**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**
**ІМЕНІ ТАРАСА ШЕВЧЕНКА**

Факультет комп'ютерних наук та кібернетики

Кафедра теорії та технології програмування

# Кваліфікаційна робота

**на здобуття ступеня бакалавра**

за спеціальністю 122 Комп'ютерні науки

на тему:

## ВЕБ-ІНТЕРФЕЙС ДЛЯ ВЗАЄМОДІЇ ТА КОНТРОЛЮ СИСТЕМИ ДОКУМЕНТООБІГУ

Виконала студентка 4-го курсу
Лісова Марія Юріївна

(підпис)

Науковий керівник:
кандидат фіз.-мат. наук
Федорова Марія Вікторівна

(підпис)

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент

(підпис)

Роботу розглянуто й допущено до захисту на засіданні кафедри теорії та технології програмування
«1» червня 2022 р.,
протокол № 10
Завідувач кафедри
М. С. Нікітченко

(підпис)

Київ – 2022

# РЕФЕРАТ

Обсяг роботи 42 сторінки, 39 ілюстрацій, 1 таблиця, 20 джерел посилань, 1 додаток.

ВЕБ-ІНТЕРФЕЙС, ВЗАЄМОДІЯ, ОДНОСТОРІНКОВА ПРОГРАМА, СИСТЕМА ДОКУМЕНТООБІГУ, ANGULAR, DEVEXTREME, “MNEME”, “MNEMEWEBUI”.

Об’єктом розробки є система документообігу, що має назву “Mneme” (грецькою “пам’ять”). Предметом розробки є веб-інтерфейс “MnemeWebUI” для взаємодії та контролю з системою документообігу. Метою роботи є створення візуальної оболонки до вже існуючої системи документообігу. Методи розроблення: методи створення веб-інтерфейсів, технології розробки односторінкових програм, методологія XP для організації робочого процесу.

Інструменти розроблення: Microsoft Visual Studio Professional 2022, сервіси Azure DevOps, мова програмування сервера C#, фреймворк Serilog для логування сервера. Мова програмування клієнта TypeScript. Для візуалізації використані HTML, CSS та бібліотека DevExtreme від компанії DevExpress.

Результати роботи: проведено порівняння найпопулярніших технологій розробки SPA, досліджено ефективність взаємодії Angular та DevExtreme, розроблено застосунок “MnemeWebUI”, що слугує веб-інтерфейсом для системи документообігу та дозволяє взаємодіяти та контролювати роботу системи.

Робота дуже тісно пов’язана з системою документообігу, до якої, власне, і створювався веб-інтерфейс. Програма постійно підтримує зв’язок з системою, надсилаючи запити і перевіряючи серцебиття.

Програма вже впроваджена в роботу поки що невеликої кількості співробітників Quipu GmbH і вже використовується за призначенням. Результати роботи також можна використовувати у якості такого посібника.

# ЗМІСТ

# СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

| | |
|---|---|
| Quipu GmbH | німецька компанія, що займається розробкою десктопних програм для банкінгу. Компанія-замовник представленого проекту |
| Система<br>система документообігу<br>система Mneme<br>Mneme | власна розробка компанії Quipu для збереження та взаємодії з документами |
| Серцебиття/Heart Beat | функція системи, що показує чи має на даний момент наш додаток зв’язок з нею |
| Теги | специфікований тип запитів, що приймає система для пошуку документів |
| Prefix, Owner, Value | складові частини тега, що мають бути заповнені для коректної відповіді системи на пошуковий запит |
| SPA<br>односторінкова програма | веб-додаток, що складається з однієї сторінки, що частково перезавантажується для відображення нової інформації |
| Сторінка | в нашій роботі це логічні блоки, на які поділена програма для кращого сприйняття та зручності |
| Логування | ведення журналу роботи програми |
| Api request | запит до системи “Mneme” |

# ВСТУП

**Оцінка сучасного стану об’єкта розробки**. Сьогодні як в Україні, так і в усьому світі існує і широко використовується банківська система, що забезпечує кругообіг платіжних засобів доступний всім і кожному. Банківською системою зараз так чи інакше користується будь-який пересічний громадянин; чи то студент, що отримує стипендію на банківську картку, чи то заощадливий програміст має депозит, чи навіть бабуся, що отримує пенсійні виплати на картку.

Перевагами добре розвиненої банківської системи користуються всі, але мало хто знає, що відбувається “по той бік барикад”. В цей час співробітники банків та розробники платіжних систем проводять кропітку працю над забезпеченням безперебійної роботи цієї величезної машини.

Одним з аспектів такої розробки є взаємодія з величезним обсягом особистої інформації користувачів банків, а саме обробка та збереження скан-копій документів, що засвідчують особу, підтверджують платоспроможність тощо. Тож розробка швидкісних та зручних застосунків для збереження, редагування та перегляду цих документів є одним з найважливіших завдань розробників банківських систем.

**Актуальність роботи та підстави для її виконання**. Зазвичай системи взаємодії з величезним обсягом даних є набором зв’язаних між собою файлів, що добре вміють обробляти різноманітні запити, але, нажаль, майже ніколи не мають візуального оформлення.

Німецька компанія Quipu GmbH, що надає комплексні рішення для банків і фінансових установ, також пропонує своє рішення даної задачі – швидкісну та зручну у використанні систему збереження документів.[1] Проте, як і більшість таких систем, вона розроблялась без зручного користувацького інтерфейсу.

З часом кількість оброблюваних даних системою зросла, відповідно збільшилась кількість помилкових записів, редагувань тощо. Тож виникла гостра

потреба у створенні візуальної оболонки системи задля більш швидкого реагування у критичній ситуації та зручного контролю її функціонування.

**Мета й завдання роботи**. Саме тому метою роботи стало створення такої візуальної оболонки до вже існуючої системи документообігу. Для досягнення поставленої мети було сформульовано наступні завдання.

- Порівняти платформи розробки та дизайну односторінкових програм.
- Дослідити обрану платформу на ефективність взаємодії з продукцією компанії DevExpress.
- Розробити веб-інтерфейс на основі обраних технологій.
- Здійснити апробацію роботи в компанії Quipu GmbH.

**Об'єкт, методи й засоби розроблення**. Об’єктом розробки є взаємодія та контроль системи документообігу, що має символічну назву “Mneme” (скорочено від Mnemosyne – у давньогрецькій міфології богиня пам’яті)[2]. Тож розроблюваний інтерфейс отримав відповідну назву “MnemeWebUI”.

Протягом розробки програми загалом використовувалась досить відома методологія екстремального програмування (XP від англ. Extreme Programing)[3]. У ситуації, що наразі склалась в Україні, ми навчились швидко реагувати на нагальні потреби та спокійно ставитись до будь-яких змін, саме тому часті консультації з кінцевим клієнтом та не менш часті демо-презентації стали частиною розробки.

Інструментом розробки стали інтегроване середовище розробки програмного забезпечення Microsoft Visual Studio Professional 2022 та сервіси Azure DevOps. Мова програмування серверної частини проекту – C#. Для логування робочого процесу серверу було підключено фреймворк Serilog. Мова програмування клієнта – TypeScript. Для візуалізації використані мова гіпертекстової розмітки HTML, каскадна таблиця стилів CSS та колекцію компонентів інтерфейсу від компанії DevExpress – DevExtreme. Поєднання вищенаведених інструментів та засобів розробки дає можливість створити високопродуктивний та адаптивний клієнт-серверний програмний продукт.

**Можливі сфери застосування**. Програма “MnemeWebUI” розроблена з метою застосування її для взаємодії та контролю системи документообігу “Mneme”. Проте результати роботи можна розглядати зі сторони дослідження взаємодії Angular з продукцією компанії DevExpress та використовувати у якості такого собі посібника.

**Взаємозв’язок з іншими роботами**. Робота дуже тісно пов’язана з системою документообігу, до якої, власне, і створювався інтерфейс. Візуальна оболонка постійно підтримує зв’язок з системою, надсилаючи запити і перевіряючи серцебиття.

**Впровадження роботи.** Програма вже впроваджена в роботу поки що невеликої кількості співробітників Quipu GmbH і вже використовується за призначенням. Наразі робота ще має недоліки, невиконані та непоставлені задачі, які надихають рухатись далі і задають вектор для подальшого розвитку проекту.

# РОЗДІЛ 1 ОГЛЯД ВИКОРИСТАНИХ ТЕХНОЛОГІЙ

## 1.1 Загальний огляд технологій

Як вже було зазначено інструментом розробки стали інтегроване середовище розробки програмного забезпечення Microsoft Visual Studio Professional 2022 та сервіси Azure DevOps.

Microsoft Visual Studio Professional 2022 – це найкраще комплексне середовище IDE для розробників .NET і C++ у Windows. Повноцінний набір інструментів та функцій для покращення та вдосконалення кожного етапу розробки програмного забезпечення.[5] Зручне, знайоме багатьом середовище розробки, яке нічим не поступається іншим IDE, крім того має безліч переваг для розробників, особливо в комплектації Professional В якості та зручності розробки конкурує хіба що із ще одним продуктом від Microsoft – Microsoft Visual Studio Code. Проте, незважаючи на свою громіздкість, Microsoft VS все одно виграє в якості, зручності та кількості додаткових інструментів розробки.

Azure DevOps надає послуги, які дозволяють командам планувати роботу, співпрацювати над розробкою коду, а також створювати й розгортати програми. Azure DevOps дозволяє організаціям створювати та вдосконалювати продукти швидшими темпами, ніж за допомогою традиційних підходів до розробки програмного забезпечення.[6]

Серверна частина проекту була написана мовою C# на базі ASP.NET Core Web API.

Наш веб-інтерфейс потребував серверну частину здебільшого саме для запуску клієнта, тому технологія для сервера обиралась за принципом продуктивності та перевіреності на власному досвіді. Тож оскільки, як заявляє Microsoft: “Продуктивність є ключовим напрямком ASP.NET Core.” – і цей програмний продукт перевірений часом, ми обрали саме його.

Крім того, в подальшому ми розраховуємо на постійне розширення додатку, в тому числі із залученням серверної частини не тільки в якості стартового

механізму, а і в якості обробника даних, безпосередньо. З огляду на те, що подібні завдання вже обговорюються, до коду серверної частини було також підключено фреймворк Serilog.

Serilog – це новітня платформа ведення журналів для .NET, що була створена для забезпечення структурованого логування.[7] Для нас у виборі логеру одним з найважливіших аспектів була можливість вести журнал Application Insights (Application Insights – це функція Azure, яка забезпечує розширене керування продуктивністю додатків та моніторинг для активних веб-програм.[8]). Крім того, Serilog надає можливості для запису подій журналу в сховище в близько 100 форматах.[9]

Після порівняння існуючих можливостей розробки на мові TypeScript (порівняння дивіться розділ 1.2), для клієнтської частини було обрано Angular проект.

TypeScript –це строго типізована мова програмування, яка базується на JavaScript, що дає вам кращі інструменти в будь-якому масштабі.[10]

Angular – це платформа дизайну додатків і розробки для створення ефективних і витончених односторінкових програм.[11]

Для створення візуально-приємного інтерфейсу були використані широковідомі HTML та CSS, а також менш відома, але не менш цікава колекція компонентів DevExtreme від компанії DevExpress.

DevExpress (Developer Express) – компанія з розробки програмного забезпечення, що базується в США. Займається розробкою інструментів та компонентів допомоги в кодуванні для розробників Delphi, C++ Builder і Microsoft Visual Studio.[13] Компанія в основному займається підтримкою десктопних застосунків, для мови C# це здебільшого WinForms та WPF застосунки. Власне, через те, що Quipu вже багато років успішно користується підтримкою DevExpress в розробці програм на базі WPF, була поставлена задача дослідити та використати їх відносно нову бібліотеку DevExtreme.

“Світ non-.NET розробки все ще має велике значення. DevExtreme – наша особиста бібліотека HTML/JavaScript – продовжує розвиватися, і ми вкладаємо

більше часу та зусиль у підтримку React, Angular та Vue, а також працюємо над кращою інтеграцією SCSS та TypeScript.”[14]

Більш детальне дослідження можливостей DevExtreme див. розділ 1.3.

## 1.2 Порівняння технологій розробки SPA

Такі схожі, але водночас такі різні Angular, React та Vue, їх об’єднує спільна мета, але різнить структура, синтаксис та поріг входження.

Angular – фронтенд-фреймворк з відкритим кодом, написаний на TypeScript, який розробляють Google, під керівництвом Angular Team.[14]

Vue.js – JavaScript-фреймворк, який у 2014 році створив Еван Ю, колишній член команди Angular у Google.[15]

React – це JavaScript-бібліотека з відкритим вихідним кодом, створена Джорданом Валке з Facebook.[16]

В таблиці 1.2.1 більш детально розглянуті та порівняні можливості кожного з них.

