KAPITEL 6 / *CHAPTER 6* [6]
SERVER SOLUTION FOR CREATION AND GENERATION DOCUMENTS IN SPECIALIZED DOCUMENT MANAGEMENT SYSTEM

DOI: 10.30890/2709-2313.2024-27-00-023

Вступ

Щороку різні організації отримують безліч запитів у формі документів [1-5]. Кожен документ, що надсилається, має відповідати певним вимогам і складається вручну. Крім того, на основі цих документів складаються протоколи засідань та витяги після їх розгляду та ухвалення рішень у кожній інстанції. Такий підхід призводить до значного навантаження та зростає вразливість до бюрократичних помилок [6-9], які можуть бути не лише граматичними неточностями, а й помилками під час передачі документів для подальшої обробки. Усі ці аргументи стосуються процесу ручної обробки документів, що робить тему дослідження *актуальною*.

Тому потрібно аналізувати та вдосконалювати методи обробки документів, зокрема збереження їх цілісності та автентичності, а також автоматизовану генерацію. Основна увага зосереджується на процесах асемблювання та реасемблювання документів.

Метою роботи є підвищення відповідності спеціалізованої автоматизованої системи документообігу (САСД) основним принципам електронного документообігу за допомогою серверного рішення процедур асемблювання та реасемблювання документів.

У даній главі монографії показано опис алгоритму роботи САСД, розроблено алгоритми процесів обробки документів на рівні кожного з департаментів. Також тут буде представлено ключові механізми САСД та їх взаємодія на рівні клієнт-сервер, зокрема формування JWT токенів для ідентифікації користувачів в програмі та хешування документів з метою запобігання їх дублікації. Також планується виконати практику відповідності розробки основним принципам САСД.

[6] ***Authors:*** *Korobeinikova Tetiana Ivanivna*

Для досягнення поставленої мети необхідно виконати такі *завдання*:

1) Розробити алгоритм роботи САСД на рівні кожного з департаментів;

2) Описати ключові механізми САСД та їх взаємодія на рівні клієнт-сервер;

3) Провести тестування серверного рішення асемблювання та реасемблювання документів у САСД.

Об'єктом дослідження є процеси асемблювання та реасемблювання документів на рівні кожного з департаментів. Процеси асемблювання та реасемблювання сприяють оптимізації опрацювання документів під час складання безпосередніх файлів. Такий ефект досягнуто шляхом автоматизованого створення файлів та генерації їх вмісту програмно.

Предметом дослідження є методи та засоби автоматизованої обробки документів.

Технології, які будуть використовуватись у розробці: Spring-Boot Framework, Hibernate, Docker, AWS EC2, AWS S3, AWS RDS, та MySQL. Кожна з цих технологій відіграє важливу роль у забезпеченні ефективності, безпеки та доступності розробленої системи. Такий підхід до розробки дозволить покращити процес обігу документів у великих організаціях та установах.

Практична цінність полягає у такому:

– Розроблено серверне рішення для асемблювання та реасемблювання документів у спеціалізованій системі документообігу;

– Розроблено покращений алгоритм асемблювання та реасемлювання документів, що надає можливість їх автоматизованої генерації та миттєвої обробки значних об'ємів файлів.

Ключовими *перевагами* запропонованої САСД можуть стати:

1) Автоматичне формування витягів з протоколів засідань структурних підрозділів;

2) Пересилання витягів з протоколів між структурними підрозділами;

3) Використання електронно-цифрового підпису для витягів;

4) Автоматизоване створення порядку денного для протоколів засідань структурних підрозділів на основі вхідних витягів з підрозділів нижчого рівня;

5) Редагування та зберігання документів у різних форматах (doc, excel, pdf);

6) Можливість розширення для використання у різних задачах.

6.1. Основні поняття та основа для реалізації

Після аналізу та оцінки отриманих даних виникає необхідність у розробці САСД, що автоматизує процес створення витягів з протоколів засідань структурних підрозділів та обміну документами між ними. Цей додаток дозволить редагувати, зберігати та передавати документи між установами, спрощуючи процес їх подання та розгляду на всіх рівнях бюрократичної ієрархії. Крім того, він забезпечить можливість застосування електронно-цифрового підпису для документів [10-13].

САСД – це така система документообігу, первинною ціллю якої є автоматизація процесу обробки документів та всіх мануальних процесів пов’язаних з ними. Технологічний ланцюжок САСД наведено на рисунку 1.

Основна ідея для реалізації серверної частини САСД полягає у застосування процедур реасемблювання та асемблювання документів.

Реасемблювання – це процес розбиття документа на окремі частини, використовуючи спеціальні маркери та регулярні вирази для подальшого аналізу цих частин програмою. Цей процес передбачає перетворення документа на програмні об'єкти для подальшого використання системою. Процес реасемблювання показано на рисунку 2. Під час цього процесу система аналізує наданий протокол і автоматично створює витяги для нього.

Асемблювання – це процес, який протистоїть реасемблюванню і полягає в перетворенні документа з програмного стану в фізичний для подальшої обробки. Зазвичай цей процес включає в себе використання порожнього (незаповненого) шаблону документа, який програма заповнює даними. Під час асемблювання формується документ за допомогою доступних програмі об'єктів та даних. Процес асемблювання показано на рисунку 3.

