Міністерство освіти і науки України
Криворізький національний університет
Факультет інформаційних технологій
Кафедра комп’ютерних систем та мереж

Пояснювальна записка
до кваліфікаційної роботи магістра
за спеціальністю 123 «Комп’ютерна інженерія»

на тему: Система контролю виконання завдань на промислових підприємствах

Проектував _____ В. В. Шевчук

Керівник роботи _____ М. О. Іщенко

Нормоконтроль _____ Д. І. Кузнєцов

Завідувач кафедри _____ А. І. Купін

Кривий Ріг
2022

Криворізький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерних систем та мереж

Ступінь вищої освіти магістр
Спеціальність 123 «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ
Завідувач кафедри, голова циклової комісії

_____ А. І. Купін

"____" _____ 20__ року

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

Шевчуку Владиславу Валерійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Система контролю виконання завдань на промислових підприємствах
керівник роботи Іщенко Микола Олександрович, к.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "__" ____ 2022 року _____
2. Строк подання студентом роботи _25.11.2022р.
3. Вихідні дані до роботи: PhpStorm, Apache, Git, Telegram API, організаційна структура ПАТ «АрселорМіттал Кривий Ріг», режими роботи (для адміністратора, робочого, менеджера та секретаря), локальна розгортка додатку.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): Проаналізувати існуючі системи управління, проаналізувати бізнес-процеси підприємства та методи їх удосконалення, визначити роль системи контролінгу на підприємстві, проаналізувати діяльність ПАТ «АрселорМіттал Кривий Ріг», розробити систему контролю виконання завдань на промислових підприємствах з інтеграцією Telegram API та виконати її тестування.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Презентація PowerPoint 24 слайди

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Іщенко М.О., доцент кафедри КСМ		
2	Іщенко М.О., доцент кафедри КСМ		
3	Іщенко М.О., доцент кафедри КСМ		

7. Дата видачі завдання_01.09.2022р._____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк виконання етапів роботи	Примітка
1	Збір та аналіз інформації про види систем управління проектами.	01.09.2022 – 04.09.2022	
2	Аналіз та порівняння систем управління проектами Trello, MS Project, Jira	05.09.2022 – 08.09.2022	
3	Аналіз та опис класифікації бізнес-процесів підприємств	09.09.2022 – 13.09.2022	
4	Аналіз методів методів удосконалення бізнес-процесів	14.09.2022 – 18.09.2022	
5	Аналіз ролі системи контролінгу на підприємстві	19.09.2022 – 22.09.2022	
6	Аналіз показників контролю ефективності	23.09.2022 – 26.09.2022	
7	Аналіз діяльності підприємства ПАТ «АсрелорМіттал Кривий Ріг»	27.09.2022 – 01.10.2022	
8	Інсталяція та налаштування веб-серверу, середовища розробки	02.10.2022 – 05.10.2022	
9	Розробка додатку	06.10.2022 – 06.11.2022	
10	Тестування додатку	07.11.2022 – 20.11.2022	
11	Оформлення звіту та підготовка презентації	21.11.2022 – 24.11.2022	

Студент _____ _____
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка: 91 сторінка, 103 рисунки, 13 таблиць, 6 додатків, 70 використаних джерел.

Проект складається з 3 розділів.

Перший розділ присвячений аналізу видів систем управління проектами, огляду та порівняння систем управління проектами Trello, MS Project та Jira.

У другому розділі описано класифікацію бізнес-процесів підприємств Розглянуто місце та роль бізнес-процесів в системі контролю виконання завдань, визначено особливості бізнес-процесів промислових підприємств на прикладі діяльності ПАТ «АрселорМіттал Кривий Ріг», визначено способи контролю та оцінки показників ефективності.

У третьому розділі виконано описано діяльність та структуру ПАТ «АрселорМіттал Кривий Ріг». Також було дано визначення веб-серверу, описано його встановлення та налаштування для розробки веб-додатку. Також визначено призначення системи контролю версій та описано роботу з Git. В даному розділі було описано процес розробки системи контролю виконання завдань, розглянуто роботу з нею та проведено її тестування.

СИСТЕМА УПРАВЛІННЯ ПРОЕКТАМИ, БІЗНЕС-ПРОЦЕС, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ДОДАТОК, ПІДПРИЄМСТВО, ЕФЕКТИВНІСТЬ.

Explanatory note: 91 pages, 103 figures, 13 tables, 6 appendices, 70 used sources.

The project consists of 3 sections.

The first chapter is devoted to the analysis of types of project management systems, an overview and comparison of project management systems Trello, MS Project and Jira.

In the second chapter, the classification of business processes of enterprises is described. The place and role of business processes in the task performance control system are considered, the features of business processes of industrial enterprises are determined on the example of PAO "ArcelorMittal Kryvyi Rih", methods of control and evaluation of performance indicators are determined.

The third chapter describes the activity and structure of ArcelorMittal Kryvyi Rih PJSC. It also defined a web server, described its installation and configuration for developing a web application. The purpose of the version control system is also defined and work with Git is described. In this section, the process of developing a task performance control system was described, work with it was considered, and its testing was carried out.

PROJECT MANAGEMENT SYSTEM, BUSINESS PROCESS, SOFTWARE, APPLICATION, ENTERPRISE, EFFICIENCY.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ІТ – інформаційні технології;
ПО – програмне забезпечення
СУП – систем управління проектами;
СКЗ – система контролю завдань;
ПП – промислові підприємства;
ВОІВ – Всесвітньої організація інтелектуальної власності;
KPI – key performance indicator – ключові показники ефективності;
ПАТ – публічне акціонерне товариство;
ПЗ – програмне забезпечення;
ECC – Error Correcting Code – код корекції помилок;
RAM – Random Access Memory – пам'ять з довільним доступом;
HTTP – HyperText Transfer Protocol – протокол прикладного рівня передачі даних;
HTTPS – Hyper Text Transfer Protocol Secure – протокол прикладного рівня передачі даних, що використовує шар шифрування SSL;
СКВЗ – система контролю виконання завдань;
IDE – Integrated development environment – інтегрована середа розробки;
СКВ – Concurrent Versions System, CVS – система контролю версій;
GUI – Graphical user interface – графічний інтерфейс користувача;
MVC – Model-View-Controller – архітектурний патерн проектування додатків, що складається з компонентів Моделі, Представлення та Контролера;
API – Application Programming Interface – інтерфейс програмних додатків;
CSV – comma-separated values – файловий формат табличних даних, у якому поля розділяються комою.

ВСТУП

На сьогоднішній день ефективність функціонування будь-якого підприємства залежить від диверсифікації виконуваних функцій, проектів – що пов’язано безпосередньо з розв’язанням великої кількості різноманітних задач, погодження документації, виконання доручень керівників та інше. Оскільки одночасно компанія, зазвичай, працює над множиною проектів, то спланувати та проконтролювати виконання усіх перерахованих видів робіт є дуже важкою задачею. Також це дуже сильно впливає на якість та швидкість виконання робіт [1].

Для вирішення цих питань використовують розробки у сфері інформаційних технологій (ІТ). Такою розробкою є система управління проектами. Сучасний ринок ІТ пропонує досить широкий вибір готових програмних рішень для керування проектами. Але навіть попри великий асортимент програмних рішень питання розробки власної системи керування проектами залишається актуальним.

Розробка власної системи дає можливість створити додатковий функціонал, що може бути обумовлений особливостями підприємства або компанії. Також така потреба може виникнути при необхідності тонкого налаштування політик конфіденційності та впевненості, що дані знаходяться у безпеці [2].

Тому **метою** даного наукового дослідження є розробка системи контролю виконання завдань на промислових підприємствах.

У відповідності до мети в магістерській роботі поставленні наступні завдання:

– провести опис та порівняння систем управління проектами;

– розглянути переваги та недоліки систем управління проектами Trello, MS Project та Jira;

– визначити місце системи контролю виконання завдань в бізнес-процесах;

– визначити особливості бізнес-процесів промислових підприємств та вказати на особливості діяльності ПАТ «АрселорМіттал Кривий Ріг» в контексті процесного підходу;

– розробити та описати програмну складову системи управління проектами враховуючи специфіку промислових підприємств.

Об’єктом дослідження даної магістерської роботи є розробка системи контролю виконання завдань на промислових підприємствах в контексті процесного підходу.

Предмет дослідження – практичне обґрунтування впровадження системи контролю виконання завдань на промислових підприємствах.

Враховуючи наступні переваги від використання систем управління проектами:

−спрощення планування проектних робіт;

−спрощення розподілу та оптимізації ресурсів за рахунок наочності завантаження кожного зі співробітників;

−спрощення відстежування прогресу розвитку проектів та можливість вчасного реагування на затримки виконання робіт;

−спрощення комунікації між співробітниками;

−спрощення звітності.

Враховуючи все вищезазначене можна стверджувати, що система управління проектами спрощує планування та управління проектами та дозволяє значно прискорити їх реалізацію.

Апробація роботи. Апробація досліджень відбувалась на конференції V Всеукраїнській науково-практичній інтернет-конференції молодих вчених та студентів «Сучасні інформаційні системи та технології» від Херсонського національного технічного університету на базі кафедри інформаційних технологій.

1 ОПИС ТА ПОРІВНЯННЯ СИСТЕМ УПРАВЛІННЯ ПРОЕКТАМИ

1.1 Види систем управління проектами

Сфера використання тієї чи іншої системи керування проектами буде залежати від того, до якої галузі вона відноситься та яка специфіка її застосування. Таким чином системами управління проектами може здійснюватися керування:

- Будівельними проектами;
- Інвестиційними проектами;
- Організаційними проектами;
- Проектами пов'язаними з розробкою та впровадження до застосування програмного забезпечення (ПЗ).

Також існують спеціалізовані системи управління проектами (СУП), які призначені для використання у певних галузях. Ще можуть бути інтегровані СУП, які дозволяють керувати декількома типами проектів [3].

В ході розвитку інформаційних систем керування проектами змінювались технології роботи систем. На сьогодні за технічними характеристиками СУП можна поділити на наступні види:

1) Локальні системи. Це програмний продукт, що інстальований на робочу станцію керівника відділу;
2) Клієнт-серверні системи. Таке програмне забезпечення складається з двох частин: серверної та клієнтської. На сервері налаштовуються основні компоненти програмного забезпечення, розгортаються певні сервіси, а на робочій станції працівника встановлюється відповідне ПЗ для роботи з сервісами;
3) Хмарні системи. Такі системи розгортаються лише на серверах. Для взаємодії з хмарними системами використовують веб-браузер [2, 4].

На даний час найбільш популярними системами є клієнт-серверні та хмарні системи, так як в умовах насичення ринку та обмеженості ресурсів, основна увага кожного підприємства приділяється ефективному управлінню проектами.

Дослідження будь-якої проблеми потребує спочатку визначення дефініцій таких категорій як: бізнес-процес, управління проектами, інновацій та інвестиції, контролінг та вибрати або синтезувати з наданих понять визначення показників системи управління проектами в відношенні до яких будуть оцінюватись показники діяльності підприємства.

Як відомо, питання управління проектами, визначення бізнес-процесів спираються на концепцію сталого розвитку, в якій основна увага приділяється створенню нововведень через обмеженість ресурсів та охорону навколишнього середовища. Тобто проблема управління проектами на підприємствах спирається на чотири основні складові: технічну, економічну, соціальну, екологічну. Про те розглянемо більш детально поняття бізнес-процес в управління проектами.

Бізнес-процес – в Оксфордському словнику, тлумачення цього терміну - серія дій, які здійснюються для досягнення певного результату [5].

Про те, в Великому тлумачному словнику сучасної мови цей термін в період з 1970 по 1980 рр. не має такого визначення.

Даний термін з'явився тільки в українському словнику в 2005 році і трактується, так: набір операцій, які спільно реалізують якесь завдання або мету організації. Також бізнес-процесом називають будь-яку діяльність або групу діяльностей яка додає вартість до вхідного продукту і забезпечує вихідним продуктом зовнішнього або внутрішнього споживача.

Щодо використання терміну бізнес-процес в наукометричній базі Гугл схолар, то згідно досліджень вітчизняних вчених поняття «бізнес-процес» було складено таблицю сфер застосувань, що наведено нижче.

Таблиця 1.1 – Сфери застосувань поняття «бізнес-процес»

Назва	Пошукова система Гугл, відповідь на запит	Питома вага,%
Бізнес-процес підприємства	7640000	26,8
Бізнес-процеси ІТ	4340000	15,2
Соціальні бізнес-процеси	7240000	25,4
Екологічні бізнес-процеси	4400000	15,5
Психологічні бізнес-процеси	4850000	17,0

Як видно з таблиці 1.1 бізнес-процеси застосовуються в різних сферах, про те в сфері ІТ дослідження науковців є не досить розповсюдженими. Розглянемо більш детально сфери застосування «бізнес-процес» в ІТ, економіці, соціології, екології.

Так, Чупріна М.О. ставить в своїх працях завдання оптимізації використання в управлінні саме сучасних програм для управління бізнес-процесами [6].

Погоджуємось з думкою Перерви П.Г., Назаренко С.М. про те, що промислові підприємства повинні самостійно розробляти ІТ комплекс для оцінки функціонування бізнес-процесів на підприємстві [7].

Що стосується конвергенції ІТ технологій і економіки, то тут слід звернути увагу на праці: Кравченко В.М., Осмятченко В. О., Пурденко О.А. – в основі яких лежить визначення ролі ІТ технологій в управлінні бізнес-процесами [8, 9].

Впровадження ІТ технологій для оцінки бізнес-процесів підприємства – це інноваційна діяльність підприємств. Розглянемо наскільки готові українські підприємства до таких інновацій. Автором зроблено прогноз частки інноваційно активних підприємств у загальній кількості промислових підприємств на 2019-2023 рр., таблиця 1.2.

Таблиця 1.2 – Прогноз інноваційно активних підприємств у загальній кількості промислових підприємств на 2019-2023 р.

Роки	Частка кількості інноваційно активних підприємств у загальній кількості промислових підприємств,%	Номер року, t	Роки	Частка кількості інноваційно активних підприємств у загальній кількості промислових підприємств,%
2014(1)	16,1	6	2019	16,46
2015(2)	17,3	7	2020	16,52
2016(3)	18,9	8	2021	16,58
2017(4)	16,2	9	2022	16,64
2018(5)	16,4	10	2023	16,7
2019(6)	16,4	-	-	-

Для проведення прогнозування використовуємо формулу середнього абсолютного приросту:

$$\overline{\Delta y} = \frac{y_k - y_0}{k-1} \quad (1.1)$$

Де:

– y_k – кінцеве значення рівня динамічного ряду;

– y_0 - початкове значення рівня динамічного ряду;

– k – кількість елементів динамічного ряду.

Отже:

$$\overline{\Delta y} = \frac{16,4 - 16,1}{6 - 1} = 0,06\%$$

На основі залежності (1.1) складемо прогноз частки інноваційно активних підприємств у загальній кількості промислових підприємств на період $(k + 1) \div n$.

–k – кількість елементів динамічного ряду.

Отже, виходячи із прогнозних значень частка промислових підприємств які бажають впроваджувати ІТ технології для оцінки бізнес-процесів (інновації) збільшується з 16,1% в 2014 році до 16,7% в 2023 році. Незначні темпи зростання свідчать про те що на дані процеси впливає безліч факторів. Розглянемо їх для вибору систему управління проектами.

Першим фактором є простота використання. Мета використання програмного забезпечення для керування проектами є оптимізація робочого процесу. Опції та інтерфейс СКВЗ має бути інтуїтивно зрозумілим, що зробить відстежування задач менш стомлюючим для вашої команди. Простота використання включає в себе наявність функцій автоматизації, що прискорюють виконання окремих дій [10].

Другим важливим фактором є гнучкість. Інструмент управління проектами повинен мати достатньо налаштувань, щоб відповідати вимогам вашої команди.

Наступним важливим фактором є інтегрованість. В компаніях, зазвичай, вже використовуються певні інструменти для роботи з проектами. Тому, важливо аби рішення для управління проектами було сумісним з ними. Наприклад, Jira, програмне забезпечення для керування проектами, може бути інтегровано з Bitbucket та Github.

Також важливим фактором вибору СУП є можливість до масштабування. Інструмент для управління проектами має підходити до проектів будь-яких розмірів [11].

Останній з найголовніших факторів вибору інструменту для керування проектами є ціна. Ціни можуть дуже сильно коливатися залежно від:

1. Кількості користувачів;
2. Наявності певних функцій;
3. Наявності додаткових налаштувань функцій.

Про те одним із вирішальних факторів впровадження є фактор залучення коштів для впровадження даних проектів на промислових підприємствах.

Розглянемо більш детально джерела залучення коштів промислових підприємств для фінансування впровадження ІТ технологій в керування проектами, таблиця .1.3.

Таблиця 1.3 – Джерела залучення коштів промислових підприємств.

Роки	Витрати на впровадження проектів			
	власні кошти підприємств, млн. грн.	кошти державного бюджету, млн. грн.	кошти інвесторів-нерезидентів, млн. грн.	інші джерела залучення коштів, млн. грн.
2016	22036	179,0	23,4	991,1
2017	7704,1	227,3	107,8	1078,3
2018	10742	639,1	107,0	692,0
2019	12474,9	556,5	42,5	1147,0
2020*	10084,6	650,9	47,3	1186
2021*	7694,4	745,3	52,1	1225

Відповідно до даних таблиці 1.3 було складено діаграму, що відображає величину сум залучення коштів промислових підприємств за роками у відсотках залежно від джерела.

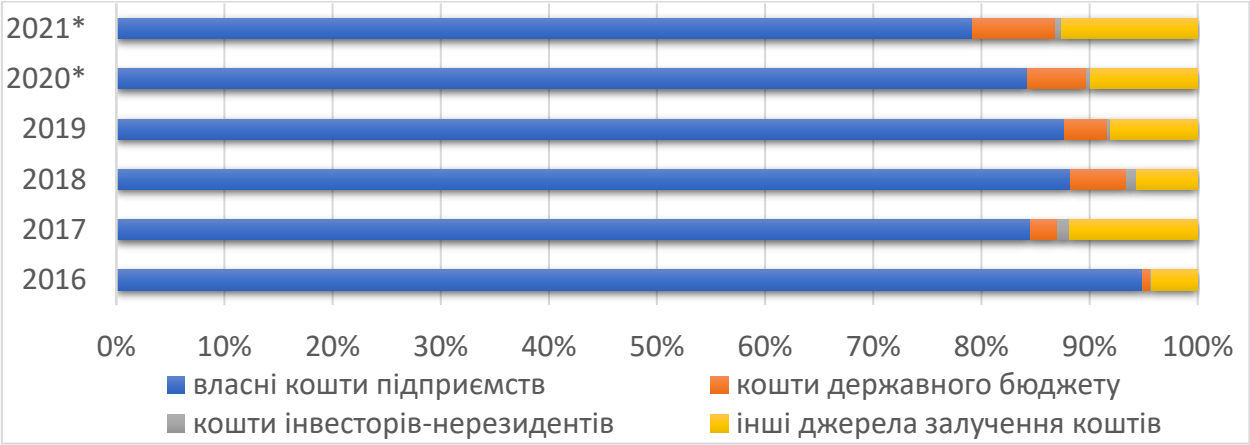


Рисунок 1.1 – Діаграма величини сум залучення коштів промислових підприємств за роками у відсотках залежно від джерела

Дані за 2020 та 2021 роки було розраховано самостійно. Отже, розглядаючи джерела фінансування проектів промислових підприємств можна стверджувати що в Україні найбільшу питому вагу в аналізований період 2016-2019 рр. (прогноз 2020-2021 рр.) займають власні кошти підприємств, часка фінансування складає 94,9 в 2016 р. та 87,7 % в 2019 р.). Частка витрат державних коштів досить невелика, про те вона поступово зростає що свідчить про те, що уряд країни розуміє потреби промислових підприємств в оцінці бізнес-процесів і систем управління проектами.

Розглянемо більш детально найпопулярніші системи управління проектами.

1.2 Інструмент керування роботою Jira

На сьогодні Jira є потужним інструментом управління роботою, що підходить майже для усіх типів проектів та задач. Найбільше Jira підійде окремим командам та компаніям, що займаються розробкою програмного забезпечення [12].

Особливості Jira:

1) Kanban-дошка - допомога команді забезпечити прозорість роботи над проектом, оптимізувати робочий процес, розподілити задачі з беклогу (списку невирішених задач).

2) Scrum-дошка - дозволяє керувати складним проектом, об’єднати команди з різних напрямків розробки продукту для досягнення однієї цілі.

3) Прив’язка програмного коду до задач за допомогою Bitbucket та Github і сумісна робота над ним.

4) Ведення документації, протоколів та інших документів за допомогою Confluence.

5) Звітність в Jira - звіти формуються за допомогою віджетів на панелі Dashboard та можуть містити інформацію про проект в цілому або про окремі його елементи. Звіти візуалізуються в графіки та діаграми.

6) Підтримка інтеграцій з множиною інструментів для розробки та інших сервісів.

Kanban-дошка це спосіб візуалізації задач за допомогою дощок, на яких задачі розташовуються в залежності від статусу. Стандартна Kanban-дошка ділиться на 4 колонки:

To do - список задач, що необхідно виконати

In progress - задачі, що взяті до роботи та вирішуються;

In QA - задачі, що виконані та перебувають на стадії тестування;

Done - задачі, що виконані та пройшли тестування [13].

До вказаних колонок можна додавати нові або видаляти зайві. На рисунку 1.2 наведено приклад Kanban-дошки.

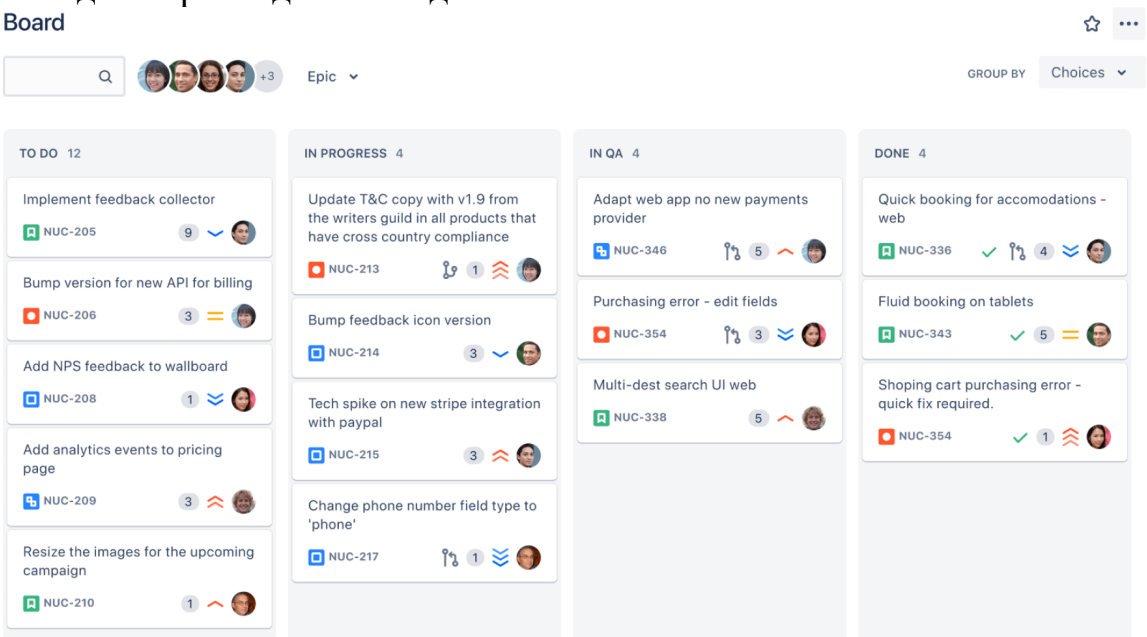


Рисунок 1.2 – Приклад Kanban-дошки

В Jira є зручний розподіл задач за їх об’ємом та типом. Види задач в Jira:

1) Epic (епік) - велика задача, що потрібно розділити на спрінти;
2) Story (історія) - частина великої задачі, яку команда може вирішити за спрінт;
3) Task (задача) - частина роботи, яку виконує один або декілька членів команди;
4) Sub-task (підзадача) - частина задачі
5) Bug (баг) - особлива задача з виправлення помилок у продукті.

На рисунку 1.3 зображено ієрархію задач за їх типами [14].

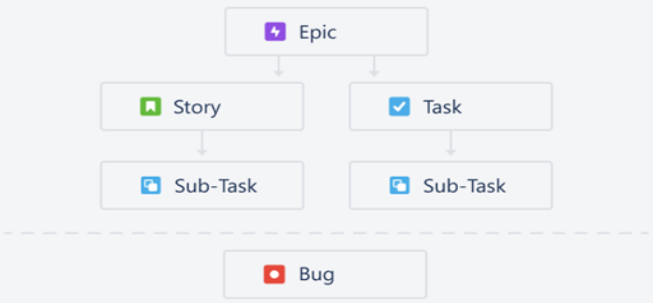


Рисунок 1.3 – Ієрархія задач за їх типами

Jira має власний мобільний додаток, що дозволяє користувачам отримати швидкий доступ до задач у будь-який час та з будь-якого місця.

Переваги мобільного додатку Jira:

1. Відповідайте швидше за допомогою push-сповіщень. Будьте у курсі що відбувається, швидше вирішуйте задачі, миттєво відповідайте на коментарі до задач, легко оновлюйте дошку та беклог команди, а також швидко переглядайте найважливіші подробиці розробки.
2. Забезпечте максимальну наочність завдяки єдиному джерелу достовірної інформації. Участь декількох команд в роботі над проектом ускладнює отримання оновлень статусів. У додатку Jira зібрані усі плани та відомості загального прогресу. Відкритий доступ до них підвищує обізнаність команд та прискорює просування робіт.
3. Просувайте роботу вперед у будь-який час та з будь-якого пристрою. Додаток Jira для iOS та Android дозволяє вести спільну роботу без прив'язки до фізичного офісу або іншого робочого середовища. Більше не треба довго чекати оновлення статусу роботи або відомостей про доступність співробітників. Інформацію не потрібно вносити двічі: усі оновлення в мобільному додатку автоматично транслюються в Jira Cloud для macOS та веб-версію.

На рисунку 1.4 зображено інтерфейс мобільного додатку Jira [15].

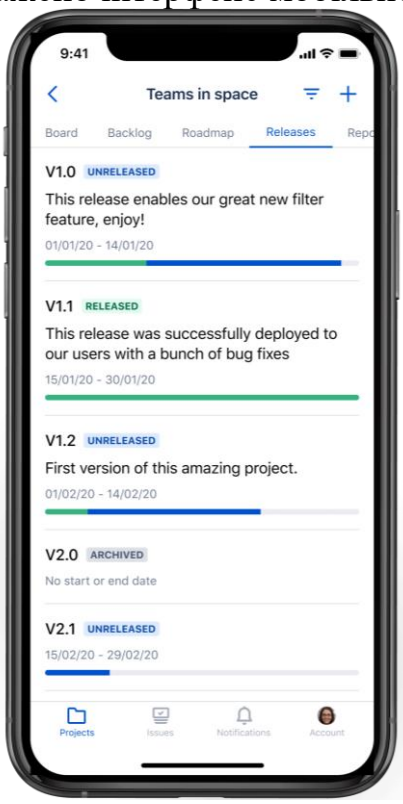


Рисунок 1.4 – Інтерфейс мобільного додатку Jira

Jira надає різного типу звіти в рамках проекту. Це допомагає аналізувати прогрес, проблеми, демонстрації та своєчасного виконання будь-якого проекту. Це також допомагає аналізувати використання ресурсів. Jira надає наступні звіти:

1. Average Age Report – середній вік (у днях) невирішених задач;
2. Created vs Resolved Issues Report – показує кількість створених задач у порівнянні з вирішеними проблемами за певний проміжок часу;
3. Pie Chart Report – показує результати пошуку з вказаного фільтру задач (або проекту) у виді кругової діаграми на основі обраної статистики;
4. Recently Created Issues Report – показує як швидко збільшується кількість задач;
5. Resolution Time Report – показує середній час на виконання задач;
6. Single Level Group By Report – показує результати пошуку з фільтру задач, що зібрані у групу за обраним полем;
7. Time Since Issues Report – показує кількість задач час на виконання спливає на задану дату;
8. User Workload Report – показує скільки робіт було призначено користувачу та скільки часу на виконання може знадобитись;
9. Version Time Tracking Report – показує скільки незавершеної роботи залишилось до завершення даної версії;
10. Workload Pie Chart Report –показує відносне робоче навантаження для відповідальних за усі задачі в певному проекту або фільтрі задач;

Також Jira пропонує інструмент для побудови дорожніх карт Advanced Roadmap. Даний інструмент доступний за підпискою Jira Software Cloud Premium. Дорожня карта об'єднує усі деталі проекту, використовується для планування великих об'ємів робіт на багато місяців наперед. Завдяки можливості встановлення залежностей між задачами команди можуть підготувати альтернативні маршрути та адаптуватися під поточний стан робіт. На рисунку 1.5 зображено дорожню карту в Jira [16].

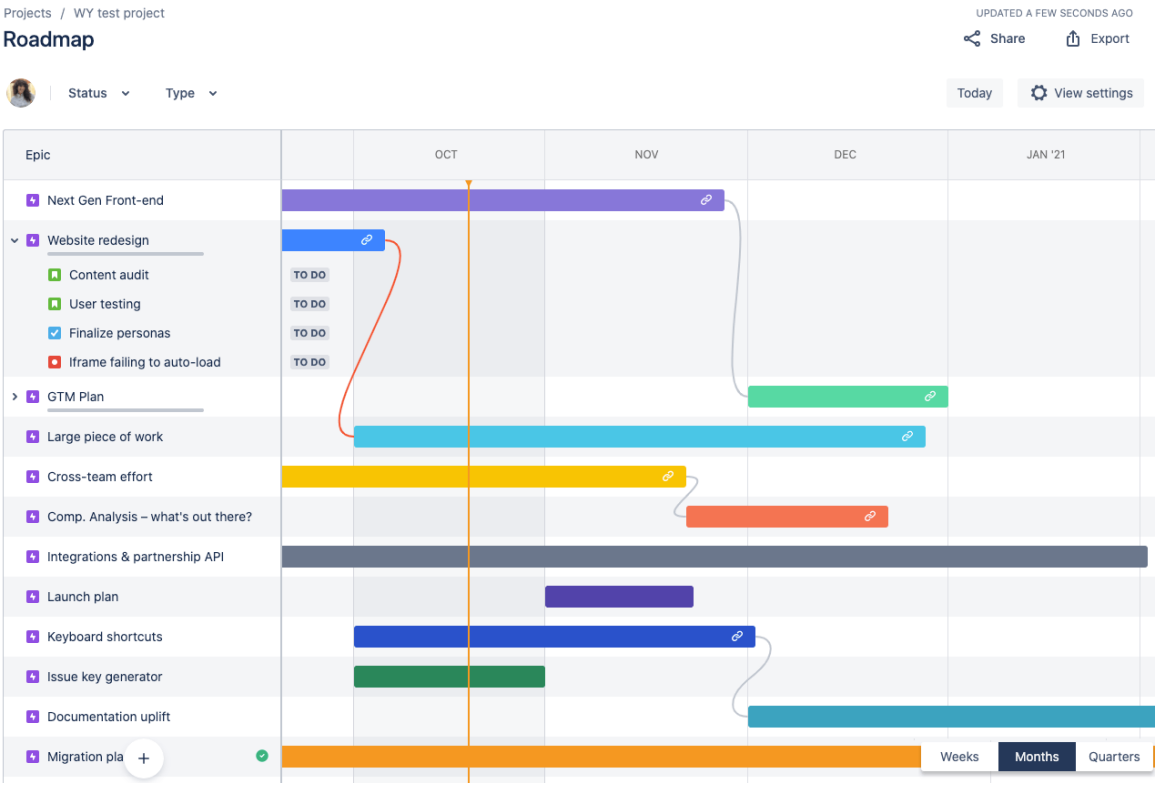


Рисунок 1.5 – Дорожня карта в Jira

Як видно з рисунку 1.5 в Jira Software можна відобразити схему залежностей прямо на дорожній мапі і за рахунок цього побачити зв'язки між Епіками. Також Advanced Roadmap передбачає розподіл за командами:

– Карти для відділу продаж. Передбачають наявність спеціальних функцій та переваг для клієнтів;

– Для команди розробників надається можливість реорганізації карт за спринтами, налаштувати відображення певних задач та проблемних зон які нанесені на шкалу часу;

– Для керівників надається можливість відстеження інформації за певними періодами (по кварталам, місяцям) за рахунок чого надаються короткі відомості про задачу або кожен з етапів розробки;

– Карти для клієнтів мета яких надати клієнтам уявлення про майбутнє продукту.

Таким чином дорожні карти допомагають організовувати спільну роботу команд , встановлювати залежності та планувати ресурси команди, а також відслідковувати прогрес [16].

Далі буде розглянуто систему управління проектами Trello.

1.3 Система управління проектами Trello

Trello – це одна з найпопулярніших систем управління проектами в режимі онлайн, яка має особливий попит серед невеликих компаній і стартапів. Вона дозволяє ефективно організовувати роботу з японської методології Kanban-дощок [17].

Trello складається з чотирьох основних елементів: робочого простору, дошки, списків та карточок. У робочому просторі відображаються усі дошки, які у вас є. Наприклад, замість використання спільної дошки для роботи та навчання можна розподілити ці процеси між двома дошками. На рисунку 1.6 зображено приклад робочого простору в Trello [18].

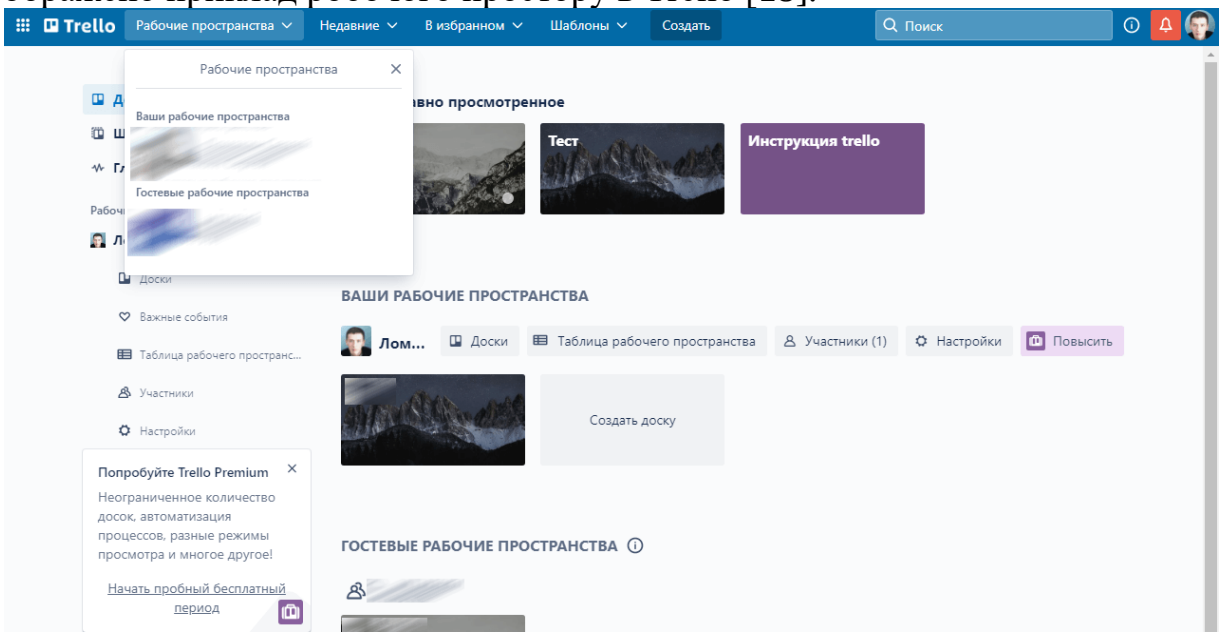


Рисунок 1.6 - Приклад робочого простору в Trello

Дошка – це простір який використовується для планування робіт та для самої роботи над певним процесом. Таким чином одну дошку ви використовуєте для ведення домашніх справ, другу для особистого проекту, третю для робочих задач і т.д.. Дошка може мати будь-яку кількість списків та карточок. На рисунку 1.7 зображено приклад дошки.

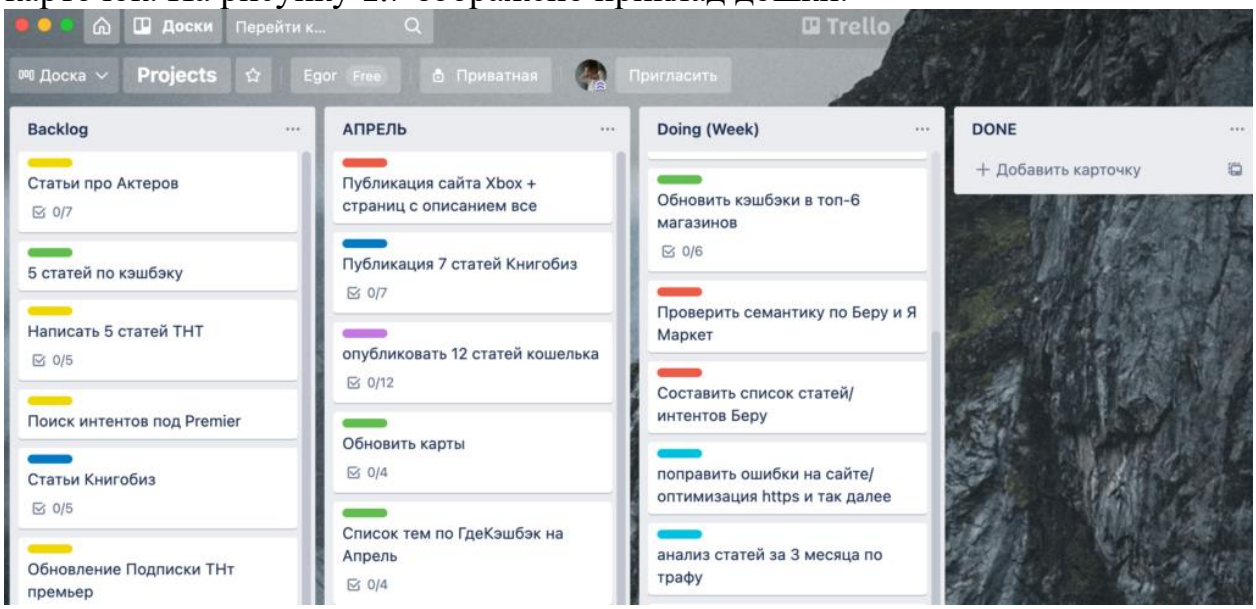


Рисунок 1.7 – Приклад дошки в Trello

Списки – вертикальні стовпці, що містять карточки. Списки можна переміщувати, копіювати та архівувати. Спискам можна дати назву, наприклад, залежно від етапу: треба виконати, в роботі, на тестуванні, виконано. Тобто дана дошка аналогічна дошці в Jira. На рисунку 1.8 зображено меню доступних дій зі списком.

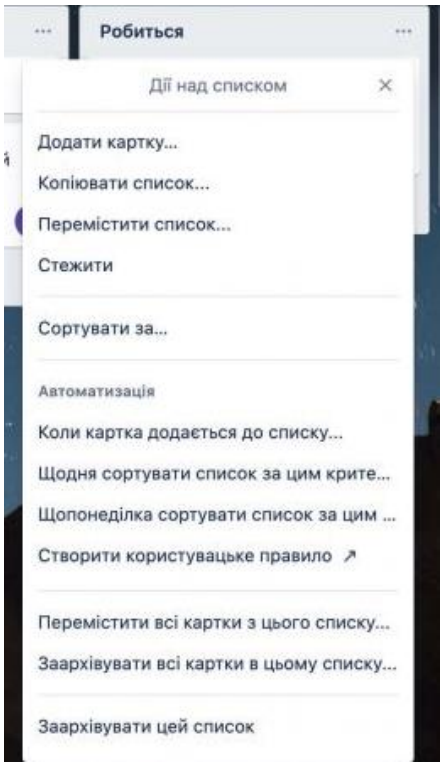


Рисунок 1.8 – Меню доступних дій зі списком у Trello

Карточка – є найменшою одиницею в Trello з найбільшим функціоналом. Саме карточка містить текст задачі. Карточка може бути переміщена між списками, що відображатиме статус виконання завдання. Якщо відкрити карточку, то можна побачити багато додаткових функцій: прикріплення файлів, зображень, створення міток, чек-листів та інше. У додатку А зображено приклад карточки. Для призначення відповідальної людини за вирішення задачі необхідно вказати його у списку учасників карточки. На рисунку 1.9 зображено приклад призначення відповідальної особи за вирішення задачі.

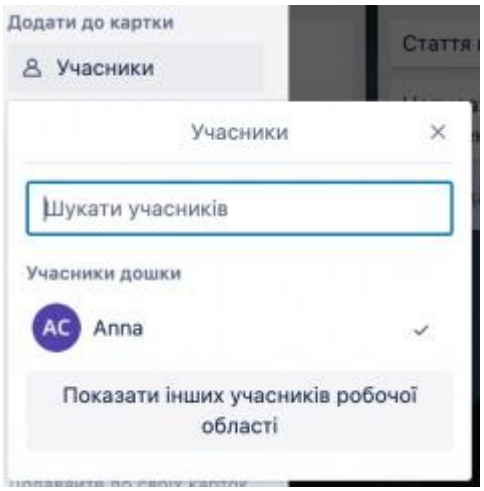


Рисунок 1.9 – Меню призначення задачі користувачу Anna

Після додавання учасників до карточки ви будете отримувати сповіщення про хід роботи та матимете можливість відстежувати їх в окремому вікні. Для перегляду усіх карточок, що призначені на вас необхідно використати відповідне меню у профілі. На рисунку 1.10 зображено меню для перегляду карточок на усіх дошках.

