Тут я хочу описати підхід до створення великих проектів та обгрунтувати обраний підхід на фоні з іншими існуючими

**Створення проекту**

Створення будь якого програмного продукту, в моєму випадку системна утиліта "файловий менеджер", часто вимагає спільної роботи команди фахівців. Це включає програмістів, тестувальників, дизайнерів, аналітиків і менеджерів проектів. Спільна робота в команді дозволяє ефективно розподілити завдання, використовувати різні навички та досвід кожного учасника, а також забезпечує контроль якості на кожному етапі розробки.

**Організація роботи команди**

Правильна організація роботи команди є критично важливою для успіху будь-якого проекту. Це включає планування, визначення ролей і відповідальностей, встановлення чітких цілей і термінів, а також комунікацію між учасниками команди. Добре організована команда здатна працювати більш продуктивно, ефективно вирішувати проблеми і своєчасно завершувати проект.

**Підходи та методології**

Для організації роботи команди існує багато різних методологій та підходів. Основні:

**Waterfall**

Вотерфолл — це традиційний підхід до розробки програмного забезпечення, який передбачає послідовне виконання етапів розробки. Проект розділяється на чітко визначені фази: аналіз вимог, дизайн, розробка, тестування, впровадження та підтримка. Кожна фаза повинна бути завершена перед тим, як розпочнеться наступна. Цей підхід підходить для проектів з чітко визначеними вимогами та стабільним середовищем.

**Agile**

Аджайл — це гнучкий підхід до розробки програмного забезпечення, який дозволяє адаптуватися до змін вимог та умов. Команда працює в коротких ітераціях (спринтах), кожна з яких завершується випуском працюючого продукту або його частини. Аджайл передбачає тісну співпрацю з замовником, регулярні зустрічі для обговорення прогресу та швидке реагування на зміни.

**Kanban**

Канбан — це метод управління проектами, який фокусується на візуалізації процесу роботи та оптимізації потоку завдань. Він використовує канбан-дошку з колонками, що представляють різні етапи роботи (наприклад, "Виконати", "В процесі", "Готово"). Завдання переміщуються по дошці відповідно до свого статусу. Канбан допомагає командам контролювати обсяг роботи, уникати перевантажень і забезпечувати безперервний потік виконання завдань.

**Можливості GitHub**

У моєму проекті для організації роботи використовується GitHub Issues та GitHub Projects. Ці інструменти дозволяють створювати, відстежувати та керувати завданнями, що забезпечує прозорість та ефективність у командній роботі.

**GitHub Issues**

GitHub Issues дозволяє створювати завдання, баги та запити на покращення. Кожне Issue може містити детальний опис, коментарі, мітки (labels), призначення виконавців (assignees) та терміни виконання (milestones). Це допомагає тримати всі завдання організованими та пріоритезованими.

**GitHub Projects**

GitHub Projects — це інструмент для управління проектами, який дозволяє створювати канбан-дошки для візуалізації та управління завданнями. Завдання з GitHub Issues додані до дошки, що допомагає відстежувати прогрес і забезпечує видимість статусу кожного завдання.

**User Stories**

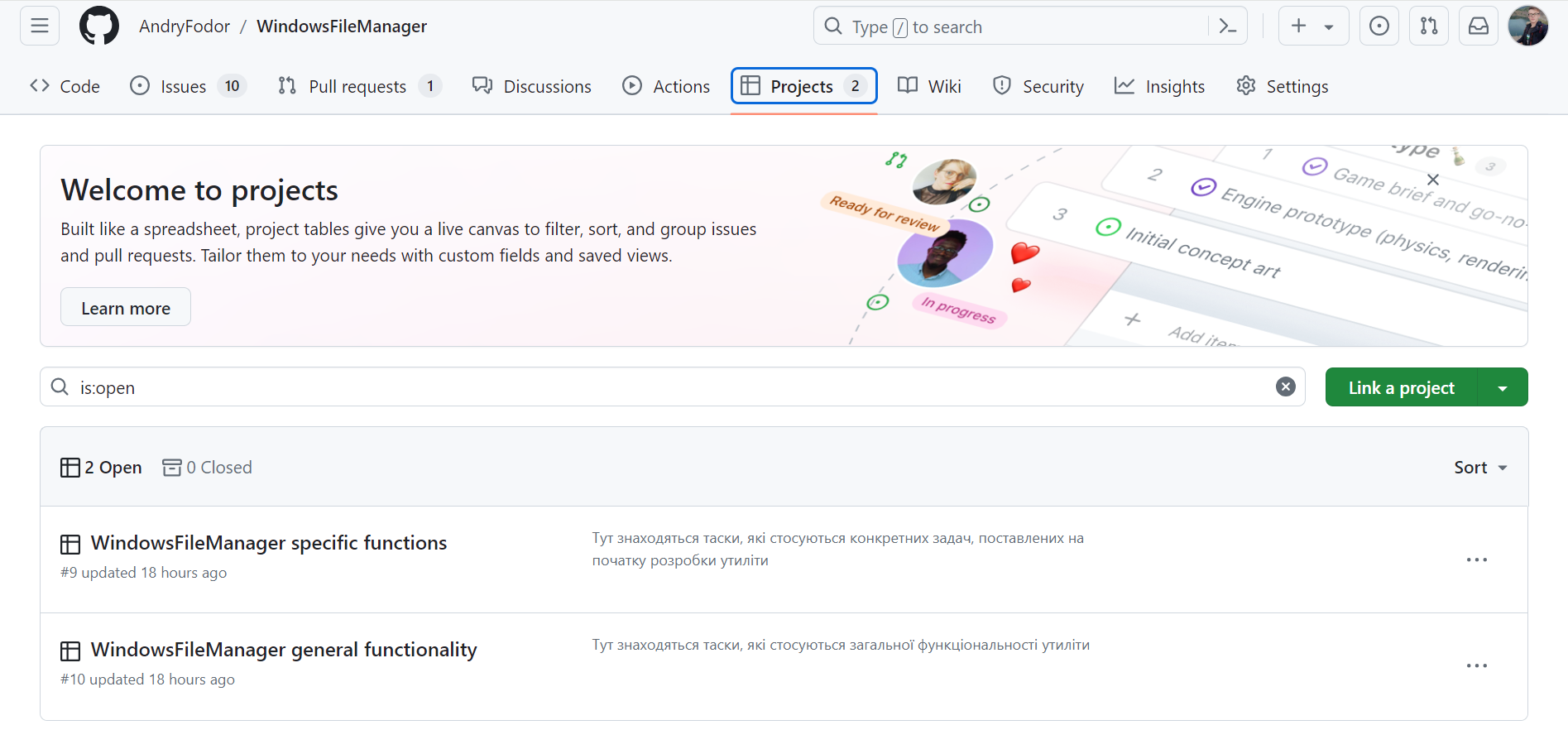
User Stories (користувацькі історії) — це інструмент, що використовується для опису функціональних вимог з точки зору кінцевого користувача. Кожна User Story зазвичай має формат "Як [користувач], я хочу [функціональність], щоб [користь]". Це допомагає команді зрозуміти потреби користувачів і фокусуватися на створенні цінності для них.

