ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МІСЬКОГО ГОСПОДАРСТВА ІМЕНІ О. М. БЕКЕТОВА
Навчально-науковий інститут енергетичної, інформаційної
та транспортної інфраструктури

Кафедра комп'ютерних наук та інформаційних технологій

Пояснювальна записка
до кваліфікаційної роботи бакалавра

на тему: «Розробка системи автоматичного розкладу занять для університету»

Виконала: студентка 4 курсу, групи КН 2021-1
спеціальності
122 Комп'ютерні науки
(шифр і назва спеціальності)


Вікторія МИШКОВА
(ім'я та прізвище)


Керівник: Борис БОЧАРОВ
(ім'я та прізвище)


Рецензент Юрій ПАХОМОВ
(ім'я та прізвище)

м. Харків – 2025 рік

Харківський національний університет міського господарства імені О. М. Бекетова
(повне найменування закладу вищої освіти)
Навчально-науковий Інститут енергетичної, інформаційної та транспортної інфраструктури
Кафедра комп'ютерних наук та інформаційних технологій
Рівень вищої освіти перший (бакалаврський)
Спеціальність 122 «Комп'ютерні науки»
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри КНтаІТ


Марина НОВОЖИЛОВА
«__26__» ___06_____2025 Року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТЦІ

Мишковій Вікторії Євгеніївні
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка системи автоматичного розкладу занять для університету

керівник роботи к.т.н., доц. Бочаров Б.П.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від «09» травня 2025 р. № 341-03

2. Термін подання студентом роботи 23.06.2025р.

3. Вихідні дані до роботи Розробка системи автоматичного розкладу занять для університету

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
Дослідження предметного середовища та огляд аналогів для автоматизованого розкладу; створення програмного завдання; вибір інструментів розробки програми; розробка і тестування системи автоматичного розкладу занять за допомогою бібліотеки від Google OR-Tools.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Презентація – 11 аркушів

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ I	Борис БОЧАРОВ	15.05.2025	16.05.2025
Розділ II	Борис БОЧАРОВ	24.05.2025	29.05.2025
Розділ III	Борис БОЧАРОВ	03.06.2025	05.06.2025
Розділ IV	Вікторія МАЛИШЕВА	07.06.2025	08.06.2025

7. Дата видачі завдання 15.05.2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Вибір теми дипломної роботи	15.05.2025	Виконано
2	Затвердження тем, наукових керівників, завдань та календарного плану підготовки дипломної роботи	16.05.2025	Виконано
3	Написання I розділу	20.05.2025	Виконано
4	Написання II розділу	25.05.2025	Виконано
5	Написання III розділу	06.06.2025	Виконано
6	Написання IV розділу	15.06.2025	Виконано
7	Подання дипломної роботи керівнику	20.06.2025	Виконано
8	Робота по усуненню зауважень керівника, уточнення і доповнення практичного матеріалу, оформлення додатків до роботи	22.06.2025	Виконано
9	Подання доопрацьованого варіанту роботи керівнику	15.06.2025	Виконано
10	Захист матеріалів дипломної роботи на засіданні кафедри	18.06.2025	Виконано
11	Офіційний захист матеріалів дипломної роботи на засіданні Державної екзаменаційної комісії	27.06.2025	Виконано

Студент _____ (підпис) Мишкова. В.Є. (прізвище та ініціали)

Керівник роботи _____ (підпис) Бочаров Б.П. (прізвище та ініціали)

АНОТАЦІЯ

Структура та обсяг роботи. Пояснювальна записка кваліфікаційної роботи бакалавра студентки групи КН 2021-1 спеціальності Комп'ютерні науки Мишкової Вікторії Євгеніївни за темою «Система автоматичного розкладу занять за допомогою бібліотеки від Google OR-Tools» складається з 4 розділів, містить 14 рисунків, 8 таблиць та 20 джерел інформації.

Основною метою кваліфікаційної роботи є розробка програмного забезпечення та побудова ефективної моделі для автоматизованого розподілу занять з урахуванням різноманітних обмежень: обмеженого ресурсу аудиторій, зайнятості викладачів, типів занять та максимального навантаження на студентські групи.

У розділі «Загальні положення» розглянуто предметне середовище тв алгоритми створення навчального плану занять, а також порівняння аналогів для цієї системи. Також відбулася постановка задачі.

Розділ «Інформаційне та математичне забезпечення» має побудову математичної моделі, бази даних, а також проєктування системи автоматичного створення розкладу.

У розділі «Програмне та технічне забезпечення» описано програмне та технічне забезпечення проєкту, також є пояснення програмного коду реалізації системи з детальним описом. Окрім цього, наведено керівництво користувача.

У розділі охорони праці визначені вимоги до організації робочого місця із урахуванням шкідливих та небезпечних виробничих факторів.

Ключові слова: ЗАДАЧА ОПТИМІЗАЦІЇ, АВТОМАТИЗАЦІЯ, АВТОМАТИЗОВАНИЙ РОЗКЛАД ЗАНЯТЬ, РЕАЛІЗАЦІЯ, ПРОГРАМУВАННЯ, БАЗА ДАНИХ, ПРОЄКТУВАННЯ СИСТЕМИ, ОБМЕЖЕННЯ, МОДЕЛЬ, БІБЛІОТЕКА.

ANNOTATION

Structure and scope of work. Explanatory note of the bachelor's qualification work of the student of the group KN 2021-1, speciality Computer Science, Myshkova Victoria Yevheniivna on the topic ‘System of automatic class scheduling using the library from Google OR-Tools’ consists of 4 sections, contains 14 figures, 8 tables and 20 sources of information.

The main purpose of the qualification work is to develop software and build an effective model for the automated distribution of classes, taking into account various constraints: limited classroom resources, teacher employment, types of classes and maximum load on student groups.

The section ‘General Provisions’ describes the subject environment and algorithms for creating a lesson plan, as well as a comparison of analogues for this system. The task was also formulated.

The section ‘Information and Mathematical Support’ includes the construction of a mathematical model, a database, and the design of an automatic timetable creation system.

The section ‘Software and hardware’ describes the software and hardware of the project, as well as an explanation of the program code for implementing the system with a detailed description. In addition, a user manual is provided.

The labour protection section defines the requirements for the organisation of the workplace, taking into account harmful and dangerous production factors.

Keywords: OPTIMISATION TASK, AUTOMATION, AUTOMATED CLASS SCHEDULE, IMPLEMENTATION, PROGRAMMING, DATABASE, SYSTEM DESIGN, CONSTRAINTS, MODEL, LIBRARY.

ЗМІСТ

ВСТУП 8
РОЗДІЛ 1 ЗАГАЛЬНІ ПОЛОЖЕННЯ 10
1.1 Опис предметного середовища 10
1.2 Огляд наявних аналогів 15
1.3 Постановка задачі 19
Висновки до розділу 21
РОЗДІЛ 2 ІНФОРМАЦІЙНЕ ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ 23
2.1 Аналіз предметної області 23
2.1.1 Вхідні данні 25
2.1.2 Вихідні дані 27
2.2 Проєктування системи 28
2.2.1 Проєктування бази даних 28
2.3 Математичне та алгоритмічне забезпечення 31
Висновки до розділу 35
РОЗДІЛ 3 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ 37
3.1 Засоби розробки 37
3.2 Вимоги до технічного та програмного забезпечення 38
3.3 Опис програмної реалізації 39
3.4 Керівництво користувача 47
Висновки до розділу 51
РОЗДІЛ 4 ОХОРОНА ПРАЦІ 52
4.1 Регулювання питань охорони праці на законодавчому рівні 52
4.2 Виявлення потенційних небезпек стосовно об’єкту проектування 55

4.3 Дослідження ризику реалізації потенційних небезпек на об’єкті проектування та розробка заходів щодо їх попередження 56

4.4 Висновки по розділу 58

ВИСНОВКИ 60

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 62

ВСТУП

В освітніх закладах складання розкладу є одним з найважливіших завдань в процесі створення кожного плану навчального року. Грамотно сформований план занять є ключовим елементом для реалізації науково-педагогічного потенціалу навчальних дисциплін і сприяє кращому засвоєнню знань студентами. Це доручення, зазвичай, виконує адміністратор навчального закладу вручну або за допомогою малоефективних засобів для створення плану занять. Задачі розподілу навчальної роботи між співробітниками кафедри, розробки навчального розкладу в значній мірі визначають ефективність організації освітнього процесу. Весь процес є складним з точки зору логістики, адже передбачає враховувати велику кількість обмежень та вимог: наявність аудиторії, доступність викладачів, навантаження студентів, специфіку навчальних дисциплін, тривалість занять, побажання викладачів або студентів, а також перерви, тощо.

Адміністративний персонал витрачає завжди багато часу на формування розкладу, використовуючи традиційний підхід, а ще й узгодження усіх параметрів вручну. Це також не тільки є трудомістким, а й ще може призвести до помилок у плануванні і неефективного використання ресурсів (людських та матеріально-технічних). До того ж, ручний спосіб створення розкладу не допоможе швидко адаптуватися до змін, які часом виникають в процесі навчального року (це може бути зміна складу груп, викладачів, тимчасова недоступність приміщень і тд.).

Таким чином, актуальність цієї проблеми зростає та потребує використання новітніх технологій, таких як штучний інтелект, математичне моделювання або операційне дослідження, яке допомагає вирішувати задачі створення автоматизації розкладу занять. Один з таких сучасних інструментів є бібліотека Google OR-Tools. Вона є потужним та сучасним інструментом з відкритим кодом, що розроблений компанією Google для розв’язання задач

комбінаторної оптимізації, включаючи не тільки планування, а й маршрутизацію та розміщення.

Мета даної дипломної роботи полягає в розробці автоматичного складання розкладу занять з використанням бібліотеки Google OR-Tools. У такий спосіб, ця система зможе ефективно вирішувати задачу з планування навчального процесу, враховуючи певну кількість обмежень та критерій. Робота передбачає собою дослідження теоретичної частини основи задачі, аналіз та опис предметного середовища, огляд можливих агалогів, їх преваги та недоліки, також проєктування і реалізація програмного рішення, яке автоматизує цей процес й зробити певні висновки.

Успішна реалізація цієї системи дозволить значно зменшити часові витрати та підвищити якість розподілу занять у навчальному процесі. Результати цієї роботи можуть бути використані у різних навчальних закладах (університетах, коледжах, школах тощо), як частина системи управління навчальним процесом.

РОЗДІЛ 1
ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Опис предметного середовища

Кожен рік відбувається планування освітнього процесу в навчальних закладах. Завдання створити розклад для студентів – це неабияка важлива умова успішної діяльності навчальної установи. Формування розкладу занять має стратегічно визначати основні шляхи розвитку навчального закладу, орієнтувати педагогів на реалізацію ціннісних пріоритетів особистості, задоволення освітніх потреб учнів, створення освітнього середовища, в якому реалізовувалася б сучасна модель випускника - особистості, готової до життя із самореалізацією сформованих компетентностей. [1]

Освітня програма, річний план роботи, календарне та поурочне планування освітньої діяльності є чинниками, що визначають якість освітньої діяльності та є джерелом інформації під час інституційного аудиту та самооцінювання закладу, а також дозволяють директору ефективно організувати освітній процес та діяльність закладу в цілому. Планування роботи інституції забезпечує реалізацію стратегії розвитку закладу, а також створює умови для організованої взаємодії учасників освітнього процесу, раціонального використання часу та інших можливостей. [1]

Планування поділяється на такі види:

- Перспективне
- Річне
- Поточне

У кожному випадку формування потрібно приймати до відома інформацію з зовнішнього та внутрішнього походження. [1] Зовнішні дані охоплюють нормативно-правові акти, зокрема документи, які надсилає Міністерство освіти України, засновника закладу освіти та органів місцевого

самоврядування у сфері освіти. До внутрішньої інформації належать дані, що генеруються всередині навчального закладу, зокрема результати аналізу освітнього процесу, кадрового забезпечення, матеріально-технічної бази, потреб учасників освітнього процесу та інших ресурсів, які впливають на прийняття управлінських рішень.

Розглянемо, чим саме являє собою перспективне планування [1]:

- Аналіз результатів діяльності установи за попередній період;
- Формулювання ключових завдань на майбутній період;
- Організаційні труднощі забезпечення розвитку закладу;
- Основні напрями покращення освітньої діяльності;
- Господарську діяльність;
- Підвищення рівня матеріально-технічної бази закладу

Річне планування забезпечує поєднання стратегічних планів з поточними завданнями на семестри та/або місяці навчального процесу.

Поточне планування є щоденними завданнями, які будуть виконуватися весь навчальний рік [1], а саме:

- Розклад предметів та гуртків;
- Календарні та поурочні плани;
- Створення плану виховної роботи класних керівників, вихователів груп подовженого дня;
- Розклад роботи шкільної, університетської бібліотеки та батьківського комітету;
- Запис учнів за алфавітом у спеціальну книгу, журнал;
- Журнали для класу, факультативних занять, обліку роботи гуртків, моніторингу пропусків та зміщення уроків вчителями та/або викладачами;
- Книга обліку трудових книжок працівників закладу.

