МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: «Web-додаток підтримки діяльності проектного менеджера у КЗАПР»

за спеціальністю 122 «Комп'ютерні науки»,
освітньо-професійна програма «Інформаційні технології проектування»

Виконавець роботи: студент групи ІТ-82-0 Райко Дмитро Іванович

Кваліфікаційна робота бакалавра
захищена на засіданні ЕК
з оцінкою _____«__» _____2022 р.

Науковий керівник _____ к.т.н., доц., Антипенко В.П.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Засвідчую, що у цій дипломній роботі немає
запозичень з праць інших авторів
без відповідних посилань.

Студент _____
(підпис)

Суми-2022

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра інформаційних технологій
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Інформаційні технології проектування»

ЗАТВЕРДЖУЮ
Зав. кафедрою ІТП

_____ В. В. Шендрик
«____»_____2022 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА СТУДЕНТУ

Райко Дмитро Іванович

1 Тема роботи *Web-додаток підтримки проектного менеджера у КЗАПР*

керівник роботи *Антипенко Вікторія Петрівна, к.т.н., доцент*,

затверджені наказом по університету від «*27*» квітня 2022 р. №0301_VI

2 Строк подання студентом роботи « 10 » червня 2022 р.

3 Вхідні дані до роботи технічне завдання на розробку web-додатку підтримки діяльності проектного менеджера у КЗАПР

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) аналіз предметної області, моделювання та проектування web-додатку, розробка web-додатку

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

6. Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7.Дата видачі завдання_____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Оформлення планування робіт	02.05.2022 – 05.05.2022	
2	Створення технічного завдання	05.05.2022 – 12.05.2022	
3	Проведення аналізу предметної області	12.05.2022 – 18.05.2022	
4	Проведення проектування web-додатку	18.05.2022 – 25.05.2022	
5	Розробка web-додатку	25.05.2022 – 08.06.2022	
6	Тестування web-додатку	08.06.2022 – 09.06.2022	
7	Завантаження web-додатку на хостинг	10.06.2022 – 11.06.2022	
8	Оформлення пояснювальної записки	02.05.2022 – 12.06.2022	

Студент _____ Райко Д.І.
(підпис)

Керівник роботи _____ к.т.н., доц. Антипенко В.П..
(підпис)

РЕФЕРАТ

Тема кваліфікаційної роботи бакалавра «Web-додаток підтримки діяльності проектного менеджера у КЗАПР».

Пояснювальна записка складається зі вступу, трьох розділів, висновків, списку використаних джерел із 30 найменувань, чотирьох додатків. Загальний обсяг пояснювальної записки складає 105 сторінок, у тому числі 52 сторінки основного тексту, 3 сторінки списку використаних джерел, 45 сторінок додатків.

Кваліфікаційну роботу бакалавра присвячено розробці web-додатку підтримки діяльності проектного менеджера у КЗАПР.

У першому розділі проведено аналіз останніх досліджень за тематикою даного проекту. Здійснено порівняння аналогів розроблюваного web-додатку, визначено їх плюси та мінуси. Також було зазначено мету, задачі та засоби реалізації проекту.

Другий розділ присвячений проведенню структурно-функціонального моделювання. Визначено варіанти використання web-додатку та спроектовано базу даних. Результатом моделювання є контекстна діаграма IDEF0 та її декомпозиція, діаграма варіантів використання, а також схема бази даних та її таблиці.

У третьому розділі описано архітектуру web-додатку та процес розробки даного програмного продукту. Проведено тестування web-додатку.

Ключові слова: WEB-ДОДАТОК, БАЗА ДАНИХ, JAVASCRIPT, REACTJS, MUI, УПРАВЛІННЯ ПРОЕКТАМИ, ПРОЕКТНИЙ МЕНЕДЖЕР, ПРОЕКТ.

ЗМІСТ

ВСТУП

Сьогодення визначає інформаційні технології як рушій розвитку будь-якої галузі, бо їх використання дозволяє прискорити та автоматизувати будь-яку розробку. В останній час особливо популярним став напрямок тайм-менеджменту. Усе більше людей підтримують розподіл виробничого та особистого часу на окремі задачі з конкретними термінами. У свою чергу це дозволяє збільшити продуктивність кожного, хто дотримується графіку їх виконання.

Сучасні організації також потребують застосування аналогічного процесу. Це необхідно для автоматизації процесів розробки проектів. У зв’язку з пандемією та/або бойовими діями більшість компаній вимушені перейти на віддалену роботу. Тому зараз є потреба в створенні програмного продукту, який допоможе організувати та забезпечити процес виконання завдань співробітниками, які фізично відсутні на робочому місці в компанії. Проектному менеджеру набагато зручніше управляти командою, назначати задачі, відслідковувати їх виконання, а також просто підтримувати комунікацію з працівниками, використовуючи програмний додаток, наприклад, web-орієнтований. І підприємства машинобудівної галузі не є винятком. Тому розробка програмного продукту підтримки діяльності проектного менеджера (PM) у комплексі засобів автоматизації проектувальних робіт (КЗАПР) є актуальною. Він повинен задовольняти всі функціональні вимоги для автоматизованого проектування робіт.

Отже, метою даного дослідження є розробка web-додатку підтримки діяльності проектного менеджера у КЗАПР та автоматизація таких процесів управління командою проекту, як назначення завдань на виконання, їх відповідальних і контроль над реалізацією проектів.

Для досягнення мети проекту необхідно виконати наступні задачі:

− дослідити предмету область, цільову аудиторію й актуальність роботи;

- проаналізувати web-додатки-аналоги та виділити їх позитивні та негативні сторони;
- зробити огляд останніх публікацій та літератури;
- спроектувати структуру web-додатку;
- обрати технології для реалізації web-додатку;
- створити прототип web-додатку;
- розробити функціонал web-додатку для підтримки проектного менеджера у КЗАПР;
- провести тестування web-додатку.

Результати роботи були апробовані на науково-практичній конференції ІМА-2022 у Сумському державному університеті [1].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд останніх досліджень і публікацій

Розробка проектів у машинобудівній інженерії – це надзвичайно складний та довготривалий вид діяльності. Він пов’язаний з потребою виконання великих обсягів дослідно-конструкторських робіт. Тому процес автоматизації певних процесів просто необхідний для ініціалізації проектів у машинобудуванні. У даній галузі для спрощення виконання окремих проектувальних процедур застосовується програмне забезпечення SolidWorks [2], AutoCAD [3] тощо. Вони дійсно використовуються в основному для автоматизованого проектування креслень і 3D моделей. Тому, проаналізувавши сучасні дослідження в цій області, можна зробити висновок, що процесу управління проектом і командою в цілому в останніх розробках приділено недостатньо уваги. А, отже, дане питання є актуальним сьогодні.

Управління проектами та його виконавцями є однією із найвагоміших сфер діяльності підприємства. Належна її організація може підвищити ефективність роботи будь-якого виробництва. Вдосконалення механізмів управління проектом та командою, пов’язане з виявленням значних змін у поточній діяльності, активної та вирішальної ролі співробітників, які задіяні в процесі управління підприємством[4]. Повсякденна робота менеджера проекту залежить від завдань, які надходять від керівництва, де потрібно звертати увагу на кожну деталь. Тому процеси пов’язані з управлінням проектами та командою є ідеальним об’єктом для автоматизації. Це значно спростить виконання багатьох процесів всередині організації. Кваліфіковані PM матимуть можливість нарощувати чисельність проектів, міць команди, обсяги виконання робіт, при цьому не збільшуючи кількість співробітників підприємства [5]. Вирішенням проблеми автоматизації процесів управління проектами та його командою є розробка відповідного web-додатку. Він повинен надавати можливість створювати, редагувати та назначити завдання та їх відповідальних, а також

контролювати здійснення робіт співробітниками, дозволяючи управляти повідомленнями. Останні будуть сповіщати виконавця про нову задачу або про приближення дедлайну поточної.

1.2 Аналіз існуючих продуктів-аналогів

Web-додатки для управління проектами вже давно є наявними в мережі Інтернет та широко використовуються PM різних організацій. Ці продукти забезпечують весь необхідний функціонал для планування завдань проектів, але не передбачають особливих функціональних можливостей необхідних для машинобудівної галузі. Наприклад, специфічна пріоритезація завдань, їх назначення співробітникам відповідної спеціалізації, управління повідомленнями щодо виконання робіт тощо.

Для визначення основних вимог до майбутнього програмного продукту було проведено порівняння таких web-додатків-аналогів, як «GANTPRO», «Trello» та «YouGile».

Web-додаток «GANTPRO» [6] забезпечує перегляд прогресу виконання задачі, управління ресурсами та завантаженістю команди. Але він має як переваги, так і недоліки. Як і всі web-додатки даної області «GANTPRO» забезпечений інструментами створення проектів та завдань до них. Його головна сторінка представлена на рисунку 1.1. На ній відображені завдання та проект у вигляді часової прямої. Проте відсутня можливість переглянути останній у формі канбан-карток завдань.

Сторінка завантаженості співробітника містить календар із кількістю занятих годин на день по певній роботі. Це є зручним інструментом для проектного менеджера, оскільки він має змогу оцінювати можливості члена команди виконати назначене завдання.

Дизайн даного web-додатку виконаний у діловому та лаконічному стилі, має синю кольорову гамму.

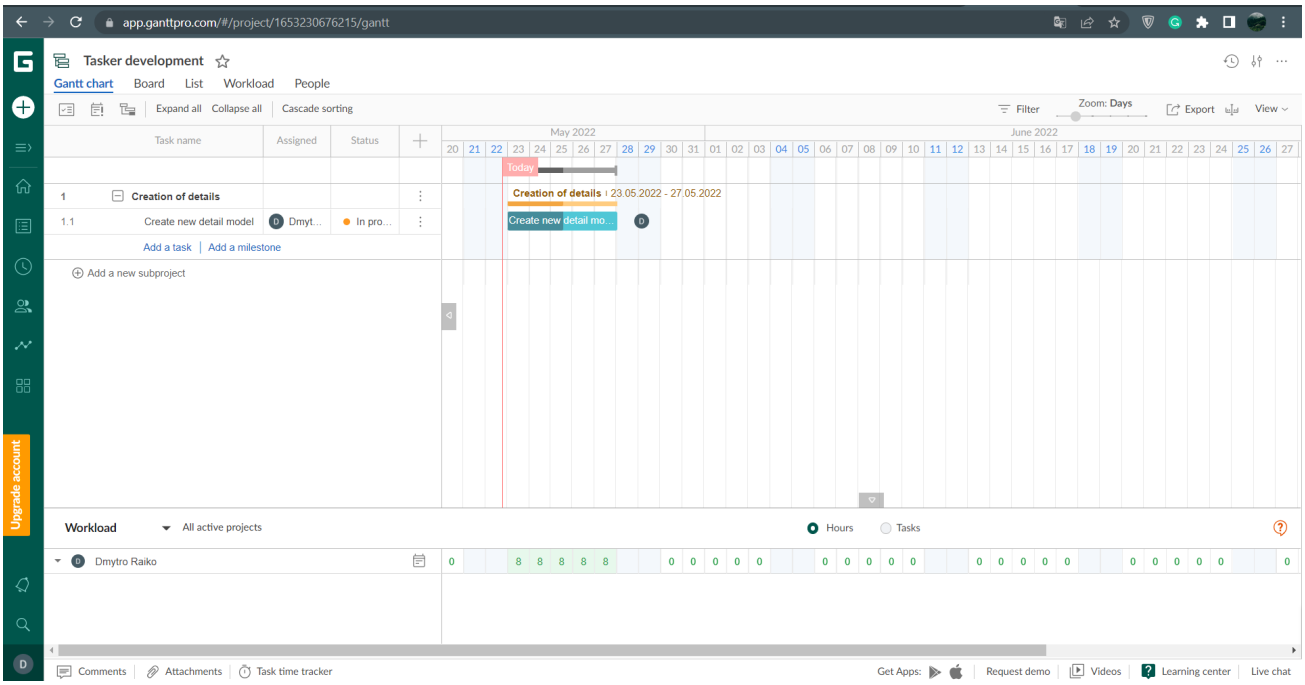


Рисунок 1.1 – Головна сторінка проекту у web-додатку «GANTPRO»

Наступним було досліджено інструмент планування проектів «Trello» [7]. Він забезпечує зручний інтерфейс створення завдань та отримання сповіщень. Головна сторінка цього web-додатку містить перелік проектів, створених або назначених на користувача (рис. 1.2). Даний сервіс має власні сильні та слабкі сторони. Наприклад, сторінка проекту містить канбан-дошку з завданнями з різними статусами їх виконання. Це є досить зручно. При цьому проектний менеджер може самостійно створювати ці задачі та переміщувати їх в необхідний статус (рис. 1.3-1.4). Проте «Trello» не забезпечує PM функціоналом контролю за процесом реалізації проекту.

Web-додаток має сучасний дизайн із синьо-блакитною кольоровою гамою. Від користувача не вимагається особливих навичок роботи з даним продуктом, оскільки інтерфейс «Trello» є досить інтерактивним та інтуїтивно-зрозумілим для застосування.

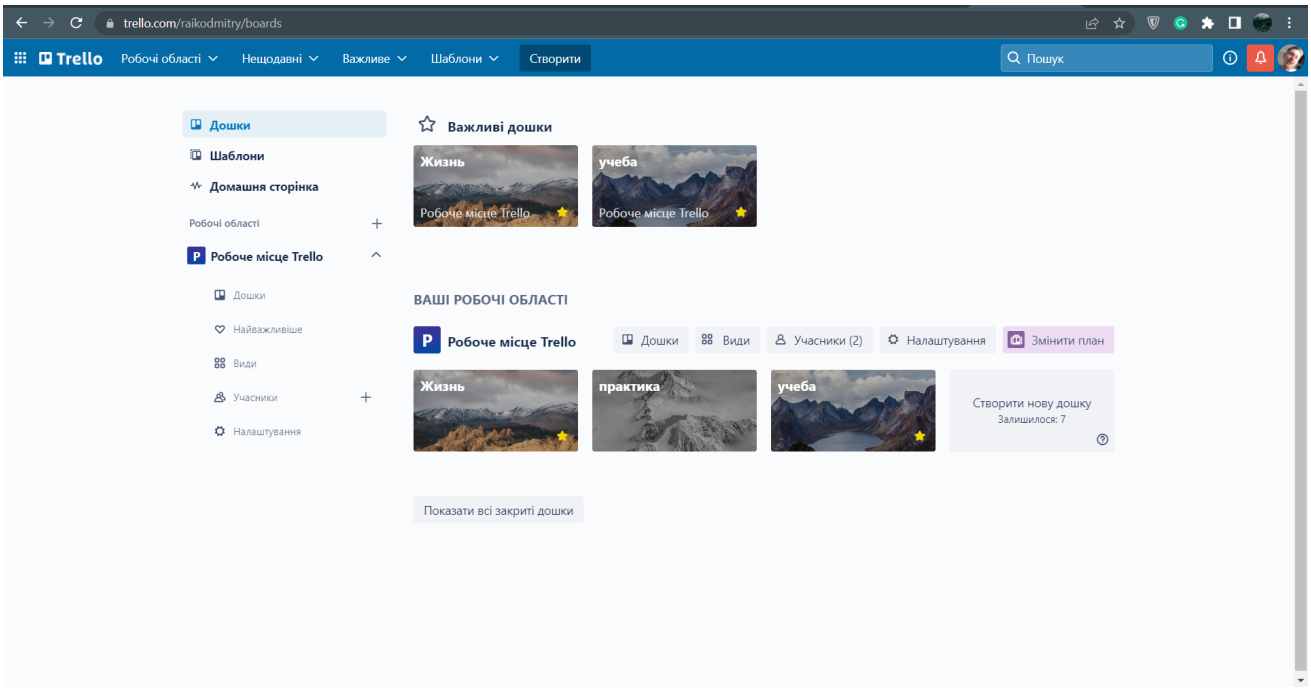


Рисунок 1.2 – Головна сторінка web-додатку «Trello»

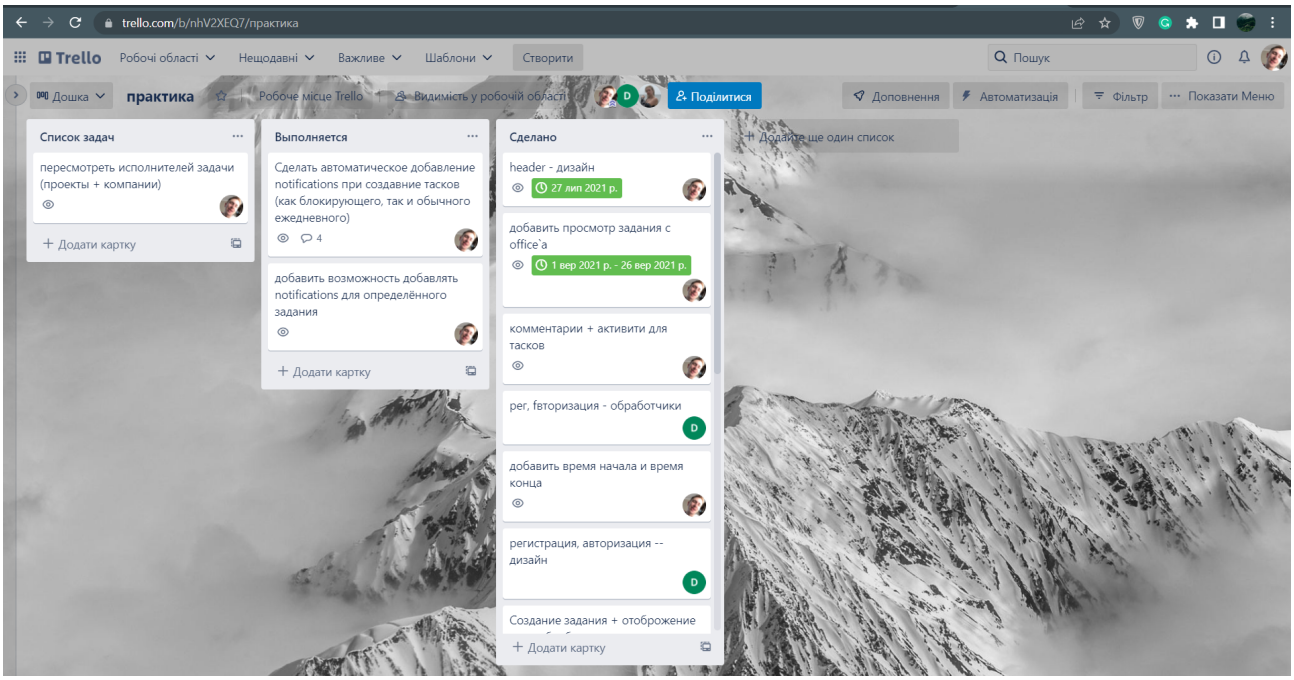


Рисунок 1.3 – Сторінка проекту web-додатку «Trello»

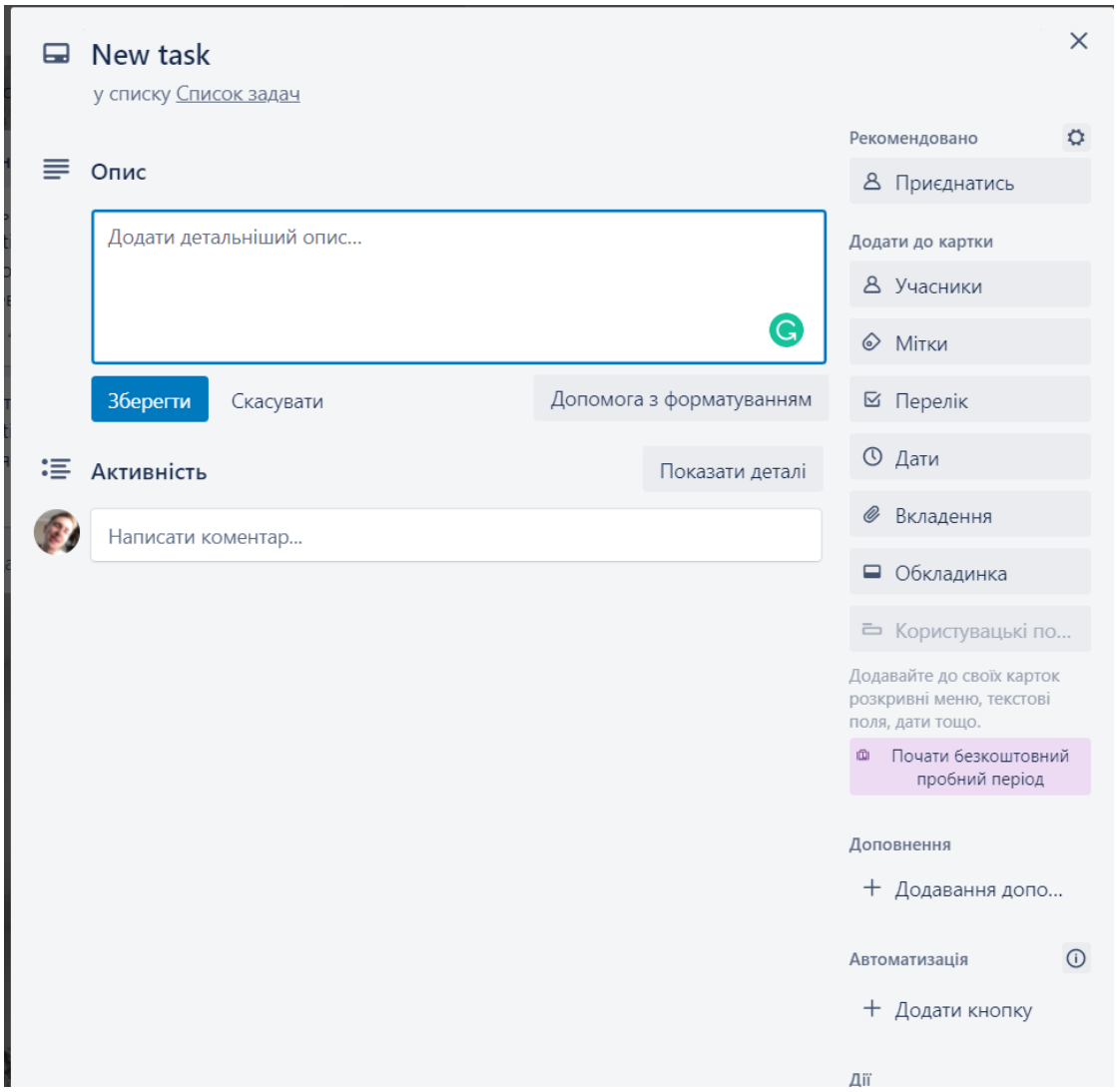


Рисунок 1.4 – Форма для створення задачі в web-додатку «Trello»

Останнім було розглянуто програмний продукт «BaseCamp» [8] призначений для управління командою під час виконання проектів. Для підтримки зв’язку між виконавцями web-додаток пропонує можливість спілкування у вбудованому чаті. Також PM є доступним функціонал розробки списків завдань та відслідковування прогресу виконання робіт у проекті. Але даний програмний продукт не є виключенням і має свої плюси та мінуси. На головній сторінці web-додатку «BaseCamp» (рис. 1.5) представлено проект, розклад співробітника та назначені на нього задачі. Сторінка «Project» містить список пов’язаних із проектом документів, список необхідних для виконання завдань та дедлайни (рис. 1.6). Однак програмний продукт не забезпечує налаштування функціонала під потреби користувача.

Дизайн web-додатку «BaseCamp» виконаний в чорній темі, яка є досить популярною сьогодні. Інтерфейс програмного продукту є логічно побудованим, що дозволяє будь-кому з легкістю ним користуватися.

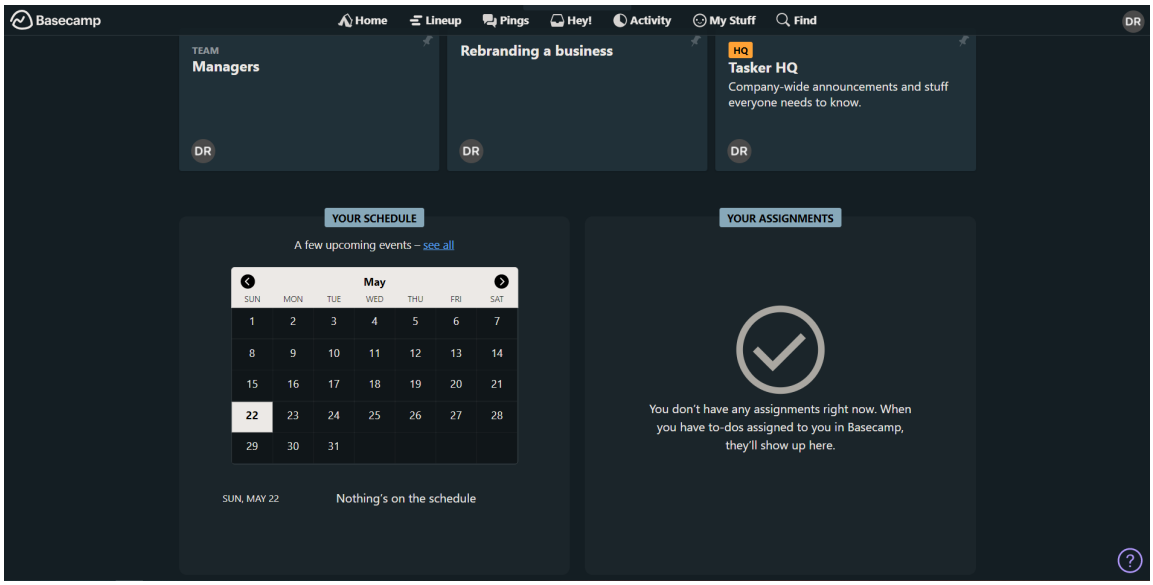


Рисунок 1.5 – Головна сторінка web-додатку «BaseCamp»

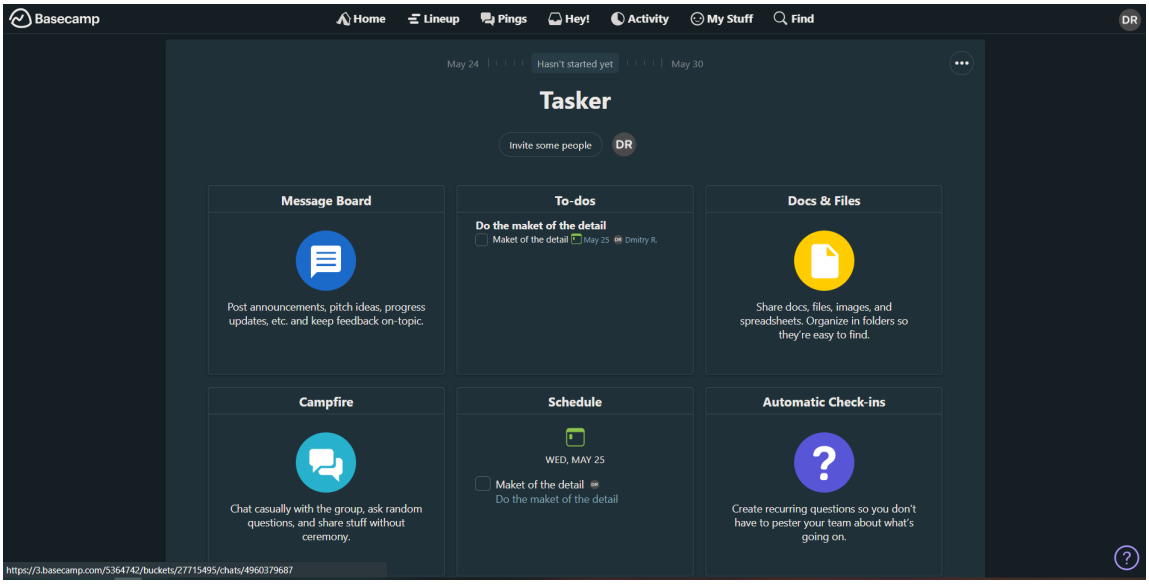


Рисунок 1.6 – Головна сторінка проекту web-додатку «BaseCamp»

Після проведеного аналізу існуючих аналогів web-додатку для управління проектами, було проведено визначення переваг та недоліків. Результати представлені в таблиці 1.1.

Таблиця 1.1 – Порівняльна характеристика web-додатків-аналогів

Характеристика/ Web-додаток	**«GANTPRO»**	**«Trello»**	**«BaseCamp»**
Зручний інтерфейс	+	+	+
Сучасний дизайн	-	+	+
Інтерактивність	-	+	+
Навігація	+	-	+
Функціональність	+	+	-
Авторизація користувачів	+	+	+
Перегляд завантаженості робітника	+	-	+
Можливість створення завдань в проекті	+	+	+
Можливість контролю за реалізацією проекту	+	-	+
Можливість створення користувацьких статусів для робіт	-	+	-

Проаналізувавши дані з таблиці 1.1, можна виділити необхідний функціонал, який треба реалізувати, базуючись на тих недоліках, які мають представлені аналоги. Створюваний web-додаток повинен мати сучасний дизайн і логічно-побудований інтерфейс. Із функціоналу варто реалізувати можливість створення задач до проекту, їх перегляд у режимі календаря й управління повідомленнями, які сповіщають людину про наближення строку виконання завдання.

1.3 Постановка задачі

Головною метою дослідження є розробка web-додатку підтримки діяльності проектного менеджера у КЗАПР та автоматизація таких процесів управління командою проекту, як назначення завдань на виконання, їх відповідальних і контроль над реалізацією проектів.

