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

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: «Web-додаток супроводження програмного серверу КЗАПР»

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

Виконавець роботи: студент групи ІТ-82-0 Могила Юрій Олександрович

Кваліфікаційна робота бакалавра
захищена на засіданні ЕК
з оцінкою _____ «___» 2022 р.

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

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

Студент _____
(підпис)

Суми-2022

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

ЗАТВЕРДЖУЮ
Зав. секцією ІТП

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

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА СТУДЕНТУ

Могила Юрій Олександрович

1 Тема роботи *Web-додаток супроводження програмного серверу КЗАПР»*

керівник роботи *Неня Віктор Григорович, к.т.н.*,

затверджені наказом по університету від « 27 » квітня 2022 р. № №0301-VI

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

3 Вхідні дані до роботи технічне завдання на розробку web-додатку супроводження програмного серверу КЗАПР

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

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

6. Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання: 05.10.2021

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Оформлення планування робіт	15.03.2022-25.03.2022	
2	Оформлення технічного завдання	25.03.2022-10.04.2022	
3	Проведення аналізу предметної області	10.04.2022-20.04.2022	
4	Проведення проектування web-додатку	20.04.2022-30.04.2022	
5	Розробка web-додатку	01.05.2022-23.05.2022	
6	Тестування web-додатку	23.05.2022-25.05.2022	
7	Завантаження web-додатку на хостинг	25.05.2022-30.05.2022	
8	Оформлення пояснювальної записки	15.03.2022-31.05.2022	

Студент _____ Могила Ю.О.
(підпис)

Керівник роботи _____ к.т.н., доц. Неня В.Г.
(підпис)

РЕФЕРАТ

Тема кваліфікаційної роботи бакалавра «web-додаток супроводження програмного серверу КЗАПР».

Пояснювальна записка складається зі вступу, трьох розділів, висновків, списку використаних джерел із 15 найменувань, чотирьох додатків. Загальний обсяг пояснювальної записки складає 55 сторінок, у тому числі 24 сторінки основного тексту, 2 сторінки списку використаних джерел, 22 сторінки додатків.

Кваліфікаційну роботу бакалавра присвячено розробці web-додатку супроводження програмного серверу КЗАПР.

У першому розділі проведено огляд останніх досліджень за тематикою роботи та проаналізовано аналоги розроблюваного web-додатку, визначено їх переваги та недоліки. Також було поставлено мету й задачі проекту, визначено засоби реалізації.

У другому розділі проведено структурно-функціональне моделювання, визначено варіанти використання web-додатку та спроектовано базу даних. У результаті було змодельовано такі діаграми як: контекстна діаграма IDEF0 та її декомпозиції, діаграма варіантів використання.

У третьому розділі описано процес розробки web-додатку, яка є результатом проектування. Також наведено архітектуру web-додатку. Проведено тестування роботи web-додатку супроводження програмного серверу КЗАПР.

Ключові слова: WEB-ДОДАТОК, БАЗА ДАНИХ, АВТОМАТИЗАЦІЯ ПРОЕКТУВАЛЬНИХ РОБІТ, СЕРВЕР, РОЗРОБКА.

ЗМІСТ

ВСТУП

Із зростанням попиту на масове виробництво речей, головним напрямом у підвищенні ефективності є його автоматизація. Із появою інформаційних технологій (ІТ) людство отримало багато нових можливостей. Наприклад, одна з них – це суттєво збільшити ефективність виготовлення продукції у великої кількості галузей промисловості. Усі масові виробництва починаються з проектних робіт. Це так, оскільки автоматизація неможлива без стандартизації та уніфікації останніх. Тому, ефективність виробництва напряму залежить від якості та швидкості виконання проектувальних робіт.

Ураховуючи вищесказане, можна зробити висновок, що сьогодні наявною є потреба в системах автоматизації проектувальних робіт. Їх застосовують для вирішення проблем підвищення якості та швидкості проведення останніх. Комплекс засобів автоматизації проектувальних робіт (КЗАПР) надає можливості суттєво зменшити витрати часу на організацію проектувальних робіт та підвищити їх якість виконання за рахунок застосування сучасних інформаційних технологій.

ІТ допомагають у проведенні розрахунків та автоматизації в промислових сферах, надають можливості моніторингу роботи віддалених підприємств. Серед останніх зростає попит на розробку власних КЗАПР.

Оскільки web-додатки за простотою в використанні та доступності лідирують серед інших рішень, ефективно буде створити КЗАПР web-орієнтованим. Працюючи з ним також треба передбачити обробку великої кількості інформації. Тому, виникає потреба в розробці програмного серверу для роботи з базою даних (БД). Тобто створюваний web-додаток повинен обробляти масиви даних.

Отже, метою даного проекту є розробка програмного серверу супроводження web-додатку КЗАПР.

Для досягнення поставленої мети треба вирішити наступні задачі:

- проаналізувати існуючі технології розробки та обрати необхідну для реалізації проекту;
- виконати аналіз аналогів;
- встановити базу даних на віддалений сервер;
- розробити алгоритми функціональних можливостей web-додатку КЗАПР;
- створити алгоритми збереження файлів;
- протестувати роботу розробленого web-додатку.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд останніх досліджень і публікацій

Серед методів вирішення проблеми автоматизації проектувальних робіт можна визначити два підходи: метод ручного контролю та управління та метод інтеграції КЗАПР.

Перший передбачає контроль над етапами роботи. Він відбувається за рахунок створення звітів працівників та наказів від керівника. Його перевагою є простота в інтеграції та контроль якості безпосередньо в місці виконання робіт. А недоліки – це низька швидкість впровадження наказів і отримання звітів, навантаження на працівника постійною розробкою останніх і складність контролю виконання робіт на декількох віддалених підприємствах.

Другий – це контроль за виконанням робіт, який проводиться шляхом інтеграції сучасних ІТ у КЗАПР. Недоліком є складність системи для не підготовлених співробітників. Серед переваг можна відзначити високу швидкість отримання звітів та впровадження наказів, можливість працювати з віддаленими підприємствами та зменшення навантаження на працівників шляхом спрощення створення репортів.

Тому, для реалізації даного проекту було обрано розробити web-додаток із можливістю авторизації в системі.

Для забезпечення роботи web-орієнтованого КЗАПР, необхідно розробити web-сервер додатку. Він має відповідати за обробку запитів від клієнтської частини та обмін інформацією с базою даних. Серед головних вимог до створюваного web-додатку є його відкритість для інтеграції з іншими додатками. Необхідно передбачити можливість розробки як мобільного, так і програмного його варіанту.

Для вирішення поставлених завдань можна виділити два архітектурні підходи: монолітний та мікросервісний [1]. Відмінність в тому, що в мікросервісній архітектурі логіка поділена на окремі сервіси які можуть бути

розгорнуті та використовуватись незалежно один від одного [1], в той час як в монолітній архітектурі вся бізнес логіка виконується лише на одному сервісі. Перевагами мікросервісної архітектури є модульність, надійність та масштабованість. Але наведені переваги реалізуються у повному обсязі лише при розробці великого та складного проекту, тому архітектурою розробки web-додатку була обрана більш простий та популярний монолітний підхід. Архітектурним стилем взаємодії компонентів web-додатку було обрано REST [2] архітектуру. Вона є досить популярною та ефективною в наш час. В її основі лежать головні властивості такі, як масштабованість, продуктивність, уніфікованість, відкритість зв'язків між компонентами та надійність. Завдяки принципу обміну запитами між клієнтською та серверною частиною можна досягти зручної масштабованості системи та можливості безпечного редагування логіки роботи серверу без небезпеки виведення з ладу клієнтської частини, макет REST архітектури можна побачити на рисунку 1.1.

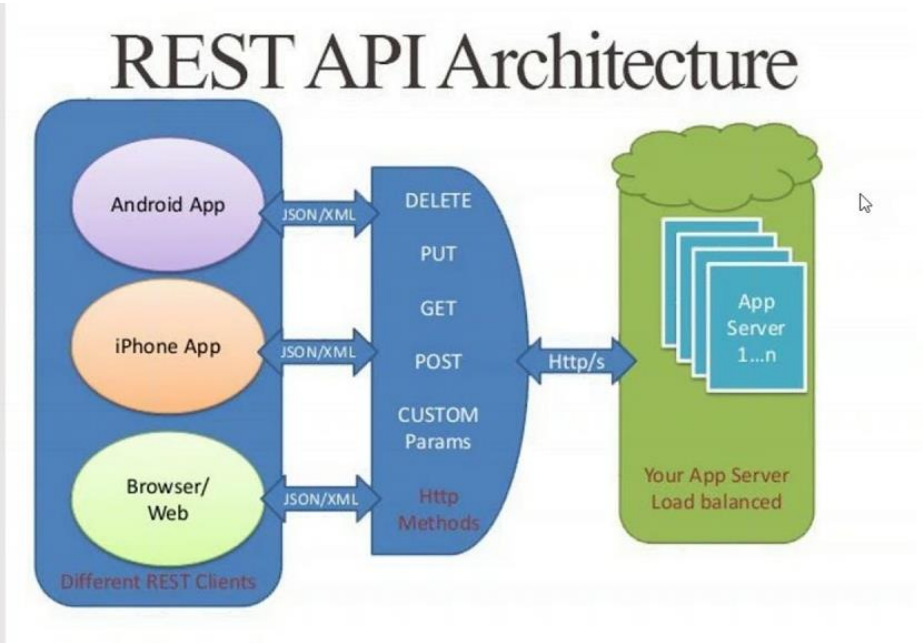

Рисунок 1.1 – Макет REST архітектури

Тому, для забезпечення децентралізованості додатку та дотримання сучасних стандартів та ресурсної доцільності було обрано монолітний

архітектурний підхід з дотриманням принципів REST у взаємодії компонентів web-додатку.

1.2 Аналіз програмних продуктів-аналогів

У сфері автоматизації проектувальних робіт розрізняють два види систем: CAD та CRM.

CAD (Computer-Aided Design) [3] – це організаційно-технічна система призначена для ведення проектної діяльності з застосуванням комп’ютерів. Їх використання надає можливість будувати календарний план робіт із можливістю побудови зв’язків між задачами, зазначення відповідального за кожну з них та функціоналом підрахунку тривалості виконання всього проекту або його частин. Також у CAD-системах реалізують можливості по управлінню бюджетом останнього на рівні планування. Одним із найпоширеніших CAD-рішень на ринку є Primavera Enterprise [4] від компанії Oracle. Використовуюи його, можна планувати роботи, складати бюджет та аналізувати хід виконання завдань. Primavera Expedition надає можливості стандартизації процесу адміністрування проектом. Це дозволяє полегшити та автоматизувати хід виконання проектних робіт.

CRM (customer relationship management) [5] – це система, яка надає можливості управління взаємовідносинами з клієнтами та оптимізації бізнес процесів. Інтеграція її в робочий процес підвищує якість взаємовідносин с кастомерами. Проте в CRM-додатках також реалізують функціонал автоматизації та контролю проектних робіт. Серед CRM-систем можна відзначити вітчизняну систему NetHunt [6], яка поєднує ефективність класичних CRM та функціонал сервісів Gmail. В області автоматизації проектних робіт NetHunt спрощує ведення звітності та полегшує роботу між різними відділами компанії. За рахунок її функціоналу сповіщення, автоматизується процес ведення проектних робіт та надання можливості

контролю над ними в реальному часі. Результати проведення аналізу двох систем наведено в таблиці 1.1.

Таблиця 1.1 – Порівняльний аналіз систем Primavera та NetHunt.

Критерії	Primavera	NetHunt
Можливість планування робіт	+	–
Управління бюджетом проекту	+	+
Система сповіщень	–	+
Функціонал ведення звітності	+	+
Відстеження ходу виконання робіт	+	+

Проаналізувавши системи Primavera та NetHunt, можемо відмітити їх певні переваги та недоліки. Наприклад, необхідно зазначити, що в кожній з них реалізовано велику кількість функціоналу непотрібного для вирішення задач планування проектувальних робіт. Висновок такий, що поєднання їх необхідних функціональних можливостей допоможе вдало розробити web-орієнтований додаток КЗАПР.

1.3 Постановка задачі

Метою роботи є створення web-додатку супроводження програмного серверу КЗАПР. Він призначений для забезпечення можливості контролю виконання проектувальних робіт та їх якістю, а також керування ними технічним керівником проекту.

Для вирішення задачі зі встановлення бази даних на віддалений сервер необхідно визначити тип бази даних що буде використовуватись, також встановити та налаштувати її. Для реалізації функціональних можливостей web-додатку необхідно розробити відповідні алгоритми для роботи с базою даних. Також для розширення функціональності реалізувати збереження файлів. В завершенні протестувати розроблений програмний сервер.

Цільовою аудиторією використання розроблюваного web-додатку є керівники організацій машинобудівного профілю, які мають потребу у пришвидшенні та автоматизації проектувальних робіт. Технічне завдання на розробку web-додатку супроводження програмного серверу КЗАПР наведено в Додатку А.

Розробка призначена для підвищення якості процесу автоматизації проектувальних робіт та зменшення витрат на його виконання за рахунок автоматизації ведення проектувальних робіт. Планування робіт дипломної роботи зображено в Додатку Б.

2 МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ WEB-ДОДАТКУ

2.1 Моделювання використання

Моделювання складається з етапів, які пов’язані між собою. Цей процес починається з абстрактної концептуальної схеми. Після неї створюються логічна та фізична моделі.

Для досягнення цих цілей спочатку будується так звана діаграми варіантів використання (use-case diagram) [7]. Вона описує функціональне призначення тієї чи іншої системи. Use-case diagram є вихідною концептуальною моделлю останньої в процесі її проектування та розробки.

Діаграма варіантів використання в расках даного дослідження в нотації UML представлена на рисунку 2.1.

