МІНІСТЕРСТВО ОСВІТИ І НАУКИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему:

«Інформаційна технологія автоматизованого управління проектами»

Завідувач випускаючої кафедри **Довбиш А.С.**

Керівник роботи **Шаповалов С .П.**

Студента групи ІН.м-02 **Сердюк Ю. О.**

СУМИ 2021

Сумський державний університет
(назва вузу)

Факультет *ЕЛІТ* Кафедра *Комп ютерних наук*

Спеціальність *«122 - коммп ютерні науки»*

Затверджую:
зав.кафедрою _____
_____
“_____” _____ 20___р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТОВІ

Сердюку Юрію Олександровичу
(прізвище, ім’я, по батькові)

1. Тема проекту (роботи) *«Інформаційна технологія автоматизованого управління проектами »*
затверджую наказом по інституту від “____” _____ 20 ___ р. № _____

2. Термін здачі студентом закінченого проекту (роботи) _____

3. Вхідні данні до проекту (роботи) _____
_____
_____

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Інформаційний огляд. 2) Аргументація дослідження. 3) Вибір метода рішення 4) Розробка інформаційного та програмного забезпечення системи

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

6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломного проекту (роботи)	Термін виконання проекту (роботи)	Примітка
1.	Інформаційний огляд		
2.	Аргументація дослідження		
3.	Вибір метода рішення		
4.	Розробка інформаційного та програмного забезпечення системи		
5.	Оформлення пояснювальної записки до дипломної роботи		

Студент – дипломник _____
(підпис)

Керівник проекту _____

РЕФЕРАТ

Записка: 52 стор., 33 рис., 10 джерел інформації, 1 таблиця, 2 додатки

Об’єкт дослідження – Гнучкі методології розробки та їх візуалізації.

Мета роботи – автоматизувати управління проектами.

Методи дослідження – Postman, Newman, Trello, JavaScript

.

Результати – Проаналізовано ринок конкурентів та поточну ситуацію на ньому. Було проведено конкурентне порівняння. Розроблено інформаційну технологію автоматизованого управління проектами за допомогою інструментарію Postman.

ЗМІСТ

ВСТУП

Ні для кого не секрет, що правильне управління часом є вирішальним елементом успіху. Тим не менш, занадто мало бізнес-професіоналів цінують час як один з найважливіших ресурсів, і на середньому робочому місці робиться дуже мало, щоб краще використовувати найбільш обмежені активи кожного. Незважаючи на велику увагу до прибутковості або сили людського капіталу, правильне управління своїм часом є, безумовно, найважливішим елементом успіху на сучасному ринку(Рис. Вступ 1). Мабуть, найважливіший урок, який можна вивчити, коли справа доходить до розумного управління своїм часом, полягає в тому, що не все, що є терміновим, є важливим, і навпаки [1].

Рисунок Вступ 1 – тайм менеджмент

Іноді потрібно буде негайно вирішити проблему, але, вона має незначне значення. В іншому місці буде зіткнення з рішеннями, які мають основне значення для компанії, які технічно можна відкласти на дні, тижні чи навіть місяці. Навчання правильно розподіляти свої обов’язки та майбутні проекти, щоб краще усвідомлювати, що важливо, а не терміново – це останній крок, щоб стати гуру тайм-менеджменту.

Що ж допомагає великому бізнесу впорядкувати свій тайм менеджмент, налагодити процеси у команді та оперативно стежити за всіма змінами? Це методика agile та її фреймворки.

Гнучка розробка програмного забезпечення відноситься до методологій розробки програмного забезпечення, зосереджених навколо ідеї ітеративної розробки, де вимоги та рішення розвиваються завдяки співпраці між самоорганізуючими міжфункціональними командами. Найголовніша цінність Agile-розробки полягає в тому, що вона дозволяє командам швидше надавати цінність, з більшою якістю та передбачуваністю, а також з більшою здатністю реагувати на зміни. Scrum і Kanban є двома найпоширенішими методологіями Agile [4].

Scrum — це підмножина Agile. Це легкий фреймворк процесу для гнучкої розробки і найбільш широко використовуваний. «process framework» — це певний набір практик, яких необхідно дотримуватися, щоб процес відповідав структурі. (Наприклад, фреймворк процесу Scrum вимагає використання циклів розробки, які називаються спринти, платформа XP вимагає парного програмування тощо.) «Легкий» означає, що накладні витрати на процес залишаються якомога меншими, щоб максимально збільшити кількість продуктивного часу, доступного для виконання корисної роботи [2].

Процес Scrum відрізняється від інших agile процесів специфічними концепціями та практиками, поділеними на три категорії: Ролі, Артефакти та Часові ящики. Ці та інші терміни, які використовуються в Scrum, визначені нижче. Scrum найчастіше використовується для управління складним програмним забезпеченням і розробкою продуктів, використовуючи ітераційні та інкрементні методи. Scrum значно підвищує продуктивність і скорочує час для отримання переваг порівняно з класичними «waterfall» процесами. Скрам-процеси дозволяють організаціям плавно пристосовуватися до вимог, що швидко змінюються, і виробляти продукт, який відповідає зростаючим бізнес-

цілям. Гнучкий процес Scrum приносить користь організації, допомагаючи їй у цьому:

- Підвищити якість результатів
- Краще справлятися зі змінами (і очікувати змін)
- Надавати кращі оцінки, витрачаючи менше часу на їх створення
- Більше контролюйте графік і стан проекту

Організовують роботу в організації у невеликих кросфункціональних командах, що містять усіх необхідних фахівців. Виділяють людину - скрам-майстра, який відповідатиме за дотримання процесів у команді та конструктивну атмосферу.

Scrum використовують не тільки для розробки ПЗ, він відмінно підходить для багатьох процесів створення продукту: від венчурних до маркетингових продуктів. І Scrum зовсім не методологія, це гнучкий управлінський фреймворк. Звідки слідує що – Scrum зазвичай доповнюють інженерними практиками з інших гнучких методологій (наприклад, практики розробки з екстремального програмування, або практики аналізу та збору вимог з ICONIX)

Вимоги розбивають на невеликі, орієнтовані користувачів, функціональні частини, які максимально незалежні один від одного, у результаті отримаєте беклог продукту. Потім упорядковують елементи беклога за важливістю і роблять відносну оцінку обсягів кожної історії (Рис. Вступ 2).

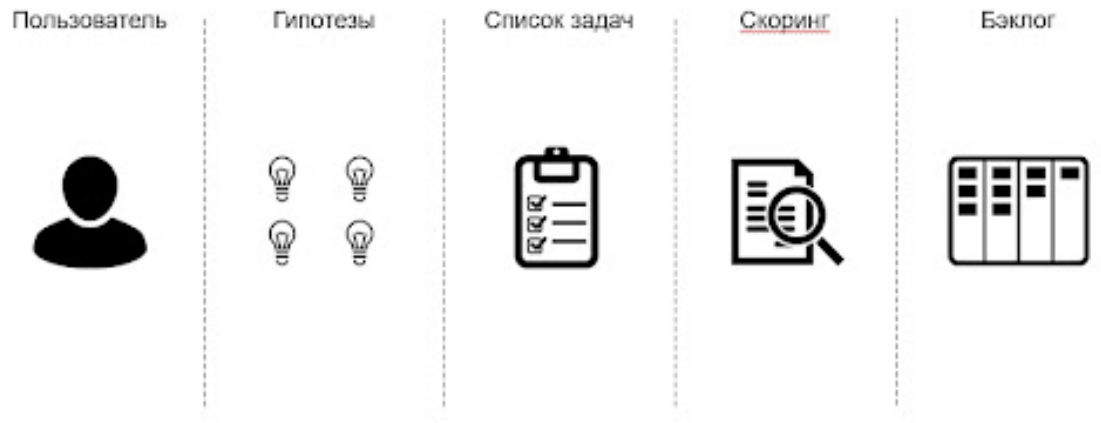

Рисунок Вступ 2 - елементи скрама

Наприкінці кожного спринту проводять огляд спринту, щоб отримати зворотний зв'язок від власника продукту, та ретроспективу спринту, щоб оптимізувати процеси. Після цього власник продукту може змінити вимоги та їх пріоритети та запустити новий спринт.

Виділяють окрему людину – власника продукту, який відповідатиме за вимоги та їх пріоритети, замикаючи на себе всіх зацікавлених осіб. Всю роботу ведуть короткими (від 1 до 4 тижнів) фіксованими ітераціями – спринтами, поставляючи наприкінці кожного їх закінчений функціонал, який можна за необхідності вивести ринку – інкремент продукту. Команда відповідно до своєї швидкості набирає завдання на незмінну за часом ітерацію – спринт. Щодня проводиться скрам-мітинг, на якому команда синхронізує свою роботу та обговорює проблеми. У процесі роботи члени команди беруть у роботу елементи беклогу згідно з пріоритетом.(Рис. Вступ 3)

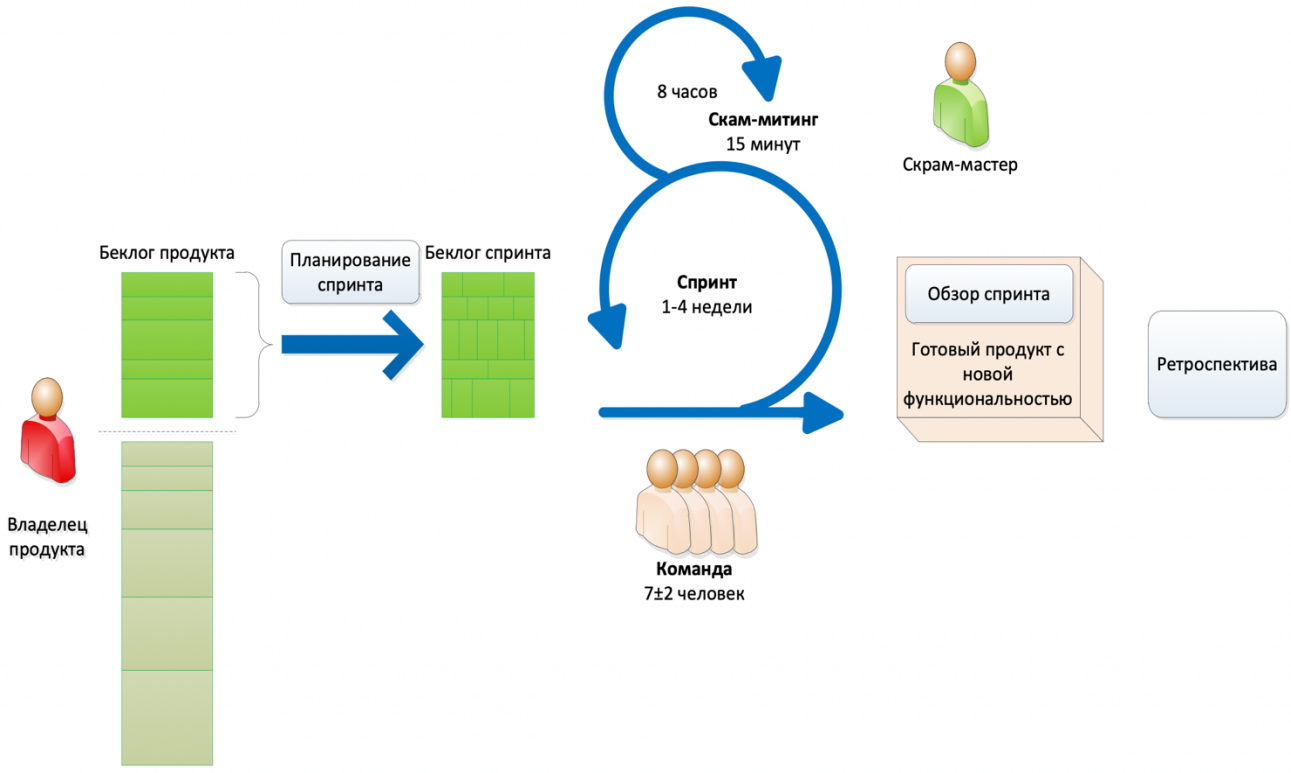

Рисунок Вступ 3 - процес роботи в скрамі

Scrum не є єдиною гнучкою методологією, хоча на даний момент він найпопулярніший серед побратимів. Але знання та розуміння інших методологій важливо, щоб розуміти їх переваги та недоліки. Крім того, на пізніх етапах адаптації Scrum можна запозичити різні практики цих методологій [2].

Також хотілося б поділитися результатами дослідження Agile Survey щодо популярності гнучких методологій (Рис Вступ 4)

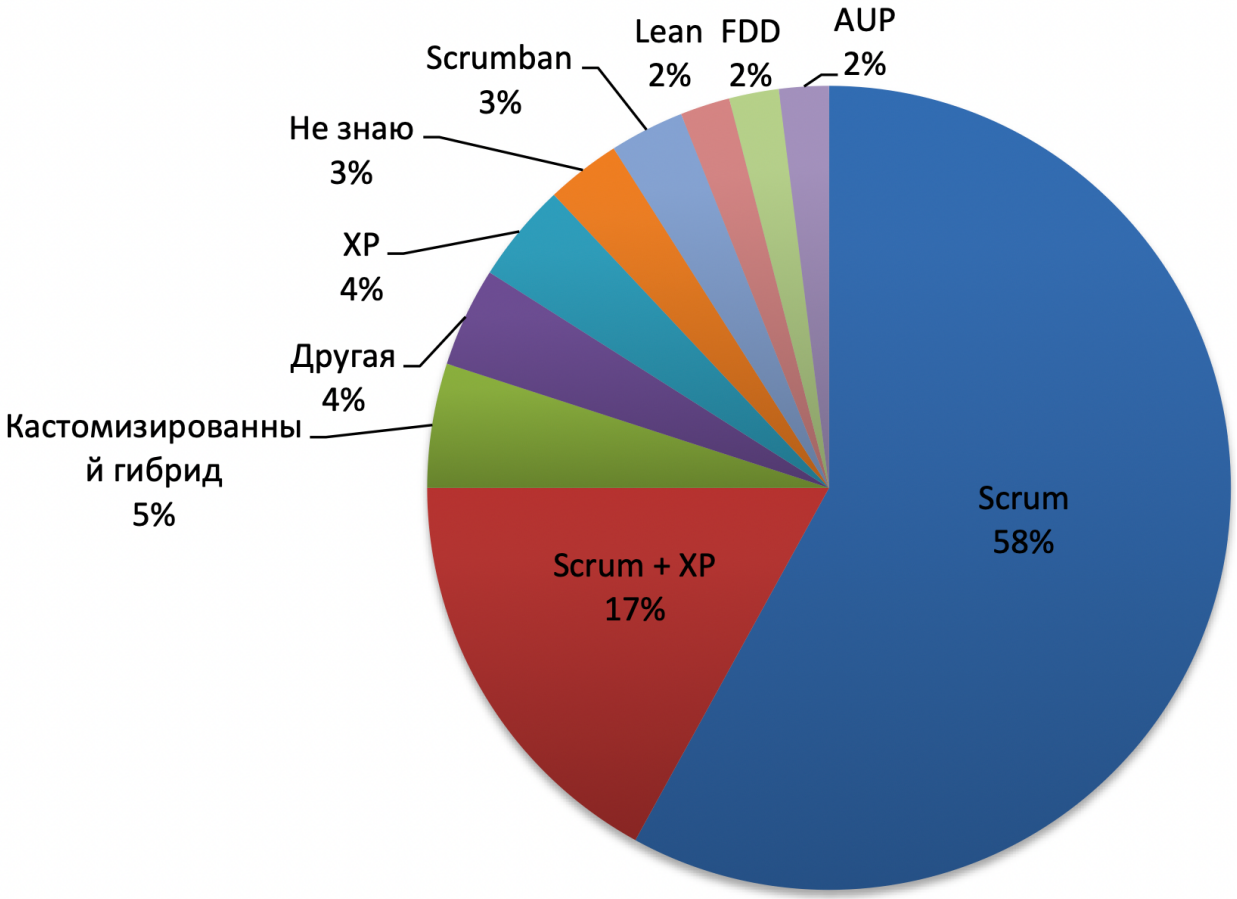

Рисунок Вступ 4 - діаграма популярності гнучких методологій

1 ІНФОРМАЦІЙНИЙ ОГЛЯД

На сьогоднішній день існує ціле сімейство продуктів, розроблених з метою спростити управління робочим процесом для різних команд. Вони створювалися як рішення для відстеження завдань та помилок. Але сьогодні це потужний інструмент управління роботою, що підходить для різних випадків, від управління вимогами і сценаріями тестування до agile-розробки програмного забезпечення.

