НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Кафедра автоматизованих систем обробки інформації і управління

До захисту допущено:
В.о. завідувача кафедри
_____ *Олександр ПАВЛОВ*
(підпис) (вл.ім'я, прізвище)
"___"_____2020 р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інформаційні управляючі системи та технології»
спеціальності 122 «Комп'ютерні науки та інформаційні технології»

на тему: «*Інформаційна система підтримки управління командою* Розробників»

Виконав: студент *IV* курсу, групи *ІС-63*

Ільєнко Роман Сергійович _____
(прізвище, ім'я, по батькові) (підпис)

Керівник *доц., к.т.н., Коган Алла Вікторівна* _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Консультант з графічної документації *ст. вик. Проскура Світлана Леонідівна* _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Рецензент *доц., к.т.н., Роковий Олександр Петрович* _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____
(підпис)

Київ – 2020 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет (інститут) *інформатики та обчислювальної техніки*
(повна назва)

Кафедра *автоматизованих систем обробки інформації і управління*
(повна назва)

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 «Комп'ютерні науки та інформаційні технології»

Освітньо-професійна програма «Інформаційні управляючі системи та технології»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ *Олександр ПАВЛОВ*
(підпис) (вл.ім’я, прізвище)
“___”_____2020 р.

З А В Д А Н Н Я
на дипломний проєкт студенту

Ільєнку Роману Сергійовичу
(прізвище, ім’я, по батькові)

1. Тема проєкту «*Інформаційна система підтримки управління* командою розробників»

керівник проєкту *Коган Алла Вікторівна, к.т.н., доцент*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від “*7*”*травня* 2020 р. №1081-с

2. Термін подання студентом проєкту “*01*”*червня* 2020 року

3. Вихідні дані до проєкту

Технічне завдання

4. Зміст пояснювальної записки

1. Загальні положення: основні визначення та терміни, опис предметного середовища, огляд ринку програмних продуктів, постановка задачі

2. Інформаційне забезпечення: вхідні дані, вихідні дані, опис структури бази даних

3. Математичне забезпечення: змістовна та математична постановки задачі, обґрунтування та опис методу розв’язання

4. Програмне та технічне забезпечення: засоби розробки, вимоги до технічного забезпечення, архітектура програмного забезпечення, побудова звітів

5. Технологічний розділ: керівництво користувача, методика випробувань програмного продукту

5. Перелік графічного матеріалу

1. Діаграма варіантів використання

2. Діаграма послідовності

3. Архітектура веб-застосування

4. Схема бази даних

5. Діаграма компонентів

6. Демонстраційний плакат до дипломного проекту

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання *«13» квітня 2020 року*

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	*Вивчення рекомендованої літератури*	*14.04.2020-16.04.2020*	
2.	*Аналіз існуючих методів розв'язання задачі*	*17.04.2020-21.04.2020*	
3.	*Постановка та формалізація задачі*	*22.04.2020-23.04.2020*	
4.	*Розробка інформаційного забезпечення*	*24.04.2020-27.04.2020*	
5.	*Алгоритмізація задачі*	*28.04.2020-01.05.2020*	
6.	*Обґрунтування використовуваних технічних засобів*	*02.05.2020-04.05.2020*	
7.	*Розробка програмного забезпечення*	*05.05.2020-10.05.2020*	
8.	*Налагодження програми*	*11.05.2020-12.05.2020*	
9.	*Виконання графічних документів*	*12.05.2020-13.05.2020*	
10.	*Оформлення пояснювальної записки*	*13.05.2020-14.05.2020*	
11.	*Подання ДП на попередній захист*	*15.05.2020*	
12.	*Подання ДП на основний захист*	*01.06.2020*	
13.	*Подання ДП рецензенту*	*02.06.2020*	

Студент Роман ІЛЬЄНКО

Керівник Алла КОГАН

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проєкт	2	
2	А4	ДП 6309.00.000 ПЗ	Пояснювальна записка	68	
3	А4	ДП 6309.01.000 ТЗ	Технічне завдання	11	
4	А3	ДП 6309.02.000 ССВ	Схема структурна варіантів використань	1	
5	А3	ДП 6309.03.000 ССП	Схема структурна послідовностей діяльності	1	
6	А3	ДП 6309.04.000 СБД	Схема бази даних	1	
7	А3	ДП 6309.05.000 ССК	Схема структурна компонентів	1	
			програмного забезпечення		
8	А3	ДП 6309.06.000 ССА	Схема структурна архітектури	1	
			веб-застосування		
9	А3	ДП 6309.07.000	Рішення з математичного забезпечення	1	

Зм.	Арк.	ПІБ	Підп.	Дата	ДП 6309 00.000.00			
Розробн.		Ільєнко Р.С.			Відомість дипломного проєкту	Літ.	Лист	Листів
Керівн.		Коган А.В.					1	1
Консульт.						КПІ ім. Ігоря Сікорського Каф. АСОІУ Гр. ІС-63		
Н/контр.		Проскура С.Л.						
В.о.зав.каф.		Павлов О.А.						

Пояснювальна записка
до дипломного проєкту

на тему: Інформаційна система підтримки управління командою розробників

Київ – 2020 року

АНОТАЦІЯ

Структура та обсяг роботи. Пояснювальна записка дипломного проєкту складається з шести розділів, містить 18 рисунків, 15 таблиць, 1 додаток, 16 джерел.

Дипломний проєкт присвячений розробці інформаційної системи підтримки управління командою розробників. Цілями розробки є реалізація паралельного алгоритму наївного баєсіва класифікатора, автоматичне визначення витрат часу на завдання на основі НБК, реалізація веб-застосунку керування станами завдань.

Для реалізації додатку потрібно вирішити такі задачі: реалізувати послідовний наївний баєсів класифікатор, розробити паралельний алгоритм наївного баєсіва класифікатора, реалізувати паралельний алгоритм наївного баєсіва класифікатора, створити навчальну вибірку для НБК, розробити схему бази даних, реалізувати базу даних, створити дизайн веб-застосунку, написати бекенд частину веб-застосунку, написати фронтенд частину веб-застосунку.

У розділі інформаційного забезпечення було визначено перелік вхідних та вихідних даних, розглянуто та описано структуру бази даних.

Розділ математичного забезпечення присвячений розробці паралельної моделі наївного баєсіва класифікатора.

Програмне забезпечення було описано діаграмами послідовності та компонентів, таблицею з переліком основних функції застосунку.

У технологічному розділі демонструється керівництво користувача для роботи із застосунком, а також випробування програмного продукту.

НАЇВНИЙ БАЄСІВ КЛАСИФІКАТОР, ЗАВДАННЯ, КІНЦЕВИЙ ТЕРМІН ЗДАЧІ, НАВЧАЛЬНА ВИБІРКА

ABSTRACT

Structure and scope of work. The explanatory note of the diploma project consists of six sections, contains 16 figures, 15 tables, 1 application, 16 sources.

The diploma project is devoted to the Information System for Development Team Management. The objectives of the development are the implementation of a parallel algorithm of naïve Bayes classifier, automatic determination of time spent on tasks based on the NBC, the implementation of a web application for managing task states.

To implement the application the following tasks need to be done: implement a consistent naïve Bayes classifier, develop a parallel algorithm for naïve Bayes classifier, implement a parallel algorithm of naïve Bayes classifier, Create a training sample for the NBC, develop a database schema, implement a database, create a web application design, write the backend part of the web application, write a frontend part of a web application.

In the section of information support the list of input and output data was defined, the structure of the database was considered and described.

The section of mathematical software is devoted to the development of a parallel model of naive classifier and comparison of its speed with a sequential version.

The software was described by sequence and component diagrams, a table with a list and information about the main functions of the software.

The technology section shows the user manual for working with the application, as well as testing the software product.

NAIVE BAYES CLASSIFIER, TASKS, DEADLINE, STUDY SAMPLING

ЗМІСТ

ВСТУП **5**

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ **7**

1.1 Опис предметного середовища 7

1.1.1 Опис процесу діяльності *8*

1.1.2 Опис функціональної моделі *9*

1.2 Огляд наявних аналогів 11

1.3 Постановка задачі 12

1.3.1 Призначення розробки *12*

1.3.2 Цілі та задачі розробки *12*

Висновок до розділу 13

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ **14**

2.1 Вхідні дані 14

2.2 Вихідні дані 14

2.3 Опис структури бази даних 15

Висновок до розділу 15

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ **17**

3.1 Змістовна постановка задачі 17

3.2 Математична постановка задачі 17

3.3 Обґрунтування методу розв’язання 18

3.4 Опис методів розв’язання 19

Висновок до розділу 20

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ **21**

4.1 Засоби розробки 21

4.2 Вимоги до технічного забезпечення 22

4.2.1 Загальні вимоги *22*

4.2.2 Опис локальної обчислювальної мережі *22*
4.3 Архітектура програмного забезпечення 23
4.3.1 Діаграма послідовності *23*
4.3.2 Діаграма компонентів *24*
4.3.3 Специфікація функцій *24*
Висновок до розділу 26
5 ТЕХНОЛОГІЧНИЙ РОЗДІЛ 27
5.1 Керівництво користувача 27
5.2 Випробування програмного продукту 31
5.2.1 Мета випробувань *32*
5.2.2 Загальні положення *32*
5.2.3 Результати випробувань *32*
Висновок до розділу 37
ЗАГАЛЬНІ ВИСНОВКИ 38
ПЕРЕЛІК ПОСИЛАНЬ 39
ДОДАТОК А ..... ERROR! BOOKMARK NOT DEFINED.

ВСТУП

В сучасному світі складність технічних проектів, що з розвитком інтернет технологій охоплюють все більшу аудиторію, постійно зростає. Якщо колись невелика команда могла випустити повноцінне програмне забезпечення, то тепер, зі складністю роботи, зростає і розмір команди. Завдання декомпозується на менші частини, передається кожному окремому вузькому спеціалісту. Потреби у якості значно зросли. Тепер недостатньо, щоб застосунок просто працював, він повинен функціонувати при будь-яких умовах, з будь-яким технічним оснащенням і максимальною швидкодією. Також значно зросли вимоги до збереження конфіденційних даних. Кожен маленький шматок коду треба покривати великою кількістю тестів задля зведення до мінімуму неочікуваних ситуацій.

Все це призвело до надзвичайно складного менеджменту проектів у сфері інформаційних технологій. Одним із методів вирішення цього питання є гнучкі методології. Суть більшості із них полягає у створенні спеціальної дошки, що поділена на секції, кожна секція відповідає за певний етап розробки. Уся розробка ділиться на менші частини, визначаються інтервали часу на їх виконання. Це дозволяє зручно контролювати процес роботи, а також вчасно обговорювати із замовником усі зміни.

Як правило, таким процесом керує окремий спеціаліст – проектний менеджер. Він, керуючись своїм досвідом, може визначити кількість витрат часу на окреме завдання і виставляти адекватні кінцеві терміни демонстрації закінчених завдань. Але що робити коли невеликий стартап чи компанія не може собі дозволити ще одне робоче місце, а вимоги до якості програмного продукту такі самі, як і до розробок великих компаній?

Усе це призвело до необхідності розробки інструментів, що автоматично можуть визначати кількість витрат часу. Дана робота використовує методи машинного навчання для прогнозування цього параметру. Вона дозволить маленьким компаніям або стартапам зменшити необхідність у додаткових витратах на менеджмент, а програмістам сконцентруватися на розробці.

Дипломний проєкт присвячений розробці інформаційної системи підтримки управління командою розробників.

Практичне значення одержаних результатів. Розроблено веб-застосунок для роботи, розроблена та реалізована паралельна модель наївного баєсіва класифікатора.

Публікації. Результати роботи були опубліковані у тезах доповідей на науково-технічних конференціях.

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Опис предметного середовища

Предметне середовище – інформаційна система підтримки управління командою розробників. Однією із найважливіших задач проектного менеджеру є визначення кількості витрат часу які треба надати тій або іншій задачі. Як правило, оцінка тим точніше, чим більше досвіду має спеціаліст. Він розуміє усі тонкощі роботи команди розробників, знає до якого класу важкості можуть відноситись поставленні замовником вимоги, передбачає можливі проблеми. Використовуючи аналіз, з кожним проектои оцінка часу, що буде витрачене на виконання замовлення, має набувати більшою значущості. Але таких менеджерів доволі важко знайти, також не кожна компанія може собі дозволити високооплачуваного фахівця та зайве робоче місце.

Система створена для автоматизації процесу визначення можливої кількості витрат часу на виконання завдання, що було поставлене замовником. Існує дошка з таких розділів: «у планах», «у розробці», «виконане». Кожен розділ відповідає за те, у якому стані знаходиться завдання. Це типова схема роботи гнучких методологій. Спочатку завдання додається у загальний список. Тут треба додати опис завдання, виконавця та кількість умовних одиниць витрат часу. Показник умовних одиниць витрат часу відповідає тому скільки з найбільшої ймовірністю знадобиться часу на виконання цього завдання. Як правило, від команди до команди, він не прив’язаний до конкретної часової міри і може бути як 1 годиною, так і одним днем. Але у даній системі цей показник стало дорівнює 2 годинам, а максимальна кількість у одного завдання - 20. По-перше, такий часовий проміжок дозволить спростити алгоритм передбачення, по-друге, більшість джерел по гнучким методологіям вказують на те, що завдання, які займають маленькі проміжки часу, повинні об’єднуватись в більші, а надто великі розбиватися на менші, це спрощує процес менеджменту та дозволяє більш точно передбачувати витрати часу. Даний показник повинен автоматично виставлятися після заповнення

опису завдання. Функціонал буде працювати на основі розпаралеленого наївного баєсіва класифікатора. Але у разі невідповідності очікуванням, його можна замінити вручну. Під час мануального введення, дані додаються до загальної навчальної вибірки. Після додавання завдання у загальний список, воно може вільно переміщатися між станами.

База даних повинна містити інформацію про проект та завдання. Також вона має бути наповнена навчальною вибіркою для роботи наївного баєсіва класифікатора.

Також існує можливість перегляду повної історії зміни станів завдань в окремому вікні.

1.1.1 Опис процесу діяльності

В даній роботі розглядається автоматизація наступних процесів:

- Менеджмент завдань всередині гнучкої методології.
- Визначення тривалості виконання завдання.

До автоматизації менеджмент завдань під час звичайного процесу здійснюється на звичайній дошці на якій відзначено певні стани завдання. Кожне завдання записується на окремому папірці і кріпиться на цю дошку. Далі воно переміщається власноруч кожним окремим членом команди розробників.

Такий спосіб керування завданнями є зручний, якщо спеціалісти знаходяться в одному приміщенні та бажано недалеко один від одного. Але він зовсім не підходить для розподіленої команди, яка працює дистанційно. До того ж використання великої кількості папірців не є розумним з точки зору охорони екології.

Також тривалість виконання завдання визначалась за допомогою окремого спеціаліста зі значним досвідом або власними зусиллями розробників. Це приводило до значної кількості проблем. У першому випадку такий фахівець може бути надто витратним, а його пошук, як правило, займає значний час. У другому варіанті амбіціозні програмісти можуть переоцінити

власні можливості і зірвати всі терміни. Від цього будуть потерпати як самі виконавці, так і замовники.

Після автоматизація керувати процесом розробки та виконання завдань можна буде дистанційно за допомогою сучасних веб-застосунків, що дозволить організовувати роботу незалежно від фізичного місцерозташування програміста. Також такий спосіб комунікації полегшить збір статистичної інформації для подальшого інтелектуального аналізу та оптимізації процесів діяльності.

Тривалість виконання завдань буде здійснюватися автоматично за допомогою методів машинного навчання. А оцінка буде тим точніше, чим більша навчальна вибірка буде присутня на момент визначення кількості умовних одиниць витрат часу по конкретному завданню.

Користувачу достатньо буде зареєструвати свій проект. Після реєстрації йому на пошту прийде унікальний uuid (стандарт ідентифікації, який використовується при створенні програмного забезпечення, затверджений Open Software Foundation (OSF) як частина Розподіленого комп'ютерного середовища (DCE)). Далі він вводить його у вікно авторизації та отримує доступ до інформаційної системи підтримки управління командою розробників. Тут з’являється можливість інтерактивного керування станами завдань, учасників та інше. Автоматичне визначення тривалості виконання завдання здійснюється після вводу опису задачі у відповідне вікно інтерфейсу програмного веб-застосунку. Таким чином система визначення витрат часу буде працювати швидко і автоматично без залучення окремих спеціалістів.

1.1.2 Опис функціональної моделі

Єдиним актором у системі є користувач - дійова особа в інформаційній системі підтримки управління командою розробників. Може редагувати дані по проекту та завданню.

Тепер визначимо дії, які можуть виконувати актор системи та відобразимо їх за допомогою структурної схеми варіантів використання.

Таблиця 1.1 – Типи варіантів використання

Актор	Варіант використання	Опис дії варіанта використання
Користувач	Зареєструватися	Користувач заповнює поля з поштою та назвою проекта на сторінці реєстрації. Потім натискає кнопку реєстрації.
Користувач	Авторизація	Користувач вводить uuid проекту та натискає кнопку авторизації на сторінці авторизації.
Користувач	Додавання користувача	Користувач натискає кнопку додавання користувача на головній сторінці, вводить ім’я та підтверджу є додавання.
Користувач	Видалення користувача	Користувач натискає кнопку видалення навпроти користувача на головній сторінці.
Користувач	Додавання завдання	Користувач натискає кнопку додавання завдання на сторінці завдань. Обирає ім’я відповідального, вводить назву завдання, опис, кількість story point та дедлайн. Підтверджує додавання.

Продовження таблиці 1.1

Користувач	Додавання завдання в активні	Користувач за допомогою миші перетягує завдання із списку неактивних завдань в список активних завдань на сторінці списку завдань.
Користувач	Зміна стану завдання	Користувач перетягує завдання в стовпчик іншого завдання на сторінці активних завдань.
Користувач	Перехід на сторінку активних завдань	Користувач натискає на кнопку активних завдань.
Користувач	Перехід на сторінку завдань	Користувач натискає на кнопку завдань
Користувач	Перехід на сторінку історії мін	Користувач натискає на кнопку історії змін

Діаграма варіантів використання наведена у графічному матеріалі

1.2 Огляд наявних аналогів

Гнучка методологія scrum є досить популярною у сфері керування процесу розробки програмного забезпечення. Було знайдено декілька відомих сервісів. Але усі популярні рішення мають закритий вихідний код і не надають змогу модифікації основних елементів, що були задіяні у даному дипломному проекті. Розглянемо один з них. На рисунку 1.1 продемонстровано сторінку власного проекту на сервісі «Trello». Як можна побачити, тут нема можливості ведення повноцінного backlog. А за весь процес організації часу відповідає сам менеджер.

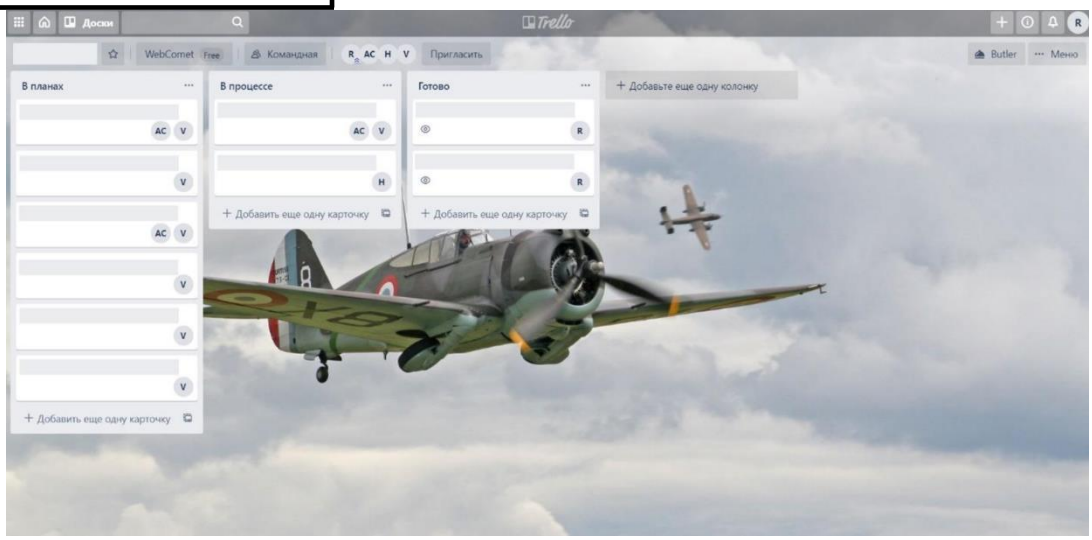


Рисунок 1.1 – Скріншот проекту на сервісі Trello

Також існує такий аналог як «meistertask», основне вікно можна побачити на рисунку 1.2, але він так само не дає змогу ведення story point та не володіє функціоналом передбачення.

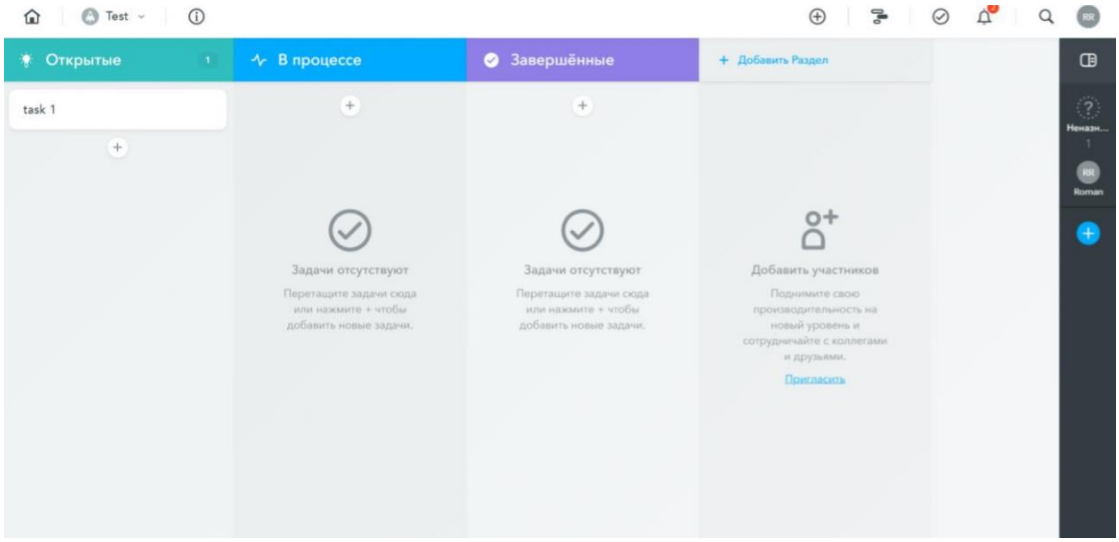


Рисунок 1.2 – Скріншот з сервісу «meistertask»

1.3 Постановка задачі

1.3.1 Призначення розробки

Призначення розробки – створення інформаційної системи підтримки управління командою розробників.

