МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХЕРСОНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
Факультет комп'ютерних наук, фізики та математики
Кафедра комп'ютерних наук та програмної інженерії

ПРОЕКТУВАННЯ ТА РОЗРОБКА КЛІЄНТ-ЧАСТИНИ ДЛЯ СЕРВІСУ ДОКУМЕНТООБІГУ

Кваліфікаційна робота (проєкт)
на здобуття ступеня вищої освіти «бакалавр»

Виконав: здобувач
спеціальності: 121 Інженерія програмного забезпечення Освітньо-професійної (наукової) програми: Інженерія програмного забезпечення
Гусаченко Сергій Андрійович
Керівник: кандидат фізико-математичних наук, професор Співаковський Олександр Володимирович
Рецензент: Морозов В'ячеслав Євгенійович Tech Lead Forex Tester Software

Херсон – Івано-Франківськ – 2023

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

HTTP - HyperText Transfer Protocol

HTML - HyperText Markup Language

CSS - Cascading Style Sheets

JSON - JavaScript Object Notation

JWT - JSON Web Token

UI - User Interface

ВСТУП

Актуальність теми: За останні кілька років, з введенням електронного документообігу, компанії все більше переходять на цифрові формати документів, щоб підвищити ефективність та скоротити час на обробку документів.

У зв'язку з цим, популярність сервісів документообігу росте, що в свою чергу вимагає розробки клієнт частини для зручного доступу до сервісу. Крім того, віддалена робота стала нормою у багатьох компаніях, що збільшує потребу у цифровій обробці документів та доступі до них віддалено.

Тому, розробка клієнт частини для сервісу документообігу допоможе компаніям збільшити продуктивність та зменшити час на обробку документів. Крім того, це також забезпечує зручний доступ до документів, що є особливо важливим для віддалених робітників. Тому, можна стверджувати, що тема "проектування та розробка клієнт частини для сервісу документообігу" є актуальною в сучасному світі та має великий потенціал для успішного використання в бізнесі.

***Об'єкт дослідження*:** проєктування та розробка клієнт-частини сервісу документообігу, що дозволяє оптимізувати процес обміну документами та покращити продуктивність користувачів.

***Предмет дослідження*:** технології проєктування та розроблення клієнт-частини системи.

Мета дослідження: проєктування та розроблення клієнт частини сервісу.

Завдання дослідження:

1. Аналіз вимог до функціональності та інтерфейсу користувача для клієнт-частини сервісу документообігу.

2. Визначення ефективної архітектури та структури клієнт-частини сервісу документообігу.

3. Розробка клієнт-частини сервісу документообігу з використанням технологій

***Практичне значення одержаних результатів*:** Отримані результати мають велике практичне значення для покращення ефективності та продуктивності документообігу в компанії. Розроблена клієнт-частина сервісу документообігу забезпечує зручний та швидкий доступ до документів, управління їх статусами та редагування. Застосування сучасних технологій, таких як React, MobX, TypeScript, Sass, HTML, CSS, дозволяє зменшити час на розробку та підтримку програмного забезпечення.

Крім того, результати дослідження можуть бути використані для подальшого покращення процесу документообігу, у випадку, якщо розроблене програмне забезпечення буде впроваджено в компанії. Програмний продукт може бути доповнений функціоналом, який дозволить більш ефективно взаємодіяти з документами та використовувати їх у робочих процесах.

Апробація одержаних результатів: Результати наукового дослідження будуть представлені на науковій міжнародній конференції ICTERI 2023.

РОЗДІЛ 1
АНАЛІЗ ВИМОГ, РОЗГЛЯД ТА ВИБІР ТЕХНОЛОГІЙ.

1.1 Аналіз вимог та аналогів

Перед тим, як розпочати процес розробки продукту, необхідно провести аналіз аналогів, оскільки розробка сервісів є складним технологічним процесом, який потребує уважного планування та виконання. Для проведення аналізу були обрані додатки, які найбільше відповідають задуму нашого сервісу.

- PandaDoc - цей сервіс дозволяє користувачам створювати шаблони документів та використовувати їх для автоматичного створення нових документів. PandaDoc надає безліч інструментів для редагування документів та має простий інтерфейс.
  Основні функції:
  - Створення, редагування та підписання документів в електронному форматі.
  - Можливість відправляти документи на підпис користувачам по електронній пошті.
  - Контроль та відстеження стану документів. https://www.pandadoc.com/
- DocuSign - цей сервіс надає можливість створення та підписування документів з використанням шаблонів. DocuSign має безліч інструментів для підписування документів та є досить простим у використанні.
  Основні функції:
  - Створення професійних документів за допомогою шаблонів та редактора.
  - Підписання документів в електронному форматі.

- Можливість відправляти документи на підпис користувачам по електронній пошті.
- Онлайн-платформа для спільної роботи над документами та їх коментування. https://www.docusign.com/

- SignNow - цей сервіс надає можливість створювати та редагувати документи з використанням шаблонів. SignNow має досить простий інтерфейс та можливість використання підписів та штампів.

  Основні функції:

  - Створення та збереження шаблонів документів.
  - Автоматичне створення документів на основі шаблонів.
  - Можливість налаштування шаблонів та прав доступу користувачів до них. https://www.signnow.com

Але ці сервіси мають свої недоліки і саме їх ми будемо намагатись уникнути:

1. Високі вартості: багато сервісів мають високі вартості для користувачів, особливо для невеликих компаній, що може бути недоцільним.
2. Обмеження для редагування документів: деякі сервіси можуть мати обмежені можливості для редагування документів або взагалі не дозволяти редагування PDF-документів.
3. Недостатня підтримка мобільних пристроїв: деякі сервіси можуть мати обмежені можливості для використання підписів на мобільних пристроях.

Базуючись на перевагах та недоліках аналогів ми склали вимоги до продукту. Сервіс документообігу з можливістю створювати документи з шаблонів повинен задовольняти наступні вимоги:

1. Наявність шаблонів. Для користувачів повинен бути доступний набір готових шаблонів документів, які можуть бути використані для створення нових документів. Ці шаблони повинні бути відповідним чином організовані, щоб користувачі могли швидко знайти потрібний шаблон.

2. Створення власних шаблонів. Користувачі повинні мати можливість створювати власні шаблони документів з відповідними полями та параметрами. Це дозволить користувачам зберігати свої налаштування та зручно створювати документи, що відповідають їх потребам.

3. Інтуїтивний інтерфейс. Сервіс повинен мати простий інтерфейс, що дозволяє користувачам швидко створювати нові документи з використанням шаблонів. Для цього повинні бути доступні такі функції, як пошук, фільтрація, перегляд інформації про шаблон, вибір параметрів та інше.

4. Безпека даних користувача.

5. Адаптивність дизайну до різних екранів користувачів. Це забезпечує зручне використання сервісу на майже будь-якому пристрої.

1.2 Вибір технологій та інструментів для розробки клієнт-частини сервісу

У розділі вибору технологій та інструментів для розробки клієнт-частини сервісу документообігу, було проаналізовано різні технології та

інструменти, які можуть бути використані. Основними критеріями вибору були: продуктивність, масштабованість, зручність розробки, ступінь складності та зручність використання для користувачів.

Так як це буде саме веб-додаток, то тут є декілька лідируючі інструменти в даному напрямку це: React, Angular Vue. Ми зробили порівняльну таблиці цих фреймворків (таблиця 1).