1.1 Види дощок та їх огляд

1. ClickUp

   ClickUp — це один із найвищих у світі інструментів для підвищення продуктивності, який люблять усі види бізнесу по всьому світу. ClickUp надає безкоштовну версію з такими функціями, як призначені коментарі, списки справ, контрольні списки завдань, над багато редагування тексту, багато задачна панель інструментів, прості/користувацькі статуси, спринти, цілі тощо(Рис 1.1). Не кажучи вже про найкращу підтримку клієнтів! ClickApps дозволяє повністю налаштувати роботу групи в

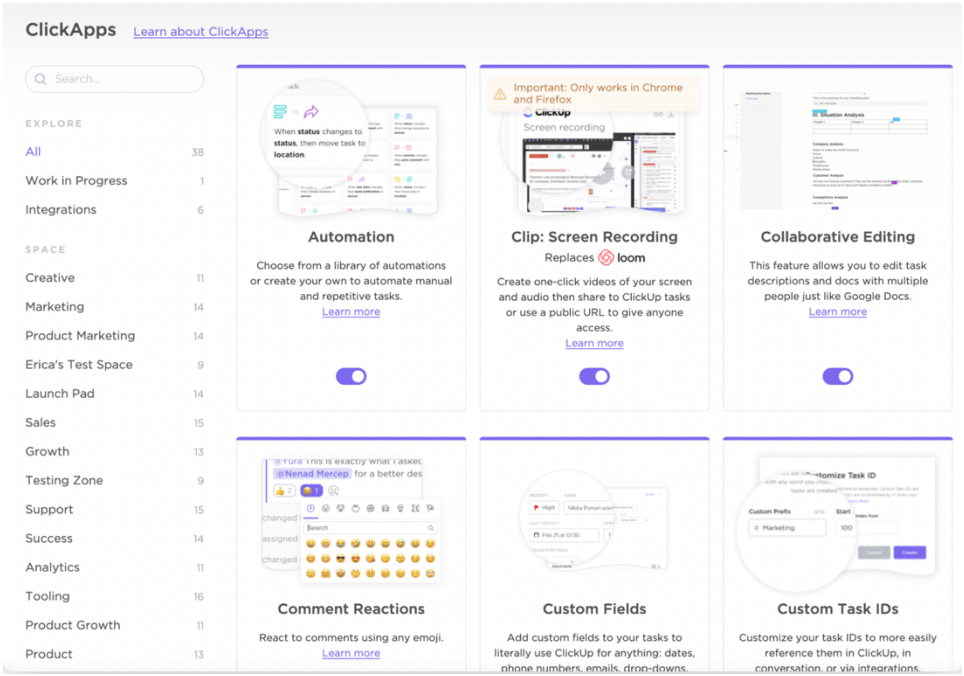

Рисунок 1.1 – дошка ClickUp

кожному окремому просторі. Використовуючи цю потужну функцію, щоб увімкнути пріоритети, теги, спеціальні поля та синхронізувати свої простори з робочим процесом. Ті, кому присвоєно звання адміністратора, мають право вмикати/вимикати ClickApps у двох розділах платформи.

2. Binfire

   Binfire — це комплексний інструмент управління проектами, який може використовувати команда інженерів та програмного забезпечення. Він надає функції керування завданнями, дошку канбан, інтерактивну діаграму Ганта тощо. Він також включає повний набір функцій спільної роботи, необхідних для керування малими та великими проектами (Рис 1.2). Binfire створює віртуальний офіс, де члени команди можуть працювати віддалено та ефективно співпрацювати один з одним. Його використовували для співпраці в проектах для великих віддалених команд на трьох континентах. Більшість інженерної групи, яка працює над проектами Binfire, знаходиться в кількох місцях. З цієї причини віддалена робота та співпраця в команді закладені в генах Binfire.

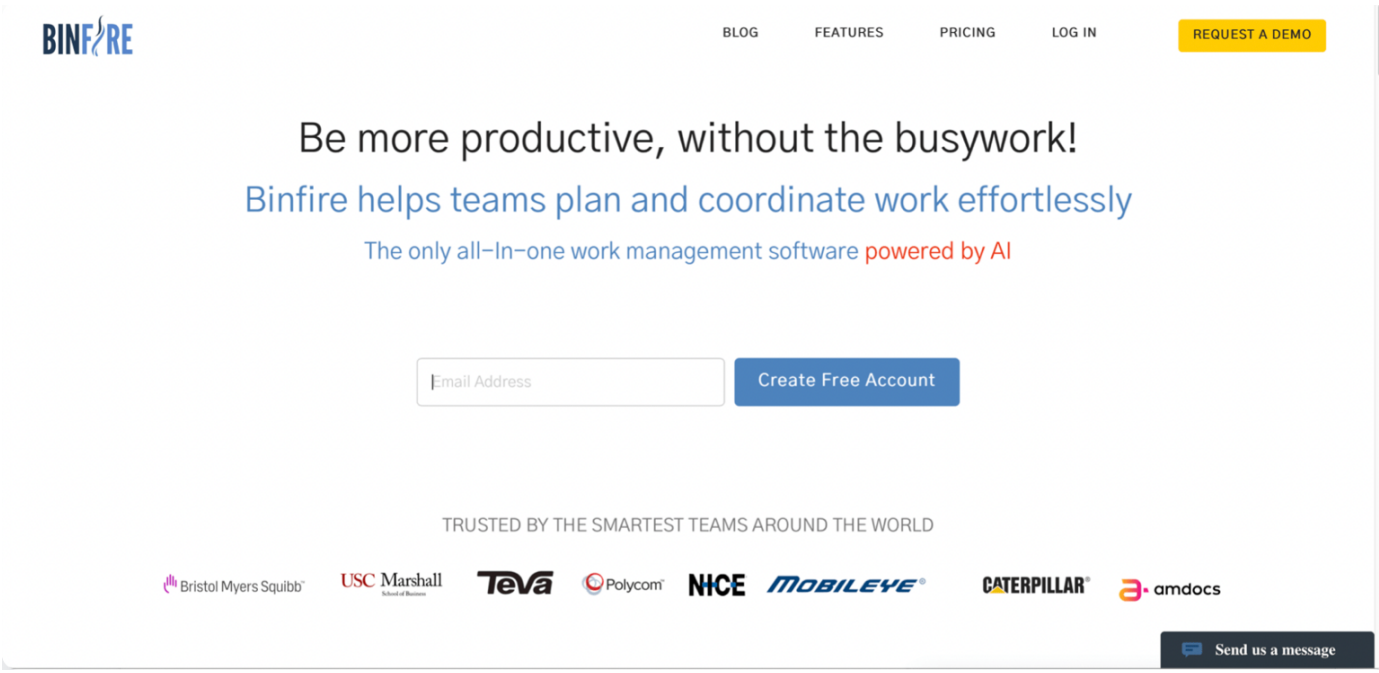

Рисунок 1.2 – дошка Binfire

3. Basecamp

   Basecamp досить ефективно спрямовує людей з різними ролями до спільної мети. Шукаючи програмне забезпечення, яке допоможе групі завершити проект разом, це програмне забезпечення саме те, що потрібно. Він забезпечує модель ціноутворення з оплатою по мірі використання без будь-якого контракту (Рис 1.3). Є річний пакет для тих, хто бажає отримати повний набір функцій системи. Вартість не залежить від кількості користувачів, тому можна залучити стільки людей, скільки потрібно.

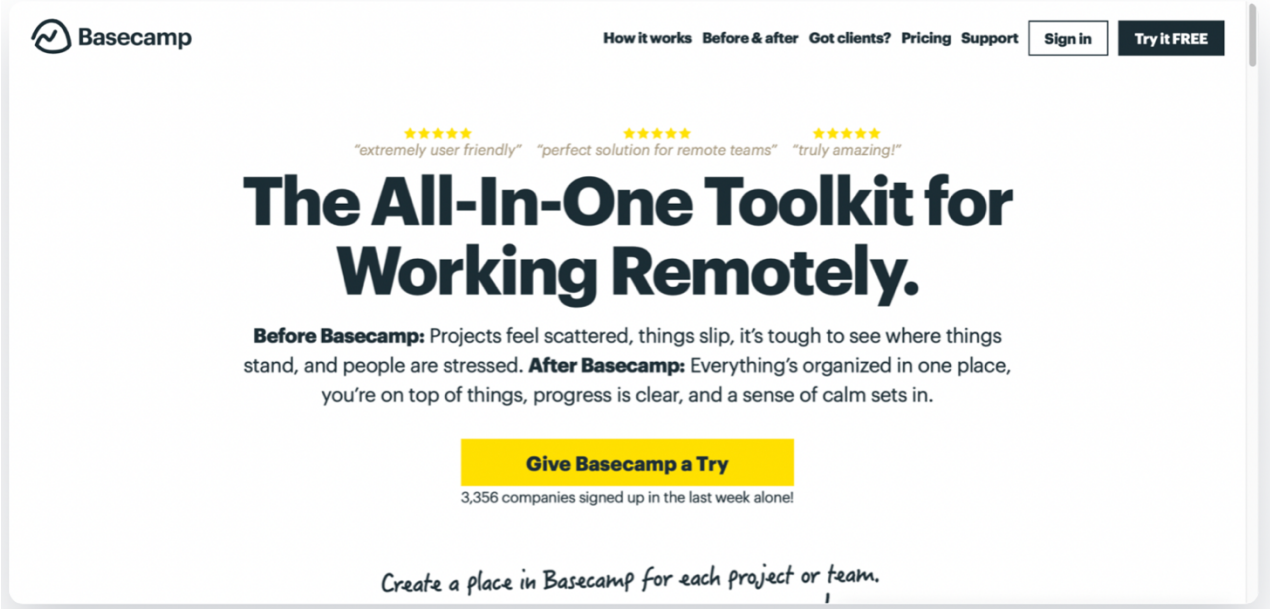

Рисунок 1.3 – дошка Basecamp

   Він дозволяє ефективно делегувати завдання та з легкістю відстежувати прогрес. Не має значення, чи знаходиться команда в різних часових поясах. Basecamp дозволяє дистанційно керувати проектами, що значно покращує швидкість роботи та продуктивність команди управління проектами.

4. Pivotal Tracker

Pivotal Tracker представляє списки відставання, функції та виправлення, щоб допомогти вашій команді вибрати, над чим працювати далі. Буде видно, як швидко працює команда розробників гнучкої програми, причому швидкість команди розраховується на основі очок історії, завершених на кожній ітерації.

Pivotal Tracker навіть допомагає планувати свої ітерації за допомогою свого інструмента відстеження(Рис 1.4), який дає змогу перервати керований обсяг роботи

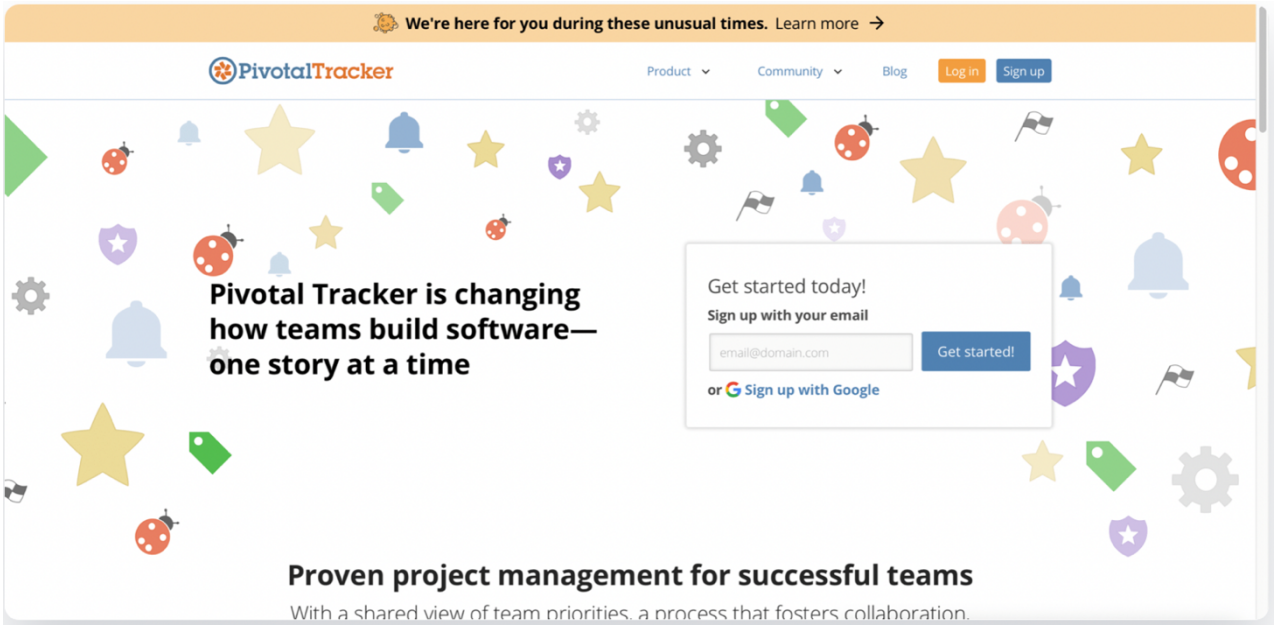

Рисунок 1.4 – дошка Pivotal Tracker

5. Asana

   Asana стала досить популярним інструментом управління проектами і не дарма. Він полегшує комунікацію та співпрацю всієї команди управління проектами. Функції Asana включають кілька робочих просторів, можливість додавати призначених і вкладення до завдань, відстеження завдань, спільну роботу в реальному часі та можливість коментувати завдання. Користувачі навіть можуть бачити завдання та пріоритети своїх колег під час завантаження. Це тип прозорості, який потрібен кожній організації та гнучкому проекту. Навіть мона відстежувати хід виконання проектів і завдань із різних пристроїв і браузерів. Немає необхідності покладатися на сторонні програми або електронну пошту для корпоративного спілкування, коли запущена Asana.

6. Clubhouse

   Clubhouse став дуже популярним через його простий, інтуїтивно зрозумілий інтерфейс. Іншими словами, користувачі отримують розумну

функціональність без захараченого та застарілого інтерфейсу користувача. Усе починається з історії, куди можна додати квитки, помилки або завдання, щоб сформувати свою історію. Він також має багато легко засвоюваних діаграм для вигоряння, оцінок тощо.

