**Міністерство освіти і науки України**

**Національний педагогічний університет імені М.П. Драгоманова**

Факультет інформатики

Кафедра Програмної інженерії

**ДИПЛОМНА РОБОТА БАКАЛАВРА**

пояснювальна записка

050103.73ІПЗ ПЗ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент гр. 41ПІ \_\_\_\_\_\_\_\_\_\_\_\_\_ Московченко Є.В.

Керівник роботи

Декан факультету інформатики \_\_\_\_\_\_\_\_\_\_\_\_\_ Франчук В.М.

Допускається до захисту

**Зав. кафедри, \_\_\_\_\_\_\_\_\_\_\_\_\_ Мучник М.М.**

2018 р.

ЗМІСТ

[ВСТУП 3](#_Toc515794172)

[1 аналіз предметної області та постановка задачі 4](#_Toc515794173)

[1.1 Основні поняття електронного документообігу 4](#_Toc515794174)

[1.2 Огляд аналогів системи електронного документообігу факультету 5](#_Toc515794175)

[2 ОПИС МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ 8](#_Toc515794176)

[2.1 Функціональні вимоги до розроблюваного програмного продукту 8](#_Toc515794177)

[2.2 Системні вимоги 9](#_Toc515794178)

[2.2.1 Мінімальні системні вимоги 9](#_Toc515794179)

[2.2.2 Рекомендовані системні вимоги 10](#_Toc515794180)

[2.3 Аналіз методів вирішення поставленої задачі 10](#_Toc515794181)

[2.4 Архітектура інформаційної системи 11](#_Toc515794182)

[2.4.1 UML діаграма прецедентів 11](#_Toc515794183)

[2.4.2 Генерація документів 16](#_Toc515794184)

[2.4.3 Компоненти системи 18](#_Toc515794185)

[2.5 Обрані програмні засоби 19](#_Toc515794186)

[2.5.1 GoogleSites 19](#_Toc515794187)

[2.5.2 Google Cloud Platform 20](#_Toc515794188)

[2.5.4 Скриптова мова JavaScript 20](#_Toc515794189)

[2.5.5 Мова розмітки HTML 21](#_Toc515794190)

[2.5.6 Таблиці стилів CSS 22](#_Toc515794191)

[2.6 План тестування розробленого програмного засобу. Керуючий граф програми 22](#_Toc515794192)

[3 АНАЛІЗ РЕЗУЛЬТАТІВ 28](#_Toc515794193)

[ВИСНОВКИ 36](#_Toc515794194)

[ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 37](#_Toc515794195)

# ВСТУП

У сучасному світі у організаціях різних типів та форм власності неможливо обійтись без документів. Збільшення навантаження на відповідні структурні підрозділи організацій призводить до того, що штат співробітників, які займаються документообігом треба постійно збільшувати, також необхідно розширювати архіви для зберігання готових документів. Гарним вирішенням таких проблем є переведення документообігу в електронний варіант, а з подальшим ростом масштабів подальша автоматизація документообігу. Постійний обіг паперів та їх кількість спонукає компанії, а також, навчальні заклади використовувати системи електронного документообігу.

Об’єкт дослідження: системи автоматизації електронного документообігу.

Предмет дослідження: створення системи електронного документообігу структурного підрозділу закладу вищої освіти.

Мета роботи: розробити власну систему електронного документообігу відповідно до потреб факультету.

Відповідно до поставленої мети, було сформульовано такі завдання:

1. проаналізувати існуючі системи електронного документообігу;
2. розробити структуру системи електронного документообігу структурного підрозділу закладу вищої освіти (факультету);
3. реалізувати систему електронного документообігу для подальшого її впровадження у діяльність підрозділу закладу вищої освіти (факультету);
4. провести тестування розробленої системи електронного документообігу структурного підрозділу закладу вищої освіти (факультету);
5. розробити супровідну документацію до розробленого програмного продукту.

# 1 аналіз предметної області та постановка задачі

## 1.1 Основні поняття електронного документообігу

Система автоматизації документообігу (Система електронного документообігу) — організаційно-технічна система, що забезпечує процес створення, управління доступом і поширення електронних документів в комп'ютерних мережах, а також що забезпечує контроль над потоками документів в організації.

Основними поняттями електронного документообігу є:

* документообіг — рух документів в організації з моменту їх створення або отримання до завершення виконання або відправлення; комплекс робіт з документами: прийом, реєстрація, розсилка, контроль виконання, формування справ, зберігання і повторне використання документації, довідкова робота;
* електронний документообіг (ЕДО) — єдиний механізм по роботі з документами, представленими в електронному вигляді, з реалізацією концепції «безпаперового діловодства»;
* електронний документ (ЕД) — документ, створений за допомогою засобів комп'ютерної обробки інформації, підписаний електронним цифровим підписом (ЕЦП) і збережений на машинному носієві у вигляді файлу відповідного формату;
* електронний цифровий підпис (ЕЦП) — аналог власноручного підпису, що є засобом захисту інформації, що забезпечує можливість контролю цілісності і підтвердження достовірності електронних документів.

Основні принципи електронного документообігу:

* однократна реєстрація документа, що дозволяє однозначно ідентифікувати документ в будь-якій інсталяції даної системи;
* можливість паралельного виконання операцій, що дозволяє скоротити час руху документів і підвищення оперативності їх виконання;
* безперервність руху документа, що дозволяє ідентифікувати відповідального за виконання документа (завдання) в кожен момент часу життя документа (процесу);
* єдина (або погоджено розподілена) база документної інформації, що дозволяє унеможливити дублювання документів;
* ефективно організована система пошуку документа, що дозволяє знаходити документ, володіючи мінімальною інформацією про нього;
* розвинена система звітності по різних статусах і атрибутах документів, що дозволяє контролювати рух документів по процесах документообігу і приймати управлінські рішення, ґрунтуючись на даних із звітів.

## 1.2 Огляд аналогів системи електронного документообігу факультету

Для початку було прийнято рішення проаналізувати існуючі системи документообігу і визначити, чи відповідають вони вимогам факультету Інформатики. Проведений аналіз виявив, що системи, які існують на ринку не відповідають в повному обсязі потребам факультету.

Порівняння деяких систем електронного документообігу наведено у таблиці 1.1:

Таблиця 1.1 Порівняння систем електронного документообігу

|  |  |  |  |
| --- | --- | --- | --- |
|  | FossDoc | СЕД (АПУ) | СЕД факультету |
| Особливості | * Система загального застосування * Орієнтована на корпоративний сектор * Платна * Закритий код | * Система загального застосування * Орієнтована на корпоративний сектор * Платна * Закритий код | * Вузькоспеціалізована система * Орієнтована на корпоративний сектор * Безкоштовна * Закритий код |
| Форма реалізації | * Додаток для ПК * Клієнт-серверна архітектура | * Додаток для ПК * Клієнт-серверна архітектура | * WEB-додаток * Клієнт-серверна архітектура |
| Переваги | * Технічна підтримка ПЗ * Постійне оновлення документної бази розробниками * «Особистого кабінету» * Реалізована з використанням технологій компанії Microsoft | * Технічна підтримка ПЗ * Постійне оновлення документної бази розробниками * Наявність «Особистого кабінету» * Реалізована з використанням технологій компанії Microsoft | * Технічна підтримка ПЗ * Можливість самостійно створювати необхідні шаблони документів та додавати їх у систему * Система займає мало місця * Для входу в систему потрібен тільки обліковий запис Google. * Наявність «Особистого кабінету» * Повністю реалізована на технологіях від компанії Google |
| Недоліки | * Встановлюється тільки на виділений-сервер * Необхідний потужний сервер | * Встановлюється тільки на виділений-сервер * Для встановлення необхідно мінімум 2 фізичних сервери (для SharePoint та бази даних) * Необхідний потужний сервер | * Встановлення відбувається в хмарних сервісах (корпоративних або загальнодоступних) * Відсутність можливості використання цифрового підпису документа |

Одним із найголовніших недоліків є необхідність інсталяції програмних засобів FossDoc та СЕД (АПУ) на користувацький пристрій грає дуже важливу роль, оскільки, в разі вибору будь-якої з цих систем, кожному студентові необхідно буде інсталювати ПЗ, по меншій мірі, на один власний пристрій.

Також FossDoc та СЕД (АПУ) потребує доволі потужний виокремлений сервер баз даних, а СЕД (АПУ) взагалі потребує 2 окремих сервера.

Крім того, ці системи вимагають купівлі ліцензії для користування нею. Також, FossDoc та СЕД (АПУ) більш спеціалізовані для використання в корпоративному секторі, що не дозволяє в повному обсязі задовольнити потреби факультету.

Серед переваг комерційних систем електронного документообігу можна виділити гарну технічну підтримку користувачів. Іншою перевагою є повна інтеграція з сервісами від компанії Microsoft. Всі системи документообігу дають користувачеві змогу використовувати «Особистий кабінет» з доступом до історії власних документів та результатом їх подачі у доволі зручній формі.