Параметр	**React**	**Angular**	**Vue**
Популярність та спільнота	Висока	Висока	Висока
Продуктивність	Висока	Середня	Висока
Складність навчання	Низька	Висока	Середня
Компонентна структура	Дуже добра	Добра	Дуже добра
Масштабованість	Дуже добра	Дуже добра	Дуже добра
Функціональність	Дуже добра	Добра	Добра
Параметр	React	Angular	Vue

Таблиця 1

Загалом, React володіє перевагами у швидкості, легкості та гнучкості, в той час як Angular відповідає за широкі можливості для тестування та масштабування. Vue вирізняється своєю простотою та високою продуктивністю. Крім того, React є найбільш популярним серед цих фреймворків, що забезпечує широку підтримку та активну спільноту.

Щоб доповнити фреймворк було обрано наступні інстумент:

- React Router — це легка, повнофункціональна бібліотека маршрутизації для бібліотеки React [12]. За допомогою неї можливо реалізувати навігацію по веб-додатку.
- MobX - це стейт-менеджер, який дозволяє легко і ефективно управляти станом додатка [7]. Він використовується для збереження та управління даних користувача.
- Mobx Persist Store - простий спосіб зберегти та відновити спостережувані властивості в сховищах mobx [8]. Вона дозволяє зберігати стан додатка в локальному сховищі браузера, так щоб при наступному запуску додатка стан залишився незмінним.
- TypeScript - це надмножина JavaScript, яка надає статичну типізацію та інші корисні функції для розробки програмного забезпечення [15]. Використання TypeScript дозволить більш легко маштабувати додаток.
- Sass - це препроцесор CSS, який надає додаткові функції та можливості для розробки стилів [13]. Використання Sass дозволить забезпечити більш гнучкий та ефективний підхід до розробки стилів.
- HTML та CSS - це основні технології для розробки веб-сторінок [**Ошибка! Источник ссылки не найден.**]. Використання HTML та CSS дозволить забезпечити створення нашого користувацького інтерфейсу.
- Axios - це клієнт HTTP на основі Promise для node.js та браузера [2]. Ця бібліотека надає можливості для здійснення запитів на сервер і обробки відповідей.
- Figma - це інструмент для розробки макетів та прототипів інтерфейсу користувача [3]. Використання Figma дозволить нам візуалізувати наші ідеї та забезпечити чітке розуміння того, яким має бути наш інтерфейс.

- Fontello - це інструмент для створення та налаштування іконок [4]. Використання Fontello дозволить надавати нашому додатку унікальний стиль та забезпечити наявність необхідних іконок.
- i18n - це бібліотека для локалізації додатків [18]. Використання i18n дозволить нам створити більш доступний та користувацький інтерфейс для нашої аудиторії, яка має різні мовні потреби

Ці технології та інструменти були вибрані на основі їх ефективності, доступності та популярності в розробницькій спільноті. Використання цих технологій дозволить нам забезпечити швидку та ефективну розробку клієнт-частини нашого сервісу, забезпечити його високу продуктивність та доступність для користувачів.

РОЗДІЛ 2
ПОБУДОВА АРХІТЕКТУРИ, РОЗРОБКА ТА ОПИС ФУНКЦІОНАЛЬНОСТІ

2.1 Проєктування взаємодії клієнт-частини та серверу

В даній науковій роботі розглядається реалізація онлайн-сервісу який має свої механіки, одною із яких є отримання даних про документ, який розмістив інший користувач і для цього застосовано дволанкову клієнт-серверну архітектуру.

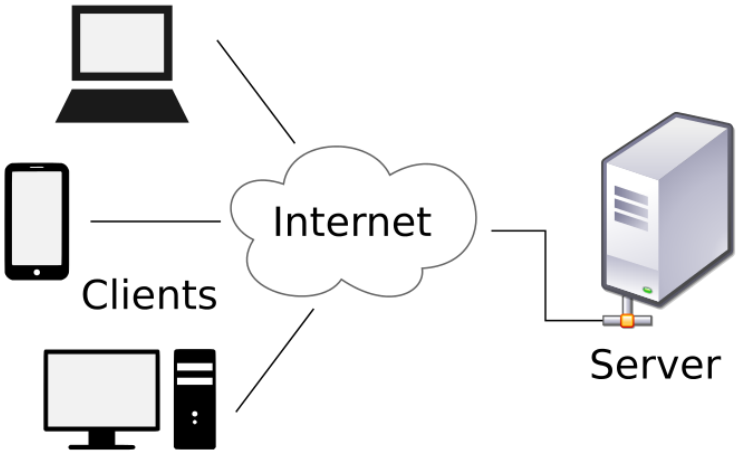

Рисунок 2.1 - Клієнт-серверна архітектура

Дволанкова клієнт-серверна архітектура передбачає взаємодію двох програмних модулів — клієнтського та серверного. В залежності від того, як між ними розподіляються наведені вище функції, розрізняють:

- модель тонкого клієнта, в рамках якої вся логіка застосунку та управління даними зосереджена на сервері. Клієнтська програма забезпечує тільки функції рівня представлення;
- модель товстого клієнта, в якій сервер тільки керує даними, а обробка інформації та інтерфейс користувача зосереджені на стороні клієнта. Товстими клієнтами часто також називають пристрої з

обмеженою потужністю: кишенькові комп'ютери, мобільні телефони та ін.[21].

В сервісі реалізовано саме модель товстого клієнта.

Для захищеного зв'язку с серверною частиною було розглянуто та застосовано JWT.

JWT — це відкритий стандарт, який використовується для обміну інформацією безпеки між двома сторонами — клієнтом і сервером. Кожен JWT містить закодовані об'єкти JSON, включаючи набір вимог. JWT підписуються за допомогою криптографічного алгоритму, щоб гарантувати, що вимоги не можуть бути змінені після видачі токена[9].

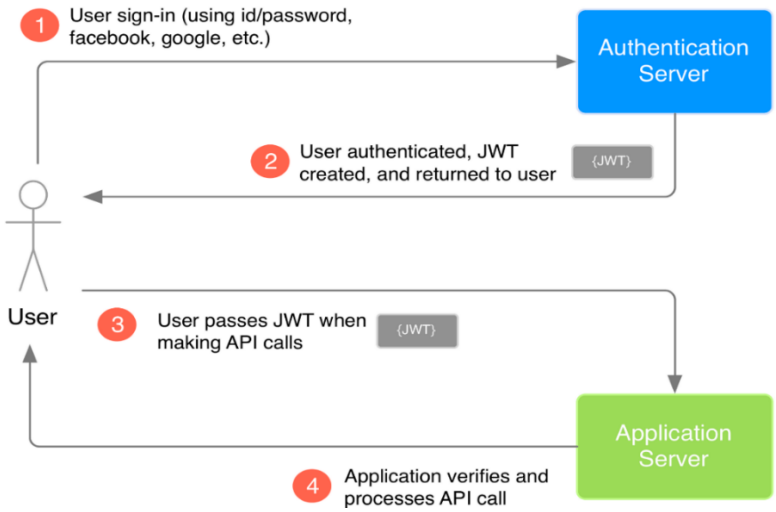

Рисунок 2.2 - Схема роботи JWT

2.2 Проєктування архітектури інтерфейсу

Також однією з вимог до до веб-сервісу є адаптивність.

