НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Навчально-науковий інститут атомної та теплової енергетики
Кафедра інженерії програмного забезпечення в енергетиці

«На правах рукопису»
УДК 004.9

До захисту допущено:
В.о. завідувача кафедри
_____ Олександр КОВАЛЬ
«___»_____202_ р.

Магістерська дисертація

За освітньою програмою «Інженерія програмного забезпечення інтелектуальних кібер-фізичних систем в енергетиці»

Спеціальності 121 Інженерія програмного забезпечення

на тему: Інформаційна система забезпечення навчально-виховної діяльності катедри.

Виконав студент 2 курсу, групи ТВ-32мп
___Дячук Андрій Олегович_____ _____
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник
___Доцент, к.ф.-м.н., доц. Свинчук О.В_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент
___професор кафедри ЦТЕ, д.т.н., проф. Отрох С.І._____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2024

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Навчально-науковий інститут атомної та теплової енергетики
Кафедра інженерії програмного забезпечення в енергетиці
Рівень вищої освіти другий, магістерський
За освітньою програмою «Інженерія програмного забезпечення інтелектуальних кібер-фізичних систем в енергетиці»
Спеціальності 121 Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Олександр КОВАЛЬ
«___»_____202__ р.

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ

Дячуку Андрію Олеговичу
(прізвище, ім'я, по батькові)

1. Тема дисертації: «Інформаційна система забезпечення навчально-виховної діяльності катедри»
Науковий керівник Свинчук Ольга Василівна, доцент, к.ф.-м.н.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «___»_____ 202_ року №_____
2. Строк подання студентом дисертації «___»_____ 202__ р.
3. Вихідні дані до роботи мова програмування Java та застосування Spring Framework, система керування базами даних PostgreSQL, шаблонізатор Thymeleaf, мова розмітки HTML, середовище розробки IntelliJ IDEA.
4. Перелік питань, які потрібно розробити розробити вебзастосунок для інформаційної системи, забезпечити автоматизацію бізнес-процесів навчально-виховної діяльності катедри, організувати централізований доступ до даних, гарантувати безпеку даних, надати можливість автоматизованого створення та зберігання звітів, реалізувати інструменти моніторингу успішності й аналітики, забезпечити комунікацію між учасниками освітнього процесу, спроєктувати зручний користувацький інтерфейс.
5. Орієнтовний перелік ілюстративного матеріалу аналіз наявних систем, засоби розробки, діаграми прецедентів, архітектура системи, скриншоти сторінок системи.

6. Орієнтовний перелік публікацій Дячук А.О., Свинчук О.В., Бандурка О.І. Інформаційна система забезпечення навчально-виховної діяльності катедри. Матеріали XXIV Всеукраїнської науково-технічної конференції молодих вчених, аспірантів та студентів «Стан, досягнення та перспективи інформаційних систем і технологій – 2024». Одеса, 18-19 квітня 2024 р. С.231-232.

7. Дата видачі завдання «__» _____ 202_ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Строки виконання етапів магістерської дисертації	Примітки
1	Отримання завдання	01.10.2023	Виконано
2	Дослідження предметної області	01.10.2023 - 01.11.2023	Виконано
3	Постановка вимог до проєктування системи	01.11.2023 – 01.12.2023	Виконано
4	Дослідження наявних рішень	01.12.2023 – 01.03.2024	Виконано
5	Підготовка публікацій	01.03.2024 – 01.04.2024	Виконано
6	Розробка програмного продукту	01.04.2024 – 14.09.2024	Виконано
7	Тестування	15.09.2024 – 13.10.2024	Виконано
8	Захист програмного продукту	14.10.2024 – 20.10.2024	Виконано
9	Підготовка магістерської дисертації	20.10.2024 – 24.11.2024	Виконано
10	Передзахист	25.11.2024 – 01.12.2024	Виконано
11	Захист	17.12.2024 – 20.12.2024	Виконано

Студент _____ ___Андрій ДЯЧУК__
(підпис) (Ім'я ПРІЗВИЩЕ)

Керівник роботи _____ _Ольга СВИНЧУК__
(підпис) (Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Структура і обсяг магістерської дисертації. Дисертація складається зі вступу, п’яти розділів, висновків. Робота включає 31 рисунок, 13 таблиць, 3 додатки та 22 посилання. Основна частина роботи міститься на 90 сторінках.

Актуальність теми обумовлена зростаючою потребою у цифровізації навчально-виховних процесів у закладах вищої освіти, зокрема на рівні катедр. Відсутність інтегрованих і автоматизованих рішень ускладнює процеси планування, аналізу та контролю якості освітньої діяльності. Сучасні потреби, такі як оптимізація документообігу та звітності, покращення комунікації та співпраці між учасниками освітнього процесу, а також автоматизація бізнес-процесів, вимагають впровадження інформаційних систем.

Метою роботи є розробка інформаційної системи, яка спрямована на підвищення ефективності управління та автоматизацію ключових бізнес-процесів навчально-виховної діяльності катедри.

Для досягнення мети поставлені такі завдання:

- визначення основних потреби користувачів і вимог до системи;
- аналіз аналогічних інформаційних систем;
- розробка вебзастосунку, який відповідає потребам навчально-виховного напряму катедри і складає функціонал системи;
- розробка персоналізованого інтерфейсу користувача з відповідними обмеженнями доступу до функціоналу та даних.

Методи дослідження. У роботі застосовано системний підхід для аналізу вимог до інформаційної системи та визначення її функціональних компонентів. Метод порівняльного аналізу використано для оцінки існуючих аналогів та виявлення їхніх недоліків, що дозволило обґрунтувати унікальність розробленої системи. Інженерні методи програмування та моделювання застосовано для розробки структури системи, діаграм та бази даних.

Практичне значення одержаних результатів полягає в розробці надійної інформаційної системи, яка реалізує потреби учасників навчально-виховної діяльності катедри.

Апробація результатів дисертації. Основні положення та висновки дослідження обговорювались на XXIV Всеукраїнській науково-технічній конференції молодих вчених, аспірантів та студентів «Стан, досягнення та перспективи інформаційних систем і технологій – 2024», м. Одеса, 18-19 квітня 2024 р.

Публікації. Зміст і результати дослідження відображено в публікації:

Дячук А.О., Свинчук О.В., Бандурка О.І. Інформаційна система забезпечення навчально-виховної діяльності катедри. *Матеріали XXIV Всеукраїнської науково-технічної конференції молодих вчених, аспірантів та студентів «Стан, досягнення та перспективи інформаційних систем і технологій – 2024»*. Одеса, 18-19 квітня 2024 р. С.231-232.

URL: https://ontu.edu.ua/download/konfi/2024/Conference_abstract-IT-2024.pdf

Ключові слова: *інформаційна система, освітній напрям, катедра, вебзастосунок, PostgreSQL, Java, Spring, автоматизація бізнес-процесів*.

ABSTARCT

The structure and scope of the thesis. The thesis consists of an introduction, five chapters, conclusions. The work includes 31 figures, 13 tables, 3 appendices and 22 references. The main part of the work is contained on 90 pages.

Relevance of topic is due to the growing need for digitalization of educational processes in higher education institutions, in particular at the department level. The lack of integrated and automated solutions complicates the processes of planning, analysis and quality control of educational activities. Modern needs, such as optimizing document flow and reporting, improving communication and cooperation between participants in the educational process, as well as automating business processes, require the implementation of information systems.

The research goal of the work is to develop an information system aimed at increasing the efficiency of management and automation of key business processes of the department's educational activities.

To achieve the goal, the following objectives have been set:

- determination of the main needs of users and requirements for the system;
- analysis of similar information systems;
- development of a web application that meets the needs of the educational direction of the department and makes up the functionality of the system;
- development of a personalized user interface with appropriate restrictions on access to the functionality and data.

Methods of research. The work uses a systems approach to analyze the requirements for the information system and determine its functional components. The comparative analysis method is used to evaluate existing analogues and identify their shortcomings, which allowed us to substantiate the uniqueness of the developed system. Engineering programming and modeling methods are used to develop the system structure, diagrams and database.

Practical value of obtained results is in the development of a reliable information system that meets the needs of participants in the educational activities of the department.

Approbation of dissertation results. The main provisions and conclusions of the study were discussed at the XXIV All-Ukrainian Scientific and Technical Conference of Young Scientists, Postgraduate Students and Students "State, Achievements and Prospects of Information Systems and Technologies - 2024", Odesa, April 18-19, 2024.

Publications. The content and results of the study are reflected in the publication:

Diachuk A.O., Svynchuk O.V., Bandurka O.I. Information system for supporting the educational activities of the department. *Materials of the XXIV All-Ukrainian Scientific and Technical Conference of Young Scientists, Postgraduate Students and Students "State, Achievements and Prospects of Information Systems and Technologies - 2024"*. Odesa, April 18-19, 2024. P.231-232.

URL: https://ontu.edu.ua/download/konfi/2024/Conference_abstract-IT-2024.pdf

Keywords: *information system, educational direction, department, web application, PostgreSQL, Java, Spring, business process automation.*

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

ІС	Інформаційна система
СКБД	Система керування базами даних
API (англ. Application Programming Interface)	Прикладний програмний інтерфейс
CRM (англ. Customer Relationship Management)	Управління взаємовідносинами з клієнтами
ERP (англ. Enterprise Resource Planning)	Система планування ресурсів підприємства для автоматизації обліку й керування
IDE (англ. Integrated Development Environment)	Інтегроване програмне середовище, яке об'єднує інструменти для написання, редагування, налагодження та компіляції коду в одному інтерфейсі
JPA (англ. Java Persistence API)	Стандарт для роботи з об'єктно-реляційним відображенням (ORM) в Java, що дозволяє зручно взаємодіяти з базами даних через об'єкти
MVC (англ. Model-View-Controller)	Архітектурний шаблон, що розділяє програму на три компоненти: модель (дані), відображення (інтерфейс користувача) і контролер (управління взаємодією між ними)

ВСТУП

Інформаційні системи є невід’ємною частиною в організаціях, які оперують великою кількістю даних. Основними цілями використання інформаційної системи є автоматизація бізнес-процесів, підвищення продуктивності праці, підтримка взаємодії між користувачами та забезпечення доступу до інформації.

Розвиток інформаційних технологій відкриває нові можливості для підвищення ефективності управління освітніми процесами в закладах вищої освіти. Зокрема, автоматизація бізнес-процесів навчально-виховної діяльності катедри може значно полегшити виконання таких завдань, як організація роботи кураторів, взаємодія з деканатом, інформування студентів і викладачів, аналіз результатів навчання, створення та збереження звітів, протоколів та інших документів, пов'язаних з навчальною діяльністю катедри. Традиційні способи управління такими процесами за допомогою ручної обробки даних та паперової документації часто є трудомісткими, що може знижувати продуктивність і точність виконання важливих освітніх завдань.

Наявні в університетах системи зазвичай не можуть повністю задовольнити усі потреби катедри, зокрема її навчально-виховний напрям. Окрім того, розробка власної системи дозволяє реалізувати додатковий функціонал для оптимізації адміністративних процесів, обробки табличних даних у власному форматі, формування звітів за шаблонами катедри тощо.

Метою роботи є розробка та впровадження інформаційної системи, яка спрямована на підвищення ефективності управління та автоматизацію ключових бізнес-процесів навчально-виховної діяльності катедри. Крім того, інформаційна система може слугувати сховищем даних для подальшого розвитку нових функцій і застосунків, що сприятиме удосконаленню освітнього процесу та підвищенню його якості.

Автоматизація таких процесів є важливим кроком на шляху до модернізації освітньої діяльності катедри та забезпечення її відповідності сучасним вимогам до організації освітнього процесу.

Відповідно до мети визначимо структуру роботи.

1. Постановка завдання, формування основи дослідження, визначення вимог до інформаційної системи. Опис потреб користувачів, функціоналу і вибір способу реалізації системи.

2. Порівняльний аналіз аналогічних інформаційних систем, виявлення їхніх обмежень і визначення можливостей для вдосконалення. Формування концептуальної основи для проєктування системи.

3. Огляд апарату вирішення для поставленої задачі, методів та інструментів розробки. Обґрунтування вибору архітектурного стилю, програмних засобів та алгоритмів, що забезпечують оптимальне функціонування системи.

4. Опис програмної системи, деталізації її структури, компонентів і взаємодій між ними. Наведення функціональних можливостей системи та принципів її використання. Огляд інтерфейсу користувача і сторінок системи.

5. Розробка стартап-проєкту, дослідження ринкових можливостей створеної системи. Опис маркетингової стратегії, конкурентних переваг. Визначення перспектив впровадження проєкту у комерційне середовище. Розробка планів розвитку та позиціонування системи на ринку освітніх технологій.

Ці розділи дозволять забезпечити ясність та структурованість у поясненні розробки та функціональності системи, а також надати повну картину її можливостей, технічної архітектури, безпеки та плану розвитку. Така структура дозволяє повною мірою розкрити цінність створеного рішення для навчально-виховної діяльності катедри та обґрунтувати його значущість у сучасному освітньому середовищі.

1 ПОСТАНОВКА ЗАВДАННЯ

Створення інформаційної системи забезпечення навчально-виховної діяльності катедри є важливим етапом оптимізації освітнього процесу. Така система повинна забезпечити автоматизацію ключових процесів, що включають організацію роботи кураторів, інформування студентів та викладачів, аналіз успішності, а також обробку й зберігання звітної документації. Необхідним завданням є У цьому розділі розглянемо основні завдання, які має вирішувати система, а також вимоги до її функціоналу, способу реалізації та архітектури.

1.1 Інформаційні системи

Організації, підприємства та фірми з кожним роком вкладають все більше коштів у створення та впровадження інформаційних систем для власних потреб. Розвиток технологій та їхній вплив на успіх у бізнесі не можуть залишатися непоміченими. З’являються нові підприємства та галузі, старі занепадають, а успішними стають ті фірми, які використовують нові технології. Це і робить інформаційні системи управління одним із основних пріоритетів сьогоднішніх індустрій.

Інформаційні системи повинні реалізовувати основні бізнес-процеси за допомогою цифрових мереж, що охоплюють всю організацію або об’єднують кілька організацій. Бізнес-процеси є сукупністю логічно пов’язаних завдань і алгоритмів для отримання конкретних бізнес-результатів, а також є унікальним способом організації та координації цієї діяльності. Катедра має власні бізнес-процеси, прикладом яких є розробка та актуалізація навчальних програм, контроль знань студентів, документообіг та ін. При цьому кожен працівник знає свої обов'язки та функції, оскільки результати кожного з цих бізнес процесів повинні відповідати певним вимогам. Для опису бізнес-процесів використовують нотації, як-от BPMN, IDEF0, EPC. Нотації є загальноприйнятими стандартами в певній

галузі або діяльності. З їхньою допомогою можна візуалізувати зони відповідальності співробітників, етапи передачі повноважень між учасниками процесу, статус задачі на момент її передачі (що і в якому вигляді працівник повинен отримати від попереднього учасника), кінцевий результат і методи його досягнення для кожного співробітника [1].

Зі зростанням компанії її робочі процеси стають дедалі складнішими, і базові схеми управління перестають забезпечувати необхідну ефективність та безперебійність функціонування. На цьому етапі доцільним стає впровадження автоматизації окремих процедур за допомогою спеціалізованих програмних рішень. Автоматизація бізнес-процесів передбачає створення в інформаційній системі алгоритмів, які реалізують послідовність дій у вигляді автоматизованого ланцюжка завдань.

Однак автоматизація не є панацеєю від неефективності: її результативність залежить від попередньо налагодженої та логічно впорядкованої структури підприємства. Автоматизація, впроваджена у хаотично організоване середовище, не лише не вирішує наявні проблеми, а може ускладнити управлінські процеси. Отже, оптимізація шляхом автоматизації має здійснюватися лише після аналізу, реорганізації та формалізації бізнес-процесів (у результаті чого можна впевнитися у їхній чіткості й узгодженості).

Варто зазначити, що сучасні системи автоматизації бізнес-процесів часто базуються на використанні принципів управління бізнес-процесами (англ. BPM). Цей підхід передбачає моделювання, впровадження, моніторинг та постійне вдосконалення процесів із застосуванням цифрових інструментів. Впровадження таких рішень дозволяє мінімізувати людський фактор, знижує ймовірність помилок, підвищує продуктивність і забезпечує гнучкість у масштабуванні діяльності підприємства.

Існує зростаюча взаємозалежність між здатністю фірми використовувати інформаційні технології та її здатністю реалізовувати корпоративні стратегії та досягати цілей. Впровадження нових можливостей у компанії часто залежить від

того, чи інформаційні системи можуть надати функціонал для їхньої якісної обробки.

Технічно інформаційну систему можна визначити як набір взаємопов'язаних компонентів, які збирають (або отримують), обробляють, зберігають і поширюють інформацію для підтримки прийняття рішень і контролю в межах організації [2]. Окрім підтримки прийняття рішень, координації та контролю, ІС можуть надавати інструменти аналізу, візуалізації, звітності, комунікації та співробітництва.

З точки зору бізнесу, інформаційна система забезпечує вирішення проблеми або виклику, що стоїть перед компанією, і є комбінацією елементів управління, організації та технології. Вимір управління інформаційних систем включає такі аспекти, як лідерство, стратегія та поведінка керівництва. Технологічний вимір складається з комп'ютерного обладнання, програмного забезпечення, технологій керування даними та мережевих/телекомунікаційних технологій. Організаційний вимір інформаційних систем включає такі питання, як ієрархія організації, функціональні особливості, бізнес-процеси, культура та інтереси користувачів.

1.1.1 Головні компоненти інформаційної системи

Інформаційна система ґрунтується на сукупності ключових компонентів, які взаємодіють між собою для забезпечення ефективного виконання завдань і безперебійної роботи системи. Ці компоненти включають апаратне та програмне забезпечення, дані, мережі та людські ресурси.

Апаратне забезпечення включає фізичні пристрої, необхідні для функціонування системи, такі як комп'ютери, сервери, периферійні пристрої (принтери, сканери) та мережеве обладнання (маршрутизатори, комутатори). Апаратна інфраструктура є базовою платформою, на якій працює програмне забезпечення та здійснюється обробка даних.

Хоча комп'ютерні інформаційні системи використовують комп'ютерні системи для обробки необроблених даних у значущу інформацію, існує чітка

різниця між комп’ютерною й інформаційною системою. Основні відмінності подані у таблиці 1.1.

Таблиця 1.1 – Відмінності між інформаційною та комп’ютерною системами

Характеристика	Інформаційна система	Комп'ютерна система
1	2	3
Мета	Управління інформацією для підтримки рішень, автоматизація бізнес-процесів	Обчислення та обробка даних
Складові	Технічні засоби, програмне забезпечення, дані, люди, процеси	Апаратне і базове програмне забезпечення
Роль у бізнесі	Забезпечення ефективної роботи всієї організації	Забезпечення технічної основи для роботи
Приклад	ERP-система, CRM-система	Сервери, ПК, мережеве обладнання

Комп’ютери та пов’язані з ними програми є технічною основою, інструментами та матеріалами сучасних інформаційних систем. Вони забезпечують обладнання для зберігання та обробки інформації. Знання того, як працюють комп’ютери та комп’ютерні програми, є важливим для розробки рішень організаційних проблем, але комп’ютери є лише частиною інформаційної системи. Інформаційна система інтегрує комп'ютерні системи, користувачів, дані та процеси для вирішення більш комплексних задач.

Все частіше комп’ютерні послуги та послуги зберігання надаються з хмари, спільних об’єктів, доступ до яких здійснюється через телекомунікаційні мережі.

Комп’ютерне програмне забезпечення поділяється на два великі класи: системне програмне забезпечення та прикладне програмне забезпечення. Основним системним програмним забезпеченням є операційна система. Вона керує обладнанням, даними та програмними файлами та іншими системними ресурсами та надає користувачеві засоби керування комп’ютером, як правило, через графічний інтерфейс користувача (GUI). Прикладне програмне забезпечення – це

програми, призначені для вирішення конкретних завдань користувачів. Прикладом є пакети програм загального призначення з електронними таблицями та текстовими редакторами. Від якісної інтеграції програмного забезпечення залежить швидкість і точність обробки інформації.

Дані є центральним елементом інформаційної системи, оскільки вони становлять інформацію, яка зберігається, обробляється та аналізується для підтримки прийняття рішень. Дані можуть бути структурованими (бази даних, електронні таблиці) та неструктурованими (текстові документи, зображення). Їхня якість, актуальність і безпека критично важливі для успішного функціонування системи. При цьому варто розрізняти поняття «дані» та «інформація».

Дані – це необроблені факти або спостереження, як правило, про фізичні явища чи бізнес-операції. Дані стосуються об’єктивних вимірювань атрибутів (характеристик) об’єктів.

Інформація – це оброблені дані, розміщені в значущому та корисному контексті для кінцевого користувача. Дані піддаються процесу «доданої вартості», коли їхня форма агрегується, маніпулюється та організовується, їхній вміст аналізується та оцінюється та поміщається в належний контекст для користувача.

Таким чином, інформація – це дані, які стали релевантними для конкретної особи для прийняття рішень. Будь-який звіт залишається даними, доки він не буде засвоєний кінцевим користувачем для прийняття рішень [3].

Інформаційна система має три етапи при роботі з даними та інформацією: введення, обробка та виведення. Введення фіксує або збирає необроблені дані всередині організації або з її зовнішнього середовища. Обробка перетворює їх у значущу форму. Виведення передає оброблену інформацію людям, які її використовуватимуть, або процесу, для якого вона використовуватиметься.

Інформаційні системи також вимагають зворотного зв’язку, тобто результату, який повертається відповідним членам організації, щоб допомогти їм оцінити або виправити етап введення.

Потік даних у інформаційній системі схематично зображено на рисунку 1.1.

Рисунок 1.1 – Етапи роботи з даними в інформаційній системі

Мережеві компоненти забезпечують зв’язок між різними елементами інформаційної системи, дозволяючи передавати дані та забезпечувати доступ до ресурсів у локальному (LAN) або глобальному (WAN) масштабі. З’єднання встановлюються через дротове або бездротове середовище. Використання сучасних технологій, таких як хмарні обчислення та віртуалізація, підвищує гнучкість і масштабованість інформаційних систем.

Кваліфіковані люди є одним із найважливіших компонентів будь-якої інформаційної системи. До технічного персоналу входять менеджери з розвитку та операцій, бізнес-аналітики, системні аналітики та дизайнери, адміністратори баз даних, програмісти, спеціалісти з комп’ютерної безпеки та оператори комп’ютерів. Крім того, усі працівники організації повинні бути якнайбільше обізнані щодо можливостей ІС. Їхні навички, досвід та взаємодія з системою визначають ефективність і раціональність її використання.

Процедури використання, експлуатації та підтримки інформаційної системи є частиною її документації.

1.1.2 Типи інформаційних систем

Існує багато різних типів інформаційних систем, адаптованих до конкретних потреб організації. Більшість компаній мають спеціальні інформаційні системи для кожного підрозділу, наприклад відділу кадрів, бухгалтерського обліку чи продажів.

Інформаційні системи можна розділити на три основні категорії: управління, підтримки прийняття рішень і стратегії.

Інформаційні системи управління використовуються для адміністрування внутрішніх процесів компанії, таких як бухгалтерський облік, управління людськими ресурсами та управління запасами. Вони призначені для автоматизації повторюваних завдань і бізнес-процесів. ІС управління часто інтегруються з іншими інформаційними системами, такими як системи планування ресурсів підприємства (ERP), щоб надати особам, які приймають рішення, глобальне уявлення про діяльність компанії.

Інформаційні системи підтримки прийняття рішень надають інформацію та аналіз, щоб допомогти відповідним особам приймати обґрунтовані рішення. Вони призначені для збору та аналізу даних, а потім представлення їх у формі звітів, графіків або інформаційних панелей. Інформаційні системи підтримки прийняття рішень часто використовуються в сферах фінансів, маркетингу та управління ланцюгами поставок.

Системи стратегічної інформації використовуються, щоб допомогти компаніям розробити довгострокові стратегії, надаючи інформацію про ринкові тенденції, конкурентів або звички споживачів. Вони розроблені, щоб допомогти менеджерам приймати важливі рішення, надаючи надійну та точну інформацію, а також дають вказівку на те, у якому напрямку компанія має рухатися, щоб досягти своїх цілей. Стратегічні інформаційні системи часто використовуються стратегічними консалтинговими фірмами [4].

1.1.3 Розробка інформаційної системи

При розробці інформаційної системи використовується одна із двох основних методологій: розробка протягом життєвого циклу або швидка розробка застосунків (RAD).

Недоліком розробки протягом життєвого циклу є тривалий час розробки й об’ємні вимоги до документації. Може статися і таке, що система не відповідатиме вимогам користувача наприкінці довгого шляху розробки.

Все частіше розробка життєвого циклу замінюється RAD. Вона фокусується на швидкому створенні прототипів. Прототип – попередня робоча версія програми, яка будується швидко й недорого, хоча й недосконало. Цей прототип передається користувачам, їхні реакції збираються, запропоновані модифікації включаються, і послідовні версії прототипу зрештою перетворюються на повну систему. Іноді RAD і розробка протягом життєвого циклу поєднуються.

Великі системи, як правило, розробляються та підтримуються через систематичний процес, відомий як життєвий цикл системи, який складається з шести етапів: техніко-економічне обґрунтування, аналіз системи, проєктування системи, програмування та тестування, встановлення та експлуатація та обслуговування. Перші п'ять етапів - це власне розвиток системи, а останній - довгострокова експлуатація. Після певного періоду використання (з обслуговуванням за потреби) інформаційна система може бути виведена з експлуатації або оновлена. У разі серйозного оновлення система переходить в інший життєвий цикл розробки.

Основна мета техніко-економічного обґрунтування полягає в тому, щоб визначити, чи є розробка система доцільною на основі планів щодо термінів користування, стратегічних ініціатив, аналізу витрат і вигод. Аналіз системи дає детальну відповідь на запитання, що буде робити нова система. Наступний етап, проєктування системи, надає розгорнутий план того, як буде організована нова система. На етапі програмування та тестування окремі програмні модулі системи розробляються, тестуються та інтегруються в єдину систему. Подальші рівні тестування забезпечують постійний контроль якості. Встановлення включає остаточне тестування системи в робочому середовищі та перенесення організаційних операцій у нову систему, а також її інтеграцію з іншими вже наявними системами. Пізніші етапи розробки включають такі заходи

впровадження, як навчання користувачів і модифікація організаційних процесів з врахуванням нових можливостей [5].

Після того, як встановлену систему передано її користувачам і оперативному персоналу, вона майже завжди буде значно модифікована протягом терміну служби в процесі, відомому як технічне обслуговування системи. Велику систему зазвичай використовують і обслуговують приблизно від 5 до 10 років або навіть довше. Здебільшого технічне обслуговування полягає в адаптації системи до мінливих потреб організації та нового обладнання або іншого програмного забезпечення. Проте не можна забувати і той факт, що протягом використання системи неминуче будуть знайдені помилки і недоліки у її роботі. Тому технічне обслуговування також передбачає їхнє виправлення.

1.1.4 Безпека інформаційної системи

При роботі з інформацією безпека та контроль повинні бути головним пріоритетом. Безпека стосується процедур і технічних заходів, які використовуються для запобігання несанкціонованому доступу, зміні, крадіжці або фізичному пошкодженню інформаційних систем. Контроль – це методи, політики та організаційні процедури, які забезпечують безпеку, точність і надійність інформації, а також операційне дотримання стандартів управління.

Дані, які зберігаються в електронному вигляді, є вразливими до набагато більшої кількості видів загроз, ніж коли вони існували в паперовому форматі. За допомогою комунікаційних мереж інформаційні системи в різних місцях пов’язані між собою. Можливість неавторизованого доступу, зловживань або шахрайства не обмежується одним місцем, а може виникнути в будь-якій точці доступу в мережі. Рисунок 1.2 ілюструє найпоширеніші загрози для сучасних інформаційних систем.