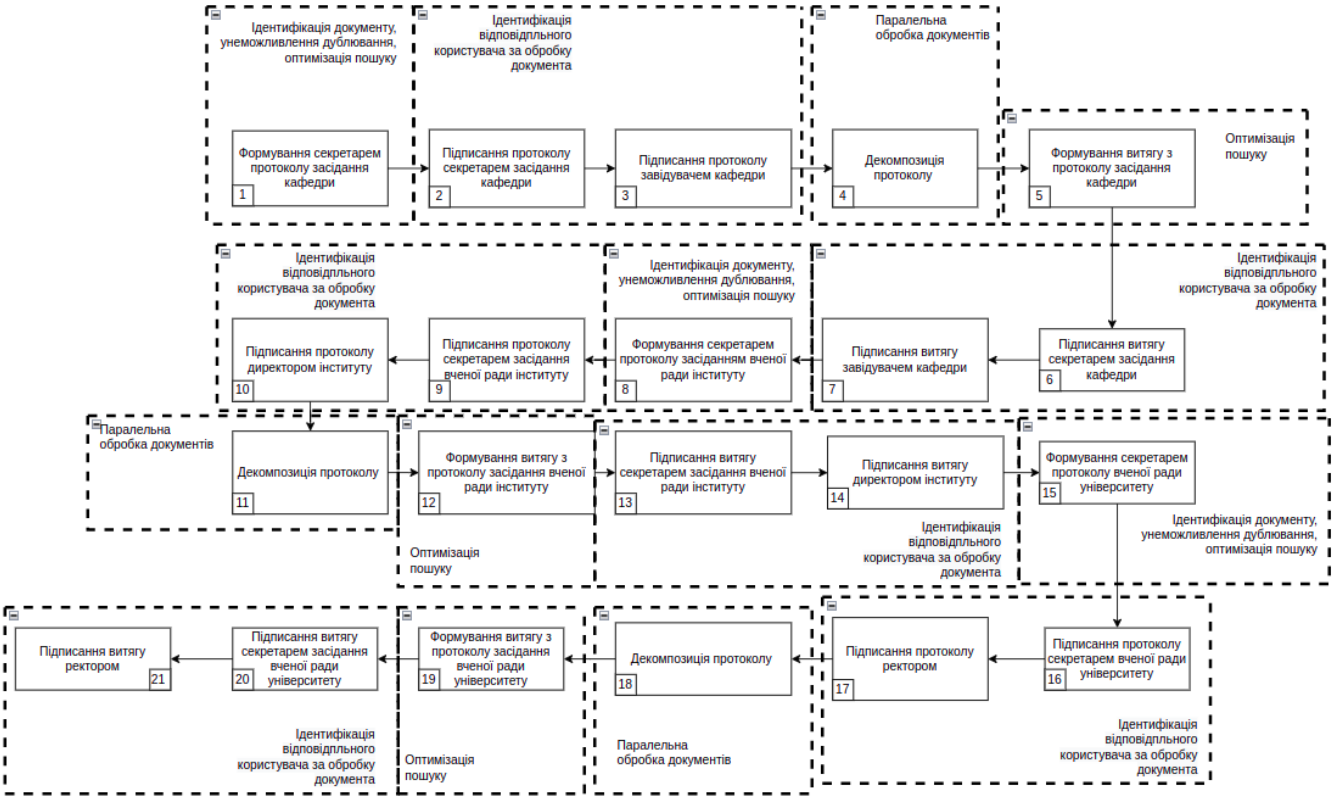


Рисунок 1 – Технологічний ланцюжок САСД

Авторська розробка

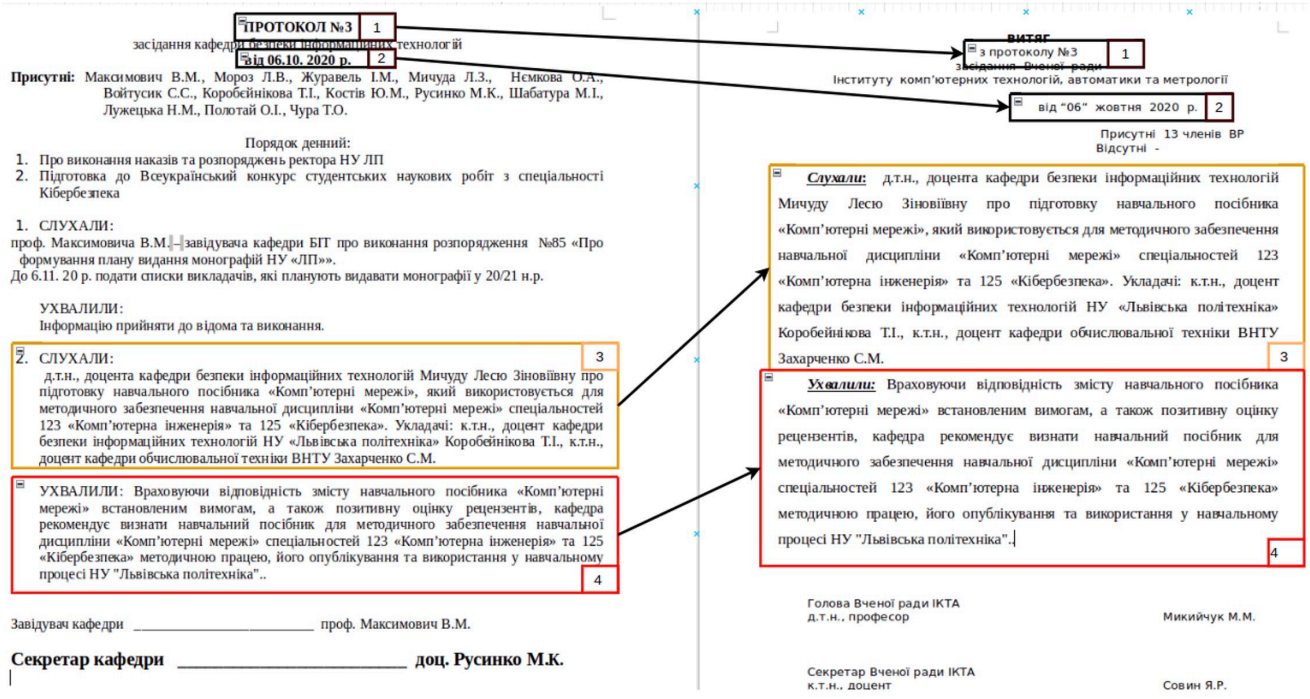
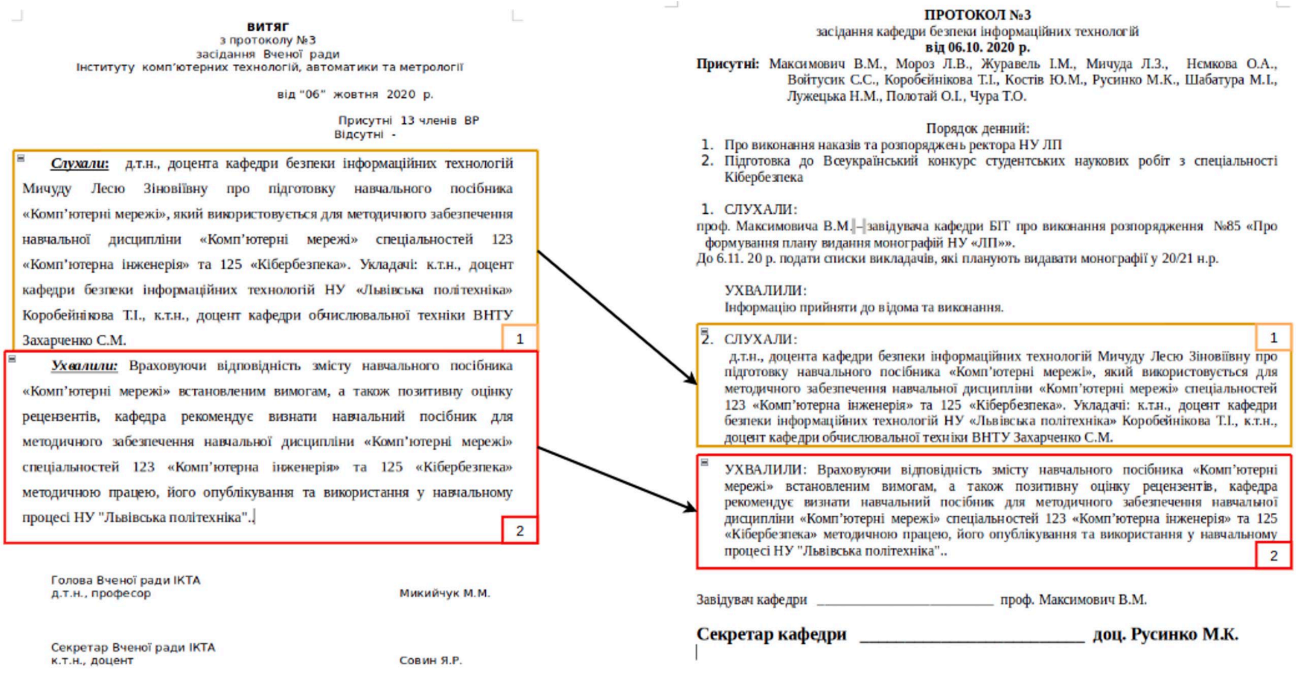


