

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Острозька академія»
Економічний факультет
Кафедра економіко-математичного моделювання та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА/ПРОЄКТ
на здобуття освітнього ступеня бакалавра

на тему: **«РОЗРОБКА МОДУЛЮ СИСТЕМИ UNIVERSITY MANAGEMENT SYSTEM: ЕЛЕКТРОННИЙ ДОКУМЕНТООБІГ»**

Виконав: студент 4 курсу, групи КН-41
першого (бакалаврського) рівня вищої освіти
спеціальності 122 Комп'ютерні науки
освітньо-професійної програми «Комп'ютерні науки»
Неручок Богдан Русланович

Керівник: кандидат психологічних наук, доцент
кафедри економіко-математичного моделювання та
інформаційних технологій,
Зубенко Ігор Ростиславович

Рецензент:
Красюк Богдан Віталійович

РОБОТА ДОПУЩЕНА ДО ЗАХИСТУ

Завідувач кафедри економіко-математичного моделювання та інформаційних
технологій _____ (проф., д.е.н. Кривицька О.Р.)
Протокол № ____ від « ____ » _____ 2022 р.

Острог, 2022

Міністерство освіти і науки України
Національний університет «Острозька академія»

Факультет: економічний

Кафедра: економіко-математичного моделювання та інформаційних технологій

Спеціальність: 122 Комп'ютерні науки

Освітньо-професійна програма: Комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри економіко-математичного моделювання
та інформаційних технологій

_____ Ольга КРИВИЦЬКА
« ____ » _____ 20__ р.

З А В Д А Н Н Я
на кваліфікаційну роботу/проект студента

Неручка Богдана Руслановича

1. *Тема роботи* Розробка модулю системи University management system електронний документообіг

керівник проекту Зубенко Ігор Ростиславович, кандидат психологічних наук, доцент кафедри економіко-математичного моделювання та інформаційних технологій.

Затверджено наказом ректора НаУОА від 29 жовтня 2021 року №110.

2. *Термін здачі студентом закінченої роботи/проекту:* 03 червня 2022 року.

3. *Вихідні дані до роботи/проекту:* данна робота полягає у розробці модуль «UM System електронний документообіг»

4. *Перелік завдань, які належить виконати:* реалізувати зчитування особистого ключа, реалізувати можливість обрання параметрів цифрового підпису, реалізувати накладання електронно-цифрового підпису на файл, реалізувати перевірку файлу на наявність цифрового підпису.

5. *Перелік графічного матеріалу:* рисунки, таблиці.

6. *Консультанти розділів роботи:*

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Зубенко І. Р.	01.12.2021р.	01.12.2021р.
2	Зубенко І. Р.	01.12.2021р.	01.12.2021р.
3	Зубенко І. Р.	01.12.2021р.	01.12.2021р.

7. Дата видачі завдання: 01.12.2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів	Примітка
1.	Затвердження теми роботи/проєкту	до 01.11.2021 р.	
2.	Постановка технічного завдання	до 01.12.2021 р.	
3.	Ознайомлення з документацією АТ "ІТ"	до 10.12.2021 р.	
4.	Розробка архітектури проєкту, погодження функціоналу	до 24.12.2021 р.	
5.	Розробка інтерфейсу системи	до 01.03.2022 р.	
6.	Розробка накладання ЕЦП	до 01.04.2022 р.	
7.	Розробка перевірки ЕЦП	до 01.05.2022 р.	
8.	Тестування системи	до 20.05.2022 р.	
9.	Попередній захист кваліфікаційної роботи/проєкту	до 01.06.2022 р.	
10.	Здача кваліфікаційної роботи/проєкту на кафедрі	03.06.2022 р.	

Студент: _____ Богдан НЕРУЧОК

Керівник кваліфікаційної роботи: _____ Ігор ЗУБЕНКО

АНОТАЦІЯ
кваліфікаційної роботи/проєкту
на здобуття освітнього ступеня бакалавра

Тема: Розробка модулю системи University Management System: електронний документообіг

Автор: Неручок Богдан Русланович

Науковий керівник: Зубенко Ігор Ростиславович, кандидат психологічних наук, доцент кафедри економіко-математичного моделювання та інформаційних технологій.

Захищена «.....»..... 20__ року.

Пояснювальна записка до кваліфікаційної роботи: 53 с., 31 рис., 2 табл., 12 джерел.

Ключові слова: бібліотека, функція, електронний цифровий підпис, ЕЦП, сертифікат, ключ, система.

Короткий зміст праці:

Завданням кваліфікаційної роботи/проєкту, було розробка модулю електронний документообіг до системи «University management system», у майбутньому для впровадження в саму систему.

Зазначений проєкт являє собою реалізацію системи управління документами в межах Національного університету «Острозька академія». З можливістю авторизуватися за допомогою власного електронно-цифрового підпису, вибір параметрів електронно цифрового підпису, накладання зовнішнього та внутрішнього електронно-цифрового підпису на документ, перевірка документу на наявність зовнішнього та внутрішнього підпису.

The task of the qualification project was to develop a module of electronic document management system “University management system”, in the future for implementation in the system itself. This project is the implementation of a document management system within the National university of “Ostroh Academy”. With the ability to log in with your own electronic digital signature, select digital signature parameters, impose external and internal electronic digital signature on the document, check the document for external and internal signatures.

ЗМІСТ

ВСТУП	6
РОЗДІЛ 1. ЗАГАЛЬНІ ПОЛОЖЕННЯ	8
1.1. Опис предметного середовища	8
1.2. Огляд наявних аналогів	12
1.3. Постановка задачі	16
Висновки до розділу 1	17
РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ	18
2.1. Аналіз предметної області	18
2.2. Проєктування системи	20
2.3. Криптографічне та алгоритмічне забезпечення	25
Висновки до розділу 2	33
Розділ 3. Програмне та технічне забезпечення	35
3.1. Засоби розробки	35
3.2. Вимоги до технічного та програмного забезпечення	36
3.3. Опис програмної реалізації	36
3.4. Керівництво користувача	40
Висновки до розділу 3	48
ВИСНОВКИ	49
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	51

ВСТУП

Події сьогодення прискорюють темпи цифровізації та автоматизації, постійне зростання кількості інформації призводить до того, що використовувати традиційні методи роботи стає все більш нерентабельно, що сприяє переходу на віддалений формат роботи та переведення багатьох процесів з офлайн в онлайн режим. Відповідним чином, перед нами ставиться завдання автоматизації процесу. Для покращення зручності та швидкості роботи університету на базі Національного університету «Острозька академія», було прийняте рішення створити загально-централізовану систему «UM System». UM System - це система, яка виконує велику кількість задач.

Наразі університет не має своєї системи електронного документообігу, тому викладачам, студентам та працівникам доводиться використовувати сторонні сервіси для обігу документів, що може призводити до певних незручностей та приватності даних. Тому було поставлено мету – створити модуль “Електронний документообіг” для «UM System», як рішення проблем накопичення та розмноження фізичних документів, проблеми відстеження та контролю порядку передачі конфіденційної інформації, що знизить трудовитрати. Впровадження цієї системи сприятиме контролю роботи з документацією, підвищенню якості роботи працівників та більшій прогнозованості термінів підготовки документів.

Мета кваліфікаційної роботи/проєкту – реалізація модуля системи «University management system» електронний документообіг.

Нами були вирішені перелік таких завдань:

1. Опис предметного середовища (технології, які використовуються);
2. Огляд наявних аналогів;
3. Розробка технічного завдання;
4. Реалізація функціоналу системи;
5. Розробка інтерфейсу користувача.

Об'єктом дослідження даної кваліфікаційної роботи/проекту є система «University Management System електронний документообіг».

Предметом дослідження є технології: html, css, bootstrap, javascript, IT library

РОЗДІЛ 1

ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1. Опис предметного середовища

В сьогоднішні нам на вибір представлено велику кількість мов програмування, технологій та бібліотек для реалізації проєктів. Переваги кожного з цих інструментів можуть проявлятися лише в контексті поставленої задачі перед програмістом, та його здобутих знань та навичок роботи з певною технологією, що є достатнім для реалізації проєкту.

Потрібно розуміти, що програмування – це процес створення програми, а програма в свою чергу – це алгоритми в поєднанні з структурою даних. Тобто мова програмування – це інструмент (команд, синтаксису та семантичних правил) за допомогою якого програміст спілкується з комп'ютером для реалізації певних задач.

Перш ніж визначитись з переліком технологій та інструментів для розробки нашої системи, потрібно мати попереднє бачення, що вона буде собою являти, та які задачі виконувати, що описано в побудованому технічному завданні (розділ 1.3).

Для реалізації системи «University Management System електронний документообіг» потрібно обрати відповідний стек технологій, які будуть використовуватись при розробці. Проєкт «University Management System електронний документообіг» повинен бути реалізований таким чином, щоб в подальшому з легкістю міг бути впроваджений в систему «UM System», тому для розробки було використано: HTML, CSS, JavaScript, Bootstrap, PHP, ІТ бібліотека ЦСК, FileSaver.js.

Система «University management system електронний документообіг» буде представлена як односторінковий додаток (SPA).

SPA додаток – це протилежність PWA (традиційний багатосторінковий веб-ресурс), він представлений як web-застосунок, який компонується з одного HTML-документу, що дозволяє при переході між вкладками підвантажувати тільки

запитуваний контент, а статичні блоки (шапки сайту та бічна панель) не перезавантажуються. З такою архітектурою контент сторінки завантажується швидше, а web-застосунок може здатись користувачеві, як повноцінна програма.

Тобто SPA – дослівно «односторінковий додаток», який динамічно підгружається на одній HTML-сторінці за допомогою AJAX-запитів на сервер. Цей підхід дозволяє нам звертатись до сервера, залишаючись на одній сторінці. Сервер у відповідь може відправляти практично будь-який тип даних в форматі JSON. На наступному рисунку зображено два підходи: перший традиційний, а другий це той, що дозволяє оновлювати зміст сторінки без перезавантаження (Рис. 1.1.).

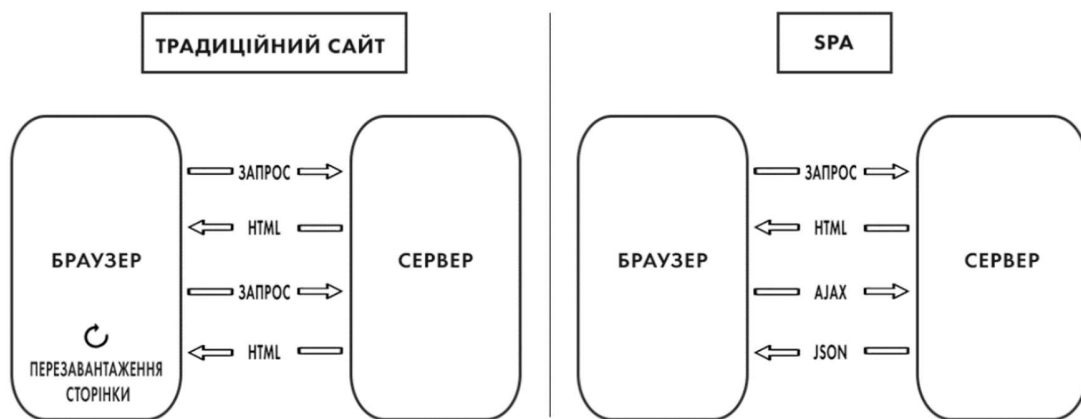


Рис. 1.1. Різниця традиційного сайту та SPA

Однак невірно думати те, що веб-застосунок (PWA) – це і є односторінковий додаток (SPA). SPA може бути PWA, але PWA не обов’язково повинен бути SPA. Для розуміння розглянемо принцип роботи двох цих процесів:

1) Принцип роботи динамічного відображення полягає в тому, що під час роботи динамічного сайту сервер розпізнає пошукових роботів, а ті в свою чергу передають запит засобу відображення. Запити від користувачів обробляються так само як і від традиційних сайтів;

2) Принцип роботи PWA-сайтів описано наступним чином, що при першому відвідуванні користувачем PWA-сайту застосовується скрипт «Service Worker», що додає

оболонку в кеш. Після завантаження оболонки програма запитує вміст для заповнення представлення, а потім робить запит на контент. І після того, коли всі запити вже було завершено, «Service Worker» переходить в стан очікування та перебуває в цьому стані до тих пір, доки мережевим запитом не буде ініційовано нової події.