1.3.2 Цілі та задачі розробки

Цілями розробки системи є:

– Реалізація паралельного алгоритму наївного баєсіва класифікатора. Автоматичне визначення витрат часу на завдання на основі НБК.
– Реалізація веб-застосунку керування станами завдань.

Для досягнення поставлених цілей необхідно реалізувати наступні задачі:

– Реалізувати послідовний наївний баєсів класифікатор.
– Розробити паралельний алгоритм наївного баєсіва класифікатора.
– Реалізувати паралельний алгоритм наївного баєсіва класифікатора.
– Створити навчальну вибірку для НБК.
– Розробити схему бази даних.
– Реалізувати базу даних.
– Створити дизайн веб-застосунку.
– Написати серверну частину веб-застосунку.
– Написати клієнтську частину веб-застосунку.

Висновок до розділу

В цьому розділі було обґрунтовано доцільність розробки інформаційної системи підтримки управління командою розробників. Було описано процес діяльності до та після автоматизації. Були висунуті функціональні вимоги до системи, сформульовано цілі та задачі розробки даної системи. Також проведено порівняльний аналіз розроблюваної системи та вже існуючих аналогів.

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Вхідні дані

Усі вхідні дані для роботи системи надходять від користувача.

При реєстрації користувач повинен вказати наступні параметри:

- Пошту
- Назву проекту

Пошта потрібна для того, щоб відправляти на неї uuid проекту. А назва проекту буде висвічуватись на головній сторінці.

Основним елементом взаємодії є завдання. Для того, щоб додати завдання треба заповнити такі поля:

- Відповідальний
- Назва
- Опис
- Кількість умовних одиниць витрат часу
- Дата дедлайну

Завдання можна редагувати. Змінювати стан або додавати, видаляти зі списку активних завдань.

Також є можливість додавання учасників проекту, для цього достатньо натиснути на відповідний елемент інтерфейсу та ввести його повне ім’я.

2.2 Вихідні дані

Вихідними даними роботи системи є формування списків завдань, визначення тривалості виконання завдань та історія змін стану завдань.

Завдання представлені наступним переліком параметрів:

- Стан завдання
- Наявність у списку активних завдань
- Назва
- Опис

– Кінцевий термін здачі
– Кількість умовних одиниць витрат часу
– Відповідальний

Тривалість виконання визначається по-замовчуванням та характеризується параметром умовних одиниць витрат часу.

Історія зміни станів завдання характеризується такими параметрами:

– Назва завдання
– Попередній стан
– Новий стан
– Дата зміни стану

2.3 Опис структури бази даних

Під час розробки системи використовувалась реляційна база даних PostgreSQL. У проекті використовується контейнерізована мікросервісна архітектура. Тому БД працює в окремому контейнері Docker. Схема БД наведена у графічних матеріалах.

Наведемо перелік сутностей системи в таблиці 2.1:

Таблиця 2.1 – Перелік сутностей

№	Назва таблиці	Сутність
1	teaching_data	Навчальна вибірка
2	task	Завдання
3	member	Учасник
4	project	проект

В таблиці 2.2 наведено детальний опис таблиць бази даних.

Таблиця 2.2 – Детальний опис таблиць бази даних.

Назва таблиці	Назва стовпця	Тип даних	Опис поля
task	task_id	SERIAL	Первинний ключ
	project_uuid	UUID	uuid проекту

Продовження таблиці 2.2

task	member_id	INTEGER	id відповідального
	name	TEXT	Назва завдання
	description	TEXT	Опис завдання
	story_points	INTEGER	Кількість story point
	deadline	DATE	Дедлайн
	isActive	BOOLEAN	Стан sprint\backlog
	state_id	INTEGER	Номер стану
teaching_data	data_id	SERIAL	Первинний ключ
	text	TEXT	Текст завдання
	story_points	INTEGER	Кількість story point
member	member_id	SERIAL	Первинний ключ
	name	TEXT	Ім’я учасника
	project_uuid	UUID	uuid проекту
project	project_uuid	UUID	Первинний ключ
	name	TEXT	Назва проекту

Висновок до розділу

В даному розділі було визначено перелік вхідних та вихідних даних. Було наведено перелік сутностей, котрі використовуються в системі, із вказанням таблиць, до яких вони відносяться. Також побудовано таблицю, в котрій наведено список таблиць бази даних з детальним описом всіх параметрів.

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Змістовна постановка задачі

Визначення тривалості виконання завдання є одним із найважливіших обов’язків проектного менеджера. Але залучення окремого спеціаліста для цього не завжди є доцільним. Однією із функції веб-застосунку є автоматичне визначення витрат часу на виконання завдання.

Вирішите це питання допомагає наївний баєсів класифікатор. Також він повинен мати паралельну модель, щоб задіяти усі потужності сучасних комп’ютерів.

Користувачу мають змогу вводити лише назву завдання, а програмне забезпечення автоматично за допомогою методів машинного навчання знаходить приблизні витрати часу.

3.2 Математична постановка задачі

Потрібно визначити час виконання певного завдання, тобто виявити до якого класу воно відноситься. Припустимо, що завдання - це документ, а кількість story point - це клас, отже треба визначити приналежність першого до другого. Спочатку будуємо модель класифікатору. Для цього існує навчальна вибірка з D документів, що постійно наповнюється коригуваннями користувачів. У кожного окремого класу c існує D_c документів. k - кількість унікальних слів у документах навчальної вибірки. Також треба підрахувати W_{ic} (скільки разів i-е слово зустрічається в навчальній вибірці c класу). А $\sum_{j=1}^{k}(W_{jc})$ - загальна кількість слів документу c в навчальній вибірці. Володіючи цими даними для кожного класу визначимо значення за формулою 3.1:

$$log\frac{D_c}{D} + \sum_{i=1}^{n} log\frac{W_{ic}+1}{k+\sum_{j=1}^{k}(W_{jc})} \quad (3.1)$$

Тоді цільова функція задачі матиме вигляд як на формулі 3.2:

$$C_{map} = [log\frac{D_c}{D} + \sum_{i=1}^{n} log\frac{W_{ic}+1}{k+\sum_{j=1}^{k}(W_{jc})}] \quad (3.2)$$

Тобто задача полягає в наступному: визначити до якого класу належить певне завдання.

3.3 Обґрунтування методу розв'язання

Було обрано наївний баєсів класифікатор через свою простоту у розпаралелюванні. При великих обсягах даних підраховувати синхронно в одному потоці кожний окремий клас може займати доволі великі показники часу. До того ж не використовується усі можливості сучасних багатопроцесорних комп'ютерів.

Вирішити це питання допомагає паралельна модель алгоритму. Порівняльні дані можна побачити на таблиці 3.1.

Таблиця 3.1 – Порівняння прискорення різних моделей алгоритму

К-ть слів	Послідовно	Паралельно	Прискорення
10	1423	34620	0,04110340843
100	23705	37440	0,6331463675
1000	38807	54954	0,7061724351
10000	360087	212068	1,69797895
100000	3099918	2336826	1,326550629
1000000	35943611	22484599	1,598588038
10000000	363140431	242544790	1,497209777

У графічних матеріалах наведено візуалізацію прискорення на певній кількості слів у класі. Видно, що на маленьких об'ємах даних послідовний алгоритм працює швидше за паралельний. Це результат того, що витрати на створення та керування потоками більші ніж витрати на виконання цього завдання послідовно. Але можна побачити як паралельна модель набуває сенсу з масивами близькими до 10000 слів.

3.4 Опис методів розв'язання

Для визначення поставленої задачі, а саме до якого класу належить певне завдання, пропонується використовувати послідовний наївний баєсів алгоритм, що базується на теоремі Баєса, за формулою 3.3:

$$P(c|d) = \frac{P(c)*P(d|c)}{P(d)}, \quad (3.3)$$

де $P(c|d)$ - ймовірність, що документ d належить класу c;

$P(d|c)$ - правдоподібність, тобто ймовірність даного значення ознаки при даному класі.

$P(d)$ та $P(c)$ - безумовна ймовірність зустріти документи d та c відповідно в корпусі документів.

В машинному навчанні наївні баєсові класифікатори - це сімейство простих ймовірнісних класифікаторів. Моделі на основі НБА дуже корисні при роботі з великими наборами даних. Метою даного метода є виявлення приналежності документу до певного класу. Так за формулою 3.4 визначається найбільш ймовірний клас:

$$C_{map} = arg \max_{c \in C} \frac{P(c)*P(d|c)}{P(d)} \quad (3.4)$$

Таким чином за формулою 3.5 необхідно для кожного класу підрахувати ймовірність та обрати той, який найбільше підходить.

$$C_{map} = arg \max_{c \in C} [P(c) * P(d|c)] \quad (3.5)$$

Після припущення умовної незалежності, де умовна ймовірність документу може бути представлена як добуток умовних ймовірностей всіх слів, що входять до цього документу, та вирішенню проблеми арифметичного переповнення, використовуючи властивість логарифма добутку $log(ab) = log(a) + log(b)$, враховуючи проблему невідомих слів разом зі згладжуванням Лапласа, кінцева формула набуває вигляду 3.6:

$$C_{map} = [log \frac{D_c}{D} + \sum_{i=1}^{n} log \frac{W_{ic}+1}{k+\sum_{j=1}^{k}(W_{jc})}] \quad (3.6)$$

Представимо запропонований алгоритм:

КРОК 1. На вхід подається навчальна вибірка з с кількості класів. В кожному класі с існує D_c документів. Також поступає і документ, клас якого треба знайти.

КРОК 2. Підраховується загальна кількість унікальних слів в усіх класах навчальної вибірки.

КРОК 3. Підраховується кількість документів в усіх класах навчальної вибірки.

КРОК 4. Для кожного класу рахується сумарна кількість слів по всіх документах.

КРОК 4.1. Додається значення логарифмів, де у чисельнику стоїть значення кількості повторень і-го слова з документу, клас якого треба визначити. До цього значення додається 1 (адитивне згладжування). А у знаменнику сума загальної кількості унікальних слів в усіх класах навчальної вибірки та значення кількості слів в цьому класі.

КРОК 4.2. До суми логарифмів додається логарифм, де у чисельнику стоїть кількість документів в цьому класі, а у знаменнику загальна кількість документів.

КРОК 5. Отримуємо деякий набір значень.

КРОК 6. З цього набору максимальне значення належить класу що з найбільшою ймовірністю відповідає даному документу.

Висновок до розділу

Після аналізу наївного баєсіва алгоритму, можна зробити висновок, що даний класифікатор гарно підходить для вирішення задачі визначення тривалості виконання завдання у інформаційній системі підтримки управління командою розробників. Найбільшої результативності він набуває з паралельною схемою роботи при великих об’ємах даних та відповідній кількості процесорів.

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Засоби розробки

Веб-застосунок був реалізований з використанням мов програмування javascript та golang. Він працює за клієнт-серверною архітектурою. Клієнтська частина написана на html, css, javascript та фреймворком react. Бекенд частина складається з двох компонентів: серверу, що написаний на javascript та відповідає за бізнес-логіку, та окремого алгоритму обчислення наївного баєсіва класифікатора, який використовує golang. Хоча проект побудований на мікросервісній архітектурі, засоби розробки обирались так, що і клієнт, і сервер писались на одній мові програмування. Це дозволило сконцентруватися на задачі, оминувши складності використання багатомовної ієрархії. Але тонкі місця, які потребують швидких багатоядерних математичних обчислень, написані на golang. Такий підхід дозволив зменшити показник time to market та швидкість підрахунку алгоритму.

React дозволяє створювати інтерактивні користувацькі інтерфейсі за допомогою веб-технологій. Він використовує компонентний підхід. Кожна окрема сторінка чи блок з інформацією може бути окремим компонентом зі своїм станом. У react стан нагадує приватні властивості або методи всередині класу в ООП. Коли оновлюється певні змінні - компонент заново опрацьовується з новими даними. Інформацію можна брати як з вводу користувача або локального вмістилища, так і з віддаленого сервера. В даному випадку відбувається комунікація з сервером по rest api.

Для статичної верстки використовувалася мова розмітки гіпертекстових документів HTML. Для надання привабливого вигляду застосунку було додано каскадні таблиці стилів CSS.

Javascript - це високорівнева мультипарадигмальна мова програмування, яка може використовуватися як на клієнтській частині, так і на серверній.

Node.js - асинхронний сценарій виконання JavaScript, він призначений для створення масштабованих мережевих додатків.

Fastify - це ефективний фремворк для написання rest api на сервері. Він має потужні архітектуру, що базується на написанні плагінів, та займає одне з перших місць по швидкодії серед усіх інших бекенд javascript фреймворків.

Golang - потужна мова програмування, розроблена google. Дозволяє отримати обчислювальні потужності близькі, а інколи швидші, за C при значно меншому часі розробки. Також ця мова програмування дозволяє надзвичайно ефективно та легко розпаралелити алгоритм під багатоядерну модель при використанні потоків.

4.2 Вимоги до технічного забезпечення

4.2.1 Загальні вимоги

Коректне функціонування даного застосунку на стороні клієнта може гарантуватися за наступних умов:

Операційна система: linux, windows, mac os.

Характеристика комп’ютера:

- процесор з тактовою частотою від 1ГГц
- оперативна пам’ять з об’ємом не менше 1гб
- підключення до інтернету

Комп’ютерна периферія:

- Мишка
- Клавіатура
- Монітор

Браузер:

- Google Chrome 80+
- Firefox 70+

4.2.2 Опис локальної обчислювальної мережі

Додаток працює на мікросервісній архітектурі. Клієнт з сервером спілкуються по REST API. На кожну дію таку як редагування інформації або

здійснення аутентифікації написаний окремий endpoint. Сервер в свою чергу спілкується з базою даних за допомогою sql запитів як можна побачити на рисунку 4.1. Також як окремий компонент функціонує додаток для обчислення математичних операцій для підвищення швидкої, викликається за допомогою системних викликів на тій самій машині, де працює сам сервер з бізнес логікою.

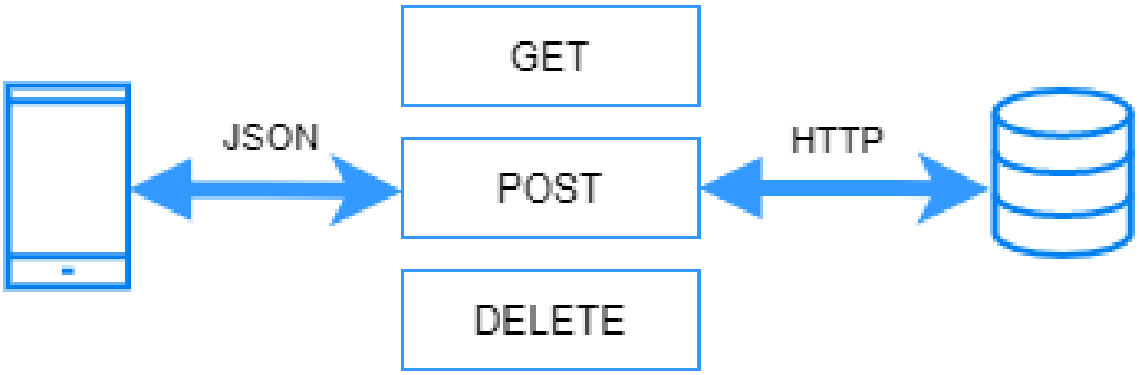


Рисунок 4.1 – Демонстрація схеми роботи застосунку

4.3 Архітектура програмного забезпечення

Архітектура програмного забезпечення наведена у графічних матеріалах.

4.3.1 Діаграма послідовності

Діаграма послідовності знаходиться у графічних матеріалах.

У даній діаграмі послідовності показана послідовність користувача. Для її відтворення весь додаток повинен функціонувати коректно.

Під час першого знайомства з додатком користувачу потрібно пройти реєстрацію. Він вводить необхідні дані. Далі вони вносяться у БД. А йому повертається унікальний uuid для подальшої авторизації.

Потім йому треба пройти етап авторизації. Для цього достатньо ввести цей ідентифікатор. Після вдалого проходження аутентифіції, користувач направляється на головну сторінку додатку.

Після потрапляння на головну сторінку, він може додати користувача. Для цього йому необхідно натиснути на відповідну кнопку та заповнити поля

з інформацією. Після чого дані надсилаються на сервер та повертається список з новими користувачами. Також користувачів можна видаляти.

Користувачу також надається можливість додавання завдання до списку завдань. Йому потрібно натиснути на відповідний елемент інтерфейсу, додати запропоновані дані та підтвердити відправлення. Завдання надсилається на сервер та додається у БД. Потім воно з'являється у списку завдань на стороні клієнтів. Так само є кнопка для видалення завдання.

Змінити стан завдання можна тільки після додавання його у список теперішніх завдань. Для цього треба перетягнути завдання з загального списку до списку теперішніх. Дані оновлюються в БД та надається змога редагування стану.

Після додавання завдання до теперішніх, можна змінити його стан на "у планах", "в процесі" чи "виконано" шляхом перетягування його у відповідну колонку. Дані оновлюються також на сервері та у БД.

4.3.2 Діаграма компонентів

Діаграму компонентів наведена у графічних матеріалах. З неї видно, що клієнт спілкується з серверов по REST API використовуючи endpoint для певної дії. Сервер в свою чергу надсилає запити та спілкується з базою даних.

4.3.3 Специфікація функцій

Специфікація основних функцій можна побачити у таблиці 4.1

Таблиця 4.1 – Специфікація основних функція

Функція	Опис
checkAuth()	Використовується для перевірки користувача
getProject()	Використовується для отримки інформації про проект
enterUuid()	Використовується для запису даних для авторизації
deleteMember()	Використовується для видалення учасника з проекту
insertMember()	Використовується для додавання користувача
setMembers()	Використовується для виводу користувачів
modalActions()	Використовується для створення модального вікна додавання завдань
addTaskToDB()	Використовується для додавання завдання у базу даних
removeTaskFromDB()	Використовується для видалення завдання з бази даних
addTask()	Використовується для додавання завдання у інтерфейс користувача
setTasksToBacklog()	Використовується для виводу завдань з БД
onDrop()	Використовується для перетягування завдання у теперішній стан

Висновок до розділу

У даному розділі було розглянуто засоби розробки, що були залучені при проектуванні дипломного проекту. Описано їх індивідуальні характеристики. Також прописані вимоги до технічного оснащення для роботи з даним додатком. Опис локальної обчислювальної системи продемонстрував схему роботи додатки з окремим клієнтом та rest api сервером, що в свою чергу виступає також і провідником з базою даних.

На діаграмі послідовності можна побачити типовий варіант використання застосунку та ілюстрацію взаємодії між користувачем (клієнтом), сервером та БД. Додано діаграму компонентів, що описує механізм взаємодії основних частин додатку.

Для ліпшого розуміння принципів роботи програми, була наведена таблиця з інформацією про основні програмні функції.

5 ТЕХНОЛОГІЧНИЙ РОЗДІЛ

5.1 Керівництво користувача

Для використання застосунку треба відкрити відповідне посилання у вікні браузера. Під час першого переходу виводиться сторінка авторизації. Для того щоб увійти у додаток, треба зареєструватися. Для цього потрібно ввести пошту та назву проекту як на рисунку 5.1.

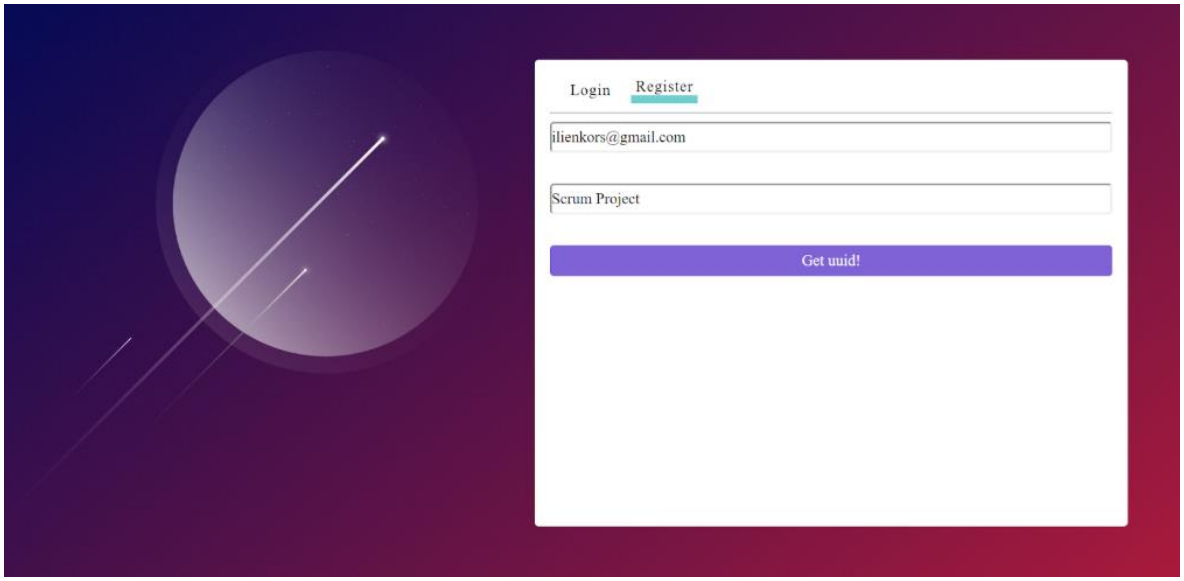


Рисунок 5.1 – Заповнення реєстраційних даних

Після реєстрації користувач отримує унікальний uuid як на рисунку 5.2.

```
▼Object                                                    login.js:24
   uuid: "2517657b-d615-4b1b-b886-4c171aacbeb3"
  ▶ __proto__: Object
```

Рисунок 5.2 – Отримання uuid проекту

Для того щоб авторизуватися треба ввести uuid проекту та натиснути кнопку підтвердження як на рисунку 5.3.

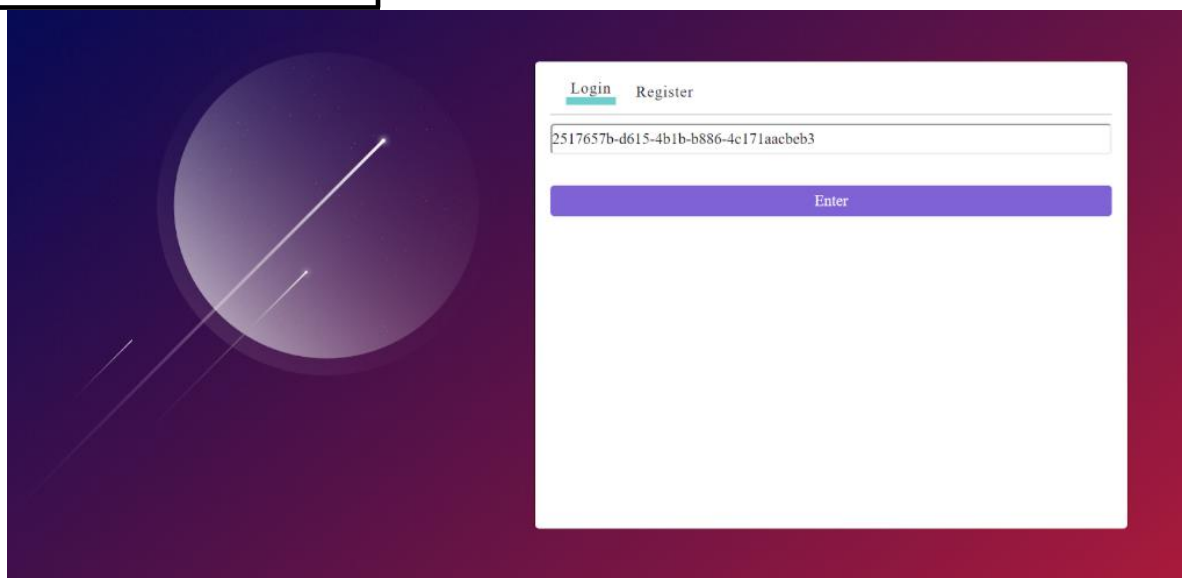


Рисунок 5.3 – введення uuid проекту

Наступним етапом після авторизації є перехід на головну сторінку. На рисунку 5.4 зліва представлена панель для додавання користувачів та три посилання. Сторінка із завданнями, сторінка з активними завданнями та історію змін в проекті.