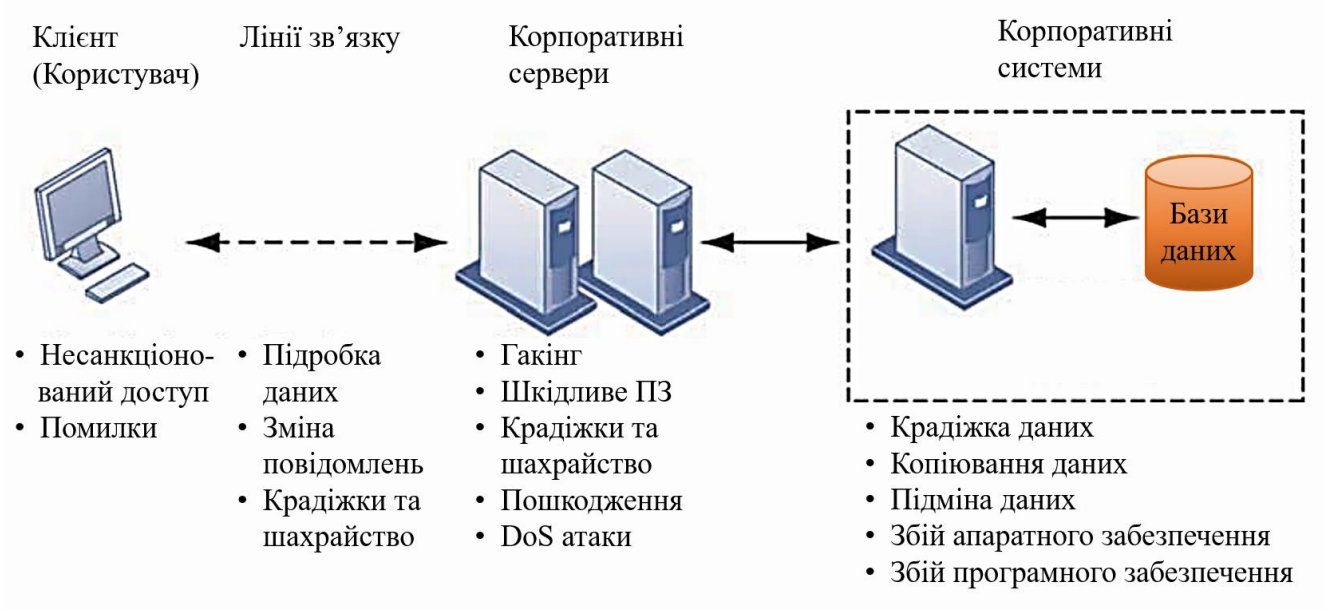


Рисунок 1.2 – Сучасні загрози безпеці та вразливі місця в ІС [2]

У проілюстрованому тут багаторівневому обчислювальному середовищі клієнт-сервер вразливості існують на кожному рівні та в зв'язку між рівнями. Користувачі на клієнтському рівні можуть завдати шкоди, вносячи помилки або звертаючись до систем без авторизації. Можна отримати доступ до даних, що передаються через мережі, викрасти цінні дані під час передачі або змінити повідомлення без авторизації. Зловмисники можуть запускати атаки на відмову в обслуговуванні або зловмисне програмне забезпечення, щоб порушити роботу вебсайтів. Ті, хто здатний проникати в корпоративні системи, можуть знищувати або змінювати корпоративні дані, що зберігаються в базах даних або файлах.

Системи не працюють, якщо апаратне забезпечення комп'ютера виходить з ладу, неправильно налаштоване або пошкоджене внаслідок неправильного використання чи злочинних дій. Помилки в програмному коді, неправильне встановлення або неавторизовані зміни призводять до збою програмного забезпечення комп'ютера. Збої в електроенергії, повені, пожежі чи інші стихійні лиха також можуть порушити роботу комп'ютерних систем.

Партнерство з іншою компанією збільшує вразливість системи, якщо цінна інформація зберігається в мережах і комп'ютерах поза контролем організації. Без

надійних гарантій цінні дані можуть бути втрачені, знищені або потрапити в чужі руки, розкриваючи важливі комерційні таємниці або інформацію, яка порушує особисту конфіденційність.

Популярність смартфонів тільки примножує ці проблеми. Портативність дозволяє легко втратити або вкрасти смартфони, планшети або ноутбуки. Смартфони мають ті самі недоліки безпеки, що й інші Інтернет-пристрої, і вони вразливі до шкідливого програмного забезпечення та проникнення сторонніх осіб. Смартфони, якими користуються співробітники компаній, часто містять конфіденційні дані, такі як дані про продажі, імена клієнтів, номери телефонів і адреси електронної пошти. Зловмисники можуть отримати доступ до внутрішніх корпоративних систем через ці пристрої.

1.2 Визначення користувачів системи

Інформаційна система забезпечення навчально-виховної діяльності катедри повинна враховувати різні категорії користувачів, кожна з яких має свої завдання та вимоги до функціоналу. Основними групами користувачів є студенти, викладачі, викладачі-куратори й адміністратор системи. Для кожної з цих груп передбачено відповідні рівні доступу до даних та можливості в системі, що забезпечує її ефективне функціонування.

Кожен студент має власний особистий кабінет, який зберігає актуальну інформацію про студента. Якщо інформація містить помилку або втратила свою актуальність, студент може надіслати повідомлення про оновлення даних кураторові або адміністратору. Доступ до інформації та функціоналу для студентів є найбільш обмеженим. Студенти можуть здійснювати пошук викладачів у системі, переглядати загальну інформацію про них та їхні контакти. Іншими можливостями є збереження індивідуальних планів, перегляд успішності та комунікація з викладачами. Система містить корисні для студентів посилання, щоб вони могли швидко знайти потрібну інформацію на інших ресурсах університету.

Викладачам надається швидкий і зручний доступ до списків груп і інформації про студентів. Ця група користувачів може переглядати кабінети як студентів, так і викладачів. Пошук може бути здійснений за ім’ям, поштою, групою або дисциплінами. Також викладач може надіслати повідомлення конкретному студентові або одразу всій групі та переглянути, чи воно було ними прочитане.

Викладачі як куратори мають можливість додавати та редагувати інформацію про студентів своєї групи, що дозволяє підтримувати актуальність даних протягом усього навчального процесу. Однією з ключових функцій системи для цієї ролі є можливість обробки файлів з даними про атестацію студентів та автоматичного формування відповідної документації для деканату, що значно спрощує роботу куратора. Куратор також може відслідковувати успішність своєї групи та порівнювати її з успішністю інших груп на катедрі за допомогою графіків і таблиць.

Адміністратор системи має доступ до всього її функціоналу. Під роллю адміністратора розглядається керівник з навчально-виховної роботи на катедрі. Адміністратор може вести облік студентів і викладачів, оновлювати інформацію про них, створювати нові групи, формувати списки кураторів і старост у групах за курсами, планувати та організовувати роботу кураторів, створювати звіти на основі даних, що були внесені до системи, аналізувати і порівнювати інформацію, керувати доступом до різних можливостей системи та забезпечувати її ефективне функціонування.

1.3 Огляд способів реалізації системи

Ефективне використання інформаційної системи значною мірою залежить від вибору способу її розробки та впровадження. Сучасні технології пропонують різні підходи, такі як створення десктопних, веб- чи мобільних застосунків, а також гібридних рішень, які поєднують переваги кількох платформ. Кожен із цих способів має свої особливості, переваги та обмеження, які необхідно враховувати відповідно до вимог користувачів, технічних ресурсів та бізнес-цілей.

Десктопні застосунки встановлюються безпосередньо на комп’ютерах користувачів і забезпечують доступ до функцій навіть без інтернет-з’єднання. Вони характеризуються високою продуктивністю, оскільки використовують локальні апаратні ресурси. Однак такі системи обмежені платформами (Windows, macOS, Linux) і потребують індивідуальних оновлень на кожному пристрої.

Вебзастосунки функціонують через браузери та доступні з будь-якого пристрою, підключеного до Інтернету. Вони легко оновлюються та масштабуються, що дозволяє швидко впроваджувати зміни. Проте якість роботи залежить від стабільності інтернет-з’єднання та серверної інфраструктури.

Мобільні застосунки створюються для смартфонів і планшетів та забезпечують зручність роботи в умовах мобільності. Вони можуть працювати в офлайн-режимі, але потребують розробки окремих версій для Android та iOS, що підвищує вартість проєкту.

Гібридні рішення, такі як Progressive Web App (PWA), поєднують риси веб- та мобільних застосунків. Вони забезпечують доступність з різних пристроїв і платформ, але їхня розробка передбачає використання більших ресурсів та складніших технічних рішень.

Для визначення найкращого способу реалізації необхідно врахувати такі вимоги до системи.

- Функціональні вимоги: система повинна відповідати завданням і вимогам катедри, спосіб реалізації не повинен ставати перешкодою у реалізації цих завдань або призводити до їхнього спрощення з втратою бажаних можливостей. Доступ до даних повинен підтримуватися з різних платформ і пристроїв. Важливою вимогою є можливість інтеграції з іншими системами університету.
- Нефункціональні вимоги: необхідно гарантувати високу продуктивність, масштабованість, безпеку та доступність системи для кінцевих користувачів.
- Економічні вимоги: вибір способу реалізації повинен враховувати обмеження бюджету та витрати на подальшу підтримку.

- Технічні вимоги: рішення повинно бути сумісним з наявною на катедрі інфраструктурою, враховувати потужності пристроїв, на яких буде запущено систему, і забезпечувати легкість розгортання й оновлення.

Таблиця 1.2 – Порівняльна таблиця способів реалізації

Параметр	Десктопний застосунок	Вебзастосунок	Гібридний підхід
1	2	3	4
Доступність	Обмежена пристроєм, де встановлено	Доступ з будь-якого пристрою з браузером	Доступний на всіх платформах
Продуктивність	Висока, залежить від апаратного забезпечення	Залежить від серверної інфраструктури	Висока, оптимізована для платформи
Розгортання	Потребує встановлення на кожен пристрій	Не потребує встановлення	Залежить від вибраного підходу
Вартість розробки	Висока (під кожну ОС окремо)	Середня (одна версія для браузерів)	Висока (через складність інтеграції)
Оновлення	Ручне оновлення для кожного пристрою	Централізоване на сервері	Централізоване або часткове
Масштабованість	Обмежена	Висока, підтримується хмарною архітектурою	Висока
Безпека	Висока, залежить від локальної захищеності	Висока за рахунок серверного контролю	Висока за рахунок інтегрованих рішень

Порівнюючи можливі варіанти реалізації, найбільш підходящим способом реалізації системи є створення вебзастосунку. Такий підхід забезпечує універсальність доступу, легкість інтеграції з іншими платформами, централізоване оновлення та швидший час розробки. Вебзастосунок відповідає ключовим вимогам завдяки можливості масштабування та інтеграції з сучасними хмарними технологіями.

1.4 Визначення основних завдань

Архітектура інформаційної системи зазвичай поділяється на модулі (або компоненти) з метою полегшення розробки, тестування та підтримки системи. Кожен модуль відповідає за конкретну функціональність, що дозволяє зосередитися на окремих частинах системи без необхідності розуміння всієї системи загалом, що полегшує розробку та підтримку коду.

Запропонована архітектура інформаційної системи забезпечення навчально-виховної діяльності катедри зображена на рисунку 1.3.

Рисунок 1.3 – Архітектура інформаційної системи

Користувацький інтерфейс в архітектурі інформаційної системи навчально-виховного напряму катедри є ключовим елементом, який забезпечує взаємодію користувачів з системою. Однією з найголовніших задач інтерфейсу є безпека даних та обмеження доступу до функціоналу для різних категорій користувачів через автентифікацію та авторизацію. Проте доступ до функціоналу повинен контролюватися як зі сторони інтерфейсу, так і зі сторони бізнес-логіки, оскільки користувачі можуть підроблювати дані для спроби несанкціонованого доступу до

конфіденційної інформації та зловживання своїми правами. Такий підхід дозволяє створити більш безпечну систему, яка відповідає вимогам сучасних стандартів безпеки та гарантує правильність роботи.

Модуль управління науково-виховною роботою дозволяє автоматизувати організацію науково-виховної діяльності катедри. Компонент дозволяє складати плани кураторів, встановлювати цілі та завдання і призначати відповідальних за їхнє виконання. Доступними також є засоби для аналізу результатів науково-виховної роботи, формування звітів і статистичної інформації.

Модуль управління навчальним процесом потрібний для планування, організації та контролю за навчальним процесом студентів та викладачів. Компонент надає можливості управління розкладом і дисциплінами на катедрі. Також він забезпечує можливість аналізу успішності студентів, формування звітів щодо навчальних досягнень і складання академічних рейтингів.

Модуль співпраці та комунікації відіграє важливу роль у забезпеченні обміну інформацією та співпраці між усіма учасниками освітнього процесу на катедрі. Модуль забезпечує можливість обміну повідомленнями, документами та навчальними матеріалами між викладачами, студентами та адміністрацією катедри. До цього модулю можна віднести планування та відстеження подій, такі як лекції, семінари, конференції та інші події, що відбуваються на катедрі.

Інформаційний модуль призначений для перегляду детальної інформації про студентів, викладачів і груп на катедрі. Якщо виникає потреба у пошуку за кількома параметрами або за пов’язаною інформацією, тоді варто забезпечити семантичний пошук.

База даних є одним з ключових компонентів архітектури інформаційної системи. Централізована база даних використовується для зберігання, організації та управління даними, які використовуються в системі. Дані у базі даних організовані у вигляді таблиць (реляційна модель), що дозволяє ефективно зберігати та управляти великим обсягом інформації, а також встановлювати зв'язки між різними елементами даних. СКБД повинна дозволяти виконувати складні

запити до даних, щоб отримувати потрібну інформацію, що є важливим фактором для підтримки різноманітних функцій системи, таких як формування звітів або аналіз даних. Нормалізація схеми бази даних є важливою для продуктивності роботи системи та зменшення кількості помилок.

Інформаційна система для навчально-виховного напряму катедри повинна мати можливість інтеграції з іншими системами університету. Використання API дозволяє зручно та обмінюватися даними між різними системами університету, забезпечуючи їх взаємодію та синхронізацію.

Висновки до розділу 1

Інформаційна система є ключовим елементом будь-якої сучасної компанії. Вона дозволяє збирати, зберігати, обробляти та поширювати інформацію всередині організації. Інформаційна система може приймати різні форми: її проєктують відповідно до потреб кожної організації. Впроваджуючи ефективні політики та процедури, компанії можуть максимізувати конкурентну перевагу, яку може надати інформаційна система.

У даному розділі була сформульована мета розробки створюваної інформаційної системи, що також включало визначення її основних задач і ролей, функціональних вимог і потреб користувачів для забезпечення ефективної розробки та повної реалізації вимог з урахуванням усіх бізнес-процесів, пов'язаних із навчально-виховним процесом на катедрі.

Одним із етапів постановки завдання було визначення способу реалізації інформаційної системи. Описавши вимоги до системи та порівнявши способи між собою, можна стверджувати, що вебзастосунок є найбільш оптимальним рішенням для поставленої задачі.

2 АНАЛІЗ НАЯВНИХ СИСТЕМ

Перед розробкою власної інформаційної система варто дослідити аналогічні системи, які існують на ринку, виділити їхні переваги і недоліки, оцінити їхню придатність. Також потрібно провести дослідження типів інформаційних систем, які частково або повністю відповідають вимогам навчально-виховного напряму катедри, і можуть бути вдосконаленими для власних потреб. У підсумку зробимо висновок щодо доцільності вибору певного типу або розробки власного рішення.

2.1 CRM – Customer Relationship Management

CRM (Customer Relationship Management, Управління взаємовідносинами з клієнтами) – це програмне забезпечення, яке спрямоване на підтримку продажів, перетворення потенційних клієнтів на реальних, керування усіма етапами взаємодії з клієнтами.

CRM – це найповніша клієнтська база, адже вся інформація у системі представлена у вигляді наочних карток. Завданням системи управління взаємовідносинами з клієнтами є збір всіх даних про клієнтів із різних каналів на одній платформі. Система впорядковує їх для зручної обробки і перетворює в інформацію. Для кожного клієнта фіксуються: контактні дані, історія звернень (листи, звернення в чатах чи соціальних мережах), дзвінки разом із їхніми записами, подальші завдання, файли (наприклад, договори чи чеки про оплату) [6].

CRM-система – це інструмент, який пропонує низку переваг для бізнесу, які призводять до більш ефективної роботи з клієнтами та сприяють підвищенню результативності. Серед переваг таких систем є забезпечення інструментів для покращення процесів у кожній точці взаємодії, можливість аналітики даних, створення звітності й автоматизація рутинних процесів, що допомагає зекономити час і зосередитись на важливих стратегічних завданнях.

Програмне забезпечення CRM використовується компаніями та галузями будь-якого розміру. Це вигідно великим підприємствам, яким потрібно легко відстежувати діяльність клієнтів в одному місці та ділитися ними між відділами, малим підприємствам, яким часто потрібно робити більше з меншими витратами. Якщо організації потрібно вести облік клієнтів і її співробітники покладаються на інформацію про цих клієнтів, то, незважаючи на галузь чи некомерційність організації, CRM-система буде найкращим рішенням цього питання.

Окрім наявного функціоналу в CRM-системах, зазвичай такі платформи підтримують інтеграцію з готовими застосунками, конекторами та шаблонами для швидшого отримання результату.

Приклад профілю клієнта у CRM-системі зображено на рисунку 2.1. Профіль клієнта містить повну інформацію про клієнта, певну статистику, критерії, записи про комунікацію. Профіль студента та викладача у власній розроблюваній системі повинен мати схожі частини та модулі, які забезпечать повноту інформації та централізований доступ до неї.

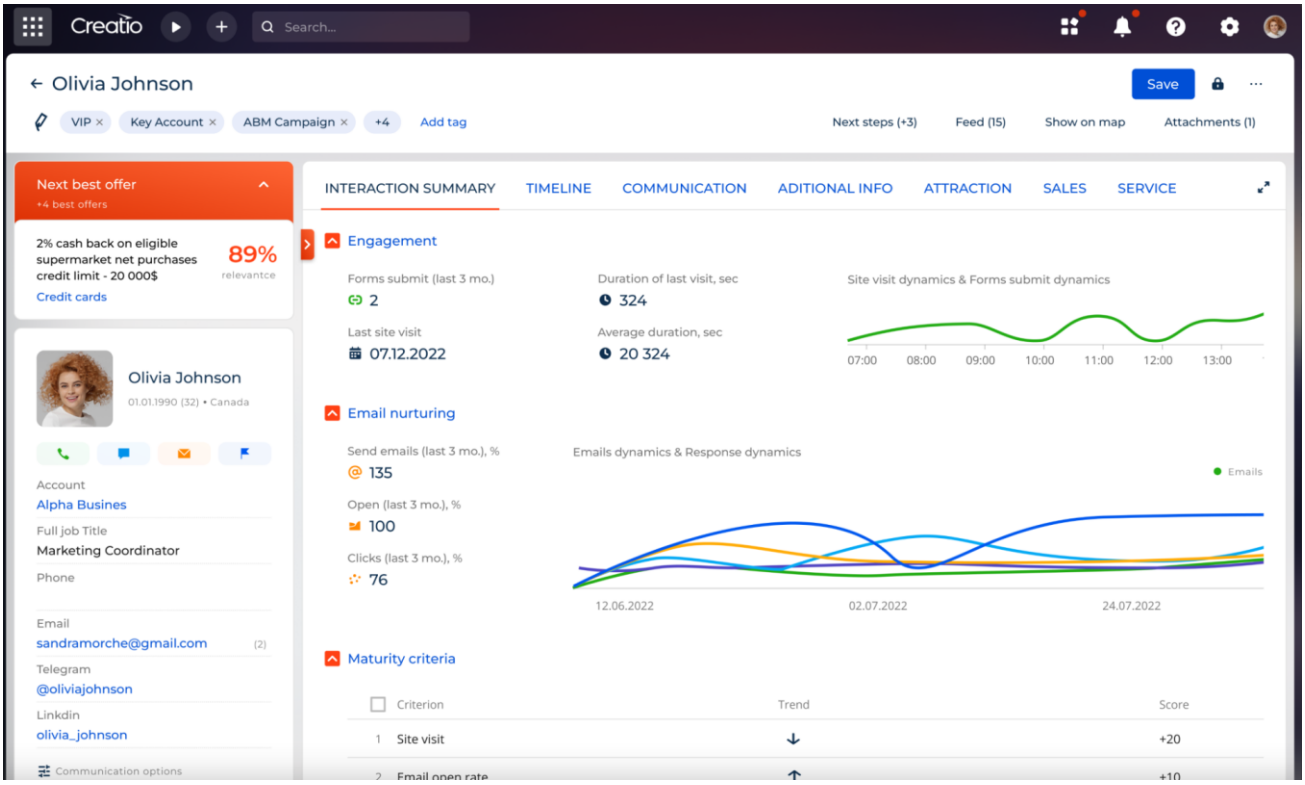


Рисунок 2.1 – Профіль клієнта в CRM-системі «Creatio Sales»

CRM-системи розроблені для комерційної сфери, де основна увага приділяється управлінню відносинами з клієнтами, продажам, маркетингу та обслуговуванню клієнтів. З цього випливає їхня обмежена функціональність для навчального-виховного процесу. CRM не має інтеграції з навчальними платформами або системами оцінювання, що є ключовими функціями для навчальних закладів. У CRM оцінюються бізнес-метрики, такі як ROI (рентабельність інвестицій), кількість продажів або ефективність рекламних кампаній. У навчальній сфері критичні метрики інші: середній бал студентів, успішність атестацій тощо. CRM-системи працюють із поняттями «клієнт», «менеджер» і «компанія», що мало відповідає структурі навчального закладу, де є студенти, викладачі й адміністрація.

Отже, хоча CRM-системи орієнтовані на комерційні процеси і не враховують специфіку академічної діяльності, їхні окремі інструменти можна адаптувати до власних потреб і реалізувати у власній інформаційній системі.

2.2 ERP – Enterprise Resource Planning

Інформаційні системи часто використовують на підприємствах управління діяльністю багатопрофільних підприємств. Одним із прикладів такої системи є системи управління ресурсами підприємства (ERP).

ERP (Enterprise Resource Planning) – це програмне забезпечення, яке дозволяє організаціям автоматизувати та інтегрувати ключові бізнес-процеси, забезпечуючи спільне використання інформації між різними підрозділами. ERP-системи охоплюють управління фінансами, людськими ресурсами, а також іншими адміністративними процесами. Основною перевагою ERP є об'єднання різних аспектів діяльності організації в єдину інформаційну систему, що полегшує управління та прийняття рішень на основі актуальних даних.

Посилаючись на публікацію компанії «Софтінформ», основним видом діяльності якої є комплексна автоматизація систем обліку та управління

підприємств, можемо зробити висновок, що впровадження ERP-систем в університетах може значно покращити ефективність управління та якість навчального процесу. Однак, для успішного впровадження потрібно врахувати специфіку освітнього процесу, забезпечити інтеграцію з існуючими системами, навчити користувачів, забезпечити захист даних та врахувати культурні та організаційні особливості катедри [7].

ERP-системи зазвичай розраховані на інтеграцію всіх процесів великих організацій, таких як університети в цілому, включаючи фінансові операції, управління персоналом та інші великі адміністративні функції. Катедра є лише одним із підрозділів університету, тому повний функціонал ERP-системи може бути надмірним для її потреб.

2.3 LMS – Learning Management System

LMS (Learning Management System) або СУН (система управління навчанням) – це інформаційна система, частіше хмарна, яке дає змогу створювати освітні продукти в електронному вигляді та організовувати онлайн- або дистанційне навчання [8]. Основними користувачами LMS є освітні заклади, онлайн школи та незалежні освітні проєкти. LMS-системи є основним інструментом для онлайн-освіти, але вони можуть бути використані також для підтримки змішаного або традиційного навчання.

Система включає такий функціонал:

- створення навчальних курсів і матеріалів;
- надання доступу до курсів через вебінтерфейс;
- оцінювання результатів навчання за допомогою тестів, завдань або інших інструментів;
- аналіз прогресу учнів за допомогою звітів;
- підтримка комунікації;
- навчання та підвищення кваліфікації співробітників.

Прикладами LMS-систем є Moodle, Google Classroom, Canvas, Blackboard.

Існує можливість розширення функціоналу системи для спеціальних потреб за допомогою плагінів і додатків. На рисунку 2.2 зображено приклад діаграм і звітів, які може переглянути менеджер при використанні LearnerScript – плагіна аналітики навчання для Moodle LMS. При чому такі плагіни можуть надавати особливі можливості для кожного з типів користувачів.

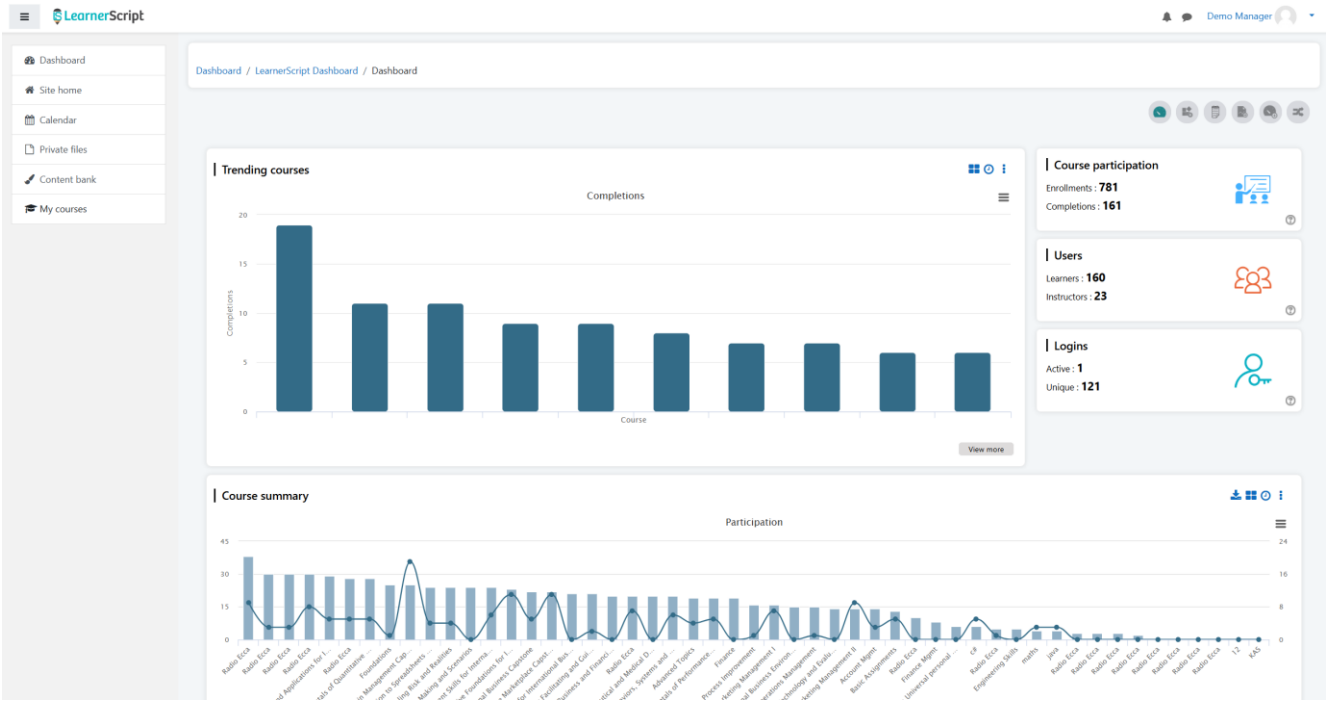


Рисунок 2.2 – Статистичні звіти Moodle для менеджерів

LMS-системи включають зручне адміністрування. Управління курсами, групами студентів та графіками навчання здійснюється в одному місці, що спрощує організацію навчального процесу. Можливості використання системи враховують ведення єдиної бази студентів, адаптацію нових працівників компанії, проведення проміжних контролів.

