Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій
(повне найменування факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)
бакалавр
(ступінь вищої освіти)

на тему СТВОРЕННЯ ПРОГРАМНОЇ СИСТЕМИ ОПРАЦЮВАННЯ ДАНИХ ДЛЯ КЕРУВАННЯ РОЗКЛАДОМ
DESIGN OF A DATA PROCESSING SYSTEM FOR SCHEDULE MANAGEMENT

Виконав(ла): студент(ка) 4 курсу, групи КНТ-213сп
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Комп'ютерні науки

ОСТАПЕНКО Р.С.
(ПРІЗВИЩЕ та ініціали)
Керівник СКРУПСЬКИЙ С.Ю.
(ПРІЗВИЩЕ та ініціали)
Рецензент КОЦУР М.І.
(ПРІЗВИЩЕ та ініціали)

2026

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
_____Сергій СУББОТІН
"____" _____2026 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ОСТАПЕНКА Романа Сергійовича
(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Створення програмної системи опрацювання даних для керування розкладом. Design of a Data Processing System for Schedule Management

керівник проєкту (роботи) к.т.н., доцент, СКРУПСЬКИЙ Степан Юрійович ,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)
затверджені наказом закладу вищої освіти від " 07 " квітня 2026 року № 139
2. Строк подання студентом проєкту (роботи) 10 червня 2026 року
3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис системи опрацювання даних. 4. Експлуатація, тестування та експериментальне дослідження системи опрацювання даних.

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

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	СКРУПСЬКИЙ С.Ю., доцент		
Нормоконтроль	КАЛІНІНА М.В., асистент		

7. Дата видачі завдання “ 20 ” квітня 2026 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій	2 тиждень	Розділ 2
	розробки.		
4	Розробка структури системи опрацювання даних.	2 тиждень	Розділ 3
5	Розробка системи опрацювання даних.	3-4 тижні	Розділи 3, 4
6	Тестування та експериментальне дослідження	5 тиждень	Розділ 4
	системи опрацювання даних.		
7	Оформлення пояснювальної записки та документів	6 тиждень	Додатки
	до неї.		
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка) _____ Роман ОСТАПЕНКО
(підпис) (Імя ПРІЗВИЩЕ)

Керівник проєкту (роботи) _____ Степан СКРУПСЬКИЙ
(підпис) (Імя ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 89 с., 4 табл., 29 рис., 3 дод., 20 джерел.

NETBEANS, PHP, АЛГОРИТМ, БАЗА ДАНИХ, МОВА ПРОГРАМУВАННЯ, РОЗКЛАД.

Об’єкт дослідження – алгоритми та процеси обчислень, пов’язані з організацією та опрацюванням даних для керування розкладом.

Предмет дослідження – програмні системи опрацювання даних для керування розкладом.

Мета роботи – розробка програмної системи опрацювання даних для керування розкладом для скорочення трудомісткості процесів планування та адміністрування розкладу.

Матеріали, методи та технічні засоби: мова програмування PHP, середовище розробки NetBeans.

Результати. Розроблено проєктні рішення для створення системи опрацювання даних для керування розкладом. Створено систему опрацювання даних для керування розкладом. Проведено дослідження розробленої системи опрацювання даних для керування розкладом.

Висновки. Мету роботи досягнуто. Розроблено систему опрацювання даних для керування розкладом за допомогою мови програмування PHP та середовища розробки NetBeans.

Галузь використання – заклади освіти та інші організації, де необхідно створювати розклад.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 89 pages, 4 tables, 29 figures, 3 appendixes, 20 sources.

NETBEANS, PHP, ALGORITHM, DATABASE, PROGRAMMING LANGUAGE, SCHEDULE.

The object of research is algorithms and computational processes related to the organization and processing of data for schedule management.

The subject of the research is data processing systems for schedule management.

The purpose of this work is to develop the data processing system for schedule management to reduce the complexity of the processes of planning and administration of the schedule.

Materials, methods and technical tools: PHP programming language, NetBeans development environment.

Results. Project solutions for the creation of data processing system for schedule management has been designed. The data processing system for schedule management has been developed. The study of the developed data processing system for schedule management has been conducted.

Conclusions. The goal of the work has been achieved. The data processing system for schedule management using the PHP programming language and the NetBeans development environment has been developed.

Scope of use – educational institutions and other organizations where it is necessary to create a schedule.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД – база даних;

ІС – інформаційна система;

КР – керування розкладом;

СОД – система опрацювання даних;

СУБД – система управління базами даних.

ВСТУП

Актуальність створення системи опрацювання даних для керування розкладом зумовлюється зростанням складності організаційних процесів та підвищенням вимог до ефективності управління часовими і ресурсними параметрами діяльності. У сучасних умовах функціонування різних організацій спостерігається постійне збільшення обсягів інформації, пов’язаної з плануванням подій, координацією людських ресурсів та використанням матеріально-технічної бази. Обробка таких даних традиційними засобами не забезпечує належного рівня точності, оперативності та узгодженості управлінських рішень, що обумовлює необхідність впровадження спеціалізованих інформаційних систем [1]-[2].

Значна увага в управлінській практиці приділяється питанню мінімізації конфліктів у розкладі та раціонального використання доступних ресурсів. У цьому контексті система опрацювання даних для керування розкладом розглядається як інструмент, здатний забезпечити формалізацію правил планування, урахування обмежень і взаємозв’язків між окремими елементами процесу, а також підтримку адаптації розкладу до змін зовнішніх і внутрішніх умов. Використання автоматизованих механізмів аналізу даних дозволяє підвищити узгодженість рішень і зменшити вплив людського чинника на результати планування [2]-[4].

Особливу актуальність набуває можливість аналітичної обробки накопичених даних з метою виявлення закономірностей і тенденцій у використанні часу та ресурсів. На основі таких даних створюються передумови для прогнозування навантаження, оцінювання ефективності чинних підходів до планування та формування рекомендацій щодо їх удосконалення. Це сприяє переходу від реактивного управління до більш обґрунтованого й адаптивного планування діяльності [4]-[5].

Отже, актуальність створення системи опрацювання даних для керування розкладом визначається об’єктивною потребою в підвищенні

ефективності планування, забезпеченні гнучкості управління та використанні аналітичних можливостей сучасних інформаційних технологій. Реалізація такої системи сприяє вдосконаленню організаційних процесів і відповідає загальним тенденціям розвитку цифрових рішень у сфері управління [6]-[7].

Проте деякі системи опрацювання даних для керування розкладом мають обмежену здатність повноцінно враховувати складні, неформалізовані або суперечливі вимоги, що виникають у процесі керування розкладом. Це призводить до ситуацій, коли автоматично сформований розклад потребує додаткового ручного коригування, що знижує очікуваний ефект від використання програмного забезпечення. Проблемним аспектом також залишається недостатня гнучкість багатьох систем опрацювання даних для керування розкладом у випадках частих змін або нестандартних ситуацій. Хоча програмні засоби орієнтовані на автоматичну перебудову розкладу, їх алгоритмічні моделі не завжди здатні оперативно адаптуватися до складних сценаріїв, пов’язаних із пріоритетами, людським фактором або зовнішніми обмеженнями. У результаті знижується рівень довіри користувачів до системи та зростає потреба у втручанні з боку відповідальних осіб. Ще одним недоліком є складність впровадження та налаштування програмних систем керування розкладом. Процес адаптації програмного забезпечення до конкретних умов організації може вимагати значних часових і ресурсних витрат, а також залучення фахівців з відповідною кваліфікацією. За відсутності належної підготовки персоналу використання таких систем може супроводжуватися помилками, некоректною експлуатацією або частковим ігноруванням функціональних можливостей програмного продукту. Крім того, обмеження користувацького інтерфейсу та недостатня зручність взаємодії з системою можуть негативно впливати на ефективність її застосування. Складні або перевантажені інтерфейси ускладнюють сприйняття інформації та уповільнюють процес прийняття рішень, що суперечить основній меті автоматизації керування розкладом. У поєднанні з низьким рівнем інтеграції з іншими інформаційними системами це може зменшувати загальну цінність

програмного рішення. Таким чином, програмні системи керування розкладом, попри їх потенціал щодо оптимізації планування, мають низку недоліків, пов’язаних з обмеженою адаптивністю, залежністю від якості даних, складністю впровадження та експлуатації [1]-[7].

Тому актуальною є розробка системи опрацювання даних для керування розкладом. У дипломній кваліфікаційній роботі бакалавра розв’язується актуальне завдання розробки системи опрацювання даних для керування розкладом для скорочення трудомісткості процесів планування та адміністрування розкладу.

Для досягнення поставленої мети у кваліфікаційній роботі бакалавра необхідно розв’язати такі задачі:

– виконати аналіз предметної області та систем опрацювання даних для керування розкладом;

– здійснити проєктування системи опрацювання даних для керування розкладом;

– створити систему опрацювання даних для керування розкладом;

– дослідити розроблену систему опрацювання даних для керування розкладом.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Системи опрацювання даних для керування розкладом

Система опрацювання даних для керування розкладом дозволяє виконувати впорядкування, зберігання та оброблення інформації, пов’язаної з плануванням подій, занять або робочих процесів у межах певної організації. Її основне призначення полягає у забезпеченні узгодженості між різними елементами розкладу, такими як час, ресурси, учасники та правила використання, шляхом централізованого управління відповідними даними. Функціонування таких систем ґрунтується на використанні єдиного інформаційного середовища, що дозволяє уникати дублювання даних і зменшувати ймовірність помилок під час планування [1].

У процесі застосування систем опрацювання даних для керування розкладом створюються умови для автоматизації рутинних операцій, які традиційно виконуються вручну. Завдяки цьому спрощується формування, коригування та аналіз розкладів, а також підвищується прозорість прийняття управлінських рішень. Такі програмні рішення зазвичай підтримують взаємодію між різними групами користувачів, забезпечуючи доступ до актуальної інформації в реальному часі та сприяючи покращенню координації дій між учасниками процесу [1].

Особливістю подібних програмних систем є здатність адаптуватися до специфіки діяльності конкретної установи або організації. Вони можуть враховувати встановлені обмеження, внутрішні правила та доступні ресурси, що дозволяє формувати збалансовані та логічно узгоджені розклади. Використання таких програмних засобів сприяє підвищенню ефективності використання часу та ресурсів, зменшенню кількості конфліктних ситуацій і покращенню загальної організації процесів. У підсумку система опрацювання даних для керування розкладом виступає універсальним інструментом підтримки планування, аналізу та оптимізації діяльності в різних сферах застосування [1].

Система керування розкладом закладу освіти трактується як інформаційне програмне середовище, призначене для впорядкування та координації навчального часу в межах освітнього закладу. Її застосування спрямоване на підтримку процесів формування, коригування та контролю розкладів занять, перевірок знань і розподілу педагогічного навантаження. У ході розвитку таких рішень було вдосконалено підходи до організації навчальної діяльності, що дало змогу адміністративному персоналу більш гнучко реагувати на зміни та оптимізувати щоденні управлінські дії [1].

Функціонування подібної програмної системи передбачає тісну взаємодію з інформаційними ресурсами, що містять відомості про учнів, що забезпечує цілісність даних та спрощує комунікацію між усіма учасниками освітнього процесу. Завдяки інтеграції з обліковими підсистемами створюються умови для узгодженого ведення навчальної документації, обміну інформацією та оперативного інформування зацікавлених сторін про актуальні зміни [1]-[2].

Застосування сучасних програмних рішень для планування навчального процесу зумовлюється потребою підвищення ефективності управління та зниження адміністративного навантаження. У межах таких програмних систем забезпечується актуалізація даних у режимі безперервного оновлення, що сприяє своєчасному реагуванню на зміни в організації занять. Водночас автоматизоване опрацювання інформації дозволяє зменшити кількість помилок, пов’язаних із ручним введенням даних, та підвищити точність обліку навчальної активності [1]-[2].

Важливою характеристикою програмних систем для планування є їхня роль у впорядкуванні внутрішніх процесів закладу освіти. За рахунок інтелектуальних механізмів автоматизації досягається оптимізація розкладів і зменшення конфліктних ситуацій, пов’язаних із перетином занять, нестачею ресурсів або нерівномірним навантаженням. Такі системи враховують доступність педагогів, вимоги навчальних програм і організаційні обмеження, формуючи узгоджену структуру навчального часу [1]-[2].

Окрему увагу приділяється плануванню різних видів навчальної та позанавчальної діяльності, що дозволяє забезпечити баланс між академічними заняттями та іншими заходами. У разі змін у розкладі інформація оперативно доводиться до користувачів через цифрові канали зв’язку, що підтримує поінформованість та знижує рівень організаційної невизначеності. Взаємодія з підсистемами обліку відвідуваності створює можливості для точного контролю участі учнів у навчальному процесі без додаткових ручних процедур [2]-[3].

Гнучкість налаштувань таких програмних систем дозволяє адаптувати їх до специфічних вимог конкретного навчального закладу. Адміністративний персонал може встановлювати власні правила планування, що відображають внутрішні політики, особливості навчальних програм та доступність спеціалізованих приміщень. У разі виникнення суперечностей система автоматично виявляє проблемні ситуації та сприяє їх усуненню, забезпечуючи стабільну й упорядковану організацію навчального процесу. Загалом використання добре спроєктованої системи керування шкільним розкладом сприяє підвищенню ефективності роботи закладу освіти, раціональному використанню ресурсів і чіткому структуруванню навчального часу [2]-[3].

Програмна система керування шкільним розкладом є ключовим елементом підтримки стабільного та впорядкованого освітнього процесу, оскільки вона забезпечує раціональну організацію навчального часу та узгодження всіх складових розкладу. Її використання спрямоване на усунення неточностей, характерних для ручного планування, а також на підвищення загальної результативності діяльності навчального закладу. Завдяки автоматизованому підходу створюються умови для більш злагодженого управління навчальними процесами та зменшення кількості організаційних збоїв [2]-[3].

Однією з основних переваг таких програмних систем є оптимізація використання часу, оскільки в межах одного інформаційного середовища узгоджуються заняття, педагогічне навантаження та навчальні групи. Це дозволяє уникати накладання уроків, нераціонального розподілу годин і

повторного призначення занять на один і той самий проміжок часу. Автоматизоване формування розкладів значно скорочує потребу в численних коригуваннях і дає змогу зосередитися на якості організації навчального процесу, а не на технічних аспектах його планування [2]-[3].

Не менш важливою є роль системи у раціональному використанні матеріальних і кадрових ресурсів. За її допомогою забезпечується збалансований розподіл викладачів, навчальних аудиторій та спеціалізованого обладнання, що зменшує ризик перевантаження персоналу й неефективного використання приміщень. Такий підхід сприяє створенню комфортних умов для проведення занять і підвищує загальний рівень організації навчального середовища [3]-[4].

Використання системи керування розкладом також покращує інформаційну взаємодію між учасниками освітнього процесу. Актуальні відомості про зміни в розкладі, контрольні заходи або додаткові події оперативно доводяться до відома учнів, викладачів і батьків через цифрові канали зв’язку. Це забезпечує своєчасне інформування та знижує рівень непорозумінь, пов’язаних з організаційними змінами [3]-[4].

Автоматизація процесів планування істотно зменшує адміністративне навантаження на працівників закладу освіти. Замість тривалого ручного узгодження даних використовується централізований інструмент, який дозволяє керувати розкладами для всіх класів і груп у єдиній системі. У результаті підвищується ефективність роботи адміністрації, а управління навчальним процесом стає більш структурованим і прозорим [3]-[4].

Функціонування таких програмних систем ґрунтується на поетапному опрацюванні інформації, починаючи з внесення вихідних даних щодо доступності викладачів, навчальних груп і ресурсів. На основі заданих правил автоматично формується узгоджений розклад, після чого система аналізує його на наявність суперечностей. У разі внесення змін інформація оновлюється без затримок, а всі зацікавлені сторони отримують відповідні повідомлення.

Завдяки цьому забезпечується безперервність навчального процесу та висока керованість освітньої діяльності [3]-[4].

Вибір ефективної програмної системи для планування навчального процесу в закладах освіти має вирішальне значення для оптимізації адміністративної роботи, оскільки автоматизація процесів замінює трудомістке ручне складання розкладів, що потребує значних витрат часу та зусиль. Використання сучасних рішень дозволяє забезпечити організоване управління навчальними заняттями, обліком ресурсів і координацію роботи педагогічного персоналу без зайвої паперової роботи та помилок [4]-[5].

Однією з основних характеристик таких систем є інтуїтивно зрозумілий інтерфейс, який спрощує взаємодію користувачів із платформою та дозволяє адміністраторам швидко налаштовувати та коригувати розклади. Підтримка інтеграції з інформаційними системами закладу забезпечує синхронізацію даних учнів, їхніх профілів, відвідуваності та результатів навчання, що дозволяє ефективно координувати процеси та уникати розривів у передачі інформації. Системи також пропонують гнучкі налаштування, що враховують специфіку конкретного навчального закладу, включаючи правила розподілу навантаження, використання приміщень та спеціалізованого обладнання, а також масштабованість для роботи з різними розмірами закладів освіти [4]-[5].

Інтеграція з системою управління інформацією про учнів дозволяє підтримувати точність і цілісність даних, забезпечуючи оновлення профілів студентів, контроль відвідуваності та планування іспитів у синхронізованому режимі. Це забезпечує безперервний обмін даними, зменшує кількість ручних помилок і підвищує ефективність адміністративної діяльності. Взаємодія з системою управління відвідуваністю учнів дозволяє автоматично відстежувати пропуски, генерувати звіти про участь у навчальному процесі та надсилати сповіщення батькам через електронну пошту або миттєві повідомлення, що підвищує прозорість та контроль за навчальним процесом [4]-[5].

У результаті застосування таких програмних систем досягається комплексне управління закладу освіти: створення розкладів, управління

ресурсами та відстеження участі здобувачів відбувається у єдиному інформаційному середовищі закладу освіти, що знижує адміністративне навантаження та підвищує продуктивність. Використання добре спроєктованих платформ для планування дозволяє закладам освіти ефективно координувати час, покращувати обмін інформацією між усіма учасниками освітнього процесу та досягати кращих навчальних результатів без додаткових витрат на ручне опрацювання даних [4]-[5].

Керування розкладом закладу освіти сьогодні сприймається не лише як просте розподілення занять за часовими слотами, а як комплексний процес організації навчальної діяльності з високим рівнем точності та водночас адаптивності. Традиційні методи часто не можуть врахувати швидкі зміни в потребах школи, що призводить до помилок, конфліктів та зниження ефективності. Сучасні навчальні заклади віддають перевагу спеціалізованому програмному забезпеченню, яке дозволяє максимально оптимізувати робочий процес, формуючи збалансований розклад для учнів і викладачів та зменшуючи перерви у навчальному процесі [5]-[6].

