

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ, МОДИФІКАЦІЯ ТА ПРОГРАМНА
РЕАЛІЗАЦІЯ МЕТОДІВ УПРАВЛІННЯ ПРОЄКТАМИ
RESEARCH, MODIFICATION AND SOFTWARE
IMPLEMENTATION OF PROECT MANAGEMENT METHODS

Виконав: студент(ка) 2 курсу, групи КНТ-119м
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)
Інженерія програмного забезпечення

Земляна А.В.

(прізвище та ініціали)

Керівник Каплієнко Т.І.

(прізвище та ініціали)

Рецензент Шитікова О.В.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти магістр
 Спеціальність 121 Інженерія програмного забезпечення
 (код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” 20 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Земляної Анастасії Володимирівни

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження, модифікація та програмна реалізація методів управління проєктами. Research, Modification and Software Implementation of Project Management Methods

керівник проєкту (роботи) Каплієнко Тетяна Ігорівна, к.т.н., доцент,
 (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від 03 листопада 2020 року №301

2. Строк подання студентом проєкту (роботи) 10 грудня 2020 року

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

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Розробка програми. 5. Експлуатація, тестування та експериментальне дослідження програми. 6. Економіко-організаційна частина. 7. Охорона праці та безпека у надзвичайних ситуаціях 8. Додатки. 5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Каплієнко Т.І., доцент		
6 Економіко-організаційна частина	Пожуєва Т.О., професор		
7 Охорона праці та безпека в надзвичайних ситуаціях	Коробко О.В., ст.викладач		
Нормоконтролер	Андрєєв М. О., асистент		

7. Дата видачі завдання 10 вересня 2020 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2–3 тиждень	Розділ 1
3	Розробка та удосконалення методів та алгоритмів вирішення задачі.	4–5 тиждень	Розділ 2
4	Розробка архітектури програми.	6 тиждень	Розділ 3
5	Розробка програми.	7–8 тиждень	Розділ 4
6	Експлуатація, тестування та експериментальне дослідження програми.	9 тиждень	Розділ 5
7	Економіко-організаційна частина.	10 тиждень	Розділ 6
8	Охорона праці та безпека у надзвичайних ситуаціях.	11 тиждень	Розділ 7
9	Оформлення пояснювальної записки та документів до неї.	12–13 тиждень	Додатки
10	Нормоконтроль та рецензування	14–15 тиждень	
11	Захист роботи.	16 тиждень	

Студент (ка) _____ Земляна А.В.
(підпис) (прізвище та ініціали)

Керівник проєкту (роботи) _____ Каплієнко Т.І.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
116 с., 10 табл., 42 рис., 2 дод., 33 джерела.

МЕТОДИ УПРАВЛІННЯ ПРОЄКТАМИ, МОБІЛЬНИЙ ВЕБСАЙТ,
ПРОГРАМНИЙ ЗАСІБ, ПРОГРАМНИЙ ПРОДУКТ, JAVA SCRIPT, MEAN,
IONIC.

Об'єкт дослідження – методи управління проєктами.

Предмет дослідження – програмні засоби для управління проєктами.

Мета роботи – створення програмної системи для спрощення та автоматизації управління проєктами різними методами.

Матеріали, методи та технічні засоби: структурне та об'єктно-орієнтоване програмування, мови програмування Java Script, Tiptoe Script, фреймворк для мобільних застосунків Ionic та стек MEAN.

Результати. Створено програмний мобільний застосунок для управління проєктами різними методами у формі вебсайту.

Висновки. Розроблено програмний мобільний вебсайт для управління проєктами різними методами, який спрощує, прискорює та автоматизує процес управління проєктами.

Галузь використання – мобільний вебсайт може застосовуватися для управління проєктами. Вебсайт розрахован для зручного способу управління колективом користувачів мережі Інтернет для здійснення поставлених цілей і завдань проєкту.

Економічна ефективність. При використанні розробленого мобільного вебсайту прибуток компанії, що його використовує збільшується на 20%.

Термін окупності роботи – 2 місяці.

ABSTRACT

Explanatory note to the qualification thesis of the master: 116 pages, 10 tables, 42 figures, 2 appendixes, 33 sources.

PROJECT MANAGEMENT METHODS, MOBILE WEBSITE, SOFTWARE, SOFTWARE PRODUCT, JAVA SCRIPT, MEAN, IONIC.

The object of research is the methods of project management.

The subject of research is the software for project management.

The purpose of the work is to create the software system to simplify and automate project management by various methods.

The materials, methods and technical means: the structural and object-oriented programming, Java Script programming languages, Type Script, Ionic mobile application framework and MEAN stack.

Results. The software mobile application has been created to manage projects by various methods in the form of a website.

Conclusions. The software mobile website for project management with various methods has been developed, which simplifies, accelerates and automates the project management process.

The scope is the mobile website can be used to manage projects. The website is designed for a convenient way to manage the team of Internet users to achieve the goals and objectives of the project.

Economic efficiency. When using the developed mobile website, the profit of the company that uses it increases by 20%.

The payback period is 2 months.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК	9
ВСТУП	10
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ	11
1.1 Особливості використання методів управління проєктами	11
1.2 Існуючі методи управління проєктами	11
1.2.1 Scrum.....	11
1.2.2 Kanban.....	14
1.2.3 Waterfall	16
1.2 Огляд існуючих методів вирішення завдання	19
1.3 Огляд існуючих програмних засобів, що вирішують аналогічні завдання.....	20
1.3.1 Jira	20
1.3.2 Trello	21
1.3.3 Asana	22
1.4 Постановка завдання роботи	23
1.5 Висновки за розділом 1	24
2 МАТЕРІАЛИ І МЕТОДИ	25
2.1 Вибір мови програмування.....	25
2.2 Огляд особливостей мови програмування.....	27
2.3 Ionic.....	28
2.4 Стек MEAN	29
2.5 Висновки з розділу 2.....	30
3 ОПИС ПРОГРАМИ	31
3.1 Призначення й умови застосування програмного продукту	31
3.1.1 Призначення програмного продукту	31
3.1.2 Функції, що виконує продукт	31

3.1.3 Умови, необхідні для застосування продукту	32
3.2 Характеристики мобільного вебсайту	32
3.3 Висновки за розділом 3	33
4 РОЗРОБКА ПРОГРАМИ.....	34
4.1 Архітектура мобільного вебсайту	34
4.2 Опис роботи з базами даних.....	34
4.2.1 Структура БД.....	35
4.3 Основні розроблені алгоритми.....	38
4.4 Основні рішення щодо модульного уявлення системи	42
4.5 Висновок за розділом 4	44
5 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРЕМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ	45
5.1 Призначення та функції програмного продукту	45
5.2 Умови роботи програмного продукту	46
5.3 Робота мобільного вебсайту	46
5.4 Висновки за розділом 5	62
6 ЕКОНОМІКО - ОРГАНІЗАЦІЙНА ЧАСТИНА.....	63
6.1 Розрахунок собівартості розробки програмної системи для спрощення та автоматизації управління проєктами різними методами	64
6.1.1 Розрахунок основної заробітної платні	64
6.1.2 Розрахунок додаткової заробітної платні	65
6.1.3 Розрахунок єдиного соціального податку	65
6.1.4 Визначення витрат на матеріали	66
6.2 Розрахунок експлуатаційних витрат реалізації проєкту.....	67
6.3 Визначення економічної ефективності проєкту.....	68
6.3.1 Фінансовий план	69
7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ	70

7.1 Аналіз потенційних небезпек	72
7.2 Заходи із забезпечення безпеки.....	73
7.3 Заходи з виробничої санітарії та гігієни праці	75
7.4. Заходи безпеки у надзвичайних ситуаціях	81
7.4.1 Заходи з пожежної безпеки	81
7.4.2 Заходи з цивільного захисту.....	83
ВИСНОВКИ.....	87
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ.....	89
ДОДАТОК А Текст програми.....	90
ДОДАТОК Б Слайди презентації	104

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

MEAN – MongoDB, Express.js, AngularJS, Node.js;

SDK – software development kit;

БД – база даних;

ЕОМ – електронно-обчислювальна машина;

МП – мова програмування;

ПЗ – програмне забезпечення;

ПК – персональний комп'ютер.

ВСТУП

У теперішній час актуальним завданням є управління і організація всього, що потрібно для досягнення мети - вчасно і в рамках бюджету. Будь то розробка нового програмного забезпечення, проведення маркетингової компанії або висадка людини на Марс - проєктне управління дозволяє домогтися успіху у всьому.

Управління проєктами займає достатньо багато часу. Складними є планування, систематизація, відстеження прогресу, чітке розуміння стадії розробки проєкту і т.д. Тому зараз як ніколи є актуальним завдання автоматизування та спрощення системи управління проєктами. Для автоматизації зазначених процесів є доречним завдання розробки програмного забезпечення для управління проєктами.

Наразі не існує ідеальної системи управління проєктами, що підходить для кожного з видів проєктів. Також не існує системи, яка б підходила кожному керівнику і була зручна для всіх членів команди [1].

Метою роботи є створення програмної системи для спрощення та автоматизації управління проєктами різними методами.

Перед розробкою програмного продукту було поставлено такі задачі:

- дослідження методів управління проєктами;
- огляд аналогів (виявлення переваг та недоліків);
- модифікація аналогів;
- розробка програмного продукту;
- проведення тестування та експериментального дослідження;
- розробка дизайну та створення відповідної програмної документації;
- аналіз економіки та організації виробництва щодо розробки програмного забезпечення;
- аналіз стану охорони і умов праці об'єкта проєктування.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Особливості використання методів управління проєктами

Успіх будь-якого серйозного проєкту залежить від використаних для управління ним методів. Але всі проєкти унікальні, і універсальної методики проєкт-менеджменту поки не існує, як не існує і методів, які підходили будь-якому керівникові і будь-якій команді.

Однак за довгий час існування управління проєктами фахівці створили досить багато стандартів і підходів. Всі вони відрізняються один від одного, а також розрізняються по формалізації, самодостатності, деталізації і сфері застосування [2].

1.2 Існуючі методи управління проєктами

Цілком очевидно, що для здійснення проєкту найпростіше розбити його на окремі завдання. А самий тривіальний інструмент з управління проєктом - це так званий чек-лист, де вказані кроки, необхідні для реалізації задумки. Але якщо проєкт складний, то необхідно володіти таким інструментом, який дозволив би відслідковувати витрати для кожного його елемента [2].

1.2.1 Scrum

Scrum – дуже гнучкий метод, визнаний в сімействі Agile найбільш структурованим. Згідно принципам Agile, в Scrum проєкт розбивається на частини, які підходять для миттєвого застосування замовником з метою отримати резерви - доробки продуктів. Згодом отриманим частинам присвоюється свій пріоритет.

Найбільш важливі частини першими відбираються для виконання в спринті (спринти в Scrum – це ітерації тривалістю від 2 до 4 тижнів). За

підсумками спринту замовник отримує робочий інкремент продукту, тобто готові до використання частини. Як тільки один спринт закінчений, проєктна команда починає наступний спринт. Тривалість спринтів завжди однакова, але команда завжди сама встановлює її, оцінюючи свою продуктивність і особливості проєкту.

Спрощена схема роботи по Scrum представлена на рис. 1.1.

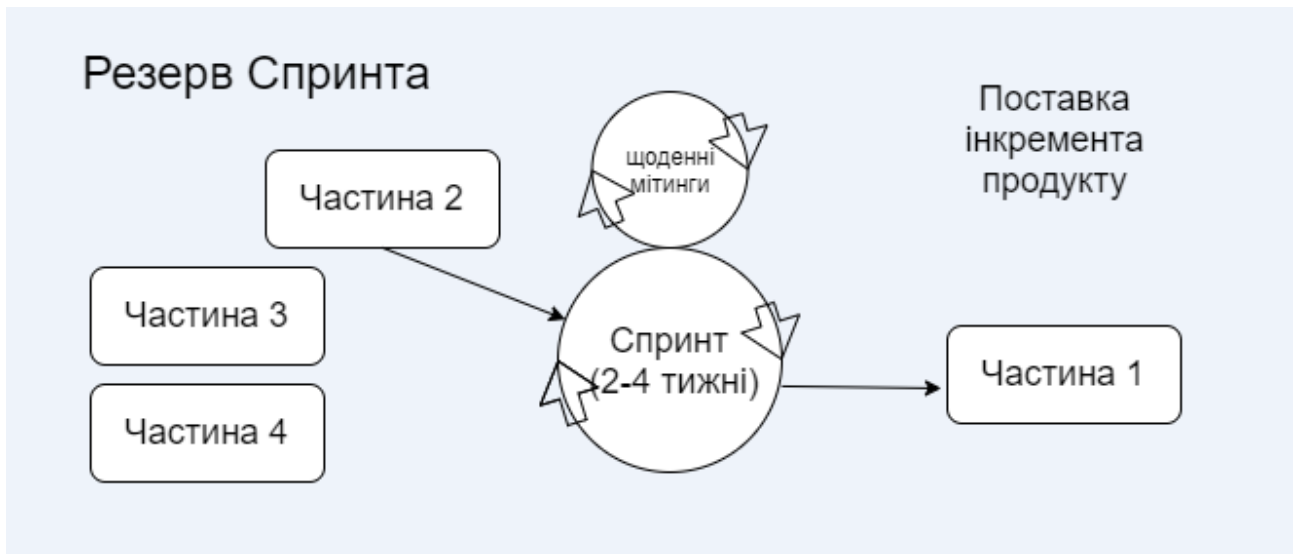


Рисунок 1.1 – Спрощена схема роботи по Scrum

Щоб переконатися у відповідності проєкту вимогам замовника, перед початком будь-якого спринту потрібно виконувати переоцінку ще не виконаного змісту проєкту і вносити в нього зміни.

Участь в цьому процесі приймає керівник проєкту, команда і ініціатор. Відповідальність розподіляється на всіх учасників.

Вся процесуальна структура методу обертається навколо п'яти основних зустрічей, які описані нижче.

Упорядкування резерву. Зустріч нагадує планування. Проводити її потрібно в перший день нового спринту. На зустрічі обговорюється те, що вже вдалося зробити за проєктом і що ще потрібно зробити, і визначаються подальші кроки. Ініціатор ставить завдання, що відповідають новому етапу. Упорядкування резерву визначає результативність нового спринту.

Планування спринту. Після розстановки пріоритетів і визначення завдань ініціатором команда приймає рішення про свої дії протягом наступної ітерації і шукає способи досягнення поставленої мети. Для цього можуть використовуватися найрізноманітніші інструменти планування і оцінки (важливо, щоб вони відповідали принципам методу). Планувати спринт потрібно на самому початку ітерації, але після закінчення зустрічі з упорядкуванням резерву.

Щоденні мітінги. Щоденні мітінги - це щоденні зустрічі, що проводяться в один і той же час (на них витрачається, як правило, до 15 хвилин). Члени команди діляться інформацією про статус своєї роботи і стан проєкту, проте ніякі рішення не приймаються і проблеми не обговорюються (для цього виділяється окремий час і влаштовуються додаткові зустрічі). Щоденні мітінги потрібні лише для обміну відомостями.

Підведення підсумків спринту. На цьому етапі досліджується і адаптується створений продукт. Члени команди діляться своїми результатами з усіма зацікавленими особами. Головним завданням тут є упевнитися в тому, що продукт спринту відповідає цілям проєкту і очікуванням учасників проєкту.

Ретроспектива спринту. Етап, який проводиться відразу ж після попереднього, але до того як почне плануватися новий спринт. Команда визначає ступінь чіткості і злагодженості пройденого етапу, досліджує проблеми що з'явилися в методології, роботі та взаємодії. Завдяки ретроспективі команда може зробити висновки і підвищити ефективність наступного спринту.

Переваги Scrum:

- підходить для проєктів, які потребують швидких результатів;
- легко адаптується до змін;
- підходить для застосування командами, де є співробітники з невеликим досвідом роботи в області реалізації конкретного проєкту, тому що всі члени команди активно взаємодіють один з одним;

- дозволяє здійснювати «швидкі помилки», тобто отримувати практично миттєвий зворотний зв'язок від виконуваних дій завдяки спринті;
- дозволяє швидко виправляти помилки і підвищувати ефективність роботи по реалізації проєкту.

Недоліки Scrum:

- висока вимогливість до проєктної команди (потрібно, щоб в команді було від 5 до 9 людей, і всі члени команди повинні володіти відразу декількома компетенціями, необхідними для реалізації проєкту, завдяки чому співробітники можуть доповнювати і замінювати один одного, а робота ніколи не буде стояти на місці);
- всі співробітники повинні вміти і хотіти працювати в команді, бути здатні до самоорганізації і активно брати на себе відповідальність;
- підходить не для всіх організацій і команд, тому що схема роботи за методом підходить для розробки далеко не кожного продукту (наприклад, побудувати будинок або створити промисловий верстат по методу Scrum буде неможливо).

Багато хто вважає, що метод Scrum складний для впровадження, вказуючи на необхідність в делегуванні повноважень, незвичайну організаційну структуру і т.п., однак при вмілому використанні і завдяки своїй гнучкості метод не дозволить роботі піти в неправильному напрямку [2].

1.2.2 Kanban

Метод Kanban передбачає передачу інкремента продукту від етапу до етапу, в результаті чого на виході з'являється готовий продукт.

Не менш важливо і те, що Kanban дозволяє призупинити виконання одного завдання на будь-якому етапі в разі, якщо з'явилися інші термінові завдання або змінився пріоритет поточної.

Незавершена редакція, підвішені дати, невизначена частина функції - для методу Kanban це норма.

Спрощена схема роботи по Kanban представлена на рис. 1.2.



Рисунок 1.2 – Спрощена схема роботи по Kanban

Відмінність цього методу від Scrum полягає в меншій суворості: немає обмежених за часом спринтів і регламентованих зустрічей, відсутні ролі членів команди і учасників (крім ініціатора проєкту). Також в Kanban один член команди може займатися виконанням декількох завдань в один і той же час.

Щоб працювати з методом, потрібно визначитися з етапами потоку операцій. Тут вони представлені у вигляді стовпців, а завдання позначаються спеціальними картками. Картки переміщуються по Етапах, на кожному з яких підвищується відсоток завершення. В результаті виходить готовий елемент продукту. Дошку ж за допомогою стовпців можна уявити і в фізичному вигляді, і в електронному, тому що і тут ніяких обмежень метод не накладає.

Створена індивідуально система Kanban може володіти такою гнучкістю, яка потрібна. Але є все ж кілька основ, на яких ґрунтується вся ця система. Вони представлені нижче.

Картки. Повинні створюватися для всіх завдань окремо. У картках фіксується вся потрібна інформація про поточне завдання. Це дуже зручно, тому що до картки можна звернутися в будь-який час.

Обмежена кількість завдань на одному етапі. Кількість карток для одного етапу завжди регламентовано. Це дозволяє відстежувати виникнення «заторів» в потоці операцій і відразу ж його усувати.

Постійний потік. Завдання резерву надходять в потік, виходячи з їх пріоритетності. Це і створює безперервний робочий процес.

Безперервне покращення. Тут основою служить японська концепція постійного покращення Kanban, суть якої полягає в перманентному аналізі робочого процесу і пошуку шляхів підвищення його ефективності.

Переваги Kanban:

- ідеально підходить для застосування згуртованими командами з налагодженою комунікацією;
- відсутні дедлайни;
- істотно заощаджує ресурси і дозволяє дотримуватися бюджет і терміни, тому що передбачає точний розрахунок навантаження на виконавців, правильну розстановку обмежень і фокусування на безперервному поліпшенні.

Недоліки Kanban:

- підходить більшою мірою для команд, члени яких мають пересічні один з одним навичками, інакше ефективність методу істотно знизиться;
- не зовсім підходить для реалізації проєктів з жорсткими дедлайнами;
- у випадках, коли доводиться мати справу з чітко встановленими термінами, краще всього використовувати метод Scrum.

1.2.3 Waterfall

Waterfall або каскадна модель - модель процесу розробки програмного забезпечення, в якій процес розробки виглядає як потік, що послідовно проходить фази аналізу вимог, проєктування, реалізації, тестування, інтеграції та підтримки.

Дотримуючись каскадної моделі, розробник переходить від однієї стадії до іншої строго послідовно. Спочатку повністю завершується етап «визначення вимог», в результаті чого виходить список вимог до програмного забезпечення (ПЗ). Після того як вимоги повністю визначені, відбувається перехід до проектування, в ході якого створюються документи, що детально описують для програмістів спосіб і план реалізації зазначених вимог. Після того як проектування повністю виконане, програмістами виконується реалізація отриманого проєкту. На наступній стадії процесу відбувається інтеграція окремих компонентів, що розробляються різними командами програмістів. Після того як реалізація і інтеграція завершені, проводиться тестування і налагодження продукту; на цій стадії усуваються всі недоліки, що з'явилися на попередніх стадіях розробки. Після цього програмний продукт впроваджується і забезпечується його підтримка - внесення нової функціональності і усунення помилок.

Спрощена схема роботи по Waterfall представлена на рис. 1.3.

