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

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: «Інформаційна система оцінки професійних досягнень співробітників ІТ-компанії»

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

Виконавець роботи: студентка групи ІТ.м-12 Чмутенко Анна Вадимівна

Кваліфікаційну роботу захищено на засіданні ЕК з оцінкою _____ «__» грудня 2022 р.

Науковий керівник _____ к.т.н., доц., Антипенко В.П.
(підпис)

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
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Інформаційні технології проектування»

ЗАТВЕРДЖУЮ

В. о. зав. кафедри ІТ
_____С. М. Ващенко

«___» _____ 2022 р.

ЗАВДАННЯ

на кваліфікаційну роботу магістра студентові

Чмутенко Анна Вадимівна
(прізвище, ім'я, по батькові)

1 Тема проекту *Інформаційна система оцінки професійних навичок співробітників ІТ-компанії*

затверджена наказом по університету від «04» листопада 2022 р. № 01013-VI

2 Термін здачі студентом закінченого проекту «___» ___грудня___ 2022 р.
3 Вхідні дані до проекту технічне завдання на розробку інформаційної системи

4 Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) аналіз предметної області, постановка задачі, методи дослідження, проектування інформаційної системи оцінки професійних досягнень співробітників ІТ-компанії, розробка інформаційної системи

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
актуальність проблеми, мета дипломної роботи, задачі проекту, функціональні та технічні вимоги до системи, проектування інформаційної системи, проектування, етапи розробки програмного продукту, практична реалізація, висновки, практичне значення, оприлюднення результатів роботи.

6. Консультанти випускної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

Дата видачі завдання _____.

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів випускної проекту	Термін виконання етапів проекту	Примітка
1	Планування робіт	до 02.11.2022	
2	Проведення аналізу предметної області	до 10.11.2022	
3	Проведення структурно-функціонального моделювання	до 15.11.2022	
4	Реалізація структури інформаційної системи	до 20.11.2022	
5	Розробка функціональної частини ІС	до 25.11.2022	
6	Оформлення та здача пояснювальної записки та файлів розробленого проекту	до 10.12.2022	

Магістрант _____ Чмутенко А.В.

Керівник роботи _____ к.т.н., доц. Антипенко В.П.

РЕФЕРАТ

Тема кваліфікаційної роботи магістра «Інформаційна система оцінки професійних досягнень співробітників ІТ-компанії».

Пояснювальна записка складається зі вступу, 4 розділів, висновків, списку використаних джерел із 15 найменувань, 2 додатків. Загальний обсяг роботи − 77 сторінок, у тому числі 50 сторінок основного тексту, 2 сторінки списку використаних джерел, 24 сторінки додатків.

Кваліфікаційну роботу магістра присвячено розробці інформаційної системи оцінки професійних досягнень співробітників ІТ-компанії. У роботі проведено аналіз предметної області. Також визначено потребу в розробці інформаційної системи. Обрано технології для програмної реалізації, поставлено мету проекту з визначенням переліку задач та описано методи дослідження. У роботі виконано проектування системи на різних рівнях, враховуючи бізнес-призначення та варіанти виристання програмного продукту. Розроблено концептуальну модель бази даних. Здійснена програмна реалізація спроектованої моделі продукту проекту. Результатом даного дослідження є розроблена інформаційна система оцінки професійних досягнень співробітників ІТ-компанії. Використання даної ІС сприятиме підтримці як кар'єрного росту виконавців, так і їх всебічному розвитку. У свою чергу представлені дані є підґрунтям для призначення тим чи іншим працівникам певних заходів щодо підвищення їх кваліфікації, здійснення перегляду заробітної плати та знаходження спеціалістів для певного проекту відповідно до навичок та знань серед співробітників ІТ-компанії.

Ключові слова: ІНФОРМАЦІЙНА СИСТЕМА, ОЦІНКА, ПРОФЕСІЙНІ НАВИЧКИ, ОСОБИСТІСНІ НАВИЧКИ, LARAVEL, PHP, JAVASCRIPT

ЗМІСТ

ВСТУП

Актуальність. Наразі питання оцінки професійних досягнень та особистісних якостей співробітників компанії будь-якої сфери життєдіяльності людини вважається актуальним. Особливо це стосується сфери інформаційних технологій (ІТ), адже вони невпинно розвиваються. Постійне навчання та самовдосконалення для спеціалістів напряму комп'ютерних технологій – це ключ до кар'єрного та професійного зростання. Тому було ухвалено розробити інформаційну систему, використання якої дозволило би оцінити прогрес розвитку hard- та soft-skills співробітника. Це у подальшому створить підґрунтя для прийняття відповідних рішень щодо заходів підвищення кваліфікації працівника, перегляд його заробітної плати, залучення до певного проекту тощо.

Тема. Інформаційна система оцінки професійних досягнень співробітників ІТ-компанії.

Мета. Розробка інформаційної системи для проведення доцільної оцінки професійних досягнень співробітників ІТ-компанії за рахунок належної організації даного процесу.

Для досягнення мети проекту необхідно виконати наступні задачі:

- виконати детальний аналіз проблемної області та огляд сучасних публікацій, визначити актуальність та цільову аудиторію використання представленої ІС;
- провести аналіз аналогів інформаційних систем;
- визначити технології на розробку інформаційної системи;
- виконати проектування моделі та структури інформаційної системи;
- реалізувати структуру інформаційної системи;
- реалізувати функціональні можливості для інформаційної системи оцінки професійних досягнень співробітників ІТ-компанії;
- провести smoke-тестування інформаційної системи.

Об’єкт дослідження. Процес проведення оцінки професійних досягнень співробітників ІТ-компанії.

Предмет дослідження. Інформаційна система оцінки професійних досягнень працівників ІТ-компанії.

Практична новизна. Скорочення часу на підготовку даних по кожному співробітнику ІТ-компанії та занесення їх до інформаційної системи «RateIT» для всебічної оцінки їх професійних досягнень за певний період часу. Використання даної ІС сприятиме підтримці як кар’єрного росту виконавців, так і їх всебічному розвитку. У свою чергу представлена оцінка є підґрунтям для призначення тим чи іншим працівникам певних заходів щодо підвищення їх кваліфікації, здійснення перегляду заробітної плати та знаходження спеціалістів для певного проекту відповідно до навичок та знань серед співробітників ІТ-компанії.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд останніх досліджень і публікацій

Використання інформаційних технологій сьогодні перетворилося з привілеї у необхідність. Зараз важко навіть уявити повсякденне життя без результатів науково-технічного прогресу. Тому початковим етапом проекту було проведення огляду останніх досліджень і публікацій з метою виявлення потреб бізнесу та суспільства.

У [1] описано сучасні методи оцінки персоналу та надається аргументація важливості використання комбінацій різних методів. Адже у наслідок діджиталізації суспільства ефективність роботи працівників компаній значно впала. Також збільшилась кількість випадків вигорання штату на роботі. Тому питання своєчасного реагування на професійну відповідність, стан соціальних навичок та аналіз кар'єрного росту співробітників компанії є актуальним на сьогодні.

Важливість розвитку професійної компетентності персоналу на управлінських посадах було досліджено у [2]. Сучасні IT-компанії зазвичай мають широку ієрархію. Тому кожен працівник, незалежно від посади, повинен мати навички управління персоналом. Наприклад, продуктова компанія працює над безліччю проектів. У кожного з них є свої департаменти і т.д. Тому великий відсоток персоналу будь-якої організації тим чи іншим чином буде виконувати управлінські задачі.

Тема впливу мотивації співробітників компаній на їх показники продуктивності широко розкрита у [3]. У статті описано діючу систему нагородження працівників основного та середнього виробництва на промисловому підприємстві, та адміністративний персонал. Також було представлено розрахунковий план виконання грошового матеріального заохочення працівників у звітному році.

Тому тема саморозвитку, навчання, мотивації та оцінювання професійних навичок є надзвичайно актуальна. Розробка інформаційної системи для підтримки організації процесів аналізу досягнень та навичок, обробки запитів на перегляд заробітної плати, ініціації зміни проекту є доречною для даної предметної області.

1.2 Аналіз інформаційних систем

Сучасні реалії вимагають повсюдне використання інформаційних технологій у всіх сферах життєдіяльності людини аби забезпечити ефективність, швидкість та якість комунікацій. Щодня виконується безліч процесів та операцій у будь-які організації незалежно від її призначення, які потребують моментального реагування. Саме тому підтримка процесів прийняття рішень щодо підвищення заробітної плати, призначення на новий проект та зміна позиції співробітників IT-компанії є актуальним питанням [4].

Інформаційні системи оцінки професійних якостей співробітників організації користуються популярністю серед компанії різного виду діяльності. Відповідальні власники будь-якого бізнесу піклуються про внутрішніх та зовнішніх споживачів, професійне зростання кожного робітника й розвиток компанії в цілому.

Для порівняльного аналізу було відібрано дві інформаційні системи оцінки професійних якостей персоналу. А саме інформаційна система «Оцінка персоналу» та сервіс «Hurma» [5,6].

До структури програмного продукту «Оцінка персоналу» входять такі компоненти, як технологічна платформа 1С:Підприємство, оцінка компетенцій методом 360 градусів, психологічне тестування та сумісність команд, створення нових тестів із тестуванням та оцінкою, оцінка персоналу за KPI та розрахунок грейдів (рис.1.1-1.2).

Основні функціональні можливості інформаційної «Оцінка персоналу» наступні:

- атестація персоналу;
- оцінка кандидатів при прийомі на роботу чи зміні позиції співробітника;
- визначення слабких та сильних сторін команд проектів;
- моніторинг соціально-психологічного клімату в компанії;
- оцінка роботи співробітників за системою показників ефективності.

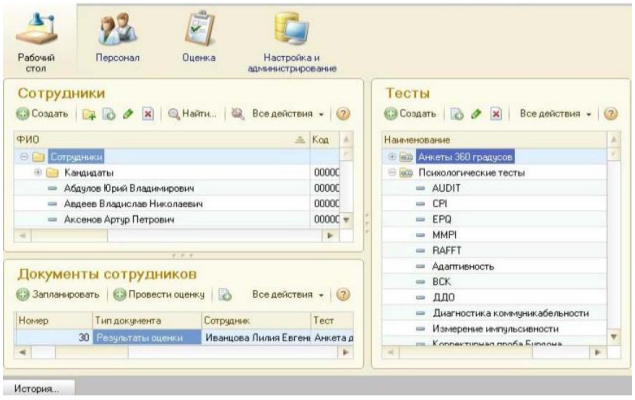


Рисунок 1.1 – Головне вікно інформаційної системи «Оцінка персоналу»

Інформаційна система має хоч простий та зрозумілий, але застарілий інтерфейс, який користувався попитом у 1990-х і на початку 2000-х. Кольорова гамма та шрифти не відповідають стандартам UI/UX [7].

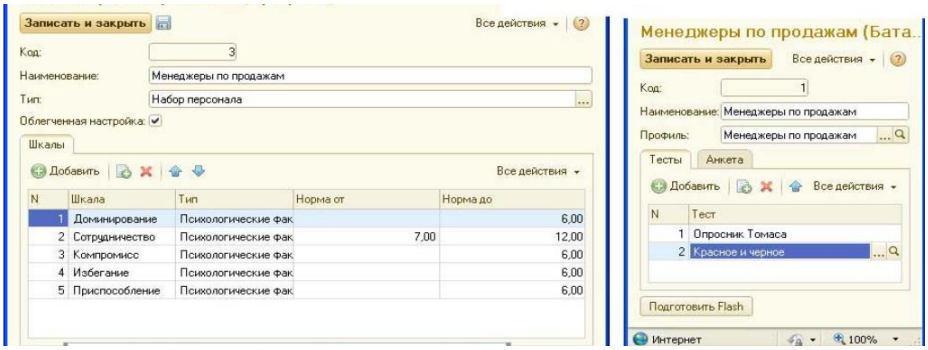


Рисунок 1.2 – Вікно тестування інформаційної системи «Оцінка персоналу»

Після проведення дослідження даної інформаційної системи, можна зробити висновок, що з першого погляду дійсно є наявним широкий набір розумних функцій, але у результаті отримуємо інформаційну систему, яка не вирішує конкретну бізнес-задачу.

Web-додаток «Hurma» була розроблена для HR-спеціалістів для оптимізації моніторингу управлінських задач і автоматизації процесу найму нових робітників компанії (рис.1.3-1.5). Даний сервіс користується великим попитом серед компаній різного напряму.

Серед функцій web-додатку «Hurma» варто виділити такі:

- робота з організаційною структурою компанії;
- автоматизація запитів до менеджера управління персоналом від співробітників компанії;
- обробка зворотного зв’язку;
- розрахунок заробітної плати;
- наявність особистого кабінету співробітника;
- оцінка продуктивності персоналу;
- систематичне оповіщення працівників компанії.

Серед переваг варто зазначити, що даний додаток має зручний, сучасний та привітний до користувача інтерфейс, інструменти аналітики для моніторингу плинності кадрів, автоматичний розрахунок заробітної плати з урахуванням відсутностей тощо. Але за рахунок великої кількості блоків та інформації на сторінках швидкодія web-додатку не на найвищому рівні.

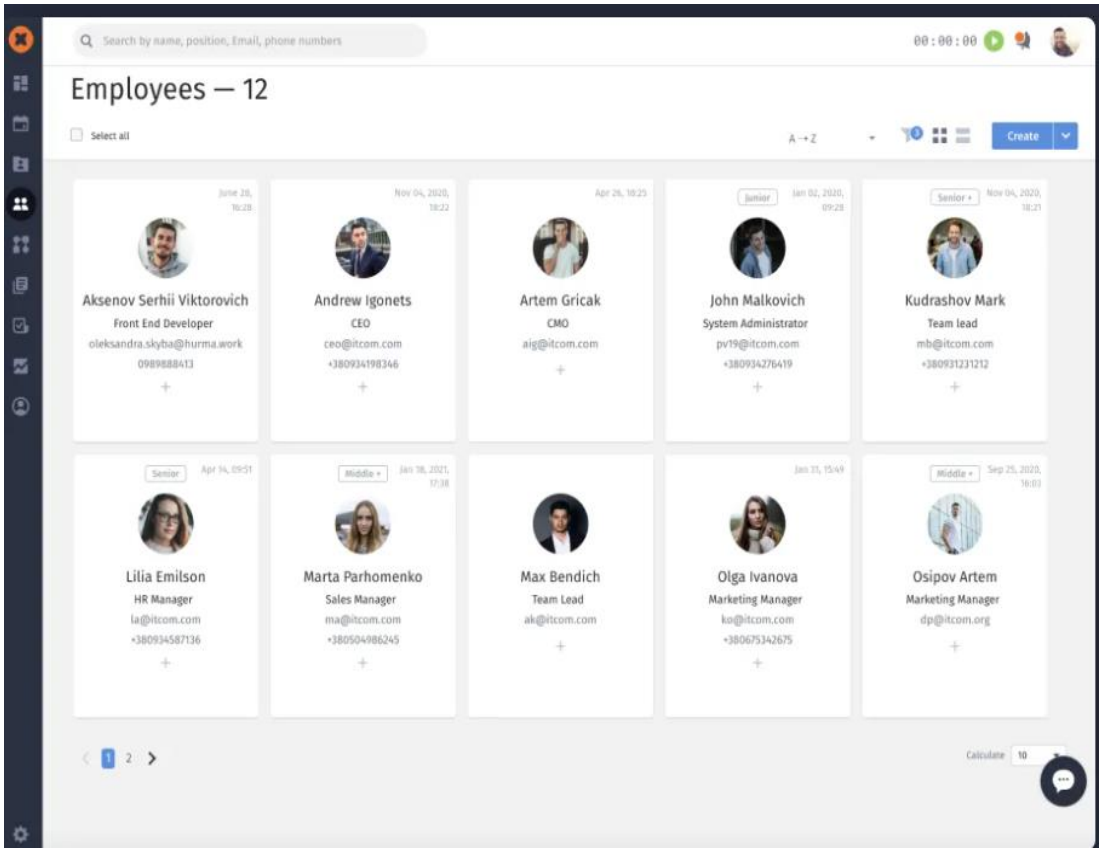


Рисунок 1.3 – Сторінка перегляду робітників компанії web-додатку «Hurma»

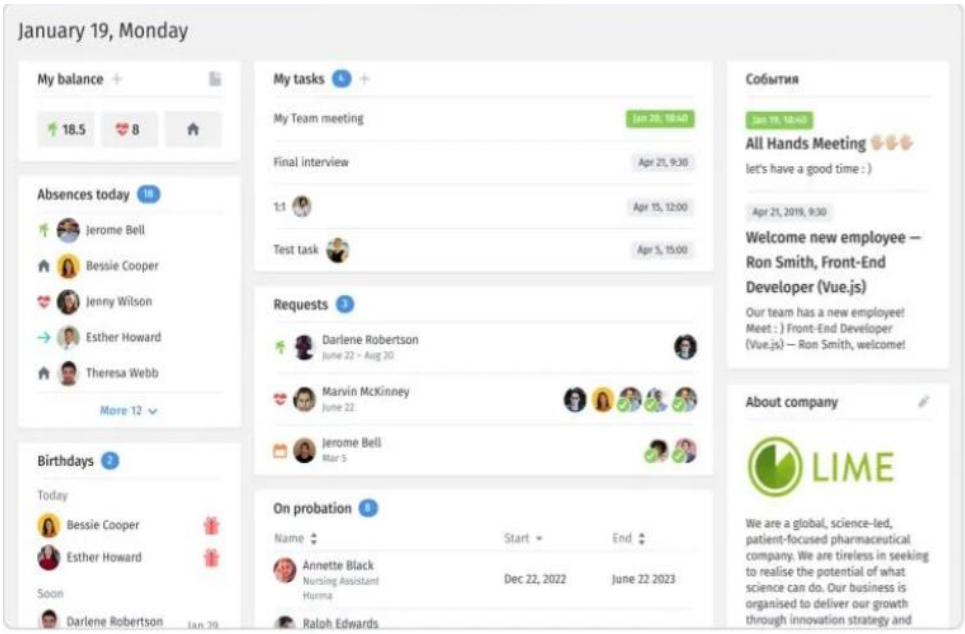


Рисунок 1.4 – Головна сторінка користувача

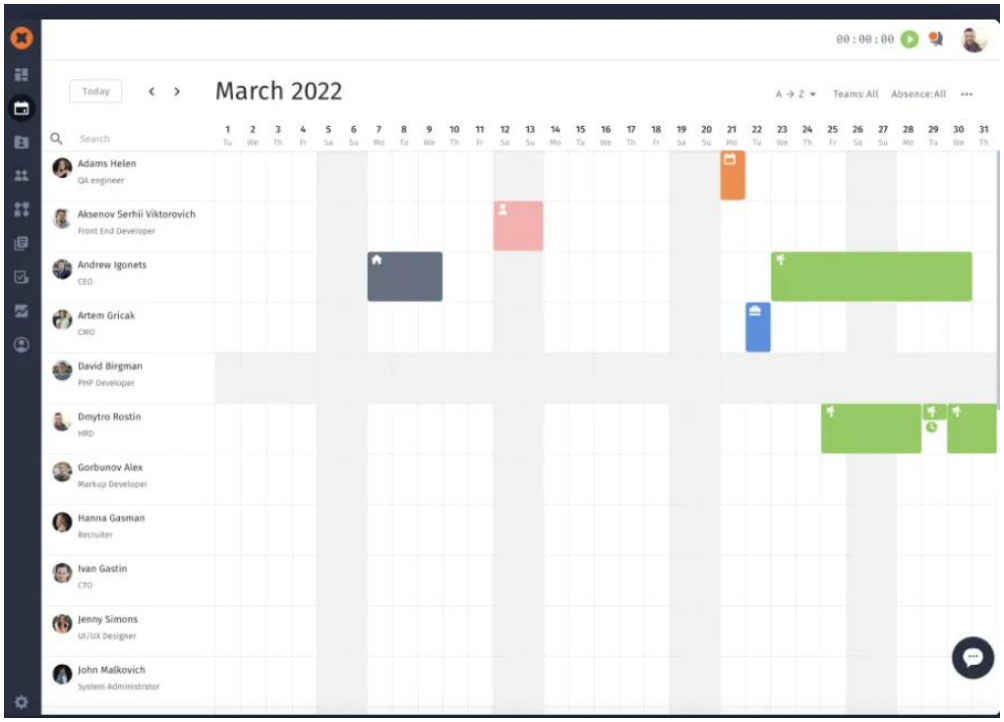


Рисунок 1.5 – Сторінка календаря відсутностей працівників

Результатом порівняльного аналізу двох вищезазначених програмних рішень є таблиця 1.1.

