МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КРИВОРІЗЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ
Фізико-математичний факультет
Кафедра інформатики та прикладної математики

«Допущено до захисту»
Завідувач кафедри
_____ Моісеєнко Н.В.
«___» _____ 2024 р.

Реєстраційний № _____
«___» _____ 20___ р.

ВЕБ-ЗАСТОСУНОК ДЛЯ УПРАВЛІННЯ ПРОЄКТАМИ

Кваліфікаційна робота студентки
групи І-20
ступінь вищої освіти бакалавр
спеціальності
014.09 Середня освіта (Інформатика)
Лубенцової Діани Дмитрівни
Керівник
доцент, к. ф.-м. н. Моісеєнко Н. В.
Оцінка:
Національна шкала _____
Шкала ECTS _____ Кількість балів _____
Голова ЕК _____ _____

Члени ЕК _____ _____
_____ _____
_____ _____
_____ _____

Кривий Ріг
2024

ЗАПЕВНЕННЯ

Я, Лубенцова Діана Дмитрівна, розумію і підтримую політику Криворізького державного педагогічного університету з академічної доброчесності. Запевняю, що ця кваліфікаційна робота виконана самостійно, не містить академічного плагіату, фабрикації, фальсифікації. Я не надавала і не одержувала недозволену допомогу під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають покликання на відповідне джерело.

Із чинним Положенням про запобігання та виявлення академічного плагіату в роботах здобувачів вищої освіти Криворізького державного педагогічного університету ознайомлена. Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі порушення академічної доброчесності робота не допускається до захисту або оцінюється незадовільно.

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному світі, де технологічний прогрес швидко набирає обертів, інтерес до інноваційних методів управління проєктами постійно зростає. Одним із таких методів є Канбан-система, яка набула особливої популярності в останні роки. Канбан дозволяє ефективно організувати управління проєктом в будь-якій сфері (чи то IT-проєкт, чи то розподіл домашніх обов'язків) та підвищити продуктивність команди.

Метою дослідження є розробити веб-застосунок для управління проєктами.

Згідно до поставленої мети дослідження були поставленні наступні **задачі дослідження:**

1) вивчити літературу з гнучкої методології управління проєктами та провести порівняльний аналіз між Scrum та Kanban;
2) проаналізувати застосунки для управління проєктами та сформулювати функціональні вимоги до розроблюваного веб-застосунку;
3) спроєктувати та програмно реалізувати веб-застосунок для управління проєктами та перевірити правильність його роботи.

Об'єкт дослідження – методи та інструменти управління проєктами.

Предмет дослідження – цифрові Канбан-дошки.

Практична цінність роботи полягає в можливості використання розробленого веб-застосунку у реальних умовах управління проєктами, що сприятиме підвищенню ефективності та організації робочого процесу. Крім того, отримані результати можуть бути використані в процесі навчання розробників програмного забезпечення.

Структура та обсяг роботи: робота складається зі вступу, двох розділів, висновків до них, загальних висновків, списку використаних джерел із 23 найменувань. Робота містить 4 таблиці та 22 рисунки. Загальний обсяг роботи 41 сторінка.

РОЗДІЛ 1
AGILE МЕТОДОЛОГІЯ РОЗРОБКИ ПРОЄКТІВ ТА ЇЇ ІНСТРУМЕНТИ

1.1. Методологія Agile. Розвиток Канбан-систем

Методологія Agile (від англ. Agile – «рухливий», «спритний», «еластичний») – це набір принципів та практик управління проєктами, спрямованих на швидке реагування на зміни та постійне вдосконалення робочих процесів [4].

Основна ідея Agile полягає в тому, щоб розробляти програмне забезпечення або виконувати проєкти у невеликих ітераціях, які називаються спринтами чи інкрементами, і пристосовувати свої плани та стратегії на основі змін у вимогах або умовах ринку.

Основні характеристики Agile включають:

- Ітеративний підхід: Проєкт розбивається на невеликі ітерації, що дозволяє швидко реагувати на зміни та використовувати фідбек для поліпшення процесів.
- Контроль ризику: Agile дозволяє зменшити ризики, виявленням їх на ранніх етапах розробки та швидким внесенням змін у плани.
- Співпраця з клієнтом: Замовник активно взаємодіє з командою розробників протягом усього процесу розробки, що дозволяє забезпечити відповідність результатів очікуванням клієнта.
- Багаторазове тестування: Програмне забезпечення тестується на кожному етапі розробки, що дозволяє виявити та виправити помилки на ранніх етапах.
- Гнучкість і адаптивність: Agile дозволяє змінювати вимоги та пріоритети в процесі розробки, щоб відповідати потребам бізнесу, які постійно змінюються.

Гнучкий процес розробки зазвичай включає кілька ключових етапів, які допомагають забезпечити ефективність та успішність проєкту незалежно від

конкретного методу Agile. Першим етапом є планування, під час якого команда обговорює майбутні ітерації або спринти та приймає рішення щодо завдань. Тут також визначаються цілі, обсяг та пріоритети роботи на наступний період.

Після планування настає етап розробки, під час якого кожна ітерація (спринт) починається з вибору завдань, які потрібно виконати протягом цього періоду. Розробники активно працюють над вибраними завданнями, включаючи програмування, тестування та інтеграцію.

Після завершення ітерації настає етап демонстрації, де команда демонструє виконану роботу замовнику або зацікавленим сторонам. Тут також здійснюється збір фідбеку щодо виконаної роботи, що є важливою складовою для подальшого вдосконалення процесу.

Після демонстрації наступає етап оцінки та відгуку, де замовник або представники бізнесу оцінюють виконану роботу та надають зворотний зв'язок команді. Від цього залежить подальша стратегія та вдосконалення проєкту.

Наприкінці циклу розробки команда готується, плануючи список завдань для наступного спринту, вибираючи пріоритети та розподіляючи ресурси. Ці етапи становлять основу агільного процесу розробки, що дозволяє команді ефективно працювати та досягати поставлених цілей.

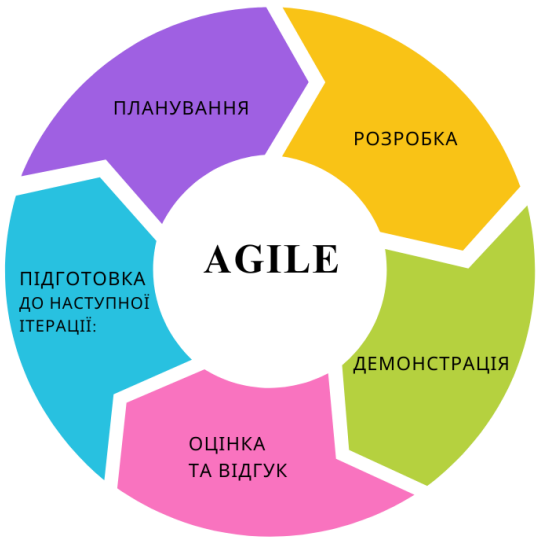

Рис.1.1. Основні етапи розробки

Канбан-система є гнучким методом управління робочим процесом, який зародився в Японії на заводі Toyota в 1950-х роках. Її концепція була розроблена Таїчі Оно як частина філософії «ощадливого виробництва» (lean manufacturing), метою якої було підвищення ефективності виробничого процесу за рахунок мінімізації відходів і непродуктивної роботи [9].

Основна ідея Канбан полягає у візуалізації потоку робіт та обмеженні роботи-в-процесі (WIP – work in progress), щоб запобігти перевантаженню та збільшити продуктивність. Система отримала свою назву від японського слова «канбан», що означає «візуальна картка» або «картка сигналів» (Рис. 1.2.).

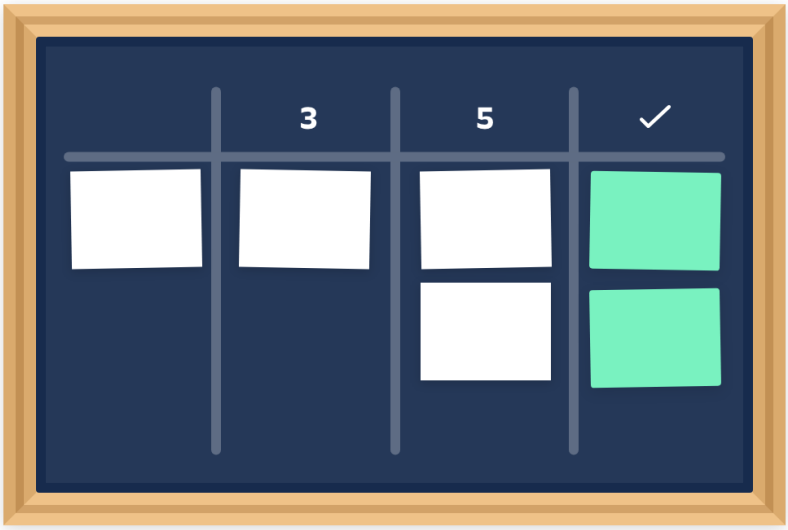

Рис.1.2. Візуальна картка

Ключові принципи Канбан, які також називають «чотирма стовпами успіху», є фундаментальними концепціями, що забезпечують ефективність і результативність Канбан-системи. Ці чотири принципи такі:

Візуалізація робочого процесу:

- Візуалізація всього потоку робіт на дошці або в програмному забезпеченні є ключовим елементом Канбан.
- Це забезпечує прозорість і загальне розуміння поточного стану проєкту чи процесу для всіх залучених сторін.

