Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ
завідувач кафедри
мережевих та інтернет технологій
_____ Ю.В. Кравченко
«_____»_____2020 року

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

галузі знань 17 «Електроніка та телекомунікації»
за спеціальністю 172 «Телекомунікації та радіотехніка»

на тему:

ІНФОРМАЦІЙНА СИСТЕМА ФОРМУВАННЯ НАВЧАЛЬНОГО РОЗКЛАДУ ЗАНЯТЬ

Виконав: студентка групи МІТ -41
Ковальчин Марія Богданівна _____
(прізвище ім'я по-батькові) (підпис)

Керівник: доцент кафедри мережевих та інтернет технологій

к.т.н. Лещенко О.О. _____
(посада, прізвище ім'я по-батькові) (підпис)

Київ 2021

Міністерство освіти і науки України
«Київський Національний університет імені Тараса Шевченка»

Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ
завідувач кафедри
мережевих та інтернет технологій
_____ Ю.В. Кравченко
«_____»_____2021 року

ЗАВДАННЯ

Здобувачу вищої освіти Ковальчин Марії Богданівни
(прізвище, ім'я, по батькові)

1. ТЕМА РОБОТИ:
«Інформаційна система формування навчального розкладу занять»

затверджена на засіданні кафедри МІТ, протокол «04» грудня 2020 р. протокол № 8
2. Термін здачі закінченої роботи «30» травня 2021 р.
3. Вихідні дані до проекту
(роботи)

Мова програмування – MySQL, PHP.

Навчальний план спеціальності, навчальне навантаження викладачів кафедри, вимоги студентів до розкладу, вимоги викладачів до розкладу, дані про аудиторії.

4. Зміст пояснювальної записки (перелік питань, що їх потрібно розробити, обсяг – 35 - 60 стор.)

Вступ

1. Аналіз задачі розробки інформаційної системи формування навчального розкладу занять.

1.1. Опис та аналіз предметної області. Призначення інформаційної системи формування розкладу

1.2. ПОРІВНЯЛЬНИЙ АНАЛІЗ ІНФОРМАЦІЙНИХ СИСТЕМ ФОРМУВАННЯ НАВЧАЛЬНОГО

1.ЗРОЗКЛАДУ ЗАНЯТЬ ВИЩИХ НАВЧАЛЬНИХ ЗАКЛАДІВ

1.4 АНАЛІЗ ВИМОГ ДО ІНФОРМАЦІЙНОЇ СИСТЕМИ. ПОСТАНОВКА ЗАДАЧІ

2. МЕТОДИ СТВОРЕННЯ ТА ОПТИМІЗАЦІЇ НАВЧАЛЬНОГО РОЗКЛАДУ ЗАНЯТЬ

2.1 ЗАГАЛЬНИЙ ОПИС АЛГОРИТМУ

2.2. ОПИС ЦІЛЬОВОЇ ФУНКЦІЇ

2.3. ПРОЕКТУВАННЯ БАЗИ ДАНИХ. РОЗРОБКА ІНФОЛОГІЧНОЇ, ЛОГІЧНОЇ ТА ФІЗИЧНОЇ МОДЕЛІ БД

3. Розробка інформаційної системи формування навчального розкладу занять

3.1. Вибір середовища програмування

3.2. СТВОРЕННЯ ТАБЛИЦЬ

3.3. РЕАЛІЗАЦІЯ ЗАПИТІВ

3.4. СТВОРЕННЯ ІНТЕРФЕЙСУ ВИКЛАДАЧА ТА СТУДЕНТА

3.5. ВИКОНАННЯ ТЕСТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ НА РЕАЛЬНИХ ДАНИХ

Висновки

5. Перелік графічного матеріалу 8-10 слайдів

Дата видачі завдання

Керівник роботи _____ доц. Лещенко О. О.

(підпис) (посада, прізвище, ім'я, по батькові)

Завдання прийняв до виконання

Ковальчин М. В.

НА ДИПЛОМНУ РОБОТУ

Дата видачі завдання

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ РОБОТИ

Номер	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Підготовчий	28.01.2021	
2	Розділ 1	01.03.2021	
3	Розділ 2	01.04.2021	
4	Розділ 3	01.05.2021	
5	Доповідь та слайди	25.05.2021	
6	Пояснювальна записка	30.05.2021	

Здобувач вищої освіти _____ М. Ковальчин

(підпис)

Керівник _____ О.О. Лещенко

(підпис)

РЕФЕРАТ

Пояснювальна записка: 69с., 26 рис., 2табл., ?? додатків, ??джерел.

Об’єкт дослідження: навчальний розклад занять для ВНЗ.

Предмет дослідження: інформаційна система формування навчального розкладу занять.

Методи дослідження: методи цілочислового програмування.

Мета даної дипломної роботи полягає в розв’язання задачі автоматичної генерації розкладу занять для вищого навчального закладу з урахуванням певних вимог та обмежень.

Для досягнення поставленої мети необхідно розв’язати наступні задачі:

- дослідити методи цілочислового програмування та обрати найоптимальніший метод для розв’язання задачі побудови розкладу;
- сформувати систему вимог та обмежень для формування розкладу занять;
- розробити структурну модель автоматизованої системи формування розкладу занять;
- виконати проектування бази даних шляхом створення концептуальної, логічної та фізичної моделей даних;
- виконати побудову бази даних;
- здійснити програмну реалізацію обраного алгоритму;
- забезпечити доступ до функціоналу програми через веб-інтерфейс.

Розроблено структурну модель інформаційної системи. Виконано проектування бази даних та її побудову.

Здійснено програмну реалізацію обраного алгоритму та забезпечено управління функціоналом системи через веб-інтерфейс.

Практичне значення одержаних результатів полягає в можливості автоматизованого проектування розкладу занять, а також оптимізації структури розкладу для вищих навчальних закладів з урахуванням певних умов та обмежень. Матеріали розробки також можуть слугувати основою для подальшого удосконалення системи формування розкладу занять.

Наукова новизна дослідження полягає у застосуванні методів цілочислового програмування для розв’язання задачі автоматичного проектування розкладу занять на основі певних вимог та обмежень.

Галузь використання – заклади здобуття вищої освіти

Напрямки подальших досліджень: удосконалена система автоматичного формування розкладу занять з можливістю врахування суб’єктивних вимог та обмежень.

Ключові слова: РОЗКЛАД ЗАНЯТЬ, АВТОМАТИЗОВАНА СИСТЕМА, МЕТОДИ ЦІЛОЧИСЛОВОГО ПРОГРАМУВАННЯ, СТРУКТУРНА МОДЕЛЬ, БАЗА ДАНИХ

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 7
ВСТУП 8
РОЗДІЛ І. АНАЛІЗ ЗАДАЧІ РОЗРОБКИ ІНФОРМАЦІЙНОЇ СИСТЕМИ ФОРМУВАННЯ НАВЧАЛЬНОГО РОЗКЛАДУ ЗАНЯТЬ 10
1.1 Опис та аналіз предметної області 10
1.2 Визначення та класифікація інформаційних систем 11
1.2.1 Призначення інформаційної системи формування розкладу занять 12
1.2.2 Аналіз вимог до інформаційної системи формування розкладу занять 13
1.3 Порівняльний аналіз інформаційних систем формування навчального розкладу занять вищих навчальних закладів. 14
1.3.1 Аналіз автоматизація формування розкладу у НУБіП 17
1.4 Визначення та технології створення баз даних 19
1.5 Розробка структурної моделі автоматизованої системи формування розкладу занять 20
РОЗДІЛ ІІ. МЕТОДИ СТВОРЕННЯ ТА ОПТИМІЗАЦІЇ РОЗКЛАДУ НАВЧАЛЬНИХ ЗАНЯТЬ 25
2.1 Основні методи розв’язання задачі формування розкладу занять .. 25
2.1.1 Методи цілочислового програмування 25
2.1.2 Метод імітації відпалу 26
2.1.3 Метод розфарбування графу 27
2.1.4 Генетичний алгоритм 28
2.1.5 Загальний опис алгоритму підстановки 33
2.2 Вибір моделі даних 38
2.3 Проектування бази даних 42
2.3.1 Проектування концептуальної моделі бази даних 42

2.3.2 Проектування логічної моделі бази даних 44
2.3.3 Проектування фізичної моделі бази даних 47
2.4 Вибір системи управління базами даних 48
2.4.1 Система управління базами даних Oracle 49
2.4.2 Система управління базами даних MySQL 49
2.4.3 Система управління базами даних PostgreSQL 50
2.5 Проектування базового інтерфейсу користувача 51
2.5.1 Використання HTML для створення веб-сторінки 52
2.5.2 Використання CSS для створення веб-сторінки 54
2.6 Вибір мови програмування 54
РОЗДІЛ ІІІ. РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ФОРМУВАННЯ НАВЧАЛЬНОГО РОЗКЛАДУ ЗАНЯТЬ 58
3.1 Створення таблиць 58
3.2 Заповнення таблиць даними 59
3.2 Проектування інтерфейсу користувача 62
3.3 Виконання тестування інформаційної системи на реальних даних 66
ВИСНОВОК 70

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ПК – персональний комп’ютер

ІС – інформаційна система

БД – база даних

СУБД – система управління базами даних

ER – діаграма сутність-зв’язок

ТЗ – технічне завдання

ІК – інтерфейс користувача

GUI – графічний інтерфейс користувача

ВСТУП

Прогресивний розвиток сучасного світу, що сприяє матеріалізації нових знань в засобах праці, виникнення новаторських ідей та розробок, наявність великих потоків інформації спонукають до автоматизації процесів майже в кожній сфері діяльності людини. В умовах науково-технічного прогресу вимоги до рівня підготовки спеціалістів у вищих навчальних закладах постійно підвищуються. Одним з основних пунктів, що сприяє забезпеченню якості підготовки фахівців є раціональна організація навчального процесу, що включає питання його ефективного планування.

Передовим завданням у ході проектування та оптимізації навчального процесу у вищих навчальних закладах є складання розкладу. Якість розкладу занять має прямий вплив на ефективність роботи викладачів, рівень засвоєння студентами навчального матеріалу, раціональне використання аудиторного фонду вищого навчального закладу тощо.

Задачу формування розкладу занять не можна назвати тривіальною для людини через чисельність та неоднозначність параметрів оптимізації, які важко зібрати в одну систему. Питання проектування розкладу полягає у визначенні найбільш оптимального підбору зустрічей студентів та викладачів у заздалегідь заданий проміжок часу, як правило протягом тижня, з врахуванням множини обмежень. На даний час для автоматизації вказаного аспекту процесу розробки навчального плану було запропоновано велику кількість програмного забезпечення. Якщо проаналізувати вже запропоновані рішення, можна дійти висновку, що до сьогодні не існує такої моделі, яка мала б ідеальні показники ефективності.

Актуальність розробки автоматизованої системи формування розкладу полягає у плануванні та забезпеченні методично правильного процесу освоєння навчального плану, що вимагає чергування усіх форм навчальної роботи з урахуванням завантаженості викладачів та студентів, вибору

аудиторій що відповідають за критерієм кількості навчальних місць, а також інших об’єктивних та суб’єктивних показників.

У даній роботі запропоновано варіант розв’язання задачі побудови оптимального розкладу занять при дотриманні всіх регламентованих обмежень.

Практичним значенням одержаних результатів є можливість автоматизованого проектування розкладу занять, а також оптимізація структури розкладу для вищих навчальних закладів з урахуванням певних умов та обмежень. Матеріали розробки також можуть слугувати основою для подальшого удосконалення системи формування розкладу занять.

І АНАЛІЗ ЗАДАЧІ РОЗРОБКИ ІНФОРМАЦІЙНОЇ СИСТЕМИ ФОРМУВАННЯ НАВЧАЛЬНОГО РОЗКЛАДУ ЗАНЯТЬ

1.1 Опис та аналіз предметної області

Предметною областю даної роботи є автоматизація розробки інформаційної системи формування навчального розкладу занять.

Розклад занять – це документ, що регламентує трудовий ритм і має прямий вплив на рівень освоєння навчального матеріалу студентами, а також творчу віддачу викладачів в ході навчального процесу. Якісний розклад занять повинен забезпечувати оптимальний робочий режим студентів протягом семестру, створювати оптимальні умови для викладацького складу, рівномірно розподіляти робоче навантаження, а також забезпечувати ефективне використання аудиторного фонду.

Предмет – це певна наукова дисципліна, система знань та умінь, що викладається студентам в рамках освітнього плану. Предмети можуть мати наступні форми занять: лекції, практики, лабораторні. Виконання даної задачі побудовано на врахуванні лекційних та практичних форм занять. Лекційні заняття група студентів повинна відвідувати повним складом, практичні заняття передбачають поділ групи на дві частини. Таким чином, різні форми занять вимагають різних за місткістю аудиторій. Цей фактор є необхідним до врахування в процесі проектування розкладу. Кожна форма занять повинна займати дві академічні години або одну пару.

Викладач – людина, яка займається викладанням певної навчальної дисципліни. Певні викладачі можуть вести лише певні форми занять. Такі вимоги регулюються через навчальний план, а також мають розглядатися як жорсткі. В процесі проектування навчального розкладу занять потрібно враховувати фізичне обмеження, яке полягає у неможливості викладача бути у двох аудиторіях одночасно.

Студенти – особи, які проходять навчання за певною освітньою програмою у вищому навчальному закладі. Студенти діляться на потоки за терміном навчання – курси. Максимальна кількість студентів у групі становить 25 осіб. При виконанні задачі із проектування розкладу занять не раціонально розглядати кожного студента, як окрему сутність. Важливо враховувати саме групи студентів. В процесі розробки розкладу занять необхідно враховувати, що кількість студентів у групі не може бути більшою, ніж числовий показник місткості аудиторії, в якій планується проведення заняття для даної групи. Кожній групі присвоєно власний код, який складається з назви спеціальності, номеру курсу та номеру групи.

Аудиторії – це обладнані спеціальним чином фізичні кімнати в приміщенні навчального закладу, використовуються для проведення занять. Кожна аудиторій облаштована під певний вид занять: лабораторії – для лабораторних та практичних робіт; лекційні аудиторії – для лекцій. В окремих аудиторіях можливе проведення кількох типів занять. Проте, така можливість є доступною тільки тоді, коли необхідна за типом аудиторія зайнята. Числовим показником кожної з аудиторій є кількість місць, на яку вона розрахована.

Якісно спроектований розклад занять може стати хорошою основою для реалізації наукового потенціалу навчальних курсів, а також сприяти кращому рівню сприйняття матеріалу студентами.

1.2 Визначення та класифікація інформаційних систем

Інформаційною системою називають сукупність технічних та організаційних засобів, що забезпечують процес збереження та обробки інформації з метою задоволення інформаційних потреб користувачів.

Автоматизована ІС - сукупність технічних та організаційних засобів, що забезпечують процес збереження та обробки інформації з метою задоволення інформаційних потреб користувачів, де аналіз та прийняття рішень відбувається на основі автоматизації інформаційних процесів.

Класифікація ІС та їх можливості:

- БД, призначенням якої є зберігання структурованої інформації. Інформація у таких системах поділяється на окремі категорії. До систем такого типу належать електронні каталоги, цифрові записники, масиви даних різних відділів в компаніях, розклад уроків;
- БД, призначенням якої є зберігання не структурованої інформації. У системах такого типу не відбувається строгий поділ інформації на окремі категорії, а розміщується в довільному порядку. Пошук необхідних даних в таких системах відбувається за рахунок введення ключового запиту до системи. Всесвітня мережа Інтернет слугує найяскравішим прикладом ІС наведеного типу.
- інформаційно-аналітична система. Прикладом таких ІС слугують: 1С, Excel, тощо.

Існує два класи ІС: файл-сервер та клієнт-сервер. Різниця між наведеними класами полягає у тому, що у випадку використання ІС класу клієнт-сервер, користувач отримує доступ до даних шляхом використання мережі Інтернет. У випадку використання ІС класу файл-сервер інформаційні масиви зберігаються та обробляються лише на одному комп'ютері.

1.2.1 Призначення інформаційної системи формування розкладу занять

Призначенням інформаційної системи формування розкладу є автоматичне генерування розкладу навчальних занять з урахуванням певних правил та обмежень. До основних задач, що повинні бути вирішені в процесі проектування розкладу належить:

- виконання робочого навчального плану та графіку процесу навчання;
- забезпечення оптимального режиму роботи студентів протягом навчального року;

- забезпечення оптимального режиму роботи професорсько-викладацького складу;
- рівномірний розподіл завантаженості студентів протягом тижня;
- раціональне використання аудиторного фонду вищого навчального закладу.

1.2.2 Аналіз вимог до інформаційної системи формування розкладу занять

Організація розкладу занять вимагає врахування багатьох факторів. Умовно такі фактори можна розділити на дві групи: жорсткі вимоги та нежорсткі вимоги. До жорстких вимог належать наступні пункти:

- рівномірний розподіл кількості академічних годин, які повинні бути відпрацьовані згідно з науковим планом освітньої програми;
- відповідність викладачів та дисципліни;
- відповідність числового показника місткості аудиторій та кількості студентів в групі (кількість місць, на яку розрахована аудиторія не може бути меншою, ніж кількість студентів в групі);
- максимальна кількість пар на день не повинна бути більшою за 3;
- лекційні та практичні заняття повинні чергуватися.