Таблиця 1.1 – Порівняльна таблиця аналогів інформаційних систем

Характеристика/Інформаційна система	**«Оцінка персоналу»**	**«Hurma»**
Сучасний дизайн	-	+
Зручний інтерфейс та навігація	-	+
Вирішення бізнес-проблеми	+	+
Наявність моніторингу аналітики	+	+
Можливість обробки заявок на підвищення заробітної плати	-	-
Можливість обробки заявки на зміну позиції чи ролі	-	-

Продовження таблиці 1.1

Характеристика/Інформаційна система	**«Оцінка персоналу»**	**«Hurma»**
Підбір співробітників на певний проект	+	-
Оцінка професійних навичок персоналу	-	-
Оцінка особистісних якостей співробітників	+	-
Можливість завантаження сертифікатів про проходження курсів	-	-
Особистий кабінет робітника	+	+

Дані з таблиці 1.1 надають можливість виділити актуальні функціональні рішення, які можна використати у власній розробці, та недоліки, які варто уникнути. Отже, створюваний програмний продукт повинен мати простий, сучасний та зручний інтерфейс, забезпечувати вирішення бізнес-задачі на обробку запитів по зміні позиції, зміна заробітної плати працівників. Важливо зазначити, що розроблюваний сервіс повинен надавати можливість моніторингу аналітики, а саме розвитку співробітників за відділами та напрямками та ріст заробітної плати з плином часу.

2 ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ

2.1 Мета та задачі дослідження

Метою кваліфікаційної роботи магістра є організація процесу оцінювання досягнень співробітників IT-компанії за рахунок впровадження створюваної інформаційної системи. Розроблене програмне рішення буде мати попит серед IT-компаній завдяки можливості надати загальну оцінку рівня знань та вмінь робітників за допомогою проходження ними відповідних тестів та отриманих сертифікатів за рівнем складності та важливості відповідно до позиції та департаменту. Упровадження створюваного сервісу в IT-компанії дозволить зменшити витрати ресурсів на збір даних про співробітників, відгуків на проектах та прийняття рішень про підвищення заробітної плати та/або позиції.

Розроблена інформаційна система має задовольняти наступні вимоги до функціональності:

- обробка запиту на підвищення заробітної плати;
- обробка запиту на зміну позиції та ролі робітника компанії;
- можливість призначення зміни проекту для співробітника;
- функція написання відгуку про працівника;
- можливість проведення тестування професійних та особистісних навичок робітників;
- можливість завантаження сертифікатів про проходження курсів на інших платформах;
- діаграма залежності професійного росту та заробітної плати працівника компанії.

Створена інформаційна система повинна мати простий та зрозумілий дизайн, із використанням мінімальної кількості візуальних ефектів. Адже її призначенням є не залучення нових клієнтів для організації, а оптимізація функціонування компанії. Варто зазначити, що всі розроблені матеріали повинні легко сприйматися та швидко завантажуватися на сторінках інформаційної системи.

Для досягнення мети даного проекту необхідно виконати такі задачі:

- провести аналіз аналогів інформаційних систем;
- визначити технології на розробку інформаційної системи;
- виконати проектування моделі та структури інформаційної системи;
- реалізувати структуру інформаційної системи;
- реалізувати функціональні можливості для інформаційної системи оцінки професійних досягнень співробітників ІТ-компанії;
- провести smoke-тестування інформаційної системи

Під час формалізації мети проекту було проведено планування робіт проекту, яке детально описано у Додатку А.

2.2 Методи дослідження

Після постановки мети проекту, визначення функціональних та нефункціональних вимог до розроблюваної інформаційної системи для оцінки професійних досягнень співробітників ІТ-компанії, формалізації переліку обов’язкових на виконання задач проекту, треба було визначити методи даного дослідження.

Існують наступні чотири основні методи проведення дослідження:

- теоретичний метод;
- системно-функціональний метод;

- емпіричний метод;
- метод моделювання [8].

Теоретичний метод полягає у більш детальному аналізі особливостей та можливостей предметної області. Під час роботи над проектом «Інформаційна система оцінки професійних досягнень співробітників ІТ-компанії» теоретичний метод було використано для аналізу бізнес-процесів, потреб та тенденцій ІТ-галузі. Також при виборі інструментів на розробку було застосовано теоретичний метод. Адже він є доцільним щоб виокремити переваги та недоліки тих чи інших технологій, їх особливості застосування у залежності від розміру, тривалості, методів збереження даних програмного продукту тощо.

Використання емпіричного методу полягає у застосуванні спостережень, вимірювань та експериментів у якості методик пізнання предметної області. Він використовувався на етапі аналізу процедур зміни позиції та перегляду заробітної плати, спираючись на особистий досвід та інтерв'ю колег у ІТ-компанії.

До методу моделювання входить розробка схем та діаграм, що показують процес роботи над реалізацією програмного продукту та яким чином даний програмний продукт буде використовуватися надалі. Розглядаючи його у контексті роботи над проектом «Інформаційна система оцінки професійних досягнень співробітників ІТ-компанії», то він застосовувався на етапі побудови діаграм IDEF0 та UseCase.

Системно-функціональний метод – це підхід до опису системи, під час застосування якого досліджуються її компоненти, залежності та взаємодія у рамках єдиного цілого. Він був використаний при розробці програмного модулю проекту. А саме здійснення варіантів використання, зображених на діаграмі UseCase [9].

Для програмної реалізації запропонованої інформаційної системи було обрано такі технології, як HTML для створення шаблону web-сторінок, каскадні таблиці стилів CSS для забезпечення адаптивності та налаштування візуальних ефектів, скриптову мову програмування JavaScript для надання динамічності web-сторінкам та створення діаграм, web-фреймворк Laravel як основа для створення бекенду даного програмного продукту. Для організації збереження даних представленої ІС у було обрано системи управління базами даних MySQL [10].

3 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Після проведення аналізу предметної області, визначення актуальності роботи та бізнес-потреби, постановки мети та задач на розробку, завершення процесу планування робіт на реалізацію проекту, було вирішено перейти до наступного етапу – це проектування даної інформаційної системи. Було розроблено діаграми нотації IDEF0, детально описано послідовність процесів, які беруть участь у автоматизації обробки заявок на перегляд заробітної плати, зміни позиції, підбір персоналу на проект та моніторинг тенденцій розвитку співробітників IT-компанії.

3.1 Діаграми нотації IDEF0

Для моделювання функцій роботи запропонованої ІС та організації бізнес-процесів використовуються різні методи об’єктно-орієнтованого та структурного підходу. Однією з таких методологій є IDEF0. Вона реалізується за допомогою графічного опису системи та процесів діяльності компанії як набір взаємопов’язаних функцій. Завдяки застосуванню IDEF0 можна дослідити функції окремо від об’єктів, які будуть ними користуватися. Діаграма містить такі дані, як вхідні, керування, вихідні та механізму [11].

Для контекстної діаграми було визначено наступні дані інформаційної системи оцінки професійних досягнень співробітників IT-компанії:

1. Вхідні дані: запит на перегляд заробітної плати, запит на підбір персоналу на проект, запит на зміну позиції, запит на зміну ролі, запит на зміну проекту.

2. Управління: результати проходження тестів для оцінки професійних навичок, відгук менеджера проекту, результати проходження сторонніх курсів, результати проходження тестів для оцінки особистісних якостей.

3. Вихідні дані: прийнято рішення щодо перегляду заробітної плати, перелік співробітників на певний проект, прийнято рішення щодо зміни позиції, прийнято рішення щодо зміни ролі чи проекту.

4. Механізми: інформаційна система, користувач, апаратне забезпечення.

На рисунках 3.1-3.5 зображена контекстна діаграма IDEF0 інформаційної системи оцінки професійних навичок співробітників IT-компанії.

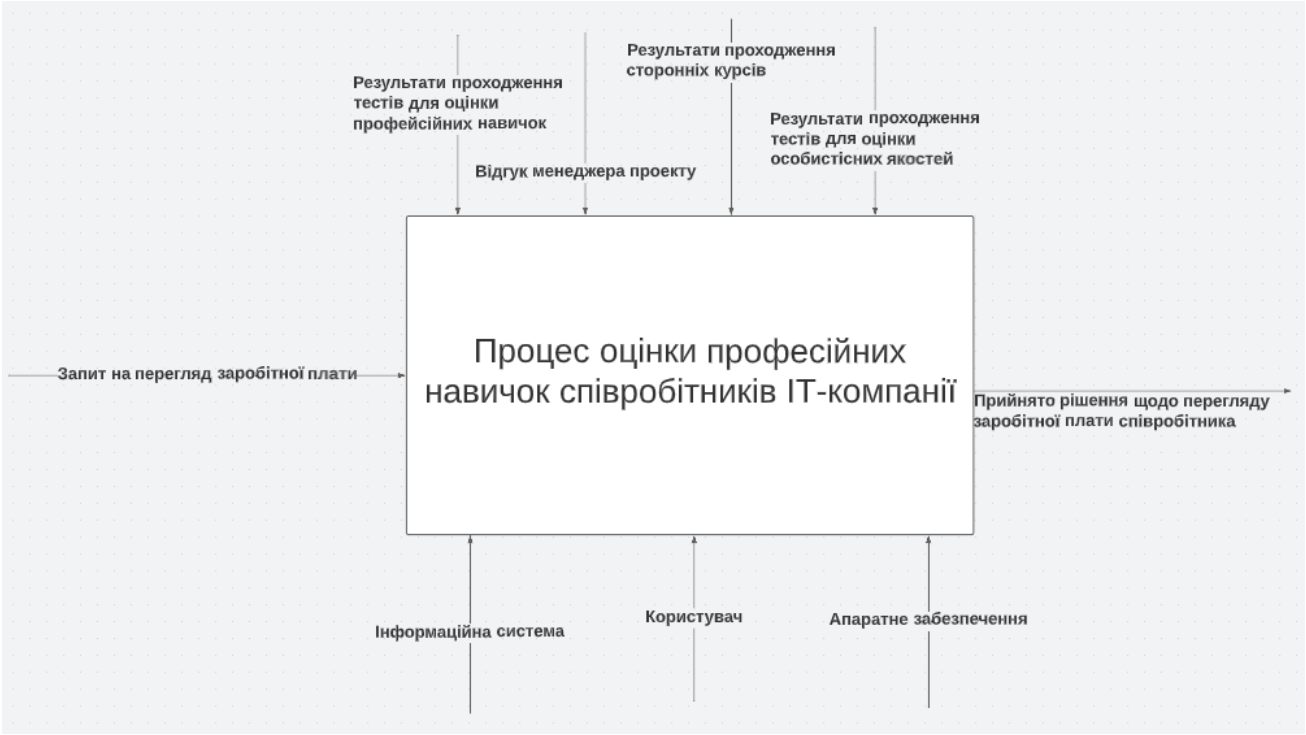


Рисунок 3.1 – Контекстна діаграма IDEF0 частина 1

Рисунок 3.2 – Контекстна діаграма IDEF0 частина 2

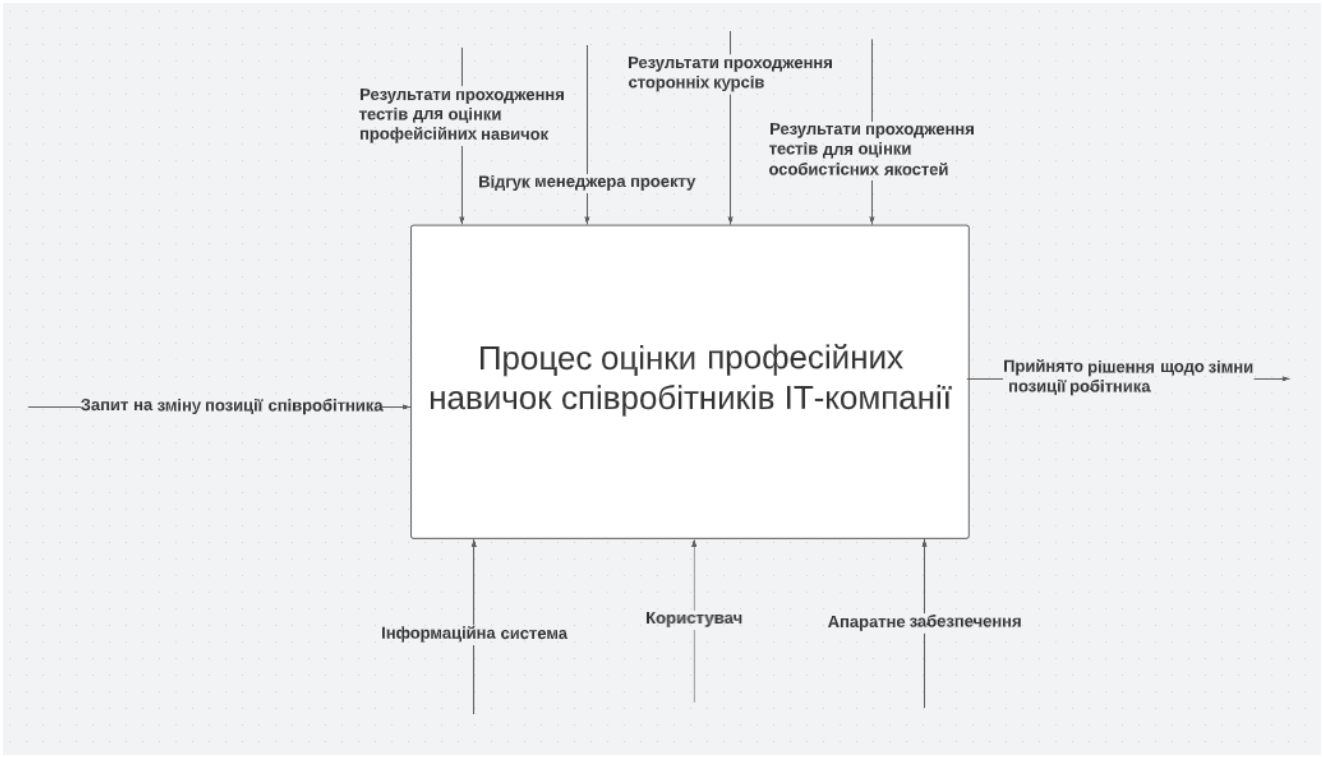


Рисунок 3.3 – Контекстна діаграма IDEF0 частина 3

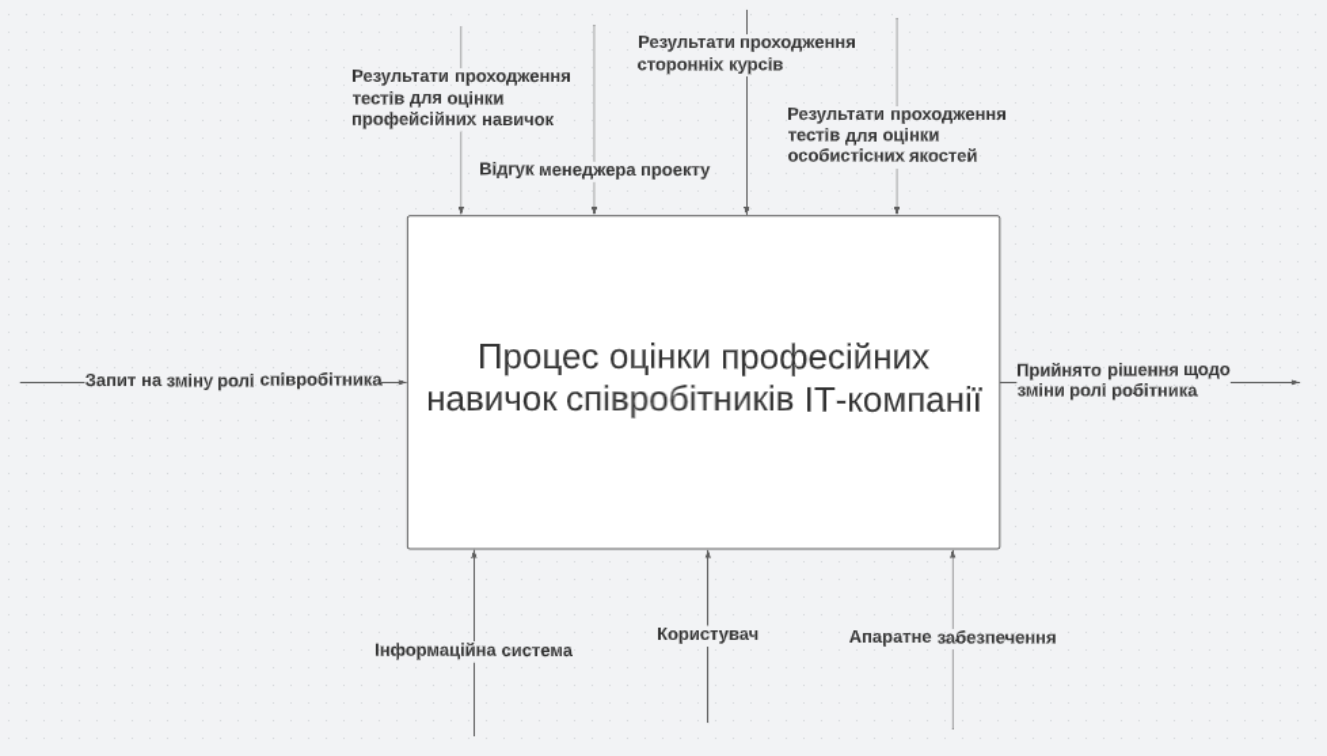


Рисунок 3.4 – Контекстна діаграма IDEF0 частина 4

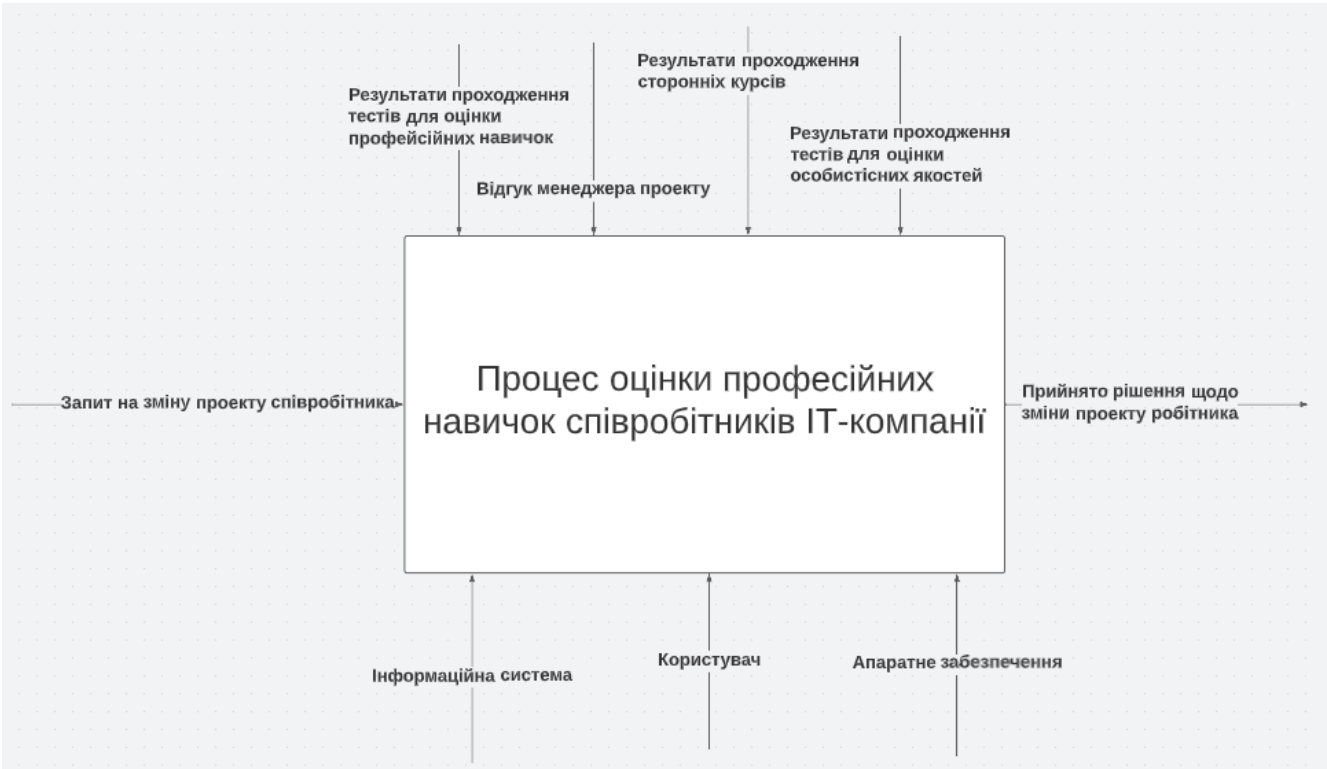


Рисунок 3.5 – Контекстна діаграма IDEF0 частина 5

Для деталізації процесів інформаційної системи було виконано декомпозицію першого рівня для процесів перегляду заробітної плати, підбору персоналу, зміна позиції та ролі робітника (рис.3.6-3.10).

Для детального представлення інформації було створено таблиці входів та виходів даних (табл. 3.1-3.5) для кожного підпроцесу.

Декомпозиція діаграми перегляду заробітної плати робітника представлена такими підпроцесами:

1. Уведення даних про робітника.
2. Прикріплення відгуку від менеджера про досягнення робітника на проекті.
3. Додавання сертифікатів про проходження сторонніх курсів для підвищення кваліфікації.
4. Додавання результатів проходження тестів для оцінки професійних навичок.
5. Додавання результатів проходження тестів для оцінки особистісних якостей робітника.

