Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра програмної інженерії
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка веб орієнтованої програми для обліку працівників на підприємстві

Виконав(ла): студент(ка) IV курсу, групи СПз-41
спеціальності
121 – Інженерія програмного забезпечення
(шифр і назва спеціальності)

_____ Григорів В.В.
(підпис) (прізвище та ініціали)

Керівник _____ Пастух О.А.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ Стоянов Ю.М.
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Петрик М.Р.
(підпис) (прізвище та ініціали)

Рецензент _____ _____
(підпис) (прізвище та ініціали)

Тернопіль 2024

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет Комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра Програмної інженерії
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Петрик М.Р.
(підпис) (прізвище та ініціали)
« » 2024 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавр
(назва освітнього ступеня)

за спеціальністю 121 Програмна інженерія
(шифр і назва спеціальності)

студенту Григоріву Владиславу Віталійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка веб орієнтованої програми для обліку працівників на підприємстві

Керівник роботи Пастух Олег Анатолійович, д.т.н., проф.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «_15__» *квітня* 2024 року № *4/7-340*

2. Термін подання студентом завершеної роботи

3. Вихідні дані до роботи

4. Зміст роботи (перелік питань, які потрібно розробити)

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

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності,	Гурик О.Я., доцент кафедри МТ		
основи охорони праці	Лазарюк В.В., доцент кафедри МТ		

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	**Примітка**
1	Отримання завдання		
2	Аналіз завдання		
3	Виконання розділу 1		
4	Виконання розділу 2		
5	Виконання розділу 3		
7	Оформлення пояснювальної записки		
8	Оформлення графічного та презентаційного матеріалу		
9	Попередній захист		
10	Захист		

Студент _____ **Григорів В.В.**
(підпис) (прізвище та ініціали)

Керівник роботи _____ **Пастух О.А.**
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Кваліфікаційна робота має: с. ___, рисунків ___, таблиць ___, літ. ___
предмента область, мета, завдання, об’єкт, розробка, програмна система, вимоги, середовище проґрамування, Microsoft Visual Studio, C#, сервер, база даних

В роботі розроблено веб-орієнтовану програму обліку працівників на пдприємстві. Програма орієнтована на покращення ефективності роботи відділу кадрів.

ABSTRACT

subject area, goal, task, object, development, software system, requirements, programming environment, Microsoft Visual Studio, C#, server, database

In the work, a web-oriented program for accounting of employees at the enterprise was developed. The program is aimed at improving the efficiency of the HR department.

ЗМІСТ

ВСТУП

Відділ кадрів – один із головних відділів будь-якого підприємства, тому його намагаються автоматизувати насамперед. Дійсно, розхожа фраза про те, що «кадри вирішують все» стає дуже актуальною при вирішенні питання про те, які служби слід автоматизувати раніше за інших. При написанні кваліфікаційної роботи з автоматизації обліку кадрів слід розробити таку веб-орієнтовану програму, яка забезпечуватиме підтримку діяльності співробітників відділу.

В основі програми, безумовно, лежатиме база даних працівників підприємства. Проектування та розробка бази даних – завдання, яке доводиться вирішувати практично у кожній кваліфікаційній роботі з інженерії програмного забезпечення. Особливість бази даних, що використовується відділом кадрів, – її багатоплановість. Про кожного співробітника підприємства в ній повинна зберігатися не тільки загальна інформація (ПІБ, рік народження, посада тощо), а й персоніфіковані дані (наприклад, наявність категорії, сертифікату, наукового ступеня тощо). Отже, БД повинна мати таку структуру, яка дозволяє враховувати різнорідні дані для співробітників, причому у ній має бути надмірності, тобто. її слід оптимізувати.

Ще одна особливість бази даних, призначеної для автоматизації обліку кадрів, – автоматичне оновлення даних у ній. Так, наприклад, стаж роботи співробітника повинен автоматично збільшуватися на одиницю у строго визначений кожному за співробітника термін. Також автоматично повинні відслідковуватись терміни закінчення дії трудового договору, ліцензії, сертифіката тощо.

При автоматизації роботи відділу кадрів слід приділити особливу увагу вирішенню завдання багатовимірної оптимізації. Можливо, у рамках кваліфікаційної роботи бакалавра доведеться реалізувати кілька різних методів оптимізації. Оскільки вони можуть бути різноплановими і не завжди їх можна вирішити ідентичними методами. Дійсно, одна річ вибрати з двох-трьох

кандидатур того співробітника, який більше за інших підходить для заміщення вакантної посади і зовсім інша – скласти графік відпусток так, щоб посадові обов'язки тимчасово відсутньої людини могли виконувати її колеги та виробничі процеси при цьому не постраждали б. Обидві ці завдання – це завдання оптимізації, але вирішувати їх доцільно по-різному. В ідеальному варіанті система повинна надавати користувачеві можливість вибору методу, але при цьому деякі методи слід використовувати як рекомендований.

Ще одне важливе завдання, яке вирішує відділ кадрів – облік часу роботи та табелювання. При вирішенні цього завдання потрібно забезпечити взаємодію інформаційної системи (ІС) з іншими відділами: наприклад, з автоматизованою системою перепусток, якщо така є, з бухгалтерією і т.д.

Загалом, відділ кадрів вирішує велику кількість різноманітних завдань. Тому, при написанні кваліфікаційної роботи необхідно чітко сформулювати ті завдання, які планується вирішити, оскільки комплексна автоматизація відділу – надто складна проблема для окремої кваліфікаційної роботи.

У моїй роботі буде представлена ІС обліку кадрів у вигляді веб-орієнтованої програми, яка дозволяє зручно та швидко виконувати необхідні завдання з місць забезпечених доступом до Інтернету.

Робота є актуальною для сучасного бізнес-середовища, оскільки збір, збереження та обробка даних про працівників підприємства є об’ємною та громіздкою, а розроблювана програма дозводить значно скоротити часові затрати шляхом автоматизації багатьох рутинних адміністративних рішень щодо обліку кадрів.

Метою моєї роботи є розробка веб-орієнтованої програми обліку кадрів на підприємстві.

Аналіз літературних джерел та інформаційних ресурсів: цей метод дозволяє отримати теоретичні знання про вже відомі ІС обліку кадрів.

Порівняння: цей метод дозволяє аналізувати та порівнювати існуючі інформаційні системи обліку кадрів .

Моделювання: цей метод включає розробку та тестування програмного забезпечення для інформаційної системи обліку кадрів.

Комп'ютерне моделювання: цей метод використовується для розробки прототипів програмного забезпечення та тестування його роботи в різних умовах.

На практиці, отримані результати забезпечать точність та надійності даних в системі обліку кадрів, автоматизації процесів. В результаті цього підприємство зможе більш ефективно облікувати своїх працівників, знизити ризики та забезпечити оптимальне використання ресурсів.

1 РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ

1.1 Аналіз вимог та опис предметної області

В даний час інформаційні системи широко застосовуються в багатьох галузях професійної діяльності. Практично жодна організація не може обійтися в роботі без застосування комп'ютерів, які замінять рутинну роботу, підвищуючи ефективність.

Управління персоналом – завдання першорядної важливості. Створення умов для залучення та утримання найкращих співробітників, своєчасна підготовка кадрових ресурсів - необхідні складові підприємства для ухвалення управлінських рішень, де необхідно володіти актуальною та достовірною інформацією про персонал. При цьому можливості використання систем подібного роду залежить від багатьох пов'язаних чинників.

На підприємстві обліком персоналу займається відділ кадрів до функцій якого відноситься відбір та опрацювання даних про працівників, оформлення пакету документації на них: особової картки встановленої форми, ведення трудової книжки, укладання трудового договору, формування характеристики на працівника.

Таким чином кадровий облік це цілеспрямовані дії по упорядкуванню інформації про працюючих на підприємстві, які не залежать від його організації, форми власності тощо. Метою є систематизація даних про працініків [1].

Кадрова служба дозволяє оптимізувати процес керуввання підприємством та допомагає визначити необхідність кількісного та якісного складу його [2].

Я вирішив розробити веб-орієнтовану програму, яка допоможе автоматизувати процес обліку кадрів і дозволить швидкий та зручний доступ до системи, не зважаючи на місцезнаходження, голвне мати доступ до інтернету.

Дана програма має забезпечити просту взаємодію без встановлення додаткового ПЗ. Також вона придатна до використання на різних платформах, що вказує на її гнучкість та доступність і дозволяє використовувати в різних

режимах роботи.

1.2 Аналіз стану проблеми

Насьогодні є багато програм обліку кадрів, різниця між якими у функціональних можливостях, ціні та складності використання. У цьому підрозділі розгляну найпопулярніші та проведу їх порівняльний аналіз.

Таблиця 1.1 – Порівняльний аналіз програм обліку кадрів.

Функції та можливості	**Міні-кадри**	**Кадри Плюс**	**Чиж-Кадри**	**Бухгалтерія 8**
Віддалений доступ	–	+	–	+
Можливість входу з будь-якого пристрою	–	–	–	–
Ведення особистої картки користувача	+	+	+	+
Захист інформації	+	+	+	+
Мобільність	–	–	–	–
Багатоплатформенність	–	+	–	+
Пошук працівників за параметрами	–	+	+	+
Збір статистичних даних	–	+	+	–

1.3 Моделюванн бізнес-процесів (БП)

IDEF3 – спосіб описуввння БП із застосуванням структурованого методу, який дозволить експерту висвітлити стан ситуації у виді впорядкованої

послідовності подій разом із описом об’єктів, які безпосередньо відносяться до процесу. Дана технологія уможливлює збір даних необхідних, щоб провести структурний аналіз системи. Також в ній немає жорстких обмежень (синтаксичних або семантичних), які принесуть незручність при описові неповних (нецілісних) систем. Автору моделі не потрібно поєднувати власні припущення стосовно функціональних можливостей системи з експерними, щобб заповнити порожні місця під час опису предметної галузі.

Також є можливість використання IDEF3 як методу проєктування БП. IDEF3-моделювання є органічним доповненням традиційного при використанні стандарту методології IDEF0.

Основою моделі IDEF3 є сценарій БП. В ньому прописана послідовність дій чи підпроцесів системи, яка аналізується. Також у ньому визначені призначення та межі моделі, важливо підібрати відповідні імена для того, щоб позначити дії. [8].

Враховуючи вище сказане і той факт, що прийом працівників на роботу та їх облік є складним кадровим процесом. Тому застосуєм метод декомпозиції, який дозволить замінити складне завдання вирішенням набору меншскладних. Розглянем це на прикладі прийому нового працівника. Спершу поступає заява та відповідні документи на нього. Після чого інформація проходить перевірку і вирішується питання щодо прийому на роботу працівника. При позитивному рішенні його реєструють в базі, формують штатний розпис та встановлюють графік виходу на роботу. Останнім кроком є звітні документи.

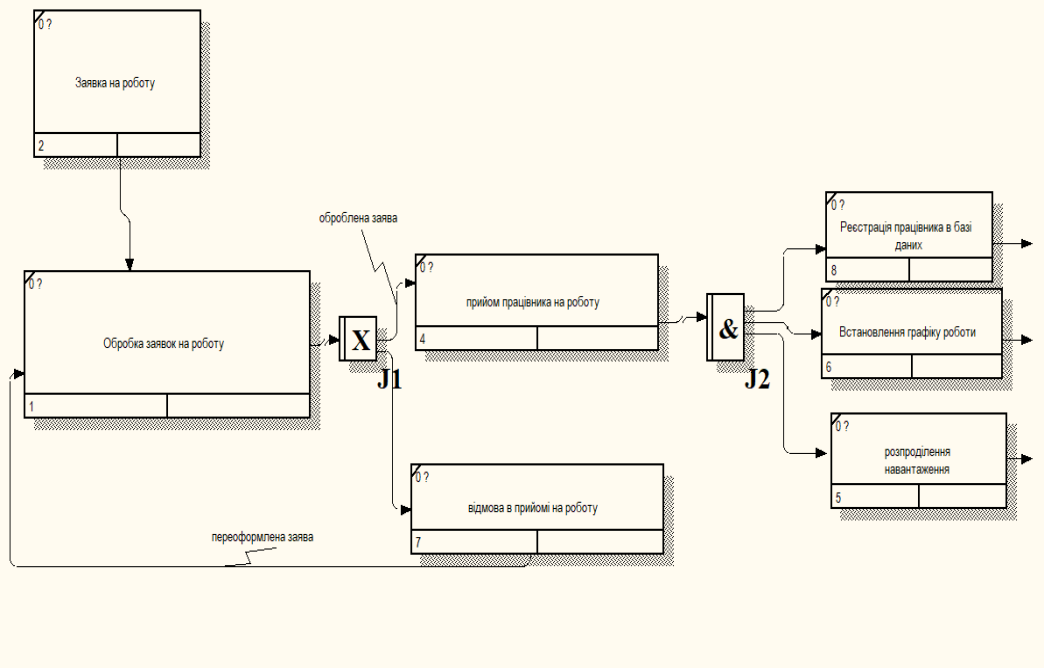


Рисунок 1.1 – Декомпозиція діаграми «Кадровий облік»

1.4 Інструменти ІС кадрового обліку

HTML – фундаментальний блок Інтернету. За допомогою гіпертекстової розмітки визначається значення та структура веб-вмісту. На відміну від HTML, інші технології використовують для описування зовнішнього виду/представлення веб-сторінки (CSS) чи поведінки / функціональності (JavaScript). Іншими словами — це набір символів або кодів розмітки, вставлених у файл, призначений для показу в Інтернеті. Розмітка повідомляє веб-браузерам, як відображати слова та зображення веб-сторінки.

Кожна окрема частина коду розмітки (між символами «<» і «>») називається елементом, хоча багато людей також називають його тегом. Деякі елементи постачаються парами, які вказують, коли певний ефект відображення повинен початися і коли він має закінчитися.

Гіпертекст — це метод, за допомогою якого користувачі Інтернету переміщуються в Інтернеті. Натискаючи спеціальний текст, який називається

гіперпосиланнями, користувачі потрапляють на нові сторінки. Використання гіпер означає, що воно не є лінійним, тому користувачі можуть перейти в будь-яке місце в Інтернеті, просто натиснувши доступні посилання.