На наступній таблиці наведена різниця між SPA та PWA (Таблиця 1.1.).

Таблиця 1.1.

Різниця між SPA та PWA

	SPA	PWA
Швидкодія	Збільшення швидкості після першого завантаження	Швидкість нижча, так як кожна сторінка завантажується наново
Безпека	Має вразливість до XSS-атак без заходів захисту	Менш вразливий, але захисту потребує кожна сторінка
Простір застосування	Додатки, в яких застосовується безперервний UX / td & amp; amp;	Складна інформаційна архітектура з категоріями та рівнями
Приклади	Gmail, Facebook	Rozetka, OLX

З усього вище описаного, ми можемо спостерігати те, що обидва підходи мають свої переваги, недоліки та сфери застосування.

На сьогоднішній день найпопулярнішою мовою програмування серед українських розробників залишається JavaScript (18,8%). JavaScript відноситься до клієнтської мови програмування для управління сценаріями веб-сторінки. Саме вона і буде становити основну частину реалізованого функціоналу системи «University management system електронний документообіг». Так як її особливістю є те, що використовуючи цю мову, ми можемо приходити до результату, коли зображені елементи можуть змінюватись під час перебування на самій сторінці без її перезавантаження. Наприклад, відображення різних повідомлень, зміни стану певних елементів, завантаження файлів і т.д.

На наступному графіку продемонстровано найбільш широко застосовані мови програмування в розробці веб-проектів (Рис. 1.2.).

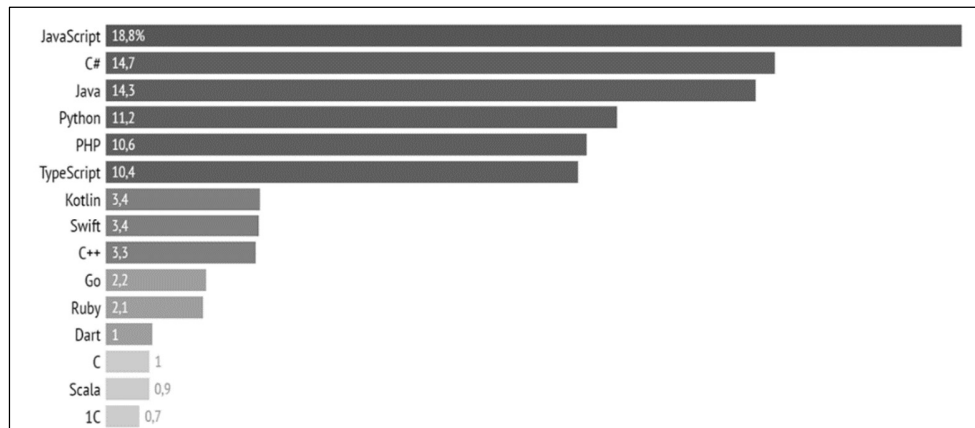


Рис. 1.2. Найбільш широко застосовані мови програмування в розробці веб-проектів

Спільно з JavaScript будуть використанні такі мови розмітки, як HTML та CSS, вони є не від'ємною складовою будь-якого веб-застосунку. HTML – це мова гіпертексту розмітки документів для перегляду веб-сторінок в браузері, CSS розшифровується як каскадна таблиця стилів, при її використанні, ми зможемо описати зовнішній вигляд веб-сторінки.

Для надання системі більш приємного вигляду, було прийняте рішення використовувати Bootstrap фреймворк, який надає нам безкоштовний набір інструментів такі як іконки, кнопки, поля навігацій, випадаючі списки та інші шаблони. Також основною перевагою залишається інструмент, який дозволяє нашому застосунку стати адаптивним під різні розміри екранів, починаючи з великих телевізорів і закінчуючи невеличким телефоном.

FileSaver.sj – це ідеальне рішення для збереження файлів на стороні клієнта та добре підходить для нашого веб-застосунку, який буде генерувати файли на стороні клієнта. Однак у майбутньому впровадженні системи в «UM System» і можливій реалізації надходження файлів з сервера можна звернути увагу на альтернативу Content-Disposition, так як він має більш міжбраузерну сумісність. FileSaver.sj є безкоштовною бібліотекою та знаходиться у відкритому доступі.

Бібліотека користувача ЦСК – це програмний комплекс, призначений для виконання функції, які пов'язані із перенаправленням запитів серверів ЦСК. Бібліотека функціонує з використанням інтепретатора PHP 5.3.x та вище. Бібліотека має великий функціонал, такі як управління ключами користувача (генерацію ключів, перевірку сформованого сертифікату, зміна паролю захисту особистого ключа та інше), доступ до сертифікатів користувача ЦСК, серверів ЦСК та захист файлів користувача (підпис файлів та перевірку файлів). На останні пункти буде найбільше зосереджено нашу увагу при розробці системи.

Для функціонування бібліотеки ЦСК було прийнято рішення використовувати PHP – це серверна мова програмування, а точніше скриптова мова сценаріїв. PHP потрібен нам для взаємодій з серверами ЦСК, так як через політику безпеки веб-браузерів, взаємодія з серверами ЦСК можлива лише за умови реалізації механізму кросдоменної взаємодії. Для реалізації взаємодії між доменами, сервер, на якому розміщено бібліотеку має реалізовувати механізм проху-сервісу, щоб була змога перенаправлення запитів серверів ЦСК.

1.2. Огляд наявних аналогів

Для реалізації системи, було зроблено попередній огляд та аналіз вже існуючих відомих інформаційних систем, що забезпечують процес створення, управління доступом і поширення електронних документів, їх підписання та перевірку вже підписаних документів, мова йде про такі системи, як Signy та DOCUMENT.online.

Це автоматизовані, багатокористувацькі системи, що дають змогу організувати спільну роботу користувачам з різними електронними документами.

Signy – це український сервіс електронного документообігу, що був реалізований компанією SmartTender – який є одним з найбільших торговельних майданчиків для проведення державних тендерів та комерційних торгів на території України.

Сервіс Signy має зручний та інтуїтивно зрозумілий інтерфейс, чимось подібний до сервісів Google, таких як email. Працюючи в Signy користувачу не доведеться встановлювати якесь додаткове програмне забезпечення на свій комп'ютер. Для того, щоб розпочати користуватись системою, користувач повинен лише пройти реєстрацію та мати пароль, або кваліфікований цифровий підпис для входу в особистий кабінет.

Сервіс надає користувачам змогу користуватись такими діями як: обмінюватись документами будь-якого формату зі іншими користувачами для підписання (документів, договорів, різних актів тощо) та фіксувати факт отримання та ознайомлення, що є набагато безпечніше та менш затратним по часу.

Крім таких функцій, як підписування, відправка та зберігання документів, сервіс також підтримує такі функції:

- Створення документів за допомогою налаштованих індивідуальних шаблонів;
- Формування власного маршруту документу, етапів підписання та узгодження;
- Змога редагувати та вносити коментарі до документів;
- Створення груп, де користувачі мають змогу спільно-внутрішньо обговорювати документ;
- Можливість інтегрування з іншими системами обліку;
- Розумний пошук документів та цілодобовий доступ до них.

Сервіс DOCUMENT.online в своїй основі має велику кількість того ж функціоналу, що нам надає Signy, таких як: перегляд завантажених документів будь-якого формату, шаблони для створення первинних документів, перегляд завантажених підписаних документів на інформацію про власника підпису, відправка, налаштування маршруту та інше.

Розглянемо детальніше інформацію про розглянуті нами сайти та проведемо загальний аналіз для визначення переваг та недоліків даних продуктів за певним рядом критеріїв у наступній таблиці (Таблиця 1.2).

Таблиця 1.2

Аналіз аналогів електронного документообігу

Критерії	Signy	DOCUMENT.online	«UM System електронний документообіг»
Відсутність реклами	+	+	+
Зрозумілий інтерфейс	+	-	+
Можливість для підписання	+	+	+
Рівень безпеки системи	+	+	+
Цінова політика	-	-	+
Мобільна версія	+	+	+
Зручність використання	+	+	+

Отже, проаналізувавши інформацію з порівняння ресурсів з схожою тематикою було прийнято рішення розробити систему з простим, інтуїтивно зрозумілим інтерфейсом та здатністю додавання в майбутньому нових функцій. З забезпеченням достовірності, цілісності та актуальності узгоджуваної інформації. Зменшення часу та трудовитрат на підготовку та пошук необхідних документів.

Нарізі заклад Національний університет «Острозька академія» віддає перевагу в використанні системи Signy для підписання документів за рахунок швидкої реєстрації та зручного зрозумілого інтерфейсу. Наша система буде мати більш простий інтерфейс та зрозумілий функціонал.

1.3. Постановка задачі

Для більш чіткого розуміння як повинна виглядати та функціонувати система «University Management System електронний документообіг» було здійснено докладний опис задачі, яку потрібно виконати.

Тема кваліфікаційної роботи, вказує нам про масштабну систему з великим функціоналом, який був описаний в сформованому технічному завданні на основі дослідження існуючих аналогів на ринку, а саме таких як document.online та signy.

ТЗ нам необхідне як вихідна інформація до проекту. Тільки ознайомившись із проектом за допомогою ТЗ, можна робити висновки про реальний обсяг роботи, а також терміни виконання. Крім того, ТЗ використовується у процесі підготовки до роботи над проектом або під час виконання. Від повноти наданої інформації, опрацювання деталей часто-густо залежить і підсумковий результат.

Тому для того, щоб вчасно та якісно реалізувати проект, було зроблено узагальнення задачі, де вся увага була направлена на реалізацію пункту роботи з накладанням електронно-цифрового підпису на документ та перевіркою документів на наявність підпису.

1. Призначення системи

- 1.1. Система призначена для підвищення ефективності роботи за рахунок скорочення термінів та підвищення прозорості процесу погодження документів;
- 1.2. Забезпечення достовірності, цілісності та актуальності узгоджуваної інформації;
- 1.3. Зменшення часу та трудовитрат на підготовку та пошук необхідних документів;
- 1.4. Підвищення ефективності виконання завдань та документів;
- 1.5. Архівне зберігання електронних документів в межах закладу вищої освіти.

2. Реєстрація та авторизація у модулі

- 2.1. Авторизація здійснюється за допомогою:
 - 2.1.1. Обрання користувачем Центру сертифікації ключів, яким був наданий ЕЦП;
 - 2.1.2. Завантаження власного електронно-цифрового ключа;
 - 2.1.3. Введення коду захисту електронно-цифрового ключа;
 - 2.1.4. Кнопка «Зчитати», якщо користувач не авторизований;
 - 2.1.5. Кнопка «Стерти», якщо користувач авторизований;
 - 2.1.6. Зберігання даних користувача після авторизації.