До списку нежорстких умов належать:

- вимоги та побажання викладачів;
- вимоги та побажання студентів;
- не рекомендується наявність вікон у викладачів та студентів.

Виконання нежорстких умов не є обов’язковим, проте, у разі їх виконання рівень ефективності організації даного аспекту планування робочого процесу підвищується.

1.3 Порівняльний аналіз інформаційних систем формування навчального розкладу занять вищих навчальних закладів.

Станом на сьогодні реалізовано деяку кількість систем, які дозволяють проектувати розклад занять в автоматизованому режимі. Дані системи відрізняються своїми функціональними можливостями, гнучкістю, швидкістю обробки інформації, мають свої переваги та недоліки. Найбільш популярними системами, які забезпечують автоматичне генерування розкладу, є наступні програмні продукти:

- ссистема «Foss Look»;
- ссистема «Ректор-ВНЗ»;
- ссистема «Галактика».

Узагальнене порівняння функціональних можливостей автоматизованих систем наведено в таблиці 1.1.

Таблиця 1.1 – Узагальнене порівняння функціональних можливостей автоматизованих систем

Функція	Foss Look	Ректор - ВНЗ	Галактика
Режим проектування розкладу	автоматичний, ручний, змішаний	автоматичний, ручний, змішаний	автоматичний, ручний, змішаний
Можливість складання кількох варіацій розкладу з можливістю вибору кращого варіанту	Так	Ні	Ні
Врахування суб’єктивних побажань студентів та викладачів	НІ	Ні	Так

Консолідація розкладів	Так	Ні	Ні
Налаштування критеріїв оптимізації	Так	Так	Так
Врахування максимальної можливої кількості занять на день для студентів	Так	Так	Так

Таблиця 1.1 – Узагальнене порівняння функціональних можливостей автоматизованих систем

Побудова розкладу для 2 і більше змін	Так	Так	Так
Оперативне резервування приміщень	Так	Ні	Так
Перегляд розкладів по веб-інтерфейсу	Так	Ні	Так
Оперативна зміна розкладу	Так	Ні	Ні
Врахування паралельних занять	Так	Так	Так
Розбиття на підгрупи	Так	Так	Так
Дотримання інтервалу між заняттями	Так	Ні	Так
Розмежування прав доступу	Так	Ні	Так

Системні вимоги підсистем:

- “Foss Look”: СУБД – MySQL (включений в інсталяцію), Microsoft SQL Server; ОС Windows XP Professional SP3 і вище;
- “Ректор ВНЗ”: СУБД не підтримується; ОС Windows XP, Windows Vista, Windows 7;
- “Галактика”: ОС Windows (XP, 7), NET Framework v.4 і вище СУБД MS Server 2008, Server 2003 [9].

Основний функціонал програмного забезпечення “Foss Look”:

- можливість запису та обробки даних, потрібних для формування розкладу занять (список викладачів, кафедр, інститутів, груп, аудиторій);
- можливість паралельного ведення 2 і більше розкладі дзвінків по різних корпусах, групах, днях тижня;
- можливість відслідковування зв’язків вільних аудиторій з досліджуваними видами робіт, дисциплінами;
- можливість налаштування пріоритетів використання ресурсів в процесі складання розкладу (пріоритети по дисциплінах, викладачах, складність досліджуваних предметів, проведення лекційних та практичних занять);
- можливість здійснення контролю групування студентів (група, підгрупа, потік), адмініструвати їх переміщення по доступних аудиторіях;
- можливість здійснення налаштування тривалості занять;
- можливість врахування вимог освітньої програми;
- можливість створення обмежень у використанні ресурсів в процесі конструювання розкладу занять, графіку викладачів та студентів, доступності аудиторій;
- можливість ведення історії внесених змін;
- можливість оцінювання ефективності генерованого розкладу шляхом розрахунку інтегральних показників якості розкладу.

Основний функціонал програмного забезпечення “Ректор ВНЗ”:

- наявність автоматичного, ручного та змішаного режимів керування;
- можливість налаштування критеріїв оптимізації;
- врахування максимальної можливої кількості пар на день для студентів;
- побудова розкладу для кількох змін;
- врахування наявності паралельних занять;
- можливість розбиття на підгрупи.

Основний функціонал програмного забезпечення “Галактика”:

- можливість ведення паралельного розкладу дзвінків в умовах різних днів, тижнів, корпусах, групах;
- можливість здійснення контролю зв’язку вільних аудиторій, в тому числі спеціально обладнаних, з факультетами, кафедрами, видами робіт, дисциплінами;
- можливість здійснення налаштування пріоритетів користування ресурсами для проектування розкладу. Наприклад, пріоритети по викладачах, дисциплінах, проведення практичних занять;
- можливість врахування віддаленості корпусів, їх пріоритетність при плануванні розкладу;
- можливість контролю групування студентів (група, підгрупа, потік), адмініструвати їх переміщення по доступних аудиторіях.

1.3.1 Аналіз автоматизація формування розкладу у НУБіП

Програмний продукт складається із кількох підсистем: керування загальною інформацією, керування документообігом, формування розкладу, управління запитами на редагування, розповсюдження розкладу. На рисунку 1.4 представлено структурну модель системи автоматизованого формування розкладу занять НУБіП.

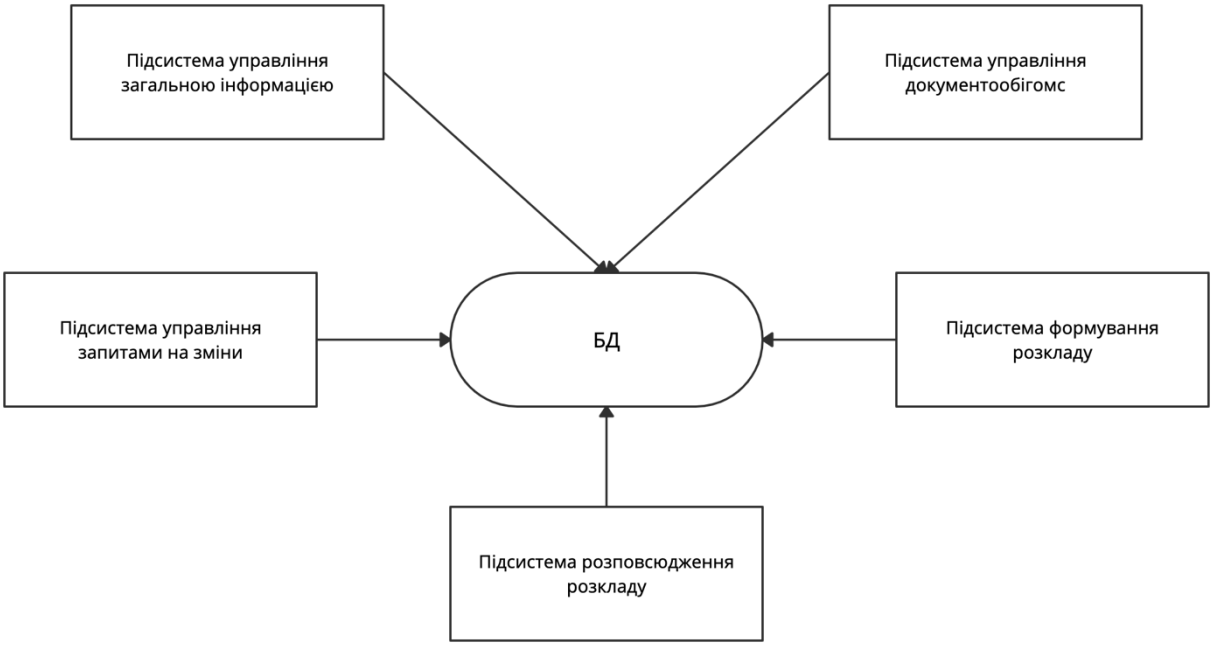


Рисунок 1.4 – Структурну модель системи автоматизованого формування розкладу занять НУБіП

Підсистема управління загальною інформацією виконує функцію керування внесення даних про структуру вищого навчального закладу (факультети, кафедри, курси, аудиторний фонд, спеціальності, групи студентів).

Підсистема керування документообігом забезпечує можливість внесення до ІС інформації, що може зазнавати систематичного оновлення. Наприклад, інформація про розподіл занять між викладачами.

Підсистема формування розкладу забезпечує безпосереднє створення, а також редагування розкладу занять. Програмний продукт дозволяє здійснювати вибір дисципліни, виду заняття, аудиторії, викладача, групу, день тижня, пару в автоматичному режимі. Розклад занять формується на базі системи обмежень, яку необхідно враховувати з метою уникнення накладок в розкладі.

Підсистема керування запитами та змінними забезпечує можливість врахування побажань викладачів щодо формування розкладу. В процесі

роботи ІС проводить обробку запиту у подальшому схваливши або відхиливши запит.

Функціонування підсистеми розповсюдження реалізовано в двох режимах: збереження розкладу у таблиці Excel для внутрішнього користування або завантаження його на сайт для загального доступу.

Централізована реляційна база даних розроблена у середовищі Microsoft SQL Server Management Studio. На рисунку 1.2.2 зображено структуру програмної системи.

Зчитування інформації про навчальні корпуси, аудиторії, групи студентів, викладачів відбувається автоматично із відповідного файлу.

1.4 Визначення та технології створення баз даних

Невід'ємною частиною сучасного світу є інформація. Прогресивний розвиток сучасного світу сприяє збільшення інформаційних потоків.

База даних — є впорядкованою структурою, сновним призначенням якої є зберігання, зміна, а також обробка взаємопов'язаної інформації, здебільшого великих обсягів. Бази даних описані та організовані на основі певної предметної області, яка обумовлює характеристику цих даних та логічні зв'язки між ними. Сфера використання БД є дуже широкою Ось список деяких систем, де можуть бути використані бази даних:

- електронні каталоги продукції;
- геоінформаційні системи;
- транспортні системи;
- системи обліку;
- системи адміністрування контенту;
- електронні словники;

Інформація в БД організовується відповідно до моделі організації даних. У своїй структурі БД містить власне дані, а також інструменти для маніпуляції

над даними. Чітко визначений взаємозв'язок інформації робить дані в БД залежними одне від одного: зміна інформації в одному рядку може спричинити зміни в інших.

Розрізняють три основні типи баз даних, а саме:

– ієрархічна модель організації;
– мережева модель організації;
– реляційна модель організації.

За розміщенням БД поділяють на локальні та розподілені. Підтримка локальної БД відбувається на одному комп’ютері. Частини розподіленої бази даних розміщуються на різних комп’ютерах.

Для забезпечення роботи з базами даних використовують системи управління базами даних. Основними характеристиками сучасних СУБД є:

– контроль за надлишковістю даних;
– несуперечливість даних;
– забезпечення цілісності бази;
– цілісність описується за допомогою обмежень;
– незалежність прикладних програм від даних;
– можливість спільного використання даних;
– високий рівень безпеки.

Програмне забезпечення СУБД працює як інтерфейс між кінцевим користувачем та базою даних, одночасно керуючи даними, механізмом бази даних та схемою бази даних, щоб полегшити організацію та маніпулювання даними. Такі системи базуються на реалізації програмних та технічних засобів, що забезпечує керування та використання баз даних. Найбільш використовуваними СУБД є Oracle, MySQL, PostgreSql.

1.5 Розробка структурної моделі автоматизованої системи формування розкладу занять

З метою забезпечення коректного функціонування, структура автоматизованої система формування розкладу занять для вищого навчального закладу реалізована за рахунок системи зв’язків між окремими модулями. Кожен з таких модулів(підсистем) виконує окремі функціональні операції.

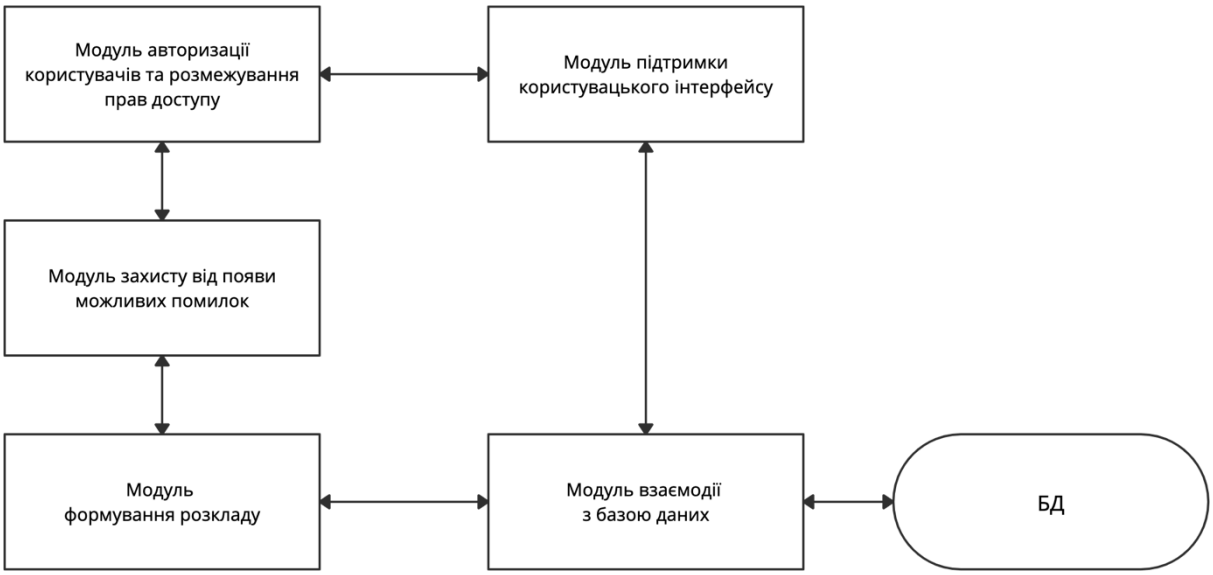


Рисунок 1.4 – Структурна модель системи автоматизованого формування розкладу занять

Основними модулями системи формування розкладу занять є:

- модуль авторизації користувачів та розмежування прав доступу;
- модуль взаємодії з базою даних;
- модуль захисту від появи можливих помилок;
- модуль формування розкладу;
- модуль підтримки користувацького інтерфейсу.

Реалізація вирішення задачі автоматизації побудови розкладу занять передбачає розробку та імплементацію окремого модуля авторизації користувачів з метою забезпечення подальшої програмної підтримки роботи в режимах доступу для користувача та адміністратора.

В процесі вирішення задачі встановлюється система прав користувача для можливості здійснення подальшого контролю санкціонованого використання об'єктів ІС. Така система прав та обмежень встановлюється системою після виконанння операцій ідентифікації та аутентифікації. При розмежуванні доступу за списками права користувача прийнято представляти списком ресурсів, доступних користувачеві, а також правами щодо доступу до кожного з цих ресурсів.

Модуль взаємодії з базою даних виконує функцію обробки набору даних для формування вхідної інформації алгоритму автоматизації формування занять з урахуванням визначеної системи вимог та обмежень.

Виконання даної задачі передбачає створення програмного модуля, що забезпечує захист інформаційної системи від появи можливих помилок. Такий принцип є фундаментальним для побудови автоматизованої системи і реалізується методом системної імплементації масивів дозволів та заборон на доступ до інформаційного забезпечення бази даних. Таким чином, система дозволятиме доступ групи лише до тих дисциплін з навчального плану, які викладаються для певної групи в поточному. До функціональних можливостей даного модуля входить також перевірка коректності та повноти введених даних (наприклад перевірка наявності викладача для кожної дисципліни).

Модуль формування розкладу базується на виконанні визначеного алгоритму автоматизації формування занять з урахуванням визначеної системи вимог та обмежень. Вхідні дані для виконання алгоритму даний модуль отримує за рахунок результату роботи модуля взаємодії з базою даних. Підсистема формування розкладу забезпечує безпосереднє створення, а також редагування розкладу занять. Програмний продукт дозволяє здійснювати вибір дисципліни, виду заняття, аудиторії, викладача, групу, день тижня, пару в автоматичному режимі. Розклад занять формується на базі системи обмежень, яку необхідно враховувати з метою уникнення накладок в розкладі.

Модуль підтримки користувацького інтерфейсу повинен гарантувати, зручну взаємодію користувача з інформаційною системою. Спираючись на

права доступу, отримані від модуля авторизації, модуль інтерфейсу користувача повинен забезпечувати можливість введення, редагування та видалення даних, необхідних для побудови розкладу, а саме: перелік наявних аудиторії, перелік дисциплін, професорсько-викладацький склад, перелік груп студентів. Даний модуль повинен також дозволяти здійснювати запуск визначеного алгоритму та відображення сформованих результатів.

Кожен з описаних модулів виконує окремі чітко визначені функціональні операції. Коректна робота інформаційної системи автоматизованого формування розкладу занять не можлива за умови відсутності будь-якого з цих модулів.