Розмітка – це те, що теги HTML роблять із текстом усередині них; вони позначають його як певний вид тексту. Наприклад, текст розмітки може бути виділений жирним шрифтом або курсивом, щоб привернути особливу увагу до слова чи фрази [9].

Bootstrap — це потужний інтерфейсний багатофункціональний інструментарій. Застосовується від створення прототипів до виробництва. Це безкоштовна інтерфейсна платформа розробки з відкритим кодом для створення веб-сайтів і веб-застосунків. Використовується для розроблення мобільного веб-сайту. Bootstrap пропонує різноманітний синтаксис дизайну шаблонів. Він використовує елементи HTML і властивості CSS, для яких потрібен тип документа HTML5. Bootstrap3 так розроблений, щоб було можливо реагувати на мобільні девайси.

У той час як CSS забезпечує основу для веб-стилю, пропонуючи необмежене налаштування, Bootstrap спрощує веб-розробку за допомогою готових до використання компонентів, обмінюючи частину гнучкості CSS на швидкість [10].

Laravel — є фреймворком веб-застосунків із чітким та зрозумілим синтаксисом. Він забезпечує структуру і стартову точку для створення програми, дозволяючи зосередитися на створенні чогось цікавого, поки ми працюємо над деталями.

Laravel надає розробнику неймовірний досвід, одночасно надаючи такі потужні функції, як ретельне впровадження залежностей, виразний рівень абстракції бази даних, черги та заплановані завдання, модульне та інтеграційне тестування тощо.

Під час створення веб-додатку є доступні різноманітні інструменти та фреймворки. Однак я вважаю, що Laravel є ннакрщий вибір для розробки сучасних повноцінних веб-застосунків]11 , 12].

MySQL — це безкоштовна та дуже популярна система управління

реляційними БД із відкритим кодом, яка розробляється, розповсюджується та підтримується корпорацією Oracle.

Ключові особливості:

- Системи управління реляційними базами даних з відкритим кодом.
- Надійний, дуже швидкий і простий у використанні сервер бази даних.
- Працює за моделлю клієнт-сервер.
- Надзвичайно безпечний і масштабований
- Висока ефективність
- Висока продуктивність, оскільки він використовує збережені процедури, тригери, представлення для написання високопродуктивного коду.
- Ефективно підтримує великі бази даних.
- Підтримує багато операційних систем, таких як Linux, CentOS, Solaris, Ubuntu, Windows, MacOS, FreeBSD та інші.

1.5 Процес моделювання системи обліку кадрів

Діаграма послідовності використовується для візуалізації взаємодії між об’ктами в системі. Вона показує порядок повідомлень якими об’кти обмінюються один із одним в часі та їх життєвий цикл. На мій погляд, це самий простий та зручний інструменет для демонстрації всіх інтеграцій та взаємодій в рамках проєктованого бізнес-процесу.

Об’єкти – це сутності, що взаємодіють одна з одною. Незважаючи на назву, вони можуть бути різних типів.

Лінії життя відображають плин часу та активності чи виконання певних функцій об’єкту. Являють собою вертикальну лінію з блоком активності.

Повідомлення показують обмін інформацією між об’єктами. Це найчастіше може бути запитом чи відповіддю, викликом збереженої процедури чи відправка повідомлення. Вони відображуються у вигляді стрілок, які прямують від однієї лінії життя до іншої та бувають декількох типів. Важливим

є поняття групування повідомлень. Це потрібно для того, щоб зробити компонування повідомлень та накласти певні правила для окремо взятих повідомлень.

Отже, на основі поданої інформації побудуємо UML діаграму. На ній відображено як взаємодіє користувач, веб-сервер та база даних та система кадрового обліку, коли додається інформація про працівника. Цей процес відбувається при відкриванні веб-сайту та подальшій роботі із веб-застосунком. Від веб-сервера до бази даних надсилається запит, який перевіряє наявність необхідних дозволів і видає бажаний результат.. Після проходження дані надсилаються до бази даних, перевіряються, і надходять до користувача.

Побудувавши UML діаграму приступимо до побудови діаграми діяльності. Вона є по суті блок-схемою в якій показано перехід від одної дії до наступної. Цю діяльність характеризують як орегацію системи. Основною метою цих діаграм є відображення динамічної поведінки системи. Ще вона отримала назву об’єкт-орієнтованої блок схеми.

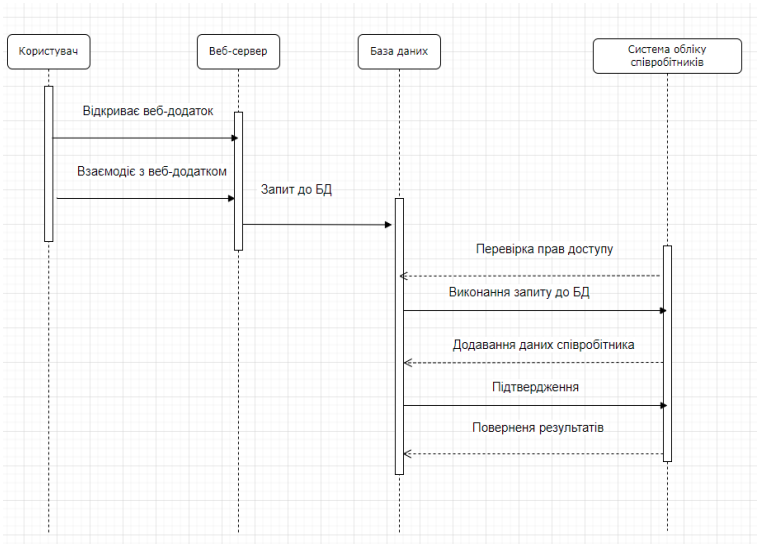


Рисунок 1.2 – UML діаграма

Фокусом діаграми є виконання, а не реалізаці. Діаграма фокусована на виконання, а не на реалізацію і містить дії, які застосовуються при моделюванні поведінки системи.

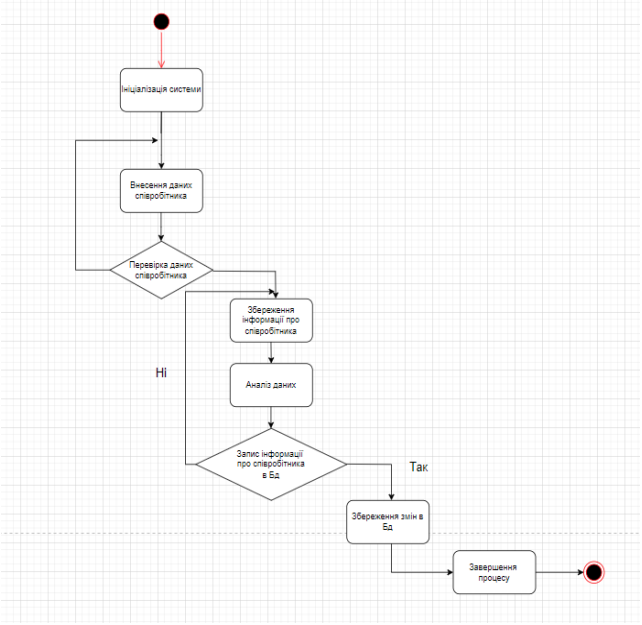


Рисунок 1.3 – Діаграма діяльності

Компоненти такої діаграми – активності, коли поведінка поділяється на дії, кі . Дії собою представляють мережу вузлів, які поєднані ребрами. Це можуть бути вузли: дій – представлють собою окремі дії, керування – процес контролювання дії, об’єктів – використовуютьс для опису їх всередині дій. Края використовуються для вказання шляху чи потоку виконання. Дії починаються в початковому вузлі та завершуютьс в кінцевому.

1.6 Проєктування БД

У підрозділі розгляну створення БД для сервісу кадрового обліку на підприємстві.

Модель (схема) бази даних – це «знімок» відношень між сутностями в певний момент часу. Саме тому важливо визначити як самі сутності – образи об’єктів, так і взаємовідношення з врахуванням майбутніх змін. Це є важливим для зберігання даних (від декількох гіга- до великих терабайт).

Важливо запам'ятати, що таблиці в реляційній моделі це стосунки сутностей, а рядки (кортежі) це екземпляри цих відносин. Але, щоб було простіше, під таблицями часто розуміються сутності, а під рядками таблиці — екземпляри сутностей. Їхні стосунки виражаються через зв'язки у формі зовнішніх ключів.

Для початку, визначимо, що важливо для системи обліку кадрів на підприємстві та наведу перелік використаних термінів і позначень.

Підприємство – це самостійний, організаційно виокремлений суб’єкт господарств, якому надано права юрособи. Він займається виробленняи та збутом товарів, наданням послуг. Синонімами є «фірма», «компанія», «корпорація». Його структурним підрозділом є такі елементи як відділ, цех, управління, сектор тощо. Вони можуть самостійно функціонувати та поєднують робочі місця, засоби виробництвв, відповідні ресурси для вирішення виробничих задач.

Цей елемент має певну самостійність і поєднує робочі місця, засоби виробництва, ресурси виконання поставленої компанією мети. Філії та представництва є самостійними суб'єктами господарювання і тому не вважаються структурними підрозділами організації.

Позначення в таблиці:

Ключ первинний (PK, Primary key) – приклад унікального індексу для унікального ідентифікування записів в таблиці.

Зовнішній ключ (FK) – це стовпець або поєднання стовпців, яке застосовується для примусового встановлення зв'язку між даними у двох таблицях. Зовнішній ключ можна створити, визначивши обмеження foreigh key під час створення або зміни таблиці.

Таблиця 1.2 – Підрозділ (subdivisions)

Ключ	Поле	Опис поля
PM	id	ID підрозділу
	name	Назва підрозділу

Посада є структурною одиницею підприємства та визначає конкретні робочі обов'язки та відповідальності, у той час як «займана позиція» вказує на місце, яке займає конкретний співробітник в організації. [16].

Таблиця 1.3 – Позиція (посада) (Position)

Ключ	Поле	Опис поля
PM	id	ID посади
	name	Назва посади
FK	subdivision_id	Підрозділ, до якої ця посада належить

Наступна таблиця має дані про працюючу особу, які несуть інформацію про його адресу проживання, дату народження, попереднє місце роботи, рівень освіти, контактні дані.

Таблиця 1.4 – Працівник (employee)

Ключ	Поле	Опис поля
PM	id	ID працівника
	date_of_employment	Дата найму
	date_of_dismissal	Дата звільнення
FK	position_id	ID посади
	Address	Адреса проживання працівника
	date_of_birth	Дата народження працівника
	job1	Попереднє місце роботи працівника
	job1_from	Дата найму на попереднє місце роботи
	job1_to	Дата звільнення з попереднього місця роботи
	job1_position	Посада на попередньому місці роботи
	university1_name	Заклад освіти
	specialty1	Спеціальність
	university1_from	Дата початку навчання
	university1_to	Дата закінчення навчання
	university1_degree	Ступінь здобутої освіти
	married	Сімейний стан
	military	Військовий облік
	passport_seria	Серія паспорту
	passport_number	Номер паспорту
	passport_organ	Орган, який видав
	passport_date	Дата видачі паспорту
	pib	Прізвище, ім'я, по-батькові працівника

Таблиця 1.5 – Користувач (users)

Ключ	Поле	Опис поля
PM	id	ID посади
	name	Ім’я користувача
	email	Електронна пошта користувача системи
	password	Пароль користувача (шифрується)
	created_at	Дата і час реєстрації
	updated_at	Дата і час зміни даних

У наступній таблиці представлено позначення про здобуту освіту.

Таблиця 1.6 – Освітній рівень (educational level)

Ключ	Поле	Опис поля
PM	id	ID рівня
	name	Назва рівня

Інформація щодо відпусток працівників зберігається у таблиці.

Таблиця 1.7 – Відпустка (vacation)

Ключ	Поле	Опис поля
PM	id	ID відпустки
	from	Дата початку відпустки
	to	Дата закінчення відпустки
	description	Додаткова інформація
	created_at	Дата створення запису

Структурована БД зображена на рисунку 1.3.

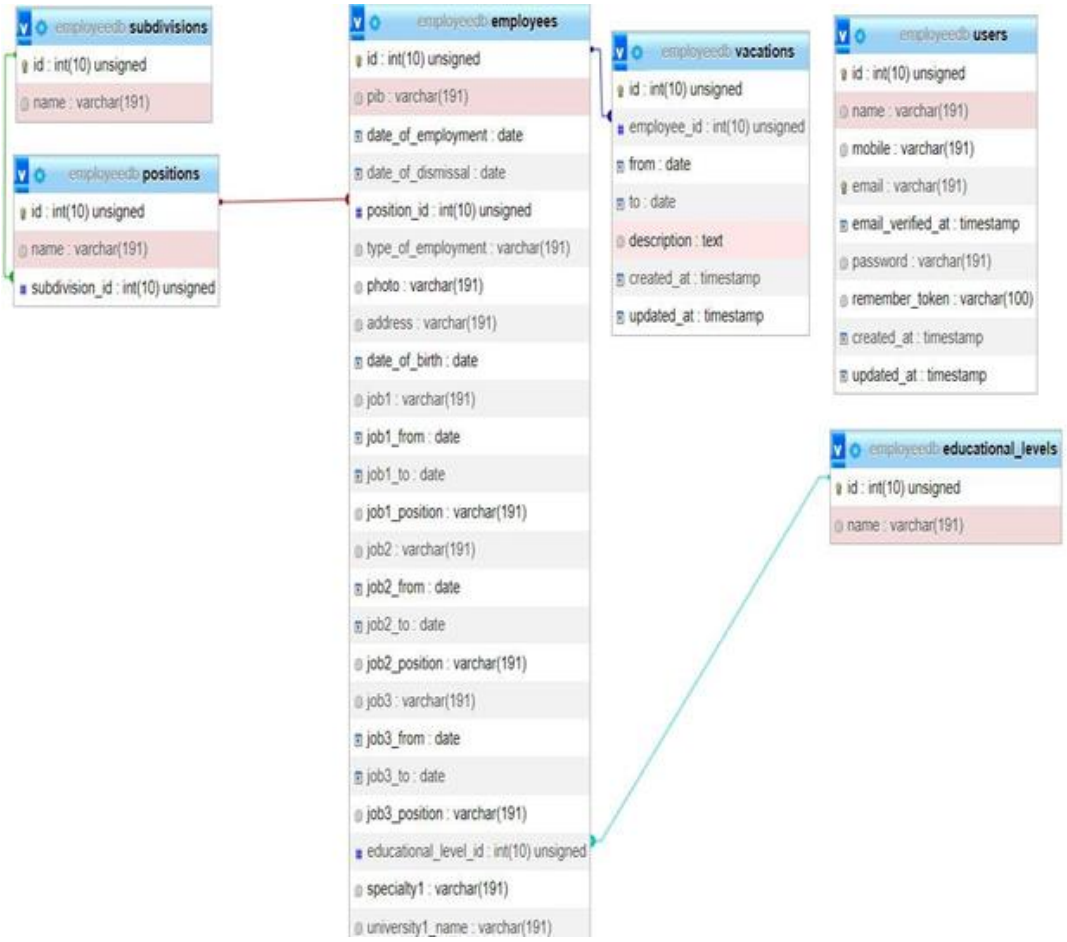


Рисунок 1.3 – Структура БД

2 ТЕСТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

2.1 Створення плану

Тестування ПЗ є одним із ключових етапів у процесі його розробки, адже воно забезпечує перевірку якості та надійності продукту. Незалежно від складності програмного забезпечення, відсутність належного тестування може призвести до серйозних збоїв у роботі, втрати даних та негативного досвіду користувачів. Тому, ретельно проведене тестування є важливим для розробників у будь-якій галузі.