7. ProofHub

   ProofHub надає додаткову гнучкість в управлінні командами та проектами. Дає змогу повністю контролювати команди, завдання, проекти та комунікації, надаючи центральне джерело правди. Користувачі отримують гнучкість в управлінні завданнями за допомогою простих списків справ, гнучких робочих процесів і дошок канбан. Існують також діаграми Ганта, які допомагають у плануванні проекту. Вбудований додаток для чату чудово працює, щоб об’єднати всі офісні комунікації в одному місці. Він також має онлайн-інструмент перевірки, який спрощує процес перегляду та обміну відгуками. Користувачі отримують спеціальний простір для обговорень у режимі реального часу та спеціальних звітів для кращого управління проектами та ресурсами.

   ProofHub також інтегрується з додатками сторонніх розробників, такими як Google Drive, OneDrive, Dropbox і Box, щоб надати центральне місце для доступу до всіх даних. Однією з найкращих переваг цього гнучкого програмного забезпечення для управління проектами є його простий у використанні інтерфейс, який забезпечує гладку адаптацію. Немає кривої навчання, і можна звикнути до неї за найкоротші терміни.

8. Kanbanize

   Kanbanize — ще один популярний інструмент керування проектами. Це дозволяє командам візуалізувати ключові ініціативи та розбивати їх на робочі елементи, комбінуючи прості у використанні дошки канбан.

   Він надає обмеження роботи, фільтри, доступ на основі ролей і спеціальні поля, щоб ви та ваші команди могли легко візуалізувати роботу так, як завгодно. Використовуючи дошки канбан, щоб створити різні

користувацькі робочі процеси або функцію робочого процесу на шкалі часу, щоб змінити їх. Він також дозволяє відстежувати час, щоб побачити години, витрачені на виконання завдань або проектів, а також аналітику для моніторингу ефективності.

9. Notion

   Notion — інструмент керування проектами та спільною роботою дозволяє робити нотатки, створювати документи, керувати проектами, створювати завдання, використовувати метод канбан тощо, все з одного місця. На перший погляд він досить простий і легкий у використанні. Тим не менш, деякі користувачі вважали, що це трохи лякає лише через численні способи використання.

10. Wrike

    Wrike — допомагає спростити планування, отримати видимість та оптимізувати робочий процес. Це хмарне програмне забезпечення для спільної роботи, яке підходить для будь-якої команди, яка використовує waterfall, agile або будь-яку іншу модель управління проектами.
    Від діаграм Ганта до надання надійних звітів для моніторингу прогресу команди, Wrike дає змогу повністю налаштувати робочий процес і дозволити членам вашої команди бути більш продуктивними.

11. Bitrix24

    Бітрікс24 — це платформа соціального підприємства, яка поєднує управління проектами, CRM та спілкування в одному місці. Незалежно від того, невелика команда, чи ні, Бітрікс24 спрощує керування завданнями та безперебійну співпрацю. У той же час користувачі можуть відстежувати продуктивність за допомогою докладних показників керування завданнями.
    Він також досить гнучкий, тому різні відділи можуть використовувати Бітрікс24, такі як продажі та маркетинг, менеджмент, кадри, юридичні роботи, обслуговування клієнтів тощо [5].

12.Jira

Jira — це перш за все програмне забезпечення для управління проектами, але воно почало своє життя в 2002 році як платформа для відстеження проблем для розробників програмного забезпечення. Зараз він пропонується в трьох окремих пакетах:

- Jira Core: основна платформа управління проектами Jira.
- Jira Software: пропонує всі функції Jira Core, але також включає додаткову функціональність Agile.
- Jira Service Desk: призначений для IT-фахівців або інших форм служби підтримки [7].

Таблиця 1.1 Оцінки клієнтів порталу **Capterra** та **G2**

	Capterra	**G2**
ClickUp	4.8/5 (20+ reviews)	4.1/5 (10+ reviews)
Binfire	4.3/5 (9,000+ reviews)	4.1/5 (4,600+ reviews)
Basecamp	4.3/5 (110+ reviews)	4.1/5 (90+ reviews)
Pivotal Tracker	4.4/5 (9,700+ reviews)	4.3/5 (7,200+ reviews)
Asana	4.7/5 (150+ reviews)	4.4/5 (30+ reviews)
Clubhouse	4.5/5 (90,000+ reviews)	4.4/5 (11,500+ reviews)
ProofHub	4.4/5 (40+ reviews)	4.5/5 (35+ reviews)
Kanbanize	4.8/5 (90+ reviews)	4/5 (10+ reviews)
Notion	4.7/5 (300+ reviews)	4.5/5 (180+ reviews)
Wrike	4.2/5 (1,660+ reviews)	4.2/5 (1,400+ reviews)
Bitrix24	4/5 (450+ reviews)	4.1/5 (350+ reviews)

Графік аналізу даних (Рис 1.5). на основі представлених даних порталом **Capterra** .

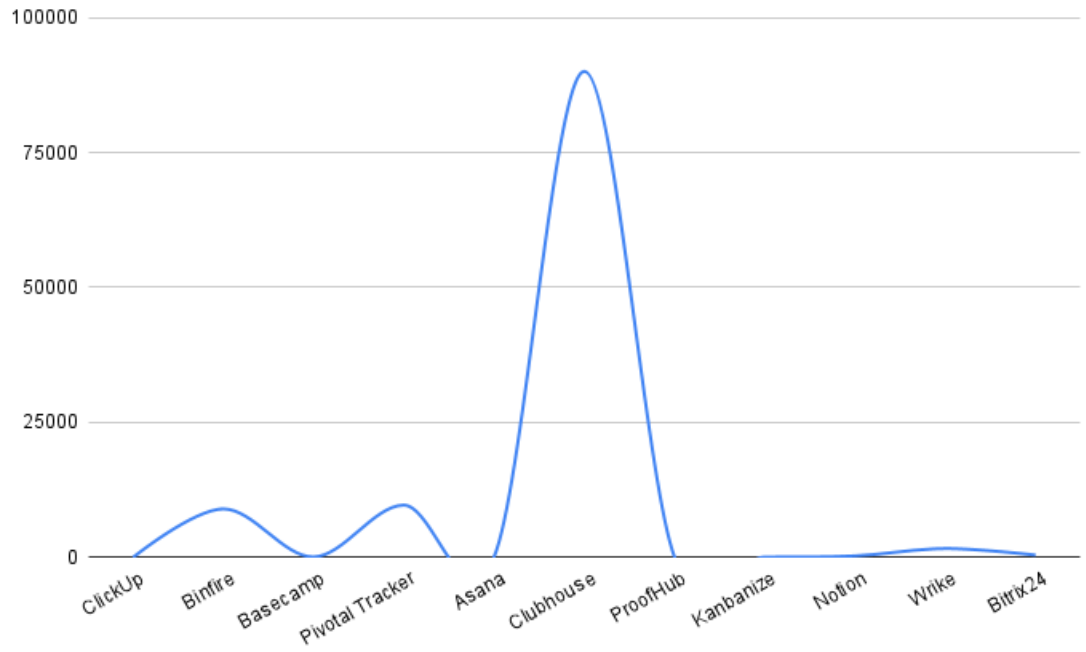

Рисунок 1.5 - графік оцінки клієнтів на основі даних порталу **Capterra**

А також графік аналізу даних (Рис 1.6). на основі представлених даних порталом **G2**

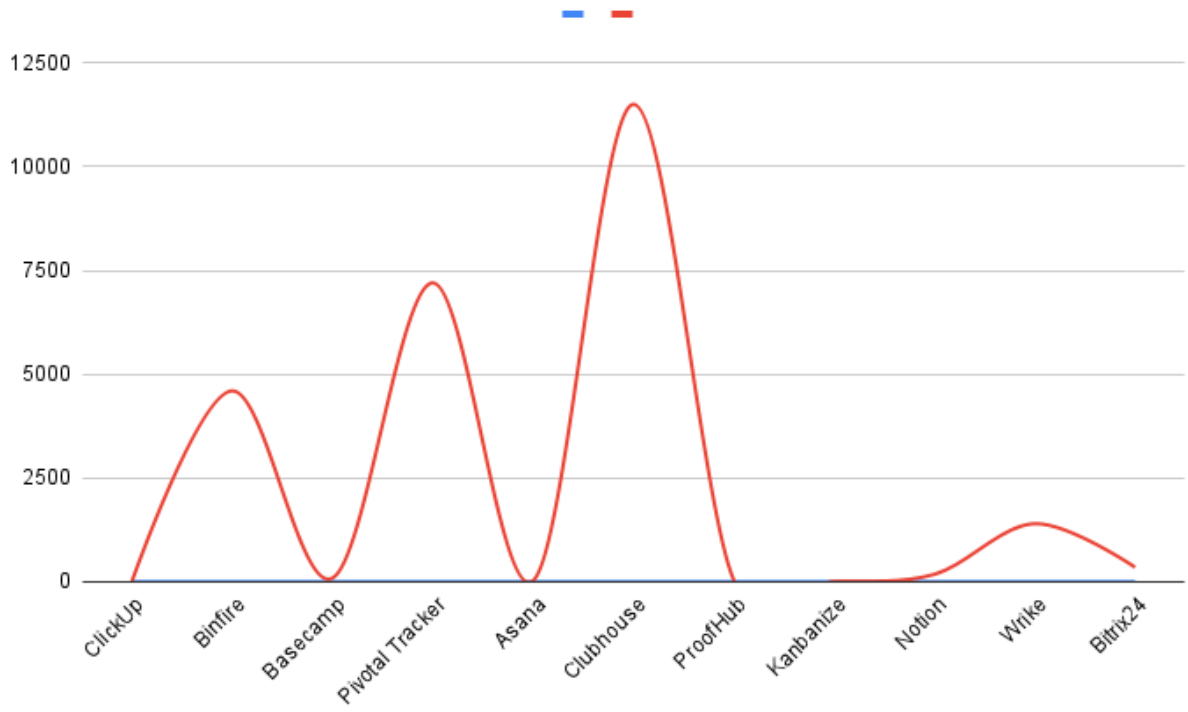

Рисунок 1.6 - графік оцінки клієнтів на основі даних порталу **G2**

1.2 Постановка задачі

Сучасні великі компанії або стартапи не можуть обійтися без систем управління проектами так як це невід'ємна частина бізнес процесів у сучасному світі. Адже це інструмент управління проектами, який можна адаптувати до процесів, щоб допомогти створювати продукти. Відстежувати проекти та завдання, використовувати Agile дошки, планувати спринти та релізи, зберігати базу знань, працювати зі звітами та панелями інструментів, створювати робочі процеси, які відповідають бізнес-процесам.

Також на рахунку кожна секунда, яка може стояти тисячі або навіть мільйони доларів, а в деяких випадках ціна вимірюється в людських життях. Задля прискорення більшості всіх процесів активно впроваджують автоматизацію у свої процеси, що дозволяє економити час та підвищує ефективність робітників.

За завдання було взято розробку автоматизації: системи управління проектами Trello з використанням інструментарію postman яка дозволяє робити наступні функції

- провести аналіз предметної області
- огляд необхідних технологій для реалізації управління проектами
- порівняти актуальні гнучкі методології розробки
- реалізація управління проектами
- написання тестів для перевірки відповіді сервера

2 АРГУМЕНТАЦІЯ ДОСЛІДЖЕННЯ

Попит на інструменти управління роботою зростав ще до пандемії Covid-19. Зростаюча популярність віддаленої роботи та гібридних робочих місць лише збільшила потребу в інструментах для координації завдань між різними командами. Насправді, Gartner очікує, що витрати на соціальні платформи та платформи для співпраці зростуть на 17% цього року до 4,5 мільярдів доларів, причому витрати на платформи керування роботою будуть найбільшим фактором. Це конкурентне поле. Поряд з Trello з'явилися різноманітні постачальники програмного забезпечення для керування роботою, які заробляють на попиті бізнесу. Серед них Asana і Monday, які завершили IPO за останні 12 місяців за оцінкою в 4 мільярди доларів і 7,5 мільярдів доларів відповідно. Останніми місяцями були придбані інші конкуренти в області, зокрема Workfront (куплений Adobe за 1,5 мільярда доларів) і Wrike (куплений Citrix за 2,25 мільярда доларів). Microsoft також має інструмент керування завданнями Planner, який є частиною її пакету Microsoft 365 [6].

Багато з цих інструментів побудовані на основі того ж підходу до списку справ у стилі Kanban, який популяризував Trello; вони пропонують подібні дошки, списки та макети карток, а також інші «погляди». Більшість із них також включають автоматизацію та функції спільної роботи. Інші конкуренти мають більш орієнтований на електронні таблиці погляди на управління проектами; до них належать добре фінансовані автономні програми, такі як Smartsheet і Airtable, а також такі продукти, як Microsoft Lists і Google Tables.

Trello вже давно займає значну частку ринку, в основному завдяки простоті використання. Згідно з останньою загальнодоступною статистикою, у 2018 році було щонайменше 50 мільйонів зареєстрованих користувачів, хоча Atlassian каже, що зараз ця цифра значно вища. Він також отримав вигоду від інтеграції в портфоліо програм Atlassian, таких як Jira.

Він поєднує простоту використання та гнучкість з потужними можливостями, що робить його ефективним інструментом у широкому

діапазоні випадків використання. Крім того, імпульс, набраний під час карантину COVID-19, показує, що Trello «природно підходить» для підтримки командної співпраці в розподіленій робочій силі, сказав він. Загалом Trello виділяється на тлі інших подібних інструментів; однак він стикається з зростаючою конкуренцією з боку суміжних категорій. Це, мабуть, буде основним викликом для подальшого просування.

Якщо потрібен більш простий вигляд дошки, але подобаються картки та оновлення статусу, то, можливо, варто спробувати Trello. Користувачі не отримають складних речей, як-от окулярів історії або відстеження проблем, але можна налаштувати різні дошки для відставання, свого спринту або наступної ітерації. Перехід на Trello може бути корисним для меншої команди розробників програмного забезпечення, яка хоче керувати кількома проектами за справедливою ціною. Що приємно в Trello, так це їхні бонуси, які дозволяють додавати діаграми вигорання або діаграму Ганта. Є можливість додати потрібні функції, не сплачуючи за них на початку. Все це в інтуїтивно у зрозумілому інтерфейсі (Рис 2.1).