Отже, можна стверджувати, що існуючі на ринку комерційні системи електронного документообігу не задовольняють потреби факультету Інформатики, оскільки, факультет потребує специфічну систему з доступом лише через корпоративний обліковий запис Google. Для максимально безперебійної роботи систему необхідно розгорнути, використовуючи хмарні технології.

Отже, для вирішення цієї проблеми було вирішено спроектувати та реалізувати власну систему автоматизації документообігу з використанням хмарних технологій від компанії Google.

# 2 ОПИС МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ

## 2.1 Функціональні вимоги до розроблюваного програмного продукту

В ході проведення бесід з керівництвом факультету Інформатики Національного педагогічного університету імені Михайла Петровича Драгоманова було визначено функціональні вимоги до Системи електронного документообігу факультету.

Перелік функціональних вимог до системи електронного документообігу факультету:

* система має бути розгорнута з використанням технології Google Cloud Platform та системи Google Sites;
* доступ до системи повинен здійснюватися лише з використанням корпоративного облікового запису;
* у СЕД повинні бути наступні ролі користувачів: адміністратор, менеджер, звичайний користувач (студент, викладач);
* дані користувача (ім’я та прізвище) повинні автоматично отримуватись із облікового запису користувача в Google+;
* система повинна генерувати документ у форматі Google Docs із даними, якими користувач заповнив форму під час її подання;
* менеджер повинен мати можливість роздрукувати згенерований документ на відповідний запит користувача (студента, викладача);
* система повинна надсилати сповіщення про виконані дії (для студента/викладача – ваша заява прийнята, оброблена, відхилена, готова до отримання, для менеджера – студент/викладач подав заяву, для адміністратора – менеджер додав заяву) на їхню корпоративну електронну пошту;
* історія документів для користувача повинна мати фільтри для зручного перегляду поданих студентом заяв та їхні статусів;
* фон веб-системи повинен бути білий;
* кнопки мають бути синього кольору (відтінок royalblue) з білими літерами;
* всі написи мають бути читабельними на різних пристроях (персональні комп’ютери, ноутбуки, смартфони, планшети) з різними діагоналями дисплеїв;
* дизайн повинен бути адаптивним, коректно відображатись і працювати у різних браузерах (Firefox, Opera, Edge, Chrome, Safari) на різних пристроях з різними ОС.

Розміщення системи на платформі Google Sites, використання Google Drive та Google Mail дозволяє вирішити декілька головних проблем, наприклад, ці системи безкоштовні, а співпраця НПУ ім. М.П. Драгоманова з Google дозволяє університету отримувати доступ до розширених можливостей технологій G.Suite (Google Drive, Google Mail, Google Apps Script). Для входу в систему повинен використовуватись обліковий запис Google, оскільки, при використанні Google Sites, що значно спрощує інтеграцію. Кольорова гамма повинна, в цілому, повторювати кольору веб-сайту факультету Інформатики [X].

Веб сайт повинен коректно відображатись у наступних браузерах – Safari, Google Chrome, Mozilla Firefox, Opera, Microsoft Edge, Google Chrome (mobile), Safari (mobile), Opera (mobile), Firefox (mobile). Так як це основні браузери, що використовуються в операційних системах Windows (desktop), Android, iOS, то відображення в них обов’язково має бути правильним. Коректністю відображення на інших браузерах було вирішено знехтувати, оскільки, доступ до Яндекс.Браузеру заборонено законодавством України, а розповсюдженість інших браузерів (наприклад, UC Browser (mobile), TOR Browser, Vivaldi (desktop), Ghost Browser (desktop), AdBlock Browser (mobile), Konqueror, Dolphin) доволі мала, крім того, більшість з них використовує платформи інших браузерів (наприклад, браузер Vivaldi використовує платформу Chromium, як і Google Chrome, та Opera).

## 2.2 Системні вимоги

### 2.2.1 Мінімальні системні вимоги

В ході розробки і тестування було визначено, що для доступу та роботи з застосуванням користувачеві потрібні мінімум наступні характеристики.

Мінімальні вимоги до клієнтського ПК:

Браузер: Chrome 50.0 / MS Edge 9.0 / Firefox 50.0 / Opera 48.0.

Мінімальні вимоги до мобільних пристроїв:

ОС: Android 4.4.4 KitKat / iOS 10.

### 2.2.2 Рекомендовані системні вимоги

Для стабільної та безперебійної роботи з застосуванням користувачам рекомендується мати пристрої з наступними параметрами.

Рекомендовані вимоги до клієнтського ПК:

Браузер: Chrome 57.0 / MS Edge 12.0 / Firefox 57.0 / Opera 49.0.

Рекомендовані вимоги до мобільних пристроїв:

ОС: Android 8.1 Oreo / iOS 11.

## 2.3 Аналіз методів вирішення поставленої задачі

Для реалізації поставленої задачі було проведено аналіз різних методів, які можна було б застосувати для реалізації задачі.

Для цього було проаналізовано розробку:

* веб-застосування на мові PHP (серверна частина) з інтеграцією до Google Sites (за необхідності);
* веб-застосування на мові ASP.NET (серверна частина) з інтеграцією до Google Sites (за необхідності);
* веб-застосування на мові Google Apps Script з інтеграцією до Google Sites;
* застосування для мобільних пристроїв на базі Android, iOS та для персональних комп’ютерів, що працюють під керуванням операційної системи Microsoft Windows, використавши при цьому клієнт-серверну архітектуру.

Клієнт-серверне застосування має суттєвий недолік — необхідна інсталяція програмного продукту на сервер для забезпечення функціонування системи, під час технічних, чи технічно-профілактичних робіт на сервері функціонування системи неможливе. Крім того, необхідно розробити клієнтську частину для декількох ОС: як мінімум, Microsoft Windows, Apple iOS, Google Android. Випуск оновлень також потребує відлагодження застосування також для цих трьох ОС.

Розгорнуте на власних серверах веб-застосування потребує постійного обслуговування роботи сервера, а в разі технічних робіт, система не може працювати. Але при такому підході необхідно розробити лише одну версію, на відміну від клієнт-серверної архітектури системи.

Використання ASP.NET загалом можливе, але це потребує додаткових витрат на ліцензію за використання технологій .NET.

Використання мови PHP є доволі гарною ідеєю, оскільки потрібно розробити лише одну версію застосування (з веб-інтерфейсом), розгорнути застосування можна на зовнішньому орендованому хостингу, або на власному університетському сервері. Проблема використання PHP, у порівнянні з GAS на Google Sites, є складність імплементації бібліотеки авторизації Google OAuth 2.0, інтеграція сервісів Google, таких як: Google Drive, Google Mail, Google Docs.

Крім того, використання Google Sites та GAS дозволяє повністю відмовитись від необхідності використання хостингу, що зменшує витрати на утримання системи, позбавляє необхідності проводити технічне обслуговування сервера.

Враховуючи співпрацю НПУ ім. М.П. Драгоманова з компанією Google, доволі часте використання студентами корпоративного облікового запису від Google і широкі можливості технологій Google та інтеграцію цих сервісів між собою, а також відсутність потреби розробляти окремі додатки під декілька операційних систем і відсутність необхідності постійно обслуговувати роботу сервера, створення системи електронного документообігу факультету на технологіях Google є найкращим варіантом.

## 2.4 Архітектура інформаційної системи

### 2.4.1 UML діаграма прецедентів

UML use-case діаграму для акторів Студент та Викладач показано на рисунку Х:



Рисунок Х – UML діаграма прецедентів для Студента та Викладача

Актори-користувачі Студент та Викладач можуть:

* відправляти форму (заяву);
* переглядати історію власних поданих заяв;
* переглядати статуси власних поданих заяв;
* переглядати категорії меню.

UML use-case діаграму для актора Менеджер показано на рисунку Х:



Рисунок Х – UML діаграма прецедентів для Менеджера

Актор-користувач Менеджер має можливість:

* відправляти форму (заяву);
* переглядати історію власних поданих заяв;
* переглядати статуси власних поданих заяв;
* переглядати категорії меню;
* опрацьовувати подані студентами заяви (форми);
* обирати заяву для опрацювання;
* приймати заяву;
* роздруковувати документи;
* відхиляти заяви;
* вказувати причину;
* видаляти існуючі документи (форми);
* обирати документ для видалення;
* додавати документи (форми) для подання;
* обирати призначення для форми;
* обирати категорію форми;
* обирати форму розміщену на Google Drive;
* обирати шаблон документа для форми.