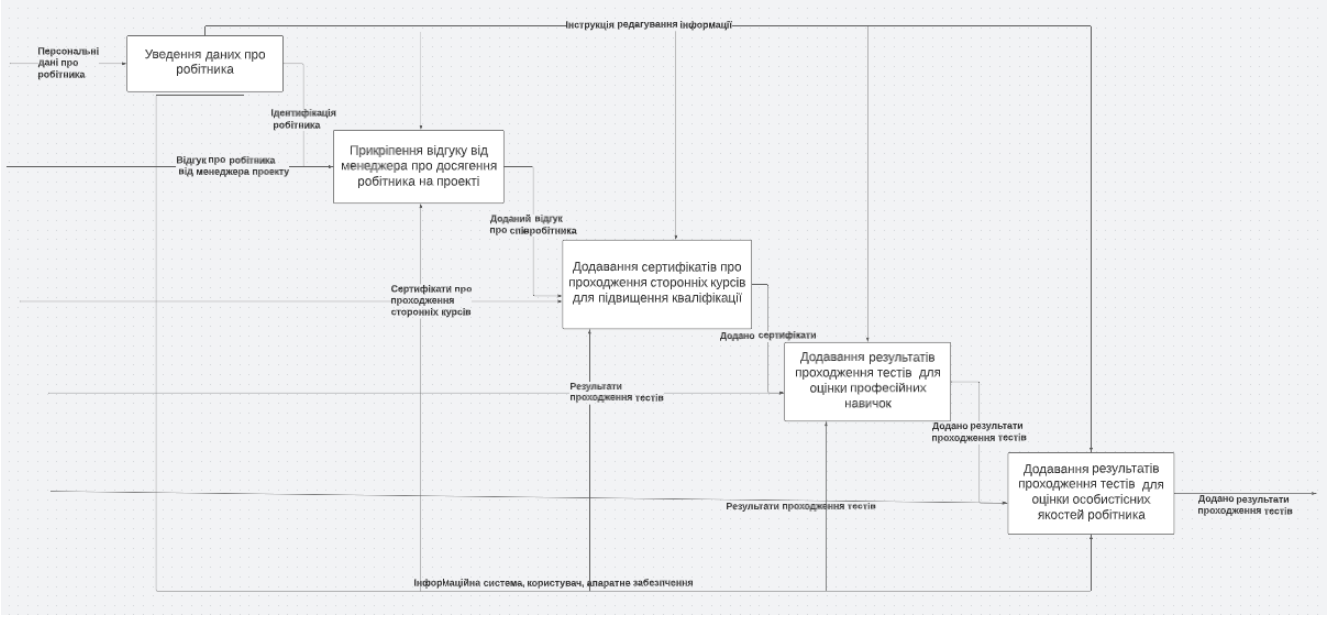


Рисунок 3.6 – Діаграма декомпозиції перегляду заробітної плати робітника

Таблиця 3.1 – Вхідні та вихідні дані для діаграми перегляду заробітної плати робітника

Стрілка/ Підпроцес	**Вхідні дані**	**Управління**	**Механізми**	**Вихідні дані**
Уведення даних про робітника	Персональні дані про робітника	Інструкція редагування інформації	Інформаційна система, користувач та апаратне забезпечення	Ідентифікація робітника
Прикріплення відгуку від менеджера про досягнення робітника на проекті	Відгук про робітника від менеджера проекту			Доданий відгук про співробітника
Додавання сертифікатів про проходження сторонніх курсів для підвищення кваліфікації	Сертифікати про проходження сторонніх курсів для підвищення кваліфікації			Додано сертифікати
Додавання результатів проходження тестів для оцінки професійних навичок;	Результати проходження тестів для оцінки професійних навичок			Додатно результати проходження тестів
Додавання результатів проходження тестів для оцінки особистісних якостей робітника	Результати проходження тестів для оцінки особистісних якостей робітника			

Декомпозиція діаграми перегляду зміни позиції робітника представлена такими підпроцесами:

1. Уведення даних про робітника.
2. Прикріплення відгуку від менеджера про досягнення робітника на проекті.
3. Введення можливої позиції робітника.
4. Додавання сертифікатів про проходження сторонніх курсів для підвищення кваліфікації.
5. Додавання результатів проходження тестів для оцінки професійних навичок.
6. Додавання результатів проходження тестів для оцінки особистісних якостей робітника.

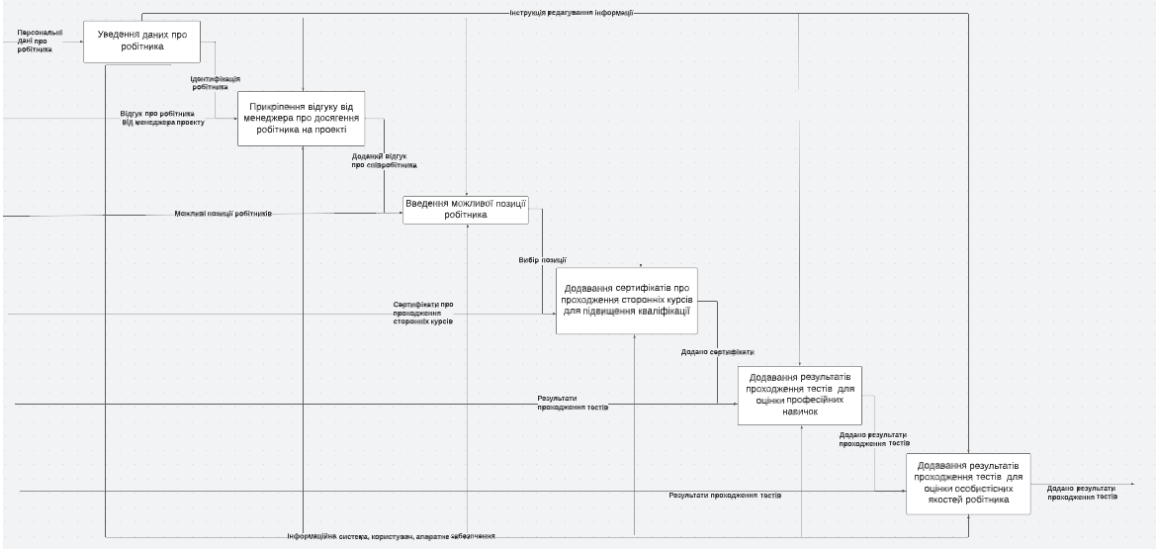


Рисунок 3.7 – Діаграма декомпозиції перегляду позиції робітника

Таблиця 3.2 – Вхідні та вихідні дані для діаграми зміни позиції робітника

Стрілка/ Підпроцес	**Вхідні дані**	**Управління**	**Механізми**	**Вихідні дані**
Уведення даних про робітника	Персональні дані про робітника	Інструкція редагування інформації	Інформаційна система, користувач та апаратне забезпечення	Ідентифікація робітника
Прикріплення відгуку від менеджера про досягнення робітника на проекті	Відгук про робітника від менеджера проекту			Доданий відгук про співробітника
Введення можливої позиції робітника	Можливі позиції співробітників			Вибір позиції
Додавання сертифікатів про проходження сторонніх курсів для підвищення кваліфікації	Сертифікати про проходження сторонніх курсів для підвищення кваліфікації			Додано сертифікати
Додавання результатів проходження тестів для оцінки професійних навичок;	Результати проходження тестів для оцінки професійних навичок			Додатно результати проходження тестів
Додавання результатів проходження тестів для оцінки особистісних якостей робітника	Результати проходження тестів для оцінки особистісних якостей робітника			

Декомпозиція діаграми перегляду зміни ролі робітника представлена такими підпроцесами:

1. Уведення даних про робітника.
2. Прикріплення відгуку від менеджера про досягнення робітника на проекті.
3. Введення можливої ролі співробітника.
4. Додавання сертифікатів про проходження сторонніх курсів для підвищення кваліфікації.
5. Додавання результатів проходження тестів для оцінки професійних навичок.
6. Додавання результатів проходження тестів для оцінки особистісних якостей робітника.

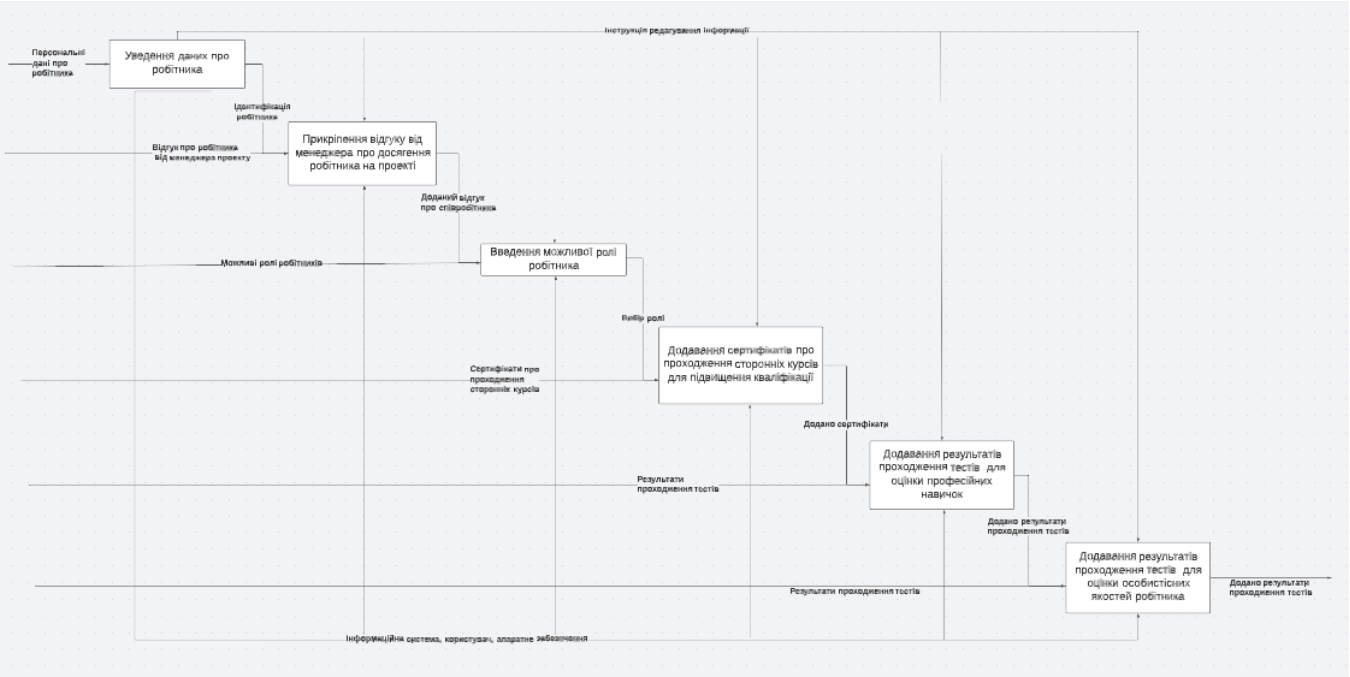


Рисунок 3.8 – Діаграма декомпозиції перегляду ролі робітника

Таблиця 3.3 – Вхідні та вихідні дані для діаграми зміни ролі робітника

Стрілка/ Підпроцес	**Вхідні дані**	**Управління**	**Механізми**	**Вихідні дані**
Уведення даних про робітника	Персональні дані про робітника	Інструкція редагування інформації	Інформаційна система, користувач та апаратне забезпечення	Ідентифікація робітника
Прикріплення відгуку від менеджера про досягнення робітника на проекті	Відгук про робітника від менеджера проекту			Доданий відгук про співробітника
Введення можливої ролі робітника	Можливі ролі співробітників			Вибір ролі
Додавання сертифікатів про проходження сторонніх курсів для підвищення кваліфікації	Сертифікати про проходження сторонніх курсів для підвищення кваліфікації			Додано сертифікати
Додавання результатів проходження тестів для оцінки професійних навичок;	Результати проходження тестів для оцінки професійних навичок			Додатно результати проходження тестів
Додавання результатів проходження тестів для оцінки особистісних якостей робітника	Результати проходження тестів для оцінки особистісних якостей робітника			

Декомпозиція діаграми підбору робітників на проект представлена такими підпроцесами:

1. Уведення даних про проект.
2. Пошук співробітників за критеріями та ключовими словами.
3. Призначення списку робітників на проект.

Рисунок 3.9 – Діаграма декомпозиції підбору співробітників на проект

Таблиця 3.4 – Вхідні та вихідні дані для діаграми підбору співробітників на проект

Стрілка/ Підпроцес	**Вхідні дані**	**Управління**	**Механізми**	**Вихідні дані**
Уведення даних про проект	Дані про проект	Інструкція редагування інформації	Інформаційна система, користувач та апаратне забезпечення	Ідентифікація проекту
Пошук співробітників за критеріями та ключовими словами	Список всіх співробітників компанії			Список робітників, що підходять за критеріями для проекту
Призначення списку робітників на проект	Список робітників, що підходять за критеріями для проекту			Призначено робітників на проект

Декомпозиція діаграми перегляду зміни проекту робітника представлена такими підпроцесами:

1. Уведення даних про робітника.
2. Прикріплення відгуку від менеджера про досягнення робітника на проекті.
3. Введення нового проекту для співробітника.

4. Додавання сертифікатів про проходження сторонніх курсів для підвищення кваліфікації.

5. Додавання результатів проходження тестів для оцінки професійних навичок.

6. Додавання результатів проходження тестів для оцінки особистісних якостей робітника.

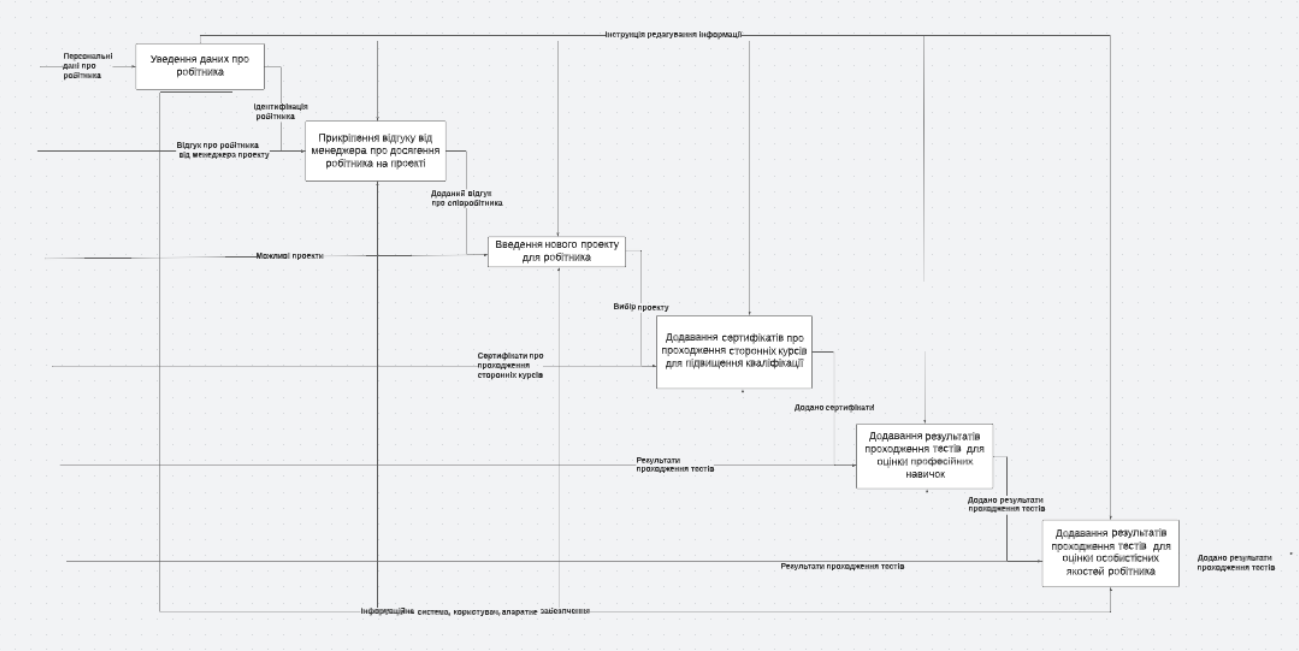


Рисунок 3.10 – Діаграма декомпозиції зміни проекту для робітника

Таблиця 3.5 – Вхідні та вихідні дані для діаграми зміни ролі робітника

Стрілка/ Підпроцес	**Вхідні дані**	**Управління**	**Механізми**	**Вихідні дані**
Уведення даних про робітника	Персональні дані про робітника	Інструкція редагування інформації	Інформаційна система, користувач та апаратне забезпечення	Ідентифікація робітника
Прикріплення відгуку від менеджера про досягнення робітника на проекті	Відгук про робітника від менеджера проекту			Доданий відгук про співробітника
Введення нового проекту для робітника	Можливі проекти компанії			Вибір проекту

Продовження таблиці 3.5

Стрілка/ Підпроцес	**Вхідні дані**	**Управління**	**Механізми**	**Вихідні дані**
Додавання сертифікатів про проходження сторонніх курсів для підвищення кваліфікації	Сертифікати про проходження сторонніх курсів для підвищення кваліфікації	Інструкція редагування інформації	Інформаційна система, користувач та апаратне забезпечення	Додано сертифікати
Додавання результатів проходження тестів для оцінки професійних навичок;	Результати проходження тестів для оцінки професійних навичок			Додатно результати проходження тестів
Додавання результатів проходження тестів для оцінки особистісних якостей робітника	Результати проходження тестів для оцінки особистісних якостей робітника			

3.2 Діаграма Use Case

Після виконання моделювання та опису бізнес-процесів майбутньої інформаційної системи необхідно було створити діаграму варіантів використання. За допомогою Use Case схематично зображують системні та програмні вимоги до розроблюваної ІС. Ключовими її складовими є актори, зв’язки, примітки та механізми розширення [12].

Для інформаційної системи оцінки професійних досягнень співробітників IT-компанії було виділено такі варіанти використання:

- ініціація зміни проекту;
- організація, виконання тестів для оцінки особистісних якостей;
- організація, виконання тестів для оцінки професійних навичок;
- створення, редагування, видалення інформації про робітників;

- завантаження сертифікатів про проходження курсів;
- написання, редагування, перегляд, видалення відгуку про працівника;
- призначення нової позиції;
- призначення перегляду заробітної плати;
- призначення на новий проект.

Акторами діаграми Use Case є менеджер проекту, директор, HR та співробітник.

Діаграма використання інформаційної системи оцінки професійних досягнень співробітників IT-компанії зображена на рисунку 3.1

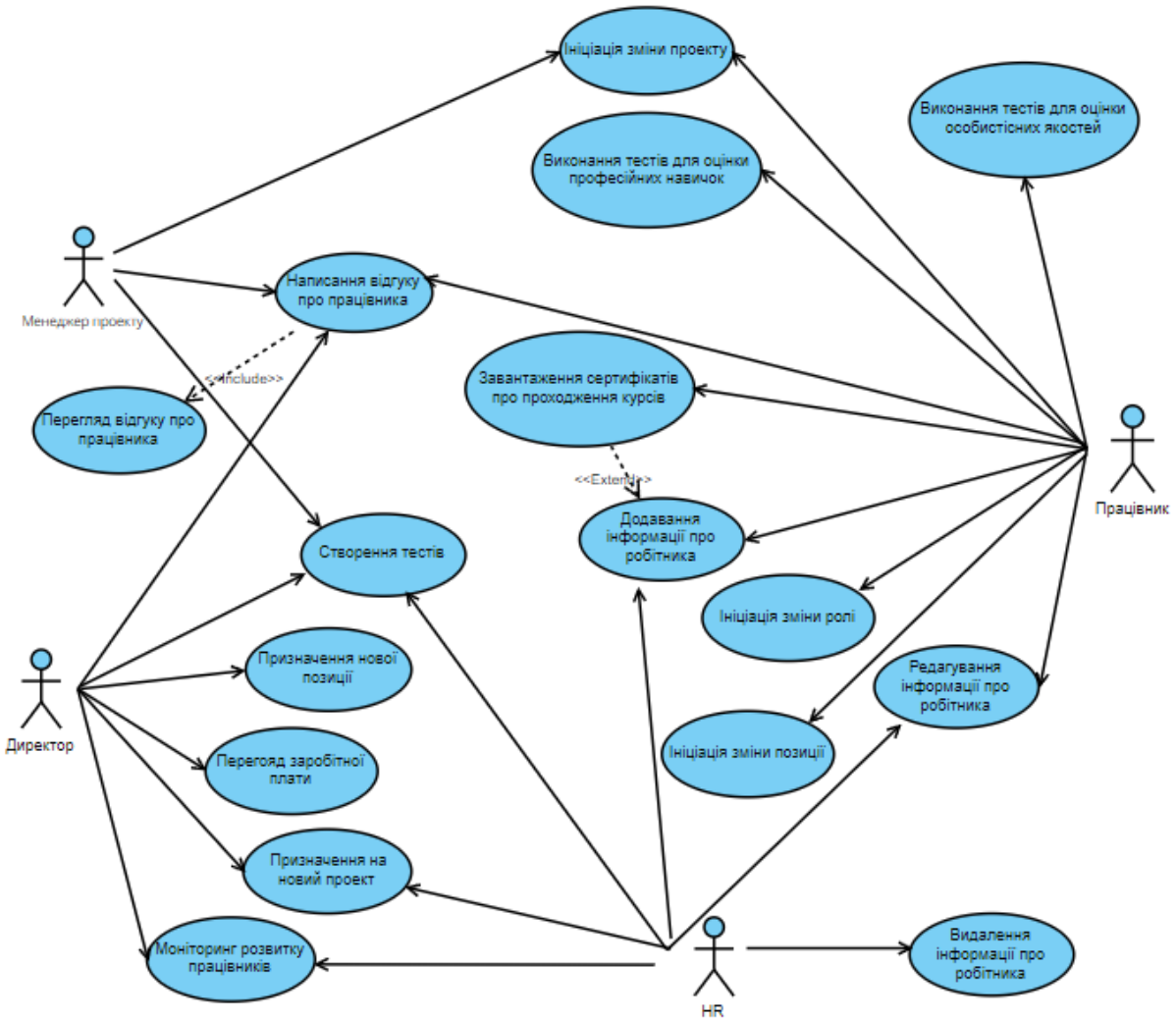


Рисунок 3.1 – Діаграма Use Case

4 РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ОЦІНКИ ПРОФЕСІЙНИХ ДОСЯГНЕНЬ СПІВРОБІТНИКІВ ІТ-КОМПАНІЇ

4.1 Архітектура інформаційної системи

Процес розробки інформаційної системи починається з визначення архітектури програмного додатку за допомогою діаграми HLD (High Level Design) [13]. Діаграма дизайну високого рівня призначена для опису компонент продукту, які можна коригувати в залежності від зміни вимог та ризиків (рис. 4.1).