- Візуалізація допомагає виявляти вузькі місця, затримки та можливості для вдосконалення.

Обмеження роботи-в-процесі (WIP):

- Канбан обмежує кількість одночасно виконуваних завдань або робіт, щоб запобігти перевантаженню та зосередитись на завершенні поточних задач.
- Обмеження WIP допомагає уникнути багатозадачності, підвищує фокус і продуктивність команди.
- Це також сприяє швидшому виявленню і вирішенню проблем, оскільки затримки стають більш помітними.

Управління потоком:

- Канбан зосереджується на плавному і безперервному потоці робіт, замість розбиття на окремі цикли або ітерації.
- Метричні показники, такі як час циклу (cycle time) та швидкість потоку (throughput), використовуються для вимірювання та оптимізації потоку робіт.
- Основна увага приділяється усуненню вузьких місць і перешкод, що заважають плавному потоку.

Безперервне вдосконалення (кайдзен):

- Канбан підкреслює важливість постійного вдосконалення процесів та практик на основі зворотного зв'язку та аналізу метричних даних.
- Команди регулярно проводять ретроспективи (збори для аналізу) та імплементують ідеї з метою покращення процесів.
- Принцип кайдзен заохочує постійне експериментування та пошук можливостей для вдосконалення.

Ці чотири стовпи – візуалізація, обмеження WIP, управління потоком та безперервне вдосконалення утворюють міцну основу для успішного впровадження та використання Канбан-системи. Дотримання цих принципів

сприяє підвищенню ефективності, продуктивності та якості робочих процесів в організації [13].

Розвиток Канбан-систем можна розділити на кілька етапів:

1. Зародження на виробничих лініях Toyota (1950-ті роки).
2. Адаптація до програмного забезпечення та управління проєктами в IT-індустрії (1980-2000-і роки).
3. Поширення серед різноманітних галузей як гнучкого інструменту управління проєктами та робочими процесами (2000-і роки – по сьогодні).

Історія Канбан-системи має свої корені у японській промисловості, зокрема, в системі виробництва компанії Toyota. Вона була розроблена та вперше запроваджена як метод управління виробництвом на початку 1950-х років. По суті, Канбан – це система візуального управління потоком матеріалів та інформації, що базується на використанні карток або табличок.

Під час періоду після Другої світової війни, підприємства Японії, включаючи Toyota, стали стикатися з обмеженими ресурсами та обмеженим простором для виробництва. У цих умовах була суттєва необхідність в ефективному управлінні виробництвом та оптимізації робочих процесів.

Перші кроки у розвитку Канбан-системи були пов'язані з інженером Toyota Тайічі Оно. Він здійснив подорож до Сполучених Штатів Америки для вивчення системи виробництва Ford, яка базувалася на концепції «just-in-time» (точно вчасно). Повернувшись до Японії, Оно спільно зі своєю командою розробив і впровадив систему Канбан в Toyota [2].

Основною ідеєю Канбан-системи було управління виробництвом шляхом регулювання потоку матеріалів на основі реальних потреб виробництва. Кожен етап виробництва мав свою візуальну дошку (Канбан-дошку), на якій були розміщені картки з відомостями про поточний стан робочих завдань та запасів.

Вперше запроваджена на виробництві автомобілів, Канбан-система виявилася дуже ефективною в управлінні процесами виробництва, сприяючи

зменшенню запасів, покращенню продуктивності та зниженню часу циклу виробництва. Згодом вона була успішно застосована і в інших галузях промисловості, а також у сферах послуг та управління проєктами.

На початку 1980-х років, зародження Канбан в IT-галузі відбулося шляхом експериментів в компаніях, таких як Microsoft та Siemens. Ці компанії вперше спробували адаптувати основні принципи Канбан, спрямовані на візуалізацію потоку робіт та обмеження робіт у процесі, до своїх процесів розробки програмного забезпечення. Впровадження Канбан дозволило їм краще контролювати прогрес проєктів, виявляти «вузькі місця» та підвищувати продуктивність команд.

У 1990-ті роки, зародження Канбан в IT-галузі отримало подальший розвиток. З'явилися перші публікації та дослідження, присвячені застосуванню Канбан в розробці програмного забезпечення. Було детально описано використання Канбан-дошок для управління потоком робіт [1, 10]. Компанія Corbis, спеціалізована на цифровій бібліотеці зображень, успішно впровадила Канбан у свої процеси розробки програмного забезпечення.

На початку 2000-х років, Канбан став ширше використовуватися в IT-компаніях для управління проєктами та процесами розробки програмного забезпечення. Популяризації сприяли публікації Девіда Андерсона «Kanban: Successful Evolutionary Change for Your Technology Business» [19] та Стівена Чіморелли «Kanban for the Supply Chain: Fundamental Practices for Manufacturing Management» [3]. Перші спеціалізовані інструменти та програмні рішення для впровадження Канбан в IT, такі як Agile Kanban Boards, LeanKit, SwiftKanban, також стали доступні [15, 16].

Канбан почав інтегруватися з іншими гнучкими методологіями розробки програмного забезпечення, такими як Agile, Scrum та Lean. З'явилися гібридні підходи, такі як Scrumban, які дозволяли адаптувати переваги різних методологій. Компанії, такі як Toyota, Siemens, Microsoft, Zara та інші, успішно застосовували Канбан для управління IT-проєктами та процесами розробки програмного забезпечення.

Адаптація Канбан в IT-індустрії відбувалася поступово і стала результатом експериментів, досліджень та обміну досвідом між компаніями. Гнучкість та здатність візуалізувати потік робіт зробили Канбан цінним інструментом для управління проєктами та процесами розробки програмного забезпечення.

У сучасному світі Канбан використовується в багатьох сферах діяльності, таких як IT, маркетинг, фінанси, управління персоналом та інших. Популярність цієї системи зростає завдяки її простоті, гнучкості та здатності підвищити продуктивність і ефективність команд.

1.2. Огляд додатку Trello

Trello – це популярний інструмент для управління проєктами, який базується на принципах Канбан. Він надає користувачам простий і інтуїтивно зрозумілий інтерфейс для організації завдань і проєктів у вигляді візуальних дощок. Trello був розроблений компанією Fog Creek Software і запущений у 2011 році. Пізніше ця компанія перейменувалася на Trello, Inc. Творцями ідеї стали Джоел Спольські (Joel Spolsky) та Майкл Прайор (Michael Pryor), які мали на меті створити простий у використанні, але потужний інструмент для організації роботи [5].

У 2017 році компанія Trello була придбана Atlassian, однією з провідних компаній у сфері розробки програмного забезпечення. Зараз Trello продовжує розвиватися та вдосконалюватися, надаючи користувачам нові функції та можливості, а також інтегруючись з іншими продуктами в екосистемі Atlassian.

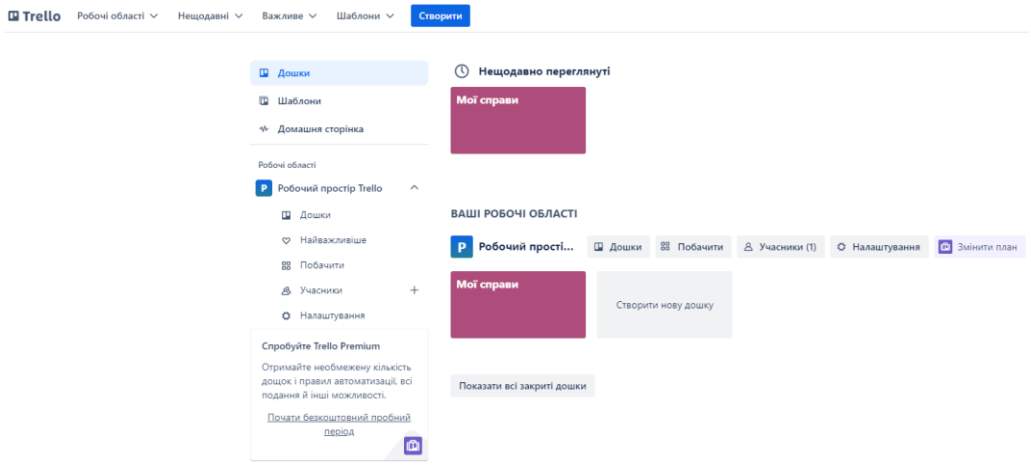

Рис. 1.3. Головна сторінка Trello

Для створення дошки в Trello необхідно увійти до свого облікового запису, натиснути на кнопку «Створити нову дошку», ввести назву, вибрати фон і, за бажанням, команду. Після цього натисніть «Створити дошку». На новій дошці додайте списки (наприклад, «To Do», «In Progress», «Done») і заповніть їх картками, що представляють окремі завдання. Картки можна налаштовувати, додаючи описи, дедлайни, вкладення та призначаючи відповідальних [14].

