

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ
ОПТИМІЗАЦІЇ СИСТЕМИ МОНІТОРИНГУ БІЗНЕС-ПРОЦЕСІВ
RESEARCH AND SOFTWARE IMPLEMENTATION OF METHODS TO
OPTIMIZE THE SYSTEMS FOR MONITORING THE BUSINESS PROCESS

Виконав: студент 2 курсу, групи КНТ-222м

Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Системи штучного інтелекту

ФЕДОНЮК Р.П.

(ПРИЗВИЩЕ та ініціали)

Керівник ШИТІКОВА О. В.

(ПРИЗВИЩЕ та ініціали)

Рецензент ГОЛУБ Т.В.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ

Кафедра програмних засобів

Ступінь вищої освіти магістр

Спеціальність 122 Комп'ютерні науки

(код і найменування)

Освітня програма (спеціалізація) Системи штучного інтелекту

(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.

Сергій СУББОТІН

“ ” 2023 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ФЕДОНЮКА Романа Павловича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів оптимізації системи моніторингу бізнес-процесів. Research and Software Implementation of Methods to Optimize the Systems for Monitoring the Business Process

керівник проєкту (роботи) к.т.н., доцент, ШИТІКОВА Олена Вікторівна,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 14 ” листопада 2023 року № 437

2. Строк подання студентом проєкту (роботи) 01 грудня 2023 року

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

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Керівництво програміста 5. Експлуатація, тестування та експериментальне дослідження програми.

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

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ШИТІКОВА О.В., доцент кафедри ПЗ		
Нормоконтроль	ДЕЙНЕГА Л.Ю., старший викладач кафедри ПЗ		

7. Дата видачі завдання “16” жовтня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій	2 тиждень	Розділ 2
	розробки		
4	Розробка структури програми.	3 тиждень	Розділ 3
5	Розробка програми.	4-7 тижні	Розділи 3,4
6	Тестування та експериментальне дослідження	8 тиждень	Розділ 5
	програмного забезпечення.		
7	Оформлення пояснювальної записки та документів	9-10 тижні	Додатки
	до неї.		
8	Нормоконтроль та рецензування.	11 тиждень	
9	Захист роботи.	12 тиждень	

Студент(ка)

(підпис) Роман ФЕДОНЮК
(Імя ПРІЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Олена ШИТІКОВА
(Імя ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
133 с., 22 рис., 5 табл., 3 дод., 35 джерел.

БІЗНЕС-ПРОЦЕСИ, C#, TYPESCRIPT, ANGULAR, JAVASCRIPT,
МЕТОД, ОПТИМІЗАЦІЯ, МОНІТОРИНГ, АНАЛІЗ, ERP.

Об'єкт дослідження – процес оптимізації методів системи моніторингу бізнес-процесів.

Предмет дослідження – програмні системи для оптимізації моніторингу бізнес-процесів.

Метою роботи є дослідження та розробка вебдодатку для оптимізації системи моніторингу бізнес-процесів.

Матеріали, методи та технічні засоби: мова програмування C#, TypeScript, вебфреймворк Angular, середовище розробки Microsoft Visual Studio Code, фреймворк Microsoft ASP .NET Core.

Результати. Досліджено та розроблено програмну реалізацію методів оптимізації системи моніторингу бізнес-процесів.

Практична цінність роботи полягає у розробці програмного забезпечення, що оптимізує системи моніторингу бізнес-процесів та підвищує ефективність аналізу та виконання бізнес-процесів.

Висновки. Виконано проєктування програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів.

Розроблено програмне забезпечення для впровадження методів оптимізації системи моніторингу бізнес-процесів. Здійснено тестування розробленого програмного забезпечення для впровадження методів оптимізації системи моніторингу бізнес-процесів.

Галузь використання – цю програму можна використовувати для керування бізнес-процесами та видачі завдань підлеглим через браузер.

ABSTRACT

Explanatory note to the diploma qualification work of the master's degree:
133 p., 22 figs., 5 tables, 3 appendixes, 35 sources.

BUSINESS PROCESSES, C#, TYPESCRIPT, ANGULAR, JAVASCRIPT,
METHOD, OPTIMISATION, MONITORING, ANALYSIS, ERP.

Object of research - the process of optimising methods of business process monitoring system.

The subject of research is software systems for optimising business process monitoring.

The purpose of the study is to research and develop a web application to optimise the business process monitoring system.

Materials, methods and tools: C# programming language, TypeScript, Angular web framework, Microsoft Visual Studio Code development environment, Microsoft ASP .NET Core framework.

Results. The software implementation of methods for optimising the business process monitoring system has been researched and developed.

The practical value of the work lies in the development of software that optimises business process monitoring systems and increases the efficiency of business process analysis and execution.

Conclusions. The design of software for implementing methods for optimising the business process monitoring system has been completed.

Software for implementing methods for optimising the business process monitoring system has been developed. The developed software for implementing methods for optimising the business process monitoring system was tested.

Field of application - this application can be used to manage business processes and issue tasks to subordinates through a browser.

ЗМІСТ

Перелік скорочень та умовних позначок.....	9
Вступ.....	10
1 Аналіз предметної області.....	11
1.1 Проблеми і особливості сучасного впровадження систем моніторингу бізнес-процесів	11
1.2 Аналіз програмних застосунків для виконання функцій автоматизації бізнес-процесів	17
1.2.1 Програмна платформа BAS.....	19
1.2.2 Програмна платформа SAP	22
1.2.3 Програмна платформа Oracle ERP.....	24
1.2.4 Програмна платформа Microsoft Dynamics	26
1.3 Висновки за розділом 1	30
2 Матеріали і методи.....	31
2.1 Вибір мови програмування та інших технологій розробки	31
2.2 Вибір фреймворку розробки для створення програмного забезпечення оптимізації системи моніторингу бізнес-процесів	34
2.3 Вибір середовища розробки для створення програмного забезпечення оптимізації системи моніторингу бізнес-процесів	38
2.4 Метод класифікації задачі в залежності від їх пріоритету та терміновості	43
2.5 Висновки за розділом 2.....	47
3 ОПИС ПРОГРАМИ.....	49
3.1 Структура програмного забезпечення.....	49

3.2 Функціонування програмного забезпечення	52
3.3 Особливості реалізації програмного забезпечення.....	53
3.3.1 Алгоритм роботи сервісу авторизації	55
3.3.2 Алгоритм роботи сервісу управління бізнес-задачами	59
3.4 Проектування бази даних вебзастосунок реалізації методів оптимізації системи моніторингу бізнес-процесів.....	61
3.5 Висновки за розділом 3	66
4 КЕРІВНИЦТВО ПРОГРАМІСТА	68
4.1 Призначення й умови використання програми.....	68
4.2 Характеристики програми	69
4.3 Структурна схема компонентів вебдодатку	70
4.4 Висновки за розділом 4.....	74
5 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ	76
5.1 Опис роботи з програмою	76
5.1.1 Звернення до програми	76
5.1.2 Вхідні й вихідні дані	76
5.2 Інструкція з використання	77
5.3 Висновки за розділом 5.....	83
Висновки	84
Перелік джерел посилання	87
Додаток А Технічне завдання	91
Додаток Б Фрагмент тексту програми	96
Додаток В Слайди презентації.....	124

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

BAF	– Business Automation Framework;
BAS	– Business Automation Software;
BPA	– Business Process Automation;
BPMN	– Business Process Model And Notation;
CORS	– Cross-Origin Resource Sharing;
CRM	– Customer Relationship Management;
CSS	– Cascading Style Sheets;
EAM	– Enterprise Asset Management;
ECM	– Enterprise Content Management;
ERP	– Enterprise Resource Planning;
JSON	– JavaScript Object Notation;
HTML	– Hypertext Markup Language;
IoT	– Internet Of Things;
ISO	– International Organization For Standardization;
RCM	– Reliability-Centered Maintenance;
SaaS	– System As A Service;
IT	– Інформаційні Технології;

ВСТУП

Сучасні технології та інформаційні системи інтегруються в усі сфери життя та діяльність людей, підприємств і країн, сприяючи науково-технічному прогресу. Використання новаторських технологій призводить до швидкої автоматизації виробництва, що значно підвищує продуктивність. Це спричиняє появу нових бізнес-процесів, однак разом з цим зростає їх складність і час на розробку та впровадження. Системи віддаленого моніторингу та контролю стали невід'ємною частиною успішного управління компаніями, забезпечуючи оперативну реакцію на події та спрощуючи управління.

Впровадження систем віддаленого моніторингу та контролю стало особливо актуальним під час пандемії COVID, коли виникла потреба у віддаленій роботі. У таких обставинах вони стали необхідним інструментом для ефективної роботи багатьох компаній у всьому світі, в тому числі й в Україні. Серед них популярна система BAS, яка пропонує широкий функціонал для будь-якого підрозділу компанії, але має обмежену підтримку на пристроях, відмінних від комп'ютерів та ноутбуків, що ускладнює її застосування в різних ситуаціях.

Головна мета даного дослідження полягає в вивченні систем віддаленого моніторингу та контролю бізнес-процесів для вирішення проблем на підприємствах. Дослідження спрямоване на аналіз алгоритмів створення бізнес-процесів, вивчення програмних підходів до розробки систем моніторингу та контролю, розгляд існуючих систем та розробку програмного забезпечення для вирішення цих завдань.

Об'єктом дослідження є процес оптимізації методів системи моніторингу бізнес-процесів.

Предметом дослідження можна вважати програмні системи для оптимізації моніторингу бізнес-процесів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Проблеми і особливості сучасного впровадження систем моніторингу бізнес-процесів

Наразі економіка України перебуває на етапі активних реформ та адаптації до нових виробничих процесів, можливостей та освоєння нових ринків збуту. На початку 2022 року, економіка мала значний потенціал для росту та високий рівень інновацій. Впровадження інформаційних технологій у сферу підприємництва та модифікації виробництва стали важливими чинниками, що стимулюють розвиток економіки країни та підтримують її конкурентоспроможність на світовому ринку.

ІТ-індустрія України є однією з найбільш швидкозростаючих галузей країни. Ця індустрія охоплює широкий спектр можливостей – від передових робіт у сфері штучного інтелекту, кібербезпеки, обробки природної мови та нанотехнологій до висококомерційних підприємств у сфері блокчейну, фінансових технологій, управління великими даними, ігор, агротехнологій та електронної комерції.

ІТ-індустрія зросла майже в 46 разів за останні 17 років зі 110 мільйонів доларів США у 2003 році до приблизно 5 мільярдів доларів США у 2020 році [1].

При стійких темпах зростання в 20-25% розвиток української ІТ-індустрії значно випереджає середні темпи зростання галузі в світі. Кількість ІТ-фахівців в Україні постійно зростає.

Україна на шляху до того, щоб стати головним технологічним центром світу. Українська ІТ-індустрія складається з понад 4000 місцевих ІТ-сервісних компаній і понад 110 провідних світових компаній мають дочірні компанії в країні. Samsung, Microsoft, Ring, Snap, Magento, Plarium, Boeing, Siemens, Ericsson – усі мають створені R&D центри в Україні [1]. Діаграма відображення ІТ-активності в Україні зображена на рисунку 1.1.



Рисунок 1.1 – Діаграма відображення ІТ-активності в Україні [1]

Ключ до успіху ІТ-індустрії України полягає в поєднанні великого резерву кадрів у сфері ІТ, вигідного географічного положення та часового поясу, сильної технічної та наукової спадщини та конкурентоспроможних переваг.

Україна відкриває величезну невикористану можливість для технологічних компаній. Це домівка для величезного резерву робочої сили освічених і талановитих ІТ-працівників, а заробітна плата є дуже конкурентоспроможною в порівнянні з іншими розвиненими країнами. Відкривши відділ досліджень і розробок в Україні, компанії будь-якої вертикалі можуть значно підвищити інновації та продуктивність [1].

Україна традиційно була однією з провідних країн Європи за кількістю випускників інженерів, випускаючи вдвічі більше щорічно, ніж такі країни, як Великобританія та Польща. Щороку тисячі досвідчених ІТ-фахівців, більшість із яких вільно володіють англійською, закінчують навчання та починають свою професійну кар'єру в ІТ. Крім того, великі українські компанії прагнуть

організовувати власні ІТ-школи для навчання та найму найвидатніших спеціалістів [2].

Сучасний економічний розвиток неможливий без широкого впровадження та повного використання цифрових технологій. Темпи цифрових інновацій зростають з кожним роком. Кількість кінцевих споживачів зростає. Дослідження Hootsuite Global Digital 2019 показує, що в середньому щодня в Інтернеті з'являється більше мільйона нових користувачів Інтернету. У 2019 році кількість: (1) мобільних користувачів зросла на 100 мільйонів (або 2%) до 5,11 мільярда; (2) активних користувачів Інтернету – на 366 млн (або 9%) до 4,39 млрд; (3) користувачі соціальних мереж – на 288 млн (або 9%) до 3,48 млрд; осіб, які користуються соціальними мережами на мобільних пристроях – на 297 млн (або на 10%) до 3,26 млрд [2]. У жовтні 2018 року кількість активних користувачів інтернет-ресурсів в Україні становила 63% населення. За даними Factum Group Ukraine, 70% користувачів постійно використовують для виходу в Інтернет смартфони, 36% – ноутбуки, 3% – робочі комп'ютери [3]. Трансформація ландшафту міжособистісних зв'язків та економічних відносин призводить до цифровізації операційних процесів і зміни бізнес-моделей, що загалом призводить до масштабної цифрової трансформації компаній.

Зараз цифровій трансформації присвячено чимало наукових і прикладних досліджень зарубіжних фахівців, проте українські науковці почали поглиблюватися у вивченні зазначених проблем порівняно недавно. Всесвітньо визнаний цифровий експерт Девід Роджерс (2016) стверджує, що цифрова трансформація полягає не в оновленні ваших технологій, а в оновленні вашого стратегічного мислення [4]. Але трансформація бізнес-моделей у більшості випадків залишає позаду сферу досліджень, оскільки основний інтерес спрямований на впровадження технологічних інновацій. За даними 2019 року, в Україні технологічний ландшафт 4.0 включав 62 компанії, які розподілені в 16 сегментах +1 за системною інтеграцією, який є

найчисленнішим (21 компанія). Серед технологічних сегментів, які включають IoT Platform, Blockchain, IoT пристрої, Drones, Robotics, Machine Vision, Mobile Technologies тощо, лідирує сегмент Big data/AI/ML (15 компаній), другим за величиною є сегмент IoT пристроїв (12 компаній), а третє місце займає AR/VR (11 компаній) [5].

Водночас слід зазначити, що процеси цифровізації та трансформації бізнес-моделей, що відбуваються переважно на рівні приватних та державних підприємств, поширюються на рівень влади та мають значний вплив на конкурентоспроможність економіки. З одного боку, цифрова трансформація, створюючи умови для збору необхідної інформації про бізнес-середовище та операційну діяльність, прийняття обґрунтованих рішень та їх швидкої реалізації, є неминуchoю та економічно доцільною (сприяє зниженню витрат, покращенню обслуговування клієнтів, масштабованості бізнесу) [6]. Як наслідок, змінюється перспектива нинішніх можливостей для майбутнього бізнесу, держави та споживачів. З іншого боку, це вимагає від тих же компаній створення гнучких організаційних умов і впровадження цифрових бізнес-моделей, необхідних для швидкої адаптації до мінливих реалій динамічних екосистем.

Згідно з поточними дослідженнями (IMD, World Digital Competitiveness (WDC), 2018) [7], рівень цифрової конкурентоспроможності, який оцінює здатність країн впроваджувати та досліджувати цифрові технології, залежить від трьох основних факторів: знання (з підфакторами: талант, підготовка та освіта, наукова концентрація), технології (нормативно-правова база, капітал, технологічна база), готовність до майбутнього (адаптивні відносини, гнучкість бізнесу, ІТ-інтеграція). У світовому рейтингу цифрової конкурентоспроможності Україна посідає 58 місце (індекс 51 292) з 63 країн, хоча у 2017 році вона посідала 60 місце. При цьому фактор знань – 39 місце, а фактори технології та готовності до майбутнього – 61 місце, що свідчить про достатньо високий потенціал.

За індексом глобальної конкурентоспроможності [8], порівняно з 2017 роком, Україна піднялася на шість позицій, але методологія рейтингу змінилася. За рік деякі позиції знизилися, і Україна посіла 83 місце зі 140. Україна набрала найбільше балів за такими компонентами, як навички (45 місце), розмір ринку (47 місце), інфраструктура (57 місце) та інноваційний потенціал (58 місце). А за показником «рівень електрифікації» навіть вдалося посісти перше місце. Компонентами, які тягнуть Україну вниз, є макроекономічна стабільність (131 місце), фінансова система (117 місце) та інститути (110 місце).

Через такі низькі показники рейтингу України та з метою стимулювання економіки та залучення інвестицій, перетворення вітчизняних виробництв на конкурентоспроможні та ефективні, вирішення проблеми цифрового розриву, створення нових можливостей для людського капіталу, розвитку інноваційних, креативних та цифрових індустрій та бізнес; для розвитку експорту цифрових продуктів і послуг (ІТ-аутсорсинг) Кабінет Міністрів України схвалив Концепцію розвитку цифрової економіки та суспільства України на 2018-2020 роки [8]. Графік прогнозу динаміки розвитку ІТ-послуг в Україні представлений на рисунку 1.2.

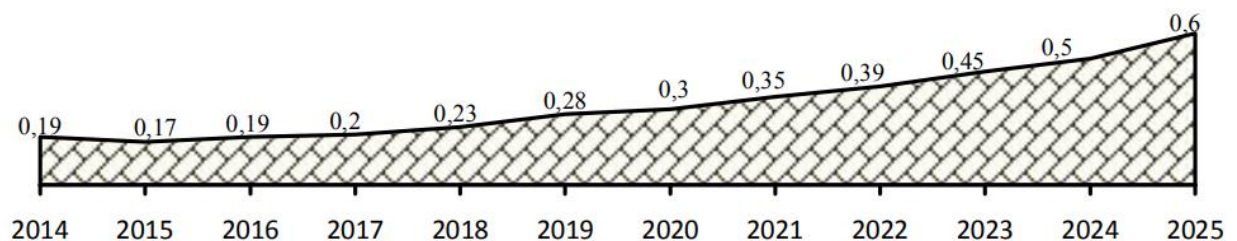


Рисунок 1.2 – Графік прогнозу динаміки розвитку ІТ-послуг в Україні, млрд дол [8].

Прийняття Концепції передувала презентація наприкінці 2016 року Digital Agenda 2020 – документу, спрямованого на затвердження пріоритетних напрямків, ініціатив, проєктів цифровізації до 2020 року [10], розробленого в рамках Стратегії «Європа 2020» з метою створення додаткових можливостей

реалізації розвитку України разом з ЄС. Саме цей документ започаткував принципи розвитку України в цифровому просторі на державному рівні та став основою для розвитку цифрової економіки. Відповідно до нього основними принципами цифровізації України є:

- будь-який член суспільства має право на доступ до цифрових технологій, Інтернету знань;
 - цифрові технології мають стати інструментом досягнення певних цілей розвитку різних секторів національної економіки;
 - можливість додаткового залучення інвестицій;
 - створення переважно українського контенту для задоволення національних потреб, що сприятиме розвитку не лише економіки, а й соціального та культурного;
 - інтеграція України в європейську та світову комунікаційну систему; -
- Необхідність розробки стандартів цифровізації, що дозволить підвищити конкуренцію та знизити витрати та собівартість продукції;
- відповідність міжнародним стандартам у фінансовій сфері;
 - уникнення орієнтації цифрової економіки лише на приватний бізнес;
 - необхідність забезпечення заходів кібербезпеки на державному рівні;
 - захист прав Інтернет-споживачів;
 - уряд має посісти одне з ключових місць у підготовці, розробці та просуванні національних цифрових стратегій [10].

Сьогодні в Україні існує безліч проєктів цифрової трансформації. Загальна ідея – створити єдину цифрову систему зв'язку. Приємно відзначити, що до цієї системи долучилися не тільки приватні компанії, а й державні структури, що свідчить про їхню зацікавленість у саморозвитку та бажання йти в ногу з сучасними технологіями, орієнтуючись на клієнта. Зокрема, створено Дорожню карту цифрової трансформації в агропродовольчому проєкті [11], яка допомагає підприємствам харчової та переробної промисловості брати участь у глобальних процесах цифрової трансформації.

Цифрова трансформація стимулює розвиток систем моніторингу та управління бізнес-процесами та підкреслює необхідність переходу підприємств на використання ERP-систем що дозволяють створювати та оптимізувати бізнес-процеси [11].

Підсумовуючи вищесказане, слід підкреслити, що Україна, маючи значний кадровий ІТ-потенціал, працює нижче своїх цифрових можливостей. Заходи, передбачені прийнятими концепціями, закладають основи для цифрової трансформації, спрямованої на цифровізацію бізнес-процесів. Однак прискорена цифровізація, паралельно з трансформацією бізнес-моделей, може додати багато мільйонів доходів до економічного зростання, додаткових міжнародних інвестицій і підвищення міжнародної конкурентоспроможності. Потрібно розуміти, що на сьогодні розвиток цифрових технологій неминучий і тільки від нас залежить, наскільки він буде ефективним і в якому майбутньому ми будемо існувати [1].

1.2 Аналіз програмних застосунків для виконання функцій автоматизації бізнес-процесів

Бізнес-процес - це послідовність дій, кроків чи етапів, які виконуються в рамках діяльності компанії з метою досягнення конкретної цілі або результату. Це може бути будь-яка операція, яка відбувається в бізнесі, від створення продукту або послуги до їхнього маркетингу, продажу, фінансового аналізу чи обслуговування клієнтів [12].

Бізнес-процеси можуть бути стандартизованими та оптимізованими для покращення ефективності та продуктивності компанії. Вони можуть включати різні етапи, ролі різних працівників, використання різних ресурсів та технологій для досягнення певних цілей. Такі процеси часто документуються, аналізуються та оптимізуються для поліпшення функціонування компанії [12].

На українському ринку представлено широкий спектр систем автоматизації бізнес-процесів, від спеціалізованих рішень для управління активами та контентом до комплексних ERP та CRM систем.

Управління активами підприємства (EAM)

EAM - це система для управління технічним обслуговуванням фізичних активів організації протягом їхнього життєвого циклу. EAM використовується для планування, оптимізації, виконання та відстеження необхідних заходів з технічного обслуговування [13].

EAM допомагає організаціям:

- знизити витрати на технічне обслуговування;
- збільшити час безвідмовної роботи обладнання;
- покращити загальну якість активів;
- відповідати регуляторним вимогам.

EAM стає все більш важливим завдяки розвитку Інтернету речей (IoT). IoT дозволяє збирати дані про активи в режимі реального часу, що дає можливість EAM використовувати предиктивне та проактивне технічне обслуговування [13].

Управління контентом підприємства (ECM)

ECM - це система для управління всіма видами контенту, який створюється або використовується компанією [13]. ECM дозволяє компаніям:

- оптимізувати бізнес-процеси;
- забезпечити доступ до необхідної інформації у потрібний час;
- збільшити продуктивність працівників;
- покращити обслуговування клієнтів;
- знизити витрати на збереження та обробку документів;
- зменшити ризик втрати даних;
- забезпечити відповідність компанії з регуляторними вимогами

[13].

ЕСМ складається з різних компонентів, таких як документообіг, управління даними, електронна архівація, відновлення даних, робота з контентом, безпека та захист даних [13].

ERP системи

ERP системи - це комплексні системи, які автоматизують основні бізнес-процеси компанії, такі як фінанси, виробництво, логістика, продажі та маркетинг [14]. ERP системи дозволяють компаніям:

- збільшити ефективність бізнес-процесів;
- покращити контроль витрат;
- знизити ризик помилок;
- покращити комунікацію між різними підрозділами компанії [14].

На українському ринку представлено безліч міжнародних та локальних ERP систем. Одними з найпопулярніших ERP систем в Україні є:

- BAS:Бухгалтерія;
- SAP;
- Oracle;
- Microsoft Dynamics.

1.2.1 Програмна платформа BAS

BAS:Підприємство — це комплексна програмна платформа для автоматизації бізнес-процесів малих і середніх підприємств. Вона розроблена компанією «NetHelp» і включає в себе різноманітні модулі, які можна використовувати для автоматизації бухгалтерського обліку, управління складом, збутом, виробництвом, кадрами, оплати праці та інших процесів [15].

BAS:Підприємство включає в себе наступні модулі:

- бухгалтерський облік — автоматизує облік фінансових операцій, включаючи облік грошових коштів, розрахунків, матеріальних цінностей, основних засобів та нематеріальних активів;

- управління складом – автоматизує облік товарів, матеріалів та інших матеріальних цінностей на складі;
- управління продажами – автоматизує процес продажів, включаючи оформлення замовлень, відвантаження товарів і послуг, а також розрахунки з клієнтами;
- управління виробництвом – автоматизує процес виробництва, включаючи планування, контроль і облік витрат;
- управління персоналом – автоматизує облік персоналу, включаючи кадровий облік, розрахунки із заробітною платою і соціальним забезпеченням;
- інші модулі – включають в себе такі модулі, як управління проєктами, управління документами, управління знаннями тощо;

Розробка BAS:Підприємство почалася в 1990-х роках для платформи DOS. У 1995 році була представлена версія 6 для Windows з графічним інтерфейсом. У 1999 році була випущена версія 7.7 з українською локалізацією та нововведеннями, які відповідали змінам у законодавстві. Ця версія підтримувалася до 2016 року, але критичні оновлення отримувала до 2020 року [15].