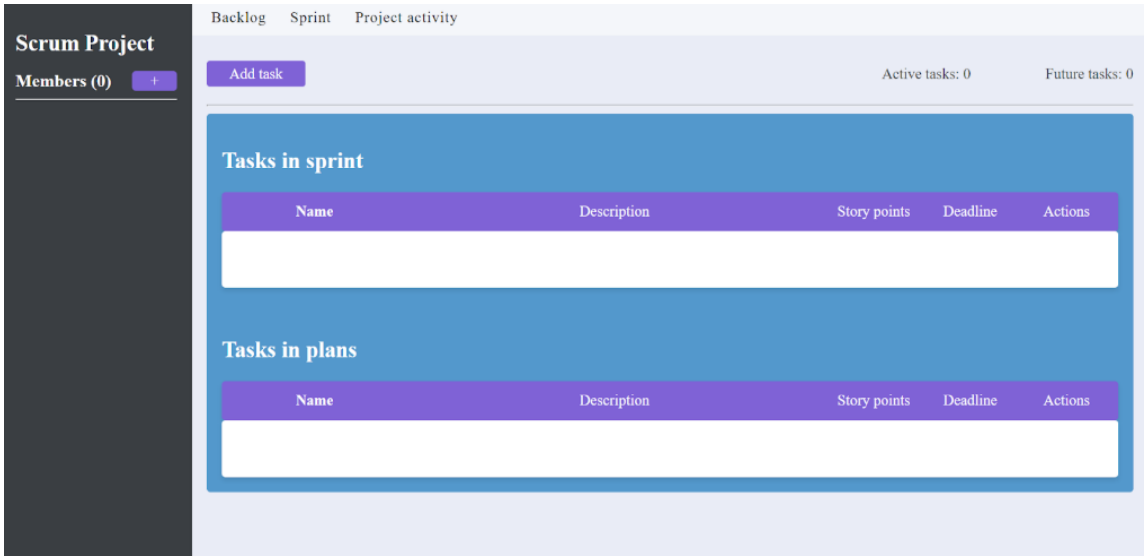


Рисунок 5.4 – сторінка із завданнями

Для того щоб додати користувача потрібно натиснути відповідний елемент інтерфейсу, ввести його ім’я та натиснути кнопку підтвердження як на рисунку 5.5.

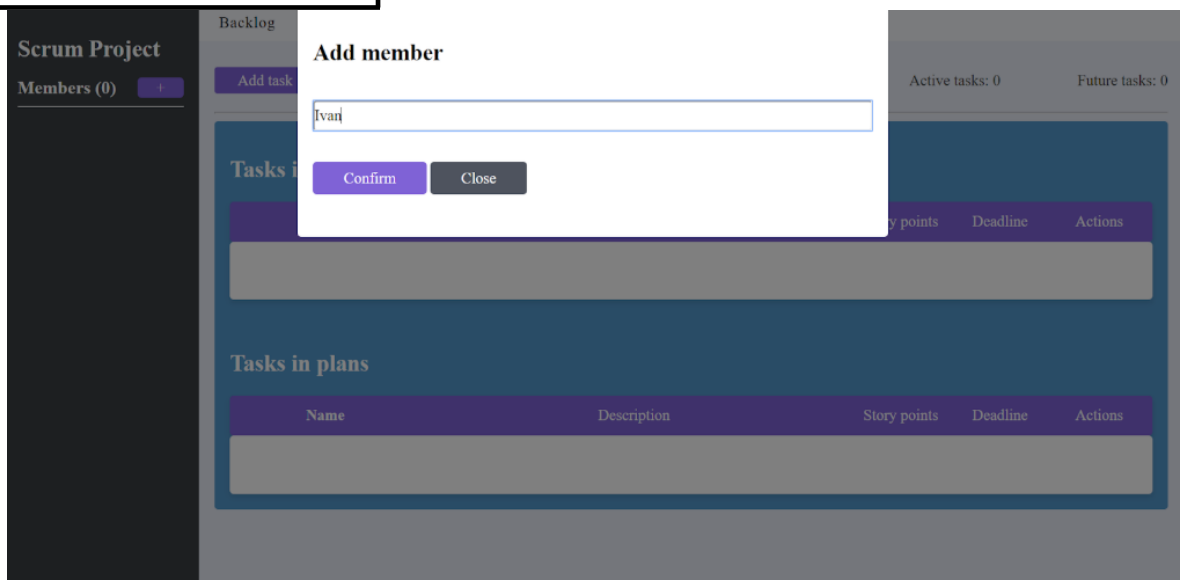


Рисунок 5.5 – додавання користувача

Після додавання користувачі додаються у панель збоку як на рисунку 5.6.

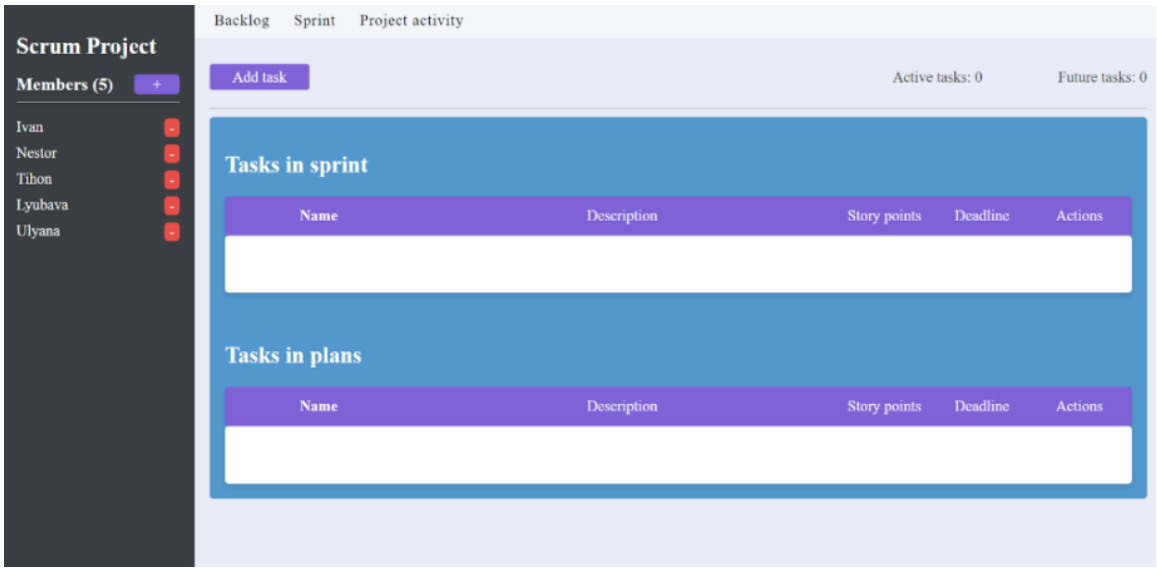


Рисунок 5.6 – вивід списку учасників

На рисунку 5.7 можна побачити сторінку завдань. На сторінці зі списком завдань можна додати нове завдання, якщо натиснути на кнопку додавання завдання. Після натискання з’являється модальне вікно з наступними полями:

- Відповідальний
- Назва завдання
- Опис завдання
- Кількість умовних одиниць витрат часу
- Дата дедлайну

Для додавання завдання потрібно натиснути кнопку підтвердження.

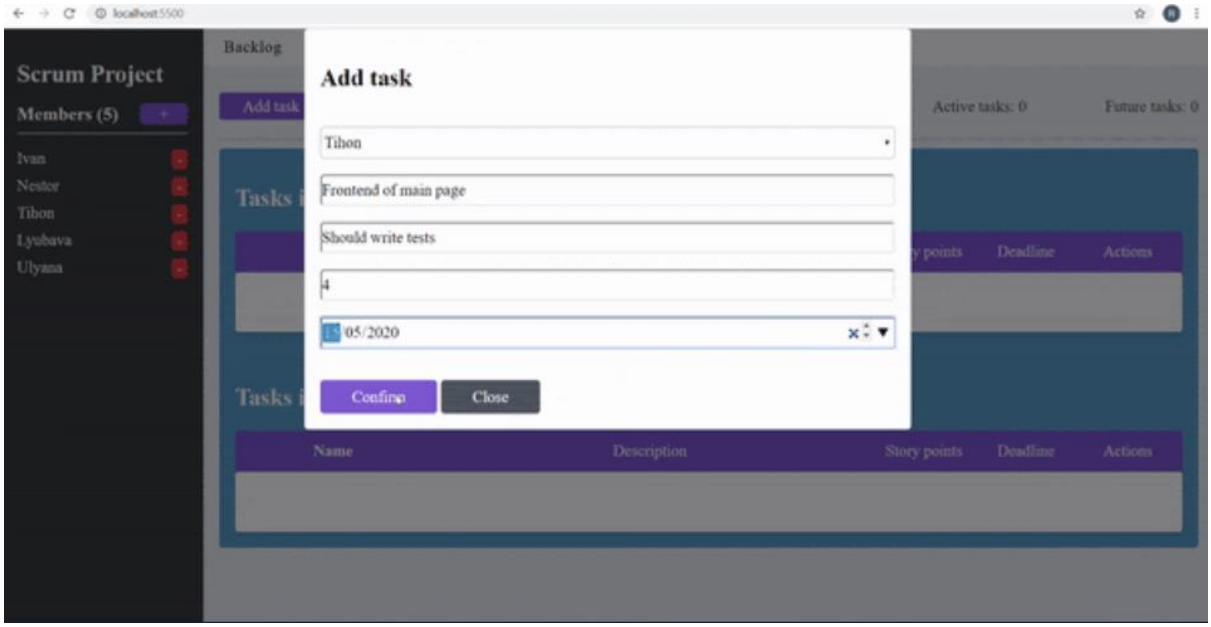


Рисунок 5.7 – додавання завдання

Завдання додається у БД та виводиться у загальному списку. Тепер його можна перетягнути в активний стан (спринт) як на рисунку 5.8

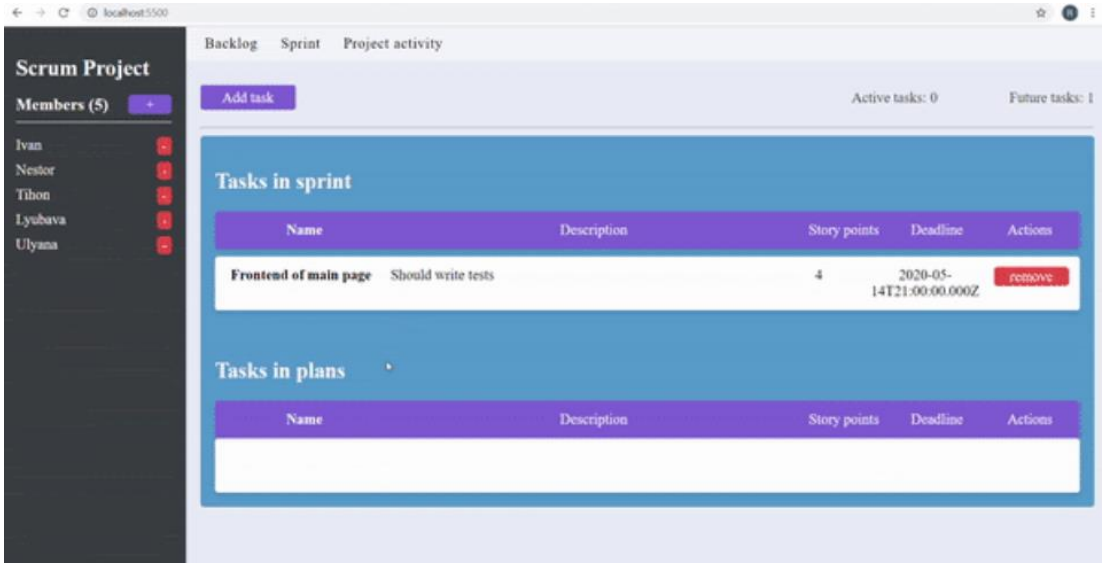


Рисунок 5.8 – Додавання завдання в активні

На сторінці активних проектів можна побачити усі проекти, що були додані у спринт. Тепер можна міняти стан завдання на один з трьох: “В планах”, “В розробці”, “Виконано”. Для цього треба перетягнути завдання з однієї колонки у іншу як на рисунку 5.9. Також у картці завдання можна побачити основну інформацію по ньому:

– Хто відповідальний

– Дату дедлайну
– Назву
– Опис

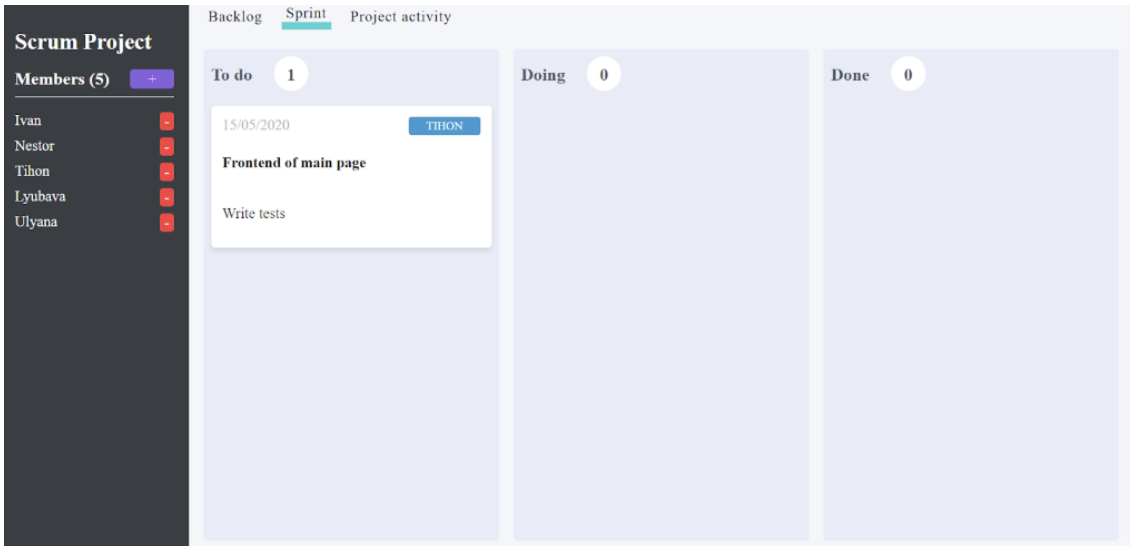


Рисунок 5.9 – вивід сторінки активних завдань

Коли завдання змінює свій стан, інформація про це записується у історію проекта. Рисунок 5.10 демонструє сторінку історії змін в проекті.

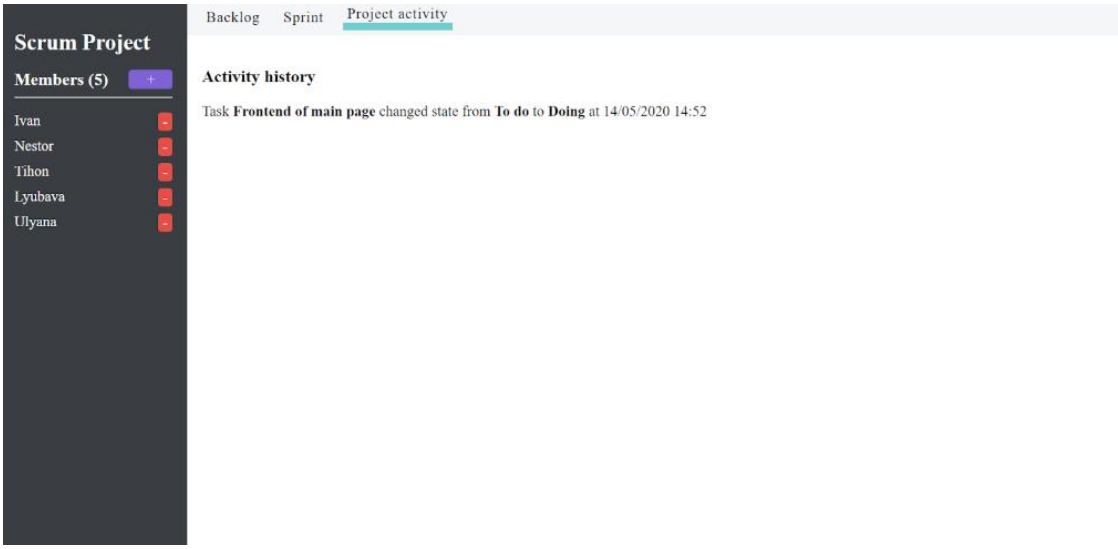


Рисунок 5.10 – історія змін в проекті

5.2 Випробування програмного продукту

В цьому розділі буде наведено опис тестів, які дозволять забезпечити відповідність розробленого програмного продукту попередньо виставленим функціональним вимогам.

5.2.1 Мета випробувань

Метою випробувань являється перевірка відповідності програмної реалізації дипломного проекту вимогам технічного завдання.

5.2.2 Загальні положення

Випробування проводяться на основі наступних документів:

– ГОСТ 34.603–92. Інформаційна технологія. Види випробувань автоматизованих систем;
– ГОСТ РД 50-34.698-90. Автоматизовані системи вимог до змісту документів.

5.2.3 Результати випробувань

В процесі тестування було перевірено відповідність програмної реалізації комплексу задач вимогам технічного завдання. Перелік випробувань основних функціональних можливостей програмної реалізації дипломного проекту наведено у таблицях 5.1 – 5.10.

Таблиця 5.1 – тестування функції «Реєстрація у системі»

Мета тесту:	Перевірка функції “Реєстрація у системі”
Початковий стан КЗ:	Відкрита сторінка регістрації
Вхідні дані:	пошта, назва проекту
Схема проведення тесту:	Заповнити поля з поштою та назвою проекта. Натиснути кнопку реєстрації
Очікуваний результат:	Надходження uuid проекту
Стан КЗ після проведення випробування	Надходження uuid проекту

Таблиця 5.2 – тестування функції «Авторизація у системі»

Мета тесту:	Перевірка функції “Авторизація у системі”
Початковий стан КЗ:	Відкрита сторінка авторизації
Вхідні дані:	uuid проекту
Схема проведення тесту:	Ввести uuid. Натиснути кнопку авторизації
Очікуваний результат:	Перехід на головну сторінку застосунку
Стан КЗ після проведення випробування	Перехід на головну сторінку застосунку

Таблиця 5.3 – тестування функції «Додавання користувача»

Мета тесту:	Перевірка функції “Додавання користувача”
Початковий стан КЗ:	Відкрита головна сторінка
Вхідні дані:	Ім’я користувача
Схема проведення тесту:	Натиснути кнопку додавання користувача, ввести його ім’я. Підтвердити додавання.
Очікуваний результат:	Виведення нового користувача у панелі.
Стан КЗ після проведення випробування	Виведення нового користувача у панелі.

Таблиця 5.4 – тестування функції «Видалення користувача»

Мета тесту:	Перевірка функції “Видалення користувача”
Початковий стан КЗ:	Відкрита головна сторінка
Вхідні дані:	Список користувачів
Схема проведення тесту:	Натиснути кнопку видалення навпроти користувача.
Очікуваний результат:	Виведення нового стану користувачів у панелі.
Стан КЗ після проведення випробування	Виведення нового стану користувачів у панелі.

Таблиця 5.5 – тестування функції «Додавання завдання»

Мета тесту:	Перевірка функції “Додавання завдання”

Початковий стан КЗ:	Відкрита сторінка завдань
Вхідні дані:	Ім’я відповідального, назва завдання, опис, кількість сторі поінтів, дедлайн
Схема проведення тесту:	Натиснути кнопку додавання завдання. Обрати ім’я відповідального, ввести назву завдання, опис, кількість сторі поінтів та дедлайн. Підтвердити додавання.
Очікуваний результат:	Виведення нового завдання у списку завдань.
Стан КЗ після проведення випробування	Виведення нового завдання у списку завдань.

Таблиця 5.6 – тестування функції «Додавання завдання в активні»

Мета тесту:	Перевірка функції “Додавання завдання в активні”
Початковий стан КЗ:	Відкрита сторінка завдань
Вхідні дані:	Список завдань
Схема проведення тесту:	Перетягнути завдання в рядок активних завдань
Очікуваний результат:	Виведення нового завдання у списку активних завдань.
Стан КЗ після проведення випробування	Виведення нового завдання у списку активних завдань.

Таблиця 5.7 – тестування функції «Зміна стану завдання»

Мета тесту:	Перевірка функції “Зміна стану завдання”
Початковий стан КЗ:	Відкрита сторінка активних завдань
Вхідні дані:	Список завдань
Схема проведення тесту:	Перетягнути завдання в стовпчик іншого стану
Очікуваний результат:	Виведення завдання в новому стані
Стан КЗ після проведення випробування	Виведення завдання в новому стані

Таблиця 5.8 – Тестування функції «Перехід на сторінку активних завдань»

Мета тесту:	Перевірка функції “Перехід на сторінку активних завдань”
Початковий стан КЗ:	Відкрита сторінка завдань
Вхідні дані:	Список сторінок
Схема проведення тесту:	Натиснути на кнопку активних завдань
Очікуваний результат:	Виведення активних завдань
Стан КЗ після проведення випробування	Виведення активних завдань

Таблиця 5.9 – тестування функції «Перехід на сторінку історії змін»

Мета тесту:	Перевірка функції “Перехід на сторінку історії змін”
Початковий стан КЗ:	Відкрита сторінка завдань
Вхідні дані:	Список сторінок
Схема проведення тесту:	Натиснути на кнопку історії змін
Очікуваний результат:	Виведення історії змін
Стан КЗ після проведення випробування	Виведення історії змін

Таблиця 5.10 – тестування функції «Перехід на сторінку завдань»

Мета тесту:	Перевірка функції “Перехід на сторінку завдань”
Початковий стан КЗ:	Відкрита сторінка активних завдань
Вхідні дані:	Список сторінок
Схема проведення тесту:	Натиснути на кнопку завдань
Очікуваний результат:	Виведення завдань
Стан КЗ після проведення випробування	Виведення завдань

Висновок до розділу

Було протестовано послідовність дій під час використання застосунку. Кожен тест мав свою мету, початковий стан, вхідні дані, схему проведення, очікуваний результат та стан КЗ після проведення випробувань. Отримані результати коректні та успішні.