Trello пропонує широкий набір функцій, які роблять його потужним інструментом для управління проєктами. Користувачі можуть створювати картки для завдань, переміщувати їх між списками, встановлювати пріоритети та дедлайни, що допомагає структурувати роботу і слідкувати за виконанням завдань. Trello підтримує командну роботу, дозволяючи кільком користувачам одночасно працювати над одним проєктом, залишати коментарі, додавати вкладення та оновлювати статус завдань.

Крім того, Trello підтримує інтеграції з численними сторонніми сервісами, такими як Google Drive, Slack, Jira, що розширює його функціональність. За допомогою Butler, вбудованого інструменту автоматизації, користувачі можуть налаштовувати автоматичні дії для часто виконуваних завдань. Доступність мобільного додатку для iOS та Android

дозволяє користувачам керувати своїми проєктами на ходу, забезпечуючи гнучкість і доступність управління завданнями в будь-який час і будь-де.

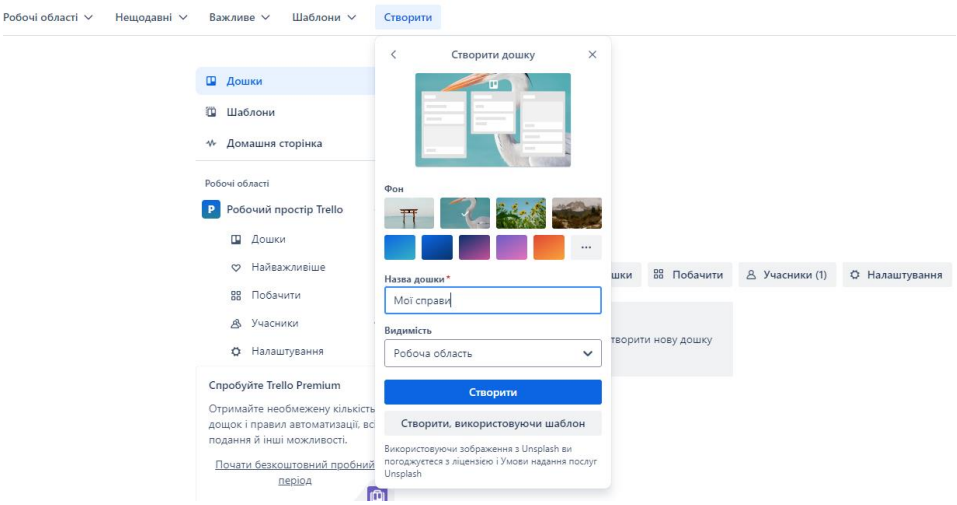

Рис. 1.4. Створення дошки у додатку Trello

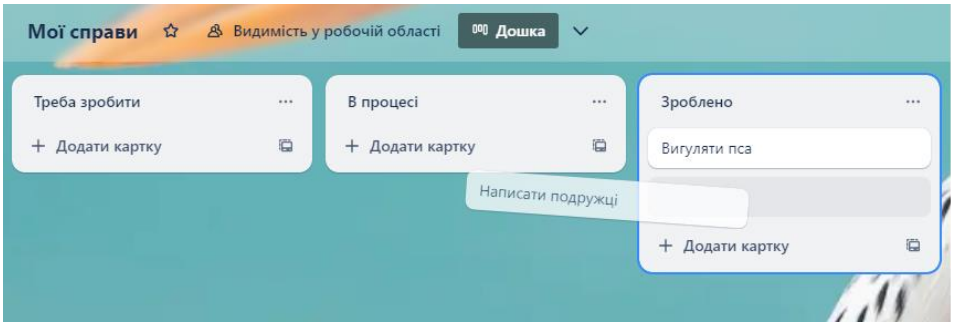

Рис. 1.5. Рух карток по стовбцях

Таблиця 1.1.

Переваги та недоліки Trello

Переваги	Недоліки
Інтуїтивний інтерфейс: Простота використання і візуальна привабливість роблять Trello доступним для широкого кола користувачів.	**Обмеженість функціоналу для великих проєктів**: Для дуже великих і комплексних проєктів Trello може бути недостатньо функціональним.

Гнучкість: Можливість налаштування дощок і карток під специфічні потреби проєкту.	**Відсутність докладної аналітики**: Хоча Trello дозволяє відслідковувати прогрес, він не надає детальних звітів та аналітики.
Співпраця: Зручні інструменти для командної роботи.	**Відсутність офлайн-доступу:** Trello доступний лише онлайн, що може бути проблемою для користувачів, які не мають постійного доступу до Інтернету.

Trello є потужним інструментом для управління проєктами, який добре підходить для невеликих та середніх команд. Його простота і гнучкість роблять його ідеальним для швидкої організації роботи і співпраці, проте для великих проєктів можуть знадобитися додаткові інструменти або інтеграції.

1.3. Огляд додатку Jira

Jira – це продукт, який з'явився завдяки внутрішній потребі вирішення проблем компанії Atlassian. У 2002 році Atlassian потребувала ефективного інструменту для відстеження помилок та управління проєктами власного розробницького циклу [18]. Тоді і виник Jira.

Початково Jira була внутрішнім інструментом Atlassian, але пізніше, побачивши його потенціал, компанія вирішила зробити його доступним для зовнішніх користувачів. Його швидко прийняли інші розробники програмного забезпечення, оскільки він пропонував потужні можливості для управління проєктами та відстеження помилок у вигляді простого та ефективного інструмента.

З роками Jira постійно розвивалася та розширювалася, надаючи користувачам нові можливості та функціональність. Її гнучка архітектура дозволяла впроваджувати різні методології управління проєктами, що робило її привабливою для різних галузей та видів діяльності.

У 2014 році Atlassian випустила хмарний варіант Jira, роблячи його ще доступнішим і простішим у використанні для користувачів. Це дозволило розширити його аудиторію та підвищити популярність серед різних видів компаній та проєктів [21]. В даний час Jira залишається одним з найпопулярніших інструментів для управління проєктами та відстеження помилок, як у сфері розробки програмного забезпечення, так і в інших галузях.

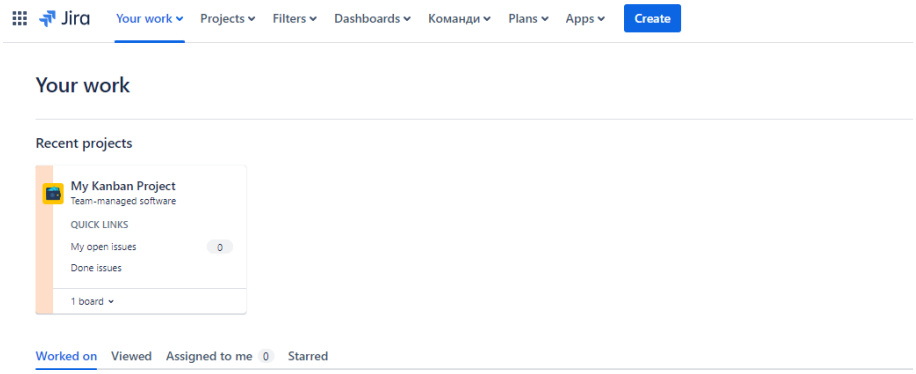

Рис. 1.6. Головна сторінка Jira

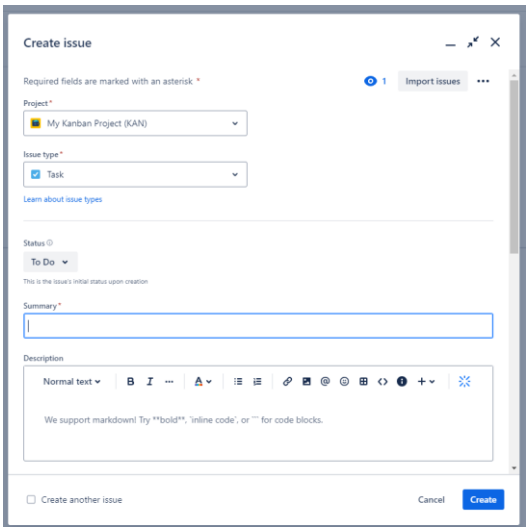

Рис. 1.7. Створення дошки у додатку Jira

Створення дошки в Jira починається з входу в систему та вибору проєкту. У розділі «Дошки» (Boards) треба натиснути «Створити дошку» (Create board) та вибрати тип дошки (Kanban або Scrum). Далі вводиться назва дошки та обираються конфігурації доступу з натисканням «Створити» (Create). Після створення дошки налаштувати колонки можна відповідно до

потреб робочого процесу, додавати завдання та встановити фільтри для відображення. Завдання можна переміщувати між колонками, оновлювати їх статуси, додавати коментарі та призначати відповідальних осіб, що робить управління проєктом ефективним та інтуїтивним.