Основними вимогами до розроблюваного програмного продукту є:

- створення web-форм для ініціалізації та редагування завдань та проектів;
- управління сповіщеннями;
- надання інформації про розпорядок завдань у вигляді календарю;
- повернення повної інформації про обране завдання користувача.

Для досягнення цілей проекту необхідно виконати наступні завдання:

- дослідити предмету область, цільову аудиторію й актуальність роботи;
- проаналізувати web-додатки-аналоги та виділити їх позитивні та негативні сторони;
- зробити огляд останніх публікацій та літератури;
- спроектувати структуру web-додатку;
- обрати технології для реалізації web-додатку;
- створити прототип web-додатку;
- розробити функціонал web-додатку для підтримки проектного менеджера у КЗАПР;
- провести тестування web-додатку.

Вимоги до структури web-додатку, видів забезпечення та його функціонування викладені в технічному завданні (Додаток А).

Для розробки даного програмного продукту було обрано наступні технології: Java-script [9] та бібліотеку React.js [10] для створення інтерфейсів користувача, бібліотеку MUI [11], яка пропонує повний набір інструментів інтерфейсу користувача, каскадні таблиці стилів CSS [12] для створення стилів сторінки та надання додатку адаптивності, та скриптову метамову Sass [13], яка перетворюється в каскадні таблиці стилів, для асинхронної роботи з серверною частиною – скриптову мову програмування JavaScript. Для реалізації бази даних було обрано Систему управління базою даних (СУБД) PostgreSQL [14].

2 ПРОЕКТУВАННЯ WEB-ДОДАТКУ

2.1 Структурно-функціональне моделювання

Методологія IDEF0 [15] дозволяє ініціалізувати реалізацію ієрархічної системи діаграм. Першочергово відбувається опис останньої та її взаємодії з довколишнім середовищем. Наступним етапом є функціональна декомпозиція. Вона дозволяє системі ділитися на підсистеми. При чому опис кожної з них відбувається окремо. Після цього підсистеми розбиваються на ще дрібніші для досягнення певного рівня деталізації.

IDEF0 діаграми складаються з блоків та дуг. Перші характеризують функції моделі. Другі поєднують блоки разом. Дуги також показують взаємозв’язки між ними.

Прямокутниками на діаграмах позначаються функціональні блоки, що означають процеси, які відбуваються протягом певного часу. Вони повертають результати, які ідентифікуються.

Функціональне моделювання web-додатку підтримки діяльності проектного менеджера у КЗАПР в IDEF0 зображене на рисунку 2.1.

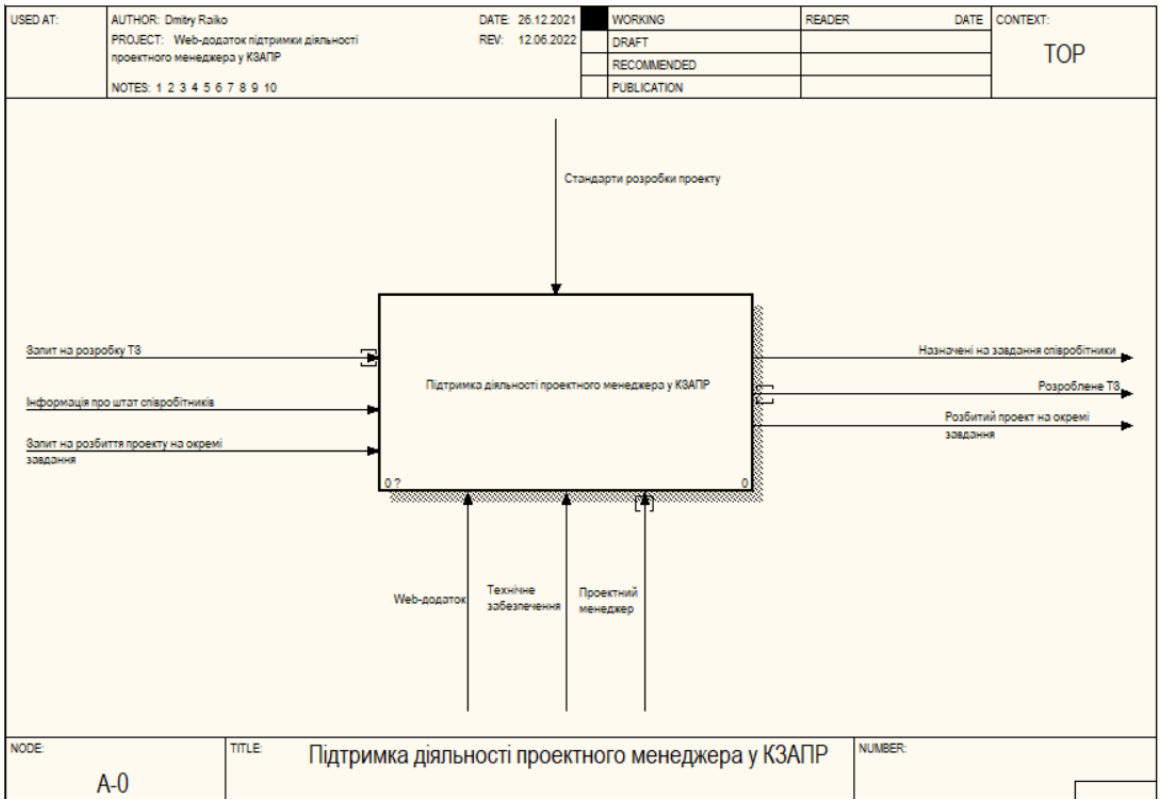


Рисунок 2.1 – IDEF0 діаграма процесу підтримки діяльності проектного менеджера у КЗАПР

Для конкретизації внутрішніх процесів діаграми IDEF0 було зроблено декомпозицію, яка відображена на рисунку 2.2.

Блоком для декомпозиції є організація процесу розбиття проекту на окремі завдання та назначення співробітників відповідальними за їх виконання.

Для узагальнення інформації було розроблено таблиці входів і виходів даних для певного підпроцесу деталізації (табл. 2.1).

Діаграма декомпозиції організації процесу розбиття проекту на окремі завдання та назначення співробітників відповідальними за їх виконання представлена такими підпроцесами:

- аналіз специфікації;
- відбір конкретних завдань;
- назначення часових рамок;
- назначення співробітників.

Таблиця 2.1 – Вхідні та вихідні дані для діаграми розбиття проекту на кремі завдання та назначення співробітників на їх виконання

Стрілка/ Підпроцес	**Вхідні дані**	**Управління**	**Механізми**	**Вихідні дані**
Аналіз специфікації	Специфікації, Запит на розбиття проекту на завдання	Стандарти розробки проекту	Web-додаток, Технічне забезпечення, Проектний менеджер, Замовник, Керівник проекту Замовник, Керівник проекту, Web-додаток, Технічне забезпечення,	Виділені групи робіт
Відбір конкретних завдань	Виділені групи робіт			Терміни розробки проекту, Виділені завдання
Назначення часових рамок	Терміни розробки проекту, Виділені завдання			Інформація про штат співробітників, Визначені часові межі виконання завдань
Назначення співробітників	Інформація про штат співробітників, Визначені часові межі виконання завдань	Стандарти розробки проекту, Трудовий договір		Розбитий проект на підзавдання, Назначені на завдання співробітники

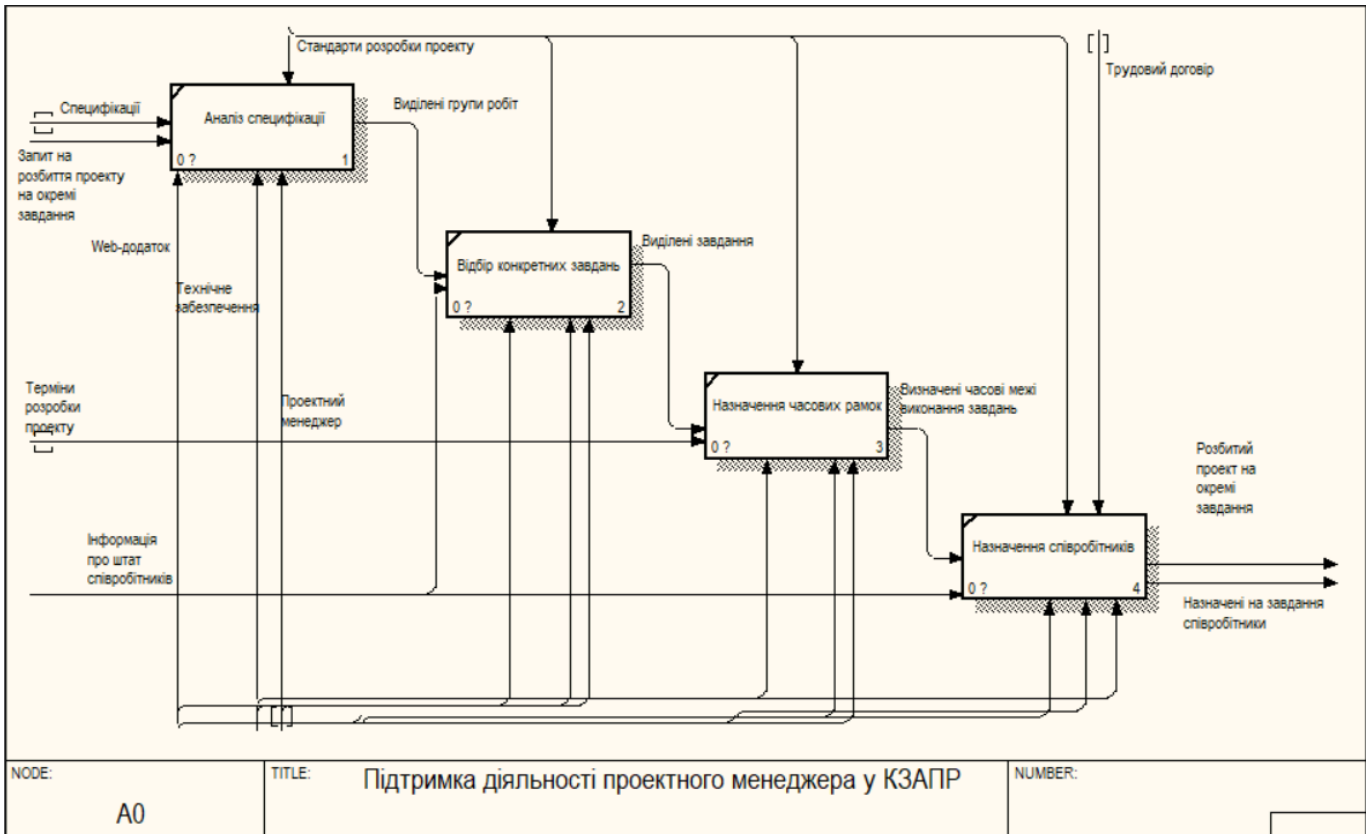


Рисунок 2.2 – Діаграма декомпозиції процесу підтримки діяльності проектного менеджера у КЗАПР

2.2 Моделювання варіантів використання

Після моделювання процесів та дослідження структури розроблюваного web-додатку необхідно створити діаграму варіантів використання (Use Case Diagram). Вона описує застосування системного проектування користувачем для досягнення певних цілей, а саме визначення функцій програмного забезпечення, які мають бути реалізовані [16].

Дана діаграма містить три основні елементи процесу. Це актори, які взаємодіють із системою; функціональні вимоги, які позначають як елементи будуть себе поводити; та цілі, яких треба досягнути при розробці.

Для web-додатку підтримки діяльності проектного менеджера варіанти використання є наступними:

- Перегляд завдань;
- Ініціалізація проекту;
- Зміна статусу завдання;
- Перегляд прогресу виконання завдань;
- Ініціалізація контракту;
- Перегляд прогресу виконання завдань;
- Підписання контракту;
- Перегляд завдань
- Створення завдань(підпроектів);
- Розбиття проекту на підпроекти.

Акторами на діаграмі є виконувач, керівник компанії, замовник, проектний менеджер.

Діаграма варіантів використання web-додатку підтримки діяльності проектного менеджера у КЗАПР представлена на рисунку 2.3.

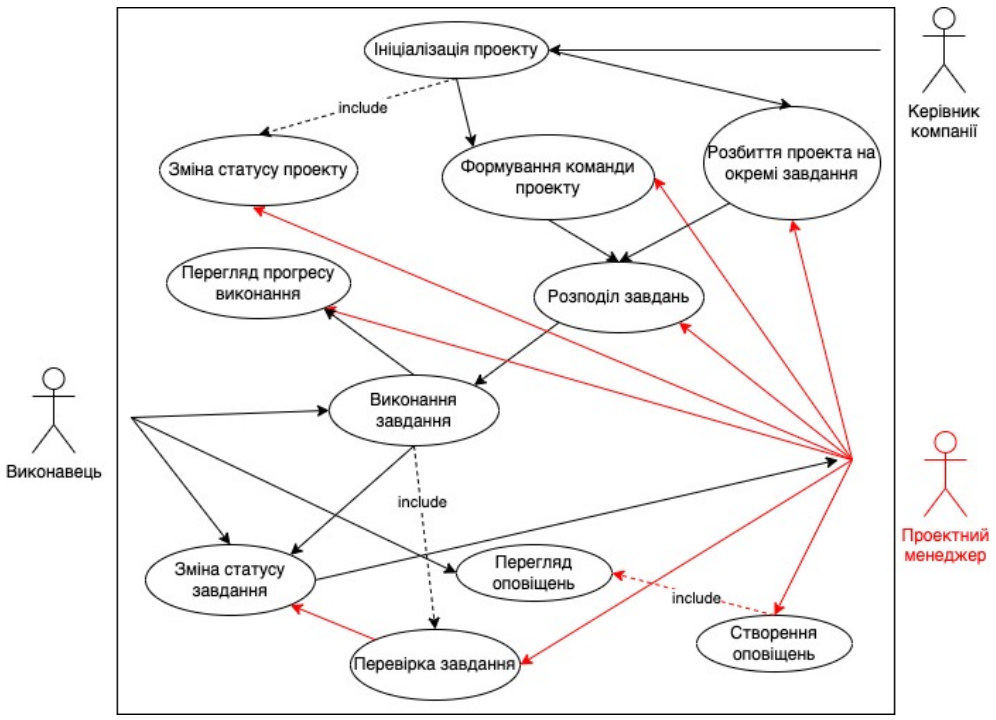


Рисунок 2.3 – Діаграма варіантів використання

2.3 Проектування бази даних

Розробка web-додатку не буде повноцінною без застосування СУБД [17] для керування та збереження даних. База даних (БД), у свою чергу, призначена для зберігання даних, які об’єднані спільною властивістю. Тож розробка БД є обов’язковою умовою для створення web-додатку та наповнення його контентом.

Спершу треба спроектувати базу даних. У ході цього процесу визначається склад таблиць та їх зв’язки між собою. Вони мають певну структуру, а саме послідовні стовпці, їх типи та розмір, а також первинний ключ.

У процесі проектування було визначено такі сутності для КЗАПР:

- Ресурс (Resource);
- Проект (Project);
- Ролі (Roles);
- Правила (Rules);
- Статус (Status);
- Логи (Logs);
- Користувач (User);
- Профіль (Profile);
- Доступ (Permission);
- Контракт (Contract);
- Список завдань (TaskList);
- Співробітники проекту (ProjectStaff);
- Повідомлення (Notification);
- Список повідомлень (NotificationList);
- Роль профілю (ProfileRoles);
- Вкладення (Attachment);
- Список вкладень (AttachmentList);
- Завдання (Task);

- Коментар (Comment);
- Запити в підтримку (Support_Requests).

На рисунку 2.4 показана фізична модель БД web-додатку підтримки діяльності проектного менеджера у КЗАПР.

Для реалізації функціональних можливостей керівника проектної організації було виділено та створено наступні сутності:

- Завдання(Task) – містить повну інформацію про завдання;
- Список завдань(TaskList) – містить інформацію про виконавців завдання та керівником співробітника;
- Коментар(Comment) – містить інформацію про зміст коментаря, інформацію про завдання, до якого був залишений коментар і ким був залишений коментар;
- Запити в підтримку(Support_Requests) – містить інформацію про запит в підтримку та ким був залишений запит;
- Повідомлення(Notification) – містить інформацію про повідомлення та його пріоритет;
- Список повідомлень(NotificationList) – містить інформацію про отримувача повідомлення та його статус;
- Вкладення(Attachment) – містить посилання на вкладення та інформацію про тип вкладення;
- Список вкладень(AttachmentList) – містить інформацію про проект, контракт, та завдання до якого було зроблено вкладення.

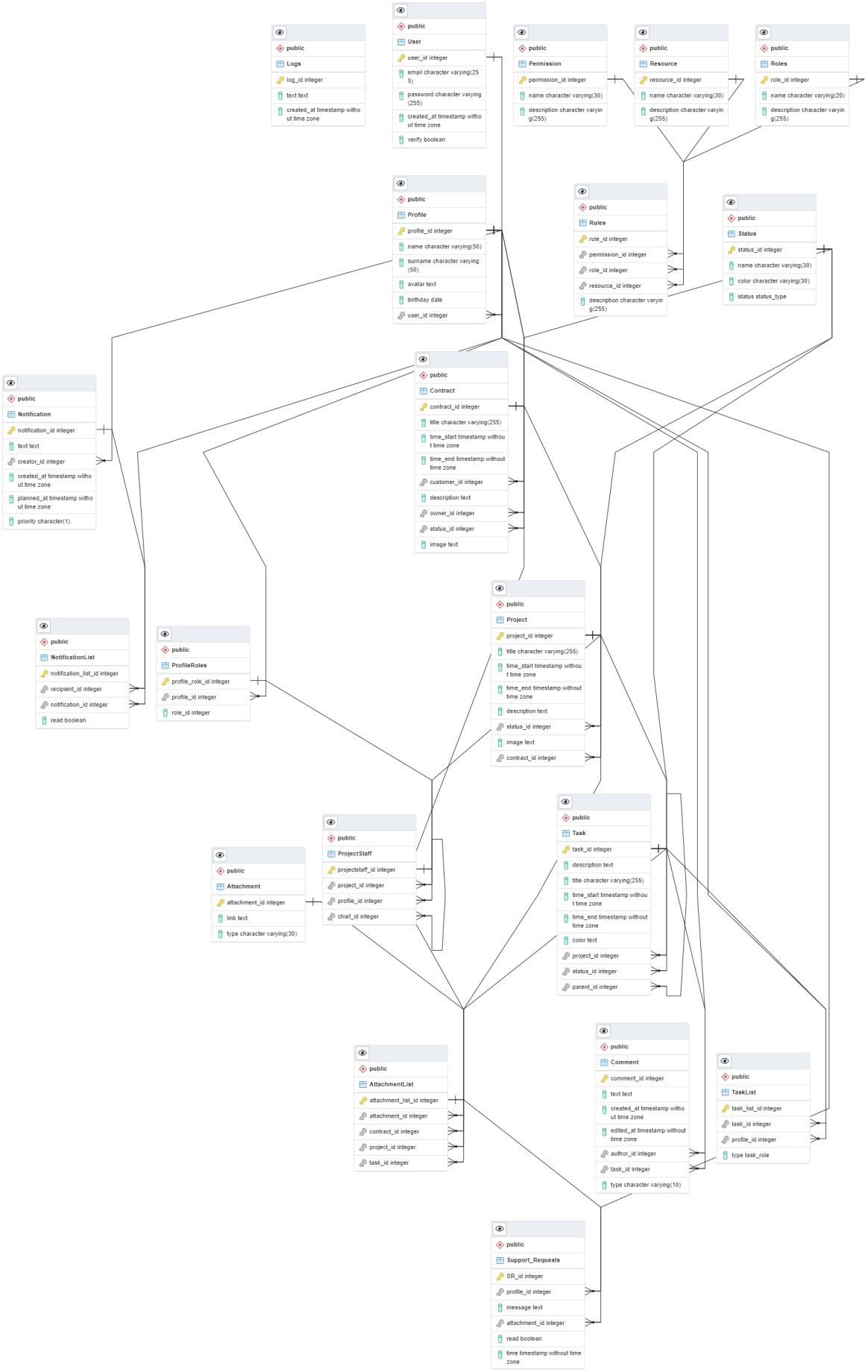


Рисунок 2.4 – Фізична модель бази даних

Таблиця Завдання(Task) має наступні атрибути:

- task_id (integer) – первинний ключ таблиці;
- title (character varying(255)) – назва завдання;
- description (text) – опис завдання;
- time_start (timestamp without time zone) – дата початку виконання завдання;
- time_end (timestamp without time zone) – дата кінця виконання завдання;
- color (text) – колір завдання;
- project_id (integer) – foreign key проекту, до якого відноситься завдання;
- status_id (integer) – foreign key статусу завдання;
- parent_id (integer) – foreign key батьківського завдання.

Таблиця Список завдань(TaskList) має наступні атрибути:

- task_list_id (integer) – первинний ключ проекту;
- task_id (integer) – foreign key завдання;
- profile_id (integer) – foreign key профілю;
- type (task_role) – привілеї доступу до завдання(може бути creator, assigned, inspector).

Таблиця Коментар(Comment) має наступні атрибути:

- comment_id (integer) – первинний ключ таблиці коментарів;
- text (text) – зміст повідомлення;
- created_at (timestamp without time zone) – часова мітка коли коментар був створений та відправлений;
- edited_at (timestamp without time zone) – часова мітка коли коментар був відредагований;
- author_id (integer) – foreign key відправника коментарю;
- task_id (integer) – foreign key завдання, до якого був залишений коментар;
- type (character varying(10)) – тип коментарю(activity – коли коментар був змінений та comment – простий коментар).

Таблиця Запити в підтримку(Support_Requests) має наступні атрибути:

- SR_id (integer) – первинний ключ таблиці запитів;
- message (text) – зміст запиту;
- read (boolean) – мітка прочитано чи ні;
- profile_id (integer) – foreign key профілю;
- attachment_id (integer) – foreign key вкладень;
- time (timestamp without time zone) – часова мітка коли запит був отриманий.

Таблиця Повідомлення(Notification) має наступні атрибути:

- notification_id (integer) – первинний ключ повідомлень;
- text (text) – зміст повідомлення;
- created_at (timestamp without time zone) – часова мітка коли повідомлення було створено;
- planned_at (timestamp without time zone) – часова мітка коли повідомлення було заплановано на відправлення;
- creator_id (integer) – foreign key автора повідомлення;
- priority (character(1)) – пріоритет повідомлення(термінове чи ні повідомлення).

Таблиця Список повідомлень(NotificationList) має наступні атрибути:

- notification_list_id (integer) – первинний ключ списку повідомлень;
- recipient_id (integer) – foreign key отримувача повідомлення
- notification_id(integer) – foreign key повідомлення;
- read (boolean) – ідентифікатор прочитане чи ні повідомлення.

Таблиця Вкладення(Attachment) має наступні атрибути:

- attachment_id (integer) – первинний ключ вкладень;
- link (text) – посилання на вкладення;
- type(character varying(30)) – тип вкладення.

Таблиця Список вкладень(AttachmentList) має наступні атрибути:

- attachment_list_id (integer) – первинний ключ списку вкладень;
- attachment_id (integer) – foreign key вкладення;
- contract_id (integer) – foreign key контракту;
- project_id (integer) – foreign key проекту;
- task_id (integer) – foreign key завдання.

3 РОЗРОБКА WEB-ДОДАТКУ ПІДТРИМКИ ДІЯЛЬНОСТІ ПРОЕКТНОГО МЕНЕДЖЕРА У КЗАПР

3.1 Архітектура web-додатку

Процес розробки web-додатку бере початок із проектування його архітектури. Web-додаток складається з двох основних модулів: серверу та клієнту. Перший відповідає за обробку запитів, які надсилає другий, та надання відповіді на них із бази даних. Тому сервер можна назвати провідником між БД та клієнтом [18].

Схема архітектури даного web-додатку представлена на рисунку 3.1.

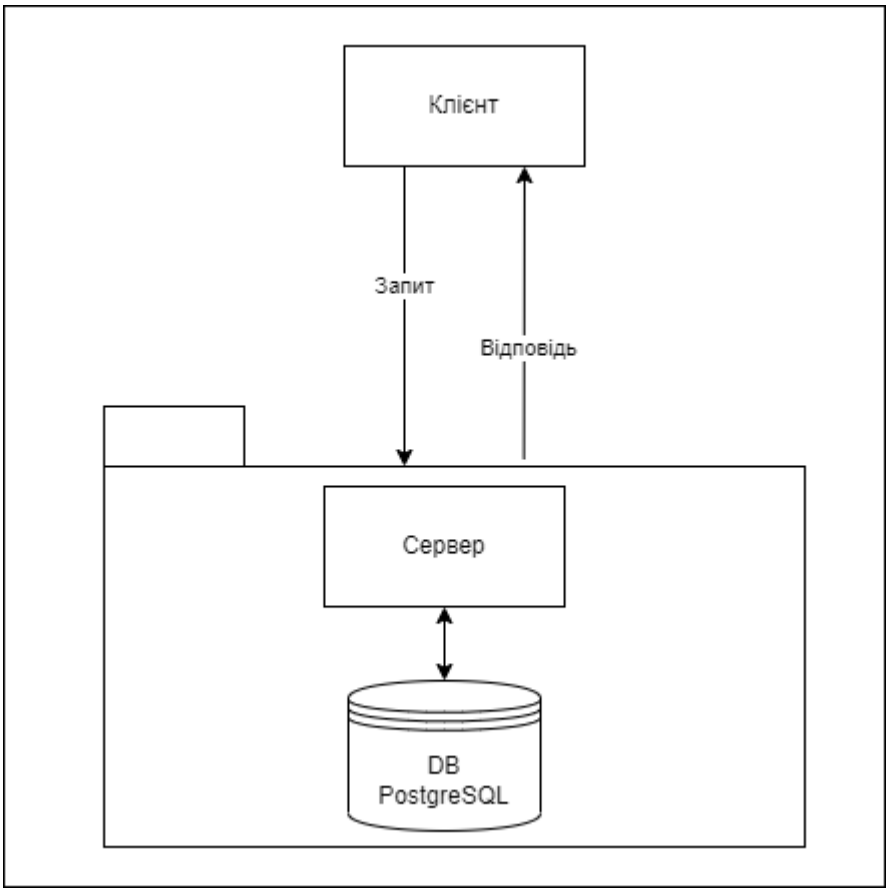


Рисунок 3.1 – Схема архітектури web-додатку

3.2 Розробка дизайну web-додатку

Після визначення архітектури web-додатку необхідно провести процес проектування його макету. Для розробки останнього треба визначитись із структурою сторінок даного програмного продукту, розміщенням основних функціональних блоків, а також із кольоровою гаммою. Орієнтуючись на вимоги до представленого проекту, які зазначені у додатку А, було розроблено відповідний макет web-додатку (рис.3.2).

Рисунок 3.2 – Макет web-додатку

У результаті аналізу дизайну типових web-додатків для управління проектами було реалізовано дизайн сторінки отримання інформації про всі завдання (рис. 3.3), сторінки їх створення та редагування (рис. 3.4), а також сторінки перегляду повідомлень (рис. 3.5).