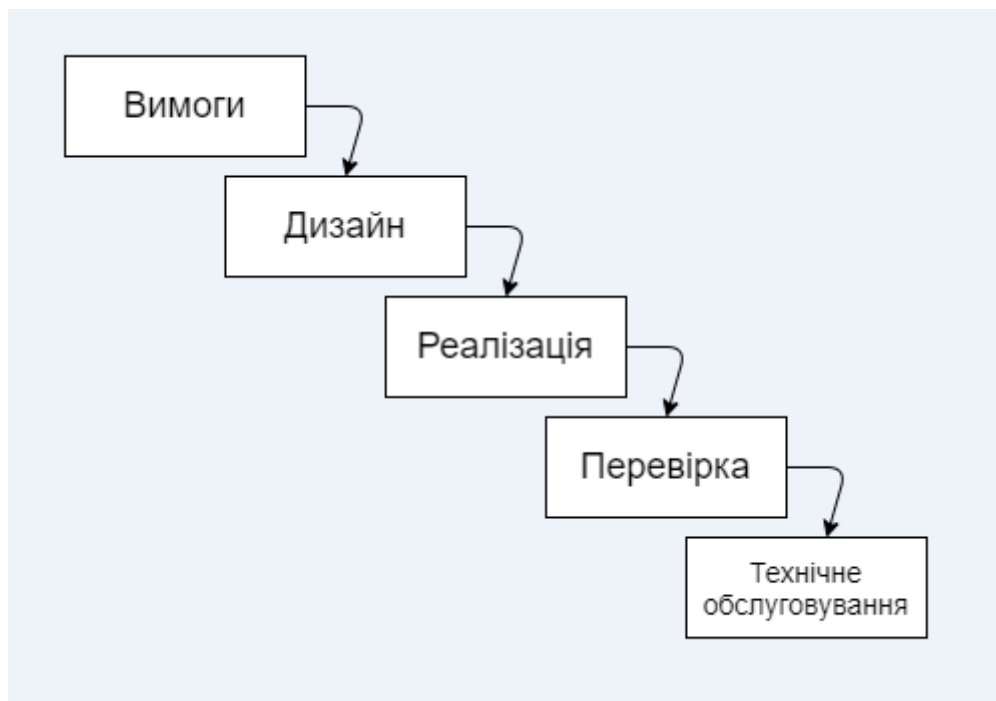


Рисунок 1.3 – Спрощена схема роботи по Waterfall

Для класичної моделі розробки програмного забезпечення виділяють наступні етапи:

- аналіз вимог проєкту – визначаються програмні вимоги для інформаційної предметної області системи;
- проєктування – розробляється і формулюється логічно послідовна технічна характеристика програмної системи, виконується деталізація системи;
- реалізація ПЗ - втілення повноцінного проєкту;
- тестування продукції – тестова експлуатація продукту;
- інтеграція системи – включає установку і офіційну приймання продукту;
- підтримка - надання технічної допомоги по продукту після запуску а комерційну експлуатацію.

Ця модель має на увазі строго послідовне і одноразове виконання кожної фази проєкту. Перехід від однієї фази до іншої можливий тільки після успішного завершення попереднього етапу. Кожен етап має на увазі детальне планування і повну коректність результату етапу.

Такі жорсткі обмеження послідовності дозволяють побудувати процес розробки, який максимально прозорий і зручний для замовника.

Недоліки Watrefall:

- watrefall проєкт повинен постійно мати актуальну документацію, обов'язкова актуалізація проєктної документації, надлишкова документація;
- дуже не гнучка методологія;
- може створити помилкове враження про роботу над проєктом (наприклад фраза «45% виконано» не несе за собою ніякої корисної інформації, а є всього лише інструментів для менеджера проєкту);
- у замовника немає можливості ознайомитися з системою заздалегідь і навіть з «Пілотом» системи;
- у користувача немає можливості звикати до продукту поступово;
- всі вимоги повинні бути відомі на початку життєвого циклу проєкту;

- виникає необхідність в жорсткому управлінні і регулярному контролі, інакше проєкт швидко вийде з графіків;
- відсутня можливість врахувати переробку, весь проєкт робиться за один раз.

Переваги Watrefall:

- висока прозорість розробки та фаз проєкту;
- чітка послідовність;
- стабільність вимог;
- строгий контроль менеджменту проєкту;
- полегшує роботу зі складання плану проєкту та збору команди проєкту;
- добре визначає процедуру по контролю якості.

Тим самим, каскадна модель має на увазі, що перехід від однієї фази розробки до іншої відбувається тільки після повного і успішного завершення попередньої фази, і що переходів назад або вперед або перекриття фаз - не відбувається [3].

1.2 Огляд існуючих методів вирішення завдання

Система управління проєктами – комплексне ПЗ, що включає в себе програми для планування завдань, складання розкладу, контролю ціни і управління бюджетом, розподілу ресурсів, спільної роботи, спілкування, швидкого управління, документування та адміністрування системи, яке використовуються спільно для управління великими проєктами [4].

Системи управління проєктами можуть бути як електронними, так і на папері. Для використання системи у електронному виді необхідний мобільний пристрій та інтернет. Цей вид системи управління проєктами зручний та актуальний на відміну від паперового. Він значно заощаджує час та має гнучку структуру.

1.3 Огляд існуючих програмних засобів, що вирішують аналогічні завдання

Для огляду було взято три найпопулярніших програмних засоби, що вирішують аналогічні завдання, а саме:

- jira;
- trello;
- asana.

1.3.1 Jira

Jira – програмний продукт, створений спеціально, щоб керувати проєктами. Має велику популярність серед багатьох великих і відомих компаній. Jira входить в трійку кращих світових рішень з управління проєктами з 9.4 балами задоволеності користувачів.

Jira призначена для створення завдань, призначення виконавців, установки пріоритетів і відстеження виконання цих завдань. Весь процес життєвого циклу застосунку контролюється за допомогою Jira. Використання такого ПЗ гарантує, що вся інформація буде збережена від початку і до кінця, а простий і зрозумілий інтерфейс спрощує взаємодію, завдяки чому робота відбувається ефективніше.

Дуже зручне рішення для команд, які займаються розробкою програмного застосунка. З його допомогою зручно планувати, ставити і контролювати завдання. Весь життєвий цикл розробки ПЗ можна відстежити, зайшовши в Jira.

Ідея створення Jira була в тому, щоб створити універсальний, гнучкий інструмент, що надає компаніям будь-якого масштабу, можливість створити необхідні системи. Jira дотримується гібридного підходу Scrumban. Це об'єднання підходів Scrum і Kanban, для більш гнучкого підстроювання до

кожного окремого сценарію. За допомогою Jira, всі поля, проблеми, помилки, типи даних отримали свої власні настройки, які можна підлаштувати під себе, при цьому при необхідності можна створювати власні поля, створювати правила для бізнес-процесів і т.д. Завдяки цьому, налаштування і взаємодія з завданнями стає простіше і витрачає менше часу (рис. 1.4) [5].

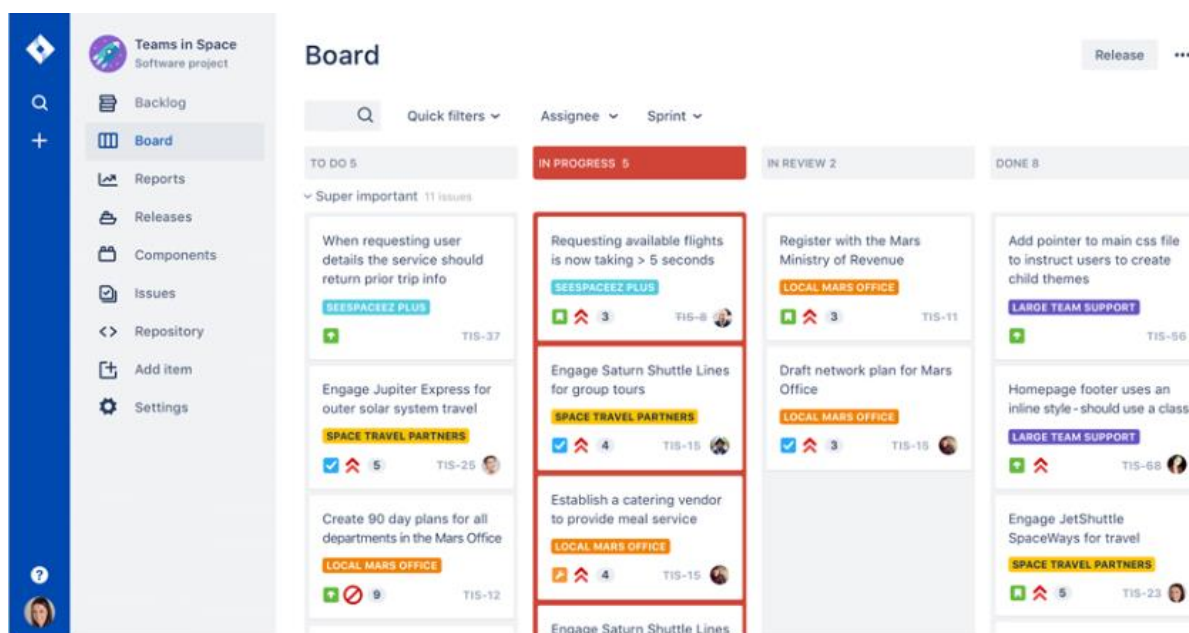


Рисунок 1.4 – Jira [5]

1.3.2 Trello

Trello – сервіс для управління проєктами та персональними завданнями. Простий, безкоштовний і гнучкий візуальний онлайн-інструмент для організації роботи команди або відділу по методології Kanban.

Trello являє собою інструмент для управління проєктами в різних нішах – комерційними, творчими, соціальними. Також сервіс і застосунок зручні для особистого планування і самоконтролю.

Для ведення промислових і інфраструктурних проєктів Trello не підійде, так само як і не вийде повноцінно організувати з ним роботу цілої компанії.

Але коли мова йде про маленькі проєкти або процеси, які не потребують спеціальних галузевих інструментів, Trello майже немає рівних.

Варто відзначити цінність трекера для управління циклами розробки. Особливо він відповідає гнучкими методологіями управління проєктами - зокрема, Scrum, Kanban.

Максимальна ефективність інструменту розкривається при командній роботі, та й в індивідуальній активності продуктивність помітно зростає. Ідеї та завдання швидко упорядковуються, структуруються, все стає наочним, набагато легше розставляти пріоритети (рис. 1.5) [6].

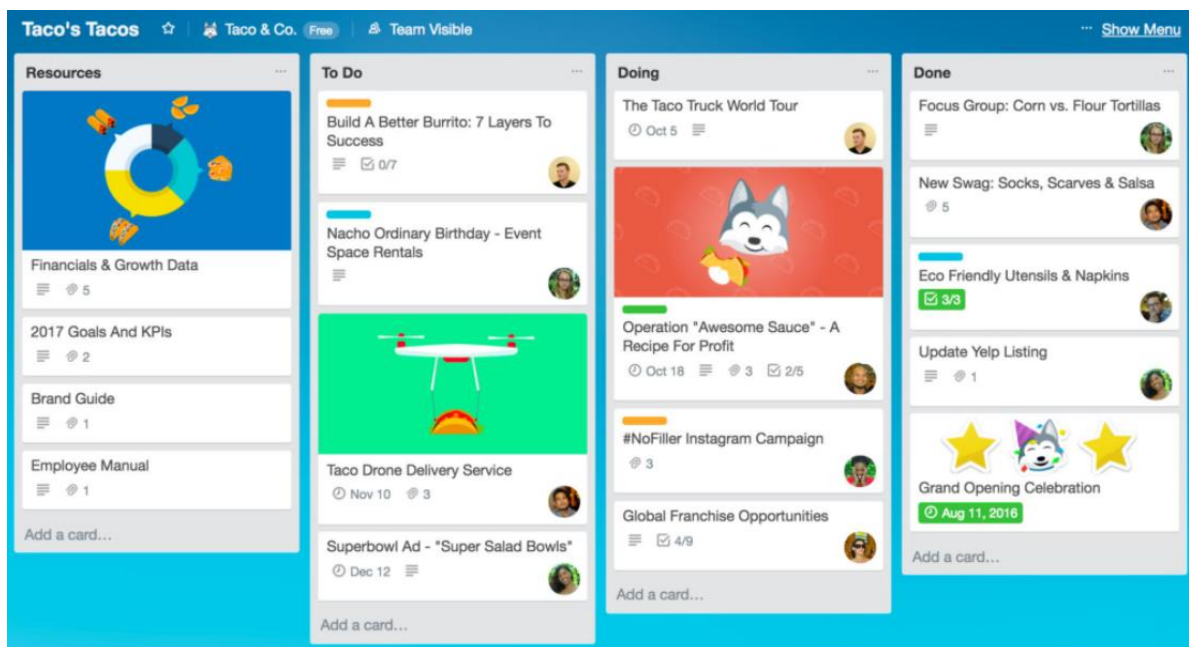


Рисунок 1.5 – Trello [6]

1.3.3 Asana

Asana – застосунок для невеликих компаній і індивідуального використання, що володіє всім необхідним набором можливостей для успішного ведення невеликих проєктів.

Концепція роботи з застосунком Asana проста і нагадує більшість систем подібного призначення. Кожна команда може створити для себе

зручний робочий простір (workspace). Кожен робочий простір може включати в себе безліч проєктів, а кожен проєкт, в свою чергу, безліч завдань. Користувачі, що мають на те повноваження, можуть доповнювати завдання, додавати коментарі і теги, прикріплювати до задачі файли. Також Asana дає користувачам можливість підписатися на цікаві для нього завдання. У разі зміни або закриття такого завдання всі підписники моментально отримують відповідне повідомлення (рис. 1.6) [7].

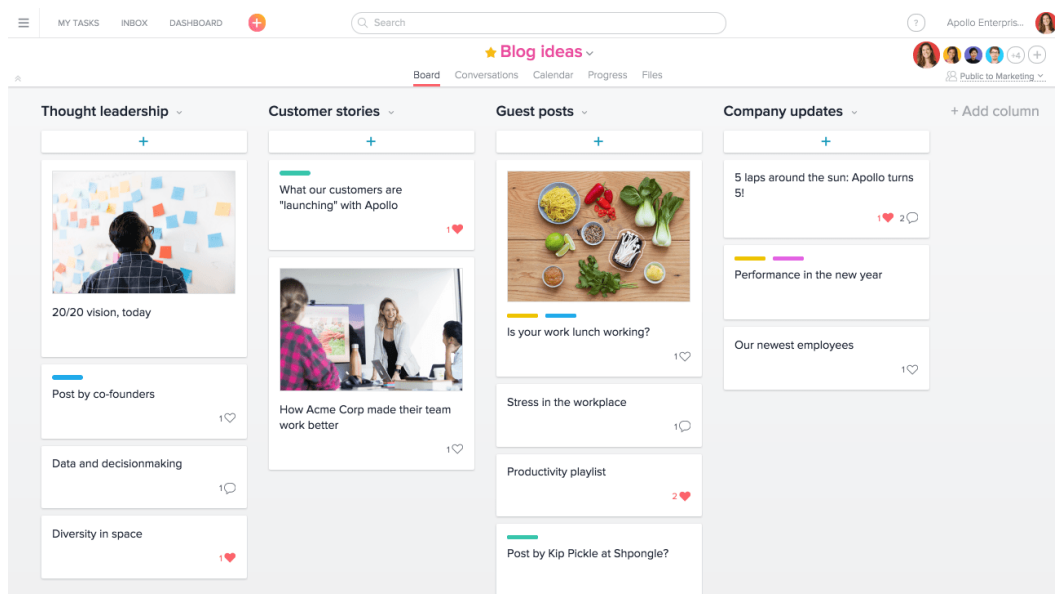


Рисунок 1.6 – Asana [7]

1.4 Постановка завдання роботи

Розглянувши «лідерів» існуючих програмних засобів, що відповідають тематиці дипломної роботи, був виділений наступний основний функціонал, що має бути реалізований, а саме:

- реєстрація та авторизація у мобільному застосунку;
- можливість обирати метод управління проєктом;
- додавання користувача до дошки чи картки;
- додавання та редагування дошок, колонок та карток;
- пересування карток між колонками;

- оцінка задачі та врахування часу, витраченого на задачу;
- отримання повідомлень про зміну на дошці чи у картці, на яку доданий користувач;
- можливість залишати замітки у картці та відзначати їх.

1.5 Висновки за розділом 1

В даному розділі досліджені існуючі методи вирішення завдання, розглянуто та зроблено порівняння існуючих програмних засобів, що вирішують аналогічні завдання. Відповідно до цих даних та вказівок викладача було поставлено завдання до роботи.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

В першу чергу було розглянуто співвідношення попит / пропозиція серед найбільш популярних мов програмування (МП) (рис. 2.1).

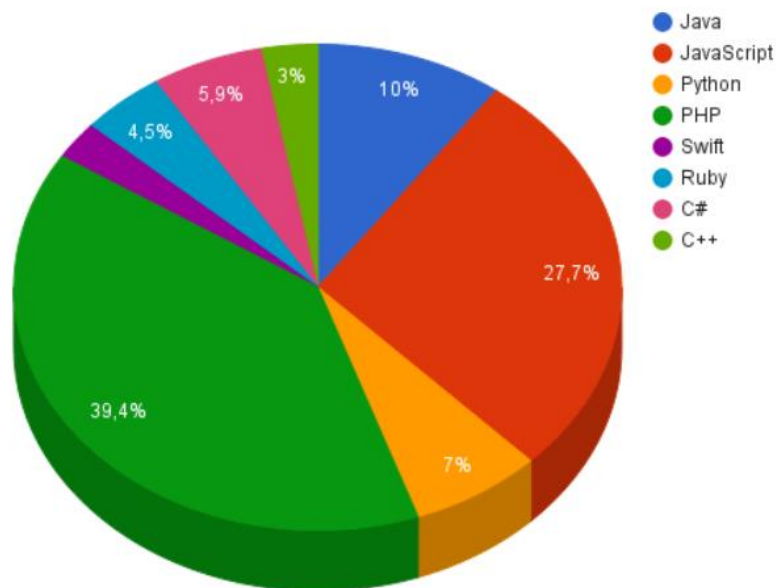


Рисунок 2.1 – Процентне співвідношення замовлень серед популярних МП

Виходячи з діаграми найпопулярнішими МП є PHP та Java Script.

Нижче в таблиці 2.1 розглянуті та порівняні основні МП.

Таблиця 2.1 – Порівняння МП

МП	Доступність	Затребуваність	Спектр застосувань	Цілі
Python	Освоюється легко	1 місце	AI, ігри, Big Data, веб, mobile та інше.	Сайти і вебзастосунки, робота з даними

Продовження таблиці 2.1

МП	Доступність	Затребуваність	Спектр застосувань	Цілі
Java Script	Освоюється легко	2 місце	Node.js перетворює JavaScript в мову загального призначення. JS це і фронтенд, і бекенд. Підходить і в роботі тестувальника, QA-інженера.	Сайти і вебзастосунки
Swift	Освоюється трохи складніше	7 місце	Створення застосунків під Mac і iOS	Застосунки для Apple
Ruby	Складний	6 місце	Використовується в веброзробці	Сайти і вебзастосунки
PHP	Освоюється легко	5 місце	Популярна мова програмування для бекенд, на ньому пишуть серверну частину сайту.	Сайти і вебзастосунки
C/C++	Складний	4 місце	C / C ++ має широкий спектр застосувань. Зі знанням цих мов буде легко вчити щось нове.	Розібратися в програмуванні, робототехніка

Продовження таблиці 2.1

МП	Доступність	Затребуваність	Спектр застосувань	Цілі
Java	Освоюється трохи складніше	3 місце	Широке застосування: Android-застосунки, серверні застосунки, автоматизація тестування, веб, Big Data, веб-застосунки та інше.	Робота з даними, застосунки для Android, вебзастосунки

Беручи до уваги розглянуті мови програмування, їх переваги, недоліки та спектр застосувань, було обрано Java Script для розробки програмного продукту.

2.2 Огляд особливостей мови програмування

Для розробки мобільного вебсайту було обрано такі мови програмування:

- JavaScript;
- TypeScript.

JavaScript – це мультипарадигмова мова програмування. Підтримує об'єктно-орієнтована, імперативний і функціональний стилі.

JavaScript зазвичай використовується як вбудована мова для програмного доступу до об'єктів застосунків. Найбільш широке застосування знаходить в браузерах як мова сценаріїв для додання інтерактивності вебсторінок [9].

TypeScript є строго типізованим надмножиною JavaScript, що означає, що він додає деякі синтаксичні переваги для мови, але при цьому дозволяє писати звичайний JavaScript. Він заохочує більш декларативний стиль програмування через такі речі, як інтерфейси і статичну типізацію, пропонує модулі і класи, а найголовніше, досить добре інтегрується з популярними бібліотеками та кодом JavaScript.

Також при розробці програмного продукту використовувалися HTML і CSS.

HTML - це мова розмітки, і використовується, щоб повідомляти браузеру, як відображати вебсторінки. HTML складається з ряду елементів, які використовуються, щоб вкладати або обертати різні частини контенту, щоб змусити контент відображатися або діяти певним чином. Теги можуть зробити слово або зображення посиланням на щось ще, можуть зробити слова курсивом, зробити шрифт більше або менше і так далі. Отриманий в результаті інтерпретації форматований текст відображається на екрані монітора комп'ютера або мобільного пристрою [10].

CSS - це мова, яка відповідає за візуальне уявлення документів користувачеві.

Призначення CSS - відокремлювати те, що задає зовнішній вигляд сторінки, від її змісту. Якщо документ створено тільки з використанням HTML, то в ньому визначається не тільки кожен елемент, але і спосіб його відображення (колір, шрифт, положення блоку і т. д.). Якщо ж підключені каскадні таблиці стилів, то HTML описує тільки черговість об'єктів. А за всі їх властивості відповідає CSS [11].

2.3 Ionic

Ionic – це SDK для створення гібридних мобільних застосунків, набір CSS і JS компонент, створений на основі AngularJS, SASS, Apache Cordova.

Іоніс побудований на базі AngularJS і забезпечує повне рішення для проектування, побудови та упаковки мобільних застосунків. Дизайн виконується за допомогою набору інструментів шаблонів і призначеної для користувача бібліотеки значків. Іоніс надає призначені для користувача компоненти CSS / SASS, а також розширення Javascript UI. Іонні застосунки можуть бути побудовані, емулювати і упаковані за допомогою інтерфейсу командного рядка (CLI).

Іонні шаблони є динамічними і адаптуються до їх середовищі, щоб забезпечити природний «зовнішній вигляд». Ця адаптація включає в себе макет, стиль і значки. Іоніс також дозволяє налаштовувати незалежну платформу. Оскільки Іоніс-програма така сама технологію вебінтерфейсу, вони також можуть бути переглянуті в браузері для більш швидкого розвитку.

Під Іоніс вже створено чималу кількість всіляких плагінів, які можна легко підключати в залежності від поставлених завдань. Область використання даної технології просто величезна, оскільки з її допомогою розробляються гібридні програми будь-якого рівня складності, причому досить швидко [12].

2.4 Стек MEAN

MEAN – набір технологій на базі JavaScript, призначених для розробки вебзастосунків. Це абревіатура, під якою об'єднані три фреймворка JavaScript і документоорієнтовані NoSQL - бази даних.

М – MongoDB – це вільна документоорієнтована база даних, організована таким чином, щоб забезпечити як масштабованість, так і гнучкість при розробці. Дані в MongoDB записуються не в таблицях і стовбцях, як в реляційної бази даних. Замість цього в MongoDB зберігаються JSON - подібні документи з динамічними схемами.