Таблиця 1.2.1 – Порівняння платформ розробки SPA

| Платформа / Властивість | Angular | React | Vue.js |
|---|---|---|---|
| Має підтримку корпорації | ✓ | ✓ | ✗ |
| Повноцінний фреймворк | ✓ | ✗ | ✓ |
| Розробка великих проектів | ✓ | ✗ | ✗ |
| Зручна структура | ✓ | ✗ | ✓ |
| Проста структура | ✗ | ✓ | ✓ |

| | | | |
|---|---|---|---|
| Створення SPA | ✓ | ✓ | ✓ |
| Швидкість | ✓ | ✓ | ✗ |
| Невеликий розмір розробленої програми | ✗ | ✗ | ✓ |
| Низький поріг входження | ✗ | ✓ | ✓ |
| Зручність масштабування | ✓ | ✗ | ✗ |
| Всього: | 7✓ | 5✓ | 6✓ |

За результатами порівняльної таблиці 1 можемо бачити, що Angular найбільше відповідає нашим потребам. Крім того, оскільки додаток створювався для користування та подальшого розвитку поміж співробітників Quipu, треба врахувати більш розповсюджений та бажаний стек технологій серед них. Таким виявився Angular.

Зважаючи на вищенаведені аргументи, було прийняте рішення обрати Angular, як основну платформу для створення клієнтської частини проекту.

### 1.3 Ефективність поєднання Angular та DevExtreme

DevExtreme представляє собою повну колекцію високопродуктивних компонентів інтерфейсу користувача для використання в традиційних односторінкових веб-додатках, а також в мобільних додатках наступного покоління. Серед набору компонентів доступні зручні багатофункціональні таблиці даних, інтерактивні діаграми, редактори даних, інтерактивні карти та багато іншого.

Засоби створення інтерфейсів DevExtreme мають можливість співпрацювати з будь-яким із трьох найпопулярніших засобів створення односторінкових

додатків, а саме з Angular, React та Vue. Оскільки платформою розробки нашої програми було обрано Angular, тому пропоную дослідити взаємодію саме DevExtreme та Angular.

Напевне, потрібно почати з того, що бібліотека DevExtreme, як і всі продукти DevExpress, має офіційну документацію. Проте, документація саме цього набору інструментів, на мою думку є найбільш зручною. Підтвердженням вищенаведеного служить те, що, потрапивши на головну сторінку документації за посиланням [17], ми в один-два кліки можемо перейти до створення свого першого застосунку з DevExtreme компонентою (рисунок 1.3.1) або ж одразу до перегляду демо-версій всіх доступних елементів розробки (рисунок 1.3.2).

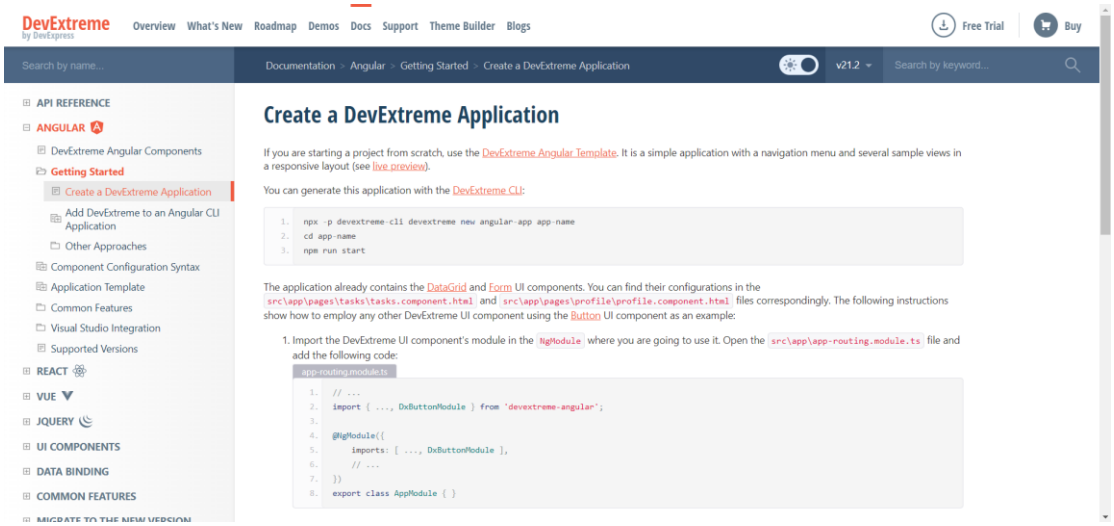



Рисунок 1.3.1 – Створення DevExtreme застосунку

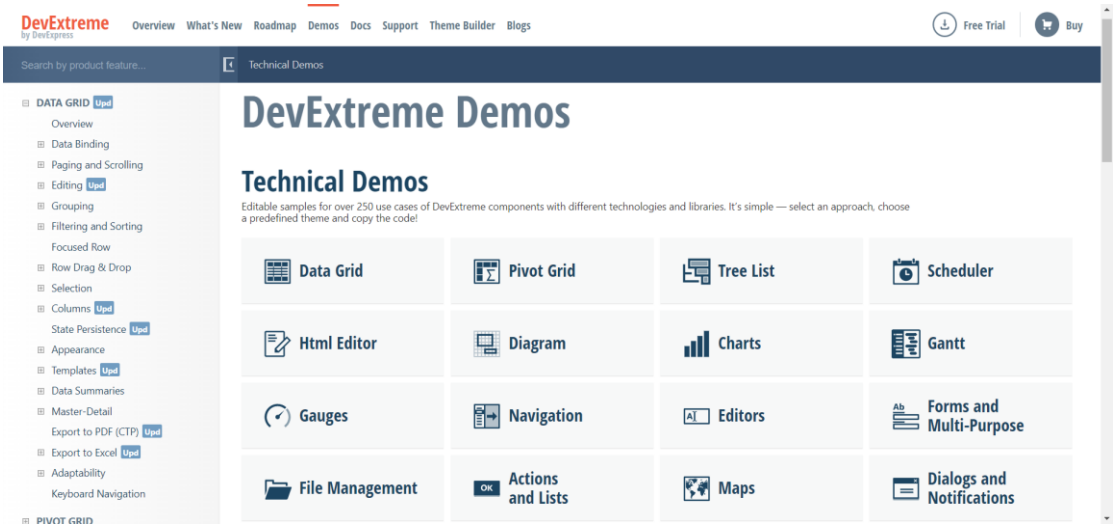



Рисунок 1.3.2 – Меню демо-версій DevExtreme компонент

Наш проект не буде базуватися на DevExtreme, а тільки використовувати його як засіб візуалізації даних. Саме тому ми детальніше розглянемо сторінку демо-версій засобів розробки користувацьких інтерфейсів.

Як ми вже бачили на рисунку 1.3.2, демо поділені на умовні категорії задля забезпечення швидкого пошуку потрібних компонент та елементів розробки. Нижче на тій самій сторінці ми бачимо приклади простих додатків, що були реалізовані розробниками DevExtreme і демонструють найрізноманітніші можливості бібліотеки.

На початковій стадії проекту було заплановано:

- Створення зручної навігації
- Додавання таблиці для перегляду даних
- Візуалізація статистичних даних.

Про них і поговоримо.

Зручна навігація. В меню демо-версій обираємо навігацію, потрапляємо на відповідну сторінку документації. Тут бачимо різні можливості курування програмою починаючи від деревовидної структури та панелі з вкладками, завершуючи акордеоном та так званою шухлядою.

Нам прийшовся до вподоби компонент під назвою шухляда, тож переходимо до нього (рисунок 1.3.3). Знову ж таки, говорячи про якість документації, в представленому демо. Окрім, власне, навігаційної панелі бачимо додаткову панель з опціями, що дає можливість порівняти доступні можливості роботи панелі.

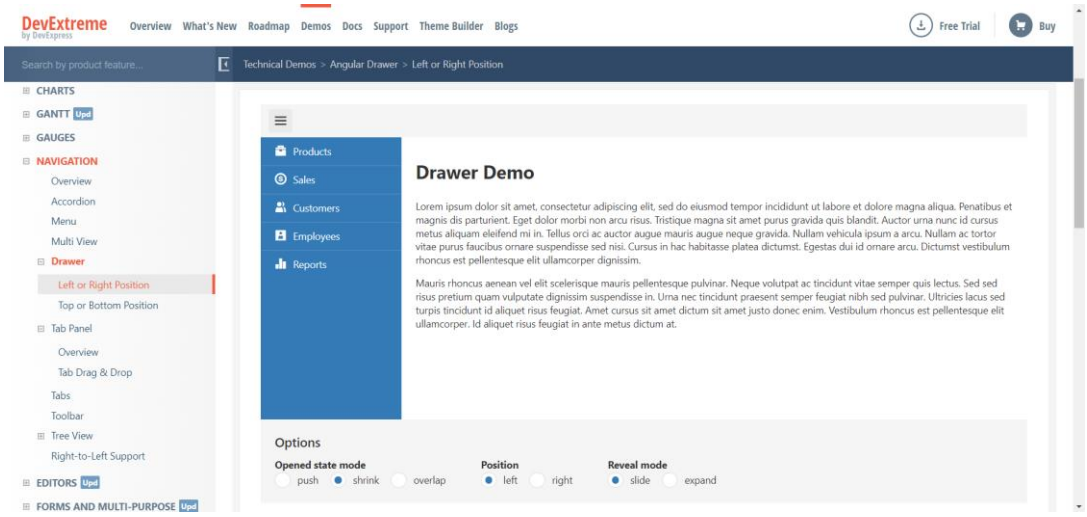


Рисунок 1.3.3 – Демо навігаційного компонента

Нижче бачимо код представленої компоненти. Хочу звернути увагу, що тут наявний код одразу до всіх можливих застосувань, тобто не тільки до обраного нами Angular, а й до React, Vue, jQuery та інших (рисунок 1.3.4). Крім того, DevExpress надає можливість редагувати представлений код прямо на сторінці, то б то одразу збирати його і показувати результат в тому ж вікні, або ж відкрити в окремому вікні пісочниці для зручності.

Власне щодо самої компоненти, то вона виявилась зручною і мінімалістичною, а це саме те, що нам було потрібно, тож ми обрали найбільш вдалий на нашу думку режим і залишили собі для майбутнього додавання в наш проект.

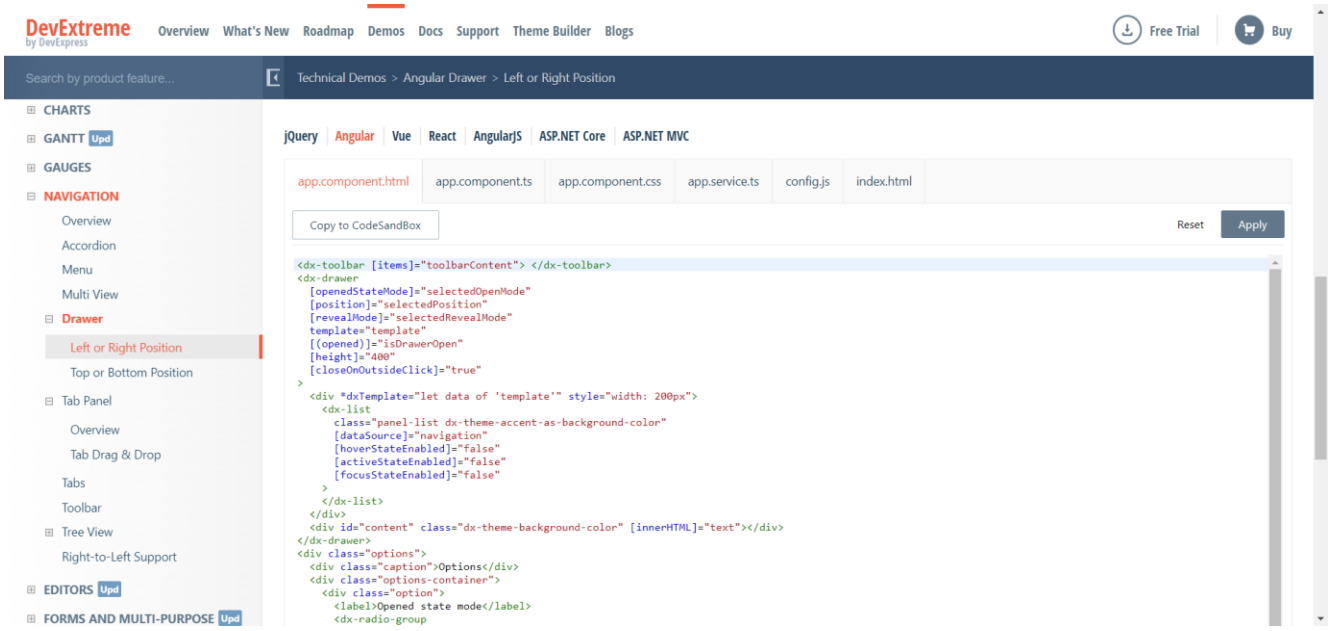



Рисунок 1.3.4 – Код навігаційної компоненти

Таблиці для перегляду даних. Компонент DevExtreme таблиць, під назвою “DataGrid”, має величезний стек доступних опцій та особливостей для розробників, в чому можете самі переконатися, поглянувши на рисунок 1.3.5. Починаючи від зв’язування з даними та групування, і завершуючи мастером деталей, що дозволяє додавати власну таблицю до кожного рядку головної таблиці. Також на рисунку 1.3.5 бачимо демо, що пропонується для першого і

найпростішого додавання таблиці в проект. Досконале поєднання мінімалістичності та функціональності, яких так часто бракує. Виглядає насправді вражаюче.

