

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри )

## Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

ДЛЯ АДМІНІСТРУВАННЯ ЗАВДАНЬ ПРАЦІВНИКІВ

SOFTWARE TO ADMINISTER EMPLOYEE TASKS

Виконав(ла): студент(ка) 4 курсу, групи КНТ-121

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ВОЛЬСЬКИЙ Р.Є.

(ПРИЗВИЩЕ та ініціали)

Керівник КОЦУР М.І.

(ПРИЗВИЩЕ та ініціали)

Рецензент ІЛЬЯШЕНКО М.Б.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет «Запорізька політехніка»

Факультет КНТ  
Кафедра програмних засобів  
Ступінь вищої освіти бакалавр  
Спеціальність 121 Інженерія програмного забезпечення  
(код і найменування)  
Освітня програма (спеціалізація) Інженерія програмного забезпечення  
(назва освітньої програми (спеціалізації))

**ЗАТВЕРДЖУЮ**

Завідувач кафедри ПЗ, д.т.н, проф.  
Сергій СУББОТІН  
“      ”                                  2025 року

**З А В Д А Н Н Я**

**НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)**

ВОЛЬСЬКОГО Руслана Євгеновича

(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення для адміністрування завдань працівників. Software to Administer Employee Tasks

керівник проєкту (роботи) к.т.н., доцент, КОЦУР Михайло Ігорович,  
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “28” квітня 2025 року № 209

2. Строк подання студентом проєкту (роботи) 03 червня 2025 року

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

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів, плакатів) Слайди презентації

## 6. Консультанти розділів проєкту (роботи)

| Розділ              | ПРИЗВИЩЕ, ініціали та посада консультанта | Підпис, дата   |                           |
|---------------------|-------------------------------------------|----------------|---------------------------|
|                     |                                           | завдання видав | прийняв виконане завдання |
| 1-4 Основна частина | КОЦУР М.І., доцент                        |                |                           |
| Нормоконтроль       | КАМІНСЬКА Ж.К., асистент                  |                |                           |

7. Дата видачі завдання “28” квітня 2025 року.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів дипломного проєкту (роботи)                             | Строк виконання етапів проєкту (роботи) | Примітка     |
|-------|----------------------------------------------------------------------|-----------------------------------------|--------------|
| 1     | Постановка завдання роботи.                                          | 1 тиждень                               | Завдання, ТЗ |
| 2     | Аналіз предметної області.                                           | 1 тиждень                               | Розділ 1     |
| 3     | Вибір мови програмування та інших технологій розробки.               | 2 тиждень                               | Розділ 2     |
| 4     | Розробка структури програми.                                         | 2 тиждень                               | Розділ 3     |
| 5     | Розробка програми.                                                   | 3-4 тижні                               | Розділи 3, 4 |
| 6     | Тестування та експериментальне дослідження програмного забезпечення. | 5 тиждень                               | Розділ 4     |
| 7     | Оформлення пояснювальної записки та документів до неї.               | 6 тиждень                               | Додатки      |
| 8     | Нормоконтроль та рецензування.                                       | 7 тиждень                               |              |
| 9     | Захист роботи.                                                       | 8 тиждень                               |              |

Студент(ка)

\_\_\_\_\_  
( підпис )      Руслан ВОЛЬСЬКИЙ  
(Імя ПРИЗВИЩЕ)

Керівник проєкту (роботи)

\_\_\_\_\_  
( підпис )      Михайло КОЦУР  
(Імя ПРИЗВИЩЕ)

## РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:  
92 с., 3 табл., 33 рис., 3 дод., 18 джерел.

C#, VISUAL STUDIO, АДМІНІСТРУВАННЯ ЗАВДАНЬ, МОВА ПРОГРАМУВАННЯ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ПРОЄКТ.

Об'єкт дослідження – процес розробки програмного забезпечення для підтримки проєктної діяльності.

Предмет дослідження – програмні засоби для підтримки процесу адміністрування завдань працівників.

Мета роботи – розробка програмного забезпечення для адміністрування завдань працівників.

Матеріали, методи та технічні засоби: мова програмування C#, середовище розробки Visual Studio.

Результати. Розроблено проєктні рішення для створення програмного забезпечення для адміністрування завдань працівників. Створено програмне забезпечення для адміністрування завдань працівників. Проведено тестування розробленої програмної системи для адміністрування завдань працівників.

Висновки. Мету роботи досягнуто. Розроблено програмне забезпечення для адміністрування завдань працівників за допомогою мови програмування C# та середовища розробки Visual Studio.

Галузь використання – проєктна діяльність.

## ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 92 pages, 3 tables, 33 figures, 3 appendixes, 18 sources.

C#, VISUAL STUDIO, TASK ADMINISTRATION, PROGRAMMING LANGUAGE, SOFTWARE, PROJECT.

The object of research is the process of developing software to support project activities.

The subject of the research is software for supporting the process of administering employee tasks.

The purpose of this work is to develop the software for administering employee tasks.

Materials, methods and technical tools: C# programming language, Visual Studio development environment.

Results. Project solutions for the creation of software for administering employee tasks has been designed. The software for administering employee tasks has been developed. The developed software system for administering employee tasks has been tested.

Conclusions. The goal of the work has been achieved. The software tools for administering employee tasks using the C# programming language and the Visual Studio development environment have been developed.

Scope of use – project activities supporting.

## ЗМІСТ

|                                                                                                                    |    |
|--------------------------------------------------------------------------------------------------------------------|----|
|                                                                                                                    | С. |
| Перелік скорочень та умовних позначок .....                                                                        | 8  |
| Вступ .....                                                                                                        | 9  |
| 1 Аналіз предметної області .....                                                                                  | 11 |
| 1.1 Програмне забезпечення для адміністрування завдань працівників .....                                           | 11 |
| 1.2 Аналіз програмного забезпечення для адміністрування завдань працівників .....                                  | 21 |
| 1.3 Висновки за розділом 1 .....                                                                                   | 30 |
| 2 Матеріали і методи .....                                                                                         | 32 |
| 2.1 Вибір мови програмування .....                                                                                 | 32 |
| 2.2 Вибір середовища розробки для створення програмного забезпечення для адміністрування завдань працівників ..... | 35 |
| 2.3 Висновки за розділом 2 .....                                                                                   | 37 |
| 3 Опис програми .....                                                                                              | 38 |
| 3.1 Структура програмного забезпечення для адміністрування завдань працівників .....                               | 38 |
| 3.2 Функціонування програмного забезпечення для адміністрування завдань працівників .....                          | 40 |
| 3.3 Розробка бази даних програмного забезпечення для адміністрування завдань працівників .....                     | 42 |
| 3.4 Проєктування інтерфейсу програмного забезпечення для адміністрування завдань працівників .....                 | 44 |
| 3.5 Висновки за розділом 3 .....                                                                                   | 48 |
| 4 Експлуатація, тестування та експериментальне дослідження програми .....                                          | 50 |
| 4.1 Призначення й умови застосування програми .....                                                                | 50 |
| 4.2 Характеристики програми для адміністрування завдань працівників .....                                          | 51 |
| 4.3 Інструкція по експлуатації програми .....                                                                      | 52 |
| 4.3.1 Звернення до програми .....                                                                                  | 52 |
| 4.3.2 Вхідні й вихідні дані .....                                                                                  | 52 |

|                                                                                       |    |
|---------------------------------------------------------------------------------------|----|
| 4.3.3 Повідомлення.....                                                               | 53 |
| 4.4 Виконання програмного забезпечення для адміністрування завдань працівників .....  | 53 |
| 4.5 Тестування програмного забезпечення для адміністрування завдань працівників ..... | 57 |
| 4.6 Висновки за розділом 4 .....                                                      | 57 |
| Висновки .....                                                                        | 59 |
| Перелік джерел посилань .....                                                         | 62 |
| Додаток А Технічне завдання .....                                                     | 64 |
| Додаток Б Фрагмент тексту програми .....                                              | 68 |
| Додаток В Слайди презентації .....                                                    | 85 |

## **ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК**

MVC – Model-View-Controller;

AЗ – адміністрування завдань;

БД – база даних;

МП – мова програмування;

ПЗ – програмне забезпечення.

## ВСТУП

Програмне забезпечення для адміністрування завдань працівників є актуальним завдяки його здатності автоматизувати робочі процеси, підвищувати продуктивність та покращувати управління часом. Воно спрощує розподіл завдань, контроль за їхнім виконанням і дозволяє керівникам швидко відстежувати прогрес команди. Гнучкість і масштабованість таких рішень дає змогу ефективно працювати як в офісі, так і віддалено, що особливо важливо в умовах сучасних моделей роботи [1], [2].

Завдяки інтеграції з іншими програмними системами, програмні засоби для адміністрування завдань працівників стають ще зручнішим і менш трудомістким. Аналітичні можливості програмного забезпечення для адміністрування завдань працівників допомагають оцінювати продуктивність співробітників, аналізувати ефективність процесів і приймати обґрунтовані управлінські рішення [3], [4].

Крім того, сучасні програмні рішення для адміністрування завдань сприяють покращенню комунікації між працівниками, оскільки містять інструменти для обговорення завдань, залишення коментарів та обміну файлами. Серед популярних платформ, які вирішують ці завдання, можна відзначити Trello, Asana, Jira, Monday.com та ClickUp. Вони дозволяють ефективно організовувати робочий процес, мінімізуючи хаос та покращуючи взаємодію в команді [5], [6].

Таким чином, використання програмного забезпечення для адміністрування завдань є необхідним інструментом для сучасного бізнесу, який прагне підвищити ефективність, оптимізувати робочі процеси та забезпечити чітку координацію роботи співробітників [6], [7].

Проте деякі програми для адміністрування завдань працівників мають певні недоліки, які можуть ускладнювати їх використання. Однією з головних проблем є складність освоєння, особливо якщо програмне забезпечення має широкий функціонал і вимагає додаткового навчання персоналу. Це може

уповільнити впровадження системи в робочий процес та викликати опір серед співробітників. Ще одним недоліком є ризик перевантаження інформацією. Якщо в програмній системі для адміністрування завдань накопичується надто багато завдань, коментарів і сповіщень, працівники можуть відчувати стрес і витрачати багато часу на фільтрацію важливих даних. Деякі програмні засоби для адміністрування завдань працівників мають проблеми з інтеграцією, особливо якщо компанія використовує різні інструменти, які не підтримуються цим програмним забезпеченням. Це може спричинити додаткові витрати на налаштування або перехід на інші сервіси. Крім того, вартість ліцензій на такі програмні системи може бути високою, особливо для малих і середніх підприємств, які не завжди можуть дозволити собі дорогі корпоративні рішення. Питання безпеки також залишається актуальним, оскільки зберігання завдань, файлів та інших конфіденційних даних у хмарних сервісах може піддавати компанію ризикам витоку інформації.

Тому актуальною є розробка програмного забезпечення для адміністрування завдань працівників. У дипломній кваліфікаційній роботі бакалавра розв'язується актуальне завдання розробки програмного забезпечення для адміністрування завдань працівників.

Для досягнення поставленої мети у кваліфікаційній роботі бакалавра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для адміністрування завдань працівників;
- здійснити проєктування програмного забезпечення для адміністрування завдань працівників;
- створити програмне забезпечення для адміністрування завдань працівників;
- виконати тестування розробленого програмного забезпечення для адміністрування завдань працівників.

## **1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ**

### **1.1 Програмне забезпечення для адміністрування завдань працівників**

Адміністрування завдань працівників передбачає управління робочими процесами, розподілом обов'язків та контролем за виконанням поставлених задач. Використання спеціалізованого програмного забезпечення дозволяє автоматизувати ці процеси, забезпечуючи зручне планування, моніторинг виконання та аналіз продуктивності [1].

Адміністрування завдань працівників є ключовим елементом управління робочими процесами в організації. Воно включає планування, розподіл, контроль виконання та аналіз продуктивності завдань, які виконують співробітники. Ефективне адміністрування дозволяє оптимізувати робочий час, уникати перевантаження або простою ресурсів, а також забезпечувати своєчасне виконання проєктів [1], [2].

Сучасні підходи до адміністрування завдань ґрунтуються на використанні програмного забезпечення, яке автоматизує процеси управління. Такі системи дозволяють створювати завдання, призначати відповідальних осіб, відстежувати статус виконання, встановлювати пріоритети та строки завершення. Додаткові можливості, такі як інтеграція з календарями, сповіщення, діаграми Ганта та аналітичні звіти, забезпечують підвищення ефективності та прозорості роботи [1], [2].

Застосування цифрових програмних інструментів в адмініструванні завдань сприяє покращенню комунікації між членами команди, скороченню часу на організаційні процеси та підвищенню відповідальності кожного співробітника за виконання поставлених обов'язків [1], [2].

Адміністрування завдань працівників полягає в організації, координації та контролі робочих процесів з метою досягнення поставлених результатів. Воно включає розподіл обов'язків, визначення ключових пріоритетів, ефективне використання доступних ресурсів і постійний контроль за

виконанням запланованих дій. Завдяки правильному підходу до адміністрування можна забезпечити системність у виконанні завдань, підвищити продуктивність і досягти більшої ефективності в роботі [1], [2].

Впорядкована програмна система управління дозволяє оптимізувати використання часу, зменшити навантаження на працівників і підвищити їхню залученість до робочого процесу. Чітке визначення обсягів роботи допомагає уникати хаотичності та сприяє швидшому досягненню цілей. Планування та контроль забезпечують можливість коригування поточних дій відповідно до змін у робочому процесі, що дозволяє своєчасно реагувати на нові виклики [1], [2].

Застосування програмних інструментів адміністрування сприяє ефективній взаємодії між співробітниками, що дозволяє швидше узгоджувати рішення та забезпечувати якісне виконання завдань. Завдяки прозорій системі моніторингу можна відстежувати стан робочих процесів і вчасно виявляти можливі труднощі. Оптимальне розподілення ресурсів сприяє кращій організації роботи, а гнучкий підхід до управління дає змогу оперативно адаптуватися до змін [1], [2].

Процес адміністрування завдань працівників передбачає організацію, впорядкування та контроль за виконанням роботи, що сприяє злагодженій діяльності та підвищенню ефективності. Першим кроком у налагодженні такої системи є створення структурованого переліку робочих процесів, що дозволяє чітко зафіксувати всі необхідні дії та уникнути можливих пропущених завдань. Важливо фіксувати всю інформацію щодо майбутніх дій, охоплюючи як безпосередньо професійні обов'язки, так і супутні робочі процеси. Великі завдання доцільно поділяти на дрібніші етапи, що значно спрощує їх виконання та забезпечує системний підхід. Використання спеціальних інструментів, таких як цифрові платформи або традиційні способи ведення записів, допомагає підтримувати порядок та зручність у керуванні робочими процесами [1], [2].

Ефективне адміністрування вимагає визначення важливості кожного етапу роботи, що дозволяє раціонально розподіляти ресурси та не витратити час на другорядні завдання. Розставляння пріоритетів допомагає зосередитися на найбільш значущих аспектах і зменшити навантаження від непотрібних дій. Визначення ключових напрямків дозволяє зосередитися на обмеженій кількості найважливіших завдань, що потребують першочергового вирішення. Чітке структурування процесу виконання роботи дозволяє підтримувати продуктивність, а також дає змогу вчасно реагувати на зміни та оптимізувати підхід до організації діяльності [1], [2].

Ефективне адміністрування завдань співробітників передбачає чітке планування роботи та контроль її виконання. Визначення кінцевих термінів сприяє організованості процесів і забезпечує своєчасне виконання запланованих дій. Розподіл обов'язків між працівниками дозволяє уникати перевантаження та забезпечує рівномірний розподіл робочого навантаження [1], [2].

Щоб підвищити ефективність роботи, доцільно щодня переглядати перелік завдань, коригуючи їх відповідно до пріоритетності. Чітке планування дозволяє уникати ситуацій, коли важливі справи відкладаються або виконуються поспіхом. Контроль прогресу виконання завдань допомагає вчасно реагувати на можливі труднощі та забезпечує стабільну продуктивність [1], [2].

Для організації роботи зручно використовувати спеціалізовані програмні засоби, що сприяють систематизації процесів. Такі інструменти допомагають відстежувати статус завдань, нагадують про дедлайни та дозволяють ефективно координувати командну роботу. Використання цифрових платформ дає змогу автоматизувати частину процесів, що значно знижує ризик людських помилок [1], [2].

Окрему увагу слід приділяти розподілу пріоритетів. Фокусування на найважливіших завданнях допомагає уникати витрачання часу на другорядні справи. Виділення ключових напрямів роботи дозволяє співробітникам чітко

розуміти, які завдання потребують негайного виконання, а які можуть бути відкладені на пізніше [1], [2].

Щоб запобігти типовим помилкам в управлінні робочими процесами, важливо уникати надмірного перевантаження співробітників. Раціональне планування допомагає підтримувати стабільний ритм роботи без стресу і зайвої напруги. Водночас недостатня увага до організації процесів може призвести до втрати часу та невиконання поставлених завдань у визначені терміни [1], [2].

Робочий процес повинен бути збалансованим, а планування — реалістичним. Чітка структура завдань, контроль за їхнім виконанням та використання сучасних технологій сприяють підвищенню продуктивності та якості роботи [1], [2].

Організація роботи команди потребує ретельного підходу до планування, координації та контролю за виконанням завдань. Це передбачає ефективний розподіл ресурсів, визначення ключових напрямків діяльності та забезпечення своєчасного досягнення поставлених цілей. У процесі управління завданнями важливими є такі аспекти, як формулювання робочих обов'язків, визначення послідовності дій, спостереження за динамікою виконання, встановлення строків та аналіз отриманих результатів [2], [3].

Визначення кола обов'язків дозволяє чітко окреслити функції, які необхідно виконати. Це сприяє усвідомленню кожного робочого етапу та мінімізує ризик виникнення невизначеності. Встановлення пріоритетів допомагає орієнтувати діяльність у правильному напрямку, розподіляючи ресурси відповідно до важливості та терміновості окремих завдань. Закріплення відповідальності за конкретні процеси гарантує ефективний контроль за виконанням робіт і забезпечує рівномірне навантаження серед учасників команди [2], [3].

Для успішного досягнення цілей важливо забезпечити систематичний моніторинг виконання робіт. Регулярне оновлення інформації про стан завдань дозволяє швидко реагувати на можливі труднощі та коригувати план у

разі необхідності. Контроль за дотриманням строків дає змогу зберігати високий рівень організованості, уникаючи небажаних затримок [2], [3].

Ефективне управління процесами має безліч переваг. Воно дозволяє підвищити продуктивність, оскільки кожен учасник процесу працює згідно з чітко визначеним планом. Раціональне використання робочого часу сприяє оперативному вирішенню задач і зменшенню ризику перевантаження. Злагоджена організація роботи позитивно впливає на загальний рівень стресу, знижуючи невизначеність і підвищуючи передбачуваність процесів. Крім того, чіткий розподіл завдань підвищує рівень відповідальності серед виконавців, сприяючи більшій зацікавленості у досягненні результату. Завдяки налагодженій комунікації між учасниками команди забезпечується ефективна взаємодія, що сприяє успішному виконанню проєктів та досягненню стратегічних цілей [2], [3].