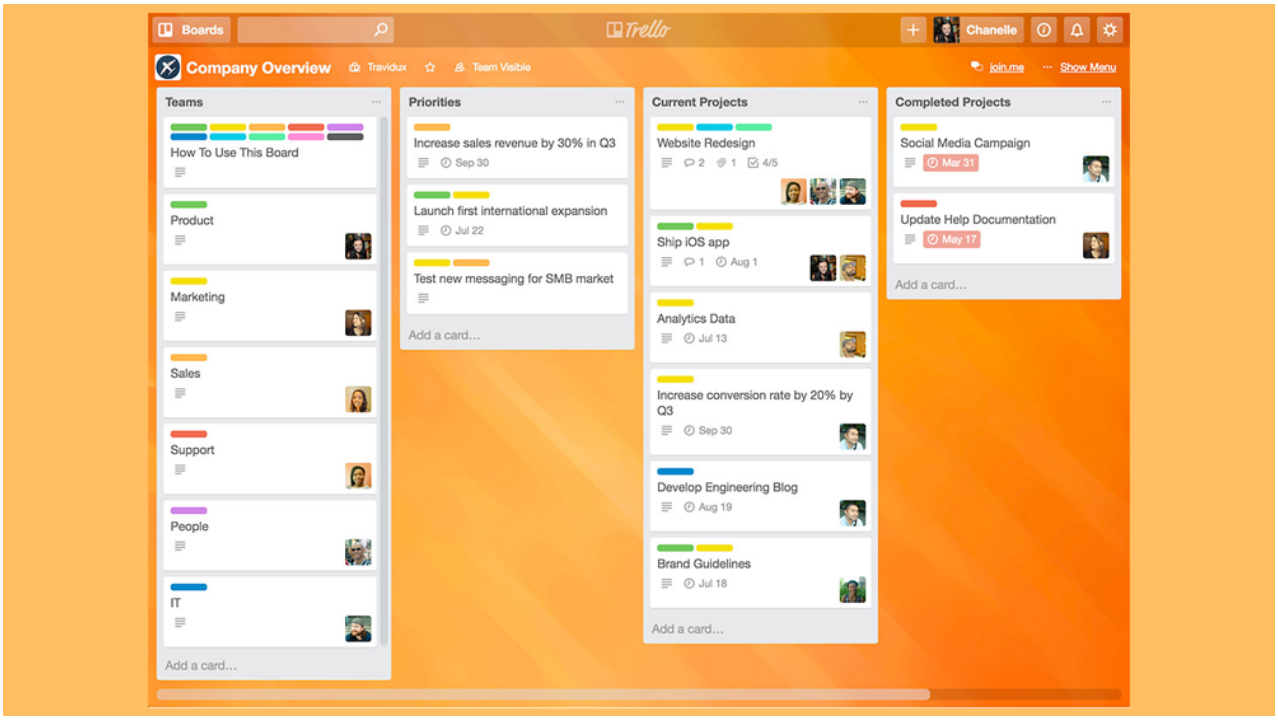

Рисунок 2.1 – приклад дошки Trello

Основна проблема з Trello? Він блокує у цьому поданні дошки, не враховуючи, як списки або перегляд часу можуть бути кращими для деяких користувачів. Крім того, картки можуть бути дуже переповненими, якщо є коментарі, що робить щось на кшталт ClickUp кращим вибором для цього.

Головні переваги перед конкурентами:

- Теги, мітки та категорії
- Перетягування карток
- Діаграми прогресу
- Встановлення повідомлень
- Вид дошки Kanban
- Призначення завдань
- Багато інтеграцій

Як Trello планує розвиватися? Раніше цього року Trello оголосила про капітальний ремонт свого додатка, щоб збігтися з 10-річним ювілеєм його запуску. Серед інших змін Trello отримав кілька нових представлень дошки: таблицю, шкалу часу, інформаційну панель і календар (Рис 2.2). Вони забезпечують різні точки зору, дозволяючи користувачам переглядати та взаємодіяти з інформацією, що міститься на дошках.

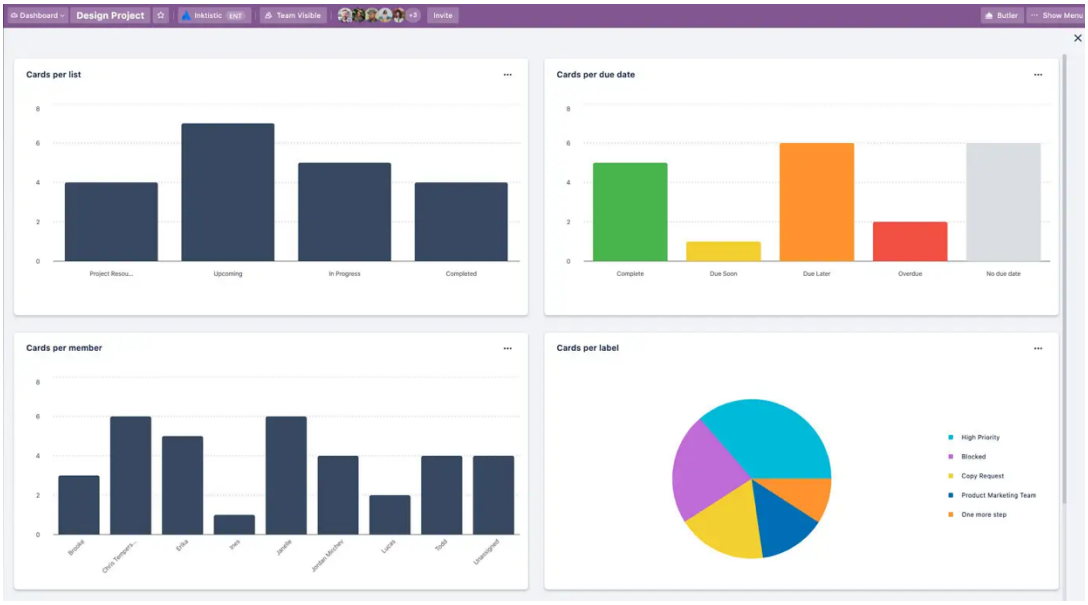

Рисунок 2.2 – нововведення Trello

Користувачі зможуть переміщати картки між різними дошками в кожному з різних представлень, так само, як це можливо з переглядом таблиці. Сторонні розробники також зможуть створювати власні програми та сервіси для зв’язку з різними видами дошки в майбутньому. Trello тепер пропонує три нові типи карток: картки посилань, картки дошки та дзеркальні картки. Картки посилань, наприклад, можуть підключатися до таких програм, як Dropbox або Google Drive, вставляючи URL-адресу в заголовок картки, і можуть відображати попередній перегляд вмісту.

Тим не менш, зараз додаток стикається з сильнішою конкуренцією, ніж це було в минулому, особливо через те, що деякі елементи програм керування роботою вбудовуються в існуючі інструменти підвищення продуктивності. Наприклад, Fluid Components є частиною Microsoft 365, а Smart Canvas включено в Google Workspace. Ці інтеграції роблять легкі функції керування завданнями доступними в ряді типів документів. Оновлення від Microsoft і Google також показують, що Trello стикається з жорсткою конкуренцією, а дорожня карта інструментів для підвищення продуктивності все більше включає можливості керування роботою.

3 ВИБІР МЕТОДУ РІШЕННЯ

3.1 Postman

У роботі був використана платформа для створення та використання API – Postman Version 8.12.1 (8.12.1) (Рис 3.1), яка спрощує кожен крок життєвого циклу API. Вона дозволяє легко зберігати, каталогізувати та співпрацювати з усіма своїми артефактами API на одній центральній платформі. Postman може зберігати та керувати специфікаціями API, документацією, рецептами робочого процесу, тестовими прикладами та результатами, метриками та всім іншим, що стосується API. Платформа включає в себе повний набір інструментів, які допомагають прискорити життєвий цикл API — від проектування, тестування, документації та мокування до спільного доступу та доступності API [8].

Рисунок 3.1 – можливості Postman

1. API client

Клієнт Postman API є основним інструментом, і дає змогу легко досліджувати, налагоджувати та тестувати API, а також дає змогу визначати складні запити API для HTTP, REST, SOAP, GraphQL та WebSockets. Клієнт API автоматично визначає мову відповіді, посилань і форматування тексту всередині тіла, щоб спростити перевірку. Також включає вбудовану підтримку протоколів аутентифікації, таких як OAuth 1.2/2.0, AWS Signature, Hawk та багато інших. Можна організувати запити в колекції Postman, щоб допомогти організувати запити для повторного використання, щоб не витрачати час на створення всього з нуля. Колекції також можуть містити код JavaScript для зв'язування запитів або автоматизації загальних робочих процесів, і можна використовувати сценарії для візуалізації відповідей API у вигляді діаграм і графіків.

2. Дизайн API

Є можливість розробити свої специфікації використовуючи формати OpenAPI, RAML, GraphQL або SOAP (Рис 3.2). Редактор схем полегшує роботу з файлами специфікацій будь-якого розміру, а також перевіряє специфікації за допомогою вбудованого механізму linting. Також є можливість створювати

Рисунок 3.2 -дизайн Postman

колекції Postman для кількох етапів життєвого циклу API — для макетів, документації, тестів, моніторів тощо — усе з файлу специфікації, все синхронізовано.

3. Тестування API

Створювати та запускати тести безпосередньо в Postman або як частину конвеєра CI/CD через Newman (Колекційний засіб, який дозволяє запускати та тестувати колекцію Postman безпосередньо з командного рядка). Postman можна використовувати для написання функціональних тестів, інтеграційних тестів, регресійних тестів тощо. Середовище виконання на основі Node.js Postman містить підтримку поширених шаблонів і бібліотек, які можна використовувати для швидкого створення тестів.

3.2 Newman

Засіб для запуску колекції командного рядка для Postman (Рис 3.3). Дозволяє запускати та тестувати колекцію Postman безпосередньо з командного рядка. Створений з урахуванням можливості розширення, щоб була можливість легко інтегрувати його зі серверами безперервної інтеграції та системами збірки.

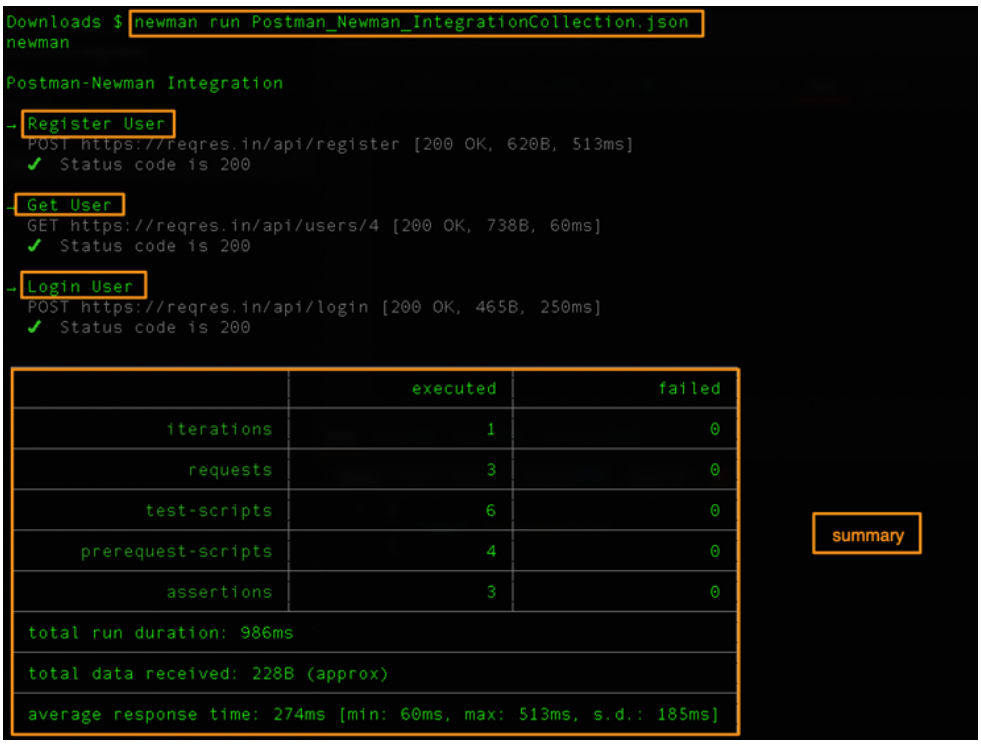

Рисунок 3.3 – приклад роботи Newman

Підтримує паритет функцій із Postman і дозволяє запускати колекції так, як вони виконуються всередині завантажувача колекцій у Postman [10].

Команда newman run дозволяє вказати колекцію, яку потрібно запустити. Потрібно експортувати свою колекцію Postman як файл json із програми Postman і запустити її за допомогою Newman. Якщо файл колекції доступний у вигляді URL-адреси (наприклад, з служби Cloud API), Newman може отримати ваш файл і також запустити його. Newman можна легко використовувати у проектах JavaScript як модуль Node.js. Весь набір функціональних можливостей Newman CLI також доступний для програмного використання.

Використання Reporters з Newman, які надають інформацію про поточний запуск колекції. Репортери можна налаштувати за допомогою параметрів -r або --reporters. Вбудовані в newman: cli, json, junit, progress і emojitrain.

CLI Reporter увімкнено за замовчуванням, коли Newman використовується як CLI, не потрібно спеціально вказувати те саме як частину параметра. Однак увімкнення одного або кількох інших репортерів не призведе до виведення CLI.

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ

4.1 Як все буде працювати

На підставі написаних можливих сценаріях, методи взаємодії з Trello будуть поміщені в окрему колекцію. Також всі змінні будуть поміщені в окрему колекцію environments для підвищення безпеки, масштабованості проекту та управління ним у майбутньому. Після цього запускатимуться з командного рядка Newman або при необхідності з будь-якої CI/CD платформи (Рис 4.1). У колекцію environments вшито секретний ключ trellokey та token, щоб сервер Trello розумів, що наша програма пов'язана з певним акаунтом на якому будуть створюватися, видалятися та оновлюватися всі необхідні сутності. Через термінал буде надсилатись http запит на сервер Trello, який по ключу визначати куди його направляти і надсилатиме у відповідь response, який написані тести будуть аналізувати.

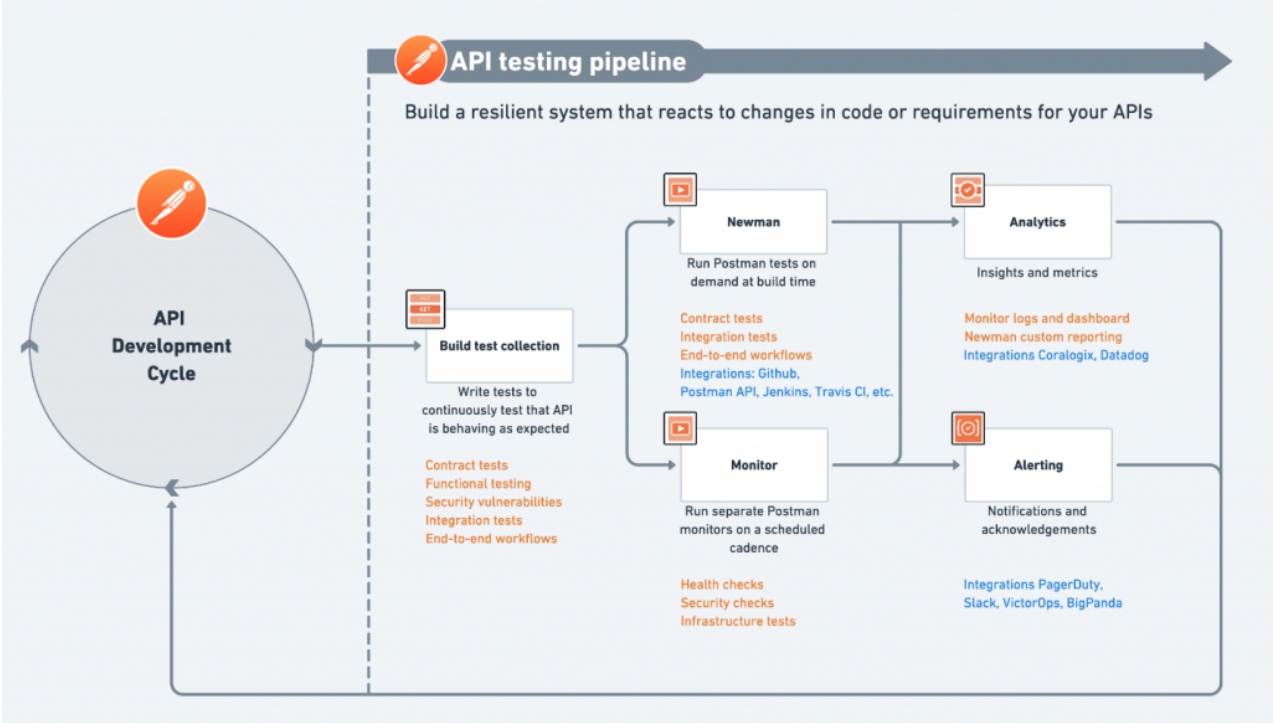

Рисунок 4.1 - схема роботи

Trello використовує делеговану автентифікацію та авторизацію, тому програмі ніколи не доведеться мати справу зі збереженням або обробкою імен користувачів або паролів. Натомість програма передає керування Trello

(ідентифікуючи себе за допомогою ключа API), і як тільки Trello дозволить користувачеві вибрати обліковий запис і ввійти, Trello передасть користувача та контроль назад програмі разом із маркером API. Щоб почати, знадобиться ключ API який можна отримати увійшовши в Trello та відвідавши https://trello.com/app-key. Ключ API буде чітко позначений у верхній частині цієї сторінки (Рис 4.2).

