МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«РЕАЛІЗАЦІЯ МОДУЛЯ KANBAN ДЛЯ СИСТЕМИ УПРАВЛІННЯ ІТ-ПРОЕКТАМИ ЗАСОБАМИ WEB-ТЕХНОЛОГІЙ»**

Виконав(ла): студент(ка) 2 курсу, групи 8.1219
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми інженерія програмного забезпечення
(назва освітньої програми)
В.К.Волков
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, доцент, к.ф.-м.н. Горбенко В.І.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент Завідувач кафедри фундаментальної математики, доцент, д.т.н. Гребенюк С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти магістр

Спеціальність 121 інженерія програмного забезпечення

(шифр і назва)

Освітня програма інженерія програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

Лісняк А.О.

(підпис)

« » 2020 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Волкову Владиславу Костянтиновичу

(прізвище, ім'я та по-батькові)

1. Тема роботи (проекту) Реалізація модуля Kanban для системи управління IT-проектами засобами web-технологій

керівник роботи (проекту) Горбенко В.І., к.ф.-м.н., доцент

(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 20 » травня 2020 року № 576-с

2. Строк подання студентом роботи 30.11.20

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1.Постановка задачі.

2.Основні теоретичні відомості.

3.Реалзація програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	01.09.2020	
2.	Збір вихідних даних.	10.09.2020	
3.	Обробка методичних та теоретичних	08.10.2020	
	джерел.		
4.	Розробка першого розділу.	20.10.2020	
5.	Розробка другого та третього розділу.	20.11.2020	
6.	Оформлення та нормоконтроль		
	кваліфікаційної роботи.	30.11.2020	
7.	Захист кваліфікаційної роботи.	15.12.2020	

Студент _____ В.К.Волков
(підпис) (ініціали та прізвище)

Керівник роботи _____ В.І.Горбенко
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.В.Кудін
(підпис) (ініціали та прізвище)

.

РЕФЕРАТ

Кваліфікаційна робота магістра «Реалізація модуля Kanban для системи управління IT-проектами засобами web-технологій»: 35 с., 32 рис., 1 табл., 13 джерел.

KANBAN, WEB-ДОДАТОК, VUE.JS, ПРОГРАМУВАННЯ.

Об’єкт дослідження – kanban модуль.

Мета роботи: визначити функціональність та показати послідовність розробки модуля Kanban для системи управління IT-проектами засобами web-техгологій.

Метод дослідження – теоретичний, експериментальний.

Методологія канбан широко використовується в IT проектах. В роботі описана методологія канбан та розглянуті додатки створені на основі методології. Наведені переваги та недоліки кожного з них, а також представлене порівняння канбан з конкурентом скрам. У роботі детально описано створення власного модуля канбан та наведено приклади роботи програми.

SUMMARY

Master’s Qualification Thesis «Implementation of Kanban module for web-based IT project management system»: 35 pages, 32 figures, 1 tables, 13 references.

KANBAN, WEB-APPLICATION, VUE.JS, PROGRAMMING.

The object of the study is kanban module.

The aim of the study is to determinate the functionality and show the sequence of development of the Kanban module for web-based IT project management system.

The methods of research are theoretical, experimental.

Kanban methodology is widely used in IT projects. The paper describes the methodology of kanban and considers applications created on the basis of the methodology. The advantages and disadvantages of each of them are given, as well as a comparison of kanban with a competitor scrum. The paper describes in detail the creation of your own kanban module and gives examples of the program.

ЗМІСТ

ВСТУП

Тема даної роботи присвячена розробці додатку для IT проектів з використанням методології канбан. Канбан – це інструмент для розробки проектів, який допомагає налагодити поточні процеси і не перевантажувати команду. Незавершені задачі не простоюють і рухають по ланцюгу створення проекту чи його підтримки. Канбан вносить в робочі процеси в команді прозорість і сфокусованість. Завдяки візуалізації процесів з'являється ясність відносно того, що відбувається з процесом. Канбан одна з самих гнучких методологій управління проектами. В той час як інші системи мають цілий ряд умов, які скорочують ряд ступенів свободи, в канбан таких умов тільки дві: візуалізація процесів проектної роботи і скорочення кількості незавершених задач. Завдяки цій гнучкості канбан може бути використаний як додатковий інструмент, оптимізуючий процеси компанії.

Запропонований в роботі приклад реалізації функціоналу не є інноваційним, проте ефективний. В роботі розглянуто аналоги, та описано реалізацію власного додатку.

РОЗДІЛ 1 КАНБАН. ОГЛЯД АНАЛОГІВ

1.1 Канбан в IT

Термін Канбан прийшов до нас з Японії завдяки відомій виробничій системі Тойота. TPS(Toyota Production System) – це система керування, яка організовує виробництво і логістику. TPS базується на двох основних принципах: точно в строк (виготовляти тільки те, що потрібно і тільки тоді, коли потрібно) та автоматизація за участі людини [4]. Канбан має дослівний переклад: кан – означає видимий, візуальний, а бан – карточка чи дошка. Зазвичай прототипи канбан мають вигляд дошки з задачами, які треба виконати. Дану методологію можна описати фразою – «зменшення кількості процесів на даний момент роботи».

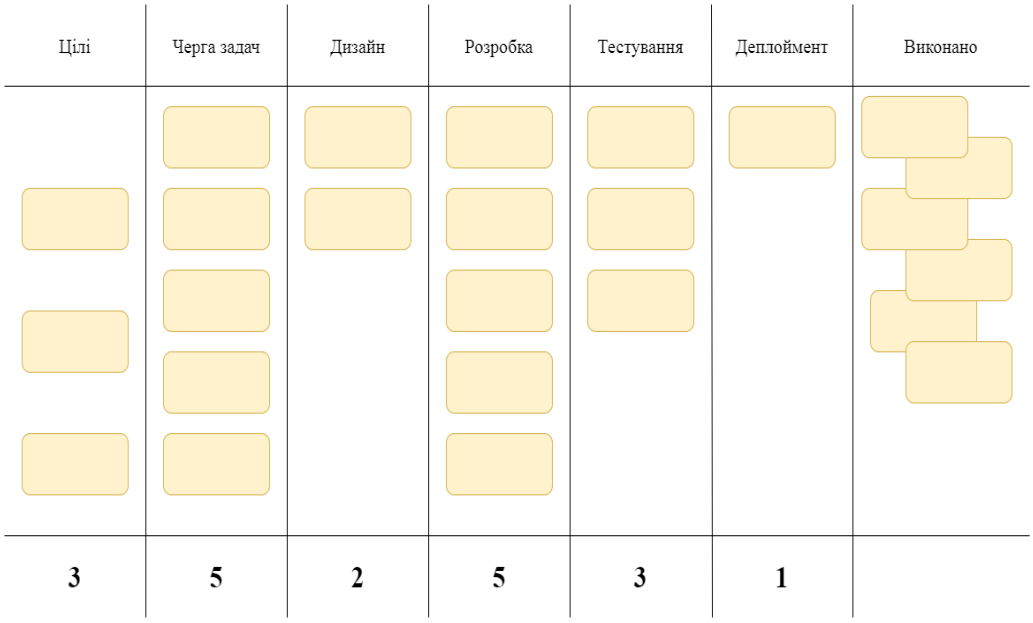

Рисунок 1.1 – Приклад дошки канбан

Цілі проекту: необов’язковий, але корисний стовбець. Сюди можна додати задачу по пришвидшенню роботи, чи іншу високорівневу задачу, яка буде не основною, але корисною.