ЗАГАЛЬНІ ВИСНОВКИ

У даному дипломному проекті було описано предметне середовище інформаційної системи управління командою розробників. Ця система створена для автоматизації процесу визначення можливої кількості витрат часу на виконання завдання, що було поставлене замовником. Було описано процес діяльності до та після автоматизації.

Також було оглянуто популярні існуючі аналоги. Основним їх недоліком є відсутня можливість модифікації через закритість вихідного коду та необхідність ведення backlog у сторонніх програмних продуктах.

Структура бази даних складається з таблиці з даними для навчання наївного баєсіва класифікатора, таблиці з проектами, користувачами та завданнями.

У математичному забезпеченні використовується наївний баєсів класифікатор. Також через розповсюдженість ефективних багатоядерних комп’ютерів було розроблено паралельну модель, що дозволило використовувати потужності обчислювальної системи з максимальною віддачою. Експерименти довели корисність використання такого підходу.

Застосунок побудований на мікросервісній архітектурі. Є клієнт, що написаний на react (html, css, js). А також rest api сервер, де за бізнес логіку відповідає node.js фреймворк fastify, а за ресурсоємні багатопроцесорні математичні обчислення - golang.

Також було створено керівництво користувача та проведені випробування програмного продукту.

ПЕРЕЛІК ПОСИЛАНЬ

1. Domingos, Pedro & Michael Pazzani (1997) «On the optimality of the simple Bayesian classifier under zero-one loss». Machine Learning
2. Rish, Irina. (2001). «An empirical study of the naive Bayes classifier». IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence.
3. Webb, G. I.; Boughton, J.; Wang, Z. (2005). "Not So Naive Bayes: Aggregating One-Dependence Estimators". Machine Learning.
4. Mozina, M.; Demsar, J.; Kattan, M.; Zupan, B. (2004). Nomograms for Visualization of Naive Bayesian Classifier.
5. Maron, M. E. (1961). "Automatic Indexing: An Experimental Inquiry". Journal of the ACM.
6. Minsky, M. (1961). Steps toward Artificial Intelligence.
7. McCallum, A. and Nigam K. «A Comparison of Event Models for Naive Bayes Text Classification»
8. Hand, DJ, & Yu, K. (2001). «Idiot’s Bayes — not so stupid after all?» International Statistical Review.
9. Математические методы обучения по прецедентам (теория обучения машин). К. В. Воронцов
10. Комп’ютерна лінгвістика : Методичні вказівки до виконання лабораторних робіт для студентів спеціальностей. доц. Фіногенов Олексій Дмитрович
11. https://www.reactjs.org документація до фронтенд фреймворку react
12. https://www.fastify.io/ документація до бекенд фреймворку fastify
13. https://www.postgresql.org/ документація до СКБД Postgres
14. https://golang.org/ документація до мови програмування golang
15. https://nodejs.org/ документація до середовища виконання javascript
16. https://developer.mozilla.org/ru/docs/Web/API/Document документація до мови програмування javascript

Додаток А

Тексти програмного коду

Інформаційна система підтримки діяльності дитячого хору

(Найменування програми (документа))

DVD-R

(Вид носія даних)

33 арк, 156 Кб

(Обсяг програми (документа) , арк.,) Кб)

Київ – 2020 року

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './components/App';

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById('root')
);
/*! normalize.css v8.0.1 | MIT License | github.com/necolas/normalize.css */

/* Document

============================================================
=============== */

/**
 * 1. Correct the line height in all browsers.
 * 2. Prevent adjustments of font size after orientation changes in iOS.
 */

html {
  line-height: 1.15;
  /* 1 */
  -webkit-text-size-adjust: 100%;
  /* 2 */
}

/* Sections

============================================================
=============== */

/**
 * Remove the margin in all browsers.
 */

body {
  margin: 0;
}

/**
 * Render the `main` element consistently in IE.
```

```
 */

main {
  display: block;
}

/**
 * Correct the font size and margin on `h1` elements within `section` and
 * `article` contexts in Chrome, Firefox, and Safari.
 */

h1 {
  font-size: 2em;
  margin: 0.67em 0;
}

/* Grouping content

==============================================================
=============== */

/**
 * 1. Add the correct box sizing in Firefox.
 * 2. Show the overflow in Edge and IE.
 */

hr {
  box-sizing: content-box;
  /* 1 */
  height: 0;
  /* 1 */
  overflow: visible;
  /* 2 */
}

/**
 * 1. Correct the inheritance and scaling of font size in all browsers.
 * 2. Correct the odd `em` font sizing in all browsers.
 */

pre {
  font-family: monospace, monospace;
  /* 1 */
  font-size: 1em;
  /* 2 */
```

```
}

/* Text-level semantics

==============================================================
=============== */

/**
 * Remove the gray background on active links in IE 10.
 */

a {
  background-color: transparent;
}

/**
 * 1. Remove the bottom border in Chrome 57-
 * 2. Add the correct text decoration in Chrome, Edge, IE, Opera, and Safari.
 */

abbr[title] {
  border-bottom: none;
  /* 1 */
  text-decoration: underline;
  /* 2 */
  text-decoration: underline dotted;
  /* 2 */
}

/**
 * Add the correct font weight in Chrome, Edge, and Safari.
 */

b, strong {
  font-weight: bolder;
}

/**
 * 1. Correct the inheritance and scaling of font size in all browsers.
 * 2. Correct the odd `em` font sizing in all browsers.
 */

code, kbd, samp {
  font-family: monospace, monospace;
  /* 1 */
```

```
  font-size: 1em;
  /* 2 */
}

/**
 * Add the correct font size in all browsers.
 */

small {
  font-size: 80%;
}

/**
 * Prevent `sub` and `sup` elements from affecting the line height in
 * all browsers.
 */

sub, sup {
  font-size: 75%;
  line-height: 0;
  position: relative;
  vertical-align: baseline;
}

sub {
  bottom: -0.25em;
}

sup {
  top: -0.5em;
}

/* Embedded content

============================================================
=============== */

/**
 * Remove the border on images inside links in IE 10.
 */

img {
  border-style: none;
}
```

```
/* Forms

========================================================================== */

/**
 * 1. Change the font styles in all browsers.
 * 2. Remove the margin in Firefox and Safari.
 */

button, input, optgroup, select, textarea {
  font-family: inherit;
  /* 1 */
  font-size: 100%;
  /* 1 */
  line-height: 1.15;
  /* 1 */
  margin: 0;
  /* 2 */
}

/**
 * Show the overflow in IE.
 * 1. Show the overflow in Edge.
 */

button, input {
  /* 1 */
  overflow: visible;
}

/**
 * Remove the inheritance of text transform in Edge, Firefox, and IE.
 * 1. Remove the inheritance of text transform in Firefox.
 */

button, select {
  /* 1 */
  text-transform: none;
}

/**
 * Correct the inability to style clickable types in iOS and Safari.
 */
```

```
button, [type="button"], [type="reset"], [type="submit"] {
  -webkit-appearance: button;
}

/**
 * Remove the inner border and padding in Firefox.
 */

button::-moz-focus-inner, [type="button"]::-moz-focus-inner, [type="reset"]::-
moz-focus-inner, [type="submit"]::-moz-focus-inner {
  border-style: none;
  padding: 0;
}

/**
 * Restore the focus styles unset by the previous rule.
 */

button:-moz-focusring, [type="button"]:-moz-focusring, [type="reset"]:-moz-
focusring, [type="submit"]:-moz-focusring {
  outline: 1px dotted ButtonText;
}

/**
 * Correct the padding in Firefox.
 */

fieldset {
  padding: 0.35em 0.75em 0.625em;
}

/**
 * 1. Correct the text wrapping in Edge and IE.
 * 2. Correct the color inheritance from `fieldset` elements in IE.
 * 3. Remove the padding so developers are not caught out when they zero out
 *    `fieldset` elements in all browsers.
 */

legend {
  box-sizing: border-box;
  /* 1 */
  color: inherit;
  /* 2 */
  display: table;
  /* 1 */
```

```
  max-width: 100%;
  /* 1 */
  padding: 0;
  /* 3 */
  white-space: normal;
  /* 1 */
}

/**
 * Add the correct vertical alignment in Chrome, Firefox, and Opera.
 */

progress {
  vertical-align: baseline;
}

/**
 * Remove the default vertical scrollbar in IE 10+.
 */

textarea {
  overflow: auto;
}

/**
 * 1. Add the correct box sizing in IE 10.
 * 2. Remove the padding in IE 10.
 */

[type="checkbox"], [type="radio"] {
  box-sizing: border-box;
  /* 1 */
  padding: 0;
  /* 2 */
}

/**
 * Correct the cursor style of increment and decrement buttons in Chrome.
 */

[type="number"]::-webkit-inner-spin-button, [type="number"]::-webkit-outer-spin-
button {
  height: auto;
}
```

```
/**
 * 1. Correct the odd appearance in Chrome and Safari.
 * 2. Correct the outline style in Safari.
 */

[type="search"] {
  -webkit-appearance: textfield;
  /* 1 */
  outline-offset: -2px;
  /* 2 */
}

/**
 * Remove the inner padding in Chrome and Safari on macOS.
 */

[type="search"]::-webkit-search-decoration {
  -webkit-appearance: none;
}

/**
 * 1. Correct the inability to style clickable types in iOS and Safari.
 * 2. Change font properties to `inherit` in Safari.
 */

::-webkit-file-upload-button {
  -webkit-appearance: button;
  /* 1 */
  font: inherit;
  /* 2 */
}

/* Interactive

============================================================
=============== */

/*
 * Add the correct display in Edge, IE 10+, and Firefox.
 */

details {
  display: block;
}
```

```
/*
 * Add the correct display in all browsers.
 */

summary {
  display: list-item;
}

/* Misc

================================================================
=============== */

/**
 * Add the correct display in IE 10+.
 */

template {
  display: none;
}

/**
 * Add the correct display in IE 10.
 */

[hidden] {
  display: none;
}

:root {
  --main-color: #3a3e42;
  --white-color: white;
  --violet: #7f62d6;
  --red: #e74e48;
  --main-background: #f4f6fa;
  --underline-color: #6fcfce;
  --card-block-color: #e9ecf6;
  --card-title-color: #4e535e;
  --card-date-color: #b5b5b5;
  --card-name-color: #5398cc;
  --brown: #e5aa64;
  --green: #4caf50;
  box-sizing: border-box;
  font-size: 20px;
  font-family: Roboto;
```

```
}

*, ::before, ::after {
  box-sizing: inherit;
}
// This optional code is used to register a service worker.
// register() is not called by default.

// This lets the app load faster on subsequent visits in production, and gives
// it offline capabilities. However, it also means that developers (and users)
// will only see deployed updates on subsequent visits to a page, after all the
// existing tabs open on the page have been closed, since previously cached
// resources are updated in the background.

// To learn more about the benefits of this model and instructions on how to
// opt-in, read https://bit.ly/CRA-PWA

const isLocalhost = Boolean(
  window.location.hostname === 'localhost' ||
    // [::1] is the IPv6 localhost address.
    window.location.hostname === '[::1]' ||
    // 127.0.0.0/8 are considered localhost for IPv4.
    window.location.hostname.match(
      /^127(?:\.(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)){3}$/
    )
);

export function register(config) {
  if (process.env.NODE_ENV === 'production' && 'serviceWorker' in navigator) {
    // The URL constructor is available in all browsers that support SW.
    const publicUrl = new URL(process.env.PUBLIC_URL, window.location.href);
    if (publicUrl.origin !== window.location.origin) {
      // Our service worker won't work if PUBLIC_URL is on a different origin
      // from what our page is served on. This might happen if a CDN is used to
      // serve assets; see https://github.com/facebook/create-react-app/issues/2374
      return;
    }

    window.addEventListener('load', () => {
      const swUrl = `${process.env.PUBLIC_URL}/service-worker.js`;

      if (isLocalhost) {
        // This is running on localhost. Let's check if a service worker still exists or
not.
        checkValidServiceWorker(swUrl, config);
```

```
        // Add some additional logging to localhost, pointing developers to the
        // service worker/PWA documentation.
        navigator.serviceWorker.ready.then(() => {
          console.log(
            'This web app is being served cache-first by a service ' +
              'worker. To learn more, visit https://bit.ly/CRA-PWA'
          );
        });
      } else {
        // Is not localhost. Just register service worker
        registerValidSW(swUrl, config);
      }
    });
  }
}

function registerValidSW(swUrl, config) {
  navigator.serviceWorker
    .register(swUrl)
    .then(registration => {
      registration.onupdatefound = () => {
        const installingWorker = registration.installing;
        if (installingWorker == null) {
          return;
        }
        installingWorker.onstatechange = () => {
          if (installingWorker.state === 'installed') {
            if (navigator.serviceWorker.controller) {
              // At this point, the updated precached content has been fetched,
              // but the previous service worker will still serve the older
              // content until all client tabs are closed.
              console.log(
                'New content is available and will be used when all ' +
                  'tabs for this page are closed. See https://bit.ly/CRA-PWA.'
              );

              // Execute callback
              if (config && config.onUpdate) {
                config.onUpdate(registration);
              }
            } else {
              // At this point, everything has been precached.
              // It's the perfect time to display a
              // "Content is cached for offline use." message.
```

```
            console.log('Content is cached for offline use.');

            // Execute callback
            if (config && config.onSuccess) {
              config.onSuccess(registration);
            }
          }
        }
      };
    };
  })
  .catch(error => {
    console.error('Error during service worker registration:', error);
  });
}

function checkValidServiceWorker(swUrl, config) {
  // Check if the service worker can be found. If it can't reload the page.
  fetch(swUrl, {
    headers: { 'Service-Worker': 'script' },
  })
    .then(response => {
      // Ensure service worker exists, and that we really are getting a JS file.
      const contentType = response.headers.get('content-type');
      if (
        response.status === 404 ||
        (contentType != null && contentType.indexOf('javascript') === -1)
      ) {
        // No service worker found. Probably a different app. Reload the page.
        navigator.serviceWorker.ready.then(registration => {
          registration.unregister().then(() => {
            window.location.reload();
          });
        });
      } else {
        // Service worker found. Proceed as normal.
        registerValidSW(swUrl, config);
      }
    })
    .catch(() => {
      console.log(
        'No internet connection found. App is running in offline mode.'
      );
    });
}
```

```
export function unregister() {
  if ('serviceWorker' in navigator) {
    navigator.serviceWorker.ready
      .then(registration => {
        registration.unregister();
      })
      .catch(error => {
        console.error(error.message);
      });
  }
}
import React from 'react';
import {
  BrowserRouter as Router,
  Switch,
  Route
} from "react-router-dom";
import { Authorization } from './../Authorization'
import { Project } from './../Project'

function App() {
  return (
    <Router>
      <Switch>
        <Route path="/project">
          <Project />
        </Route>
        <Route path="/">
          <Authorization />
        </Route>
      </Switch>
    </Router>
  );
}

export default App;
import React, { useState } from 'react'
import AuthorizationView from './AuthorizationView'

const AuthorizationController = () => {
    const [isRegister, setIsRegister] = useState(false)
    const [emailInput, setEmailInput] = useState('')
    const [projectNameInput, setProjectNameInput] = useState('')
    const [projectUuidInput, setProjectUuidInput] = useState('')
```

```
const postProject = async (url = '', data = {}) => {
    const response = await fetch(url, {
        method: 'POST',
        mode: 'cors',
        cache: 'no-cache',
        credentials: 'same-origin',
        headers: {
            'Content-Type': 'application/json'
        },
        redirect: 'follow',
        referrerPolicy: 'no-referrer',
        body: JSON.stringify(data)
    });
    return await response.json();
}

const getProject = async (url = '') => {
    const response = await fetch(url, {
        method: 'GET',
        mode: 'cors',
        cache: 'no-cache',
        credentials: 'same-origin',
        headers: {
            'Content-Type': 'application/json'
        },
        redirect: 'follow',
        referrerPolicy: 'no-referrer',
    });
    return await response.json();
}

const register = () => {
    postProject('http://localhost:5000/project', { name: projectNameInput })
        .then((data) => {
            console.log(`Email sent to ${emailInput}: ${data.project_uuid}`)
            setIsRegister(false)
        });
}

const login = () => {
    getProject(`http://localhost:5000/project/${projectUuidInput}`)
        .then((data) => {
            if (data.hasOwnProperty('name')) {
                console.log(`Project name is ${data.name}`)
```

```
                localStorage.setItem('project_uuid', projectUuidInput)
                localStorage.setItem('name', data.name)
                window.location = 'http://localhost:3000/project'
            } else
                alert('Невірний uuid!')
        });
    }

    return (
        <AuthorizationView
            isRegister={isRegister}
            setIsRegister={setIsRegister}
            setEmailInput={setEmailInput}
            setProjectNameInput={setProjectNameInput}
            register={register}
            setProjectUuidInput={setProjectUuidInput}
            login={login}
        />
    )
}

export default AuthorizationController
import React from 'react'
import './index.css'

const AuthorizationView = ({
    isRegister,
    setIsRegister,
    setEmailInput,
    setProjectNameInput,
    register,
    setProjectUuidInput,
    login
}) => {
    let inputSection
    if (isRegister) {
        inputSection = (
            <div className="register">
                <input type="text" placeholder="Enter email" onChange={event =>
setEmailInput(event.target.value)} />
                <input type="text" placeholder="Enter project name" onChange={event
=> setProjectNameInput(event.target.value)} />
                <button className="get-uuid" onClick={() => register()}>Get
uuid!</button>
            </div>
```

```
        )
    } else {
        inputSection = (
            <div className="login" id="login">
                <input type="text" placeholder="Enter uuid" onChange={event =>
setProjectUuidInput(event.target.value)} />
                <button className="login-button" onClick={() =>
login()}>Enter</button>
            </div>
        )
    }

    return (
        <div className="body">
            <main className="main-auth">
                <div className="top-menu">
                    <button onClick={() => setIsRegister(false)}>Login</button>
                    <button onClick={() => setIsRegister(true)}>Register</button>
                </div>
                {inputSection}
            </main>
        </div>
    )
}

export default AuthorizationView
.body {
    min-height: 100vh;
    background-image: url("../../images/authorization-background.jpg");
    background-position: center;
    /* Center the image */
    background-repeat: no-repeat;
    /* Do not repeat the image */
    background-size: cover;
    /* Resize the background image to cover the entire container */
    display: flex;
    justify-content: flex-end;
    align-items: center;
}

.main-auth {
    height: 80vh;
    width: 50vw;
    background-color: var(--white-color);
    border-radius: 0.25em;
```

```
    margin-right: 5vw;
    padding: 1em;
}

.top-menu {
    height: 5vh;
    display: flex;
    justify-content: flex-start;
    align-items: center;
}

.top-menu button {
    margin: 0;
    letter-spacing: 1px;
    font-weight: 500;
    border: 0;
    background: transparent;
    height: fit-content;
    margin-left: 1em;
}

.top-menu button:hover {
    border-bottom: 0.5em solid var(--underline-color);
}

.login input {
    width: 100%;
    border-radius: 0.25em;
    height: 2em;
}

.login-button {
    margin-top: 2em;
    width: 100%;
    border: 0;
    background: var(--violet);
    height: 2em;
    color: var(--white-color);
    border-radius: 0.25em;
}

.register input {
    width: 100%;
    border-radius: 0.25em;
    height: 2em;
```

```
}

.register input+input {
    margin-top: 2em;
}

.get-uuid {
    margin-top: 2em;
    width: 100%;
    border: 0;
    background: var(--violet);
    height: 2em;
    color: var(--white-color);
    border-radius: 0.25em;
}

.login-hide {
    display: none;
}

.register-hide {
    display: none;
}
export { default as Authorization } from './AuthorizationController'
import React, { useState, useContext } from 'react'
import MemberModalView from './MemberModalView'
import { Context } from './../LeftSideContext'

const MemberModalController = () => {
    const [memberInput, setMemberInput] = useState('')
    const { setModalShow, members, setMembers, removeMember, memberCount,
setMemberCount } = useContext(Context)

    const postMember = async (url = '', data = {}) => {
        const response = await fetch(url, {
            method: 'POST',
            mode: 'cors',
            cache: 'no-cache',
            credentials: 'same-origin',
            headers: {
                'Content-Type': 'application/json'
            },
            redirect: 'follow',
            referrerPolicy: 'no-referrer',
            body: JSON.stringify(data)
```

```
        });
        return await response.json();
    }

    const addMember = () => {
        postMember('http://localhost:5000/member', { name: memberInput,
project_uuid: localStorage.getItem('project_uuid') })
            .then((data) => {
                console.log(data.member_id)
                setMemberCount(memberCount + 1)
                setMembers([
                    ...members,
                    <li className='member-list__member' key={data.member_id}>
                        {data.name}
                        <button className="member-list__remove-member" onClick={()
=> removeMember(data.member_id)}>-</button>
                    </li>
                ])
            });
    }

    return (
        <MemberModalView
            setMemberInput={setMemberInput}
            setModalShow={setModalShow}
            addMember={addMember}
        />
    )
}

export default MemberModalController
import React from 'react'
import './index.css'

const MemberModalView = ({ setMemberInput, setModalShow, addMember })
=> {
    return (
        <div className="left-panel__modal-background">
            <div className="left-panel__modal-body">
                <input type="text" className="left-panel__modal-input"
placeholder="Введіть ім'я учасника" onChange={event =>
setMemberInput(event.target.value)} />
                <button className="left-panel__modal-submit"
onClick={addMember}>Додати учасника</button>
```

```
            <button className="left-panel__modal-submit" onClick={() =>
setModalShow(false)}>Закрити</button>
          </div>
        </div>
    )
}

export default MemberModalView
.left-panel__modal-background {
    position: absolute;
    bottom: 0;
    right: 0;
    top: 0;
    left: 0;
    background-color: rgba(0, 0, 0, 0.35);
}

.left-panel__modal-body {
    width: 40vw;
    height: 40vh;
    background-color: white;
    border-radius: 0.2em;
    margin: 0 auto;
    padding: 1em;
}

.left-panel__modal-input {
    margin-top: 3vh;
    width: 100%;
    border-radius: 0.15em;
}

.left-panel__modal-submit {
    display: block;
    margin: 1em auto;
    padding: 0.5em 1.5em;
    letter-spacing: 1px;
    font-weight: 500;
    border: 0;
    background: rgb(174, 0, 255);
    color: white;
    border-radius: 0.15em;
    height: fit-content;
}
export { default as MemberModal } from './MemberModalController'
```