Рисунок 4.2 - API ключ Trello

Ключ API має бути рядком із 32 символів, що складається з випадкових буквено-цифрових символів. Через те, як працює потік авторизації, ключ API має бути загальнодоступним. Ключ API сам по собі не надає доступу до даних Trello користувача. Однак, оскільки токени API надають доступ до даних користувача, їх слід тримати в секреті. На тій самій сторінці є гіперпосилання «Token» під ключем API, після його натискання з'явиться такий екран та можливість отримати token (Рис 4.3)

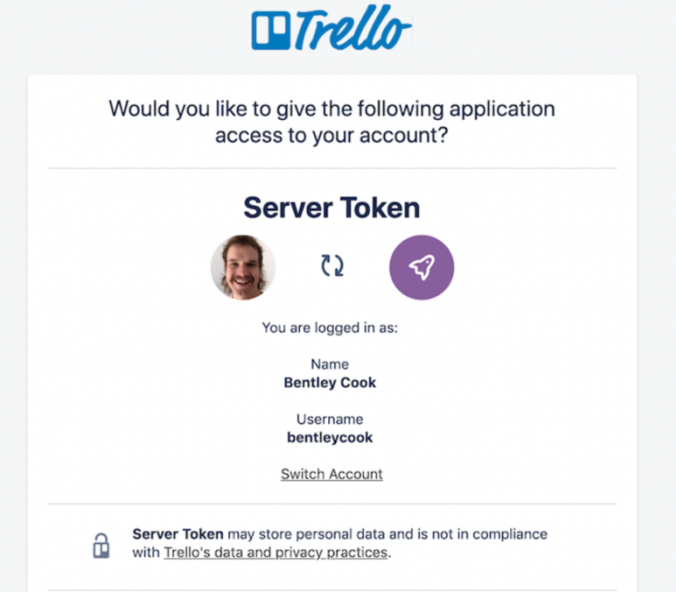

Рисунок 4.3 – Token Trello

Цей token разом із trello key API використовувати для читання та запису для всього облікового запису Trello.

Наступним кроком скористаємося офіційною документацією Trello Rest API [9] і використовуючи навігацію знайдемо необхідні методи, які потрібні для реалізації, а саме:

- Create a Board
- Create a new List
- Create a new Card
- Update a Card
- Delete a Board
- Get a Board

І відразу можна побачити детальний опис методів (Рис 4.4) та параметрів до них.

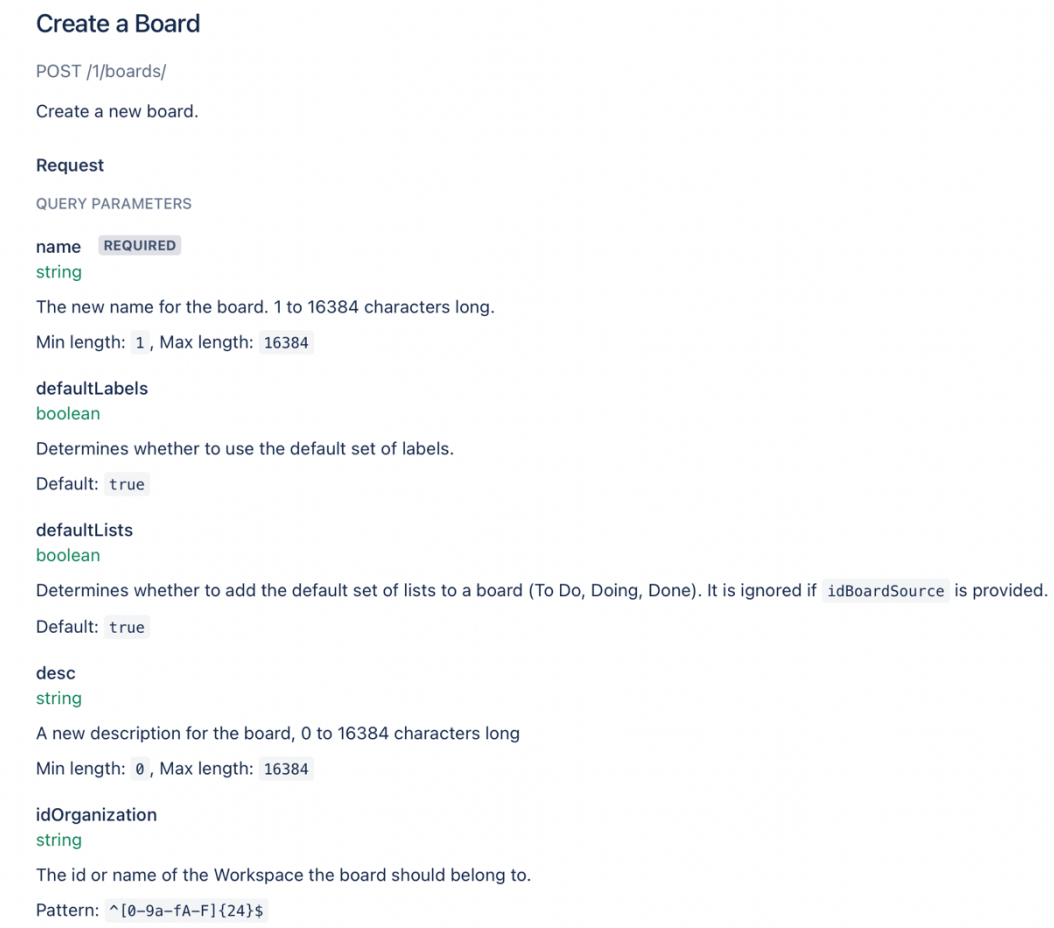

Рисунок 4.4 - опис методів

4.2 Програмна реалізація

Для початку створимо всі необхідні змінні для автоматизації управління проектом і додамо їх у колекцію environments (Рис 4.5).

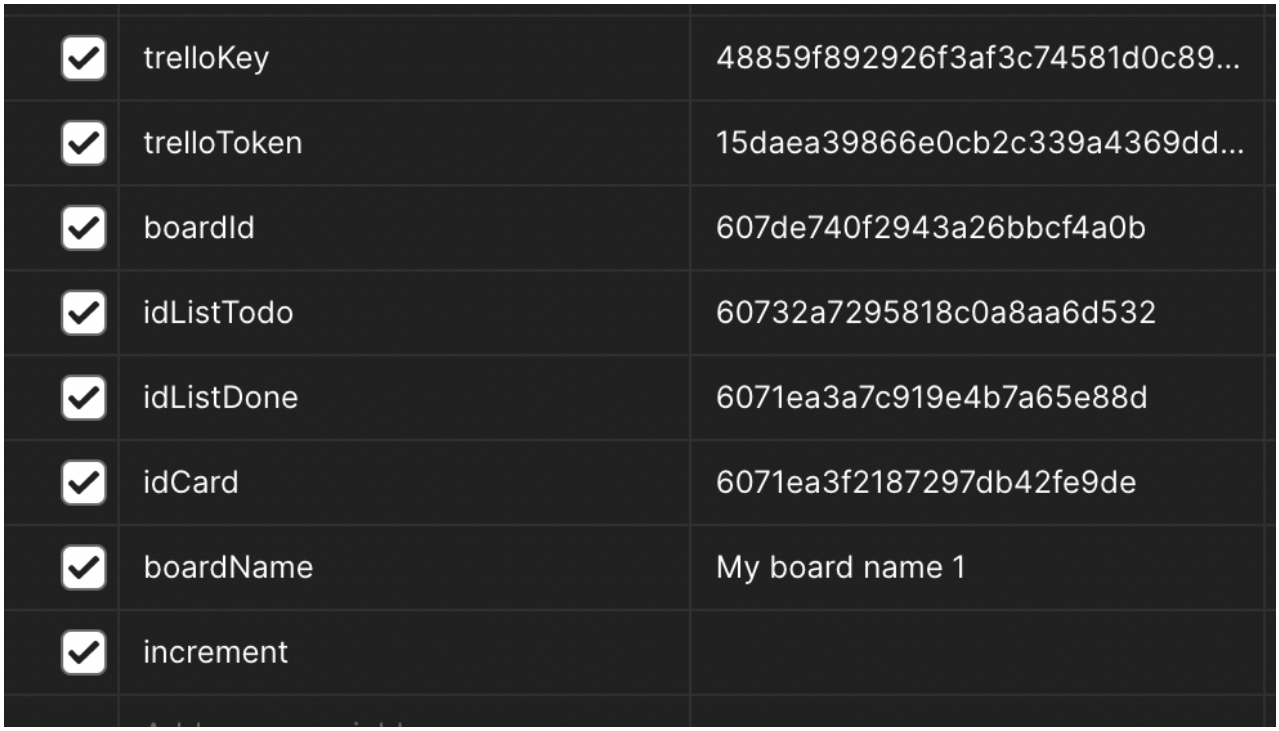

Рисунок 4.5 - для автоматизації управління проектом

- trelloKey - ключ для взаємодії з trello
- trelloToken – токен для взаємодії з trello
- boardID – id дошки для роботи інших методів
- idListTodo – id листа для планування
- IdListDone – id листа для статусу готово
- idCard - id карти
- boardName - ім'я дошки
- increment - змінна для створення унікального імені дошки

Найголовнішим і першим методом у реалізації буде створення дошки. Оскільки надалі всі сутності створюватимуться саме на ній. Створення відбувається за допомогою http POST методу:

https://api.trello.com/1/boards/?name={{boardName}}&key={{trelloKey}}&token={{trelloToken}}.

Перед відправленням методу відбувається створення унікального імені дошки для уникнення дублікатів та унікалізації даних

```
function getUniqueBoardName () {
    if( pm.environment.get("increment") === undefined)
    {
        pm.environment.set("increment", "0");
    }
    let variableForName =
pm.environment.get("increment");
    const boardName = "My board name " +
++variableForName;
    pm.environment.set("increment", variableForName);
    return boardName;
    }
```

Після отримання відповіді від сервера відбувається перевірка статус коду. Надалі ця перевірка буде на всіх методах.

```
pm.test("Status code is 200", function () {
    pm.response.to.have.status(200);
});
```

А також валідація на те, що дошка створилася саме з тим ім'ям, яке було задано і що вона приватна та запис у змінну id дошки для подальшого створення на ньому листів та карток.

```
pm.test("Board should be created", function () {
pm.expect(jsonData.name).to.eql(pm.environment.get("boar
dName"));
    pm.expect(jsonData.closed).to.eql(false);
});
pm.test("Board should be private", function () {
pm.expect(jsonData.prefs.permissionLevel).to.eql("privat
e");
```

```
    pm.environment.set("boardId", jsonData.id);
});
```

Після успішного створення дошки потрібно створення двох листів зі статусами TODO та DONE. Ці статуси потрібні для простеження прогресу активних завдань та їх оновлення. Створення відбувається за допомогою http POST методу:

https://api.trello.com/1/lists?key={{trelloKey}}&token={{trelloToken}}&name=TODO&idBoard={{boardId}}

У даних методах присутня валідація на статус код, яка була згадана раніше. Також валідація на ім'я відповідно для кожного листа своє ім'я:

```
pm.test(" the name of the list is DONE", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.name).to.eql("DONE");
});
```

Те, що листи створені під саме тією дошкою з вказаним boardId:

```
pm.test("list was created in the desired board",
function () {
    const Id =
postman.getEnvironmentVariable("boardId");
    var jsonData = pm.response.json();
    pm.expect(jsonData.idBoard).to.eql(Id);
});
```

Після цього відбувається створення картки із завданням. Створення відбувається за допомогою http POST методу:

https://api.trello.com/1/cards?key={{trelloKey}}&token={{trelloToken}}&name=Postman&idList={{idListTodo}}

Де після отримання відповіді відбувається перевірка на створення під дошкою із зазначеним boardId та листом зі стасусом TODO

```
pm.test("test that the card was created in the desired
TODO list", function () {
```

```
    const                    Id                    =
postman.getEnvironmentVariable("idListTodo");
    var jsonData = pm.response.json();
    pm.expect(jsonData.idList).to.eql(Id);
});
pm.test("list was created in the desired board", function
() {
    const Id = postman.getEnvironmentVariable("boardId");
    var jsonData = pm.response.json();
    pm.expect(jsonData.idBoard).to.eql(Id);
});
```

Також було додано перевірку створення картки із зазначеним ім'ям у методі:

```
pm.test(" the name of the list is Postman", function () {
    var jsonData = pm.response.json();
    pm.expect(jsonData.name).to.eql("Postman");
});
```

Після цього відбувається апдейт картки із завданням. Вона переходить з листа TODO в лист DONE. Апдейт відбувається за допомогою http PUT методу:

https://api.trello.com/1/cards/:id?key={{trelloKey}}&token={{trelloToken}}&name=Postman&idList={{idListDone}}

Де при отриманні відповіді відбувається перевірка, що картка перейшла під інший лист і так само все ще під потрібною дошкою

```
pm.test("test that the card was created in the desired
TODO list", function () {
    const                    Id                    =
postman.getEnvironmentVariable("idListTodo");
    var jsonData = pm.response.json();
    pm.expect(jsonData.idList).to.eql(Id);
});
```

```
pm.test("list was created in the desired board", function
() {
    const Id = postman.getEnvironmentVariable("boardId");
    var jsonData = pm.response.json();
    pm.expect(jsonData.idBoard).to.eql(Id);
});
```