3. Накладання параметрів ЕЦП та підписання файлу

- 3.1. Якщо користувач не авторизований, то поле «Обрати файл для підпису», кнопка «Підписати документ» не активні;
- 3.2. Користувач має змогу обрати параметри електронно-цифрового підпису:
 - 3.2.1. Використовувати внутрішній підпис;
 - 3.2.2. Додавати позначку часу від даних;
 - 3.2.3. Додавати повні дані для перевірки;
 - 3.2.4. Випадаючий список формату ЕЦП:
 - 3.2.4.1. Базовий;
 - 3.2.4.2. З позначкою часу від ЕЦП;
 - 3.2.4.3. З посиланням на певні дані для перевірки;
 - 3.2.4.4. З повними даними для перевірки;
 - 3.2.4.5. З повними даними ЦСК для перевірки.
 - 3.2.5. Поле для завантаження документу, на який буде накладено ЕЦП;
 - 3.2.6. Кнопка «Підписати документ» здійснює накладання підпису на документ;
 - 3.2.7. Автоматичне завантаження документу з накладеним ЕЦП після натискання на кнопку «Підписати документ».

4. Перевірка файлу на наявність ЕЦП

- 4.1. При використанні внутрішнього підпису (пункт 3.2.1), поле «Обрати файл з підписом» (пункт 4.3) не активне;
- 4.2. Поле «Оберіть файл для перевірки»;
- 4.3. Поле «Оберіть файл з підписом».

Висновки до розділу 1

У результаті опрацювання першого розділу, було зроблено опис предметного середовища, а саме здійснено перелік технологій, які були задіяні для реалізації системи «University Management System електронний документообіг», а саме: HTML, CSS,

JavaScript, Bootstrap, PHP, IT бібліотека ЦСК, FileSaver.js. Коротке представлення, що з себе представляє кожна з перерахованих технологій та аргументоване обґрунтування того, чому ми використовуємо дані засоби для розробки системи. Також в першому розділі було досліджено та проаналізовано існуючі аналоги, такі як: signy та document.online, було виконано спостереження переваг та недоліків даних систем. Та в заключенні нами було сформоване технічне завдання, де було описано перелік вимог та задач, які повинна мати та виконувати наша система.

РОЗДІЛ 2

ІНФОРМАЦІЙНЕ ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Аналіз предметної області

Сьогодні електронне підписання документів вже не є чимось фантастичним та незвичним, а досить реальною практикою в багатьох учбових закладах. Це надійно, зручно, крім того, значно економить час роботи бухгалтерів та інших працівників у роботі з документами і звітами.

За допомогою електронно-цифрового підпису, який представляє з себе набір символів, розміщених в певній послідовності, користувач проходить автентифікацію, яка підтверджує належність електронного підпису даній особі і в подальшому користувач може використовувати його для накладання на файли. Особистий ключ захищений кодом захисту, який потрібно вводити при накладанні ЕЦП. Код захисту особистого ключа відновити неможливо, тому при його втраті користувач змушений звернутись до одного з кваліфікованих постачальників електронних довірчих послуг для отримання нового ключа.

Для розуміння концепту електронно-цифрового підпису (ЕЦП) потрібно розглянути інші поняття, які тісно з ним переплітаються а саме:

- 1) Електронний ключ;
- 2) Сертифікат електронного підпису.

Удосконалений тип електронного підпису можна розділити на два види: кваліфікований та не кваліфікований. Удосконалений кваліфікований підпис, можна розпізнати за такими властивостями:

- 1) Удосконалений ключ перевірки вказаний в кваліфікованому сертифікаті;
- 2) Для формування та перевірки удосконаленого ЕЦП, повинен бути використаний засіб, який відповідає вимогам чинного законодавства.

Всі електронні документи, які підписуються ЕЦП є юридично значущими на законодавчому рівні. ЕЦП має таку ж саму юридичну силу, як і рукописний підпис.

Тепер можемо сформулювати більш чітко розуміння того, що ЕЦП – це дані в електронній формі, які були отримані в результаті криптографічного перетворення, які в подальшому докладаються до інших даних, файлів чи документів, що забезпечує неподільність та ідентифікацію автора. Саме так, ми можемо впевнитись в тому, що підпис справжній і те, що інформацію не було змінено чи замінено з часу накладання підпису.

Створення ЕЦП відбувається за допомогою алгоритмів криптографічного методу шифрування інформації. Формується цифровий код з ключами, які зашифровують та розшифровують інформацію. В основному використовується симетричне та асиметричне шифрування:

- Симетричне шифрування – це метод шифрування даних, при якому один і той же самий ключ використовується для кодування і декодування інформації (два однакових електронних ключі);
- Асиметричне шифрування – це метод шифрування, при якому задіюється дві пари різних ключів: відкритий та закритий. При шифруванні даних відкритим ключем, розшифрувати їм можна закритим, та навпаки.

В Україні та в нашій системі ЕЦП накладається особистим ключем, а перевірка здійснюється за допомогою відкритого. Працездатність особистого ключа можлива лише в парі з відкритим ключем, що знаходиться в сертифікаті відкритого ключа, який містить дані про власника та строк дії ключа. Сертифікат користувача розміщується та зберігається на офіційному інформаційному ресурсі.

На наступній схемі відображено два процеси підписання даних та верифікації підписаного документу (Рис. 2.1.).



Рис. 2.1. Процеси підписання даних та верифікації підписаного документу

Підписання документу реалізовано шляхом хешування даних, шифрування хеш-суми за допомогою приватного ключа та додавання до даних сертифікату відкритого ключа, що в результаті дає підписані дані.

Верифікація ж виконується шляхом хешування документу в його стартовому вигляді та дешифруванням хеш-суми за допомогою публічного ключа, який був доданий до даних. Після чого дві хеш-суми порівнюються для перевірки цілісності документу.

Отже, предметна область нашого курсового проєкту – це робота над файлами з метою накладання на них електронно-цифрового підпису, перевірка сертифікатів електронного підпису та зчитування інформації підписаних файлів.

2.2. Проєктування системи

Спершу представимо модель процесу роботи у нотації IDEF, де графічно описано систему і процеси діяльності на нульовому рівні. Весь процес представлено як функціональний блок, який містить усі відповідні робочі та керуючі об'єкти. Також на

діаграмі відображено необхідні нам дані та вхідну інформацію, яка буде використана для авторизації в системі, роботі накладання ЕЦП на документ та перевірку накладеного підпису. Тобто дана модель показує систему у вигляді комплексу функцій, які поєднані за допомогою стрілок, що зображені на взаємопідпорядкованих схемах. Діаграму нульового рівня відображено на рисунку (Рис. 2.2.).

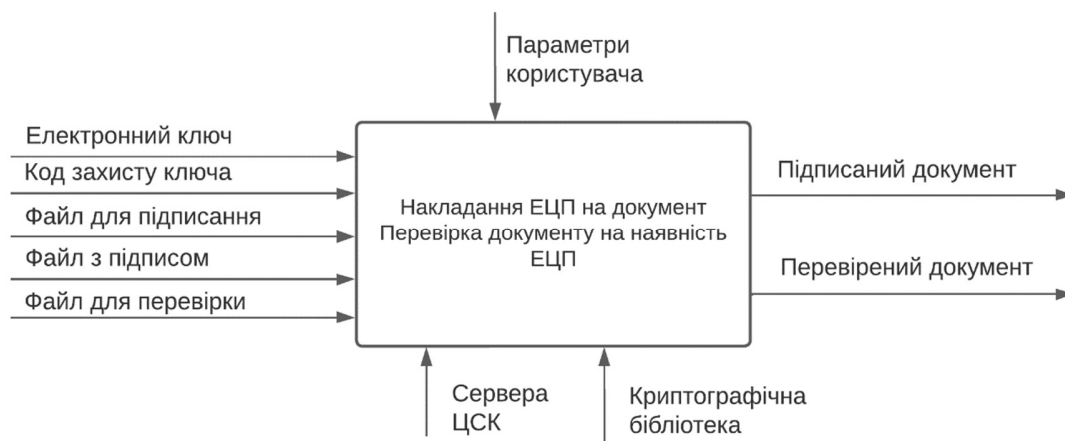


Рис. 2.2. Контекстна модель

На першій діаграмі рівнів було описано функцію обробки нульового рівня.

Вхідними даними в нас виступають:

- Електронний ключ користувача;
- Код захисту ключа, який користувач вводить при авторизації;
- Файл для підписання;
- Файл з підписом;
- Файл для перевірки на наявність підпису.

За керуючу частину відповідатимуть параметри користувача при перебуванні в системі.

Механізмами виконання функції являються:

- Сервера ЦСК (для отримання інформації про стан сертифікатів користувача);

- Криптографічна бібліотека (для перевірки та накладання ЕЦП).

Вихідні дані, які демонструють результат роботи функції є:

- Підписаний документ;
- Інформація про перевірений документ на наявність ЕЦП.

Розкладаємо функціональний блок 0 на набір, взаємопов'язаний між собою під функції. Представлення даної моделі декомпозиції продемонстровано на рисунку (Рис.2.3.).

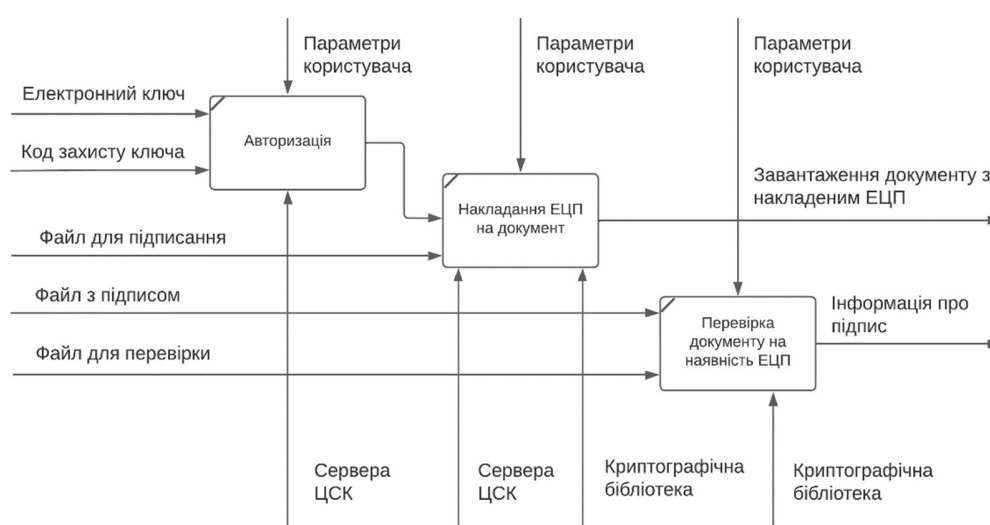


Рис. 2.3. Модель декомпозиції

У моделі декомпозиції, ми зобразили процес на основі попередньої моделі. У цій моделі, ми виділили об'єкти та основні вузли системи, які більш чітко демонструють нам послідовність роботи системи, що веде до вихідних даних. Ця декомпозиція виконується в декілька етапів, де на першому етапі у відповідності до цілей визначаються основні завдання системи.