Рисунок 2 – Візуалізація процесу реасемблювання

Авторська розробка



ВИТЯГ
з протоколу №3
засідання Вченої ради
Інституту комп'ютерних технологій, автоматики та метрології
від "06" жовтня 2020 р.
Присутні 13 членів ВР
Відсутні -

Слухали: д.т.н., доцента кафедри безпеки інформаційних технологій Мичуду Лесю Зіновіївну про підготовку навчального посібника «Комп'ютерні мережі», який використовується для методичного забезпечення навчальної дисципліни «Комп'ютерні мережі» спеціальностей 123 «Комп'ютерна інженерія» та 125 «Кібербезпека». Укладачі: к.т.н., доцент кафедри безпеки інформаційних технологій НУ «Львівська політехніка» Коробейнікова Т.І., к.т.н., доцент кафедри обчислювальної техніки ВНТУ Захарченко С.М. 1

Ухвалили: Враховуючи відповідність змісту навчального посібника «Комп'ютерні мережі» встановленим вимогам, а також позитивну оцінку рецензентів, кафедра рекомендує визнати навчальний посібник для методичного забезпечення навчальної дисципліни «Комп'ютерні мережі» спеціальностей 123 «Комп'ютерна інженерія» та 125 «Кібербезпека» методичною працею, його опублікування та використання у навчальному процесі НУ "Львівська політехніка". 2

Голова Вченої ради ІКТА
д.т.н., професор Микийчук М.М.

Секретар Вченої ради ІКТА
к.т.н., доцент Совин Я.Р.

ПРОТОКОЛ №3
засідання кафедри безпеки інформаційних технологій
від 06.10. 2020 р.

Присутні: Максимович В.М., Мороз Л.В., Журавель І.М., Мичуда Л.З., Немкова О.А., Войтусик С.С., Коробейнікова Т.І., Костів Ю.М., Русинко М.К., Шабатура М.І., Лужецька Н.М., Полотай О.І., Чура Т.О.

Порядок денний:
1. Про виконання наказів та розпоряджень ректора НУ ЛП
2. Підготовка до Всеукраїнський конкурс студентських наукових робіт з спеціальності Кібербезпека

1. СЛУХАЛИ:
проф. Максимовича В.М. – завідувача кафедри БІТ про виконання розпорядження №85 «Про формування плану видання монографій НУ «ЛП»».
До 6.11. 20 р. подати списки викладачів, які планують видавати монографії у 20/21 н.р.

УХВАЛИЛИ:
Інформацію прийняти до відома та виконання.

2. СЛУХАЛИ: 1
д.т.н., доцента кафедри безпеки інформаційних технологій Мичуду Лесю Зіновіївну про підготовку навчального посібника «Комп'ютерні мережі», який використовується для методичного забезпечення навчальної дисципліни «Комп'ютерні мережі» спеціальностей 123 «Комп'ютерна інженерія» та 125 «Кібербезпека». Укладачі: к.т.н., доцент кафедри безпеки інформаційних технологій НУ «Львівська політехніка» Коробейнікова Т.І., к.т.н., доцент кафедри обчислювальної техніки ВНТУ Захарченко С.М.

УХВАЛИЛИ: Враховуючи відповідність змісту навчального посібника «Комп'ютерні мережі» встановленим вимогам, а також позитивну оцінку рецензентів, кафедра рекомендує визнати навчальний посібник для методичного забезпечення навчальної дисципліни «Комп'ютерні мережі» спеціальностей 123 «Комп'ютерна інженерія» та 125 «Кібербезпека» методичною працею, його опублікування та використання у навчальному процесі НУ "Львівська політехніка".. 2

Завідувач кафедри _____ проф. Максимович В.М.

Секретар кафедри _____ **доц. Русинко М.К.**

Рисунок 3 – Візуалізація процесу асемблювання

Авторська розробка

Підвищення ефективності обробки документів шляхом нівелювання недоліків мануального підходу обробки документів використовуючи хмарні технології та засоби парсингу для автоматичного формування нових документів та швидкого отримання інформації, що може бути оброблена програмно.

6.2. Призначення САСД та постановка типової задачі

У більшості випадків в процесі обробки паперових документів у державних установах люди стикаються з незручними недоліками. Такі недоліки перетворюють документ на предмет, що вимагає постійної уваги та супроводу з боку особи, що його ініціює [17-22]. Одним з таких недоліків є різнорівневість інстанцій.

Алгоритм автоматизованої системи документообігу, що пропонується у даній роботі покликаний спростити цю проблему та покращити досвід користування послугами закладів з високою бюрократичною завантаженістю.

Даний алгоритм буде частково базуватись на прикладах систем роботи з документами описаних в попередньому розділі.

Окрім створення та збереження документів в хмарі запропонована автоматизованої системи документообігу буде надавати такі можливості:

– закріплення документу за певним користувачем;

– автоматизоване створення витягів з протоколів шляхом реасемблювання документу;

– автоматизоване створення протоколів на основі баз даних, що надаються користувачами;

– повністю автоматизований процес розгляду, підписання документу та його обробка в інстанціях всіх рівнів без участі користувача.

Спрощена постановка тривіальної задачі: є потреба для видавництва Львівської Політехніки надати витяги з протоколу засідання вченої ради університету, інституту та засідання кафедри. Можна це реалізувати особисто пройшовши по всім інстанціям, а можна автоматизувати цей процес та використати переваги інформаційних технологій. Відповідно до поставленої задачі, нижче буде описано роботу ЕСД між структурними одиницями у вузі.