Адаптивний вебдизайн - дизайн вебсторінок, що забезпечує оптимальне відображення та взаємодію сайту з користувачем незалежно від роздільної здатності та формату пристрою, з якого здійснюється перегляд сторінки[19].

Метою адаптивного вебдизайну є практичне відображення інформації та зручна навігація на всіх пристроях із доступом до інтернету (від стаціонарних ПК до мобільних телефонів). За технологією адаптивного вебдизайну не потрібно створювати окремі версії вебсайту. Один сайт може працювати на всьому спектрі пристроїв.

У розробці адаптивного дизайну існує два підходи до побудови адаптивного веб дизайну. Розробка починається або з мобільного інтерфейсу, а далі відбувається адаптація інтерфейсу для інших розширень (англ. mobile first), або, навпаки, розробка починається від вигляду на екранах стаціонарних ПК і закінчується інтерфейсом на мобільних телефонах (англ. desktop first).

Рисунок 2.3 - Desktop first (зліва) та mobile first (справа)

Враховуючи те, що сервіс документообігу буде веб-додатком, було прийняте рішення будувати інтерфейс спочатку під зручне використання на ПК. Оскільки користувачі, які використовують девайси з більшим екраном, можуть бути більш зацікавлені в використанні сервісу на ПК, ніж на мобільному пристрої.

Для проєктування інтерфейсу було застосовано Figma. З допомогою цієї програми можна створювати високоякісні макети для веб-сайтів, мобільних додатків та інших інтерфейсів. Також сервіс забезпечує можливість зберігати та ділитися проектами в обласному сховищі, що полегшує співпрацю та комунікацію між учасниками проекту.

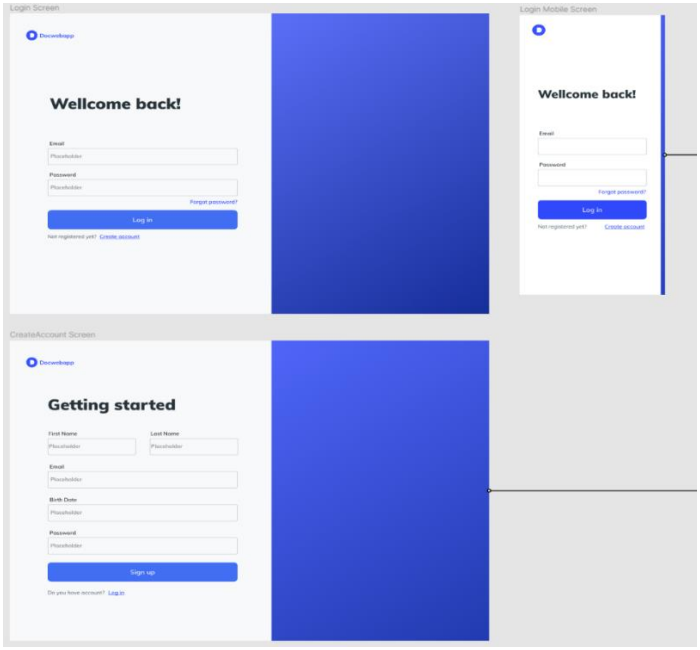

Рисунок 2.4 - Приклад розробленого дизайну аутентифікації для клієнт-частини

2.3 Діаграми прецедентів та діяльності

Діаграми прецедентів є важливим інструментом для визначення вимог користувачів до системи та її функціональності. Вони дозволяють проектній команді зрозуміти, які операції будуть виконуватись системою та як користувачі будуть з нею взаємодіяти.

Для створення діаграм прецедентів ми використовували програму Lucid. При розробці діаграм було враховано вимоги користувачів, описані в розділі 1.1.

Проаналізувавши роботу інших систем документообігу, ми визначили набір прецедентів для нашої системи. Основними прецедентами є:

- Створення робочого простору
- Створення робочого простору
- Додавання та управління користувачами в робочому просторі

- Сторення та управління документами
- Підписання документів користувачем
- Запит підпису у користувача в робочому просторі
- Сторення та управління шаблонами документів
- Створення документу за шаблоном

Діаграми прецедентів були побудовані на основі цих прецедентів, враховуючи взаємодію користувачів з системою та її функціональність.

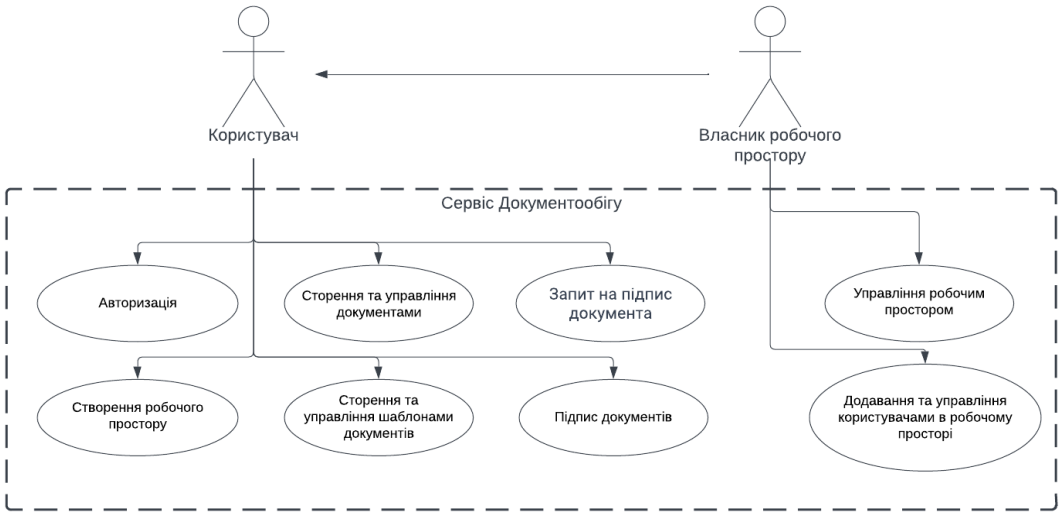

Рисунок 2.5 - Діаграма прецендентів

Також для відображення певних бізнес процесів ми побудували діаграму діяльності. Діаграма діяльності — це діаграма поведінки UML, яка показує потік керування або потік об’єкта з акцентом на послідовність і умови потоку [16]. Вона допомагає зобразити послідовність дій, які потрібно зробити користувачу, щоб завершити певну задачу. Наприклад діаграма яка відображає процес створення документа користувачем (Рисунок 2.3.2)