В колонці черга задач зберігаються задачі, які можна виконувати. Вгорі колонки міститься пріоритетна задача, яка і береться на виконання, після чого карточка з задачею переміщається в інший стовбець.

Наступні колонки до «Виконано» можуть змінюватись, оскільки в процесі розробки та тестування можуть з’явитись правки, які необхідно внести в проект.

В колонці дизайн містяться задачі, для яких дизайн інтерфейсу ще не вирішений. Після того, як дизайн проекту затверджений задача переміщується в наступний стовбець.

У стовбці розробка знаходяться задачі до тих пір, доки розробка додатку чи блоку не завершена. Після завершення вона переміщується у наступний стовбець, але якщо з’являються якісь правки то задачу можна повернути на попередній стовбець.

У стовбці тестування задача знаходиться допоки тестується. Якщо знайдені помилки, то вона повертається на попередній стовбець, якщо все добре, то переміщується на наступний.

У колонці деплоймент виконується збірка проекту. Це може бути завантаження нової версії на сервер, чи завантаження проекту у репозиторій. Після того, як задача повністю виконана, вона потрапляє у стовбець виконано. Нижче задач знаходяться цифри, які означають число задач, які можуть бути одночасно у стовбцях. Цифри підбираються відповідно до кількості людей в команді. Якщо в команді 6 людей і під задачами стоїть число 6, то це означає, що кожен член команди виконує по одній задачі [3].

1.2 Огляд додатку Trello

Trello – простий додаток для командної чи індивідуальної роботи [2]. Після створення команди з головної сторінки можна створити дошку, чи запросити когось до команди.

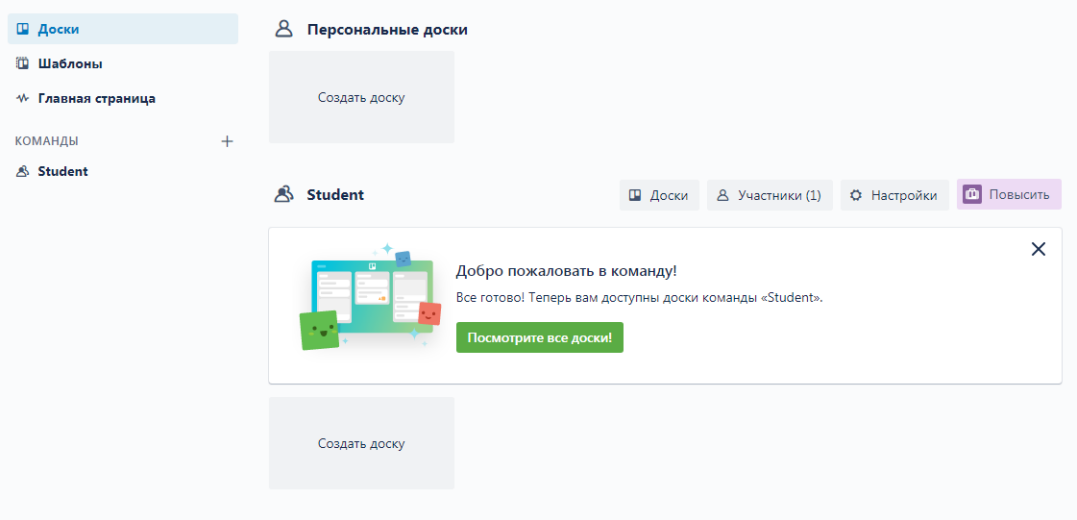

Рисунок 1.2 – Головна сторінка Trello

При створенні дошки додаток просить ввести назву дошки та обрати команду, яка буде виконувати проект.

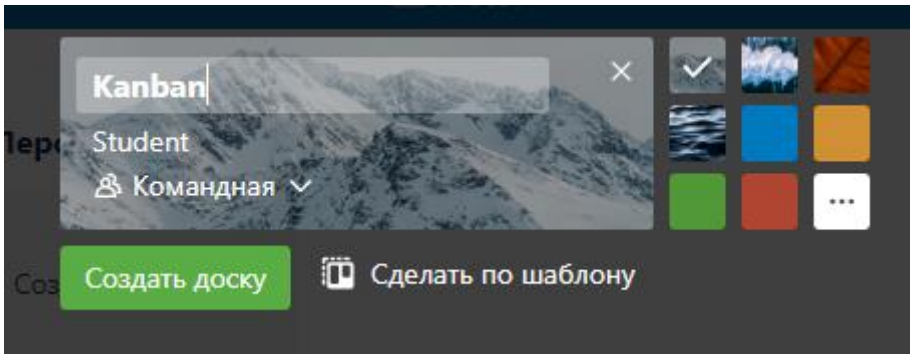

Рисунок 1.3 – Створення дошки у додатку Trello

Після чого користувач має можливість створити картки з задачами, додати учасників, які будуть виконувати задачу, та вказати строк виконання.

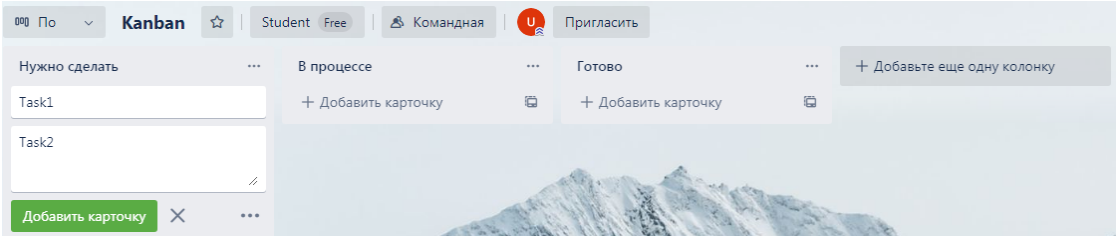

Рисунок 1.4 – Створення картки на дошці

Як і вказано у методології канбан, після виконання задачі картка переміщається у наступну колонку, в іншому випадку повертається у попередню.

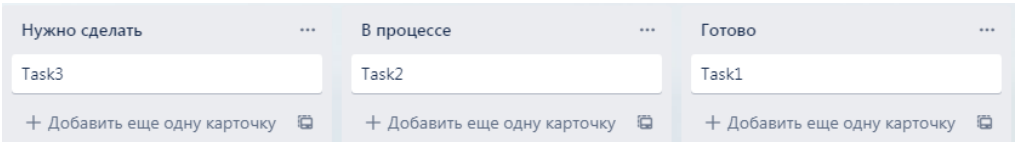

Рисунок 1.5 Рух карток по стовбцях

Перевагами даного додатку є:

1) на одній дошці може розміститись як невеликий, так і великий проект з великою кількістю карток, які можуть бути як простими задачами, так і невеликими проектами;

2) картки можна переміщати не тільки по стадіям виконання, але і по пріоритетах зміщуючи карту вгору чи вниз;

3) для певної задачі може бути призначений окремий виконавець або декілька;

4) завдяки підтримці хмарних сервісів, таких як Google Drive та Dropbox, картки можна створювати автоматично.