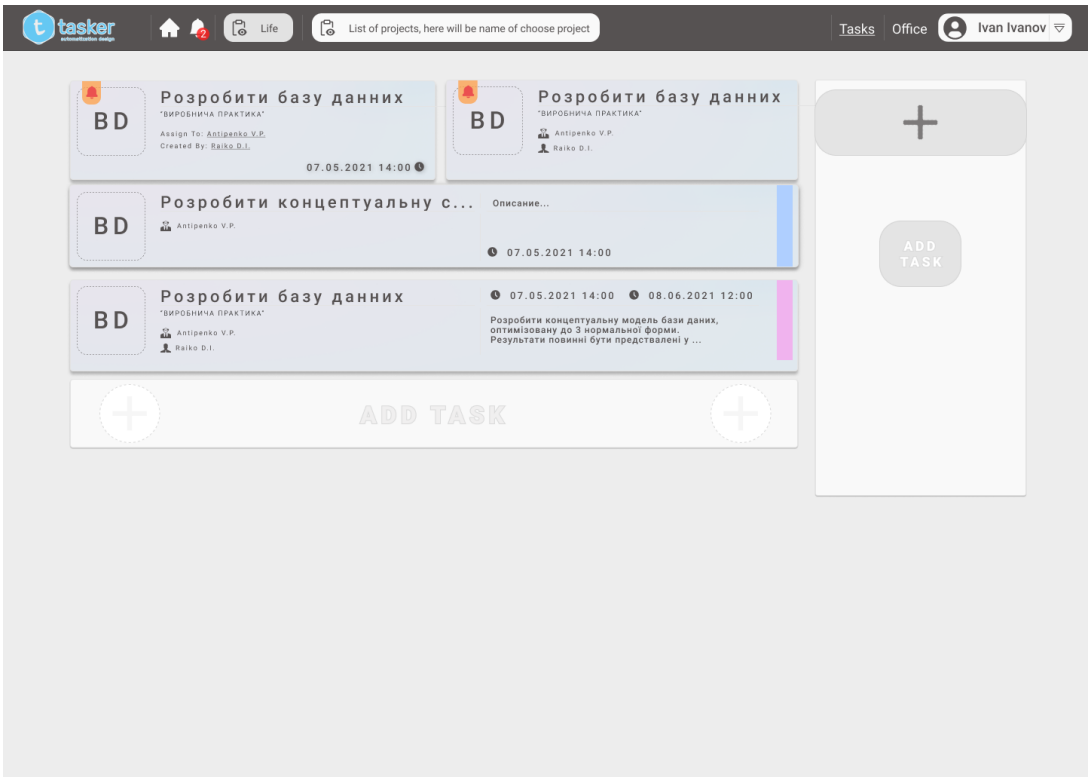


Рисунок 3.3 – Дизайн сторінки з завданнями для проектного менеджера

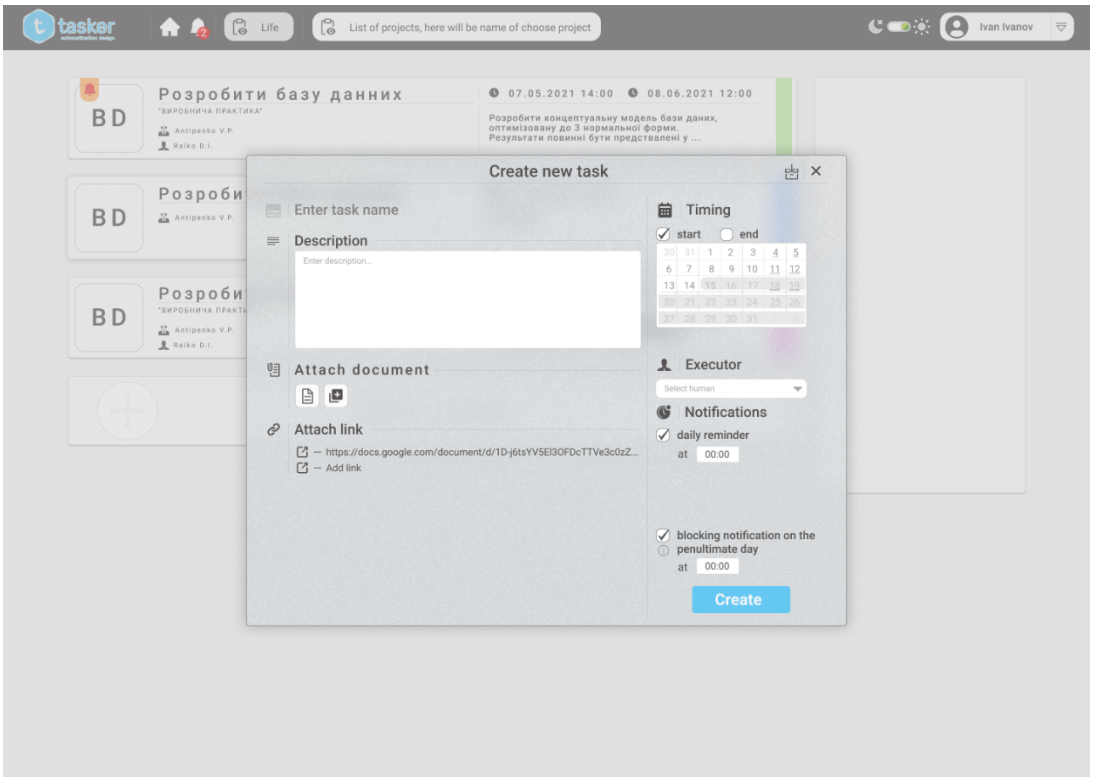


Рисунок 3.4 – Дизайн сторінки створення/редагування завдання

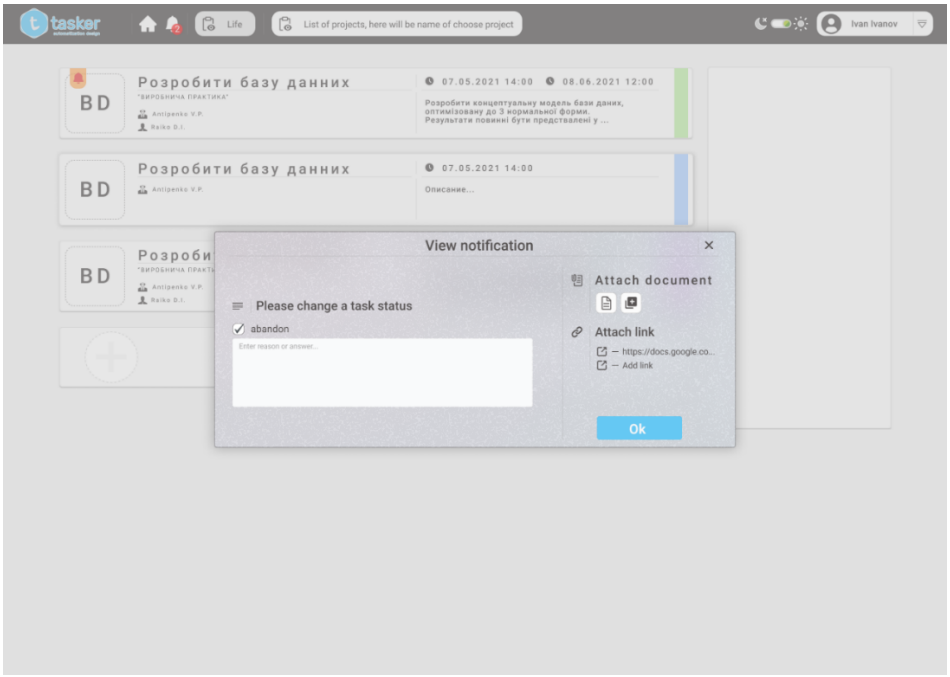


Рисунок 3.5 – Дизайн сторінки перегляду повідомлень

Для забезпечення індивідуальності даного web-додатку було розроблено його логотип, який зображений на рисунку 3.6. Він відповідає стилістиці дизайну та містить основні кольори (білий та блакитний). Шестикутник позначає шестерню. У свою чергу це символізує рух та автоматизацію.

Рисунок 3.6 – Логотип web-додатку

Головним шрифтом додатку було обрано шрифт Roboto.

Розташування блоків на web-сторінках (рис. 3.7) виконано з використанням бібліотеки MUI та ReactJS й інтерпретатора SASS для стилізації елементів у даній розробці.

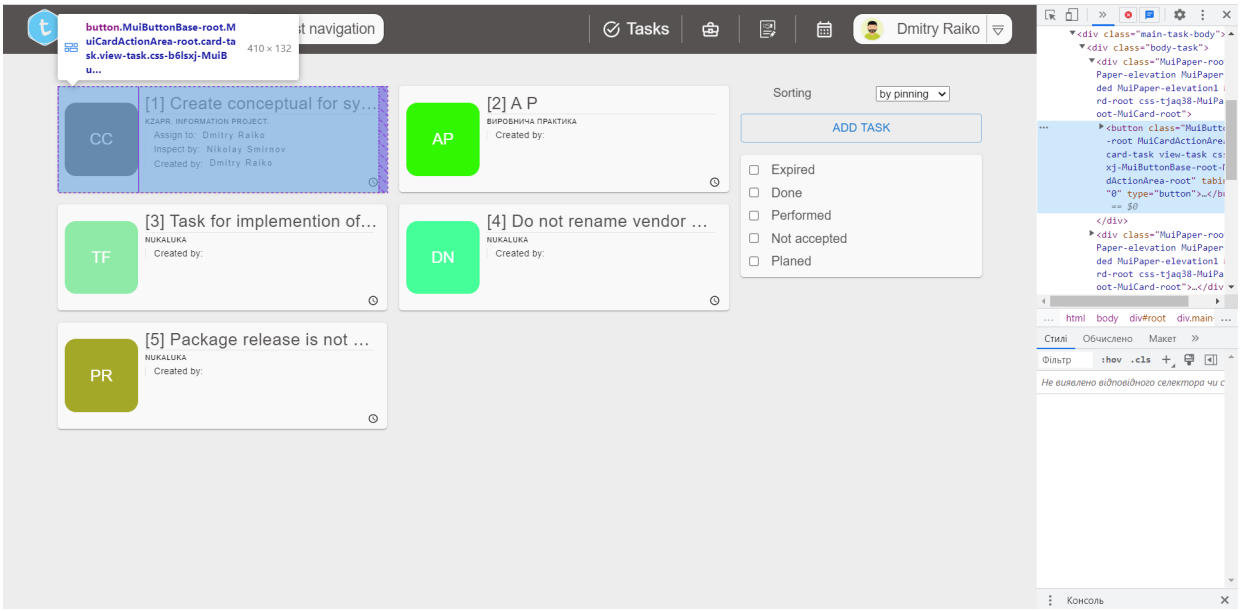


Рисунок 3.7 – Розташування блоків за допомогою бібліотеки MUI

3.3 Програмна реалізація web-додатку

Перший етап програмної розробки – це підготування середовища для реалізації web-додатку. Спочатку треба створити директорію на персональному комп’ютері. У консолі необхідно вибрати шлях до неї та за допомогою команди npx create-react-app «Tasker» встановити туди ReactJS. Після цього ініціюємо репозиторій в GitHub, використовуючи команду git init, та додатково встановлюємо на комп’ютер інтерпретатор SASS для CSS. Для написання програмного коду було обрано IDE WebStorm [19-20].

Контейнер головної сторінки для проектного менеджера з відображенням усіх завдань представлено на рисунку 3.9.

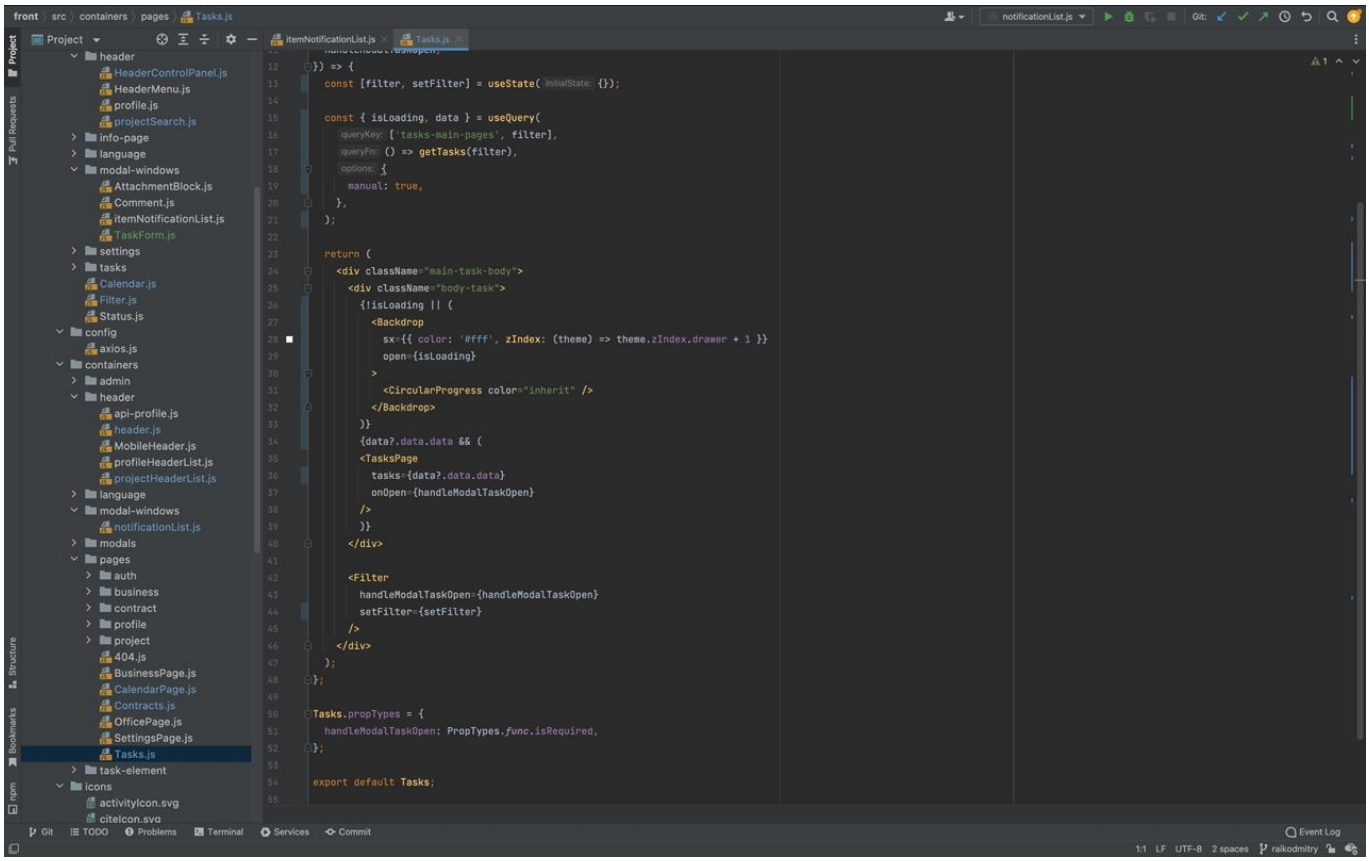


Рисунок 3.9 – Контейнер для відображення завдань
на головній сторінці web-додатку

Для надання повної інформації про завдання було розроблено модальне вікно. Приклад компоненту для цього процесу представлено на рисунку 3.10.

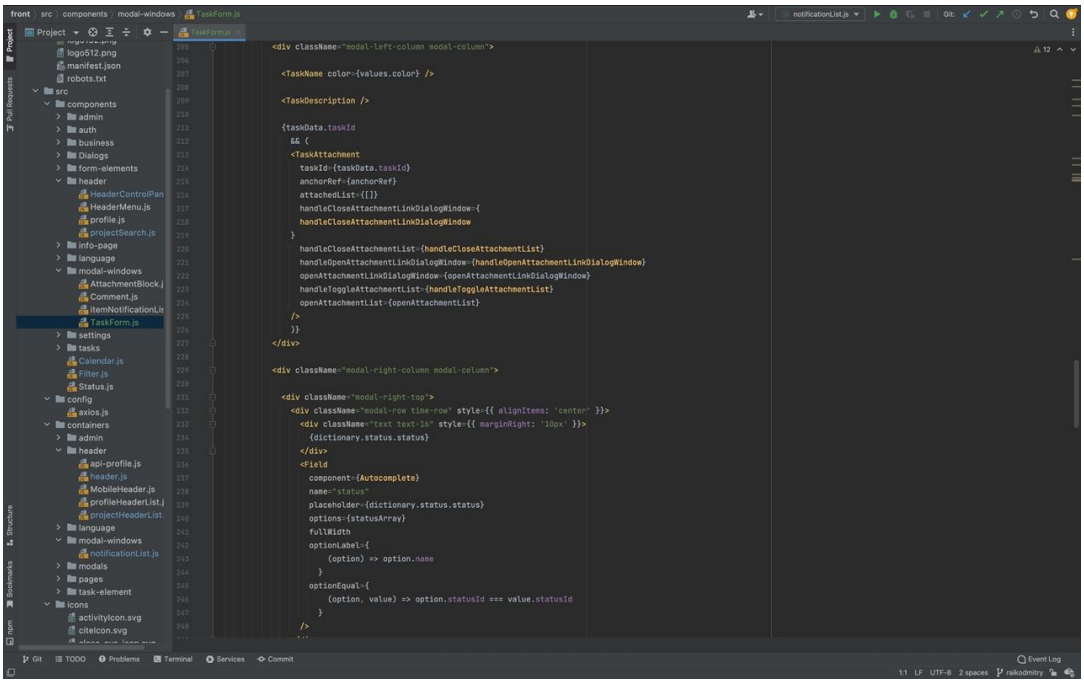


Рисунок 3.10 – Відображення модального вікна

Перегляд нового повідомлення в блоці з повідомленнями показане на рисунку 3.11.

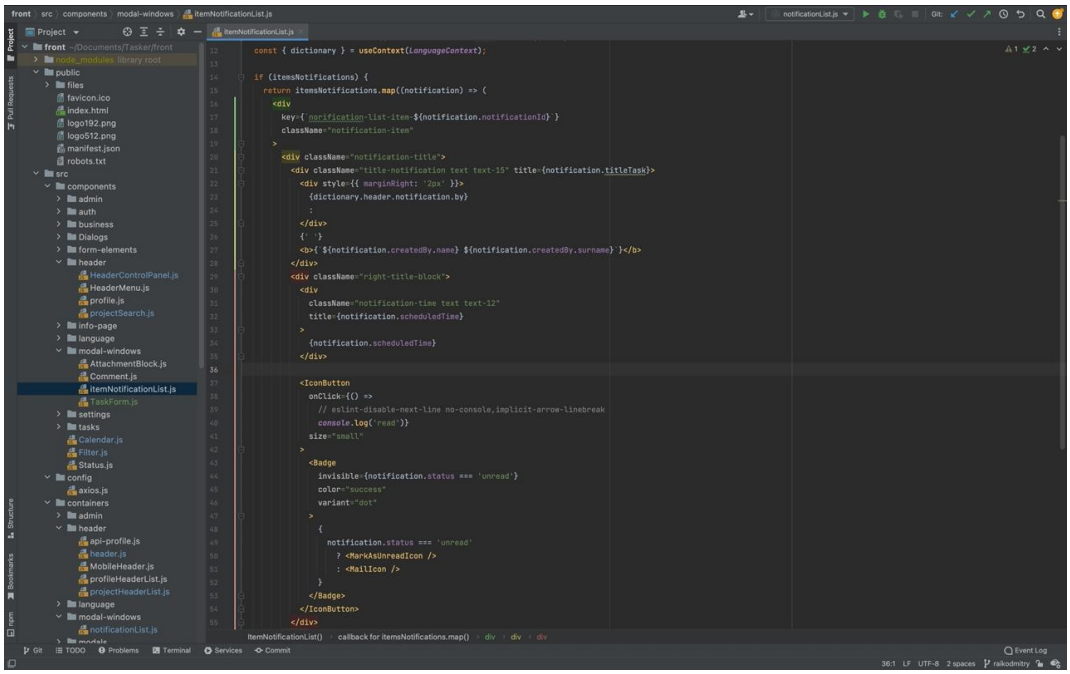


Рисунок 3.11 – Компонент для відображення
нового повідомлення у блоці повідомлень

Для зручної фільтрації завдань на головній сторінці було розроблено фільтр (рис. 3.12).

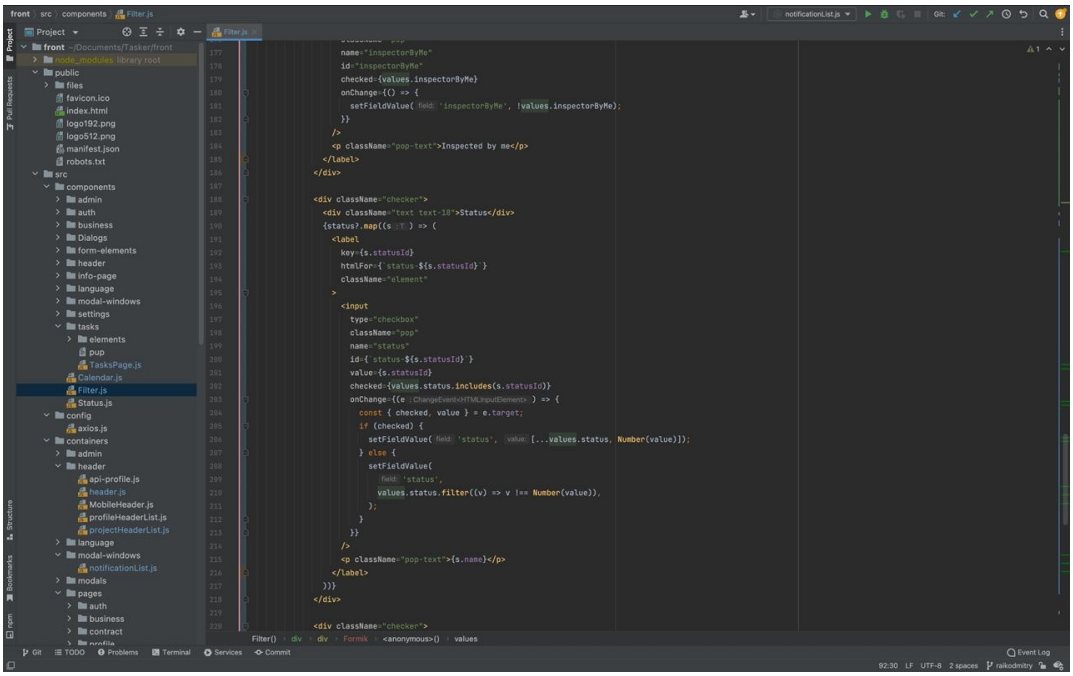


Рисунок 3.12 – Компонент фільтру завдань

Для швидкої навігації по даному web-додатку було розроблено відповідний блок із пошуком (рис. 3.13).

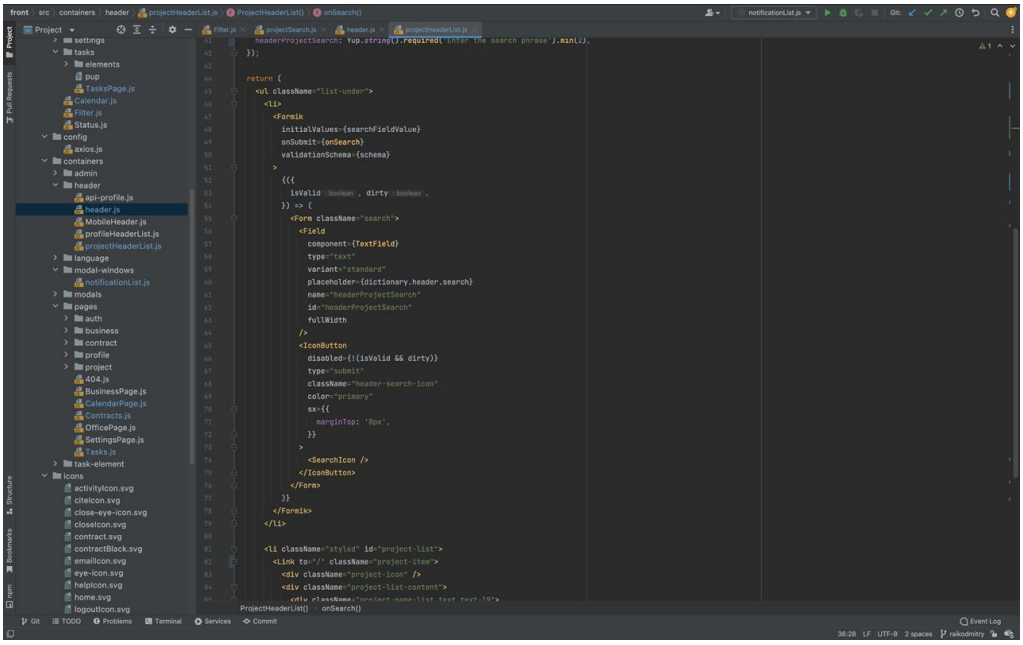


Рисунок 3.13 – Контейнер пошуку за web-додатком

Програмний код основних модулів web-додатку підтримки діяльності проектного менеджера у КЗАПР описано у Додатку Г.

3.4 Демонстрація роботи web-додатку

Головна сторінка web-додатку для проектного менеджера має меню із посиланнями на домашню сторінку, блок із повідомленнями, блок пошуку по ньому, посилання на календар та налаштування профілю (рис. 3.14-3.17). Основний контент сторінки складається з двох зон: зона відображення завдань, в яких залучений користувач, та засоби управління завданнями (фільтр, сортування та кнопка для створення нових).