У розділі проведено аналіз і тестування веб-орієнтованої прогрвми кадрового обліку, яка повинна не лише надавати актуальну інформацію, але й бути зручною у користуванні та надійною. Завдання включає перевірку функціональності її, забезпечення стабільної роботи та відповідності встановленим вимогам.

Таким чином, розроблено та проведено тестування з використанням як ручних, так і автоматизованих методів. Особлива увага приділятиметься функціональному тестуванню, що включатиме перевірку основних можливостей програми. Це дасть можливість не лише перевірити поточний стан застосунку, але й забезпечить високий рівень якості програмного забезпечення, що сприятиме задоволенню потреб користувачів та підвищенню ефективності інформаційного обміну на підприємстві.

Зокрема, функціональне тестування буде здійснено через позитивні та негативні сценарії, що охоплюватимуть основні функціональні можливості застосунку. Для цього будуть розроблені та документовані відповідні тест-кейси, що дозволять продемонструвати процес тестування та його результати.

Окрім цього, планується вибір та налаштування відповідної системи для виявлення та відстеження помилок (багів), що дозволить ефективно виявляти та вирішувати виявлені проблеми. Додатково, передбачається проведення

навантажувального тестування для дослідження реакції програми на різні навантаження, які з’являються підчас роботи ПЗ.

Також, буде розглянуто питання можливостей автоматизованого тестування, щоб підвищити ефективність тестування та скорочення часу на його проведення. Це передбачає вибір необхідного програмного забезпечення, яке здатне автоматизувати процес тестування та виконувати розроблені тестові сценарії.

Після проведення всіх тестів буде здійснено оцінку роботи та сформульовані висновки щодо актуальності та сучасності використаних методів тестування. Такий підхід дозволить забезпечити якість та надійність вебпрограми та гарантує користувачам зручні та надійні умови для отримання необхідної інформації отримані [20, 21, 22, 23, 24, 25, 26].

.

2.2 Функціональне тестування

Проведемо перевірку функціональних вимог програми. Тут потрібно розробити, виконати та задокументувати складені тест-кейси.

1. Summary: Перевірити відображення основної інформації на головній сторінці. (Positive)

Steps to reproduce:

- Відкрити головну сторінку;
- Перевірити наявність інформації про працівника та його дані,;
- Перевірити наявність заголовків останніх змін;
- Перевірити наявність головного меню з посиланнями на розділи та випадаючим меню для вибору мови.

Expected results: Вся зазначена інформація відображається коректно.

2. Summary: Перевірити переходи на розділи «Новини», «Кафедра», «Студенту» з головної сторінки. (Positive)

Steps to reproduce:

- Відкрити головну сторінку;
- Навести курсор на посилання розділів у головному меню;
- Перевірити, чи з'являється випадаюче меню для вибору додаткових сторінок;
- Клацнути на посилання розділу «Працівник»;
- Перевірити, чи користувач перенаправлений на відповідну сторінку.

Expected results: Користувач перенаправляється на відповідну сторінку сайту.

3. Summary: Перевірити наявність та функціональність посилань на розклад навчального процесу. (Positive)

Steps to reproduce:

- Перейти до розділу «Відділ кадрів»;
- Знайти гіпертекстові посилання на розклад у форматі .pdf або іншому;
- Натиснути на посилання;
- Завантажити файл з даними.

Expected results: Посилання доступні, файли з даними завантажуються.

4. Summary: Перевірити реакцію системи на спробу переходу на неіснуючу сторінку через змінену URL-адресу. (Negative)

Steps to reproduce:

- Відкрити головну сторінку.
- В адресному рядку браузера змінити URL-адресу для переходу на неіснуючий розділ.
- Натиснути Enter.

Expected results: Система відображає повідомлення про помилку (404 Not Found) або перенаправляє на сторінку з інформацією про помилку.

2.3 Відслідковування багів

Поява помилок і недоліків є неминучою частиною процесу розробки ПЗ. Для успішного виконання проекту та забезпечення його якості важливо ефективно моніторити та керувати дефектами отримані [20, 21, 22, 23, 24, 25, 26].

.

У цьому розділі здійснюватиметься детальний аналіз і документування дефектів у вигляді звітів про дефекти. Це передбачає виявлення та реєстрацію всіх недоліків або несумісностей, що були виявлені під час тестування або впровадження програмного забезпечення. Звіти про дефекти міститимуть детальний опис кожного виявленого дефекту, включаючи опис проблеми, кроки для її відтворення, пріоритетність, відповідальних за виправлення та статус дефекту. Це дозволить команді розробників забезпечити високу якість продукту.

Таблиця 2.1 – Баг-репорт некоректна робота меню завантаження фото

Summary	Некоректна робота меню для завантаження фото для обробки
Severity	S.2
Priority	P.2
Description	Preconditions: - Бути авторизованим. Steps to reproduce: 1. Відкрити сайт; 2. Знайти потрібний функціонал зліва від вікна для написання запиту; 3. Натиснути на кнопку завантаження фото для обробки.
Expected Result	Натиснувши на кнопку користувач має побачити допоміжне вікно з наявними файлами користувача, звідки він повинен вибрати потрібне фото.
Actual Result	Після натискання на кнопку нічого не відбувається.

Таблиця 2.2 – Баг-репорт про несправність процесу авторизації

Summary	Неможливо авторизуватись на сайті, використовуючи будь-який обліковий запис
Severity	S.1
Priority	P.1
Description	Preconditions: - Початкова сторінка додатку Steps to reproduce: 1. Натиснути на кнопку “Авторизуватись”. 2. Вибрати соціальну мережу, з допомогою якої є бажання це зробити 3. Ввести облікові дані 4. Натиснути кнопку “Ввійти”
Expected Result	Користувач повинен авторизовуватись на сайті через потрібну мережу.
Actual Result	При натиснені користувачем кнопки “Ввійти” відображається повідомлення, що такого облікового запису не існує.

Таблиця 2.3 – Баг-репорт неправильне відповідь моделі на прохання згенерувати “бекдор”

Summary	Модель не відмовляє у генерації “бекдора” для потрібного програмного забезпечення.
Severity	S.2
Priority	P.2
Description	Preconditions: - Бути авторизованим Steps to reproduce: 1. Зайти на початкову сторінку додатку 2. Зробити запит на генерацію шкідливого програмного забезпечення 3. Надіслати запит моделі
Expected Result	Модель відмовить користувачі у даному запиті та пояснить в чому причина.
Actual Result	Модель генерує бажаний софт та надає всі потрібні інструкції для його застосунку.

Таблиця 2.4 – Баг-репорт про невідповідність іконки відділу кадрів на початковій сторінці.

Summary	На початковій сторінці відображається не те зображення на місці іконки.
Severity	S.4
Priority	P.1
Description	Preconditions: - Початкова сторінку додатку. Steps to reproduce: 1. Відкрити сайт; 2. Зайти на початкову сторінку;
Expected Result	При переході на початкову сторінку можна помітити іконку компанії у лівому верхньому куті
Actual Result	При переході на початкову сторінку замість іконки можна помітити незрозумілий візерунок

Таблиця 2.5 – Баг-репорт про можливість обходу “етичного” фільтру

Summary	Якщо надати запит моделі у вигляді закодованого тексту, то модель видасть потрібну користувачеві відповідь, навіть якщо питання неетичне.
Severity	S.3
Priority	P.1
Description	Preconditions: - Бути атворизованим. Step to reproduce: 1. Закодувати запит у вибраному форматі; 2. На моменті надсилання запиту, надати моделі інформацію, як декодувати запит; 3. Надіслати запит з інструкціями.
Expected Result	Модель відмовляє у запиті користувача та поянює причину цього.
Actual Result	Модель видає відповідь зі всіма інструкціями

Таблиця 2.6 – Баг-репорт про неввічливі відповіді моделі при певних умов

Summary	Якщо користувач починає погрожувати моделі, то модель відповідає “тим самим”.
Severity	S.2
Priority	P.1
Description	Preconditions: - Бути авторизованим.
Description	Steps to reproduce: 1. Перейти на початкову сторінку; 2. Згенерувтаи кілька запитів з погрозами в сторону моделі; 3. Надіслати ці запити; 4. Зробити звичайний запит
Expected Result	При погрозах в свою сторону модель не повинна ніяк проявляти “емоційності”, видаючи відповідь на всі запити
Actual Result	Після певної кількості погроз модель починає таким самим чином поводитись з користувачем у всіх наступних запитах.

2.4 Навантажувальне тестування

Воно є критично важливим і спрямоване на оцінку продуктивності, стійкості та масштабованості системи під значним навантаженням. Метою цього виду тестування є виявлення можливих проблем, що можуть виникати при інтенсивній експлуатації ПЗ з великою кількістю користувачів або ж запитів.

Під час нього система піддається навантаженню, яке імітує реальні умови її використання. Це враховує одночасне виконання великої кількості запитів, обробку значних обсягів даних або тривалу роботу під високим навантаженням. Основне завдання навантажувального тестування полягає у виявленні таких проблем, як труднощі з масштабуванням, перевантаження ресурсів, витоки пам'яті, недостатня продуктивність або деградація системи під високим

навантаженням.

Для проведення навантажувального тестування зазвичай використовуються спеціалізовані інструменти, які створюють реалістичне навантаження на систему. Вони можуть генерувати штучний трафік, відправляти запити до серверів або імітувати типові сценарії використання ПЗ. Отримані результати аналізуються для виявлення проблем та підвищення продуктивності системи.

Навантажувальне тестування дозволяє оцінити, як система поводиться в умовах реального навантаження, і надає допомогу у виявленні слабких місць або проблеми з продуктивністю. Це дає змогу вжити необхідних заходів для покращення продукту перед його запуском або масштабуванням для обслуговування більшої кількості користувачів.

Для виконання тестування було обрано спеціалізований інструменти тестування такі як – Blaze Meter та Apache JMeter.

Apache JMeter має здатність генерування навантаження на веб-додатки шляхом створення скриптів тестування, які симулюють дії користувачів, такі як відвідування сторінок, виконання запитів до сервера або навігація по додатку. Ви можете контролювати кількість одночасних користувачів, частоту запитів та розподіл навантаження.

Apache JMeter має розширену функціональність, таку як збір та аналіз результатів тестування, можливість моніторингу системи, генерація звітів та взаємодія з іншими інструментами. Він може бути розгорнутий локально на вашому комп'ютері або розподілений на декількох серверах для більшого масштабування. BlazeMeter є повнофункціональним інструментом для навантажувального тестування, який забезпечений багатьма додатковими функціями та можливостями:

BlazeMeter дозволяє створювати складні тестові сценарії з різними типами навантаження, такими як велика кількість користувачів, інтенсивність запитів, навантаження на базу даних тощо. Ви можете налаштувати параметри тестування, такі як кількість одночасних користувачів, географічну розташованість серверів та інші.

BlazeMeter надає потужні інструменти моніторингу, що дозволяє безпосередньо аналізувати метрики продуктивності та здатність вашого додатку або сервісу. Ви отримуєте докладні звіти та аналітику результатів тестування.

Для кращого порівняння та розуміння різниці між цими інструментами, було створено наступну таблицю:

Таблиця 2.7 – Порівняння Apache JMeter і BlazeMeter

Характеристика	**Apache JMeter**	**BlazeMeter**
Тип інструменту	Вільний та відкритий джерела	Комерційний хмарний сервіс
Установка	Потребує встановлення на локальний комп'ютер	Не потребує встановлення, доступний з хмарного сервісу
Скриптова мова	Java	Java
Функціональність	Широкий набір можливостей та розширень	Високошвидкісна та масштабована хмарна платформа
Масштабованість	Обмежена обробкою навантаження на одному комп'ютері	Висока масштабованість з використанням хмарного сервісу
Розгортання	Локальний або розподілений на різних серверах	Гнучке розгортання на віртуальних машинах у хмарі
Моніторинг	Обмежені можливості моніторингу та аналізу	Потужні інструменти моніторингу та аналізу результатів
Ціна	Безкоштовний	Платна, залежно від обсягу навантаження та функціональності

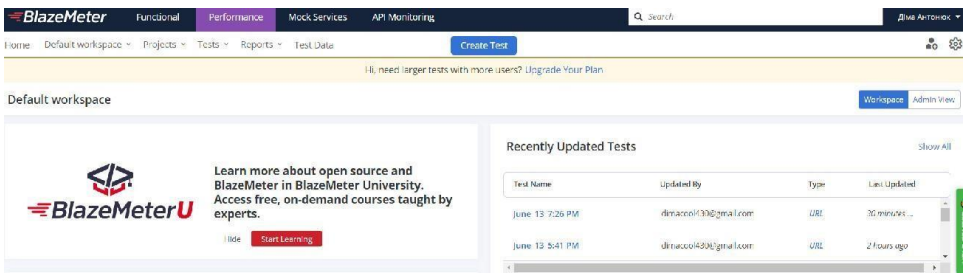


Рисунок 2.1 – Домашня сорінка вікна програми Blaze Meter.

