

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

ДЛЯ ПІДТРИМКИ УПРАВЛІННЯ РОБОЧИМИ ПРОЦЕСАМИ

SOFTWARE TO SUPPORT

WORKFLOW MANAGEMENT

Виконав(ла): студент(ка) 4 курсу, групи КНТ-112сп

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

МОРМУЛЬ Є.О.

(ПРИЗВИЩЕ та ініціали)

Керівник ІЛЬЯШЕНКО М.Б.

(ПРИЗВИЩЕ та ініціали)

Рецензент ШИТІКОВА О.В.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2025 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

МОРМУЛЯ Євгенія Олеговича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення для підтримки управління робочими процесами. Software to Support Workflow Management

керівник проєкту (роботи) к.т.н., доцент, ІЛЬЯШЕНКО Матвій Борисович,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 28 ” квітня 2025 року № 209

2. Строк подання студентом проєкту (роботи) 03 червня 2025 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів) _____

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ІЛЪЯШЕНКО М.Б., доцент		
Нормоконтроль	КАМІНСЬКА Ж.К., асистент		

7. Дата видачі завдання “ 28 ” квітня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка структури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділи 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

(підпис) Євгеній МОРМУЛЬ
(Імя ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Матвій ІЛЪЯШЕНКО
(Імя ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
84 с., 3 табл., 34 рис., 3 дод., 18 джерел.

ANGULARJS, JAVASCRIPT, МОВА ПРОГРАМУВАННЯ,
ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, РОБОЧИЙ ПРОЦЕС, ФРЕЙМВОРК.

Об'єкт дослідження – процес розробки програмного забезпечення для підтримки управління робочими процесами.

Предмет дослідження – програмні засоби для підтримки управління робочими процесами.

Мета роботи – розробка програмного забезпечення для підтримки управління робочими процесами.

Матеріали, методи та технічні засоби: мова програмування JavaScript, фреймворк AngularJS.

Результати. Розроблено проєктні рішення для створення програмного забезпечення для підтримки управління робочими процесами. Створено програмне забезпечення для підтримки управління робочими процесами. Проведено тестування розробленої програмної системи для підтримки управління робочими процесами.

Висновки. Мету роботи досягнуто. Розроблено програмне забезпечення для підтримки управління робочими процесами за допомогою мови програмування JavaScript та фреймворку AngularJS.

Галузь використання – управління робочими процесами підприємства.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 84 pages, 3 tables, 34 figures, 3 appendixes, 18 sources.

ANGULARJS, JAVASCRIPT, PROGRAMMING LANGUAGE, SOFTWARE, WORKFLOW, FRAMEWORK.

The object of research is the process of developing software to support workflow management.

The subject of the research is software for supporting workflow management.

The purpose of this work is to develop the software to support workflow management.

Materials, methods and technical tools: JavaScript programming language, AngularJS framework.

Results. Project solutions for the creation of software to support workflow management has been designed. The software to support workflow management has been developed. The developed software system to support workflow management has been tested.

Conclusions. The goal of the work has been achieved. The software tools to support workflow management using the JavaScript programming language and the AngularJS framework have been developed.

Scope of use – workflow management.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ	9
1 Аналіз предметної області	11
1.1 Програмне забезпечення для підтримки управління робочими процесами	11
1.2 Аналіз програмного забезпечення для підтримки управління робочими процесами	21
1.3 Висновки за розділом 1	30
2 Матеріали і методи	32
2.1 Вибір мови програмування	32
2.2 Вибір фреймворка для створення програмного забезпечення для підтримки управління робочими процесами	34
2.3 Висновки за розділом 2	38
3 Опис програми	39
3.1 Структура програмного забезпечення для підтримки управління робочими процесами	39
3.2 Функціонування програмного забезпечення для підтримки управління робочими процесами	41
3.3 Особливості реалізації програмного забезпечення для підтримки управління робочими процесами	46
3.4 Проєктування інтерфейсу програмного забезпечення для підтримки управління робочими процесами	47
3.5 Висновки за розділом 3	51
4 Експлуатація, тестування та експериментальне дослідження програми	53
4.1 Призначення й умови застосування програми	53
4.2 Характеристики програми для підтримки управління робочими процесами	54
4.3 Інструкція по експлуатації програми	55

4.3.1	Звернення до програми.....	55
4.3.2	Вхідні й вихідні дані.....	55
4.3.3	Повідомлення.....	56
4.4	Виконання програмного забезпечення для підтримки управління робочими процесами.....	56
4.5	Тестування програмного забезпечення для підтримки управління робочими процесами.....	59
4.6	Висновки за розділом 4	60
	Висновки.....	61
	Перелік джерел посилань.....	64
	Додаток А Технічне завдання.....	66
	Додаток Б Фрагмент тексту програми	70
	Додаток В Слайди презентації	78

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

CRM – Customer Relationship Management;

ERP – Enterprise Resource Planning;

БД – база даних;

ПЗ – програмне забезпечення.

ВСТУП

Програмне забезпечення для підтримки управління робочими процесами стає дедалі актуальнішим через зростаючу складність бізнес-процесів, необхідність автоматизації та оптимізації операційної діяльності. Воно дозволяє значно підвищити ефективність, автоматизуючи рутинні завдання, зменшуючи кількість помилок і покращуючи продуктивність завдяки стандартизації процесів [1], [2].

Сучасні системи управління процесами забезпечують гнучкість і адаптивність, дозволяючи швидко вносити зміни відповідно до нових вимог ринку та підтримуючи різні методології управління. Крім того, вони оснащені потужними аналітичними інструментами, які допомагають відстежувати ефективність процесів, виявляти проблемні місця та знаходити шляхи для їх оптимізації [3], [4].

Інтеграція з іншими системами, такими як системи управління ресурсами підприємства (Enterprise Resource Planning, ERP) та системи управління взаємовідносинами з клієнтами (Customer Relationship Management, CRM), значно розширює можливості організацій, дозволяючи створювати єдине інформаційне середовище. Також такі програмні рішення допомагають дотримуватись внутрішніх регламентів і зовнішніх нормативних вимог, що особливо важливо для компаній, які працюють у регульованих сферах [5], [6].

Останні технологічні тренди, зокрема використання штучного інтелекту та машинного навчання, хмарних платформ, роблять ці програмні системи ще більш доступними, масштабованими та зручними у впровадженні. Таким чином, програмне забезпечення для управління робочими процесами стає ключовим інструментом для компаній, які прагнуть підвищити ефективність, зменшити витрати та залишатися конкурентоспроможними в динамічному бізнес-середовищі [1]-[6].

Попри значні переваги, програмне забезпечення для підтримки управління робочими процесами має і певні недоліки. Одним із головних викликів є висока вартість та складність впровадження таких програмних систем. Ще одним недоліком програмних засобів для підтримки управління робочими процесами є можливі труднощі з інтеграцією, оскільки не всі програмні рішення легко поєднуються з уже використовуваними корпоративними системами, що може спричинити проблеми з обміном даними та призвести до затримок у роботі. Не менш важливою проблемою є залежність від постачальника програмного забезпечення підтримки управління робочими процесами. Якщо розробник програмного забезпечення для підтримки управління робочими процесами припиняє супровід або змінює модель ліцензування, компанія може зіткнутися з труднощами у підтримці та масштабуванні своїх процесів [1]-[6].

Тому актуальною є розробка програмного забезпечення для підтримки управління робочими процесами. У дипломній кваліфікаційній роботі бакалавра розв'язується актуальне завдання розробки програмного забезпечення для підтримки управління робочими процесами.

Для досягнення поставленої мети у кваліфікаційній роботі бакалавра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для підтримки управління робочими процесами;
- здійснити проєктування програмного забезпечення для підтримки управління робочими процесами;
- створити програмне забезпечення для підтримки управління робочими процесами;
- виконати тестування розробленого програмного забезпечення для підтримки управління робочими процесами.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Програмне забезпечення для підтримки управління робочими процесами

Забезпечення ефективного управління робочими процесами потребує використання спеціалізованого програмного забезпечення, яке дозволяє автоматизувати та координувати виконання завдань. Використання таких технологій сприяє підвищенню продуктивності та забезпечує безперервний контроль за всіма етапами процесів. Завдяки інтеграції з іншими інформаційними системами створюється єдине середовище для обміну даними та синхронізації дій між учасниками [1].

Організація робочих процесів вимагає систематичного підходу до планування, розподілу завдань і контролю виконання. Для досягнення поставлених цілей необхідно забезпечити чітку координацію всіх етапів роботи, визначити відповідальних за виконання та встановити реалістичні терміни. Процес управління передбачає безперервний аналіз поточного стану справ, оцінку ефективності прийнятих рішень і внесення коригувань у разі необхідності [1].

Значну роль відіграє використання спеціалізованого програмного забезпечення, що дозволяє автоматизувати багато аспектів управління, підвищити продуктивність і забезпечити прозорість у виконанні завдань. Використання таких рішень сприяє оптимальному розподілу ресурсів, підвищенню відповідальності учасників і зниженню ризиків, пов'язаних із можливими затримками або помилками [1].

Ефективна організація робочого процесу ґрунтується на постійному моніторингу виконання завдань, аналізі отриманих результатів і впровадженні покращень, що сприяють підвищенню якості виконуваних робіт. Завдяки сучасним підходам до управління забезпечується стабільний розвиток, можливість швидкого реагування на зміни та досягнення стратегічних цілей [1].

У процесі організації та контролю робочих завдань важливо забезпечити послідовність дій, зосередженість на найважливіших аспектах і ефективне використання доступних можливостей. Для досягнення результатів необхідно чітко визначати черговість виконання робіт, враховуючи їхню значущість та актуальність. Використання спеціальних підходів дозволяє розподіляти навантаження оптимальним чином, що сприяє дотриманню запланованих термінів та узгодженості всіх процесів [1].

Моніторинг виконання є невід'ємною частиною організаційного процесу. Завдяки поетапному аналізу вдається контролювати досягнення проміжних результатів і своєчасно виявляти можливі труднощі. Для ефективної координації застосовуються інструменти, що дозволяють відстежувати стан виконання завдань, оцінювати їхній прогрес і приймати обґрунтовані рішення щодо подальших дій [1].

Рациональне використання наявних ресурсів забезпечує стабільність виконання поставлених завдань. Важливо правильно розподіляти матеріали, фінансові можливості та робочий час, уникаючи надмірного навантаження та нераціональних витрат. Взаємодія між учасниками робочого процесу сприяє досягненню спільних цілей. Організація ефективного обміну інформацією, узгодження дій та спільне обговорення поточних питань дозволяють підвищити продуктивність і покращити якість виконуваних робіт [1].

Використання спеціалізованих програмних інструментів дозволяє значно підвищити ефективність спільної роботи над завданнями, забезпечуючи прозорість процесів та контроль за виконанням. Завдяки цьому можна систематизувати інформацію про всі етапи роботи, встановлювати чіткі терміни та оперативно коригувати дії відповідно до змін у проєкті. Автоматизація рутинних операцій зменшує витрати часу на ручне адміністрування та полегшує координацію між усіма учасниками [1].

Формування чіткої структури управління дозволяє розподіляти ресурси відповідно до поточних потреб, забезпечуючи своєчасне виконання ключових завдань. Завдяки впорядкованому підходу до планування вдається

контролювати послідовність дій, уникати хаосу та забезпечувати злагоджену взаємодію всіх учасників [1].

Для впровадження ефективної системи керування необхідно спочатку визначити основні завдання та потреби, а також оцінити, які інструменти можуть найкраще відповідати поставленим цілям. Ознайомлення з доступними технологічними рішеннями дозволяє вибрати платформу, що відповідає вимогам команди, враховуючи її функціональність та інтеграцію з іншими засобами [1].

Після вибору відповідного інструменту важливо розробити оптимальну систему роботи, яка передбачає створення структури завдань, визначення ролей учасників та встановлення чітких правил взаємодії. Розподіл відповідальності між учасниками та надання їм доступу до актуальної інформації сприяє кращій організації та зменшенню ймовірності помилок [1].

Для ефективного використання необхідно забезпечити навчання всіх залучених осіб, ознайомивши їх з можливостями нового підходу та правилами його застосування. Надання інструкцій, проведення тренінгів та підтримка обміну досвідом сприяють швидшій адаптації до нової системи [1], [2].

Процес управління потребує регулярного аналізу та коригування, що дозволяє вчасно реагувати на зміни в умовах виконання проєкту. Постійний моніторинг прогресу та внесення необхідних коректив забезпечують стабільність роботи та сприяють досягненню поставлених цілей у визначені терміни [1], [2].

Процес управління задачами охоплює координацію дій, розподіл ресурсів і контроль виконання завдань. Планування, визначення пріоритетів і оптимізація роботи дозволяють досягати поставлених цілей. У межах цього підходу забезпечується організація роботи над задачами, що дає змогу ефективно розподіляти обов'язки, відстежувати хід виконання завдань і підтримувати узгодженість у команді [1], [2].

Для організації роботи використовуються спеціальні інструменти, що сприяють структурованому підходу до виконання задач. Вони допомагають

координувати дії всіх учасників, забезпечувати контроль за строками виконання, підтримувати ефективну комунікацію та оптимізувати розподіл навантаження. З їхньою допомогою спрощується управління процесами, мінімізуються ризики втрати важливої інформації та підвищується продуктивність [1], [2].

Застосування таких інструментів дозволяє зосереджуватися на першочергових завданнях, уникати хаосу та ефективно розподіляти ресурси. Спрощений доступ до інформації, можливість швидкого внесення змін і прозорість процесів сприяють ефективній роботі команди. Відстеження прогресу та аналіз виконаних завдань допомагають вчасно виявляти проблеми й ухвалювати правильні рішення [1], [2].

Функціонал таких платформ спрямований на спрощення співпраці: коментування, обмін файлами, актуалізація статусів та інтеграція з іншими сервісами дозволяють злагоджено виконувати роботу. Внаслідок цього досягається узгодженість дій, покращується взаємодія між учасниками задачу та підвищується загальна ефективність роботи [1], [2].

Ефективна організація роботи команди дозволяє покращити координацію та оптимізувати виконання завдань. Завдяки впровадженню відповідних підходів до управління досягається чіткий розподіл обов'язків, що сприяє узгодженості в діях та підвищенню продуктивності. Наявність єдиного середовища для планування та відстеження ходу виконання завдань допомагає уникнути плутанини та забезпечує прозорість у роботі [2], [3].

Правильна організація робочого процесу дозволяє ефективно використовувати доступні ресурси, що знижує непродуктивні витрати та сприяє рівномірному розподілу навантаження. Чітко визначені ролі та відповідальність кожного учасника сприяють більшій дисциплінованості та допомагають досягати поставлених цілей у встановлені терміни. Автоматизація процесів та скорочення рутинних операцій дозволяють зосереджуватися на пріоритетних завданнях, що значно прискорює виконання роботи [2], [3].

Використання сучасних технологій у керуванні задачами допомагає структурувати роботу та створює сприятливі умови для ефективної взаємодії. Чітке формулювання завдань та визначення ключових етапів дозволяє уникати затримок і контролювати прогрес. Розподіл завдань відповідно до компетенцій команди забезпечує раціональне використання людських ресурсів і сприяє стабільному досягненню результатів [2], [3].

Постійний моніторинг процесів дає змогу оперативно виявляти труднощі та вживати заходів для їх усунення. Регулярний аналіз виконаної роботи сприяє вдосконаленню підходів до управління та дозволяє використовувати накопичений досвід для підвищення ефективності в майбутньому [2], [3].

Процес організації роботи над завданнями передбачає системний підхід та використання спеціалізованого програмного забезпечення, спрямованого на ефективне планування та реалізацію намічених дій, що забезпечує впорядкованість та контроль на всіх етапах виконання. Оптимальне керування завданнями є основою високої продуктивності, незалежно від того, чи йдеться про виконання особистих справ або про участь у масштабних спільних проектах [3], [4].

Кожне завдання проходить через певні етапи, що потребують різних методів і підходів. Вміння правильно організувати ці процеси сприяє підвищенню ефективності та дотриманню встановлених термінів. Використання гнучких стратегій дозволяє адаптуватися до змін і забезпечує послідовне досягнення запланованих результатів [3], [4].

Сучасні програмні інструменти організації роботи створені для універсального застосування, оскільки допомагають оптимізувати управління як особистими справами, так і великими проектами. Аналіз особливостей кожного етапу виконання дає змогу правильно розподілити ресурси та налаштувати процеси відповідно до конкретних вимог. Раціональний підхід до планування дозволяє зменшити витрати часу та підвищити якість виконання завдань [3], [4].

Керування окремими завданнями та управління проектами мають відмінності у своїй структурі. Організація роботи над конкретними завданнями зосереджується на контролі та виконанні окремих дій у повсякденному житті або професійній діяльності. Водночас управління проектами охоплює комплексні процеси, які мають чітко визначений початок, завершення та орієнтовані на досягнення конкретних результатів із врахуванням очікувань зацікавлених сторін [3], [4].

Організація роботи над завданнями передбачає проходження кількох ключових етапів. Спочатку завдання формується, після чого передається на виконання. Далі відбувається його реалізація з поступовим контролем якості та аналізом ефективності. У деяких випадках завдання може бути переглянute, змінене або навіть скасоване. Ефективність такого процесу значною мірою залежить від здатності до адаптації та своєчасного реагування на зміну обставин. Гнучкість в управлінні завданнями дозволяє знаходити оптимальні рішення, що сприяють стабільному досягненню поставлених цілей [4], [5].

Основою успішного керування завданнями є послідовне проходження важливих етапів, що забезпечують ефективність і продуктивність у роботі підприємства. Часто нехтування цими ключовими аспектами ускладнює досягнення бажаного результату, тоді як уважне дотримання відповідних методів сприяє значному спрощенню процесу та швидшому виконанню поставлених цілей [4], [5].

Раціональна організація роботи передбачає зосередження уваги на найбільш важливих завданнях. Чітке визначення пріоритетів дає змогу сконцентрувати ресурси на ключових аспектах, відкладаючи другорядні питання на пізніший термін. Завдяки цьому вдається уникнути зайвих затримок та зберегти баланс між виконанням основних завдань і додаткових обов'язків [4], [5].

Постійний контроль за перебігом роботи дозволяє своєчасно виявляти проблеми та коригувати стратегію виконання. Аналізуючи отримані результати, можна визначити слабкі місця та внести необхідні зміни, щоб

уникнути повторення помилок. Такий підхід не лише підвищує ефективність, а й додає мотивації, оскільки забезпечує чітке уявлення про прогрес і сприяє подальшому вдосконаленню процесів [4], [5].