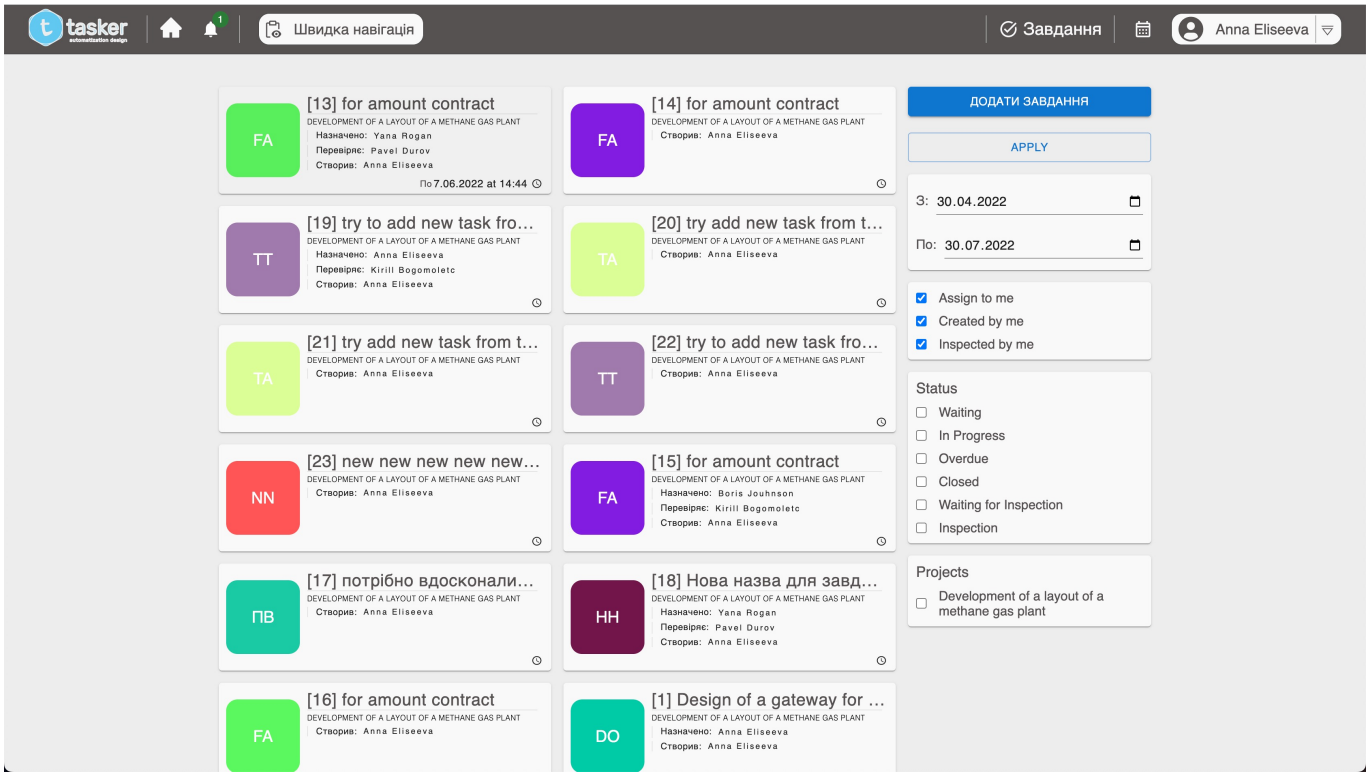


Рисунок 3.14 – Головна сторінка проектного менеджера у web-додатку

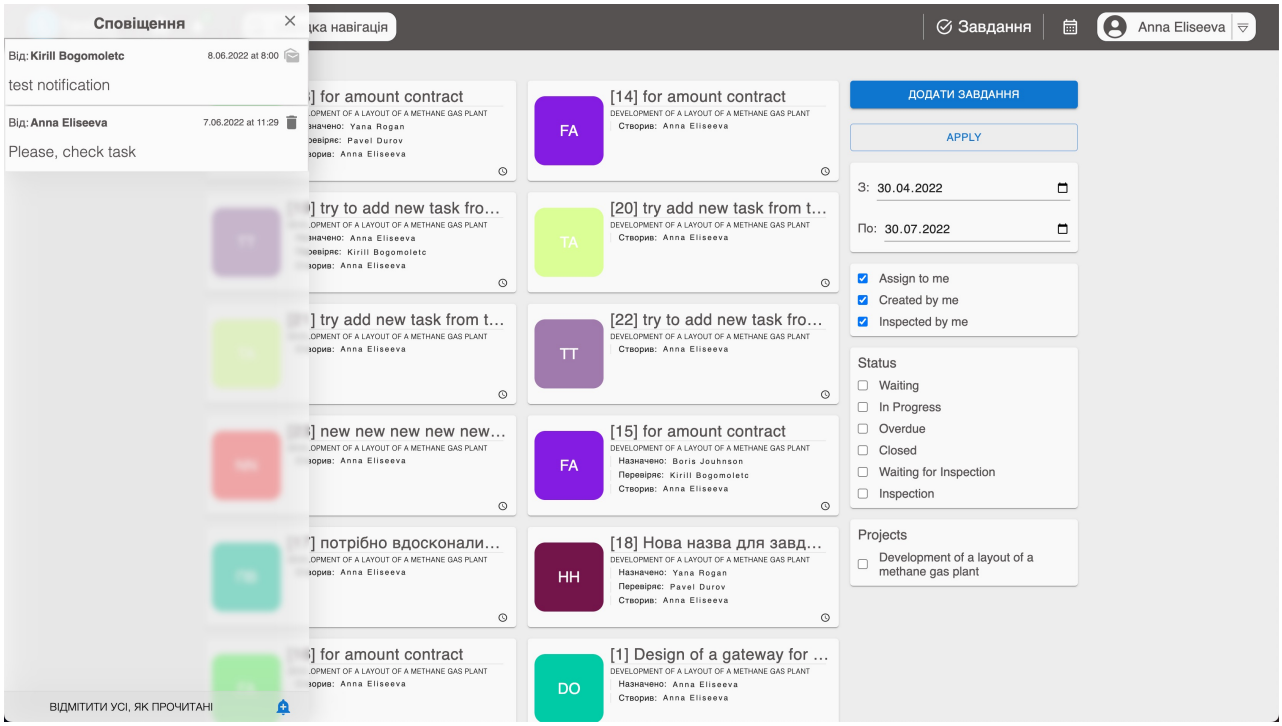


Рисунок 3.15 – Блок із повідомленнями

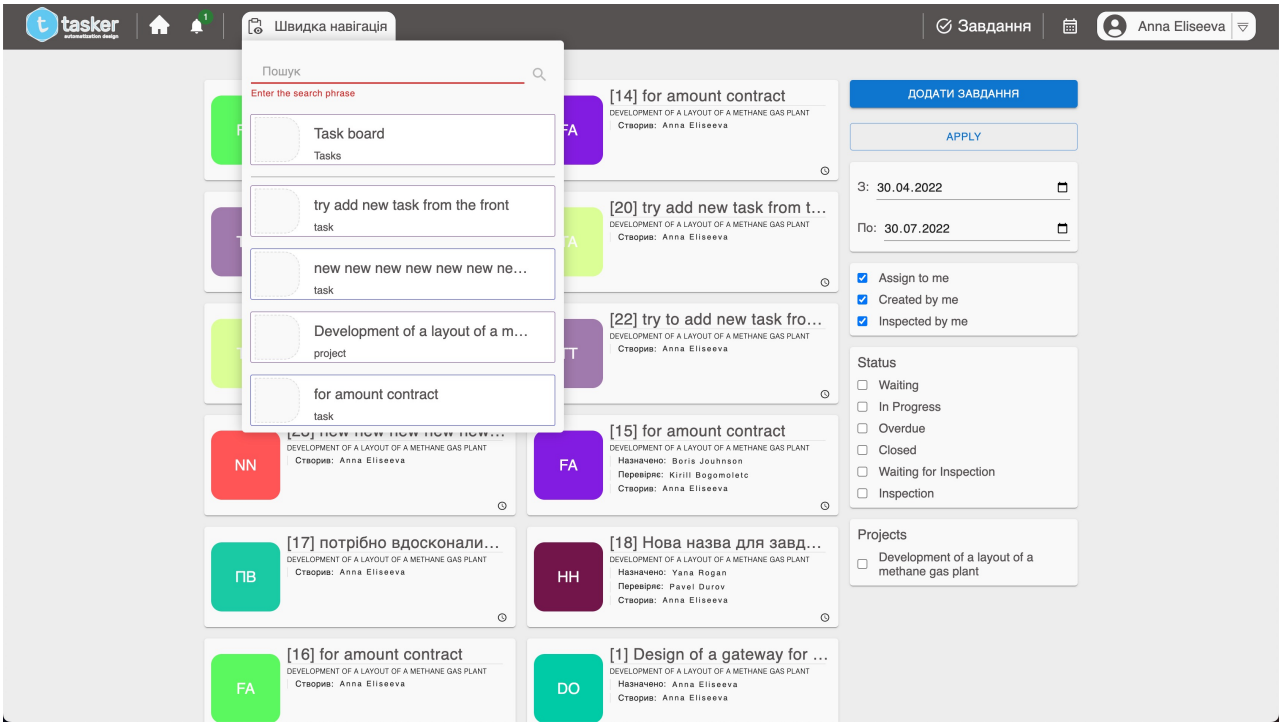


Рисунок 3.16 – Блок з пошуком по web-додатку

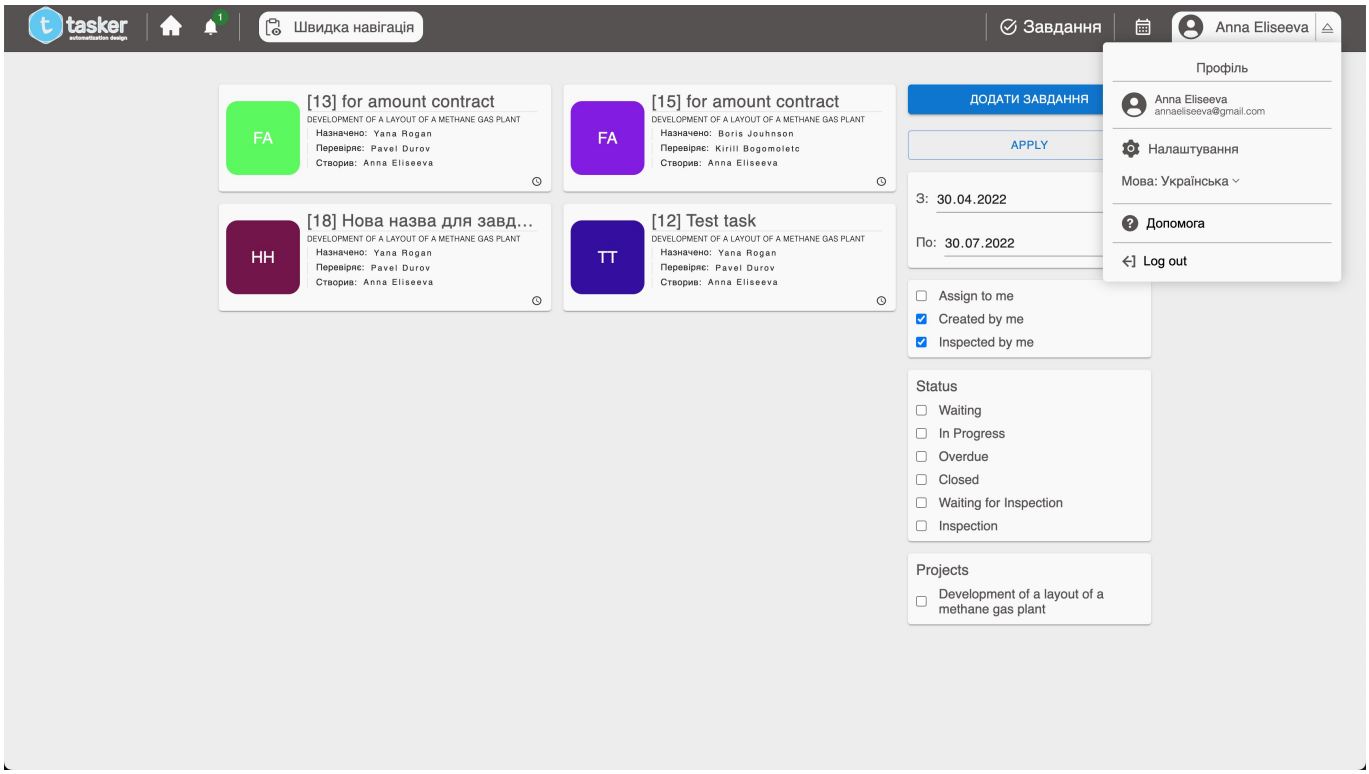


Рисунок 3.17 – Блок із налаштуваннями профілю web-додатку

Пункт меню «Повідомлення» (рис. 3.15) містить повідомлення, які мають певний статус, і функцію створення нового. Також доступним є функціонал «Mark all as read». Він дозволяє прочитати всі вхідні повідомлення. Приклад роботи даної функції зображено на рисунках 3.18-3.19. Модальне вікно створення повідомлення та результату його відправлення представлено на рисунках 3.20-3.21.

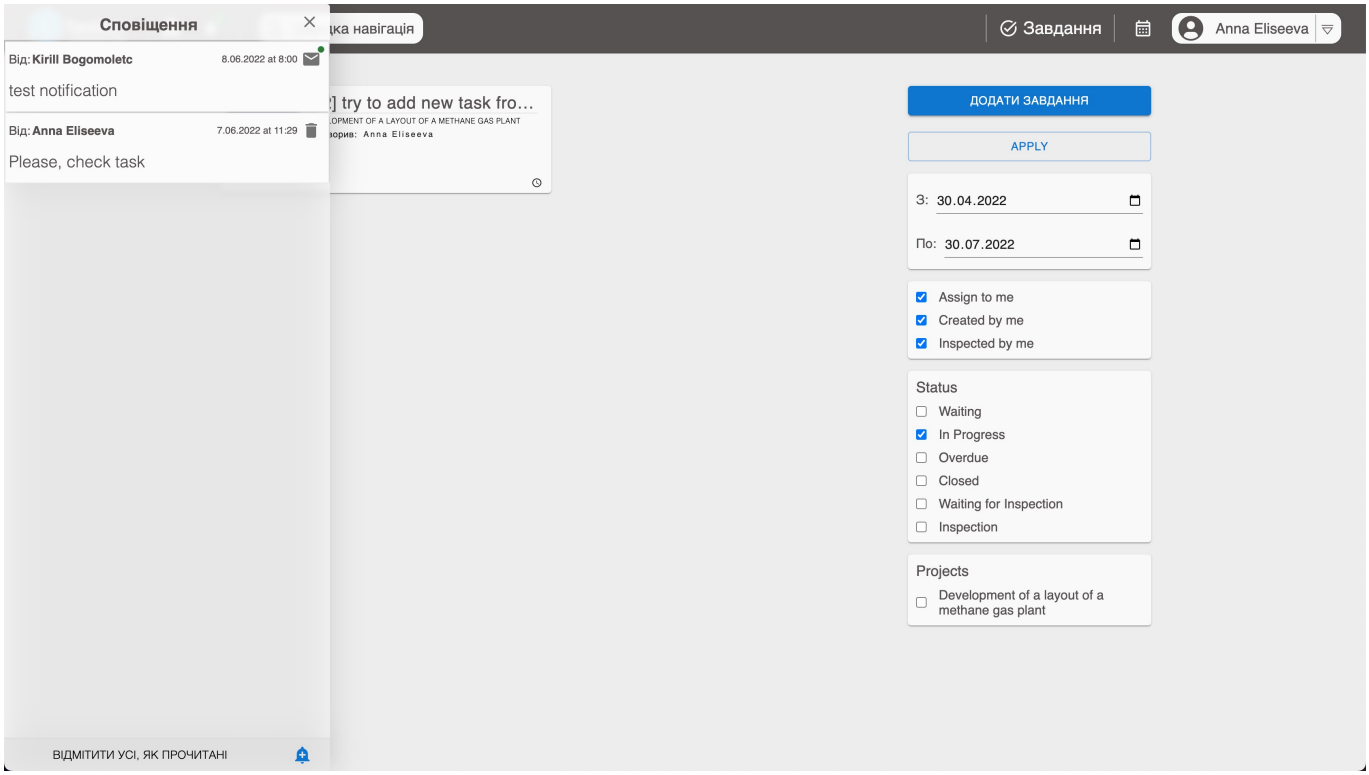


Рисунок 3.18 – Повідомлення, які мають статус непрочитані

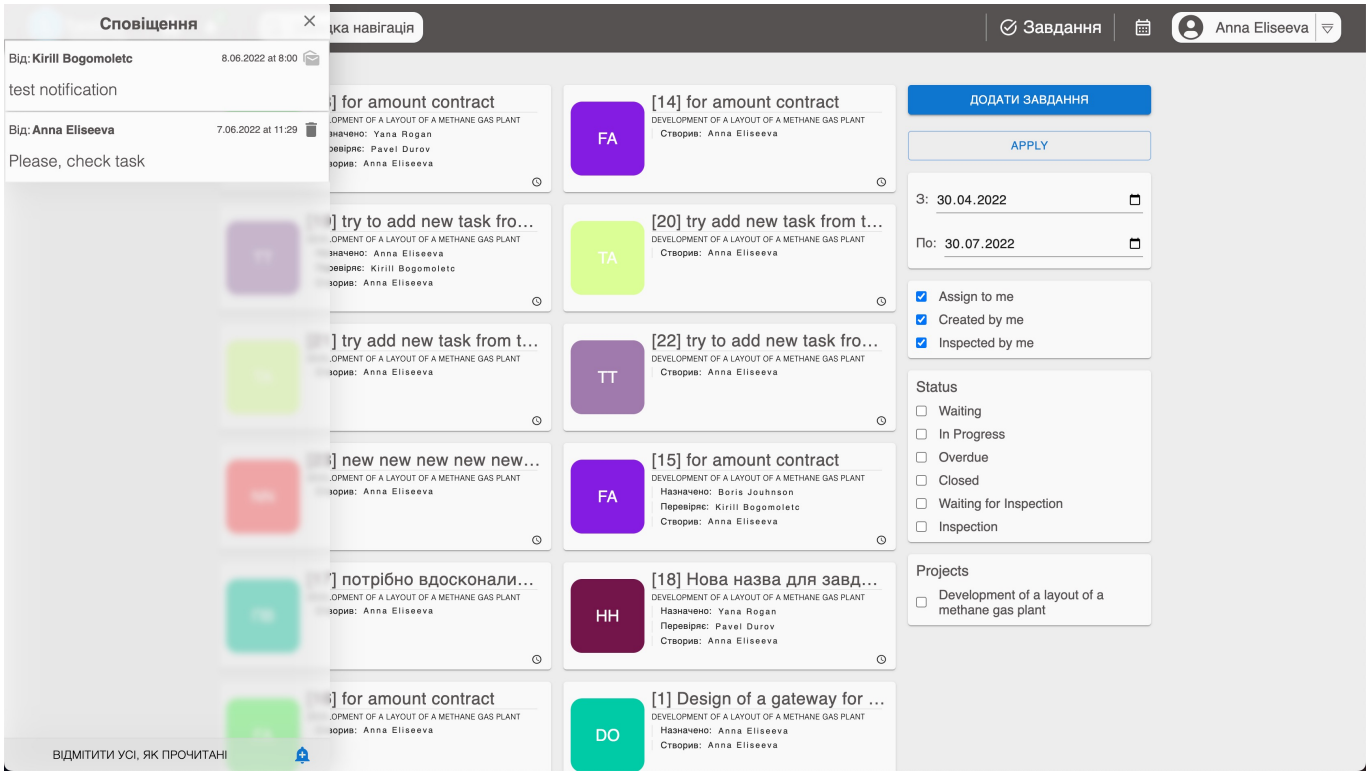


Рисунок 3.19 – Результат роботи функції «Mark all as read»

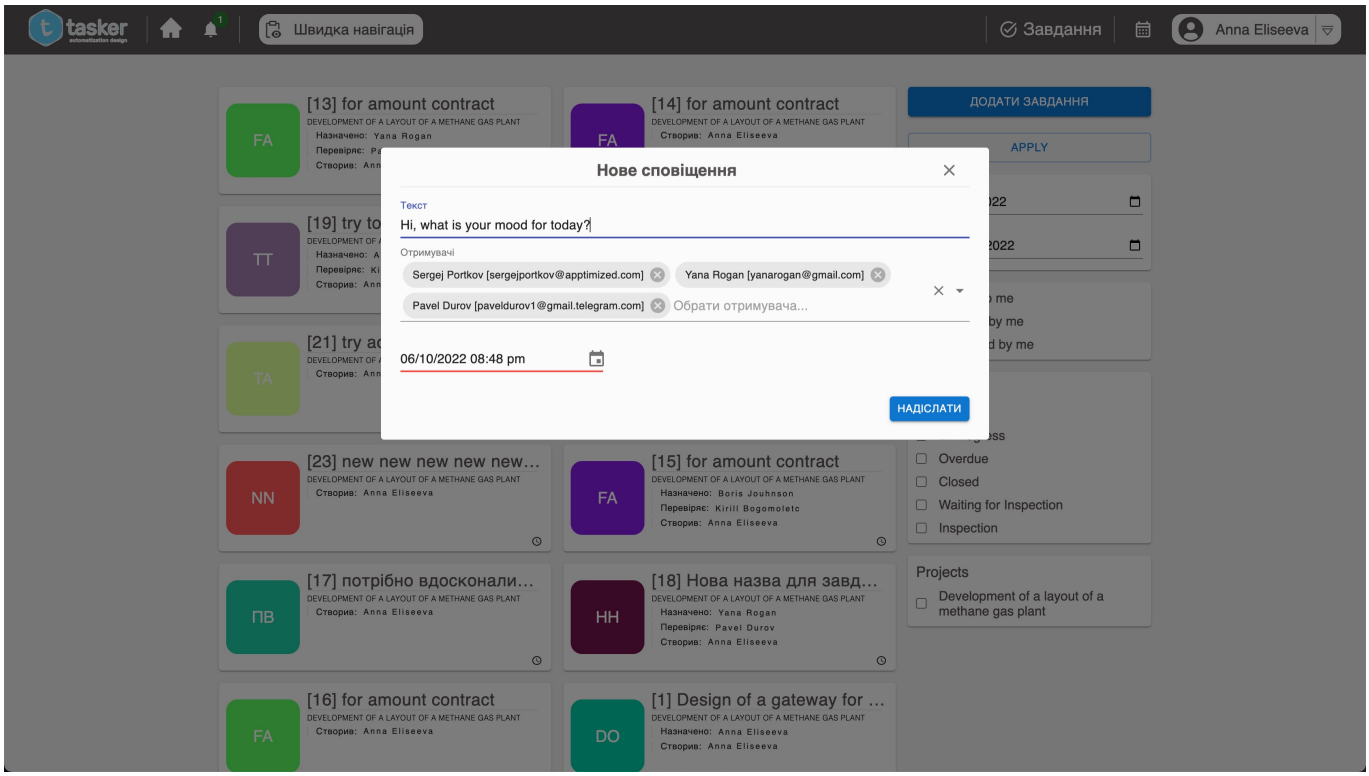


Рисунок 3.20 – Заповнена форма відправлення нового повідомлення користувачу

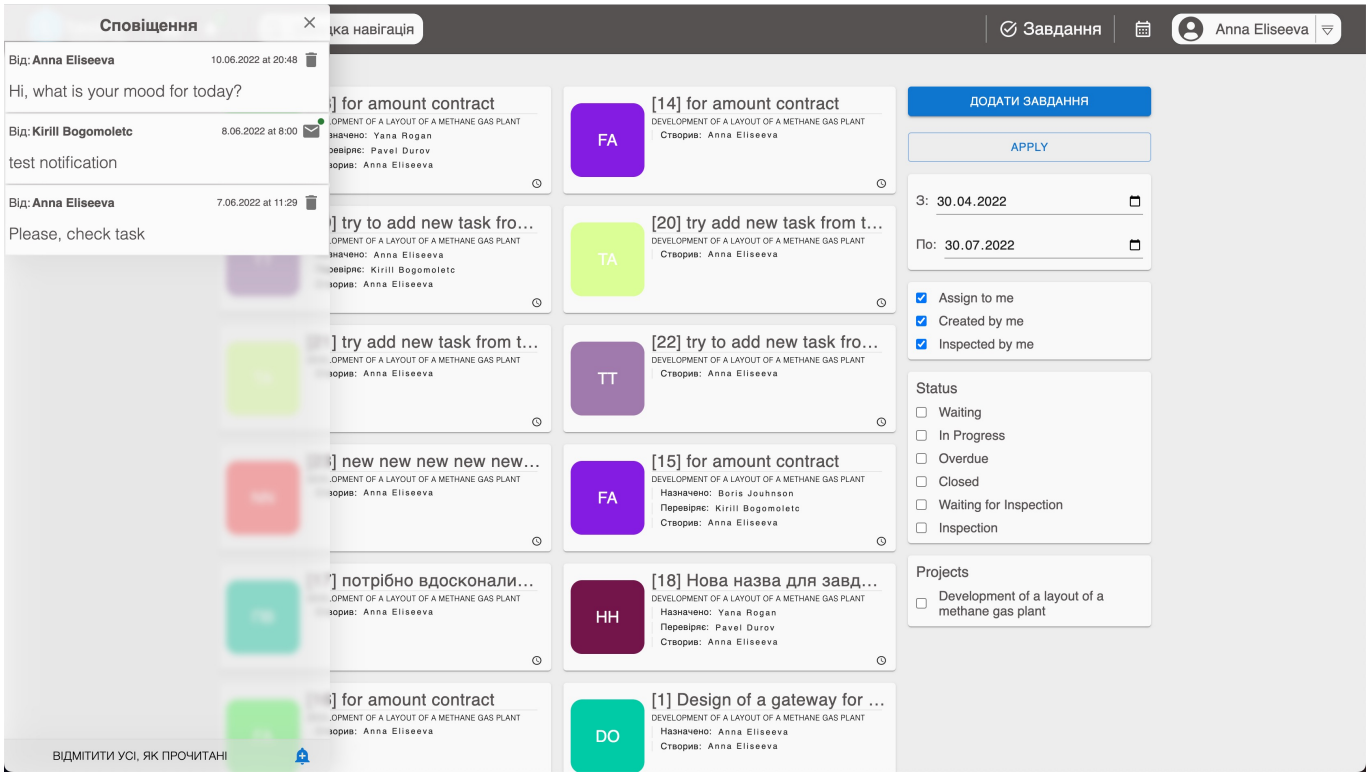


Рисунок 3.21 – Результат відправки нового повідомлення

Блок навігації по web-додатку містить поле для пошуку, а також пропозиції щодо вибору проекту (рис. 3.16). Наслідок даного процесу містить збіги щодо введеної у відповідне поле фрази (рис. 3.22). Якщо пошук не дав жодних результатів, то відображається перше стандартне посилання на сторінку з завданнями (рис. 3.23).

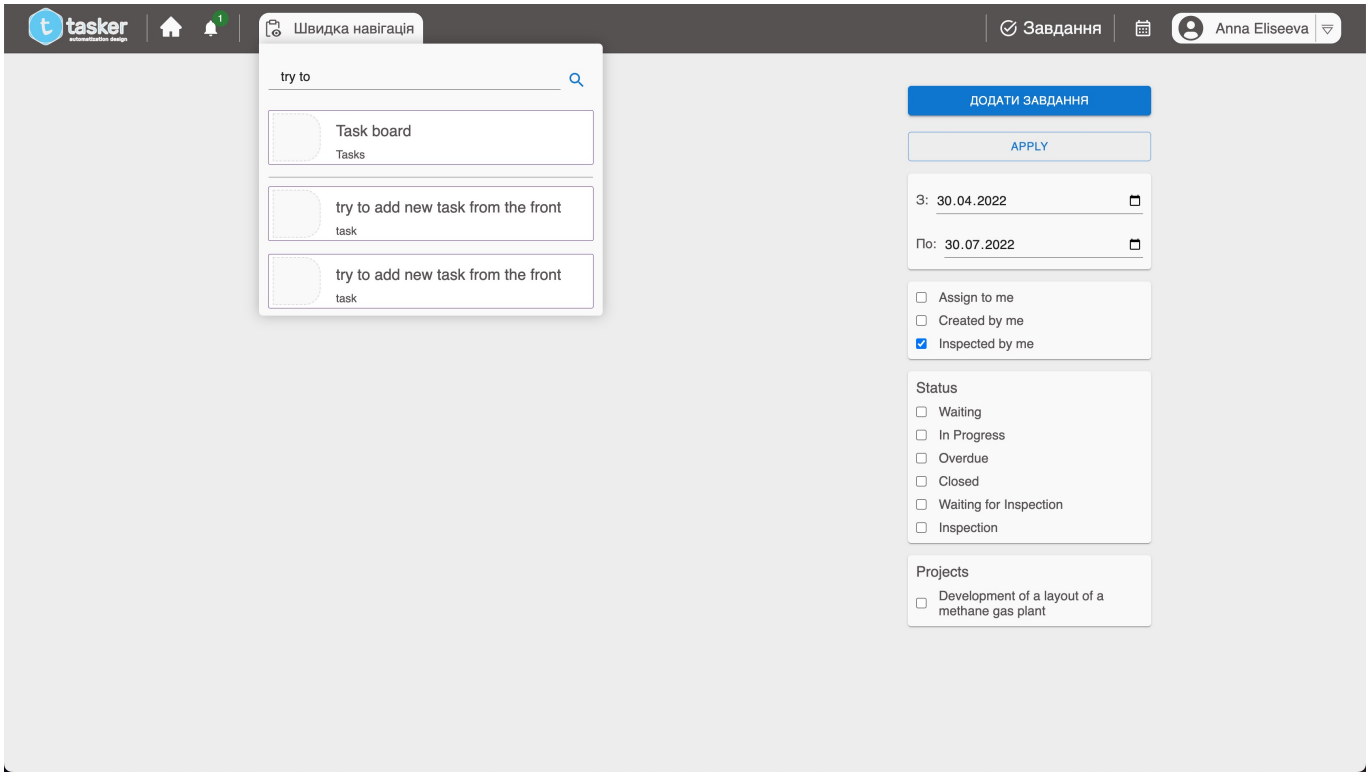


Рисунок 3.22 – Відображення результату пошуку інформації у відповідному полі

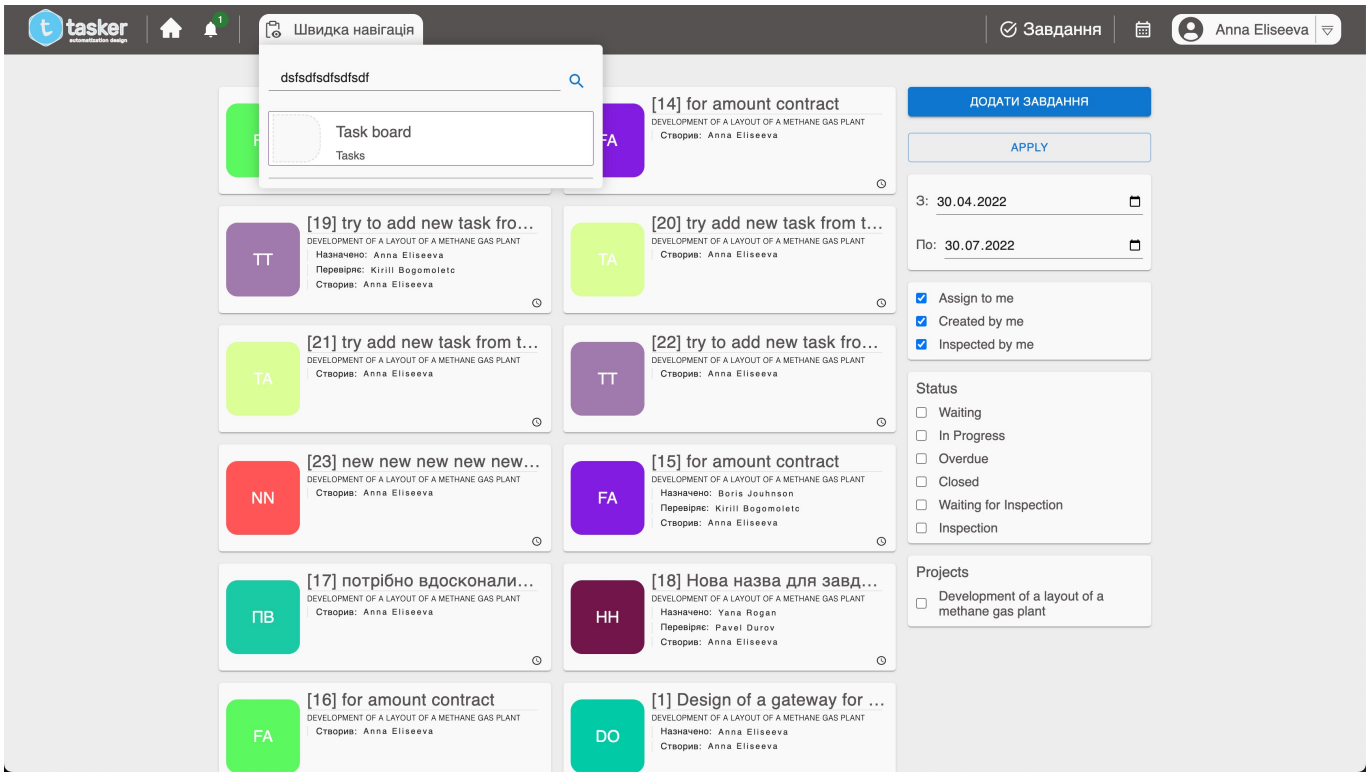


Рисунок 3.23 – Результат пошуку невалідних даних

Сторінка з завданнями дозволяє проектному менеджеру переглянути ті задачі, які виконують його підлеглі (рис. 3.14). Відповідні блоки містять основну інформацію про них. Це назва, часові рамки виконання, до якого проекту відноситься, кому назначене завдання, хто буде його перевіряти та хто його створив. Також можна прикріпити вкладення до задачі. Для цього треба натиснути на кнопку «Attach». Потім обрати тип вкладення – посилання або файл. І в залежності від вибору завантажити останній або вставити лінк й натиснути на кнопку «Add». Додання посилання зображене на рисунках 3.24-3.25. Для редагування завдання необхідно натиснути на блок із ними й обрати поле, щоб змінити дані. Потім треба клікнути на нього. Під час редагування поле підкреслюється синім кольором (рис. 3.26).