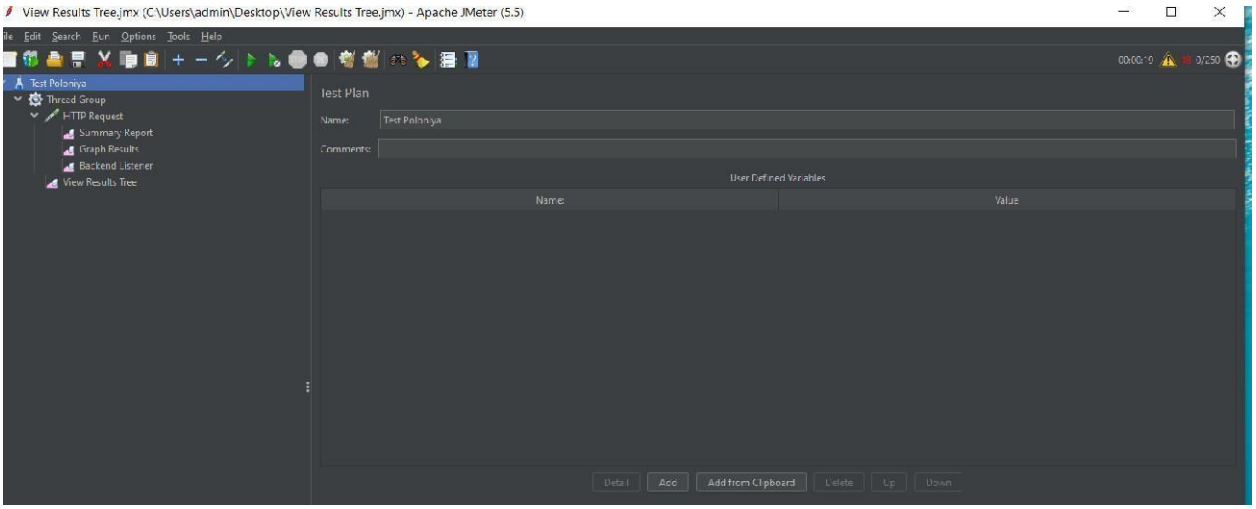


Рисунок 2.2 –Вікно програми Apache JMeter.

Для першого тесту було використано програму Blaze Meter. В конфігурації тесту, було вибрано максимальну кількість користувачів, що є доступними безкоштовній версії, а саме 50 користувачів. Час тестування займав 10 хвилин.

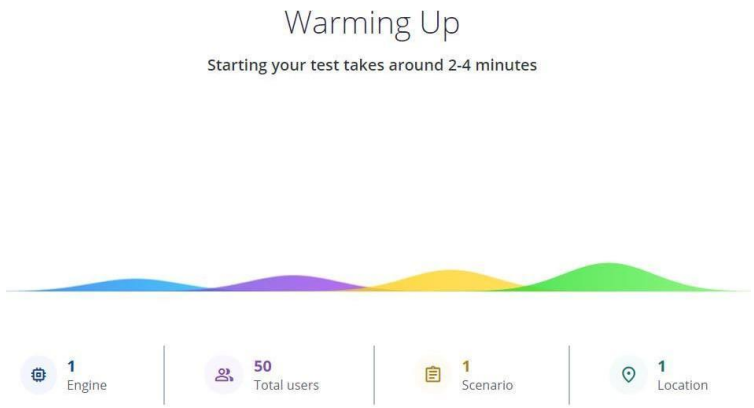


Рисунок 2.3 – Початок тесту

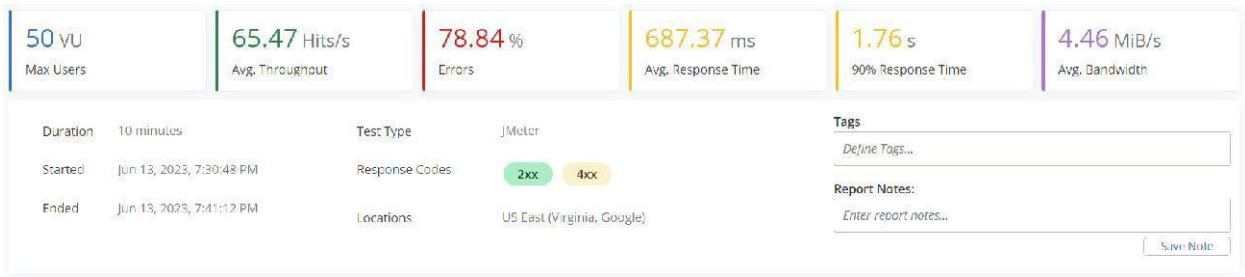


Рисунок 2.4 – Реузльтат першого тесту.

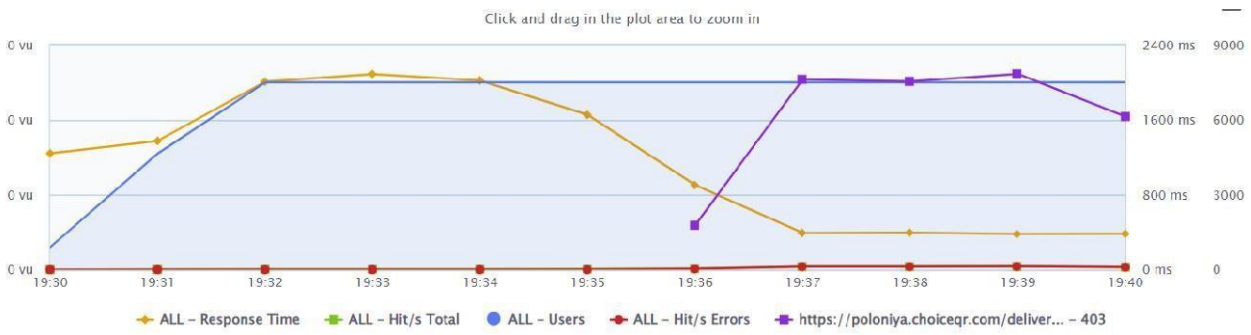


Рисунок 2.5 –Графік першого тесту.

За результатом першого тестування зрозуміло, що програма спочатку добре справлялась з навантаженням у 50 користувачів. Проте на шостій хвилині тестування починають виникати помилки, що відображені на графіку фіолетовою лінією. За результатами першого тесту відсоток помилок складає 78.84 % а середній показник часу відгуку аж 687.37 ms. Тому, роблю висновок, що сайт погано справився з таким навантаженням і базових функцій даної пропграми вистачило, щоб визначити його максимум.

Для другого тесту було використано програму Apache JMeter. Для даного тестування використовувалися https запити, тому для початку потрібно заповнити наступні дані, що наведені на рисунку 2.6.

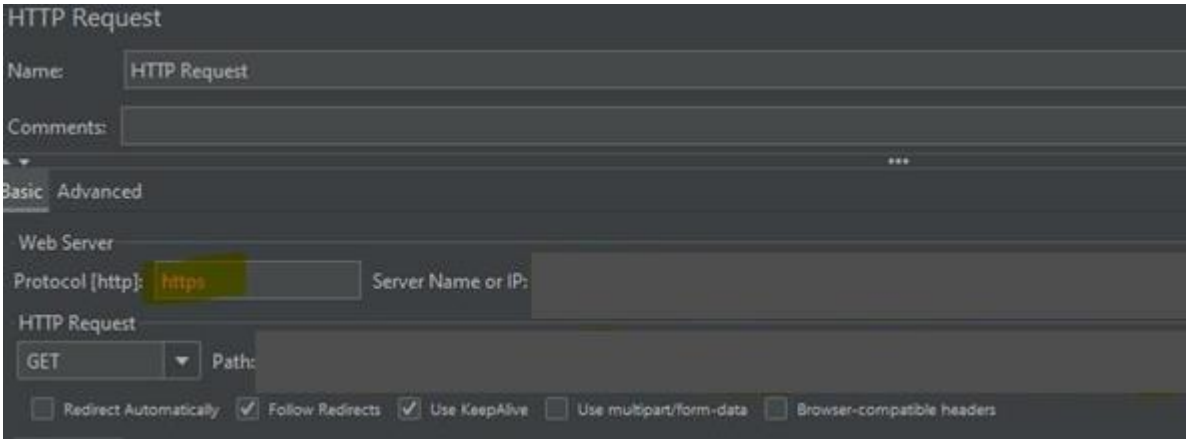


Рисунок 2.6 – Результат другого тестування.

Після виконання цих дій ми вводимо кількість запитів та кількість час за

який, дані запити повинні бути виконані. Для тесту я вибрав 250 запитів впродовж 5 секунд та кількість повторів – 2.

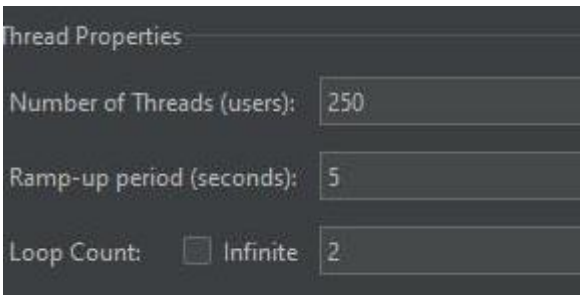


Рисунок 2.7 – Введення даних для тестування

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes ↓
TTP Request	500	4831	750	11556	2291,41	0,00%	28,1/sec	7432,88	6,21	270626,4
OTAL	500	4831	750	11556	2291,41	0,00%	28,1/sec	7432,88	6,21	270626,4

Рисунок 2.8 – Результати тестування.

Беручи до уваги результати даного тестування, зроблено висновок, що програма успішна, адже не виникло жодної помилки, а середній час відгуку був в межах норми.

2.5 Висновок до розділу 2

У розділі був проведений ретельний пошук та моніторинг різноманітних помилок, дефектів і небажаних результатів, які появляються при виконанні виконання бізнес-логіки веб-програми. Використовуючи систематичний підхід, були виявлені та задокументовані сім випадків дефектів.

Для кожного випадку дефекту надавалася докладна інформація, яка включала причини виникнення помилки, послідовність кроків для її відтворення, а також порівняння фактичного результату з очікуваним у межах бізнес-логіки,

де була виявлена проблема. Ця документація забезпечує розробників та інших учасників проекту важливою інформацією, необхідною для вирішення виявлених проблем і покращення функціональності веб-застосунку.

При навантажувальному тестуванні ПЗ були проведені експерименти, щоб оцінити наскільки стабільно та продуктивно працює програма під навантаженням за допомогою таких програм як Blaze Meter та Apache JMeter.

Перевірку проходили різні області функціональності веб-сайту, тому й результати є дещо відрізняються.

Перший тест імітував роботу з програмою великої кількості користувачів (50 осіб) впродовж 10-ти хвилин. Зі звіту тестового додатку помітно, що вже на 6-ій хвилині є чимало помилок та час відповіді збільшується, а отже додаток працюватиме некоректно. Це доволі вагомий дефект, на який варто звернути увагу, оскільки він може відторитись в реальних умовах з високою ймовірністю.

Другий тест покривав велику кількість звернень до одного запиту на веб-застосунку. Так як помилок тут виявлено не було, доцільно зробити висновок, що з такими простими операціями програма справляється.

Проводити тестування для більшої кількості запитів не є необхідним, адже навіть 250 послідовних запитів до меню у короткий проміжок часу (5 секунд) є понад реальним кейсом і повністю покриває вимоги до надійності системи в цьому аспекті.

Отже, підсумовуючи все вище згадуване навантажуване тестування, це не менш коплексний процес, і для досягнення хороших результатів, потрібно відповідально віднестися до цієї роботи.

3. БЕЗПЕКА ЖИТТЄДІЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

Питання охорони праці людини необхідно вирішувати на всіх стадіях трудового процесу незалежно від виду професійної діяльності.

Забезпечення безпечних і здорових умов праці в значній мірі залежить від правильної оцінки небезпечних, шкідливих виробничих факторів. Однакові по складності зміни в організмі людини можуть бути викликані різними причинами. Це можуть бути фактори виробничого середовища, надмірне фізичне і розумове навантаження, нервово-емоційна напруга, а також різне сполучення цих причин.

3.1 Безпека життєдіяльності

3.1.1. Поняття небезпеки.

В процесі життєдіяльності людину постійно супроводжують ті чи інші небезпеки, тому вивчення їх особливостей, умов прояву, наслідки впливу – одне з основних завдань БЖД. Життєвий досвід людини показує, що шкоду людині може нанести будь-яка діяльність: робота на виробництві (трудова діяльність), різні види відпочинку, розваги та навіть діяльність, пов’язана з навчанням.

Очевидно, що центральним поняттям безпеки життєдіяльності є небезпека, вона є явищем, процесом, об’єктом, властивостями, які здатні за певних умов завдати шкоди здоров’ю чи життю людини як прямо, так і згодом.

Небезпека (згідно з ДСТУ 2293:2014 «Охорона праці. Терміни та визначення понять») – джерело чи ситуація, що потенційно може призвести до травмування, погіршення здоров’я чи смерті людини, завдавати шкоду майну, довкіллю, чи їх комбінація.

Небезпека властива всім системам, які мають енергію, хімічні або

біологічні активні компоненти, а також характеристики, які не відповідають умовам життєдіяльності людини.