Архітектура інформаційної системи оцінки професійних досягнень співробітників ІТ-компанії містить наступне:

– модель, яка взаємодіє з сервіс-контейнером та системою об’єктно-реляційного відображення;

– контролер, який пов’язаний з моделлю;

– вид для візуалізації зображень;

– базу даних, яка забезечує даними модель.

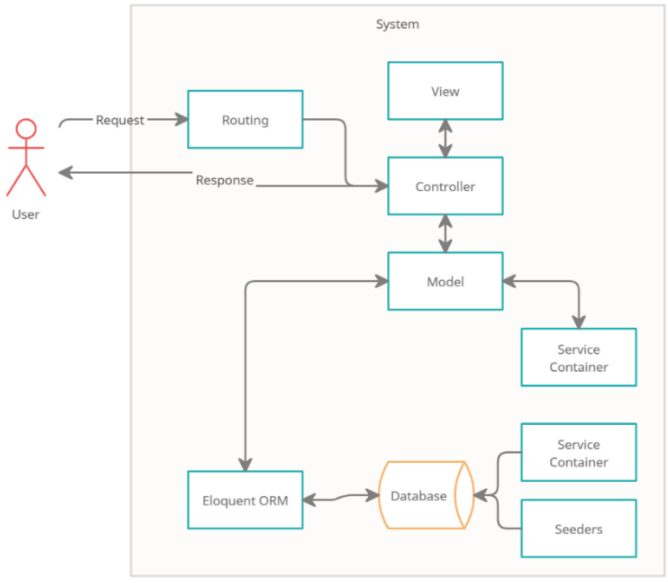


Рисунок 4.1 – Діаграма високого рівня

4.2 Проектування бази даних інформаційної системи

Результатом проектування бази даних (БД) інформаційної системи є сукупність взаємопов’язаних таблиць та відношень. Вона називається структурою [14]. Першим кроком у розробці бази даних було виділення сутностей системи, де на кожну з них повинна існувати таблиця.

Під час створення структури БД було виділено наступні сутності:

- якості (qualities);
- позиції (positions);
- ролі (roles);
- проекти (projects);
- відділи (departments);
- працівники (employees);
- відповіді (answers);
- варіанти (variants);
- питання (questions);
- ключові слова (keywords);
- тести (tests);
- категорії тестів (test categories):
- сертифікати (certificates);
- відгуки (feedbacks);
- запити на зміну проекту (request project);
- запити на перегляд заробітної плати (request salaries);
- запити на зміну позиції (request positions);
- запити на зміну ролі (request roles);
- вимоги до позиції (position requirements).

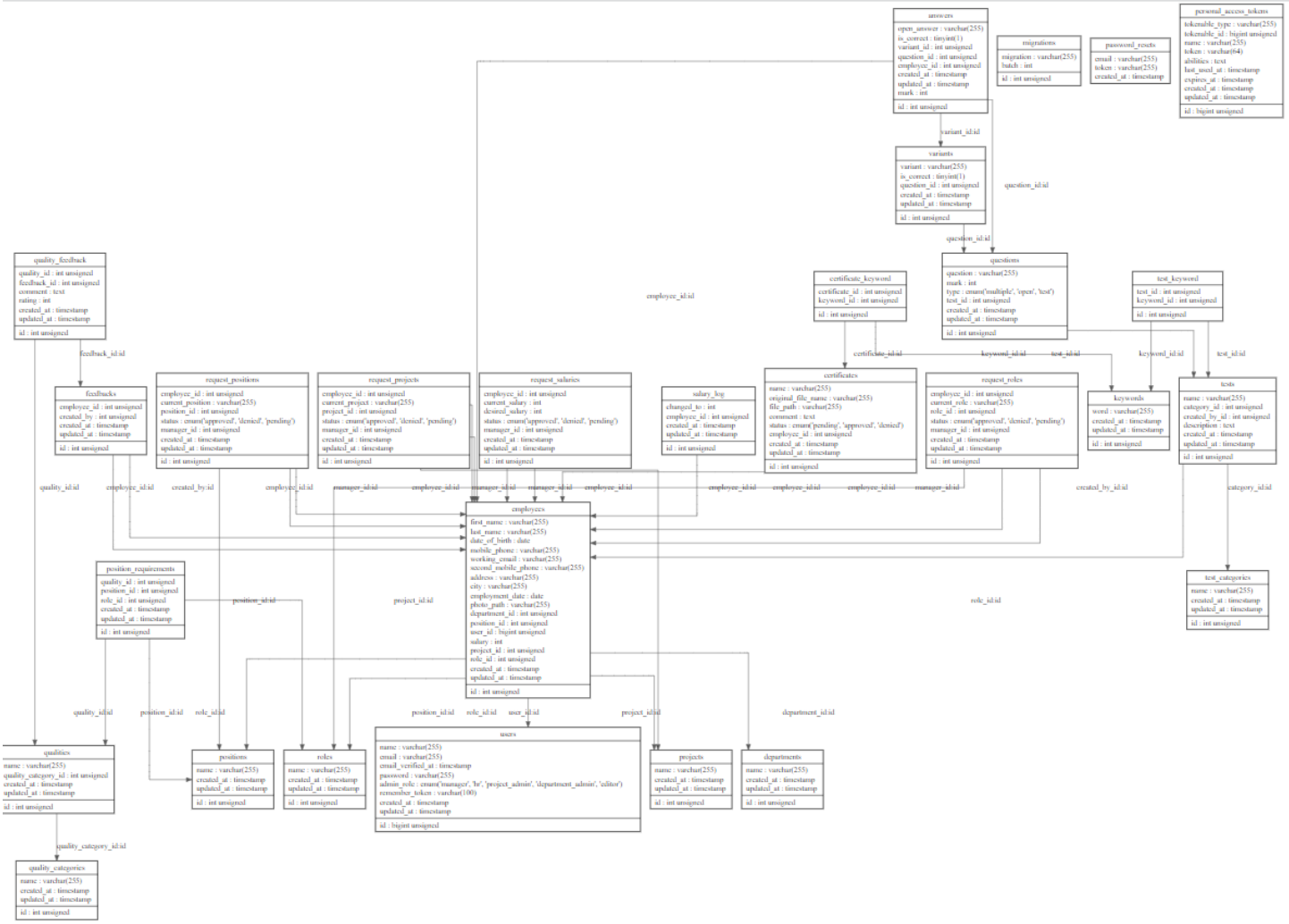


Рисунок 4.2 – Логічна модель бази даних

4.3 Програмна реалізація інформаційної системи

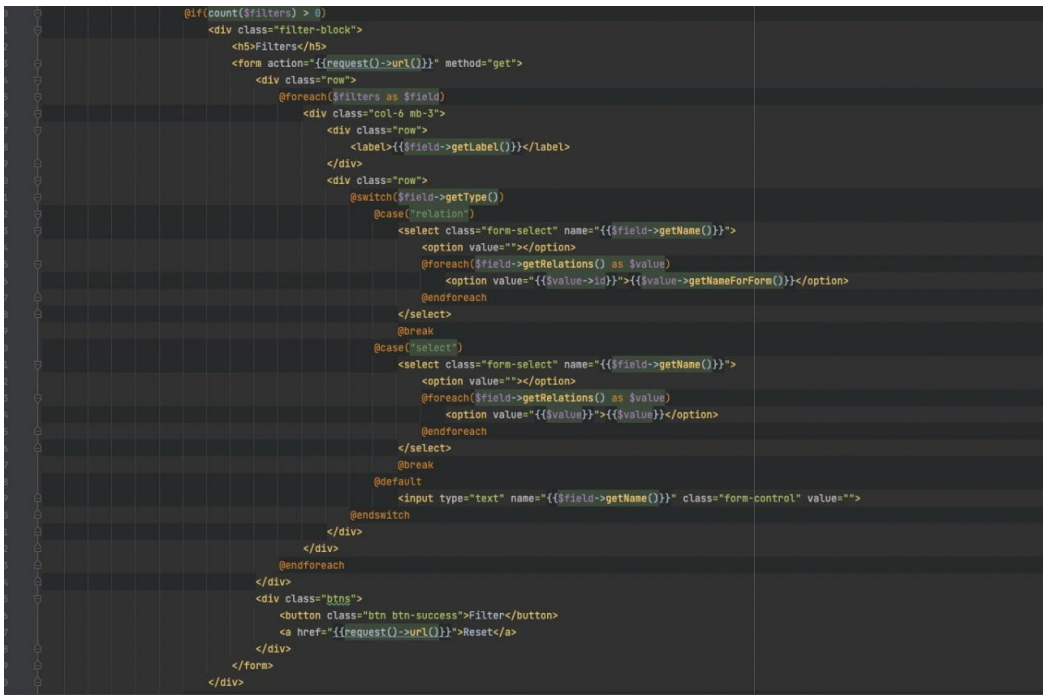
Підготовчим етапом до розробки програмного продукту є завантаження та інсталяція локального web-серверу XAMPP. Спочатку треба створити папку для інформаційної системи з адресою C:\XAMPP\htdocs\Rateit та налаштувати віртуальний хост у файлах httpd-vhosts.conf та hosts. Для написання програмного коду ІС було обрано IDE PHPStorm, щоб забезпечити контроль розробки та версій – Git [15]. Після

встановлення усіх необхідних додатків було розроблено базу даних відповідно до логічної моделі описаної у попередньому розділі.

За допомогою адміністративної панелі користувач має доступ до отримання, створення, редагування будь-яких сутності: працівників, тести, запити і т.д. Шаблони верстки адміністративної сторінки для створення та фільтрації сутностей представлено на рисунках 4.3-4.4.

```
@include('parts.errors')
<form action="{{route($store_route)}}" method="post" enctype="multipart/form-data">
    @csrf
    @foreach($fields as $field)
        <div class="form-group">
            <div class="row">
                <div class="col-12 col-lg-6">
                    <label for="last-name">{{ucfirst($field->getName())}}</label>
                    @switch($field->getType())
                        @case(\Illuminate\Support\Carbon::class)
                            <input type="date" name="{{$field->getName()}}" class="form-control" value="" placeholder="Input {{$field->getName()}}">
                            @break
                        @case("relation")
                            <select class="form-select" name="{{$field->getName()}}">
                                <option value=""></option>
                                @foreach($field->getRelations() as $value)
                                    <option value="{{$value->id}}">{{$value->getNameForForm()}}</option>
                                @endforeach
                            </select>
                            @break
                        @case("select")
                            <select class="form-select" name="{{$field->getName()}}">
                                <option value=""></option>
                                @foreach($field->getRelations() as $value)
                                    <option value="{{$value}}">{{$value}}</option>
                                @endforeach
                            </select>
                            @break
                        @case("file")
                            <input type="file" name="{{$field->getName()}}" class="form-control" value="" placeholder="Uplpad {{$field->getName()}}">
                            @break
                        @default
                            <input type="text" name="{{$field->getName()}}" class="form-control" value="" placeholder="Input {{$field->getName()}}">
                    @endswitch
                </div>
            </div>
        </div>
    @endforeach
    <button class="btn btn-success mt-3">Save</button>
</form>
```

Рисунок 4.3 – Шаблон створення сутностей



```
@if(count($filters) > 0)
    <div class="filter-block">
        <h5>Filters</h5>
        <form action="{{request()->url()}}" method="get">
            <div class="row">
                @foreach($filters as $field)
                    <div class="col-6 mb-3">
                        <div class="row">
                            <label>{{$field->getLabel()}}</label>
                        </div>
                        <div class="row">
                            @switch($field->getType())
                                @case("relation")
                                    <select class="form-select" name="{{$field->getName()}}">
                                        <option value=""></option>
                                        @foreach($field->getRelations() as $value)
                                            <option value="{{$value->id}}">{{$value->getNameForForm()}}</option>
                                        @endforeach
                                    </select>
                                    @break
                                @case("select")
                                    <select class="form-select" name="{{$field->getName()}}">
                                        <option value=""></option>
                                        @foreach($field->getRelations() as $value)
                                            <option value="{{$value}}">{{$value}}</option>
                                        @endforeach
                                    </select>
                                    @break
                                @default
                                    <input type="text" name="{{$field->getName()}}" class="form-control" value="">
                            @endswitch
                        </div>
                    </div>
                @endforeach
            </div>
            <div class="btns">
                <button class="btn btn-success">Filter</button>
                <a href="{{request()->url()}}">Reset</a>
            </div>
        </form>
    </div>
```

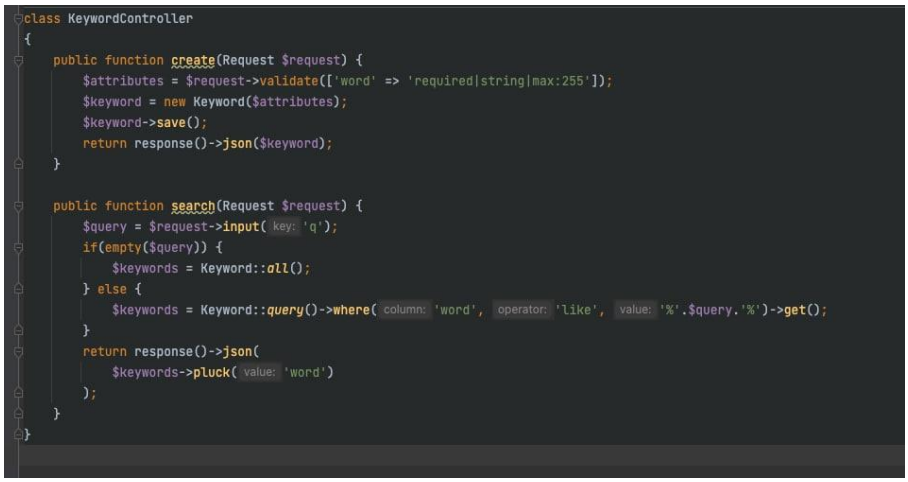
Рисунок 4.4 – Шаблон фільтрації даних

Інформаційна система має вимоги до перегляду позицій для співробітників. Наприклад, достатній рівень володіння англійською мовою та/або певними мовами програмування чи інші навички. Перевірка відповідності робітника до вимог була реалізована за домогою vue.js компоненти (рис. 4.5).

```
export default {
    name: "Raise",
    props: ['positions'],
    data() {
        return {
            selectedPosition: [],
            requirements: []
        }
    },
    methods: {
        loadRequirements() {
            axios.get( url: `/webapi/me/positions/${this.selectedPosition}/requirements` ).then((res) => {
                this.requirements = res.data
            })
        }
    }
}
</script>
```

Рисунок 4.5 – Компонента завантаження вимог
щодо перегляду позиції робітника

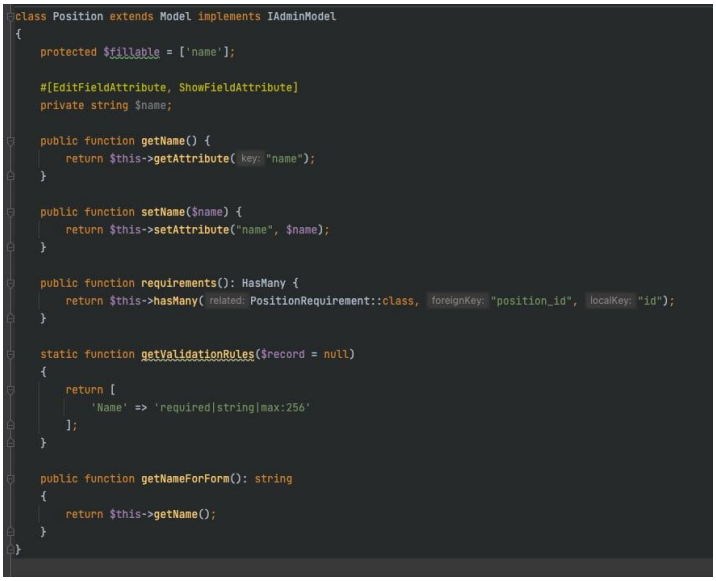
Оскільки безліч функціональних рішень для сутностей потребують пошук по ключовим словам – було розроблено контролер KeyWordController для створення та пошуку наявних ключових слів у системі (рис.4.6).



```php
class KeywordController
{
    public function create(Request $request) {
        $attributes = $request->validate(['word' => 'required|string|max:255']);
        $keyword = new Keyword($attributes);
        $keyword->save();
        return response()->json($keyword);
    }

    public function search(Request $request) {
        $query = $request->input( key: 'q');
        if(empty($query)) {
            $keywords = Keyword::all();
        } else {
            $keywords = Keyword::query()->where( column: 'word', operator: 'like', value: '%'.$query.'%')->get();
        }
        return response()->json(
            $keywords->pluck( value: 'word')
        );
    }
}
```

Рисунок 4.6 – Контролер управління ключовими словами

Для управління інформацією з таблиць бази даних було створено моделі сутностостей. Наприклад, моделі сутностей «Позиція» та «Проект» зображені на рисунках 4.7-4.8.