Програмні системи планування нового покоління суттєво скорочують обсяг ручної роботи, забезпечують більш раціональний розподіл ресурсів і зменшують ймовірність дублювання завдань. Автоматизація процесів дозволяє адміністраторам більш ефективно контролювати навчальну діяльність, забезпечуючи оптимальний розклад та підтримуючи освітній прогрес без зайвого втручання в деталі планування. Використання таких рішень економить час, знижує навантаження на персонал та сприяє стабільності академічного процесу [5]-[6].

Сучасні програмні системи для складання розкладу відрізняються рядом важливих характеристик. Такі системи здатні автоматично виявляти та усувати конфлікти у завданнях викладачів, розподілі класів та кімнат, що дозволяє уникати накладок та хаосу у навчальному процесі. Система підтримує персоналізовані шаблони, що враховують унікальні правила та особливості конкретного закладу, а також зовнішні заходи та події. Вона оперативно реагує

на непередбачувані зміни, оновлюючи розклади в режимі реального часу, наприклад у випадках відсутності викладачів або зміни приміщень. Інтеграція з іншими шкільними системами забезпечує узгодженість даних між базами учнів, інструментами оцінювання та адміністративними платформами. Зручний інтерфейс робить процес управління розкладом простим та інтуїтивним, дозволяючи вирішувати складні організаційні завдання без додаткового навчання персоналу [5]-[6].

Автоматизація радикально змінює підхід до планування, замінюючи громіздкий та схильний до помилок ручний процес на більш ефективну і точну систему. Сучасні програмні системи для складання розкладів використовує дані та алгоритми для створення збалансованих графіків, гарантуючи оптимальний розподіл кожного заняття, лабораторної роботи та позакласної активності без втрати часу та ресурсів. Завдяки цьому підходу забезпечується рівний доступ студентів до курсів та справедливе навантаження викладачів, що створює безперебійний навчальний процес та підвищує задоволеність усіх учасників освітнього середовища [5]-[6].

Поширені проблеми, пов’язані з плануванням, включають перевантаження викладачів та здобувачів, обмежену доступність приміщень, накладення курсів та необхідність оперативно вносити зміни. Програмні системи вирішують ці питання шляхом рівномірного розподілу годин викладачів, оптимізації використання аудиторій та балансування побажань студентів для максимального вибору курсів. Будь-які зміни при використанні відповідних програмних систем можна внести швидко, зберігаючи організованість графіка та мінімізуючи стрес для персоналу та учнів [5]-[6].

Визначено, що система опрацювання даних для керування розкладом дозволяє виконувати впорядкування, зберігання та оброблення інформації, пов’язаної з плануванням подій, занять або робочих процесів у межах певної організації. Її основне призначення полягає у забезпеченні узгодженості між різними елементами розкладу, такими як час, ресурси, учасники та правила використання, шляхом централізованого управління відповідними даними.

1.2 Аналіз систем опрацювання даних для керування розкладом

Проаналізуємо програмні системи опрацювання даних для керування розкладом [7]-[14].

Програмна система для керування розкладом Coursedog (рис. 1.1) позиціонується як комплексне рішення для освітніх установ, орієнтоване на підвищення ефективності академічного планування та оптимізацію внутрішніх процесів. Платформа використовується як інструмент стратегічної підтримки управлінських рішень у сфері організації навчального процесу та зарекомендувала себе як надійний партнер для закладів освіти різного масштабу. У процесі експлуатації система Coursedog демонструє відчутний позитивний вплив на структуру навчальних семестрів, зменшення кількості накладок між заняттями та раціональніше використання фінансових і просторових ресурсів навчального закладу [7], [12].

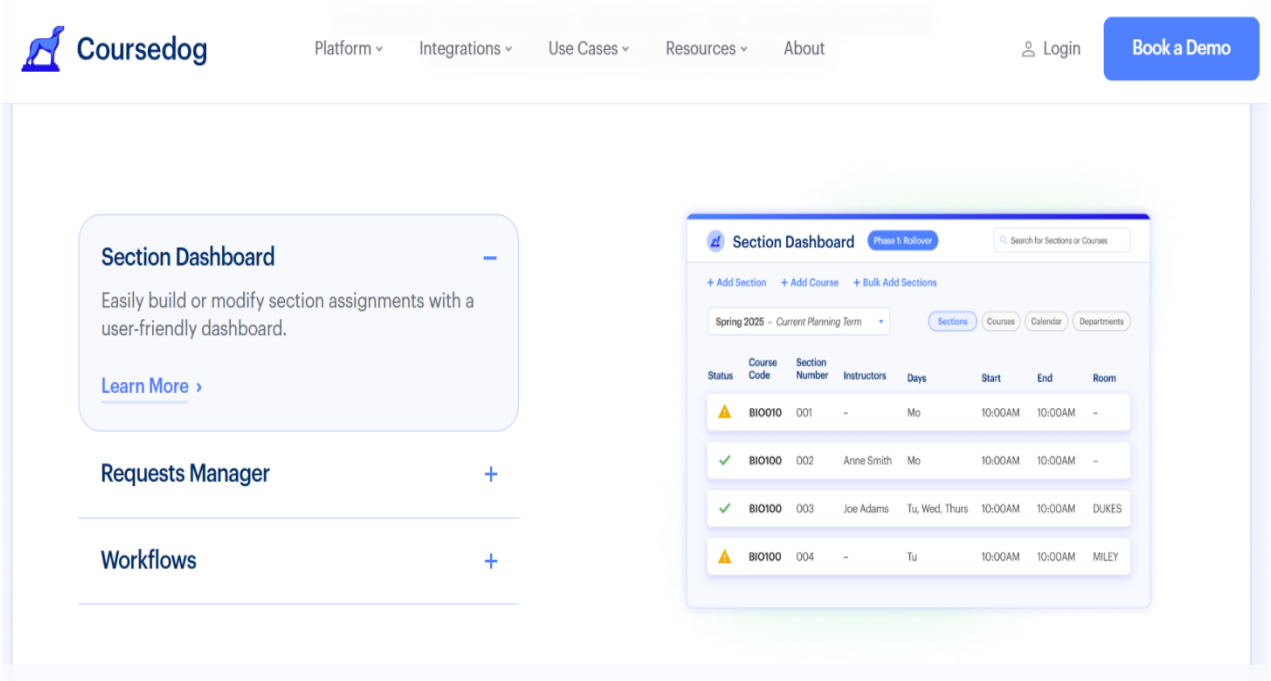

Рисунок 1.1 – Програмна система Coursedog [12]

Функціональні можливості Coursedog дозволяють формувати розклади з урахуванням наперед визначених правил і обмежень, що забезпечує узгодженість між курсами, аудиторіями та викладацьким складом. У

програмній системі Coursedog реалізовано механізми встановлення логічних зв’язків між навчальними приміщеннями, завдяки чому мінімізується ймовірність конфліктів під час розміщення занять. Значна увага приділяється аналізу використання простору, що дає змогу створювати інформаційні панелі для оцінювання завантаженості аудиторій та адаптації розкладу відповідно до потреб студентів, забезпечуючи збалансований і комфортний навчальний досвід [7], [12].

Система Coursedog також охоплює широкий спектр інструментів для управління академічними програмами, ведення навчальних каталогів і довідкових даних, а також підтримує аналітичні функції, спрямовані на оцінювання структури навчальних планів. Реалізовані засоби академічного планування сприяють більш прозорому розподілу навантаження між викладачами та дозволяють узгоджувати навчальні активності з подіями позааудиторного характеру. Окрему роль відіграють модулі прогнозування попиту на курси, які допомагають закладам освіти краще адаптувати пропозицію дисциплін до очікувань студентів і поточних тенденцій [7], [12].

Використання Coursedog також передбачає підтримку процесів оцінювання та відстеження їх циклів, що сприяє системному підходу до контролю якості освітніх програм. Завдяки поєднанню інструментів планування, аналітики та управління ресурсами, система розглядається як цілісне програмне середовище, здатне підвищити керованість навчального процесу, зменшити організаційні втрати та створити умови для сталого розвитку освітньої установи [7], [12].

Основними характеристиками програмного забезпечення Coursedog є такі [7], [12]:

– централізоване планування: програмна система Coursedog забезпечує єдине середовище для формування та узгодження розкладів, навчальних планів і подій з урахуванням академічних правил та внутрішніх політик закладу освіти [7], [12];

– керування навчальними програмами: програмна система Coursedog надає інструменти для створення, оновлення та аналізу освітніх програм закладів освіти, що дозволяє підтримувати їх актуальність і відповідність потребам студентів [7], [12];

– запобігання конфліктам: у програмній системі Coursedog реалізовано механізми автоматичного виявлення накладок між заняттями, аудиторіями та викладачами, що знижує ризик помилок під час складання розкладу [7], [12];

– аналітична підтримка: у програмній системі Coursedog використовується розширена аналітика для оцінювання ефективності використання ресурсів, навантаження викладачів та структури навчального процесу [7], [12];

– управління простором: у програмній системі Coursedog передбачено засоби аналізу зайнятості аудиторій і навчальних приміщень, що сприяє раціональному розподілу просторових ресурсів [7], [12];

– прогнозування попиту: у програмній системі Coursedog застосовуються інструменти для оцінювання очікуваного інтересу до курсів, що допомагає адаптувати розклад до реальних потреб студентів [7], [12];

– підтримка прийняття рішень: програмна система Coursedog надає узагальнену інформацію у зручному вигляді, що полегшує стратегічне та оперативне управління академічною діяльністю [7], [12];

– гнучка адаптація: програмна система Coursedog може налаштовуватися під особливості конкретного закладу освіти, включно з правилами, структурою та організаційними вимогами [7], [12];

– інтеграційні можливості: у програмній системі Coursedog підтримується взаємодія з іншими інформаційними системами навчального закладу для забезпечення цілісності даних та процесів [7], [12].

Серед недоліків програмного забезпечення Coursedog можна відзначити відносно високу вартість впровадження та супроводу, що може бути суттєвим обмеженням для невеликих освітніх установ. Також певні труднощі може викликати складність початкового налаштування та необхідність адаптації

персоналу до роботи з багатофункціональним програмним середовищем, що потребує часу та додаткових організаційних зусиль [7], [12].

Програмна система для керування розкладом Ad Astra (рис. 1.2) являє собою комплексне рішення для організації та оптимізації навчального процесу в освітніх установах. У процесі її використання забезпечується формування узгоджених розкладів навчальних занять, що дозволяє ефективно поєднувати академічні курси, викладацьке навантаження та доступні ресурси. Функціонування системи орієнтоване на інтелектуальне планування, завдяки чому створюються умови для зменшення накладок між дисциплінами та більш збалансованого розподілу занять між секціями. В результаті досягається підвищення прозорості навчального процесу та покращення можливостей для відстеження академічної активності студентів [7], [13].

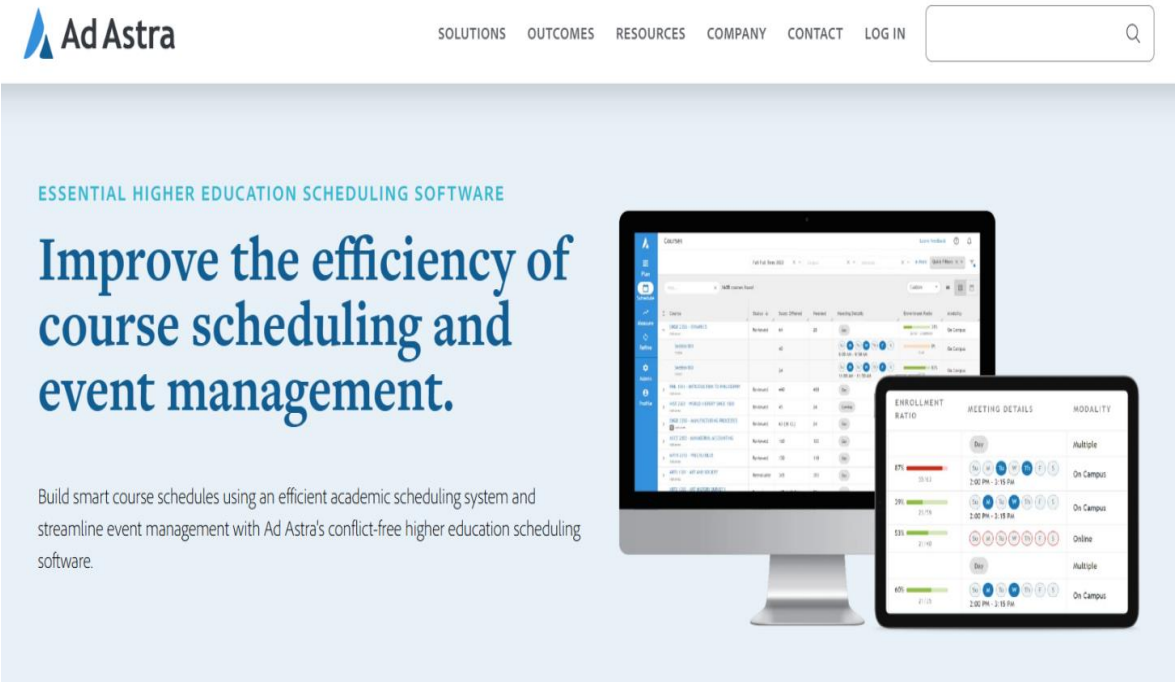

Рисунок 1.2 – Програмна система Ad Astra [13]

Під час інтеграції програмної системи Ad Astra в інформаційну інфраструктуру закладу освіти підтримується централізований підхід до управління розкладом, що спрощує координацію між підрозділами та учасниками освітнього процесу. Система надає інструменти для візуального відображення планування, редагування та узгодження навчальних секцій, а

також для публікації актуальної інформації у студентських сервісах. Додатково реалізуються механізми контролю доступу та спільної роботи з елементами розкладу, що сприяє впорядкуванню робочих процесів і зменшенню кількості організаційних помилок [7], [13].

Особливістю програмної системи Ad Astra є використання аналітичних можливостей для дослідження історії змін у розкладах та оцінювання ефективності прийнятих рішень. Це дозволяє враховувати попередній досвід під час планування майбутніх навчальних періодів і поступово вдосконалювати структуру академічного розкладу. Завдяки такому підходу програмна система Ad Astra виступає не лише інструментом технічного формування розкладу, а й засобом підтримки управлінських рішень у сфері освітнього планування [7], [13].

Програмна система Ad Astra має такі особливості [7], [13]:

– інтелектуальне планування: програмна система Ad Astra забезпечує формування навчальних розкладів із урахуванням доступних ресурсів, академічних обмежень та взаємозв’язків між курсами, що дозволяє зменшити кількість конфліктів і підвищити узгодженість навчального процесу [7], [13];

– візуалізація розкладів: програмна система Ad Astra надає інструменти для наочного відображення календарів і занять, що спрощує аналіз навантаження та коригування розкладу занять на різних етапах його формування [7], [13];

– керування секціями: у програмній системі Ad Astra реалізується можливість редагування, узгодження та об’єднання навчальних секцій, що сприяє більш гнучкому управлінню структурою курсів [7], [13];

– централізоване управління: у програмній системі Ad Astra підтримується єдиний інформаційний простір для роботи з розкладом, що забезпечує узгоджену взаємодію між адміністративним персоналом і викладачами [7], [13];

– контроль доступу: програмна система Ad Astra дозволяє розмежовувати права користувачів, що підвищує безпеку даних та впорядковує робочі процеси [7], [13];

– спільна робота: у програмній системі Ad Astra передбачено механізми колективного редагування та позначення елементів розкладу, що полегшує координацію дій між учасниками планування [7], [13];

– аналіз історії змін: програмна система Ad Astra зберігає дані про попередні версії розкладів, що дає змогу аналізувати прийняті рішення та враховувати їх у подальшому плануванні [7], [13];

– інтеграція з інформаційними системами: у програмній система Ad Astra забезпечується можливість поєднання з іншими освітніми сервісами для публікації та використання актуальних даних розкладу [7], [13].

Серед недоліків програмного забезпечення Ad Astra можна відзначити складність первинного впровадження та налаштування, а також необхідність додаткового навчання персоналу для повноцінного використання всіх функціональних можливостей системи, що може збільшувати часові та організаційні витрати на початкових етапах експлуатації [7], [13].

Програмна система для керування розкладом Arlo (рис. 1.3) являє собою комплексне рішення для організації навчальної діяльності в умовах поєднання різних форматів навчання [7], [14]. У процесі використання програмної системи Arlo забезпечується автоматизація значної частини операцій, які традиційно виконуються вручну під час планування та адміністрування освітніх програм. Функціональні можливості системи орієнтовані на підтримку викладачів і адміністративного персоналу, що дозволяє спростити підготовку навчальних заходів та зменшити навантаження, пов’язане з рутинними діями [7], [14].

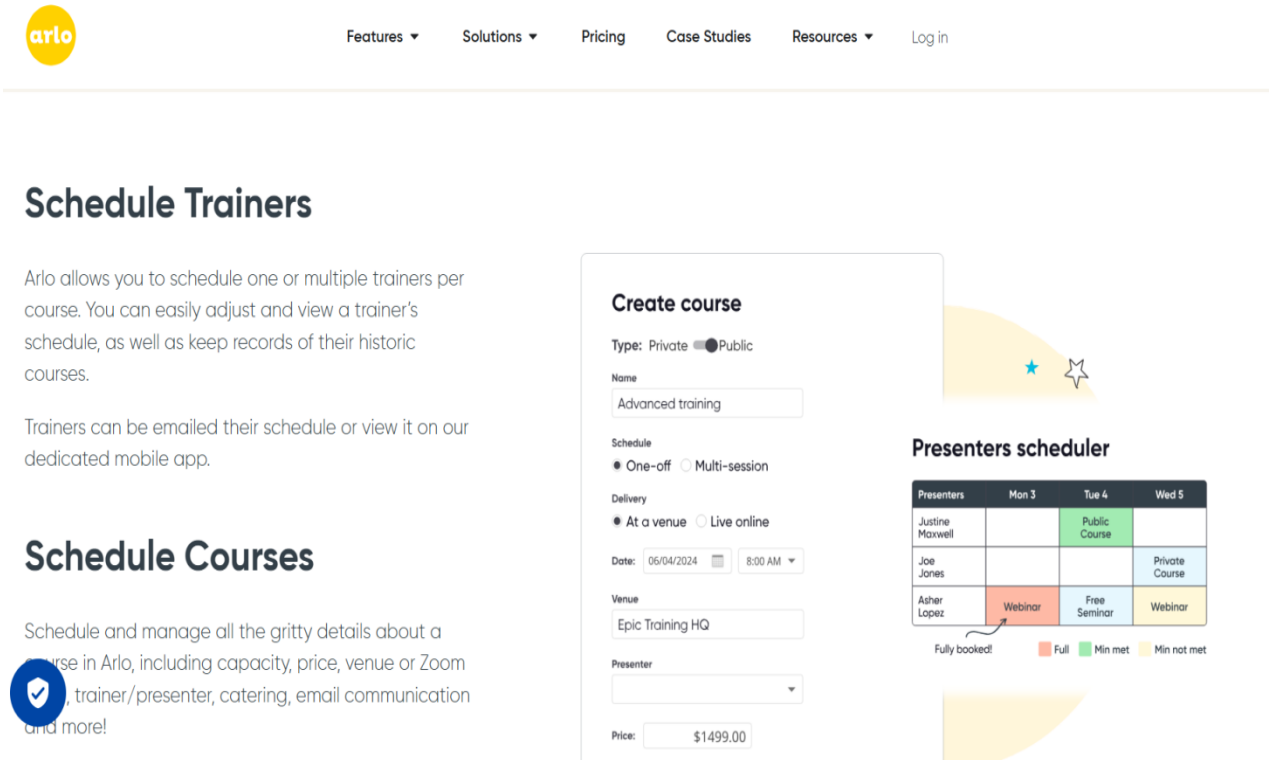

Рисунок 1.3 – Програмна система Arlo [14]

Під час інтеграції програмної системи Arlo в інформаційне середовище закладу освіти створюються умови для централізованого управління навчальними подіями, слухачами та запрошеними фахівцями. Система забезпечує узгоджене планування занять, координацію виступів тренерів і доповідачів, а також ведення обліку учасників навчального процесу. Завдяки поєднанню інструментів для очного, дистанційного та комбінованого навчання досягається гнучкість у формуванні розкладів і можливість адаптації до різних освітніх моделей [7], [14].