У закладах загальної середньої освіти розклад уроків є одним з основних організаційних документів. Саме він встановлює діяльність учнів та

педагогічних працівників. Планування заняттів у школі зазвичай створює заступник директора. Це не просте завдання, адже потрібно враховувати не тільки коли і який саме урок буде відбуватися, а ще й вікові особливості учнів і матеріально-технічне забезпечення школи, і побажання колег. Також важливо включати працездатність учнів, тому що від цього залежить продуктивність дітей та пріоритетність заняття. Нижче наведено таблицю 1.1, де можна краще ознайомитися з ефективністю учнів працювати та сприймати краще інформацію. [2]

Таблиця 1.1 – Продуктивність учнів

Класи	Продуктивні уроки	Непродуктивні уроки
1-2	1, 2, 4	3
3-5	1, 2, 3	4
6-7	1, 2, 3, 5	4, 6
8	1, 2, 3, 5	4, 6
9	1, 2, 3, 5	4, 6
10	1, 2, 3, 5, 6	4
11	1, 2, 3, 4, 5, 6	

Також потрібно враховувати сприятливість днів тижня для навчання. Нижче можна побачити таблицю 1.2, яка показує чітку інформацію про учнів. [2]

Таблиця 1.2 – Сприятливі дні тижня

Класи	Сприятливі дні	Несприятливі дні
1-4	Вівторок, середа, четвер	Понеділок, п’ятниця, субота
5	Понеділок, вівторок	Понеділок, п’ятниця, субота

6	Вівторок, четвер, п'ятниця, субота	Понеділок, середа
7-8	Понеділок, середа, четвер, субота	Понеділок, вівторок
9-10	Середа, четвер, п’ятниця	Субота

Враховуючи денний та тижневий цикли зміни працездатності учнів розклад має бути спрямований на збереження високого рівня працездатності учнів під час проходження всього навчального процесу. Крім того, загальновідомо, що працездатність учнів у другій половині дня хоча й піддається аналогічним коливанням, проте взагалі нижча, ніж у першій [2]. Тому адміністрація освітнього закладу повинна вжити всіх можливих заходів для раціонального розподілу класів за змінами з врахуванням вікових особливостей за умови навчання в першу зміну максимально можливої кількості учнів.

У вищих навчальних закладах пари планують на семестровій основі відповідно до робочих навчальних планів, освітніх програм та графіків начального процесу. До основних завдань цього процесу належать забезпечення повного виконання навчального етапу; оптимізація режиму навчання; створення сприятливих умов праці для студентів та викладачів; рівномірний розподіл навчального навантаження; ефективне використання аудиторного фонду [3].

Навчальний рік, семестр, спеціальність, курс, група, тиждень (парний/непарний), назва дисциплін, форма проведення занять (лекції, лабораторні, практичні, семінарські заняття, консультації) – все це входить до відомостей розкладу занять [3]. Він організований у дві зміни:

- Перша зміна: початок о 08:30
- Друга зміна – о 12:35 або 14:20

Одна академічна година становить, як правило, 45 хвилин. За допомогою Комплексної інформаційної системи «Автоматизована система

управління навчальним процесом», формується план пар для студентів. Приблизно за 2 місяці до початку семестру завідувачі кафедр пропонують основні ідеї для первинного наповнення даних, які потім подаються до навчально-методичного відділу, диспетчер формує за місяць до початку семестру готовий розклад навчальних занять [3]. При складанні плану занять враховуються наступні аспекти:

- тривалість аудиторних занять для студентів не повинна перевищувати 8 годин на день, включаючи факультативні дисципліни;
- при складанні розкладу занять слід рівномірно розподіляти кількість занять за днями тижня, бажано уникати наявності «вікон»;
- лекції необхідно включати в розклад на початку навчального дня але не більше шести академічних годин поспіль;
- практичні, лабораторні, семінарські заняття доцільно ставити після теоретичних занять;
- не рекомендується проводити в день більше трьох практичних або/та лабораторних занять;
- проведення занять з дисциплін, які вивчаються факультативно, планується на першу або останню пару [3].

Як і в ЗЗОС при складанні розкладу враховують теж побажання окремих викладачів, пов'язані з їх участю в науковій, навчально-методичний та виховній роботі, а також з сімейними обставинами за поданням завідувача відповідної кафедри та погодженням з деканом факультету (директором інституту/центру).

У розкладі не можна виключати той факт, що можуть відбуватися заміни та/або перенесення у, звичайно, виняткових випадках, наприклад, зміни у навчальному навантаженні, відрядження, хвороба викладачів тощо. Зміни у плані подає завідувач кафедри при погодженні з деканом факультету

(директором інституту/центру), начальником навчально-методичного відділу та з дозволу першого проректора [3].

1.2 Огляд наявних аналогів

Існує вже не так багато додатків та веб-сторінок, які допомагають створити розклад для навчальних закладів. Більшість призначені допомагати з планерами та здійснювати саме завдання, які потрібно виконати впродовж дня, наприклад, сходити в магазин, піти до лікаря, відвідати музей тощо. Як приклад аналогів, розглянемо програми «aSc Timetables», «UniTime» та ручне складання рішень.

«aSc Timetables» - це програмне рішення, призначене для автоматизації створення шкільних розкладів, що допомагає викладачам та адміністраторам ефективно керувати розкладом та мінімізувати конфлікти [4]. Програма пропонує такі функції, як автоматичне створення розкладу, ручне коригування, імпорт даних та мобільний доступ, що робить її комплексним інструментом для складання шкільного розкладу.

Спочатку я пропоную розглянути плюси додатку «aSc Timetables». Цей аналог може оцінити до 5 000 000 варіантів для створення розкладу, щоб створити хороший план для студентів [4]. Розробники цього додатку переконують, що створення розкладу займає 5 хвилин та внесення корегування до нього. Замінами теж можна керувати онлайн, бо програма не тільки може швидко додати заміну в розкладі, та ще й порадить вчителя-замінника або вчитель може обрати його сам. Також є функція «Журнал класу», яка відображає зміни, внесені до розкладу, заміни та події [4]. Просто можна вибрати тему уроку зі свого навчального плану та перевіряти відвідуваність учнів. Таким чином, нудне адміністрування перетворюється на корисну функцію. Батьки краще проінформовані про те, що вивчили їхні діти. Вчителі або адміністрація можуть додавати шкільні події, такі як: обговорення, екскурсії чи шкільні канікули, до загального шкільного

календаря. Він пов'язаний з журналом класу, тож забронювати кімнату або нараду вчителів легко спонтанно, за допомогою додатку. Система поєднує актуальну інформацію з розкладів, інформації про відсутніх вчителів та подій [4]. Вона знаходить найбільш підходящу кімнату для відведеної зустрічі у зручний для всіх учасників час. Бронювання одразу відображається в розкладі та в журналі відвідувачів. Усі учасники отримують миттєве сповіщення. Кімнату також можна забронювати безпосередньо в журналі відвідувачів. Таким чином жодні два класи ніколи не будуть зустрічатися в одному класі.

З вище розглянутих переваг аналогу, бачимо, що він має велику популярність та хорошу позицію на маркеті. Хоча додаток досить корисний, але він не має українську мову інтерфейсу [4]. Це може бути складно для тих, хто не розуміє англійську чи іншої європейської мови [4]. Також це може призвести до помилок у складанні розкладу та програма не зможе виконати план занять, так як не розрахована для державної мови. Другий пункт – зависока ціна цього додатку. Вартість «aSc Timetables» залежить від обраної версії. Існують різні варіанти з різними цінами та функціями [4]. Наприклад, версія «Початкові школи» (для початкових класів) коштує дешевше, тоді як «PRO» коштує дорожче для більш вимогливих потреб з індивідуальними розкладами (табл. 1.3) [4].

Таблиця 1.3 - Ціна кожної версії додатку

Вид версії	Опис	Ціна
«Початкові школи»	Спеціальна ціна для початкових класів	Одноразова оплата 399$
«Стандарт»	Необмежена кількість варіантів розкладу для шкіл	Одноразовий платіж 499$
«Преміум»	Допомога у складанні розкладу від команди aSc	Одноразовий платіж 995$

«PRO»	Створення індивідуальних розкладів для учнів	Одноразовий платіж 3995$

Другий аналог «UniTime» - це комплексна система планування навчального процесу, яка підтримує розробку розкладів курсів та іспитів, управління змінами в цих розкладах, спільне використання аудиторій з іншими подіями, а також запис студентів на окремі заняття [5]. Це розподілена система, яка дозволяє багатьом університетським та факультетським менеджерам координувати зусилля для створення та модифікації розкладу, який відповідає їхнім різноманітним організаційним потребам, дозволяючи мінімізувати конфлікти між студентськими курсами. Вона може використовуватися окремо для створення та підтримки розкладу занять та/або іспитів, або бути інтегрованою з існуючою студентською інформаційною системою.

Система була розроблена спільними зусиллями викладачів, студентів та співробітників університетів Північної Америки та Європи [5]. Програмне забезпечення розповсюджується безкоштовно за ліцензією з відкритим вихідним кодом в надії, що інші коледжі та університети зможуть принести користь своїм студентам за рахунок кращого планування розкладу або захочуть зробити свій внесок у дослідження в цій галузі, що тривають [5]. Проєкт «UniTime» став спонсорським проектом Фонду Apereo у березні 2015 року [5].

Хоча цей аналог і безкоштовний для використання, та він також не є ідеальним вибором для українських навчальних закладів, особливо якщо врахувати специфічні адміністративні, технічні та мовні реалії української системи освіти. Складність процесу налаштування та обслуговування UniTime являється однією з проблем, що передбачає конфігурацію серверів додатків на основі мови програмуванн Java, систем баз даних, таких як MySQL або PostgreSQL, а також детальне розуміння як технічної інфраструктури, які вона підтримує. Якщо не вистачає спеціального IT-персоналу з відповідним

досвідом, то це буде важко реалізувати. Крім того, «UniTime» пропонує обмежену мову інтерфейсу, хоча й технічно можливо його перекласти, українська мова не повністю буде підтримуватися [5]. Документація та системні повідомлення залишаться англійською мовою, що є бар’єром для впровадження в установах, де адміністративному персоналу, викладачам та студентам зручніше працювати українською мовою.

Розклад уроків це один з основних документів, які створюються в останні дні літніх канікул і за ним школа буде жити весь рік. Тому вимоги до розкладу уроків стоять високі. Ручне складання розкладу уроків потребує тримати в пам'яті багато параметрів. На початку навчального року багато сторонніх питань – зосередитись можна тільки у вечері чи на вихідні, часті термінові зміни розкладу на початку (вересень) вимагають швидкої переробки розкладу [6]. Як допомогу в створенні розкладу, використовують працівники навчальних закладів такі програми як «Excel» [6]. Вона допомагає легко та структуровано працювати та заповнювати таблиці. Але цей метод найважчий серед двох вище згаданих аналогів.

Розглянемо основні етапи ручного складання:

1. Збір інформації (дані про клас, кількість дітей, уроків, побажання вчителів та вікові особливості учнів тощо).
2. Розподіл предметів (чергування предметів, їх порядок та складність).
3. Формування розкладу (створення плану уроків та забезпечення рівномірного навантаження).
4. Коригування розкладу (аналіз на можливі помилки в розкладі).
5. Публікація розкладу (оприлюднення навчального плану в закладі освіти)

Цей метод може займати багато часу та зусиль для великих навчальних закладів, такі як: гімназії, коледжі або університети тощо. Не можна виключати, що при такому способі буде чимало помилок у розкладі, які призводять до конфліктів у часі, неправильно обрані аудиторії та навантаження як на учнів, так і на викладачів. Щоб ввести якісь зміни у

розклад, потрібно все переробляти заново, що займає знову багато часу та роботи. Тому сучасні навчальні заклади все більше обирають автоматизовані системи, що покращують роботу, спрощують складання розкладу, зменшують кількість помилок та оптимізують використання ресурсів.

1.3 Постановка задачі

В освітньому процесі складання розкладу це найважливіша та найскладніших організаційних задач. Необхідно враховувати велику кількість змінних: кількість студентів та груп, доступність викладачів, аудиторії, розклад занять, а також врахування та обмеження всіх учасників процесу [6]. За допомогою ручного складання, буде важко це все врахувати та створити розклад без помилок, крім цього ця робота займатиме багато часу. Як наслідок, це призведе до конфліктів в розкладі, перенавантаження викладачів та студентів.

Мета дипломної роботи – створення автоматизованого плану занять, що допоможе швидко та ефективно розподіляти заняття, беручи до уваги обмеження та пріоритетів. Це рішення повинно дати високу якість будування розкладу, можливість користуватися цим не тільки у вищих закладах, а й у освітніх установах різного рівня.

Для вирішення цієї проблеми є хорошим вибором використання бібліотеки Google OR-Tools [7]. Вона являється це програмним пакетом з відкритим вихідним кодом для оптимізації, налаштований на вирішення найскладніших світових проблем в області маршрутизації транспортних засобів, потоків, цілочисельного та лінійного програмування, а також програмування з обмеженнями. Як бачимо, ця бібліотека підтримує різноманітні алгоритми, тож вона добре вирішує складні комбінації задач, що виникають при складанні розкладів.

Через зазначені вище плюси бібліотеки Google OR-Tools, вона обрана для реалізації системи автоматизованого плану заняттів [7]. Вона добре

підтримується Google, а також є гнучка й потужна в роботі з задачами, які враховують чисельні обмеження і залежності. Бібліотека має ефективні оптимізаційні алгоритми для знаходження рішень для описання обмеження в природній формі. OR-Tools має ідеальний модуль Constraint Solver, який може створити рішення для розкладів, де потрібно враховувати численні вимоги й одночасно досягти оптимального або наближеного рішення [7].

З ціллю автоматизації цієї організаційної задачі виникає потреба у створенні програмної системи, що підбирає час, місця та послідовність заняття, відповідно до навчального плану; не забуває про змогу викладачів викладати предмет, доступність аудиторії; розподіляє рівномірно заняття та оптимізує навантаження викладачів протягом тижня; мінімізує кількість «вікон» у розкладі; також надає можливість швидкого перегляду та модифікації сформованого розкладу.