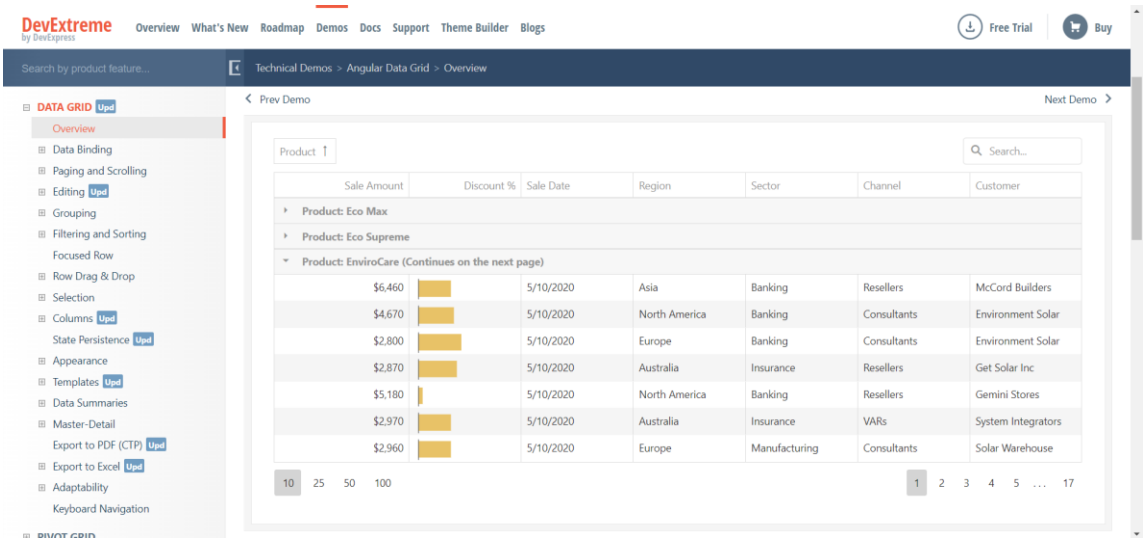



Рисунок 1.3.5 – Код навігаційної компоненти

Візуалізація статистичних даних. Для цього в бібліотеці DevExtreme є цілий розділ під назвою “Charts”. Тут ви можете знайти що завгодно, починаючи від звичайної лінійної діаграми, завершуючи полярними графіками та зображенням ієрархічних структур. Для використання в своєму додатку ми розглянули представлені на рисунках 1.3.6-1.3.8 лінійну, стовпчикову та кругову діаграми.

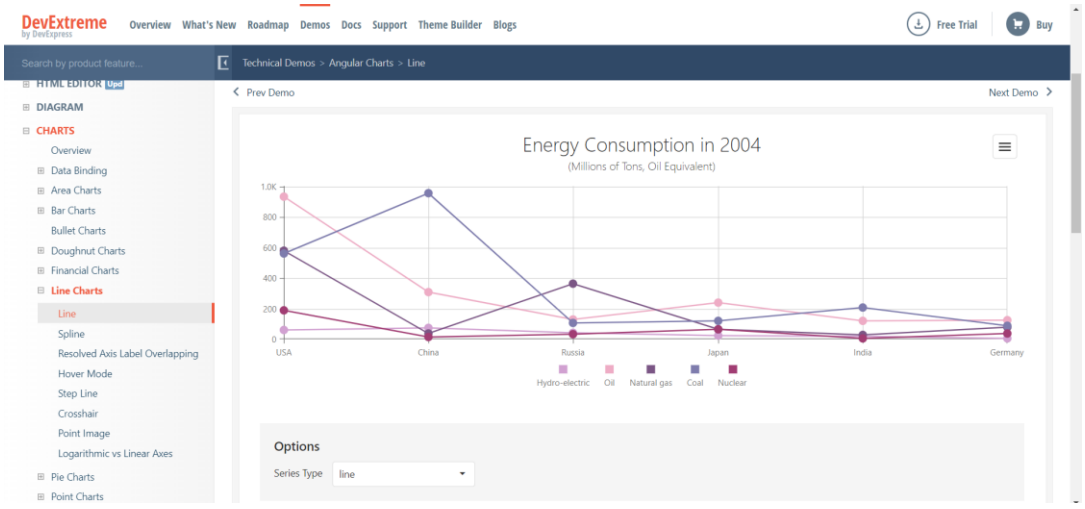



Рисунок 1.3.6 – Приклад лінійної діаграми

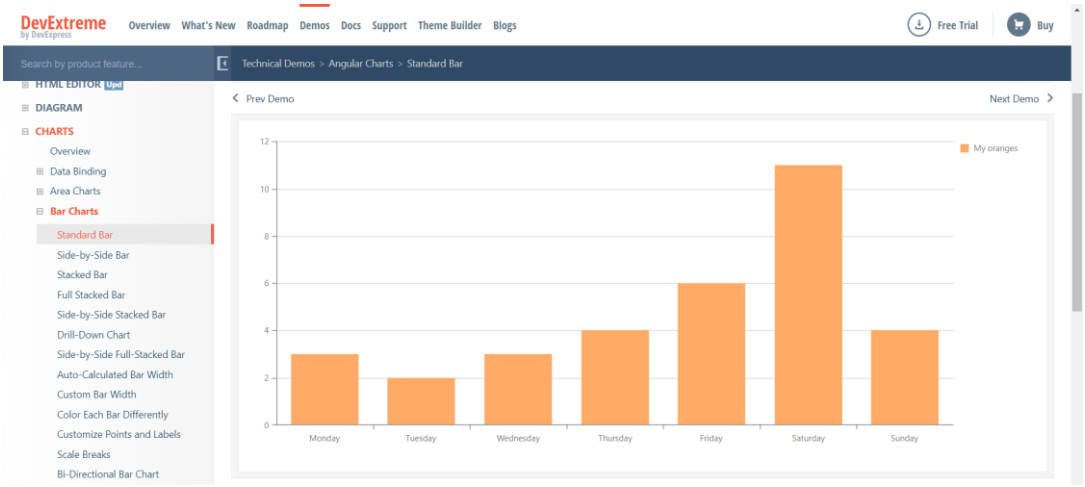



Рисунок 1.3.7 – Приклад стовпчикової діаграми

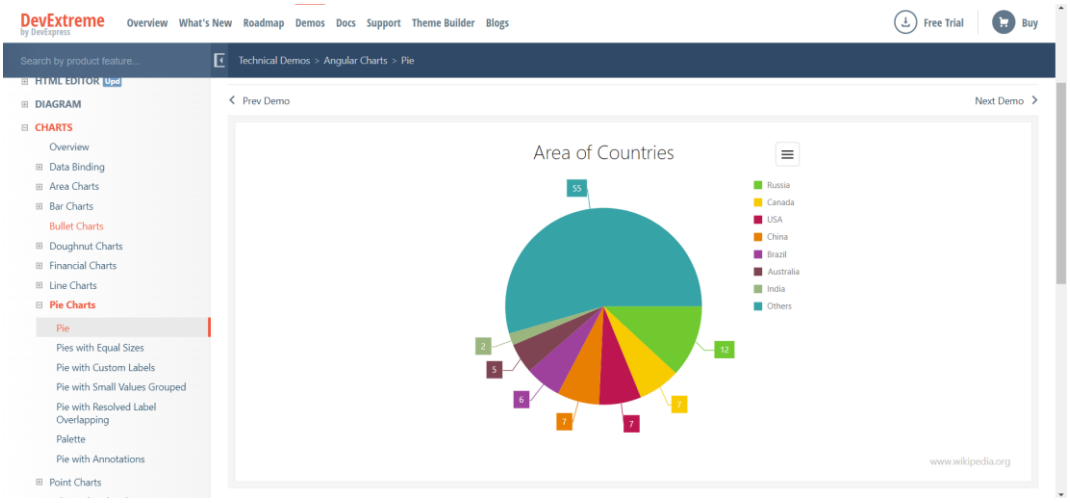



Рисунок 1.3.8 – Приклад кругової діаграми

Крім того, вже в процесі розробки додатку було вирішено додати стилізовані сповіщення. Для цього в бібліотеці DevExtreme було знайдено відповідний інструмент під назвою тост (рисунок 1.3.9). Тост підтримує 4 основні режими: успіх, інформація, попередження, помилка, а також додатковий п’ятий користувацький режим, що дає змогу створювати власні режими. Для того щоб мати засіб в своєму проекті потрібно скопіювати всього 1 рядок html коду та декілька рядків у typeScript файлі.

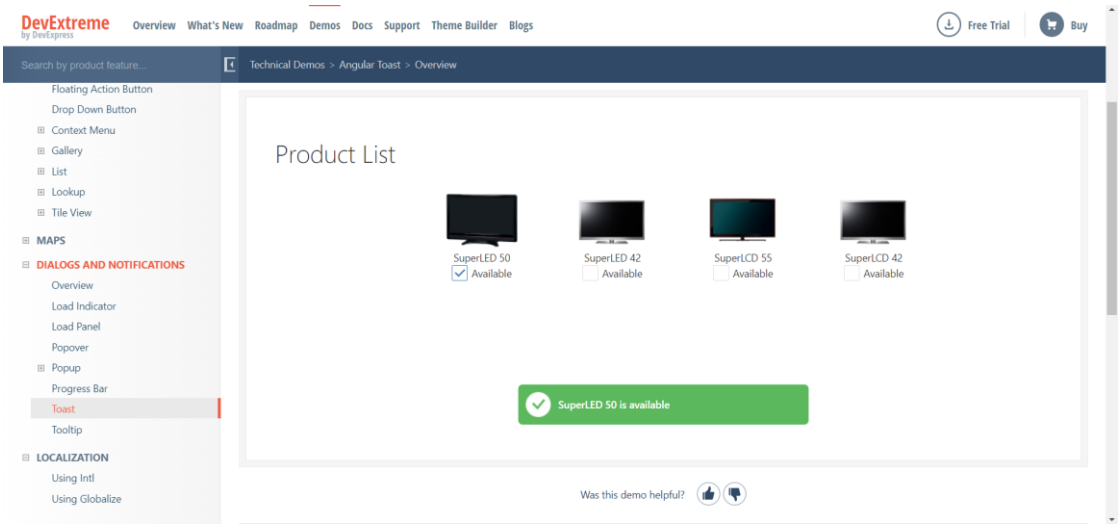



Рисунок 1.3.9 – Інструмент тост

# РОЗДІЛ 2 ОПИС СТРУКТУРНИХ ЧАСТИН ПРОГРАМИ

## 2.1 Вимоги до програми

**Вимоги до функціонування програми.** Розроблюваний додаток повинен виконувати роль веб-інтерфейсу для системи документообігу Mneme. Під цим ми розуміємо забезпечення користувача інформацією про систему; швидкою авторизацією; можливістю переглядати інформацію про документи, відкривати та завантажувати їх; а також надавати статистичні дані та доступ до методів адміністрування.

Основна інформація про систему Mneme повинна бути присутня на головній сторінці застосунку. Вона повинна включати в себе загальний опис системи Mneme, а також інформацію про веб-інтерфейс, версію та рік.

Якщо говорити детальніше про авторизацію, то кожен співробітник компанії повинен мати можливість авторизації в системі через свій корпоративний акаунт Quipu. То б то наш додаток не повинен надавати можливості реєстрації, а тільки перевіряти наявність введених користувачем даних у системі Quipu.