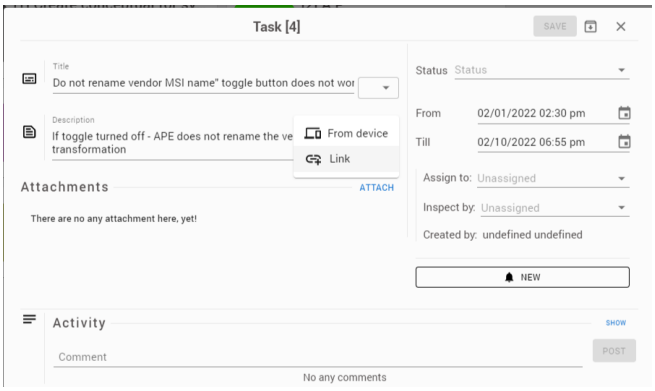


Рисунок 3.24 – Обрання типу вкладення «Link»

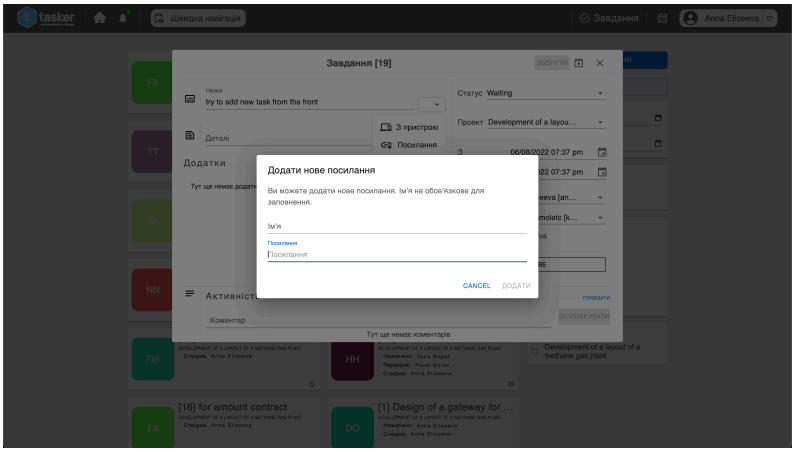


Рисунок 3.25 – Додання нового посилання

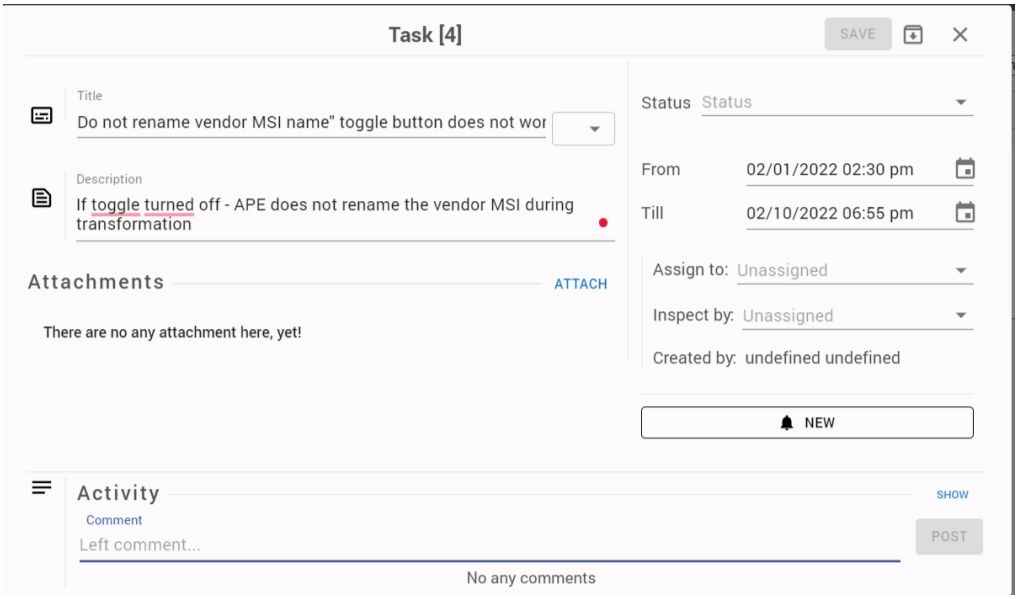


Рисунок 3.26 – Редагування поля «Comment»

Для створення нового завдання необхідно натиснути на кнопку «Add task», яка знаходиться в правій частині сторінки «Завдання». Після клікання відкриється модальне вікно з полями, які необхідно буде заповнити (рис. 3.27).

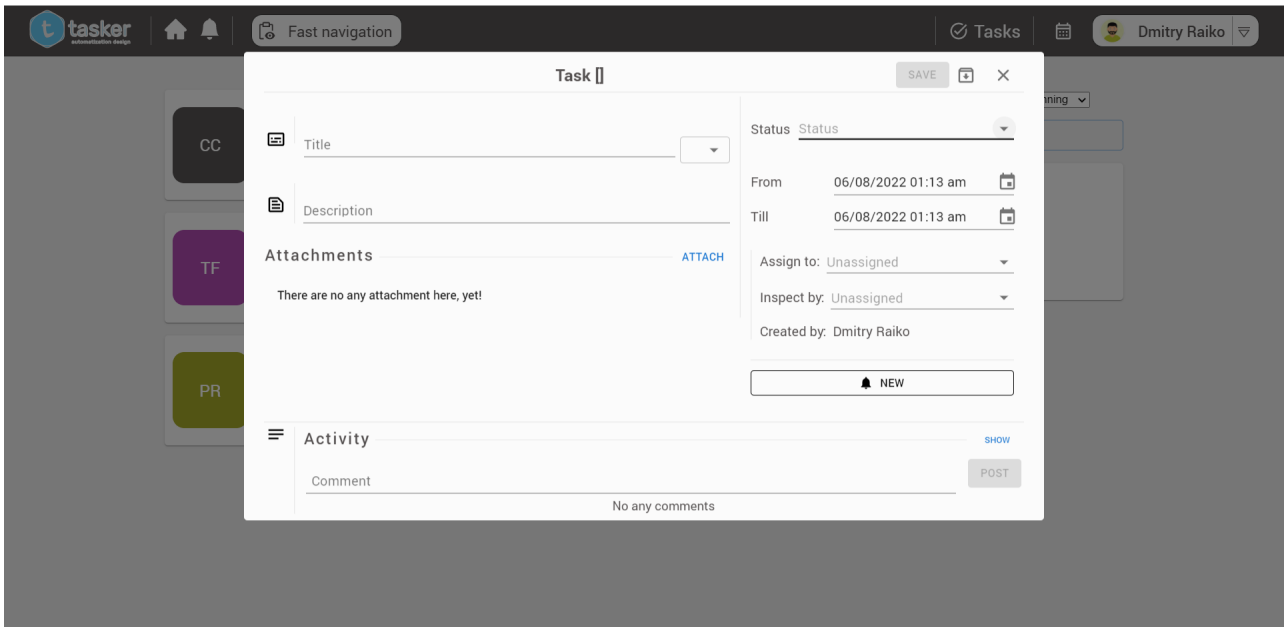


Рисунок 3.27 – Модальне вікно створення завдання

Для збереження змін треба натиснути на кнопку «Save». Вона буде активною тільки після того, як проектний менеджер заповнить всі обов'язкові поля для завдання – це «Title» та «Status». Після додавання нового завдання воно стає доступним серед списку поточних задач.

Також наявною є функція сортування. Це фільтрація завдань за статусом. Вона є досить корисною. Для цього необхідно обрати статус, за яким треба виконати сортування та звернути увагу на список завдань. Принцип та результат роботи фільтру зображений на рисунках 3.28-3.29.

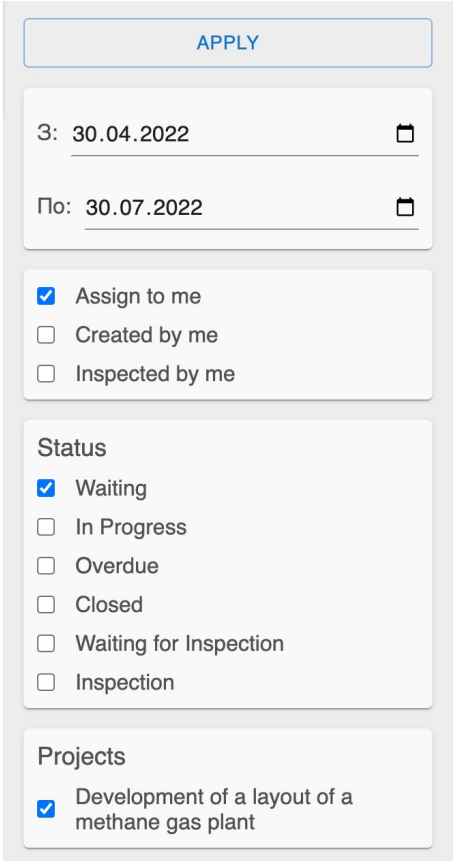


Рисунок 3.28 – Фільтр із обраними статусами на сторінці «Завдання»

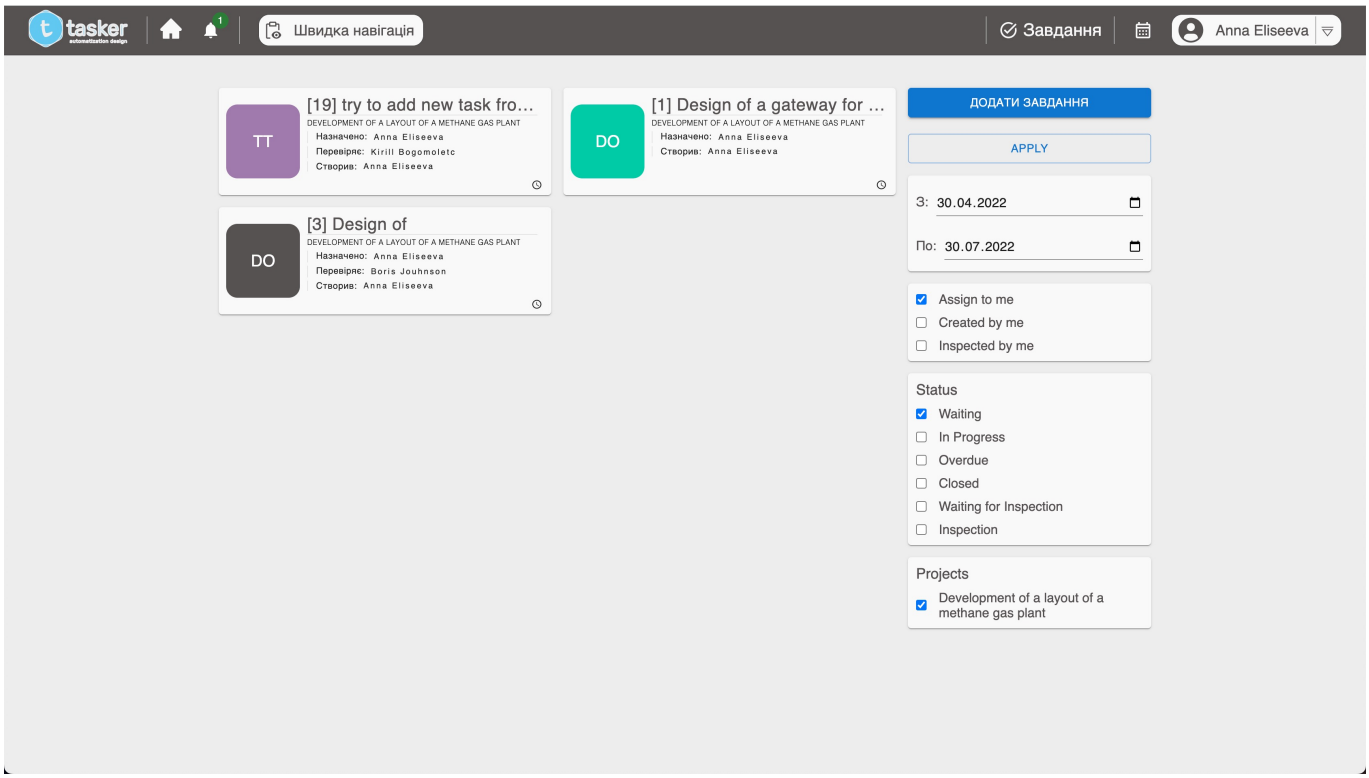


Рисунок 3.29 – Результат роботи фільтру

Сторінка «Календар» у своїй структурі має кнопки навігації по ньому, кнопки зміни його вигляду, а також саме поле відображення завдань у формі календаря (рис. 3.30).

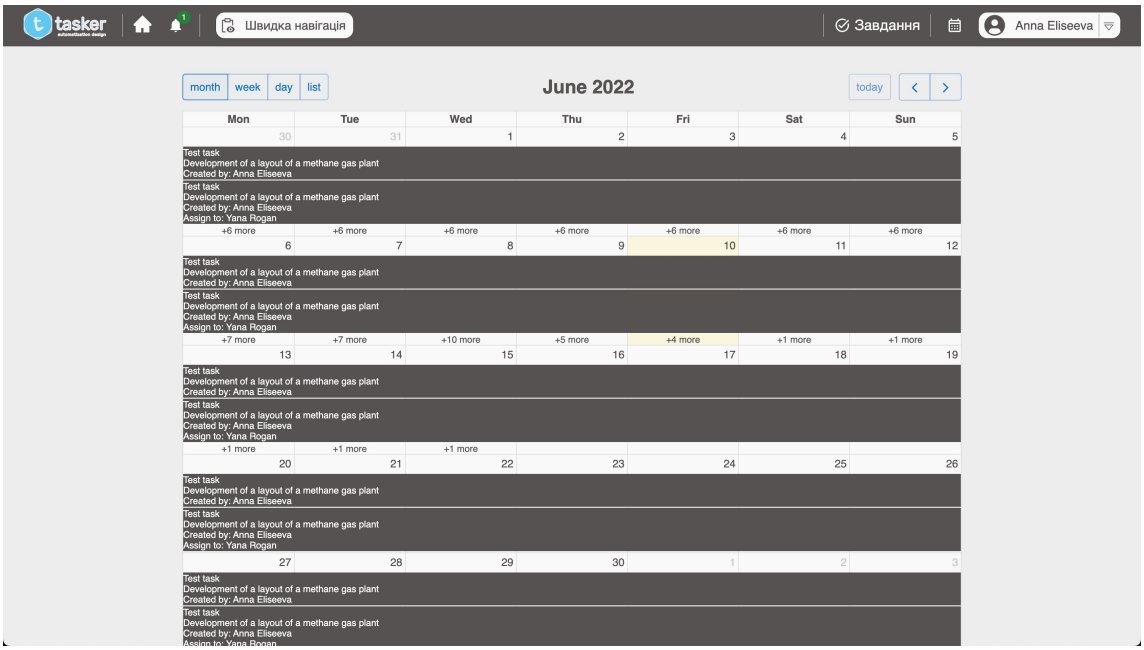


Рисунок 3.30 – Сторінка «Календар» у режимі відображення за місяцем

Завдання у календарі містить наступну інформацію: власну назву, назву проекту, до якого воно відноситься, хто створив та хто є виконавцем. Задачі відображаються на інтервалі, протягом якого треба їх зробити. Для перегляду завдання необхідно натиснути на поле з його назвою. У результаті відкривається модальне вікно, в якому користувач може переглянути або відредагувати інформацію.

На рисунках 3.31, 3.32, та 3.33 зображено відображення календаря у трьох різних режимах – за тиждень, за день та список завдань.

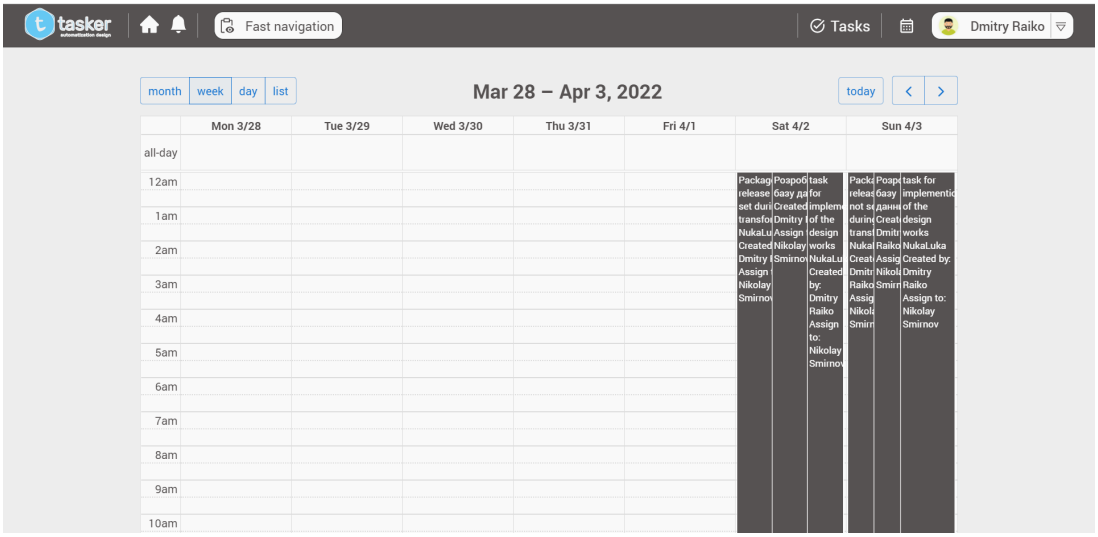


Рисунок 3.31 – Сторінка «Календар» у режимі відображення задач за тиждень

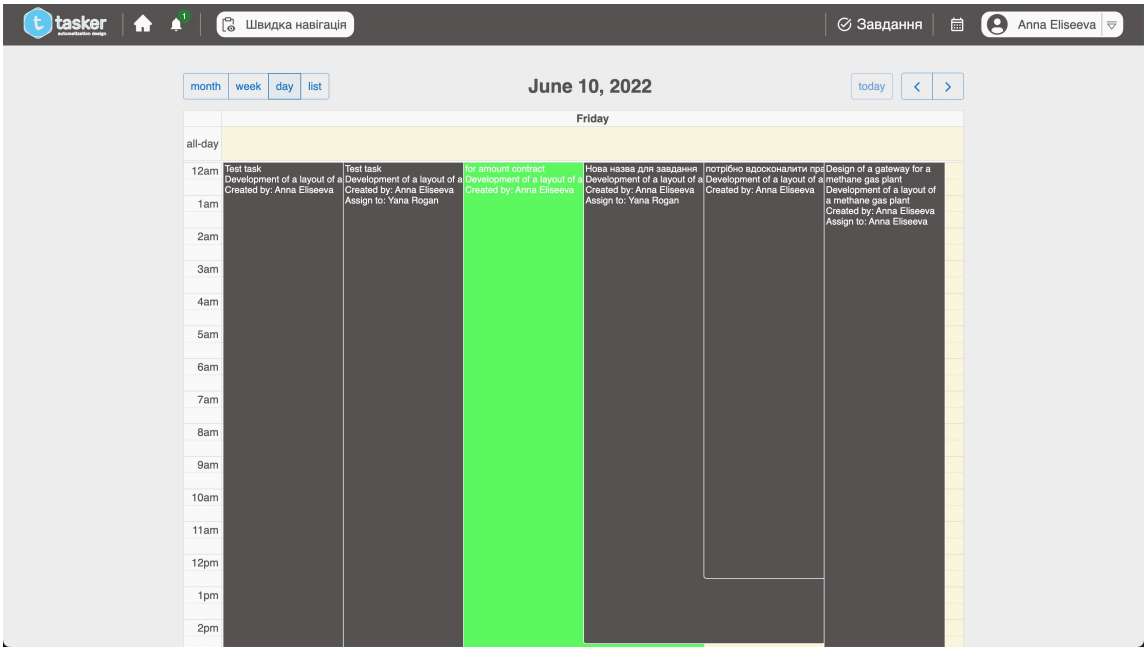


Рисунок 3.32 – Сторінка «Календар» у режимі відображення завдань за день

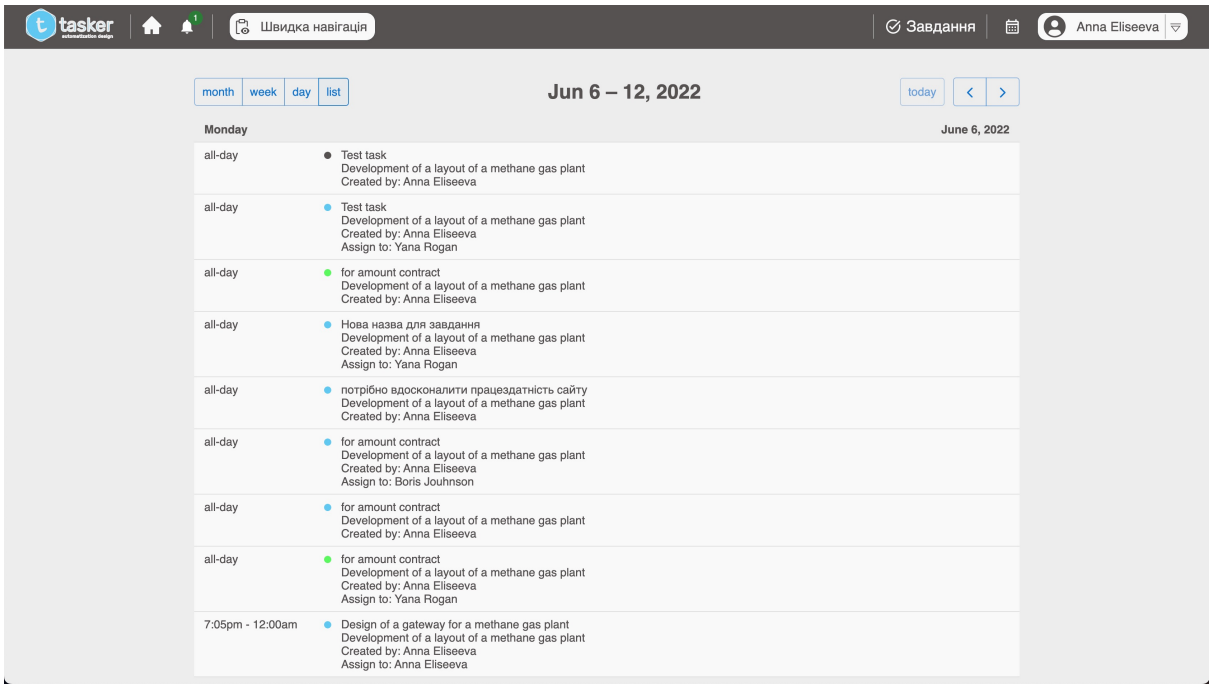


Рисунок 3.33 – Сторінка «Календар» у режимі відображення задач списком

Якщо завдання на певну дату відсутні, на сторінці «Календаря» відображається повідомлення, що на цей день немає запланованих задач (рис. 3.34).

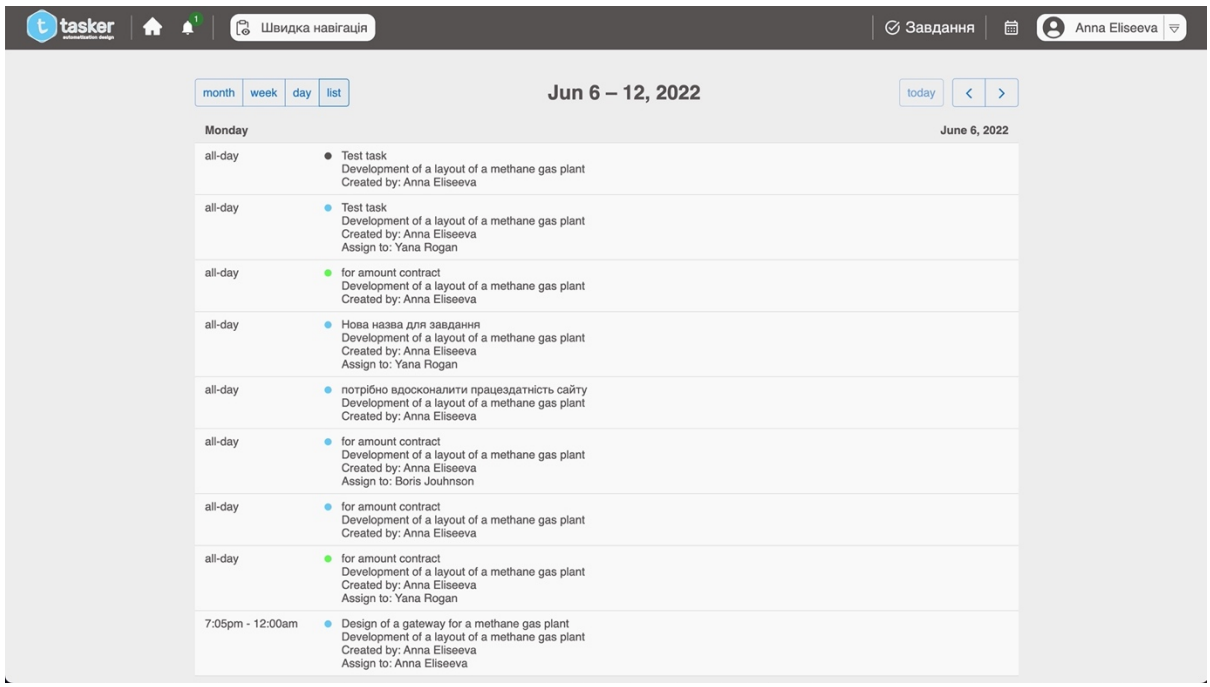


Рисунок 3.34 – Відображення сторінки «Календар»
без завдань на певну дату

Сторінка «Профіль» містить повну інформацію про співробітника. Це його контактні дані, логотип, посада, керівник та список завдань, які пов’язані з цією людиною (рис. 3.35). Останні можна переглянути, натиснувши на них.

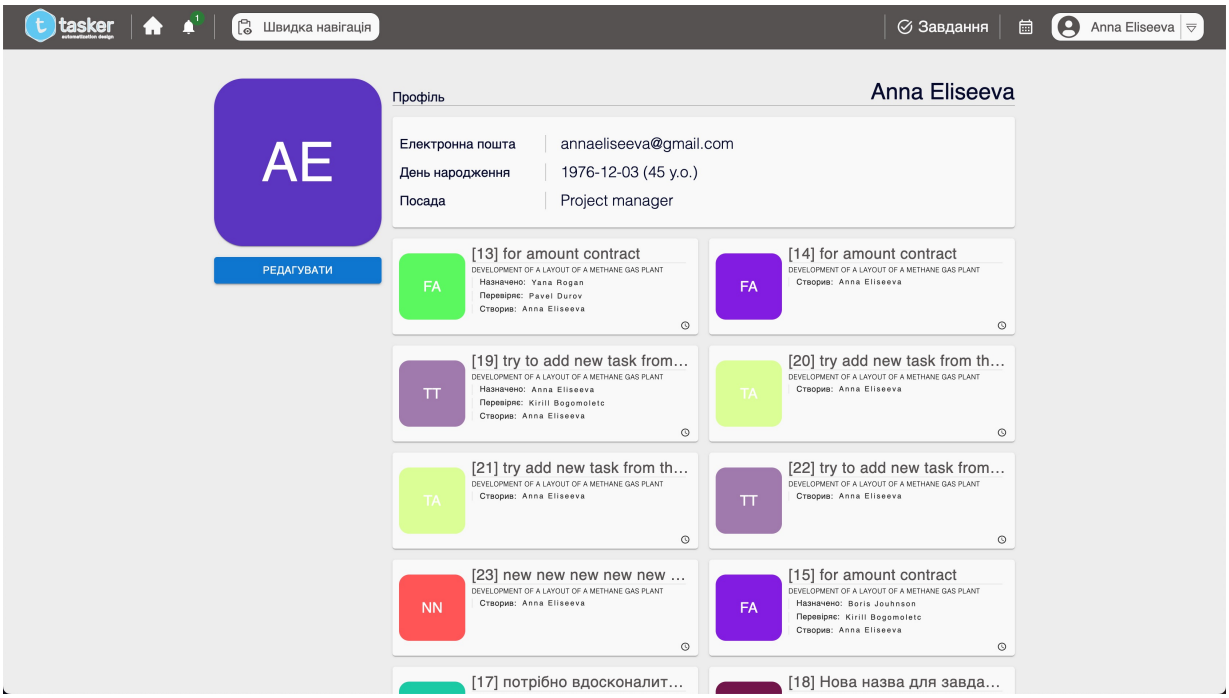


Рисунок 3.35 – Сторінка «Профілю» користувача

Також можна відредагувати профіль користувача. Для цього на відповідній сторінці треба натиснути на кнопку «Edit». Форма редагування інформації профілю містить наступні поля: ім’я, прізвище, дата народження та поле «Про себе». Після маніпуляцій з даними необхідно натиснути на кнопку «Save» (рис. 3.36).

Також для профілю можна встановити аватар. Для цього треба спочатку натиснути на кнопку «Set new photo». Потім необхідно обрати фото на комп’ютері й обрізати його під певний розмір. Заключний крок – це натиснути на кнопку «Set» (рис. 3.37-3.38).