Висновки до розділу І

В процесі виконання розділу було визначено, що предметною областю даної роботи є автоматизація розробки інформаційної системи формування навчального розкладу занять. Проведено детальний аналіз наведеної предметної області.

Сформовано визначення інформаційної системи та наведено загальну класифікацію інформаційних систем. Розглянуто призначення інформаційної системи формування розкладу занять. Преведено аналіз вимог до інформаційної системи формування розкладу.

Проведено порівняльний аналіз ІС формування розкладу занять для вищих навчальних закладів, а також проведено аналіз автоматизації формування розкладу у НУБіП.

Сформовано визначення БД та наведено основні технології створення БД.

Розроблено структурну модель автоматизованої системи формування розкладу занять.

ІІ МЕТОДИ СТВОРЕННЯ ТА ОПТИМІЗАЦІЇ РОЗКЛАДУ НАВЧАЛЬНИХ ЗАНЯТЬ

2.1 Основні методи розв’язання задачі формування розкладу занять

Задачі формування розкладу занять належать до виду комбінаторних. Важливе значення для такого класу задач має розмірність. Розмірністю називають величину системи даних, необхідних для розв’язування конкретної задачі. Розмірність в задачах формування розкладу може бути настільки великою, що метод розв’язання їх класичним перебором варіантів є неможливим. Таким чином розв’язування задачі формування розкладу занять зводиться до розв’язування задач цілочислового лінійного програмування. Для їх розв’язування застосовуються методи відсікання гілок або меж. Основними методами для розв’язання задач формування розкладу занять є:

- комбінаторні методи;
- мережні методи;
- евристичні методи.

2.1.1 Методи цілочислового програмування

Метод цілочислового програмування базується на виділенні окремих змінних, побудови математичної моделі задачі у вигляді обмежень, виконання яких є обов’язковим для виконання задачі. Значення змінних необхідно знайти в процесі виконання задачі. Алгоритм методів цілочислового програмування містить наступні кроки:
виділення змінних:

- проектування математичної моделі задачі;
- складання цільової функції;

– знаходження точок максимуму і мінімуму за умови використання різних математичних методів.

Методи цілочислового програмування мають свої недоліки. До списку недоліків належить:

– гарантія отримання прийнятого рішення відсутня;
– важкість оцінки впливу різних чинників на результат вирішення задачі;
– збільшення витрат часу;
– збільшення розмірності задачі.

2.1.2 Метод імітації відпалу

Основна ідея алгоритму імітації відпалу базується на дослідженнях про поведінку атомів металу під час його відпалу. Якщо нагріти метал до температури, яка є вищою за точку його плавлення, це спричинить хаотичний рух атомів. За умов високої температури енергія атомного руху перешкоджає процесу досягнення атомами мінімальної енергії, як це відбувається в усіх фізичних системах. Поступове зниження температури металу призводить до появи низькоенергетичних станів атомів. Процес охолодження триває до тих пір, поки не буде досягнуто глобальний мінімум – найнижчий з можливих станів енергії атомів.

В умовах задачі формування розкладу занять таку енергію можна розглядати у вигляді цільової функції, спроєктованої на основі ідеї, де кожен невдало складений розклад карається так званим штрафом, а розклад, складений вдало, розглядати як низькоенергетичний стан.

Схема для презентації ідеї алгоритму імітації відпалу для задачі формування розкладу:

– під час першої ітерації необхідно сформувати розклад X^0, який вважається єдиним розв’язком задачі на даному етапі ($X = X^0$);

- відбувається встановлення початкового високого значення температури T^0, а також процес мутації розкладу. Операція мутації може бути здійснена шляхом зміни дати та часу проведення занять;
- після виконання операцій мутації, а також поточного розкладу відбувається формування нового варіанту розкладу занять X'. Новостворений варіант розкладу повинен відрізнятися від попереднього;
- формується висновок з приводу зміни цільової функції $\Delta f = f(X) - f(X')$. Якщо $\Delta f < 0$, тоді можна зробити висновок, що розв'язок задачі не погіршився. Отже, новий варіант розкладу стає поточним ($X = X'$) з ймовірністю $1 - p$. Якщо $\Delta f > 0$, можна зробити висновок, що розв'язок задачі погіршився. Отже, новий варіант розкладу стає поточним з ймовірністю $p = e^{-\Delta f/T}$.
- виконується функція зміни температури, яка була встановлена попередньо. Під час проходження кожної ітерації температура зменшується. Відповідно відбувається зменшення ймовірності прийняти поточного розкладу з більшим значенням цільової функції.
- якщо не виконується виконання заданого числа ітерацій або виконання заданого числа ітерацій без покращення цільової функції на задане значення, необхідно перейти до третього кроку алгоритму.

Швидкість зміни температури впливає на точність розв'язку, отриманого за умови використання даного алгоритму.

Даний підхід є ефективним для формування невеликих розкладів.

2.1.3 Метод розфарбування графу

Розв'язування задачі формування розкладу занять можна здійснити за умови використання методу розфарбування графу. Основною ідеєю задачі розфарбування графу є пошук мінімального числа кольорів, необхідних для

розфарбування певного графу таким чином, щоб для розфарбування кожної пари сусідніх вершин графу використовувались різні кольори. Таку задачі називають також пошуком хроматичного числа графу.

Для розв’язання задачі формування розкладу методом розфарбування графу необхідно здійснити побудову деякого графу. Кожну вершину такого графа необхідно розглядати як заняття, заплановане навчальним планом ВНЗ. Між двома вершинами графа можуть виникати конфлікти. Прикладом конфлікту в умовах даного методу слід вважати ситуацію, коли в один і той самий час відбувається проведення занять з одним викладачем. Якщо між двома деякими вершинами виникає конфлікт, такі вершини з’єднуються ребром. Кожен колір відповідає одному періоду занять.

Використання описаного алгоритму вважається не достатньо ефективним. Проте, застосування алгоритму розфарбування графа може бути корисним у випадку його комбінації з іншими алгоритмами.

2.1.4 Генетичний алгоритм

З метою підвищення якості отриманого розкладу занять, пропонується використання генетичного алгоритму.

Генетичний алгоритм – це еволюційний алгоритм пошуку, який часто застосовується для розв’язування задач з оптимізації та моделювання, реалізується методом послідовного підбору, комбінування і варіативності шуканих параметрів з використанням механізмів, що мають фундаментальну схожість з біологічною еволюцією. Перевагою даного методу є те, що додаткові вимоги до цільової функції не ставляться. Під час кожної ітерації відбувається робота з множиною рішень, що дає змогу в багатьох випадках більш детально в порівнянні з іншими методами багатовимірної оптимізації аналізувати простір пошуку. Використання оператора “схрещення” є основною особливістю пропонованого алгоритму.

Перевагами генетичного алгоритму є:

- простота та зрозумілість концепції реалізації;
- можливість розпаралелювання;
- простота кодування вхідної та вихідної інформації;
- можливість використання різного виду параметрів досліджуваних систем;
- адаптивність параметрів генетичного пошуку до особливостей вирішуваної задачі;
- можливість пошуку складних рішень за великої розмірності систем.

На рисунку 2.1 наведено структурну схему роботи генетичного алгоритму.

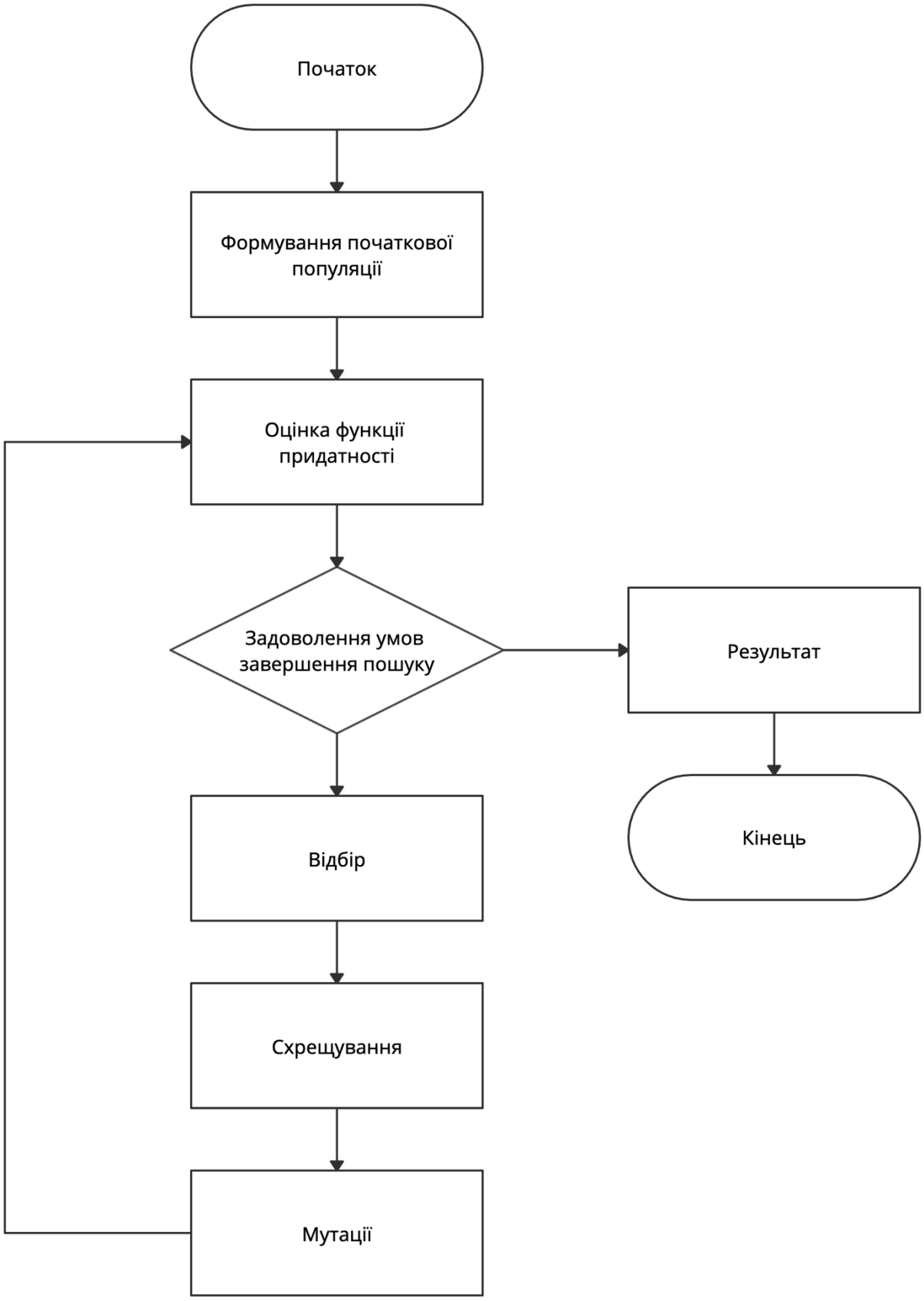


Рисунок 2.1 - Структурна схема генетичного алгоритму

Будівельною одиницею генетичного алгоритму є хромосома. Хромосома - це набір генів. Спираючись на умови розв'язуваної задачі, хромосомою масив даних, кожен елемент якого це заняття в розкладі. Для створення хромосоми необхідно використовувати таку вхідну інформацію:

1) $T = \{tw, td, tp\}$ – множина часу проведення навчальних занять, де $\mathrm{t}_{\mathrm{w}}^{\mathrm{k}} == \{1..Nw\}$ – номер тижня, $\mathrm{t}_{\mathrm{d}}^{\mathrm{k}} = \{1..N\mathrm{d}\}$ – номер дня тижня, $\mathrm{t}_{\mathrm{p}}^{\mathrm{k}} = \{1..N\mathrm{p}\}$ – номер пари протягом дня;
2) G = {g1, g2...INg} – множина груп студентів, де gi - номер групи студентів, Ng - загальна кількість груп студентів;
3) $L = \{l1, l2 \ldots INI\}$ – множина викладачів, де li – номер викладача, NI - загальна кількість викладачів;
4) $D = \{ d1, d2 \ldots dNd \}$ – множина дисциплін, де Di– номер навчальної дисципліни, Nd– загальна кількість дисциплін в університеті;
5) $A = \{ a1, a2 \ldots aNa\}$ – множина аудиторій університету, де ai– номер аудиторії, Na – загальна кількість аудиторій в університеті.

Для деталізації вхідних даних проведемо поділ аудиторій на два типи: аудиторії для проведення лекційних занять - A_b, та аудиторії для проведення практичних і лабораторних занять - A_s,. Тип аудиторії приймає значення b, s. Отже, $\mathrm{A} = \mathrm{A}_b \cup A_s$.

Множини дисциплін, викладачів та груп студентів утворюють між собою тісний зв'язок. В наслідок цього утворюється нова множина, яку можна представити наступним чином:

$$\mathrm{Z} = \{\mathrm{z_i}\} = (\mathrm{z_i^l}, \mathrm{z_i^t},)\mathrm{z_i^d}, \mathrm{z_i^g}, \mathrm{z_i^a},$$

Де $\mathrm{z_i^l}$- викладачі, $\mathrm{z_i^l}$ - дисципліни, $\mathrm{z_i^d}$- дисципліни, $\mathrm{z_i^g}$ - групи студентів, $\mathrm{z_i^t}$ - тип занять, $\mathrm{z_i^a}$ - аудиторії.

Висновок щодо хромосоми виводиться спираючись на значення цільової функції. Під час формування висновку відбувається перевірка на придатність для подальшого використання. Значення цільової функції оцінюється окремо для кожної хромосоми. На основі цього значення приймається рішення про

використання конкретної хромосоми або переходу до етапу удосконалення хромосом, використовуючи генетичні оператори схрещення та мутації. Для виконання задачі формування розкладу пропонується використання наступної цільової функції:

$$F(H_j) = w_p^{td} \times Q \times V_g / k_t \rightarrow \max$$

H_jj – оцінювана хромосома;

w_p^{td} – ваговий коефіцієнт викладача p, щодо проведення занять у td день тижня;

td = {1…Nd}- номер дня тижня, Nd – кількість робочих днів у тижні;

V_g – ваговий коефіцієнт, щодо наявності у td день тижня проміжних періодів у розкладі («вікон») для g групи;

k_t – кількість вікон у день td (якщо вікон немає то k_{td} = 1);

Q – бінарна змінна, яка побудована на основі жорстких обмежень і визначається наступним чином:

$$Q = y(t, g) \times x(t, p)$$

$$y(t, g) = \begin{cases} 0, \text{якщо у часовий період t у групи g триває заняття;} \\ 1, \text{в іншому випадку;} \end{cases}$$

$$y(t, g) = \begin{cases} 0, \text{якщо у часовий період t у групи g триває заняття;} \\ 1, \text{в іншому випадку;} \end{cases}$$

$$t = (t_w, t_d, t_{p)}$$

де t_w = {1…Nw}- номер тижня;

t_d = {1…Nd}- - номер дня тижня;

$t_{p)}$= {1…Np}- номер пари протягом дня.

Якщо результат цільової функції рівний 0, тоді, можна зробити висновок, що хоча б одне жорстке обмеження є невиконане, отже, досліджувана хромосома є непридатною для використання і потребує покращення своїх характеристик. Цільова функція застосовується для перевірки кожної з хромосом, орієнтована на знаходження максимального значення.

Для покращення характеристик хромосом використовують генетичні оператори. До генетичних операторів належать операції схрещування та мутації.

Схрещування - один з видів оператора зміни комбінації генетичного алгоритму. Метою даного оператора є покращення структури хромосом, утворення з існуючої множини нових рішень, де генетична особина буде нащадком деяких двох елементів попередньої популяції, відповідно, зберігати в собі інформації кожного батька.

Оператор мутації передбачає собою зміну набору генів у випадково вибраних хромосомах. Метою даного оператора є диверсифікація - підвищення різноманітності пошуку і введення в популяцію нових хромосом з метою більш глибокого дослідження. Мутація сприяє урізноманітненню популяції, що забезпечує можливість здійснення аналізу більшої кількості точок в просторі пошуку.

2.1.5 Загальний опис алгоритму підстановки

Внаслідок детального аналізу вимог, необхідних до виконання в процесі проектування навчального розкладу занять, було прийнято рішення про необхідність створення алгоритму із подальшою можливістю розширення списку вимог щодо складання розкладу, а також можливістю редагування пріоритетності з виконання окремих вимог та обмежень.

Фундаментальною ідеєю запропонованого алгоритму є оцінка свободи розташування певного заняття в розкладі. Було встановлено, що заняття, проведення яких потребує виконання обов’язкових вимог, мають меншу свободу розташування. Наприклад заняття, які вимагають наявність спеціального обладнання в аудиторіях можуть бути проведені лише в обмеженій кількості аудиторій, обладнаних певним чином під даний тип занять. Лекційні заняття можуть бути проведені лише в аудиторіях, де кількісний показних місткості аудиторії більший або дорівнює кількості

студентів в групі. Заняттям, проведення яких не вимагає спеціально обладнаних аудиторій присвоюється вищий показник свободи розташування. Таким чином, зменшення кількості обов’язкових вимог призводить до збільшення оцінки свободи розташування певного заняття в розкладі.