На засіданні кафедри секретар веде протокол засідання кафедри (ПЗК), з ПЗК роблять витяги ПЗК, система буде містити стандартну форму ПЗК та витягу з ПЗК. З даних ПЗК система автоматично буде формувати витяги по всім питанням. Далі буде можливість зробити цифровий підпис для сформованих документів, а після цього – скерувати в який саме підрозділ в подальшому йде витяг ПЗК (наприклад, на вчену раду (ВР) інституту). Якщо витяг не потрібен – буде передбачено процес неформування витягу. Також буде можливість зберігання витягу і звертання до нього коли він передбачений, але зберігається на кафедрі. В результаті користувачу буде доступний підписаний витяг ПЗК секретарем, а також можливість надіслати його на цифровий підпис завідувачу кафедри. Знову ж, користувачеві стає доступним підписаний витяг з ПЗК секретарем та завкафедрою, а секретарю – можливість підтвердити надсилання до секретаря ВР інституту (СВРІ). Після цього, у СВРІ з'являється можливість

підтвердити прийняття витягу, у випадку підтвердження – система автоматично включає дані з витягу ПЗК у протокол засіданні ВР інституту (ЗВРІ), у результаті чого формується проект протоколу ЗВРІ, у який, за потреби, СВРІ може вносити зміни, додавати нові пункти. Після цього, у СВРІ з'являється можливість його підписати. Далі користувачу стає доступним підписаний витяг ПЗВРІ секретарем, який автоматично надсилається директору інституту, у якого з'являється можливість його підписати. У результаті – користувач має підписаний витяг з ПЗВРІ секретарем та директором, а СВРІ має можливість скерувати витяг до секретаря ВР університету (СВРУ), якщо є така потреба. Далі СВРУ може прийняти витяг, у випадку прийняття – система автоматично включає дані з витягу СВРІ у протокол ЗВРУ та формує проект протоколу ЗВРУ, який, також, у разі потреби, СВРУ може редагувати чи додавати нові пункти. Після завершення внесення змін, СВРУ має можливість його підписати. Знову ж користувачу стає доступним підписаний витяг ПЗВРУ секретарем, який також надсилається ректору для підписання. Як результат – користувач має підписаний витяг з ПЗВРУ секретарем та ректором. Останній крок – підписаний витяг з ПЗВРУ – надсилається голові видавництва ЛП.

6.**3. Розробка алгоритму роботи САСД**

Розглянемо алгоритм роботи САСД поетапно на прикладі обробки документу на рівні кожного з департментів:

1) на рівні кафедри – структурного підрозділу 1-го рівня.
2) на рівні вченої ради інституту – структурного підрозділу 2-го рівня.
3) на рівні вченої ради університету– структурного підрозділу 3-го рівня.

6.3.1. Обробка документів на рівні кафедри

Обробка документів на рівні кафедри передбачає врахування операцій асемблювання та реасемблювання, отримання підписів акторів структурного

підрозділу першого рівня: секретаря та (рис. 4).

1) Початок алгоритму

2) Користувач створює необхідний файл, а система завантажує його у хмару і реєструє запис в БД. Створений документ синхронізується (для перегляду на інших пристроях користувача) і надсилається секретареві обраного департаменту. Новоствореному документу надається помітка "НАДІСЛАНО".

3) Секретар отримує сповіщення про отриманий файл і включає даний документ до протоколу засідання кафедри. У той же час файлу включеному в протокол надається помітка "ЗАТВЕРДЖЕНО". В іншому разі відхилений файл повертається до автора документу з поміткою "ВІДХИЛЕНО".

4) У разі якщо секретар не включив файл до протоколу засідання кафедри відхилений файл повертається до автора документу з поміткою "ВІДХИЛЕНО".

5) Далі відбувається оффлайн засідання кафедри на якому розглядаються документи включені секретарем до протоколу.

6) Секретар створює його фінальну версію (з рішенням засідання стосовно кожного з документів) використовуючи інтерфейс програми. Для цього секретарю необхідно надати інформацію стосовно дати проведеного засідання, присутніх членів, порядку деннного, матеріали розглянутих питань і рішень по кожному з питань. Після чого надані матеріали будуть автоматично сформовані у кінцеву версію протоколу. Після формування кінцевої версії протоколу, протокол засідання кафедри (ПЗК) підписується електронно-цифровим підписом (ЕЦП) секретаря кафедри. Далі протокол сканується на предмет визначення документів необхідних для витягу для кожного із надсилаючих користувачів.

7) САСД визначає файли, що мають бути сформовані у витяг.

8) У разі якщо надсилаючий користувач не запросив витяг надісланого ним документу – документ залишається в ПЗК з поміткою "ЗАКРИТО".

9) У разі якщо документ необхідно оформити у витяг з протоколу вибираються всі матеріали, що стосуються даного витягу і формуються в окремий файл. Сформованому файлу надається помітка "ПІДПИСАНО".

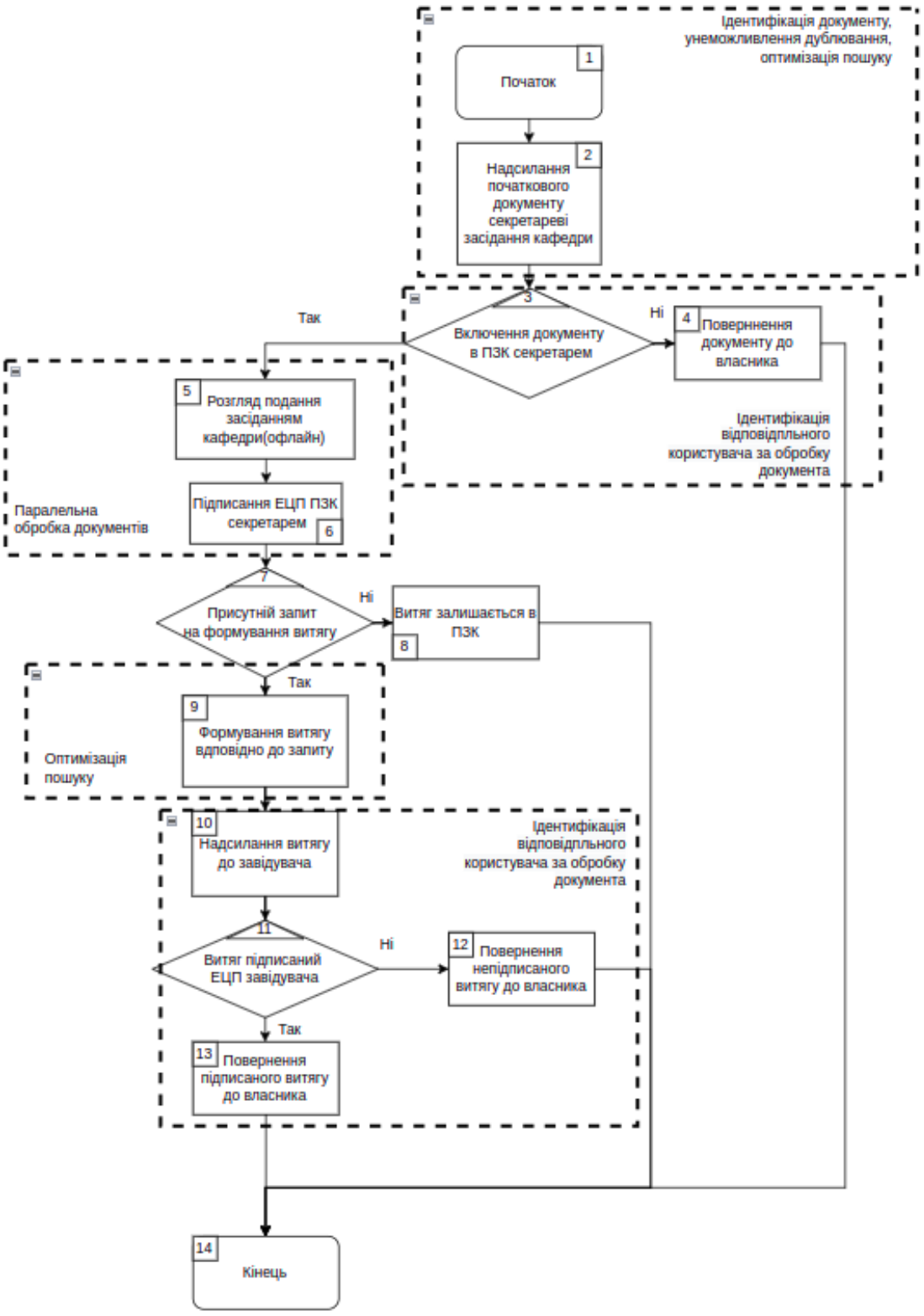


Рисунок 4 – Алгоритм обробки документів на рівні засідання кафедри

Авторська розробка

10) Далі всі витяги, що вимагають підпису завідувача кафедри надсилаються завідувачеві, про що останній отримує відповідне сповіщення.