По перше, для досягнення мети, яка поставлена для вирішення такої задачі потрібно проаналізувати предметну область. Іншими словами важливо визначити структуру даних, учасників процесу та обмеження при складанні розкладу. По друге, необхідно вивчити та переглянути наявні підходи та аналоги, а саме: аналіз сучасних програмних продуктів для автоматизації розкладу та розглянути переваги і недоліки. Крім цього, потрібно вивчити та обрати інструменти розробки бібліотеки Google OR-Tools, основні її функції та методи для основного способу моделювання та оптимізації [7].

Розробка програмного забезпечення є найважливішим етапом цієї дипломної роботи. Потрібно реалізувати модель та дозволити взаємодію з користувачем і дозволити редагувати розклад. До цього ж створення інтуїтивно зрозумілого інтерфейсу користувача, щоб працювати з системою та вносити дані: група, викладач, аудиторія, час, дата тощо. Також веб-сайт повинен виводити результат, щоб провести перевірку та редагування відповіді.

Автоматизація створення розкладу є вагомим практичним значенням. Насамперед, це значно скорочує час та ресурси, які витрачаються на

формування плану занять вручну. Наступним чином, щоб уникнути помилок, які людина може зробити при цьому процесі, автоматизація є хорошим рішенням цієї проблеми. Крім того ж, ця ідея підвищить ефективність освітнього процесу та оптимальність використання ресурсів начального заклад

Висновки до розділу

Як показує опис розробки розкладу занять, можна сказати, що це може займати багато часу та зусиль. Тому було розглянуто, як можна зробити це легше та процес був не таким довготривалим. Формування розкладу, як це розглянуто на аналізі предметного середовища, потребує урахування великої кількості обмежень, таких як навантаження студентів та викладачів, доступні аудиторії, тільки одна група може мати практичне заняття у одного вчителя, особливості навчального плану, побажання персоналу тощо [1].

Проведено аналіз поточного стану справ у цій справі, а саме розглянуто традиційний підхід складання розкладу вручну, а також автоматизовані рішення, що частково застосовуються в деяких навчальних закладах. Серед яких виявлено недоліки існуючих систем, таких як немає українського інтерфейсу, відсутність логіці в побудові.

У світлі цих викликів українські заклади можуть виявити, що більш практичним та ефективним є використання простіших, локалізованих рішень - чи то спеціально розроблених платформ для управління школами, що відповідають українським нормативним вимогам та навчальним програмам, чи то більш легких інструментів планування, які легко інтегруються з існуючими цифровими робочими процесами - замість того, щоб намагатися впровадити систему на кшталт «UniTime» чи «aSc Timetables», яка, хоча і є надійною у власному контексті, в кінцевому підсумку може виявитися невідповідною адміністративній культурі, технічній спроможності та освітній структурі в Україні [5, 6].

У результаті обґрунтовано вибір розробки нової інформаційної системи з використанням бібліотеки Google OR-Tools для вирішення задач комбінованої оптимізації та пошуку в умовах обмежень [7]. Такий підхід дозволить краще моделювати правила та обмеження, які зустрічаються при складанні розкладу у навчальних закладах та автоматично знаходити рішення.

РОЗДІЛ 2
ІНФОРМАЦІЙНЕ ТА МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Аналіз предметної області

Загалом, розклад занять – це нормативний документ, який встановлює робочий ритм і впливає на ефективність творчої діяльності педагогічних працівників та зайнятість студентів [3]. Його можна визначити як один із факторів удосконалення навчального процесу. Щоб побудувати хорошу та чітку систему автоматичного розкладу, потрібно враховувати детальний опис вхідних даних і багато факторів, таких як: корпус освітньої установи, аудиторія, час, викладачі, тощо. Це все може впливати на зміни в подальшому плануванні занять і тому це потрібно включати в рішення до постановки задачі. Ігнорування цих даних може призвести до помилок, як наприклад, дві різні групи мають одну й ту ж аудиторію одночасно, або ж учитель не може бути вдвох місцях за один раз.

Для цього потрібно використовувати оптимізацію обмежень, або іншими словами програмування за обмеженнями (ОП) [13]. Воно визначає можливі рішення з дуже великого набору кандидатів, коли проблема може бути змодельована в термінах великих обмежень. Проблеми ОП виникають у багатьох наукових та інженерних дисциплінах. (Слово «програмування» є дещо неправильно назвою, подібно до того, як «комп’ютер» колись означало «людина, яка обчислює». Тут же слово «програмування» відноситься більше до організації плану, а не до програмування комп’ютерною мовою) [13].

ОП базується на здійсненності (знаходження здійсненного рішення), а не на оптимізації (знаходженні оптимального рішення), і фокусується на обмеженнях і змінних, а не на цільовій функції [13]. Насправді, задача програмування за обмеженнями може навіть не мати цільової функції – метою

може бути звуження дуже великої множини можливих рішень до більш керованої підмножини шляхом додавання обмежень до задачі.

Розглянемо більш складніший приклад складання графіків не для навчального закладу, а для роботи співробітників на одній фірмі. Проблема виникає, коли компаніям, які працюють безперервно, наприклад, заводам, потрібно створити тижневі графіки для своїх працівників [13]. Ось дуже спрощений приклад: компанія працює в три 8-годинні зміни на день і призначає трьох з чотирьох своїх працівників на різні зміни кожного дня, а четвертому дає вихідний. Навіть у такому невеликому випадку, кількість можливих розкладів величезна, тобто кожного дня існує 24 можливих призначення працівників, отже, кількість можливих тижневих розкладів становить 247, тобто понад 4,5 мільярда [13]. Зазвичай існують інші обмеження, які зменшують кількість можливих рішень – наприклад, кожен працівник повинен працювати принаймні мінімальну кількість днів на тиждень. Метод програмування за обмеженнями відстежує, які рішення залишаються можливими, коли додаються нові обмеження, що робить його потужним інструментом для вирішення великих, реальних проблем планування [13].

Оптимізація обмежень має широку і дуже активну спільноту в світі з спеціалізованими науковими журналами, конференціями та арсеналом різних методів розв’язування. ОП успішно застосовується в плануванні, складанні розкладів та багатьох інших областях з різнорідними обмеженнями [13].

Після детального прикладу ОП, можна перейти до аналізу ключових сутностей предметної області, які впливатимуть в цьому випадку на рішення поставленої мети. Найголовніші сутності, які мають значення для розкладу та кількості можливих рішень цієї задачі:

- Група – є невеликою сукупністю студентів, приблизно з 20-25 людей, які відвідують певний набір занять.
- Дисципліна – предмет, який викладається у певному обсязі, з зазначеною кількістю годин на семестр.

- Вид предмету – лекція чи практичне заняття, де під час лекції може бути одночасно 2-4 групи, а під час практичної пари всього лиш одна група.
- Викладач – має один чи більше предметів, яка може вест їх у різних групах.
- Аудиторія – місце, де проводиться пара з тієї чи іншої дисципліни.
- Час – кількість фіксованих слотів (день/пара), коли проводяться заняття.

Кожна сутність має певний зв’язок між собою, які потім утворюють обмеження для завдання.

2.1.1 Вхідні данні

Як вже відомо, складання розкладу – це складна задача що включає в себе безліч сутностей, правил, обмеження та пріоритети [1]. У системі автоматичного розкладу занять важливу роль відіграє формалізація вхідних даних. Саме для конкретної роботи планування, за допомогою Googlr OR-Tools, слід брати до уваги такі основні групи інформації: аудиторії, викладачі, предмети та навчальні групи, а також часові слоти [7].

Всі ці дані будуть передаватися у одну функцію, яка генеруватиме підходящий план для студентів та викладачів, що задовольняє всі обмеженя, які були задані раніше. Формалізація вхідних даних має велике значення у системі автоматичного розкладу занять. Це інформація, яка визначає контекст навчального процесу: академічні групи, предмети, викладачі, аудиторії, час, пара та тип заняття (лекція або практичне заняття). Вся ця інформація передається у функцію «generate_schedule_ortools», що пропонує можливий розклад, беручи за основу вхідні дані, й задовольняє задані обмеження. Як приклад, було розглянуто основні типи даних, які реалізовано за допомогою Python-коду [8].

Групи студентів подаються у вигляді масиву, наприклад, «groups = [’КН-22’, ’УПКН-22’]». Кожна група відвідує пари у різні часові слоти та мають свій набір дисциплін. Кожна група є унікальним ідентифікатором у цьому випадку.

Кожна пара має свій тип занять, які в коді виглядають так: «art_of_subject = {’Лк’, ’Пз’}». За допомогою цієї множини можна відокремити формат занять, це потрібно для проведення лекцій відразу для кількох груп, а практичні заняття можуть бути тільки в однієї.

Дисципліни представлені як кладений словник (dict of dicts), де для кожна група має пари, які потрібно відвідувати, та обов’язково тип предмету, що допоможе врахувати той факт – пара викладається у формі лекції та/або практичного заняття [8]. Частина коду виглядає наступним чином:

subjects = { 'УПКН-21': {'Бази Даних': ['Лк', 'Пз'], 'Математика': ['Лк', 'Пз'], 'Програмування': ['Лк', 'Пз']},

'КН-22': {'Бази Даних': ['Лк', 'Пз'], 'Програмування': ['Лк', 'Пз'], 'Математика': ['Лк', 'Пз']} }

Другий словник відображає викладачів та предмети, які вони проводять «teachers = {'Математика': 'Петренко І.К.', 'Програмування': 'Іванов П.В.', 'Бази Даних': 'Сидоренко Ф.А.',}». У складніших випадках структуру можна покращити до списку викладачів або окремого об’єкта з графіком доступності.

Кожна аудиторія внесена також в один масив «rooms = [’101’, ’102’, ’103’]», він слугує як ідентифікатор класу, що доступні для проведення занять [8]. Тут є обмеження, що практичне заняття тільки однієї групи може проводитися у відповідний час. У базовій реалізації кімнати розглядаються як взаємозамінні, але у розширеній можна врахувати і тип аудиторій (наприклад, лабораторна або лекційна).

Також створено список навчальних днів з понеділка по п’ятницю «days = ['Пн', 'Вт', 'Ср', 'Чт', 'Пт']», саме тоді можливе проведення занять [8]. Створено список часових слотів, які відповідають за час проведення пар [8]. Кожен час відповідає одній парі «hours = ['08:30', '10:15', '12:35', '14:20', '16:05', '17:45']».

Для всіх цих вхідних даних було обрано легку та зручну для обробки структуру: списки (list), множини (set) та словники (dict) [8]. Вони просто інтегруються з бібліотекою Google OR-Tools, що допомагає перетворити їх у змінні задачі задоволення обмежень [7]. Таким чином, вхідні дані є базою для

побудови обмежень у моделі, щоб сформувати задачу таким чином, щоб алгоритми Google OR-Tools ефективно її вирішили [7].

2.1.2 Вихідні дані

Результатом вихідних даних в автоматичному формуванні розкладу є запропонований результат самого плану занять, тобто розв'язок задачі з обмеженнями, яку вирішує CP-SAT Solver з бібліотеки Google OR-Tools [7]. Коли система знаходить розклад, який задовольняє, то вона генерує структурований тижневий розклад для кожної академічної групи, дотримуючись усіх визначених обмежень.

Як тільки виконується функція «generate_schedule_ortools», формується розклад для кожної групи. Сама ж відповідь виглядає у відформатованому переліку занять з вказанням: дня тижня, час початку заняття, назва предмету, тип заняття (лекція або практика), викладач та аудиторія. Вони генеруються шляхом зчитування значень із відповідних змінних, зо створюються у формуванні математичної моделі часових слотів «slot_vars» та аудиторій «room_vars». Метод solver.Value(…) зчитує ці дані [7].

У цьому розкладі кожен запис має формат словника зі структурою: { "day_idx": day_idx, "hour_idx": hour_idx, "day": day, "hour": hour, "subject": subj, "type": typ, "teacher": teacher, "room": room}. Ця структура легко адаптує вивід до будь-якого способу представлення, а саме: друк у консоль, експорт у PDF, веб-інтерфейс або інтеграцію з електронним журналом.

Наприклад, так може виглядати вивід розкладу для кожної групи:

Розклад для групи УПКН-21:

Пн 14:20 - Програмування (Лк) (Іванов) в ауд. 103

Пн 16:05 - Математика (Пз) (Петренко) в ауд. 102

Пн 17:45 - Математика (Лк) (Петренко) в ауд. 101

Розклад для групи КН-22:

Пн 10:15 - Програмування (Пз) (Іванов) в ауд. 101

Пн 12:35 - Математика (Пз) (Петренко) в ауд. 101

Пн 14:20 - Програмування (Лк) (Іванов) в ауд. 101

За допомогою сортування функції group_schedule.sort(…) можемо бачити упорядкований розклад по днях і годинах, який показує логічну послідовність плану впродовж тижня. Якщо ж рішення було не знайдено, то програма виводить повідомлення про це «Рішення не знайдено.». Користувач може виявити потім проблему вхідних даних або завелику кількість обмежень, що не дозволили знайти правильне рішення цієї задачі.

Таким чином, усі ці отримані дані всієї системи допоможуть адміністрації правильно спланувати робочий процес, викладачам не отримати надмірного навантаження, а студентам доступ до інформації про навчальний розклад. Окрім того, структура вихідної інформації зберігає результати у базу даних та передає їх для візуалізації.

2.2 Проєктування системи

Модульний підхід є основою системи автоматизованого розкладу, що може забезпечити гнучкість, масштабованість та розділення логіки, тобто введення інформації, обробка та планування, формування вихідного розкладу.