LMS-платформи допомагають спростити процес навчання як для студентів, так і для викладачів. Ці системи фокусуються виключно на навчальному процесі. Вони не враховують усі потреби у навчально-виховній роботі катедри, яка, окрім навчальних завдань, виконує адміністративні, наукові та виховні функції.

Для забезпечення комплексної діяльності необхідно розробити власне рішення з ширшими можливостями для адміністративної та виховної роботи. Доцільно врахувати найкращі елементи LMS, зокрема автоматизацію моніторингу успішності студентів, інтеграцію мультимедійних матеріалів, надання аналітики та звітів, забезпечення функції комунікації.

2.4 HRMS – Human Resource Management System

HRMS (Human Resource Management System) – це комплекс програмного забезпечення, призначений для автоматизації та оптимізації процесів управління людськими ресурсами в організаціях. Ці інформаційні системи допомагають керувати всіма аспектами життєвого циклу співробітника, від найму та адаптації до обліку робочого часу та нарахування заробітної плати. Завдяки HRMS компанії можуть полегшити повторювані процеси та отримати цінні аналітичні дані для прийняття стратегічних рішень [9].

HRMS використовується в різних організаціях для:

1. обліку персоналу: ведення бази даних про співробітників, включаючи їхні контактні дані, кваліфікацію, досвід роботи (рис. 2.3), що зменшує ручну роботу, пов’язану з обліком співробітників;
2. розрахунку заробітної плати: автоматизація нарахування зарплати, податків і бонусів;
3. управління відпустками та графіками: відстеження днів відпусток, лікарняних або відряджень;
4. оцінки продуктивності: формування індивідуальних KPI та моніторинг їх виконання;
5. розвитку персоналу: організація навчання, тренінгів, підвищення кваліфікації;

6. аналітики HR-процесів: генерація звітів для аналізу тенденцій і оптимізації роботи з персоналом, що забезпечують керівництво аналітичними даними для ефективного планування кадрової політики.

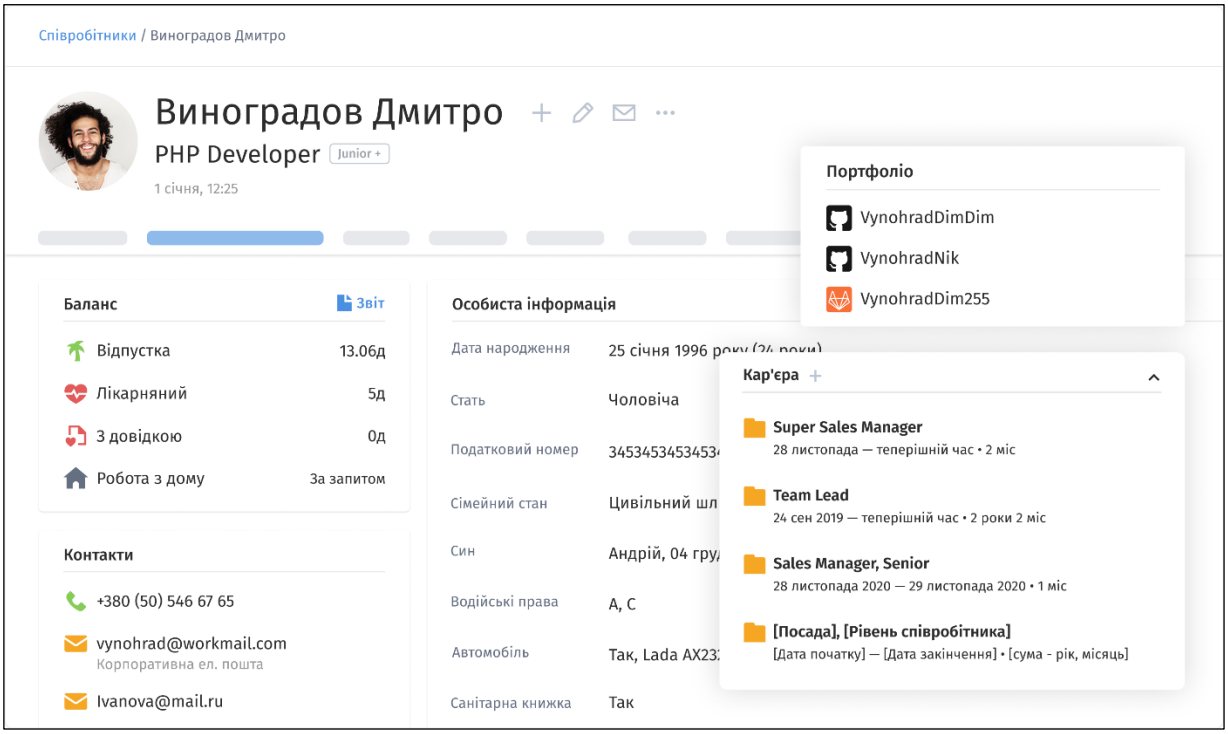


Рисунок 2.3 – Приклад профілю співробітника в HRMS

Сучасні системи управління людськими ресурсами (HRMS) забезпечують автоматизацію документообігу, пов’язаного з управлінням персоналом. До ключових функцій у цій сфері належать створення, зберігання, управління та архівування документів, таких як трудові договори, накази про прийняття чи звільнення співробітників, заяви на відпустки, листки тимчасової непрацездатності та інші супутні документи. Автоматизація документообігу дозволяє суттєво знизити залежність від паперових носіїв. Це допомагає оптимізувати управлінські процеси, адже цифрова обробка документів значно пришвидшує виконання повторюваних завдань і дозволяє зосередитися на стратегічних аспектах роботи з персоналом. Використання стандартизованих шаблонів і автоматичних перевірок даних забезпечує точність та узгодженість інформації. Оцифровані документи

легко зберігати, шукати та обробляти завдяки централізованій базі даних, доступ до якої можна налаштовувати відповідно до рівнів доступу користувачів.

Попри численні переваги, HRMS-системи не задовольняють ключових потреб навчально-виховної діяльності катедри. HRMS розроблена для роботи із співробітниками. Вона не включає функції, потрібні для управління студентами, їхньою успішністю, організацією занять чи виховною роботою. Комплексна HRMS може виявитися дорогою для катедри, хоча більшість її функцій залишаться невикористаними. HRMS оцінює показники ефективності співробітників (KPI, продуктивність), але не враховує освітні чи виховні метрики, такі як успішність студентів чи участь у наукових проєктах.

Впровадження автоматизованого документообігу є доцільним і для інформаційної системи, що розробляється для навчально-виховної діяльності катедри. Така система має також враховувати специфіку академічної роботи, включаючи:

- зберігання індивідуальних планів студента за усі роки навчання;
- зберігання планів кураторів;
- списки кураторів і старост;
- моніторинг академічної успішності студентів через автоматичне формування звітів та відомостей.

Таким чином, інтеграція функцій автоматизованого документообігу у власну інформаційну систему дозволить оптимізувати адміністративні процеси катедри, мінімізувати витрати часу та уникнути помилок у веденні документації.

Висновки до розділу 2

Було розглянуто такі типи систем, які дотичні до поставлених завдань: CRM – система управління взаємовідносинами з клієнтами, LMS – система управління навчанням, ERP – система управління ресурсами і HRMS – система управління персоналом. Переваги і недоліки наведені у таблиці 2.1.

Таблиця 2.1 – Порівняння типів систем

Тип системи	Фокус	Переваги	Недоліки
1	2	3	4
CRM	Управління взаємовідносинами з клієнтами	Широкий спектр інструментів для обліку, простота комунікації	Непридатна для навчального процесу
LMS	Управління навчанням	Онлайн-курси, тестування, моніторинг успішності	Немає функцій адміністрування катедри
ERP	Управління ресурсами	Інтеграція адміністративних процесів	Висока вартість і складність адаптації
HRMS	Управління персоналом	Планування навантаження викладачів, автоматизований документообіг	Орієнтація лише на кадрові процеси

Жоден із розглянутих типів систем не відповідає всім потребам катедри, зокрема інтеграції навчально-виховних та адміністративних процесів.

- CRM підходить для обліку, але не здатна реалізувати усі бізнес-процеси.
- LMS ефективна для організації навчання, проте обмежена у функціях адміністрування.
- ERP має широкий функціонал, але занадто складна й дорога для рівня катедри.
- HRMS є зручною системою для обліку студентів, але система не надає інструментів для управління студентами, їхньою успішністю чи виховною роботою.

З огляду на це, доцільно створити власну інтегровану інформаційну систему, яка поєднуватиме переваги LMS (для управління навчанням), елементів ERP (для адміністрування ресурсів), функцій CRM (для централізованого доступу до даних) і можливостей HRMS (для автоматизованого документообігу). Така система зможе враховувати специфіку катедри та забезпечити її потреби, мінімізуючи недоліки готових рішень.

3 ЗАСОБИ РОЗРОБКИ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Розділ присвячений опису інструментів і технологій, які використовуються для розробки інформаційної системи забезпечення навчально-виховної діяльності катедри. Розглядаються обґрунтування вибору бази даних, мов програмування, фреймворків та інших допоміжних інструментів, які дозволяють створити ефективну систему. Особлива увага приділяється вибору засобів для роботи з даними, реалізації логіки програми, розробки інтерфейсу користувача та гарантування безпеки. Такий підхід забезпечує не лише функціональність та продуктивність системи, а й спрощує її підтримку та подальший розвиток.

3.1 СКБД PostgreSQL

PostgreSQL – відкрита система керування базами даних (СКБД), яка використовується для зберігання, управління та обробки структурованих і неструктурованих даних. Вона підтримує SQL (Structured Query Language) для роботи з даними.

PostgreSQL підтримує створення власних функцій, типів даних та індексів. СКБД підтримує як реляційні, так і об'єктно-реляційні моделі даних, що дозволяє працювати зі складними структурами, такими як JSON, XML, масиви, ієрархії даних. PostgreSQL забезпечує високу продуктивність і може використовуватися як для невеликих застосунків, так і корпоративних систем. Система має інтерфейси для взаємодії з багатьма мовами програмування, такими як Java, Python, C#, та підтримує різноманітні протоколи.

PostgreSQL є повністю відкритим продуктом із ліцензією BSD, що дозволяє використовувати його без додаткових витрат. Завдяки ефективній оптимізації запитів і використанню індексів, PostgreSQL забезпечує швидкий доступ до даних. Підтримка транзакцій з гарантією ACID (атомарність, узгодженість, ізольованість, надійність) забезпечує цілісність даних навіть у разі збою. СКБД включає

вбудовану підтримку для процедурних мов (PL/pgSQL, PL/Python), розширені функції індексації (GIN, GiST, BRIN) та реплікації даних [10].

PostgreSQL підходить для розробки вебзастосунків, аналітичних систем і звітності, систем із великими обсягами даних, таких як системи обліку, CRM та ERP. У контексті створення інформаційної системи забезпечення навчально-виховної діяльності катедри PostgreSQL є ключовим елементом інфраструктури.

PostgreSQL дозволяє створювати складні схеми баз даних для збереження даних про студентів, викладачів, групи та навчальні матеріали. Реалізація ключових обмежень (первинний, зовнішній ключі) гарантує зв'язність і узгодженість даних. СКБД використовується для збереження великих обсягів даних і може з високою швидкістю обробляти дані тисяч користувачів. Інструменти PostgreSQL дозволяють легко масштабувати систему та додавати нові функції та розширення.

PostgreSQL є одним із найкращих виборів для створення сучасних інформаційних систем. Продуктивність і надійність цієї СКБД робить її ідеальним інструментом для проєктів у сфері освіти, зокрема для автоматизації навчально-виховної діяльності катедри.

3.2 Середовище розробки IntelliJ IDEA

IntelliJ IDEA – це інтегроване середовище розробки (IDE) від компанії JetBrains, яке широко використовується для створення програмного забезпечення мовами Java, Kotlin, Groovy, Scala, Python, JavaScript, що робить його універсальним рішенням. Завдяки своїм інструментам і функціям, IntelliJ IDEA забезпечує продуктивний і зручний процес розробки програмних систем.

Програма аналізує контекст коду та пропонує варіанти автодоповнення, що значно скорочує час написання коду. IntelliJ IDEA може виявляти помилки, некоректний код або недоліки в реальному часі, пропонуючи їх швидке виправлення. Середовище підтримує інтеграцію з системами контролю версій: Git,

GitHub, SVN, Mercurial, що спрощує командну роботу над проєктами. Наявними є інструменти для тестування, вбудовані засоби для запуску й аналізу юніт-тестів, що допомагає забезпечити якість програмного забезпечення. IntelliJ IDEA пропонує розширені функції рефакторингу, які дозволяють безпечно змінювати структуру коду, зберігаючи його функціональність. IDE має тісну інтеграцію з популярними фреймворками, такими як Spring, Hibernate, Maven, Gradle, що робить розробку складних програм швидкою та ефективною. IntelliJ IDEA дозволяє підключатися до баз даних, виконувати запити SQL та переглядати структуру бази безпосередньо у середовищі розробки.

При створенні інформаційної системи забезпечення навчально-виховної діяльності катедри IntelliJ IDEA є центральним інструментом. IntelliJ IDEA забезпечує безшовну інтеграцію з фреймворком Spring, дозволяючи швидко налаштовувати контексти, контролери, репозиторії та інші компоненти. IDE дозволяє безпосередньо взаємодіяти з PostgreSQL, перевіряти коректність SQL-запитів і синхронізувати схему бази даних із кодом. Використання плагінів для таких інструментів, як Thymeleaf і JavaScript, полегшує розробку фронтенд-частини системи.

Завдяки оптимізованому інтерфейсу та інструментам автоматизації IntelliJ IDEA дозволяє розробникам швидко виконувати задачі. IDE допомагає орієнтуватися у великих системах завдяки навігації між файлами, класами, методами. IntelliJ IDEA підходить як для досвідчених розробників, так і для початківців, забезпечуючи комфортний старт і професійні інструменти.

3.3 Java і Spring фреймворк

Spring Framework – це фреймворк для розробки корпоративних Java-застосунків, який забезпечує багатофункціональне середовище для побудови продуктивних і надійних програм. Spring є одним із найпоширеніших виборів для

створення сучасних вебсистем завдяки своїй гнучкості, модульності та широкій підтримці різноманітних технологій.

Spring дозволяє використовувати лише ті компоненти, які потрібні для конкретного проєкту, що робить його універсальним і адаптивним. Використання концепції залежностей (Dependency Injection) зменшує обсяг коду та полегшує його тестування [11]. Spring підходить для невеликих застосунків, великих монолітів і мікросервісної архітектури. Spring інтегрується з великою кількістю технологій, таких як Hibernate, Thymeleaf, Kafka, REST, GraphQL, що дозволяє швидко додавати нові функції. Spring має багатий набір документів і прикладів, а також активну спільноту, що допомагає розробникам швидко знаходити рішення.

3.3.1 Spring MVC

Spring MVC (Model-View-Controller) – це компонент Spring фреймворку, призначений для розробки вебзастосунків за архітектурою, спрямованою на розділення бізнес-логіки, обробки запитів і відображення даних, що підвищує модульність, зручність підтримки та розширення програм. Застосунок, архітектура якого заснована на патерні MVC, схематично зображений на рисунку 3.1.

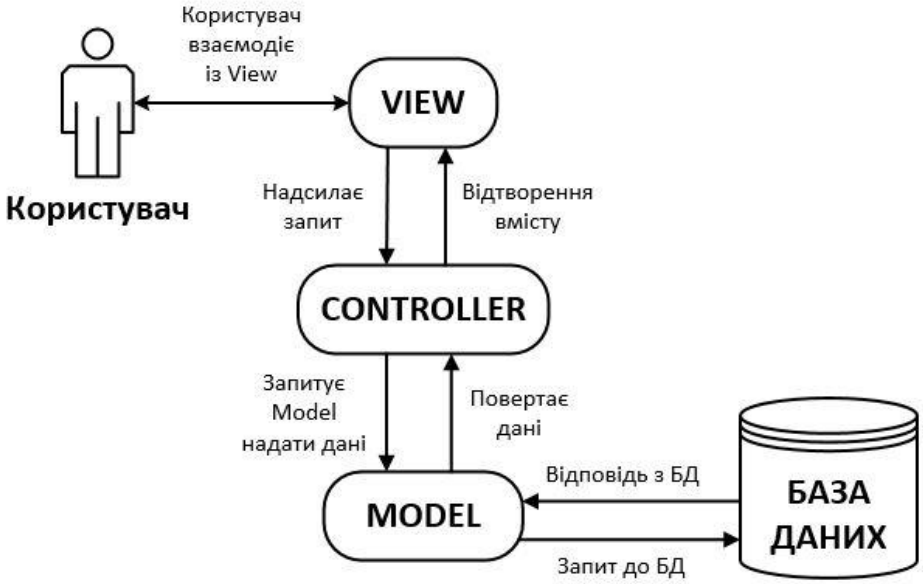


Рисунок 3.1 – Архітектура застосунку заснованого на патерні MVC

Model (Модель) використовується для управління даними застосунку. Вона отримує дані з бази даних (або інших джерел), обробляє їх і передає у потрібному форматі для відображення. У Spring MVC модель представлена об'єктами, сервісами та репозиторіями.

View (Відображення, Вид, Представлення) відповідає за відображення даних користувачеві. Відображенням можуть бути HTML-сторінки, згенеровані за допомогою таких шаблонізаторів, як Thymeleaf, JSP чи інших технологій. Spring MVC дозволяє передавати до представлення дані через об'єкти Model..

Controller (Контролер) приймає HTTP-запити від клієнта, взаємодіє з моделлю для обробки запиту та повертає відповідне відображення.

Центральним компонентом Spring MVC є DispatcherServlet, який приймає всі запити у вебзастосунку і спрямовує їх до відповідних контролерів. Він обробляє запит відповідно до конфігурації маршрутизації. Послідовність обробки запиту розподільчим сервлетом проілюстровано на рисунку 3.2.

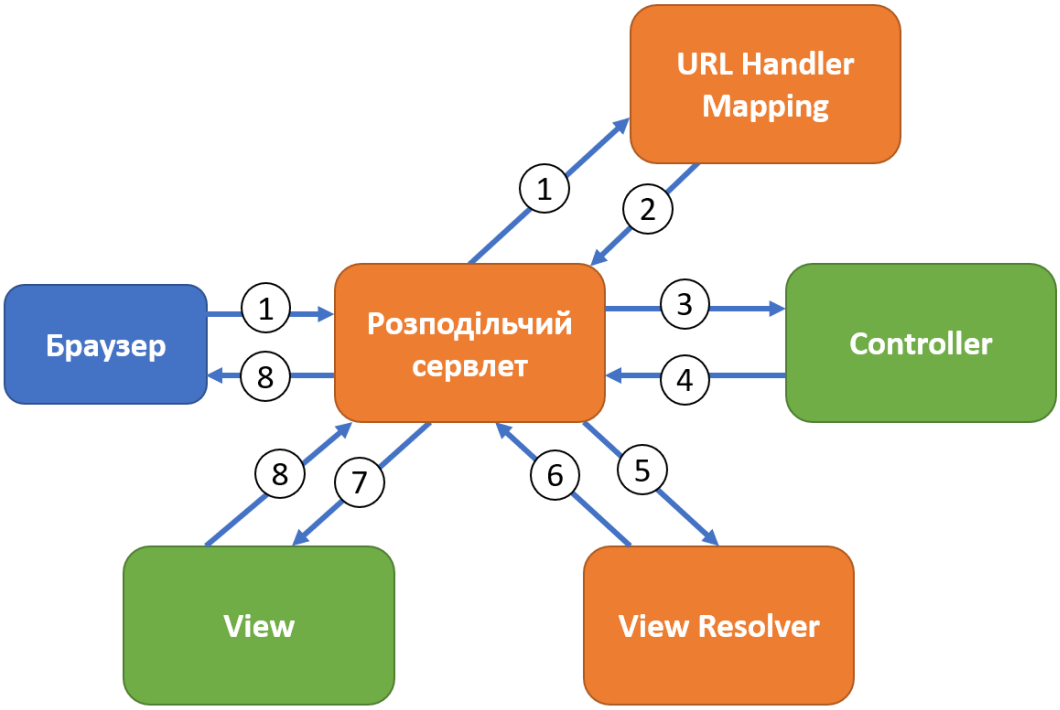


Рисунок 3.2 – Схема послідовності обробки запиту в Spring MVC: 1 – запит, 2 – контролер, 3 – бізнес-логіка, 4 – Model і назва View, 5 – назва View, 6 – View, 7 – об’єкт Model, 8 – відповідь

Користувач надсилає HTTP-запит через браузер. Розподільчий сервлет спрямовує запит до URL Handler Mapping, щоб визначити, який контролер має обробляти запит. Це виконується на основі маршрутизації (мапінгу URL), визначеного в застосунку. Визначивши відповідний контролер, система перенаправляє запит до нього. Контролер – це клас, який містить методи для обробки конкретних запитів.

Контролер обробляє запит, виконує необхідну бізнес-логіку (наприклад, отримання даних із бази) та формує відповідь у вигляді моделі (Model) і передає інструкцію для відображення (View). Потім результат повертається до розподільчого сервлета. Сервлет передає інформацію до View Resolver, який знаходить відповідний файл представлення (наприклад, шаблон у Thymeleaf або JSP) та передає дані моделі для відображення. Система генерує кінцевий вигляд сторінки, використовуючи дані з моделі, та передає результат до розподільчого сервлета, який повертає сформовану вебсторінку у браузер користувача як відповідь на початковий HTTP-запит [12].

У розробці інформаційної системи забезпечення навчально-виховної діяльності катедри Spring MVC використовується для створення інтуїтивно зрозумілих вебінтерфейсів. Сторінки системи, такі як профілі студентів, дані груп або звіти, реалізовані через MVC-компоненти. Контролери (@Controller) обробляють запити користувачів, наприклад, отримання списку студентів або додавання нового викладача. Модель взаємодіє з базою даних через сервіси та репозиторії, виконуючи обробку даних. Відображення (.html) створюються за допомогою Thymeleaf, відображаючи інформацію користувачеві.

3.3.2 Spring Boot

Spring Boot – це розширення фреймворку Spring, яке спрощує розробку, тестування та розгортання застосунків. Основна мета Spring Boot – зменшити

кількість конфігурацій вручну та забезпечити розробнику "готове до роботи" середовище з мінімальними зусиллями.

Spring Boot автоматично налаштовує застосунок на основі залежностей, які є в проєкті. Наприклад, якщо додано залежність до бази даних, Spring Boot налаштовує з'єднання з нею без додаткових налаштувань. Spring Boot інтегрує вбудовані сервери, такі як Tomcat, Jetty чи Undertow. Це дозволяє запускати додаток як звичайний виконуваний файл (jar або war), без необхідності встановлювати окремий сервер.

Spring Boot надає набори залежностей (starters), які дозволяють додавати потрібний функціонал в проєкт без необхідності ручного підбору бібліотек. Наприклад, spring-boot-starter-web для розробки вебзастосунків або spring-boot-starter-data-jpa для роботи з базами даних [13].

У створенні інформаційної системи забезпечення навчально-виховної діяльності катедри Spring Boot використовується як основа для побудови вебзастосунку. Його використання забезпечує швидке налаштування серверної частини через автоматичну конфігурацію, просте підключення до бази даних PostgreSQL за допомогою spring-boot-starter-data-jpa і розробку вебінтерфейсу з використанням spring-boot-starter-thymeleaf. Його використання дозволило зосередитися на логіці бізнес-процесів і функціональних можливостях системи, не витрачаючи час на налаштування інфраструктури.

3.3.3 Spring Data

Spring Data — це модуль Spring, яка надає інструменти для роботи з даними і допомагає спростити доступ до різних джерел даних (SQL, NoSQL, інші) за допомогою високорівневого API, який дозволяє розробникам зосередитися на бізнес-логіці, а не на деталях реалізації запитів до бази даних.

Spring Data забезпечує інтеграцію з багатьма джерелами даних, такими як реляційні бази (PostgreSQL, MySQL), документоорієнтовані бази (MongoDB), графові бази (Neo4j), індексні бази (Elasticsearch) та багато інших.

Розглянемо основні компоненти Spring Data.

1. Spring Data JPA – модуль для роботи з реляційними базами даних через API Java Persistence API (JPA). Завдяки цьому модулю можна автоматизувати виконання CRUD-операцій без написання SQL-запитів. Spring Data JPA інтегрується з популярними ORM-фреймворками, такими як Hibernate.

2. Репозиторії (Repositories) є абстракціями для доступу до даних, що дозволяють створювати методи для запитів на основі іменування (наприклад, findByName або findByEmail). Для цього не потрібно писати SQL-запити вручну.

3. Query Methods: репозиторії дозволяють створювати запити на основі методів із певними іменами. Spring Data автоматично реалізує запит і створить SQL-запит на основі назви методу, що відповідає спеціальному синтаксису [14].

4. JPQL та SQL: для складніших випадків підтримується написання запитів на мові JPQL (Java Persistence Query Language) або навіть нативному SQL. Такі запити можна використовувати через анотацію @Query.

5. Пагінація і сортування: Spring Data забезпечує вбудовану підтримку посторінкового виводу даних і сортування, що дозволяє ефективно працювати з великими наборами даних.

Spring Data допомагає пришвидшити розробку: більшість базових операцій із даними (зберігання, оновлення, видалення, пошук) виконується без написання додаткового коду. Модуль підтримує широкий спектр баз даних, як реляційних, так і NoSQL, дозволяючи легко змінювати джерело даних.

Spring Data легко інтегрується з іншими модулями Spring, такими як Spring Security, Spring Boot, Spring MVC.

У системі забезпечення навчально-виховної діяльності катедри Spring Data відіграє ключову роль у роботі з базою даних PostgreSQL. Використання Spring

Data дозволяє автоматично враховувати обмеження бази даних та мінімізувати ризик помилок.

Spring Data значно спрощує доступ до бази даних і дозволяє розробникам зосередитися на бізнес-логіці, а не на деталях роботи з SQL. Завдяки Spring Data проєкт отримав надійний і зручний інструмент для роботи з даними, що є важливим аспектом для стабільності та продуктивності системи.

3.3.4 Spring Security

Spring Security – це модуль, який забезпечує високий рівень безпеки для вебзастосунків, побудованих на базі Spring фреймворку. Його основними завданнями є автентифікація, авторизація та захист від поширених вразливостей, таких як XSS, CSRF та інші. Spring Security є ключовим компонентом для побудови безпечних і масштабованих систем [15].

Автентифікація відповідає за перевірку особи користувача на основі наданих облікових даних (наприклад, логін і пароль). Spring Security підтримує кілька підходів до автентифікації, зокрема:

- вхід через форму (form-based authentication);
- HTTP Basic і Digest Authentication;
- OAuth2 і SSO (Single Sign-On);
- використання токенів, таких як JWT.

Spring Security обробляє кожен вхідний HTTP-запит через Security Filter Chain. Фільтри виконують завдання автентифікації, авторизації, перевірки сесій та CSRF-токенів. Менеджери автентифікації перевіряють надані користувачем облікові дані, наприклад, шляхом перевірки в базі даних. Інформація про автентифікованого користувача зберігається в контексті безпеки. Цей контекст дозволяє отримати доступ до даних про користувача у будь-якій частині програми.

Spring Security є потужним інструментом для побудови безпечних застосунків. Його гнучкість, інтеграція з іншими модулями Spring і сучасні

механізми захисту дозволяють швидко налаштовувати систему безпеки та адаптувати її до вимог проєкту. У розроблюваній системі Spring Security реалізує як базову автентифікацію користувачів, так і авторизацію для доступу до певних модулів, що забезпечує надійність і захист даних.