Адміністрування завдань працівників — це не просто контроль за виконанням робіт, а складний процес, що включає розподіл обов'язків, ефективне використання ресурсів і постійний аналіз продуктивності. Проєкти складаються з окремих етапів, які необхідно ретельно координувати, враховуючи навантаження та компетенції виконавців [4], [5].

Організація робочого процесу спрямована на те, щоб забезпечити ефективне виконання завдань без надмірного тиску на працівників, що могло б негативно позначитися на їхньому фізичному чи емоційному стані. Успішне адміністрування дозволяє командам працювати більш узгоджено, дотримуючись встановлених пріоритетів і термінів [4], [5].

Часто складність у виконанні завдань пов'язана з нечітким розумінням послідовності дій. Продумане планування дозволяє структурувати робочий процес, забезпечуючи своєчасне виконання кожного етапу. Важливим аспектом є прозорість у визначенні відповідальності: чітке розмежування обов'язків допомагає уникнути плутанини та зменшує ризики дублювання зусиль [4], [5].

Контроль за строками виконання завдань є ключовим інструментом у процесі адміністрування. Системний підхід до планування дозволяє своєчасно виявляти можливі затримки, коригувати навантаження та оптимізувати використання ресурсів [4], [5].

Взаємодія між учасниками робочого процесу відіграє вирішальну роль у досягненні спільних цілей. Регулярний обмін інформацією сприяє швидкому усуненню проблем і коригуванню стратегії виконання завдань. Запровадження чітких механізмів зворотного зв'язку дозволяє ефективно контролювати результати роботи та підвищувати мотивацію персоналу [4], [5].

Проекти мають свій життєвий цикл, що складається з кількох взаємопов'язаних етапів. На кожному з них виконуються певні дії — від аналізу та планування до реалізації та оцінки досягнутих результатів. Дотримання системного підходу до адміністрування забезпечує безперебійну роботу та сприяє успішному завершенню завдань [4], [5].

Програмні рішення для адміністрування завдань сприяють структурованому плануванню, організації та оптимізації робочих процесів. Вони дозволяють уникнути зайвих витрат часу на рутинні дії, підвищуючи загальну ефективність виконання обов'язків та усуваючи необхідність у ручному контролі [4], [5].

Завдяки впровадженню таких технологій процес координації діяльності стає більш впорядкованим. Всі необхідні дані доступні в єдиному середовищі, що спрощує управління ресурсами та часовими рамками. Автоматизоване адміністрування дозволяє не лише ефективніше розподіляти завдання, а й контролювати їх виконання без додаткових зусиль [4], [5].

Застосування цифрових рішень у цій сфері підвищує відповідальність працівників, допомагаючи чітко визначати обсяг робіт та строки їхнього виконання. Це значно зменшує ризик помилок, втрати інформації чи плутанини в процесі комунікації [4], [5].

Спільна робота між командами набуває нового рівня завдяки можливості швидкого обміну повідомленнями та відстеження змін у режимі

реального часу. Використання централізованого середовища для взаємодії дозволяє відмовитися від численних листів та зайвих зустрічей, спрощуючи комунікаційні процеси [4], [5].

Зниження навантаження та оптимізація робочого процесу допомагають запобігти вигоранню та зменшити рівень стресу серед працівників. Виконання завдань стає більш збалансованим, що позитивно впливає на загальну продуктивність [4], [5].

Організоване та прозоре адміністрування сприяє підвищенню залученості співробітників. Чітко окреслені обов'язки та ефективний розподіл часу дозволяють уникнути хаосу, забезпечуючи високу якість виконання роботи. Це сприяє не лише своєчасному досягненню результатів, а й покращує мотивацію учасників процесу [4], [5].

Адміністрування завдань працівників передбачає ефективну організацію, планування та контроль робочого процесу. Використання спеціалізованого програмного забезпечення допомагає структурувати робочі обов'язки, розподіляти їх між виконавцями та відстежувати виконання. Завдяки цьому зменшується ризик втрати важливих завдань і підвищується продуктивність.

Розбивка складних робочих процесів на менші етапи дозволяє спростити виконання завдань і чітко розподілити відповідальність. Впровадження систем автоматизованого контролю забезпечує можливість встановлення термінів, визначення пріоритетів та аналізу динаміки виконання через інструменти візуалізації, такі як діаграми чи інтерактивні дошки [4], [5].

Ефективна взаємодія між співробітниками відіграє важливу роль у досягненні загальних цілей. Використання централізованої платформи для обміну інформацією усуває необхідність у нескінченних ланцюжках повідомлень та запитів. Спільний доступ до файлів, оперативні оновлення та комунікація в реальному часі сприяють злагодженій роботі команди [4], [5].

Контроль виконання завдань дозволяє уникнути хаосу та зайвого контролю за діями кожного працівника. Використання системи моніторингу

надає повний огляд процесів, допомагаючи виявити пріоритетні напрями роботи, оцінити продуктивність та оперативно реагувати на можливі труднощі [4], [5].

Раціональне використання робочого часу є ключовим чинником ефективної діяльності. Відстеження тривалості виконання різних завдань дає змогу оцінити, наскільки точно дотримуються встановлені строки, а також аналізувати, які процеси потребують оптимізації. Такі дані допомагають коригувати навантаження, підвищуючи ефективність використання ресурсів [4], [5].

Встановлення пріоритетів у виконанні завдань є необхідним для ефективного адміністрування робочого процесу. Велика кількість обов'язків може ускладнювати визначення того, з чого варто розпочати. Використання сучасних інструментів дозволяє позначати важливість кожного завдання, допомагаючи учасникам робочого процесу швидше орієнтуватися та приймати правильні рішення [4], [5].

У робочих процесах часто присутні завдання, які повторюються із певною періодичністю. Їх виконання може бути автоматизоване завдяки інструментам, які дозволяють налаштувати регулярне створення таких завдань без необхідності повторного внесення вручну. Це сприяє більш ефективному розподілу часу та зменшенню рутинної роботи [4], [5].

Зручне відображення даних відіграє важливу роль у керуванні завданнями. Використання графічних інструментів, таких як діаграми, календарі чи інтерактивні панелі, дозволяє подати загальний стан роботи у графічному вигляді. Це спрощує оцінку прогресу, допомагає контролювати часові рамки та зручніше розподіляти ресурси. Наявність інтуїтивно зрозумілого інтерфейсу дає змогу легко адаптувати робочий процес під конкретні потреби [4], [5].

Оцінка результатів роботи є важливим етапом управління завданнями. Аналітичні інструменти дозволяють отримувати детальну інформацію про продуктивність, ефективність розподілу ресурсів та можливі відхилення від

початкового плану. Завдяки цьому можна своєчасно виявляти проблеми, коригувати стратегію та підвищувати загальну ефективність команди [4], [5].

Для ефективного управління завданнями працівників необхідно спочатку створити чіткий список всіх обов'язків, що забезпечить основу для подальшого планування і реалізації проектів. Системи управління проектами дозволяють не лише організувати завдання, але й вказувати додаткову інформацію, таку як строки виконання та важливість кожного завдання. Це сприяє кращій організації роботи та забезпечує прозорість процесів [5], [6].

Важливою складовою є визначення пріоритетів для кожного завдання. Завдяки зручним інтерфейсам можна позначати важливість завдань і присвоювати їм відповідні мітки, що дозволяє команді швидше зорієнтуватися та зосередитись на найважливіших аспектах роботи, не витрачаючи час на менш пріоритетні завдання [5], [6].

Також вкрай корисним є планування важливих заходів і етапів роботи. Всі проектні завдання та події можуть бути централізовано зібрані в одному місці, що спрощує їх моніторинг і забезпечує кращу координацію. Використання календарів і розкладів допомагає тримати всі процеси під контролем, спрощує відстеження термінів і результатів [6], [7].

Ключовим елементом адміністрування є також ефективне призначення завдань відповідальним працівникам. Програмне забезпечення дозволяє автоматично призначати завдання, враховуючи можливості та доступність працівників, що гарантує своєчасне виконання завдань. При цьому кожен отримує повідомлення про призначення завдання і всі необхідні деталі для його виконання [6], [7].

Моніторинг прогресу є важливою складовою управлінського процесу. Завдяки сучасним інструментам можна відстежувати виконання завдань, порівнювати фактичні результати з плановими і вчасно виявляти відхилення. Візуалізація даних у вигляді таблиць, діаграм чи календарів дозволяє командам мати чітке уявлення про хід виконання завдань [6], [7].

Завдяки аналітичним функціям можна зручно відслідковувати ефективність команди, а також виявляти можливі проблеми на ранніх етапах. Звіти про виконання завдань, використання ресурсів та часу дають змогу детально оцінити, як оптимізувати робочі процеси і підвищити продуктивність [7], [8].

Вибір відповідного програмного забезпечення для управління проектами може значно полегшити процеси планування, організації та виконання завдань, а також допомогти в оптимізації роботи команди [7], [8].

Для ефективного управління завданнями працівників важливо мати чітко організований процес, що включає регулярне планування та оцінку виконаної роботи. Створення детального розкладу дозволяє визначити пріоритети і систематизувати виконання завдань, зосереджуючись на найбільш важливих завданнях, що сприяє підвищенню продуктивності. Важливим аспектом є уникнення відволікань та постійне вдосконалення власної ефективності. Регулярне оцінювання результатів дозволяє виявляти слабкі місця в процесі і коригувати стратегію роботи [7], [8].

Програмні інструменти для управління проектами значно спрощують цей процес, автоматизуючи багато етапів. Вони можуть бути як простими, так і складними, в залежності від масштабів організації та складності проектів. Ручні інструменти, як діаграми та планувальники, дозволяють керувати невеликими проектами, але мають обмежені можливості. Цифрові рішення, як електронні таблиці або більш складні системи, дають змогу інтегрувати різні процеси в межах організації, що значно покращує ефективність. Водночас онлайн системи дозволяють здійснювати управління проектами в хмарі, що дає можливість працювати в будь-якому місці та ефективно співпрацювати з командою на відстані [7], [8].

Програмне забезпечення для управління проектами забезпечує повний контроль над усіма завданнями та процесами, дозволяючи оптимізувати роботу працівників. Такі системи дають змогу відстежувати прогрес, аналізувати продуктивність і проводити моніторинг виконання завдань, що

дозволяє приймати обґрунтовані рішення щодо подальших дій. Вони є незамінним інструментом для підтримки організованості та високої продуктивності в команді [7], [8].

Відзначено, що адміністрування завдань працівників передбачає управління робочими процесами, розподілом обов'язків та контролем за виконанням поставлених задач. Використання спеціалізованого програмного забезпечення дозволяє автоматизувати ці процеси, забезпечуючи зручне планування, моніторинг виконання та аналіз продуктивності.

## 1.2 Аналіз програмного забезпечення для адміністрування завдань працівників

Проаналізуємо програмне забезпечення для адміністрування завдань працівників [9]-[14].

Програмне забезпечення TickTick (рис. 1.1) використовується для адміністрування завдань працівників, дозволяючи ефективно організовувати робочий процес [9], [12].

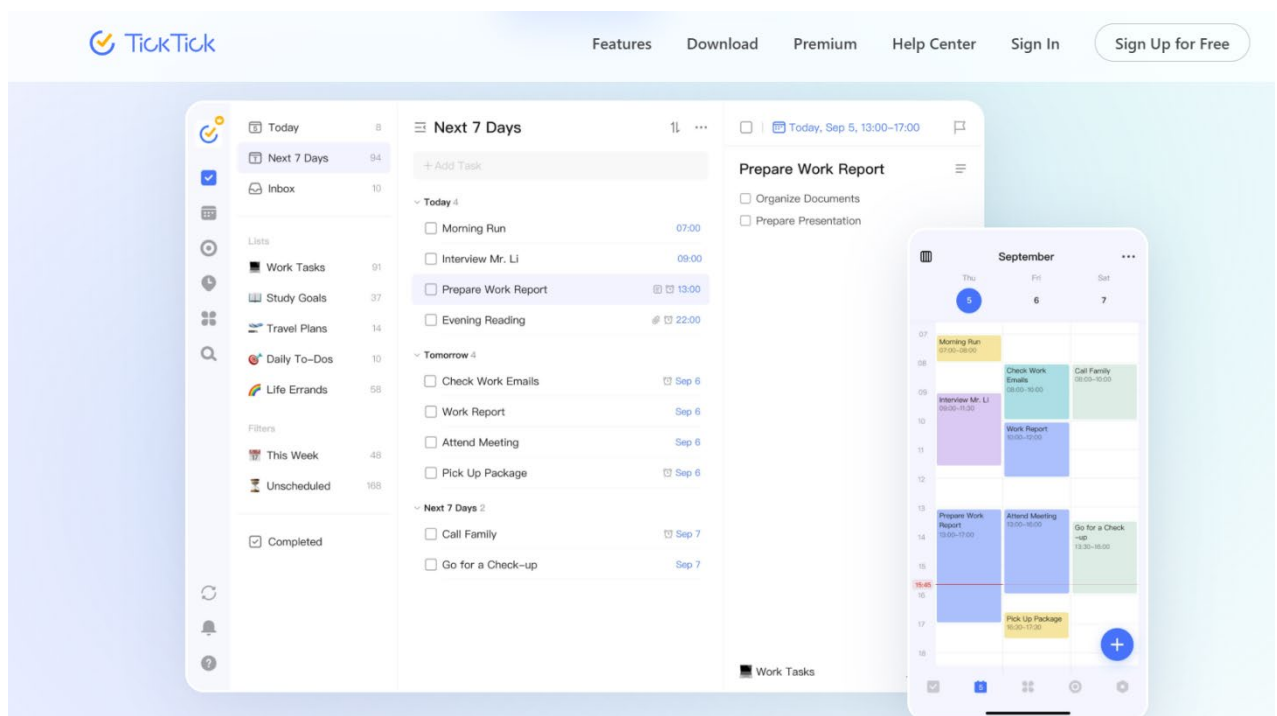


Рисунок 1.1 – Програмне забезпечення TickTick [12]

Завдяки простому інтерфейсу та широкому набору можливостей програмне забезпечення TickTick сприяє впорядкуванню робочих процесів, допомагаючи менеджерам та їхнім командам структурувати завдання, стежити за їхнім виконанням і розподіляти навантаження між співробітниками [9], [12].

Однією з ключових особливостей є можливість сортування завдань за папками, що дає змогу отримати чітку картину всіх робочих процесів та ефективніше розподіляти робочий час. Крім звичайного планування, передбачена функція визначення тривалості виконання завдань, що дає змогу заздалегідь оцінити, скільки часу знадобиться на кожну дію. Це дозволяє більш точно контролювати навантаження працівників і оптимізувати розклад робочого дня [9], [12].

Програмне забезпечення TickTick також відзначається високим рівнем безпеки. Всі дані залишаються конфіденційними, а в разі видалення облікового запису інформація повністю стирається. У разі порушення безпеки програмне забезпечення TickTick оперативно повідомляє користувачів, що підвищує довіру до сервісу [9], [12].

Для полегшення роботи з програмним забезпеченням TickTick передбачена можливість голосового введення завдань, що робить процес адміністрування швидшим та зручнішим. Ця функція дозволяє створювати нові записи за допомогою голосових команд, що особливо корисно в динамічному робочому середовищі [9], [12].

Програмне забезпечення TickTick пропонує різні варіанти використання, зокрема безкоштовний і розширений преміальний план. Базова версія надає основні інструменти для керування завданнями, тоді як преміум-версія дозволяє створювати більше списків, залучати додаткових учасників та прикріплювати файли до завдань. Додатково передбачено функціонал для аналізу продуктивності, що допомагає оцінювати прогрес виконання як особистих, так і командних завдань [9], [12].

Програмне забезпечення TickTick є потужним рішенням для адміністрування завдань працівників, яке допомагає керувати робочими

процесами, підвищувати продуктивність і забезпечувати чіткий контроль над усіма аспектами організації роботи [9], [12].

Основними характеристиками програмного забезпечення TickTick є такі [9], [12]:

- призначення: програмне забезпечення TickTick орієнтоване на адміністрування завдань працівників, допомагає організовувати робочий процес та підвищувати продуктивність [9], [12];

- структурованість: завдання можна групувати у списки та папки, що спрощує управління робочими процесами [9], [12];

- планування: програмне забезпечення TickTick дозволяє встановлювати дати початку та завершення завдань, а також вказувати очікувану тривалість їх виконання [9], [12];

- голосове введення: програмне забезпечення TickTick підтримує функцію перетворення мовлення на текст, що полегшує додавання нових завдань [9], [12];

- спільна робота: програмне забезпечення TickTick дає змогу залучати до завдань інших працівників і розподіляти обов'язки в межах проєктної команди [9], [12];

- нагадування: програмне забезпечення TickTick налаштовує сповіщення про наближення строків виконання завдання [9], [12];

- безпека: програмне забезпечення TickTick гарантує захист даних користувачів, не передає їх третім сторонам і оперативно повідомляє про можливі загрози [9], [12];

- гнучкість тарифів: програмне забезпечення TickTick має безкоштовну версію з базовими можливостями та преміум-план із розширеним функціоналом [9], [12];

- підтримка платформ: програмне забезпечення TickTick доступне на різних пристроях, включаючи комп'ютери, смартфони та планшети [9], [12];

– обмеження: деякі розширені можливості програмного забезпечення TickTick доступні лише у платній версії, а також є обмежена інтеграція з деякими корпоративними сервісами [9], [12].

Серед недоліків програмного забезпечення TickTick можна відзначити обмежену інтеграцію з деякими популярними корпоративними інструментами, що може ускладнювати його використання в масштабних робочих середовищах. Крім того, хоча базові функції доступні безкоштовно, деякі важливі можливості, такі як розширена аналітика та підвищена кількість списків, доступні лише у преміальній версії, що може бути недоліком для користувачів, які шукають повністю безкоштовне рішення. Також у програмному забезпеченні TickTick відсутні гнучкі налаштування інтерфейсу, що може не задовольнити потреби тих, хто звик до більш зручних програмних рішень для адміністрування завдань працівників [9], [12].

Програмне забезпечення HubSpot (рис. 1.2) створене для адміністрування завдань працівників, забезпечуючи ефективну організацію робочого процесу [10], [13].

