхідний йому алгоритм генерації ключів та їх довжину. Надалі публічні ключі яку будуть генеруватись на пристроях користувачів вноситимуться до загальнодоступного списку ключів як другий можливий варіант перевірки відправника. Також необхідно урегулювати одночасне видалення користувачами застосунку з пристрою з блокуванням їх облікового запису для інформування що даний користувач більше не дійсний. Проте видалення всієї інформації вважається не доцільним оскільки його історія повинна бути збережена для перегляду інакшими користувачами, з якими він взаємодіяв. Також можливо впровадження часу дії сертифікатів ключів

Другий етап це розширення користувацького функціоналу мобільного застосунку. Перш за все імплементація історії взаємодії з документом, а саме статуси:

- Перегнуто
- Надіслано
- В обробці

Впровадження функціоналу для редагування документів, тобто мінімальний набір функцій який включає в себе:

- Модифікація шрифту

- Модифікація розмітки сторінки
- Робота зі списками
- Робота з посиланнями
- Коментарі та відгуки

Реалізація системи управління і генерації ключів для одного користувача в залежності від вказаних налаштувань під час створення користувача.

3.4 Впровадження системи

Для інтеграції реалізованої системи зі вже існуючими центрами сертифікації або запуск відокремлено від інакших систем, необхідно розгорнути середовище для серверного застосунку і бази даних.

Необхідні налаштування середовища:

- Node.js - v8.15.1
- React – v16.8.3
- React Native – v0.59.6
- PostgreSQL – v11.2
- npm - v6.4.1

Серверний застосунок запускається командами по порядку з кореневого каталогу:

1. npm install
2. npm audit fix
3. npm run start

Мобільний застосунок запускається командами по порядку з кореневого каталогу:

1. npm install

2. react-native update
3. react-native upgrade
4. react-native link
5. react native run-android/ios – в залежності від бажаної системи

Висновки до третього розділу

Реалізовано систему, яка позбавлена більшості недоліків аналогів. Насамперед, запропонована система не має необхідності передавати приватний ключ користувача через сторонні канали передачі даних. Генерація ключа відбувається безпосередньо в застосунку при реєстрації нового користувача у системі, що значно зменшує ризик перехоплення, втрати і модифікації приватного ключа користувача зловмисниками. Весь потік інформації між сервером та застосунком відбувається через https протокол з використанням SSL сертифікату. В майбутньому функціонал всієї системи можливо розширити відповідно до потреб користувачів та з відповідністю до вимог безпеки. Також, мобільний застосунку і сервер можливо інтегрувати зі вже існуючими центрами сертифікації.

ВИСНОВКИ

В ході роботи проаналізовано структури систем документообігу, центрів сертифікації та принципів їх взаємодії між собою і користувачем. Також проведено аналіз можливих загроз безпеці таких систем, та визначено доцільність розробки захищеної системи документообігу.

Був проведений аналіз захищеності існуючих систем документообігу на мобільній платформі, насамперед систем, що використовують платформу Android. Більшість програм, які розміщені у вільному доступі не можуть гарантувати верифікацію відправника і захищеність системи в цілому. Єдиний прототип системи, що позбавлений цього недоліку і використовує справжній електронний цифровий підпис на основі ключів, які видає АКЦС, є українською розробкою, але застосунок находиться в режимі тестування. Важливим недоліком даної системи є необхідність завантаження ключа сертифікату через сторонні канали передачі інформації.

Результатом роботи є створений прототип захищеної системи документообігу на мобільній платформі, яка позбавлена більшості недоліків аналогів. Насамперед, у запропонованій системі відсутня необхідності передавати приватний ключ користувача через сторонні канали передачі даних. Генерація ключа відбувається в застосунку при реєстрації нового користувача на його персональному пристрої, що значно зменшує ризики втрат приватного ключа. Весь потік інформації в системі відбувається за допомогою https протоколу з використанням SSL сертифікату. В майбутньому функціонал системи можливо розширити відповідно до потреб користувачів та вимог безпеки.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. PandaDoc [Електронний ресурс] – Режим доступу до ресурсу: https://www.pandadoc.com/.
2. DocuSign [Електронний ресурс] – Режим доступу до ресурсу: https://www.docusign.com/.
3. HelloSign [Електронний ресурс] – Режим доступу до ресурсу: https://www.hellosign.com/.
4. Про затвердження Концепції створення Єдиної державної автоматизованої паспортної системи [Електронний ресурс]. – 1997. – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/40-97-%D0%BF/ed20060315/find?text=%C7%E0%F5%E8%F1%F2+%B3%ED%F4%EE%F0%EC%E0%F6%B3%BF.
5. Безпека інформаційно-комунікаційних систем [Текст] / М. В. Грайворонський, О. М. Новіков. – Київ: BHV, 2009. – 594 с.
6. Безпека інформаційно-комунікаційних систем [Текст] / М. В. Грайворонський, О. М. Новіков. – С. 21
7. Класифікація та аналіз загроз інформаційній безпеці в ключових системах інформаційної інфраструктури [Електронний ресурс] – Режим доступу до ресурсу: http://pnzzi.kpi.ua/29/29_p56.pdf.
8. Досмухамедов Б.Р. Анализ угроз информации систем электронного документооборота [Текст] // Компьютерное обеспечение и вычислительная техника. – 2009. – № 6. – С. 140–143.
9. Сабанов А.А. Некоторые аспекты защиты электронного документооборота [Текст] // Connect! Мир связи. – 2010. – № 7. – С. 62–64.
10. Джхунян, В.Л. Электронная идентификация [Текст] / В.Л. Джхунян, В.Ф. Шаньгин. – М.:NT Press, 2004. – 695 с.