UML use-case діаграму для актора Адміністратор показано на рисунку Х:



Рисунок Х – UML діаграма прецедентів для Адміністратора

Актор-користувач Адміністратор має можливість:

* відправляти форму (заяву);
* переглядати історію власних поданих заяв;
* переглядати статуси власних поданих заяв;
* переглядати категорії меню;
* опрацьовувати подані студентами заяви (форми):
* обирати заяву для опрацювання;
* приймати заяву:
* роздруковувати документи;
* відхиляти заяви;
* вказувати причину;
* видаляти існуючі документи (форми);
* обирати документ для видалення;
* додавати документи (форми) для подання;
* обирати призначення для форми;
* обирати категорію форми;
* обирати форму розміщену на Google Drive;
* обирати шаблон документа для форми;
* керувати користувачами;
* переглядати адміністраторів системи;
* керувати менеджерами;
* переглядати закріплені за менеджерами форми;
* переглядати менеджерів системи;
* додавати нових менеджерів;
* вказувати E-Mail нового менеджера;
* змінювати налаштування системи;
* переглядати існуючі категорії заяв;
* додавати нові категорії;
* додавати домен та статус;
* переглядати поточну кількість спроб;
* задавати нову кількість спроб;
* видаляти систему.

Можливості актора Система показано на рисунку Х:



Рисунок Х – UML діаграма прецедентів для Системи

Актор Система має можливість:

* генерувати Google документ;
* перевіряти введені користувачем дані у форму;
* працювати з файлами (читання / запис);
* перевіряти обліковий запис користувача;
* визначати ім’я;
* визначати прізвище;
* визначати кількість спроб, що залишилась на поточний день;
* визначати права;
* генерувати пункти меню відповідно до прав користувача.

### 2.4.2 Генерація документів

Алгоритм генерації документів показано на рисунку Х:



Рисунок Х – алгоритм генерації документів

Після надходження відповіді на конкретну форму від користувача у таблиці відповідей змінюються статус обробки у менеджера та користувача. Якщо користувач не має власний каталог у відповідній директорії системи, йому створюється каталог, який буде застосовуватись для подальшого зберігання відповідей даного користувача на форми у системі. Для створення каталогу генерується унікальне ім’я, потім до глобальної таблиці файлів заносяться наступні дані про каталог: id, назва, тип каталогу (МІМЕ), автор, дата створення каталогу.

Коли користувач має власний каталог, проводиться перевірка на наявність у вищезгаданому каталозі іншого, для зберігання відповідей форми. За його відсутності, такий створюється (з використанням генерації унікального ім’я та занесенням даних про каталог (id, назва, тип (МІМЕ), автор, дата створення) до глобальної таблиці файлів).

Потім виконується пошук шаблону документа закріпленого за даною формою. Після цього виконується пошук полів підстановки в документі та заміна їх відповідями з форми (на місце для вставки імені вставляється ім’я і т. д.).

Коли всі поля підстановки буде замінено у відповідний каталог (який іменується «id\_відповіді»).

В кінці користувачеві на електронну пошту відправляється лист-сповіщення про початок обробки заяви менеджером.

### 2.4.3 Компоненти системи

Структура розміщення компонентів системи електронного документообігу показана на рисунку Х:



Рисунок Х – діаграма розміщення СЕД

Система розміщується у каталозі, який задається адміністратором при встановленні системи. В кореневому каталозі (“[root]”) міститься файл (“FilesList”) у форматі таблиць Google (“[G.Sheets]”), який зберігає перелік всіх файлів встановленої системи. Там же містяться каталог налаштувань системи : “Settings” з файлами “Domain”, “General”, “Managers”, “Translate”. Всі у форматі таблиць Google (“[G.Sheets]”).

Каталог “Documents” містить шаблони документів для подання через систему та документи, подані користувачами (а також згенеровані документи на заяви користувачів). У каталозі “Users” створюється каталог з назвою, що відповідає електронній пошті користувача, що хоча б раз подавав заяви через систему, або є менеджером системи. У цьому каталозі зберігається файл з історією користувача, що має назву, ідентичну до назви каталогу (електронна пошта користувача), у форматі таблиць Google (“[G.Sheets]”). Якщо користувач є менеджером, то у даному каталозі буде створено файл з історією заяв, що були подані на документ, доданий цим менеджером (файл “manager\_[UserName@Mail]” формату Google таблиць — “[G.Sheets]”).

## 2.5 Обрані програмні засоби

### 2.5.1 GoogleSites

Система була розроблена на платформі Google Sites, з використанням мов програмування Google Apps Script та Javascript, а також мови розмітки HTML та стильової мови CSS.

Сайти Google (англ. Google Sites) — спрощений безкоштовний хостинг на базі вікі-рушія. Може використовуватися як частина Google Apps. Дозволяє за допомогою технології wiki зробити інформацію доступною для людей, які потребують її швидкої подачі. Користувачі сайту можуть працювати разом, додавати інформацію з інших додатків Google, наприклад Документи Google, Календар Google, YouTube, Picasa та з інших джерел.

Творець сайту може запрошувати інших користувачів для спільної роботи над сайтом, контролювати їх доступ до матеріалів. Сайт може бути використаний в private режимі, наприклад, для організації особистого wiki блокнота або для ведення записів по приватному проекту з доступом до інформації тільки після авторизації.

Служба «Сайти Google» замінила застарілу Google Page Creator.

### 2.5.2 Google Cloud Platform

Google Apps Script (GAS)- це скриптова мова для розробки невеликих додатків на платформі Google Apps. Вона заснована на JavaScript 1.6 з деякими частинами 1.7 і 1.8 і забезпечує підмножина API ECMAScript 5, однак замість того, щоб працювати в клієнті, вона запускається в Google Cloud. За даними Google, Google Apps Script "надає прості способи автоматизації завдань у продуктах Google та сторонніх сервісах". Apps Script також є інструментом, який надає додаткам полегшений доступ до Документів, Таблиць та Презентацій Google.

Особливості GAS:

* створена на основі JavaScript; легко вчиться;
* відлагоджувач Cloud дозволяє виконувати налагодження сценаріїв додатків безпосередньо у веб-браузері;
* може бути використана для створення простих інструментів внутрішнього споживання організації;
* може бути використана для виконання простих системних завдань;
* підтримка та розвиток здійснюється з допомогою спільноти користувачів.

### 2.5.4 Скриптова мова JavaScript

Деяку частину графічного інтерфейсу користувача (front-end) та деякий набір функцій серверної частини (back-end) було реалізовано за допомогою скриптової мови JavaScript. Використання різноманітних бібліотек (як то Bootstrap, JQuery, AngularJS) не було необхідним, оскільки базових можливостей мови було цілком достатньо.

JavaScript (JS) — динамічна, об'єктно-орієнтована мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується для створення сценаріїв веб-сторінок, що надає можливість на стороні клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд веб-сторінки.

JavaScript класифікують як прототипну (підмножина об'єктно-орієнтованої), скриптову мову програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми програмування (імперативну та частково функціональну) і деякі відповідні архітектурні властивості, зокрема: динамічна та слабка типізація, автоматичне керування пам'яттю, прототипне наслідування, функції як об'єкти першого класу.

### 2.5.5 Мова розмітки HTML

Для створення розмітки веб застосування було вирішено використати мову розмітки HTML, оскільки вона є найпоширенішою для таких потреб і її використання забезпечує вирішеннях всіх задач, для яких вона призначена.

HTML (англ. HyperText Markup Language — мова розмітки гіпертекстових документів) — стандартна мова розмітки веб-сторінок в Інтернеті. Більшість веб-сторінок створюються з допомогою мови HTML (або XHTML). Документ HTML оброблюється браузером та відтворюється на екрані у звичному для людини вигляді.

HTML є похідною мовою від SGML, успадкувавши від неї визначення типу документа та ідеологію структурної розмітки тексту.

Попри те, що HTML — штучна комп'ютерна мова, вона не є мовою програмування.

HTML разом із каскадними таблицями стилів та вбудованими скриптами — це три основні технології побудови веб-сторінок.

HTML впроваджує засоби для:

* створення структурованого документа шляхом позначення структурного складу тексту: заголовки, абзаци, списки, таблиці, цитати та інше;
* отримання інформації із Всесвітньої мережі через гіперпосилання;
* створення інтерактивних форм;
* включення зображень, звуку, відео, та інших об'єктів до тексту.

### 2.5.6 Таблиці стилів CSS

Використання мови розмітки HTML є недоцільним, оскільки сайт буде мати не дуже привабливий вигляд і розташування елементів на сторінці неможливо змінити, окрім як, розташовувати елементи один під одним. Для надання сайтам кращого вигляду застосовують каскадні таблиці стилів – CSS.

Каскадні таблиці стилів (англ. Cascading Style Sheets або скорочено CSS) — спеціальна мова, що використовується для опису зовнішнього вигляду сторінок, написаних мовами розмітки даних.

Найчастіше CSS використовують для візуальної презентації сторінок, написаних HTML та XHTML, але формат CSS може застосовуватися до інших видів XML-документів.

Специфікації CSS були створені та розвиваються Консорціумом Всесвітньої мережі.