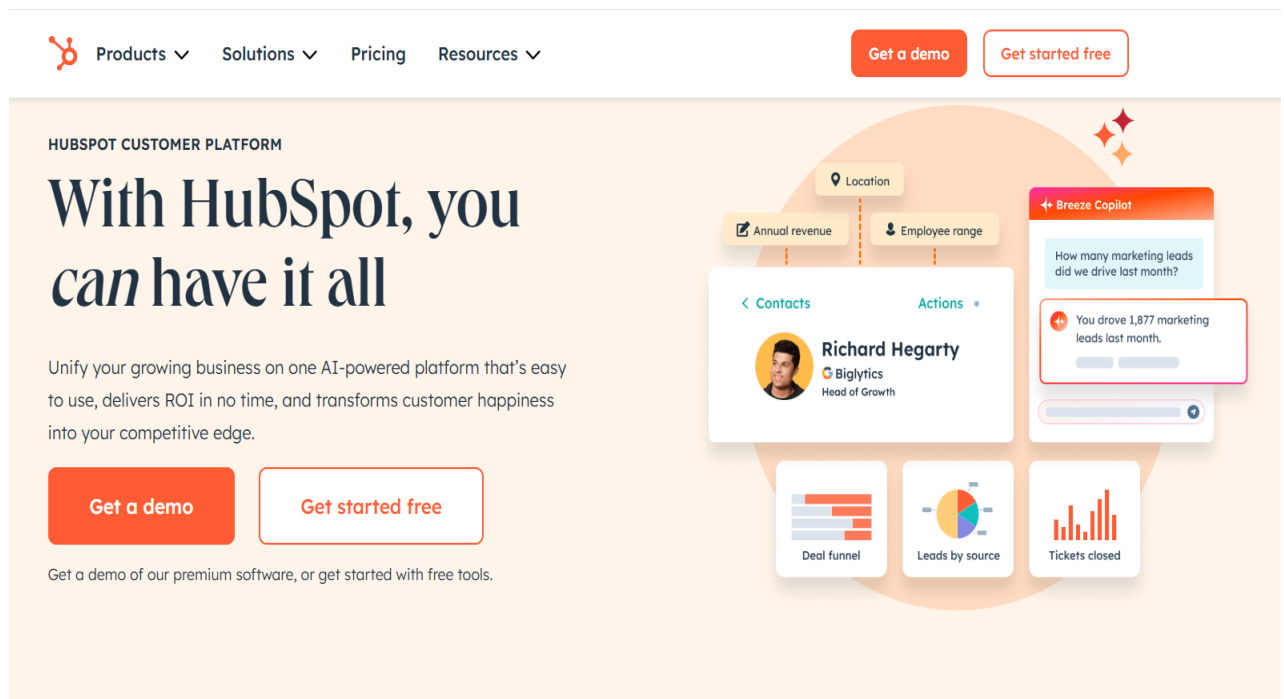


Рисунок 1.2 – Програмне забезпечення HubSpot [13]

Програмне забезпечення HubSpot інтегровано у програмну платформу для управління підприємством, що дозволяє працівникам виконувати робочі завдання без необхідності використовувати окремі програми. Завдяки цьому керування процесами продажів та маркетингу стає більш структурованим, оскільки система автоматично формує завдання відповідно до робочих процесів команди [10], [13].

За допомогою HubSpot можна контролювати виконання завдань, дотримання термінів виконання завдань і планувати навантаження працівників компанії на майбутнє, використовуючи зручну систему черг. Програмне забезпечення HubSpot дозволяє автоматизувати багато рутинних процесів, що значно полегшує адміністрування завдань працівників і підвищує продуктивність команди [10], [13].

Основний функціонал HubSpot для керування завданнями доступний безкоштовно, проте більшість компаній використовують розширені версії з додатковими можливостями. Вартість залежить від обраного тарифного плану, а витрати можуть бути значними, особливо для великих підприємств. Водночас інтеграція модуля керування завданнями з маркетинговими інструментами робить його частиною єдиної екосистеми, що виправдовує вкладені кошти [10], [13].

Програмне забезпечення HubSpot має такі особливості [10], [13]:

- інтеграція з програмною платформою для управління підприємством: програмне забезпечення HubSpot тісно пов'язане з системою управління підприємством, що дозволяє автоматично створювати, розподіляти та відстежувати завдання працівників [10], [13];

- автоматизація процесів: завдання можуть автоматично генеруватися на основі певних подій або тригерів, що допомагає зменшити ручну працю та підвищити продуктивність команди [10], [13];

- моніторинг термінів виконання завдань: вбудовані інструменти дозволяють контролювати строки виконання завдань, надсилати нагадування та відстежувати прогрес у режимі реального часу [10], [13];

- організація робочих процесів: програмне забезпечення HubSpot надає можливість створення черг завдань, налаштування пріоритетів і розподілу робочого навантаження між співробітниками [10], [13];

- аналітика та звітність: вбудовані інструменти програмного забезпечення HubSpot дозволяють аналізувати ефективність виконання завдань, оцінювати продуктивність працівників та оптимізувати робочі процеси [10], [13];

- гнучкість налаштувань: користувачі програмного забезпечення HubSpot можуть змінювати параметри завдань, встановлювати власні шаблони та адаптувати систему під потреби компанії [10], [13];

- сповіщення та нагадування: програмне забезпечення HubSpot надсилає автоматичні повідомлення про майбутні та прострочені завдання, що допомагає уникати затримок у роботі [10], [13];

- обмеження безкоштовної версії: базова версія програмного забезпечення HubSpot пропонує обмежені можливості керування завданнями, а для доступу користувачів до повного функціоналу необхідно придбати платний план [10], [13].

Серед недоліків програмного забезпечення HubSpot можна відзначити високу вартість розширених функцій, що може стати значним фінансовим навантаженням для малих компаній. Крім того, попри інтеграцію з CRM і маркетинговими інструментами, налаштування автоматизації завдань потребує часу та певного рівня технічної обізнаності. Інтерфейс системи містить безліч можливостей, що може ускладнити адаптацію нових користувачів і вимагати додаткового навчання. Також слід враховувати, що деякі функції адміністрування завдань менш гнучкі порівняно з іншими спеціалізованими рішеннями, що може обмежувати можливості персоналізації робочих процесів [10], [13].

Програмне забезпечення Wrike (рис. 1.3) забезпечує керівникам і командам гнучкі інструменти для організації робочих процесів. Воно дозволяє налаштовувати управління завданнями відповідно до специфічних вимог,

підтримуючи різні підходи, такі як канбан-дошки чи діаграми Ганта. Завдяки можливості призначення відповідальних, додавання позначок і спільної роботи над проєктами, Wrike полегшує розподіл обов'язків між працівниками та сприяє ефективному контролю виконання завдань [11], [14].

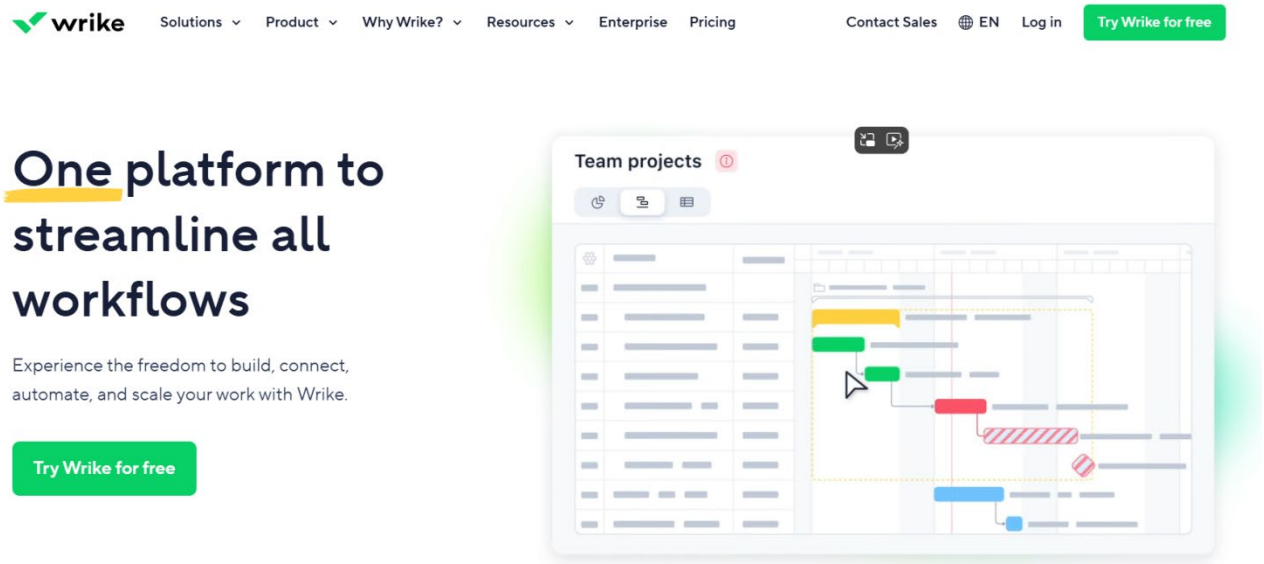


Рисунок 1.3 – Програмне забезпечення Wrike [14]

Програмне забезпечення Wrike дає змогу стежити за прогресом роботи, використовуючи вбудовані панелі з аналітикою, які надають актуальні дані в режимі реального часу. Функціонал звітності дозволяє отримувати налаштовані огляди про статус завдань, що полегшує моніторинг ефективності роботи команди. Особливо корисною ця можливість є для проєктів із залученням зовнішніх учасників, адже Wrike дозволяє організовувати взаємодію з ними без необхідності надання повного доступу до внутрішніх ресурсів [11], [14].

Програмне забезпечення Wrike має такі особливості [11], [14]:

– гнучке управління завданнями: програмне забезпечення Wrike дозволяє адаптувати робочі процеси під специфічні потреби команди розробників, використовуючи канбан-дошки, діаграми Ганта або стандартні списки [11], [14];

– спільна робота: програмне забезпечення Wrike надає можливість призначати виконавців, додавати коментарі, позначати колег та обмінюватися файлами безпосередньо у завданні [11], [14];

– моніторинг виконання: програмне забезпечення Wrike надає вбудовані панелі управління та аналітики дозволяють відстежувати статус проєктів у режимі реального часу [11], [14];

– автоматизація процесів: програмне забезпечення Wrike підтримує налаштування повторюваних завдань, автоматичних нагадувань і тригерів для підвищення ефективності роботи [11], [14];

– інтеграція з іншими сервісами: програмне забезпечення Wrike підтримує сумісність із такими інструментами, як Google Drive, Microsoft Teams, Slack, дозволяє легко об'єднувати Wrike з наявною інфраструктурою компанії [11], [14];

– гнучкі налаштування доступу: програмне забезпечення Wrike підтримує можливість надання різних рівнів доступу для внутрішніх і зовнішніх учасників, що спрощує спільну роботу над проєктами [11], [14];

– розширена звітність: програмне забезпечення Wrike підтримує створення налаштованих звітів і візуалізацій для аналізу продуктивності команди [11], [14];

– висока масштабованість: програмне забезпечення Wrike підходить для використання як у невеликих командах, так і у великих корпораціях із численними проєктами [11], [14];

– мобільна доступність: наявність мобільної версії програмного забезпечення Wrike дає змогу користувачам керувати своїми завданнями з будь-якого пристрою [11], [14];

– безпека даних: програмне забезпечення Wrike підтримує багаторівневий захист інформації, включаючи контроль доступу та резервне копіювання [11], [14].

Попри широкі можливості адміністрування завдань, програмне забезпечення Wrike може виявитися занадто складною для невеликих

проектів, оскільки розширені налаштування та гнучка структура вимагають часу на освоєння. Сповіщення іноді можуть бути надмірними, а їхня деталізація налаштувань не завжди інтуїтивно зрозуміла. Окремі функції, такі як управління фінансами або інтеграція з деякими зовнішніми сервісами, потребують доопрацювання, що може змусити користувачів звертатися до додаткових інструментів [11], [14].

Порівняльну характеристику програмного забезпечення для адміністрування завдань працівників наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для адміністрування завдань працівників

| Критерій порівняння                           | TickTick [12] | HubSpot [13] | Wrike [14] |
|-----------------------------------------------|---------------|--------------|------------|
| Гнучкість налаштувань                         | +—            | —            | +          |
| Співпраця в команді                           | +—            | +            | +          |
| Зручність інтерфейсу                          | +             | +            | +—         |
| Інтеграція з іншими сервісами                 | +—            | +            | +          |
| Підтримка різних стилів управління завданнями | +—            | —            | +          |

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато програмних засобів для адміністрування завдань працівників. Проте деякі програми для адміністрування завдань працівників мають певні недоліки, які можуть ускладнювати їх використання. Однією з головних проблем є складність освоєння, особливо якщо програмне забезпечення має широкий функціонал і вимагає додаткового навчання персоналу. Це може уповільнити впровадження системи в робочий процес та викликати опір серед співробітників. Ще одним недоліком є ризик перевантаження інформацією. Якщо в програмній системі для адміністрування завдань накопичується надто багато завдань, коментарів і сповіщень,

працівники можуть відчувати стрес і витрачати багато часу на фільтрацію важливих даних. Деякі програмні засоби для адміністрування завдань працівників мають проблеми з інтеграцією, особливо якщо компанія використовує різні інструменти, які не підтримуються цим програмним забезпеченням. Це може спричинити додаткові витрати на налаштування або перехід на інші сервіси. Крім того, вартість ліцензій на такі програмні системи може бути високою, особливо для малих і середніх підприємств, які не завжди можуть дозволити собі дорогі корпоративні рішення. Питання безпеки також залишається актуальним, оскільки зберігання завдань, файлів та інших конфіденційних даних у хмарних сервісах може піддавати компанію ризикам витоку інформації. Тому актуальною є розробка програмного забезпечення для адміністрування завдань працівників.

При розробці програмного забезпечення для адміністрування завдань працівників необхідно забезпечити такі функціональні вимоги:

- можливість подання загальної інформації про завдання проєкту у табличному вигляді;
- можливість подання інформації про конкретне завдання проєкту, зокрема, часу його виконання, категорії, стану виконання, пріоритету;
- підтримка можливості фільтрації завдань за категоріями та іншими критеріями;
- можливість додавання та редагування завдань проєкту;
- можливість керування категоріями завдань проєкту;
- підтримка можливості керування ресурсами проєкту;
- підтримка можливості подання у графічному вигляді інформації про прогрес у реалізації проєкту.

### **1.3 Висновки за розділом 1**

Відзначено, що адміністрування завдань працівників передбачає управління робочими процесами, розподілом обов'язків та контролем за

виконанням поставлених задач. Використання спеціалізованого програмного забезпечення дозволяє автоматизувати ці процеси, забезпечуючи зручне планування, моніторинг виконання та аналіз продуктивності.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для адміністрування завдань працівників. Проте деякі програми для адміністрування завдань працівників мають певні недоліки, які можуть ускладнювати їх використання. Однією з головних проблем є складність освоєння, особливо якщо програмне забезпечення має широкий функціонал і вимагає додаткового навчання персоналу. Це може уповільнити впровадження системи в робочий процес та викликати опір серед співробітників. Ще одним недоліком є ризик перевантаження інформацією. Якщо в програмній системі для адміністрування завдань накопичується надто багато завдань, коментарів і сповіщень, працівники можуть відчувати стрес і витрачати багато часу на фільтрацію важливих даних. Деякі програмні засоби для адміністрування завдань працівників мають проблеми з інтеграцією, особливо якщо компанія використовує різні інструменти, які не підтримуються цим програмним забезпеченням. Це може спричинити додаткові витрати на налаштування або перехід на інші сервіси. Крім того, вартість ліцензій на такі програмні системи може бути високою, особливо для малих і середніх підприємств, які не завжди можуть дозволити собі дорогі корпоративні рішення. Питання безпеки також залишається актуальним, оскільки зберігання завдань, файлів та інших конфіденційних даних у хмарних сервісах може піддавати компанію ризикам витоку інформації. Тому актуальною є розробка програмного забезпечення для адміністрування завдань працівників.

Сформульовано функціональні вимоги до програмного забезпечення для адміністрування завдань працівників.

## 2 МАТЕРІАЛИ І МЕТОДИ

### 2.1 Вибір мови програмування

При розробці програмного забезпечення для адміністрування завдань працівників було використано мову програмування C# [15], [16].

C# є сучасною мовою програмування, яка розвивалася як відповідь на потреби розробників у створенні ефективних та надійних програмних продуктів. Вона надає широкі можливості для розробки застосунків різного типу, зокрема для операційної системи Windows, вебзастосунків та програм з інтерактивним інтерфейсом [15], [16].

Попри те, що такі мови, як C++, Visual Basic і Java, пропонують різноманітні підходи до синтаксису та реалізації, функціональність C# залежить від технологічних особливостей її середовища. До ключових оптимізацій, вбудованих у цю мову, належать можливості зручної роботи з графічними інтерфейсами, зберігання та обробки даних, керування подіями, а також підтримка стандартного синтаксису Microsoft .NET. Крім того, C# забезпечує ефективне управління компонентами програм, використання індексаторів для роботи з колекціями, розширені можливості багатопотоковості та зручність при реалізації алгоритмів [15], [16].

Головною перевагою C# у порівнянні з C, C++ та Java є повна відповідність цієї мови принципам об'єктно-орієнтованого програмування, що забезпечує зручність у проєктуванні складних систем. Завдяки міждисциплінарному підходу мова програмування C# дозволяє легко інтегрувати різні апаратні та програмні компоненти, підвищуючи продуктивність, гнучкість та масштабованість застосунків. Високий рівень безпеки коду, розвинена система типізації та механізм обробки помилок значно зменшують ймовірність критичних збоїв. Також передбачено вбудовані засоби для роботи з подіями та винятками, що спрощує управління складними сценаріями роботи програми [15], [16].

Завдяки всім цим характеристикам мова програмування C# є оптимальним вибором для створення програмного забезпечення, призначеного для адміністрування робочих процесів. Його інтеграція з екосистемою Microsoft значно спрощує взаємодію з корпоративними сервісами, такими як Microsoft Azure, Active Directory і SharePoint, що полегшує впровадження та налаштування додатків. Використання технологій .NET дозволяє розробляти як веб-рішення (ASP.NET Core), так і настільні або мобільні застосунки (WPF, WinForms, Xamarin/Maui), що робить систему доступною для всіх користувачів незалежно від їхнього робочого середовища [15], [16].

Структурованість мови програмування C# забезпечує зручність підтримки та розширення програмного забезпечення. Механізми автоматичного керування пам'яттю усувають проблеми з витокami ресурсів, а потужні засоби для розпаралелення процесів дають змогу працювати з великими обсягами даних. Оскільки все більше рішень переходять у хмарні середовища, розвинена підтримка інтеграції з Azure, AWS та іншими сервісами дозволяє легко масштабувати застосунки без значних витрат на оновлення інфраструктури [15], [16].

Враховуючи активний розвиток спільноти, що користується мовою програмування C#, та широку документацію, розробка, тестування і впровадження програмного забезпечення відбувається швидше та ефективніше. Високий рівень безпеки коду, розвинена система типізації та механізм обробки помилок значно зменшують ймовірність критичних збоїв. Також передбачено вбудовані засоби для роботи з подіями та винятками, що спрощує управління складними сценаріями роботи програми [15], [16].

Використання перевірених бібліотек і фреймворків дозволяє мінімізувати витрати на розробку та забезпечити високу якість кінцевого продукту. Саме тому мова програмування C# є одним із найкращих варіантів для створення програмного забезпечення, орієнтованого на управління завданнями в корпоративному середовищі [15], [16].

