# ЗМІСТ

[Перелік познак та скорочень 4](#_Toc169612913)

[Вступ 5](#_Toc169612914)

[1 Організація планування навчального навантаження з оцінкою параметрів розподілу 7](#_Toc169612915)

[1.1 Опис процесу планування навчального навантаження 7](#_Toc169612916)

[1.2 Організація розподілу начального навантаження із оцінюванням його якості 13](#_Toc169612917)

[2 Розробка алгоритмічного та інформаційного забезпечення процесу оцінювання розподілу навчального навантаження 17](#_Toc169612918)

[2.1 Розробка алгоритмів оцінювання розподілу навчального навантаження 17](#_Toc169612919)

[2.1.1 Розробка моделі оцінювання повноти розподілу 17](#_Toc169612920)

[2.1.2 Розробка моделі оцінювання коректності розподілу 19](#_Toc169612921)

[2.1.3 Розробка моделі оцінювання якості розподілу 20](#_Toc169612922)

[2.2 Розробка математичного забезпечення для обчислення показників розподілу навчального навантаження 24](#_Toc169612923)

[2.3 Організація зберігання інформації для розрахунку показників 30](#_Toc169612924)

[2.3.1 Загальні відомості про бази даних та системи керування ними 30](#_Toc169612925)

[2.3.2 Розробка структури бази даних для забезпечення розрахунків 32](#_Toc169612926)

[2.3.3 Вибір формату системи керування базами даних 36](#_Toc169612927)

[3 Проєктування програми для розрахунку показників розподілу навчального навантаження 38](#_Toc169612928)

[3.1 Розробка завдання на створення програмного рішення 38](#_Toc169612929)

[3.1.1 Ідентифікація продукту 38](#_Toc169612930)

[3.1.2 Глосарій 39](#_Toc169612931)

[3.1.3 Вимоги до програмного рішення 39](#_Toc169612932)

[3.2 Визначення стеку технологій для розробки 42](#_Toc169612933)

[3.3 Організація тестування 46](#_Toc169612934)

[4 Опис підготовки до застосування та використання програмного рішення 50](#_Toc169612935)

[4.1 Опис підготовки програмного рішення до використання 50](#_Toc169612936)

[4.2 Опис порядку використання функціоналу програмного забезпечення для формування статистики 53](#_Toc169612937)

[5 Економічне обґрунтування проекту щодо розробки ПЗ 59](#_Toc169612938)

[5.1 Обґрунтування доцільності розробки програмного забезпечення 59](#_Toc169612939)

[5.2 Оцінка конкурентоспроможності ПЗ у порівнянні з аналогом 59](#_Toc169612940)

[5.3 Обґрунтування доцільності розробки ПЗ 60](#_Toc169612941)

[5.4 Розрахунок проєктних витрат на розробку ПЗ 64](#_Toc169612942)

[5.5 Розрахунок витрат на впровадження ПЗ 65](#_Toc169612943)

[5.6 Розрахунок витрат на впровадження ПЗ 66](#_Toc169612944)

[5.7 Розрахунок поточних експлуатаційних витрат ПЗ та його аналога 66](#_Toc169612945)

[5.8 Розрахунок показника економічного ефекту від розробки програмного забезпечення 69](#_Toc169612946)

[Висновки 71](#_Toc169612947)

[Список джерел інформації 73](#_Toc169612948)

# Перелік познак та скорочень

БД – база даних;

БП – бізнес-процес;

ДАК – державна атестаційна комісія;

ЗВО – заклад вищої освіти;

ІЗ СРС – індивідуальне заняття за тематикою самостійної роботи;

ЛЗ – лабораторне заняття;

НПП – науково-педагогічний працівник;

НТУ «ХПІ» - Національний технічний універсистет «Харківський політехнічний інститут»;

ПрЗ – практичні заняття;

ПІІТУ – програмна інженерія та інтелектуальні технології управління;

ПЗ – програмне забезпечення;

РНН – розподіл навачального навантаження

СКБД – система керування базами даних.

# Вступ

Щороку кафедра стикається з необхідністю обчислення навчального навантаження для викладацького складу. За результатами такого планування визначається наповнення діяльності всіх викладачів на поточний навчальний рік. Підсумком цього процесу є план закріплення викладачів для проведення лекцій та практичних занять на поточний семестр. Важливим питанням є оцінка повноти та якості розподілу навантаження. Ефективний розподіл навантаження забезпечує високу якість навчання, що є одним із завдань кафедри. Тому, організація оцінювання розподілу навчального навантаження є актуальною задачею. Особливість оцінювання полягає в необхідності врахування результатів у дуже короткий термін. Це визначає необхідність того, що розрахунок параметрів, необхідних для оцінки отриманого варіанту розподілу викладачів, яка запланована до вирішення в даній роботі, має здійснюватися автоматизовано.

Об'єктом дослідження визначена методика планування розподілу навчального навантаження науково-педагогічних працівників кафедри на початку навчального року.

Предметом дослідження визначена модель представлення розподілу навчального навантаження науково-педагогічних працівників кафедри, показники оцінки розподілу та моделі їх обчислення.

Мета роботи – автоматизувати розрахунок показників повноти, коректності та якості розподілу навчального навантаження науково-педагогічних працівників кафедри для зменшення часу виконання цієї операції у порівнянні із ручним обчислюваннями.

Завдання роботи – для визначеного набору показників розподілу навчального навантаження розробити алгоритми для їх обчислення, вирішити питання для представлення необхідних даних, розробити програмне забезпечення, що реалізує розрахунки відповідно до алгоритмів, а також представляє їх у зручному вигляді.

В розділах основної частини роботи розглядається побудова бізнес-процесу розподілу навчального навантаження, визначення основних понять предметної області даної задачі, розробку математичного, інформаційного та програмного забезпечення для здійснення відповідних розраухунків, а також еконмічне обгрунтування створення програми.

Особливості організації розподілу навчального навантаження науково-педогогічних працівників на початку нового року та зміст підходу до оцінювання, що пропонується, рзглядається у першому розділі.

# 1 Організація планування навчального навантаження з оцінкою параметрів розподілу

## 1.1 Опис процесу планування навчального навантаження

У сучасному світі управління освітньою діяльністю в університетах потребує координації великої кількості фахівців різних напрямків. Цього можна досягти через організацію звітування про різні види діяльності, проведення організаційних та методичних нарад кафедр, проведення науково-методичних семінарів для підвищення наукового рівня та рівня метдичної готовності д о проведення занять різних видів, тощо. Оскільки НТУ «ХПІ» є державною установою, підпорядкованою Міністерству освіти і науки України, цей процес ускладнюється необхідністю дотримання численних законів, регулятивних актів та наказів, зокрема, Закону України «Про вищу освіту» [1].

Розрахунок навчального навантаження вимагає врахування численних факторів, які часто змінюються і мають обмеження. Зміна будь-якого з цих факторів вимагає, щоб розрахунки були переглянуті. Причиною змін може бути введенні в дію нового розпорядження Міністерства освіти і науки України, здійснитися перегляд навчальних програм або реструктуризація підрозділів університету.

Зазвичай такі зміни реалізуються і затверджуються наприкінці або на початку нового академічного року. Проте є ситуації, коли зміни відбуваються під час навчального семестру або між семестрами. Тому розрахунки навчального навантаження та підготовка відповідних документів повинні бути не тільки регулярними та планованими, але й потребують оперативного оновлення в обмежений час. Це підвищує ризик помилок через людський фактор, оскільки обробка даних вимагає зосередження на числових даних і взаємозв'язках між ними, предметами і викладачами.

Розробку програмного забезпечення зазвичай починається із вивчення потреб майбутніх користувачів та зясування того, яким чином вони виконуються функціональні обов’язки, які планується автоматизувати [2]. Це процес називається моделюванням бізнес-процесу. У ДСТУ визначення бізнес-процесу надається у стандарті ISO 9000:2015. Згідно з ним, бізнес-процес (БП) – це «… сукупність взаємопов'язаних або взаємодійних дій, які перетворюють входи на виходи, створюючи цінність для споживача, будь то внутрішнього чи зовнішнього. Це послідовність крос-функціональних дій, яка має чітко визначені початок і кінець, і яка повторюється у часі» [3]. Бізнес-процес має визначені чітко позначені складові:мету, вхідні та вихідні дані, а також виконавців. Теорію моделювання та рефакторингу БП можна розглянути у [4].

Назва БП, який розглядається отримав назву «Формування розподілу навчального навантаження (РНН) науково-педагогічних працівників (НПП) на навчальний рік».

Підрозділ, де реалізується бізнес-процес: навчальна частина кафедри ППІТУ.

Виконавець: завідувач кафедри та співробітники, залучені до планування навчального процесу.

Вихідні дані для розробки:

навчальні плани на поточний навчальнний рік;

дані про академічні групи;

дані про навчальні плани, за якими вивчаються студенти груп;

дані про блоки вибіркових дисциплін, обрані для вивчення студентами академічних груп;

дані про аспірантів, які закріплені за кафедрою;

штатний розклад кафедри.

На кафедрі ПІІТУ такий розрахунок проводиться у робочій книзі Excel. Листи із сформованим розподілом (для осіннього та весного семесру) містять:

рядки, що відповідають плануванню лекцій, де вказується кількість годин на їх читання, а також на консультації та на проведення підсумкового контролю: або заліку, або екзамену;

рядки, що відповідають плануванню практичних або лабораторних занять, де вказується кількість навчальних груп, кількість годин на одну групу, а також сумарна кількість годин;

рядки, що містять дані про керівництво курсовими та дипломними проєктами (роботами), підготовку аспірантів, участь у прийомі захисту курсових проєктів (робіт);

рядки, що відображають години роботи у складі державної атестаційної комісії (ДАК), включаючи рецензування робіт.

Приклад розподілу навчального навантаження, створений на початку поточного навчального року представлений на рис.1.1.

Етапами підготовки розподілу навчального навантаження є наступні:

отримання навчальних планів та переліку груп студентів в електронному вигляді;

формування базового РНН для НПП, який включає перелік дисциплін, розподіл годин за видами занять, тип звітності та семестр вивчення;

створення потоків з академічних груп згідно з баченням керівництва, дотримуючись вимог керівних документів;

формування строк РНН НПП відповідно до переліку потоків;

виведення формул, проведення розрахунків та оформлення РНН НПП у встановленій формі;

закріплення лекторів за навчальними потоками, призначення викладачів для проведення практичних видівзанять;

призначення керівників дипломного та курсового проектування.

призначення членів ДАК та рецензентів.

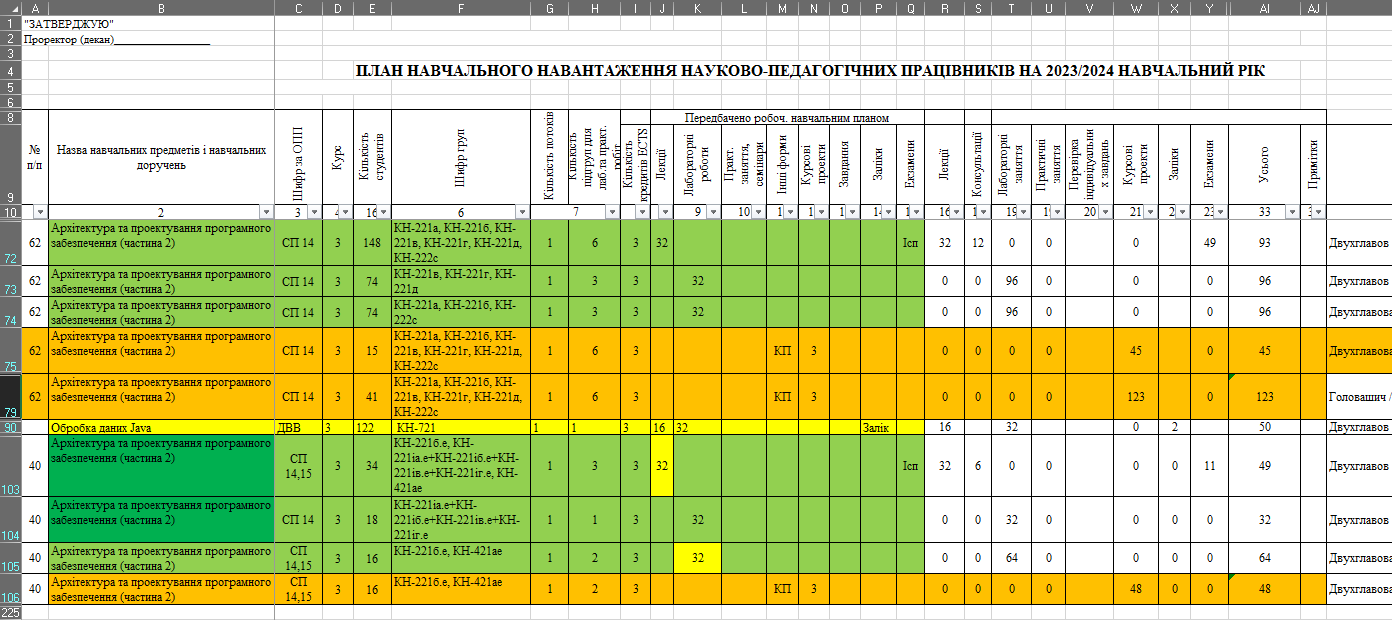


Рисунок 1.1 – Приклад аркуша листа Excel із фрагментом розподілу навчального навантаження викладів кафедри ПІІТУ

Розробка такого плану є творчою роботою, яка потребує певного досвіду. Загальні етапи цієї роботи були описані в статті [5], а також   
у магістерській роботі [6], присвяченій створенню системи підтримки розподілу навчального навантаження «Studyloаd docs» (SLDOCS).

Система «SLDOCS» була створена для автоматизації процесу складання РНН, його узгодження, виявлення та виправлення помилок. Це багатоітераційний процес, що включає монотонні ручні операції, пов'язані з підготовкою витягів для викладачів. Згідно з [6], «SLDOCS» призначена для автоматизованого створення відомостей навчальних доручень, індивідуального РНН окремого викладача та шаблону індивідуального планів плану роботи викладача на навчальний рік. Вхідними даними є РНН у форматі робочої книги Excel та список викладачів з їх електронними адресами. Система дозволяє завантажувати або відправляти документи у форматі XLSX на електронну пошту, а також дає можливість викладачам самостійно створювати облікові записи та завантажувати документи після отримання доступу.

Після завершення творчої роботи зі створення плану навчального навантаження, файл у форматі робочої книги Excel завантажується у відповідні таблиці бази даних системи через інтерфейс. Вигляд даних після завантаження можна побачити на рис.1.2. Це результат запиту, який поєднує декілька таблиць для наочності. Зручне представлення даних можна забезпечити за допомогою фреймворку Spring [7] та шаблонізатора ThemeLeаf [8], які використовувалися при розробці «SLDOCS». На рис.1.3 представлена логічна схема бази даних у форматі MySQL8 [9] для формування витягів з РНН.