Е – Express.js – це фреймворк для сервера застосунків Node.js, призначений для створення односторінкових, багатосторінкових і гібридних вебзастосунків. Він є стандартним серверним фреймворком для node.js.

A – AngularJS – це структурний фреймворк для динамічних вебзастосунків. У ньому можна використовувати HTML в якості мови шаблонів, а також розширювати синтаксис HTML для чіткого і лаконічного опису компонентів застосування. Зв'язування даних і впровадження залежностей в Angular дозволяють позбутися від маси коду, який довелося б писати в іншому випадку.

N – Node.js – це вільне багатоплатформне виконуюче середовище для розробки серверних вебзастосунків. Застосунки Node.js пишуться на мові JavaScript і можуть запускатися в виконуючому середовищі Node.js в операційних системах OS X, Microsoft Windows, Linux, FreeBSD, NonStop, IBM AIX, IBM System z і IBM (рис. 2.1) [13].

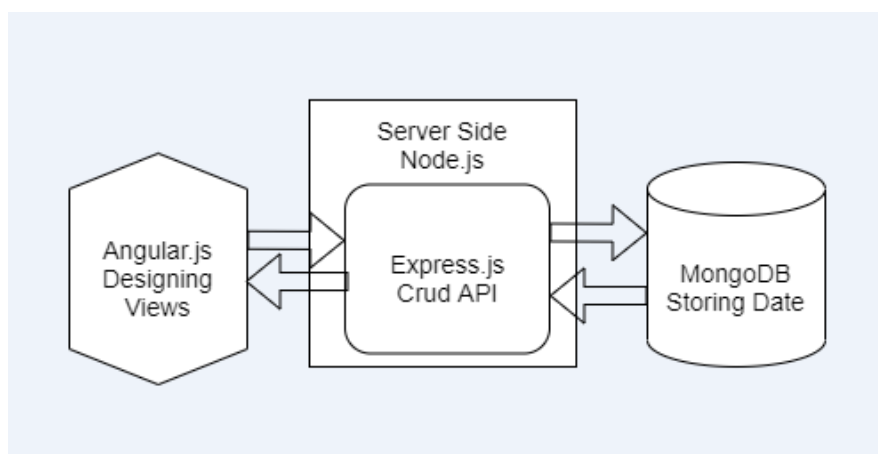


Рисунок 2.1 – Стек MEAN

2.5 Висновки з розділу 2

Для розробки мобільного вебсайту мовами програмування було обрано JavaScript, TypeScript, HTML та CSS за зручність, потужність та гнучкість мови, широкий спектр можливостей.

Також у розробці використовувався SDK Ionic та набір серверного програмного забезпечення MEAN.

3 ОПИС ПРОГРАМИ

3.1 Призначення й умови застосування програмного продукту

3.1.1 Призначення програмного продукту

Мобільний вебсайт призначений для швидкого, зручного керування проєктами в режимі онлайн, використовуючи різні методології управління проєктами.

Вебсайтом може користуватися людина з мінімальними навичками володіння мобільним пристроєм, сайт легко впровадити в робочий процес без довгої адаптації з боку персоналу.

3.1.2 Функції, що виконує продукт

Основні функції, що має виконувати програмна система для спрощення та автоматизації управління проєктами різними методами:

- авторизація у системі;
- створення дошок за певним методом управління проєктами (Scrum, Kanban, Waterflow, «Вільна дошка»);
- дошка створена за обраним методом має відповідати правилам одному з методів управління проєктами (Scrum, Kanban, Waterflow, «Вільна дошка»);
- створювання колонок та карток;
- переміщення карток між колонками;
- додавання користувачів до потрібної дошки або картки;
- оцінка та врахування часу у картці;
- можливість залишати коментарі у картці;
- можливість ставити пріоритет картці;
- отримання повідомлень про дії на дошці, до якої доданий користувач;
- перегляд історії картки.

3.1.3 Умови, необхідні для застосування продукту

Умови, необхідні для застосування продукту:

- мобільний девайс з підтримкою інтернет з'єднання;
- об'єм оперативної пам'яті – для виконання своїх функцій, мобільному вебсайту достатньо 500 Мб оперативної пам'яті, але, виходячи з того, що для функціонування операційної системи необхідно мінімум 32 Мб оперативної пам'яті, то рекомендується використовувати вебсайт на девайсі, що має оперативної пам'яті більше 1 Гб;
- вимоги до програмного забезпечення – системні програмні засоби, які використовуються вебсайтом, повинні бути представлені локальною версією ОС девайсу;
- вимоги до інтернету – для роботи з вебсайтом потрібно мати інтернет-підключення зі швидкістю не менше ніж 4 Мбіт/с;
- вимоги до персоналу (програміста) – програміст повинен мати мінімум середню технічну освіту і повинен бути атестованим мінімум на II кваліфікаційну групу по електробезпеці (для роботи з конторським обладнанням).

В перелік задач, виконуваних програмістом, повинні входити: задача підтримки працездатності технічних засобів; задача підтримки працездатності системних програмних засобів; задача підтримки працездатності мобільного вебсайту.

3.2 Характеристики мобільного вебсайту

Мобільний вебсайт може працювати цілодобово та безперервно. Сайт дає достатньо швидку відповідь, завантаження сторінок. Має користувацький інтерфейс. Вебсайт автоматично зберігає зміни, які були зроблені під час роботи.

Програмна система для спрощення та автоматизації управління проектами різними методами розрахована для користування бізнес компаніями або компаніями у сфері інформаційних технологій.

3.3 Висновки за розділом 3

У розділі 3 було сформовано основні рішення щодо розробки програмного продукту, а саме були виявлені основні характеристики, які матиме розроблене програмного забезпечення. Отримані вимоги необхідні для застосування продукту. А також сформовані та перераховані функції, які має виконувати продукт для досягнення поставленої мети.

4 РОЗРОБКА ПРОГРАМИ

4.1 Архітектура мобільного вебсайту

Відповідно до аналізу технічного завдання дипломного проєкту і з урахуванням обраного ПЗ, архітектура розробленої програмної системи для спрощення та автоматизації управління проєктами різними методами має вигляд, представлений на рисунку 4.1.

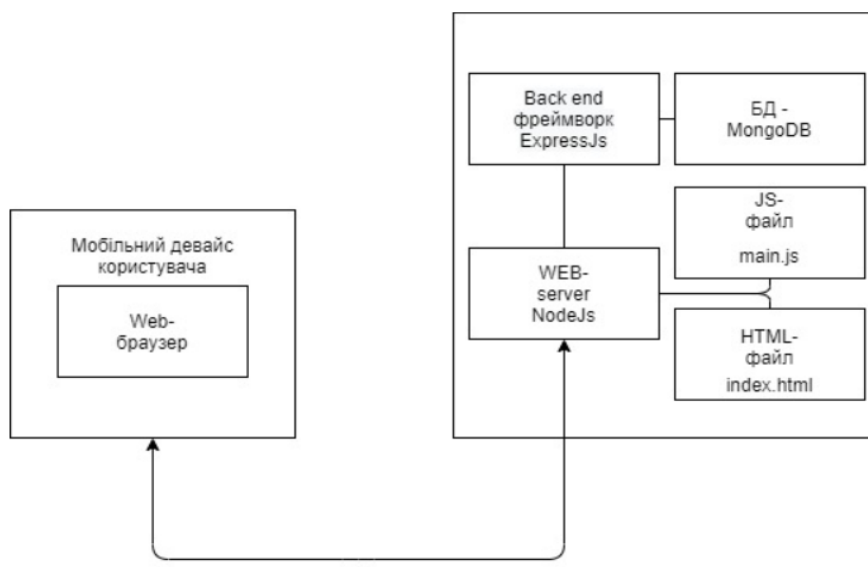


Рисунок 4.1 – Архітектура розробленої програмної системи для спрощення та автоматизації управління проєктами різними методами

4.2 Опис роботи з базами даних

При розробці мобільного вебсайту використовувалась MongoDB – документо - орієнтована система керування базами даних.

База даних (БД) була використана для створення необхідних полів для роботи мобільного вебсайту.

Було побудовано БД, що містить наступну інформацію:

- інформація про користувача – повне ім'я, електронна адреса, зашифрований пароль, дата створення користувача, фото користувача;

– інформація про дошку – об’єкт власника дошки, прив’язані до дошки користувачі, метод керування проектом, список колонок, ім’я дошки, дата створення;

– інформація про колонку – об’єкт дошка, до якої належить колонка, об’єкт власника дошки, список карток, ім’я колонки, дата створення;

– інформація про картку – об’єкт колонка, до якої належить картка, об’єкт власника картки, прив’язані до дошки користувачі, список заміток, тип пріоритету, ім’я картки, опис картки, позиція картки, врахований час, оцінка картки, історія картки, дата створення;

– інформація про замітку – об’єкт картки, до якої належить замітка, об’єкт власника замітки, список користувачів, яким сподобалась замітка, ім’я, дата створення.

4.2.1 Структура БД

У таблицях 4.1 – 4.5 наведено опис всіх таблиць БД, розміщених в MongoDB.

Таблиця 4.1 – Опис таблиці Користувач

Назва	Ідентифікатор	Тип поля	Тип зв’язку	Обов’язкове чи ні
Повне ім’я	Full name	String		Обов’язкове
Електронна адреса	Email	String		Обов’язкове
Зашифрований пароль	HashedPassword	Sring		Обов’язкове
Дата створення користувача	CreatedAt	Date		Обов’язкове
Фото користувача	Avatar	String		Необов’язкове

Таблиця 4.2 – Опис таблиці Дошка

Назва	Ідентифікатор	Тип поля	Тип зв'язку	Обов'язкове чи ні
Об'єкт власника дошки	Owner	User	M:1	Обов'язкове
Прив'язані до дошки користувачі	Users	Array of users	M:M	Обов'язкове
Метод керування проектом	Type	String		Обов'язкове
Список колонок	Columns	Array of columns	1:M	Необов'язкове
Ім'я дошки	Name	String		Обов'язкове
Дата створення	Date	Date		Обов'язкове

Таблиця 4.3 – Опис таблиці Колонка

Назва	Ідентифікатор	Тип поля	Тип зв'язку	Обов'язкове чи ні
Об'єкт дошка, до якої належить колонка	Board	Board	M:1	Обов'язкове
Об'єкт власника дошки	Owner	User	1:1	Обов'язкове
Список карток	Cards	Array of cards	1:M	Необов'язкове
Ім'я колонки	Name	String		Обов'язкове
Дата створення	Date	Date		Обов'язкове

Таблиця 4.4 – Опис таблиці Картка

Назва	Ідентифікатор	Тип поля	Тип зв'язку	Обов'язкове чи ні
Об'єкт колонка, до якої належить картка	Column	Column	M:1	Обов'язкове
Об'єкт власника картки	Owner	User	1:1	Обов'язкове
Дата створення	Date	Date		Обов'язкове
Прив'язані до дошки користувачі	Users	Array of users	M:M	Обов'язкове
Список заміток	Notes	Array of notes	1:M	Необов'язкове
Тип пріоритету	Priority	String		Необов'язкове
Ім'я картки	Name	String		Обов'язкове
Опис картки	Description	String		Необов'язкове
Позиція картки	Position	Number		Обов'язкове
Врахований час	LoggedTime	Array of loggedTime		Необов'язкове
Оцінка картки	EstimateTime	Array of estimateTime		Необов'язкове
Історія картки	Actions	Array of actions		Обов'язкове
Дата створення	Date	Date		Обов'язкове

Таблиця 4.5 – Опис таблиці Замітка

Назва	Ідентифікатор	Тип поля	Тип зв'язку	Обов'язкове чи ні
Об'єкт картки, до якої належить замітка	Card	Card	М:М	Обов'язкове
Об'єкт власника замітки	Owner	User	1:1	Обов'язкове
Список користувачів, яким сподобалась замітка	Likes	Array of users	М:М	Необов'язкове
Ім'я	Name	String		Обов'язкове
Дата створення	Date	Date		Обов'язкове

4.3 Основні розроблені алгоритми

Під час розробки програмної системи для спрощення та автоматизації управління проектами різними методами було реалізовано алгоритми, які відповідають тим чи іншим функціональним частинам програми, що допомагає організувати досить швидку реалізацію мобільного вебсайту. А саме:

- авторизація користувачів;
- створення дошок методом Scrum;
- створення дошок методом Kanban;
- створення дошок методом Waterflow;
- створення дошок вільним методом;
- створення колонок та карток;
- отримання дошок, колонок та карток з усіма деталями;
- редагування колонок та карток;

- видалення дошок, колонок та карток;
- сортування карток;
- та інші.

Опис основних алгоритмів представлений у таблиці 4.6.

Таблиця 4.6 – Основні розроблені алгоритми

Алгоритм	Кроки
Створення дошок методом Scrum	1. Користува має увійти у систему. 2. На екрані Дошки користувач повинен натиснути кнопку «+» для створення дошки. 3. Користувач переходить на екран, де необхідно обрати метод управління проектом «Scrum». 4. Після вибору методу проекту, користувач має ввести ім'я дошки та додати користувачів до дошки, якщо це необхідно. 5. Після того як користувач натискає на кнопку «Створити дошку», йде запит на серверну частину для створення даних для дошки методом Kanban за адресою: – дошки - /api/boards; – колонки - /api/boards/id_дошки/columns; – картки - /api/columns/id_колонки/cards. 6. Після цього відбувається створення даних у базі даних у таблицях: – Boards - для дошок; – Columns - для колонок; – Cards - для карток; 7. Далі відправка оновлених даних на інтерфейсну частину.

Продовження таблиці 4.6

Алгоритм	Кроки
	8. Виведення даних на екран користувача. При цьому, перед тим як створити дошку, користувач створює спринт. Після вибору спринта створюються такі колонки: «Резерв», «До роботи», «У роботі», «Заблоковано», «Тест», «Виконано». Користувач не може створювати нові колонки.
Створення дошок методом Kanban	1. Користувач має увійти у систему. 2. На екрані Дошки користувач повинен натиснути кнопку «+» для створення дошки. 3. Користувач переходить на екран, де необхідно обрати метод управління проєктом «Kanban». 4. Після вибору методу проєкта, користувач має ввести ім'я дошки та додати користувачів до дошки якщо це необхідно. 5. Після того як користувач натискає на кнопку «Створити дошку», йде запит на серверну частину для створення даних для дошки методом Kanban за адресою: – дошки - /api/boards; – колонки - /api/boards/id_дошки/columns; – картки - /api/columns/id_колонки/cards. 6. Створення даних у базі даних у таблицях: – Boards - для дошок; – Columns - для колонок; – Cards - для карток; 7. Відправка оновлених даних на інтерфейсну частину. 8. Виведення даних на екран користувача. При цьому створюються такі колонки: «Резерв», «До роботи», «У роботі», «Заблоковано», «Тест», «Виконано».

Продовження таблиці 4.6

Алгоритм	Кроки
Створення дошок методом Waterflow	1. Користувач має увійти у систему. 2. На екрані Дошки користувач повинен натиснути кнопку «+» для створення дошки. 3. Користувач переходить на екран, де необхідно обрати метод управління проєктом «Waterflow». 4. Після вибору методу проєкта, користувач має ввести ім'я дошки та додати користувачів до дошки якщо це необхідно. 5. Після того як користувач натискає на кнопку «Створити дошку», йде запит на серверну частину для створення даних для дошки методом Waterfall за адресою: – дошки - /api/boards; – колонки - /api/boards/id_дошки/columns; – картки - /api/columns/id_колонки/cards. 6. Створення даних у базі даних у таблицях: – Boards - для дошок; – Columns - для колонок; – Cards - для карток; 7. Відправка оновлених даних на інтерфейсну частину. 8. Виведення даних на екран користувача. При цьому створюються такі колонки: «Резерв», «До роботи», «У роботі», «Заблоковано», «Тест», «Виконано». Користувач не може створювати нові колонки. Картки можна створювати тільки в колонці «Резерв» та перетягувати її тільки в сусідню колонку (тобто не можна перетягнути з «Резерв» відразу в «Тест»).

Продовження таблиці 4.6

Алгоритм	Кроки
Створення дошок вільним методом	1. Користувач має увійти у систему. 2. На екрані Дошки користувач повинен натиснути кнопку «+» для створення дошки. 3. Користувач переходить на екран, де необхідно обрати метод управління проєктом «Вільний метод». 4. Після вибору методу проєкта, користувач має ввести ім'я дошки та додати користувачів до дошки якщо це необхідно. 5. Після того як користувач натискає на кнопку «Створити дошку», йде запит на серверну частину для створення даних для дошки вільним методом за адресою: – дошки - /api/boards; – колонки - /api/boards/id_дошки/columns; – картки - /api/columns/id_колонки/cards. 6. Створення даних у базі даних у таблицях: – Boards - для дошок; – Columns - для колонок; – Cards - для карток; 7. Відправка оновлених даних на інтерфейсну частину; 8. Виведення даних на екран користувача.

4.4 Основні рішення щодо модульного уявлення системи

Програмна система для спрощення та автоматизації управління проєктами різними методами розподілена на модулі та компоненти з даними і

функціями. Таке рішення дозволяє з легкістю використовувати ці данні для виконання своїх задач для роботи системи (рис. 4.2).

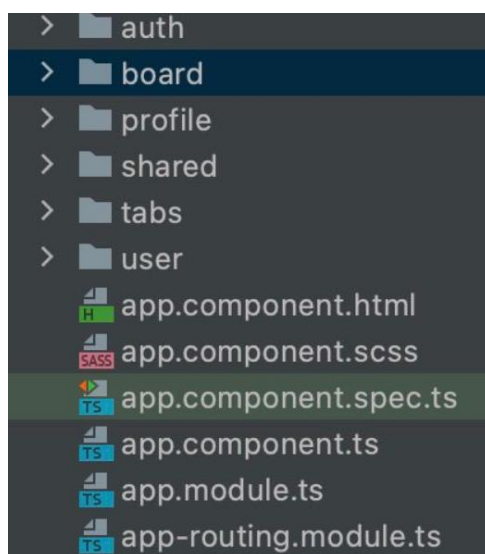


Рисунок 4.2 – Програмна система з файлами інтерфейсу

Розглянемо ці модулі більш детально:

- Auth – модуль авторизації та реєстрації;
- Board – модуль створення, перегляду та редагування дошки та її компонентів (колонка, картка);
- Profile – модуль перегляду та редагування профілю користувача;
- Shared – модуль з функціями, які використовуються у різних частинах сайту;
- Tabs – модуль з таб баром, який розміщено знизу екрану;
- User – модуль перегляду інших користувачів;
- app.component.html – опис інтерфейса головного компонента сайту;
- app.component.scss – стилі головного компонента сайту;
- app.component.ts – функції головного компонента сайту;
- app.module.ts – реалізація головного модуля;
- app-routing.module.ts – реалізація роутінгу головного модуля;

Нижче на рисунку 4.3 представлені файли серверної частини програмного продукту.

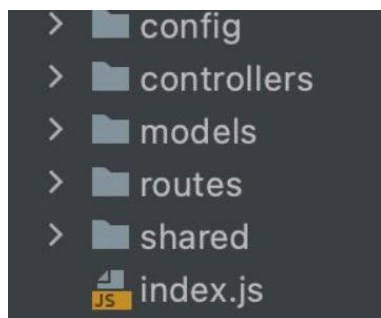


Рисунок 4.3 – Програмна система з файлами серверної частини

Розглянемо файли серверної частини більш детально:

- Config – папка, у якій знаходяться основні налаштування серверної частини;
- Controllers – папка, у якій міститься бізнес частина проєкту;
- Models – папка з моделями сайту;
- Routes – папка з роутінгами сайту;
- Shared – папка з функціями, які використовуються у різних частинах сайту;
- Index.js - імпорти усіх налаштувань та функції запуску серверу.

4.5 Висновок за розділом 4

У цьому розділі було описано основні рішення щодо розробки програмного продукту, а саме були описані основні алгоритми та методи, реалізовані під час розробки програмної системи для спрощення та автоматизації управління проєктами різними методами.

При розробці програмного продукту був використаний механізм моделей та компонентів.

5 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРЕМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

5.1 Призначення та функції програмного продукту

Програмна система призначена для бізнес або ІТ компаній для спрощення та автоматизації управління проєктами різними методами. Мобільним вебсайтом зазвичай керує проєкт менеджер, та користуватися їм можуть абсолютно всі члени компанії, працюючі над тим чи іншим проєктом.

Використовувати програмний продукт може людина з мінімальними знаннями керування проєктами та мінімальними навичками володіння мобільним девайсом. Зручний, інтуїтивний та ергономічний інтерфейс спрощує пристосування до нового програмного продукту.

Основні функції, що має програмний продукт:

- авторизація у системі;
- створення дошок за певним методом управління проєктами (Scrum, Kanban, Waterflow, «Вільна дошка»);
- дошка створена за обраним методом відповідає правилам одному з методів управління проєктами (Scrum, Kanban, Waterflow, «Вільна дошка»);
- створювання колонок та карток;
- переміщення карток між колонками;
- додавання користувачів до потрібної дошки або картки;
- оцінка та врахування часу у картці;
- можливість залишати коментарі у картці;
- можливість ставити пріоритет картці;
- отримання повідомлень про дії на дошці, до якої доданий користувач;
- перегляд історії картки.

5.2 Умови роботи програмного продукту

Основні вимоги до мобільного девайсу, необхідні для стабільної роботи і функціональності програмного продукту. В характеристиках вказані мінімальні параметри.

Системні вимоги:

- процесор 4 або 8 ядер;
- процесор з частотою від 1,2 ГГц;
- 1-2 Гб оперативної пам'яті
- 8-16 Гб вбудованої пам'яті;
- операційна система: iOS або Android;
- інтернет-підключення зі швидкістю не менше ніж 4 Мбіт/с.

5.3 Робота мобільного вебсайту

Для початку роботи з програмним продуктом необхідно виконати наступні кроки:

- упевнитися у відповідності характеристик мобільного пристрою, на якому буде працювати мобільний вебсайт;
- зайти в мобільний браузер, ввести адресу вебсайту в адресному рядку та перейти на нього.

Подальша робота з мобільним вебсайтом здійснюється за допомогою GUI інтерфейсу. Нижче представлений інтерфейс мобільного вебсайту та його детальний опис.