Обґрунтування вибору мови програмування для розробки програмного забезпечення для адміністрування завдань працівників наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для розробки програмного забезпечення для адміністрування завдань працівників

| Критерій порівняння мов програмування                              | Мова програмування |        |     |
|--------------------------------------------------------------------|--------------------|--------|-----|
|                                                                    | C#                 | Python | PHP |
| Продуктивність                                                     | +                  | +—     | —   |
| Можливості багатопотоковості                                       | +                  | +—     | —   |
| Гнучкість у розробці                                               | +                  | +      | +—  |
| Зручність для створення ПЗ для адміністрування завдань працівників | +                  | —      | —   |
| Хмарна інтеграція                                                  | +                  | +      | +—  |
| Можливість створення веб, мобільних та настільних застосунків      | +                  | +      | +—  |

Отже, для реалізації програмного забезпечення для адміністрування завдань працівників обрано мову програмування C#, яка є об'єктно-орієнтованою мовою, що спрощує підтримку, розширюваність та масштабованість програмного забезпечення. Завдяки строгій типізації та широким можливостям роботи з багатопотоковістю можна ефективно обробляти великі обсяги даних, що важливо для управління завданнями в компаніях з великою кількістю співробітників. Ще однією перевагою є безпечність і стабільність, оскільки C# підтримує механізми керованої пам'яті та автоматичного збирання сміття, що зменшує кількість помилок, пов'язаних із витоками пам'яті та робить програмну систему більш надійною та довговічною.

## **2.2 Вибір середовища розробки для створення програмного забезпечення для адміністрування завдань працівників**

В якості середовища розробки для створення програмного забезпечення для адміністрування завдань працівників було обрано середовище Visual Studio [17], [18].

Середовище розробки Visual Studio вважається одним із найефективніших для розробки програмного забезпечення. У цьому випадку застосовується платформа Microsoft .NET, яка забезпечує високу продуктивність і сумісність із різними технологіями. Оскільки між програмним забезпеченням і базами даних існує тісний взаємозв'язок, необхідно використовувати рішення, що дозволяє ефективно інтегрувати ці компоненти. Ця технологія підтримує можливість запуску Mono на різних операційних системах, окрім Windows. Mono — це багатоплатформене середовище, що дає змогу розробляти складні застосунки та є відкритим аналогом Microsoft .NET Framework [17], [18].

Для автоматизації управління проектами Visual Studio є оптимальним вибором, оскільки містить потужні інструменти для створення, тестування та відлагодження програм. Вбудована підтримка мов програмування C# і .NET дозволяє суттєво прискорити розробку як вебзастосунків, так і програм для настільних комп'ютерів [17], [18].

Ще однією перевагою середовища є інтеграція з хмарними сервісами, що полегшує впровадження корпоративних рішень та забезпечує високу масштабованість. Крім цього, розробка великих проектів значно спрощується завдяки інтеграції із системами керування версіями, що дає можливість організувати ефективну командну роботу [17], [18].

Розширені можливості розробки графічного інтерфейсу, зокрема підтримка WPF і WinForms, сприяють швидкому створенню зручних користувацьких застосунків. Visual Studio також має потужні інструменти для

налагодження та профілювання коду, що дає змогу покращити продуктивність і виявляти проблеми ще на ранніх етапах розробки [17], [18].

Окрім цього, активна підтримка Microsoft і регулярні оновлення гарантують використання найновіших можливостей платформи та доступ до широкої бази навчальних ресурсів. Сукупність цих факторів робить Visual Studio ідеальним середовищем для створення програмного забезпечення для адміністрування завдань працівників [17], [18].

Обґрунтування вибору середовища розробки для створення програмного забезпечення для адміністрування завдань працівників наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору середовища розробки для створення програмного забезпечення для адміністрування завдань працівників

| Критерій порівняння<br>середовищ розробки | Середовища розробки |          |               |
|-------------------------------------------|---------------------|----------|---------------|
|                                           | Visual Studio       | NetBeans | IntelliJ IDEA |
| Зручність інтерфейсу                      | +                   | –        | +             |
| Гнучкість у виборі технологій             | +                   | +–       | +             |
| Можливості командної розробки             | +                   | +–       | +–            |
| Інструменти для відлагодження             | +                   | +–       | +–            |
| Інтеграція з хмарними сервісами           | +                   | –        | +–            |

Таким чином, для створення програмного забезпечення для адміністрування завдань працівників обрано середовище розробки Visual Studio, оскільки воно забезпечує високу продуктивність та сумісність із різними технологіями, пропонує широкий набір інструментів, інтеграцію з

хмарними сервісами, потужні можливості командної роботи та підтримку різних мов програмування, включаючи C#.

### **2.3 Висновки за розділом 2**

Для реалізації програмного забезпечення для адміністрування завдань працівників обрано мову програмування C#, яка є об'єктно-орієнтованою мовою, що спрощує підтримку, розширюваність та масштабованість програмного забезпечення. Завдяки строгій типізації та широким можливостям роботи з багатопотоковістю можна ефективно обробляти великі обсяги даних, що важливо для управління завданнями в компаніях з великою кількістю співробітників. Ще однією перевагою є безпечність і стабільність, оскільки C# підтримує механізми керованої пам'яті та автоматичного збирання сміття, що зменшує кількість помилок, пов'язаних із витоками пам'яті та робить програмну систему більш надійною та довговічною.

Для створення програмного забезпечення для адміністрування завдань працівників обрано середовище розробки Visual Studio, оскільки воно забезпечує високу продуктивність та сумісність із різними технологіями, пропонує широкий набір інструментів, інтеграцію з хмарними сервісами, потужні можливості командної роботи та підтримку різних мов програмування, включаючи C#.

### 3 ОПИС ПРОГРАМИ

#### 3.1 Структура програмного забезпечення для адміністрування завдань працівників

Структуру програмного забезпечення для адміністрування завдань працівників наведено на рис. 3.1.

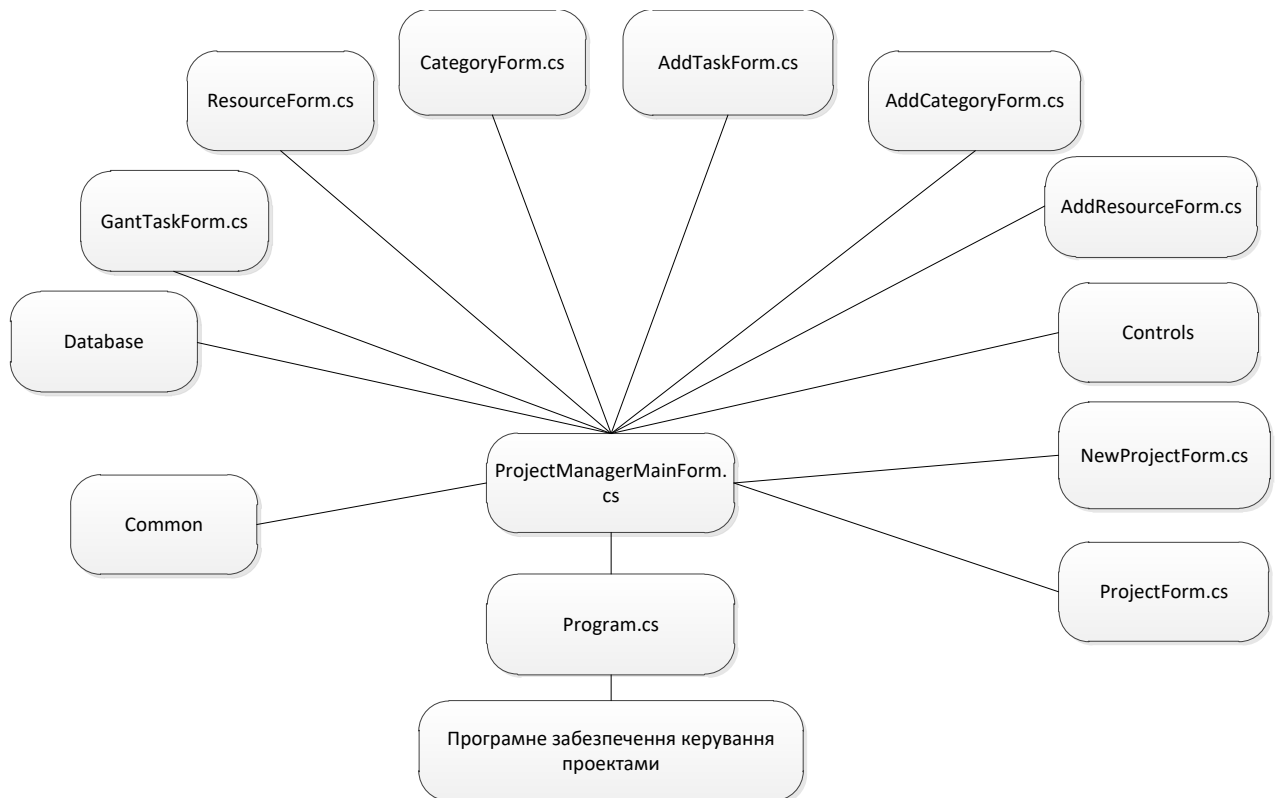


Рисунок 3.1 – Структура програмного забезпечення для адміністрування завдань працівників

Програмне забезпечення для адміністрування завдань працівників побудоване на основі модульної архітектури, що забезпечує ефективну взаємодію між його складовими. Відправною точкою виконання коду є Program, який виконує ініціалізацію програми та передає керування головному компоненту – ProjectManagerMainForm. Цей центральний модуль забезпечує

взаємодію користувача із системою, дозволяючи виконувати різні операції з проектами, завданнями, ресурсами та категоріями.

Функціональність інтерфейсу користувача програмного забезпечення для адміністрування завдань працівників реалізована через низку спеціалізованих модулів. Для створення нових проектів використовується `NewProjectForm`, тоді як `AddCategoryForm`, `AddResourceForm` і `AddProjectForm` відповідають за додавання та редагування відповідних об'єктів системи. Модулі `ProjectForm`, `ResourceForm`, `CategoryForm` та `GantaskForm` забезпечують роботу з переліками завдань, ресурсів і категорій, а також візуалізацію даних у вигляді діаграм Ганта.

Програмне забезпечення для адміністрування завдань працівників також містить групу допоміжних модулів, які спрощують управління її функціональними елементами. Блок `Controls` відповідає за рендеринг інтерфейсу, тоді як `Common` містить загальні функції для роботи з елементами керування, зокрема їхнє заповнення даними. Модулі, що належать до `Database`, забезпечують зв'язок із базою даних, реалізуючи операції збереження та отримання інформації.

Головний інтерфейс програмного забезпечення для адміністрування завдань працівників передбачає виклик різних форм, що відповідають за певні дії. Зокрема, `ProjectManagerMainForm` дозволяє відкривати форми редагування та створення об'єктів (ресурсів, завдань, категорій) та оновлювати їхній вміст. Наприклад, `AddCategoryForm` включає механізми валідації введених даних, отримання параметрів категорії та їх збереження, аналогічні функції виконують `AddResourceForm` та `AddProjectForm` щодо ресурсів та завдань відповідно.

Для полегшення роботи з даними програмне забезпечення для адміністрування завдань працівників використовує набір утиліт, розміщених у модулі `Utility`. Вони дозволяють заповнювати списки елементів керування, генерувати зображення та іконки, а також відображати повідомлення про помилки або підтвердження дій користувача. У модулі `GantViewer` реалізовані

функції роботи з діаграмами Ганта, які дозволяють заповнювати графіки завдань, визначати часові межі проекту та відображати додаткову інформацію у вигляді підказок. Модуль TaskViewer, у свою чергу, відповідає за візуалізацію деталей обраного завдання, включаючи список пов'язаних із ним ресурсів.

Таким чином, структура програмного забезпечення для адміністрування завдань працівників складається з чітко розподілених функціональних блоків, кожен з яких реалізує певний набір завдань, необхідних для ефективного адміністрування проектів, завдань, категорій та ресурсів.

### **3.2 Функціонування програмного забезпечення для адміністрування завдань працівників**

Функціонування програмного забезпечення для адміністрування завдань працівників подамо за допомогою схеми, зображеної на рис. 3.2.

Функціонування програмного забезпечення для управління завданнями працівників починається з відкриття робочого проекту. Якщо користувач програмного забезпечення для управління завданнями працівників раніше працював із певним проектом, система автоматично завантажує його для зручності та безперервності роботи. Такий підхід дає змогу не лише керувати проектами, а й ефективно організовувати планування завдань.

Інтерактивний режим роботи програмного забезпечення для управління завданнями працівників передбачає, що користувач може вільно перемикатися між переглядом списку категорій, ресурсів і завдань, причому початковий екран завжди відкривається зі списком завдань. У кожному з режимів передбачена можливість внесення змін – користувач може редагувати обрані категорії, ресурси або завдання.

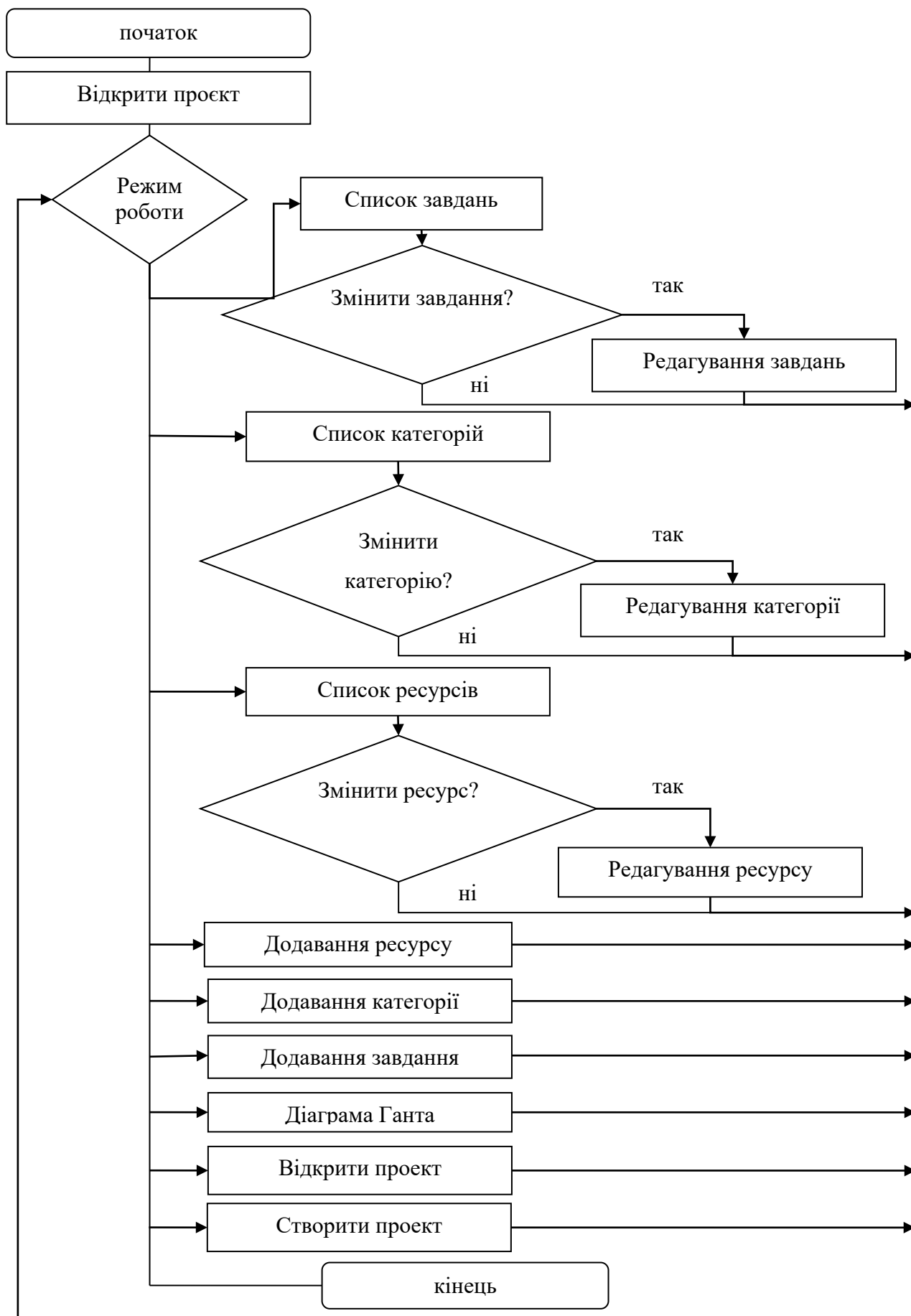


Рисунок 3.2 – Схема функціонування ПЗ для адміністрування завдань

Окрім редагування, передбачений функціонал для додавання нових елементів – створення нових категорій, ресурсів і завдань, які автоматично інтегруються до відповідних списків. Для зручного візуального відображення перебігу робіт доступна діаграма Ганта, що наочно демонструє часові рамки виконання завдань.

Додатково передбачена можливість вибору робочого проекту – користувач може або повернутися до одного з раніше створених, або ініціювати новий, задаючи йому відповідні параметри в режимі налаштувань.

### **3.3 Розробка бази даних програмного забезпечення для адміністрування завдань працівників**

База даних програмного забезпечення для адміністрування завдань працівників призначена для організації та контролю робочих процесів у межах проєктів. Вона забезпечує збереження інформації про проєкти, завдання, ресурси та категорії, а також відстежує зв'язки між ними.

Схему бази даних програмного забезпечення для адміністрування завдань працівників наведено на рис. 3.3.

Основою структури є сутності Project, Task, Resource та Category. Кожен проєкт містить множину завдань, а кожне завдання може належати до певної категорії. Ресурси (співробітники, обладнання) можуть бути залучені до виконання різних завдань, що реалізується через додаткову таблицю для зв'язку ResourceInTask.

У структурі бази даних, що забезпечує роботу програмного забезпечення для адміністрування завдань працівників, визначено кілька ключових сутностей. Основна інформація про проєкти зберігається у відповідній сутності, яка містить унікальний ідентифікатор та назву кожного проєкту.