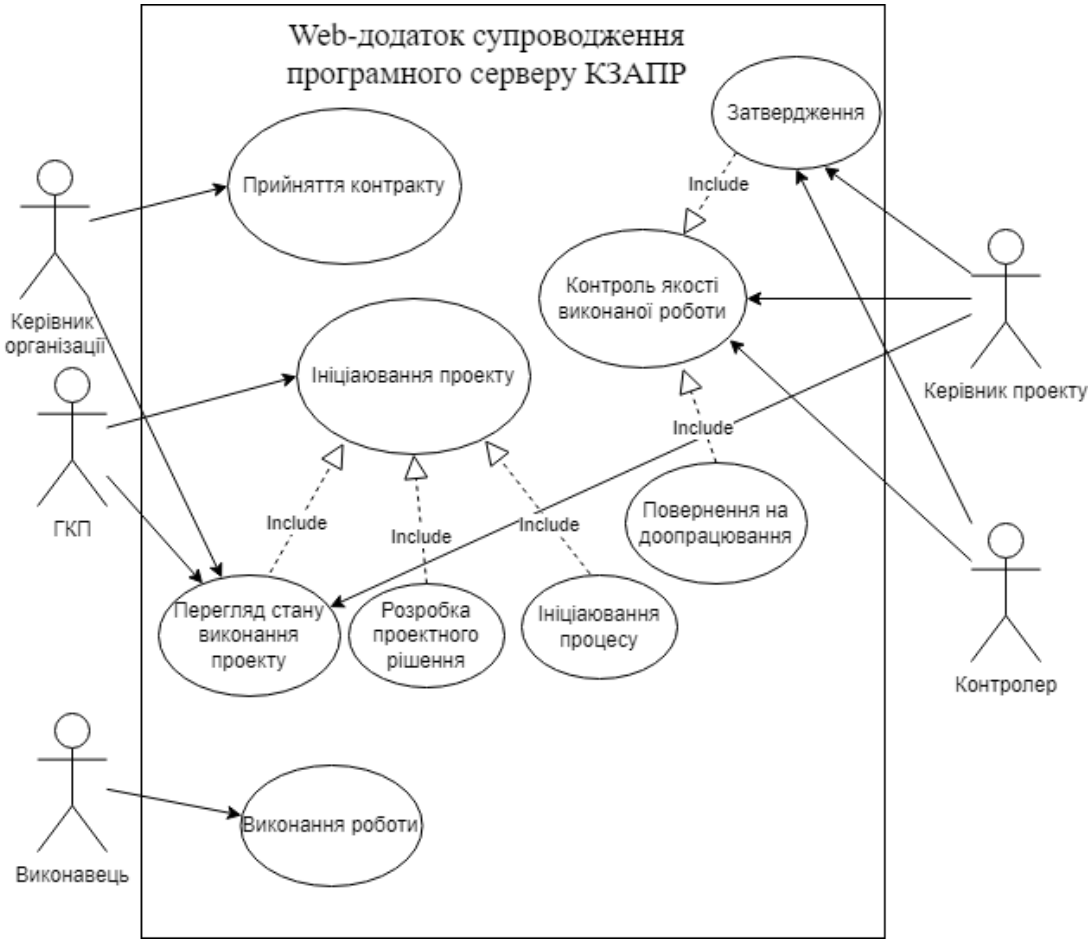

Рисунок 2.1 – Діаграма варіантів використання

2.2 Структурно-функціональне моделювання

Функціональне моделювання web-додатку супроводження програмного серверу КЗАПР в нотації IDEF0 [8] представлено на рисунку 2.2.

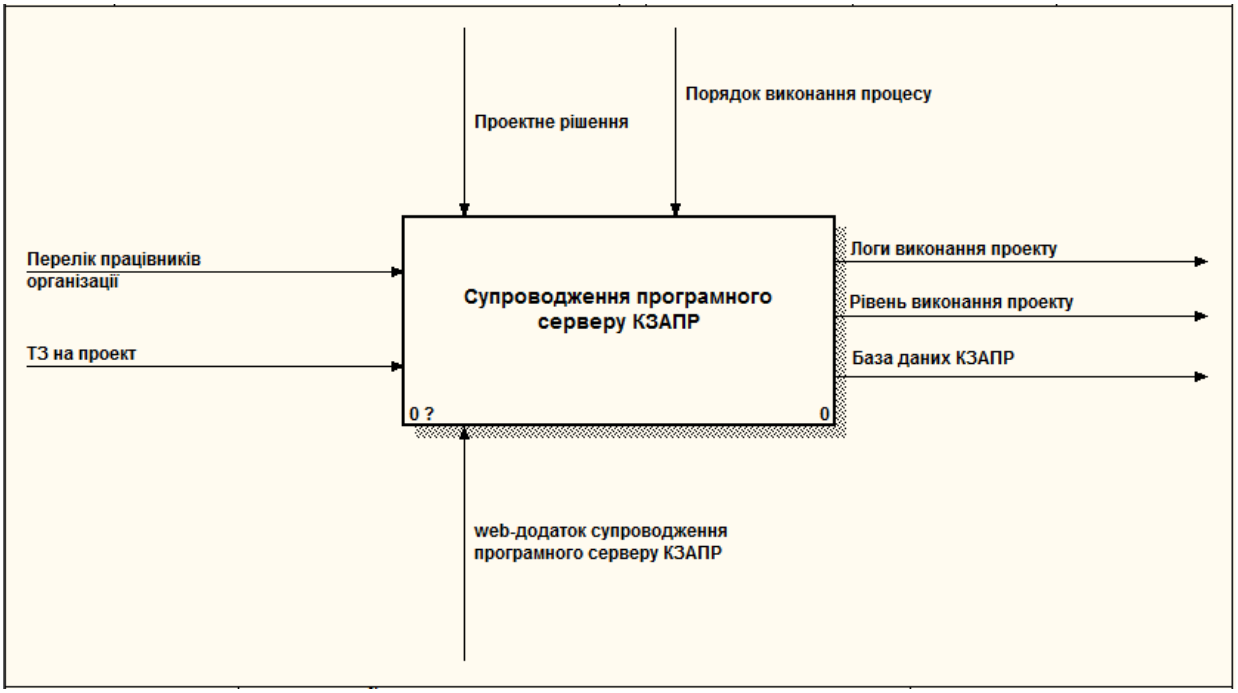

Рисунок 2.2 – Функціональна діаграма IDEF0

На вхід до роботи web-додатку супроводження програмного серверу КЗАПР подається «технічне завдання (ТЗ) на проект» та «Перелік працівників організації». За допомогою проектного рішення та порядку виконання процесів даний програмний продукт надає функціонал делегування відповідальних та контролерів.

Декомпозиція функціональної моделі web-додатку супроводження програмного серверу КЗАПР представлено на рисунку 2.3.

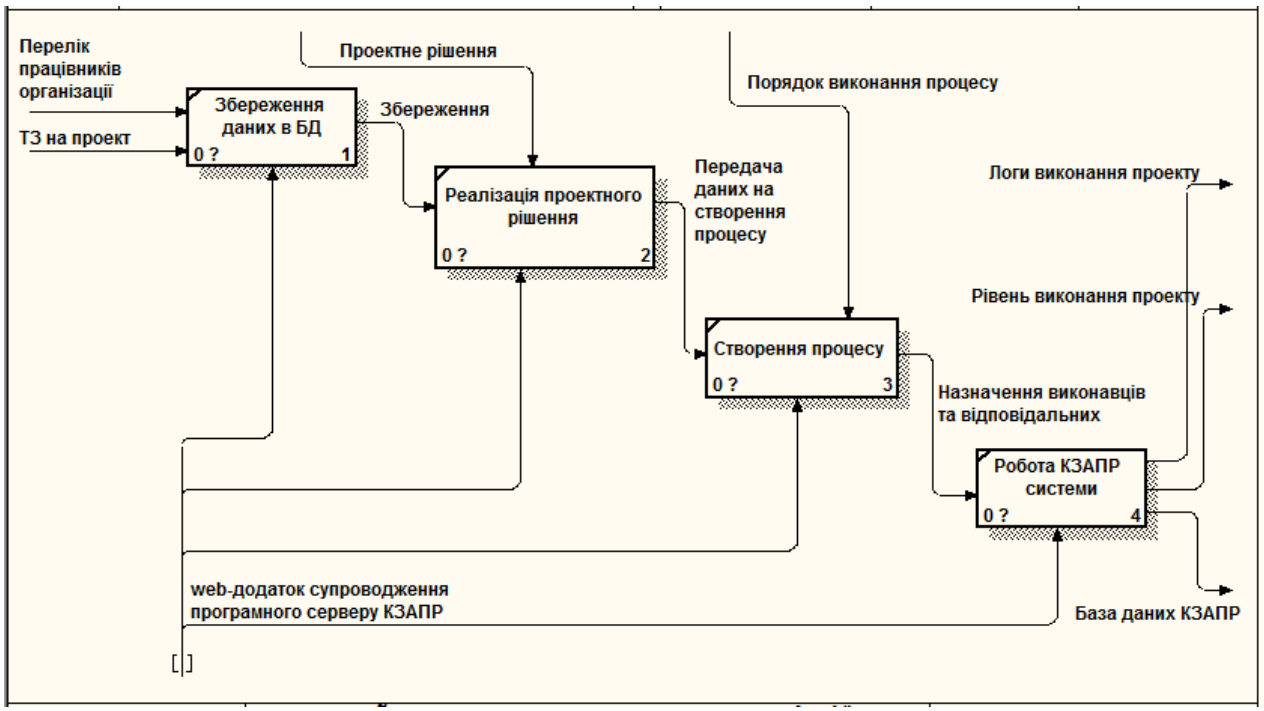

Рисунок 2.3 – Декомпозиція функціональної моделі

Для створення конкретної фізичної системи необхідно реалізувати всі елементи логічного опису в конкретних матеріальних елементах. Для представлення останніх призначене фізичне подання моделі. У мові UML це означає сукупність зв’язаних елементів, включаючи програмне й апаратне забезпечення, а також персонал для виконання спеціальних завдань. Для фізичного подання моделей тих чи інших систем використовуються діаграми реалізації. Вона дозволяє визначити архітектуру розробки. Основними графічними елементами діаграми компонентів є компоненти, інтерфейси та залежності між ними.

2.3 Діаграма розгортання

Діаграма розгортання [9] відображає обчислювальні вузли під час роботи системи. Web-додаток інтерфейсу обмінюється даними з

користувачем. У свою чергу web-інтерфейс посилає запит на сервер для обробки або отримання даних із MySQL бази даних.

На рисунку 2.4 представлена діаграма розгортання web-додатку супроводження програмного серверу КЗАПР.

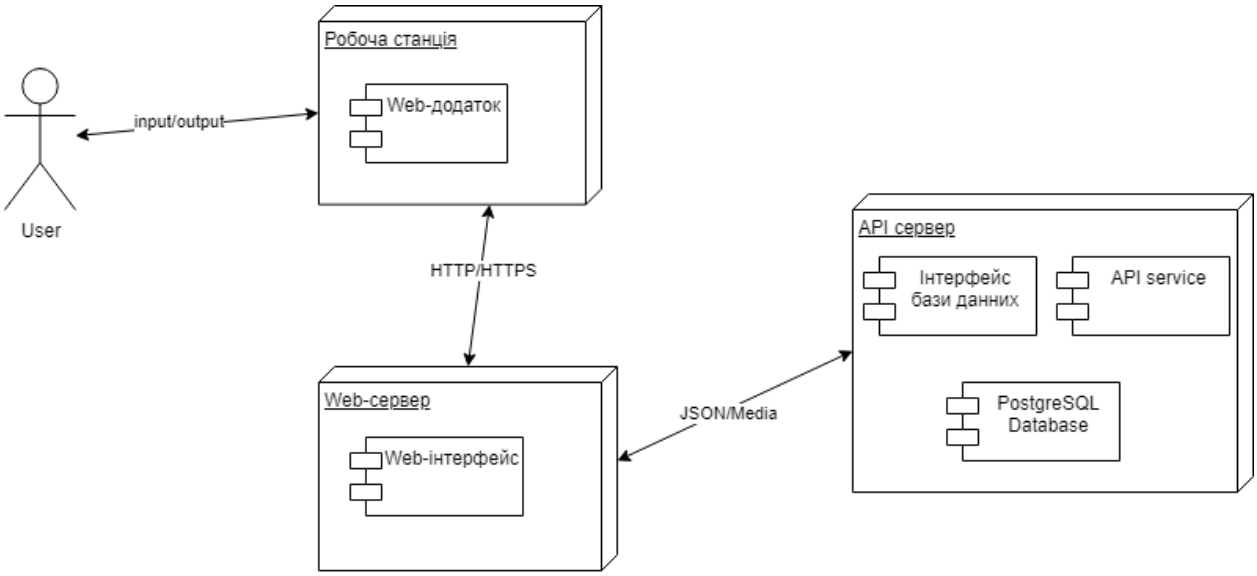

Рисунок 2.4 – Діаграма розгортання

2.4 Моделювання даних

У діаграмі [10] web-додатку супроводження програмного серверу КЗАПР відображено потік даних в межах його роботи (рис. 2.5).

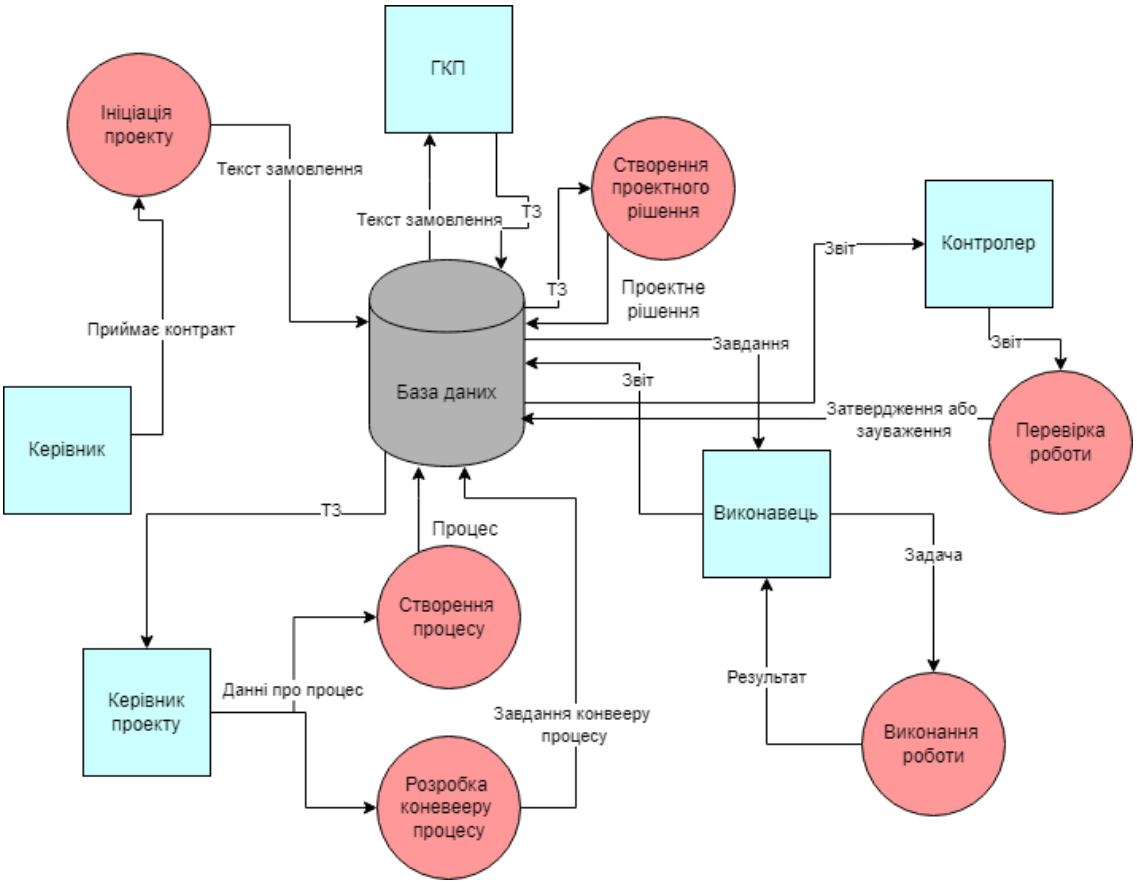

Рисунок 2.5 – Діаграма потоків даних web-додатку супроводження програмного серверу КЗАПР

На рисунку 2.6 представлена логічна модель бази даних [11] web-додатку супроводження програмного серверу КЗАПР.