До недоліків додатку Trello можна віднести:

1) незручність при користуванні девайсом з невеликим екраном [5];

2) неможливість працювати без доступу до мережі;

1.3 Огляд додатку Kanban Tool

Kanban Tool – програмне забезпечення для роботи з проектами, створене на основі методології канбан. При створенні дошки необхідно вказати назву, після чого створити колонки-етапи.

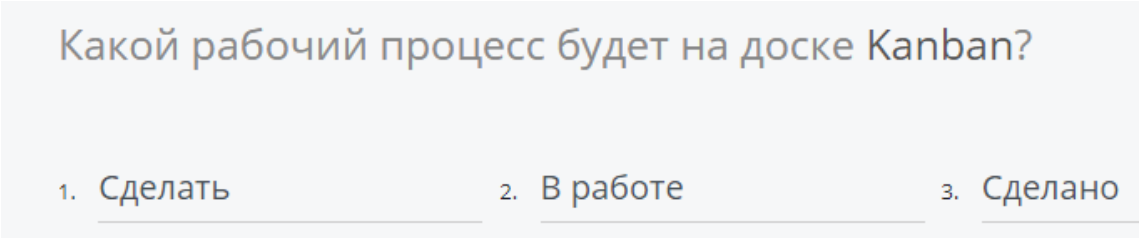

Рисунок 1.6 – Колонки-етапи в Kanban Tool

Після завершення створення дошки маємо мінімалістичний робочий стіл з мінімальною кількістю кнопок.

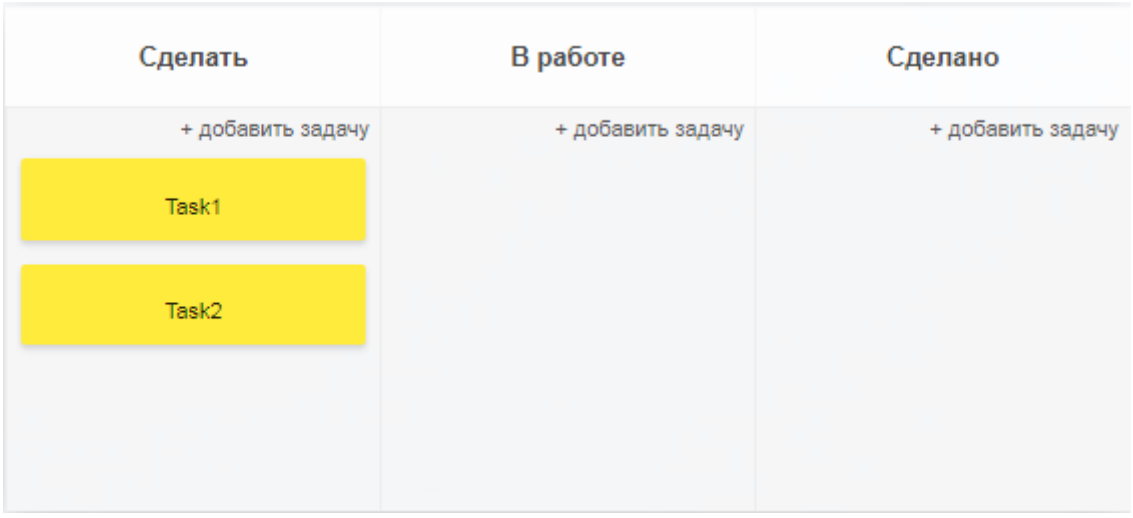

Рисунок 1.7 – Дошка Kanban Tool

На цій дошці можна створити нові картки, переміщати їх по мірі завершеності. В самій картці можна додати опис, поєднати її з іншою карткою, вказати тип картки(звичайна чи термінова), а також вказати пріоритет, дату та додати виконавця.

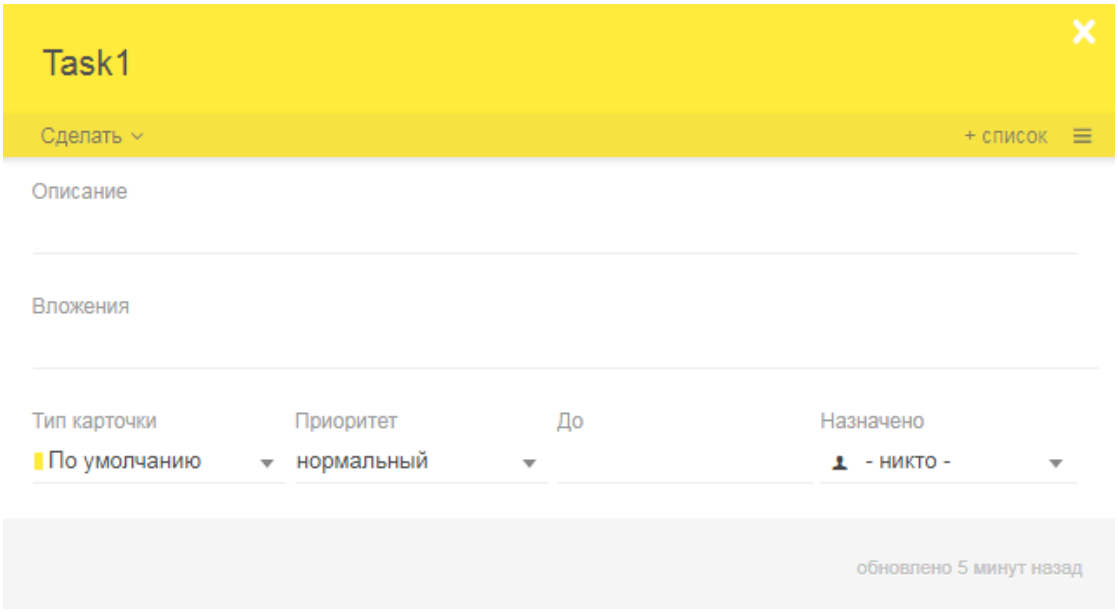

Рисунок 1.8 – Картка дошки Kanban Tool

Перевагами Kanban Tool є:

1) мінімалістичний інтерфейс, вся увага прикута до карток з задачами;

2) має підримку API та хмарних сервісів;

До недостатків можна віднести також мінімалістичний інтерфейс, оскільки для того, щоб створити новий етап треба перейти в окрему вкладку з налаштуваннями, в той час як у Trello ця функція доступна прямо на дошці.

1.4 Огляд додатку KanbanFlow

KanbanFlow – мінімалістичний додаток для роботи з канбан дошкою з вагомим набором функцій, таких як таймер, ліміт кількості задач у колонках, та аналітика. Після реєстрації користувач одразу потрапляє на шаблон дошки, яку він маже змінити під свій проект. На відміну від інших аналогів, KanbanFlow має складніший інтерфейс.