Щодо перегляду документів та інформації про них. Система Mneme реалізована таким чином, що документи отримуються шляхом пошуку по так званим тегам, то б то наша програма повинна надавати можливість вводити теги для пошуку або обирати із запропонованих.

Крім того важливою складовою є візуалізація, тому отриману після пошуку інформацію потрібно подавати у вигляді таблиці з можливістю окремо переглянути додаткову інформацію про файл та сам документ. Для зручності у використанні табличних даних, повинна надаватись можливість групувати, сортувати та фільтрувати рядки.

Окрім вищезазначеного, застосунок має відображати статистичну інформацію про документи, то б то отримувати дані за обраний рік та візуалізувати їх у вигляді графіків та діаграм тощо.

Говорячи про методи адміністрування, потрібно зупинитися на двох: HeartBeat та ReIndex.

Метод HeartBeat – це метод Mneme, що слугує своєрідним датчиком того, що система працює і ми до неї під’єднані. Відповідно періодичний виклик і постійна візуалізація підключення чи не підключення до Mneme мають бути присутні.

Метод ReIndex – це метод системи, що переіндексовує наявні в ній файли, тим самим зменшуючи розмір системи, що невпинно зростає. Інколи виникає потреба у негайному виклику цього методу, саме тому необхідно реалізувати доступ до нього з нашого додатку.

**Технічні вимоги.** Розроблюваний веб-інтерфейс повинен відкриватись в браузерах Google Chrome, Microsoft Edge та Apple Safari, включаючи їх останні версії.

## 2.2 Інтерфейс програми

Розроблений застосунок має мінімалістичний інтерфейс, що не буде привертати до себе багато уваги і дозволить зосередитись на робочому процесі.

Програма поділена на 4 сторінки задля відокремлення логічних частин одна від одної, а також має бічну навігацію для зручного переміщення по сторінкам.

Головна сторінка містить загальну інформацію про систему Mneme та розроблюваний нами інтерфейс.

Сторінка таблиці має, мабуть, найбільш громіздкий інтерфейс в порівнянні з іншими сторінками, що обумовлено її найбільшою функціональністю. Саме ця сторінка містить в собі поля для вводу та вибору тегів для пошуку, таблицю з отриманими результатами, а також допоміжні кнопки, що забезпечують перегляд додаткової інформації про документи, надають можливість завантажувати отриману таблицю у форматі pdf та xlsx, групувати, сортувати та фільтрувати рядки даних. В той же час, з цієї сторінки, при натисканні на кнопку додаткової

інформації, ми перейдемо до діалогового вікна. Це вікно в свою чергу забезпечує нас можливістю відкривати та завантажувати обраний документ.

Сторінка статистики представляє собою набір даних про документи додані за рік. За замовченням завантажуються дані за поточний рік, але ми також маємо можливість обрати будь-який потрібний нам рік для і переглянути статистику по ньому. Статистичні дані в зрозумілій та візуально приємній формі представлені на інформаційній панелі та трьох діаграмах.

І нарешті сторінка адміністрування, поки що найменш заповнена. На ній присутні дві функціональні частини. Одна з них називається. “ReIndex” і дозволяє натисканням однієї кнопки переіндексувати файли системи Mneme. Друга –“Heart Beat”, що постійно демонструє наявність серцебиття, то б то зв’язку з системою Mneme. Крім того, на верхній панелі, що доступна з будь-якої точки сайту, постійно демонструється та оновлюється поле з відповідною інформацією для постійного контролю серцебиття.

Для авторизації додана кнопка в правому верхньому кутку екрану, що також доступна з будь-якої точки додатку .

Стилізація та оформлення сторінок загалом зроблено в білому та блакитному кольорах. Деякі елементи, що потребують особливої уваги додатково виділяються зеленим, червоним та помаранчевим кольорами. На сторінці зі статистичними даними візуальні елементи оформлені додатковим набором кольорів задля забезпечення помітної різниці між інформаційними блоками.

Більш детально про роботу та стилізацію додатку дивіться розділи 2.3-2.4.

## 2.3 Опис структури програми

Структуру програми можна поділити на три основні частини:

- Сервер
- Система документообігу
- Клієнт

**Сервер.** В ролі сервера в моєму додатку виступає проект шаблону ASP .NET Core Web API. Власне в основному ціль його створення і була у використанні його як стартової точки для запуску клієнта.

**Система документообігу.** Як вже було зазначено вище, це система під назвою Mneme, що представляє собою певний набір файлів. До системи можна звертатись за допомогою апі-запитів, що і робить моя клієнтська частина.

**Клієнт.** Проект типу Standalone Angular Template, що добре взаємодіє з вищеназваним ASP .NET Core Web API і дозволяє створювати повноцінного клієнта окремо від сервера. Це в свою чергу в майбутньому може дуже допомогти з міграцією та/або масштабуванням додатку.

Angular проект має власну структуру, що складається з моделей, компонентів, сервісів та представлень. Наш проект також має відповідну структуру. Загалом клієнт містить в собі сім компонентів з представленнями окремо для кожного з них, а також два сервіси (рисунок 2.3.1).

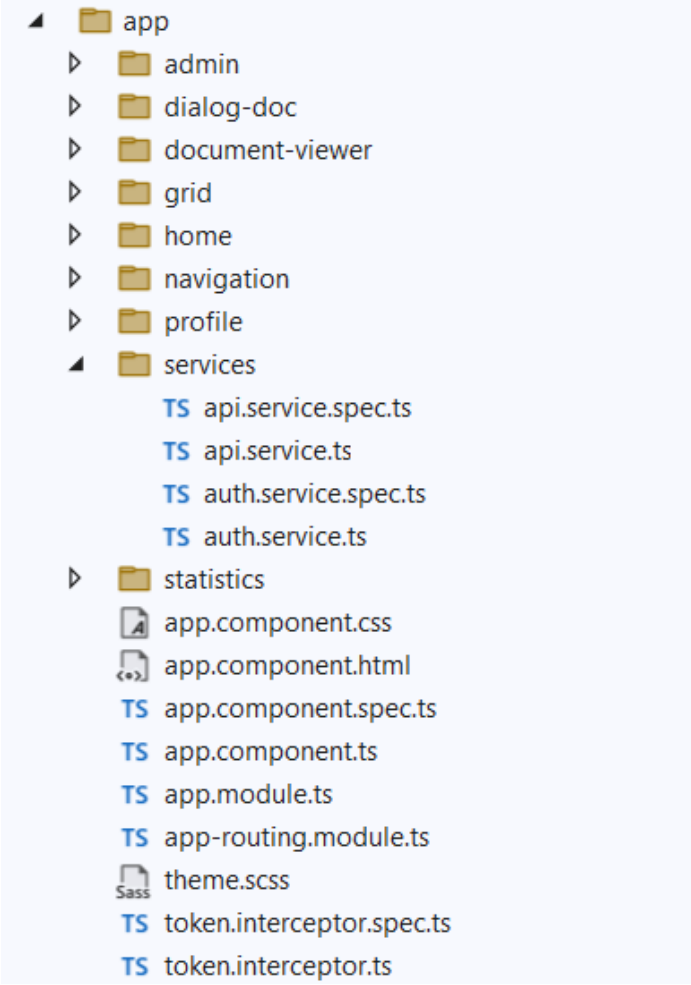



Рисунок 2.3.1 – Структура програми

Головна компонента (app.component) відображає ліву навігаційну та верхню інформаційну панелі. Крім того приймає в себе наповнення з усіх інших компонент, то б то є такою собі оболонкою.

Серед інших шести компонент присутні чотири, що відповідають за повноцінні сторінки програми і мають відповідні назви: home, grid, statistic та admin (для сторінки administration). А також дві, що відповідають за додаткові вспливаючі вікна для сторінки grid (dialog-doc та document-viewer).

На рисунку 2.3.1 також присутня папка services, вона містить в собі два сервіси. Перший – api service відповідає за звернення нашого клієнта до системи документообігу через апі запити. Він робить запит, обробляє результат, і вже в компонентах ми звертаємось до цього сервісу і беремо з нього потрібні дані.

Другий сервіс – auth (authorization) service, він в свою чергу відповідає за перевірку кожного запиту на те, чи автентифікований він чи ні. Цей сервіс

використовується в інтерсепторі, що додає в кожен запит до системи Mneme авторизаційний токен. Це зроблено тому що система обробляє запити тільки від відомих їй користувачів.

Крім сервісів та компонент потрібно ще звернути увагу на файли модулів на рисунку 2.3.1. Файл app.module – це файл, що зберігає і викликає всі підключені модулі та створені нами компоненти. Файл app-routing.module – це файл, в якому визначаються шляхи для виклику компонент, якщо це потрібно.

Детальніше про реалізацію та наповнення кожної структурної частини в розділі 2.4 Реалізація програми.

## 2.4 Реалізація програми

Проект реалізований відповідно до структури описаної у розділі 2.3, тож можемо детальніше розглянути структуру та принцип роботи кожної структурної частини.

**Сервер.** Як вже було сказано вище, ціллю його створення була у використанні його як стартової точки для запуску клієнта, тому стандартний проект ASP .NET Core Web API залишився майже без змін. Єдиною зміною на даний момент є імплементований фреймворк для логування Serilog (рисунок 2.4.1). Ми підключили йому можливість ведення журналу в базу даних, файл та Application Insights.

```
// Add logger
var logger = new LoggerConfiguration()
    .ReadFrom.Configuration(builder.Configuration)
    .CreateLogger();

builder.Logging.ClearProviders();
builder.Logging.AddSerilog(logger);
```

Рисунок 2.4.1 – Додавання логеру до сервера

Для того щоб додати можливість записувати логи потрібно прописати додаткові параметри ведення журналу у файлі appsettings.json. Приклад таких параметрів для запису у файл зображена на рисунку 2.4.2.

```
"WriteTo": [
  {
    "Name": "File",
    "Args": {
      "path": "logs/log.txt",
      "rollingInterval": "Day",
      "fileSizeLimitBytes": null,
      "retainedFileCountLimit": null
    }
  },
```

Рисунок 2.4.2

**Система документообігу.** Відповідно до зазначеного в попередньому розділі, клієнтська частина нашої програми звертається безпосередньо до системи через апі запити. Тому пропоную перейти до розгляду клієнта.

**Клієнт.** Містить в собі сім компонентів з представленнями та два сервіси.

Пропоную почати з сервісів. Сервіс api.service відповідає за звернення до системи за допомогою апі-запитів, тому реалізує близько 7-10 функцій, кожна з яких звертається до системи Mneme. Приклад однієї з таких функцій наведено на рисунку 2.4.3.

```
Search() {
  var formData: any = new FormData();

  this.tags = [{ 'Prefix': this.selectedPrefix, 'Value': this.value, 'Owner': this.selectedOwner }];
  console.log("tags", this.tags);

  formData.append("tags", JSON.stringify(this.tags));
  console.log(formData);

  return this.http.post<any>("[redacted]", formData)
    .subscribe(
      response => {
        console.log(response);
        this.docs = response;
        if (this.docs.length == 0) {
          this.type = 'warning';
          this.message = 'This request does not give any answer. \nPlease, enter another tags!';
          this.isVisible = true;
          return;
          }

      });
}
```

Рисунок 2.4.3

Сервіс auth.service має два методи, що перевіряють наявність токену користувача, то б то чи цей користувач зареєстрований. В свою чергу token interceptor, про який також йшлося в попередньому розділі, приймає відповідь і додає цей токен до всіх запитів до системи документообігу (рисунок 2.4.4).

```
export class TokenInterceptor implements HttpInterceptor {
  constructor(public auth: AuthService) { }
  intercept(request: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>> {

    request = request.clone({
      setHeaders: {
        Authorization: `Bearer ${this.auth.getToken()}`
      }
    });
    return next.handle(request);
  }

}
```

Рисунок 2.4.4 – Токен інтерсептор

На цьому повноваження сервісів завершені, тож можемо переходити до компонентів та їх представлень.

Головний компонент на даний момент є оболонкою, що приймає в себе наповнення інших компонент, а також містить в собі навігаційну ліву та інформаційну верхню панель. Крім того, на верхній панелі присутня кнопка авторизації, тому наш app component обробляє і авторизаційні дані також.

Авторизація в нашому проекті виконана за допомогою Microsoft Graph API. Це технологія, яка дозволяє використовувати базу користувачів Microsoft і обмежувати створення нових. Саме цей метод був для нас зручним, адже ми якраз маємо потрібну базу. Приклад взаємодії нашого проекту з користувацькою базою Microsoft зображено на рисунку 2.4.5 (зліва клієнт нашого додатку, справа база користувачів).