Наприкінці флоу буде відбуватися видалення дошки за допомогою методу DELETE:

https://api.trello.com/1/boards/:id?key={{trelloKey}}&token={{trelloToken}}

а також перевірка, що дана дошка була видалена за допомогою методу GET:

https://api.trello.com/1/boards/:id?key={{trelloKey}}&token={{trelloToken}}

Використовуючи раніше встановлений newman за допомогою команди: npm install -g newman (Рис 4.6)

```
==> Installing dependencies for newman: brotli, c-ares, icu4c, libnghttp2, libuv, ca-certificates, openssl@1.1 and node
==> Installing newman dependency: brotli
==> Pouring brotli--1.0.9.arm64_big_sur.bottle.tar.gz
🍺  /opt/homebrew/Cellar/brotli/1.0.9: 25 files, 2.3MB
==> Installing newman dependency: c-ares
==> Pouring c-ares--1.18.1.arm64_big_sur.bottle.tar.gz
🍺  /opt/homebrew/Cellar/c-ares/1.18.1: 87 files, 665.6KB
==> Installing newman dependency: icu4c
==> Pouring icu4c--69.1.arm64_big_sur.bottle.tar.gz
🍺  /opt/homebrew/Cellar/icu4c/69.1: 259 files, 73.3MB
==> Installing newman dependency: libnghttp2
==> Pouring libnghttp2--1.46.0.arm64_big_sur.bottle.tar.gz
🍺  /opt/homebrew/Cellar/libnghttp2/1.46.0: 13 files, 690.3KB
==> Installing newman dependency: libuv
==> Pouring libuv--1.42.0.arm64_big_sur.bottle.tar.gz
🍺  /opt/homebrew/Cellar/libuv/1.42.0: 49 files, 3.4MB
==> Installing newman dependency: ca-certificates
==> Pouring ca-certificates--2021-10-26.all.bottle.tar.gz
==> Regenerating CA certificate bundle from keychain, this may take a while...
🍺  /opt/homebrew/Cellar/ca-certificates/2021-10-26: 3 files, 208.5KB
==> Installing newman dependency: openssl@1.1
==> Pouring openssl@1.1--1.1.1l_1.arm64_big_sur.bottle.tar.gz
🍺  /opt/homebrew/Cellar/openssl@1.1/1.1.1l_1: 8,073 files, 18MB
==> Installing newman dependency: node
==> Pouring node--17.2.0.arm64_big_sur.bottle.tar.gz
🍺  /opt/homebrew/Cellar/node/17.2.0: 2,018 files, 44.4MB
==> Installing newman
==> Pouring newman--5.3.0.arm64_big_sur.bottle.tar.gz
🍺  /opt/homebrew/Cellar/newman/5.3.0: 5,584 files, 37.8MB
==> `brew cleanup` has not been run in the last 30 days, running now...
Disable this behaviour by setting HOMEBREW_NO_INSTALL_CLEANUP.
Hide these hints with HOMEBREW_NO_ENV_HINTS (see `man brew`).
Removing: /opt/homebrew/Cellar/openssl@1.1/1.1.1l... (8,073 files, 18MB)
Removing: /Users/serdu4ok/Library/Caches/Homebrew/wget--1.21.1_1... (1.4MB)
Removing: /Users/serdu4ok/Library/Logs/Homebrew/wget... (64B)
Removing: /Users/serdu4ok/Library/Logs/Homebrew/libidn2... (64B)
Removing: /Users/serdu4ok/Library/Logs/Homebrew/libunistring... (64B)
Removing: /Users/serdu4ok/Library/Logs/Homebrew/gettext... (64B)
Removing: /Users/serdu4ok/Library/Logs/Homebrew/openssl@1.1... (64B)
Pruned 0 symbolic links and 3 directories from /opt/homebrew
```

Рисунок 4.6 – встановлення newman

Експортуючи колекцію та змінні енвайронменту як файл json запустимо колекцію з енвайронментом через термінал і побачимо результат роботи та відповіді з логами

4.3 Тестування

4.3.1 Ручне тестування (Postman)

Успішне виконання методу створення дошки (Рис 4.7)

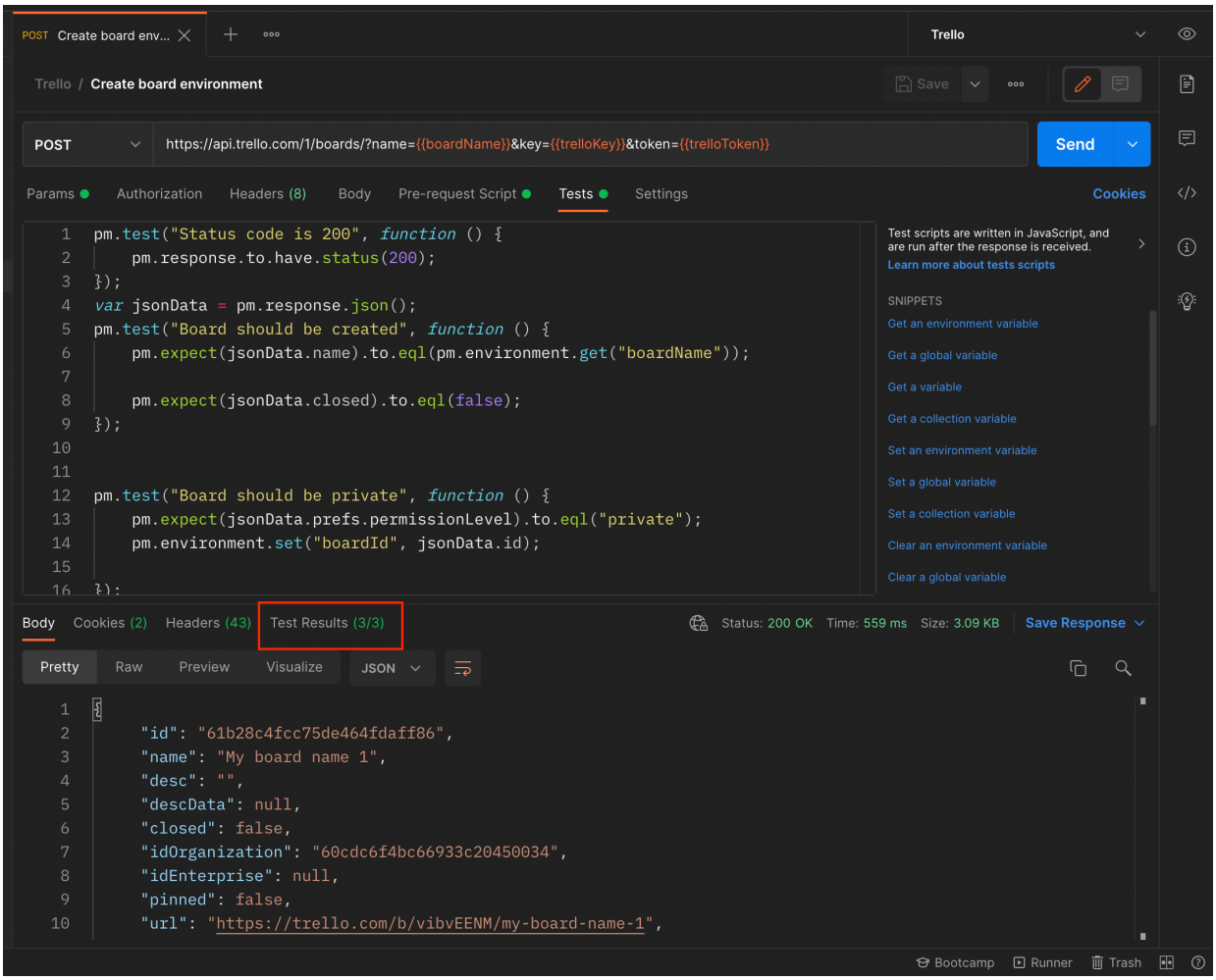

Рисунок 4.7 – створення дошки

Та результат відображення на сайті (Рис 4.8)

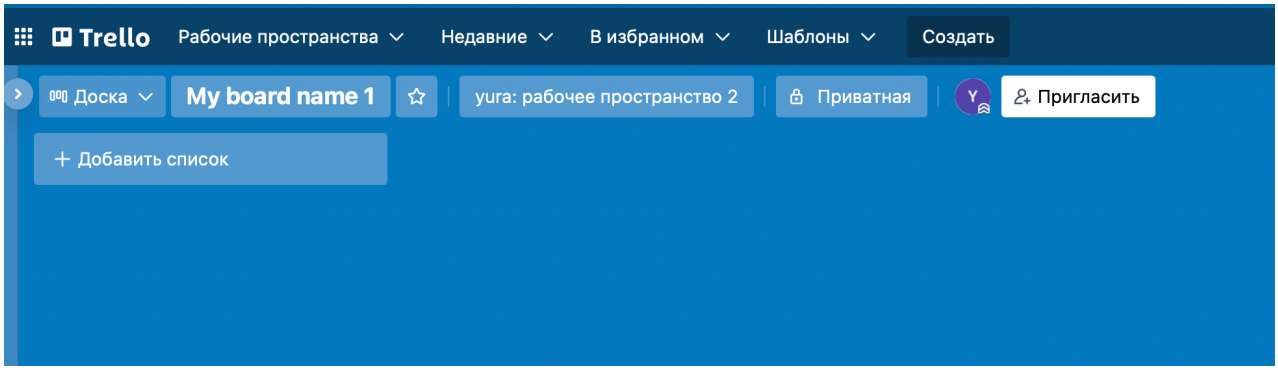

Рисунок 4.8 - відображення дошки на сайті

Успішне виконання методу створення листа (Рис 4.9)

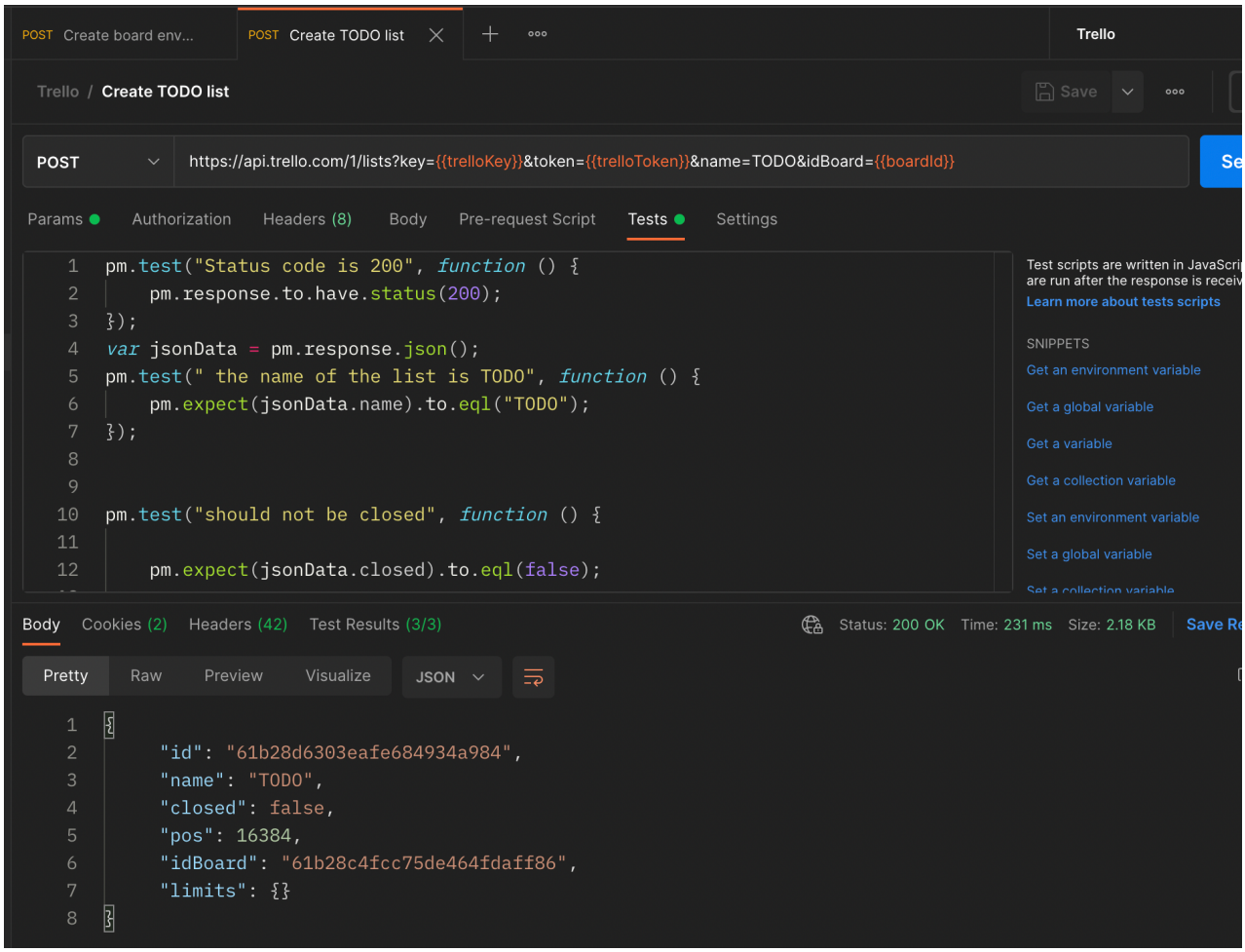

Рисунок 4.9 – створення листа

Та результат відображення створення листа на сайті (Рис 4.10)

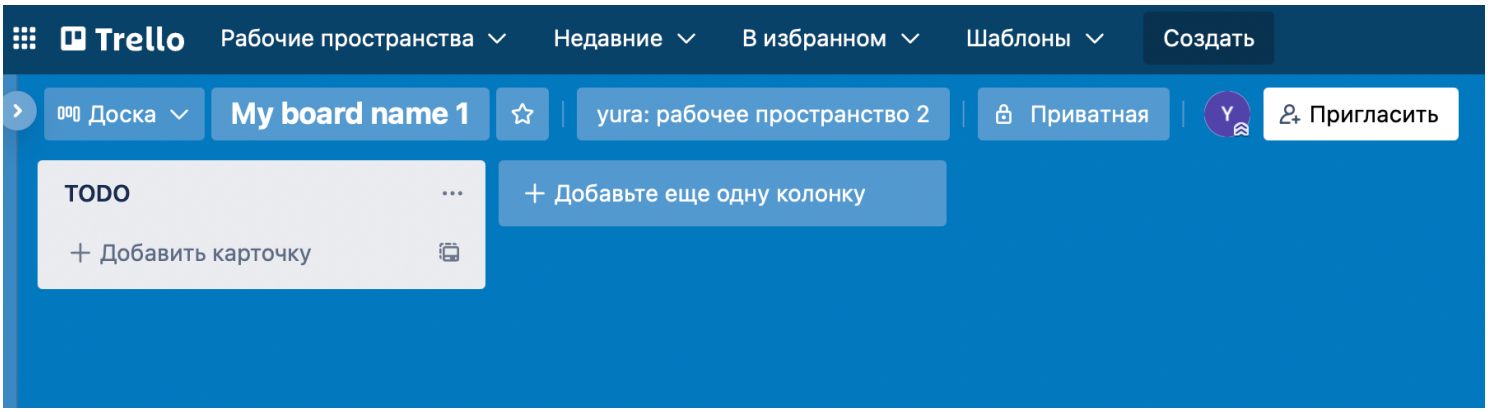

Рисунок 4.10 - відображення листа на сайті

Як можна побачити, лист успішно був створений на дошці та всі написані тести пройшли успішно.