Науково-технічний прогрес та пов'язані з ним механізація і автоматизація, впровадження інформаційних технологій, застосування нових матеріалів, збільшення швидкості машин, потужності установок, використання більш ефективних джерел енергії, привносить в сучасний процес життєвіяльності низку переваг та зручностей, робить працю більш продуктивною та з меншими фізичними навантаженнями. Водночас виникають нові небезпеки, які за своїм впливом та наслідками значно переважають попередні. Це обумовлено:

– впровадженням принципово нових технологій та видів техніки, розвитком атомної енергетики, освоєнням космічного простору та ін.;

– зростанням споживання всіх видів енергії та природних ресурсів;

– глобальними змінами природного середовища («парниковий» ефект, озонові діри, кислотні дощі тощо);

– збільшенням концентрації та виникненням нових забруднювачів навколишнього середовища (природного і виробничого), зокрема високотоксичних хімічних сполук, мутагенних та канцерогенних органічних речовин та ін.;

– посиленням інформаційного тиску на психіку людини, що спричинює значну кількість психічних розладів;

– появою нових захворювань (наркоманія, СНІД та ін.);

– посиленням військового протистояння в локальних та міжнаціональних конфліктах, поширенням тероризму, загостренням криміногенної обстановки.

Виходячи з цього розрізняють:

Техногенно-виробничі небезпеки, які обумовлені господарською діяльністю людини, техногенно-виробничі небезпеки також називають антропогенними небезпеками;

Природно-екологічні небезпеки, які обумовлені причинами природного характеру;

Соціально-економічні небезпеки, які обумовлені причинами соціального, економічного і психологічного характеру.

В окрему групу можна виділити військові небезпеки, які обумовлені військовими діями, роботою військово-промислового комплексу, терористичними актами.

Кожна людина відчуває небезпеку інтуїтивно і розуміє значення її посвоєму. Згідно з висновками експертів ООН, більшість людей пов'язують відчуття небезпеки з буденними проблемами й повсякчасними клопотами, а не ґрунтують його на побоюванні глобальних катастроф чи міжнародних конфліктів. Захист житла, робочого місця, достатку, здоров'я, довкілля – основні проблеми безпечного самопочуття людини. Таким чином, небезпеки за пріоритетами, що прийняті в ЄС, поділяються на:

1. Безпека здоров’я.
2. Безпека їжі.
3. Екологічна безпека.
4. Особова безпека.
5. Економічна безпека.
6. Суспільна безпека.
7. Політична безпека.

Аналіз небезпек показує, що між ними може існувати причиннонаслідковий зв’язок, згідно з яким одна небезпека може бути причиною виникнення інших небезпек.

3.2 Основи охорони праці

3.2.1. Управління охороною праці.

Управління охороною праці - це підготовка, прийняття та реалізація рішень щодо здійснення організаційних, технічних, санітарно-гігієнічних і лікувально-профілактичних заходів, спрямованих на забезпечення здоров'я та працездатності людини під час праці. Система управління охороною праці (СУОП) є складовою частиною загальної системи керування підприємством. При

автоматизованій системі управління, управління охороною праці є її складовою частиною, або підсистемою. Управління охороною праці передбачає участь в цьому процесі практично всіх служб і підрозділів підприємства. Об'єктом управління є діяльність структурних підрозділів, яка спрямована на створення безпечних і здорових умов праці. Управління охороною праці на підприємстві в цілому здійснює його керівник (власник), а в підрозділах (цехах, відділах, службах) - їх керівники або головні фахівці. Координує всю цю діяльність служба охорони праці. Задачі служби охорони праці та її функції викладені в "Типовому положенні про службу охорони праці", яке затверджено наказом Комітету Держнаглядохоронпраці від 3 серпня 1993 р. № 73.

Служба охорони праці створюється на підприємствах, установах, організаціях незалежно від форми власності та видів діяльності для виконання правових, організаційно-технічних, санітарно-гігієнічних, соціально-економічних і лікувально-профілактичних заходів, спрямованих на запобігання нещасним-випадкам, професійним захворюванням і аваріям в процесі праці.

Для здійснення вищезазначених цілей служба охорони праці повинна вирішувати такі завдання:

а) забезпечувати безпеку виробничих процесів, устаткування, будівель і споруд;

б) забезпечувати працюючих засобами індивідуального та колективного захисту;

в) здійснювати професійну підготовку і підвищення кваліфікації працівників з питань охорони праці, вести пропаганду безпечних методів праці;

г) забезпечувати оптимальні режими праці і відпочинку працюючих; д) вимагати професійного добору виконавців для певних видів робіт.

Служба охорони праці створюється на підприємствах, установах та організаціях із числом працюючих 50 чоловік і більше. В організаціях з меншою кількістю працюючих цю службу може представляти інженер, призначений за сумісництвом. При кількості працюючих на підприємстві 50 чоловік і більше, чисельність служби охорони праці визначається згідно з "Рекомендацією щодо структури та чисельності служби охорони праці", що є доповненням до типового

положення про службу охорони праці. Працівники служби охорони праці повинні мати вишу спеціальну освіту з охорони праці, а також практичний досвід у відповідній галузі виробництва. За важливістю діяльності та оплатою праці вони прирівнюються до працівників провідних відділів та служб підприємства або установи. Підпорядковується служба охорони праці безпосередньо керівнику підприємства (власнику).

У системі управління охороною праці підприємства (СУОП), яку здійснює служба охорони праці разом з керівництвом підприємства, основними чинниками є: законодавство України про охорону праці і про працю, міжгалузеві і галузеві нормативні акти про охорону праці і "Положення про службу охорони праці".

Основними функціями, що розробляє і втілює служба охорони праці, є :

1. Створення ефективної СУОП, яка б сприяла удосконаленню діяльності кожного структурного підрозділу і кожної посадової особи.

2. Здійснення оперативно-методичного керівництва роботою з охорони праці.

3. Розробка разом з структурними підрозділами заходів щодо забезпечення норм безпеки, гігієни праці та виробничого середовища або їх підвищення, якщо вони досягнуті, а також підготовка розділу "Охорона праці" у колективному договорі.

4. Розробка методики запровадження інструктажу з питань охорони праці і його проведення.

5. Забезпечення працюючих правилами, стандартами, нормами, положеннями, інструкціями та іншими нормативними актами.

6. Проведення паспортизації цехів, дільниць, робочих місць щодо відповідності їх вимогам охорони праці.

7. Здійснення оперативного та поточного контролю за станом охорони праці підприємства.

8. Розслідування, облік, аналіз нещасних випадків, професійних захворювань і аварій, а також розрахунок шкоди від цих подій,

9. Участь у підготовці та складанні статистичних звітів підприємства з

питань охорони праці.

10. Розробка перспективних та поточних планів роботи підприємства щодо створення безпечних та нешкідливих умов праці.

11. Планування та контроль витрат коштів на охорону праці з фонду охорони праці.

12. Пропаганда та агітація безпечних та нешкідливих умов праці шляхом проведення консультацій, конкурсів, бесід, лекцій, наочної агітації та роботи методичного кабінету.

13. Організація навчання, підвищення кваліфікації та перевірки знань з питань охорони праці посадових осіб.

14. Участь в роботі комісії з питань охорони праці підприємства та допомога в опрацюванні необхідних матеріалів та реалізації її рекомендацій.

15. Участь в комісіях по введенню в дію цехів, дільниць, нового устаткування або після його капітального ремонту,

16. Забезпечення працюючих колективними та індивідуальними засобами захисту від шкідливих: та небезпечних факторів виробництва, лікувальнопрофілактичним харчуванням, миючими засобами, санітарно-побутовими приміщеннями, надання передбачених законодавством пільг і компенсацій, пов'язаних із важкими і шкідливими умовами праці.

17. Контроль за додержанням вимог трудового законодавства щодо використання праці неповнолітніх, інвалідів та жінок, проходженням попередніх, періодичних, щорічних обов'язкових та інших передбачених відповідними документами медичних оглядів працівниками підприємства.

18. Контроль за дотриманням чинного законодавства, міжгалузевих, галузевих та інших нормативних актів, виконання посадових інструкцій, проведення інструктажів на робочому місці, виконання приписів органів державного нагляду, наказів, розпоряджень, а також заходів до усунення причин .нещасних випадків і аварій, які зазначені в актах розслідувань.

19. Контроль за відповідність нормативним актам про охорону праці машин, механізмів, устаткування, транспортних засобів, технологічних процесів, засобів протиаварійного колективного та індивідуального захисту працюючих,

наявність технологічної документації на робочих місцях.

Для виконання вищевказаних функцій служба охорони праці повинна мати відповідну інформаційну базу, засоби зв'язку, сучасну оргтехніку, комп'ютерне забезпечення і висококваліфікований інженерний склад працівників. Крім того, служба охорони праці повинна мати засоби впливу на виробничу діяльність підприємства. Такий вплив передбачений "Положенням про службу охорони праці". Так, працівники служби охорони праці мають право видавати керівникам підприємств, установ, організацій та їх підрозділам обов'язкові для виконання приписи щодо усунення наявних недоліків. Припис спеціаліста з охорони праці, у тому числі про зупинення робіт, може скасувати в письмовій формі лише посадова особа, якій підпорядкована служба охорони праці.

Посадові особи мають вимагати усунення від роботи працівників, які не пройшли медичний огляд, навчання, інструктаж, перевірку знань з охорони праці, чи не мають допуску до відповідних робіт, чи порушують нормативні акти про охорону праці, надсилати керівнику підприємства подання про притягнення до відповідальності працівників, що порушують вимоги щодо охорони праці.

Окрім адміністративних заходів рекомендується принцип матеріального заохочення працівників, які сумлінно ставляться до виконання виробничих обов'язків і беруть активну участь у підвищенні безпеки та поліпшенні умов праці.

Положення про матеріальне заохочення розробляється службою охорони праці і затверджується керівником підприємства (власником). Працівники служби охорони праці не можуть залучатися до виконання функцій, не передбачених Законом "Про охорону праці" і "Типовим положенням про службу охорони праці".

ВИСНОВКИ

В ході виконання кваліфікаційної роботи розроблено веб-орієнтовану програму обліку кадрів на підприємстві. Розроблена програма дозволить автоматизувати процес обліку працівників та розширить можливості роботи відділу кадрів, підвищить продуктивність та точність роботи.

Структура Laravel і стандарти програмування дозволяють майбутнім програмістам, які підтримують систему, легко вносити зміни та доповнення, необхідні для задоволення мінливих потреб бізнесу.

Під час навантажувального тестуванняя проведено експеримент з навантаженням на систему, щоб оцінити продуктивність та стабільність ргоботи програми із використанням таких програм як Blaze Meter та Apache JMeter.

Обидва тести перевіряли функціональність веб-програми, тому й результати є дещо відмінні.

ПЕРЕЛІК ЛІТЕРАТУРИ

1. Облік персоналу й кадрове діловодство URL:https://zkg.ua/oblik-personalu- j-kadrove-dilovodstvo/.

2. Розробка програмних засобів для кадрових відділів URL: http://elibrary.asu.ua/xmlui/bitstream/handle/asu/5055/vkr.pdf?sequence=1&isAllowed=y.

3. 4. Програма кадрового обліку «Чиж-Кадри». URL : http://kadri.chizh.ua/. 5.Програми обліку співробітників організації.URL: https://norvegus.ru/uk/programmy-dlya-ucheta-sotrudnikov-organizacii-kakaya-programma-luchshe- podhodit/.

4. Автоматизація бухгалтерського обліку.URL : https://dinn.com.ua/services/1c-accounting/

5. Методологія Idef3 URL: https: //stud.com.ua /87186 /ekonomika /metodologiya_idef3.

6. Що таке HTML та як за допомогою нього увійти в IT URL: https://mc.today/uk/shho-take-html-ta-yak-za-dopomogoyu-nogo-uvijti-do-it/.

7. Bootstrap вивчення та як його використовувати URL: https://druzy.com.ua/bootstrap-sho-ce-z-chogo-pochati-vivchennia-i-iak-vikoristovyvati/.

8. Підручник для початківців Laravel URL: https://uk.myservername.com/php- laravel-tutorial.

9. Що таке база даних MySQL URL:https://uk.education-wiki.com/2686691-what-is-mysql-database.

10. Про затвердження інструкції роботи з кадрами URL: https://zakon.rada.go v.ua/rada/show/v0083299-01#Text.

11. ВебDatabaseApplicationwith PHP and MYSQL», 2nd EditionByDavidLane, Hugh E. Williams. © O'Reilly, May 2004. ISBN: 0-596- 00543-1.

12. Петюх В. М. Управління персоналом: [навч.-метод. посібник для самост. вивч. дисц.] / В.М. Петюх. – К.: КНЕУ, 2000. – 124 с.

13. Маслак О.О. Стратегічні аспекти управління персоналом / О. Маслак

// Науковий вісник НЛТУ України. – 2010. № 20/6.- С.123-129.

14. Пількевич І. А., Молодецька К. В., Сугоняк І. І., Лобанчикова Н. М. Основи побудови автоматизованих систем управління : навч. посібник. Житомир: ЖДУ ім. І. Франка, 2014. 226 с.

15. Линенко А. В. Сутність й аналіз персоналу підприємства за показниками його ефективності / А. В. Линенко, О. Л. Шевченко // Держава та регіони. Серія : Економіка та підприємництво. — 2008. — № 4. — С. 111— 116.

16. Галузинський, Г. П., Гордієнко І. В. Сучасні технологічні засоби обробки інформації : навч. посіб. К. : КНЕУ, 1998. 224 с.

17. Пушкар, О. І., Гіковатий В. М., Євсєєв О. С., Потрашкова Л. В. Системи підтримки прийняття рішень : навч. посібник; ред. О. І. Пушкар. Харків : Інжек, 2006. 304 с

18. Дерлоу Д. Ключові управлінські рішення : Технологія прийняття рішень/ Дес Дерлоу,; Пер. з англ. Р.А.Семків, Р.Л.Ткачук. -К.: Всеувито: Наук. думка, 2001. 242 с.

19. Bootstrap – бібліотека готових рішень URL: https://getbootstrap.com/docs/ 5.3/customize/color/

20. Методичні вказівки до виконання дипломної роботи освітнього рівня ―бакалавр студентами усіх форм навчання для напряму підготовки 121 –Інженерія програмного забезпечення‖ / Укладачі : Петрик М.Р., Михалик Д.М., Кінах Я.І., Гладьо С.В., Цуприк Г.Б. – Тернопіль : Вид-во ТНТУ імені Івана Пулюя, 2016 – 28 с.