Перш ніж надіслати такі витяги програма перевіряє кожен з них на предмет присутності підпсису секретаря засідання кафедри для кожного з них в БД.

11) Завідувач кафедри переглядає документи надіслані йому задля отримання підпису і підписує потрібні. На цьому етапі програма викликає окремий сервіс для генерації підпису завідувача, прив'язки його до відповідного документу і створення запису в БД.

12) Після успішного підписання документи автоматично надсилаються секретареві вченої ради інституту і їм надаєтсья мітка "НАДІСЛАНО".

13) У разі якщо документ не було підписано завідувачем кафедри сформований витяг надсилається до його автора, а мітка документу змінюється на мітку "ВІДХИЛЕНО".

14) Кінець алгоритму

6.3.2. Обробка документів на рівні вченої ради інституту

Обробка документів на рівні кафедри передбачає врахування операцій асемблювання та реасемблювання, отримання підписів акторів структурного підрозділу другого рівня: секретаря та директора інституту. Блок-схема алгоритму обробки документів на рівні кафедри є на рис. 5 і містить такі кроки:

1) Початок алгоритму

2) На етапі надсилання підписаного витягу секретареві вченої ради інституту (СВРІ), секретар отримує сповіщення про надісланий йому документ. Перш ніж надіслати такі витяги програма перевіряє кожен з них на предмет присутності підпсису завідувача кафедри для кожного з них в БД.

3) Секретар включає надісланий йомудокумент до протоколу засідання вченої ради інституту (ПЗВРІ). У той же час файлу включеному в протокол надається помітка "ЗАТВЕРДЖЕНО". В іншому разі відхилений файл повертається до автора документу з поміткою "ВІДХИЛЕНО".

4) У разі якщо секретар не включив файл до протоколу засідання кафедри відхилений файл повертається до автора документу з поміткою "ВІДХИЛЕНО"

5) Далі відбувається оффлайн засідання вченої ради інституту на якому

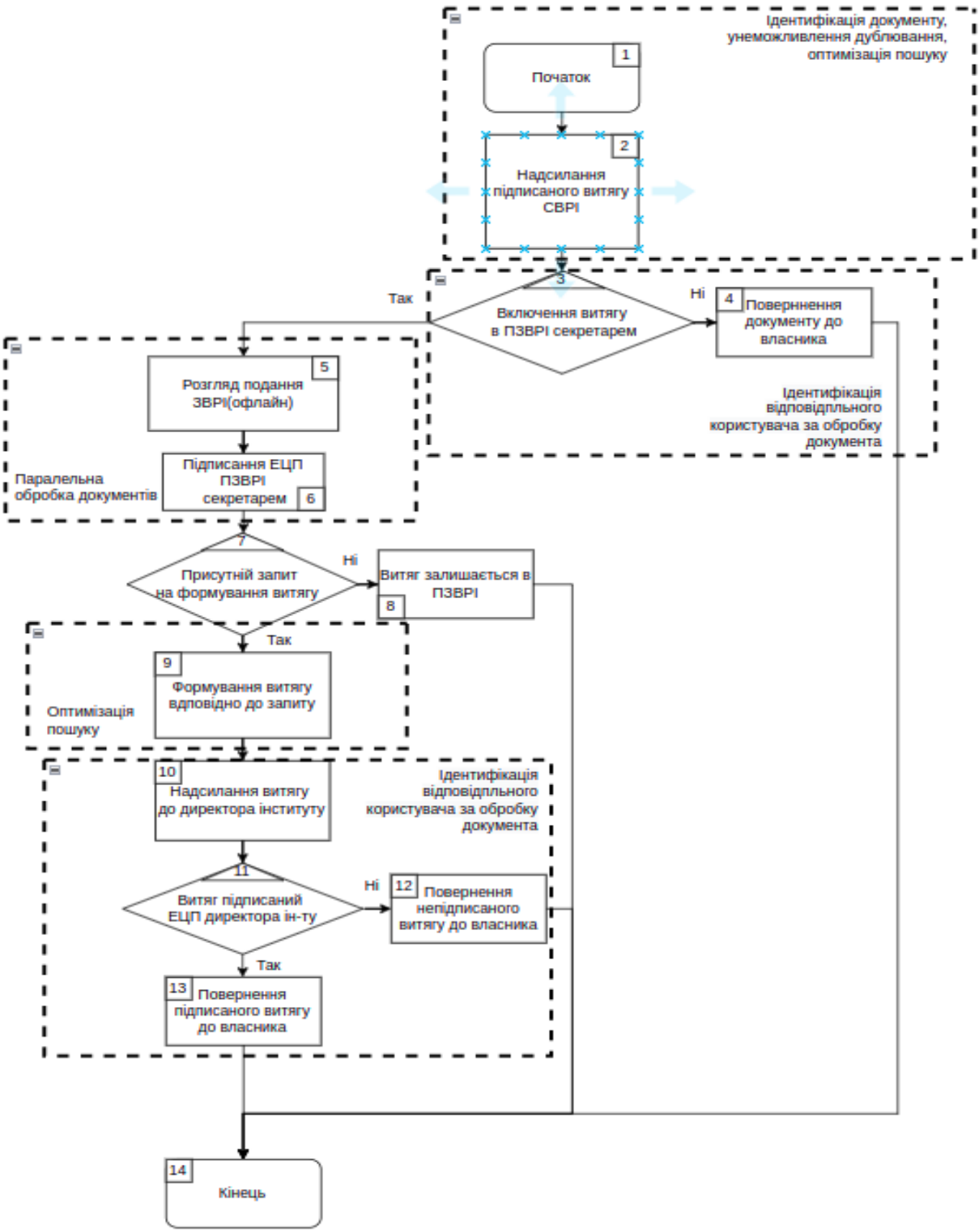


Рисунок 5 – Алгоритм обробки документів на рівні засідання вченої ради інституту

Авторська розробка

6) розглядаються документи включені секретарем до протоколу.