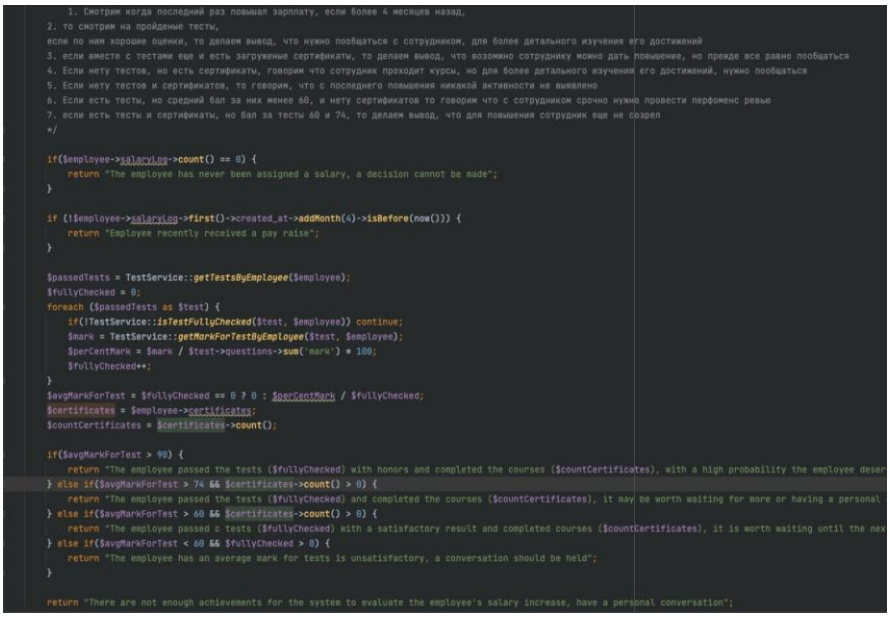
```php
class Position extends Model implements IAdminModel
{
    protected $fillable = ['name'];

    #[EditFieldAttribute, ShowFieldAttribute]
    private string $name;

    public function getName() {
        return $this->getAttribute( key: "name");
    }

    public function setName($name) {
        return $this->setAttribute("name", $name);
    }

    public function requirements(): HasMany {
        return $this->hasMany( related: PositionRequirement::class, foreignKey: "position_id", localKey: "id");
    }

    static function getValidationRules($record = null)
    {
        return [
            'Name' => 'required|string|max:256'
        ];
    }

    public function getNameForForm(): string
    {
        return $this->getName();
    }
}
```

Рисунок 4.7 – Модель сутності «Позиція»

```php
class Project extends Model implements IAdminModel
{
    protected $fillable = ['name'];


    #[EditFieldAttribute, ShowFieldAttribute]
    private string $name;

    public function getName() {
        return $this->getAttribute( key: "name");
    }

    public function setName($name) {
        return $this->setAttribute("name", $name);
    }

    static function getValidationRules($record = null)
    {
        return [
            'Name' => 'required|string|max:256'
        ];
    }

    public function getNameForForm(): string
    {
        return $this->getName();
    }

}
```

Рисунок 4.8 – Модель сутності «Проект»

Для підтримки прийняття рішення щодо перегляду заробітної плати було написано метод, що враховує, коли останнього разу було переглянуто заробітну плату, якість проходження тестів та сертифікатів (рис. 4.9). Програмний код основних модулів інформаційної системи представлено у додатку Б.



```php
    1. Смотрим когда последний раз повышал зарплату, если более 4 месяцев назад,
2. то смотрим на пройденые тесты,
если по ним хорошие оценки, то делаем вывод, что нужно пообщаться с сотрудником, для более детального изучения его достижений
3. если вместе с тестами еще и есть загруженые сертификаты, то делаем вывод, что возможно сотруднику можно дать повышение, но прежде все равно пообщаться
4. Если нету тестов, но есть сертификаты, говорим что сотрудник проходит курсы, но для более детального изучения его достижений, нужно пообщаться
5. Если нету тестов и сертификатов, то говорим, что с последнего повышения никакой активности не выявлено
6. Если есть тесты, но средний бал за них менее 60, и нету сертификатов то говорим что с сотрудником срочно нужно провести перфоменс ревью
7. если есть тесты и сертификаты, но бал за тесты 60 и 74, то делаем вывод, что для повышения сотрудник еще не созрел
*/

if($employee->salarylog->count() == 0) {
    return "The employee has never been assigned a salary, a decision cannot be made";
}

if (!$employee->salarylog->first()->created_at->addMonth(4)->isBefore(now())) {
    return "Employee recently received a pay raise";
}

$passedTests = TestService::getTestsByEmployee($employee);
$fullyChecked = 0;
foreach ($passedTests as $test) {
    if(!TestService::isTestFullyChecked($test, $employee)) continue;
    $mark = TestService::getMarkForTestByEmployee($test, $employee);
    $perCentMark = $mark / $test->questions->sum('mark') * 100;
    $fullyChecked++;
}
$avgMarkForTest = $fullyChecked == 0 ? 0 : $perCentMark / $fullyChecked;
$certificates = $employee->certificates;
$countCertificates = $certificates->count();

if($avgMarkForTest > 90) {
    return "The employee passed the tests ($fullyChecked) with honors and completed the courses ($countCertificates), with a high probability the employee deser
} else if($avgMarkForTest > 74 && $certificates->count() > 0) {
    return "The employee passed the tests ($fullyChecked) and completed the courses ($countCertificates), it may be worth waiting for more or having a personal
} else if($avgMarkForTest > 60 && $certificates->count() > 0) {
    return "The employee passed c tests ($fullyChecked) with a satisfactory result and completed courses ($countCertificates), it is worth waiting until the nex
} else if($avgMarkForTest < 60 && $fullyChecked > 0) {
    return "The employee has an average mark for tests is unsatisfactory, a conversation should be held";
}

return "There are not enough achievements for the system to evaluate the employee's salary increase, have a personal conversation";
```

Рисунок 4.9 – Метод підтримки рішення щодо перегляду заробітної плати працівника

4.4 Демонстрація роботи інформаційної системи

Інформаційна система оцінки професійних досягнень співробітників IT-компанії містить 5 видів користувачів у системі, які відрізняються рівнями доступу до даних (рис. 4.10):

– MANAGER – це адміністратор, який має необмежені права щодо додавання, редагування та видалення інформації;

– HR – це менеджер персоналу компанії, який має доступ до всіх функціональних можливостей, окрім оновлення заробітної плати робітника;

– PROJECT_ADMIN – це керівник проекту, який має доступ до управління даними співробітників на його проекті;

– DEPARTMENT_ADMIN – менеджер департаменту компанії, який може редагувати інформацію, що стосується його підрозділу;

– користувач без зазначення імені – це звичайний робітник компанії, який має право редагувати тільки власні дані, запити, сертифікати та тести.

Кожен із типів користувачів має 2 режими користування інформаційною системою: профіль та адміністратор. Варто зауважити, що на адміністраторській панелі кожен юзер має тільки ті функції, які дозволені відповідно до його прав. Нижче представлено деякі функціональні можливості, які доступні адміністратору з рівнем доступу MANAGER.

Web-сторінка створення та видалення користувача, що доступна тільки для MANAGER, зображена на рисунку 4.11.

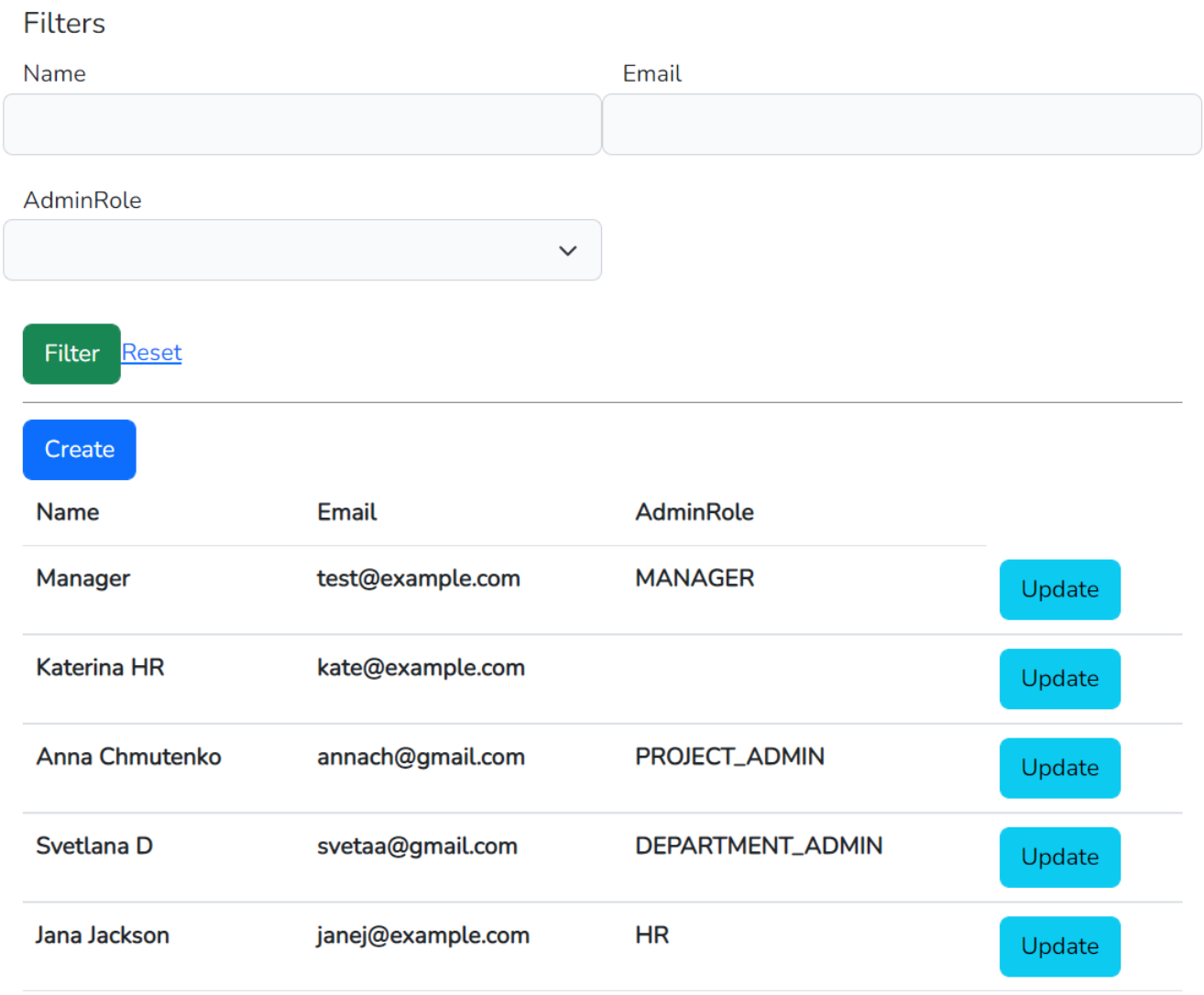


Рисунок 4.10 – Типи користувачів інформаційної системи

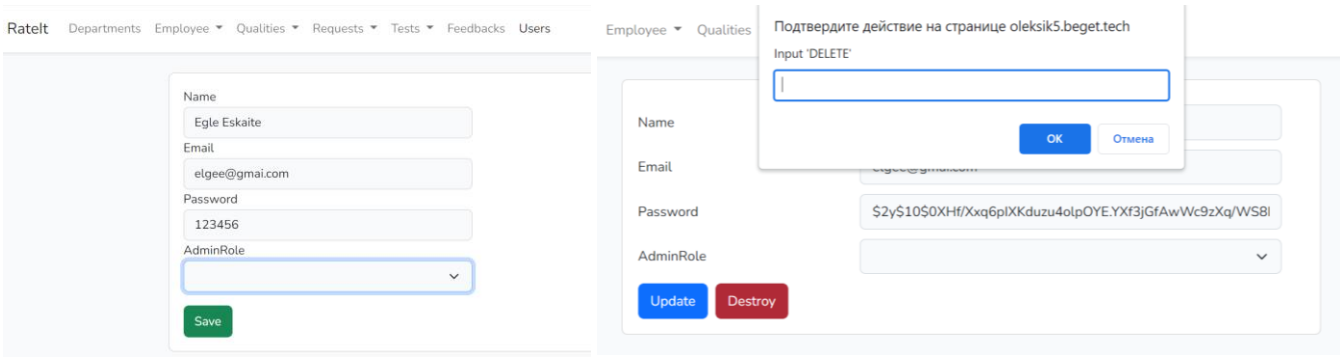


Рисунок 4.11 – Web-сторінка створення та видалення користувача

Функція написання відгуків про працівника, що доступна усім типам користувачів, окрім користувача без зазначення імені, організована у вигляді вибору робітника, підбору навичок, написання коментаря та виставлення оцінки (рис. 4.12).

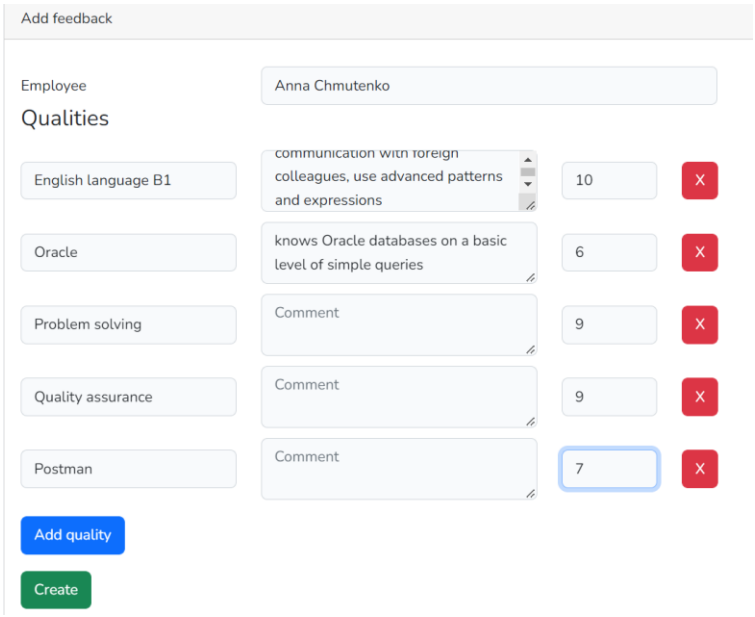


Рисунок 4.12 – Web-сторінка написання відгуку про робітника

Для розробки тесту перевірки професійних чи особистісних якостей співробітника, користувач має перейти на пункт меню «Tests» (рис. 4.13). Спершу необхідно обрати категорію, якщо останньої не існує, її можна створити, перейшовши на відповідну вкладку меню (рис.4.14). Web-сторінка розробки тесту доступна всім користувачам окрім користувача без зазначеного імені, зображена на рисунку 4.15.

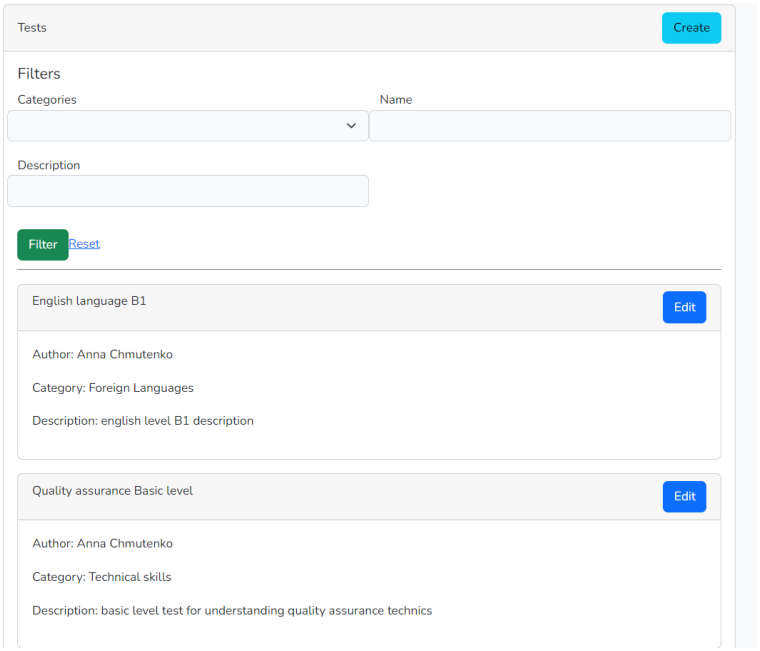


Рисунок 4.13 – Web-сторінка «Tests»

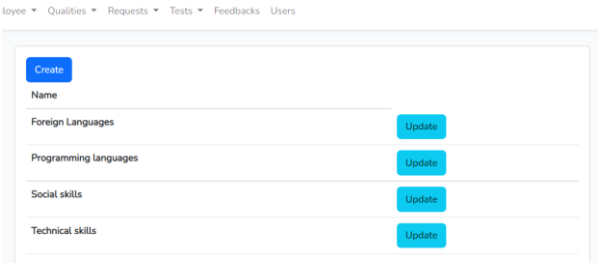


Рисунок 4.14 – Web-сторінка списку категорій для тестів

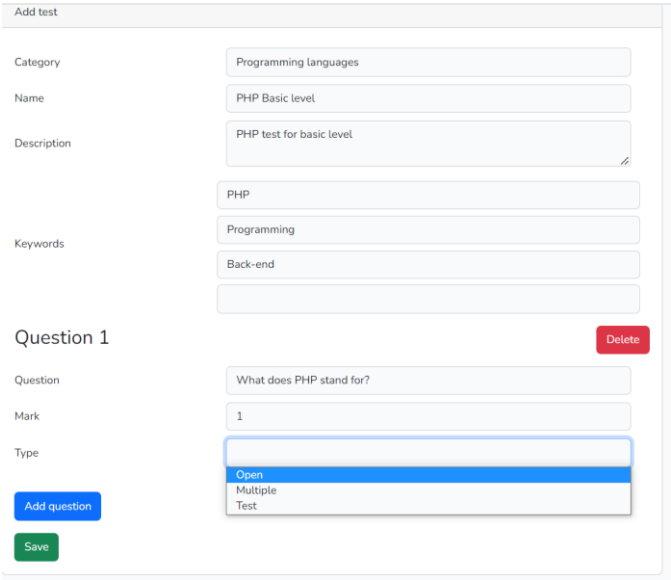


Рисунок 4.15 – Web-сторінка створення тесту

Для того, щоб проаналізувати наскільки корисним для співробітників було тестування, необхідно перейти в пункт меню «Tests». Далі треба обрати тест. Після стануть доступними 3 підпункти меню. Це список запитань, працівників, які пройшли цей тест, та статистика (рис. 4.16-4.17).

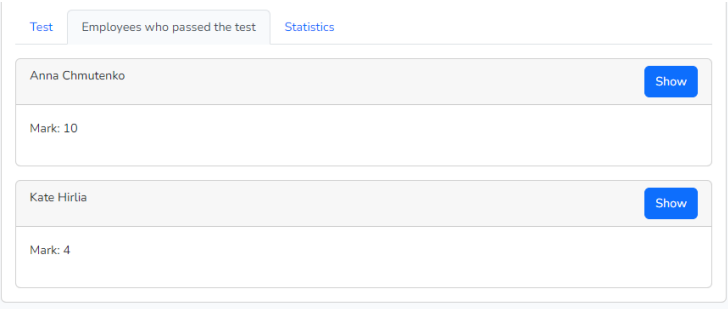


Рисунок 4.16 – Список робітників, що пройшли тест

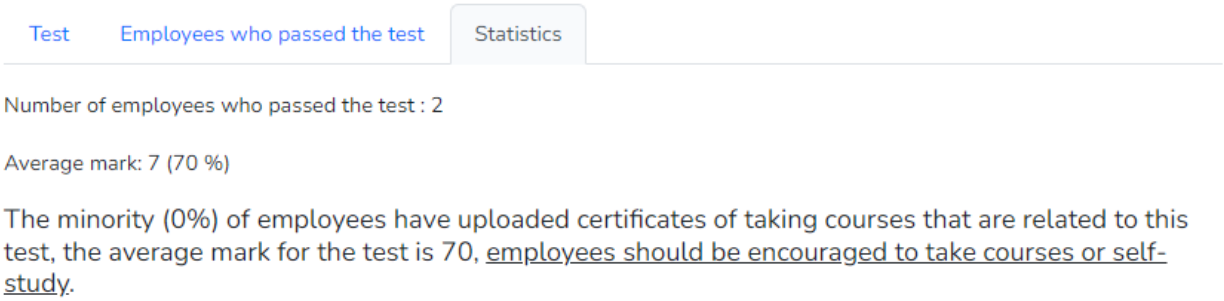


Рисунок 4.17 – Рекомендації по тесту, що спираються
на статистику проходження тесту

Пункт меню «Requests» містить запити на зміну заробітної плати (ЗП), проекту, ролі та позиції до розгляду. Ці дії виконуються відповідним адміністратором. На рисунку 4.18 зображена фільтрація запитів на перегляд заробітної плати по певному працівнику. Після того, як даний реквест на зміну ЗП було підтверджено, відповідний адміністратор може змінити її значення у профілі співробітника (рис. 4.19).