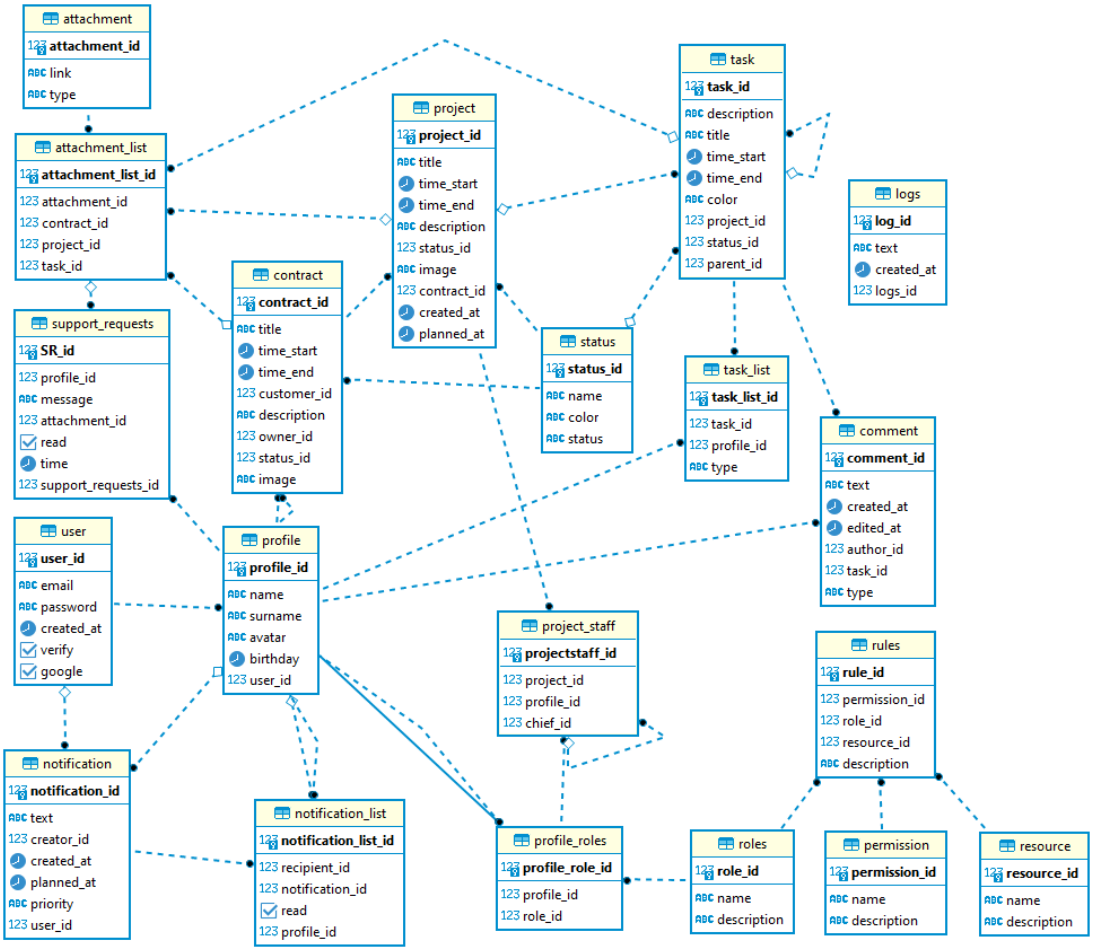

Рисунок 2.6 – Логічна модель даних web-додатку супроводження програмного серверу КЗАПР

У базі даних було передбачено рекурсивне делегування повноважень, закріплення файлів за завданнями та контрактами. Серед головних сутностей можна відзначити «Contract», «Project», «Profile», «Task», «Role».

3 РОЗРОБКА WEB-ДОДАТКУ СУПРОВОДЖЕННЯ ПРОГРАМНОГО СЕРВЕРУ КЗАПР

3.1 Архітектура web-додатку

Перед розробкою web-додатку супроводження програмного серверу КЗАПР необхідно провести моделювання компонентів, які будуть використані в процесі. Такі потреби вирішує модель HLD (High Level Design) [12]. HLD модель описує складові архітектури всієї системи, без використання складних термінів та глибоких особливостей архітектури.

Архітектура web-додатку супроводження програмного серверу КЗАПР складається з наступних елементів:

- база даних системи;
- об’єктна модель бази даних;
- об’єктів транспортування даних;
- контролер, що взаємодіє з об’єктною моделлю, виконує алгоритми та оброблює помилки;
- інтерфейсів репозиторіїв які реалізують функціонал роботи з базою даних.

Об’єктна модель представляє собою об’єктне відображення мовою програмування сутностей реляційної бази даних, а об’єкти транспортування даних використовуються для серіалізації даних та їх обміну між компонентами системи.

Діаграма високого рівня web-додатку супроводження програмного серверу КЗАПР представлена на рисунку 3.1

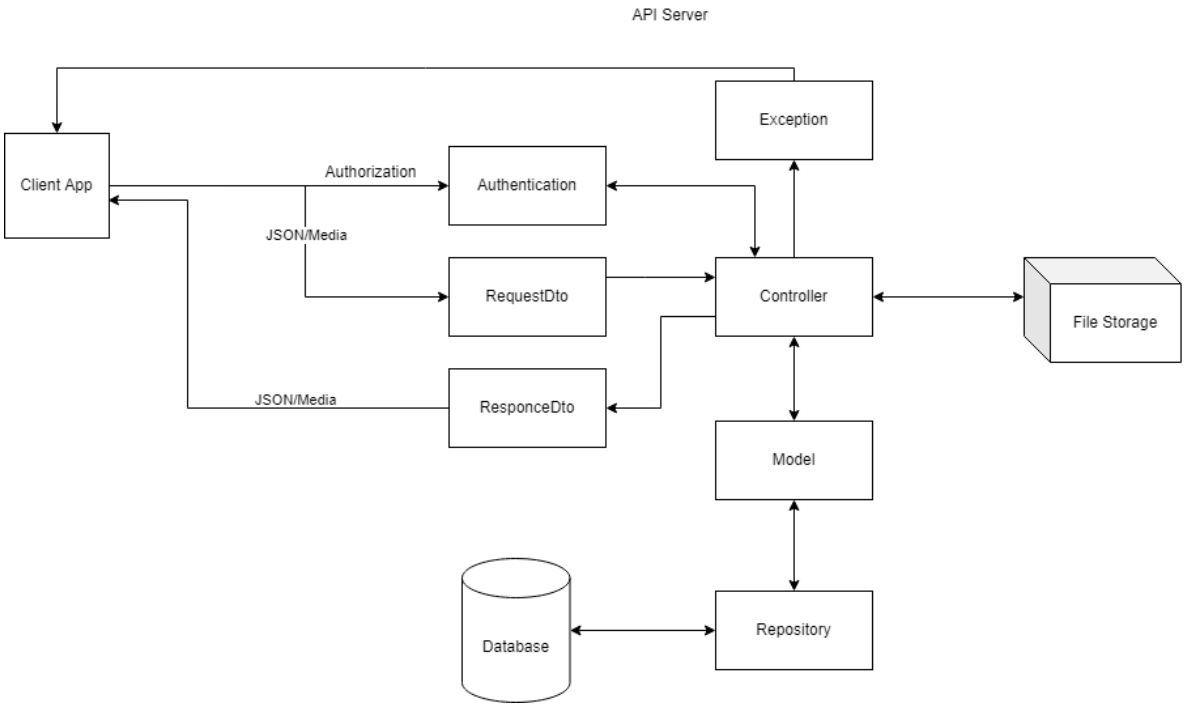

Рисунок 3.1 – Діаграма високого рівня web-додатку

3.2 Налагодження віддаленого серверу

Для реалізації клієнт-серверної архітектури, web-додаток супроводження програмного серверу КЗАПР необхідно встановити на віддаленому сервері. Останній має публічну IP адресу, за якою клієнтська частина КЗАПР буде виконувати запити для роботи з даними.

На віддалений сервер було встановлено пакети JRE (Java Runtime Environment), які надають можливість запускати програмні додатки Java, так як програмний сервер вирішено розроблювати на цій мові. Базою даних для КЗАПР було обрано PostgreSQL [13]. На рисунку 3.2 зображено версії Java та PostgreSQL встановлених на віддалений сервер.

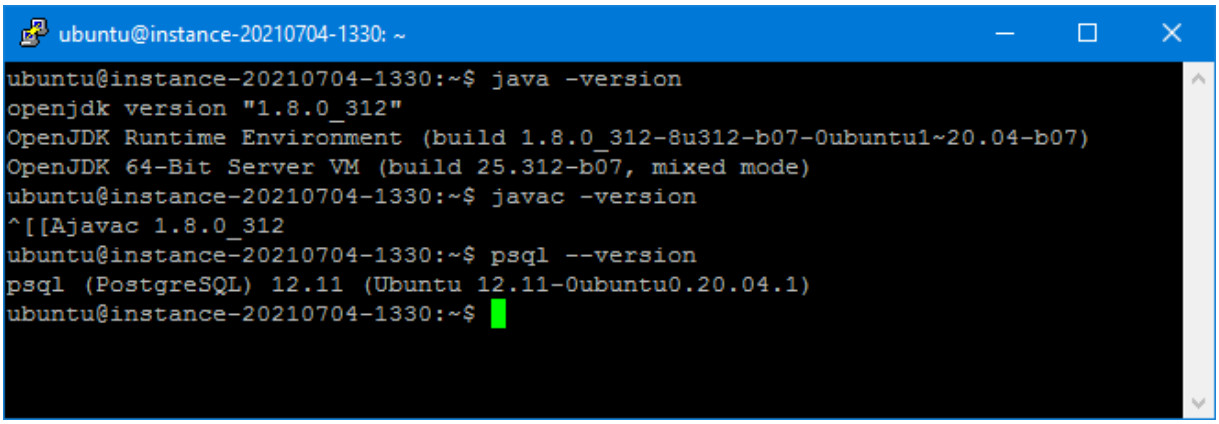

Рисунок 3.2 – Версії Java та PostgreSQL на віддаленому сервері

Для адміністрування та роботи з базою даних було обрано додаток PgAdmin, її інтерфейс представлено на рисунку 3.3.

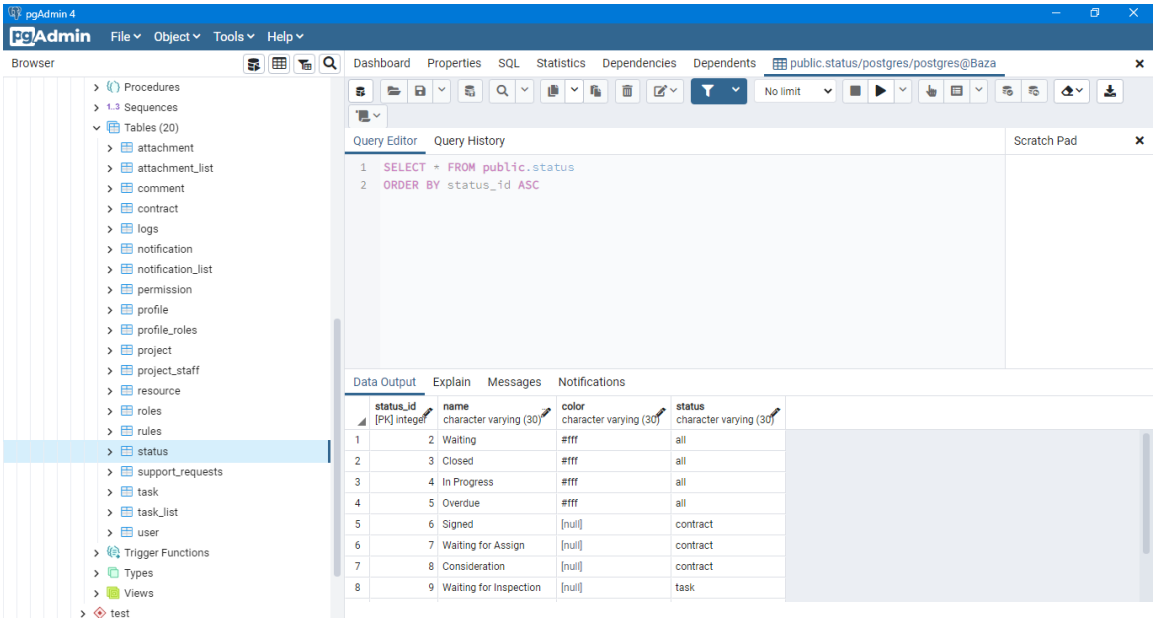

Рисунок 3.3 – Інтерфейс додатку PgAdmin

Систему управління реляційними базами даних PostgreSQL було обрано за рахунок її високої популярності, підтримці передових функцій та надійності.

3.3 Програмна реалізація

Архітектурою розробки web-додатку супроводження програмного серверу КЗАПР, було обрано REST. Тому доцільно було обрати програмний каркас (фреймворк) Spring [14] для написання програмного серверу КЗАПР. Spring – це фреймворк із відкритим кодом, що досяг великої популярності в розробці клієнт-серверних додатків на REST архітектурі. У нього інтегровано багато модулів для роботи з базами даних, тестування, віддаленого керування та багато іншого.

Перш за все, необхідно розробити об’єктну модель БД, яка є відображенням таблиць, полів та зв’язків в базі даних PostgreSQL. Бібліотека Hibernate, яка входить до складу фреймворку Spring, надає можливості роботи з базою даних із можливістю налаштування в коді з використанням анотацій. На рисунку 3.4 зображено приклад створення об’єктної моделі для таблиці «Contract».