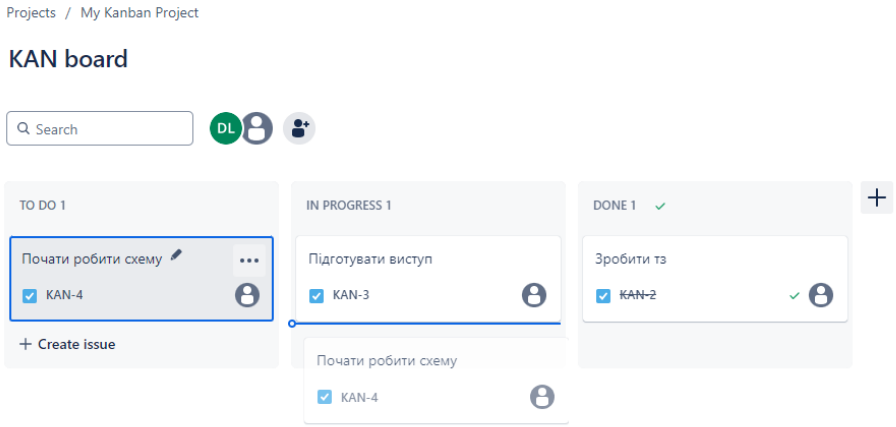

Рис. 1.8. Рух карток по стовбцях

Таблиця 1.2.

Переваги та недоліки Jira

Переваги	Недоліки
Гнучкість: Підтримка різних методологій управління проєктами, включаючи Kanban і Scrum.	**Складність використання:** Через широку функціональність та налаштування, Jira може бути складною для новачків.
Розширена функціональність: Потужні інструменти для відстеження завдань, аналітики, автоматизації та інтеграції.	**Вартість:** Висока ціна для великих команд і підприємств, особливо з додаванням платних розширень і інтеграцій.
Масштабованість: Підходить для малих команд і великих підприємств завдяки можливості налаштування і адаптації до різних потреб.	**Відсутність офлайн-доступу:** неможливість працювати без доступу до мережі

Jira є одним з найпотужніших інструментів для управління проєктами та відстеження помилок, особливо популярним серед команд, що працюють за методологіями Agile [17]. Її гнучкість, розширена функціональність і можливості інтеграції роблять її незамінним інструментом для багатьох компаній, хоча складність і вартість можуть стати перешкодою для деяких користувачів.

1.4. Огляд додатку Worksection

Worksection – це інструмент для управління проєктами та спільної роботи, призначений для команд будь-якого розміру та галузі. Заснований у 2010 році, цей додаток швидко здобув популярність завдяки своїй простоті використання та низці корисних функцій для організації робочих процесів.

Історія створення додатку Worksection має свої коріння в 2010 році, коли команда розробників та професіоналів з управління проєктами в Україні спільно працювала над розробкою власної системи управління проєктами. Основна мета була створити інструмент, який би відповідав потребам різних команд та підприємств, надаючи їм зручні та ефективні засоби для планування, виконання та відстеження робочих завдань [23].

Розробка Worksection була ініційована групою фахівців з різних сфер, таких як програмування, менеджмент проєктів, дизайн та маркетинг. Ця різноманітність компетенцій дозволила створити продукт, який би враховував потреби різних користувачів та забезпечував їм широкий спектр інструментів для співпраці та управління проєктами.

Початково Worksection був спрямований на український та російський ринки, проте з часом його популярність почала зростати, а також з'явилися користувачі з інших країн, що підштовхнуло команду до розширення функціональності та міжнародного розвитку продукту.

Сьогодні Worksection є відомим та широко використовуваним інструментом для управління проєктами, що забезпечує спільну роботу команд та підтримує ефективне планування та виконання завдань у різних

галузях та компаніях. Зусилля команди за роки розвитку привели до того, що Worksection став надійним та шанованим інструментом у своєму сегменті, надаючи користувачам потужні можливості для успішного управління проєктами.

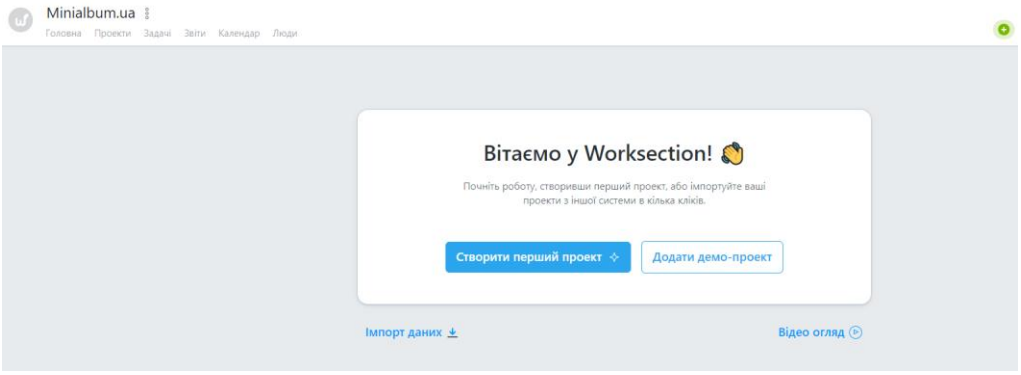

Рис. 1.9. Головна сторінка Worksection

Створення дошки в Worksection – це перший крок до ефективного управління проєктом. Після входу в систему та вибору відповідного проєкту або створення нового, користувач отримує можливість створити нову дошку. Цей процес починається з визначення типу дошки, наприклад, Kanban або Task List.

Після вибору типу дошки, користувачу надається можливість налаштувати різні параметри, такі як назва дошки, опис, колір, та інші. Після цього створюються колонки, які відображають етапи робочого процесу. Наприклад, колонки можуть включати «To Do», «In Progress», «Done» тощо. Коли колонки створені, користувач може почати додавати завдання до кожної з них.

Кожне завдання може бути детально описане, містити важливу інформацію, а також бути призначене відповідним учасникам проєкту та встановлено пріоритет. Таким чином, створення дошки в Worksection дозволяє структурувати робочі процеси та ефективно керувати завданнями для досягнення успішних результатів у проєкті.

Одним з важливих аспектів при створенні завдання є встановлення його пріоритету (Рис. 1.10). В Worksection користувач може встановити рівень важливості кожного завдання, що допомагає визначити порядок виконання та зосередити увагу на найважливіших завданнях.

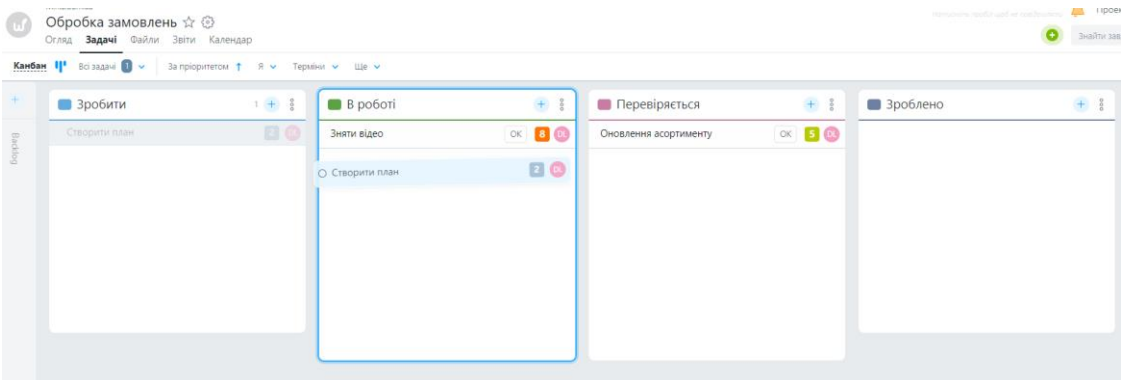

Рис. 1.10. Рух карток по стовбцях

Пріоритетність завдань може відображати їх важливість та терміновість виконання, що сприяє ефективному розподілу ресурсів та досягненню поставлених цілей у проєкті. Таким чином, надання пріоритету завданням в Worksection дозволяє команді чітко визначити їх значимість та успішно керувати робочим процесом.