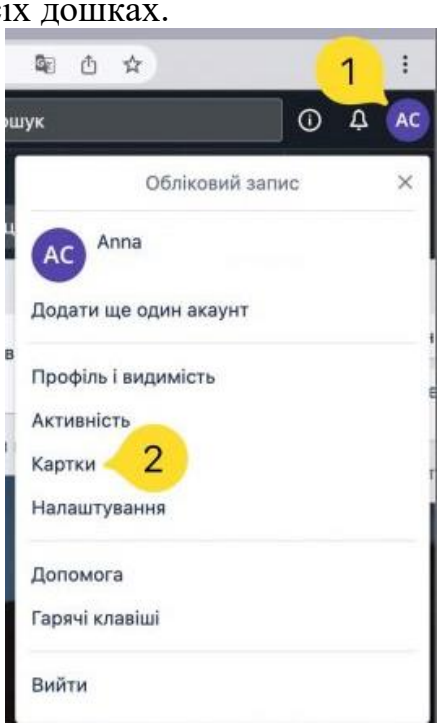


Рисунок 1.10 – Меню для перегляду карточок на усіх дошках

Кожна карточка-задача зберігає детальні логи взаємодії з нею. На рисунку 1.11 зображено історії дій в карточці [19].

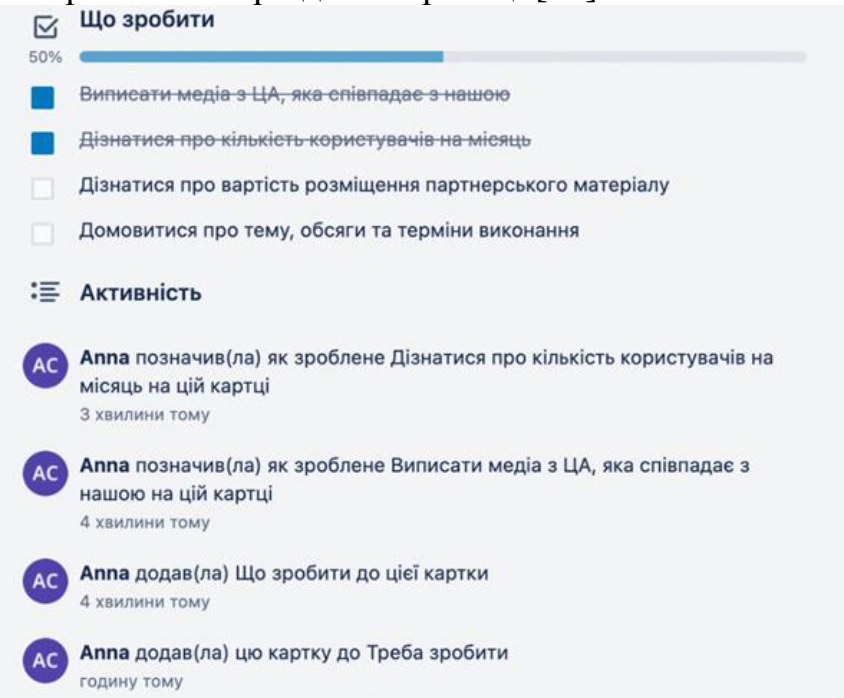


Рисунок 1.11 – Історія дій в карточці

Також карточки можна сортувати за такими параметрами, як: учасники, кінцеві терміни виконання задачі (дедлайн), мітки. На рисунку 1.12 зображено меню сортування карточок.

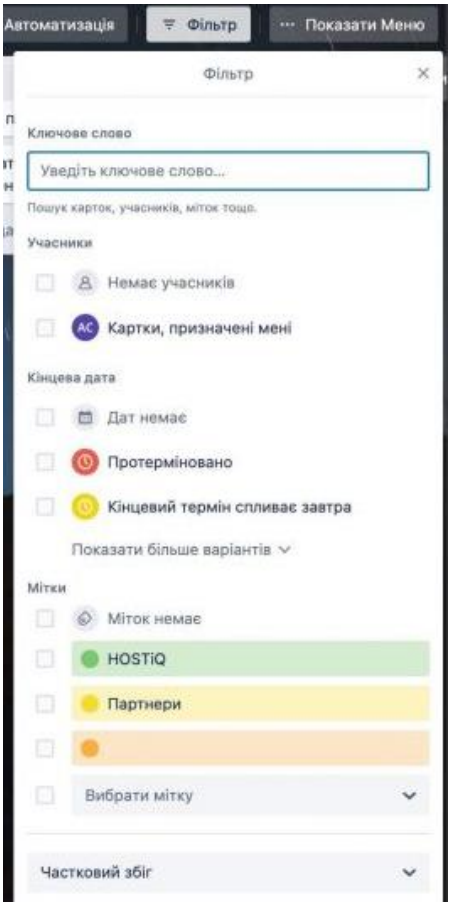


Рисунок 1.12 – Меню сортування карточок

Для ведення промислових та інфраструктурних проектів Trello не підійде, також не вийде повноцінно організувати з ним роботу цілої компанії. Але якщо потрібно організувати роботу над маленькими проектами або процесах, що не потребують спеціальних інструментів, то Trello є лідером серед аналогів [20].

Далі буде розглянуто систему управління проектами MS Project.

1.3 Система управління проектами MS Project

Система управління проектами MS Project було випущена у 2007 році компанією Microsoft. Управління проектами передбачає контроль над комплексом заходів, що направлені на управління ресурсами в рамках проекту, аналізу об’єму виконаних робіт та відстеження досягнутого прогресу. Дане програмне забезпечення є потужним та надійним рішенням для проектних груп з досвідом. MS Project у порівнянні з іншими інструментами складніший в освоєнні, але вивчивши тонкості даного ПЗ буде важко знайти альтернативне рішення подібного рівня [21].

Для початку роботи з програмою рекомендовано, щоб користувач мав базові знання про основи проектного підходу. Далі користувач має ознайомитися з методами та принципами проектування, що дозволить правильно використовувати наявні інструменти. Також необхідно вивчити техніку планування та навчитись будувати «проектний трикутник», що враховує час, об’єм та вартість робіт. Загалом Microsoft Project надає можливість виконувати наступні задачі:

– Покрокова розробка проекту;

– Виконувати розподіл ресурсів, постановку задач, задання термінів виконання задач, створення моделі проекту;

– Виконувати побудову мережевої діаграми, що містить задачі з їхніми пріоритетами та пов’язані з ними ресурсів;

– Виконувати оптимізацію плану проектних робіт на основі аналізу групи проектів та моніторингу наявних ресурсів;

– Виконувати розрахунок критичного шляху;

– Відмічати проблемні аспекти проектів, проблемні задачі;

– Редагувати кінцеві терміни виконання задач;

– Виконувати аналіз поточного стану проекту та адекватно оцінювати його перспективи;

– Створювати шаблони проектів;

– Моделювання різних варіантів вирішення різних ситуацій та розрахунок можливих наслідків.

Розглянемо приклад планування проекту із застосуванням MS Project. Після запуску додатку створимо пустий файл проекту, для цього необхідно обрати пункт «Blank Project».

Наступним кроком нам треба створити кінцеву задачу проекту. Для цього у вкладці Format необхідно встановити прапорець навпроти поля Project Summary Task, що зображено на рисунку 1.3.

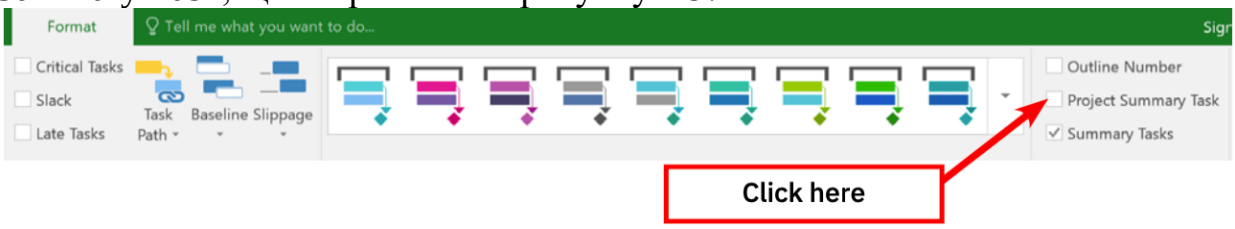


Рисунок 1.13 – Вкладка Format

Після цього в робочій області з'явиться запис з назвою Project, яку необхідно замінити на власну, наприклад проект магазину шоколаду. На рисунку 1.14 зображено запис створеного проекту.

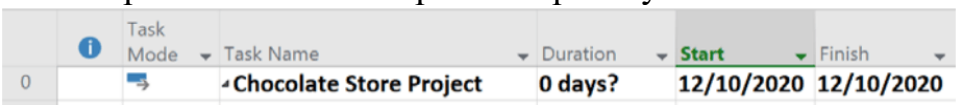


Рисунок 1.14 – Запис створеного проекту

Також одразу можна вказати дати початку та кінця проекту, але їх також можна налаштувати пізніше. Для редагування дат необхідно перейти до вкладки Project та обрати пункт Project Information. На рисунку 1.15 зображено вкладку пункт меню Project Information.

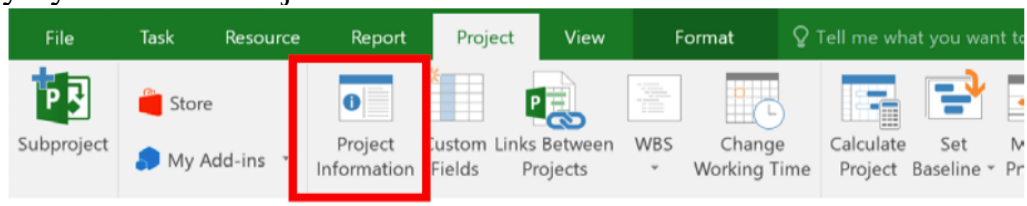


Рисунок 1.15 – Пункт меню Project Information

У відритому вікні можна налаштувати режим планування таким чином, Microsoft Project планував проект наперед визначеної дати або ж навпаки – назад від дати завершення. Таким чином у випадку коли перед вами стоїть умова, що проект повинен бути завершений до певної дати, то можна вказати її та обрати режим планування від дати здачі проекту до поточної. Для цього необхідно обрати «Schedule from Project End Date». На рисунку 1.16 зображено задання початкової та кінцевої дати виконання проекту [22].

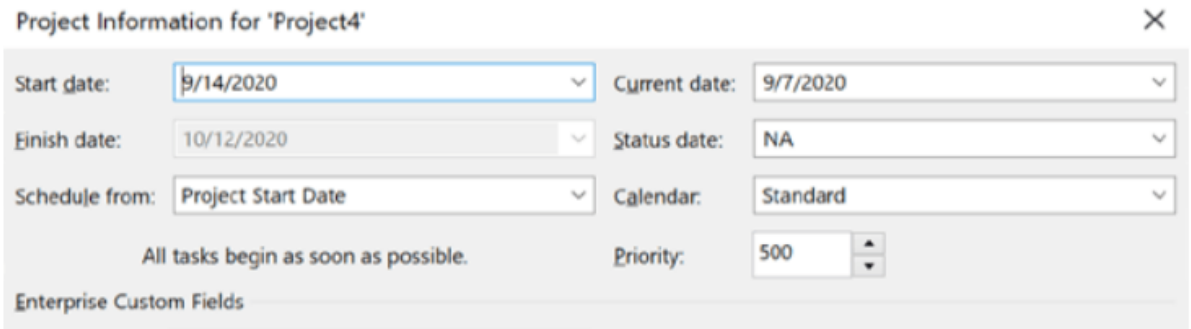
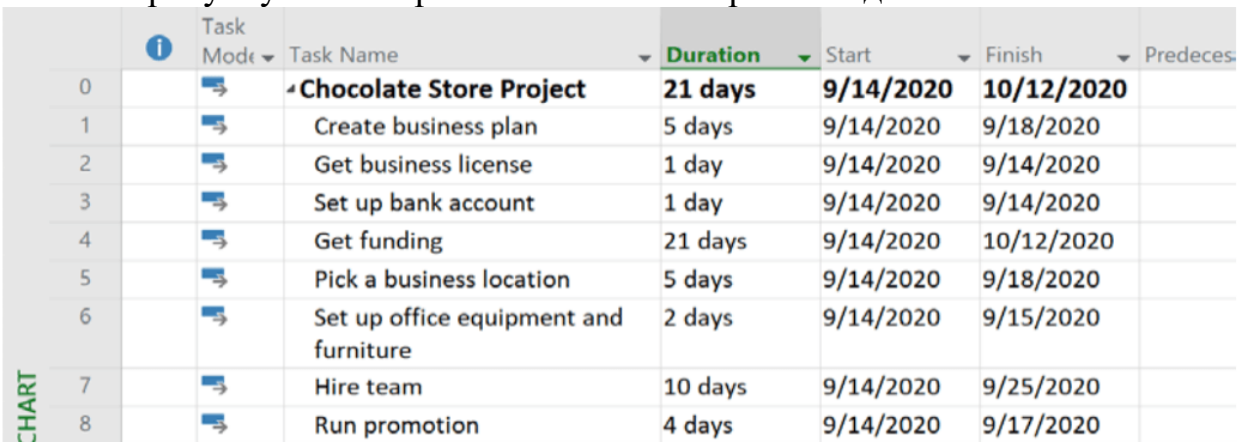


Рисунок 1.16 – Задання початкової та кінцевої дати проекту

Далі до проекту потрібно додати задачі, що необхідно виконати за даним проектом, наприклад:

– Створити бізнес-план;
– Отримати ліцензію на ведення бізнесу;
– Відкрити банківський рахунок;
– Отримати фінансування;
– Обрати місце для бізнесу;
– Розташувати офісну техніку та меблі;
– Найняти працівників;
– Запустити рекламу.

Після додання задач необхідно задати очікуваний час вирішення кожної з них. На рисунку 1.17 зображено список створених задач.



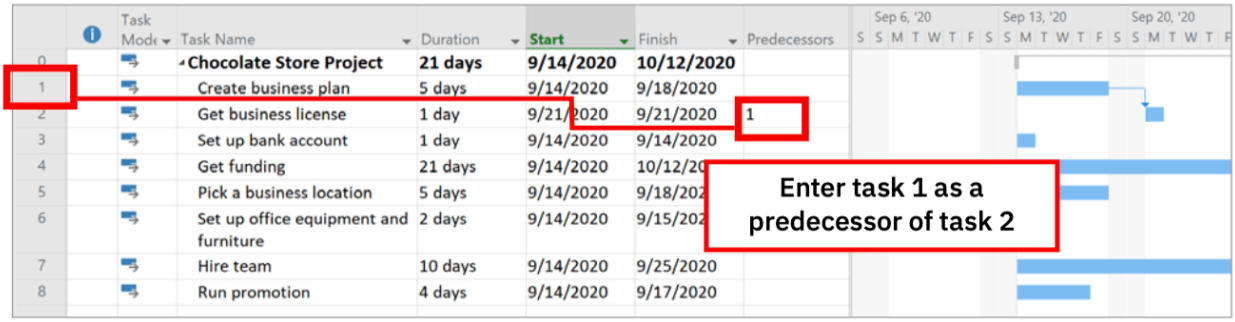
	Task Mode	Task Name	Duration	Start	Finish	Predeces
0		**Chocolate Store Project**	**21 days**	**9/14/2020**	**10/12/2020**	
1		Create business plan	5 days	9/14/2020	9/18/2020	
2		Get business license	1 day	9/14/2020	9/14/2020	
3		Set up bank account	1 day	9/14/2020	9/14/2020	
4		Get funding	21 days	9/14/2020	10/12/2020	
5		Pick a business location	5 days	9/14/2020	9/18/2020	
6		Set up office equipment and furniture	2 days	9/14/2020	9/15/2020	
7		Hire team	10 days	9/14/2020	9/25/2020	
8		Run promotion	4 days	9/14/2020	9/17/2020	

Рисунок 1.17 – Список створених задач

Також необхідно зазначити, що задачі мають виконуватися у певному порядку оскільки між ними є певний зв'язок. Так, відкриття банківського рахунку неможливе без наявності ліцензії на бізнес. Також при відсутності ліцензії на бізнес неможливо отримати фінансування. Існує два способи утворення зв'язку між задачами:

1) Вказати у кожній задачі яка задача має передувати їй;
2) За допомогою кнопки Link у меню.

Якщо ми хочемо вказати, що розробка бізнес-плану має бути завершена до отримання бізнес-ліцензії, то необхідно у колонці Predecessors для відповідного запису вказати номер запису задачі створення бізнес-плану. На рисунку 1.18 зображено вказання задачі яка має бути виконана перед обраною.



	Task Mode	Task Name	Duration	Start	Finish	Predecessors
0		**Chocolate Store Project**	**21 days**	**9/14/2020**	**10/12/2020**	
1		Create business plan	5 days	9/14/2020	9/18/2020	
2		Get business license	1 day	9/21/2020	9/21/2020	1
3		Set up bank account	1 day	9/14/2020	9/14/2020	
4		Get funding	21 days	9/14/2020	10/12/20	
5		Pick a business location	5 days	9/14/2020	9/18/202	
6		Set up office equipment and furniture	2 days	9/14/2020	9/15/202	
7		Hire team	10 days	9/14/2020	9/25/2020	
8		Run promotion	4 days	9/14/2020	9/17/2020	

Рисунок 1.18 – Вказання задачі яка передує обраній

Це був перший спосіб вказання задачі, що має бути виконана перед переходом до виконання обраної. Для вказання передуючих задач іншим скористаємося другим методом. Для цього необхідно виділити усі задачі окрім тої що має виконуватися першою та на вкладці Task натискаємо на кнопку посилання. На рисунку 1.19 зображена кнопка посилання.

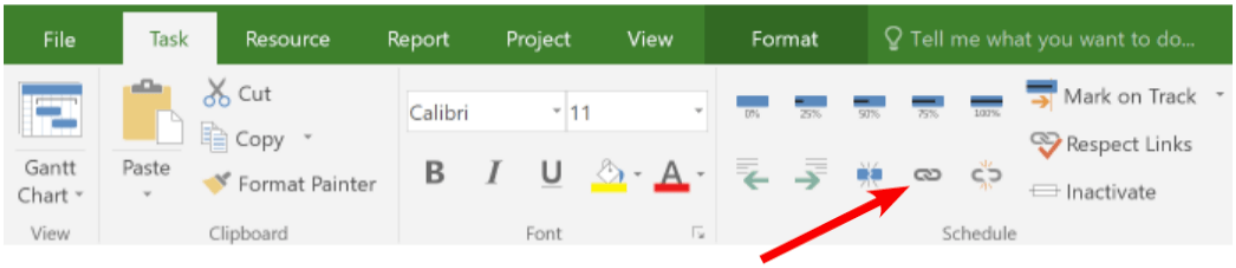
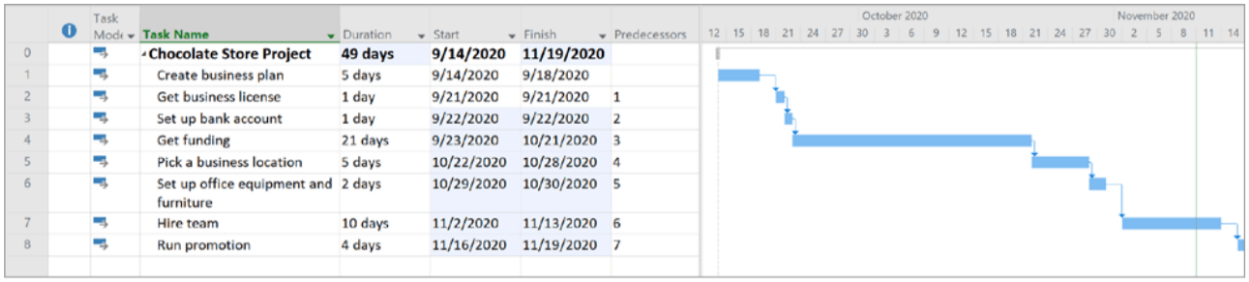


Рисунок 1.19 – Кнопка посилання

Після натискання кнопки буде автоматично побудовано діаграму Ганта. Варто відзначити, що на діаграмі Ганта у вихідні дні присутні пропуски, оскільки в налаштуваннях календарю було вказано, що вихідні дні не повинні бути робочими. На рисунку 1.20 зображено побудовану діаграму Ганта.



	Task Mode	Task Name	Duration	Start	Finish	Predecessors
0		Chocolate Store Project	49 days	9/14/2020	11/19/2020	
1		Create business plan	5 days	9/14/2020	9/18/2020	
2		Get business license	1 day	9/21/2020	9/21/2020	1
3		Set up bank account	1 day	9/22/2020	9/22/2020	2
4		Get funding	21 days	9/23/2020	10/21/2020	3
5		Pick a business location	5 days	10/22/2020	10/28/2020	4
6		Set up office equipment and furniture	2 days	10/29/2020	10/30/2020	5
7		Hire team	10 days	11/2/2020	11/13/2020	6
8		Run promotion	4 days	11/16/2020	11/19/2020	7

Рисунок 1.20 – Діаграма Ганта

Отже, далі буде проведено порівняння розглянутих систем.

1.4 Порівняння Trello, MS Project та Jira

Для порівняння Trello, MS Project та Jira було складено таблицю за основними характеристиками. У таблиці 1.4 наведено порівняння систем управління проектами Trello, MS Project та Jira за функціями.

Таблиця 1.4 – Порівняння систем управління проектами Trello, MS Project та Jira за функціями

Функції	Jira	MS Project	Trello
Керування задачами	+	+	+
Відстеження часу	+	+	–
Керування ресурсами	+	+	+
Формування звітів/графіків	+	+	–
Керування документами	+	–	+

Продовження таблиці 1.4

Відстежування та керування помилками	+	+	–
Перегляд дошки	+	–	+
Перегляд карт проектів	+	+	+
Наявність мобільного додатку	+	–	+

Інтеграції зі сторонніми сервісами. Сьогодні Trello налічує понад 30 можливих інтеграцій з іншими додатками, в той же час Jira забезпечує більше 100. Але варто зазначити, що MS Project взагалі не підтримує будь-які додатові інтеграції. Jira та Trello підтримують широко використовувані додатки, такі як: Slack, Evernote, Github, Bitbucket, Dropbox та інструменти Google. Також Trello та Jira мають великий набір плагінів, додатків та інструментів автоматизації [23].

Хостинг. Сьогодні у компанії є великий вибір, коли мова йде про зберігання даних. Одним з таких варіантів є використання хмари, а не власного серверу. У обох методів є свої переваги та недоліки. Вибір того чи іншого методу залежить від багатьох факторів. Але від обраного методу залежить яка система управління проектами буде використовуватись. Оскільки Trello доступна лише у якості хмарної служби, то розгорнути її на локальному сервері не має можливості. З іншої сторони, Jira пропонує використання хмари та надає можливість локального розгортання. Щодо MS Project, то даний додаток доступний у якості хмарної служби, а також пропонує локальну версію [24].

Також було проведено порівняння цін та складено таблицю, що наведена нижче.

Таблиця 1.5 – Порівняння цін за тарифними планами Trello, MS Project та Jira

Тарифний план	Jira в хмарі	MS Project в хмарі	MS Project локальна версія	Trello
Безкоштовний план	+, до 10 користувачів на сайт, 1 проект	–	–	+, до 10 дощок для робочого простору
Стандартний план за 1 користувача (за місяць)	7,5$	10$	680$ (за ліцензію)	5$

Продовження таблиці 1.5

Тарифний план 2 рівня	–	30$	1 130$ (преміум план)	–
Преміум план за 1 користувача (за місяць)	14,5$	50$	Вартість варіанту розміщення на власному сервері потребує узгодження	10$

Ціни на версію Jira, що розташовується у дата центрі починаються від 17,5 $ за користувача при загальній кількості користувачів більше 50 тисяч. Та зростає до 84 $ за користувача при 500 користувачах, що використовують сервіс. Ціни для тарифного плану Enterprise не вказуються [25].

Отже, якщо потрібен всеосяжний, повністю налаштовуваний інструмент управління командою та проектами для середньої компанії, то найкращим рішенням буде обрати систему Jira. При виборі Jira треба враховувати, що дана система потребує значного часу для налаштування та адаптування інструменту під задачі компанії. Але, після одного налаштування ви отримаєте потужний та зручний інструмент для управління проектами з великою кількістю функцій. В той же час якщо компанія невелика і потрібен простий у використанні інструмент, то варто обрати Trello. Дана система має мінімальний набір найнеобхідніших функцій. Щодо MS Project, то його основними недоліками є: складний інтерфейс, високі ціни, відсутність автоматизації робочого процесу. Але попри ці недоліки дана СУП має досить потужний функціонал і може підійти для команд, що працюють над складними проектами та можуть дозволити придбати ліцензію за вказаною ціною.

Висновки

В даному розділі було встановлено, що системи управління проектами поділяються відповідно до сфер застосування і можуть бути призначені для керування: будівельними, інвестиційними, організаційними проектами, проектами пов’язаних з розробкою та впровадження ПО та для інших вузькоспеціалізованих галузей. Також було встановлено, що сучасні СУП за технічними характеристиками поділяють на клієнт-серверні та хмарні системи.

Далі було проведено аналіз двох найпопулярніших систем управління проектами, а саме Jira, MS Project та Trello. В ході аналізу було встановлено, що Jira більше підходить для середніх та великих компаній, Trello – для малих. Також було встановлено, що Jira має три варіанти розміщення ПО: у хмарі в орендованому дата-центрі; клієнт-серверний додаток (з можливістю

розгортання на власному сервері для Enterprise клієнтів) та у хмарному середовищі. У свою чергу Trello пропонує версію лише з розміщенням у хмарному середовищі. При порівнянні вартості хмарних версій додатків встановлено, що Trello має дещо нижчу вартість, а саме на 2,5$ за одного користувача на місяць у стандартному тарифному плані та на 4,5$ у преміум плані. MS Project виявився найдорожчим серед конкурентів, має найбільше вбудованих функцій, більше підходить для складних проектів. MS Project має варіанти використання у хмарі, локальний варіант та варіант розміщення на власному сервері.

Отже, Trello та Jira пропонують обмежену кількість користувачів для безкоштовного користування. Також для отримання звітності про продуктивність команд у Jira треба додатково оплатити дану функцію, а Trello взагалі не має даного функціоналу. Окрім цього варто зазначити складність налаштування Jira та нагромаджений інтерфейс, що потребує витрат додатково часу для ознайомлення та адаптації з інструментом. У обох додатках відсутні сповіщення про наближення відведених термінів на виконання задач, але у Jira можна підключити дану послугу за додаткову плату. Варто зазначити, що у MS Project також відсутні сповіщення про дедлайни, але на відміну від конкурентів у даному ПЗ взагалі відсутня дана опція.

Таким чином, проаналізувавши переваги та недоліки додатків Jira, MS Project та Trello було прийнято рішення про розробку та впровадження власної системи контролю завдань (СКЗ) для промислових підприємств (ПП). Дана система матиме мінімально необхідний функціонал, що дозволятиме відслідковувати хід виконання завдань, визначати завантаженість відділів та вчасно реагувати на будь-які затримки у робочих процесах. Також дана система буде належати до відкритого програмного забезпечення (open source software) та поширюватися на безкоштовній основі.

Як було встановлено в даному розділі, система керування проектами значно спрощує планування завдань та пришвидшу їх виконання. Це відбувається за рахунок впровадження СКЗ у модель бізнес-процесів. Тому, в наступному розділі буде розглянуто, що являють собою бізнес-процеси та місце СКЗ в даних процесах.

2 МІСЦЕ СИСТЕМИ КОНТРОЛЮ ВИКОНАННЯ ЗАВДАНЬ В БІЗНЕС-ПРОЦЕСАХ

2.1 Класифікація бізнес-процесів підприємства та їх удосконалення

Для будь-якої компанії або ж підприємства основною ціллю її діяльності є отримання прибутку. Діяльність компанії складається з низки операцій та дій, які пов'язані з управлінням. Тому, в англомовній літературі застосовують поняття «business process»», що в перекладі означає «діловий процес». В даному контексті слово «бізнес» використовується для відокремлення процесів, які опосередковано або напряму відносяться до створення економічної цінності, від всіх інших видів процесів: математичних, хімічних, фізичних та інших, в залежності від специфіки діяльності компанії. Щодо процесів, то вони характеризуються за наступними властивостями:

– наявністю вхідних ресурсів;

– формуванням відповідного ресурсного забезпечення необхідного для виконання процесу;

– наявністю перехідних станів;

– наявністю вихідних результатів.

Отже, бізнес-процес можна назвати сукупністю цілеспрямованих взаємопов'язаних дій, що перетворюють входи (ресурси) у виходи (результати, продукт) які мають цінність для зовнішнього або внутрішнього споживача. Таким чином бізнес-процеси описують функції які виконує компанія для досягнення своїх цілей. На рисунку 2.1 зображено схему взаємозв'язку функцій і бізнес-процесів [26, 27].

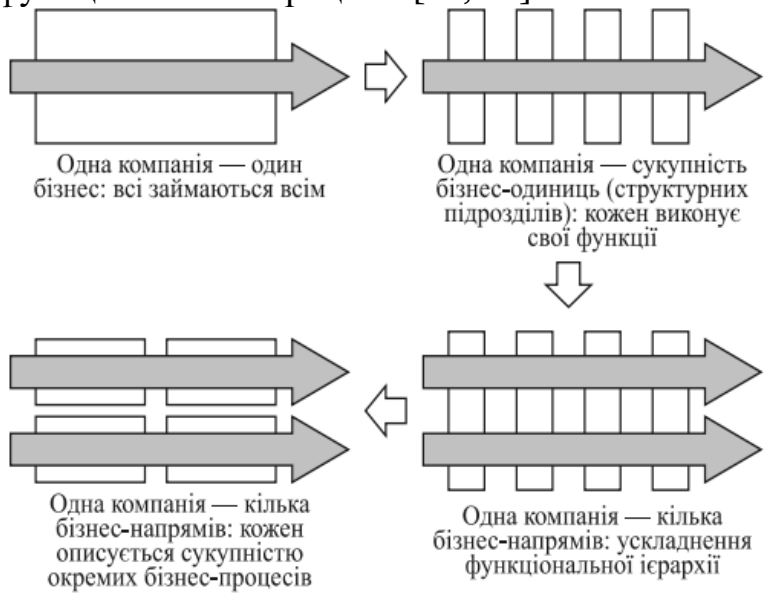


Рисунок 2.1 – Схема взаємозв'язку функцій і бізнес-процесів

Згідно вище сказаного та схеми взаємозв'язку функцій з бізнес-процесами можна зробити висновок, що для досягнення стратегічних цілей компаніям або підприємствам необхідно визначити найважливіші бізнес-процеси та провести їх удосконалення.

Для початку розглянемо класифікацію бізнес-процесів. Сьогодні на підприємствах застосовують процесний підхід. При даному підході діяльність підприємства розглядається як сукупність взаємодіючих процесів, що протікають всередині організаційної структури підприємства та реалізують його цілі. Використовуючи даний підхід керівники різних рівнів повинні звертати увагу на взаємодію учасників процесів, оскільки найбільші втрати інформації та часу відбуваються через їх невизначеність і призводять до економічних втрат. Таким чином підприємство отримує наступні переваги від використання процесного підходу:

–забезпечення умов для широкого делегування задач разом з відповідальністю за їх виконання, що сприяє підвищенню якості продукції і процесів;

–кожен працівник прив'язаний до кінцевого результату та відповідає за його якість;

–зменшення кількості рівнів прийняття рішень, що дозволяє підвищити оперативність та адаптивність діяльності підприємства;

–створення умов для автоматизації виконання бізнес-процесів.

Основою реалізації процесного підходу є класифікація бізнес-процесів за певними ознаками. Дану класифікацію наведено на рисунку 2.2 [26, 28].

Рисунок 2.2 – Загальна класифікація бізнес-процесів

До основних бізнес-процесів відносять процеси, що орієнтовані на надання послуг та виробництво продукції які становлять цінність для клієнта, та є джерелом доходів підприємства.

Допоміжні бізнес-процеси це ті, що описують адміністративно-господарське забезпечення, бухгалтерський облік та фінансову діяльність, підготовку персоналу, ІТ-забезпечення та інші подібні процеси.

Управлінські бізнес-процеси це специфічний вид допоміжних процесів. Вони забезпечують функціонування допоміжних та основних бізнес-процесів, регулюють поточну діяльність підприємства та впливає на його конкурентоспроможність.

Ціль бізнес-процесів розвитку є забезпечити розвиток підприємства у довгостроковій перспективі та формування доданої вартості. Також дані бізнес-процеси можна назвати центрами формування інвестицій. До таких процесів часто відносять бізнес-проекти, що складаються з проведення автоматизації, реінжинірингу, реструктуризації та інших проектів, що покращують розвиток підприємства [28].

Розглядаючи основні і допоміжні бізнес-процеси в контексті функціонування промислових підприємств можна зробити висновок про інноваційність даного підходу.

Отже, налагодження та покращення бізнес-процесів є головною задачею для успішного існування підприємства. Далі буде розглянуто систему контролінгу.

2.2 Система контролінгу на підприємстві

На сьогоднішній день успіх підприємницької діяльності значною мірою залежить від гнучкості менеджменту організації та його здатності передбачати виникнення кризових ситуацій. На рисунку 2.3 наведено причини виникнення кризових ситуацій.

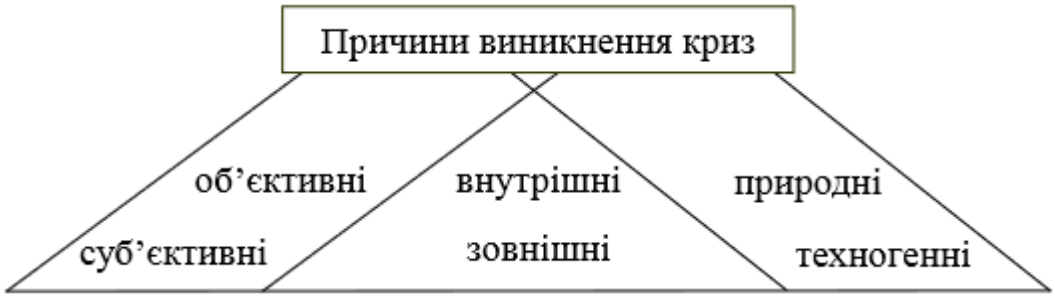


Рисунок 2.3 – Причини виникнення кризових ситуацій за різними ознаками

При виникненні загрози кризової ситуації розв'язувати проблеми управління необхідно за допомогою використання антикризового та ситуаційного менеджменту, та контролінгу [29].

Контролінг є основою управління бізнес-процесів. За допомогою контролю суб'єкт управління отримує актуальну інформацію про стан керованої системи, що дає змогу вчасно приймати управлінські рішення з

усунення причин які перешкоджають досягненню успіхів або ж рішення для закріплення досягнутого результату.

Таким чином контролінг є елементом антикризового управління, що спроможне попередити або є пом'якшити наслідки кризової ситуації на підприємстві. Також в кризових ситуаціях контролінг мінімізує витрати в період кризи та сприяє виведенню компанії з неї. На рисунку 2.4 наведено роль контролінгу в системі антикризового управління [29, 30].

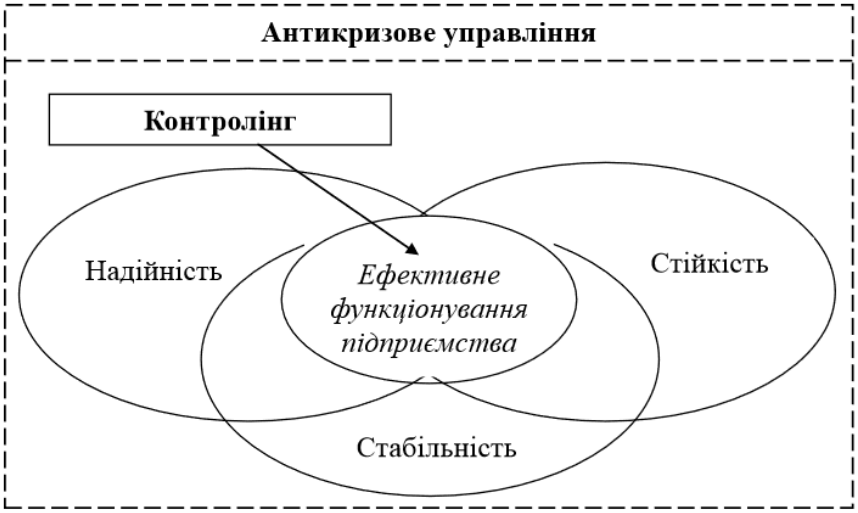


Рисунок 2.4 – Роль контролінгу в системі антикризового управління

Процес контролінгу передбачає:

– отримання оцінки результату діяльності підприємства загалом, результатів діяльності її окремих структурних підрозділів;

– визначення ступеню гнучкості управління та його здатності приймати адекватні рішення при зміні зовнішнього середовища та здатності пристосування до нього;

– аналіз і виявлення факторів перешкоджаючих досягненню запланованих результатів;

– оцінка внесків кожного працівника у забезпеченні кінцевих результатів;

– зворотній зв'язок для мотивації та інформування персоналу;

– мінімізацію витрат;

– вирішення проблем, що перешкоджають зростанню підприємства тощо.

Контролінг є актуальним через своєчасне виявлення відхилень у роботі, що зумовлені впливом певних чинників зовнішнього середовища або низькою виконавчою дисципліною, та внесенні коректив які створюють належні умови для реалізації цілей підприємства [29, 30].

Контролінг – це якісно новий рівень управління, він виступає в ролі саморегулюючого механізму на підприємстві, що забезпечує зворотній зв'язок у контурі управління. Згідно класифікації складеної М. Месконом, М. Альберта, та Ф. Хедоурі контролінг виконує 4 функції менеджменту, а

саме: планування, організація взаємодії, мотивація та контроль. Залежно від функції управління роль контролінгу наступна:

1. При плануванні система контролінгу надає інформацію для складання та розробки планів, перевірки планів, що складені окремими структурними підрозділами;
2. При організації взаємодії роль контролінгу полягає у пошуку шляхів налагодження обліку та контролю витрат, а також веденні статистики з результативності підрозділів підприємства;
3. Для мотивації підлеглих розробляються механізми стимулювання виконання поставлених завдань;
4. Функція контролю виконує систематичне спостереження за об'єктами підприємства та процесами його діяльності, регулярну перевірку досягнутих результатів та зіставлення з запланованими, при виникненні обставин що перешкоджають досягненню цілей виконує пошук варіантів вирішення зменшення негативного впливу та пропозицій з усунення перешкод.

На рисунку 2.5 зображено роль контролінгу у контурі управління [29, 31].

Рисунок 2.5 – Роль контролінгу у контурі управління.

Контролінг прийнято поділяти за двома основними видами: стратегічний та оперативний (або тактичний). На рисунку 2.6 зображено види контролінгу та його призначення за рівнями управління.

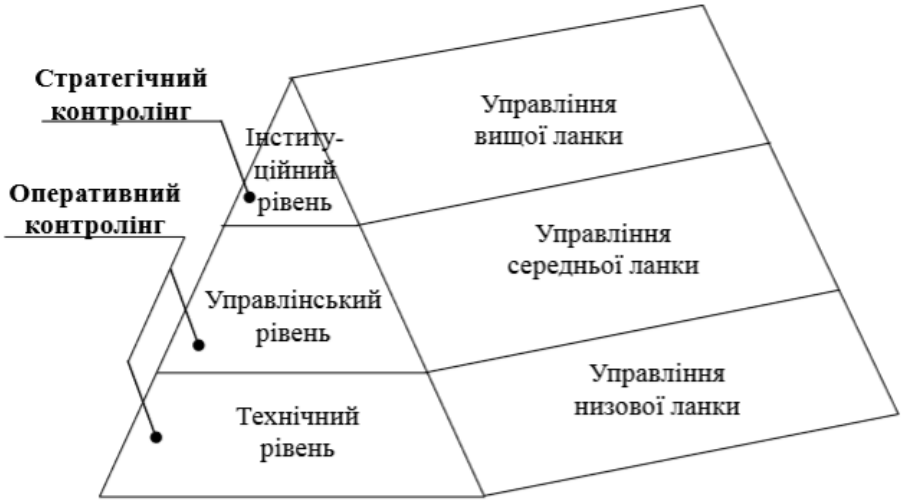


Рисунок 2.6 – Види контролінгу та його призначення за рівнями управління

Згідно рисунку 2.6 стратегічний контролінг обслуговує інституційний рівень управління. Даний вид контролінгу призначений для моніторингу

діяльності підприємства у довгостроковій перспективі та аналізу внутрішнього і зовнішнього середовища. Таким чином головна ціль стратегічного контролінгу є створення системи управління для відслідковування руху підприємства до його стратегічних цілей [31].

Для постановки стратегічних цілей необхідно провести попередній аналіз зовнішніх та внутрішніх умов функціонування суб'єкта господарювання. На рисунку 2.7 зображено складові для аналізу в системі стратегічного контролінгу [31].

Рисунок 2.7 – Складові для аналізу в системи стратегічного контролінгу

Оперативний контролінг обслуговує управлінській та технічний рівні, відповідає за дослідження економічної ефективність і рентабельність діяльності підприємства. На рисунку 2.8 зображено цілі та завдання оперативного контролінгу [31, 32].

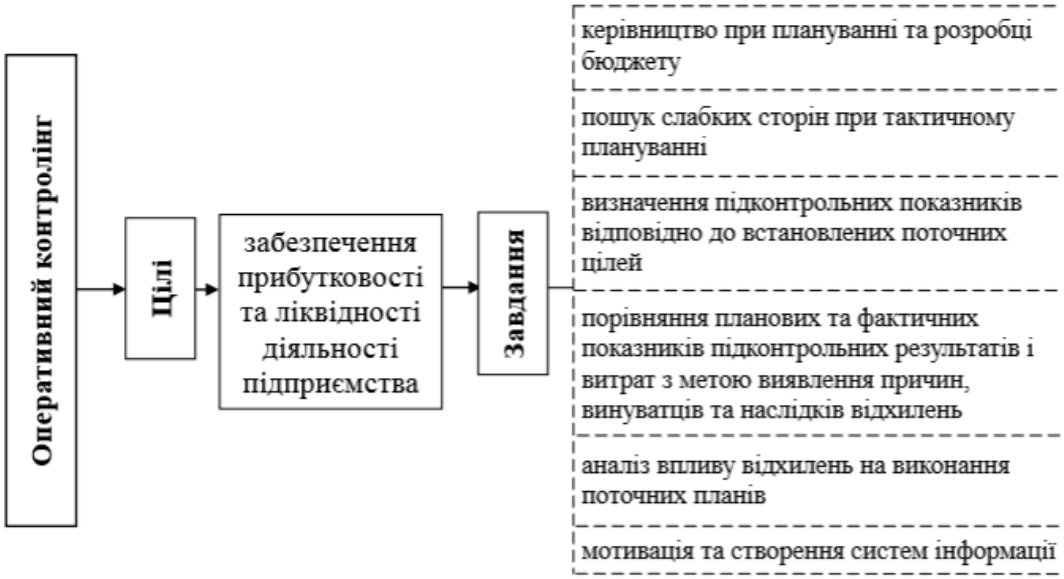


Рисунок 2.8 – Цілі та завдання оперативного контролінгу

Так, згідно рисунку 2.7 в систему стратегічного контролінгу входить проведення науково-дослідних та конструкторських робіт, що сприяє підвищенню інноваційної діяльності підприємств в розрізі контролінг бізнес-процесів.