3.4 HTML і CSS

Для створення інтерфейсу користувача інформаційної системи були використані мова розмітки гіпертексту HTML (HyperText Markup Language) і мова опису стилів CSS (Cascading Style Sheets). Ці технології є основою веброзробки.

Незважаючи на те, що HTML та CSS з'явилися кілька десятиліть тому, вони досі активно використовуються для створення вебсторінок. Усі сучасні браузери підтримують ці стандарти. Вони лежать в основі кожного вебсайту, незалежно від того, наскільки складний чи технологічно просунутий сайт. Новіші технології, такі як React, Angular чи Vue.js, все одно компілюються в HTML і CSS для відображення у браузері. HTML і CSS стандартизовані організацією W3C, що забезпечує їхню узгодженість у всіх браузерах. Завдяки цьому розробники можуть бути впевнені, що їхній код працюватиме однаково на всіх платформах і пристроях.

HTML слугує основою для створення структури вебсторінок. За допомогою HTML було побудовано каркас інтерфейсу користувача, що включає заголовки, параграфи та списки для відображення текстової інформації, форми для введення даних користувачами (логін, реєстрація, редагування інформації), таблиці для представлення структурованих даних, таких як списки студентів чи результати атестацій, інтерактивні елементи, такі як кнопки, посилання та випадаючі списки.

CSS було використано для представлення зовнішнього вигляду вебсторінок. Завдяки каскадним таблицям стилів вдалося створити красивий дизайн, забезпечуючи зручність використання системи. CSS (каскадна або блокова верстка) прийшла на заміну табличній верстці вебсторінок. Головна перевага блокової верстки - розділення змісту сторінки (даних) та її візуальної презентації

(оформлення) [16]. CSS дозволяє налаштувати кольорові схеми для відображення інформації, використовувати різні шрифти для покращення читабельності, стилізувати кнопки, форми й випадаючі списки, забезпечуючи зручну взаємодію користувача з системою.

У поєднанні HTML і CSS дозволили створити структуровані вебсторінки, що відповідають сучасним вимогам до зручності користувацького досвіду (UX) і дизайну інтерфейсу (UI).

HTML і CSS постійно оновлюються. HTML5 (2008) та CSS3 (2011) значно розширили їхню функціональність, додавши нові елементи, властивості та можливості. Наприклад у HTML5 з'явилися семантичні теги (<header>, <footer>, <section>), мультимедійні елементи (<audio>, <video>), валідація форм без JavaScript. З появою CSS3 стали доступними анімації, градієнти, flexbox, grid, адаптивні медіа-запити для створення дизайну, що підлаштовується під різні пристрої.

Таким чином, HTML і CSS стали ключовими технологіями для реалізації інтерфейсу користувача інформаційної системи, забезпечуючи її функціональність, зручність і привабливий зовнішній вигляд.

3.5 Thymeleaf

Thymeleaf — це серверний шаблонізатор Java, який використовується для створення динамічних вебсторінок. Thymeleaf інтегрується зі Spring фреймворком, зокрема з Spring MVC, і надає зручний спосіб роботи з HTML-шаблонами. Його основна мета — зробити процес створення вебінтерфейсів максимально ефективним, забезпечуючи зручність для як розробників, так і дизайнерів.

Код Thymeleaf виглядає як стандартний HTML із додатковими атрибутами, тому шаблони можна відкривати та редагувати безпосередньо у браузері або редакторах HTML. Це полегшує співпрацю між фронтенд і бекенд та розробниками.

Thymeleaf повністю підтримує Spring MVC, надаючи можливість легко передавати дані з контролерів у шаблони. Шаблонізатор дозволяє додавати динамічний контент у вебсторінки за допомогою спеціальних атрибутів, таких як th:text, th:href, th:each тощо. Існує можливість додавання або приховування елементів залежно від умов із використанням атрибутів th:if та th:unless. Доступним є механізм інтернаціоналізації через файли перекладів, що дозволяє створювати багатомовні застосунки [17].

Thymeleaf підтримує плагіни та користувацькі діалекти, що дозволяє налаштовувати його під потреби конкретного проєкту. Можна створювати повторювані частини сторінки, такі як хедер, футер або меню, і повторно використовувати їх у різних шаблонах за допомогою атрибута th:replace.

Завдяки інтуїтивно зрозумілим атрибутам і природній інтеграції з HTML, розробка є відносно простою. Шаблони Thymeleaf виглядають як звичайні HTML-сторінки, тому їх легко редагувати без спеціальних знань Java. Thymeleaf оптимізований для серверного рендерингу, що робить його швидким і надійним для вебзастосунків.

У системі забезпечення навчально-виховної діяльності катедри Thymeleaf використовується для створення динамічних вебсторінок. Thymeleaf став ключовим інструментом для розробки вебінтерфейсу системи. Завдяки йому вдалося створити зручний, інтерактивний та привабливий інтерфейс. Thymeleaf забезпечує гнучкість, продуктивність та зручність підтримки, що робить його оптимальним вибором для реалізації системи.

Висновки до розділу 3

Вибір інструментів для реалізації інформаційної системи був зумовлений необхідністю забезпечення високої продуктивності, надійності та масштабованості розробки, а також гнучкості у підтримці і подальшому розвитку системи. Використання PostgreSQL як основної системи управління базами даних дозволило

реалізувати стабільне і надійне середовище для зберігання, обробки та маніпуляції даними. Завдяки її підтримці реляційної моделі даних, механізмам транзакційності та відповідності сучасним стандартам інтеграції, PostgreSQL повною мірою задовольнила вимоги, поставлені до бази даних у цьому проєкті.

IntelliJ IDEA, як інтегроване середовище розробки, стало основним інструментом для написання, тестування та рефакторингу коду. Його потужний функціонал, включаючи засоби автодоповнення, інтеграцію з системами контролю версій і підтримку фреймворків, значно скоротив час розробки та мінімізував кількість помилок. Застосування Spring Framework забезпечило модульність і структурованість системи. Зокрема, модулі Spring MVC, Spring Data та Spring Security надали можливість ефективно організувати бізнес-логіку, управління базами даних та механізми автентифікації й авторизації користувачів.

Для створення динамічного вебінтерфейсу було обрано Thymeleaf як шаблонізатор, що забезпечив зручність у роботі з HTML і інтеграцію із Spring MVC. Це дозволило реалізувати інтерактивний інтерфейс користувача з підтримкою адаптивності та забезпеченням високої якості відображення даних. Водночас, система контролю версій Git стала ключовим інструментом для управління змінами у проєкті, забезпечуючи командну співпрацю, резервування даних та аналіз історії змін.

Таким чином, комбінація цих інструментів забезпечила високий рівень продуктивності на кожному етапі розробки. Вони розв’язують комплекс завдань, починаючи від створення фізичної моделі бази даних і закінчуючи інтеграцією бізнес-логіки з користувацьким інтерфейсом. Це підкреслює важливість правильного вибору інструментів у контексті сучасної розробки програмного забезпечення, особливо в умовах необхідності створення масштабованих, продуктивних та інтегрованих систем.

4 ОПИС ПРОГРАМНОЇ СИСТЕМИ

4.1 Фізична модель бази даних

Фізична модель бази даних – це конкретна реалізація структури даних на рівні системи керування базами даних (СКБД). У нашому випадку використовується PostgreSQL, яка дозволяє оптимально організувати зберігання даних і забезпечити їх цілісність, безпечність та ефективність.

Основні компоненти фізичної моделі.

1. Таблиці є основними об’єктами, які використовуються для зберігання даних у базі. Кожна таблиця відповідає сутності концептуальної моделі, наприклад, "Студенти", "Викладачі", "Групи" тощо.

2. Поля описують атрибути сутностей, які зберігаються в таблиці. Наприклад, у таблиці "Студенти" є такі поля, як id, first_name, last_name, group_id, email.

3. Для кожного поля задається тип даних, який визначає, які значення можуть бути збережені. PostgreSQL підтримує числові, рядкові, дата-часові та інші типи.

4. Для забезпечення зв’язків між сутностями використовуються зовнішні ключі. Наприклад, поле group_id у таблиці "Студенти" пов’язане з полем id у таблиці "Групи".

5. Обмеження цілісності використовуються для забезпечення правильності даних. У базі даних використовуються такі обмеження:

- PRIMARY KEY — для унікальної ідентифікації записів;
- FOREIGN KEY — для забезпечення зв’язків між таблицями;
- NOT NULL — для обов’язкових полів;
- UNIQUE — для полів, значення яких мають бути унікальними (наприклад, email).

6. Індекси створюються для підвищення швидкодії запитів, які часто звертаються до певних полів [18]. Наприклад, індекс на полі email таблиці «users» дозволяє швидко знаходити записи за електронною поштою.

На рисунку 4.1 подано фізичну модель розробленої бази даних.

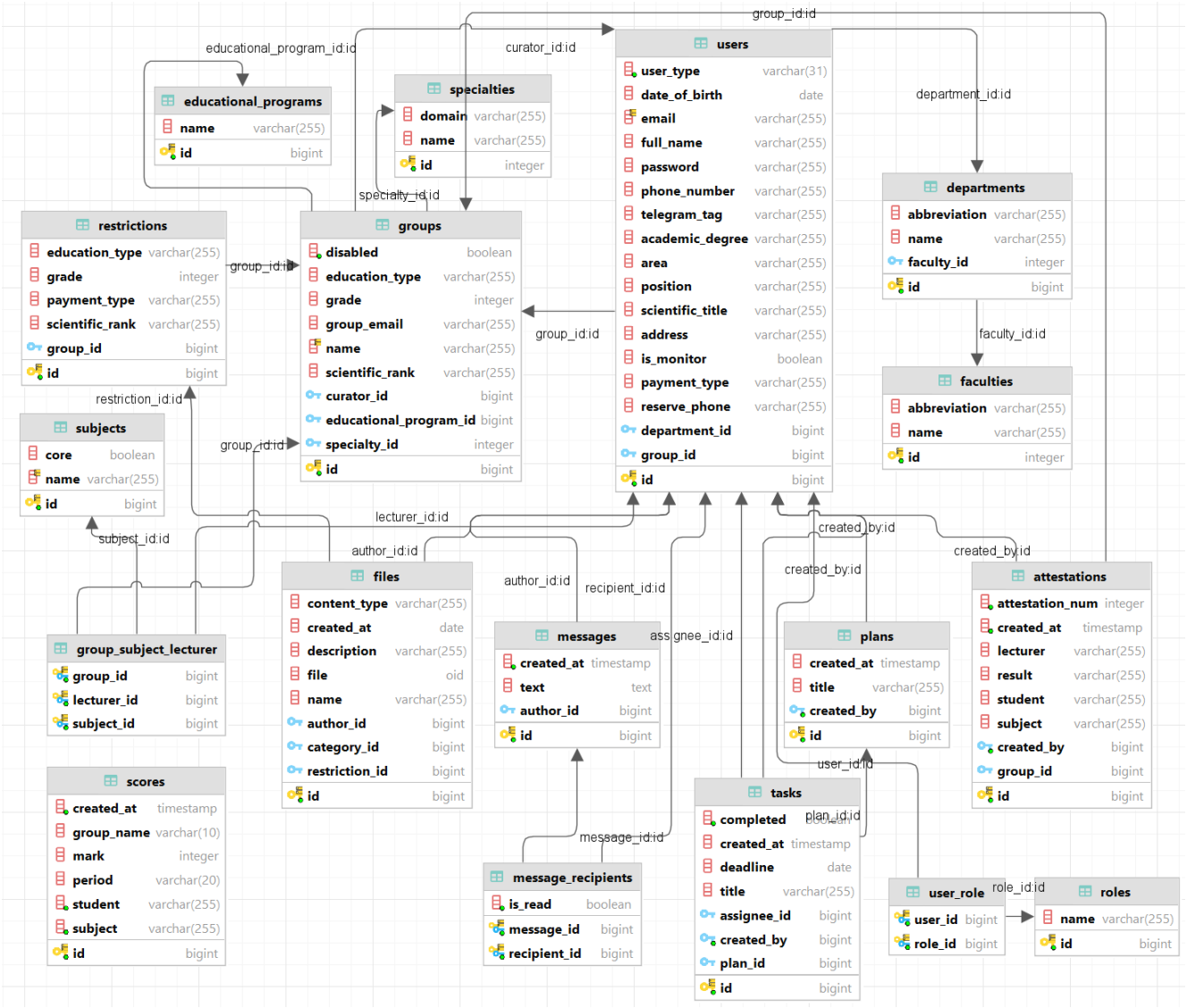


Рисунок 4.1 – Фізична модель бази даних

У результаті в базі даних було створено 17 таблиць, які містять інформацію про користувачів, їхні можливі ролі, факультети та катедри, освітні програми, спеціальності, групи на катедрі, файли, дані про атестацію, плани кураторів і завдання, оцінки та повідомлення.

Структура бази даних побудована за правилами нормалізації, що дозволяє уникнути надлишкового дублювання даних. Індекси створюються для полів, які часто використовуються у фільтрах або сортуванні, таких як email або full_name. Для зовнішніх ключів встановлені каскадні дії при видаленні або оновленні записів, що забезпечує цілісність даних. Для таблиць із великою кількістю записів використовуються механізми секціонування, які дозволяють підвищити швидкодію роботи бази.

4.2 Структура застосунку

Розроблений застосунок побудований на основі багаторівневої архітектури, яка включає такі основні компоненти: моделі, репозиторії, сервіси, контролери та відображення.

Моделі представляють собою основні сутності доменної області системи, які використовуються для роботи з даними. У цьому застосунку моделі реалізовано як класи Java з використанням анотацій JPA, що дозволяє автоматично генерувати структуру бази даних. Кожна модель має свою відповідну таблицю у базі даних. Моделі також включають обмеження цілісності даних через валідатори та зв’язки типу «один до багатьох», «багато до одного» тощо.

Репозиторії відповідають за доступ до бази даних і виконання операцій CRUD (створення, читання, оновлення, видалення). Вони реалізуються як інтерфейси на основі Spring Data JPA, що значно спрощує написання коду доступу до даних. Spring Data JPA дозволяє автоматично генерувати базові методи доступу до даних, а для складніших запитів використовуються спеціальні запити JPQL або SQL [19].

Сервісний шар є проміжним між репозиторіями та контролерами. Основною функцією є інкапсуляція бізнес-логіки. Це забезпечує незалежність бізнес-логіки від деталей доступу до даних. Сервіси також беруть участь у валідації вхідних даних.

Контролери є вхідною точкою для обробки HTTP-запитів. Вони реалізовані за допомогою анотацій Spring MVC і забезпечують взаємодію між користувачем та системою. Контролери також повертають відповідні відображення або JSON-об’єкти у випадку REST-запитів, які використовуються в застосунку для динамічного оновлення інформації на сторінці.

Шар представлення реалізовано за допомогою Thymeleaf, що дозволяє створювати динамічні HTML-сторінки.

Структура вебзастосунку схематично проілюстрована на рисунку 4.2.

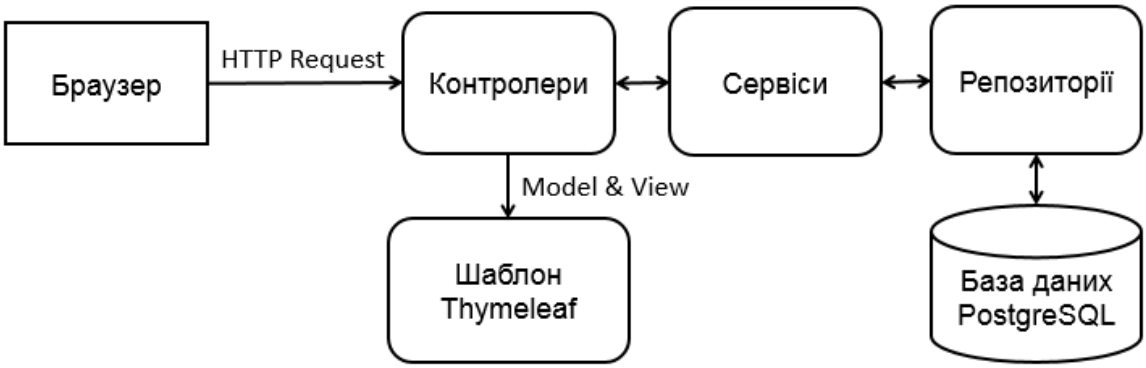


Рисунок 4.2 – Структура застосунку

Компоненти системи працюють у тісній взаємодії.

1. Користувач надсилає запит через сторінку у браузері (View).
2. Запит обробляється контролером, який викликає відповідний сервіс для виконання бізнес-логіки.
3. Сервіс звертається до репозиторія для виконання операцій із базою даних.
4. Результати повертаються через сервіси до контролера, який надсилає відповідь користувачеві.

Окрім основних компонентів, система включає конфігураційні класи, які забезпечують налаштування компонентів.

4.3 Діаграми прецедентів

Діаграми прецедентів (Use Case Diagrams) є одним із ключових засобів моделювання функціональних вимог у розробці інформаційних систем. Вони відображають взаємодію між користувачами (акторами) та системою, визначаючи функції (прецеденти), які система повинна виконувати. У розробці інформаційної системи забезпечення навчально-виховної діяльності катедри діаграми прецедентів були використані для ідентифікації вимог до системи і формалізації її функціоналу.

Актори представляють типи користувачів, які взаємодіють із системою. У розробленій системі актори включають керівника з навчально-виховної діяльності катедри, викладачів, кураторів і студентів. Діаграми прецедентів для кожного із акторів зображені на рисунках 4.3-4.5.

Рисунок 4.3 – Діаграма прецедентів керівника з навчально-виховного напряму

На діаграмі прецедентів керівника з навчально-виховного напряму можемо побачити первинних і вторинних акторів. Первинний актор ініціює взаємодію з системою, в той час як вторинний реагує на деяку взаємодію Вторинні актори використовуються системою, але не ініціюють роботу самостійно [20]. Для даної діаграми це означає, що система «Кампус КПІ» (а саме її API) застосовується тільки тоді, коли керівник імпортує розклад і інформацію про викладачів з наявних систем університету.

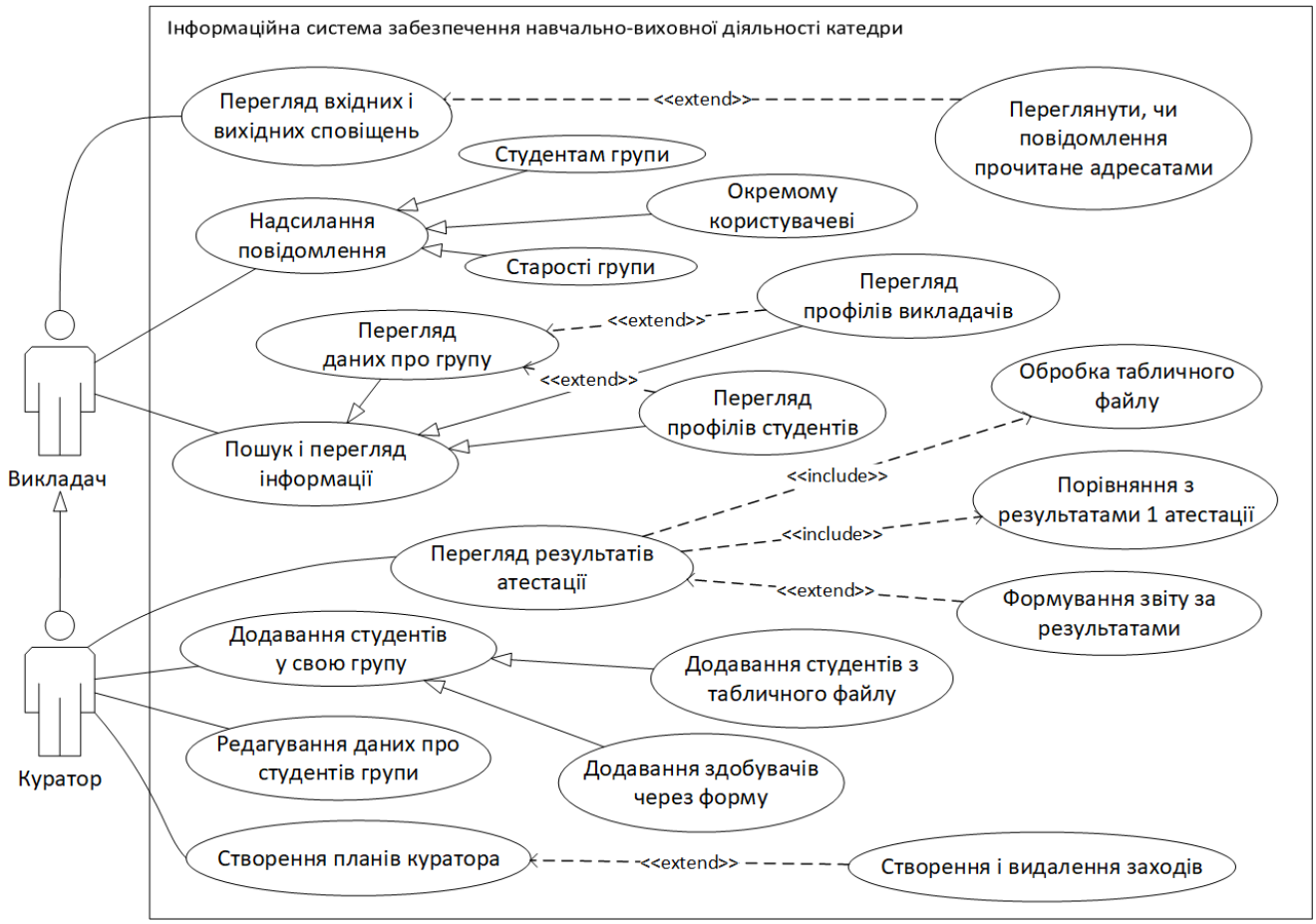


Рисунок 4.4 – Діаграма прецедентів викладача і куратора

Куратор успадковує увесь функціонал, визначений для викладача. Однак куратор має більше функціональних можливостей і ширший доступ до даних. Таке відношення називається узагальненням актора. У цьому випадку викладач є предком і куратор є нащадком.

Рисунок 4.5 – Діаграма прецедентів студента

Діаграми прецедентів стали ключовим інструментом для аналізу та проєктування системи, забезпечивши всебічний огляд її функціональності та взаємодії з користувачами.

4.4 Автентифікація і авторизація

Для початку роботи з системою користувач повинен пройти автентифікацію й авторизацію.

Автентифікація – це процес перевірки особи користувача. Її завданням є підтвердження, що людина, яка намагається отримати доступ до системи, є тією, за кого себе видає. У системі автентифікація здійснюється через введення логіну і пароля. Логіном є електронна пошта користувача. У контексті розробленої системи автентифікація надає початковий доступ до функціоналу: користувач вводить свої дані на сторінці входу, після чого система перевіряє їх у базі даних. Якщо автентифікація пройшла успішно, для користувача створюється сесія.

Авторизація – це процес перевірки прав доступу користувача до певних ресурсів або функцій системи. Іншими словами, навіть якщо користувач успішно пройшов автентифікацію, він може отримати доступ лише до тих даних і функцій, які дозволені його роллю. Авторизація у системі забезпечується через налаштування ролей і прав доступу, які визначаються під час проєктування. Це гарантує, що кожен користувач взаємодіє лише з тими даними та модулями, до яких йому надано доступ.

Доступ до функціоналу системи мають лише зареєстровані та перевірені користувачі, що гарантує безпеку даних і правильність роботи системи. Якщо користувач намагається отримати доступ до будь-якого модуля системи через збережене посилання, але його сесія завершилася (наприклад, через тривалий час неактивності) або він увійшов із нового пристрою, система автоматично перенаправить його на сторінку входу для повторної автентифікації.

Сторінка авторизації має простий і зрозумілий інтерфейс для введення логіна та пароля. Вона зображена на рисунку 4.6.

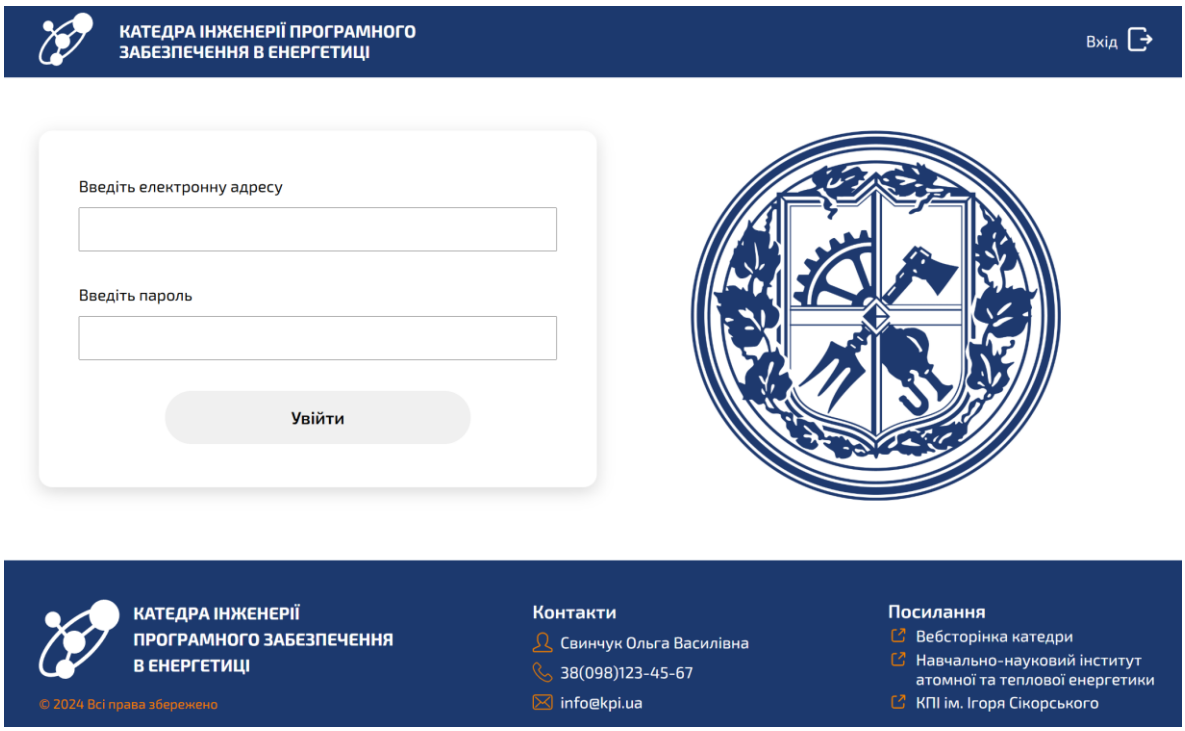


Рисунок 4.6 – Сторінка для входу

Користувач також може вручну вийти із системи, натиснувши кнопку «Вихід» на навігаційній панелі. У цьому випадку сесійні дані про вхід видаляються, і для подальшого доступу до системи користувачу потрібно буде знову ввести логін і пароль.

4.5 Інформаційний модуль

4.5.1 Сторінки для додавання нових користувачів і груп

Сторінка додавання нового викладача є необхідною у системі, оскільки на катедрі та в університеті з’являються нові викладачі. Ця сторінка доступна тільки адміністратору. Дана сторінка зображена на рисунку 4.7. Вона використовується і для редагування даних про наявних у системі викладачів.