**Висновок**

Використання правильних методологій та інструментів для організації роботи команди є ключем до успішної розробки програмного продукту. Вибір підходящого методу, такого як Waterfall, Agile або Kanban, а також використання інструментів, таких як GitHub Issues та Projects, допомагає забезпечити ефективність, продуктивність і якість роботи команди.

**Реалізація GitHub Projects на моєму проекті**

У своєму проекті я створив необхідні user stories як картки issues та додав їх до двох проектів GitHub Projects.



*Рис. 1. Два GitHub projects на моєму проекті*

### Перший проект має назву [WindowsFileManager specific functions](https://github.com/users/AndryFodor/projects/9), другий -[WindowsFileManager general functionality](https://github.com/users/AndryFodor/projects/10)

### В кожному проекті зберігаються issues з конкретними user stories. Це можна побачити на рисунку 2

### 

### *Рис. 2. Проект WindowsFileManager specific functions*

### Як можна побачити, цей інструмент дозволяє організовувати роботу в стилі Kanban таблиць. Коли розробник, який має виконати відповідне завдання (воно позначене на ньому), починає над ним працювати, він перетягує відповідну картку в таблицю In Progress. Завдання знаходиться тут до того часу, поки розробник не закінчить працювати над відповідною задачею. Коли завдання виконане та відтестоване, розробник пересуває картку в Done. Таким чином проджект менеджери можуть легко відслідковувати роботу команди та керувати нею.

**Реалізація user stories як issues на моєму проекті**

### Для того, щоб переглянути всі issues, можна перейти у однойменну вкладку в репозиторії і переглянути їх. Також, якщо вони прикріплені до якихось конкретних проектів, їх можна знайти там. Виглядає вкладка Issues так, як

### зображено на рисунку 3. Якщо переглянути деталі конкрентої задачі, можна побачити заголовок, опис, лейбли, мейлстоуни, кому призначена відповідні задача, до якого проекту відпоситься, де розробляється (в якій гілці) та в якому стані виконання знаходиться, коментарі та історю її зміни.

### 

### *Рис. 3. Вкладка Issues*

### Issues 1

### [Меню](https://github.com/AndryFodor/WindowsFileManager/issues/2)

### 

### Issues 2

### [Вміст поточної дерикторії](https://github.com/AndryFodor/WindowsFileManager/issues/3)

### 

### Issues 3

### [Видаляти папки та файли](https://github.com/AndryFodor/WindowsFileManager/issues/4)

### 

### Issues 4

### [Створювати файли та папки](https://github.com/AndryFodor/WindowsFileManager/issues/5)

### 

### Issues 5

### [Отримання властивостей файлу](https://github.com/AndryFodor/WindowsFileManager/issues/6)

### 

### Issues 6

### [Відкривати файли і бачити їхній вміст](https://github.com/AndryFodor/WindowsFileManager/issues/7)

### 

### Issues 7

### [Перейменовувати файли та дерикторії](https://github.com/AndryFodor/WindowsFileManager/issues/8)

### 

### Issues 8

### [Копіювати та вставляти файли та дерикторії](https://github.com/AndryFodor/WindowsFileManager/issues/9)

### 

### Issues 9

### [Переміщати файли та дерикторії з одного місця в інше](https://github.com/AndryFodor/WindowsFileManager/issues/10)

### 

### Issues 10

### [Працювати як з абсолютними шляхами, так і з відносними](https://github.com/AndryFodor/WindowsFileManager/issues/11)

### 

### Issues 11

### [Зручні команди (круті фічі) роботи з файловою системою](https://github.com/AndryFodor/WindowsFileManager/issues/12)

### 

### Issues 12

### [Баги](https://github.com/AndryFodor/WindowsFileManager/issues/13)

### 