З метою найбільш оптимального розташування занять в розкладі доцільно розпочинати складання розкладу із додавання в нього занять, що мають найменшу оцінку свободи розташування. Робота алгоритму повинна базуватись на ідеї додавання занять із найменшою оцінкою свободи розташування, рухаючись до занять, що мають вищу оцінку свободи розташування.

Першим кроком алгоритму є зчитування попередньо-введених даних стосовно викладачів, групи студентів, аудиторії та дисципліни. У разі, якщо дані не введено, система пропонує користувачеві здійснити заповнення таблиць необхідними даними. Для введення даних передбачається ручний режим роботи. На рисунку 2.2 наведено структурну схему запропонованого алгоритму.


Введення даних, необхідних для роботи алгоритму
Зчитування даних
Формування списку занять
Оцінка свободи розташування занять в розкладі
Сортування списку занять за критерієм збільшення оцінки свободи розташування занять в розкладі
Пошук найбільш оптимального часу та аудиторії для проведення занять
Аналіз якості складеного розкладу занять
Редагування складеного розкладу занять за необхідності

Рисунок 2.2 – Структурна схема алгоритму

В результаті обробки даних відбувається формування списку занять. Приклад генерованого списку занять наведено в таблиці 2.1.

Таблиця 2.1 – Частина згенерованого списоку занять

Група	Назва дисципліни	Тип заняття	Викладач	Вимоги до аудиторії
МІТ - 11	Вступ до фаху	Лекція	Кравченко Ю.В	>= 25 осіб
МІТ - 11	Вступ до фаху	Практика	Кравченко Ю.В	-
МІТ - 21	Теорія алгоритмів	Лекція	Мухін	>= 23 осіб

З наведеного вище прикладу списку занять видно, що можливе повторення дисциплін. Таке явище спричинене тим, що протягом тижня передбачається проведення двох занять однакового типу.

В процесі оцінки свободи розміщення і-го заняття в розкладі відбувається підрахунок:

- кількість аудиторій ai, які підходять для проведення заняття, базуючись на вимогах заняття стосовно обладнання та місткості;
- кількість занять на тиждень з певної дисципліни pi, для заданої групи студентів;
- загальна кількість занять на тиждень gi для заданої групи студентів;

На основі наведених даних визначається оцінка свободи розташування заняття в розкладі:

$$Si = \frac{a_i}{g_i \times p_i}$$

S_i – оцінка свободи розміщення і-го заняття в розкладі;

a_i – кількість аудиторій які підходять для проведення заняття;

p_i – кількість занять на тиждень, з певної дисципліни для заданої групи студентів);

g_i – загальна кількість занять на тиждень для заданої групи студентів.

Наступним етапом є сортування списку занять за критерієм збільшення оцінки свободи розташування заняття в розкладі.

$$Si < Si + 1, i = 1..n.$$

В процесі додавання кожного заняття до розкладу відбувається підбір найбільш вигідної аудиторії та часу (номер пари, день тижня) для проведення відповідного заняття. Для виконання такого підбору необхідно виконати перевірку за заданими критеріями:

- аудиторія вільна. У випадку, якщо аудиторія зайнята, проведення іншого заняття в той самий час є неможливим;
- аудиторія обладнана відповідно до заняття, яке в ній буде проводитись;
- показник місткості аудиторії більший або дорівнює кількості студентів в групі;

У випадку, якщо виконані всі обов'язкові умови, відбувається оцінка якості розміщення занять згідно з наступними критеріями:

- наявність вільної пари у розкладі групи студентів;
- наявність вільної пари у розкладі викладачів;
- зайва кількість місць в аудиторії відносно до кількості студентів в групі;
- вільна пара у розкладі використання аудиторій;

Оцінка якості розміщення занять за списком перелічених критеріїв може бути використана з метою отримання загальної оцінки якості розміщення занять, що необхідна для подальшого вибору найбільш оптимального місця і часу для проведення занять. Для отримання оцінки якості розміщення занять використано наступну формулу:

$$R_{il} = \sum_{j=1}^{m} w_j k_{jl,}$$

Де R_{il} – якість розміщення і-го заняття на позиції l в розкладі;

K_{jl} – значення, отримане згідно з критерієм j оцінки якості розміщення заняття на позиції l в розкладі;

w_j – ваговий коефіцієнт оцінки якості за критерієм j;

m – кількість критеріїв оцінки якості розміщення занять.

Після закінчення етапу оцінки якості всіх можливих варіантів розміщення занять в розкладі, відбувається вибір варіанту, за виконання якого досягається максимальне значення оцінки якості розміщення:

$$\mathrm{R_i} = max(\mathrm{R_{il}}), l = 1..h,$$

де l – можлива позиція заняття i в розкладі;

$\mathrm{R_i}$ – якість розміщеного заняття в розкладі;

h – кількість можливих варіантів розміщення даного заняття в розкладі.

Наступним етапом, після розміщення всіх занять в розкладі, є підрахунок оцінки якості складеного розкладу. Для загальної оцінки якості розкладу вирішено використовувати суму всіх оцінок якості розміщення кожного заняття. З метою отримання загальної оцінки якості розкладу використовується формула:

$$R = \sum_{i=1}^{n} R_{i,}$$

де R – оцінка якості складеного розкладу;

n – кількість занять.

2.2 Вибір моделі даних

Базою даних називають систему даних, а також зв’язків між ними, масив інформації різного типу, систематизованої за певними критеріями. Такі

критерії передбачають використання деяких принципів опису, збереження, а також обробки даних.

Розрізняють три основні типи баз даних, а саме: ієрархічні, мережеві та реляційні.

Ієрархічна модель даних істотно відрізняється від інших двох типів. Для її побудови застосовується певний принцип. Модель ієрархічної бази даних має деревовидну структуру, дані в такій системі зберігають у вигляді двійкового дерева. Кожен вузол структури має певний відповідний сегмент. Термін “сегмент” в даному випадку означає поля даних з присвоєним іменем для кожного поля. Такі поля, як правило, збудовані в один лінійний кортеж. Для кожного структурного елементу існує лише одне місце в ієрархічній системі. Розпочинає деревовидну структуру кореневий елемент. Таким чином, на першому рівні такої структури може розташовуватись один і тільки один елемент. Для кожного підлеглого вузла існує лише один предок. У кожного предка може бути кілька нащадків. Доступ до кожного дочірнього вузла є унікальним і здійснюється через його вузол-предок. Таким чином ієрархічна модель даних підтримує тільки лінійні шляхи доступу.

Переваги ієрархічної моделі даних:

- простота структури побудови БД;

Недоліки ієрархічної моделі даних:

- обмеженість числа зв’язків у вершині структури;
- необхідність перебудови структури БД за необхідності введення нових даних;
- ймовірність зберігання зайвих даних;
- складність реалізації операцій видалення та додавання;
- видалення вузла-предка спричиняє видалення дочірніх вузлів;
- складність реалізації зв’язків типу N:N;
- ускладнений доступ до вузлів нижніх рівнів організації.

Мережева модель даних - це логічна модель даних, що є доповненням до ієрархічного моделі даних. База даних, побудована за принципом мережевої моделі, складається з системи елементів певного типу запису, а також відповідного типу зв’язків між цими записами. Тип зв’язку визначається для типу запису “предок” та типу запису “нащадок”. Мережева модель має ті ж самі основні складові, що і ієрархічна модель, а саме: вузол, рівень, зв'язок. Проте, характер відносин між цими складовими суттєво відрізняється. На відміну від ієрархічної моделі, в мережевій моделі прийнятним є вільний зв'язок між елементами різних рівнів. Кожен сегмент бази даних мережевої структури може бути пов'язаний з будь-яким іншим сегментом.

Переваги мережевої моделі даних:

– ефективна реалізація за показником витрати пам’яті;

– ефективна реалізація за показником ефективності.

Недоліки мережевої моделі даних:

– складність структури моделі даних;

– складність зберігання й пошуку інформації про всі зв'язки;

– складність структури БД, побудованої на основі мережевої моделі.

Реляційна модель даних – є логічною моделлю даних. База даних, генерована із застосуванням концепції реляційної моделі даних, складає систему нормалізованих відношень різного рівня. Реляційна база даних є організованою сукупністю сегментів даних, що організовані методом формального опису таблиць. Реляційна модель даних забезпечує найвищий ступінь абстракції даних в порівнянні з ієрархічною та мережевою моделями. Також в реляційної моделі відсутнє поняття групових відносин. Відображення асоціацій між кортежами різних відносин відбувається за рахунок копіювання їх ключів. Основною особливістю даної моделі є простота структури даних, зручні для користувача табличні представлення. Це тип бази даних, що здійснює пошук даних в одній таблиці на підставі визначених ключових полів іншої таблиці. На відміну від мережевої та ієрархічної моделей даних,

реляційна модель бази даних звільняє користувача від виконання рутинних операцій адміністрування даних. Частина полів таблиць зберігає дані, що стосуються безпосередньо запису, а частина – посилання на записи інших таблиць. Отже, зв'язки між записами є невід'ємною властивістю реляційної моделі. Зміна запису в одній з таблиць не призведе до порушення коректної роботи програми. у випадку створення запитів немає необхідності знати конкретну організацію БД у зовнішній пам’яті, достатньо розуміти зв’язки між таблицями даних. Великою перевагою є також те, що зміни в прикладній програмі при зміні реляційної БД мінімальні. Сутності (таблиці) реляційної бази даних є двовимірними, що зумовлює простоту роботи в процесі виконання математичних та логічних операцій. Порядок запису значень атрибутів, а також записів а таблицях може бути довільним. Стовпець в межах однієї реляційної таблиці повинен зберігати дані однакового типу. Складові компоненти рядків у БД реляційного типу не можуть повторюватися. Кожному стовпцеві реляційної таблиці має бути присвоєно індивідуальне ім’я. Основна інформаційна конструкція – таблиця.

Переваги реляційної моделі даних:

- простота структури побудови БД;
- простота доступності до елементів БД;
- незалежність даних;
- суворі правила проектування;
- у випадку створення запитів немає необхідності знати конкретну організацію БД у зовнішній пам’яті.

Недоліки реляційної моделі даних:

- ймовірна можливість важкості організації предметної області у вигляді таблиць;
- велика кількість реляційних таблиць може спричинити складну структуру;
- відносно великий об’єм використовуваної пам’яті;

– відносно низька швидкість доступу до даних.

Отже, детально проаналізувавши різні моделі баз даних, можна зробити висновок, що для проектування автоматизованої системи створення навчального розкладу занять, найбільш доречно використовувати реляційну модель даних.

2.3 Проектування бази даних

База даних – це систематизована структура, що виконує функцію накопичення, зберігання, зміни та обробки взаємопов’язаної інформації. Проектуванням бази даних називається процес побудови схеми БД, а також визначення необхідних обмежень цілісності. При проектуванні бази даних основними завданнями є:

– забезпечення зберігання в БД певної вибірки даних;
– забезпечення отримання даних внаслідок виклику всіх необхідних запитів;
– зменшення ймовірності дублювання даних;
– забезпечення цілісності БД.

Проектування бази даних включає в себе три етапи:

– інфологічне (концептуальне) проектування;
– логічне проектування;
– фізичне проектування.

Для проектування коректної БД, виконання кожного з наведених етапів є обов’язковим.

2.3.1 Проектування концептуальної моделі бази даних

Фундаментальним етапом під час проектування БД є розробка її концептуальної моделі. Концептуальна модель застосовується на другому

етапі проектування бази даних, після опису предметної області і визначення мети та призначення ІС. Концептуальна модель повинна включати формалізований опис предметної області, який буде зрозумілий не тільки для фахівця з баз даних, а й для сторонніх людей.

Проектована база даних повинна забезпечувати не лише можливість акумуляції та збереження даних, але й дозволяти формувати необхідні вихідні дані в результаті запитів до неї. На інфологічному рівні здійснюється інтегрований опис предметної області, для якої проектується база даних. Даний етап дозволяє окреслити межі предметної області, визначити її основні сутності, зв’язки між сутностями. Така семантична модель будується без орієнтиру на конкретну систему управління базами даних. Етап семантичного проектування передбачає опис предметної області, використовуючи терміни формальної мови. Існує безліч моделей для представлення знань в процесі концептуального проектування. Одним із найпопулярніших та найзручніших інструментів уніфікованого представлення даних, автономного від програмного забезпечення, є модель “сутність-зв'язок” (ER-модель). Це модель даних, за допомогою якої відбувається опис концептуальної схеми за допомогою узагальненої та спрощеної конструкції блоків. До основних переваг ER-моделі належить:

- моделі дозволяють проектування баз даних, що вміщують велику кількість об’єктів та атрибутів;
- наочність;
- ER-моделі реалізовані в багатьох системах автоматизованого проектування БД.

Елементами ER-моделі є:

- сутності(об’єкти);
- атрибути об’єктів;
- зв’язки між об’єктами.

Системний аналіз предметної області, проведений в минулих розділах даної роботи, дозволяє здійснити побудову концептуальної моделі бази даних автоматизованої системи складання розкладу навчальних занять. На рисунку 2.3 представлена діаграма “сутність-зв’язок” для проектування БД досліджуваної предметної області.

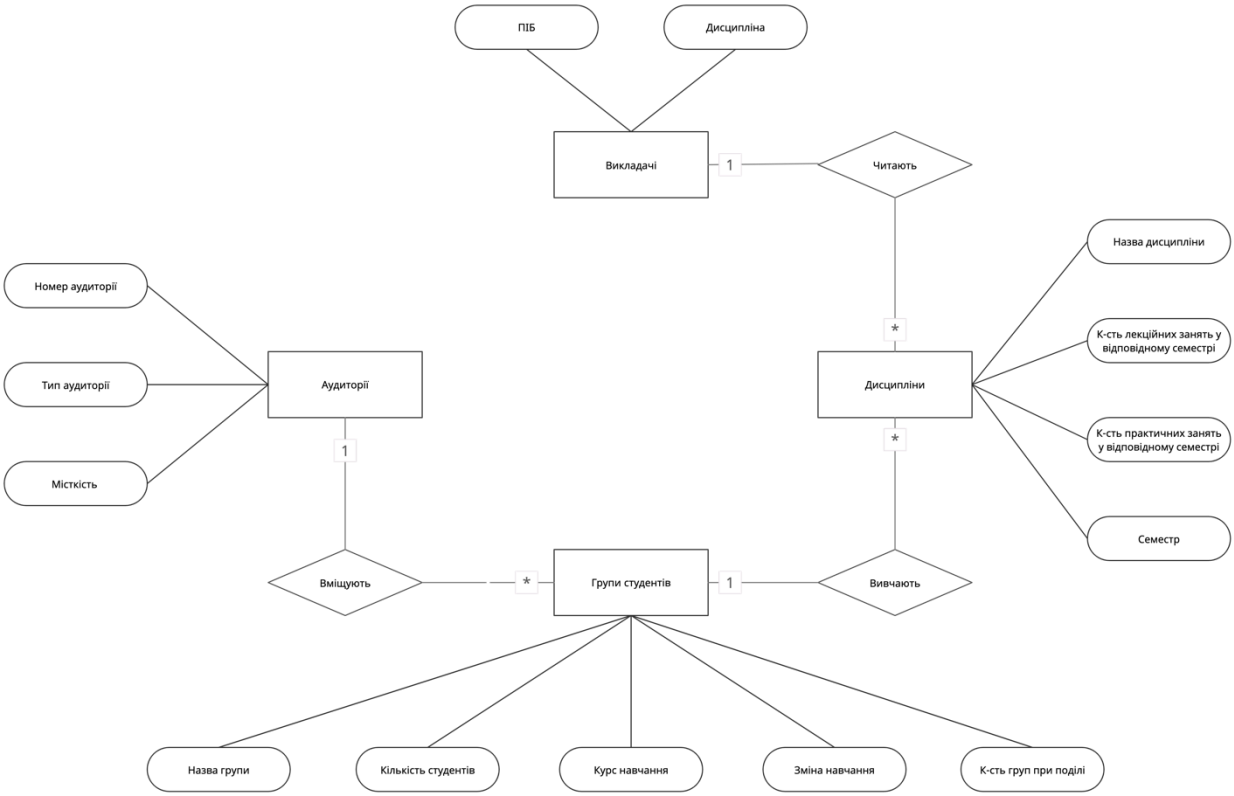


Рисунок 2.3 - Діагама “сутність-зв’язок” для БД

2.3.2 Проектування логічної моделі бази даних

Логічне моделювання - проектування схеми бази даних на основі певної моделі даних, наприклад, реляційної моделі БД. Логічним проектуванням для реляційної моделі є формування набору сутностей з вказаними первинними ключами, а також зв’язками між цими сутностями, що утворюють зовнішні ключі. Основою для побудови логічної моделі слугує концептуальна модель даних. В процесі створення логічної моделі важливо враховувати особливості

конкретної моделі даних. Врахування специфіки конкретної СУБД не є обов’язковим.