Особливістю програмної системи Arlo є її спрямованість на підтримку трансформаційних процесів у навчальних організаціях. Система виступає не лише засобом планування, а й платформою для впорядкування навчальних операцій, що сприяє підвищенню прозорості управління та покращенню взаємодії між усіма учасниками освітнього процесу. Завдяки накопиченому практичному досвіду використання таких рішень Arlo демонструє здатність ефективно підтримувати різні формати навчання та забезпечувати стабільну організацію навчальних заходів [7], [14].

Програмна система Arlo має такі особливості [7], [14]:

– змішаний формат навчання: програмна система Arlo підтримує поєднання очних, дистанційних і комбінованих освітніх заходів, що забезпечує гнучкість у формуванні розкладів та адаптацію до різних моделей навчального процесу [7], [14];

– автоматизація адміністративних процесів: програмна система Arlo спрямована на скорочення ручної роботи під час організації навчальних програм, планування подій та керування пов’язаними завданнями [7], [14];

– централізоване планування курсів: у програмній системі Arlo реалізується можливість узгодженого керування розкладом курсів, семінарів і тренінгів у межах єдиного інформаційного середовища [7], [14];

– керування ресурсами: програмна система Arlo дозволяє планувати використання навчальних приміщень, залучення тренерів і доповідачів без виникнення конфліктів [7], [14];

– реєстрація учасників: програмна система Arlo забезпечує облік слухачів та спрощує процес їх зарахування до навчальних заходів [7], [14];

– інтеграція з онлайн сервісами: у програмній системі Arlo підтримується взаємодія з інструментами дистанційного навчання, що розширює можливості організації освітнього процесу [7], [14];

– підтримка навчального менеджменту: у програмній системі Arlo реалізуються інструменти для супроводу навчальних операцій і впорядкування інформації щодо проведених заходів [7], [14];

– масштабованість використання: програмна система Arlo орієнтована на роботу з різною кількістю навчальних подій та користувачів без втрати стабільності [7], [14].

Серед недоліків програмного забезпечення Arlo можна відзначити обмежену гнучкість налаштувань для специфічних освітніх сценаріїв, а також залежність від зовнішніх онлайн сервісів, що може ускладнювати використання системи в умовах нестабільного мережевого доступу [7], [14].

Порівняльну характеристику програмних систем опрацювання даних для керування розкладом наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмних систем опрацювання даних для керування розкладом

Критерій порівняння	Coursedog [12]	Ad Astra [13]	Arlo [14]
Академічне планування та розклад	+	+	+–
Підтримка складних правил і обмежень	+	+–	–
Підтримка змішаного навчання	+–	+–	+
Простота впровадження та використання	+–	–	+
Аналітика та історія змін розкладу	+	+	–
Гнучкість налаштувань	+	+–	+–

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато систем опрацювання даних для керування розкладом. Проте деякі системи опрацювання даних для керування розкладом мають обмежену здатність повноцінно враховувати складні, неформалізовані або суперечливі вимоги, що виникають у процесі керування розкладом. Ще одним недоліком є складність впровадження та налаштування програмних систем керування розкладом. Процес адаптації програмного забезпечення до конкретних умов організації може вимагати значних часових і ресурсних витрат, а також залучення фахівців з відповідною кваліфікацією. Крім того, обмеження користувацького інтерфейсу та недостатня зручність взаємодії з системою можуть негативно впливати на ефективність її застосування. Складні або перевантажені інтерфейси ускладнюють сприйняття інформації та

уповільнюють процес прийняття рішень, що суперечить основній меті автоматизації керування розкладом. У поєднанні з низьким рівнем інтеграції з іншими інформаційними системами це може зменшувати загальну цінність програмного рішення. Таким чином, програмні системи керування розкладом, попри їх потенціал щодо оптимізації планування, мають низку недоліків, пов’язаних з обмеженою адаптивністю, залежністю від якості даних, складністю впровадження та експлуатації. Тому актуальною є розробка системи опрацювання даних для керування розкладом.

При розробці системи опрацювання даних для керування розкладом необхідно забезпечити такі функціональні вимоги:

– підтримка можливості збереження та відображення даних про розклад занять у базі даних;

– підтримка можливості додавання, редагування та видалення інформації, пов’язаної з розкладом;

– забезпечення реєстрації та авторизації користувачів із використанням механізмів контролю доступу та розмежування прав відповідно до ролей;

– забезпечення збереження та оброблення довідкових даних, що стосуються факультетів, кафедр, навчальних груп, викладачів та іншої інформації;

– забезпечення коректної взаємодії між клієнтською та серверною частинами з використанням стандартизованих форматів обміну даними;

– підтримка можливості зміни паролів для входу у систему зареєстрованих користувачів.

1.3 Висновки за розділом 1

Визначено, що система опрацювання даних для керування розкладом дозволяє виконувати впорядкування, зберігання та оброблення інформації, пов’язаної з плануванням подій, занять або робочих процесів у межах певної організації. Її основне призначення полягає у забезпеченні узгодженості між

різними елементами розкладу, такими як час, ресурси, учасники та правила використання, шляхом централізованого управління відповідними даними.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато систем опрацювання даних для керування розкладом. Проте деякі системи опрацювання даних для керування розкладом мають обмежену здатність повноцінно враховувати складні, неформалізовані або суперечливі вимоги, що виникають у процесі керування розкладом. Це призводить до ситуацій, коли автоматично сформований розклад потребує додаткового ручного коригування, що знижує очікуваний ефект від використання програмного забезпечення. Ще одним недоліком є складність впровадження та налаштування програмних систем керування розкладом. Процес адаптації програмного забезпечення до конкретних умов організації може вимагати значних часових і ресурсних витрат, а також залучення фахівців з відповідною кваліфікацією. Крім того, обмеження користувацького інтерфейсу та недостатня зручність взаємодії з системою можуть негативно впливати на ефективність її застосування. Складні або перевантажені інтерфейси ускладнюють сприйняття інформації та уповільнюють процес прийняття рішень, що суперечить основній меті автоматизації керування розкладом. У поєднанні з низьким рівнем інтеграції з іншими інформаційними системами це може зменшувати загальну цінність програмного рішення. Таким чином, програмні системи керування розкладом, попри їх потенціал щодо оптимізації планування, мають низку недоліків, пов’язаних з обмеженою адаптивністю, залежністю від якості даних, складністю впровадження та експлуатації. Тому актуальною є розробка системи опрацювання даних для керування розкладом.

Сформульовано функціональні вимоги до системи опрацювання даних для керування розкладом.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

При розробці системи опрацювання даних для керування розкладом було використано мову програмування PHP [15]-[16].

PHP характеризується стабільністю та передбачуваністю поведінки в процесі виконання серверних застосунків, що є важливим для систем, орієнтованих на безперервну роботу та обробку запитів від великої кількості користувачів. Така властивість сприяє підвищенню надійності програмного забезпечення та зменшенню ризиків збоїв під час експлуатації [15]-[16].

Мова програмування PHP надає розвинені механізми для захисту вебзастосунків від типових загроз. Наявність стандартних засобів для фільтрації вхідних даних, керування сесіями та контролю доступу дозволяє реалізувати базові та розширені підходи до захисту інформації. У поєднанні з перевіреними практиками розробки це створює передумови для побудови безпечних систем опрацювання даних [15]-[16].

Важливим аспектом є можливість інтеграції PHP із зовнішніми сервісами та програмними компонентами. Мова підтримує різноманітні протоколи обміну даними та формати представлення інформації, що спрощує взаємодію з клієнтськими застосунками, сторонніми платформами та іншими інформаційними системами. Це дозволяє розглядати PHP як ефективний інструмент для побудови модульних і розширюваних архітектур [15]-[16].

Мова PHP є досить гнучкою у контексті архітектурних рішень та може використовуватися як у невеликих проєктах, так і в складних багаторівневих системах, що передбачають розмежування відповідальності між логікою обробки даних, рівнем представлення та збереження інформації. Така універсальність забезпечує можливість поступового розвитку програмного продукту без необхідності повної зміни технологічної основи [15]-[16].

Окремим аргументом є наявність великої професійної спільноти та значного обсягу документації, що полегшує процес розробки, тестування та

супроводу програмного забезпечення. Доступність навчальних матеріалів і практичних напрацювань сприяє швидшому розв’язанню технічних проблем і підвищенню якості кінцевого продукту. У наукових і навчальних роботах цей аспект може бути розглянутий як чинник, що знижує ризики реалізації проєкту [15]-[16].

Доцільність вибору мови програмування PHP для створення системи опрацювання даних для керування розкладом зумовлюється сукупністю технічних, архітектурних та експлуатаційних чинників, які відповідають вимогам сучасних веборієнтованих інформаційних систем. Оскільки подібні системи, як правило, реалізуються у вигляді вебзастосунків із централізованим доступом для різних категорій користувачів, використання PHP як серверної мови програмування є обґрунтованим з огляду на її первинну орієнтацію на веброзробку та обробку запитів у мережевому середовищі [15]-[16].

PHP забезпечує ефективну роботу з даними, що є ключовим аспектом у системах керування розкладом. Мова має розвинені засоби взаємодії з системами керування базами даних, що дозволяє реалізувати надійне збереження, обробку та оновлення інформації про часові інтервали, ресурси, обмеження та взаємозв’язки між елементами розкладу. Це сприяє побудові логічно цілісної моделі даних та забезпечує швидке виконання типових операцій, пов’язаних із формуванням і коригуванням розкладу [15]-[16].

Важливим аргументом на користь використання PHP є її поширеність і стабільність у серверному середовищі. Широка підтримка з боку хостинг-провайдерів і сумісність з популярними вебсерверами спрощують розгортання та подальшу експлуатацію системи без необхідності використання складної або дороговартісної інфраструктури. Це особливо актуально для інформаційних систем, які передбачають тривалу експлуатацію та можливість масштабування [15]-[16].

Крім того, PHP характеризується наявністю зрілої екосистеми фреймворків і бібліотек, що дозволяє структурувати програмний код, підвищити його підтримуваність і спростити реалізацію бізнес-логіки.

Використання таких засобів сприяє підвищенню якості програмного забезпечення, забезпеченню безпеки доступу до даних і реалізації механізмів автентифікації та авторизації, які є важливими для систем керування розкладом із багатокористувацьким доступом [15]-[16].

Доцільність застосування PHP також визначається відносно низьким порогом входу та зрозумілою синтаксичною структурою мови, що полегшує підтримку та розвиток програмного продукту в майбутньому. Це створює умови для швидкої адаптації системи до змін вимог, розширення функціональних можливостей і модифікації алгоритмів опрацювання даних без суттєвих ризиків для стабільності роботи [15]-[16].

Обґрунтування вибору мови програмування для розробки системи опрацювання даних для керування розкладом наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для розробки системи опрацювання даних для керування розкладом

Критерій порівняння мов програмування	Мова програмування		
	PHP	Python	C#
Орієнтація на веброзробку	+	+–	+–
Простота розгортання на хостингу	+	+–	–
Швидкість розробки прикладних систем	+	+	+–
Підтримка типової серверної логіки	+	+–	+
Зручність супроводу в малих і середніх проєктах	+	+	+–

Таким чином, вибір мови програмування PHP для створення системи опрацювання даних для керування розкладом є доцільним з огляду на її

веборієнтованість, ефективні засоби роботи з даними, простоту впровадження, широку підтримку та можливості для подальшого розвитку програмного забезпечення [15], [16].

Отже, для реалізації системи опрацювання даних для керування розкладом обрано мову програмування PHP, яка поєднує природну веборієнтованість, простоту розгортання, ефективні засоби роботи з базами даних і відносно низькі вимоги до інфраструктури. Інші мови програмування демонструють високий потенціал у певних аспектах, однак потребують складнішого налаштування, більших ресурсів або є менш орієнтованими на класичні серверні вебзастосунки. Саме тому PHP може розглядатися як найбільш ефективний і практично доцільний вибір для реалізації подібних систем.

2.2 Вибір середовища розробки для створення системи опрацювання даних для керування розкладом

В якості середовища розробки для створення системи опрацювання даних для керування розкладом було обрано середовище NetBeans [17]-[18].

Середовище розробки NetBeans є інструментом створення прикладних програмних систем. Зокрема, NetBeans вирізняється високим рівнем інтеграції компонентів середовища, що забезпечує цілісність процесу розробки та зменшує потребу у використанні сторонніх програмних засобів. Така інтегрованість позитивно впливає на організацію роботи розробника та сприяє зниженню ймовірності помилок, пов’язаних із розбіжностями між окремими етапами розробки [17]-[18].

Середовище NetBeans підтримує концепцію модульності, яка дозволяє адаптувати середовище під конкретні потреби проєкту. Завдяки можливості підключення додаткових модулів і плагінів середовище може бути розширене функціональністю, необхідною для роботи з різними технологіями, фреймворками та інструментами. Це створює передумови для гнучкого

налаштування процесу розробки без втрати стабільності роботи середовища [17]-[18].

Значну роль відіграє орієнтація NetBeans на стандарти та усталені підходи до програмування. Підтримка поширених стандартів сприяє формуванню зрозумілої та уніфікованої структури програмних проєктів, що полегшує читання, аналіз і супровід вихідного коду. У контексті командної розробки це дозволяє забезпечити узгодженість стилю та логіки реалізації програмних компонентів [17]-[18].

Наявність вбудованих засобів відлагодження NetBeans дозволяє здійснювати детальний аналіз виконання програмного коду, виявляти логічні помилки та перевіряти коректність роботи окремих модулів системи. Це є важливим чинником для розробки систем опрацювання даних, де помилки можуть призводити до порушення цілісності інформації або некоректного формування результатів [17], [18].

Крім того, NetBeans характеризується зручною організацією користувацького інтерфейсу, що сприяє підвищенню продуктивності праці розробника. Логічне розміщення інструментів, можливість налаштування робочого простору та швидкий доступ до основних функцій дозволяють зосередитися на реалізації логіки системи, а не на технічних аспектах роботи із середовищем [17]-[18].

Отже, NetBeans є не лише інструментом написання коду, а й комплексний засобом підтримки повного життєвого циклу створення програмного забезпечення, що є важливим для реалізації складних систем опрацювання даних [17]-[18].

Доцільність вибору середовища розробки NetBeans для створення системи опрацювання даних для керування розкладом обумовлюється його функціональною повнотою, універсальністю та орієнтацією на підтримку повного циклу розробки програмного забезпечення. NetBeans надає інтегроване середовище, у межах якого забезпечується розробка, налагодження, тестування та супровід програмних рішень, що є особливо важливим для інформаційних

систем з розвиненою серверною логікою та складною структурою опрацювання даних [17]-[18].

Суттєвою перевагою NetBeans є глибока підтримка мов програмування, що широко використовуються у веброзробці, зокрема PHP. Середовище забезпечує зручні засоби роботи з вихідним кодом, автоматичне доповнення, підсвічування синтаксису та статичний аналіз, що сприяє зменшенню кількості помилок і підвищенню якості програмної реалізації. Завдяки цьому розробка логіки керування розкладом та обробки даних стає більш структурованою і контрольованою [17]-[18].

Важливим аспектом є наявність у NetBeans інструментів для ефективної роботи з базами даних. Середовище дозволяє встановлювати з’єднання з різними системами керування базами даних, виконувати запити, переглядати структуру таблиць та аналізувати результати без необхідності використання зовнішніх засобів. Це є доцільним у процесі розробки системи керування розкладом, де значна частина функціональності пов’язана зі збереженням, оновленням та аналізом даних [17]-[18].

NetBeans також сприяє впорядкуванню проєктної структури та дотриманню єдиних підходів до організації коду. Використання стандартних механізмів керування проєктами полегшує масштабування системи та її подальшу модифікацію. Це особливо актуально для систем, що можуть розширюватися шляхом додавання нових функціональних модулів або зміни алгоритмів опрацювання даних [17]-[18].

Окремої уваги заслуговує можливість інтеграції середовища розробки NetBeans із засобами контролю версій, що підвищує надійність розробки та спрощує командну роботу. Контроль змін у вихідному коді сприяє зменшенню ризиків втрати даних і полегшує супровід програмного забезпечення протягом усього життєвого циклу. У контексті навчальних та прикладних проєктів це створює сприятливі умови для організованого й послідовного процесу розробки [17]-[18].

Обґрунтування вибору середовища розробки для створення системи опрацювання даних для керування розкладом наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору середовища розробки для створення системи опрацювання даних для керування розкладом

Критерій порівняння середовищ розробки	Середовища розробки		
	NetBeans	PhpStorm	Eclipse
Повнота інтегрованого середовища	+	+	+–
Зручність роботи з базами даних	+	+–	+–
Вбудовані засоби тестування	+	+	+–
Продуктивність середовища	+	+–	+–
Підтримка широкого спектра фреймворків PHP	+	+	+–

Таким чином, вибір середовища розробки NetBeans для створення системи опрацювання даних для керування розкладом є доцільним завдяки поєднанню зручних інструментів роботи з кодом, підтримки баз даних, впорядкованості проєктної структури та можливостей супроводу програмного продукту, що відповідає вимогам до сучасних інформаційних систем [17]-[18].

2.3 Вибір засобів зберігання та опрацювання даних

Для зберігання та опрацювання даних було обрано систему управління базами даних MySQL [19]-[20].

Для створення системи опрацювання даних для керування розкладом на мові PHP доцільно використати реляційну систему управління базами даних

(СУБД) MySQL. Обґрунтування цього вибору ґрунтується на технічних, архітектурних та практичних аспектах, що забезпечують ефективну роботу PHP-застосунків [19]-[20].

По-перше, MySQL добре інтегруються з мовою програмування PHP. Існують нативні розширення (mysqli), які дозволяють зручно виконувати запити, обробляти результати та реалізовувати транзакції. Це забезпечує надійне та швидке зберігання даних про розклад, ресурси, обмеження та взаємозв’язки між ними [19]-[20].