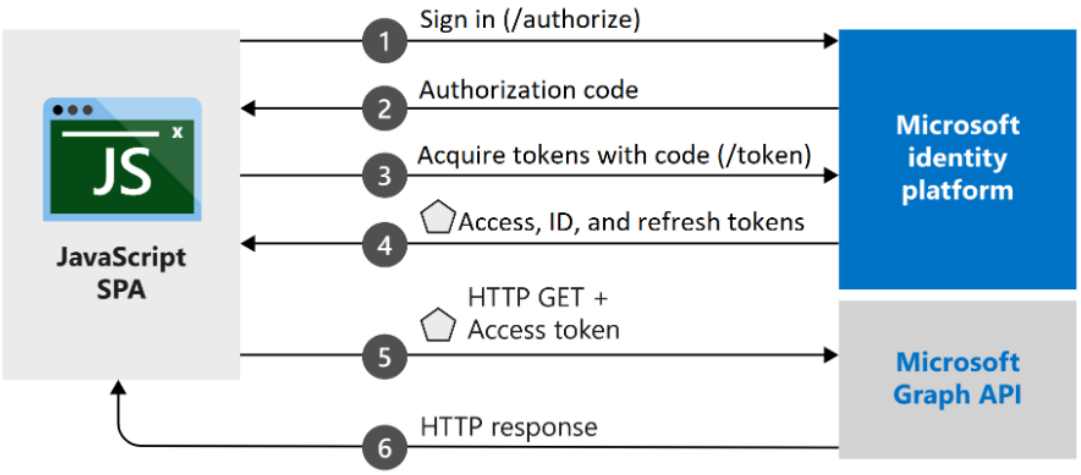



Рисунок 2.4.5

Наш проект використовує представлену технологію і відповідно має код для авторизації користувача в системі, що зображений на рисунку 2.4.6.

```
login() {
  if (this.msalGuardConfig.authRequest) {
    this.authService.loginRedirect({...this.msalGuardConfig.authRequest } as RedirectRequest);
  } else {
    this.authService.loginRedirect();
  }
}

logout() {
  this.authService.logoutRedirect({
    postLogoutRedirectUri: 'http://localhost:4200/'
  })
}

setLoginDisplay() {
  this.loginDisplay = this.authService.instance.getAllAccounts().length > 0;
  if (this.loginDisplay) {
    this.userName = this.authService.instance.getAllAccounts()[0].name;
    this.userEmail = this.authService.instance.getAllAccounts()[0].username;
  }
}
```

Рисунок 2.4.6

Також, компонент має навігаційну панель під назвою dx-drawer, що є продуктом DevExtreme бібліотеки. Він має html код, представлений на рисунку 2.4.7. Рядок 35 містить в собі кнопки для виклику всіх доступних нам сторінок, а рядок 80 власне приймає в себе наповнення інших компонент.

```
28   <dx-drawer openedStateMode="overlap"
29              position="left"
30              revealMode="expand"
31              template="template"
32              [(opened)]="isDrawerOpen"
33              [closeOnOutsideClick]="true">
34     <div *dxTemplate="let data of 'template'">
35       <ul class="items">…</ul>
79     </div>
80     <div class="container">…</div>
125  </dx-drawer>
```

Рисунок 2.4.7

Верхня інформаційна панель в свою чергу представлена також елементом з DevExtreme бібліотеки. Вона має назву dx-toolbar і показана на рисунку 2.4.8. Як і більшість елементів-обгорток з цієї бібліотеки, цей елемент надає можливість створювати зручну структуру всередині за рахунок елементу dxi-item.

```
1    <dx-toolbar class="bar">
2      <dxi-item>…</dxi-item>
6
7      <dxi-item>…</dxi-item>
17
18     <dxi-item>…</dxi-item>
24
25   </dx-toolbar>
```

Рисунок 2.4.8

Далі пропоную розглянути компонент таблиці. Це найбільш функціонально наповнений компонент нашого клієнта, адже він вміщає в себе поля для вибору та введення тегів для пошуку, таблицю результатів пошуку та кнопки для перегляду обраного файлу та додаткової інформації про нього.

Для надання можливості користувачу обирати серед доступних овнерів та префіксів ми за допомогою апі сервісу звертаємось до системи і отримуємо вичерпний список. Після цього відображаємо його як зображено на рисунку 2.4.9. До речі, також за допомогою елементу з бібліотеки DevExtreme під назвою dx-lookup.

```
<dx-lookup class="select"
           [dataSource]="ownerDataSource"
           [grouped]="false"
           placeholder=""
           valueExpr="id"
           displayExpr="name"
           label="Owner"
           labelMode="floating"
           (onValueChanged)="ownerChanged($event)">
  <dxo-drop-down-options [closeOnOutsideClick]="true" [showTitle]="true" title="Owner">
  </dxo-drop-down-options>
</dx-lookup>
```

Рисунок 2.4.9

Поле велью в залежності від обраного префіксу може бути довільним, тому це звичайний mat-input від бібліотеки Angular Material, навіть без валідації.

Таблиця зі знайденими документами виконана за допомогою елемента dx-data-grid також з DevExtreme бібліотеки. Загалом цей елемент має дуже велику кількість додаткових налаштувань, частину з них ми використали в своїй реалізації, адже ці налаштування додають зручності у використанні таблиці (основна частина налаштувань таблиці на рисунку ). Наприклад, dxo-pager дозволяє додати налаштування кількості показаних рядків в таблиці, а також можливість зміни цієї кількості самим користувачем.

```
<dx-data-grid id="gridContainer"
              #grid
              class="grid"
              [dataSource]="docs"
              [remoteOperations]="false"
              [allowColumnReordering]="true"
              [rowAlternationEnabled]="true"
              [showBorders]="true"
              (onContentReady)="contentReady($event)"
              (onSelectionChanged)="but($event)">

  <dxo-paging [pageSize]="10"></dxo-paging>
  <dxo-pager [showPageSizeSelector]="true"
             [allowedPageSizes]="[10, 25, 50, 100]"
             [showInfo]="true"
             [showNavigationButtons]="true"></dxo-pager>
  <dxo-search-panel [visible]="true"
                    [highlightCaseSensitive]="true">
  </dxo-search-panel>
  <dxo-group-panel class="group" [visible]="true"></dxo-group-panel>
  <dxo-grouping [autoExpandAll]="false"></dxo-grouping>
  <dxo-selection mode="single" class="select"></dxo-selection>
  <dxo-editing [useIcons]="true"></dxo-editing>
  <dxo-toolbar>…</dxo-toolbar>

  <dxi-column dataField="originalFileName" type="string">
```

Рисунок 2.4.10

З цього ж компонента ми можемо компонент перегляду додаткової інформації про файл, що має в собі кнопки перегляду та завантаження файла. Кнопка перегляду в свою чергу відкриє ще однин компонент. Обидва додаткових компонента представлені елементом mat-dialog з бібліотеки Angular Material.

Далі переходимо до компонента статистики. Для її створення було використано три різні види діаграм: лінійну, стовпчасту та кругову. Лінійна представляє з себе просто набір графіків для кожного префікса і має код зображений на рисунку 2.4.11. Дві інші діаграми мають схожу але більш громіздку структуру коду.

```
<dx-chart id="chart" [dataSource]="shortData"
          palette="soft blue" [rotated]="true">
  <dxo-common-series-settings argumentField="tagPrefix"
                              type="bar"
                              valueField="count"
                              [ignoreEmptyPoints]="true">
  </dxo-common-series-settings>
  <dxo-series-template nameField="tagPrefix"></dxo-series-template>
  <dxo-legend [margin]="30"
              horizontalAlignment="right"
              verticalAlignment="top"></dxo-legend>
  <dxo-tooltip [enabled]="true">
  </dxo-tooltip>

</dx-chart>
```

Рисунок 2.4.11

І останній компонент адміністрування. Говорячи про його функціональність, потрібно зупинитися на двох методах: ReIndex та HeartBeat.

Метод ReIndex – це метод системи, що переіндексовує наявні в ній файли, тим самим зменшуючи розмір системи, що невпинно зростає. Загалом система сама вміє викликати цей метод з заданою періодичністю. Проте інколи, коли, наприклад, Quipu підписує контракт з новим банком, відповідно на деякий час кількість звернень до системи різко зростає. Відповідно до цього зростає і потреба переіндексовувати файли частіше.

В подальшому, з розширенням функціональності системи, планується також збільшення наповнюваності сторінки адміністрування.

# РОЗДІЛ 3 ІНСТРУКЦІЯ КОРИСТУВАЧА

Ласкаво просимо до нашого додатку MnemeWebUI. Після запуску ми опиняємось на головній сторінці під назвою Home (рисунок 3.1). На ній одразу бачимо коротку інформацію про систему Mneme загалом та про розроблений нами WebUI.

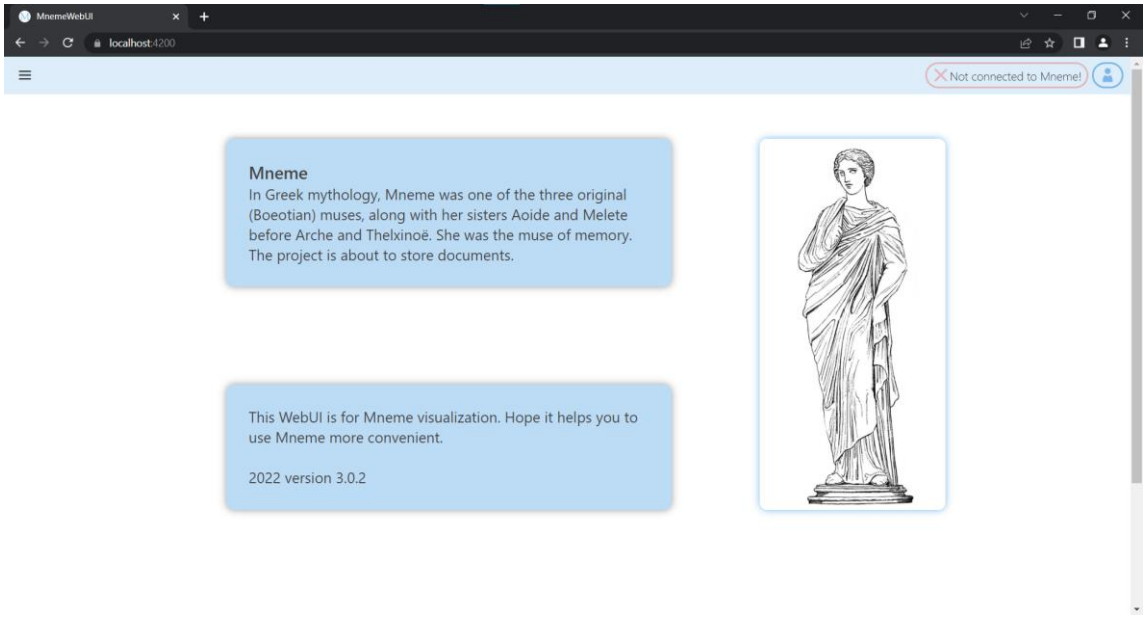



Рисунок 3.1 – Головна сторінка

Також на рисунку 3.1 ми бачимо верхню панель, на якій зліва-направо: кнопка відкриття/закриття навігаційного меню, індикатор підключення до системи Mneme та кнопка для відкриття панелі користувача. Відкривши панель користувача, можна побачити попередження, що для використання застосунку потрібно авторизуватись, і кнопку входу (рисунок 3.2).

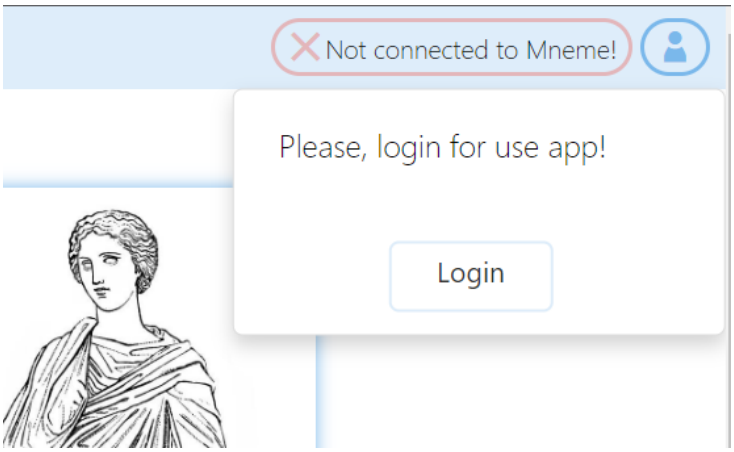



Рисунок 3.2

Ми натискаємо кнопку входу і авторизуємось в системі. Після цього застосунок повертає нас на цю ж саму сторінку (рисунок 3.3). Тут ми можемо звернути увагу на те, що індикатор підключення до Mneme став зеленим і сигналізує нам про можливість взаємодії з системою. А також на те, що кнопка відкриття панелі користувача почала відображати ім’я користувача, а відкривши панель можна побачити також пошту та кнопку виходу (рисунок 3.4).