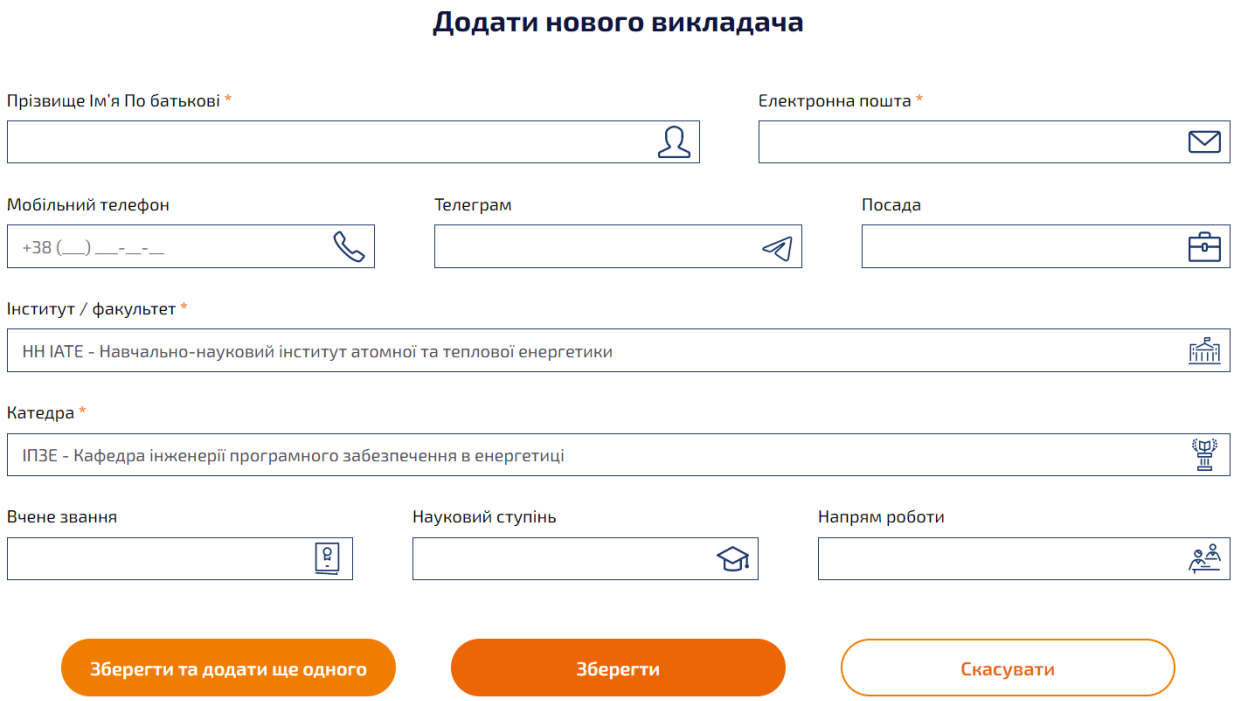


Рисунок 4.7 – Сторінка для додавання викладачів

Форма вимагає введення ПІБ, електронної пошти інституту / факультету та катедри. Поле катедра є випадаючим списком, який оновлюється залежно від вибраного факультету. Поля «Мобільний телефон», «Телеграм», «Посада», «Вчене

звання», «Науковий ступінь» і «Напрям роботи» є необов'язковими для заповнення. Після натиснення кнопки збереження система перевіряє введені дані на відповідність визначеному формату, автоматично генерує пароль і призначає відповідну роль створеному користувачеві. У разі успішної перевірки система інформує адміністратора про успішне додавання викладача або вказує на помилки, які потрібно виправити.

Кожного навчального року додаються нові групи, які також потрібно внести до системи. Для цього керівник може скористатися сторінкою додавання групи, продемонстрованій на рисунку 4.8. Обов'язковими для заповнення є назва групи, спеціальність, форма навчання і номер курсу. Багато полів містять дані за замовчуванням.

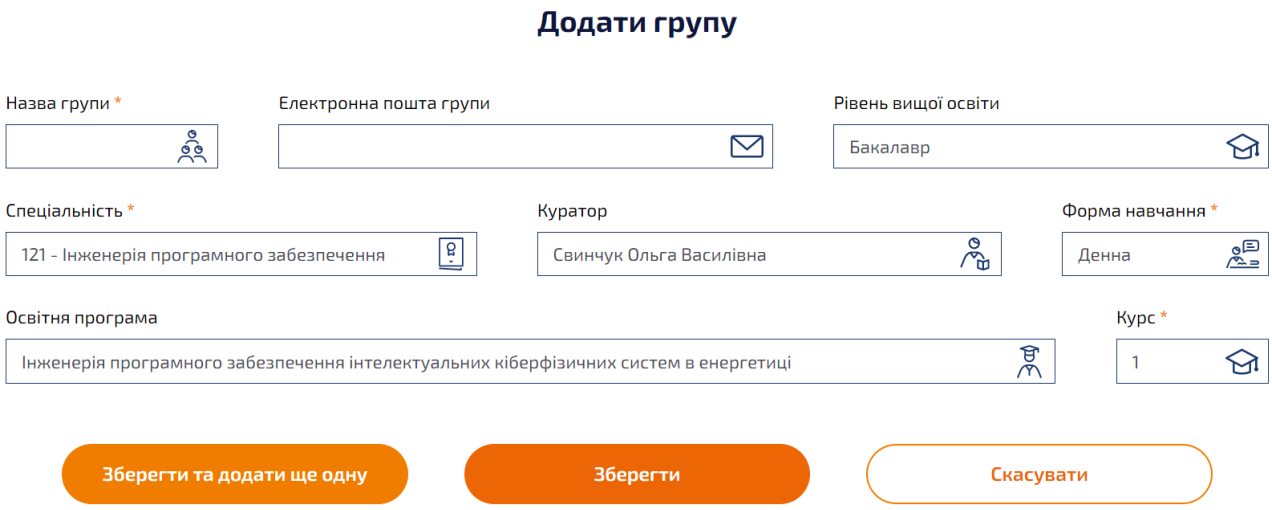


Рисунок 4.8 – Сторінка для додавання групи

Ця сторінка значно спрощує процес адміністрування системи та забезпечує швидке і зручне додавання нових груп. Ця сама форма використовується і для редагування інформації про групу, зокрема для зміни куратора групи.

Для зручного обліку студентів необхідною є сторінка додавання студентів, що зображена на рисунку 4.9. Вона призначена для внесення персональної та академічної інформації про студента і надає простий та зрозумілий інтерфейс для керівника або кураторів.

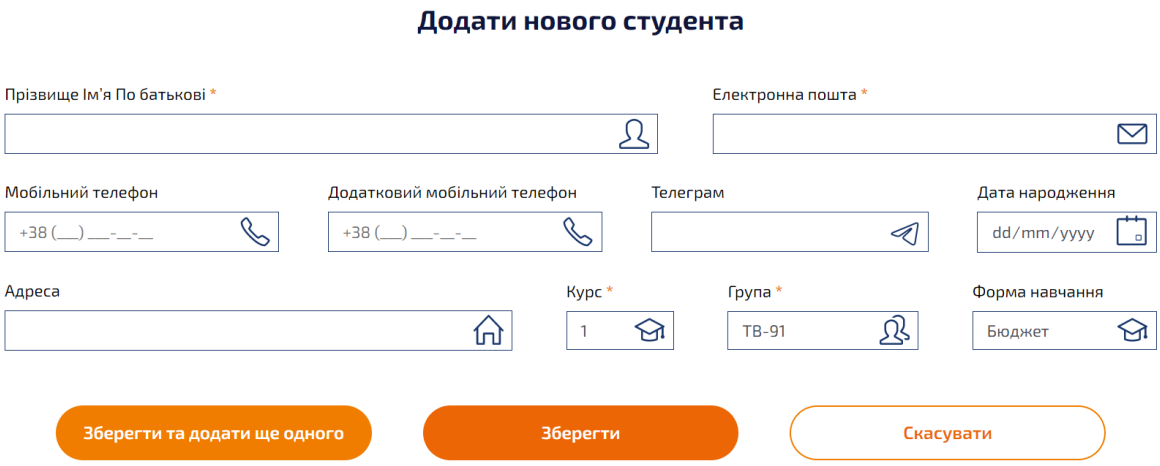


Рисунок 4.9 – Сторінка для додавання студента

Обов’язковими для введення є ПІБ, електронна пошта, курс і група студента. Навчальну групу можна обрати із випадаючого списку. Групи, доступні у списку, залежать від вибраного номера курсу. Контактні дані, дату народження й адресу можна ввести за необхідності. Система виконує автоматичну перевірку введених даних, щоб уникнути помилок. У разі невідповідності формату система вкаже на помилки у даних. При успішному додаванні користувачу автоматично призначається пароль, який куратор і керівник зможуть переглянути у профілі студента.

Існує і інший спосіб додавання студентів. Для цього система пропонує сторінку для додавання студентів із табличного файлу, яка продемонстрована на рисунку 4.10. На ній присутнє повідомлення з очікуваними назвами стовпців для успішного імпорту та правильної обробки.

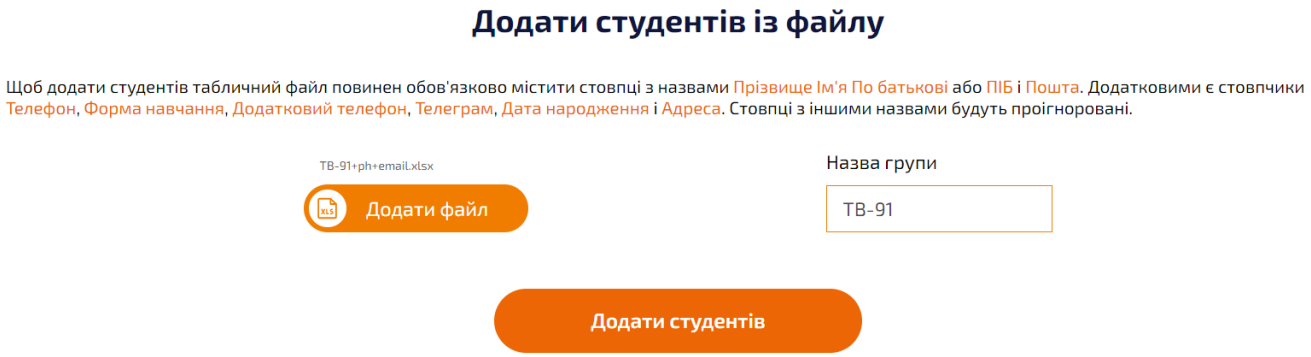


Рисунок 4.10 – Сторінка для додавання даних про студентів із файлу

Щоб додати дані про студентів із табличного файлу потрібно додати сам файл і вказати назву групи, яка при переході зі сторінки групи буде автозаповнена. Якщо файл буде оброблений успішно, система виведе список групи з доданими студентами.

4.5.2 Профіль студента

Профіль студента є центральним елементом для роботи з інформацією про здобувача освіти. На сторінці є кілька блоків, зображених на рисунку 4.11.

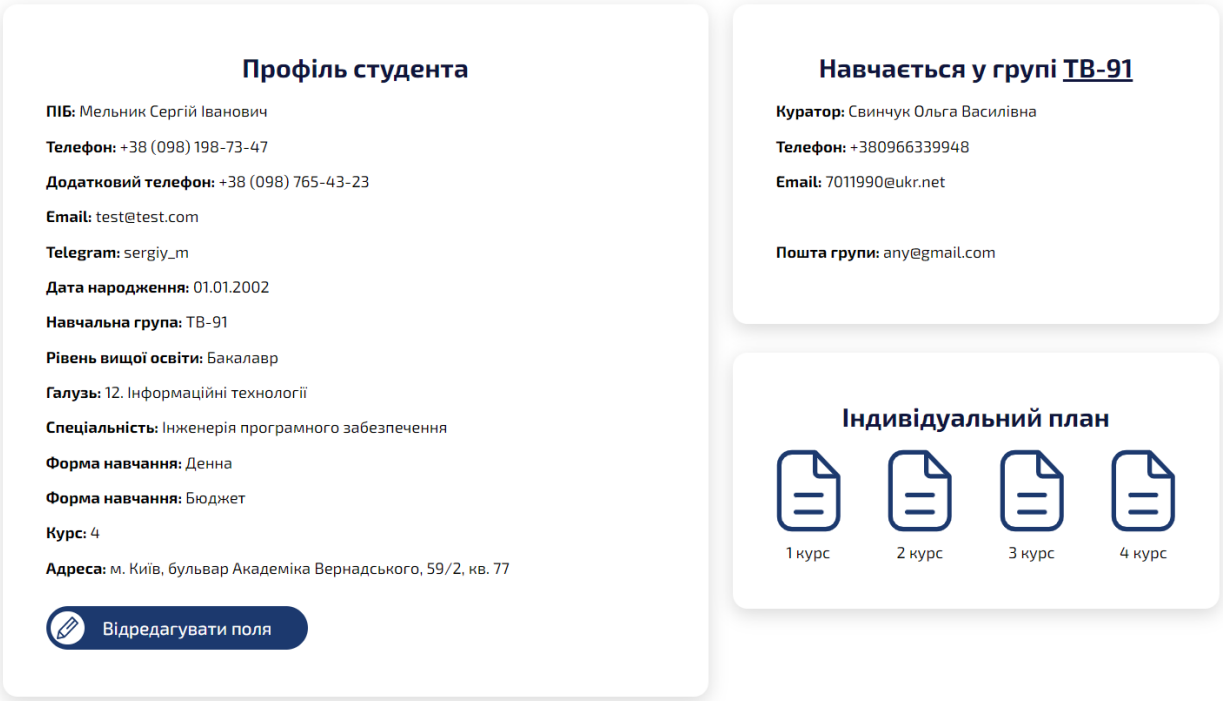


Рисунок 4.11 – Сторінка профілю студента

Блок з назвою «Профіль студента» є основним і демонструє усю інформацію про студента та його контакти. Для куратора і керівника доступна кнопка редагування даних студента.

Інший блок демонструє групу студента і коротку інформацію про неї, зокрема ім’я куратора, телефон і email групи. За швидким посиланням можна переглянути сторінку групи. Також присутній блок з індивідуальними планами студента.

4.5.3 Сторінка групи

Сторінка групи є інструментом для перегляду інформації про групу, включаючи загальні дані, куратора та склад. Прикладом сторінки групи зображений на рисунку 4.12, який демонструє групу і студента, які були створені за допомогою сторінок, описаних раніше.

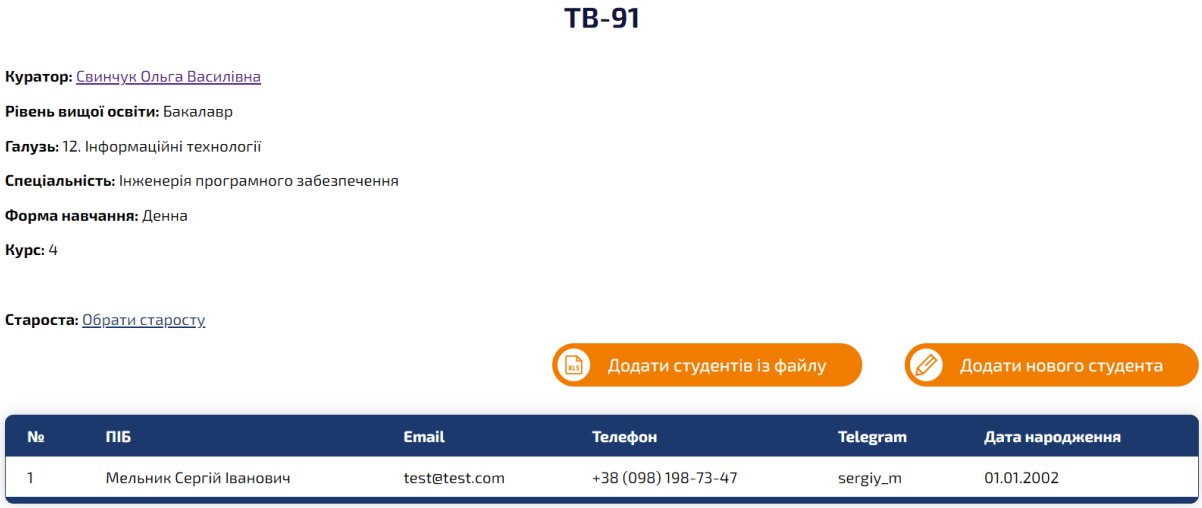


Рисунок 4.12 – Приклад сторінки групи

На сторінці групи передбачена можливість обрання старости. При натисненні на гіперпосилання «Обрати старосту» з'явиться вікно, зображене на рисунку 4.13, зі списком усіх студентів групи. Після натиснення на якогось зі студентів, сторінка оновиться з інформацією про вибраного старосту. Надалі ця інформація буде відображатися для інших користувачів системи.

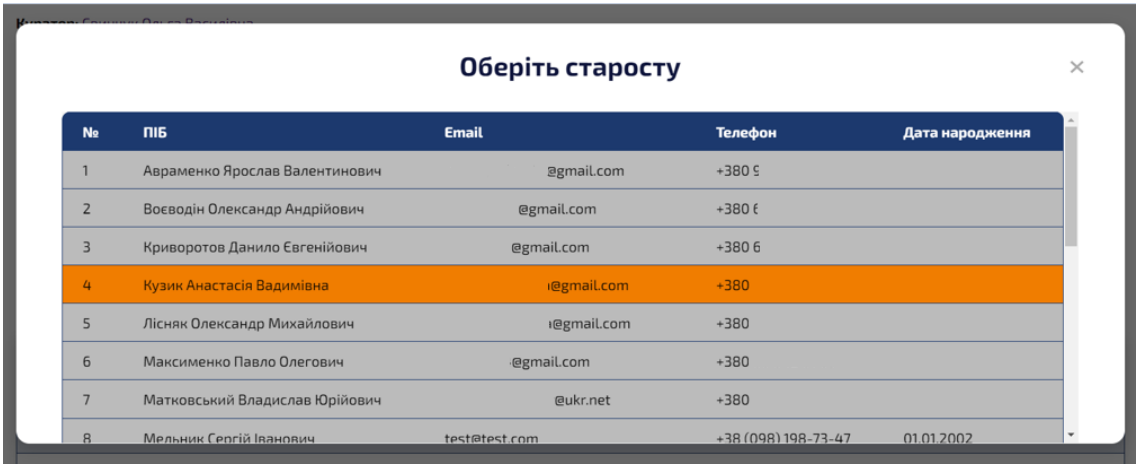


Рисунок 4.13 – Вікно вибору старости в групі

Також сторінка містить кнопки «Додати студентів із файлу» і «Додати нового студента». Вони ведуть на сторінки описані раніше і є доступними тільки для адміністратора та куратора цієї групи.

4.5.4 Сторінка пошуку

Сторінка пошуку використовують для пошуку студентів, викладачів і групи за різними пошуковими параметрами: частина імені, пошта користувача, назва групи. Ця сторінка зображена на рисунку 4.14. Повнота результату та кількість знайдених користувачів залежить від ролі користувача. Наприклад, студент не може здійснювати пошук інших студентів.

Рисунок 4.14 – Сторінка пошуку

Пошук проводиться для усіх категорій одразу: групи, студенти і викладачі. При введенні пошукового запиту система покаже тільки групи, які якось пов’язані з запитом, відповідних студентів і викладачів.

Завдяки інтеграції з системою «Розклад КПІ», пошук можна здійснювати і за дисциплінами, які вивчають студенти. При цьому виведені будуть і викладачі, які викладають ці дисципліни.

Сторінка має мінімалістичний інтерфейс із фокусом на пошуковому полі та результатах. Для швидкого перегляду потрібної категорії на сторінці імплементована пагінація результатів пошуку, щоб сторінка не була занадто довгою, а отже, менш зручною. Результати пошуку є по суті кнопками, які ведуть користувача до сторінки групи, профілю викладача або студента.

4.6 Модуль управління навчально-виховною роботою

4.6.1 Сторінка планів куратора

Кожен куратор має можливість створювати плани куратора. Приклад плану зображено на рисунку 4.15.

ПЛАН-ГРАФІК
роботи куратора групи ТІ-02
на 2023-2024 н.р.

№	Захід	Термін	Відмітка про виконання	Видалити
1	Провести зі студентами он-лайн зустрічі. Довести до відома сайти НТУУ КПІ https://osvita.kpi.ua/index.php/docs https://dnvr.kpi.ua/ https://t.me/dnvr_31 https://kpi.ua/4students	2024-09-01	☑	×
2	Сформувати індивідуальні навчальні плани (ІНП), підготувати і завантажити їх з системи My.kpi, надати для ознайомлення, роздрукування та підписання студентами	2024-09-08	☑	×
3	Надавати студентам інформацію щодо часу та порядку проведення навчальних заходів поточного, календарного та семестрового контролю з надаванням відповідної документації	2024-12-31	☐	×
4	Зібрати зі студентами копії документів для замовлення дипломів та подати в деканат	2024-12-31	☐	×
5	Надавати здобувачам освіти інформацію від завідуючого кафедрою, директора інституту та офіційних осіб університету	2024-12-31	☐	×
6	Проводити аналіз результатів календарного контролю успішності студентів та семестрового контролю	2024-12-31	☐	×

+ Додати завдання

Рисунок 4.15 – Сторінка плану куратора

Куратор має можливість додавати заходи до плану, видаляти їх та редагувати інформацію. Назву плану теж можна змінити, натиснувши на неї. План можна завантажити як документ.

Керівник з навчально-виховної роботи може також назначати заходи іншим кураторам. Куратори отримають сповіщення про нове завдання. Оскільки куратор може мати кілька планів, то він може сам обрати до якого з них додати цей захід. Керівник може переглянути кураторів, які вже виконали призначений захід. Приклад призначеного заходу зі списком виконавців продемонстрований на рисунку 4.16.

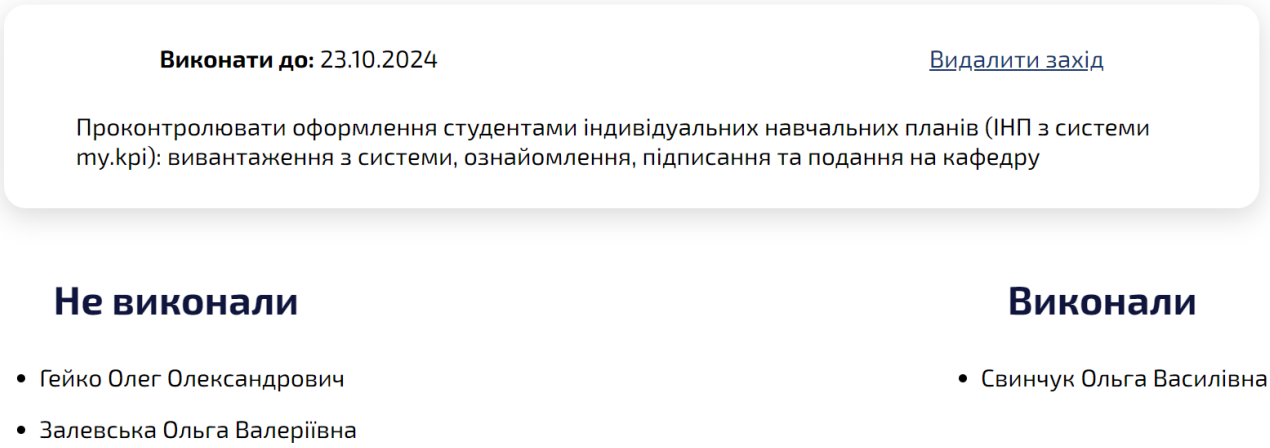


Рисунок 4.16 – Сторінка перегляду призначеного завдання

У разі введення помилкових даних або вибору неправильної категорії кураторів у адміністратора є можливість видалення заходу для усіх виконавців.

4.6.2 Сторінка аналізу академічної успішності

У кінці навчального року або перед захистом дипломної роботи в деканаті формують табличний файл з оцінками для кожного студента протягом навчання. Приклад вхідного файлу зображений на рисунку 4.17.

	A	B	C	D	E	F	G	H	
1	**ТВ-32мп**	Семестри	Години	Кредити	**Марченко**		**Шевченко**		
2		1	150	5	Інженерія даних та знань	90	Інженерія даних та знань	82	Інженері
3		1	135	4.5	Інноваційний менеджмент та інтелектуа	64	Інноваційний менеджмент та інтеле	66	Інноваці
4		1	180	6	Інтелектуальний аналіз даних для задач	93	Інтелектуальний аналіз даних для	90	Інтелект
5		1	60	2	Науково-дослідна робота за темою магіс	95	Науково-дослідна робота за темою	97	Науково
6		1	150	5	Розробка застосунків Інтернету речей та	85	Розробка застосунків Інтернету реч	93	Розробк
7		1	30	1	Розробка застосунків Інтернету речей та	86	Розробка застосунків Інтернету реч	93	Розробк
8		1	150	5	Хмарні та Грід-технології	74	Хмарні та Грід-технології	74	Хмарні т
9									
10	**2023-2024**	**2**	**150**	**5**	**Візуалізація статистичних потокових д**	**75**	**Візуалізація статистичних потоко**	**85**	**Візуаліз**
11		**2**	**150**	**5**	**Графові бази даних**	**95**	**Графові бази даних**	**96**	**Графов**
12		**2**	**120**	**4**	**Методи та засоби виявлення уразливо**	**95**	**Методи та засоби виявлення ура**	**95**	**Методи**
13		**2**	**120**	**4**	**Методологія інженерії програмного за**	**60**	**Методологія інженерії програмно**	**70**	**Методо**
14		**2**	**45**	**1.5**	**Методологія інженерії програмного за**	**60**	**Методологія інженерії програмно**	**70**	**Методо**
15		**2**	**60**	**2**	**Науково-дослідна робота за темою ма**	**82**	**Науково-дослідна робота за темо**	**82**	**Науково**
16		**2**	**150**	**5**	**Оброблення надвеликих масивів дан**	**66**	**Оброблення надвеликих масивів**	**86**	**Обробл**
17		**2**	**90**	**3**	**Практичний курс іноземної мови для**	**66**	**Практичний курс іноземної мови**	**85**	**Практич**
18		**2**	**120**	**4**	**Проектування інформаційно-діагності**	**60**	**Проектування інформаційно-діаг**	**70**	**Проекту**
19		**2**	**60**	**2**	**Сталий інноваційний розвиток**	**75**	**Сталий інноваційний розвиток**	**80**	**Сталий**
20									
21	2024-2025	3	420	14	Практика	95		97	
22		3	360	12					

Рисунок 4.17. – Список оцінок студентів групи протягом періоду навчання

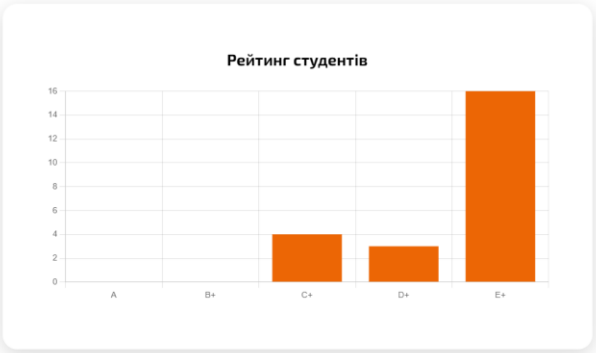
Керівник з навчально-виховної роботи може додати цей файл до системи. При цьому обраховується кількість і частка отриманих оцінок, середній бал і кількість відмінників. Після обробки на пристрій користувача завантажується табличний файл з відомостями про результати навчання студентів. Приклад цього файлу зображений на рисунку 5.18.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	**ВІДОМОСТІ ПРО РЕЗУЛЬТАТИ НАВЧАННЯ СТУДЕНТІВ, ДОПУЩЕНИХ ДО АТЕСТАЦІЇ,**													
2														
3	**в навчально-науковому інституті атомної та теплової енергетики**													
4	**за освітньо-професійною програмою «Інженерія програмного забезпечення інтелектуальних кібер-фізичних систем в енергетиці»**													
5	**спеціальності 121 «Інженерія програмного забезпечення»**													
6														
7	Зазначені нижче здобувачі вищої освіти повністю виконали індивідуальні навчальні плани і отримали оцінки:													
9	№ з/п	Прізвище, ім'я, по батькові	Кількість і частка отриманих оцінок										Частка оцінок «відмінно»	Частка оцінок «задовільно» та «достатньо»
10			«відмінно»		«дуже добре»		«добре»		«задовільно»		«достатньо»			
11			100-95 балів		94-85 балів		84-75 балів		74-65 балів		64-60 балів			
12	1	Бараненко Василь Семенович	33	54%	18	30%	5	8%	5	8%	0	0%	54%	8%

Рисунок 4.18 – Сформований файл із відомостями про результати навчання

Після формування файлу з відомостями на сторінці також з'являється аналіз та аналітика академічної успішності групи. Сторінка містить рейтинг студентів, середній бал по дисциплінах у групі, порівняльні таблиці та діаграми. Для кожного студента обраховується середній бал за кожний навчальний рік окремо, середній бал за усі роки та відхилення від середнього балу по курсу. Студенти групуються

за успішністю. На діаграмі, представленій на рисунку 4.19, можна переглянути, як змінювався середній бал студентів протягом навчання.