BAS:Підприємство впроваджено на підприємствах різних галузей економіки, включаючи промисловість, торгівлю, будівництво, транспорт, освіту та інші. За даними компанії «NetHelp», станом на 2023 рік BAS:Підприємство використовують понад 100 000 підприємств в Україні [15].

BAS:Підприємство – це потужна і багатофункціональна програмна платформа, яка може бути використана для автоматизації бізнес-процесів малих і середніх підприємств. Вона має тривалу історію розробки і впроваджена на підприємствах різних галузей економіки [15]. Приклад інтерфейсу програмної системи BAS зображений на рисунку 1.3.

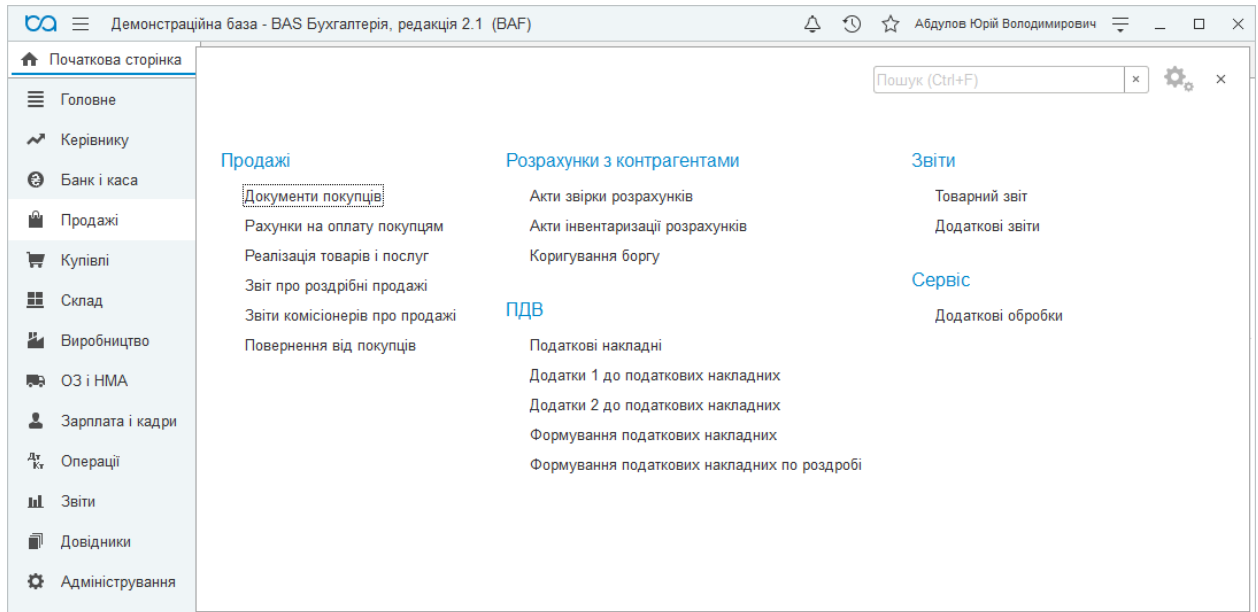


Рисунок 1.3 – Інтерфейс програмної системи BAS [15]

Business Automation Framework (BAF) – це фреймворк для автоматизації бізнес-процесів, який складається з набору рекомендацій та інструментів, що дозволяють створювати автоматизовані бізнес-процеси в компанії з мінімальними зусиллями та максимальною ефективністю [15].

BAF базується на наступних принципах:

- розбиття бізнес-процесів на етапи. Це дозволяє упорядкувати та уніфікувати процеси, а також полегшити їх автоматизацію;
- зосередження на результаті. При проєктуванні бізнес-процесів важливо враховувати кінцевий результат, який повинен бути досягнутий;
- постійна оптимізація процесів. Після впровадження автоматизованих бізнес-процесів важливо регулярно їх аналізувати та впроваджувати в них зміни для підвищення ефективності.

BAF передбачає, що кожен етап бізнес-процесу повинен бути детально розроблений та мати чітко визначені ролі та відповідальності працівників, що беруть участь у процесі. Це дозволяє забезпечити ефективне виконання процесів та мінімізувати ризик помилок [15].

BAF також рекомендує використовувати автоматизовані системи для керування бізнес-процесами. Такі системи дозволяють відслідковувати

ефективність процесів, забезпечувати їх стабільність та прискорювати їх виконання [15].

BAF – це потужний інструмент для підвищення ефективності роботи компанії. Він дозволяє зменшувати витрати, підвищувати якість продукту або послуги та підвищувати конкурентоспроможність компанії.

1.2.2 Програмна платформа SAP

SAP (Systems Applications and Products in Data Processing) – це комплексна система управління ресурсами підприємства (ERP), яка автоматизує основні бізнес-процеси компанії, такі як фінанси, виробництво, логістика, продажі та маркетинг [16].

SAP була розроблена німецькою компанією SAP SE в 1972 році і є однією з найбільш популярних ERP-систем у світі. Вона використовується компаніями всіх розмірів і галузей економіки.

SAP складається з набору модулів, які можна використовувати для автоматизації різних бізнес-процесів. Основні модулі SAP включають:

- фінанси: управління фінансами, бухгалтерський облік, бюджетування, звітність;
- виробництво: управління виробництвом, планування виробництва, контроль якості;
- логістика: управління складом, управління транспортом, управління закупками;
- продажі: управління продажами, управління клієнтами, управління маркетингом.

Приклад головного вікна інтерфейсу програми SAP, представлено на рисунку 1.4.

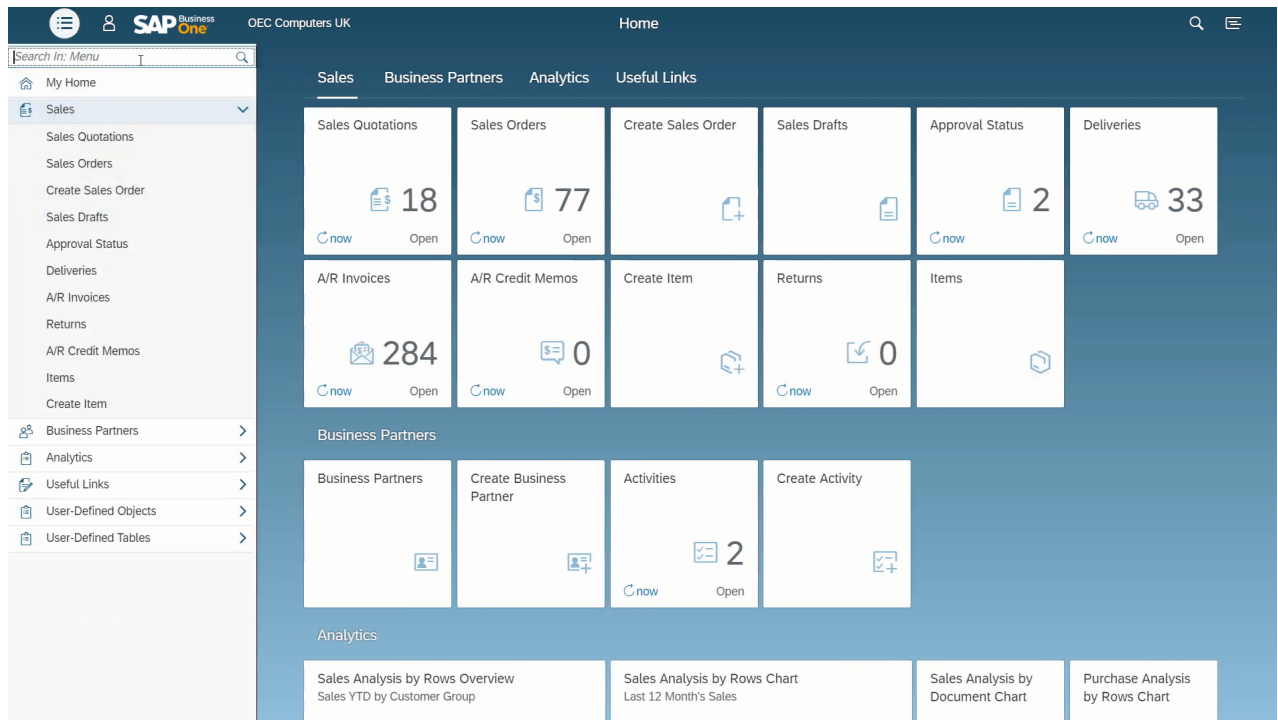


Рисунок 1.4 – Головне вікно програми SAP [16]

SAP також включає в себе ряд додаткових модулів, які можуть бути використані для автоматизації таких бізнес-процесів, як управління персоналом, управління проєктами, управління документами та ін [16].

Функції та можливості SAP:

- автоматизація бізнес-процесів: SAP дозволяє автоматизувати основні бізнес-процеси компанії, що призводить до підвищення ефективності та продуктивності праці;
- інтеграція бізнес-даних: SAP дозволяє інтегрувати дані з різних джерел, що забезпечує уніфікований погляд на бізнес-процеси компанії;
- аналіз даних: SAP дозволяє аналізувати дані з бізнес-процесів компанії, що дозволяє приймати більш обґрунтовані рішення;
- мобільні рішення: SAP пропонує широкий спектр мобільних рішень, що дозволяє працівникам компанії працювати з будь-якого пристрою;
- безпека: SAP забезпечує високий рівень безпеки даних компанії.

SAP є потужним інструментом, який може допомогти компаніям підвищити ефективність та конкурентоспроможність.

1.2.3 Програмна платформа Oracle ERP

Oracle ERP (Oracle Enterprise Resource Planning) – це комплексна система управління ресурсами підприємства (ERP), яка автоматизує основні бізнес-процеси компанії, такі як фінанси, виробництво, логістика, продажі та маркетинг [17].

Oracle ERP була розроблена компанією Oracle Corporation в 1988 році і є однією з найбільш популярних ERP-систем у світі. Вона використовується компаніями всіх розмірів і галузей економіки.

Oracle ERP складається з набору модулів, які можна використовувати для автоматизації різних бізнес-процесів [17]. Основні модулі Oracle ERP включають:

- фінанси: управління фінансами, бухгалтерський облік, бюджетування, звітність;
- виробництво: управління виробництвом, планування виробництва, контроль якості;
- логістика: управління складом, управління транспортом, управління закупками;
- продажі: управління продажами, управління клієнтами, управління маркетингом.

Oracle ERP також включає в себе ряд додаткових модулів, які можуть бути використані для автоматизації таких бізнес-процесів, як управління персоналом, управління проєктами, управління документами та ін [17].

Приклад інтерфейсу головного вікна програмної системи Oracle ERP представлено на рисунку 1.5.

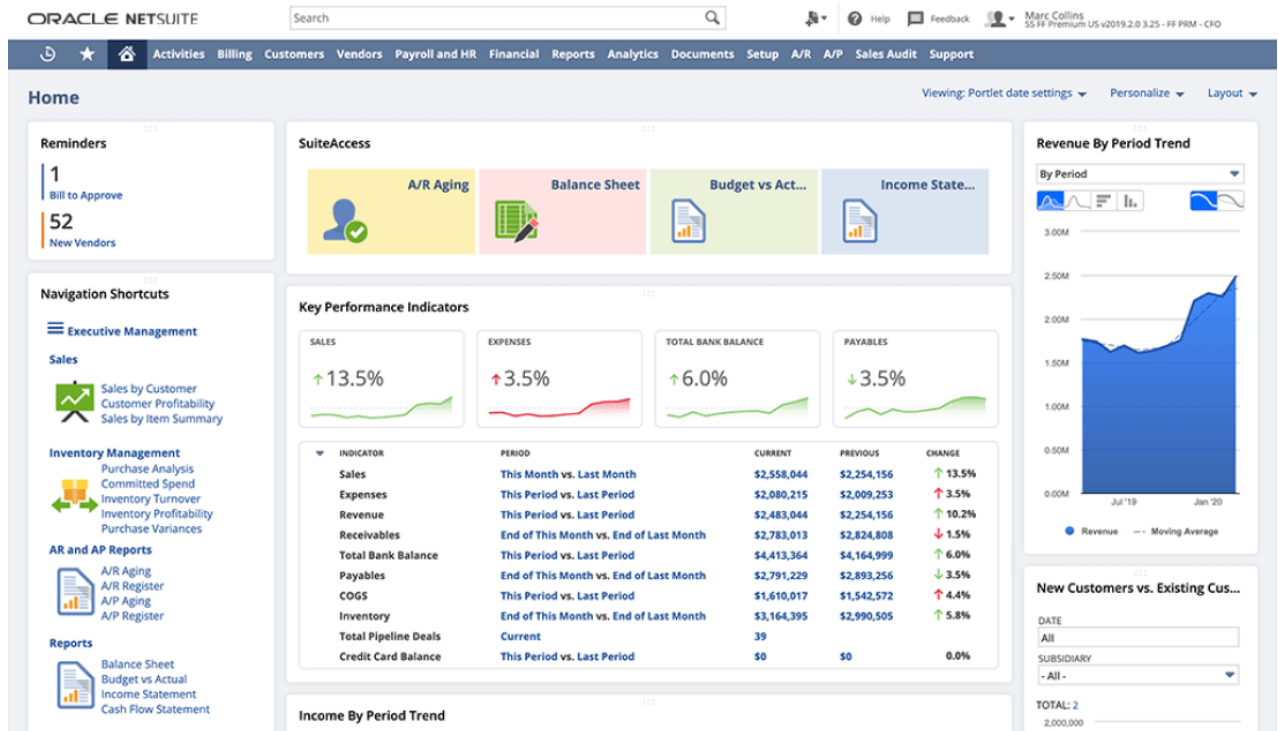


Рисунок 1.5 – Головне вікно програмного інтерфейсу Oracle ERP [17]

Функції та можливості Oracle ERP:

- автоматизація бізнес-процесів: Oracle ERP дозволяє автоматизувати основні бізнес-процеси компанії, що призводить до підвищення ефективності та продуктивності праці;
 - інтеграція бізнес-даних: Oracle ERP дозволяє інтегрувати дані з різних джерел, що забезпечує уніфікований погляд на бізнес-процеси компанії;
 - аналіз даних: Oracle ERP дозволяє аналізувати дані з бізнес-процесів компанії, що дозволяє приймати більш обґрунтовані рішення;
 - мобільні рішення: Oracle пропонує широкий спектр мобільних рішень, що дозволяє працівникам компанії працювати з будь-якого пристрою;
 - безпека: Oracle забезпечує високий рівень безпеки даних компанії.
- Oracle ERP є потужним інструментом, який може допомогти компаніям підвищити ефективність та конкурентоспроможність [17].

Додаткова інформація:

- Oracle ERP доступна як локальна, так і хмарні версії;

- Oracle ERP підтримує широкий спектр мов і валют;
- Oracle ERP є сертифікованою системою ISO 27001.

1.2.4 Програмна платформа Microsoft Dynamics

Microsoft Dynamics — це сімейство програмного забезпечення для управління бізнесом (ERP), розроблене компанією Microsoft. Microsoft Dynamics включає в себе широкий спектр продуктів, які можуть бути використані для автоматизації різних бізнес-процесів, таких як фінанси, виробництво, логістика, продажі та маркетинг [18].

Приклад головного вікна програмної системи наведено на рисунку 1.6.

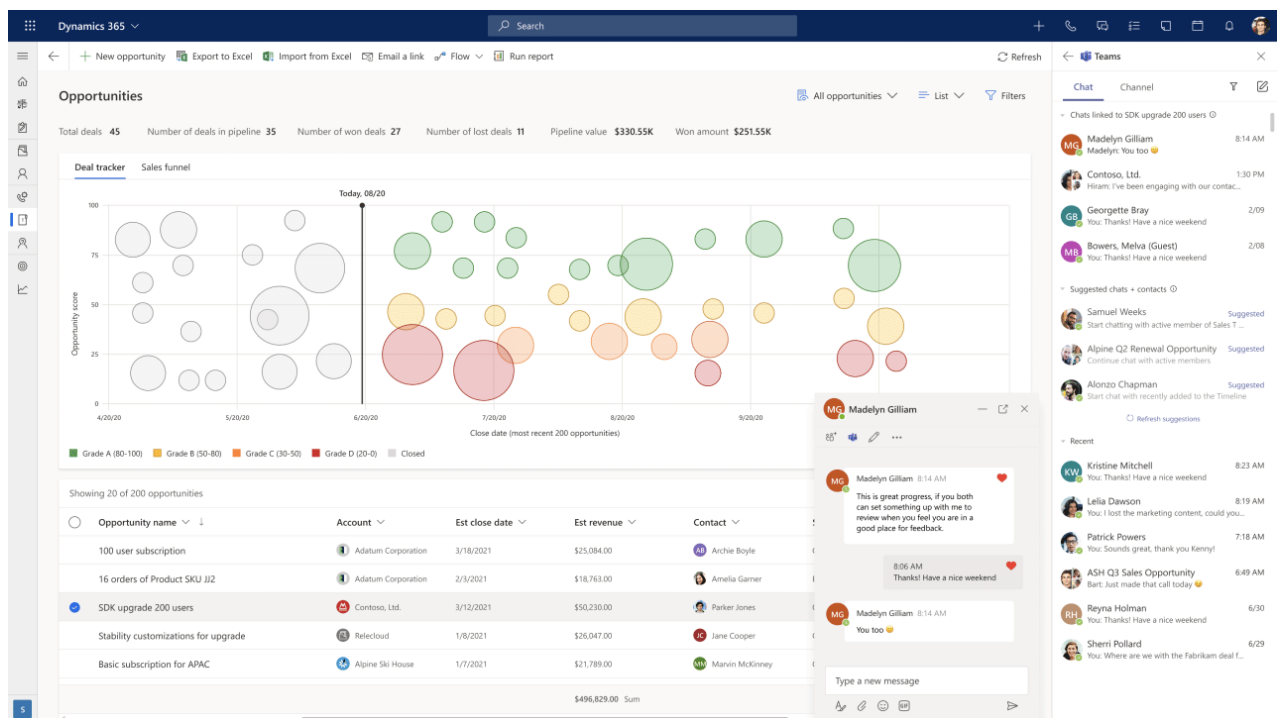


Рисунок 1.6 – Головне вікно програмної системи Microsoft Dynamics [18]

Microsoft Dynamics пропонує широкий спектр функцій та можливостей, які можуть допомогти компаніям підвищити ефективність та конкурентоспроможність [18]. До основних функцій та можливостей Microsoft Dynamics відносяться:

- автоматизація бізнес-процесів: Microsoft Dynamics дозволяє автоматизувати основні бізнес-процеси компанії, що призводить до підвищення ефективності та продуктивності праці;
- інтеграція бізнес-даних: Microsoft Dynamics дозволяє інтегрувати дані з різних джерел, що забезпечує уніфікований погляд на бізнес-процеси компанії;
- аналіз даних: Microsoft Dynamics дозволяє аналізувати дані з бізнес-процесів компанії, що дозволяє приймати більш обґрунтовані рішення;
- мобільні рішення: Microsoft пропонує широкий спектр мобільних рішень, що дозволяє працівникам компанії працювати з будь-якого пристрою;
- безпека: Microsoft забезпечує високий рівень безпеки даних компанії.

Microsoft Dynamics доступна як локальна, так і хмарна версія. Локальна версія Microsoft Dynamics встановлюється на серверах компанії, а хмарна версія Microsoft Dynamics розміщується на серверах Microsoft [18].

Microsoft Dynamics підтримує широкий спектр мов і валют.

Microsoft Dynamics є сертифікованою системою ISO 27001.

Основні продукти Microsoft Dynamics:

- Microsoft Dynamics 365: це єдина платформа для управління всіма аспектами бізнесу, включаючи фінанси, виробництво, логістику, продажі та маркетинг;
- Microsoft Dynamics NAV: це комплексна система управління ресурсами підприємства (ERP), яка автоматизує основні бізнес-процеси компанії, такі як фінанси, виробництво, логістика, продажі та маркетинг [18];
- Microsoft Dynamics AX: це масштабована система управління ресурсами підприємства (ERP), яка автоматизує основні бізнес-процеси компанії, такі як фінанси, виробництво, логістика, продажі та маркетинг;
- Microsoft Dynamics CRM: це система управління взаємовідносинами з клієнтами (CRM), яка допомагає компаніям налагодити

зв'язок з клієнтами, підвищити рівень обслуговування клієнтів і збільшити продажі [18].

Вибір між різними продуктами Microsoft Dynamics залежить від конкретних потреб компанії. Якщо компанії потрібна система для управління всіма аспектами бізнесу, то Microsoft Dynamics 365 може бути кращим вибором. Якщо компанії потрібна система для автоматизації основних бізнес-процесів, то Microsoft Dynamics NAV або Microsoft Dynamics AX може бути кращим вибором. Якщо компанії потрібна система для управління взаємовідносинами з клієнтами, то Microsoft Dynamics CRM може бути кращим вибором [18].

Порівняння розглянутих ERP систем для створення та роботи з бізнес-процесами представлено в таблиці 1.1.

Таблиця 1.1 – Порівняння систем роботи з бізнес-процесами

Характеристика	BAS	SAP	Oracle	Microsoft Dynamics
1	2	3	4	5
Розробник	NetHelp	SAP SE	Oracle Corporation	Microsoft
Функціональність	Бухгалтерський облік	Інтегрована система управління ресурсами підприємства (ERP)	ERP, CRM, HR, SCM, BI	ERP, CRM, SCM, HR
Призначення	Малий та середній бізнес	Різні галузі та масштаби підприємств	Різні галузі та масштаби підприємств	Різні галузі та масштаби підприємств
Інтеграція	Підтримує інтеграцію з іншими системами, але обмежена функціональність	Висока рівень інтеграції з іншими системами	Висока рівень інтеграції з іншими системами	Висока рівень інтеграції з іншими системами
Модульність	Основно спрямована на бухгалтерський облік	Широкий спектр модулів для управління різними аспектами бізнесу	Широкий спектр модулів для управління різними аспектами бізнесу	Широкий спектр модулів для управління різними аспектами бізнесу

Продовження таблиці 1.1

1	2	3	4	5
Клієнтська база	В основному малий та середній бізнес	Різні масштаби підприємств	Різні масштаби підприємств	Різні масштаби підприємств
Сфера застосування	Фінанси та бухгалтерський облік	Різні галузі та аспекти бізнесу	Різні галузі та аспекти бізнесу	Різні галузі та аспекти бізнесу
Доступність	Локалізована, підходить для певних регіонів	Глобальне охоплення, доступна для багатьох країн	Глобальне охоплення, доступна для багатьох країн	Глобальне охоплення, доступна для багатьох країн

За результатами проведеного порівняльного аналізу можна зробити висновок, що у сучасних програмних засобах для управління та оптимізації бізнес-процесів присутні різні переваги та недоліки які притаманні більшості комерційно орієнтованих систем. У розглянутих програмних засобах можна виділити такі основні недоліки як: висока вартість купівлі ліцензії, перевантаженість зайвим функціоналом, складність інтеграції у існуючі бізнес-процеси компанії, незручний користувацький інтерфейс.

Тому актуальною є розробка програмного забезпечення, що оптимізує алгоритми системи моніторингу бізнес-процесів.

При розробці програмна реалізація методів оптимізації системи моніторингу бізнес-процесів необхідно забезпечити такі функціональні вимоги:

- можливість створення персонально облікового запису з налаштованими правами доступу;
- можливість генерації та налаштування задач;
- можливість призначення задачі обраному співробітнику;
- пошук необхідного співробітника у базі даних;
- можливість генерації звіту виконаних завдань;
- можливість гнучкого підходу до взаємодії зі створеними бізнес-процесами.

1.3 Висновки за розділом 1

Після аналізу ринкової ситуації щодо систем проектування та автоматизації бізнес-процесів прийшли до висновку, що більшість компаній використовує рішення від BAS. Ці системи відповідають усім необхідним вимогам для автоматизації процесів управління підприємством. Однак очевидним недоліком цих систем є обмежена можливість взаємодії через мережу Інтернет. Розробник передбачив дуже обмежений програмний модуль для управління бізнес-процесами через мережу Інтернет, використовуючи сучасні кросплатформенні браузерери.

Виходячи з обмежень що описані вище було прийнято рішення розробити програмний продукт, що буде позбавлений обмеження що до роботи в мережі інтернет.