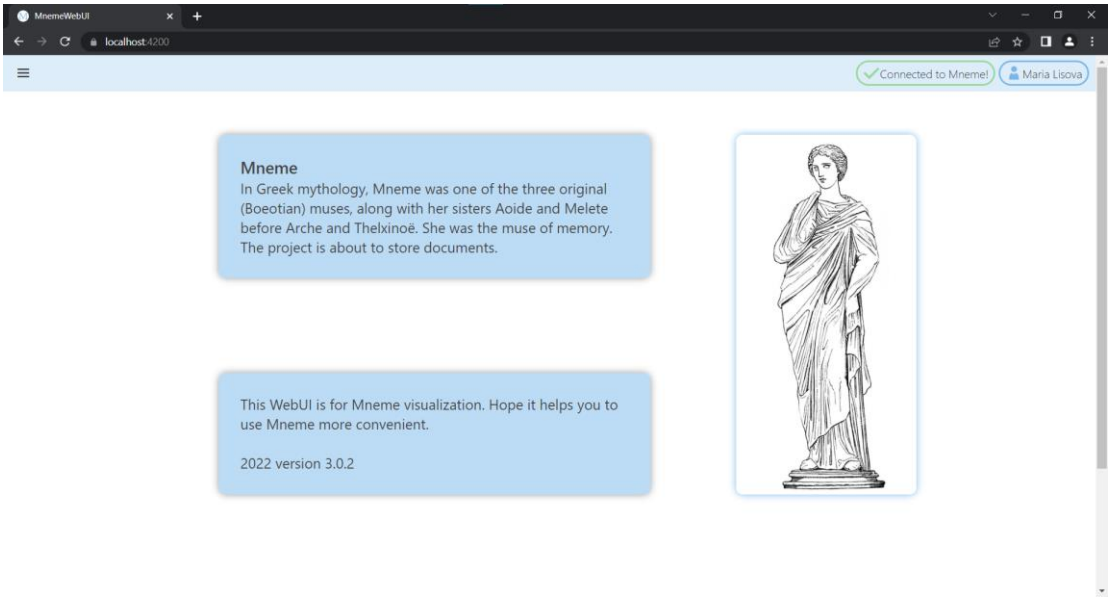



Рисунок 3.3 – Головна сторінка після авторизації

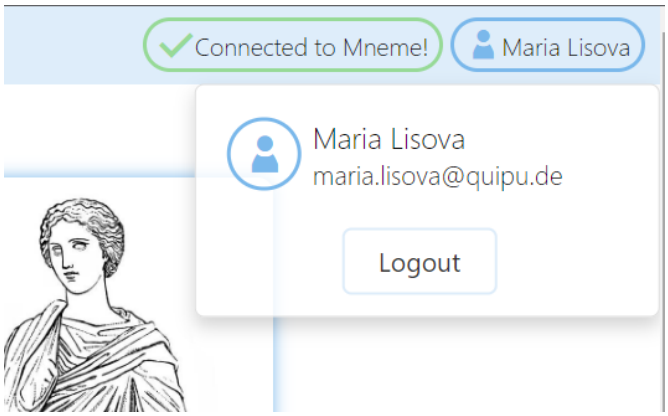



Рисунок 3.4

Тут також можемо відкрити навігаційну панель зліва і побачити доступні нам сторінки. Оскільки ми зараз на сторінці Home, пропоную перейди до наступної під назвою Grid.

На рисунку 3.5 зображена сторінка таблиці. На ній ми бачимо панелі для вибору (Owner, Prefix) та вводу (Value) тегів, кнопку пошуку (Search), кнопку отримання додаткової інформації (Get additional info), а також таблицю.

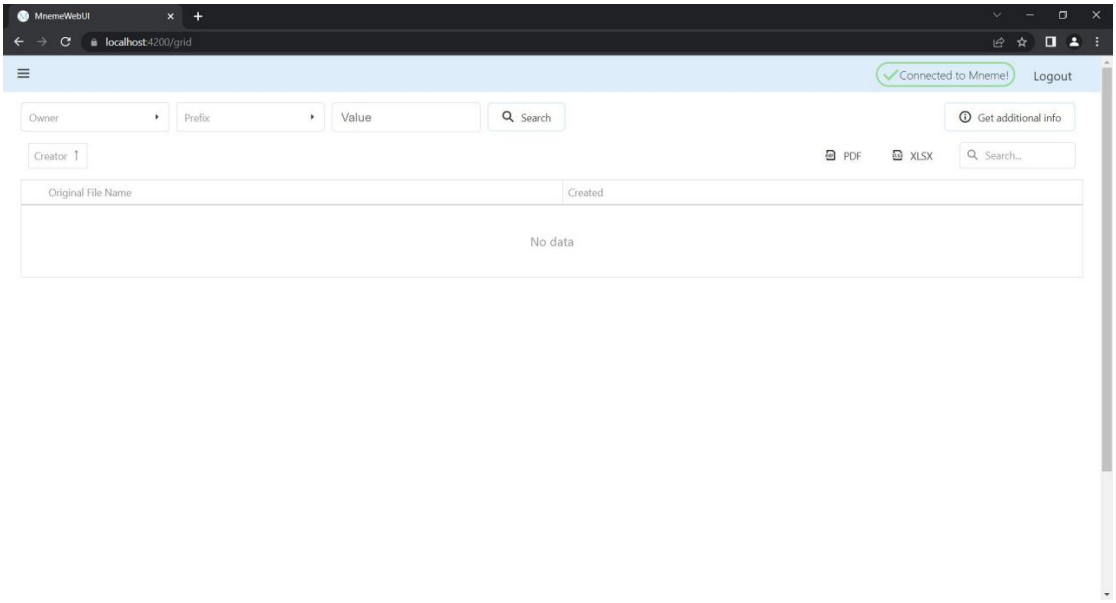



Рисунок 3.5 – Сторінка Grid

Для демонстрації роботи таблиці обираємо допустимі значення для тегів Owner та Prefix (рисунок 3.6) і заповнюємо значенням поле Value. Натискаємо кнопку Search. І отримуємо результати представлені на рисунку 3.7.

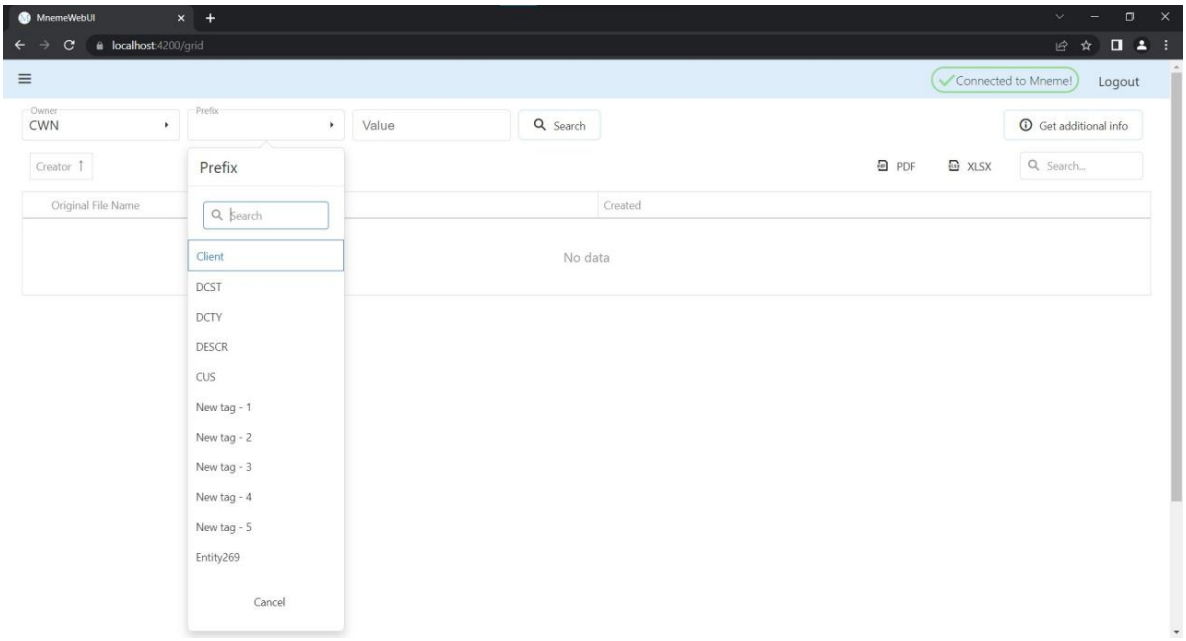



Рисунок 3.6 – Вибір тега

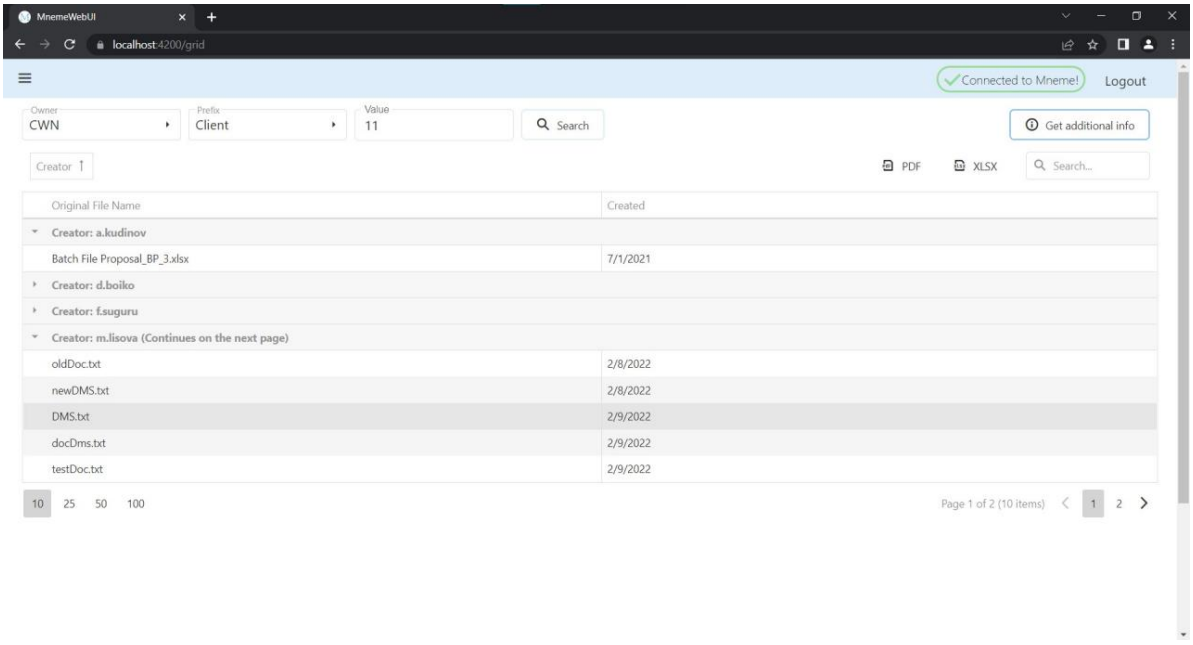



Рисунок 3.7 – Результати пошуку

Тепер, коли ми маємо заповнену таблицю, можна переходити до більш детального розгляду її можливостей.

Пропоную звернути увагу на прямокутник з написом “Creator” зліва зверху. Він призначений для групування рядків таблиці на сортування їх по стовпцю у прямому та оберненому порядку. Власне, зараз таблиця згрупована по імені співробітника, що додав цей файл.

Ще один дуже зручним інструментом таблиці є поле пошуку у правому верхньому кутку. Якщо ми знаємо який конкретно файл нам потрібен, або хочемо відфільтрувати по людині, що додавала ці файли, просто вводимо потрібну назву або ім’я. Наприклад, на рисунку 3.6 видно, що я почала вводити своє ім’я, і таблиця одразу залишила рядки тільки з документами доданими мною.


Рисунок 3.8 – Відфільтрована таблиця

Далі розглянемо кнопки, що знаходяться біля поля пошуку (рисунок 3.9). Мова йде про ті, що мають назву PDF та XLSX відповідно. Це дуже зручні інструменти для збереження таблиці для подальшої обробки. Натиснувши на будь-яку з них, ви миттєво завантажите результати роботи з таблицею у відповідному форматі (рисунок 3.10).