По-друге, реляційна модель даних дозволяє формалізувати логіку розкладу через таблиці, зв’язки між ними та обмеження цілісності. Це особливо важливо для систем, де критичною є точність і узгодженість даних, наприклад, у випадках одночасного доступу кількох користувачів, коригування розкладу чи контролю ресурсів [19]-[20].

По-третє, MySQL відома високою продуктивністю, стабільністю та широкою підтримкою на різних серверах і хостингах. Вони дозволяють реалізовувати масштабовані рішення, які можуть обробляти значні обсяги даних і підтримувати багатокористувацький режим без суттєвих витрат на інфраструктуру [19]-[20].

Додатковою перевагою є велика кількість документації, прикладів інтеграції з PHP та активна спільнота розробників. Це дозволяє швидко вирішувати проблеми, оптимізувати запити та застосовувати передові практики при побудові системи [19]-[20].

Отже, для розробки системи опрацювання даних для керування розкладом на PHP MySQL є найбільш доцільними СУБД, оскільки поєднують простоту інтеграції, надійність, високу продуктивність та підтримку реляційної структури даних, що забезпечує ефективне функціонування веборієнтованої інформаційної системи [19]-[20].

Обґрунтування вибору засобів зберігання та опрацювання даних наведено у таблиці 2.3.

Таблиця 2.3 – Обґрунтування вибору засобів зберігання та опрацювання даних

Критерій порівняння систем управління базами даних	Системи управління базами даних		
	MySQL	PostgreSQL	SQLite
Інтеграція з PHP	+	+–	+
Простота налаштування та розгортання	+	+–	+
Продуктивність для вебзастосунків	+	+–	+–
Масштабованість для багатокористувацьких систем	+	+	–
Підтримка транзакцій і цілісності даних	+	+	+–

Таким чином, для зберігання та опрацювання даних обрано систему управління базами даних MySQL, яка демонструє оптимальне поєднання простоти інтеграції з PHP, високої продуктивності для вебзастосунків, масштабованості та надійності. MySQL дозволяє ефективно обробляти багатокористувацькі запити, легко налаштовується на різних хостингах і має великий обсяг прикладів та документації для розробників PHP.

2.4 Висновки за розділом 2

Для реалізації системи опрацювання даних для керування розкладом обрано мову програмування PHP, яка поєднує природну веборієнтованість, простоту розгортання, ефективні засоби роботи з базами даних і відносно низькі вимоги до інфраструктури. Інші мови програмування демонструють високий потенціал у певних аспектах, однак потребують складнішого налаштування,

більших ресурсів або є менш орієнтованими на класичні серверні вебзастосунки. Саме тому PHP може розглядатися як найбільш ефективний і практично доцільний вибір для реалізації подібних систем.

Для створення системи опрацювання даних для керування розкладом обрано середовище розробки NetBeans завдяки поєднанню зручних інструментів роботи з кодом, підтримки баз даних, впорядкованості проєктної структури та можливостей супроводу програмного продукту, що відповідає вимогам до сучасних інформаційних систем.

Для зберігання та опрацювання даних обрано систему управління базами даних MySQL, яка демонструє оптимальне поєднання простоти інтеграції з PHP, високої продуктивності для вебзастосунків, масштабованості та надійності. MySQL дозволяє ефективно обробляти багатокористувацькі запити, легко налаштовується на різних хостингах і має великий обсяг прикладів та документації для розробників PHP.

3 ОПИС СИСТЕМИ ОПРАЦЮВАННЯ ДАНИХ

3.1 Структура системи опрацювання даних для керування розкладом

Структуру системи опрацювання даних для керування розкладом наведено на рис. 3.1.

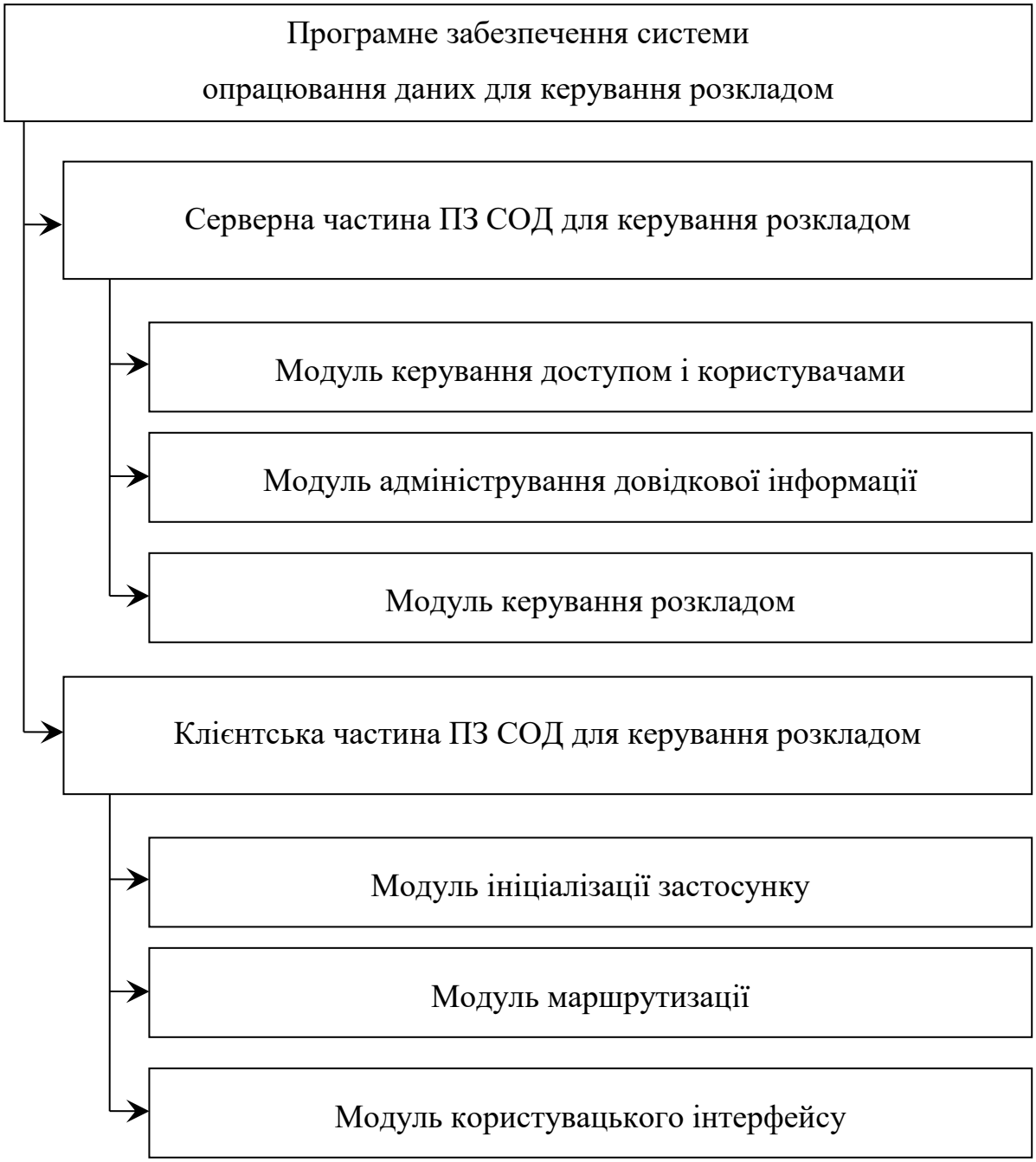

Рисунок 3.1 – Структура системи опрацювання даних для керування розкладом

Структура програмного забезпечення системи опрацювання даних для керування розкладом побудована за принципом чіткого розмежування серверної та клієнтської частин, що відповідає сучасним підходам до створення веборієнтованих інформаційних систем. Такий підхід забезпечує логічне відокремлення процесів обробки даних від процесів їх представлення, підвищує масштабованість системи та спрощує її супровід і розвиток.

Клієнтська частина системи призначена для взаємодії з користувачем і реалізує функції візуалізації розкладу, навігації, фільтрації та ініціювання запитів до серверної частини. Центральним елементом клієнтської частини є кореневий модуль, який виконує функцію ініціалізації застосунку, об’єднує всі компоненти інтерфейсу та забезпечує їх узгоджену роботу. Він відповідає за запуск клієнтської логіки, керування станом застосунку та взаємодію з маршрутизацією. Поруч із ним функціонує модуль маршрутизації, який визначає логіку переходів між сторінками системи, обробляє запити користувача до різних розділів та забезпечує коректне відображення відповідних компонентів інтерфейсу.

Важливою складовою клієнтської частини системи опрацювання даних для керування розкладом є модуль користувацького інтерфейсу, що охоплює основну сторінку, сторінки розкладу, елементи навігаційного меню та сторінки обробки помилок. Цей модуль забезпечує зручний доступ до функціональних можливостей системи, відображення розкладу дзвінків, навчальних занять і навантаження викладачів, а також інформування користувача у випадку виникнення помилкових або неіснуючих запитів. Доповненням до нього є модуль обробки та фільтрації даних, який реалізує функції пошуку, сортування та відбору інформації про розклад за заданими параметрами. Саме цей модуль забезпечує гнучку взаємодію користувача з великими обсягами даних без перевантаження інтерфейсу.

Серверна частина системи зосереджена на реалізації бізнес-логіки, управлінні даними та забезпеченні взаємодії з базою даних. Вона побудована у вигляді сукупності логічно пов’язаних модулів, кожен з яких відповідає за

окрему предметну область. Ключовим є модуль керування доступом і користувачами, який реалізує механізми реєстрації, ідентифікації та авторизації, а також забезпечує контроль прав доступу до функціональних можливостей системи. Цей модуль дозволяє розмежовувати повноваження між адміністраторами та звичайними користувачами, що є необхідним для захисту даних.

Окрему роль відіграє модуль адміністрування довідкової інформації, який забезпечує опрацювання даних про кафедри, підрозділи, аудиторії, навчальні групи та дисципліни. У межах цього модуля реалізуються функції додавання, редагування та видалення відповідних сутностей, що дозволяє підтримувати актуальність інформаційної бази системи. Така структуризація спрощує логіку управління даними та забезпечує цілісність інформації, яка використовується під час формування розкладу.

Центральним елементом серверної частини є модуль керування розкладом, який відповідає за обробку даних щодо навчальних занять, розкладу студентів і викладачів, а також за узгодження часових і ресурсних параметрів. У межах цього модуля реалізуються алгоритми формування, збереження та коригування розкладу, а також перевірка його на відповідність установленим обмеженням. Доповнює його модуль управління навчальними одиницями, який забезпечує роботу з класами, предметами та іншими структурними елементами навчального процесу, що безпосередньо впливають на побудову розкладу.

Таким чином, програмне забезпечення системи опрацювання даних для керування розкладом має чітко визначену модульну структуру, у якій клієнтська частина відповідає за взаємодію з користувачем і представлення інформації, а серверна частина — за обробку даних і реалізацію бізнес-логіки. Такий поділ забезпечує узгоджену роботу компонентів, підвищує надійність системи та створює умови для її подальшого розвитку і масштабування.

3.2 Функціонування та алгоритми системи опрацювання даних для керування розкладом

Функціонування системи опрацювання даних для керування розкладом подамо за допомогою схеми, зображеної на рис. 3.2.

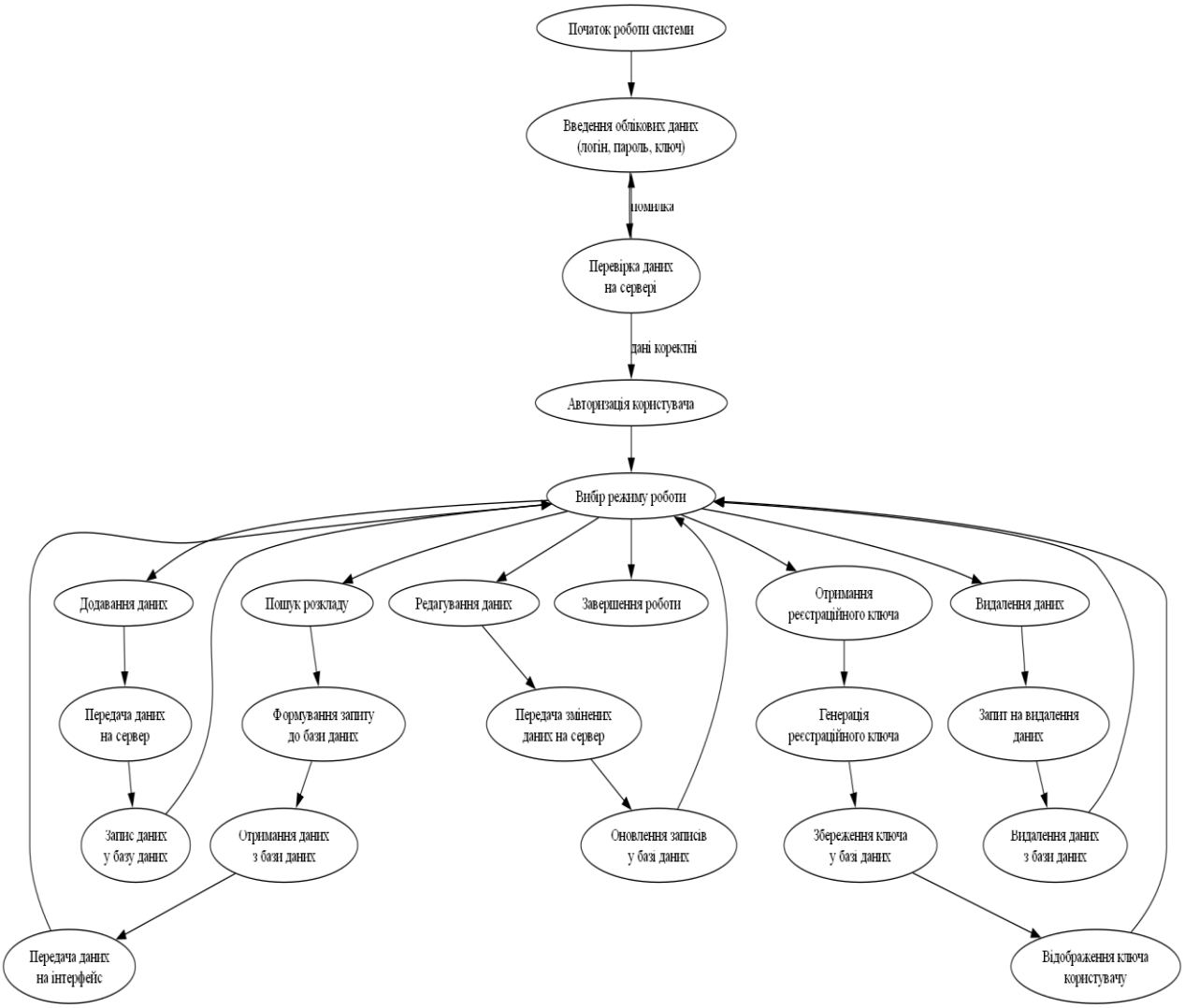

Рисунок 3.2 – Схема функціонування системи опрацювання даних для керування розкладом

Функціонування системи опрацювання даних для керування розкладом ґрунтується на послідовній взаємодії клієнтської та серверної частин і реалізується як впорядкований ланцюг дій, спрямованих на введення, обробку, збереження та відображення інформації. На початковому етапі роботи системи

ініціюється звернення користувача до вебзастосунку, у результаті чого здійснюється завантаження інтерфейсної частини та підготовка середовища для подальшої взаємодії. Після цього система переходить у режим очікування введення облікових даних або реєстраційного ключа, необхідного для ідентифікації користувача.

У процесі реєстрації або авторизації відбувається введення користувацьких даних, які передаються з клієнтської частини на сервер для перевірки коректності та відповідності наявним записам. Серверна частина здійснює обробку отриманої інформації, виконує звернення до бази даних і визначає можливість надання доступу. У разі успішного підтвердження даних формується відповідь, яка повертається на клієнтську сторону, після чого відбувається перехід до основного функціонального середовища системи.

Після проходження етапу автентифікації система забезпечує доступ до операцій керування даними розкладу. У випадку ініціювання пошуку відбувається формування запиту з параметрами, визначеними користувачем, який передається на серверну частину. Сервер виконує вибірку відповідних даних із бази даних на основі ідентифікаторів або інших умов, після чого результат обробки повертається до клієнтської частини для подальшого відображення у вигляді таблиць або структурованих блоків інформації.

У разі потреби внесення нових даних ініціюється процес додавання інформації, що супроводжується передаванням запиту на сервер. Серверна частина виконує перевірку отриманих даних, формує відповідний запит до бази даних та здійснює їх збереження. Після успішного завершення операції формується відповідь, яка містить оновлену інформацію та підтвердження виконання дії, що дозволяє клієнтській частині актуалізувати відображення даних.

Процес редагування реалізується шляхом отримання ідентифікатора відповідного запису та передавання змінених значень на серверну сторону. Сервер виконує оновлення відповідного запису в базі даних, забезпечуючи збереження цілісності інформації. Після завершення цієї операції система

повертає результат обробки на клієнтську частину, що дає змогу негайно відобразити внесені зміни у користувацькому інтерфейсі.

Функціонування алгоритму видалення даних відбувається за схожою логікою, коли з клієнтської частини передається запит із зазначенням ідентифікатора запису, що підлягає вилученню. Серверна частина здійснює відповідну операцію видалення з бази даних та формує відповідь про завершення процесу, після чого інтерфейсна частина оновлює відображення даних відповідно до поточного стану системи.

Окремим етапом функціонування системи є отримання реєстраційного ключа, який генерується або обробляється серверною частиною на основі відповідного запиту. Після збереження необхідної інформації в базі даних сформований результат передається на клієнтську сторону та відображається користувачеві. Завершальним етапом кожного циклу взаємодії є повернення системи в стан очікування наступної дії, що забезпечує безперервну та послідовну роботу всієї системи опрацювання даних для керування розкладом.

3.3 Розробка бази даних системи опрацювання даних для керування розкладом

База даних системи опрацювання даних для керування розкладом спроєктована як цілісна реляційна структура, призначена для зберігання, оброблення та узгодженого використання інформації, пов’язаної з навчальним процесом, користувачами та часовими параметрами занять. Логічна модель такої бази даних орієнтована на мінімізацію надлишковості даних, забезпечення цілісності зв’язків між об’єктами предметної області та підтримку основних функціональних можливостей системи, зокрема формування, перегляду й адміністрування розкладу.

Схему бази даних системи опрацювання даних для керування розкладом наведено на рис. 3.3.