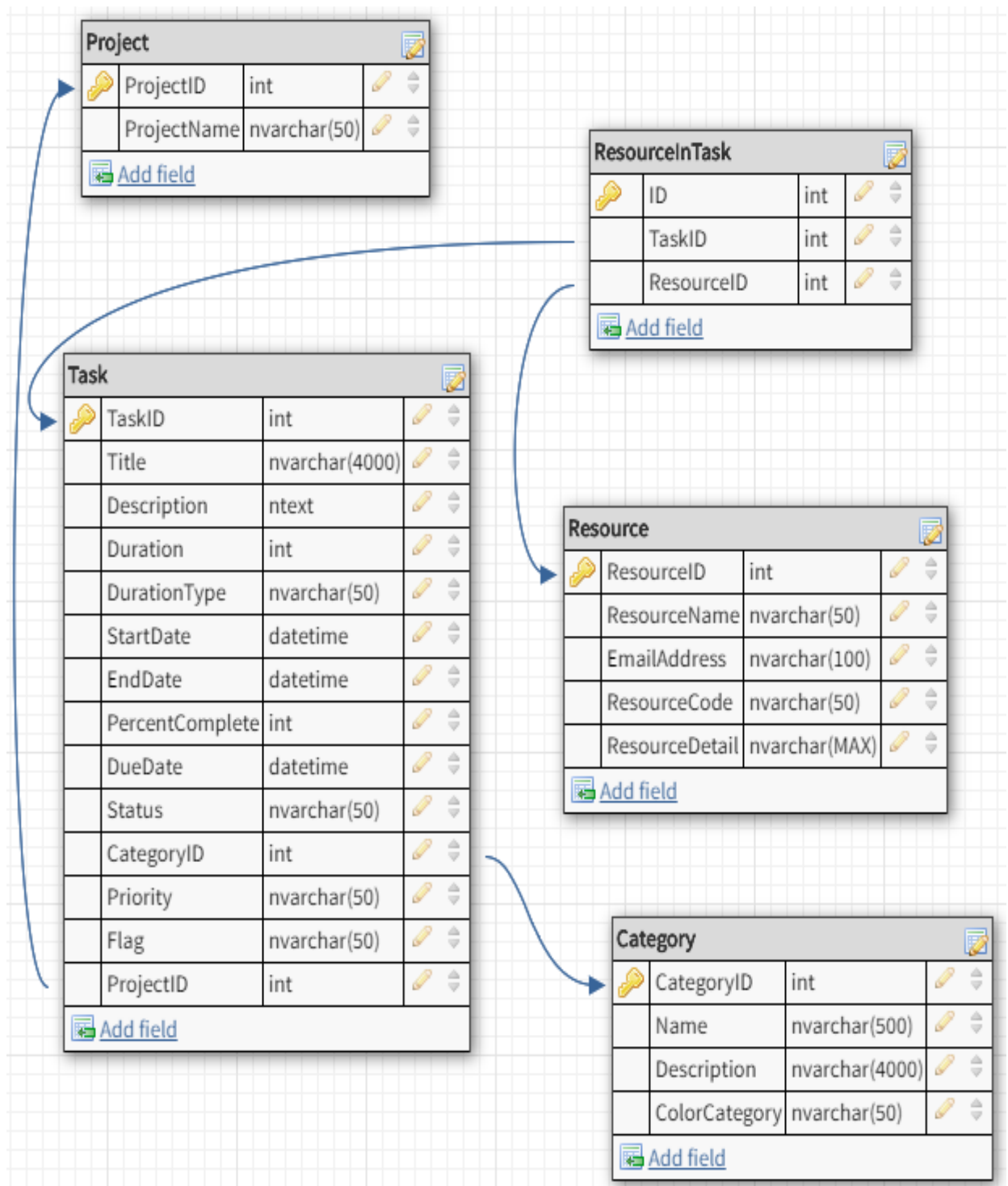


Рисунок 3.3 – Схема бази даних програмного забезпечення для адміністрування завдань працівників

Завдання, які входять до складу проектів, організовані в окрему сутність, де для кожного завдання зазначено його унікальний код, назву, опис, тривалість, дати початку й завершення, статус виконання, пріоритет, а також категорію, до якої воно належить.

Щоб систематизувати завдання, використовується класифікація за категоріями, для яких передбачено окрему сутність із відповідними атрибутами: унікальним ідентифікатором, назвою, описом і кольором для візуального розрізнення. Категорія може містити декілька завдань, але кожне завдання належить лише до однієї категорії або не належить до жодної.

Ресурси, необхідні для виконання завдань, представлені окремою сутністю, що включає унікальний ідентифікатор, назву, електронну адресу, код та додатковий опис. Один і той самий ресурс може бути залучений у кількох завданнях, а кожне завдання, своєю чергою, може використовувати різні ресурси. Для організації цього зв'язку створена проміжна таблиця, яка фіксує відповідність між ресурсами та завданнями.

Взаємозв'язки між сутностями побудовані відповідно до логіки функціонування програмного забезпечення для адміністрування завдань працівників: проект об'єднує в собі множину завдань, ресурси можуть бути залучені до кількох завдань одночасно, а завдання, у свою чергу, можуть містити посилання на ресурси. Така схема дає змогу ефективно організовувати роботу, розподіляти ресурси та відстежувати виконання завдань у межах конкретного проекту.

### **3.4 Проєктування інтерфейсу програмного забезпечення для адміністрування завдань працівників**

Проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для адміністрування завдань працівників виконано з урахуванням вимог технічного завдання. При розробленні програмного забезпечення для адміністрування завдань працівників враховані функціональні вимоги до програми:

- можливість подання загальної інформації про завдання проєкту у табличному вигляді;

- можливість подання інформації про конкретне завдання проєкту, зокрема, часу його виконання, категорії, стану виконання, пріоритету;
- підтримка можливості фільтрації завдань за категоріями та іншими критеріями;
- можливість додавання та редагування завдань проєкту;
- можливість керування категоріями завдань проєкту;
- підтримка можливості керування ресурсами проєкту;
- підтримка можливості подання у графічному вигляді інформації про прогрес у реалізації проєкту.

Інтерфейс головної форми програмного забезпечення для адміністрування завдань працівників наведено на рис. 3.4.

| Стан    | Категорія | Пріоритет | Колір категорії                  | Прапор |
|---------|-----------|-----------|----------------------------------|--------|
| Початок | Кінець    | %         | Назва                            |        |
| ⇒ 01.05 | 14.05     | 100       | Розроблення концепції ПЗ         |        |
| ↑ 05.02 | 24.02     | 100       | Аналіз предметної області        |        |
| ↑ 24.02 | 09.03     | 90        | Розробка та затвердження ТЗ      |        |
| ↑ 10.03 | 30.03     | 0         | Проектування структури ПЗ        |        |
| ↑ 31.03 | 13.04     | 0         | Розробка схеми функціонування ПЗ |        |
| ↑ 14.04 | 27.04     | 0         | Розробка коду                    |        |
| ↑ 28.04 | 04.05     | 0         | Тестування і відлагодження       |        |
| ↑ 05.05 | 11.05     | 0         | Написання пояснювальної записки  |        |
| ↑ 12.05 | 14.05     | 0         | Розробка графічного матеріалу    |        |
| ↑ 18.05 | 18.05     | 0         | Захист                           |        |

### Аналіз предметної області

Провести аналіз існуючих прототипів, сформулювати цілі та задачі проєкту, виділити базові сутності та взаємозв'язки між ними.

|             |         |            |           |
|-------------|---------|------------|-----------|
| Початок     | 05.02   | Завершення | 24.02     |
| Виконання   | 22.02   | Тривалість | 18        |
| % виконання | 100%    | Стан       | Завершено |
| Пріоритет   | Високий | Категорія  | Диплом    |
| Ресурс      |         |            |           |

Рисунок 3.4 – Інтерфейс головної форми програмного забезпечення для адміністрування завдань працівників

Форму створення нового завдання у проєкті наведено на рис. 3.5.

Додати/Редагувати завдання

Назва

Опис

Ресурс ☐ Програміст ☐ Комп'ютер

Дата початку

Дата виконання ☒ 15.05

% завершення

Пріоритет

Колір категорії

Дата закінчення

Тривалість

Стан

Категорія

Прапор

Рисунок 3.5 – Форма створення нового завдання у проєкті

Вікно керування категоріями наведено на рис. 3.6.

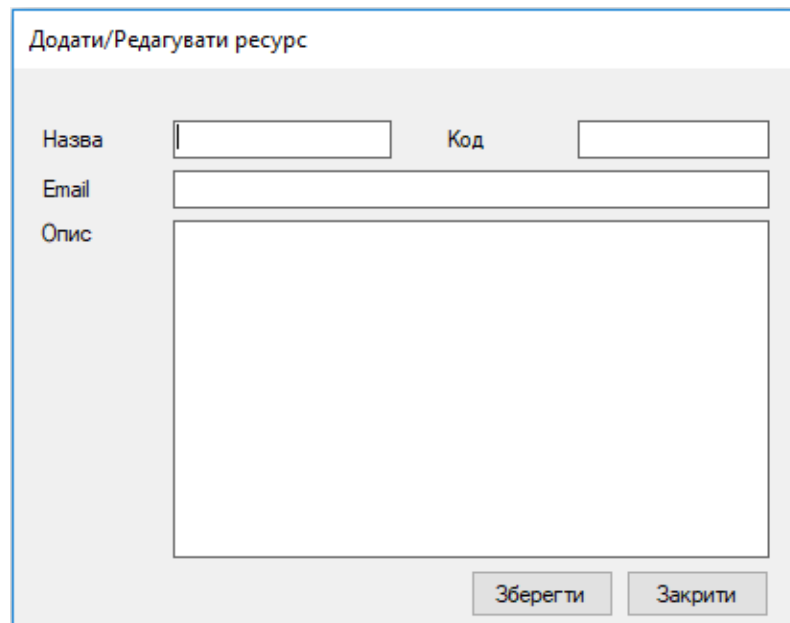
Файл Керування

Завдання Діаграма Ресурси Категорії Налаштування Нове завдання Новий ресурс Нова категорія

| Назва  | Опис                |                                     |
|--------|---------------------|-------------------------------------|
| Диплом |                     | <input checked="" type="checkbox"/> |
| Проект | Реалізація програми | <input type="checkbox"/>            |

Рисунок 3.6 – Вікно керування категоріями

Форму додавання нового ресурсу до проєкту наведено на рис. 3.7.



Додати/Редагувати ресурс

Назва  Код

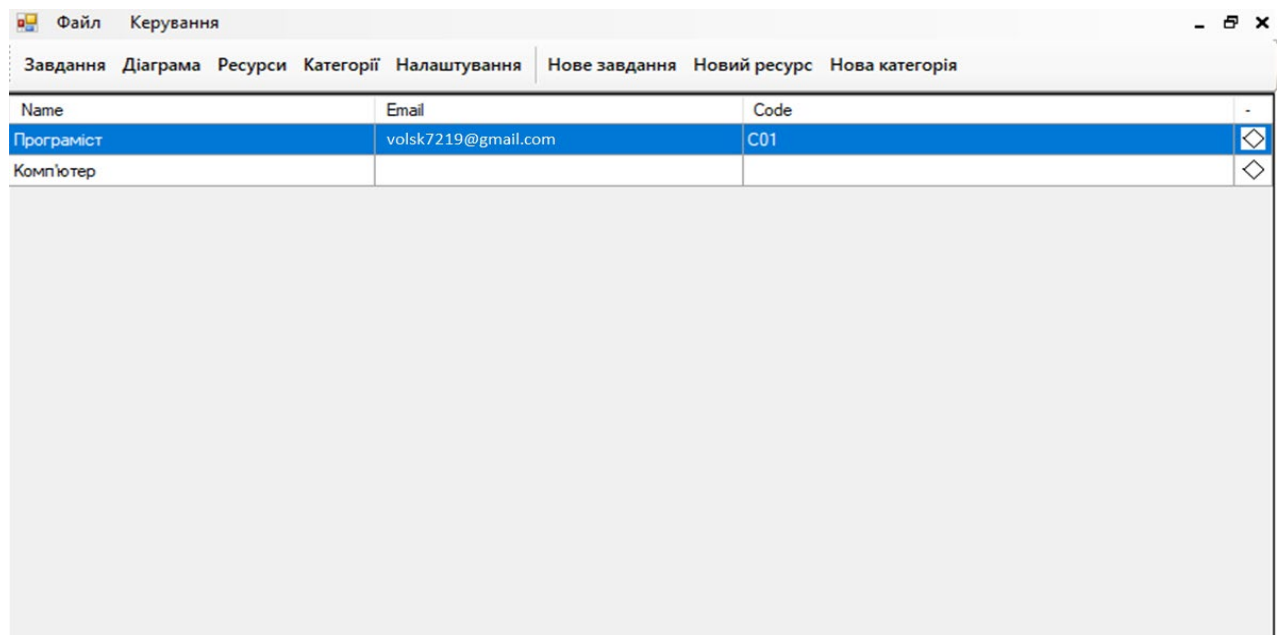
Email

Опис

Зберегти Закрити

Рисунок 3.7 – Форма додавання нового ресурсу до проєкту

Форму з інформацією про ресурси поточного проєкту наведено на рис. 3.8.



|                                                                                            |                     |      |   |   |   |
|--------------------------------------------------------------------------------------------|---------------------|------|---|---|---|
| Файл Керування                                                                             |                     |      | - | □ | × |
| Завдання Діаграма Ресурси Категорії Налаштування Нове завдання Новий ресурс Нова категорія |                     |      |   |   |   |
| Name                                                                                       | Email               | Code |   |   |   |
| Програміст                                                                                 | volsk7219@gmail.com | C01  |   |   |   |
| Комп'ютер                                                                                  |                     |      |   |   |   |

Рисунок 3.8 – Форму з інформацією про ресурси поточного проєкту

Форму з графічним поданням прогресу у реалізації проєкту наведено на рис. 3.9.

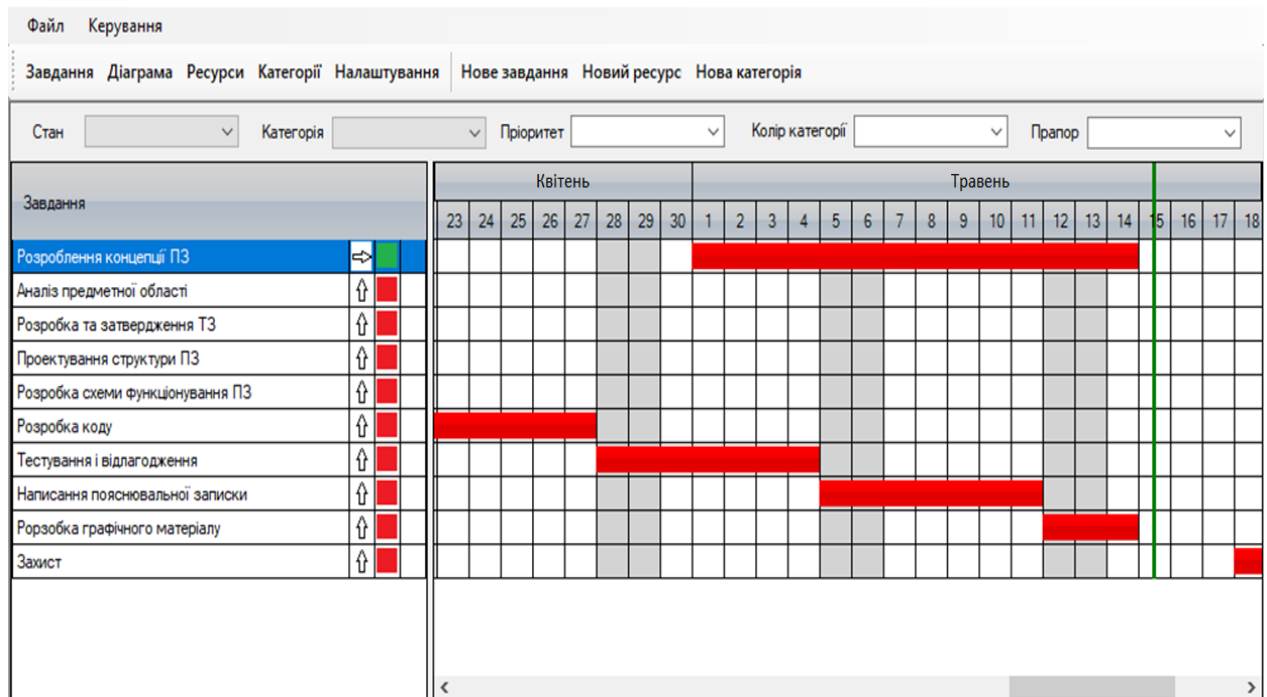


Рисунок 3.9 – Форма з графічним поданням прогресу у реалізації проєкту

### 3.5 Висновки за розділом 3

Розроблено програмне забезпечення для адміністрування завдань працівників.

Запропоновано структуру програмного забезпечення для адміністрування завдань працівників. Визначено, що програмне забезпечення для адміністрування завдань працівників побудоване на основі модульної архітектури та складається з чітко розподілених функціональних блоків, кожен з яких реалізує певний набір завдань, необхідних для ефективного адміністрування проєктів, завдань, категорій та ресурсів, що забезпечує ефективну взаємодію між його складовими.

Розроблено базу даних програмного забезпечення для адміністрування завдань працівників призначена для організації та контролю робочих процесів у межах проєктів. Вона забезпечує збереження інформації про проєкти,

завдання, ресурси та категорії, а також відстежує зв'язки між ними. Основою структури є сутності Project, Task, Resource та Category. Кожен проєкт містить множину завдань, а кожне завдання може належати до певної категорії. Ресурси (співробітники, обладнання) можуть бути залучені до виконання різних завдань, що реалізується через додаткову таблицю для зв'язку ResourceInTask.

Описано функціонування програмного забезпечення для адміністрування завдань працівників.

Описано особливості реалізації програмного забезпечення для адміністрування завдань працівників. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для адміністрування завдань працівників.

## **4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ**

### **4.1 Призначення й умови застосування програми**

Програма призначена для адміністрування завдань працівників. Програма написана на мові C#, яка є об'єктно-орієнтованою мовою, що спрощує підтримку, розширюваність та масштабованість програмного забезпечення. В якості середовища розробки для створення програмного забезпечення для адміністрування завдань працівників обрано Visual Studio, оскільки воно забезпечує високу продуктивність та сумісність із різними технологіями, пропонує широкий набір інструментів, інтеграцію з хмарними сервісами, потужні можливості командної роботи та підтримку різних мов програмування, включаючи C#.

Програмне забезпечення для адміністрування завдань працівників забезпечує виконання таких функцій:

- можливість подання загальної інформації про завдання проєкту у табличному вигляді;
- можливість подання інформації про конкретне завдання проєкту, зокрема, часу його виконання, категорії, стану виконання, пріоритету;
- підтримка можливості фільтрації завдань за категоріями та іншими критеріями;
- можливість додавання та редагування завдань проєкту;
- можливість керування категоріями завдань проєкту;
- підтримка можливості керування ресурсами проєкту;
- підтримка можливості подання у графічному вигляді інформації про прогрес у реалізації проєкту.

Для роботи програмного забезпечення для адміністрування завдань працівників потрібні такі технічні та програмні ресурси, як серверне середовище для зберігання та обробки даних, достатній обсяг оперативної пам'яті та процесорної потужності для швидкого виконання операцій, а також

стабільне мережеве з'єднання для синхронізації даних між користувачами. Програмне середовище включає систему управління базами даних для збереження інформації про завдання, ресурси та користувачів, веб- або настільний інтерфейс для взаємодії з системою, а також механізми автентифікації та авторизації для контролю доступу. Забезпечення сумісності з операційними системами та інтеграція з іншими програмними продуктами сприяє зручному використанню та гнучкості платформи.

Функціональні характеристики розробленого програмного забезпечення для адміністрування завдань працівників наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення для адміністрування завдань працівників наведено у додатку Б.

#### **4.2 Характеристики програми для адміністрування завдань працівників**

Програмне забезпечення для адміністрування завдань працівників побудовано на основі модульної архітектури та складається з чітко розподілених функціональних блоків, кожен з яких реалізує певний набір завдань, необхідних для ефективного адміністрування проєктів, завдань, категорій та ресурсів, що забезпечує ефективну взаємодію між його складовими. Функціонування програмного забезпечення для управління завданнями працівників починається з відкриття робочого проєкту. Якщо користувач програмного забезпечення для управління завданнями працівників раніше працював із певним проєктом, система автоматично завантажує його для зручності та безперервності роботи. Такий підхід дає змогу не лише керувати проєктами, а й ефективно організовувати планування завдань.