Українським інститутом інтелектуальної власності, станом на 01.12.2022 р., зазначено , що з 1992 року в нашій країні здійснено реєстрацію 659 562 об'єктів промислової власності, градацію яких представлено графічно на рисунку 2.9 [33].

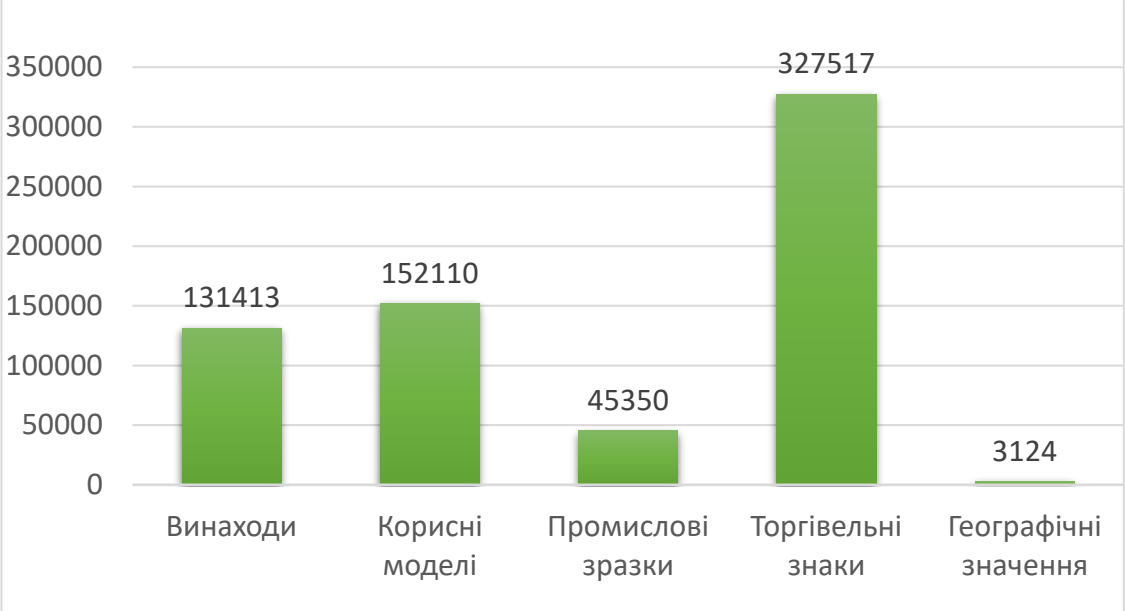


Рисунок 2.9 – Об'єкти промислової власності

Як видно з рисунку 2.9 забезпечення інноваційної діяльності підприємств підтримується за рахунок: винаходів, корисних моделей, промислових зразків, товарних знаків. Впровадження і розробка програмного забезпечення відноситься до винаходів. Так, згідно звіту «Global Innovation Index 2020» Всесвітньої організації інтелектуальної власності (ВОІВ) серед інноваційної діяльності 131 країни та економіки світу Україна 45 місце [34].

Так як функціонування сучасних промислових підприємств базується на перетині ретельно розроблених моделей поведінки та бізнесу. Цифрові продукти (інновації) є вираженням цього перетину та найважливішим засобом досягнення значущих і прибуткових результатів для компаній та їхніх клієнтів.

Рішення щодо провадженні інноваційної діяльності відноситься до класу комплексних та складних і повинно прийматись на основі зважування всіх «ТАК» і «НІ» в діяльності підприємства.

Схему оцінки інноваційної діяльності підприємства можна зобразити за допомогою блок-схеми (рисунок 2.10)

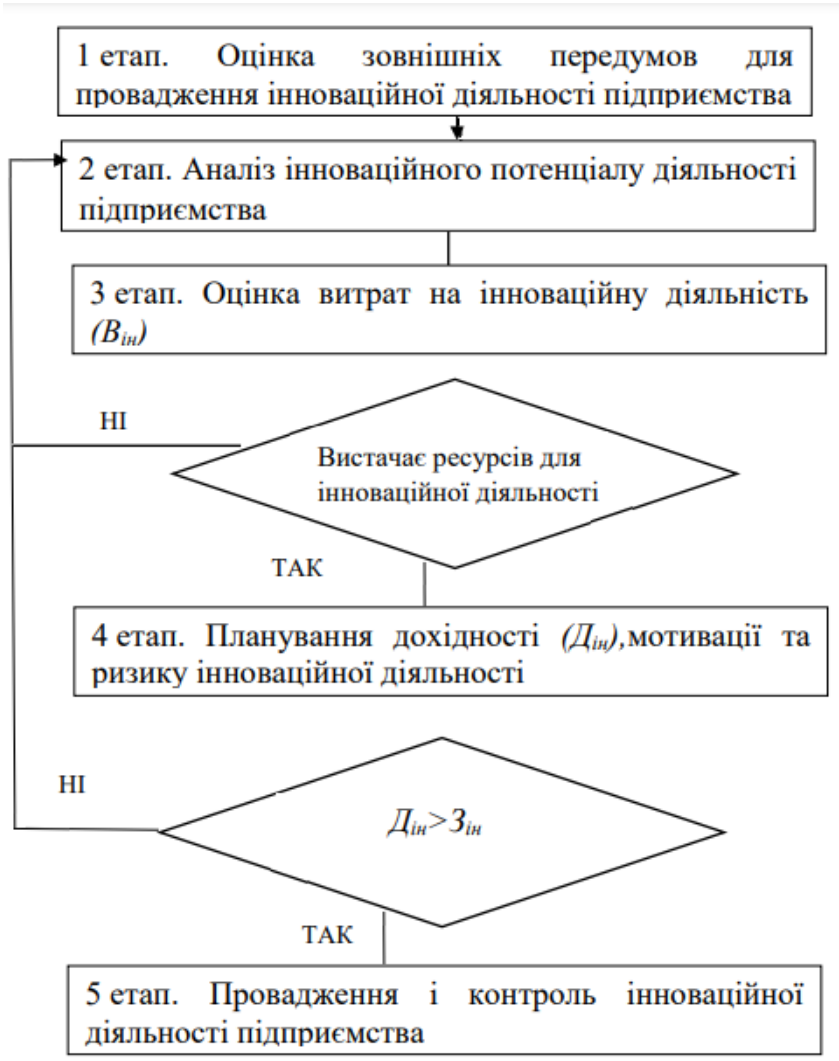


Рисунок 2.10 – Схема оцінки впровадження діяльності підприємства

Виходячи з схеми оцінки інноваційної діяльності можна визначити проблеми управління інноваційною діяльністю підприємства:

1. На першому етапі потрібно визначити нормативні акти, державні програми підтримки інноваційної діяльності, визначити конкурентів в інноваційній діяльності.

Головною проблемою є брак державного фінансування. Крім того, відсутність досліджень та прогнозування інструментів державного регулювання інноваційного розвитку економіки.

2.3 Контроль показників ефективності

Як було вже зазначено вище, контролінг використовується для своєчасного виявлення відхилень у роботі, що можуть бути зумовлені низькою виконавчою дисципліною. Для контролю виконавчої дисципліни використовують «key performance indicator», що в переводі з англійської означає – ключові показники ефективності (KPI) або ж, більш коректний варіант – ключові показники результатів діяльності. Виходячи з назви, KPI це такий індикатор, що показує робочий прогрес окремого співробітника або

відділу по досягненню певної поставленої цілі. Суть даних показників полягає у відображенні близькості фактичних показників ефективності до розрахункових, чим ближче – тим краще [35].

Залежно від галузі використовуються різні ключові показники результативності діяльності. Наприклад, KPI для співробітників по управлінню проектами оцінюватиме:

– продуктивність;
– дотримання встановлених термінів;
– частку завершених або незавершених в термін проектів;
– рентабельність інвестицій.

Ключові показники результативності діяльності поділяють на:

⁻запізнілі – вони відображають результати діяльності після її закінчення, одними з таких показників є фінансові;

⁻оперативні – вони відображають поточну діяльність робітників, підрозділів або компанії в цілому і дозволяють оцінити можливість виникнення ризиків (наприклад, необхідність перенесення дедлайну або ж необхідність збільшення фінансових витрат).

Для управління бізнес-процесами використовують наступні KPI:

– результату – відображає який результат і в якому обсязі було отримано;
– функціонування – відображає на скільки відповідає виконуваний бізнес-процес необхідному алгоритму робіт з його виконання, в якому обсязі виконані вимоги;
– ефективності – відображає співвідношення отриманих результатів до витрачених ресурсів;
– продуктивності – відображає співвідношення отриманих результатів до витраченого часу;
– витрат – показує обсяг витрачених ресурсів.

Підприємство впроваджуючи KPI може отримати такі позитивні зміни:

– Отримання можливості визначати ефективність кожного з відділів або є працівників та на підставі отриманих результатів вживати заходи по покращенню даних показників або ж їх збереженню;
– В процесі впровадження даних показників визначаються головні фактори від яких залежить успішність компанії;
– Визначаються реалістичні вимоги та нормативи;
– Отримання оперативних даних про поточний стан робочих процесів та можливість їх корегування залежно від пріоритетів.

Усі перелічені переваги від впровадження KPI зводяться до збільшення ефективності підприємства, покращенню морального стану співробітників за рахунок встановлення чітких вимог до їх роботи, збереженні рейтингу за рахунок виконання робіт у поставлені терміни [35].

Оскільки єдиного підходу до вирішення усіх проблем не існує, то для отримання перелічених переваг важливо правильно визначити ключові

показники результатів діяльності. Оцінка ефективності діяльності співробітників повинна слугувати інструкцією з подальших безпомилкових дій. Така система повинна відповідати наступним критеріям:

- не містити показників, що не можуть бути чітко виміряними;
- не містити показників які неможливо співвіднести з прибутком;
- використовувати метрики на які співробітників може вплинути;
- оцінювати ступінь конкретних досягнутих результатів;
- система має мотивувати співробітників для підвищення ефективності їх роботи та усіх відділів загалом [36].

Отже, ключові показники результатів діяльності є ефективним інструментом для прийняття стратегічних, тактичних та оперативних рішень, що спираються на аналітичні дані та реальні результати. При правильному підході до використання KPI зростає швидкість досягнення стратегічних цілей підприємства.

Висновки

В даному розділі було визначено, що бізнес-процес це сукупність цілеспрямованих взаємопов’язаних дій, що перетворюють входи (ресурси) у виходи (результати) які мають цінність для зовнішнього або внутрішнього споживача. Також було встановлено, що бізнес-процеси поділяють на основні, допоміжні, управлінські та бізнес-процеси розвитку. Управлінські бізнес-процеси це специфічний вид допоміжних процесів. Вони забезпечують функціонування допоміжних та основних бізнес-процесів, регулюють поточну діяльність підприємства та впливає на його конкурентоспроможність.

Також в даному розділі було розглянуто систему контролінгу на підприємстві та визначено, що вона поділяється на два види: стратегічний та оперативний. Було визначено, що система контролінгу є основою управління бізнес-процесів. За допомогою контролю підприємство отримує актуальну інформацію про стан керованої системи, що дає змогу вчасно приймати управлінські рішення з усунення причин які перешкоджають досягненню успіхів або ж рішення для закріплення досягнутого результату.

Далі в даному розділі було розглянуто контроль показників ефективності, що застосовуються в оперативному контролінгу. Було визначено що для управління бізнес-процесами використовують KPI: результативності, ефективності, продуктивності, функціонування. Було визначено основні вимоги до системи контролю ключових показників результатів діяльності.

В наступному розділі буде описано розробку системи контролю виконання завдань на підприємстві.

3 РОЗРОБКА, ОПИС ТА ТЕСТУВАННЯ СИСТЕМИ КОНТРОЛЮ ВИКОНАННЯ ЗАВДАНЬ

3.1 Опис підприємства ПАТ «АрселорМіттал Кривий Ріг»

Одним із важливих чинників подолання кризи в економіці України є належне забезпечення потреб держави корисними копалинами та їх ефективне використання, тому цей напрямок є одним із пріоритетних згідно з «Загальнодержавною програмою розвитку мінерально-сировинної бази України на період до 2030 року». Так як мінеральна база - це сукупність розвіданих і оцінених запасів корисних копалин і пов'язаних з ними компонентів, які можуть експлуатуватися в галузях народного господарства за умови досягнення економічних вигід на рівні, достатньому для здійснення розширеного виробництва з метою забезпечення економічної безпеки [37].

Так, саме основною складовою розвитку цієї програми є корисні запаси категорії А: добування залізних руд, виробництво коксу та коксопродуктів та виробництво чавуну сталі та феросплавів, що є основною діяльністю ПАТ "АрселорМіттал Кривий Ріг", а саме - види корисних копалин, які інтенсивно видобуваються в Україні, мають значні розвідані запаси корисних копалин та компонентів і експортуються або можуть розглядатися як такі для забезпечення короткострокових валютних надходжень та надходжень до державного бюджету. Підприємство займає 6 місце із 100 підприємств в Україні за версією журналу Forbes, і має прибуток 741 млн.грн.

На підприємстві середньооблікова численність працівників в 2020-2021 рр. складає – 19469 осіб.

На ПАТ «АрселорМіттал Кривий Ріг» протягом 2015-2020 рр. спостерігалися такі тенденції:

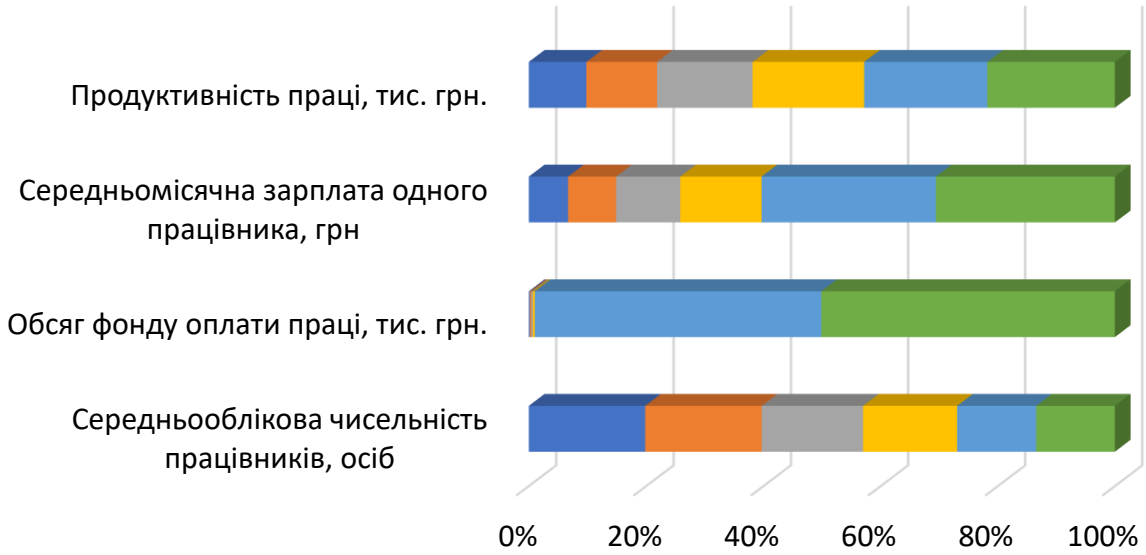
– в відбулося скорочення середньооблікової чисельності працівників на 9262 осіб ;

– спостерігається о збільшення фонду оплати праці на 4607388 тис. грн. або в 2,5 рази в 2020 році в порівнянні з 2015 роком;

– варто зазначити, що у 2020 р. порівняно з 2015 р., відбулося зростання середньомісячної заробітної плати працівників;

– протягом 5 років відбулося зростання продуктивності праці працівників на 2,41 тис. грн. або на 93,56% за рахунок збільшення обсягу виручки, що свідчить про правильну організацію праці на підприємстві.

На рисунку 3.1.1 зображено діаграму та таблицю показників ефективності персоналу в ПАТ «АрселорМіттал Кривий Ріг» протягом 2015-2020 рр.



	Середньооблікова чисельність працівників, осіб	Обсяг фонду оплати праці, тис. грн.	Середньомісячна зарплата одного працівника, грн	Продуктивність праці, тис. грн.
2015	28731.00	17962.00	5209.80	2.57
2016	28731.00	21937.00	6362.70	3.15
2017	25012.00	25485.00	8490.90	4.26
2018	23256.00	30027.00	10759.60	4.98
2019	19504.00	4505589.00	23100.00	5.50
2020	19469.00	4625350.00	23757.00	5.70

Рисунок 3.1.1 – Показники ефективності персоналу в ПАТ «АрселорМіттал Кривий Ріг» протягом 2015-2020 рр.

Організаційна структура підприємства представлена на рисунку 3.1.2

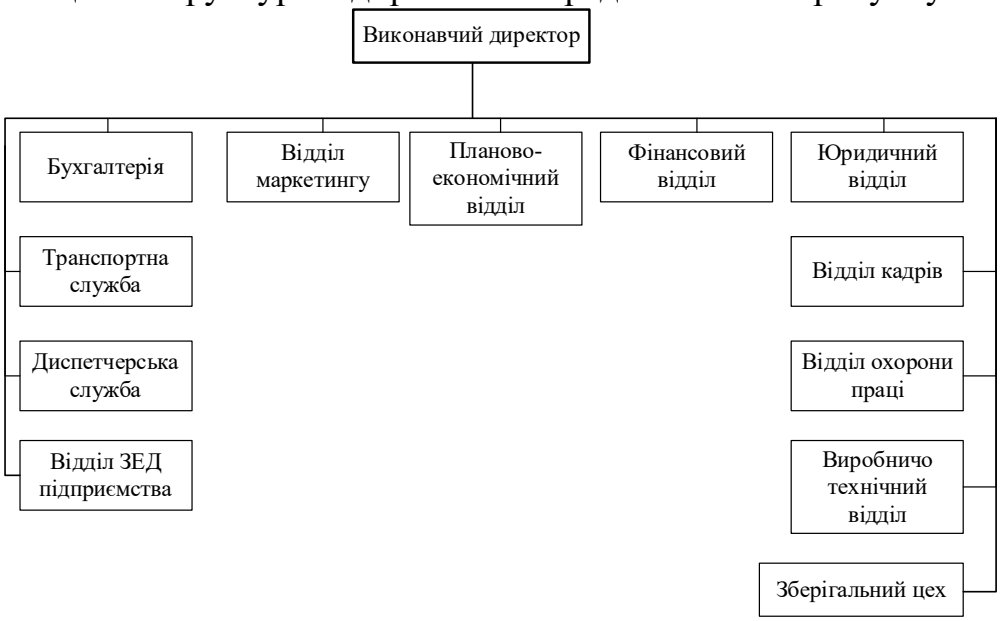


Рисунок 3.1.2 – Організаційна структура підприємства ПАТ «АрселорМіттал Кривий Ріг»

Інформаційною базою для здійснення оцінки діяльності стала звітність ПАТ «АрселорМіттал Кривий Ріг» за 2019 – 2020 роки є:

– форма № 1 «Баланс»;

– форма №2 «Звіт про фінансові результати»;

– довідкові дані бухгалтерії, планово-економічного та фінансового відділу;

– чинні нормативні-правові акти законодавства України, зокрема Господарський кодекс України, Закон України «Про бухгалтерській облік та фінансову звітність в Україні» та інші.

Проаналізуємо обсяги виробництва продукції в ПАТ «АрселорМіттал Кривий Ріг» – табл. 3.1.1.

Таблиця 3.1.1 – Динаміка виробітку продукції ПАТ «АрселорМіттал Кривий Ріг» 2018-2020 рр.

Види продукції	2018	2019	2020
Видобуток залізної руди	405540	4097915	4468 000
Виготовлення готової металопродукції	48526,8	3368473	3417000
Всього	454066	7466388,2	7885000

Збільшення обсягів виробництва продукції ПАТ «АрселорМіттал Кривий Ріг» призводить до покращення фінансового стану підприємства, табл. 3.1.2.

Таблиця 3.1.2 – Аналіз основних показників діяльності ПАТ «АрселорМіттал Кривий Ріг» за 2019-2020 рр., тис.грн

Назва показників	Роки		Відхилення
	2020	2019	2020/2019
Чистий дохід від реалізації (товарів, робіт і послуг)	63496684	62409226	+1087458
Собівартість реалізованої продукції,	58845530	63503128	-4657598
Фінансовий результат від операційної діяльності	2066296	-2750338	-684042
Чистий фінансовий результат	740902	-2265232	-1524330
Дебіторська заборгованість	22080986	19143229	+2937757
Кредиторська заборгованість	2209245	2581278	-372033
Матеріальні затрати	60244083	66244089	-6000006

Так, за даними проведеного аналізу, спостерігаємо, що чистий дохід від реалізації товарів, робіт і послуг (без ПДВ) у 2020році склав 63496684 тис.грн., що на 1087458 тис.грн (1,74%) більше чистого доходу 2019 року.

Собівартість реалізованої продукції у 2020 році зменшилась на 7,3% (4657598 тис.грн.)

Результатом операційних діяльності підприємства впродовж 2019 року є збиток, а в 2020 році - є прибуток.

Чистий фінансовий результат (збиток) після сплати податку на прибуток 2019 році склав -2265232 тис.грн., в 2020 році цей показник зріс на 2,7%.

Дебіторська заборгованість перед комбінатом на 01.01.2020 року складала 22080986 тис.грн, що на 2937757 тис.грн. більше ніж за минулий рік.

Кредиторська заборгованість за товари (роботи, послуги) впродовж досліджуваного року зменшилась на 372033 тис.грн.

Впродовж 2019-2020 років, комбінат повністю відмовився від кредитів банків.

За ці роки комбінат своєчасно та в повному обсязі здійснював поточні платежі до бюджету, в Пенсійний фонд та фонди соціального страхування, своєчасно сплачував заробітну плату працівникам комбінату.

Про те як видно з показників діяльності підприємства для покращення фінансового стану підприємства потрібна достовірна оцінка бізнес-процесів що на ньому відбуваються в режимі реального часу з використання ІТ технологій.

Зі зростанням динаміки економічних і соціальних процесів, що відбуваються на підприємстві, швидкими змінами на внутрішньому та зовнішньому ринках залізорудної продукції зростає роль управління бізнес-процесами на підприємстві. Тому, далі буде описано процес розробки СКВЗ, задача якого є нормалізування та покращення перебігу бізнес-процесів.

3.2 Веб-сервер

Поняття веб-серверу використовують як для програмного забезпечення так і апаратного, або, навіть, одночасно до обох частин, що працюють одночасно.

У випадку апаратного забезпечення веб-сервер являє собою електронно-обчислювальну машину або за сучасною термінологією – комп’ютер. Зазвичай, для веб-серверів використовуються спеціалізовані комп’ютери, а саме серверні. Хоча у ролі серверу може і виступати звичайний комп’ютер зі встановленим спеціальним програмним забезпеченням (ПЗ). Такого комп’ютера вистачить для організації невеликої мережі. Але для більш навантажених мереж необхідно аби комп’ютер вам спеціальне апаратне забезпечення, а саме: серверну материнську плату (дозволяє встановлювати декілька процесорів), спеціалізовану оперативну пам’ять (розрахована на великі навантаження, безперервного функціонування протягом тривалого часу, має високу відмовостійкість, спеціальний захист від пошкодження та втрати інформації), спеціалізований процесор, накопичувачі (відповідного типу залежно від призначення системи), належну систему вентиляції та охолодження внутрішніх компонентів [38].

Оскільки передбачається, що сервер має працювати у безперервному режимі, то відмовостійкість оперативної пам’яті є не менш важливим фактором ніж її ефективність. Тому, серверна DDR (Double Data Rate) пам'ять має певні спеціальні технічні рішення. Для серверів встановлюється спеціальна реєстрова пам’ять. Така пам'ять має додатковий чіп на платі реєстрової пам’яті для буферизації даних. Даний чіп використовується для автоматичної корекції помилок ECC (Error Correcting Code). Таким чином на кожні 8 звичайних чіпів на платі оперативної пам’яті присутній додатковий 1 буферний. На рисунку 3.2.1 зображено модуль ECC RAM та No-ECC RAM [39].

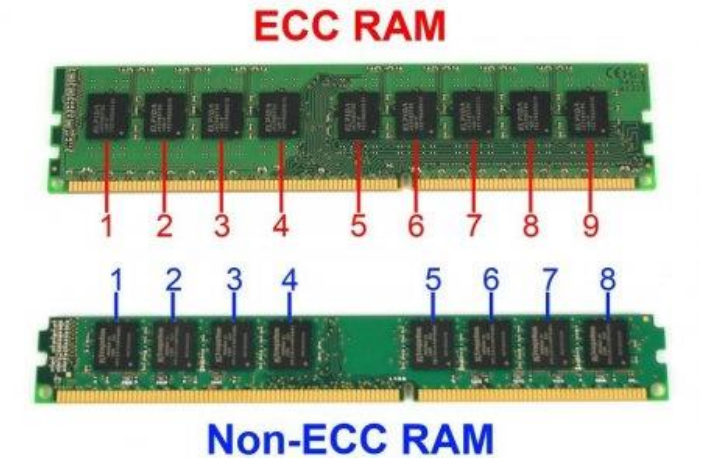


Рисунок 3.2.1 – Модуль ECC RAM та No-ECC RAM

Веб-сервер з програмної точки зору забезпечує зв’язок між користувачем та сервером. Така програма включає декілька компонентів, що відповідають за контроль доступу до файлів, що розміщені на сервері. Оскільки в даній роботі розглядається створення веб-додатку, то розглянемо принцип отриманні веб-сторінки.

Для отримання веб-сторінки браузер надсилає запит до веб-серверу за допомогою HTTP-протоколу (HyperText Transfer Protocol). Сервер після отримання даного запиту знаходить потрібний документ та надсилає його. У випадку якщо сервер не знайшов необхідний документ, то відправляється повідомлення про помилку з номером 404. На рисунку 3.2.2 зображено спрощену схему зв’язку клієнт-сервер через HTTP [40].

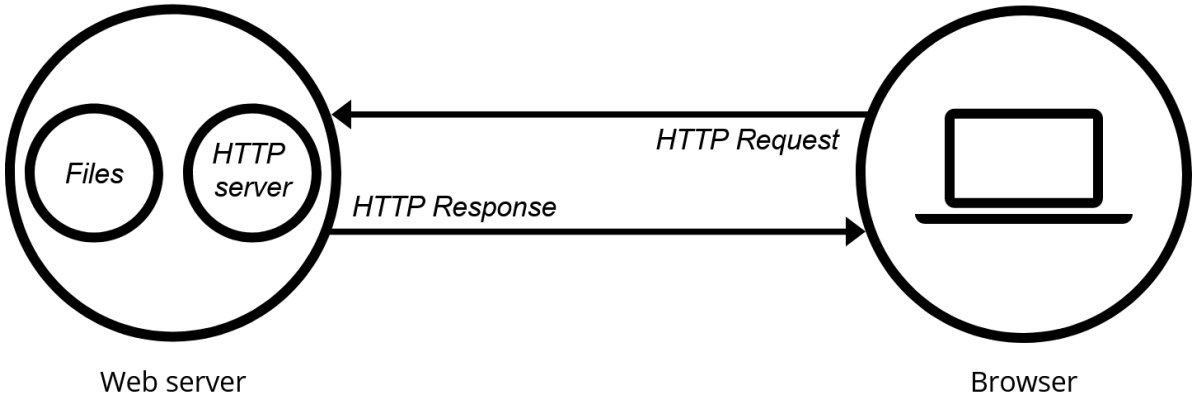


Рисунок 3.2.2 – Спрощена схему зв’язку клієнт-сервер через HTTP

Насправді при надсиланні HTTP запиту необхідно задіяти додаткові служби та сервіси. Розглянемо виконання HTTP запиту для отримання веб-сторінки покроково:

1) Після надсилання користувачем URL (Uniform Resource Locator) адреси запитуваного сайту браузер надсилає його до DNS-серверу (Domain Name System);

2) DNS-сервер шукає IP-адресу (Internet Protocol address), що відповідає даному URL;

3) Далі браузер встановлює TCP (Transmission Control Protocol) з‘єднання з веб-сервером.

4) Якщо запит виконується за допомогою протоколу HTTPS (HyperText Transfer Protocol Secure), то над TCP встановлюється TLS (Transport Layer Security, наслідник SSL – secure sockets layer, шар захисних соетів) з’єднання;

5) Формується HTTP запит та відправляється до сервера;

6) Браузер отримує HTTP-відповідь від сервера, що містить сторінку;

7) Зачиняється з’єднання (для HTTP/1.0);

8) Браузер виконує парсинг документа та відображує строінку [41].

Отже, схема зв’язку клієнт-сервер за допомогою HTTP виглядатиме так:

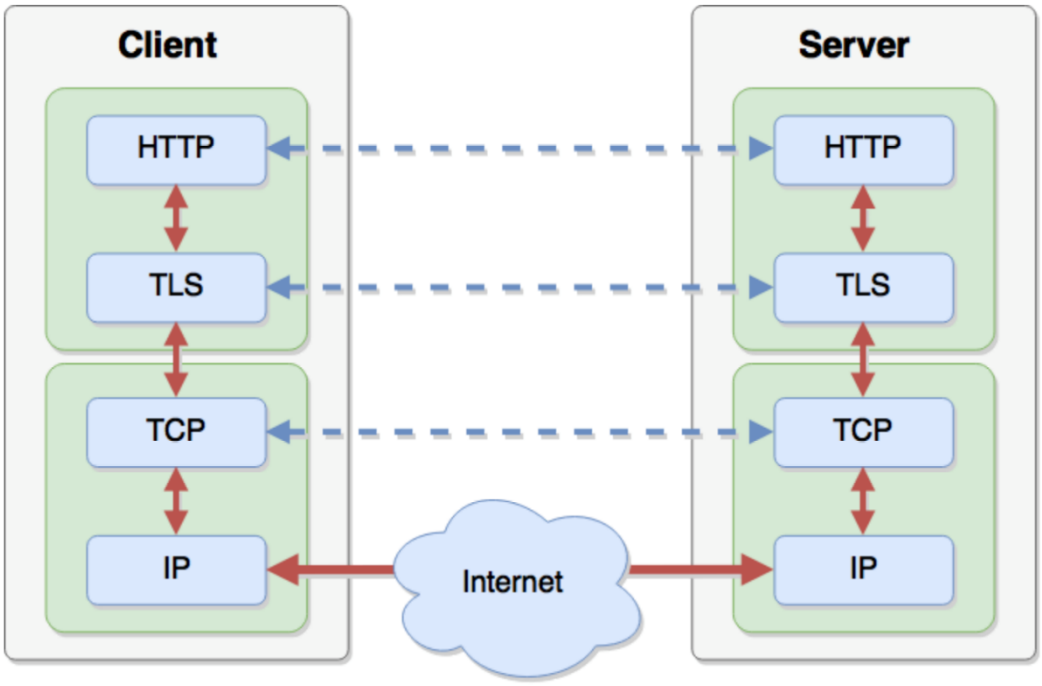


Рисунок 3.2.3 – Схема зв’язку клієнт-сервер за допомогою HTTP

Синіми пунктирними лініями позначені не фактичні виклики, оскільки неможливо напряму звернутися від клієнта до HTTP-серверу. Для цього необхідно спускатися ієрархією протоколів до IP.

Щодо DNS – це сервер, що надає доступ до бази даних доменних імен. DNS сервер зберігає таблиці співставлень імен хостів з їх IP-адресами. Оскільки при кожному запиті необхідне перетворення імені на ip-адресу, то такі сервери є надзвичайно навантаженими. Для зменшення навантаження було створено розподілену систему DNS серверів. За допомогою розподіленої системи організовано делегування зон відповідальності серверів.

Для пошуку IP-адреси сайту зазвичай використовують декілька DNS-серверів. На рисунку 3.2.4 зображено запит IP-адреси сайту chrome.google.com [41, 42].

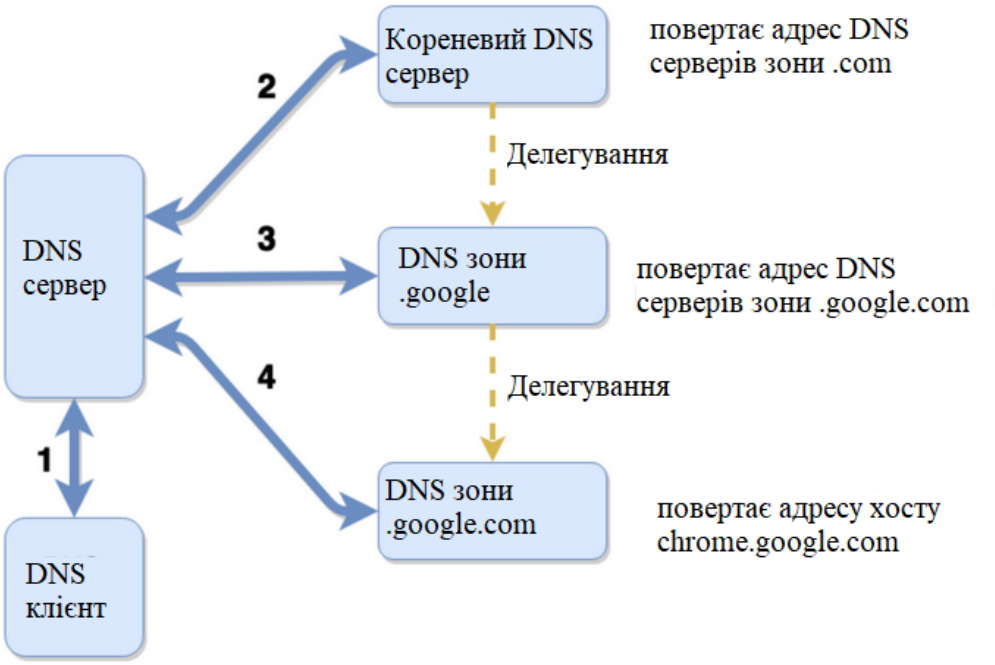


Рисунок 3.2.4 – Отримання IP-адреси сайту chrome.google.com

Кожен браузер має DNS-клієнт, що надсилає запити до DNS-серверу. Розглянемо отримання IP-адреси по крокам:

1) Після вводу доменного імені браузер в першу чергу перевіряє локальний файл hosts.txt, що зберігається на вашому комп'ютері. Якщо необхідна адреса відсутня в даному файлі, то далі браузер звертається до DNS-серверу інтернет-провайдера;

2) Якщо DNS-сервер інтернет-провайдера не містить шуканої адреси, то браузер надсилає запит до кореневого DNS-сервера (адресу якого надіслав сервер провайдера), що видає DNS-сервери доменів першого рівня, наприклад, ORG, COM, NET, UA;

3) Далі відбувається запит до DNS-сервера зони домену другого рівня (.google);

4) Та DNS-сервер .google.com повертає шукану адресу [42].

Веб-сервери бувають динамічними та статичними. Статичним веб-сервером називають той, що лише надсилає потрібні файли до браузера. До динамічних веб-серверів відносять ті, що мають спеціальне додаткове програмне забезпечення, яке може змінювати вихідні файли перед відправкою до браузера. Тобто, перед відправкою можуть здійснюватися запити до баз даних, проводитися додаткові обчислення та генерується відповідь [43].

Далі розглянемо деякі популярні веб-сервери, а саме: nginx, Apache, та IIS. Ці веб-сервери є одними з найпоширеніших у світі згідно більшості статистик. Статистику використання веб-серверів від W3Techs наведено на діаграмі 3.2.5 [44].

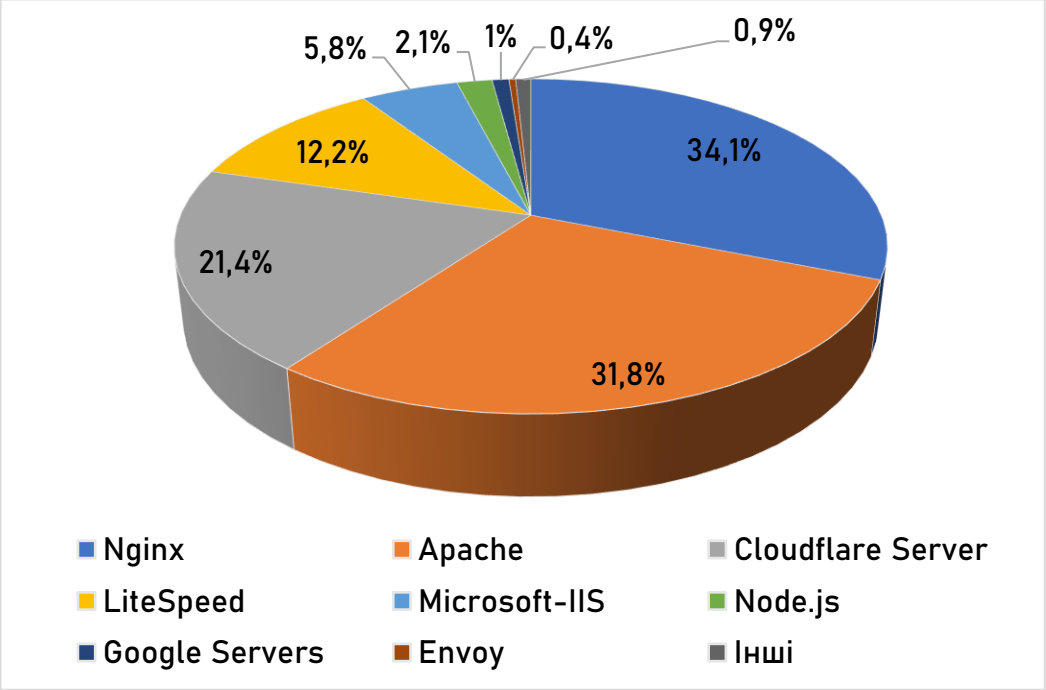


Рисунок 3.2.5 – Статистика використання веб-серверів за версією W3Techs

Також про поширеність даних веб-серверів свідчить статистика від Netcraft. На рисунку 3.2.6 зображено ринкову долю веб-серверів за роками [45].

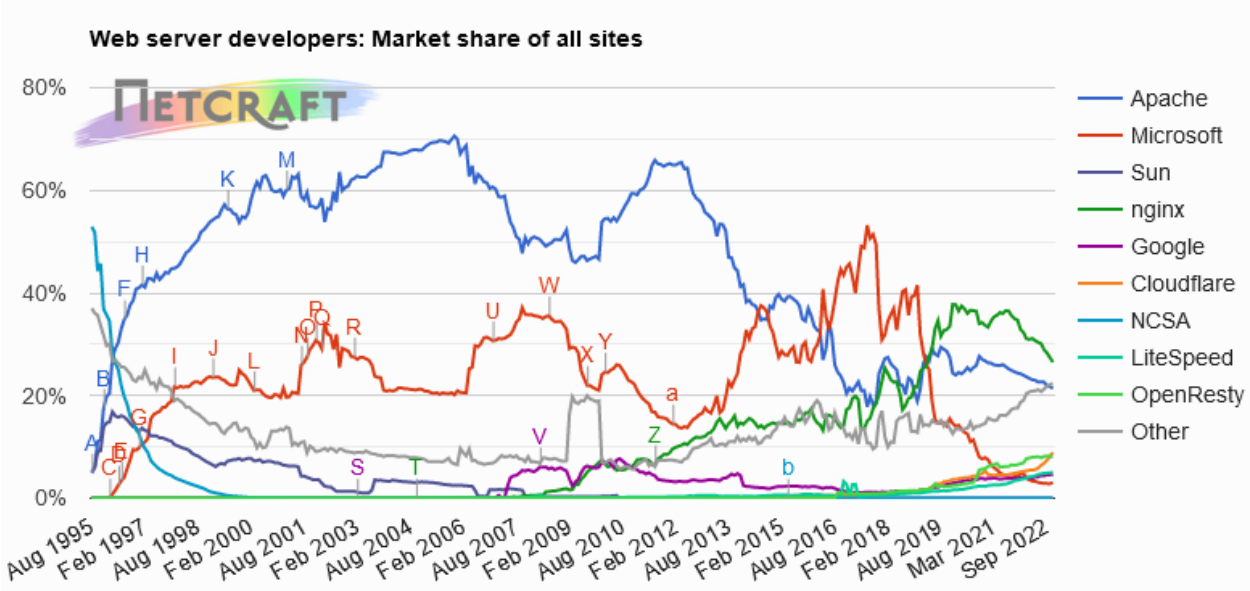


Рисунок 3.2.6 – Статистика ринкової долі веб-серверів за роками

Веб-сервер Nginx це безкоштовний веб та проксі-сервер, поштовий сервер. Даний додаток розроблявся з ціллю вирішення проблеми масштабування, що має назву «10 тисяч з'єднань» (C10k). Тобто існуючі веб-сервери були не здатні одночасно обробляти користувацькі запити більше ніж з 10 тисяч підключень. Даний веб-сервер має асинхронну подійно-орієнтовану архітектуру (Event driven architecture), що дозволяє швидко масштабуватися з мінімальними ресурсами. Особливістю архітектури, що використовує Nginx є

обробка множини підключень в єдиному потоці. Веб-сервер поділяє один запит на множину менших запитів, чим спрощує обробку кожного з них. Такі запити отримали назву робочих з'єднань. Оброблені запити збираються в одному віртуальному контейнері після чого формуються в початковий запит та відправляються користувачу. Одне з'єднання здатне оброблювати одночасно не більше ніж 1024 запити користувача. Також веб-сервер використовує виділений сегмент пам'яті, що має назву «пул» (pool). Даний пул є динамічним та може розширюватися при збільшенні довжини запиту, він дозволяє зменшити навантаження на оперативну пам'ять [46].