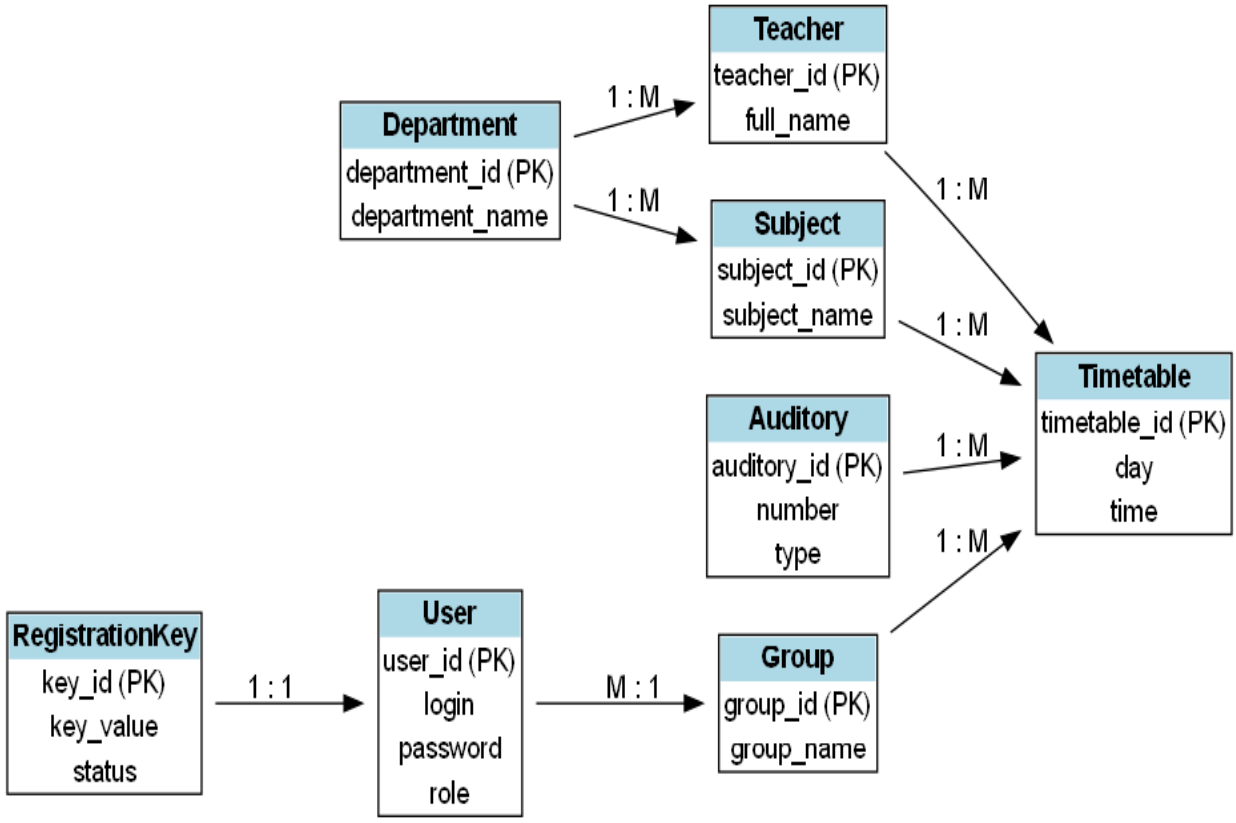

Рисунок 3.3 – Схема бази даних системи опрацювання даних для керування розкладом

Центральним елементом логічної моделі є сутність, що відповідає за збереження інформації про користувачів системи. У відповідній таблиці фіксуються облікові дані, роль користувача та службова інформація, необхідна для реалізації механізмів авторизації й розмежування доступу. Дана таблиця логічно пов’язана з іншими сутностями системи, оскільки саме користувач ініціює операції додавання, редагування або перегляду розкладу, а також може бути пов’язаний із певною навчальною групою чи викладацькою діяльністю.

Ключове місце у структурі бази даних займає таблиця, що зберігає безпосередньо дані розкладу. У ній акумулюється інформація про навчальні заняття з урахуванням часу проведення, навчальної дисципліни, аудиторії, групи та викладача. Ця сутність виконує роль інтеграційної, оскільки через зовнішні ключі вона пов’язується з таблицями, що описують навчальні дисципліни, викладачів, навчальні групи та аудиторії. Такий підхід дозволяє

формувати розклад на основі вже наявних довідкових даних і забезпечує логічну узгодженість усіх елементів.

Окремі таблиці логічної моделі призначені для зберігання довідкової інформації. До них належать сутності, що описують навчальні дисципліни, кафедри або підрозділи, навчальні групи та аудиторії. Між цими таблицями встановлюються зв’язки типу один до багатьох, що відображає реальні залежності предметної області, наприклад, належність дисципліни до певного підрозділу або закріплення групи за визначеним напрямом підготовки. Аудиторії, у свою чергу, пов’язані з розкладом таким чином, щоб унеможливити накладання занять у межах одного часу.

Логічна модель також передбачає наявність таблиці, призначеної для збереження реєстраційних ключів, які використовуються під час створення нових облікових записів або надання розширених прав доступу. Зв’язок між цією таблицею та таблицею користувачів забезпечує контроль процесу реєстрації та дозволяє реалізувати механізм обмеженого доступу до адміністративних функцій системи.

Загалом логічна модель бази даних побудована таким чином, щоб кожна сутність відображала окремий об’єкт предметної області, а зв’язки між ними забезпечували узгоджене та несуперечливе зберігання інформації. Така організація бази даних створює передумови для ефективного виконання запитів, спрощує супровід системи та забезпечує можливість її подальшого масштабування без суттєвих змін у вже реалізованій структурі.

3.4 Проєктування інтерфейсу системи опрацювання даних для керування розкладом

Проєктування інтерфейсу взаємодії користувача з системою опрацювання даних для керування розкладом виконано з урахуванням вимог технічного завдання. При розробленні системи опрацювання даних для керування розкладом враховані функціональні вимоги до програми:

– підтримка можливості збереження та відображення даних про розклад занять у базі даних;

– підтримка можливості додавання, редагування та видалення інформації, пов’язаної з розкладом;

– забезпечення реєстрації та авторизації користувачів із використанням механізмів контролю доступу та розмежування прав відповідно до ролей;

– забезпечення збереження та оброблення довідкових даних, що стосуються факультетів, кафедр, навчальних груп, викладачів та іншої інформації;

– забезпечення коректної взаємодії між клієнтською та серверною частинами з використанням стандартизованих форматів обміну даними;

– підтримка можливості зміни паролів для входу у систему зареєстрованих користувачів.

Інтерфейс головної форми системи опрацювання даних для керування розкладом наведено на рис. 3.4.

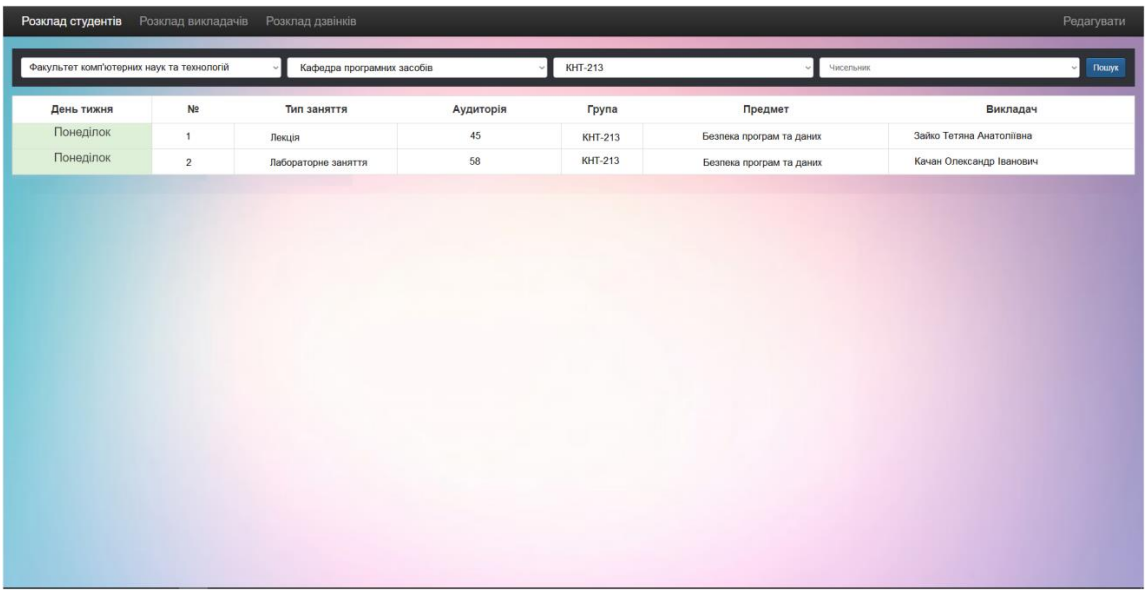

Рисунок 3.4 – Інтерфейс головної форми системи опрацювання даних для керування розкладом

Інтерфейс форми авторизації наведено на рис. 3.5.

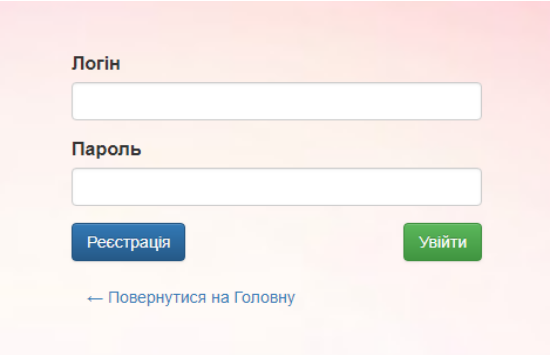

Рисунок 3.5 – Інтерфейс форми авторизації

Інтерфейс прототипу сторінки адміністратора наведено на рис. 3.6.

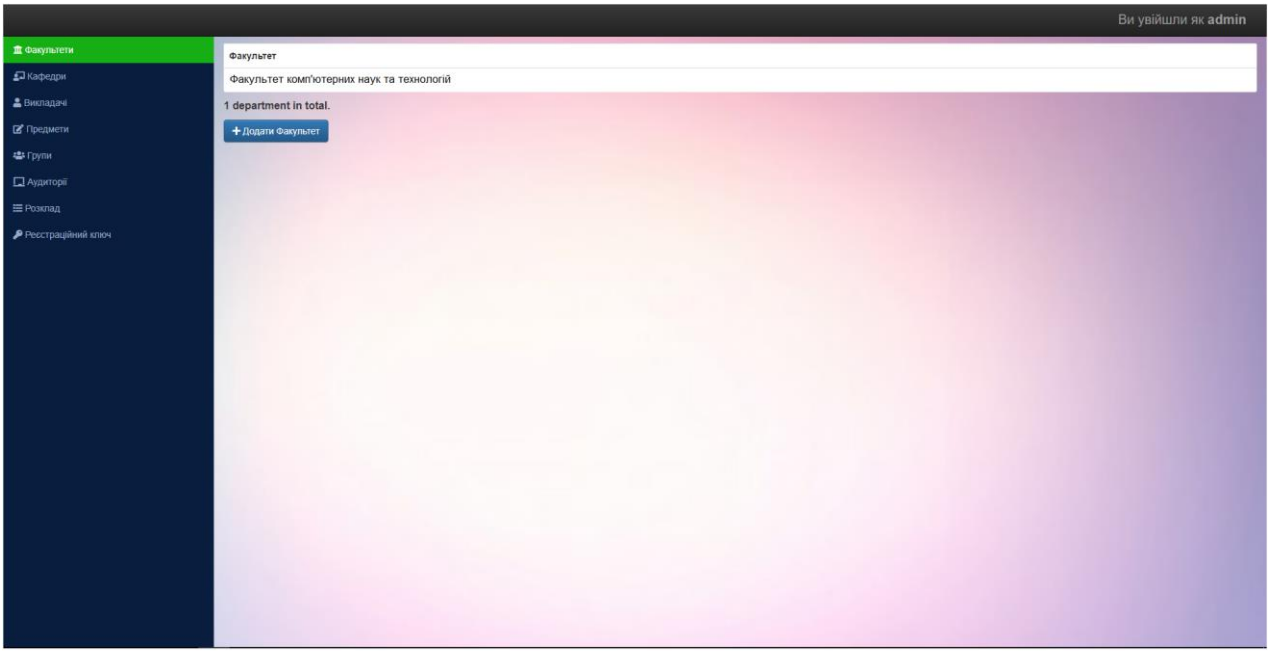

Рисунок 3.6 – Інтерфейс прототипу сторінки адміністратора

3.5 Висновки за розділом 3

Розроблено систему опрацювання даних для керування розкладом.

Визначено, що структура програмного забезпечення системи опрацювання даних для керування розкладом побудована за принципом чіткого розмежування серверної та клієнтської частин, що відповідає сучасним підходам до створення веборієнтованих інформаційних систем. Такий підхід забезпечує логічне відокремлення процесів обробки даних від процесів їх

представлення, підвищує масштабованість системи та спрощує її супровід і розвиток.

У підрозділі 3.2 було розглянуто принципи та логіку функціонування системи опрацювання даних для керування розкладом, а також послідовність взаємодії між клієнтською та серверною частинами вебзастосунку. Описані алгоритми охоплюють основні процеси роботи системи, зокрема реєстрацію та авторизацію користувачів, пошук розкладу, додавання, редагування й видалення даних, а також отримання реєстраційного ключа. Це дозволяє забезпечити цілісність даних, контроль доступу та коректну обробку інформації на всіх етапах роботи системи.

Реалізована логіка функціонування демонструє послідовний та структурований підхід до опрацювання запитів користувача, при якому всі основні операції виконуються на серверній стороні з використанням бази даних, а результати передаються на інтерфейсну частину для відображення. Така організація роботи підвищує надійність, безпеку та масштабованість системи, а також створює передумови для її подальшого розширення та адаптації до нових вимог.

Розроблено базу даних системи опрацювання даних для керування розкладом, яка спроєктована як цілісна реляційна структура, призначена для зберігання, оброблення та узгодженого використання інформації, пов’язаної з навчальним процесом, користувачами та часовими параметрами занять. Логічна модель бази даних орієнтована на мінімізацію надлишковості даних, забезпечення цілісності зв’язків між об’єктами предметної області та підтримку основних функціональних можливостей системи, зокрема формування, перегляду й адміністрування розкладу.

Виконано проєктування інтерфейсу взаємодії користувача з системою опрацювання даних для керування розкладом.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ СИСТЕМИ ОПРАЦЮВАННЯ ДАНИХ

4.1 Призначення й умови застосування системи опрацювання даних

Система опрацювання даних призначена для підтримки процесу керування розкладом. Система опрацювання даних написана на мові PHP, яка поєднує природну веборієнтованість, простоту розгортання, ефективні засоби роботи з базами даних і відносно низькі вимоги до інфраструктури. В якості середовища розробки для створення системи опрацювання даних для керування розкладом обрано NetBeans завдяки поєднанню зручних інструментів роботи з кодом, підтримки баз даних, впорядкованості проєктної структури та можливостей супроводу програмного продукту, що відповідає вимогам до сучасних інформаційних систем.

Система опрацювання даних для керування розкладом забезпечує виконання таких функцій:

– підтримка можливості збереження та відображення даних про розклад занять у базі даних;

– підтримка можливості додавання, редагування та видалення інформації, пов’язаної з розкладом;

– забезпечення реєстрації та авторизації користувачів із використанням механізмів контролю доступу та розмежування прав відповідно до ролей;

– забезпечення збереження та оброблення довідкових даних, що стосуються факультетів, кафедр, навчальних груп, викладачів та іншої інформації;

– забезпечення коректної взаємодії між клієнтською та серверною частинами з використанням стандартизованих форматів обміну даними;

– підтримка можливості зміни паролів для входу у систему зареєстрованих користувачів.

Для роботи системи опрацювання даних для керування розкладом потрібні такі технічні та програмні ресурси: апаратне забезпечення, яке забезпечує стабільне виконання серверних і клієнтських операцій, а також відповідне програмне середовище для коректного функціонування вебзастосунку. З боку апаратного забезпечення доцільним є використання персонального комп’ютера або сервера з процесором із тактовою частотою не нижче базового рівня, достатнього для оброблення запитів користувачів у режимі реального часу. Обсяг оперативної пам’яті має бути не менше мінімального значення, необхідного для одночасної роботи вебсервера, системи керування базами даних та клієнтського браузера без зниження продуктивності. Для зберігання програмних файлів, бази даних і службової інформації потрібен накопичувач із вільним дисковим простором, що забезпечує резерв для подальшого розширення системи. Наявність стабільного мережевого з’єднання є обов’язковою умовою, оскільки система орієнтована на роботу в клієнт-серверному середовищі.

Програмне забезпечення для функціонування системи опрацювання даних для керування розкладом передбачає наявність операційної системи, що підтримує роботу вебсерверів і сучасних мережевих технологій. На серверній стороні необхідне встановлення вебсервера з підтримкою мови програмування PHP відповідної версії, а також системи управління базами даних, яка забезпечує збереження та оброблення структурованої інформації розкладу. Для коректної взаємодії між компонентами системи мають бути налаштовані відповідні програмні бібліотеки та розширення PHP, зокрема для роботи з базами даних і обміну даними у форматі JSON. На клієнтській стороні достатньо використання сучасного веббраузера, який підтримує стандартні вебтехнології та забезпечує коректне відображення інтерфейсу користувача. Зазначені вимоги дозволяють забезпечити стабільну, надійну та ефективну роботу системи опрацювання даних для керування розкладом у звичайних умовах експлуатації, а також створюють передумови для її масштабування та

подальшого функціонального розвитку без необхідності суттєвої модернізації апаратної чи програмної платформи.

Функціональні характеристики розробленої системи опрацювання даних для керування розкладом наведено у технічному завданні (додаток А). Фрагмент тексту програмного коду системи опрацювання даних для керування розкладом наведено у додатку Б.

4.2 Характеристики системи опрацювання даних для керування розкладом

Система опрацювання даних для керування розкладом характеризується як інтегрований програмний продукт, орієнтований на централізоване зберігання, оброблення та представлення інформації, пов’язаної з організацією навчального процесу. Її функціонування ґрунтується на клієнт-серверній взаємодії, що забезпечує відокремлення інтерфейсної частини від логіки оброблення даних та дозволяє досягти узгодженості, керованості й надійності під час виконання основних операцій. Архітектура системи сприяє чіткому розмежуванню відповідальності між окремими компонентами, завдяки чому підвищується зручність супроводу та можливість подальшого розширення функціональних можливостей.

Важливою характеристикою системи опрацювання даних для керування розкладом є орієнтація на роботу з структурованими даними, що зберігаються у реляційній базі даних та пов’язані між собою логічними залежностями. Такий підхід дозволяє забезпечити цілісність інформації про розклад, навчальні групи, викладачів, дисципліни та аудиторії, а також унеможливити появу суперечливих або дубльованих записів. Система опрацювання даних для керування розкладом підтримує оперативне оновлення даних, що дає змогу швидко вносити зміни та одразу відображати їх у користувацькому інтерфейсі без необхідності додаткових перетворень.

Характерною рисою системи опрацювання даних для керування розкладом є наявність механізмів контролю доступу, які забезпечують безпечну роботу з даними та диференціацію прав користувачів залежно від їхньої ролі. Це сприяє захисту службової інформації та зменшує ризик несанкціонованого втручання у процес формування або редагування розкладу. Водночас інтерфейс системи орієнтований на інтуїтивну взаємодію, що полегшує використання програмного забезпечення без потреби у спеціальній підготовці.