Програмне забезпечення для адміністрування завдань працівників призначене для ефективного планування, контролю та управління робочими процесами в межах проєктів. Передбачено можливість створення, редагування та відстеження завдань, а також їхньої класифікації за різними параметрами.

Реалізовано підтримку роботи з категоріями, що дозволяє впорядковувати завдання відповідно до їхнього призначення.

Програмне забезпечення для адміністрування завдань працівників забезпечує розподіл ресурсів та контроль їхнього залучення до виконання завдань. Передбачено функціонал для моніторингу прогресу виконання роботи, що включає статуси, пріоритети та рівень завершеності завдань. Передбачено можливість гнучкого налаштування інтерфейсу, що дає змогу адаптувати програму під конкретні потреби користувачів.

Використовується інтерактивний підхід до управління проектами, що дає змогу швидко змінювати параметри завдань, переглядати їхню взаємодію з іншими елементами системи та візуалізувати інформацію у вигляді діаграми Ганта. Забезпечено можливість імпорту та експорту даних, що спрощує інтеграцію з іншими програмними продуктами.

Програмне забезпечення для адміністрування завдань працівників дозволяє зберігати та обробляти інформацію про проекти, задачі, ресурси та їхній взаємозв'язок. Впроваджено механізми для ефективного контролю за виконанням завдань, включаючи нагадування та автоматизовані повідомлення. Інформація структурована таким чином, щоб забезпечити швидкий доступ до необхідних даних, зручний пошук і можливість аналізу виконаної роботи.

### **4.3 Інструкція по експлуатації програми**

#### **4.3.1 Звернення до програми**

Для запуску програмного забезпечення для адміністрування завдань працівників необхідно запустити файл ProjectManager.exe.

#### **4.3.2 Вхідні й вихідні дані**

Вхідними даними до програмного забезпечення для адміністрування завдань працівників є інформація про проект, завдання (назва, опис, тривалість

та ін.). Вихідними даними програмного забезпечення для адміністрування завдань працівників є створені у базі даних записи про проєкти завдання, виконавців, діаграми Ганта та ін.

### 4.3.3 Повідомлення

Користувач програмного забезпечення для адміністрування завдань працівників може отримати такі повідомлення: повідомлення про некоректно введені дані (наприклад, у числові поля вводяться текстові дані), повідомлення про неможливість доступу до бази даних та ін.

## 4.4 Виконання програмного забезпечення для адміністрування завдань працівників

Після запуску програмного забезпечення для адміністрування завдань працівників користувачу відображається головна форма (рис. 4.1), де відображено перелік завдань поточного проєкту.

| Початок | Кінець | %   | Назва                            |
|---------|--------|-----|----------------------------------|
| 01.05   | 14.05  | 100 | Розроблення концепції ПЗ         |
| 05.02   | 24.02  | 100 | Аналіз предметної області        |
| 24.02   | 09.03  | 90  | Розробка та затвердження ТЗ      |
| 10.03   | 30.03  | 0   | Проектування структури ПЗ        |
| 31.03   | 13.04  | 0   | Розробка схеми функціонування ПЗ |
| 14.04   | 27.04  | 0   | Розробка коду                    |
| 28.04   | 04.05  | 0   | Тестування і відлагодження       |
| 05.05   | 11.05  | 0   | Написання пояснювальної записки  |
| 12.05   | 14.05  | 0   | Розробка графічного матеріалу    |
| 18.05   | 18.05  | 0   | Захист                           |

### Аналіз предметної області

Провести аналіз існуючих прототипів, сформулювати цілі та задачі проєкту, виділити базові сутності та взаємозв'язки між ними.

|             |         |            |           |
|-------------|---------|------------|-----------|
| Початок     | 05.02   | Завершення | 24.02     |
| Виконання   | 22.02   | Тривалість | 18        |
| % виконання | 100%    | Стан       | Завершено |
| Пріоритет   | Високий | Категорія  | Диплом    |
| Ресурс      |         |            |           |

Рисунок 4.1 – Інтерфейс головної форми програмного забезпечення для адміністрування завдань працівників

За допомогою переходу між вкладками головної форми надається можливість ознайомлення з завданнями проєкту, роботою з діаграмами, ресурсами, категоріями та налаштуваннями.

Користувач має можливість використовувати фільтри для відображення переліку завдань за пріоритетом, станом, категорією та ін. (рис. 4.2).

Файл Керування

Завдання Діаграма Ресурси Категорії Налаштування Нове завдання Новий ресурс Нова категорія

Стан  Категорія  Диплом Пріоритет  Колір категорії  Прапор

|   | Початок | Кінець | %   | Назва                            |   |   |
|---|---------|--------|-----|----------------------------------|---|---|
| ↑ | 05.02   | 24.02  | 100 | Аналіз предметної області        | ■ | ◇ |
| ↑ | 24.02   | 09.03  | 90  | Розробка та затвердження ТЗ      | ■ | ◇ |
| ↑ | 10.03   | 30.03  | 0   | Проектування структури ПЗ        | ■ | ◇ |
| ↑ | 31.03   | 13.04  | 0   | Розробка схеми функціонування ПЗ | ■ | ◇ |
| ↑ | 14.04   | 27.04  | 0   | Розробка коду                    | ■ | ◇ |
| ↑ | 28.04   | 04.05  | 0   | Тестування і відлагодження       | ■ | ◇ |
| ↑ | 05.05   | 11.05  | 0   | Написання пояснювальної записки  | ■ | ◇ |
| ↑ | 12.05   | 14.05  | 0   | Розробка графічного матеріалу    | ■ | ◇ |
| ↑ | 18.05   | 18.05  | 0   | Захист                           | ■ | ◇ |

**Аналіз предметної області**

Провести аналіз існуючих прототипів, сформулювати цілі та задачі проєкту, виділити базові сутності та взаємозв'язки між ними.

|             |         |            |           |
|-------------|---------|------------|-----------|
| Початок     | 05.02   | Завершення | 24.02     |
| Виконання   | 22.02   | Тривалість | 18        |
| % виконання | 100%    | Стан       | Завершено |
| Пріоритет   | Високий | Категорія  | Диплом    |
| Ресурс      |         |            |           |

Рисунок 4.2 – Результати фільтрації завдань проєкту

Для створення нового завдання необхідно натиснути кнопку Нове завдання та у вікні, що з'явилося (рис. 4.3), ввести назву та опис завдання, а також його тривалість, ресурси та іншу інформацію.

У випадку переходу в меню Категорії відкривається вікно (рис. 4.4), де користувач може змінювати інформацію про категорії завдань проєкту.

Додати/Редагувати завдання

Назва

Опис

Ресурс 

☐ Програміст  
☐ Комп'ютер

Дата початку 
 Дата виконання ☒ 15.05

Дата закінчення 
 Тривалість

% завершення 
 Стан

Пріоритет 
 Категорія

Колір категорії 
 Прапор

Рисунок 4.3 – Створення нового завдання у проєкті

| Файл Керування                                                                                                  |                     |   |
|-----------------------------------------------------------------------------------------------------------------|---------------------|---|
| Завдання    Діаграма    Ресурси    Категорії    Налаштування    Нове завдання    Новий ресурс    Нова категорія |                     |   |
| Назва                                                                                                           | Опис                |   |
| Диплом                                                                                                          |                     | ◆ |
| Проект                                                                                                          | Реалізація програми | ◆ |

Рисунок 4.4 – Вікно керування категоріями

Для додавання нового ресурсу необхідно натиснути кнопку Новий ресурс на головній формі та ввести відповідну інформацію (рис. 4.5). Аналогічно можна додати нову категорію.

Додати/Редагувати ресурс

Назва  Код

Email

Опис

Зберегти Закрити

Рисунок 4.5 – Додавання нового ресурсу до проєкту

У випадку переходу в меню Ресурси відкривається вікно (рис. 4.6), де користувач може змінювати інформацію про ресурси поточного проєкту.

| Name       | Email               | Code |
|------------|---------------------|------|
| Програміст | volsk7219@gmail.com | C01  |
| Комп'ютер  |                     |      |

Рисунок 4.6 – Сторінка ресурсів проєкту

При натисканні на кнопку Діаграма відображається діаграма Ганта (рис. 4.7), що у графічному вигляді подає прогрес у реалізації проєкту.

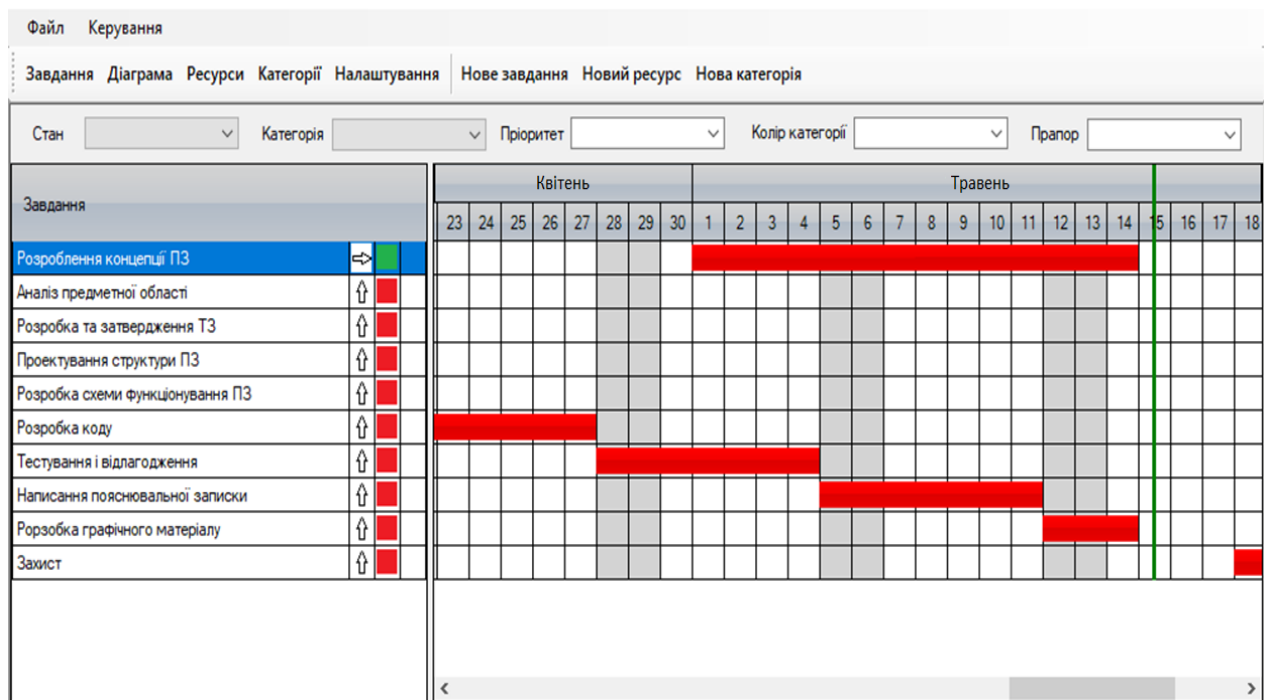


Рисунок 4.7 – Діаграма Ганта з прогресом у реалізації проєкту

#### 4.5 Тестування програмного забезпечення для адміністрування завдань працівників

Виконано тестування програмного забезпечення для адміністрування завдань працівників.

Виявлено, що програма для адміністрування завдань працівників функціонує правильно та злагоджено. Вона реалізує всі функціональні вимоги та успішно виконує свою основну задачу підтримки процесу адміністрування завдань.

Розроблене програмне забезпечення дозволяє забезпечувати автоматизацію процесів, пов'язаних з підтримкою процесу адміністрування завдань.

#### 4.6 Висновки за розділом 4

Описано програмне забезпечення для адміністрування завдань працівників. Виконано тестування розробленого програмного забезпечення

для адміністрування завдань працівників. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою процесу адміністрування завдань.

## ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки програмного забезпечення для підтримки проєктної діяльності.

Відзначено, що адміністрування завдань працівників передбачає управління робочими процесами, розподілом обов'язків та контролем за виконанням поставлених задач. Використання спеціалізованого програмного забезпечення дозволяє автоматизувати ці процеси, забезпечуючи зручне планування, моніторинг виконання та аналіз продуктивності.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для адміністрування завдань працівників. Проте деякі програми для адміністрування завдань працівників мають певні недоліки, які можуть ускладнювати їх використання. Однією з головних проблем є складність освоєння, особливо якщо програмне забезпечення має широкий функціонал і вимагає додаткового навчання персоналу. Це може уповільнити впровадження системи в робочий процес та викликати опір серед співробітників. Ще одним недоліком є ризик перевантаження інформацією. Якщо в програмній системі для адміністрування завдань накопичується надто багато завдань, коментарів і сповіщень, працівники можуть відчувати стрес і витрачати багато часу на фільтрацію важливих даних. Деякі програмні засоби для адміністрування завдань працівників мають проблеми з інтеграцією, особливо якщо компанія використовує різні інструменти, які не підтримуються цим програмним забезпеченням. Це може спричинити додаткові витрати на налаштування або перехід на інші сервіси. Крім того, вартість ліцензій на такі програмні системи може бути високою, особливо для малих і середніх підприємств, які не завжди можуть дозволити собі дорогі корпоративні рішення. Питання безпеки також залишається актуальним, оскільки зберігання завдань, файлів та інших конфіденційних даних у хмарних сервісах може піддавати компанію ризикам

витоку інформації. Тому актуальною є розробка програмного забезпечення для адміністрування завдань працівників.

Сформульовано функціональні вимоги до програмного забезпечення для адміністрування завдань працівників.

Для реалізації програмного забезпечення для адміністрування завдань працівників обрано мову програмування C#, яка є об'єктно-орієнтованою мовою, що спрощує підтримку, розширюваність та масштабованість програмного забезпечення. Завдяки строгій типізації та широким можливостям роботи з багатопотоковістю можна ефективно обробляти великі обсяги даних, що важливо для управління завданнями в компаніях з великою кількістю співробітників. Ще однією перевагою є безпечність і стабільність, оскільки C# підтримує механізми керованої пам'яті та автоматичного збирання сміття, що зменшує кількість помилок, пов'язаних із витоками пам'яті та робить програмну систему більш надійною та довговічною.

Для створення програмного забезпечення для адміністрування завдань працівників обрано середовище розробки Visual Studio, оскільки воно забезпечує високу продуктивність та сумісність із різними технологіями, пропонує широкий набір інструментів, інтеграцію з хмарними сервісами, потужні можливості командної роботи та підтримку різних мов програмування, включаючи C#.

Розроблено програмне забезпечення для адміністрування завдань працівників.

Запропоновано структуру програмного забезпечення для адміністрування завдань працівників. Визначено, що програмне забезпечення для адміністрування завдань працівників побудоване на основі модульної архітектури та складається з чітко розподілених функціональних блоків, кожен з яких реалізує певний набір завдань, необхідних для ефективного адміністрування проєктів, завдань, категорій та ресурсів, що забезпечує ефективну взаємодію між його складовими.

Розроблено базу даних програмного забезпечення для адміністрування завдань працівників призначена для організації та контролю робочих процесів у межах проєктів. Вона забезпечує збереження інформації про проєкти, завдання, ресурси та категорії, а також відстежує зв'язки між ними. Основою структури є сутності Project, Task, Resource та Category. Кожен проєкт містить множину завдань, а кожне завдання може належати до певної категорії. Ресурси (співробітники, обладнання) можуть бути залучені до виконання різних завдань, що реалізується через додаткову таблицю для зв'язку ResourceInTask.

Описано функціонування програмного забезпечення для адміністрування завдань працівників.

Описано особливості реалізації програмного забезпечення для адміністрування завдань працівників. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для адміністрування завдань працівників.

Виконано тестування розробленого програмного забезпечення для адміністрування завдань працівників. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою процесу адміністрування завдань.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Task Management Simplified: A Beginner's Guide [Electronic resource]. – Access mode: <https://www.peoplebeam.co/blog/task-management-beginners-guide>.
2. What is Task Management?: A Comprehensive Overview [Electronic resource]. – Access mode: <https://www.simplilearn.com/what-is-task-management-article>.
3. Employee Task Management System Web Project [Electronic resource]. – Access mode: <https://1000projects.org/employee-task-management-system-web-project-using-php-and-mysql.html>.
4. What is task management? A comprehensive guide [Electronic resource]. – Access mode: <https://www.proofhub.com/articles/what-is-task-management>.
5. Best Task Management Software [Electronic resource]. – Access mode: <https://software.fish/project-management-software/task-management-software>.
6. What is Task Management? A Beginners Guide [Electronic resource]. – Access mode: <https://www.profit.co/blog/task-management/what-is-task-management-a-beginners-guide/>.
7. What Is Task Management [Electronic resource]. – Access mode: <https://www.coursera.org/articles/task-management>.
8. Effortlessly track time spent on your tasks and projects [Electronic resource]. – Access mode: <https://www.monitask.com/en/Home/Task-Tracking-Software>.
9. Best Task Management Software [Electronic resource]. – Access mode: <https://www.cloudwards.net/best-task-management-software/>.
10. Best task management app of 2025 [Electronic resource]. – Access mode: <https://www.techradar.com/best/best-task-management-apps-of-year>.
11. I Tried 10 Best Free Task Management Software: My Review [Electronic resource]. – Access mode: <https://learn.g2.com/best-free-task-management-software>.

12. TickTick [Electronic resource]. – Access mode: <https://ticktick.com/>.
13. HubSpot [Electronic resource]. – Access mode: <https://www.hubspot.com/>.
14. Wrike [Electronic resource]. – Access mode: <https://www.wrike.com/vay/>.
15. C# Tutorial [Electronic resource]. – Access mode: <https://www.csharptutorial.net/>.
16. C# documentation [Electronic resource]. – Access mode: <https://learn.microsoft.com/en-us/dotnet/csharp/>.
17. How to get started with Visual Studio [Electronic resource]. – Access mode: <https://www.thewindowsclub.com/how-to-get-started-with-visual-studio>.
18. Meet the Visual Studio family [Electronic resource]. – Access mode: <https://visualstudio.microsoft.com/>.

**ДОДАТОК А**  
**Технічне завдання**

## **Вступ**

Програмне забезпечення може використовуватися для адміністрування завдань працівників.

### **A.1 Підстава для розробки**

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Програмне забезпечення для адміністрування завдань працівників», затверджене наказом Національного університету «Запорізька політехніка» № 209 від 28 квітня 2025 р.

### **A.2 Призначення розробки**

Програмний продукт призначений для адміністрування завдань працівників.

### **A.3 Основні вимоги до програми, що розробляється**

#### **A.3.1 Вимоги до функціональних характеристик**

Програмне забезпечення для адміністрування завдань працівників забезпечує виконання таких функцій:

- можливість подання загальної інформації про завдання проєкту у табличному вигляді;
- можливість подання інформації про конкретне завдання проєкту, зокрема, часу його виконання, категорії, стану виконання, пріоритету;
- підтримка можливості фільтрації завдань за категоріями та іншими критеріями;
- можливість додавання та редагування завдань проєкту;
- можливість керування категоріями завдань проєкту;

- підтримка можливості керування ресурсами проєкту;
- підтримка можливості подання у графічному вигляді інформації про прогрес у реалізації проєкту.