Успішне виконання методу створення картки (Рис 4.11)

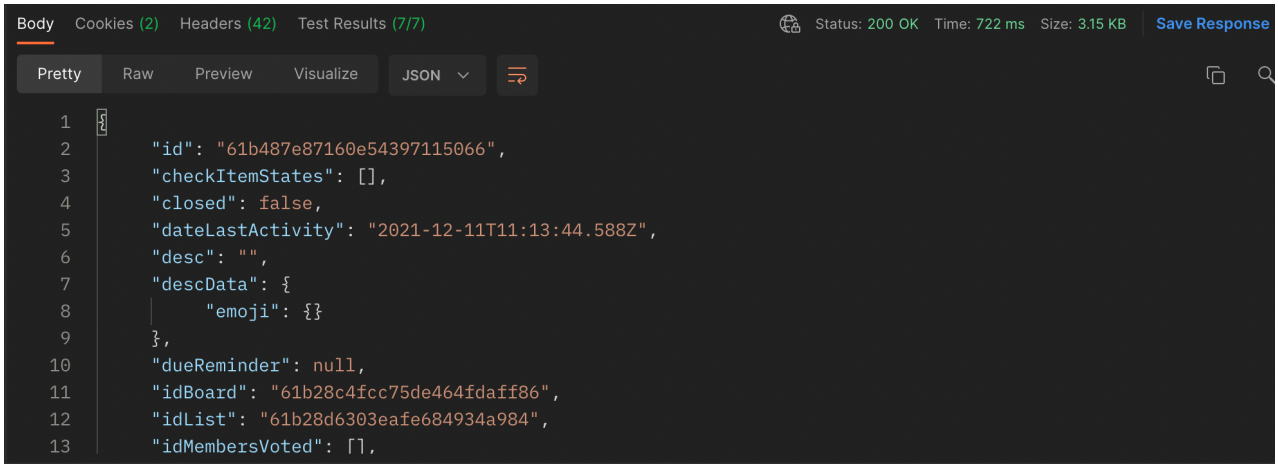

Рисунок 4.11 – створення картки

Та результат відображення створення картки на сайті (Рис 4.12)

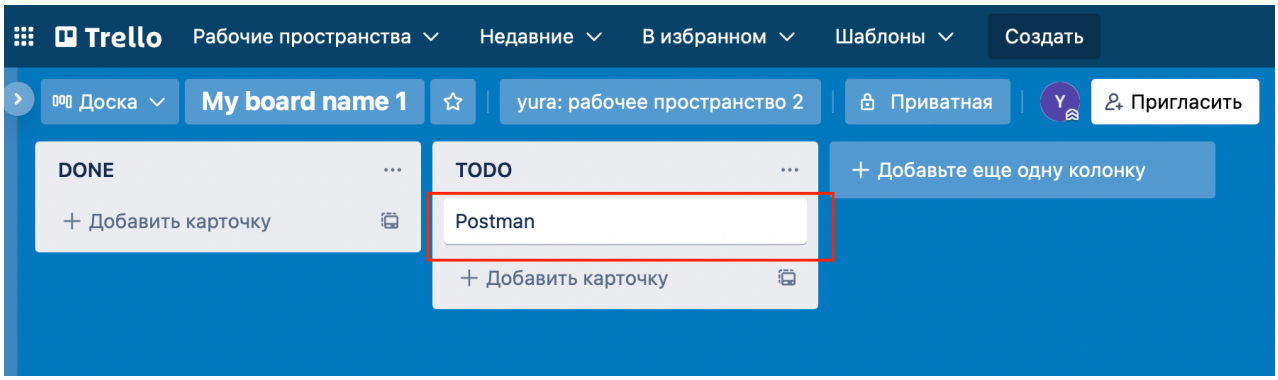

Рисунок 4.12 - відображення картки на сайті

Результат оновлення картки та перенесення в іншу дошку (Рис 4.13)

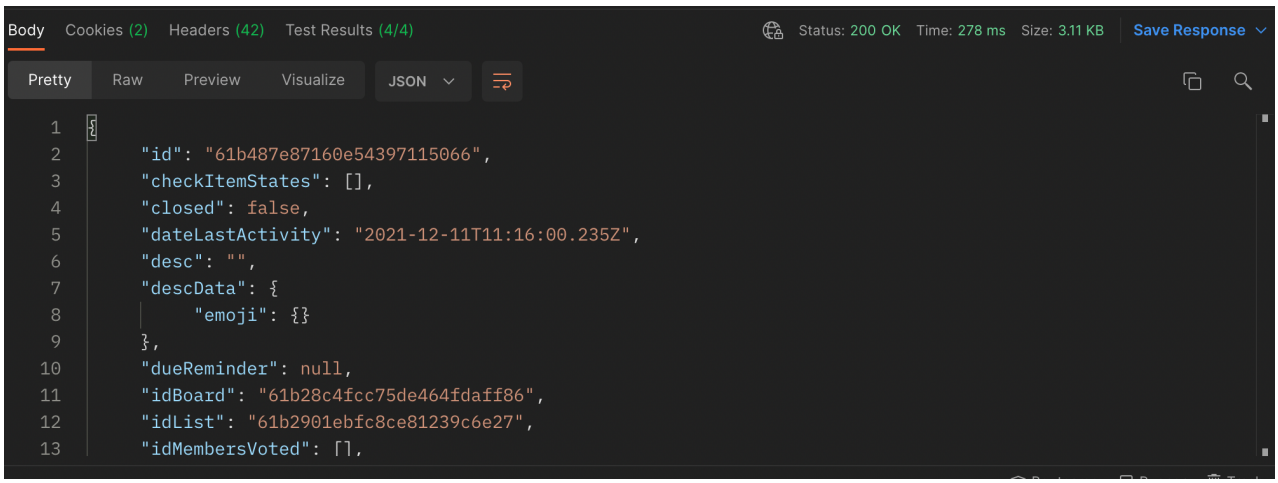

Рисунок 4.13 – апдейт картки

Та результат відображення оновленої картки на сайті (Рис 4.14)

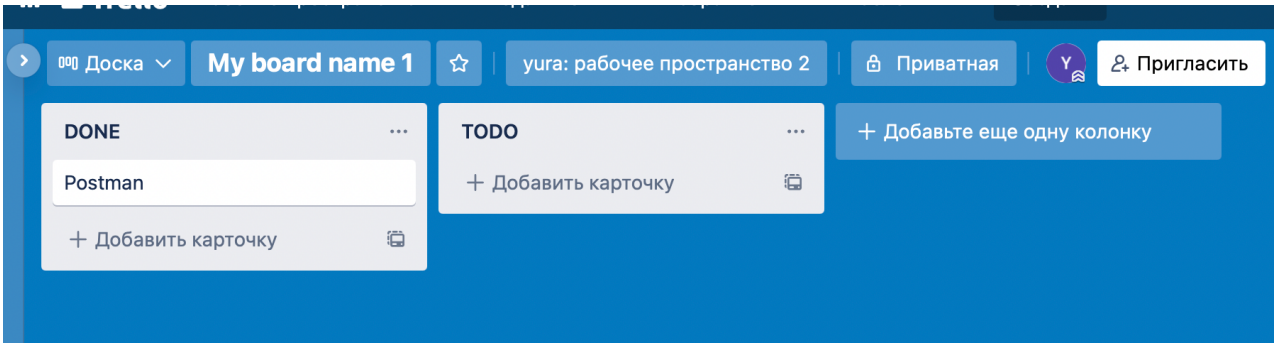

Рисунок 4.14 – відображення оновленої картки на сайті

Далі виконаємо команду віддалення дошки(Рис 4.15) і вдамося в тому що вона успішно була видалена (Рис 4.16)

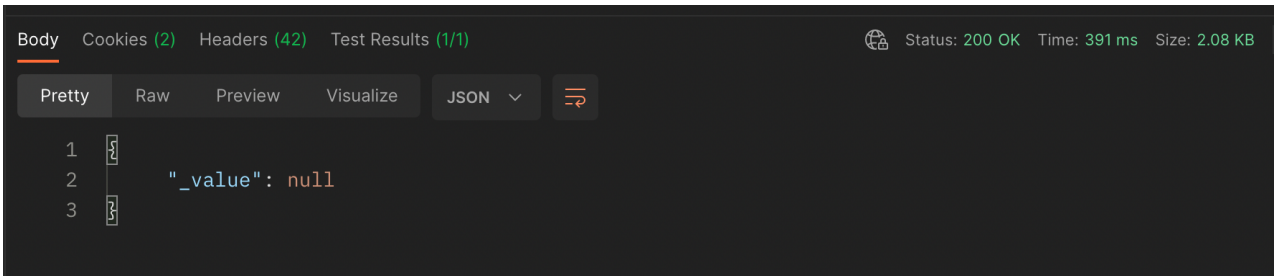

Рисунок 4.15 – видалення дошки

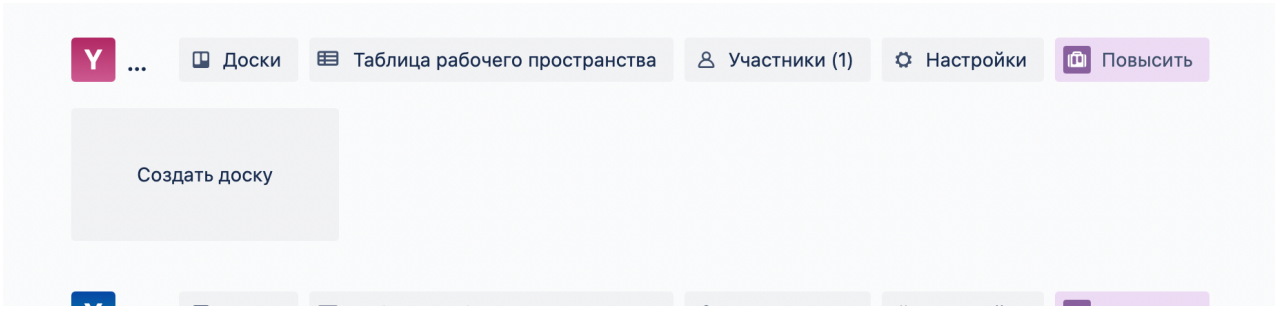

Рисунок 4.16 - відображення на сайті видаленної картки

Як можна побачити було отримано очікувані результати.

4.3.2 Автоматичне тестування (Postman)

Запустимо колекцію за допомогою автоматичного запуску (Рис 4.17)

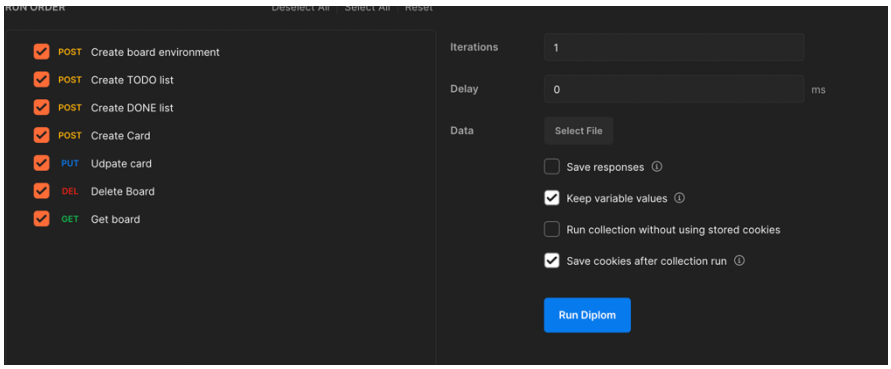

Рисунок 4.17 - запуск колекції

Як можна побачити всі тести пройшли успішно та отримано очікуваний результат (Рис 4.18)

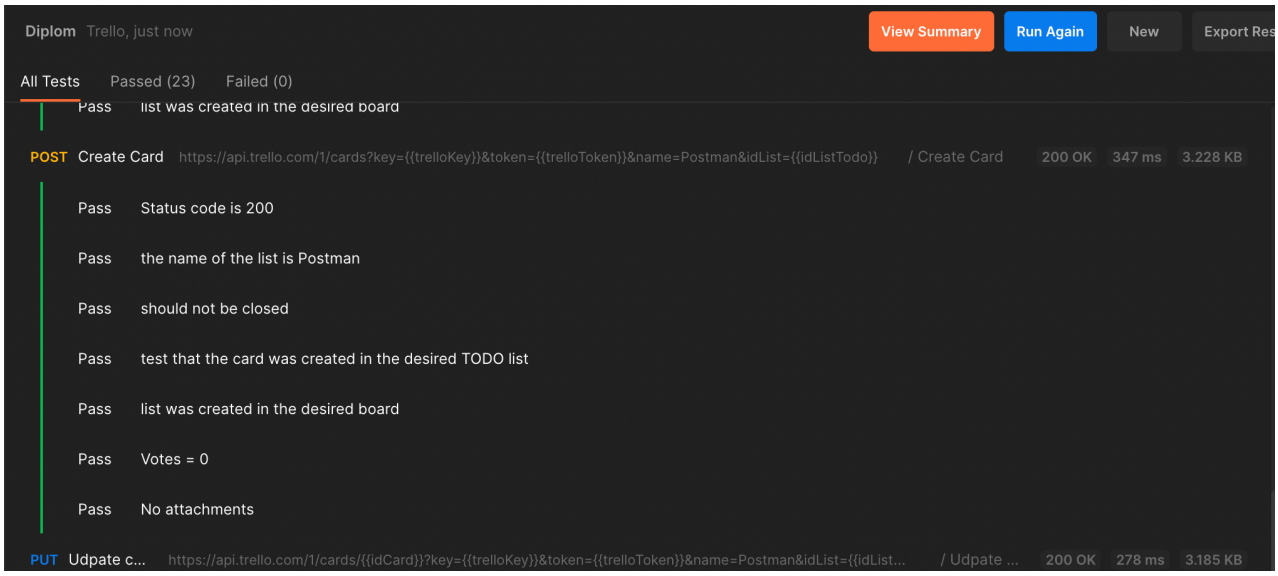

Рисунок 4.18 - результат виконання колекції

4.3.3 Автоматичне тестування (Newman)

Вивантаживши заздалегідь колекцію та енвайронмент запустимо через термінал за допомогою команди.

```
newman run Diplom.postman_collection.json -e Trello.postman_environment.json
```

І перевіримо результат у терміналі (Рис 4.19)

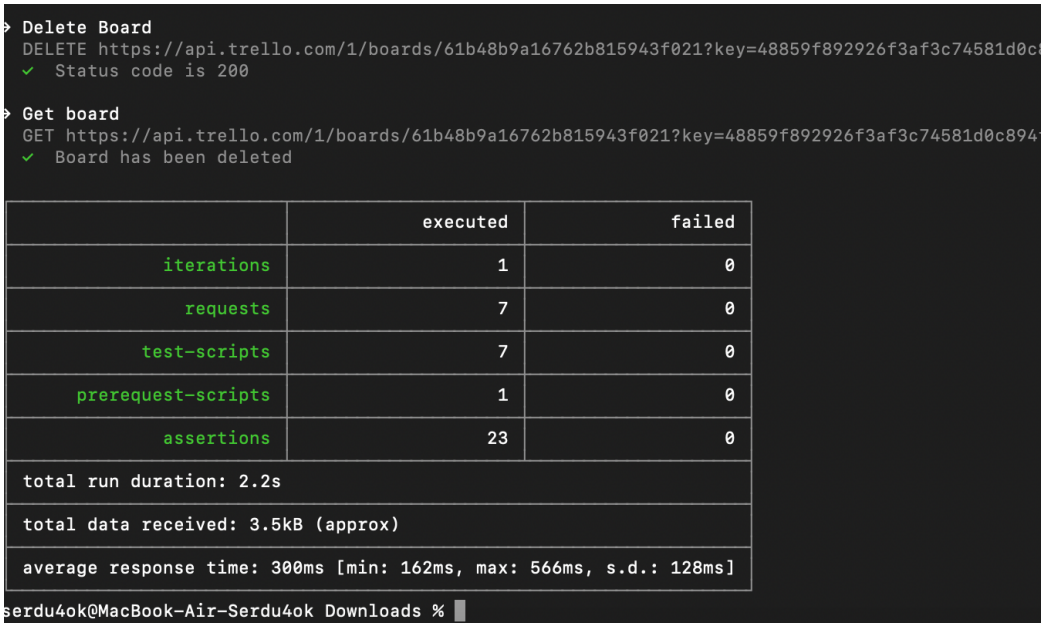

Рисунок 4.19 - результат роботи через термінал

ВИСНОВОК

У ході роботи було проаналізовано ринок та обрано найбільш оптимальну платформу. Було розроблено Інформаційну технологію автоматизованого управління проектами за допомогою інструментарію Postman. Дозволяє створювати дошки, листи з певними статусами, картки, а також оновлювати картки. Видаляє дошку та перевіряє її статус. Є можливість впровадження в CI/CD і система, що легко машстабується.