Перший крок – це авторизація (рис. 5.1).

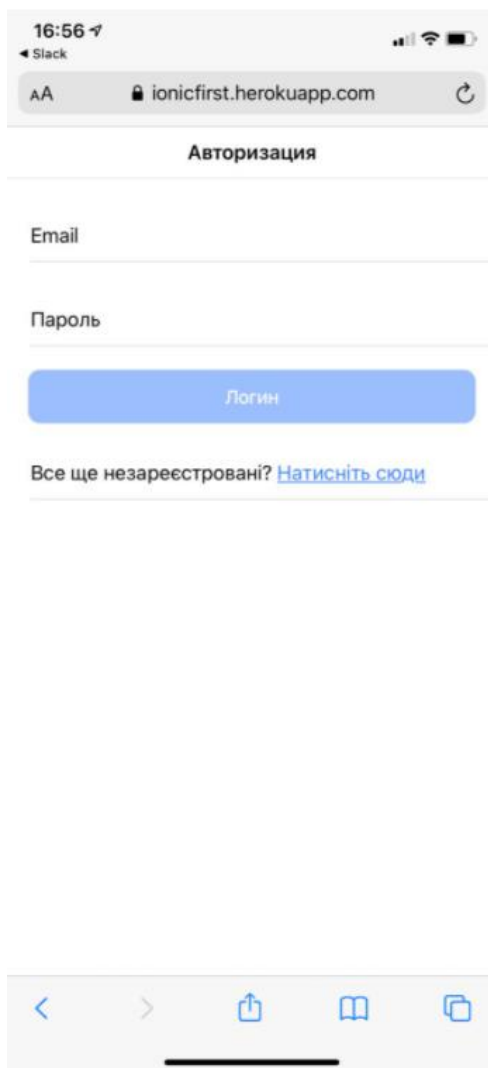


Рисунок 5.1 – Авторизації користувача

Для початку користування продуктом користувачу необхідно авторизуватися. Для цього необхідно ввести e-mail та пароль в поля для вводу та натиснути кнопку «Логін» (рис. 5.1).

Якщо користувач не має акаунту, потрібно натиснути кнопку «Натисніть сюди». На сторінці що з'явилась в поля для вводу потрібно ввести повне ім'я, e-mail, пароль та підтвердження паролю, потім треба натиснути кнопку «Реєстрація» (рис. 5.2).

18:20 18:20 18:20

AA ionicfirst.herokuapp.com

Реєстрація

Повне ім'я

Email

Пароль

Введіть пароль ще раз

Реєстрація

Вже зареєстровані? [Авторизуйтесь тут](#)

< > ⬆️ 📖 📄

Рисунок 5.2 – Реєстрація

Наступний крок – головна сторінка – Дошки (рис. 5.3).

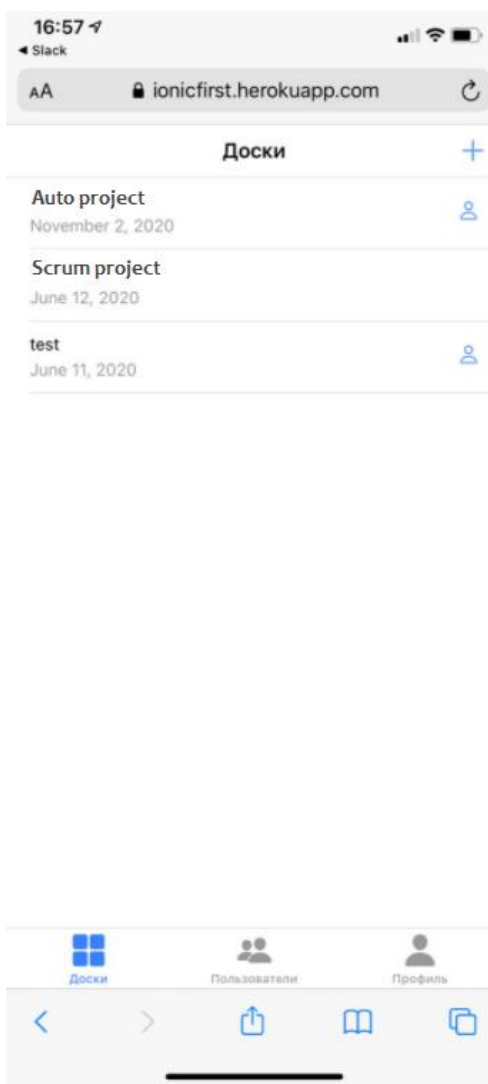


Рисунок 5.3 – Доски

Після авторизації користувач потрапляє на екран з доступними дошками (якщо вони були створені).

Тут користувач може створити нову дошку, якщо натисне на кнопку «+», або перейти на вже створену дошку.

Також на цьому екрані розміщено Bottom bar для переходу до списку користувачів (кнопка «Користувачі») та профілю (кнопка «Профіль»).

Після цього користувач переходить до створення дошки (рис. 5.4 – 5.6).

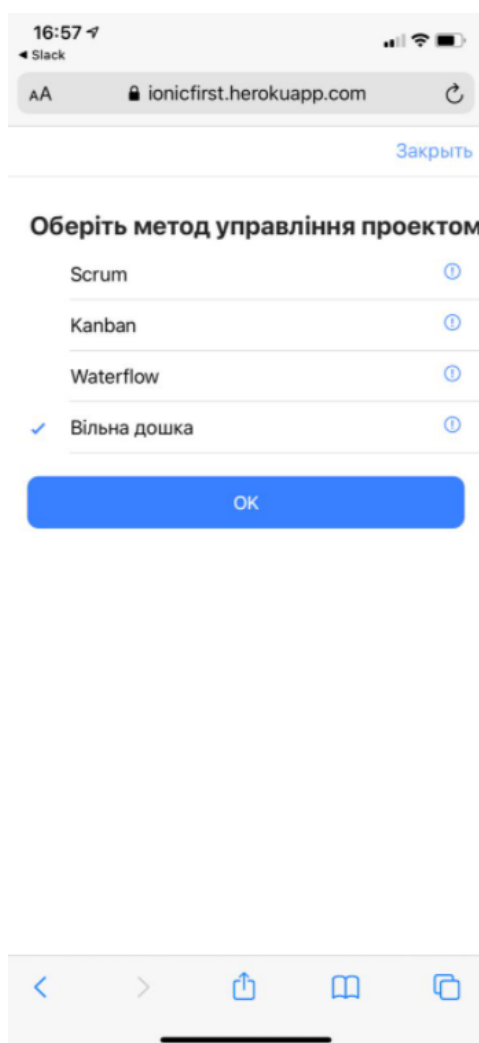


Рисунок 5.4 – Створення дошки

Перш ніж створити дошку користувачу необхідно обрати метод управління проектом. На екрані користувач може обрати один з чотирьох методів.

1. Scrum. При виборі методу Scrum користувач має можливість створювати спринти та задачі у них. Проектна команда може почати наступний спринт тільки коли попередній закінчений (рис. 5.5). Тобто спершу користувач має можливість створити спринти, в яких зможе створювати задачі та переміщувати їх між колонками. При створенні спринта, відкривається дошка з колонками «Резерв», «До роботи», «У роботі», «Заблоковано», «Тест», «Виконано». Немає можливості створювати нові колонки. Також користувач має можливість переміщати картки між спринтами.

2. Kanban. При виборі методу Kanban користувач має можливість вільно пересуватися між стадіями розробки та призупинити виконання завдання на будь-якому етапі в разі, якщо з'явилися інші термінові завдання або змінився пріоритет поточної. Створюється дошка з колонками «Резерв», «До роботи», «У роботі», «Заблоковано», «Тест», «Виконано». Немає можливості створювати нові колонки.

3. Waterflow. При виборі методу Waterflow користувач має можливість переходити від однієї стадії до іншої строго послідовно. Створюється дошка з колонками «Резерв», «До роботи», «У роботі», «Тест», «Виконано». Немає можливості створювати нові колонки. Картки можна створювати тільки в «Резерв». Картку можна перетягувати тільки в сусідню колонку (тобто не можна перетягнути з «Резерв» відразу в «Тест»).

4. Вільна дошка. При виборі вільного методу користувач має можливість створювати будь-які колонки і переміщуватись вільно між ними.

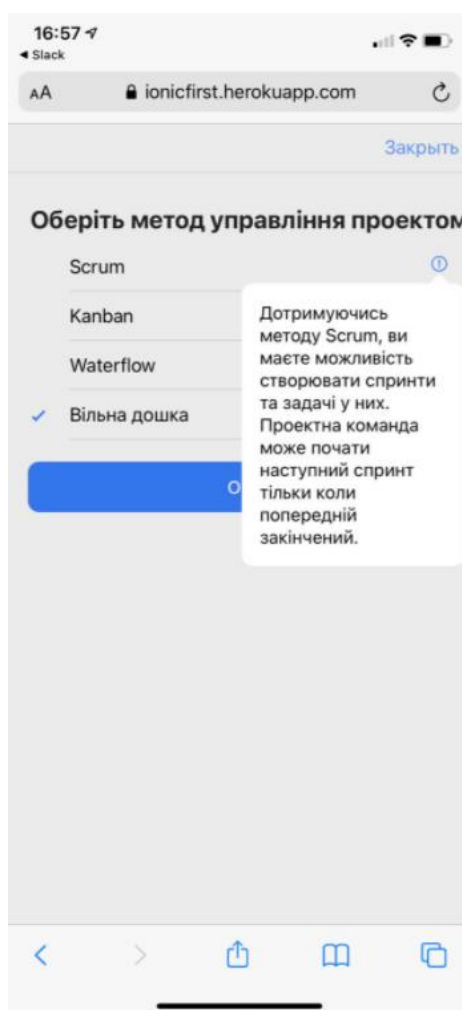


Рисунок 5.5 – Створення дошки за методом Scrum

Після того як користувач обрав метод управління проектом, необхідно ввести ім'я дошки. Також користувач має можливість додати інших користувачів до своєї дошки, для цього необхідно натиснути на поле «Додати користувача до дошки» та почати вводити ім'я користувача (рис. 5.6). Якщо хоч один користувач доданий до дошки або картки, на них буде відображатися іконка користувача. Також кожному доданому користувачу при будь-яких змінах на дошці або картці прийде повідомлення.

Після цього користувачу потрібно натиснути кнопку «Створити дошку».

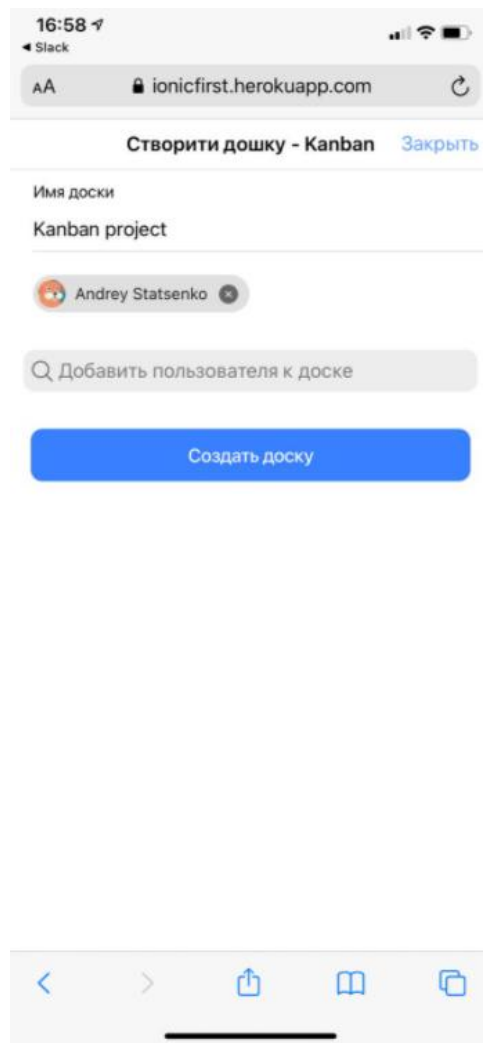


Рисунок 5.6 – Створення дошки

Далі користувач переходить до дошки, екрану з колонками (рис. 5.7).

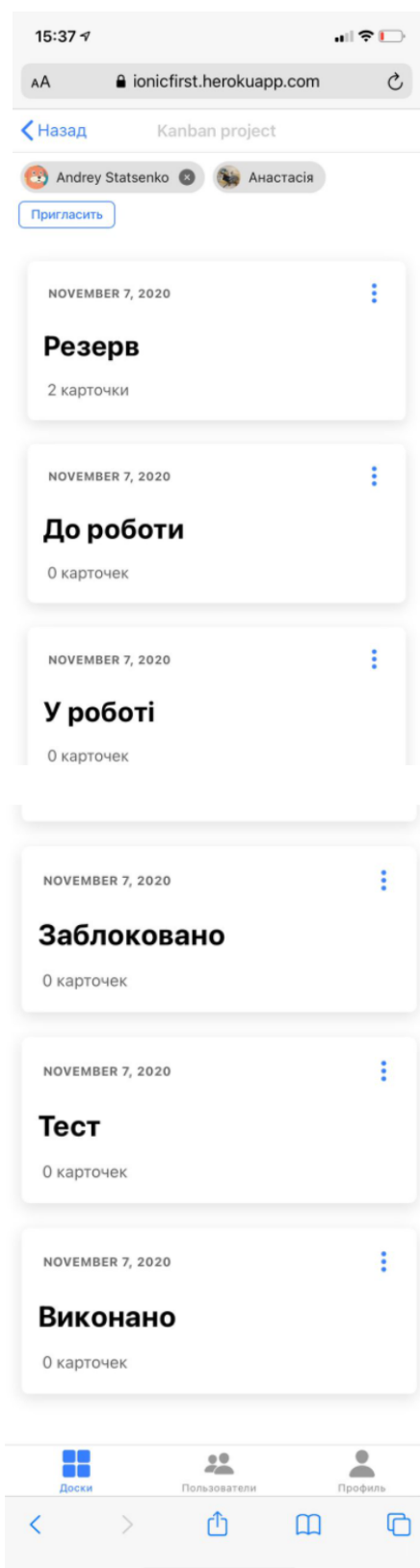


Рисунок 5.7 – Дошка, екран з колонками

На екрані дошки користувач може побачити ім'я дошки, користувачів що запрошені до цієї дошки та перелік колонок цієї дошки.

Користувач може запросити інших користувачів до своєї дошки натиснувши на кнопку «Запросити». Також користувач може перейти до будь – якої колонки, видалити її або створити нову (якщо обран вільний метод управління проєктом).

Також на цьому екрані розміщено Bottom bar для переходу до списку користувачів (кнопка «Користувачі») та профілю (кнопка «Профіль»).

Далі користувач переходить до дошки, екрану з картками (рис. 5.8).

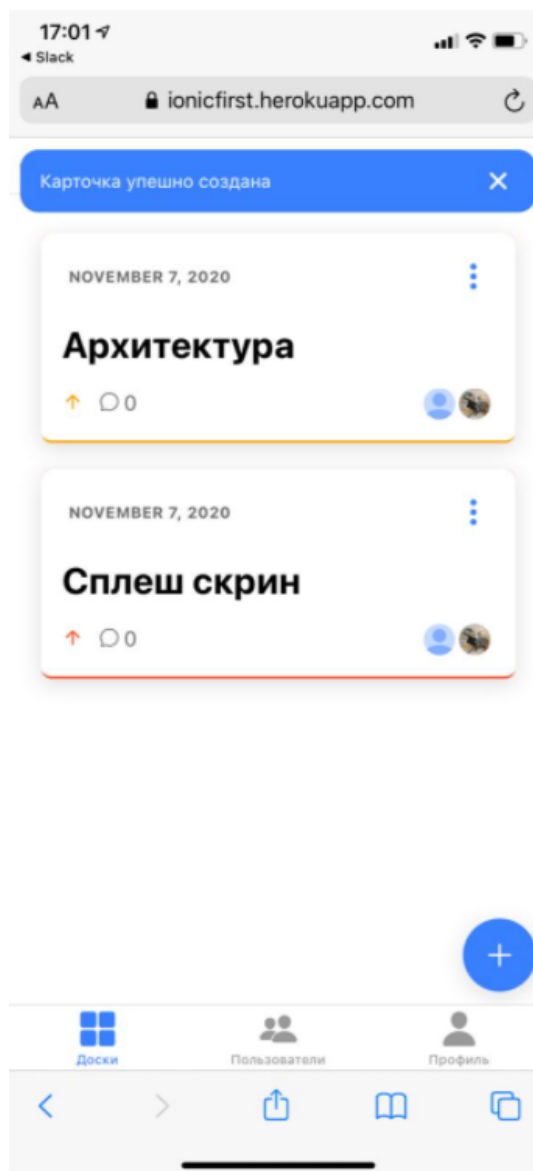
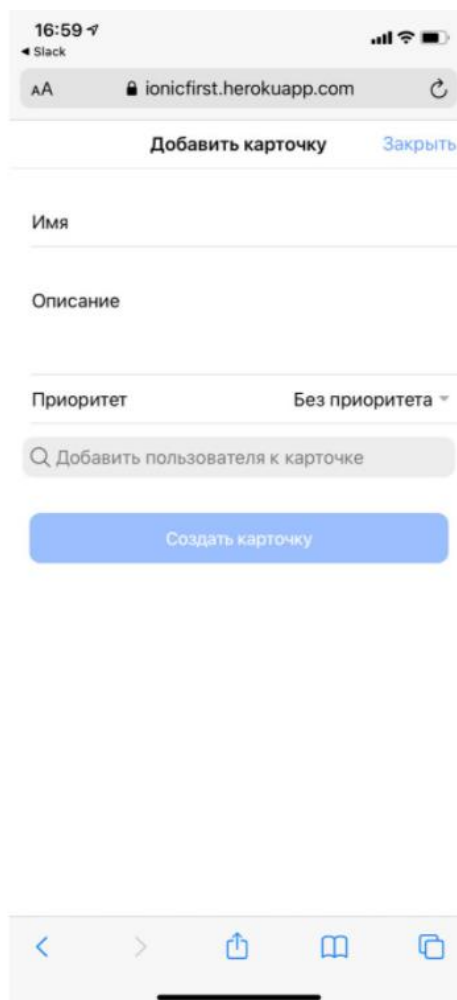


Рисунок 5.8 – Дошка, экран з картками

На цьому екрані користувач має можливість створити необхідні картки. Для цього потрібно натиснути на кнопку «+». На екрані, що з'явиться, потрібно ввести ім'я картки, опис, виставити пріорітет та додати користувача до картки, якщо необхідно (рис. 5.9). Після цього треба натиснути на кнопку «Створити картку», вона з'явиться на попередньому екрані та користувач побачить повідомлення «Картка успішно створена».



The screenshot shows a mobile application interface for creating a card. At the top, there is a status bar with the time 16:59 and a signal strength indicator. Below the status bar, there is a header with a back arrow and the word 'Slack'. The main content area contains a form with the following elements: a text input field labeled 'Имя' (Name), a text input field labeled 'Описание' (Description), a dropdown menu labeled 'Приоритет' (Priority) with the selected option 'Без приоритета' (No priority), and a search bar labeled 'Добавить пользователя к карточке' (Add user to card). At the bottom of the form is a large blue button labeled 'Создать карточку' (Create card). The bottom of the screen shows a mobile home indicator bar.

Рисунок 5.9 – Створення картки

У обраній колонці з картками користувач має можливість оцінити картку, занести у неї витрачений час, перемістити картку у іншу колонку, переглянути деталі картки, редагувати та видалити. Всі ці дії можна побачити натиснувши на три крапки на будь-якій картці (рис. 5.10).

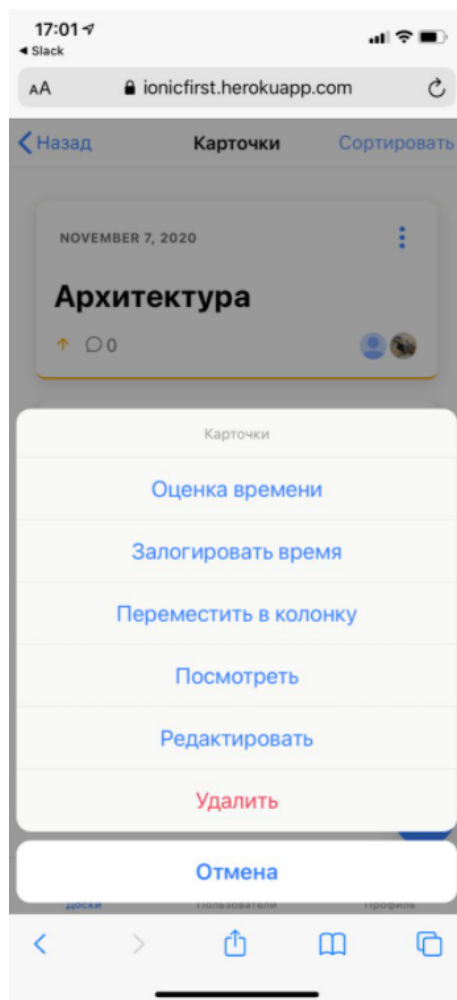


Рисунок 5.10 – Дії, які користувач може робити з карткою

Наступний крок – це деталі картки (рис. 5.11).

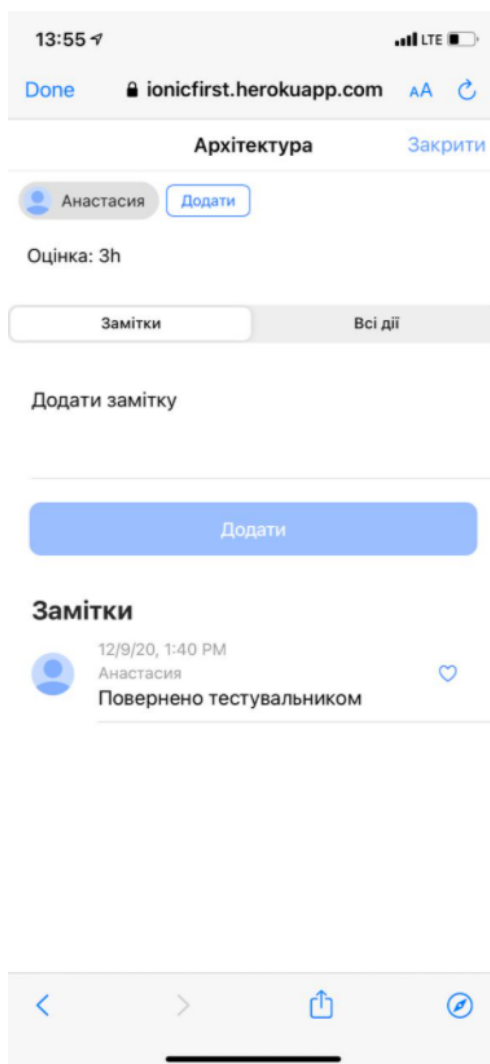


Рисунок 5.11 – Деталі картки

При перегляді картки користувач може побачити ім'я картки, користувачів, що додані до картки, опис картки, оцінку, витрачений час, замітки до картки та історію дій користувачів.