Система використовує локальну структуру даних (словники Python), а також може працювати через підключення до Системи керування базами даних (СКБД, наприклад, SQLite або PostgreSQL), якщо потрібно зберегти дані на певний проміжок часу або функціонувати через веб-інтерфейс [7, 9].

2.2.1 Проєктування бази даних

Ефективними способом опрацьовувати великий обсяг даних можна лише за умови їх зберігання у структурованому вигляді та мати наявність добре налагоджений доступ до інформації. Для зберігання, накопичення, опрацювання та швидкого пошуку інформації існують електронні бази даних — файли (сукупності файлів) спеціального формату, які містять структуровані дані.

Система керування базами даних (СКБД) — це система, заснована на програмних та технічних засобах, яка забезпечує визначення, створення, маніпулювання, контроль, керування та використання баз даних [9]. Застосунки для роботи з базою даних можуть бути частиною СКБД або автономними. СКБД дозволяють ефективно працювати з базами даних, обсяг яких робить неможливим їх ручне опрацювання. Через тісний зв'язок баз даних з СКБД під терміном «база даних» інколи необґрунтовано та неточно мають на увазі систему керування базами даних. [9] Але варто розрізняти базу даних — сховище даних, та СКБД — засоби для роботи з базою даних. СКБД з інформаційної системи може бути видалена, але база даних продовжить існувати. І навпаки: СКБД може функціонувати без жодної бази даних. [9, 10]

Загалом, переносити базу даних з однієї СКБД до іншої неможливо без деяких ускладнень. Системи керування базами даних класифікують переважно за їх типом моделі організації даних. Найпоширеніші СКБД використовують реляційну модель, що передбачає представлення даних у вигляді структурованих таблиць [10]. Безпосередня взаємодія кінцевого користувача (та прикладних програм) з базою даних, звичайно, неможлива. Доступ до даних можуть мати шляхом формування спеціалізованих запитів, що передаються до системи керування базами даних. СКБД обробляє ці запити та повертає відповіді, вона є єдиним компонентом, що взаємодіє з базою даних. Як правило, база даних має такі об’єкти, як схеми, таблиці, подання (віртуальні таблиці), збережені процедури та інші елементи структури. [10]

Сучасні СКБД мають різні функції, які можна поділити на такі групи [10]:

- Оголошення даних – створення, заміна та видалення, які описують організацію даних.
- Модифікація даних – додавання даних, їх редагування та видалення.
- Отримання даних – надання даних за запитом застосунку у формі, яка дозволяє їх безпосереднє використання. Дані можуть

надаватись або у формі, в якій вони зберігаються у базі даних, або в іншій формі (наприклад, через поєднання різних даних).

- Адміністрування даних – реєстрування та відслідковування дій користувачів, дотримання безпеки роботи з даними, забезпечення надійності та цілісності даних, моніторинг продуктивності, резервне копіювання та відновлення даних тощо.

У рамках реалізації системи обрано структуру вкладених словників, які імітують сутності бази даних. Основними сутностями є група, дисципліна, тип заняття, викладач, аудиторія та тимчасовий слот.

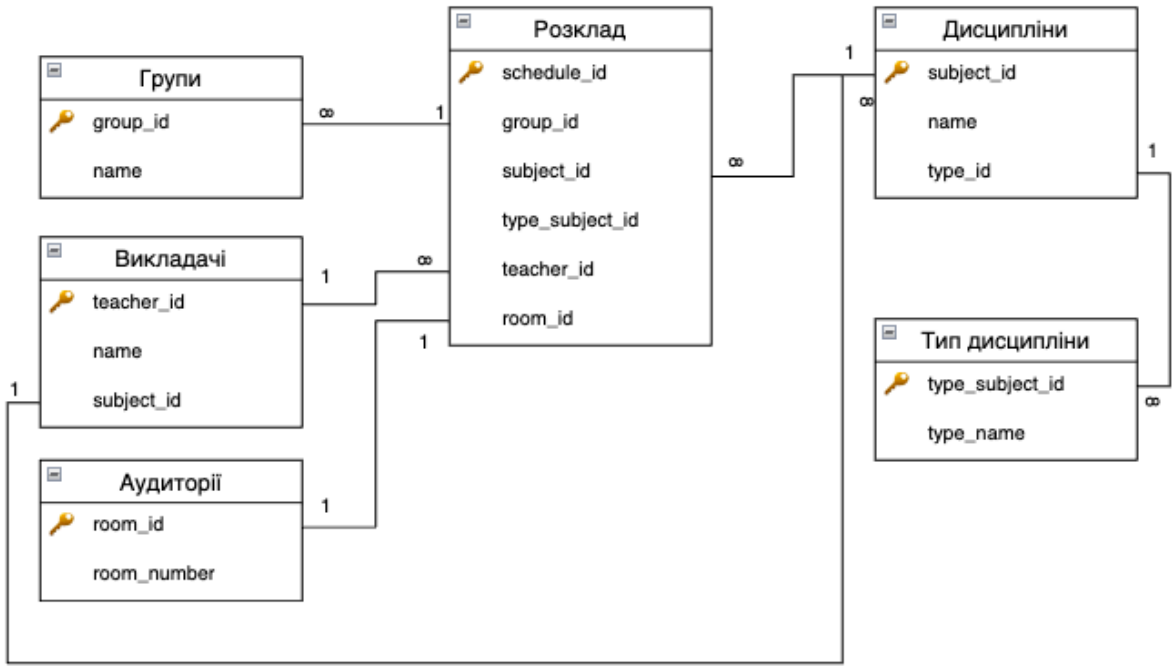


Рисунок 2.1 – Основні сутності бази даних

Система автоматичного розкладу складається з трьох логічних блоків (рис. 2.3)

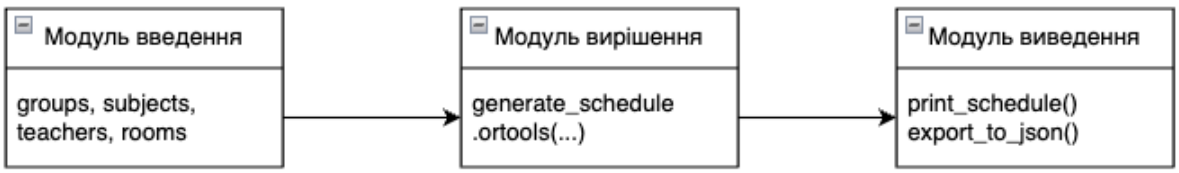


Рисунок 2.2 – UML діаграма

Модуль введення даних створює початкові дані для планування. Він зчитує та ініціалізує групи, предмети, викладачів, типів занять та аудиторії, а також перевіряє на коректність наповнення, зв’язки та унікальність.

Модуль вирішення (планування) будує розклад на основі вхідних даних та обмежень. Тут використовуються бібліотека Google OR-Tools з функцією CP-SAT Solver та обмеження, де є максимум 4 пари на день, відсутність накладок по викладачах, аудиторіях і групах [7]. Як результат, програма повертає розклад у вигляді словника, який є оптимальним та допустимим для робітників навчального закладу.

Модуль виведення результату відображає та експортує сформований розклад. Він може виводити, як текстовий вивід у консоль, так і вивід у JSON (формат, можливий для подальшої візуалізації), а також експорт у веб-інтерфейсі (при розширені).

2.3 Математичне та алгоритмічне забезпечення

Система автоматизованого розкладу – це типовий приклад комбінованої задачі з достатньою кількістю обмежень. Щоб вирішити цю задачу, потрібно використовувати методи математичного програмування, точніше модель задачі з задоволення обмежень (CSP, Constraint Satisfaction Problem). Вона реалізована за допомогою Google OR-Tools для оптимізаційних задач [7].

Методи математичного програмування базуються на розв'язанні задачі комп'ютерного молекулярного дизайну як оптимізаційної задачі, де цільова функція визначається в термінах критеріїв ефективності, а цільові властивості, які повинні бути задоволені, вводяться в якості обмежень [11]. В основному розглядаються два підходи до розв'язання (на основі декомпозиції та прямого розв'язання). При підході на основі декомпозиції основна проблема декомпозується на набір підпроблем, які вирішуються відповідно до ієрархічного порядку. При цьому до останньої підпроблеми визначаються

тільки рішення, що задовольняють підмножину обмежень. З кожним рішенням підзадачі простір рішень зменшується.

Математичне програмування складається з трьох етапів [11]:

- Створення математичної моделі – це переклад реальних проблем математично, визначаючи питання, які будуть поставлені (змінні рішення), обмеження та цілі, яких потрібно досягти.
- Розробка алгоритмів – вирішують ці математичні моделі програмування.
- Запуск алгоритмів – етап, коли потрібно прогнати свою модель через вирішувач, щоб знайти відповідь на проблему (тобто відповіді на запитання, виходячи з унікальних цілей і обмежень).

Математичне програмування - це підхід до вирішення проблем, який використовує математичні моделі та алгоритми для оптимізації процесів прийняття рішень [11]. З іншого боку, комп'ютерне програмування - це написання коду для створення програмного забезпечення або систем, які можуть виконувати комп'ютери [12]. Хоча вони обидва включають слово «програмування», вони мають різні фокуси та цілі.

Задача про задоволення обмежень - це математична задача, в якій рішення повинно відповідати ряду обмежень. У ній метою є присвоєння значень змінним таким чином, щоб задовольнити всі обмеження. [13] Багато програм штучного інтелекту використовують задачу з задоволення обмежень для вирішення проблем прийняття рішень, які передбачають управління або організацію ресурсів відповідно до суворих інструкцій. До поширених застосувань цих задач відносяться [13]:

- Планування розкладу, яке призначає ресурси, такі як працівники або обладнання, дотримуючись обмежень за часом і доступністю.
- Планування – організація завдань з конкретними дедлайнами або послідовністю.
- Розподіл ресурсів, який призначений для ефективного розподілу ресурсів без надмірного використання.

Типи проблем, пов'язаних із задоволенням обмежень можна класифікувати на різні типи залежно від їхніх обмежень та характеристик проблеми:

Бінарні задачі з обмеженнями це ті задачі, у яких кожне обмеження стосується лише двох змінних. Як і в задачі планування, обмеження може вказувати на те, що завдання А має бути завершене раніше завдання Б. [13]

Небінарні задачі з обмеженнями мають обмеження, які включають більше двох змінних. Наприклад, у задачі розсаджування в класі обмеження може вказувати, що троє людей не можуть сидіти поруч один з одним. [13]

Жорсткі та м'які обмеження – жорсткі обмеження повинні бути суворо виконані, тоді як м'які обмеження можуть бути порушені, але за певну ціну. Це часто використовується в реальних додатках, де не всі обмеження є однаково важливими. [13]

Всього задача з обмеженнями має три ключові компоненти, такі як змінні, домени та обмеження [13]. Змінні – це те, для чого нам потрібно знайти значення. Кожна змінна представляє щось, що потрібно вирішити. Наприклад, у головоломці Судоку кожна порожня клітинка - це змінна, яка потребує числа. Змінні можуть бути різних типів, наприклад, так/ні (булеві), цілі числа (цілі) або категорії, такі як кольори або імена.

Домени змінних – набір можливих значень, які може мати змінна. Домен показує нам, які значення ми можемо вибрати для кожної змінної. У Судоку доменом для кожної клітинки є числа від 1 до 9, тому що кожна клітинка повинна містити одне з цих чисел. Деякі області є маленькими і обмеженими, в той час як інші можуть бути дуже великими або навіть нескінченними. [13]

Обмеження – правила, які обмежують те, як змінним можна присвоювати значення. Обмеження визначають, які комбінації значень дозволені. Вони поділяються на три типи: унарні (обмеження застосовуються до однієї змінної), бінарні (обмеження стосуються двох змінних) та обмеження вищого порядку (включають три або більше змінних). [13]

Для програмування з обмеженнями існує багато розв'язувачів з відкритим вихідним кодом, але вони зазвичай не масштабуються так добре, як MIP-розв'язувачі, і гірше оптимізують цільові функції. У той час як MIP-розв'язувачі (mixed-integer programming) часто здатні оптимізувати задачі з сотнями тисяч змінних і обмежень, класичні CP-розв'язувачі часто борються із задачами з більш ніж кількома тисячами змінних і обмежень. Однак відносно новий CP-SAT з набору OR-Tools від Google подолав багато з цих недоліків і є життєздатною альтернативою MIP-розв'язувачам (mixed-integer programming), будучи конкурентоспроможним для багатьох задач, а іноді навіть кращим за них. [7, 13]

Для системи автоматизованого розкладу занять формальна модель буде виглядати так:

$G = \{g_1, g_2 \dots, g_n\}$ – множина груп

$S = \{s_1, s_2 \dots, s_m\}$ – множина навчальних дисциплін

$T = \{t_1, t_2 \dots, t_k\}$ – множна викладачів

$R = \{r_1, r_2 \dots, r_p\}$ – множина аудиторій

$D = \{Пн, Вт, Ср, Чт, Пт\}$ – дні тижня

$H = \{08{:}30, 10{:}15 \dots\}$ – години

Кожна лекція або практичне заняття для групи g_1 по предмету s_1 типу $typ \in \{Лк, Пз\}$ описується двома змінними: слот часу (TimeSlot) – комбінація та години; аудиторія (room) – індекс доступної аудиторії.

У Google OR-Tools ці змінні створюються за допомогою [7]: «slot_vars[(group, subj, typ)] = model.NewIntVar(0, num_time_slots - 1, f"slot_{var_name}")
room_vars[(group, subj, typ)] = model.NewIntVar(0, len(rooms) - 1, f"room_{var_name}")».

У таблиці 2.1 нижче показано логічні обмеження, які повинна дотримуватися система.