Ще однією особливістю даного веб-серверу є можливість використання його як зворотного проксі-серверу. Тобто сервер при отриманні запиту від клієнту не обробляє його самостійно, а частково або повністю перенаправляє на обробку до інших серверів. Варто зазначити, що сервер самостійно надсилає запити і повертає відповідь клієнту. Таким чином при використанні nginx у ролі зворотного проксі-сервера отримуємо наступні переваги:

– балансування навантаження – розподіл навантаження за рахунок перенаправлення запитів клієнтів до проксі-серверів;

– сервер зберігає в кеші вміст відповіді отриманої від проксі-серверів та використовує його для відповіді клієнтам, що прискорює надання відповіді на однакові запити;

– nginx може оброблювати та розшифровувати вхідні SSL-підключення та шифрувати відповіді, таким чином виступаючи у ролі кінцевої точки SSL для з'єднання з клієнтами;

– можливість налаштувати веб-сервер для стиснення відповідей перед їх відправленням, у випадку коли проксі-сервер не відправляє стисненні відповіді;

– захист від DDoS-атак, за рахунок обмеження кількості підключень з однієї IP-адреси та обмеження вхідних запитів, блокування доступу до серверу на основі місцезнаходження клієнту та значення заголовків запиту [47].

Веб-сервер nginx має низьке споживання ресурсів системи, високу швидкість роботи, тому чудово підходить для веб-проектів з великою кількістю відвідувань. Але, слабкою сторону даного серверу є нездатність працювати з динамічним контентом, тому його використовують разом з іншими веб-серверами (наприклад Apache). У таких випадках усі запити по HTTP надходять до Nginx, і залежно від запиту веб-сервер оброблює його або відправляє до Apache HTTP Server, що виступає у ролі проксі-серверу [48].

Веб-сервер IIS (Internet Information Services) є розробкою компанії Microsoft. Даний веб-сервер поставляється разом з операційною системою Windows і працює лише з даною ОС. Підтримкою IIS займається виключно компанія Microsoft, що є одночасно перевагою та недоліком. Велика кількість спеціалістів в області інформаційних технологій вважають IIS одним з небагатьох комерційних продуктів, що може бути конкурентом «open-source» рішень. Завдяки виправленню проблем з безпекою, покращенню продуктивності та зручності інструментів адміністрування IIS збільшив

ринкову долю з 21% у 2010 році до 32% у лютому 2014, що можна спостерігати на рисунку 3.2.6. Найбільші зміни торкнулися безпеки. Так, версія IIS 6.0 мала вразливість до атак з підміною вмісту веб-сайту на банер (відомий вірус Code Red). Для розширення базового функціоналу IIS використовує модулі. Так, при роботі з файлами через FTP використання маршрутизації Application Request Routing дозволяє збалансувати навантаження та підвищити відмовостійкість. Оскільки IIS є розробкою компанії Microsoft, то він передбачає високу сумісність з ASPX (Active Server Pages) та програмною платформою .NET Framework [49].

Веб-серве Apache належить до програм з відкритим вихідним кодом та розповсюджується як вільне програмне забезпечення. Даний веб-сервер є крос-платформовим. Apache чудово себе зарекомендував при роботі за масштабними проектами, є дуже гнучким у налаштуванні, що надає можливість розкрити усі особливості розміщуваного веб-ресурсу. Веб-сервер Apache складається з ядра, що написано мовою С та модулів. За допомогою модулів можна розширити функціональність ядра веб-серверу, вибір яких здійснюється на основі особливостей вмісту, що зберігається на сервері. Модулі розширюють функціональність ядра Apache, розробкою яких займаються як і сторонні програмісти, так і розробниками Apache. Загальна кількість модулів для Apache складає більше 500. Цілями розробки даних модулів є:

– Розширити наявний функціонал;

– Виправлення багів, зміни основних функцій;

– Покращення безпеки, усунення вразливостей;

– Додати підтримку певної мови програмування.

Налаштування параметрів ядра здійснюється за допомогою редагування конфігураційних файлів:

– httpd.conf – файл з параметри рівня сервера;

– extra/httpd-vhosts.conf – файл з параметрами віртуального хоста;

– .htaccess – файл з параметрами рівня каталогу.

Таким чином конфігурація веб-сервера є децентралізованою. Також завдяки інтерпретації конфігурації є можливість внесення зміни до налаштувань серверу та бачити зміни без необхідності перезавантаження веб-серверу. Крім того, .htaccess-файли дозволяють обмежити доступ до певних функцій веб-застосунку залежно від того чи мають вони права адміністратора [50].

Для розробки веб-додатку необхідно в першу чергу встановити локальний веб-сервер. Зручним рішенням є використання пакету програмного рішення. Такими пакетами є: AMPPS, WAMP, LAMP, MAMP, XAMPP.

З перелічених пакетів було обрано AMPPS. Даний пакет є потужною платформою для створення веб-ресурсів з повноцінною бібліотекою додатків. Даний пакет підтримує розповсюджені системи керування вмістом (CMS – Content Management System), постачається разом з Apache, СУБД MySQL,

PHP. Також має підтримку MongoDB, Perl, Python. Даний пакет може бути встановлений на Windows, Linux та Mac OS [51].

MySQL є системою управління реляційними базами даних, що належить до ПЗ з відкритим вихідним кодом та підтримується компанією Oracle. Дане ПЗ поширюється безкоштовно та є крос-платформовим. Дана СУБД підтримує різноманітні типи даних, що модна поділити на наступні групи:

1) Текстові;
2) Числові;
3) Типи для роботи з датою та часом;
4) Спеціальні типи даних.

До текстових типів даних належать:

– CHAR – є рядком фіксованої довжини, що зберігає певну кількість символів, здатний зберігати до 255 байт;

– VARCHAR – представляє собою рядок змінної довжини, відрізняється від CHAR тим, що займає стільки місця скільки потрібно для зберігання. Тобто, якщо задано тип поля VARCHAR(10), а зберігається рядок довжиною лише 6 символів, то даний рядок буде займати 6 символів та додатковий байт для зберігання довжини рядка. Максимально VARCHAR зберігає 65 535 байт;

– BINARY та VARBINARY – це аналог CHAR та VARCHAR, але на відміну від них зберігають дані у бінарному форматі;

– TEXT та BLOB – подібні до BINARY та CHAR, головною відмінністю є підтримка більших розмірів. Дані типи даних є динамічними та займають стільки місця скільки необхідно для зберігання даних, тобто не має потреби вказувати заздалегідь довжину даних. BLOB на відміну від TEXT зберігає двійкові рядки, даний тип зазвичай використовують для зберігання файлів, зображень у вигляді двійкових рядків;

– ENUM – є рядковим типом даних, для якого заздалегідь вказуються дозволені значення для зберігання;

– SET – це об’єкт рядків, схожий на ENUM, але може зберігати декілька значень з дозволеного діапазону [52].

До числових типів даних належать:

– INT та INT UNSIGNED – типи даних, що підтримують зберіганні цілих чисел. Тип INT зберігає числа в діапазоні від -2 147 483 648 до 2 147 483 647 та займає 4 байти. Тип INT UNSIGNED зберігає лише позитивні цілі числа, а саме від 0 до 4 294 967 295 і також займає 4 байти;

– TINYINT – використовується для зберігання чисел від -128 до 127 та займає 1 байт, також має тип UNSIGNED який здатен зберігати числа від 0 до 255 і також займає 1 байт;

– MEDIUMINT – зберігає числа від -8 388 608 до 8 388 607, займає 3 байти, також може бути UNSIGNED і зберігати числа в діапазоні від 0 до 16 777 215;

– SMALLINT – зберігає числа від -32768 до 32767, займає 2 байти, має тип UNSIGNED, що зберігає числа від 0 до 65 535;

– BIGINT – зберігає числа від -9 223 372 036 854 775 808 до 9 223 372 036 854 775 807, займає 8 байт, має тип UNSIGNED, що зберігає числа від 0 до 18 446 744 073 709 551 615;

– FLOAT та DOUBLE – зберігають числа з плаваючою точкою, тип FLOAT займає 4 байти та зберігає числа одинарної точності від $-3.4028 * 10^{38}$ до $3.4028 * 10^{38}$, тип DOUBLE займає 8 байт та зберігає числа з плаваючою точкою подвійної точності від $-1.7976 * 10^{308}$ до $1.7976 * 10^{308}$ та займає 8 байт;

– DECIMAL – зберігає числа з фіксованою точністю, здатен зберігати числа довжиною до 65 символів.

До типів для роботи з датою та часом відносяться:

– DATE – використовується для зберігання дати у межах від 1 січня 1000 року до 31 грудня 9999 року, займає 3 байти;.

– TIME – використовується для зберігання часу в межах від -838:59:59 до 838:59:59, займає 3 байти;.

– DATETIME – є об'єднанням дати та часу, таким чином зберігає мітки часу в межах від "1000-01-01 00:00:00" до "9999-12-31 23:59:59", займає 8 байт;

– TIMESTAMP – є аналогом DATETIME, але зберігає дату та час в діапазоні від "1970-01-01 00:00:01" UTC до "2038-01-19 03:14:07" UTC, займає 4 байти;

– YEAR – даний тип зберігає рік у вигляді 4 цифр у діапазоні від 1901 до 2155 року, займає 1 байт [53].

До спеціальних типів даних належать:

– BOOL – зберігає логічне значення – true або false, займає 1 байт;

– JSON – зберігає об'єкти у нотації JSON, що полегшує зберігання документів даного типу та прискорює їх пошук [52, 53].

Для встановлення AMPPS необхідно завантажити інсталяційний файл з офіційного сайту (https://ampps.com/downloads/). На рисунку 3.2.7 зображено сторінку завантаження AMPPS.

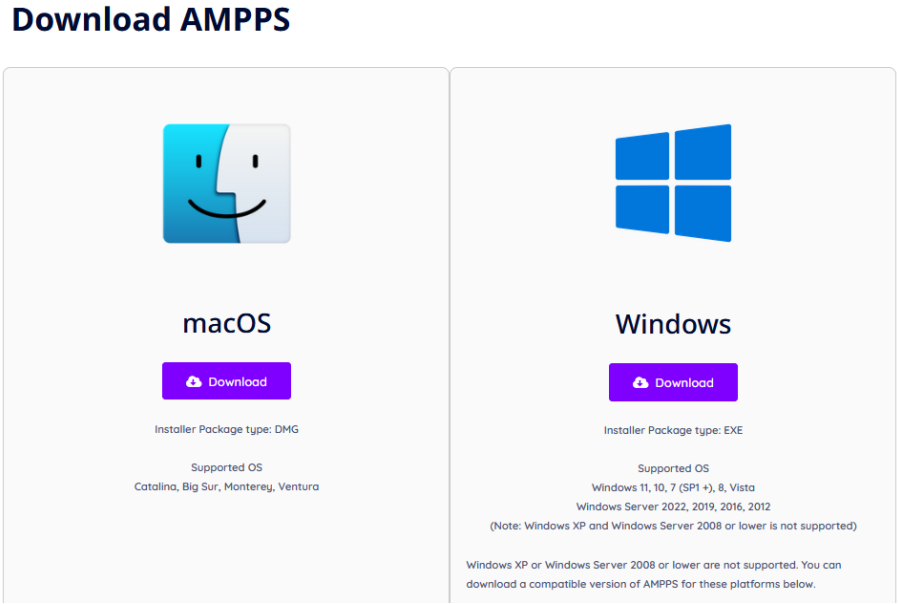


Рисунок 3.2.7 – Сторінка завантаження AMPPS

Після запуску інсталяційного файлу буде відкрито вікно з привітанням, де необхідно натиснути кнопку Next. На другому кроці необхідно обрати шлях для встановлення додатку. На рисунку 3.2.8 зображено вікно вибору шляху.

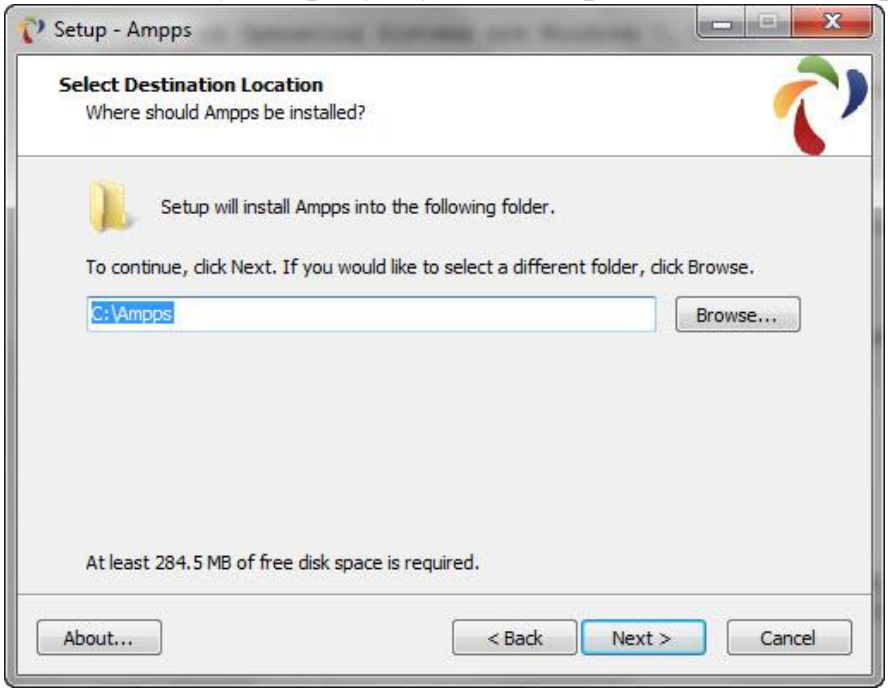


Рисунок 3.2.8 – Вибір шляху для встановлення

Після завершення інсталяції необхідно налаштувати локальний сервер. Для цього необхідно запустити AMPPS та навпроти пункту Apache натиснути на іконку налаштувань і обрати пункт host file. На рисунку 3.2.9 зображено пункт host file.

Рисунок 3.2.9 – Вибір пункту host file

Після вибору пункту host file відкриється файл hosts. В даному файлі додаються імена та адреси локальних веб-ресурсів, для додавання нової адреси необхідно вказати IP-адресу локального серверу та його назву. За допомогою знаку “#” можна додати коментар. На рисунку 3.2.10 наведено налаштування адреси локального серверу.

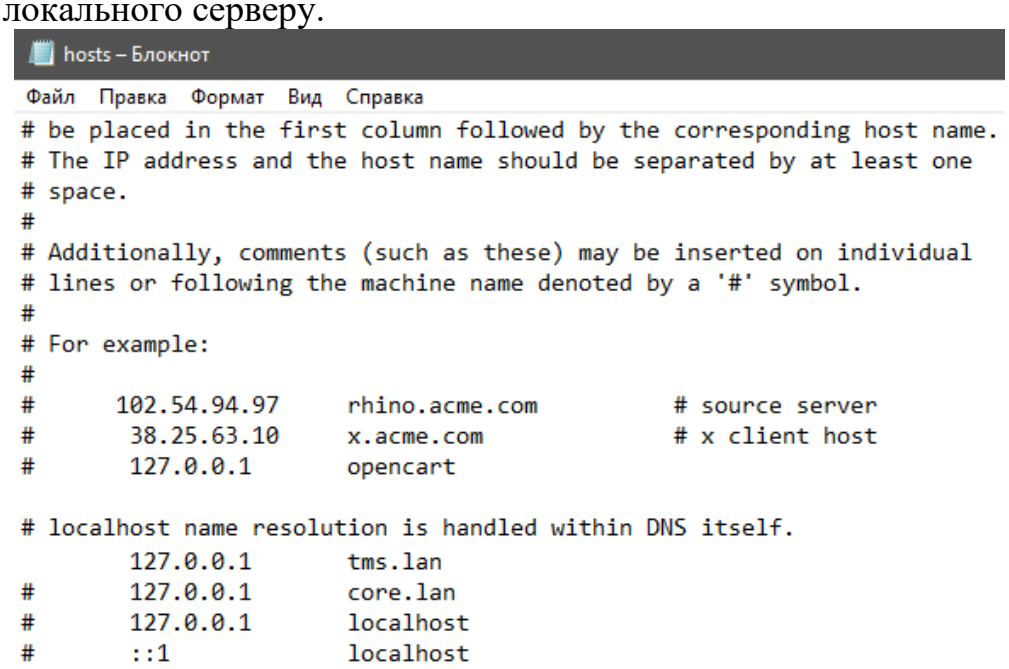


Рисунок 3.2.10 – Налаштування локального серверу

Також необхідно налаштувати віртуальний хостинг. Для цього у меню AMPPS необхідно навпроти пункту Apache натиснути іконку налаштувань та обрати пункт virtual host. Далі відкриється файл httpd-vhosts.conf. В даному файлі необхідно вказати шлях до сайту. Налаштування віртуального хостингу:

```
#### tms VirtualHost ####


<VirtualHost 127.0.0.1:80>
<Directory "D:/Ampps/www/tms">
Options FollowSymLinks Indexes
AllowOverride All
Order deny,allow
allow from All
</Directory>
ServerName tms
ServerAlias tms
ScriptAlias /cgi-bin/ "D:/Ampps/www/tms/cgi-bin/"
DocumentRoot "D:/Ampps/www/tms"
ErrorLog "D:/Ampps/apache/logs/tms.err"
CustomLog "D:/Ampps/apache/logs/tms.log" combined
</VirtualHost>

####################################
```

Отже, таким чином було налаштовано локальний веб-сервер. Далі буде розглянуто встановлення та налаштування середовища розробки.

3.3 Встановлення та налаштування середовища розробки PhpStorm

Оскільки було прийнято рішення розробляти веб-додаток із застосуванням мови PHP, то у якості середовища розробки буде використовуватися PhpStorm.

PhpStorm є крос-платформовою інтегрованою середою розробки (Integrated development environment – IDE), що розроблена компанією JetBrains та випущений в 2009 році. Дане середовище розробки є надзвичайно потужним інструментом розробки, має інтеграцію найважливіших інструментів та зручний інтерфейс. PhpStorm спеціально розроблений для програмування мовою PHP, тому має підтримку усіх основних бекенд-фреймворків та CMS, а саме: Symfony, Drupal, WordPress, Zend Framework, Laravel, Magento, Joomla!, CakePHP, Yi та інші. PhpStorm ретельно аналізує структуру коду та надає підказки по виправленню помилок, підтримує усі можливості мови PHP з врахуванням усіх версій інтерпретатора, що надає можливість працювати із застарілим кодом. Даний редактор підтримує автодоповнення коду та рефакторинг. Також дана IDE надає дуже зручний пошук, можна обирати де саме здійснювати пошук, у певній директорії або ж по усьому проекту. На рисунку 3.3.1 зображено пошук де у проекті використовується функція is_admin [54].

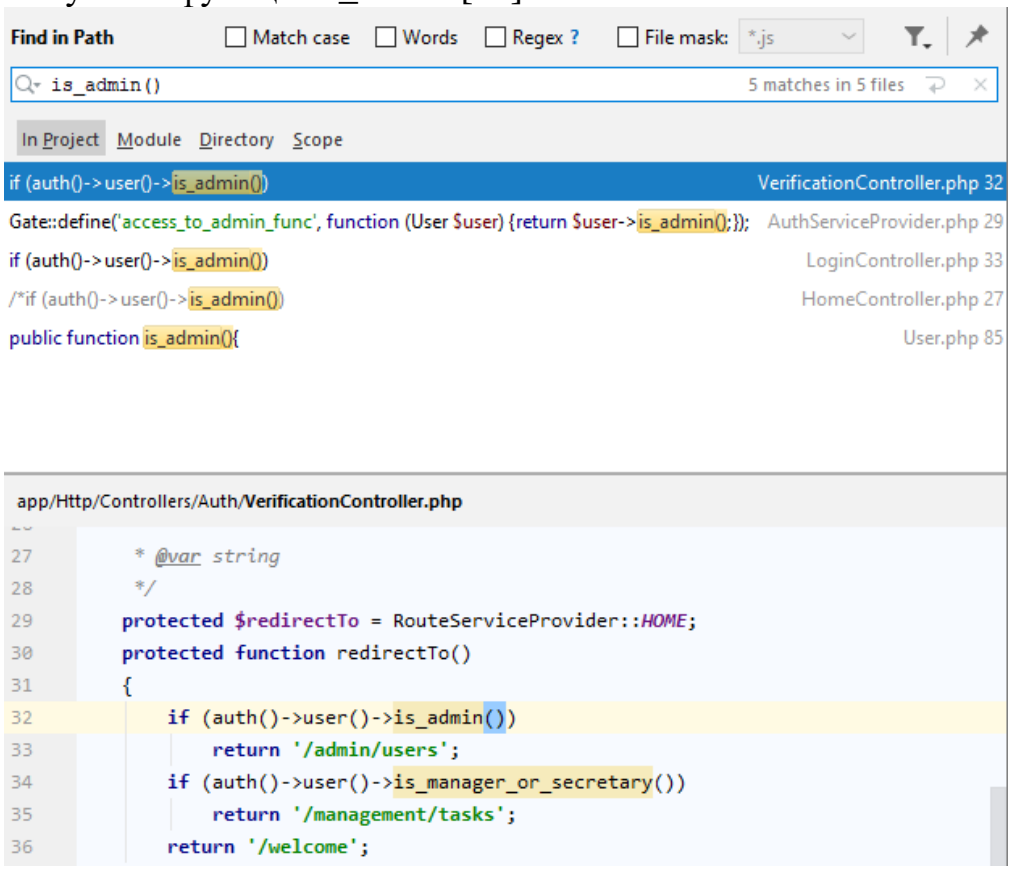


Рисунок 3.3.1 – Пошук місць використання функції is_admin

Як видно з рисунку 3.3.1 IDE надає список з рядками коду де здійснюється виклик функції is_admin, відображає назву файлу та шлях до нього, надає можливість одразу переглянути вміст файлів та редагувати їх.

Також PhpStorm підтримує мову розмітки HTML, розмітку файлів CSS, SCSS, LESS та JavaScript. Окрім цього має підтримку таких сучасних веб-технологій розробки, як: CofeeScript, ECMAScript, Harmony, Dart та багато іншого. Дане середовище має функцію Live Edit, що зберігає усі зміни коду автоматично, таким чином гарантується збереження усіх внесених змін та зменшуються ризики втрати виконаної роботи. А також дана функція дозволяє бачити внесені зміни у реальному часі, що економить багато часу [55].

PhpStorm підтримує інтеграцію з популярними системами контролю версій (СКВ, source code management, SCM), а саме: Subversion, Git, Perforce, CVS, Mercurial, TFS. Усі одноманітні задачі, наприклад додавання або видалення файлів виконуються автоматично (при створенні нового файлу відкриється вікно з підтвердженням додавання файлу до індексації чи ні, можна обрати відповідний чекбокс аби запам’ятати вибір). Конфлікти злиття файлів легко вирішуються за допомогою visual merge, що є вбудованим інструментом. Локальні зміни позначаються підсвіткою в редакторі, що забезпечую просту та інтуїтивно зрозумілу навігацію. На рисунку 3.3.2 зображено приклад відображення файлів у дереві проекту залежно від їхнього статусу індексації [56].

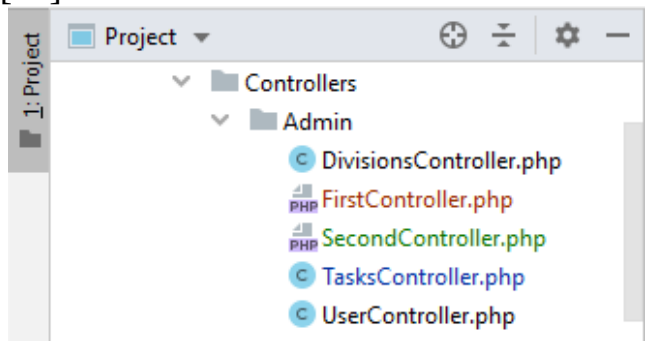


Рисунок 3.3.2 – Відображення файлів у залежності від їхнього статусу індексації у системі контролю версій

На рисунку 3.3.2 зображено вміст каталогу Admin, де файли DivisionController.php, UserController.php додано до індексації і усі зміни були збережені у СКВ. Зміни у файлі FirstController.php не відслідковуються, оскільки файл не додано до СКВ, тому його назва має помаранчевий колір. Файл SecondController.php є щойно доданим до СКВ для відслідковування та ще не був «закомічений» (мається на увазі, що даний файл ще не увійшов до історії змін). Файл TasksController.php відслідковується на наявність змін, синій колір його назви означає, що з останнього «коміту» файл було відредаговано. Окрім цього можна переглянути історію збережених змін, для цього необхідно обрати кореневу папку проекту та перейти за шляхом «VCS-Git-Show history…». На рисунку 3.3.3 зображено історію проекту:

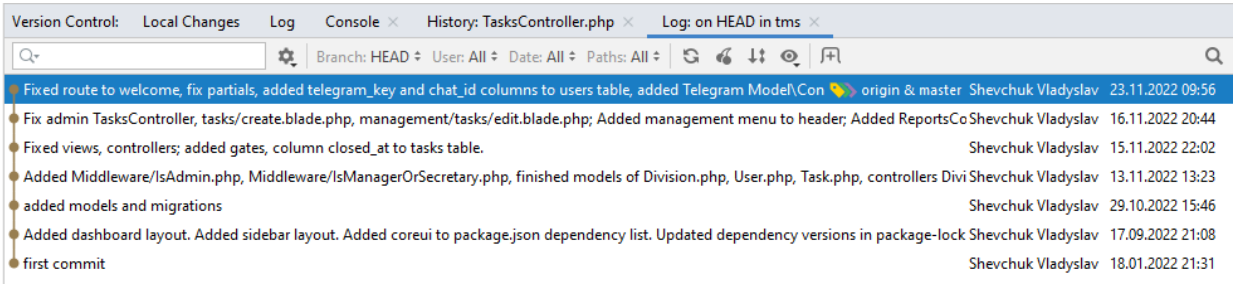


Рисунок 3.3.3 – Перегляд збережених змін за проектом у СКВ

Для перегляду збережених змін певного файлу достатньо обрати потрібний та виконати ті ж самі дії, що описано для перегляду змін у проекті.

Також у PhpStorm можна налаштувати підключення за допомогою FTP, FTPS, SFTP. Для цього необхідно перейти за шляхом «Tools-Deployment-Configurate…». На рисунку 3.3.4 зображено меню Deployment в якому є можливість додати нові з'єднання [57].

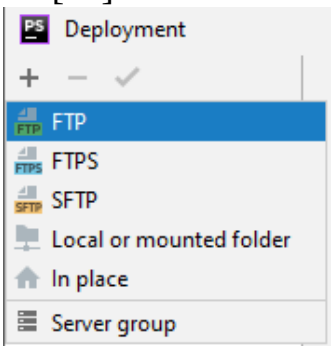


Рисунок 3.3.4 – Меню Deployment та додавання FTP з'єднання

Далі необхідно ввести параметри налаштування FTP підключення. На рисунку 3.3.5 наведено вікно вводу параметрів FTP підключення.

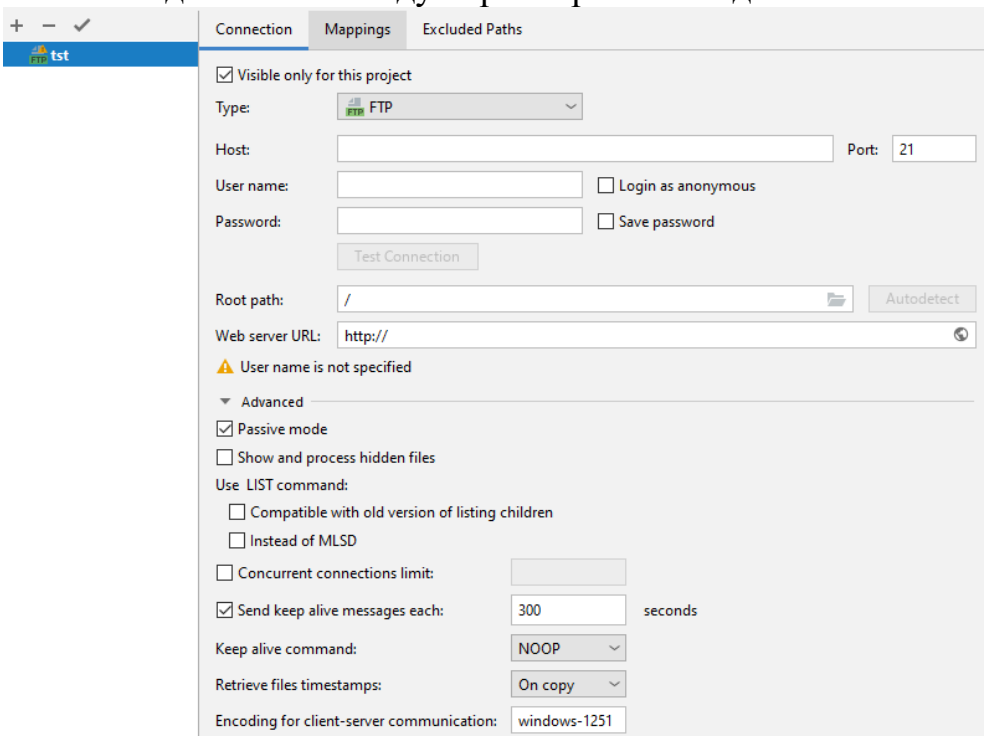


Рисунок 3.3.5 – Вікно вводу параметрів FTP з'єднання

Завдяки інтеграції FTP з'єднань можна налаштувати автоматичне завантаження та синхронізацію файлів, прискорити розгортання файлів на віддаленому сервері.

Ще одна з найважливіших інтеграцій це можливість взаємодії з базою даних. Для підключення бази даних необхідно перейти за шляхом «View-Tool windows-Database». Після чого відкриється вікно додавання з'єднання. У відчиненому вікні необхідно натиснути «+» перейти до пункту Data Source та обрати потрібну СУБД зі списку. Список СУБД наведено на рисунку 3.3.6.

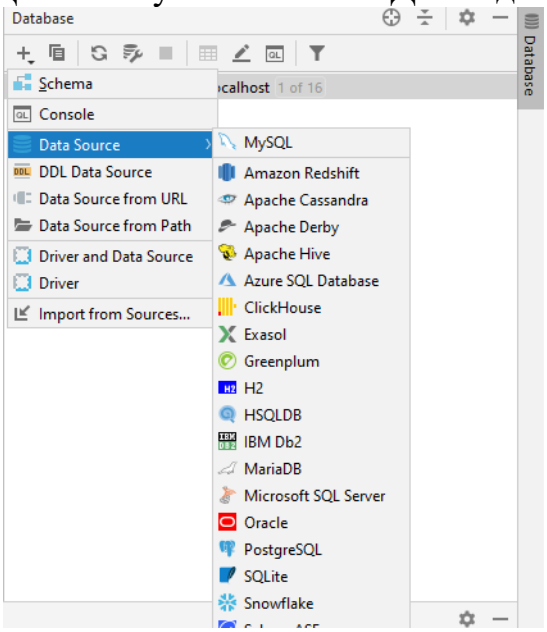


Рисунок 3.3.6 – Список доступних СУБД для підключення

Розглянемо приклад додавання підключення до MySQL. Після вибору відповідного пункту відкривається вікно налаштування підключення. У даному вікні потрібно ім'я підключення, адресу хосту, порт, ім'я користувача БД, назву БД та URL. Після введення даних необхідно перевірити підключення натиснувши кнопку «Test Connection». На рисунку 3.3.7 зображено вікно введення даних для підключення до MySQL.

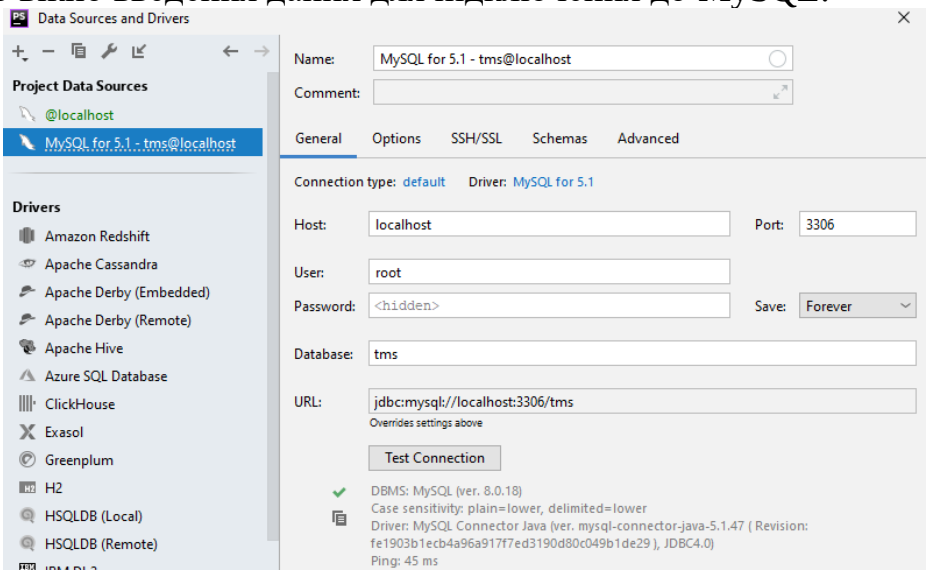


Рисунок 3.3.7 – Вікно введення даних для підключення до MySQL

Після створення підключення у вікні Database буде відображено дерево бази даних, де можна виконувати маніпуляції з її вмістом. На рисунку 3.3.8 зображено дерево бази даних.

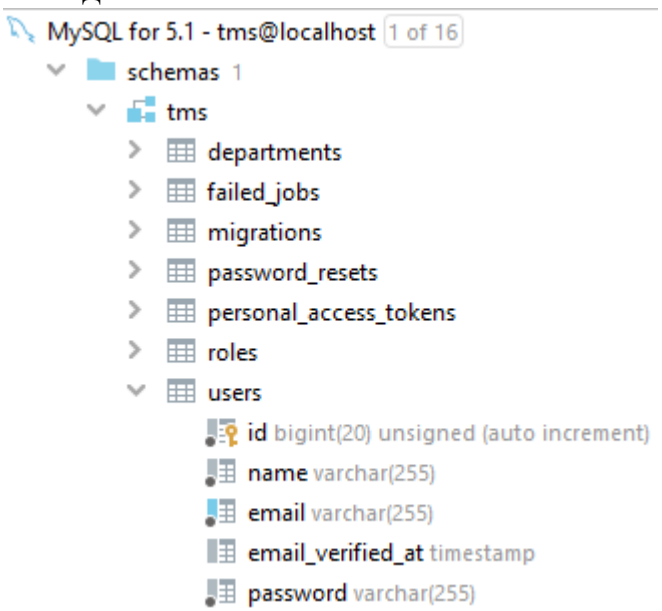


Рисунок 3.3.8 – Відображення дерева бази даних у IDE PhpStorm

Дана IDE є платною, але у JetBrains існують певні програми за якими надається право безкоштовно використовувати PhpStorm. Одна з цих програм передбачає безкоштовне користування програмою для студентів та викладачів з акредитованих навчальних закладів (вузів, коледжів, шкіл). Для цього необхідно отримати власну адресу електронної пошти від навчального закладу. Далі можна перейти на офіційний сайт JetBrains за посиланням: https://www.jetbrains.com/shop/eform/students. На даній сторінці необхідно заповнити одну із запропонованих форм, що зображено на рисунку 3.3.9.

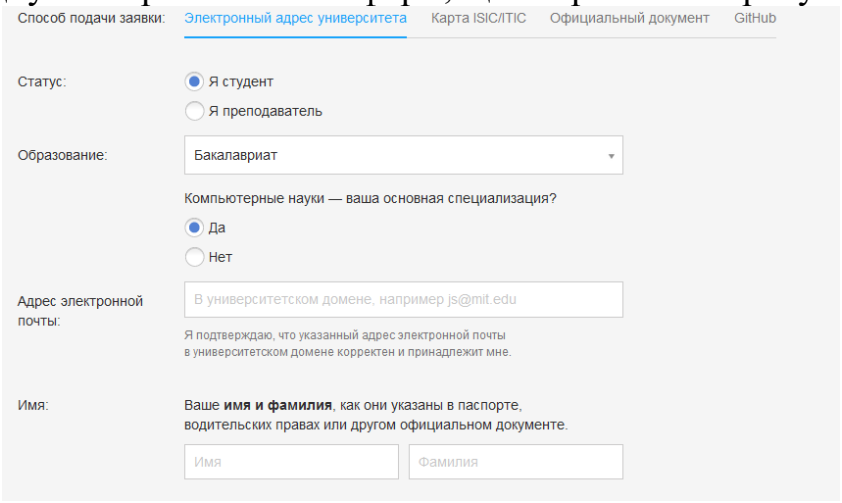


Рисунок 3.3.9 – Вибір способу подачі заявки на безкоштовний доступ до PhpStorm та інших продуктів JetBrains

Одним з доступних варіантів є використання акаунту Github. У свою чергу Github пропонує доступ до багатьох інтернет-ресурсів за програмою для студентів включаючи доступ до продуктів JetBrains. Оскільки в мене вже

наявний обліковий запис Github, що зареєстровано на електрону пошту вузу, то я обрав саме цей варіант подання заявки. На рисунку 3.3.10 зображено форма подачі заявки через авторизацію у Github.

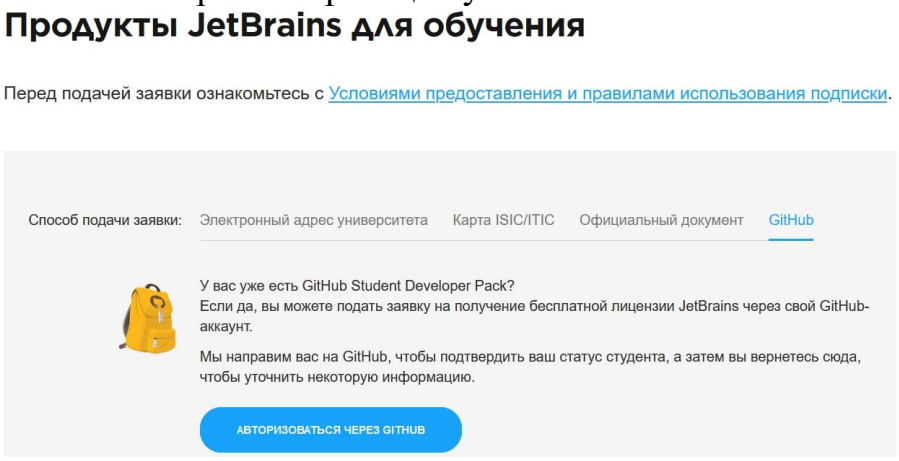


Рисунок 3.3.10 – Форма подання заявки за допомогою авторизації через Github

Далі буде запропоновано увійти до аккаунту Github. На рисунку 3.3.11 зображено вікно авторизації у JetBrains за допомогою Github.

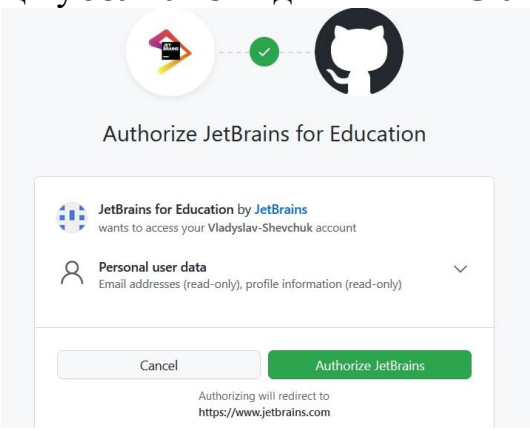


Рисунок 3.3.11 – Вікно авторизації у JetBrains за допомогою Github

Далі буде надіслано повідомлення на вашу електронну пошту з підтвердженням. На рисунку 3.3.12 зображено лист з підтвердженням.

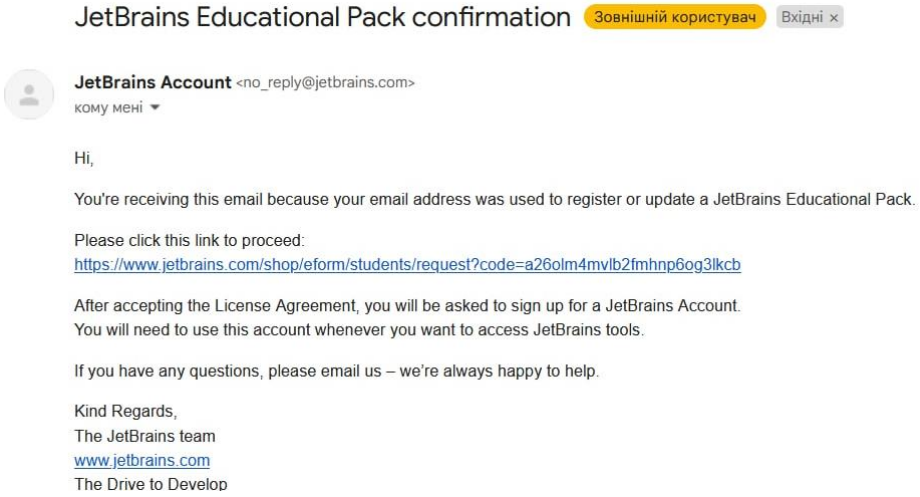


Рисунок 3.3.12 – Лист підтвердження від JetBrains

В даному листі міститься посилання за яким необхідно перейти для активації акаунту. Після переходу за посиланням Ви отримаєте повідомлення про успішний вхід і Вам буде запропоновано створити пароль для облікового запису. Також на даній сторінці можна буде обрати мову якою надходитимуть листи від JetBrains. Після створення паролю необхідно перейти до вкладки «Licenses». На даній сторінці можна переглянути усі придбані ліцензії. На рисунку 3.3.13 зображено продукти, що входять до студентської ліцензії.