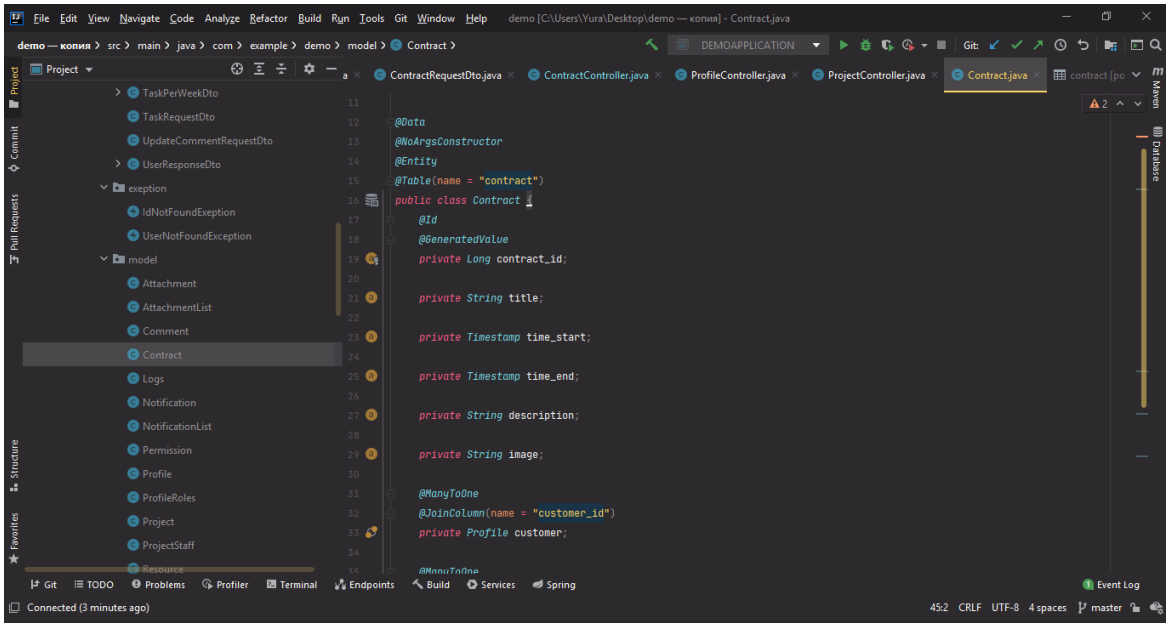

Рисунок 3.4 – Створення моделі таблиці «Замовник»

Для реалізації логіки роботи web-додатку супроводження програмного серверу КЗАПР необхідно розробити класи контролери, які будуть обробляти запити, що приходять на сервер. Тобто в контролерах зазначаються які саме requests можна виконати с сервером та логіка їх обробки. На рисунку 3.5 зображено приклад розробки контролера для запитів с таблицею «Замовник».

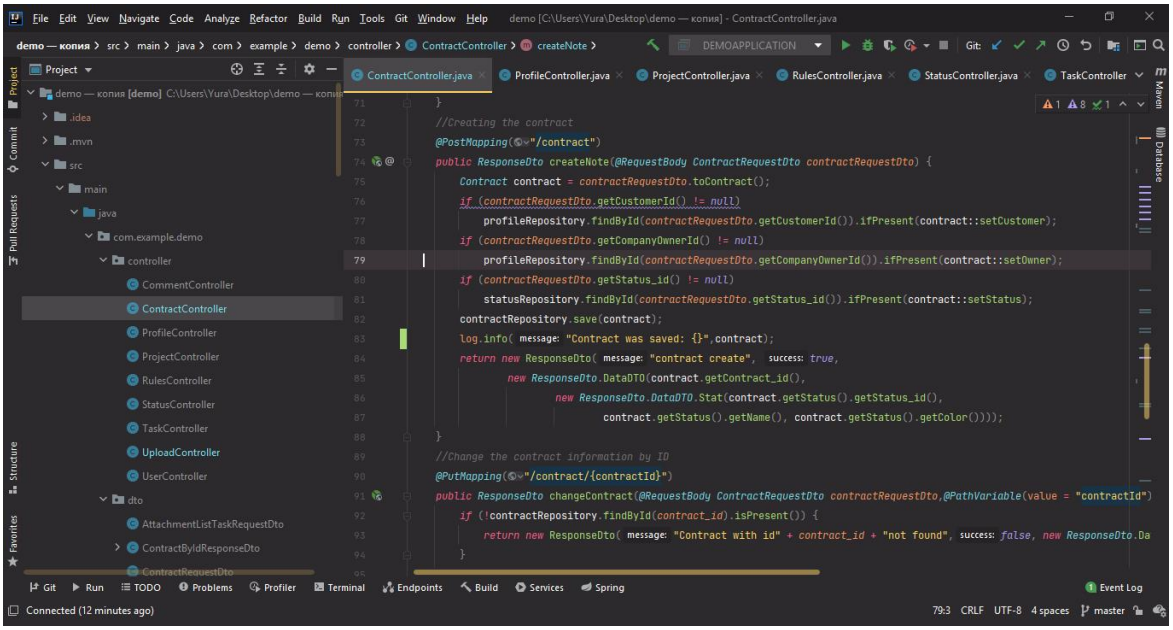

Рисунок 3.5 – Створення контролеру для роботи з контрактами

У процесі роботи КЗАПР виникає необхідність у збереженні файлів. Тому необхідно розробити контролер, який буде приймати файли та зберігати їх у відповідному репозиторії. На рисунку 3.6 зображено приклад створення контролеру завантаження файлів.

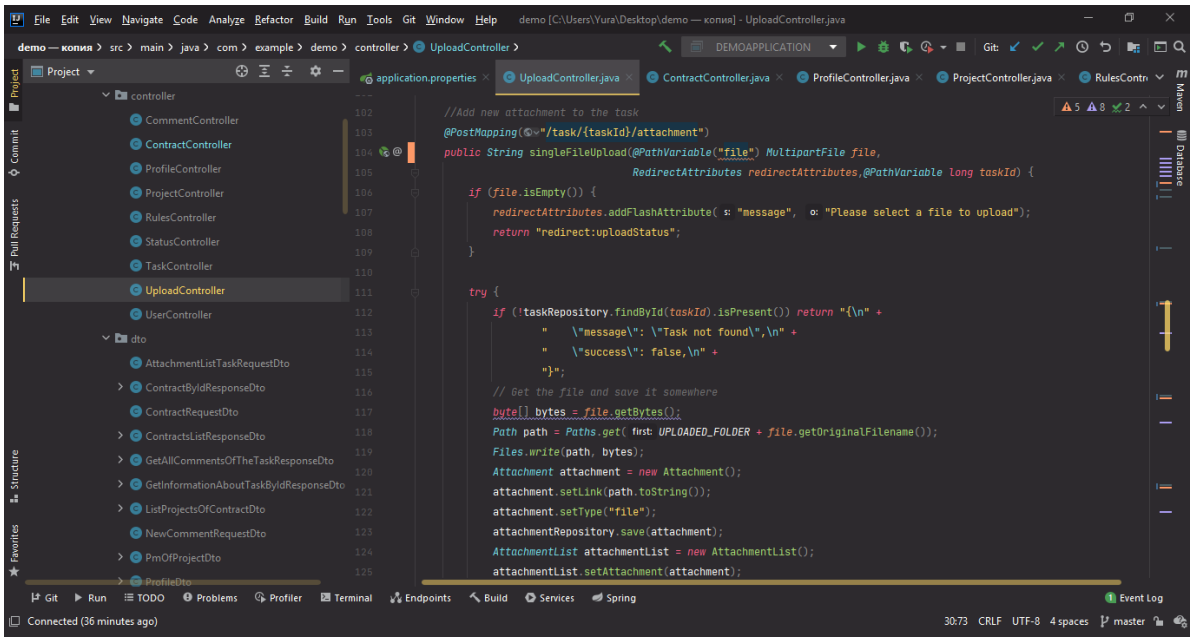

Рисунок 3.6 – Створення контролеру для завантаження файлів

При зберіганні файлу до бази даних в таблицю «Attachment» також додається запис, в якому зберігається шлях до файлу.

3.4 Демонстрація роботи

Для демонстрації роботи web-додатку супроводження програмного серверу КЗАПР, було обрано програмну утиліту Postman [15]. Це інструмент, який надає можливості перевірки запитів з клієнта на сервер. Також він забезпечує отримання відповіді від останнього. Отже, Postman є інструментом, що надає можливість повністю зімітувати роботу клієнта для перевірки коректності роботи серверної частини.

При завантаженні файлу на сервер необхідно вказати завдання, контракт чи проект у тілі запиту. На рисунку 3.7 зображено роботу request на додавання файлу до завдання під номером 1.

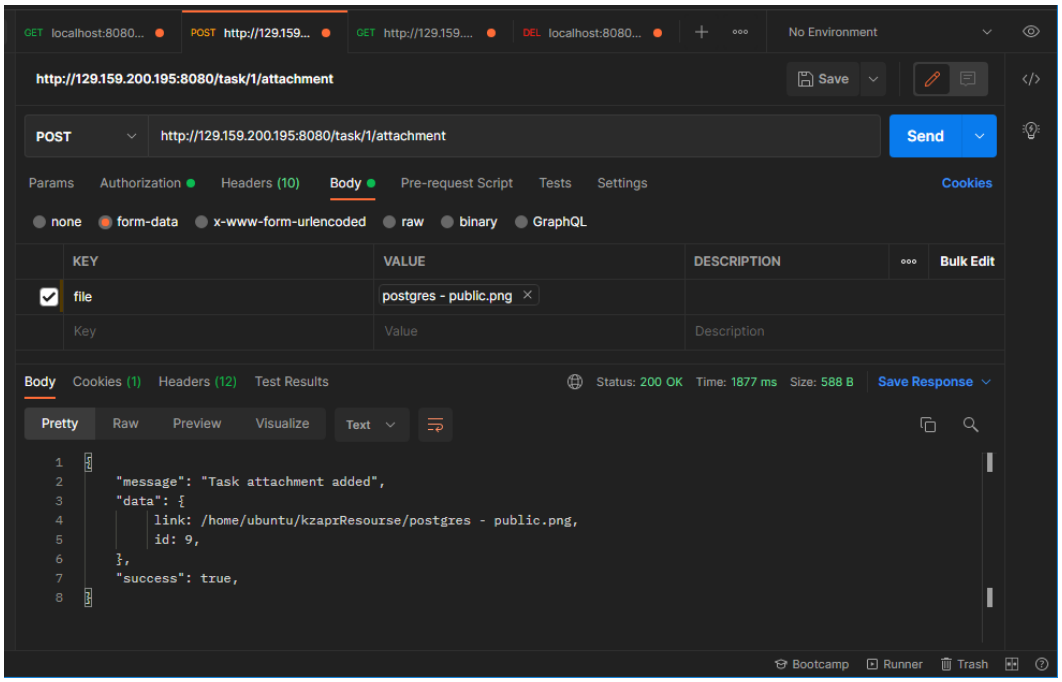

Рисунок 3.7 – Процес додавання файлу

Перевіремо коректність роботи запиту. Для цього введемо не існуючий номер завдання. У результаті отримано відповідь, що шуканого завдання не знайдено (рис. 3.8).

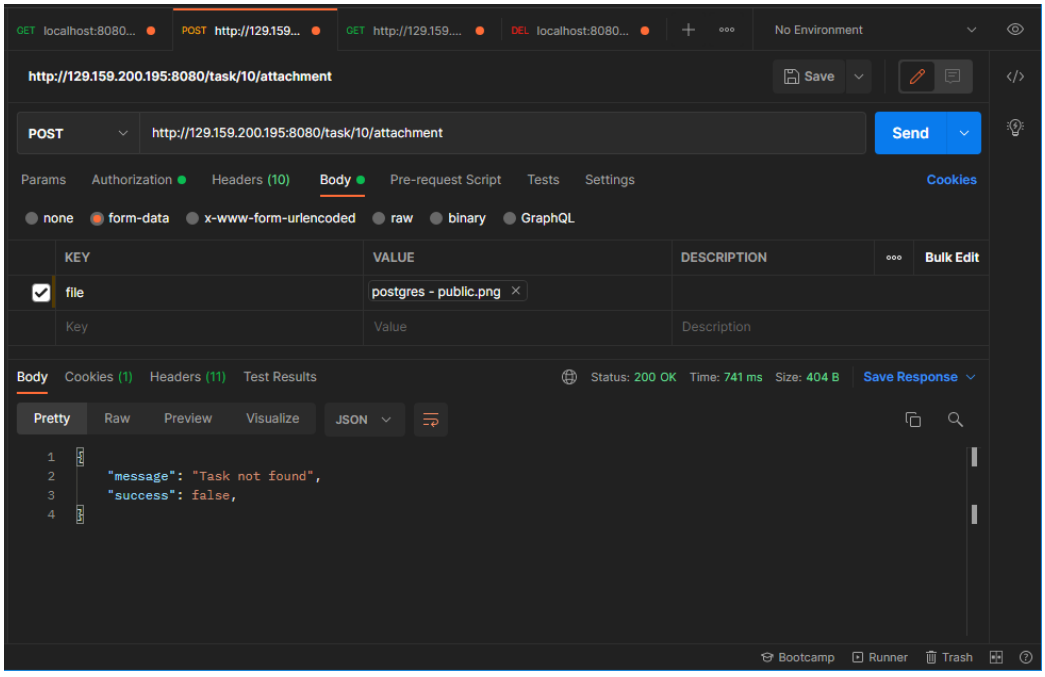

Рисунок 3.8 – Додавання файлу до не існуючого завдання

Результат завантаження файлу на віддалений сервер зображено на рисунку 3.9. Як бачимо файл знаходиться у відповідному каталозі.

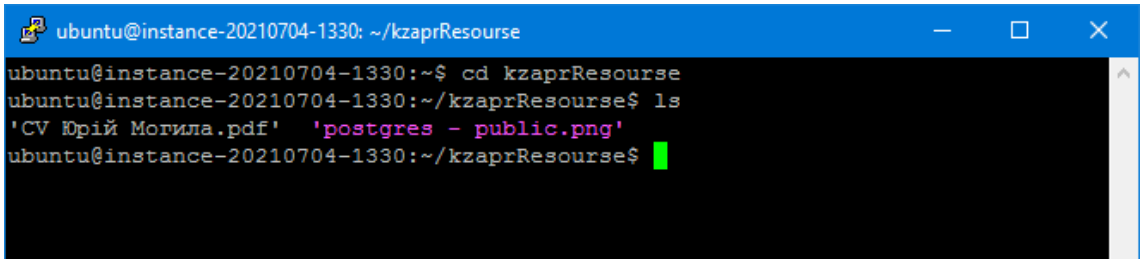

Рисунок 3.9 – Результат завантаження файлу

Для перегляду списку контрактів необхідно виконати запит «/contracts» до серверу. При його здійсненні отримуємо у відповідь масив усіх існуючих відповідних записів. Результат виконання зображено на рисунку 3.10.