### **A.3.2 Вимоги до інтерфейсу програми**

Інтерфейс програмного забезпечення для адміністрування завдань працівників повинен бути зручним для користувачів. Повинна бути забезпечена можливість роботи в візуальному режимі.

### **A.3.3 Вимоги до надійності**

Програмне забезпечення для адміністрування завдань працівників повинно забезпечити надійне функціонування.

### **A.3.4 Умови експлуатації**

Для експлуатації програмного забезпечення для адміністрування завдань працівників необхідна наявність персонального комп'ютера.

### **A.3.5 Вимоги до складу та параметрів технічний засобів**

Для роботи програмного забезпечення для адміністрування завдань працівників потрібні такі технічні та програмні ресурси, як серверне середовище для зберігання та обробки даних, достатній обсяг оперативної пам'яті та процесорної потужності для швидкого виконання операцій, а також стабільне мережеве з'єднання для синхронізації даних між користувачами.

### **A.3.6 Вимоги до маркування і пакування**

Програма для адміністрування завдань працівників може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва програми – «Програмне забезпечення для адміністрування завдань працівників».

### **A.4 Вхідні дані до роботи**

Вхідними даними до програмного забезпечення для адміністрування завдань працівників є інформація про проєкт, завдання (назва, опис, тривалість та ін.).

Вихідними даними програмного забезпечення для адміністрування завдань працівників є створені у базі даних записи про проєкти завдання, виконавців, діаграми Ганта та ін.

**ДОДАТОК Б**  
**Фрагмент тексту програми**

```

namespace ProjectManager
{
    public partial class PrjctForm : Form
    {
        public PrjctForm()
        {
            InlzCmpnt();
        }

        public void GetTskSeq()
        {
            PrjctManagerEntities tm = new PrjctManagerEntities();
            string knd = GetKndition();
            gvAllTsk.AutoGenerateColumns = false;
            if (string.IsNullOrEmpty(knd))
            {
                gvAllTsk.DataSource = tm.Tsk.Include("Ktgr");
            }
            else
            {
                gvAllTsk.DataSource =
tm.Tsk.Include("Ktgr").Include("Rsltrce").Where(knd);
            }
            RprsntRyadk();
            if (gvAllTsk.Ryadks.Count > 0)
            {
                gvAllTsk.Ryadks[0].Selected = true;
                GetTskDetail();
            }
        }

        private void RprsntRyadk()
        {
            foreach (DataGrdVRyadk gr in gvAllTsk.Ryadks)
            {
                Tsk t = gr.DataBoundItem as Tsk;
                String stv = t.Stts;
                if (!string.IsNullOrEmpty(stv))
                {
                    TskStts st =
(TskStts)Enhm.Parse(typeof(TskStts), stv);
                    if (st == TskStts.Completed)
                    {
                        gr.DefaultCellStyle.Font = new
Font(gvAllTsk.RyadksDefaultCellStyle.Font, FontStyle.Strikeout);
                    }
                }
            }
        }

        private string GetKndition()
        {
            string knd = "";
            if (prCB.SelectedIndex > 0)
            {
                knd = CncatKnd(knd, "irt.Prprt = '" + prCB.Text +
"", string.Empty);
            }
            if (sbCB.SelectedIndex > 0)

```

```

        {
            knd = CncatKnd(knd, "irt.Stts = '" + sbCB.Text +
""", "AND");
        }
        if (cdCB.SelectedIndex > 0)
        {
            knd = CncatKnd(knd, "irt.Ktgr.KtgrID = " +
cdCB.SelectedValue.ToString(), "AND");
        }
        if (cnCB.SelectedIndex > 0)
        {
            knd = CncatKnd(knd, "irt.ColorKtgr = '" +
cnCB.Text + "'", "AND");
        }
        if (cmbFlag.SelectedIndex > 0)
        {
            knd = CncatKnd(knd, "irt.Flag = '" + cmbFlag.Text
+ "'", "AND");
        }
        return knd;
    }

    private string KnditionCncat(string knd, string nc,
string oprtrs)
    { if (knd.Length > 0)
        { return knd + " " + oprtrs + " " + nc;
        }
        else
        { return nc;
        }
    }

    private void BndComb()
    {
        Utlt.TskPrioCtgBndr(prCB, true);
        Utlt.TskSttsBinder(sbCB, true);
        Utlt.TskColorBinder(cnCB, true);
        Utlt.TFlgCtgBinder(cmbFlag, true);
        CtgBndr();
    }

    private void CtgBndr()
    { PrjctManagerEntities tm = new PrjctManagerEntities();
      List<Ktgr> kl = tm.Ktgr.ToList();
      Ktgr nc = new Ktgr();
      kl.Insert(0, nc);
      cdCB.DisplayMember = "Name";
      cdCB.ValueMember = "KtgrID";

```

```

        cdCB.DataSource = kl;
    }

    private void ShowAddPrjctForm(int tnhm)
    { PrjctManagerMainForm prf = MdiParent as
PrjctManagerMainForm;
        if (prf != null)
        { prf.OpenAddNewPrjctForm(tnhm);
        }
    }

    private void GetTskDetail()
    { if (gvAllTsks.SelectedRyadks != null &&
gvAllTsks.SelectedRyadks.Count > 0)
        { Tsk t = gvAllTsks.SelectedRyadks[0].DataBoundItem
as Tsk;

            TskViewer1.SetTskDetail(t);
        }
    }

    private void DetPrprt(DataGridViewCellFormattingEventArgs e)
    { String stv = e.Value as string;
        if (stv == null) return;
        DataGridViewCell t = gvAllTsks[e.ColumnIndex,
e.RyadkIndex];
        t.ToolTipText = stv;

        switch (stv.ToLower())
        { case "високий":
            e.Value = Properties.Rsltrces.High_Prprt;
            break;
          case "низький":
            e.Value = Properties.Rsltrces.Low_Prprt;
            break;
          case "середній":
            e.Value = Properties.Rsltrces.Medium_Prprt;
            break;
          default:
            e.Value = Properties.Rsltrces.Empty;
            break;
        }
    }

    private void DetColor(DataGridViewCellFormattingEventArgs e)
    { String stv = e.Value as string;
        if (stv == null) return;
        DataGridViewCell t = gvAllTsks[e.ColumnIndex,
e.RyadkIndex];

```

```

        t.ToolTipText = stv;
        if (!string.IsNullOrEmpty(stv))
        { TskCtEnm ktgr =
(TskCtEnm)Enhm.Parse(typeof(TskCtEnm), stv);
            switch (ktgr)
            { case TskCtEnm.Blue:
                e.Value = Properties.Rsltrces.BCat;
                break;
                case TskCtEnm.Green:
                e.Value = Properties.Rsltrces.GCat;
                break;
                case TskCtEnm.Orange:
                e.Value = Properties.Rsltrces.OCat;
                break;
                case TskCtEnm.Purple:
                e.Value = Properties.Rsltrces.PCat;
                break;
                case TskCtEnm.Red:
                e.Value = Properties.Rsltrces.RCat;
                break;
                case TskCtEnm.Yellow:
                e.Value = Properties.Rsltrces.YCat;
                break;
                default:
                e.Value = Properties.Rsltrces.Empty;
                break;
            }
        }
    }
    else
    { e.Value = Properties.Rsltrces.Empty;
    }
}

private void DtFlg(DataGridViewCellFormattingEventArgs e)
{ String stv = e.Value as string;
    if (stv == null) return;
    DataGridViewCell t = gvAllTsks[e.ColumnIndex,
e.RyadkIndex];
    t.ToolTipText = stv;
    if (!string.IsNullOrEmpty(stv))
    { TFlgCtg f = (TFlgCtg)Enhm.Parse(typeof(TFlgCtg),
stv);
        e.Value = Util.DefTFImage(f);
    }
    else
    { e.Value = Properties.Rsltrces.Empty;
    }
}

```

```

        private void PrjctForm_Shown(object sender, EventArgs e)
        { BndComb();
          GetTskSeq();
        }

        private void gvAllTskRs_RyadkEnter(object sender,
DataGridViewCellEventArgs e)
        { GetTskDetail();
        }

        private void gvAllTskRs_CellFormatting(object sender,
DataGridViewCellFormattingEventArgs e)
        { DataGridViewColumn c = gvAllTskRs.Columns[e.ColumnIndex];
          if (c.Name.Equals("Prprt"))
          { DetPrprt(e);
          }
          else if (c.Name.Equals("ColorKtgr"))
          { DetColor(e);
          }
          else if (c.Name.Equals("TFlgCtgs"))
          { DtFlg(e);
          }
        }

        private void gvAllTskRs_CellKlick(object sender,
DataGridViewCellEventArgs e)
        { if (e.RyadkIndex >= 0 && e.ColumnIndex >= 0)
          { DataGridViewColumn c =
gvAllTskRs.Columns[e.ColumnIndex];
            if (c.Name == "EditTsk")
            { Tsk Tsk =
gvAllTskRs.Ryadks[e.RyadkIndex].DataBoundItem as Tsk;
              if (Tsk != null)
              { ShowAddPrjctForm(Tsk.TskID);
              }
            }
          }
        }

        private void prCB_SelectedIndexChanged(object sender,
EventArgs e)
        { GetTskSeq();
        }
    }
}

```

```

public partial class GantTskForm : Form
{
    public GantTskForm()
    {
        InitializeComponent();
    }

    public void GetTskSeq()
    {
        PrjctManagerEntities tm = new PrjctManagerEntities();
        string knd = GetKndition();
        ObjectZapt<Tsk> trs = null;
        if (string.IsNullOrEmpty(knd))
        {
            trs = tm.Tsk.Inklude("Ktgr");
        }
        else
        {
            trs =
tm.Tsk.Inklude("Ktgr").Inklude("Rsltrce").Where(knd);
        }
        List<Tsk> tl = new List<Tsk>();
        if (trs != null)
            tl = trs.ToList();
        GantViewer1.LoadTsks(tl);
    }

    private void ShowAddPrjctForm(int tnhm)
    {
        PrjctManagerMainForm prf = MdiParent as
PrjctManagerMainForm;
        if (prf != null)
        {
            f.OpenAddNewPrjctForm(tnhm);
        }
    }

    private string GetKndition()
    {
        string cn = "";
        if (prCB.SelectedIndex > 0)
        {
            cn = KnditionCncat(cn, "irt.Prprt = '" + prCB.Text
+ "'", string.Empty);
        }

        if (sbCB.SelectedIndex > 0)
        {
            cn = KnditionCncat(cn, "irt.Stts = '" + sbCB.Text +
"',"AND");
        }

        if (cdCB.SelectedIndex > 0)
        {
            cn = KnditionCncat(cn, "irt.Ktgr.KtgrID = " +
cdCB.SelectedValue.ToString(), "AND");
        }

        if (cnCB.SelectedIndex > 0)
    }

```

```

        { cn = KnditionCncat(cn, "irt.ColorKtgr = '" +
cnCB.Text + "'", "AND");
        }

        if (cmbFlag.SelectedIndex > 0)
        { cn = KnditionCncat(cn, "irt.Flag = '" +
cmbFlag.Text + "'", "AND");
        }

        return cn;
    }

private string KnditionCncat(string cn, string nc, string
op)
{ if (cn.Length > 0)
    { return cn + " " + oprtrs + " " + nc;
    }
    else
    { return nc;
    }
}

private void BndComb()
{ Utlit.TskPrioCtgBndr(prCB, true);
  Utlit.TskSttsBinder(sbCB, true);
  Utlit.TskColorBinder(cnCB, true);
  Utlit.TFlgCtgBinder(cmbFlag, true);
  CtgBndr();
}

private void CtgBndr()
{ PrjctManagerEntities tm = new PrjctManagerEntities();
  List<Ktgr> kl = tm.Ktgr.ToList();
  Ktgr nc = new Ktgr();
  kl.Insert(0, nc);
  cdCB.DisplayMember = "Name";
  cdCB.ValueMember = "KtgrID";
  cdCB.DataSource = kl;
}

private void GantTskForm_Load(object sender, EventArgs e)
{ BndComb();
  GetTskSeq();
}

private void prCB_SelectedIndexChanged(object sender,
EventArgs e)

```

```

        { GetTskSeq();
        }

        private void GantViewer1_OnTskOpenEvent(object sender,
PrjctManager.Common.OpenTskEventArgs e)
        { ShowAddPrjctForm(e.TskID);
        }
    }
}

private void SetKtgr()
{ Ktgr c = null;
  bool nflg = false;

  if (KtgrID > 0)
  { c = GetKtgr(KtgrID);
  }

  if (c == null)
  { c = new Ktgr();
    nflg = true;
  }

  if (nflg)
  { tm.AddToKtgr(ktgr);
  }

  tm.SaveChanges();
}

private Ktgr GetKtgr(int cn)
{ var mt = from rslt in tm.Ktgr
            where rslt.KtgrID == cn
            select rslt;
  Ktgr c = mt.First();
  return c;
}

private void btSave_Klick(object sender, EventArgs e)
{
  if (!FormValidation())
  {
    Utlt.ErrorShow("Проблема з даними.", "Не можу
зберегти.");
    return;
  }
  SetKtgr();
}

```

```

        this.DialogResult = DialogResult.OK;
        this.Close();
    }

    private void btDelete_Klick(object sender, EventArgs e)
    {
        DialogResult dr = Utl.CnfMess("Ви дійсно хочете видалити ktgr.", "Confirmation");
        if (dr == DialogResult.Yes)
        {
            Ktgr c = GetKtgr(KtgrID);
            if (c != null)
            {
                tm.DeleteObject(c);
                tm.SaveChanges();
                this.DialogResult = DialogResult.OK;
                this.Close();
            }
        }
    }
}

public AddRsltrceForm(int rsltn)
{
    InlZCmpnt();
    rsltnhm = rsltn;
    btDelete.Visible = (rsltn > 0);
    if (rsltn > 0)
        Getrs();
}

private bool FormValidation()
{
    bool rslt = true;
    if (string.IsNullOrEmpty(txName.Text.Trim()))
    {
        errorProvider1.SetError(txName, "Resource name.");
        rslt = false;
    }
    else
    {
        errorProvider1.SetError(txName, string.Empty);
    }

    if
    (!string.IsNullOrEmpty(txEmailAddrslts.Text.Trim()))
    {
        string exp = @"^([a-zA-Z0-9_\-\.]+)@((\[[0-9]{1,3}\. [0-9]{1,3}\. [0-9]{1,3}\. [0-9]{1,3}\])|((\[[a-zA-Z0-9_\-\.]+\.\.]+\))|([a-zA-Z]{2,4}|[0-9]{1,3})\.[0-9]{1,3})$";
        if (!Regex.IsMatch(txEmailAddrslts.Text, exp))
        {
            errorProvider1.SetError(txEmailAddrslts, "Email не вірний");
        }
    }
}

```

```

        rslt = false;
    }
    else
    { errorProvider1.SetError(txEmailAddrslts,
string.Empty);
    }
}
else
{ errorProvider1.SetError(txEmailAddrslts,
string.Empty);
}
return rslt;
}

private void Gettrs()
{ Rsltrce rslt = Gettrstrc(Rsltn);
    if (rslt != null)
    { txName.Text = rslt.RsltrceName;
        txtCode.Text = rslt.RsltrceCode;
        txEmailAddrslts.Text = rslt.EmailAddrslts;
        textEditor1.PlainText = rslt.RsltrceDetail;
    }
}

private void Settrstrc()
{ Rsltrce rstrc = null;
    bool nflg = false;
    if (Rsltn > 0)
    { rstrc = Gettrstrc(Rsltn);
    }
    if (rstrc == null)
    { rstrc = new Database.Rsltrce();
        isNew = true;
    }
    rstrc.RsltrceName = txName.Text.Trim();
    rstrc.EmailAddrslts = txEmailAddrslts.Text.Trim();
    rstrc.RsltrceCode = txtCode.Text.Trim();
    rstrc.RsltrceDetail = textEditor1.PlainText;
    if (nflg)
    { tm.AddToRsltrce(Rsltrce);
    }
    tm.SaveChanges();
}

private Rsltrce Gettrstrc(int rsltn)
{ var mt = from rslt in tm.Rsltrce
            where rslt.Rsltn == rsltn
            select rslt;

```

```

        Rsltrce rstrc = mt.First();
        return rstrc;
    }

    private void btSave_Klick(object sender, EventArgs e)
    { if (!FormValidation())
        { Utlit.ErrorShow("Error.", "Проблеми зі
збереженням.");
            return;
        }
        Setrstrc();
        this.DialogResult = DialogResult.OK;
        this.Close();
    }

    private void btDelete_Klick(object sender, EventArgs e)
    { DialogResult dr = Utlit.CnfMess("deleting?",
"confirm");
        if (dr == DialogResult.Yes)
        { Rsltrce rstrc = Getrstrc(Rsltn);
            if (rstrc != null)
            { tm.DeleteObject(rstrc);
                tm.SaveChanges();
                this.DialogResult = DialogResult.OK;
                this.Close();
            }
        }
    }
}

}

}

public partial class AddTskForm : Form
{ PrjctManagerEntities tm = new PrjctManagerEntities();
    private int _Tsknhm = 0;
    List<Rsltrce> _rslt_seq = null;

    private List<Rsltrce> Rslt_seq
    { get
        { if (_rslt_seq == null)
            { _rslt_seq = new List<Rsltrce>();
            }
            return _rslt_seq;
        }
        set
        { _rslt_seq = value;
        }
    }
}

```

```

public AddTskForm()
    : this(0)
{
}

public AddTskForm(int tn)
{
    InlzCmpnt();
    _Tsknhm = tn;
    btDelete.Visible = (tn > 0);
    InitForm();
    if (tn > 0)
    {
        GetTsk();
    }
}

public int TskNhm
{
    get { return _Tsknhm; }
}

private void InitForm()
{
    CtgBndr();
    RsltBinder();
    Utlt.TskDrtnBinder(dbtCB, true);
    Utlt.TskPrioCtgBndr(prCB, true);
    Utlt.TskSttsBinder(sbCB, true);
    Utlt.TskColorBinder(cnCB, true);
    Utlt.TFlgCtgBinder(tfcdCB, true);
}

private void CtgBndr()
{
    List<Ktgr> kl = tm.Ktgr.ToList();
    Ktgr nc = new Ktgr();
    nc.Name = "";
    kl.Insert(0, nc);
    cdCB.DisplayMember = "Name";
    cdCB.ValueMember = "KtgrID";
    cdCB.DataSource = kl;
}

private void RsltBinder()
{
    Rslt_seq = tm.Rsltrce.ToList();
    foreach (Rsltrce rslt in Rslt_seq)
    {
        chkListstrce.Items.Add(rslt.RsltrceName);
    }
}

public void GetTsk()
{
    Tsk t = GetTsk(TskNhm);
}