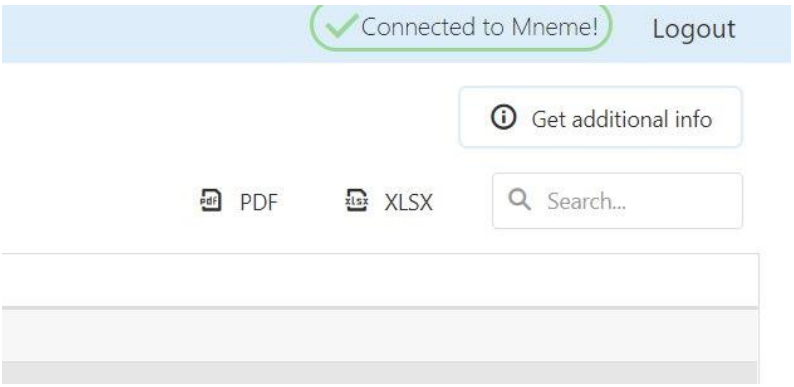
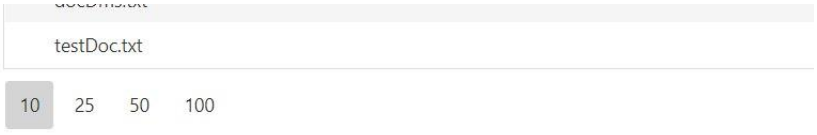



Рисунок 3.9



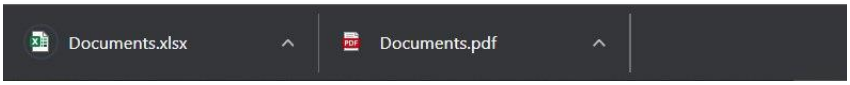




Рисунок 3.10

Також, на рисунку 3.9 ми бачимо кнопку під назвою Get additional info. Вона призначена для отримання додаткової інформації. Якщо ми просто натиснемо на неї, застосунок підкаже нам, що потрібно обрати документ, про який ми хочемо отримати інформацію (рисунок 3.11). Якщо ж ми вже обрали рядок і після цього натиснули кнопку, то нам відкриється нове невеличке вікно з більш докладною інформацією про файл (рисунок 3.12).


Рисунок 3.11

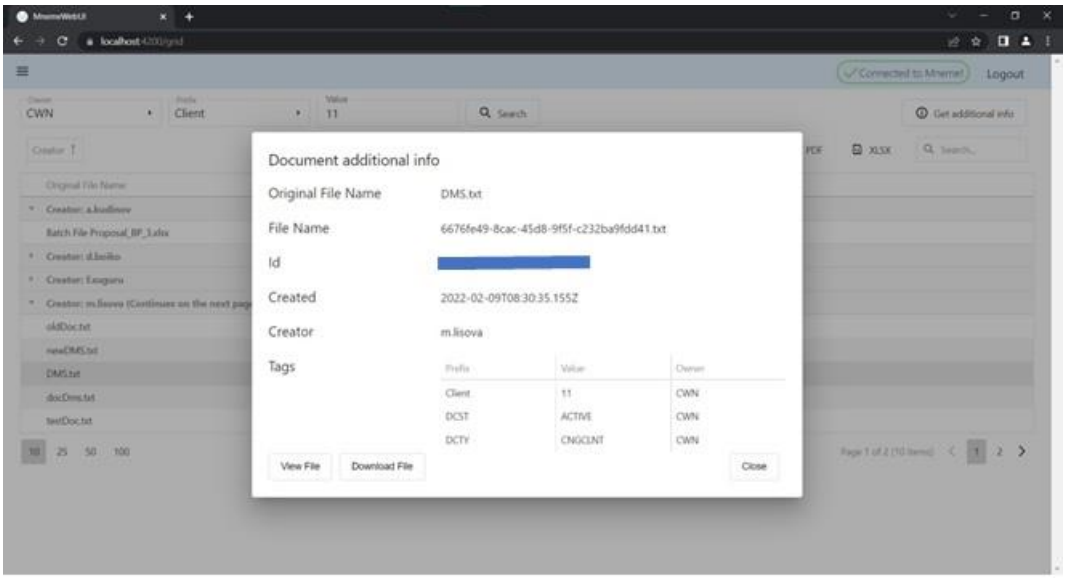



Рисунок 3.12

У цьому вікні відображається вся інформація, що міститься в системі, про обраний документ. Блок також має дві додаткові можливості переглядати та завантажувати даний файл, View та Download File відповідно. При спробі відкрити файл з’явиться нове вікнце з наповненням документу (рисунок 3.13). При спробі завантаження відбудеться теж саме, що показано раніше на рисунку 3.10.

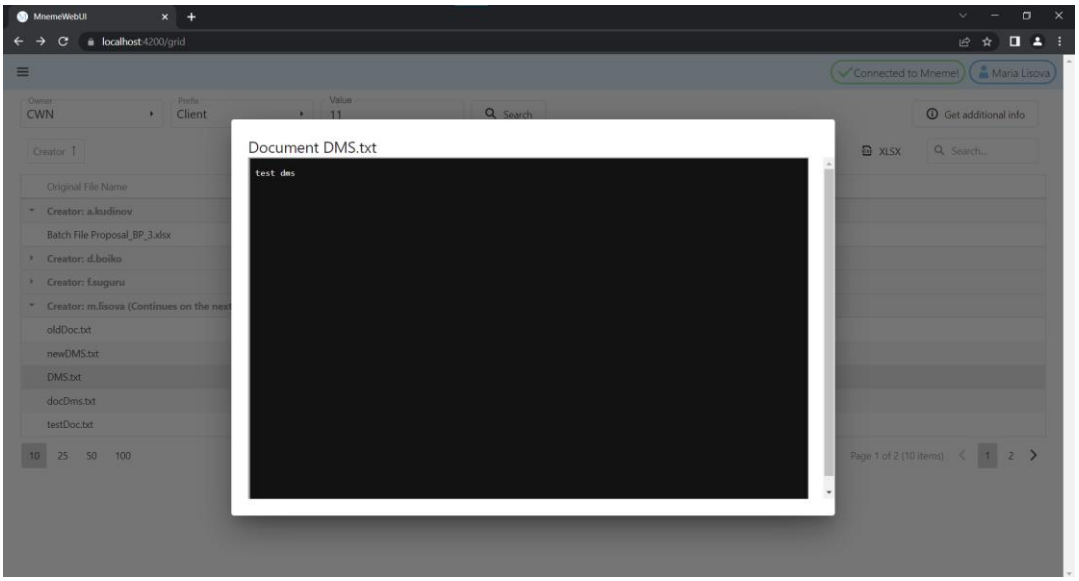



Рисунок 3.13

На цьому функціонал сторінки Grid поки що вичерпався, тому переходимо на наступну сторінку під назвою Statistics (Рисунок 3.14). При першому відкритті цієї сторінки, нам завантажуються дані за поточний рік.

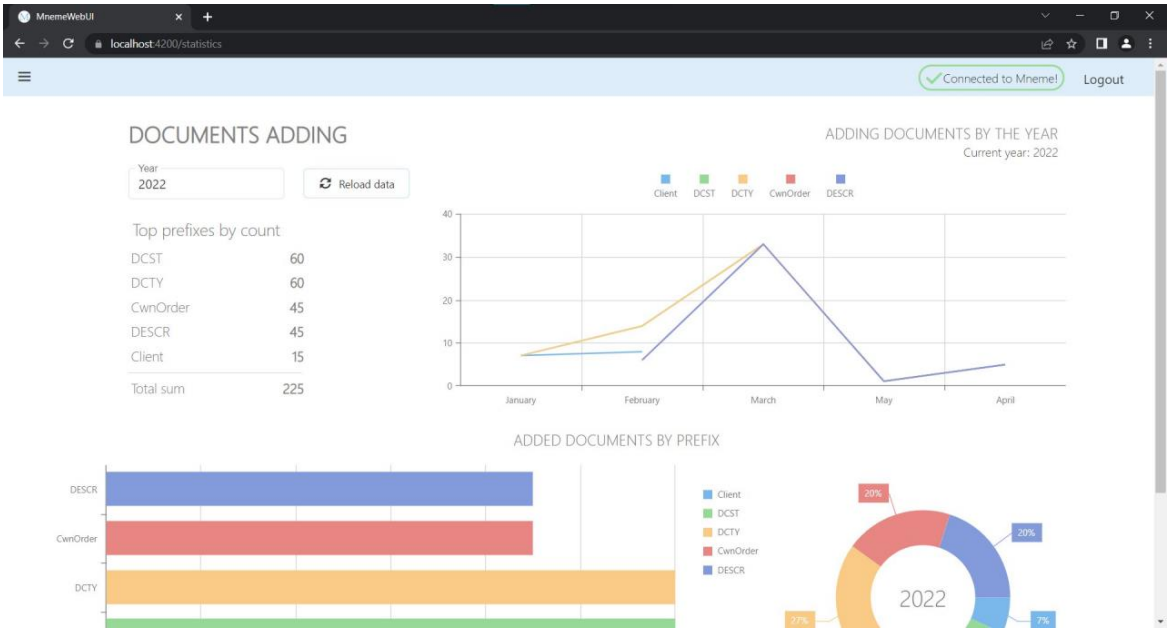



Рисунок 3.14 – Сторінка Statistics

Тут ми можемо бачити три основні блоки, один текстовий і два графічних. Текстовий блок окремо зображений на рисунку 3.15. В ньому також є два блоки. перший надає нам можливість ввести інший рік для оновлення інформації (рисунок 3.16). Для цього потрібно ввести потрібний рік в поле Year та натиснути

кнопку Refresh. Другий показує топ п’ять префіксів за кількістю документів в них та загальну суму документів по всім префіксам за обраний рік.

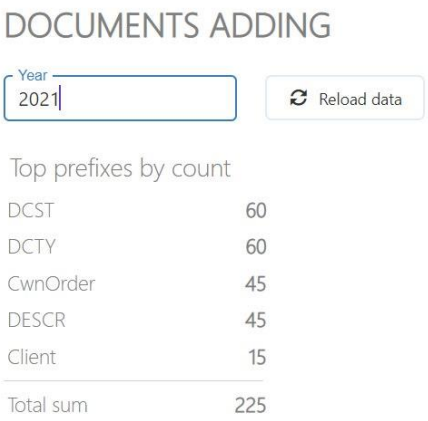



Рисунок 3.15

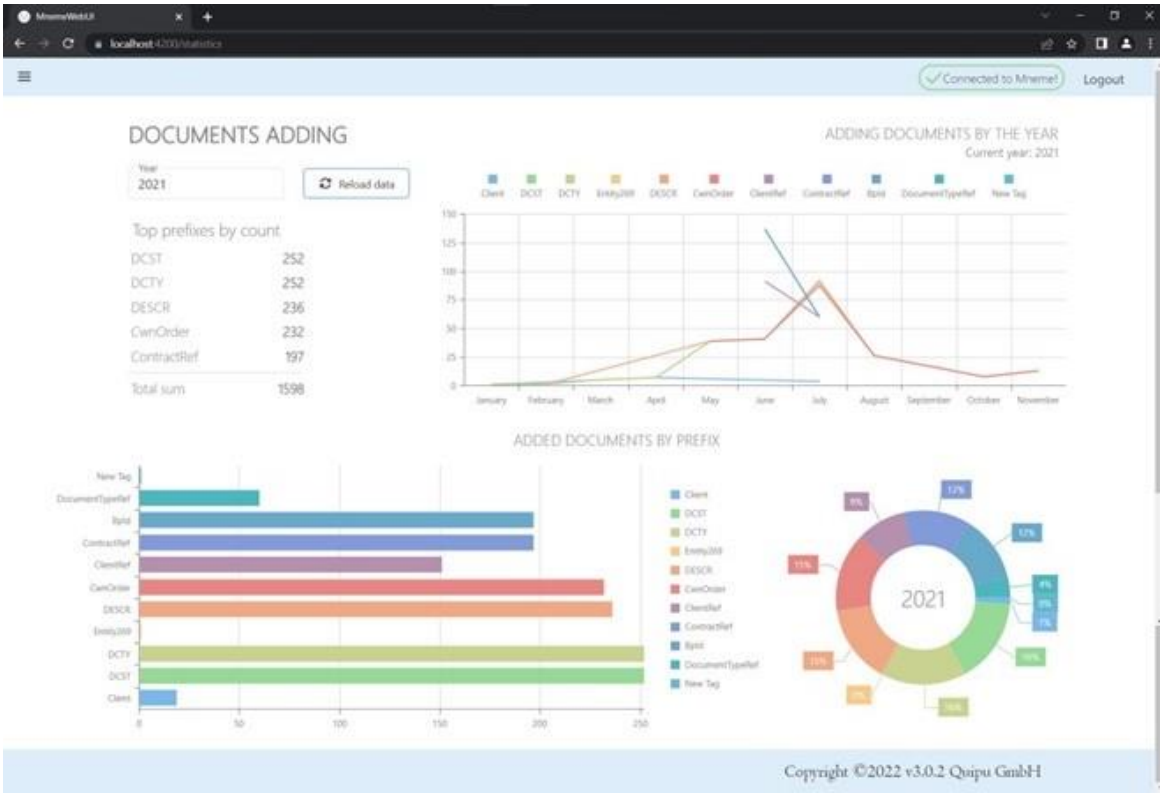



Рисунок 3.16 – Інформація за попередній 2021 рік.