11. Кухарев Г. А. Биометрические системы: методы и средства идентификации личности человека. [Текст] – СПб.: Политехника, 2001. – 240 с.
12. Шрамко В.Н. Комбинированные системы идентификации и аутентификации [Текст]// PCWeek/RE. - 2004. - №45.
13. Електронний реєстр чинних, блокованих та скасованих сертифікатів відкритих ключів [Електронний ресурс] – Режим доступу до ресурсу: https://www.czo.gov.ua/ca-registry.
14. Про центральний засвідчувальний орган [Електронний ресурс] – Режим доступу до ресурсу: https://www.czo.gov.ua/.
15. Отримання електронних довірчих послуг [Електронний ресурс] – Режим доступу до ресурсу: https://acskidd.gov.ua/r_kor.
16. UKey [Електронний ресурс] – Режим доступу до ресурсу: https://www.ukey.ua/.
17. React Native Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://facebook.github.io/react-native/.
18. Node.js Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://nodejs.org/dist/latest-v10.x/docs/api/.

ДОДАТКИ

1. Файл server.js

```
import express from 'express';
import bodyParser from 'body-parser';
import dbQueries from './db/queries'
import serverConfig from './config';
import bcrypt from 'bcryptjs';
import AuthController from './auth-service/authController';

const server = express();

server.use(bodyParser.json({limit: '100mb'}));
server.use(
  bodyParser.urlencoded({
    extended: true,
    limit: '100mb'
  })
);

server.post('/api/v1/login', async (req, res) => {
  try {
    const result = await AuthController.login(req);
    return res.status(200).send(result);
  } catch (error) {
    return res.status(500).send(error);
  }
});

server.get('/api/v1/users/:id', AuthController.verifyToken, async (req, res) => {
  try {
    await dbQueries.getUserById(req.params.id).then(result => {
      if (result) {
        return res.status(200).json(result)
      }
    });
  } catch (error) {
    return res.status(500).send(error)
  }
});

server.put('/api/v1/users/:id/changePassword', AuthController.verifyToken, async (req, res) => {
  try {
    await dbQueries.changePassword(req.body, req.params.id).then(result => {
      return res.status(200).json(result)
    });
  } catch (error) {
    return res.status(500).send(error)
  }
});

server.put('/api/v1/users/:id', AuthController.verifyToken, async (req, res) => {
  try {
    await dbQueries.updateUser(req.body, req.params.id).then(result => {
      return res.status(200).json(result)
    });
  } catch (error) {
```

```
    return res.status(500).send(error)
  }
});

server.get('/api/v1/users/:id/receivers', AuthController.verifyToken, async (req, res) => {
  try {
    const currentId = req.params.id;

    await dbQueries.getUsers(currentId).then(result => {
      return res.status(200).json(result)
    });
  } catch (error) {
    return res.status(500).send(error)
  }
});

server.post('/api/v1/users', async (req, res) => {
  try {
    const hashedPassword = bcrypt.hashSync(req.body.password, 8);

    await dbQueries.createUser(Object.assign(req.body, {password: hashedPassword})).then(result => {
      return res.status(200).json(result)
    });
  } catch (error) {
    return res.status(500).send(error)
  }
});

server.delete('/api/v1/users/:id', AuthController.verifyToken, async (req, res) => {
  try {
    await dbQueries.deleteUser(req.params.id).then(result => {
      return res.status(200).json(result)
    });
  } catch (error) {
    return res.status(500).send(error)
  }
});

server.post('/api/v1/transactions', AuthController.verifyToken, async (req, res) => {
  try {
    await dbQueries.createTransaction(req.body).then(result => {
      return res.status(200).json(result)
    });
  } catch (error) {
    return res.status(500).send(error)
  }
});

server.get('/api/v1/transactions/:type/:id', AuthController.verifyToken, async (req, res) => {
  try {
    const currentId = req.params.id;
    const type = req.params.type || 'all';
    await dbQueries.getTransactions(currentId, type).then(result => {
      return res.status(200).json(result)
    });
  } catch (error) {
    return res.status(500).send(error)
  }
});
```

```
const PORT = process.env.PORT || serverConfig.port;

server.listen(PORT, () => {
  console.log(`server running on port ${PORT}`)
});
```