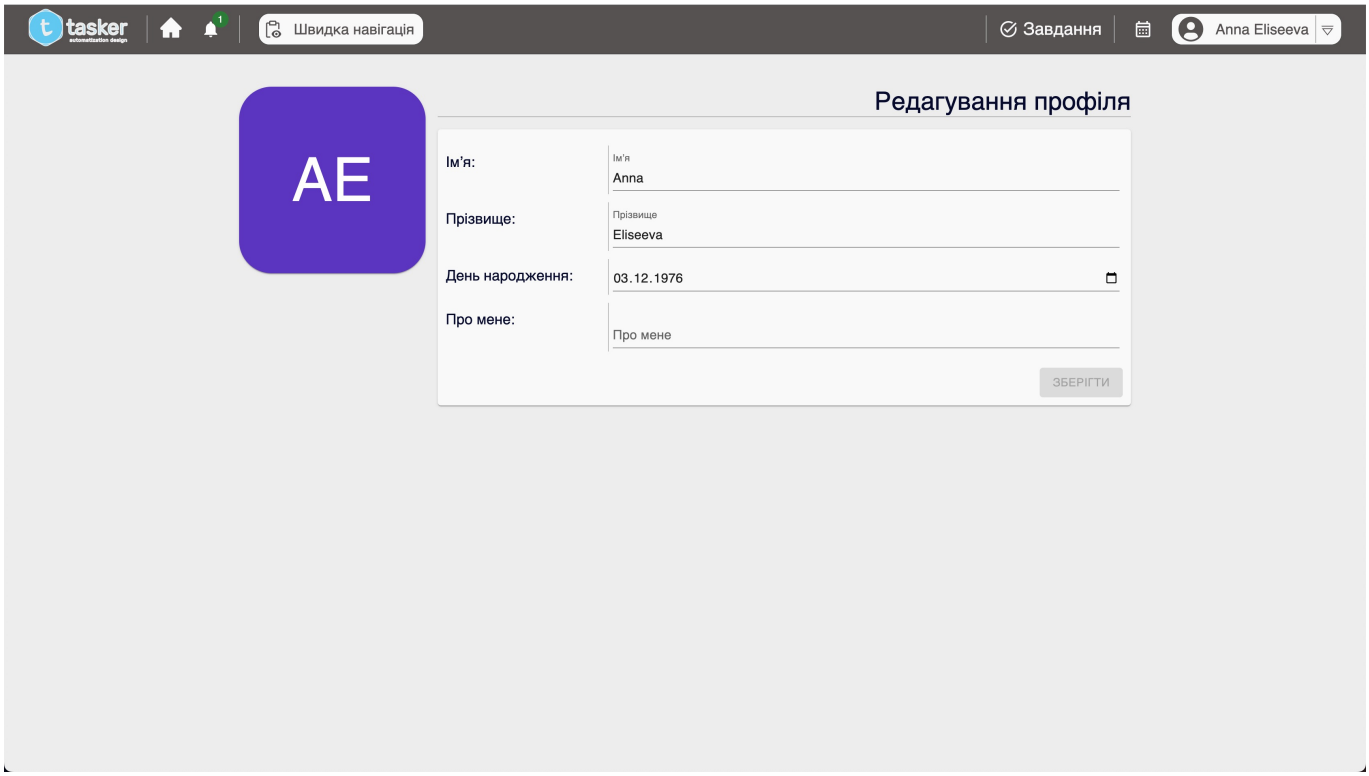


Рисунок 3.36 – Форма редагування інформації профілю користувача

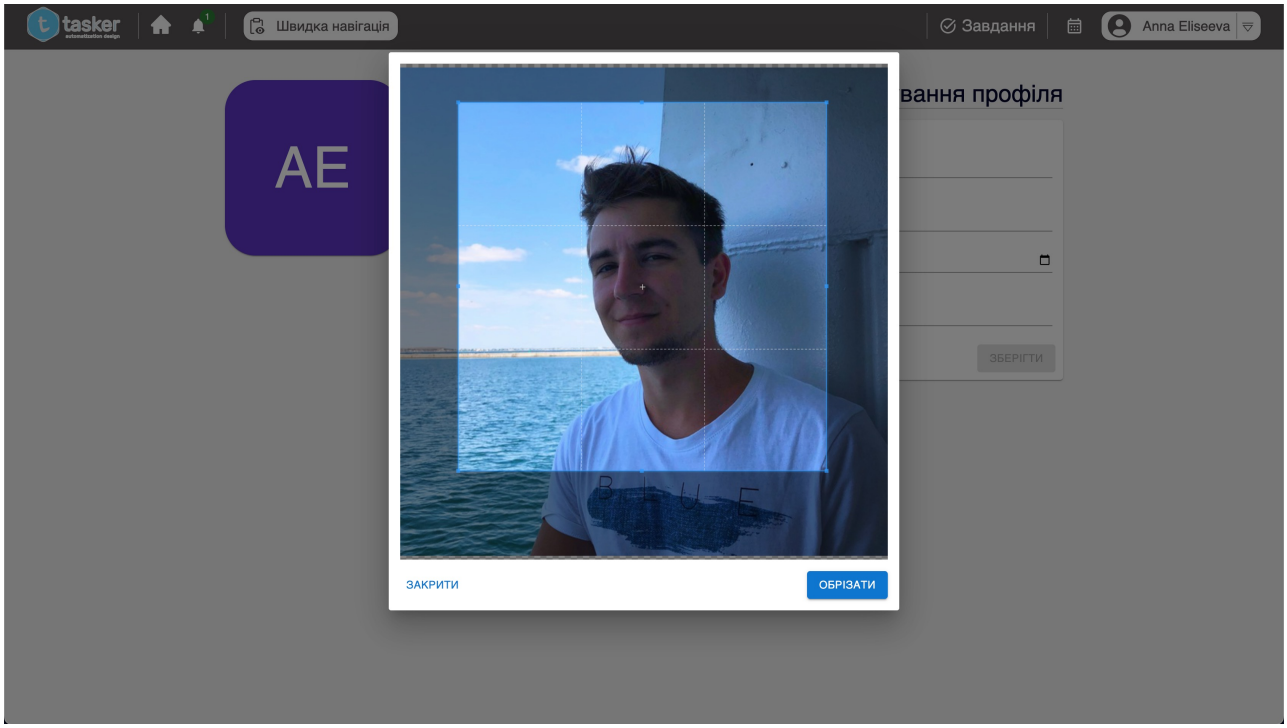


Рисунок 3.37 – Форма встановлення нового фото профілю

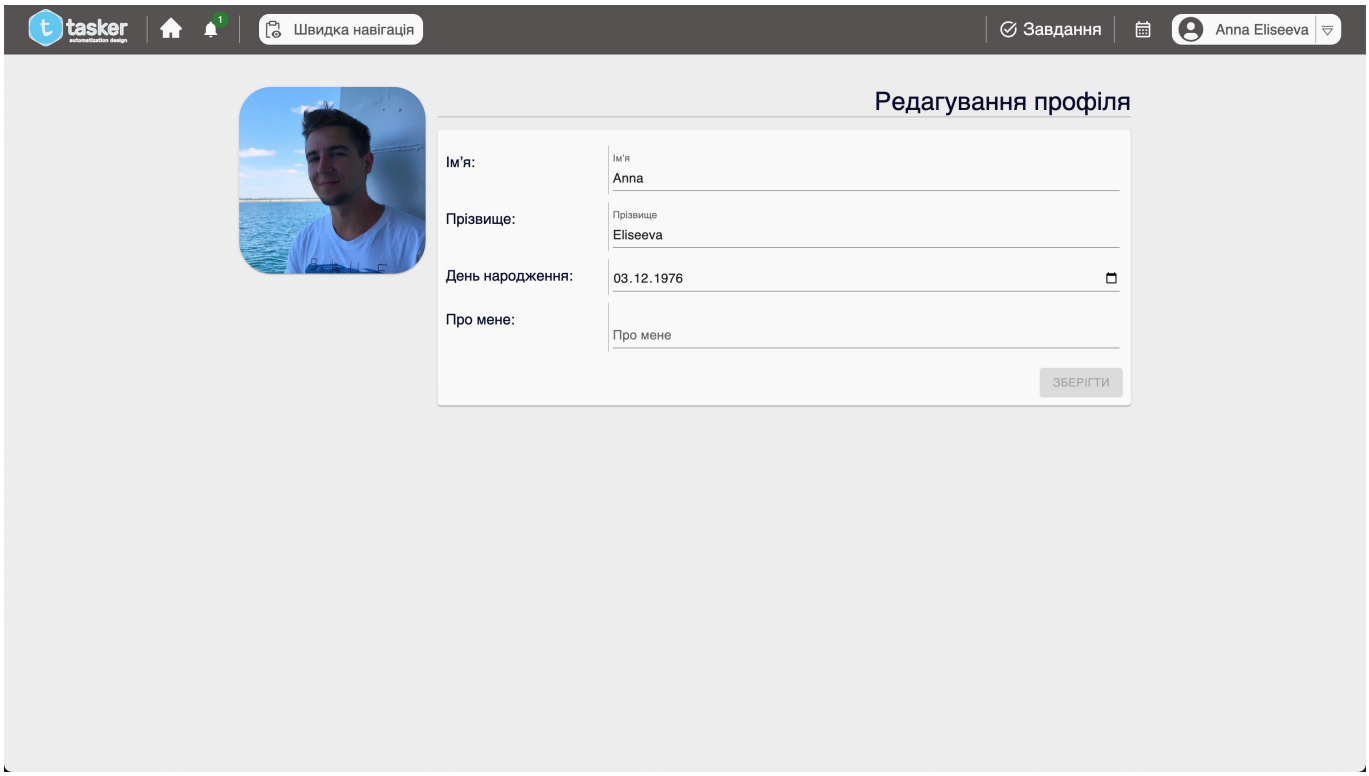


Рисунок 3.38 – Результат встановлення нового фото профілю

3.5 Тестування web-додатку

Для того, щоб впевнитись у дієздатності розробленого web-додатку необхідно виконати функціональне тестування. Його треба провести у декількох напрямках. Це тестування форм створення/редагування на коректність введених даних і посилань на сторінці.

Даний процес починається перевіркою головного меню. Відкриття необхідної сторінки за посиланням відповідає очікуваному результату. Кнопка «Home» перенаправляє проектного менеджера у правильне місце. А саме на головну сторінку з задачами. Перевірено гіперпосилання на профіль співробітника в модальному вікні завдання. Вони функціонують правильно. Модальні вікна завдань містять повну інформацію про них та відображають коректні дані. Тестування форм web-додатку було проведено використовуючи техніку Black Box

[21], в якій перевірка відбувається без знання коду та тонкощів реалізації. Результати проведеного тестування надано у таблиці 3.1.

Таблиця 3.1 – Тестування форм, які знаходяться на клієнтській частині реалізованого web-додатку

№	Назва	Очікуваний результат	Фактичний результат	0/1
1	Перевірка форми створення завдання заповненням полів з коректними даними	Нове завдання зображено на сторінці із завданнями.	Створене завдання відображається на сторінці завдань.	1
2	Перевірка форми створення завдання заповненням полів з некоректними даними	Нове завдання не створене через не активну кнопку «Save», поля з некоректними даними підсвічені червоним кольором та містять повідомлення про неправильно введені дані		1
3	Перевірка форми створення повідомлень з коректними даними	Повідомлення надіслане одержувачу і відображається у боковому меню повідомлень	Створене повідомлення відображається у блоці повідомлень одержувача.	1
4	Перевірка форми створення повідомлень некоректними даними	Повідомлення не надіслане через неактивну кнопку “Send”, поле з некоректними даними підсвічено червоним кольором та містить текст повідомлення про неправильно введені дані		1
5	Перевірка поля пошуку валідними даними	Результат пошуку відображений у блоці пошуку	Результат пошуку, яке містить ключове слово, відображається у блоці пошуку	1

Продовження таблиці 3.1

6	Перевірка поля пошуку невалідними даними	Результат запиту не відображається у блоці пошуку, відображається посилання на всі завдання	Результат пошуку не відображається у блоці з пошуком, першим елементом відображається посилання на сторінку з завданнями	1

Проаналізувавши дані з вищезазначеної таблиці, можна зробити висновок, що даний web-додаток не має дефектів з високою Severity та високим Priority.

ВИСНОВОК

У результаті виконання кваліфікаційної роботи бакалавра було створено web-додаток підтримки діяльності проектного менеджера у КЗАПР.

Під час розробки дипломного проекту було проведено дослідження предметної області використання web-додатку підтримки діяльності проектного менеджера у КЗАПР, проаналізовано сучасні аналоги даного продукту та виконано огляд публікацій. У результаті визначено мету та задачі на розробку проекту «Web-додаток підтримки діяльності проектного менеджера у КЗАПР». Також обрано засоби для реалізації даного програмного продукту. Також було проведено планування робіт та дослідження ризиків, які можуть виникати під час розробки програмного продукту, та реагування на них (Додаток Б).

Проектна частина дипломної роботи містить структурно-функціональне моделювання та діаграму використання web-додатку. Під час розробки бази даних було виділено сутності додатку, їх атрибути та дані, які зберігатимуться.

Практична частина дипломної роботи містить схему архітектури web-додатку, опис розробленого його дизайну та програмну реалізацію. Продемонстровано роботу даного програмного продукту. Виконано успішне тестування web-додатку.

Використання даної розробки дозволить співробітникам машинобудівної галузі, які відповідальні за управління персоналом, автоматизувати механізм розподілення завдань та сповіщення членів команди проекту про нові завдання, та зробить робочий процес більш швидким та ефективним.

Результати роботи були апробовані на науково-практичній конференції ІМА-2022 у Сумському державному університеті (Додаток В).

Лістинг основних модулів розробленого web-додатку представлено у Додатку Г.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Web application to support the activities of the project manager in KZAPR // Матеріали та програма МІЖНАРОДНОЇ НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ студентів та молодих учених / – Суми: МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ МІЖНАРОДНИЙ УНІВЕРСИТЕТ «АСТАНА», 2022. – С. 83.

2. AutoCAD [Електронний ресурс] – Режим доступу до ресурсу: https://www.autodesk.com/products/autocad/overview?term=1-YEAR&tab=subscription.

3. SolidWorks [Електронний ресурс] – Режим доступу до ресурсу: https://www.solidworks.com/ru.

4. Автоматизація управління інноваційними проектами [Електронний ресурс] – Режим доступу до ресурсу: https://dspace.nau.edu.ua/bitstream/NAU/33606/1 Ніколаєва%20тези.pdf.

5. Сучасні інформаційні технології в системі управління проектами [Електронний ресурс] – Режим доступу до ресурсу: http://www.kntu.kr.ua/doc/science/zahody/vikl/2021/11-tez.pdf#page=26.

6. GantPRO [Електронний ресурс] – Режим доступу до ресурсу: https://app.ganttpro.com.

7. Trello [Електронний ресурс] – Режим доступу до ресурсу: https://trello.com.

8. BaseCamp [Електронний ресурс] – Режим доступу до ресурсу: https://basecamp.com.

9. JavaScript [Електронний ресурс] – Режим доступу до ресурсу: https://www.javascript.com.

10. React.js [Електронний ресурс] – Режим доступу до ресурсу: https://reactjs.org.

11. MUI [Електронний ресурс] – Режим доступу до ресурсу: https://mui.com.

12. CSS [Електронний ресурс] – Режим доступу до ресурсу: https://www.w3.org/Style/CSS/Overview.en.html.

13. Sass [Електронний ресурс] – Режим доступу до ресурсу: https://sass-lang.com.

14. PostgreSQL: The World`s Most Advanced Open Source Relational Database [Електронний ресурс] – Режим доступу до ресурсу: https://www.postgresql.org.

15. IDEF0 [Електронний ресурс] – Режим доступу до ресурсу: https://stud.com.ua/87184/ekonomika/metodologiya_idef0.

16. Use-case діаграма [Електронний ресурс] – Режим доступу до ресурсу: http://hi-news.pp.ua/kompyuteri/8924-use-case-dagrama-prikladi-vikoristannya.html.

17. Що таке СУБД? [Електронний ресурс] – Режим доступу до ресурсу: https://uk.education-wiki.com/6943337-what-is-dbms.

18. Клієнт-серверна архітектура [Електронний ресурс] – Режим доступу до ресурсу: https://training.qatestlab.com/blog/technical-articles/client-server-architecture.

19. What is an IDE? [Електронний ресурс] – Режим доступу до ресурсу: https://www.codecademy.com/article/what-is-an-ide.

20. IDE WebStorm [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/webstorm.

21. Тестування методом чорного ящика [Електронний ресурс] – Режим доступу до ресурсу: https://qalight.ua/baza-znaniy/white-black-grey-box-testuvannya.

22. Що таке діаграма Ганта та як вона використовується у бізнес-плані [Електронний ресурс] – Режим доступу до ресурсу: https://monetary-flow.com/shto-take-dagrama-ganta-ta-yak-vona-vikoristovutysya-u-bznes-planuvann/.

23. Поняття ER-моделі [Електронний ресурс] – Режим доступу до ресурсу: https://www.bestprog.net/uk/2019/01/24/the-concept-of-er-model-the-concept-of-essence-and-communication-attributes-attribute-types-ua/.

24. Modern Web-development using reactjs [Електронний ресурс] – Режим доступу до ресурсу: http://ijrra.net/Vol5issue1/IJRRA-05-01-27.pdf.

25. Основні команди Git [Електронний ресурс] – Режим доступу до ресурсу: http://webit.in.ua/article/Osnovni-komandi-Git/.

26. Сучасні тенденції розвитку системи управління проектами [Електронний ресурс] – Режим доступу до ресурсу: http://umo.edu.ua/images/content/institutes/imp/vydannya/konferenc/ТЕЗИ_збірник_30_03_2016_1.pdf#page=91.

27. Методологічні підходи до управління ризиками [Електронний ресурс] – Режим доступу до ресурсу: http://projects.dune-hd.com/bitstream/handle/2010/36469/sts_0521.pdf?sequence=3&isAllowed=y#page=134.

28. Системи автоматизації управління робочим часом [Електронний ресурс] – Режим доступу до ресурсу: http://projects.dune-hd.com/bitstream/handle/2010/36469/sts_0521.pdf?sequence=3&isAllowed=y#page=134.

29. Посібник: знайомство з React [Електронний ресурс] – Режим доступу до ресурсу: https://uk.reactjs.org/tutorial/tutorial.html.

30. UML для бізнес-моделювання: для чого потрібні діаграми процесів [Електронний ресурс] – Режим доступу до ресурсу: https://evergreens.com.ua/ua/articles/uml-diagrams.html.

ДОДАТОК А

ТЕХНІЧНЕ ЗАВДАННЯ
на розробку
«Web-додаток підтримки діяльності проектного менеджера у КЗАПР»

ПОГОДЖЕНО:

Доцент кафедри комп’ютерних наук

_____ Антипенко В.П.

Студент групи ІТ-82-0

_____ Райко Д.І..

Суми 2022

1 Призначення й мета web-додатку підтримки діяльності проектного менеджера у КЗАПР

1.1 Призначення web-додатку

Web-додаток призначений для підтримки діяльності проектного менеджера у КЗАПР.

1.2 Мета створення web-додатку

Головна мета проекту – це розробка web-додатку підтримки діяльності проектного менеджера у КЗАПР та автоматизація таких процесів управління командою проекту, як назначення завдань на виконання, їх відповідальних і контроль над реалізацією проектів.

1.3 Цільова аудиторія

Цільовою аудиторією даного проекту є проектна організація та замовник, які зацікавлені в автоматизації, прискоренні й покращенні якості виконання завдань та відслідковування етапів робіт у проекті.

2 Вимоги до проекту

2.1 Вимога до проекту в цілому

2.1.1 Вимога до структури й функціонування

Web-додаток для підтримки діяльності проектного менеджера у КЗАПР повинен бути реалізований стабільними web-інструментами, які нададуть повноту та повноцінність в використанні функціоналу системи.

У результаті реалізації має бути представлено web-додаток із лаконічним і зрозумілим інтерфейсом та з повноцінним функціоналом.

2.1.2 Вимоги до персоналу

Користувачі повинні бути ознайомлені з базовим використанням комп'ютерних технологій та web-браузером.

2.1.3 Вимоги до збереження інформації

Уся інформація надана у web-додатку повинна зберігатися у базі даних реалізованій засобами системи управління базами даних PostgreSQL.

2.1.4 Вимоги до розмежування доступу

Web-додаток розробляється для внутрішнього використання в проектних організаціях за допомогою Інтернет технологій.

Проектний менеджер повинен слідкувати над етапами виконання завдань та правильний розподіл часу та працівників по завданням.

2.2 Структура web-додатку

2.2.1 Загальна інформація про структуру web-додатку

При відкритті web-додатку за посиланням головної сторінки користувач може переглянути інформацію про завдання, в яких він залучений. Натиснувши на певний проект співробітник отримає список задач, які він виконує або делегує. Натиснувши на завдання людині стануть доступними докладна інформація про нього та пов'язані з ним активності. Користувач може натиснути на пункт «Office», який призначений для відображення ієрархії задач у проекті та додати нові. Після того, як PM назначить певному співробітнику завдання, буде отримане сповіщення, яке користувач може переглянути в пункті «Сповіщення». У лівій частині можна переглянути його деталі, натиснувши на нього.

Сповіщення надає можливість завантажити документ або посилання на виконане завдання, або ж просто надати відповідь. Адміністративна панель дозволяє редагувати, видаляти або створювати дані.

2.2.2 Навігаційне меню

Для зручної навігації між сторінками повинно бути створене головне меню з можливістю переходів на головні та суміжні сторінки web-додатку. Воно має бути розміщено у верхній частині десктопної версії сторінки, або ж меню «burger» на мобільній версії.

2.2.3 Управління контентом

Управління контентом в деякій мірі здійснюється кожним користувачем, у залежності від привілеїв. Уся інформація наповнення повинна зберігатися у базі даних та на сервері.

2.2.4 Дизайн web-додатку

Дизайн web-додатку має бути виконаний у мінімалістичному та сучасному стилі. За основні кольори було обрано контрастні кольори, а саме білий, сірий та блакитний. Тому під час розробки даного програмного продукту доцільно застосовувати саме ці кольори.

Увесь web-додаток повинен бути виконаний в одному стилі. Текст повинен бути комфортним для читання. Шаблон майбутньої розробки зображено на рисунку А.1.

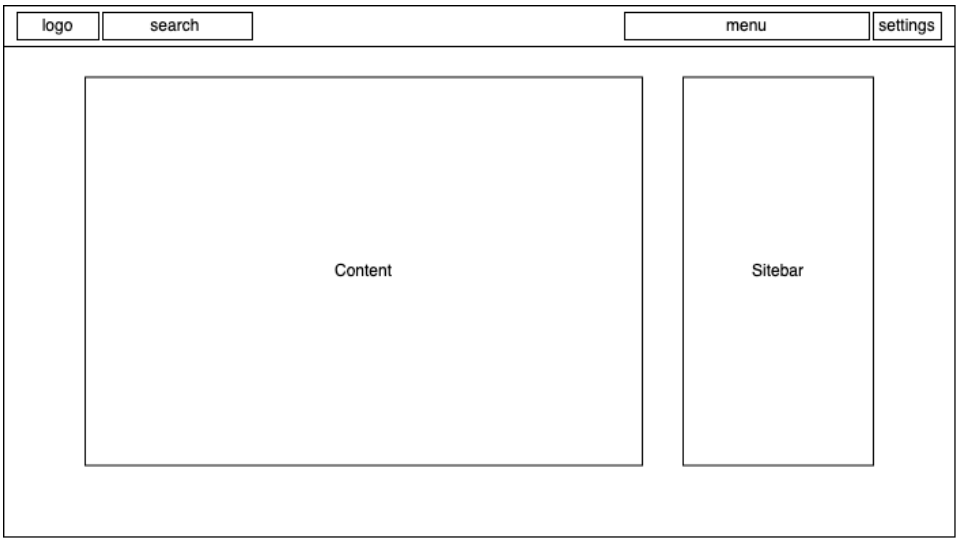


Рисунок А.1 – Схема сторінки проектного менеджера

2.2.5 Система навігації (карта web-додатку)

Карта web-додатку зображена на рисунку А.2.

Рисунок А.2 – Система навігації

2.3 Вимоги до видів забезпечення

2.3.1 Вимоги до лінгвістичного забезпечення

Текстові елементи web-додатку мають бути виконані англійською мовою, але з максимальними можливостями мультимовності.

2.3.2 Вимоги до програмного забезпечення

Для підтримки правильного відображення функціоналу та для стабільної роботи має бути Internet Explorer 11 і вище, або Firefox 5 і вище, або Opera 10.32 і вище, або Safari 10 і вище, або Chrome 2 і вище, або ж Microsoft Edge.

2.4 Вимоги до функціонування web-додатку

2.4.1 Потреби користувача

Потреби користувача, визначені на основі рішення замовника, представлені у таблиці А.1.

Таблиця А.1 – Потреби користувача

ID	Потреби користувача	Джерело
UN-01	Створення, видалення компанії	Адміністратор
UN-02	Редагування інформації про компанію	Керівник компанії, адміністратор
UN-03	Створення проектів в компанії	Керівник компанії
UN-04	Додавання, редагування, видалення завдань	Керівник компанії, проектний менеджер
UN-05	Можливість оновлення статусу завдання	Керівник компанії, проектний менеджер, призначений виконавець
UN-06	Можливість створення особистих завдань	Керівник компанії, проектний менеджер, призначений виконавець
UN-07	Створення запланованих повідомлень для осіб, які прив’язані до завдання	Керівник компанії, проектний менеджер
UN-08	Створення блокувальних повідомлень для завдань	Керівник компанії, проектний менеджер
UN-09	Видалення застарілої інформації	Адміністратор
UN-10	Відповіді на звертання до підтримки web-додатку	Адміністратор
UN-11	Можливість звернутися до підтримки web-додатку	Усі користувачі

2.4.2 Системні вимоги

Проаналізувавши потреби користувачів, було визначено наступні вимоги:

- наявність реєстрації та авторизації користувачів;
- створення, редагування та видалення завдань, проектів та компаній;
- створення відтермінованих та блокуючих повідомлень;
- створення підтримку web-додатку.

3 Склад і зміст робіт зі створення web-додатку підтримки діяльності проектного менеджера у КЗАПР

Детальний опис етапів створення web-додатку наведено в таблиці А.2.

Таблиця А.2 – Етапи створення web-додатку

№	Склад і зміст робіт	Строк розробки
1	Розробка шаблону web-додатку	1 день
2	Створення верстки сторінок web-додатку	14 днів
3	Розробка модуля завдань	10 днів
4	Розробка модуля повідомлень	10 днів
5	Розробка модуля Office	10 днів
6	Розробка бази даних	9 днів
7	Створення записів в базі даних	3 дні
8	Beta-тестування	7 днів
9	Alpha-тестування	5 днів
10	Розміщення на хостингу	1 день
11	Перевірка працездатності	1 день
12	Написання супровідної документації	2 дні
13	Реліз web-додатку	2 дні
	Загальна тривалість робіт	75 днів

4 Вимоги до складу й змісту робіт із введення web-додатку в експлуатацію

Web-додаток має бути затверджено та розміщено на хостингу.

ДОДАТОК Б

Планування робіт

У зв’язку з пандемією, бойовими діями, розвитком міжнародних компаній тощо за останні роки збільшився попит на застосування віддаленої роботи. Раніше для зв’язку та контролю над виконанням проекту в організаціях використовували соціальні мережі. Люди планували особистий розпорядок дня, проте виникла необхідність планувати процеси всередині компаній. І організації машинобудівного профілю не є виключенням.

Тому вдосконалення процесів автоматизації проектувальних робіт позитивно впливатиме не тільки на продуктивність роботи організації. Зокрема, якщо поліпшити механізм управління проектами та командою, це дозволить покращити працездатність конкретного співробітника.

Деталізація мети проекту методом SMART. Для того, щоб проект був конкурентоспроможним та успішним треба правильно визначити його мету за допомогою SMART-методу. Результат деталізації цим методом можна розглянути в таблиці Б.1.

Таблиця Б.1 – Деталізація мети проекту методом SMART

Specific (конкретна)	Створити web-додаток підтримки діяльності проектного менеджера у комплексі засобів автоматизованого проектування робіт.
Measurable (вимірювана)	Розробити web-додаток до кінця 4 курсу, використовуючи мінімум ресурсів.
Achievable (досяжна, узгоджена)	Для розробки проекту потрібні знання HTML, CSS, мови програмування JavaScript, PHP, MySQL та навички написання документації.

Продовження табл. Б.1

Relevant (реалістична)	Створений web-додаток буде автоматизувати процес розробки проектів, полегшуючи механізм управління командою їх виконавців.
Time-framed (обмежена в часі)	Термін досягнення мети проекту визначено за навчальним планом до червня 2022 року.

Планування змісту робіт. Структура розбивки робіт (WBS) – це ключовий результат проекту, який розбиває роботу манди на керовані частини. Якщо коротко, це візуальний план елементів проекту, що згруповані ієрархією.

Гарний спосіб впевнитися, чи дотримується правило 100%, – це перевірити результат кожного з батьків, для того щоб переконатися, що обсяг кроків, які мають виконати його діти, еквівалентний 100% роботи батьків.

Найвищим рівнем повинен бути розміщений проектний продукт. На наступному етапі розташовані набори дій та заходів, що спрощують декомпозицію до тих пір, поки вони не стануть елементарними. У свою чергу елементарні – це дії, які мають чіткий, визначений результат та яку виконує одна конкретна особа. Через це можна легко обчислити витрати на виконання, як грошові, так і фінансові. WBS з розробки web-додатку підтримки проектного менеджера в КЗАПР представлено на рисунку Б.1.

Планування структури виконавців. Після декомпозиції етапів виконання процесів наступає час розподілення обов’язків серед виконавців. Структура розбивки (OBS) зазвичай поєднує пакет завдань на найнижчому рівні з організацією підрозділу. Для кращої деталізації визначення OBS використовується матриця розподілу відповідальності.

У ролі назначених осіб виступають співробітники, які кваліфікуються на вузькопрофільній елементарній роботі, яка визначена у WBS.