Було проаналізовано проблеми і особливості сучасного впровадження систем моніторингу бізнес-процесів та задач, розглянуто сучасні ERP – системи що надають широкий спектр функції для взаємодії з бізнес-процесами та їх налаштуванням та оптимізації. Проведено порівняння існуючих на ринку ERP – систем, перелічено їх переваги та недоліки, та виходячи з виявлених недоліків було сформовано функціональні вимоги до розроблюваного ПЗ.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування та інших технологій розробки

Вибір мови програмування, типу проєктованої системи та технологій розробки є одними із головних етапів створення будь якого програмного продукту. В цій роботі для розробки проєкту використовувались мови програмування TypeScript зі сторони фронтенду і мова C# зі сторони бекенду.

TypeScript, розроблена компанією Microsoft, є високорівневою мовою програмування з відкритим вихідним кодом, що надає статичну типізацію для JavaScript за допомогою необов'язкових анотацій типів. Основною метою TypeScript є розробка великих програмних додатків, які можуть бути скомпільовані в JavaScript. Будучи підмножиною JavaScript, всі програми, написані на JavaScript, є синтаксично коректними для TypeScript. Однак з точки зору безпеки типів, вони можуть не пройти перевірку типів, що спрямована на забезпечення безпеки програми [19].

TypeScript застосовується для розробки програмних додатків на JavaScript як на клієнтській, так і на серверній стороні (наприклад, у випадку з Node.js або Deno). Існує кілька варіантів трансляції коду. Можна використовувати вбудований компілятор TypeScript або викликати компілятор Babel для перетворення коду TypeScript в код JavaScript [20].

TypeScript включає підтримку файлів визначень, що можуть вміщувати в собі інформацію про типи наявних бібліотек JavaScript, аналогічно до файлів заголовків у C++, що описують структуру існуючих об'єктних файлів. Цей механізм дозволяє іншим програмам використовувати значення, описані у цих файлах, як статично типізовані об'єкти TypeScript. Для популярних бібліотек, таких як jQuery, MongoDB та D3.js, існують сторонні файли заголовків. Крім того, доступні заголовки TypeScript для модулів бібліотеки Node.js, що дозволяє розробляти програми на Node.js у межах середовища TypeScript [21].

Компілятор TypeScript є програмою, написаною мовою TypeScript, яка перетворюється в код JavaScript під час компіляції. Його ліцензування відбувається за умовами ліцензії Apache License 2.0. Андерс Хейлсберг, провідний архітектор C#, відомий як творець Delphi та Turbo Pascal, очолював розробку TypeScript [22].

TypeScript виникла внаслідок виявлених недоліків у JavaScript, особливо в контексті розробки об'ємних додатків, як у Microsoft, так і серед зовнішніх користувачів цієї технології. Проблематика, пов'язана з складністю коду JavaScript, породила попит на спеціалізовані інструменти для полегшення процесу розробки компонентів на цій мові програмування [23].

Розробники TypeScript націлювалися на знаходження рішення, яке б не порушувало стандарти та крос-платформенну сумісність. Однією з ключових мотивацій було прагнення використовувати майбутні можливості стандарту ECMAScript, що обіцяв підтримку програмування на основі класів. TypeScript був побудований на основі цих перспектив, розробивши компілятор JavaScript з розширенням мовних можливостей, що включали в себе певні синтаксичні розширення, відомі як надмножина, перетворення яких у звичайний JavaScript. Функціональність класу в TypeScript передбачала попереднє уявлення про те, що можна очікувати від стандарту ECMAScript 2015. Однак, унікальним аспектом, який не міститься у пропозиціях, але доданий до TypeScript, є можливість необов'язкової статичної типізації, що відома як поступова типізація. Це дозволяє статичний аналіз мови для полегшення роботи з інструментами та підтримки розробки в середовищі IDE [23].

У таблиці 2.1 проведено порівняння мови програмування TypeScript з іншими мовами програмування.

Таблиця 2.1 – Порівняння мов програмування

Характеристика	TypeScript	JavaScript	Java	C#
Тип мови	Статична з поступовою типізацією	Динамічна зі слабкою типізацією	Статична	Статична
Використання	Розробка вебдодатків, масштабні проекти	Веброзробка, повсякденні завдання	Великі платформи, мобільні додатки	Розробка на платформі .NET, ігри
Специфікація	Розширення JavaScript, додає статичну типізацію	Основна мова для вебу, динамічна типізація	Основа на віртуальній машині, статична типізація	Основа на платформі .NET, статична типізація
Екосистема	Розширює JavaScript, включає в себе всі його функції	Має широкий спектр бібліотек і фреймворків	Велика кількість бібліотек, включаючи Java SE та Java EE	Широкий вибір бібліотек та фреймворків у мережі .NET
Компіляція	Компілюється у JavaScript	Виконується безпосередньо у браузері або на сервері	Компілюється у байт-код JVM, який виконується на віртуальній машині	Компілюється у мову проміжного рівня CIL, яка виконується на платформі .NET
Строга типізація	Є статична типізація	Немає строгої типізації	Має строгу типізацію	Має строгу типізацію

У підсумку розглянутих аспектів, вважаю, що мова програмування TypeScript виявляється оптимальним вибором для виконання даного проєкту. Завдяки трансляції у JavaScript, вона має широку підтримку в більшості браузерів. Строга типізація забезпечує високий рівень безпеки програми. Простота самої мови дозволяє легко внести зміни та покращення у розроблені програмні продукти.

2.2 Вибір фреймворку розробки для створення програмного забезпечення оптимізації системи моніторингу бізнес-процесів

У сучасному світі для створення вебдодатків широко використовуються різні фреймворки. Вони допомагають скоротити час розробки, надають визначену структуру проєкту та дозволяють легко підтримувати та модифікувати великі проєкти. Найбільш популярними з них є такі: Angular, React, Vue.js.

React є відкритою бібліотекою JavaScript, призначеною для розробки інтерфейсів користувача, яка спрямована на вирішення проблем часткового оновлення вмісту вебсторінок, що часто виникають у процесі розробки односторінкових застосунків. Цей інструмент розробляється компанією Meta (раніше відомою як Facebook) та активною спільнотою індивідуальних розробників [24].

React сприяє розробці об'ємних вебдодатків, які маніпулюють даними, що постійно змінюються, без перезавантаження сторінки. Його головною метою є швидкість, простота та масштабованість. React фокусується виключно на користувацькому інтерфейсі у застосунках. Це концептуально відповідає шаблону модель-вид-контролер (MVC) і може використовуватися спільно з іншими бібліотеками JavaScript або великими фреймворками MVC, такими як AngularJS. Також React може інтегруватися як надбудова для опрацювання частин, що не стосуються користувацького інтерфейсу, у побудові вебзастосунків. У ролі бібліотеки інтерфейсу користувача React часто використовується разом з іншими бібліотеками, такими як Redux [25].

Vue.js представляє собою фреймворк JavaScript з відкритим вихідним кодом, який базується на архітектурі MVVM для розробки інтерфейсів користувача та односторінкових додатків. Його розробив Еван Ю і підтримується ним та іншими активними учасниками основної команди розробників [26].

Vue.js використовує поступово адаптивну архітектуру, яка акцентується на декларативному рендерингу та композиції компонентів. Основна бібліотека спрямована виключно на рівень представлення. Розширені можливості, необхідні для складних додатків, такі як маршрутизація, управління станом та інструменти збірки, доступні через офіційно підтримувані вспоміжні бібліотеки та пакети [27].

Vue.js надає можливість розширювати HTML за допомогою атрибутів, що інтегруються у вигляді директив. Ці директиви розширюють можливості HTML-додатків і можуть бути вбудованими у фреймворк або визначеними користувачем [27].

Angular, також відомий як "Angular 2+", є безкоштовним фреймворком вебзастосунків з відкритим вихідним кодом, побудованим на основі TypeScript. Він розробляється командою Angular Team у Google спільно з приватними особами та корпораціями у рамках спільноти [28].

Повністю переписаний тією ж командою, що створила AngularJS, Angular став фреймворком для односторінкових додатків, що використовується для швидкого розроблення вебзастосунків. Він втілює концепцію односторінкових додатків, де інтерфейс користувача завантажується під час початкового запиту, а подальші запити стосуються лише необхідних даних, що робить його високошвидкісними [28].

Angular, розроблений Google як переписаний з нуля AngularJS, відрізняється від свого попередника кількома ключовими особливостями:

- він відсутній поняття "області видимості" та контролерів, замість цього використовує ієрархію компонентів як основний архітектурний елемент;
- Angular використовує власний синтаксис виразів, зосереджений на "[]" для прив'язки властивостей та "()" для прив'язки подій;
- модульність - значна частина основної функціональності перенесена в модулі;

– Angular рекомендує використовувати мову TypeScript від Microsoft, що пропонує такі можливості як статична типізація, включаючи узагальнені типи, анотації типів, динамічне завантаження, асинхронні компіляції шаблонів, ітеративні зворотні виклики через RxJS та підтримку запуску Angular-додатків на серверах [28].

Архітектура додатків у фреймворку Angular ґрунтується на визначених фундаментальних концепціях. Основними структурними елементами фреймворку Angular є Angular-компоненти, які організовані в NgModules. NgModules спрямовані на об'єднання пов'язаного коду у логічні набори; конфігурація Angular-додатку визначається набором NgModules. У будь-якому додатку Angular завжди присутній кореневий модуль, що забезпечує його завантаження, а також, зазвичай, існує значно більше функціональних модулів [29].

Компоненти визначають представлення у вигляді наборів екранних елементів, з якими Angular взаємодіє, здійснюючи їх вибір та зміну відповідно до логіки програми та даних.

Компоненти використовують сервіси для надання специфічної функціональності, не прив'язаної безпосередньо до представлення. Постачальники послуг можуть бути додані до компонентів у вигляді залежностей, що забезпечує модульність, повторне використання та ефективність коду [28].

Модулі, компоненти та сервіси у фреймворку Angular представлені класами, які використовують декоратори. Ці декоратори встановлюють свій тип і надають метадані, які допомагають Angular у їх використанні.

Метадані для класу компонента спрямовані на зв'язок з його шаблоном, що визначає його вигляд. Шаблон об'єднує стандартний HTML з директивами Angular та розміткою зв'язування, що дає можливість Angular модифікувати HTML перед його відображенням на екрані [28].

Метадані для класу сервісу містять інформацію, необхідну Angular для забезпечення доступу до нього компонентами за допомогою ін'єкції залежностей (DI).

Компоненти програми часто описують багато представлень, що ієрархічно розташовані. Angular пропонує сервіс Router, який сприяє визначенню маршрутів між цими представленнями. Маршрутизатор забезпечує розширені можливості для навігації в браузері [29].

Також для вирішення простих задач користувацького інтерфейсу, використовувалась UI бібліотека PrimeNG.

PrimeNG - це бібліотека компонентів для веброзробки, побудована на базі Angular. Вона надає широкий спектр готових до використання UI-компонентів, таких як таблиці, форми, кнопки, меню, каруселі, графіки та багато іншого. PrimeNG дозволяє швидко створювати стильні та функціональні вебінтерфейси за допомогою готових компонентів, що спрощує процес розробки вебдодатків на базі Angular [30].

Порівняння вищевказаних фреймворків показано у таблиці 2.2.

Таблиця 2.2 - Порівняння сучасних фреймворків веброзробки

Критерій	Angular	React	Vue.js
Розробник	Google	Facebook	Evan You (індивідуально)
Тип	Повноцінний фреймворк	Бібліотека для UI	Прогресивний фреймворк
Спрощення	Є багато вбудованих функцій	Компонентно-орієнтований	Легкий у використанні
Мова	TypeScript за замовчуванням	JavaScript/JSX	JavaScript
Підтримка	Широка спільнота і підтримка	Велика активна спільнота	Активна спільнота розробників
Реактивність	Має вбудовані інструменти реактивності	Використовує віртуальний DOM	Пропонує підхід reactivity-based
Навчальний кривий	Висока	Середня	Низька
Розмір додатків	Великі	Середні	Малі
Швидкість роботи	Висока	Середня	Висока

Було вибрано Angular в якості фреймворку розробки з урахуванням його ряду важливих переваг. Цей вибір обумовлений його багатим набором вбудованих функціональних можливостей, які не потребують використання зовнішніх бібліотек, високою продуктивністю та значною популярністю серед розробників. Також він користується підтримкою від великих компаній. Angular відповідає усім технічним вимогам, викладеним у технічному завданні для дипломної роботи.

2.3 Вибір середовища розробки для створення програмного забезпечення оптимізації системи моніторингу бізнес-процесів

Серед різноманітних середовищ розробки для створення різних програмних продуктів було обрано Visual Studio Code для розробки фронтенд частини, для розробки бекенд частини була обрана Microsoft Visual Studio.

Visual Studio є інтегрованим середовищем розробки (IDE), створеним компанією Microsoft. Це програмне забезпечення застосовується для розробки комп'ютерних програм, включаючи вебсайти, вебзастосунки, вебсервіси та мобільні додатки. Visual Studio підтримує платформи розробки програмного забезпечення від Microsoft, такі як Windows API, Windows Forms, Windows Presentation Foundation, Windows Store та Microsoft Silverlight. Ця платформа дозволяє створювати як низькорівневий, так і керований код для різноманітних програмних проєктів [31].

Visual Studio містить редактор коду, що включає в себе підтримку IntelliSense (автоматичне завершення коду), а також інструменти для рефакторингу коду. Його інтегрований відладчик дозволяє проводити налагодження як на рівні вихідного коду, так і на машинному рівні. Серед інших вбудованих інструментів є профілювальник коду, конструктори для створення графічних інтерфейсів користувача, вебдизайнер, конструктори класів і схем баз даних. Visual Studio також підтримує плагіни, які

розширюють функціональність майже на всіх рівнях, включаючи додавання підтримки систем контролю версій коду (наприклад, Subversion і Git) і набори інструментів для різних аспектів розробки програмного забезпечення (наприклад, Azure DevOps: Team Explorer для співпраці над проєктами) [31].

Visual Studio підтримує 36 різних мов програмування, що дозволяє його редактору коду та інструментам для налагодження працювати з практично будь-якою мовою програмування за умови наявності відповідної служби для конкретної мови. У вбудований перелік мов входять C, C++, C++/CLI, Visual Basic .NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML та CSS. Підтримка інших мов, таких як Python, Ruby, Node.js та M, доступна за допомогою плагінів. У минулому Java (та J#) також були підтримані.

Спрощена версія Visual Studio, Community, пропонується безкоштовно. Слоган Visual Studio Community edition: "Безкоштовне повнофункціональне Інтегроване Середовище Розробки для студентів, розробників з відкритим вихідним кодом та індивідуальних розробників".

Станом на 22 листопада 2023 року, Visual Studio 2022 є останньою версією, готовою для виробництва. Версії 2013, 2015 та 2017 перебувають у режимі розширеної підтримки, тоді як версія 2019 - у режимі основної підтримки [31].

Visual Studio не вбудовує жодної конкретної мови програмування, рішення або інструменту; замість цього вона дозволяє підключати компоненти, які реалізовані у вигляді VSPackage. Після їх встановлення функціонал стає доступним як сервіс. IDE надає три основних сервіси: SVsSolution, який відповідає за управління проєктами та рішеннями; SVsUIShell, який забезпечує функціональність вікон та інтерфейсу користувача (зокрема вкладки, панелі інструментів та вікна інструментів); та SVsShell, який відповідає за реєстрацію VSPackages. Крім цього, IDE також відповідає за координацію та забезпечення взаємодії між сервісами. Усі редактори, дизайнери, типи проєктів та інші інструменти реалізовані у вигляді

VSPackages. Для доступу до VSPackages Visual Studio використовує COM. В Visual Studio SDK також включений Managed Package Framework (MPF) - набір керованих обгортки для COM-інтерфейсів, які дозволяють писати пакунки на будь-якій мові, сумісній з CLI. Однак MPF не надає повного функціоналу, який мають COM-інтерфейси Visual Studio. Сервіси можуть бути використані для створення інших пакунків, які розширюють функціональність IDE Visual Studio [32].

Підтримка мов програмування у Visual Studio здійснюється через спеціальний компонент VSPackage, відомий як мовна служба. Цей компонент визначає різні інтерфейси, які реалізація VSPackage може використовувати для надання підтримки різних функціональних можливостей. Серед можливих додаткових функцій є підсвічування синтаксису, автодоповнення операторів, автоматичне вирівнювання дужок, відображення підказок щодо параметрів, списків членів та індикатори помилок для фонові компіляції. Коли інтерфейс реалізований, ця функціональність стає доступною для конкретної мови програмування. Мовні служби втілюються окремо для кожної мови і можуть використовувати код синтаксичного аналізатора або компілятора цієї мови для повторного використання. Реалізації мовних служб можуть бути виконані або на мові власного коду, або на керованій мові. Для власного коду можуть використовуватися власні COM-інтерфейси або Babel Framework (частина Visual Studio SDK). У випадку керованого коду, MPF (Managed Package Framework) містить обгортки для створення керованих мовних сервісів [32].

Visual Studio не володіє вбудованою інтеграцією систем контролю версій для вихідного коду, проте надає два альтернативних методи взаємодії з системами контролю версій у своєму середовищі. VSPackage для контролю версій може постачати власний інтерфейс користувача. У протиположному, плагін для контролю версій, що використовує інтерфейс MSSCCI (Microsoft Source Code Control Interface), пропонує набір функцій, що використовуються для реалізації різноманітних операцій контролю версій у стандартному

інтерфейсі користувача Visual Studio. Початково MSSCCI було використано для інтеграції Visual SourceSafe з Visual Studio 6.0. У версіях Visual Studio 2005, 2008 і 2010 використовується MSSCCI версії 1.3, що додає підтримку перейменування і видалення, а також асинхронне відкриття файлів [33].

Visual Studio Code - це редактор вихідного коду, який використовується для розробки програм з різними мовами програмування, такими як C, C#, C++, Fortran, Go, Java, JavaScript, Node.js, Python, та Rust. Цей редактор ґрунтується на Electron Framework, використовуючи механізм Blink для веброзробки на Node.js. Він використовує той же компонент редактора (з внутрішньою назвою "Monaco"), що й Azure DevOps (раніше відомий як Visual Studio Online і Visual Studio Team Services) [34].

Visual Studio Code має базову підтримку для більшості поширених мов програмування, включаючи підсвічування синтаксису, вирівнювання дужок, скорочення коду та налаштовані фрагменти. Крім цього, він постачається з IntelliSense для JavaScript, TypeScript, JSON, CSS, та HTML, а також надає засоби налагодження для Node.js. Додаткова підтримка мов може бути додана безкоштовними розширеннями на VS Code Marketplace [34].

Замість системи проєктів, VS Code дозволяє користувачам відкривати один або кілька каталогів, які потім можна зберегти в робочих областях для подальшого використання. Це надає можливість робити його універсальним редактором коду для будь-якої мови програмування. VS Code підтримує багато мов програмування та надає різноманітні функції для кожної з них. Крім того, небажані файли та папки можна виключити з дерева проєкту через налаштування. Багато функцій VS Code доступні через палітру команд, а не через інтерфейс користувача чи меню [34].

VS Code можна розширити за допомогою розширень, доступних через центральне сховище, які включають різноманітні додатки для редактора та підтримку різних мов. Ці розширення можуть додавати підтримку нових мов,

тем оформлення, налагоджувачів, підтримку подорожей у часі, проведення статичного аналізу коду та лінтери через протокол Language Server [34].

У VS Code вбудована функція керування версіями. У ньому є відповідна вкладка всередині панелі меню, через яку користувачі можуть керувати налаштуваннями систем керування версіями та переглядати зміни в поточному проєкті. Щоб скористатися цією функцією, VS Code повинен бути підключений до підтримуваної системи керування версіями (Git, Apache Subversion, Perforce тощо). Це дозволяє користувачам створювати репозиторії та виконувати операції push і pull безпосередньо з програми VS Code [34].

Також у Visual Studio Code є кілька розширень для FTP, що робить його безкоштовною альтернативою для веброзробки, дозволяючи синхронізувати код між редактором та сервером без необхідності встановлювати додаткове програмне забезпечення [34].

Порівняння розглянутих середовищ розробки представлено в таблиці 2.3.

Таблиця 2.3 – Порівняння сучасних IDE

Характеристика	Visual Studio Code	Sublime Text	IntelliJ IDEA	Eclipse
Мови підтримувані	Багато (різні мови)	Багато (різні мови)	Багато (різні мови)	Багато (різні мови)
Спрощеність використання	Дуже зручний та інтуїтивно зрозумілий	Простий та швидкий	Дуже потужний та розширюваний	Розширені можливості, але менш інтуїтивний
Підтримка розширень	Широкі можливості	Можливості розширення	Широкий спектр розширень	Багато розширень
Налагодження	Вбудований налагоджувач	Підтримка налагоджування	Повний функціонал налагоджувача	Налагодження доступне
Видання	Безкоштовний та відкритий	Платний, але є пробна версія	Платний, але є Community версія	Безкоштовний та відкритий

2.4 Метод класифікації задач в залежності від їх пріоритету та терміновості

У запропонованому методі класифікації задач в залежності від пріоритету та терміновості на початковому етапі, необхідно створити екземпляр задачі (e), що буде собою являти одним із кроків бізнес-процесу. Задача повинна мати назву, опис, час створення, приблизну оцінку часу на виконання, кінцеву дату та час обов'язкового виконання.

Наступним кроком користувачу необхідно визначити пріоритет задачі аналітичним шляхом, в залежності від ліквідності цілей задачі. Що визначаються по шкалі від 1 до 10 умовних одиниць (p).

Наступним етапом необхідно обчислити терміновість створення задачі (z). Яка буде обчислена у відсотковому відношенні часу необхідного на виконання задачі на весь доступний час. Використовуючи наступну формулу (2.1):

$$Z = \frac{s}{\Delta(t_2 - t_1)}, \quad (2.1)$$

де s – час необхідний для виконання задачі;

t_1 – час створення задачі;

t_2 – дедлайн.

Існують багато методів аналітичної оцінки пріоритету та терміновості задачі. У даній роботі для цієї частини використовується Матриця Ейзенхауера.

Матриця Ейзенхауера – це метод оцінки завдань, який дозволяє ранжувати та визначати пріоритетність завдань відповідно до їх терміновості та пріоритету. За допомогою цього методу ви ділите свої завдання на чотири частини: завдання, які ви виконуєте в першу чергу, завдання, які ви

відкладаєте на потім, завдання, які ви делегуєте, і завдання, які ви відкидаєте [35].

Матриця будується на двох осях що відповідають за показники терміновості та пріоритету. Як результат оцінювання буде отримано колекцію з таких категорій задач:

- категорія А - важливі і термінові справи. Ці завдання потребують негайної уваги та важливі для досягнення мети або результату. Це дуже важливі завдання, які або вийшли з-під контролю або виникли раптово, їх слід виконати негайно, щоб уникнути наслідків. Несвоєчасне виконання таких завдань може негативно вплинути на досягнення наших цілей, стати причиною негативних подій;

- категорія В - важливі, але не термінові справи. Ці завдання також важливі, але не мають термінового статусу. Вони потребують планування та можуть бути виконані у майбутньому. Пріоритетні та перспективні завдання, спрямовані безпосередньо на досягнення наших цілей, які слід планувати та виконувати у відведений для них час. Цим справам потрібно приділяти максимум уваги та інвестувати більшу частину свого часу, вони допомагають рухатися до довгострокових цілей із найціннішими результатами;

- категорія С - не важливі, але термінові справи. Ці завдання термінові, але не є критичними для досягнення головних цілей. Вони можуть бути делеговані або вирішені швидко, але не мають великого впливу на результати. Цим справам потрібно приділяти мінімум уваги та мінімум часу, а також їх можна делегувати, автоматизувати або оптимізувати;

- категорія D - не важливі і не термінові справи. Малозначні та необов'язкові спонтанні бажання, або навіть шкідливі, але улюблені, які майже ніяк не впливають на наші цінності та не вимагають негайних дій. Це найлегші і приємніші справи в нашому списку через це деякі люди починають свій день саме з цих завдань, витрачаючи на них свій найпродуктивніші години праці. В ідеалі бажано, щоб ця категорія залишилася порожньою. За цими заняттями

ми проводимо години, але не отримуємо від них практичної користі у довгостроковій перспективі [35].

Для розподілу задачі до однієї з категорії необхідно проаналізувати значення терміновості виконання задачі (z) та заданий показник пріоритету (p).

Якщо значення терміновості буде знаходитись в діапазоні між 50% та 100%, задача буде вважатися терміною.

Якщо значення пріоритету буде знаходитись в діапазоні між 5 та 10 умовних одиниць, то така задача буде вважатися пріоритетною.

Якщо жодне з значень не входить у вказані діапазони, то вказана задача вважається недоцільною до виконання.

Для задач що вважаються терміновими і пріоритетними необхідно віднести її до категорії А.

Для задач що вважаються пріоритетними необхідно віднести її до категорії В.

Для задач що вважаються терміновими необхідно віднести її до категорії С.

Усі інші задачі необхідно віднести до категорії D.

Для визначення рейтингу важливості задачі необхідно зробити математичний розрахунок за наступною формулою (2.2):

$$R = z \times p \quad (2.2)$$

де R – рейтинг важливості задачі.