Логічна модель розширює концептуальну завдяки визначенню для кожної сутності атрибутів, описує та уточнює склад сутностей та відношення між ними. На рисунку 2.4 наведено результат проектування логічної моделі бази даних.

Логічне проектування використовується для перенесення концептуального (інфологічного) проектування на внутрішню модель обраної СУБД. Для реляційної СУБД логічне проектування є проектуванням таблиць. Основою аналізу коректності схеми являються функціональні залежності між атрибутами БД.

Перший етап побудови логічної моделі – визначення атрибутів для кожної сутності.

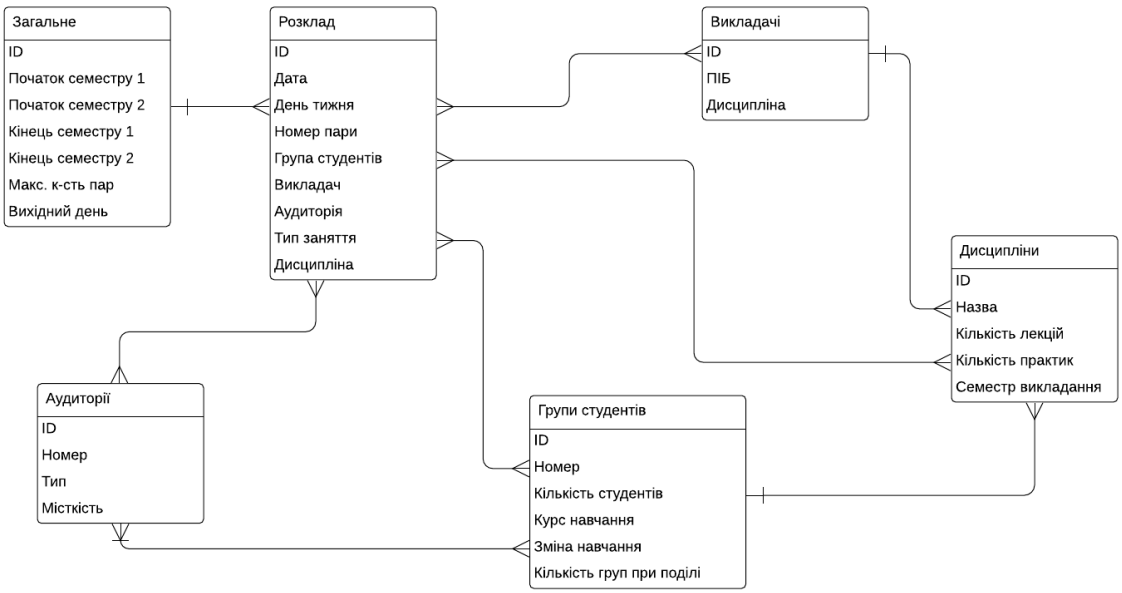


Рисунок 2.4 – Логічна модель інформаційної системи

Основними сутностями бази даних є викладачі, групи студентів, навчальні дисципліни та аудиторії.

До сутності “Викладачі” належать два атрибути: ПІБ викладача, а також дисципліна, яку читає викладач. Сутність “Дисципліни” містить інформацію стосовно назви дисципліни, кількості лекційних занять у відповідному семестрі та кількість практичних занять у відповідному семестрі. Сутність “Групи студентів” містить дані про назву групи, кількість студентів, що навчається в даній групі, курс навчання, зміна навчання (інформаційна система припускає наявність двох змін навчання), кількість груп при поділі для певних дисциплін. Сутність “Аудиторії” містить інформацію про номер аудиторії, тип аудиторії (лекційна чи практична), а також місткість аудиторії. Створення сутностей “Розклад” та “Загальне” відбувається динамічно в процесі роботи інформаційної системи. Таблиця “Загальне” вміщує службову інформацію, необхідну для створення розкладу занять. До такої інформації належить дата початку та закінчення першого семестру, тата початку та закінчення другого семестру, максимальна кількість пар на день для однієї групи студентів. До даної таблиці також можна вносити дані стосовно вихідних днів у конкретному робочому році. У таблиці БД “Розклад” відбувається побудова розкладу занять. Дана таблиця складається з наступних полів: дата, день тижня, номер пари, група студентів, викладач, аудиторія, тип заняття, дисципліна.

Логічна модель відображає абстрактну структуру інформаційної системи. Основними причинами проектування логічної структури даних є:

- відображає загальну концепцію системи сутностей певної предметної області;
- слугує фундаментом для створення бази даних;
- знижує надмірність даних, що допомагає уникнути неузгодженості даних;
- сприяє повторному використанню даних;
- полегшує розробку да подальшу підтримку бази даних.

2.3.3 Проектування фізичної моделі бази даних

Побудова фізичної моделі даних відбувається на базі попередньо проектованої логічної моделі даних. Основними компонентами фізичної моделі даних є таблиці, колонки разом зі своїми значеннями, зовнішні та первинні ключі, зв’язки між таблицями.

Фізична модель даних описує реалізацію об’єктів логічної моделі на рівні об’єктів конкретної бази даних у вибраній СУБД. У логічній моделі не має значення, який тип даних має атрибут. На відміну від логічної моделі даних, у фізичній моделі даних необхідно визначити всю інформацію про конкретні фізичні об’єкти – таблиці, колонки, індекси, тощо. На рисунку 2.5 наведено результат проектування фізичної моделі бази даних.

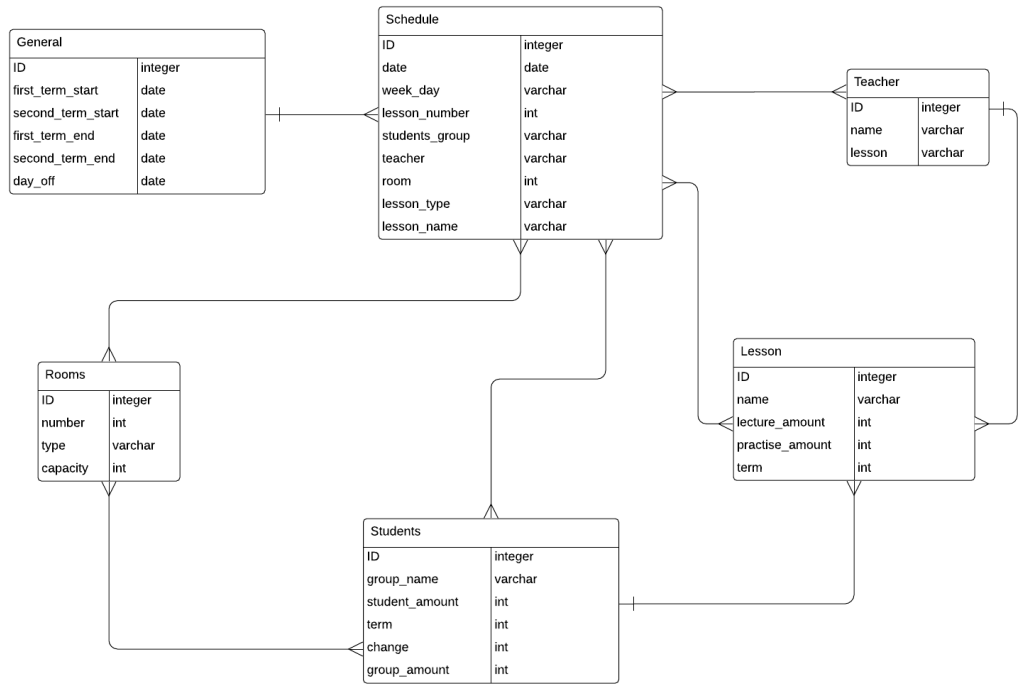


Рисунок 2.5 – Фізична модель інформаційної системи

Для реалізації переходу від логічної моделі даних до фізичної моделі даних, необхідно виконати наступний алгоритм:

– кожну сутність поставити у відповідність відношення;

- кожному атрибуту сутності відповідає окремий атрибут відповідного відношення;
- первинний ключ сутності являється PK відповідного відношення. Атрибути, які належать PK, є обов'язковими для заповнення (NOT NULL).
- У сутності, які являються підпорядкованими іншим сутностям, додається набір атрибутів основної сутності, які являються для неї первинним ключем. Ці атрибути в підпорядкованій сутності стають зовнішніми ключами FK.
- Відношення типу «багато-до-багатьох повинні бути розкритими. Такі відношення необхідно замінити «один-до-багатьох», створивши додаткову таблицю, яка буде пов'язувати дві такі сутності.

2.4 Вибір системи управління базами даних

Для роботи та маніпуляції над базами даних використовують систему управління базами даних. Програмне забезпечення СУБД функціонує насамперед як інтерфейс між кінцевим користувачем та базою даних, одночасно керуючи даними, механізмом бази даних та схемою бази даних, щоб полегшити організацію та маніпулювання даними. Хоча функції СУБД сильно різняться, загальні функції та можливості СУБД повинні включати: доступний користувачеві каталог, що описує метадані, систему управління бібліотекою СУБД, абстракцію та незалежність даних, безпеку даних, ведення журналу та аудит діяльності, підтримку паралельності та транзакцій, підтримку для авторизації доступу, підтримки доступу з віддалених місцеположень, підтримки відновлення даних СУБД у разі пошкодження та забезпечення обмежень для забезпечення відповідності даних певним

правилам. Найпопулярнішими системами управління базами даних вважаються Oracle, MySQL, PostgreSql.

2.4.1 Система управління базами даних Oracle

Oracle – об'єктно-реляційна система керування базами даних від Oracle Corporation.

До переваг даної СУБД належать:

- централізована система контролю та управління;
- чітка стандартизація;
- клієнт-серверне середовище (можливість відокремлювати сервер бази даних від програм, я якими працює користувачем. Такий клієнт-серверний підхід забезпечує швидкість роботи бази даних);
- можливість одночасного звернення до бази даних багато користувачів;
- можливість розгортання на різних операційних системах;
- високий рівень безпеки даних;
- доступність даних;
- висока ефективність роботи з транзакціями.

Недоліки СУБД Oracle:

- високі технічні характеристики, що спричиняють складність при встановленні та подальшому використанні СУБД;
- неможливість реалізації рекурсивної обробки даних;
- висока вартість ліцензії (Oracle Database 11g Standard Edition – приблизно 6000 доларів США);
- значні затримки обробки даних при збільшенні їх об’єму.

2.4.2 Система управління базами даних MySQL

MySQL - це найбільш поширена повноцінна серверна СУБД. MySQL є дуже функціональною та вільно розповсюджуваною, успішно працює з різними сайтами і веб додатками. До переваг даної СУБД належать:

- швидке функціонування;
- відкритий код;
- легке встановлення та адміністрування;
- присутня система контролю доступу даних, що забезпечує шифрування даних при передаванні;
- можливість розгортання на різних операційних системах;
- високий рівень масштабованості. MySQL здатна підтримувати функціонування БД значних розмірів.

Недоліки СУБД MySQL:

- багатопоточність – паралельні операції читання запису можуть спричиняти проблеми у роботі СУБД;
- повільна розробка;

2.4.3 Система управління базами даних PostgreSQL

PostgreSQL - популярна об'єктно-реляційна система управління базами даних. PostgreSQL вільно розповсюджується і максимально відповідає стандартам SQL. Переваги СУБД PostgreSQL:

- відкрите ПЗ відповідає стандарту SQL PostgreSQL – безкоштовне ПЗ з відкритим вихідним кодом;
- велике співтовариство. Існує широка спільнота, в якій можна відшукати відповіді на питання, що виникають в процесі роботи;
- велика кількість доповнень - існує дуже багато доповнень, що спрощують управління над СУБД;
- можливість розширення функціоналу за рахунок можливості збереження власних процедур;

– об'єктна орієнтованість – PostgreSQL це не тільки реляційна СУБД, але також і об'єктно-орієнтована з підтримкою успадкування і багато іншого.

Недоліки PostgreSQL:

– можливе уповільнення швидкості роботи СУБД;
– складний процес налаштування перед початком роботи із СУБД.

Для проектування автоматизованої системи створення навчального розкладу занять було використано СУБД MySQL.

2.5 Проектування базового інтерфейсу користувача

Інтерфейс користувача – інструмент, основним завданням якого є забезпечення найбільш зручної взаємодії між користувачем та інформаційною системою. ІК є комплексом апаратних та програмних засобів. Завдяки ІК користувач має доступ до використання функціоналу програми. ІК забезпечує підтримку прийняття рішень у певній предметній галузі, а також визначає порядок використання програмного забезпечення та документації до нього.

З метою створення найбільш зручного для користувача інтерфейсу необхідно дотримуватися базових принципів його побудови. До списку таких принципів належить:

– інтуїтивність. Для управління процесом роботи з програмним продуктом всі елементи інтерфейсу повинні працювати на інтуїтивному для користувача рівні;
– несуперечливість. Наявність ідентичних методів роботи для виконання однакових операцій в усіх частинах інтерфейсу;
– ненадмірність. Мінімальне навантаження користувача для взаємодії з системою;
– забезпечення користувача всіма необхідними інструкціями в процесі роботи з програмним продуктом;

– гнучкість. Комбінування застосування всіх елементів інтерфейсу з метою досягнення його найвищого рівня зручності, забезпечення можливості вибору, наприклад, можливість використання клавіатури замість миші.

Можливість адаптації системи до роботи користувача реалізується за рахунок можливості приймати, аналізувати та виконувати запити. Природньо-мовний інтерфейс виступає у ролі посередника між людиною та БД. На рисунку 2.6 представлено схему потоків даних природньо-мовного інтерфейсу.

Рисунок 2.6 – Схема потоків даних природньо-мовного інтерфейсу

Природньо-мовний інтерфейс приймає запити, що надходять звичною для користувача мовою та переводить їх у формальне представлення, яке сприймається базою даних. БД обробляє запити, представлені у формі SQL та за допомогою необхідних алгоритмів, віддає результат.

Веб-інтерфейс - це сукупність елементів, за допомогою яких користувач взаємодіє з веб-сайтом або будь-яким іншим додатком через браузер. Веб-інтерфейси набули широкого поширення в зв'язку з ростом популярності всесвітньої павутини Інтернет. Класичним методом створення веб-інтерфейсу є застосування HTML, CSS та Java Script.

2.5.1 Використання HTML для створення веб-сторінки

HTML – мова розмітки гіпертексту. За допомогою цієї мови відбувається реалізація веброзкладусторінки для мережі Інтернет. Браузер отримує доступ до документа HTML через веб-сервер або з локальної пам’яті комп’ютера. Структура веб-сторінки реалізовується семантично за допомогою засобів HTML.

HTML забезпечує впровадження засобів для:

- побудови інтерактивних форм;
- імплементація фото, відео, аудіо та інших об’єктів у веб-сторінку;
- побудова структурованого документу за умови семантичного позначення структурованого складу тексту (списки, заголовки, абзаци, цитати);
- імплементація гіперпосилань для можливості отримання інформації з мережі Інтернет.

Чотири основні компоненти утворюють розмітку в HTML: елементи та їх атрибути, базові типи даних, символьна мнемоніка, декларації типу документа. Доступ до елементів HTML сторінки відбувається за рахунок звернення до їх блоків через id та класи.

HTML5 – є новою специфікацією мови розмітки HTML, яка використовується для виконання завдання створення веб-сторінок. HTML5 вміщує в собі набір різнопланових модулів, що забезпечують імплементацію відео, векторної графіки формату SVG, растрової графіки, локальних баз даних, а також різних API браузера. HTML5 забезпечує мінімальне навантаження процесора, не вимагає установки плагінів та оновлень, а отже, є стійким до хакерських атак.

Основні переваги HTML:

- висока швидкість завантаження;
- простота;
- невелика кількість займаної пам’яті;
- незначне серверне навантаження.

Основні недоліки:

- оновлюваний контент знаходиться безпосередньо у файлі розмітки html (потрібно редагувати всю сторінку);
- негнучкість (для додавання нового розділу чи сторінки потрібно також найняти розробника.

2.5.2 Використання CSS для створення веб-сторінки

CSS відповідає за візуальне оформлення веб-сторінки, написаної на HTML. CSS базується на принципі блочної або каскадної верстки. Документ CSS може містити інформацію про стиль для усього сайту або лише до його певної частини. Це дозволяє швидко змінювати стиль сайту за необхідності. Існує також можливість розмежовувати варіанти стильового оформлення для різних користувачів сайту.

Синтаксис CSS є відносно постим. Для визначення стилю необхідно описати список правил. Щоб прив’язати такий список до певного елемента сторінки використовують селектори. Визначення блоку реалізується за рахунок оточеного фігурними дужками списку його властивостей.

Також CSS значно покращує час завантаження та трафік, завдяки зберіганню інформації про форматування в окремому файлі конфігурації, який завантажується лише один раз, на відміну від сайту без CSS, тож і переформатувати документ html можна за допомогою CSS можна значно швидше, адже ми маємо змінити лише код CSS і усі документи html, що використовують дану таблицю стилей автоматично будуть переформатовані.

2.6 Вибір мови програмування

Важливим кроком в розробці будь-якої системи є вибір мови програмування. У сучасному високотехнологічному світі PHP та Python стали найпопулярнішими серед серверних мов програмування.