Заздалегідь розроблений план допомагає оптимально розподілити час і ресурси для виконання кожного завдання. Чітке розуміння термінів і обсягу робіт дає змогу уникнути хаотичності та спрощує оцінку проміжних і кінцевих результатів [4], [5].

Відсутність системного підходу до виконання завдань призводить до втрати організованості та неузгодженості в роботі. Завдяки впровадженню ефективних методів керування завданнями процеси набувають чіткої структури, що сприяє швидкому та якісному виконанню роботи. Раціональне управління дозволяє зменшити витрати часу, забезпечити логічний зв'язок між окремими етапами та підвищити загальну продуктивність [4], [5].

Процес керування завданнями не має жодних обмежень у використанні, оскільки підходить будь-кому, хто прагне ефективніше організовувати свою діяльність. Особливо корисним він є для тих, хто має велику кількість обов'язків і потребує чіткої системи для їх виконання. У ситуаціях, коли навантаження є мінімальним, необхідність у такому підході зменшується, проте кожна людина стикається із завданнями, які потребують організованості [4], [5].

Керування завданнями активно використовується тими, хто займається координацією великих обсягів роботи. Воно відіграє важливу роль у забезпеченні ефективності, дозволяючи злагоджено виконувати обов'язки та уникати хаосу в процесі роботи. Завдяки впровадженню чіткої системи стає можливим краще розподіляти навантаження та координувати дії різних учасників робочого процесу [4], [5].

Підтримка контролю над завданнями за допомогою спеціалізованого програмного забезпечення сприяє підвищенню продуктивності, оскільки усуває зайві елементи, що можуть уповільнювати робочий процес. Раціональне використання наявного часу дає змогу оперативніше досягати

результатів та уникати затримок. Крім того, впровадження такої методики дозволяє знизити рівень стресу, що позитивно позначається як на професійній діяльності, так і на загальному самопочутті [4], [5].

Розумне управління дозволяє краще використовувати час, адже його втрата є незворотною. Багато труднощів у роботі виникають через неефективне розподілення ресурсів, а завдяки структурованому підходу ці втрати мінімізуються. Оптимізація часу дає можливість швидше переходити до нових завдань, що підвищує загальну результативність роботи [4], [5].

Ще однією важливою перевагою є покращення здатності вирішувати проблеми. Можливість поділу складних завдань на дрібніші етапи значно спрощує їх виконання, дозволяючи застосовувати творчий підхід. Крім того, спрощення структури роботи допомагає ухвалювати рішення швидше, залишаючи більше простору для зосередження на критично важливих питаннях [4], [5].

Для ефективного впровадження управління завданнями можуть використовуватися як спеціальні інструменти, так і певні техніки. Деякі з них вимагають застосування програмного забезпечення, інші ж залежать від індивідуальних вподобань і можуть змінюватися відповідно до потреб. Усі ці засоби сприяють організації роботи, оптимізації планування та підвищенню ефективності виконання поставлених цілей [4], [5].

Серед основних інструментів, що допомагають керувати завданнями, можна виділити класичні списки, які використовуються для ведення покупок, планування дня та виконання основних справ. Простота такого підходу робить його незмінно популярним, і саме з нього починається більшість сучасних систем управління завданнями. Візуальні дошки допомагають структурувати роботу, розставляючи пріоритети завдяки наочному представленню інформації. Календарі дозволяють контролювати терміни виконання справ, встановлювати нагадування та організовувати робочий процес. Використання спеціального програмного забезпечення надає доступ до хмарних рішень, які

дають змогу легко планувати робочі процеси та досягати поставлених цілей [4], [5].

Крім програмних інструментів, ефективність управління завданнями значною мірою залежить від методів, що застосовуються в роботі. Досягти більшої продуктивності можна, якщо навчитися уникати справ, що не мають високого пріоритету. Це стосується як особистих, так і професійних обов'язків, що відволікають від головних завдань. Усунення зайвих відволікаючих факторів, таких як соціальні мережі чи телевізор, допомагає зберегти зосередженість та оптимізувати час. Важливим аспектом є також правильний розподіл навантаження: доцільно починати з менших завдань, поступово переходячи до складніших, уникаючи надмірного перевантаження. Одночасне виконання завдань, хоч і здається ефективним способом роботи, насправді лише знижує загальну продуктивність, оскільки увага розпорошується, а якість виконання знижується. Встановлення чітких інтервалів між періодами активної роботи та відпочинку дозволяє підтримувати стабільний темп виконання завдань без перевтоми [4], [5].

Управління завданнями залишається одним із ефективних способів організації як робочого, так і особистого життя. Використовуючи відповідні методи та інструменти, можна значно підвищити ефективність виконання справ, оптимізувати витрати часу та зменшити рівень стресу, пов'язаного з надмірним навантаженням [4], [5].

Процес управління завданнями передбачає розподіл загального обсягу роботи на окремі етапи, їх координацію та постійний контроль за виконанням. Основна мета полягає в тому, щоб оптимально розподілити завдання відповідно до компетенції виконавців, забезпечити раціональне використання часу та ресурсів, а також гарантувати ефективну співпрацю в команді. Важливо своєчасно визначати пріоритетність дій, щоб кожен етап реалізації проекту відбувався відповідно до актуальних потреб. Крім того, управління завданнями сприяє постійному вдосконаленню робочих процесів і забезпечує швидке виконання поставлених цілей із мінімальними витратами [5], [6].

Ефективне керування завданнями дозволяє зосереджувати увагу на найважливіших етапах роботи саме тоді, коли це необхідно. У процесі реалізації проекту пріоритети можуть змінюватися, тому важливо швидко адаптуватися до нових умов. Завдяки цьому вдається уникнути неефективного витрачання ресурсів і сфокусуватися на ключових аспектах [5], [6].

Ще однією перевагою є можливість отримання детальної аналітичної інформації про перебіг виконання завдань. Візуалізація поточного стану проекту дозволяє оцінити динаміку роботи, структурувати процеси та своєчасно виявляти слабкі місця. Це сприяє прийняттю обґрунтованих рішень, покращенню командної взаємодії та створенню звітів для зацікавлених сторін. Контроль за витратами та дотриманням термінів відіграє важливу роль у загальному успіху проекту. Завдяки правильному управлінню завданнями можна постійно оцінювати використання ресурсів, перерозподіляти їх у разі необхідності та уникати перевищення бюджету. Систематичний моніторинг процесу виконання роботи допомагає оперативно усувати перешкоди та підтримувати високий рівень ефективності [6], [7].

Координація завдань також сприяє покращенню комунікації всередині команди. Співробітники отримують змогу обмінюватися зворотним зв'язком, оперативно повідомляти про зміни у статусі виконання роботи та коригувати свої дії відповідно до зауважень керівництва. Такий підхід формує злагоджений робочий процес, у якому всі учасники мають чітке розуміння своїх обов'язків і відповідальності [6], [7].

Додатковою перевагою є підвищення загальної ефективності роботи. Чітка структура виконання завдань дозволяє уникати зайвих простоїв та швидко реагувати на можливі труднощі. Це дає змогу зберігати високий темп реалізації проекту та досягати поставлених цілей у запланований термін. Системний підхід до управління завданнями також позитивно впливає на рівень продуктивності команди. Постійний моніторинг прогресу, ефективна комунікація та підтримка з боку керівництва створюють умови, у яких кожен учасник процесу працює максимально результативно. Завдяки детальному

аналізу робочих процесів стає можливим визначення слабких місць і впровадження необхідних змін для підвищення продуктивності [6], [7].

Відзначено, що забезпечення ефективного управління робочими процесами потребує використання спеціалізованого програмного забезпечення, яке дозволяє автоматизувати та координувати виконання завдань. Використання таких технологій сприяє підвищенню продуктивності та забезпечує безперервний контроль за всіма етапами процесів. Завдяки інтеграції з іншими інформаційними системами створюється єдине середовище для обміну даними та синхронізації дій між учасниками.

1.2 Аналіз програмного забезпечення для підтримки управління робочими процесами

Проаналізуємо програмне забезпечення для підтримки управління робочими процесами [8]-[14].

Програмне забезпечення MeisterTask (рис. 1.1) є інтуїтивним інструментом для організації робочих процесів, що дає змогу командам ефективно координувати виконання завдань. Завдяки наочному інтерфейсу платформа спрощує планування, відстеження й управління проектами, сприяючи прозорості та злагодженій співпраці. Функціонал сервісу включає можливість створення візуальних дошок у стилі Канбан, що дозволяє гнучко розподіляти завдання та відстежувати їхній статус. Для покращення комунікації передбачені повідомлення в реальному часі та коментарі, що дає змогу учасникам проекту залишатися в курсі оновлень без необхідності використання сторонніх сервісів для спілкування. Платформа MeisterTask підтримує гнучку настройку робочих процесів, що дозволяє адаптувати її до специфіки кожної команди. Завдяки інтеграції з популярними сервісами, такими як Google Workspace, Slack і GitHub, а також можливості підключення інших інструментів через Zapier, забезпечується безперебійний обмін інформацією між додатками [8], [10], [12].

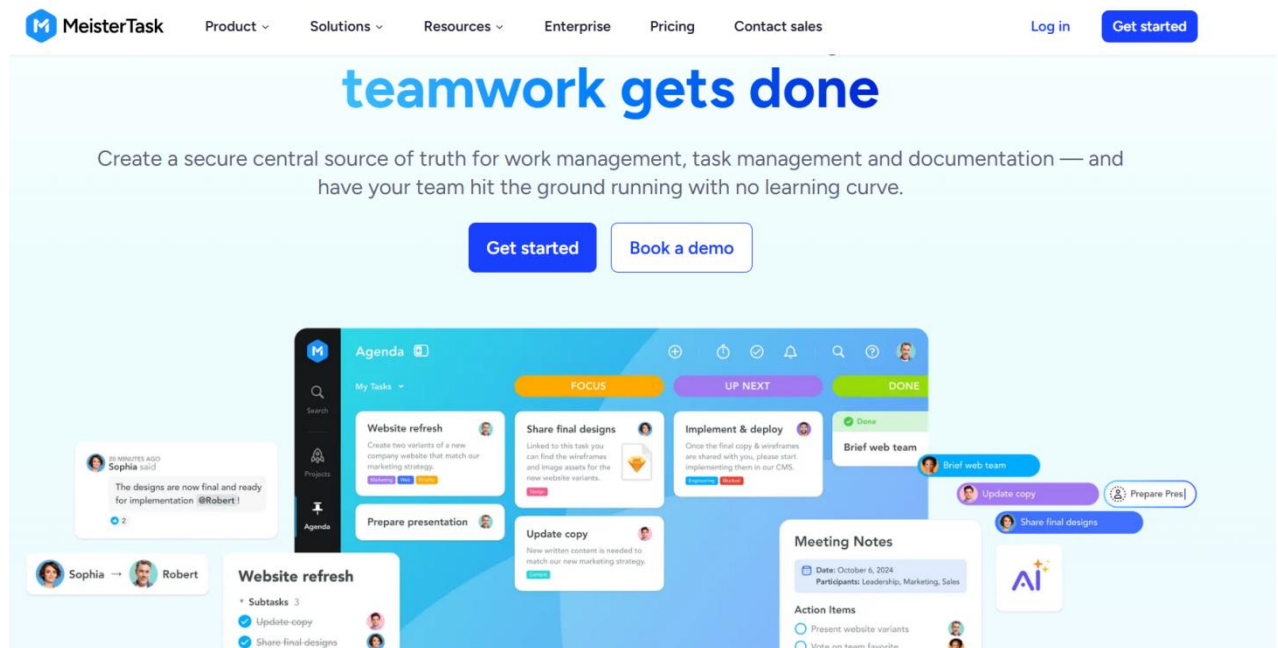


Рисунок 1.1 – Програмне забезпечення MeisterTask [12]

Серед ключових функцій — спеціальний режим, що зменшує кількість відволікаючих факторів, відображаючи лише актуальні завдання. Можливість використання тегів допомагає структурувати завдання відповідно до їхнього статусу, пріоритетності чи інших параметрів. Також реалізовано механізм відстеження часу, що дозволяє оцінювати тривалість виконання завдань і покращувати процес планування [8], [10], [12].

Користувачам доступні додатки для комп'ютерів і мобільних пристроїв, що робить керування завданнями зручним у будь-якому середовищі. Інтерфейс сервісу має простий і привабливий дизайн, що полегшує взаємодію. Крім того, особлива увага приділяється питанням безпеки, що гарантує захист даних користувачів [8], [10], [12].

Завдяки можливості персоналізації головної панелі кожен користувач може отримати зручний доступ до проектів і завдань або переглядати свій індивідуальний розклад. Інструменти для моніторингу кінцевих термінів виконання і обліку витраченого часу вбудовані безпосередньо в платформу, що сприяє підвищенню ефективності виконання завдань [8], [10], [12].

Крім управління процесами, програмне забезпечення MeisterTask також слугує ефективним майданчиком для обговорення завдань. За допомогою

коментарів, згадок і реакцій можна швидко передавати інформацію та координувати роботу [8], [10], [12].

Основними характеристиками програмного забезпечення MeisterTask є такі [8], [10], [12]:

- зручний інтерфейс: програмне забезпечення MeisterTask характеризується простим та інтуїтивно зрозумілим дизайном, що полегшує навігацію та використання [8], [10], [12];

- засоби роботи з канбан дошками: програмне забезпечення MeisterTask надає можливість подати процес виконання завдань у графічному вигляді за допомогою інтерактивних колонок [8], [10], [12];

- автоматизація: програмне забезпечення MeisterTask надає підтримку автоматизованих дій для зменшення рутинних операцій та прискорення робочих процесів [8], [10], [12];

- сповіщення в реальному часі: оперативне інформування про зміни у завданнях, що допомагає залишатися в курсі подій [8], [10], [12];

- гнучкі робочі процеси: програмне забезпечення MeisterTask надає можливість налаштовувати етапи виконання завдань відповідно до потреб команди [8], [10], [12];

- коментарі та згадки: програмне забезпечення MeisterTask надає інтегровану систему обговорення завдань із можливістю згадувати членів команди [8], [10], [12];

- відстеження часу: програмне забезпечення MeisterTask надає вбудований таймер для контролю витраченого на завдання часу та аналізу продуктивності [8], [10], [12];

- інтеграції: сумісність із такими сервісами, як Google Workspace, Slack, GitHub та іншими через Zapier [8], [10], [12];

- кросплатформність: програмне забезпечення MeisterTask надає доступність на настільних і мобільних пристроях для зручної роботи у будь-якому місці [8], [10], [12];

– система тегів: програмне забезпечення MeisterTask надає можливість додавати категорії та фільтрувати завдання для кращої організації проєктів компанії [8], [10], [12].

Серед недоліків програмного забезпечення MeisterTask можна відзначити обмежені можливості для роботи зі складною ієрархією завдань, оскільки платформа не підтримує повноцінні під завдання, а лише контрольні списки. Деякі корисні функції, зокрема автоматизація повторюваних процесів і налаштування періодичних завдань, доступні лише в платних версіях, що може створювати незручності для користувачів без підписки. Також інструмент має менше варіантів гнучкого налаштування у порівнянні з деякими конкурентами, що може обмежувати можливості адаптації до специфічних потреб команди. Попри інтеграцію з популярними сервісами, розширені можливості підключення до сторонніх додатків часто потребують додаткових витрат [8], [10], [12].

Програмне забезпечення Kintone (рис. 1.2) є універсальним інструментом для організації та контролю робочих процесів, що дає змогу створювати власні рішення без необхідності програмування [8], [13].

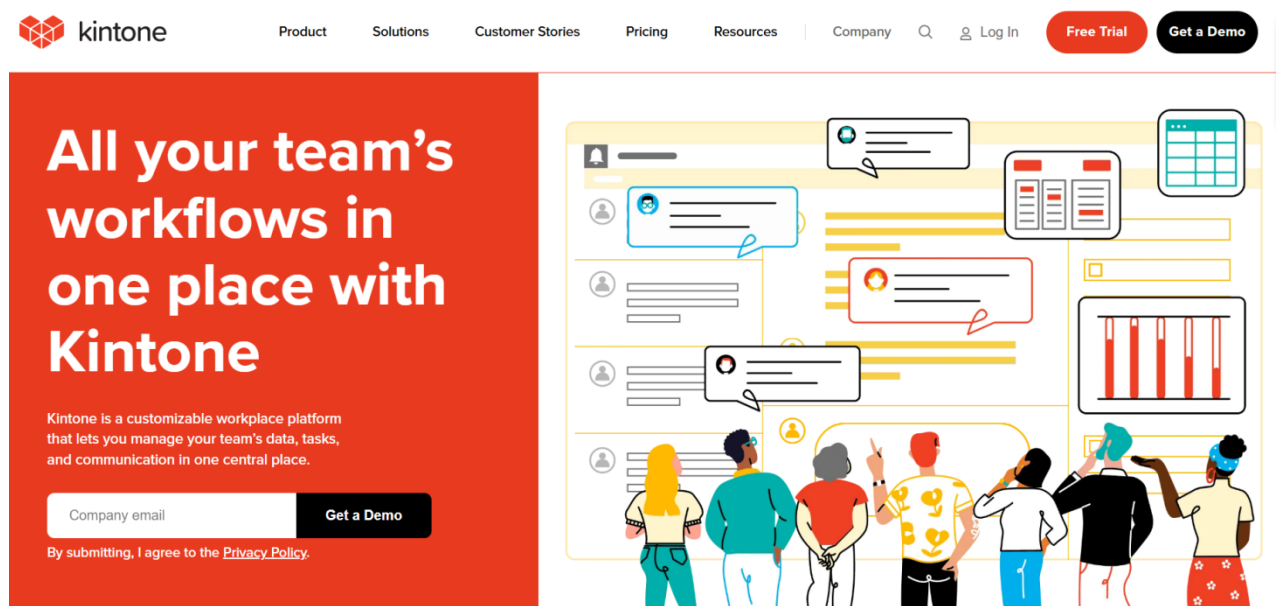


Рисунок 1.2 – Програмне забезпечення Kintone [13]

Завдяки гнучкому підходу до керування завданнями, цей програмне забезпечення Kintone надає можливість розробляти індивідуальні додатки для роботи з даними, використовуючи готові шаблони або імпортуючи інформацію з електронних таблиць [8], [13].

Програмне забезпечення Kintone надає можливість простого налаштування інтерфейсу шляхом додавання необхідних елементів на сторінку, зокрема текстових полів, випадаючих списків, місць для вкладень, автоматичних обчислень, таблиць та взаємопов'язаних даних. Після створення власного додатка можна ефективно управляти інформацією та контролювати виконання завдань, використовуючи систему сповіщень і налаштувань доступу [8], [13].