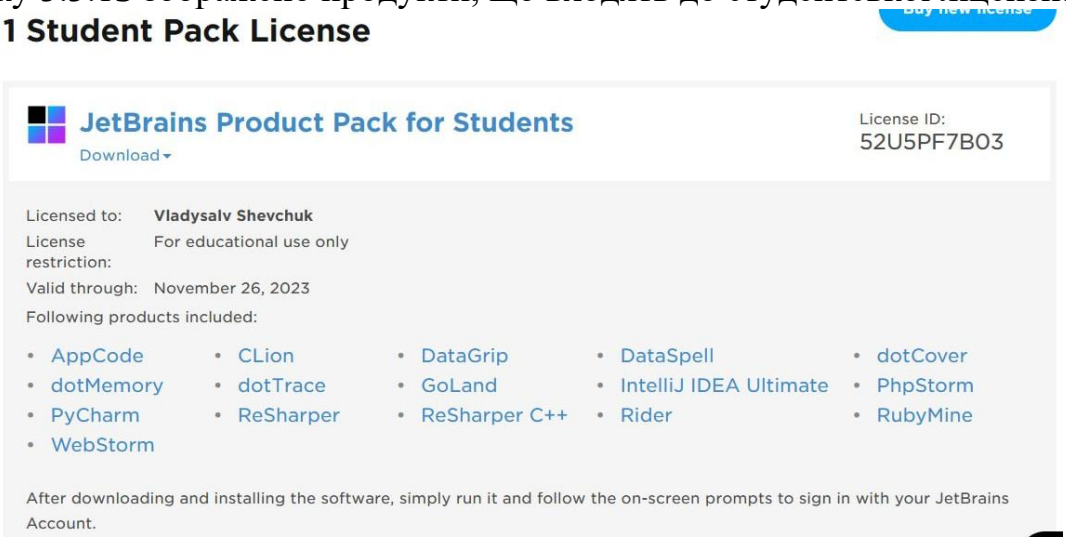


Рисунок 3.3.13 – Продукти що входять до студентської ліцензії

Для завантаження PhpStorm необхідно натиснути на відповідний пункт у списку, дане посилання перенаправляє користувача на головну сторінку JetBrains. На даній сторінці необхідно перейти до завантажень. На рисунку 3.3.14 зображено сторінку завантаження PhpStorm.

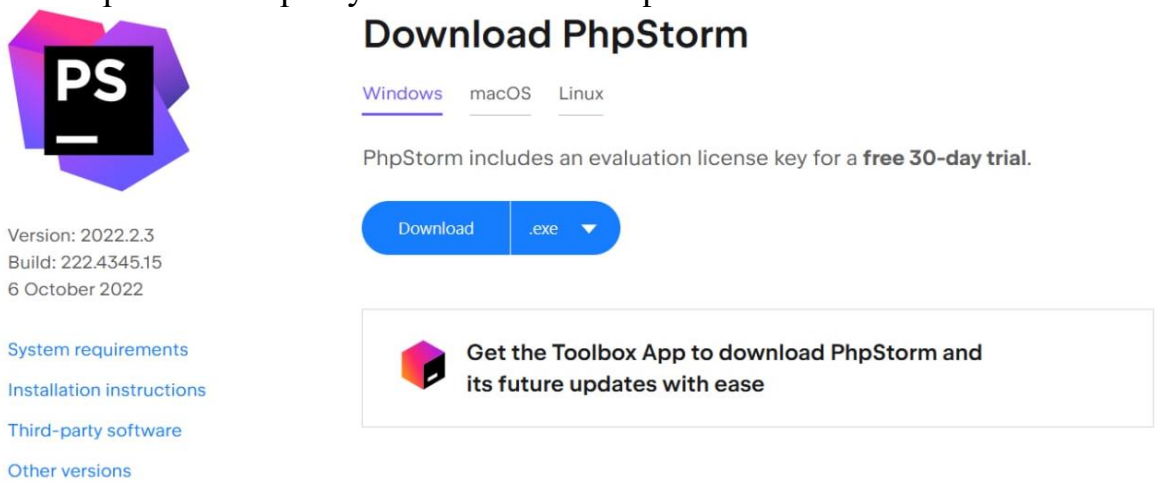


Рисунок 3.3.14 – Сторінка завантаження PhpStorm

Для завантаження доступно два варіанти: exe-файл та zip-файл. Мною було обрано exe-файл. Після завантаження необхідно запустити файл інсталяції. На першому кроці Вас зустрічає вікно привітання, де вказується, що рекомендовано закрити усі інші додатки перед початком інсталяції. На наступному кроці необхідно обрати місце встановлення IDE. На рисунку 3.3.15 зображено вікно вибору місця встановлення додатку.

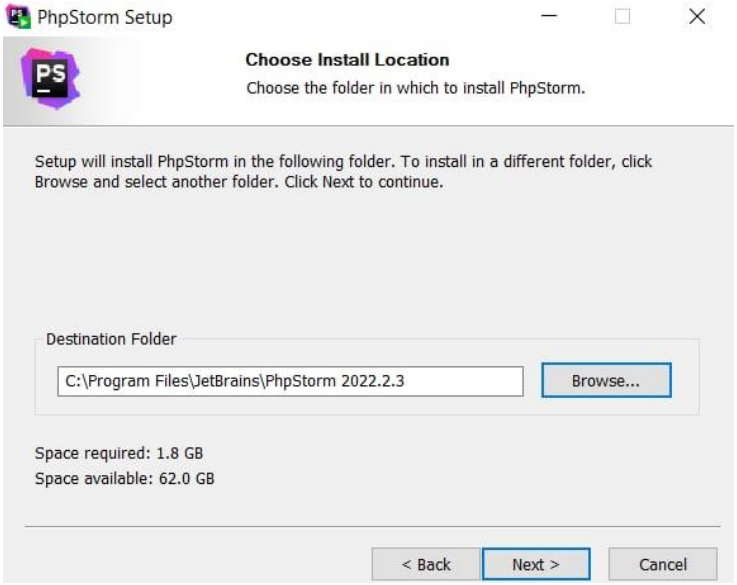


Рисунок 3.3.15 – Вікно вибору місця встановлення додатку

Наступним кроком необхідно обрати потрібні опції для налаштування. На даному кроці пропонується створити ярлик додатку на робочому столі, оновити змінну PATH, додати до контекстного меню можливість відкривати папки як проекти за допомогою PhpStorm, додати асоціації до файлів залежно від їх типу. Після інсталяції застосунку буде запропоновано перезавантажити комп'ютер одразу або ж виконати перезавантаження вручну пізніше. На рисунку 3.3.16 зображено вікно вибору опцій інсталятора PhpStorm.

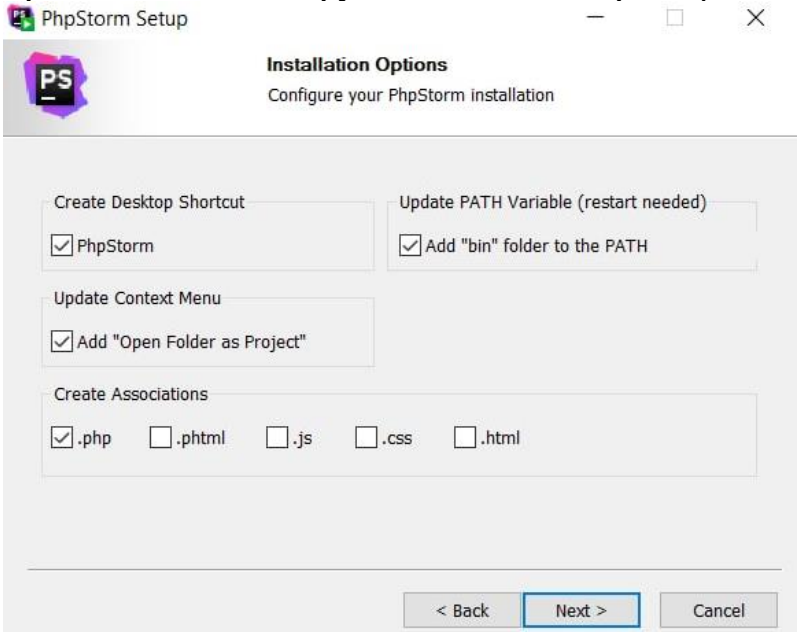


Рисунок 3.3.16 – Вікно опцій інсталятора PhpStorm

При запуску додатку необхідно ознайомитися та прийняти політики конфіденційності. Далі Вам буде запропоновано способи активації додатку, серед яких є: активація за допомогою входу до акаунту JetBrains, активація за допомогою ліцензійного коду, активація за допомогою серверу ліцензій або ж активація пробного періоду на 30 днів. Отже, обираємо активацію за

допомогою JetBrains акаунту. Далі буде відкрито браузер зі сторінкою авторизації у JetBrains. Після авторизації можна повернутися до вікна додатку та побачити, що ліцензію було застосовано. На рисунку 3.3.17 зображено вікно додатку з повідомленням про активацію ліцензії.

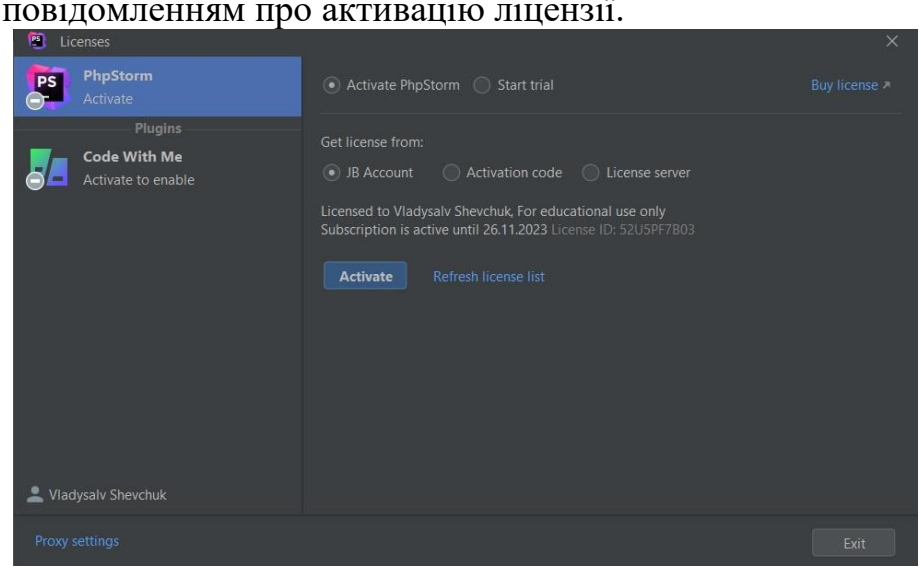


Рисунок 3.3.17 – Вікно активації PhpStorm

Після активації додатку можна переходити до розробки. Для зміни зовнішнього вигляду вікон додатку можна перейти до меню налаштування. Для цього необхідно обрати в меню пункт «Customize». На рисунку 3.3.18 зображено вікно налаштування програми PhpStorm.

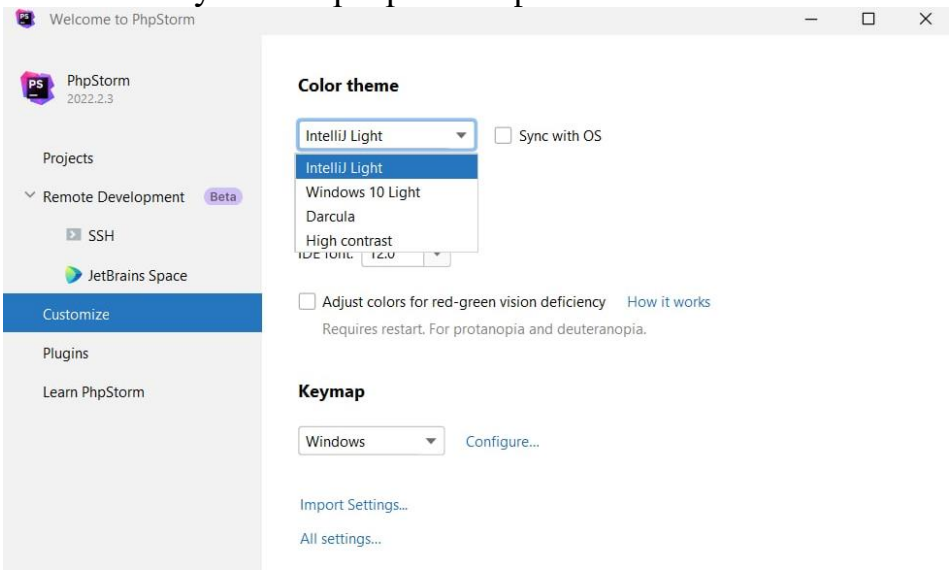


Рисунок 3.3.18 – Вікно налаштування програми PhpStorm

У відчиненому вікні можна змінити кольорову тему. Також є можливість синхронізувати тему з операційною системою. Таким чином при зміні тем у операційній системи додаток автоматично змінюватиме власну тему (з темної на світлу або навпаки). Також у даному меню можна налаштувати комбінації швидких команд.

Далі буде розглянуто роботу з системою контролю версій Git.

3.4 Робота з Git

Git це система контролю версій, що була створена у 2005 році Лінусом Торвальдсом. Git є крос-платформовим додатком, належить до програм з відкритим вихідним кодом та розповсюджується безкоштовно. Дана система надає можливість відстежувати історію розробки ПЗ та спрощує роботу команд розробників над складними проектами з будь-якої точки світу.

Система контролю версій веде історію змін. Так, наприклад, якщо графічний дизайнер має бажання зберегти кожну з версій макету або зображення, то він може використати СКВ. Дана система дозволяє переглянути різницю між файлами різних версій, переглянути інформацію про те хто вносив зміни і коли, та багато іншого. Також у разі втрати будь-якого файлу СКВ дозволяє відновити його, або ж при необхідності повернутися до певної з версій файлу. Досить поширеним випадком є копіювання файлу до окремої директорії для зберігання оригіналу. Але при великій кількості копій існує висока ймовірність виникнення помилок. При такому підході можна забути до якої директорії було скопійовано потрібний файл, або ж помилково замінити не той файл. Для вирішення даної проблеми програмістами було розроблено локальні СКВ. Такі системи зберігають усі зміни в файлах під контролем версій. На рисунку 3.4.1 зображено локальні системи контролю версій [58, 59].

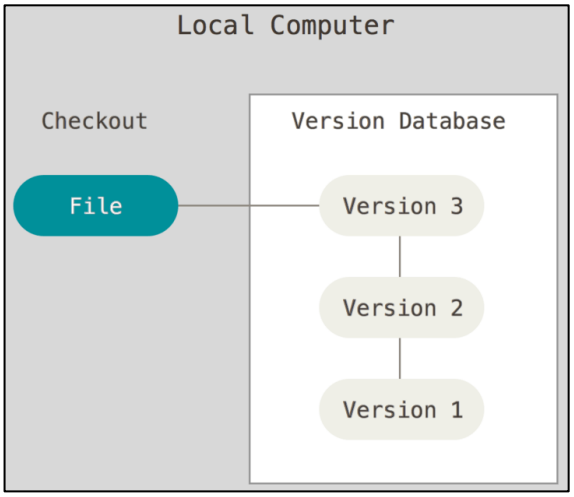


Рисунок 3.4.1 – Локальні системи контролю версій

Однією з найпоширеніших локальних СКВ була RCS, що й досі застосовується певними користувачами. Дана системи зберігає відмінності між файлами в спеціальному форматі на диску. Відмінності зберігаються у виді латок, відновлення файлів відбувається за рахунок з'єднання цих латок.

Але локальні СКВ не вирішують питання співпраці з іншими людьми. Для таких цілей було розроблено централізовані системи контролю версій (ЦСКВ), такі як: Subversion, CVS, Perforce. Дані системи мають єдиний сервер, що зберігає усі версії файлів та число клієнтів, що можуть отримувати файли зі сховища. На рисунку 3.4.2 зображено централізовані системи контролю версій [59, 60].

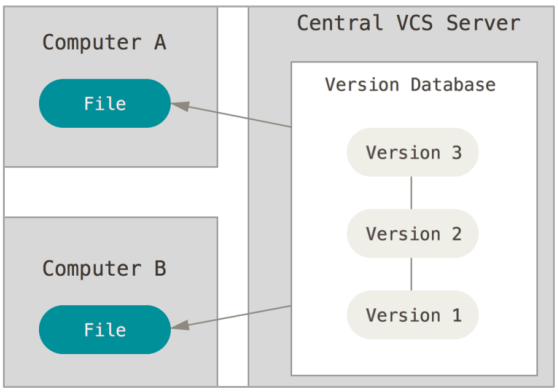


Рисунок 3.4.2 – Централізовані системи контролю версій

Перевагою даного підходу є можливість адміністратора контролювати дозволи для користувачів, також спрощується адміністрування порівняно з локальними. Головним недоліком даних систем є єдина точки відмови – централізований сервер. У випадку відмови централізованого сервер усі користувачі втрачають змогу співпрацювати, підвищений ризик втрати даних у випадку виходу з ладу накопичувача сервера. Але варто зазначити, що локальні СКВ також мають дані вразливості [59, 60].

Найкращим рішенням для зберігання файлів є використання децентралізованих систем контролю версій (ДСКВ) до яких належать: Mercurial, Git, Bazaar. У цих системах користувач зберігає повну копію сховища, що включає повну історію, усі знімки файлів. Таким чином у кожного з користувачів зберігається резервна копія усіх даних. Отже, у ДСКВ у випадку виходу з ладу серверу, що використовується як репозиторій не призведе до серйозних проблем. Якщо з будь-якої причин репозиторій з даними було втрачено, то його можна відновити за допомогою локальної копії. На рисунку 3.4.3 зображено децентралізовані системи контролю версій [59, 61].

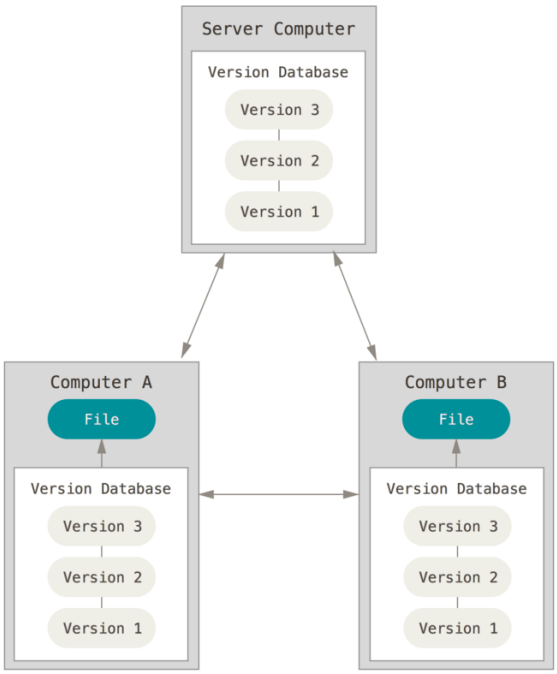


Рисунок 3.4.3 – Децентралізовані системи контролю версій

Такі системи як Subversion, Bazaar, Perforce зберігають інформацію у вигляді списку файлів та внесених змін протягом деякого часу. Такий контроль базується на дельтах контролю версій.

На відміну від вище згаданих систем подібних до них, Git розглядає дані як серію знімків файлової системи. При створенні коміту Git запам’ятовує усі файли та зберігає посилання на їх знімок. У випадку, якщо у файлі не відбулося ніяких змін, то Git робить посилання на попередній знімок, що відповідає даному стану файлу. На рисунку 3.4.4 зображено зберігання даних як знімків проекту за хронологією [62].

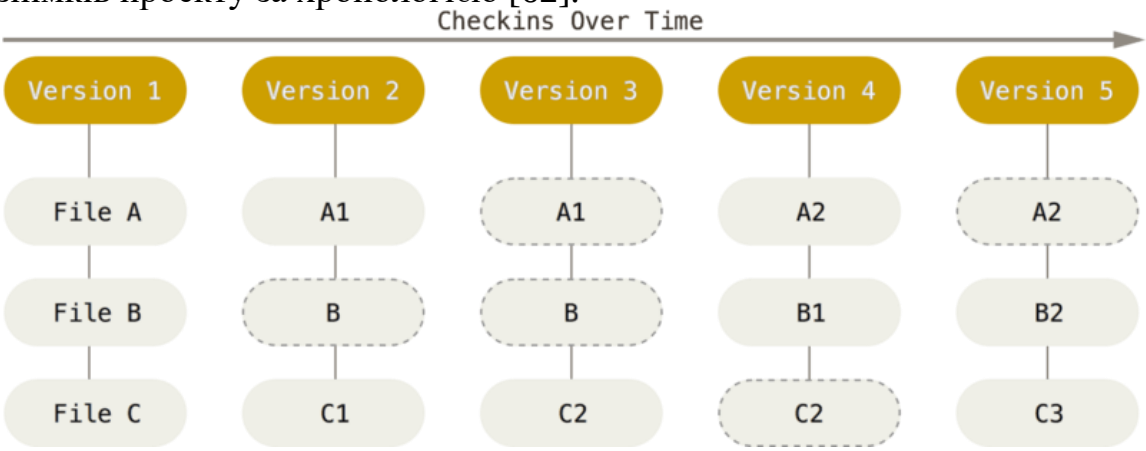


Рисунок 3.4.4 – Зберігання даних як знімків проекту за хронологією

Для одночасної роботи над проектом декількома людьми Git надає можливість створювати окремі гілки (форки). Форки створюються на основі певний гілок. Для роботи з гілками створили спеціальний набір правил, так створили методологію Gitflow. На рисунку 3.4.5 зображено приклад моделі розгалуження [63].

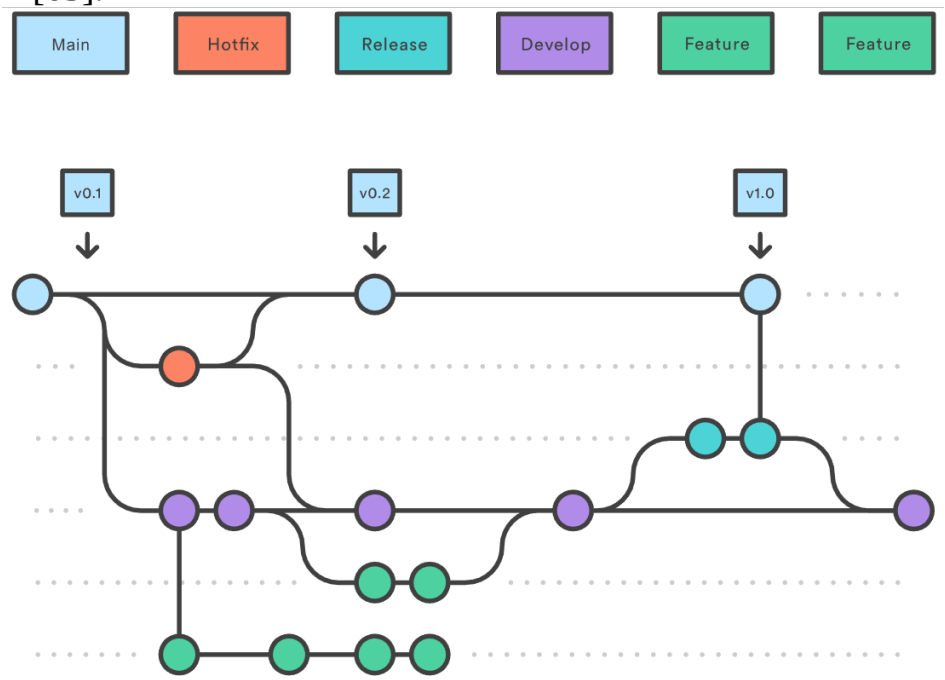


Рисунок 3.4.5 – Приклад моделі розгалуження за методологією Gitflow

В Gitflow для реєстрації історії робочого проекту замість одної гілки main використовуються дві. Головна гілка main зберігає офіційну історію релізу, а гілка develop зберігає повну історію проекту. Також до кожного коміту у гілці main прийнято додавати номер версії. Першим кроком у робочому процесі є створення гілки develop, що наслідується від main. Для цього простіше за все буде знаходячись на гілці main локально створити гілку develop та відправити її на сервер, для цього у терміналі Git необхідно ввести команди:

```
git branch develop
git push -u origin develop
```

Після цього інші розробники мають клонувати центральний репозиторій та створити гілку, що відслідковується для гілки develop [64].

Гілка Feature. Дана гілка є найбільш поширеним типом в робочому процесі Git, оскільки вона використовується для додавання нового функціоналу до проекту. Таким чином при роботі над новою функцією розробник повинен створити нову гілку Feature, що створюється на основі гілки Develop. По закінченню розробки функції та її тестування виконується злиття гілки Feature з гілкою Develop. Головне правило гілки Feature, таке, що для однієї функції – використовується одна гілка Feature. Отже, якщо команда працює одночасно, наприклад, над 3 функціями, то і гілок Feature має бути теж 3. Для створення гілки Feature необхідно знаходитися на гілці Develop та виконати наступні команди:

```
git checkout develop
git checkout -b feature_branch
```

Для злиття гілки Feature до Develop необхідно виконати наступні команди:

```
git checkout develop
git merge feature_branch
```

Гілка Release. При наближенні дати релізу, або при накопиченні достатньої кількості нових функцій/виправлень для випуску, то створюється гілка release на основі гілки develop. На даному етапі допустимим є виправлення багів, створення документації, але додавання нових функцій має відбуватися лише для наступного релізу. По закінченню підготовки гілки release для злиття з main їй задається номер версії. Також обов’язково необхідно виконати злиття гілки release з гілкою develop, оскільки з моменту створення гілки release на гілці розробки могли відбутися зміни. Завдяки такому розгалуженню команда розробників може поділитися на групи і таким чином одна група працюватиме над підготовкою релізу, інша – над функціями для наступного або ж виправленням багів. Для створення гілки release необхідно виконати наступні команди:

```
git checkout develop
git checkout -b release/0.1.0
```

Після завершення підготовки релізу дана гілка зливається з гілками main та develop, а сама гілка release видаляється. В даному випадку є доцільним використання запиту pull. Таким чином при завершені роботи з гілкою release необхідно виконати наступні команди:

```
git checkout develop
git merge release/0.1.0
git checkout main
git merge release/0.1.0
git push -d origin release/0.1.0
git branch -d release/0.1.0
```

Гілка Hotfix. Дана гілка призначена для підтримки та внесення швидких виправлень до релізів. Головною відмінністю гілки hotfix від release та feature є те, що створюється на основі гілки main, а не develop. Після виправлення помилок необхідно виконати злиття даної гілки з гілками main та develop або ж з поточною гілкою релізу. Також до гілки main необхідно додати оновлений номер версії. Таким чином завдяки спеціально виділеній гілці команда має можливість вносити виправлення не очікуючи наступного релізу [63, 64].

Для створення гілки hotfix необхідно виконати наступні команди:

```
git checkout main
git checkout -b hotfix_branch
```

Після завершення роботи з гілкою hotfix необхідно виконати її злиття з main та develop за допомогою наступних команд:

```
git checkout main
git merge hotfix_branch
git checkout develop
git merge hotfix_branch
git branch -D hotfix_branch
```

Для початку роботи з Git на комп'ютері, його необхідно встановити. Для цього необхідно перейти на офіційний сайт (https://git-scm.com/) та перейти до завантажень. На рисунку 3.4.6 зображено сторінку завантаження Git.

Рисунок 3.4.6 – Сторінка завантаження Git

Серед запропонованих версій програми необхідно обрати інсталятор відповідно до версії вашої операційної системи. Після завантаження та запуску інсталятора необхідно прийняти політики конфіденційності, обрати необхідні компоненти для встановлення та включити або виключити певні опції. На наступному кроці буде запропоновано обрати редактор для Git, за замовчуванням використовується редактор Vim, але також можна обрати інший із запропонованого списку. Вікно з вибором редактора зображено на рисунку 3.4.7.

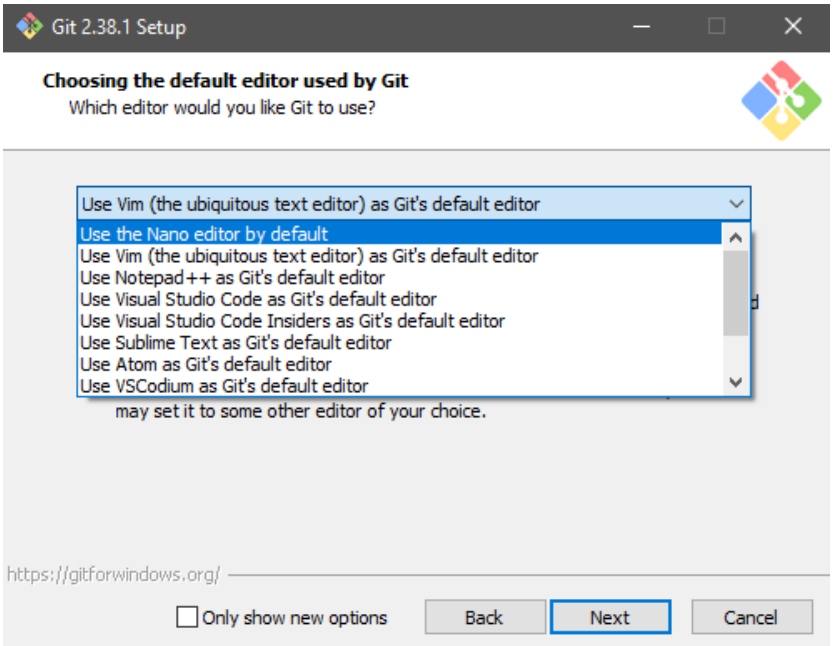


Рисунок 3.4.7 – Вибір редактору для Git

Після встановлення системи контролю версій Git у контекстному меню з'являться нові пункти, а саме: Git GUI Here, Git Bash Here. Для роботи з Git рекомендовано використовувати Git Bash, оскільки термінал має найновіші функції та дозволяє повністю контролювати дії з репозиторієм. На рисунку 3.4.8 зображено контекстне меню з пунктами Git GUI Here та Git Bash Here

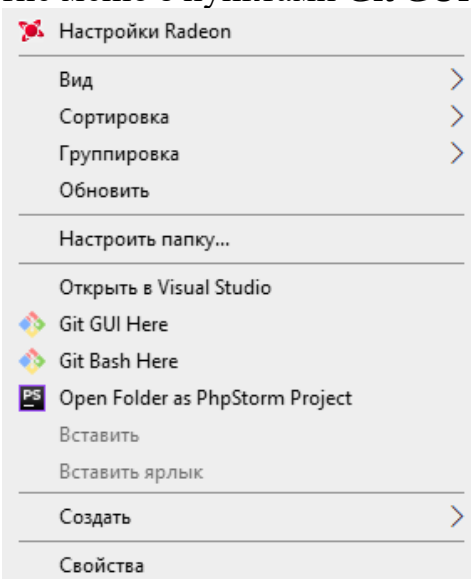


Рисунок 3.4.8 – Контекстне меню з пунктами для відкриття Git Bash або Git GUI

Наступним кроком необхідно налаштувати Git. Для початку необхідно вказати ім'я користувача та адресу електронної пошти. Для цього в контекстному меню необхідно обрати пункт Git Bash Here. У відкритому вікні необхідно ввести наступні команди:

```
git config --global user.name "My Username"
git config --global user.email email@example.com
```

Далі необхідно створити новий репозиторій. Для цього необхідно створити папку, де зберігатимуться файли проекту та запустити термінал Git з даного каталогу. У терміналі Git необхідно виконати наступну команду:

```
git init
```

Дана команда створює у поточному каталозі приховану папку «.git» у якій буде зберігатися історія репозиторію та його конфігурація. Після створення файлів їх необхідно додати до репозиторію, для цього необхідно скористатися командою «git add -A» або ж «git add <назва файлу>» для додавання конкретного файлу. Для фіксування внесених змін необхідно скористатися командою:

```
git commit -m "Initial commit"
```

Дана команда фіксує внесені зміни та додає опис до коміту, що міститься у подвійних лапках.

Але таким чином у нас створено лише локальний репозиторій, для використання усіх переваг децентралізованої системи контролю версій необхідно задіяти віддалений репозиторій. Найпопулярнішими віддаленими репозиторіями є Github, Bitbucket та Gitlab. Оскільки в мене вже є зареєстрований аккаунт Github, то буде використано саме його. Для початку необхідно створити віддалений репозиторій. Для цього треба перейти на сайт Github, авторизуватися та на головній сторінці натиснути кнопку «Create a new repository». На рисунку 3.4.9 зображено головну сторінку Github після авторизації.

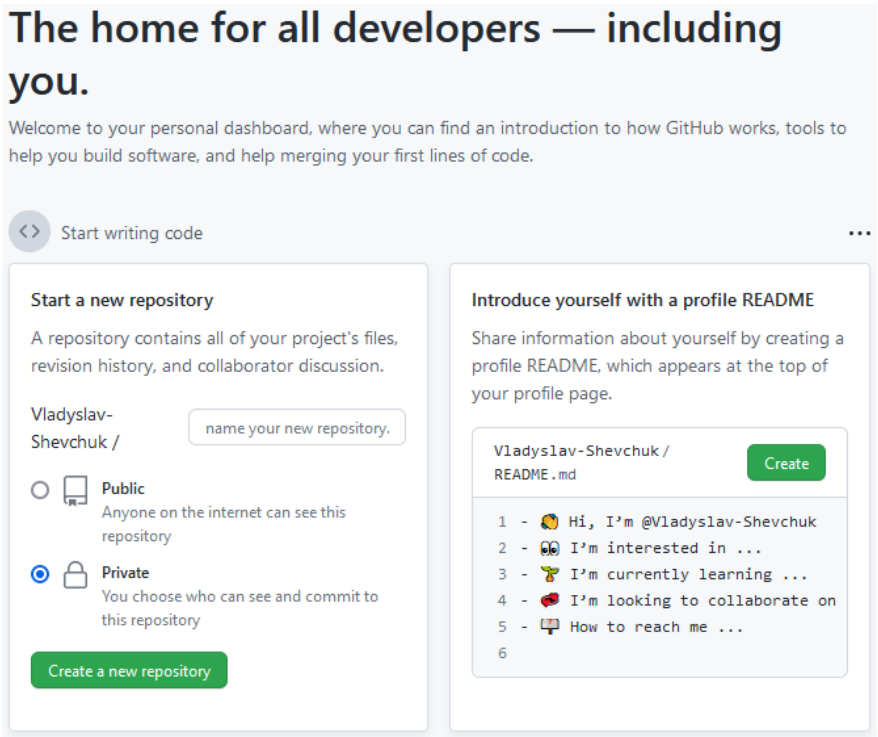


Рисунок 3.4.9 – Головна сторінка Github для авторизації

На сторінці створення нового репозиторію необхідно вказати його ім'я, опис, обрати тип репозиторію, є можливість додати файл README та файл .gitignore із застосуванням популярних шаблонів. Також на даній сторінці можна вказати файл ліцензії для обмеження дій з Вашим кодом.

На рисунку 3.4.10 зображено сторінку створення нового репозиторію.

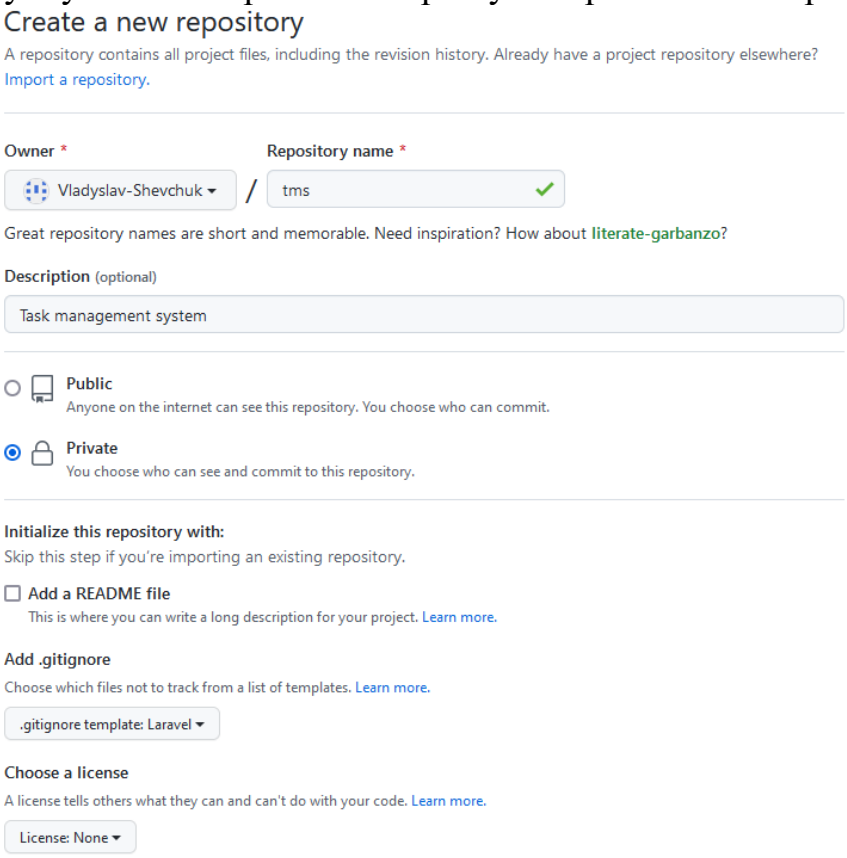


Рисунок 3.4.10 – Сторінка створення нового репозиторію

Після створення репозиторію можна зв’язати локальний репозиторій з віддаленим. Для цього необхідно перейти до каталогу з проектом та відкрити термінал Git, та ввести наступну команду:

```
git remote add origin «Посилання на репозиторій»
```

Посилання на репозиторій можна знайти на його сторінці в Github. На рисунку 3.4.11 зображено посилання на репозиторій проекту в Github.

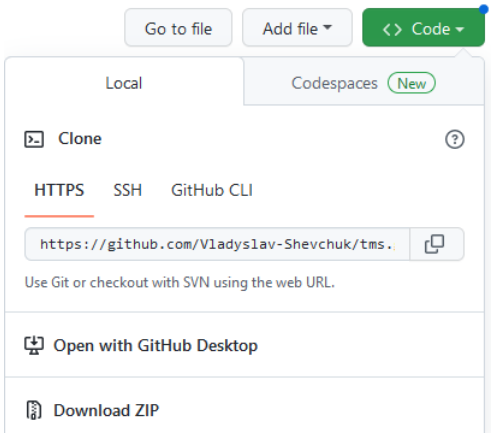


Рисунок 3.4.11 – Посилання на репозиторій проекту

Після налаштування з’єднання з віддаленим репозиторієм можна завантажити файли локального репозиторію до нього за допомогою команди:

```
git push origin master
```

Отже, далі можна перейти до розробки. Оскільки розробка буде проводитись із застосуванням фреймворку Laravel, то далі розглянемо архітектуру додатку з його застосуванням.

3.5 Архітектура додатку на основі фреймворку Laravel

Після створення проекту на основі фрейморку Laravel кореневий каталог містить наступні папки:

– app – папка, що містить основну частину додатку – моделі, контролери, команди та PHP-код домену;

– bootstrap – містить файли які виконують початкове завантаження фреймворку та відповідають за налаштування автоматичного завантаження класів.

– config - дана папка зберігає конфігураційні файли додатку.

– database – каталог що містить файли міграцій бази даних та файли «посіву» даних (Traits);

– public – папка, що містить статичні файли (css, js, images та інші);

– resources – містить шаблони відображення (Views), файли локалізації, робочі файли LESS або SASS та js-додатки із застосуванням фронтенд фреймворків таких, як: Ember, Vue, React та інші;

– storage – каталог, що має права для запису в нього ззовні, в даному каталозі Laravel зберігає скомпільовані Blade-шаблони, файловий кеш, файли сесії, файли логів та інші необхідні файли;

– test – дана папка використовується для зберігання юніт-тестів;

– vendor – використовується менеджером пакетів Composer для встановлення пакетів, що вказані у файлі composer.json [10865)].

Також в кореневому каталозі містяться файли, а саме:

– .editorconfig – містить інструкції для середовища розробки;

– .env та env.example – зберігають змінні середовища, зазвичай вони є різними в різних середовищах, тому файл .env прийнято не додавати до СКВ. Щодо env.example, то даний файл містить шаблон для створення файлу .env;

– .gitignore та .gitattributes – належать до конфігураційних файлів СКВ Git;

– artisan – надає доступ до команд утиліти Artisan у командному рядку;

– composer.json та composer.lock – це конфігураційні файли, що використовуються менеджером пакетів Composer;

– package.json – аналогічний файл до composer.json, але призначений для ресурсів клієнтської частини, містить команди для менеджера пакетів NPM та необхідні залежності;

– phpunit.xml – містить конфігурацію для інструменту PHPUnit, що використовується для тестування системи;

– readme.md – даний файл містить базові відомості про фреймворк, має формат Markdown;

– server.php – є резервним сервер, що дозволяє попередньо переглядати додаток Laravel;

– webpack.mix.js – це конфігураційний файл для Laravel Mix [66].

При розробці програмного забезпечення прийнято застосовувати архітектурні шаблони програмного забезпечення, що дозволяють закласти фундамент для додатку. Архітектурні шаблони дозволяють структурувати та організувати код програми залежно від функцій коду. Оскільки фреймворк Laravel базується на принципах архітектурного шаблону MVC, то розглянемо даний шаблон.

Шаблон MVC розшифровується як Model-View-Controller (Модель-Представлення-Контролер). Мета даного шаблону полягає у тому, щоб відділити інтерфейс додатку від моделі. Таким чином при внесенні змін до інтерфейсу не змінюється принцип роботи з даними, відсутня необхідність вносити зміни до моделі даних. Це робить дизайн більш гнучким. Даний патерн передбачає поділ функцій між компонентами. Так, модель являє собою представлення таблиці бази даних, містить бізнес-логіку додатку [67].

Представлення являє собою шаблон відображення даних, містить HTML, CSS, а також JavaScript код. У проекті може бути множина представлень, що застосовуються для певних сторінок. Також залежно від ролі та прав користувача можуть застосовуватися різні представлення даних.