Таблиця 2.1 – Логічні обмеження

№	Тип обмеження	Формулювання
1	Часовий конфлікт групи	Група не може мати два заняття в один і той самий час.
2	Часовий конфлікт викладача	Один викладач не може викладати одночасно у двох групах, якщо це не лекція.
3	Обмеження на кількість занять вдень	Кожна група не може мати більше 4 занять в день.
4	Конфлікти аудиторій	Неможливо призначити дві пари в одну й ту саму аудиторію одночасно.
5	Спільна лекція для кількох груп	Лекція повинна відбуватись одночасно в одній аудиторії для всіх груп.

Google OR-Tools використовує CP-SAT Solver, що є сучасним та потужним алгоритмом для вирішення задач з обмеженнями [7]. Ця функція ефективно виключає заборонені варіанти на основі обмежень, замість того, щоб перебирати всі можливі комбінації. Рішення представляються у вигляді логічних змінних і логічних формул.

Висновки до розділу

У цьому розділі проаналізовано предметну область системи автоматизованого розкладу занять. Окрім цього, визначено сутності та їхній взаємозв’язок, а також ключові змінні та обмеження, які мають великий вплив на побудову розкладу.

Також було виявлено, що вхідні дані для цієї задачі охоплюють п’ять таких компонентів, як: групи, предмети, викладачі, аудиторії та часові слоти. Уся ця інформація може зберігатися в базі даних або файлах, наприклад, у форматі JSON/CSV, які легко інтегруються у програмну систему. Коректне моделювання вхідних даних забезпечує точну постановку задачі з задоволеними обмеженнями. Саме ці сутності формують базу, на якій будується об’єктна модель та математичне забезпечення системи.

Проєктування системи розподіляється на відповідальності між модулями введення, обробки та виводу даних. Завдяки використанню

структурованих словників, система легко дозволяє маштабуватися та розширбватися. Таким чином, архітектура система може забезпечити гнучкість при інтеграції з іншими інформаційними системами або перетворенні її у повноцінний веб-застосунок.

Також була створена математична модель для системи планування розкладу, що допомогло чітко сформулювати змінні, домени та обмеження. За допомогою інструментів Google OR-Tools і CP-SAT Solver вдалося реалізувати ефективну масштабовану систему створення навчального розкладу, що може працювати з десятками груп, викладачів та предметів [7]. Таким чином, це забезпечує наочність та практичну придатність для впровадження в навчальні заклади.

РОЗДІЛ 3
ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Засоби розробки

Щоб створити систему автоматичного розкладу, потрібні сучасні засоби програмування, а також обчислювальне середовище. Вони зможуть забезпечити гнучкість і продуктивність.

До технічних засобів розробки входять комп’ютер, який має модель MacBook Pro 13" (2022), процесор Apple M2, оперативну пам’ять 8 Гб та загальну пам’ять 256 Гб, а також операційну систему macOS Ventura. Цей ноутбук допомагає добре обчислювати потужність для розробки, тестувати та виконувати задачі оптимізації.

До програмних засобів мова програмування Python 3.13.2, фреймворк Django 5.2.3, бібліотека оптимізації Google OR-Tools (CP-SAT Solver), середовище розробки Visual Studio Code [7, 8, 15]. Вибір Python був зумовлений тим, що це інтерпретована, об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою [8]. Високорівневі вбудовані структури даних у поєднанні з динамічною типізацією та динамічним зв'язуванням роблять її дуже привабливою для швидкої розробки додатків, а також для використання в якості мови сценаріїв або мови-клею для з'єднання існуючих компонентів між собою. Окрім цього, одна з причин, обрати цю мову програмування, зумовлена тим, що вона підтримується фреймворком Django. Django - це високорівневий веб-фреймворк Python, який заохочує швидку розробку та чистий, прагматичний дизайн [8, 15]. Він бере на себе більшу частину клопоту з веб-розробкою, тому можна зосередитися на написанні додатку.

Середовище розробки Visual Studio Code має простий та інтуїтивний дизайн, що допомагає краще читати та писати код програми. Він значно спрощує розробку, налагодження, пошук посилок і запуск коду.

3.2 Вимоги до технічного та програмного забезпечення

Щоб розробити та тестувати систему автоматизованого розкладу, користувачу слід мати комп’ютер з мінімальними характеристиками, які наведені нижче в таблиці 3.1.

Таблиця 3.1 - Технічні вимоги для технічого забезпечення

Компоненти	Мінімальні характеристики
Процесор	2-ядерний, 2.0 ГГц або вище
Оперативна пам’ять	8 Гб і вище
Жорсткий диск	500 Мб вільного місця
Оперативна система	Windows 10+, macOS 11+, Linux
Підключення	Інтернет

Щоб створити функціональну та робочу програму для системи, потрібно також брати до уваги програмні вимоги для проєкту. Саме вони впливають на розробку та покращення продуктивності. Нижче наведено таблицю 3.2 з програмними характеристики системи.

Таблиця 3.2 - Технічні вимоги для програмного забезпечення

Компоненти	Характеристики
Python	Версія 3.10 або вище
Django	Версія 4.0 та вище
Google OR-Tools	Версія 9.0 та вище
PostgreSQL	Для зберігання даних
Visual Studio Code	Версія 1.75 та вище

3.3 Опис програмної реалізації

Основою системи автоматизованого розкладу занять є комплекс сучасних технологій, що ефективно поєднують в собі моделювання задачі, зручний та інтуїтивно простий інтерфейс для користувача, а також надійну роботу з даними. Ґрунт системи – логіка створення плану занять, що реалізована на мові програмування Python з використанням бібліотеки Google OR-Tools [7]. Вона забезпечує програму потужним інструментом для формулювання задачі з обмеженнями, а також допомагає розв’язувати їх, беручи до уваги багато сутностей: наявність аудиторій, викладачів, промідків часу та вимоги до кількості занять у день.

У програмі Visual Studio Code було здійснено розробку програми. Visual Studio Code дозволяє зручно редагувати код, інтегрувати системи контролю версій, таким чином, було завантажено актуальну версію фреймворку Django та Python [8, 15]. Загалом, проєкт, складається з модульної структури, де є окремо виділені частини, які відповідають за введення даних, за алгоритмічну обробку та планування, і відображення результатів для користувачів на веб-сторінці.

Часто програмісти використовують Python через підвищену продуктивність, яку він забезпечує. Налагоджувати програми на Python легко: помилка або неправильний ввід ніколи не призведе до помилки сегментації. [8] Натомість, коли інтерпретатор виявляє помилку, він згенерує виключення. Якщо програма не перехоплює виняток, інтерпретатор виводить трасування стеку. Налагоджувач на рівні коду дозволяє перевіряти локальні та глобальні змінні, обчислювати довільні вирази, встановлювати точки зупинки, переглядати код по одному рядку за раз тощо. Налагоджувач написаний самою мовою Python, що свідчить про інтроспективну силу Python [8]. З іншого боку, часто найшвидший спосіб налагодити програму - це додати кілька операторів виводу у вихідний код: швидкий цикл редагування-тестування-налагодження робить цей простий підхід дуже ефективним.

Django будо використано, щоб працювати з даними, а також створити веб-сторінку для виведення навчального розкладу [15]. Фреймворк слугує, як серверна частина, що зберігає інформацію про групи студентів, предмети, викладачів, аудиторії, а також розклад у базі даних. За допомогою Django створено динамічну веб-сторінку, яка інтуїтивно зрозуміла користувачеві та легка для використання. [15]. Через них юзер може додавати або редагувати вхідні параметри, запускати процес генерації плану занять і, звісно, побачити запропонований розклад у зручному форматі. Object-Relational Mapping – це функція, яка допомагає працювати з базою даних через об’єкти Python, при цьому уникати необхідність написання складних запитів SQL [8].

На ринку існує декілька веб-фреймворків. Django написаний на мові Python і є одним з багатьох веб-фреймворків на Python [8, 15]. Однак, розробники часто віддають перевагу Django перед іншими веб-фреймворками з наступних причин. Фреймворк Django добре організований і простий у встановленні та вивченні, тому можна розпочати роботу за лічені години. Розробники Django створили фреймворк для швидкої реалізації будь-якої веб-архітектури в коді. Він підтримує швидку розробку та чистий, прагматичний дизайн. Це дозволяє написати код всього за кілька рядків, тому що Django надає готову структуру для декількох поширених завдань веб-розробки, таких як [15]:

- Аутентифікація користувачів,
- Адміністрування контенту,
- Карти сайту.

Будь-який веб-додаток складається з двох частин: серверного та клієнтського коду. Клієнт або відвідувач веб-сайту має браузер. Коли він набирає URL-адресу в браузері, він надсилає запит на веб-сервер, на якому запущено веб-додаток. Сервер обробляє запит за допомогою бази даних і надсилає інформацію назад клієнту у відповідь. Клієнтський код відображає інформацію відвідувачу у вигляді веб-сторінки.

Створення автоматизованого розкладу закладена у модулі, що є основною логікою цієї програми. Вона отримує вхідні дані у вигляді списків і словників: груп студентів, типів занять (наприклад, лекція чи практичне заняття), дисциплін, викладачів, аудиторій та часових слотів. Ці дані передаються до Google OR-Tools, де створюються змінні, що відповідають за розподіл пар по часових інтервалах і аудиторіях [16]. Система накладає низку жорстких обмежень – наприклад, викладач не може вести дві пари одночасно, а також група не може бути зайнята більше ніж одним заняттям в той самий час, або група не може мати більше чотирьох занять на день. Крім того, реалізовано спільні лекції для кількох груп, які повинні проводитися одночасно у одній аудиторії.

Логіка побудови програми для автоматизованого розкладу базується на моделі задоволення обмежень, яку вирішує CP-SAT Solver – сучасний та ефективний розв’язувач, вбудований в бібліотеку Google OR-Tools [16].

Система отримує вхідні дані у вигляді (рис. 3.1):

- Переліку академічних груп,
- Доступних днів та часових слотів,
- Навчальних дисциплін та їх типів (лекціх та практичні заняття),
- Викладачів, які асоційовані із предметами,
- Списку доступних аудиторій.

Завдання системи – сформувати допустимий розклад, який задовольняє всі задані обмеження.

```python
# ⚠️ Дані можна взяти з БД або json
GROUPS = ['УПКН-21', 'КН-22']
DAYS = ['Пн', 'Вт', 'Ср', 'Чт', 'Пт']
HOURS = ['08:30', '10:15', '12:35', '14:20', '16:05', '17:45']
ROOMS = ['101', '102', '201', '202']

# Фейкові предмети і викладачі
SUBJECTS = {
    'УПКН-21': {'Бази Даних': ['Лк', 'Пз'], 'Математика': ['Лк', 'Пз'],
                'Програмування': ['Лк', 'Пз'], 'Історія': ['Лк', 'Пз'],
                'Англійська мова': ['Пз'], 'Веб-дизайн': ['Лк', 'Пз']},
    'КН-22': {'Бази Даних': ['Лк', 'Пз'],
              'Програмування': ['Лк', 'Пз'], 'Математика': ['Лк', 'Пз'], 'Історія': ['Лк', 'Пз'],
              'Англійська мова': ['Пз'], 'Веб-дизайн': ['Лк', 'Пз']}
}
TEACHERS = {'Математика': 'Іваненко А.В.', 'Програмування': 'Петров І.С.',
            'Бази Даних': 'Сидоренко К.Б.', 'Історія': 'Коваленко Т.С.', 'Англійська мова': 'Жук П.К.',
            'Веб-дизайн': 'Вовчок П.К.',}
```

Рисунок 3.1 – Ініціалізація змінних

В основі реалізації лежить функція «generate_schedule_ortools(…)», в яку передаються вхідні змінні та яка створює математичну модель задачі. Для кожної комбінації (група, предмет, тип заняття) створюється змінна, що відповідає за її часовий слот (slot_vars) та аудиторію (room_vars), (рис. 3.2). [8]

```python
def generate_schedule_ortools(groups, types, subjects, teachers, rooms, days, hours):
    model = cp_model.CpModel()

    # Параметри
    num_days = len(days)
    num_hours = len(hours)
    num_time_slots = num_days * num_hours

    # Допоміжна функція для перетворення індексу в день + годину
    def slot_to_day_hour(slot_index):
        return days[slot_index // num_hours], hours[slot_index % num_hours]

    # Змінні: (група, предмет) -> слот, кімната
    slot_vars = {}
    room_vars = {}
    shared_lecture_slots = {}
    shared_lecture_rooms = {}
```

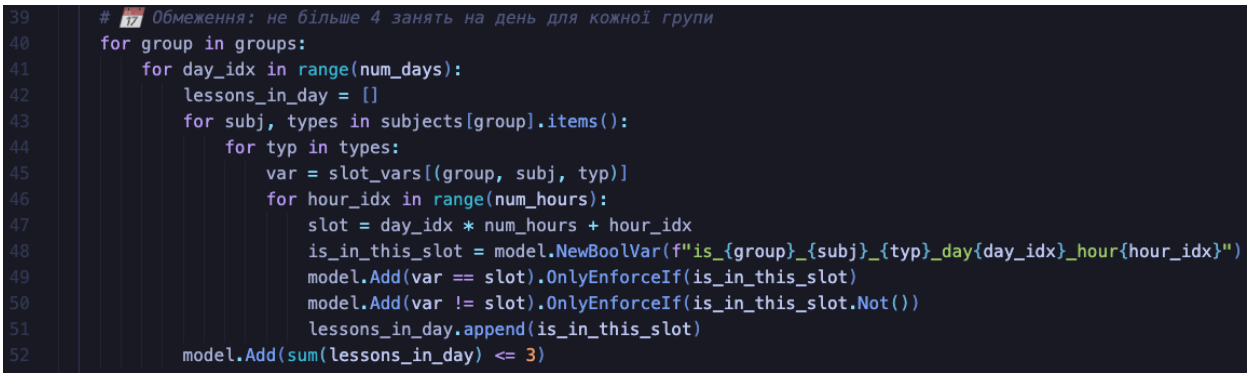
Рисунок 3.2 – Створення додаткових змінних у функцію