На рисунку Б.2 можна переглянути організаційну структуру планування проекту. Усі виконавці та їх ролі описані в таблиці Б.2

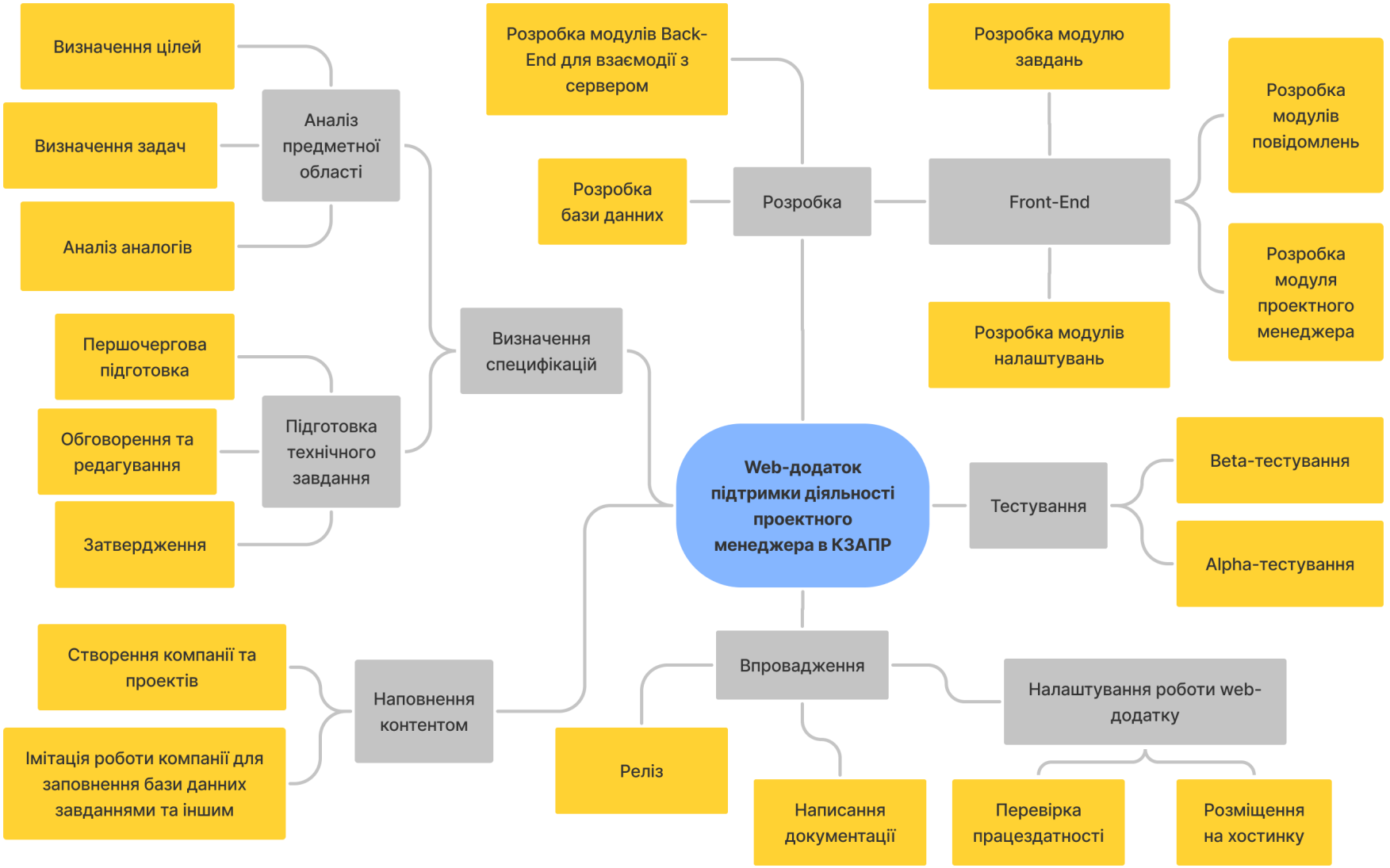


Рисунок Б.1 – WBS-структура робіт проекту

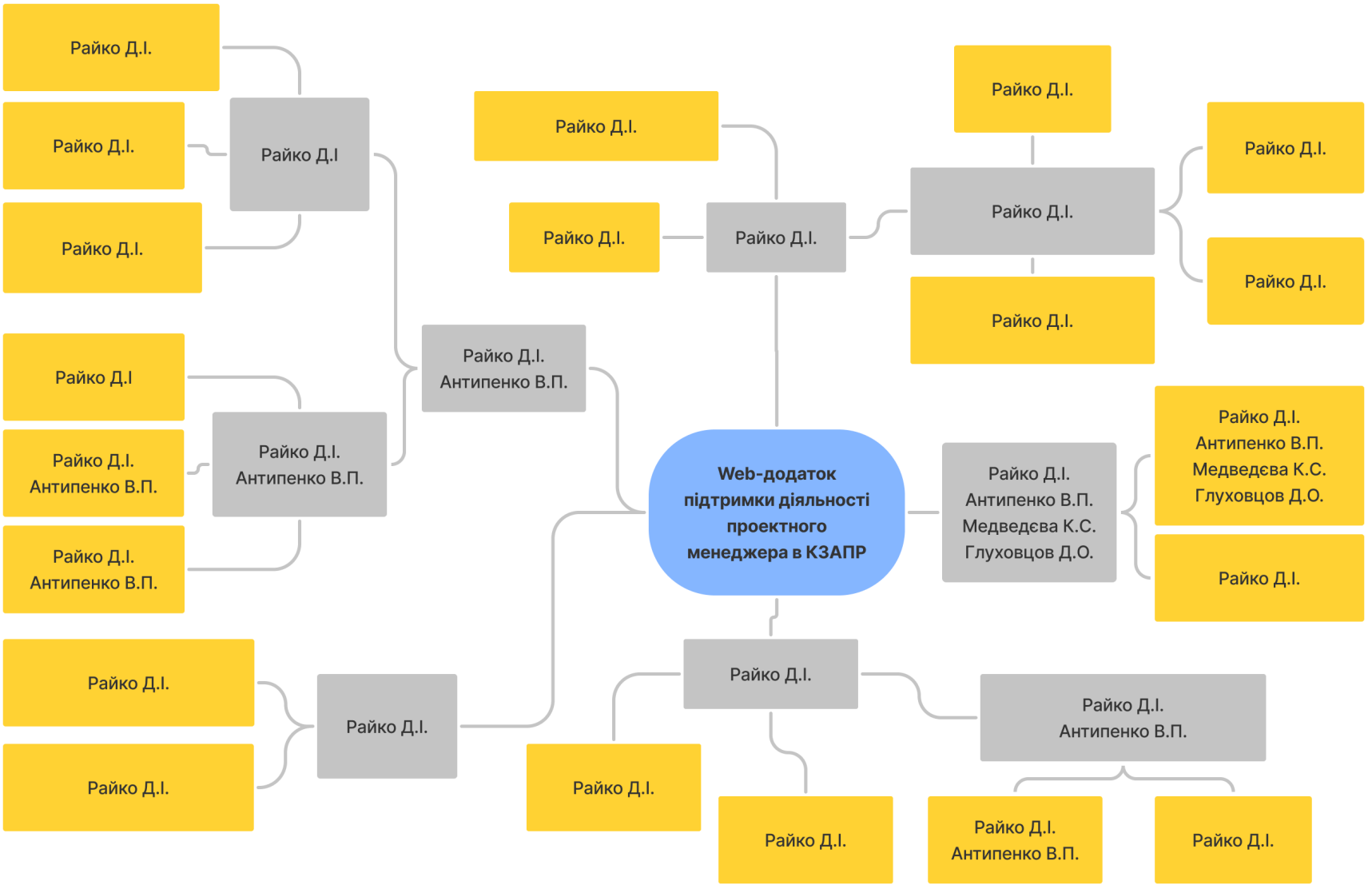


Рисунок Б.2 – OBS-структура робіт проекту

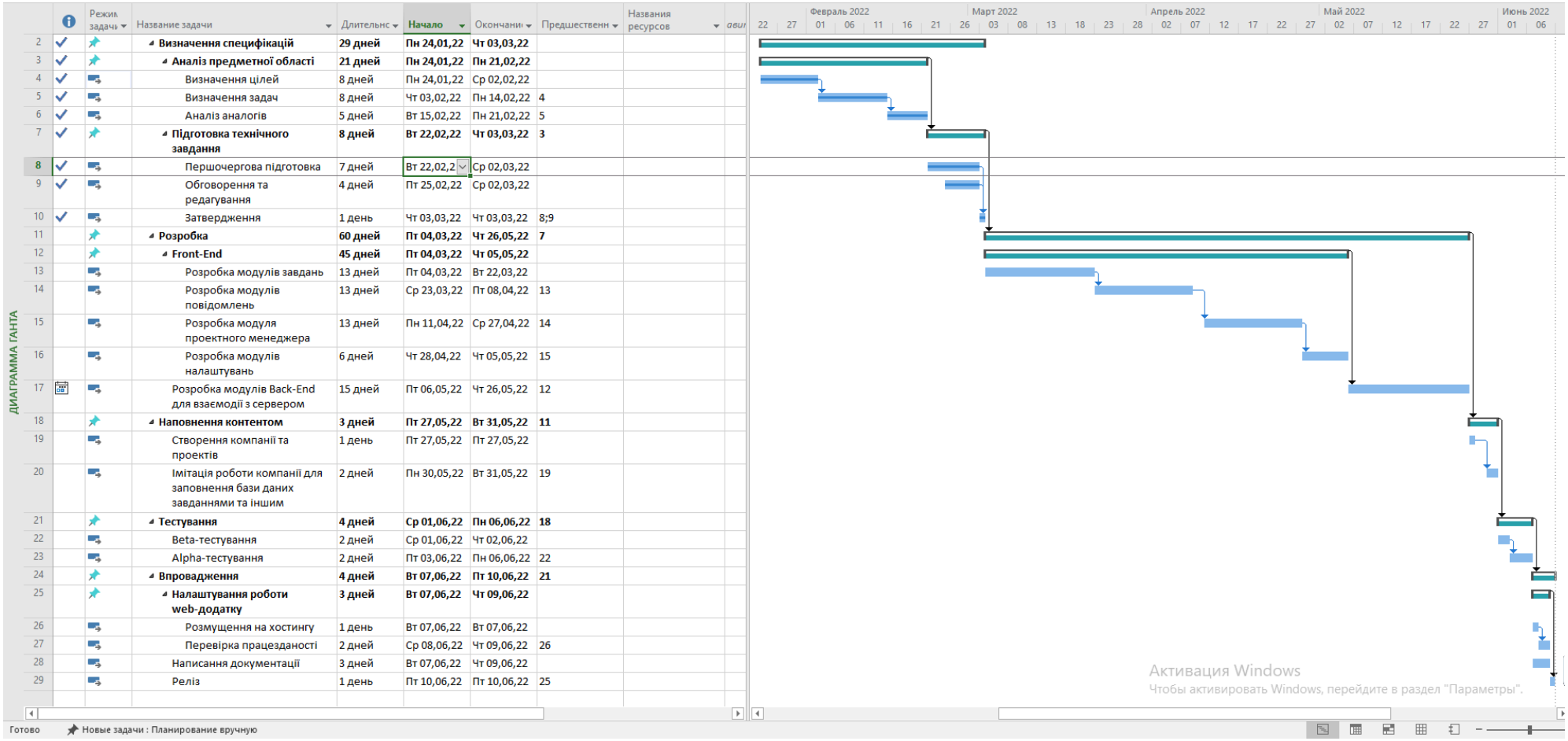
Таблиця Б.2 – Виконавці проекту

Роль	**Ім’я**	**Проектна роль**
Розробник	Райко Д.І.	Виконує Front-end та Back-end розробки.
Проектувальник	Райко Д.І.	Відповідає за проектування бази даних та структуру програмного забезпечення.
Тестувальник	Медведєва К.С., Глуховцов Д.О.	Проводе тестування дизайну та функціональної частини web-додатку.
Керівник проекту	Антипенко В.П.	Видає завдання на розробку проекту.
Менеджер проекту	Райко Д.І.	Несе відповідальність за розподіл ресурсів, завдань та своєчасне виконання

Діаграма Ганта. Проектування календарного графіку є одним з найважливіших етапів життя проекту. Він представляє собою розклад виконання завдань з прив’язкою до часової шкали. З допомогою чого можна правильно розрахувати та графічно показати повну картину проекту. Оцінивши повні масштаби роботи, достовірно оцінюються надані ресурси з урахуванням вихідних, свят та форс-мажорних ситуацій.

У свою чергу контроль за змістом, термінами та послідовністю виконання робіт стає реальним за допомогою координуючого календарного плану.

Діаграму Ганта проекту представлено на рисунку Б.3.



	Режим задачи	Название задачи	Длительнс	Начало	Окончани	Предшественн	Названия ресурсов
2		**Визначення специфікацій**	**29 дней**	**Пн 24,01,22**	**Чт 03,03,22**		
3		**Аналіз предметної області**	**21 дней**	**Пн 24,01,22**	**Пн 21,02,22**		
4		Визначення цілей	8 дней	Пн 24,01,22	Ср 02,02,22		
5		Визначення задач	8 дней	Чт 03,02,22	Пн 14,02,22	4	
6		Аналіз аналогів	5 дней	Вт 15,02,22	Пн 21,02,22	5	
7		**Підготовка технічного завдання**	**8 дней**	**Вт 22,02,22**	**Чт 03,03,22**	**3**	
8		Першочергова підготовка	7 дней	Вт 22,02,2	Ср 02,03,22		
9		Обговорення та редагування	4 дней	Пт 25,02,22	Ср 02,03,22		
10		Затвердження	1 день	Чт 03,03,22	Чт 03,03,22	8;9	
11		**Розробка**	**60 дней**	**Пт 04,03,22**	**Чт 26,05,22**	**7**	
12		**Front-End**	**45 дней**	**Пт 04,03,22**	**Чт 05,05,22**		
13		Розробка модулів завдань	13 дней	Пт 04,03,22	Вт 22,03,22		
14		Розробка модулів повідомлень	13 дней	Ср 23,03,22	Пт 08,04,22	13	
15		Розробка модуля проектного менеджера	13 дней	Пн 11,04,22	Ср 27,04,22	14	
16		Розробка модулів налаштувань	6 дней	Чт 28,04,22	Чт 05,05,22	15	
17		Розробка модулів Back-End для взаємодії з сервером	15 дней	Пт 06,05,22	Чт 26,05,22	12	
18		**Наповнення контентом**	**3 дней**	**Пт 27,05,22**	**Вт 31,05,22**	**11**	
19		Створення компанії та проектів	1 день	Пт 27,05,22	Пт 27,05,22		
20		Імітація роботи компанії для заповнення бази даних завданнями та іншим	2 дней	Пн 30,05,22	Вт 31,05,22	19	
21		**Тестування**	**4 дней**	**Ср 01,06,22**	**Пн 06,06,22**	**18**	
22		Beta-тестування	2 дней	Ср 01,06,22	Чт 02,06,22		
23		Alpha-тестування	2 дней	Пт 03,06,22	Пн 06,06,22	22	
24		**Впровадження**	**4 дней**	**Вт 07,06,22**	**Пт 10,06,22**	**21**	
25		**Налаштування роботи web-додатку**	**3 дней**	**Вт 07,06,22**	**Чт 09,06,22**		
26		Розмущення на хостингу	1 день	Вт 07,06,22	Вт 07,06,22		
27		Перевірка працезданості	2 дней	Ср 08,06,22	Чт 09,06,22	26	
28		Написання документації	3 дней	Вт 07,06,22	Чт 09,06,22		
29		Реліз	1 день	Пт 10,06,22	Пт 10,06,22	25	

Рисунок Б.3 – Діаграма Ганта

Управління ризиками проекту. Під час аналізу та оцінки ризиків потрібно визначити такі, які мають найвищий пріоритет виправлення. Відповідно чим вище ризик, тим швидше реагування на нього. Наступним кроком проектування є виконання кількісного оцінювання ризиків. У таблиці Б.3 представлено шкалу класифікацій за величиною впливу на проект та шанс виникнення.

Таблиця Б.3 – Шкала оцінювання ризиків за ймовірністю виникнення та величиною впливу.

Оцінка	**Ймовірність виникнення**	**Вплив ризику**	**Тип ризику**
1	Низька	Низький	Прийнятні
2	Середня	Середній	Виправдані
3	Висока	Високий	Недопустимі

Для зменшення впливу ризиків до мінімального впливу на проект потрібно розробити план реагування на них. Для цього потрібно визначити оцінку наслідків та ефективність розробки на проект. Оцінювання відбувається за таблицею Б.3, після чого отримаємо ймовірності виникнення ризиків, що зображено на рисунку Б.4.

Кожний колір на схемі зображає ризики, відповідно зелений – прийнятні, помаранчеві – виправдані та червоні – недопустимі.

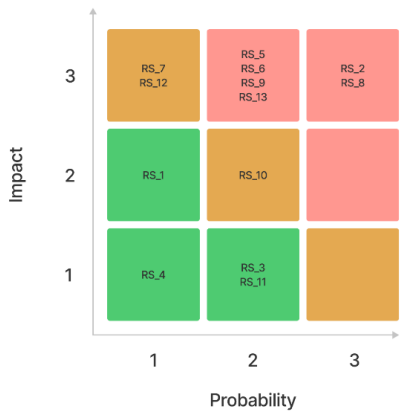


Рисунок Б.4 – Матриця ймовірності

Отриманий індекс для кожного ризику відповідає класифікації, які представлені у таблиці Б.4. Наступним етапом потрібно розробити стратегії реагування на кожну загрозу, вони описані у таблиці Б.5.

Таблиця Б.4 – Шкала оцінювання за рівнем ризику

№	**Назва**	**Межі**	**Ризики, які входять (номера)**
1	Прийнятні	$1 \leq R \leq 2$	4, 3, 11
2	Виправдані	$3 \leq R \leq 4$	1, 10
3	Недопустимі	$6 \leq R \leq 9$	7, 12, 5, 6, 9, 13, 2, 8

Таблиця Б.5 – Ризики та стратегії реагування

№	**Статус ризику**	**Опис ризику**	**Ймовірність виникнення**	**Вплив ризику**	**Ранг ризику**	**План А**	**Тип стратегії реагування**	**План Б**
RS_1	Відкритий	Проблеми з роботою ПЗ у розробника	Низька	Середній	4	1. Мати резервні програмні засоби. 2. Найняти спеціаліста, що спеціалізується в усуненні даного роду проблем	Попередження	Замінити ПЗ.
RS_2	Відкритий	Допущення помилок під час проектування проекту	Високий	Високий	9	1. Своєчасна перевірка даних. 2. Задіяти додаткову особу для перевірки етапу проектування.	Пом’якшення	Проводити постійний контроль та перевірки етапів проектування.

Продовження табл. Б.5

№	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_3	Відкритий	Відключення світла в процесі розробки проекту	Середня	Низький	2	1. Мати резервний генератор світла, або ж користуватися технікою, яка має акумуляторну батарею. 2. Налаштувати сповіщення у разі планового відключення світла.	Попередження	Використовувати попередні повідомлення про відключення світла або ж техніку з акумулюючими батареями.
RS_4	Відкритий	Розробка непотрібного функціоналу	Низька	Низький	1	1. Надати замовнику інформацію про додатковий функціонал. 2. Додати опис функціоналу до ТЗ.	Використання	Проаналізувати наслідки та недоліки від можливих змін. Занести зміни до ТЗ.

Продовження табл. Б.5

№	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_5	Відкритий	Відсутність резервних копій даних	Середня	Високий	8	1. Використовувати ПЗ яке має включає функцію автоматичного збереження. 2. Використовувати хмарні сховища для зберігання копій.	Попередження	Робити копії даних. Використовувати ПЗ з автоматичним збереженням.
RS_6	Відкритий	Зміна ТЗ під час розробки проекту	Середня	Високий	8	1. Своєчасно виділити вимоги до проекту. 2. Обговорити з замовником та виконувачем всі технічні питання і умови реалізації.	Пом’якшення	Переоцінка та перепланування проекту після кожної зміни.

Продовження табл. Б.5

№	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_7	Відкритий	Затримка релізу продукту	Низька	Високий	7	1. Під час проектування правильно розподілити час виконання етапів проекту. 2. Попередити замовника проекту про затримку.	Прийняття	
RS_8	Відкритий	Не правильний розподіл часу	Висока	Високий	9	1. Провести аналіз та правильно проаналізувати усі ризики затримок. 2. Внести зміни до системи планування.	Пом’якшення	Перепланувати етапи виконання роботи. Змінити пріоритети робіт. Знайти можливість під налаштування проекту під нові часові межі.

Продовження табл. Б.5

№	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_9	Відкритий	Не правильний розподіл бюджету	Середня	Високий	8	1. Зв’язатися з замовником проекту та уточнити фінансування проекту. 2. Змінити план фінансування.	Пом’якшення	Проінформувати замовника про помилки в розрахунку та запропонувати змінити бюджет проекту.
RS_10	Відкритий	Недостатня база знань у розробників	Середня	Середній	5	1. Підвищити базу знать виконавця. 2. Пройти деякі онлайн курси.	Пом’якшення	Перепланувати етапи виконання робіт так, щоб можна було враховувати час на додаткову підготовку спеціалістів.

Продовження табл. Б.5

№	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_11	Відкритий	Затримки у розробці дизайну продукту	Середня	Низький	2	1. Продовжити виконувати функціонал проекту. 2. Зробити перепланування робіт для оптимального використання часу затримок розробки дизайну.	Використання	Планувати виконання проекту так, щоб розробка функціоналу не залежала від розробки дизайну.
RS_12	Відкритий	Зміна специфікацій замовником на етапі розробки	Низька	Високий	7	1. Зв’язатися з замовником та повністю проговорити усі специфікації ще під час планування. 2. Додати до планування робіт можливість зміни специфікацій під час розробки.	Пом’якшення	Переоцінка та перепланування проекту після кожної зміни.

Продовження табл. Б.5

№	**Статус ризику**	**Опис ризику**	**Ймовірність виникнення**	**Вплив ризику**	**Ранг ризику**	**План А**	**Тип стратегії реагування**	**План Б**
RS_13	Відкритий	Хвороба розробника	Середня	Високий	8	1. Мати запасний варіант для заміни розробника. 2. Пропрацювати можливі хвороби, відпустки на момент планування проекту	Попередження	Зробити умови у яких буде важко захворіти, або ж умови для гнучкого виконання робіт.

ДОДАТОК В
АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

IMA :: 2022 *СЕКЦІЯ 2: Інформаційні технології проєктування*

Web application to support the activities of the project manager in KZAPR

Dmytro Raiko, *Student of IT-82*; Viktoriia Antypenko, *Associate Professor*

Department "Information Technology"
Sumy State University, Sumy, Ukraine

Today world defines information technologies (IT) as a modern engine of the development of any industry. It is actually so considering temps of current progress, especially in the machine-building field.

Use of IT allows to accelerate and automate any development. Recently, the direction of time management has become especially popular. More and more people support the division of manufacturing and personal time into parts to implement separate tasks with specific deadlines. In its turn, this increases the productivity of everyone who adheres to his or her schedule.

Modern organizations also need to use a similar process. They will get an obvious profit from this. It's necessary to automate project development processes. Due to the pandemic and hostilities, most companies are forced to move to remote work. Therefore, now there is a need to create a software product that will help organize and ensure the process of performing tasks by employees who are physically absent at the company's workplace.

It is much easier for the project manager to manage the team, assign tasks, track their performance, and simply maintain communication with employees using a web-oriented software application. It must meet all functional requirements for the automated designing of works. And machine-building enterprises are no exception. Thus, research and development of information technologies to support the project manager's activities through the tools complex of automation the designing works (KZAPR) are topical.

The current task embodies automated mechanisms for assigning and receiving tasks, timely notification of deadlines, control of the activities of the entire organization as well as the accumulation of knowledge about the manufacturing competencies of staff etc.

The result of this work is a web application to support the activities of the project manager in KZAPR, which provides automated forecasting, organization of planning, distribution, and control of designing work at the enterprise of machine-building profile.

ДОДАТОК Г

ЛІСТИНГ ПРОГРАМНОГО КОДУ ОСНОВНИХ МОДУЛІВ WEB-ДОДАТКУ

Посилання на репозиторій GitHub: https://github.com/DmytroRaiko/TaskerKZAPR

Tasks.js

```
import React, { useContext, useState } from 'react';
import PropTypes from 'prop-types';
import { useQuery } from 'react-query';
import Backdrop from '@mui/material/Backdrop';
import CircularProgress from '@mui/material/CircularProgress';
import TasksPage from '../../components/tasks/TasksPage';
import Filter from '../../components/Filter';
import { getTasks } from '../../services/crud/tasks';

const user = JSON.parse(sessionStorage.getItem('token'));

const Tasks = ({
  handleModalTaskOpen,
}) => {
  const [filter, setFilter] = useState({});

  const { isLoading, data } = useQuery(
    ['tasks-main-pages', filter],
    () => getTasks(filter, user.id),
    {
      manual: true,
    },
  );

  return (
    <div className="main-task-body">
      <div className="body-task">
        {!isLoading
          || (
          <Backdrop
            sx={{ color: '#fff', zIndex: (theme) => theme.zIndex.drawer + 1 }}
            open={isLoading}
          >
            <CircularProgress color="inherit" />
          </Backdrop>
          )}
        {data?.data.data && (
        <TasksPage
          tasks={data?.data.data}
          onOpen={handleModalTaskOpen}
        />
        )}
      </div>

      <Filter
        handleModalTaskOpen={handleModalTaskOpen}
        setFilter={setFilter}
      />
```

```
    </div>
  );
};

Tasks.propTypes = {
  handleModalTaskOpen: PropTypes.func.isRequired,
};

export default Tasks;
```

TasksPage.js (Компонент відображення завдань)

```
import React, { useContext } from 'react';
import PropTypes from 'prop-types';
import Avatar from '@mui/material/Avatar';
import Card from '@mui/material/Card';
import { CardActionArea } from '@mui/material';
import { Link as RouterLink } from 'react-router-dom';
import Link from '@mui/material/Link';
import AccessTimeOutlinedIcon from '@mui/icons-material/AccessTimeOutlined';
import stringAvatar from '../../services/icons/avatarIcon';
import { dateFormat } from '../../services/time';
import { LanguageContext } from '../../containers/language/Language';

const TasksPage = ({
  tasks, onOpen,
}) => {
  const { dictionary } = useContext(LanguageContext);

  return (
    tasks?.map((taskItem) => (
      <Card
        key={`tasks-page-task-${taskItem.taskId}`}
        sx={{
          width: '100%',
        }}
      >
        <CardActionArea
          className="card-task view-task"
          onClick={() => onOpen(taskItem.taskId)}
        >
          <Avatar
            className="card-icon"
            {...stringAvatar(taskItem.title, taskItem.color)}
          />
          <div className="card-info">
            <div className="info-title text-22">
              [
              {taskItem.taskId}
              ]
              {' '}
              {taskItem.title}
              <div className="hor-line" />
            </div>
            <Link
              component={RouterLink}
              to={`/project/${taskItem.projectId}`}
              className="project-name text-9"
              underline="hover"
              onClick={(e) => {
                e.stopPropagation();
              }}
```

```
>
  {taskItem.projectName}
</Link>

{taskItem?.assignToProfileId && (
  <div className="employee-task">
    <p
      className="text-11"
      style={{
        marginBottom: '2px',
      }}
    >
      {dictionary.tasksList.assignTo}
    </p>

    <Link
      component={RouterLink}
      to={`/profile/${taskItem?.assignToProfileId}`}
      color="inherit"
      underline="hover"
      className="emp text-10"
      onClick={(e) => {
        e.stopPropagation();
      }}
    >
      {taskItem?.assignToName}
      {' '}
      {taskItem?.assignToSurname}
    </Link>
  </div>
)}

{taskItem?.inspectorProfileId && (
<div className="employee-task">
  <p
    className="text-11"
    style={{
      marginBottom: '2px',
    }}
  >
    {dictionary.tasksList.inspector}
  </p>

  <Link
    component={RouterLink}
    to={`/profile/${taskItem?.inspectorProfileId}`}
    color="inherit"
    underline="hover"
    className="emp text-10"
    onClick={(e) => {
      e.stopPropagation();
    }}
  >
    {taskItem?.inspectorName}
    {' '}
    {taskItem?.inspectorSurname}
  </Link>
</div>
)}

<div className="executor-task">
  <p className="text-11">
    {dictionary.tasksList.createdBy}
```

```
              </p>
              <Link
                component={RouterLink}
                to={`/profile/${taskItem?.createdByProfileId}`}
                color="inherit"
                underline="hover"
                className="ex text-10"
                onClick={(e) => {
                  e.stopPropagation();
                }}
              >
                {taskItem?.createdByName}
                {' '}
                {taskItem?.createdBySurname}
              </Link>
            </div>

            <div className="date">
              <div className="pre-date text text-11">
                {dictionary.tasksList.until}
              </div>
              <div className="date-time-end text-13">
                {' '}
                {dateFormat(taskItem.dateEnd)}
              </div>
              <AccessTimeOutlinedIcon fontSize="middle" />
            </div>
          </div>
        </CardActionArea>
      </Card>
    ))
  );
};

TasksPage.propTypes = {
  tasks: PropTypes.arrayOf(
    PropTypes.shape({}),
  ),
  onOpen: PropTypes.func.isRequired,
};

TasksPage.defaultProps = {
  tasks: [],
};

export default TasksPage;
```