Для того щоб додати замітку, потрібно написати її у полі «Додати замітку» та натиснути кнопку «Додати». Будь-яку замітку можна оцінити, натиснувши на іконку серце біля замітки.

Для того щоб побачити історію дій користувачів з цією карткою, потрібно натиснути на кнопку «Всі дії» (рис. 5.12).

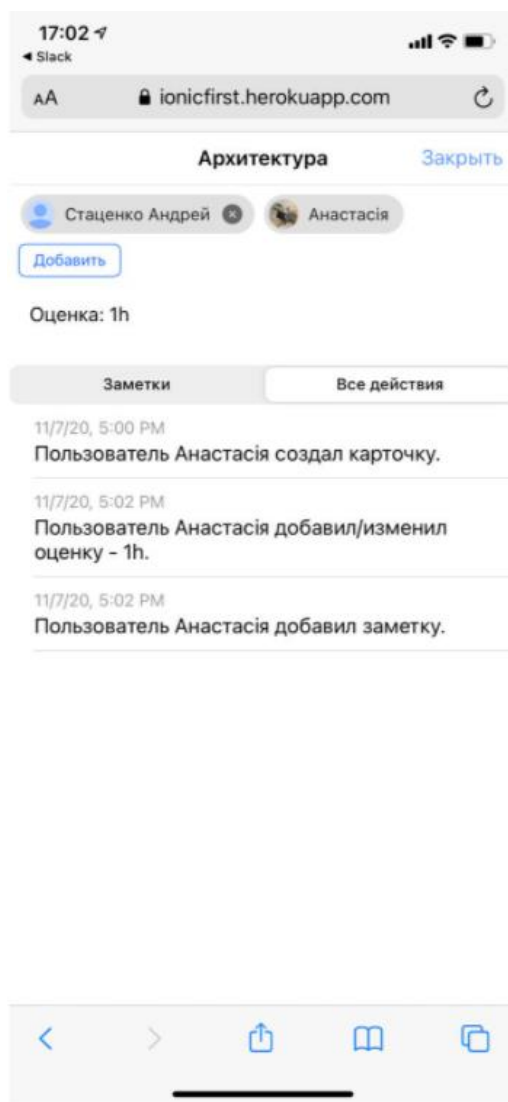


Рисунок 5.12 – Історія дій користувачів картки

Також користувач може потрапити на екран з усіма користувачами (рис. 5.13).

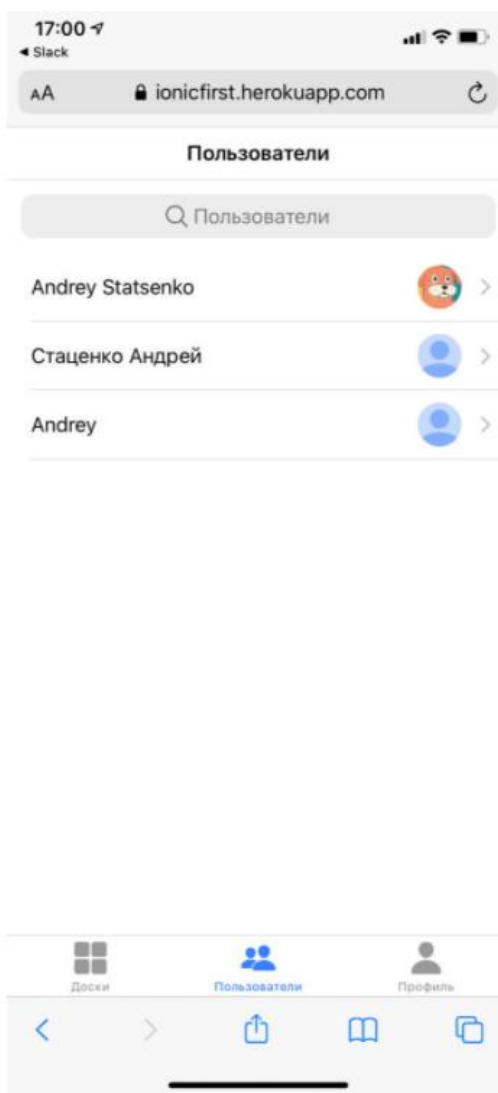


Рисунок 5.13 – Користувачі

На цьому екрані відображені всі інші користувачі, які користуються програмною системою для спрощення та автоматизації управління проєктами різними методами. Натиснувши на користувача можна переглянути інформацію про нього. Також на цьому екрані є можливість шукати користувачів за ім'ям.

Наступний крок – профіль (рис. 5.14).

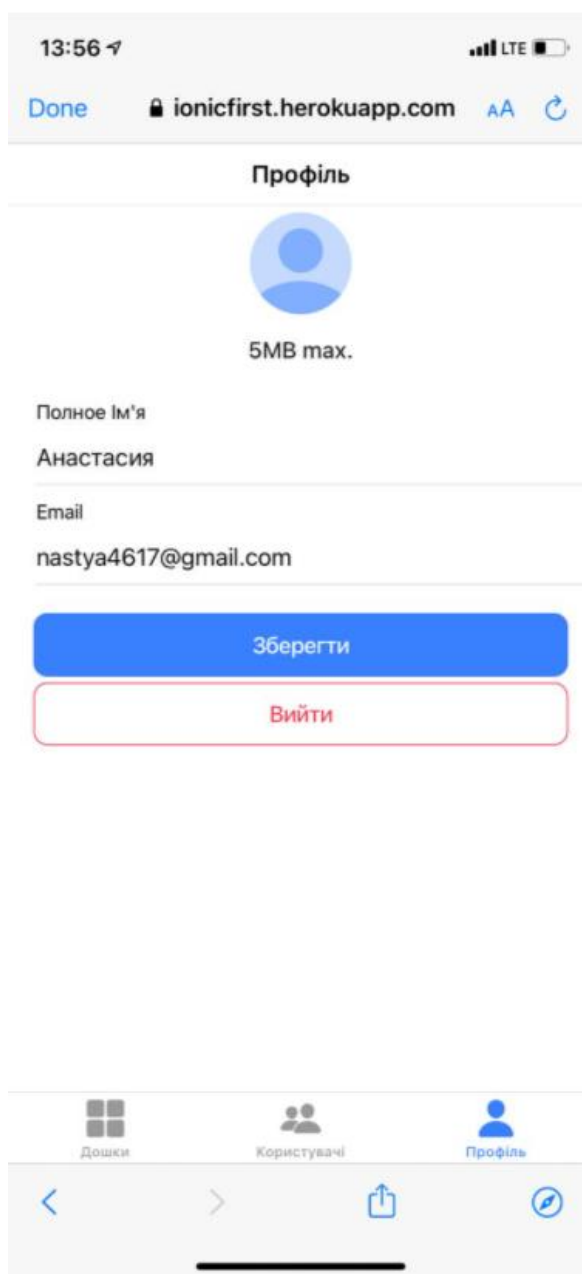


Рисунок 5.14 – Профіль

На екрані профіль користувач має можливість змінити своє фото, ім'я та електронну адресу. Для збереження відредагованих даних потрібно натиснути на кнопку «Зберегти». Також користувач має можливість вийти зі свого акаунту, натиснувши на кнопку «Вийти».

5.4 Висновки за розділом 5

В даному розділі було розглянуто інтерфейс програмного продукту та детально описано кроки, що виконує програма. Також було реалізовано «Керівництво користувача».

6 ЕКОНОМІКО - ОРГАНІЗАЦІЙНА ЧАСТИНА

Економічна ефективність – це результат, який можна отримати, порівнюючи показники прибутковості виробництва по відношенню до загальних витрат і використаних ресурсів [14].

Система показників ефективності виробництва повинна давати всебічну оцінку використання всіх ресурсів підприємства і містити всі загальноекономічні показники.

Процес розробки програмного продукту супроводжується необхідністю вирішення багатьох економічних проблем. Одна з важливих економічних проблем - це визначення вартості програмного продукту.

Витрати на розробку програмного продукту включають в себе наступні пункти:

- витрати на оплату праці;
- додаткова заробітна платня;
- єдиний соціальний податок;
- витратні матеріали;
- витрати на обладнання;
- експлуатаційні витрати;
- інші витрати;
- прибуток.

Для фірми-розробника вебсайту джерелом доходу є продаж програмного продукту замовникам. Витрати фірми включають в себе витрати на розробку. Джерелом фінансування є власні кошти фірми-розробника.

Для підприємства-замовника джерелом економії виступає заміна ручної праці машинною, що в значній мірі скорочує час проведення обробки даних. Витрати підприємства складаються з одноразових витрат на розробку дизайну сайту, створення шаблонів, створення таблиць стилів, наповнення програмним

кодом, а також витрат, безпосередньо пов'язаних з впровадженням і підтримкою сайту.

6.1 Розрахунок собівартості розробки програмної системи для спрощення та автоматизації управління проєктами різними методами

6.1.1 Розрахунок основної заробітної платні

Розробку сайту проводять два фахівці: проєктувальник і програміст. Зарплата проєктувальник становить 150 грн / годину, програміста - 200 грн / годину. При цьому тривалість робочого дня кожного з них становить 8 годин.

Розрахунок основної заробітної плати наведено в таблиці 6.1.

Таблиця 6.1 – Розрахунок основної заробітної плати

Етапи	Виконавець			Час на виконання/ година	Сума заробітної платні
	Кількість	Фах	Заробітна платня		
Формулювання вимог до продукту	1	Проєктувальник	150	8	1 200
Створення базового шаблону веб-сторінки	1	Проєктувальник	150	8	1 200
	1	Програміст	200	16	3 200
Реалізація серверної частини	1	Програміст	200	32	6 400
Реалізація алгоритмів	1	Програміст	200	24	4 800

Продовження таблиці 6.1

Етапи	Виконавець			Час на виконання/ година	Сума заробітної платні
	Кількість	Фах	Заробітна платня		
Зв'язування вебсайту	1	Програміст	200	16	3 200
Етап тестування і налагодження	1	Проектувальник	150	8	1 200
	1	Програміст	200	8	1 600
Всього				120	22 800

6.1.2 Розрахунок додаткової заробітної платні

До додаткової заробітної платні відносяться: оплата відпусток, виплата винагороди за вислугу років і т.д. Додаткова заробітна платня становить 20% від основної:

$$22\,800 \cdot 0,2 = 4\,560 \text{ грн.}$$

6.1.3 Розрахунок єдиного соціального податку

Величина єдиного соціального податку визначається в процентному відношенні до суми основної відповідно до встановленого нормативу (22%).

$$22\,800 \cdot 0,22 = 5\,016 \text{ грн.}$$

6.1.4 Визначення витрат на матеріали

До інших витрат слід віднести витрати на обслуговування ПК та плату за електроенергію. А також вартість основних і допоміжних матеріалів (папір, флеш-накопичувач, картридж і т.д.) необхідних для розробки проєкту. Розрахунок наводиться в табл. 6.2.

Витрати на електроенергію розраховуються виходячи зі споживаної потужності пристрою і тарифу на електроенергію. Для розробки програмної системи для спрощення та автоматизації управління проєктами різними методами передбачається використання комп'ютера з потужністю 0,2 кВт/годину. Вартість одного кВт години електроенергії дорівнює 1 грн. Час використання електроенергії в процесі розробки – 120 годин.

Отже, плата за електроенергію складає:

$$0,2 \cdot 120 \cdot 1 = 24 \text{ грн.}$$

Витрати на обслуговування ЕОМ визначаються з вартості ЕОМ і часу її експлуатації, після закінчення якого, вона підлягає заміні (зазвичай цей час не перевищує 3-х років), протягом року ЕОМ використовує 254 робочих дні. В середньому ЕОМ коштує близько 20 000 грн. Тоді:

$$(20\,000 / (3 \cdot 8 \cdot 254)) \cdot 120 = 393,7 \text{ грн.}$$

Таблиця 6.2 – Визначення витрат на матеріали

Найменування матеріалу, вим.	Кількість	Ціна, грн	Загальні витрати, грн
Бумага для принтеру, уп.	2	103	206
Картридж	1	280	280

Продовження таблиці 6.2

Найменування матеріалу, вим.	Кількість	Ціна, грн	Загальні витрати, грн
Канцтовари	Набор	250	250
Флеш-пам'ять (8Gb)	1	350	350
Сервер	1	200	200
Інтернет	1 місяць	150	150
Електроенергія			24
Обслуговування ЕОМ			393,7
Всього			1 857,7

6.2 Розрахунок експлуатаційних витрат реалізації проєкту

Експлуатаційні витрати, на відміну від капітальних, повторюються в кожному циклі виробництва, а розраховуються в сумі за рік. Експлуатаційні витрати здійснюються синхронно з виробництвом. Експлуатаційні витрати становлять собівартість продукції або послуг. До складу експлуатаційних витрат входять наступні витрати:

- заробітна платня управлінського персоналу (менеджера), що працює з використанням програмного продукту;
- витрати на підтримку і актуалізацію продукту (відновлення, контроль даних і програм та інші операції) – 32 години / місяць;
- витрати на споживану електроенергію;
- витрати на сервер;
- витрати на інтернет;
- витрати на ремонт ЕОМ;
- інші витрати.

Розрахунок експлуатаційних витрат за рік наведено в таблиці 6.3.

Таблиця 6.3 – Розрахунок експлуатаційних витрат

Найменування витрат	Кількість	Ціна, грн	Загальні витрати за рік, грн
Заробітна платня управлінського персоналу	1	100 грн / годину	203 200
Актуалізація продукту	16 години / місяць;	200 грн / годину	38 400
Електроенергія	2 032 години / рік	1 грн / 0,2 кВт	406,4
Сервер	1	200 / місяць	2 400
Інтернет	1	150 / місяць	1 800
Обслуговування ЕОМ	1	20 000 / 3 роки	833,3
Всього			247 039,7

6.3 Визначення економічної ефективності проєкту

Беручи до уваги те, що програмний продукт значно заощаджує час працівників компанії, яка використовує мобільний вебсайт, було вираховано на скільки збільшиться прибутковість підприємства.

Середньо статистична компанія робить близько 25 проєктів на місяць, маючи у штатному складі 5 менеджерів проєкту. Тобто один менеджер зазвичай веде близько 5 проєктів. Використання мобільного вебсайту заощадить час менеджера та дозволить узяти кожному менеджеру додатковий проєкт. В результаті в місяць компанія буде робити близько 30 проєктів, а в рік – близько 360.

Отже передбачається, що з впровадженням програмної системи для спрощення та автоматизації управління проєктами різними методами прибутковість підприємства збільшиться на 20% в порівнянні з попереднім роком за рахунок економії часу, роботи з більшою кількістю проєктів, та відповідно можливістю обслуговувати більше нових клієнтів.

Обороти підприємства, що планує купити продукт за 2019 рік склали 123 240 000 грн, отже додаткова виручка дорівнює:

$$123\,240\,000 \cdot 0,2 = 24\,648\,000 \text{ грн.}$$

Тепер знайдемо додатковий прибуток за формулою (6.1):

$$\Pi_{\text{доп.}} = (B_{\text{доп.}} - P_{\text{ус.}}) \cdot R, \quad (6.1)$$

де $\Pi_{\text{доп.}}$ – додатковтй прибуток за рік, в гривнях;

$B_{\text{доп.}}$ – додаткова виручка за рік, в гривнях;

$P_{\text{ус.}}$ – витрати на утримання сайту за рік, в гривнях;

R – рентабельність, в%.

$$(24\,648\,000 - 247\,039,7) \cdot 20\% = 4\,880\,192 \text{ грн.}$$

Додатковий прибуток становить майже 5 млн. грн, що значно перевищує витрати на утримання сайту, які дорівнюють майже 250 тис. грн на рік.

6.3.1 Фінансовий план

На розробку програмного продукту необхідно виділити кошти, які складаються з витрат на оплату праці та додаткової заробітної платні, єдиного

соціального податку, коштів на витратні матеріали, витрати на обладнання та експлуатаційні витрати:

$$22\,800 + 4\,560 + 5\,016 + 1\,857 = 34\,233 \text{ грн.}$$

Отже собівартість мобільного вебсайту складає 34 233 грн.

Було вирішено, що для успішної реалізації програмного продукту його ціна повинна бути не вище відносної більшості програмних продуктів подібного типу. А також з урахуванням підрахованого вище річного прибутку підприємства, оптимальною вартістю на одну копію програмного продукту буде – 25 000 грн.

Таким чином для повернення витрачених коштів достатньо реалізувати 2 копії програмного продукту, всі наступні реалізовані копії почнуть приносити дохід. З огляду на охоплення цільової аудиторії, якій може знадобитися інформаційний ресурс, це прийнятно.

На початковому етапі фірма-розробник не може собі дозволити розширити персонал. Отже з урахуванням того, що розробку сайту проводять два фахівці: проєктувальник і програміст, та необхідного часу на реалізацію одного вебсайту, планується в середньому розробляти та продавати 2-3 копії програмного забезпечення на місяць.

Отже середній дохід за місяць складе:

$$62\,500 - 34\,233 = 28\,267 \text{ грн.}$$

Вирахуємо прибуток фірми-розробник:

$$28\,267 \cdot 12 = 339\,204 \text{ грн.}$$

Вирахуємо термін окупності капітальних вкладень:

$$34\,233 / 28\,267 = 1,2 \text{ місяці.}$$

Отже прибуток за рік становить 339 204 грн. Розробка і впровадження системи для спрощення та автоматизації управління проєктами різними методами є економічно ефективною, при цьому термін окупності капітальних вкладень складе 2 місяці.

7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ

7.1 Аналіз потенційних небезпек

У даному розділі розглянуто основні потенційні небезпеки приміщення, де безпосередньо проводиться дослідження, модифікація та програмна реалізація методів управління проєктами.

Основна робота при дослідженні проводиться за комп'ютером, тому було визначено фактори, що впливають на дослідника в процесі роботи:

- незадовільні параметри мікроклімату робочої зони (значення показників мікроклімату у виробничих приміщеннях повинні задовільняти нормам, зазначеними у ДСН 3.3.6.042-99 Санітарні норми мікроклімату виробничих приміщень [15];

- відсутність або недостатність природного освітлення;

- недостатня освітленість робочого місця, згідно ДБН В.2.5-28:2018. Природне і штучне освітлення [16];

- широкий спектр випромінювання від дисплея, який включає рентгенівську, ультрафіолетову і інфрачервону області, а так само широкий діапазон електромагнітних випромінювань інших частот (згідно Державних санітарних норм і правил 3.3.6.096-2002 «Державними санітарними нормами і правилами при роботі з джерелами електромагнітних полів» [17], і ГОСТ 12.1.045-84 «Система стандартів безпеки праці. Електростатичні поля. Допустимі рівні на робочих місцях і вимоги до проведення контролю» [18]);

- небезпека виникнення пожежі (згідно ДБН В.1.1-7:2016 «Пожежна безпека об'єктів будівництва. Загальні вимоги» [19]);

- статична напруга, яка здатна вивести з ладу електронні компоненти;

- небезпечний рівень напруги який може видавати блок живлення ПК;

- погано організоване робоче місце, що може привести до уражень струмом, механічних травм та інше.

Дослідження проводяться у приміщенні, що має розміри 25х15м². Також у цьому приміщенні розташовано 24 робочих місця, обладнаних персональними комп'ютерами.

У приміщенні присутні 3 вікна, розташованих з одного боку та не по центру приміщення, що може призвести до проблеми з освітленням. Також приміщення обладнане великою кількістю розеток, що мають 3 контакти (фаза, нуль та заземлення), згідно з ДСТУ Б В.2.5-82:2016 Електробезпека в будівлях і спорудах. Вимоги до захисних заходів від ураження електричним струмом [20].

Для забезпечення відповідного мікроклімату приміщення обладнано двома спліт-системами кондиціювання. Вони ж і забезпечують іонізацію повітря та його зволоження.

7.2 Заходи із забезпечення безпеки

Робота з обчислювальною технікою зумовлює виникнення фактору небезпеки, що пов'язаний з електрострумом.

Ураження електрострумом може бути наслідком невиконання техніки безпеки, при несправності електроприладів, при недотриманні вимог ДСТУ Б В.2.5-82:2016 Електробезпека в будівлях і спорудах. Вимоги до захисних заходів від ураження електричним струмом [21].

Залежно від категорії приміщення здійснюються певні заходи, які забезпечують достатню безпеку при експлуатації і ремонті електроустаткування.

З метою попередження уражень електричним струмом до роботи допускаються лише особи, які добре освоїли основні правила техніки безпеки.

Для запобігання ураження людини при дотику до частин ЕОМ, що можуть опинитися під напругою, виконується захисне заземлення. Суть захисного заземлення – зменшення напруги на корпусі електроустаткування при пробії фази на корпус до безпечної величини. Сучасні ЕОМ включаються в розетку за допомогою триполюсною вилки, один з полюсів якої

використовується для заземлення. Згідно Правил улаштування електроустановок опір захисного заземлення в приміщенні для комп'ютерної техніки не перевищує 4 Ом, що значно менше опору тіла людини, тому небезпека ураження при дотику до несправного електроустаткування значно зменшується, так як струм в цьому випадку піде шляхом найменшого опору [22].

Специфічна небезпека електроустановок – дроти, за якими проходить струм, корпуси стійок ЕОМ і іншого устаткування. Реакція людини на електричний струм виникає лише при протіканні струму через тіло людини.

Дуже важливе значення для запобігання електротравматизму має правильна організація обслуговування електроустановок, які діють, проведення ремонтних, монтажних і профілактичних робіт, а також організація правильного заземлення обладнання.