Середній бал по дисциплінах

Навчальний рік	Предмет	Середній бал
2020-2021	Основи програмування 2. Модульне програмування	81,91
2020-2021	Основи програмування 1. Базові конструкції	81,96

Рисунок 4.19 – Частина сторінки аналізу академічної успішності

На основі того, як змінювався середній бал студентів протягом навчальних років для групи вираховується тенденція до покращення або погіршення. Для усіх дисциплін обраховується середній бал. Це дає змогу проаналізувати, які дисципліни є більш складними для студентів у кожному навчальному році.

4.7 Модуль співпраці та комунікації

Модуль співпраці та комунікації розроблений для забезпечення ефективної взаємодії між усіма учасниками навчально-виховного процесу. Він використовується для спрощення обміну інформацією, оповіщення та забезпечення зручного доступу до важливих ресурсів і матеріалів. Цей модуль особливо важливий для взаємодії студентів, викладачів, кураторів та керівника навчально-виховної діяльності катедри.

Система забезпечує можливість внутрішньої комунікації між користувачами через сторінку сповіщень. Користувачі об’єднані за певними властивостями. Наприклад, одне повідомлення можна надіслати усім студентам певної групи або

кураторам певного курсу. Це дозволяє уникнути ручного дублювання одного й того самого повідомлення великій кількості користувачів.

Сторінка сповіщень із прикладами повідомлень продемонстрована на рисунку 4.20.

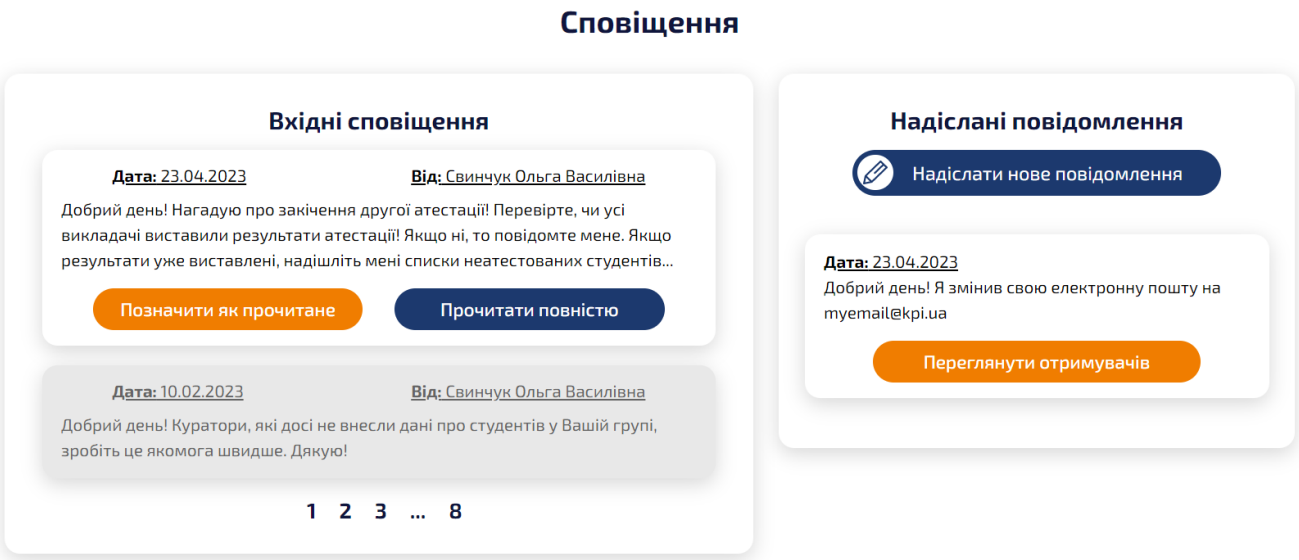


Рисунок 4.20 – Сторінка перегляду сповіщень

Відправник може переглянути отримувачів сповіщення. Адресат може позначити повідомлення як прочитане, що також відобразиться адресанту цього повідомлення. У разі потреби відправник може видалити повідомлення для всіх його отримувачів.

4.8 Модуль управління навчальним процесом

4.8.1 Сторінка атестації

Система надає можливість обробки табличного файлу з даними про атестації студентів у групі. Доступ до цієї сторінки має тільки куратор і адміністратор. Табличний файл можна завантажити в системі «Кампус КПІ». Після прикріплення цього файлу, введення назви групи та вибору номеру атестації на сторінку виводиться таблиця з результатами аналізу. Інтерфейс цієї сторінки і приклад

неатестованого студента наведений на рисунку 4.21. Також, якщо обрати другу атестацію, система може відобразити, чи є співпадіння з результатами першої атестації.

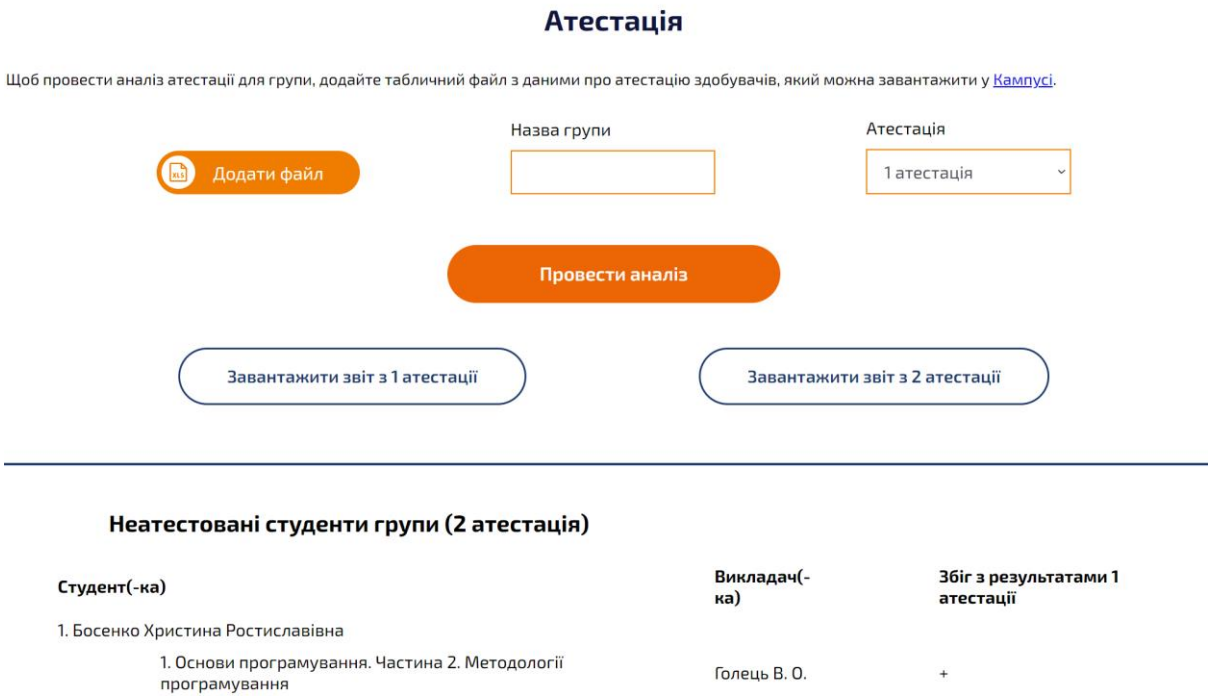


Рисунок 4.21 – Сторінка для аналізу файлу з результатами атестації

Окрім перегляду неатестованих студентів для кожної групи окремо існує можливість завантаження звітів з першої і другої атестацій. Там наведена загальна статистика по курсах щодо кількості неатестованих студентів. Звіт містить таблицю з усіма неатестованими студентами у кожній групі катедри. Для кожного студента наведені дисципліни, з яких він має неатестацію. У звіті з другої атестації також є порівняння з результатами першої атестації.

Після додавання табличних файлів до системи дані можна проаналізувати та переглянути у вигляді діаграм поданих на рисунку 4.22.

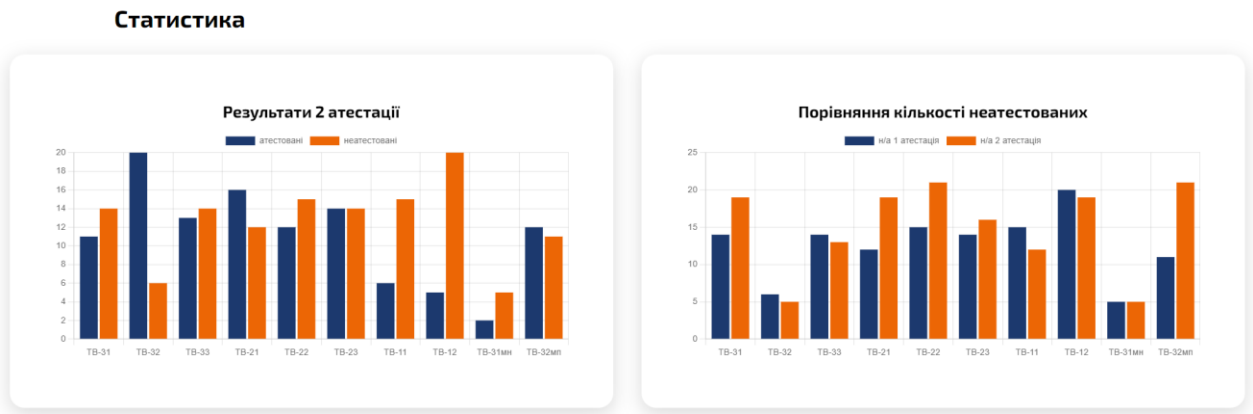


Рисунок 4.22 – Перегляд статистики по атестаціях

На першій діаграмі можна побачити кількість атестованих і неатестованих студентів у кожній групі за результатами останньої атестації. Друга діаграма демонструє порівняння кількості неатестованих студентів у кожній групі за першу і другу атестацію.

4.8.2 Сторінка імпорту розкладу

Інформаційна система підтримує інтеграцію з іншими системами університету. Керівник може імпортувати дисципліни, які вивчають студенти кожної групи катедри з системи «Розклад КПІ». При цьому використовується API системи «Кампус КПІ». Для цього існує сторінка дисциплін, зображена на рисунку 4.23.

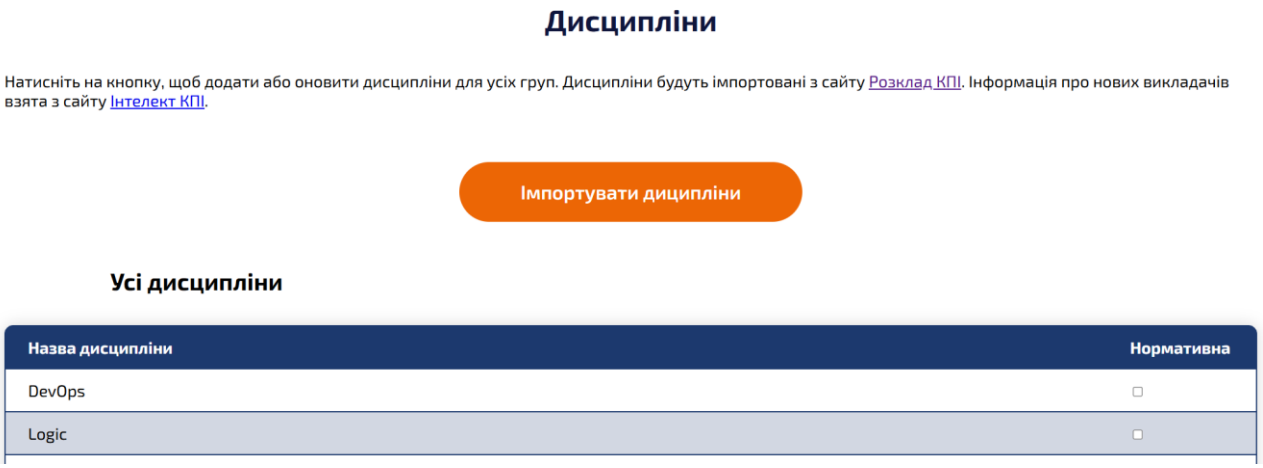


Рисунок 4.23 – Сторінка дисциплін

Окрім імпорту дисциплін імпортуються і викладачі інших катедр і факультетів. Щоб не перезаписувати наявних викладачів, система перевіряє, чи викладач з таким ім’ям вже існує в системі. Якщо ні, то він буде імпортований з даними, наявними в системі «Інтелект КПІ». Зазвичай там зберігається катедра і факультет викладача, його контактні дані, посада, вчене звання, тобто всі ті дані, якими оперує розроблена система. Після імпорту нових викладачів, вони з’являться у списку викладачів на сторінці пошуку.

Оскільки викладачі імпортуються одразу при імпорті розкладу, то на сторінці профілю викладача можна одразу переглянути і дисципліни, які він викладає.

Якщо виникає потреба у оновленні розкладу, керівник може оновити всі зв’язки між групами, викладачами і дисциплінами, натиснувши кнопку «Імпортувати дисципліни» на сторінці дисциплін.

Окрім імпорту розкладу і встановленні зв’язків на сторінці є список усіх дисциплін. Керівник може позначити, чи є дисципліна нормативною, чи вибірковою. Це допоможе у подальшій аналітиці.

Висновки до розділу 4

Розділ описує програмну реалізацію і технічні аспекти розробки інформаційної системи. Застосунок побудований на основі сучасних підходів до розробки програмного забезпечення з чітким розділенням логіки. Це забезпечує легкість масштабування, тестування та підтримки системи. Описані компоненти дозволяють реалізовувати бізнес-логіку, ефективно взаємодіяти з базою даних та надавати зручний інтерфейс для користувачів.

Зображення діаграм прецедентів і опис частини сторінок системи забезпечує уявлення про її інтерфейс та функціональність. Кожна сторінка була розроблена з урахуванням потреб усіх категорій користувачів. Інтуїтивно зрозумілий дизайн гарантує простоту роботи з системою та високу продуктивність її використання.

5 РОЗРОБКА СТАРТАП-ПРОЄКТУ

Розробка стартап-проєкту є важливим етапом створення інноваційного продукту, спрямованого на вирішення конкретних проблем або задоволення потреб ринку. Цей процес включає формування унікальної ідеї, аналіз ринкових умов, оцінку технологічних можливостей, розроблення стратегії просування та планування маркетингових заходів. Особливо актуальним є впровадження стартап-проєктів у сфері освіти, яка потребує нових підходів для оптимізації роботи й автоматизації притаманних їй бізнес-процесів.

Розділ присвячений деталізації ключових аспектів розробки стартап-проєкту. Він розкриває основну ідею, виконує технологічний авдит, аналізує ринкові можливості, формулює стратегію виходу на ринок та описує маркетингову програму. Завданням є обґрунтування доцільності впровадження проєкту та визначення шляхів його успішного просування.

Запропонований стартап-проєкт, орієнтований на навчально-виховну діяльність катедри. Він може стати ефективним рішенням для освітніх установ, забезпечуючи їхню відповідність інноваційним тенденціям і розвитку інформаційних технологій.

5.1 Опис ідеї проєкту

Ідея проєкту полягає у створенні інноваційного рішення, яке задовольнить певну потребу цільової авдиторії. Ключовими елементами ідеї є унікальна цінність продукту, яка виділяє його серед конкурентів, та адаптивність до вимог ринку. У рамках стартапу пропонується розробка інформаційної системи, яка полегшує управління навчальними даними, взаємодію між студентами і викладачами, а також інтегрується із сучасними освітніми технологіями. У таблиці 5.1 описані напрямки застосування і вигоди для користувачів продукту.

Таблиця 5.1 – Опис ідеї проєкту.

Зміст ідеї	Напрямки застосування	Вигоди для користувача
1	2	3
Створення інформаційної системи забезпечення навчально-виховної діяльності катедри	1. Облік студентів і викладачів	Централізований доступ до актуальних даних про студентів, викладачів і кураторів, що спрощує пошук інформації та забезпечує її точність.
	2. Комунікація між учасниками освітнього процесу	Вирішення проблеми інформування користувачів системи. Користувачам не потрібно шукати контактні дані для зв’язку, достатньо знати ім’я адресата. Окрім того, користувачів можна об’єднувати за групами, ролями, курсами, що дозволяє уникнути дублювання однакових повідомлень.
	3. Аналітика та звітність	Генерація звітів, уникнення помилок, пов'язаних із ручною обробкою даних, і перегляд аналітичної інформації для вдосконалення навчального процесу.
	4. Організація роботи кураторів груп	Куратори отримують інструменти для спрощення рутинних завдань, таких як оновлення списків студентів, аналіз успішності, планування, створення звітів для деканату.

Продукт вирізняється своєю орієнтацією на комплексну автоматизацію саме катедрального рівня управління навчально-виховними процесами. Це не просто загальний освітній або адміністративний інструмент, а рішення, адаптоване для вузької ніші, що враховує специфіку роботи катедр або відділів в освітніх установах. Це відрізняє його від універсальних LMS, ERP чи CRM-систем.

Конкуренти не закривають усі потреби. Жодна з наявних систем не враховує унікальності катедрального рівня: моніторинг успішності, специфіку виховної діяльності, складання планів науково-дослідної роботи тощо. Великі ERP-рішення є дорогими, що робить їх недоцільним для невеликих підрозділів, таких як катедра. LMS надають базові інструменти для дистанційного навчання, але не пропонують

адміністративних рішень, а CRM-системи не адаптовані для освітнього середовища.

Продукт заповнює нішу, яка залишалася поза увагою конкурентів. Він забезпечує інтеграцію функцій управління, комунікації й обліку для катедр, пропонуючи рішення, яке є простим у використанні, доступним і водночас повністю адаптованим до освітньої діяльності.

5.2 Технологічний авдит проєкту

Технологічний авдит – це комплексний аналіз технологічної бази проєкту, який дозволяє оцінити відповідність обраних рішень поставленим завданням, виявити можливі ризики та визначити напрямки для покращення. У цьому розділі проведено оцінку технологій, використаних для створення інформаційної системи, а також їх відповідності вимогам функціональності, продуктивності, безпеки та масштабованості.

Проєкт розроблено мовою програмування Java, яка є одним із найпопулярніших виборів для створення масштабованих і багатофункціональних систем. Java забезпечує високу продуктивність, платформну незалежність та широку екосистему бібліотек і фреймворків, що є ключовим для реалізації складних бізнес-процесів.

Фреймворк Spring забезпечує модульну архітектуру, що дозволяє легко розробляти, тестувати та розширювати функціональність проєкту. Особливу увагу приділено використанню модулів:

- Spring Boot для швидкого налаштування проєкту та інтеграції з іншими компонентами.
- Spring MVC для реалізації архітектури Model-View-Controller, що забезпечує чітке розмежування відповідальностей.
- Spring Data для роботи з базою даних.

- Spring Security для забезпечення автентифікації, авторизації та захисту даних.

У проєкті використано PostgreSQL, яка є надійною реляційною СКБД з відкритим кодом. PostgreSQL відзначається високою продуктивністю, підтримкою складних запитів, можливістю розширення. Ця технологія забезпечує зберігання та обробку даних із дотриманням вимог до цілісності та безпеки.

Для реалізації інтерфейсу користувача було використано Thymeleaf, який інтегрується зі Spring MVC. Цей шаблонізатор дозволяє створювати динамічні вебсторінки, що відповідають сучасним стандартам. HTML і CSS були обрані для структурування та стилізації інтерфейсу.

Обрані інструменти забезпечують підтримку всіх необхідних функцій. Використання фреймворку Spring і PostgreSQL дозволяє обробляти великі обсяги даних і виконувати складні запити без значних затримок, що є великим плюсом для продуктивності. Spring Security надає потужні механізми для захисту доступу до даних. Система легко адаптується для додавання нових функцій або інтеграції з іншими сервісами. Thymeleaf та Java забезпечують простоту розробки і підтримки коду, оскільки є досить поширеними інструментами для веброзробки.

Можливі технологічні ризики.

1. Відсутність мобільної версії. Наразі система орієнтована лише на вебінтерфейс, що може знизити її привабливість для користувачів, які очікують мобільну доступність.

2. Обмежені можливості інтеграції. Система не має модулів для інтеграції з популярними зовнішніми платформами, такими як Zoom, Moodle чи Google Workspace.

3. Недостатність логування дій інших користувачів. Адміністратор системи не має достатньо засобів для перевірки дій або змін, зробленими іншими користувачами для моніторингу стану системи і актуальності даних.

Варто взяти до уваги і рекомендації щодо можливого покращення розробленої системи. Доцільно впровадити мобільний застосунок або адаптивний

дизайн для вебінтерфейсу. Корисною була б розробка API для інтеграції з іншими освітніми платформами. Гарним рішенням є впровадження розширених інструментів аналітики для кращої візуалізації даних і прогнозування. Для адміністратора має сенс реалізація додаткових можливостей для перегляду логів про зміни в системі, зроблені іншими користувачами.

Проведений технологічний авдит підтверджує, що обраний стек технологій є оптимальним для реалізації поставлених завдань проєкту. Система володіє значним потенціалом для подальшого розвитку і масштабування, забезпечуючи високу продуктивність, гнучкість та безпеку. Водночас існують напрямки для вдосконалення, які дозволять збільшити конкурентоспроможність і привабливість системи для ширшого кола користувачів.

5.3 Аналіз ринкових можливостей стартап-проєкту

Аналіз ринкових можливостей є ключовим етапом у розробці стартап-проєкту, оскільки дозволяє визначити потенційних клієнтів, оцінити попит на продукт і виявити конкурентні переваги. Для інформаційної системи, що забезпечує навчально-виховну діяльність катедри, проведений аналіз враховує потреби освітніх установ, технологічні тренди та динаміку розвитку ринку EdTech.

Необхідно визначити основних споживачів продукту і потреби цільової авдиторії. Дана характеристика наведена в таблиці 5.2.

Таблиця 5.2 – Характеристика потенційних споживачів продукту

№	Потреби ринку	Цільова авдиторія	Вимоги клієнтів
1	2	3	4
1	Автоматизація процесів	Університети та інші заклади вищої освіти, приватні освітні центри та курси	Заклади освіти прагнуть мінімізувати рутинні адміністративні задачі, такі як облік студентів, створення звітів і аналіз академічної успішності.

Продовження таблиці 5.2

1	2	3	4
2	Простота використання	Університети та інші заклади вищої освіти, приватні освітні центри та курси	Інтуїтивно зрозумілий інтерфейс для користувачів без спеціальних технічних знань.
3	Доступ до аналітики		Освітні установи все частіше використовують дані для прийняття рішень, тому попит на системи з функціями аналітики постійно зростає

Фактори загроз проєкту – це можливі події чи обставини, які можуть негативно вплинути на успішне завершення та впровадження проєкту. Вони можуть виникати з різних причин, зокрема технічних, фінансових, соціальних або зовнішніх факторів. Фактори загроз для стартап-проєкту вказані у таблиці 5.3.

Таблиця 5.3 – Фактори загроз проєкту

№	Фактор	Опис загрози	Способи мінімізації
1	2	3	4
1	Нестабільність серверної інфраструктури	Проблеми з хостингом або перебої в роботі серверів можуть призвести до втрати даних і вплинути на доступність системи.	Використання хмарних рішень, впровадження резервних серверів.
2	Адаптація до унікальних потреб	Оскільки система створювалася для потреб навчально-виховного напряму конкретної катедри, її функціонал може виявитися недостанім або вимагати адаптацій для інших користувачів.	Інформування про можливості продукту, підтримка під час адаптації.
3	Зниження популярності та підтримки використаних технологій	Швидкий розвиток технологій може призвести до того, що використовувані інструменти стануть застарілими і вимагатимуть значних інвестицій в оновлення.	Постійний моніторинг технологічних трендів, своєчасне оновлення.

Продовження таблиці 5.3

1	2	3	4
4	Неавторизований доступ до даних	Несанкціонований доступ до конфіденційної інформації користувачів, викрадення особистих даних.	Використання двофакторної автентифікації, регулярне оновлення паролів, шифрування даних, захист з’єднань через HTTPS та інші сучасні механізми безпеки.
5	Неправильне управління даними	Помилки користувачів або адміністраторів при обробці, внесенні або зміні даних можуть призвести до їхнього пошкодження або втрати.	Використання чітких політик управління даними, навчання персоналу, введення системи перевірок і підтверджень при виконанні критичних операцій.

Визначення та аналіз цих загроз є важливою частиною стратегії управління ризиками і дозволяє підготуватися до потенційних проблем, що можуть виникнути під час реалізації проєкту.

SWOT-аналіз ринкових можливостей – це інструмент стратегічного управління, який дозволяє визначити сильні і слабкі сторони, а також можливості та загрози, що пов'язані з ринком і зовнішнім середовищем. Цей аналіз допомагає зрозуміти, які фактори можуть позитивно чи негативно вплинути на розвиток продукту.

У цьому контексті ми аналізуємо інформаційну систему, розроблену для автоматизації бізнес-процесів катедри та забезпечення її навчально-виховної діяльності, з акцентом на її функціональні можливості, використання та перспективи на ринку.

Матриця SWOT-аналізу подана у таблиці 5.4.

Таблиця 5.4 – Матриця SWOT-аналізу

Сильні сторони (S)	Слабкі сторони (W)
• Автоматизація бізнес-процесів • Інтуїтивно зрозумілий інтерфейс • Гнучкість • Збереження та організація даних	• Обмеженість функціоналу • Необхідність у оновленнях • Залежність від інтернет-з'єднання
Можливості (O)	Загрози (T)
• Розширення функціоналу • Адаптація системи для інших університетів • Використання штучного інтелекту та аналітики	• Конкуренція на ринку освітніх технологій • Технічні проблеми та безпека • Проблеми з масштабуванням • Складність адаптації для інших клієнтів

SWOT-аналіз показав, що система має сильні конкурентні переваги, такі як простота використання, адаптація до специфіки навчальних закладів, можливість автоматизації багатьох процесів і покращена організація роботи з даними. Розуміння сильних сторін застосовується для залучення користувачів, маркетингу і позиціонування на ринку.

Аналіз дозволив визначити аспекти, які потребують вдосконалення, наприклад, обмеженість функціоналу або потреба в додаткових заходах із забезпечення безпеки даних. Завдяки цьому ми можемо планувати ресурси і розробляти покращення для усунення цих недоліків у майбутніх ітераціях системи.

SWOT-аналіз допоміг виявити перспективи для розширення використання системи, що дає можливість створювати плани розвитку, орієнтовані на ринковий попит. Аналіз загроз (зовнішніх ризиків) дав змогу врахувати потенційні проблеми, такі як конкуренція з іншими платформами, загрози кібербезпеки або ризики низької адаптації серед цільової авдиторії. Це дозволяє заздалегідь розробити стратегії для мінімізації цих загроз.

Отже, SWOT-аналіз не лише показав поточний стан проєкту, але й створив основу для прийняття стратегічних рішень, які допоможуть успішно вивести систему на ринок і забезпечити її подальший розвиток.

5.4 Розроблення ринкової стратегії програмного продукту

Ринкова стратегія програмного продукту є ключовим інструментом, який визначає, як продукт буде впроваджений на ринок, як він відповідатиме потребам цільової авдиторії та які переваги забезпечуватиме в умовах конкуренції. Для розробки такої стратегії враховуються результати SWOT-аналізу, дослідження ринку, аналіз конкурентів і специфікації самого продукту.