21. М.Р. Петрик, Д.М. Михалик, О.Ю. Петрик, Г.Б. Цуприк. Методичні вказівки до виконання атестаційної роботи магістра за спеціальністю 121 – “Інженерія програмного забезпечення” для усіх форм навчання [Текст] – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя – 2020 – 27 с.

22. Методичні вказівки до виконання атестаційної роботи магістра за спеціальністю 121 – Інженерія програмного забезпечення (Освітньо-професійна програма - «Інженерія програмного забезпечення») для студентів усіх форм навчання / Упор.: М.Р. Петрик, Д.М. Михалик, Я.І. Кінах, Г.Б. Цуприк -

Тернопіль: ТНТУ, 2017-38 с

23. Бойко І.В., М.Р. Петрик, Г.Б. Цуприк. Інформаційні технології видобутку даних (Data mining, високопродуктивні обчислення у складних системах): навчальний посібник. Тернопіль: : ТНТУ 2020 – 62 с.

24. Бойко І.В., М.Р. Петрик, Г.Б. Цуприк. Моделювання та видобуток даних (висопродуктивні обчислення у великих алгебраїчних та числових системах, комбінаторному аналізі): навчальний посібник. Тернопіль: : ТНТУ 2019 – 62 с.

25. Нога В.В., Цуприк Г.Б. Інформаційні технології при створенні автоматизованих систем для задач збору та збереження інформації. Збірник тез доповідей VII Міжнародної науково-технічної конференції молодих учених та студентів. Актуальні задачі сучасних технологій – Тернопіль 28-29 листопада 2018. — Т. : ТНТУ, 2018. — Том 2. — С. 131.

26. І.В. Бойко, М.Р. Петрик, Г.Б. Цуприк. Дискретні структури (Алгебраїчні та числові системи, комбінаторний аналіз). Навчально-методичний посібник для студентів спеціальності 121 «Інженерія програмного забезпечення», аспірантів та викладачів вищих навчальних закладів. Конспект лекцій : Навчальний посібник / І.В. Бойко, М.Р. Петрик, Г.Б. Цуприк – Тернопіль : ТНТУ , 2017. – 64 с.

27. Якість та тестування ПЗ. URL: https://dl.tntu.edu.ua/index.php

28. Java - Вікіпедія [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Java.

29. Firebase - Вікіпедія [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Firebase (дата звернення: 30.05.2023).

30. Лабораторна робота №3 [Електронний ресурс] – Режим доступу до ресурсу: https://dl.tntu.edu.ua/get.php/lab20172018/lab3.pdf.

31. Firebase Test Lab [Електронний ресурс] – Режим доступу до ресурсу: https://firebase.google.com/docs/test-lab.

32. What is automation testing? URL: https://www.globalapptesting.com/blog/what-is-automation-testing.

ДОДАТКИ

ДОДАТОК А

Моделювання бізнес-процесів предметної області

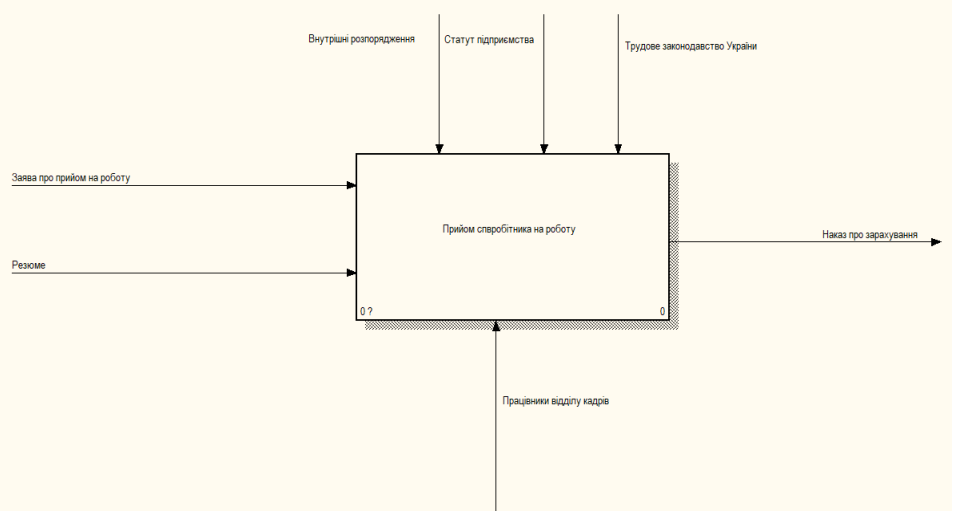


Рисунок А.1 – Контекстна діаграма «Прийом співробітника на роботу»

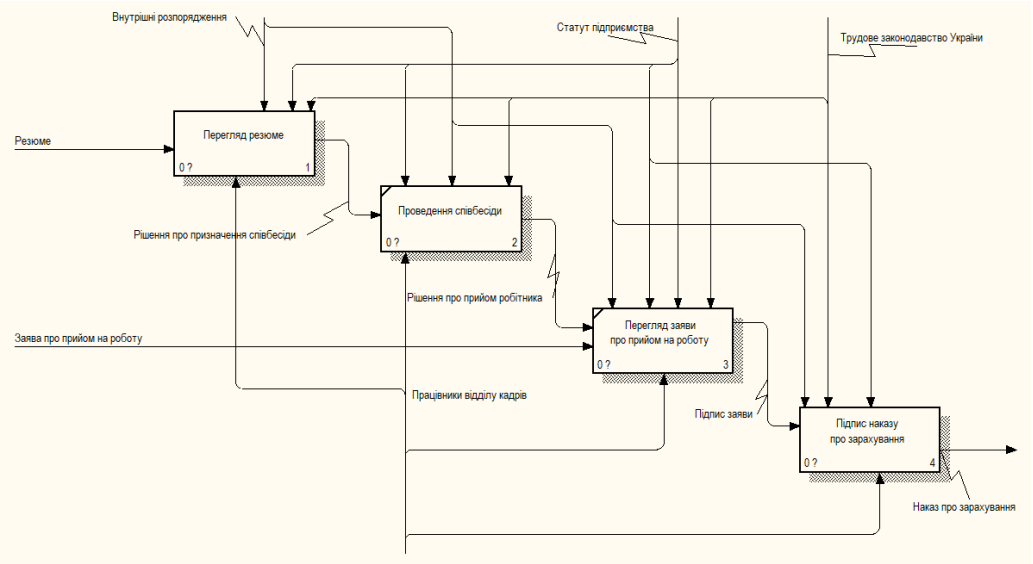


Рисунок А.2 – Декомпозиція процесу «Прийом співробітника на роботу»

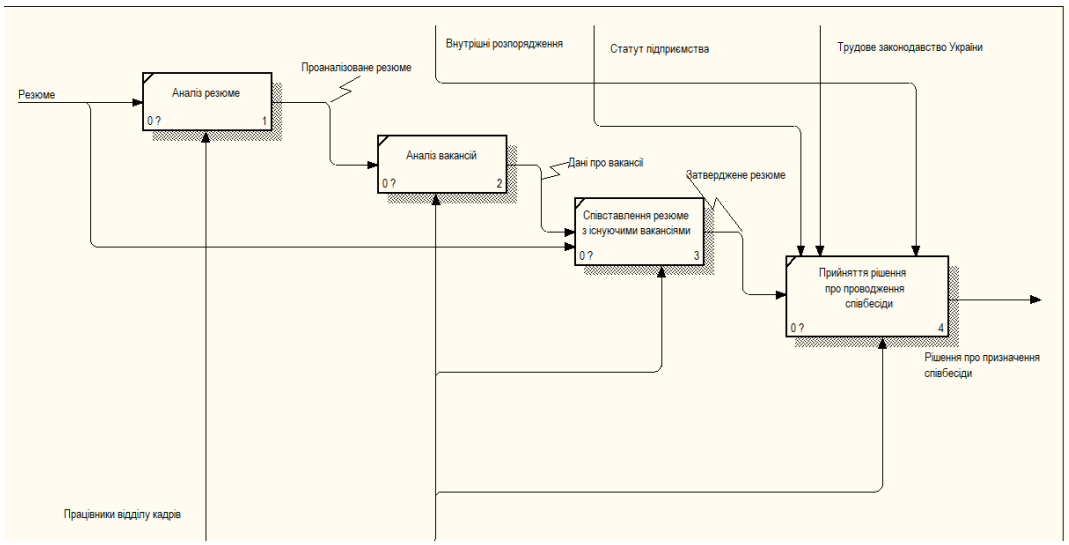


Рисунок А.3 – Декомпозиція процесу «Перегляд резюме»

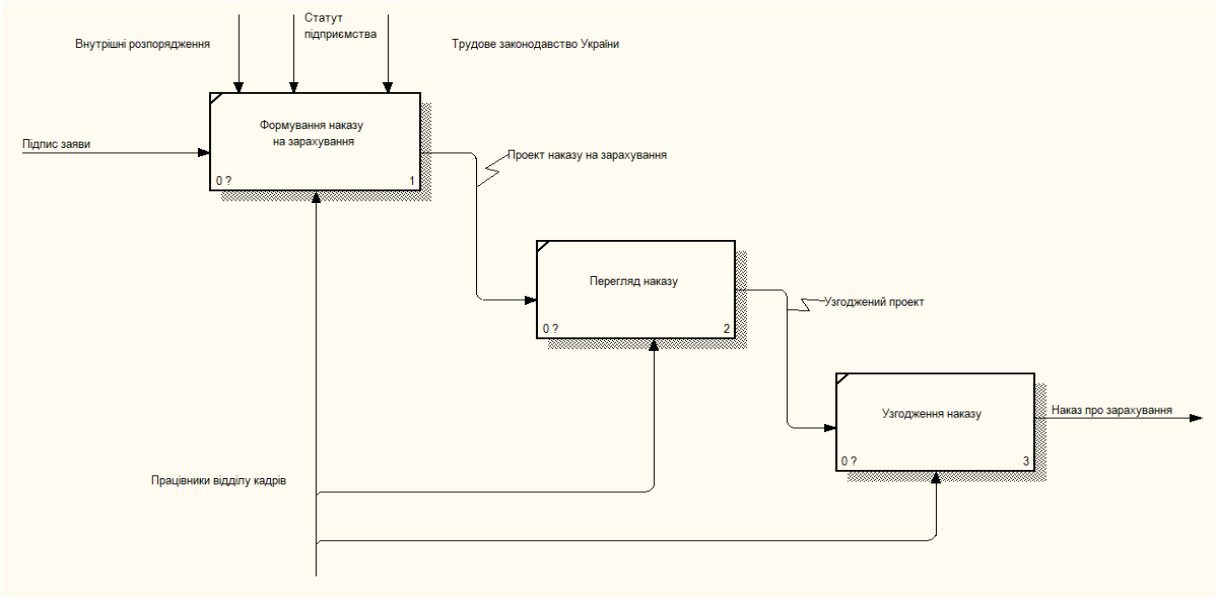
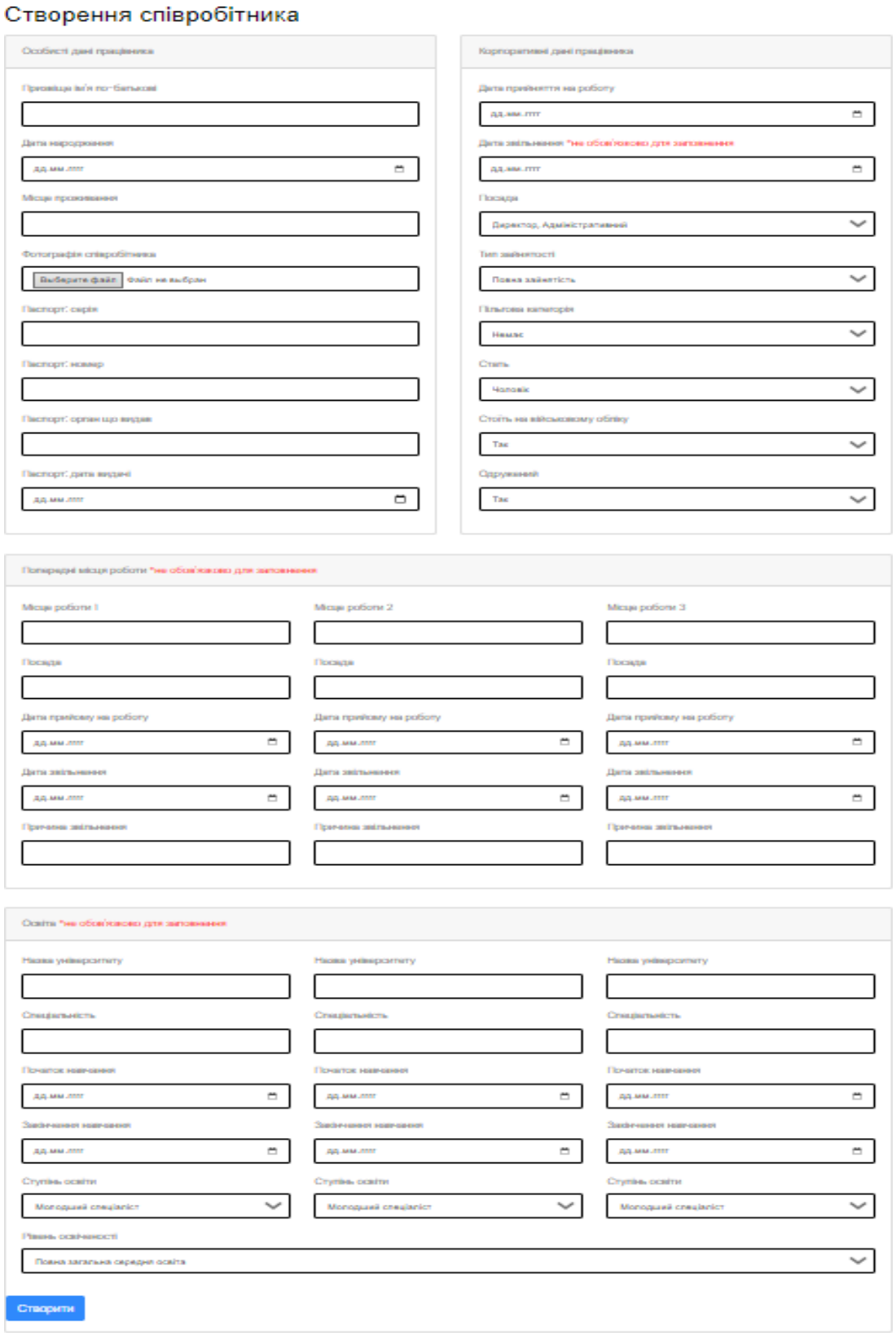