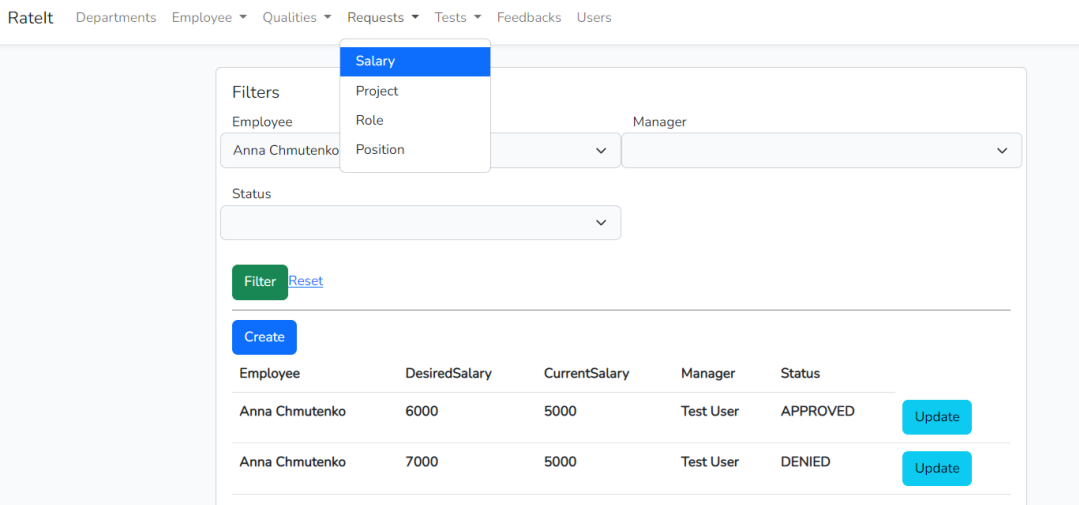


Рисунок 4.18 – Web-сторінка списку запитів
на перегляд заробітної плати

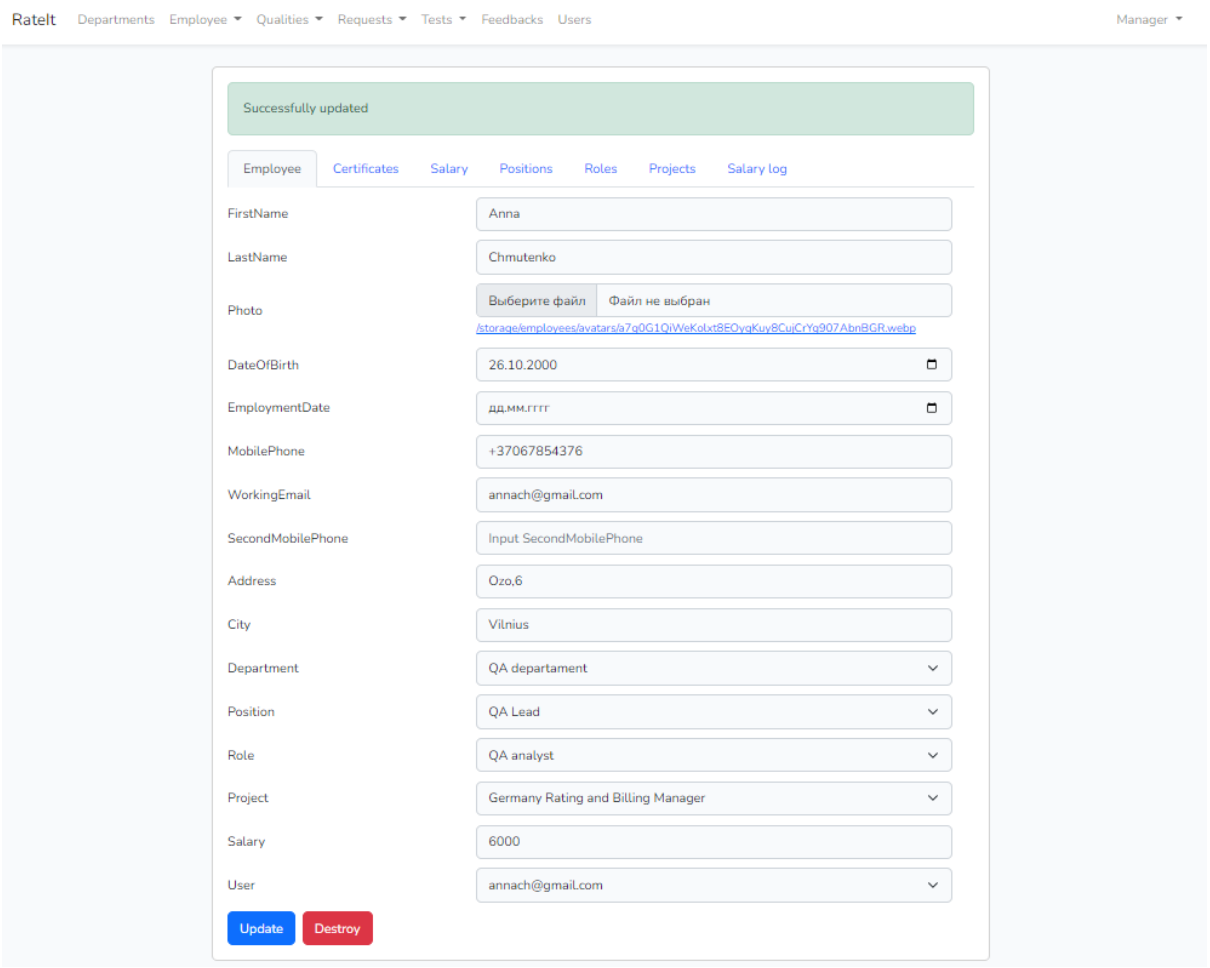


Рисунок 4.19 – Web-сторінка редагування інформації про користувача

Для підтримки процесу оцінювання навичок працівників було створено web-сторінки «Qualities» та «Categories for qualities». З їх допомогою можна визначати skills по категоріям для вимог до позицій, писати відгуки та здійснювати підбір співробітників на проект (рис. 4.20).

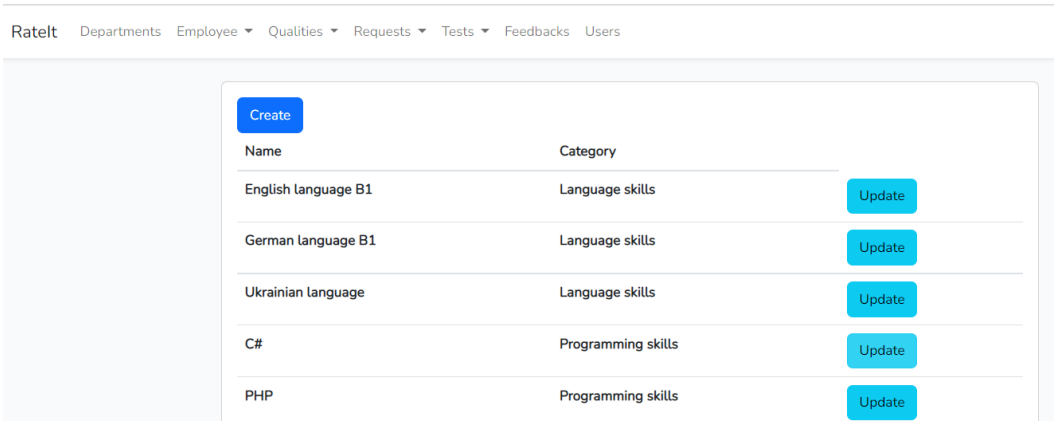


Рисунок 4.20 – Web-сторінка «Qualities»

Останніми пунктами меню адміністративної панелі є «Departments» та «Employee», де можна створювати, редагувати, фільтрувати та видаляти інформацію про наступні сутності (рис. 4.21-4.22):

– департаменти;
– працівники;
– сертифікати;
– проекти;
– ролі;
– позиції;
– вимоги до позицій.

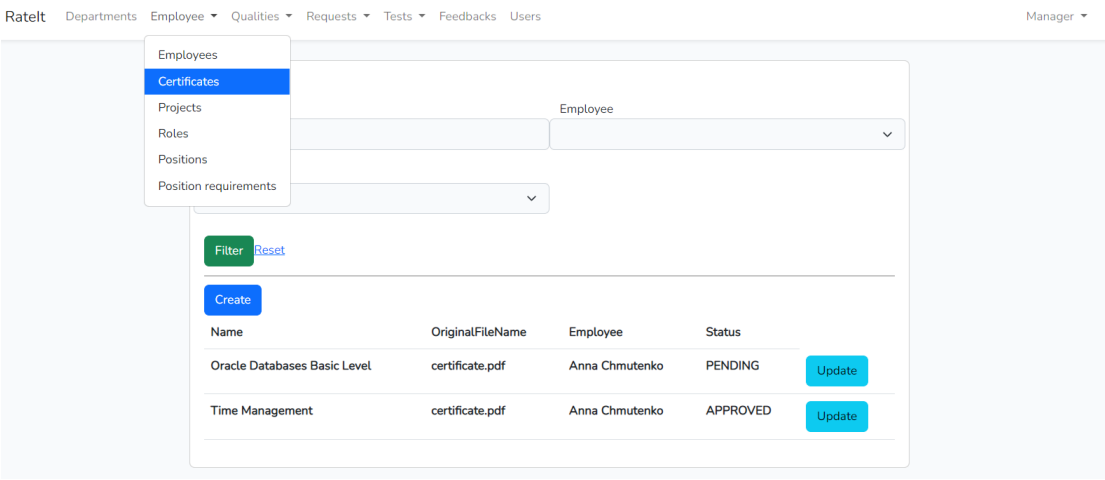


Рисунок 4.21 – Web-сторінка «Certificates»

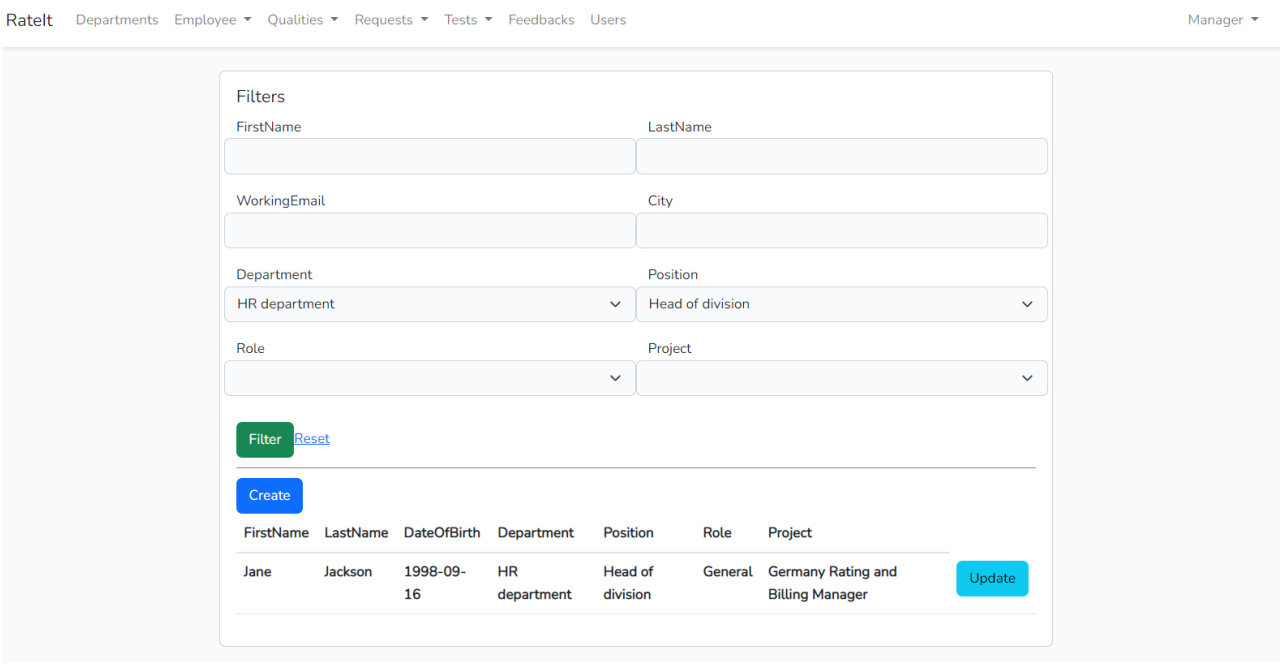


Рисунок 4.22 – Фільтрація робітників за департаментом та позицією

Для прикладу маємо робітника Anna Chmutenko, яка має рівень доступу PROJECT_ADMIN (рис. 4.23). Усі користувачі мають 2 режими роботи в інформаційній системі. Це профіль (рис. 4.24) та адмін панель (де доступні лише дозволені функції відповідно до рівня доступу).

На головній сторінці профілю даний користувач може виконувати наступні операції:

– переглядати, редагувати інформацію про себе;

– переглядати відгуки;

– створювати та переглядати запити на перегляд заробітної плати, підбір робітників на проекти та зміну проекту, позиції, ролі;

– переглядати зміну заробітної плати з часом (рис. 4.25);

– завантажувати, переглядати сертифікати;

– шукати, проходити тести та переглядати результати пройдених тестів (рис. 4.26);

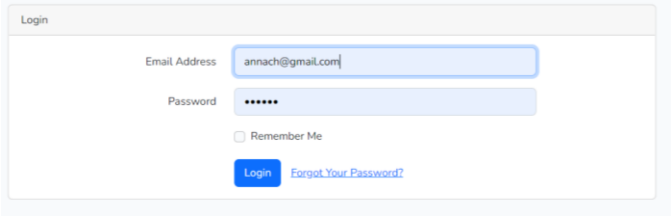


Рисунок 4.23 – Авторизація користувача

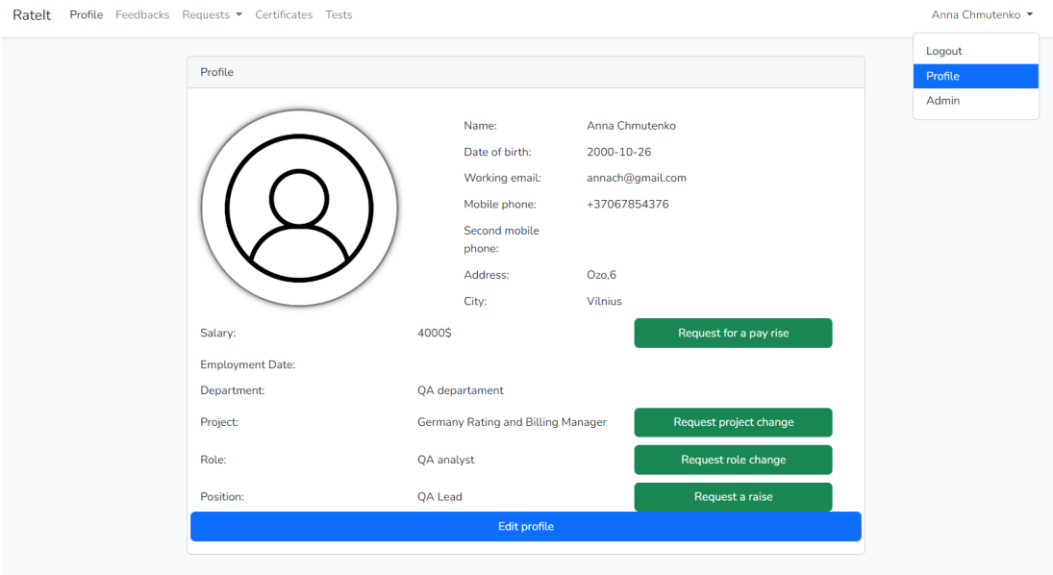


Рисунок 4.24 – Web-сторінка профілю користувача

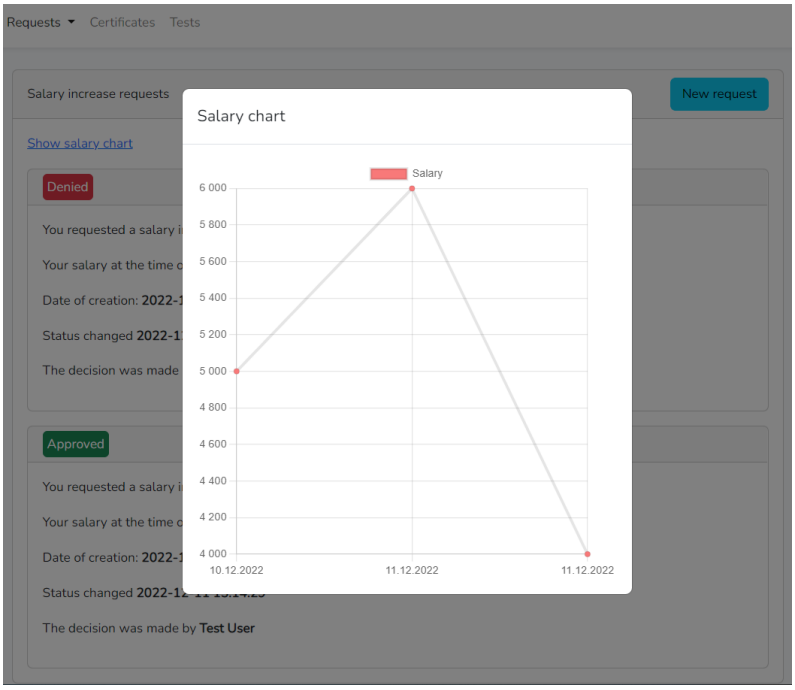


Рисунок 4.25 – Діаграма зміни заробітної плати з часом

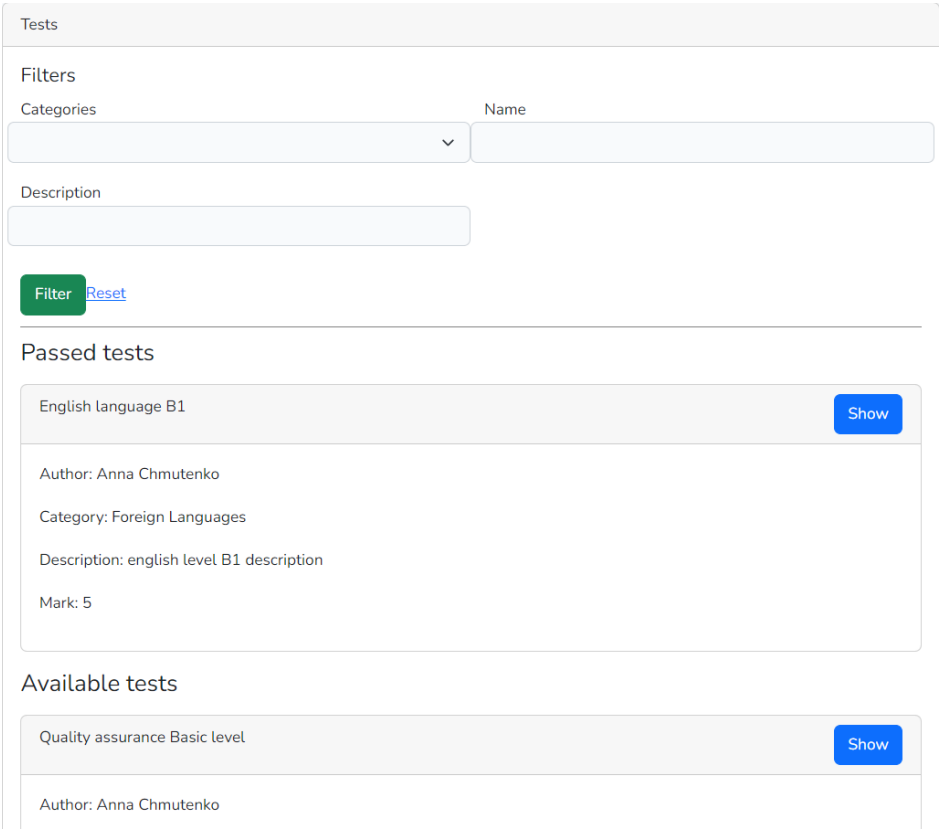


Рисунок 4.26 – Web-сторінка тестів користувача

У адміністративному режимі користувач Anna Chmutenko може виконувати наступне:

– шукати, переглядати, редагувати інформацію про робітників її проекту;
– створювати, редагувати проекти;
– підтверджувати, редагувати сертифікати співробітників;
– створювати, редагувати та перевіряти тести (рис. 4.27);
– шукати, переглядати, редагувати відгуки про співробітників проекту.

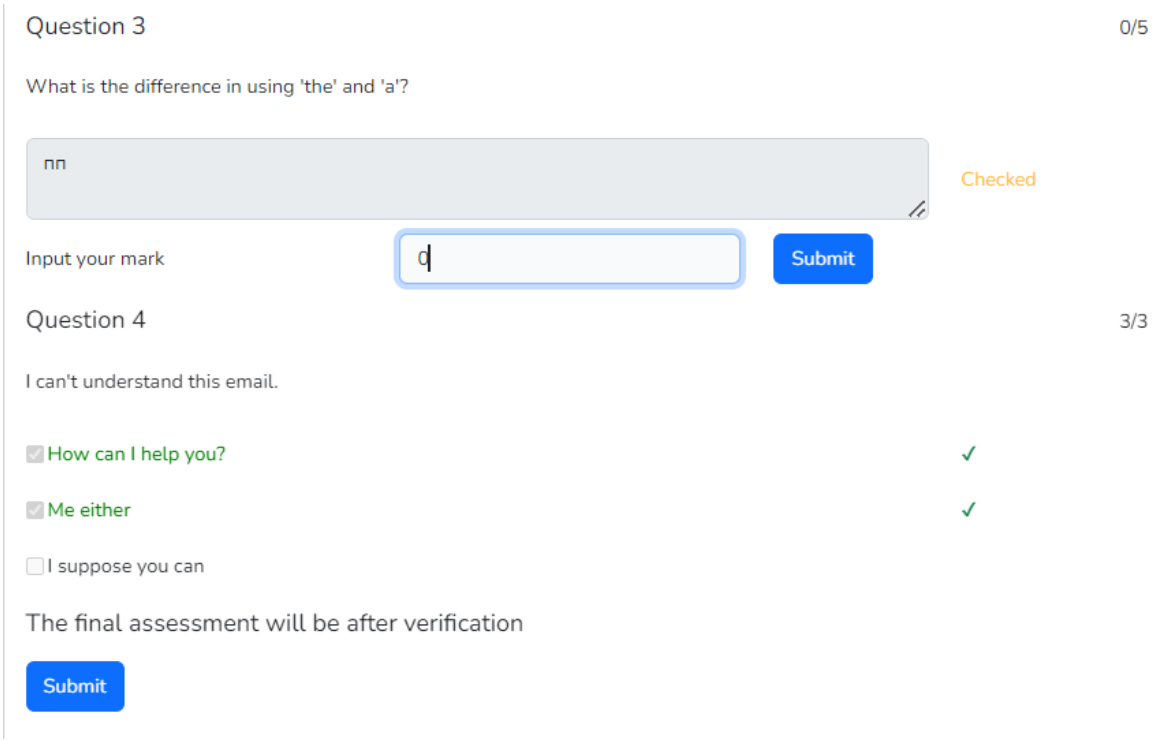


Рисунок 4.27 – Web-сторінка перевірки тесту користувачем

ВИСНОВКИ

У ході виконання переддипломної практики було виконано проектування інформаційної системи оцінки професійних досягнень співробітників IT-компанії відповідно до основних функціональних вимог:

- можливість обробки запиту на перегляду заробітної плати;
- можливість обробки запиту на зміну позиції та ролі робітника компанії;
- можливість призначення зміни проекту для співробітника;
- наявність функції написання відгуку про працівника;
- можливість проведення тестування професійних та особистісних навичок робітників;
- можливість завантаження сертифікатів про проходження курсів на інших платформах;

Розроблений програмний продукт матиме попит серед IT-компаній та буде нести практичну цінність, оскільки дозволить автоматизувати вищеописані бізнес-процеси.