Python - високорівнева об’єктно орієнтована мова програмування. Python властиві вбудовані структури даних, а також динамічна типізація та прив’язка. Завдяки таким властивостям Python програмування є максимально швидким. Python також властива підтримка модулів та пакетів, що в свою чергу дозволяє користувачеві використовувати модульність системи та повторно використовувати код. Перевагами Python є:

- простий код (програми, написані на Python, мають легкий синтаксис, що дозволяє мінімізувати час, необхідний для розробки, а також полегшує процес виявлення помилок в програмі);
- відсутність зайвих задач (автоматичне видалення об’єктів, до яких немає доступу);
- кросфункціональність (підтримка Python вбудована в різні програмні платформи та операційні системи, мову можна інтегрувати з Java, C і C ++);
- наявність бібліотек та можливість їх швидкого підключення. Недоліки Python:
- низька швидкість (операції на Python можуть виконуватися паралельно, що призводить до уповільнення швидкості їх виконання, а також потребують більше пам’яті);
- динамічна типізація (процес динамічної типізації призводить до споживання більшої кількості ресурсів, ніж у мовах без наявності динамічної типізації);

PHP одна з найпопулярніших мов програмування, що використовуються у сфері веб-розробок. З використанням даної мови часто створюють сайти та веб-додатки. PHP інтегрується практично з усіма веб-серверами. PHP є проектом відкритого програмного забезпечення, здатна працювати з усіма

операційними системами.

Перевагами PHP є:

- високий рівень продуктивності;
- можливість взаємодії з різним платформами;
- масштабованість (зріст продуктивності системи у наслідок додавання апаратних ресурсів)
- динамічний розвиток мови програмування;
- велика спільнота;
- детальна документація.

До списку недоліків PHP належить:

- не систематизований синтаксис (схожі назви функцій, наявність командних елементів, запозичених з мов програмування C та Java);
- можливість наявності помилки в коді (при наявності помилки в коді, PHP все одно дозволяє виконувати програму).

Отже, проаналізувавши найпопулярніші серверні мови програмування, можна зробити висновок, що для проектування автоматизованої системи створення навчального розкладу занять, найбільш доцільно використовувати серверну мову програмування PHP.

Висновки до розділу ІІ

Отже, в процесі виконання розділу було розглянуто основні методи розв’язання задачі формування розладу занять, а саме: методи цілочислового програмування, метод імітації відпалу, метод розфарбування графу, генетичний алгоритм, наведено загальний опис алгоритму підстановки.

Проаналізувавши різні моделі баз даних, було встановлено, що для проектування автоматизованої системи створення навчального розкладу занять, найбільш доречно використовувати реляційну модель даних. Здійснено побудову концептуальної, логічної та фізичної моделей даних.

Проаналізовано такі системи управління базами даних як Oracle, MySQL та PostgreSql. В результаті аналізу для проектування автоматизованої системи створення навчального розкладу занять було використано СУБД MySQL.

Розглянуто методи проектування інтерфейсу користувача та базові структури для створення інтерфейсу користувача, а саме: мова розмітки гіпертексту HTML, каскадна таблиця стилів CSS.

Проаналізувавши найпопулярніші серверні мови програмування, а саме: Python та PHP, було зроблено висновок, що для проектування автоматизованої системи створення навчального розкладу занять, найбільш доцільно використовувати серверну мову програмування PHP.

ІІІ РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ФОРМУВАННЯ НАВЧАЛЬНОГО РОЗКЛАДУ ЗАНЯТЬ

3.1 Створення таблиць

Для створення таблиць всередині бази, а також роботи з ними використовують мову SQL. Існує два способи створення таблиць:

- використання візуального інтерфейсу СУБД для інтерактивного створення таблиць та маніпуляції з ними;
- написання запитів, використовуючи оператори SQL.

В даному проекті генерація та модифікування таблиць виконувались за умови використання візуального інтерфейсу СУБД. Варто зазначити, що під час використання інтерактивного інструменту СУБД, створення та редагування таблиць відбувається так само за умови виконання запиту до БД, всі запити система формує самостійно на основі взаємодії користувача з візуальним інтерфейсом.

Створення таблиць в БД відбувається на основі фізичної моделі даних, спроектованої в попередніх розділах даного проекту. В процесі виконання даного кроку, в БД створено такі таблиці: “Timetable”, “Lessons”, “Teachers”, “Classrooms”, “Students”, “General”. На рисунках 3.1 – 3.4 відображено результат створення таблиць “Classrooms”, “Lessons”, “Teachers”, “Students” шляхом використання візуального інтерфейсу СУБД.

	#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1	id	int(11)			Ні	*Немає*		AUTO_INCREMENT	
☐	2	number_room	int(3)			Ні	*Немає*			
☐	3	type_room	varchar(16)	utf8_general_ci		Ні	*Немає*			
☐	4	capacity	int(2)			Ні	*Немає*			

☐ Перевірити все *Вибрані:*

Рисунок 3.1 – Результат створення таблиці “Classrooms”

	#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1	id	int(11)			Ні	*Немає*		AUTO_INCREMENT	
☐	2	turn	tinyint(1)			Ні	5			
☐	3	name	varchar(256)	utf8_general_ci		Ні	*Немає*			
☐	4	semestr	tinyint(1)			Ні	*Немає*			
☐	5	lectures	int(3)			Так	*NULL*			
☐	6	practical	int(3)			Так	*NULL*			
☐	7	id_introduced	tinyint(1)			Ні	0			

Перевірити все *Вибрані:*

Рисунок 3.2 – Результат створення таблиці “Lessons”

	#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1	id	int(11)			Ні	*Немає*		AUTO_INCREMENT	
☐	2	surname	varchar(50)	utf8_general_ci		Ні	*Немає*			
☐	3	subject_1	varchar(256)	utf8_general_ci		Ні	*Немає*			
☐	4	subject_2	varchar(256)	utf8_general_ci		Ні	*Немає*			
☐	5	subject_3	varchar(256)	utf8_general_ci		Ні	*Немає*			
☐	6	subject_4	text	utf8_general_ci		Ні	*Немає*			
☐	7	subject_5	text	utf8_general_ci		Ні	*Немає*			

Перевірити все *Вибрані:*

Рисунок 3.3 – Результат створення таблиці “Teachers”

	#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1	id	int(11)			Ні	*Немає*		AUTO_INCREMENT	
☐	2	name_group	varchar(100)	utf8_general_ci		Ні	*Немає*			
☐	3	cours	tinyint(4)			Ні	*Немає*			
☐	4	number_students	tinyint(4)			Ні	*Немає*			
☐	5	changing	tinyint(4)			Ні	*Немає*	зміна		
☐	6	division	tinyint(4)			Ні	*Немає*	к-сть груп при поділі		

Перевірити все *Вибрані:*

Рисунок 3.4 – Результат створення таблиці “Students”

3.2 Заповнення таблиць даними

Заповнення таблиці даними відбувається шляхом взаємодії користувача з веб-інтерфейсом. Після введення користувачем інформації, яка необхідна для заповнення таблиць, відбувається зчитування системою введеної інформації та запис цієї інформації до БД у відповідні таблиці. На рисунку 3.5 – 3.8 відображено процес внесення користувачем даних до таблиць БД через взаємодію з веб-інтерфейсом.

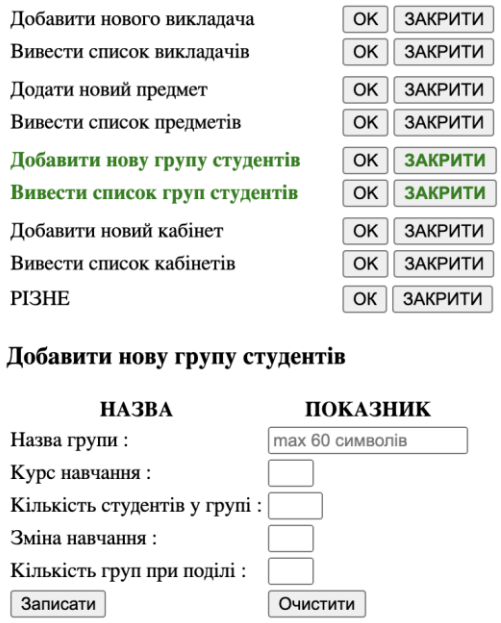


Рисунок 3.5 – Внесення даних до таблиці “Students” через взаємодію користувача з веб-інтерфейсом

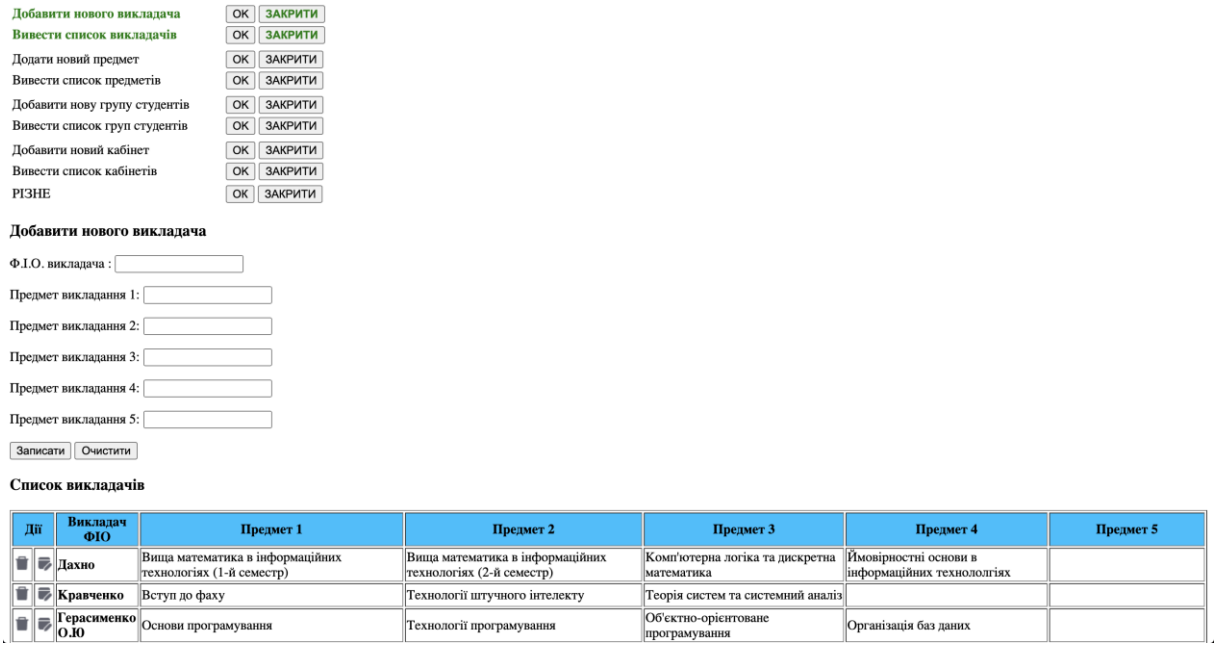


Рисунок 3.6 – Внесення даних до таблиці “Teachers” через взаємодію користувача з веб-інтерфейсом

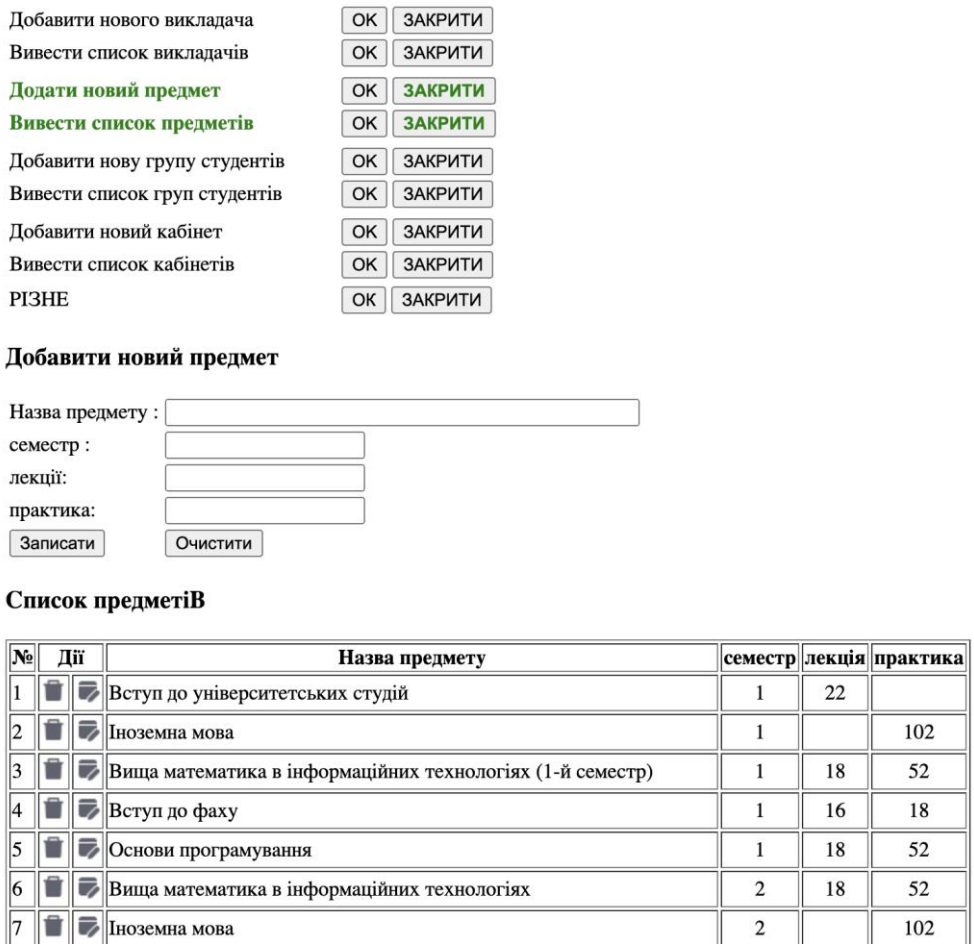


Рисунок 3.7 – Внесення даних до таблиці “Lessons” через взаємодію користувача з веб-інтерфейсом

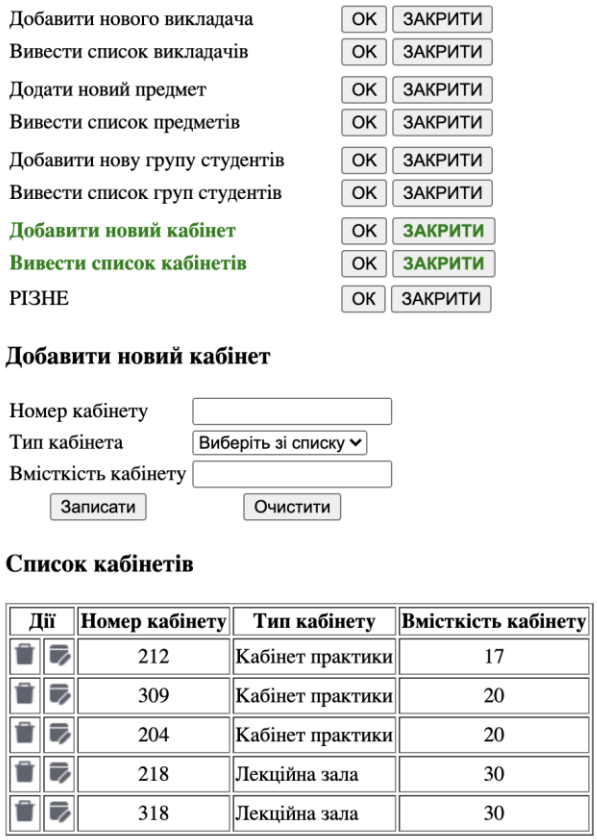


Рисунок 3.7 – Внесення даних до таблиці “Classrooms” через взаємодію користувача з веб-інтерфейсом

На рисунку 3.8 показано збереження даних, які ввів користувач через веб-інтерфейс, у відповідну таблицю БД.

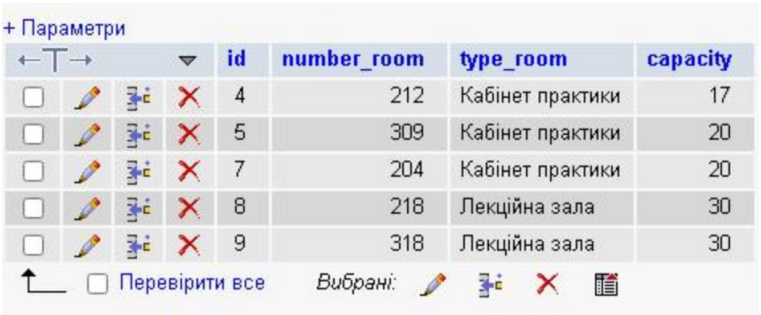


Рисунок 3.8 – Результат збереження даних в таблиці БД “Classrooms”

3.2 Проектування інтерфейсу користувача

Модуль підтримки користувацького інтерфейсу гарантує, зручну взаємодію користувача з інформаційною системою. Спираючись на права доступу, отримані від модуля авторизації, модуль інтерфейсу користувача забезпечує можливість введення, редагування та видалення даних, необхідних для побудови розкладу, а саме: перелік наявних аудиторії, перелік дисциплін, професорсько-викладацький склад, перелік груп студентів. Даний модуль також дозволяє здійснювати запуск визначеного алгоритму та відображення сформованих результатів.