Щодо графічних блоків, то на них зображено порівняльні графіки та діаграми для кількості документів по кожному з префіксів. Крім того, на графіку під назвою Adding Documents By The Year цей графік розподілений по місяцям року задля більш зручного відстеження активності в різні періоди.

Далі ми переходимо на сторінку Administration. На рисунку 3.17 видно, що тут поки що лише два функціональні блоки. Блок Heart Beat відповідає за демонстрацію підключення до системи, як і індикатор на верхній панелі, про який йшла мова на початку розділу. На цій сторінці блок серцебиття має можливість “запускати” його, то б то при натисканні застосунок буде звертатись до сервера, а також індикатором, то б то серцебиття буде помітне, коли зв’язок є (картинка серця анімована), і не помітне, коли зв’язку немає.

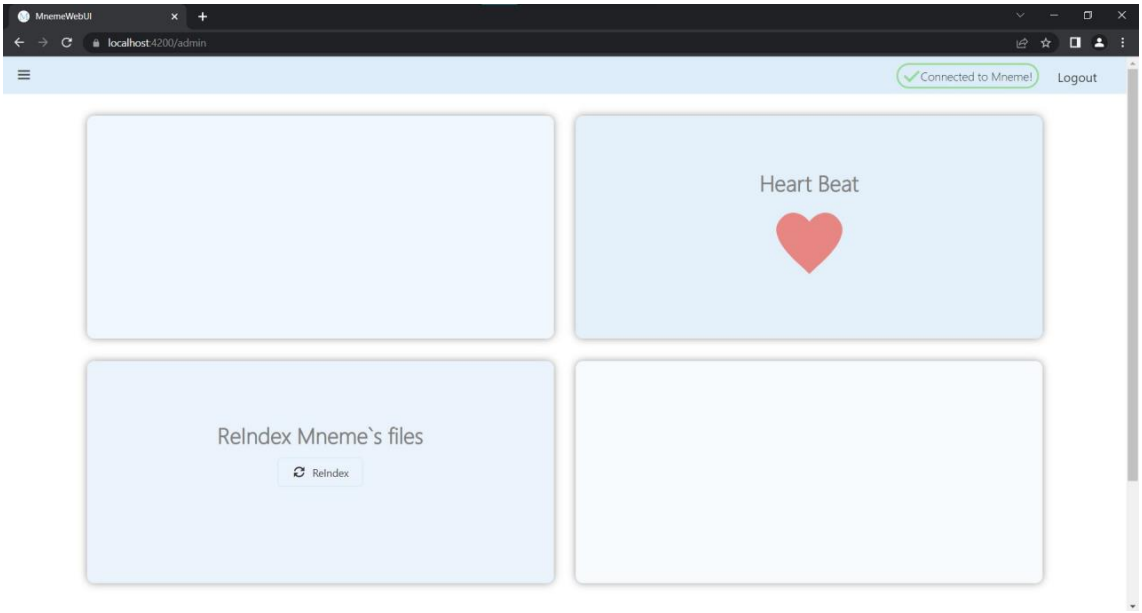



Рисунок 3.17 – Сторінка Administration

Ще один функціональний блок містить в собі кнопку ReIndex і дозволяє переіндексувати файли системи, про що вже йшлося в попередніх розділах. Після натискання на кнопку, система обов’язково повідомить Вас про успішність або неуспішність виконаної операції (рисунок 3.18).

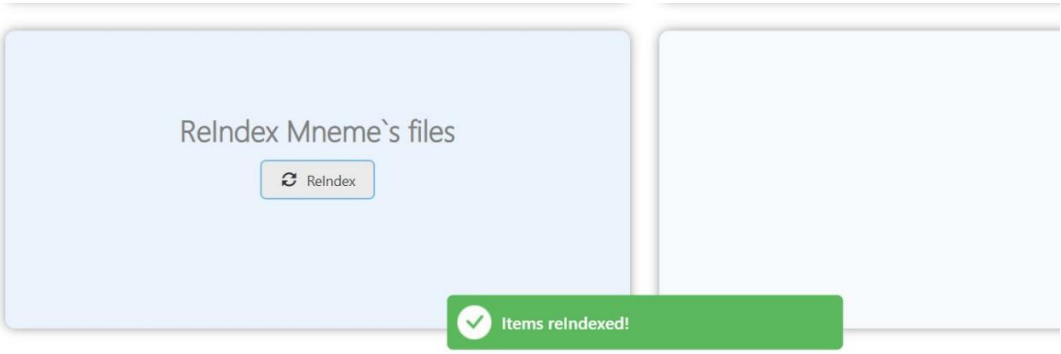



Рисунок 3.18 – Відфільтрована таблиця

# ВИСНОВКИ

В результаті виконаної роботи було створено веб-інтерфейс MnemeWebUI. Також було виконано всі завдання поставлені на початку роботи. А саме, ми порівняли найбільш відомі платформи розробки односторінкових додатків та обрали з них найкращий для нашого застосунку, таким виявився Angular. Ми дослідили взаємодію Angular та бібліотеки DevExtreme від компанії DevExpress, а також успішно застосували бібліотеку під час розробки нашого додатку. Крім того, ми успішно провели апробацію роботи в компанії Quipu GmbH, деякі співробітники якої вже успішно використовують програму та пропонують способи покращення.

Розроблений застосунок має ряд переваг, таких як зручність у використанні, повнота відображуваної інформації про документи, можливість легкої міграції та масштабування. Крім того, додаток розроблявся під безпосереднім контролем людей, що будуть його використовувати. То б то він повністю відповідає потребам майбутніх користувачів, що також є величезним плюсом.

Програма поки що має певні недоліки та, на мою думку, не повний функціонал. Але ми на цьому не зупиняємось, саме тому можна точно сказати, що розробка застосунка буде повністю завершена і поширена серед більшої кількості співробітників компанії Quipu GmbH вже в найближчий час.

З науково-технічної точки зору розроблена програма може стати першою сходинкою для значного розширення можливостей системи Mneme та створення цілої низки подібних веб-інтерфейсів. Крім того результати представленої роботи можна розглядати як дослідження взаємодії Angular з продукцією компанії DevExpress та використовувати у якості посібника.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ


1. Quipu GmbH [Електронний ресурс] – Режим доступу до ресурсу: https://jobs.dou.ua/companies/quipu-gmbh/
2. Мнемосіна [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%9C%D0%BD%D0%B5%D0%BC%D0%BE%D1%81%D1%96%D0%BD%D0%B0
3. Екстремальне програмування [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%95%D0%BA%D1%81%D1%82%D1%80%D0%B5%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F
4. Міллер Р. Екстремальне програмування: постановка процесу з перших кроків та до переможного кінця. / Р. Міллер, К. Ауер. – Пітер: СПб, 2003. – 368 с.
5. Microsoft Build. Visual Studio family [Електронний ресурс] – Режим доступу до ресурсу: https://visualstudio.microsoft.com
6. What is Azure DevOps? [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/azure/devops/user-guide/what-is-azure-devops?view=azure-devops
7. Serilog Tutorial for .NET Logging: 16 Beat Practices and Tips [Електронний ресурс] – Режим доступу до ресурсу: https://stackify.com/serilog-tutorial-net-logging/
8. Application Insights overview [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/azure/azure-monitor/app/app-insights-overview
9. Provided Sinks [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/serilog/serilog/wiki/Provided-Sinks

10. What is TypeScript? [Електронний ресурс] – Режим доступу до ресурсу: https://www.typescriptlang.org/
11. Introduction to the Angular Docs [Електронний ресурс] – Режим доступу до ресурсу: https://angular.io/docs
12. Фрімен А. Angular для професіоналів / А. Фрімен. – Пітер: СПб, 2017. – 800 с.
13. DevExpress [Електронний ресурс] – Режим доступу до ресурсу: https://www.componentsource.com/brand/devexpress
14. DevExpress in 2020s [Електронний ресурс] – Режим доступу до ресурсу: https://www.devexpress.com/aboutus/
15. Angular vs Vue.js: що обрати [Електронний ресурс] – Режим доступу до ресурсу: https://dou.ua/lenta/columns/angular-vs-vuejs/
16. React vs Vue.js: що вивчати в 2021 році[Електронний ресурс] – Режим доступу до ресурсу: https://dou.ua/lenta/articles/react-vs-vuejs/
17. JavaScript Component Suite for Responsive Web Development [Електронний ресурс] – Режим доступу до ресурсу: https://js.devexpress.com/
18. Як правильно оформити головну сторінку сайту [Електронний ресурс] – Режим доступу до ресурсу: https://hostiq.ua/blog/ukr/main-page/
19. Кольори для дизайну сайту та їх вплив на користувачів [Електронний ресурс] – Режим доступу до ресурсу: https://www.itk-agency.com/kolori-dlya-dizajnu-sajtu-ta-yih-vpliv-na-koristuvachiv/
20. Застосування золотого перерізу в Web [Електронний ресурс] – Режим доступу до ресурсу https://habr.com/ru/post/27491/

# ДОДАТОК 1. Впровадження роботи

Quipu
Part of the ProCredit Group

Representation of the Company "Quipu GmbH", 11 Krasylivska, 03150 Kyiv, Ukraine

**ВПРОВАДЖЕННЯ**
**кваліфікаційної роботи бакалавра на тему:**
**«Веб-інтерфейс для взаємодії та контролю системи документообігу»**

**студентки 4 курсу освітньо-професійної програми «Інформатика»**
**факультету комп'ютерних наук та кібернетики**
**Київського національного університету імені Тараса Шевченка**
**Лісової Марії**

Робота Лісової Марія присвячена проектуванню та розробці веб-інтерфейсу для взаємодії та контролю системи документообігу.

Наша система документообігу Mneme працює вже декілька років. З плином часу, кількість оброблюваних файлів значно збільшилась і, відповідно, почала зростати кількість запитів на створення веб-інтерфейсу. Вирішення цього питання було запропоновано знайтиМарії.

Були сформульовані чіткі завдання щодо розробки та явне бачення фінального результату. Марія була вільна у виборі технологій, але скористалась нашою порадою, обравши стек, з яким добре знайомі працівники нашої компанії. Крім того, ми рекомендували Марії для оформлення інтерфейсу використовувати пакет DevExtreme з метою практичного вивчення його можливостей з розробки нових single page додатків та розширення функціоналу та юзабіліті у існуючих.

Марія виконала всі поставлені та скореговані в процесі розробки завдання. Успішноосвоїла незнайому раніше їй технологію Angular, застосувавши отримані знання під час розробки веб-інтерфейсу. Уважно дослідила взаємодію Angular з пакетом DevExtreme та за допомогою компонент цієї бібліотеки створила зручне та мінімалістичне оформлення програми.

Наразі проект вже впроваджено в роботу співробітників компанії Quipu GmbH та успішно використовується. Крім того, потрібно зазначити, що розробка проекту продовжується, і Марія має завдання надалі покращувати створений веб-інтерфейс.

Project Coordinator Quipu GmbH

Кудінов Олексій

Alexey Kudinov
Digitally signed by Alexey Kudinov Date: 2022.06.09 18:31:14 +06'00'

Head of CustomWare.Net Development Department Quipu GmbH

Бойко Дмитро

Київ 2022

Representation of the Company "Quipu GmbH"
11 Krasylivska
03150 Kyiv, Ukraine

Tel.: +38 (0) 44 521 24 01
office_kiev@quipu.de
www.quipu.de

Представництво "КІПУ ГмбХ"
вул.Боженка 86
Київ, Україна 03150

Тел.: +38 (0) 44 521 24 01
office_kiev@quipu.de
www.quipu.de