Контролер відповідає за отримання HTTP-запиту, отримання потрібних даних з бази даних, відповідає за валідацію даних та повернення відповіді користувачеві. На рисунку 3.5.1 зображено патерн MVC [66].

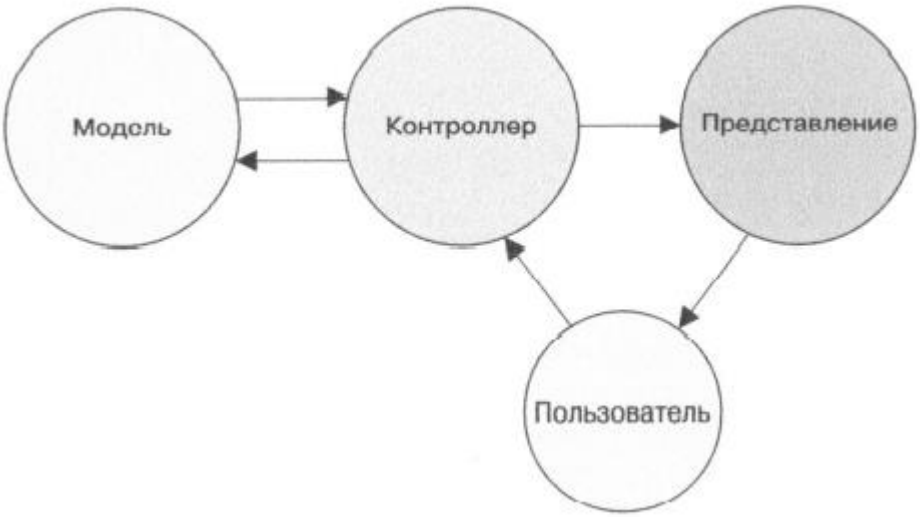


Рисунок 3.5.1 – Патерн MVC

Даний шаблон чітко прослідковується у каталозі app, що містить підкаталоги Controllers та Models. Щодо представлень, то вони розташовуються у каталозі resources.

У створюваному веб-додатку існуватимуть наступні ролі та права:

− admin – користувач, що має необмежені права та доступ до системи контролю, виконує первинне налаштування системи та подальшу підтримку її працездатності;

− manager – користувач, що несе відповідальність за вчасне виконання завдань, проводить моніторинг прогресу виконання завдань на регулярній основі, має доступ до створення нових завдань, редагуванню вже створених та видаленню неактуальних, має доступ до звітів;

− secretary – користувач, що несе відповідальність за розподіл задач між підлеглими, має доступ до створення/редагування та видалення задач, має доступ до звітів. Дана роль створена для явного розмежування зони відповідальності працівника

− worker – користувач, що відповідає за виконання завдань, має доступ до переліку задач призначених на нього, має можливість змінювати статус задач.

Оскільки manager та secretary мають однакові права в системі, то для них було створено групу Management. Таким чином каталог представлень містить 2 підкаталоги для підставлень сторінок на основі ролі: Admin та Management. Окремого представлення для користувачів з роллю worker не створювалось, оскільки передбачається, що взаємодія з додатком буде здійснюватися через Telegram-бот. Також, було створено відповідні групи у контролерах.

Далі розглянемо процес розробки веб-додатку.

3.6 Процес розробки додатку

Для початку розробки додатку необхідно перейти до каталогу проекту та викликати командний рядок. У командному рядку необхідно виконати команду:

```
laravel new назва_додатку
```

Дана команда створить новий проект Laravel з вказаною назвою додатку. На рисунку 3.6.1 зображено повідомлення у консолі про успішну підготовку проекту.

```
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi
Discovered Package: facade/ignition
Discovered Package: fideloper/proxy
Discovered Package: fruitcake/laravel-cors
Discovered Package: laravel/sail
Discovered Package: laravel/tinker
Discovered Package: nesbot/carbon
Discovered Package: nunomaduro/collision
Package manifest generated successfully.
74 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
> @php artisan key:generate --ansi
Application key set successfully.

Application ready! Build something amazing.
```

Рисунок 3.6.1 – Повідомлення про готовність проекту для подальшої розробки

На даному етапі при переході за адресою локального веб-сервера буде відображено стандартну сторінку привітання Laravel. На рисунку 3.6.2 наведено стандартну головну сторінку додатку Laravel.

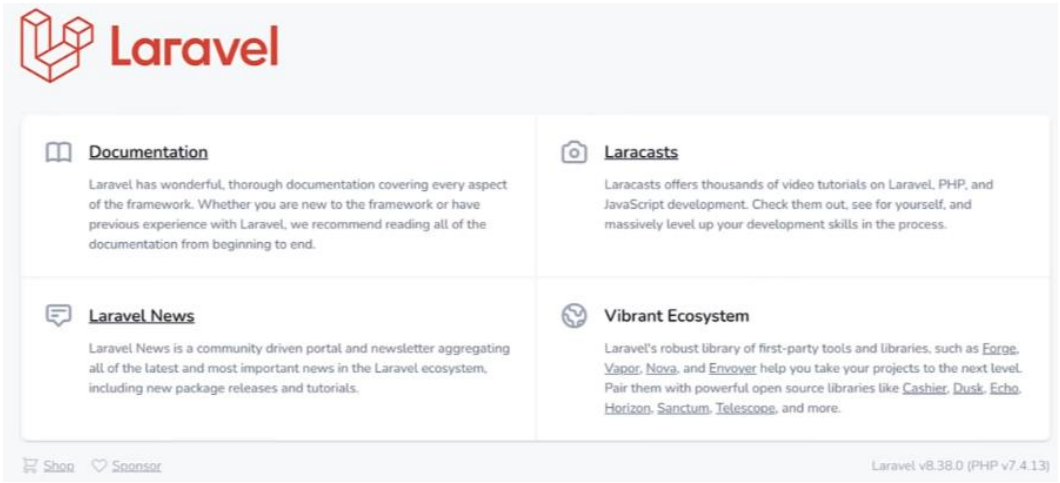


Рисунок 3.6.2 – Стандартна головна сторінка Laravel

Наступним кроком необхідно підключити авторизацію. Для цього у командному рядку необхідно виконати команди:

```
composer require laravel/ui
php artisan ui bootstrap –auth
npm run & npm dev
```

Таким чином було створено стандартні форми реєстрації та авторизації користувачів. На рисунку 3.6.3 зображено форму реєстрації користувачів.

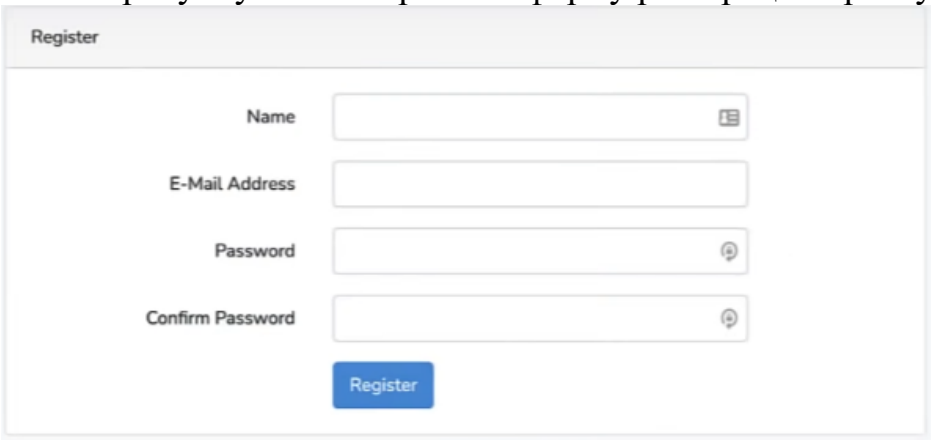


Рисунок 3.6.3 – Форма реєстрації користувачів

Оскільки додавання користувачів буде здійснюватися адміністратором, то доступ до даної форми буде обмежено. На рисунку 3.6.4 зображено форму авторизації користувачів.

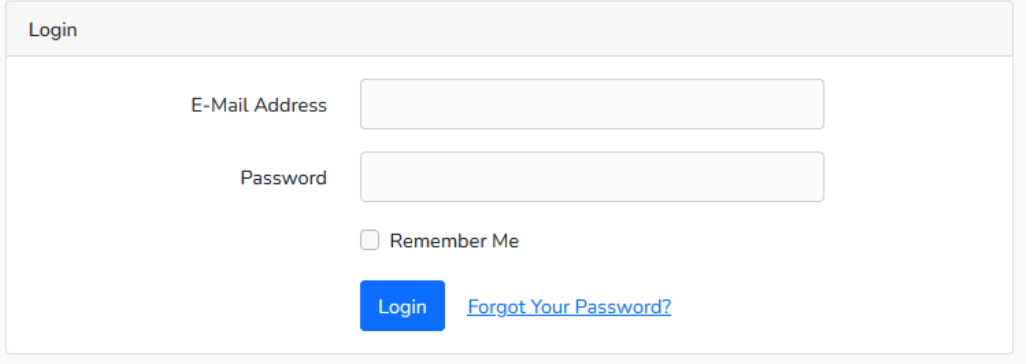


Рисунок 3.6.4 – Форма авторизації користувачів

Окрім даних форм було створено міграцію для створення таблиці Users, що зберігається у файлі 2014_10_12_000000_create_users_table.php. Лістинг коду, що відповідає за створення таблиці:

```
public function up()
    {
       Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('email')->unique();
            $table->timestamp('email_verified_at')
             ->nullable();
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
    }
```

Як видно з лістингу, то за замовчуванням створюються наступні поля:

– Id – дане поле є автоінкрементним, що містить унікальний ідентифікатор запису, в даному випадку унікальний ідентифікатор користувача;

– Name – використовується для зберігання імені користувача;

– Email – призначене для зберігання електронної пошти користувача;

– Email_verified_at – зберігає дату та час підтвердження пошти;

– Password – зберігає хеш паролю;

– rememberToken – створює стовбець remember_token, що зберігає ключі сесій при використанні функції «запам’ятати мене»;

– timestamps – створює стовпці created_at та updated_at, що зберігають мітки дати та часу створення облікового запису користувача та його оновлення.

Окрім зазначеної таблиці також створюються такі таблиці, як: personal_access_tokens, failed_jobs та password_resets.

Для створення даних таблиць необхідно виконати команду:`php artisan migrate`.

Для спрощення розробки користувацького інтерфейсу було прийнято рішення використовувати бібліотеку компонентів користувацького інтерфейсу CoreUI, що підтримує інтеграцію з набором інструментів Bootstrap. Бібліотеку CoreUI було підключено у шаблонах за допомогою посилань на cdn (мережу постачання контенту) [68].

Далі за було виконано команду: `php artisan make:migration create_roles_tables`. У створеному файлі міграції було описано необхідні поля для таблиці ролей. Лістинг коду міграції, що відповідає за створення таблиці наведено нижче:

```
public function up() {
       Schema::create('roles', function (Blueprint $table) {
            $table->id();
            $table->string('name')->unique();
        });}
```

Також необхідно додати поле ролі до таблиці користувачів, та вказати зв'язок між полем ролі та таблицею ролей. Лістинг коду додавання поля role_id до таблиці користувачів:

```
public function up()
    {
        Schema::table('users', function (Blueprint $table) {
            $table->unsignedBigInteger('role_id')
             ->nullable()->default(null);
            $table->foreign('role_id')
                ->references('id')
                ->on('roles');
        });
    }
```

На наступному кроці було створено таблицю задач. Лістинг коду створення таблиці задач:

```
public function up(){
Schema::create('tasks', function (Blueprint $table) {
            $table->id();
            $table->string("title");
            $table->string("description");
            $table->foreignId("executor_id")
                ->references("id")
                ->on("users");
            $table->foreignId("assigned_by_id")
                ->references("id")
                ->on("users");
            $table->foreignId("supervisor_id")
                ->references("id")
                ->on("users");
            $table->date("deadline");
            $table->timestamps();
        });
```

Далі було створено таблицю відділів. Лістинг коду створення таблиці divisions:

```
public function up(){
   Schema::create('divisions', function (Blueprint $table) {
            $table->id();
            $table->string('name')->unique();
        });
    }
```

Наступним кроком було додано поле division_id до таблиці користувачів. Лістинг коду даної міграції, що відповідає за додавання поля:

```
public function up(){
       Schema::table('users', function (Blueprint $table) {
            $table->unsignedBigInteger('division_id')
             ->nullable()->default(null);
            $table->foreign('division_id')
                ->references('id')
                ->on('divisions');
        });
    }
```

Аналогічним чином було створено таблицю статусів задач, що містить поле Id та унікальне поле Name (статус задачі). Але дана таблиця має бути наповнена конкретними значеннями. Для наповнення бази даних Laravel надає клас DatabaseSeeder. Для створення наповнювача необхідно виконати наступну команду: `php artisan make:seeder TaskStatusesTableSeeder`. Далі було створено функцію заповнення даними таблиці task_statuses:

```
public function run()
    {
        TaskStatus::truncate();
        $task_statuses =  [
            ['id' => 1, 'name' => 'Open'],
            ['id' => 2, 'name' => 'To do'],
            ['id' => 3, 'name' => 'In progress'],
            ['id' => 4, 'name' => 'Done'],
            ['id' => 5, 'name' => 'Closed'],
            ['id' => 6, 'name' => 'Cancelled'           ]
        ];
        TaskStatus::insert($task_statuses);
    }
```

Для запуску наповнювача було використано команду: `php artisan db:seed --class=UserSeeder`.

Також було додано поле task_statuses_id до таблиці задач. Далі було додано поле phone_number до таблиці користувачів та додано поле closed_at до таблиці задач для відслідковування дати закриття задач. Також було додано поля telegram_key та chat_id до таблиці користувачів для подальшої інтеграції з Telegram-ботом. Таким чином було створено базу даних, кінцеву базу даних зображено на діаграмі у додатку Б.

Окрім внесених змін до бази даних треба створити відповідні моделі даних. Так, у моделі User необхідно вказати зв’язки між стовпцями та таблицями за допомогою ключових полів. Нижче наведено лістинг коду встановлення зв’язків між стовпцями та таблицями:

```
public function role(){
        return $this->belongsTo(Role::class);
    }
    public function division(){
        return $this->belongsTo(Division::class);
    }
    public function task_executor(){
        return $this->hasMany(Task::class, 'executor_id');
    }
    public function task_assigned_by(){
       return $this->hasMany(Task::class, 'assigned_by_id');
    }
    public function task_supervisor(){
        return $this->hasMany(Task::class, 'supervisor_id');
    }
```

Таким чином у контролері UserController є можливість використання зв’язку role. Розглянемо детальніше використання зв’язку. Нижче наведено

лістинг функції, що повертає список користувачів для відображення на сторінці Users для адміністраторів:

```
public function index()
    {
        $users = User::with('role', 'division')
            ->orderBy('role_id', 'DESC')
            ->paginate(10);
        return view('admin.users.index', compact('users'));
    }
```

Завдяки таким зв'язкам у представленні можна відобразити замість role_id назву ролі користувача. Лістинг частини коду шалону таблиці для відображення списку користувачів адміністратуру:

```
@foreach($users as $user)
<tr class="align-middle">
  <td class="text-center">{{ $user->id }}</td>
  <td class="text-center"> {{$user->name}}</td>
  <td class="text-center"> {{$user->role->name}}</td>
  <td   class="text-center">   {{$user->division   ?   $user-
>division->name : "no assigned"}}</td>
  <td class="text-center"> {{$user->phone_number}}</td>
   <td>
```

На рисунку 3.6.5 зображено список користувачів, що відображається адміністратору.

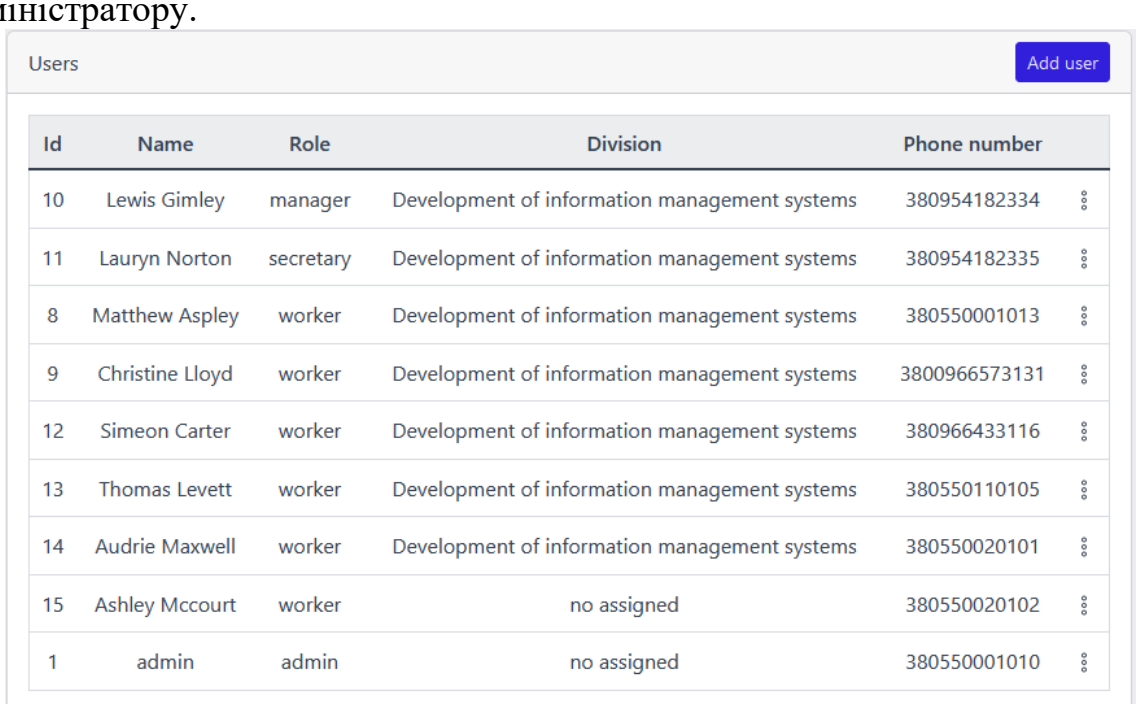


Рисунок 3.6.5 – Список користувачів який бачить адміністратор

Як видно з рисунку 3.6.5, замість ідентифікатору ролі дійсно відображається її назва. Подібним чином було встановлено зв'язки у інших моделях.

Для порівняння зовнішнього вигляду списку користувачів на рисунку 3.6.6 було наведено приклад відображення даного списку для менеджерів та секретарів.

Users

Id	Name	Role	Phone number	Asiggned tasks	In progress
10	Lewis Gimley	manager	380954182334	0	0
11	Lauryn Norton	secretary	380954182335	0	0
8	Matthew Aspley	worker	380550001013	4	1
9	Christine Lloyd	worker	3800966573131	3	1
12	Simeon Carter	worker	380966433116	4	1
13	Thomas Levett	worker	380550110105	2	0
14	Audrie Maxwell	worker	380550020101	3	1

Рисунок 3.6.6 – Список користувачів який бачать менеджери та секретарі

Як видно з рисунку 3.6.6 менеджери та секретарі бачать лише тих користувачів, що відносяться до їх відділу. Також менеджери та секретарі бачать додаткові два поля «Assigned tasks» та «In progress». Поле «Assigned tasks» відображає кількість задач, що призначено для виконання певним користувачем. У свою чергу поле «In progress» відображає кількість задач, що виконується користувачем на даний час.

Для обмеження доступу до певного функціоналу в Laravel існують шлюзи (Gates). Зазвичай, шлюзи створюються у методі boot класу App\Providers\AuthServiceProvider з використанням фасаду Gate. Шлюзам завжди першим параметром передають екземпляр користувача, а також можуть отримувати додаткові аргументи, наприклад модель Eloquent. Лістинг коду метода boot:

```
    public function boot(){
            $this->registerPolicies();
            Gate::define('access_to_admin_func', function (User
$user) {return $user->is_admin();});
            Gate::define('access_to_management_func',   function
(User $user) {return $user->is_manager_or_secretary();});
        }
```

Згідно приведеного коду видно, що було створено два шлюзи, а саме «access_to_admin_func» та «access_to_management_func». Перший з них викликає метод is_admin класу User, що повертає true або false залежно від того чи є поточний користувач адміністраторам або ж ні. Другий шлюз викликає метод is_manager_or_secretary теж класу User, який повертає true або false залежно від того чи є користувач менеджером або секретарем. Дані функції попередньо було створено у моделі User. Лістинг функцій is_admin та is_manager_or_secretary:

```
    public function is_admin(){
            return $this->role_id == Role::IS_ADMIN;
        }
        public function is_manager_or_secretary() {
            return $this->role_id == Role::IS_MANAGER || $this-
>role_id == Role::IS_SECRETARY;}
```

З коду видно, що у даних функціях виконується перевірка чи відповідає ідентифікатор поточного користувача константі моделі Role.

Розглянемо приклад використання шлюзів. Так, у шаблоні sidebar, використовуючи директиву can шаблонізатора Blade можна відобразити різні пункти меню залежно від ролі користувача. Лістинг коду файлу «sidebar.blade.php», що відповідає за відображення пунктів меню в залежності від ролі користувача наведено додатку В.

Але подібний спосіб приховування контенту є недієвим. Оскільки, якщо користувач все ж отримає посилання на сторінку, яку він не має бачити, то він зможе перейти на неї. Для усунення даної проблеми можна використати додатково проміжне програмне забезпечення або ж посередника (Middleware). Для створення посередника необхідно виконати команду: `php artisan make:middleware IsAdmin`. Далі було створено функцію перевірки чи є користувач адміністратором. Лістинг класу IsAdmin:

```
public function handle(Request $request, Closure $next){
        if((auth()->user()->role_id != Role::IS_ADMIN))
            return redirect()->route('welcome');
        return $next($request);
    }
```

Таким чином, якщо користувач не є адміністратором, але спробує перейти за посиланням що веде до списку відділів (наприклад), то його буде переадресовано на головну сторінку (сторінка «welcome» використовується у якості головної). Подібним чином було створено посередника IsManagerOrSecretary. Також важливо зареєструвати посередника. Для цього необхідно у файлі «app/Http/Kernel.php» додати до властивості $routeMiddleware рядок підключення посередника. Окрім цього необхідно вказати посередника у файлі «web.php». На рисунку 3.6.7 зображено вміст файлу «web.php».

```
Route::get( uri: '/', [\App\Http\Controllers\WelcomeController::class, 'index'])->name( name: 'welcome');
Auth::routes();
Route::group(['middleware' => 'auth'], function () {
    Route::group([
        'prefix' => 'admin',
        'middleware' => 'is_admin',
        'as' => 'admin.'
    ], function () {
        Route::resource( name: 'divisions', controller: \App\Http\Controllers\Admin\DivisionsController::class);
        Route::resource( name: 'users', controller: \App\Http\Controllers\Admin\UserController::class);
        Route::resource( name: 'tasks', controller: \App\Http\Controllers\Admin\TasksController::class);
    });
    Route::group([
        'prefix' => 'management',
        'middleware' => 'is_manager_or_secretary',
        'as' => 'management.'
    ], function () {
        Route::resource( name: 'tasks', controller: \App\Http\Controllers\Management\TasksController::class);
        Route::resource( name: 'users', controller: \App\Http\Controllers\Management\UserController::class);
        Route::get( uri: 'daily-report', action: '\App\Http\Controllers\Management\CsvReports\ReportsController@getDailyReport')
            ->name( name: 'daily-report');
        Route::get( uri: 'monthly-report', action: '\App\Http\Controllers\Management\CsvReports\ReportsController@getMonthlyReport')
            ->name( name: 'monthly-report');
        Route::get( uri: 'next-month-report', action: '\App\Http\Controllers\Management\CsvReports\ReportsController@getNextMonthReport')
            ->name( name: 'next-month-report');
    });
});
```

Рисунок 3.6.7 – Вміст файлу «web.php»

Для користувачів, що мають роль менеджера або секретаря було додано можливість завантажувати звіти. Для цього було створено ReportsController. Лістинг даного контролеру наведено у додатку Г. Створена система надає змогу отримувати наступні типи звітів:

– daily report – щоденний звіт, який надає список задач, що знаходяться на виконанні;

– monthly report – місячний звіт, містить перелік задач, що було виконано за минулий тиждень;

– plan for next month – тип звіту, що містить перелік задач термін виконання яких спливає наступного місяця.

Аби сформувати щоденний звіт необхідно обрати усі задачі за відділом та відфільтрувати за поточною датою. Використовуючи Eloquent модель даний запит виглядає наступним чином:

```
    $tasks_rel = Task::with('task_executor', 'task_assigned_by',
'task_supervisor')
    ->whereHas('task_executor', function ($query) {
                $query->where('division_id',     auth()->user()-
>division_id);
            })
                ->where('task_statuses_id', 3)
                ->orderBy('deadline', 'ASC')
                ->get();
```

У 7 рядку вказано відповідність поля «task_statuses_id» цифрі 3, це зроблено для фільтрування задач за статусом, а саме для отримання тих задач які знаходяться на виконанні.

Для формування звіту про виконані задачі за минулий місяць необхідно виконати схожий на попередній запит, але тепер необхідно виконати фільтрацію за датою у полі «closed_at» у межах між першим та останнім днем попереднього місяцю. Даний запит матиме вигляд:

```
    $firstDayPreviousMonth = date("Y-m-d", strtotime("first day
of previous month"));
            $lastDayPreviousMonth        =        date("Y-m-d",
strtotime("last day of previous month"));
            $tasks_rel       =       Task::with('task_executor',
'task_assigned_by', 'task_supervisor')->whereHas('task_executor',
function ($query) {
                $query->where('division_id',     auth()->user()-
>division_id);
            })
                ->where('task_statuses_id', 5)
                ->whereBetween('closed_at',
[$firstDayPreviousMonth, $lastDayPreviousMonth])
                ->orderBy('deadline', 'ASC')
                ->get();
```

Як видно з коду для зберігання дати першого та останнього дня попереднього місяця було створено змінні та використано функцію date. Функція date приймає на вхід один обов’язковий параметр – формат дати. Додатково функція date приймає необов’язковий параметр timestamp, що за

замовчуванням відповідає поточному локальному часу. Параметр timestamp є цілочисельним типом та допускає значення null. Також для отримання дати було використано функцію strtotime, що виконує перетворення текстового представлення дати на мітку часу Unix. Для фільтрації за датою закриття задач було використано метод whereBetween.

Для формування звіту про список задач на наступний місяць було створено функцію getNextMonthReport. Для отримання задач, термін виконання яких спливає наступного місяця необхідно виконати наступний запит:

```
    $firstDayNextMonth = date("Y-m-d", strtotime("first day of
next month"));
            $lastDayNextMonth = date("Y-m-d", strtotime("last day
of next month"));
            $tasks_rel       =       Task::with('task_executor',
'task_assigned_by',     'task_supervisor',     'task_statuses')-
>whereHas('task_executor', function ($query) {
                $query->where('division_id',     auth()->user()-
>division_id);
            })
                ->where('task_statuses_id', '!=', 6)
                ->whereBetween('deadline',  [$firstDayNextMonth,
$lastDayNextMonth])
                ->orderBy('deadline', 'ASC')
                ->get();
```

Даний запит схожий на попередній, але фільтрація звітів виконується по полю deadline. Також для сортування задач за кінцевою датою було використано метод orderBy. За допомогою даного методу задачі сортуються у порядку зростання дати.

Для виконання запитів використовується об’єктно-реляційна проекція (Object-relational mapping, ORM, OPM) Eloquent. За допомогою технології OPM спрощується робота з СУБД, замість написання запитів взаємодія з БД реалізується через операції з об’єктами. Так, у проекті створюються окремі моделі, що відповідають кожній з наявних таблиць. У кожній моделі описуються поля таблиці, зв’язки по ключовим полям та користувацькі методи роботи з таблицею. Таким чином моделі надають можливість виконувати запити даних з таблиць, створювати нові записи та редагувати існуючі. На рисунку 3.6.8 зображено структуру ORM-технології [69].

Рисунок 3.6.8 – Структура ORM-технології

Далі на рисунку 3.6.8 наведено приклад редагування запису з таблиці User.

```php
public function update(Request $request, $id)
{
    $user = User::findOrFail($id);
    $request->validate([
        'name' => 'required|min:4|string|max:255',
        'email' => 'required|email|string|max:255',
        'phone_number' => 'required|numeric|min:10',
    ]);
    $input = $request->input();
    if (Role::where('id', '=', $input['role_id'])->count() <= 0)
        throw ValidationException::withMessages(['role' => 'The specified role does not exist!']);
    $user->name = strip_tags($input['name']);
    $user->email = strip_tags($input['email']);
    $user->role_id = strip_tags($input['role_id']);
    $user->phone_number = strip_tags($input['phone_number']);
    if ((int)$input['division_id'])
        $user->division_id = strip_tags($input['division_id']);
    else
        $user->division_id = null;
    if (strlen(trim($input['password'])) > 0) {
        $request->validate(['password' => ['required', 'string', 'min:8', 'max:20']]);
        $user->password = Hash::make($input['password']);
    }
    try {
        $user->save();
    } catch (\Exception $exception) {
        return back()->withError($exception->getMessage())->withInput();
    }
    return \Redirect::route('admin.users.edit', [$user->id])->with('message', 'User has been updated!');
}
```

Рисунок 3.6.9 – Оновлення запису в таблиці User

Як видно з рисунку 3.6.9 для оновлення запису таблиці User необхідно для початку отримати його. Для отримання запису здійснюється пошук за ідентифікатором запису, що було передано з представлення форми редагування. Далі оголошуються правила валідації полів та отримання введених даних до форми зі змінної request, що містить дані запиту. Наступним кроком виконується перевірка чи існує вказана користувачем роль, це зроблено для забезпечення цілісності даних. Після цього виконується перезапис полів поточного запису таблиці User відповідними даними із заповненої форми. Для зберігання паролю застосовується метод make класу Hash для створення хешу паролю. Після внесення змін до об’єкту запису необхідно викликати метод save.

Використання хешування є обов’язковою операцією для під час розробки додатку. При зберіганні паролю у «чистому» створюється небезпека викрадення паролю у разі компрометації бази даних. Використання хешування паролів перед збереженням паролю до бази даних унеможливлює його розгадування і в той же час зберігається можливість порівняння отриманого хешу з оригінальним паролем [70].

Отже, було розглянуто основні аспекти створення веб-додатку, далі буде розглянуто інтеграцію Telegram API до додатку.

3.7 Інтеграція Telegram API

Для створення Telegram-боту спочатку необхідно знайти бота @BotFather у Telegram та створити нового бота за допомогою команди /newbot. Далі необхідно ввести назву боту після чого Ви отримаєте токен доступу HTTP API.

Отриманий токен потрібен для встановлення веб-хуку за яким отримуватимемо повідомлення від боту. Для роботи веб-хуків сервер, що приймає запити має використовувати HTTPS з'єднання. Тобто веб-додаток має знаходитися на хостингу, що надає SSL сертифікат. Але можна скористатися мережевою утилітою Ngrok. Дана утиліта надає тимчасовий домен з HTTPS з'єднанням. Для його запуску необхідно перейти на офіційний сайт (https://ngrok.com/) та завантажити його. На рисунку 3.7.1 зображено форму завантаження утиліти Ngrok.

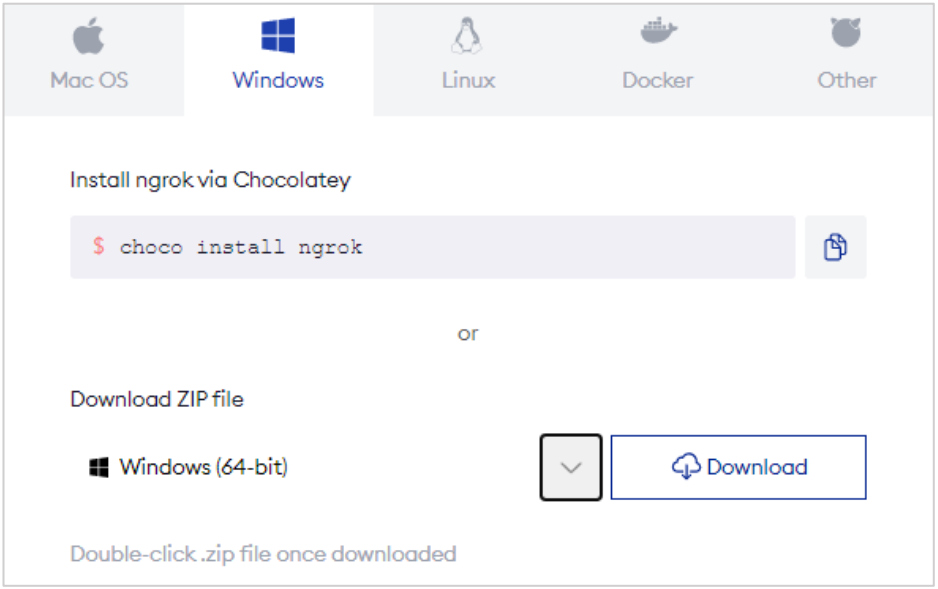


Рисунок 3.7.1 – Форма завантаження утиліти Ngrok

Дана утиліта не потребує інсталяції, достатньо розпакувати завантажений архів та запустити файл «ngrok.exe». Для отримання домену необхідно ввести команду: `ngrok http --host-header=rewrite tms.lan:80`. Після чого у консолі з'явиться посилання на домен з HTTPS. На рисунку 3.7.2 зображено вікно ngrok з тимчасовим доменом.

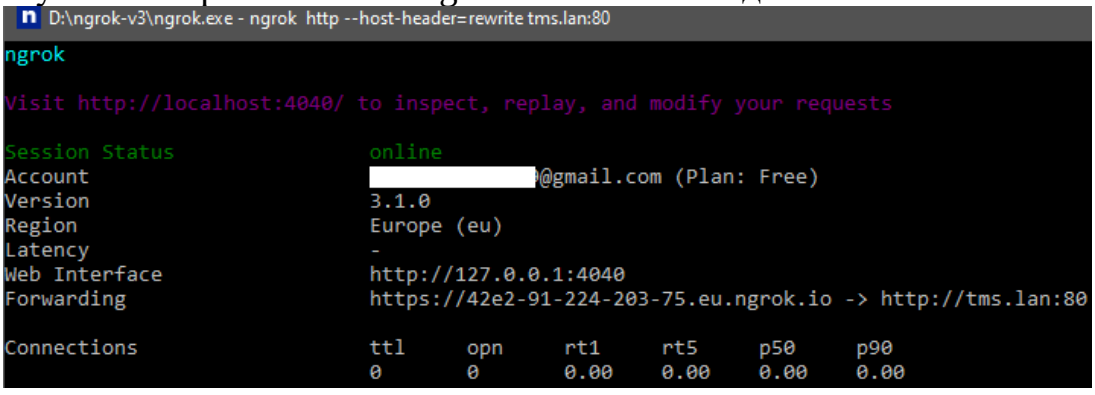


Рисунок 3.7.2 – Вікно ngrok з тимчасовим доменом.

Далі необхідно встановити веб-хук за вказаною адресою (https://42e2-91-224-203-75.eu.ngrok.io). Для цього виконаємо GET-запит (або ж можна POST, telegram api підтримує обидва) за наступною адресою:

```
https://api.telegram.org/botAPI_KEY/setWebhook?url=https://42e2-91-224-203-75.eu.ngrok.io/telegram
```

Де, API_KEY відповідає токену доступу, що було отримано від BotFather. Після успішного встановлення веб-хуку отримуємо відповідне повідомлення. На рисунку 3.7.3 зображено відповідь про успішне встановлення веб-хуку.

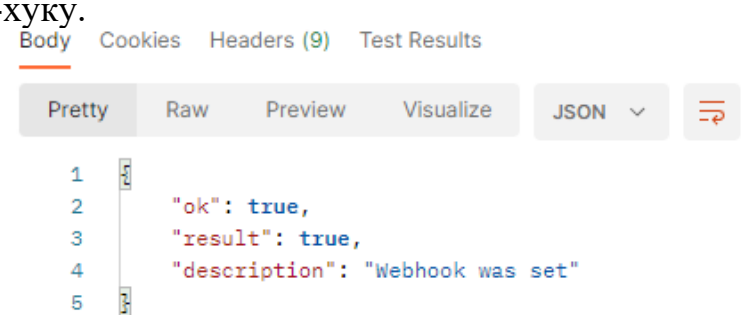


Рисунок 3.7.3 – Повідомлення про успішне встановлення веб-хуку

Для обробки вхідних повідомлень було створено TelegramController та модель Telegram. У моделі Telegram зберігається API_KEY боту та методи для обробки повідомлень та надсилання відповідей користувачам. У додатку Д наведено лістинг TelegramController, а у додатку Е наведено лістинг моделі Telegram.

Для початку роботи з ботом необхідно перейти за запрошувальним посиланням, що має надіслати адміністратор. Також можна авторизуватися у системі Tms (Task management system) та на головній сторінці скористатися посиланням на Telegram. На рисунку 3.7.4 зображено головну сторінку розроблюваного додатку Tms.

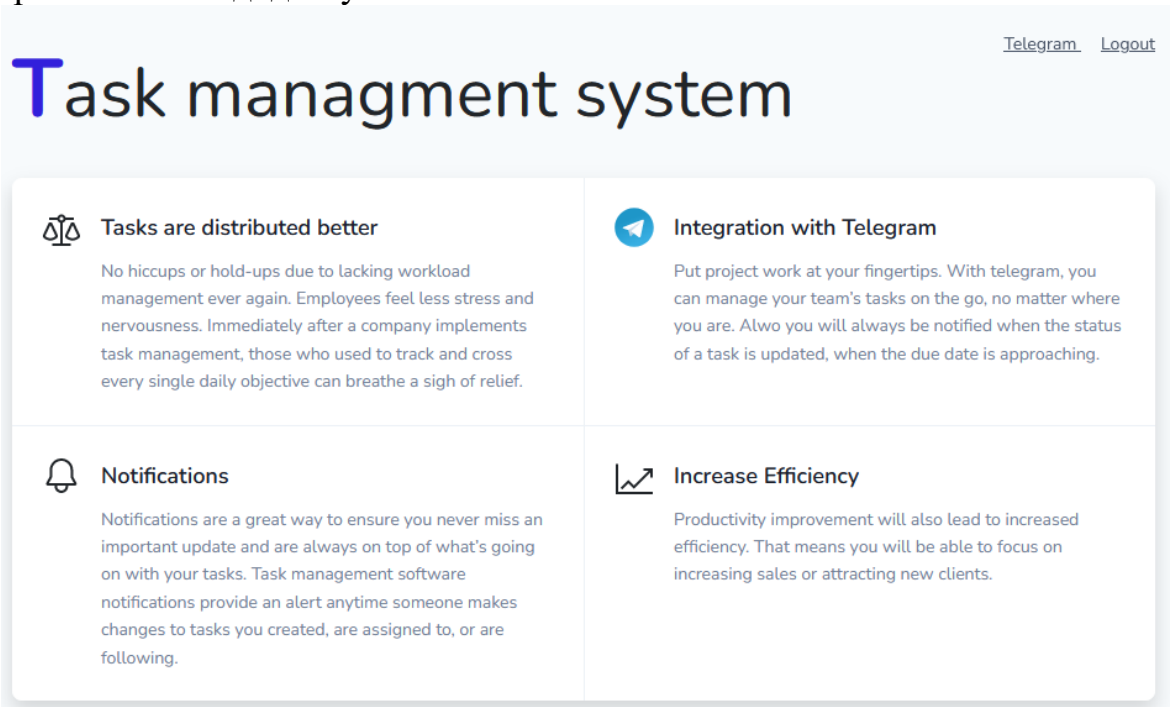


Рисунок 3.7.4 – Головна сторінка Tms

При переході за посиланням відкривається веб-версія telegram де необхідно обрати спосіб використання telegram. На рисунку 3.7.5 зображено сторінку вибору версії telegram.

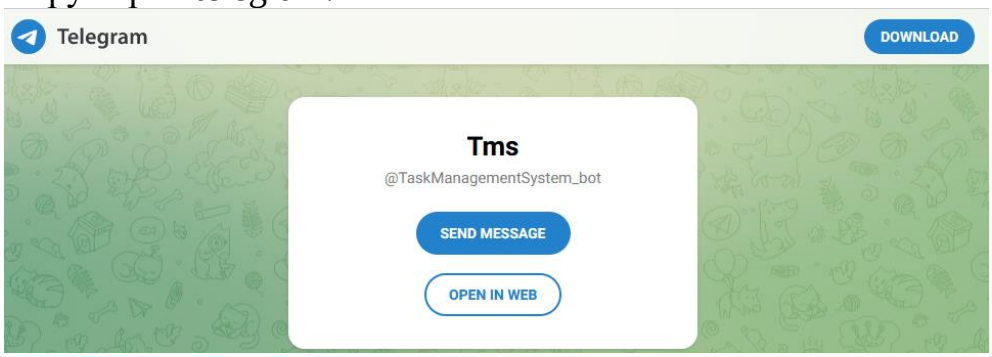


Рисунок 3.7.5 – Вибір версії Telegram

При наявності встановленого додатку Telegram він відкривається автоматично. Для початку роботи з ботом необхідно натиснути «START». У разі, якщо посилання виявилося пошкодженим, то користувач отримає повідомлення: «The invitation link is invalid!». Також не можна розпочати роботу з ботом при прямому переході до боту, в такому випадку користувач отримає повідомлення: «Please use the invitation link!». У разі, якщо користувач перейшов за посиланням зі сторінки додатку Tms, то після натискання кнопки старт він отримає повідомлення з привітанням. Також користувачу стане доступним перегляд списку його задач. На рисунку 3.7.6 зображено повідомлення з вітанням та меню користувача.