У таблиці 5.5 надано оцінку цільових груп, які можуть зацікавитися впровадженням інформаційної системи.

Таблиця 5.5 – Аналіз цільових груп споживачів

№	Опис профілю цільової групи	Готовність споживачів до сприйняття	Орієнтовний попит в межах цільової групи	Інтенсивність конкуренції в сегменті	Простота входу в сегмент
1	2	3	4	5	6
2	Викладачі та куратори	Середня	Середній	Низька	Висока
3	Керівники напрямів катедр (відділів)	Висока	Високий	Середня	Середня
4	Студенти	Середня	Середній	Низька	Висока

Виділені три цільові групи, проте вони усі є частиною однієї організації, наприклад одного ВНЗ. Тому для інформаційної системи забезпечення навчально-виховної діяльності катедри найдоцільніше обрати стратегію концентрованого маркетингу, яка спрямована на вузьку цільову авдиторію та максимальне задоволення її потреб. Завдяки фокусу на одному сегменті продукт можна позиціонувати як вузькоспеціалізований і краще адаптований до потреб ринку порівняно з універсальними конкурентами.

Визначення стратегії розвитку є фундаментом для довгострокового планування проєкту. Цей процес дозволяє встановити, яким чином продукт буде

масштабуватися, адаптуватися до ринкових змін і підвищувати свою конкурентоспроможність. Аналіз стратегії розвитку наведений у таблиці 5.6.

Таблиця 5.6 – Визначення стратегії розвитку

Обрана альтернатива розвитку проєкту	Стратегія охоплення ринку	Ключові конкурентноспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	2	3	4
Розробка додаткового функціоналу для корпоративних клієнтів	Розширене охоплення, залучення освітніх та бізнес-організацій	Гнучкість системи, масштабованість, висока продуктивність	Диверсифікація ринкових сегментів

Визначення стратегії конкурентної поведінки дозволяє визначити, як саме продукт буде позиціонуватися порівняно з конкурентами, які його унікальні переваги мають бути підкреслені, та як відповідати на виклики, що виникають у ринковому середовищі. Аналіз поданий у таблиці 5.7.

Таблиця 5.7 – Визначення стратегії конкурентної поведінки

Чи унікальний продукт?	Чи буде компанія шукати нових споживачів?	Чи передбачається копіювання можливостей конкурентів?	Стратегія поведінки
1	2	3	4
Так	Так	Так	Стратегія диференціації

На основі аналізу, представленого в таблиці, найдоцільнішою стратегією поведінки є стратегія диференціації. Ця стратегія передбачає фокусування на унікальних особливостях продукту, які вирізняють його серед конкурентів, а також націлення на конкретні сегменти ринку, зокрема вищі навчальні заклади.

Успіх впровадження інформаційної системи залежить не лише від її технічної реалізації, а й від правильного позиціонування на ринку. Вибір позиціонування визначає, як продукт сприйматиметься потенційними користувачами, у чому вони вбачатимуть його цінність і які переваги оберуть для себе найважливішими. Для розроблюваної системи автоматизації навчально-виховних процесів найкращим рішенням є використання стратегії позиціонування за споживачем. Це дозволяє максимально адаптувати продукт під потреби кожної групи користувачів, забезпечуючи відповідність очікуванням студентів, викладачів та адміністрації університетів.

Позиціонування за споживачем ґрунтується на аналізі потреб та пріоритетів цільової авдиторії. Інформаційна система підтримує кілька типів користувачів, кожен з яких має специфічні функції. Позиціонування системи таким чином, щоб кожна група користувачів побачила вирішення своїх проблем, дозволяє підвищити рівень довіри до продукту.

Конкурентною перевагою такого підходу є орієнтація на реальні потреби користувачів, а не лише технічні можливості продукту. У порівнянні з конкурентами, які можуть мати більш універсальні, але менш персоналізовані рішення, ця система вигідно виділяється саме за рахунок врахування специфіки навчально-виховних процесів. Вона також підкреслює свою адаптивність і гнучкість, що особливо важливо у сфері освіти, яка є динамічною та часто змінюється під впливом зовнішніх умов.

Позиціонування за споживачем також забезпечує стійкість продукту на ринку. Орієнтація на вирішення конкретних задач дозволяє швидко адаптувати систему до змін, вимог акредитації або внутрішніх процедур університетів. Важливим аспектом є також те, що такий підхід полегшує створення позитивного користувацького досвіду, що є ключовим фактором для довготривалого використання продукту та збереження конкурентоспроможності [21].

5.5 Розроблення маркетингової програми стартап-проєкту

Маркетингова програма стартап-проєкту є етапом, який дозволяє визначити шляхи просування продукту на ринку, встановити ефективні канали комунікації з потенційними клієнтами та розробити стратегії взаємодії з цільовою авдиторією.

Для початку необхідним є розуміння потреб ринку і переваг продукту. Це визначає таблиця 5.8.

Таблиця 5.8 – Ключові переваги продукту

№	Потреба	Вигода	Переваги
1	2	3	4
1	Автоматизація бізнес-процесів	Зниження навантаження на викладачів, кураторів і адміністрацію	Спеціалізація на конкретні потреби
2	Моніторинг успішності	Зручність для викладачів відстеженні результатів	Аналіз успішності, можливість зберігання історії успіхів студентів
3	Комунікація між учасниками освітнього процесу	Полегшення взаємодії між студентами, викладачами та адміністрацією	Швидкий обмін інформацією, нагадування про важливі події, централізований доступ до ресурсів
4	Створення та зберігання звітів	Спрощення звітності та збереження документів	Автоматизоване створення звітів, електронне зберігання документів
5	Безпека даних студентів та викладачів	Захист персональних даних та безпека інформації	Використання сучасних методів шифрування і контролю доступу

Трирівнева маркетингова модель описує різні рівні, на яких компанія може взаємодіяти зі своїми клієнтами та реалізовувати свою стратегію продажів [22]. Три рівні цієї моделі подані у таблиці 5.9.

Таблиця 5.9 – Трирівнева маркетингова модель

Рівень	Складові	
1	2	
Товар за задумом	Інформаційна система забезпечення навчально-виховної діяльності катедри у вигляді вебзастосунку	
Товар у реальному виконанні	Ознаки	
	Якість	Відповідність стандартам ДСТУ ISO 23026:2016 у розробленні та керуванні WEB-сайтами для систем
	Властивості	1. Інформаційний модуль 2. Модуль управління науково-виховною роботою 3. Модуль управління навчальним процесом 4. Модуль співпраці та комунікації
	Дизайн	Вебзастосунок
	Назва марки	ІС «Катедра»
	Пакування	Відсутнє. ПЗ надається у вигляді архівів
Товар із підкріпленням	Навчання користувачів щодо використання системи , початкове налагодження, адаптація до спеціальних вимог, підтримка системи після впровадження	

Таблиця 5.10 визначає канали просування та комунікації. Ця інформація допоможе обрати рекламні інструменти та визначити бюджет стратегії.

Таблиця 5.10 – Канали просування

№	Канал	Цільова авдиторія	Інструменти	Очікуваний ефект
1	2	3	4	5
1	Освітні конференції	Викладачі, куратори	Презентації, демонстрація продукту	Залучення партнерів, зміцнення репутації
2	Email-маркетинг	Адміністратори	Персоналізовані листи	Підвищення інтересу, укладення угод
3	Партнерство з ВНЗ	Усі учасники освітнього процесу	Прямі домовленості, інтеграція	Розширення бази клієнтів

Висновки до розділу 5

У розділі здійснено всебічний аналіз та обґрунтування ключових аспектів створення інформаційної системи, яка має потенціал стати інноваційним продуктом у сфері автоматизації навчально-виховних процесів. Описано ідею проєкту, її актуальність та технологічну основу, проведено авдит технологічних рішень, а також виконано аналіз ринкових можливостей, що забезпечило чітке розуміння потенціалу проєкту.

У процесі дослідження визначено ринкову стратегію розвитку проєкту, яка базується на потребах цільових груп користувачів, зокрема студентів, викладачів, кураторів та адміністративного персоналу. Значну увагу приділено розробці стратегії позиціонування продукту, орієнтованої на диференціацію за функціональністю та користувацькою цінністю. Аналіз конкурентних переваг та загроз дозволив ідентифікувати сильні та слабкі сторони системи, а також визначити зовнішні можливості для зростання і розвитку.

У розділі також було розроблено маркетингову програму стартап-проєкту, що включає використання сучасних інструментів просування та охоплення цільових авдиторій. Впровадження трирівневої маркетингової моделі дозволило сформувати комплексний підхід до просування продукту, враховуючи всі рівні його реалізації: від основної концепції до додаткових послуг і переваг.

Отже, розроблений проєкт поєднує сучасні технологічні рішення з чіткою ринковою стратегією, що створює умови для його успішного впровадження та подальшого розвитку. Проведений аналіз доводить, що стартап має значний потенціал для досягнення конкурентних переваг у сегменті освітніх IT-рішень.

ВИСНОВКИ

Робота присвячена розробці інформаційної системи забезпечення навчально-виховної діяльності катедри, яка є актуальним завданням у контексті цифровізації освітнього процесу. У межах дослідження було розглянуто всі аспекти створення системи: від постановки завдання до розробки стартап-проєкту.

Сформульовано завдання розробки інформаційної системи, яка забезпечує автоматизацію основних бізнес-процесів катедри, таких як облік студентів і викладачів, підтримка взаємодії між користувачами, створення звітності, управління документами тощо. На основі визначення основних вимог до функціоналу системи і потреб користувачів було обґрунтовано доцільність її розробки.

Проведено порівняння наявних типів систем, визначено їх переваги та недоліки, а також запропоновано вдосконалення, які враховують специфіку роботи катедри. Зокрема, було відзначено, що більшість готових рішень не враховують гнучкість налаштування для окремих освітніх підрозділів, що виправлено у запропонованій системі.

Описано архітектуру інформаційної системи, модель бази даних, визначено основні компоненти програмної реалізації. Використання сучасних технологій, таких як фреймворк Spring, PostgreSQL та інші, забезпечило оптимальну продуктивність і надійність системи. Значна увага була приділена питанням забезпечення безпеки даних, враховуючи специфіку роботи із конфіденційною інформацією.

Розглянуто функціональні можливості системи для різних ролей користувачів, зокрема студентів, викладачів, кураторів та адміністратора. Описано користувацький інтерфейс вебзастосунку.

Розроблено концепцію стартап-проєкту для реалізації системи на ринку. Проведено аналіз ринкових можливостей, визначено цільову авдиторію та конкурентні переваги продукту. В межах роботи було створено ринкову стратегію,

позиціонування та маркетингову програму, що дозволяє не лише впроваджувати систему у внутрішньому середовищі навчальних закладів, але й розширювати її використання серед інших установ, які потребують автоматизації освітніх процесів.

У результаті виконаної роботи досягнуто поставлених цілей, зокрема створено інформаційну систему у вигляді вебзастсоунку, здатну підвищити ефективність навчально-виховної діяльності катедри. Проведені дослідження підтвердили її практичну доцільність, конкурентоспроможність та можливість масштабування. Розроблена система не лише полегшить роботу викладачів, студентів та адміністрації, а й створить умови для подальшого розвитку відповідно до потреб катедри й університету загалом.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Що таке бізнес-процеси в підприємницькій діяльності? Школа бізнесу Нова Пошта. URL: https://online.novaposhta.education/blog/scho-take-biznes-protsesi-v-pidpriyemnitskij-diyalnosti (дата звернення: 30.11.2024).
2. Laudon K. C., Laudon J. P. Management Information Systems: Managing the Digital Firm. 17th ed. Hoboken : Pearson Education, 2016. 648 p.
3. Information Systems, Southern Africa: Mancosa. 2019. URL: https://www.academia.edu/42801669/INFORMATION_SYSTEMS_Module_Guide (дата звернення: 30.11.2024).
4. Exploring Information Systems (IS): Definition and Components. DataScientest. URL: https://datascientest.com/en/exploring-information-systems-is-definition-and-components (дата звернення: 30.11.2024).
5. Zwass V. Information system infrastructure and architecture. Encyclopedia Britannica. URL: https://www.britannica.com/topic/information-system/Information-system-infrastructure-and-architecture (дата звернення: 30.11.2024).
6. Що таке CRM? Creatio. URL: https://www.creatio.com/ua/crm/what-is-crm (дата звернення: 30.11.2024).
7. SoftInform, Спілка Автоматизаторів Бізнесу. ERP для університетів. Особливості функціональності та впровадження. Публікації. URL: https://www.softinform.com.ua/news/erp-dlia-universytetiv-osoblyvosti-funktsionalnosti-ta-vprovadzhennia/ (дата звернення: 23.10.2024).
8. Цапліна А. Що таке LMS та як підібрати собі LMS-систему. SendPulse. URL: https://sendpulse.ua/blog/learning-management-system (дата звернення: 30.11.2024).
9. Що таке HRMS. HURMA. URL: https://hurma.work/blog/hrms-systema-dlya-biznesu/ (дата звернення: 30.11.2024).
10. Comparing Database Management Systems. AltexSoft. URL: https://www.altexsoft.com/blog/comparing-database-management-systems-mysql-

postgresql-mssql-server-mongodb-elasticsearch-and-others/ (дата звернення: 30.11.2024).

11. Pro Spring 5: An In-Depth Guide to the Spring Framework and Its Tools. 5th Edition / I.Cosmina, R. Harrop, C. Schaefer, C. Ho. Apress, 2017. 878 p.

12. Deck P. Spring MVC : A Tutorial, Second Edition. Brainy Software Inc., 2016. 338 p.

13. Spring Boot. Документація Spring. URL: https://spring.io/projects/spring-boot (дата звернення: 23.10.2024).

14. Spring Data JPA – Reference Documentation. URL: https://docs.spring.io/spring-data/jpa/reference/jpa/query-methods.html (дата звернення: 30.11.2024).

15. Spilca L. Spring Security in Action. Manning Publications : Shelter Island, 2020. 560 p.

16. Що таке CSS? W3SchoolsUA. URL: https://w3schoolsua.github.io/css/index.html (дата звернення: 30.11.2024).

17. Sahu N.S. Understanding Thymeleaf: A Powerful Java Template Engine. DEV Community. URL: https://dev.to/nikhilxd/understanding-thymeleaf-a-powerful-java-template-engine-3bnh (дата звернення: 30.11.2024).

18. Garcia-Molina H., Ullman J. D., Widom J. Database Systems : The Complete Book 2nd Edition. Upper Saddle River, New Jersey : Pearson, 2008. 1248 p.

19. M. Deinum, D. Rubio, J. Long. Spring 5 Recipes: A Problem-Solution Approach. 4th Edition. Apress, 2017. 870 p.

20. Каграманова Ю. Як будувати UML-діаграми. Розбираємо три найпопулярніші варіанти. DOU. URL: https://dou.ua/forums/topic/40575/ (дата звернення: 30.11.2024).

21. User Positioning. Fiveable Inc. URL: https://library.fiveable.me/key-terms/corporate-communication/user-positioning (дата звернення: 30.11.2024).

22. Григорчук Т. Маркетинг. Друга частина : Навч. посіб. Київ : Ун-т "Україна", 2007. 380 с.

ДОДАТОК А Фрагменти лістингу програми

Моделі користувачів системи

User.java

```
@Entity
@Table(name = "users")
@AllArgsConstructor
@NoArgsConstructor
@Data
@DiscriminatorColumn(
      name = "user_type",
      discriminatorType = DiscriminatorType.STRING)
public abstract class User implements UserDetails {

   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;

   @NotBlank(message = "Введіть, будь ласка, повне ім'я!")
   private String fullName;

   @Column(unique = true)
   private String email;

   private String phoneNumber;

   private String telegramTag;

   private LocalDate dateOfBirth;

   private String password;

   @ManyToMany(fetch = FetchType.EAGER)
   @JoinTable(
         name = "user_role",
         joinColumns = @JoinColumn(name = "user_id"),
         inverseJoinColumns = @JoinColumn(name = "role_id"))
   private Set<Role> roles;

   @Transient
   private final boolean isAccountNonExpired = true;
```

```
    @Transient
    private final boolean isCredentialsNonExpired = true;

    @Transient
    private final boolean isEnabled = true;

    @Transient
    private final boolean isAccountNonLocked = true;

    @Override
    public Collection<? extends GrantedAuthority> getAuthorities() {
        return this.roles;
    }

    @Override
    public String getPassword() {
        return password;
    }

    @Override
    public String getUsername() {
        return getEmail();
    }

    public abstract Group getGroup();
}
```

Student.java

```
@Entity
@AllArgsConstructor
@NoArgsConstructor
@Data
@DiscriminatorValue(value = "student")
public class Student extends User {

    private String reservePhone;

    @ManyToOne(cascade = CascadeType.PERSIST)
    @JoinColumn(name = "group_id")
    private Group group;

    @Enumerated
```

```
    private PaymentType paymentType;

    private String address;

    private boolean isMonitor;
}
```

Lecturer.java

```
@Entity
@AllArgsConstructor
@NoArgsConstructor
@Data
@DiscriminatorValue(value = "lecturer")
public class Lecturer extends User{

    @ManyToOne
    @JoinColumn(name = "department_id")
    private Department department;

    private String position;

    private String scientificTitle;

    private String academicDegree;

    @ToString.Exclude
    @EqualsAndHashCode.Exclude
    @OneToOne(mappedBy = "curator")
    private Group group;

    private String area;
}
```

Приклад репозиторію

StudentRepository.java

```
@Repository
public interface StudentRepository extends JpaRepository<Student, Long> {
  Set<Student> findByFullNameContaining(String name);
  Set<Student> findByEmailStartingWith(String email);
  List<Student> findAllByOrderByFullName();
  Student findStudentByGroupIdAndIsMonitorIsTrue(Long groupId);
```

```
  List<Student> findAllByGroupIdOrderByFullName(Long groupId);
  List<Student> findAllByGroupNameOrderByFullName(String groupName);
  List<Student> findAllByIsMonitorIsTrue();
}
```

Приклад контролера

CuratorController.java

```
@Slf4j
@Controller
@Secured({"ROLE_ADMINISTRATOR", "ROLE_CURATOR"})
public class CuratorController {
   private final GroupService groupService;
   private final StudentService studentService;
   private final FileProcessor fileProcessor;

   public CuratorController(GroupService groupService, StudentService studentService,
         FileProcessor fileProcessor) {
      this.groupService = groupService;
      this.studentService = studentService;
      this.fileProcessor = fileProcessor;
   }

   @GetMapping("/new/student")
   public String getStudentCreationPage(final Model model) {
      model.addAttribute("student", new StudentDTO());
      model.addAttribute("groups", groupService.getAllGroups());
      return "new_student";
   }

   @PostMapping("/save/student")
   public String saveStudent(@Valid @ModelAttribute("student") StudentDTO student,
                     final Errors errors) {
      if (errors.hasErrors()) {
         log.error(errors.toString());
         return "new_student";
      }
      studentService.saveStudent(student);
      return "redirect:/new/student";
   }

   @GetMapping("/attestation")
```

```
    public String getAttestationPage() {
        return "attestation";
    }

    @PostMapping("/attestation")
    public String readExcelFile(@RequestParam("file") final MultipartFile
attestationFile,
                                @RequestParam(required = false, defaultValue = "1") String
attestationNo,
                                @RequestParam("group_name") String groupName,
                                final RedirectAttributes redirectAttributes) {
        final Map<String, Set<AttestationDTO>> parsingResults =
                fileProcessor.parseAttestationFile(attestationFile);
        if (isNull(parsingResults)) {
            redirectAttributes.addFlashAttribute("error",
attestationFile.getOriginalFilename());
        }
        redirectAttributes.addFlashAttribute("parsingResults", parsingResults);
        redirectAttributes.addFlashAttribute("attestationNo", attestationNo);
        redirectAttributes.addFlashAttribute("excelFileName",
                "Результати " + attestationNo + " атестації " + groupName);
        return "redirect:/attestation";
    }

    @GetMapping("/add/students")
    public String getStudentsFromFileAddingPage() {
        return "new_students";
    }

    @PostMapping("/add/students")
    public String getStudentsFromFile(@RequestParam("file") final MultipartFile file,
                                      @RequestParam String groupName) {
        studentService.saveStudentsFromFile(file, groupName);
        return "redirect:/group/" + URLEncoder.encode(groupName,
StandardCharsets.UTF_8);
    }

    @PostMapping("/delete/student/{id}")
    public String deleteStudent(@PathVariable Long id) {
        Student student = studentService.getStudentById(id);
        studentService.deleteStudent(student);
        return "redirect:/group/" +
                URLEncoder.encode(student.getGroup().getName(),
```

```
StandardCharsets.UTF_8);
    }

    @GetMapping("/edit/student/{id}")
    public String editStudent(@PathVariable Long id, Model model) {
        Student student = studentService.getStudentById(id);
        model.addAttribute("student", studentService.getDTOForModel(student));
        model.addAttribute("groups", groupService.getAllGroups());
        return "new_student";
    }

    @PostMapping("/group/setMonitorForGroup")
    @ResponseBody
    public String setMonitorForGroup(@RequestParam final String group,
                                     @RequestParam final Long student) {
        return studentService.setMonitorForGroup(student, group);
    }
}
```

Приклад сервіса

DefaultStudentService.java

```
@Slf4j
@Service
public class DefaultStudentService implements StudentService {
  @Resource
  private StudentRepository studentRepository;

  @Resource
  private GroupService groupService;

  @Override
  public Set<Student> findByKey(final String key) {
    final Set<Student> students = studentRepository.findByFullNameContaining(key);
    students.addAll(studentRepository.findByEmailStartingWith(key));
    return students;
  }

  @Override
  public List<Student> getAllStudents() {
    return studentRepository.findAllByOrderByFullName();
  }
```

```
@Override
public Student getStudentById(String id)
{
  try {
    return studentRepository.findById(Long.parseLong(id)).orElse(null);
  } catch (NumberFormatException e) {
    return null;
  }
}

@Override
public Student getStudentById(Long id) {
  return studentRepository.findById(id).orElse(null);
}

@Override
public String setMonitorForGroup(final Long studentId, final String group) {
  final Student student = studentRepository.findById(studentId).orElse(null);
  final Group studentsGroup = groupService.getGroupByName(group);
  if (nonNull(student) && student.getGroup().equals(studentsGroup)) {
    student.setMonitor(true);
    studentRepository.save(student);
    return student.getFullName();
  }
  return null;
}

@Override
public Student getMonitorInGroup(final Long groupId) {
  return studentRepository.findStudentByGroupIdAndIsMonitorIsTrue(groupId);
}

@Override
public List<Student> getAllStudentsByGroupId(Long groupId) {
  return studentRepository.findAllByGroupIdOrderByFullName(groupId);
}

@Override
public List<Student> getAllStudentsByGroupName(String groupName) {
  return studentRepository.findAllByGroupNameOrderByFullName(groupName);
}
```

```
  @Override
  public List<Student> getAllMonitors() {
    return studentRepository.findAllByIsMonitorIsTrue();
  }

  @Override
  public void deleteStudent(Student student) {
    studentRepository.delete(student);
  }
}
```

Сервіс для обробки та створення файлів

FileProcessor.java

```
@Slf4j
@Service
public class FileProcessor {
  private static final String NA = "na";

  private static final int GRADES_AVAILABLE = 6;

  @Resource
  private GroupService groupService;

  @Resource
  private StudentService studentService;

  public Map<String, Set<AttestationDTO>> parseAttestationFile(MultipartFile
attestationFile) {
    try (final Workbook workbook =
WorkbookFactory.create(attestationFile.getInputStream())) {
      final Sheet worksheet = workbook.getSheetAt(0);

      final Row teachers = worksheet.getRow(0);
      final Row subjects = worksheet.getRow(1);

      final Map<String, Set<AttestationDTO>> results = new LinkedHashMap<>();
      for (int i = 2; i < worksheet.getPhysicalNumberOfRows(); i++) {
        final Row row = worksheet.getRow(i);
        final String student = row.getCell(1).getStringCellValue();
        final Set<AttestationDTO> subjectAttestation = new LinkedHashSet<>();
        for (int j = 2; j < row.getPhysicalNumberOfCells(); j++) {
```

```
            if (NA.equalsIgnoreCase(row.getCell(j).getStringCellValue()))
            {
              final AttestationDTO attestation = new AttestationDTO();
              attestation.setLecturer(teachers.getCell(j).getStringCellValue());
              attestation.setSubject(subjects.getCell(j).getStringCellValue());
              subjectAttestation.add(attestation);
            }
          }
          if (!subjectAttestation.isEmpty()) {
            results.put(student, subjectAttestation);
          }
        }

        return results;
      } catch (IOException e) {
        log.error(e.getMessage());
        return null;
      }
    }

public void createGroupsInfoFile() throws IOException {
    XWPFDocument document = new XWPFDocument();
    FileOutputStream out = new FileOutputStream(new File("groupsInfo.docx"));

    XWPFParagraph paragraph = document.createParagraph();
    paragraph.setAlignment(ParagraphAlignment.CENTER);
    XWPFRun title = paragraph.createRun();
    title.setFontSize(16);
    title.setBold(true);
    title.setFontFamily("Times New Roman");
    title.setText("Старости та куратори груп катедри ІПЗЕ");
    title.addBreak();
    title.setText("Денна форма");
    title.addBreak();

    XWPFTable table = document.createTable(1, 5);
    table.setWidthType(TableWidthType.PCT);
    setTableColumnsWidth(table, 1400, 2000, 3000, 2500, 3000);

    XWPFTableRow headerRow = table.getRow(0);
    setCellText(headerRow.getCell(0), "Група", 11, true, false, true);
    setCellText(headerRow.getCell(1), "ПІБ старости", 11, true, false, true);
    setCellText(headerRow.getCell(2), "Контакти старости", 11, true, false, true);
```

```
    setCellText(headerRow.getCell(3), "Пошта групи", 11, true, false, true);
    setCellText(headerRow.getCell(4), "ПІБ і контакти куратора", 11, true, false,
true);

    for (int grade = 1; grade <= GRADES_AVAILABLE; grade++) {
      addMergedRow(table, 5, grade);
      List<Group> firstCourseGroups = groupService.getAllGroupsByGrade(grade);
      for (Group group : firstCourseGroups) {
        Student monitor = studentService.getMonitorInGroup(group.getId());
        if (Objects.nonNull(monitor)) {
          Lecturer curator = group.getCurator();

          XWPFTableRow row = table.createRow();
          setCellText(row.getCell(0), group.getName(), 13, false, false, false);
          setCellText(row.getCell(1), monitor.getFullName(), 11, false, false, false);
          setCellText(row.getCell(2), monitor.getPhoneNumber() + "\n" +
monitor.getEmail(), 11,
                  false, false, false);
          setCellText(row.getCell(3), group.getGroupEmail(), 11, false, false, false);
          setCellText(row.getCell(4),
                  curator.getFullName() + ",\nт." + curator.getPhoneNumber() + ",\n" +
                          curator.getEmail(), 11, false, false, false);
        }
      }
    }
    document.write(out);
    out.close();
  }

  private void addMergedRow(XWPFTable table, int cols, int grade) {
    XWPFTableRow row = table.createRow();
    CTHMerge hMerge = CTHMerge.Factory.newInstance();
    hMerge.setVal(STMerge.RESTART);
    row.getCell(0).getCTTc().addNewTcPr();
    row.getCell(0).getCTTc().getTcPr().setHMerge(hMerge);
    setCellText(row.getCell(0), grade + " курс", 13, true, false, true);

    CTHMerge hMerge1 = CTHMerge.Factory.newInstance();
    hMerge1.setVal(STMerge.CONTINUE);
    for (int i = 1; i < cols; i++) {
      row.getCell(i).getCTTc().addNewTcPr();
      row.getCell(i).getCTTc().getTcPr().setHMerge(hMerge1);
    }
```

```
  }

  private static void setTableColumnsWidth(XWPFTable table, long... widths) {
    CTTblGrid grid = table.getCTTbl().addNewTblGrid();
    for (long w : widths) {
      grid.addNewGridCol().setW(BigInteger.valueOf(w));
    }
  }

  private static void setCellText(XWPFTableCell cell, String text, int fontSize, boolean
bold,
        boolean addBreak, boolean center) {
    cell.removeParagraph(0);
    XWPFParagraph paragraph = cell.addParagraph();
    if (center) {
      paragraph.setAlignment(ParagraphAlignment.CENTER);
    }
    XWPFRun run = paragraph.createRun();
    run.setFontFamily("Times New Roman");
    run.setFontSize(fontSize);
    run.setText(text);
    run.setBold(bold);
    if (addBreak) {
      run.addBreak();
    }
  }
}
```

ДОДАТОК Б Презентація

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Навчально-науковий інститут атомної та теплової енергетики
Кафедра інженерії програмного забезпечення в енергетиці

ТЕМА: «ІНФОРМАЦІЙНА СИСТЕМА ЗАБЕЗПЕЧЕННЯ НАВЧАЛЬНО-ВИХОВНОЇ ДІЯЛЬНОСТІ КАТЕДРИ»

Виконав студент групи ТВ-32мп
Дячук Андрій Олегович
Керівник доцент, к.ф.-м.н.
Свинчук Ольга Василівна

Київ - 2024

АКТУАЛЬНІСТЬ ТЕМАТИКИ

- Потреба в ефективному управлінні даними в навчальному процесі.
- Автоматизація адміністративних процесів.
- Організація зручної комунікації між викладачами і студентами.
- Відсутність готових програмних рішень, які здатні повністю задовольнити потреби катедри.
- Забезпечення централізованого доступу до актуальної інформації.
- Підвищення ефективності трудових ресурсів.
- Зменшення кількості помилок при роботі з даними.