Kintone забезпечує зручні засоби фільтрації та впорядкування завдань за різними параметрами, що дозволяє формувати аналітичні звіти та налаштовувати інформаційні панелі відповідно до потреб команди. Автоматизація робочих процесів дає змогу оптимізувати передачу завдань між співробітниками та полегшити виконання багатоступеневих проєктів [8], [13].

Програмне забезпечення Kintone надає можливість зв'язку між додатками, що сприяє автоматичному оновленню даних у межах процесів. Інструмент також підтримує створення спеціалізованих робочих просторів, де можна централізовано зберігати файли, відстежувати завдання та вести обговорення у зручному форматі [8], [13].

Програмне забезпечення Kintone інтегрується з популярними сервісами, такими як Slack, Gmail, Google Calendar, Dropbox, Salesforce, WordPress та багатьма іншими. Доступні розширені можливості підключення через API або платну інтеграцію із Zapier [8], [13].

Переваги включають гнучкість у роботі з даними, широкий набір інструментів для адміністраторів і спеціальні умови для освітніх закладів і некомерційних організацій. Водночас серед обмежень – ліміт пам'яті на користувача та вимога мінімальної кількості учасників у командному тарифному плані [8], [13].

Програмне забезпечення Kintone має такі особливості [8], [13]:

- можливість налаштування: програмне забезпечення Kintone дозволяє створювати власні застосунки для управління завданнями без написання коду, використовуючи шаблони або електронні таблиці [8], [13];
- інтеграції: програмне забезпечення Kintone підтримує з'єднання з популярними сервісами, такими як Slack, Gmail, Google Calendar, Dropbox, Salesforce через Zapier або API [8], [13];
- організація роботи: програмне забезпечення Kintone передбачає створення спеціальних робочих просторів для команд, що містять додатки, файли та дискусії в єдиному середовищі [8], [13];
- автоматизація: програмне забезпечення Kintone забезпечує можливість автоматичного оновлення даних, створення нагадувань та передачі завдань між учасниками команди [8], [13];
- фільтрація та звітність: програмне забезпечення Kintone дозволяє надавати пріоритети завданням, налаштовувати перегляди інформаційної панелі та створювати персоналізовані звіти [8], [13];
- управління доступом: програмне забезпечення Kintone надає розширені налаштування дозволів для контролю над доступом до інформації та функцій системи [8], [13];
- обмеження сховища: програмне забезпечення Kintone надає кожному користувачу обмежений обсяг пам'яті, що може впливати на зберігання великих обсягів даних [8], [13];
- тарифні плани: відсутня можливість використання програмного забезпечення для одного користувача, оскільки мінімальний розмір команди повинен відповідати встановленим вимогам [8], [13].

Серед недоліків програмного забезпечення Kintone можна відзначити обмеження доступного дискового простору для кожного користувача, що може стати проблемою при роботі з великими обсягами даних. Крім того, сервіс не передбачає індивідуальних тарифних планів, оскільки мінімальна кількість користувачів у команді повинна бути не меншою за встановлений

ліміт. Деякі функціональні можливості, зокрема інтеграція з зовнішніми сервісами через Zapier або API, доступні лише у платних версіях, що може впливати на доступність певних інструментів для користувачів із обмеженим бюджетом [8], [13].

Програмне забезпечення Backlog (рис. 1.3) є інструментом для управління завданнями, який підходить як для менеджерів проєктів і продуктів, так і для розробників. Система може працювати як локально встановлене рішення або в хмарному середовищі, що дозволяє масштабувати її відповідно до потреб як невеликих колективів, так і великих підприємств. Така універсальність вигідно відрізняє платформу від багатьох альтернатив, що працюють виключно у хмарі. Доступні мобільні додатки дозволяють отримувати доступ до функцій системи з різних пристроїв [8], [14].

backlog
by rakudo

Contact us Log in

**One tool for
doing it all,
together**

**Backlog is the task
management tool for
creators**

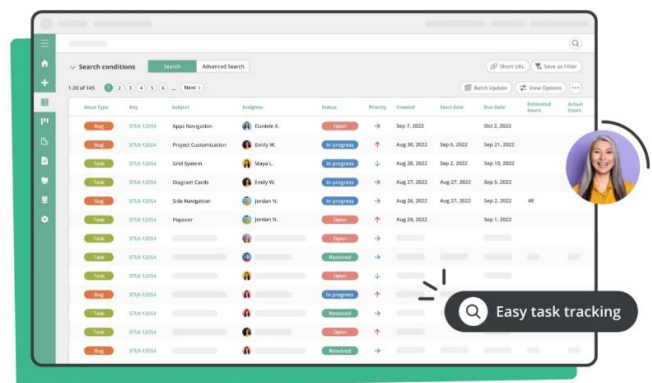


Рисунок 1.3 – Програмне забезпечення Backlog [14]

Однією з ключових переваг програмного забезпечення Backlog є зрозумілий та зручний інтерфейс, що дає змогу працювати із завданнями, календарями та проєктами без зайвих труднощів. Дошки у стилі Канбан, діаграми Ганта та гнучкі оновлення допомагають організовувати робочі

процеси ефективно. Окрім того, програмне забезпечення Backlog надає можливість використовувати запити на злиття та розгалуження, переглядати код, створювати репозиторії Git або Subversion і співпрацювати через вбудовані інструменти документування [8], [14].

Функціонал програмного забезпечення Backlog включає систему відстеження помилок, яка спрощує їх виявлення, розподіл між учасниками команди та усунення. Крім того, реалізована підтримка ієрархічної структури завдань, що допомагає організовувати робочий процес, дозволяючи створювати основні та підлегли завдання. Такий підхід підвищує прозорість і спрощує управління складними проєктами [8], [14].

Програмне забезпечення Backlog підтримує інтеграцію з різними сервісами, такими як Typetalk, Cacao, Redmine, Jira Importer, Google Sheets та інші. Додаткові можливості підключення доступні через API, що дозволяє адаптувати платформу під специфічні потреби [8], [14].

Серед переваг програмного забезпечення Backlog можна виділити зручне ведення журналу помилок, можливість налаштовувати параметри задач відповідно до вимог проєкту та простоту у розподілі завдань за етапами. Водночас, система має певні обмеження, зокрема складність у створенні резервних копій даних та обмежену можливість сортування за декількома критеріями одночасно [8], [14].

Програмне забезпечення Backlog має такі особливості [8], [14]:

- інтерфейс: інтуїтивно зрозумілий, що дозволяє легко переглядати та редагувати проєкти, завдання, календарі та використовувати дошки Канбан і діаграми Ганта [8], [14];

- підтримка кодів: можливість роботи з системами контролю версій Git і Subversion, а також надання функцій для перегляду та співпраці з кодом через сховища wiki [8], [14];

- ієрархія завдань: програмне забезпечення Backlog дає змогу створювати батьківські та дочірні завдання, що забезпечує чітку структуру для організації складних проєктів [8], [14];

- функція відстеження помилок: дозволяє команді швидко фіксувати, призначати та вирішувати проблеми в межах завдань проекту [8], [14];
- мобільний доступ: програмне забезпечення Backlog підтримує можливість роботи на мобільних пристроях [8], [14];
- інтеграції: підтримка інтеграцій з Trello, Asana, Redmine, Jira, iCal, електронною поштою, Google Sheets і можливість створення власних інтеграцій через API [8], [14].

Серед недоліків програмного забезпечення Backlog можна відзначити обмеження доступного дискового простору для кожного користувача, що може стати проблемою при роботі з великими обсягами даних, а також складність у створенні повних резервних копій і управлінні версіями проекту, а також відсутність можливості сортувати завдання за допомогою кількох фільтрів одночасно [8], [14].

Порівняльну характеристику програмного забезпечення для підтримки управління робочими процесами наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для підтримки управління робочими процесами

Критерій порівняння	MeisterTask [12]	Kintone [13]	Backlog [14]
Інтуїтивність інтерфейсу	+	+—	+
Підтримка спеціалізованих застосунків	—	+	+—
Гнучкість налаштувань	+—	+	+—
Засоби автоматизації робочих процесів	+—	+	+—
Підтримка багатоступеневих проєктів	+—	+	+

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато програмних засобів для підтримки управління робочими процесами. Проте деякі програмні засоби для підтримки управління робочими процесами характеризуються високою вартістю та складністю впровадження. Ще одним недоліком програмних засобів для підтримки управління робочими процесами є можливі труднощі з інтеграцією, оскільки не всі програмні рішення легко поєднуються з уже використовуваними корпоративними системами, що може спричинити проблеми з обміном даними та призвести до затримок у роботі. Не менш важливою проблемою є залежність від постачальника програмного забезпечення підтримки управління робочими процесами. Якщо розробник програмного забезпечення для підтримки управління робочими процесами припиняє супровід або змінює модель ліцензування, компанія може зіткнутися з труднощами у підтримці та масштабуванні своїх процесів. Тому актуальною є розробка програмного забезпечення для підтримки управління робочими процесами.

При розробці програмного забезпечення для підтримки управління робочими процесами необхідно забезпечити такі функціональні вимоги:

- підтримка можливості керування проєктами та відображення інформації про них у вигляді Канбан дошки;
- підтримка керування робочими процесами проєктів;
- можливість зміни зовнішнього вигляду відображення проєктів та задач;
- можливість перемикання між проєктами;
- підтримка можливості архівування карток проєкту з метою збереження інформації про реалізовані та завершені задачі та робочі процеси.

1.3 Висновки за розділом 1

Відзначено, що забезпечення ефективного управління робочими процесами потребує використання спеціалізованого програмного

забезпечення, яке дозволяє автоматизувати та координувати виконання завдань. Використання таких технологій сприяє підвищенню продуктивності та забезпечує безперервний контроль за всіма етапами процесів. Завдяки інтеграції з іншими інформаційними системами створюється єдине середовище для обміну даними та синхронізації дій між учасниками.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для підтримки управління робочими процесами. Проте деякі програмні засоби для підтримки управління робочими процесами характеризуються високою вартістю та складністю впровадження. Ще одним недоліком програмних засобів для підтримки управління робочими процесами є можливі труднощі з інтеграцією, оскільки не всі програмні рішення легко поєднуються з уже використовуваними корпоративними системами, що може спричинити проблеми з обміном даними та призвести до затримок у роботі. Не менш важливою проблемою є залежність від постачальника програмного забезпечення підтримки управління робочими процесами. Якщо розробник програмного забезпечення для підтримки управління робочими процесами припиняє супровід або змінює модель ліцензування, компанія може зіткнутися з труднощами у підтримці та масштабуванні своїх процесів. Тому актуальною є розробка програмного забезпечення для підтримки управління робочими процесами.

Сформульовано функціональні вимоги до програмного забезпечення для підтримки управління робочими процесами.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

При розробці програмного забезпечення для підтримки управління робочими процесами було використано мову програмування JavaScript [15], [16].

JavaScript є універсальною мовою програмування, що відзначається високою ефективністю та прозорістю. Спочатку вона була створена для додавання інтерактивності вебсторінкам, дозволяючи програмам взаємодіяти з HTML-кодом і автоматично запускатися під час завантаження сторінки у веббраузері [15], [16].

Розробка на JavaScript не потребує спеціальної підготовки, що робить її доступною для широкого кола користувачів. На відміну від мови Java, яка існувала ще до появи JavaScript, початкова назва цієї мови була LiveScript, однак згодом, з маркетингових міркувань, вона була змінена. Попри схожість у назві, JavaScript не є спрощеним варіантом Java. З часом ця мова розвинулася настільки, що стала основою для ECMAScript – окремого стандарту, який не має спільного коріння з Java [15], [16].

ECMAScript виконує роль базової технології, на якій будуються інші мови сценаріїв, але при цьому має низький рівень продуктивності. Велика кількість функцій, доступних у мові програмування JavaScript, може ускладнити його вивчення, однак поступова практика допомагає краще розібратися у можливостях мови [15], [16].

JavaScript функціонує не лише у браузерах, а й в інших середовищах, однак для цього необхідно мати спеціальне програмне забезпечення. Веб-браузери оснащені вбудованими інтерпретаторами JavaScript, що дає змогу запускати сценарії без додаткових налаштувань. Проте сфера його застосування не обмежується лише вебпереглядачами [15], [16].

Сучасний JavaScript позиціонується як безпечна мова програмування, що орієнтована на загальні завдання. Його можливості залежать від

конкретного середовища, у якому він використовується. У вебзастосунках ця мова забезпечує обробку подій, зміну структури HTML-документів, маніпуляцію стилями та взаємодію з користувачем. Зокрема, JavaScript дає змогу додавати або приховувати елементи сторінки, реагувати на кліки миші або натискання клавіш, здійснювати запити до сервера за допомогою AJAX, а також працювати з cookie-файлами [15], [16].

Попри свою потужність і швидкодію, JavaScript має певні обмеження, встановлені браузерами з міркувань безпеки. Такі заходи унеможливають несанкціонований доступ до особистих даних користувачів або потенційне пошкодження їхніх пристроїв. Водночас ці обмеження не діють, якщо JavaScript використовується поза межами браузерного середовища, наприклад, на сервері. Додаткові можливості можуть бути розширені завдяки спеціальним плагінам і розширенням, однак для їхньої інсталяції потрібні відповідні інструменти [15], [16].

Функціональність мови має певні технічні обмеження. Доступ до файлової системи комп'ютера відсутній, тому JavaScript не може безпосередньо зчитувати або змінювати файли, а також запускати сторонні програми. Браузери підтримують обробку файлів у спеціальному захищеному середовищі, проте такі можливості є обмеженими. Робота JavaScript відбувається в межах однієї вкладки, і взаємодія між різними вкладками чи вікнами заборонена, якщо вони належать до різних доменів, портів або протоколів. Це обмеження запроваджене з міркувань безпеки, щоб запобігти несанкціонованому переміщенню між вкладками через скрипти [15], [16].

Однією з ключових переваг JavaScript є можливість надсилати запити до сервера, з якого завантажується сторінка, що забезпечує динамічне оновлення контенту. Запити до сторонніх ресурсів також можливі, хоча їх реалізація потребує додаткових налаштувань [15], [16].

Обґрунтування вибору мови програмування для розробки програмного забезпечення для підтримки управління робочими процесами наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для розробки програмного забезпечення для підтримки управління робочими процесами

Критерій порівняння мов програмування	Мова програмування		
	JavaScript	Python	C#
Підтримка різних стилів програмування	+	+—	+—
Велика екосистема для веброзробки	+	+	—
Кросплатформність	+	+	+—
Зручність для створення ПЗ для підтримки управління робочими процесами	+	—	—
Асинхронна модель обробки подій	+	+—	+

Отже, для реалізації програмного забезпечення для підтримки управління робочими процесами обрано мову програмування JavaScript, яка дозволяє виконувати код безпосередньо у браузері без необхідності встановлення додаткового програмного забезпечення. Крім того, JavaScript підтримує об'єктно-орієнтоване, функціональне та імперативне програмування, що робить цю мову гнучкою для розробки різних типів застосунків. Іншою перевагою є підтримка асинхронної моделі обробки подій, яка дозволяє ефективно працювати з великими обсягами даних, вебсокетами та мережевими запитами без блокування потоку виконання.

2.2 Вибір фреймворка для створення програмного забезпечення для підтримки управління робочими процесами

В якості фреймворка для створення програмного забезпечення для підтримки управління робочими процесами було обрано AngularJS [17], [18].

Фреймворк AngularJS є платформою з відкритим вихідним кодом, створеною для розробки вебзастосунків, що працюють у межах однієї сторінки. Основний підхід орієнтований на використання архітектури MVC, що спрощує тестування та розробку. Завдяки особливій інтеграції з HTML ця технологія дозволяє створювати шаблонні рішення на основі JavaScript, де значення змінних можуть встановлюватися як вручну, так і динамічно через JSON-дані [17], [18].

Підхід до декларативного програмування у цій платформі дає змогу ефективніше визначати структуру додатків, а використання імперативного стилю полегшує керування бізнес-логікою. Однією з важливих функцій є підтримка двостороннього зв'язування даних, що забезпечує автоматичну синхронізацію між моделлю та відображенням. Це зменшує вплив на модель керування об'єктами і підвищує зручність тестування [17], [18].

Функціональність AngularJS спрямована на логічну організацію структури, що покращує керованість кодом і спрощує перевірку його працездатності. Окрему увагу приділено розділенню клієнтської та серверної частин, що дозволяє працювати над додатком у межах різних команд. Водночас фреймворк підтримує повний цикл розробки, від створення інтерфейсу до тестування логіки. Використання підходу модель-вигляд-контролер допомагає ефективніше узгоджувати роботу даних, представлення та логічних компонентів, а система впровадження залежностей перетворює традиційні процеси на гнучку паралельну обробку [17], [18].

AngularJS є зручним інструментом для розробки одно сторінкових вебзастосунків, забезпечуючи високу швидкість та ефективність роботи. Фреймворк підтримує модульний підхід, що дає змогу розділяти код на автономні частини, спрощуючи тестування, масштабування та повторне використання. Вбудована система ін'єкції залежностей сприяє зручнішому керуванню компонентами та покращує тестування коду [17], [18].

Використання AngularJS дозволяє створювати інтерактивні інтерфейси без необхідності перезавантаження сторінки. Завдяки цьому додатки

працюють швидше та забезпечують плавний користувацький досвід. Фреймворк містить гнучкі механізми для створення динамічного вмісту, включаючи можливість визначення власних HTML-інструкцій для додавання нових елементів без зміни вихідного коду [17], [18].

Однією з ключових особливостей є двостороннє зв'язування даних, що автоматично оновлює відображення при зміні моделі та навпаки. Це значно спрощує інтеграцію даних і дозволяє мінімізувати ручне управління їхнім станом. Вбудовані засоби перевірки форм забезпечують зручність обробки користувацького вводу, що є важливим для бізнес-додатків [17], [18].

Гнучкість платформи дає можливість інтегрувати різноманітні бібліотеки та інструменти, що полегшує розширення функціональності проєкту. Завдяки цьому зменшується час, необхідний для реалізації нових можливостей. Важливою особливістю AngularJS є його орієнтація на тестування: вбудовані інструменти дозволяють контролювати працездатність коду на кожному етапі розробки [17], [18].

Ще однією важливою особливістю є інтеграція з іншими інструментами. AngularJS добре працює з різними технологіями для обробки даних у реальному часі, що є важливим для управління робочими процесами, де необхідна актуалізація даних без перезавантаження сторінки. Завдяки цьому можна ефективно відслідковувати статуси завдань і процесів в реальному часі [17], [18].

Використання AngularJS спрощує взаємодію з сервером та шаблонізацію контенту. Натомість звичайної роботи з моделлю об'єктів, ця технологія оперує внутрішньою моделлю даних, яка автоматично синхронізується з HTML. Змінені значення відразу відображаються у вікні перегляду, що усуває необхідність у ручному оновленні елементів сторінки та сприяє ефективнішій розробці вебзастосунків [17], [18].