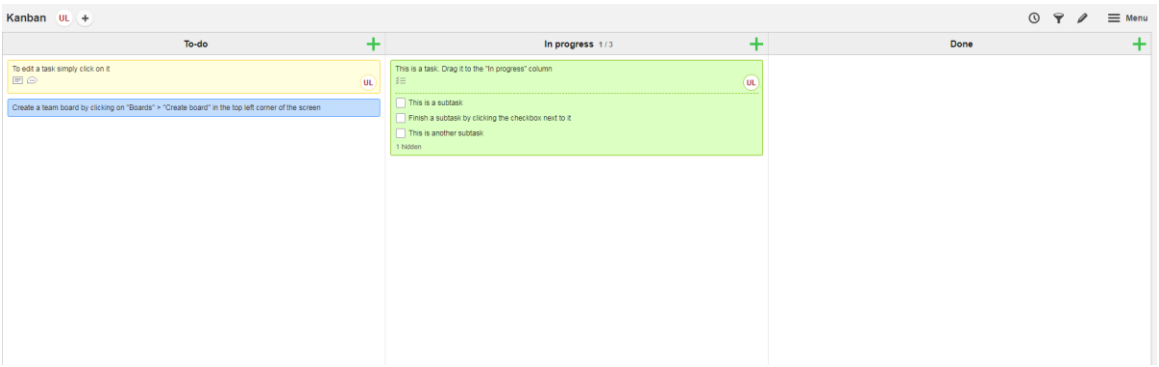

Рисунок 1.9 – Дошка KanbanFlow

По функціоналу KanbanFlow найбільший серед переглянутих додатків. Користувач має можливість запросити колегу вказавши лише адресу електронної пошти, лист створиться автоматично. Можна створити таймер для проекту або взагалі перемкнутися на методолгію Pomodoro. Користувач має можливість змінювати картки за власним бажанням: змінювати опис, додавати коментарі, та змінювати колір картки. Також можна ввімкнути таймер для певної картки, та додати виконавця.

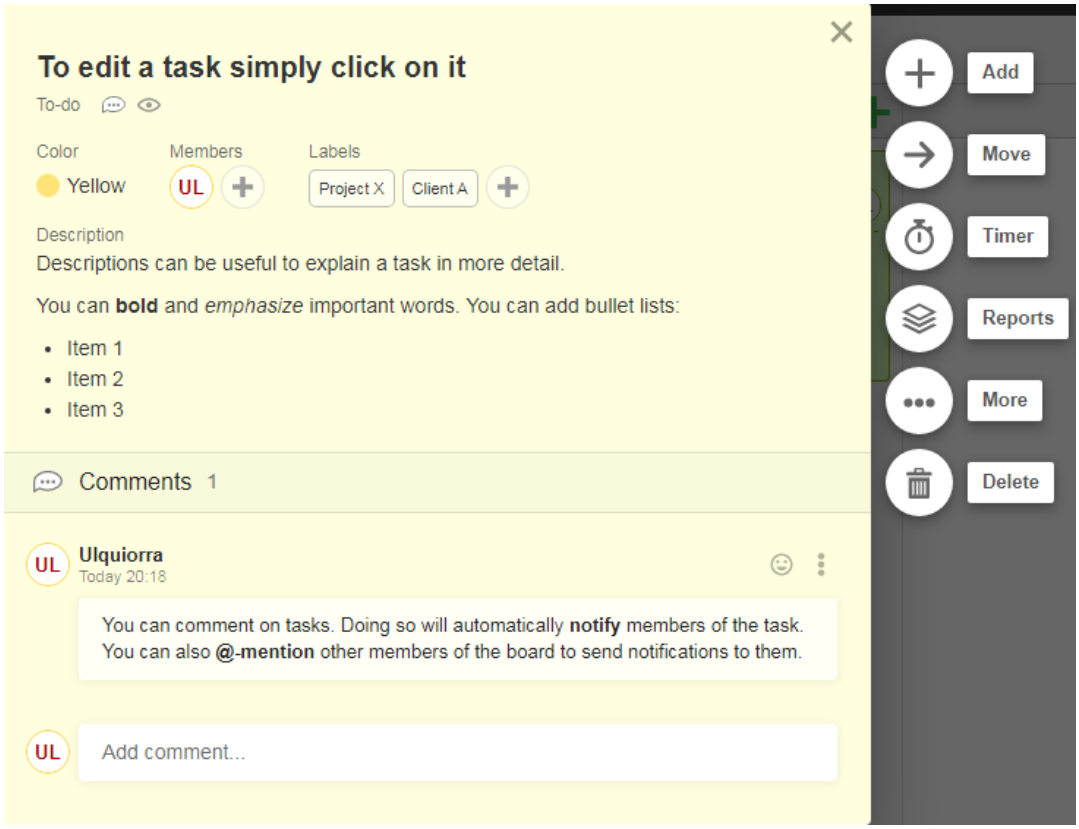

Рисунок 1.10 – Редагування картки в KanbanFlow

Переваги KanbanFlow:

1) вагомий функціонал;
2) мінімалістичний інтерфейс;
3) підтримка Pomodoro.

Недоліки:

1) складне налаштування дошки;
2) відсутність локалізації, додаток доступний лише англійською мовою.

1.5 Порівняння Scrum та Kanban

Scrum – підхід управління проектами для гнучкої розробки програмного забезпечення, який включає набір методів і попередньо визначених ролей. Scrum має декілька етапів: Product backlog, sprint backlog, burndown chart [6].

Product backlog – це список того, що має бути реалізовано. Елементи цього списку називають «історіями». Product backlog має такі атрибути: ID – унікальний ідентифікатор; назва; важливість; попередня оцінка – оцінка об’єму робіт; як продемонструвати – пояснення того, який вигляд матиме завершена задача. Також можуть бути додаткові поля такі, як категорія, компоненти, ініціатор запиту та ID у системі обліку помилок.

Sprint backlog містить функціональність проекту, обрану замовником. Функції розділені на задачі, кожна з яких має свій пріоритет. Кожного дня команда оцінює об’єм робіт необхідний для завершення задачі.

Burndown chart показує прогрес виконання проекту, скільки задач виконано і скільки залишилось.

Перехід від одного етапу до іншого називають «спринтом». Протягом кожного спринту, тривалість якого визначається командою, колеги вирішують що необхідно реалізувати далі. Функціональність яка імплементується кожного спринту приходить з Product backlog, який має найвищий пріоритет.

Історії, що визначені під час планування спринту переміщуються у етап спринту. Під час планування замовник повідомляє про завдання, які повинні бути виконані. Після цього команда визначає, скільки з бажаного вони можуть виконати, щоб завершити необхідні частини протягом спринту. Протягом спринту команда виконує визначений список завдань. В цей час ніхто не може змінювати перелік задач та порядок виконання.

Перевагами Scrum є орієнтованість на клієнта та адаптивність. Клієнт має можливість робити зміни в будь-який момент. У даній методології упор робиться на багатофункціональну команду, яка самостійно вирішує необхідні задачі, що є і недоліком, оскільки це призводить до підвищення затрат на відбір персоналу , мотивацію та навчання. За певних обставин формування повноцінної команди може бути неможливим [7].