CSS має різні рівні та профілі. Наступний рівень CSS створюється на основі попередніх, додаючи нову функціональність або розширюючи вже наявні функції. Рівні позначаються як CSS1, CSS2 та CSS3. Профілі — сукупність правил CSS одного або більше рівнів, створені для окремих типів пристроїв або інтерфейсів. Наприклад, існують профілі CSS для принтерів, мобільних пристроїв тощо.

## 2.6 План тестування розробленого програмного засобу. Керуючий граф програми

Для тестування коректності роботи можливостей системи і розробки плану тестування було складено керуючий граф програми. Згідно графу було проведено тестування (детальніше описано у розділі 3), яке показало, що керуючий граф програми покриває практично всі можливі шляхи виконання програми, по меншій мірі, всі критично важливі шляхи, використання системи електронного документообігу без яких взагалі не має сенсу.

Керуючий граф програми – відображає потік керування програми. Це граф G(V,A), де V(V1,… Vm) – множина вершин (операторів), A(A1,… An) – множина дуг (керування), що з’єднують вершини.

Шлях – послідовність вершин та дуг керуючого графу програми, в якій будь-яка дуга виходить із вершини Vi і приходить в вершину Vj.

Гілка – шлях (V1, V2, … Vk), де V1 – або перший, або умовний оператор, Vk – або умовний оператор, або оператор виходу з програми, а всі інші оператори – безумовні. Гілки – лінійні частини програми, їхня кінцева кількість.

Кількість шляхів в програмі може бути необмежено (шляхи, що різняться хоча б одним числом проходження циклу – різні).

Існують можливі та неможливі шляхи в програмі, в неможливі шляхи в звичайних умовах потрапити не можна[X].

Керуючий граф програми зображено на рисунку Х:



Рисунок Х – Керуючий граф програми

Можливі шляхи виконання програми:

1. 1-2;
2. 1-3;
3. 1-4;
4. 1-5;
5. 1-6;
6. 1-7;
7. 1-8;
8. 1-9;
9. 1-10;
10. 1-11;
11. 1-12;
12. 1-20-39-40-41-42;
13. 1-13-15-22;
14. 1-13-15-23-49;
15. 1-13-15-23-50;
16. 1-13-16-24-54;
17. 1-13-16-24-51-54;
18. 1-13-16-24-52-54;
19. 1-13-16-24-53-54;
20. 1-13-16-25-55-56;
21. 1-13-16-26-57-58;
22. 1-13-16-27-59-60;
23. 1-13-16-27-59-61;
24. 1-13-17-28-29-30-31;
25. 1-13-18-32-33;
26. 1-13-18-34-35;
27. 1-13-19-36-37;
28. 1-13-19-36-38;
29. 1-13-20-39-40-41-42;
30. 1-13-21-43-44-45-46-47;
31. 1-13-21-48;
32. 1-13-14-15-22;
33. 1-13-14-15-23-49;
34. 1-13-14-15-23-50;
35. 1-13-14-16-24-54;
36. 1-13-14-16-24-51-54;
37. 1-13-14-16-24-52-54;
38. 1-13-14-16-24-53-54;
39. 1-13-14-16-25-55-56;
40. 1-13-14-16-26-57-58;
41. 1-13-14-16-27-59-60;
42. 1-13-14-16-27-59-61;
43. 1-13-14-17-28-29-30-31;
44. 1-13-14-18-32-33;
45. 1-13-14-18-34-35;
46. 1-13-14-19-36-37;
47. 1-13-14-19-36-38;
48. 1-13-14-20-39-40-41-42;
49. 1-13-14-21-43-44-45-46-47;
50. 1-13-14-21-48.

Пункти керуючого графу:

1. Авторизація;
2. Остання активність сайту;
3. Поскаржитись;
4. Друк сторінки;
5. Видалити право доступу;
6. Google Sites;
7. Подати заяву;
8. Зворотній зв’язок із розробниками;
9. Редагувати сторінку;
10. Додати сторінку;
11. Налаштування сайту;
12. Відкрити доступ;
13. Відкрити меню;
14. Всі категорії;
15. Керування користувачами;
16. Налаштування системи;
17. Додати документ;
18. Опрацювання документів;
19. Видалити документ;
20. Подати документ;
21. Історія документів;
22. Перелік менеджерів, перелік закріплених форм, Перелік адміністраторів системи;
23. Додати нового менеджера;
24. Додати домен та статус;
25. Змінити кількість спроб;
26. Додати категорію;
27. Видалити систему;
28. Обрати призначення;
29. Обрати категорію;
30. Обрати форму;
31. Обрати шаблон документа;
32. Відхилити;
33. Вести причину;
34. Опрацювати;
35. Готово;
36. Обрати;
37. Підтвердити;
38. Відмінити;
39. Обрати форму;
40. Заповнити дані;
41. Надіслати;
42. Надіслати ще один документ;
43. Розгорнути фільтри;
44. Обрати документ;
45. Обрати кількість записів;
46. Обрати статус;
47. Застосувати;
48. Очистити історію;
49. Менеджера успішно додано;
50. Помилка додавання менеджера;
51. Додати ще одну пару «Домен-Статус»
52. Прибрати останню додану пару;
53. Очистити введені у поля дані
54. Підтвердити додавання;
55. Ввести нову кількість спроб;
56. Підтвердити зміну;
57. Ввести назву категорії
58. Підтвердити додавання;
59. Видалити систему;
60. Підтвердити видалення;
61. Скасувати видалення.

Таким чином, після бесід з керівництвом факультету було вирішено спроектувати власну систему документообігу, яка б використовувала платформу Google Cloud Platform. Для оформлення дизайну системи було вирішено використати стандартну зв’язку HTML та CSS. Крім того, було складено керуючий граф програми для подальшого його використання при проведенні тестування програми.

# 3 АНАЛІЗ РЕЗУЛЬТАТІВ

В ході розробки системи електронного документообігу факультету було проведено тестування можливостей та коректності роботи системи.

Тестування – процес проведення керованих експериментів з програмним продуктом за допомогою виконання тестів з метою виявлення в ньому невідповідностей, згідно з технічним завданням.

Тест – контрольне завдання для перевірки коректності роботи функцій системи, або модулів цієї системи.

«Вдалим» тестом вважається такий, при котрому виконання програми закінчилось з помилкою і навпаки. Тестування виконує дві основні задачі [X]:

демонстрація якості функціонування ПЗ;

знаходження і усунення помилок в ПЗ.

Головною метою тестування є збільшення ймовірності того, що ПЗ, яке проходить тестування, буде відповідати вимогам.

Тестування – це ітераційний процес. Після того, як виявлено та виправлено помилку обов’язково слідує повторення тестів, що має на меті перевірити працездатність програми. Більш того, для ідентифікації причини виявленої проблеми може бути потрібно проведення спеціального додаткового тестування. При цьому завжди необхідно пам’ятати фундаментальний висновок, що зробив професор Едсгером Дейкстрою в 1972 р.: «Тестування програм може служити доказом наявності помилок, але ніколи не доведе їх відсутність!» [X].

Структурне тестування, також називають тестуванням за принципом «білої скриньки» або «скляної скриньки», полягає у перевірці внутрішньої структури елементів системи.

Основним видом тестування є функціональне. Функціональне тестування застосовують для програмного забезпечення у цілому, а також для програмних об’єктів будь-якого рівня (процедури, модулі, підсистеми, системи). Структурне тестування доповнює функціональне. Цей вид тестування можливий для рівня не вище рівня програмного модуля. Виконання функціонального і структурного тестування системи може бути здійснене незалежно одне від одного.

Структурне тестування програмного забезпечення може бути реалізоване такими методами:

* тестуванням маршрутів;
* тестуванням циклів;
* тестуванням обробки даних.