Організацію заземлення корпусу персонального комп'ютера (ПК) через нульовий дріт мережі живлення неприпустимо, оскільки при обриві робочого нуля корпус ПК може опинитися під «фазною» напругою, крім того, при великому навантаженні в електричній мережі на робочому нулі зосереджується значний потенціал.

Для забезпечення безпеки робіт в електроустановках, які діють, виконуються наступні організаційні заходи:

- призначення осіб, відповідальних за організацію і виконання робіт;
- оформлення робіт нарядом або усним розпорядженням;
- допуск до роботи;
- спостереження під час проведення робіт;
- забезпечення перерви в роботі.

При проведенні робіт зі зняттям напруги в електроустановках, які діють, або поблизу них, необхідно:

- відключення установки від джерела живлення електроенергією;
- механічне замикання приводів відключених комутаційних апаратів;
- зняття запобіжників;

- від'єднання кінців ліній живлення;
- заходи, які забезпечують неможливість помилкової подачі напруги до місця роботи;
- використання заземлень.

Окрім ураження електрострумом при дослідженнях можна отримати механічні пошкодження від інструментів, що необхідні для розробки мережі.

А саме, для роботи з кабелем (витою парою) необхідно використовувати спеціальний інструмент – кримпер, що має гострі леза для перерізання кабелів та має спеціальний механізм для стискання конекторів. Тому, це досить травмонебезпечний пристрій, який може спричинити механічні пошкодження частин тіла людини. Тому при роботі з ним необхідно дотримуватись наступних правил безпеки:

- ознайомитись з інструкцією приладу;
- одягнути спеціальні захисні перчатки для захисту рук;
- переконатися у тому, що робоче місце підходить для роботи з інструментом;
- переконатись, що інструмент справний;
- після роботи з інструментом, закрити гострі елементи спеціальним захисним ковпачком [23].

7.3 Заходи з виробничої санітарії та гігієни праці

У приміщеннях, де встановлені персональні комп'ютери, повинні дотримуватися визначені параметри мікроклімату. У Державних санітарних нормах 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень» установлені величини параметрів мікроклімату, що створюють комфортні умови. Ці норми встановлюються в залежності від часу року, характеру трудового процесу і характеру виробничого приміщення і становлять:

– оптимальні норми температури повітря в робочій зоні виробничих приміщень у холодний і перехідний періоди року при II категорії робіт – 22...24°C, відносна вологість – 60...40%, швидкість руху повітря – 0.2 м/сек.

– у теплий період року температура – 23...25°C, відносна вологість – 60...40%, швидкість руху повітря – 0.2 м/сек [15].

У виробничому приміщенні застосовується водяна система опалення, яка рівномірно прогріває повітря, при цьому відсутнє виділення запахів гару, таке опалення відповідає основним санітарно-гігієнічним вимогам.

Робота у приміщенні пов'язана з розумовою працею. Тому для попередження стомлення і підвищення працездатності цих осіб визначено найбільш раціональний режим роботи і відпочинку.

На працівників впливає рівень шумів. Допустимий рівень шуму в приміщенні становить 50 – 71 дБ згідно Державних санітарних норм 3.3.6-037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку» [24].

На робочих місцях в приміщеннях шум створюється технічними засобами, від зовнішніх джерел шуму (вентиляторів, кондиціонерів) і ін.

Зниження шуму здійснюється наступними методами:

- зменшенням шуму в джерелі;
- раціональним плануванням приміщення;
- акустичною обробкою приміщення.

Відповідно, зменшення шуму у приміщенні для роботи досягається збільшенням звукоізоляції конструкцій, ущільненням по периметру притворів вікон, дверей [23].

Окрім цього на злюдину впливає якість освітлення. При проєктуванні освітлювальної установки вирішуються такі питання:

- вибір типу джерела світла;
- визначення системи освітлення;
- вибір типу світильників з урахуванням умов середовища;
- розподіл світильників і визначення їх кількості;

– визначення норми освітленості на робочому місці.

У приміщенні, де виконується робота, застосовується комбіноване освітлення (природне і штучне). Найбільш прийнятним для подібних приміщень є використання люмінесцентних ламп білого світла.

Згідно Державних будівельних норм В.2.5-28:2018 «Природне і штучне освітлення» для найменшого розрізнення об'єктів (розряд зорової роботи III (б)) 0,3–0,5 мм значення коефіцієнта природного освітлення повинно дорівнювати 1,5%. Метою розрахунку штучного освітлення є перевірка його відповідності ДБН В.2.5-28:2018 [16].

Для забезпечення нормальних санітарно-гігієнічних умов у відповідності до вимог Державних будівельних норм В.2.-67:2013 «Опалення, вентиляція та кондиціювання» [25] в приміщеннях передбачається загально обмінна припливно-витяжна механічна вентиляція і кондиціювання повітря. Основним завданням кондиціювання повітря є підтримка параметрів повітряного середовища в допустимих межах, що забезпечують надійну роботу ЕОМ, тривале зберігання носіїв інформації та комфортні умови для обслуговуючого персоналу. Кондиціювання здійснюється побутовими кондиціонерами, встановленими в вікнах. Окрім цього приміщення обладнане вентиляцією.

Кількість вентиляційного повітря визначається за формулою (7.1):

$$V_{\text{вент}} = \frac{3600 \cdot Q_{\text{надл}}}{C \cdot \rho \cdot (t_{\text{вуд}} - t_{\text{пр}})}, \quad (7.1)$$

де $Q_{\text{надл}}$ – виділення в приміщенні явного тепла, Вт;

C – теплоємність повітря ($C = 1000 \text{ Дж} / \text{кг} \cdot \text{гр}$);

ρ – питома щільність повітря ($\rho = 1,3 \text{ кг} / \text{м}^3$);

$t_{\text{вуд}}$, $t_{\text{пр}}$ – температури повітря, що видаляється та приточного, $^{\circ}\text{C}$.

Температура повітря, що видаляється визначається з формули (7.2):

$$t_{\text{вуд}} = t_{\text{зн}} + d(h - 2), \quad (7.2)$$

де $t_{\text{зн}}$ – температура повітря в зоні праці ($t_{\text{зн}} = 21^{\circ}\text{C}$);

d – коефіцієнт наростання температури на кожен метр висоти ($d = 1,5^{\circ}\text{C}/\text{м}$);

h – висота приміщення ($h = 5 \text{ м}$).

Звідси:

$$t_{\text{вуд}} = 21 + 1,5(5 - 2) = 25,5^{\circ}\text{C}.$$

Кількість надлишкового тепла визначається з теплового балансу, як різниця між теплом, що надходить у приміщення і теплом, що видаляється з приміщення і що поглинається в ньому, за формулою (7.3):

$$Q_{\text{надл}} = Q_{\text{надх}} - Q_{\text{випр}}. \quad (7.3)$$

Тепло, що надходить у приміщення, визначається за формулою (7.4):

$$Q_{\text{надх}} = Q_{\text{устатк}} + Q_{\text{л}} + Q_{\text{осв}} + Q_{\text{рад}}, \quad (7.4)$$

де $Q_{\text{устатк}}$ – тепло від роботи устаткування, $^{\circ}\text{C}$;

$Q_{\text{л}}$ – тепло, що надходить від людей, $^{\circ}\text{C}$;

$Q_{\text{осв}}$ – тепло від джерел освітлення, $^{\circ}\text{C}$;

$Q_{\text{рад}}$ – тепло від сонячної радіації, що надходить через вікна, $^{\circ}\text{C}$.

Відповідно за формулою (7.5):

$$Q_{\text{устатк}} = E_{\text{устатк}} \cdot P_{\text{уст}} = 0,15 \cdot 750 = 112,5 \text{ Вт}, \quad (7.5)$$

де $E_{уст}$ – частка енергії, що переходить у тепло, Дж;

$P_{уст}$ – потужність установки, Вт.

Тепло, що надходить від людей, розраховується за формулою (7.6):

$$Q = n \cdot q = 24 \cdot 90 = 2160 \text{ Вт}, \quad (7.6)$$

де n – кількість осіб у залі ($n = 24$);

q – кількість тепла, що виділяється людиною ($q = 90 \text{ Вт}$).

Відповідно за формулою (7.7):

$$Q_{осв} = E_{осв} \cdot P_{осв} = 0,4 \cdot 305 = 122 \text{ Вт}, \quad (7.7)$$

де $E_{л} = 0,4$ для люмінесцентних ламп;

$P_{осв}$ – потужність освітлювальної установки, Вт.

Тепло від сонячної радіації, що надходить через вікна, розраховується за формулою (7.8):

$$Q_{рад} = A \cdot k \cdot S \cdot m = 180 \cdot 0,8 \cdot 4,5 \cdot 3 = 1944 \text{ Вт}, \quad (7.8)$$

де A – теплонадходження в приміщення з 1 м^2 скла ($127 - 234 \text{ Вт/м}^2$);

S – площа вікна ($S = 4,5 \text{ м}^2$);

m – кількість вікон ($m = 3$);

k – коефіцієнт, що враховує характер скління ($k = 0,8$).

Тоді

$$Q_{надх} = 112,5 + 2160 + 122 + 1944 = 4338,5 \text{ Вт},$$

$$Q_{вигр.} = 0,1 \cdot Q_{надх} = 0,1 \cdot 4338,5 = 433,85 \text{ Вт}.$$

Звідси за формулою (7.3) знаходимо:

$$Q_{\text{надл}} = 4338,5 - 433,85 = 3904,65 \text{ Вт}.$$

Знаходимо необхідний повітрообмін:

$$V_{\text{вент}} = \frac{3600 \cdot 3904,65}{1000 \cdot 1,3 (25,5 - 18)} = \frac{14056740}{9750} = 1441,7 \text{ м}^3/\text{год}.$$

Оптимальним варіантом є кондиціонування повітря, тобто автоматична підтримка його стану в приміщенні відповідно до визначених вимог (задана температура, вологість, рухливість повітря) незалежно від зміни стану зовнішнього повітря й умов у самому приміщенні.

Визначаємо необхідну кратність повітрообміну за формулами (7.9, 7.10):

$$K = \frac{V_{\text{вент}}}{V_{\text{прим}}}, \quad (7.9)$$

$$V_{\text{прим}} = n \cdot S_{\text{ос}} \cdot h, \quad (7.10)$$

де n – число осіб у приміщенні;

$S_{\text{ос}}$ – площа виробничого приміщення, що приходить на 1 особу (за нормами для розумово-фізичної праці $S_{\text{ос}} = 6 \text{ м}^2$);

h – висота приміщення, м.

Кратність повітрообміну для даного приміщення складає:

$$K = \frac{1441,7}{24 \cdot 6 \cdot 5} = 2.$$

7.4. Заходи безпеки у надзвичайних ситуаціях

7.4.1 Заходи з пожежної безпеки

Заходи з протипожежної безпеки розроблені для виробничого приміщення обладнаного ПК з ВДТ розроблені згідно вимог НАПБ А.01.001-2014 «Правила пожежної безпеки в Україні» [26].

Виходячи з аналізу речовин та матеріалів, які використовуються при роботі у приміщенні, обладнаному ПК з ВДТ.

Відповідно до вимог ДСТУ Б В.1.1-36:2016 «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою» [27], воно належить до категорії «Д» з пожежної небезпеки – простір у приміщенні, у якому перебувають тверді горючі речовини та матеріали, у приміщенні, обладнаному ПК з ВДТ можлива пожежа класу – А та Е;.

Оскільки приміщення належить до категорії «Д» з пожежної небезпеки, тому відповідно до вимог ДБН В.1.1-7:2016 «Пожежна безпека об'єктів будівництва. Загальні вимоги» [19] воно має II ступінь вогнестійкості.

У разі виникнення пожежі у приміщенні обладнаному ПК з ВДТ для евакуації персоналу відповідно до вимог ДБН В.1.1-7:2016 «Пожежна безпека об'єктів будівництва. Загальні вимоги» передбачені виходи, по обидві сторони приміщення, з одного боку вікно (на пожежні сходи), а з іншого – вхідні двері.

Обладнання, силові та освітленні мережі приміщення, обладнаного ПК з ВДТ відповідають вимогам пожежної безпеки, оскільки виконані відповідно до вимог НПАОП 40.1-1.32-01 «Правила будови електроустановок. Електрообладнання спеціальних установок» [28], та мають ступінь захисту ізоляції обладнання IP44 яка відповідає класу пожежанебезпечної зони П-Па до якої належить приміщення.

Для виявлення початкової стадії загоряння, швидкого й точного оповіщення служби пожежної охорони про час і місце виникнення пожежі встановлені системи автоматичної пожежної сигналізації (АПС). Вони можуть

самостійно пускати в хід установки пожежогасіння, коли пожежа не досягла великих розмірів. Системи АПС складаються з пожежних сповіщувачів, ліній зв'язку й прийомних пультів (станцій). Згідно вимог ДБН В.2.5-56:2014 «Системи протипожежного захисту», в приміщенні обладнаному ПК з ВДТ встановлена система пожежної й охоронної сигналізації «Фотон-ОФІС», яка забезпечує виявлення теплових і димових ознак пожежі і місця виникнення пожежі з точністю до місця розміщення датчика.

Оскільки приміщення (дослідницької лабораторії, конструкторського бюро, тощо) що обладнане ПК з ВДТ має площу 375 м², тому відповідно до вимог п. 3.8 розділу «Типові норми належності вогнегасників» ДСТУ 4297:2004 «Пожежна техніка. Технічне обслуговування вогнегасників. Загальні технічні вимоги» [29] для гасіння електроустановок, що знаходяться під напругою, передбачені вуглекислотні вогнегасники типу ВВК-3,5 у кількості 19 штук (з розрахунку один вогнегасник с величиною заряду вогнегасної речовини 3,5 кг і більше, на 20 м² площі приміщення). Відстань між вогнегасниками та місцями можливих загорянь не перевищує 10 м.

Перезарядка і ремонт вогнегасників виробляються в спеціалізованих організаціях на зарядних станціях. Експлуатація вогнегасників без чеки і пломби заводу-виготовлювача або організації, що проводила перезарядку, не дозволяється.

На стенді у приміщенні знаходиться інструкція щодо використання вогнегасників та дій при пожежі. Також робітники повинні пройти тестування з техніки безпеки, перед тим як приступити до роботи. Такі тестування проводяться регулярно.

На випадок пожежі передбачені шляхи евакуації працюючих- проходи, проїзди, евакуаційні виходи.

Щодо самої будівлі, то коридори й проходи, призначені для евакуації, мають найменшу довжину й мінімальну кількість поворотів. На всьому протязі проходу немає порогів або проміжних щаблів. Також місця евакуації позначені відповідними знаками та мають підсвітку.

Окрім цього будівля об'єкта обладнана пожежною сигналізацією, що спрацьовує в разі пожежі. Також в будівлі проводяться тестування сигналізації та виконуються тренувальні евакуації для робітників.

Важливу роль у забезпеченні безпечного виходу людей грає протидимовий захист евакуаційних ходів. У будинках висотою до 9 поверхів незадимлюваність сходових кліток на час евакуації досягається їхньою ізоляцією від підвалів, горищ і поверхів. Для цього влаштовані відособлені входи в підвали, вхід на сходову клітку з поверхів здійснюють через тамбур - шлюз із підпором повітря, відокремлюють горища від сходових кліток перекриттями з негорючих матеріалів.

7.4.2 Заходи з цивільного захисту

Дії виробничого персоналу об'єкта при можливих аваріях, катастрофах і стихійних лихах. Заходи з ліквідації наслідків і захисту персоналу.

Стихійні дії сил природи, поки ще не в повній мірі підвладні людині, наносять економіці держави і населенню величезний збиток. Стихійні лиха - такі явища природи, які викликають екстремальні ситуації, порушують нормальну життєдіяльність людей і роботу об'єктів.

Великі аварії і катастрофи на об'єктах можуть виникати в результаті стихійного лиха, а також порушення технології виробництва, правил експлуатації різних машин, обладнання та встановлених заходів безпеки. Їх впливу подібні стихійних лих.

Під аварією розуміють раптову зупинку роботи або порушення процесу виробництва на промисловому підприємстві, транспорті, інших об'єктах, що призводять до пошкодження або знищення матеріальних цінностей.

Під катастрофою розуміють раптове лихо; подія, що несе за собою трагічні наслідки. Катастрофи супроводжуються руйнуванням будівель, різних споруд, знищенням матеріальних цінностей і загибеллю людей.

Найбільш повне і організоване виконання заходів цивільного захисту на

об'єкті досягається завчасної розробкою плану заходів, які необхідно провести при виникненні загрози.

При ліквідації наслідків стихійних лих, великих аварій і катастроф передбачається організація оповіщення робітників і службовців працюючої зміни, формувань і населення відомчих будинків про загрозу виникнення або виникли аварії та стихійні лиха. Зазвичай для цього використовуються мережі внутрішнього радіомовлення, телефонної і диспетчерського зв'язку. Передбачаються заходи щодо організації та проведення евакуації робітників, службовців з цехів (підрозділів) об'єкта, а населення - з відомчих будинків, вказуються маршрути і пункти евакуації, порядок видачі засобів індивідуального захисту особового складу формувань, робочим, службовцям цехів (підрозділів). Визначаються сили і засоби для проведення робіт з ліквідації наслідків великих аварій і катастроф, порядок управління, сили і засоби зв'язку, щоб забезпечити управління. Крім того, передбачають організацію харчування особового складу формувань цивільного захисту і постраждалого населення.

При загрозі для життя виробляють евакуацію персоналу. Для евакуації людей використовують основні та запасні входи і виходи, віконні прорізи і балкони, люки в перекриттях, іноді спеціально проробляють отвори в перегородках і стінах [30].

Ліквідацію аварій на водопровідних мережах роблять у два етапи. Спочатку їх локалізують, а потім відновлюють пошкоджені ділянки. Локалізація аварій дозволяє обмежити або запобігти розмиву ґрунту, доріг, затоплення підвальних і інших приміщень.

Серед захисних заходів цивільної оборони, здійснюваних завчасно, особливо важливе місце займає організація оповіщення органів цивільного захисту, формувань і населення про загрозу нападу противника і про застосування ним ядерної, хімічної, бактеріологічної (біологічної) зброї і інших сучасних засобів нападу. Особливе значення оповіщення стоїть у разі раптового нападу противника, коли реальний час для попередження населення буде вкрай

обмеженим і обчислюватися хвилинами.

З метою своєчасного попередження населення міст і сільських населених пунктів про виникнення безпосередньої небезпеки застосування супротивником ядерного, хімічного, бактеріологічного (біологічного) або іншої зброї і необхідності застосування заходів захисту встановлені наступні сигнали оповіщення цивільної оборони: «Повітряна тривога» «Відбій повітряної тривоги», «Радіаційна небезпека»; «Хімічна тривога».

Сигнал «Повітряна тривога» подається для всього населення. Він попереджає про безпосередню небезпеку ураження противником даного (району). За радіотрансляційної мережі передається текст: «Увага! Увага! Громадяни! Повітряна тривога! Повітряна тривога!» Одночасно з цим сигнал дублюється звуком сирен, гудками дзвінків і транспортних засобів. На об'єктах сигнал буде дублюватися усіма, наявними в їх розпорядженні засобами. Тривалість сигналу 2-3 хвилини.

За цим сигналом об'єкти припиняють роботу, транспорт зупиняється і все населення ховається в захисних спорудах. Робітники і службовці припиняють роботу відповідно до встановленої інструкцією і вказівками адміністрації, що виключають виникнення аварій.

Сигнал «Радіаційна небезпека» подається в населених пунктах і районах, у напрямку до яких рухається радіоактивна хмара, що утворилася при вибуху ядерного боєприпасу.

За сигналом «Радіаційна небезпека» необхідно надягнути респіратор, протипилову тканинну маску і ватна-марлеву пов'язку, а при їх відсутності - протигаз, взяти підготовлений запас продуктів, індивідуальні засоби медичного захисту, предмети першої необхідності і піти в притулок, протирадіаційне або найпростіше укриття.

Сигнал «Хімічна тривога» подається при загрозі або безпосередньому виявленні хімічного або бактеріологічного нападу (зараження). З цього приводу сигналу необхідно швидко надіти протигаз, а в разі необхідності - засоби захисту шкіри і при першій же можливості сховатися в захисній споруді.

Якщо захисної споруди поблизу не виявиться, то в разі ураження аерозолями отруйних речовин і бактеріальних засобів можна сховатися в житлових, виробничих або підсобних приміщеннях.

Якщо буде встановлено, що супротивник застосував бактеріологічна (біологічна) зброя, то по системам оповіщення населення отримає рекомендації про подальші дії [30].

Необхідно бути гранично уважними і строго виконувати розпорядження органів цивільної оборони. Про те, що небезпека нападу супротивника минула, і про порядок подальших дій розпорядження надійде по тих же каналах зв'язку, що і сигнал оповіщення.

При виникненні загрози нападу противника місцевими органами влади і штабами цивільного захисту за допомогою засобів масової інформації передаються населенню постанови чи розпорядження про порядок дій. З цього часу радіо, телевізори повинні бути постійно включені для прийому нових повідомлень. У найкоротші терміни населення має вжити необхідних заходів захисту і включитися у виконання заходів, що проводяться для цивільного захисту.

ВИСНОВКИ

Під час виконання магістерської дипломної роботи були досліджені основні методи управління проєктами, а саме Scrum, Kanban та Waterflow. Була розроблена програмна система для спрощення та автоматизації управління проєктами різними методами. Також було виконано огляд та аналіз існуючих методів вирішення завдання та існуючих програмних засобів.

Практична цінність розробки програмної системи полягає в можливості застосування її на підприємстві для спрощення та автоматизації управління проєктами різними методами.

Наукова новизна полягає у тому, що вперше, на основі дослідження методів управління проєктами та їх практичної реалізації, був модифікований підхід до створення подібних сервісів шляхом реалізації вибору метода управління проєктами та реалізації свого варіанту керування проєктами «Вільна дошка».

При розробці програмного продукту був використаний механізм моделей та компонентів. Було розроблено та детально описано алгоритми, які відповідають тим чи іншим функціональним частинам програми, що допомагає організувати досить швидку реалізацію мобільного вебсайту. А саме авторизація користувачів, створення дошок методом Scrum, створення дошок методом Kanban, створення дошок методом Waterflow, створення дошок вільним методом, створення колонок та карток, отримання дошок, колонок та карток з усіма деталями, редагування колонок та карток, видалення дошок, колонок та карток, сортування карток, та інші.