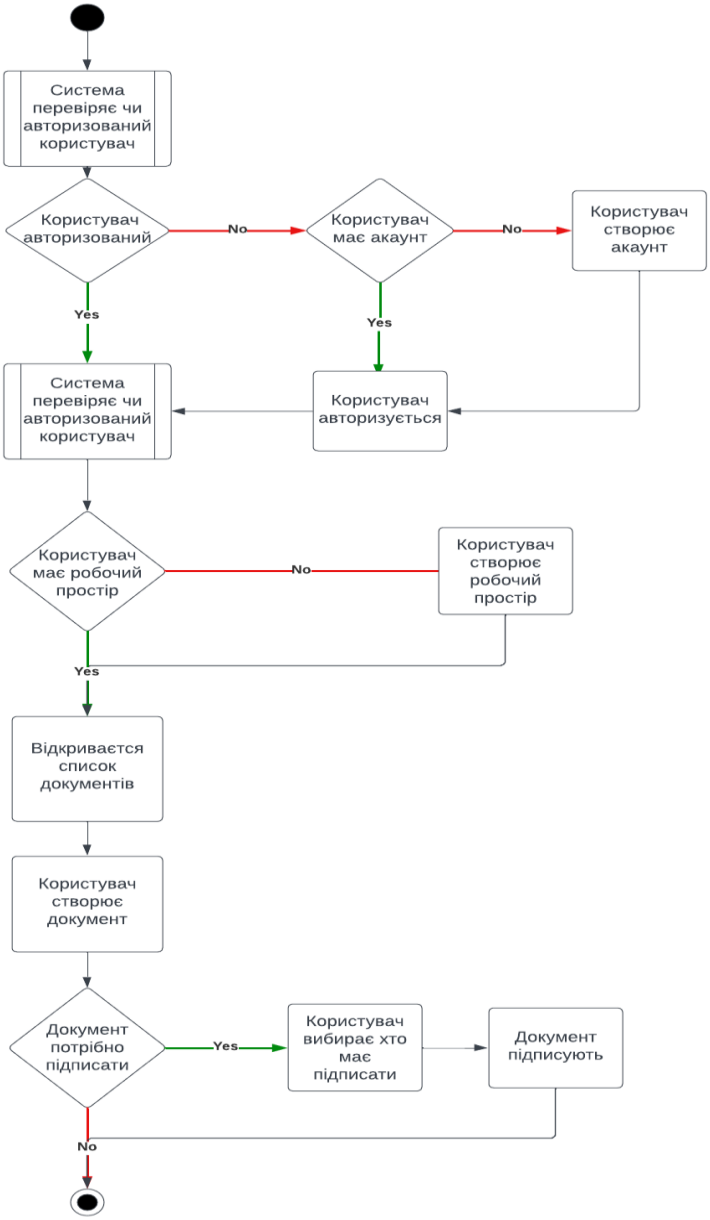

Рисунок 2.6 - Діаграма діяльності зі створення документа

РОЗДІЛ 3
РОЗРОБКА ТА ОПИС ФУНКЦІОНАЛЬНОСТІ КЛІЄНТ-ЧАСТИНИ

3.1 Розробка стуктури клієнт-частини

Структуру клієнт-частину сервісу можна розділити на декілька частин публічної та приватної.

Публічна частина (public) - це та частина сервісу, яка доступна користувачам без необхідності увійти в систему. Сюди входять такі екрани, як:

- Головна сторінка (Landing) - перший екран, який дозволяє користувачеві зрозуміти, які можливості надає сервіс і як він може бути корисним.
- Сторінка авторизації (Auth) - це екран, на якому користувач може ввійти до свого облікового запису, якщо він уже зареєстрований, або зареєструватися в системі, якщо він новий користувач.

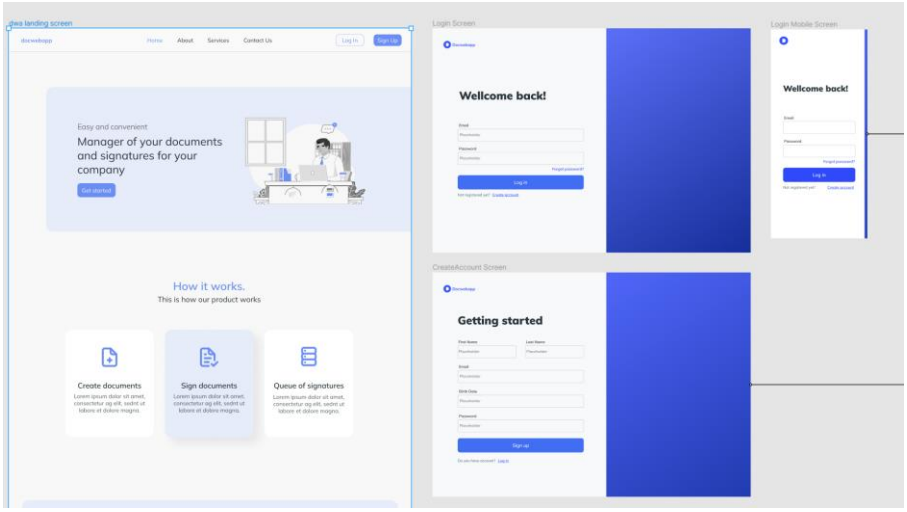

Рисунок 3.1 - Публічна частина частина

Приватна частина (private) - частина сервісу яка доступна користувачу якій ввійшов до системи. Сюди входять такі екрани, як наприклад:

- Сторінка вибору створення робочого середовища – яка дозволяє або приєднатись до вже існуючого, або створити своє.
- Сторінка списоку документів (Documents) – це екран, який дозволяє відслідковувати документи та їх статуси доступні користувачеві.
- Сторінка списку співробітників (People) – це екран, на якому користувач може побачити список співробітників в його середовищі.
- Сторінка з налаштуваннями (Settings) – екран, який дозволяє змінювати певні налаштування відображення сервісу для користувача.

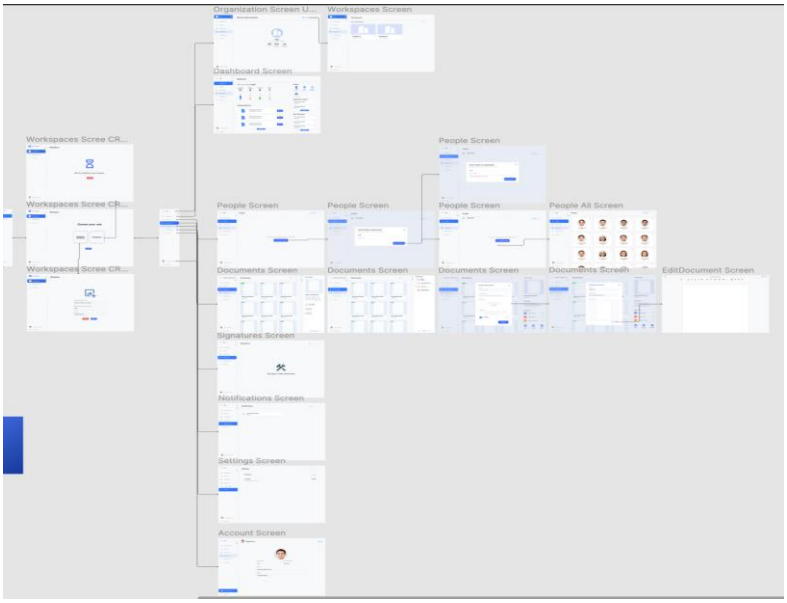

Рисунок 3.2 - Приватна частина сервісу

3.2 Розробка файлової структури клієнт-частини

Структура папок у клієнтській частині сервісу організована за функціональними категоріями. Вони розбиті так, щоб розробник мав

легкий доступ до певних файлів та міг легко зрозуміти де ці файли розташовані.

Рисунок 3.3 - Файлова структура клієнт-частини

Папка Assets

Зберігає в собі файли, які не треба завантажувати з серверної частини такі як:

- Шрифти для тексту, та конвертовані в html та css іконки.
- Зображення які використовуються на сторінках.
- Іконки – для покращення взаємодії з користувачем.
- Локалізацію
- Анімації, які збереженні в певному форматі та використовуються за допомогою Lottie.
- Стилі – глобальні scss стилі.