У ході роботи над проектом було визначено потребу в розробці інформаційної системи для організації процесу оцінки професійних досягнень співробітників IT-компанії за рахунок проведення детального аналізу предметної області. Також було обрано декілька аналогів запропонованої ІС для проведення порівняльного аналізу, у результаті якого було визначено які переваги варто використати у проекті, а які недоліки треба подолати.

Наступним етапом роботи над дипломним проектом була ідентифікація та деталізація мети, постановка функціональних та нефункціональних вимог до системи, задач на реалізацію. Також було проведено планування робіт для оцінки часових рамок проекту.

Далі розробником було проведено проектування роботи реальних кейсів та варіанти експлуатації інформаційної системи, використовуючи діаграми нотації IDEF0, її декомпозиції на більш конкретні процеси. Даний етап дозволив пропрацювати детально список дійових осіб створеного програмного продукту, його функціональні можливості та рівні доступу до них, розбити загальні завдання на більш дрібні.

Представлена інформаційна система була розробленна з використанням фреймворку Laravel у якості основи для бекенду, MySQL як систему управління базами даних, HTML та CSS для створення верстки web-сторінок, скриптову мову програмування JavaScript для реалізації діаграм та надання динамічності web-сторінкам.

Після завершення етапу розробки інформаційної системи було проведення smoke-тестування. Під час нього не було виявлено блокуючих чи критичних дефектів.

Отже, створена інформаційна система оцінки професійних досягнень співробітників IT-компанії дозволить оптимізувати робочий час управлінської гілки ієрархії робітників компанії, відслідковувати досягнення та розвиток персоналу та спростити процес збору інформації, необхідної для затвердження перегляду заробітної плати чи зміни позиції працівника.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Дідур К. М. СУЧАСНІ МЕТОДИ ОЦІНКИ ПЕРСОНАЛУ [Електронний ресурс] / К. М. Дідур – Режим доступу до ресурсу: https://dspace.dsau.dp.ua/bitstream/123456789/3025/1/1.pdf.

2. РОЗВИТОК ПРОФЕСІЙНОЇ КОМПЕТЕНТНОСТІ УПРАВЛІНСЬКОГО ПЕРСОНАЛУ [Електронний ресурс] – Режим доступу до ресурсу: http://repository.hneu.edu.ua/bitstream/123456789/7258/1/%d0%9b%d0%b8%d1%82%d0%be%d0%b2%d1%87%d0%b5%d0%bd%d0%ba%d0%be%20%d0%86.%d0%92..pdf.

3. Estimation of worker encouragement system at industrial enterprise [Електронний ресурс] – Режим доступу до ресурсу: https://www.revistaespacios.com/a18v39n28/a18v39n28p22.pdf.

4. ПЕРЕВАГИ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ В БІЗНЕСІ [Електронний ресурс] – Режим доступу до ресурсу: https://conferences.vntu.edu.ua/index.php/mn/mn2021/paper/viewFile/11292/9401.

5. Оцінка персоналу : автоматизація за допомогою програмного забезпечення [Електронний ресурс] – Режим доступу до ресурсу: https://tqm.com.ua/ua/1s-avtomatyzatsiia-haluzi-industry/1s-ocinka-personalu.

6. HURMA – преміальний HR-продукт, який рекомендують гуру галузі [Електронний ресурс] – Режим доступу до ресурсу: https://hurma.work/.

7. 9 Rules For UX Design [Електронний ресурс] – Режим доступу до ресурсу: https://usabilitygeek.com/rules-for-ux-principles/.

8. Методи і техніка досліджень [Електронний ресурс] – Режим доступу до ресурсу: https://elib.tsatu.edu.ua/dep/mtf/ophv_10/page3.html.

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

10. 17 переваг фреймворку Laravel [Електронний ресурс] – Режим доступу до ресурсу: https://icstudio.online/post/17-perevag-frejmvorku-laravel.

11. Створення схем IDEF0 [Електронний ресурс] – Режим доступу до ресурсу: https://support.microsoft.com/uk-ua/office/%D1%81%D1%82%D0%B2%D0%BE%D1%80%D0%B5%D0%BD%D0%BD%D1%8F-%D1%81%D1%85%D0%B5%D0%BC-idef0-ea7a9289-96e0-4df8-bb26-a62ea86417fc.

12. Use Case Diagram Tutorial (Guide with Examples) [Електронний ресурс] – Режим доступу до ресурсу: https://creately.com/blog/diagrams/use-case-diagram-tutorial/.

13. What is a High Level Design (HLD)? [Електронний ресурс] – Режим доступу до ресурсу: https://ipwithease.com/what-is-a-high-level-design-hld/.

14. Relational database structure [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/docs/es/mam/7.6.0?topic=design-relational-database-structure.

15. What is Git: Features, Command and Workflow in Git [Електронний ресурс] – Режим доступу до ресурсу: https://www.simplilearn.com/tutorials/git-tutorial/what-is-git.

ДОДАТОК А

Планування робіт

Попит автоматизації бізнес процесів компаній будь-якого напрямку зростає щодня. Наразі це питання не лише задоволення потреб внутрішніх користувачів, а й впровадження нововведень задля забезпечення гнучкості бізнесу та ефективності роботи з клієнтами. Тому завдяки автоматизації внутрішніх процесів оцінки роботи працівників IT-компанії можна скоротити час аналізу діяльності, успіхів та досягнень співробітників, мотивувати їх розвиватися у нових напрямках, щоб в подальшому мати більш кваліфікований та обізнаний персонал.

Деталізація мети методом SMART. Постановка однозначної та зрозумілої мети на етапі концептуального проектування – це ключ до успіху всього проекту. Результати деталізації мети проекту методом SMART представлено у таблиці А.1

Таблиця А.1 – Деталізація мети проекту методом SMART

Specific (Конкретна)	Створити ІС оцінки професійних досягнень співробітників IT-компанії з використанням сучасних інструментів розробки програмного забезпечення для підтримки організації процесів оцінки професійних досягнень співробітників IT-компанії.
Measurable (Вимірювана)	Результатом роботи проекту є розроблена інформаційна система для оцінювання професійних здобутків працівників IT-компанії.
Achievable (Досяжна)	Для виконання проекту наявні необхідні знання HTML, CSS, мови програмування JavaScript, PHP, фреймворку Laravel, баз даних MySQL та навичок написання документації. Враховуючи доступні ресурсні можливості та обмеження мета є такою, яку можливо досягти.

Продовження таблиці А.1

Relevant (Реалістична)	Розроблена інформаційна система дозволить автоматизувати процес оцінки досягнень співробітників ІТ-компанії та позитивно вплине на якість роботи персоналу за рахунок за рахунок збільшення мотивації.
Time-framed (Обмежена у часі)	Ціль має часове обмеження. Термін досягнення мети проекту визначено за домовленістю між замовником та виконавцем і складає два місяці.

Планування змісту робіт. WBS (Work Breakdown Structure) – це графічна структура елементів проекту, що ієрархічно розташовані та поєднані з продуктом проекту. Метою розробки ієрархічної моделі робіт є організація командної роботи, контроль та оцінка термінів роботи. На найвищому рівні знаходиться продукт проекту. Заходи для забезпечення досягнення мети розташовані на другому рівні. Подальші рівні складають елементи дрібної декомпозиції. Декомпозиція задач виконується до тих пір, поки вид робіт не стане елементарно простим. Елементарними роботами називають дії, виконання яких спрямовано на однозначний та чіткий результат, призначена відповідальна особа, для якої можна обчислити терміни виконання та витрати праці. На рисунку А.1 представлено WBS на розробку інформаційної системи оцінки професійних досягнень співробітників ІТ-компанії.

Рисунок А.1 – WBS-структура робіт проекту

Планування структури виконавців. Розробка організаційної структури виконавців або OBS є наступним етапом після декомпозиції процесів. OBS (Organizational Breakdown Structure) – графічний вигляд структури відповідальних осіб, що беруть участь у розробці проекту та призначені на виконання елементарних робіт. На рисунку А.2 представлено організаційну структуру планування робіт проекту. Учасників проекту описано у таблиці А.2.

Таблиця А.2 – Виконавці проекту

Роль	Ім’я	Проектна роль
Розробник	Чмутенко А.В.	Виконує frond-end та back-end розробку
Проектувальник	Чмутенко А.В.	Виконує проектування бази даних та розробляє структуру інформаційної системи
Тестувальник	Чмутенко А.В.	Проводить тестування інформаційної системи
Керівник проекту	Антипенко В.П.	Формує завдання на розробку проекту
Менеджер проекту	Чмутенко А.В.	Відповідає за дотримання термінів, розподіл ресурсів та завдань між учасниками проекту. Виконує збір та аналіз даних.

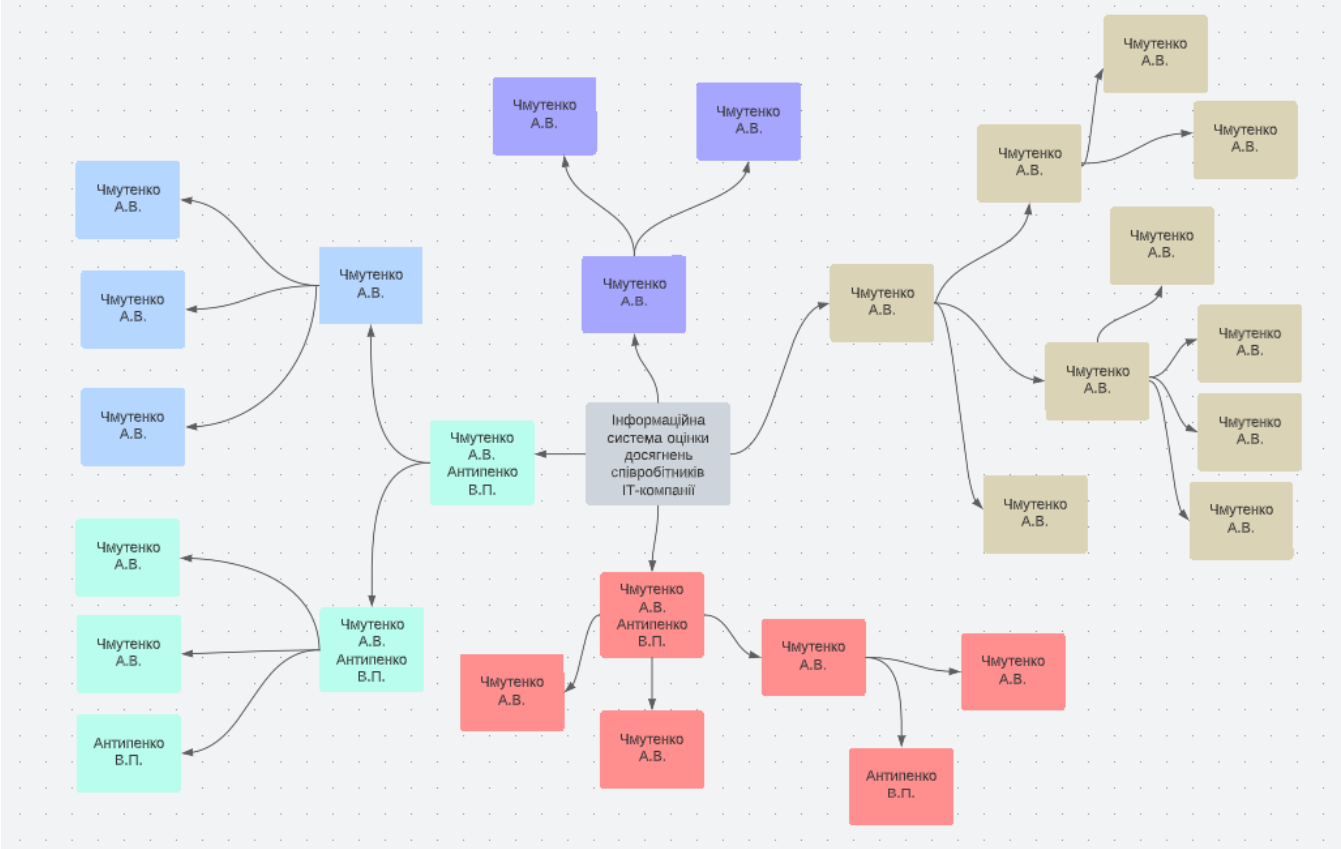


Рисунок А.2 – OBS-структура робіт проекту

Діаграма Ганта. Побудова календарного плану проекту є важливим етапом роботи над проектом та виглядає як графік виконання робіт з реальним розподілом дат. Даний календарний план надає можливість мати достовірне уявлення про тривалість виконання етапів проекту, враховуючи обмеження у ресурсах.

Календарний графік проекту представлено на рисунку А.3.

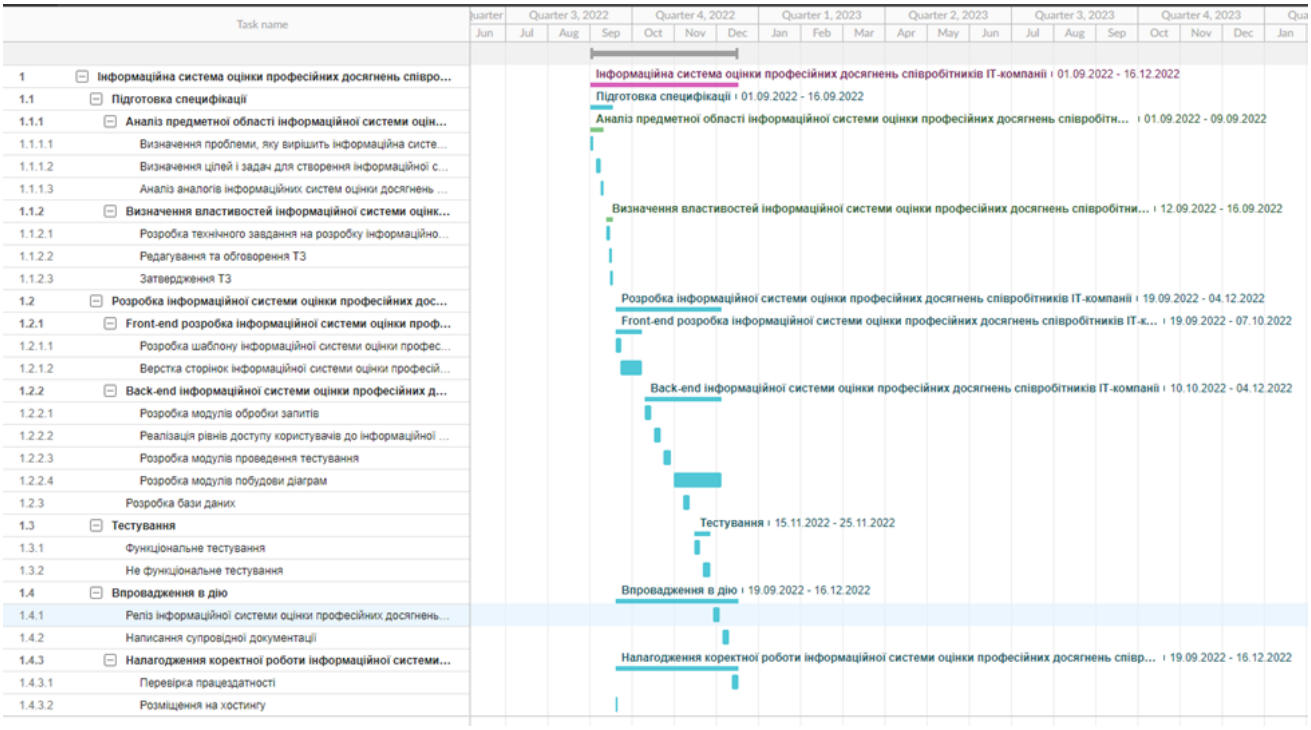


Рисунок А.3 – Календарний графік виконання робіт

Управління ризиками проекту. Наступним етапом планування робіт проекту є оцінка та робота над ризиками: кількісне та якісне оцінювання та заходи для усунення або зменшення впливу ризиків на проект. У таблиці А.3 представлено шкалу оцінки ризиків за класифікацією, величиною впливу та ймовірністю виникнення.

Таблиця А.3 – Шкала оцінювання ризиків за ймовірністю виникнення на величиною впливу.

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

Для того, щоб знизити негативний вплив 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 А.3. У результаті планування реагування було отримано матрицю ймовірності виникнення ризиків та впливу ризику, що зображена на рисунку А.4. Зеленим кольором на матриці позначають прийнятні ризики, жовтим – виправдані, а червоним – недопустимі.

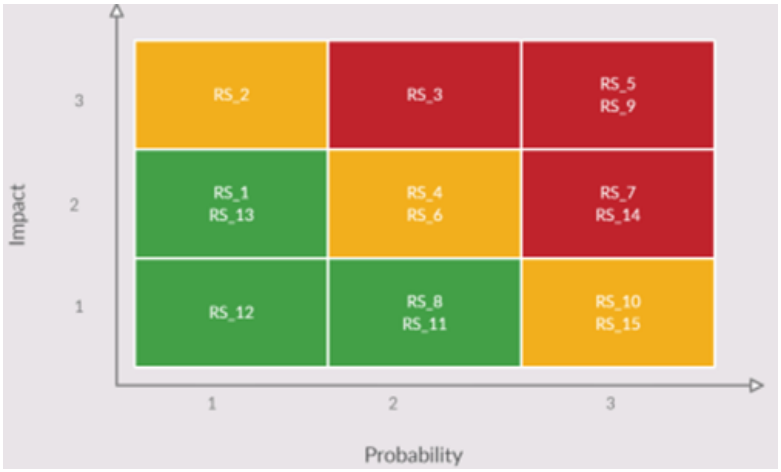


Рисунок А.4. – Матриця ймовірності

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

Таблиця А.4 – Шкала оцінювання за рівнем ризику.

№	Назва	Межі	Ризики, які входять(номера)
1	Прийнятні	1≤R≤2	1,8,11,12,13
2	Виправдані	3≤R≤4	2,4,6,10,15
3	Недопустимі	6≤R≤9	3,5,7,9,14

Таблиця А.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	Поява альтернативного продукту	Низька	Високий	3	1.Провести попереднє дослідження альтернативних продуктів. 2. Вибрати унікальну стратегію створення проекту.	Прийняття	
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 А.5

ID	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_5	Відкритий	Неоптимальний розподіл часу	Висока	Високий	9	Провести аналіз актуальності найважливіших процесів та робіт. Звернути особливу увагу на правильність розподілу часу. Правильно визначити пріоритети виконання робіт. Чітко дотримуватися календарного плану.	Пом’якшення	Змінити порядок пріоритетів робіт. Знайти способи оптимізації роботи з вже існуючою розстановкою. Обговорити варіанти внесення поправок до термінів реалізації із замовником.
RS_6	Відкритий	Часте внесення змін у ТЗ	Середня	Середній	4	1.Виділити всі необхідні параметри проекту 2.Чітко описати вимоги до проекту. 3.Обговорити всі технічні засоби виконання проекту та умови реалізації.	Перенесення	Узгодити всі положення з замовником, у разі потреби внести необхідні зміни та поправки.

Продовження таблиці А.5

ID	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_7	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_8	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_9	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 А.5

ID	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_10	Відкритий	Збої в роботі програмного забезпечення	Низька	Високий	3	1.Підготувати резерв програмних засобів. 2.Залучити спеціаліста для усунення збоїв.	Попередження	Замінити програмне забезпечення.
RS_11	Відкритий	Відсутність резервних копій даних	Низька	Середній	2	1.Налаштувати автоматичне збереження даних. 2.Зберігати дані на різних носіях інформації.	Попередження	Робити копію даних після кожного виконаного етапу.
RS_12	Відкритий	Реалізація непотрібного функціоналу	Низька	Низький	1	Попередити замовника про можливість додаткового функціоналу.	Використання	Обговорити вигоди і збитки від можливих змін проекту.
RS_13	Відкритий	Невиконання моніторингу проекту	Середня	Низький	2	Здійснювати проміжний контроль результатів в ході виконання проекту. Здійснювати моніторинг проекту працівниками.	Перенесення	Здійснювати моніторинг проекту замовником. Надання проміжних результатів виконання проекту після кожного етапу.