```
import React from 'react'

export const Context = React.createContext()
import React, { useState, useEffect } from 'react'
import LeftSideView from './LeftSideView'
import { Context } from './LeftSideContext'
import './index.css'

const LeftSideController = () => {
    const [title] = useState(localStorage.getItem('name'))
    const [members, setMembers] = useState([])
    const [memberCount, setMemberCount] = useState(0)
    const [modalShow, setModalShow] = useState(false)

    const deleteMember = async (url = '') => {
        const response = await fetch(url, {
            method: 'DELETE',
            mode: 'cors',
            cache: 'no-cache',
            credentials: 'same-origin',
            headers: {
                'Content-Type': 'application/json'
            },
            redirect: 'follow',
            referrerPolicy: 'no-referrer',
        });
        return await response.json();
    }

    const getMembers = async (url = '') => {
        const response = await fetch(url, {
            method: 'GET',
            mode: 'cors',
            cache: 'no-cache',
            credentials: 'same-origin',
            headers: {
                'Content-Type': 'application/json'
            },
            redirect: 'follow',
            referrerPolicy: 'no-referrer',
        });
        return await response.json();
    }

    const refreshMembers = () => {
```

```
getMembers(`http://localhost:5000/members/${localStorage.getItem('project_uuid'
)}`)
        .then((data) => {
            let membersArray = []
            data.forEach(member => {
                membersArray.push(
                    <li className='member-list__member'
key={member.member_id}>
                        {member.name}
                        <button className="member-list__remove-member"
onClick={() => removeMember(member.member_id)}>-</button>
                    </li>
                )
            });
            setMembers(membersArray)
            setMemberCount(data.length)
        });
    }

    const removeMember = (member_id) => {
        deleteMember(`http://localhost:5000/member/${member_id}`)
        .then((data) => {
            refreshMembers()
        });
    }

    useEffect(() => {
        refreshMembers()
    }, [])

    return (
        <Context.Provider value={{ setModalShow, members, setMembers,
removeMember, memberCount, setMemberCount }}>
        <LeftSideView
            title={title}
            modalShow={modalShow}
            setModalShow={setModalShow}
            memberCount={memberCount}
            members={members}
        />
        </Context.Provider>
    )
}
```

```
export default LeftSideController
import React from 'react'
import { MemberModal } from './MemberModal'

const LeftSideView = ({ title, modalShow, setModalShow, memberCount,
members }) => {
    let modal
    if (modalShow) {
        modal = <MemberModal />
    } else {
        modal = null
    }

    return (
        <aside className="left-panel">
            {modal}
            <h1 className="left-panel__project-title" id="project-title">{title}</h1>
            <div className="left-panel__members">
                <h3 className="left-panel__members-title">Учасники
({memberCount})</h3>
                <button className="left-panel__add-member" id="add-member"
onClick={() => setModalShow(true)}>+</button>
            </div>
            <hr />
            <ul className="member-list" id="member-list">
                {members}
            </ul>
        </aside>
    )
}

export default LeftSideView
.left-panel {
    min-height: 100vh;
    padding: 1em;
    background-color: var(--main-color);
    color: var(--white-color);
}

.left-panel__project-title {
    font-size: 1.5em;
}

.left-panel__members {
    display: flex;
```

```
    flex-wrap: nowrap;
    justify-content: space-between;
}

.left-panel__members-title {
    margin: 0;
}

.left-panel__add-member {
    border: 0;
    border-radius: 0.25em;
    width: 3em;
    background: var(--violet);
    color: white;
}

.member-list {
    list-style: none;
    padding: 0;
}

.member-list__member {
    margin-top: 0.5em;
    display: flex;
    flex-wrap: nowrap;
    justify-content: space-between;
}

.member-list__remove-member {
    border: 0;
    border-radius: 0.25em;
    width: 1em;
    background: var(--red);
    color: white;
}
export { default as LeftSide } from './LeftSideController'
import React from 'react'
import BacklogView from './BacklogView'

const BacklogController = () => {
    return (
        <BacklogView />
    )
}
```

```
export default BacklogController
import React from 'react'
import './index.css'

const BacklogView = () => {
   return (
      <div className="backlog-page">
         <div className="backlog-page__top">
            <button className="backlog-page__add-task" id="add-task">Додати
завдання</button>
            <p className="backlog-page__info">Активні завдання: <span
id="active-tasks">3</span></p>
            <p className="backlog-page__info">Планові завдання: <span
id="future-tasks">3</span></p>
         </div>
         <hr />
         <div className="backlog-page__active">
            <h2 className="backlog-page__title-active">Список активних
завдань</h2>
            <div className="task-title">
               <h4>Назва</h4>
               <p>Опис</p>
               <p>Витрати часу</p>
               <p>Кінцевий термін</p>
               <p>Дії</p>
            </div>
            <div class="tasks" id="backlog-active"></div>
         </div>
         <div class="backlog-page__plans">
            <h2 class="backlog-page__title-active">Список планових
завдань</h2>
            <div class="task-title">
               <h4>Назва</h4>
               <p>Опис</p>
               <p>Витрати часу</p>
               <p>Кінцевий термін</p>
               <p>Дії</p>
            </div>
            <div class="tasks" id="backlog-plans"></div>
         </div>
      </div>
   )
}

export default BacklogView
```

```
.backlog-page {
    padding: 1em;
    background-color: var(--card-block-color);
}

.backlog-page__top {
    display: flex;
    justify-content: flex-end;
    align-items: center;
    color: var(--main-color);
}

.backlog-page__add-task {
    margin: 0;
    margin-right: auto;
    border: 0;
    background: var(--violet);
    height: fit-content;
    border-radius: 0.2em;
    padding: 0.3em 1.5em;
    color: var(--white-color);
}

.backlog-page__info+.backlog-page__info {
    margin-left: 5em;
}

.backlog-page__active {
    padding: 1em;
    background-color: var(--card-name-color);
    border-top-left-radius: 0.25em;
    border-top-right-radius: 0.25em;
}

.backlog-page__title-active {
    color: var(--white-color);
}

.backlog-page__plans {
    padding: 1em;
    background-color: var(--card-name-color);
    border-bottom-left-radius: 0.25em;
    border-bottom-right-radius: 0.25em;
}
```

```
.modal_hide {
    display: none;
}

.modal__modal-body {
    width: 50vw;
    min-height: 40vh;
    background-color: var(--white-color);
    border-radius: 0.25em;
    padding: 1em;
}

.modal__modal-input {
    width: 100%;
    border-radius: 0.25em;
    height: 2em;
    margin-top: 1em;
}

.modal__confirm {
    margin-top: 2em;
    border: 0;
    border-radius: 0.25em;
    background-color: var(--violet);
    color: var(--white-color);
    padding: 0.5em 2em;
}

.modal__close {
    margin-top: 2em;
    border: 0;
    border-radius: 0.25em;
    background-color: var(--card-title-color);
    color: var(--white-color);
    padding: 0.5em 2em;
}

.modal {
    position: absolute;
    width: 100%;
    height: 100%;
    background: rgba(0, 0, 0, 0.5);
    display: flex;
    flex-direction: column;
    justify-content: space-between;
```

```
    align-items: center;
}

.task-title {
    margin-top: 1em;
    color: white;
    background-color: var(--violet);
    padding: 0.75em;
    border-radius: 0.25em;
    box-shadow: 0em 0.2em 0.3em rgba(0, 0, 0, 0.1);
    display: grid;
    grid-template-columns: 2fr 5fr repeat(3, 1fr);
    column-gap: 1em;
}

.task-title * {
    text-align: center;
    margin: 0;
}

.tasks {
    background-color: var(--white-color);
    border-radius: 0.25em;
    box-shadow: 0em 0.2em 0.3em rgba(0, 0, 0, 0.1);
    min-height: 40vh;
}

.task {
    margin-top: 0.5em;
    border-bottom: 1px solid var(--card-name-color);
    padding: 0.75em;
    display: grid;
    grid-template-columns: 2fr 5fr repeat(3, 1fr);
    column-gap: 1em;
}

.task * {
    text-align: center;
    margin: 0;
}

.task__description {
    text-align: justify;
}
```

```
.task__add-to-sprint {
    color: var(--white-color);
    border: 0;
    height: fit-content;
    border-radius: 0.25em;
    padding: 0.;
    background-color: var(--green);
    width: 4em;
    padding: 0.1em 0.1em;
}

.task__remove-task {
    color: var(--white-color);
    border: 0;
    height: fit-content;
    border-radius: 0.25em;
    padding: 0.;
    background-color: var(--red);
    width: 100%;
    padding: 0.1em 0.1em;
}
export { default as Backlog } from './BacklogController'
import React from 'react'
import ProjectActivityView from './ProjectActivityView'

const ProjectActivityController = () => {
    return (
        <ProjectActivityView />
    )
}

export default ProjectActivityController
import React from 'react'
import './index.css'

const ProjectActivityView = () => {
    return (
        <div className="project-activity-page">
            <h3>Activity history</h3>
            <p>Task changed state from to at</p>
        </div>
    )
}

export default ProjectActivityView
```

```
.project-activity-page {
    padding: 1em;
    background-color: var(--white-color);
    border-radius: 0.25em;
    box-shadow: 0em 0.2em 0.3em rgba(0, 0, 0, 0.1);
}
export { default as ProjectActivity } from './ProjectActivityController'
import React from 'react'
import SprintView from './SprintView'

const SprintController = () => {
    return (
        <SprintView />
    )
}

export default SprintController
import React from 'react'
import './index.css'

const SprintView = () => {
    return (
        <div className="sprint-page">
            <div className="card-block">
                <h3 className="sprint-page__title">To do <span className="card-
block__count" id="to-do-count">3</span></h3>
                <div className="card">
                    <div className="card__info">
                        <p className="card__date">12 jun</p>
                        <p className="card__member">Roma</p>
                    </div>
                    <p className="card__description">description</p>
                </div>
            </div>
            <div className="card-block">
                <h3 className="sprint-page__title">Doing <span className="card-
block__count" id="doing-count">3</span></h3>
            </div>
            <div className="card-block">
                <h3 className="sprint-page__title">Done <span className="card-
block__count" id="done-count">3</span></h3>
            </div>
        </div>
    )
}
```

```
export default SprintView
.sprint-page {
    padding: 1em;
    display: grid;
    grid-template-columns: repeat(3, 1fr);
    column-gap: 1em;
}

.card-block {
    background: var(--card-block-color);
    border-radius: 0.25em;
    padding: 0.5em;
}

.sprint-page__title {
    margin: 0;
    color: var(--card-title-color);
}

.card-block__count {
    text-align: center;
    display: inline-block;
    margin-left: 1em;
    padding: 0.5em;
    border-radius: 50%;
    width: 2em;
    height: 2em;
    background: var(--white-color);
}

.card {
    margin-top: 1em;
    background-color: var(--white-color);
    padding: 0.75em;
    border-radius: 0.25em;
    box-shadow: 0em 0.2em 0.3em rgba(0, 0, 0, 0.1);
    display: grid;
    grid-template-rows: 1fr;
}

.card__info {
    display: flex;
    justify-content: space-between;
}
```

```
.card__info p {
    margin: 0;
}

.card__date {
    color: var(--card-date-color)
}

.card__member {
    text-transform: uppercase;
    font-size: 0.8em;
    font-weight: 500;
    color: var(--white-color);
    padding: 0.2em 1.5em;
    background: var(--card-name-color);
    border-radius: 0.25em;
}

.card__description {
    color: var(--main-color);
}
export { default as Sprint } from './SprintController'
import React, { useState } from 'react'
import MainView from './MainView'
import { Backlog } from './Backlog'
import { ProjectActivity } from './ProjectActivity'
import { Sprint } from './Sprint'

const MainController = () => {
    const [currentPage, setCurrentPage] = useState(Backlog)

    const changeCurrentPage = (choice) => {
        if (choice === 'backlog')
            setCurrentPage(Backlog)
        if (choice === 'projectActivity')
            setCurrentPage(ProjectActivity)
        if (choice === 'sprint')
            setCurrentPage(Sprint)
    }

    return (
        <MainView
            currentPage={currentPage}
            changeCurrentPage={changeCurrentPage}
```

```
        />
    )
}

export default MainController
import React from 'react'
import './index.css'

const MainView = ({ currentPage, changeCurrentPage }) => {
    return (
        <main className="main">
            <div className="top-menu">
                <button className="top-menu__nav-button" id="backlog" onClick={()
=> changeCurrentPage('backlog')}>Список завдань</button>
                <button className="top-menu__nav-button" id="sprint" onClick={()
=> changeCurrentPage('sprint')}>Теперішні завдання</button>
                <button className="top-menu__nav-button" id="project-activity"
onClick={() => changeCurrentPage('projectActivity')}>Історія змін</button>
            </div>
            {currentPage}
        </main>
    )
}

export default MainView
.main {
    background: var(--main-background);
    display: grid;
    grid-template-rows: 1fr 15fr;
}

.top-menu {
    color: var(--main-color);
    display: flex;
    justify-content: flex-start;
    align-items: center;
}

.top-menu__nav-button {
    letter-spacing: 1px;
    font-weight: 500;
    border: 0;
    background: transparent;
    height: fit-content;
    margin-left: 1em;
```

```
}

.top-menu__nav-button:hover {
    border-bottom: 0.5em solid var(--underline-color);
}
export { default as Main } from './MainController'

import React from 'react'
import ProjectView from './ProjectView'

const ProjectController = () => {
    return <ProjectView/>
}

export default ProjectController

import React from 'react'
import { LeftSide } from './LeftSide'
import { Main } from './Main'
import './index.css'

const ProjectView = () => {
    return (
        <div className="container">
            <LeftSide />
            <Main />
        </div>
    )
}