Для реалізації програмного продукту було використано такі мови програмування, як JavaScript, TypeScript, HTML та CSS. Також при розробці використовувався SDK для створення гібридних мобільних застосунків – Ionic та набір серверного програмного забезпечення MEAN. Були описані особливості перерахованих мов програмування і засобів та їх переваги.

Був розрахований та детально описаний економічно ефективний фінансовий план по впровадженню системи для спрощення та автоматизації управління проєктами різними методами.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Управление проектами [Текст] / Харпер - Смит П. – М.: ДиС, 2011. – 240 с.
2. Методы управления проектами [Электрон. ресурс]. – Режим доступа: <https://4brain.ru/project/methods.php>.
3. WATERFALL МЕТОДОЛОГИЯ РАЗРАБОТКИ [Электрон. ресурс]. – Режим доступа: <https://qaevolution.ru/metodologiya-menedzhment/waterfall/>.
4. Scrum. Революционный метод управления проектами [Текст]/ Джефф Сазерленд. – М.: Манн, Иванов и Фербер, 2020. — 288 с.
5. Что такое Jira? [Электрон. ресурс]. – Режим доступа: <https://softlist.com.ua/articles/chto-takoe-jira/>.
6. Как пользоваться Trello: инструкция и полезные хитрости [Электрон. ресурс]. – Режим доступа: <https://blog.calltouch.ru/kak-polzovatsya-trello-instruktsiya-i-poleznye-hitrosti/>.
7. Asana [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Asana>.
8. Сравнение языков программирования [Электрон. ресурс]. – Режим доступа: <https://habr.com/ru/post/280041/>
9. JavaScript [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/JavaScript>.
10. Основы HTML [Электрон. ресурс]. – Режим доступа: https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics.
11. CSS [Электрон. ресурс]. – Режим доступа: <https://wiki.rookee.ru/css/>.
12. Начало работы с ионной каркасом [Электрон. ресурс]. – Режим доступа: <https://riptutorial.com/ru/ionic-framework>.
13. Саймон Холмс. Стек MEAN. Mongo, Express, Angular, Node = Getting MEAN with Mongo, Express, Angular, and Node [Текст] / Саймон Холмс. – М.: КУДИЦ-Пресс, 2016. — 496 с.

14. Богданов В. Управление проектами. Корпоративная система - шаг за шагом. – М.: Манн, Иванов и Фербер, 2012. – 248 с.
15. Санітарні норми мікроклімату виробничих приміщень: ДСН 3.3.6.042-99. – [Чинний від 1999-01-03]. – Х.: Форт, 2000. – 24 с.
16. Природне та штучне освітлення. посібник: проблеми природного і штучного освітлення: ДБН В.2.5-28:2018. – [Чинний від 2018-02-28]. – К.: Етін, 2019. – 180 с.
17. Державні санітарні норми і правила при роботі з джерелами електромагнітних полів: ДСанПіН 3.3.6.096-2002. – Офіційний вісник України від 07.09.2009. – 2009 р., № 66. – 11 с.
18. Система стандартов безопасности труда. Электростатические поля. Допустимые уровни на рабочих местах и требования к проведению контроля: ГОСТ 12.1.045-84. – [Введен с 2001-07-18]. – М.: Стандартинформ, 2006. – 7 с.
19. Пожежна безпека об'єктів будівництва. Загальні вимоги: ДБН В.1.1-7:2016. – [Чинний від 2016-04-01]. – К.: Міністерство регіонального розвитку, будівництва та житлово-комунального господарства України, 2017. – 39 с.
20. Електробезпека в будівлях і спорудах. Вимоги до захисних заходів від ураження електричним струмом: ДСТУ Б В.2.5-82:2016. – [Чинний від 2016-09-22]. – К.: ДП "УкрНДНЦ", 2016. – 109 с.
21. Правила улаштування електроустановок [Текст] : ПУЕ-2017. – На заміну ПУЕ-86. – [Чинний від 2017-08-21]. – К.: Міненерговугілля України, 2017. – 617 с.
22. Основи професійної безпеки та здоров'я людини / під ред. проф. В.В. Березуцького : підручник / [В.В. Березуцький та ін.]. – Харків : НТУ "ХП", 2018. – 553 с.
23. Санітарні норми виробничого шуму, ультразвуку та інфразвуку: ДСН 3.3.6.037-99. – [Чинний від 1999-01-07]. – К.: МОЗ України, 1999. – 24 с.

24. Опалення, вентиляція та кондиціонування: 1ДБН В.2.5-67:2013. – [Чинний від 2013-03-11]. – К.: Мінрегіон України, 2013. – 240 с.
25. Правила пожежної безпеки в Україні: Харків : Форт, 2015. – 124 с.
26. Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою: ДСТУ Б В.1.1-36:2016. – [Чинний від 2016-02-06]. – К.: Мінрегіонбуд України, 2016. – 66 с.
27. Правила будови електроустановок. Електрообладнання спеціальних установок: НПАОП 40.1-1.32-01. – [Чинний від 2006-07-16]. – Х.: Індустрія, 2010. - 118 с.
28. Пожежна техніка. Технічне обслуговування вогнегасників. Загальні технічні вимоги: ДСТУ 4297:2004. – [Чинний від 2004-04-01]. – К.: Мінрегіонбуд України, 2019. – 48 с.
29. Стеблюк М.І. Цивільна оборона та цивільний захист [Текст] : навч. посіб. для вузів / М.І. Стеблюк. – К.: Знання, 2013, – 487 с.
30. Комп'ютерні науки, інформаційні технології та інженерія програмного забезпечення : навчальний посібник / під заг. ред. С.О. Субботіна. – Т.1 Виконання, оформлення та захист випускних робіт бакалавра та атестаційних робіт магістра / [С. О. Субботін, С. К. Корнієнко, А. О. Олійник та ін.]. – Запоріжжя: НУЗП, 2020. – 139 с.
31. Analysis and design of intelligent systems using soft computing techniques / [eds.: P. Melin, O.R. Castillo, E.G. Ramirez, J. Kacprzyk]. – Heidelberg: Springer, 2007. – 855 p.
32. Автоматизовані системи. Терміни та визначення : ДСТУ 2226-93. – [Чинний від 1993-04-01]. – К.: Держстандарт України, 1993. – 86с. – (Національний стандарт України).
33. Управление проектами. [Текст] / Ивасенко А Г., Никонова Я.И., Каркавин М.В. – М., Феникс, 2011. – 336 с.

ДОДАТОК А
Текст програми

Вибір методу управління проєктами

Інтерфейсна частина

choose-board-by-type-modal.component.ts

```

public checkedBoardType: BoardTypes = BoardTypes.Default;
public boardTypes: {type: BoardTypes, infoText: string}[] = [
  {
    type: BoardTypes.Default,
    infoText: 'Вибираючи Вільний метод, ви маєте можливість створювати
будь-які колонки і перемішуватися вільно між ними.'
  },
  {
    type: BoardTypes.Kanban,
    infoText: 'Дотримуючись моделі Kanban, ви маєте можливість вільно
пересуватися між стадіями розробки та призупинити виконання завдання на будь-
якому етапі в разі, якщо з\'явилися інші термінові завдання або змінився
пріоритет поточної.'
  },
  {
    type: BoardTypes.Scrum,
    infoText: 'Дотримуючись методу Scrum, ви маєте можливість створювати
спринти та задачі у них. Проєктна команда може почати наступний спринт тільки
коли попередній закінчений.',
  },
  {
    type: BoardTypes.Waterflow,
    infoText: 'Дотримуючись моделі Waterflow, ви маєте можливість
переходити від однієї стадії до іншої строго послідовно.'
  },
];

constructor(public modalController: ModalController, public
popoverController: PopoverController) {

  public dismiss() {
    this.modalController.dismiss();
  }

  public chooseBoardType(boardType: BoardTypes): void {
    this.checkedBoardType = boardType;
  }

  public submit() {
    this.modalController.dismiss(this.checkedBoardType);
  }

  async showInfoPopover(ev: Event, boardInfoText: string) {
    ev.stopPropagation();
    const popover = await this.popoverController.create({
      component: BoardTypeInfoPopoverComponent,

```

```

        event: ev,
        translucent: false,
        componentProps: {
            text: boardInfoText,
        },
    });
    return await popover.present();
}

```

Створення дошки

Інтерфейсна частина

create-board-modal.component.ts

```

@Input() boardType: BoardTypes;
public boardName = '';
public selectedUsers = [];
private subs = new SubSink();

constructor(public modalController: ModalController,
             private boardService: BoardService,
             private toastService: ToastService,
             private loaderService: LoaderService) {

}

ngOnDestroy() {
    this.subs.unsubscribe();
}

public dismiss() {
    this.modalController.dismiss();
}

public onUserSelected(selectedUser): void {
    this.selectedUsers = [...this.selectedUsers, selectedUser];
}

public removeSelectedUser(user): void {

```

```

        this.selectedUsers = this.selectedUsers.filter(item => item._id !==
user._id);
    }

    public async createBoard() {
        await this.loaderService.presentLoading('Сохранение...');

        this.subs.sink = this.boardService.createBoard({name:
this.boardName, users: this.selectedUsers.map(item => item._id), type:
this.boardType})
        .pipe(finalize(() => this.loaderService.dismissLoading()))
        .subscribe((response: Board) => {
            this.dismiss();
            this.toastService.presentToast('Доска успешно создана');
        }, (error) => {
            this.toastService.presentErrorToast();
        }));
    }
}

```

Створення дошки

Серверна частина

board.controller.js

```

async function createBoard(req, res) {
    const users = req.body.users || [];
    const type = req.body.type;

    const boardData = {
        ...req.body,
        user: req.user._id,
        owner: req.user._id,
        users: [...users, req.user._id],
        notifiedUsers: [req.user._id],
    };
    const newBoard = new Board(boardData);
    let savedBoard = await newBoard.save();
}

```

```

    if (type) {
      await addColumnsByBoardType(savedBoard.type, savedBoard._id,
req.user._id);
      savedBoard = await Board.findById(savedBoard._id)
        .populate({
          path: 'columns',
          populate: {path: 'cards', options: {sort: 'position'}}
        })
        .populate({
          path: 'users',
        });
    }

    res.json(savedBoard);
  }

```

board.model.js

```

const BoardSchema = Schema({
  user: { type: Schema.Types.ObjectId, ref: 'User', required: true},
  users: [{ type: Schema.Types.ObjectId, ref: 'User', required: true}],
  notifiedUsers: [{ type: Schema.Types.ObjectId, ref: 'User', required:
false}],
  owner: { type: Schema.Types.ObjectId, ref: 'User', required: true},
  columns: [{ type: Schema.Types.ObjectId, ref: 'Column' }],
  type: {
    type: String,
    enum : ['', 'scrum', 'kanban', 'waterflow'],
    default: '',
  },
  name: {
    type: String,
    required: true,
    maxlength: 40,
    minlength: 1
  }

```

```

    },
    createdAt: {
      type: Date,
      default: Date.now
    }
  });

```

Додавання користувача до дошки

Інтерфейсна частина

add-user-to-board-modal.component.ts

```

public selectedUsers = [];
private subs = new SubSink();

@Input() board: Board;

constructor(public modalController: ModalController,
             private boardService: BoardService,
             private toastService: ToastService,
             private loaderService: LoaderService) {

}

ngOnDestroy() {
  this.subs.unsubscribe();
}

get excludedUsers(): User[] {
  return [...this.board.users, ...this.selectedUsers];
}

public dismiss(data?) {
  this.modalController.dismiss(data);
}

public onUserSelected(selectedUser): void {
  this.selectedUsers = [...this.selectedUsers, selectedUser];
}

```

```

    }

    public removeSelectedUser(user): void {
        this.selectedUsers = this.selectedUsers.filter(item => item._id !==
user._id);
    }

    public async submit() {
        await this.loaderService.presentLoading('Добавление...');

        this.subs.sink = this.boardService.addUsersToBoard(this.board._id,
this.selectedUsers.map(item => item._id))
            .pipe(finalize(() => this.loaderService.dismissLoading()))
            .subscribe((response: User[]) => {
                this.dismiss(response);
                this.toastService.presentToast('Пользователи успешно добавлены');
            }, (error) => {
                this.toastService.presentErrorToast();
            });
    }
}

```

Додавання користувача до дошки

Серверна частина

board.controller.js

```

async function addUsersToBoard(req, res) {
    const board = await Board
        .findOneAndUpdate({_id: req.params.boardId}, {$push: {users: {$each:
req.body.users}}}, {new: true})
        .populate('users')
        .exec();

    board.users.forEach((user) => {
        const socketId = global.socketUsers.find(item => item.userId ===
user._id);
    });
}

```

```

        if (socketId) {

global.io.to(socketId).emit(boardSocketChannelNames.attachedToBoardChannelName,
{board});

        }
    })

    res.json(board.users);
}

```

Додавання користувача до картки

Інтерфейсна частина

add-user-to-card-modal.component.ts

```

public selectedUsers = [];
private subs = new SubSink();

@Input() card: Card;

constructor(public modalController: ModalController,
             private cardService: CardService,
             private toastService: ToastService,
             private loaderService: LoaderService) {

}

ngOnDestroy() {
    this.subs.unsubscribe();
}

get excludedUsers(): User[] {
    return [...this.card.users, ...this.selectedUsers];
}

public dismiss(data?) {
    this.modalController.dismiss(data);
}

```

```

    }

    public onUserSelected(selectedUser): void {
        this.selectedUsers = [...this.selectedUsers, selectedUser];
    }

    public removeSelectedUser(user): void {
        this.selectedUsers = this.selectedUsers.filter(item => item._id !==
user._id);
    }

    public async submit() {
        await this.loaderService.presentLoading('Добавление...');

        this.subs.sink = this.cardService.addUsersToCard(this.card._id,
this.selectedUsers.map(item => item._id))
            .pipe(finalize(() => this.loaderService.dismissLoading()))
            .subscribe((response: Card) => {
                this.dismiss(response);
                this.toastService.presentToast('Пользователи успешно добавлены');
            }, (error) => {
                this.toastService.presentErrorToast();
            });
    }
}

```

Додавання користувача до картки

Серверна частина

card.controller.js

```

async function addUsersToCard(req, res) {
    const oldCardData = await
Card.findById(req.params.cardId).populate('users');
    const oldUsers = oldCardData.users.map(item => item._id);

    let card = await Card

```

```

        .findOneAndUpdate({_id: req.params.cardId}, {$push: {users: {$each:
req.body.users}}}, {new: true})
        .populate('users')
        .exec();

const      addedUserNames      =      card.users.filter(item      =>
!oldUsers.includes(item._id)).map(item => item.fullname).join(', ');

card = await Card
    .findOneAndUpdate({_id: req.params.cardId}, {
        $push: {actions: {message: `Пользователь  ${req.user.fullname}
добавил ${addedUserNames} к карточке.`}}
    }, {new: true})
    .populate({
        path: 'notes',
        populate: {path: 'owner'}
    })
    .populate('users')
    .populate('column')
    .exec();
res.json(card);
}

```

Оцінка часу, потрібного для виконання задачі у картці

Інтерфейсна частина

card-add-estimate-time.component.ts

```

public logTimeValues: { value: CardLogTimeSuffixType; viewValue: string;
}[] = [...logTimeValues];
public estimateTimeForm: FormGroup;
private subs = new SubSink();

@Input() card: Card;

constructor(public modalController: ModalController,
            private cardService: CardService,

```

```

        private fb: FormBuilder,
        private toastService: ToastService,
        private loaderService: LoaderService) {
    }

    ngOnInit() {
        this.estimateTimeForm = this.fb.group({
            value: [this.card.estimateTime ? this.card.estimateTime.value :
null, Validators.min(0)],
            suffix: [this.card.estimateTime ? this.card.estimateTime.suffix :
this.logTimeValues[0].value, Validators.required],
        });
    }

    ngOnDestroy() {
        this.subs.unsubscribe();
    }

    public dismiss(data?) {
        this.modalController.dismiss(data);
    }

    public async submit() {
        await this.loaderService.presentLoading('Сохранение...');
        const data: CardEstimateTimeSubmitDataInterface = {
            ...this.estimateTimeForm.value,
        };

        this.subs.sink = this.cardService.estimateTime(this.card._id, data)
            .pipe(finalize(() => this.loaderService.dismissLoading()))
            .subscribe((response: Card) => {
                this.dismiss(response);
                this.toastService.presentToast('Время успешно сохранено');
            }, (error) => {
                this.toastService.presentErrorToast();
            });
    }
}

```

Оцінка часу, потрібного для виконання задачі у картці

Серверна частина

card.controller.js

```
function estimateTime(req, res) {
  Card
    .findOneAndUpdate({_id: req.params.cardId}, {
      estimateTime: {...req.body},
    }, {new: true})
    .findOneAndUpdate({_id: req.params.cardId}, {
      $push: {actions: {message: `Пользователь ${req.user.fullname}
добавил/изменил оценку - ${req.body.value}${req.body.suffix}.`}}
    }, {new: true})
    .populate({
      path: 'notes',
      populate: {path: 'owner'}
    })
    .populate('users')
    .populate('column')
    .exec((err, card) => {
      if (err) {
        res.status(404).send({message: 'Oh uh, something went wrong'});
      }
      res.json(card);
    })
}
```

Врахування часу, витраченого на виконання задачі у картці

Інтерфейсна частина

card-log-time.component.ts

```
public logTimeValues: { value: string; viewValue: string; }[] = [
  {
    value: 'м',
    viewValue: 'Минут',
  },
];
```

```

    },
    {
        value: 'h',
        viewValue: 'Часов',
    },
];
public logTimeForm: FormGroup;
private subs = new SubSink();

@Input() card: Card;

constructor(public modalController: ModalController,
             private cardService: CardService,
             private fb: FormBuilder,
             private toastService: ToastService,
             private loaderService: LoaderService) {

    ngOnInit() {
        this.logTimeForm = this.fb.group({
            date: [new Date().toISOString(), Validators.required],
            value: [null, Validators.min(0)],
            suffix: [this.logTimeValues[0].value, Validators.required],
        });
    }

    ngOnDestroy() {
        this.subs.unsubscribe();
    }

    public dismiss(data?) {
        this.modalController.dismiss(data);
    }

    public async submit() {
        await this.loaderService.presentLoading('Сохранение...');
        const data: CardLogTimeSubmitDataInterface = {

```

```

    ...this.logTimeForm.value,
    date: new Date(this.logTimeForm.value.date).toISOString(),
  });

  this.subs.sink = this.cardService.logTime(this.card._id, data)
    .pipe(finalize(() => this.loaderService.dismissLoading()))
    .subscribe((response: Card) => {
      this.dismiss(response);
      this.toastService.presentToast('Время успешно залогировано');
    }, (error) => {
      this.toastService.presentErrorToast();
    });
}

```

Врахування часу, витраченого на виконання задачі у картці

Серверна частина

card.controller.js

```

function logTime(req, res) {
  Card
    .findOneAndUpdate({_id: req.params.cardId}, {$push: {loggedTime:
    {...req.body}}}, {new: true})
    .findOneAndUpdate({_id: req.params.cardId}, {
      $push: {actions: {message: `Пользователь ${req.user.fullname}
      залогировал время - ${req.body.value}${req.body.suffix}.`}}
    }, {new: true})
    .populate({
      path: 'notes',
      populate: {path: 'owner'}
    })
    .populate('users')
    .populate('column')
    .exec((err, card) => {
      if (err) {
        res.status(404).send({message: 'Oh uh, something went wrong'});
      }
    })
  }