Діаграма послідовності – це діаграма з демонстрацією взаємодії об'єктів впорядкованих за часом, задіяні об'єкти і черговість надсилання повідомлень. Дана діаграма є одним із способів формалізації сценарію використання об'єктів. Її особливість в тому, що ми можемо ще на ранніх стадіях опису сценаріїв, з'ясувати

складові взаємодіючих компонентів і зобразити потік повідомлень між компонентами. В наведеній нижче діаграмі демонструється послідовність авторизації користувача в системі за допомогою свого ЕЦП, коду захисту та обраного ЦСК (Рис.2.4.).

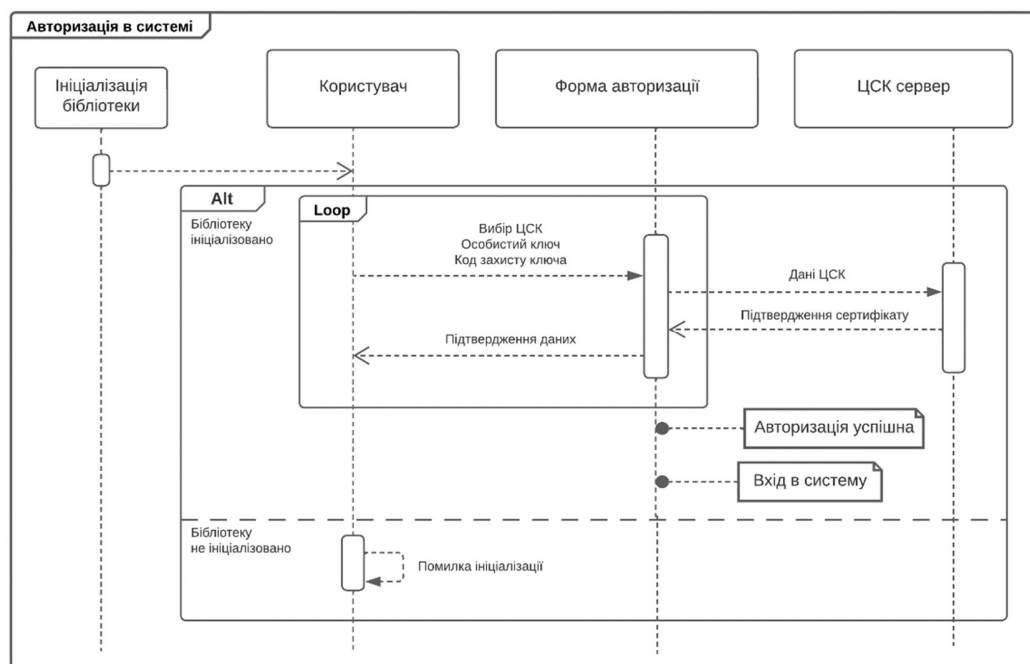


Рис. 2.4. Діаграма послідовності авторизації в системі

В наведеній вище діаграмі першим етапом виконується ініціалізація криптографічної бібліотеки. При успішній ініціалізації, користувач має змогу передати свої дані для входу в систему, а саме свій особистий приватний ключ, код захисту до цього ключа та обрати відповідний центр сертифікації ключів, яким був виданий. Після чого дані обробляються і користувачу надходить відповідь про успішність авторизації або код помилки.

Після успішної авторизації в користувача з'являється можливість накладання свого персонального ЕЦП на документ, яка до входу в систему є недоступною. Не зважаючи на те, авторизований користувач в системі, чи ні в нього є можливість взаємодіяти з параметрами для налаштування формату ЕЦП. Увійшовши в систему, користувач має змогу завантажити файл для накладання електронно-цифрового підпису. Цей процес послідовності зображено на наступній діаграмі (Рис. 2.5.).

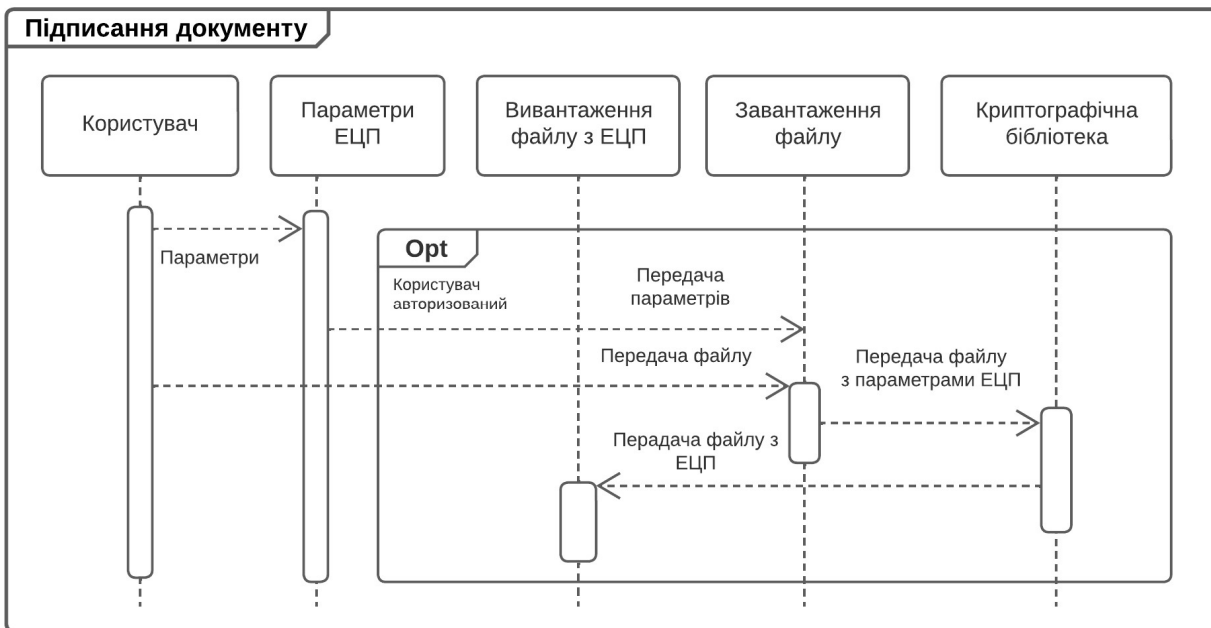


Рис. 2.5. Діаграма послідовності підписання документу

Перевірка документу на наявність електронного підпису може здійснюватися без авторизації в системі. На наступній діаграмі послідовності, зображено процес перевірки документу на наявність ЕЦП (Рис. 2.6.).

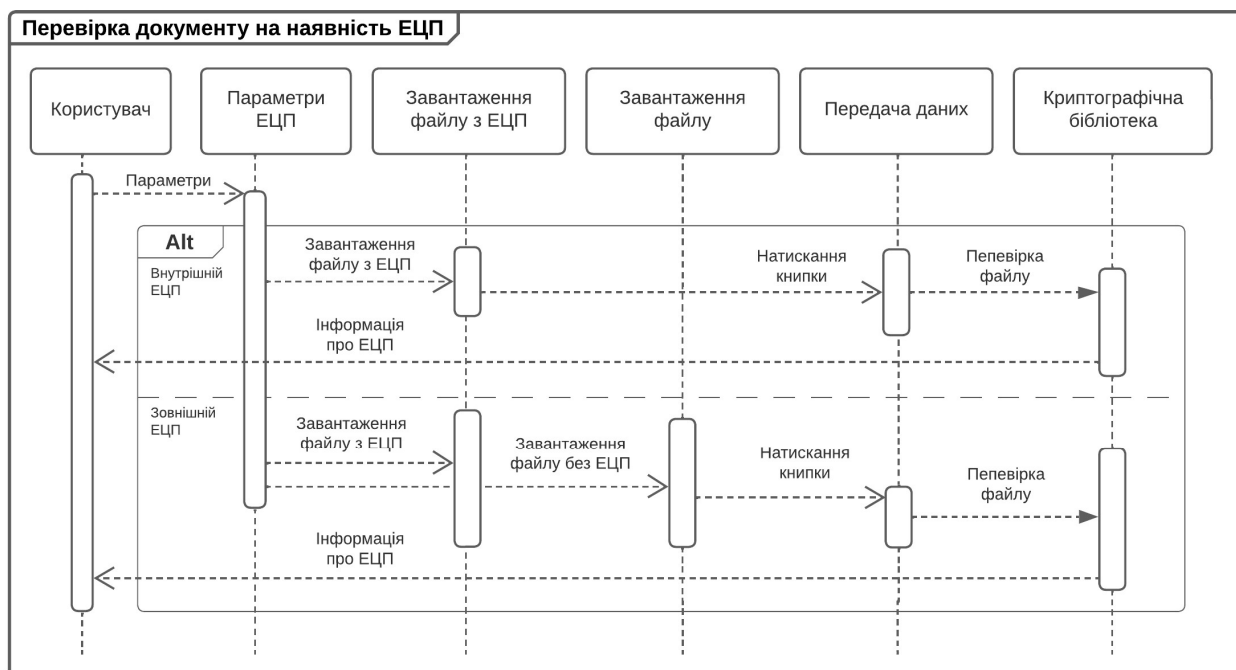


Рис. 2.6. Діаграма послідовності перевірки документу на наявність ЕЦП

При умові обрання користувачем параметру «Використовувати внутрішній підпис», він має можливість завантажити лише один документ для перевірки, в протилежній ситуації користувачу потрібно завантажити файл, на який накладався ЕЦП в його першочерговому стані та сам підписаний документ. При успішному циклу виконання, користувачу буде виведено інформацію про успішну перевірку, якщо користувач встановив параметр відображення інформації, буде виведено інформацію про підпис. Якщо документ пошкоджено або є невідповідність, то в результаті буде повернено помилку.

2.3. Криптографічне та алгоритмічне забезпечення

З стрімким поширенням писемності та інформації криптографія поступово перейшла до розділу самостійної науки. Появу перших криптосистем, ми можемо спостерігати вже на початку нашої ери. Значним імпульсом до розвитку криптографічних систем послужили роки Першої та Другої світових війн, де з 1925 року до закінчення Другої світової війни користувалась попитом шифрувальна машина

«Enigma», яка згодом була зламана британською розвідкою, що внесло помітний вклад в перемогу. З повоєнного часу і до наших днів поява обчислювальних систем значно прискорила розробку та удосконалення різних криптографічних методів.

Використання криптографічних методів в інформаційних системах зараз стала особливо актуальною. Розширилось використання комп'ютерних мереж, якими передаються великі обсяги інформації державного, комерційного, військового та особистого характеру, що не дає можливості доступитись до неї стороннім особам.

Шифрування поділяється на симетричні та асиметричні:

- Симетричний метод шифрування використовує один і той самий ключ як для шифрування так і для дешифрування;
- Асиметричний метод задіює два пов'язані ключі – пара ключів.

Симетричне шифрування має проблему того, що відбувається передача ключа, для розшифрування інформації, в такому разі ключ може бути перехоплено. Тоді як в асиметричному шифруванні є відкритий та закритий ключ. Відкритий ключ – це публічний ключ, до якого мають доступ ті, хто зашифрує інформацію. Закритий ключ – приватний, має перебувати лише в того, хто розшифрує інформацію. Тобто інформацію, яку було зашифровано за допомогою відкритого ключа можна розшифрувати лише з застосуванням того самого алгоритму і з використанням приватного ключа. Механізмом формування та перевірки ЕЦП використаємо криптографічний стандарт ДСТУ-4145, у якому за замовчуванням використовується функція гешування ГОСТ 34.311-95.