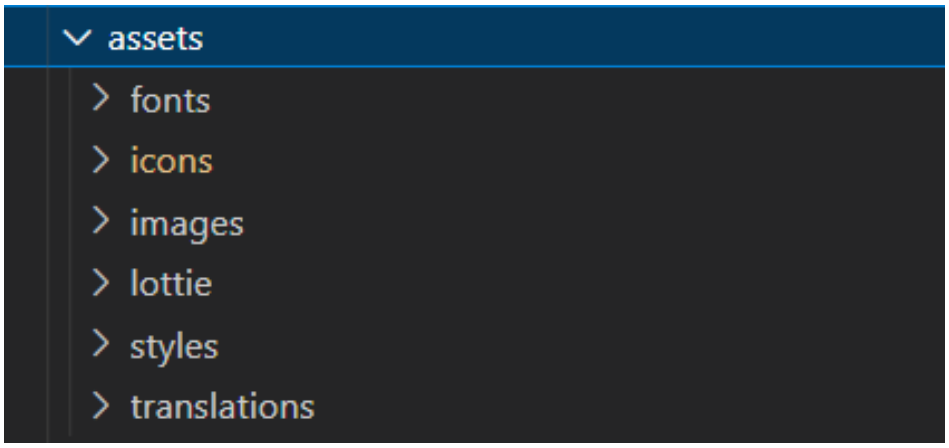

Рисунок 3.4 - Файлова структура папки Assets

Папка Common

Створенний для структурування глобальних методів або певних данних та має в собі такі підрозділи як:

- Constants - зберігає в собі певні сталі данні.
- Hooks – містить певні React hooks які можна використовувати глобально в компонетах.
- Requests – функції зі запитами на сервер.
- Services – обгортки над сторонніми сервісами або певні класи.
- Types – глобальні типи.
- Utils – функцію які можна використовувати глобально.

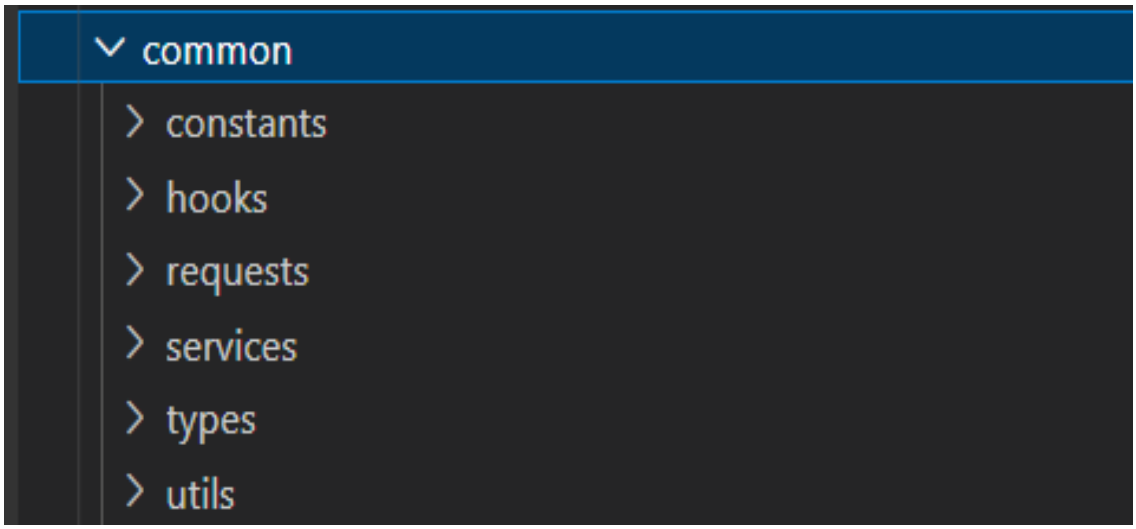

Рисунок 3.5 - Файлова структура папки Common

Папка Mobx

Зберігає в собі модулі з певною бізнес логікою та сховищем. Кожен модуль відповідає за свій розділ наприклад:

- Модуль Auth – відповідає на авторизацію та реєстрацію користувача.
- Модуль User – відповідає за збереження стану та даних про користувача.
- Модуль Documents – відповідає за отримання, додавання, збереження та редагування документів.
- Модуль App – відповідає за збереження налаштувань та глобальних даних сервісу.
- Модуль Workspace – відповідає за отримання даних про робоче середовище та його створення або зміну, також там зберігаються данні про робітників в цьому середовищі та функції їх додавання або видалення.

Папка View

Відповідає за UI компоненту клієнт-частини. В ній зберігаються всі React компоненти, які взаємодіють з користувачем. Вона складається з таких розділів:

- Components – відповідає за універсальні компоненти, які можна перевикористовувати в різних контейнерах, або сторінках.
- Containers – мітить в собі певні UI-елементи, які мають певну логіку й складаються з певних компонентів.
- Routers – відповідає за навігацію по сервісу, та містить в собі розділення сервісу на дві частини, приватну та публічну.
- Screens – тут знаходяться всі сторінки, які доступні користувачеві.

- Окремий файл RootView - файл, який об’єднує всі вище перераховані розділи, та імпортується в головний файл.

Рисунок 3.6 - Файлова структура папки View

Інші файли

Також є інші файли без яких робота клієнт-частини була б неможливою.

- App.tsx - цей файл відповідає за ініціювання певних функцій при старті, обгортання контенту в MobX Provider та рендеру UI.
- Index.tsx – імпортує стилі перед ініціюванням App.tsx та запускає його рендер.

3.3 Розробка навігації по клієнт-частині сервісу

Розробка навігації по клієнт-частині сервісу передбачає створення системи переходів між різними сторінками та компонентами, яка б дозволяла користувачам легко переміщатися по функціоналу сервісу.

Один з популярних підходів до розробки навігації - використання роутера. У клієнт додатку, так як він працюює на платформі React, для цього використовується бібліотека React Router. Вона дозволяє описувати шляхи (routes) до різних сторінок та компонентів, а також налаштовувати параметри роутів (наприклад, передавати параметри в URL).

Наприклад, ось реалізація розділення додатку на дві частини, приватну та публічну.

```
export const MainRouter = observer(({ element: Component, ...rest }: any) => {
  const loginStatus: boolean = !userStore.user;

  // Redirect to other screen if the condition is not true
  return (
    <Routes>
      <Route path="*" element={loginStatus ? <PublicRouter /> : <PrivateRouter />} />
    </Routes>
  );
});
```

Рисунок 3.7 - Реалізація розділення на дві частини

За допомогою цього, ми можемо відокремити авторизованого користувача від публічної складової сервісу та перенаправляти його на приватну, в разі спробі переходу, наприклад, на сторінку авторизації. І навпаки - неавторизованого користувача, ми будемо перенаправляти на сторінку авторизації, або на домашню сторінку. А визначити, що користувач авторизований, ми можемо завдяки MobX, який при оновленні даних передасть в компонент актуальну інформацію про користувача.

Також було передбачено, що користувач може ввести не коректну URL адресу і в цьому разі сервіс буде перенаправляти його на спеціальну сторінку, в якій буде повідомлятись, що ця URL адреса не коректна.