Продовження таблиці А.5

ID	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	План А	Тип стратегії реагування	План Б
RS_14	Відкритий	Виникнення проблем із програмним забезпеченням користувачів	Середня	Високий	6	1.Розробка проекту з урахуванням вимог до програмного забезпечення користувачів проекту. 2.Модифікація проекту з урахуванням різних версій програмного забезпечення, яке буде застосовуватися.	Прийняття	
RS_15	Відкритий	Зміна вимог замовника в процесі розробки проекту	Низька	Високий	3	Узгодити всі питання на початкових етапах, щоб мінімізувати кількість змін під час розробки.	Пом’якшення	Переоцінка проекту, кожного разу, коли вимоги змінюються

ДОДАТОК Б

Лістинг програмного коду

AdminBaseController.php

```
<?php

namespace App\Http\Controllers\Admin;

use App\Attributes\DefaultValueAttribute;
use App\Attributes\EditFieldAttribute;
use App\Attributes\FilterFieldAttribute;
use App\Attributes\SelectFieldAttribute;
use App\Attributes\ShowFieldAttribute;
use App\Attributes\UploadFileAttribute;
use App\Data\Field;
use App\Data\FilterField;
use App\Http\Controllers\Controller;
use App\Service\Filters\FilterService;
use App\Service\Resolvers\AdminRoleBaseFieldResolver;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Http\Request;
use Illuminate\Support\Collection;
use Illuminate\Support\Facades\Storage;
use ReflectionClass;

abstract class AdminBaseController extends Controller
{
  abstract protected function getModel(): string;

  public function index(Request $request)
  {
    $model = $this->getModel();
    $filterService = new FilterService($model);
    $items = $filterService->filter($request->all('filters')['filters']);
    $filterFields = [];

    $reflectInstance = new \ReflectionClass($model);
    $showableFields = $reflectInstance->getProperties();
    $headers = [];
    foreach ($showableFields as $property) {
      if($property->getAttributes(ShowFieldAttribute::class)) {
        $headers[] = ucfirst($property->name);
      }
      if($property->getAttributes(FilterFieldAttribute::class)) {
        $field = new FilterField();
        if(!$property->getType()->isBuiltin()) {
          $instance = new ($property->getType()->getName())();
          if($instance instanceof Model) {
            $field->setRelations($instance::all());
            $field->setType("relation");
          }
        } else if($attr = $property->getAttributes(SelectFieldAttribute::class)) {
          $field->setType("select");
          $field->setRelations(new Collection($attr[0]->getArguments()[0]));
        } else {
          $field->setType($property->getType()->getName());
        }
        $field->setName("filters[".$property->getName()."]");
        $field->setLabel(ucfirst($property->getName()));
```

```
        $filterFields[] = $field;
      }
    }

    return view('erp.admin.layouts.index', [
      'create_route' => $this->getBaseRouteName().'.create',
      'update_route' => $this->getBaseRouteName().'.edit',
      'items' => $items, 'headers' => $headers,
      'filters' => $filterFields
    ]);
  }

  public function create() {
    $model = $this->getModel();

    $reflection = new ReflectionClass($model);
    $fields = [];
    foreach ($reflection->getProperties() as $property) {
      $field = $this->processProperty($property);
      if(!$field) {
        continue;
      }
      $fields[] = $field;
    }

    return view('erp.admin.layouts.create', ['store_route' => $this-
>getBaseRouteName().'.store', 'fields' => $fields]);
  }

  public function store(Request $request) {
    $model = $this->getModel();
    $record = new $model();
    $attributes = $request->validate($model::getValidationRules());
    $reflection = new ReflectionClass($record);
    foreach ($attributes as $field => $value) {
      $property = $reflection->getProperty(lcfirst($field));
      $getter = 'get'.$field;
      if($attrs = $property->getAttributes(UploadFileAttribute::class)) {
        if(!$request->hasFile($field)) continue;
        if($record->$getter()) {
          Storage::delete($record->$getter());
        }
        $value = $request->file($field)->store($attrs[0]->getArguments()[0]);
      }
      $setter = "set" . $field;
      $record->$setter($value);
    }
    $record->save();

    return redirect()->route($this->getBaseRouteName().'.edit', $record)->with('success',
'Successfully created');
  }

  public function edit($recordId) {
    $model = $this->getModel();

    $record = $model::where('id', $recordId)->first();

    $reflection = new ReflectionClass($record);
    $fields = [];
    foreach ($reflection->getProperties() as $property) {
      $field = $this->processProperty($property);
      if(!$field) {
        continue;
      }
```

```
      $getter = 'get'.ucfirst($property->getName());
      $field->setValue($record->$getter());
      $fields[] = $field;
    }

    return view('erp.admin.layouts.edit', ['record' => $record, 'fields' => $fields,
      'update_route' => $this->getBaseRouteName().'.update',
      'destroy_route' => $this->getBaseRouteName().'.destroy',
    ]);
  }

  public function update($recordId, Request $request) {
    $model = $this->getModel();
    $record = $model::find($recordId);
    $attributes = $request-
>validate($record::getValidationRules($record));//Validator::make($request->all(),
$record::getValidationRules())->validated();

    $reflection = new ReflectionClass($record);
    foreach ($attributes as $field => $value) {
      $property = $reflection->getProperty(lcfirst($field));
      $getter = 'get'.$field;
      if($attrs = $property->getAttributes(UploadFileAttribute::class)) {
        if(!$request->hasFile($field)) continue;
        if($record->$getter()) {
          Storage::delete($record->$getter());
        }
        $value = $request->file($field)->store($attrs[0]->getArguments()[0]);
      }
      $setter = "set" . $field;
      $record->$setter($value);
    }

    foreach ($reflection->getProperties() as $props) {
      if(!($attrs = $props->getAttributes(DefaultValueAttribute::class))) {
        continue;
      }
      $setter = "set" . ucfirst($props->getName());
      $record->$setter();
    }

    $record->save();

    return redirect()->route($this->getBaseRouteName().'.edit', $record)->with('success',
'Successfully updated');
  }

  public function destroy($recordId) {
    $model = $this->getModel();
    $model::destroy($recordId);
    return redirect()->route($this->getBaseRouteName().'.index')->with('success',
'Successfully deleted');
  }

  protected function processProperty(\ReflectionProperty $property): ?Field {
    if(!($attr = $property->getAttributes(EditFieldAttribute::class))) {
      return null;
    }
    $args = $attr[0]->getArguments();
    if(count($args) > 0) {
      if(!AdminRoleBaseFieldResolver::canEdit($args[0])) {
        return null;
      }
    }
```

```
    $field = new Field();

    if(!$property->getType()->isBuiltin()) {
      $instance = new ($property->getType()->getName())();
      if($instance instanceof Model) {
        $field->setRelations($instance::all());
        $field->setType("relation");
      } else {
        $field->setType($property->getType()->getName());
      }
    } else {
      if($property->getAttributes(UploadFileAttribute::class)) {
        $field->setType("file");
      } else if($attr = $property->getAttributes(SelectFieldAttribute::class)) {
        $field->setType("select");
        $field->setRelations(new Collection($attr[0]->getArguments()[0]));
      } else {
        $field->setType($property->getType()->getName());
      }
    }
    $field->setName(ucfirst($property->getName()));

    return $field;
  }

  protected function getClassName(): string {
    $explodedNamespace = explode('\\', $this->getModel());
    return $explodedNamespace[count($explodedNamespace)-1];
  }

  protected function getBaseRouteName(): string {
    $className = $this->getClassName();
    return 'erp.admin.'.lcfirst($className).'s';
  }
}
```

TestController.php

```
<?php

namespace App\Http\Controllers\Api\Admin\Tests;

use App\Constants\TestType;
use App\Http\Requests\Admin\TestRequest;
use App\Http\Resource\Profile\Test\QuestionResource;
use App\Models\Employee;
use App\Models\Keyword;
use App\Models\Test\Answer;
use App\Models\Test\Question;
use App\Models\Test\Test;
use App\Models\Test\Variant;
use App\Models\TestKeyword;
use App\Service\Stats\TestService;
use Illuminate\Http\JsonResponse;
use Illuminate\Http\Request;

class TestController
{
  public function create(TestRequest $request) {
    $attributes = $request->validated();

    $test = new Test();
    $test->name = $attributes['name'];
    $test->category_id = $attributes['category'];
```

```
    $test->description = $attributes['description'];
    $test->created_by_id = $request->user()->employee->id;
    $test->save();

    TestKeyword::query()->where('test_id', $test->id)->delete();
    foreach ($attributes['keywords'] as $keyword) {
      if(!$keyword) continue;
      $keywordModel = Keyword::query()->where('word', $keyword)->first();
      if (!$keywordModel) {
        $keywordModel = new Keyword();
        $keywordModel->word = $keyword;
        $keywordModel->save();
      }
      if(!TestKeyword::query()->where('keyword_id', $keywordModel->id)->where('test_id',
$test->id)->exists()) {
        $pivotModel = new TestKeyword(['test_id' => $test->id, 'keyword_id' =>
$keywordModel->id]);
        $pivotModel->save();
      }
    }

    foreach ($attributes['questions'] as $question) {
      $questionModel = new Question();
      $questionModel->question = $question['question'];
      $questionModel->type = $question['type'];
      $questionModel->mark = $question['mark'];
      $questionModel->test_id = $test->id;
      $questionModel->save();

      foreach ($question['variants'] as $index => $variant) {
        $variantModel = new Variant();
        $variantModel->variant = $variant['variant'];
        $variantModel->question_id = $questionModel->id;
        if($questionModel->type == TestType::TEST->name) {
          if($index == $question['correct_variant']) {
            $variantModel->is_correct = true;
          } else {
            $variantModel->is_correct = false;
          }
        } else if($questionModel->type == TestType::MULTIPLE->name &&
array_key_exists('is_correct', $variant)) {
          $variantModel->is_correct = $variant['is_correct'] == "on";
        } else if($questionModel->type == TestType::OPEN->name) {
          $variantModel->is_correct = true;
        } else {
          $variantModel->is_correct = false;
        }

        $variantModel->save();
      }
    }
    return response()->json(['redirectTo' => route('erp.admin.tests.edit', ['test' =>
$test])]);
  }

  public function update(TestRequest $request, Test $test) {
    if(TestService::getEmployeesWhoPassedTest($test)->count() > 0) {
      return response()->json(['error' => 'You can`t edit test if it has been passed!']);
    }

    $attributes = $request->validated();

    $test->name = $attributes['name'];
    $test->category_id = $attributes['category'];
    $test->description = $attributes['description'];
```

```
    $test->created_by_id = $request->user()->employee->id;
    $test->save();

    TestKeyword::query()->where('test_id', $test->id)->delete();
    foreach ($attributes['keywords'] as $keyword) {
      if(!$keyword) continue;
      $keywordModel = Keyword::query()->where('word', $keyword)->first();
      if (!$keywordModel) {
        $keywordModel = new Keyword();
        $keywordModel->word = $keyword;
        $keywordModel->save();
      }
      if(!TestKeyword::query()->where('keyword_id', $keywordModel->id)->where('test_id',
$test->id)->exists()) {
        $pivotModel = new TestKeyword(['test_id' => $test->id, 'keyword_id' =>
$keywordModel->id]);
        $pivotModel->save();
      }
    }

    $test->questions()->delete();

    foreach ($attributes['questions'] as $question) {
      $questionModel = new Question();
      $questionModel->question = $question['question'];
      $questionModel->type = $question['type'];
      $questionModel->mark = $question['mark'];
      $questionModel->test_id = $test->id;
      $questionModel->save();

      foreach ($question['variants'] as $index => $variant) {
        $variantModel = new Variant();
        $variantModel->variant = $variant['variant'];
        $variantModel->question_id = $questionModel->id;
        if($questionModel->type == TestType::TEST->name) {
          if($index == $question['correct_variant']) {
            $variantModel->is_correct = true;
          } else {
            $variantModel->is_correct = false;
          }
        } else if($questionModel->type == TestType::MULTIPLE->name &&
array_key_exists('is_correct', $variant)) {
          $variantModel->is_correct = $variant['is_correct'] == "on";
        } else if($questionModel->type == TestType::OPEN->name) {
          $variantModel->is_correct = true;
        } else {
          $variantModel->is_correct = false;
        }

        $variantModel->save();
      }
    }

    return response()->json(['redirectTo' => route('erp.admin.tests.edit', ['test' =>
$test])]);
  }

  public function updateMark(Answer $answer, Request $request) {
    $attributes = $request->validate(['mark' => 'required|integer']);
    $mark = $attributes['mark'];

    if($mark > $answer->question->mark) {
      return response([], 422);
    }
```

```
    $answer->mark = $mark;
    $totalMark = $answer->question->mark;
    $perCentMark = $mark / $totalMark * 100;
    if($perCentMark < 60) {
      $answer->is_correct = false;
    } else {
      $answer->is_correct = true;
    }
    $answer->save();
    return response()->json([
      'is_correct' => $answer->is_correct
    ]);
  }

  public function get(Test $test, Request $request): JsonResponse
  {
    $attributes = $request->validate([
      'employee_id' => 'required|exists:employees,id'
    ]);
    $employee = Employee::find($attributes['employee_id']);
    $questions = QuestionResource::customCollection($test->questions, $employee);
    $mark = TestService::getMarkForTestByEmployee($test, $employee);
    return response()->json([
      'mark' => $mark,
      'questions' => $questions
    ]);
  }
}
```

Employee.php

```
<?php

namespace App\Models;

use App\Attributes\EditFieldAttribute;
use App\Attributes\FilterClassAttribute;
use App\Attributes\FilterFieldAttribute;
use App\Attributes\ShowFieldAttribute;
use App\Attributes\UploadFileAttribute;
use App\Constants\AdminRole;
use App\Service\Filters\EmployeeFilterService;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Relations\BelongsTo;
use Illuminate\Database\Eloquent\Relations\HasMany;
use Illuminate\Database\Eloquent\Relations\HasOne;
use Illuminate\Support\Carbon;
use Illuminate\Support\Facades\Storage;
use Illuminate\Validation\Rule;

#[FilterClassAttribute(EmployeeFilterService::class)]
class Employee extends Model implements IAdminModel
{
  use HasFactory;

  protected $fillable = ["first_name", "last_name", "photo_path", "date_of_birth",
"employment_date" , "mobile_phone", "working_email", "second_mobile_phone", "address",
"city", "department_id", "position_id", "role_id", "project_id", "salary", "user_id"];

  protected $hidden = [
    'password',
    'remember_token',
```

```
  ];

  #[ShowFieldAttribute,EditFieldAttribute, FilterFieldAttribute('like', 'first_name')]
  private string $firstName;

  #[ShowFieldAttribute,EditFieldAttribute, FilterFieldAttribute('like', 'last_name')]
  private string $lastName;

  #[EditFieldAttribute]
  #[UploadFileAttribute("public/employees/avatars")]
  private string $photo;

  #[ShowFieldAttribute, EditFieldAttribute]
  private Carbon $dateOfBirth;

  #[EditFieldAttribute([AdminRole::MANAGER, AdminRole::HR])]
  private Carbon $employmentDate;

  #[EditFieldAttribute]
  private string $mobilePhone;

  #[EditFieldAttribute, FilterFieldAttribute('like', 'working_email')]
  private string $workingEmail;

  #[EditFieldAttribute]
  private string $secondMobilePhone;

  #[EditFieldAttribute]
  private string $address;

  #[EditFieldAttribute, FilterFieldAttribute('=', 'city')]
  private string $city;

  #[ShowFieldAttribute,EditFieldAttribute([AdminRole::MANAGER, AdminRole::HR]),
FilterFieldAttribute('=', 'department_id')]
  private Department $department;

  #[ShowFieldAttribute,EditFieldAttribute([AdminRole::MANAGER, AdminRole::HR]),
FilterFieldAttribute('=', 'position_id')]
  private Position $position;

  #[ShowFieldAttribute,EditFieldAttribute([AdminRole::MANAGER, AdminRole::HR]),
FilterFieldAttribute('=', 'role_id')]
  private Role $role;

  #[ShowFieldAttribute,EditFieldAttribute([AdminRole::MANAGER, AdminRole::HR]),
FilterFieldAttribute('=', 'role_id')]
  private Project $project;

  #[EditFieldAttribute([AdminRole::MANAGER])]
  private int $salary;

  #[EditFieldAttribute([AdminRole::MANAGER, AdminRole::HR])]
  private User $user;

  public function getFirstName()
  {
    return $this->getAttribute('first_name');
  }

  public function setFirstName($firstName): void
  {
    $this->setAttribute('first_name', $firstName);
  }
```

```
public function getLastName()
{
  return $this->getAttribute('last_name');
}


public function setLastName($lastName): void
{
  $this->setAttribute('last_name', $lastName);
}

public function getPhoto()
{
  if(!$this->getAttribute('photo_path')) return null;
  return Storage::url($this->getAttribute('photo_path'));
}

public function setPhoto($photo): void
{
  $this->setAttribute('photo_path', $photo);
}

public function getDateOfBirth()
{
  return $this->getAttribute('date_of_birth');
}

public function setDateOfBirth($dateOfBirth): void
{
  $this->setAttribute('date_of_birth', $dateOfBirth);
}

public function getEmploymentDate()
{
  return $this->getAttribute('employment_date');
}

public function setEmploymentDate($employmentDate): void
{
  $this->setAttribute('employment_date', $employmentDate);
}

public function getMobilePhone()
{
  return $this->getAttribute('mobile_phone');
}

public function setMobilePhone($mobilePhone): void
{
  $this->setAttribute('mobile_phone', $mobilePhone);
}

public function getWorkingEmail()
{
  return $this->getAttribute('working_email');
}

public function setWorkingEmail($workingEmail): void
{
  $this->setAttribute('working_email', $workingEmail);
}

public function getSecondMobilePhone()
{
  return $this->getAttribute('second_mobile_phone');
```

```
}

public function setSecondMobilePhone($secondMobilePhone): void
{
  $this->setAttribute('second_mobile_phone', $secondMobilePhone);
}

public function getAddress()
{
  return $this->getAttribute('address');
}

public function setAddress($address): void
{
  $this->setAttribute('address', $address);
}

public function getCity()
{
  return $this->getAttribute('city');
}

public function setCity($city): void
{
  $this->setAttribute('city', $city);
}

public function getDepartment()
{
  $this->load('department');
  return $this->getRelation('department');
}

public function setDepartment($department): void
{
  $this->setAttribute('department_id', $department);
}

public function getPosition()
{
  $this->load('position');
  return $this->getRelation('position');
}

public function setPosition($position): void
{
  $this->setAttribute('position_id', $position);
}

public function getRole()
{
  $this->load('role');
  return $this->getRelation('role');
}

public function setRole($role): void
{
  $this->setAttribute('role_id', $role);
}

public function getProject()
{
  $this->load('project');
  return $this->getRelation('project');
}
```

```
public function setProject($project): void
{
  $this->setAttribute('project_id', $project);
}

public function getSalary()
{
  return $this->getAttribute('salary');
}

public function setSalary($salary): void
{
  $this->setAttribute('salary', $salary);
}

public function getUser()
{
  $this->load('user');
  return $this->getRelation('user');
}

public function setUser($user): void
{
  $this->setAttribute("user_id", $user);
}

public static function getValidationRules($record = null): array {
  if($record) {
    $userValidation = 'nullable|exists:users,id|unique:employees,user_id,'.$record->id;
  } else {
    $userValidation = 'nullable|exists:users,id|unique:employees,user_id';
  }
  return [
    'FirstName' => 'required|max:255',
    'LastName' => 'required|max:255',
    'DateOfBirth' => 'required|date|max:255',
    'WorkingEmail' => 'nullable|max:255|email',
    'MobilePhone' => 'nullable|max:255',
    'SecondMobilePhone' => 'nullable|max:255',
    'Address' => 'nullable|max:255',
    'City' => 'nullable|max:255',
    'Salary' => 'nullable|integer|max:100000|min:100',
    'User' => $userValidation,
    'Project' => 'nullable|exists:projects,id',
    'Role' => 'nullable|exists:roles,id',
    'Position' => 'nullable|exists:positions,id',
    'Department' => 'nullable|exists:departments,id',
    'Photo' => 'nullable|image|max:2048'
  ];
}

public function getPhotoLinkAttribute() {
  if(!$this->photo_path) {
    return null;
  }
  return Storage::url($this->photo_path);
}

public function department(): BelongsTo {
  return $this->belongsTo(Department::class, "department_id", "id");
}

public function getFullNameAttribute(): string {
  return $this->first_name." ".$this->last_name;
```

```
  }

  public function getNameForForm(): string
  {
    return $this->fullName;
  }

  public function user(): BelongsTo {
    return $this->belongsTo(User::class, "user_id", "id");
  }

  public function feedbacks(): HasMany {
    return $this->hasMany(Feedback::class, "employee_id", "id")->orderBy("updated_at",
"desc");
  }

  public function requestSalaries(): HasMany {
    return $this->hasMany(RequestSalary::class, "employee_id", "id")->orderBy("updated_at",
"desc");
  }

  public function requestPositions(): HasMany {
    return $this->hasMany(RequestPosition::class, "employee_id", "id")-
>orderBy("updated_at", "desc");
  }

  public function requestProjects(): HasMany {
    return $this->hasMany(RequestProject::class, "employee_id", "id")->orderBy("updated_at",
"desc");
  }

  public function requestRoles(): HasMany {
    return $this->hasMany(RequestRole::class, "employee_id", "id")->orderBy("updated_at",
"desc");
  }

  public function certificates(): HasMany {
    return $this->hasMany(Certificate::class, "employee_id", "id")->orderBy("created_at",
"desc");
  }

  public function salaryLog(): HasMany {
    return $this->hasMany(SalaryLog::class, "employee_id", "id")->orderBy("created_at",
"desc");
  }

  public function project(): HasOne {
    return $this->hasOne(Project::class, "id", "project_id");
  }

  public function position(): HasOne {
    return $this->hasOne(Position::class, "id", "position_id");
  }

  public function role(): HasOne {
    return $this->hasOne(Role::class, "id", "role_id");
  }
}
```