Заключним етапом буде отримання завданням рейтингу важливості задачі та додавання завдання до списку вже відсортованих завдань. Та сповіщення назначеної до виконання людини.

Кінцевим результатом виконання методу буде колекція завдань, відранжованих за критеріями пріоритету для кінцевої мети та терміновості виконання. Виконавець отримає сповіщення про появу нового завдання.

Таким чином запропоновано метод оптимізації бізнес - процесів який націлений на оптимізацію витрати часу окремими завданнями, ранжуванню їх за важливістю та терміновістю.

Схематичне представлення розробленого методу оптимізації задач представлено на рисунку 2.1.



Рисунок 2.1 - Схематичне представлення розробленого методу оптимізації задач

2.5 Висновки за розділом 2

Для створення програмної реалізації методів оптимізації системи моніторингу бізнес-процесів було обрано мову програмування TypeScript як мова програмування фронтенд частиною, C# як мова програмування бекенд частиною, які є сучасними високорівневими мовами програмування, та надають багато можливостей при створенні вебдодатків та відповідають всім потребам користувачів.

Були розглянуті сучасні вебфреймворки, проведений аналіз та створена порівняльна таблиця, за різними критеріями порівняння. Як фреймворк розробки фронтенд частини було обрано Angular. Він відповідає всім критеріям розробки, має широкі можливості створення вебзастосунків, дозволяє легко створювати, підтримувати та масштабувати вебдодатки.

Були розглянуті сучасні середовища розробки додатків. Для реалізації проєкту було обране середовище розробки Visual Studio Code, для розробки бекенд частини. Він має широку підтримку TypeScript та інших мов програмування, надає можливість гнучко налаштовувати проєкт завдяки модульній системі компонентів. Для бекенд частини було обрано середовище розробки Microsoft Visual Studio 2022. Це середовище розробки що надає широкі можливості розробки у багатьох сферах, підтримує мову програмування C# та багато інших. Надає широкі можливості з розробки, тестування та відлагодження створюваних програм.

Запропоновано метод класифікації задач в залежності від пріоритету та терміновості, у якому на першому етапі створюється нова задача, ініціалізується необхідними значеннями, далі необхідно визначити пріоритет задачі аналітичним шляхом, відповідно до цілей підприємства. Після чого проводиться обчислення терміновості задачі математичним способом вказаним в формулі 2.1. Далі необхідно розподілити задачу до категорії відповідно до отриманих значень терміновості та пріоритету. Останнім кроком буде

обчислення рейтингу важливості задачі, використовуючи формулу 2.2, та збереження отриманих результатів у колекцію відранжованих за критеріями пріоритету для кінцевої мети і терміновості виконання.

3 ОПИС ПРОГРАМИ

3.1 Структура програмного забезпечення

Структура програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів зображена на рисунку 3.1.

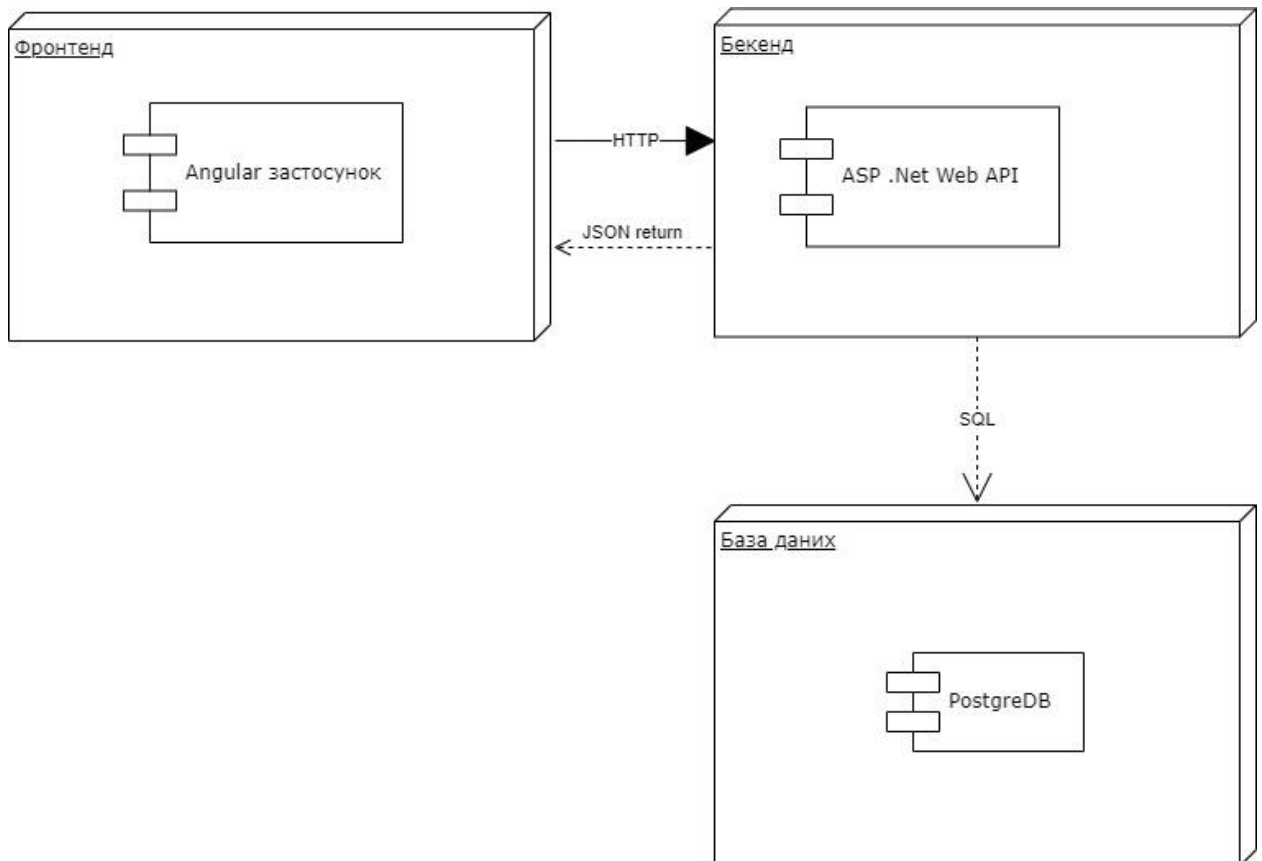


Рисунок 3.1 – Діаграма компонентів програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів.

Структура програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів виконана з використанням принципу чистої архітектури, та складається з таких частин:

- користувачька частина, виконана як самостійний Angular додаток;
- серверна частина, виконана як .Net Web API додаток;
- модуль взаємодії з клієнтською частиною;
- шар бізнес логіки;

- шар доступу до даних;
- модуль опрацювання бізнес задач;
- модуль взаємодії з базою даних;
- метод оптимізації та класифікації задач в залежності від їх терміновості та пріоритету.

Розглянемо детальніше призначення кожного компоненту програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів.

Користувачською частиною вебдодатку виступає Angular додаток, що був створений та виконується за допомогою фреймворку Angular та мови програмування TypeScript, що маніпулює HTML сторінками та стилями SCSS за допомогою згенерованого JavaScript коду. Кінцевим результатом є односторінковий вебзастосунок, що може виконуватись у будь якому сучасному браузері з підтримкою виконання JavaScript коду.

Серверною частиною виступає .Net Web API додаток, написаний на мові C#.

Модуль взаємодії з клієнтською частиною виконує такі функції як:

- обробки запиту на авторизацію в системі;
- обробку запиту на реєстрацію в системі;
- створення JWT токена авторизації користувача;
- перевірку валідності JWT токена авторизації користувача.

Шар бізнес логіки відповідає обробку запитів до створених сутностей завдань, їх валідацію та обробку відповідно до конкретних бізнес вимог.

Шар доступу до даних представляє собою провайдера даних, у даному випадку це Entity Framework Core, та відповідає за взаємодію з базою даних.

Модуль опрацювання бізнес задач відповідає за обробку дій з конкретною задачею, таких як створення, видалення, оновлення та модифікація.

Модуль взаємодії з базою даних відповідає безпосередньо за створення зв'язку з базою даних, та надає доступ для провайдеру даних.

Метод оптимізації та класифікації задач в залежності від їх терміновості та пріоритету – це окрема частина модулю опрацювання бізнес задач, що відповідає за виконання та обчислення методу оптимізації та класифікації задач в залежності від їх терміновості та пріоритету, та кінцевим результатом повертає значення рейтингу терміновості та пріоритету для кожної задачі.

Основна функціональність розроблюваної системи програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів:

- можливість створення персонального акаунта у системі;
- створення задач;
- аналіз та класифікація задачі відповідно до методу.

Вебзастосунок реалізації методів оптимізації системи моніторингу бізнес-процесів являє собою кросплатформений вебсервіс, доступ до якого здійснюється за допомогою сучасних браузерів які підтримують виконання JavaScript коду, за допомогою мережі інтернет.

Для роботи вебдодатку необхідна база даних, що забезпечують збереження усіх даних вебдодатку. Надає захищений та безперебійний доступ даних, та маніпуляцію цими даними шляхом виконання коду на мові запитів SQL. В процесі розробки була обрана СКБД Postgre SQL. Перевагами якої є швидка робота, кросплатформенність, надійність та зручність використання. СКБД була розроблена за допомогою мови програмування C з відкритим кодом, та має велику базу користувачів та розробників з різних компаній.

3.2 Функціонування програмного забезпечення

Схема функціонування програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів представлена на рис. 3.2.



Рисунок 3.2 – Схема функціонування програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів

Як можна побачити на схемі, після переходу на вебсайт користувача буде направлено на сторінку авторизації, або реєстрації. Користувачу буде необхідно ввести свої облікові дані для доступу до будь яких інших сторінок сайту.

Після проходження авторизації, у користувача з'явиться можливість переглядати існуючі та створювати нові задачі до виконання.

Наступним кроком користувач заповнює всі необхідно поля для створення нової задачі.

Після проходження валідації даних, буде проведено розрахунок рейтингу терміновості та збережено задачу до бази даних.

Результатом взаємодії користувач отримує список проаналізованих та оцінених за критерієм терміновості задач, що буде зображено у списку та помічено різними кольорами, в залежності від рейтингу терміновості.

3.3 Особливості реалізації програмного забезпечення

Загальний алгоритм роботи вебсервісу був реалізований наступним чином. Користувач заходить на сайт, після чого, якщо він не був авторизований, буде направлений на сторінку авторизації та реєстрації. Авторизація необхідна для подальшої роботи з сайтом, а також для ідентифікації користувача, визначення його прав доступу та взаємодії з задачами. Після авторизації користувач буде мати можливість доступу до сторінок задач, та перегляду даних про себе.

При переході на сторінку задач, система спробує знайти задачі в базі що призначені для авторизованого користувача, та відобразити їх в таблиці. Для додавання нової задачі необхідно натиснути кнопку «Create task». У формі що з'явилась необхідно заповнити всі поля, та натиснути кнопку «Add». Після цього дані будуть відправлені на сервер, для валідації. Якщо валідація успішно

пройдена, то буде обчислений рейтинг терміновості та задача буде збережена в базу. Алгоритм роботи вебдодатку представлено на рисунку 3.3.

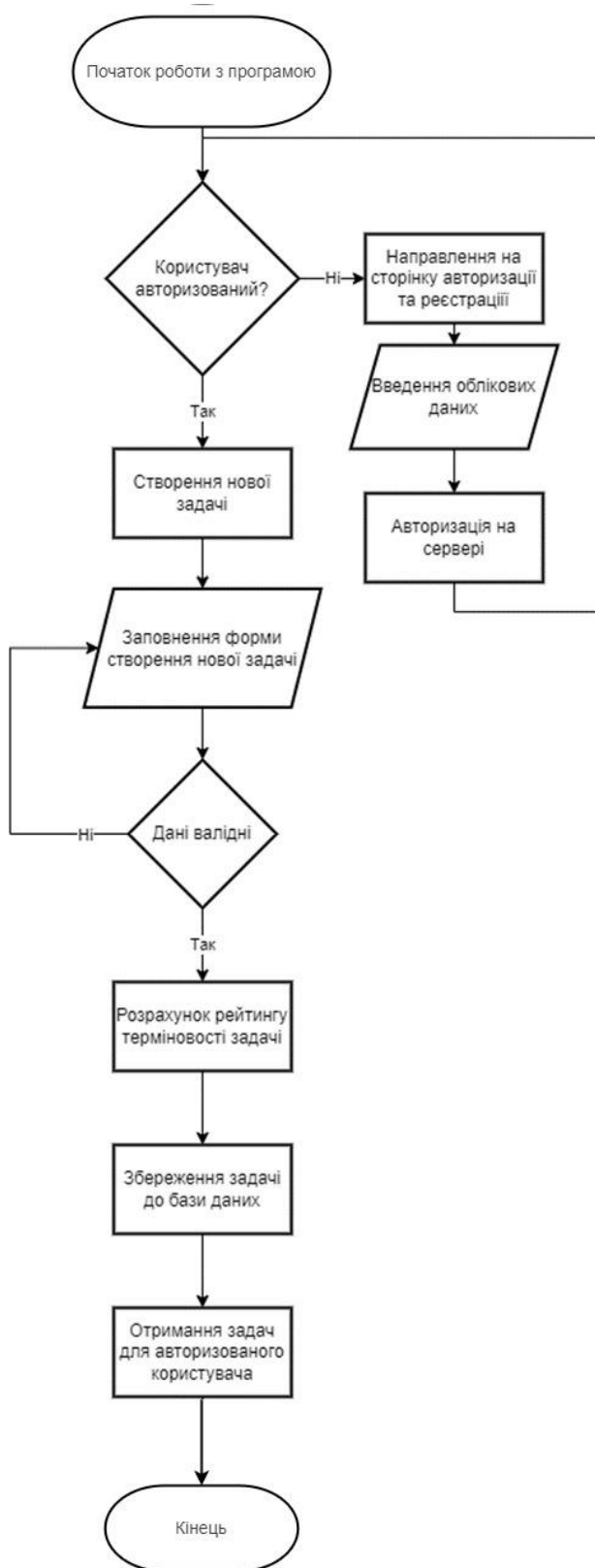


Рисунок 3.3 – Загальний алгоритм роботи вебдодатку

3.3.1 Алгоритм роботи сервісу авторизації

Функція авторизації користувачів реалізована як програмний сервіс, що вбудовується в програму за допомогою механізму ін'єкції залежностей, що підтримує мова C#. Авторизація необхідна для можливості відокремити можливості різних користувачів та мати посилання на конкретного користувача для взаємодії завдань з ним.

Якщо авторизація не пройдена тоді користувача буде направлено на сторінку авторизації та реєстрації, а весь інший функціонал вебдодатку буде недоступний.

Якщо користувач авторизований, йому буде відкрито доступ до, відповідних його ролі, функцій сайту. Алгоритм роботи сервісу авторизації представлений на рисунку 3.4.

Як основа для реалізації сервісу була використана програмна бібліотека Microsoft Identity, що надає інкапсулює базу даних користувачів та функції взаємодії з обліковими записами користувачів, що забезпечує швидкий та захищений спосіб комунікації між бізнес логікою та базою даних користувачів.

Функція входу до облікового запису користувача відповідає за перевірку валідності введених даних, та перевіряє чи існує користувач у базі даних, кінцевим результатом роботи для валідних даних буде JWT токен доступу.

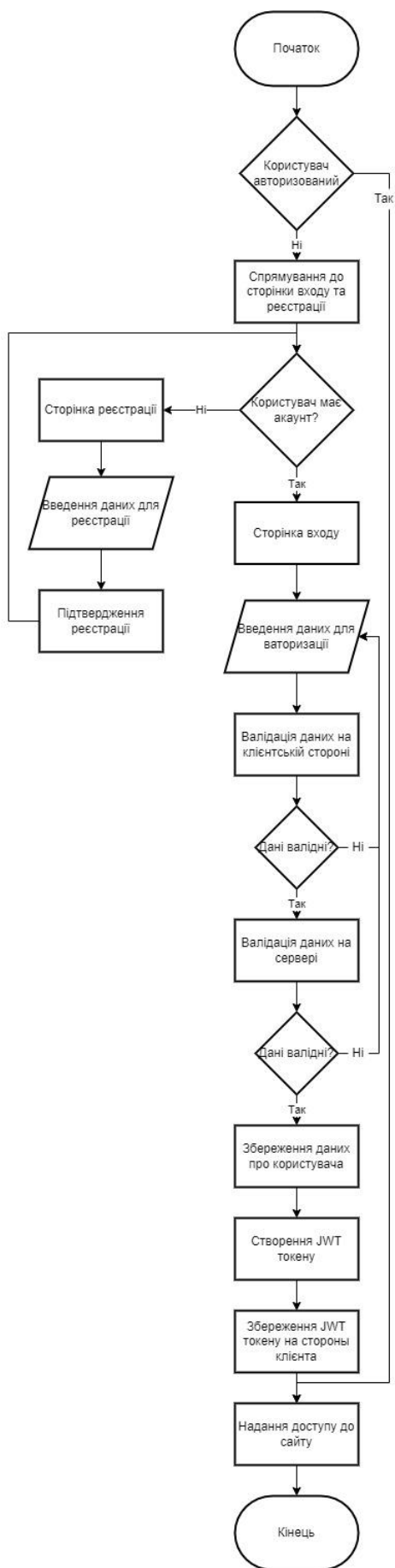


Рисунок 3.4 – Алгоритм роботи сервісу авторизації

Функція входу до облікового запису користувача.

```

public async Task<AuthResponse> Login(AuthRequest request)
{
    if (request == null)
    {
        return new AuthResponse();
    }

    LoginValidator validator = new();
    var validationResult = validator.Validate(request);

    if (!validationResult.IsValid)
    {
        return new AuthResponse();
    }

    var managedUser = await
_userManager.FindByNameAsync(request.UserName);

    if (managedUser == null)
    {
        return new AuthResponse();
    }

    bool isValidPassword = await
_userManager.CheckPasswordAsync(managedUser, request.Password);

    if (!isValidPassword)
    {
        return new AuthResponse();
    }

    var userRoles = await
_userManager.GetRolesAsync(managedUser);

    var claims = new List<Claim>
    {
        new Claim(ClaimTypes.Name, managedUser.UserName),
        new Claim(JwtRegisteredClaimNames.Jti,
Guid.NewGuid().ToString()),
    };

    claims.AddRange(userRoles.Select(claim => new
Claim(ClaimTypes.Name, claim)));

```

```

        string token = GenerateToken(claims);

        AuthResponse authResponse =
managedUser.Adapt<AuthResponse>();
        authResponse.Token = token;

        return authResponse;
    }

```

Функція реєстрації облікового запису користувача відповідає за перевірку введених даних у формі реєстрації та збереження нового користувача до бази даних.

Функція реєстрації облікового запису користувача

```

public async Task<bool> Register(RegisterRequest request)
{
    if (request == null)
    {
        return false;
    }

    RegisterValidator validator = new();
    var validationResult = await validator.ValidateAsync(request);

    if (!validationResult.IsValid)
    {
        return false;
    }

    var userExist = await
_userManager.FindByNameAsync(request.UserName);
    if (userExist != null)
        return false;

    AppUser user = request.Adapt<AppUser>();

    var createUserResult = await _userManager.CreateAsync(user,
request.Password);
    if (!createUserResult.Succeeded)
        throw new Exception($"User {request.UserName} not
{createUserResult.Errors}");
}

```

```

        var findUser = await _context.Users.FirstOrDefaultAsync(x =>
x.UserName == request.UserName);

        return findUser == null ? throw new Exception($"User
{request.UserName} not found") : true;
    }

```

Функція генерації JWT токена відповідає за генерацію токена входу відповідно до існуючих в користувача прав та відповідної йому ролі.

Функція генерації JWT токена

```

private string GenerateToken(IEnumerable<Claim> claims)
{
    JWTOptions? jwtOptions =
_configuration.GetSection(JWTOptions.Jwt).Get<JWTOptions>();
    var authSigningKey = new
SymmetricSecurityKey(Encoding.UTF8.GetBytes(jwtOptions.Secret));
    var tokenDescriptor = new SecurityTokenDescriptor
    {
        Issuer = jwtOptions.Issuer,
        Audience = jwtOptions.Audience,
        Expires = DateTime.UtcNow.AddHours(5),
        SigningCredentials = new SigningCredentials(authSigningKey,
SecurityAlgorithms.HmacSha256),
        Subject = new ClaimsIdentity(claims)
    };
    var tokenHandler = new JwtSecurityTokenHandler();
    var token = tokenHandler.CreateToken(tokenDescriptor);
    return tokenHandler.WriteToken(token);
}

```

3.3.2 Алгоритм роботи сервісу управління бізнес-задачами

Головним сервісом що відповідає за всі дії з задачами є сервіс управління бізнес-задачами. Його було реалізовано як окремий клас, що інтегрується в сервер за допомогою механізму ін'єкції залежностей.

Реалізація цього модуля надає доступ до функцій отримання, створення, модифікації та видалення бізнес-завдань, та виступає проміжним елементом між базою даних та логічною частиною. Алгоритм роботи сервісу управління бізнес-задачами представлено на рисунку 3.5.



Рисунок 3.5 – Алгоритм роботи сервісу управління бізнес-задачами

У процесі створення програмного продукту були реалізовані такі функції як:

- функція отримання завдань;
- функція отримання одного завдання;
- функція модифікації завдання;
- функція створення нового бізнес-процесу;
- функція видалення бізнес-завдання;

- функція розрахунку рейтингу терміновості.

Створений програмний код усіх вищевказаних функцій наведено у розділі пояснювальної записки «Додаток Б» у файлі `VpTaskServices.cs`.

Функція отримання завдань є основною функцією що повертає всі необхідні дані. Вона була виконана асинхронному варіанті для підвищення швидкодії обробки та можливості доступу до неї одразу декількох користувачів. Також функція має підтримку можливості пагінації, що надасть тільки необхідний користувачеві об'єм даних.

Функція отримання одного завдання за допомогою ідентифікатора необхідна для роботи з бізнес-завданням ідентифікатор якого відомий. Вона дозволяє отримати конкретний екземпляр завдання для подальшого його опрацювання.

Функція модифікації завдання необхідна для виконання будь яких модифікацій вже існуючого в базі даних завдання. Вона приймає в якості вхідного параметру модифіковане завдання, а результатом роботи функції є збереження модифікованого завдання до бази даних.

Функція створення нового бізнес-завдання необхідна для створення нового екземпляру бізнес-завдання. Вона виконує перевірку вхідних даних та кінцевим результатом є створення та збереження нового завдання в базу даних.

Функція видалення бізнес-завдання необхідна для опрацювання запиту на видалення з бази даних.

Функція розрахунку рейтингу терміновості необхідна для оцінки рейтингу терміновості для одного обраного завдання.

3.4 Проєктування бази даних вебзастосуноку реалізації методів оптимізації системи моніторингу бізнес-процесів

У процесі розробки програмної реалізації методів оптимізації системи моніторингу бізнес-процесів з'явилася необхідність збереження великої кількості різнорідних даних та забезпечення швидкого та безпечного доступу до них. Для вирішення цих завдань було прийнято рішення використовувати сучасні СКБД.

Сучасні СКБД поділяють на SQL та NoSQL бази даних. Це два основних типи систем керування базами даних, які використовуються для зберігання та організації інформації, але вони відрізняються за своїм підходом до структури та обробки даних.

База даних SQL – це реляційна база даних, яка використовує мову структурованих запитів (SQL) для зберігання, отримання та маніпулювання даними. Він кваліфікується як мова програмування. Бази даних SQL є найбільш поширеним типом реляційних баз даних, і їх використовують різні підприємства та організації [36].

База даних NoSQL - це нереляційна база даних, де не використовується традиційна таблична структура реляційних баз даних. Бази даних NoSQL часто використовуються для обробки великих обсягів даних, які не дуже добре підходять для реляційної моделі [36].

Зазвичай сучасні програми маніпулюють наборами даних із різними моделями, використаннями та сховищами. Підходів «одного розміру для всіх» недостатньо для неоднорідних даних, сховищ і схем. Зростання нових типів сховищ даних і обробки, як-от сховищ даних NoSQL, розподілених файлових систем і нових інфраструктур обробки даних, принесло нові можливості для задоволення вимог цього сценарію. Однак гетерогенність семантики, схеми та сховища, автономність і розподілена обробка все ще є одними з головних проблем при створенні додатків, керованих даними [36].

Сучасні системи керування базами даних (СКБД) включають в себе різноманітні рішення, які забезпечують ефективне зберігання, організацію та доступ до даних [36]. Одні з найбільш популярних сьогодні СКБД:

- MySQL: Це відкрита реляційна база даних, яка використовує мову запитів SQL. Вона є однією з найпоширеніших у світі та має широкий спектр застосувань, від вебдодатків до великих корпоративних систем.
- PostgreSQL: Інша популярна відкрита реляційна СКБД з високою надійністю, розширюваністю та підтримкою великого обсягу даних. Вона також має розширення для роботи з геопросторовими даними.
- MongoDB: Це NoSQL база даних, яка використовує документ-орієнтовану модель замість традиційної реляційної моделі. Вона дозволяє зберігати дані у вигляді JSON-подібних документів та добре підходить для сучасних додатків, що вимагають гнучкості у схемі даних.
- Microsoft SQL Server: Це реляційна СКБД, розроблена корпорацією Microsoft. Вона має широкий спектр функцій, включаючи аналітику, обробку даних у реальному часі та високу надійність.
- SQLite: Це легковагова, вбудовувана СКБД, яка не потребує окремого сервера. Вона часто використовується в мобільних додатках та малих проектах через свою простоту використання.