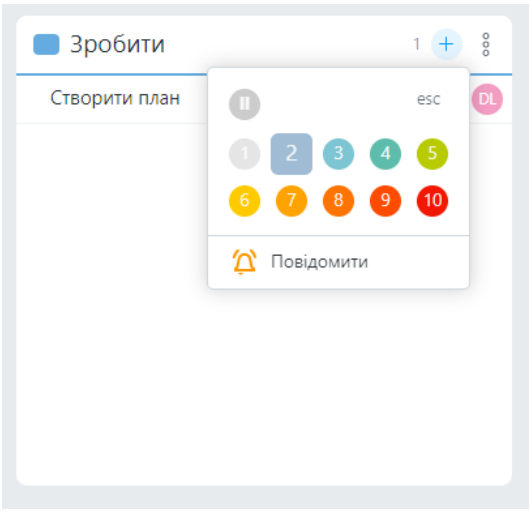

Рис. 1.11. Оцінка пріоритету

Таблиця 1.3.

Переваги та недоліки Worksection

Переваги	Недоліки
Простота використання: Має простий та інтуїтивно зрозумілий інтерфейс, що робить його доступним для користувачів будь-якого рівня досвіду.	**Відсутність деяких функцій:** не пропонує деякі функції, які є в інших інструментах управління проєктами, таких як чат, управління документами та мобільні додатки
Універсальність: Можна використовувати для управління проєктами будь-якого розміру та складності, підходить як для особистих завдань, так і для великих проєктів .	**Необхідність налаштування:** для роботи з великими обсягами даних чи об'ємними проєктами може знадобитися додаткове налаштування та оптимізація робочих процесів.
Підтримка клієнтів: Worksection пропонує якісну підтримку клієнтів, яка включає в себе онлайн-документацію, навчальні відео та цілодобову службу підтримки.	**Вартість:** Безкоштовний план пропонує обмежені функції, а платні плани можуть бути дорожчими, ніж деякі інші інструменти
Наявність зручних інструментів для співпраці: Ви можете призначати завдання, додавати коментарі, ставити мітки та пріорітети задач.	

1.5. Порівняння Scrum та Kanban

Порівняння між Scrum і Kanban – це не лише порівняння двох методологій управління проєктами, але й розгляд їхніх відмінностей та спільних рис, що дозволяють вибрати найбільш підходящий метод для конкретного проєкту чи команди. Обидві методології є популярними в сфері

розробки програмного забезпечення та інших галузях, і кожна має свої особливості та переваги [8].

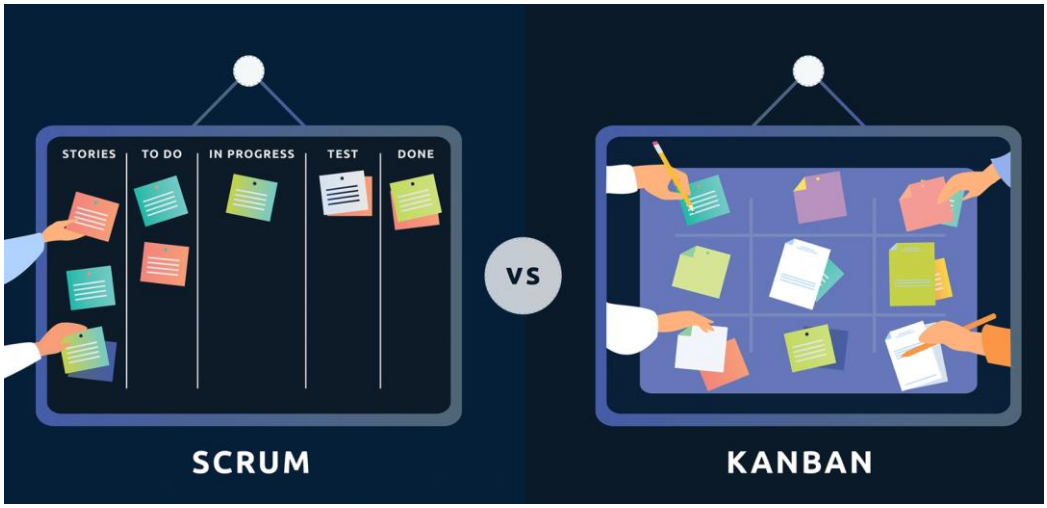

Рис. 1.12. Scrum і Kanban

Scrum – це ітеративний метод, що базується на роботі в спринтах, які зазвичай тривають від 2 до 4 тижнів. В Scrum команда зобов'язана завершити конкретний набір робіт до кінця кожного спринту, після чого проводиться ретроспектива для визначення покращень. Scrum має чіткі ролі (Product Owner, Scrum Master, і розробники), а також зустрічі, такі як Daily Standup, Sprint Planning, Sprint Review і Sprint Retrospective [7].

Kanban – це метод, що базується на візуальному управлінні завданнями та обмеженні робочого процесу (WIP). У Kanban завдання переміщуються по дошці від стадії до стадії, відображаючи поточний стан процесу. Основною метою Kanban є максимізація продуктивності, управління потоком роботи та зниження часу перебування завдань у системі.

У Scrum існують чітко визначені ролі, які включають Product Owner, Scrum Master та Розробника. Product Owner відповідає за управління продуктовим беклогом, Scrum Master забезпечує виконання процесу Scrum, а розробники відповідають за виконання завдань на спринт. У Kanban ролі менш стандартизовані, що надає більшу гнучкість команді у їх виконанні.

Таблиця 1.4.

Порівняння Scrum та Kanban

Характеристика	Scrum	Kanban
Ролі	Є чітко визначені ролі: Product Owner, Scrum Master, Розробник	Ролі не стандартизовані, більш гнучкість
Процес	Робота в ітераціях (спринтах) з фіксованою тривалістю	Більш гнучкий процес без чіткого розподілу на ітерації
Часові рамки	Робота в ітераціях з фіксованою тривалістю	Не має фіксованих часових рамок
Пріоритети завдань	Встановлюються на кожен спринт	Можливість зміни пріоритетів у будь-який момент
Формат робочої дошки	Колонки для відслідковування завдань на кожен спринт	Візуальна дошка з колонками, що відображають різні стадії процесу

Kanban має більш гнучкий процес, і він не розподіляється на фіксовані ітерації. Робота відбувається без чіткого розподілу на спринти, що дозволяє більшу гнучкість у виконанні завдань [12]. Scrum використовує процес, що базується на ітераціях, відомих як спринти, з фіксованою тривалістю. Кожен спринт має чітко визначений часовий рамки і складається зі зустрічей, таких як планування, ретроспектива та перегляд, для оцінки результатів.

Scrum робота відбувається в межах фіксованих ітерацій, кожен з яких має точно визначений початок та кінець. На відміну від цього, у Kanban не має чітко встановлених часових рамок, і завдання переміщуються через стадії процесу по мірі готовності.

У Kanban пріоритети можна змінювати у будь-який момент, що дозволяє команді швидко адаптуватися до змін вимог або ситуації, а у Scrum пріоритети завдань встановлюються на кожен спринт, зазвичай під час зустрічі планування.

Для відстеження завдань Scrum використовує дошку з колонками, які представляють різні стадії завдань на кожен спринт. Kanban також використовує візуальну дошку, але вона відображає різні стадії процесу розробки, такі як To Do, In Progress, Testing, Done, дозволяючи більш детально відслідковувати прогрес завдань [11].

Обидві методології мають свої переваги та недоліки, і вибір між ними залежить від конкретних потреб проєкту та вподобань команди. Наприклад, Scrum може бути кращим вибором для проєктів з великими командами та потребою у чіткому плануванні, тоді як Kanban може бути ефективним для проєктів з невеликою командою та потребою у гнучкому управлінні процесами.

Висновки до розділу 1

У цьому розділі ми розглянули методологію Agile, розвиток Канбан-систем, а також огляд трьох популярних інструментів для управління проєктами: Trello, Jira та Worksection.

Методологія Agile пропонує гнучкий підхід до управління проєктами, який зосереджується на швидкій адаптації до змін та постійному вдосконаленні. Її основні принципи включають ітеративний процес, контроль ризиків, співпрацю з клієнтом та багаторазове тестування.

Канбан-системи є гнучким методом управління робочим процесом, який візуалізує потік роботи та обмежує роботу в процесі (WIP). Цей метод допомагає підвищити продуктивність, скоротити час циклу та покращити якість роботи.

Trello – це простий у використанні інструмент для управління проєктами, який базується на принципах Канбан. Він пропонує візуальні дошки, списки та картки для організації завдань та співпраці з командою.

Jira – це потужний інструмент для управління проєктами та відстеження помилок, який пропонує широкий спектр функцій для планування, виконання та відстеження завдань. Він добре підходить для команд, які використовують методології Agile, такі як Scrum.

Worksection – це універсальний інструмент для управління проєктами, який пропонує простий інтерфейс, зручні функції для співпраці та підтримку різних методологій управління проєктами.

Вибір найкращого інструменту для управління проєктами залежить від потреб вашої команди та специфіки вашого проєкту. Trello може бути хорошим вибором для невеликих команд, які потребують простого та візуального інструменту. Jira – це потужний інструмент, який добре підходить для команд, які використовують Scrum або інші методології Agile. Worksection – це універсальний інструмент, який може бути використаний командами будь-якого розміру та для проєктів будь-якої складності.