Система опрацювання даних для керування розкладом також вирізняється адаптивністю до змін умов експлуатації, оскільки її логіка побудована з урахуванням можливості масштабування та інтеграції з іншими інформаційними ресурсами. Така характеристика дозволяє розглядати розроблений програмний продукт не лише як інструмент для вирішення поточних завдань керування розкладом, а і як основу для подальшого розвитку інформаційної інфраструктури навчального закладу або організації.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для запуску системи опрацювання даних для керування розкладом на серверній частини необхідно запустити відповідний файл.

4.3.2 Вхідні й вихідні дані

Вхідними даними до системи опрацювання даних для керування розкладом є інформація, що вводиться користувачем у процесі взаємодії з інтерфейсом програмного забезпечення та передається на серверну частину для подальшого опрацювання. До таких даних належать облікові відомості, необхідні для автентифікації та авторизації, параметри пошуку розкладу, а також відомості, що використовуються під час створення, редагування або коригування записів, пов’язаних з навчальними групами, викладачами,

дисциплінами, аудиторіями та часовими характеристиками занять.

Вихідними даними системи опрацювання даних для керування розкладом є результати оброблення запитів користувача, які формуються на серверній стороні та відображаються у клієнтському інтерфейсі. До таких даних належить сформований розклад занять відповідно до заданих параметрів пошуку, оновлені відомості після внесення змін до бази даних, а також службова інформація, що підтверджує успішне виконання операцій.

4.3.3 Повідомлення

Користувач системи опрацювання даних для керування розкладом може отримати такі повідомлення: повідомлення про успішну або неуспішну авторизацію, інформаційні повідомлення щодо результатів виконання операцій додавання, редагування чи видалення даних, а також попередження або повідомлення про помилки у разі некоректного введення інформації чи відсутності потрібних даних у базі.

4.4 Виконання системи опрацювання даних для керування розкладом

Після запуску системи опрацювання даних для керування розкладом відображається сторінка авторизації (рис. 4.1), де потрібно ввести логін і пароль та натиснути кнопку Увійти. Якщо користувач не зареєстрований, потрібно пройти процедуру реєстрації, натиснувши кнопку Реєстрація.

Після входу до системи користувачу відображається головна сторінка (рис. 4.2), де є можливість ознайомитися з розкладом занять для студентів та розкладом роботи викладачів.

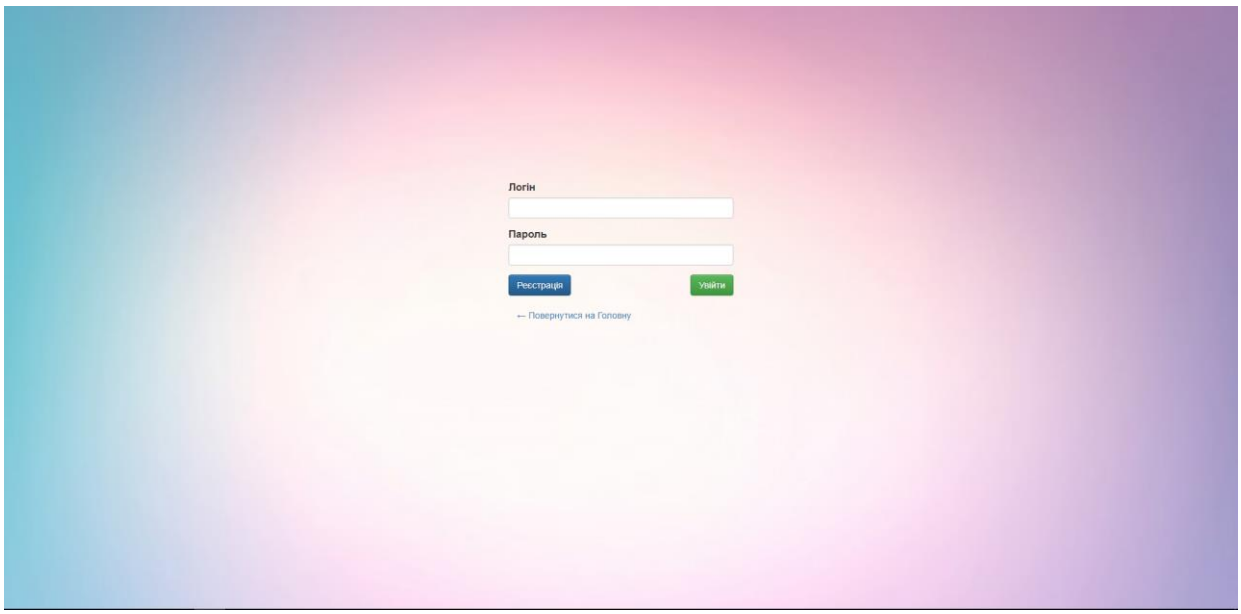

Рисунок 4.1 – Сторінка авторизації системи опрацювання даних для керування розкладом

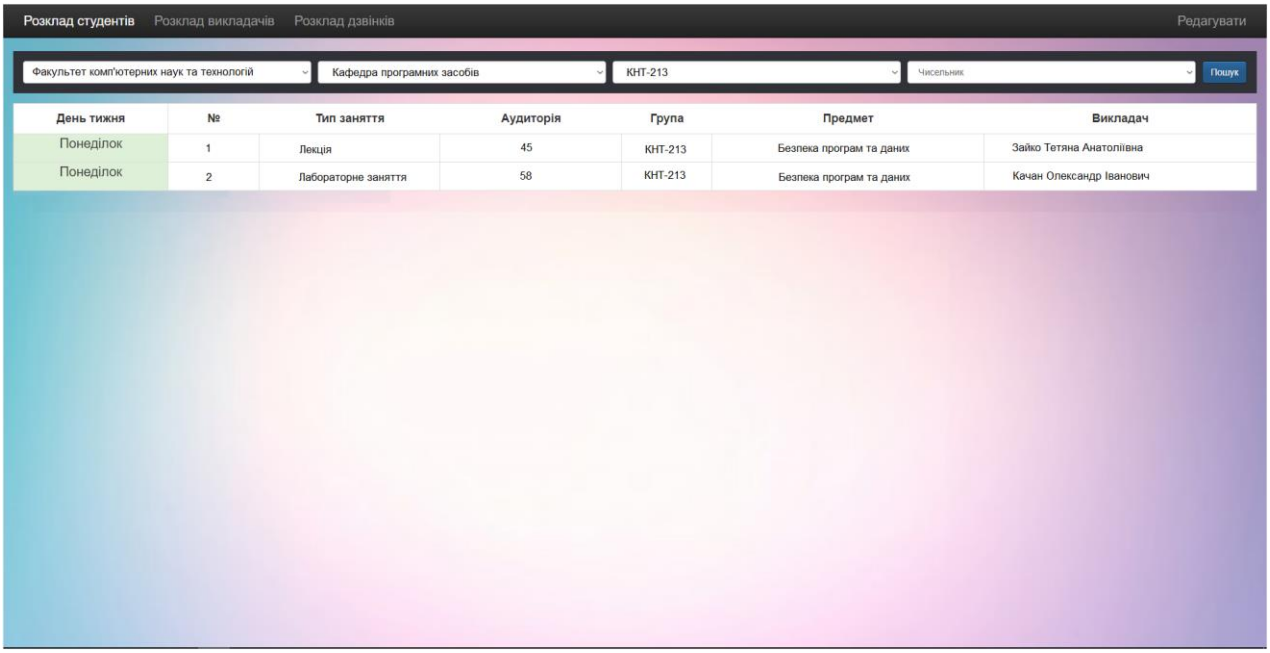

Рисунок 4.2 – Головна сторінка

Для ознайомлення з інформацією про початок та кінець проведення занять необхідно перейти у вкладку Розклад дзвінків (рис. 4.3).

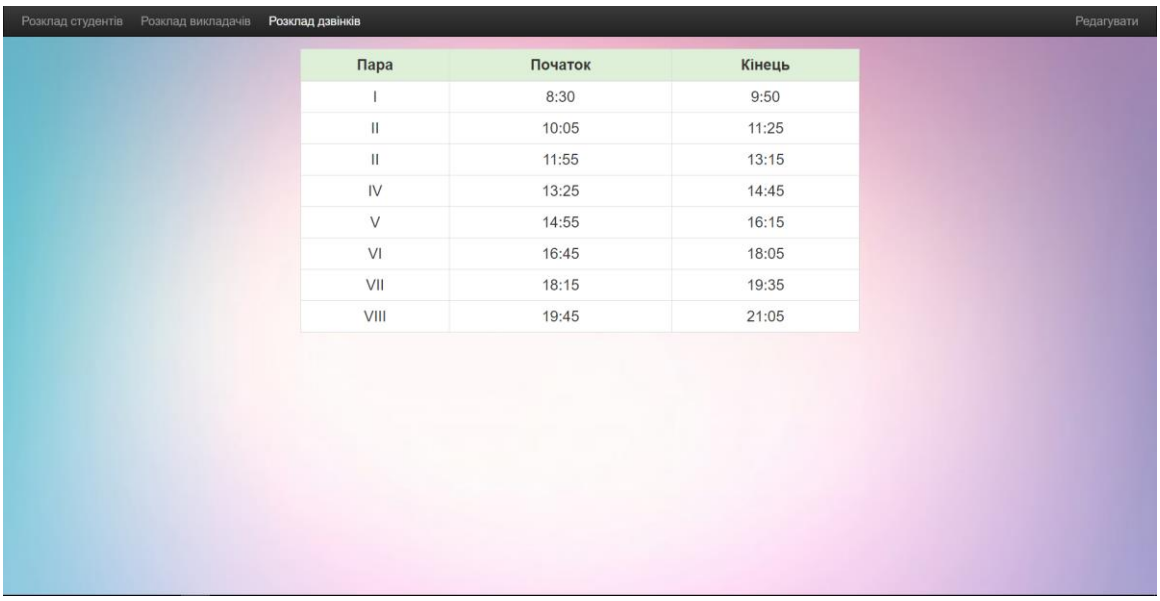

Рисунок 4.3 – Вкладка розкладу дзвінків

При вході до системи користувача з правами адміністратора відображається відповідна сторінка (рис. 4.4), де можна отримувати та редагувати інформацію про факультети, кафедри предмети, групи, розклад та ін.

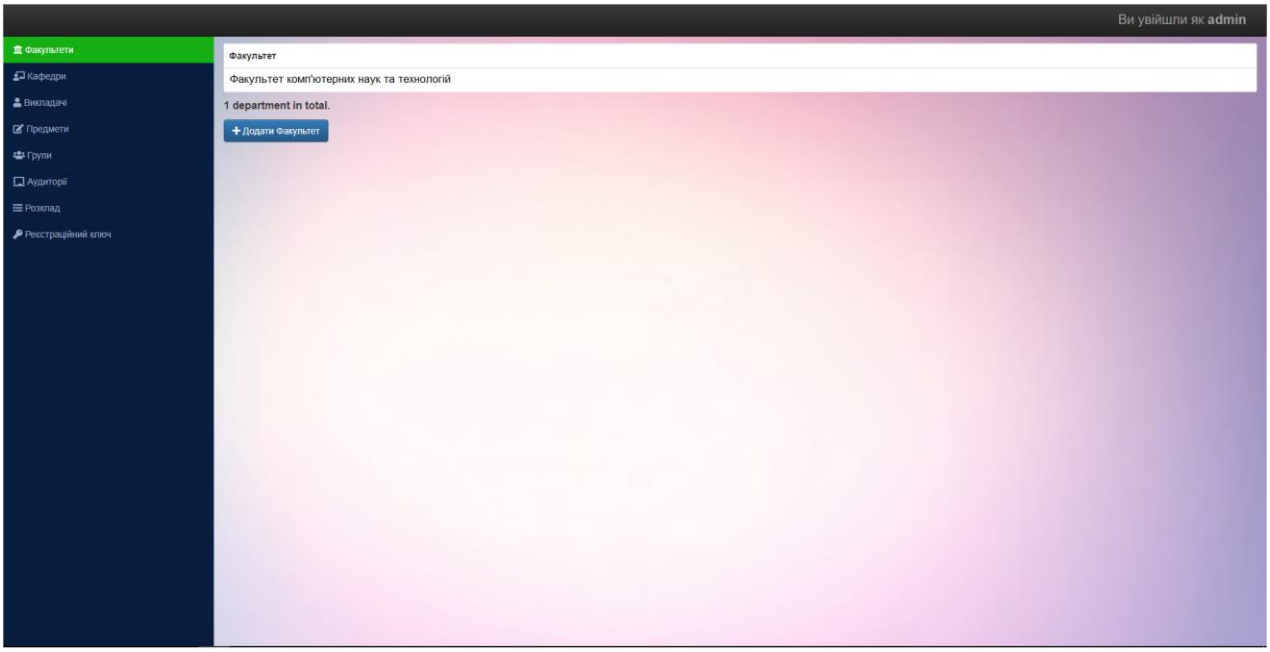

Рисунок 4.4 – Сторінка адміністратора

При переході на відповідні сторінки існує можливість редагування, додавання та видалення інформації. На рис. 4.5 наведено приклад додавання інформації про новий факультет закладу освіти.

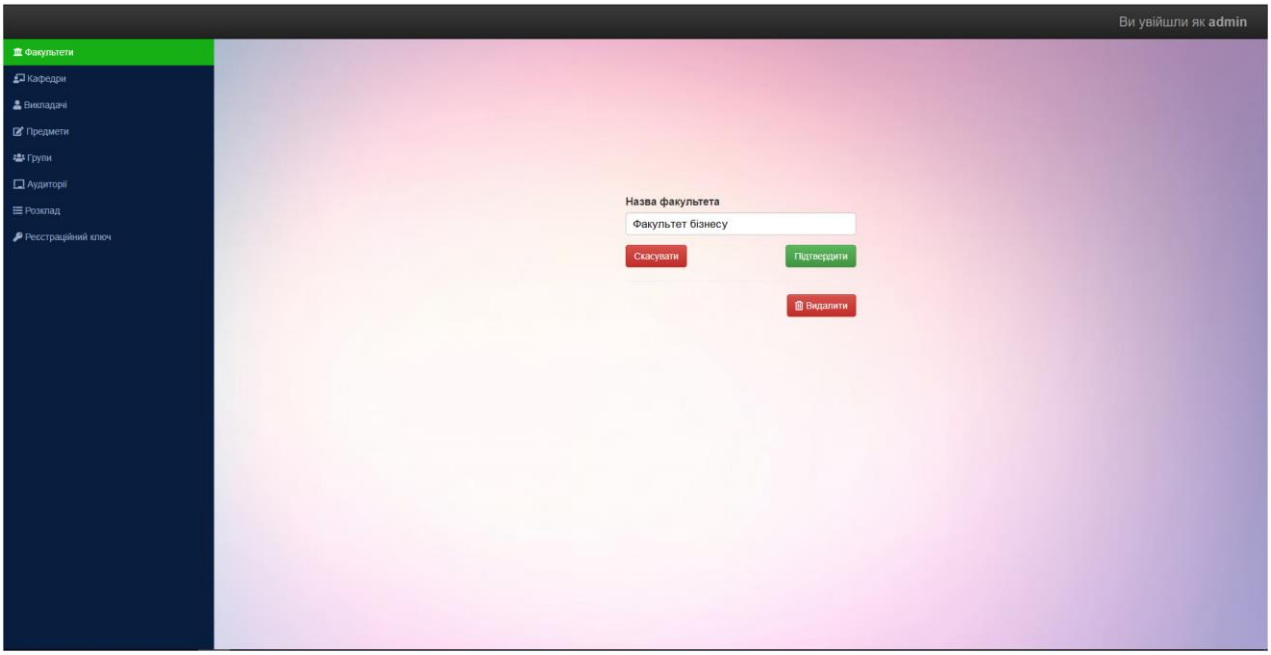

Рисунок 4.5 – Додавання факультету

На персональній сторінці користувача існує можливість зміни паролю (рис. 4.6) для входу у систему.

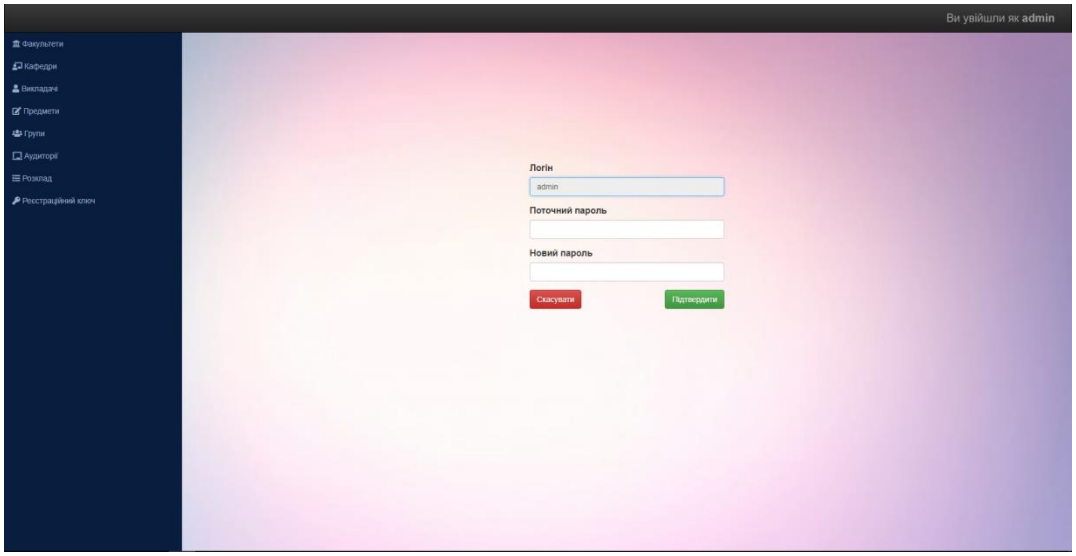

Рисунок 4.6 – Зміна авторизаційних даних

4.5 Тестування системи опрацювання даних для керування розкладом

Проведено тестування системи опрацювання даних для керування розкладом.

Підтверджено, що системи опрацювання даних для керування розкладом працює коректно і стабільно, виконуючи всі необхідні функціональні вимоги та успішно справляючись із основною задачею підтримки процесу керування розкладом.

Розроблена система забезпечує організацію та опрацювання даних, пов’язаних із підтримкою процесів керування розкладом.

4.6 Висновки за розділом 4

Описано систему опрацювання даних для керування розкладом.

Проведено тестування створеної системи опрацювання даних для керування розкладом. Результати тестування показали, що система забезпечує організацію та опрацювання даних, пов’язаних із підтримкою процесів керування розкладом.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено алгоритми та процеси обчислень, пов’язані з організацією та опрацюванням даних для керування розкладом.