Перш за все було протестовано можливість додавання менеджерів. Для цього необхідно мати права адміністратора (показано на UML діаграмі прецедентів для актора Адміністратор на рис. Х). Це шляхи керуючого графа (рис. Х) №14 та №33. Для того, щоб додати менеджера потрібно відкрити меню, перейти до пункту «Керування користувачами» (також можливо перейти до пункту «Всі категорії» і обрати «керування користувачами»), розгорнути пункт «Керування менеджерами», розгорнути пункт «Додати нового менеджера». У полі з підписом «E-mail нового менеджера» потрібно записати електронну пошту нового менеджера, якого необхідно додати. Потім натиснути кнопку «Додати» у з’явившомуся діалоговому вікні підтвердити отримання результату про успішне додавання, або про помилку роботи системи (всі дії послідовно показано на рис. Х):

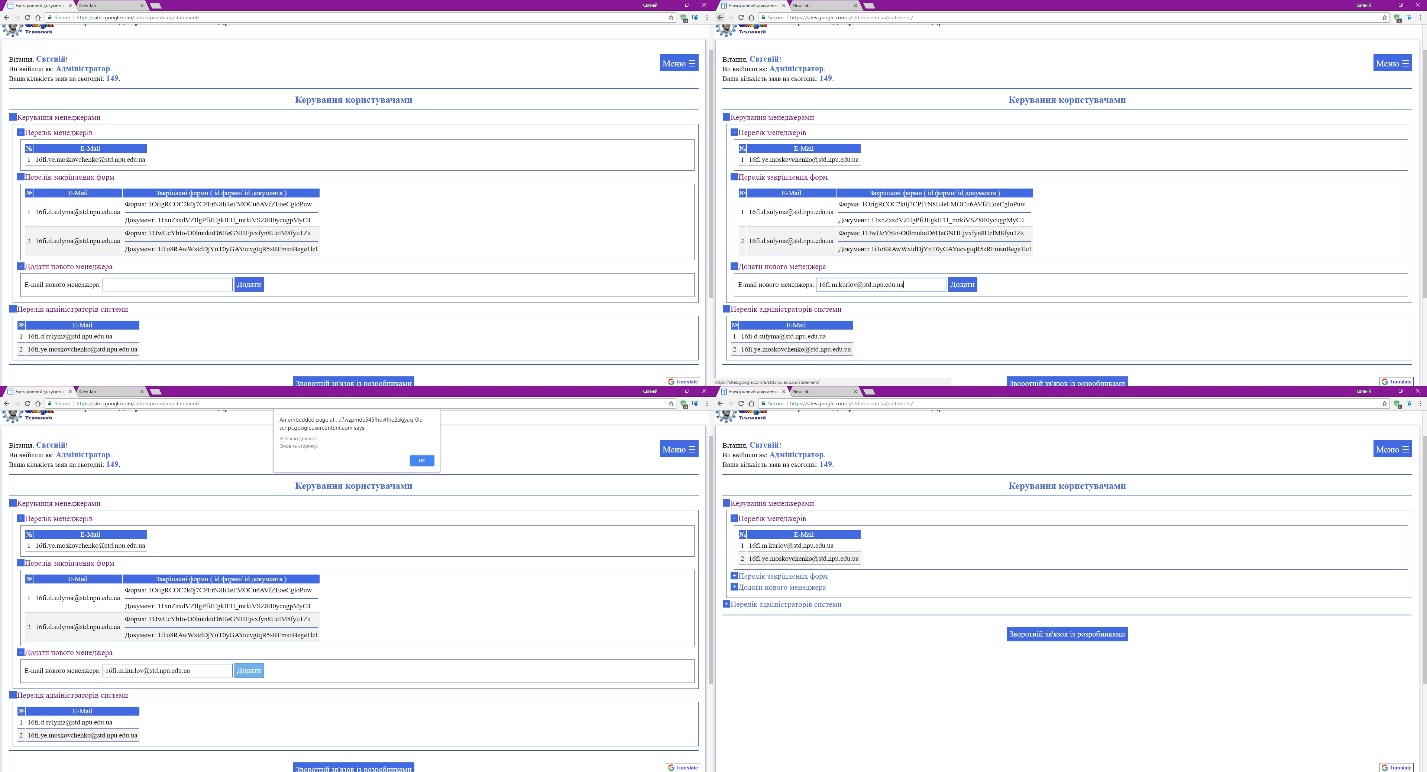


Рисунок Х – додавання нового менеджера

Далі було протестовано можливість додавання доменів та статусів. Ця дія також доступна лише адміністраторам системи (рис. Х). На керуючому графі програми це шляхи №№16, 17, 18, 19, 35, 36, 37, 38.

Для додавання домену, необхідно відкрити меню, перейти до пункту «Налаштування системи» (або перейти до пункту «Всі категорії» і обрати «Налаштування системи»), розгорнути пункт Статуси. Далі необхідно ввести у відповідні поля («Домен» та «Статус») потрібні значення і натиснути підтвердити. Кнопка «+» додає ще одну пару полів «Домен-Статус» для вводу, кнопка «–» видаляє останню додану полів. Кнопка «Очистити» очищує всі введені дані у поля вводу. Послідовність дій показана на рисунку Х.

Додавання категорій на керуючому графі показано шляхами №21 та №40.

Щоб додати категорію потрібно перейти до налаштувань системи («Меню – Налаштування системи», або «Меню – Всі категорії – Налаштування системи»), розгорнути пункт «Категорії». У полі з підписом «Додати категорію» ввести назву нової категорії та натиснути кнопку «Підтвердити». Послідовність дій для додавання категорій показано на рисунку Х:

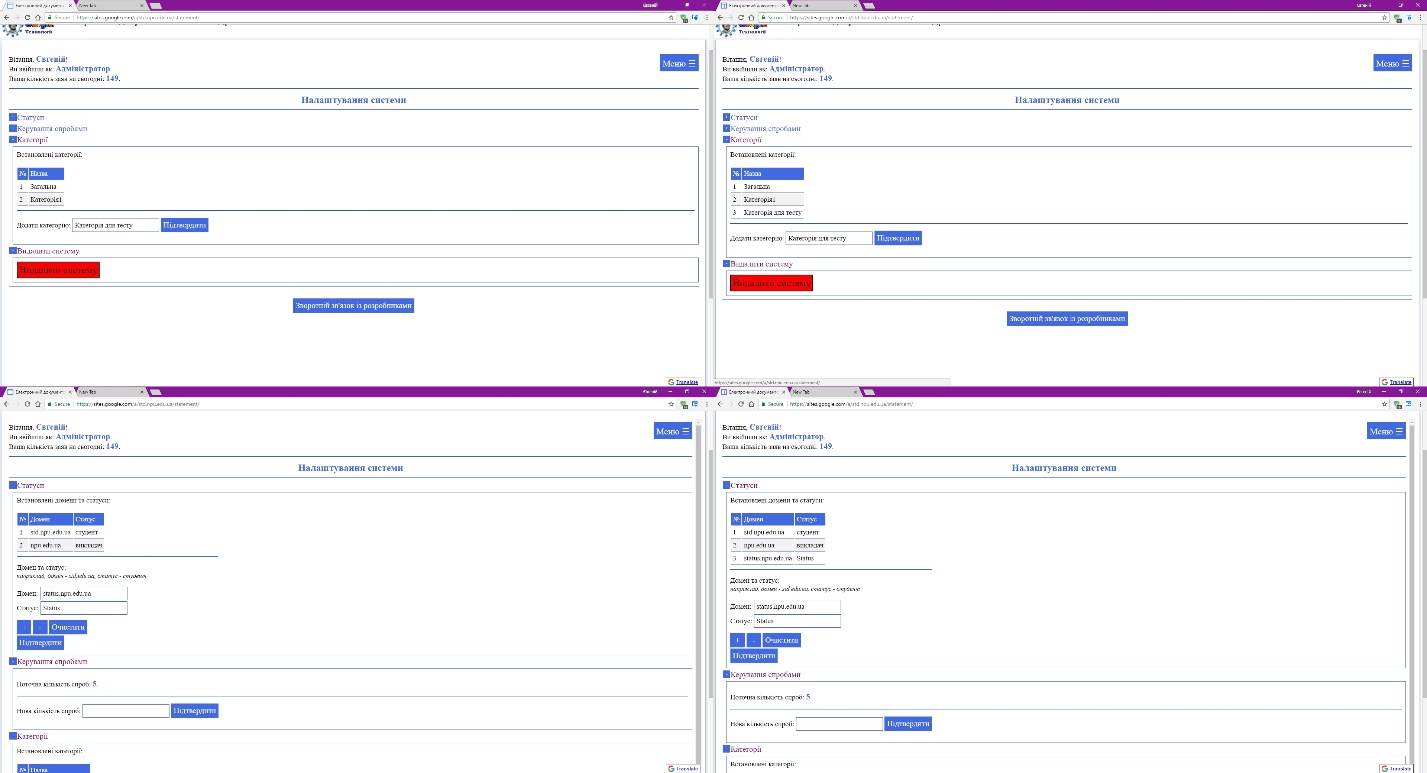


Рисунок Х – додавання категорій та доменів

Користувачі Адміністратор та Менеджер можуть додавати шаблони документів для подання іншими користувачами (рис. Х, рис. Х). На керуючому графі це шляхи №24 та №43 (рис. Х).