Обґрунтування вибору фреймворку для створення програмного забезпечення для підтримки управління робочими процесами наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору фреймворку для створення програмного забезпечення для підтримки управління робочими процесами

Критерій порівняння фреймворків	Фреймворки		
	AngularJS	React	Ember.js
Забезпечення модульності	+	+—	+
Вбудовані функції (маршрутизація, управління станом)	+	—	+
Документація і підтримка спільноти	+	+	+—
Легкість у вивченні	+	+—	—
Підтримка двостороннього зв'язування даних	+	—	+—

Таким чином, для створення програмного забезпечення для підтримки управління робочими процесами обрано фреймворк AngularJS, що надає зручну можливість створення модульних застосунків завдяки своїй архітектурі та дозволяє розробникам розбивати великі проєкти на малі та взаємопов'язані компоненти, що робить код легшим для підтримки та тестування, а також спрощує масштабування системи в міру розширення робочих процесів. Ще однією важливою особливістю є інтеграція з іншими інструментами та технологіями для обробки даних у реальному часі, що є важливим для управління робочими процесами, де необхідна актуалізація даних без перезавантаження сторінки, завдяки чому можна ефективно відслідковувати статуси завдань і процесів в реальному часі. Іншою перевагою є інструменти для тестування, зокрема вбудовані можливості для тестування, що дозволяє розробникам впевнено тестувати функціональність додатків і гарантувати їх безпеку та стабільність.

2.3 Висновки за розділом 2

Для реалізації програмного забезпечення для підтримки управління робочими процесами обрано мову програмування JavaScript, яка дозволяє виконувати код безпосередньо у браузері без необхідності встановлення додаткового програмного забезпечення. Крім того, JavaScript підтримує об'єктно-орієнтоване, функціональне та імперативне програмування, що робить цю мову гнучкою для розробки різних типів застосунків. Іншою перевагою є підтримка асинхронної моделі обробки подій, яка дозволяє ефективно працювати з великими обсягами даних, вебсокетами та мережевими запитами без блокування потоку виконання.

Для створення програмного забезпечення для підтримки управління робочими процесами обрано фреймворк AngularJS, що надає зручну можливість створення модульних застосунків завдяки своїй архітектурі та дозволяє розробникам розбивати великі проєкти на малі та взаємопов'язані компоненти, що робить код легшим для підтримки та тестування, а також спрощує масштабування системи в міру розширення робочих процесів.

3 ОПИС ПРОГРАМИ

3.1 Структура програмного забезпечення для підтримки управління робочими процесами

Структуру програмного забезпечення для підтримки управління робочими процесами наведено на рис. 3.1.

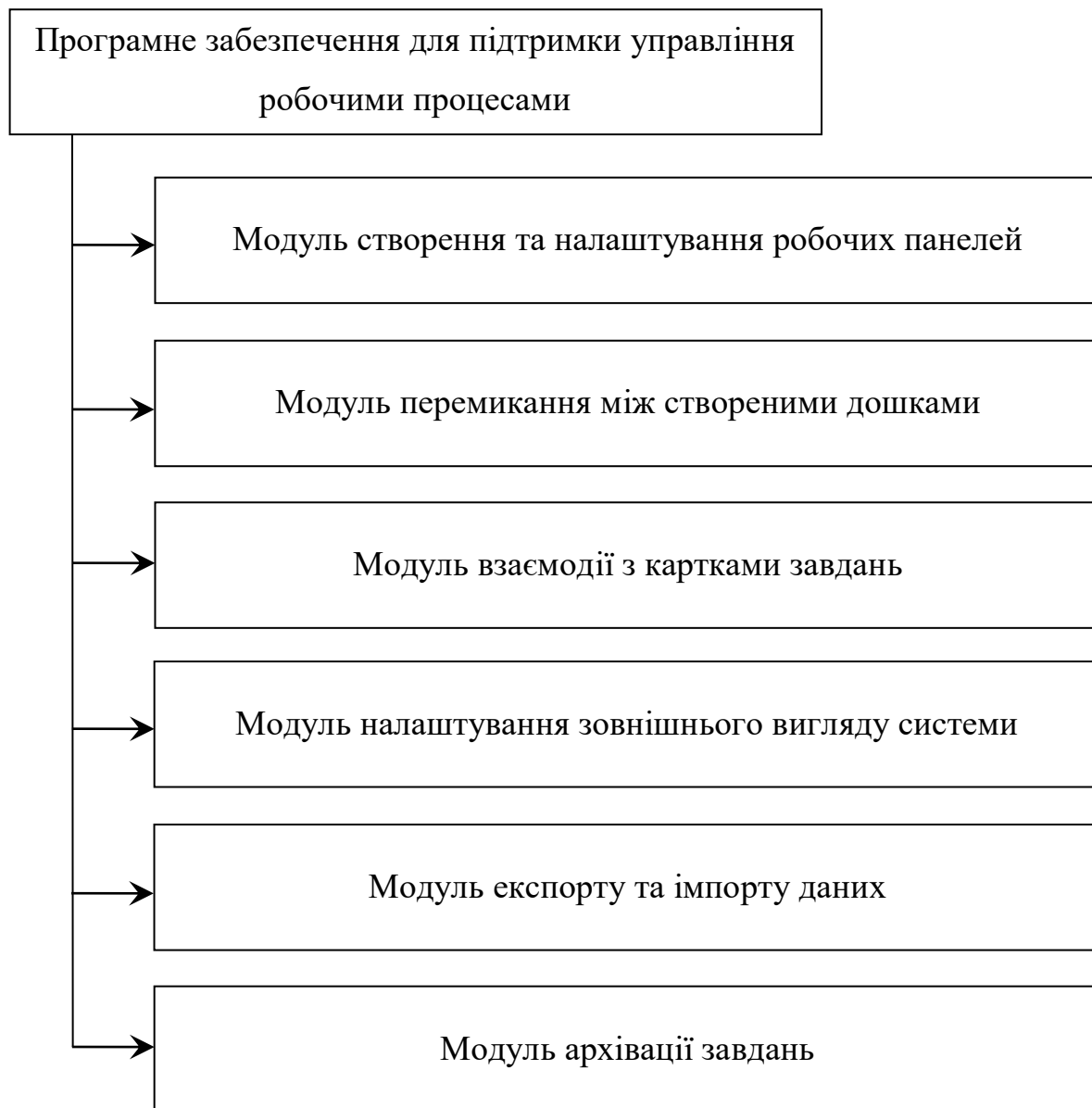


Рисунок 3.1 – Структура програмного забезпечення для підтримки управління робочими процесами

Структура програмного забезпечення для підтримки управління робочими процесами включає шість основних модулів, які забезпечують повний функціонал системи: модуль створення та налаштування робочих панелей, модуль перемикання між створеними дошками, модуль взаємодії з картками завдань, модуль налаштування зовнішнього вигляду системи, модуль експорту та імпорту даних, модуль архівації завдань.

Програма реалізована у вигляді вебзастосунку, що дозволяє створювати візуальні панелі для керування завданнями. Центральним елементом є механізм побудови інтерактивних робочих просторів, що містять картки із завданнями. Вона передбачає модуль для генерації нових панелей, який дозволяє користувачам налаштовувати робочі області відповідно до своїх потреб. Окремий модуль забезпечує можливість швидкого перемикання між створеними панелями, що робить управління кількома процесами одночасно більш зручним.

Для роботи із завданнями передбачено окремий модуль, який дозволяє додавати, змінювати та вилучати картки, а також розміщувати їх у потрібних стовпцях. Окрім цього, реалізовано функціонал налаштування зовнішнього вигляду панелей, що дозволяє змінювати стилі відображення для зручності користувачів. Додатково система підтримує можливість збереження та відновлення даних за допомогою модуля експорту та імпорту панелей. Останній модуль відповідає за архівацію завершених завдань, що дозволяє користувачам зберігати історію роботи та за потреби повертатися до раніше виконаних етапів.

Перший модуль відповідає за створення та налаштування робочих панелей. Він дозволяє користувачам формувати нові канбан дошки, змінювати їхню структуру та визначати ключові параметри, такі як назва, кількість стовпців і базові налаштування відображення.

Другий модуль реалізує можливість перемикання між створеними дошками. Завдяки цьому користувач може швидко переходити між різними

проектами або робочими процесами, що полегшує керування завданнями у великих командах або за умов одночасної роботи над кількома проектами.

Третій модуль забезпечує взаємодію з картками завдань. Він дозволяє додавати нові картки, редагувати їхній вміст, переміщувати між стовпцями, змінювати їхній статус або видаляти після завершення роботи.

Четвертий модуль відповідає за налаштування зовнішнього вигляду системи. У цьому блоці користувачі можуть змінювати кольорові схеми, шрифти, оформлення карток та інших елементів, що дозволяє адаптувати інтерфейс відповідно до особистих вподобань або корпоративного стилю.

П'ятий модуль реалізує функцію експорту та імпорту даних. Він дає змогу зберігати робочі дошки у зовнішніх файлах, передавати їх між користувачами або відновлювати збережені конфігурації у разі потреби.

Шостий модуль відповідає за архівацію завдань. Завершені картки можна переміщати до архіву, зберігаючи інформацію про виконані роботи, що дозволяє користувачам аналізувати минулі завдання та вести історію процесів.

Завдяки цим модулям програмне забезпечення надає користувачам повний набір інструментів для ефективного керування робочими процесами, підвищення продуктивності та організації спільної роботи.

Таким чином, розроблене рішення надає користувачам широкий спектр можливостей для управління завданнями, включаючи створення та видалення панелей, перемикання між ними, редагування карток і їхнє впорядкування. Завдяки цьому система забезпечує ефективну організацію робочого процесу та підвищує продуктивність команди.

3.2 Функціонування програмного забезпечення для підтримки управління робочими процесами

Функціонування програмного забезпечення для підтримки управління робочими процесами подамо за допомогою діаграми послідовностей, зображеної на рис. 3.2.

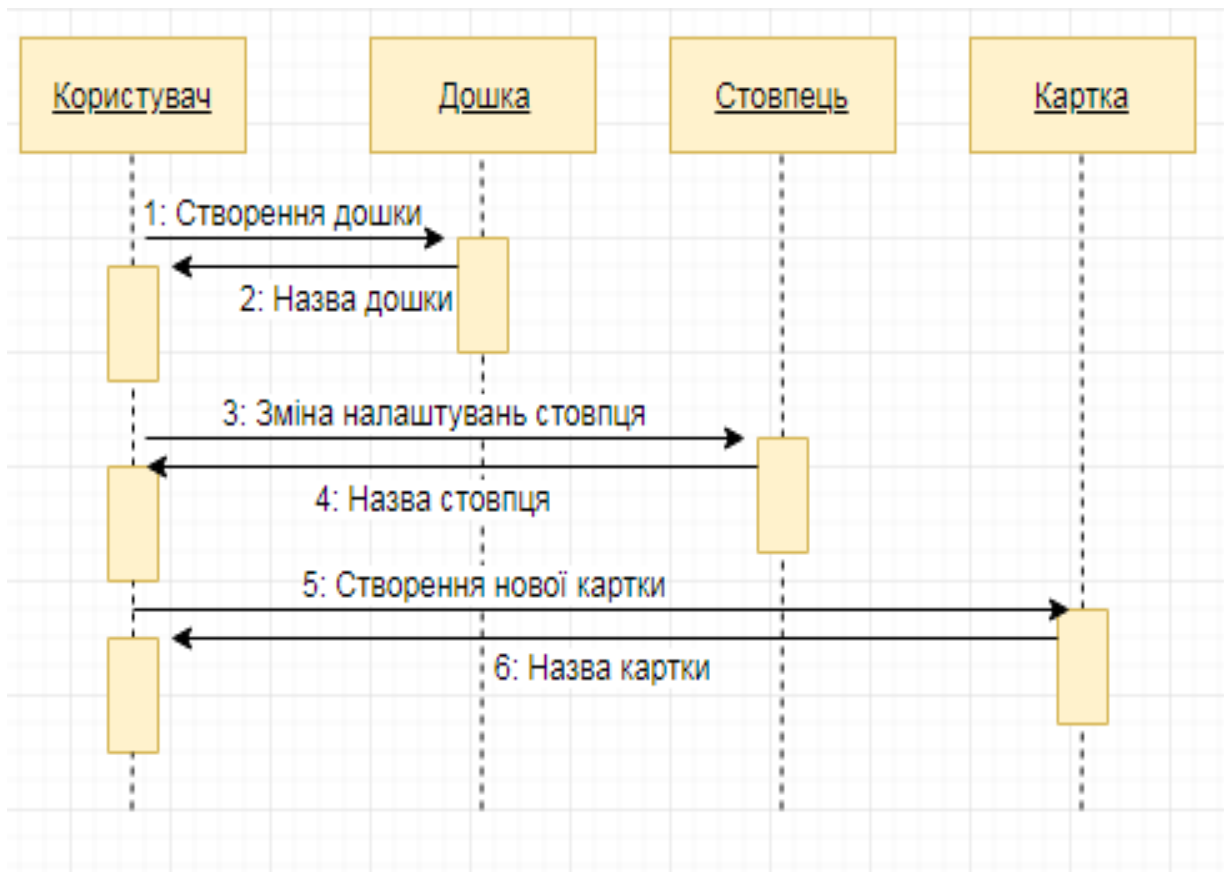


Рисунок 3.2 – Схема функціонування програмного забезпечення для підтримки управління робочими процесами у вигляді діаграми послідовностей

Діаграма послідовностей ілюструє основні етапи взаємодії користувача з програмним забезпеченням для підтримки управління робочими процесами.

Спочатку користувач ініціює створення нової дошки, передаючи відповідний запит системі. Після цього дошка генерується, і користувач отримує її назву у відповідь. Далі користувач змінює налаштування окремих стовпців на дошці, відправляючи запит на зміну параметрів. Система оновлює стовець та надсилає користувачеві його нову назву.

На наступному етапі користувач створює картку в одному з визначених стовпців. Після обробки запиту система генерує картку і передає її назву користувачеві.

Таким чином, діаграма відображає логічну послідовність дій, що забезпечують гнучке управління робочими процесами: від створення дошки до налаштування стовпців та додавання карток.

Побудуємо діаграми прецедентів програмного забезпечення для підтримки управління робочими процесами. Діаграму прецеденту Створення нової дошки у програмному забезпеченні для підтримки управління робочими процесами наведено на рис. 3.3.

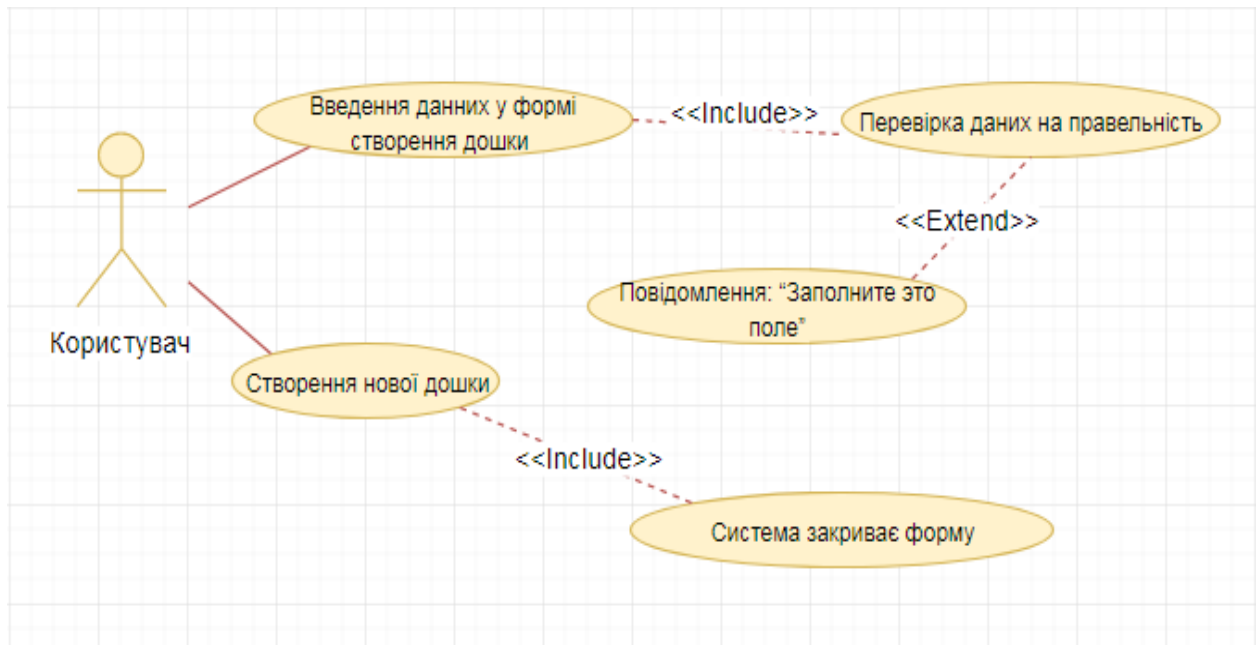


Рисунок 3.3 – Діаграма прецеденту Створення нової дошки у програмному забезпеченні для підтримки управління робочими процесами

Спочатку відкривається список, після чого обирається опція створення нової дошки. У відповідь система відкриває спеціальну форму для введення необхідних параметрів. Далі заповнюється назва дошки, задається кількість стовпців, а якщо вже існує інша дошка, можна обрати один із доступних шаблонів. Після внесення всіх даних підтверджується створення дошки натисканням кнопки, що ініціює перевірку правильності введення інформації. Якщо всі обов'язкові поля заповнені, система завершує процес і закриває форму.

У разі натискання кнопки скасування здійснюється вихід із форми без збереження змін. Якщо поле з назвою залишене порожнім, система виводить відповідне повідомлення про необхідність його заповнення й очікує внесення цієї інформації.

Діаграму прецеденту Створення нової картки у програмному забезпеченні для підтримки управління робочими процесами наведено на рис. 3.4.