Визначено, що система опрацювання даних для керування розкладом дозволяє виконувати впорядкування, зберігання та оброблення інформації, пов’язаної з плануванням подій, занять або робочих процесів у межах певної організації. Її основне призначення полягає у забезпеченні узгодженості між різними елементами розкладу, такими як час, ресурси, учасники та правила використання, шляхом централізованого управління відповідними даними.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато систем опрацювання даних для керування розкладом. Проте деякі системи опрацювання даних для керування розкладом мають обмежену здатність повноцінно враховувати складні, неформалізовані або суперечливі вимоги, що виникають у процесі керування розкладом. Це призводить до ситуацій, коли автоматично сформований розклад потребує додаткового ручного коригування, що знижує очікуваний ефект від використання програмного забезпечення. Ще одним недоліком є складність впровадження та налаштування програмних систем керування розкладом. Процес адаптації програмного забезпечення до конкретних умов організації може вимагати значних часових і ресурсних витрат, а також залучення фахівців з відповідною кваліфікацією. Крім того, обмеження користувацького інтерфейсу та недостатня зручність взаємодії з системою можуть негативно впливати на ефективність її застосування. Таким чином, програмні системи керування розкладом, попри їх потенціал щодо оптимізації планування, мають низку недоліків, пов’язаних з обмеженою адаптивністю, залежністю від якості даних, складністю впровадження та експлуатації. Тому актуальною є розробка системи опрацювання даних для керування розкладом.

Сформульовано функціональні вимоги до системи опрацювання даних для керування розкладом.

Для реалізації системи опрацювання даних для керування розкладом обрано мову програмування PHP, яка поєднує природну веборієнтованість, простоту розгортання, ефективні засоби роботи з базами даних і відносно низькі вимоги до інфраструктури. Інші мови програмування демонструють високий потенціал у певних аспектах, однак потребують складнішого налаштування, більших ресурсів або є менш орієнтованими на класичні серверні вебзастосунки. Саме тому PHP може розглядатися як найбільш ефективний і практично доцільний вибір для реалізації подібних систем.

Для створення системи опрацювання даних для керування розкладом обрано середовище розробки NetBeans завдяки поєднанню зручних інструментів роботи з кодом, підтримки баз даних, впорядкованості проєктної структури та можливостей супроводу програмного продукту, що відповідає вимогам до сучасних інформаційних систем.

Для зберігання та опрацювання даних обрано систему управління базами даних MySQL, яка демонструє оптимальне поєднання простоти інтеграції з PHP, високої продуктивності для вебзастосунків, масштабованості та надійності. MySQL дозволяє ефективно обробляти багатокористувацькі запити, легко налаштовується на різних хостингах і має великий обсяг прикладів та документації для розробників PHP.

Розроблено систему опрацювання даних для керування розкладом.

Визначено, що структура програмного забезпечення системи опрацювання даних для керування розкладом побудована за принципом чіткого розмежування серверної та клієнтської частин, що відповідає сучасним підходам до створення веборієнтованих інформаційних систем. Такий підхід забезпечує логічне відокремлення процесів обробки даних від процесів їх представлення, підвищує масштабованість системи та спрощує її супровід і розвиток.

У підрозділі 3.2 було розглянуто принципи та логіку функціонування системи опрацювання даних для керування розкладом, а також послідовність взаємодії між клієнтською та серверною частинами вебзастосунку. Описані алгоритми охоплюють основні процеси роботи системи, зокрема реєстрацію та авторизацію користувачів, пошук розкладу, додавання, редагування й видалення даних, а також отримання реєстраційного ключа. Це дозволяє забезпечити цілісність даних, контроль доступу та коректну обробку інформації на всіх етапах роботи системи.

Реалізована логіка функціонування демонструє послідовний та структурований підхід до опрацювання запитів користувача, при якому всі основні операції виконуються на серверній стороні з використанням бази даних, а результати передаються на інтерфейсну частину для відображення. Така організація роботи підвищує надійність, безпеку та масштабованість системи, а також створює передумови для її подальшого розширення та адаптації до нових вимог.

Розроблено базу даних системи опрацювання даних для керування розкладом, яка спроєктована як цілісна реляційна структура, призначена для зберігання, оброблення та узгодженого використання інформації, пов’язаної з навчальним процесом, користувачами та часовими параметрами занять. Логічна модель бази даних орієнтована на мінімізацію надлишковості даних, забезпечення цілісності зв’язків між об’єктами предметної області та підтримку основних функціональних можливостей системи, зокрема формування, перегляду й адміністрування розкладу.

Виконано проєктування інтерфейсу взаємодії користувача з системою опрацювання даних для керування розкладом.

Проведено тестування створеної системи опрацювання даних для керування розкладом. Результати тестування показали, що система забезпечує організацію та опрацювання даних, пов’язаних із підтримкою процесів керування розкладом.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. School Scheduling Software: Top Features and Benefits. URL: https://www.teachngo.com/blog/school_scheduling_software (date of access: 28.04.2026).

2. A Complete Guide to School Schedule Management Systems. URL: https://wpschoolpress.com/school-schedule-management-systems/ (date of access: 28.04.2026).

3. Top Rated School Schedule Software: Managing Complex Schedules with Ease. URL: https://www.usascheduler.com/blog/top-rated-school-schedule-software (date of access: 28.04.2026).

4. Making Optimum Use of Resources with School Management Software. URL: https://timly.com/en/school-management-software/ (date of access: 28.04.2026).

5. What Is Software for School Management? A Guide to School Software Tools and How They Are Used. URL: https://www.teachngo.com/blog/software_for_school_management (date of access: 28.04.2026).

6. Timetable Manager. URL: https://www.isams.com/platform/modules/timetable-manager/ (date of access: 28.04.2026).

7. Best Scheduling Software for Schools, Academics. URL: https://theninehertz.com/blog/scheduling-software-schools-academic (date of access: 28.04.2026).

8. Free and Open Source School Administration Software. URL: https://www.capterra.com/resources/top-free-school-administration-software/ (date of access: 28.04.2026).

9. Best School Management System Software. URL: https://www.guru99.com/school-management-software.html (date of access: 28.04.2026).

10. Top Open Source and Free School Management Software. URL: https://www.techjockey.com/blog/top-5-open-source-free-school-management-software (date of access: 28.04.2026).

11. Best School Management Software. URL: https://www.softwareadvice.com/school-management/ (date of access: 28.04.2026).

12. Coursedog. URL: https://www.coursedog.com/products/academic-course-scheduling-software (date of access: 28.04.2026).

13. Ad Astra. URL: https://www.aais.com/solutions/essential-scheduling (date of access: 28.04.2026).

14. Arlo. URL: https://www.arlo.co/training-course-scheduling-system (date of access: 28.04.2026).

15. PHP Introduction. URL: https://www.w3schools.com/php/php_intro.asp (date of access: 28.04.2026).

16. PHP The Right Way. URL: https://phptherightway.com/ (date of access: 28.04.2026).

17. Apache NetBeans. URL: https://netbeans.apache.org/front/main/index.html (date of access: 28.04.2026).

18. NetBeans. URL: https://dev.to/netbeans (date of access: 28.04.2026).

19. Why MySQL. URL: https://www.mysql.com/why-mysql/ (date of access: 28.04.2026).

20. MySQL Tutorial. URL: https://www.mysqltutorial.org/ (date of access: 28.04.2026).

ДОДАТОК А

Технічне завдання

Вступ

Система опрацювання даних може використовуватися для підтримки процесу керування розкладом.

А.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Створення програмної системи опрацювання даних для керування розкладом», затверджене наказом Національного університету «Запорізька політехніка» № 139 від 07 квітня 2026 р.

А.2 Призначення розробки

Система опрацювання даних призначена для забезпечення керування розкладом.

А.3 Основні вимоги до програми, що розробляється

А.3.1 Вимоги до функціональних характеристик

Система опрацювання даних для керування розкладом забезпечує виконання таких функцій:

– підтримка можливості збереження та відображення даних про розклад занять у базі даних;

– підтримка можливості додавання, редагування та видалення інформації, пов’язаної з розкладом;

– забезпечення реєстрації та авторизації користувачів із використанням механізмів контролю доступу та розмежування прав відповідно до ролей;

– забезпечення збереження та оброблення довідкових даних, що стосуються факультетів, кафедр, навчальних груп, викладачів та іншої інформації;

– забезпечення коректної взаємодії між клієнтською та серверною частинами з використанням стандартизованих форматів обміну даними;

– підтримка можливості зміни паролів для входу у систему зареєстрованих користувачів.

А.3.2 Вимоги до інтерфейсу програми

Інтерфейс системи опрацювання даних для керування розкладом повинен бути зручним для користувачів. Повинна бути забезпечена можливість роботи в візуальному режимі.

А.3.3 Вимоги до надійності

Система опрацювання даних для керування розкладом повинна забезпечити надійне функціонування.

А.3.4 Умови експлуатації

Для експлуатації системи опрацювання даних для керування розкладом необхідна наявність персонального комп’ютера.

А.3.5 Вимоги до складу та параметрів технічний засобів

Для роботи системи опрацювання даних для керування розкладом потрібні такі технічні та програмні ресурси: апаратне забезпечення, яке

забезпечує стабільне виконання серверних і клієнтських операцій, а також відповідне програмне середовище для коректного функціонування вебзастосунку. З боку апаратного забезпечення доцільним є використання персонального комп’ютера або сервера з процесором із тактовою частотою не нижче базового рівня, достатнього для оброблення запитів користувачів у режимі реального часу. Обсяг оперативної пам’яті має бути не менше мінімального значення, необхідного для одночасної роботи вебсервера, системи керування базами даних та клієнтського браузера без зниження продуктивності. Для зберігання програмних файлів, бази даних і службової інформації потрібен накопичувач із вільним дисковим простором, що забезпечує резерв для подальшого розширення системи. Наявність стабільного мережевого з’єднання є обов’язковою умовою, оскільки система орієнтована на роботу в клієнт-серверному середовищі.

Програмне забезпечення для функціонування системи опрацювання даних для керування розкладом передбачає наявність операційної системи, що підтримує роботу вебсерверів і сучасних мережевих технологій. На серверній стороні необхідне встановлення вебсервера з підтримкою мови програмування PHP відповідної версії, а також системи управління базами даних, яка забезпечує збереження та оброблення структурованої інформації розкладу. Для коректної взаємодії між компонентами системи мають бути налаштовані відповідні програмні бібліотеки та розширення PHP, зокрема для роботи з базами даних і обміну даними у форматі JSON. На клієнтській стороні достатньо використання сучасного веббраузера, який підтримує стандартні вебтехнології та забезпечує коректне відображення інтерфейсу користувача. Зазначені вимоги дозволяють забезпечити стабільну, надійну та ефективну роботу системи опрацювання даних для керування розкладом у звичайних умовах експлуатації, а також створюють передумови для її масштабування та подальшого функціонального розвитку без необхідності суттєвої модернізації апаратної чи програмної платформи.

А.3.6 Вимоги до маркування і пакування

Система опрацювання даних для керування розкладом може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва – «Система опрацювання даних для керування розкладом».

А.4 Вхідні дані до роботи

Вхідними даними до системи опрацювання даних для керування розкладом є інформація, що вводиться користувачем у процесі взаємодії з інтерфейсом програмного забезпечення та передається на серверну частину для подальшого опрацювання. До таких даних належать облікові відомості, необхідні для автентифікації та авторизації, параметри пошуку розкладу, а також відомості, що використовуються під час створення, редагування або коригування записів, пов’язаних з навчальними групами, викладачами, дисциплінами, аудиторіями та часовими характеристиками занять.

Вихідними даними системи опрацювання даних для керування розкладом є результати оброблення запитів користувача, які формуються на серверній стороні та відображаються у клієнтському інтерфейсі. До таких даних належить сформований розклад занять відповідно до заданих параметрів пошуку, оновлені відомості після внесення змін до бази даних, а також службова інформація, що підтверджує успішне виконання операцій.

ДОДАТОК Б

Фрагмент тексту програми

```
class CnfgMdl
{
    public static function gtDbHst()
    {
        return "localhost";
    }

    public static function gtDbNm()
    {
        return "schdl_db";
    }

    public static function gtDbUsr()
    {
        return "usr_schdl";
    }

    public static function gtDbPss()
    {
        return "pss_schdl";
    }
}

class DbCnntMdl
{
    private $cnnt;

    public function __construct()
    {
        $hst = CnfgMdl::gtDbHst();
        $nm = CnfgMdl::gtDbNm();
        $sr = CnfgMdl::gtDbUsr();
        $ps = CnfgMdl::gtDbPss();
        $this->cnnt = new PDO("mysql:host=".$hst.";dbname=".$nm,
$sr, $ps);
        $this->cnnt->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
    }

    public function gtCnnt()
    {
        return $this->cnnt;
    }
}

class UsrMdl
{
```

```
    private $db;

    public function __construct()
    {
        $dbCnnt = new DbCnntMdl();
        $this->db = $dbCnnt->gtCnnt();
    }

    public function rgstrUsr($lgnStr, $pssStr, $rllStr)
    {
        $hsPss = password_hash($pssStr, PASSWORD_BCRYPT);
        $sqlStr = "INSERT INTO usr_tbl (lgn_str, pss_str,
rll_str) VALUES (:lgn, :pss, :rll)";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":lgn", $lgnStr);
        $sttm->bindValue(":pss", $hsPss);
        $sttm->bindValue(":rll", $rllStr);
        $sttm->execute();
        return true;
    }

    public function thntUsr($lgnStr, $pssStr)
    {
        $sqlStr = "SELECT * FROM usr_tbl WHERE lgn_str = :lgn";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":lgn", $lgnStr);
        $sttm->execute();
        $rslt = $sttm->fetch(PDO::FETCH_ASSOC);
        if ($rslt) {
            if (password_verify($pssStr, $rslt["pss_str"])) {
                $_SESSION["usr_id"] = $rslt["usr_id"];
                $_SESSION["rll_str"] = $rslt["rll_str"];
                return true;
            }
        }
        return false;
    }

    public function gtUsrDt($srchId)
    {
        $sqlStr = "SELECT usr_id, lgn_str, rll_str FROM usr_tbl
WHERE usr_id = :id";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":id", $srchId);
        $sttm->execute();
        return $sttm->fetch(PDO::FETCH_ASSOC);
    }
}
```

```
class GrpMdl
{
    private $db;

    public function __construct()
    {
        $dbCnnt = new DbCnntMdl();
        $this->db = $dbCnnt->gtCnnt();
    }

    public function crtGrp($grpNmt)
    {
        $sqlStr = "INSERT INTO grp_tbl (grp_nmt) VALUES (:nmt)";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":nmt", $grpNmt);
        $sttm->execute();
        return true;
    }

    public function gtGrpLst()
    {
        $sqlStr = "SELECT * FROM grp_tbl";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->execute();
        return $sttm->fetchAll(PDO::FETCH_ASSOC);
    }
}

class TchrMdl
{
    private $db;

    public function __construct()
    {
        $dbCnnt = new DbCnntMdl();
        $this->db = $dbCnnt->gtCnnt();
    }

    public function crtTchr($nmStr)
    {
        $sqlStr = "INSERT INTO tchr_tbl (nm_str) VALUES (:nm)";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":nm", $nmStr);
        $sttm->execute();
        return true;
    }
```

```
    public function gtTchrLst()
    {
        $sqlStr = "SELECT * FROM tchr_tbl";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->execute();
        return $sttm->fetchAll(PDO::FETCH_ASSOC);
    }
}

class SbjMdl
{
    private $db;

    public function __construct()
    {
        $dbCnnt = new DbCnntMdl();
        $this->db = $dbCnnt->gtCnnt();
    }

    public function crtSbj($sbjNmt)
    {
        $sqlStr = "INSERT INTO sbj_tbl (sbj_nmt) VALUES (:nmt)";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":nmt", $sbjNmt);
        $sttm->execute();
        return true;
    }

    public function gtSbjLst()
    {
        $sqlStr = "SELECT * FROM sbj_tbl";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->execute();
        return $sttm->fetchAll(PDO::FETCH_ASSOC);
    }
}

class TmTblMdl
{
    private $db;

    public function __construct()
    {
        $dbCnnt = new DbCnntMdl();
        $this->db = $dbCnnt->gtCnnt();
    }
```

```
    public function crtTmTbl($grpId, $tchrId, $sbjId, $dyStr,
$tmStr)
    {
        $sqlStr = "INSERT INTO tm_tbl (grp_id, tchr_id, sbj_id,
dy_str, tm_str)
                   VALUES (:grp, :tchr, :sbj, :dy, :tm)";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":grp", $grpId);
        $sttm->bindValue(":tchr", $tchrId);
        $sttm->bindValue(":sbj", $sbjId);
        $sttm->bindValue(":dy", $dyStr);
        $sttm->bindValue(":tm", $tmStr);
        $sttm->execute();
        return true;
    }

    public function gtTmTblByGrp($grpId)
    {
        $sqlStr = "SELECT * FROM tm_tbl WHERE grp_id = :grp";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":grp", $grpId);
        $sttm->execute();
        return $sttm->fetchAll(PDO::FETCH_ASSOC);
    }

    public function dtTmTbl($tmId)
    {
        $sqlStr = "DELETE FROM tm_tbl WHERE tm_id = :id";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":id", $tmId);
        $sttm->execute();
        return true;
    }
}

session_start();

$rqstMth = $_SERVER["REQUEST_METHOD"];
$rqstPth = $_SERVER["REQUEST_URI"];

if ($rqstMth === "POST" && strpos($rqstPth, "lgn") !== false) {
    $usrMdl = new UsrMdl();
    $usrMdl->thntUsr($_POST["lgn"], $_POST["pss"]);
    header("Location: /hm");
}

if ($rqstMth === "POST" && strpos($rqstPth, "rgstr") !== false) {
    $usrMdl = new UsrMdl();
```

```
    $usrMdl->rgstrUsr($_POST["lgn"], $_POST["pss"], "usr");
    header("Location: /lgn");
}

if ($rqstMth === "POST" && strpos($rqstPth, "crt_tm") !== false)
{
    $tmMdl = new TmTblMdl();
    $tmMdl->crtTmTbl($_POST["grp"], $_POST["tchr"],
$_POST["sbj"], $_POST["dy"], $_POST["tm"]);
    header("Location: /tm");
}

if ($rqstMth === "GET" && strpos($rqstPth, "tm") !== false) {
    $tmMdl = new TmTblMdl();
    $dt = $tmMdl->gtTmTblByGrp($_GET["grp"]);
    echo json_encode($dt);
}

class DbWrkMdl
{
    private $db;

    public function __construct()
    {
        $dbCnnt = new DbCnntMdl();
        $this->db = $dbCnnt->gtCnnt();
    }

    public function chckTbl($tblStr)
    {
        $sqlStr = "SHOW TABLES LIKE :tbl";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":tbl", $tblStr);
        $sttm->execute();
        $rslt = $sttm->fetch(PDO::FETCH_ASSOC);
        return $rslt ? true : false;
    }

    public function clnTbl($tblStr)
    {
        $sqlStr = "DELETE FROM ".$tblStr;
        $sttm = $this->db->prepare($sqlStr);
        $sttm->execute();
        return true;
    }

    public function gtCntFrmTbl($tblStr)
    {
        $sqlStr = "SELECT COUNT(*) AS cnt FROM ".$tblStr;
```

```
        $sttm = $this->db->prepare($sqlStr);
        $sttm->execute();
        $rslt = $sttm->fetch(PDO::FETCH_ASSOC);
        return $rslt["cnt"];
    }
}

class RprtMdl
{
    private $db;

    public function __construct()
    {
        $dbCnnt = new DbCnntMdl();
        $this->db = $dbCnnt->gtCnnt();
    }

    public function gtTmRprtByDy($dyStr)
    {
        $sqlStr = "SELECT * FROM tm_tbl WHERE dy_str = :dy";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":dy", $dyStr);
        $sttm->execute();
        return $sttm->fetchAll(PDO::FETCH_ASSOC);
    }

    public function gtTmRprtByTchr($tchrId)
    {
        $sqlStr = "SELECT * FROM tm_tbl WHERE tchr_id = :tchr";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":tchr", $tchrId);
        $sttm->execute();
        return $sttm->fetchAll(PDO::FETCH_ASSOC);
    }

    public function gtTmRprtBySbj($sbjId)
    {
        $sqlStr = "SELECT * FROM tm_tbl WHERE sbj_id = :sbj";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":sbj", $sbjId);
        $sttm->execute();
        return $sttm->fetchAll(PDO::FETCH_ASSOC);
    }
}

class VldtnMdl
{
    public function chckStrLn($strVal)
    {
```

```
        if (strlen($strVal) < 3) {
            return false;
        }
        if (strlen($strVal) > 255) {
            return false;
        }
        return true;
    }

    public function chckNmrc($vlVal)
    {
        if (!is_numeric($vlVal)) {
            return false;
        }
        if ($vlVal < 0) {
            return false;
        }
        return true;
    }

    public function chckArr($rrVal)
    {
        if (!is_array($rrVal)) {
            return false;
        }
        if (count($rrVal) === 0) {
            return false;
        }
        return true;
    }
}

class SsnMdl
{
    public function stSsnVl($kyStr, $vlStr)
    {
        $_SESSION[$kyStr] = $vlStr;
        return true;
    }

    public function gtSsnVl($kyStr)
    {
        if (isset($_SESSION[$kyStr])) {
            return $_SESSION[$kyStr];
        }
        return null;
    }

    public function clrSsn()
```

```
    {
        session_unset();
        session_destroy();
        return true;
    }
}

class LgMdl
{
    private $db;

    public function __construct()
    {
        $dbCnnt = new DbCnntMdl();
        $this->db = $dbCnnt->gtCnnt();
    }

    public function crtLg($ctgStr, $dtlStr)
    {
        $sqlStr = "INSERT INTO lg_tbl (ctg_str, dtl_str) VALUES
(:ctg, :dtl)";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":ctg", $ctgStr);
        $sttm->bindValue(":dtl", $dtlStr);
        $sttm->execute();
        return true;
    }

    public function gtLgLst()
    {
        $sqlStr = "SELECT * FROM lg_tbl ORDER BY lg_id DESC";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->execute();
        return $sttm->fetchAll(PDO::FETCH_ASSOC);
    }
}

class SttMdl
{
    private $db;

    public function __construct()
    {
        $dbCnnt = new DbCnntMdl();
        $this->db = $dbCnnt->gtCnnt();
    }

    public function gtSttDt()
    {
```

```
        $rslt = [];
        $sqlUsr = "SELECT COUNT(*) AS cnt FROM usr_tbl";
        $sqlGrp = "SELECT COUNT(*) AS cnt FROM grp_tbl";
        $sqlTm = "SELECT COUNT(*) AS cnt FROM tm_tbl";

        $stUsr = $this->db->prepare($sqlUsr);
        $stUsr->execute();
        $rslt["usr"] = $stUsr->fetch(PDO::FETCH_ASSOC)["cnt"];

        $stGrp = $this->db->prepare($sqlGrp);
        $stGrp->execute();
        $rslt["grp"] = $stGrp->fetch(PDO::FETCH_ASSOC)["cnt"];

        $stTm = $this->db->prepare($sqlTm);
        $stTm->execute();
        $rslt["tm"] = $stTm->fetch(PDO::FETCH_ASSOC)["cnt"];

        return $rslt;
    }
}

class CnflctMdl
{
    private $db;

    public function __construct()
    {
        $dbCnnt = new DbCnntMdl();
        $this->db = $dbCnnt->gtCnnt();
    }

    public function chckTmCnflct($grpId, $dyStr, $tmStr)
    {
        $sqlStr = "SELECT * FROM tm_tbl WHERE grp_id = :grp AND
dy_str = :dy AND tm_str = :tm";
        $sttm = $this->db->prepare($sqlStr);
        $sttm->bindValue(":grp", $grpId);
        $sttm->bindValue(":dy", $dyStr);
        $sttm->bindValue(":tm", $tmStr);
        $sttm->execute();
        $rslt = $sttm->fetch(PDO::FETCH_ASSOC);
        return $rslt ? true : false;
    }

    public function chckTchrCnflct($tchrId, $dyStr, $tmStr)
    {
        $sqlStr = "SELECT * FROM tm_tbl WHERE tchr_id = :tchr AND
dy_str = :dy AND tm_str = :tm";
        $sttm = $this->db->prepare($sqlStr);
```

```
            $sttm->bindValue(":tchr", $tchrId);
            $sttm->bindValue(":dy", $dyStr);
            $sttm->bindValue(":tm", $tmStr);
            $sttm->execute();
            $rslt = $sttm->fetch(PDO::FETCH_ASSOC);
            return $rslt ? true : false;
        }
    }
```