РОЗДІЛ 2
ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-ЗАСТОСУНКУ ДЛЯ УПРАВЛІННЯ ПРОЄКТАМИ

2.1. Інтерфейс користувача та його проєктування

Інтерфейс користувача (User Interface, UI) є одним із найважливіших елементів будь-якої програми, зокрема Kanban-дошки. Від його якості залежить зручність використання та ефективність роботи користувачів [20]. У цьому підрозділі розглянемо основні принципи проєктування UI для цифрової Kanban-дошки.

Принцип простоти є фундаментальним у проєктуванні інтерфейсу користувача. Інтерфейс повинен бути інтуїтивно зрозумілим і легким у використанні навіть для новачків. Користувачі повинні мати можливість швидко орієнтуватися в інтерфейсі та знаходити необхідні функції без потреби в тривалому навчанні або додаткових інструкціях.

Для досягнення простоти ми використовували мінімальну кількість елементів, які не відволікають увагу користувача від основних завдань. В макеті використали зрозумілі піктограми та мітки, що легко асоціюються з відповідними функціями. Врахували структурованість проєкту з достатнім простором між елементами для запобігання перевантаженню інформацією.

Остаточний дизайн нашого інтерфейсу користувача складається з кількох ключових компонентів:

- Заголовок: Містить назву поточної дошки, кнопку для відкриття/закриття бічної панелі, перемикач автоматичного збереження даних, а також кнопки для збереження, видалення поточної дошки та відкриття налаштувань.
- Бічна панель: Розташована зліва та дозволяє переглядати список усіх наявних дошок. Також надає можливість додавати нові дошки через текстове поле введення.

- Область карток: Центральна частина інтерфейсу, де відображаються всі картки поточної дошки. Користувач може додавати нові картки, вводячи їх назву у текстове поле.
- Картка: Візуальне представлення окремої картки задачі, що містить її назву, список завдань та можливість додавати нові завдання.
- Контекстне меню картки: Випливаюче меню, що відкривається після натискання правої кнопки миші на картці та дозволяє виконувати дії, такі як видалення або дублювання картки.
- Модальні вікна: Використовуються для відображення додаткової інформації або підтвердження дій користувача, наприклад, видалення дошки або очищення картки.

Рис. 2.1. Макет інтерфейсу

Адаптивність інтерфейсу означає його здатність ефективно працювати на різних пристроях та екранах різного розміру. Це особливо важливо в сучасному світі, де користувачі можуть працювати з додатком на комп'ютерах, планшетах та смартфонах. Ми проводили тестування макету на різних пристроях для забезпечення його коректної роботи на всіх платформах.

Візуальна привабливість інтерфейсу є важливим аспектом, який впливає на загальне враження користувачів від роботи з додатком. Гарний дизайн підвищує задоволеність користувачів та сприяє тривалому використанню програми. В нашому проєкті Kanban-дошки ми врахували кілька ключових принципів для забезпечення візуальної привабливості.

Правильний вибір кольорової палітри є ключовим для створення естетично привабливого інтерфейсу, який не перевантажує зір. Ми обрали приємну для очей кольорову схему, що складається з кількох основних кольорів, які доповнюють один одного (Рис. 2.2.).

Рис. 2.2. Палітра кольорів

Синій колір часто асоціюється з почуттям спокою, миру та впевненості, цей колір використовується для фону, кнопок та тексту. Білий колір використовується для тексту та значків, він робить інформацію легко читабельною і контрастує з синім фоном.

В нашому проєкті цифрової Kanban-дошки ми особливу увагу приділили уніфікації всіх елементів інтерфейсу для забезпечення цілісного та професійного вигляду програми. Всі кнопки, поля введення, картки завдань та інші інтерактивні елементи мають узгоджені форми та стилі. Це допомагає користувачам легко впізнавати функціональність кожного елемента без необхідності в додатковому навчанні.

2.2. Розробка функціоналу Kanban-дошки

Розробка функціоналу Kanban-дошки включає в себе створення логіки та механізмів, що дозволяють користувачам ефективно управляти своїми завданнями. Для реалізації нашого проєкту «Kards» ми обрали клієнтський підхід, використовуючи HTML, CSS та JavaScript. Цей вибір був обумовлений кількома важливими факторами, які сприяли створенню простого, ефективного та навчального інструменту для управління завданнями.

Однією з ключових переваг клієнтського підходу є простота розгортання. Відсутність серверної частини означає, що весь додаток працює виключно на стороні клієнта, тобто в браузері користувача. Це має кілька важливих переваг, наприклад, користувачі можуть отримати доступ до додатку без необхідності встановлення додаткового програмного забезпечення або налаштування серверного оточення. Вони можуть просто відкрити додаток у своєму браузері і відразу почати користуватись ним. Також для розгортання додатку достатньо завантажити файли HTML, CSS та JavaScript на веб-сервер. Це значно спрощує процес впровадження та знижує витрати на підтримку додатку [6].

Для реалізації Kanban-дошки було створено кілька ключових компонентів у вигляді JavaScript класів. Board – клас, що представляє окрему дошку з колекцією карток (Рис. 2.3.). Card – клас для картки задачі на дошці, що містить заголовок та список завдань (Рис. 2.4.). Item – клас для окремого завдання на картці (Рис. 2.5.).

Для представлення канбан-дошки у додатку було використано ієрархічну структуру даних. На найвищому рівні знаходиться об'єкт «Board», який зберігає інформацію про окрему дошку, таку як її назва, ідентифікатор та налаштування. Кожна дошка може містити багато об'єктів «Card», які представляють окремі картки задач. В свою чергу, кожна картка може мати багато об'єктів «Item», що відображають окремі завдання всередині картки.

```
class Board {
    constructor(name, id, settings, identifier=0) {
        this.name = name;
        this.id = id;
        this.settings = settings;
        this.cards = []; // Колекція карток на дошці
        this.identifier = identifier;
    }
    // ...
}
```

Рис. 2.3. Клас Board

```
class Card {
    constructor(name, id, parentBoardId) {
        this.name = name;
        this.items = []; // Список завдань на картці
        this.id = id;
        this.parentBoardId = parentBoardId;
    }

    addItem(item) {
        this.items.push(item);
        renderCard(this.id); // Перерендеринг картки
    }

    renderItems() {
        // Генерує HTML-розмітку для списку завдань
    }

    generateElement() {
        // Генерує HTML-розмітку для всієї картки
    }
    // ...
}
```

Рис. 2.4. Клас Card

Кожна картка візуалізується як окремий блок зі своєю назвою та списком завдань. Завдання відображаються як окремі елементи списку всередині блоку картки. Користувач може легко додавати нові картки та завдання, а також редагувати їх назви.

```
class Item {
    constructor(title, description=null, id, parentCardId) {
        this.title = title;
        this.description = description;
        this.id = id;
        this.isDone = false;
        this.parentCardId = parentCardId;
    }

    check(chk=true) {
        // Позначає завдання як виконане/невиконане
    }

    update() {
        // Додає обробники подій для завдання
    }
}
```

Рис. 2.5. Клас Item

Візуалізація Kanban-дошки відбувається за допомогою функцій, які генерують HTML-розмітку для компонентів на основі поточного стану даних (Рис. 2.6.).

```
function renderBoard(board) {
    // Оновлює поточну дошку та її назву
    // ...
    renderAllCards();
}

function renderAllCards() {
    // Перерендеринг всіх карток на дошці
    for (let _card of currentCards()) {
        let _generated = _card.generateElement();
        e_cardsContainer.insertBefore(_generated, ...);
        _card.update(); // Оновлення обробників подій
    }
}

function renderCard(cardID) {
    // Перерендеринг окремої картки
    let _card = currentCards().find(e => e.id === cardID);
    if (_card) {
        let _generated = _card.generateElement();
        // Заміна HTML-розмітки картки
    }
    _card.update(); // Оновлення обробників подій
}
```

Рис. 2.6. Генерація HTML-розмітки

Однією з ключових функцій канбан-дошки є можливість переміщувати картки та завдання між різними колонками або всередині тієї самої колонки. Для реалізації цього функціоналу було використано механізм перетягування (drag-and-drop) за допомогою відповідних подій миші у JavaScript.

При перетягуванні елементу (картки або завдання) змінюється його позиційне CSS-оформлення на «absolute», щоб від'єднати елемент від основного потоку документа. Під час перетягування відстежуються координати курсору миші та відповідно коригується позиція елементу, створюючи ефект «присмоктування» до курсору.

Після завершення перетягування визначається, над яким елементом (карткою або завданням) знаходиться курсор, і відповідно оновлюється структура даних додатку. Після оновлення даних відбувається перерендеринг усієї дошки для відображення змін.