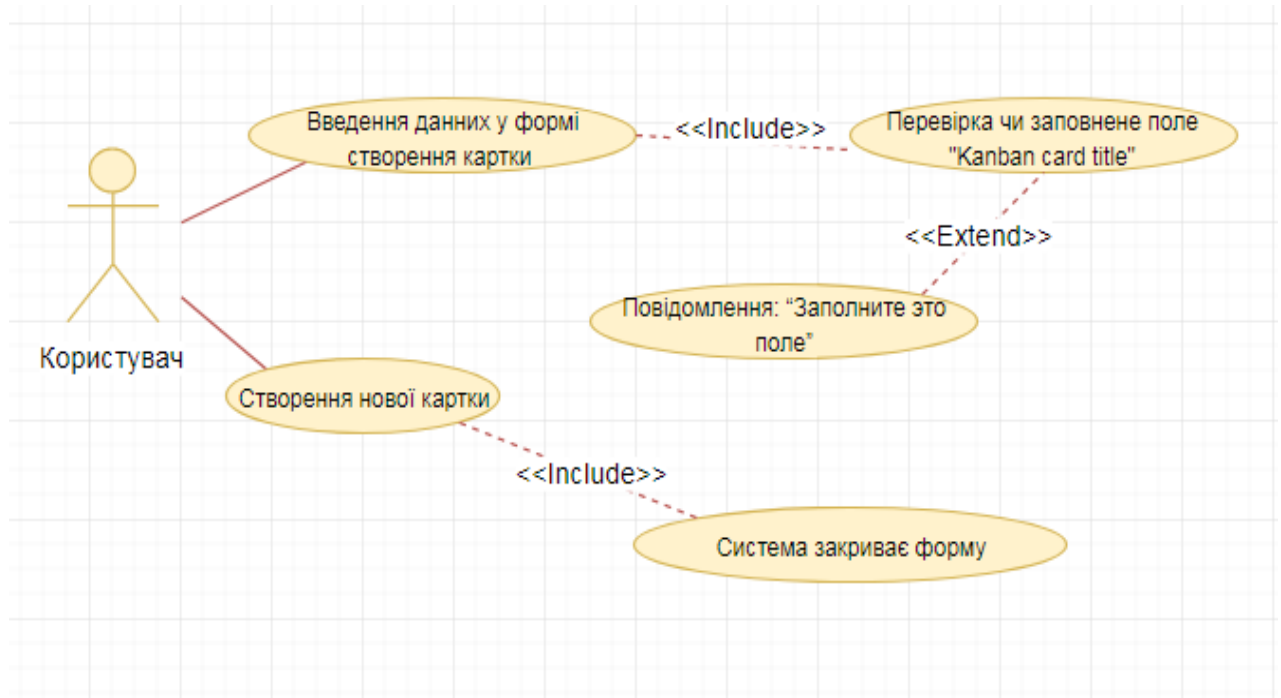


Рисунок 3.4 – Діаграма прецеденту Створення нової картки у програмному забезпеченні для підтримки управління робочими процесами

Прецедент Створення нової картки описує процес додавання нового елемента до робочої дошки. Процес починається з натискання кнопки для додавання картки, після чого система відкриває відповідну форму введення. Далі вводиться необхідна інформація, включаючи заголовок картки, її детальний опис і вибирається відповідний колір. Після підтвердження введених даних система виконує перевірку заповнення основного поля. Якщо заголовок картки зазначено, форма автоматично закривається, і картка додається на дошку. Якщо замість підтвердження дії обирається опція закриття форми, система скасовує створення картки та виходить без збереження змін. У випадку відсутності назви картки в обов'язковому полі виводиться повідомлення про необхідність його заповнення, після чого очікується внесення відповідної інформації.

Діаграму прецеденту зміни налаштування стовпця у програмному забезпеченні для підтримки управління робочими процесами наведено на рис. 3.5.

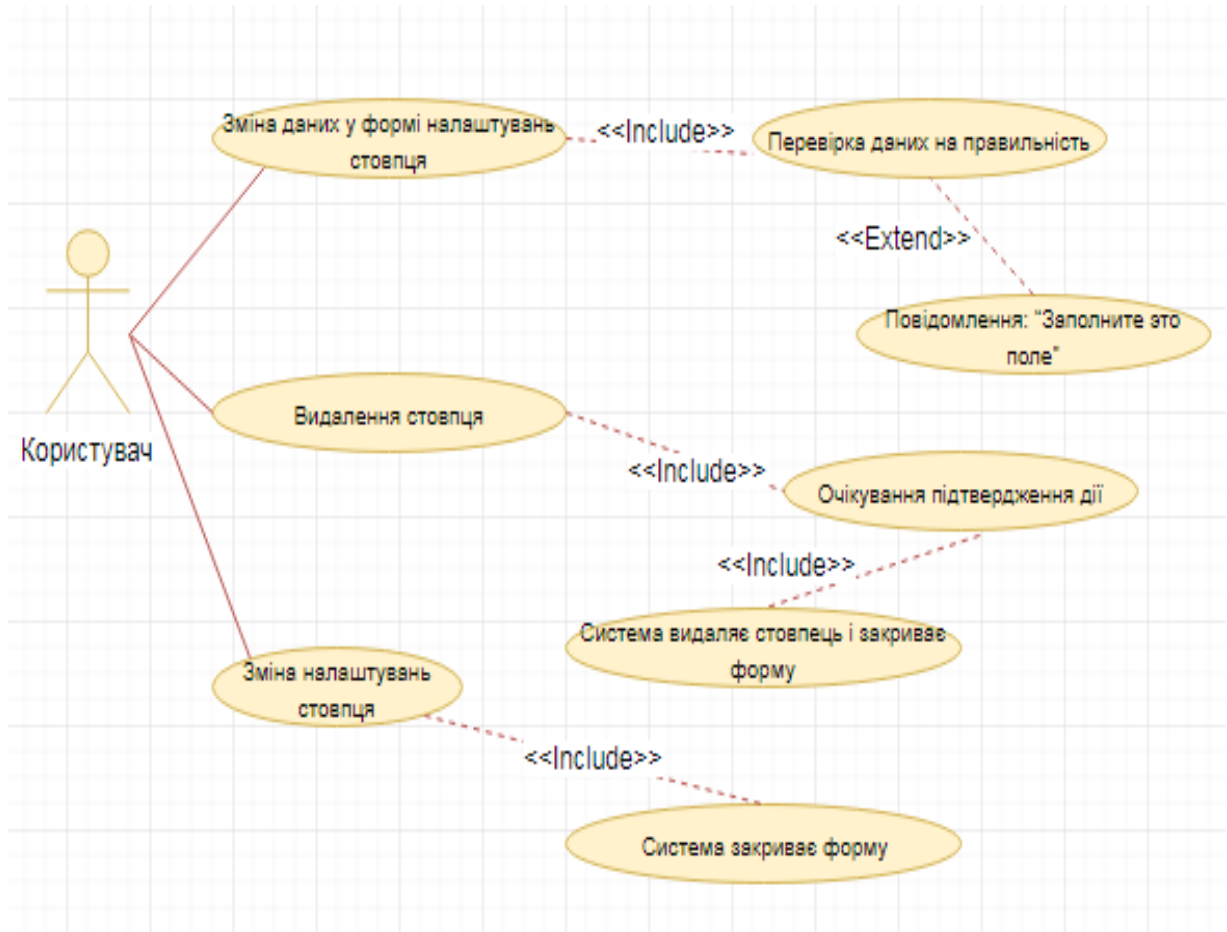


Рисунок 3.5 – Діаграма прецеденту зміни налаштування стовпця у програмному забезпеченні для підтримки управління робочими процесами

Прецедент зміни налаштувань стовпця описує процедуру редагування параметрів стовпця на дошці. Процес розпочинається з відкриття форми налаштувань після натискання відповідної піктограми. У вікні редагування вводиться необхідна інформація, така як оновлена назва стовпця, вибирається колір і за необхідності встановлюється обмеження на кількість карток. Після підтвердження змін система перевіряє, чи поле назви заповнене. Якщо інформація внесена коректно, вікно налаштувань закривається, і внесені зміни застосовуються.

Якщо форма закривається без підтвердження змін, система скасовує редагування і повертається до попереднього стану. Якщо поле назви стовпця залишається порожнім, з'являється повідомлення з вимогою його заповнення.

За умови натискання кнопки видалення система змінює її текст на попередження та відображає іконку застереження. Для завершення видалення користувач має підтвердити дію повторним натисканням, після чого система видаляє стовпець і закриває вікно налаштувань.

3.3 Особливості реалізації програмного забезпечення для підтримки управління робочими процесами

Реалізація програмного забезпечення для підтримки управління робочими процесами передбачає використання різних технологій та бібліотек, що забезпечують функціональність канбан дошки. Основна структура включає індексний файл, який є точкою входу, а також окремі файли для налаштування тем, зовнішніх скриптів і стилів.

Для роботи інтерфейсу застосовується набір компонентів Bootstrap, що містить HTML, CSS і JavaScript-елементи. Використання Glyphicon Halflings дозволяє інтегрувати стандартні іконки. Окрім цього, у проєкті використано бібліотеки AngularJS та Angular-route, які відповідають за реалізацію логіки та маршрутизацію, а також модуль angular-spectrum-colorpicker для налаштування кольорових схем.

Збереження даних виконується за допомогою технології localStorage (рис. 3.6), що дає змогу зберігати інформацію у веббраузері без необхідності надсилати її на сервер. Дані залишаються доступними до моменту їхнього примусового видалення користувачем або через JavaScript. Використання StorageEvent дає змогу відстежувати зміни у сховищі, надсилаючи сповіщення у відкриті вкладки цього ж вебзастосунку.

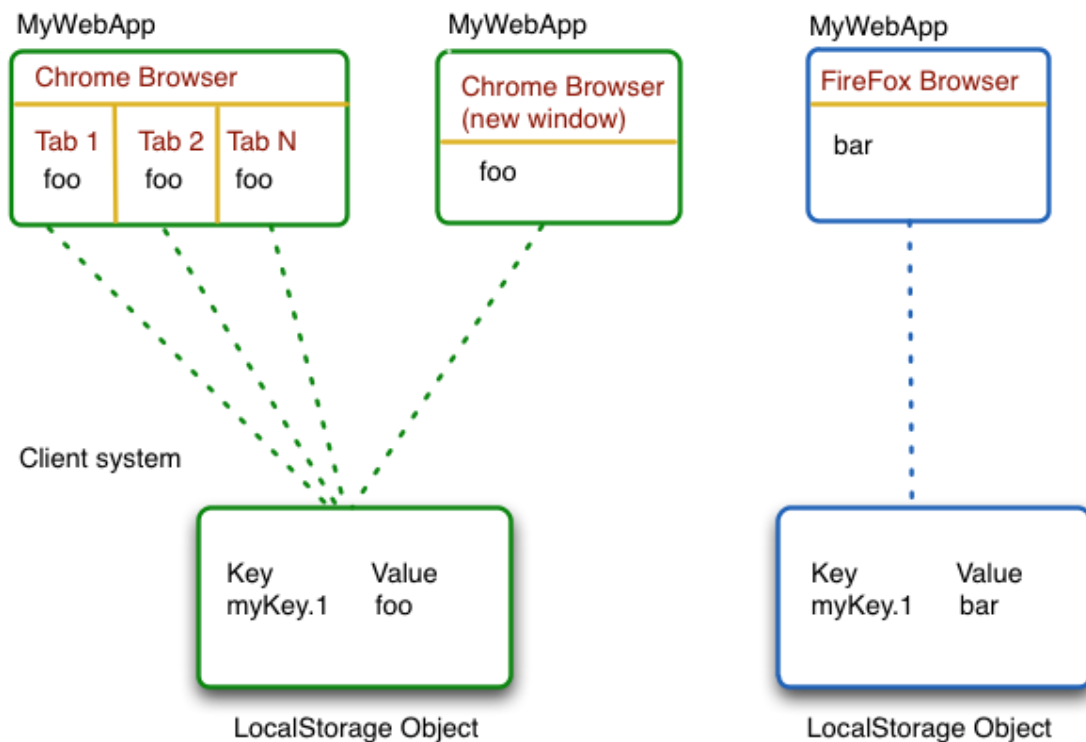


Рисунок 3.6 – Функціонування localStorage

Для забезпечення динамічності та інтерактивності програмного забезпечення для підтримки управління робочими процесами використано jQuery та набір jQuery UI, що дозволяє створювати гнучкий користувацький інтерфейс. Крім того, підтримується завантаження файлів у межах проєкту завдяки FileSaver.

Усі ці технології інтегруються для створення ефективного програмного забезпечення для підтримки управління робочими процесами, що підтримує налаштування стовпців, зміну кольорових тем і збереження даних без необхідності постійного підключення до сервера.

3.4 Проєктування інтерфейсу програмного забезпечення для підтримки управління робочими процесами

Проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для підтримки управління робочими процесами виконано з урахуванням вимог технічного завдання. При розробленні програмного

забезпечення для підтримки управління робочими процесами враховані функціональні вимоги до програми:

- підтримка можливості керування проєктами та відображення інформації про них у вигляді Канбан дошки;
- підтримка керування робочими процесами проєктів;
- можливість зміни зовнішнього вигляду відображення проєктів та задач;
- можливість перемикання між проєктами;
- підтримка можливості архівування карток проєкту з метою збереження інформації про реалізовані та завершені задачі та робочі процеси.

Інтерфейс головної форми програмного забезпечення для підтримки управління робочими процесами наведено на рис. 3.7.

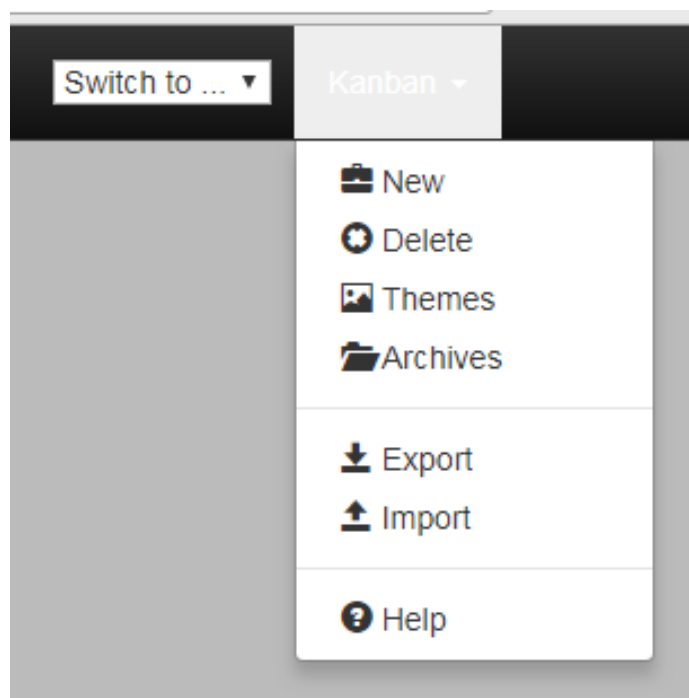


Рисунок 3.7 – Інтерфейс головної форми програмного забезпечення для підтримки управління робочими процесами

Форма створення нового проєкту наведено на рис. 3.8.

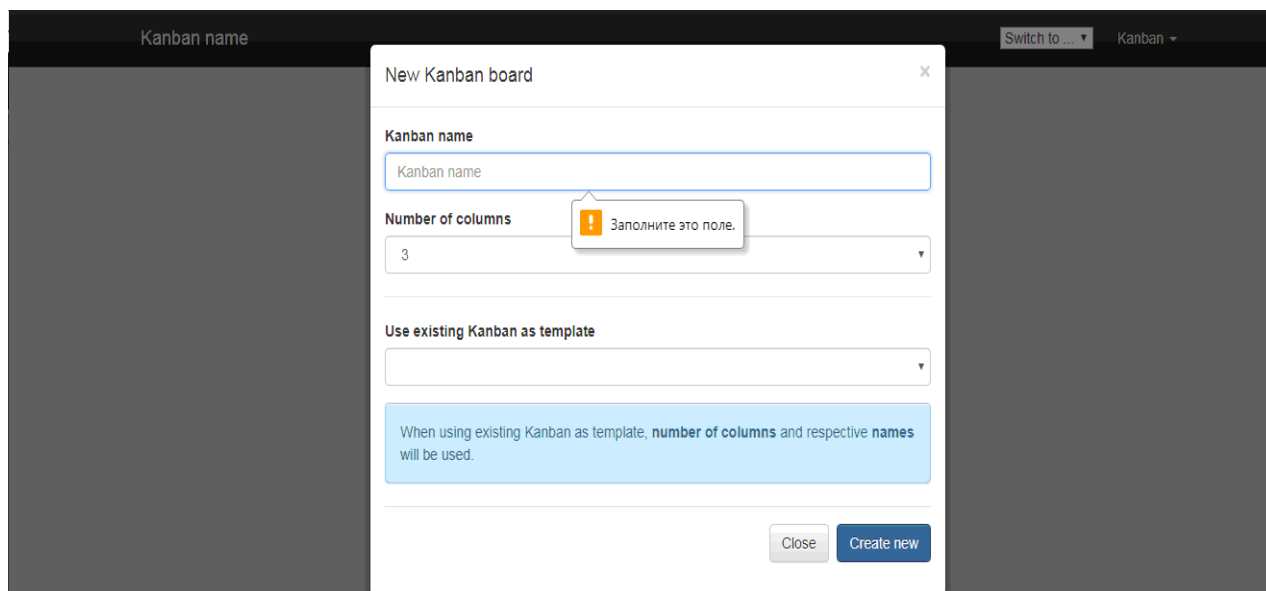


Рисунок 3.8 – Форма створення нового проекту

Форма додавання нової картки до поточного проекту наведено на рис. 3.9.

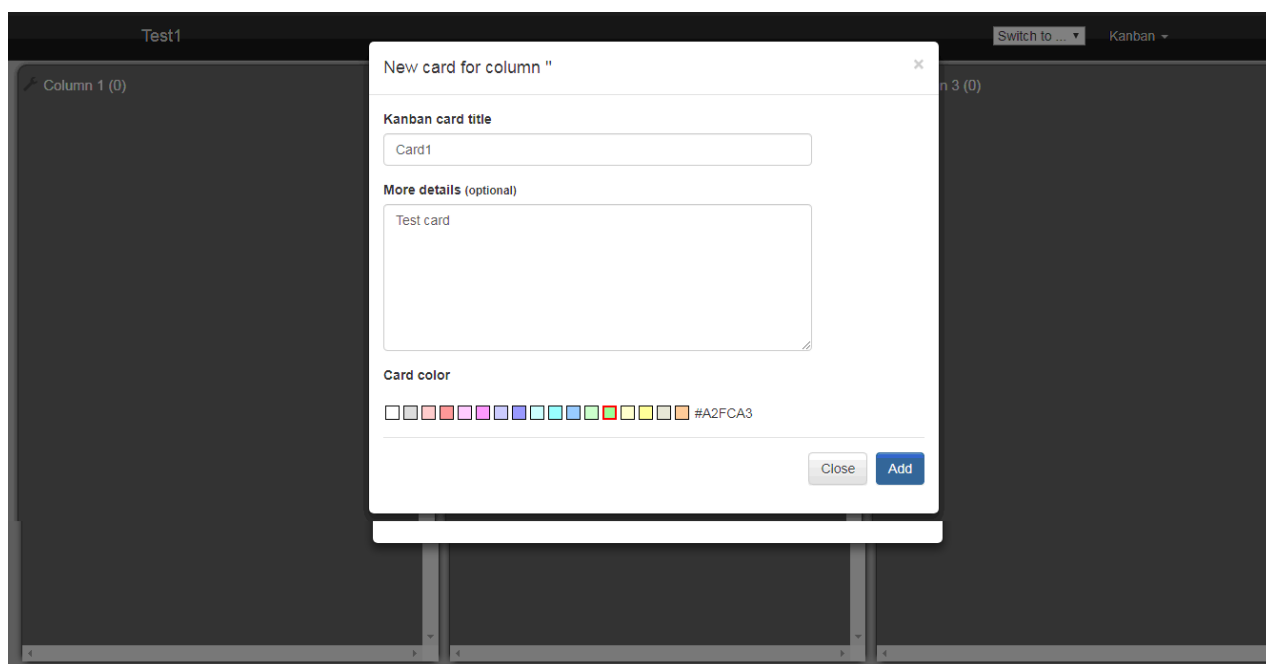


Рисунок 3.9 – Форма додавання нової картки до поточного проекту

Меню перемикання між робочими проектами наведено на рис. 3.10.