Під час розробки програмної реалізації методів оптимізації системи моніторингу бізнес-процесів було обрано СКБД PostgreSQL через простоту використання, відповідність усім необхідним вимогам розроблюваної системи. Ця СКБД є однією з найкращих реляційних баз даних, переваги якої наведено у таблиці 3.1.

PostgreSQL – це безкоштовна СКБД для реляційних баз даних, з відкритим вихідним кодом.

Таблиця 3.1 – Таблиця порівняння сучасних СКБД

Характеристики / СКБД	MySQL	PostgreSQL	MongoDB	Microsoft SQL Server	SQLite
1	2	3	4	5	6
Тип	Реляційна	Реляційна	NoSQL (Документ-орієнтована)	Реляційна	Реляційна вбудована
Мова	SQL	SQL	BSON (Binary JSON)	SQL	SQL
Схема даних	Фіксована	Фіксована	Гнучка	Фіксована	Фіксована
Масштабованість	Висока	Висока	Горизонтальна	Висока	Низька
Розширюваність	Середня	Висока	Висока	Висока	Низька
Підтримка JSON	Частково	Так	Так	Так	Ні
Геодані	Ні	Так	Ні	Так	Ні
Надійність	Добра	Дуже висока	Добра	Дуже висока	Добра
Вбудованість	Ні	Ні	Ні	Ні	Так
Використання	Широке	Широке	Вебдодатки, мобільні додатки	Широке	Локальні додатки

Схематичне зображення створеної бази даних програмної реалізації методів оптимізації системи моніторингу бізнес-процесів представлено на рисунку 3.6.

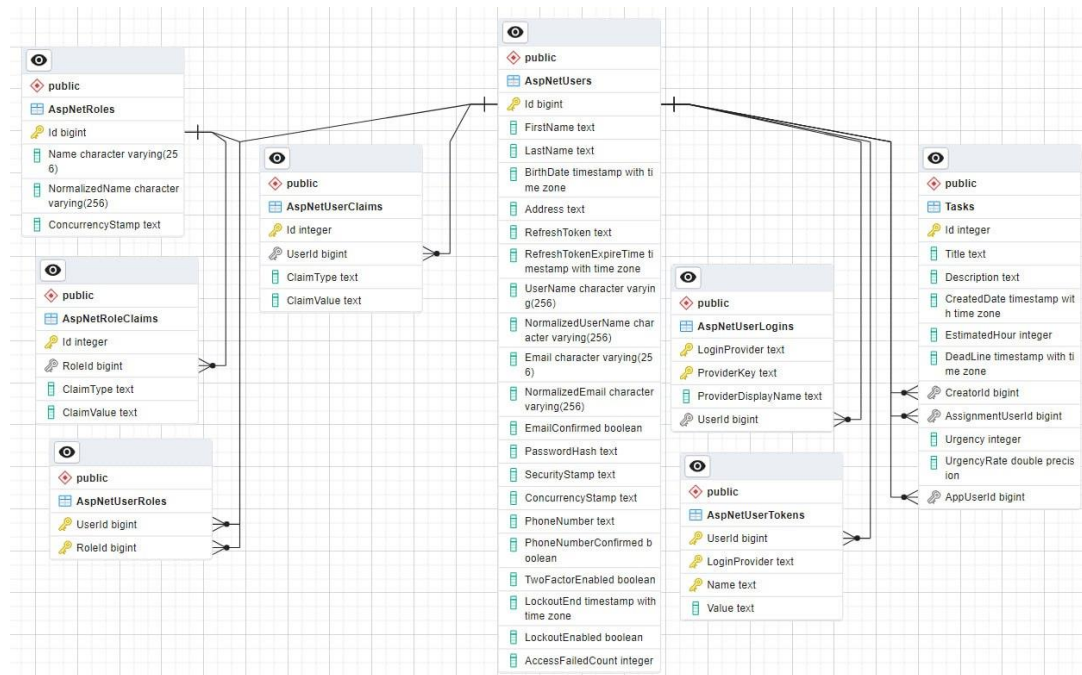


Рисунок 3.6 - Схематичне зображення бази даних програмної реалізації методів оптимізації системи моніторингу бізнес-процесів

На зображенні можна побачити що у базі даних зберігається вся необхідна інформація для роботи програмної реалізації методів оптимізації системи моніторингу бізнес-процесів.

На схемі представлені наступні таблиці бази даних:

- AspNetUsers - таблиця що зберігає всі дані про користувача;
- AspNetRoles - таблиця що зберігає дані про ролі;
- AspNetUserClaims - таблиця що зберігає дані про твердження користувачів;
- AspNetRoleClaims - таблиця що зберігає дані про твердження ролей;
- AspNetUserRoles - таблиця що необхідна для зв'язування таблиць користувачів та ролей;
- AspNetUserLogins - таблиця що зберігає дані логування;
- AspNetUserTokens - таблиця що зберігає токен відповідного користувача;
- Tasks - таблиця що зберігає дані про завдання.

Розроблені таблиці бази даних, що представлені на схемі, пов'язані між собою зв'язками за допомогою первинних та зовнішніх ключів.

Таблиці мають наступні зв'язки:

- таблиця “AspNetUsers” пов'язана з таблицею “AspNetRoles” зв'язком багато до багатьох за допомогою проміжної таблиці “AspNetUserRoles ”;
- таблиця “AspNetUsers” пов'язана з таблицею “AspNetUserClaims” зв'язком один до багатьох;
- таблиця “AspNetUsers” пов'язана з таблицею “AspNetRoles” зв'язком один до багатьох;
- таблиця “AspNetRoles” пов'язана з таблицею “AspNetRoleClaims” зв'язком один до багатьох;
- таблиця “AspNetUsers” пов'язана з таблицею “AspNetUserLogins” зв'язком один до багатьох;
- таблиця “AspNetUsers” пов'язана з таблицею “AspNetUserTokens” зв'язком один до багатьох;
- таблиця “AspNetUsers” пов'язана з таблицею “Tasks” зв'язком один до багатьох.

3.5 Висновки за розділом 3

Розроблено програмну реалізацію методів оптимізації системи моніторингу бізнес-процесів. Наведено структуру та описано основні сервіси, компоненти та модулі програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів. У ході розробки було визначено що програма буде складатися з двох частин: клієнтської частини, що представляє з себе Angular застосунок, та серверної частини, що представляє собою Asp .Net Web API.

Наведено схему функціонування програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів.

Описано особливості реалізації розробленого програмного забезпечення, наведено алгоритм роботи. Детально розглянуто алгоритми авторизації та створення нової задачі. Наведено приклади реалізації функцій що виконуються у цих алгоритмах.

Проведено огляд існуючих СКБД. Наведена порівняльна таблиця сучасних систем управління базами даних. Здійснена розробка та створення бази даних для забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення й умови використання програми

Розроблений програмний продукт призначений для програмної оптимізації методів системи моніторингу бізнес-процесів. Вебзастосунок розроблений за допомогою мов програмування C# та TypeScript. Для візуальної частини сайту використовувалась мова HTML та скриптова метамова SCSS, яка інтерпретується в CSS. Були використані такі технології як ASP.Net Web Api, Angular. Була використана фронтенд бібліотека компонентів PrimeNG. Для взаємодії з базою даних використовувався ORM технологія Entity Framework Core, з підключенням до нього провайдером даних PostgreSQL/Npgsql.

Програмна реалізація методів оптимізації системи моніторингу бізнес-процесів забезпечує виконання таких функцій:

- можливість створення персонального облікового запису користувача;
- авторизація користувача;
- відображення інформації про авторизованого користувача;
- можливість перегляду усіх завдань у відсортованому за рейтингом вигляді;
- можливість створення нового завдання;
- можливість редагування вже існуючого завдання;
- можливість видалення завдання;
- можливість позначення завдання як виконане.

Для комфортної роботи з програмним забезпеченням необхідно використовувати таке апаратне та програмне забезпечення:

- оперативна пам'ять – не менше 2Гб;
- операційна система з можливістю взаємодії через мережу інтернет;

- браузер з підтримкою виконання JavaScript коду;
- графічна карта з підтримкою DirectX 10 або краще;
- процесор – 1.8 ГГц або швидше;
- порт RJ45 або мережевий адаптер.

4.2 Характеристики програми

Вебзастосунок програмної оптимізації методів системи моніторингу бізнес-процесів після процесу розвертання на сервері може безперервно працювати, в залежності від потреб користувачів, швидкість роботи вебзастосунку залежить від пропускної здатності інтернет підключення та апаратних та програмних можливостей серверу на якому встановлена програма.

Під час виконання програмний продукт оптимізації методів системи моніторингу бізнес-процесів надає інформацію користувачу у вигляді динамічно сформованих вебсторінок які відображають наступний контент:

- форми авторизації та реєстрації;
- секція з інформацією про активного користувача;
- таблиця відранжованих завдань.

Під час роботи програмний продукт оптимізації методів системи моніторингу бізнес-процесів активно використовує можливості міждоменної взаємодії, для спілкування між фронтенд та бекенд частиною, керуючись наявними правилами CORS.

Для роботи з даними програмний продукт оптимізації методів системи моніторингу бізнес-процесів активно використовує усі можливості сучасних СКБД для асинхронних операцій взаємодії з великими об'ємами даних, що утворюються в процесі роботи.

4.3 Структурна схема компонентів вебдодатку

Проектована система була розроблена так що складається з двох частин, клієнтська частина створена на фреймворку Angular, серверна частина розроблена за допомогою технології ASP .Net Web API.

Таке розподілення надає проекту гнучкості, яке виражається у тому що, при необхідності, до серверної частини можливо створити інший клієнтський модуль, що буде незалежно отримувати дані та взаємодіяти з ними. Також в такій архітектурі при необхідності заміни серверної частини, користувач не відчує ніяких незручностей у роботі з сервісом. Структурна схема компонентів вебдодатку зображена на рисунку 4.1.

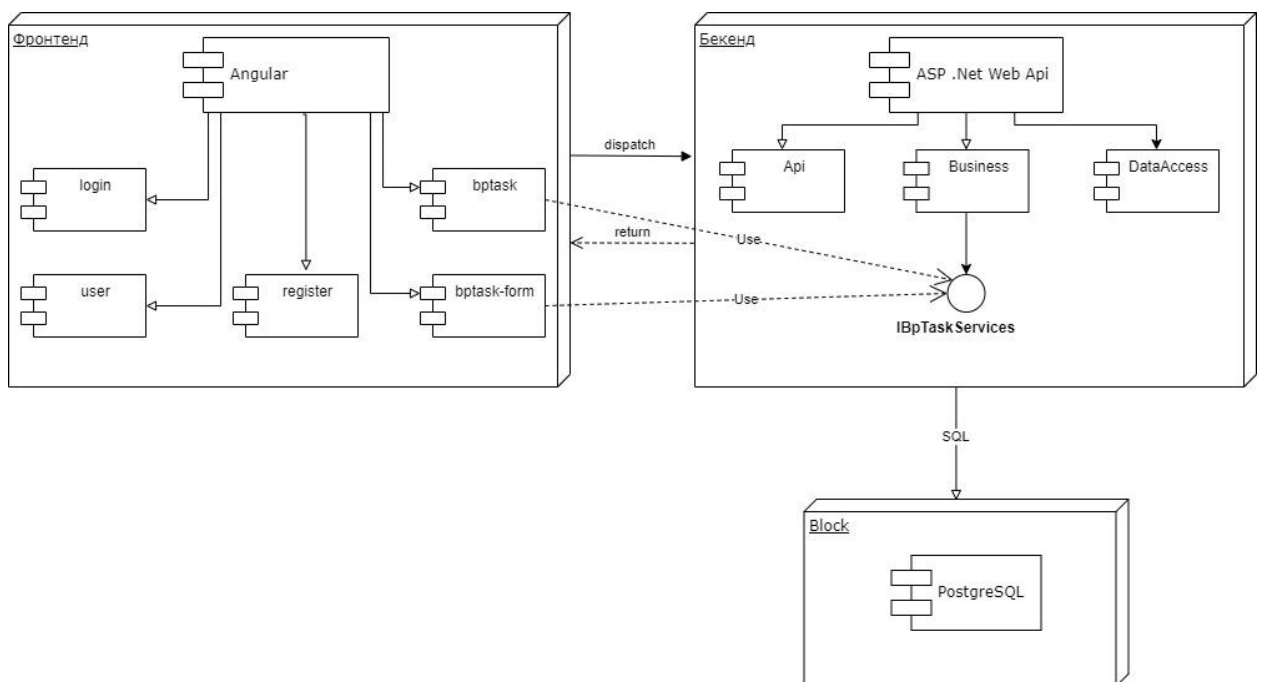


Рисунок 4.1 - Структурна схема компонентів вебдодатку

Розглянемо детальніше клієнтську частину вебдодатку.

Клієнтська частина розробленої системи написана на мові програмування TypeScript та в якості фреймворку розробки було використано Angular. Для формування компонентів на сторінках було використано ui

бібліотеку PrimeNG. Структурна схема компонентів клієнтського Angular додатку представлена на рисунку 4.2.

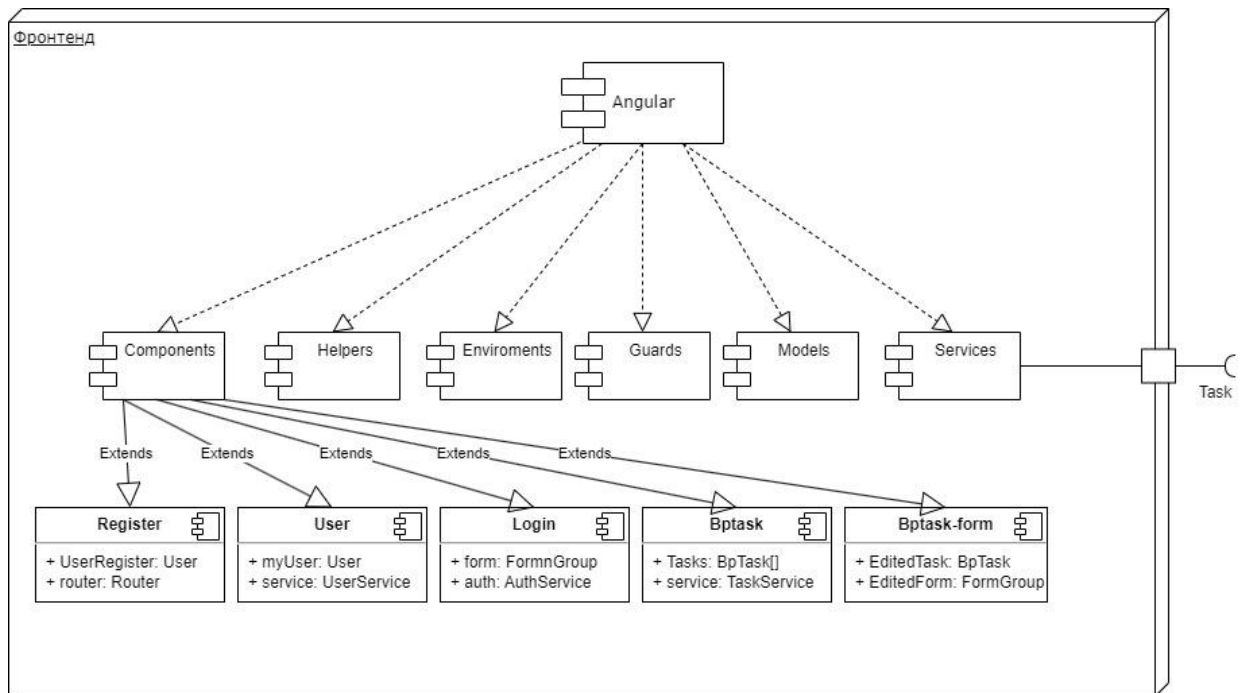


Рисунок 4.2 – Структурна схема компонентів клієнтського Angular додатку

На структурній схемі представлено всі компоненти які були створені у процесі реалізації методів системи моніторингу бізнес-процесів.

Компонент **Services** – відповідає за запити до серверної частини. Цей компонент містить такі сервіси як:

- **AuthService** – сервіс відповідає за запити, до кінцевих точок серверу, що відповідають за авторизацію користувача;
- **UserService** – сервіс відповідає за запити, до кінцевих точок серверу, що відповідають за дані про користувачів;
- **TasksService** – сервіс відповідає за запити, до кінцевих точок серверу, що відповідають за дані про завдання.

Компонент **Models** – відповідає за створені моделі даних. Він містить такі елементи як:

- **BpTask** – модель, що представляє дані про завдання;

- `ResponseWrapper` – модель, що представляє уніфіковану відповідь від серверу;
- `PagedResponse` – модель, що представляє пагіновану відповідь від серверу;
- `Token` – модель, що представляє токен доступу до вебзастосунку;
- `User` – модель, що представляє користувача вебзастосунку;
- `UserRegister` – модель, що представляє дані які користувач заповнює у компоненті реєстрації.

Компонент `Guards` – відповідає за обмеження маршрутизації та завантаження Angular компонентів.

Компонент `Enviroments` – відповідає за змінні оточення, за допомогою яких додаток вирішує який сервер може бути відкритим для взаємодії.

Компонент `Helpers` – відповідає за додаткові сервіси, які неможливо однозначно віднести до одного з існуючих. Він містить такі сервіси як:

- `ErrorInterceptor` – спеціальний сервіс що виконує перехват повідомлень про помилки;
- `JwtInterceptor` – спеціальний сервіс що встановлює JWT токен автентифікації для кожного запиту до серверу.

Папка `Components` містить компоненти фреймворку Angular, що представляють собою самостійні частини функціоналу зі своєю логікою, HTML шаблоном та CSS стилями. Кожен такий компонент для користувача представляє сторінку, зі своєю особливою логікою.

Компонент `Register` – відповідає за сторінку реєстрації, валідацію форми вводу та логіку обробки та відправки форми на сервер.

Компонент `Login` – відповідає за сторінку входу, валідацію форми вводу та логіку обробки та відправки форми на сервер.

Компонент `User` – відповідає за сторінку відображення інформації про користувача.

Компонент Vptask – відповідає за сторінку відображення завдань та надає доступ до операцій над завданнями, керує логікою та обміном даними з сервером.

Компонент Vptask-form – відповідає за сторінку відображення форми створення та редагування завдань та передає отримані дані до відправки на сервер.

Розглянемо детальніше серверну частину вебдодатку.

Серверна частина розроблена на мові C# з використанням технології ASP .Net Web API. Вона складається з п'яти логічних частин: Api, Business, Common, DataAccess, Models. Структурна схема логічних частин серверної частини представлена на рисунку 4.3.

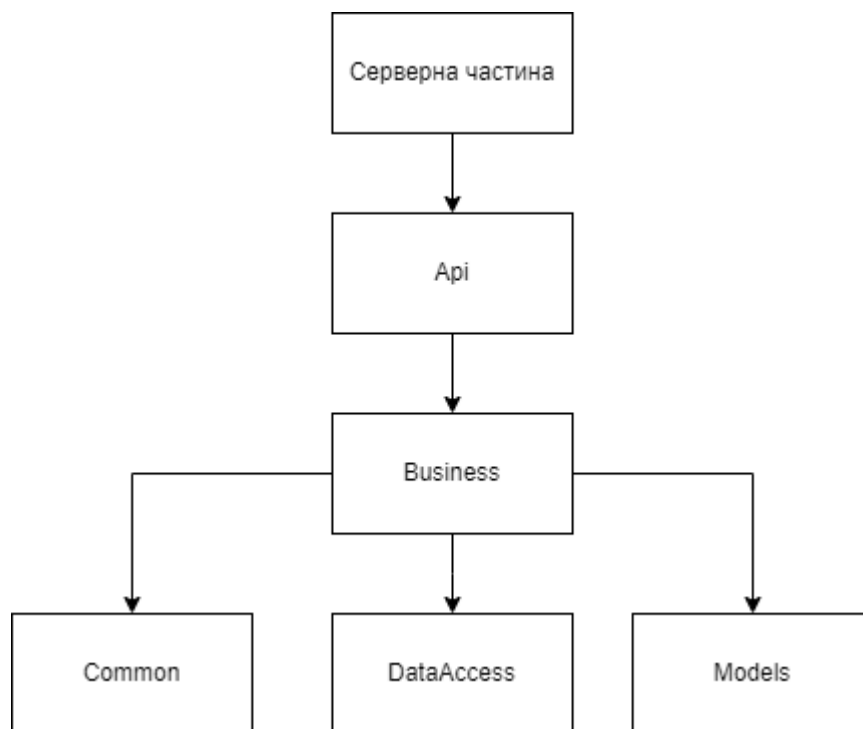


Рисунок 4.3 – структурна схема логічних частин серверної частини

Models – відповідає за надання моделей для представлення вхідних та вихідних даних.

Common – відповідає за надання сервісів які не відносяться до окремої частини програми, та надають загальну логіку.

DataAccess – відповідає за доступ до бази даних, визначає сутності бази даних, налаштування бази даних та відношення сутностей один з одним, зберігає стан та міграції бази даних.

Business – відповідає за виконання основної логіки додатку, містить інтерфейси логіки для інверсії залежностей та валідаційні правила Fluent валідатора.

Арі – виступає як основний web арі проєкт, відповідає за взаємодію з клієнтською частиною, надає кінцеві точки взаємодії, виконує компонування інших компонентів. Точкою входу програми є файл Program.cs, він відповідає за побудову конфігурації проєкту, виконує ін'єкцію залежностей сервісів в проєкт, підключення контексту бази даних, налаштування правил міждоменної комунікації, налаштування конфігурації автентифікації за допомогою JWT схеми доступу та виконує кінцеву побудову проєкту. Також в цьому файлі визначаються налаштування доступні лише для розробників, такі як реалізація специфікації OpenAPI, програмний інструмент Swagger, що використовується для швидкого та безпечного тестування додатку.

4.4 Висновки за розділом 4

У цьому розділі було розглянуто призначення й умови використання програмної оптимізації методів системи моніторингу бізнес-процесів.

Було наведено основні характеристики створеного програмного забезпечення.

Описано структурну схему розробленої системи.

Було розглянуто структурну схему компонентів вебдодатку. Детально розглянуто структурну схему клієнтської частини, описано функціональність створених компонентів.

Детально розглянуто структурну схему серверної частини, описані логічні модулі серверної частини. Описано функціональність створених логічних модулів.

5 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

5.1 Опис роботи з програмою

Цей вебзастосунок розроблено для програмної реалізації методів оптимізації системи моніторингу бізнес-процесів. Реалізований на мові програмування TypeScript у фреймворку Angular, він дозволяє взаємодіяти з завданнями віддалено через будь-який пристрій, що підтримує взаємодію через мережу Інтернет через браузер.

5.1.1 Звернення до програми

Звернення користувачів до вебзастосунку програмної реалізації методів оптимізації системи моніторингу бізнес-процесів виконується за допомогою будь-якого сучасного веббраузера, що підтримує виконання JavaScript коду.

5.1.2 Вхідні й вихідні дані

Вхідними даними до вебзастосунку програмної реалізації методів оптимізації системи моніторингу бізнес-процесів є HTTPS запит який в собі може містити дані, над якими потрібно виконати дії, з вказаним типом запиту у заголовку.

Вихідними даними програмного продукту реалізації методів оптимізації системи моніторингу бізнес-процесів є згенерована HTML сторінка з CSS стилями, що відповідає на HTTPS запит користувача.

5.2 Інструкція з використання

Після переходу за вебадресою доступу до сторінок сервісу, перед користувачем буде згенерована форма входу, що містить у собі навігаційне меню у верхній частині сторінки, та форму вводу облікових даних, по середині сторінки (рис. 5.1).