Стилізація веб-сторінки відбувається за рахунок використання засобів CSS. Документ CSS вміщує інформацію про стиль для усього програмного продукту. Такий підхід дозволяє швидко змінювати стиль програми за необхідності. Існує також можливість розмежовувати варіанти стильового оформлення для різних користувачів сайту.

На рисунку 3.9 зображено стартове вікно інтерфейсу програми для автоматизації розкладу занять.

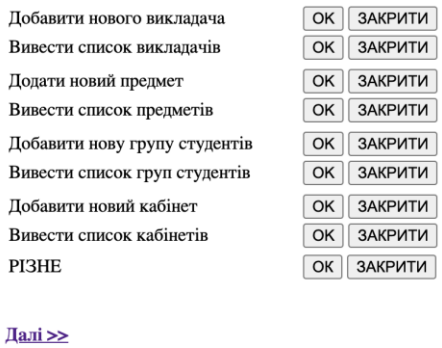


Рисунок 3.9 – Основне вікно інтерфейсу програми

На даному вікні зосереджено весь основний інструментарій, використання якого є необхідним для створення розкладу. Кнопки, розташовані на стартовому вікні інтерфейсу дозволяють здійснювати певну частину адміністрування базою даних, а саме додавати нового викладача, дисципліну, групу студентів та аудиторію до відповідних таблиць БД. Дане

вікно інтерфейсу містить також кнопки, що дозволяють здійснювати перегляд вищезгаданих таблиць БД.

При натисканні користувача на кнопку, системою передбачено переправлення користувача до окремого вікна інтерфейсу. На кожному вікні інтерфейсу доступний інструментарій, необхідний для внесення нових даних до таблиці БД, їх редагування та видалення. На рисунку 3.10 зображено приклад вікна інтерфейсу для перегляду таблиці БД “Групи студентів”.

Рисунок 3.10 – Вікно інтерфейсу для відображення списку груп студентів

За умови використання кнопок “Видалення” та “Редагування” користувачеві доступна можливість вилучити з таблиці БД певну групу студентів або ж змінити дані для групи. Реалізація інтерфейсу для перегляду списку викладачів, списку дисциплін та списку аудиторій виконано за аналогічним принципом.

На рисунку 3.11 зображено вікно інтерфейсу для додавання нової групи студентів до БД.

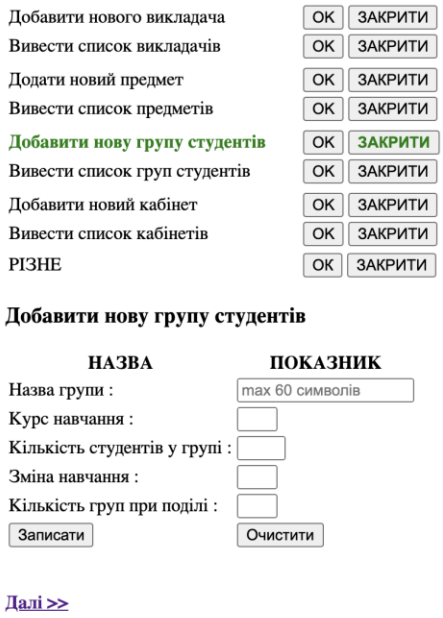
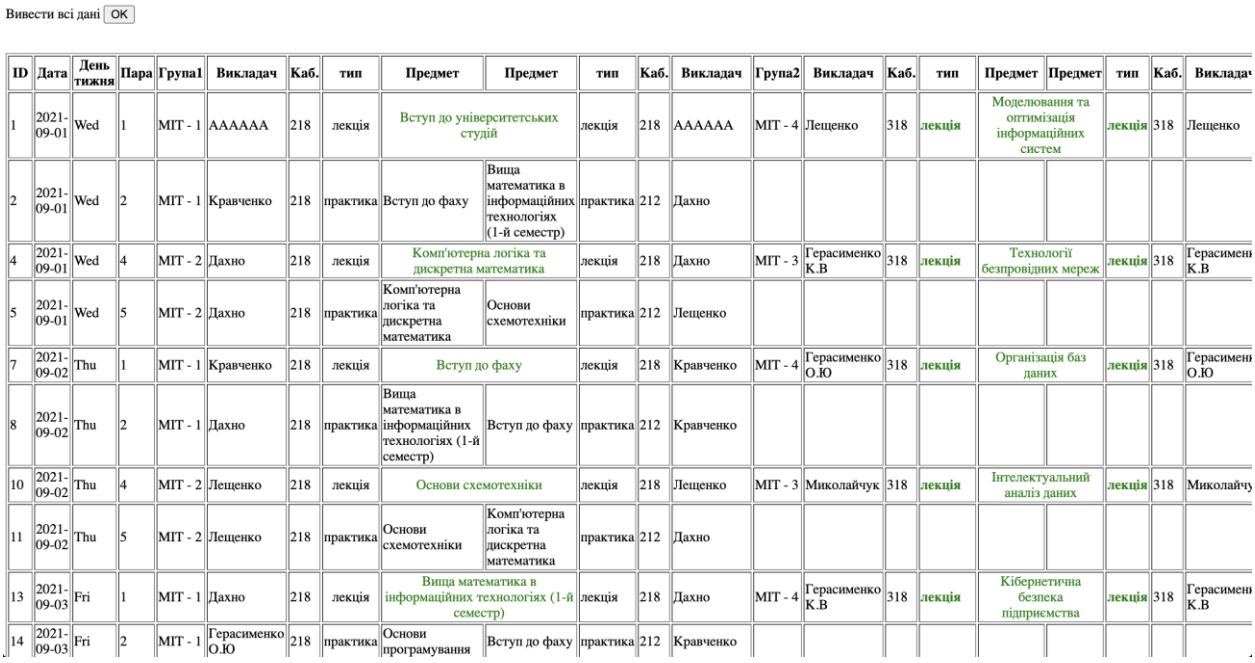


Рисунок 3.11 – Вікно інтерфейсу для введення нової групи студентів

Кнопка “Генерувати розклад” відповідає за безпосередню генерацію розкладу. Внаслідок натискання на цю кнопку відбувається запуск алгоритму, необхідного для генерування розкладу. Після успішної генерації, відбудеться виведення системою розкладу занять на окреме вікно інтерфейсу (Рисунок 3.12).



ID	Дата	День тижня	Пара	Група1	Викладач	Каб.	тип	Предмет	Предмет	тип	Каб.	Викладач	Група2	Викладач	Каб.	тип	Предмет	Предмет	тип	Каб.	Викладач
1	2021-09-01	Wed	1	МІТ - 1	ААААА	218	лекція	Вступ до університетських студій		лекція	218	ААААА	МІТ - 4	Лещенко	318	лекція	Моделювання та оптимізація інформаційних систем		лекція	318	Лещенко
2	2021-09-01	Wed	2	МІТ - 1	Кравченко	218	практика	Вступ до фаху	Вища математика в інформаційних технологіях (1-й семестр)	практика	212	Дахно									
4	2021-09-01	Wed	4	МІТ - 2	Дахно	218	лекція	Комп'ютерна логіка та дискретна математика		лекція	218	Дахно	МІТ - 3	Герасименко К.В	318	лекція	Технології безпровідних мереж		лекція	318	Герасименко К.В
5	2021-09-01	Wed	5	МІТ - 2	Дахно	218	практика	Комп'ютерна логіка та дискретна математика	Основи схемотехніки	практика	212	Лещенко									
7	2021-09-02	Thu	1	МІТ - 1	Кравченко	218	лекція	Вступ до фаху		лекція	218	Кравченко	МІТ - 4	Герасименко О.Ю	318	лекція	Організація баз даних		лекція	318	Герасименко О.Ю
8	2021-09-02	Thu	2	МІТ - 1	Дахно	218	практика	Вища математика в інформаційних технологіях (1-й семестр)	Вступ до фаху	практика	212	Кравченко									
10	2021-09-02	Thu	4	МІТ - 2	Лещенко	218	лекція	Основи схемотехніки		лекція	218	Лещенко	МІТ - 3	Миколайчук	318	лекція	Інтелектуальний аналіз даних		лекція	318	Миколайчук
11	2021-09-02	Thu	5	МІТ - 2	Лещенко	218	практика	Основи схемотехніки	Комп'ютерна логіка та дискретна математика	практика	212	Дахно									
13	2021-09-03	Fri	1	МІТ - 1	Дахно	218	лекція	Вища математика в інформаційних технологіях (1-й семестр)		лекція	218	Дахно	МІТ - 4	Герасименко К.В	318	лекція	Кібернетична безпека підприємства		лекція	318	Герасименко К.В
14	2021-09-03	Fri	2	МІТ - 1	Герасименко О.Ю	218	практика	Основи програмування	Вступ до фаху	практика	212	Кравченко									

Рисунок 3.12 – Виведення системою розкладу занять

3.3 Виконання тестування інформаційної системи на реальних даних

З метою перевірки функціонування програми в базу даних додатку було введено набір тестових даних. Даними заповнено таблиці “Викладачі” (Рисунок 3.13), “Групи студентів” (Рисунок 3.14), “Дисципліни” (Рисунок 3.15), “Аудиторії” (Рисунок 3.16). Було розглянуто перший та другий семестр чотирьох груп факультету (курс 1, 2, 3, 4). На рисунку 3.17 представлено процес автоматичного формування розкладу системою, таблиці БД якої заповнено реальними даними.

Дії	Викладач ФІО	Предмет 1	Предмет 2	Предмет 3	Предмет 4	Предмет 5
	Дахно	Вища математика в інформаційних технологіях (1-й семестр)	Вища математика в інформаційних технологіях (2-й семестр)	Комп'ютерна логіка та дискретна математика	Ймовірностні основи в інформаційних технололгіях	
	Кравченко	Вступ до фаху	Технології штучного інтелекту	Теорія систем та системний аналіз		
	Герасименко О.Ю	Основи програмування	Технології програмування	Об'єктно-орієнтоване програмування	Організація баз даних	
	Мухін	Теорія алгоритмів	Захищені інформаційні технології			
	Труш	Електротехніка та електроніка	Захист інформації в інформаційних системах			
	Герасименко К.В	Основи побудови інфокомунікаційних мереж	Технології та протоколи мультисервісних мереж	Технології безпровідних мереж	Кібернетична безпека підприємства	Інфраструктура мереж майбутнього
	Лещенко	Основи схемотехніки	Архітектура комп'ютерів	Моделювання та оптимізація інформаційних систем		
	Махович	Основи інформаційної безпеки С #	Побудова систем інтернет речей	Хмарні технології	Сучасні інтернет технології	
	Миколайчук	Веб-дизайн та веб-програмування	Сучасні інформаційні системи та технології	Обробка даних в інформаційних технологіях	Інтелектуальний аналіз даних	Бази даних та інформаційні системи
	Дуднік	Операційні системи	Системи зв'язку з рухомими об'єктами	Системне програмування	Компю'терна графіка	Оптичні транспортні системи та мережі
	Нікіфоров	Веб-дизайн та веб-програмування				
	Ковальчин	Іноземна мова (2-й семестр)				
	Дорик	Українська та зарубіжна культура				
	Хомин	Науковий образ світу				
	Файчак	Менеджмент та маркетинг				
	Герасимчук	Філософія				
	Барна	Соціально-політичні студії				
	Удич	Вибрані розділи трудового права і основ підприємницької діяльності				
	Юзифович	Іноземна мова (7-й семестр)				

Рисунок 3.13 – Введення реальнихданих до таблицы БД “Викладачі”

Дії	Назвва групи	Курс	К-сть студентів	Зміна	К-сть груп
	МІТ - 1	1	25	1	2
	МІТ - 2	2	23	2	2
	МІТ - 3	3	27	2	2
	МІТ - 4	4	25	1	2

Рисунок 3.14 – Введення реальнихданих до таблицы БД “Групи студентів”

№	Дії		Назва предмету	семестр	черговість	лекція	практика
1			Вступ до університетських студій	1	2	22	
2			Вступ до фаху	1	4	16	18
3			Іноземна мова	1	5		102
4			Вища математика в інформаційних технологіях (1-й семестр)	1	5	18	52
5			Основи програмування	1	5	18	52
6			Іноземна мова	2	5		102
7			Вища математика в інформаційних технологіях	2	5	18	52
8			Електротехніка та електроніка	2	5	18	52
9			Теорія алгоритмів	2	5	18	52
10			Основи побудови інфокомунікаційних мереж	2	5	18	34
11			Комп'ютерна логіка та дискретна математика	3	3	16	34
12			Основи схемотехніки	3	3	16	18
13			Веб-дизайн та веб-програмування	3	5	18	52
14			Технології та протоколи мультисервісних мереж	3	5	18	52
15			Основи інформаційної безпеки С #	3	5	18	34
16			Компю'терна графіка	3	5	16	18
17			Українська та зарубіжна культура	3	5	16	
18			Обробка даних в інформаційних технологіях	4	5	18	52
19			Сучасні інформаційні системи та технології	4	5	18	52
20			Ймовірностні основи в інформаційних технололгіях	4	5	16	34
21			Оптичні транспортні системи та мережі	4	5	16	34
22			Архітектура комп'ютерів	4	5	16	18
23			Менеджмент та маркетинг	4	5	18	
24			Науковий образ світу	4	5	34	

Рисунок 3.15 – Введення реальнихданих до таблицы БД “Дисципліни”

Дії		Номер кабінету	Тип кабінету	Вмісткість кабінету
		212	Кабінет практики	17
		309	Кабінет практики	20
		204	Кабінет практики	20
		218	Лекційна зала	30
		318	Лекційна зала	30

Рисунок 3.16 – Введення реальнихданих до таблицы БД “Аудиторії”

Вивести всі дані OK

ID	Дата	День тижня	Пара	Група1	Викладач	Каб.	тип	Предмет	Предмет	тип	Каб.	Викладач	Група2	Викладач	Каб.	тип	Предмет	Предмет	тип	Каб.	Викладач
1	2021-09-01	Wed	1	МІТ - 1	ААААА	218	лекція	Вступ до університетських студій		лекція	218	ААААА	МІТ - 4	Лещенко	318	лекція	Моделювання та оптимізація інформаційних систем		лекція	318	Лещенко
2	2021-09-01	Wed	2	МІТ - 1	Кравченко	218	практика	Вступ до фаху	Вища математика в інформаційних технологіях (1-й семестр)	практика	212	Дахно									
4	2021-09-01	Wed	4	МІТ - 2	Дахно	218	лекція	Комп'ютерна логіка та дискретна математика		лекція	218	Дахно	МІТ - 3	Герасименко К.В	318	лекція	Технології безпровідних мереж		лекція	318	Герасименко К.В
5	2021-09-01	Wed	5	МІТ - 2	Дахно	218	практика	Комп'ютерна логіка та дискретна математика	Основи схемотехніки	практика	212	Лещенко									
7	2021-09-02	Thu	1	МІТ - 1	Кравченко	218	лекція	Вступ до фаху		лекція	218	Кравченко	МІТ - 4	Герасименко О.Ю	318	лекція	Організація баз даних		лекція	318	Герасименко О.Ю
8	2021-09-02	Thu	2	МІТ - 1	Дахно	218	практика	Вища математика в інформаційних технологіях (1-й семестр)	Вступ до фаху	практика	212	Кравченко									
10	2021-09-02	Thu	4	МІТ - 2	Лещенко	218	лекція	Основи схемотехніки		лекція	218	Лещенко	МІТ - 3	Миколайчук	318	лекція	Інтелектуальний аналіз даних		лекція	318	Миколайчук
11	2021-09-02	Thu	5	МІТ - 2	Лещенко	218	практика	Основи схемотехніки	Комп'ютерна логіка та дискретна математика	практика	212	Дахно									
13	2021-09-03	Fri	1	МІТ - 1	Дахно	218	лекція	Вища математика в інформаційних технологіях (1-й семестр)		лекція	218	Дахно	МІТ - 4	Герасименко К.В	318	лекція	Кібернетична безпека підприємства		лекція	318	Герасименко К.В
14	2021-09-03	Fri	2	МІТ - 1	Герасименко О.Ю	218	практика	Основи програмування	Вступ до фаху	практика	212	Кравченко									

Рисунок 3.17 – Виконання тестування ІС на реальних даних

Проаналізувавши наведений рисунок, можна побачити, що автоматичне формування розкладу занять відбувається за умови виконання всіх вимог та обмежень. Системою забезпечено рівномірний розподіл кількості годин, які повинні бути відпрацьовані згідно з науковим планом освітньої програми, кожній дисципліні присвоєно відповідного викладача, числовий показник місткості аудиторій задовольняє кількість студентів в групі. В процесі виконання алгоритму було визначено, що для відпрацювання занять, передбачених освітньою програмою, необхідне проведення двох пар на день. Забезпечено чергування лекційних та практичних занять. Наявність вікон у викладачі та студентів зведено до мінімуму.