Хеш-функція або функція хешування – це те що створює унікальний цифровий відбиток з вихідної інформації. Кінцева хешована інформація називається хеш-сума (або просто хеш). Працює хеш-функція наступним чином, вона бере певну інформацію, наприклад, якийсь фрагмент тексту, пароль від акаунту або ж окремий файл та переоброблює дану інформацію в стрічку певної довжини, і довжина цієї стрічки

абсолютно завжди буде становити однакову довжину, незалежно від того, який розмір становила вхідна інформація. Вище сказане наведено графічно на рисунку (Рис. 2.7.).



Рис. 2.7. Процес хеш-функції

Тобто можемо сказати, що криптографічна функція хешування $H(m)$ або хеш-функція (hash-function) – це математичний алгоритм, що відтворює дані довільної розмірності бутового масиву фіксованого розміру.

Існує доволі велика кількість різних хеш алгоритмів, наприклад, sha-1, md5, sha256, sha 215 та інші. Наприклад, якщо ми проведемо моє ім'я «Bogdan» через хеш-функцію SHA-1, яка є однією з найрозповсюдженіших наряду з MD5 та SHA-2, то ми отримаємо стрічку довжиною 40 символів, яка буде виглядати так: 4d9f91b5e3391893c47f308803522cc337a2f140. Моє ім'я має два варіанти написання «Bogdan» та «Bohdan», які відрізняються лише зміною одного символу. Але якщо ми проведемо «Bohdan», використовуючи той же самий алгоритм SHA-1, ми також отримаєм стрічку довжиною 40 символів, яка матиме наступний вигляд: 83021eee554977c1c793520318c25bf334a690af. Якщо ми порівняємо ці два результати, то помітимо, що різниця між двома цими хешами суттєва, незважаючи на те, що нами було замінено лише один символ. Порівняння можна спостерігати на наступному рисунку (Рис. 2.8.).

Bogdan:	4d9f91b5e3391893c47f308803522cc337a2f140
Bohdan:	83021eee554977c1c793520318c25bf334a690af

Рис. 2.8. Порівняння двох хешів

Довжина хешу не залежить від довжини даних, які передаються для хешування, тобто довжина мого хешованого імені та довжина хешованого цього абзацу буде становити ті ж самі 40 символів. Нижче зображено все вище сказане у вигляді схеми (Рис. 2.9.).

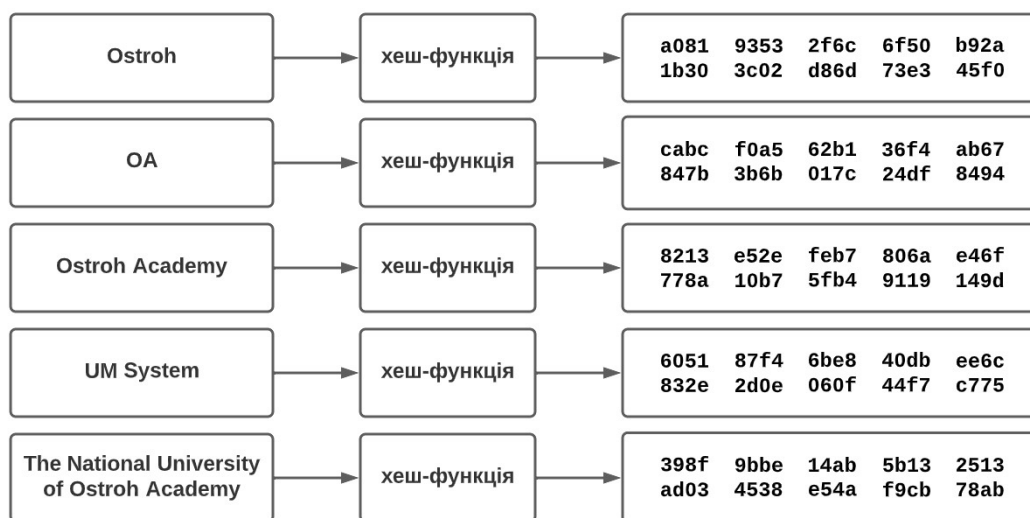


Рис. 2.9. Результат хешування різних повідомлень

При цьому всьому не може існувати два різні масиви даних, які були б перетвореними в однакові хеші. Так як це, називається, колізія, яка може бути присутня в більшості хеш-функцій. Але в надійних хеш функціях, частота появи колізії близька теоретичному нулю (Рис. 2.10.).



Рис. 2.10. Два різні хеші

Хеш-функція повинна відповідати наступному ряду вимог:

1. Однонаправленість. Хеш-функція працює лише в одну сторону, пропустивши інформацію через неї, ми не можемо отримати вихідне значення інформації знаючи значення хешу, вона повинна бути однонаправленою;
2. Детермінізм. Фіксований розмір одержуваного хешу незалежно від введеної інформації, що було продемонстровано на рисунку (Рис. 2.9.);
3. Відсутність колізії. Унікальність одержуваного хешу і практично повна відсутність колізії;
4. Висока швидкість хешування. Хеш-функція повинна швидко хешувати інформацію, що є вразливістю для брутфорс атак;
5. Наявність лавинного ефекту. Лавинний ефект забезпечує навіть при зміні одного байту в вихідній інформації отриманий хеш кардинально зміниться (Рис. 2.8.).

Практичне застосування хеш функція має при реєстрації користувачів, де логін та пароль пропускається через хеш функцію і в результаті отримане значення хешу записується в базу даних (це зроблено для того, щоб не зберігати логін та пароль в текстовому вигляді). Після авторизації логін та пароль знову проходять через хеш-функцію і в подальшому, значення отриманого хешу порівнюється з тим значенням хешу, що було записано в базу даних після реєстрації, якщо значення обох хешів ідентичні, то система авторизує користувача. Саме тут велику роль відіграє однонаправленість, що робить неможливим отримання вихідних даних, тому зламавши базу даних, замість логіну та пароля зловмисник отримає таблицю даних з хешами.

Ми ж використовуємо хешування для перевірки цілісності підписаних даних, а саме здійснюємо перевірку того, чи не були дані змінені після підписання. В стандарті ДСТУ-4145 задіяно алгоритм криптографічного хешування SHA-1 (Secure Hash Algorithm 1), який був розроблений Агентством національної безпеки США (АНБ) та оприлюднений Національним інститутом стандартів та технологій США як

федеральний стандарт. Даний стандарт користується довірою по всьому світу, та як правило, вимагається на всіх ПК, які використовуються урядом або збройними силами США. Даний алгоритм прийшов на заміну попереднім слабким алгоритмам, наприклад, MD5.

У нашому випадку алгоритм SHA-1 буде призначений для обчислення дайджесту документу, файлу або повідомлення. Коли вхідне повідомлення має довільну довжину $<2^{64}$ біт, SHA-1 видає 160-бітовий код, який називається дайджест повідомлення, що стане вихідною інформацією для алгоритму електронного підпису, який формується для верифікації повідомлення. Підпис повідомлення покращує ефективність процесу, так як дайджест в основному має розмір значно менший розміру повідомлення. Цей же самий алгоритм використовується для верифікацій цифрового підпису. Будь-яке внесення змін для модифікації повідомлення при передачі призведе до того, що дайджест зміниться і підпис не буде верифіковано.

Реалізація алгоритму SHA-1. Символ «+» вказує на операцію складання по модулю 2^{32} , F – нелінійна функція, \lll_n – зміщення коду ліворуч на n розрядів, W_t та K_t – константи, A, B, C, D, E – 32-бітові символи. Все це продемонстровано на наступній схемі (Рис. 2. 11.).

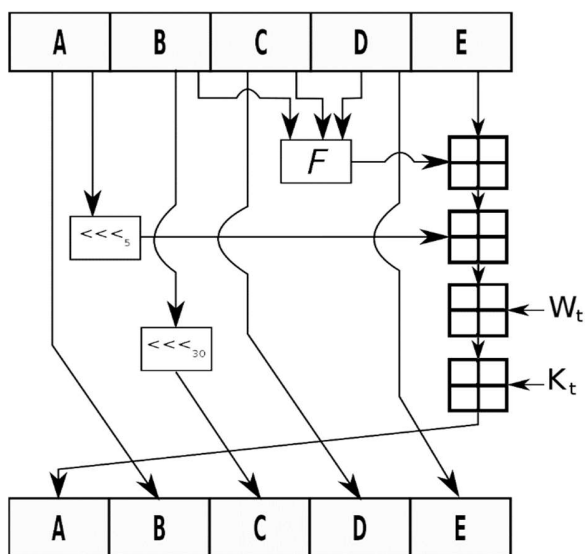


Рис. 2.11. Схема алгоритму SHA-1

В своїй основі одним з найефективніших та з великою надійністю є підхід, який використовується в розв'язуванні задач, які пов'язані з автентифікацією даних і джерелами повідомлень, являється процедура цифрового підпису, яка побудована на основі асиметричних криптографічних алгоритмах.

Процес обчислення електронно-цифрового підпису влаштований так, що кожен цифровий підпис має свою виняткову будову, пов'язаний з повідомленнями та уподібненими даними власника особистого ключа. Для встановлення істинності цифрового підпису, нам доводиться задіювати алгебраїчні співвідношення між ЕЦП і величинами, обрахованими за даними, відштовхуючись із зв'язку між відкритим ключем та особистим ключем. Даний зв'язок не дозволяє відновлення особистого ключа з відкритого. Виходячи з цього, відкритий ключ має унікальний параметр, який дозволяє зробити перевірку ЕЦП конкретній особі.

Електронний цифровий підпис в нашій системі накладається за допомогою ДСТУ-4145. Цей стандарт встановлює механізм цифрового підписування, який базується на властивостях груп точок еліптичних кривих над полями $GF(2^m)$ і правилах застосування даного механізму до повідомлень, що обробляються у комп'ютерних системах.

Зобразимо роботу алгоритму підписання документів наступній блок схемі (Рис. 2.12.).

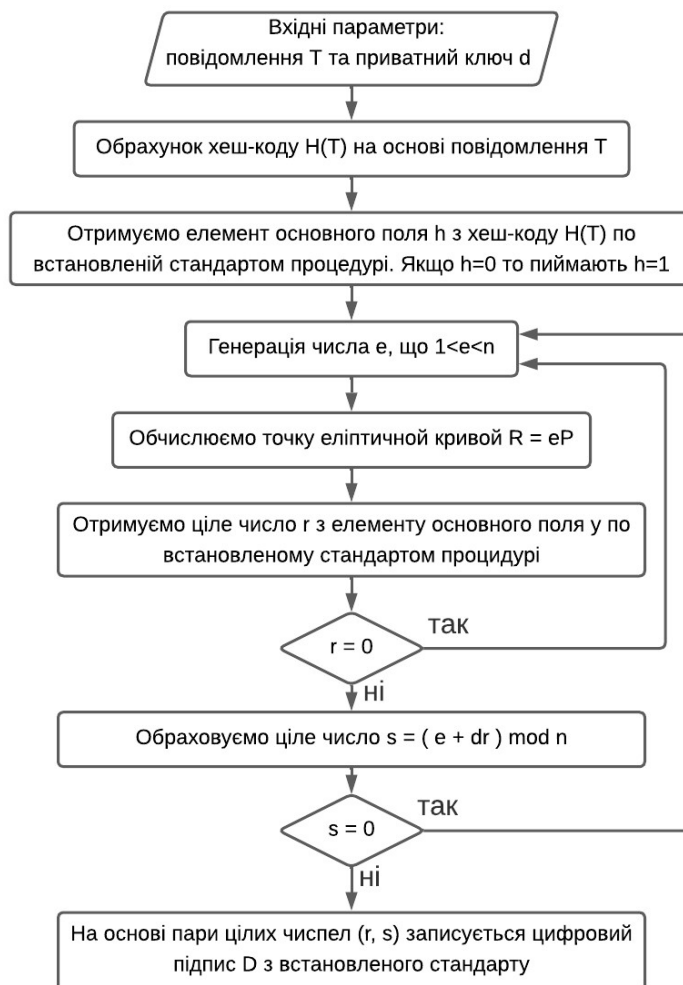


Рис. 2.12. Блок-схема алгоритму підписування

На блок-схемі нижче, відображено роботу алгоритму верифікації цифрового підпису (Рис. 2.13.).