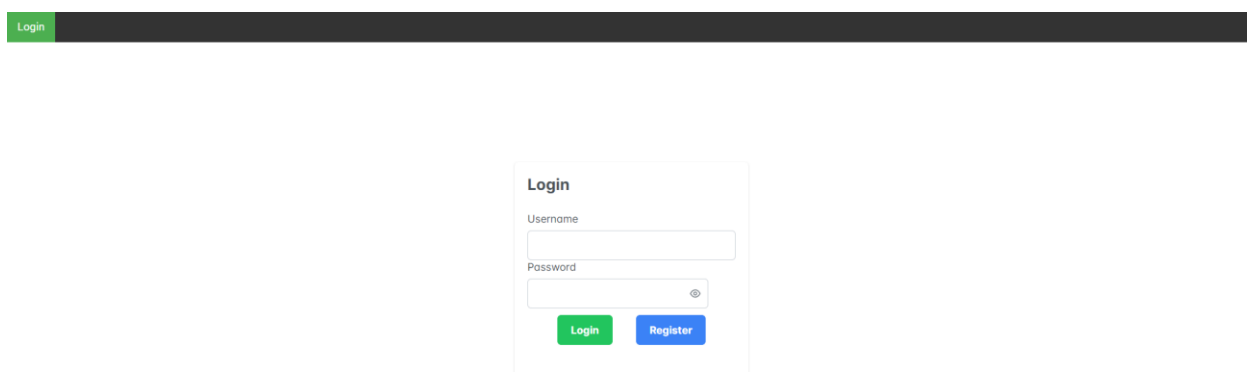


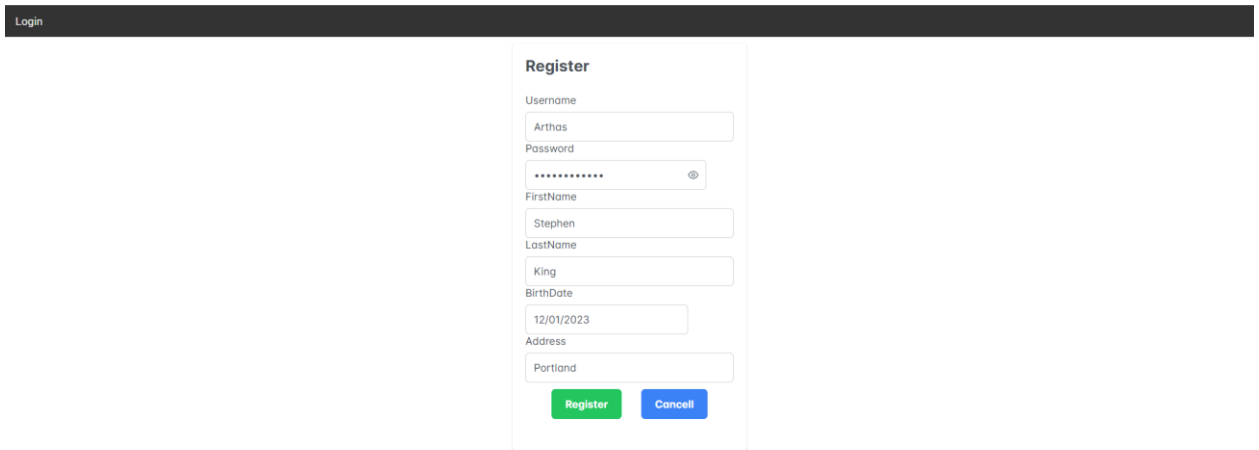
Рисунок 5.1 – Сторінка входу до ведзастосунку

На цій сторінці користувач повинен вказати свої облікові дані у відповідних полях. Користувачу необхідно зберігати логін та пароль входу до сервісу у секреті, щоб невідомі користувачі не мали доступу до внутрішніх даних.

Для неавторизованих користувачів у навігаційному меню доступний варіант тільки сторінки входу.

Ця форма валідує введені дані і сповістить користувача, якщо введені дані є некоректними.

Якщо користувач бажає створити новий обліковий запис, йому необхідно натиснути на кнопку реєстрації. Сторінка реєстрації зображена на рисунку 5.2.



The image shows a web application interface with a dark header bar containing a 'Login' link. Below the header is a 'Register' form. The form has the following fields and values:

- Username: Arthas
- Password: ***** (with an eye icon for toggling visibility)
- FirstName: Stephen
- LastName: King
- BirthDate: 12/01/2023
- Address: Portland

At the bottom of the form are two buttons: a green 'Register' button and a blue 'Cancel' button.

Рисунок 5.2 – Сторінка реєстрації

На цій сторінці користувач побачить форму реєстрації, з відповідними полями: «логін», «пароль», «ім'я», «фамілія», «день народження», «адреса».

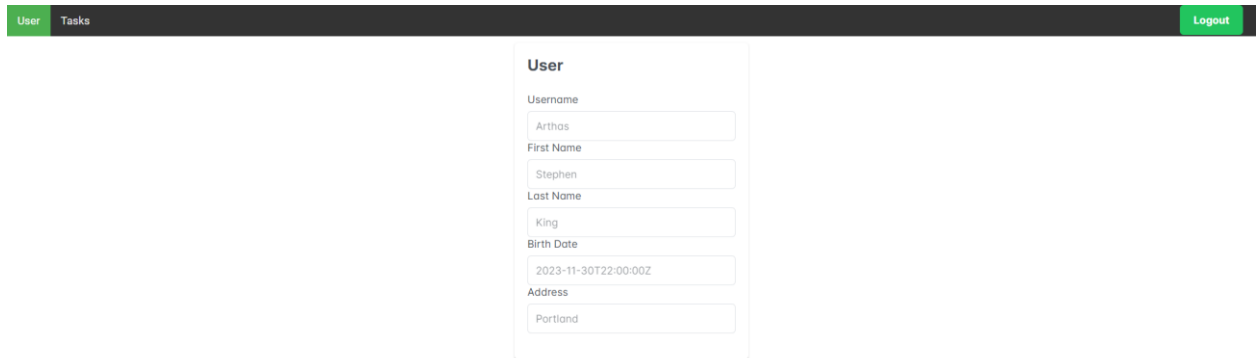
Усі поля форми мають відповідати валідаційним правилам, якщо всі правила не будуть виконані кнопка підтвердження реєстрації буде неактивна.

Поля «пароль» та «день народження», використовують компоненти з бібліотеки PrimeNG, що надає гнучкий та зручний спосіб взаємодії з користувачем, для вводу даних.

Після натискання кнопки підтвердження реєстрації, дані будуть відправлені на сервер, оброблені, та якщо вони пройдуть перевірку буде створений новий користувач.

Наступним кроком користувача буде повернуто до сторінки входу, для авторизації нового користувача в системі.

Після успішної авторизації користувача, йому буде видано токен JWT доступу, що обмежений в часі, та направлено на головну сторінку сайту (рис. 5.3).



The screenshot shows a web application interface. At the top, there is a dark navigation bar with a green 'User' button on the left and a green 'Logout' button on the right. Below the navigation bar, the main content area is white. In the center, there is a white box with a light gray border titled 'User'. Inside this box, the following information is displayed in a form-like structure:

- Username**: A text input field containing 'Arthas'.
- First Name**: A text input field containing 'Stephen'.
- Last Name**: A text input field containing 'King'.
- Birth Date**: A text input field containing '2023-11-30T22:00:00Z'.
- Address**: A text input field containing 'Portland'.

Рисунок 5.3 – Головна сторінка вебзастосунку

На цій сторінці відображено всю інформацію про активного користувача, але заборонено її змінювати чи видаляти.

Також на цій сторінці видно що в навігаційному меню з'явилися усі доступні функції.

У правому куті з'явилася кнопка виходу з облікового запису користувача.

Перша функція – це посилання на головну сторінку, що відображує інформацію про користувача. Підсвічена функція показує активну сторінку на даний момент.

Друга функція – це посилання на сторінку завдань, що відображує інформацію про завдання та надає можливість виконувати над ними операції (рис. 5.4).

User

Tasks

Logout

Tasks

Create task

Title	Description	Created Date	Estimated Hour	Deadline	Creator Id	Assignment User Id	Urgency	Urgency Rate	Actions		
string5	string5	2023-11-05T22:00:00Z	7	2023-11-06T22:00:00Z	1	1	3	87.5	Edit	Delete	Done
string6	string6	2023-11-05T22:00:00Z	6	2023-11-06T22:00:00Z	1	1	2	75	Edit	Delete	Done
string7	string7	2023-11-05T22:00:00Z	5	2023-11-06T22:00:00Z	1	1	2	62.5	Edit	Delete	Done
string4	string4	2023-11-04T22:00:00Z	3	2023-11-05T22:00:00Z	1	1	2	37.5	Edit	Delete	Done
string8	string8	2023-11-07T22:00:00Z	6	2023-11-09T22:00:00Z	1	1	7	37.5	Edit	Delete	Done
string5	string5	2023-11-04T22:00:00Z	5	2023-11-06T22:00:00Z	1	1	3	31.25	Edit	Delete	Done
string3	string3	2023-11-05T22:00:00Z	2	2023-11-29T22:00:00Z	1	1	2	0	Edit	Delete	Done

<<

<

1

2

>

>>

Рисунок 5.4 – Сторінка взаємодії з завданнями

Після переходу на сторінку, буде виконане завантаження усіх активних завдань з серверу.

На сторінці відображено активні завдання у табличній формі. Таблиця поділяється на такі стовпці як: «Заголовок», «Опис», «Час створення», «Час необхідний для виконання», «Кінцева дата виконання», «Унікальний ідентифікатор автора», «Унікальний ідентифікатор призначеного до виконання користувача», «Показник терміновості», розрахований «Рейтинг терміновості».

Кожен рядок завдання виділений своїм кольором в залежності від значення «Рейтингу терміновості». Сірий означає неважливі справи, що є не обов’язковими для виконання, та які майже не впливають на кінцеву мету. Зеленим позначені термінові справи, але не важливі для кінцевої мети, час життя таких справ обмежений тому вони мають більший пріоритет. Жовтим позначені важливі справи але не термінові у даний момент часу, вони мають більший пріоритет тому що з плином часу їх терміновість буде тільки зростати та в кінцевому етапі вони стануть позначатися червоним виділенням. Червоні термінові та важливі справи, це найважливіші та найбільш пріоритетні задачі,

вони повинні бути виконані у якомога швидкий термін, тому що мають велике значення для кінцевої мети.

Ця таблиця використовує механізм пагінації, щоб зменшити кількість одночасно завантажуваних даних та кращої продуктивності опрацювання великої кількості записів у базі даних.

Останній заголовок «Дії» визначає які дії можливо виконати над кожним завданням.

Після натискання кнопки «Видалити», завдання буде видалене з бази даних, через виконання запиту до серверної частини вебзастосунку.

Після натискання кнопки «Завершити», завдання буде видалено з таблиці, а на серверній частині вебзастосунку завдання буде позначене як виконане.

Після натискання на кнопки «Створити нове завдання» або «Редагувати завдання» буде викликано сторінку створення та редагування завдання (рис 5.5). В залежності від того яку кнопку натиснув користувач отримана сторінка буде містити дані редагованого завдання, чи не буде містити ніяких даних, для того щоб почати створювати нове завдання.

Title	Description	Created Date	Rate	Actions
string5	string5	2023-11-05T22:00:00Z		Edit Delete Done
string6	string6	2023-11-05T22:00:00Z		Edit Delete Done
string7	string7	2023-11-05T22:00:00Z		Edit Delete Done
string4	string4	2023-11-04T22:00:00Z		Edit Delete Done
string8	string8	2023-11-07T22:00:00Z		Edit Delete Done
string5	string5	2023-11-04T22:00:00Z		Edit Delete Done
string3	string3	2023-11-05T22:00:00Z		Edit Delete Done

Рисунок 5.5 – Сторінка створення та редагування

Ця сторінка представлена у вигляді модального вікна з формою. Форма повинна виконати всі валідаційні правила, щоб кнопка «Додати», стала активною. Кнопка «Скасувати» дозволить повернутись на попередню сторінку.

У формі присутні поля: «Заголовок», «Опис», «Час створення», «Час необхідний для виконання», «Кінцева дата виконання», «Унікальний ідентифікатор автора», «Унікальний ідентифікатор призначеного до виконання користувача» та «Показник терміновості».

Після введення всіх полів та натискання кнопки «Додати», дані про задачу будуть відправлені на сервер до обробки. Сервер в автоматичному режимі виконає розрахунок рейтингу терміновості для нової задачі та виконає її збереження у базу даних.

Після цих дій таблиця, на сторінці взаємодії з даними, буде перезавантажена та відобразить нову задачу відповідно до його рейтингу.

При натисканні на кнопку «Редагувати», також буде викликана сторінка створення та редагування (рис 5.6). але в ній будуть змінені назви та функції що буде виконуватись при підтвердженні відправки форми.

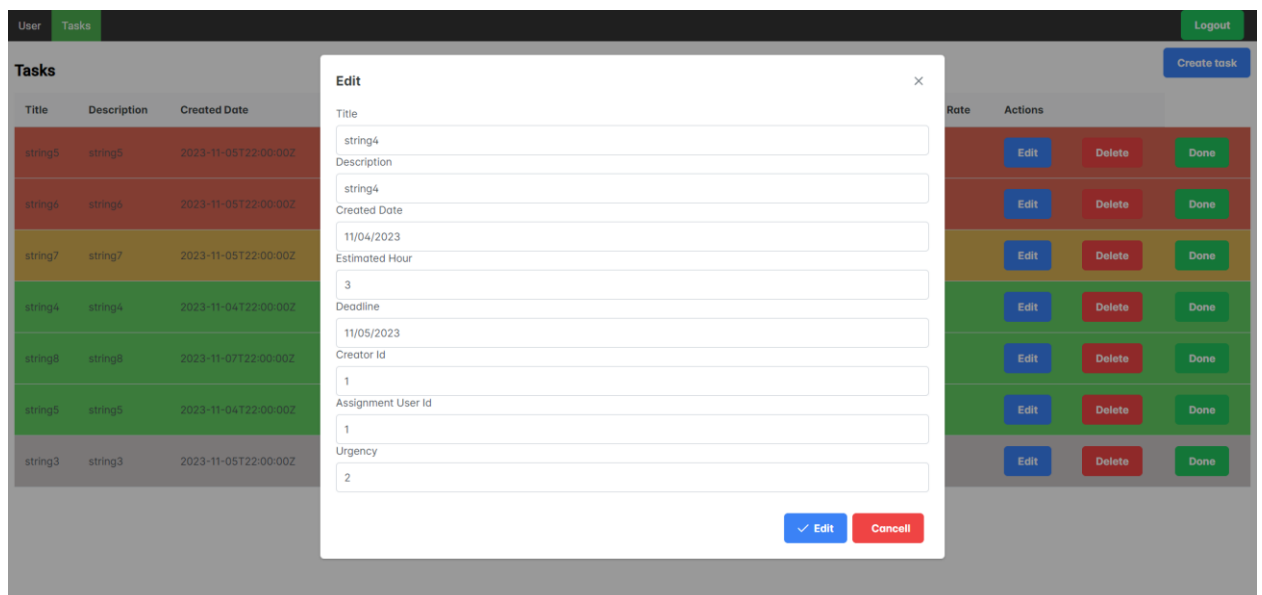


Рисунок 5.6 – Сторінка створення та редагування відкрита у режимі редагування завдання.

На цій сторінці буде представлена вся існуюча інформація про завдання, та у користувача буде можливість модифікувати ці значення.

Після натискання кнопки підтвердження «Редагувати», до серверної частини буде надіслано дані що вказані в формі, але заголовок запиту буде змінений на запит редагування, що змусить сервер перевірити наявність завдання у базі даних, та якщо запис буде знайдений відредагує його.

Кінцевим результатом буде перезавантаження таблиці відображення завдань, з оновленими даними.

5.3 Висновки за розділом 5

Було описано програмне забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів. Описано спосіб звернення до програми, та вхідні і вихідні дані.

Була розроблена поетапна інструкція з використання вебдодатку програмної реалізації методів оптимізації системи моніторингу бізнес-процесів. Наведена кожна сторінка, з описом HTML елементів на ній. Також описані функції усіх елементів з якими може взаємодіяти користувач.

Показаний процес створення нової задачі.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено методи оптимізації системи моніторингу бізнес-процесів.

Було вивчено основні поняття, напрямки і особливості сучасного впровадження систем моніторингу бізнес-процесів, а потім виконано огляд і аналіз існуючих програмних рішень, серед яких найбільшою популярністю користуються такі: програмна платформа BAS, програмна платформа SAP, програмна платформа Oracle ERP, програмна платформа Microsoft Dynamics. Найперший з яких має найбільшу популярність в Україні. Наведено порівняльну таблицю вищезазначених програмних платформ.

Для створення програмної реалізації методів оптимізації системи моніторингу бізнес-процесів було обрано такі мови програмування як C# для бекенд частини та мову TypeScript для фронтенд частини. Вони є сучасними кросплатформенними мовами широкого застосування, та надають безліч можливостей для створення різноманітних вебзастосунків. Наведена порівняльна таблиця деяких сучасних високорівневих мов програмування.

Для створення фронтенд частини програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів було використано фронтенд фреймворк Angular, що є потужним інструментом для розробки вебзастосунків, з відкритим вихідним кодом. Також для створення функціональних елементів HTML сторінки використовувалась бібліотека Angular компонентів PrimeNG.

Для створення бекенд частини була використана технологія ASP .Net Web API, що дозволяє створювати високопродуктивні веб сервери за допомогою мови програмування C#.

Для реалізації програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів були використані середовища розробки

Visual Studio Code для фронтенд частини та Microsoft Visual Studio 2022 для бекенд частини, вони підтримують багато мов програмування за допомогою інтегрованих мовних модулів та надають широкі можливості розробки та аналізу програмного коду.

У ході розробки було запропоновано метод класифікації задачі в залежності від їх пріоритету та терміновості, у якому на першому етапі користувач створює новий екземпляр задачі, заповнює всі необхідні поля, що характеризують задачу, наступним етапом буде оцінка пріоритету задачі аналітичним шляхом, наступним етапом буде вирахування показника терміновості задачі, наступним кроком необхідно розподілити задачу по матриці Ейзенхауера використовуючи раніше отримані значення пріоритету та терміновості, останнім етапом буде вирахування рейтингу важливості задачі. Результатом виконання цього методу буде оцінена та відранжована задача.

Визначено структуру та описані основні модулі, компоненти та сервіси програмної реалізації методів оптимізації системи моніторингу бізнес-процесів. Під час створення програмного продукту були визначені основні компоненти програмної реалізації методів оптимізації системи моніторингу бізнес-процесів: користувацька частина, що представляє собою Angular застосунок, та надає можливість взаємодії користувача з вебсервісом, серверна частина, що є ASP .Net Web API додатком, яка виконує обробку запитів від користувацької частини, модуль авторизації, модуль бізнес логіки, модуль взаємодії з базою даних та модуль обчислення методу класифікації задачі в залежності від її пріоритету та терміновості.

Описано схему роботи сервісу авторизації програмної реалізації методів оптимізації системи моніторингу бізнес-процесів.

Описано схему роботи сервісу управління бізнес задачами програмної реалізації методів оптимізації системи моніторингу бізнес-процесів.

Виконане проєктування бази даних вебдодатку програмної реалізації методів оптимізації системи моніторингу бізнес-процесів.

Визначено структурну схему компонентів програмної реалізації методів оптимізації системи моніторингу бізнес-процесів, що демонструє взаємодію компонентів між собою.

Створено керівництво програміста, в якому детально описано кожен компонент створеної програмної реалізації методів оптимізації системи моніторингу бізнес-процесів, наведено функціональні можливості та спосіб взаємодії.

Створено керівництво користувача, в якому поетапно описано алгоритм взаємодії з вебдодатком.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Innovation Technologies [Electronic resource]. – Access mode : <https://ukraineinvest.gov.ua/en/industries/innovation-technologies/>.
2. Global Digital 2019 reports [Electronic resource]. – Access mode : <https://wearesocial.com/blog/2019/01/digital-2019-global-internet-use-accelerates>.
3. Дослідження інтернет аудиторії України [Електрон. ресурс]. - Режим доступу: <https://inau.ua/news/oglashenydannye-yssledovanyu-ynternet-audytoryu-ukrayny-za-3-kvartal-20>.
4. David L., Rogers The Digital Transformation Playbook: Rethink Your Business for the Digital Age / L. David. – New York: Columbia University Press. – 2016. – 296 p.
5. Ukrainian Landscape Industry 4.0 Technologies [Електрон. ресурс]. - Режим доступу: <https://industry4-0-ukraine.com.ua/2019/05/27/ukrainian-landscape-industry4-0-druga-vers%d1%96ya/>.
6. Ivanov D. International Journal of Production Research / D. Ivanov, A. Dolgui. – London: Taylor and Francis Ltd. – 2019. – 57 p.
7. IMD World Digital Competitiveness Ranking [Electronic resource]. – Access mode : <https://www.imd.org/centers/wcc/world-competitiveness-center/rankings/world-digital-competitiveness-ranking/>.
8. The World Economic Forum. The Global Competitiveness Report Internet Asotsiatsiia Ukrainy [Electronic resource]. – Access mode : <http://www3.weforum.org/docs/GCR2018/05FullReport/TheGlobalCompetitivenessReport2018.pdf>.
9. Розпорядження Кабінету Міністрів України «Про схвалення Концепції розвитку цифрової економіки та суспільства України на 2018-2020 роки та затвердження плану заходів щодо її реалізації» від 17 січня 2020 р. № 67-2018р. [Електрон. ресурс]. - Режим доступу: <https://zakon.rada.gov.ua/laws/show/67-2018-%D1%80>.

10. Цифрова адженда України – 2020 [Електрон. ресурс]. - Режим доступу: <https://uccr.org.ua/uploads/files/58e78ee3c3922.pdf>.
11. Навіщо потрібні дорожні карти цифрової трансформації [Електрон. ресурс]. - Режим доступу: <https://industry4-0-ukraine.com.ua/2019/03/19/why-we-need-digital-transformation-roadmap-in-manufacturing/>.
12. Rosing M. The Complete Business Process Handbook: Body of Knowledge from Process Modeling to BPM / M. Rosing, H. von Scheel, A.W. Scheer. – Vol. 1 – Burlington: Morgan Kaufmann. – 2014. – 703 p.
13. Doyle K. IBM Solution Approach For Enterprise Asset Management / K. Doyle, J. Saulman, B. Cary. – Armonk: IBM. – 2017. – 146 p.
14. RISE with SAP for Modular Cloud ERP: A New Way of Working [Electronic source]. – Access mode: <https://news.sap.com/2021/06/sapphire-now-rise-with-sap-for-modular-cloud-erp/>.
15. Головна сторінка сайту програмного продукту BAS [Електронний ресурс]. – Режим доступу: <https://www.bas-soft.eu/>.
16. Головна сторінка сайту програмного продукту SAP [Електронний ресурс]. – Режим доступу: <https://www.sap.com/ukraine/index.html>.
17. Головна сторінка сайту програмного продукту Oracle [Електронний ресурс]. – Режим доступу: <https://www.oracle.com/ua/erp/>.
18. Головна сторінка сайту програмного продукту Microsoft Dynamics [Електронний ресурс]. – Режим доступу: <https://www.microsoft.com/ru-ru/dynamics-365>.
19. Bright P. Microsoft TypeScript: the JavaScript we need, or a solution looking for a problem? / P. Bright. – Boston: Ars Technica. – 2012. – 110 p.
20. TypeScript Programming with Visual Studio Code [Electronic source]. – Access mode: <https://code.visualstudio.com/docs/languages/typescript>.
21. Definitely Typed [Electronic source]. – Access mode: <https://github.com/DefinitelyTyped/DefinitelyTyped>.

22. Jackson J. Microsoft Augments Javascript for Large-scale Development / J. Jackson. – Needham: IDG Enterprise. – 2012. – 5 p.
23. What is TypeScript and why with Anders Hejlsberg [Electronic source]. – Access mode: <https://learn.microsoft.com/en-us/shows/web-wednesday/what-is-typescript>.
24. Development page of the React project on the GitHub platform [Electronic source]. – Access mode: <https://github.com/facebook/react>.
25. Angular vs. React: What to Choose for Your Web App? [Electronic source]. – Access mode: <https://www.cleveroad.com/blog/react-vs-angular-ultimate-performance-research-2017>.
26. Головна сторінка сайту фреймворку Vue.js [Електронний ресурс]. – Режим доступу: <https://ua.vuejs.org/guide/introduction.html>.
27. What is Vue.js? [Electronic source]. – Access mode: https://www.w3schools.com/whatis/whatis_vue.asp.
28. What's the difference between AngularJS and Angular? [Electronic source]. – Access mode: <https://gorrion.io/blog/angularjs-vs-angular/>.
29. Home page of the Angular framework website [Electronic source]. – Access mode: <https://angular.io/guide/architecture>.
30. Home page of the PrimeNG library website [Electronic source]. – Access mode: <https://primeng.org/>.
31. Visual Studio Product Lifecycle and Servicing [Electronic source]. – Access mode: <https://docs.microsoft.com/en-us/visualstudio/productinfo/vs-servicing>.
32. VSPackages and Managed Package Framework (MPF) – Microsoft [Electronic source]. – Access mode: [https://msdn.microsoft.com/en-us/library/bb166554\(VS.80\).aspx](https://msdn.microsoft.com/en-us/library/bb166554(VS.80).aspx).
33. Managed Language Services overview – Microsoft [Electronic source]. – Access mode: [https://msdn.microsoft.com/en-us/library/bb166360\(VS.80\).aspx](https://msdn.microsoft.com/en-us/library/bb166360(VS.80).aspx).

34. Home page of the Microsoft Visual Studio Code IDE site – [Electronic source]. – Access mode: <https://code.visualstudio.com/>.