Висновки до розділу ІІІ

Отже, в процесі виконання розділу відбулося створення таблиць в БД, необхідних для розробки автоматизованої системи генерації розкладу занять, а саме: “Timetable”, “Lessons”, “Teachers”, “Classrooms”, “Students”, “General”.

Розроблено модуль підтримки користувацького інтерфейсу, що забезпечує можливість взаємодії користувача з БД можливість введення, редагування та видалення даних, необхідних для побудови розкладу, а саме: перелік наявних аудиторії, перелік дисциплін, професорсько-викладацький склад, перелік груп студентів. Даний модуль також дозволяє здійснювати запуск визначеного алгоритму та відображення сформованих результатів.Стилізація веб-сторінки досягнуто за рахунок використання засобів CSS.

З метою перевірки функціонування програми в базу даних додатку було введено набір тестових даних. Даними заповнено таблиці “Timetable”, “Lessons”, “Teachers”, “Classrooms”, “Students”. Протестовано автоматичне створення розкладу занять для першого та другого семестру чотирьох груп факультету.

ВИСНОВОК

Отже, результатом виконання даної роботи є розв’язання задачі автоматичної генерації розкладу занять для вищого навчального закладу з урахуванням певних вимог та обмежень.

В процесі розв’язання задачі було здійснено аналіз інформаційних та математичних моделей, а також алгоритмів для розв’язання задачі автоматичної генерації розкладу занять. Проведено огляд поширених методів цілочислового програмування. На основі результату таких досліджень було запропоновано алгоритм, що базується на методі підстановки та розв’язує задачу автоматичної побудови розкладу. Використання даного методу дало можливість забезпечити виконання робочого навчального плану та графіку процесу навчання, забезпечити оптимальний режим роботи для викладачів та студентів, рівномірний розподіл навантаженості протягом тижня.

Проведено розробку структурної моделі інформаційної системи. Виконано проектування бази даних шляхом створення концептуальної, логічної та фізичної моделей даних.

Проведено аналіз переваг та недоліків найпопулярніших СУБД. Для створення бази даних та її подальшого адміністрування було використано СУБД MySQL. Створено базу даних разом з необхідними таблицями.

Проведено аналіз найбільш використовуваних серверних мов програмування. На основі результату аналізу було прийнято рішення використовувати мову програмування PHP.

Здійснено розробку веб-інтерфейсу для взаємодії користувача з інформаційною системою. Модуль інтерфейсу користувача забезпечує можливість введення, редагування та видалення даних, необхідних для побудови розкладу, а саме: перелік наявних аудиторії, перелік дисциплін, професорсько-викладацький склад, перелік груп студентів. Даний модуль повинен також дозволяє здійснювати запуск визначеного алгоритму та відображення сформованих результатів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Снитюк В.Є. Про особливості формування цільової функції та обмежень в задачі складання розкладу занять / Снитюк В.Є., Сіпко Є.Н. // Математичні машини і системи – 2014 - №3 – С. 67-76
2. Снитюк В.Є. Аспекти формування цільової функції в задачі складання розкладу занять у вищих навчальних закладах на основі суб'єктивних переваг / Снитюк В.Є., Сіпко Є.Н. // Автоматика. Автоматизація. Електротехнічні комплекси і системи - 2013 – №2 – С.98-104
3. Бевз С. В. Розробка автоматизованої системи формування розкладу магістратури / Бевз С. В., Войтко В. В., Бурбело С. М., Шоботенко А. М. // Інформаційні технології та комп'ютерна техніка – 2009 - №4 – С. 30-65
4. Бевз С. В. Розробка автоматизованої системи формування розкладу магістратури / Бевз С. В., Войтко В. В., Бурбело С. М., Шоботенко А. М. // Інформаційні технології та комп'ютерна техніка – 2009 - №4 – С. 30-65
5. Бевз С. В. Автоматизація процесу формування розкладу сесії. / Бевз С.В., Войтко В.В., Бурбело С.М., Куба Т.О., Сухоносов О.О.// Принципові концепції та структурування різних рівнів освіти з оптикоелектронних інформаційно-енергетичних технологій – 2009 - №4 – С. 25-36
6. Астахова І.Ф. Створення розкладу навчальних занять на основі генетичного алгоритму / Астахова І.Ф., Фірас А.М. // Вісник воронежского державного університету, серія: «Системний аналіз и інформаційні технології». – 2013. – № 2. – С 93-99.
7. Деканова М.В. Математична модель и алгоритм побудови розкладу навчальних занятий університету / Деканова М.В. // Вісник

Полоцкого державного університету. Серія С. – 2013. – №12. – С. 24-33.

8. Верьовкін В.И. Автоматизоване створення розкладів навчальних занять вишу с урахуванням складності дисциплін і втомленості 79 студентів / Верьовкін В.И., Ісмагілова О.М., Атавін Т.А. // Доповіді ТУСУР. – 2009. – №1 (19), частина 1. – С. 221-225.
9. Бабкіна Т.С. Задача складання розкладу: рішення на основі багатоагентного підходу / Бабкіна Т.С. // Бізнес-інформатика. – 2008. – №1. – С.23-28.
10. Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечеткие системы — М.: Горячая линия, 2006. 452 с
11. Панченко Т.В. Генетические алгоритмы: учебнометодическое пособие — 2007. — Режим доступа: http://window.edu.ru/catalog/pdf2txt/ 394/39394/17112?p_page=1
12. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности / Г. К.Вороновский и др. Х.: ОСНОВА, 1997. 112 с.
13. Каширина И.Л. Генетический алгоритм решения многокритериальной задачи о назначениях / И.Л. Каширина, Б.А. Семенов // Информационные технологии. – 2007. – № 5. – С. 62-68.
14. Введение в ГА и генетическое программирование. [Электронный ресурс]. – Режим доступа к источнику: http://algolist.manual.ru/ai/ga.
15. Курейчик В.М. Эволюционные вычисления: генетическое и эволюционное программирование / В.М. Курейчик, С.И. Родзин // Новости искусственного интеллекта. – 2003. – № 5. – С. 13-19.
16. Genetic Algorithms: Mathematics. [Электронный ресурс]. – Режим доступа: http://articles.mql4.com/134.

17. Ротштейн А.П., Интеллектуальные технологии идентификации: нечеткие множества, генетические алгоритмы, нейронные сети. – Винница: УНІВЕРСУМ–Вінниця, 1999. – 320 с
18. Database [Електронний ресурс] // Oracle FAQ. – 2008. – Режим доступу до ресурсу: http://www.orafaq.com/wiki/Database
19. Microsoft SQL Server [Електронний ресурс] // Wikipedia. – 2017. – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/Microsoft_SQL_Server.
20. MySQL [Електронний ресурс] // Wikipedia. – 2017. – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/MySQL.

ДОДАТОК А

Програмний код автоматизованої системи формування розкладу занять.

```
<?php
session_start();
require 'db.php';

// Перевіряємо чи зайшов адміністратор
   if ($_SESSION[user][access] != 1)   {
      header('Location:index.php');   exit;   // якщо не адмін - відправляємо на початок
   }
// КІНЕЦЬ => Перевіряємо чи зайшов адміністратор

   // Редагування викладача
   if (isset($_GET[edit_teach_id])){
      $_SESSION[teacher_plus] = true;
   }

   // Редагування групи студентів
   if (isset($_GET[edit_Group_id]))    {
      $_SESSION[group_stud_plus] = true;
   }
   // Редагування предмета
   if (isset($_GET[edit_less_id]))     {
      $_SESSION[lesson_plus] = true;
   }
   // Редагування кабінету
   if (isset($_GET[edit_cl_room_id]))  {
      $_SESSION[cabin_plus] = true;
   }

//***** Кнопки ВИКЛАДАЧ ***** //
   if ($_SESSION[teacher_plus] == true)    {
      $name_teach_plus = 'teacher_plus_NO';
      $class_butt_tp = 'butt_no';
      $text_butt_tp = 'ЗАКРИТИ';
   }   else    {
      $name_teach_plus = 'teacher_plus';
      $class_butt_tp = 'butt_ok';
      $text_butt_tp = 'ДОБАВИТИ';
   }

   if ($_SESSION[teacher_list] == true)    {
      $name_teach_list = 'teacher_list_NO';
      $class_butt_tl = 'butt_no';
      $text_butt_tl = 'ЗАКРИТИ';
   }   else    {
      $name_teach_list = 'teacher_list';
      $class_butt_tl = 'butt_ok';
      $text_butt_tl = 'Список';
```

```
    }
// КІНЕЦЬ => ***** Кнопки ВИКЛАДАЧ ***** //

// *****  К А Б І Н Е Т И  ***** //
    if ($_SESSION[cabin_plus] == true)    {
        $name_cab_plus = 'cabin_plus_NO';
        $class_butt_cab_plus = 'butt_no';
        $text_butt_cab_plus = 'ЗАКРИТИ';
    }   else    {
        $name_cab_plus = 'cabin_plus';
        $class_butt_cab_plus = 'butt_ok';
        $text_butt_cab_plus = 'ДОБАВИТИ';
    }
    // Список
    if ($_SESSION[cabinet_list] == true)    {
        $name_cab_list = 'cabinet_list_NO';
        $class_butt_cab_list = 'butt_no';
        $text_butt_cab_list = 'ЗАКРИТИ';
    }   else    {
        $name_cab_list = 'cabinet_list';
        $class_butt_cab_list = 'butt_ok';
        $text_butt_cab_list = 'Список';
    }
// КІНЕЦЬ => *****  К А Б І Н Е Т И  ***** //

// ***** кнопки ПРЕДМЕТИ ***** //
if ($_SESSION[lesson_plus] == true)    {
    $name_less_plus = 'lesson_plus_NO';
    $class_butt_lep = 'butt_no';
    $text_butt_lep = 'ЗАКРИТИ';
}   else    {
    $name_less_plus = 'lesson_plus';
    $class_butt_lep = 'butt_ok';
    $text_butt_lep = 'ДОБАВИТИ';
}
// Список
if ($_SESSION[lesson_list] == true)    {
    $name_less_list = 'lesson_list_NO';
    $class_butt_lel = 'butt_no';
    $text_butt_lel = 'ЗАКРИТИ';
}   else    {
    $name_less_list = 'lesson_list';
    $class_butt_lel = 'butt_ok';
    $text_butt_lel = 'Список';
}
// КІНЕЦЬ => ***** кнопки ПРЕДМЕТИ ***** //

// ***** ГРУПИ  СТУДЕНТІВ ***** //
    if ($_SESSION[group_stud_plus] == true)    {
        $name_group_stud_plus = 'group_stud_plus_NO';
        $class_butt_gr_stud = 'butt_no';
        $text_butt_gr_stud = 'ЗАКРИТИ';
```

```
    }   else    {
        $name_group_stud_plus = 'group_stud_plus';
        $class_butt_gr_stud = 'butt_ok';
        $text_butt_gr_stud = 'ДОБАВИТИ';
    }
    // Список
    if ($_SESSION[group_stud_list] == true)    {
        $name_gr_stud_list = 'group_stud_list_NO';
        $class_butt_gr_stud_list = 'butt_no';
        $text_butt_gr_stud_list = 'ЗАКРИТИ';
    }   else    {
        $name_gr_stud_list = 'group_stud_list';
        $class_butt_gr_stud_list = 'butt_ok';
        $text_butt_gr_stud_list = 'Список';
    }
// КІНЕЦЬ => ***** ГРУПИ  СТУДЕНТІВ ***** //

// ***** Р І З Н Е ***** //
    if ($_SESSION[different_ok] == true)    {
//        $name_diff = 'group_stud_list_NO';
        $class_butt_diff = 'butt_no';
//        $text_butt_gr_stud_list = 'ЗАКРИТИ';
    }   else    {
        $class_butt_diff = 'butt_ok';
    }
// КІНЕЦЬ => ***** Р І З Н Е ***** //
?>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/html">
<head>

    <meta charset="UTF-8">
    <title>Кабінет адміна</title>
    <link rel="stylesheet" href="styles.css">

</head>
<body>

<!--// Лінки - кнопки - меню-->
    <div class="section_left">
        <a href="exit.php" class="link_button">ВИЙТИ</a>
        <a href="timetable.php" class="link_button">Розклад</a>
        <br>
    </div>
<!--// Лінки - кнопки - меню-->
<!--кнопка "ДАЛІ >>"-->
<div class="section_right">
    <form action="control.php">
        <button class="button" style="vertical-align:middle"><span>Далі </span></button>
    </form>
</div>
<!--кнопка "ДАЛІ >>"-->
```

```
   <div class="section_center">
      <form action="work_insert.php" method="POST" id="center-form">
         <table>
   <!--          Викладач-->
            <tr>
               <td width="150" align="center">
                  ВИКЛАДАЧ
               </td>
               <td>
                  <button type="submit" name="<?php   echo $name_teach_plus;  ?>"
                        class="<?php   echo $class_butt_tp;  ?>"> <?php   echo $text_butt_tp;  ?>
</button>
               </td>
               <td>
                  <button type="submit" name="<?php   echo $name_teach_list;  ?>"
                        class="<?php   echo $class_butt_tl;  ?>"> <?php   echo $text_butt_tl;  ?>
</button>
               </td>
            </tr>
   <!--          Викладач-->
   <!--          Кабінети-->
            <tr>
               <td width="150" align="center">
                  КАБІНЕТИ
               </td>
               <td>
                  <button type="submit" name="<?php   echo $name_cab_plus;  ?>"
                        class="<?php   echo $class_butt_cab_plus;  ?>"> <?php   echo
$text_butt_cab_plus;  ?> </button>
               </td>
               <td>
                  <button type="submit" name="<?php   echo $name_cab_list;  ?>"
                        class="<?php   echo $class_butt_cab_list;  ?>"> <?php   echo
$text_butt_cab_list;  ?> </button>
               </td>
            </tr>
   <!--          Кабінети-->
   <!--          Предмет-->
            <tr>

               <td width="150" align="center">
                  ПРЕДМЕТ
               </td>
               <td>
                  <button type="submit" name="<?php   echo $name_less_plus;  ?>"
                        class="<?php   echo $class_butt_lep;  ?>"> <?php   echo $text_butt_lep;
?></button>
               </td>
               <td>
                  <button type="submit" name="<?php   echo $name_less_list;  ?>"
```

```
                class="<?php   echo $class_butt_lel;  ?>"> <?php   echo $text_butt_lel;
?></button>
            </td>
        </tr>
    <!--        Предмет-->
    <!--        Група студентів-->
        <tr>

            <td width="150" align="center">
                ГРУПА
            </td>
            <td>
                <button type="submit" name="<?php   echo $name_group_stud_plus;  ?>"
                        class="<?php   echo $class_butt_gr_stud;  ?>"> <?php   echo
$text_butt_gr_stud;  ?></button>
            </td>
            <td>
                <button type="submit" name="<?php   echo $name_gr_stud_list;  ?>"
                        class="<?php   echo $class_butt_gr_stud_list;  ?>"> <?php   echo
$text_butt_gr_stud_list;  ?></button>
            </td>
        </tr>
    <!--        Група студентів-->
    <!--        Різне-->
        <tr>

            <td width="150" align="center">
                РІЗНЕ
            </td>
            <td>
                <button type="submit" name="different_ok"
class="butt_ok">ВИВЕСТИ</button>
            </td>
            <td>
                <button type="submit" name="different_no" class="<?php echo $class_butt_diff
?>">Сховати</button>
            </td>
        </tr>
    <!--        Різне-->
      </table>
    </form>
  </div>
<?php
// Викладачі : додати / вивести список
if ($_SESSION[teacher_plus] == true) {
//    echo '<section class = "section_center">';
   include 'f_insert/plus_teacher.php';     // виводимо таблицю додавання викладдачів
//    echo '</section>';
}
if ($_SESSION[teacher_list] == true) {
   include 'f_insert/list_teachers.php';    //  виводимо список внесених викладачів
}
```

```
// КІНЕЦЬ -> Викладачі : додати / вивести список
// Кабінети: додати / вивести список
    if ($_SESSION[cabin_plus] == true)      {
        include 'f_insert/classroom_plus.php';
    }
    if ($_SESSION[cabinet_list] == true)   {
        include 'f_insert/classroom_list.php';
    }
// КІНЕЦЬ -> Кабінети: додати / вивести список
// Предмети : додати / вивести список
if ($_SESSION[lesson_plus] == true)    {
    include 'f_insert/lesson_plus.php';
}
if ($_SESSION[lesson_list] == true)    {
    include 'f_insert/lesson_list.php';
}
// Групи студентів : додати / вивести список
if ($_SESSION[group_stud_plus] == true)    {
    include 'f_insert/group_stud_plus.php';
}
if ($_SESSION[group_stud_list] == true)    {
    include 'f_insert/group_stud_list.php';
}
// РІЗНЕ
if ($_SESSION[different_ok] == true)    {
    include 'f_insert/different.php';
}
// КІНЕЦЬ -> РІЗНЕ
?>

</body>
</html>
```