Для реалізації перетягування карток та завдань реалізовані функції (Рис. 2.7.).

Для забезпечення додаткової функціональності, такої як видалення або дублювання карток, було реалізовано контекстне меню, що викликається під час натискання правої клавіші миші на картці. Контекстне меню відображається у вигляді невеликого випливаючого вікна з відповідними опціями.

Механізм контекстного меню реалізовано за допомогою подій миші у JavaScript. При натисканні правої клавіші визначається, над якою карткою знаходиться курсор, і відповідно будується та відображається контекстне меню. Після вибору опції у меню викликається відповідна функція для обробки цієї дії.

Для реалізації контекстного меню картки використовуються наступні функції (Рис. 2.8.).

```
const cardDrag_startDragging = (e) => {
    // Виконується при початку перетягування елементу
    cardDrag_mouseDown = true;
    cardDrag_mouseDownOn = e.target;
    // Зміна позиційного CSS для перетягування
    toggleHoverStyle(true); // Відображення ефекту наведення
};

const cardDrag_stopDragging = (e) => {
    // Виконується при завершенні перетягування
    toggleHoverStyle(false); // Приховування ефекту наведення

    // Визначення цільового місця для переміщення
    let _hoverCard = getMouseOverCard();
    let _hoverItem = getMouseOverItem();

    // Оновлення структури даних відповідно до нового розташування
    if (_hoverCard) {
        let _hoverCardObject = getCardFromElement(_hoverCard);
        let _heldItemObject = getItemFromElement(cardDrag_mouseDownOn);

        // Логіка переміщення елементу
        if (_hoverItem) {
            _hoverCardObject.items.move(...);
        } else {
            _hoverCardObject.items.push(_heldItemObject);
        }

        renderCard(_hoverCardObject.id);
        renderCard(_heldItemObject.getParentCard().id);
    }

    cardDrag_mouseDown = false;
    // Скидання стилів для перетягування
};

const cardDrag_update = (e) => {
    // Функція, що викликається під час перетягування
    // для оновлення позиції елементу відповідно до курсору
};
```

Рис. 2.7. Перетягування карток та завдань

Ці функції забезпечують відображення контекстного меню, його приховування при натисканні поза його межами та виконання дій при виборі опцій меню, таких як видалення картки.

Одним з ключових компонентів інтерфейсу користувача додатку «Kards» є модальні вікна. Вони відіграють важливу роль у забезпеченні

зручної взаємодії з користувачем та отриманні підтвердження перед виконанням критичних операцій.

```
const cardContextMenu_show = (e) => {
    // Відображає контекстне меню у позиції курсору миші
    cardContextMenu_currentCard = getMouseOverCard();
    e_cardContextMenu.style.top = mouseY + 'px';
    e_cardContextMenu.style.left = mouseX + 'px';
    e_cardContextMenu.classList.add('visible');
};

const cardContextMenu_hide = (e) => {
    // Приховує контекстне меню, якщо клік був поза його межами
    if (e.target.offsetParent != e_cardContextMenu) {
        e_cardContextMenu.classList.remove('visible');
    }
};

const cardContextMenu_deleteCard = () => {
    // Виконується при виборі опції "Видалити картку"
    createConfirmDialog('Are you sure to delete this card', () => {
        let _currentCardObject = getCardFromElement(cardContextMenu_currentCard);
        currentCards().splice(currentCards().indexOf(_currentCardObject), 1);
        renderCard(_currentCardObject.id);
    });
}
```

Рис. 2.8. Контекстне меню

У додатку використовуються два типи модальних вікон:

Вікно підтвердження дії з'являється, коли користувач намагається виконати дію, яка може призвести до втрати даних або має деструктивний характер, наприклад, видалення дошки чи очищення картки. Вікно підтвердження містить текстовий опис дії, яку збирається виконати користувач, та дві кнопки: «Підтвердити» та «Скасувати».

Якщо користувач натискає «Підтвердити», виконується відповідна дія (видалення чи очищення). Якщо ж натиснути «Скасувати», модальне вікно закривається, і жодних дій не відбувається.

Використання вікна підтвердження дозволяє уникнути випадкового видалення або модифікації важливих даних, забезпечуючи додатковий крок перевірки перед виконанням незворотних операцій.

Вікно налаштувань відкривається при натисканні на кнопку налаштувань у заголовку додатку. Воно призначене для відображення та

редагування різноманітних налаштувань користувача, таких як ім'я користувача, тема інтерфейсу тощо. Вікно налаштувань містить форму з полями введення для редагування відповідних параметрів. Після внесення змін користувач може зберегти нові налаштування або скасувати дію.

Модальні вікна були реалізовані за допомогою HTML, CSS та JavaScript. Для їх відображення використовується напівпрозора підкладка (оверлей), що затемнює решту вмісту сторінки, та діалогове вікно у центрі екрану. Анімації відкриття та закриття вікон забезпечують плавний та привабливий перехід (Рис. 2.9.).

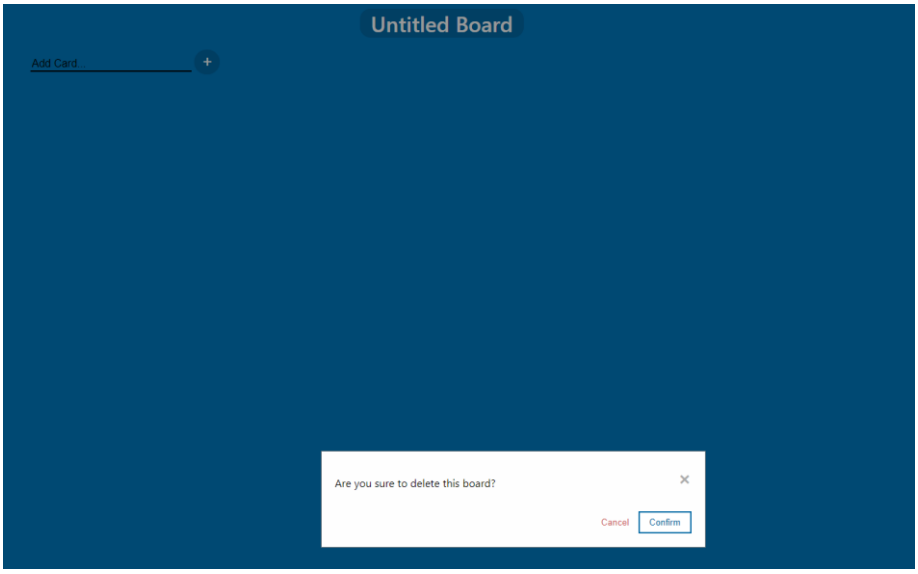

Рис. 2.9. Напівпрозора підкладка

Застосування модальних вікон у веб-додатку «Kards» дозволяє забезпечити зручний та безпечний спосіб взаємодії з користувачем, одночасно запобігаючи випадковим діям та втраті даних (Рис. 2.10.).

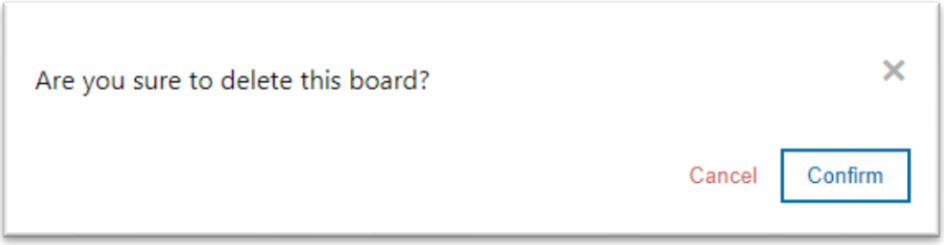

Рис. 2.10. Модальне вікно

Для забезпечення постійного доступу до даних канбан-дошки було реалізовано функціонал збереження та завантаження даних у локальному сховищі браузера (Local Storage). Дані додатку серіалізуються у форматі JSON і зберігаються у Local Storage при кожній зміні (в автоматичному режимі або при явному збереженні користувачем).

При завантаженні додатку відбувається зворотній процес: дані зчитуються з Local Storage, десеріалізуються з JSON і відновлюється повна структура даних додатку. Завдяки цьому всі дошки, картки та завдання користувача стають доступними після повторного входу у додаток.

Цей підхід забезпечує постійність даних, але водночас обмежує доступність додатку лише однією платформою (браузером і машиною), оскільки дані зберігаються локально.

2.3. Сфера застосування веб-застосунку для управління проєктами

Розроблений веб-застосунок «Kard» може знайти застосування в різноманітних сферах для візуалізації та керування як робочими процесами та проєктами так і особистими. Канбан-дошка, що лежить в основі додатку, є гнучким та ефективним інструментом для планування, відстеження прогресу та співпраці в команді.