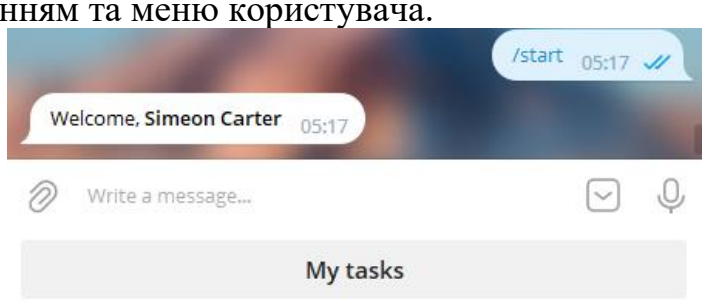


Рисунок 3.7.6 – Повідомлення з привітанням та меню працівника

При натисканні на кнопку «My tasks» користувач отримує список своїх задач з поточними їх статусами та кінцевими датами для виконання який наведено на рисунку 3.7.7. Для зручності навігації задачі було пронумеровано, підпис статусу задач виділено курсивом, а сам статус оформлено стилем коду. Також курсивом було виділено підпис Deadline, а кінцева дата виконання задачі виділено жирним шрифтом. Для оформлення списку було застосовано мову розмітки Markdown, що використовується Telegram. Таким чином можливі наступні варіанти оформлення тексту:

```
**жирний**
__курсив__
`код`
~~перекреслений~~
```блок коду```
||скритий текст||
```
```

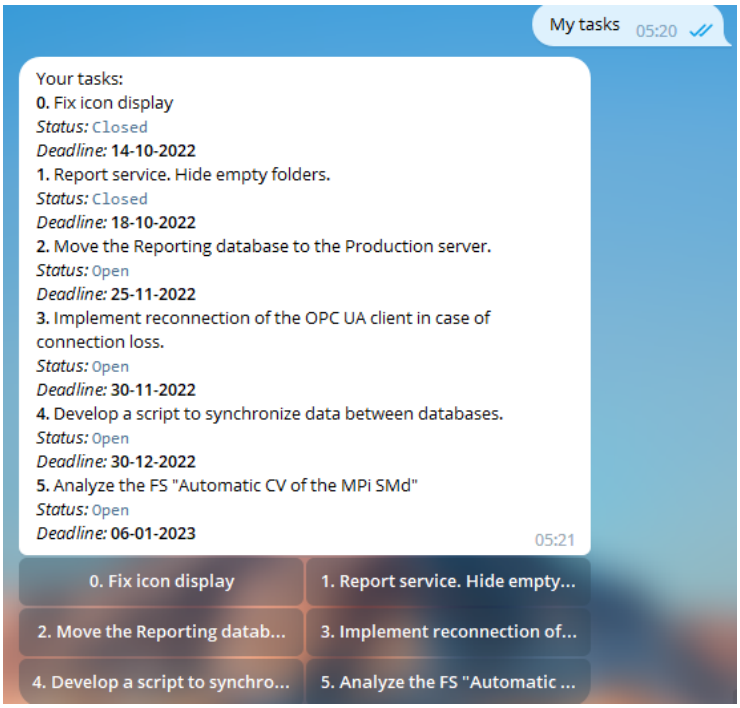


Рисунок 3.7.7 – Список завдань користувача

Для реалізації оповіщень про наближення терміну виконання задачі можна скористатися планувальником команд Laravel. У планувальнику команд необхідно створити задачу. Розклад задач додається у методі schedule, що знаходиться у файлі «app/Console/Kernel.php». Для початку необхідно створити користувацьку команду яка виконуватиме Cron задачі планувальника. Для цього необхідно виконати наступну команду:

```
php artisan make:command RemindUsersOfPendingTasksCron --command= remind_pending_tasks:cron.
```

Після виконання даної команди буде створено каталог Commands за шляхом «app\Console». В даному каталозі буде створено файл з вказаною назвою у команді (RemindUsersOfPendingTasksCron.php). Даний файл містить поле signature, що зберігає назву консольної команди та поле description з її описом. Також у даному класі є конструктор класу та метод handle, що виконується при виклику даної команди. Після написання логіки у методі handle достатньо додати у файлі ««app/Console/Kernel.php» створену команду до поля commands, та налаштувати її виклик кожного понеділку о 8 годині ранку. Для цього у методі schedule необхідно додати наступний рядок:

```
$schedule->command('remind_pending_tasks:cron')->weeklyOn(1, '8:00')->runInBackground();
```

Також додатково було застосовано метод runInBackground для виконання задачі у фоновому режимі. Це необхідно для подальшої сумісності з іншими командами, оскільки при наявності декількох задач, за замовчуванням, вони виконуються послідовно. Так, у випадку якщо одна із задач виявиться складною і на її виконання необхідно буде витратити значний час, то усі наступні задачі виконаються із великою затримкою. Отже, далі розглянемо застосування додатку на прикладі відділу розробки інформаційно-керуючих систем.

3.8 Робота з додатком

Початок роботи з розробленим веб-додатком є вхід адміністратора до системи управління задачами. Його головними задачами є:

- Додати необхідні відділи до системи;
- Додати усіх користувачів до системи;
- Сповістити користувачам дані для входу у систему;
- Надіслати посилання робітникам для приєднання до telegram-боту.

Для додавання нового відділу потрібно перейти до вкладки Divisions, у полі вводу ввести назву відділу та натиснути кнопку «Add new». На рисунку 3.8.1 зображено форму додавання відділу.

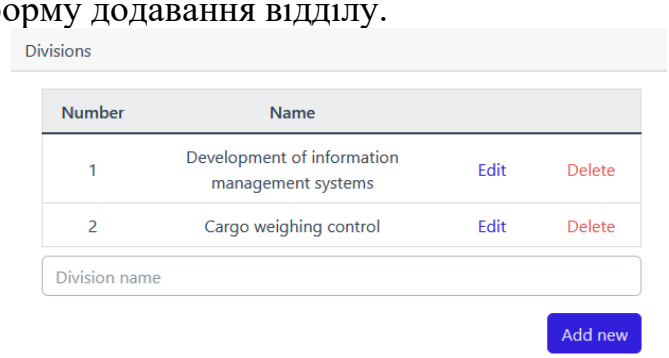


Рисунок 3.8.1 – Форма додавання відділу

Для додавання користувача необхідно перейти на вкладку Users та натиснути кнопку «Add new». На рисунку 3.8.2 зображено вкладку Users для адміністратора.

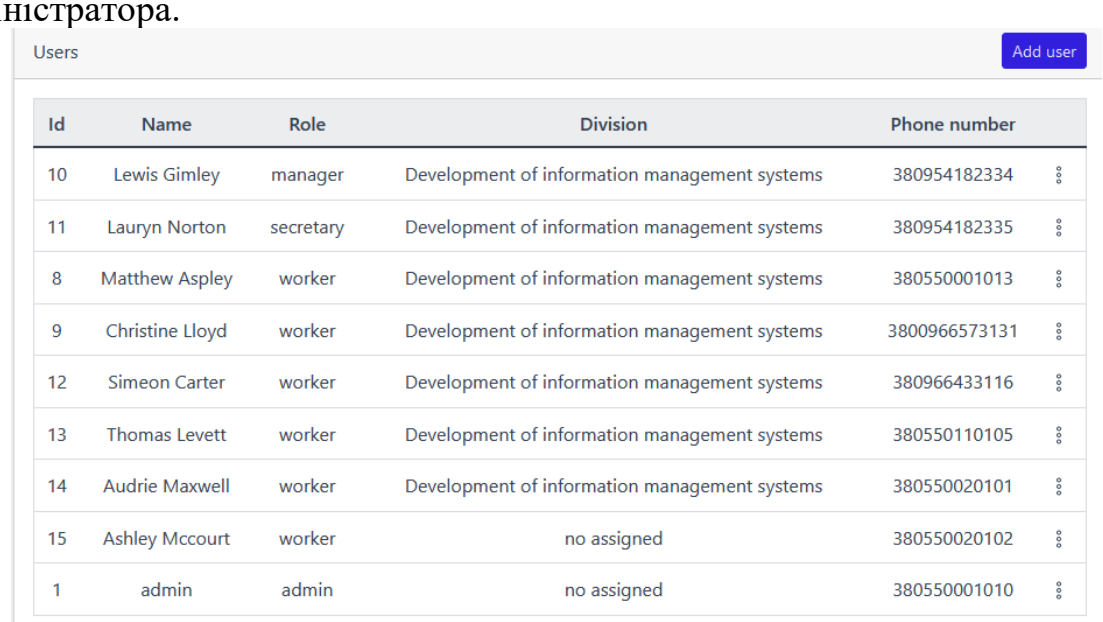


Рисунок 3.8.2 – Вкладка Users для адміністратора

Для додавання нового користувача необхідно заповнити усі поля, а саме: Full name (повне ім’я), Phone number (мобільний телефон користувача), Email (електронну пошту), обрати роль, обрати належність до відділу та ввести початковий пароль для облікового запису. На рисунку 3.8.3 зображено форму додавання нового користувача.

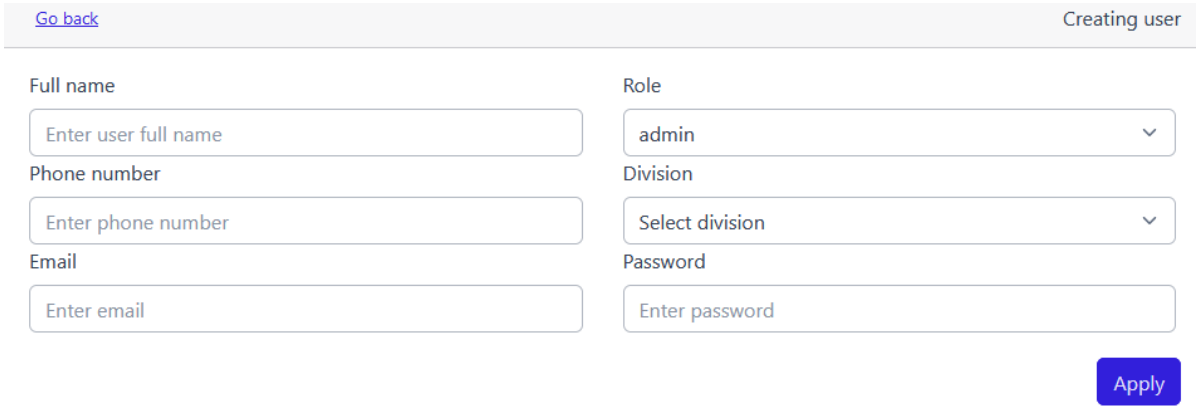


Рисунок 3.8.3 – Форма додавання нового користувача

Також адміністратор має повний доступ до управління задачами. Дана можливість додана для можливості підтримки користувачів.

Менеджери та секретарі. Дані ролі мають можливості переглядати список користувачів за своїм відділом, мають доступ до створення, редагування та видалення задач. Для створення задачі необхідно перейти до вкладки Tasks та натиснути кнопку «Add task». На рисунку 3.8.4 зображено список задач, що відображаються менеджеру.

Id	Title	Executor	Assigned By	Supervisor	Status	Created At	Deadline
12	Create a traffic matrix for access to the CHPP-2 PC.	Audrie Maxwell	Lewis Gimley	Lewis Gimley	Open	2022-11-22	2023-01-31
14	Organization of data collection from field devices for the AmFuelGas system.	Thomas Levett	Lewis Gimley	Lewis Gimley	Open	2022-11-22	2023-01-13
17	ARMP. Implementation of the downtime accounting system for Converter №3.	Christine Lloyd	Lewis Gimley	Lewis Gimley	Open	2022-11-22	2023-01-13

Рисунок 3.8.4 – Вигляд списку задач для перегляду менеджером

Для створення нової задачі необхідно заповнити поле Title (заголовок задачі), поле Description (опис задачі), обрати виконавця, обрати контролера та задати кінцеву дату виконання задачі. На рисунку 3.8.5 зображено форму створення задачі.

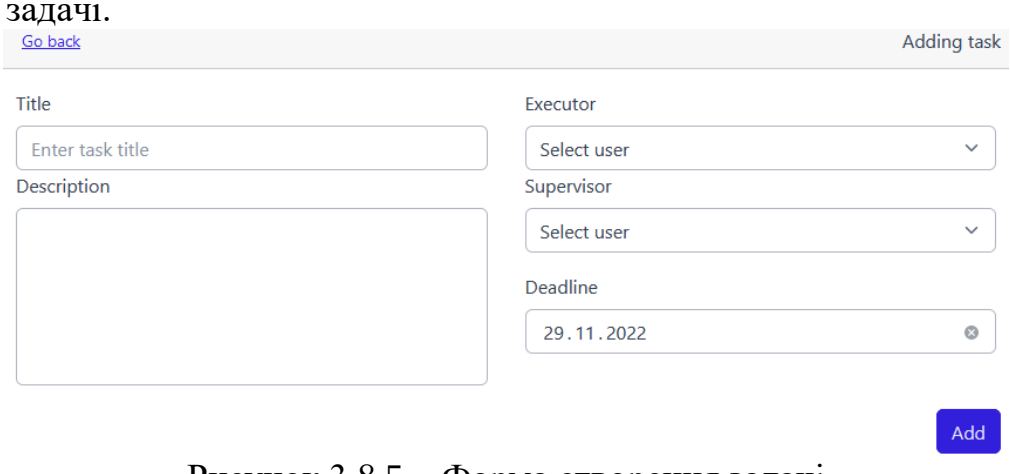


Рисунок 3.8.5 – Форма створення задачі

Для редагування вже існуючої задачі необхідно перейти на вкладку Tasks та натиснути на кнопку із зображенням вертикальних 3 крапок та обрати пункт Edit. На рисунку 3.8.6 зображено форму редагування задачі.

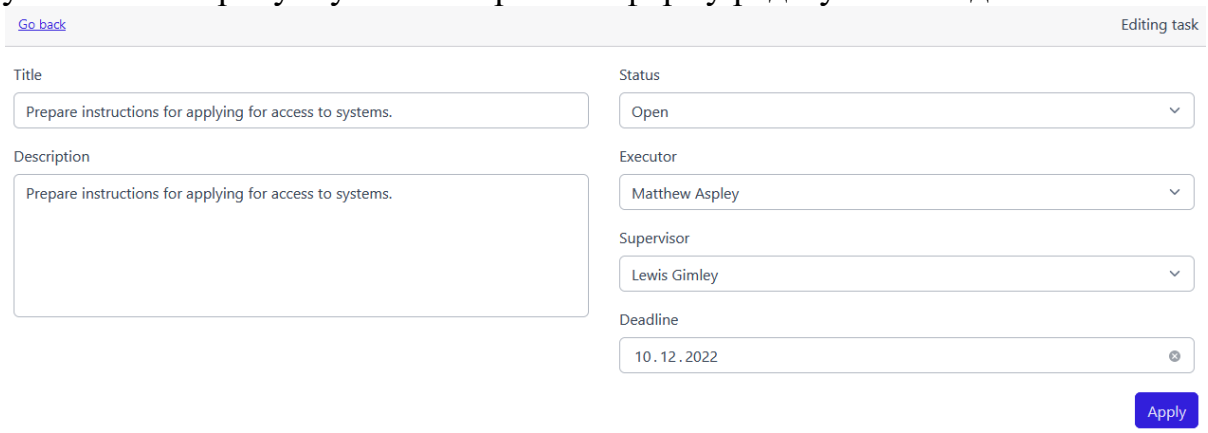


Рисунок 3.8.6 – Форма редагування задачі

Також менеджерам та секретарям доступні для завантаження звіти. Для завантаження звіту необхідно натиснути на стрілку у верхньому правому кутку, після чого відкриється вікно вибору звіту. На рисунку 3.8.7 зображено меню завантаження звіту.

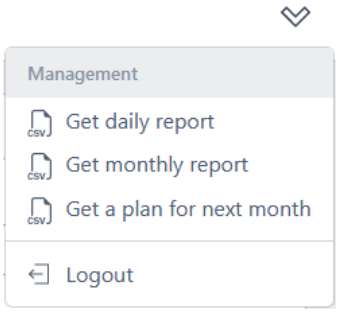


Рисунок 3.8.7 – Меню завантаження звіту

Звіти зберігаються у форматі CSV, тож їх можна переглянути за допомогою програми Microsoft Excel. На рисунку 3.8.8 зображено щоденний звіт.

	A	B	C	D	E	F	G
1	Number	Executor	Title	Description	Supervisor	Assigned By	Due Date
2	1	Audrie Maxwell	Implement reading data usi	Implement reading dat	Lewis Gimley	Lewis Gimley	23.11.2022
3	2	Christine Lloyd	ARMP. Implementation of t	ARMP. Implementation	Lewis Gimley	Lewis Gimley	23.11.2022
4	3	Simeon Carter	Move the Reporting databa	Move the Reporting da	Lewis Gimley	Lewis Gimley	25.11.2022
5	4	Matthew Aspley	Transferring ACS to a produ	Transferring ACS to a pr	Lewis Gimley	admin	16.12.2022

Рисунок 3.8.8 – Щоденний звіт про поточні роботи

Варто зазначити, що файли мають зручне найменування, так щоденний звіт має назву «tasks-on-16-11-2022.csv», що дає чітке розуміння про зміст звіту та за яку дату він зроблений. Щоденний звіт містить поля: ім’я виконавця задачі, назва задачі, опис задачі, ім’я людини яка контролює виконання завдання, ім’я людини яка призначила задачу та кінцеву дату виконання завдання.

Окрім щоденного звіту також можна переглянути звіт виконаних задач за минулий місяць. Такі звіти носять назви типу «closed-tasks-10-2022», де число 10 означає номер місяцю за який буде враховано закриті задачі. Даний тип звіту містить такі поля, як:

– Executor – містить ім’я людини, що має виконати завдання;
– Title – містить назву задачі;
– Description – містить опис задачі, додаткові коментарі;
– Supervisor – містить ім’я людини, що відповідає за перевірку виконання завдання та його прийом;
– Assigned By – містить ім’я користувача, що створив задачу;
– Due Date – містить дату кінцевого терміну виконання задачі;
– Closed At – містить дату коли задача була вирішена.

На рисунку 3.8.9 зображено звіт про завершені задачі минулого місяця.

	A	B	C	D	E	F	G	H
1	Number	Executor	Title	Description	Supervisor	Assigned B	Due Date	Closed At
2	1	Simeon Carter	Fix icon displ	Hieroglyphs	Lewis Gimley	admin	14.10.2022	13.10.2022
3	2	Simeon Carter	Report servic	Hide folders	Lewis Gimley	Lewis Gimle	18.10.2022	16.10.2022
4	3	Matthew Aspley	Create an Aut	Create functi	Lewis Gimley	admin	19.10.2022	18.10.2022

Рисунок 3.8.9 – Звіт про завершені задачі минулого місяця

Також є можливість переглянути план робіт на наступний місяць. Даний тип звіту матиме назву «tasks-on-month-11-2022», де число 11 означає номер місяця. Даний тип звіту як і місячний містить поля Executor, Title, Supervisor, Assigned By, Due Date, а також поле Status (містить поточний статус задачі). На рисунку 3.8.10 зображено план робіт на наступний місяць.

	A	B	C	D	E	F	G	H
1	Number	Executor	Title	Descriptic	Superviso	Assigned I	Status	Due Date
2	1	Matthew	Prepare ir	Prepare ir	Lewis Gim	Lewis Gim	Open	10.12.2022
3	2	Matthew	Prepare ir	Prepare ir	Lewis Gim	Lewis Gim	Open	10.12.2022
4	3	Christine	ARMP. Im	ARMP. Im	Lewis Gim	Lewis Gim	Open	13.12.2022
5	4	Thomas Le	ARMP mo	zabbix: AF	Lewis Gim	Lewis Gim	Open	13.12.2022
6	5	Matthew	Transferri	Transferri	Lewis Gim	admin	In progres	16.12.2022
7	6	Simeon Ca	Develop a	You need	Lewis Gim	admin	Open	30.12.2022

Рисунок 3.8.10 – План робіт на наступний місяць

Отже в даному підрозділі було розглянуто роботу з додатком, далі буде виконано тестування додатку.

3.9 Тестування

Розглянемо приклад застосування системи управління проектами на прикладі компанії з розробки програмного забезпечення (ПЗ). На рисунку 3.9.1 наведено організаційну структуру компанії.

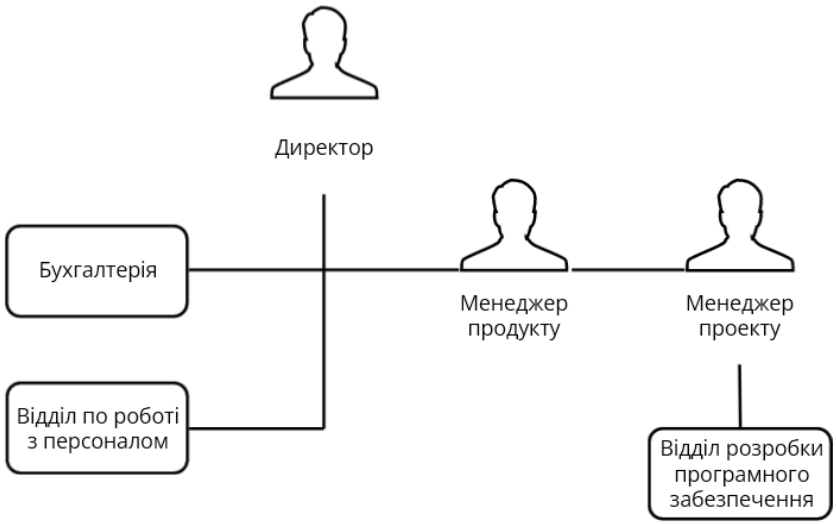


Рисунок 3.9.1 – Організаційна структура компанії

Отже, розглянемо більш детально структуру компанії, вона складається з:

– Директора;
– Двох співробітників бухгалтерії;
– Двох робітників по роботі з персоналом;
– Менеджера продукту;
– Менеджера проекту;
– Команди розробників, що налічує 15 чоловік.

На початку нового року директором було поставлено ряд задач. Першою задачею було додати новий функціонал до системи планування ресурсів (Enterprise Resource Planning, ERP). Ціллю даного проекту було покращити контролю та планування управлінням виконанням замовлень. Задача передбачала:

– Завершення впровадження одного з модулів ERP, що вже працює декілька років;

– Впровадження нового модуля;

– Забезпечення обміну інформацією з системою автоматизації технологічних процесів (АСУТП).

Другою задачею було впровадити систему управління взаємовідносинами з клієнтами (Customer Relationship Management, CRM). Ціллю впровадження CRM було підвищення кількості продажів, орієнтовно на 20-30%.

Директор розробив наступний план виконання проектів, що зображено на рисунку нижче.

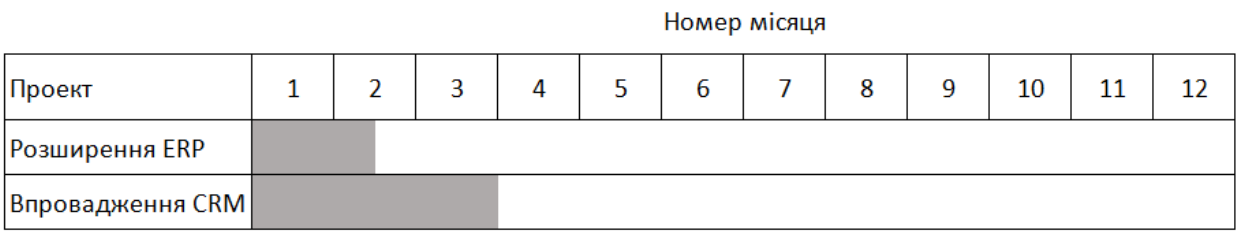


Рисунок 3.9.2 – План розробки проектів побудований директором

Тобто, згідно з розробленим планом робота над двома проектами повинна виконуватись одночасно. Це було зумовлено прагненням директора збільшити прибутки компанії задіявши усі ресурси компанії. Таким чином, відповідно до графіку робота над розширенням ERP мала тривати 1 місяць та 2 неділі, а для впровадження CRM було відведено 3 місяці.

Також було проведено розрахунки бюджету проектів і суми необхідних витрат. Даний бюджет розраховувався на команду розробників з 10-ти людей, оскільки саме стільки працівників планувалося задіяти на цих проектах. Дані суми витрат та бюджети зведено до таблиці, що наведена нижче.

Таблиця 3.9.1 – Статті витрат та бюджет проектів згідно плану виконання проектів без використання СКВЗ

Стаття витрат	Витрати за 1 місяць, грн.	Загальна кількість місяців	Загальна сума, грн.
На з/п директора	18.000	3	54.000
На з/п бухгалтерії	15.000		45.000
На з/п відділу по роботі з персоналом	14.000		42.000
На з/п менеджеру продукту	16.000		48.000
На з/п менеджера проекту	14.000		42.000
На з/п команди розробників (10 чоловік)	50.000		150.000
Витрати на технічні засоби	60.000		60.000
Витрати на оренду робочого приміщення	5.000		15.000
Сума:	456.000		
Загальний бюджет на проекти, грн.			
Розширення ERP	100.000		
Впровадження CRM	600.000		
Сума:	700.000		

Відповідно до проведених розрахунків прибуток компанії за 3 місяці має скласти 244 000 грн.

Але на практиці, на жаль, встановлені терміни не були витримані.

Основні причини недотримання термінів. По-перше, реальний початок робіт над проектами відбувся на 7 днів пізніше, через новорічні свята;

По-друге, не було враховано попередження менеджеру проекту про нестачу ресурсів для одночасної роботи над двома великими проектами. На практиці над проектом замість 10 людей працювало 5, через завантаженість інших п’яти.

По-третє, технічні ресурси компанії не дозволили розмістити покращену та розширену ERP. Потужність серверів, на яких розгортається програмне забезпечення було вичерпано, через що у користувачів система переставала відповідати. Через це довелося призупинити розробку ERP. Також аналогічна проблема виникла під час впровадження CRM.

Отже, у зв'язку з вичерпанням технічних та людських ресурсів час виконання проектів значно збільшився. Також додалися нові роботи з розробки власного центру обчислення даних (ЦОД) для розширення технічних ресурсів. Для розвантаження персоналу розробка ERP та CRM перейшли на утримання. Також менеджер продукту прийняв рішення розробити каталог IT сервісів та вимог до цих сервісів (Service Level Agreement, SLA). Розробка SLA потрібна для того, щоб узгодити вимоги до підтримки CRM та ERP, а також до технічних засобів.

Директор врахував нові обставини та склав оновив план робіт. На рисунку 3.9.3 зображено оновлений план виконання проектів.

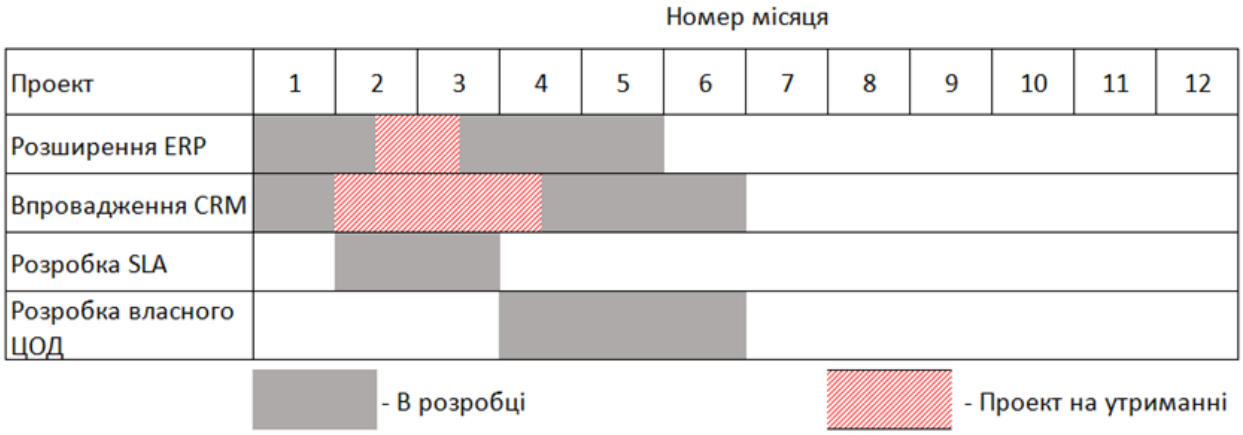


Рисунок 3.9.3 – Оновлений план виконання проектів

Під час виконання робіт за новим планом виникли нові фактори, що спричинили затримку розробки.

По-перше, замовники забажали мати доступ до інформації з мобільних телефонів, що потребує додаткового часу на розробку та розширення технічних засобів. Через складність розробки власного ЦОД та змін у технічному завданні було прийнято рішення використання орендованого ЦОД.

По-друге, директором було прийнято рішення впровадження IT стратегії. Дана стратегія потрібна для уточнення вимог до ERP та CRM, визначення пріоритетів автоматизації усіх бізнес-процесів компанії. Також в даній стратегії передбачалось допрацювати план проектів.

Після внесення чергових змін директором знову був розроблений план виконання проектів. На рисунку 3.9.4 зображено остаточний план виконання проектів.

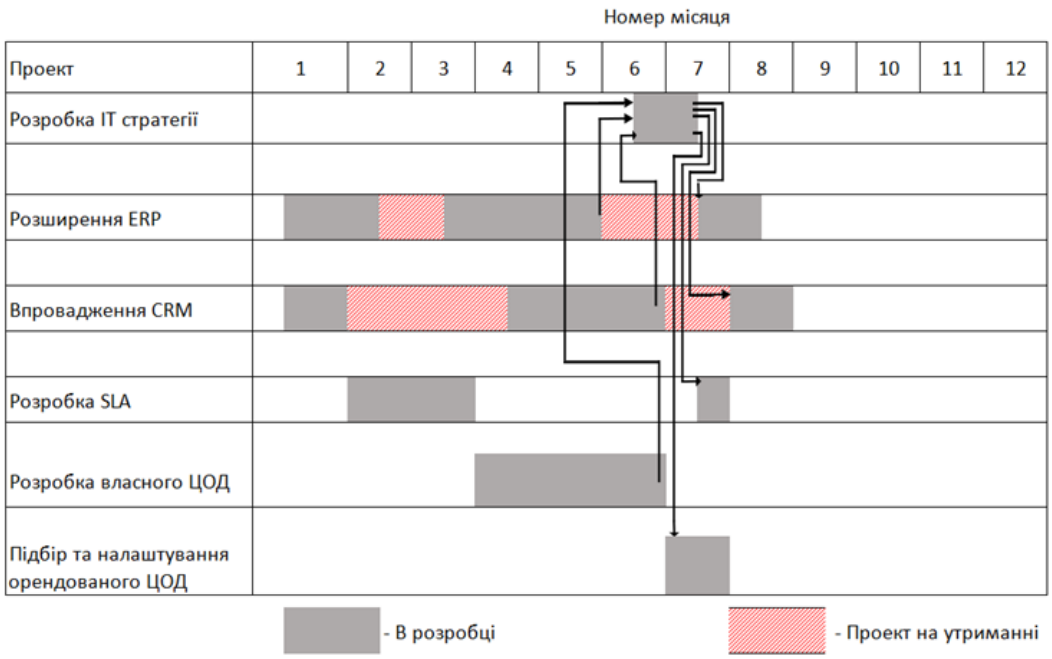


Рисунок 3.9.4 - Остаточний план виконання проектів

На плані стрілками додатково позначено зв'язок між задачами. Таким чином після невдалих спроб планування розробки було зроблено висновки та під час розробки IT стратегії враховано минулі помилки. Паралельно з розробкою стратегії відбувався підбір ЦОД для оренди.

Отже, у зв'язку з неправильним планування виконання проектів реальна кінцева дата розробки суттєво змістилася, замість запланованих 3 місяців роботи було витрачено 8. Причиною помилок при розрахунку реальних термінів виконання завдань є:

- Відсутність заздалегідь розробленої IT стратегії;
- Погана комунікація між менеджером проекту, менеджером продукту та замовниками;
- Відсутність у директора розуміння про реальну кількість вільних ресурсів;
- Неправильне визначення пріоритетів під час розробки.

Окрім зазначених вище проблем на затримки процесу реалізації проекту певним чином впливало проведення додаткових нарад через постійні переноси термінів та появі нових обставин. Загальний час на проведення даних нарад за увесь час розробки склав 12 днів.

Кінцеві бюджети по витратам компанії та замовника були підготовлені бухгалтерією на зведено до таблиці, що наведена нижче.

Таблиця 3.9.2 – Кінцеві бюджети компанії та замовника

Стаття витрат	Витрати за 1 місяць, грн.	Загальна кількість місяців	Загальна сума, грн.
На з/п директора	18.000	8	144.000
На з/п бухгалтерії	15.000		120.000

Продовження таблиці 3.9.2

На з/п відділу по роботі з персоналом	14.000	8	112.000
На з/п менеджеру продукту	16.000		128.000
На з/п менеджера проекту	14.000		112.000
На з/п команди розробників (5 чоловік)	50.000		400.000
Витрати на оренду та налаштування ЦОД	15.000		15.000
Витрати на оренду робочого приміщення	5.000		40.000
Сума:	1.071.000		
Витрати замовника на проекти, грн.			
Розширення ERP	100.000		
Додаткові витрати на власний ЦОД	30.000		
Впровадження CRM	600.000		
Оренда ЦОД	20.000		
Додаткові витрати на розширення ERP, у зв'язку з новими вимогами	21.000		
Додаткові витрати на впровадження CRM, у зв'язку з новими вимогами	50.000		
Сума:	821.000		

З нових розрахунків видно, що замовнику довелося збільшити витрати на виконання проектів, що негативно впливає на репутацію компанії. Також компанія не отримала прибутку, а тільки навпаки, зазнала збитків на суму 250 000 грн.

Аби вирішити усі перелічені проблеми застосовують системи управління проектами. Тому, далі буде розглянуто приклад планування тих самих робіт але із застосуванням СКВЗ.

Для правильного складання плану необхідно діяти в наступній послідовності:

– Директор та менеджер продукту мають визначити цілі проекту разом із замовником, визначити модель розробки проекту та пріоритети проектів;

– Менеджер продукту має визначити головні особливості проекту, та сумісно з менеджером проекту підготувати приблизний графік робіт з урахуванням наявних ресурсів та врахувати витрати на додаткове програмне і/або технічне забезпечення. Також менеджер продукту має погодити з менеджером проекту можливість виконання проектів згідно з вказаних пріоритетів;

– Менеджер проекту має скласти технічне завдання та направити до директора. Також менеджер проекту може погоджувати технічне завдання з розробниками ПО, менеджером проекту та директором;

– Бухгалтерія має провести розрахунки вартості робіт та потрібний бюджет проекту та передати директору;

– Директор з планом робіт та розрахованим бюджетом погоджує із замовником початок розробки. У випадку наявності зауважень з боку менеджеру продукту або менеджеру проекту щодо пріоритетів або додаткових витрат - ці питання також обговорюються із замовником. Також погоджується технічне завдання;

– При наявності зауважень щодо технічного завдання менеджер проекту має бути проінформований та надати новий план робіт при необхідності;

Отже, враховуючи Таким чином було складено план підготовчого етапу. Даний план наведено у таблиці нижче.

Таблиця 3.9.3 – План підготови робіт

Задачі	Відповідальна особа	Дата початку	Кінцева дата	Дні
Провести зустріч із замовником для обговорення цілей проекту	Директор та менеджер продукту	1/10	1/11	2
Складання приблизного плану робіт	Менеджер проекту та менеджер продукту	1/12	1/13	2
Складання технічного завдання	Менеджер проекту	1/13	1/15	3
Розрахунок приблизного бюджету	Бухгалтерія	1/14	1/14	1
Проведення повторної зустрічі із замовником для погодження початку розробки	Директор та менеджер продукту	1/17	1/17	1
Ознайомлення з технічним завданням та складанням детального плану розробки проектів	Менеджер проекту	1/19	1/19	1

Отже, підготовчі роботи будуть виконані з 10 січня по 19 січня. Для підготовки повного плану виконання проектів необхідно мати повне уявлення про вільні ресурси компанії для об’єктивної оцінки витрат часу. Розглянемо приклад розробки плану з використанням СКВЗ.

Менеджери мають змогу переглядати кількість назначених задач на кожного з робітників, що дозволяє оцінити вільні ресурси. За допомогою даного функціоналу директором було встановлено, що над проектом може працювати 5 розробників. Таким чином відповідно до погоджених робіт із замовником та кількості вільних ресурсів директор розробив план робіт. На рисунку 3.9.5 зображено даний план.

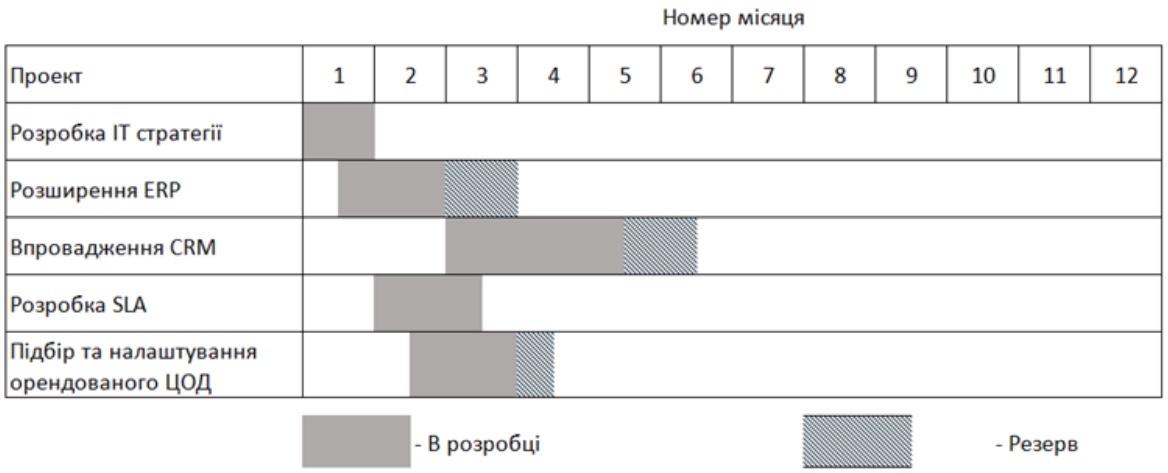


Рисунок 3.9.5 – План виконання робіт з використанням СУП

Відповідно до плану робіт та технічного завдання було складено та погоджено бюджети компанії та замовника. Дані бюджет наведено в таблиці 3.9.4.

Таблиця 3.9.4 – Бюджети компанії та замовника з використанням СКВЗ

Стаття витрат	Витрати за 1 місяць, грн.	Загальна кількість місяців	Загальна сума, грн.
На з/п директора	18.000	5,5	99.000
На з/п бухгалтерії	15.000		82.500
На з/п відділу по роботі з персоналом	14.000		77.000
На з/п менеджеру продукту	16.000		88.000
На з/п менеджера проекту	14.000		77.000
На з/п команди розробників (5 чоловік)	50.000		275.000
Витрати на оренду та налаштування ЦОД	15.000		15.000
Витрати на оренду робочого приміщення	5.000		27.500
Сума:			741.000
Витрати замовника на проекти			
Розширення ERP	180.000		
Впровадження CRM	700.000		
Оренда ЦОД	20.000		
Сума:	900.000		

Відповідно до розрахованих бюджетів компанія має заробити 159 000 грн.

З плану виконання робіт видно, що кінцева дата виконання проектів з врахуванням резерву є середина шостого місяця, тобто середина червня. Порівняно з планом робіт без використання СКВЗ кінцева дата є значно пізнішою. Суттєве зміщення кінцевої дати було обумовлене резервуванням запасного часу для кожної задач на випадок виникнення непередбачених проблем або складнощів з вирішення певних задач. Розрахунок резервного часу базується на основі досвіду з попередніх проектів та статистики.

Однак, на практиці вдалося зекономити три місяці оскільки основні ризики були значно зменшені. Таким чином реальні витраті компанії виявилися меншими ніж розраховували. Остаточні суми витрат наведено у таблиці 3.9.5.

Таблиця 3.9.5 – Остаточні витрати компанії при плануванні виконання проектів з використанням СКВЗ

Стаття витрат	Витрати за 1 місяць, грн.	Загальна кількість місяців	Загальна сума, грн.
На з/п директора	18.000	5	90.000
На з/п бухгалтерії	15.000		75.000
На з/п відділу по роботі з персоналом	14.000		70.000
На з/п менеджеру продукту	16.000		80.000
На з/п менеджера проекту	14.000		70.000
На з/п команди розробників (5 чоловік)	50.000		250.000
Витрати на оренду та налаштування ЦОД	15.000		15.000
Витрати на оренду робочого приміщення	5.000		25.000
Сума:			675.000

Таким чином загальний прибуток компанії склав 225 000 грн. .Отже, система управління проектами при складанні плану відіграла значну роль. Оскільки за допомогою отриманої статистики використання ресурсів було визначено більш реальні терміни виконання завдань. Також на основі статистики виконання проектів було передбачено резервний час. Використання СКВЗ допомогло визначити пріоритети задач, що посприяло зменшенню ризиків. Окрім цього покращилась комунікація і було вчасно визначено, що наявних технічних засобів не вистачає для розгортання власного ЦОД та було прийнято рішення про оренду. І в результаті використання СКВЗ компанія вклалася у визначені терміни, отримала більший прибуток ніж очікувала.

Отже, складемо порівняльну таблицю роботи компанії з використанням СКВЗ та без. Варто зазначити, що у випадку невикористання СКВЗ перехідні

процеси також впливали на терміни виконання проектів. Під перехідними процесами мається на увазі проведення нарад для погодження нових бюджетів, підходів до розробки, вибору певних технічних рішень. У таблиці 3.9.6 наведено порівняння процесів виконання проектів з використанням СКВЗ та без.

Таблиця 3.9.6 – Порівняння процесів виконання проектів з використанням СКВЗ та без

Варіант	Планування без використання СУП	Планування з використанням СУП
Кіл-ть місяців на виконання проектів, план/факт	3/8	5,5/5
Кіл-ть днів на планування проекту	6	3
Загальний час проведення нарад, кіл-ть днів	10	5
Запланований бюджет на витрати компанії, план/факт грн.	456 000/1 071 000	741 000/675 000
Запланований бюджет замовника, план/факт грн.	700 000/821 000	900 000

Відповідно до таблиці розрахуємо продуктивність компанії при планування без використання СКВЗ за формулою:

$$\text{Пр} = \frac{\text{кіл-ть проектів}}{\text{кіл-ть витрачених місяців}} * 100\% \quad (3.9.1)$$

$$\text{Пр} = 2 / 8 * 100\% = 25\%$$

Тобто компанія без використання СКВЗ за місяць виконує 25% робіт по одному проекту. Далі розрахуємо продуктивність компанії при плануванні з використанням СКВЗ:

$$\text{Пр} = 2 / 5 * 100\% = 40\%$$

Отже, використання СКВЗ дозволила збільшити продуктивність компанії на 15%. Далі буде розраховано ефективність компанії без використання СКВЗ:

$$\eta = \text{оп. вик. пр.} - \text{вит. реал. пр.} \quad (3.9.2)$$

$$\eta = 821\,000 - 1\,071\,000 = -250\,000 \text{ грн.},$$

Де оп.вик.пр - оплата виконаних проектів; вит.реал.пр. - витрати на реалізацію проектів.