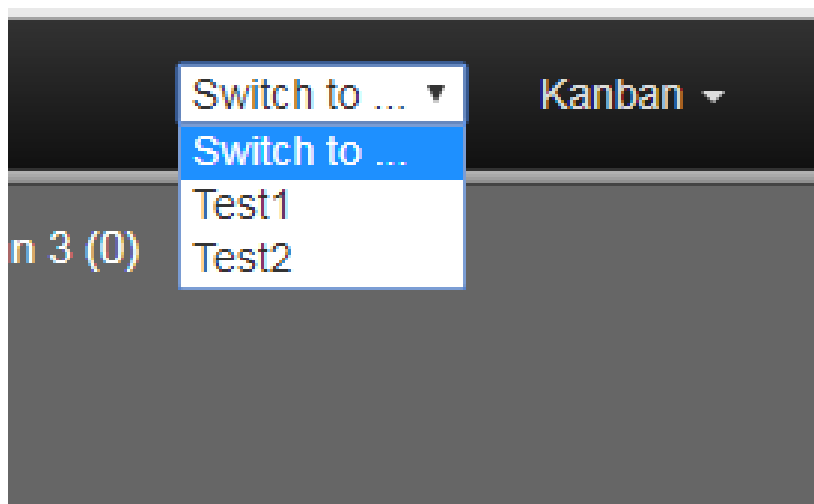


Рисунок 3.10 – Меню перемикавання між робочими проєктами

Форму зміни зовнішнього вигляду подання дошки поточного проєкту наведено на рис. 3.11.

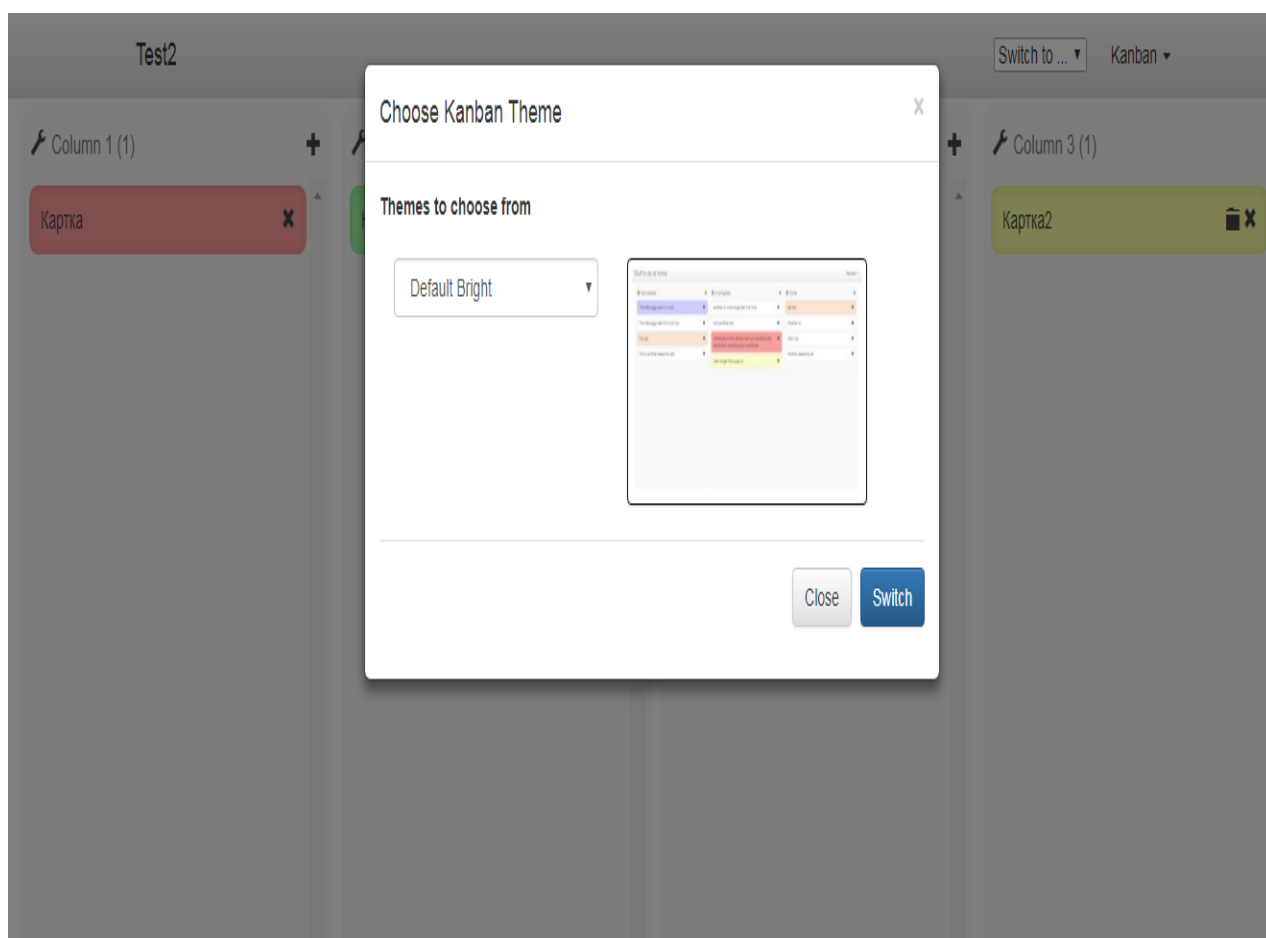


Рисунок 3.11 – Зміна зовнішнього вигляду подання дошки поточного проєкту

Форму архівованих карток наведено на рис. 3.12.

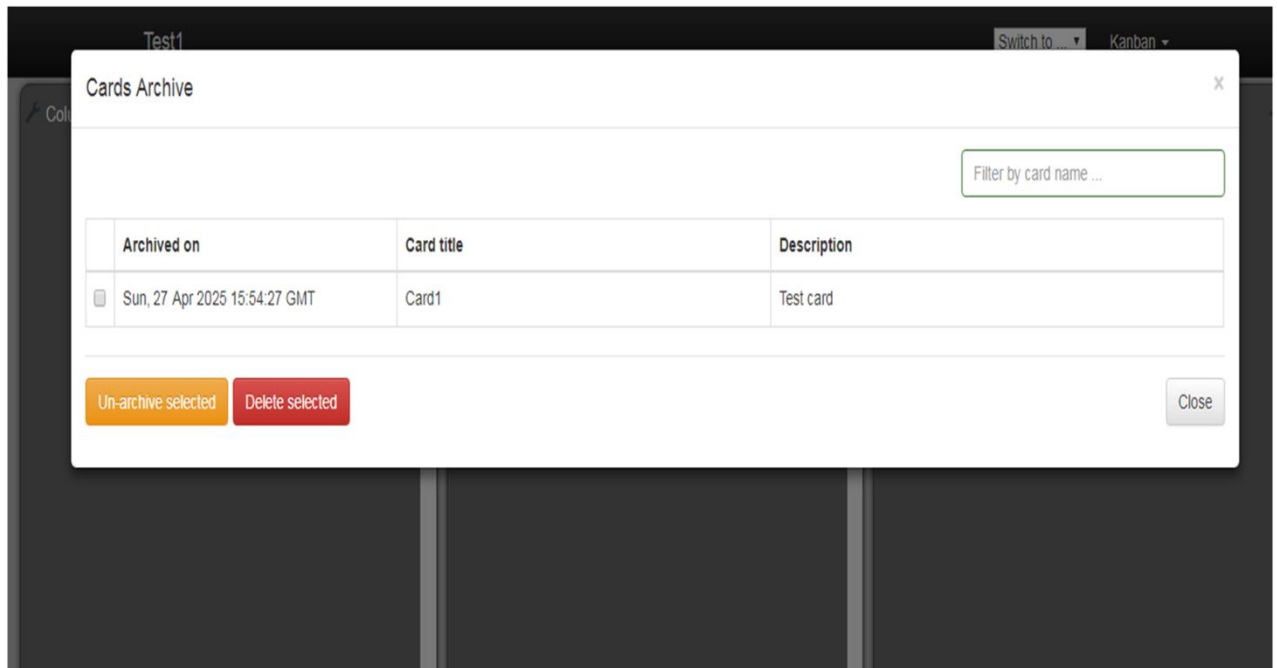


Рисунок 3.12 – Форма архівованих карток

3.5 Висновки за розділом 3

Розроблено програмне забезпечення для підтримки управління робочими процесами.

Запропоновано структуру програмного забезпечення для підтримки управління робочими процесами, що складається з шести основних модулів, які забезпечують повний функціонал системи: модуль створення та налаштування робочих панелей, модуль перемикання між створеними дошками, модуль взаємодії з картками завдань, модуль налаштування зовнішнього вигляду системи, модуль експорту та імпорту даних, модуль архівації завдань.

Описано функціонування програмного забезпечення для підтримки управління робочими процесами за допомогою діаграми послідовностей. Діаграма послідовностей ілюструє основні етапи взаємодії користувача з програмним забезпеченням для підтримки управління робочими процесами.

Описано особливості реалізації програмного забезпечення для підтримки управління робочими процесами. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для підтримки управління робочими процесами.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для підтримки управління робочими процесами. Програма написана на мові JavaScript, яка дозволяє виконувати код безпосередньо у браузері без необхідності встановлення додаткового програмного забезпечення. В якості фреймворку для створення програмного забезпечення для підтримки управління робочими процесами обрано AngularJS, що надає зручну можливість створення модульних застосунків завдяки своїй архітектурі та дозволяє розробникам розбивати великі проєкти на малі та взаємопов'язані компоненти, що робить код легшим для підтримки та тестування, а також спрощує масштабування системи в міру розширення робочих процесів.

Програмне забезпечення для підтримки управління робочими процесами забезпечує виконання таких функцій:

- підтримка можливості керування проєктами та відображення інформації про них у вигляді Канбан дошки;
- підтримка керування робочими процесами проєктів;
- можливість зміни зовнішнього вигляду відображення проєктів та задач;
- можливість перемикання між проєктами;
- підтримка можливості архівування карток проєкту з метою збереження інформації про реалізовані та завершені задачі та робочі процеси.

Для роботи програмного забезпечення для підтримки управління робочими процесами потрібні такі технічні та програмні ресурси: обчислювальні потужності серверів або хмарних платформ, які забезпечують виконання основних операцій і обробку даних у реальному часі. Необхідна відповідна програмна інфраструктура, включаючи операційні системи, системи керування базами даних та інструменти для автоматизації бізнес-

процесів. Важливу роль відіграють мережеві ресурси, що забезпечують безперебійну передачу даних між користувачами та програмним забезпеченням для підтримки управління робочими процесами.

Функціональні характеристики розробленого програмного забезпечення для підтримки управління робочими процесами наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення для підтримки управління робочими процесами наведено у додатку Б.

4.2 Характеристики програми для підтримки управління робочими процесами

Програмне забезпечення для підтримки управління робочими процесами складається з шести основних модулів, які забезпечують повний функціонал системи: модуль створення та налаштування робочих панелей, модуль перемикання між створеними дошками, модуль взаємодії з картками завдань, модуль налаштування зовнішнього вигляду системи, модуль експорту та імпорту даних, модуль архівації завдань.

Програмне забезпечення для підтримки управління робочими процесами характеризується здатністю автоматизувати виконання завдань, контролювати їхній стан та координувати взаємодію між користувачами. Забезпечується інтеграція з іншими інформаційними системами, що дозволяє працювати з різними джерелами даних та обмінюватися інформацією між підсистемами.

Важливою особливістю є гнучкість налаштування процесів, що дає змогу адаптувати роботу системи до потреб організації без значних змін у програмному коді. Підтримується можливість візуального моделювання бізнес-процесів, що спрощує їхнє проектування та подальшу оптимізацію.

Забезпечується контроль доступу до даних та операцій, що гарантує безпеку інформації та запобігає несанкціонованим діям. Підтримується аудит

та відстеження змін, що дозволяє аналізувати історію виконання завдань і підвищувати ефективність управління.

Програмне забезпечення для підтримки управління робочими процесами здатне працювати в багатокористувацькому режимі, що сприяє спільній роботі над процесами та підвищує рівень координації між учасниками. Наявність засобів аналітики дозволяє оцінювати ефективність виконання процесів, виявляти вузькі місця та приймати обґрунтовані рішення для їхнього вдосконалення.

Забезпечується сумісність з мобільними пристроями, що надає можливість працювати із системою незалежно від місця перебування. Підтримується масштабованість, що дозволяє використовувати програмне забезпечення в організаціях різного рівня та обсягу діяльності.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для запуску програмного забезпечення для підтримки управління робочими процесами необхідно перейти у веббраузері на відповідну вебсторінку.

4.3.2 Вхідні й вихідні дані

Вхідними даними до програмного забезпечення для підтримки управління робочими процесами є інформація про робочі процеси у вигляді карток та дошки.

Вихідними даними програмного забезпечення для підтримки управління робочими процесами є додана до системи інформація про робочі процеси у вигляді карток та дошки.

4.3.3 Повідомлення

Користувач програмного забезпечення для підтримки управління робочими процесами може отримати повідомлення про успішне додавання інформації про робочі процеси.

4.4 Виконання програмного забезпечення для підтримки управління робочими процесами

Після запуску програмного забезпечення для підтримки управління робочими процесами користувачу відображається головна сторінка, де можна створити новий проєкт компанії у вигляді Канбан дошки. Для цього у головному меню (рис. 4.1) необхідно обрати пункт меню New.

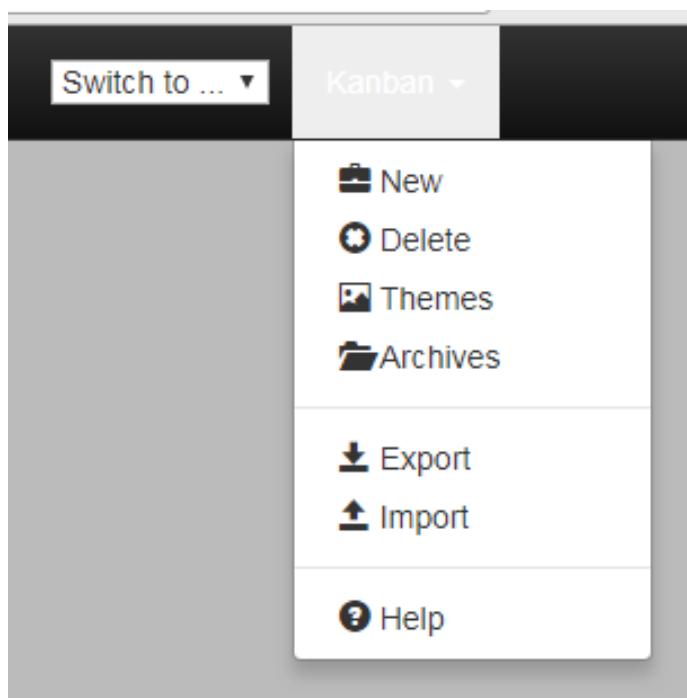


Рисунок 4.1 – Інтерфейс головної форми та створення нового проєкту компанії

Після цього у вікні, що відкрилося (рис. 4.2), необхідно ввести параметри створюваної дошки, що відображає проєкт компанії або організації.

Рисунок 4.2 – Форма створення нового проекту

Після створення нового проекту користувач може додати нові картки (рис. 4.3), які являють собою завдання або робочі процеси поточного проекту.

Рисунок 4.3 – Форма додавання нової картки до поточного проекту

Користувач має можливість вибору кольорів та інших параметрів картки, що відображає робочі процеси. Картки (задачі проекту) можна переглядати або переміщувати між колонками, відображаючи таким чином

прогрес у робочих процесах компанії. За наявності кількох проєктів у компанії існує можливість створення декількох дошок (кожна буде відображати свій проєкт). Перемикання між дошками здійснюється за допомогою меню (рис. 4.4).

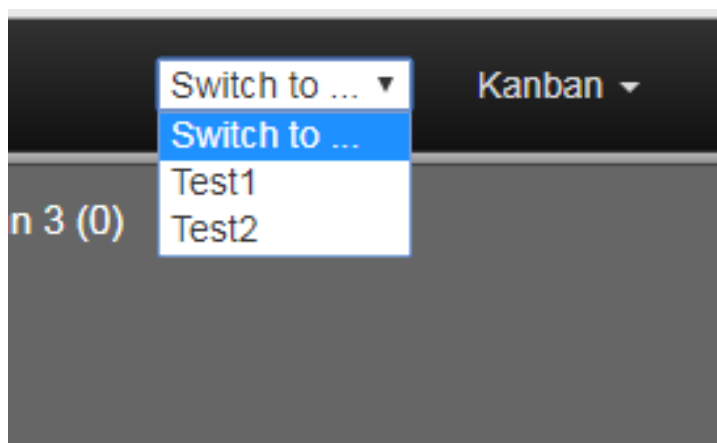


Рисунок 4.4 – Перемикання між робочими проєктами

При виборі пункту меню Themes у головному меню Kanban відображається форма (рис. 4.5), де користувач може обрати зовнішній вигляд подання дошки поточного проєкту компанії.