3.4 Опис функціональності клієнт-частини системи документообігу

При першому запуску, якщо користувач не авторизований, на екрані відображається домашня сторінка, на якій відображається інформація про сервіс та його головні можливості, також є кнопки для переходу на екран авторизації.

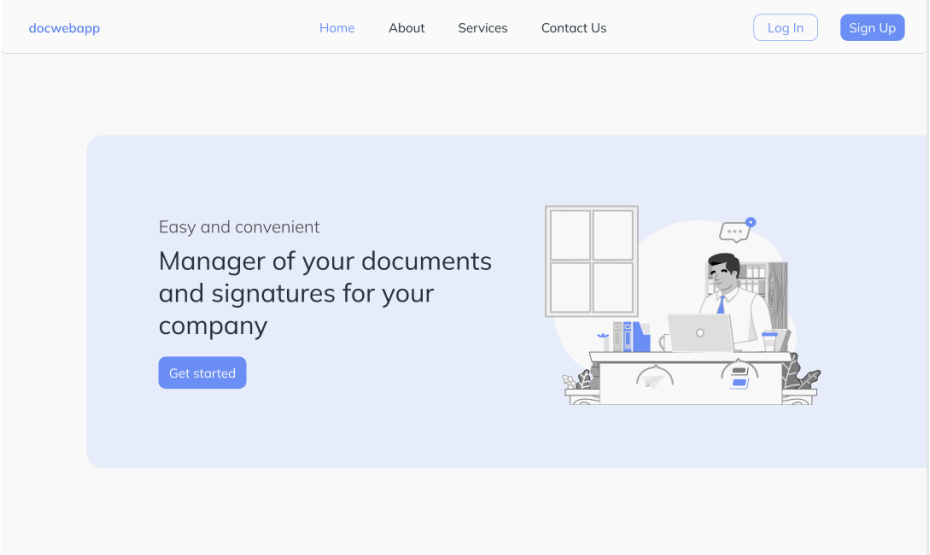

Рисунок 3.8 - Початкова сторінка

Після натискання кнопки «Log in» або «Sign up «користувач переходить на екран авторизації. На цьому екрані він має можливість, або ввійти в свій аккаунт, використовуючи свій пароль та логін, або перейти на екран реєстрації та створити новий.

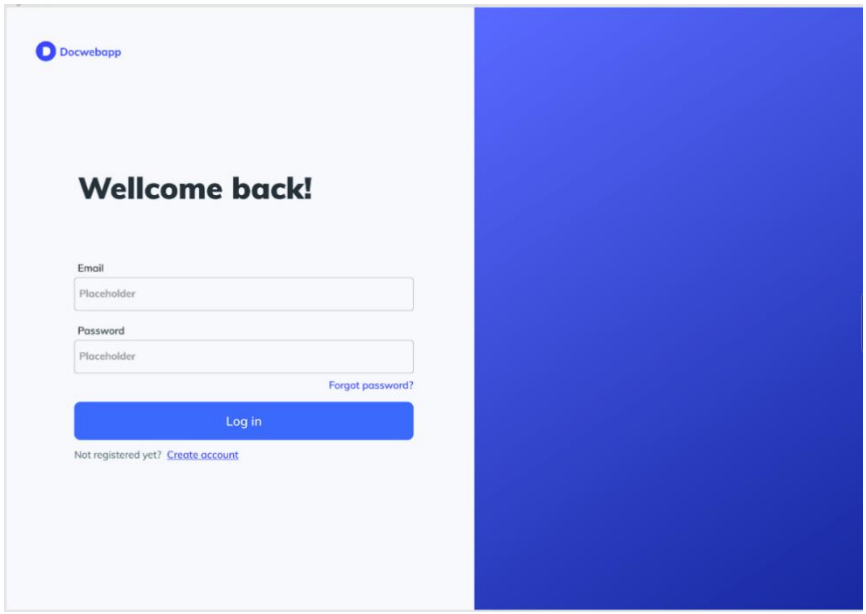

Рисунок 3.9 - Екран авторизації

Далі після успішної реєстрації/авторизації користувач потрапляє в приватну частину сервісу та відкривається екран робочого простору (Workspace). Якщо користувач ще не має створеного робочого простору,

або його не додали до існуючого, він може його створити. Також зліва можна побачити бокову панель, на якій відображено головні вкладки, які дають змогу переміщатись по певним розділам, або вийти з аккаунту.

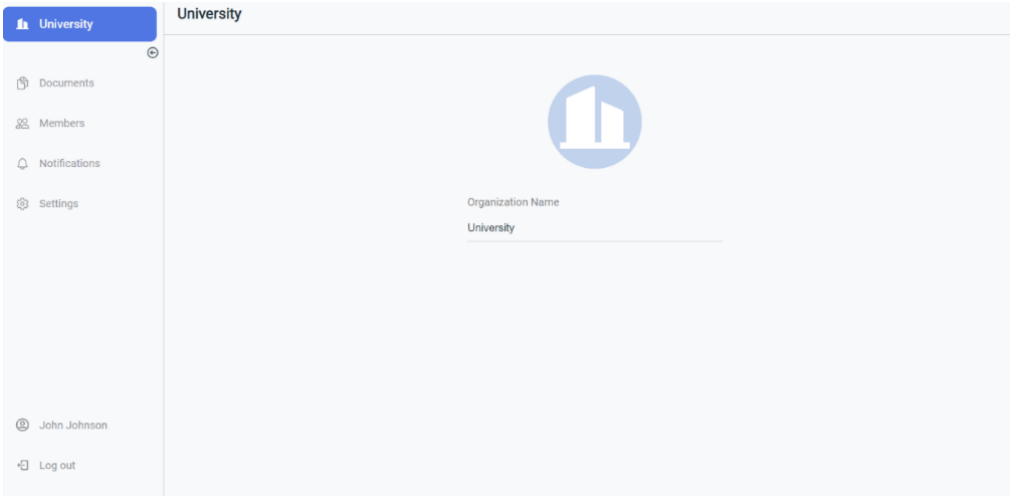

Рисунок 3.10 - Робочій простір

При натисканні на вкладку «Документи», користувача буде перенаправлено на сторінку з списком документів, доступних на цьому робочому просторі. На цій сторінці користувач має змогу бачити всі доступні йому документи, або додати новий, чи створити з доступного шаблону. До документу можна додати назву та опис, також є можливість додати підписи на цей документ, та запросити підписи у інших користувачів у вашому робочому просторі.

Після успішного створення документу, користувач може побачити цей документ у його робочому просторі. Також користувач може відслідковувати прогрес підписання документу.