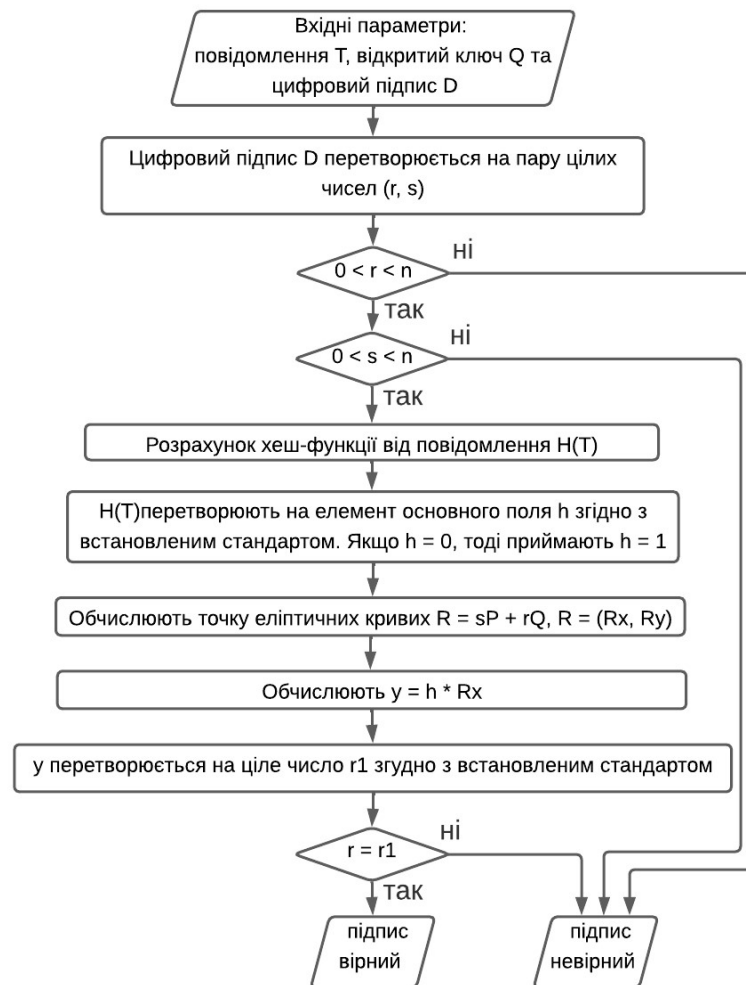


Рис. 2.13. Блок-схема алгоритму верифікації

Висновки до розділу 2

Таким чином, нами було описано предметну область нашого курсового проекту, що забезпечує здійснення таких функцій як: робота над файлами з метою накладання на них електронно-цифрового підпису, перевірка сертифікатів електронного підпису, зчитування (перевірка) інформації підписаних файлів та зчитування особистого ключа.

Було представлено схеми та UML- діаграми, такі як: діаграма послідовності, модель декомпозиції, контекстна модель та блок схеми функціонування алгоритмів підпису та верифікації.

Здійснено детальний огляд роботи хешування, хеш-функції, та які бувають хеш-алгоритми, який супроводжувався наочними схемами та рисунками. В алгоритмі роботи системи, буде задіюватись хеш-алгоритм SHA-1.

Здійснення аналізу предметної області, проектування системи та опис алгоритмічного функціонування нашої системи в подальшому допоможе нам для її реалізації.

РОЗДІЛ 3

ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Засоби розробки

Для розробки даного проєкту, було надано перевагу у використанні IDE Visual Studio Code, який є розповсюдженим і безкоштовним засобом для розробки та редагування сучасних веб-застосунків. Даний редактор містить в собі інструментарій для роботи з Git та GitHub, засоби рефакторингу, автоматичне виправлення типових конструкцій, підказку по коду та навігацію по коду.

Даний редактор підтримує велику кількість мов програмування та легко налаштовується під користувача, де ми можемо встановлювати різні комбінації клавіш та кольорові схеми. Доволі ефективним в розробці є використання стрічки пошуку, де ми можемо ввести запитване нами значення і редактор вкаже на місце, де воно знаходиться. При необхідності, ми можемо замінити значення за допомогою поля заміни.

Великою перевагою VS Code є те, що він містить в собі вбудований дебагер, що відрізняє його від інших редакторів коду. Після проведення налаштування, можна здійснювати пошук багів в коді прямо з редактора. Наявність консолі в редакторі дає змогу виводити результат роботи або ж повідомлення про помилку.

Також працюючи в цьому середовищі розробки, перед нами відкривається великий ряд можливостей при використанні розширень, які налаштовують та покращують VS Code, які включають в себе додаткові параметри, функції або сценарії застосування для існуючих вже інструментів.

Системою контролювання версії нашого проєкту виступала технологія Git та ресурс GitHub. Так як при розробці проєкту, ми регулярно вносимо зміни та оновлюємо наш код, виправляємо помилки та додаємо нові функції. Система контролювання версії

допомагає нам відслідковувати зміни в нашому проєкті, щоб мати змогу відновити старий код.

3.2. Вимоги до технічного та програмного забезпечення

Сховище файлів із сертифікатами та SHS повинно містити кореневі сертифікати CSC (файли з розширеннями * .cer, * .crt, * .p7b), з якими працюватиме бібліотека. При використанні серверів CSC (OCSP, TSP, CMP), необхідно додатково розміщувати сервісні сертифікати.

Параметри для доступу до сервера TSP для отримання позначки часу, вибираються в такому порядку:

1. Поле “Точка доступу до TSP-сервера” з сертифіката підписувача;
2. Поле “DNS імені або якогось іншого технічного засобу” з сертифіката TSP сервера того ж ЦСК, що і сертифікат підписувача.

У випадку, якщо поле “Точка доступу до TSP-сервера” з сертифіката підписувача чи поле “DNS імені або якогось іншого технічного засобу” з сертифіката TSP сервера не містить в собі порт для підключення, використовується значення “Port” для TSP сервера з параметрів крипто бібліотеки.

Відповідно до політики безпеки веб-браузерів, взаємодія з серверами CSC можлива лише за умови реалізації механізму міждоменної взаємодії. Щоб реалізувати взаємодію між доменами, сервер, на якому розташована бібліотека, повинен реалізувати механізм проксі-сервісу.

Бібліотека, робить синхронний запит (POST або GET) до сервера за протоколом HTTP. При використанні веб сервером інтерпретатора PHP буде мати вигляд:

```
/ProxyHandler.php?address=http://ca.server.ua, де
```

```
/ProxyHandler.php – адреса проксі-сервісу, встановлюється функцією бібліотеки SetXMLHTTPProxyService;
```

?address= - параметр, що передається в запиті, містить адресу на яку потрібно перенаправити запит бібліотеки;

<http://ca.server.ua> – адреса, на яку направлено запит бібліотеки.

В нашому випадку, систему було розгорнуто на портативному програмному середовищі Open Server Panel – це локальний веб сервер для Windows та програмне середовище, створено для веб розробки.

3.3. Опис програмної реалізації

Систему було реалізовано за допомогою опису ряду функцій, які взаємодіють з функціоналом, який нам надає бібліотека «ІТ бібліотека ЦСК».

1. Функція **initialize**, що являє собою функцію загального призначення реалізує ініціалізацію бібліотеки;

2. Функція **loadCAsSettings**, що реалізує завантаження налаштувань з файлу CAs.json, в якому описано дані для підключення до серверів ЦСК, дані передаються в елемент випадального меню form-select, для обрання користувачем ЦСК;

3. Функція **setDefaultSettings** встановлює параметри за замовчуванням, тобто встановлення адреси проксі-серверу HTTP-з'єднання бібліотеки, де створюються порожні об'єкти з параметрами серверів;

4. Функція **setCASettings** реалізовано встановлення параметрів центру сертифікації.

Наступні чотири функції взаємодіють з обраними даними елементу на сторінці form-select, а саме з випадальним меню вибору ЦСК;

5. Функція **getCAserver** отримує обраний CA Server;

6. Функція **loadCAserver** завантажує сервер ЦС;

7. Функція **storeCAserver** зберігає дані обраного сервера, які будуть відтворюватися після перезавантаження сторінки;

8. Функція **removeCAserver** видаляє збережені дані в системі про обраний ЦСК. Функція викликається функцією `removeStoredPrivateKey`;

9. Функція **storePrivateKey** зберігає закритий ключ в системі до тих пір, поки не буде викликана функція **removeStoredPrivateKey**;
10. Функція **removeStoredPrivateKey** видаляє збережений приватний ключ (видаляє дані авторизованого користувача). Викликається функцією **readPrivateKeyButtonClick** при умові, що користувач авторизований в системі;
11. Функція **selectPrivateKeyFile** застосовується для отримання файлу приватного ключа;
12. Функція **getPrivateKeyCertificatesByCMP** отримує сертифікати приватного ключа від CMP;
13. Функція **getPrivateKeyCertificates**, що застосовується для отримання сертифікатів приватного ключа. Викликається функцією **readPrivateKey**;
14. Функція **readPrivateKey** виконує читання особистого ключа. Викликається функціями **readPrivateKeyButtonClick** та **readPrivateKeyAsStoredFile**;
15. Функція **readPrivateKeyAsStoredFile** зчитує закритий ключ як збережений файл (після перезавантаження сторінки). Викликається функцією **initialize**;
16. Функція **readPrivateKeyButtonClick** викликається при натисканні на кнопку «Зчитати», виконує зчитування особистого ключа;
17. Функція **showOwnerInfo** викликається при натисканні на кнопку «Переглянути про власника» та виконує вивід на екран інформацію про власника (ПІБ, ЦСК та серійний номер);
18. Функція **showOwnCertificates** викликається при натисканні на кнопку «Переглянути сертифікати» та виконує вивід інформації про сертифікати користувача;
19. Функція **signData** викликається функцією **signFile** для отримання налаштувань про формування внутрішнього (підпис знаходиться разом з даними) ЕЦП або зовнішнього (підпис знаходиться окремо від даних) електронного цифрового підпису;

20. Функція **getSignTypeString** отримує інформацію про тип підпису та повертає текстову стрічку функції `verifyFile`, яка її викликає. Підпис може бути наступного формату:

- CAdES-BES (базовий);
- CAdES-T (з позначкою часу від ЕЦП);
- CAdES-C (з посиланням на повні дані для перевірки);
- CAdES-X Long (з повними даними для перевірки);
- CAdES-X Long Trusted (з повними даними ЦСК для перевірки).