Особливістю моделі є те, що лекції кількох груп однієї спеціальності, наприклад, загальна лекція для двох підгруп. Вони мають один спільний часовий слот та аудиторію, що задається через відповідно спільну змінну (рис. 3.3). Для інших типів занять створюються окремі змінні для кожної групи.

```python
# 🔧 Створення змінних для кожного (group, subject, type)
for group in groups:
    for subj, types in subjects[group].items():
        for typ in types:
            if typ == 'Лк':
                if (subj, typ) not in shared_lecture_slots:
                # Створити один слот і кімнату на всю лекцію
                    shared_lecture_slots[(subj, typ)] = model.NewIntVar(0, num_time_slots - 1, f"slot_shared_{subj}_{typ}")
                    shared_lecture_rooms[(subj, typ)] = model.NewIntVar(0, len(rooms) - 1, f"room_shared_{subj}_{typ}")
                # Присвоїти цю змінну групі
                slot_vars[(group, subj, typ)] = shared_lecture_slots[(subj, typ)]
                room_vars[(group, subj, typ)] = shared_lecture_rooms[(subj, typ)]
            else:
            # Звичайна (не лекція) — окрема змінна для кожної групи
                var_name = f"{group}_{subj}_{typ}"
                slot_vars[(group, subj, typ)] = model.NewIntVar(0, num_time_slots - 1, f"slot_{var_name}")
                room_vars[(group, subj, typ)] = model.NewIntVar(0, len(rooms) - 1, f"room_{var_name}")
```

Рисунок 3.3 – Окремі змінні для кожної групи

У моделі враховано ключові обмеження (рис. 3.4), а саме академічна група не може мати більше трьох занять на день; не використовувати аудиторію одночасно для двох занять; якщо кілька груп мають спільну лекцію, то це допускається, але лише за умови, що аудиторія одна і та сама.



```python
# 📅 Обмеження: не більше 4 занять на день для кожної групи
for group in groups:
    for day_idx in range(num_days):
        lessons_in_day = []
        for subj, types in subjects[group].items():
            for typ in types:
                var = slot_vars[(group, subj, typ)]
                for hour_idx in range(num_hours):
                    slot = day_idx * num_hours + hour_idx
                    is_in_this_slot = model.NewBoolVar(f"is_{group}_{subj}_{typ}_day{day_idx}_hour{hour_idx}")
                    model.Add(var == slot).OnlyEnforceIf(is_in_this_slot)
                    model.Add(var != slot).OnlyEnforceIf(is_in_this_slot.Not())
                    lessons_in_day.append(is_in_this_slot)
        model.Add(sum(lessons_in_day) <= 3)
```

Рисунок 3.4 – Створення обмежень

Також потрібно створити обмеження, яке уникає різні конфлікти (рис. 3.5). Найважливіші конфлікти, які можуть виникнути в програмі: у групи не може бути два заняття одночасно; викладач не може викладати більше одного предмету в один і той самий час.

```python
# Обмеження: уникаємо конфліктів
all_lessons = list(slot_vars.keys())

for i in range(len(all_lessons)):
    for j in range(i + 1, len(all_lessons)):
        g1, s1, t1 = all_lessons[i]
        g2, s2, t2 = all_lessons[j]

        is_same_subject = s1 == s2
        is_lecture = t1 == t2 == 'Лк'

        if g1 == g2:
            model.Add(slot_vars[(g1, s1, t1)] != slot_vars[(g2, s2, t2)])

        if teachers[s1] == teachers[s2]:
            if not (is_same_subject and is_lecture):
              model.Add(slot_vars[(g1, s1, t1)] != slot_vars[(g2, s2, t2)])

        if not (is_same_subject and is_lecture):
            slots_differ = model.NewBoolVar(f"slot_diff_{g1}_{s1}_{t1}_{g2}_{s2}_{t2}")
            rooms_differ = model.NewBoolVar(f"room_diff_{g1}_{s1}_{t1}_{g2}_{s2}_{t2}")

            model.Add(slot_vars[(g1, s1, t1)] != slot_vars[(g2, s2, t2)]).OnlyEnforceIf(slots_differ)
            model.Add(slot_vars[(g1, s1, t1)] == slot_vars[(g2, s2, t2)]).OnlyEnforceIf(slots_differ.Not())

            model.Add(room_vars[(g1, s1, t1)] != room_vars[(g2, s2, t2)]).OnlyEnforceIf(rooms_differ)
            model.Add(room_vars[(g1, s1, t1)] == room_vars[(g2, s2, t2)]).OnlyEnforceIf(rooms_differ.Not())

            model.AddBoolOr([slots_differ, rooms_differ])
```

Рисунок 3.5 – Обмеження конфліктів

Після створення математичної моделі та усіх змінних для формування навчального розкладу, а також накладання обмежень на модель, використовується CP-SAT Solver з бібліотеки Google OR-Tools. [16]. Ця функція допомагає ефективно шукати допустимі рішення, беручи до уваги всі обмеження задачі (рис. 3.6).

```python
# Рішення
solver = cp_model.CpSolver()
status = solver.Solve(model)
```

Рисунок 3.6 – Розв’язок за допомогою CP-SAT Solver

Коли рішення було знайдене «status == cp_model.FEASIBLE or status == cp_model.OPTIMAL», створюється словник для остаточного рішення навчального розкладу, де кожна група має список занять, які відсортовані по дням та часовим слотам (рис. 3.7).

```
    if status == cp_model.FEASIBLE or status == cp_model.OPTIMAL:
        result = {}
        for group in groups:
            print(f"\n📅 Розклад для групи {group}:")
            group_schedule = []
            for subj, types in subjects[group].items():
                for typ in types:
                    slot = solver.Value(slot_vars[(group, subj, typ)])
                    room_index = solver.Value(room_vars[(group, subj, typ)])
                    day_idx = slot // len(hours)
                    hour_idx = slot % len(hours)
                    day, hour = slot_to_day_hour(slot)
                    teacher = teachers[subj]
                    room = rooms[room_index]
                    group_schedule.append({
                        "day_idx": day_idx,
                        "hour_idx": hour_idx,
                        "day": day,
                        "hour": hour,
                        "subject": subj,
                        "type": typ,
                        "teacher": teacher,
                        "room": room
                    })
            group_schedule.sort(key=lambda x: (x["day_idx"], x["hour_idx"]))
```

Рисунок 3.7 – Сортування розкладу

Результат розв’язання потім обробляється сервером, і у зручній формі передається у фронтенд (рис. 3.8), де виводиться користувачу на екран у вигляді простої таблиці або списку занять із зазначенням днів, часу, предметів, викладачів, і аудиторій.

```python
def schedule_view(request):
    form = GroupSelectForm()
    form.fields['group'].choices = [(g, g) for g in GROUPS]
    schedule = None

    if request.method == 'POST':
        form = GroupSelectForm(request.POST)
        form.fields['group'].choices = [(g, g) for g in GROUPS]
        if form.is_valid():
            group = form.cleaned_data['group']
            result = generate_schedule_ortools(
                groups=GROUPS,
                types=['Лк', 'Пз'],
                subjects=SUBJECTS,
                teachers=TEACHERS,
                rooms=ROOMS,
                days=DAYS,
                hours=HOURS
            )
            if result:
                schedule = result.get(group, [])

    return render(request, 'scheduler/schedule.html', {
        'form': form,
        'schedule': schedule,
    })
```

Рисунок 3.8 – Передача результатів у фронтенд

Ця реалізація дає змогу будувати і гнучкий навчальний розклад, який може легко масштабуватись на більшу кількість груп, викладачів і занять. Окрім цього, модель дозволяє у майбутньому додавати нові обмеження, наприклад, переваги викладачів щодо днів або годин, у яких вони бажають працювати, або врахування вільних аудиторій тощо.

Таким чином, програма пропонує потужні алгоритмічні методи, зручність роботи з даними через веб-інтерфейс і гнучкість у налаштуванні параметрів навчального плану занять, що може забезпечити ефективне створення адаптивного та коректного розкладу закладів освіти.

3.4 Керівництво користувача

Щоб зручно взаємодіяти з системою автоматизованого плану навчальних занять, створено інтуїтивно зрозумілий веб-інтерфейс. Він складається з трьох основних сторінок:

- «Логін» сторінка (рис. 3.10).
- «Загальна» сторінка (рис. 3.11).
- Сторінка «Генерація розкладу» (рис. 3.12).

Перед тим як почати працювати з веб-сторінкою, користувач повинен переконатися, що всі програмні забезпечення встановлено (Python, Django, Google OR-Tools, PostgreSQL, Visual Studio Code тощо), а також запущено серверну частину коду [7, 8, 10, 15]. Для запуску системи необхідно зробити наступні дії:

1. Відкити термінал у Visual Studio Code або командний рядок.
2. Перейти до директорії проєкту, де розміщена Django-програма [15].
3. Потім запустити сервер за допомогою команди «python manage.py runserver» (рис. 3.9).

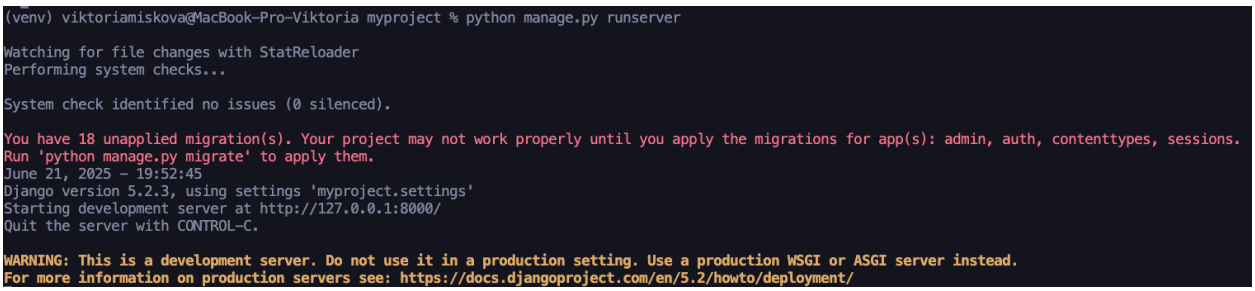


Рисунок 3.9 – Запуск сервера

4. У браузері (Google Chrome, Firefox або Safari) перейти за адресою «http://127.0.0.1:8000/schedule_app/».
5. Після відкриття веб-сайту, користувач автоматично потрапляє на «Логін» сторінку.

Сторінка входу/реєстрації - це місце в додатку або на веб-сайті, де людина може зареєструвати новий обліковий запис або увійти в існуючий [16]. Ця сторінка дозволяє платформі розпізнавати своїх користувачів і пропонувати їм персоналізовані послуги або контрольований доступ до певних функцій.

Значна частина додатків і веб-сайтів використовує цей метод для управління користувачами, полегшуючи отримання даних і налаштувань, зберігаючи при цьому безпеку. Платформи залежать від цих сторінок, щоб розрізняти користувачів і забезпечувати безпеку облікових записів. [16]

Сторінка реєстрації облікового запису зазвичай містить важливу інформацію, необхідну для створення облікового запису користувача [16]. Сюди входить адреса електронної пошти або ім'я користувача разом з паролем. Також може знадобитися основна інформація про профіль, наприклад, ваше повне ім'я, залежно від вимог сервісу. Потім ці дані використовуються платформою для ідентифікації користувача, захисту облікового запису та персоналізації користувацького досвіду.

Різні типи сервісів мають різні рівні вимог. Більш критичні сервіси можуть вимагати більше інформації для верифікації та безпеки. Однак, при введенні інформації слід уникати надання занадто великої кількості персональних даних. Рекомендується заповнювати лише обов'язкові поля, якщо ви не можете впевнено пояснити, чому потрібна додаткова інформація, і не довіряєте суб'єкту, який її запитує. [16]

У програмі автоматизованого розкладу занять, «Логін» сторінка (рис. 3.10) виконує функцію безпеки та обмеження доступу. Завдяки авторизації, тільки користувачі, які вже зареєстровані або мають довірені права на цій сторінці, можуть вносити зміни до системи, запускати генерацію складання плану навчальних занять, а також переглядати результати. Це дуже важливо у навчальному закладі, де зміни до розкладу мають здійснюватися лише адміністраторами або іншими відповідальними особами.