7) На етапі підписання протоколу засідання вченої ради інституту (ПЗВРІ) секретар створює його фінальну версію (з рішенням засідання стосовно кожного з документів) використовуючи інтерфейс програми. Для цього секретарю необхідно надати інформацію стосовно дати проведеного засідання, присутніх членів засідання, порядку деннного, матеріали розглянутих питань і рішень винесених по кожному з питань окремо. Після чого надані матеріали будуть автоматично сформовані у кінцеву версію протоколу. Після формування кінцевої версії протоколу, протокол підписується електронно-цифровим підписом (ЕЦП) секретаря, Далі протокол сканується на предмет визначення документів необхідних для витягу для кожного із надсилаючих користувачів.

8) На цьому етапі система визначає файли, що мають бути сформовані у витяг.

9) У разі якщо надсилаючий користувач не запросив витяг надісланого ним документу – документ залишається в ПЗВРІ з поміткою “ЗАКРИТО”.

10) У разі якщо документ необхідно оформити у витяг з протоколу вибираються всі матеріали, що стосуються даного витягу і формуються в окремий файл. Сформованому файлу надається помітка “ПІДПИСАНО”.

11) Далі всі витяги, що вимагають підпису директора інституту надсилаються директорові, про що останній отримує відповідне сповіщення. Перш ніж надіслати такі витяги програма перевіряє кожен з них на предмет присутності підпсису секретаря засідання вченої ради інституту для кожного з них в БД.

12) Директор інституту переглядає документи надіслані йому задля отримання підпису і підписує потрібні. На цьому етапі програма викликає окремий сервіс для генерації підпису директора, прив’язки його до відповідного документу і створення запису в БД.

13) Після успішного підписання документи автоматично надсилаються секретареві вченої ради університету і їм надаєтсья мітка “НАДІСЛАНО”.

14) У разі якщо документ не було підписано директором інституту сформований витяг надсилається до його автора, а мітка документу змінюється

на мітку “ВІДХИЛЕНО”.

15) Кінець алгоритму

6.3.3. Обробка документів на рівні вченої ради університету

Обробка документів на рівні кафедри передбачає врахування операцій асемблювання та реасемблювання, отримання підписів акторів структурного підрозділу третього рівня: секретаря та ректора. Блок-схема алгоритму обробки документів на рівні кафедри є на рис. 6 і містить такі кроки:

1) Початок алгоритму

2) На етапі надсилання підписаного витягу секретареві вченої ради університету (СВРУ), секретар отримує сповіщення про надісланий йому документ. Перш ніж надіслати такі витяги програма перевіряє кожен з них на предмет присутності підпсису завідувача кафедри для кожного з них в БД.

3) Секретар включає надісланий йомудокумент до протоколу засідання вченої ради університету (ПЗВРУ). У той же час файлу включеному в протокол надається помітка “ЗАТВЕРДЖЕНО”. В іншому разі відхилений файл повертається до автора документу з поміткою “ВІДХИЛЕНО”.

4) У разі якщо секретар не включив файл до протоколу засідання кафедри відхилений файл повертається до автора документу з поміткою “ВІДХИЛЕНО”

5) Далі відбувається оффлайн засідання вченої ради університету на якому розглядаються документи включені секретарем до протоколу.

6) На етапі підписання протоколу засідання вченої ради університету (ПЗВРУ) секретар створює його фінальну версію (з рішенням засідання стосовно кожного з документів) використовуючи інтерфейс програми. Для цього секретарю необхідно надати інформацію стосовно дати проведеного засідання, присутніх членів засідання, порядку деннного, матеріали розглянутих питань і рішень винесених по кожному з питань окремо. Після чого надані матеріали будуть автоматично сформовані у кінцеву версію протоколу. Після формування кінцевої версії протоколу, протокол підписується електронно-цифровим підписом (ЕЦП) секретаря, Далі протокол сканується на предмет визначення

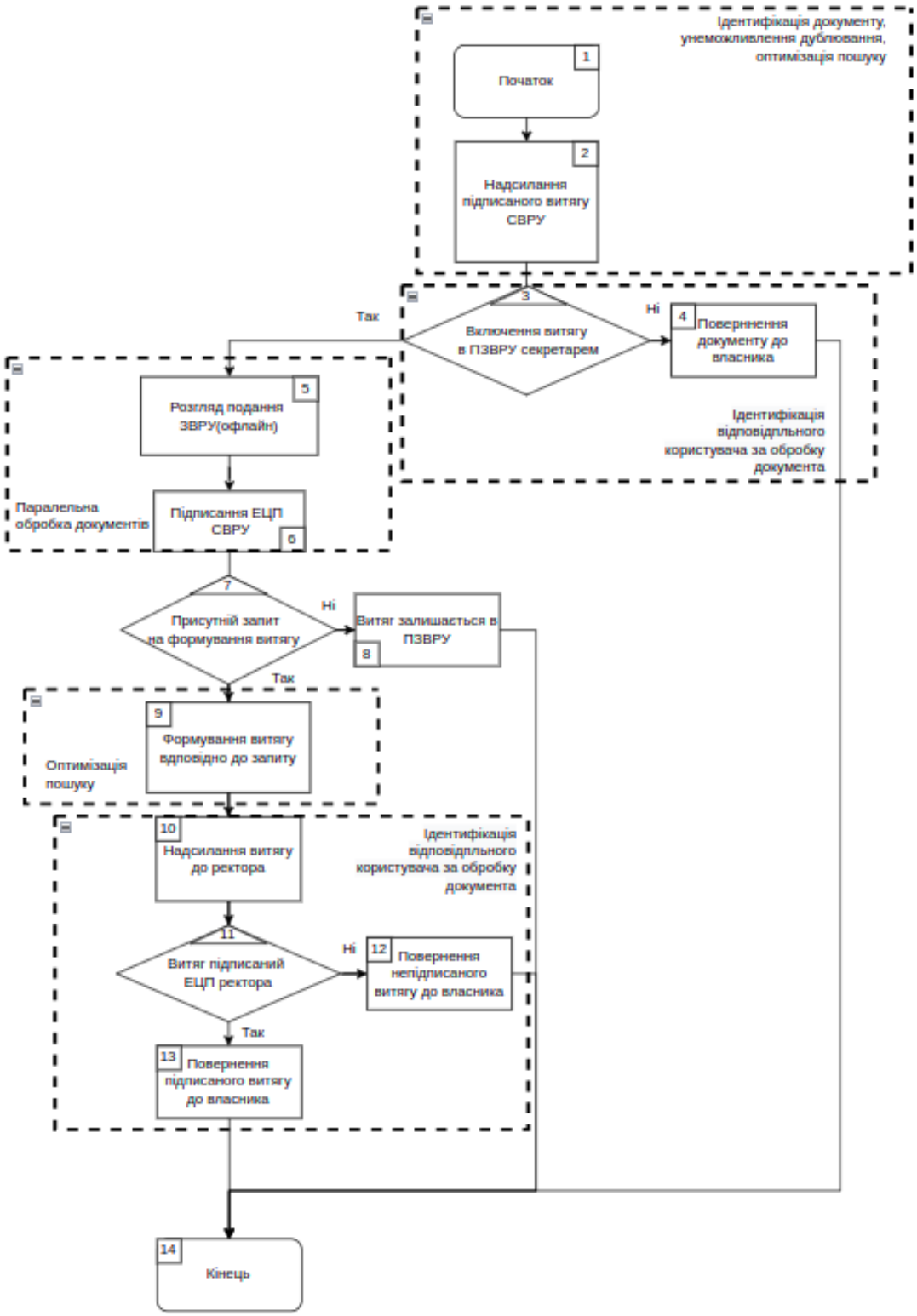


Рисунок 6 – Алгоритм обробки документів на рівні засідання вченої ради університету

Авторська розробка

7) документів необхідних для витягу для кожного із надсилаючих користувачів.