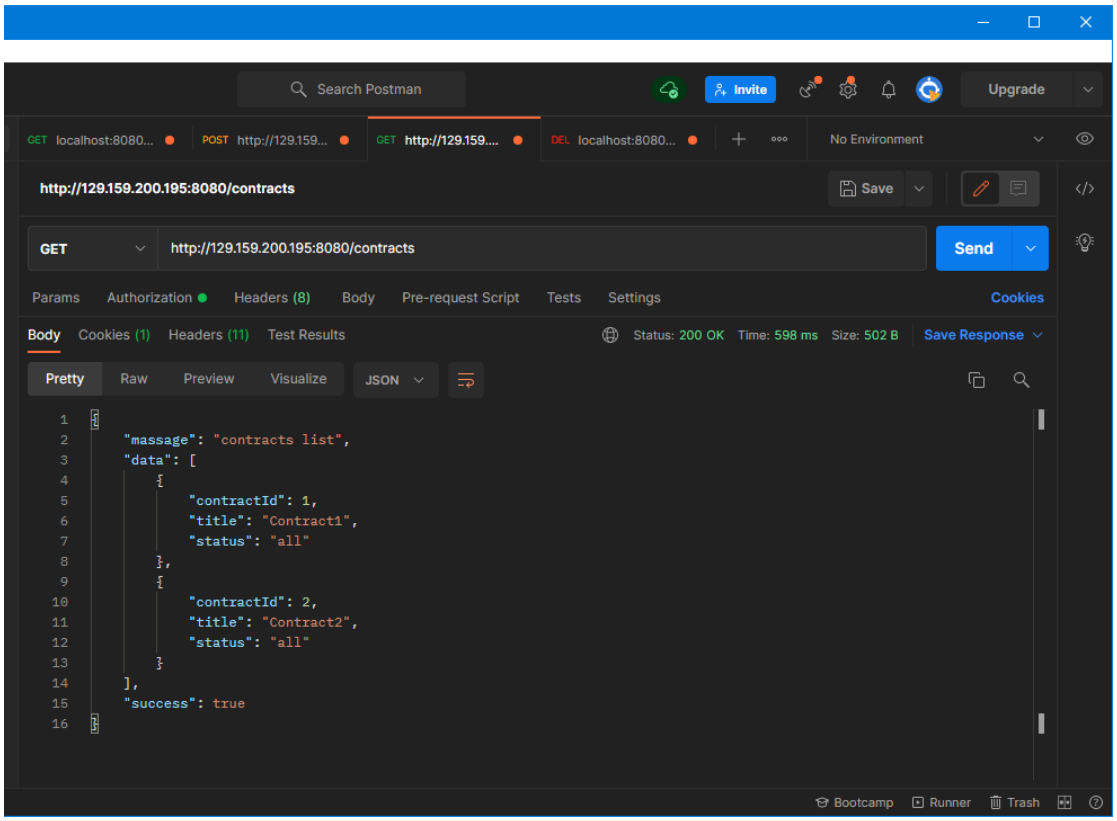

Рисунок 3.10 – Перегляд списку контрактів

Для додавання нового контракту необхідно виконати метод POST та додати до нього JSON об’єкт із параметрами контракту (рис. 3.11).

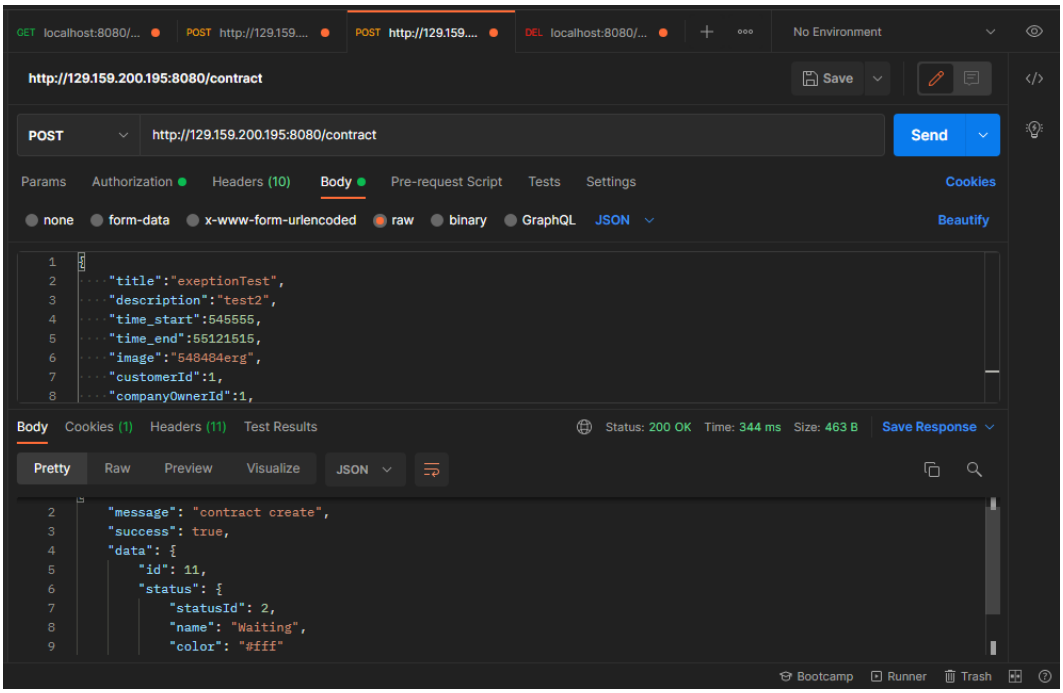

Рисунок 3.11 – Результат додавання контракту

Для перегляду детальної інформації про контракт необхідно в запиті вказати відповідний його id. На рисунку 3.12 продемонстровано результат здійснення такого request. У детальну інформаціюу входить також дані про замовника та відповідальну особу.

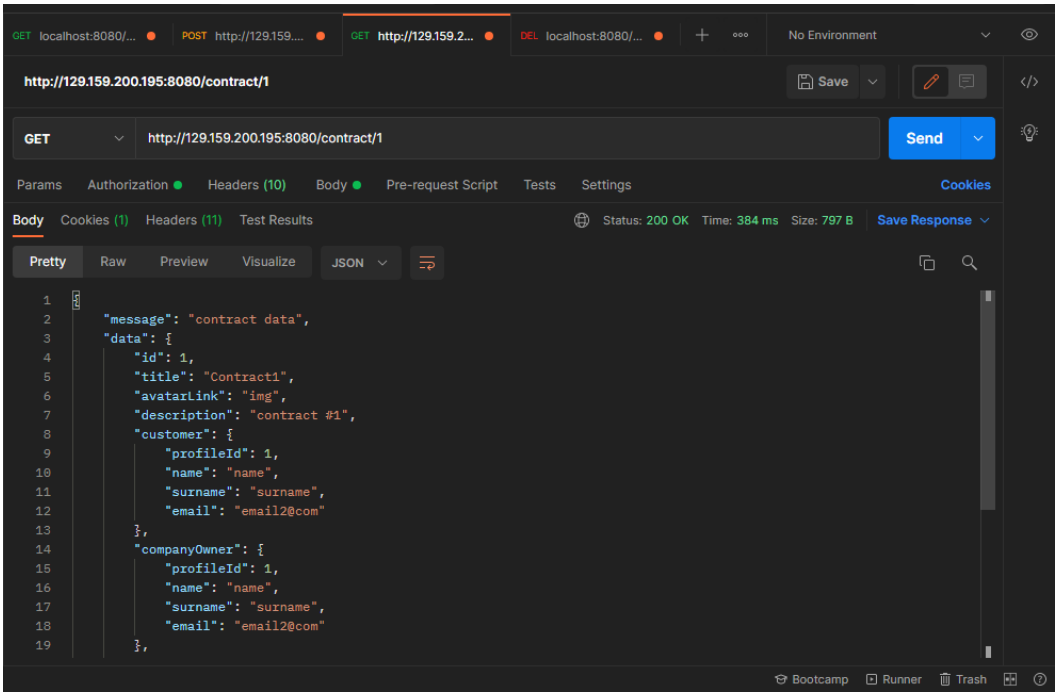

Рисунок 3.12 – Детальна інформація про контракт

На рисунках 3.13-3.14 зображено результати пошуку інформації про проекти контракту та перегляд кількості виконаних завдань за останній тиждень, розподілених по дням.

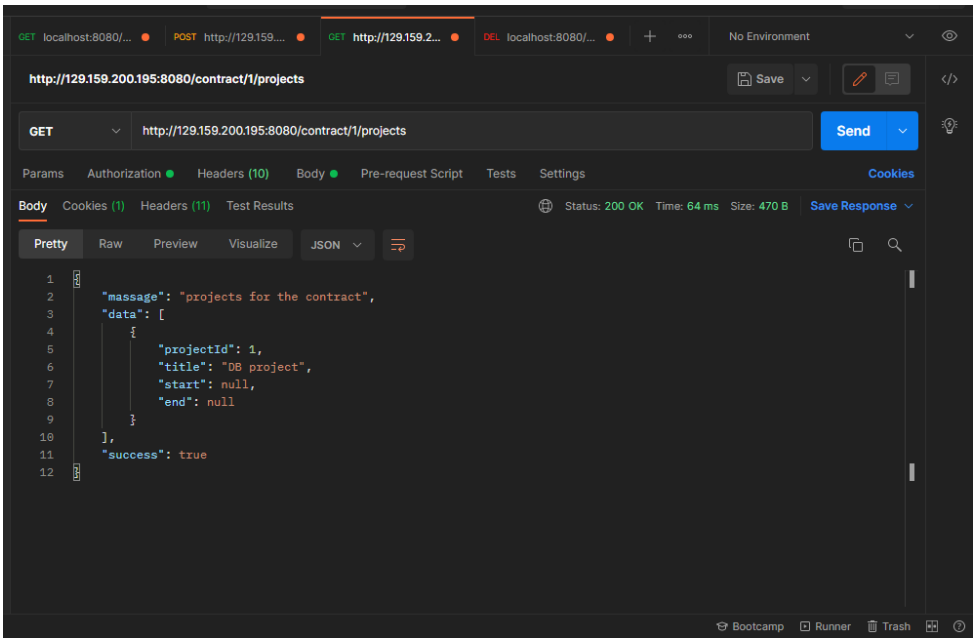

Рисунок 3.13 – Перелік проектів за контрактом

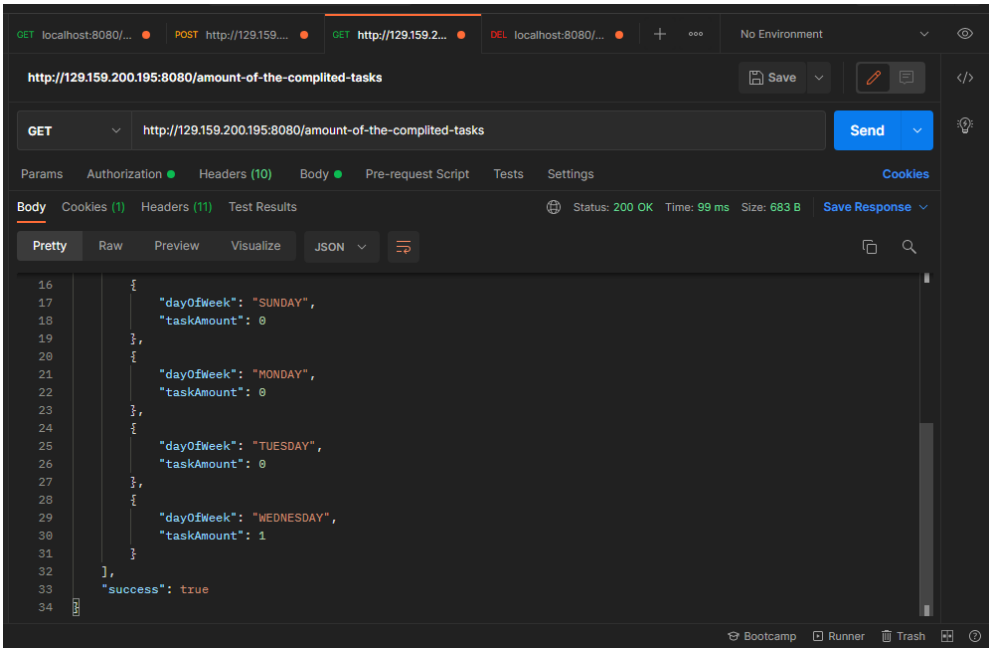

Рисунок 3.14 – Кількість виконаних завдань за останній тиждень

Не потрібні файли необхідно видаляти як з репозиторію серверу, так і з бази даних. Для запитів такого типу використовують методи DELETE. На

рисунку 3.15 зображено результат видалення файлу з КЗАПР. Знтщується як сам файл із репозиторію, так і запис про нього у таблиці «Attachment».

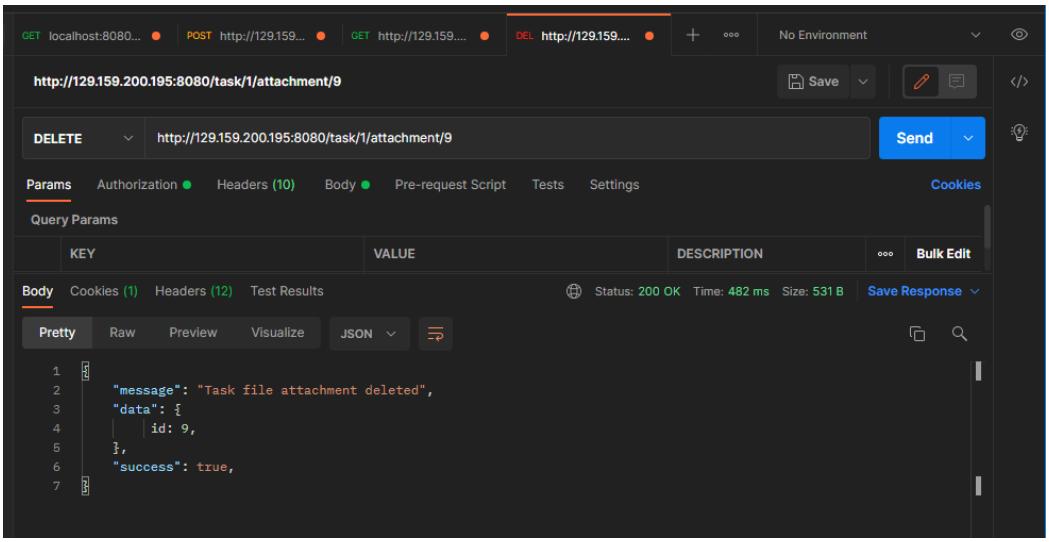

Рисунок 3.15 – Результат видалення файлу з серверу

ВИСНОВОК

У результаті виконання кваліфікаційної роботи бакалавра було розроблено web-додаток супроводження програмного серверу КЗАПР.

Під час реалізації дипломного проекту було проведено дослідження предметної області на розробку web-додатоку супроводження програмного серверу КЗАПР. У результаті було визначено актуальність даного проекту, основні вимоги й характеристики для створення програмного продукту та проаналізовано аналоги web-додатків. Як наслідок, сформульовано мету та задачі на розробку проекту «Web-додаток супроводження програмного серверу КЗАПР». Також було визначено технології для реалізації.

У проектній частині дипломної роботи було виконано структурно-функціональне моделювання та побудовано діаграму використання web-додатку. Під час проектування бази даних було визначено сутності, їх атрибути та дані, які будуть зберігатися.