export default ProjectView

.container {
    display: grid;
    grid-template-columns: 1fr 5fr;
}
export { default as Project } from './ProjectController'
```

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”
Кафедра автоматизованих систем обробки інформації і управління

УЗГОДЖЕНО	**ЗАТВЕРДЖУЮ**
Керівник проекту	**В.о. завідувача кафедри**
_____ *Коган А.В.*	_____ *О.А. Павлов*
(підпис) (ініціали, прізвище)	(підпис) (ініціали, прізвище)
“__” квітня 2020 р.	“__” квітня 2020 р.

«Інформаційна система підтримки управління командою розробників»

ТЕХНІЧНЕ ЗАВДАННЯ

Шифр *ДП ІС-6309.1081-с.ТЗ*

На 11 сторінках

Київ – 2020 року

					ДП 6309.01.000 ТЗ			
		Прізвище	*Підпис*	*Дата*				
Розроб.		*Ільєнко Р. С.*			Інформаційна система підтримки управління командою розробників	*Літ.*	*Лист*	*Листів*
Перевірив.		*Коган А.В.*						
						КПІ ім. Ігоря Сікорського Каф. АСОІУ Гр. ІС-63		
Н. кон.		*Проскура С.Л.*						
Затв.		*Павлов О. А.*						

Зміст

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ 2
1.1 Повне найменування системи та її умовне позначення 2
1.2 Найменування організації-замовника та організації-учасника робіт 2
1.3 Перелік документів, на підставі яких створюється система 2
1.4 1.4 Планові терміни початку і закінчення роботи зі створення системи . 2
2 ПРИЗНАЧЕННЯ І ЦІЛІ СТВОРЕННЯ СИСТЕМИ 4
2.1 Призначення системи 4
2.2 Цілі створення системи 4
3 ХАРАКТЕРИСТИКА ОБ'ЄКТА АВТОМАТИЗАЦІЇ 5
4 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 6
5 Вимоги до функціональних характеристик 6
5.1 Вимоги до надійності 7
5.2 Вимоги до складу і параметрів технічних засобів 7
6 СТАДІЇ ТА ЕТАПИ РОЗРОБКИ 9
7 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ СИСТЕМИ 10
7.1 Види випробувань 10

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Повне найменування системи та її умовне позначення

Повна назва системи: Інформаційна система підтримки управління командою розробників.

Коротке найменування: «Scrum App».

1.2 Найменування організації-замовника та організації-учасника робіт

Замовником є кафедра автоматизованих систем обробки інформації та управління Національного технічного університету України "Київський політехнічний інститут ім. Ігоря Сікорського" (далі за текстом — Замовник). Адреса замовника: м. Київ, п. Перемоги 37, 18 корпус ФІОТ.

Розробник сервісу — студент групи ІС-63 кафедри автоматизованих систем обробки інформації та управління Національного технічного університету України "Київський політехнічний інститут ім. Ігоря Сікорського" Ільєнко Роман Сергійович.

1.3 Перелік документів, на підставі яких створюється система

При розробці системи і створення проектно-експлуатаційної документації Виконавець повинен керуватися вимогами наступних нормативних документів:

- ДСТУ 19.201-78. Технічне завдання. Вимоги до змісту і оформлення;
- ДСТУ 34.601-90. Комплекс стандартів на автоматизовані системи. Автоматизовані системи. Стадії створення;
- ДСТУ 34.201-89. Інформаційні технології. Комплекс стандартів на автоматизовані системи. Види, комплексність і позначення документів при створенні автоматизованих систем.

1.4 Планові терміни початку і закінчення роботи зі створення системи

Плановий термін початку роботи над створенням інформаційної системи підтримки управління командою розробників – 1 грудня 2019 року.

Плановий термін по закінченню роботи над створенням інформаційної системи підтримки управління командою розробників – 22 травня 2020 року.

2 ПРИЗНАЧЕННЯ І ЦІЛІ СТВОРЕННЯ СИСТЕМИ

2.1 Призначення системи

Інформаційна система призначена для підтримки управління командою розробників.

2.2 Цілі створення системи

Цілями розробки системи є:

- Реалізація паралельного алгоритму наївного баєсіва класифікатора. Автоматичне визначення витрат часу на завдання на основі НБК.
- Реалізація веб-застосунку керування станами завдань.

Для досягнення поставлених цілей необхідно реалізувати наступні задачі:

- Реалізувати послідовний наївний баєсів класифікатор.
- Розробити паралельний алгоритм наївного баєсіва класифікатора.
- Реалізувати паралельний алгоритм наївного баєсіва класифікатора.
- Створити навчальну вибірку для НБК.
- Розробити схему бази даних.
- Реалізувати базу даних.
- Створити дизайн веб-застосунку.
- Написати бекенд частину веб-застосунку.
- Написати фронтенд частину веб-застосунку.

3 ХАРАКТЕРИСТИКА ОБ'ЄКТА АВТОМАТИЗАЦІЇ

Для того, щоб користуватися системою, користувач повинен зареєструвати проект та перейти до нього по унікальному ідентифікатору.

Об'єктами автоматизації є менеджмент завдань команди розробників та визначення тривалості виконання завдання.

4 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Вимоги до функціональних характеристик

Інформаційна система повинна надавати засоби керування завданнями усім членам команди.

Система має виконувати такі функції:

- Реєстрація в системі:
  - Система повинна надати користувачу можливість ввести свою пошту та назву проекту, після чого ідентифікатор прийде на пошту.
- Авторизація в системі:
  - Система повинна надати користувачу можливість ввести унікальний ідентифікатор проекту та перейти до нього.
- Додавання учасників до системи
- Додавання завдань
- Визначення тривалості виконання завдання:
  - Система повинна автоматично заходити скільки часу знадобиться на виконання певного завдання
- Зміна стану завдання:
  - Система повинна надавати можливість перенести завдання у стан «У планах»
  - Система повинна надавати можливість перенести завдання у стан «В роботі»
  - Система повинна надавати можливість перенести завдання у стан «Виконане»
- Робота с беклогом:
  - Система повинна надавати можливість додати завдання у спринт.

- Система повинна надавати можливість видалити завдання зі спринту.

Система повинна мати функцію відстежування зміни станів завдань та виведення історії по проекту.

4.2 Вимоги до надійності

Система повинна функціонувати безвідмовно незважаючи на наявність ймовірних дефектів, які можуть проявлятися під час експлуатації. Виправлення таких дефектів повинно виконуватися на етапах розробки, бета-тестування та в процесі введення в дію і в межах промислової експлуатації.

Відмова програмного забезпечення сервісу не повинна призводити до руйнувань даних в інформаційних сховищах.

Будь-які аварійні ситуації повинні бути негайно задокументовані і надіслані розробникам задля усунення причин збою в роботі системи.

4.3 Вимоги до складу і параметрів технічних засобів

Коректне функціонування даного програмного продукту на стороні користувача може бути гарантоване тільки за наступних умов:

- наявність комп’ютера з наступними технічними характеристиками:
  1) процесор з тактовою частотою не менше 1 ГГц;
  2) об’єм оперативної пам’яті не менше 1 ГБ;
  3) підключення до мережі інтернет;
  4) інші частини комп’ютеру можуть мати будь-які параметри, так як не мають значного впливу.
- комп'ютерна периферія, до складу якої входить:
  1) монітор;
  2) мишка;
  3) клавіатура;
- встановлений один з веб-браузерів:

1) Google Chrome 80+
2) Firefox 75+
3) Safari 12+

5 СТАДІЇ ТА ЕТАПИ РОЗРОБКИ

Основні етапи виконання робіт з розробки інформаційної системи підтримки управління командою розробників.

№ з/п	Назва етапу роботи	Термін виконання етапу	Результат виконання
1.	Підготовка технічного завдання на розробку програмного продукту	01.12.20119	
2.	Розробка сценарію роботи	13.12.2019	
3.	Технічне проектування – функціональність, модулі, задачі, цілі тощо	24.12.2019	
4.	Узгодження з керівником інтерфейсу користувача	10.01.2020	
5.	Розробка інформаційного забезпечення	20.01.2020	
6.	Розробка програмного забезпечення	07.02.2020	
7.	Налагодження програми	07.04.2020	
8.	Тестування програми	17.04.2020	
9.	Здача готового програмного продукту замовнику	22.05.2020	

6 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ СИСТЕМИ

6.1 Види випробувань

Задля перевірки правильності роботи програмного продукту буде проведено функціональне тестування. В ході тестування буде виконано перевірку всіх функціональних характеристик веб-застосунку. Також, система буде перевірена на відмовостійкість шляхом виконання некоректних дій користувачем. Окремими тестами буде перевірена безпека системи в цілому.

UNICHECK

Власник документу:
Попенко Володимир Дмитрович

ID перевірки:
1003882088

Дата перевірки:
08.06.2020 18:25:01 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
09.06.2020 16:58:57 EEST

ID користувача:
77149

Назва документу: Iljenko_bachelor_is63

ID файлу: 1003896975 **Кількість сторінок:** 42 **Кількість слів:** 6155 **Кількість символів:** 50454 **Розмір файлу:** 1.84 MB

13.2% Схожість

Найбільша схожість: 3.62% з джерело бібліотеки. **ID файлу:** 1003798194

5.46% Схожість з Інтернет джерелами 51 ..... Page 44

12.6% Текстові збіги по Бібліотеці акаунту 230 ..... Page 44

0.26% Цитат

Цитати 2 ..... Page 45

Вилучення переліку посилань вимкнено

0% Вилучень

Вилучений текст відсутній

Підміна символів

Заміна символів 2

АНОТАЦІЯ

Структура та обсяг роботи. Пояснювальна записка дипломного проєкту складається з шести розділів, містить 18 рисунків, 15 таблиць, 1 додаток, 16 джерел.

Дипломний проєкт присвячений розробці інформаційної системи підтримки управління командою розробників. Цілями розробки є реалізація паралельного алгоритму наївного баєсіва класифікатора, автоматичне визначення витрат часу на завдання на основі НБК, реалізація веб-застосунку керування станами завдань.

Для реалізації додатку потрібно вирішити такі задачі: реалізувати послідовний наївний баєсів класифікатор, розробити паралельний алгоритм наївного баєсіва класифікатора, реалізувати паралельний алгоритм наївного баєсіва класифікатора, створити навчальну вибірку для НБК, розробити схему бази даних, реалізувати базу даних, створити дизайн веб-застосунку, написати бекенд частину веб-застосунку, написати фронтенд частину веб-застосунку.

У розділі інформаційного забезпечення було визначено перелік вхідних та вихідних даних, розглянуто та описано структуру бази даних.

Розділ математичного забезпечення присвячений розробці паралельної моделі наївного баєсіва класифікатора.

Програмне забезпечення було описано діаграмами послідовності та компонентів, таблицею з переліком основних функції застосунку.

У технологічному розділі демонструється керівництво користувача для роботи із застосунком, а також випробування програмного продукту.

НАЇВНИЙ БАЄСІВ КЛАСИФІКАТОР, ЗАВДАННЯ, КІНЦЕВИЙ ТЕРМІН ЗДАЧІ, НАВЧАЛЬНА ВИБІРКА

Схожість Цитати Посилання Коментар Підміна символів

Джерел на сторінці: 1-5, 7-13, 15-16, 19, 22-24, 29-38, 41-42, 44-45, 48, 53, 56-57, 68

ABSTRACT

Structure and scope of work. The explanatory note of the diploma project consists of six sections, contains 16 figures, 15 tables, 1 application, 16 sources.

The diploma project is devoted to the Information System for Development Team Management. The objectives of the development are the implementation of a parallel algorithm of naïve Bayes classifier, automatic determination of time spent on tasks based on the NBC, the implementation of a web application for managing task states.

To implement the application the following tasks need to be done: implement a consistent naïve Bayes classifier, develop a parallel algorithm for naïve Bayes classifier, implement a parallel algorithm of naïve Bayes classifier, Create a training sample for the NBC, develop a database schema, implement a database, create a web application design, write the backend part of the web application, write a frontend part of a web application.

In the section of information support the list of input and output data was defined, the structure of the database was considered and described.

The section of mathematical software is devoted to the development of a parallel model of naive classifier and comparison of its speed with a sequential version.

The software was described by sequence and component diagrams, a table with a list and information about the main functions of the software.

The technology section shows the user manual for working with the application, as well as testing the software product.

NAIVE BAYES CLASSIFIER, TASKS, DEADLINE, STUDY SAMPLING

ЗМІСТ

ВСТУП.....4

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ.....5

1.1 Опис предметного середовища.....5

1.1.1 Опис процесу діяльності.....6

1.1.2 Опис функціональної моделі.....8

1.2 Огляд наявних аналогів.....9

1.3 Постановка задачі.....10

1.3.1 Призначення розробки.....10

1.3.2 Цілі та задачі розробки.....11

Висновок до розділу.....11

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ.....12

2.1 Вхідні дані.....12

2.2 Вихідні дані.....12

2.3 Опис структури бази даних.....13

Висновок до розділу.....14

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ.....15

3.1 Змістовна постановка задачі.....15

3.2 Математична постановка задачі.....15

3.3 Обґрунтування методу розв’язання.....16

3.4 Опис методів розв’язання.....17

Висновок до розділу.....19

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ.....20

4.1 Засоби розробки.....20

4.2 Вимоги до технічного забезпечення.....21

4.2.1 Загальні вимоги.....21

4.2.2 Опис локальної обчислювальної мережі.....22

4.3 Архітектура програмного забезпечення.....22

4.3.1 Діаграма послідовності.....23

4.3.2 Діаграма компонентів.....24

4.3.3 Специфікація функцій.....25

Висновок до розділу.....27

5 ТЕХНОЛОГІЧНИЙ РОЗДІЛ.....28

5.1 Керівництво користувача.....28

5.2 Випробування програмного продукту.....33

5.2.1 Мета випробувань.....33

5.2.2 Загальні положення.....33

5.2.3 Результати випробувань.....33

Висновок до розділу.....39

ЗАГАЛЬНІ ВИСНОВКИ.....40

ПЕРЕЛІК ПОСИЛАНЬ.....41

ДОДАТОК А.....42

ВСТУП

В сучасному світі складність технічних проектів, що з розвитком інтернет технологій охоплюють все більшу аудиторію, постійно зростає. Якщо колись невелика команда могла випустити повноцінне програмне забезпечення, то тепер, зі складністю роботи, зростає і розмір команди. Завдання декомпозується на менші частини, передається кожному окремому вузькому спеціалісту. Потреби у якості значно зросли. Тепер недостатньо, щоб застосунок просто працював, він повинен функціонувати при будь-яких умовах, з будь-яким технічним оснащенням і максимальною швидкодією. Також значно зросли вимоги до збереження конфіденційних даних. Кожен маленький шматок коду треба покривати великою кількістю тестів задля зведення до мінімуму неочікуваних ситуацій.

Все це призвело до надзвичайно складного менеджменту проектів у сфері інформаційних технологій. Одним із методів вирішення цього питання є гнучкі методології. Суть більшості із них полягає у створенні спеціальної дошки, що поділена на секції, кожна секція відповідає за певний етап розробки. Уся розробка ділиться на менші частини, визначаються інтервали часу на їх виконання. Це дозволяє зручно контролювати процес роботи, а також вчасно обговорювати із замовником усі зміни.

Як правило, таким процесом керує окремий спеціаліст – проектний менеджер. Він, керуючись своїм досвідом, може визначити кількість витрат часу на окреме завдання і виставляти адекватні кінцеві терміни демонстрації закінчених завдань. Але що робити коли невеликий стартап чи компанія не може собі дозволити ще одне робоче місце, а вимоги до якості програмного продукту такі самі, як і до розробок великих компаній?

Усе це призвело до необхідності розробки інструментів, що автоматично можуть визначати кількість витрат часу. Дана робота використовує методи машинного навчання для прогнозування цього параметру. Вона дозволить маленьким компаніям або стартапам зменшити

необхідність у додаткових витратах на менеджмент, а програмістам сконцентруватися на розробці.

Дипломний проєкт присвячений розробці інформаційної системи підтримки управління командою розробників.

Практичне значення одержаних результатів. Розроблено веб-застосунок для роботи, розроблена та реалізована паралельна модель наївного баєсіва класифікатора.

Публікації. Результати роботи були опубліковані у тезах доповідей на науково-технічних конференціях.

Схожість Цитати Посилання Коментар Підміна символів
Джерел на сторінці: 18, 24, 46

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Опис предметного середовища

Предметне середовище – інформаційна система підтримки управління командою розробників. Однією із найважливіших задач проектного менеджеру є визначення кількості витрат часу які треба надати тій або іншій задачі. Як правило, оцінка тим точніше, чим більше досвіду має спеціаліст. Він розуміє усі тонкощі роботи команди розробників, знає до якого класу важкості можуть відноситись поставленні замовником вимоги, передбачає можливі проблеми. Використовуючи аналіз, з кожним проектои оцінка часу, що буде витрачене на виконання замовлення, має набувати більшою значущості. Але таких менеджерів доволі важко знайти, також не кожна компанія може собі дозволити високооплачуваного фахівця та зайве робоче місце.

Система створена для автоматизації процесу визначення можливої кількості витрат часу на виконання завдання, що було поставлене замовником. Існує дошка з таких розділів: «у планах», «у розробці», «виконане». Кожен розділ відповідає за те, у якому стані знаходиться завдання. Це типова схема роботи гнучких методологій. Спочатку завдання додається у загальний список. Тут треба додати опис завдання, виконавця та кількість умовних одиниць витрат часу. Показник умовних одиниць витрат часу відповідає тому скільки з найбільшої ймовірністю знадобиться часу на виконання цього завдання. Як правило, від команди до команди, він не прив’язаний до конкретної часової міри і може бути як 1 годиною, так і одним днем. Але у даній системі цей показник стало дорівнює 2 годинам, а максимальна кількість у одного завдання - 20. По-перше, такий часовий проміжок дозволить спростити алгоритм передбачення, по-друге, більшість джерел по гнучким методологіям вказують на те, що завдання, які займають маленькі проміжки часу, повинні об’єднуватись в більші, а надто великі

розбиватися на менші, це спрощує процес менеджменту та дозволяє більш точно передбачувати витрати часу. Даний показник повинен автоматично виставлятися після заповнення опису завдання. Функціонал буде працювати на основі розпаралеленого наївного баєсіва класифікатора. Але у разі невідповідності очікуванням, його можна замінити вручну. Під час мануального введення, дані додаються до загальної навчальної вибірки. Після додавання завдання у загальний список, воно може вільно переміщатися між станами.

База даних повинна містити інформацію про проект та завдання. Також вона має бути наповнена навчальною вибіркою для роботи наївного баєсіва класифікатора.

Також існує можливість перегляду повної історії зміни станів завдань в окремому вікні.

1.1.1 Опис процесу діяльності

В даній роботі розглядається автоматизація наступних процесів:

- Менеджмент завдань всередині гнучкої методології.
- Визначення тривалості виконання завдання.

До автоматизації менеджмент завдань під час звичайного процесу здійснюється на звичайній дошці на якій відзначено певні стани завдання. Кожне завдання записується на окремому папірці і кріпиться на цю дошку. Далі воно переміщається власноруч кожним окремим членом команди розробників.

Такий спосіб керування завданнями є зручний, якщо спеціалісти знаходяться в одному приміщенні та бажано недалеко один від одного. Але він зовсім не підходить для розподіленої команди, яка працює дистанційно. До того ж використання великої кількості папірців не є розумним з точки зору охорони екології.

Також тривалість виконання завдання визначалась за допомогою окремого спеціаліста зі значним досвідом або власними зусиллями

розробників. Це приводило до значної кількості проблем. У першому випадку такий фахівець може бути надто витратним, а його пошук, як правило, займає значний час. У другому варіанті амбіціозні програмісти можуть переоцінити власні можливості і зірвати всі терміни. Від цього будуть потерпати як самі виконавці, так і замовники.

Після автоматизація керувати процесом розробки та виконання завдань можна буде дистанційно за допомогою сучасних веб-застосунків, що дозволить організовувати роботу незалежно від фізичного місцерозташування програміста. Також такий спосіб комунікації полегшить збір статистичної інформації для подальшого інтелектуального аналізу та оптимізації процесів діяльності.

Тривалість виконання завдань буде здійснюватися автоматично за допомогою методів машинного навчання. А оцінка буде тим точніше, чим більша навчальна вибірка буде присутня на момент визначення кількості умовних одиниць витрат часу по конкретному завданню.

Користувачу достатньо буде зареєструвати свій проект. Після реєстрації йому на пошту прийде унікальний uuid (стандарт ідентифікації, який використовується при створенні програмного забезпечення, затверджений Open Software Foundation (OSF) як частина Розподіленого комп'ютерного середовища (DCE)). Далі він вводить його у вікно авторизації та отримує доступ до інформаційної системи підтримки управління командою розробників. Тут з’являється можливість інтерактивного керування станами завдань, учасників та інше. Автоматичне визначення тривалості виконання завдання здійснюється після вводу опису задачі у відповідне вікно інтерфейсу програмного веб-застосунку. Таким чином система визначення витрат часу буде працювати швидко і автоматично без залучення окремих спеціалістів.

1.1.2 Опис функціональної моделі

Єдиним актором у системі є користувач - дійова особа в інформаційній системі підтримки управління командою розробників. Може редагувати дані по проекту та завданню.

Тепер визначимо дії, які можуть виконувати актор системи та відобразимо їх за допомогою структурної схеми варіантів використання.