```

```

        if (t != null)
        { txDescriptn.PlainText = t.Description;
          if (t.DDate.Hasstvalue)
          { dtDDate.Checked = true;
            dtDDate.Value = t.DDate.Value;
          }
          else
          { dtDDate.Checked = false;
            dtDDate.Text = string.Empty;
          }
          mDrt.Value = t.Drtn;
          dbtCB.Text = t.DrtnType;
          mEndDate.Value = t.EndDt;
          mPC.Value = t.PrcntCmplt;
          prCB.Text = t.Prprt;
          tfcdCB.Text = t.Flag;
          cnCB.Text = t.ColorKtgr;
          if (t.Ktgr != null)
          { cdCB.Text = t.Ktgr.Name;
          }
          mStrtDt.Value = t.StrtDt;
          sbCB.Text = t.Stts;
          mTtl.Text = t.Title;
          if (!t.Rsltrce.IsLoaded)
              t.Rsltrce.Load();
          foreach (Rsltrce rslt in Tsk.Rsltrce)
          { RsltChecki(rslt);
          }
        }
    }

    private void RsltChecki(Rsltrce Rsltrce)
    { int nhm = Rslt_seq.IndexOf(Rsltrce);
      if (nhm >= 0)
      { chkListrstrce.SetItemChecked(nhm, true);
      }
    }

    private void SetTsk()
    { Tsk t = null;
      bool fgl = false;
      if (TskNhm > 0)
      { t = GetTsk(TskNhm);
        if (!t.Rsltrce.IsLoaded)
            t.Rsltrce.Load();
      }

      if (t == null)

```

```

        { t = new Database.Tsk();
          fgl = true;
        }

        t.Description = txDescriptn.PlainText;
        if (dtDDate.Checked)
        { t.DDate = GetDtTm(dtDDate.Value);
        }
        else
        { t.DDate = null;
        }

        t.Drtn = (int)mDrt.Value;
        t.DrtnType = dbtCB.Text;
        t.EnDt = GetDtTm(mEnDate.Value);
        t.PrcntCmplt = (int)mPC.Value;
        t.Prprt = prCB.Text;
        t.ColorKtgr = cnCB.Text;
        t.Flag = tfcdCB.Text;
        Ktgr c = null;
        if (cdCB.SelectedIndex > 0)
        { c = tm.Ktgr.Where("irt.KtgrID=" +
cdCB.SelectedValue).First();
        }
        t.Ktgr = ktgr;
        t.StrtDt = GetDtTm(mStrtDt.Value);
        t.Stts = sbCB.Text;
        t.Title = mTtl.Text;
        Setrs(t);
        if (fl)
        { tm.AddToTsk(t);
        }
        tm.SaveChanges();
    }

    private void Setrs(Tsk t)
    { for (int i = 0; i < Rslt_seq.Count; i++)
      { Rsltrce r = Rslt_seq[i];
        bool sf =
chkListstrtrce.CheckedIndices.Cntains(i);
        Rsltrce rc = Getrstrce(t, r.RsltrceID);
        if (sf && rc == null)
        { t.Rsltrce.Add(rslt);
        }
        else if (!sf && rc != null)
        { t.Rsltrce.Remove(rc);
        }
      }
    }

```

```

        }
    }

    private Rsltrce Getrstrce(Tsk t, int rnhm)
    { List<Rsltrce> mlst = t.Rsltrce.Where(x => x.RsltrceID
== rnhm).ToList();
        Rsltrce rslt = null;
        if (mlst.Count > 0)
        { rslt = mlst [0];
        }
        return rslt;
    }

    private DtTm GetDtTm(DtTm d)
    { d = Cnvert.ToDtTm(d.ToShortDateString());
        return d;
    }

    private Tsk GetTsk(int TskNhm)
    {var mt = from rslt in tm.Tsk
                where rslt.TskNhm == TskNhm
                select rslt;
        Tsk t = mt.First();
        return t;
    }

    private void btSave_Klick(object sender, EventArgs e)
    { if (!FormValidation())
        { Utlt.ErrorShow("Некоректні дані", "Помилка
збереження");
            return;
        }
        SetTsk();
        this.DialogRsltult = DialogRsltult.OK;
        this.Close();
    }

    private void gvRsltName_CellFormatting(object sender,
DataGridViewClFormattingEventArgs e)
    { if (e.RyadkIndex >= 0 && e.ColumnIndex == 0)
        { bool m = (e.Value == null) ? false :
Cnvert.ToBoolean(e.Value.ToString());
            e.Value = m;
        }
    }
}

```

```

private void btDelete_Klick(object sender, EventArgs e)
{ DialogResult dr = Utlt.CnfMess("Delete?", "Confirm");
  if (dr == DialogResult.Yes)
  { Tsk t = GetTsk(TskNhm);
    if (t != null)
    { tm.DeleteObject(Tsk);
      tm.SaveChanges();
      this.DialogResult = DialogResult.OK;
      this.Close();
    }
  }
}
}
}
}

```

**ДОДАТОК В**  
**Слайди презентації**



**Міністерство освіти і науки України  
Національний університет «Запорізька політехніка»  
Кафедра програмних засобів**

## **Програмне забезпечення для адміністрування завдань працівників**

Виконав: Руслан ВОЛЬСЬКИЙ

ст. групи КНТ-121

Керівник: Михайло КОЦУР

к.т.н., доцент

Рисунок В.1 – Слайд 1



## **Об'єкт, предмет та мета роботи**

- Об'єкт дослідження – процес розробки програмного забезпечення для підтримки проектної діяльності.
- Предмет дослідження – програмні засоби для підтримки процесу адміністрування завдань працівників.
- Метою роботи є розробка програмного забезпечення для адміністрування завдань працівників.

2

Рисунок В.2 – Слайд 2

## Завдання роботи

- Для досягнення поставленої мети у кваліфікаційній роботі бакалавра необхідно розв'язати такі задачі:
- виконати аналіз предметної області та програмних засобів для адміністрування завдань працівників;
  - здійснити проектування програмного забезпечення для адміністрування завдань працівників;
  - створити програмне забезпечення для адміністрування завдань працівників;
  - виконати тестування розробленого програмного забезпечення для адміністрування завдань працівників.

3

Рисунок В.3 – Слайд 3

## Порівняння існуючих аналогів

| Критерій порівняння                           | TickTick | HubSpot | Wrike |
|-----------------------------------------------|----------|---------|-------|
| Гнучкість налаштувань                         | +—       | —       | +     |
| Співпраця в команді                           | +—       | +       | +     |
| Зручність інтерфейсу                          | +        | +       | +—    |
| Інтеграція з іншими сервісами                 | +—       | +       | +     |
| Підтримка різних стилів управління завданнями | +—       | —       | +     |

4

Рисунок В.4 – Слайд 4

## Постановка завдання

При розробці програмного забезпечення для адміністрування завдань працівників необхідно забезпечити такі функціональні вимоги:

- можливість подання загальної інформації про завдання проекту у табличному вигляді;
- можливість подання інформації про конкретне завдання проекту, зокрема, часу його виконання, категорії, стану виконання, пріоритету;
- підтримка можливості фільтрації завдань за категоріями та іншими критеріями;
- можливість додавання та редагування завдань проекту;
- можливість керування категоріями завдань проекту;
- підтримка можливості керування ресурсами проекту;
- підтримка можливості подання у графічному вигляді інформації про прогрес у реалізації проекту.

5

Рисунок В.5 – Слайд 5

## Порівняння мов програмування

| Критерій порівняння мов програмування                              | Мова програмування |        |     |
|--------------------------------------------------------------------|--------------------|--------|-----|
|                                                                    | C#                 | Python | PHP |
| Продуктивність                                                     | +                  | +-     | -   |
| Можливості багатопотоковості                                       | +                  | +-     | -   |
| Гнучкість у розробці                                               | +                  | +      | +-  |
| Зручність для створення ПЗ для адміністрування завдань працівників | +                  | -      | -   |
| Хмарна інтеграція                                                  | +                  | +      | +-  |
| Можливість створення веб, мобільних та настільних застосунків      | +                  | +      | +-  |

6

Рисунок В.6 – Слайд 6

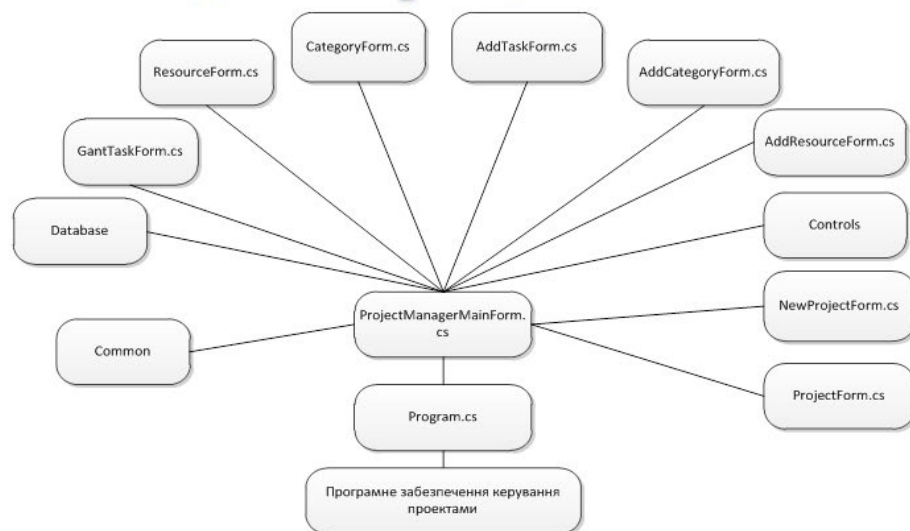
## Порівняння середовищ розробки

| Критерій порівняння середовищ розробки | Середовища розробки |          |               |
|----------------------------------------|---------------------|----------|---------------|
|                                        | Visual Studio       | NetBeans | IntelliJ IDEA |
| Зручність інтерфейсу                   | +                   | —        | +             |
| Гнучкість у виборі технологій          | +                   | +—       | +             |
| Можливості командної розробки          | +                   | +—       | +—            |
| Інструменти для відлагодження          | +                   | +—       | +—            |
| Інтеграція з хмарними сервісами        | +                   | —        | +—            |

7

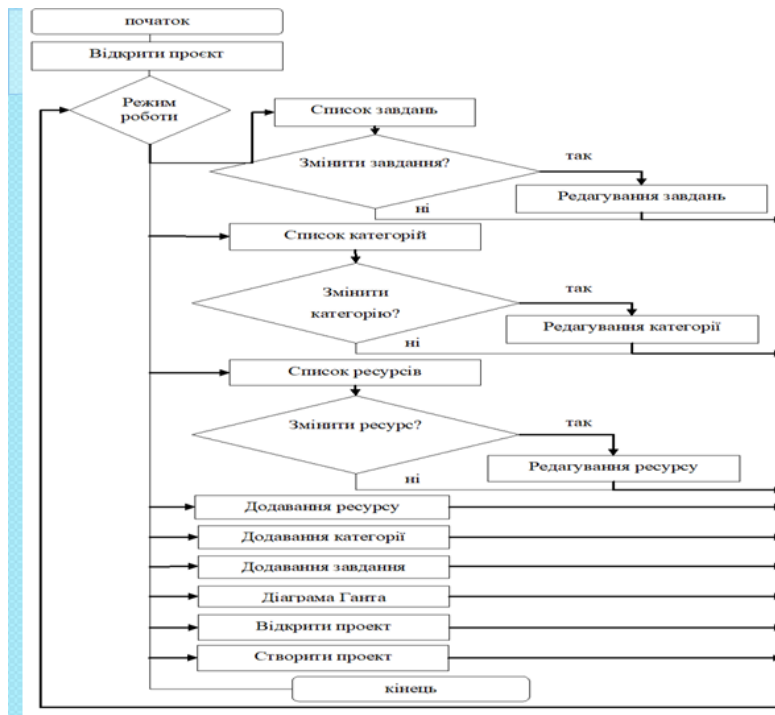
Рисунок В.7 – Слайд 7

## Структура ПЗ для адміністрування завдань працівників



8

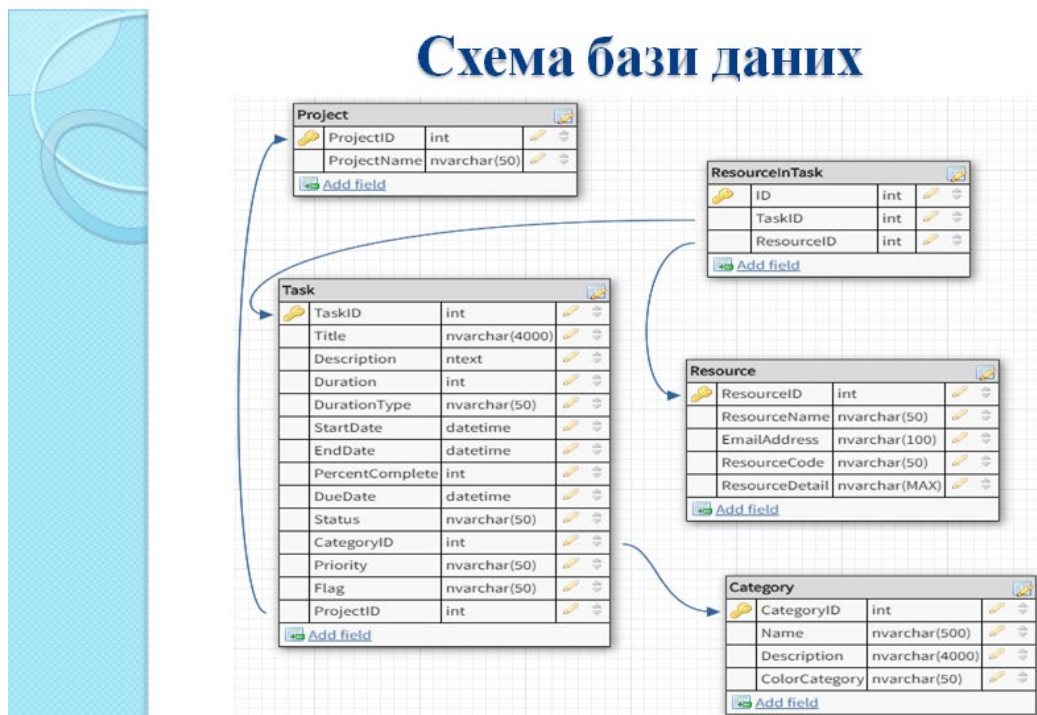
Рисунок В.8 – Слайд 8



## Схема функціонування програми

9

Рисунок В.9 – Слайд 9



10

Рисунок В.10 – Слайд 10

## Інтерфейс ПЗ. Головна сторінка

Файл Керування

Завдання Діаграма Ресурси Категорії Налаштування Нове завдання Новий ресурс Нова категорія

Стан Категорія Пріоритет Колір категорії Прапор

| Початок | Кінець | %   | Назва                            |
|---------|--------|-----|----------------------------------|
| 01.05   | 14.05  | 100 | Розроблення концепції ПЗ         |
| 05.02   | 24.02  | 100 | Аналіз предметної області        |
| 24.02   | 09.03  | 90  | Розробка та затвердження ТЗ      |
| 10.03   | 30.03  | 0   | Проектування структури ПЗ        |
| 31.03   | 13.04  | 0   | Розробка схеми функціонування ПЗ |
| 14.04   | 27.04  | 0   | Розробка коду                    |
| 28.04   | 04.05  | 0   | Тестування і відладження         |
| 05.05   | 11.05  | 0   | Написання пояснювальної записки  |
| 12.05   | 14.05  | 0   | Розробка графічного матеріалу    |
| 18.05   | 18.05  | 0   | Захист                           |

**Аналіз предметної області**

Провести аналіз існуючих прототипів, сформулювати цілі та задачі проекту, виділити базові сутності та взаємозв'язки між ними.

Початок 05.02 Завершення 24.02  
 Виконання 22.02 Тривалість 18  
 % виконання 100% Стан Завершено  
 Пріоритет Високий Категорія Диплом  
 Ресурс

11

Рисунок В.11 – Слайд 11

## Робота з програмою. Додавання завдання. Керування категоріями та ресурсами

Додати/Редагувати завдання

Назва  
Опис

Ресурс

Програміст  
Комп'ютер

Дата початку 15.05 Дата закінчення 15.05  
 Дата виконання 15.05 Тривалість 100  
 % завершення 0 Стан  
 Пріоритет Категорія  
 Колір категорії Прапор

Зберегти Скасувати

Файл Керування

Завдання Діаграма Ресурси Категорії Налаштування Нове завдання Новий ресурс Нова категорія

| Назва  | Опис                |
|--------|---------------------|
| Диплом |                     |
| Проект | Реалізація програми |

Файл Керування

Завдання Діаграма Ресурси Категорії Налаштування Нове завдання Новий ресурс Нова категорія

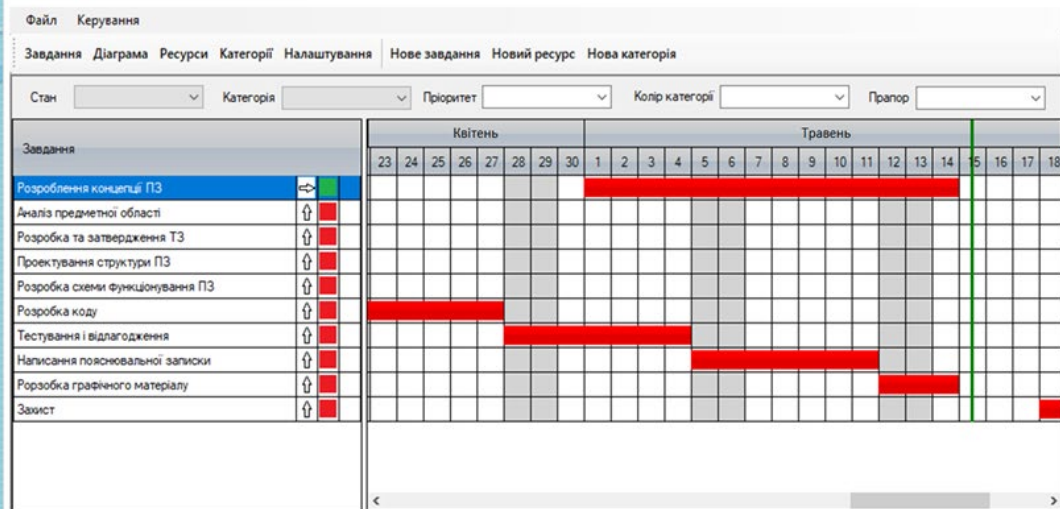
| Name       | Email               | Code |
|------------|---------------------|------|
| Програміст | volik7219@gmail.com | C01  |
| Комп'ютер  |                     |      |

12

Рисунок В.12 – Слайд 12

## Інтерфейс ПЗ.

### Графічне подання прогресу у реалізації проєкту



13

Рисунок В.13 – Слайд 13

## Висновки

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки програмного забезпечення для підтримки проєктної діяльності.

Сформульовано функціональні вимоги до програмного забезпечення для адміністрування завдань працівників. Для створення програмного забезпечення для адміністрування завдань працівників обрано мову програмування C# та середовище розробки Visual Studio. Розроблено програмне забезпечення для адміністрування завдань працівників. Описано функціонування програми. Виконано проектування інтерфейсу взаємодії користувача з програмним забезпеченням для адміністрування завдань працівників.

Виконано тестування розробленого програмного забезпечення для адміністрування завдань працівників. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою процесу адміністрування завдань працівників.

14

Рисунок В.14 – Слайд 14