21. Функція **verifyData** здійснює перевірку зовнішнього ЕЦП та виводить інформацію даних про підписаний документ. Викликається функцією `verifyFile`;

22. Функція **signFile** викликається при натисканні на клавішу «Підписати документ», виконує формування електронно-цифрового підпису;

23. Функція **verifyFile** здійснює перевірку внутрішнього ЕЦП та вивід інформації про підпис. Викликається функція при натисканні на кнопку «Перевірити документ на наявність підпису»;

24. Функція **useInternalSignCheckBoxClick** приймає значення від `type="checkbox"` з `id="InternalSignCheckbox"`, якщо значення `checked` то поле `input` з `id="FileWithSign"` буде `disabled`;

25. Функція **DSCAdESTypeChanged** приймає значення `value` з поля `select`, який має `id="DSCAdESTypeSelect"` для передачі формату підпису;

26. Функція **signAddContentTimestampCheckBoxClick** приймає значення від `type="checkbox"` з `id="SignAddContentTimestampCheckbox"`, якщо значення `checked`, то до підпису буде додано позначку часу від даних. За замовчуванням цей параметр рівний `TRUE`. Позначку часу буде додано лише за умови використання TSP-сервера в онлайн режимі роботи бібліотеки;

27. Функція **signAddCAsCertificatesCheckBoxClick** приймає значення від `type="checkbox"` з `id="SignAddCAsCertificatesCheckbox"`, якщо значення `checked`, то до

підпису буде додано повні дані для перевірки. Даний параметр за замовчуванням дорівнює TRUE;

28. Функція **privateKeyReaded** приймає параметр **isReaded**. Якщо приватний ключ зчитано, то поля для вибору ЦСК, особистого ключа та паролю приймуть властивість **disabled**.

3.4. Керівництво користувача

При запуску системи, першочергово представлено блок для авторизації, де користувач має обрати той ЦСК, яким йому було надано особистий ключ, далі завантажити свій особистий ключ, який має формат «.jks», ввести пароль захисту до особистого ключа та натиснути на кнопку «Зчитати». Все вище описане, зображено на наступному рисунку (Рис. 3.1.).

Рис. 3.1. Форма для авторизації

Якщо користувач ввів невірний пароль захисту особистого ключа або ж ключ пошкоджено, то система видає повідомлення, результат якого є помилка (Рис. 3.2.).

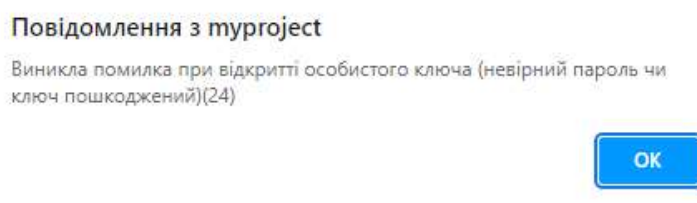


Рис. 3.2. Помилка авторизації

Якщо ж пароль правильний і ключ цілісний, то система виводить інформацію про користувача і авторизує (Рис. 3.3.).

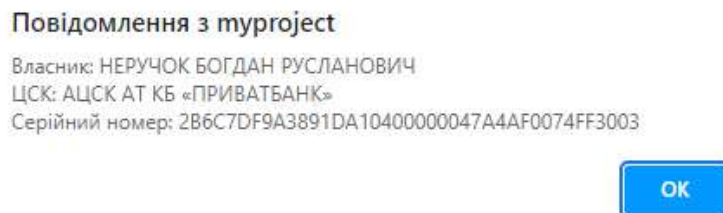


Рис. 3.3. Повідомлення про успішну авторизацію

Після чого, до натискання кнопки «стерти», усі поля стають недоступними для обрання та внесення інформації (Рис. 3.4.).

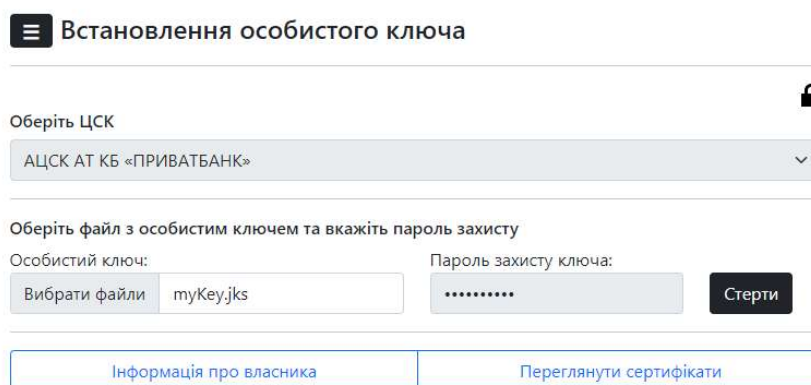


Рис. 3.4. Користувач авторизований в системі

Далі знаходяться дві кнопки: кнопка «Інформація про власника» виводить інформацію, яка з'являлася при авторизації, що було зображено на рисунку (Рис. 3.3.), друга кнопка «Переглянути сертифікати», виводять на екран інформацію про сертифікати користувача, що зображена на рисунку (Рис. 3.5.).

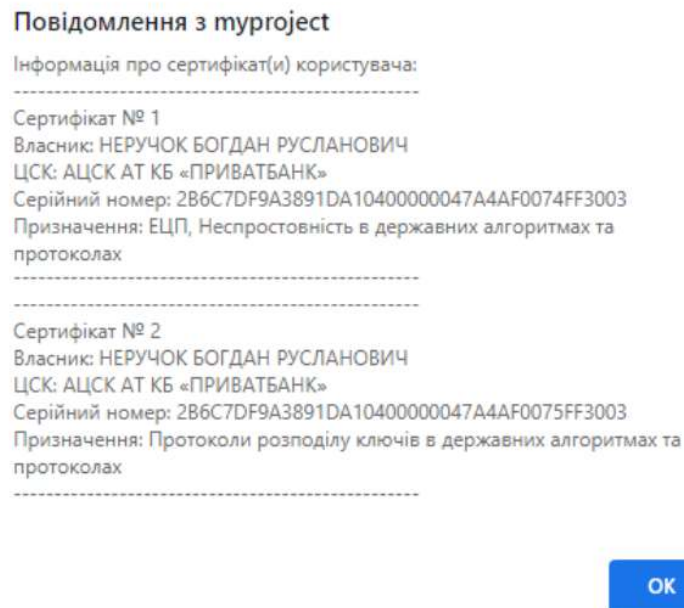
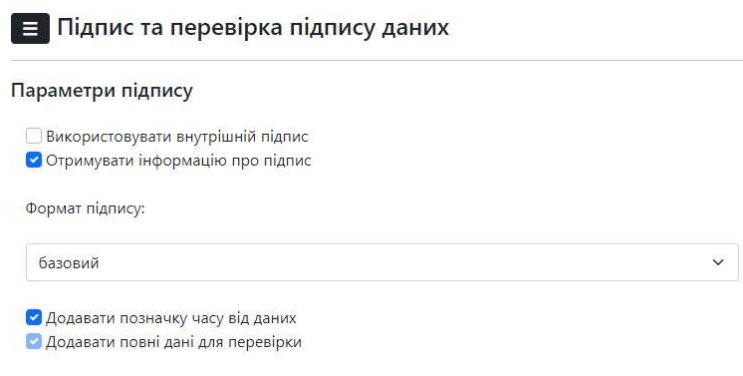


Рис. 3.5. Інформація про сертифікати користувача

Наступний блок містить функції для налаштування підпису, де користувач може вказати власні параметри, він надає такі можливості як:

- Використовувати внутрішній підпис (при використанні даного параметру, поле для обрання файлу з підписом буде недоступне і для перевірки підпису користувачу лише потрібно мати підписаний документ);
 - Отримувати інформацію про підпис (якщо прапорець не встановлено то при успішній перевірці підпису на екран буде виведено повідомлення «Підпис успішно перевірено» без уточнюючої інформації);
 - Формат підпису (користувач може обрати один з форматів представлених у випадаючому меню);
 - Додавати позначку часу від даних;
 - Додавати повні дані для перевірки;
- Все вище описане, зображено на наступному рисунку (Рис. 3.6.).



Підпис та перевірка підпису даних

Параметри підпису

Використовувати внутрішній підпис
 Отримувати інформацію про підпис

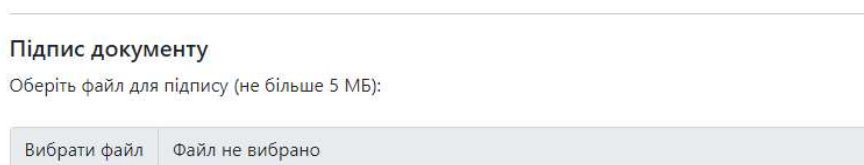
Формат підпису:

базовий

Додавати позначку часу від даних
 Додавати повні дані для перевірки

Рис. 3.6. Параметри ЕЦП

Якщо користувач не авторизований, то форма для підписання документу є йому не доступною (Рис. 3.7.).



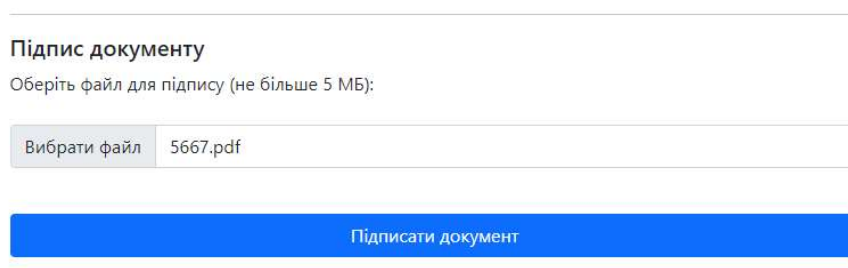
Підпис документу

Оберіть файл для підпису (не більше 5 МБ):

Вибрати файл Файл не вибрано

Рис. 3.7. Неактивна форма підписання

Якщо ж користувач знаходиться в системі, він має можливість завантажити файл (не більше 5 МБ) для підписання (Рис. 3.8.).



Підпис документу

Оберіть файл для підпису (не більше 5 МБ):

Вибрати файл 5667.pdf

Підписати документ

Рис. 3.8. Форма для підписання файлу

Підпишемо документ 5667.pdf, використовуючи внутрішній підпис та базовий формат підпису. Після натискання кнопки «підписати документ», на екран буде виведено повідомлення про те, що файл успішно підписано, після чого файл з накладеним ЕЦП буде завантажено на ПК (Рис. 3.9.).



Рис.3.9. Завантаження файлу

В наступному блоці, ми маємо змогу перевірити наш щойно підписаний документ. Для отримання інформації про накладений на нього ЕЦП (Рис.3.10.).

Перевірка підпису
Оберіть файл для перевірки (не більше 5 МБ):

Вибрати файл 5667.pdf.p7s

Оберіть файл з підписом:

Вибрати файл Файл не вибрано

Перевірити документ на наявність підпису

Рис.3.10. Форма для перевірки підпису

Після завантаження нашого документу з внутрішнім підписом до форми та натиснувши на кнопку перевірити документ на наявність підпису, ми отримуємо інформацію про підписаний документ, при умові, що в параметрах в нас стоїть прапорець «Отримувати інформацію про підпис» (Рис. 3.11.).