Filter.js

```
import React, { useContext } from 'react';
import { Button } from '@mui/material';
import PropTypes from 'prop-types';
import * as Yup from 'yup';
import { Field, Form, Formik } from 'formik';
import { useQuery, useQueryClient } from 'react-query';
import { TextField } from 'formik-mui';
import { LanguageContext } from '../containers/language/Language';
import { getStatus, getProjects } from '../services/crud/pages';

const user = JSON.parse(sessionStorage.getItem('token'));

const Filter = ({
```

```
  handleModalTaskOpen, setFilter,
}) => {
  const { dictionary } = useContext(LanguageContext);
  const queryClient = useQueryClient();

  const date = new Date();
  const firstDay = new Date(date.getFullYear(), date.getMonth() - 1, 1);
  const lastDay = new Date(date.getFullYear(), date.getMonth() + 2, 0);

  const sQuery = useQuery(
    'main-page-status',
    () => getStatus('task'),
  );
  const status = sQuery?.data?.data.data || [];

  const pQuery = useQuery(
    'main-page-projects',
    () => getProjects(user.id),
  );
  const projects = pQuery?.data?.data.data || [];

  const formSchema = Yup.object().shape({
    status: Yup.array().of(
      Yup.number(),
    ),
    projects: Yup.array().of(
      Yup.number(),
    ),
    dateStart: Yup.string().required('Date is required'),
    dateEnd: Yup.string().required('Date is required'),
    assignToMe: Yup.string().oneOf(['yes', 'no']),
    createdByMe: Yup.string().oneOf(['yes', 'no']),
    inspectorByMe: Yup.string().oneOf(['yes', 'no']),
  });
  const statusValues = status ? Array.from(status, (s) => s.statusId) : [];
  const projectsValues = projects ? Array.from(projects, (s) => s.projectId) : [];

  const formValues = {
    status: statusValues,
    projects: projectsValues,
    dateStart: new Date(firstDay).toISOString().slice(0, 10),
    dateEnd: new Date(lastDay).toISOString().slice(0, 10),
    assignToMe: 'yes',
    createdByMe: 'yes',
    inspectorByMe: 'yes',
  };

  const onApply = (formData, { setSubmitting }) => {
    setFilter({
      status: (formData?.status.length) ? formData.status : statusValues,
      projects: (formData?.projects.length) ? formData.projects : projectsValues,
      dateStart: formData.dateStart,
      dateEnd: formData.dateEnd,
      assignToMe: formData.assignToMe,
      createdByMe: formData.createdByMe,
      inspectorByMe: formData.inspectorByMe,
    });
    queryClient.invalidateQueries('tasks-main-pages');
    setSubmitting(false);
  };
  return (
    <div className="body-filter">
      {(user.roleName === 'pgd' || user.roleName === 'inspector'
```

```
    || user.roleName === 'employee' || user.roleName === 'pm')
  && (
  <Button
    onClick={() => handleModalTaskOpen(null, 1)}
    variant="contained"
    fullWidth
  >
    {dictionary.addTask}
  </Button>
  )}

<div className="filter">
  <Formik
    onSubmit={onApply}
    validationSchema={formSchema}
    initialValues={formValues}
  >
    {({
      setFieldValue, values,
    }) => (
      <Form>
        <Button
          type="submit"
          fullWidth
          variant="outlined"
        >
          Apply
        </Button>

        <div className="checker">
          <div className="element">
            <div className="text" style={{ marginRight: '10px' }}>
              {dictionary.task.from}
              {': '}
            </div>
            <Field
              component={TextField}
              type="date"
              name="dateStart"
              variant="standard"
              fullWidth
              margin="dense"
            />
          </div>

          <div className="element">
            <div className="text" style={{ marginRight: '10px' }}>
              {dictionary.task.till}
              {': '}
            </div>
            <Field
              component={TextField}
              type="date"
              name="dateEnd"
              variant="standard"
              fullWidth
              margin="dense"
            />
          </div>
        </div>

        <div className="checker">
          <label
```

```
                htmlFor="assignToMe"
                className="element"
              >
                <input
                  type="checkbox"
                  className="pop"
                  name="assignToMe"
                  id="assignToMe"
                  checked={values.assignToMe === 'yes'}
                  onChange={() => {
                    setFieldValue('assignToMe', values.assignToMe === 'yes' ? 'no'
: 'yes');
                  }}
                />
                <p className="pop-text">Assign to me</p>
              </label>

              <label
                htmlFor="createdByMe"
                className="element"
              >
                <input
                  type="checkbox"
                  className="pop"
                  name="createdByMe"
                  id="createdByMe"
                  checked={values.createdByMe === 'yes'}
                  onChange={() => {
                    setFieldValue('createdByMe', values.createdByMe === 'yes' ? 'no'
: 'yes');
                  }}
                />
                <p className="pop-text">Created by me</p>
              </label>

              <label
                htmlFor="inspectorByMe"
                className="element"
              >
                <input
                  type="checkbox"
                  className="pop"
                  name="inspectorByMe"
                  id="inspectorByMe"
                  checked={values.inspectorByMe === 'yes'}
                  onChange={() => {
                    setFieldValue('inspectorByMe', values.inspectorByMe === 'yes' ?
'no' : 'yes');
                  }}
                />
                <p className="pop-text">Inspected by me</p>
              </label>
            </div>

            <div className="checker">
              <div className="text text-18">Status</div>
              {status?.map((s) => (
                <label
                  key={s.statusId}
                  htmlFor={`status-${s.statusId}`}
                  className="element"
                >
                  <input
```

```
                      type="checkbox"
                      className="pop"
                      name="status"
                      id={`status-${s.statusId}`}
                      value={s.statusId}
                      checked={values.status.includes(s.statusId)}
                      onChange={(e) => {
                        const { checked, value } = e.target;
                        if (checked) {
                          setFieldValue('status', [...values.status, Number(value)]);
                        } else {
                          setFieldValue(
                            'status',
                            values.status.filter((v) => v !== Number(value)),
                          );
                        }
                      }}
                    />
                    <p className="pop-text">{s.name}</p>
                  </label>
                ))}
              </div>

              <div className="checker">
                <div className="text text-18">Projects</div>
                {projects?.map((p) => (
                  <label
                    key={p.projectId}
                    htmlFor={`project-${p.projectId}`}
                    className="element"
                  >
                    <input
                      type="checkbox"
                      className="pop"
                      name="projects"
                      id={`project-${p.projectId}`}
                      value={p.projectId}
                      checked={values.projects.includes(p.projectId)}
                      onChange={(e) => {
                        const { checked, value } = e.target;
                        if (checked) {
                          setFieldValue('projects',              [...values.projects,
Number(value)]);
                        } else {
                          setFieldValue(
                            'projects',
                            values.projects.filter((v) => v !== Number(value)),
                          );
                        }
                      }}
                    />
                    <p className="pop-text">{p.projectName}</p>
                  </label>
                ))}
              </div>
            </Form>
          )}
        </Formik>
      </div>
    </div>
  );
};
```

```
Filter.propTypes = {
  handleModalTaskOpen: PropTypes.func.isRequired,
};

export default Filter;
```

TaskForm.js (Компонент форми редагування та створення завдання)

```
import React, { useContext, useState } from 'react';
import { Field, Form, Formik } from 'formik';
import { Button, IconButton, Link } from '@mui/material';
import ArchiveOutlinedIcon from '@mui/icons-material/ArchiveOutlined';
import CloseRoundedIcon from '@mui/icons-material/CloseRounded';
import { Link as RouterLink } from 'react-router-dom';
import * as Yup from 'yup';
import { useMutation, useQueryClient } from 'react-query';
import TaskName from '../tasks/elements/TaskName';
import TaskDescription from '../tasks/elements/TaskDescription';
import TaskAttachment from '../tasks/elements/TaskAttachment';
import Autocomplete from '../form-elements/Autocomplete';
import DateTime from '../tasks/elements/DateTime';
import TaskActionButtons from '../tasks/elements/TaskActionButtons';
import { LanguageContext } from '../../containers/language/Language';
import { updateTask, createTask } from '../../services/crud/tasks';

const user = JSON.parse(sessionStorage.getItem('token'));

const TaskForm = ({
  taskData, open, handleNotificationModal, onClose, statusArray, assignToOptions,
projects,
}) => {
  const { dictionary } = useContext(LanguageContext);
  const anchorRef = React.useRef(null);
  const queryClient = useQueryClient();

  const [openAttachmentList, setOpenAttachmentList] = useState(false);
  const [openAttachmentLinkDialogWindow, setOpenAttachmentLinkDialogWindow] =
useState(false);

  const handleToggleAttachmentList = () => {
    setOpenAttachmentList((prevOpen) => !prevOpen);
  };

  const handleOpenAttachmentLinkDialogWindow = () => {
    setOpenAttachmentLinkDialogWindow(true);
  };

  const handleCloseAttachmentList = (event) => {
    if (anchorRef.current && anchorRef.current.contains(event.target)) {
      return;
    }

    setOpenAttachmentList(false);
  };

  const handleCloseAttachmentLinkDialogWindow = () => {
    setOpenAttachmentLinkDialogWindow(false);
  };

  const mutation = useMutation(
    `edit-task-${taskData.taskId}`,
    (formData) => updateTask(taskData.taskId, formData),
```

```
  {
    onSuccess: () => {
      queryClient.invalidateQueries('tasks-main-pages');
    },
  },
);

const mutationCreate = useMutation(
  'create-task-new',
  (formData) => createTask(formData, user.id),
  {
    onSuccess: () => {
      queryClient.invalidateQueries('tasks-main-pages');
    },
  },
);

const taskFormValues = {
  status: taskData.statusId ? {
    statusId: taskData.statusId,
    name: taskData.statusName,
    color: taskData.statuColor,
  } : null,
  title: taskData.title || '',
  description: taskData.description || '',
  dateTimeStart: new Date(taskData.dateStart),
  dateTimeEnd: new Date(taskData.dateEnd),
  assignTo: taskData.assignToProfileId ? {
    profileId: taskData.assignToProfileId,
    name: taskData.assignToName,
    surname: taskData.assignToSurname,
    email: taskData.assignToEmail,
  } : null,
  project: taskData.projectId ? {
    projectId: taskData.projectId,
    projectName: taskData.projectName,
  } : null,
  inspector: taskData.inspectorProfileId ? {
    profileId: taskData.inspectorProfileId,
    name: taskData.inspectorName,
    surname: taskData.inspectorSurname,
    email: taskData.inspectorEmail,
  } : null,
  color: taskData.color || 'auto',
};

const taskFormSchema = Yup.object().shape({
  title: Yup.string().required('Enter title!').max(255, 'So long title'),
  description: Yup.string(),
  dateTimeStart: Yup
    .date('Invalid format')
    .typeError('Invalid format. mm/dd/yyyy hh:mm (a/p)m')
    .required('Select start date and time!'),
  dateTimeEnd: Yup
    .date('Invalid format')
    .typeError('Invalid format. mm/dd/yyyy hh:mm (a/p)m')
    .required('Select end date and time!'),
  assignTo: Yup.object().shape({}).nullable(),
  inspector: Yup.object().shape({}).nullable(),
  project: Yup.object().shape({}).nullable().required(),
  color: Yup.string().required(),
  status: Yup.object().shape({}).required(),
});
```

```
const prevOpen = React.useRef(open);

React.useEffect(() => {
  if (prevOpen.current === true && open === false) {
    anchorRef.current.focus();
  }

  prevOpen.current = open;
}, [open]);

const saveTask = (formData, { setSubmitting }) => {
  const data = {
    title: formData.title,
    description: formData.description,
    timeStart: formData.dateTimeStart,
    timeEnd: formData.dateTimeEnd,
    color: formData.color,
    projectId: formData.project.projectId,
    statusId: formData.status.statusId,
    assignToId: formData.assignTo?.profileId,
    inspectorId: formData.inspector?.profileId,
  };

  if (taskData.taskId) {
    mutation.mutate(data);
  } else {
    mutationCreate.mutate(data);
  }
  setSubmitting(false);
};

return (
  <Formik
    initialValues={taskFormValues}
    onSubmit={saveTask}
    validationSchema={taskFormSchema}
  >
    {({
      setFieldValue, values, isValid, dirty,
    }) => (
      <Form className="form form-task">
        <div
          className="color-task-back"
          style={{
            background: values.color,
          }}
        />
        <div className="modal-header">
          <div className="title-header text text-20">
            {dictionary.task.task}
            {' '}
            {taskData.taskId && (
              `[${taskData.taskId}]`
            )}
          </div>

          <Button
            type="submit"
            variant="contained"
            size="small"
            disabled={!(isValid && dirty)}
          >
            {dictionary.task.save}
```

```
            </Button>

            <IconButton
              aria-label="close"
              size="medium"
              onClick={onClose}
              sx={{
                marginRight: '5px',
              }}
            >
              <ArchiveOutlinedIcon />
            </IconButton>

            <IconButton
              aria-label="close"
              size="medium"
              onClick={onClose}
              sx={{
                marginRight: '5px',
              }}
            >
              <CloseRoundedIcon />
            </IconButton>
          </div>
          <div className="modal-body">
            <div className="modal-left-column modal-column">

              <TaskName color={values.color} />

              <TaskDescription />

              {taskData.taskId
                && (
                <TaskAttachment
                  taskId={taskData.taskId}
                  anchorRef={anchorRef}
                  attachedList={[]}
                  handleCloseAttachmentLinkDialogWindow={
                  handleCloseAttachmentLinkDialogWindow
                }
                  handleCloseAttachmentList={handleCloseAttachmentList}

handleOpenAttachmentLinkDialogWindow={handleOpenAttachmentLinkDialogWindow}
                  openAttachmentLinkDialogWindow={openAttachmentLinkDialogWindow}
                  handleToggleAttachmentList={handleToggleAttachmentList}
                  openAttachmentList={openAttachmentList}
                />
                )}
            </div>

            <div className="modal-right-column modal-column">

              <div className="modal-right-top">
                <div className="modal-row time-row" style={{ alignItems: 'center' }}>
                  <div className="text text-16" style={{ marginRight: '10px' }}>
                    {dictionary.status.status}
                  </div>
                  <Field
                    component={Autocomplete}
                    name="status"
                    placeholder={dictionary.status.status}
                    options={statusArray}
                    fullWidth
```

```
        optionLabel={
            (option) => option.name
          }
        optionEqual={
            (option, value) => option.statusId === value.statusId
          }
      />
    </div>
  </div>

  <div className="modal-right-top">
    <div className="modal-row time-row" style={{ alignItems: 'center' }}>
      <div className="text text-16" style={{ marginRight: '10px' }}>
        {dictionary.pages.project}
      </div>
      <Field
        component={Autocomplete}
        name="project"
        placeholder={dictionary.pages.project}
        options={projects}
        fullWidth
        optionLabel={
          (option) => option.projectName
        }
        optionEqual={
          (option, value) => option.projectId === value.projectId
        }
      />
    </div>
  </div>

  <div className="modal-right-top">
    <div className="modal-row time-row">
      <div className="text text-16">
        {dictionary.task.from}
      </div>
      <DateTime
        name="dateTimeStart"
        label="From"
        setFieldValue={setFieldValue}
        time={values.dateTimeStart}
      />
    </div>

    <div className="modal-row time-row">
      <div className="text text-16">
        {dictionary.task.till}
      </div>
      <DateTime
        name="dateTimeEnd"
        label="Till"
        setFieldValue={setFieldValue}
        time={values.dateTimeEnd}
      />
    </div>

    <div className="modal-row">
      <div className="row-right-side">
        <div className="block-right-col executor-view-block">
          <div className="text text-16 task-member">
            <div className="pre-member-block">
              {dictionary.tasksList.assignTo}
            </div>
```

```
                        <Field
                          component={Autocomplete}
                          name="assignTo"
                          placeholder={dictionary.task.unassigned}
                          options={assignToOptions}
                          optionLabel={
                            (option)     =>     `${option.name}     ${option.surname}
[${option.email}]`
                          }
                          optionEqual={
                            (option, value) => option.profileId === value.profileId
                          }
                        />
                      </div>

                      <div className="text text-16 task-member">
                        <div className="pre-member-block">
                          {dictionary.tasksList.inspector}
                        </div>

                        <Field
                          component={Autocomplete}
                          name="inspector"
                          placeholder={dictionary.task.unassigned}
                          options={assignToOptions}
                          optionLabel={
                            (option)     =>     `${option.name}     ${option.surname}
[${option.email}]`
                          }
                          optionEqual={
                            (option, value) => option.profileId === value.profileId
                          }
                        />
                      </div>

                      <div className="text text-16 task-member created">
                        <div className="pre-member-block">
                          {dictionary.tasksList.createdBy}
                        </div>
                        <Link
                          component={RouterLink}
                          to={`/profile/${taskData.createdByProfileId || user.id}`}
                          className="text-16"
                          underline="hover"
                          color="inherit"
                        >
                          {`${taskData.createdByName           ||           user.name}
${taskData.createdBySurname || user.surname}`}
                        </Link>
                      </div>
                    </div>
                  </div>
                </div>

                {taskData.taskId
                  && (
                  <TaskActionButtons handleNotificationModal={
                  () => handleNotificationModal([values.assignTo, values.inspector])
                }
                  />
                  )}
              </div>
```

```
            </div>
          </div>
        </Form>
      )}
    </Formik>
  );
};

export default TaskForm;
```

ProfilePage.js

```
import React, { useContext } from 'react';
import { useParams, Link as DomLink } from 'react-router-dom';
import PropTypes from 'prop-types';
import { useQuery } from 'react-query';
import Backdrop from '@mui/material/Backdrop';
import CircularProgress from '@mui/material/CircularProgress';
import Button from '@mui/material/Button';
import AvatarPage from '../../../components/info-page/AvatarPage';
import TitlePage from '../../../components/info-page/TitlePage';
import ContentProfilePage from '../../../components/info-page/ContentProfilePage';
import TasksPage from '../../../components/tasks/TasksPage';
import { LanguageContext } from '../../language/Language';
import {
  getProfileById, getTasksByProfileById,
} from '../../../services/crud/profiles';
import contextUser from '../../../services/context-user';

const ProfilePage = ({ handleModalTaskOpen }) => {
  const { dictionary } = useContext(LanguageContext);
  const { profileId } = useParams();
  const { user } = useContext(contextUser);

  const profileQuery = useQuery(
    `profile-${profileId}`,
    () => getProfileById(profileId),
    {
      manual: true,
    },
  );
  const profile = profileQuery?.data?.data?.data?.profile;
  const positions = profileQuery?.data?.data?.data?.positions;
  const chiefs = profileQuery?.data?.data?.data?.chiefs;
  const isLoadingProfile = profileQuery.isLoading;

  const tasksQueries = useQuery(
    `profile-tasks-${profileId}`,
    () => getTasksByProfileById(profileId),
    {
      manual: true,
    },
  );
  const tasks = tasksQueries?.data?.data?.data;
  return (
    <div className="main-task-body page-info">
      {!isLoadingProfile
        || (
        <Backdrop
          sx={{ color: '#fff', zIndex: (theme) => theme.zIndex.drawer + 1 }}
          open={isLoadingProfile}
        >
```

```
          <CircularProgress color="inherit" />
        </Backdrop>
        )}

      <div className="left-bar">
        {profile
          && (
          <AvatarPage
            avatarLink={profile.avatarLink}
            title={`${profile.name} ${profile.surname}`}
          />
          )}
        {Number(profileId) === Number(user.id)
          && (
          <Button
            fullWidth
            component={DomLink}
            to="edit"
            variant="contained"
          >
            {dictionary.edit}
          </Button>
          )}
      </div>
      <div className="content">
        {profile
          && (
            <>
              <TitlePage
                type={dictionary.pages.profile.profile}
                title={`${profile.name} ${profile.surname}`}
              />

              <div className="block-content">
                <ContentProfilePage
                  aboutMe={profile.aboutMe}
                  pm={chiefs}
                  position={positions}
                  email={profile.email}
                  birthday={profile.birthday || null}
                />
              </div>
            </>
          )}

        {tasks && (
          <div className="tasks-block">
            <TasksPage
              tasks={tasks}
              onOpen={handleModalTaskOpen}
            />
          </div>
        )}
      </div>
    </div>
  );
};
ProfilePage.propTypes = {
  handleModalTaskOpen: PropTypes.func.isRequired,
};

export default ProfilePage;
```

Header.js

```
import React, { useContext, useState, useEffect } from 'react';
import { useQuery } from 'react-query';
import siteIconSvg from '../../icons/citeIcon.svg';
import searchHeaderIcon from '../../icons/searchHeaderIcon.svg';
import HeaderProfile from '../../components/header/profile';
import ProjectHeaderList from './projectHeaderList';
import NotificationList from '../modal-windows/notificationList';
import HeaderControlPanel from '../../components/header/HeaderControlPanel';
import HeaderMenu from '../../components/header/HeaderMenu';
import { LanguageContext } from '../language/Language';
import { countUnreadNotifications } from '../../services/crud/notifications/index';

const user = JSON.parse(sessionStorage.getItem('token'));

const Header = ({ handleShowNotificationModal }) => {
  const { dictionary } = useContext(LanguageContext);
  const [amountState, setAmountState] = useState();
  const [viewNotificationList, setViewNotificationList] = useState('none');

  const closeNotificationList = () => {
    setViewNotificationList('none');
  };

  const openNotificationList = () => {
    setViewNotificationList('block');
  };

  const { data } = useQuery(
    'amountUnread',
    () => countUnreadNotifications(user.id),
  );

  useEffect(() => {
    setAmountState(data?.data?.data);
  }, [data?.data?.data]);

  return (
    <header className="site-header">
      <div className="left-header">

        <div className="site-logo">
          <img src={siteIconSvg} alt="logo" />
        </div>

        <HeaderControlPanel
          openNotificationList={openNotificationList}
          count={amountState || 0}
        />

        <div className="project-list-button list">
          <div className="icon">
            <img src={searchHeaderIcon} alt="Search header" />
          </div>
          <div className="text text-18">
            {dictionary.header.searchTitle}
          </div>
          <ProjectHeaderList />
        </div>
      </div>
```

```
      <div className="right-header">
        <HeaderMenu />

        <HeaderProfile user={user} />
      </div>
      <NotificationList
        displaySetting={viewNotificationList}
        onClose={closeNotificationList}
        handleShowNotificationModal={handleShowNotificationModal}
      />
    </header>
  );
};

export default Header;
```

ItemNotificationList.js (Компонент відображення сповіщень)

```
import React, { useContext } from 'react';
import Badge from '@mui/material/Badge';
import MailIcon from '@mui/icons-material/Mail';
import IconButton from '@mui/material/IconButton';
import MarkAsUnreadIcon from '@mui/icons-material/MarkAsUnread';
import DeleteIcon from '@mui/icons-material/Delete';
import { useMutation, useQueryClient } from 'react-query';
import noNotificationsIcon from '../../icons/noNotificationsIcon.svg';
import { LanguageContext } from '../../containers/language/Language';
import { dateFormat } from '../../services/time';
import { simpleRead, deleteNotification } from '../../services/crud/notifications';

const user = JSON.parse(sessionStorage.getItem('token'));

const ItemNotificationList = ({ itemsNotifications = null }) => {
  const { dictionary } = useContext(LanguageContext);
  const queryClient = useQueryClient();

  const readSimple = useMutation(
    'read-simple',
    (id) => simpleRead(id, user.id),
    {
      onSuccess: () => {
        queryClient.invalidateQueries('notification-list');
        queryClient.invalidateQueries('amountUnread');
      },
    },
  );

  const deleteSimple = useMutation(
    'read-simple',
    (id) => deleteNotification(id, user.id),
    {
      onSuccess: () => {
        queryClient.invalidateQueries('notification-list');
        queryClient.invalidateQueries('amountUnread');
      },
    },
  );

  if (itemsNotifications) {
    return itemsNotifications.map((notification) => (
      <div
```

```
        key={`norification-list-item-${notification.notificationId}-
${notification.notificationListId}`}
        className="notification-item"
      >
        <div className="notification-title">
          <div className="title-notification text text-15">
            <div style={{ marginRight: '2px' }}>
              {dictionary.header.notification.by}
              :
            </div>
            {' '}
            <b>{`${notification.creatorName} ${notification.creatorSurname}`}</b>
          </div>
          <div className="right-title-block">
            <div
              className="notification-time text text-12"
              title={dateFormat(notification.plannedAt)}
            >
              {dateFormat(notification.plannedAt)}
            </div>

            {user.id !== notification.creatorProfileId
              && (
              <IconButton
                onClick={() => readSimple.mutate(notification.notificationId)}
                size="small"
                disabled={notification.read}
              >
                <Badge
                  invisible={notification.read}
                  color="success"
                  variant="dot"
                >
                  {
                  notification.read
                    ? <MarkAsUnreadIcon />
                    : <MailIcon />
                }
                </Badge>
              </IconButton>
              )}

            {user.id === notification.creatorProfileId
              && (
                <IconButton
                  onClick={() => deleteSimple.mutate(notification.notificationId)}
                  size="small"
                >
                  <DeleteIcon />
                </IconButton>
              )}
          </div>
        </div>
        <div className="notification-text text">
          {notification.text}
        </div>
      </div>
    ));
  }
  return (
    <div className="no-notifications">
      <img src={noNotificationsIcon} alt="No notifications" />
```

```
      <div className="text text-24">
        {dictionary.noNotifications}
      </div>
    </div>
  );
};

export default ItemNotificationList;
```

NewNotification.js (Модальне вікно створення сповіщень)

```
import React, { useContext } from 'react';
import { Field, Form, Formik } from 'formik';
import * as Yup from 'yup';
import PropTypes from 'prop-types';
// materials [MUI]
import {
  IconButton, Modal, Button,
} from '@mui/material';
// icon materials [MUI]
import CloseRoundedIcon from '@mui/icons-material/CloseRounded';
// own containers
import { TextField } from 'formik-mui';
// own components
import { useMutation, useQueryClient } from 'react-query';
import Autocomplete from '../../components/form-elements/Autocomplete';
import DateTime from '../../components/tasks/elements/DateTime';
import { LanguageContext } from '../language/Language';
import { createNewNotification } from '../../services/crud/notifications';

const user = JSON.parse(sessionStorage.getItem('token'));

const NewNotification = ({
  state, handleModal, recipients,
}) => {
  const queryClient = useQueryClient();
  const { dictionary } = useContext(LanguageContext);

  const formValues = {
    text: '',
    recipient: state.employees || [],
    plannedAt: new Date(),
  };

  const mutation = useMutation(
    'create-notification',
    (data) => createNewNotification(data, user.id),
    {
      onSuccess: () => {
        queryClient.invalidateQueries('amountUnread');
        queryClient.invalidateQueries('notification-list');
      },
    },
  );

  const formSchema = Yup.object().shape({
    text: Yup.string().required('Enter your message'),
    recipient: Yup.array()
      .of(
        Yup.object().shape({
          profileId: Yup.number().required(),
```

```
      }).required('Please, choose the recipient'),
    ).required('Please, choose the recipient'),
  plannedAt: Yup
    .date()
    .typeError('Invalid format. mm/dd/yyyy hh:mm (a/p)m')
    .required('Select planned date and time!'),
});

const addNotification = (formData, { setSubmitting }) => {
  mutation.mutate(formData);
  setSubmitting(true);
};

return (
  <Modal
    open={state.show}
    onClose={handleModal}
    sx={{
      height: '100vh',
      overflow: 'auto',
    }}
  >
    <div className="modal-of modal-of-center notification-modal modal-of-show">
      <div className="modal notification-modal-view">
        <Formik
          initialValues={formValues}
          onSubmit={addNotification}
          validationSchema={formSchema}
        >
          {({
            setFieldValue, values, isValid, dirty,
          }) => (
            <Form className="form form-notification">
              <div className="modal-header">
                <div className="title-header text text-20">
                  {dictionary.newNotification}
                </div>

                <IconButton
                  aria-label="close"
                  size="medium"
                  onClick={handleModal}
                  sx={{
                    marginRight: '5px',
                  }}
                >
                  <CloseRoundedIcon />
                </IconButton>
              </div>
              <div className="modal-body">
                <Field
                  component={TextField}
                  className="form-field text text-19"
                  type="text"
                  name="text"
                  label={dictionary.text}
                  variant="standard"
                  placeholder={dictionary.text}
                  fullWidth
                  multiline
                />

                <Field
```

```
                    component={Autocomplete}
                    name="recipient"
                    label={dictionary.recipients}
                    multiple
                    placeholder={dictionary.notificationWillReceive}
                    variant="outlined"
                    options={recipients}
                    optionLabel={(option)    =>    `${option.name}    ${option.surname}
[${option.email}]`}
                    optionEqual={(option,    value)    =>    option.profileId    ===
value.profileId}
                    className="form-field"
                  />

                  <div className="form-field date-time">
                    <DateTime
                      name="plannedAt"
                      label={dictionary.plannedAt}
                      setFieldValue={setFieldValue}
                      time={values.plannedAt}
                      minDate={new Date()}
                      minTime={new Date()}
                    />
                  </div>

                  <div className="btn-row">
                    <Button
                      type="submit"
                      variant="contained"
                      size="small"
                      disabled={!(isValid && dirty)}
                    >
                      {dictionary.send}
                    </Button>
                  </div>
                </div>
              </Form>
            )}
          </Formik>
        </div>
      </div>
    </Modal>
  );
};

NewNotification.propTypes = {
  state: PropTypes.shape({
    show: PropTypes.bool.isRequired,
    employees: PropTypes.arrayOf(
      PropTypes.shape({}),
    ),
  }).isRequired,
  handleModal: PropTypes.func.isRequired,
};

export default NewNotification;
```