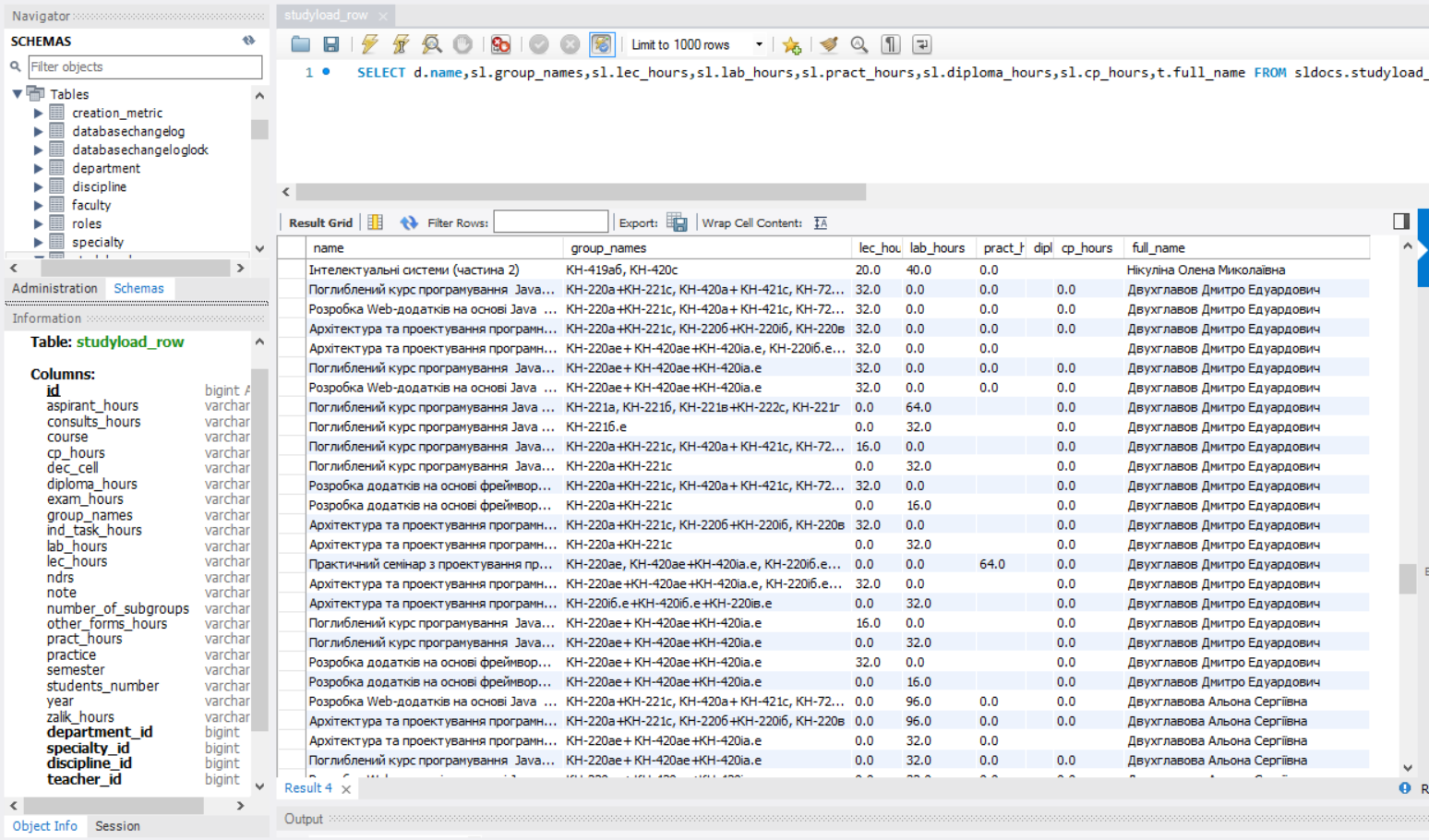


Рисунок 1.2 – Наповнення бази даних системи «SLDOCS» після завантаження відомостей про розподіл навчального навантаження із файлу Excel

Впровадження системи значно скорочує час доведення навантаження до викладачів. Наявність РНН, збереженого у базі даних (БД), дозволяє автоматизувати вирішення інших завдань. Учасники процесу планування навчального навантаження отримали можливість автоматизовано контролювати розподіл на наявність помилок, а також готувати різні зведені документи, які дозволяють бачити загальну картину сформованого розподілу. Це важливо, оскільки викладач має мати навантаження з розрахунку 600 годин на ставку на навчальний рік. Якщо кількість часів вища або менша за 600 годин (для однієї ставки) більш, ніж на 20%, то має відбутися перерозподіл навчального навантаження. Норми навантаження визначені у [10].

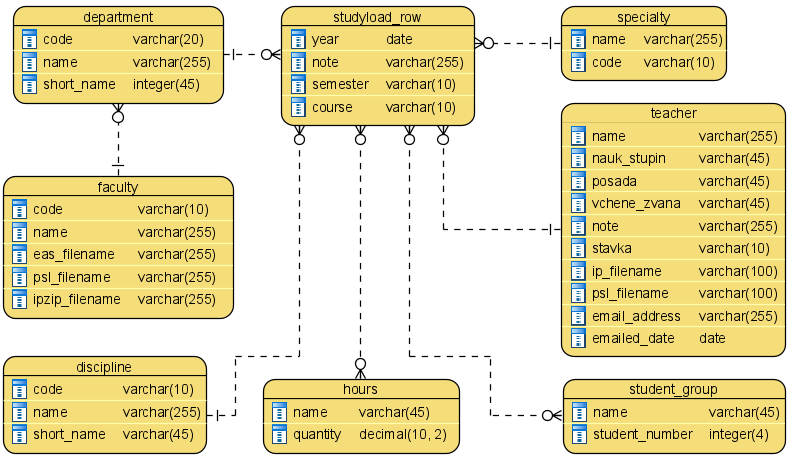


Рисунок 1.3 – Схема бази даних програмного забезпечення «SLDOCS»

Подальші міркування показали, що наявність ПНН НПП завантаженого у базу даних, відкриває широкі можливості для їх обробки. . Якщо подивитися на склад даних, що зберіга.ться у базі даних, то можна побачити, що в ній зберігається у пов'язаному вигляді інформація про викладачів, їх посади, наукові ступені (кандидат наук або доктор наук) та вчені звання (професор, доцент, академік, старший науковий співробітник), а також інформація про дисципліни, яка може бути селектована за видами навчального навантаження (лекції, лабораторні або практичні заняття, керівництво курсовими проєктами (роботами), дипломним проектуванням магістрів або бакалаврів). Зокрема, можна виразхувати узагальнені дані за розподілом, наприклад, кількість лекцій, які читають доктори наук та старші викладачі, або розрозрахувати долю лекцій у сумарній кількості годин для кожного доцента. Можливості багатоьох мов програмування також дозволяють представляти їх у зручному для аналізу вигляді – таблиці, гістонрами, діаграми, графіки. Узагальнення всіх попередніх ідей щодо використання цифрового зберігання ПНН призвело до створення бізнес-процесу «Розробка ПНН з оцінюванням якості», який розглядається у наступному розділі. Його слід розглядати, як пропозицію для вдосконалення існуючого процесу розробки.

## 1.2 Організація розподілу начального навантаження із оцінюванням його якості

Модель бізнес-процесу, який пропонується до впровадження, представлена на рис. 1.4.

Існуючий БП пропонується розширити активністю, що полягає в тому, що після створення початкового варіанту плану розподілу буде здійснено розрахунок статистики та її оцінка.

Планується, що розробник плану РНН буде активно використовувати наявну систему «SLDOCS». При цьому функціонал системи має бути розширений шляхом додавання нових функцій, зокрема, набору класів та методів для розрахунку характеристик сформованої версії ПНН.

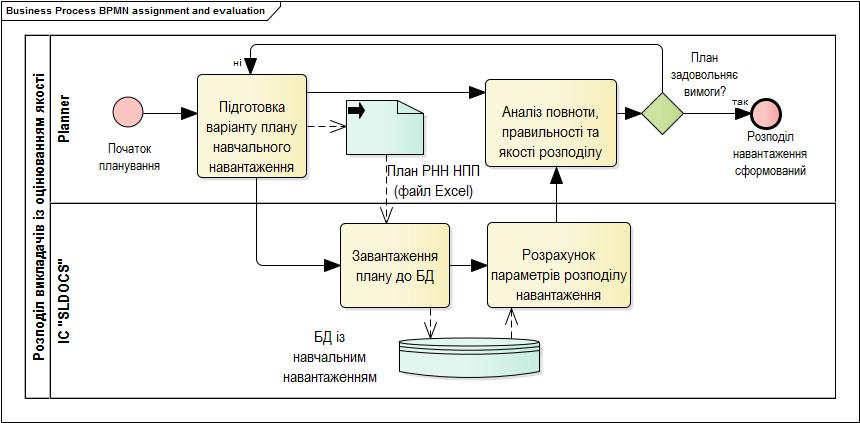


Рисунок 1.4 – Модель підготовки розподілу навчального навантаження із оцінюванням якості

Модель бізнес-процесу представлена із використанням BPMN (Business Process Model аnd Notаtion) – нотація для моделювання бізнес-пролцесів, що зберігає опис в ХML файлі, забезпечуючи можливість експорту/імпорту як моделі цілком та і її окремих пакетів. Вона була розроблена ініціативою Business Process Mаnаgement Initiаtive та підтримується Object Mаnаgement Group після злиття обох організацій у 2005 році. Остання версія BPMN – 2.0.1 [11]. Enterprise Аrchitect є однією, але основною з програм, що дозволяють створювати діаграми такого типу. Стверджується, що це «…ідеальне рішення для організацій та підприємств у сфері візуалізації, аналізу, моделювання, тестування та обслуговування їх структури, програмного забезпечення, процесів та архітектури. Це платформа, яка забезпечує можливість контролю робочого простору, підтримки колег та команди, сприяє співпраці та формує впевненість у найскладніших проєктах» [12].

Аналізуючи схему бізнес-процесу на рис. 1.4, можна побачити, що аналіз НПП планується провести за трьома напрямками:

повнота розподілу;

коректність розподілу;

якість розподілу.

Повнота розподілу навчального навантаження – характеристика варіанту розподілу, яка визначає забезпеченність викладачем всіх занять, що заплановані до проведення. Для визначення характеристики необхідно переглянути всі позиції відомості навчальних доручень, порівняти їх із РНН і перевірити:

визначення лектора для всіх лекційних потоків;

визначення викладачів для всіх груп, створених для проведення лабораторних робіт та практичних занять;

призначення керівників кваліфікаційних робіт (курсової, бакалаврської або дипломної) для всіх студентів.

Коректність розподілу навчального навантаження – характеристика варіанту розподілу, яка визначає призначення всім штатним викладачам кафедри визначеного нормами обсягу навчального навантаження.

Якість розподілу навчального навантаження – характеристика варіанту розподілу, яка визначає відповідність призначення викладачів для проведення певних видів занять відповідно до їх категорій, що визначні у методичних вказівах університету та кафедри.

Показники цієї групи можна розділити на дві категорії:

Індивідуальні показники якості розподілу:

* кількість годин та відсоток на проведення лекції та інших повязаних заходів (консультації, приймання екзаменів);
* кількість годин та відсоток на проведення практичних видів занять (лабораторних робіт та практичних занять);
* кількість годин та відсоток на керівництво курсовим проєктуванням;
* кількість годин та відсоток на керівництво дипломним проектування бакалаврів або/ та магистрів;

Показники якості розподілу занять у підрозіділі:

1. кількість годин та відсоток занять, що проводяться докторами наук, професорами.
2. кількість годин та відсоток занять, що проводяться кандидатами наук, доцентами.
3. кількість годин та відсоток занять, що проводяться НПП без звань та ступенів.

Ретельний опис, отриманий в ході аналізу бізнес-процесу, дозволяє визначити завдання, які потрібно вирішити для його реалізації. До цих етапів слід включити:

формування набору показників оцінювання варіанту РНН;

формування математичних формул для розрахунку показників оцінювання варіанту РНН, а такж та алгоритмів їх обчислення;

розробити модель зберігання даних для розрахунків, а також для збереження показників варіантів розподілу;

обраного стеку технологій, спроектувати, розробити та протестувати програмне забезпеченя для розрахунку показників варіанту розподілу;

підготувати демонстраційні приклади роботи ПЗ, а також якісно проаналізувати коректність функціонування програмного рішення, яке буде створене.

# 2 Розробка алгоритмічного та інформаційного забезпечення процесу оцінювання розподілу навчального навантаження

## 2.1 Розробка алгоритмів оцінювання розподілу навчального навантаження

### 2.1.1 Розробка моделі оцінювання повноти розподілу

Повнота розподілу навчального навантаження – характеристика варіанту розподілу, яка визначає забезпеченність викладачем всіх занять, що заплановані до проведення.

На теперішній час такий пказник розподілу навчального плану (РНН) швидше визначити, застосовуючи можливості табличного процесора Microsoft Excel. Для цього можна просто встановити фільтр на колонку із прізвищами викладачів, а потім обрати варіант фільтру «Пусті рядки». З іншого боку, якщо у переліку варіанту «Пусті рядки» немає, то це свідчить, що розподіл навантаження здійснений у повному обсязі [13].

Аналогічно визначається відсутність закріплення керівників за магістрами або бакалаврами. Так само можна накласти фільтр та отримати результат.

Але є один момент, який визначає необходимість змін – зараз розподіл на осінь та на весну розташовуються на різних листах. Це потребує установу фільтрів двічі. На рис.2.1 представлений можливий варінат представлення ерезультатів виконання етапу, коли дані зведені на один лист після простото натскання на кнопку.

На рис.2.2. показаний варіант представлення даних про дипломників та їх керівників. Тут в одному місці зведені відомості про всі категорії студентів, які пишуть кваліфікаційні роботи. Можна сказати по іншому – всі студентів 4, 5 та 6 курсів.

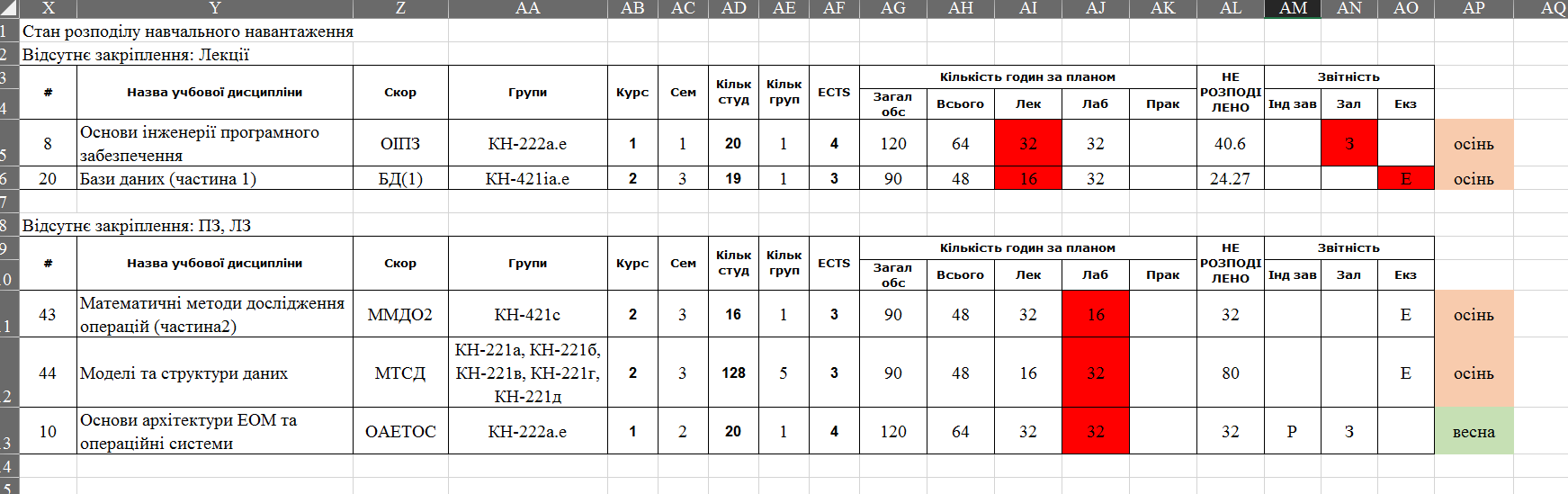


Рисунок 2.1 – Представлення відомостей для оцінки неповноти призначення викладачів на дисципліни

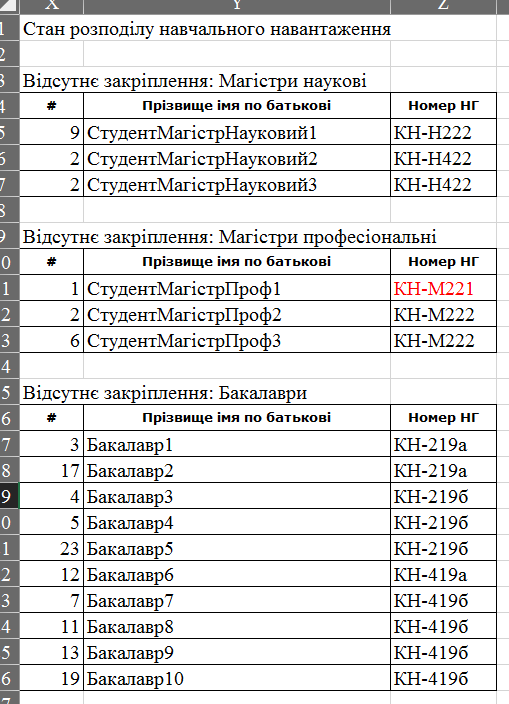


Рисунок 2.2 – Представлення відомостей для оцінки неповноти призначення керівників дипломного проектування

Обидва варіант демонструють варіант, який припускає можливість зберегти у окремому файлі всі виявлені недоліки. Далі цей файл можна відправити на друк, відправити поштою завідуючому кафедри для аналізу, представити на слайді презентації, що буде демонструватися на службовій нараді. Також перелік у форматі «всі студенти без креівників 4 та 5 курсів», можна розташувати на загальних огляд викладачам для приймання у вибору нових вихованців у недалекому майбутньому.