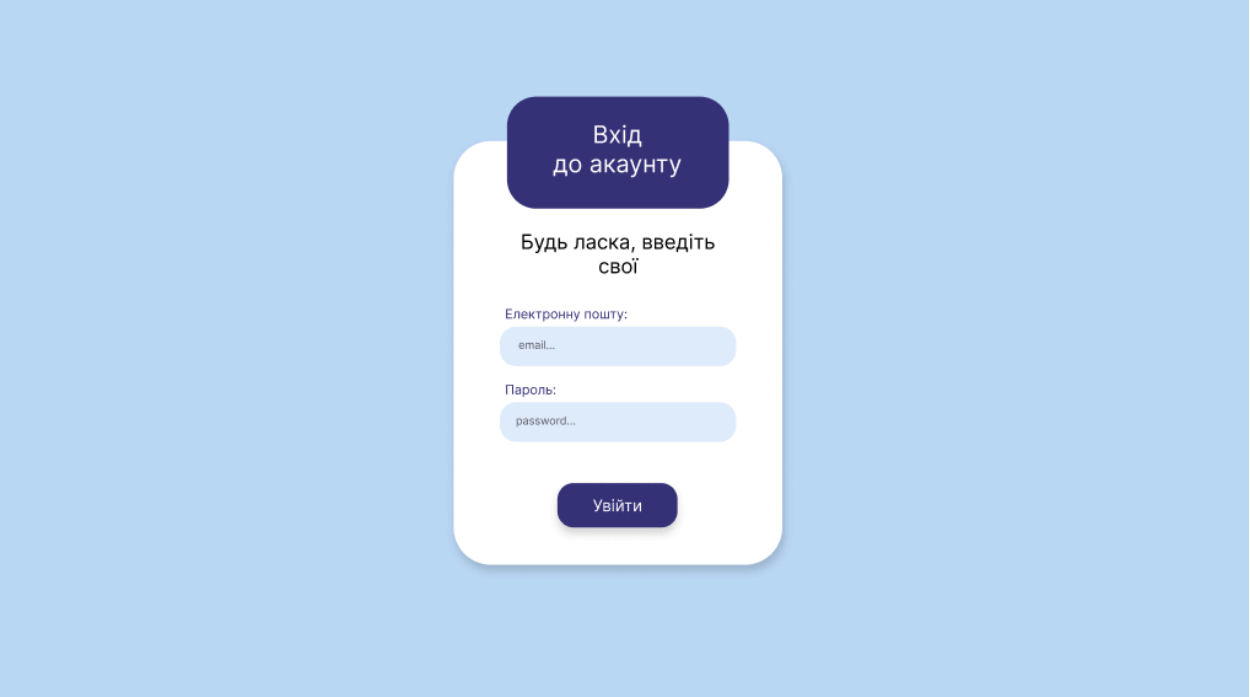


Рисунок 3.10 – "Логін" сторінка

На цій сторінці користувачу необхідно ввести свої облікові дані, а саме пароль та електронну пошту. Якщо авторизація пройде успішно, його буде перенаправлено на «Загальна» сторінку з меню. У разі дані були введені неправильно, користувач отримає повідомлення про помилку і буде запропоновано повторити спробу входу.

Після входу користувач потрапляє на «Загальна» сторінку. Вона є навігаційним центром системи та містить головне меню, яке дозволяє переходити до основних функціональних блоків програми. Вона включає в себе такі пункти (рис. 3.11):

- Календар
- Розклад
- Успішність
- Налаштування

Веб-сторінка надає логічну структуру для переходу між розділами системи та забезпечує зручність у користуванні.

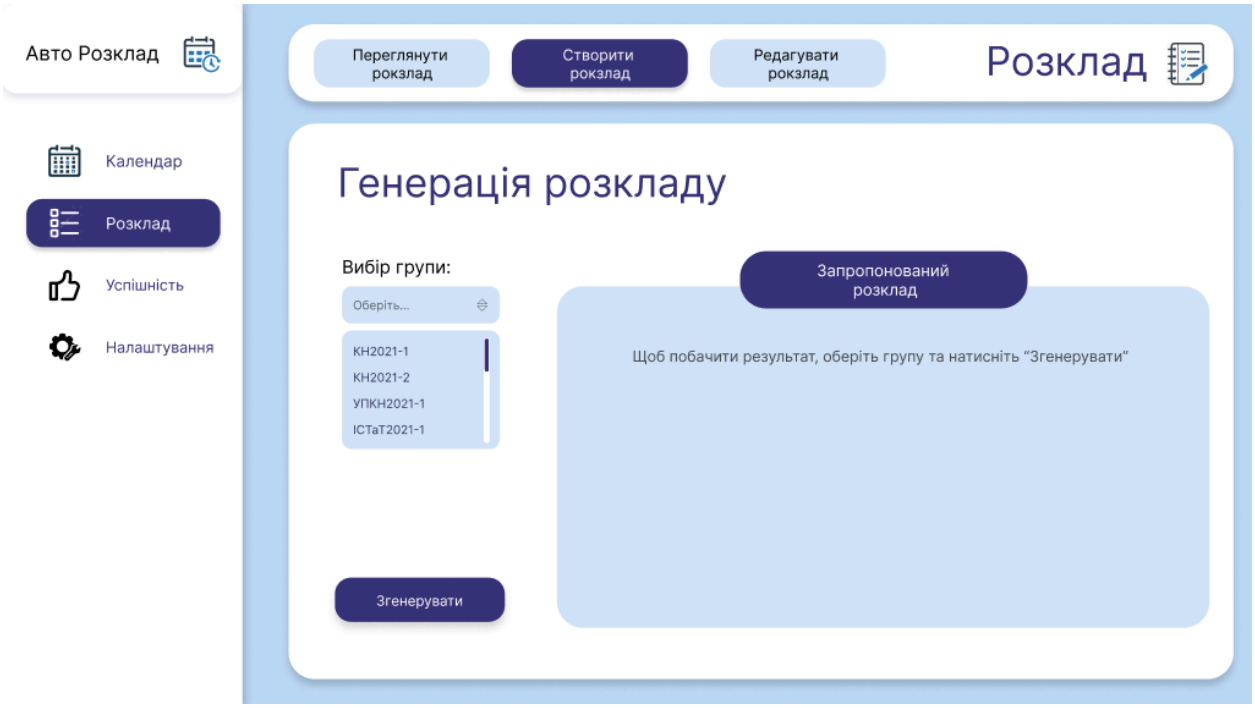


Рисунок 3.11 – Загальна сторінка з відкритим меню "Розклад"

Після того, як користувач нажимає на кнопку «Розклад» в меню з права, він потрапляє на «Генерація розкладу» сторінку. Саме тут реалізована головна функція системи – створення оптимального навчального плану на основі попередньо введених даних. Інтерфейс цієї сторінки містить випадаюче вікно, де адміністратор обирає академічну групу, для якої потрібно створити навчальний розклад. Також тут знаходиться невеличкий блок, де показано детальний список згенерований план занять, який сортированний по днях та годинах та показано предмет, його тип, викладача та аудиторію. Окрім цього, зверху є три кнопки, які спрямовують користувача до сторінки «Переглянути розклад», «Створити розклад» та «Редагувати розклад».

Після натискання кнопки «Згенерувати», система передає дані до обчислюваного модуля, який має основу на Google OR-Tools, де відбувається обробка та побудова розкладу [7]. У разі успішного завершення, результат можна побачити на невеликому полі у вигляді зручного розкладу. Якщо ж алгоритм не може знайти допустиме рішення, користувач отримає відповідне повідомлення, щоб перевірити введені параметри.

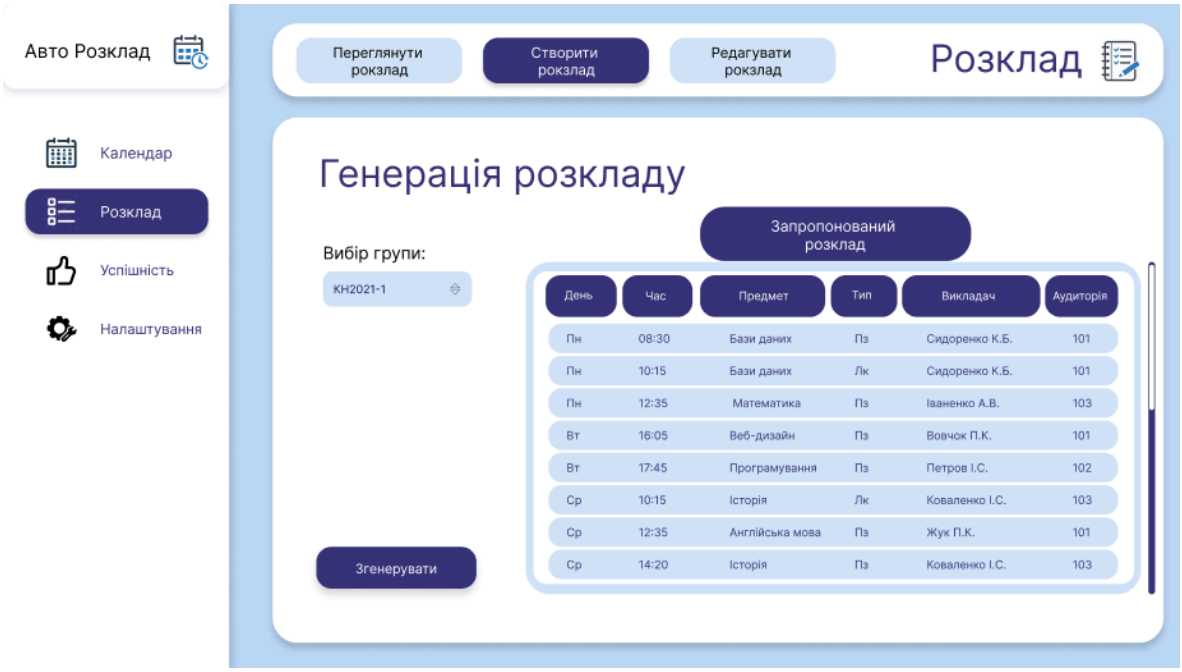


Рисунок 3.12 – Сторінка "Генерація розкладу"

Висновки до розділу

У цьому розділі було розглянуто основну функцію системи автоматизації навчального розкладу. Також було описано, які саме програмні та технічні забезпечення (Python, Django, Google OR-Tools, PostgreSQL, Visual Studio Code тощо) потрібні для того, щоб реалізувати цей проєкт [7, 8, 10, 15]. Сама ж розробка здійснювалася за допомогою використання відкритих, сучасних і потужних інструментів, які зазначено вище. Реалізована система може бути легко адаптована, як до вищих навчальний закладів, так і загальноосвітніх шкіл та гімназій. Вона має простий інтерфейс та ефективно вирішує завдання планування розкладу, при тому зберігає логіку плану без конфліктів.

До того ж, веб-сторінка є зрозумілою для кожного користувача, є послідовною та зручною, без глибоких технічних знань. Авторизація забезпечує безпечний доступ, головна сторінка дає змогу орієнтуватися у функціоналі, а сторінка генерації надає ефективний інструмент для формування розкладу з урахуванням обмежень.

РОЗДІЛ 4
ОХОРОНА ПРАЦІ

4.1 Регулювання питань охорони праці на законодавчому рівні

Охорона праці – це система заходів та засобів, спрямованих на збереження життя та здоров'я працівників у процесі трудової діяльності.

Роль охорони праці у формуванні безпеки робочих місць багатоаспектна та включає наступні напрямки.

У рамках створення безпечних умов праці охорона праці забезпечує:

- проектування та організацію робочих місць з урахуванням ергономіки, санітарних норм, стандартів безпеки;

- оснащення робочих місць справним обладнанням, інструментами та засобами захисту;

- контроль за безпекою виробничих процесів.

Реалізація цих положень дозволяє знизити ймовірність травм та профзахворювань.

Система охорони праці включає ідентифікацію та оцінку ризиків на робочих місцях:

- ризики аналізуються (наприклад, ризики падіння, ураження струмом, впливу шкідливих речовин);

- розробляються заходи щодо їх зниження чи усунення (інженерні рішення, ЗІЗ, організаційні заходи).

Охорона праці визначає обов'язкові правила поведінки, технології виконання і порядок використання устаткування:

- формуються інструкції з охорони праці для кожного робочого місця;

- проводяться регулярні перевірки стану робочих місць (виробничий контроль, аудит, атестація робочих місць за умовами праці).

Охорона праці забезпечує підготовку та інформування працівників:

- проведення вступного та періодичного інструктажу;

- навчання безпечним прийомам роботи;

- доведення до працівників інформації про ризики та заходи захисту.

Це формує у працівників культуру безпеки та усвідомлену поведінку на робочому місці.

Заходи з охорони праці сприяють безперервному покращенню системи безпеки:

- виявлення та усунення причин аварій та нещасних випадків;

- впровадження нових технологій та засобів для підвищення рівня безпеки;

- вдосконалення нормативної бази всередині організації.

Реалізація зазначених положень неможлива без формування відповідної бази нормативних документів, починаючи з міжнародного рівня.

На міжнародному рівні величезну роль у сфері охорони праці відіграє діяльність Міжнародної організації праці (МОП). МОП розробляє конвенції, рекомендації та інші акти, які визначають міжнародні стандарти в галузі охорони праці. До ключових конвенцій МОП, які безпосередньо стосуються охорони праці, відносяться:

- Конвенція № 155 (1981 р.) «Про безпеку та гігієну праці та виробниче середовище»: встановлює загальні засади національної політики у галузі безпеки та гігієни праці, вимагає створення національних систем управління охороною праці;

- Конвенція №187 (2006 р.) «Про рамкову політику в галузі безпеки та гігієни праці»: спрямована на постійне покращення системи охорони праці, запроваджує концепцію «культури профілактики у галузі охорони праці»;

- Конвенція № 81 (1947 р.) «Про інспекцію праці в промисловості та торгівлі»: регулює діяльність державних інспекцій праці, включаючи контроль за дотриманням правил охорони праці;

- Конвенція №1 (1919 р.) «Про тривалість робочого часу у промисловості»: перша конвенція МОП, яка встановлює обмеження робочого часу, що безпосередньо пов'язано з умовами праці;

- Конвенція № 119 (1963 р.) «Про захисні пристрої на машинах»: вимагає оснащення обладнання пристроями, які запобігають травмам;

- Конвенція № 127 (1967 р.) «Про максимальну вагу, що переноситься одним працівником»: регулює норми ручного підйому та переміщення вантажів;

- Конвенція № 161 (1985 р.) «Про організацію служб гігієни праці»: потребує створення служб охорони здоров'я працівників на підприємствах.