Проведено тестування на декількох видах а саме:

- Ручне (Postman)
- Автоматичне(Postman)
- З терміналу (Newman)

СПИСОК ЛІТЕРАТУРИ

1. Porteous C. Why Time Is Your Most Important Business Resource [Електронний ресурс] / Chris Porteous – Режим доступу до ресурсу: 1. https://www.entrepreneur.com/article/328205.
2. Гибкие методологии разработки. – 111 с.
3. WHAT IS AGILE? WHAT IS SCRUM? [Електронний ресурс] – Режим доступу до ресурсу: https://www.cprime.com/resources/what-is-agile-what-is-scrum/.
4. Collaboration Software [Електронний ресурс] – Режим доступу до ресурсу: https://www.computerworld.com/article/3226447/what-is-trello-a-guide-to-atlassians-collaboration-and-work-management-tool.html.
5. 12 Best Jira Alternatives to Try in 2021 (Free and Paid) [Електронний ресурс] – Режим доступу до ресурсу: https://clickup.com/blog/jira-alternatives/.
6. What is Trello? A guide to Atlassian’s collaboration and work management tool [Електронний ресурс] – Режим доступу до ресурсу: https://www.computerworld.com/article/3226447/what-is-trello-a-guide-to-atlassians-collaboration-and-work-management-tool.html.
7. What Is Jira: An Overview of a Unique Project Management Tool [Електронний ресурс] – Режим доступу до ресурсу: https://www.fool.com/the-blueprint/what-is-jira/.
8. What is Postman? [Електронний ресурс] – Режим доступу до ресурсу: https://www.postman.com/product/what-is-postman/.
9. The Trello Rest API [Електронний ресурс] – Режим доступу до ресурсу: https://developer.atlassian.com/cloud/trello/rest/api-group-actions/.
10. Newman the cli companion for postman [Електронний ресурс] – Режим доступу до ресурсу: https://www.npmjs.com/package/newman.

ДОДАТОК А

•

```
{
  "info": {
    "_postman_id": "9105e5fd-2d9e-4cbb-9423-45be2738a815",
    "name": "Diplom",
    "schema":
"https://schema.getpostman.com/json/collection/v2.1.0/collection.json"
  },
  "item": [
    {
      "name": "Create board environment",
      "event": [
        {
          "listen": "test",
          "script": {
            "exec": [
              "pm.test(\"Status code is 200\", function () {",
              "    pm.response.to.have.status(200);",
              "});",
              "var jsonData = pm.response.json();",
              "pm.test(\"Board should be created\", function () {",
              "
pm.expect(jsonData.name).to.eql(pm.environment.get(\"boardName\"));",
              "",
              "    pm.expect(jsonData.closed).to.eql(false);",
              "});",
              "",
              "",
              "pm.test(\"Board should be private\", function () {",
              "
pm.expect(jsonData.prefs.permissionLevel).to.eql(\"private\");",
              "    pm.environment.set(\"boardId\", jsonData.id);",
              "    ",
              "});"
            ],
            "type": "text/javascript"
          }
        },
        {
          "listen": "prerequest",
          "script": {
            "exec": [
              "if(pm.environment.get(\"increment\") ===
pm.info.iterationCount)",
              "{",
              "    pm.environment.unset(\"increment\");",
              "}",
              "pm.environment.set(\"boardName\", getUniqueBoardName());",
              "",
              "",
              "",
              "function getUniqueBoardName () {",
              "    if( pm.environment.get(\"increment\") === undefined)",
              "    {",
              "        pm.environment.set(\"increment\", \"0\");",
              "    }",
              "    let variableForName = pm.environment.get(\"increment\");",
              "    ",
```

```
                "    const boardName = \"My board name \" + ++variableForName;",
                "    pm.environment.set(\"increment\", variableForName);",
                "    return boardName;",
                "    ",
                "}",
                ""
              ],
              "type": "text/javascript"
            }
          }
        ],
        "request": {
          "method": "POST",
          "header": [],
          "url": {
            "raw":
"https://api.trello.com/1/boards/?name={{boardName}}&key={{trelloKey}}&token={{
trelloToken}}",
            "protocol": "https",
            "host": [
              "api",
              "trello",
              "com"
            ],
            "path": [
              "1",
              "boards",
              ""
            ],
            "query": [
              {
                "key": "name",
                "value": "{{boardName}}"
              },
              {
                "key": "key",
                "value": "{{trelloKey}}"
              },
              {
                "key": "token",
                "value": "{{trelloToken}}"
              }
            ]
          }
        },
        "response": []
      },
      {
        "name": "Create TODO list",
        "event": [
          {
            "listen": "test",
            "script": {
              "exec": [
                "pm.test(\"Status code is 200\", function () {",
                "    pm.response.to.have.status(200);",
                "});",
                "var jsonData = pm.response.json();",
                "pm.test(\" the name of the list is TODO\", function () {",
                "    pm.expect(jsonData.name).to.eql(\"TODO\");",
                "});",
                "",
```

```
                "",
                "pm.test(\"should not be closed\", function () {",
                "    ",
                "    pm.expect(jsonData.closed).to.eql(false);",
                "    ",
                "    pm.environment.set(\"idListTodo\", jsonData.id);",
                "});",
                "",
                ""
              ],
              "type": "text/javascript"
            }
          }
        ],
        "request": {
          "method": "POST",
          "header": [],
          "body": {
            "mode": "raw",
            "raw": ""
          },
          "url": {
            "raw":
"https://api.trello.com/1/lists?key={{trelloKey}}&token={{trelloToken}}&name=TO
DO&idBoard={{boardId}}",
            "protocol": "https",
            "host": [
              "api",
              "trello",
              "com"
            ],
            "path": [
              "1",
              "lists"
            ],
            "query": [
              {
                "key": "key",
                "value": "{{trelloKey}}"
              },
              {
                "key": "token",
                "value": "{{trelloToken}}"
              },
              {
                "key": "name",
                "value": "TODO"
              },
              {
                "key": "",
                "value": "",
                "disabled": true
              },
              {
                "key": "idBoard",
                "value": "{{boardId}}"
              }
            ]
          }
        },
        "response": []
      },
```

```
    {
      "name": "Create DONE list",
      "event": [
        {
          "listen": "test",
          "script": {
            "exec": [
              "pm.test(\"Status code is 200\", function () {",
              "    pm.response.to.have.status(200);",
              "});",
              "pm.test(\" the name of the list is DONE\", function () {",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.name).to.eql(\"DONE\");",
              "});",
              "pm.test(\"should not be closed\", function () {",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.closed).to.eql(false);",
              "    pm.environment.set(\"idListDone\", jsonData.id);",
              "});",
              "pm.test(\"list was created in the desired board\", function ()
{",
              "    const Id = postman.getEnvironmentVariable(\"boardId\");",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.idBoard).to.eql(Id);",
              "});"
            ],
            "type": "text/javascript"
          }
        }
      ],
      "request": {
        "method": "POST",
        "header": [],
        "url": {
          "raw":
"https://api.trello.com/1/lists?key={{trelloKey}}&token={{trelloToken}}&name=DO
NE&idBoard={{boardId}}",
          "protocol": "https",
          "host": [
            "api",
            "trello",
            "com"
          ],
          "path": [
            "1",
            "lists"
          ],
          "query": [
            {
              "key": "key",
              "value": "{{trelloKey}}"
            },
            {
              "key": "token",
              "value": "{{trelloToken}}"
            },
            {
              "key": "name",
              "value": "DONE"
            },
            {
              "key": "idBoard",
```

```
              "value": "{{boardId}}"
            },
            {
              "key": "",
              "value": "",
              "disabled": true
            }
          ]
        }
      },
      "response": []
    },
    {
      "name": "Create Card",
      "event": [
        {
          "listen": "test",
          "script": {
            "exec": [
              "pm.test(\"Status code is 200\", function () {",
              "    pm.response.to.have.status(200);",
              "});",
              "pm.test(\" the name of the list is Postman\", function () {",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.name).to.eql(\"Postman\");",
              "});",
              "pm.test(\"should not be closed\", function () {",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.closed).to.eql(false);",
              "    pm.environment.set(\"idCard\", jsonData.id);",
              "});",
              "",
              "pm.test(\"test that the card was created in the desired TODO
list\", function () {",
              "    const Id = postman.getEnvironmentVariable(\"idListTodo\");",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.idList).to.eql(Id);",
              "});",
              "",
              "pm.test(\"list was created in the desired board\", function ()
{",
              "    const Id = postman.getEnvironmentVariable(\"boardId\");",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.idBoard).to.eql(Id);",
              "});",
              "",
              "pm.test(\"Votes = 0\", function () {",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.badges.votes).to.eql(0);",
              "}); ",
              "",
              "",
              "pm.test(\"No attachments\", function () {",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.attachments).to.eql([]);",
              "}); ",
              "",
              ""
            ],
            "type": "text/javascript"
          }
        }
```

```
      ],
      "request": {
        "method": "POST",
        "header": [],
        "url": {
          "raw":
"https://api.trello.com/1/cards?key={{trelloKey}}&token={{trelloToken}}&name=Po
stman&idList={{idListTodo}}",
          "protocol": "https",
          "host": [
            "api",
            "trello",
            "com"
          ],
          "path": [
            "1",
            "cards"
          ],
          "query": [
            {
              "key": "key",
              "value": "{{trelloKey}}"
            },
            {
              "key": "token",
              "value": "{{trelloToken}}"
            },
            {
              "key": "name",
              "value": "Postman"
            },
            {
              "key": "idList",
              "value": "{{idListTodo}}"
            }
          ]
        }
      },
      "response": []
    },
    {
      "name": "Udpate card",
      "event": [
        {
          "listen": "test",
          "script": {
            "exec": [
              "pm.test(\"Status code is 200\", function () {",
              "    pm.response.to.have.status(200);",
              "});",
              "pm.test(\"The name of the card is still Postman\", function ()
{",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.name).to.eql(\"Postman\");",
              "});",
              "",
              "",
              "pm.test(\"test that the card was moved to the desired DONE
list\", function () {",
              "    const Id = postman.getEnvironmentVariable(\"idListDone\");",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.idList).to.eql(Id);",
```

```
              "});",
              "",
              "pm.test(\"list was created in the desired board\", function ()
{",
              "    const Id = postman.getEnvironmentVariable(\"boardId\");",
              "    var jsonData = pm.response.json();",
              "    pm.expect(jsonData.idBoard).to.eql(Id);",
              "});",
              " ",
              "",
              ""
            ],
            "type": "text/javascript"
          }
        }
      ],
      "request": {
        "method": "PUT",
        "header": [],
        "url": {
          "raw":
"https://api.trello.com/1/cards/:id?key={{trelloKey}}&token={{trelloToken}}&nam
e=Postman&idList={{idListDone}}",
          "protocol": "https",
          "host": [
            "api",
            "trello",
            "com"
          ],
          "path": [
            "1",
            "cards",
            ":id"
          ],
          "query": [
            {
              "key": "key",
              "value": "{{trelloKey}}"
            },
            {
              "key": "token",
              "value": "{{trelloToken}}"
            },
            {
              "key": "name",
              "value": "Postman"
            },
            {
              "key": "idList",
              "value": "{{idListDone}}"
            }
          ],
          "variable": [
            {
              "key": "id",
              "value": "{{idCard}}"
            }
          ]
        }
      },
      "response": []
    },
```

```
    {
      "name": "Delete Board",
      "event": [
        {
          "listen": "test",
          "script": {
            "exec": [
              "pm.test(\"Status code is 200\", function () {",
              "    pm.response.to.have.status(200);",
              "});",
              "",
              ""
            ],
            "type": "text/javascript"
          }
        }
      ],
      "request": {
        "method": "DELETE",
        "header": [],
        "url": {
          "raw":
"https://api.trello.com/1/boards/:id?key={{trelloKey}}&token={{trelloToken}}",
          "protocol": "https",
          "host": [
            "api",
            "trello",
            "com"
          ],
          "path": [
            "1",
            "boards",
            ":id"
          ],
          "query": [
            {
              "key": "name",
              "value": "My new board",
              "disabled": true
            },
            {
              "key": "key",
              "value": "{{trelloKey}}"
            },
            {
              "key": "token",
              "value": "{{trelloToken}}"
            }
          ],
          "variable": [
            {
              "key": "id",
              "value": "{{boardId}}"
            }
          ]
        }
      },
      "response": []
    },
    {
      "name": "Get board",
      "event": [
```

```
        {
          "listen": "test",
          "script": {
            "exec": [
              "pm.test(\"Board has been deleted\", function () {",
              "    pm.response.to.have.status(404);",
              "});",
              "",
              ""
            ],
            "type": "text/javascript"
          }
        }
      ],
      "request": {
        "method": "GET",
        "header": [],
        "url": {
          "raw":
"https://api.trello.com/1/boards/:id?key={{trelloKey}}&token={{trelloToken}}",
          "protocol": "https",
          "host": [
            "api",
            "trello",
            "com"
          ],
          "path": [
            "1",
            "boards",
            ":id"
          ],
          "query": [
            {
              "key": "name",
              "value": "My new board",
              "disabled": true
            },
            {
              "key": "key",
              "value": "{{trelloKey}}"
            },
            {
              "key": "token",
              "value": "{{trelloToken}}"
            }
          ],
          "variable": [
            {
              "key": "id",
              "value": "{{boardId}}"
            }
          ]
        }
      },
      "response": []
    }
  ]
}
```

ДОДАТОК В


```
{
  "id": "a9485b74-ba68-4997-8f4c-c8d3d8ac9d9f",
  "name": "Trello",
  "values": [
    {
      "key": "uuid",
      "value": "ef8b4aa1-fe56-417c-ac70-4a0ea5bb241e",
      "enabled": true
    },
    {
      "key": "trelloKey",
      "value": "48859f892926f3af3c74581d0c894f63",
      "enabled": true
    },
    {
      "key": "trelloToken",
      "value": "ibhwnde0cb2c30vuejnf0000vyewdbfj1110e4a78625c1e8be2222",
      "enabled": true
    },
    {
      "key": "boardId",
      "value": "607de740f2943a26bbcf4a0b",
      "enabled": true
    },
    {
      "key": "idListTodo",
      "value": "60732a7295818c0a8aa6d532",
      "enabled": true
    },
    {
      "key": "idListDone",
      "value": "6071ea3a7c919e4b7a65e88d",
      "enabled": true
    },
    {
      "key": "idCard",
      "value": "6071ea3f2187297db42fe9de",
      "enabled": true
    },
    {
      "key": "boardName",
      "value": "My board name 1",
      "enabled": true
    },
    {
      "key": "increment",
      "value": "",
      "enabled": true
    }
  ],
  "_postman_variable_scope": "environment",
  "_postman_exported_at": "2021-12-08T21:49:00.935Z",
  "_postman_exported_using": "Postman/8.12.1"
}
```