Таблиця 1.1 – Порівняння Канбан та скрам

Методологія / **Властивість**	**Канбан**	**Скрам**
Темп	Безперервний процес [8]	Спринти фіксованої тривалості
Випуск релізу	Потік триває без перерв [1]	Вкінці кожного спринту після схвалення менеджером
Ролі	Команда під керівництвом проектного менеджера	Замовник, скрам-майстер, команда розробників
Головні показники	Час	Швидкість команди
Прийнятність змін	Зміни можуть бути в будь-який момент	У ході спринту зміни не бажані

РОЗДІЛ 2 ОГЛЯД ТЕХНОЛОГІЙ

2.1 Платформа Node.js

Node.js – програмна платформа, яка виконує JavaScript з вузькоспеціалізованої мови в мову загального користування. Node.js додає можливість JavaScript взаємодіяти з пристроями вводу-виводу через власний API(написаний на C++), підключати інші зовнішні бібліотеки, забезпечуючи виклики до них з JavaScript-коду. Node.js виконує роль веб-серверу, але і є можливість розробляти додатки на Node.js (за допомогою NW.js, AppJs) та навіть програмувати мікроконтроллери [9]. Раніше JavaScript можна було запустити тільки в браузері, але згодом розробники розширили його, і тепер його можна запускати на власному комп'ютері в якості окремого додатку. Node.js запускається у середовищі виконання V8. Цей движок використовує JavaScript код і перетворює його в більш швидкий машинний код. Машинний – низькорівневий код, який комп'ютер може запускати без необхідності інтерпритації. Головна особливість Node.js – подійно-орієнтований підхід. Такий підхід спрощує програмування додатків, особливо при реалізації ігтерфейсу вводу-виводу, оскільки сервер оброблює одразу два запити, а не чекає відаовіді від першого. По мірі розширення соціальних мереж та інших інтерактивних сайтів виросла затребуваність Node.js, як платформи, яка реагує а дії користувача.

Важливою частиною Node.js є пакетний менеджер npm – Node Package Manager. Як зазначається у вікіпедії [10]: npm (Node Package Manager) – це менеджер пакунків для мови програмування JavaScript. Для середовища виконання Node.js це менеджер за змовчуванням. Включає в себе клієнт командного рядка (npm), а також онлайн базу даних пакунків, яка називається

реєстром npm. Для Node.js випущено більше 650000 пакетів програмного забезпечення з відкритим кодом.

Для інсталяції Node.js необхідно перейти на офіційний сайт [11] та завантажити інсталятор.

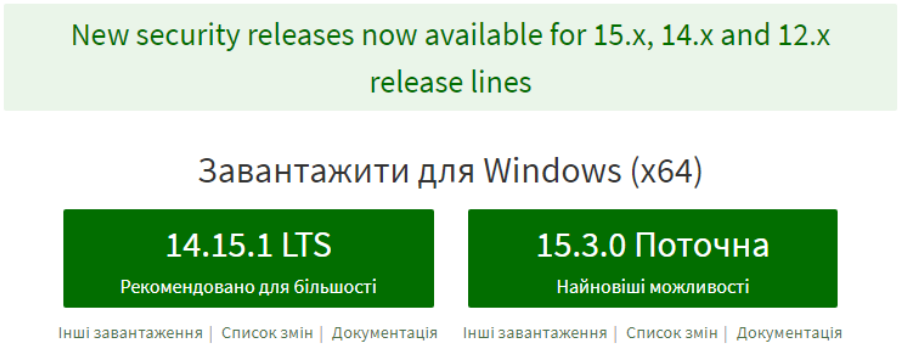

Рисунок 2.1 – Завантаження Node.js

Після чого виконується стандартна інсталяція на комп’ютер.

2.2 Огляд та інсталяція Vue.js

Vue.js – JavaScript фреймворк для створення веб-інтерфейсів. Vue.js має широку функціональність для рівня представлення і може легко інтегруватись в інші проекти. Vue використовує синтаксис шаблонів на основі HTML. Всередині Vue компілює шаблони в рендерингові функції віртуального DOM. Vue розроблений з упором на максимальну продуктивність. Він здатний обчислити кількість компонентів для ре-рендингу та виконати мінімальну кількість операцій з DOM. Компоненти допомагають розширити основні HTML-елементи. Компоненти – це повторно використовувані частини користувацького інтерфейсу. На етапі проектування розбиваємо додаток на незалежні частини і отримуємо деревовидну структуру проекту. Якщо порівнювати Vue.js з найпопулярнішими front-end фреймворками, то Angular

– фреймворк, який має чітки вимоги до структури додатку, а також є поноцінним рішенням для enterprise-додатків. React та Vue – універсальні. Їх бібліотеки можуть бути сумісними з іншими типами пакетів. Всі ці фреймворки засновані на компонентах. Vue.js найменший фреймворк та найкраще підходить для невеликих проектів, де не потребується аналіз та використання додаткових інструментів. Також React та Vue можна використовувати просто додавши бібліотеку у вихідний код, але це не можливо зробити з Angular. React та Vue пропонують більшу гнучкість для переходу від односторінкових сайтів до мікросервісів. React та Vue мають Virtual DOM(document object model), який створює копію об’єктного представлення документа, що дозволяє працювати з візуальною копією, а не самим представленням. Цей підхід підвищує продуктивність фреймворків і таким швидкість роботи додатку буде більшою. Vue має кращий розподіл пам’яті, але в основному ці фреймворки схожі по своїм характеристикам.

РОЗДІЛ 3 РЕАЛІЗАЦІЯ КАНБАН МОДУЛЯ

3.1 Створення проекту

Запустивши командний рядок від імені адміністратора, виконуємо команду npm install –g @vue/cli для інсталяції фреймворку Vue.js. Наступним кроком виконуємо команду vue create project, де project – назва створюваного проекту, та обираємо пресет за змовчуванням, чи обираємо необхідні утиліти. Запускаємо проект у текстовому редакторі. Структура проекту має наступний вигляд:

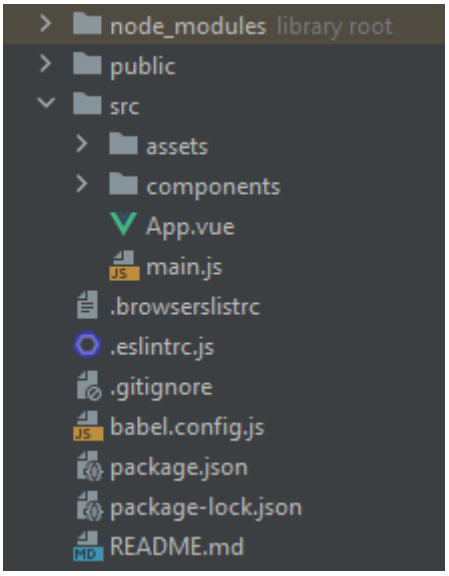

Рисунок 3.1 – Структура проекту

У папці node_modules містяться всі завантажені бібліотеки node.js. У папці public знаходиться HTML-файл, в який буде зібрано проект. В src міститься папка з компонентами, файл main.js, в якому описаний рендер додатку, та головний файл App.vue. Саме в цьому файлі імпортуються компоненти і формується вигляд додатку.

Додаток матиме вигляд дошки з різними колонками та картами з завданнями, як зображено на рисунку 1.1.

3.2 Створення компонентів

Проект міститиме наступні компоненти: TodoList міститиме список з елементів, TodoItem – компонент, в якому знаходяться самі елементи, AddItem – компонент, в якому створюються нові елементи, та головний компонент App, в який є основою додатку. В створюваному додатку буде реалізований функціонал створення нової задачі, можливість її відредагувати та видалити. Також користувач матиме можливість переміщувати задачі у потрібні колонки, які відповідають етапам розробки.