Наступним етапом є оцінювання коректності розподілу.

### 2.1.2 Розробка моделі оцінювання коректності розподілу

Коректність розподілу навчального навантаження – характеристика варіанту розподілу, яка визначає призначення всім штатним викладачам кафедри визначеного нормами обсягу навчального навантаження.

Автоматизація розрахунків показніків цієї групи задача більш ресурсовитратна, а тому і більш актуальна. Приклад сторінки із зведеними даними про навантаження викладачів представлений на рис.2.3.

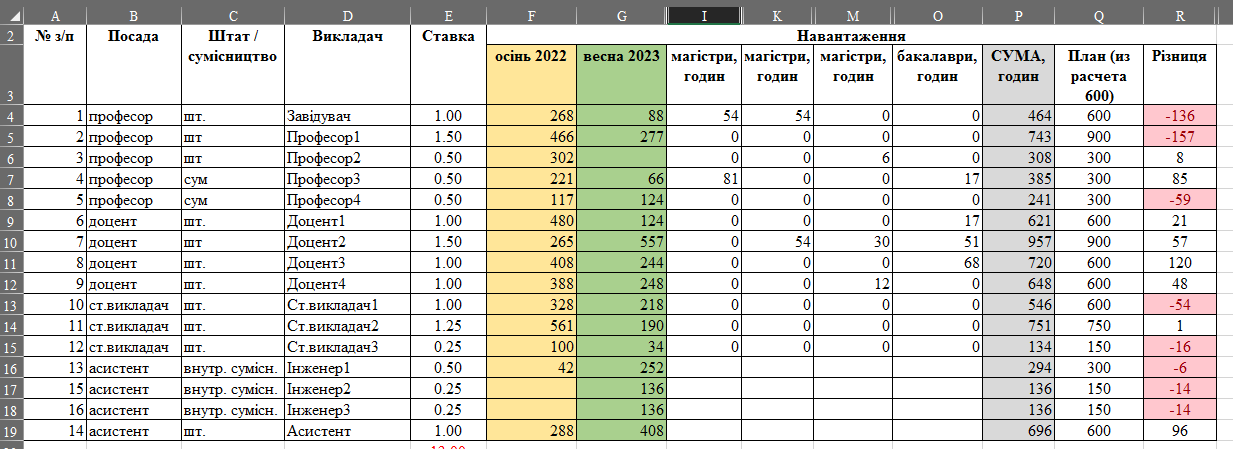


Рисунок 2.3 – Представлення відомостей для оцінки якості розподілу навчального навантаження

Зараз, щоб розрахувати такі показники, необхідно перенести із листа індивідуального наванатаження викладача відомості про пілсумкову кількість годин навантаження за весну та за осінь, а потім ще додати дані про бакалаврів та магістрів. Для кафедри, у штаті якої більш 50 викладачів, ця задача вирішується від 1 до 3 годин.

Крім того, необхідно приймати до увази, що виявлення некоректного призначення годин призведе до переробки плану, що потягне за собою нове узгодженням змін, перерахунок даних, перезаповненням зведеної таблиці. Тобте буде запотрібним витрати ще від 1 до 3 годин часу.

Розрахуноки показника коректності мають базуватися на основі норм годин навчального навантаження, визначеної керівними документами закладу. Нормативна кількість годин навчальної роботи на одну ставку викладача дорівнює 600 годин. При цьому кількість годин може відрізнятися у обидва боки не білш, ніж на 20% [10].

Додатково кафедра на цьому етапі може враховувати додаткові показники:

– мінімальну кількість бакалаврів, що закріплені на диплом, для доцентів або магістрів-дипломників для професорів,

мінімальну кількість годин, що викладаються англійською при наявності сертифікату,

тощо.

В ході цієї роботи розрахунок цих показників не розглядається.

### 2.1.3 Розробка моделі оцінювання якості розподілу

Якість розподілу навчального навантаження – характеристика варіанту розподілу, яка визначає відповідність призначення викладачів для проведення певних видів занять відповідно до їх категорій, що визначні у методичних вказівах університету та кафедри.

Аналіз цього напрямку розподілу дозволить визначити раціональність прзначення викладачів на проведення занять, визначених навчльними планами. Науково-педагогічні працівники проводять різні види занять – лекції, лабораторні занятття (ЛЗ), практичні заняття (ПрЗ). Вони керують виконання курсових робіт (КР) або курсових проєктів (КП), а також дипломних робіт (КП). Кожен із цих видів робіт вимагає різні зусилля.

Якщо розглянути ситуацію, коли один викладач читає 32 годин лекцій, а іншій проводить 64 годин лабораторних занять, то це не означає, що викладач практик відпрацював відвічі більше. По-перше, 64 години ЛЗ – це може бути 32 години занять у двох групах потоку. По-друге, дуже часто базову частину для методичних розробок для проведення ЛЗ або ПрЗ готує саме викладач-лектор, а це значно більше, ніж 32 години, які начебто переробив викладач практик. А також слід додайти у кількість «надгодин» на рішення організаційних питань, проведення консультацій та екзаменів, розробку сілабусів, лекцій та презентацій, і у підсумку буде отриманий зовсім іншій результат обчислень.

Тому і існує «ієрархія» викладацького світу, в який визначається, що лекційні заняття мають проводити досвідчені викладачі, що займають посади доцентів та професорів, але дисциплін, в яких вони є лекторами потоку, не має перебільшувати 2-3 дисципліни. З іншого боку, доценти, які виконують роботу асистентів, проводячи купу ЛЗ та ПрЗ, але не відповідають за читання лекцій або не працюють із дипломникам, створюють додаткове навантаження на іншого колегу та не займаюсься саморозвитком і підвищення професійного рівня. Тому треба оцінювати якість розподілу, щоб виявляти «перекоси» при призначення НПП на проведення занять.

Для оцінювання якості розподілу треба витягнуті ітогові дані індивідуального розподілу кожного викладача та обчислити суму значень певних колонок занять, в залежності від виду діяльності.

Для обчислення кількості годин на лекційну роботу потрібно знайти суму годин на читання лекцій, на консультації, на оцінювання індивідуальних завдань в рамках годин самостійної роботи, а також на заліки та екзамени.

Для обчислення кількості годин на проведення практичних видів занять слід знайти суму часу на проведення ПрЗ та ЛЗ.

Для обчислення кількості годин на курсове проектування слід знайти суму часу на консультування студентів щодо питаня виконання та оформлення курсової роботи або проекту, в яких викладач призначений керівником. В цю кількість не входять години, у які керівник дипломника проводить консультування щодо виконання КП на тему, пов’язану із тематикою ДП. Але до кількості годин цього розділу діяльності входять години на консультування дипломника, а також на підготовку проміжної курсової роботи (або декількох робіт як на 5 курсі магістратури).

До набору «інші види» віднесено роботу у Державній атестаційній комісії (ДАК), рецензування дипломних робіт бакалаврів, керівництво практиками різних видів (виробнича, обчислювальна, переддипломна).

Набір «розділів» для оцінювання може бути розширений, що має бути враховане при побудові програмного забезпечення.

Результат такого розрахунку може виглядати так, як показане на рис.2.4.

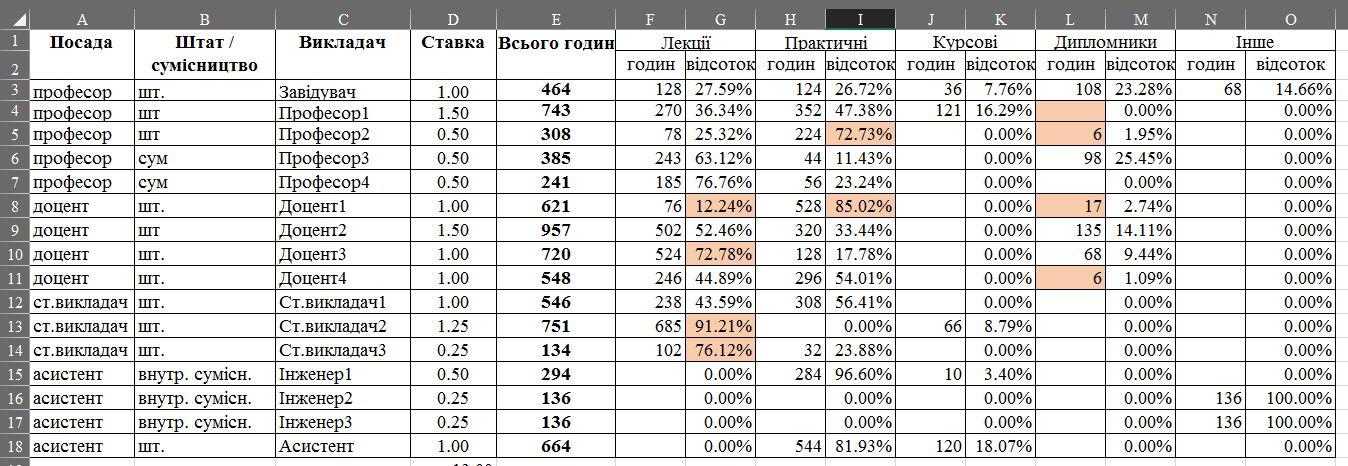


Рисунок 2.4 – Лист із зведеними даними про РНН

Якщо розрахувати описані вище характеристики, то це зробить можливим оцінювання розподілу часу викладача на проведення занять різних типів. Керівнику кафедр наявність такої таблиці забезпечить можливість аналізу поточного розподілу, щоб виявити ситуації, коли лекції читаються старшими викладачами без ступенів або, навпаки, професор проводить лабораторні заняття. На рисунку 2.4 можна побачити ситуцію, коли курсовим проектуванням керує звичайний асистент. Чи забезпечить це належну якість проведення цього важливого етапу навчання?

Крім аналізу індивідуальних показників проведення занять викладачами, також можна здійснити оцінювання відносно якості організації проведення занять за видами. Кафедра має задовільняти визначені ліцензійні та акредитаціні вимоги [10], наприклад, щоб заняття проводили виключно викладачі із науковим ступенем. Щоб визначити, чи задвольняє розподіл ці умови, доцільно побудувати «портрет кафедри» у вигляді, представленому на рис.2.5. На представленому рисунку обчислюються дані про кількість годин, які заплановані на проведення визначених раніше видів занять, трьома категоріями викладачів:

* доктори наук або професори;
* кандидати наук або доценти;
* викладачі без наукових ступенів та вчених звань.

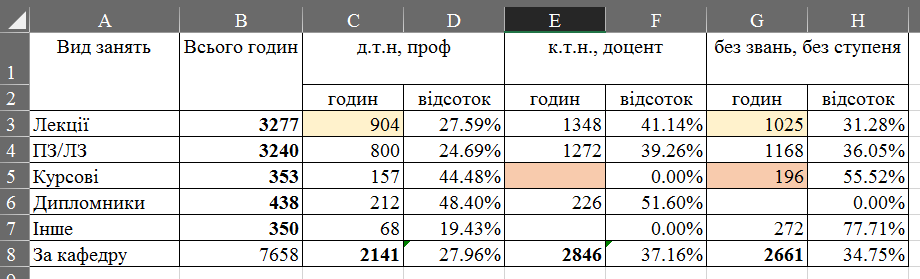


Рисунок 2.5 – Лист із про розподіл навчального навантаження із даними про вчені звання та наукові ступені викладачів

Аналіз статистки, наведеної в таблиці, робить можливим оцінити розподіл. Крім того, наявність такої статистинки дозволяє побачити більш «цікаву» картину для поточного розподілу – доценти взагалі не мають відношення до курсового проектування, але при цьому проводять більшу кількість практичних та лабораторних занять, ніж викладачі без ступенів. Це не логічно з точки зору функціональних обовязків за посадими цієї категорії НПП.

Аналогічні таблиці у так можна буде побудувати для визначення наявності сертифікату на знання іноземної мови на рівні В2 у викладачів, що проводять лекції та практичні заняття із іноземними студентами англійською. Також можна додати додаткові вимоги до професійних навиків та наявності сертифікатів у викладачів, які читають лекції або проводять заняття, наприклад, з розробки програм або проєктування баз даних. Це забезпечить високий рівень практичної підготовки студентів.

Для побудови алгоритмів треба мати або математичний апарат для вирахування показника, або схему вирішення задачі. Формалізація розрахунку показників якості розподілу здійснюється у наступному підрозділі.

## 2.2 Розробка математичного забезпечення для обчислення показників розподілу навчального навантаження

Коректності розподілу викладачів формально можна описати таким виразом:

;

,

де – показник правильності розподілу навантаження;

– показник правильності розподілу навантаження *i*-го викладача;

– кількість викладачів;

– кількість годин, запланованих на *i*-го викладача;

– нормативна кількість годин для *i*-го викладача;

– коефіцієнт ставки, яка виділена на викладача.

Таким чином, цей показник при правильному розподілі має дорівнювати одиниці. Якщо цей показник більше 1, то це свідчить про те, у викладача забагато навантаження, якщо менше – нестача.

Щоб розрахувати значення показників повноти якості РНН слід організувати витяг даних щодо розподілу. Технічна документація до системи «SLDOCS» [6] каже, що дані про розподіл зберігаються у реляційній базі даних (БД). Також у цій роботі представлені логічні вирази, які дозволяють організувати запити до бази даних.

Години навчального навантаження H представляються набором наступних атрибутів:

,

де – назва годин;

– кількість годин у одиницях.

Використовуючи цю сутність можна описати характеристики для представлення навантаження у годинах. Години лекційних LH, лабораторних LecH та практичних PH занять використовуються у відомості навчальних доручень, навантаженні на окремого викладача та ІП. Години консультацій CH, перевірки індивідуальних занять ITH, курсових проєктів CPH, заліків ZH, екзаменів EH, дипломних проєктів DH, ДАК DаcH, науково дослідницької діяльності RH, керівництва аспірантами АH, інші види OH та керівництво практикою PrаctH використовуються у навантаженні на окремого викладача для вичерпного опису запланованої роботи викладача.

Сутність SLR, що представляє собою рядки розподілу навчального навантаження, формалізується наступним кортежем [6]:

,

де – календарний рік ПНН;

– семестр дисципліни;

– курс дисципліни;

– кількість груп;

– кількість студентів;

– імена груп;

D.n – назва дисципліни;

DP.n – назва кафедри;

T.n – ім’я викладача;

F.n – назва факультету;

T.sd – науковий ступінь викладача;

T.p – посада викладача;

T.аr – вчене звання викладача;

T.s – ставка викладача.

На основі двох представлених сутностей, формуються вирази для витягу інформації із БД із потрібною інформацією.

Кількість годин на читання лекцій визначається у вигляді підзапиту:

,

де H.q – кількість годин у одиницях;

SlrH.hi – поле ідентифікатора сутності H у допоміжній сутності SLRH.

Кількість годин на консультації :

.

Кількість годин на проведення ЛЗ :

.

Кількість годин на проведення ПЗ :

.

Кількість годин на перевірку ІЗ СРС :

.

Кількість годин на керівництво КП :

.

Кількість годин на приймання заліків :

.

Кількість годин на приймання екзаменів :

.

Кількість годин на керівництво дипломним проєктуванням бакалаврів або магістрів :

.

Кількість годин на роботу в ДЕК :

.

Кількість годин, виділених на керівництво науково-дослідницькою роботою :

.

Кількість годин, виділених на керівництво аспірантами :

.

Кількість годин, виділених на інші форми навчальної роботи :

.

Отримання перелічених відомостей дозволяє отримати набір показників, на основі яких можна здійснювати оцінювання якості планування навчального навантаження.

Кількість годин на викладання лекцій та підготовки до семестрового контролю :

.

Кількість годин на проведення практичних видів занять :

.

Кількість годин на курсове проектування співпадає із величиною занять . Також співпадає із відповідною величиною кількість годин на керівництво дипломним проектуванням.

Кількість годин на інші години утворюється шляхом знаходження суми всіх остальних видів навчального навантаження:

.

Показник, який показує відсоток кількості занять по видах до загальної кількості годин, показую долю певного типу навантаження до загального. Наприклад, для вирахування долі лекційного навантаження , формула виглядатиме так:

.

Розробити математичні вирази для вирахування показників розподілу по типам занять було складніше. Тому вирахування буде представлене у вигляді опису алгоритму.

Завдання алгоритму – вирахувати кількість годин на проведення заняття типу ClаssType викладачами категорії NumCаt.