2. Файл queries.js

```
const Pool = require('pg').Pool;
import {CryptoController} from '../crypto-service/controllers/cryptoController';
import fs from 'fs-extra'
import bcrypt from 'bcryptjs';

const crypto = new CryptoController();

// const dbConfig = new Pool({
//   user: 'postgres',
//   host: 'localhost',
//   database: 'signer_db',
//   password: 'Crashday',
//   port: 5432,
// });

const dbConfig = new Pool({
  user: 'uvqgooiedextxu',
  host: 'ec2-23-23-241-119.compute-1.amazonaws.com',
  database: 'd2moojmcls4l84',
  password: '69f9f317965a67206b355b1a0de5659358e723c0566147cedd8286c96b1a2c91',
  port: 5432,
});

const getUsers = async (currentId) => {
  try {
    const result = await dbConfig.query(`SELECT id, login, name, surname, email, avatar_img AS "avatarImg",
      public_pem AS "publicPem" FROM users WHERE id != $1 ORDER BY id ASC`, [currentId]);
    return result.rows;
  } catch (error) {
    console.log('Error: ', error);
    return error;
  }
};

const getUserById = async (id) => {
  try {
    const result = await dbConfig.query('SELECT * FROM users WHERE id = $1', [id]);
    return result.rows[0]
  } catch (error) {
    console.log('Error: ', error);
    throw error;
  }
};

const createUser = async (body) => {
  try {
    const {name, surname, login, password, email, publicPem} = body;
    const getUsers = await dbConfig.query('SELECT * FROM users WHERE login = $1 LIMIT 1', [login]);
```

```
    if (getUsers.rows && getUsers.rows.length > 0) {
      throw {success: false, message: 'Login already exists'};
    }
    const avatar = toBase64('app/assets/defaultAvatar.jpg');
    const result = await dbConfig.query(`INSERT INTO users (name, surname, login, password, email,
public_pem, avatar_img)
      VALUES ($1, $2, $3, $4, $5, $6, $7) RETURNING id`, [name, surname, login, password, email, publicPem,
avatar]);
    return {success: true, message: 'User was successfully created'};
  } catch (error) {
    console.log('Error: ', error);
    throw error;
  }
};

function toBase64(file) {
  const bitmap = fs.readFileSync(file);
  return new Buffer(bitmap).toString('base64');
}


const createTransaction = async (body) => {
  try {
    const {sender, receiver, data, signHex, fileName} = body;
    const createDate = new Date();
    const dbSender = await getUserById(sender);
    const isVerified = await crypto.verifySign(signHex, data, dbSender.public_pem);
    const result = await dbConfig.query(`INSERT INTO transactions (sender, receiver, data, create_date,
is_verified, file_name)
      VALUES ($1, $2, $3, $4, $5, $6) RETURNING id`, [sender, receiver, data, createDate, isVerified, fileName]);
    return {success: true, message: 'Transaction was successfully created'};
  } catch (error) {
    console.log('Error:', error);
    throw error;
  }
};

const getTransactions = async (id, type) => {
  try {
    let result = null;
    switch (type) {
      case 'all': {
        result = await dbConfig.query(`SELECT u1.avatar_img AS "senderAvatarImg", u1.login AS "senderLogin",
        u2.avatar_img AS "receiverAvatarImg", u2.login AS "receiverLogin", data, create_date AS "createDate",
        file_name AS "fileName", is_verified AS "isVerified" FROM transactions
        LEFT JOIN users AS u1 ON transactions.sender = u1.id
        LEFT JOIN users AS u2 ON transactions.receiver = u2.id
        WHERE receiver = $1 OR sender = $1 ORDER BY create_date DESC;`, [id]);
        break;
      }
      case 'received': {
        result = await dbConfig.query(`SELECT u1.avatar_img AS "senderAvatarImg", u1.login AS "senderLogin",
        u2.avatar_img AS "receiverAvatarImg", u2.login AS "receiverLogin", data, create_date AS "createDate",
        file_name AS "fileName", is_verified AS "isVerified" FROM transactions
        LEFT JOIN users AS u1 ON transactions.sender = u1.id
        LEFT JOIN users AS u2 ON transactions.receiver = u2.id
        WHERE receiver = $1ORDER BY create_date DESC;`, [id]);
        break;
      }
```

```
      case 'sent': {
        result = await dbConfig.query(`SELECT u1.avatar_img AS "senderAvatarImg", u1.login AS "senderLogin",
        u2.avatar_img AS "receiverAvatarImg", u2.login AS "receiverLogin", data, create_date AS "createDate",
        file_name AS "fileName", is_verified AS "isVerified" FROM transactions
        LEFT JOIN users AS u1 ON transactions.sender = u1.id
        LEFT JOIN users AS u2 ON transactions.receiver = u2.id
        WHERE sender = $1 ORDER BY create_date DESC;`, [id]);
        break;
        break;
      }
    }
    return result.rows;
  } catch (error) {
    console.log('Error: ', error);
    throw error;
  }
};

const updateUser = async (body, id) => {
  try {
    const {name, surname, login, email} = body;
    const result = await dbConfig.query(
      'UPDATE users SET name = $1, surname = $2, login = $3, email = $4 WHERE id = $5',
      [name, surname, login, email, id]);
    return {success: true, message: 'User was successfully updated'};
  } catch (error) {
    console.log('Error: ', error);
    throw error;
  }
};

const changePassword = async (body, id) => {
  try {
    const {password} = body;
    console.log('Body: ', body);
    console.log('id: ', id);
    const newPassword = bcrypt.hashSync(password, 8);
    const result = await dbConfig.query(
      'UPDATE users SET password = $1 WHERE id = $2',
      [newPassword, id]);
    return {success: true, message: 'Password was successfully updated'};
  } catch (error) {
    console.log('Error: ', error);
    throw error;
  }
};

const deleteUser = async (id) => {
  try {
    const result = await dbConfig.query('DELETE FROM users WHERE id = $1', [id]);
    return {success: true, message: 'User was successfully deleted'};
  } catch (error) {
    console.log('Error: ', error);
    throw error;
  }
};

const getUserByLogin = async (login) => {
  try {
```

```
  const result = await dbConfig.query('SELECT * FROM users WHERE login = $1 LIMIT 1', [login]);
  if (result.rows && result.rows.length > 0) {
   return result.rows[0];
  } else {
   throw {success: false, message: 'Wrong password or login'}
  }
 } catch (error) {
  console.log('Error: ', error);
  throw error;
 }
};

module.exports = {
 getUsers,
 getUserById,
 createUser,
 getTransactions,
 updateUser,
 deleteUser,
 getUserByLogin,
 createTransaction,
 changePassword
};
```