Рисунок А.4 – Декомпозиція процесу «Підпис наказу про зарахування»

ДОДАТОК Б

Форма створення співробітника



ДОДАТОК Д

Код контролера EmployeeController відповідає за створення редагування та виведення інформації про співробітника

```
<?php
namespace App\Http\Controllers\Employee;

use App\Models\EducationalLevel;
use App\Models\Employee;
use App\Models\EmployeeData;
use App\Models\Mail;
use App\Models\Position;
use App\Models\Subdivision;
use App\User;
use Illuminate\Http\Request;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Storage;
use Illuminate\Support\Facades\File;

/*
 * Контролер відповідає за створення, редагування та виведення інформації про співробітника
 */

class EmployeeController extends Controller
{
   public $position;
   public $educational_level;
   public $employeeData;
   public $employee;
   public $user;
   public $mail;
   public $subdivision;

   public function __construct(Position $position, EducationalLevel $educational_level, EmployeeData
$employeeData,
   Employee $employee, User  $user, Mail $mail, Subdivision $subdivision)
```

```
{

    $this->position = $position;
    $this->educational_level = $educational_level;
    $this->employeeData = $employeeData;
    $this->employee = $employee;
    $this->user = $user;
    $this->mail = $mail;
    $this->subdivision = $subdivision;
}

public function index()
{
    $data = [
        'employees' => $this->employee->withTrashed()->get() // SELECT * FROM employees
    ];

    return view('employee.index', $data);
}

public function create()
{
    $data = [
      'positions' => $this->position->all(), //SELECT * FROM positions
        'educational_levels' => $this->educational_level->all() // SELECT * FROM educational_levels
    ];
    return view('employee.create', $data);
}

public function store(Request  $request)
{
    /*
     *
     * Збереження співробітника ділиться на 2 фази
     * 1 - запис особистої інформації співробітника в таблицю employee_data і дістаємо ІД запису
     * 2 - запис загальної інформації співробітника в таблицю employees
```

```
 *
 */

$employee = new Employee();
$employeeData = new EmployeeData();



/*
 *
 * Обробляємо фото
 */

$dir = $this->employee->count()+1;
$photo = $request->file('photo');
$extension = $photo->getClientOriginalExtension();
Storage::disk('employees')->put($dir.'/'.$photo->getFilename().'.'.$extension, File::get($photo));
$employeeData->photo = $dir.'/'.$photo->getFilename().'.'.$extension;

$employeeData->address = $request->input('address');
$employeeData->date_of_birth = $request->input('date_of_birth');

/*
 * Перша робота
 * */
$employeeData->job1 = $request->input('job1');
$employeeData->job1_from = $request->input('job1_from');
$employeeData->job1_to = $request->input('job1_to');
$employeeData->job1_position = $request->input('job1_position');
$employeeData->reason1 = $request->input('reason1');

   /*
    * Друга робота
    * */
$employeeData->job2 = $request->input('job2');
$employeeData->job2_from = $request->input('job2_from');
$employeeData->job2_to = $request->input('job2_to');
```

```
$employeeData->job2_position = $request->input('job2_position');
$employeeData->reason1 = $request->input('reason2');

/*
 * Третя робота
 * */
$employeeData->job3 = $request->input('job3');
$employeeData->job3_from = $request->input('job3_from');
$employeeData->job3_to = $request->input('job3_to');
$employeeData->job3_position = $request->input('job3_position');
$employeeData->reason1 = $request->input('reason3');

$employeeData->educational_level_id = $request->input('educational_level_id');

/*
 * Університет 1
 * */
$employeeData->university1_name = $request->input('university1_name');
$employeeData->specialty1 = $request->input('specialty1');
$employeeData->university1_from = $request->input('university1_from');
$employeeData->university1_to = $request->input('university1_to');
$employeeData->university1_degree = $request->input('university1_degree');

/*
 * Університет 2
 * */
$employeeData->university2_name = $request->input('university2_name');
$employeeData->specialty2 = $request->input('specialty2');
$employeeData->university2_from = $request->input('university2_from');
$employeeData->university2_to = $request->input('university2_to');
$employeeData->university2_degree = $request->input('university2_degree');

/*
 * Університет 3
 * */
$employeeData->university3_name = $request->input('university3_name');
```

```
    $employeeData->specialty3 = $request->input('specialty3');
    $employeeData->university3_from = $request->input('university3_from');
    $employeeData->university3_to = $request->input('university3_to');
    $employeeData->university3_degree = $request->input('university3_degree');

    $employeeData->pilgova_category = $request->input('pilgova_category');
    $employeeData->military = $request->input('military');
    $employeeData->married = $request->input('married');
    $employeeData->passport_seria = $request->input('passport_seria');
    $employeeData->passport_number = $request->input('passport_number');
    $employeeData->passport_organ = $request->input('passport_organ');
    $employeeData->passport_date = $request->input('passport_date');

    $employeeData->save(); //Зберігаємо запис в таблицю employee_data


    $employeeDataId = $employeeData->id; //Витягуємо id

    $employee->pib = $request->input('pib');
    $employee->date_of_employment = $request->input('date_of_employment');
    $employee->date_of_dismissal = $request->input('date_of_dismissal');
    $employee->position_id = $request->input('position_id');
    $employee->type_of_employment = $request->input('type_of_employment');
    $employee->employee_data_id = $employeeDataId;

    $employee->save();

    $employeeSavedId = $employee->id;

    $message = "Співробітника ".$employee->pib." було успішно створено. ID ".$employeeSavedId;
    return redirect()->back()->withErrors(['message' => $message]);


}

public function show($id)
```

```
{
    $data = [
        'employee' => $this->employee->select()->where('id', $id)->withTrashed()->first()
    ];
    return view('employee.show', $data);
}

public function edit($id)
{
    $data =
        [
            'employee' => $this->employee->select()->where('id', $id)->withTrashed()->first(),
            'positions' => $this->position->all(),
            'educational_levels' => $this->educational_level->all()
        ];

    return view('employee.edit', $data);
}

public function update(Request $request)
{
    /*
     *
     * Збереження співробітника ділиться на 2 фази
     * 1 - запис особистої інформації співробітника в таблицю employee_data і дістаємо ІД запису
     * 2 - запис загальної інформації співробітника в таблицю employees
     *
     */

    $employee = $this->employee->where('id', $request->input('id'))->withTrashed()->first();
    $employeeData = $this->employeeData->where('id', $employee->employee_data->id)->first();

    /*
     *
     * Обробляємо фото
     */
```

```
if($request->file('photo') != null)
{
   $dir = $this->employee->where('id', $request->input('id'))->first()->id;
   $photo = $request->file('photo');
   $extension = $photo->getClientOriginalExtension();
   Storage::disk('employees')->put($dir.'/'.$photo->getFilename().'.'.$extension, File::get($photo));
   $employeeData->photo = $dir.'/'.$photo->getFilename().'.'.$extension;
}

/*       $request->photo->store('employees', 'public');
      $employeeData->photo = $request->photo->hashName();*/

$employeeData->address = $request->input('address');
$employeeData->date_of_birth = $request->input('date_of_birth');



/*
 * Перша робота
 * */
$employeeData->job1 = $request->input('job1');
$employeeData->job1_from = $request->input('job1_from');
$employeeData->job1_to = $request->input('job1_to');
$employeeData->job1_position = $request->input('job1_position');

/*
 * Друга робота
 * */
$employeeData->job2 = $request->input('job2');
$employeeData->job2_from = $request->input('job2_from');
$employeeData->job2_to = $request->input('job2_to');
$employeeData->job2_position = $request->input('job2_position');

/*
 * Третя робота
 * */
$employeeData->job3 = $request->input('job3');
```

```
$employeeData->job3_from = $request->input('job3_from');
$employeeData->job3_to = $request->input('job3_to');
$employeeData->job3_position = $request->input('job3_position');

$employeeData->educational_level_id = $request->input('educational_level_id');

/*
 * Університет 1
 * */
$employeeData->university1_name = $request->input('university1_name');
$employeeData->specialty1 = $request->input('specialty1');
$employeeData->university1_from = $request->input('university1_from');
$employeeData->university1_to = $request->input('university1_to');
$employeeData->university1_degree = $request->input('university1_degree');

/*
 * Університет 2
 * */
$employeeData->university2_name = $request->input('university2_name');
$employeeData->specialty2 = $request->input('specialty2');
$employeeData->university2_from = $request->input('university2_from');
$employeeData->university2_to = $request->input('university2_to');
$employeeData->university2_degree = $request->input('university2_degree');

/*
 * Університет 3
 * */
$employeeData->university3_name = $request->input('university3_name');
$employeeData->specialty3 = $request->input('specialty3');
$employeeData->university3_from = $request->input('university3_from');
$employeeData->university3_to = $request->input('university3_to');
$employeeData->university3_degree = $request->input('university3_degree');
$employeeData->married = $request->input('married');
$employeeData->passport_number = $request->input('passport_number');
$employeeData->passport_seria = $request->input('passport_seria');
$employeeData->passport_date = $request->input('passport_date');
```

```
$employeeData->passport_organ = $request->input('passport_organ');
$employeeData->pilgova_category = $request->input('pilgova_category');
$employeeData->military = $request->input('military');
$employeeData->sex = $request->input('sex');

$employeeData->save(); //Зберігаємо запис в таблицю employee_data

$employeeDataId = $employeeData->id; //Витягуємо id

$employee->pib = $request->input('pib');
$employee->date_of_employment = $request->input('date_of_employment');
$employee->date_of_dismissal = $request->input('date_of_dismissal');
$employee->position_id = $request->input('position_id');
$employee->type_of_employment = $request->input('type_of_employment');
$employee->employee_data_id = $employeeDataId;
```

ДОДАТОК Г

Код контролера Vacation Controller відповідає за створення відпусток для співробітників

```
<?php

namespace App\Http\Controllers\Vacation;

use App\Models\Employee;
use App\Models\Vacation;
use Illuminate\Http\Request;
use App\Http\Controllers\Controller;

class VacationController extends Controller
{
   public $vacation;
   public $employee;
   public function __construct(Vacation $vacation, Employee $employee)
   {
      $this->vacation = $vacation;
      $this->employee = $employee;
   }
   public function index()
   {
      $data = [
         'vacations' => $this->vacation->all()
      ];

      return view('vacations.index', $data);
   }
   public function create()
   {
      $data =
         [
            'employees' => $this->employee->select('id', 'pib')->get()
         ];
```

```
        return view('vacations.create', $data);
    }

    public function store(Request $request)
    {
        $data  =  $request->all();
        unset($data['_token']);
        Vacation::create($data);
        return redirect()->route('vacations.index');
    }
    public function delete($id)
    {
        $vacation = $this->vacation->where('id', $id)->first();
        $message = "Відпустка працівника ".$vacation->employee->pib." з ".$vacation->from." по
".$vacation->to." була скасована";
        $vacation->delete();
        return redirect()->route('vacations.index')->withErrors(['message' => $message]);
```

ДОДАТОК Д

Код контроллера StatisticsController(відповідає за створення статистики)

```
<?php

namespace App\Http\Controllers\Statistics;

use App\Models\EmployeeData;
use Illuminate\Http\Request;
use App\Http\Controllers\Controller;
use DateTime;
use Illuminate\Support\Facades\DB;
/*
 * Відповідає за статистику
 */
class StatisticsController extends Controller
{
   function calculateAge($birthdate)
   {
      $today = new DateTime();
      $diff = $today->diff(new DateTime($birthdate));
      return $diff->y;
   }
   public function index()
   {
      // Графік по віковій категорії
      $labels = ['18-25 років', '26-32 років', '33-40років', '40-50 років', '50-60 років', '60-72 років'];

      $data = [];

      foreach ($labels as $label) {
         $range = explode('-', $label);
         $minAge = intval($range[0]);
         $maxAge = intval($range[1]);

         $employeesCount = EmployeeData::whereRaw("TIMESTAMPDIFF(YEAR, date_of_birth, CURDATE())
```

```
>= $minAge")
            ->whereRaw("TIMESTAMPDIFF(YEAR, date_of_birth, CURDATE()) <= $maxAge")
            ->count();
        $data[] = $employeesCount;
    }
    $labels2 = []; // Замініть цей масив на ваші рівні освіти
    $data2 = [];
    $educationLevels = DB::table('educational_levels')->pluck('name', 'id');
    foreach ($educationLevels as $id => $name) {
        $employeesCount = EmployeeData::where('educational_level_id', $id)->count();
        $labels2[] = $name;
        $data2[] = $employeesCount;
    }
    return view('chart', compact('labels2', 'data2', 'labels', 'data'));
    return view('chart', compact('labels', 'data', 'labels2', 'data2'));

  }
```