Таблиця 1.1 – Типи варіантів використання

Актор	Варіант використання	Опис дії варіанта використання
Користувач	Зареєструватися	Користувач заповнює поля з поштою та назвою проекта на сторінці реєстрації. Потім натискає кнопку реєстрації.
Користувач	Авторизація	Користувач вводить uuid проекту та натискає кнопку авторизації на сторінці авторизації.
Користувач	Додавання користувача	Користувач натискає кнопку додавання користувача на головній сторінці, вводить ім’я та підтверджу є додавання.
Користувач	Видалення користувача	Користувач натискає кнопку видалення навпроти користувача на головній сторінці.
Користувач	Додавання завдання	Користувач натискає кнопку додавання завдання на сторінці завдань. Обирає ім’я відповідального, вводить назву завдання, опис, кількість story point та дедлайн. Підтверджує

додавання.

Продовження таблиці 1.1

Користувач	Додавання завдання в активні	Користувач за допомогою миші перетягує завдання із списку неактивних завдань в список активних завдань на сторінці списку завдань.
Користувач	Зміна стану завдання	Користувач перетягує завдання в стовпчик іншого завдання на сторінці активних завдань.
Користувач	Перехід на сторінку активних завдань	Користувач натискає на кнопку активних завдань.
Користувач	Перехід на сторінку завдань	Користувач натискає на кнопку завдань
Користувач	Перехід на сторінку історії мін	Користувач натискає на кнопку історії змін

Діаграма варіантів використання наведена у графічному матеріалі

1.2 Огляд наявних аналогів

Гнучка методологія scrum є досить популярною у сфері керування процесу розробки програмного забезпечення. Було знайдено декілька відомих сервісів. Але усі популярні рішення мають закритий вихідний код і не надають змогу модифікації основних елементів, що були задіяні у даному дипломному проекті. Розглянемо один з них. На рисунку 1.1 продемонстровано сторінку власного проекту на сервісі «Trello». Як можна побачити, тут нема можливості ведення повноцінного backlog. А за весь процес організації часу відповідає сам менеджер.

Рисунок 1.1 – Скріншот проекту на сервісі Trello

Також існує такий аналог як «meistertask», основне вікно можна побачити на рисунку 1.2, але він так само не дає змогу ведення story point та не володіє функціоналом передбачення.

Рисунок 1.2 – Скріншот з сервісу «meistertask»

1.3 Постановка задачі

1.3.1 Призначення розробки

Призначення розробки – створення інформаційної системи підтримки управління командою розробників.

1.3.2 Цілі та задачі розробки

Цілями розробки системи є:

- Реалізація паралельного алгоритму наївного баєсіва класифікатора. Автоматичне визначення витрат часу на завдання на основі НБК.
- Реалізація веб-застосунку керування станами завдань.

Для досягнення поставлених цілей необхідно реалізувати наступні задачі:

- Реалізувати послідовний наївний баєсів класифікатор.
- Розробити паралельний алгоритм наївного баєсіва класифікатора.
- Реалізувати паралельний алгоритм наївного баєсіва класифікатора.
- Створити навчальну вибірку для НБК.
- Розробити схему бази даних.
- Реалізувати базу даних.
- Створити дизайн веб-застосунку.
- Написати серверну частину веб-застосунку.
- Написати клієнтську частину веб-застосунку.

Висновок до розділу

В цьому розділі було обґрунтовано доцільність розробки інформаційної системи підтримки управління командою розробників. Було описано процес діяльності до та після автоматизації. Були висунуті функціональні вимоги до системи, сформульовано цілі та задачі розробки даної системи. Також проведено порівняльний аналіз розроблюваної системи та вже існуючих аналогів.

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Вхідні дані

Усі вхідні дані для роботи системи надходять від користувача.

При реєстрації користувач повинен вказати наступні параметри:

- Пошту
- Назву проекту

Пошта потрібна для того, щоб відправляти на неї uuid проекту. А назва проекту буде висвічуватись на головній сторінці.

Основним елементом взаємодії є завдання. Для того, щоб додати завдання треба заповнити такі поля:

- Відповідальний
- Назва
- Опис
- Кількість умовних одиниць витрат часу
- Дата дедлайну

Завдання можна редагувати. Змінювати стан або додавати, видаляти зі списку активних завдань.

Також є можливість додавання учасників проекту, для цього достатньо натиснути на відповідний елемент інтерфейсу та ввести його повне ім’я.

2.2 Вихідні дані

Вихідними даними роботи системи є формування списків завдань, визначення тривалості виконання завдань та історія змін стану завдань.

Завдання представлені наступним переліком параметрів:

- Стан завдання
- Наявність у списку активних завдань
- Назва

- Опис
- Кінцевий термін здачі
- Кількість умовних одиниць витрат часу
- Відповідальний

Тривалість виконання визначається по-замовчуванням та характеризується параметром умовних одиниць витрат часу.

Історія зміни станів завдання характеризується такими параметрами:

- Назва завдання
- Попередній стан
- Новий стан
- Дата зміни стану

2.3 Опис структури бази даних

Під час розробки системи використовувалась реляційна база даних PostgreSQL. У проекті використовується контейнерізована мікросервісна архітектура. Тому БД працює в окремому контейнері Docker. Схема БД наведена у графічних матеріалах.

Наведемо перелік сутностей системи в таблиці 2.1:

Таблиця 2.1 – Перелік сутностей

№	Назва таблиці	Сутність
1	teaching_data	Навчальна вибірка
2	task	Завдання
3	member	Учасник
4	project	проект

В таблиці 2.2 наведено детальний опис таблиць бази даних.

Таблиця 2.2 – Детальний опис таблиць бази даних.

Назва таблиці	Назва стовпця	Тип даних	Опис поля
task	task_id	SERIAL	Первинний ключ
	project_uuid	UUID	uuid проекту

Продовження таблиці 2.2

task	member_id	INTEGER	id відповідального
	name	TEXT	Назва завдання
	description	TEXT	Опис завдання
	story_points	INTEGER	Кількість story point
	deadline	DATE	Дедлайн
	isActive	BOOLEAN	Стан sprint\backlog
	state_id	INTEGER	Номер стану
teaching_data	data_id	SERIAL	Первинний ключ
	text	TEXT	Текст завдання
	story_points	INTEGER	Кількість story point
member	member_id	SERIAL	Первинний ключ
	name	TEXT	Ім’я учасника
	project_uuid	UUID	uuid проекту
project	project_uuid	UUID	Первинний ключ
	name	TEXT	Назва проекту

Висновок до розділу

В даному розділі було визначено перелік вхідних та вихідних даних. Було наведено перелік сутностей, котрі використовуються в системі, із вказанням таблиць, до яких вони відносяться. Також побудовано таблицю, в котрій наведено список таблиць бази даних з детальним описом всіх параметрів.

Схожість Цитати Посилання Коментар Підміна символів

Джерел на сторінці: 28

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Змістовна постановка задачі

Визначення тривалості виконання завдання є одним із найважливіших обов'язків проектного менеджера. Але залучення окремого спеціаліста для цього не завжди є доцільним. Однією із функції веб-застосунку є автоматичне визначення витрат часу на виконання завдання.

Вирішите це питання допомагає наївний баєсів класифікатор. Також він повинен мати паралельну модель, щоб задіяти усі потужності сучасних комп'ютерів.

Користувачу мають змогу вводити лише назву завдання, а програмне забезпечення автоматично за допомогою методів машинного навчання знаходить приблизні витрати часу.

3.2 Математична постановка задачі

Потрібно визначити час виконання певного завдання, тобто виявити до якого класу воно відноситься. Припустимо, що завдання - це документ, а кількість story point - це клас, отже треба визначити приналежність першого до другого. Спочатку будуємо модель класифікатору. Для цього існує навчальна вибірка з D документів, що постійно наповнюється коригуваннями користувачів. У кожного окремого класу с існує D_c документів. k - кількість унікальних слів у документах навчальної вибірки. Також треба підрахувати W_{ic} (скільки разів i-е слово зустрічається в навчальній вибірці с класу). А $\sum_{j=1}^{k}(W_{jc})$ - загальна кількість слів документу с в навчальній вибірці. Володіючи цими даними для кожного класу визначимо значення за формулою 3.1:

$$\log\frac{D_c}{D}+\sum_{i=1}^{n}\log\frac{W_{ic}+1}{k+\sum_{j=1}^{k}(W_{jc})} \qquad (3.1)$$

Схожість Цитати Посилання Коментар Підміна символів

Джерел на сторінці: 1-3, 5-7, 9-10, 14-16, 22-24, 29-32, 34, 36-41, 43-46, 67

Тоді цільова функція задачі матиме вигляд як на формулі 3.2:

$$\log \frac{W_{ic}+1}{k+\sum_{j=1}^{k}(W_{jc})} \qquad (3.2)$$

$$\log \frac{D_c}{D}+\sum_{i=1}^{n} ¿$$

$$C_{map}=¿$$

Тобто задача полягає в наступному: визначити до якого класу належить певне завдання.

3.3 Обґрунтування методу розв’язання

Було обрано наївний баєсів класифікатор через свою простоту у розпаралелюванні. При великих обсягах даних підраховувати синхронно в одному потоці кожний окремий клас може займати доволі великі показники часу. До того ж не використовується усі можливості сучасних багатопроцесорних комп’ютерів.

Вирішити це питання допомагає паралельна модель алгоритму. Порівняльні дані можна побачити на таблиці 3.1.

Таблиця 3.1 – Порівняння прискорення різних моделей алгоритму

К-ть слів	Послідовно	Паралельно	Прискорення
10	1423	34620	0,04110340843
100	23705	37440	0,6331463675
1000	38807	54954	0,7061724351
10000	360087	212068	1,69797895
100000	3099918	2336826	1,326550629
1000000	35943611	22484599	1,598588038
10000000	363140431	242544790	1,497209777

У графічних матеріалах наведено візуалізацію прискорення на певній кількості слів у класі. Видно, що на маленьких об'ємах даних послідовний алгоритм працює швидше за паралельний. Це результат того, що витрати на створення та керування потоками більші ніж витрати на виконання цього завдання послідовно. Але можна побачити як паралельна модель набуває сенсу з масивами близькими до 10000 слів.

3.4 Опис методів розв'язання

Для визначення поставленої задачі, а саме до якого класу належить певне завдання, пропонується використовувати послідовний наївний баєсів алгоритм, що базується на теоремі Баєса, за формулою 3.3:

$$P(c \vee d) = \frac{P(c) * P(d \vee c)}{P(d)} \quad , \tag{3.3}$$

де $P(c \vee d)$ - ймовірність, що документ d належить класу c;

$P(d \vee c)$ - правдоподібність, тобто ймовірність даного значення ознаки при даному класі.

$P(d)$ та $P(c)$ - безумовна ймовірність зустріти документи d та c відповідно в корпусі документів.

В машинному навчанні наївні баєсові класифікатори - це сімейство простих ймовірнісних класифікаторів. Моделі на основі НБА дуже корисні при роботі з великими наборами даних. Метою даного метода є виявлення приналежності документу до певного класу. Так за формулою 3.4 визначається найбільш ймовірний клас:

$$C_{map} = \arg\max_{c \in C} \frac{P(c) * P(d \vee c)}{P(d)} \tag{3.4}$$

Таким чином за формулою 3.5 необхідно для кожного класу підрахувати ймовірність та обрати той, який найбільше підходить.

$$C_{map} = \arg\max_{c \in C} [P(c) * P(d \vee c)] \tag{3.5}$$

Після припущення умовної незалежності, де умовна ймовірність документу може бути представлена як добуток умовних ймовірностей всіх слів, що входять до цього документу, та вирішенню проблеми арифметичного переповнення, використовуючи властивість логарифма добутку $\log(ab)=\log(a)+\log(b)$, враховуючи проблему невідомих слів разом зі згладжуванням Лапласа, кінцева формула набуває вигляду 3.6:

$$\log \frac{W_{ic}+1}{k+\sum_{j=1}^{k}(W_{jc})} \log \frac{D_c}{D}+\sum_{i=1}^{n} ¿ \quad C_{map}=¿ \qquad (3.6)$$

Представимо запропонований алгоритм:

КРОК 1. На вхід подається навчальна вибірка з с кількості класів. В кожному класі с існує D_c документів. Також поступає і документ, клас якого треба знайти.

КРОК 2. Підраховується загальна кількість унікальних слів в усіх класах навчальної вибірки.

КРОК 3. Підраховується кількість документів в усіх класах навчальної вибірки.

КРОК 4. Для кожного класу рахується сумарна кількість слів по всіх документах.

КРОК 4.1. Додається значення логарифмів, де у чисельнику стоїть значення кількості повторень і-го слова з документу, клас якого треба визначити. До цього значення додається 1 (адитивне згладжування). А у знаменнику сума загальної кількості унікальних слів в усіх класах навчальної вибірки та значення кількості слів в цьому класі.

КРОК 4.2. До суми логарифмів додається логарифм, де у чисельнику стоїть кількість документів в цьому класі, а у знаменнику загальна кількість документів.

КРОК 5. Отримуємо деякий набір значень.

КРОК 6. З цього набору максимальне значення належить класу що з найбільшою ймовірністю відповідає даному документу.

Висновок до розділу

Після аналізу наївного баєсіва алгоритму, можна зробити висновок, що даний класифікатор гарно підходить для вирішення задачі визначення тривалості виконання завдання у інформаційній системі підтримки управління командою розробників. Найбільшої результативності він набуває з паралельною схемою роботи при великих об’ємах даних та відповідній кількості процесорів.

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Засоби розробки

Веб-застосунок був реалізований з використанням мов програмування javascript та golang. Він працює за клієнт-серверною архітектурою. Клієнтська частина написана на html, css, javascript та фреймворком react. Бекенд частина складається з двох компонентів: серверу, що написаний на javascript та відповідає за бізнес-логіку, та окремого алгоритму обчислення наївного баєсіва класифікатора, який використовує golang. Хоча проект побудований на мікросервісній архітектурі, засоби розробки обирались так, що і клієнт, і сервер писались на одній мові програмування. Це дозволило сконцентруватися на задачі, оминувши складності використання багатомовної ієрархії. Але тонкі місця, які потребують швидких багатоядерних математичних обчислень, написані на golang. Такий підхід дозволив зменшити показник time to market та швидкість підрахунку алгоритму.

React дозволяє створювати інтерактивні користувацькі інтерфейсі за допомогою веб-технологій. Він використовує компонентний підхід. Кожна окрема сторінка чи блок з інформацією може бути окремим компонентом зі своїм станом. У react стан нагадує приватні властивості або методи всередині класу в ООП. Коли оновлюється певні змінні - компонент заново опрацьовується з новими даними. Інформацію можна брати як з вводу користувача або локального вмістилища, так і з віддаленого сервера. В даному випадку відбувається комунікація з сервером по rest api.

Для статичної верстки використовувалася мова розмітки гіпертекстових документів HTML. Для надання привабливого вигляду застосунку було додано каскадні таблиці стилів CSS.

Javascript - це високорівнева мультипарадигмальна мова програмування, яка може використовуватися як на клієнтській частині, так і на серверній.

Node.js - асинхронний сценарій виконання JavaScript, він призначений для створення масштабованих мережевих додатків.

Fastify - це ефективний фремворк для написання rest api на сервері. Він має потужні архітектуру, що базується на написанні плагінів, та займає одне з перших місць по швидкодії серед усіх інших бекенд javascript фреймворків.

Golang - потужна мова програмування, розроблена google. Дозволяє отримати обчислювальні потужності близькі, а інколи швидші, за C при значно меншому часі розробки. Також ця мова програмування дозволяє надзвичайно ефективно та легко розпаралелити алгоритм під багатоядерну модель при використанні потоків.

4.2 Вимоги до технічного забезпечення

4.2.1 Загальні вимоги

Коректне функціонування даного застосунку на стороні клієнта може гарантуватися за наступних умов:

Операційна система: linux, windows, mac os.

Характеристика комп’ютера:

- процесор з тактовою частотою від 1ГГц
- оперативна пам’ять з об’ємом не менше 1гб
- підключення до інтернету

Комп’ютерна периферія:

- Мишка
- Клавіатура
- Монітор

Браузер:

- Google Chrome 80+
- Firefox 70+

4.2.2 Опис локальної обчислювальної мережі

Додаток працює на мікросервісній архітектурі. Клієнт з сервером спілкуються по REST API. На кожну дію таку як редагування інформації або здійснення аутентифікації написаний окремий endpoint. Сервер в свою чергу спілкується з базою даних за допомогою sql запитів як можна побачити на рисунку 4.1. Також як окремий компонент функціонує додаток для обчислення математичних операцій для підвищення швидкої, викликається за допомогою системних викликів на тій самій машині, де працює сам сервер з бізнес логікою.

Рисунок 4.1 – Демонстрація схеми роботи застосунку

4.3 Архітектура програмного забезпечення

Архітектура програмного забезпечення наведена у графічних матеріалах.

4.3.1 Діаграма послідовності

Діаграма послідовності знаходиться у графічних матеріалах.

У даній діаграмі послідовності показана послідовність користувача. Для її відтворення весь додаток повинен функціонувати коректно.

Під час першого знайомства з додатком користувачу потрібно пройти реєстрацію. Він вводить необхідні дані. Далі вони вносяться у БД. А йому повертається унікальний uuid для подальшої авторизації.

Потім йому треба пройти етап авторизації. Для цього достатньо ввести цей ідентифікатор. Після вдалого проходження аутентифіції, користувач направляється на головну сторінку додатку.

Після потрапляння на головну сторінку, він може додати користувача. Для цього йому необхідно натиснути на відповідну кнопку та заповнити поля з інформацією. Після чого дані надсилаються на сервер та повертається список з новими користувачами. Також користувачів можна видаляти.

Користувачу також надається можливість додавання завдання до списку завдань. Йому потрібно натиснути на відповідний елемент інтерфейсу, додати запропоновані дані та підтвердити відправлення. Завдання надсилається на сервер та додається у БД. Потім воно з'являється у списку завдань на стороні клієнтів. Так само є кнопка для видалення завдання.

Змінити стан завдання можна тільки після додавання його у список теперішніх завдань. Для цього треба перетягнути завдання з загального списку до списку теперішніх. Дані оновлюються в БД та надається змога редагування стану.

Після додавання завдання до теперішніх, можна змінити його стан на "у планах", "в процесі" чи "виконано" шляхом перетягування його у відповідну колонку. Дані оновлюються також на сервері та у БД.

4.3.2 Діаграма компонентів

Діаграму компонентів наведена у графічних матеріалах. З неї видно, що клієнт спілкується з серверов по REST API використовуючи endpoint для певної дії. Сервер в свою чергу надсилає запити та спілкується з базою даних.

4.3.3 Специфікація функцій

Специфікація основних функцій можна побачити у таблиці 4.1

Таблиця 4.1 – Специфікація основних функція

Функція	Опис
checkAuth()	Використовується для перевірки користувача
getProject()	Використовується для отримки інформації про проект
enterUuid()	Використовується для запису даних для авторизації
deleteMember()	Використовується для видалення учасника з проекту
insertMember()	Використовується для додавання користувача
setMembers()	Використовується для виводу користувачів
modalActions()	Використовується для створення модального вікна додавання завдань
addTaskToDB()	Використовується для додавання завдання у базу даних
removeTaskFromDB()	Використовується для видалення завдання з бази даних
addTask()	Використовується для додавання завдання у інтерфейс користувача
setTasksToBacklog()	Використовується для виводу завдань з БД
onDrop()	Використовується для перетягування завдання у теперішній стан

Висновок до розділу

У даному розділі було розглянуто засоби розробки, що були залучені при проектуванні дипломного проекту. Описано їх індивідуальні характеристики. Також прописані вимоги до технічного оснащення для

роботи з даним додатком. Опис локальної обчислювальної системи продемонстрував схему роботи додатки з окремим клієнтом та rest api сервером, що в свою чергу виступає також і провідником з базою даних.

На діаграмі послідовності можна побачити типовий варіант використання застосунку та ілюстрацію взаємодії між користувачем (клієнтом), сервером та БД. Додано діаграму компонентів, що описує механізм взаємодії основних частин додатку.

Для ліпшого розуміння принципів роботи програми, була наведена таблиця з інформацією про основні програмні функції.

5 ТЕХНОЛОГІЧНИЙ РОЗДІЛ

5.1 Керівництво користувача

Для використання застосунку треба відкрити відповідне посилання у вікні браузера. Під час першого переходу виводиться сторінка авторизації. Для того щоб увійти у додаток, треба зареєструватися. Для цього потрібно ввести пошту та назву проекту як на рисунку 5.1.

Рисунок 5.1 – Заповнення реєстраційних даних

Після реєстрації користувач отримує унікальний uuid як на рисунку 5.2.

Рисунок 5.2 – Отримання uuid проекту

Для того щоб авторизуватися треба ввести uuid проекту та натиснути кнопку підтвердження як на рисунку 5.3.

Рисунок 5.3 – введення uuid проекту

Наступним етапом після авторизації є перехід на головну сторінку. На рисунку 5.4 зліва представлена панель для додавання користувачів та три посилання. Сторінка із завданнями, сторінка з активними завданнями та історію змін в проекті.

Рисунок 5.4 – сторінка із завданнями

Для того щоб додати користувача потрібно натиснути відповідний елемент інтерфейсу, ввести його ім’я та натиснути кнопку підтвердження як на рисунку 5.5.

Рисунок 5.5 – додавання користувача

Після додавання користувачі додаються у панель збоку як на рисунку 5.6.

Рисунок 5.6 – вивід списку учасників

На рисунку 5.7 можна побачити сторінку завдань. На сторінці зі списком завдань можна додати нове завдання, якщо натиснути на кнопку додавання завдання. Після натискання з'являється модальне вікно з наступними полями:

- Відповідальний
- Назва завдання
- Опис завдання

- Кількість умовних одиниць витрат часу
- Дата дедлайну

Для додавання завдання потрібно натиснути кнопку підтвердження.

Рисунок 5.7 – додавання завдання