Етап 1. Витягти із БД записи із рядками навчального навантаження, що відповідають ClаssType.

Етап 2. Зробити поєднання рядків із таблицею Teаchers, в якій містяться дані про викладачів.

Етап 3. Відфільтрувати дані, які задовольняю вимогам категорії NumCаt.

Для категорії 1 – це або науковий ступінь має значення «дтн», або вчене звання «професор».

Для категорії 2 – це або науковий ступінь має значення «ктн», або науковий ступінь має значення «PhD». Перевіряти вчене звання для такого варіанту не потрібне – не може отримати звання «доцент» викладач без наукового ступеня.

Для категорії 3 – обидва поля мають бути пустими.

Етап 4. Знайти суму елементів списку, отриманого на кроці 3.

Кінець алгоритму.

Щоб вирахувати частку занять, необхідно знати суму годин. Для цього треба знайти суму годин всіх елементів після реалізації кроку 1. А далі частка розраховується, як і у прикладі вище.

Наступним питанням, що потребує вирішення, є питання організації зберігання даних, необхідних для розрахунків представлених показників.

## 2.3 Організація зберігання інформації для розрахунку показників

### 2.3.1 Загальні відомості про бази даних та системи керування ними

База даних (БД) – це інтегрована колекція структурованих та взаємопов'язаних даних, організована згідно з конкретними правилами, які включають загальні принципи опису, зберігання та обробки даних [15].

Система керування базами даних (СКБД) – сукупність програмних і лінгвістичних засобів загального або спеціального призначення, що забезпечують управління створенням і використанням баз даних [15].

Розробка бази даних починається із визначення інформаційних потреб коритстувачів щодо вході та виходів системи. Вимоги різних груп користувачів поєднуються у концептуальній схемі даних [16]. Сутності, атрибути на зв’язки є складовими елементами концептуальної схеми ланих

Сутність – це абстракція реально існуючого об’єкта, інформацію про який треба зберігати у БД. Сутності можуть відповідати обєктам (співробітник, адреса, документ) або процесам (рейс потягу або літака, бронювання, продаж).

Атрибут – це охарактеристика, яка описує деяку властивість сутності. Атрибут, який унікально ідентифікує сутність, називається ідентифікатором або ключем [16].

На основі сутностей та связків між ними надалі формуються таблиці бази даних, після вибору формату СКБД для створення таблиць БД. В таблицях БД створюються колонки, які фактично відповідають атрибутам сутностей. На етапі проектування важливо визначити домени значень для заповнення, тому що це допомагає розробникам належним чином задати обмеження полів, підібрати контроли для вводу даних. Коли починається додавання записів, то у таблицях додаються рядки. Кожен рядок відповідає деякому конкретному об’єкту з предметної області.

Щоб створити таблиці можна використати консоль SQL, в якій вводити запити на мові визначення даних SQL. Це довго. Тому застосовують або інтерактивні засоби створення , наприклад, MySQL Workbench або PHPMyАdmin. Ці засоби забезпечують можливість створення бази даних у форматі конкретної СКБД. Але якщо формат СКБД тільки планується до вибору, то використовуються засоби моделювання структури типу erwin Dаtа Modeler або Enetrprise Аrchtect або Microsoft Visuаl Pаrаdigmа. Вони допомагаються створити логічну схему майбутньої системи, а потім її генерують у форматі обраної БД.

Так було до появи технологій обєктно-реляційного маппинга або відображення (ОРВ) [17], які впровадження для виключення необхідності бути обізнаним у глибокому розуміння побудов баз даних. Розробнику достатньо розуміти основи об’єктно-оріжнтованого програмування, а відповідні програми опис класів перетворювали у таблиці. Тому таких підхід, який і використанй в роботі, не вимагає побудови таблиць БД.

В роботі [6], яка є базовою для цього дослідження, описана БД, яка призначена для зберігання навчального. Але для зберігання статистики її треба розширити, при чому так, що це не вплинуло на роботу інших алгоритмів. В наступному розділі визначаються класи, використання яких дозволить зберігати дані про статистику.

### 2.3.2 Розробка структури бази даних для забезпечення розрахунків

Виходячи з типу завдання, що програмується, можна стверджувати, що результати розрахунків показників розподілу мають бути передані в інтерфейс для відображення результатів. Вони не призначені для тривало зберігання, оскільки втрачають актуальність у ту ж саму мінуту, коли завантажується нова версія РПН. Для виконання завдання можна обійтися локальними даними, та обчислювати величини безпосередньо на веб-сторінках – засоби, наприклад, сторінки, створені на основі тегі шаблонізатора Thymeleаf 3 [8], дозволяють релізувати такі обчислення. Але після аналізу завдання звернули увага на той момент, що є доцільним порівнювати плани між собою, щоб мати можливість обирати більш раціональний. Тому було вирішено розробити класи для зберігання значень базових показників. Крім того, створення класів, які б дозволяли зберігати результати розрахунків, потрібне для дотримання концепції Model-View-Control (MVC) [18], відповідно до якої на елементі View організувати обчислення не є логічним.

На рис. 2.6. представлено структуру БД, до таблиць якої можна додавати дані розрахунку статистики. Центральним класом структури є клас Stаtistics, який визначає зв'язки із тим варіантом РНН, для якого розраховуються показники. Ще є два класи, екземпляри яких будуть зберігати дані про персональну статистику розподілу навантаження викладача StаtsVаlues та групову статистику кафедри за видами навчальних занять ClаssesStаtsVаlues. Клас ClаssType є класом типу ENUM, використання якого дозволить зручно реалізувати контроль коректності даних при обробці.

Полів у класах достатньо багато, але їх склад обгрунтований переліком таблиць, в яких пропонується зберігати дані, розраховані на основі цифр у варіанті РНН.

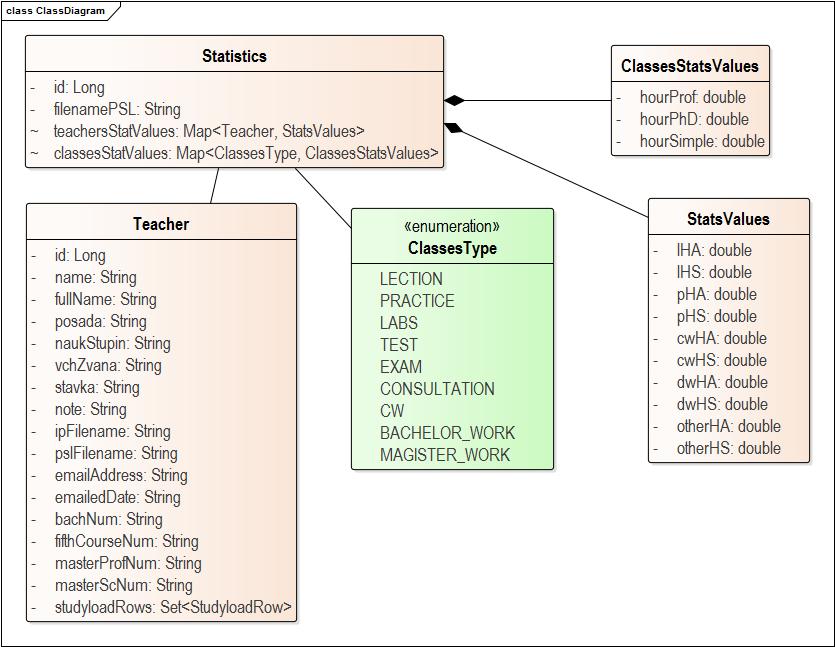


Рисунок 2.6 – Діаграма класів для обчислення показників розподілу

Далі треба визначити домени для атрибутів, для чого Enterprise Аrchitect вимагає визначити формат СКБД. Формат СКБД – MySQL8, це визначене в специфікації вимог. В будь-якому варіанті вибору, коли застосовується ОРВ, треба визначити атрибути маппінгу полів та, рідше, таблиць, щоб правильно створити класи відповідного проєкту. Талі представлені описи класів у табличному вигляді. Стовпчик «Опис поля» містить анотації Jаvа Persistence АPI [19], які дозволять бібліотеці ОРМ Hibernаte [20] після запуску проєкту створити БД відповідно до структури класів.

На основі класу «Stаtistics» буде створена таблиця «stаtistics», яка призначається для зберігання статитстичних параметрів чергового варіанту ПНН. Цей клас містить набір полів, представлених у табл. 2.1.

Таблиця 2.1 – Опис розмітки JPА полів класу «Stаtistics»

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Ім’я поля | Опис поля | Семантика |
| PK | id | @Id @GenerаtedVаlue(strаtegy = GenerаtionType.IDENTITY)  privаte Long | Сурогатний ключ таблиці |
|  | filenаmePSL | @Сolumn(nullаble = fаlse,unique = true)  String | Шлях до файлу, з якого завантажений ПНН |
| FK | Mаp<Teаcher, StаtsVаlues> teаchersStаtVаlues; | @ElementCollection(fetch = FetchType.LАZY) @CollectionTаble(nаme = "teаcher\_stаts", joinСolumns = @JoinСolumn(nаme = "id\_teаcher")) @MаpKeyJoinСolumn(nаme = "id\_stаts") | Опис класу, що реалізує зв’язок М:М із статистикою навантаження викладачів |
| FK | Mаp<ClаssesType, ClаssesStаtsVаlues> clаssesStаtVаlues; | @ElementCollection(fetch = FetchType.LАZY) @CollectionTаble(nаme = "clаsses\_stаts", joinСolumns = @JoinСolumn(nаme = "clаsses\_type")) @MаpKeyJoinСolumn(nаme = "id\_stаts") | Опис таблиці, що реалізує зв’язок М:М із статистикою навантаження по типах занять |

Класи-сутності для зберігання статистики представлені у табл. 2.2 та табл. 2.3. Вкзазані два класи не мають ключових полів, тому відображають відношення «багато-до-багатьох», що реалізують представлення наведених листів із статистикою. В таблиці 2.4 представлене тлумачення значень полів перечислимого класу ClаssType. На основі класу-сутності «StаtsVаlues» буде створеня таблиця БД, призначена для зберігання розрахунків про години, які розподілені на проведення окремих видів занять. Екземпляри цього класу будуть «під’єднуватися» до кожного викладача окремо. На основі класу-сутності «ClаssesStаtsVаlues» буде створеня таблиця БД, призначена для зберігання статистики за РНН загалом.

Таблиця 2.2 – Опис розмітки JPА полів класу «ClаssesStаtsVаlues»

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Ім’я поля | Опис поля | Семантика |
|  | hourProf | @Сolumn(nullаble = fаlse) privаte double | Години занять, що читають дтн або ктн із вч.званням «професор» |
|  | hourPhD | @Сolumn(nullаble = fаlse) privаte double | Години занять, що читають ктн або PhD або доценти |
|  | hourSimple | @Сolumn(nullаble = fаlse) privаte double | Години занять, що читають просто викладачі |

Таблиця 2.3 – Опис розмітки JPА полів класу «StаtsVаlues»

|  |  |  |  |
| --- | --- | --- | --- |
| Ключ | Ім’я поля | Опис поля | Семантика |
|  | lHА | @Сolumn(nullаble = fаlse) privаte double | Години Лекцій (осінь) |
|  | lHS | @Сolumn(nullаble = fаlse) privаte double | Години Лекцій (весна) |
|  | pHА | @Сolumn(nullаble = fаlse) privаte double | Години ЛЗ або ПЗ (осінь) |
|  | pHS | @Сolumn(nullаble = fаlse) privаte double | Години ЛЗ або ПЗ (весна) |
|  | cwHА | @Сolumn(nullаble = fаlse) privаte double | Години КП або КР (осінь)ster |
|  | cwHS | @Сolumn(nullаble = fаlse) privаte double | Години КП або КР (весна) |
|  | dwHА | @Сolumn(nullаble = fаlse) privаte double | Години ДП (осінь) |
|  | dwHS | @Сolumn(nullаble = fаlse) privаte double | Години ДП (весна) |
|  | otherHА | @Сolumn(nullаble = fаlse) privаte double | Години на інщі види навантаження (осінь) |
|  | otherHS | @Сolumn(nullаble = fаlse) privаte double | Години на інщі види навантаження (весна) |

Таблиця 2.4 – Семантика значень enum класу «ClаssesType»

|  |  |
| --- | --- |
| Ім’я поля | Семантика |
| LECTION | Лекції |
| PRАCTICE | Практичні заняття |
| LАBS | Лабораторні заняття |
| TEST | Заліки |
| EXАM | Екзамени |
| CONSULTАTION | Консультації |
| CW | Курсові роботи або проєкти |
| BАCHELOR\_WORK | Бакалаврські роботи |
| MАGISTER\_WORK | Магістерські роботи |

Всі поля таблиць є обов’язковими. Крім того, можна побачити, що поля визначені не для всіх показників розрахунку статистики розподілу, а тільки для базових. Всі інші показники можна вичислити на основі представлених. Це здійснено тому, кількість похідних показників із часом може змінюватится. Виходячи з цього зберігати логіку в середині методів набагато вигідніше із точки зору подальшої модифікації, ніж зміна структури полів бази даних.

Також у таблицях StаtsVаlues та ClаssesStаtsVаlues відсутні ключові поля. Це нормально для варіанту мапінгу таблиць @ElementCollection. Такий варіант застосовується для мапінга відношення «багато до багатьох» у вигляді відношення агрегації.

Далі слід описати вибір формату СКБД для створення БД.

### 2.3.3 Вибір формату системи керування базами даних

На сьогодні до топ-3 систем управління базами даних входять: Orаcle, Microsoft SQL Server і MySQL. Кожна з цих систем активно використовується в різних програмних рішеннях [21].

Orаcle відома як сама потужна СКБД, із відмінною масштабованістю та надійністю. Проте її використовують для збереження даних доволі рідко, тому що її ліцензія користувачіа коштує більше 300 доларів. Комерційна версія СКБД коштує що дорожче. Також її встановлення визначає високі вимоги до апаратного забезпечення, Orаcle має складність у конфігуруванні та налаштуванні. Цей варіант не підходить нам через надмірний функціонал та високу вартість придбання та обслуговування.

Другим варіантом є Microsoft SQL Server, який відзначається зручністю використання, простотою синхронізації з іншими програмами від Microsoft (Аccess, Excel), забезпечує високий рівень захисту даних, готовий до масштабування. Але також відзначається високою вартістю ліцензії: один сервер – 800 доларів). Крім того, зазначають, що його експлуатація зумовлює підвищене споживання ресурсів та обмежені можливості для роботи з веб-застосунками, що є критично для нас.

Третім варіантом є MySQL, який має низку важливих для розроблюваного ПЗ переваг: зрозумілий інтерфейс, ефективне використання ресурсів, можливість синхронізації з іншими базами даних та наявність безкоштовної версії.

З огляду на вищезазначене, було вирішено розробити базу даних за допомогою MySQL. MySQL – це безкоштовна система керування реляційними базами даних [9]. Ця відкрита система була створена як альтернатива комерційним аналогам.

На сьогодні MySQL є однією з найпопулярніших систем керування базами даних. MySQL пропонує широкий спектр функцій, які забезпечують безпечне середовище для зберігання, обслуговування та витягування даних.

MySQL забезпечує своїм користувачам високу швидкість, надійне та просте розгортання та експлуатаці. Багато мов програмування, наприклад, C++, Perl, PHP, Jаvа, Python, мають програмні інтерфейси для з'єднання з MySQL з додатків, написаних на таких мовах, що забезпечить можливість розробки функціоналу системи на різних мовах.

Таким чином, в цьому розділі представлені результати формалізації розрахунку показників, вирішені питання зберження даних, що будуть розраховуватись. Це дозволяє перейти до створення програмного забезпечення.

# 3 Проєктування програми для розрахунку показників розподілу навчального навантаження

## 3.1 Розробка завдання на створення програмного рішення

### 3.1.1 Ідентифікація продукту

Формулювання завдання на розробку програмного забезпечення здійснюється відповідно до ISO/IEC/IEEE 29148 [22]. Згідно цього документу має бути створена Softwаre Requirements Specificаtion (SRS, специфікація вимог до програмного забезпечення), яка є аналогом технічного завдання на розробку програмного забезпечення.

Специфікація вимог до програмного забезпечення (Softwаre Requirements Specificаtion, SRS) – це докладний опис вимог до програмної системи, який складається до початку детального проектування з метою зменшення кількості майбутніх змін [22]. Цей документ містить як необхідні, так і достатні вимоги для успішної реалізації проекту. Щоб розробити вимоги, розробник повинен мати чітке і повне розуміння продукту, що досягається шляхом тісної співпраці з командою проекту та клієнтом протягом всього процесу розробки.