Одним з очевидних напрямків застосування «Kards» є управління проєктами у сфері інформаційних технологій. Завдяки своїй гнучкості та здатності адаптуватися до різних методологій, канбан-дошка може використовуватися як для традиційних каскадних моделей розробки програмного забезпечення, так і для ітеративних та гнучких підходів, таких як Scrum або Agile.

Команди розробників можуть створювати окремі дошки для різних проєктів або модулів системи, візуалізуючи потік завдань від початкової стадії планування до остаточного випуску. Картки можуть представляти окремі функції, завдання з усунення помилок або історії користувачів, а їх переміщення між колонками відображатиме поточний стан виконання [22].

Крім того, канбан-дошка дозволяє обмежувати кількість одночасно виконуваних завдань у кожній колонці, що допомагає уникнути перевантаження та забезпечити належну концентрацію зусиль команди.

Канбан-дошка також може бути корисною для візуалізації та оптимізації різноманітних бізнес-процесів у будь-якій галузі. Картки можуть представляти окремі завдання, замовлення, запити або навіть документи, що проходять через певні етапи обробки.

Наприклад, у сфері управління ланцюгами поставок картки можуть відображати окремі партії товарів, а колонки – різні стадії, такі як виробництво, транспортування, митне оформлення та доставка. Це дозволяє наочно відстежувати рух товарів і виявляти вузькі місця у процесі.

В офісному середовищі канбан-дошка може використовуватися для керування завданнями, документообігом або процесами найму персоналу, забезпечуючи прозорість та підвищуючи ефективність співпраці між різними підрозділами.

Нарешті, «Kards» може бути корисним інструментом для особистого планування та підвищення продуктивності. Користувачі можуть створювати власні дошки для організації своїх завдань, проєктів або навіть побутових справ.

Картки можуть представляти окремі завдання або цілі, а колонки – їх статус або пріоритет. Перетягування карток між колонками дозволяє наочно відстежувати прогрес та фокусуватися на найбільш важливих завданнях.

Завдяки простоті використання та гнучкості налаштувань наш проєкт може стати зручним помічником для планування та організації діяльності в різних сферах життя.

Таким чином, розроблений веб-додаток «Kards» має широкі можливості застосування для управління проєктами, бізнес-процесами та особистою продуктивністю завдяки зручній візуалізації робочих потоків у вигляді канбан-дошки та можливості легкого переміщення завдань між різними стадіями виконання.

Висновки до розділу 2

Робота над проєктом цифрової Kanban-дошки виявилася досить важливим та цікавим завданням, оскільки вона охоплювала різні аспекти, починаючи від проєктування інтерфейсу користувача та закінчуючи розробкою функціональності та застосуванням для управління проєктами. В ході роботи були враховані ключові принципи дизайну UI, такі як простота, зручність використання та візуальна привабливість.

Проєктування UI базувалося на ідеї мінімалізму та інтуїтивності, щоб забезпечити зручність взаємодії з користувачем. Кожен елемент інтерфейсу був ретельно продуманий для оптимальної ефективності та зрозумілості. Ми врахували адаптивність інтерфейсу для різних пристроїв та його візуальну привабливість, щоб зробити роботу з додатком приємною та комфортною для користувачів.

У розділі про розробку функціональності Kanban-дошки було розглянуто клієнтський підхід з використанням HTML, CSS та JavaScript. Цей підхід дозволив створити простий та ефективний інструмент для управління завданнями. Були розглянуті ключові компоненти додатку, такі як дошки, картки та завдання, а також механізми їх відображення та взаємодії.

Застосування Kanban-дошки для управління проєктами показало свою ефективність у різних галузях, починаючи від розробки програмного забезпечення і закінчуючи управлінням ланцюгами поставок та офісними процесами. Вона дозволяє командам ефективно організовувати робочі процеси, відстежувати прогрес та співпрацювати над проєктами, забезпечуючи прозорість та підвищену продуктивність.

У цілому, проєкт «Kards» є значним внеском у розробку інструментів для управління завданнями та проєктами. Його гнучкість, простота та ефективність роблять його корисним інструментом як для професійних команд, так і для особистого використання, сприяючи підвищенню продуктивності та успішному виконанню завдань.

ВИСНОВКИ

У процесі виконання роботи та дослідження проблеми розробки веб-застосунку для управління проєктами розв’язані всі поставлені завдання та отримані результати, узагальнення яких надає можливість зробити наступні висновки.

1. В процесі вивчення літератури з гнучкої методології управління проєктами та проведення порівняльного аналізу між Scrum та Kanban було обрано для розробки застосунку Kanban метод як більш гнучкий та універсальний.

2. Аналіз застосунків для управління проєктами таких як Trello, Jira та Worksection дав можливість сформулювати функціональні вимоги до розроблюваного веб-застосунку як до цифрової канбан-дошки.

3. Спроєктований та програмно реалізований веб-застосунок для управління проєктами та перевірена правильність його роботи

Практичним результатом роботи є гнучкий та ефективний інструмент, який допомагає командам досягати більшої продуктивності та успішно виконувати завдання в будь-якій сфері діяльності.

Розробка цифрової Kanban-дошки включала в себе не лише створення зручного інтерфейсу користувача, але й реалізацію різноманітних функціональних можливостей, спрямованих на полегшення управління завданнями та проєктами. Використання передових методів розробки програмного забезпечення та увага до деталей дозволили створити потужний інструмент, який відповідає вимогам сучасного бізнесу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Agile practice guide. Project Management Institute, 2017. 210 p.
2. Agile Development at Scale: The Next Frontier. URL: https://www.researchgate.net/publication/331448369_Agile_Development_at_Scale_The_Next_Frontier (дата звернення: 12.11.2023).
3. Cimorelli S.C. Kanban for the Supply Chain: Fundamental Practices for Manufacturing Management. Productivity Press, 2005. 144 p.
4. Classical Project Management vs Agile Project Management. URL: https://www.visual-paradigm.com/scrum/classical-vs-agile-project-management/ (дата звернення: 6.12.2023).
5. Collaboration Software. URL: https://www.computerworld.com/article/3226447/what-is-trello-a-guide-toatlassians-collaboration-and-work-management-tool.html. (дата звернення: 2.11.2023).
6. How to choose a right software architecture. URL: https://techbeacon.com/app-dev-testing/top-5-software-architecture-patternshow-make-right-choice (дата звернення: 12.12.2023).
7. Iterative Model: What Is It And When Should You Use It? URL: https://airbrake.io/blog/sdlc/iterativemodel (дата звернення: 5.11.2023).
8. Kanban. URL: https://www.agilealliance.org/kanban (дата звернення: 6.11.2023).
9. Lean manufacturing. URL: https://www.techtarget.com/searcherp/definition/lean-production#:~:text=Lean%20manufacturing%20is%20a%20methodology,not%20willing%20to%20pay%20for (дата звернення: 17.12.2023).
10. Poppendieck M., Poppendieck T.D., Poppendieck T. Lean Software Development: An Agile Toolkit. Addison-Wesley Professional, 2003. 203 с.
11. Scrum URL: http://www.mountaingoatsoftware.com/scrum (дата звернення: 26.11.2023).

12. The 2020 Scrum Guide. URL: https://scrumguides.org/scrum-guide.html#scrum-definition (дата звернення: 3.11.2023).
13. The Kanban Method. URL: https://kanban.university/kanban-guide/ (дата звернення: 10.12.2023).
14. The Trello Rest API. URL: https://developer.atlassian.com/cloud/trello/rest/api-group-actions/ (дата звернення: 29.11.2023).
15. Todaro D. The Epic Guide to Agile: More Business Value on a Predictable Schedule with Scrum. R9 Publishing LLC, 2019. 518 p.
16. What is the Agile methodology? URL :https://www.atlassian.com/agile/ (дата звернення: 2.11.2023).
17. WHAT IS AGILE? WHAT IS SCRUM? URL: https://www.cprime.com/resources/what-is-agile-what-isscrum/ (дата звернення: 9.11.2023).
18. What Is Jira: An Overview of a Unique Project Management Tool. URL: https://www.fool.com/theblueprint/what-is-jira/ (дата звернення: 20.11.2023).
19. Андерсон Д. Дж. Канбан. Успішні еволюційні зміни для вашого технологічного бізнесу. Фабула, 2021. 288 с.
20. Інтерфейс користувача. URL: https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0 (дата звернення: 9.01.2024).
21. Що таке Jira і як з нею працювати URL: https://iampm.club/ua/blog/shho-take-jira-i-yak-z-neyu-praczyuvati/ (дата звернення: 27.11.2023).
22. Як дошки Канбан працюють в IT-секторі. URL: https://speka.media/yak-kanban-doski-pracyut-v-it-sektori-pyxnqv (дата звернення: 15.11.2023).

23. Як український сервіс Worksection підвищує ефективність бізнесу. URL: https://ain.ua/2023/05/30/yak-ukrayinskyj-servis-worksection-pidvyshhuye-efektyvnist-biznesu/ (дата звернення: 4.12.2023).