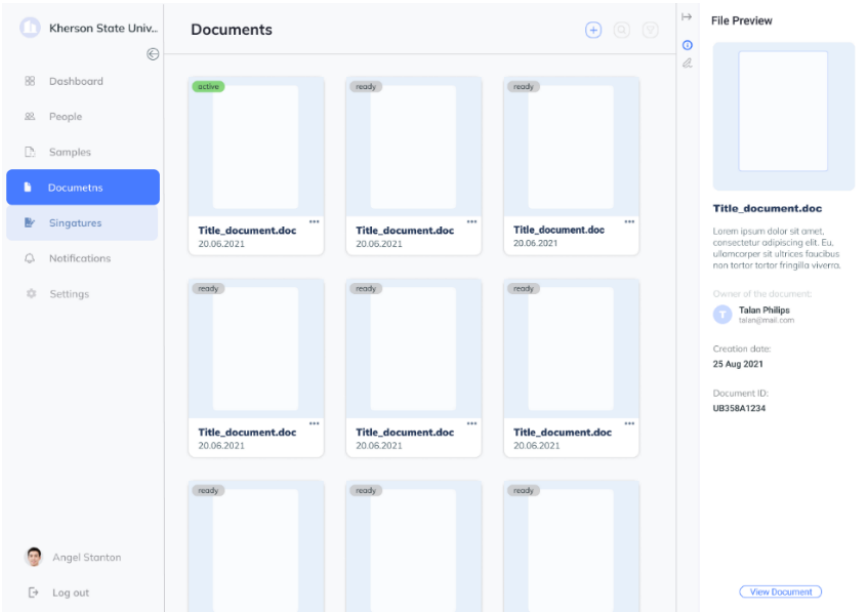

Рисунок 3.11 - Сторінка документів

Також було створено нову сторінку для управління збережених шаблонів документів. Щоб перейти на цю сторінку треба натисути на вкладку «Шаблони». На ній можна побачити вже існуючи шаблони, керувати ними, або перейти до додавання нового шаблону.

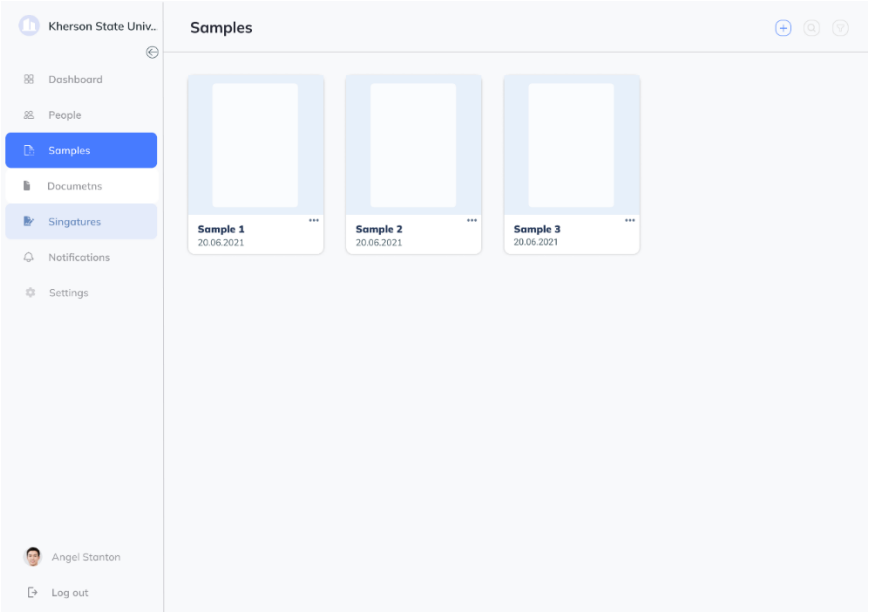

Рисунок 3.12 - Екран зі створеними шаблонами документів

На першому етапі створення шаблону треба ввести назву нового шаблону та завантажити документ який буде слугувати шаблоном.

Документ обов’язково повинен бути зі спеціальними спец-ключами для розпізнавання полів зі змінними даними.

По бажанню можна додати опис нового шаблону.

На другому етапі створення шаблону, за допомогою спеціального парсера, сервіс знаходить спеціальні ключі, а користувачу треба визначити дня них назву та тип даних.

Для легшого створення, користувач має змогу переглянути доданий документ та при необхідності повернутись до першого етапу створення.

Якщо документ шаблону було змінено, то парсер повторно просканує файл для знаходження спеціальних ключів, але при цьому заповненні поля будуть очищені автоматично.

Якщо все правильно і всі необхідні поля заповнено, то користувач може перейти до останнього етапу.

На третьому етапі, користувач може перевірити створений ним шаблон та при необхідності повернутись на необхідний етап для редагування шаблону.

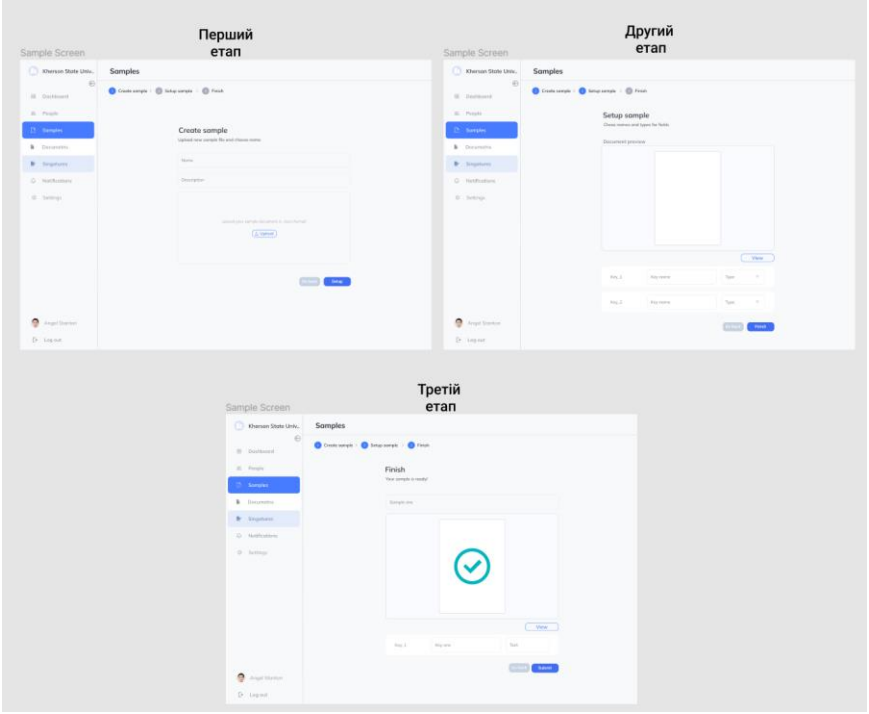

Рисунок 3.13 - Екран з етапами створення шаблону

ВИСНОВКИ

У ході наукової роботи була поставлена мета спроектувати та розробити сервіс документообігу.

Досягнення зазначеної мети здійснено шляхом вирішення таких основних завдань:

1. Аналіз вимог до функціональності та інтерфейсу користувача для клієнт-частини сервісу документообігу.
2. Визначення ефективної архітектури та структури клієнт-частини сервісу документообігу.
3. Розробка клієнт-частини сервісу документообігу з використанням технологій

Сервіс має перспективи подальшого розвитку.