У практичній частині дипломної роботи було розроблено web-додаток програмного серверу КЗАПР, та встановлено його на віддалений сервер. Також було виконано планування робіт та реагування на ризики, які можуть виникнути під час розробки програмного продукту (Додаток Б).

Використання розробленого web-додатку сприятиме підвищенню якості та швидкості проектувальних робіт у машинобудівній галузі. Також його застосування допоможе автоматизувати деякі організаційні процеси.

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

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

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Мікросервісна архітектура: плюси та мінуси [Електронний ресурс] – Режим доступу до ресурсу: https://blog.iteducenter.ua/ua/articles/microservices-architecture-advantages-and-disadventages/.

2. REST API Modeling Languages -A Developer ' s Perspective // IJSTE - International Journal of Science Technology and Engineering

3. Computer-aided design (CAD) системи [Електронний ресурс] – Режим доступу до ресурсу: http://mpm-rp.kpi.ua/metodichki/sistemi-avtomatizovanogo-proektuvannya-sapr/.

4. Primavera P6 Enterprise [Електронний ресурс] – Режим доступу до ресурсу: https://www.oracle.com/cis/industries/construction-engineering/primavera-p6/

5. CRM (customer relationship management) [Електронний ресурс] – Режим доступу до ресурсу: https://www.techtarget.com/searchcustomerexperience/definition/CRM-customer-relationship-management

6. Надійна CRM для всієї команди NetHunt [Електронний ресурс] – Режим доступу до ресурсу: https://nethunt.ua/

7. Мова UML. Діаграма використання [Електронний ресурс] – Режим доступу до ресурсу: http://p4ilka.blogspot.com/2018/12/uml.html.

8. Методологія IDEF0 [Електронний ресурс] – Режим доступу до ресурсу: https://studme.org/87184/ekonomika/metodologiya_idef0.

9. Діаграма розгортання і правила її побудови [Електронний ресурс] – Режим доступу до ресурсу: http://um.co.ua/9/9-2/9-29953.html

10. Діаграми потоків даних DFD [Електронний ресурс] – Режим доступу до ресурсу: http://um.co.ua/8/8-2/8-218941.html.

11. Проектування баз даних [Електронний ресурс] – Режим доступу до ресурсу: https://pidru4niki.com/11570718/bankivska_sprava/proektuvannya_baz_danih.

12. High-Level Design [Електронний ресурс] – Режим доступу до ресурсу: https://theopenarch.com/writing-an-effective-end-to-end-high-level-design/

13. PostgreSQL: Up and Running: A Practical Guide to the Advanced Open Source - Regina O. Obe, Leo S. Hsu

14. Spring Boot in Action - Craig Walls, 2015. – С. 10-40.

15. Postman [Електронний ресурс] – Режим доступу до ресурсу: https://www.postman.com/product/what-is-postman/

16. Analysis of REST API Implementation (International Journal of Scientific Research in Computer Science, Engineering and Information Technology) «2018 IJSRCSEIT»

17. Design and Implementation of REST API for Academic Information System «A A Prayogi et al 2020 IOP Conf.»

ДОДАТОК А

ТЕХНІЧНЕ ЗАВДАННЯ

на розробку

«Web-додаток супроводження програмного серверу КЗАПР»

ПОГОДЖЕНО:

Доцент кафедри комп’ютерних наук

_____ Неня В.Г.

Студент групи ІТ-82-0

_____ Могила Ю.О.

2022

1. Призначення й мета web-додатку супроводження програмного серверу КЗАПР

1.1 Призначення web-додатку

Web-додаток супроводження програмного серверу КЗАПР призначений для обміну даними з базою даних та виконанням операцій над ними.

1.2 Мета створення web-додатку

Головна мета проекту – це забезпечення супроводу web-додатку КЗАПР за рахунок розробленого серверного програмного додатку.

1.3 Цільова аудиторія

Цільовою аудиторією даного проекту є підприємства що мають потребу в системах автоматизації проектування.

2 Вимоги до проекту

2.1 Вимоги до проекту в цілому

2.1.1 Вимоги до структури й функціонування

Web-додаток супроводження програмного серверу КЗАПР, повинен коректно опрацьовувати команди клієнтської частини комплексу засобів автоматизованого проектування робіт, та бути встановленим на віддаленому сервері.

2.1.2 Вимоги до персоналу

Персонал закладу не повинен мати особливих технічних навичок для роботи з web-додатком і його підтримкою. Єдиною вимогою є наявність навичок користування персональним комп’ютером та web-браузером.

2.1.3 Вимоги до збереження інформації

Уся інформація надана у web-додатку повинна зберігатися у базі даних реалізованій засобами системи управління базами даних MySQL, обмін даними повинен виконуватись за рахунок JSON запитів.

2.1.4 Вимоги до розмежування доступу

Розроблюваний web-додаток супроводження програмного серверу КЗАПР, повинен розмежувати користувачів за двома групами: користувач та адміністратор. Адміністратор має необмежений доступ до програмного серверу, а користувач лише користується функціоналом системи в залежності від делегованого йому рівня.

2.1.5 Загальна інформація про структуру web-додатку

КЗАПР буде розроблено у вигляді WEB-додатку з можливостю авторизації працівників підприємства. При створенні web-додатка буде дотримано принципу REST архітектури, тобто частина що відповідає за відображення обмінюється повідомленнями з API сервером за допомогою HTTP методів, сервер оброблює логіку та працює з базою даних, макет на рисунку А1. Такий підхід облегшить масштабування КЗАПР на інші платформи наприклад створення мобільного додатку для роботи з системою.

2.2 Вимоги до видів забезпечення

2.2.1 Вимоги до лінгвістичного забезпечення

Всі системні повідомлення web-додатку супроводження програмного серверу КЗАПР мають бути виконані українською або англійською мовами.

2.2.2 Вимоги до програмного забезпечення

Для забезпечення стабільної роботи програмного серверу web-додатку КЗАПР, на віддаленому сервері повинна бути встановлено Java 1.8 та база даних MySQL.

2.3 Вимоги до функціонування системи

2.3.1 Потреби користувача

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

Таблиця А.1 – Потреби користувача

ID	**Потреби користувача**	**Джерело**
UN-01	Перегляд та редагування БД адміністратором системи.	Адміністратор
UN-02	Створення нового проекту	Користувач
UN-03	Створення ТЗ	Користувач
UN-04	Відстеження рівня завершеності проекту	Користувач
UN-05	Повідомлення про надходження за завершення виконання завдання.	Користувач
UN-06	Механізм збереження документів	Користувач

2.3.2 Системні вимоги

Програмний сервер має бути розміщено на віддаленому linux сервері зі встановленим середовищем для роботи с Java додатками (JRE) версії 1.8.

3 Склад і зміст робіт зі створення web-додатку супроводження програмного серверу КЗАПР

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

Таблиця А.2 – Етапи створення web-додатку

№	Склад і зміст робіт	Строк розробки
1	Аналіз предметної області	9 днів
2	Визначення властивостей додатку	11 днів
3	Затвердеження бази даних	7 днів
4	Розробка функціональних алгоритмів	21 день
5	Налагодження віддаленого серверу	14 днів
6	Тестування	7 днів
7	Написання супровідної документації	7 днів
8	Реліз Web-додатоку супроводження програмного серверу КЗАПР	7 днів
	Загальна тривалість робіт	83 днів

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

Web-додаток супроводження програмного серверу КЗАПР, має бути розроблений та розміщений на віддаленому сервері.

ДОДАТОК Б

Планування робіт

Сучасна економічна система невід'ємно пов'язана з використанням інформаційних технологій (ІТ). Їх використання забезпечує підвищення продуктивності робіт в різних галузях промисловості.

Програмні продукти допомагають в проведенні розрахунків та автоматизації виробництва в промислових сферах, надають можливості моніторингу роботи віддалених підприємств тощо. Серед організацій машинобудівного напрямку зростає попит на розробку комплексу засобів автоматизації проектувальних робіт (КЗАПР).

Так як web-додатки за простотою в використанні та доступності лідирують серед інших рішень. Тому розроблення КЗАПР у вигляді web-додатка є ефективним.

Web-додаток КЗАПР повинен обробляти масиви даних тому, виникає потреба в розробці програмного серверу для роботи з базою даних.

Отже, метою проекту є забезпечення супроводу web-додатку КЗАПР за рахунок розробленого серверного програмного додатку.

Деталізація мети проекту методом SMART. Для успішності та конкурентоспроможності проекту треба на концептуальному етапі правильно визначити його мету за допомогою SMART-методу. Результати деталізації методом SMART розміщено у таблиці Б.1.

Таблиця Б.1 – Деталізація мети проекту методом SMART

Specific (конкретна)	Забезпечення супроводу web-додатку КЗАПР за рахунок розробленого серверного програмного додатку.
Measurable (вимірювана)	Розроблений сервер web-додатку КЗАПР.

Продовження табл. Б1

Achievable (досяжна, узгоджена)	Для виконання проекту наявні необхідні знання Java, SpringBoot, REST, баз даних MySQL та навичок написання документації. Враховуючи доступні ресурсні можливості та обмеження мета є такою, яку можливо досягти. Є затверджене технічне завдання.
Relevant (реалістична)	Створена серверна частина Web-додатку КЗАПР, реалізує алгоритми необхідні для зберігання відтворення та обробки інформації з бази даних КЗАПР.
Time-framed (обмежена в часі)	Ціль має часове обмеження. Термін досягнення мети проекту визначено за домовленістю між замовником та виконавцем і складає три місяці.

Планування змісту робіт. WBS (Work Breakdown Structure – ієрархічна структура робіт) – це графічний вигляд елементів проекту, які згруповані ієрархією у єдине ціле з продуктом проекту. Структура декомпозиції робіт орієнтована на досконале виконання робіт по частинам і сама є ключовою частиною проекту, яка спрямована на організацію командної роботи. Елементами декомпозиції можуть бути продукти, дані та послуги. Більше того, WBS забезпечує необхідним каркасом для ретельної оцінки термінів та контролю та графіків роботи.

На найвищому (першому) рівні розміщений продукт проекту. Основні дії та заходи, що забезпечують досягнення мети проекту, зафіксовані на другому рівні декомпозиції. Декомпозиція робіт виконується до тих пір, поки вони не стануть елементарними (простими).

Елементарні роботи – це дії, які мають однозначний чіткий результат, на які призначена відповідальному одна конкретна особа, для якої можна обчислити витрати праці і тривалість виконання. На рисунку Б.1 представлено WBS з розробки Web-додатоку супроводження програмного серверу КЗАПР.

Планування структури виконавців. Наступним етапом після декомпозиції процесів є розробка організаційної структури виконавців або OBS, яка визначається як графічна структура відображення учасників або відповідальних осіб, які беруть участь у реалізації проекту.

У ролі відповідальних осіб виступають співробітники, що відповідають за організацію і виконання елементарної роботи, що зазначена у WBS. Кожну елементарну роботи можна розглядати як окремий проект.

На рисунку Б.2 представлено організаційну структуру планування проекту.

Список виконавців, що функціонують в проекті описано в таблиці Б.2.