Отже, ефективність компанії без використання СКВЗ склала -250 000 грн., що однозначно є негативним результатом. Далі розрахуємо ефективність компанії при використанні СКВЗ:

$$\eta = 900\,000 - 645\,000 = 225\,000 \text{ грн.},$$

Таким чином ефективність компанії при використанні СКВЗ є набагато кращою ніж без її застосування.

Висновки

Отже, в даному розділі було розглянуто діяльність ПАТ «АрселорМіттал Кривий Ріг», показники ефективності персоналу даного підприємства за роками відповідно до яких було зроблено висновки про те, що у компанії добре організована праця. Також було розглянуто організацій структуру ПАТ «АрселорМіттал Кривий Ріг» та проаналізовано його основні показники діяльності. Відповідно до зробленого аналізу було встановлено, що чистий дохід від реалізації товарів, робіт і послуг збільшився на 1,74% порівняно з 2019 роком, собівартість реалізованої продукції у 2020 році зменшилась на 7,3%. Чистий фінансовий результат (збиток) після сплати податку на прибуток 2019 році склав -2265232 тис.грн., в 2020 році цей показник зріс на 2,7%. Таким чином, на основі показників діяльності підприємства було зроблено висновок про необхідність достовірної оцінки бізнес-процесів для покращення фінансового стану підприємства. Для покращення оцінювання бізнес-процесів можна застосовувати СКВЗ.

Далі було розглянуто що таке веб-сервер та процес встановлення і налаштування веб-серверу для розробки системи контролю виконання завдань. Також було розглянуто встановлення, налаштування та роботу із середовищем розробки PhpStrom. Окрім цього було розглянуто роботу з системою контролю версій Git, використання репозиторіїв для сумісної роботи над проектом та резервного зберігання копій версій проекту.

Далі було розглянуто процес розробки додатку з використанням фреймворку Laravel, бібліотек компонентів графічного дизайну CoreUI та Bootstrap, реалізовано інтеграцію з Telegram API. Також було описано роботу з додатком та проведено тестування. В ході тестування додатку було отримано результати, що свідчать про отримані переваги від використання СКВЗ, а саме: підвищення продуктивності компанії на 15% та підвищення ефективності компанії.

ВИСНОВКИ

В ході написання випускної роботи було встановлено, що ефективність функціонування будь-якого підприємства залежить від диверсифікації виконуваних функцій, проектів – що пов’язано безпосередньо з розв’язанням великої кількості різноманітних задач, погодження документації, виконання доручень керівників та інше. Оскільки одночасно компанія, зазвичай, працює над множиною проектів, то для планування та контролю виконання усіх перерахованих видів робіт може бути суттєво спрощено за рахунок впровадження системи контролю виконання завдань.

Далі було розглянуто місце системи контролю виконання завдань у бізнес-процесах підприємств. В ході даного аналізу було встановлено, що використання СКВЗ покращує перебіг управлінських бізнес-процесів які у свою чергу впливають на поточну діяльність підприємства та його конкурентоспроможність. Таким чином розглядаючи основні і допоміжні бізнес-процеси в контексті функціонування промислових підприємств було зроблено висновок про інноваційність даного підходу.

Далі було проаналізовано діяльність підприємства ПАТ «АрселорМіттал Кривий Ріг» та визначено, що у 2019 році підприємство зазнало збитків на - 2265232 грн.. Таким чином було зроблено висновки про необхідність достовірної оцінки бізнес-процесів що на ньому відбуваються в режимі реального часу з використання IT технологій для покращення фінансового стану підприємства.

Наступним етапом роботи була розробка системи контролю виконання завдань. Мною було створено веб-додаток на основі фреймворку Laravel, у якості СУБД використовувалась MySQL. Розробка проводилась із дотриманням патерну MVC, для спрощення подальшої підтримки та розширення функціоналу. Також було використано Telegram API для створення боту, що сповіщає про наближення термінів виконання задач, надає можливість доступу до перегляду та керування своїми завданнями.

Далі було проведено тестування додатку на прикладі компанії з розробки програмного забезпечення. В ході тестування додатку було отримано результати, що свідчать про отримані переваги від використання СКВЗ, а саме: підвищення продуктивності компанії на 15% та підвищення ефективності компанії.

Відносно розробленої системи планується подальша розробка та впровадження нових функцій, а саме: доступ менеджерів та секретарів для створення та управління задачами через telegram-бота, додати можливість завантаження звітів менеджерами через telegram-бот, додати можливість фільтрування задач за різними критеріями.

Також було виконано апробацію досліджень на V конференції Всеукраїнській науково-практичній інтернет-конференції молодих вчених та студентів «Сучасні інформаційні системи та технології» від Херсонського національного технічного університету на базі кафедри інформаційних технологій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Варламов С. В. Система управления проектами организации: анализ подходов и существующих программных решений / Сергей Владимирович Варламов, Павел Валерьевич Скородумов // Вопросы территориального развития. – 2015. – № 5.
2) Системи управління проектами – що це? Огляд кращих програм [Електронний ресурс] // Роби Бізнес, Укр – 2019. – Режим доступу до ресурсу: https://робибізнес.укр/teoriya/sistemi-upravl-nnya-proektami/.
3) Lynn R. What is a Task Management Tool? [Електронний ресурс] / Rachaelle Lynn // Planview – Режим доступу до ресурсу: https://www.planview.com/resources/articles/lkdc-task-management-tool/.
4) Информационная система управления проектами [Електронний ресурс] // Обучение и трудоустройство проектных менеджеров. – Режим доступу: https://pm-way.com/materials/material/show/205
5) Pratt M. business process [Електронний ресурс] / M. Pratt, M. Roy, E. McLaughlin // TechTarget. – 2022. – Режим доступу до ресурсу: https://www.techtarget.com/searchcio/definition/business-process.
6) Інформаційні ресурси ендогенно спрямованого зростання економічних систем / В.В. Микитенко, М.О. Чупріна // Вісник економічної науки України. — 2009. — № 2 (16). — С. 91-95. — Бібліогр.: 19 назв. — укр.
7) Перерва П. Г. Маркетинг і цифрові технології / П. Г. Перерва, С. М. Назаренко. // Державний університет «Одеська політехніка». – 2021. – Т. 5 – №3. – С. 18–29.
8) Бухгалтерський облік в умовах застосування інформаційних технологій [Текст] : монографія / В. О. Осмятченко ; ДВНЗ "Київ. нац. екон. ун-т ім. В. Гетьмана". - К. : КНЕУ, 2010. - 263 с. : рис., табл. - Бібліогр.: с. 250-261. - 300 прим.
9) Purdenko O., Melnik V. Innovative business tools in trade. Zovnishnja torgivlja: ekonomika, finansy, pravo. 2021. № 3. S. 77-84. Serija. Ekonomichni nauky.
10) How to Choose a Project Management Software in 2022 [Електронний ресурс] // TrustRadius. – Режим доступу: https://www.trustradius.com/buyer-blog/how-to-choose-project-management-software.
11) Uzialko A. Choosing Project Management Software [Електронний ресурс] / Adam Uzialko // Business News Daily. – Режим доступу: https://www.businessnewsdaily.com/9976-project-management-software-buyers-guide.html

12) Chappell E. Jira Review 2022 (Features, Pros, Cons, Pricing) [Електронний ресурс] / Erica Chappell // ClickUp. – 2022. – Режим доступу до ресурсу: https://clickup.com/blog/jira-review/.
13) Непрерывная поставка начинается с доски Kanban в Jira [Електронний ресурс] // Atlassian. – Режим доступу: https://www.atlassian.com/
14) JIRA - Issue Types [Електронний ресурс] // Tutorialspoint – Режим доступу до ресурсу: https://www.tutorialspoint.com/jira/jira_issue_types.htm.
15) Знакомство с мобильным приложением Jira | Atlassian [Електронний ресурс] // Atlassian. – Режим доступу: https://www.atlassian.com/
16) Дорожные карты в Jira Software [Електронний ресурс] // Atlassian – Режим доступу до ресурсу: https://www.atlassian.com/ru/software/jira/guides/roadmaps/
17) Саенко Е. Что такое Trello и как им пользоваться [Електронний ресурс] / Екатерина Саенко // Laba. – 2020. – Режим доступу до ресурсу: https://l-a-b-a.com/blog/1710-chto-takoe-trello-i-kak-im-polzovatsya.
18) Learn Trello board basics [Електронний ресурс] // Trello – Режим доступу до ресурсу: https://trello.com/guide/trello-101.
19) Trello: что это такое и как им пользоваться [Електронний ресурс] // HOSTiQ.ua. – 2022. – Режим доступу до ресурсу: https://hostiq.ua/blog/what-is-trello-2/.
20) Ravenscraft E. How to Use Trello [Електронний ресурс] / Eric Ravenscraft // Zapier. – 2019. – Режим доступу до ресурсу: https://zapier.com/blog/trello-tutorial/.
21) Morpus N. Microsoft Project Review [Електронний ресурс] / Nicholas Morpus // The Ascent. – 2022. – Режим доступу до ресурсу: https://www.fool.com/the-ascent/small-business/project-management/microsoft-project-review/.
22) Neumeyer A. Microsoft Project Example – Let’s create your first real project in a just few steps [Електронний ресурс] / Adrian Neumeyer // TacticalProjectManager.com – Режим доступу до ресурсу: https://www.tacticalprojectmanager.com/microsoft-project-example.
23) Jira и Trello Отличия 2022: Управление Задачами и Проектами в Trello и Jira - Merehead [Електронний ресурс] // Merehead. – Режим доступу: https://merehead.com/ru/blog/jira-and-trello-2020/
24) Порівняння Jira vs Trello: що вибрати для моєї команди [Електронний ресурс] // Український Спектр. – 2020. – Режим доступу до ресурсу: https://uaspectr.com/2020/10/14/jira-vs-trello-raund/.
25) Цены на Jira – Стоимость месячной и годовой подписки для одного пользователя [Електронний ресурс] // Atlassian. – Режим доступу: https://www.atlassian.com/ru/software/jira/pricing?tab=cloud

26) Швиданенко Г.О. Формування бізнес-моделі підприємства [Електронний ресурс] / Г. О. Швиданенко, Н. В. Ревуцька // ДВНЗ «Київський національний економічний університет імені Вадима Гетьмана». – 2013. – Режим доступу до ресурсу: https://ir.kneu.edu.ua/.
27) Організація бізнес-процесів [Електронний ресурс] // Школа розвитку SPE – Режим доступу до ресурсу: https://spe.org.ua/blog/bi/orhanizatsiia-biznes-protsesiv/.
28) Демиденко В. В. Управління бізнес-процесами як складова процесного підходу до управління підприємством [Електронний ресурс] / В. В. Демиденко // Ефективна економіка. – 2015. – Режим доступу до ресурсу: http://www.economy.nayka.com.ua/?op=1&z=4517.
29) Брітченко І. Г., Князевич А.О. Контролінг : навч. посіб. / І. Г. Брітченко, А. О. Князевич. – Рівне: Волинські обереги, 2015. – 280с.
30) Контроль виробничих процесів: поняття, зміст, види. [Електронний ресурс] // Освіта.ua. – 2011. – Режим доступу до ресурсу: https://osvita.ua/vnz/reports/management/14372/.
31) Основи менеджменту: Навчальний посібник для студентів напрямів підготовки 1004 «Транспортні технології» і 0502 «Менеджмент». Харків: ХНАМГ, 2006
32) Бужимська К. О. Формування та розвиток системи контролінгу на промислових підприємствах [Електронний ресурс] / К. О. Бужимська, І. М. Царук // Державний університет «Житомирська політехніка». – 2021. – Режим доступу до ресурсу: http://eztuir.ztu.edu.ua/handle/123456789/7878.
33) Всього зареєстровано [Електронний ресурс] // Державне підприємство «Український інститут інтелектуальної власності». – 2022. – Режим доступу до ресурсу: https://ukrpatent.org/uk/articles/vsjogo.
34) Ukraine ranks 45th among the 131 economies featured in the GII 2020 [Електронний ресурс] // THE GLOBAL INNOVATION INDEX. – 2020. – Режим доступу до ресурсу: https://www.wipo.int/edocs/pubdocs/en/wipo_pub_gii_2020/ua.pdf.
35) Федоренко О. Приклади KPI: як розрахувати ключові показники ефективності [Електронний ресурс] / Олександр Федоренко // Waytobi. – 2020. – Режим доступу до ресурсу: https://waytobi.com/ua/blog/kpi-examples-how-to-calculate-kpi.html.
36) Key Performance Indicators, KPI [Електронний ресурс] // IT.UA – Режим доступу до ресурсу: https://www.it.ua/knowledge-base/technology-innovation/key-performance-indicators-kpi.

37) ЗАКОН УКРАЇНИ Про затвердження Загальнодержавної програми розвитку мінерально-сировинної бази України на період до 2030 року [Електронний ресурс] // Верховна Рада України. – 2012. – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/3268-17#Text.
38) Чим сервер відрізняється від звичайного комп'ютера [Електронний ресурс] // moyaosvita – Режим доступу до ресурсу: https://moyaosvita.com.ua/internet/chim-server-vidriznyayetsya-vid-zvichajnogo-kompyutera/.
39) Чим відрізняється серверна оперативна пам'ять від звичайної? Ddr3 Server Memory [Електронний ресурс] // hi-news.pp.ua. – 2018. – Режим доступу до ресурсу: https://hi-news.pp.ua/kompyuteri/14011-chim-vdrznyayetsya-serverna-operativna-pamyat-vd-zvichaynoyi-ddr3-server-memory.html.
40) Что такое веб-сервер [Електронний ресурс] // Mozilla Foundation. – 2022. – Режим доступу до ресурсу: https://developer.mozilla.org/
41) Miskov A. Учебник по фронтенд-разработке [Електронний ресурс] / Miskov – Режим доступу до ресурсу: https://amiskov.github.io/frontend-handbook/.
42) Як працює DNS [Електронний ресурс] // HOSTiQ.ua.. – 2022. – Режим доступу до ресурсу: https://hostiq.ua/blog/ukr/how-does-dns-work/.
43) Веб-сервер [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://vyspiansky.gitbook.io/introduction-to-web-development/getting-started/
44) Usage statistics of web servers [Електронний ресурс] // Q-Success – Режим доступу до ресурсу: https://w3techs.com/technologies/overview/web_server.
45) November 2022 Web Server Survey [Електронний ресурс] // Netcraft Ltd. – 2022. – Режим доступу до ресурсу: https://news.netcraft.com/
46) Nginx [Електронний ресурс] // Itglobal.com – Режим доступу до ресурсу: https://itglobal.com/ru-ru/company/glossary/nginx/.
47) Настройка обратного прокси Nginx [Електронний ресурс] // Routerus. – 2020. – Режим доступу до ресурсу: https://routerus.com/nginx-reverse-proxy/.
48) Що таке NGINX і навіщо він потрібний [Електронний ресурс] // VPS.ua. – Режим доступу до ресурсу: https://vps.ua/wiki/ukr/nginx/.
49) Tunggal A. T. IIS vs Apache: Which is the Best Web Server? [Електронний ресурс] / Abi Tyas Tunggal // UpGuard. – 2022. – Режим доступу до ресурсу: https://www.upguard.com/blog/iis-apache.
50) Apache – що це? [Електронний ресурс] // FREEhost.com.ua – Режим доступу до ресурсу: https://freehost.com.ua/ukr/faq/wiki/apache-chto-eto/.
51) AMPPS: вражаюче середовище веб-розробки на GNU / Linux [Електронний ресурс]. – 2020. – Режим доступу до ресурсу: https://blog.desdelinux.net/uk/

52) Типы данных MySQL [Електронний ресурс] // metanit.com. – 2018. – Режим доступу до ресурсу: https://metanit.com/sql/mysql/2.3.php.
53) MySQL Data Types [Електронний ресурс] // W3Schools – Режим доступу до ресурсу: https://www.w3schools.com/mysql/mysql_datatypes.asp.
54) PhpStorm [Електронний ресурс]. – 2021. – Режим доступу до ресурсу: https://www.wiki.uk-ua.nina.az/PhpStorm.html.
55) Filatov A. How to be Efficient in PhpStorm: An IDE that Really Matters [Електронний ресурс] / Alexey Filatov – Режим доступу до ресурсу: https://www.toptal.com/php/how-to-be-efficient-in-phpstorm.
56) Возможности [Електронний ресурс] // JetBrains – Режим доступу до ресурсу: https://www.jetbrains.com/ru-ru/phpstorm/features/.
57) Create a remote server configuration [Електронний ресурс] // JetBrains – Режим доступу до ресурсу: https://www.jetbrains.com/help/phpstorm/creating-a-remote-server-configuration.html#config
58) Introduction to Git and Types of Version Control Systems [Електронний ресурс]. – 2020. – Режим доступу до ресурсу: https://serengetitech.com/tech/introduction-to-git-and-types-of-version-control-systems/.
59) Вступ - Про систему контролю версій [Електронний ресурс] – Режим доступу до ресурсу: https://git-scm.com/book/uk/v2/.
60) What is a centralized version control system [Електронний ресурс] // GitLab – Режим доступу до ресурсу: https://about.gitlab.com/topics/version-control/what-is-centralized-version-control-system/.
61) The benefits of a distributed version control system [Електронний ресурс] // Gitlab – Режим доступу до ресурсу: https://about.gitlab.com/topics/version-control/benefits-distributed-version-control-system/.
62) Вступ - Основи Git [Електронний ресурс] // Git – Режим доступу до ресурсу: https://git-scm.com/book/uk/v2/.
63) Рабочий процесс Gitflow Workflow [Електронний ресурс] // Atlassian – Режим доступу до ресурсу: https://www.atlassian.com/ru/git/tutorials/comparing-workflows/gitflow-workflow.
64) Driessen V. A successful Git branching model [Електронний ресурс] / Vincent Driessen. – 2010. – Режим доступу до ресурсу: https://nvie.com/posts/a-successful-git-branching-model/.
65) Структура приложения [Електронний ресурс] // Laravel Framework – Режим доступу до ресурсу: https://laravel.su/docs/5.2/structure.
66) Laravel. Полное руководство. 2-е изд. - СПб.: Питер, 2020. - 512 с.: ил. - (Серия «Бестселлеры O'Reilly»).

67) MVC: Model, View, Controller [Електронний ресурс] // Codecademy – Режим доступу до ресурсу: https://www.codecademy.com/article/mvc.
68) Bootstrap Admin Dashboard Template & UI Components Library [Електронний ресурс] // CoreUI – Режим доступу до ресурсу: https://coreui.io/.
69) Доросинский Л. Р. Розробка інтернет-додатків [Електронний ресурс] / Л. Р. Доросинский // stud.com.ua. – 2018. – Режим доступу до ресурсу: https://stud.com.ua/121236/informatika/rozrobka_internet-dodatkiv.
70) Безопасное хеширование паролей [Електронний ресурс] // PHP Group – Режим доступу до ресурсу: https://www.php.net/manual/ru/faq.passwords.php.

Додаток А
Приклад карточки в Trello

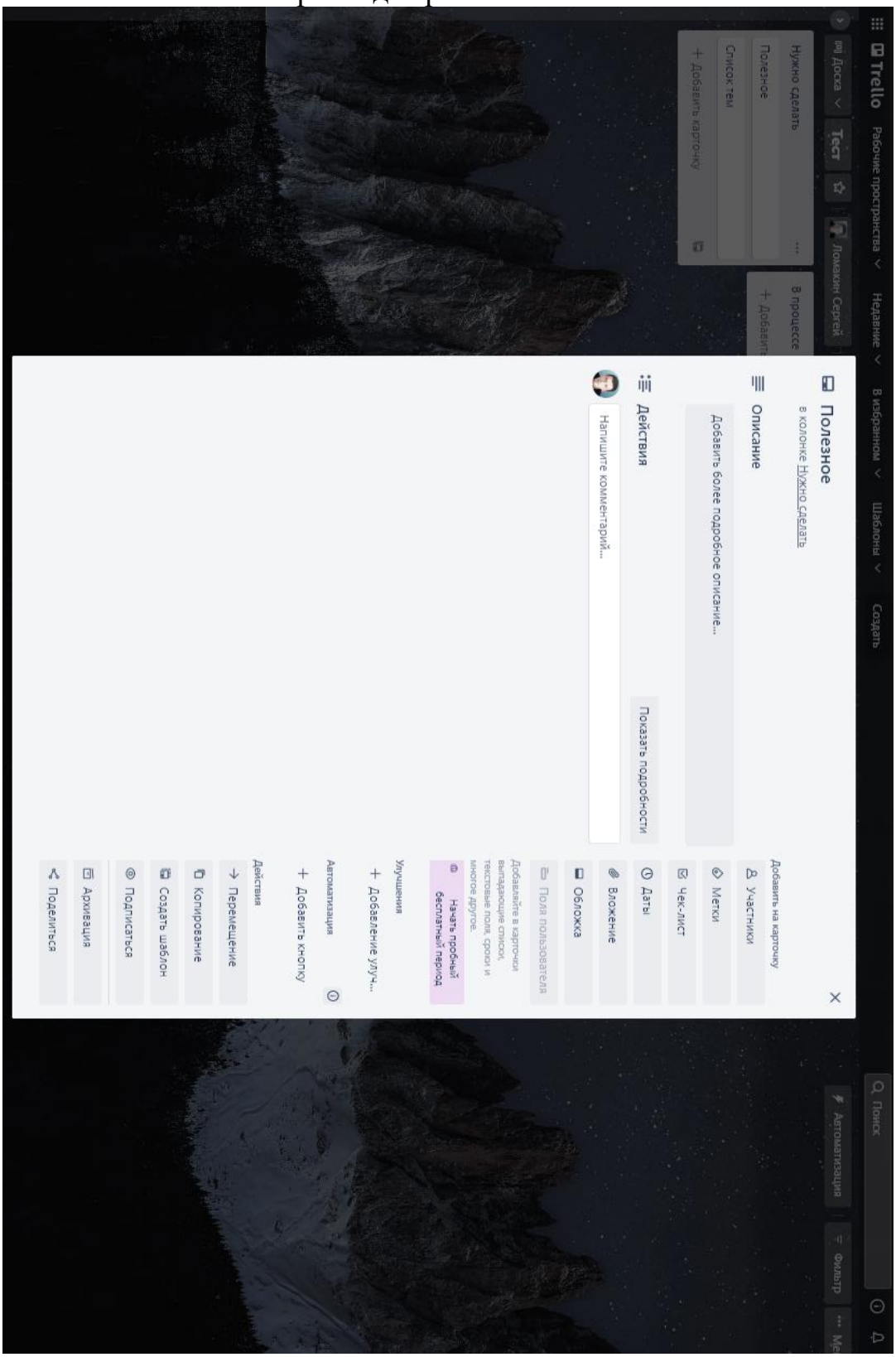


Рисунок А – Приклад карточки в Trello

Додаток Б
Діаграма бази даних додатку

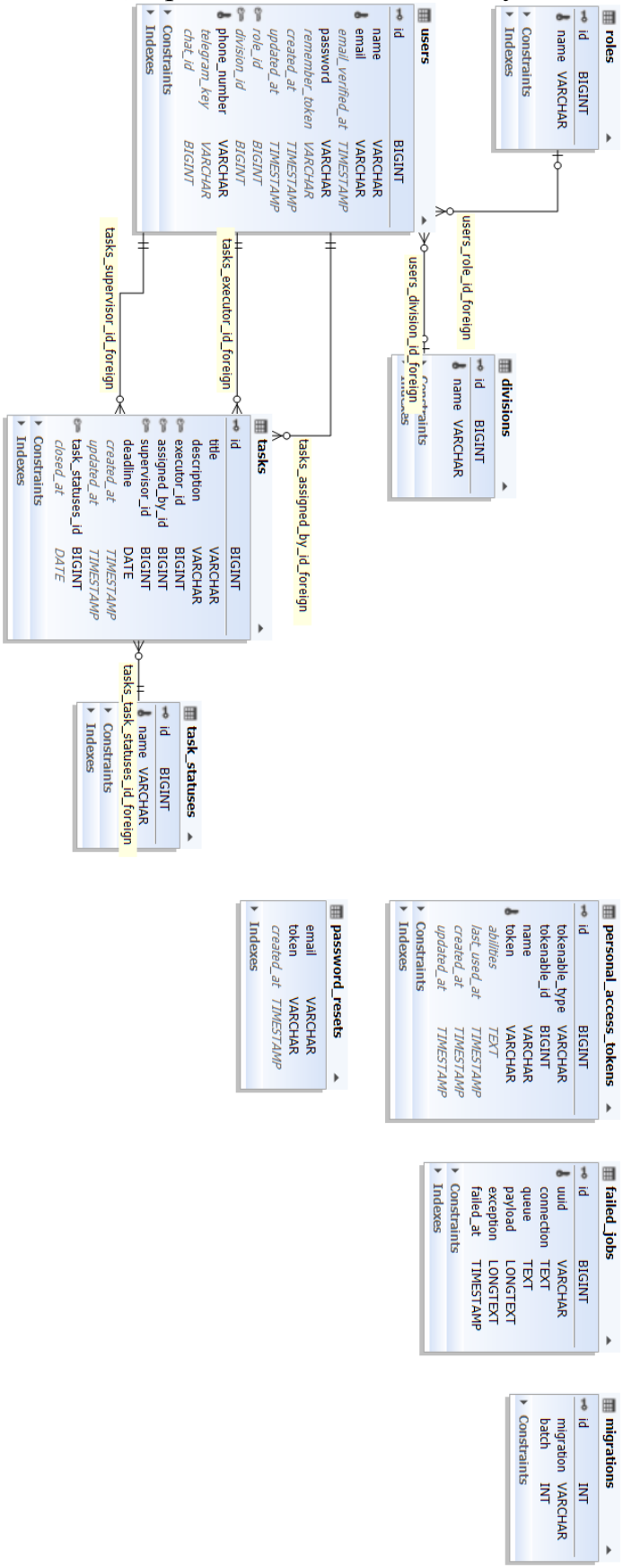


Рисунок Б – Діаграма бази даних додатку

Додаток В

Лістинг коду файлу «sidebar.blade.php», що відповідає за відображення пунктів меню в залежності від ролі користувача

```
@can('access_to_admin_func')
        <li class="nav-item">
            <a class="nav-link" href="{{
route('admin.users.index') }}">
                <svg class="nav-icon">
                    <use
xlink:href="{{asset('vendors/@coreui/icons/svg/free.svg#cil-
people')}}"></use>
                </svg>
                Users</a>
        </li>
           {{-- 464381587--}}
            <li class="nav-item">
                <a class="nav-link" href="{{
route('admin.tasks.index') }}">
                    <svg class="nav-icon">
                        <use
xlink:href="{{asset('vendors/@coreui/icons/svg/free.svg#cil-
layers')}}"></use>
                    </svg>
                    Tasks</a>
            </li>
        <li class="nav-item">
            <a class="nav-link" href="{{
route('admin.divisions.index') }}">
                <svg class="nav-icon">
                    <use
xlink:href="{{asset('vendors/@coreui/icons/svg/free.svg#cil-
sitemap')}}"></use>
                </svg>
                Divisions</a>
        </li>
        @endcan
        @can('access_to_management_func')
            <li class="nav-item">
              <a class="nav-link" href="{{ route('welcome') }}">
                    <svg class="nav-icon">
                        <use
xlink:href="{{asset('vendors/@coreui/icons/svg/free.svg#cil-
home')}}"></use>
                    </svg>
                    Main</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="{{
route('management.tasks.index') }}">
                    <svg class="nav-icon">
```

```
                        <use
xlink:href="{{asset('vendors/@coreui/icons/svg/free.svg#cil-
layers')}}"></use>
                    </svg>
                    Tasks</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="{{
route('management.users.index') }}">
                    <svg class="nav-icon">
                        <use
xlink:href="{{asset('vendors/@coreui/icons/svg/free.svg#cil-
people')}}"></use>
                    </svg>
                    Users</a>
            </li>
        @endcan
```

Додаток Г
Лістинг файлу «ReportsController.php»

```
<?php
namespace App\Http\Controllers\Management\CsvReports;
use App\Models\Task;
use Illuminate\Http\Request;
use App\Http\Controllers\Controller;
class ReportsController extends Controller
{
    public function getDailyReport()
    {
      //  task_statuses_id = 3 - In progress
        $tasks_rel = Task::with('task_executor',
'task_assigned_by', 'task_supervisor')-
>whereHas('task_executor', function ($query) {
            $query->where('division_id', auth()->user()-
>division_id);
        })
            ->where('task_statuses_id', 3)
            ->orderBy('deadline', 'ASC')
            ->get();
        $fileName = 'tasks-on-'.date('d-m-Y',
strtotime('now')).'.csv';
        $headers = array(
            "Content-type"        => "text/csv",
            "Content-Disposition" => "attachment;
filename=$fileName",
            "Pragma"              => "no-cache",
            "Cache-Control"       => "must-revalidate, post-
check=0, pre-check=0",
            "Expires"             => "0"
        );
        $columns = array('Number', 'Executor', 'Title',
'Description', 'Supervisor', 'Assigned By', 'Due Date');
        $lang = substr($_SERVER['HTTP_ACCEPT_LANGUAGE'], 0, 2);
        $delimiter = $lang == 'ru' || $lang == 'ua' ? ';' : ',';
        $callback = function() use($tasks_rel, $columns,
$delimiter) {
            $file = fopen('php://output', 'w');
            fputcsv($file, $columns, $delimiter);
            foreach ($tasks_rel as $key => $task){
                fputcsv($file, [
                    'number' => $key + 1,
                    'executor' => $task->task_executor->name,
                    'title' => $task->title,
                    'description' => $task->description,
                    'supervisor' => $task->task_supervisor-
>name,
                    'assigned_by' => $task->task_assigned_by-
>name,
                    'deadline' => /*date('Y-m-d', $task-
>deadline)*/$task->deadline->format('d-m-Y')
```

```
                ], $delimiter);
            }
            fclose($file);
        };
        return response()->stream($callback, 200, $headers);
    }
    public function getMonthlyReport() {
        // task_statuses_id = 5 - Closed
        $firstDayPreviousMonth = date("Y-m-d", strtotime("first
day of previous month"));
        $lastDayPreviousMonth = date("Y-m-d", strtotime("last
day of previous month"));
        $tasks_rel = Task::with('task_executor',
'task_assigned_by', 'task_supervisor')-
>whereHas('task_executor', function ($query) {
            $query->where('division_id', auth()->user()-
>division_id);
        })
            ->where('task_statuses_id', 5)
            ->whereBetween('closed_at', [$firstDayPreviousMonth,
$lastDayPreviousMonth])
            ->orderBy('deadline', 'ASC')
            ->get();
        $fileName = 'closed-tasks-'.date('m-Y',
strtotime($lastDayPreviousMonth)).'.csv';
        $headers = array(
            "Content-type"        => "text/csv",
            "Content-Disposition" => "attachment;
filename=$fileName",
            "Pragma"              => "no-cache",
            "Cache-Control"       => "must-revalidate, post-
check=0, pre-check=0",
            "Expires"             => "0"
        );
        $columns = array(
            'Number',
            'Executor',
            'Title',
            'Description',
            'Supervisor',
            'Assigned By',
            'Due Date',
            'Closed At'
        );
        $lang = substr($_SERVER['HTTP_ACCEPT_LANGUAGE'], 0, 2);
        $delimiter = $lang == 'ru' || $lang == 'ua' ? ';' : ',';
        $callback = function() use($tasks_rel, $columns,
$delimiter) {
            $file = fopen('php://output', 'w');
            fputcsv($file, $columns, $delimiter);
            foreach ($tasks_rel as $key => $task){
                fputcsv($file, [
                    'number' => $key + 1,
```

```
                    'executor' => $task->task_executor->name,
                    'title' => $task->title,
                    'description' => $task->description,
                    'supervisor' => $task->task_supervisor-
>name,
                    'assigned_by' => $task->task_assigned_by-
>name,
                    'deadline' => $task->deadline->format('d-m-
Y'),
                    'closed_at' => $task->closed_at->format('d-
m-Y')
                ], $delimiter);
            }
            fclose($file);
        };
        return response()->stream($callback, 200, $headers);
    }
    public function getNextMonthReport() {
        $firstDayNextMonth = date("Y-m-d", strtotime("first day
of next month"));
        $lastDayNextMonth = date("Y-m-d", strtotime("last day of
next month"));
        $tasks_rel = Task::with('task_executor',
'task_assigned_by', 'task_supervisor', 'task_statuses')-
>whereHas('task_executor', function ($query) {
            $query->where('division_id', auth()->user()-
>division_id);
        })
            ->where('task_statuses_id', '!=', 6)
            ->whereBetween('deadline', [$firstDayNextMonth,
$lastDayNextMonth])
            ->orderBy('deadline', 'ASC')
            ->get();
        $fileName = 'tasks-on-month-'.date('m-Y',
strtotime($lastDayNextMonth)).'.csv';
        $headers = array(
            "Content-type"        => "text/csv",
            "Content-Disposition" => "attachment;
filename=$fileName",
            "Pragma"              => "no-cache",
            "Cache-Control"       => "must-revalidate, post-
check=0, pre-check=0",
            "Expires"             => "0"
        );
        $columns = array('Number', 'Executor', 'Title',
'Description', 'Supervisor', 'Assigned By', 'Status', 'Due
Date');
        $lang = substr($_SERVER['HTTP_ACCEPT_LANGUAGE'], 0, 2);
        $delimiter = $lang == 'ru' || $lang == 'ua' ? ';' : ',';
        $callback = function() use($tasks_rel, $columns,
$delimiter) {
            $file = fopen('php://output', 'w');
            fputcsv($file, $columns, $delimiter);
```

```
            foreach ($tasks_rel as $key => $task){
                fputcsv($file, [
                    'number' => $key + 1,
                    'executor' => $task->task_executor->name,
                    'title' => $task->title,
                    'description' => $task->description,
                    'supervisor' => $task->task_supervisor-
>name,
                    'assigned_by' => $task->task_assigned_by-
>name,
                    'status' => $task->task_statuses->name,
                    'deadline' => $task->deadline->format('d-m-
Y')
                ], $delimiter);
            }
            fclose($file);
        };
        return response()->stream($callback, 200, $headers);
    }
}
```

Додаток Д
Лістинг TelegramController

```
<?php
namespace App\Http\Controllers\Telegram;
use App\Http\Controllers\Controller;
use App\Models\Telegram;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Http;
use Illuminate\Support\Facades\Log;
class TelegramController {
    const BASE_URL ='https://api.telegram.org/bot';
    const HTTPS_URL = '';
    const API_KEY =
'5771389620:AAE7dUiYASOifE0FNfhZy8JZWhKVRZOuNQI';
    public function index(\Illuminate\Http\Request $request) {
        $telegram = new Telegram();
        $request_obj = $telegram->getRequestObject($request);
        $from = $request_obj->from;
        $message_type = $telegram->getMessageType($request);
        if($message_type == 'message')
            $this->messageHandler($telegram, $request_obj,
$from->id);
        elseif ($message_type == 'callback_query')
            $telegram->sendMessage($from->id, 'coming soon..');
        else
            $telegram->sendMessage($from->id, 'Unknown message
type.');
        return 0;
    }
    protected function messageHandler(Telegram $telegram,
$request_obj, $chat_id){
        try {
            $text = $request_obj->text;
            Log::info('text '. $text);
            if ($text === '/start') {
                $telegram->setChatMenuButton($chat_id);
                return $telegram->sendMessage($chat_id, "Please
use the invitation link!");
            }
            elseif (str_starts_with($text, '/start ')) {
                $telegram_key = explode(' ', $text)[1];
                //User::class
                $user = $telegram->getUser($telegram_key);
                if (!$user) {
                    return $telegram->sendMessage($chat_id->id,
"The invitation link is invalid!");
                }
                if (!$user->chat_id) {
                    $telegram->addTelegramUser($user, $chat_id);
                }
                $telegram->setChatMenuButton($chat_id);
                return $telegram->sendCommandButtons($user);
```

```
            }
            elseif ($text === '/my_tasks' || $text === 'My
tasks') {
                $user = $telegram->getUser(null, $chat_id);
                Log::info('user '. $user->role->name);
                if (!$user)
                    return $telegram->sendMessage($chat_id,
"<b>Log in first!</b> Request an invitation link from the
administrator.");
                if (in_array($user->role->name, ['admin',
'manager', 'secretary']))
                    return $telegram->sendMessage($chat_id,
"Unknown command.");
                $telegram->setChatMenuButton($chat_id);
                return $telegram->sendWorkerTasks($user);
            }
            else
                return $telegram->sendMessage($chat_id, "Unknown
command, please use the buttons.");
        }
        catch(\Exception $ex){
            Log::error('error '.serialize(json_encode($ex-
>getMessage(), true)));
            return 1;
        }
        return 0;
    }
}
?>
```

Додаток Е
Лістинг моделі Telegram

```
<?php
namespace App\Models;
use Illuminate\Support\Facades\Http;
use Illuminate\Support\Facades\Log;
class Telegram
{
   const BASE_URL = 'https://api.telegram.org/bot';
   const API_KEY = ' ';
   public function getMessageType(\Illuminate\Http\Request $request)
   {
      $request_all = (object)$request->all();
      if (property_exists($request_all, 'callback_query'))
         return 'callback_query';
      if (property_exists($request_all, 'message'))
         return 'message';
      return '';
   }
   public function getRequestObject(\Illuminate\Http\Request $request)
   {
      $request_all = $request->all();
      $request_all = (object)$request_all;
      if (property_exists($request_all, 'callback_query'))
         return (object)[
            'from' => (object)$request_all->callback_query['message']['chat']
         ];
      if (property_exists($request_all, 'message'))
         return (object)[
            'from' => (object)$request_all->message['chat'],
            'text' => $request_all->message['text']
         ];
      return (object)[];
   }
   public function getUser($telegram_key = null, $chat_id = null)
   {
      if ($telegram_key)
         return User::with('role', 'division')->where('telegram_key', $telegram_key)-
>first();
      if ($chat_id)
         return User::with('role', 'division')->where('chat_id', $chat_id)->first();
      return null;
   }
   public function sendMessage($chat_id, $message)
```

```
{
    Http::post(
        self::BASE_URL . self::API_KEY . '/sendMessage',
        [
            'chat_id' => $chat_id,
            'text' => $message,
            'parse_mode' => 'html'
        ]
    );
    return 0;
}
public function setChatMenuButton($chat_id)
{
    $keyboard = [
        "keyboard" => [
            [
                ["text" =>  urldecode('My tasks')]
            ]
        ],
        "resize_keyboard" => true
    ];
    $encodedKeyboard = json_encode($keyboard);
    $res=Http::post(
        self::BASE_URL . self::API_KEY . '/sendMessage',
        [
            'chat_id' => $chat_id,
            'reply_markup' => $encodedKeyboard
        ]
    );
    return 0;
}
public function sendCommandButtons(User $user)
{
    $keyboard = [
        'inline_keyboard' => [
            [
                ['text' => 'My tasks', 'callback_data' => '/my_tasks']
            ]
        ]
    ];
    $encodedKeyboard = json_encode($keyboard);
    $message = "Welcome, <b>" . $user->name . "</b>";
    Http::post(
        self::BASE_URL . self::API_KEY . '/sendMessage',
```

```
            [
                'chat_id' => $user->chat_id,
                'text' => $message,
                'parse_mode' => 'html',
            ]
        );
        return 0;
    }
    public function sendWorkerTasks(User $user)
    {
        $keyboard = [
            'inline_keyboard' => [ [] ]
        ];
        $user_tasks = Task::with('task_statuses')
            ->where('executor_id', $user->id)
            ->orderBy('deadline', 'ASC')
            ->get();
        $count_tasks = count($user_tasks);
        if($count_tasks < 1) {
            $this->sendMessage($user->chat_id, 'No tasks.');
            return 0;
        }
        $cols = 2;
        $count_rows = $count_tasks > $cols ? ceil($count_tasks / $cols) :
$count_tasks;
        $tasks_list_message ='';
        for ($r = 0, $i=0; $r < $count_rows; $r++) {
            for ($c = 0; $c < $cols; $c++, $i++) {
                $keyboard['inline_keyboard'][$r][$c] =
                    [
                        'text' => $i.". ".$user_tasks[$i]->title,
                        'callback_data' => "/user_task|" . $user_tasks[$r + $c]->id
                    ];
                $tasks_list_message = urldecode(
                    $tasks_list_message."*".$i. "*. " . $user_tasks[$i]->title
                        ."\n_Status:_ `" . $user_tasks[$i]->task_statuses->name ."`"
                    . "\n_Deadline:_ *"
                    .date('d-m-Y', strtotime($user_tasks[$i]->deadline))."*\n"
                );
            }
            if ($count_tasks == ($i))
                break;
        }
        $message = urldecode("Your tasks:\n"). $tasks_list_message;
```

```
        $encodedKeyboard = json_encode($keyboard);
        Http::post(
            self::BASE_URL . self::API_KEY . '/sendMessage',
            [
                'chat_id' => $user->chat_id,
                'text' => $message,
                'parse_mode' => 'markdown',
                'reply_markup' => $encodedKeyboard
            ]
        );
        return 0;
    }
    public function addTelegramUser(User $user, $chat_id)
    {
        $user->chat_id = $chat_id;
        $user->save();
    }
    public function userIsAuthorized(User $user, $chat_id)
    {
        //  $user = \Illuminate\Foundation\Auth\User::where('chat_id', $chat_id)-
>first();
        return $user->chat_id ? true : false;
    }
}

?>
```