МЕТА І ЗАВДАННЯ РОБОТИ

- **Мета роботи:** розробка інформаційної системи забезпечення навчально-виховної діяльності катедри.
- **Об'єкт дослідження:** навчально-виховна діяльність катедри.
- **Предмет дослідження:** інформаційна система забезпечення навчально-виховної діяльності катедри.

Завдання:

- проаналізувати аналогічні інформаційні системи;
- визначити основні потреби користувачів і вимоги до системи;
- розробити вебзастосунок, який відповідає потребам навчально-виховного напряму катедри і складає функціонал системи;
- розробити зручний персоналізований інтерфейс.

АНАЛІЗ НАЯВНИХ СИСТЕМ

Тип системи	Фокус	Переваги	Недоліки
CRM	Управління взаємовідносинами з клієнтами	Широкий спектр інструментів для обліку, простота комунікації	Непридатна для навчального процесу
LMS	Управління навчанням	Онлайн-курси, тестування, моніторинг успішності	Немає функцій адміністрування катедри
ERP	Управління ресурсами	Інтеграція адміністративних процесів	Висока вартість і складність адаптації
HRMS	Управління персоналом	Планування навантаження викладачів, автоматизований документообіг	Орієнтація лише на кадрові процеси

ЗАСОБИ РОЗРОБКИ

spring

spring boot

spring MVC

spring SECURITY

spring DATA

Java

THE APACHE SOFTWARE FOUNDATION

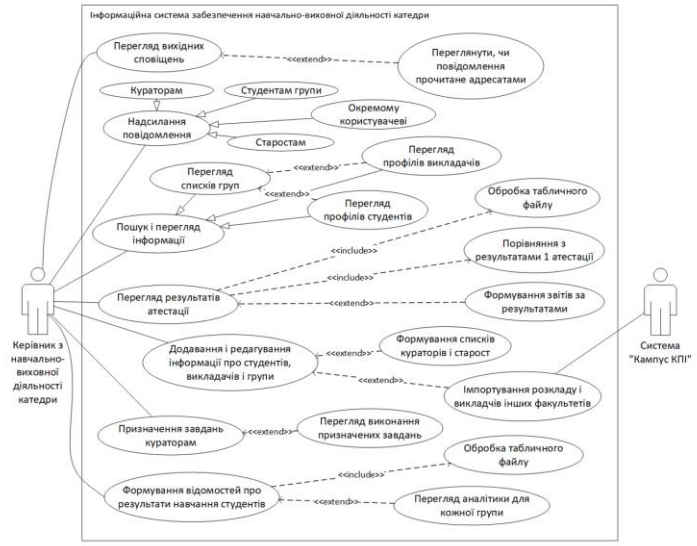
Thymeleaf

HTML CSS JS

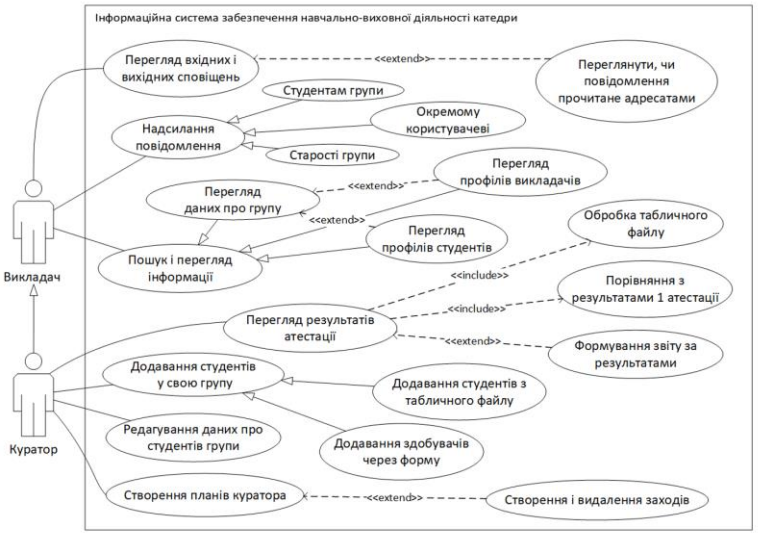
PostgreSQL

IntelliJ IDEA

ДІАГРАМА ПРЕЦЕДЕНТІВ КЕРІВНИКА НАПРЯМУ

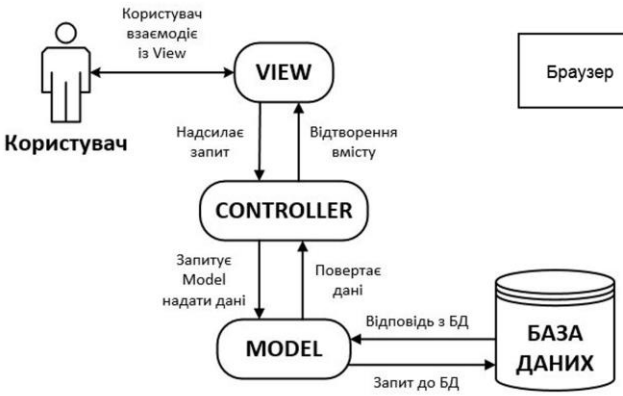


ДІАГРАМА ПРЕЦЕДЕНТІВ ВИКЛАДАЧА

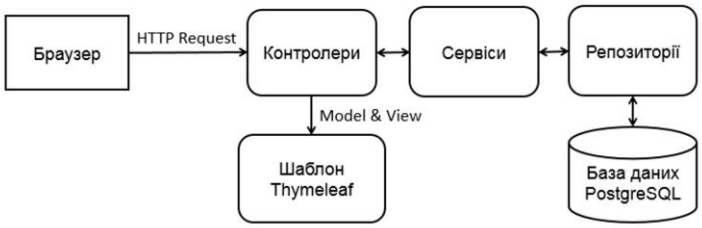


ДІАГРАМА ПРЕЦЕДЕНТІВ СТУДЕНТА

АРХІТЕКТУРА СИСТЕМИ



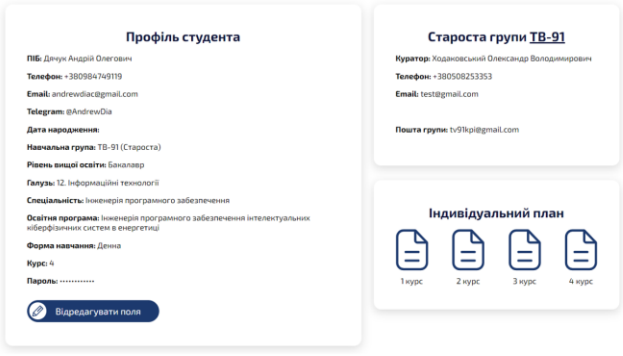
Архітектура системи



Структура проєкту

СТОРІНКА ГРУПИ І ПРОФІЛЬ СТУДЕНТА

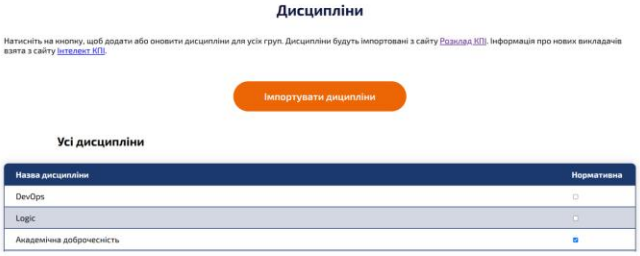
Сторінка групи зі списком студентів



Приклад профілю студента

СТОРІНКА ПОШУКУ ТА СТОРІНКА ІМПОРТУ РОЗКЛАДУ

Сторінка пошуку

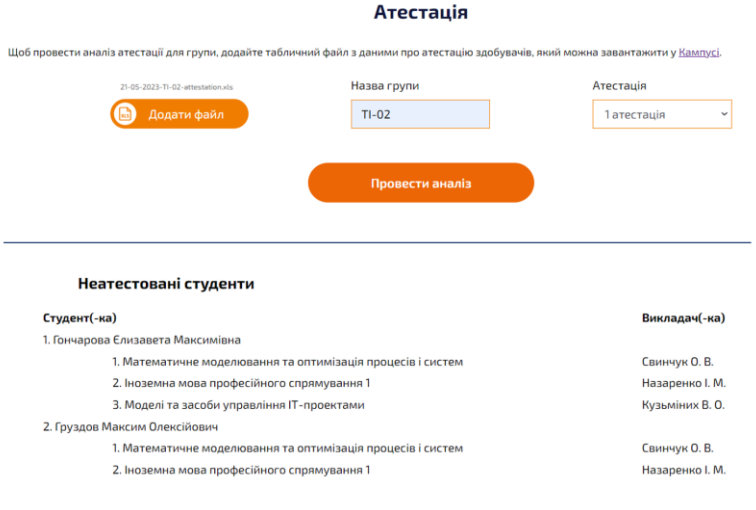


Сторінка імпорту розкладу та дисциплін

СТОРІНКА АТЕСТАЦІЇ

	A	B	D	E
1			Дацюк О. А. Розподілені бази даних	Свинчук О. В. Математичні моделі процесів і систем
2	№	ПІБ студента		
3	1	Власюк Іван Вікторович	nv	a
4	2	Гончарова Єлизавета Максимівна	nv	nv
5	3	Груздов Максим Олексійович	nv	nv
6	4	Дудкін Олексій Миколайович	nv	a
7	5	Занченко Олексій Дмитрович	a	a
8	6	Захарчук Євгеній Миколайович	na	nv
9	7	Котова Анна Андріївна	nv	nv
10	8	Кравченко Дарія Олександрівна	a	nv
11	9	Кубрак Сергій Володимирович	nv	nv
12	10	Кушнірук Тетяна Олександрівна	nv	a
13	11	Лагойда Михайло Мирославович	nv	nv
14	12	Мангуллі Юлія Дмитрівна	nv	nv
15	13	Марченко Кіріл Андрійович	nv	nv
16	14	Міньков Іван Вікторович	nv	nv
17	15	Мітлицький Владислав Віталійович	nv	nv
18	16	Моругий Олег Петрович	a	nv
19	17	Сарахман Артур Олегович	nv	nv
20	18	Сідорська Анна Володимирівна	nv	nv
21	19	Сокирка Кирило Вікторович	nv	a
22	20	Сугулов Єгор Сергійович	nv	a

21-05-2023-TI-02-attestation -



СТОРІНКА УСПІШНОСТІ

	A	B	C	D	E	F	G
1	ТВ-32мп	Семестри	Години	Кредити	Волинець		Гаврилюк
2		1	150	5	Інженерія даних та знань	90	Інженерія даних та знан
3		1	135	4.5	Інноваційний менеджмент та інтелектуа	64	Інноваційний менеджме
4		1	180	6	Інтелектуальний аналіз даних для задач	93	Інтелектуальний аналіз
5		1	60	2	Науково-дослідна робота за темою магі	95	Науково-дослідна робо
6		1	150	5	Розробка застосунків Інтернету речей та	85	Розробка застосунків Ін
7		1	30	1	Розробка застосунків Інтернету речей та	86	Розробка застосунків Ін
8		1	150	5	Хмарні та Грід-технології	74	Хмарні та Грід-технолог
9							
10	2023-2024	2	150	5	Візуалізація статистичних потокових д	75	Візуалізація статистич
11		2	150	5	Графові бази даних	95	Графові бази даних
12		2	120	4	Методи та засоби виявлення уразливо	95	Методи та засоби вияв
13		2	120	4	Методологія інженерії програмного за	60	Методологія інженерії
14		2	45	1.5	Методологія інженерії програмного за	60	Методологія інженерії
15		2	60	2	Науково-дослідна робота за темою ма	82	Науково-дослідна роб
16		2	150	5	Оброблення надвеликих масивів дани	66	Оброблення надвелик
17		2	90	3	Практичний курс іноземної мови для д	66	Практичний курс іноз
18		2	120	4	Проектування інформаційно-діагності	60	Проектування інформ
19		2	60	2	Сталий інноваційний розвиток	75	Сталий інноваційний р
20							
21	2024-2025	3	420	14	Практика	95	
22		3	360	12			

Відомості про результати навчання студентів

Додати файл

Провести аналіз

ВІДОМОСТІ ПРО РЕЗУЛЬТАТИ НАВЧАННЯ СТУДЕНТІВ, ДОПУЩЕНИХ ДО АТЕСТАЦІЇ,

в навчально-науковому інституті атомної та теплової енергетики
за освітньо-професійною програмою «Інженерія програмного забезпечення інтелектуальних кібер-фізичних систем в енергетиці»
спеціальності 121 «Інженерія програмного забезпечення»

Зазначені нижче здобувачі вищої освіти повністю виконали індивідуальні навчальні плани і отримали оцінки:

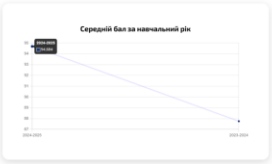
№ з/п	Прізвище, ім'я, по батькові	Кількість і частка отриманих оцінок										Частка оцінок «відмінно»	Частка оцінок «задовільно» та «достатньо»
		«відмінно» 100-95 балів		«дуже добре» 94-85 балів		«добре» 84-75 балів		«задовільно» 74-65 балів		«достатньо» 64-60 балів			
1	Волинець Ілля Олегович	4	22%	4	22%	3	17%	3	17%	4	22%	22%	39%
2	Гаврилюк Вадим Миколайович	4	22%	6	33%	3	17%	5	28%	0	0%	22%	28%
3	Гришкевич Максим Миколайович	9	50%	3	17%	3	17%	1	6%	2	11%	50%	17%
4	Дудченко Людмила Олександрівна	7	39%	6	33%	3	17%	2	11%	0	0%	39%	11%
5	Дичук Андрій Олегович	8	44%	6	33%	3	17%	1	6%	0	0%	44%	6%
6	Єрмоєєв Олексій Олександрович	8	44%	8	44%	1	6%	1	6%	0	0%	44%	6%
7	Єрмоєєва Олеся Олександрівна	10	56%	6	33%	1	6%	1	6%	0	0%	56%	6%
8	Левкун Дар'я Павлівна	6	33%	7	39%	3	17%	1	6%	1	6%	33%	11%
9	Лукінський Димітр Дмитрович	14	78%	4	22%	0	0%	0	0%	0	0%	78%	0%
10	Макрушин Андрій Миколайович	5	28%	4	22%	6	33%	1	6%	2	11%	28%	17%
11	Муляк Олександр Вікторович	8	44%	6	33%	2	11%	1	6%	1	6%	44%	11%
12	Омельченко Гліб Олександрович	10	56%	3	17%	3	17%	0	0%	2	11%	56%	11%
13	Палій Іван Олександрович	2	11%	2	11%	3	17%	5	28%	6	33%	11%	61%
14	Полякова Анастасія Русланівна	7	39%	7	39%	1	6%	0	0%	3	17%	39%	17%
15	Пятниця Марина Володимирівна	8	44%	8	44%	1	6%	1	6%	0	0%	44%	6%
16	Садовська Ірина Ігорівна	11	61%	3	17%	3	17%	1	6%	0	0%	61%	6%
17	Стрельченко Ганна-Марія Григорівна	8	44%	9	50%	0	0%	1	6%	0	0%	44%	6%
18	Таратонова Олександра Сергіївна	8	44%	6	33%	2	11%	2	11%	0	0%	44%	11%
19	Трощинський Олексій Андрійович	14	78%	2	11%	2	11%	0	0%	0	0%	78%	0%

Директор НН ІАТЕ Євген ПИСЬМЕННИЙ

АНАЛІЗ УСПІШНОСТІ

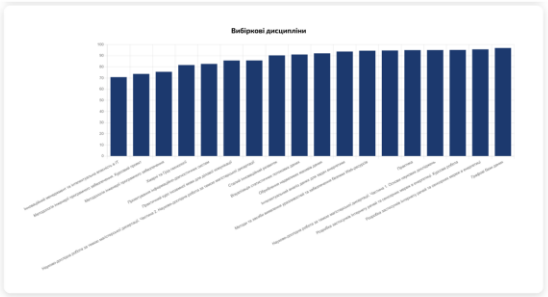
Аналіз академічної успішності

Студент	Середній бал студента за навчальний рік		Середній бал за всі роки	Відхилення від середнього бала по курсу
	2024-2025	2023-2024		
Волинець Ілля Олегович	95,00	77,71	78,67	-8,26
Гаврилюк Вадим Миколайович	97,00	83,18	83,94	-2,98
Гришкевич Максим Миколайович	98,00	86,47	87,11	0,18
Дудченко Людмила Олександрівна	96,00	87,24	87,12	0,79



Середній бал по дисциплінах

Навчальний рік	Предмет	Середній бал
2023-2024	Інноваційний менеджмент та інтелектуальна власність в IT	70,79
2023-2024	Методологія інженерії програмного забезпечення. Курсовий проект	73,68
2023-2024	Методологія інженерії програмного забезпечення	75,58



Рейтинг студентів

Студенти, які мають оцінку *Дуже добре* або вище (85+) з усіх предметів

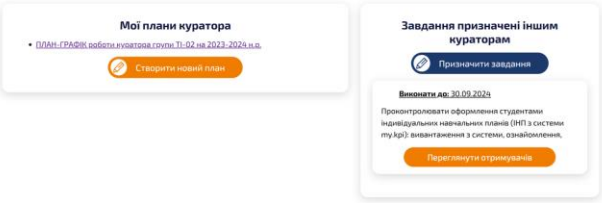
1. Лукінський Димітр Дмитрович

Студенти, які мають оцінку *Добре* або вище (75+) з усіх предметів

1. Трощинський Олексій Андрійович

Студенти, які мають оцінку *Задовільно* або вище (65+) з усіх предметів

СТОРІНКА ПЛАНІВ КУРАТОРА



Сторінка планів куратора для керівника

ПЛАН-ГРАФІК
роботи куратора групи ТІ-02
на 2023-2024 н.р.

№	Захід	Термін	Відмітка про виконання	Видалити
1	Провести зі студентами он-лайн зустрічі. Довести до відома сайти НТУУ КПІ https://osvita.kpi.ua/index.php/docs https://dnvr.kpi.ua/ https://t.me/dnvr_31 https://kpi.ua/4students	2024-09-01	☑	×
2	Сформувати індивідуальні навчальні плани (ІНП), підготувати і завантажити їх з системи My.kpi, надати для ознайомлення, роздрукування та підписання студентами	2024-09-08	☑	×
3	Проконтролювати оформлення студентами індивідуальних навчальних планів (ІНП з системи my.kpi): вивантаження з системи, ознайомлення, підписання та подання на кафедру	2024-09-30	☑	×
4	Проводити аналіз результатів календарного контролю успішності студентів та семестрового контролю	2024-12-31	☐	×
5	Надавати здобувачам освіти інформацію від завідуючого кафедрою, директора інституту та офіційних осіб університету	2024-12-31	☐	×
6	Надавати студентам інформацію щодо часу та порядку проведення навчальних заходів поточного, календарного та семестрового контролю з наданням відповідної документації	2024-12-31	☐	×
7	Зібрати зі студентами копії документів для замовлення дипломів та подати в деканат	2024-12-31	☑	×

+ Додати завдання

Приклад плану куратора

ВИСНОВКИ

Створено інформаційну систему забезпечення навчально-виховної діяльності катедри, яка є комплексним інструментом, розробленим для ефективного керування навчальними даними та адміністративними процесами, пов'язаними з навчально-виховною діяльністю катедри.

- Проаналізовано аналогічні інформаційні системи.
- Визначено основні потреби користувачів і вимоги до системи.
- Розроблено вебзастосунок, який відповідає потребам навчально-виховної діяльності катедри і складає функціонал системи.
- Розроблено зручний персоналізований інтерфейс.

Дана робота апробована на XXIV Всеукраїнській науково-технічній конференції молодих вчених, аспірантів та студентів «Стан, досягнення та перспективи інформаційних систем і технологій».

ДОДАТОК В Публікації за темою дисертації

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Одеський національний технологічний університет
Університет Інформатики і прикладних знань, м.Лодзь, Польща
Інститут комп`ютерної інженерії, автоматизації, робототехніки та програмування ім.П.Н.Платонова

XXIV Всеукраїнська науково-технічна конференція молодих вчених, аспірантів та студентів

«СТАН, ДОСЯГНЕННЯ ТА ПЕРСПЕКТИВИ ІНФОРМАЦІЙНИХ СИСТЕМ І ТЕХНОЛОГІЙ»

Матеріали конференції

Одеса

18-19 квітня 2024 р.

УДК 004.9

ІНФОРМАЦІЙНА СИСТЕМА ЗАБЕЗПЕЧЕННЯ НАВЧАЛЬНО-ВИХОВНОЇ ДІЯЛЬНОСТІ КАТЕДРИ

ДЯЧУК А.О., СВИНЧУК О.В., БАНДУРКА О.І. (anduadia@gmail.com)
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Інформаційна система для навчально-виховного напряму катедри є комплексним програмним забезпеченням, що об'єднує в собі різноманітні модулі та інструменти, спрямовані на автоматизацію та підтримку різноманітних аспектів діяльності катедри, пов'язаних з науковою роботою, навчанням і вихованням студентів та організаційно-управлінською діяльністю.

Сьогодні інформаційні технології стають невід'ємною частиною діяльності будь-якої організації. Наявні в університетах системи зазвичай не можуть повністю задовольнити усі потреби катедри, зокрема її навчально-виховний напрям. Створення інформаційної системи допоможе автоматизувати бізнес-процеси на катедрі, покращити доступ до інформації, сприяти комунікації між науково-педагогічними працівниками і студентами катедри. Ці задачі допомагають оптимізувати людські ресурси та підвищують якість освіти.

Архітектура інформаційної системи зазвичай поділяється на модулі (або компоненти) з метою полегшення розробки, тестування та підтримки системи. Кожен модуль відповідає за конкретну функціональність, що дозволяє зосередитися на окремих частинах системи без необхідності розуміння всієї системи загалом, що полегшує розробку та підтримку коду. Модульна архітектура дозволяє легко масштабувати систему шляхом додавання або зміни модулів без значного впливу на інші частини системи [1].

Запропонована архітектура інформаційної системи забезпечення навчально-виховної діяльності катедри зображена на рисунку 1.

Рис. 1. Архітектура інформаційної системи

Користувацький інтерфейс в архітектурі інформаційної системи для навчально-виховного напряму катедри є ключовим елементом, який забезпечує взаємодію користувачів з системою. Однією з найголовніших задач інтерфейсу є безпека даних та обмеження доступу до функціоналу для різних категорій користувачів через автентифікацію та авторизацію. Проте доступ до функціоналу повинен контролюватися як зі сторони інтерфейсу, так і зі сторони бізнес-логіки, оскільки користувачі можуть підроблювати дані для спроби несанкціонованого доступу до

конфіденційної інформації та зловживання своїми правами. Такий підхід дозволяє створити більш безпечну систему, яка відповідає вимогам сучасних стандартів безпеки та гарантує правильність роботи.

Модуль управління науково-виховною роботою дозволяє покращити організацію та ефективність науково-виховної діяльності катедри, сприяючи розвитку науки та підвищенню кваліфікації студентів та викладачів. Компонент дозволяє складати плани науково-виховної діяльності катедри, встановлювати цілі та завдання і призначати відповідальних за їхнє виконання. Доступними також є засоби для аналізу результатів науково-виховної роботи, формування звітів і статистичної інформації.

Модуль управління навчальним процесом потрібний для ефективного планування, організації та контролю за навчальним процесом студентів та викладачів. Компонент надає можливості створення і редагування розкладу занять, формування індивідуальних планів студентів, призначення викладачів для проведення занять. Також він забезпечує можливість аналізу успішності студентів, формування звітів щодо навчальних досягнень, складання академічних рейтингів.

Модуль співпраці та комунікації відіграє важливу роль у забезпеченні ефективного обміну інформацією та співпраці між усіма учасниками освітнього процесу на катедрі. Модуль забезпечує можливість обміну повідомленнями, документами та навчальними матеріалами між викладачами, студентами та адміністрацією катедри. До цього модулю можна віднести планування та відстеження подій, такі як лекції, семінари, конференції та інші події, що відбуваються на катедрі.

Інформаційний модуль призначений для перегляду детальної інформації про студентів, викладачів і груп на катедрі. Звісно, вся ця інформація повинна бути обмеженою залежно від ролі, яку має користувач у системі. Для оптимізації та підвищення якості пошуку інформації варто забезпечити семантичний пошук. Завдяки розумінню семантики запиту, семантичний пошук може надавати більш точні та релевантні результати.

База даних є одним з ключових компонентів архітектури інформаційної системи. Централізована база даних використовується для зберігання, організації та управління даними, які використовуються в системі. Дані у базі даних організовані у вигляді таблиць (реляційна модель), що дозволяє ефективно зберігати та управляти великим обсягом інформації, а також встановлювати зв'язки між різними елементами даних. СКБД повинна дозволяти виконувати складні запити до даних, щоб отримувати потрібну інформацію, що є важливим фактором для підтримки різноманітних функцій системи, таких як формування звітів або аналіз даних. Нормалізація схеми бази даних є важливою для продуктивності роботи системи та зменшення кількості помилок.

Інформаційна система для навчально-виховного напряму катедри повинна мати можливість інтеграції з іншими системами університету. API (прикладний програмний інтерфейс) може бути ключовим компонентом для забезпечення інтеграції з іншими системами. Використання API дозволяє зручно та ефективно обмінюватися даними між різними системами, забезпечуючи їх взаємодію та синхронізацію.

Отже, інформаційна система забезпечення навчально-виховної діяльності катедри спрямована на автоматизацію та покращення управління освітнім процесом та науковою діяльністю на катедрі. Архітектура системи включає в себе ряд модулів, які дозволяють ефективно планувати, вести облік та аналізувати наукову та навчальну діяльність катедри.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. *Information Systems*, Southern Africa: Mancosa, 2019. [Online]. Available: https://www.academia.edu/42801669/INFORMATION_SYSTEMS_Module_Guide. Accessed on: 12.04.2024.