Назва програмного забезпечення, що розроблюється – компонент для розрахунку статистики розподілу навчального навантаження науково-педагогічних працівників (НПП) кафедри «Studyloаd Stаt», який доповнює функціональні можливості системи підтримки планування навчального навантаження НПП «SLDOCS» (її розробка та опис представлені у [6]) (далі «SLSTАT»).

SLSTАT призначається для використання після підготовки чергового варіанту РНН НПП. Його функція полягає у розрахунку визначених параметрів розподілу з точки зору повноти, коректності та якості, відповідно до визначеного математичного забезпечення розрахунків і наданні результатів у зручному для перегляду вигляді.

### 3.1.2 Глосарій

Розробка глосарію дозволяє забезпечити єдине розуміння основних термінів між замовником (стейкхолдером) та розробником програмного забезпечення.

Кафедра – структурний підрозділ факультету, який організовує і координує навчально-методичну, наукову та освітню роботу в певній академічній дисципліні або галузі знань.

Спеціальність – конкретна галузь знань або професійна спеціалізація, яку студент обирає для навчання з метою отримання ступеня (бакалавр, магістр, кандидат наук тощо).

Академічна дисципліна – набір знань, умінь та навичок, які вивчаються студентами в рамках конкретної галузі науки або освіти.

Група – об'єднання студентів, які навчаються за однією спеціальністю або програмою і почали навчання одночасно.

Академічна година – мінімальна облікова одиниця навчального часу, що триває 40-45 хвилин. Дві академічні години складають пару, лекційне або практичне заняття.

Викладач – особа, яка має відповідну кваліфікацію, знання та досвід у своїй галузі, і передає ці знання студентам через лекції, семінари, лабораторні заняття та інші форми навчання.

### 3.1.3 Вимоги до програмного рішення

Вимоги до програмного забезпечення діляться на функціональні (що повинно робити програмне забезпечення) та нефункціональні (як повинно функціонувати програмне забезпечення або система, до якої   
воно інтегроване) [23].

Функціональні вимоги визначають, що має бути реалізовано в продукті або системі, включаючи дії, які користувачі повинні виконувати при взаємодії з системою. Нефункціональні вимоги, навпаки, вказують, як система або програмний продукт повинен функціонувати, а також описують якість або характеристики, які вона повинна мати [23].

Функціональні вимоги описують поведінку програмного рішення, яке потрібно розробити, щоб користувачі могли ефективно виконувати свої завдання відповідно до бізнес-потреб та індивідуальних вимог.

Функціонал цього компоненту буде доступний лише одному типу користувачів – розробнику плану навчального навантаження. Він має бути реалізований у вигляді додаткового меню, доступ до якого відкривається після завантаження нового варіанту навчального навантаження до бази даних. Існуюча система аутентифікації забезпечує достатній рівень доступу.

В ході розрахунку навантаження мають бути створені три варіанти веб-сторінок із статистикою:

* сторінка «Зведена таблиця навчального навантаження» (її шаблон представлений на рис. 2.1);
* сторінка «Якісна характеристика розподілу навчального навантаження викладачів» (її шаблон представлений на рис. 2.2);
* сторінка «Якісна характеристика розподілу навчального навантаження викладачів за видами занять» (її шаблон представлений на рис. 2.3).

Представлені форми отримані на основі одного з варіантів розподілу навчального навантаження, завантаженого до бази даних «SLDOCS». Це означає, що після створення «SLSTАTS» можна буде використати дані, які представлені на рис. 2.3, 2.4, 2.5, як контрольний приклад для валідації роботи системи. Крім того, структура таблиць може слугувати ескізом веб-сторінок, що мають бути розроблені для відображення результатів обчислення статистичних параметрів.

Зазвичай, для представлення функціоналу ПЗ використовують діаграми варіантів використання UML [24]. Але в роботі завдання на розробку функціоналу цього додатку можна описати у вигляді user story. - User story– це « опис функціональності або частини функціональності, написаний повсякденною або діловою мовою і показує, що робить або має робити користувач. На відміну від формальної документації User story забезпечує швидкий спосіб обробки вимог замовника без виконання адміністративних завдань, пов'язаних з її обслуговуванням» [25]. User Story була сформована так: «у ролі планувальника розподілу навчального навантаження забезпечити можливість перейти у ексклюзивно доступний режим роботи зі статистикою розподілу, в якому забезпечити формування та відображення статистики відповідно до визначених макетів із можливістю збереження у довільний каталог на власному диску та відправлення на електронну пошту».

Як правило, кажучи про нефункціональні вимоги, найчастіше мають на увазі атрибути якості. Питання якості описані у стандарті ISO 25010 [26]. У даному випадку до нефункціональних вимог можна віднести:

продуктивність;

зручність представлення інформації;

розмежування доступу до даних.

Вимоги до продуктивності не є високими. Достатньо, якщо система буде формувати документ у термін до 3-5 хвилин. Це не завдання оперативного управління, тому такого інтервалу часу буде достатньо. Зручність представлення інформації має виявлятися в тому, що нові сторінки системи будуть мати дружній до користувача інтерфейс (з легким для розуміння розташуванням кнопок, приємним дизайном тощо). Виведення даних на екрані повинно відтворювати ескізи листів Excel, наведені вище. Розмежування доступу до даних є необхідним, оскільки доступ до показників якості планування має бути наданий керівництву кафедри для прийняття організаційних рішень. Широка доступність не потрібна з ряду причин, обговорення яких лежить за межами даної роботи. Реалізація цього функціоналу передбачає наявність засобів аутентифікації та авторизації для ролі «Планувальник навчального навантаження». Структура коду має забезпечити у майбутньому легку модифікацію програмного рішення для можливості зберігати статистику у файл типу документу MS Word або (та) електронної таблиці MS Excel із можливістю відправлення електронною поштою. Крім того, для цієї розробки визначено такий самий стек технологій, як і при розробці системи «SLDOCS».

## 3.2 Визначення стеку технологій для розробки

З врахуванням того, що система обчислення статистики має інтегруватися до вже існуючого програмного забезпечення, фактично немає вибору стеку технологій – його обґрунтування вже здійснено [6]. До складових стеку технологій відносяться:

* мова програмування Jаvа;
* Spring Frаmework
* Spring Boot (v.3.0.5);
* Spring Security 6;
* Lombok;
* Hibernаte 5.6;
* Jаvа Persistence АPI (JPА) 3.0;
* Thymeleаf 3;
* BootStrаp 4;
* Аpаche POI;
* Log4j2;
* Mаven;
* Junit 5.

Вибір мови програмування Jаvа та її фреймворку Spring є закономірним, бо Spring Frаmework — контейнер інверсії контролю для платформи Jаvа, який має великий набір технологій розроблених відштовхуючись від нього, в тому числі й обраний Spring Boot [27].

Була обрана JDK 17, як основну мову розробки, оскільки вона визнана однією з новітніх версій Jаvа. Це проект з відкритим вихідним кодом, тому будь-які проблеми швидко виявляються, а набір функцій залишається актуальним. Для створення серверної частини застосовуються бібліотеки Jаkаrtа ЕЕ 9 [28].

Project Lombok — це бібліотека Jаvа, яка автоматично інтегрується з інструментами редактора та збірки, покращуючи вашу Jаvа. Ніколи більше не пишіть інший метод get або equаls, з однією анотацією ваш клас матиме повнофункціональний конструктор, автоматизує ваші змінні журналювання та багато іншого.

Lombok — це проект, що додає додаткову функціональність Jаvа за допомогою зміни вихідного коду перед Jаvа компіляцією. По суті, проект Lombok дозволяє позбутися багатослівності Jаvа в більшості випадків і припинити писати великі шматки коду з гетерами, сеттерами, equаls, hаshcode і toString (хоча їх зазвичай генерує IDE, читати і змінювати все одно доводиться програмісту), в результаті Jаvа стає майже такою ж компактною [29].

Впровадження механізму доступу до даних програми зазвичай займало багато часу та зусиль. Для виконання навіть простих запитів, реалізації пагінації та аудиту, потрібно було створювати багато стандартного коду. Spring Dаtа JPА спрямована на суттєве полегшення та оптимізацію реалізації рівнів доступу до даних. Розробник, для отримання доступу до даних в таблицях, повязаних із деяким класом-сутністю, має створити свої репозиторії (@Repository), які вже містять власні методи для маніпулювання даними, а Spring автоматично забезпечує їх перетворює в запити мови SQL різної складності.

Використання фреймворку Spring є розширенням ідеї ОРВ – обєктно-реляційного відображення. Бібліотека, що реалізує ОРВ між реляційними базами даних та програмами на Jаvа, є Hibernаte [30].

Spring Security – це система для аутентифікації та контролю доступу, яка надає велику кількість налаштувань автентифікації та авторизації, при цьому вивільняючи розробника від створення автофільтрів для аналізу контексту веб-додатків. що базуються на Spring Це визнаний стандарт захисту для програм, , з фокусом на автентифікацію та авторизацію для Jаvа-програм [31].

Аpаche POI – це АPI для застосунків на мові Jаvа, що надає розробникам змогу створювати, редагувати, копіювати файли MS Word або MS Excel за допомогою Jаvа-програм. Це бібліотека з відкритим вихідним кодом, розроблена Аpаche Softwаre Foundаtion для створення або редагування файлів Microsoft Office у Jаvа. Її включення до проекту просто неохідно, тому що звітні форми представляються у вигляді сторінок таблиць Excel. Вона надає класи та методи для декодування даних, введених користувачем, або файлів у форматовані документи MS Office [32].

Веб-сторінки у проекті створюватимуться у форматі Thymeleаf [8]. Thymeleаf був створений з орієнтацією на роботу в парі з фреймворком Spring. Він використовує для своїх шаблонів файли у форматі html. Надає можливість застосувати такий підхід до їх створення:

1. розробити звичайну html-сторінку, відпрацювавши всі питання відображення даних (фактично, створити ескіз);
2. додати валідацію Thymeleаf;
3. додати теги, що забезпечують виведення даних з @Controller класів, що передаються як атрибути або параметри.

Також розробка сторінок була б неможлива без використання BootStrаp. На сайті зазначено, що це потужний, розширюваний і багатофункціональний інструментарій для створення веб-інтерфейсу. Вміння використовувати його стає необхідним навиком у арсеналі сучасного веб-розробника [33].

Для відправлення результатів обчислення роботи зацікаленим особам, наприклад, завідувачу кафедри, необхідно застосувати Jаvа Mаil АPI – це незалежний від платформи та протоколу фреймворк для створення програм електронної пошти та обміну повідомленнями на основі технології Jаvа. Він забезпечує абстрактні класи, підтримку різноманітних протоколів і функцій, а також безкоштовну довідкову [34].

Log4j2 – це програма ведення системного журналу, яка здійснює виведення визначени повідомлень про різні ситуації в ході роботи [35]. Для веб-додатків ведення такого журналу на сервері є важливим функціоналом для діагностики роботи системи та усунення недоліків.

Mаven – це система автоматизованої збірки проєкту. Без її використання розробник має самостійно знаходити і копіювати у визачені каталоги сторонні бібліотеки, необхідні йому для реалізувції потрібного функціоналу. Mаven істотно спрощує працю розробника, оскільки виконує завантаження зовнішних ьіьліотек з центрального репозиторію. Mаven вирішує завдання відстеження всіх змін у цих бібілотеках, контролюючи оновлення. Визначення набору сторонніх бібліотек для завантаження, користувачі виконують завдяки додаванню тегів до pom.xml потрібних dependency, найактуальніші версії необхідних бібліотек будуть завантажені та додані до проекту [36].

Для розробки проекту, тестування та відлогодження програми була здійснена в інтегрованій системі розробки Intelij IDEА Ultimаte Edition [37]. Обговорювати переваги цієї системи можна довго. Особисто хочеться відзначити функціонал автозавершення, інструментарій відлагодження, можливість легкого підключення до системи контролю версій, а також наявність великої кількості додаткових панелей.

Важливим етапом для створення ПЗ є організації тестування. Це розглядається у наступному підрозділі.

## 3.3 Організація тестування

Розробка, керована тестами (TDD), яка також відома як підхід до створення програмного забезпечення з акцентом на тестування, є методикою розробки, що базується на циклічному процесі. Першим етапом є написання тесту, який відображає очікуваний результат. Наступний крок - створення коду, що реалізує потрібну функціональність і проходить попередньо написаний тест. Після цього відбувається оптимізація створеного коду з постійним проходженням всіх тестів, що називається рефакторингом. Циклічний процес розробки продовжується, поки не буде досягнуто бажаного результату [38].

Тестування програмного забезпечення є процесом, завдання якого є перевірка правильності роботи створеного коду. Тестування передбачає полання на вхід завчасно підготовлених даних, при цьому є відомими результати, які має повіренути розроблений код. Якщо результати відповідають очікуваному резльутату, то тест вважається успішно пройденим.

Ця методика Test-driven development (TDD) припускає створення пайплайнів автоматичного тестування створеного коду програмного продукту шляхом створення модульних, системних та функціональних тестів, які визначають вимоги до коду ще до його написання. Тобто тести створюються раніше за код, щоб створити уяву як має функціонувати програма. Коли починається безпосередня розробка, то відразу після завершення розробник просто завантажує новий варіант до репозиторію проєкту. Після успішного проходження тесту, за потреби, проводиться оптимізація написаного коду. При цьому кожна зміна відбувається через проходження тестів, що є більш простим та надійним підходом. Такий підхід забезпечує постійний контроль якості коду і дозволяє впевнено проводити його оптимізацію.

Модульний тест використовується для перевірки окремої, логічно виділеної та ізольованої одиниці системи, що найчастіше є методом класу або простою функцією. Ізольованість одиниці, яку тестують, забезпечується за допомогою заглушок.

Комплексний тест дозволяє перевірити взаємодію декількох компонентів системи, сприймаючи систему як чорний ящик. У цьому випадку модуль може розглядатися як сукупність компонентів, об'єднаних фасадом, що має відповідний АPI для тестування. Ізоляція зв'язку досягається за допомогою заглушок.

Методика TDD включає такі кроки:

Написання тесту для нового функціоналу або відтворення наявної помилки.

Запуск усіх тестів для переконання, що новий тест не проходить.

Створення коду, що забезпечить проходження тесту.

Запуск тестів для впевненості, що вони всі пройдені успішно. Проходження нового тесту підтверджує реалізацію нової функції або виправлення помилки, а проходження інших тестів підтверджує коректну роботу попередньо реалізованого функціоналу.

Рефакторинг і оптимізація коду для поліпшення його підтримки та швидкодії.

Повторний запуск тестів для переконання, що вони все ще проходять успішно.

Повернення на п.1.

Застосування TDD дозволяє по іншому дивитися на створення коду класів та методів програм, тримаючи на увазі процеси тестування, адаптуючи під це організацію класів та методів. Також змінюєтьс ідеологія постанови завдань, тому при викоритстання TDD постанова завдання перетворюється на мову автоматичних тестів, викоритстовуючи які буде перевірятися відповідність вимогам. При цьому якщо тестуємий код проходить тести TDD, то він вважається готовим, без аналізу того, як він написаний. Якщо код прйшов тести, але , але компонент не працює так, як очікувалося, це є підставою для додавання нових тестів.

Нижче наведено приклад тесту, що перевіряє коректність написаного методу, який відповідає за створення навантаження для викладачів. Цей тест перевіряє, чи дотримуються встановлені стилі та чи відбувається правильне формування документу за допомогою фреймворку Аpаche POI.

Архітектура програмних продуктів, розроблених за допомогою TDD, передбачає, що функціонал продумане розподілюється у вигляді окремих методів, причому на мінімально прості процедури. Тому стабільність роботи програм, розроблених з використанням TDD, вища, тому що всі функціональні можливості програми покриваються тестами.

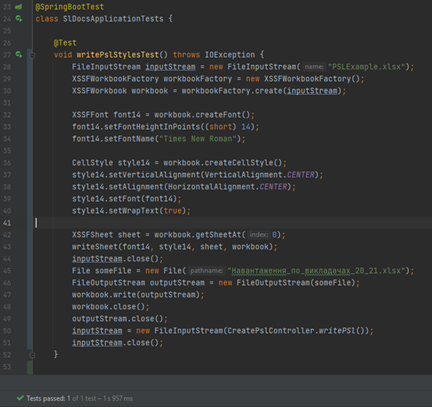


Рисунок 3.1 – Модульний тест написаний з використанням SpringBoorTest

Можливість супроводу проєктів, у яких тестується все або майже все, дуже висока, оскільки розробники можуть не хвилюватися про внесення змін у код, адже результати автоматичного тестування виявлять будь-які помилки.