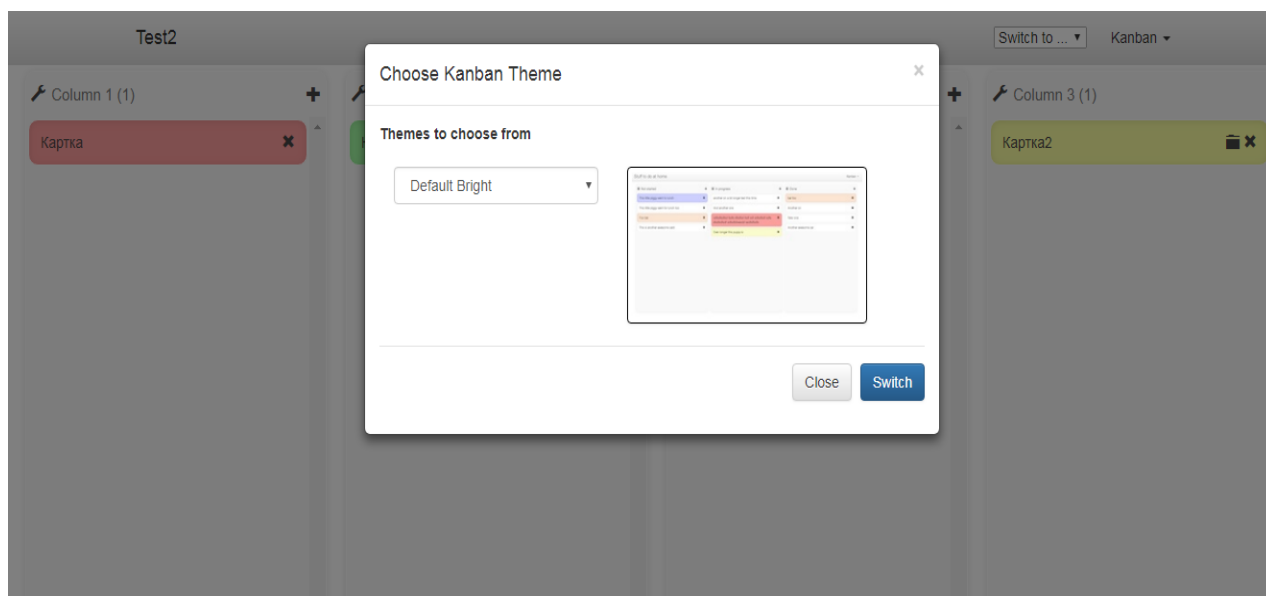


Рисунок 4.5 – Зміна зовнішнього вигляду подання дошки поточного проєкту

Для архівування карток з метою збереження інформації про реалізовані та завершені задачі та робочі процеси поточного проєкту відповідні картки можна архівувати. Для цього картка повинна знаходитися у правому стовпці дошки, який відображає процеси, близькі до завершення. При натисканні на піктограмі архівування у зображенні відповідної картки вона буде додана до архіву.

Для перегляду архіву карток необхідно у головному меню Kanban обрати пункт меню Archive (рис. 4.6).

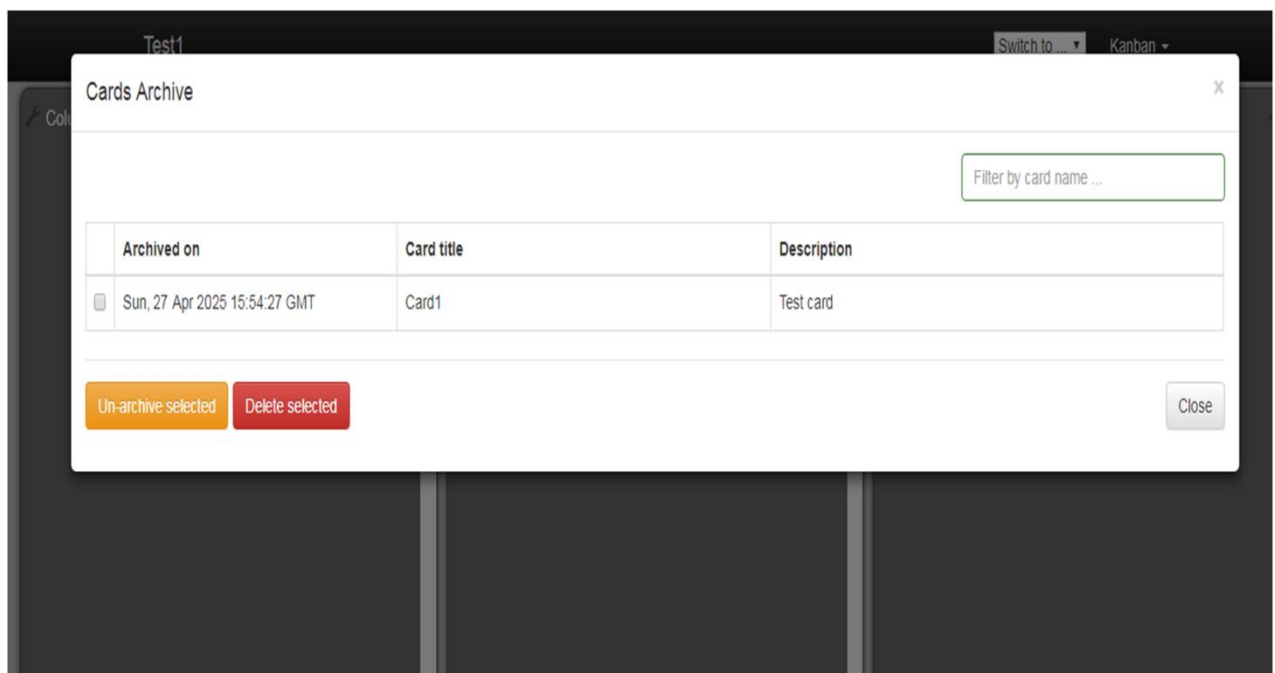


Рисунок 4.6 – Форма архівованих карток

4.5 Тестування програмного забезпечення для підтримки управління робочими процесами

Виконано тестування програмного забезпечення для підтримки управління робочими процесами.

Тестування програмного забезпечення для підтримки управління робочими процесами виконано у різних веббраузерах та різних операційних системах. На рис. 4.7 наведено відображення системи у веббраузері Opera.

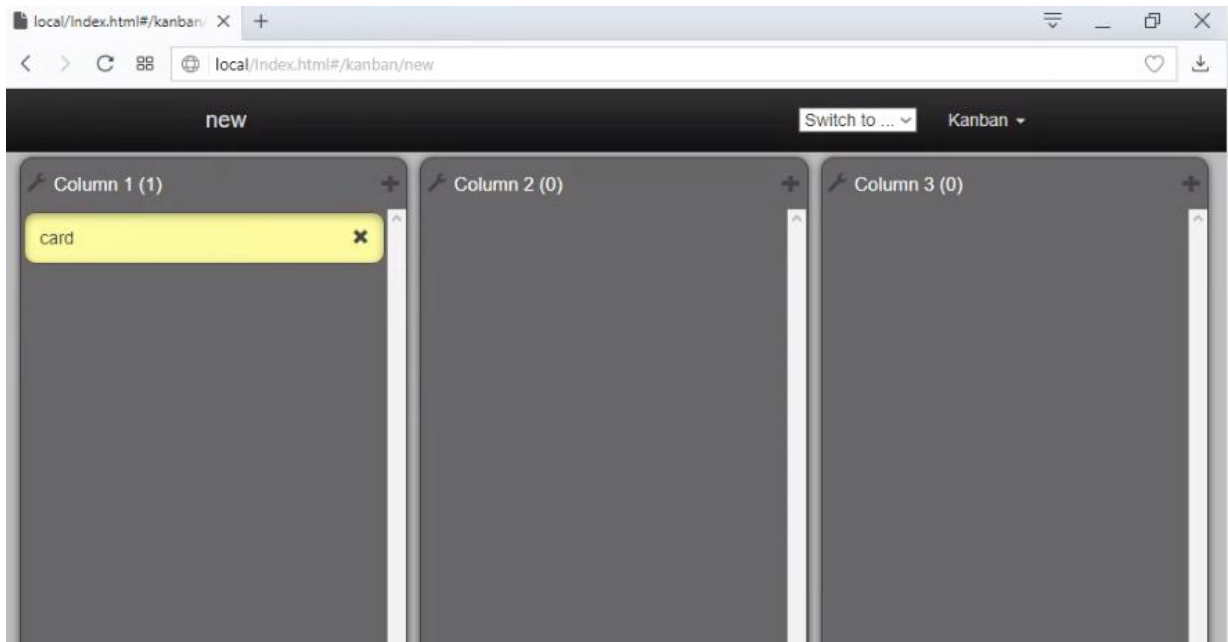


Рисунок 4.7 – Відображення програмного забезпечення для підтримки управління робочими процесами у веббраузері Opera

Виявлено, що програма для підтримки управління робочими процесами функціонує правильно та злагоджено. Вона реалізує всі функціональні вимоги та успішно виконує свою основну задачу підтримки процесу управління робочими процесами.

Розроблене програмне забезпечення дозволяє забезпечувати автоматизацію процесів, пов'язаних з підтримкою процесу управління процесами на підприємстві.

4.6 Висновки за розділом 4

Описано програмне забезпечення для підтримки управління робочими процесами. Виконано тестування розробленого програмного забезпечення для підтримки управління робочими процесами. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою управління процесами на підприємстві.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки програмного забезпечення для підтримки управління робочими процесами.

Відзначено, що забезпечення ефективного управління робочими процесами потребує використання спеціалізованого програмного забезпечення, яке дозволяє автоматизувати та координувати виконання завдань. Використання таких технологій сприяє підвищенню продуктивності та забезпечує безперервний контроль за всіма етапами процесів. Завдяки інтеграції з іншими інформаційними системами створюється єдине середовище для обміну даними та синхронізації дій між учасниками.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для підтримки управління робочими процесами. Проте деякі програмні засоби для підтримки управління робочими процесами характеризуються високою вартістю та складністю впровадження. Ще одним недоліком програмних засобів для підтримки управління робочими процесами є можливі труднощі з інтеграцією, оскільки не всі програмні рішення легко поєднуються з уже використовуваними корпоративними системами, що може спричинити проблеми з обміном даними та призвести до затримок у роботі. Не менш важливою проблемою є залежність від постачальника програмного забезпечення підтримки управління робочими процесами. Якщо розробник програмного забезпечення для підтримки управління робочими процесами припиняє супровід або змінює модель ліцензування, компанія може зіткнутися з труднощами у підтримці та масштабуванні своїх процесів. Тому актуальною є розробка програмного забезпечення для підтримки управління робочими процесами.

Сформульовано функціональні вимоги до програмного забезпечення для підтримки управління робочими процесами.

Для реалізації програмного забезпечення для підтримки управління робочими процесами обрано мову програмування JavaScript, яка дозволяє виконувати код безпосередньо у браузері без необхідності встановлення додаткового програмного забезпечення. Крім того, JavaScript підтримує об'єктно-орієнтоване, функціональне та імперативне програмування, що робить цю мову гнучкою для розробки різних типів застосунків. Іншою перевагою є підтримка асинхронної моделі обробки подій, яка дозволяє ефективно працювати з великими обсягами даних, вебсокетами та мережевими запитами без блокування потоку виконання.

Для створення програмного забезпечення для підтримки управління робочими процесами обрано фреймворк AngularJS, що надає зручну можливість створення модульних застосунків завдяки своїй архітектурі та дозволяє розробникам розбивати великі проєкти на малі та взаємопов'язані компоненти, що робить код легшим для підтримки та тестування, а також спрощує масштабування системи в міру розширення робочих процесів.

Розроблено програмне забезпечення для підтримки управління робочими процесами.

Запропоновано структуру програмного забезпечення для підтримки управління робочими процесами, що складається з шести основних модулів, які забезпечують повний функціонал системи: модуль створення та налаштування робочих панелей, модуль перемикання між створеними дошками, модуль взаємодії з картками завдань, модуль налаштування зовнішнього вигляду системи, модуль експорту та імпорту даних, модуль архівації завдань.

Описано функціонування програмного забезпечення для підтримки управління робочими процесами за допомогою діаграми послідовностей. Діаграма послідовностей ілюструє основні етапи взаємодії користувача з програмним забезпеченням для підтримки управління робочими процесами.

Описано особливості реалізації програмного забезпечення для підтримки управління робочими процесами. Виконано проєктування

інтерфейсу взаємодії користувача з програмним забезпеченням для підтримки управління робочими процесами.

Виконано тестування розробленого програмного забезпечення для підтримки управління робочими процесами. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою управління процесами на підприємстві.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. What is Task Management [Electronic resource]. – Access mode: <https://www.geeksforgeeks.org/what-is-task-management/>.
2. What is Task Management? Definition, Steps, Importance, Benefits, Tools and Techniques [Electronic resource]. – Access mode: <https://ideascale.com/blog/task-management-definition/>.
3. What is Task Management? A Beginners Guide [Electronic resource]. – Access mode: <https://www.profit.co/blog/task-management/what-is-task-management-a-beginners-guide/>.
4. What is Task Management? How to Break a Project Down into (Actionable) Tasks [Electronic resource]. – Access mode: <https://plan.io/blog/what-is-task-management/>.
5. Effortlessly track time spent on your tasks and projects [Electronic resource]. – Access mode: <https://www.monitask.com/en/Home/Task-Tracking-Software>.
6. Best Task Management Software [Electronic resource]. – Access mode: <https://software.fish/project-management-software/task-management-software>.
7. What Is Task Management? 10 Key Tools and More [Electronic resource]. – Access mode: <https://www.coursera.org/articles/task-management>.
8. Best Task Management Software Reviewed For 2025 [Electronic resource]. – Access mode: <https://thedigitalprojectmanager.com/tools/best-task-management-software/>.
9. Best Task Management Software [Electronic resource]. – Access mode: <https://www.forbes.com/advisor/business/software/best-task-management-software/>.
10. Best task management app [Electronic resource]. – Access mode: <https://www.techradar.com/best/best-task-management-apps-of-year>.
11. Best Task Management Software [Electronic resource]. – Access mode: <https://www.g2.com/categories/task-management-software>.

12. MeisterTask [Electronic resource]. – Access mode:
<https://www.meistertask.com/>.
13. Kintone [Electronic resource]. – Access mode:
<https://www.kintone.com/en-us/>.
14. Backlog [Electronic resource]. – Access mode:
<https://nulab.com/backlog-lp/task-management/>.
15. JavaScript modules [Electronic resource]. – Access mode:
<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Modules>.
16. JavaScript Debugging in the browser [Electronic resource]. – Access mode: <https://javascript.info/debugging-chrome>.
17. AngularJS Tutorial for Beginners: Learn AngularJS Step by Step [Electronic resource]. – Access mode: <https://www.guru99.com/angularjs-tutorial.html>.
18. AngularJS MVC Architecture [Electronic resource]. – Access mode: https://www.tutorialspoint.com/angularjs/angularjs_mvc_architecture.htm.

ДОДАТОК А
Технічне завдання

Вступ

Програмне забезпечення може використовуватися для підтримки управління робочими процесами.

А.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Програмне забезпечення для підтримки управління робочими процесами», затверджене наказом Національного університету «Запорізька політехніка» № 209 від 28 квітня 2025 р.

А.2 Призначення розробки

Програмний продукт призначений для підтримки управління робочими процесами.

А.3 Основні вимоги до програми, що розробляється

А.3.1 Вимоги до функціональних характеристик

Програмне забезпечення для підтримки управління робочими процесами забезпечує виконання таких функцій:

- підтримка можливості керування проєктами та відображення інформації про них у вигляді Канбан дошки;
- підтримка керування робочими процесами проєктів;
- можливість зміни зовнішнього вигляду відображення проєктів та задач;
- можливість перемикання між проєктами;
- підтримка можливості архівування карток проєкту з метою збереження інформації про реалізовані та завершені задачі та робочі процеси.

А.3.2 Вимоги до інтерфейсу програми

Інтерфейс програмного забезпечення для підтримки управління робочими процесами повинен бути зручним для користувачів. Повинна бути забезпечена можливість роботи в візуальному режимі.

А.3.3 Вимоги до надійності

Програмне забезпечення для підтримки управління робочими процесами повинно забезпечити надійне функціонування.

А.3.4 Умови експлуатації

Для експлуатації програмного забезпечення для підтримки управління робочими процесами необхідна наявність персонального комп'ютера.

А.3.5 Вимоги до складу та параметрів технічний засобів

Для роботи програмного забезпечення для підтримки управління робочими процесами потрібні такі технічні та програмні ресурси: обчислювальні потужності серверів або хмарних платформ, які забезпечують виконання основних операцій і обробку даних у реальному часі. Необхідна відповідна програмна інфраструктура, включаючи операційні системи, системи керування базами даних та інструменти для автоматизації бізнес-процесів. Важливу роль відіграють мережеві ресурси, що забезпечують безперебійну передачу даних між користувачами та програмним забезпеченням для підтримки управління робочими процесами.

A.3.6 Вимоги до маркування і пакування

Програма для підтримки управління робочими процесами може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва програми – «Програмне забезпечення для підтримки управління робочими процесами».

A.4 Вхідні дані до роботи

Вхідними даними до програмного забезпечення для підтримки управління робочими процесами є інформація про робочі процеси у вигляді карток та дошки.

Вихідними даними програмного забезпечення для підтримки управління робочими процесами є додана до системи інформація про робочі процеси у вигляді карток та дошки.