Першим кроком в папці components створимо файл TodoList з розширенням .vue(всі vue.js-файли мають розришення .vue), та імпортуємо його в App.vue. Для цього в секції script пропишемо import TodoList from "@/components/TodoList"; вказавши назву файлу та його директорію. Vue файли мають три основні секції: template, script та style. В тезі template створюються HTML-блоки. Обов’язково повинен бути кореневий тег, зазвичай це div. Тег script містить імпорти файлів та бібліотек, також в ньому створюються об’єкти та функції. В тезі style стилізують додаток за допомогою css. Також новостворені компоненти необхіно оголосити в components.

```
components:{
  AddItem,
  TodoList,
  TodoItem,
  draggable
}
```

Рисунок 3.2 – Підключені компоненти

Перейшовши в файл TodoList створюємо тег template, в якому кореневим елементом буде div з класом todo-list. Нижче розмістимо тег slot, в якому будуть розміщуватись картки. Створимо ще один компонент TodoItem.vue і імпортуємо його в App.vue та реєструємо в компонентах. Створюємо шаблон з кореневим тегом div з класом todo-item. В тезі script в export default, в якому і реєструють компоненти, реалізуємо метод props з елементом item. Тепер використовуючи інтерполяцію вказуємо об’єкт item з елементом todo: {{item.todo}}. Таким чином на сторінці буде відображатись todo-елемент об’єкта item, який і є карткою з завданням. В файлі App.vue у функції data створимо масив todos з об’єктом в якому буде властивість todo, яка міститиме назву завдання.

```
export default {
  data(){
    return{
      todos:[
        {
          id:1,
          todo: 'Learn vue draggable'
        },
```

Рисунок 3.4 – Масив з об’єктами

Для того, щоб всі події, які відбуваються в файлах компонентах відображались на сторінці необхідно імпортувати їх, що вже зроблено, та вказати назву файлу чи назву класу кореневого елемента у вигляді тегу. Так-як кореневі теги мають класи, то використаємо даний варіант. Файл App.vue має кореневий тег div з класом app. Всередині цього тегу напишемо назву класу з файлу TodoList. Оскільки об’єкти створюються в файлі TodoItem, а відображаються у файлі TodoList, то в App.vue всередині тегу <todo-list></todo-list> пишемо тег <todo-item></todo-item>. Наступним кроком у відкриваючому тезі todo-item використаємо директиву v-for для відображення елементів з масиву на основі. Всі vue директиви мають префікс v. <todo-item

v-for="todo in todos" :key="todo.id" :item="todo"></todo-item> це цикл в якому перебираються елементи todo з масива todos. Тепер об’єкт, який був створений в масиві todos відображається на сторінці.

Далі в файлі TodoList реалізуємо тег div з класом title, де використовуючи інтерполяцію виведемо title, який буде містити назви колонок, в яких будуть міститися завдання.

```
<template>
  <div class="todo-list">
    <div class="title">
      {{title}}
    </div>
    <slot></slot>
  </div>
</template>
```

Рисунок 3.5 – Текстова інтерполяція

Тепер в експорті створюємо метод props, де параметром передаємо title. Тепер в App.vue у відкриваючому тезі todo-list можна вказати параметр title з назвою стовбця: <todo-list title="Todo">Скопіюємо тег повністю, створимо декілька копій, та змінимо назву тайтлів і назву масивів у циклі. Це будуть Todo, In Progress та Completed. По аналогії з масивом todos створюємо масиви з відповідними стовбцям назвами: inProgress та completed.

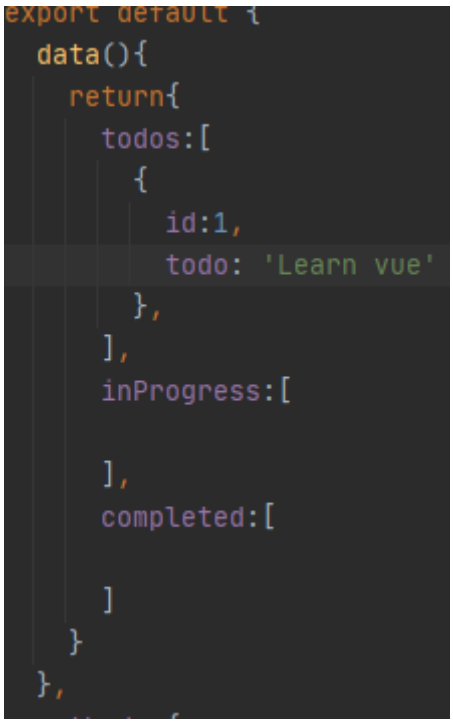

Рисунок 3.6 – Масиви з об’єктами

В головний тег додамо заголовок рівня h1. Поки що на сторінці маємо лише список який виводиться з масивів. Тому стилізуємо список в секції style. Для зручності стилізації підключимо css-фреймворк Bootstrap 4. Для цього в консолі виконаємо команду: npm install bootstrap@4.5.3 popper.js jquery –save –dev. Npm завантажить всі файли та підключить залежності в файл package.json. Залишається лише імпортувати bootstrap.css. Для цього вкінці тегу style вкажемо: @import "~bootstrap/dist/css/bootstrap.css";. Всі теги todo-list обгорнемо в div з класом desk.

Елементи класу app виставимо в рядок, вирівняємо по центру, задамо відступи та шрифт.

```
<style>
  .app{
    display: block;
    margin: 50px 0;
    text-align: center;
    font-family: "Comic Sans MS";
  }
  .app h1{
    display: flex;
    justify-content: center;
    margin-bottom: 50px;
  }
  .head{
    display: inline-block;
    margin-bottom: 30px;
  }
  .desk{
    display: flex;
    justify-content: center;
    font-family: "Comic Sans MS";
  }
```

Рисунок 3.7 – Стилізація app компонента

Стилізуємо TodoList компонент. Задаємо відступи використовуючи css-властивості padding та margin, створимо межу сірого кольору за допомогою border, розміром один піксель, вкажемо шрифт і ширину самого блоку. Для TodoItem також вкажемо відступи та межу.

```
<style>
  .todo-list{
    padding: 10px 20px;
    margin: 0 10px;
    border: 1px solid #ccc;
    font-family: "Comic Sans MS";
    width: 300px;
    text-align: center;
  }
  .title{
    padding: 10px 20px;
    font-size: 20px;
    font-weight: 600;
    margin-bottom: 10px;
    border-bottom: 1px solid #ccc;
  }
</style>
```

Рисунок 3.8 – Стилізація TodoList компонента

3.3 Реалізація перетягування завдань

Для реалізації функціоналу преміщення використовуюємо бібліотеку vuedraggable [12]. Для її інсталяції в консолі виконуємо команду: npm install vuedragable. Після цього імпортуємо бібліотеку import draggable from 'vuedraggable' та реєструємов компонентах вказавши draggable. Так-як елементи будуть переміщатись, то створимо стиль зміни курсора при наведенні на елемент, який можна змістити. Для цього в style TodoItem файлу вкажемо: .todo-item:hover{cursor: move;}. Щоб елементи можна було перетягувати необхідно кожен тег todo-item обгорнути в тег draggable. Всередині тегу вкажемо директиву v-bind з назвою list, яку скорочено можна записати як «:list», та параметром передамо назву масиву :list="todos". Для кожного todo-item відповідний параметр: для todos – todos, для In Progress – inProgress, для Completed – completed. Зараз можемо перетягувати елементи списку, але лише в межах батьківського блоку. Для того, щоб перетягувати елементи в інші блоки необхідно створити «droppable зони». Для цього кожному блоку після директиви v-bind вказуємо group="todosapp". Тепер елементи списку можна перетягувати в інші блоки та змінювати їх порядок. Зробимо перетягування трохи гарнішими. Щоб це зробити задамо кожному блоку ghostClass=”on-drag”, та стилізуємо його.

```
.on-drag{
  background-color: rgb(103, 174, 255);
  color: white;
  z-index: 10;
}
```

Рисунок 3.9 – Оформлення анімації перетягування

Тепер при переміщенні елемент буде змінювати колір тексту та фону на вказані.