Для додавання шаблонів необхідно перейти до відповідної сторінки. Для цього потрібно відкрити меню, обрати «Додати документ» (або відкрити меню, обрати «Всі категорії», потім обрати «Додати документ»). Далі із випадаючих списків потрібно обрати призначення документа (призначення є існуючі статуси у системі), категорію. Також потрібно обрати форму та шаблон документа для генерації, які повинні бути попередньо завантажені користувачем на власний Google Drive. Після того необхідно натиснути кнопку «Надіслати». Послідовність маніпуляцій зображено на рисунку Х:

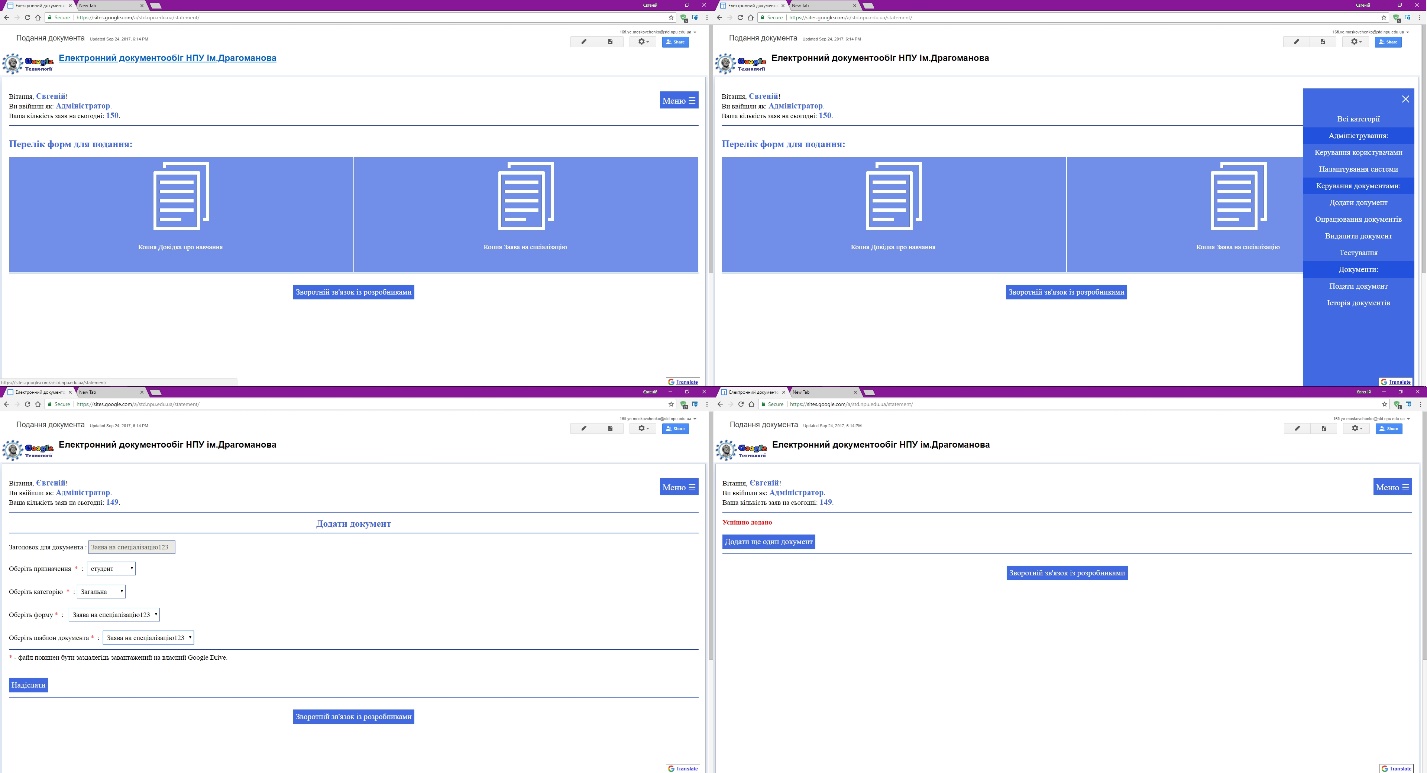


Рисунок Х – додавання документу для подання

Менеджер та адміністратор мають можливість опрацьовувати заяви, подані на додані ними шаблони документів. Для цього потрібно, відкривши меню, перейти до пункту «Опрацювання документів» (або відкрити меню, обрати «Всі категорії», перейти до «Опрацювання документів»).

При наявності ще не оброблених заяв, вони будуть виділені жовтим кольором. Натискання на рядок заяви відобразить більш детальну інформацію про заяву та дає можливість повернутись назад, відхилити заяву, або згенерувати документ. При натисканні «Згенерувати документ» буде виведено повідомлення про створення документа. У разі успішної генерації документа буде виведено посилання на цей файл, а статус заяви буде змінено із «В черзі» на «В обробці». Потім користувачеві необхідно повернутись назад, до переліку документів, обрати цей же документ і натиснути кнопку готово. У поле, що з’явилося, можна написати примітку, яку буде відправлено у повідомленні на електронну пошту користувачеві, що подавав документ. При натисканні кнопки «Підтвердити», статус документа буде змінено на «Готово», а користувачеві буде відправлено повідомлення про успішне створення документа та примітка, при наявності. Послідовність дій показано на рисунках Х та Х:

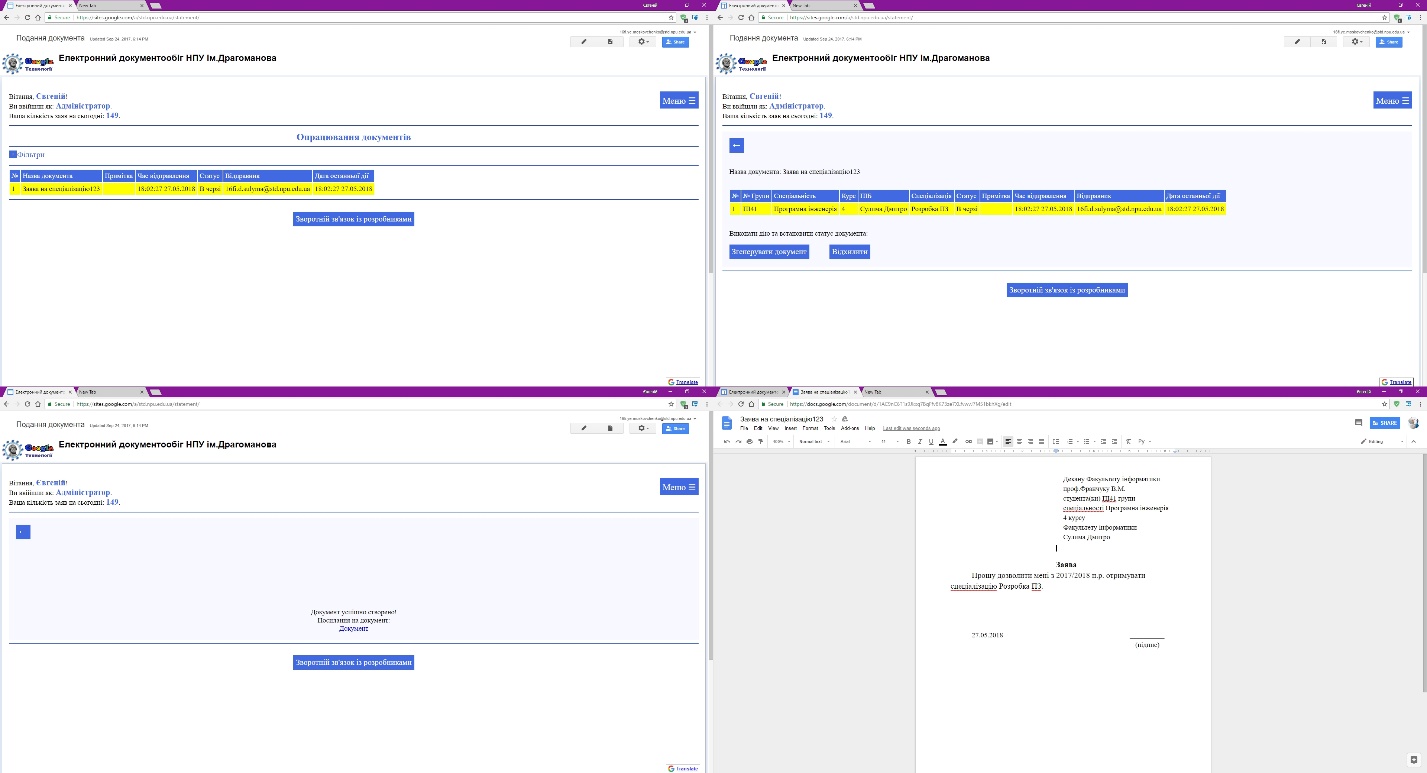


Рисунок Х – генерація документа

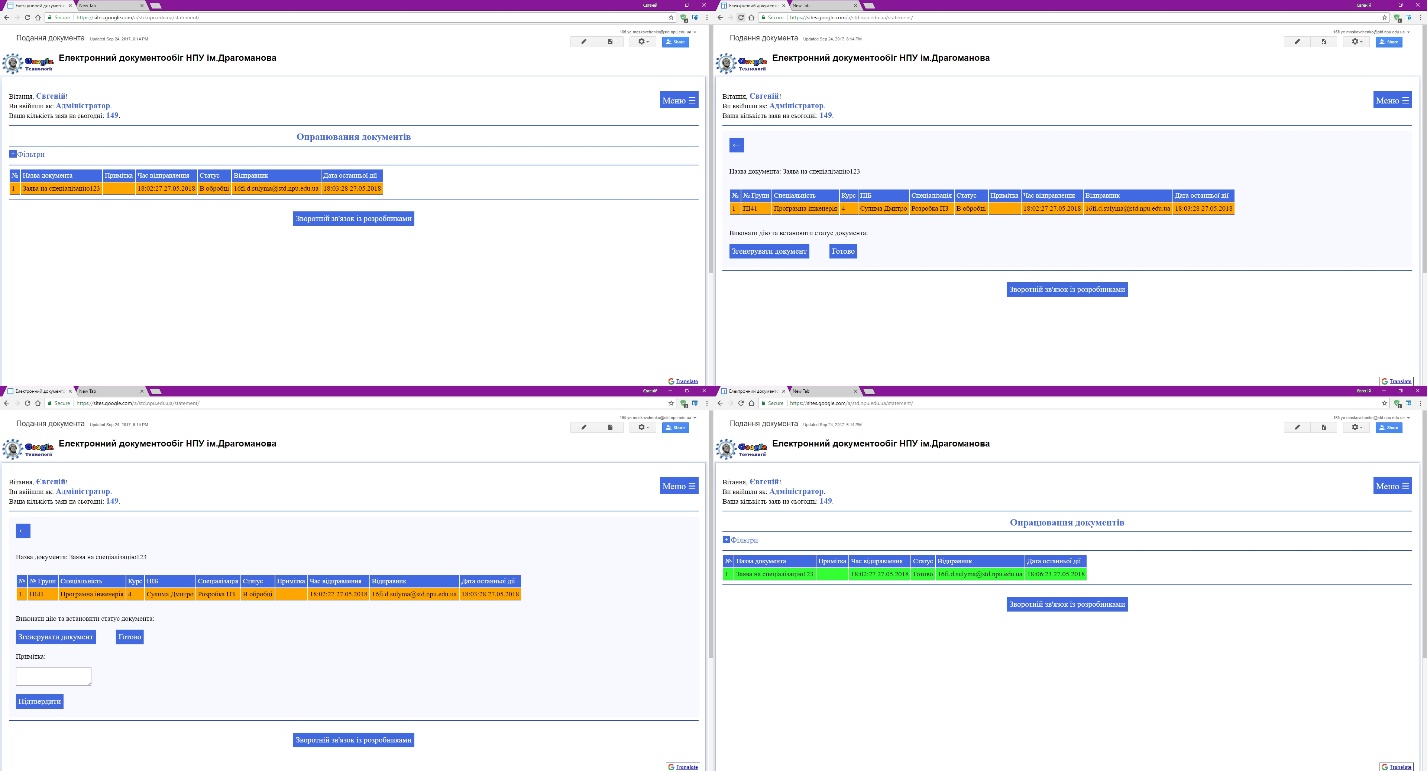


Рисунок Х – підтвердження поданої заяви

Користувач може видаляти, додані ним, шаблони документів. Для цього потрібно відкрити меню, обрати пункт «Видалити документ» (або відкрити меню, перейти до «Всі категорії», перейти до «Видалити документ»). Обрати документ для видалення, натиснувши на відповідний рядок таблиці. Користувач може видаляти лише додані ним документу. Далі потрібно підтвердити видалення у діалоговому вікні. Після декількох секунд обробки запиту серверами буде виведено повідомлення про успішне видалення документа (або, навпаки, про помилку) і прохання оновити сторінку. Послідовність дій показано на рисунку Х:

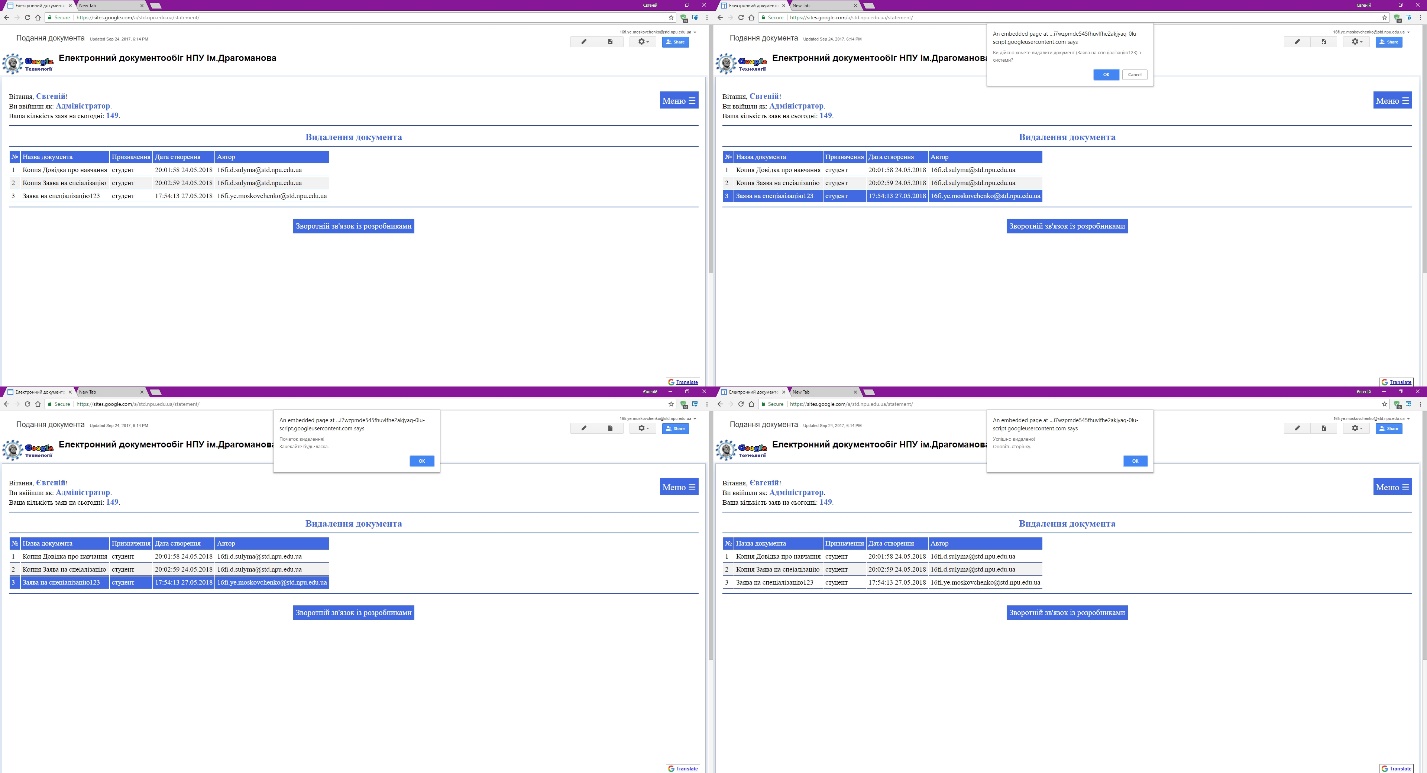


Рисунок Х – видалення доданого документа

Всі користувачі (адміністратори, менеджери та інші, в залежності від доданих доменів) можуть подавати заяви на шаблони, які були додані менеджерами, або адміністраторами. Для подачі документа потрібно обрати документ на головній сторінці (це можна зробити також відкривши меню, обрати «Подати документ», або відкрити меню, обрати «Всі категорії», перейти до пункту «Подати документ»). Потім потрібно заповнити необхідні поля (обов’язкові для заповнення поля відмічено червоними зірочками), при бажанні, можна вказати примітку. Після заповнення полів потрібно натиснути кнопку «Надіслати». У разі успіху буде показано повідомлення про успішне виконання запиту. Послідовність маніпуляцій показано на рисунку Х:

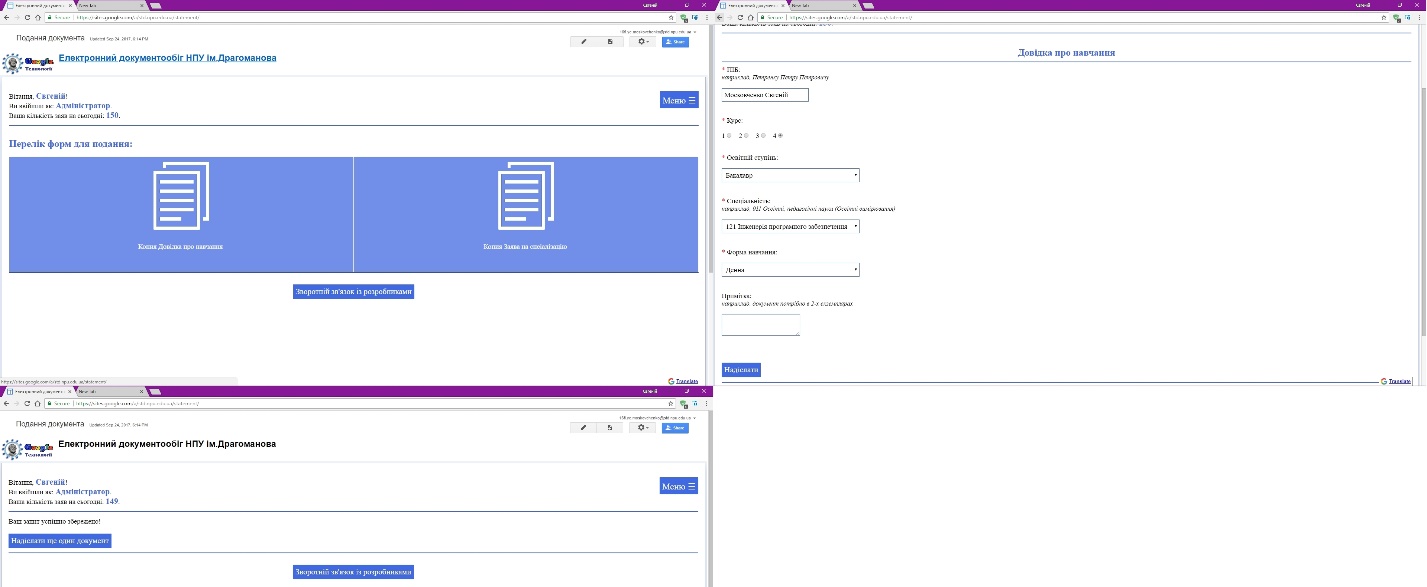


Рисунок Х – подання заяви

Будь-який із користувачів може переглянути історію. Для цього потрібно відкрити меню, обрати пункт «Історія документів» (або, відкривши меню, перейти до «Всі категорії» і обрати «Історія документів»). У таблиці буде показано подані даним користувачем заяви. Жовтий колір означає статус «В черзі», оранжевий – «В обробці», зелений – «Готово», тобто документ готовий до видачі, червоний – «Документ» відхилено. Алгоритм дій для перегляду історії поданих документів зображено на рисунку Х:

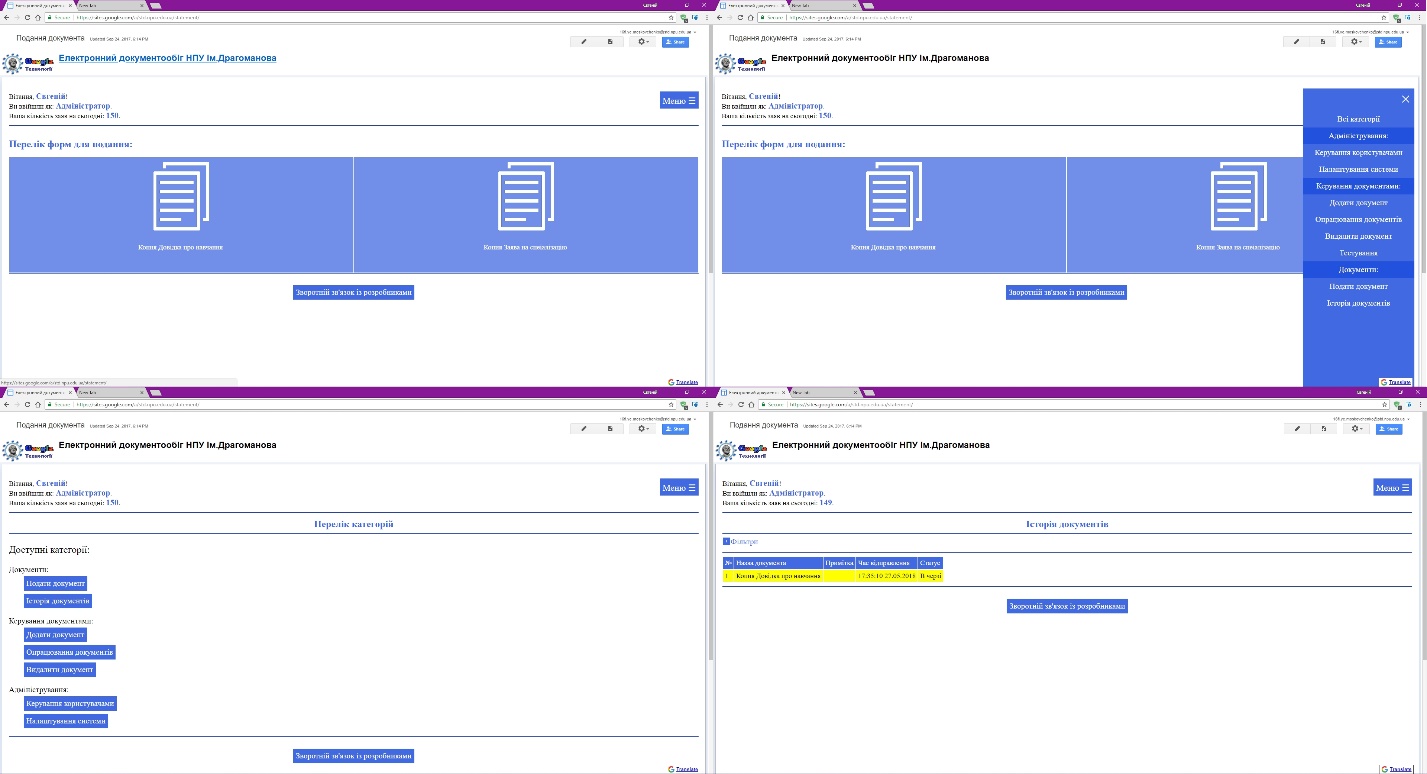


Рисунок Х – перегляд історії поданих документів

В ході тестування було встановлено, що функції системи виконуються згідно вимог. Критичних помилок на даному етапі виявлено не було. Прийнято рішення почати підготовку до розгортання на факультеті Інформатики Національного педагогічного університету ім. М.П. Драгоманова.

# ВИСНОВКИ

В ході аналізу існуючих систем електронного документообігу було встановлено, що жодна з існуючих систем не відповідає вимогам факультету. Тому було прийнято рішення розробити власну СЕД. Для цього при проектуванні системи необхідно було враховати вимоги факультету, які було визначено в ході бесід з керівництвом факультету. Систему було вирішено розгорнути на платформі Google Cloud Platform, з використанням Google Sites, Google Drive, Google Mail та мови Google Apps Script, оскільки в такому разі відпадає необхідність утримання окремого хостингу для роботи системи. Доступ до системи надається лише власникам корпоративного облікового запису Google факультету Інформатики, оскільки саме цей обліковий запис використовується студентами факультету для входу в освітню систему Moodle [X].

Тестування показало, що функції системи виконуються коректно, система готова до впровадження на факультеті інформатики.

Після впровадження на факультеті Інформатики систему планується поширювати на інші факультети Національного педагогічного університету ім М.П. Драгоманова. В разі, якщо система покаже себе успішно в НПУ ім. Драгоманова, можливо буде запропонувати розгортання системи іншим навчальним закладам.

В подальшому планується розширення функціоналу системи, як то збільшення кількості параметрів для налаштувань, збільшення кількості ролей користувачів. В разі значного успіху системи планується вивчити можливість інтеграції штучного інтелекту для аналізу заяв користувачів. Така можливість дозволить зменшити кількість менеджерів для обробки заяв, оскільки це буде виконуватись системою автоматично.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Веб-ресурс: http://www.eos.com.ua/
2. [http://www.intalev.ua/](http://www.intalev.ua/ua/services/bp/)
3. <https://techexpert.ua/vprovadzhennya-sistem-elektronnogo-dokumentoob%D1%96gu-v%D1%96d-adm%D1%96n%D1%96strats%D1%96-prezidenta-ukraini/>
4. <http://fi.npu.edu.ua/>
5. <https://studfiles.net/preview/4350057/page:5/#14>
6. Липаев В.В. Тестирование программ [Текст] / В.В. Липаев. – М.: Радио и связь, 1986. – 296 с.
7. Дейкстра Дисциплина программирования [Текст] / Э.Дейкстра; пер. с англ. И. Х. Зусман ; ред. Э. З. Любимский. – М.: Мир, 1978. – 275 с. – (Математическое обеспечение).
8. Основные принципы веб-дизайна и их характеристики [Electronic resource] / www.designonstop.com. – 2018. – Mode of access : http://www.designonstop.com/webdesign/article/osnovnye-principy-veb-dizajna-i-ix-xarakteristiki.htm.
9. ДСТУ 2844-94. Програмні засоби ЕОМ. Забезпечення якості. Терміни та визначення. – Введ. 1.08.1995.
10. ДСТУ2850-94. Програмні засоби ЕОМ. Показники і методі оцінювання якості. – К.: Держстандарт України, 1994.
11. ISO/IEC 9126. 2001 Software Engineering – Software Product quality – Geneva, Switzerland: International Organization for Standartization.
12. Веб-ресурс: https://fossdoc.com/
13. Веб-ресурс: http://infoplus.ua/