Завдання додається у БД та виводиться у загальному списку. Тепер його можна перетягнути в активний стан (спринт) як на рисунку 5.8

Рисунок 5.8 – Додавання завдання в активні

На сторінці активних проектів можна побачити усі проекти, що були додані у спринт. Тепер можна міняти стан завдання на один з трьох: “В планах”, “В розробці”, “Виконано”. Для цього треба перетягнути завдання з однієї колонки у іншу як на рисунку 5.9. Також у картці завдання можна побачити основну інформацію по ньому:

- Хто відповідальний
- Дату дедлайну
- Назву
- Опис

Рисунок 5.9 – вивід сторінки активних завдань

Коли завдання змінює свій стан, інформація про це записується у історію проекта. Рисунок 5.10 демонструє сторінку історії змін в проекті.

Схожість Цитати Посилання Коментар Підміна символів

Джерел на сторінці: 2-21, 25

Рисунок 5.10 – історія змін в проекті

5.2 Випробування програмного продукту

В цьому розділі буде наведено опис тестів, які дозволять забезпечити відповідність розробленого програмного продукту попередньо виставленим функціональним вимогам.

5.2.1 Мета випробувань

Метою випробувань являється перевірка відповідності програмної реалізації дипломного проекту вимогам технічного завдання.

5.2.2 Загальні положення

Випробування проводяться на основі наступних документів:

- ГОСТ 34.603-92. Інформаційна технологія. Види випробувань автоматизованих систем;
- ГОСТ РД 50-34.698-90. Автоматизовані системи вимог до змісту документів.

5.2.3 Результати випробувань

В процесі тестування було перевірено відповідність програмної реалізації комплексу задач вимогам технічного завдання. Перелік випробувань основних функціональних можливостей програмної реалізації дипломного проекту наведено у таблицях 5.1 – 5.10.

Таблиця 5.1 – тестування функції «Реєстрація у системі»

Мета тесту:	Перевірка функції “Реєстрація у системі”
Початковий стан КЗ:	Відкрита сторінка регістрації
Вхідні дані:	пошта, назва проекту
Схема проведення тесту:	Заповнити поля з поштою та назвою проекта. Натиснути кнопку реєстрації
Очікуваний результат:	Надходження uuid проекту
Стан КЗ після проведення випробування	Надходження uuid проекту

Таблиця 5.2 – тестування функції «Авторизація у системі»

Мета тесту:	Перевірка функції “Авторизація у системі”
Початковий стан КЗ:	Відкрита сторінка авторизації
Вхідні дані:	uuid проекту
Схема проведення тесту:	Ввести uuid. Натиснути кнопку авторизації
Очікуваний результат:	Перехід на головну сторінку застосунку
Стан КЗ після проведення випробування	Перехід на головну сторінку застосунку

Таблиця 5.3 – тестування функції «Додавання користувача»

Мета тесту:	Перевірка функції “Додавання користувача”
Початковий стан КЗ:	Відкрита головна сторінка
Вхідні дані:	Ім’я користувача
Схема проведення тесту:	Натиснути кнопку додавання користувача, ввести його ім’я. Підтвердити додавання.
Очікуваний результат:	Виведення нового користувача у панелі.
Стан КЗ після проведення випробування	Виведення нового користувача у панелі.

Таблиця 5.4 – тестування функції «Видалення користувача»

Мета тесту:	Перевірка функції “Видалення користувача”
Початковий стан КЗ:	Відкрита головна сторінка
Вхідні дані:	Список користувачів
Схема проведення тесту:	Натиснути кнопку видалення навпроти користувача.
Очікуваний результат:	Виведення нового стану користувачів у панелі.
Стан КЗ після проведення випробування	Виведення нового стану користувачів у панелі.

Таблиця 5.5 – тестування функції «Додавання завдання»

Мета тесту:	Перевірка функції “Додавання завдання”
Початковий стан КЗ:	Відкрита сторінка завдань
Вхідні дані:	Ім’я відповідального, назва завдання, опис, кількість сторі поінтів, дедлайн
Схема проведення тесту:	Натиснути кнопку додавання завдання.

Схожість Цитати Посилання Коментар Підміна символів

Джерел на сторінці: 2-21, 25, 62, 77

	Обрати ім’я відповідального, ввести назву завдання, опис, кількість сторі поінтів та дедлайн. Підтвердити додавання.
Очікуваний результат:	Виведення нового завдання у списку завдань.
Стан КЗ після проведення випробування	Виведення нового завдання у списку завдань.

Таблиця 5.6 – тестування функції «Додавання завдання в активні»

Мета тесту:	Перевірка функції “Додавання завдання в активні”
Початковий стан КЗ:	Відкрита сторінка завдань
Вхідні дані:	Список завдань
Схема проведення тесту:	Перетягнути завдання в рядок активних завдань
Очікуваний результат:	Виведення нового завдання у списку активних завдань.
Стан КЗ після проведення випробування	Виведення нового завдання у списку активних завдань.

Таблиця 5.7 – тестування функції «Зміна стану завдання»

Мета тесту:	Перевірка функції “Зміна стану завдання”
Початковий стан КЗ:	Відкрита сторінка активних завдань
Вхідні дані:	Список завдань
Схема проведення тесту:	Перетягнути завдання в стовпчик іншого стану
Очікуваний результат:	Виведення завдання в новому стані
Стан КЗ після проведення випробування	Виведення завдання в новому стані

Таблиця 5.8 – Тестування функції «Перехід на сторінку активних завдань»

Мета тесту:	Перевірка функції “Перехід на сторінку активних завдань”
Початковий стан КЗ:	Відкрита сторінка завдань
Вхідні дані:	Список сторінок
Схема проведення тесту:	Натиснути на кнопку активних завдань
Очікуваний результат:	Виведення активних завдань
Стан КЗ після проведення випробування	Виведення активних завдань

Схожість Цитати Посилання Коментар Підміна символів
Джерел на сторінці: 2-21, 25

Таблиця 5.9 – тестування функції «Перехід на сторінку історії змін»

Мета тесту:	Перевірка функції “Перехід на сторінку історії змін”
Початковий стан КЗ:	Відкрита сторінка завдань
Вхідні дані:	Список сторінок
Схема проведення тесту:	Натиснути на кнопку історії змін
Очікуваний результат:	Виведення історії змін
Стан КЗ після проведення випробування	Виведення історії змін

Таблиця 5.10 – тестування функції «Перехід на сторінку завдань»

Мета тесту:	Перевірка функції “Перехід на сторінку завдань”
Початковий стан КЗ:	Відкрита сторінка активних завдань
Вхідні дані:	Список сторінок
Схема проведення тесту:	Натиснути на кнопку завдань
Очікуваний результат:	Виведення завдань
Стан КЗ після проведення випробування	Виведення завдань

Висновок до розділу

Було протестовано послідовність дій під час використання застосунку. Кожен тест мав свою мету, початковий стан, вхідні дані, схему проведення, очікуваний результат та стан КЗ після проведення випробувань. Отримані результати коректні та успішні.

ЗАГАЛЬНІ ВИСНОВКИ

У даному дипломному проекті було описано предметне середовище інформаційної системи управління командою розробників. Ця система

створена для автоматизації процесу визначення можливої кількості витрат часу на виконання завдання, що було поставлене замовником. Було описано процес діяльності до та після автоматизації.

Також було оглянуто популярні існуючі аналоги. Основним їх недоліком є відсутня можливість модифікації через закритість вихідного коду та необхідність ведення backlog у сторонніх програмних продуктах.

Структура бази даних складається з таблиці з даними для навчання наївного баєсіва класифікатора, таблиці з проектами, користувачами та завданнями.

У математичному забезпеченні використовується наївний баєсів класифікатор. Також через розповсюдженість ефективних багатоядерних комп'ютерів було розроблено паралельну модель, що дозволило використовувати потужності обчислювальної системи з максимальною віддачою. Експерименти довели корисність використання такого підходу.

Застосунок побудований на мікросервісній архітектурі. Є клієнт, що написаний на react (html, css, js). А також rest api сервер, де за бізнес логіку відповідає node.js фреймворк fastify, а за ресурсоємні багатопроцесорні математичні обчислення - golang.

Також було створено керівництво користувача та проведені випробування програмного продукту.

ПЕРЕЛІК ПОСИЛАНЬ

1. Domingos, Pedro & Michael Pazzani (1997) «On the optimality of the simple Bayesian classifier under zero-one loss». Machine Learning
2. Rish, Irina. (2001). «An empirical study of the naive Bayes classifier». IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence.
3. Webb, G. I.; Boughton, J.; Wang, Z. (2005). "Not So Naive Bayes: Aggregating One-Dependence Estimators". Machine Learning.
4. Mozina, M.; Demsar, J.; Kattan, M.; Zupan, B. (2004). Nomograms for Visualization of Naive Bayesian Classifier.
5. Maron, M. E. (1961). "Automatic Indexing: An Experimental Inquiry". Journal of the ACM.
6. Minsky, M. (1961). Steps toward Artificial Intelligence.
7. McCallum, A. and Nigam K. «A Comparison of Event Models for Naive Bayes Text Classification»
8. Hand, DJ, & Yu, K. (2001). «Idiot's Bayes — not so stupid after all?» International Statistical Review.
9. Математические методы обучения по прецедентам (теория обучения машин). К. В. Воронцов
10. Комп'ютерна лінгвістика : Методичні вказівки до виконання лабораторних робіт для студентів спеціальностей. доц. Фіногенов Олексій Дмитрович
11. https://www.reactjs.org документація до фронтенд фреймворку react
12. https://www.fastify.io/ документація до бекенд фреймворку fastify
13. https://www.postgresql.org/ документація до СКБД Postgres
14. https://golang.org/ документація до мови програмування golang
15. https://nodejs.org/ документація до середовища виконання javascript
16. https://developer.mozilla.org/ru/docs/Web/API/Document документація до мови програмування javascript

Схожість

Схожість із джерелами з Інтернету 51

25	https://mykonspekts.ru/2-105.html	2 Джерело	1.69%
26	https://en.wikipedia.org/wiki/Naive_Bayesian_classifier		1.4%
27	https://www.wikizero.com/uk/%D0%9D%D0%B0%D1%97%D0%B2%D0%BD%D0%B8%D0%B9_%D0%B1%D0%B0%D1%94%D1...		1.38%
29	https://ela.kpi.ua/bitstream/123456789/30982/1/Liubyvyi_bakalavr.pdf		1.48%
38	https://ela.kpi.ua/bitstream/123456789/31069/1/Yashchuk_bakalavr.pdf		1.17%
44	https://ela.kpi.ua/bitstream/123456789/30254/1/Vykhlyayeva_bakalavr.pdf		0.94%
45	https://ela.kpi.ua/bitstream/123456789/30997/1/Cherkov_bakalavr.pdf		0.78%
48	https://ela.kpi.ua/bitstream/123456789/31041/1/Danylenko_bakalavr.pdf	2 Джерело	0.71%
51	https://refdb.ru/look/2533440-p5.html	3 Джерело	0.57%
57	https://ela.kpi.ua/bitstream/123456789/30987/1/Zhydkov_bakalavr.pdf	5 Джерело	0.39%
78	https://ela.kpi.ua/bitstream/123456789/23001/3/Popko_magistr.pdf		0.15%
79	https://studopedia.ru/20_28399_presuvannya-poroshkovih-materialiv.html	30 Джерело	0.15%
82	https://ru.react.js.org/docs/introducing-jsx.html	2 Джерело	0.13%

Схожість по Бібліотеці акаунту 230

1	Slepchuk_bachelor	ID **файлу**: 1003798194	Institution: National Technical University of Ukraine "Kyiv Polyte	36 Джерело	3.62%
2	Lubivij_bachelor	ID **файлу**: 1000044105	Institution: National Technical University of Ukraine "Kyiv Polytech	23 Джерело	3.54%
3	Nabokov_bachelor	ID **файлу**: 5882670	Institution: National Technical University of Ukraine "Kyiv Polytechn	10 Джерело	3.23%
4	Klimenko_bachelor	ID **файлу**: 5824711	Institution: National Technical University of Ukraine "Kyiv Polytechn	12 Джерело	3.2%
5	Melnichuk_bachelor	ID **файлу**: 1000017749	Institution: National Technical University of Ukraine "Kyiv Polyte	2 Джерело	3.18%
6	Shpakov_bachelor	ID **файлу**: 5978350	Institution: National Technical University of Ukraine "Kyiv Polytechni	6 Джерело	3.12%

7	Kosjak_bachelor	ID **файлу**: 1000017966	Institution: National Technical University of Ukraine "Kyiv Polytechn	3 Джерело	2.91%
8	Klimenko_Voloshin_bachelor	ID **файлу**: 5824717	Institution: National Technical University of Ukraine "Kyiv Polytechn...		2.76%
9	Bober_bachelor	ID **файлу**: 1000018789	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institu...		2.73%
10	Mihajlenko	ID **файлу**: 5850223	Institution: National Technical University of Ukraine "Kyiv Polytechnic Instit	4 Джерело	2.57%
11	Akimov_bachelor	ID **файлу**: 12272832	Institution: National Technical University of Ukraine "Kyiv Polytechni	2 Джерело	2.52%
12	Rogoza_bachelor	ID **файлу**: 5850171	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institute"		2.5%
13	Galaktionov_bachelor	ID **файлу**: 5882634	Institution: National Technical University of Ukraine "Kyiv Polytechnic Inst...		2.5%
14	Stepovyi_bachelor_ISz51	ID **файлу**: 1000017785	Institution: National Technical University of Ukraine "Kyiv Polytechn...		2.47%
15	Kovalenko_bachelor	ID **файлу**: 5859077	Institution: National Technical University of Ukraine "Kyiv Polytechnic Instit...		2.47%
16	Lavrinovich_bachelor	ID **файлу**: 5962508	Institution: National Technical University of Ukraine "Kyiv Polytech	2 Джерело	2.45%
17	Andrushchenko_Ilutsa_bachelor	ID **файлу**: 5978403	Institution: National Technical University of Ukraine "Kyiv Polyte...		2.45%
18	Avramenko_bachelor_is63	ID **файлу**: 1003897026	Institution: National Technical University of Ukraine "Kyiv	2 Джерело	2.44%
19	Kaminskij_Mushtruk_bachelor_v2	ID **файлу**: 6031680	Institution: National Technical University of Ukraine "	2 Джерело	2.39%
20	Rjumshina_bachelor_is62	ID **файлу**: 1003896782	Institution: National Technical University of Ukraine "Kyiv P	3 Джерело	2.27%
21	Marochkanich_Rogoza_bachelor	ID **файлу**: 5850176	Institution: National Technical University of Ukraine "Ky	2 Джерело	1.95%
22	Shmalko_bachelor	ID **файлу**: 5839518	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institut...		1.84%
23	Gychkevych_bachelor	ID **файлу**: 1000017950	Institution: National Technical University of Ukraine "Kyiv Polytechnic I...		1.82%
24	Busov_bachelor_is61	ID **файлу**: 1003896982	Institution: National Technical University of Ukraine "Kyiv Polyt	2 Джерело	1.74%
28	Ishenko_Levchuk_bachelor_Common	ID **файлу**: 1003798167	Institution: National Technical University of Ukr	3 Джерело	1.62%
30	Kaminskij_bachelor	ID **файлу**: 1000044112	Institution: National Technical University of Ukraine "Kyiv Polytechnic Ins...		1.4%
31	Ustenko_bachelor_isz51	ID **файлу**: 1000066367	Institution: National Technical University of Ukraine "Kyiv Polytechni...		1.4%
32	Truba_bachelor	ID **файлу**: 5850219	Institution: National Technical University of Ukraine "Kyiv Polytechnic In	2 Джерело	1.38%

33	Kalenichenko_bachelor	ID **файлу**: 5850099	Institution: National Technical University of Ukraine "Kyiv Polyte	2 Джерело	1.35%
34	Sborik_bachelor	ID **файлу**: 12135378	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institute"		1.35%
35	Homenko_bachelor_is61	ID **файлу**: 1003897031	Institution: National Technical University of Ukraine "Kyiv Polytechn...		1.28%
36	Andrushkiv_bachelor_is62	ID **файлу**: 1003897111	Institution: National Technical University of Ukraine "Kyiv Polytech...		1.28%
37	Javorskij_bachelor_is63	ID **файлу**: 1003896996	Institution: National Technical University of Ukraine "Kyiv Polytechnic...		1.27%
39	Jaschuk_bachelor_isz51	ID **файлу**: 1000066577	Institution: National Technical University of Ukraine "Kyiv Polytechnic...		1.15%
40	Chernenko_bachelor	ID **файлу**: 5938239	Institution: National Technical University of Ukraine "Kyiv Polytechnic Instit...		1.02%
41	Umanskij_bachelor	ID **файлу**: 1000017791	Institution: National Technical University of Ukraine "Kyiv Polytechnic Ins...		0.99%
42	Trukhan_Bachelor	ID **файлу**: 12191109	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institu...		0.99%
43	Didik_bachelor	ID **файлу**: 5849567	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institute"		0.96%
46	Storchevij_bachelor	ID **файлу**: 5839552	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institu...		0.76%
47	Чорна_КП-71мн_Пояснювальна записка	ID **файлу**: 11997243	Institution: National Technical University	5 Джерело	0.7%
49	Doroshenko_bachelor (1)	ID **файлу**: 1000018076	Institution: National Technical University of Ukraine "Kyiv Polytechn...		0.68%
50	Hlianko_bachelor	ID **файлу**: 5839304	Institution: National Technical University of Ukraine "Kyiv Polytechnic	3 Джерело	0.62%
52	Kasjan_bachelor_ISz41	ID **файлу**: 5978351	Institution: National Technical University of Ukraine "Kyiv Polytechnic Ins...		0.5%
53	Fedorko_bachelor	ID **файлу**: 5857686	Institution: National Technical University of Ukraine "Kyiv Polytechnic	4 Джерело	0.5%
54	Medvednikov_bachelor	ID **файлу**: 1000017739	Institution: National Technical University of Ukraine "Kyiv Polytechnic...		0.42%
55	Zvorskij_bachelor	ID **файлу**: 5882689	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institute"		0.42%
56	Osinnij_bachelor	ID **файлу**: 5882681	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institute"		0.41%
58	Shishman_bachelor	ID **файлу**: 12272988	Institution: National Technical University of Ukraine "Kyiv Polytechnic Instit...		0.36%
59	Javdoschuk_bachelor	ID **файлу**: 6002507	Institution: National Technical University of Ukraine "Kyiv Polytechnic Insti...		0.34%
60	Prokopenko_bachelor	ID **файлу**: 1000017741	Institution: National Technical University of Ukraine "Kyiv Polytechnic ...		0.32%

61	Knysh_PZ	ID **файлу**: 5151604	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institut	4 Джерело	0.31%
62	Fomin_bachelor	ID **файлу**: 1003798192	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institu...		0.31%
63	КРАТ_Юрій_cc7d62_main_part	ID **файлу**: 1000758166	Institution: National Technical University of Ukraine "Kyiv Pol...		0.1%
64	Dorogin_bachelor	ID **файлу**: 1000037718	Institution: National Technical University of Ukraine "Kyiv Polytechnic Insti...		0.29%
65	ПЗБП_АУТС_КПІ_2018_ПоштаренкоДА	ID **файлу**: 6031583	Institution: National Technical University of Ukraine "K...		0.29%
66	Antonenko_bachelor_ISZ21в	ID **файлу**: 5938261	Institution: National Technical University of Ukraine "Kyiv Polytechn...		0.29%
67	Студентська робота	ID **файлу**: 3377847	Institution: Lviv Polytechnic National University	3 Джерело	0.28%
68	Gavriluk_Osipenko_bachelor	ID **файлу**: 5911001	Institution: National Technical University of Ukraine "Kyiv Polytechn...		0.26%
69	Levitskij_Chernij_bachelor	ID **файлу**: 5907720	Institution: National Technical University of Ukraine "Kyiv Polytechnic...		0.26%
70	Butusov_bachelor	ID **файлу**: 5857457	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institute"		0.26%
71	ПЗМД_КПІ_АУТС_2018_ПопкоАВ	ID **файлу**: 5699024	Institution: National Technical University of Ukraine	4 Джерело	0.26%
72	Гребенюк КА-74мп	ID **файлу**: 8081528	Institution: National Technical University of Ukraine "Kyiv Polytechnic Insti...		0.26%
73	Kulivnyk-1-2018	ID **файлу**: 5968134	Institution: National Technical University of Ukraine "Kyiv Polytechnic In	2 Джерело	0.24%
74	Студентська робота	ID **файлу**: 1001164170	Institution: National Aviation University		0.18%
75	Студентська робота	ID **файлу**: 1086135	Institution: Lviv Polytechnic National University		0.16%
76	Студентська робота	ID **файлу**: 46843	Institution: Lviv Polytechnic National University		0.16%
77	Студентська робота	ID **файлу**: 1003682892	Institution: V.I. Vernadsky Taurida National University		0.16%
80	Студентська робота	ID **файлу**: 12188732	Institution: Donetsk National Technical University	30 Джерело	0.15%
81	ПЗМД_КПІ_АУТС_2019_СавінМС	ID **файлу**: 1000778861	Institution: National Technical University of Ukra	12 Джерело	0.15%
83	Kolischak_magistr_2	ID **файлу**: 8348200	Institution: National Technical University of Ukraine "Kyiv Polytechnic Instit...		0.13%
84	YakovenkoM_KV41	ID **файлу**: 5901990	Institution: National Technical University of Ukraine "Kyiv Polytechnic Institu...		0.13%
85	Студентська робота	ID **файлу**: 1003395390	Institution: Lviv Polytechnic National University		0.13%

Цитати

Цитати	2

1 (1997) «On the optimality of the simple Bayesian classifier under zero-one loss»

2 Rish, Irina. (2001).

Графічний матеріал
до дипломного проєкту

на тему: Інформаційна система підтримки управління командою розробників

Київ – 2020 року

Зм.	Арк.	№ докум.	Підп.	Дата
Розроб.		Ільєнко Р.С.		
Перев.		Коган А.В.		
Т. Кон.				
Н. Кон.		Проскура С.Л.		
Затв.		Коган А.В.		

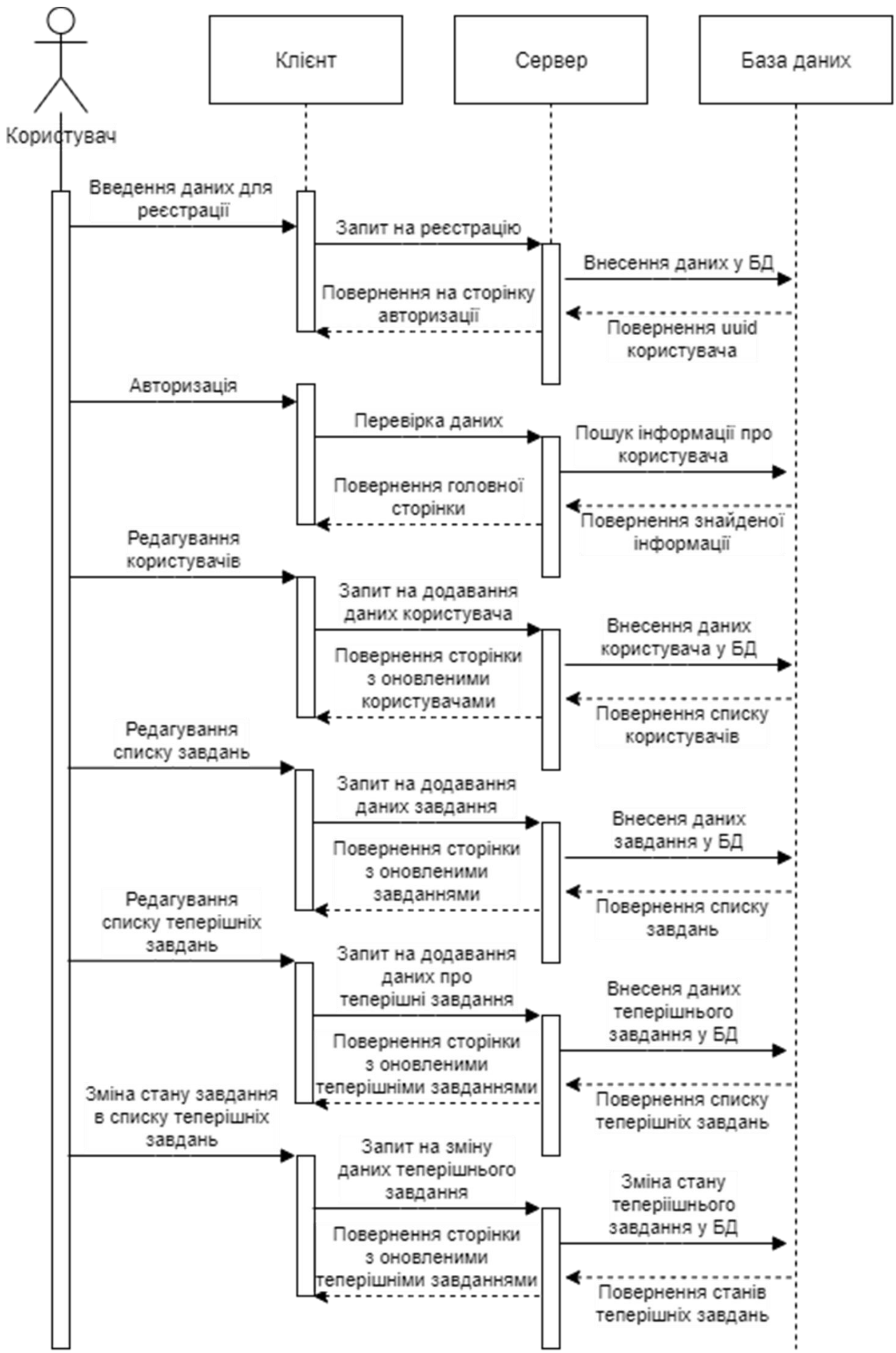
ДП 6309.02.000 ССВ

Діаграма варіантів використання

Лит.	Маса	Масштаб
Аркуш 1	Аркушів 1	

Інформаційна система підтримки управління командою розробників

КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-63



Зм.	Арк.	№ докум.	Підп.	Дата
Розроб.		Ільєнко Р.С.		
Перев.		Коган А.В.		
Т. Кон.				
Н. Кон.		Проскура С.Л.		
Затв.		Коган А.В.		

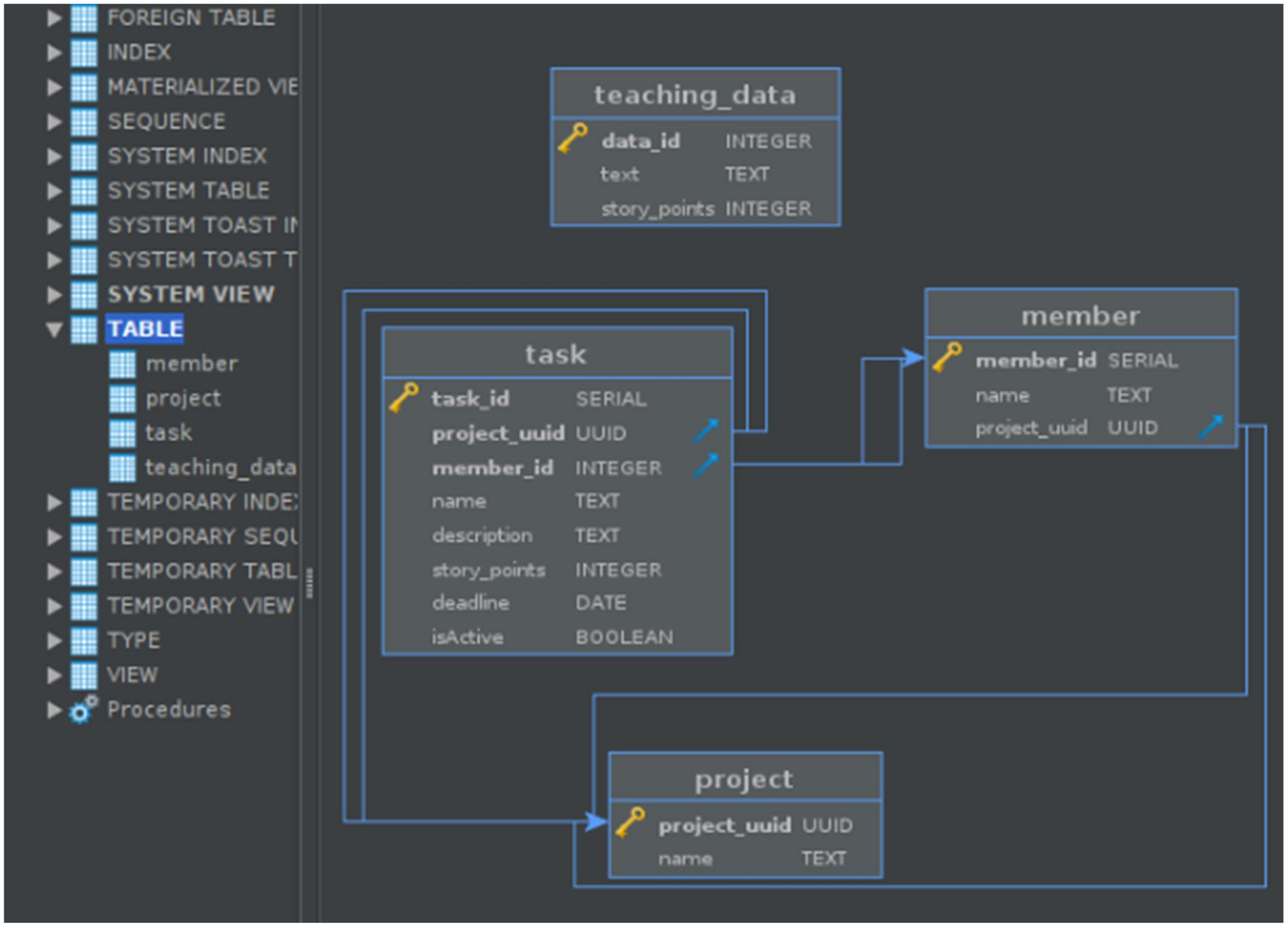
ДП 6309.03.000 ССП

Діаграма послідовності

Лит.	Маса	Масштаб
Аркуш 1	Аркушів 1	

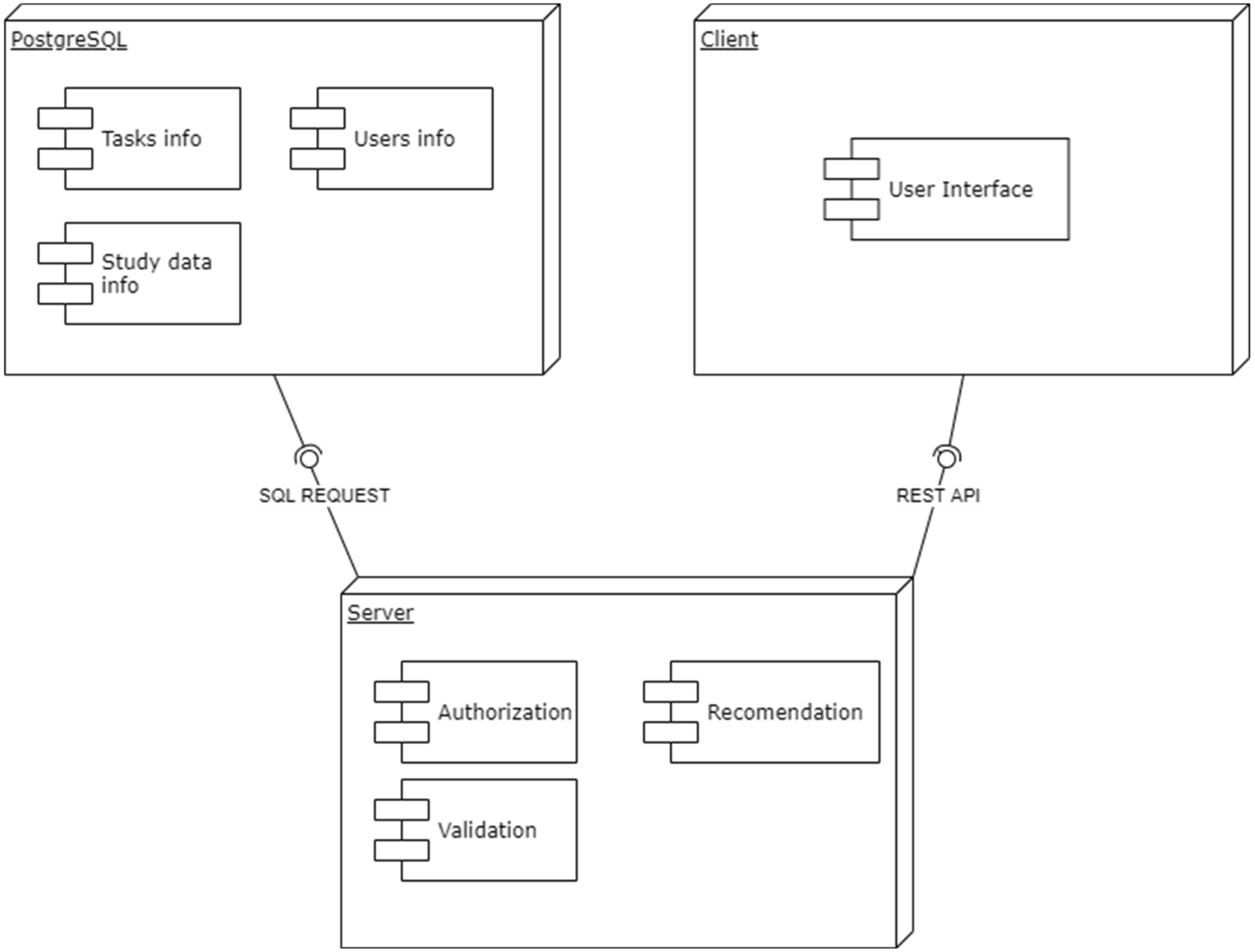
Інформаційна система підтримки управління командою розробників

КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-63



					ДП 6309.04.000 СБД			
Зм.	Арк.	№ документа	Підпис	Дата	**Схема бази даних**	Літера	Маса	Масштаб
Розробив		Ільєнко Р.С.						
Перевірив		Коган А.В.						
Т. кон.						Аркуш 1	Аркушів 1	
					Інформаційна система підтримки управління командою розробників	*КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-63*		
Н. кон.		Проскура С.Л.						
Затвердив		Коган А.В.						

ДП 6309.04.000 СБД



					ДП 6309.05.000 ССК			
Зм.	Арк.	№ документа	Підпис	Дата	***Діаграма компонентів***	Літера	Маса	Масштаб
Розробив		Ільєнко Р.С.						
Перевірив		Коган А.В.				Аркуш 1	Аркушів 1	
Т. кон.								
Н. кон.		Проскура С.Л.			*Інформаційна система підтримки управління командою розробників*	*КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-63*		
Затвердив		Коган А.В.						

ДП 6309.05.000 ССК

					ДП 6309.06.000 ССА			
Зм.	Арк.	№ документа	Підпис	Дата	**Архітектура веб-застосування**	Літера	Маса	Масштаб
Розробив		Ільєнко Р.С.						
Перевірив		Коган А.В.				Аркуш 1	Аркушів 1	
Т. кон.								
Н. кон.		Проскура С.Л.			*Інформаційна система підтримки управління командою розробників*	*КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-63*		
Затвердив		Коган А.В.						