ДОДАТОК Б
Фрагмент тексту програми

```

class D12X {
  constructor() {
    this.boards = [];
    this.init();
  }

  init() {
    const savedData = localStorage.getItem('kanbanData');
    if (savedData) {
      this.boards = JSON.parse(savedData);
    }
    this.render();
  }

  save() {
    localStorage.setItem('kanbanData',
JSON.stringify(this.boards));
  }

  addBoard(n99z) {
    this.boards.push({ id: Date.now(), name: n99z, cards: []
});

    this.save();
    this.render();
  }

  addCard(g0p1, z77w) {
    const board = this.boards.find(b => b.id === g0p1);
    if (board) {
      board.cards.push({ id: Date.now(), text: z77w,
archived: false });
      this.save();
      this.render();
    }
  }
}

```

```

editCard(bId, cId, nText) {
  const board = this.boards.find(b => b.id === bId);
  if (board) {
    const card = board.cards.find(c => c.id === cId);
    if (card) {
      card.text = nText;
      this.save();
      this.render();
    }
  }
}

removeCard(bId, cId) {
  const board = this.boards.find(b => b.id === bId);
  if (board) {
    board.cards = board.cards.filter(c => c.id !== cId);
    this.save();
    this.render();
  }
}

archiveCard(bId, cId) {
  const board = this.boards.find(b => b.id === bId);
  if (board) {
    const card = board.cards.find(c => c.id === cId);
    if (card) {
      card.archived = true;
      this.save();
      this.render();
    }
  }
}

changeStyle(cssString) {
  document.getElementById('kanbanStyles').innerHTML =
cssString;
}

```

```

exportBoards() {
    return JSON.stringify(this.boards, null, 2);
}

importBoards(jsonData) {
    try {
        this.boards = JSON.parse(jsonData);
        this.save();
        this.render();
    } catch (e) {
        console.error("Import failed", e);
    }
}

render() {
    const container =
document.getElementById('kanbanContainer');
    container.innerHTML = '';
    this.boards.forEach(board => {
        const boardEl = document.createElement('div');
        boardEl.className = 'board';
        boardEl.innerHTML = `

### ${board.name}</h3>`; const cardContainer = document.createElement('div'); cardContainer.className = 'cards'; board.cards.forEach(card => { const cardEl = document.createElement('div'); cardEl.className = `card ${card.archived ? 'archived' : ''}`; cardEl.innerText = card.text; const delBtn = document.createElement('button'); delBtn.innerText = 'X'; delBtn.onclick = () => this.removeCard(board.id, card.id);


```

```

        const archiveBtn =
document.createElement('button');
        archiveBtn.innerText = 'A';
        archiveBtn.onclick = () =>
this.archiveCard(board.id, card.id);

        cardEl.appendChild(delBtn);
        cardEl.appendChild(archiveBtn);
        cardContainer.appendChild(cardEl);
    });

    boardEl.appendChild(cardContainer);
    container.appendChild(boardEl);
});
}
}

const kanban = new D12X();

document.getElementById('addBoardBtn').addEventListener('click',
() => {
    const name = prompt('Enter board name:');
    if (name) kanban.addBoard(name);
});

document.getElementById('exportBtn').addEventListener('click', ()
=> {
    const exportedData = kanban.exportBoards();
    navigator.clipboard.writeText(exportedData);
    alert('Data copied to clipboard');
});

document.getElementById('importBtn').addEventListener('click', ()
=> {
    const jsonData = prompt('Paste the exported data:');
    if (jsonData) kanban.importBoards(jsonData);
});

```

```

    moveCard(bId, cId, targetBId) {
        const sourceBoard = this.boards.find(b => b.id === bId);
        const targetBoard = this.boards.find(b => b.id ===
targetBId);
        if (sourceBoard && targetBoard) {
            const cardIndex = sourceBoard.cards.findIndex(c =>
c.id === cId);
            if (cardIndex > -1) {
                const [card] =
sourceBoard.cards.splice(cardIndex, 1);
                targetBoard.cards.push(card);
                this.save();
                this.render();
            }
        }
    }

    changeStyle(cssString) {
        document.getElementById('kanbanStyles').innerHTML =
cssString;
    }

    exportBoards() {
        return JSON.stringify(this.boards, null, 2);
    }

    importBoards(jsonData) {
        try {
            this.boards = JSON.parse(jsonData);
            this.save();
            this.render();
        } catch (e) {
            console.error("Import failed", e);
        }
    }

```

```

render() {
    const container =
document.getElementById('kanbanContainer');
    container.innerHTML = '';
    this.boards.forEach(board => {
        const boardEl = document.createElement('div');
        boardEl.className = 'board';
        boardEl.innerHTML = `

### ${board.name}</h3>`; const cardContainer = document.createElement('div'); cardContainer.className = 'cards'; board.cards.forEach(card => { const cardEl = document.createElement('div'); cardEl.className = `card ${card.archived ? 'archived' : ''}`; cardEl.innerText = card.text; const delBtn = document.createElement('button'); delBtn.innerText = 'X'; delBtn.onclick = () => this.removeCard(board.id, card.id); const archiveBtn = document.createElement('button'); archiveBtn.innerText = 'A'; archiveBtn.onclick = () => this.archiveCard(board.id, card.id); const unarchiveBtn = document.createElement('button'); unarchiveBtn.innerText = 'U'; unarchiveBtn.onclick = () => this.unarchiveCard(board.id, card.id); cardEl.appendChild(delBtn); cardEl.appendChild(archiveBtn);


```

```

        cardEl.appendChild(unarchiveBtn);
        cardContainer.appendChild(cardEl);
    });

    boardEl.appendChild(cardContainer);
    container.appendChild(boardEl);
});
}
}

const kanban = new D12X();

document.getElementById('addBoardBtn').addEventListener('click',
() => {
    const name = prompt('Enter board name:');
    if (name) kanban.addBoard(name);
});

document.getElementById('exportBtn').addEventListener('click', ()
=> {
    const exportedData = kanban.exportBoards();
    navigator.clipboard.writeText(exportedData);
    alert('Data copied to clipboard');
});

document.getElementById('importBtn').addEventListener('click', ()
=> {
    const jsonData = prompt('Paste the exported data:');
    if (jsonData) kanban.importBoards(jsonData);
});

```

ДОДАТОК В
Слайди презентації

**Міністерство освіти і науки України
Національний університет «Запорізька політехніка»
Кафедра програмних засобів**

Програмне забезпечення для підтримки управління робочими процесами

Виконав: Євгеній МОРМУЛЬ
ст. групи КНТ-112сп

Керівник: Матвій ІЛЬЯШЕНКО
к.т.н., доцент

Рисунок В.1 – Слайд 1

Об'єкт, предметта мета роботи

Об'єкт дослідження – процес розробки програмного забезпечення для підтримки управління робочими процесами.

Предмет дослідження – програмні засоби для підтримки управління робочими процесами.

Метою роботи є розробка програмного забезпечення для підтримки управління робочими процесами.

2

Рисунок В.2 – Слайд 2

Завдання роботи

Для досягнення поставленої мети у кваліфікаційній роботі бакалавра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для підтримки управління робочими процесами;
- здійснити проектування програмного забезпечення для підтримки управління робочими процесами;
- створити програмне забезпечення для підтримки управління робочими процесами;
- виконати тестування розробленого програмного забезпечення для підтримки управління робочими процесами.

3

Рисунок В.3 – Слайд 3

Порівняння існуючих аналогів

Критерій порівняння	MeisterTask	Kintone	Backlog
Інтуїтивність інтерфейсу	+	+—	+
Підтримка спеціалізованих застосунків	—	+	+—
Гнучкість налаштувань	+—	+	+—
Засоби автоматизації робочих процесів	+—	+	+—
Підтримка багатоступеневих проектів	+—	+	+

4

Рисунок В.4 – Слайд 4

Постановка завдання

При розробці програмного забезпечення для підтримки управління робочими процесами необхідно забезпечити такі функціональні вимоги:

- підтримка можливості керування проєктами та відображення інформації про них у вигляді Канбан дошки;
- підтримка керування робочими процесами проєктів;
- можливість зміни зовнішнього вигляду відображення проєктів та задач;
- можливість перемикання між проєктами;
- підтримка можливості архівування карток проєкту з метою збереження інформації про реалізовані та завершені задачі та робочі процеси.

5

Рисунок В.5 – Слайд 5

Порівняння мов програмування

Критерій порівняння мов програмування	Мова програмування		
	JavaScript	Python	C#
Підтримка різних стилів програмування	+	+–	+–
Велика екосистема для веброзробки	+	+	–
Кросплатформність	+	+	+–
Зручність для створення ПЗ для підтримки управління робочими процесами	+	–	–
Асинхронна модель обробки подій	+	+–	+

6

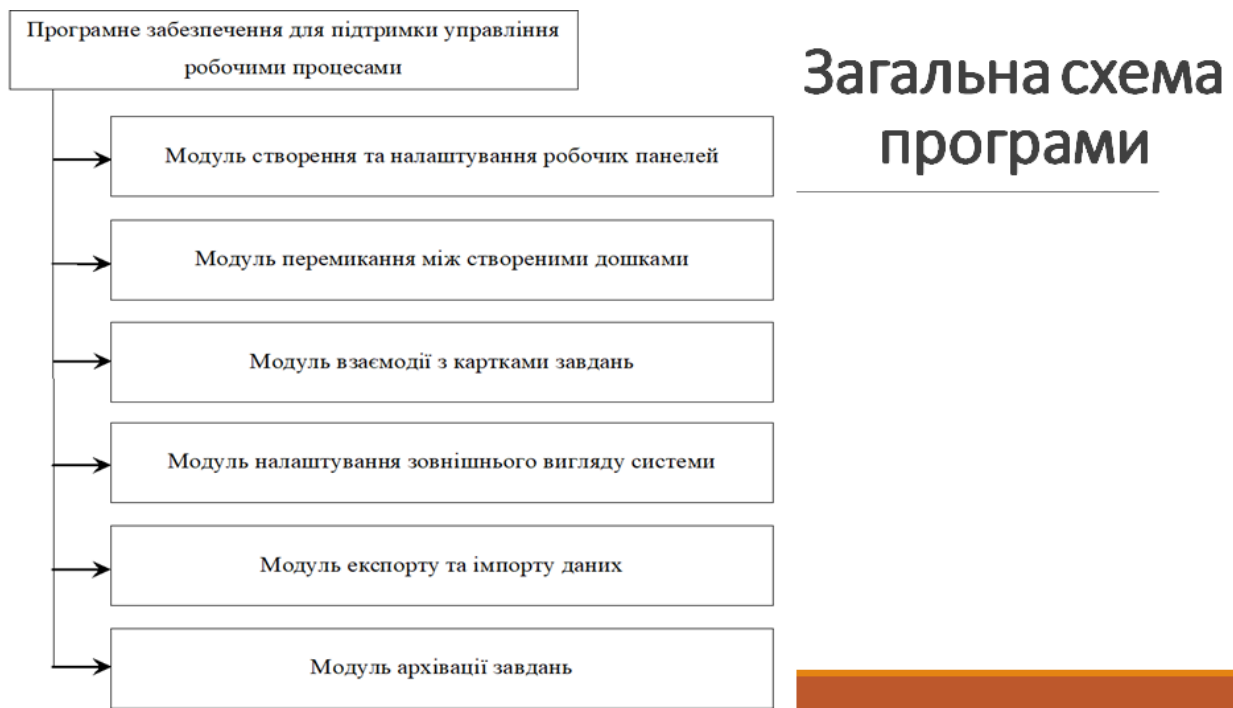
Рисунок В.6 – Слайд 6

Порівняння фреймворків

Критерій порівняння фреймворків	Фреймворки		
	AngularJS	React	Ember.js
Забезпечення модульності	+	+–	+
Вбудовані функції (маршрутизація, управління станом)	+	–	+
Документація і підтримка спільноти	+	+	+–
Легкість у вивченні	+	+–	–
Підтримка двостороннього зв'язування даних	+	–	+–

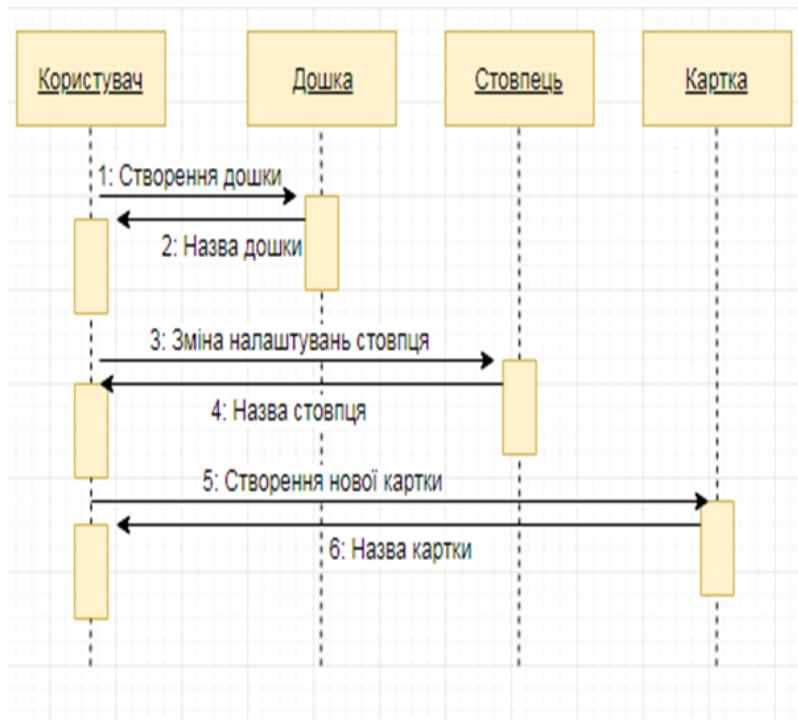
7

Рисунок В.7 – Слайд 7



8

Рисунок В.8 – Слайд 8

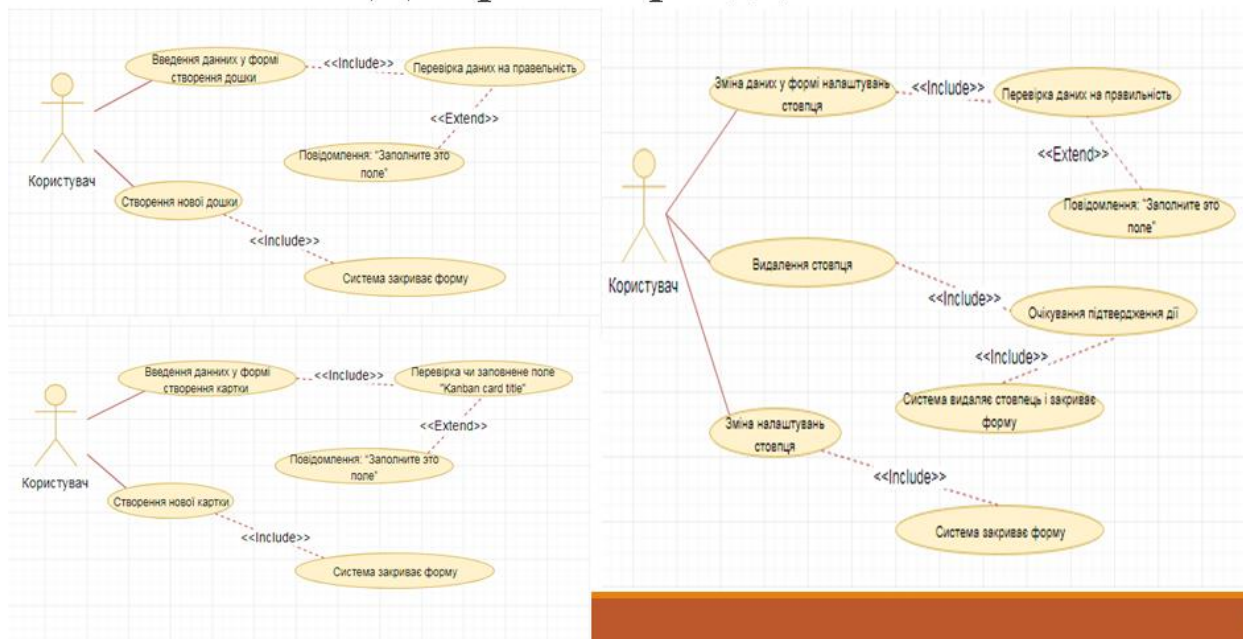


Функціонування програми

9

Рисунок В.9 – Слайд 9

Діаграми прецедентів



10

Рисунок В.10 – Слайд 10

Тестування програми

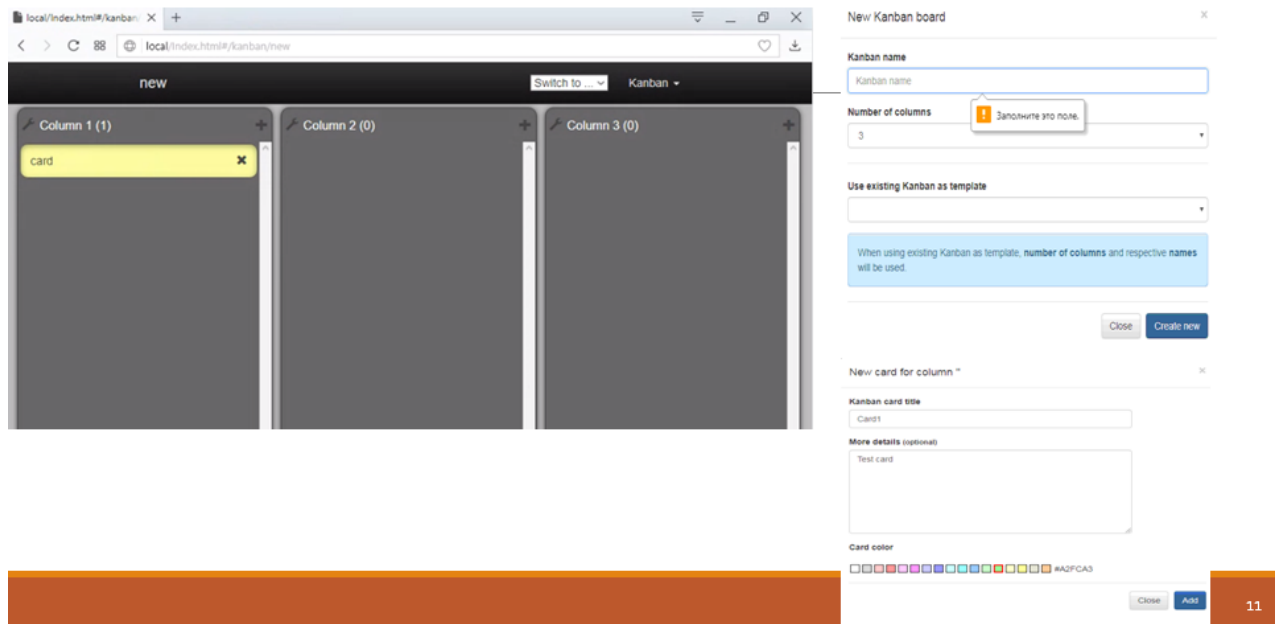


Рисунок В.11 – Слайд 11

Висновки

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки програмного забезпечення для підтримки управління робочими процесами.

Для реалізації програмного забезпечення для підтримки управління робочими процесами обрано мову програмування JavaScript та фреймворк AngularJS.

Запропоновано структуру програмного забезпечення для підтримки управління робочими процесами, що складається з шести основних модулів, які забезпечують повний функціонал системи: модуль створення та налаштування робочих панелей, модуль перемикання між створеними дошками, модуль взаємодії з картками завдань, модуль налаштування зовнішнього вигляду системи, модуль експорту та імпорту даних, модуль архівації завдань. Описано функціонування програмного забезпечення для підтримки управління робочими процесами за допомогою діаграми послідовностей.

В результаті виконання завдання було отримано систему, що відповідає всім вимогам визначеним на початку розробки та закріпленим в технічному завданні. Розроблена система є зручною для використання та виконує всі необхідні функції.

Усі завдання дипломної кваліфікаційної роботи бакалавра повністю виконано.

Рисунок В.12 – Слайд 12