8) Тут система визначає файли, що мають бути сформовані у витяг.

9) У разі якщо надсилаючий користувач не запросив витяг надісланого ним документу – документ залишається в ПЗВРУ з поміткою “ЗАКРИТО”.

10) У разі якщо документ необхідно оформити у витяг з протоколу вибираються всі матеріали, що стосуються даного витягу і формуються в окремий файл. Сформованому файлу надається помітка “ПІДПИСАНО”.

11) Далі всі витяги, що вимагають підпису ректора надсилаються ректорові, про що останній отримує відповідне сповіщення. Перш ніж надіслати такі витяги програма перевіряє кожен з них на предмет присутності підпсису секретаря засідання вченої ради університету для кожного з них в БД.

12) Ректор переглядає документи до підпису і підписує потрібні. На цьому етапі програма викликає окремий сервіс для генерації підпису директора, прив’язки його до відповідного документу і створення запису в БД.

13) Після успішного підписання документи автоматично надсилаються їх авторові, а файлу надається помітка “ЗАКРИТО”.

14) Якщо документ не був підписано ректором, звіт про це надсилається автору, а мітка документу змінюється на мітку “ВІДХИЛЕНО”.

15) Кінець алгоритму

6.4. Ключові механізми САСД та їх взаємодія на рівні клієнт-сервер

Перш ніж деталізувати ключові механізми слід розглянути базові інструменти, що програма використовує для ідентифікації та аутентифікації користувачів та інших об’єктів в системі. Будь яку сутність, що необхідно представити програмно зберігається в БД у вигляді об’єктів (рис. 7). Об’єкти в свою чергу містять набір даних у вигляді полів, які дозволяють дізнатись більше про власне об’єкт.

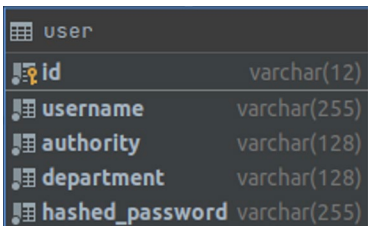


Рисунок 7 – Представлення об'єкту користувача в БД

Авторська розробка

Одним із таких полів є полу ID. Кожен об'єкт на спосіб аутентифікації користувачів в системі. Процес аутентифікації імплементований на базі використання JWT(JSON Web Token) токенів. JWT токен – це стрічка зашифрованих секретом даних, що містять відомості про ID (рис. 8).

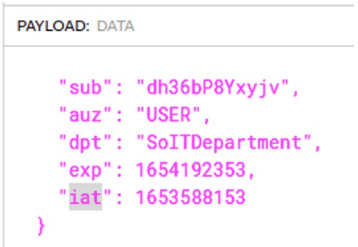
eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJkaDM2Yl
A4WXh5anYiLCJhdXoiOiJVU0VSIiwiZHB0IjoiU
29JVERlcGFydG1lbnQiLCJleHAiOjE2NTQxOTIz
NTMsImlhdCI6MTY1MzU4ODE1M30.1QdPdE5iGlM
SptxTYtNtHsXJnhkgV2NXvedQRqzCgYg

Рисунок 8 – Приклад JWT токену

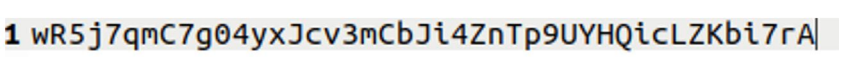
Авторська розробка

Зазвичай JWT токен складається з заголовку (рис. 9, а), вмісту (рис. 9, б) та підпису (рис. 10, в). Заголовок вказує на алгоритм шифрування використаний для формування токену.

а)



б)



в)

Рисунок 9 – Представлення заголовку (а), вмісту (б) та підпису (в) JWT токена

Авторська розробка

Вміст складається з даних, що використовуються для авторизації користувача. Кожна частина даних вмісту ідентифікується власним іменем, які можуть міститися в регістрі тверджень JWT специфікації або бути надані системою, що згенерувала токен. В даному випадку ID користувача зберігається під іменем “sub”, час генерації токена - “iat”, а департамент якому належить користувач – “dpt”.

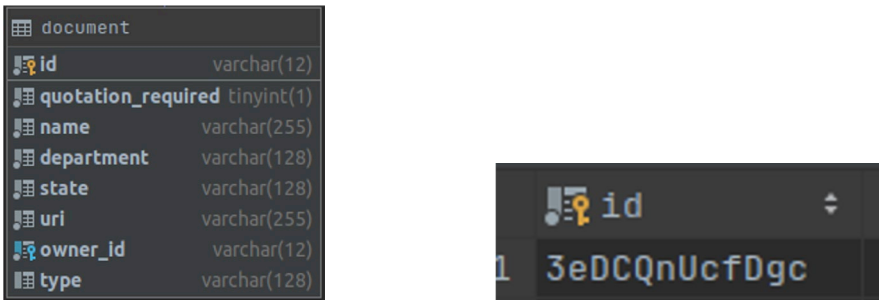
Підпис служить верифікатором токена. Він будується шляхом конкатенації двох попередніх частин через крапку, шифрування алгоритмом Base64 та подальшої передачі побудованої строки до криптографічного алгоритму вказаного в заголовку.

Всі три частини (заголовок, вміст, підпис) у зашифрованому вигляді згодом конкатинуються через крапку формуючи завершений JWT токен.

6.5. Практика відповідності розробки основним принципам ЕСД

6.5.1. Ідентифікація документу

Даний механізм за однократної реєстрації документа дозволяє однозначно ідентифікувати його в будь-якій інсталяції даної системи завдяки парі полів ID та ID користувача власника (рис.10, а).



а) б)

Рисунок 10 – Представлення об’єкту документа (а) в БД та його ID (б)

Авторська розробка

Кожен об'єкт, що вимагає ідентифікації містить уніфіковане поле ID (рис. 10, б), строкового типу, що складається з 12 символів. Цими символами можуть бути літери латинського алфавіту верхнього та нижнього регістрів та цифри.

Під час створення об'єкту система використовує процес аутентифікації для визначення власника документа та генерує унікальне значення ID, яке одразу присвоюється об'єкту документа і згодом зберігається в БД. Дана система використовує єдину базу даних та єдиний процес аутентифікації, що дозволяє користувачеві переглядати документи власником яких він є з будь-яких пристроїв.

6.5.2 Зменшення часу на рух документів

Скорочення часу на обробку документів досягається за допомогою автоматизованого руху документів між департаментами, надсилання документів та витягів секретарям, автоматичного формування витягів.