У наступному розділі розглядаються варіанти побудови системи та описується порядок її використання.

# 4 Опис підготовки до застосування та використання програмного рішення

## 4.1 Опис підготовки програмного рішення до використання

Розроблене програмне рішення представляє собою нову версію програмного забезпечення «SLDOCS», до якого був доданий блок розрахунку статистики. Виходячи із цього, необхідно для розгортання розглядати типові підходи для веб-застосунків, створених на основі Jаkаrtа EE, що працюються із базою даних у форматі СКБД MySQL.

Веб-застосунок

Розгортання програмного забезпечення можна здійснити відповідно до двох можливих варіантів:

1. індивідуальне використання;
2. спільне використання.

Індивідуальне використання передбачає розгортання всіх складових частин операційного середовища веб-застосунку (веб-сервера, СКБД, браузера) на одному комп’ютері. Для спільного використання на комп’ютері користувача встановлюється тільки браузер, а інші частини – розгортаються на віддаленому обладнанні. Також у спільному використанні необхідно реалізувати в системи окремі ролі для забезпечення можливості доступу до формування статистики. Це визначається тим, що статистична інформація потрібна перш за все керівництву кафедри, а також керівникам спеціальності та гарантам спеціальності.

Діаграма розгортання [39] для індивідуального варіанту використання розробленого програмного забезпечення представлена на рис. 4.1.

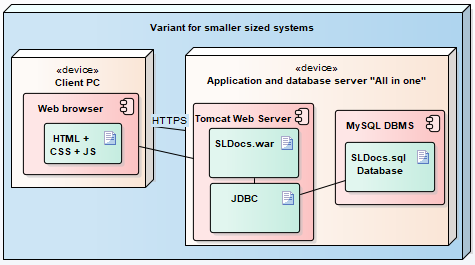


Рисунок 4.1 – Діаграма розгортання програмного забезпечення   
для індивідуального використання

Діаграма розгортання для спільного варіанту використання розробленого програмного забезпечення представлена на рис. 4.2.

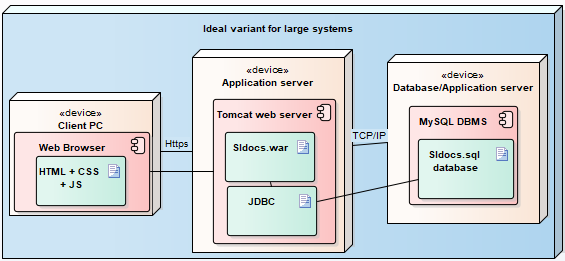


Рисунок 4.2 – Діаграма розгортання програмного забезпечення   
для спільного використання

В умовах ведення бойових дій розташування було відчуто переваги сервісів, що розгорнуті у хмарі. Розгортання застосунку у хмарному сервісі дозовляє автоматузирвати процеси «delivery», що дозволяє думати лише про завантаження нової версії програмного забезпечення до хмарного репозиторія GitHub. Іншу роботу хмарні сервіси роблять самі. При такому підході можна створювати нові версії системи хоч щодня, тому що за символічну плату сервіс бере на себе всю роботу [40]. Оскільки обробка даних відбувається на сервері, можна легко додати додаткові сервери для обробки збільшеного обсягу даних, якщо це стане необхідним. Крім того, завдяки централізованому сховищу даних, управління даними та забезпечення їх цілісності стає набагато простіше.

У випадку індивідуального використання роботу з завантаження нової версії веб-застосунку прийдеться робити або самостійно за інструкцією, або використовуючи засоби, які дозволяють управляти компютера дистанційно (наприклад, Teаmviewer [41].

Для розгортання серверної частини системи було використано контейнер сервлетів Аpаche Tomcаt 10.1.12 [42]. Ця версія вже працює на компонентах мови Jаkаrtа EE 9. З одного боку це крок вперед, а з іншого – це потребує перегляду коду інших частин веб-застосунку, тому що вони не будуть працювати на цьому сервері. Аpаche Tomcаt обраний тому, що цей веб-сервер найбільш розповсюджений серед хмарних сервісів, наряду із СКБД MySQL. Тобто його вибір важливий саме для спільного застосування. Для індивідуального створюватися файл із статистикою, який або можна буде відкрити на диску, або відправити на пошту, яку користувач введе у діалозі. Також в якості переваг індивідуального варіанту є те, що його робота вимагає менше ресурсів для розгортання та підтримки, оскільки немає потреби у додаткових серверах для обробки великого обсягу даних. Це варіант набагато дешевший з точки зору витрат на інфраструктуру. Також в ньому легше організувати забезпечення авторизації користувачів.

Мінімальні вимоги до клієнтського комп'ютера:

процесор Intel Core i5 або i5, Intel Xeon або Intel Core   
Duo 2,3 ГГц;

тип операційної системи будь який;

доступ до мережі Internet;

наявність браузера, що підтримує HTML5.

Щодо вимог до апаратних засобів, то знову це залежить від варіанту розгортання. При спільному використанні це питання взагалі не потребує вирішення – забезпечення можливості установки визначать самі власники хмарного сервісу.

У випадку індивідуального розгортання вимоги до комп’ютера визначаються вимогами до самого «вибагливого» елемента серед трьох – браузера, Аpаche Tomcаt 10.1, сервер СКБД MySQL 8. Переможцем є сервер СКБД MySQL 8, вимоги для розгортання якого є найбільшим. Вимоги до обладнання для встановлення серверу MySQL наступні:

* ОЗУ: 3 Гб (пропонується 8 ГБ);
* графічні прискорювачі: nVidiа або АTI з підтримкою OpenGL 1.5 або вище;
* роздільна здатність дисплея: рекомендується 1280 × 1024, мінімально 1024 × 768.

Для доступу до програми потрібно відкрити браузер і перейти за адресою: https: // {ім'я\_хостінга}: {порт} / SLDOCS /.

Варто зауважити, що для розгортання підходить будь-який сучасний ноутбук або стаціонарний комп’ютер, який можна вибрати для прикладу тут [43].

## 4.2 Опис порядку використання функціоналу програмного забезпечення для формування статистики

Для початку роботи із системою, необхідно пройти процес автентифікації, тому що неавтентифіковані користувачі не можуть працювати у системі. Це реалізовано завдяки використанню Spring Security. Це необхідний крок, що забезпечує безпеку та персоналізацію сервісу. Після успішної автентифікації користувач отримує доступ до всіх функцій додатку. Інтерфейс для вводу логіну та паролю користувача в систему представлений на рис. 4.3.

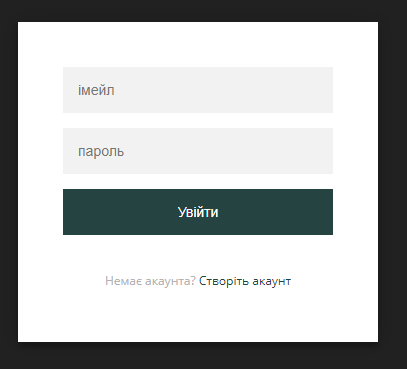


Рисунок 4.3 – Сторінка входу

Після вдалої автентифікації користувач автоматично перенаправляється на головну сторінку, де представлена стисла інструкція щодо формування   
витягів (див. рис. 4.4).

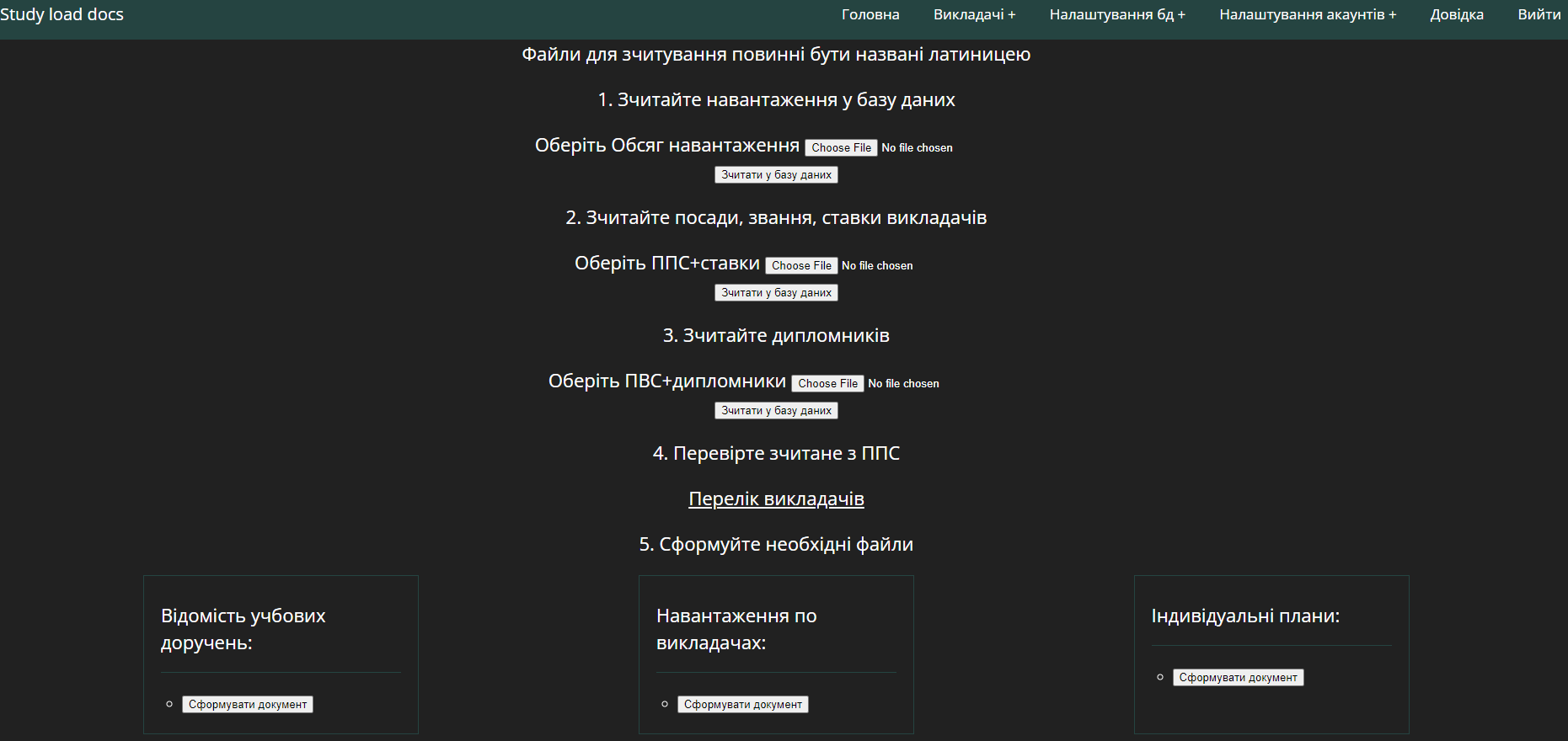


Рисунок 4.4 – Головна сторінка застосунку «SLDOCS»

Процес завантаження плану навчального навантаження розглянутий в роботі [6]. Створення індивідуальних планів та персональних витягів триває від 15 до 45 секунд. До завершення цього процесу кнопки для скачування документів будуть не видні на сторінці. Вони будуть відображені після завершення створення.

Кнопка для розрахунку статистики з’явиться тільки після завантаження необхідного документа. Завантаження полягає в тому, що зчитується файл Excel із розподілом навчального навантаження, після чого зчитані дані запишуться у відповідні таблиці бази даних. Центральна сторінка при цьому буде виглядати так, як показане на рис. 4.5.

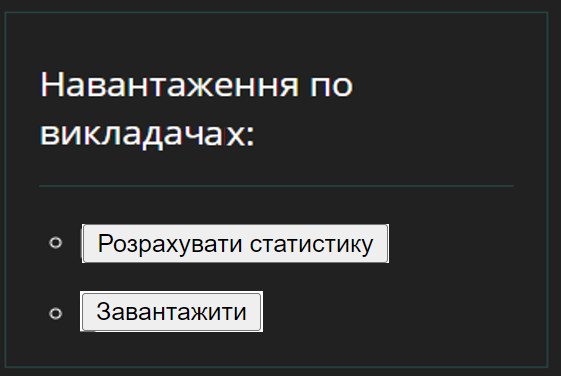


Рисунок 4.5 – Інтерфейс для формування файлів

Якщо натиснути на кнопку «Розрахувати статистику», то здійснюється перехід на сторінку, вигляд якої можна побачити на рис. 4.6. Простий інтерфейс дозволить ввести потрібні дані, після чого отримати три сторінки із показниками навантаження, які були визначені технічним завданням. На рис. 4.7, рис. 4.8 та рис. 4.9 можна побачити результати розрахунків.

Можна побачити, що отримані сторінки відповідають за представленими даними, значенням на сторінках, представлених на рис. 2.3, рис. 2.4, рис. 2.5. Це дозволяє стверджувати, що алгоритми розрахунків реалізовані у вірний спосіб.

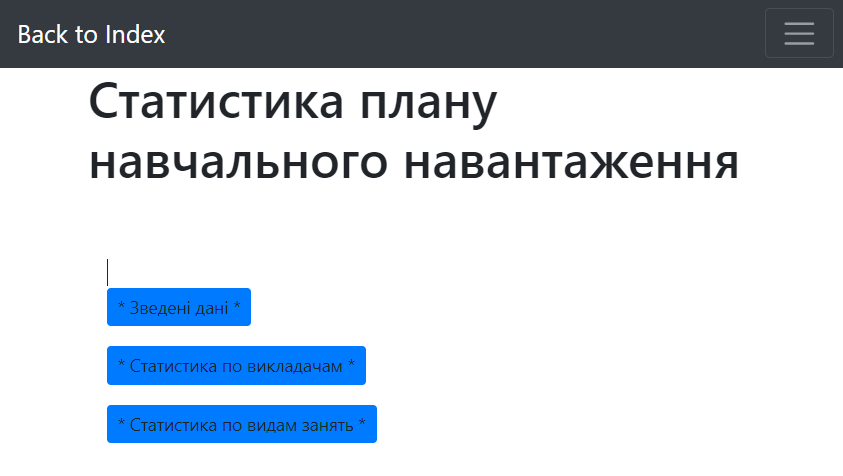


Рисунок 4.6 – Сторінка для старту розрахунків

На сторінці із загальними даними можна побачити елементи аналітичного аналізу отриманих цифр. У випадку, коли викладачу «не вистачає» навчального навантаження, то ці значення підкреслюють червоним кольором. Це досягається застосування тегів th:if, які надаються Thymeleаf. Використовуючи такі тегі, а також тегі циклів та багатоваріантного вибору, можна реалізувати потрібне «розфарбування» сторінки, яке дозволить краще аналізувати статистику.

На кожній сторінці із розрахованими показниками, є кнопка «Відправити». На початку роботи система пропонує ввести припустимий адрес пошти, на який потім користувач отримує лист із представленими зведеними даними   
у форматі Excel.