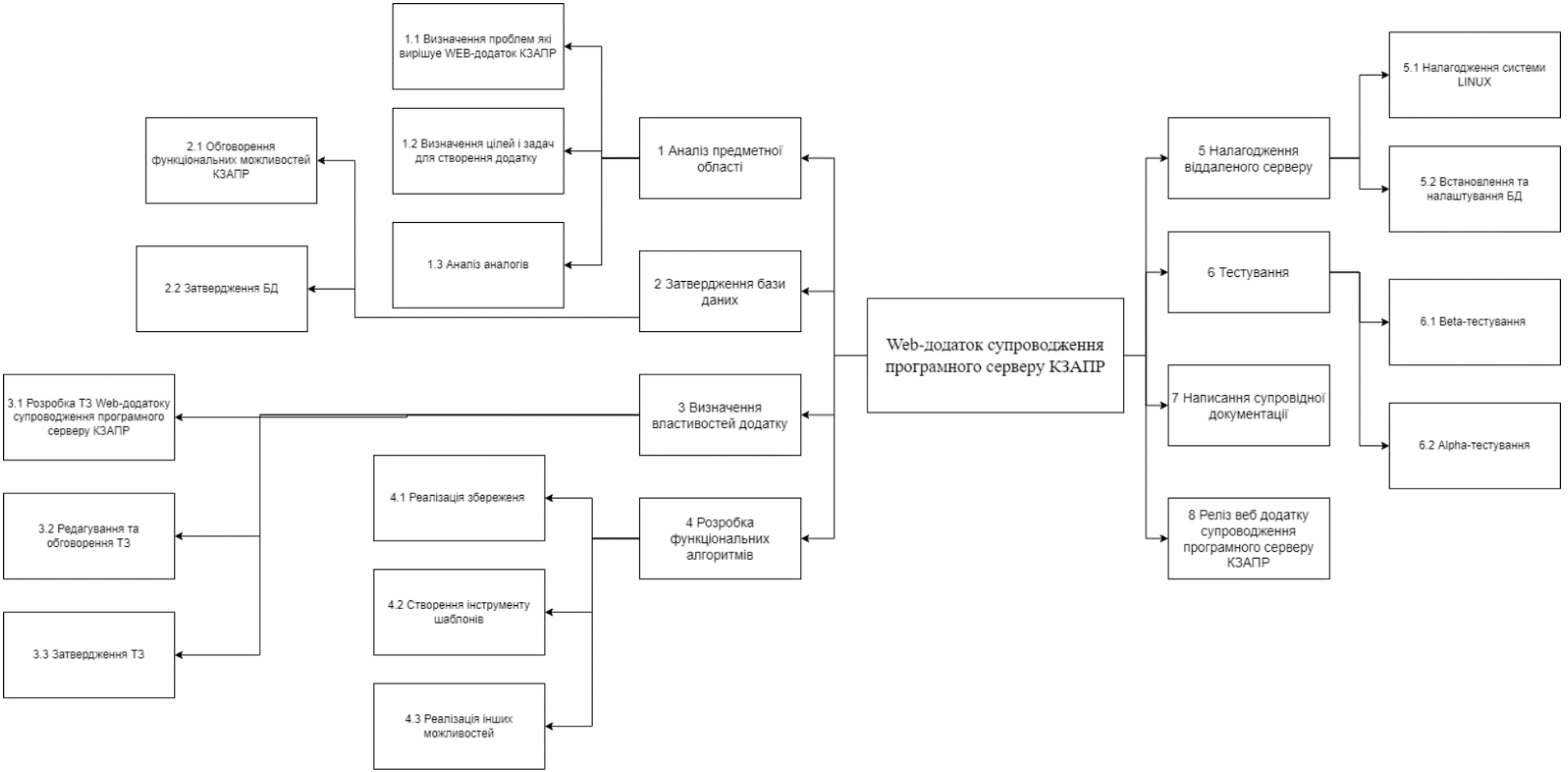

Рисунок Б.1 – WBS-структура робіт проекту

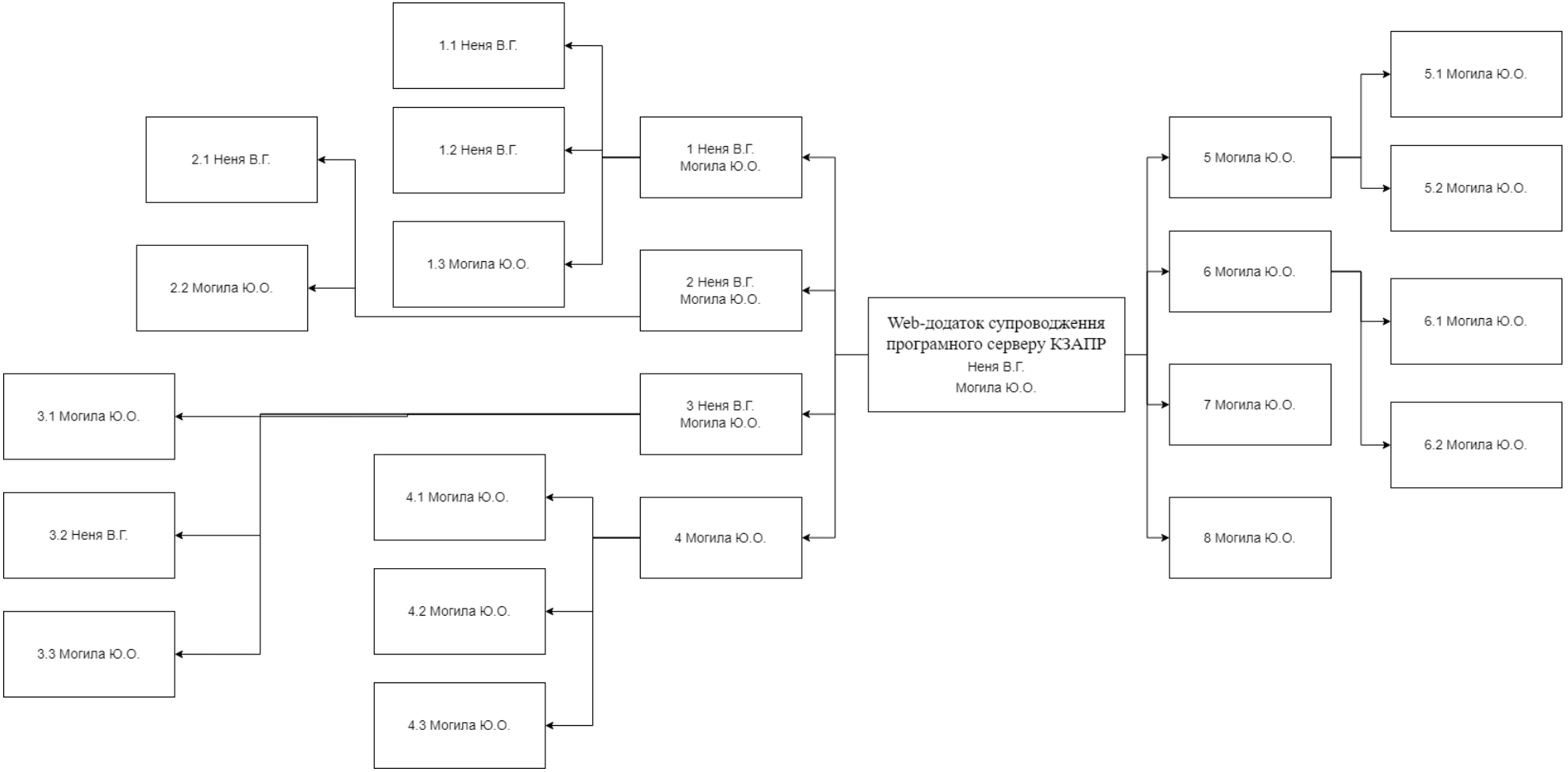

Рисунок Б.2 – OBS-структура робіт проект

Таблиця Б.2 – Виконавці проекту

Роль	**Ім'я**	**Проектна роль**
Розробник	Могила Ю.О.	Виконує back-end розробку
Проектувальник	Могила Ю.О.	Виконує проектування бази даних та розробляє структуру web-додатку.
Тестувальник	Могила Ю.О.	Відповідає за тестування функціоналу та дизайну web-додатку.
Керівник проекту	Неня В.Г.	Формує завдання на розробку проекту.
Менеджер проекту	Могила Ю.О.	Відповідає за виконання термінів, розподіл ресурсів та завдань між учасниками. Виконує збір та аналіз даних.

Діаграма Ганта. Побудова календарного графіку (діаграми Ганта) є одним з важливих етапів планування проекту, що виглядає як розклад виконання робіт з реальним розподілом дат. Завдяки йому можна отримати достовірне уявлення про тривалість процесів з обмеженнями у ресурсах, урахуванням вихідних днів та свят.

Календарний графік проекту представлено на рисунках Б.3-Б.4.

Ид.	Режим задачи	Название задачи	Длительност	Начало	Окончание	Предшественники	Названия ресурсов
1		**Web-додаток супроводження програмного серверу КЗАПР**	**90 дней**	**Вт 18.01.22**	**Пн 23.05.22**		
2		**Аналіз предметної області**	**9 дней**	**Вт 18.01.22**	**Пт 28.01.22**		
3		Визначення проблем які вирішує WEB-додаток КЗАПР	3 дней	Вт 18.01.22	Чт 20.01.22		
4		Визначення цілей і задач для створення додатку	3 дней	Пт 21.01.22	Вт 25.01.22	3	
5		Аналіз аналогів	3 дней	Ср 26.01.22	Пт 28.01.22	4	
6		**Визначення властивостей додатку**	**11 дней**	**Пн 31.01.22**	**Пн 14.02.22**	**2**	
7		Розробка ТЗ Web-додатоку супроводження програмного серверу КЗАПР	5 дней	Пн 31.01.22	Пт 04.02.22	5	
8		Редагування та обговорення ТЗ	3 дней	Пн 07.02.22	Ср 09.02.22	7	
9		Затвердження ТЗ	3 дней	Чт 10.02.22	Пн 14.02.22	8	
10		**Затвердеження бази даних**	**7 дней**	**Вт 15.02.22**	**Ср 23.02.22**	**6**	
11		Обговорення функціональних можливостей КЗАПР	3 дней	Вт 15.02.22	Чт 17.02.22	6	
12		Затвердження БД	4 дней	Пт 18.02.22	Ср 23.02.22	11	
13		**Розробка функціональних алгоритмів**	**21 дней**	**Чт 24.02.22**	**Чт 24.03.22**	**10**	
14		Реалізація збереженя	7 дней	Чт 24.02.22	Пт 04.03.22	12	
15		Створення інструменту шаблонів	7 дней	Пн 07.03.22	Вт 15.03.22	14	
16		Реалізація інших можливостей	7 дней	Ср 16.03.22	Чт 24.03.22	15	

Рисунок Б.3 – Календарний графік проекту

Ид.	Режим задачи	Название задачи	Длительност	Начало	Окончание	Предшественники	Названия ресурсов
17		**Налагодження віддаленого серверу**	**14 дней**	**Пт 25.03.22**	**Ср 13.04.22**	13	
18		Налагодження системи LINUX	7 дней	Пт 25.03.22	Пн 04.04.22	16	
19		Встановлення та налаштування БД	7 дней	Вт 05.04.22	Ср 13.04.22	18	
20		**Тестування**	**7 дней**	**Чт 14.04.22**	**Пт 22.04.22**	17	
21		Beta-тестування	3 дней	Чт 14.04.22	Пн 18.04.22	19	
22		Alpha-тестування	4 дней	Вт 19.04.22	Пт 22.04.22	21	
23		**Написання супровідної документації**	**7 дней**	Пн 25.04.22	Вт 03.05.22	20	
24		**Реліз Web-додатоку супроводження програмного серверу КЗАПР**	**7 дней**	Чт 12.05.22	**Пн 23.05.22**	23	

Рисунок Б.3 – Продовження календарного графіку проекту

Управління ризиками проекту.

Під час виконання якісної оцінки ризиків треба визначити ризики, які мають бути усунені якнайшвидше. В залежності від ступеня важливості ризику – реагування буде відповідне. Наступним етапом є виконання кількісного оцінювання ризиків. Кількісне та якісне оцінювання можуть виконувати одночасно або окремо, що залежить від ступеня забезпечення проекту. У таблиці Б.3 представлено шкалу для класифікації ризиків за величиною впливу на проект та ймовірністю виникнення.

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

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

Для зменшення негативного впливу ризиків на проект треба виконати

планування реагування на них. До нього входить визначення ефективності розробки та оцінка наслідків впливу на проект. Оцінювання виконується за показниками, що описані в таблиці Б.3. У результаті планування реагування було отримано матрицю ймовірності виникнення ризиків та впливу ризику, що зображена на рисунку Б.4.

Зеленим кольором на матриці позначають прийнятні ризики, жовтим – виправдані, а червоним – недопустимі.

Рисунок Б.4. – Матриця ймовірності

Класифікація ризиків за рівнем, відповідно до отриманого значення індексу, представлена у таблиці Б.4. У таблиці Б.5 описано ризики та стратегії реагування на кожен з них.

Таблиця Б.4 – Шкала оцінювання за рівнем ризику.

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

Таблиця Б.5 – Ризики та стратегії реагування

ID	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_1	Відкритий	Непорозуміння між розробником та замовником	Низька	Середній	3	1. Налагодити гарні відносини між розробником та керівником. 2. Дотримуватися ділового етикету спілкування. 3. Створити комфортні умови для співпраці	Попередження	При виявлення непорозуміння потрібно вияснити, що саме стало причиною непорозуміння обсудити її та створити здорову атмосферу в колективі.
RS_2	Відкритий	Низька кваліфікація розробників	Середня	Середній	4	1.Підвищити кваліфікацію персоналу. 2. Переглянути онлайн-ресурси для підвищення рівня знань.	Пом’якшення	Врахувати час на підготовку працівників. Видати літературу, переглянути онлайн-уроки.
RS_3	Відкритий	Нечітке завдання на розробку	Середня	Високий	6	1.Ясно і однозначно обговорити із замовником усі види вимог. 2.Скласти глосарій для запобігання розбіжностей у розумінні слів та термінів. 3.Періодичний контроль замовником етапів роботи.	Попередження	При виявленні невідповідностей деяких характеристик продукту заявленим вимогам потрібно уважно та чітко окреслити те, що було виконано невірно та зробити правки.

Продовження таблиці Б.5.

ID	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_4	Відкритий	Вибір не ефективної технології розробки	Середня	Високий	6	1.Проаналізувати методи та засоби, для виконання проекту. 2.Обрати зрозумілу та легку в використанні технологію розробки.	Пом'якшення	Виділити час та ресурсі на пошуки покращення обраної технології. Застосувати допоміжні ресурси.
RS_5	Відкритий	Неправильна оцінка в масштабі проекту	Низька	Середній	2	1.Провести детальний аналіз проекту. 2.Визначити основні етапу проекту, розподілити час на їх виконання. 3.Проаналізувати масштаби проекту на основі додаткових джерел.	Пом'якшення	Переоцінка масштабів проекту. Перебудова стратегії реалізації проекту.
RS_6	Відкритий	Помилки проектування	Висока	Високий	9	На етапі проектування тісно співпрацювати із замовником та на певних етапах демонструвати поточні результати.	Пом'якшення	Здійснювати проміжний контроль результатів в ході виконання проекту.

Продовження таблиці Б.5.

RS_7	Не ефективна технологія розробки	Низька	Середній	3	Провести аналіз методів розробки та обрати найпростішу	Пом’якшення	Проаналізувати основні методи розробки та обрати підходящий.
RS_8	Збої в роботі програми	Середня	Середній	4	Регулярне створення резервних копій.	Попередження	Зміна сервера та забезпечення
RS_9	Реалізація зайвого функціоналу	Низька	Низький	1	Слідування раніше створеній та узгодженій документації	Пом’якшення	
RS_10	Недостатня кількість тестового покриття	Середня	Середній	4	Залучення спеціалістів	Попередження	Створення тест плану та тест кейсів на початку розробки

ДОДАТОК В

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Web-додаток супроводження програмного серверу КЗАПР

Могила Ю.О, *студент*; Неня В.Г., доцент
Сумський державний університет, м. Суми, Україна

Застосування інформаційних технологій допомагає в проведенні розрахунків і автоматизації багатьох процесів машинобудівної галузі. Тому зростає попит на розробку комплексу засобів автоматизації проектувальних робіт (КЗАПР).

Сьогодні web-додатки є лідером серед багатьох існуючих рішень. Вони прості у використанні та доступності. Тому було вирішено розробити КЗАПР на платформі web-технологій. Серед переваг можна зазначити доступність із будь-якого місця за наявності підключення до мережі Інтернет та високу працездатність незалежно від обраної операційної системи. Метою даного дослідження є розробка інформаційної технології забезпечення підтримки виробничого процесу організації з проектування штучних об'єктів.

Для забезпечення злагодженої роботи всіх підсистем КЗАПР запропоновано єдиний для виконуваних завдань механізм супроводження програмного серверу, який відповідає за обробку та роботу з даними, отриманими з бази даних. Реалізація виконана мовою програмування Java. Її універсальність в роботі з будь-яким апаратним забезпеченням та наявність великої кількості бібліотек для розробки серверу визначили її пріоритетність.

Програмний сервер КЗАПР розробляється за архітектурою REST. У її основі лежить обмін даними між клієнтською та серверною частинами шляхом обміну даними за рахунок HTTP протоколу. Така реалізація КЗАПР надає можливість зручного масштабування та створення комплексу із різних апаратних платформ, наприклад, розробка мобільного або стаціонарного програмного додатку. Також розмежування відображення та логіки дозволяє виконувати налаштування та зміни в одній частині, не впливаючи на працездатність іншої.

Даний web-додаток прийнято рішення встановити на віддаленому сервері Oracle під управлінням операційної системи Linux. Остання використовується в більшості випадках на підставі підвищеної надійності та меншими потребами використання апаратних ресурсів у порівнянні з іншими операційними системами.