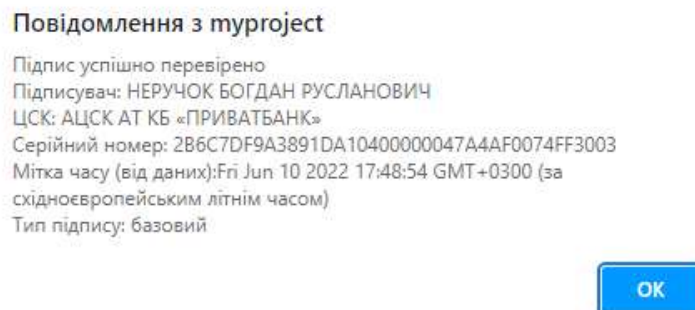


Рис.3.11. Інформація про перевірений ЕЦП

Перевіримо цей же самий файл на державному ресурсі: sign.dii.gov.ua/verify, результат можемо спостерігати на наступному рисунку (Рис. 3.12.).

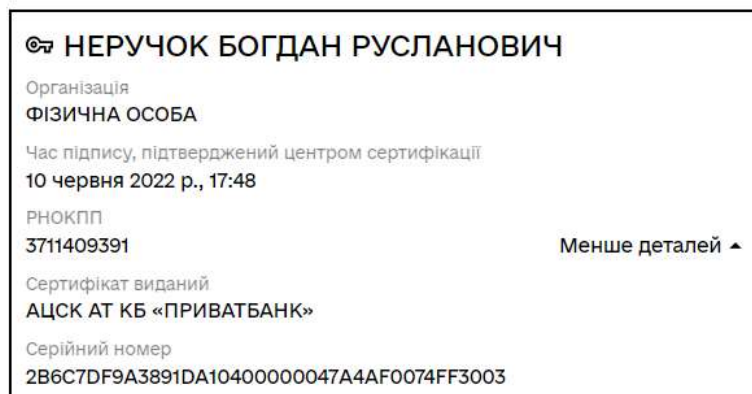


Рис.3.12. Інформація про перевірений ЕЦП

Тепер накладемо зовнішній ЕЦП на документ і формат підпису оберемо з повними даними ЦСК для перевірки. Після накладання підпису перевіримо файл, але для цього нам тепер потрібно обрати ще файл без ЕЦП, для перевірки, що зображено на рисунку (Рис. 3.13.).

Перевірка підпису

Оберіть файл для перевірки (не більше 5 МБ):

Вибрати файл 5667.pdf

Оберіть файл з підписом:

Вибрати файл 5667.pdf (1).p7s

Перевірити документ на наявність підпису

Рис. 3.13. Форма для перевірки документу

Як результат перевірки файлу з зовнішнім ЕЦП отримуємо повідомлення (Рис. 3.14.).

Повідомлення з турproject

Підпис успішно перевірено
 Підписувач: НЕРУЧОК БОГДАН РУСЛАНОВИЧ
 ЦСК: АЦСК АТ КБ «ПРИВАТБАНК»
 Серійний номер: 2B6C7DF9A3891DA10400000047A4AF0074FF3003
 Мітка часу (від даних): Fri Jun 10 2022 18:23:38 GMT+0300 (за
 східноєвропейським літнім часом)
 Мітка часу (від підпису): Fri Jun 10 2022 18:23:38 GMT+0300 (за
 східноєвропейським літнім часом)
 Тип підпису: з повними даними ЦСК для перевірки



Рис. 3.14. Вивід інформації про перевірений ЕЦП

Тепер перевіримо цей файл на ресурсі Міністерства цифрової трансформації України, результат можна спостерігати на наступному рисунку (Рис. 3.15.).

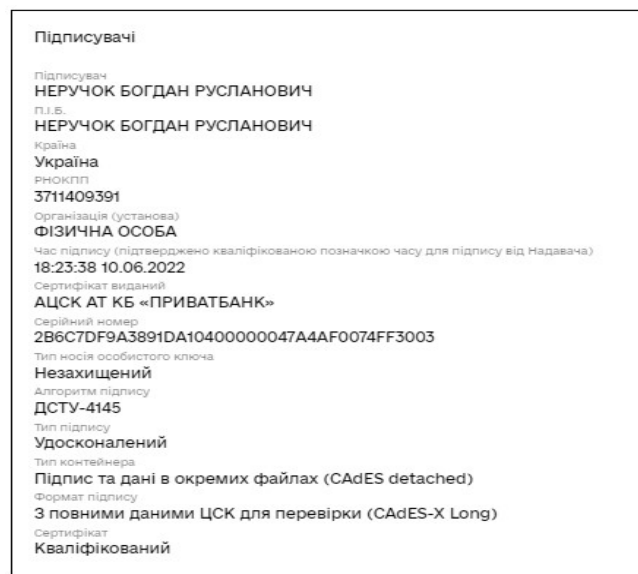


Рис. 3.15. Інформація про перевірений ЕЦП

За допомогою публічних державних ресурсів, було проведено процес тестування та перевірку на функціональність нашої системи, виконуваних всіх поставлених завдань шляхом спостереження за роботою, штучно створених ситуації, а саме накладання ЕЦП та перевірку підписаного документу з різними параметрами.

При відсутності інтернет з'єднання система не зможе додати позначку часу від даних, що в результаті не підтвердить час підпису і в такому разі підпис не буде відповідати вимогам Закону України «Про електронні довірчі послуги». Результат цієї помилки можна спостерігати далі (Рис. 3.16.).

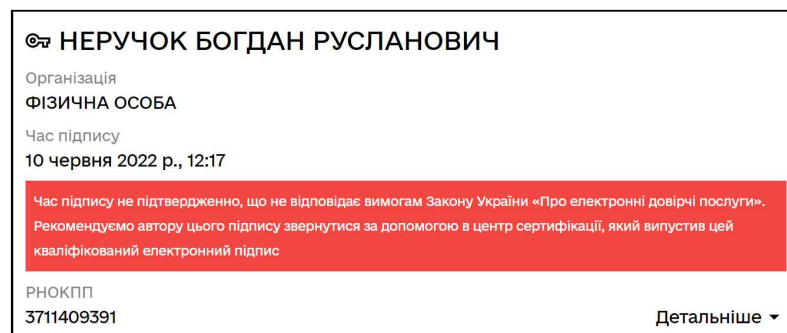


Рис. 3.16. Помилка про не підтвердження часу підпису

Висновки до розділу 3

В даному розділі було здійснено опис засобів розробки та обґрунтування саме їхнього застосування, за допомогою яких, було виконано розробку системи та усього необхідного функціоналу. Написання коду відбувалося в середовищі розробки Visual Studio Code та зручним контролюванням версії нашого проекту з використанням технології Git, що дозволяло нам розміщувати наш код на платформі GitHub та отримувати до нього доступ з будь-якого іншого ПК.

Описали вимоги до технічного та програмного забезпечення для реалізації зв'язку з серверами ЦСК. Наша система розгорталась на нашому ПК та реалізацію сервера було здійснено за допомогою портативної програми Open Server Panel.

Було здійснено детальний опис програмної реалізації, з точки зору, програміста, де було описано функції системи з використанням вхідних та вихідних даних.

І в результаті, було продемонстровано роботу системи, з точки зору, користувача. Опис різних етапів взаємодії користувача з системою. Таких як: авторизація користувача в системі за допомогою зчитування його особистого ключа та коду захисту до нього, вибір параметрів електронно-цифрового підпису, який в подальшому буде накладено на

документ, безпосередньо накладання ЕЦП та перевірку інформації про електронно-цифровий підпис внутрішнього чи зовнішнього формату.

ВИСНОВКИ

Отже, нами була пророблена робота, для реалізації системи: «University Management System електронний документообіг». Для початку, ми зробили опис предметного середовища, тобто було здійснено перелік технологій, які власне і були задіяні для реалізації даного проєкту, а це такі як: HTML, CSS, JavaScript, Bootstrap, PHP, ПТ бібліотека ЦСК, FileSaver.js. Ми коротко описали, що являє собою кожна з цих представлених технологій та аргументували, чому обрали саме ці дані засоби для розробки системи. Також нами було досліджено та проаналізовано вже існуючі аналоги, а саме: signy та document.online, ми визначили їхні переваги та недоліки. Та ще, нами було сформоване технічне завдання, де ми описали перелік вимог та задач, які повинна мати і виконувати наша система.

Далі, ми описали предметну область нашого проєкту, що забезпечує здійснення функцій, а саме: робота над файлами з метою накладання на них електронно-цифрового підпису, перевірка сертифікатів електронного підпису, зчитування (перевірка) інформації підписаних файлів та зчитування особистого ключа. Також, ми здійснили детальний огляд, як саме працює хешування та хеш-функції і які бувають хеш-алгоритми, нами наочно це було представлено у вигляді схем та рисунків. В алгоритмі роботи системи, буде задіюватись хеш-алгоритм SHA-1. Завдяки здійсненню аналізу предметної області, проєктування та опис алгоритму функціонування нашої системи в подальшому допоможе нам для того, щоб її реалізувати.

Також, ми зробили детальний опис засобів розробки та обґрунтування саме їхнього застосування, за допомогою яких, було виконано розробку системи та всього необхідного функціоналу. Написання коду було здійснено в середовищі розробки Visual Studio Code та зручним контролюванням версії нашого проєкту з використанням технології Git, що дало нам змогу розміщувати наш код на платформі GitHub та отримувати до нього доступ з будь-якого ПК.

Нами було описано вимоги до технічного і програмного забезпечення для реалізації зв'язку з сервером ЦСК. Також було здійснено детальний опис програмної реалізації, саме, з точки зору, програміста, де ми описали функції системи з використанням вхідних та вихідних даних.

Та в результаті, ми продемонстрували роботу системи, з точки зору, користувача. Нами був зроблений опис різних етапів взаємодії користувача з системою, а саме: авторизація користувача в системі за допомогою зчитування його особистого ключа та коду захисту до нього, вибір параметрів електронно-цифрового підпису, який в подальшому буде накладено на документ, безпосередньо накладання ЕЦП та перевірка інформації про електронно-цифровий підпис внутрішнього та зовнішнього формату.

Отже, завдяки цій системі, робота в Національному університеті «Острозька академія» може стати набагато зручнішою та швидшою.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Шаталов О., Кочубінський А. Криптографічний захист інформації цифровий підпис, що ґрунтується на еліптичних кривих. Розроблено Малим підприємством «Дина», 2003. № 31. С. 31.
2. Інститут інформаційних технологій. Програмний комплекс користувача ЦСК Бібліотека користувача ЦСК «Перенаправлення запитів серверів ЦСК (модуль розширення РНР)». Затверджений ЄААД.21107-13-ЛЗ. Харків, 2015. С.13.
3. Інститут інформаційних технологій. Програмний комплекс користувача ЦСК Бібліотека користувача ЦСК «Підпис (java-скрипт)». Затверджений ЄААД.21107-13-ЛЗ. Харків, 2021. С. 9.
4. АТ "ІІТ". URL: <https://iit.com.ua//index.php?page=itemdetails>ype=1&type=1&id=38>.
5. Open server. URL: <https://ospanel.io/>.
6. Github.com. URL: <https://github.com/bode4ok/Qualification-project>.
7. Bootstrap. URL: <https://getbootstrap.com/>.
8. FileSever.js. github.com. URL: <https://github.com/eligrey/FileSaver.js>.
9. Document.online. URL: <https://document.online/>.
10. Signy. URL: <https://signy.online/>.
11. Державні послуги онлайн. URL: <https://sign.djia.gov.ua>.
12. Центральний засвідчувальний орган Міністерство цифрової трансформації України. URL: <https://czo.gov.ua/verify>.