ДОДАТОК В

Слайди презентації

Міністерство освіти і науки України
Національний університет «Запорізька політехніка»
Кафедра програмних засобів

Створення програмної системи опрацювання даних для керування розкладом

Виконав: Роман ОСТАПЕНКО
студент групи КНТ-213сп
Керівник: Степан СКРУПСЬКИЙ
к.т.н., доцент НУ «Запорізька політехніка»

Рисунок В.1 – Слайд 1

2

Об’єкт, предмет та мета роботи

Об’єкт дослідження – алгоритми та процеси обчислень, пов’язані з організацією та опрацюванням даних для керування розкладом.

Предмет дослідження – програмні системи опрацювання даних для керування розкладом.

Метою роботи є розробка програмної системи опрацювання даних для керування розкладом для скорочення трудомісткості процесів планування та адміністрування розкладу.

Рисунок В.2 – Слайд 2

3

Завдання роботи

Для досягнення поставленої мети у кваліфікаційній роботі бакалавра необхідно розв'язати такі задачі:

– виконати аналіз предметної області та систем опрацювання даних для керування розкладом;

– здійснити проєктування системи опрацювання даних для керування розкладом;

– створити систему опрацювання даних для керування розкладом;

– дослідити розроблену систему опрацювання даних для керування розкладом.

Рисунок В.3 – Слайд 3

4

Порівняння існуючих аналогів

Критерій порівняння	Coursedog	Ad Astra	Arlo
академічне планування та розклад	+	+	+–
підтримка складних правил і обмежень	+	+–	–
підтримка змішаного навчання	+–	+–	+
простота впровадження та використання	+–	–	+
аналітика та історія змін розкладу	+	+	–
гнучкість налаштувань	+	+–	+–

Рисунок В.4 – Слайд 4

5

Вимоги до програмного забезпечення

При розробці програмного забезпечення для відтворення відеофайлів необхідно забезпечити такі функціональні вимоги:

– підтримка можливості збереження та відображення даних про розклад занять у базі даних;

– підтримка можливості додавання, редагування та видалення інформації, пов'язаної з розкладом;

– забезпечення реєстрації та авторизації користувачів із використанням механізмів контролю доступу та розмежування прав відповідно до ролей;

– забезпечення збереження та оброблення довідкових даних, що стосуються факультетів, кафедр, навчальних груп, викладачів та іншої інформації;

– забезпечення коректної взаємодії між клієнтською та серверною частинами з використанням стандартизованих форматів обміну даними;

– підтримка можливості зміни паролів для входу у систему зареєстрованих користувачів.

Рисунок В.5 – Слайд 5

6

Порівняння мов програмування

Критерій порівняння мов програмування	Мова програмування		
	PHP	Python	C#
Орієнтація на веброзробку	+	+–	+–
Простота розгортання на хостингу	+	+–	–
Швидкість розробки прикладних систем	+	+	+–
Підтримка типової серверної логіки	+	+–	+
Зручність супроводу в малих і середніх проєктах	+	+	+–

Рисунок В.6 – Слайд 6

7

Порівняння середовищ розробки

Критерій порівняння середовищ розробки	Середовища розробки		
	NetBeans	PhpStorm	Eclipse
Повнота інтегрованого середовища	+	+	+–
Зручність роботи з базами даних	+	+–	+–
Вбудовані засоби тестування	+	+	+–
Продуктивність середовища	+	+–	+–
Підтримка широкого спектра фреймворків PHP	+	+	+–

Рисунок В.7 – Слайд 7

8

Порівняння засобів зберігання та опрацювання даних

Критерій порівняння систем управління базами даних	Системи управління базами даних		
	MySQL	PostgreSQL	SQLite
Інтеграція з PHP	+	+–	+
Простота налаштування та розгортання	+	+–	+
Продуктивність для вебзастосунків	+	+–	+–
Масштабованість для багатокористувацьких систем	+	+	–
Підтримка транзакцій і цілісності даних	+	+	+–

Рисунок В.8 – Слайд 8

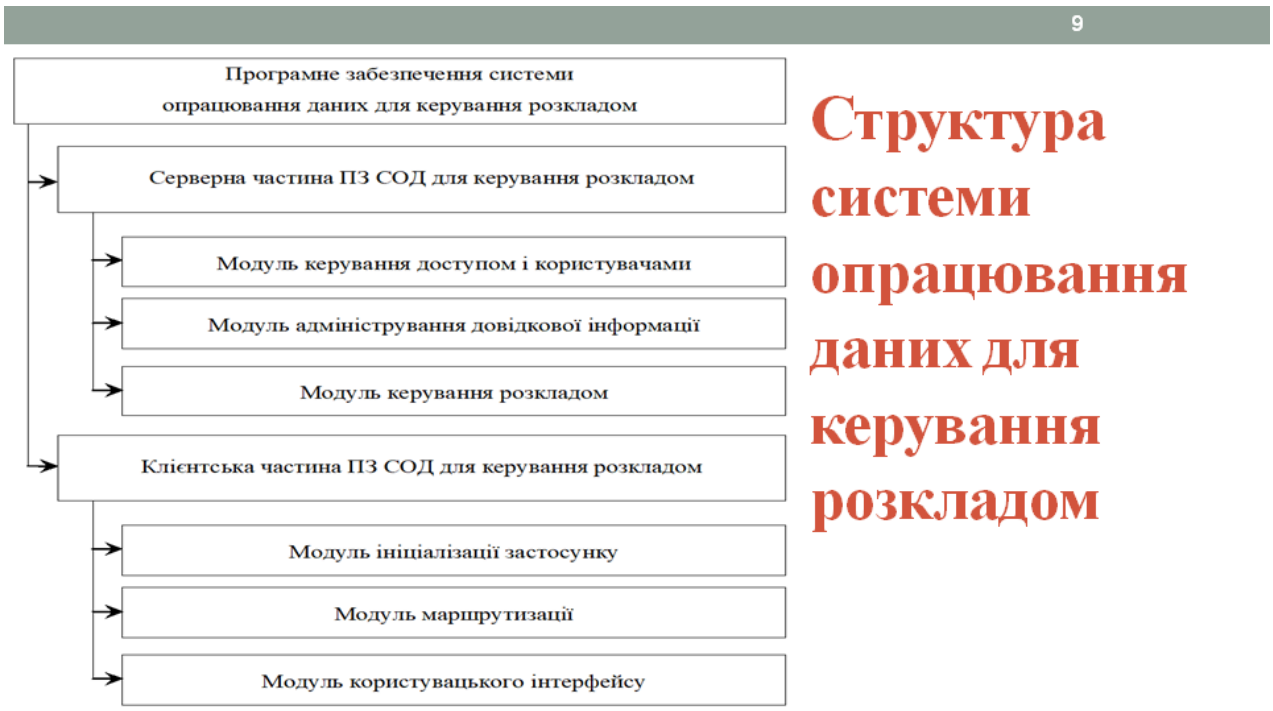

Рисунок В.9 – Слайд 9

Рисунок В.10 – Слайд 10

11

Схема бази даних

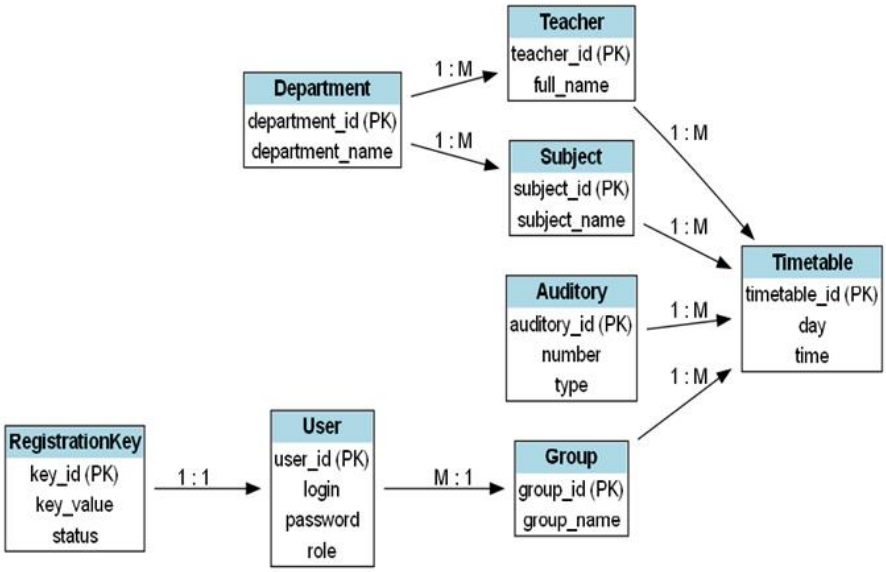

Рисунок В.11 – Слайд 11

12

Інтерфейс програми. Сторінка звичайного користувача

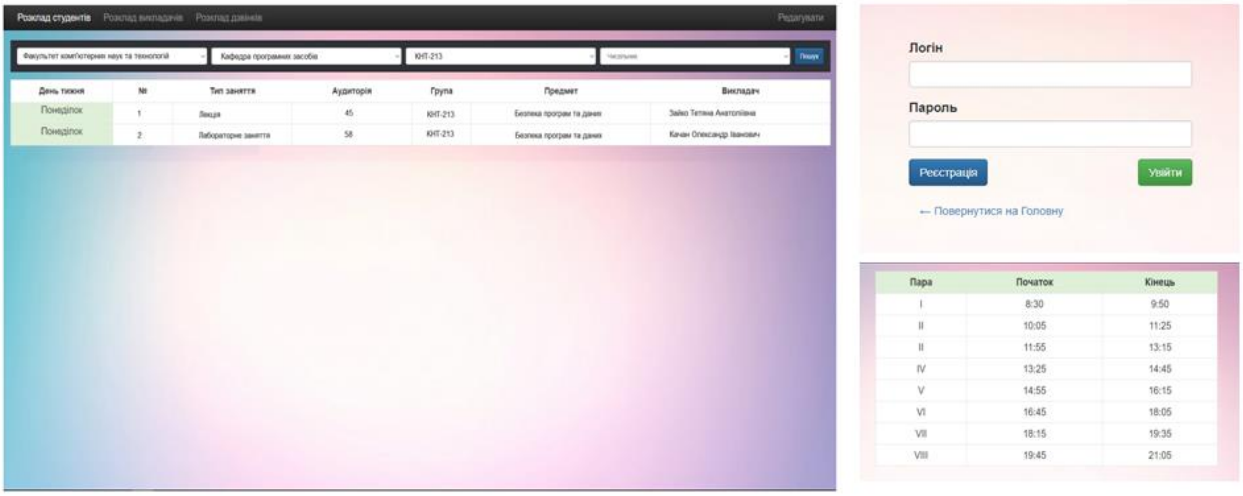

Рисунок В.12 – Слайд 12

13

Інтерфейс програми. Сторінка адміністратора

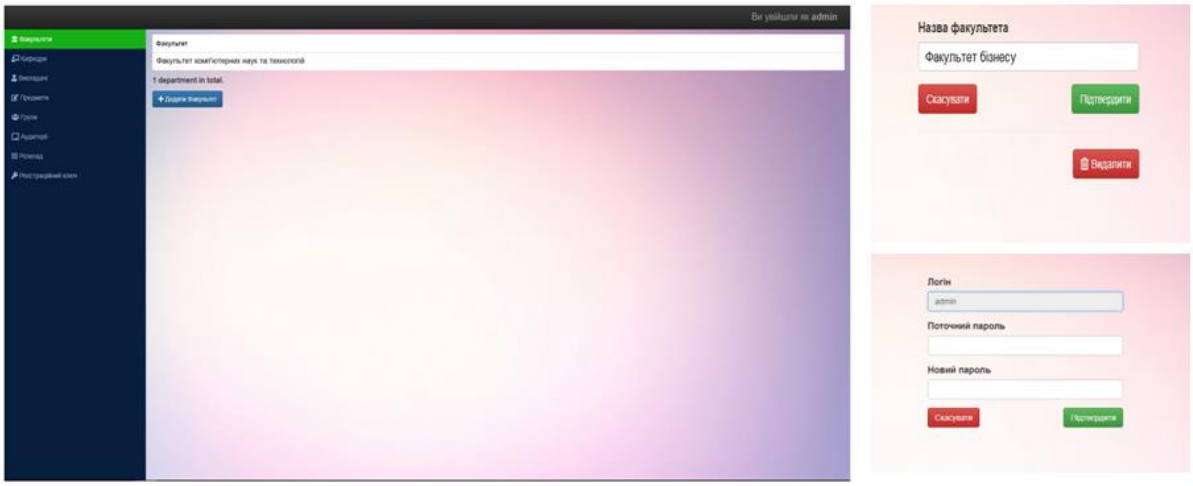

Рисунок В.13 – Слайд 13

14

Висновки

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено алгоритми та процеси обчислень, пов'язані з організацією та опрацюванням даних для керування розкладом.

Для реалізації системи опрацювання даних для керування розкладом обрано мову програмування PHP, середовище розробки NetBeans та систему управління базами даних MySQL.

Структура програмного забезпечення системи опрацювання даних для керування розкладом побудована за принципом чіткого розмежування серверної та клієнтської частин, що відповідає сучасним підходам до створення веборієнтованих інформаційних систем.

В результаті виконання завдання було отримано програмну систему, яка відповідає вимогам, представленим в технічному завданні, зручна для використання та виконує всі необхідні функції.

Усі завдання випускної дипломної роботи бакалавра повністю виконано.

Рисунок В.14 – Слайд 14