35. The Eisenhower Matrix: Prioritize Your Time on What Matters Most – [Electronic source]. – Access mode: <https://slab.com/blog/eisenhower-matrix/>.

ДОДАТОК А
Технічне завдання

Вступ

Програмне забезпечення може використовуватися для оптимізації методів системи моніторингу бізнес-процесів.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Дослідження та програмна реалізація методів оптимізації системи моніторингу бізнес-процесів», затверджене наказом Національного університету «Запорізька політехніка» № 437 від 14 листопада 2023 р.

A.2 Призначення розробки

Програмний продукт призначений для забезпечення процесу оптимізації методів системи моніторингу бізнес-процесів.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Програмне забезпечення дослідження та оптимізації методів системи моніторингу бізнес-процесів забезпечує виконання таких функцій:

- можливість створення персонально облікового запису з налаштованими правами доступу;
- можливість генерації та налаштування задач;
- можливість призначення задачі обраному співробітнику;
- пошук необхідного співробітника у базі даних;
- можливість генерації звіту виконаних завдань;

- можливість гнучкого підходу до взаємодії зі створеними бізнес-процесами.

А.3.2 Вимоги до інтерфейсу програми

Інтерфейс програмного продукту реалізації методів оптимізації системи моніторингу бізнес-процесів повинен бути доступним на всіх сучасних інформаційних пристроях, мати зручний та зрозумілий дизайн для користувачів.

А.3.3 Вимоги до надійності

Програмне забезпечення дослідження та оптимізації методів системи моніторингу бізнес-процесів повинно забезпечити надійне та постійне стабільне функціонування.

А.3.4 Умови експлуатації

Для експлуатації програмного забезпечення дослідження та оптимізації методів системи моніторингу бізнес-процесів у користувач повинен мати пристрій під керуванням сучасної операційної системи, з можливістю взаємодії з мережею інтернет.

A.3.5 Вимоги до складу та параметрів технічний засобів

Для зручної експлуатації програмного забезпечення дослідження та оптимізації методів системи моніторингу бізнес-процесів необхідно мати таке або краще апаратне та програмне забезпечення:

- оперативна пам'ять – не менше 2Гб;
- операційна система з можливістю взаємодії через мережу інтернет;
- браузер з підтримкою виконання JavaScript коду;
- графічна карта з підтримкою DirectX 10 або краще;
- процесор – 1.8 ГГц або швидше;
- порт Ethernet або мережевий адаптер.

A.3.6 Вимоги до маркування і пакування

Програмне забезпечення дослідження та оптимізації методів системи моніторингу бізнес-процесів може бути розповсюджено як цифрова копія на будь якому носії інформації. На пакуванні повинна бути назва програми – «Програмне забезпечення дослідження та оптимізації методів системи моніторингу бізнес-процесів».

A.4 Вхідні дані до роботи

Вхідними даними до вебзастосунку програмної реалізації методів оптимізації системи моніторингу бізнес-процесів є HTTPS запит який в собі може містити дані, над якими потрібно виконати дії, з вказаним типом запиту у заголовку.

Вихідними даними програмного продукту реалізації методів оптимізації системи моніторингу бізнес-процесів є згенерована HTML сторінка з CSS стилями, що відповідає на HTTPS запит користувача.

ДОДАТОК Б
Фрагмент тексту програми

Файл Program.sc

```

using System.Text;
using Dipl.BProces.Api.Services;
using Dipl.BProces.Api.Services.Interfaces;
using Dipl.BProces.Business.BusinessLogic;
using Dipl.BProces.Common.WebApi.Handlers;
using Dipl.BProces.DataAccess;
using Dipl.BProces.DataAccess.DbOptions;
using Dipl.BProces.DataAccess.Entities;
using Microsoft.AspNetCore.Authentication.JwtBearer;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Identity;
using Microsoft.EntityFrameworkCore;
using Microsoft.IdentityModel.Tokens;
using Microsoft.OpenApi.Models;

namespace Dipl.BProces.Api
{
    public class Program
    {
        public static void Main(string[] args)
        {
            var builder = WebApplication.CreateBuilder(args);

            builder.Services.Configure<JWTOptions>(builder.Configuration.GetSection(JWT
Options.Jwt));
            builder.Services.AddScoped<IAuthService, AuthService>();
            builder.Services.AddScoped<IUserService, UserService>();
            builder.Services.AddScoped<IBpTaskServices, BpTaskServices>();

            ConnectionStringsOptions? stringsOptions = builder.Configuration
.GetSection(ConnectionStringsOptions.ConnectionStrings).Get<ConnectionStrings
Options>();

            builder.Services.AddDbContext<BProcDataContext>(opt =>
opt.UseNpgsql(stringsOptions?.DefaultPostgreDB));

            builder.Services.AddCors(options =>
{

```

```

options.AddDefaultPolicy(build =>
{
    build.AllowAnyOrigin()
        .AllowAnyMethod()
        .AllowAnyHeader();
});

});

JWTOptions? jwt =
builder.Configuration.GetSection(JWTOptions.Jwt).Get<JWTOptions>();
builder.Services.AddAuthentication(opt =>
{
    opt.DefaultAuthenticateScheme =
JwtBearerDefaults.AuthenticationScheme;
    opt.DefaultChallengeScheme =
JwtBearerDefaults.AuthenticationScheme;
})
.AddJwtBearer(opt =>
{
    opt.TokenValidationParameters = new TokenValidationParameters
    {
        ValidateIssuer = false,
        ValidateAudience = false,
        ValidateLifetime = true,
        ValidateIssuerSigningKey = true,
        ValidIssuer = jwt?.Issuer,
        ValidAudience = jwt?.Audience,
        IssuerSigningKey = new
SymmetricSecurityKey(Encoding.UTF8.GetBytes(jwt.Secret))
    };
});

builder.Services.AddAuthorization(opt =>
opt.DefaultPolicy = new
AuthorizationPolicyBuilder(JwtBearerDefaults.AuthenticationScheme)
.RequireAuthenticatedUser()
.Build());

builder.Services.Configure<IdentityOptions>(options =>
{
    options.Password.RequireDigit = false;
    options.Password.RequireNonAlphanumeric = false;
    options.Password.RequiredLength = 8;

```

```

        options.Password.RequireUppercase = false;
    });

    builder.Services.AddIdentity<AppUser, IdentityRole<long>>()
        .AddEntityFrameworkStores<BProcDataContext>()
        .AddUserManager<UserManager<AppUser>>()
        .AddSignInManager<SignInManager<AppUser>>();

    // Add services to the container.

    builder.Services.AddControllers(options =>
    {
        options.Filters.Add(typeof(OperationCancelledExceptionFilter));
    });
    builder.Services.AddEndpointsApiExplorer();
    builder.Services.AddSwaggerGen(options =>
    {
        options.AddSecurityDefinition("Bearer", new
OpenApiSecurityScheme()
        {
            Description =
                "JWT Authorization header using the Bearer scheme. \r\n\r\n" +
                "Enter 'Bearer' [space] and then your token in the text input below.
\r\n\r\n",
            Name = "Authorization",
            In = ParameterLocation.Header,
            Scheme = "Bearer"
        });
        options.AddSecurityRequirement(new OpenApiSecurityRequirement()
        {
            {
                new OpenApiSecurityScheme
                {
                    Reference = new OpenApiReference
                    {
                        Type = ReferenceType.SecurityScheme,
                        Id = "Bearer"
                    },
                    Scheme = "oauth2",
                    Name = "Bearer",
                    In = ParameterLocation.Header
                },
                new List<string>()

```

```

        }
    });
});

var app = builder.Build();

// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}

app.UseMiddleware<ExceptionHandlerMiddleware>();

app.UseHttpsRedirection();
app.UseAuthentication();
app.UseAuthorization();

app.UseCors(build =>
    build.AllowAnyOrigin()
        .AllowAnyMethod()
        .AllowAnyHeader());

app.MapControllers();

app.Run();
}
}
}

```

Файл AuthController.cs

```

using Dipl.BProces.Api.Services.Interfaces;
using Dipl.BProces.DataAccess;
using Dipl.BProces.DataAccess.Entities;
using Dipl.BProces.Models;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Mvc;

namespace Dipl.BProces.Api.Auth.Controllers
{

```

```

[Route("api/[controller]")]
[ApiController]
public class AuthController : ControllerBase
{
    private readonly UserManager<AppUser> _userManager;
    private readonly BProcDataContext _context;
    private readonly IConfiguration _configuration;
    private readonly IAuthService _authService;

    public AuthController(BProcDataContext context, UserManager<AppUser>
userManager, IConfiguration configuration, IAuthService authService)
    {
        _context = context;
        _userManager = userManager;
        _configuration = configuration;
        _authService = authService;
    }

    [HttpPost("login")]
    public async Task<ActionResult> Login([FromBody] AuthRequest request)
    {
        AuthResponce loggedUser = await _authService.Login(request);

        if (loggedUser.Token == null)
        {
            return BadRequest("Wrong credentials");
        }

        return Ok(loggedUser);
    }

    [HttpPost("register")]
    public async Task<ActionResult> Register([FromBody] RegisterRequest
request)
    {
        if (!ModelState.IsValid) return BadRequest(request);

        bool registerResult = await _authService.Register(request);
        if (!registerResult)
        {
            return BadRequest("User already exist");
        }

        return Ok();
    }
}

```

```

    }
}
}

```

Файл UserController.cs

```

using Dipl.BProces.Api.Services.Interfaces;
using Dipl.BProces.DataAccess;
using Dipl.BProces.Models;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;

namespace Dipl.BProces.Api.Auth.Controllers
{
    [Authorize]
    [Route("api/Auth/[controller]")]
    [ApiController]
    public class UserController : ControllerBase
    {
        private readonly BProcDataContext _context;
        private readonly IUserService _userService;

        public UserController(BProcDataContext context, IUserService userService)
        {
            _context = context;
            _userService = userService;
        }

        [HttpGet]
        public async Task<ActionResult> GetCurrentUser()
        {
            UserDTO currentUserDto = await
            _userService.GetCurrentUser(User.Identity.Name);

            return Ok(currentUserDto);
        }

        [HttpGet("{id}")]
        public async Task<ActionResult> GetUser(long id)
        {
            UserDTO currentUserDto = await _userService.GetUser(id);
            if (currentUserDto.Id == 0)
            {
                return BadRequest();
            }
        }
    }
}

```

```

        return Ok(currentUserDto);
    }
}
}

```

Файл BpTasksController.cs

```

using Dipl.BProces.Business.BusinessLogic;
using Dipl.BProces.Common.WebApi.Models;
using Dipl.BProces.DataAccess;
using Dipl.BProces.Models;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;

namespace Dipl.BProces.Api.Controllers.BpTasksManager
{
    [Authorize]
    [Route("api/[controller]")]
    [ApiController]
    public class BpTasksController : ControllerBase
    {
        private IBpTaskServices BpTasksService { get; set; }

        public BpTasksController(BProcDataContext context, IBpTaskServices
bpTaskServices)
        {
            BpTasksService = bpTaskServices;
        }

        [HttpGet("")]
        public async
Task<ActionResult<ResponseWrapper<PagedResponse<List<BpTaskDto>>>>>
GetTasks(Cancellation token, int skip = 0, int take = 10)
        {
            return await BpTasksService.GetTasks(skip, take, token);
        }

        [HttpGet("{id}")]
        public async Task<ActionResult<ResponseWrapper<BpTaskDto>>>
GetTask(int id, Cancellation token)
        {
            return await BpTasksService.GetTask(id, token);
        }
    }
}

```

```

[HttpPut("{id}")]
public async Task<ActionResult<ResponseWrapper<BpTaskDto>>>
PutTask(int id, BpTaskDto taskDto, CancellationToken token)
{
    return await BpTasksService.PutTask(id, taskDto, token);
}

[HttpPost]
public async Task<ActionResult<ResponseWrapper<BpTaskDto>>>
PostTask(BpTaskDto taskDto, CancellationToken token)
{
    return await BpTasksService.PostTask(taskDto, token);
}

[HttpDelete("{id}")]
public async Task<ActionResult<ResponseWrapper<BpTaskDto>>>
DeleteTask(int id, CancellationToken token)
{
    return await BpTasksService.DeleteTask(id, token);
}
}

```

Файл IAuthService.cs

```

using Dipl.BProces.Models;

namespace Dipl.BProces.Api.Services.Interfaces
{
    public interface IAuthService
    {
        public Task<AuthResponse> Login(AuthRequest request);

        public Task<bool> Register(RegisterRequest request);
    }
}

```

Файл AuthService.cs

```

using System.IdentityModel.Tokens.Jwt;
using System.Security.Claims;
using System.Text;
using Dipl.BProces.Api.Services.Interfaces;

```

```

using Dipl.BProces.Business.Validators;
using Dipl.BProces.DataAccess;
using Dipl.BProces.DataAccess.Entities;
using Dipl.BProces.Models;
using Mapster;
using Microsoft.AspNetCore.Identity;
using Microsoft.EntityFrameworkCore;
using Microsoft.IdentityModel.Tokens;

namespace Dipl.BProces.Api.Services
{
    public class AuthService : IAuthService
    {
        private readonly UserManager<AppUser> _userManager;
        private readonly BProcDataContext _context;
        private readonly IConfiguration _configuration;

        public AuthService(BProcDataContext context, UserManager<AppUser>
userManager, IConfiguration configuration)
        {
            _context = context;
            _userManager = userManager;
            _configuration = configuration;
        }

        public async Task<AuthResponse> Login(AuthRequest request)
        {
            if (request == null)
            {
                return new AuthResponse();
            }

            LoginValidator validator = new();
            var validationResult = validator.Validate(request);

            if (!validationResult.IsValid)
            {
                return new AuthResponse();
            }

            var managedUser = await
_userManager.FindByNameAsync(request.UserName);

            if (managedUser == null)

```

```

    {
        return new AuthResponce();
    }

    bool isPasswordValid = await
        _userManager.CheckPasswordAsync(managedUser, request.Password);

    if (!isPasswordValid)
    {
        return new AuthResponce();
    }

    var userRoles = await _userManager.GetRolesAsync(managedUser);

    var claims = new List<Claim>
    {
        new Claim(ClaimTypes.Name, managedUser.UserName),
        new Claim(JwtRegisteredClaimNames.Jti, Guid.NewGuid().ToString()),
    };

    claims.AddRange(userRoles.Select(claim => new
        Claim(ClaimTypes.Name, claim)));
    string token = GenerateToken(claims);
    AuthResponce authResponce = managedUser.Adapt<AuthResponce>();
    authResponce.Token = token;

    return authResponce;
}

public async Task<bool> Register(RegisterRequest request)
{
    if (request == null)
    {
        return false;
    }

    RegisterValidator validator = new();
    var validationResult = await validator.ValidateAsync(request);
    if (!validationResult.IsValid)
    {
        return false;
    }

    var userExist = await
        _userManager.FindByNameAsync(request.UserName);

```

```

        if (userExist != null)
            return false;
        AppUser user = request.Adapt<AppUser>();
        var createUserResult = await _userManager.CreateAsync(user,
request.Password);
        if (!createUserResult.Succeeded)
            throw new Exception($"User {request.UserName} not
{createUserResult.Errors}");

        var findUser = await _context.Users.FirstOrDefaultAsync(x =>
x.UserName == request.UserName);

        return findUser == null ? throw new Exception($"User
{request.UserName} not found") : true;
    }

    private string GenerateToken(IEnumerable<Claim> claims)
    {
        JWTOptions? jwt =
_configuration.GetSection(JWTOptions.Jwt).Get<JWTOptions>();
        var authSigningKey = new
SymmetricSecurityKey(Encoding.UTF8.GetBytes(jwt.Secret));
        var tokenDescriptor = new SecurityTokenDescriptor
        {
            Issuer = jwt.Issuer,
            Audience = jwt.Audience,
            Expires = DateTime.UtcNow.AddHours(5),
            SigningCredentials = new SigningCredentials(authSigningKey,
SecurityAlgorithms.HmacSha256),
            Subject = new ClaimsIdentity(claims)
        };

        var tokenHandler = new JwtSecurityTokenHandler();
        var token = tokenHandler.CreateToken(tokenDescriptor);
        return tokenHandler.WriteToken(token);
    }
}
}

```

Файл BpTaskServices.cs

```

using Dipl.BProces.Business.Validators;
using Dipl.BProces.Common.WebApi.Models;
using Dipl.BProces.DataAccess.Entities;

```

```

using Dipl.BProces.DataAccess;
using Dipl.BProces.Models;
using Mapster;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Dipl.BProces.Business.BusinessLogic
{
    public interface IBpTaskServices
    {
        public Task<ResponseWrapper<PagedResponse<List<BpTaskDto>>>>
GetTasks(int skip, int take,
        CancellationToken token);

        public Task<ResponseWrapper<BpTaskDto>> GetTask(int id,
CancellationToken token);
        public Task<ResponseWrapper<BpTaskDto>> PutTask(int id, BpTaskDto
taskDto, CancellationToken token);

        public Task<ResponseWrapper<BpTaskDto>> PostTask(BpTaskDto?
taskDto, CancellationToken token);
        public Task<ResponseWrapper<BpTaskDto>> DeleteTask(int id,
CancellationToken token);
    }

    public class BpTaskServices : IBpTaskServices
    {
        private BProcDataContext Context { get; set; }

        public BpTaskServices(BProcDataContext dbContext)
        {
            Context = dbContext;
        }

        public async
Task<ResponseWrapper<PagedResponse<List<BpTaskDto>>>> GetTasks(int
skip, int take, CancellationToken token)
        {
            var count = Context.Tasks.Count();

```

```

        var data = Context.Tasks.Include(c => c.Creator).Include(u =>
u.AssignmentUser).OrderByDescending(x=>x.UrgencyRate).AsQueryable().Skip(
skip).Take(take).Adapt<List<BpTaskDto>>()); ;
        PagedResponse<List<BpTaskDto>> pagedResult = new(count, data);
        if (!pagedResult.Items.Any())
        {
            return new
ResponseWrapper<PagedResponse<List<BpTaskDto>>>(errors: new
List<string>() { "Collection is empty" });
        }
        return
ResponseWrapper<PagedResponse<List<BpTaskDto>>>.WrapToResponse(paged
Result);
    }

```

```

        public async Task<ResponseWrapper<BpTaskDto>> GetTask(int id,
Cancellation token)
        {
            ResponseWrapper<BpTaskDto> response = new(errors: new
List<string>());
            var result = await Context.Tasks.FirstOrDefaultAsync(a => a.Id == id,
token);
            if (result == null)
            {
                response?.Errors.Add("Not found");
                return response;
            }
            return
ResponseWrapper<BpTaskDto>.WrapToResponse(result.Adapt<BpTaskDto>());
        }

```

```

        public async Task<ResponseWrapper<BpTaskDto>> PutTask(int id,
BpTaskDto taskDto, Cancellation token)
        {
            ResponseWrapper<BpTaskDto> response = new(errors: new
List<string>());
            if (id != taskDto.Id)
            {
                response.Errors?.Add("Bad request!");
                return response;
            }

            var bpTask = taskDto.Adapt<BpTask>();
            BpTaskValidator validator = new BpTaskValidator();

```

```

        var validationResult = validator.Validate(bpTask);
        response.Errors?.AddRange(validationResult.Errors.Select(e =>
e.ErrorMessage).ToList());
        if (!validationResult.IsValid) return response;
        if (!TaskExists(id))
        {
            response.Errors?.Add("Object not exist");
            return response;
        }
        if (!UserExist(taskDto.CreatorId))
        {
            response.Errors?.Add("Author not exist");
            return response;
        }

        Context.Entry(bpTask).State = EntityState.Modified;

        try
        {
            await Context.SaveChangesAsync(token);
        }
        catch (DbUpdateConcurrencyException ex)
        {
            response.Errors?.Add(ex.Message);
            return response;
        }

        return ResponseWrapper<BpTaskDto>.WrapToResponse(taskDto);
    }

    public async Task<ResponseWrapper<BpTaskDto>> PostTask(BpTaskDto?
taskDto, CancellationToken token)
    {
        ResponseWrapper<BpTaskDto> response = new(errors: new
List<string>());
        if (taskDto == null)
        {
            response.Errors?.Add("Bad request");
            return response;
        }

        var bpTask = taskDto.Adapt<BpTask>();

        bpTask.UrgencyRate = RateCalculate(bpTask);
    }

```

```

        BpTaskValidator validator = new BpTaskValidator();
        var validationResult = validator.Validate(bpTask);
        response.Errors?.AddRange(validationResult.Errors.Select(e =>
e.ErrorMessage).ToList());
        if (!validationResult.IsValid) return response;

        if (!UserExist(taskDto.CreatorId))
        {
            response.Errors?.Add("Author not exist");
            return response;
        }

        Context.Tasks.Add(bpTask);
        try
        {
            await Context.SaveChangesAsync(token);
        }
        catch (DbUpdateConcurrencyException ex)
        {
            response.Errors?.Add(ex.Message);
            return response;
        }

        return ResponseWrapper<BpTaskDto>.WrapToResponse(taskDto); ;
    }

    public async Task<ResponseWrapper<BpTaskDto>> DeleteTask(int id,
Cancellation token)
    {
        ResponseWrapper<BpTaskDto> response = new(errors: new
List<string>());
        var bpTask = await Context.Tasks.FindAsync(id, token);
        if (bpTask == null)
        {
            response.Errors?.Add("Not found");
            return response;
        }

        Context.Tasks.Remove(bpTask);
        try
        {
            await Context.SaveChangesAsync(token);

```

```

    }
    catch (DbUpdateConcurrencyException ex)
    {
        response.Errors?.Add(ex.Message);
        return response;
    }

    return ResponseWrapper<BpTaskDto>.WrapToResponse(new
BpTaskDto()); ;
}

private bool TaskExists(int id)
{
    return (Context.Tasks?.Any(e => e.Id == id)).GetValueOrDefault();
}

private bool UserExist(int id)
{
    return (Context.Users?.Any(e => e.Id == id)).GetValueOrDefault();
}

private double RateCalculate(BpTask task)
{
    double rate = 0;
    var deltaHours = task.DeadLine.Subtract(task.CreatedDate).Days * 8;
    rate = task.EstimatedHour / (double)deltaHours;
    return rate * task.Urgency;
}
}
}

```

Файл **bptask.component.ts**

```

import { Component, OnInit } from "@angular/core";
import { FormBuilder, FormGroup, Validators } from "@angular/forms";
import { Router } from "@angular/router";
import { TablePageEvent } from "primeng/table";
import { BpTask } from "src/app/models/bpTask";
import { TasksService } from "src/app/services/tasks.service";

```

```

@Component({
  selector: "app-books",
  templateUrl: "./bptask.component.html",
  styleUrls: ["./bptask.component.scss"],
})
export class BpTaskComponent implements OnInit {
  Tasks: BpTask[] = [];
  editedBpTask: BpTask | null = null;
  displayAddModal: boolean = false;
  editForm: FormGroup;
  totalRecords: number = 20;
  rows: number = 10;
  page: number = 0;

  constructor(private router: Router, private bpTaskService: TaskService, private
fb: FormBuilder) {
    this.editForm = this.fb.group({
      title: ["", Validators.required],
      description: ["", Validators.required],
      createdAt: ["", Validators.required],
      estimatedHour: [Number, Validators.required],
      deadLine: ["", Validators.required],
      creatorId: ["", Validators.required],
      assignmentUserId: ["", Validators.required],
      urgency: [Number, Validators.required],
    });
  }
}

```

```
ngOnInit(): void {
```

```
    this.getAllBpTasks(0, this.rows);
```

```
}
```

```
getAllBpTasks(skip: number, take: number) {
```

```
    this.bpTaskService.getAllBpTasks(skip, take).subscribe((us) => {
```

```
        this.Tasks = us.data.items.sort((a, b) => b.urgencyRate - a.urgencyRate);
```

```
    });
```

```
}
```

```
deleteBpTask(id: number) {
```

```
    this.bpTaskService.deleteBpTask(id).subscribe((ans) => {
```

```
        this.refresh();
```

```
    });
```

```
}
```

```
showAddModal() {
```

```
    this.displayAddModal = true;
```

```
    this.editedBpTask = null;
```

```
}
```

```
hideAddModal(isClosed: boolean) {
```

```
    this.displayAddModal = !isClosed;
```

```
}
```

```
showEditModal(book: BpTask) {
```

```
    this.displayAddModal = true;
```

```
    this.editedBpTask = book;
```

```
}
```

```

refresh() {
    this.getAllBpTasks(this.page * this.rows, this.rows);
}

onPageChange(event: TablePageEvent) {
    this.page = event.first;
    this.rows = event.rows;

    this.bpTaskService.getAllBpTasks(this.page, this.rows).subscribe((us) => {
        this.Tasks = us.data.items;
        this.totalRecords = us.data.total;
    });
}

setUrgencyColor(task: BpTask) {
    if (task.urgencyRate > 70) return "#d96653";
    else if (task.urgencyRate > 40) return "#d9b053";
    else if (task.urgencyRate > 20) return "#62cd62";
    else return "#cbc7c7";
}
}