ДОДАТОК Г

ЛІСТИНГ ПРОГРАМНОГО КОДУ ОСНОВНИХ МОДУЛІВ

ContractController

```
@Log4j2
@RestController
@AllArgsConstructor
public class ContractController {

    @Autowired
    private final ContractRepository contractRepository;
    @Autowired
    private final ProfileRepository profileRepository;
    @Autowired
    private final StatusRepository statusRepository;
    @Autowired
    private final ProjectRepository projectRepository;
    @Autowired
    private final AttachmentListRepository attachmentListRepository;
    @Autowired
    private final TaskRepository taskRepository;
    //Get information about the contract by ID
    @GetMapping("/contract/{id}")
    public ContractByIdResponseDto getContractById(@PathVariable(value = "id") Long
contract_id) {
        Contract contract = contractRepository.getById(contract_id);
        ContractByIdResponseDto.Data.DataProfile customerProfile = new
ContractByIdResponseDto.Data.DataProfile(contract.getCustomer().getProfile_id(),
                contract.getCustomer().getName(), contract.getCustomer().getSurname(),

profileRepository.getUserEmailByProfileId(contract.getCustomer().getProfile_id()));
        ContractByIdResponseDto.Data.DataProfile ownerProfile = new
ContractByIdResponseDto.Data.DataProfile(contract.getOwner().getProfile_id(),
                contract.getOwner().getName(), contract.getOwner().getSurname(),

profileRepository.getUserEmailByProfileId(contract.getOwner().getProfile_id()));
        return new ContractByIdResponseDto("contract data",
                new ContractByIdResponseDto.Data(contract.getContract_id(),
contract.getTitle(), contract.getImage(),
                        contract.getDescription(), new
ContractByIdResponseDto.Data.DataProfile(customerProfile.getProfileId(),
customerProfile.getName(), customerProfile.getSurname(), customerProfile.getEmail()),
                        new
ContractByIdResponseDto.Data.DataProfile(ownerProfile.getProfileId(),
ownerProfile.getName(), ownerProfile.getSurname(), ownerProfile.getEmail()),

ContractByIdResponseDto.Data.attachedDataList(attachmentListRepository.getAttachmentLi
stByContractId(contract.getContract_id())),
                        new
ContractByIdResponseDto.Data.Dates(contract.getTime_start(), contract.getTime_end()),
                        new
ContractByIdResponseDto.Data.Status(contract.getStatus().getStatus_id(),
contract.getStatus().getName(), contract.getStatus().getColor())
                ), true);
    }

    @GetMapping("/contracts")
    public ContractsListResponseDto getUnclosedContracts() {
        return new ContractsListResponseDto("contracts list",

ContractsListResponseDto.toDataList(contractRepository.getUnclosedContractList()),
true);
    }
```

```
    @GetMapping("/project/{profile_id}/project-managers")
    public PmOfProjectDto getAllPmOfProject(@PathVariable(value = "profile_id") Long
profile_id) {
        return new PmOfProjectDto("projects for the contract",

PmOfProjectDto.toDataList(profileRepository.getProfileIdByProjectIdForPm(profile_id)),
true);
    }

    @GetMapping("/contract/{contractId}/projects")
    public ListProjectsOfContractDto getUnclosedContracts(@PathVariable(value =
"contractId") Long contract_id) {
        return new ListProjectsOfContractDto("projects for the contract",

ListProjectsOfContractDto.toDataList(projectRepository.getProjectsByContract(contract_
id)), true);
    }
    //Creating the contract
    @PostMapping("/contract")
    public ResponseDto createNote(@RequestBody ContractRequestDto contractRequestDto)
{
        Contract contract = contractRequestDto.toContract();
        if (contractRequestDto.getCustomerId() != null)

profileRepository.findById(contractRequestDto.getCustomerId()).ifPresent(contract::set
Customer);
        if (contractRequestDto.getCompanyOwnerId() != null)

profileRepository.findById(contractRequestDto.getCompanyOwnerId()).ifPresent(contract:
:setOwner);
        if (contractRequestDto.getStatus_id() != null)

statusRepository.findById(contractRequestDto.getStatus_id()).ifPresent(contract::setSt
atus);
        contractRepository.save(contract);
        log.info("Contract was saved: {}",contract);
        return new ResponseDto("contract create", true,
                new ResponseDto.DataDTO(contract.getContract_id(),
                        new
ResponseDto.DataDTO.Stat(contract.getStatus().getStatus_id(),
                                contract.getStatus().getName(),
contract.getStatus().getColor())));
    }
    //Change the contract information by ID
    @PutMapping("/contract/{contractId}")
    public ResponseDto changeContract(@RequestBody ContractRequestDto
contractRequestDto,@PathVariable(value = "contractId") Long contract_id) {
        if (!contractRepository.findById(contract_id).isPresent()) {
            return new ResponseDto("Contract with id" + contract_id + "not
found",false, new ResponseDto.DataDTO());
        }
        Contract contract = contractRepository.getById(contract_id);
        contract.setTitle(contractRequestDto.getTitle());
        contract.setImage(contractRequestDto.getImage());
        contract.setTime_start(contractRequestDto.getTime_start());
        contract.setTime_end(contractRequestDto.getTime_end());

        if (contractRequestDto.getCustomerId() != null)

profileRepository.findById(contractRequestDto.getCustomerId()).ifPresent(contract::set
Customer);
        if (contractRequestDto.getCompanyOwnerId() != null)

profileRepository.findById(contractRequestDto.getCompanyOwnerId()).ifPresent(contract:
:setOwner);
        if (contractRequestDto.getStatus_id() != null)

statusRepository.findById(contractRequestDto.getStatus_id()).ifPresent(contract::setSt
atus);
        contractRepository.save(contract);
```

```
        return new ResponseDto("change contract", true,
                new ResponseDto.DataDTO(contract.getContract_id(),
                        new
ResponseDto.DataDTO.Stat(contract.getStatus().getStatus_id(),
                                contract.getStatus().getName(),
contract.getStatus().getColor())));
    }
    //Get the completed task per the day of the week
    @GetMapping("/amount-of-the-complited-tasks")
    public TaskPerWeekDto getTasksPerWeek() {
        final LocalDate currentWeek = LocalDate.now().minusDays(7);
        final LocalDate endOfWeek = LocalDate.now();

        List<Task> tasks = taskRepository.getTasksByStatusName("Closed")
                .stream()
                .filter(task ->
currentWeek.isBefore(task.getTime_end().toLocalDateTime().toLocalDate()))
                .collect(Collectors.toList());

        List<TaskPerWeekDto.TaskPerWeek> taskPerWeek = new ArrayList<>();

        for (LocalDate iterator = currentWeek; iterator.isBefore(endOfWeek); iterator
= iterator.plusDays(1)) {
            DayOfWeek dayOfWeek = iterator.getDayOfWeek();
            long count = tasks.stream()
                    .filter(task ->
task.getTime_end().toLocalDateTime().getDayOfWeek().equals(dayOfWeek))
                    .count();

            taskPerWeek.add(new TaskPerWeekDto.TaskPerWeek(dayOfWeek.name(), (int)
count));
        }
        return new TaskPerWeekDto(taskPerWeek);
    }
}
```

UploadController

```
@RestController
public class UploadController {
    @Autowired
    private AttachmentRepository attachmentRepository;
    @Autowired
    private TaskRepository taskRepository;
    @Autowired
    private AttachmentListRepository attachmentListRepository;
    @Autowired
    private ProjectRepository projectRepository;
    @Autowired
    private ContractRepository contractRepository;
    //Save the uploaded file to this folder
    private static String UPLOADED_FOLDER = "/home/ubuntu/kzaprResourse/";
    //private static String UPLOADED_FOLDER = "C://temp//";
    //Delete attachments of the task
    @DeleteMapping("/task/{taskId}/attachment/{attachmentId}")
    public String deleteAttachment(@PathVariable long taskId, @PathVariable long
attachmentId){
        if (taskRepository.findById(taskId).isPresent() &&
attachmentRepository.findById(attachmentId).isPresent()){
            try {

Files.delete(Paths.get(attachmentRepository.getById(attachmentId).getLink()));
            } catch (IOException e) {
                e.printStackTrace();
            }
            attachmentRepository.deleteById(attachmentId);
            return "{\n" +
                    "    \"message\": \"Task file attachment deleted\",\n" +
                    "    \"data\": {\n" +
                    "         id: "+ attachmentId +",\n" +
                    "     },\n" +
```

```
                    "    \"success\": true,\n" +
                    "}";
        }
        return "{\n" +
                "    \"message\": \"Task or attachment not found\",\n" +
                "    \"success\": false,\n" +
                "}";
    }
    //Delete attachments of the project
    @DeleteMapping("/project/{project_id}/attachment/{attachmentId}")
    public   String   deleteAttachmentOfTheProject(@PathVariable   long   project_id,
@PathVariable long attachmentId){
        if            (projectRepository.findById(project_id).isPresent()            &&
attachmentRepository.findById(attachmentId).isPresent()){
            try {

Files.delete(Paths.get(attachmentRepository.getById(attachmentId).getLink()));
            } catch (IOException e) {
                e.printStackTrace();
            }
            attachmentRepository.deleteById(attachmentId);
            return "{\n" +
                    "    \"message\": \"Project file attachment deleted\",\n" +
                    "    \"data\": {\n" +
                    "         id: "+ attachmentId +",\n" +
                    "     },\n" +
                    "    \"success\": true,\n" +
                    "}";
        }
        return "{\n" +
                "    \"message\": \"Project or attachment not found\",\n" +
                "    \"success\": false,\n" +
                "}";
    }
    //Delete attachments of the contract
    @DeleteMapping("/contract/{contractId}/attachment/{attachmentId}")
    public   String   deleteAttachmentOfTheContract(@PathVariable   long   contractId,
@PathVariable long attachmentId){
        if            (contractRepository.findById(contractId).isPresent()            &&
attachmentRepository.findById(attachmentId).isPresent()){
            try {

Files.delete(Paths.get(attachmentRepository.getById(attachmentId).getLink()));
            } catch (IOException e) {
                e.printStackTrace();
            }
            attachmentRepository.deleteById(attachmentId);
            return "{\n" +
                    "    \"message\": \"Contract file attachment deleted\",\n" +
                    "    \"data\": {\n" +
                    "         id: "+ attachmentId +",\n" +
                    "     },\n" +
                    "    \"success\": true,\n" +
                    "}";
        }
        return "{\n" +
                "    \"message\": \"Contract or attachment not found\",\n" +
                "    \"success\": false,\n" +
                "}";
    }

    //Add new attachment to the task
    @PostMapping("/task/{taskId}/attachment")
    public String singleFileUpload(@PathVariable("file") MultipartFile file,
                                   RedirectAttributes  redirectAttributes,@PathVariable
long taskId) {
        if (file.isEmpty()) {
            redirectAttributes.addFlashAttribute("message",  "Please  select  a  file  to
upload");
            return "redirect:uploadStatus";
```

```
        }

        try {
            if (!taskRepository.findById(taskId).isPresent()) return "{\n" +
                    "    \"message\": \"Task not found\",\n" +
                    "    \"success\": false,\n" +
                    "}";
            // Get the file and save it somewhere
            byte[] bytes = file.getBytes();
            Path path = Paths.get(UPLOADED_FOLDER + file.getOriginalFilename());
            Files.write(path, bytes);
            Attachment attachment = new Attachment();
            attachment.setLink(path.toString());
            attachment.setType("file");
            attachmentRepository.save(attachment);
            AttachmentList attachmentList = new AttachmentList();
            attachmentList.setAttachment(attachment);
            attachmentList.setTask(taskRepository.getById(taskId));
            attachmentListRepository.save(attachmentList);
            String successString = "{\n" +
                    "    \"message\": \"Task attachment added\",\n" +
                    "    \"data\": {\n" +
                    "         link: "+ path.toString() +",\n" +
                    "         id: "+ attachment.getAttachment_id() +",\n" +
                    "    },\n" +
                    "    \"success\": true,\n" +
                    "}";
            redirectAttributes.addFlashAttribute("message",
                    "You successfully uploaded '" + file.getOriginalFilename() + "'");
            return successString;
        } catch (IOException e) {
            e.printStackTrace();
        }
        return "redirect:uploadStatus";
    }
    //Add new attachment to the project
    @PostMapping("/project/{projectId}/attachment")
    public String attachmentToTheProject(@PathVariable("file") MultipartFile file,
                                    RedirectAttributes  redirectAttributes,@PathVariable
long projectId) {
        if (file.isEmpty()) {
            redirectAttributes.addFlashAttribute("message", "Please select a file to
upload");
            return "redirect:uploadStatus";
        }
        try {
            if (!projectRepository.findById(projectId).isPresent()) return "{\n" +
                    "    \"message\": \"Project not found\",\n" +
                    "    \"success\": false,\n" +
                    "}";
            byte[] bytes = file.getBytes();
            Path path = Paths.get(UPLOADED_FOLDER + file.getOriginalFilename());
            Files.write(path, bytes);
            Attachment attachment = new Attachment();
            attachment.setLink(path.toString());
            attachment.setType("file");
            attachmentRepository.save(attachment);
            AttachmentList attachmentList = new AttachmentList();
            attachmentList.setAttachment(attachment);
            attachmentList.setProject(projectRepository.getById(projectId));
            attachmentListRepository.save(attachmentList);
            String successString = "{\n" +
                    "    \"message\": \"Project attachment added\",\n" +
                    "    \"data\": {\n" +
                    "         link: "+ path.toString() +",\n" +
                    "         id: "+ attachment.getAttachment_id() +",\n" +
                    "    },\n" +
                    "    \"success\": true,\n" +
                    "}";
            redirectAttributes.addFlashAttribute("message",
```

```
                    "You successfully uploaded '" + file.getOriginalFilename() + "'");
            return successString;
        } catch (IOException e) {
            e.printStackTrace();
        }
        return "redirect:uploadStatus";
    }
    //Add new attachment to the contract
    @PostMapping("/contract/{contractId}/attachment")
    public String attachmentToTheContract(@PathVariable("file") MultipartFile file,
                                          RedirectAttributes
redirectAttributes,@PathVariable long contractId) {
        if (file.isEmpty()) {
            redirectAttributes.addFlashAttribute("message", "Please select a file to
upload");
            return "redirect:uploadStatus";
        }
        try {
            if (!contractRepository.findById(contractId).isPresent()) return "{\n" +
                    "    \"message\": \"Contract not found\",\n" +
                    "    \"success\": false,\n" +
                    "}";
            byte[] bytes = file.getBytes();
            Path path = Paths.get(UPLOADED_FOLDER + file.getOriginalFilename());
            Files.write(path, bytes);
            Attachment attachment = new Attachment();
            attachment.setLink(path.toString());
            attachment.setType("file");
            attachmentRepository.save(attachment);
            AttachmentList attachmentList = new AttachmentList();
            attachmentList.setAttachment(attachment);
            attachmentList.setContract(contractRepository.getById(contractId));
            attachmentListRepository.save(attachmentList);
            String successString = "{\n" +
                    "    \"message\": \"Contract attachment added\",\n" +
                    "    \"data\": {\n" +
                    "         link: "+ path.toString() +",\n" +
                    "         id: "+ attachment.getAttachment_id() +",\n" +
                    "    },\n" +
                    "    \"success\": true,\n" +
                    "}";
            redirectAttributes.addFlashAttribute("message",
                    "You successfully uploaded '" + file.getOriginalFilename() + "'");
            return successString;
        } catch (IOException e) {
            e.printStackTrace();
        }
        return "redirect:uploadStatus";
    }
}
```