Наприклад:

- Планується додати управління структурою компанії
- Розширити типи підтримуваних документів
- Додати можливість посилатися на інші документи та співробітників

Тож у майбутньому планується розробляти проект дотримуючись раніше встановлених версій, поступово впроваджуючи нові модулі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Angular [Електронний ресурс]. URL: https://angular.io/ (дата звернення: 03.04.2023).
2. Axios: [Електронний ресурс]. URL: https://axios-http.com/uk/docs/intro (дата звернення: 03.04.2023).JWT.IO. JSON Web Tokens - jwt.io. [Електронний ресурс]: https://jwt.io/
3. Figma: [Електронний ресурс]. URL: https://www.figma.com/ (дата звернення: 03.04.2023).
4. Fontello: [Електронний ресурс]. URL: https://fontello.com/ (дата звернення: 03.04.2023).
5. HTML та CSS: [Електронний ресурс]. URL: https://www.w3.org/ (дата звернення: 03.04.2023).
6. Lucidchart [Електронний ресурс]. URL: https://lucid.app/ (дата звернення: 03.04.2023).
7. MobX: [Електронний ресурс]. URL: https://mobx.js.org/README.html (дата звернення: 03.04.2023).
8. mobx-persist-store", [Електронний ресурс]. URL: (https://www.npmjs.com/package/mobx-persist-store) (дата звернення: 03.04.2023).
9. React vs Angular vs Vue.js: The Complete Comparison? [Електронний ресурс]. URL: https://scopicsoftware.com/blog/react-vs-angular-vs-vue-js-the-complete-comparison/
10. React vs Angular vs Vue: Which Framework to Choose? [Електронний ресурс]. URL: https://altar.io/resources/react-vs-angular-vs-vue/
11. React – JavaScript-бібліотека для створення користувацьких інтерфейсів. React – JavaScript-бібліотека для створення користувацьких інтерфейсів. [Електронний ресурс]: https://uk.reactjs.org/ (дата звернення: 30.05.2022).
12. react-router [Електронний ресурс]. URL: https://github.com/remix-run/react-router (дата звернення: 03.04.2023).
13. Sass: [Електронний ресурс]. URL: https://sass-lang.com/ (дата звернення: 03.04.2023).
14. ThinkMobiles. React vs Angular vs Vue: What to Choose for Your Next Project? [Електронний ресурс]. URL: https://thinkmobiles.com/blog/react-vs-angular-vs-vue/ (дата звернення: 03.04.2023).
15. TypeScript: [Електронний ресурс]. URL: https://www.typescriptlang.org/ (дата звернення: 03.04.2023).

16. UML Activity Diagrams, UML Diagrams [Електронний ресурс]. URL: https://www.uml-diagrams.org/activity-diagrams.html (дата звернення: 03.04.2023).
17. Vue.js [Електронний ресурс]. URL: https://vuejs.org/ (дата звернення: 03.04.2023).
18. i18n: [Електронний ресурс]. URL: https://github.com/i18next/i18next (дата звернення: 03.04.2023).
19. Адаптивний вебдизайн – Вікіпедія. Вікіпедія. [Електронний ресурс]: https://uk.wikipedia.org/wiki/Адаптивний_вебдизайн (дата звернення: 30.05.2022).
20. Вимоги до програмного забезпечення [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/%D0%92%D0%B8%D0%BC%D0%BE%D0%B3%D0%B8_%D0%B4%D0%BE_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F
21. Клієнт-серверна архітектура та ролі серверів. [Електронний ресурс] – https://medium.com/@IvanZmerzlyi/%D0%BA%D0%BB%D1%96%D1%94%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80%D0%BD%D0%B0-%D0%B0%D1%80%D1%85%D1%96%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0-%D1%82%D0%B0-%D1%80%D0%BE%D0%BB%D1%96-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80%D1%96%D0%B2-9893d8048229).

ДОДАТКИ

Додаток А

КОДЕКС АКАДЕМІЧНОЇ ДОБРОЧЕСНОСТІ
ЗДОБУВАЧА ВИЩОЇ ОСВІТИ
ХЕРСОНСЬКОГО ДЕРЖАВНОГО УНІВЕРСИТЕТУ

Я, Гусаченко Сергій Андрійович, учасник освітнього процесу Херсонського державного університету, **УСВІДОМЛЮЮ**, що академічна доброчесність – це фундаментальна етична цінність усієї академічної спільноти світу.

ЗАЯВЛЯЮ, що у своїй освітній і науковій діяльності **ЗОБОВ'ЯЗУЮСЯ**:

– дотримуватися:

· вимог законодавства України та внутрішніх нормативних документів університету, зокрема Статуту Університету;

· принципів та правил академічної доброчесності;

· нульової толерантності до академічного плагіату;

· моральних норм та правил етичної поведінки;

· толерантного ставлення до інших;

· дотримуватися високого рівня культури спілкування;

– надавати згоду на:

· безпосередню перевірку курсових, кваліфікаційних робіт тощо на ознаки наявності академічного плагіату за допомогою спеціалізованих програмних продуктів;

· оброблення, збереження й розміщення кваліфікаційних робіт у відкритому доступі в інституційному репозитарії;

· використання робіт для перевірки на ознаки наявності академічного плагіату в інших роботах виключно з метою виявлення можливих ознак академічного плагіату;

– самостійно виконувати навчальні завдання, завдання поточного й підсумкового контролю результатів навчання;

– надавати достовірну інформацію щодо результатів власної навчальної (наукової, творчої) діяльності, використаних методик досліджень та джерел інформації;

– не використовувати результати досліджень інших авторів без використання покликань на їхню роботу;

– своєю діяльністю сприяти збереженню та примноженню традицій університету, формуванню його позитивного іміджу;

– не чинити правопорушень і не сприяти їхньому скоєнню іншими особами;

– підтримувати атмосферу довіри, взаємної відповідальності та співпраці в освітньому середовищі;

– поважати честь, гідність та особисту недоторканність особи, незважаючи на її стать, вік, матеріальний стан, соціальне становище, расову належність, релігійні й політичні переконання;

– не дискримінувати людей на підставі академічного статусу, а також за національною, расовою, статевою чи іншою належністю;

– відповідально ставитися до своїх обов’язків, вчасно та сумлінно виконувати необхідні навчальні та науково-дослідницькі завдання;

– запобігати виникненню у своїй діяльності конфлікту інтересів, зокрема не використовувати службових і родинних зв’язків з метою отримання нечесної переваги в навчальній, науковій і трудовій діяльності;

– не брати участі у будь-якій діяльності, пов’язаній із обманом, нечесністю, списуванням, фабрикацією;

– не підроблювати документи;

– не поширювати неправдиву та компрометуючу інформацію про інших здобувачів вищої освіти, викладачів і співробітників;

– не отримувати і не пропонувати винагород за несправедливе отримання будь-яких переваг або здійснення впливу на зміну отриманої академічної оцінки;

– не залякувати й не проявляти агресії та насильства проти інших, сексуальні домагання;

– не завдавати шкоди матеріальним цінностям, матеріально-технічній базі університету та особистій власності інших студентів та/або працівників;

– не використовувати без дозволу ректорату (деканату) символіки університету в заходах, не пов’язаних з діяльністю університету;

– не здійснювати і не заохочувати будь-яких спроб, спрямованих на те, щоб за допомогою нечесних і негідних методів досягати власних корисних цілей;

– не завдавати загрози власному здоров’ю або безпеці іншим студентам та/або працівникам.

УСВІДОМЛЮЮ, що відповідно до чинного законодавства у разі недотримання Кодексу академічної доброчесності буду нести академічну та/або інші види відповідальності й проти мене можуть бути застосовані заходи дисциплінарного характеру за порушення принципів академічної доброчесності.

12.09.2019 р.		Гусаченко Сергій
(дата)	(підпис)	(ім’я, прізвище)