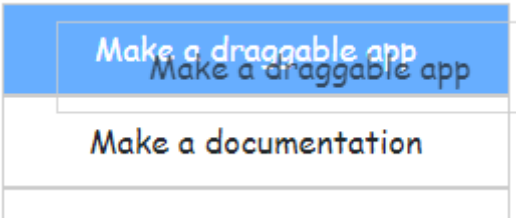

Рисунок 3.10 – Результат стилів при перетягуванні

Наступним кроком створимо форму для створення нових завданнь. Створюємо компонент AddTask.vue, імпортуємо його, та реєструємо в компонентах. Кореневим компонентом буде форма в якій створимо лейбл з текстом «Add new task», input для вводу назви завдання з параметром placholder з текстом «Type task name..», та кнопкою для пітвердження форми. Для стилізації в тезі input вкажемо клас form-control, а в button – класи btn btn-primary. В полі methods створюєо функцію onSubmit в якій перевіряємо чи не пустий input, та створюємо об’єкт newTodo, параметрами якого є: id та todo. Після цього повідомляємо серверу що є подія add-todo, яку передаємо в newTodo.

```
onSubmit(){
  if(this.todo.trim()){
    const newTodo = {
      id: Date.now(),
      todo: this.todo
    }
    this.$emit( event: 'add-todo', newTodo)
    this.todo=''
```

Рисунок 3.11 – Функція створення нового елемента

Далі в App.vue стіорюємо клас head в який додаємо <AddItem/>, в середину якого додаємо прослуховування події v-on, яку можна записати як «@», в яку передаємо щойно створену подію add-todo.

```
<div class="head">
  <AddItem @add-todo="addTodo"/>
</div>
```

Рисунок 3.12 – Прослуховування події

Тепер після того, як користувач ввів назву завдання та нажав кнопку – в списку з’явиться завдання з тією ж назвою, та унікальним ідентифікатором.

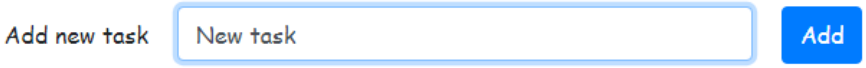

Рисунок 3.13 – Створення завдання

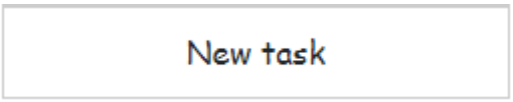

Рисунок 3.14 – Новостворене завдання

Додамо інтерактивності додавши модальне вікно для створення нової задачі. Для цього використаємо шаблон модального вікна Bootstrap, який можна знайти в документації Bootstrap 4 [13].

```
<div class="modal fade" id="exampleModal" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="exampleModalLabel">Modal title</h5>
        <button type="button" class="close" data-dismiss="modal" aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
        ...
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Save changes</button>
      </div>
    </div>
  </div>
</div>
```

Рисунок 3.15 – Код модального вікна

В тіло цього вікна перенесемо форму створення нової задачі. В стилях класу modal додамо властивість display: block для відображення модального вікна. А відображатись вікно буде при натисканні кнопки. Для цього в полі data() створимо булеву змінну visible зі значенням false. До класу modal додамо директиву v-if зі значеннм visible. Для кнопки, при натисканні якої буде відкриватись вікно, додамо подію @click зі значенням visible=true. Для кнопок, які будуть закривтаи вікно додаємо аналогічну подію, але зі значенням false.

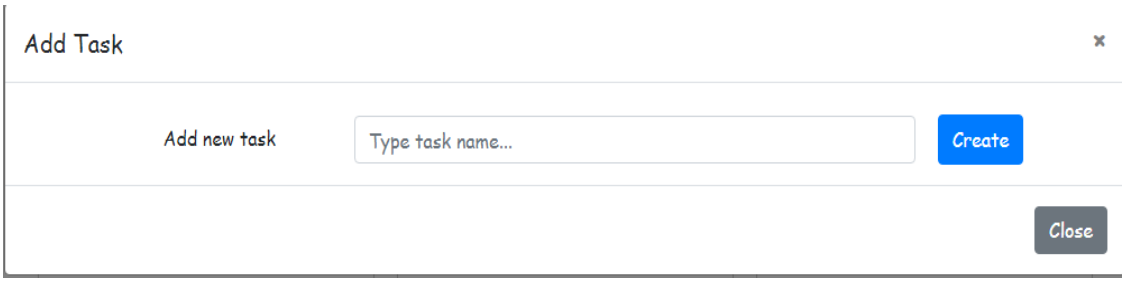

Рисунок 3.16 – Вспливаюче вікно для створення задачі

Тепер додамо інтерактивності для елементів списку. При натисканні на елемент списку буде відкриватись вікно з інформацією про завдання, можливістю відредагувати чи видалити завдання. Для цього в файлі TodoItem створимо аналогічне модальне вікно, в тіло якого стіорюємо луйбл, input та кнопку. В тег лейбл додамо {{item.todo}} для відображення назви завдання в полі лейблу. В input додаємо директиву v-model зі значенням editTodo, а в кнопку подію @click.prevent зі значенням editTask. Параметр prevent дозволяє виконувати дії без перезавантаження сторінки. Опишемо функцію editTask. Для того щоб відредагувати завдання достатньо теперішньому todo привласнити занчення моделі editTodo.

```
editTask(){
  this.item.todo = this.editTodo
```

Рисунок 3.17 – Функція редагування картки

Наступним кроком реалізуємо функціонал видалення картки з завданням. Для цього в вікні редагування картки створимо кнопку з подією @click="$emit('del', item.id)". Так параметр $emit повідомляє головному компоненту про подію del, яка буде виконуватись по параметру item.id. Тепре в файлі App.vue опишемо функцію видалення. Для цього використаємо функцію filter, в якій виконується стрілкова функція, що повертає сувору нерівність ідентифікатора.

```
delTodo(id){
  this.todos = this.todos.filter(t => t.id !== id)
```

Рисунок 3.18 – Функція видалення картки

Тепер вікно редагування картки має наступний вигляд:

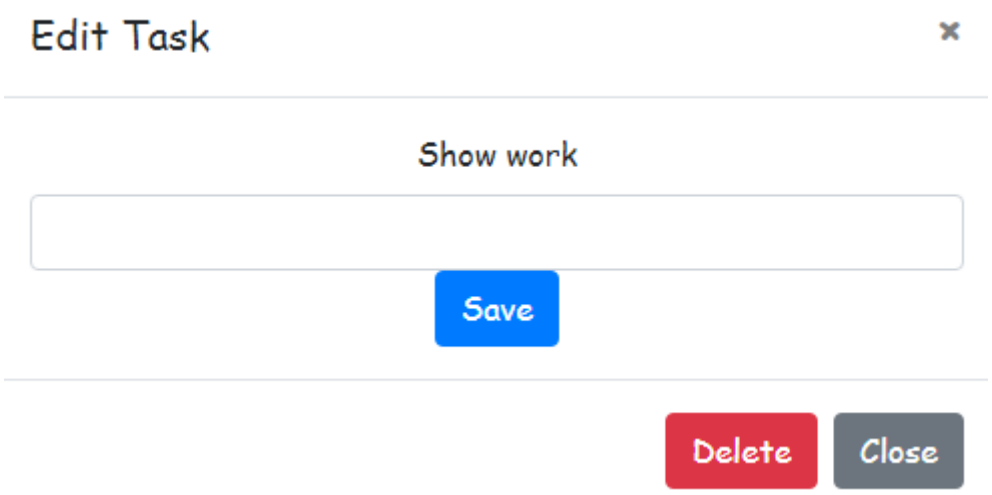

Рисунок 3.19 – Вікно редагування картки

Додамо можливість збереження стану додатку. Всі дані будуть зберігатись на стороні клієнту. Для цього використаємо local storage. Для цього в App.vue оголосимо хук mounted, в якому передаємо дані в local storage та оброблюємо помилку коли дані пошкоджені.

```
mounted(){
  if(localStorage.getItem( key: 'todos'), localStorage.getItem( key: 'inProgress')
    try{
      this.todos = JSON.parse(localStorage.getItem( key: 'todos'))
      this.inProgress = JSON.parse(localStorage.getItem( key: 'inProgress'))
      this.completed = JSON.parse(localStorage.getItem( key: 'completed'))
    }
    catch (e){
      localStorage.removeItem( key: 'todos')
      localStorage.removeItem( key: 'inProgress')
      localStorage.removeItem( key: 'completed')

    }
  }
},
```

Рисунок 3.20 – Хук mounted

Після цього в методах створимо функцію saveTodos(), яка буде парсити та зберігати дані в local storage. Створимо кнопку з подією save, в яку передаємо функцію saveTodos. SaveToodos також передаємо в функції додавання та видалення, для того щоб при кожній зміні дані зберігались. А після переміщення карток між колонками необхідно натиснути кнопку save, оскільки при перетягуванні ніякі функції не відбуваються.

```
saveTodos(){
  const parsed = JSON.stringify(this.todos)
  const pars = JSON.stringify(this.inProgress)
  const prs = JSON.stringify(this.completed)
  localStorage.setItem('todos', parsed)
  localStorage.setItem('inProgress', pars)
  localStorage.setItem('completed', prs)
}
```

Рисунок 3.21 – Функція save

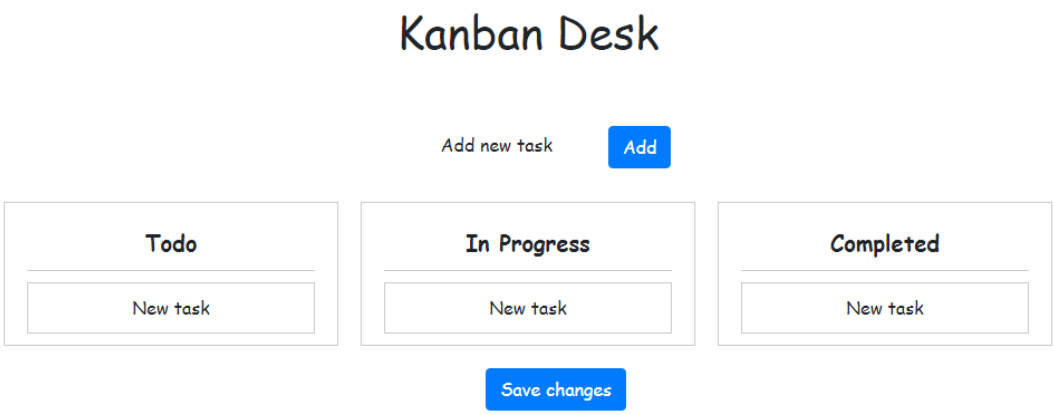

Рисунок 3.22. – Фінальний вигляд додатку

ВИСНОВКИ

Результати проведеного дослідження дозволяють зробити такі висновки:

- у першому розділі описуються визначення та основні принципи методології канбан, розглянуті додатки, створені на основі даної методології, а також порівняно канбан з скрам;
- у другому розділі розглянуто основні технології необхідні для розробки модуля канбан;
- у третьому розділі детально описаний процес розробки модуля за допомогою фреймворка Vue.js та бібліотеки vuedraggable.

На основі виконаних завдань рекомендується скласти учбовий матеріал, в якому описуються принципи та нюанси реалізації бажаною мовою програмування.

ПЕРЕЛІК ПОСИЛАНЬ

1. Канбан управление проектами – примеры применения Kanban методологии в проектах. URL: https://leadstartup.ru/db/kanban-project-management (дата звернення: 20.10.2020).

2. Kanban: основные принципы и польза. URL: https://l-a-b-a.com/blog/1529-kanban-principy-i-polza?utm_source=google&utm_medium=cpc&utm_campaign=kiseleva_projectit_531&utm_content=search&utm_term=blog&gclid=CjwKCAjwz6_8BRBkEiwA3p02VW25tCtICnPmwcOQ8maNKwOO4bEBocCt1WX8EoYMlFYS4qG0HcOONBoCTTsQAvD_BwE (дата звернення: 20.102020).

3. Канбан в IT (Kanban Development) URL: https://habr.com/ru/post/64997/ (дата звернення: 21.10.2020).

4. Toyota production system URL: https://en.wikipedia.org/wiki/Toyota_Production_System (дата звернення: 22.10.2020).

5. Организация вашей жизни с Trello URL: http://aphd.ua/orhanyzatsyia-vashei-zhyzny-s-pomoshchiu-trello/ (дата звернення: 24.10.2020).

6. Скрам URL:https://uk.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B0%D0%BC (дата звернення: 26.10.2020).

7. Гибкая методология разработки «Scrum» URL: https://habr.com/ru/post/247319/ (дата звернення: 26.10.2020).

8. Що таке канбан і чим він корисний? URL: https://worksection.com/ua/blog/kanban.html (дата звернення: 26.10.2020).

9. Node.js URL: https://ru.wikipedia.org/wiki/Node.js (дата звернення: 20.11.2020).

10. Node package manager URL: https://uk.wikipedia.org/wiki/Npm (дата звернення: 20.11.2020).

11. Node.js URL: https://nodejs.org/uk/ (дата звернення: 20.11.2020).

12. Бібліотека Vuedraggable URL: https://www.npmjs.com/package/vuedraggable?activeTab=readme (дата звернення: 20.11.2020).

13. Документація Bootstrap URL: https://getbootstrap.com/docs/4.5/components/modal/ (дата звернення: 20.11.2020).