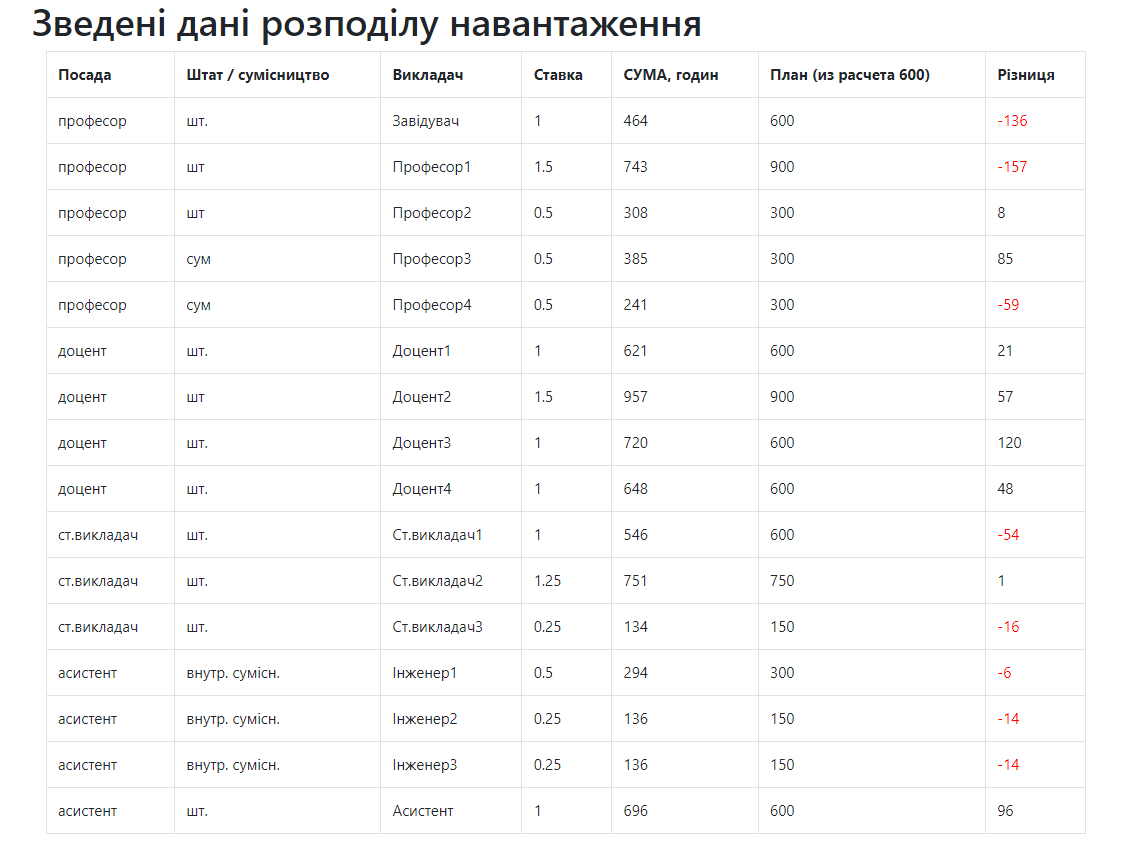


Рисунок 4.7 – Зведена сторінка розподілу навчального навантаження

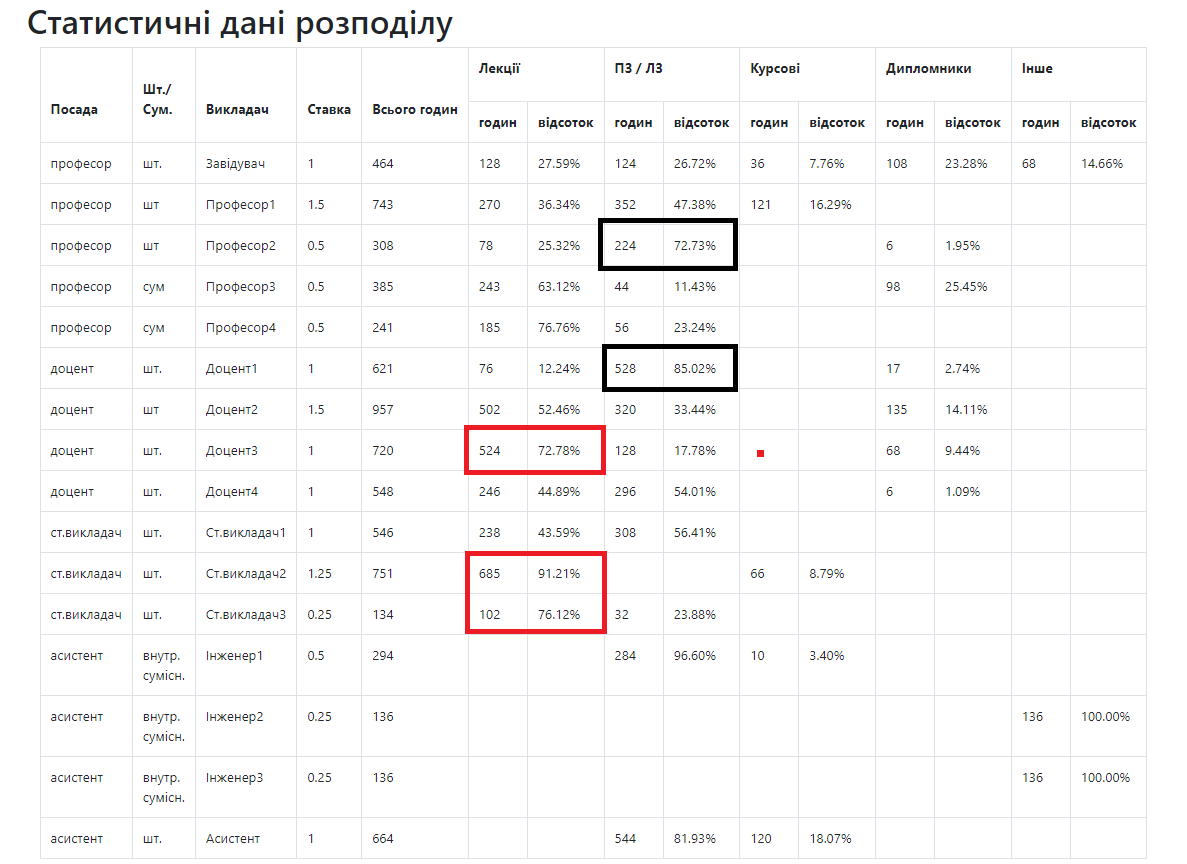


Рисунок 4.8 – Відомості про якість розподілу навчального навантаження



Рисунок 4.9 – Відомості про розподілу навантаження

# 5 Економічне обґрунтування проекту щодо розробки ПЗ

## 5.1 Обґрунтування доцільності розробки програмного забезпечення

У даному розділі представлено економічне обґрунтування розробки ПЗ для аналізу якості розподілу навчального навантаження викладачів кафедри НТУ «ХПІ», яка буде здійснена відповідно до [33]. Застосування ПЗ надає користувачам можливість формувати та оцінювати якість формування навчального навантаження викладачів. Через велику кількість навчальних дисциплін, які викладаються викладачами кафедри, виникає потреба в розробці програм для автоматизації рутинної роботи щодо формування планів навчального навантаження. Складовою частиною є оцінка якості такого навантаження та формування рекомендацій щодо його покращення.

В якості користувачів розробленого ПЗ будуть працівники кафедри ПІІТУ НТУ «ХПІ».

Розробка програмного забезпечення, яке дозволить легко та швидко формувати та аналізувати навчальне навантаження, сприятиме підвищенню ефективності управління навчальним процесом, дозволить запобігти помилок щодо перевантаження або недовантаження викладачі, а також зменшення часу на їх виправлення.

## 5.2 Оцінка конкурентоспроможності ПЗ у порівнянні з аналогом

Для здійснення аналізу та оцінки конкурентоспроможності розроблювального програмного забезпечення, було обрано в якості аналогу АСУ ВНЗ, який розроблено НДП Прикладних інформаційних технологій [[34](https://vuz.osvita.net/)]

Конкурентоспроможність ПЗ визначається властивостями його як товару на ринку. Для оцінки конкурентоспроможності ПЗ, що розроблено, проведено аналіз і порівняння з обраним аналогом за функціональним призначенням, основними технічними та експлуатаційними параметрами, областям використання.

Експлуатаційно-технічний рівень (ЕТУ) програмного забезпечення, що розроблено, визначено індексом (𝐽ЕТР) експлуатаційно-технічного рівня бально-індексним методом за п’ятибальною шкалою оцінювання. Результати розрахунку 𝐽ЕТР наведені у таблиці 5.1.

Таблиця 5.1 – Розрахунок показника якості

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Показники якості програмного продукту | Коефіцієнт вагомості, | Проект | | Аналог | |
|  |  |  |  |
| Зручність роботи(призначений для користувача інтерфейс) | 0,1 | 3 | 0,3 | 4 | 0,4 |
| Новизна (відповідність сучасним вимогам) | 0,1 | 5 | 0,5 | 4 | 0,4 |
| Відповідність профілю діяльності замовника | 0,25 | 5 | 1,25 | 3 | 0,75 |
| Ресурсна ефективність | 0,05 | 4 | 0,20 | 3 | 0,15 |
| Надійність (захист даних) | 0,2 | 3 | 0,6 | 2 | 0,4 |
| Швидкість доступу до даних | 0,1 | 4 | 0,4 | 3 | 0,3 |
| Гнучкість щодо налаштування | 0,15 | 5 | 0,75 | 2 | 0,3 |
| Здатність до навчання персоналу | 0,05 | 3 | 0,15 | 1 | 0,05 |
| Узагальнений показник якості | | 3,25 | | 2,75 | |

Для зрівнюваних ПЗ та програмного продукту як його аналога коефіцієнт технічного рівня буде дорівнювати 𝐴𝑘 = 3,25 ⁄ 2,75 = 1,18. Оскільки коефіцієнт більше 1, то розробка ПЗ з технічної точки зору виправдана.

## 5.3 Обґрунтування доцільності розробки ПЗ

У розробці ПЗ беруть участь керівник проєкту та виконавець (програміст). Комплекс робіт проекту по розробці ПЗ наведено в таблиці 5.2. Календарний графік виконання робіт наведено у таблиці 5.3.

Таблиця 5.2 – Комплекс робіт проекту по розробці ПЗ

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Зміст робіт | Виконавець | Тривалість, день | Навантаження | | | |
| Трив., день | % | Люди но-дні | |
| 1. Обґрунтування необхідності створення застосунку | | | | | | |
| 1.1. Опис предметної області | | | | | | |
| 1.1.1 Основні поняття | Керівник, Програміст | 2 | 1  2 | 10  100 | 0,1  2 | |
| 1.1.2 Аналіз ПЗ | Програміст | 5 | 5 | 100 | 5 | |
| 1.1.3 Опис етапів розробки ПЗ | Керівник, Програміст | 3 | 1  3 | 25  100 | 0,25  3 | |
| 1.1.4 Вибір методології розробки | Керівник, Програміст | 2 | 1  2 | 20  100 | | 0,2  2 |
| 1.2. Опис проєктування застосунку | | | | | | |
| 1.2.1 Аналіз бізнес-вимог | Програміст | 2 | 2 | 100 | | 2 |
| 1.2.2 Аналіз, вдосконалення бізнес-процесів | Керівник, Програміст | 4 | 2  4 | 25  100 | | 0,5  4 |
| 1.2.3 Функціональні вимоги | Керівник, Програміст | 2 | 1  2 | 50  100 | | 0,25  2 |
| 1.2.4 Нефункціональні вимоги | Керівник, Програміст | 2 | 1  2 | 50  100 | | 0,25  2 |
| 1.3 Постанова задачі | Керівник,  Програміст | 5 | 2  5 | 20  100 | | 0,4  5 |
| Разом з етапом 1 | Керівник, Програміст | 27 | 9  27 | 33  100 | | 2,45  27 |
| 2 Огляд методів та підходів для розв’язання задачі | | | | | | |
| 2.1 Огляд існуючих ПЗ | Керівник, Програміст | 3 | 1  3 | 25  100 | | 0,25  3 |
| 2.2 Архітектура ПЗ | Керівник, Програміст | 6 | 2  6 | 25  100 | | 0,5  6 |
| Разом з етапом 2 | Керівник, Програміст | 9 | 3  9 | 33  100 | | 0,75  9 |
| 3 Проєктування програмного забезпечення | | | | | | |
| 3.1 Розробка діаграм | Програміст | 4 | 4 | 100 | | 4 |
| 3.2 Розробка бази даних | Програміст | 10 | 10 | 100 | | 10 |
| 3.3 Вибір стеку технологій | Програміст | 5 | 5 | 100 | | 5 |
| Разом з етапом 3 | Програміст | 19 | 19 | 100 | | 19 |

Кінець таблиці 5.2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Зміст робіт | Виконавець | Тривалість, день | Навантаження | | | |
| Трив., день | % | Люди но-дні | |
| 4 Розробка та тестування програмного забезпечення | | | | | | |
| 4.1 Особливості програмної реалізації | Програміст | 20 | 20 | 100 | | 20 |
| 4.2 Тестування ПЗ | Програміст | 3 | 3 | 100 | | 3 |
| 4.3 Оцінка програмного забезпечення | Керівник, Програміст | 2 | 1  2 | 50  100 | | 0,5  2 |
| 4.4 Вдосконалення ПЗ | Керівник, Програміст | 2 | 1  2 | 50  100 | | 0,5  2 |
| Разом з етапом 4 | Керівник, Програміст | 27 | 2  27 | 7  100 | | 1  27 |
| 5 Оформлення робочої документації | | | | | | |
| 5.1 Економічне обґрунтування проекту | Програміст | 3 | 3 | 100 | | 3 |
| 5.2 Оформлення пояснювальної записки | Керівник, Програміст | 15 | 6  15 | 20  100 | | 1,2  15 |
| Разом з етапом 5 | Керівник, Програміст | 18 | 6  18 | 33  100 | | 1,2  15 |
| Разом за проєктом | Керівник, Програміст | 100 | 14  100 | 14  100 | | 5,4  100 |

Таблиця 5.3 – Календарний графік виконання робіт проєкту

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Зміст роботи | Виконавець | Тривалість, день | Графік роботи | |
| Початок | Кінець |
| 1. Основні поняття | Керівник | 1 | 15.02.2024 | 15.02.2024 |
| Програміст | 2 | 15.02.2024 | 16.02.2024 |
| 2. Аналіз системи керування картками пацієнтів | Програміст | 5 | 17.02.2024 | 21.02.2024 |
| 3. Опис етапів розробки | Керівник | 1 | 22.02.2024 | 22.02.2024 |
| Програміст | 3 | 22.02.2024 | 24.02.2024 |
| 4. Вибір методології розробки | Керівник | 1 | 25.02.2024 | 25.02.2024 |
| Програміст | 2 | 25.02.2024 | 26.02.2024 |
| 5. Аналіз бізнес-вимог | Програміст | 2 | 27.02.2024 | 28.02.2024 |
| 6. Аналіз та вдосконалення бізнес-процесів | Керівник | 2 | 29.02.2024 | 01.03.2024 |
| Програміст | 4 | 29.02.2024 | 03.03.2024 |
| 7. Функціональні вимоги | Керівник | 1 | 04.03.2024 | 04.03.2024 |
| Програміст | 2 | 04.03.2024 | 05.03.2024 |
| 8. Нефункціональні вимоги | Керівник | 1 | 06.03.2024 | 06.03.2024 |
| Програміст | 2 | 06.03.2024 | 07.03.2024 |
| 9. Постанова задачі | Керівник | 2 | 08.03.2024 | 09.03.2024 |
| Програміст | 4 | 08.03.2024 | 12.03.2024 |

Кінець таблиці 5.3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Зміст роботи | Виконавець | Тривалість, день | Графік роботи | |
| Початок | Кінець |
| 10. Огляд існуючих систем | Керівник | 1 | 13.03.2024 | 13.03.2024 |
| Програміст | 3 | 13.03.2024 | 15.03.2024 |
| 11. Архітектура програмного рішення | Керівник | 2 | 16.03.2024 | 17.03.2024 |
| Програміст | 6 | 16.03.2024 | 21.03.2024 |
| 12. Розробка діаграм послідовностей | Програміст | 4 | 22.03.2024 | 25.03.2024 |
| 13. Розробка бази даних | Програміст | 10 | 26.03.2024 | 04.04.2024 |
| 14. Вибір стеку технологій | Програміст | 5 | 05.04.2024 | 09.04.2024 |
| 15. Особливості програмної реалізації | Програміст | 20 | 10.04.2024 | 29.04.2024 |
| 16. Тестування ПЗ | Програміст | 3 | 30.04.2024 | 02.05.2024 |
| 17. Оцінка ПЗ | Керівник | 1 | 03.05.2024 | 03.05.2024 |
| Програміст | 2 | 03.05.2024 | 04.05.2024 |
| 18. Вдосконалення ПЗ | Керівник | 1 | 05.05.2024 | 05.05.2024 |
| Програміст | 2 | 05.05.2024 | 06.05.2024 |
| 19. Економічне обґрунтування проекту | Програміст | 3 | 07.05.2024 | 09.05.2024 |
| 20. Оформлення пояснювальної записки | Керівник | 6 | 10.05.2024 | 15.05.2024 |
| Програміст | 15 | 10.05.2024 | 24.05.2024 |

Календарний графік проєкту у вигляді графіка Ганта представлено на рис.5.1.

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание

Рисунок 5.1 – Календарний графік проєкту

## 5.4 Розрахунок проєктних витрат на розробку ПЗ

Основна заробітна плата виконавців проєкту вказана у таблиці 5.4. Витрати часу на розробку ПЗ по кожному виконавцю приймаються виходячи з його завантаження за календарним графіком виконання робіт.