Рекомендації МОП пояснюють, як застосовувати положення конвенцій практично. Наприклад:

- Рекомендація № 164 (до Конвенції № 155) щодо заходів щодо покращення охорони праці;

- Рекомендація № 197 (до Конвенції № 187) щодо розвитку національної політики у сфері охорони праці.

Оскільки Україна є членом Міжнародної організації праці, то, ратифікувавши Конвенції МОП, вона зобов’язалася впровадити їх положення у національне законодавство, що чітко прослідковується у основних законодавчих документах країни у сфері охорони праці.

Так, у Кодексі законів про працю України у Главі IV «Робочий час» стаття 50 визначає норми тривалості робочого часу – не більше 40 годин на тиждень, що корелює з Конвенцією №1, а статті 10-11 закону «Про охорону праці» визначають питання охорони праці жінок та неповнолітніх, в тому числі їх залучення до робіт, пов’язаних із підійманням та перенесенням речей, маса яких перевищує встановлені для них граничні норми, що також корелює із іншою конвенцією – Конвенція №127.

Конвенція 155 знайшла своє відображення у статті 13 закону «Про охорону праці», яка відзначає необхідність забезпечення функціонування системи управління охороною праці на підприємстві з метою реалізації

завдання із створення безпечних та здорових умов праці для усіх працівників, незалежно від виду їх діяльності.

4.2 Виявлення потенційних небезпек стосовно об’єкту проектування

У дипломній роботі бакалавра розглядається питання розробки системи автоматичного розкладу занять для університету, що є корисним ресурсом як для студентів, так і для викладачів. У розділі з охорони праці можна розглянути робоче місце викладача як користувача системи розкладу занять.

Тип роботи: переважно розумова праця (лекції, семінари, підготовка матеріалів, наукова робота), з елементами адміністративних функцій та взаємодії з аудиторією.

Робоче місце викладача це аудиторія, кабінет, інколи лабораторія чи комп'ютерний клас.

Таблиця 4.1 – Результати аналізу умов праці викладача

Фактор	Характеристика	Можливі наслідки для здоров’я
Мікроклімат (температура, вологість, вентиляція)	Перегрів/переохолодження викладачів під час перебування у приміщеннях, сухе повітря, що впливає на слизові оболонки очей та носоглотки	Стомлюваність, зниження концентрації, захворювання верхніх дихальних шляхів, головний біль
Освітленість (природна та штучна)	Недостатнє чи надмірне освітлення, відблиски від моніторів	Втома зору, головний біль, зниження гостроти зору (астенопія)
Шум	Шум в аудиторіях, коридорах, від роботи оргтехніки	Стомлюваність, зниження працездатності, дратівливість, порушення сну

Продовження таб. 4.1

Електромагнітні випромінювання (від комп'ютерів, Wi-Fi, проекторів)	Низькоінтенсивні, але тривалі впливи	Можливий вплив на нервову систему, підвищена стомлюваність
Монотонність та статичне навантаження	Тривале сидіння за комп'ютером, малорухливість, статична поза біля дошки	Хвороби хребта (остеохондроз, сколіоз), захворювання суглобів, варикоз, м'язові болі
Психоемоційні навантаження	Спілкування з великою аудиторією, стрес через відповідальність, дедлайни, конфлікти	Неврози, емоційне вигоряння, гіпертонія, депресії
Робота з оргтехнікою та електроприладами	Ризик ураження електричним струмом (малоймовірно при справній техніці)	Електротравми у випадку порушення правил експлуатації
Хімічні речовини (під час роботи в лабораторії)	Реагенти, пара хімічних речовин (якщо є лабораторна робота)	Алергії, захворювання дихальних шляхів, отруєння (у разі аварійних ситуацій)

Таким чином, можна зробити висновок про наявність ряду небезпек, характерних для роботи викладачів, які впливають на їх стан здоров’я.

4.3 Дослідження ризику реалізації потенційних небезпек на об’єкті проектування та розробка заходів щодо їх попередження

Професійні ризики – це ризики, які випливають із професійних небезпек.

Практична, всеосяжна стратегія управління ризиками потребує часу та зусиль. Вона включає три сегменти – виявлення ризиків, оцінку ризиків і визначення пріоритетів ризиків – кожен з яких вимагає ретельності уваги та відповідних знань та навичок від посадових осіб, які реалізовують цю процедуру. Внесок працівників є невід'ємною частиною процесу, оскільки вони безпосередньо наражаються на вплив небезпек, тому залучення їх до процедури оцінки ризиків, зокрема, до складових, які стосуються ідентифікації небезпек, є особливо важливим.

Поширені помилки у типових підходах до управління ризиками – це ігнорування дрібниць та зосередження лише на великих небезпеках. Хоча, безумовно, важливо пом'якшувати ризики, що загрожують життю, в той же час

робота роботодавця полягає в тому, щоб стежити за всіма небезпеками на робочому місці, незалежно від їх наслідків – для цього у процедурі оцінки ризику використовується підхід із урахуванням пріоритетності ризиків, що дозволяє встановити ієрархію небезпек та прийняти відповідні рішення.

Методологія оцінки ризиків – це систематичний підхід до виявлення, оцінки та пріоритезації ризиків. Він вимагає стратегічного аналізу основних напрямів бізнесу, включаючи бізнес-операції та ініціативи щодо забезпечення відповідності. Цей структурований метод оцінки допомагає організаціям розробляти дієві стратегії для усунення потенційних загроз і вразливостей у своїх системах, узгоджуючи ширші цілі управління ризиками.

Організації можуть посилити свою позицію управління ризиками, включивши визнані стандарти та керівництва, а також створивши матриці ризиків, використовуючи як якісні, так і кількісні підходи, один з яких розглядається у розділі.

Таблиця 4.2 – Матриця оцінювання ризиків

Психоемоційні навантаження				
Визначення категорії серйозності небезпеки		Визначення рівня ймовірності небезпеки		Індекс ризику небезпеки
Вид, категорія	Опис	Вид, рівень	Опис	
III – гранична	Неврози, емоційне вигоряння, гіпертонія, депресії	А	Стани, що виникають часто та пов’язані із особливістю професійної діяльності викладача	IIIА – неприпустимий (надмірний) рівень ризику
Монотонність та статичні навантаження				
Визначення категорії серйозності небезпеки		Визначення рівня ймовірності небезпеки		Індекс ризику небезпеки
Вид, категорія	Опис	Вид, рівень	Опис	
III – гранична	Хвороби хребта (остеохондроз, сколіоз), захворювання суглобів, варикоз, м'язові болі	А	Стани, що виникають часто та пов’язані із особливістю професійної діяльності викладача	IIIА – неприпустимий (надмірний) рівень ризику

Таким чином, проведений аналіз та оцінка ризиків свідчать про актуальність питання надання рекомендацій щодо зниження впливу аналізованих небезпек.

До основних організаційних заходів із зниження негативного впливу фактору статичних навантажень можна віднести:

- введення обов'язкових коротких перерв (5–10 хвилин) кожні 45-60 хвилин для розминки та зміни пози;

- використання принципу «динамічної паузи»: вставати та рухатися між парами, лекціями, зміною слайдів тощо;

- планування робочого часу так, щоби змінювалися види навантаження: робота за комп'ютером, лекції, практичні заняття, адміністративна робота;

- використання мультимедіа та інтерактивних форм навчання, які дозволяють викладачеві більше рухатися (наприклад, робота з дошкою, демонстрація практичних дій).

До основних технічних та ергономічних рішень із зниження негативного впливу фактору статичних навантажень можна віднести:

- регульоване по висоті крісло з підтримкою попереку;

- регульований по висоті стіл (по можливості з функцією робочого місця, що стоїть);

- правильне розташування монітора (на рівні очей, на відстані 50-70 см);

- використання допоміжних пристроїв: підставка для ніг, якщо крісло високе, тримачі для книг та документів, щоб уникнути постійного нахилу голови.

4.4 Висновки по розділу

У розділі з охорони праці приділено увагу аналізу законодавства з охорони праці проведено паралель між міжнародним та національним законодавством, що свідчить про його актуальність та відповідність високим вимогам.

Також у розділі проведено аналіз умов праці на робочому місці викладача університету як користувача розроблюваної системи автоматичного розкладу занять, визначено основні небезпеки наслідки їх впливу.

Для такої небезпеки, як статичні навантаження, проведено оцінку ризиків за допомогою матриці оцінки ризиків та розглянуто ряд рекомендацій із покращення умов праці.

ВИСНОВКИ

Під час виконання дипломної роботи було проведено аналіз предметного середовища, де обґрунтовано вибір розробки нової інформаційної системи з використанням бібліотеки Google OR-Tools для вирішення задач комбінованої оптимізації та пошуку в умовах обмежень. Крім цього, було розроблено та реалізовано систему автоматичного складання розкладу за допомогою бібліотеки Google OR-Tools, що дозволило практично показати ефективність сучасних інструментів комбінаторної оптимізації для вирішення складних задач планування у сфері освіти.

Дослідження показало, що задача створення розкладу потребує значних обчислювальних ресурсів при наявності великої кількості даних та обмежень. Внаслідок цього використання спеціалізованих алгоритмів та бібліотек оптимізації є не лише доцільним, а й необхідним для забезпечення якісного рішення.

У ході реалізації програми для автоматичного формування розкладу було проаналізовано вимоги до навчального розкладу, визначено основні обмеження та критерії оптимальності. До того ж, досліджено функціонал функції CP-SAT Solver з бібліотеки Google OR-Tools; реалізовано прототип системи, що обробляє вхідні дані та генерує розклад, спираючись на доні обмеження до задачі. Створено веб-сторінку для користування цією системою.

Отримані результати показують практичну цінність запропонованого підходу, а саме автоматизація процесу дозволяє значно зменшити кількість ручної праці, мінімізувати помилки та адаптувати зміни до розкладу, що дуже важливі у динамічному освітньому середовищі.

Як показує дослідження, якісно укладений розклад допомагає рівномірно розподілити навантаження як між академічними групами, так і серед професорсько-викладацького складу. Від ефективності розкладу залежить успішність викладання, рівень засвоєння матеріалу студентами, а

також раціональне використання, як інтелектуальних, так і матеріальних ресурсів навчального закладу. Таким чином, дипломна робота досягла поставленої мети та відкриває можливості для подальшого вдосконалення системи в напрямі її практичного впровадження у закладах освіти.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Планування роботи розкладу. [Електронний ресурс] // Режим доступу: https://vseosvita.ua/c/pedagogy/post/44283

2. Продуктивність уроків. [Електронний ресурс] // Режим доступу: http://pohybliak.edukit.ck.ua/navchaljnij_proces/vimogi_do_skladannya_rozkladu/

3. Положення про розклад занять. Головна сторінка: https://www.kname.edu.ua/images/Files/Normativny_Dokumenty/pologennya_pro_rosklad_zanyat.pdf

4. Timetables. [Електронний ресурс] // Режим доступу: https://www.asctimetables.com/?goto=share_timetables

5. UniTime. [Електронний ресурс] // Режим доступу: https://www.apereo.org/programs/software/unitime

6. Розклад уроків. [Електронний ресурс] // Режим доступу: https://school24-7.blogspot.com/2017/04/blog-post_7.html

7. Google OR-Tools. [Електронний ресурс] // Режим доступу: https://developers.google.com/optimization#:~:text=OR%2DTools%20is%20an%20open,linear%20programming%2C%20and%20constraint%20programming

8. Документація Python. [Електронний ресурс] // Режим доступу: https://docs.python.org/3/whatsnew/3.13.html

9. Поняття бази даних. [Електронний ресурс] // Режим доступу: https://vseosvita.ua/lesson/poniattia-bazy-danykh-osnovni-funktsii-upravlinnia-bazamy-danykh-74830.html

10. Системи керування базами даних. [Електронний ресурс] // Режим доступу: https://www.miyklas.com.ua/p/informatica/9-klas/bazi-danikh-sistemi-keruvannia-bazami-danikh-361840/sistemi-keruvannia-bazami-danikh-352450/re-b0f14256-7ac2-4823-9f53-656061b85eaa

11. Математичне програмування. [Електронний ресурс] // Режим доступу: https://www.sciencedirect.com/topics/computer-science/mathematical-programming

12. Різниця між математичним та комп'ютерним програмуванням. [Електронний ресурс] // Режим доступу: https://www.gurobi.com/resources/lp-chapter-1-mathematical-programming/

13. Проблема задоволення обмежень. [Електронний ресурс] // Режим доступу: https://www.geeksforgeeks.org/constraint-satisfaction-problems-csp-in-artificial-intelligence/

14. Google OR-Tools' CP-SAT Solver. [Електронний ресурс] // Режим доступу: https://d-krupke.github.io/cpsat-primer/

15. Документація Django. [Електронний ресурс] // Режим доступу: https://www.djangoproject.com/

16. Сторінка входу та реєстрації. [Електронний ресурс] // Режим доступу: https://hocoos.com/answers/what-is-a-website-login-signup-page/

17. International Labour Organisation: List of conventions. – Режим доступу: https://normlex.ilo.org/dyn/nrmlx_en/f?p=NORMLEXPUB:12200:0::NO:::.

18. Кодекс законів про працю України. . – Офіційний сайт Верховної Ради України. – Режим доступу: https://zakon.rada.gov.ua/laws/show/322-08.

19. Закон України «Про охорону праці». – Офіційний сайт Верховної Ради України. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2694-12.

20. Кульчинський О.В. Охорона праці в закладах освіти: методичний посібник / Відділ освіти Новосанжарської РДА, 2010 р. – 64 с.