```

Файл `bptask.component.html`

```

<div class="data-container">
    <div class="flex justify-content-between">
        <h2>Tasks</h2>
        <p-button label="Create task" (click)="showAddModal()"></p-button>
    </div>

```

```

<p-table
  [lazy]="true"
  [paginator]="true"
  (onPage)="onPageChange($event)"
  [first]="page"
  [rows]="rows"
  [totalRecords]="totalRecords"
  [value]="Tasks"
  [tableStyle]="{ 'min-width': '50rem' }"
>

<ng-template pTemplate="header">
  <tr>
    <th>Title</th>
    <th>Description</th>
    <th>Created Date</th>
    <th>Estimated Hour</th>
    <th>Deadline</th>
    <th>Creator Id</th>
    <th>Assignment User Id</th>
    <th>Urgency</th>
    <th>Urgency Rate</th>
    <th colspan="2">Actions</th>
  </tr>
</ng-template>

<ng-template pTemplate="body" let-bTask>
  <div></div>
  <tr [ngStyle]="{ 'background-color': setUrgencyColor(bTask) }">
    <td>{{ bTask.title }}</td>
    <td>{{ bTask.description }}</td>

```

```

<td>{{ bTask.createdDate }}</td>
<td>{{ bTask.estimatedHour }}</td>
<td>{{ bTask.deadLine }}</td>
<td>{{ bTask.creatorId }}</td>
<td>{{ bTask.assignmentUserId }}</td>
<td>{{ bTask.urgency }}</td>
<td>{{ bTask.urgencyRate | number : "1.0-3" }}</td>
<td>
    <p-button label="Edit" (click)="showEditModal(bTask)"> </p-button>
</td>
<td>
    <p-button      label="Delete"      (click)="deleteBpTask(bTask.id)"
severity="danger"></p-button>
</td>
<td>
    <p-button      label="Done"      (click)="deleteBpTask(bTask.id)"
severity="success"></p-button>
</td>
</tr>
</ng-template>
</p-table>

<app-bptask-form      [dispalyAddModal]="dispalyAddModal"
[editedBpTask]="editedBpTask" (clickClose)="hideAddModal($event)">
</app-bptask-form>
</div>

```

Файл bptask-form.component.ts

```
import { Component, EventEmitter, Input, OnChanges, OnInit, Output,
SimpleChanges } from "@angular/core";
import { FormBuilder, FormGroup, Validators } from "@angular/forms";
import { Router } from "@angular/router";
import { BpTask } from "src/app/models/bpTask";
import { TasksService } from "src/app/services/tasks.service";
```

```
@Component({
  selector: "app-bptask-form",
  templateUrl: "./bptask-form.component.html",
  styleUrls: ["./bptask-form.component.scss"],
})
```

```
export class BpTaskFormComponent implements OnChanges {
  @Input() displayAddModal: boolean = true;
  @Input() editedBpTask: BpTask | null = null;
  @Output() clickClose: EventEmitter<boolean> = new EventEmitter<boolean>();
  modalType = "Add";

  currentTask: BpTask | null = null;
  editForm: FormGroup;
  edit: boolean = false;
  constructor(private fb: FormBuilder, private bpTaskserv: TasksService, private
router: Router) {
    this.editForm = this.fb.group({
      title: ["", Validators.required],
      description: ["", Validators.required],
      createDate: ["", Validators.required],
      estimatedHour: ["", Validators.required],
```

```

        deadLine: [ "", Validators.required ],
        creatorId: [ "", Validators.required ],
        assignmentUserId: [ "", Validators.required ],
        urgency: [ Number, Validators.required ],
    });
}

ngOnChanges(changes: SimpleChanges): void {
    if (this.editedBpTask) {
        this.modalType = "Edit";
        this.editForm.patchValue(this.editedBpTask);
    } else {
        this.modalType = "Add";
        this.editForm.reset();
    }
}

closeModal() {
    this.clickClose.emit(true);
}

addEditBook() {
    const taskData = this.editForm.value;
    if (this.editForm.valid) {
        this.currentTask = new BpTask(
            0,
            taskData.title,
            taskData.description,
            taskData.createdDate,
            taskData.estimatedHour,

```

```

        taskData.deadLine,
        taskData.creatorId,
        taskData.assignmentUserId,
        taskData.urgency,
        0,
    );
    if (this.editedBpTask == null) {
        this.bpTaskserv.createBpTask(this.currentTask).subscribe((respon) => {
            this.refresh();
        });
    } else {
        this.currentTask.id = this.editedBpTask.id;
        this.bpTaskserv.updateBpTask(this.editedBpTask.id,
this.currentTask).subscribe((respon) => {
            console.log(respon.errors);
            this.refresh();
        });
        this.editedBpTask = null;
    }
}

this.editForm.reset();
this.closeModal();
}

refresh() {
    window.location.reload();
}
}

```

Файл app.module.ts

```

import { NgModule } from "@angular/core";
import { BrowserModule } from "@angular/platform-browser";
import { HTTP_INTERCEPTORS, HttpClient, HttpClientModule } from
"@angular/common/http";
import { FormsModule } from "@angular/forms";
import { ReactiveFormsModule } from "@angular/forms";
import { JwtModule } from "@auth0/angular-jwt";
import { BrowserAnimationsModule } from "@angular/platform-
browser/animations";
import { AUTH_API_URL, BPTASK_API_URL } from "../app-injection-tokens";
import { environment } from "../environments/environment";
import { ACCESS_TOKEN_KEY } from "../services/auth.service";
import { UserComponent } from "../components/user/user.component";
import { LoginComponent } from "../components/login/login.component";
import { RegisterComponent } from "../components/register/register.component";
import { JwtInterceptor } from "../_helpers/jwtinterceptor";
import { AppRoutingModuleModule } from "../app-routing.module";
import { AppComponent } from "../app.component";
import { TableModule } from "primeng/table";
import { CardModule } from "primeng/card";
import { ButtonModule } from "primeng/button";
import { PasswordModule } from "primeng/password";
import { InputTextModule } from "primeng/inputtext";
import { CalendarModule } from "primeng/calendar";
import { DialogModule } from "primeng/dialog";
import { ErrorInterceptor } from "../_helpers/ErrorInterceptor";
import { PaginatorModule } from "primeng/paginator";
import { BpTaskComponent } from "../components/bptask/bptask.component";

```

```

import { BpTaskFormComponent } from "../components/bptask-form/bptask-
form.component";
export function tokenGetter() {
    return localStorage.getItem(ACCESS_TOKEN_KEY);
}
@NgModule({
    declarations: [AppComponent,    UserComponent,    LoginComponent,
RegisterComponent, BpTaskComponent, BpTaskFormComponent],
    imports: [
        FormsModule,
        BrowserModule,
        AppRoutingModule,
        BrowserAnimationsModule,
        HttpClientModule,
        ReactiveFormsModule,
        TableModule,
        CardModule,
        ButtonModule,
        PasswordModule,
        InputTextModule,
        CalendarModule,
        DialogModule,
        PaginatorModule,

        JwtModule.forRoot({
            config: {
                tokenGetter,
                allowedDomains: environment.tokenWhitelistedDomains,
            },
        })
    ]
})

```

```
    }),  
  ],  
  providers: [  
    { provide: HTTP_INTERCEPTORS, useClass: JwtInterceptor, multi: true },  
    { provide: HTTP_INTERCEPTORS, useClass: ErrorInterceptor, multi: true },  
    {  
      provide: AUTH_API_URL,  
      useValue: environment.monoApi,  
    },  
    {  
      provide: BPTASK_API_URL,  
      useValue: environment.monoApi,  
    },  
  ],  
  bootstrap: [AppComponent],  
})  
export class AppModule {}
```

ДОДАТОК В
Слайди презентації

Міністерство освіти та науки України
Національний університет «Запорізька політехніка»

Кваліфікаційна робота магістра

ТЕМА: Дослідження та програмна реалізація методів
оптимізації системи моніторингу бізнес-процесів

1

Виконав Федонюк Роман Павлович
ст. гр. КНТ-222м

Керівник Шитікова Олена Вікторівна
к.т.н доцент кафедри ПЗ

Рисунок В.1 – Слайд № 1

2

Об'єкт та предмет дослідження. Мета роботи

- Об'єкт дослідження – процес оптимізації методів системи моніторингу бізнес-процесів.
- Предмет дослідження – програмні системи для оптимізації моніторингу бізнес-процесів.
- Метою роботи є дослідження та розробка вебдодатку для оптимізації системи моніторингу бізнес-процесів.

Рисунок В.2 – Слайд № 2

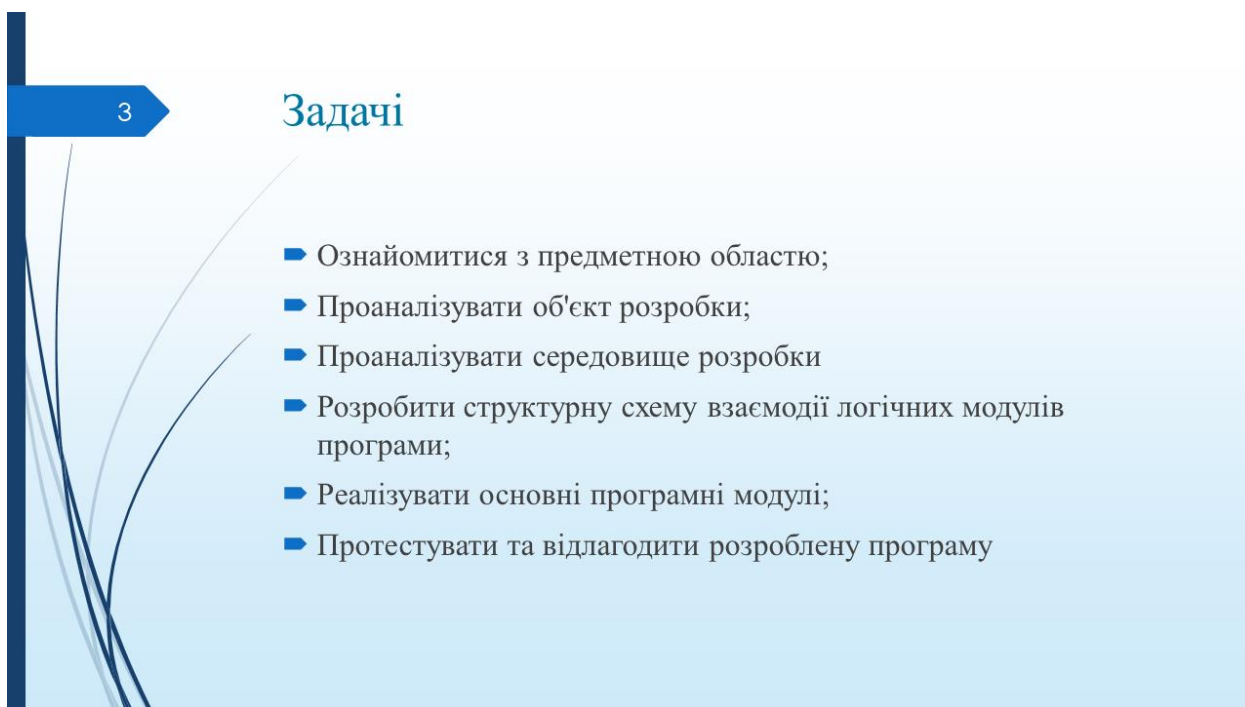


Рисунок В.3 – Слайд № 3

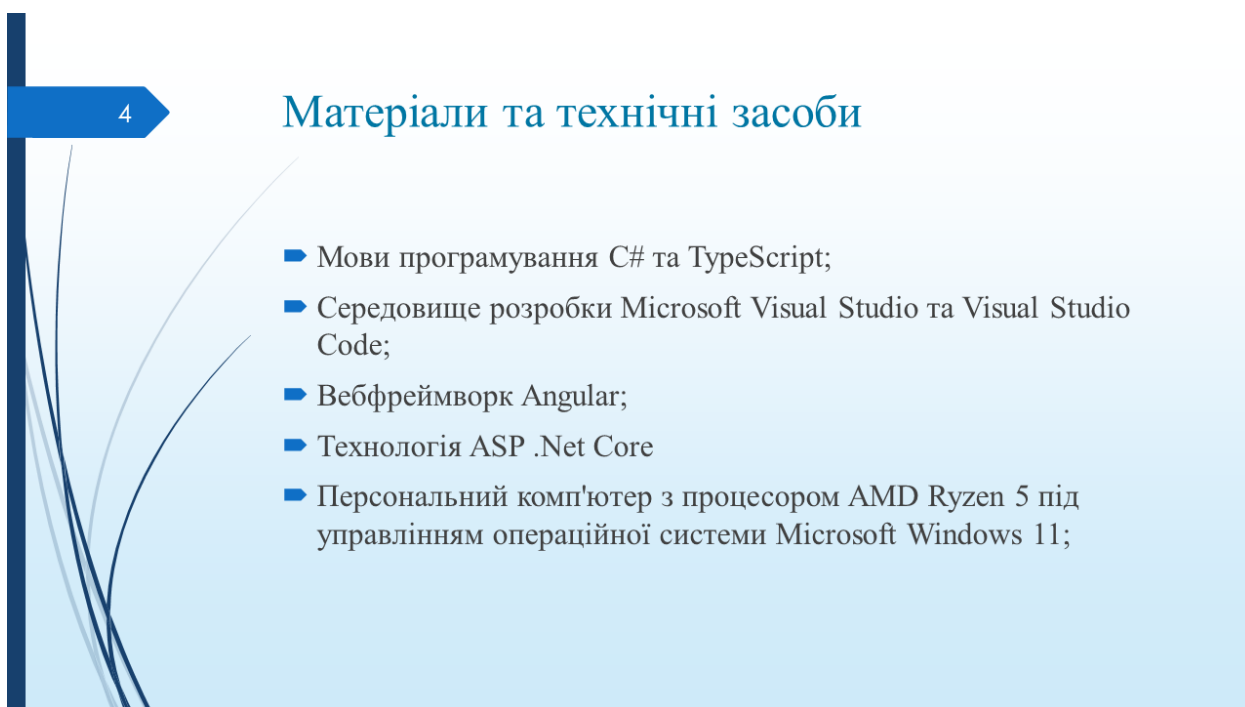


Рисунок В.4 – Слайд № 4

5

Порівняльна таблиця аналогів

Характеристика	BAS	SAP	Oracle	Microsoft Dynamics
Розробник	NetHelp	SAP SE	Oracle Corporation	Microsoft
Функціональність	Бухгалтерський облік	ERP	ERP, CRM, HR, SCM, BI	ERP, CRM, SCM, HR
Призначення	Малий та середній бізнес	Різні галузі та масштаби підприємств	Різні галузі та масштаби підприємств	Різні галузі та масштаби підприємств
Модульність	Основно спрямована на бухгалтерський облік	Широкий спектр модулів для управління різними аспектами бізнесу	Широкий спектр модулів для управління різними аспектами бізнесу	Широкий спектр модулів для управління різними аспектами бізнесу
Клієнтська база	В основному малий та середній бізнес	Різні масштаби підприємств	Різні масштаби підприємств	Різні масштаби підприємств
Сфера застосування	Фінанси та бухгалтерський облік	Різні галузі та аспекти бізнесу	Різні галузі та аспекти бізнесу	Різні галузі та аспекти бізнесу
Доступність	Локалізована, підходить для певних регіонів	Глобальне охоплення, доступна для багатьох країн	Глобальне охоплення, доступна для багатьох країн	Глобальне охоплення, доступна для багатьох країн

Рисунок В.5 – Слайд № 5

6

Актуальність дослідження

- За результатами проведеного порівняльного аналізу можна зробити висновок, що у сучасних програмних засобах для управління та оптимізації бізнес-процесів присутні різні переваги та недоліки які притаманні більшості комерційно орієнтованих систем. У розглянутих програмних засобах можна виділити такі основні недоліки як: висока вартість купівлі ліцензії, перевантаженість зайвим функціоналом, складність інтеграції у існуючі бізнес-процеси компанії, незручний користувацький інтерфейс.
- Тому актуальною є розробка програмного забезпечення, що оптимізує алгоритми системи моніторингу бізнес-процесів.

Рисунок В.6 – Слайд № 6

7

Функціональні вимоги

- При розробці програмна реалізація методів оптимізації системи моніторингу бізнес-процесів необхідно забезпечити такі функціональні вимоги:
 - можливість створення персонально облікового запису з налаштованими правами доступу;
 - можливість генерації та налаштування задач;
 - можливість призначення задачі обраному співробітнику;
 - пошук необхідного співробітника у базі даних;
 - можливість генерації звіту виконаних завдань;
 - можливість гнучкого підходу до взаємодії зі створеними бізнес-процесами.

Рисунок В.7 – Слайд № 7

8

Метод класифікації задачі в залежності від їх пріоритету та терміновості

У створеному методі класифікації задач для визначення категорії задачі використовується матриця Ейзенхауера, яка поділяє задачі на такі категорії:

- категорія А - важливі і термінові справи;
- категорія В - важливі, але не термінові справи;
- категорія С - не важливі, але термінові справи;
- категорія D - не важливі і не термінові справи.

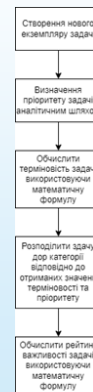


Рисунок В.8 – Слайд № 8

9

Графічне представлення матриці Ейзенхауера



Рисунок В.9 – Слайд № 9

10

Порівняння мов програмування

Характеристика	TypeScript	JavaScript	Java	C#
Тип мови	Статична з поступовою типізацією	Динамічна зі слабкою типізацією	Статична	Статична
Використання	Розробка вебдодатків, масштабні проекти	Веброзробка, повсякденні завдання	Великі платформи, мобільні додатки	Розробка на платформі .NET, ігри
Специфікація	Розширення JavaScript, додає статичну типізацію	Основна мова для вебу, динамічна типізація	Основа на віртуальній машині, статична типізація	Основа на платформі .NET, статична типізація
Екосистема	Розширює JavaScript, включає в себе всі його функції	Має широкий спектр бібліотек і фреймворків	Велика кількість бібліотек, включаючи Java SE та Java EE	Широкий вибір бібліотек та фреймворків у мережі .NET
Компіляція	Компілюється у JavaScript	Виконується безпосередньо у браузері або на сервері	Компілюється у байт-код JVM, який виконується на віртуальній машині	Компілюється у мову проміжного рівня CIL, яка виконується на платформі .NET
Строга типізація	Є статична типізація	Немає строгої типізації	Має строгу типізацію	Має строгу типізацію

Рисунок В.10 – Слайд № 10

11

Структура програмного забезпечення реалізації методів оптимізації системи моніторингу бізнес-процесів

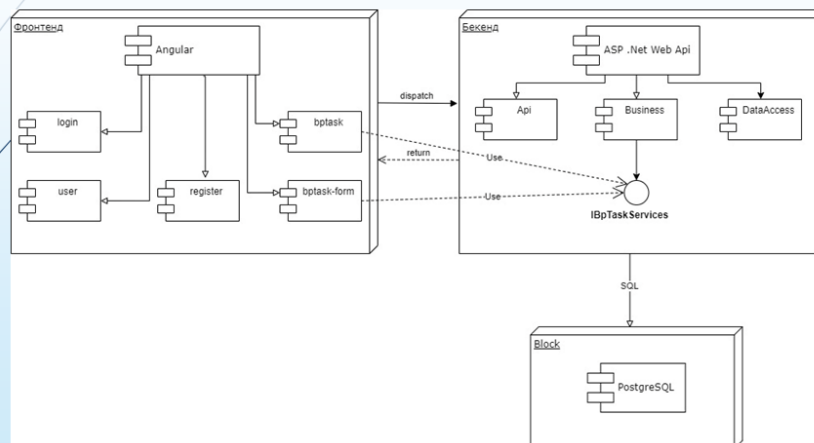


Рисунок В.11 – Слайд № 11

12

Структурна схема компонентів клієнтського Angular додатку

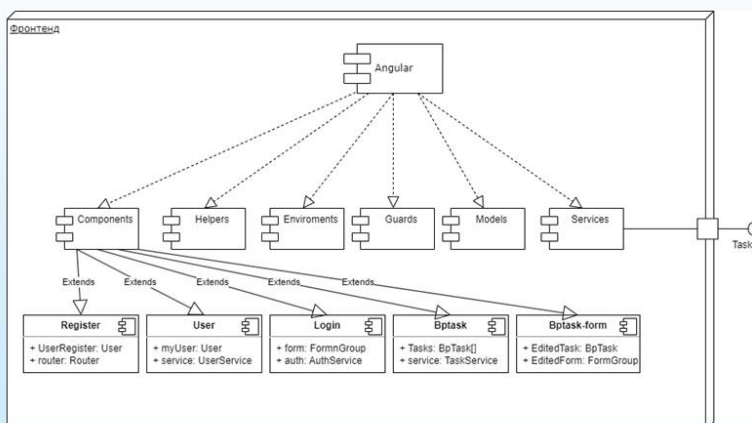


Рисунок В.12 – Слайд № 12

13

Алгоритми роботи програми

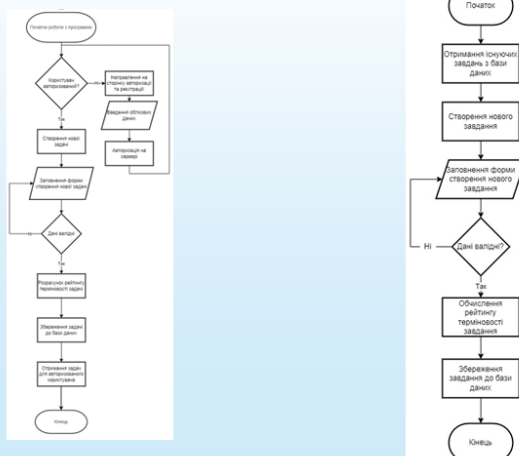


Рисунок В.13 – Слайд № 13

14

Схематичне зображення бази даних

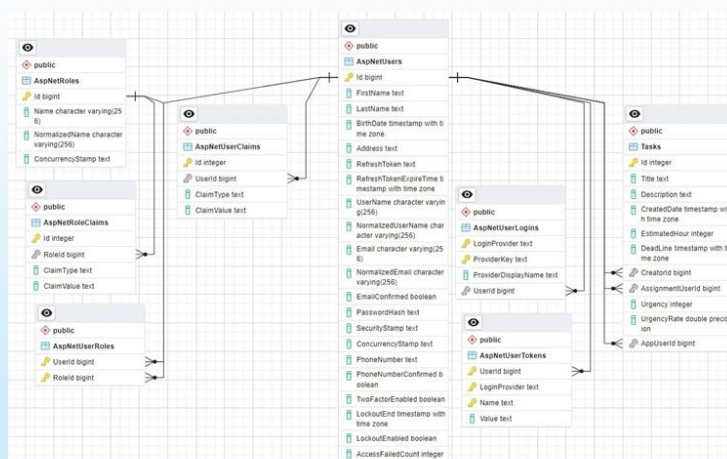


Рисунок В.14 – Слайд № 14

15

Інтерфейс головно вікна програми

Title	Description	Created Date	Estimated Hour	Deadline	Creator Id	Assignment User Id	Urgency	Urgency Rate	Actions
string5	string5	2023-11-05T22:00:00Z	7	2023-11-04T22:00:00Z	1	1	3	87.5	Edit Delete Done
string4	string4	2023-11-05T22:00:00Z	6	2023-11-04T22:00:00Z	1	1	2	75	Edit Delete Done
string7	string7	2023-11-05T22:00:00Z	5	2023-11-04T22:00:00Z	1	1	2	62.5	Edit Delete Done
string4	string4	2023-11-04T22:00:00Z	3	2023-11-05T22:00:00Z	1	1	2	37.5	Edit Delete Done
string8	string8	2023-11-07T22:00:00Z	6	2023-11-04T22:00:00Z	1	1	7	37.5	Edit Delete Done
string5	string5	2023-11-04T22:00:00Z	5	2023-11-04T22:00:00Z	1	1	3	31.25	Edit Delete Done
string3	string3	2023-11-05T22:00:00Z	2	2023-11-29T22:00:00Z	1	1	2	0	Edit Delete Done

Рисунок В.15 – Слайд № 15

16

Інтерфейс сторінки входу та реєстрації

Login

Username

Password

Login Register

Register

Username

Password

FirstName

LastName

BirthDate

Address

Register Cancel

Рисунок В.16 – Слайд № 16

Висновки

- В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено методи оптимізації системи моніторингу бізнес-процесів.
- Було вивчено основні поняття, напрямки і особливості сучасного впровадження систем моніторингу бізнес-процесів, а потім виконано огляд і аналіз існуючих програмних рішень.
- Для створення програмної реалізації методів оптимізації системи моніторингу бізнес-процесів було обрано такі мови програмування як C# для бекенд частини та мову TypeScript для фронтенд частини.
- У ході розробки було створено метод класифікації задачі в залежності від їх пріоритету та терміновості. Результатом виконання цього методу буде оцінена та відранжована задача.
- Була розроблена система моніторингу бізнес-процесів.

Рисунок В.17 – Слайд № 17