```

```
        res.json(card);  
    })  
}
```

ДОДАТОК Б
Слайди презентації

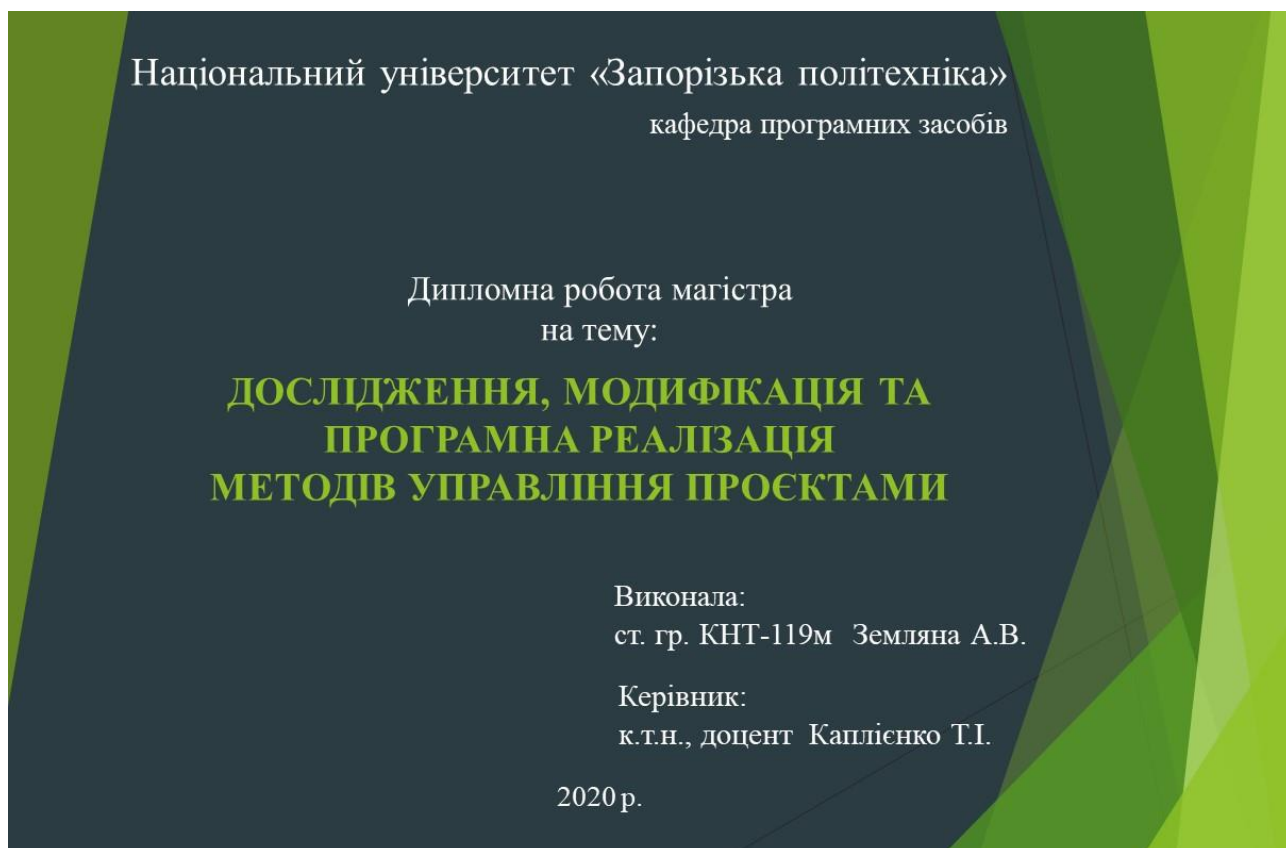


Рисунок Б.1 – Слайд №1

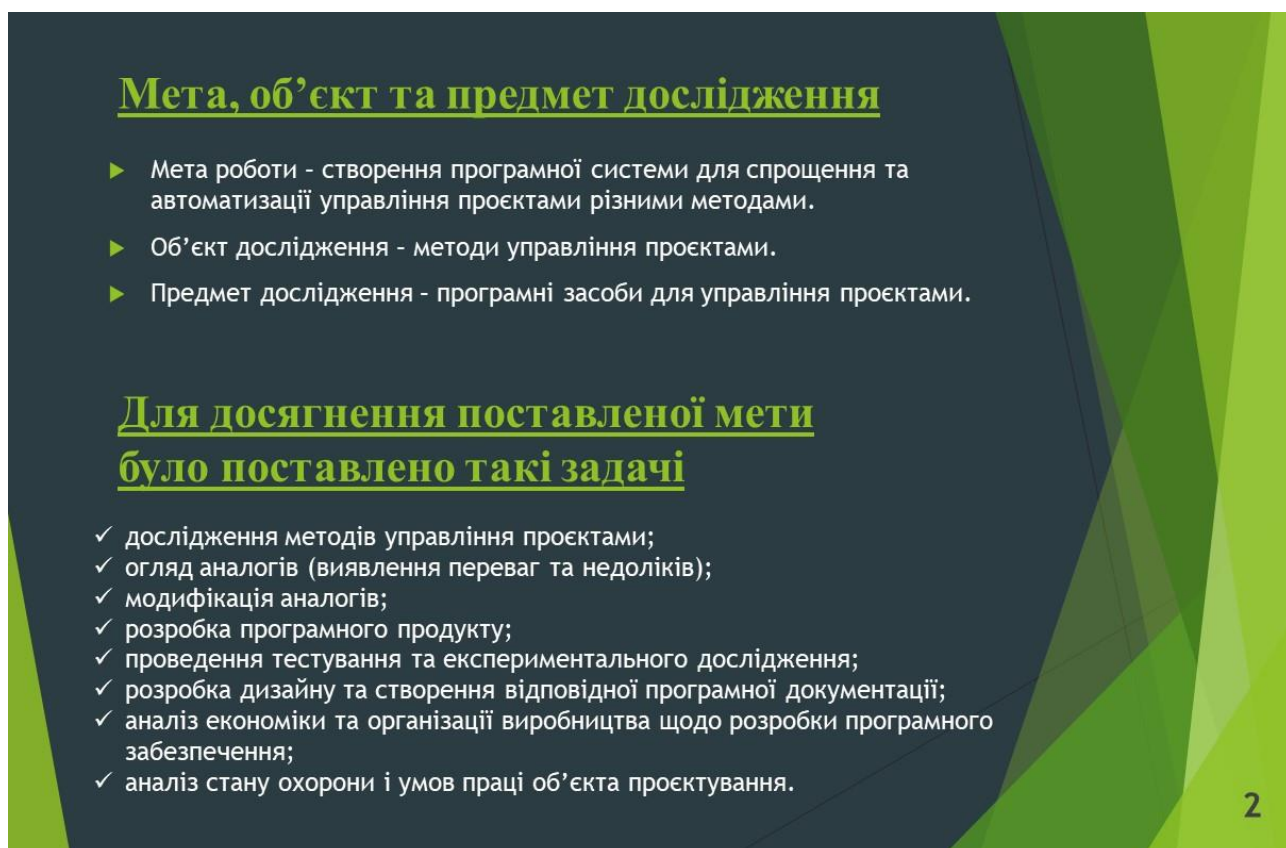


Рисунок Б.2 – Слайд №2

Аналіз предметної області

За довгий час існування управління проєктами фахівці створили досить багато стандартів і підходів. Всі вони відрізняються один від одного, а також розрізняються по формалізації, самодостатності, деталізації і сфері застосування.

Найпоширеніші існуючі методи управління проєктами:

- Scrum
- Kanban
- Waterfall



3

Рисунок Б.3 – Слайд №3

Огляд існуючих програмних засобів

Програмний засіб	Підходить для розроблення програмного забезпечення	Широка цільова аудиторія	Широкі функціональні можливості
Jira	+		+
Trello		+	
Asana		+	+

4

Рисунок Б.4 – Слайд №4

Таблиця порівняння мов програмування

МП	Доступність	Затребуваність	Спектр застосувань	Цілі
Python	Освоюється легко	1 місце	AI, ігри, Big Data, веб, mobile та інше.	Сайти і вебзастосунки, робота з даними
Java Script	Освоюється легко	2 місце	Node.js перетворює JavaScript в мову загального призначення. JS це і фронтенд, і бекенд. Підходить і в роботі тестувальника, QA-інженера.	Сайти і вебзастосунки
Swift	Освоюється трохи складніше	7 місце	Створення застосунків під Mac і iOS	Застосунки для Apple
Ruby	Складний	6 місце	Використовується в веброзробці	Сайти і вебзастосунки
PHP	Освоюється легко	5 місце	Популярна мова програмування для бекенд, на ньому пишуть серверну частину сайту.	Сайти і вебзастосунки
C/C++	Складний	4 місце	C / C ++ має широкий спектр застосувань. Зі знанням цих мов буде легко вчити щось нове.	Розібратися в програмуванні, робототехніка
Java	Освоюється трохи складніше	3 місце	Широке застосування: Android-застосунки, серверні застосунки, автоматизація тестування, веб, Big Data, веб- застосунки та інше.	Робота з даними, застосунки для Android, вебзастосунки

5

Рисунок Б.5 – Слайд №5

Матеріали та методи

За зручність, гнучкість мови та широкий спектр можливостей, мовами програмування було обрано:

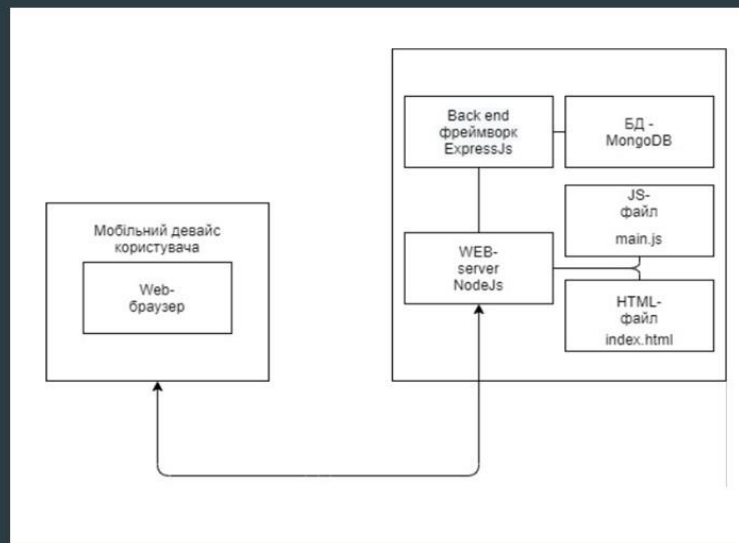
- JavaScript;
- TypeScript.

У розробці використовувався набір серверного програмного забезпечення MEAN, а також SDK для створення гібридних мобільних застосунків Ionic.

6

Рисунок Б.6 – Слайд №6

Архітектура мобільного вебсайту



7

Рисунок Б.7 – Слайд №7

Опис роботи з базами даних

При розробці мобільного вебсайту використовувалась MongoDB - документо - орієнтована система керування базами даних.

Було побудовано БД, що містить наступну інформацію:

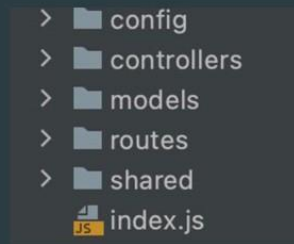
Таблиця	Назва таблиці	Данні таблиці
Інформація про користувача	Users	Повне ім'я, електронна адреса, зашифрований пароль, дата створення користувача, фото користувача
Інформація про дошку	Boards	Об'єкт власника дошки, прив'язані до дошки користувачі, метод керування проектом, список колонок, ім'я дошки, дата створення
Інформація про колонку	Columns	Об'єкт дошки, до якої належить колонка, об'єкт власника дошки, список карток, ім'я колонки, дата створення
Інформація про картку	Cards	Об'єкт колонки, до якої належить картка, об'єкт власника картки, прив'язані до дошки користувачі, список заміток, тип пріоритету, ім'я картки, опис картки, позиція картки, врахований час, оцінка картки, історія картки, дата створення
Інформація про замітку	Notes	Об'єкт картки, до якої належить замітка, об'єкт власника замітки, список користувачів, яким сподобалась замітка, ім'я, дата створення

8

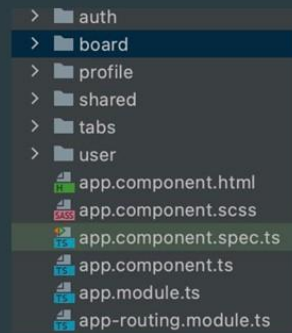
Рисунок Б.8 – Слайд №8

Основні рішення щодо модульного уявлення системи

Програмна система розподілена на модулі та компоненти з даними і функціями. Таке рішення дозволяє з легкістю використовувати ці данні для виконання своїх задач для роботи системи.



Програмна система
з файлами
серверної частини



Програмна система
з файлами інтерфейсу

9

Рисунок Б.9 – Слайд №9

Основні розроблені алгоритми

- ✓ авторизація користувачів;
- ✓ створення дошок методом Scrum;
- ✓ створення дошок методом Kanban;
- ✓ створення дошок методом Waterflow;
- ✓ створення дошок вільним методом;
- ✓ створення колонок та карток;
- ✓ отримання дошок, колонок та карток з усіма деталями;
- ✓ редагування колонок та карток;
- ✓ видалення дошок, колонок та карток;
- ✓ сортування карток;
- ✓ та інші.

10

Рисунок Б.10 – Слайд №10

Розроблений алгоритм - створення дошок методом Kanban

1. Користувач має увійти у систему.
 2. На екрані Дошки користувач повинен натиснути кнопку «+» для створення дошки.
 3. Користувач переходить на екран, де необхідно обрати метод управління проєктом «Kanban».
 4. Після вибору методу проєкта, користувач має ввести ім'я дошки та додати користувачів до дошки якщо це необхідно.
 5. Після того як користувач натискає на кнопку «Створити дошку», йде запит на серверну частину для створення даних для дошки методом Kanban за адресою:
 - дошки - /api/boards;
 - колонки - /api/boards/id_дошки/columns;
 - картки - /api/columns/id_колонки/cards.
 6. Створення даних у базі даних у таблицях:
 - Boards - для дошок;
 - Columns - для колонок;
 - Cards - для карток;
 7. Відправка оновлених даних на інтерфейсну частину.
 8. Виведення даних на екран користувача.
- При цьому створюються такі колонки: «Резерв», «До роботи», «У роботі», «Заблоковано», «Тест», «Виконано».

11

Рисунок Б.11 – Слайд №11

Розроблений алгоритм - створення дошок вільним методом

1. Користувач має увійти у систему.
2. На екрані Дошки користувач повинен натиснути кнопку «+» для створення дошки.
3. Користувач переходить на екран, де необхідно обрати метод управління проєктом «Вільний метод».
4. Після вибору методу проєкта, користувач має ввести ім'я дошки та додати користувачів до дошки якщо це необхідно.
5. Після того як користувач натискає на кнопку «Створити дошку», йде запит на серверну частину для створення даних для дошки вільним методом за адресою:
 - дошки - /api/boards;
 - колонки - /api/boards/id_дошки/columns;
 - картки - /api/columns/id_колонки/cards.
6. Створення даних у базі даних у таблицях:
 - Boards - для дошок;
 - Columns - для колонок;
 - Cards - для карток;
7. Відправка оновлених даних на інтерфейсну частину;
8. Виведення даних на екран користувача.

12

Рисунок Б.12 – Слайд №12

Техніко-економічні показники

Собівартість мобільного вебсайту складає 34 233 грн.

Було вирішено, що для успішної реалізації програмного продукту його ціна повинна бути не вище відносної більшості програмних продуктів подібного типу. А також з урахуванням підрахованого вище річного прибутку підприємства, оптимальною вартістю на одну копію програмного продукту буде - 25 000 грн.

Таким чином для повернення витрачених коштів достатньо реалізувати 2 копії програмного продукту, всі наступні реалізовані копії почнуть приносити дохід. Планується в середньому розробляти та продавати 2-3 копії програмного забезпечення на місяць.

Розробка і впровадження системи для спрощення та автоматизації управління проєктами різними методами є економічно ефективною, при цьому термін окупності капітальних вкладень складе 2 місяці.

13

Рисунок Б.13 – Слайд №13

Функції, що виконує продукт

- | | |
|--|---|
| ✓ авторизація у системі; | ✓ додавання користувачів до потрібної дошки або картки; |
| ✓ створення дошок за певним методом управління проєктами (Scrum, Kanban, Waterflow, «Вільна дошка»); | ✓ оцінка та врахування часу у картці; |
| ✓ дошка створена за обраним методом має відповідати правилам одному з методів управління проєктами (Scrum, Kanban, Waterflow, «Вільна дошка»); | ✓ можливість залишати коментарі у картці; |
| ✓ створювання колонок та карток; | ✓ можливість ставити пріоритет картці; |
| ✓ переміщення карток між колонками; | ✓ отримання повідомлень про дії на дошці, до якої доданий користувач; |
| | ✓ перегляд історії картки. |

14

Рисунок Б.14 – Слайд №14

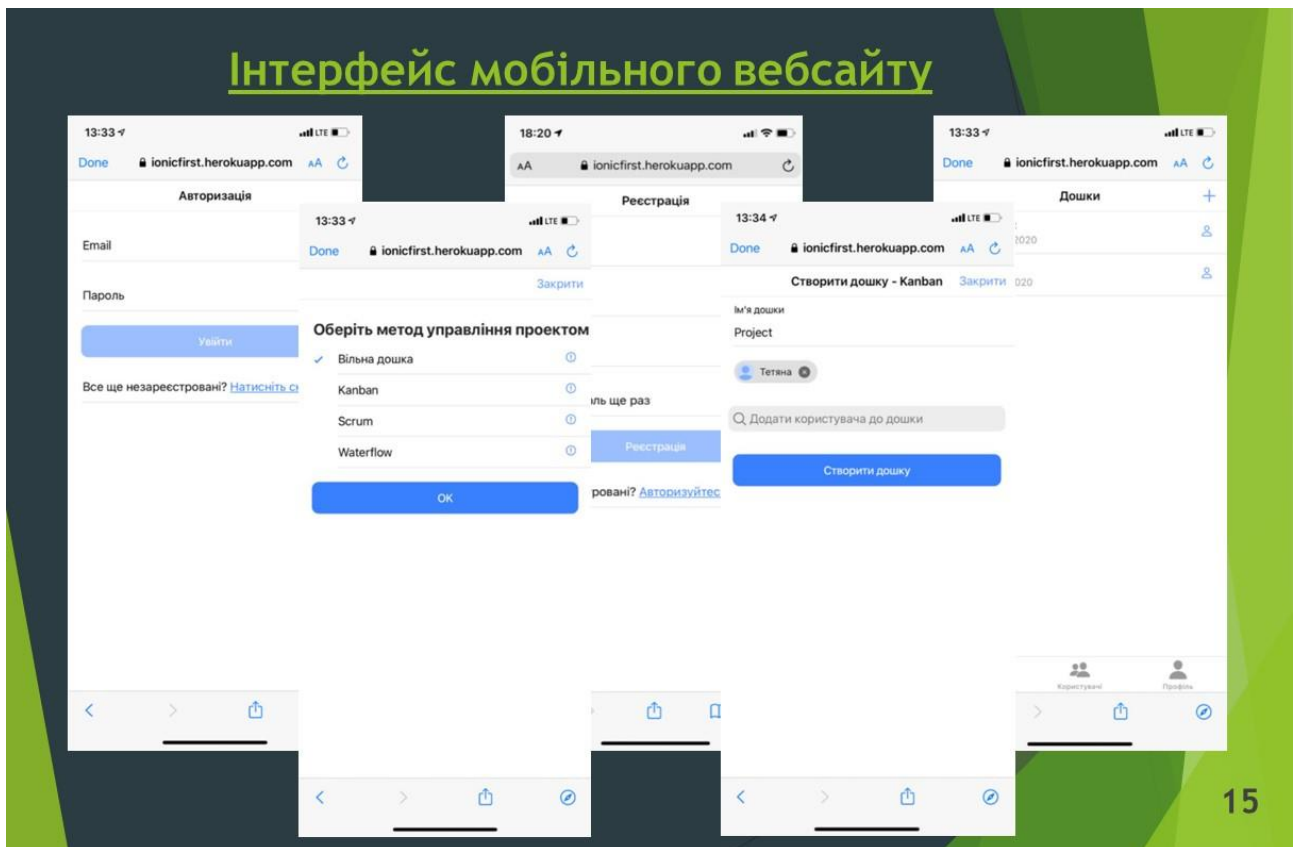


Рисунок Б.15 – Слайд №15

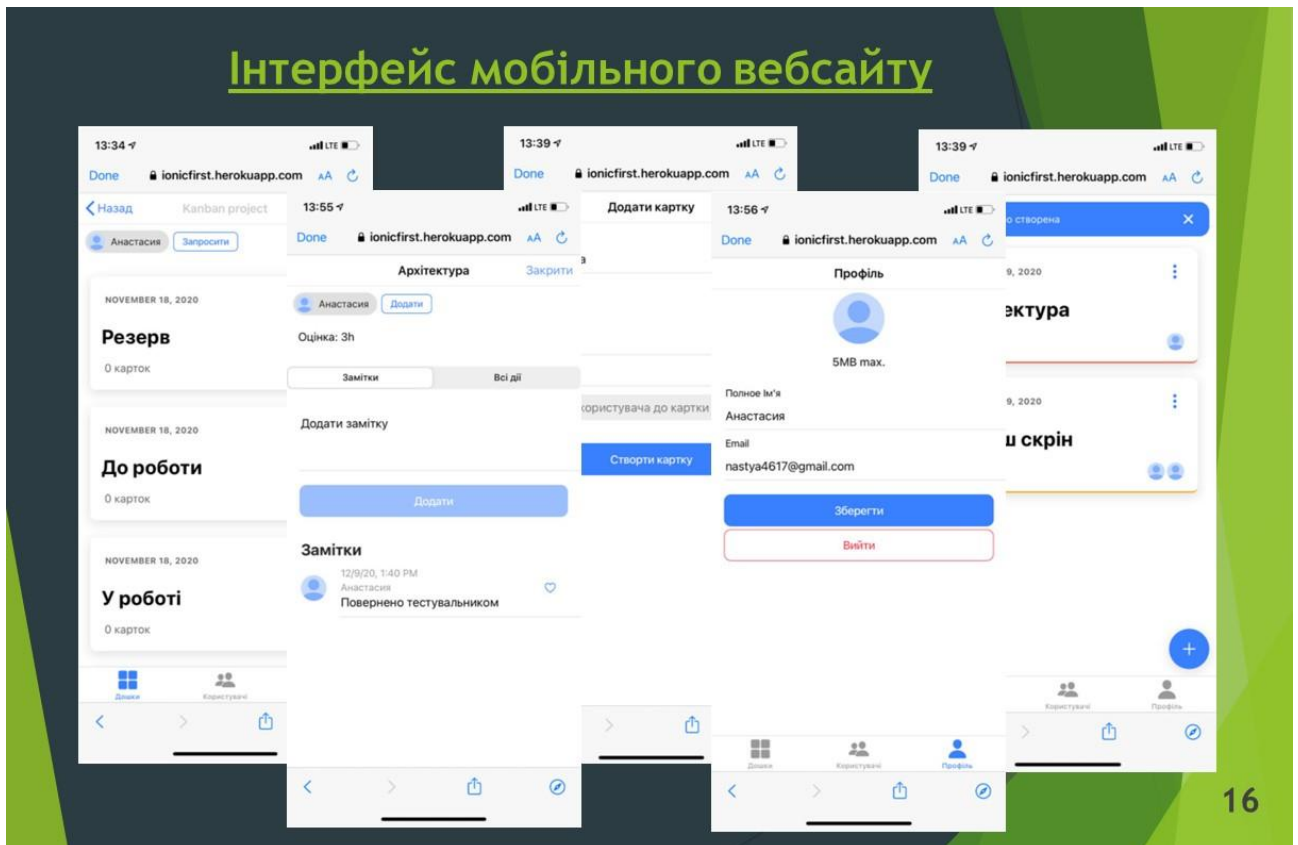


Рисунок Б.16 – Слайд №16

Висновки

Під час виконання магістерської дипломної роботи були досліджені найбільш поширені методи управління проєктами, а саме Scrum, Kanban та Waterflow. Була розроблена програмна система для спрощення та автоматизації управління проєктами різними методами. Також було виконано огляд та аналіз існуючих методів вирішення завдання та існуючих програмних засобів.

Практична цінність розробки програмної системи полягає в можливості застосування її на підприємстві для спрощення та автоматизації управління проєктами різними методами.

Наукова новизна полягає у тому, що вперше, на основі дослідження методів управління проєктами та їх практичної реалізації, був модифікований підхід до створення подібних сервісів шляхом реалізації вибору метода управління проєктами та реалізації свого варіанту керування проєктами «Вільна дошка».

17

Рисунок Б.17 – Слайд №17