3. Файл cryptoController.js

```
import rs from 'jsrsasign';

export class CryptoController {

 test() {
  const sign = new rs.Signature({alg: 'SHA512withRSA'});
  const keyPair = new rs.KEYUTIL.generateKeypair('RSA', 2048);
  sign.init(keyPair.prvKeyObj);
  sign.updateString('testString');
  const signHex = sign.sign();

  const newSign = new rs.Signature({alg: 'SHA512withRSA'});
  newSign.init(keyPair.pubKeyObj);
  newSign.updateString('testString');
  const isValid = newSign.verify(signHex);

  return {encoded: signHex, verify: isValid, keyPair: keyPair};
 }

 async verifySign(signHex, data, publicPem) {
  try {
   console.log('signHex: ', signHex);
   console.log('data: ', data);
   console.log('publicPem: ', publicPem);

   const newSign = new rs.Signature({alg: 'SHA512withRSA'});
   const publicKey = rs.KEYUTIL.getKey(publicPem);
   newSign.init(publicKey);
   newSign.updateString(data);
   const isValid = newSign.verify(signHex);
   return isValid;
  } catch (error) {
```

```
    console.log('Error =====> ', error);
    throw(error);
   }
  }
}
```

4. Файл authController.js

```
import jwt from 'jsonwebtoken';
import bcrypt from 'bcryptjs';
import queries from '../db/queries';
import config from '../config';

const login = async (req) => {
 try {
  const user = await queries.getUserByLogin(req.body.login);
  const isPasswordValid = bcrypt.compareSync(req.body.password, user.password);
  if (isPasswordValid) {
   const token = jwt.sign({id: user.id}, config.serverSecret, {
    expiresIn: 86400
   });
   return {success: true, auth: true, user: {id: user.id, login: user.login, token: token}};
  } else {
   throw {success: false, auth: false, message: 'Wrong password or login'};
  }
 } catch (error) {
  return error;
 }
};

const verifyToken = async (req, res, next) => {
 try {
  const token = req.headers['x-access-token'];
  if (!token) {
   throw res.status(403).send({success: false, auth: false, message: 'Failed to authenticate token.'});
  } else {
   await jwt.verify(token, config.serverSecret, async (err, decoded) => {
    if (err) throw res.status(403).send({success: false, auth: false, message: 'Failed to authenticate token.'});
    next();
   });
  }
 } catch (error) {
  throw error;
 }
};

module.exports = {
 login,
 verifyToken
};
```