Система містить список всіх департаментів розташованих в ієрархічному порядку. Об'єкт документу містить поле, що вказує на департамент яким документ обробляється та мітку, що вказує на стан документу (наприклад "ЗАТВЕРДЖЕНО", "ПІДПИСАНО"). Після підписання документу завідувачем кафедри чи директором інституту поле департанту в документі змінюється на департамент слідуючий вище за визначеною ієрархією, а документ надсилається секретареві нового департаменту.

Формування витягу з протоколу ради чи засідання формується шляхом парсингу документу за ключовими словами та помітками в документі. Результатом парсингу є новостворений об'єкт документу витягу, що зберігається в базі даних (рис. 11).

```
public enum Department {
    SoITDepartment( fullName: "Security of Informational Technologies Department", authority: 1),
    ACI( fullName: "Academic Council of Institute", authority: 2),
    ACU( fullName: "Academic Council of University", authority: 3),
```

Рисунок 11 – Список департаментів в програмі

Авторська розробка

6.5.3 Безперервність руху документів

Оптимізація в обробці документів досягається через автоматизацію більшості процесів, які раніше вимагали ручної обробки. Так через автоматизацію передачі документів між департаменту, зникає необхідність для залучення автора документа на всіх рівнях обробки того чи іншого документа. В той же час автор документа може легко визначити на якій стадії обробки знаходиться документ та яким департаментом він обробляється, оскільки об'єкт документу містить поля департаменту і стану, що надають відповідну інформацію. Також система надає можливість авторові документа втрутитись в процес обробки і вилучити документ якщо немає необхідності обробляти його далі.

6.5.4 Унеможливлення дублювання

Запобігання дублюванню імплементоване задля уникнення повторного розгляду однаких документів засіданнями з метою економії часу. Перевірка документів на предмет дублювання відбувається шляхом хешування вмісту кожного з них алгоритмом SHA256. Результатом хешування файлу є унікальна строка, що містить набір, які відповідають документу тільки з таким самим вмістом як і хешований оригінал. Стрічка хешу зберігається системою в базі і є прив'язаною до документа, що використовувався для хешування. Процес хешування відбувається під час створення початкового документу автором та занесенням даних про нього в БД. Якщо документ з рівнозначним хешем в базі даних вже існує система видасть помилку і вкаже на документ дублікатом якого був би новий файл.

6.5.5 Система звітності

Система звітності імплементована для підвищення прозорості в процесі обробки документів та їх подальшої верфікації. Підпис імплементований на базі алгоритму асинхронного шифрування RSA. Кожен користувач має власну унікальний приватний ключ, що формується на базі ID користувача та секретної

строки, яка зберігається на сервері.

Для підписання документу використовується його хеш, що є унікальним для кожного документу за вмістом та приватний ключ підписанта. В результаті підписання документу система отримує строку символів, що потім закріпляється в БД за певним документом. Так після завершення обробки документу певним департаментом витяг матиме прив'язаний підпис секретаря та завідувача чи директора відповідного департаменту. Використовуючи згадані підписи згодом автор може надати підтвердження того, що документ був підписаний відповідними особами. За необхідності будь-який користувач може переглянути список всіх користувачів, що підписали документ.

6.6. Тестування серверного рішення асемблювання та реасемблювання документів у спеціалізованій системі документообігу

Для перевірки коректного функціонування розроблених алгоритмів було складено набір тестів, що сканують програму на предмет можливих вразливостей та непередбачуваної поведінки. Розглянемо тестування даної системи детальніше на прикладі складених інтеграційних та юніт тестів.

6.6.1. Юніт тестування

Юніт тести – клас тестів, що використовуються розробниками в ході написання програмного забезпечення для перевірки коректного функціонування окремих частин програми в ізольованому середовищі.

Ізольоване середовище передбачає використання підроблених даних, що задовольняли б умовам для успішного проходження тестування. Будь яка взаємодія з модулями алгоритму поза зоною тестування чи базою даних у випадку юніт тестування має бути запрограмована для повернення зразків з набору тестових даних заздалегідь визначених програмістом.

Нижче наведено приклад юніт тесту, що перевіряє функцію отримання документів для підписання завідувачем. В прикладі коду можна явно бачити

програмне мімікування взаємодії з БД, що покликане забезпечити ізольованість даного тесту від бази даних (рис. 12).

```
@Test
public void testFindUserDocuments() {
    GenesisUserDetails genesisUserDetails = new GenesisUserDetails();
    genesisUserDetails.setUsername(RandomStringUtils.randomAlphanumeric( count: 12));
    genesisUserDetails.setDepartment(Department.SoITDepartment);
    genesisUserDetails.setAuthority(UserAuthority.HEAD);
    SecurityContextHolder.getContext().setAuthentication(new MockJwtToken(genesisUserDetails));

    Mockito.when(documentRepository.findPendingHeadSignDocuments(Mockito.any(Department.class)))
            .thenReturn(Collections.emptyList());
    Mockito.when(documentConverter.toDto(Mockito.anyList()))
            .thenReturn(Collections.emptyList());

    Assertions.assertTrue(documentService.findPendingHeadSignDocuments().isEmpty());

    Mockito.verify(documentRepository)
            .findPendingHeadSignDocuments(Mockito.any(Department.class));
    Mockito.verify(documentConverter)
            .toDto(Mockito.anyList());
}
```

Рисунок 12 – Приклад юніт тесту САСД

Авторська розробка

6.6.2. Інтеграційне тестування

Інтеграційне тестування – це етап тестування програмного забезпечення, на якому окремі програмні модулі об'єднуються та тестуються як група. Інтеграційне тестування проводиться для оцінки відповідності системи або компонента визначеним функціональним вимогам. Це відбувається після модульного тестування і до тестування системи. Інтеграційне тестування бере як вхідні модулі, які пройшли модульне тестування, групує їх у більші агрегати, застосовує до цих агрегатів тести, визначені в плані інтеграційного тестування, і надає як вихід інтегровану систему, готову до тестування системи.

Нижче наведено приклад інтеграційного тесту, що перевіряє програму на предмет наявності помилок під час запуску програми цілком.

В прикладі коду з рисунка 13 можна явно бачити, що даний тест не містить мімікування взаємодій між модулями власне програми.

```java
package com.genesis;

import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest
public class GenesisAppTest {
    @Test
    public void testIntegration() {
        System.out.println("This is an integration test");
    }
}
```

Рисунок 13 – Приклад інтеграційного тесту САСД

Авторська розробка

Висновки

Представлена робота використовувала системний підхід до вирішення задачі асемблювання та реасемблювання документів у спеціалізованій системі документообігу. Зокрема, основна увагу приділена опису та реалізації алгоритму спеціалізованої автоматизованої системи документообігу на стороні сервера та опис обробки документів на рівні кожного з департаментів. Також розглянуто ключові механізми САСД, що дозволяють забезпечувати формування надійних підписів для документів та уникати дублювання документів, зокрема формування JWT токенів для ідентифікації користувачів в програмі та хешування документів за допомогою алгоритму SHA256.