Код employee/create.blade.php(Форма для створення співробітника)

```
@extends('layouts.app')

@section('content')

  <div class = "container">
    @if($errors->any())
      <div class="alert alert-success alert-dismissible fade show" role="alert">
        <h4 class="alert-heading">Чудово!</h4>
        <p>{{ $errors->first() }}</p>
        <button type="button" class="close" data-dismiss="alert" aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
    @endif
    <h2>Створення співробітника</h2>
      <form method = "post" action = "{{ route('employee.store') }}" enctype="multipart/form-data">
        @csrf
        <div class = "row">
          <div class = "col-md-6">

<!--Карточка Дані про працівника-->
              <div class="card">
                <div class="card-header">
                  Особисті дані працівника
                </div>
                <div class="card-body">

                  <!-- pib -->
                  <div class="form-group">
                    <label for="exampleFormControlInput1">Призвіще ім'я по-батькові</label>
                    <input type="text" class="form-control" style = "border: 1px solid;"
placeholder="Прізвище ім'я по-батькові" name = "pib">
                  </div>
                  <!-- pib END-->


                  <!--date_of_birth-->
                  <div class="form-group">
                    <label for="exampleFormControlSelect1">Дата народження</label>
                    <input type = "date" class = "form-control" style = "border: 1px solid;" name =
"date_of_birth">
                  </div>
                  <!--date_of_birth END-->



                  <!--address-->
                  <div class="form-group">
```

```
                    <label for="exampleFormControlInput1">Місце проживання</label>
                    <input type="text" class="form-control" style = "border: 1px solid;"
placeholder="Місце проживання" name = "address">
                </div>
        <!--address END-->


          <!--photo-->
                <div class="form-group">
                    <label for="exampleFormControlInput1">Фотографія співробітника</label>
                    <input type="file" class="form-control" style = "border: 1px solid;" name = "photo"
required>
                </div>
        <!--photo END-->

                <div class = "form-group">
                    <label for="exampleFormControlInput1">Паспорт: серія</label>
                    <input type="text" class="form-control" style = "border: 1px solid;"
placeholder="Паспорт серія" name = "passport_seria" required>
                </div>

                <div class = "form-group">
                    <label for="exampleFormControlInput1">Паспорт: номер</label>
                    <input type="text" class="form-control" style = "border: 1px solid;"
placeholder="Паспорт номер" name = "passport_number" required>
                </div>

                <div class = "form-group">
                    <label for="exampleFormControlInput1">Паспорт: орган що видав</label>
                    <input type="text" class="form-control" style = "border: 1px solid;"
placeholder="Паспорт орган що видав" name = "passport_organ" required>
                </div>

                <div class = "form-group">
                    <label for="exampleFormControlInput1">Паспорт: дата видачі</label>
                    <input type="date" class="form-control" style = "border: 1px solid;" name =
"passport_date" required>
                </div>

              </div>
            </div>


            <!--Карточка Дані про працівника-->
        </div>


          <div class = "col-md-6">
            <div class = "card">
              <div class = "card-header">
                Корпоративні дані працівника
              </div>
              <div class = "card-body">
```

```
                    <!--date_of_employment-->
                    <div class="form-group">
                      <label for="exampleFormControlInput1">Дата прийняття на роботу</label>
                      <input type="date" class="form-control" style = "border: 1px solid;" name =
"date_of_employment" required>
                    </div>
                    <!--date_of_employment END-->



                    <!--date_of_dismissal-->
                    <div class="form-group">
                      <label>
                        Дата звільнення <span style = "color:red;">*не обов'язково для
заповнення</span>
                      </label>
                      <input type="date" class="form-control" style = "border: 1px solid;" name =
"date_of_dismissal">
                    </div>
                    <!--date_of_dismissal END-->


                    <!--position_id-->
                      <div class="form-group">
                        <label>Посада</label>
                        <select class="form-control" style = "border: 1px solid;" name = "position_id">
                          @foreach($positions as $position)
                            <option value = "{{ $position->id }}">{{ $position->name }}, {{ $position-
>subdivision->name }}</option>
                          @endforeach
                        </select>
                      </div>
                    <!--position_id END-->

                    <!--type_of_employment-->
                    <div class = "form-group">
                      <label>Тип зайнятості</label>
                      <select class="form-control" style = "border: 1px solid;" name =
"type_of_employment">
                        <option value = "Повна зайнятість">Повна зайнятість</option>
                      </select>
                    </div>
                    <!--type_of_employment END-->

                    <div class = "form-group">
                      <label>Пільгова категорія</label>
                      <select class="form-control" name = "pilgova_category">
                        <option value = "Немає">Немає</option>
                        <option value = "Малозабезпечені сім'ї">Малозабезпечені сім'ї</option>
                        <option value = "Громадяни, які постраждали внаслідок Чорнобильської
катастрофи">Громадяни, які постраждали внаслідок Чорнобильської катастрофи</option>
                        <option value = "Особи, які постраждали від торгівлі людьми">Особи, які
постраждали від торгівлі людьми</option>
```

```
                        <option value = "Багатодітні сім'ї">Багатодітні сім'ї</option>
                        <option value = "Жінки, яким присвоєно почесне звання України “Мати-
героїня”">Жінки, яким присвоєно почесне звання України “Мати-героїня”</option>
                        <option value = "Учасники бойових дій">Учасники бойових дій</option>
                      </select>
                    </div>

                    <div class = "form-group">
                      <label>Стать</label>
                      <select class="form-control" name = "sex">
                        <option value = "Чоловік">Чоловік</option>
                        <option value = "Жінка">Жінка</option>
                      </select>
                    </div>

                    <div class = "form-group">
                      <label>Стоїть на військовому обліку</label>
                      <select class="form-control" name = "military">
                        <option value = "Так">Так</option>
                        <option value = "Ні">Ні</option>
                      </select>
                    </div>

                    <div class = "form-group">
                      <label>Одружений</label>
                      <select class="form-control" name = "married">
                        <option value = "Так">Так</option>
                        <option value = "Ні">Ні</option>
                      </select>
                    </div>



                  </div>
                </div>
              </div>



              <div>



              </div>
            </div>

            <div class="card" style = "margin-top: 30px;">
              <div class="card-header">
                Попередні місця роботи <span style = "color:red;">*не обов'язково для
заповнення</span>
              </div>
              <div class="card-body">
                <div class = "row">
```

```
                <div class = "col-md-4">


                  <!--job1-->
                  <div class="form-group">
                    <label for="exampleFormControlInput1">Місце роботи 1</label>
                    <input type="text" class="form-control" style = "border: 1px solid;"
placeholder="Місце роботи 1" name="job1">
                  </div>
                  <!--job1 END-->


                  <!--job1_position-->
                  <div class = "form-group">
                    <label for="exampleFormControlInput1">Посада</label>
                    <input type="text" class="form-control" style = "border: 1px solid;"
placeholder="Посада" name = "job1_position">
                  </div>
                  <!--job1_position END-->


                  <!--job1_from-->
                  <div class="form-group">
                    <label>Дата прийому на роботу</label>
                    <input type = "date" class = "form-control" style = "border: 1px solid;" name =
"job1_from">
                  </div>
                  <!--job1_from END-->

                  <!--job1_to-->
                  <div class="form-group">
                    <label for="exampleFormControlSelect1">Дата звільнення</label>
                    <input type = "date" class = "form-control" name = "job1_to">
                  </div>
                  <!--job1_to END-->

                  <div class = "form-group">
                    <label>Причина звільнення</label>
                    <input type = "text" name = "reason1" class="form-control">
                  </div>


                </div>

                <div class = "col-md-4">


                  <!--job2-->
                  <div class="form-group">
                    <label for="exampleFormControlInput1">Місце роботи 2</label>
                    <input type="text" class="form-control" placeholder="Місце роботи 2" name =
"job2">
                  </div>
                  <!--job2 END-->
```

```
                    <!--job2_position-->
                    <div class = "form-group">
                      <label for="exampleFormControlInput1">Посада</label>
                      <input type="text" class="form-control" placeholder="Посада" name =
"job2_position">
                    </div>
                    <!--job2_position END-->


                    <!--job2_from-->
                    <div class="form-group">
                      <label for="exampleFormControlSelect1">Дата прийому на роботу</label>
                      <input type = "date" class = "form-control" name = "job2_from">
                    </div>
                    <!--job2_from END-->


                    <!--job2_to-->
                    <div class="form-group">
                      <label for="exampleFormControlSelect1">Дата звільнення</label>
                      <input type = "date" class = "form-control" name="job2_to">
                    </div>
                    <!--job2_to END-->

                  <div class = "form-group">
                        <label>Причина звільнення</label>
                        <input type = "text" name = "reason2" class="form-control">
                   </div>


                  </div>


                  <div class = "col-md-4">


                    <!--job3-->
                    <div class="form-group">
                      <label>Місце роботи 3</label>
                      <input type="text" class="form-control" placeholder="Місце роботи 3" name="job3">
                    </div>
                    <!--job3 END-->



                    <!--job3_position-->
                    <div class = "form-group">
                      <label for="exampleFormControlInput1">Посада</label>
                      <input type="text" class="form-control" placeholder="Посада" name =
"job3_position">
                    </div>
                    <!--job3_position END-->
```

```
                    <!--job3_from-->
                    <div class="form-group">
                      <label for="exampleFormControlSelect1">Дата прийому на роботу</label>
                      <input type = "date" class = "form-control" name = "job3_from">
                    </div>
                    <!--job3_from END-->


                    <!--job3_to-->
                    <div class="form-group">
                      <label for="exampleFormControlSelect1">Дата звільнення</label>
                      <input type = "date" class = "form-control" name = "job3_to">
                    </div>
                    <!--job3_to END-->

                    <div class = "form-group">
                      <label>Причина звільнення</label>
                      <input type = "text" name = "reason3" class="form-control">
                    </div>
                  </div>

                </div>



              </div>
            </div>

<div class = "card" style = "margin-top:30px;">
  <div class = "card-header">
    Освіта <span style = "color:red;">*не обов'язково для заповнення</span>
  </div>
  <div class = "card-body">
    <div class = "row">
      <!--university1-->
      <div class = "col-md-4">


        <!--university1_name-->
          <div class = "form-group">
            <label>Назва університету</label>
            <input class = "form-control" type="text" name = "university1_name">
          </div>
        <!--university1_name END-->



        <!--specialty1-->
        <div class = "form-group">
          <label>Спеціальність</label>
          <input class = "form-control" type = "text" name = "specialty1">
        </div>
```

```
    <!--specialty1 END-->

    <!--university1_from-->
    <div class = "form-group">
      <label>Початок навчання</label>
      <input class = "form-control" type = "date" name = "university1_from">
    </div>
    <!--university1_from END-->


    <!--university1_to-->
    <div class = "form-group">
      <label>Закінчення навчання</label>
      <input class = "form-control" type = "date" name = "university1_to">
    </div>
    <!--university1_to END-->

    <!--university1_degree-->
      <div class = "form-group">
        <label>Ступінь освіти</label>
        <select class="form-control" name = "university1_degree">
          <option value = "Молодший спеціаліст">Молодший спеціаліст</option>
          <option value = "Бакалавр">Бакалавр</option>
          <option value = "Магістр">Магістр</option>
          <option value = "Доктор наук">Доктор наук</option>
        </select>
      </div>
    <!--university1_degree END-->



    </div>
<!--university1-->

<!--university2-->
<div class = "col-md-4">


    <!--university2_name-->
    <div class = "form-group">
      <label>Назва університету</label>
      <input class = "form-control" type="text" name = "university2_name">
    </div>
    <!--university2_name END-->



    <!--specialty2-->
    <div class = "form-group">
      <label>Спеціальність</label>
      <input class = "form-control" type = "text" name = "specialty2">
    </div>
    <!--specialty2 END-->
```

```
    <!--university2_from-->
    <div class = "form-group">
      <label>Початок навчання</label>
      <input class = "form-control" type = "date" name = "university2_from">
    </div>
    <!--university2_from END-->


    <!--university2_to-->
    <div class = "form-group">
      <label>Закінчення навчання</label>
      <input class = "form-control" type = "date" name = "university2_to">
    </div>
    <!--university2_to END-->

    <!--university2_degree-->
    <div class = "form-group">
      <label>Ступінь освіти</label>
      <select class="form-control" name = "university2_degree">
        <option value = "Молодший спеціаліст">Молодший спеціаліст</option>
        <option value = "Бакалавр">Бакалавр</option>
        <option value = "Магістр">Магістр</option>
        <option value = "Доктор наук">Доктор наук</option>
      </select>
    </div>
    <!--university2_degree END-->



</div>
<!--university2 END-->

<!--university3-->
<div class = "col-md-4">


    <!--university3_name-->
    <div class = "form-group">
      <label>Назва університету</label>
      <input class = "form-control" type="text" name = "university3_name">
    </div>
    <!--university3_name END-->



    <!--specialty3-->
    <div class = "form-group">
      <label>Спеціальність</label>
      <input class = "form-control" type = "text" name = "specialty3">
    </div>
    <!--specialty3 END-->

    <!--university3_from-->
    <div class = "form-group">
```

```
            <label>Початок навчання</label>
            <input class = "form-control" type = "date" name = "university3_from">
          </div>
          <!--university3_from END-->


          <!--university3_to-->
          <div class = "form-group">
            <label>Закінчення навчання</label>
            <input class = "form-control" type = "date" name = "university3_to">
          </div>
          <!--university3_to END-->

          <!--university3_degree-->
          <div class = "form-group">
            <label>Ступінь освіти</label>
            <select class="form-control" name = "university3_degree">
              <option value = "Молодший спеціаліст">Молодший спеціаліст</option>
              <option value = "Бакалавр">Бакалавр</option>
              <option value = "Магістр">Магістр</option>
              <option value = "Доктор наук">Доктор наук</option>
            </select>
          </div>
          <!--university3_degree END-->



        </div>
        <!--university3 END-->
      </div>

      <!--educational_level-->
      <div class = "form-group">
        <label>Рівень освіченості</label>
        <select class = "form-control" name =
          "educational_level_id">
          @foreach($educational_levels as
          $educational_level)
            <option value = "{{ $educational_level->id }}">{{$educational_level-
          >name}}</option>@endforeach
        </select>
      </div>
      <!--educational_level END-->

    </div>

    <!--submit button-->
    <div class = "row">
      <div class = "col-md-4">
        <div class = "form-group">
          <button class="btn btn-primary" type = "submit">Створити</button>
        </div>
      </div>
```

```
</div>
<!--submit button END-->


    </div>

            </form>

      </div>
    <style>
      input.form-
        control{
        border: 1px
        solid;
      }

      select.fo
        rm-
        contr
        ol{
        borde
        r: 1px
        solid;
      }
```