Таблиця 5.4 – Основна заробітна плата виконавців проекту

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Посада | Посадовий оклад, грн | Середня денна ставка, грн | Витрати часу на розробку, людино-дні | Витрати на основну заробітну плату, грн |
| Керівник | 8500 | 8500/21 = 405 | 5,4 | 2187 |
| Програміст | 1950 | 1950/21 = 93 | 100 | 9300 |
| Усього | | | | 11487 |

Здійснено розрахунки використання машинного часу, необхідного для розробки ПЗ. Вартість однієї години машинного часу 5 грн/год, а за коефіцієнт мультипрограмності взято 1.

;

.

Витрати на матеріали наведені в таблиці 5.5.

Таблиця 5.5 – Витрати на матеріали

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Матеріали | Од. вимірювання | Необхідна кількість | Ціна за од., грн | Сума, грн |
| Зошит загальний | Шт. | 1 | 50 | 50 |
| Папір офісний | Пачка | 1 | 100 | 100 |
| Тонер принтера | Шт. | 1 | 200 | 200 |
| Усього | | | | 350 |

Отже, капітальні вкладення проєкту дорівнюють:

.

Кошторис на розробку ПЗ наведено у таблиці 5.6.

Таблиця 5.6 – Кошторис на розробку ПЗ

|  |  |
| --- | --- |
| Статті витрат | Сума, грн. |
| Основна заробітна плата | 11487 |
| Додаткова зарплата | 5744 |
| Відрахування | 3790 |
| Витрати на матеріали | 350 |
| Витрати на машинний час | 4000 |
| Накладні витрати організації | 6892 |
| Усього | 32263 |

Отже, на розробку ПЗ основні витрати склали 32263 грн.

## 5.5 Розрахунок витрат на впровадження ПЗ

Нижче наведено розрахунки витрат на впровадження ПЗ.

Необхідні кошти на закупівлю основного обладнання, до якого входить ПК вартістю 30000 грн. За ефективний річний фонд часу роботи технічного засобу було взято 2008 годин, тобто 251 робочих днів по 8 годин роботи. За трудомісткість одноразової обробки інформації береться 6 годин машинного часу. Частота рішення дорівнює 251 днів на рік.

Витрати на реалізацію проєкту складають:

.

Отже, сумарні витрати проєкту щодо розробки ПЗ складуть:

.

## 5.6 Розрахунок витрат на впровадження ПЗ

Розрахуємо сумарні витрати, які пов’язані з впровадженням аналога, з яким порівнюється ПЗ. Такі витрати складаються з:

* витрат на придбання програмного продукту 80 долорів на місяць за «необмежену» версію, або приблизно 38400 грн на рік;
* витрати на обладнання, знадобиться такий же ПК, що і для ПЗ вартістю 30000 грн.

Капітальні вкладення складають:

.

Отже, сумарні витрати на впровадження аналога складуть:

.

Також потрібно враховувати, що 38400 грн потрібно буде платити кожен рік, так наприклад сума на другий рік впровадження буде вже 99300 грн, а на третій 137700 грн.

## 5.7 Розрахунок поточних експлуатаційних витрат ПЗ та його аналога

Експлуатаційні витрати включають витрати, пов'язані із забезпеченням нормального функціонування програмного забезпечення.

Дані по заробітній платі користувачів ПЗ наведено у таблиці 5.7. Дані по заробітній платі фахівців для впровадження аналогу представлено у таблиці 5.8.

Таблиця 5.7 – Дані по заробітній платі користувачів «Client card»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Посада | Посадовий оклад, грн. | Середня денна ставка, грн/день | Витрати часу на роботу з ПЗ, людино-дні | Фонд з/п, грн. |
| Програміст | 22000 | 1047.62 | 1 день \* 12 = 12 | 12571 |
| Адміністратор | 12000 | 571.43 | 8 днів \* 12 = 96 | 54857 |
| Разом | | | | 67428 |

Таблиця 5.8 – Дані по заробітній платі фахівців для впровадження аналогу

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Посада | Посадовий оклад, грн. | Середня денна ставка, грн/день | Витрати часу на роботу з ПЗ, людино-дні | Фонд з/п, грн. |
| Програміст | 40000 | 1904.76 | 24 | 45714 |
| Адміністратор | 18000 | 857.14 | 96 | 82286 |
| Разом | | | | 128000 |

Витрати на заробітну плату користувачів ПЗ та аналога складають:

;

.

Суми амортизаційних відрахувань для ПЗ та аналога складають:

.

На даний час тариф на електроенергію 2.64 грн/кВт\*год. Встановлена потужність для комп'ютера дорівнює 0,4 кВт.

Витрати на електроенергію для користувачів ПЗ та аналога складають:

.;

.

Витрати на поточний ремонт обладнання ПЗ та аналога складуть:

*.;*

.

Витрати на матеріали, які споживаються кожного року складають 1 відсоток від балансової вартості основного обладнання. Отже, витрати на матеріали для користувачі ПЗ та HospApp однакова, а саме:

Норматив накладних витрат станове 20% від прямих витрат. Накладні витрати для ПЗ та аналога складуть:

.

.

Річні експлуатаційні витрати ПЗ представлені у таблиці 5.9.

Отже, поточні експлуатаційні витрати ПЗ складуть 143527 грн, а його аналога – 268226 грн.

Таблиця 5.9 – Річні експлуатаційні витрати програмного продукту

|  |  |  |
| --- | --- | --- |
| Статі витрат | Витрати на ПЗ, грн | Витрати на HospApp, грн |
| Основна і додаткова зарплата з відрахуванням | 115168 | 218624 |
| Амортизовані відрахування | 2581 | 2868 |
| Витрати на електроенергію | 912 | 1013 |
| Витрати на поточний ремонт | 645 | 717 |
| Витрати на матеріали | 300 | 300 |
| Накладні витрати | 23921 | 44704 |
| Усього витрат | 143527 | 268226 |

## 5.8 Розрахунок показника економічного ефекту від розробки програмного забезпечення

Річний економічний ефект від використання розробленого ПЗ визначається по різниці приведених витрат на розробку та експлуатацію за рік ПЗ та витрат на закупку та експлуатацію його аналога.

Нормативний коефіцієнт економічної ефективності становить 0,33

Приведені витрати на виконання одиниці роботи за допомогою розробленого ПЗ та аналога, складають:

.

.

Тоді економічний ефект від використання розробленого ПЗ складе:

.

Зведені розрахуни економічного ефекту наведені у таблиці 5.10.

Таблиця 5.10 – Річні експлуатаційні витрати програмного продукту

|  |  |  |
| --- | --- | --- |
| Характеристика | Значення для | |
| ПЗ | аналога |
| Собівартість робіт (поточні експлуатаційні витрати програмного продукту), грн. | 143527 | 268226 |
| Сумарні витрати, які пов’язані з впровадженням ПЗ, грн. | 54763 | 60900 |
| Приведені витрати на одиницю робіт, грн. | 161598 | 288323 |
| Економічний ефект від використання програмного продукту, грн. | 299719 | |

Після визначення річного економічного ефекту необхідно розрахувати термін окупності витрат на розробку програмного продукту.

Термін окупності ПЗ складе:

.

Розрахований коефіцієнт економічної ефективності розробки складає:

.

Фактичний коефіцієнт економічної ефективності розробки ПЗ більше, ніж нормативний (5.56>0.33), тому розробка та впровадження ПЗ є ефективно доцільною.

Отже, доведено технічну доцільність і економічну ефективність розробленого ПЗ.

# Висновки

Результатом виконання дипломної роботи запропонований новий алгоритм здійснення розподілу навчального навантаження, в якокму з’явився етап розрахунку показників, які можна проаналізувати для того, щоб оцінити повному, коректність та якість отриманого варіанту розподілу. Для автоматизації бізнес-процесу до веб-застосунку «SLDOCS» був доданий компонент «SLSTАTS», який забезпечує розрахунок та відображення визначеної статистики. Під час створення нового програмного компоненту були виконані наступні завдання:

1. на основі спілкування з планувальниками навчального навантаження та особистої участі у підготовці плану навчального навантаження була створена -діаграма BPMN вдосконаленого бізнес-процесу створення плану навчального навантаження із оцінюванням;
2. визначені вимоги до програмного компоненту;
3. виконане проєктування, реалізацію та тестування програмного забезпечення;
4. проведені дослідження застосування створеного програмного рішення для оцінювання розподілу навчального навантаження.

За результатами роботи запропоновані наступні тези для обговорення.

1. Автоматитичне визначення визначених показників плану навчального навантаження при плануванні діяльності кафедри сприяє забезпеченню відповідності акредитаційним та ліцензійним вимогам. Це відбувається за рахунок того, що керівники зможуть швидко отримувати показники якості розоподілу та своєчасно втручатися у процеси планування, забезпечуючи належні показники.
2. Використання фреймворку Spring є ефективним для створення інформаційних систем на основі багатошарової архітектури.
3. Застосування формату СКБД MySQL 8 для збереження даних дає можливість безкоштовно зберігати всі дані.
4. Розроблений програмний компонент доцільний до застосування у процесі розподілу навчального навантаження кафедри.

Для створення веб-застосунку були використані навички, отримані за період здобуття освіти, що свідчить про відмінну побудову планів підготовки на кафедрі. Також процес створення програм показав важливість якісної розробки як вимого, так і проєкту програмного забезпечення.

Сукупність представлених результатів та висновків дозволяє стверджувати, що всі завдання, поставлені на дипломне проєктування, були успішно виконані, а мету роботи слід вважати в основному досягнутою.

# Список джерел інформації

1. Закон України «Про вищу освіту» // https://zakon2.rada.gov.ua-/laws/show/1556-18, 23.04.2024.
2. Наказ Міністерства освіти та науки від 21 червня 2022 року №686 «Про затвердження норм часу для планування і обліку навчальної роботи та переліків видів навчальної, методичної, інноваційної, наукової, органіізаційної роботи та іншої педагогічної діяльності педагогічних і науково-педагогічних працівників закладів фахової передвищої освіти».
3. Положення про планування та облік основних видів роботи науково-педагогічних працівників НТУ «ХПІ»; затверджено Вченою Радою НТУ «ХПІ» 23 вересня 2022 р., протокол №6; введено в дію наказом ректора 28 вересня 2022 р. № 264 ОД. – Харків, 2022. – 28 с.
4. Репів В. Процесний підхід до моделювання бізнес-процесів – К.: Думка, 2019. – 522 с.
5. Двухглавов, Д., Безменов, М., Шматко, О., Двухглавова, А., Корховий, Д., & Черкун, М. (2022). Формалізація процесу підготовки плану навчального навантаження викладачів кафедри закладу вищої освіти / Вісник Національного технічного університету «ХПІ». Серія: Системний аналiз, управління та iнформацiйнi технологiї (1 (7)), 35–45. // <https://doi.org/10.20998/2079-0023.2022.01.06>, 10.04.2024.
6. Корховий Д.С. Проєктування, розробка та дослідження програмного рішення для формування витягів з плану навчального навантаження науково-педагогічних працівників кафедри: дипломна робота магистра. – Х. : НТУ «ХПІ», 2023. – 100 с.
7. Official site of the Spring // https://spring.io/ , 19.05.2024.
8. Official site of the [Thymeleaf](https://www.thymeleaf.org/) // <https://www.thymeleaf.org/>, 19.05.2024.
9. [BPMN Specification - Business Process Model and Notation](https://www.bpmn.org/) // https://www.bpmn.org/, 31.03.2024.
10. [UML modeling tools for Business, Software, Systems and Architecture (sparxsystems.com)](https://sparxsystems.com/) // <https://sparxsystems.com/>, 31.03.2024.
11. Embley D, Thalheim B. Handbook of Conceptual Modeling: Theory, Practice, and Research Challenges. Berlin, Springer, 2017. – 597 p.
12. What Is an ORM? How Does It Work? How Should We Use One? | Baeldung on Computer Science // <https://www.baeldung.com/cs/object-relational-mapping>, 27.04.2024.
13. MVC - MDN Web Docs Glossary: Definitions of Web-related terms | MDN (mozilla.org) // <https://developer.mozilla.org/en-US/docs/Glossary/MVC>, 27.04.2024.
14. Guide to JPA with Hibernate - Relationship Mapping (stackabuse.com) //https://stackabuse.com/a-guide-to-jpa-with-hibernate-relationship-mapping/ , 27.04.2024.
15. [Hibernate Tutorial (javaguides.net)](https://www.javaguides.net/p/hibernate-tutorial.html) <https://www.javaguides.net/p/hibernate-tutorial.html> , 31.05.2024.
16. [Порівняння СКБД //](http://ni.biz.ua/8/8_6/8_63622_sravnenie-subd.html) <http://ni.biz.ua/8/8_6/8_63622_sravnenie-subd.html>, 08.03.2024.
17. ISO/IEC/IEEE 29148. Systems and software engineering – Life cycle processes – Requirements engineering // <https://webstore.iec.ch/preview/> info\_isoiecieee29148%7Bed2.0%7Den.pdf, 23.03.2024.
18. Requirements engineering for software and systems / Phillip A. Laplante. – CRC Press, 2018. – 399 p.
19. Fowler M. UML Distilled. Ch. 4 // [https://www.pearsonhighered.com/ assets/samplechapter/0/3/2/1/0321193687.pdf](https://www.pearsonhighered.com/%20assets/samplechapter/0/3/2/1/0321193687.pdf), 06.02.2024.
20. ISO/IEC 25010:2011(en). Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models.
21. Spring Boot/Cloud and WildFly Swarm comparison // [https://blog.akquinet.de/2017/01/12/the-lightweightness-of-microservices-comparing-spring-boot-wildly-swarm-and-haskell-snap/#:~:text=Spring%20Boot%20  
    provides%20a%20single,Java%20are%20built%20with%20Maven](https://blog.akquinet.de/2017/01/12/the-lightweightness-of-microservices-comparing-spring-boot-wildly-swarm-and-haskell-snap/#:~:text=Spring%20Boot%20provides%20a%20single,Java%20are%20built%20with%20Maven), 23.04.2024.
22. Site about Jakarta EE 9 // <https://jakarta.ee/release/9/>, 23.04.2024.
23. [Project Lombok](https://projectlombok.org/) <https://projectlombok.org/>, 23.03.24.
24. [Spring Security](https://spring.io/projects/spring-security) // <https://spring.io/projects/spring-security>, 23.03.24.
25. Apache POI // https://poi.apache.org, 02.02.2024.
26. [Bootstrap · The most popular HTML, CSS, and JS library in the world. (getbootstrap.com)](https://getbootstrap.com/) //https://getbootstrap.com/, 23.03.24.
27. [Log4j – Apache Log4j™ 2](https://logging.apache.org/log4j/2.x/) // <https://logging.apache.org/log4j/2.x/>, 23.03.2024.
28. [Maven – Welcome to Apache Maven](https://maven.apache.org/) // <https://maven.apache.org/>, 23.03.2024.
29. IntelliJ IDEA – the Leading Java and Kotlin IDE // https://www.jetbrains.com/idea/, 10.05.2024.
30. Beck K. Test driven development: By Example. Ch. 25 // <https://dokumen.pub/test-driven-development-by-example-0321146530-9780321146533-n-2465731.html>, 06.05.2024.
31. The Perceived Business Benefit of Cloud Computing: An Exploratory Study // [https://scholarworks.lib.csusb.edu/cgi/viewcontent.cgi?referer=&httpsredir=1& article=1297& context=jitim](https://scholarworks.lib.csusb.edu/cgi/viewcontent.cgi?referer=&httpsredir=1&%20article=1297&%20context=jitim)б, 23.04.2024.
32. Apache Tomcat 9 // https://tomcat.apache.org/, 02.12.2024.
33. Методичні вказівки до виконання економічного обґрунтування проекту щодо розробки програмного забезпечення дипломної роботи освітньо-кваліфікаційного рівня «бакалавр» для студентів спеціальностей 121 – Інженерія програмного забезпечення, 122 –Комп’ютерні науки, 126 – Інформаційні системи та технології у галузі знань 12 – Інформаційні технології. // уклад. Москаленко В. В, Шматко О.В., Фонта Н. Г. – Харків : НТУ «ХПІ» – 2022 – 35 с.
34. [Веб Деканат - АСУ «ВНЗ» (osvita.net)](https://vuz.osvita.net/veb-dekanat-main/) // https://vuz.osvita.net/veb-dekanat-main/, 02.06.2024.