

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістра

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ
ВИЗНАЧЕННЯ ТРИВАЛОСТІ НАВЧАННЯ
RESEARCH AND SOFTWARE IMPLEMENTATION OF METHODS FOR
DETERMINING THE DURATION OF STUDY

Виконав: студент(ка) 2 курсу, групи КНТ-219м
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Системи штучного інтелекту

Костюченко І.В.

(прізвище та ініціали)

Керівник Льовкін В.М.

(прізвище та ініціали)

Рецензент Зеленьова І.Я.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ

Кафедра програмних засобів

Ступінь вищої освіти магістр

Спеціальність 122 Комп'ютерні науки

(код і найменування)

Освітня програма (спеціалізація) Системи штучного інтелекту

(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.

С.О. Субботін

“ ” 2020 року

ЗАВДАННЯ

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Костюченка Івана Вячеславовича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів визначення тривалості навчання

Research and Software Implementation of Methods for Determining the Duration of Study

керівник проєкту (роботи) Льовкін Валерій Миколайович, к.т.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “3” листопада 2020 року № 301

2. Строк подання студентом проєкту (роботи) грудень 2020 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз проблеми підтримки дистанційного навчання. 2. Матеріали і методи. 3. Проєктування програмного забезпечення. 4. Основні рішення щодо реалізації компонентів системи. 5. Експлуатація та тестування програми. 6. Економіко-організаційна частина, 7. Охорона праці та безпека в надзвичайних ситуаціях.

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

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Льовкін В.М. доцент		
Економіко-організаційна частина	Пожуєва Т.О., професор		
Охорона праці та безпека в надзвичайних ситуаціях	Коробко О.В., ст. викладач		
Нормоконтролер	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання “ 7 ” вересня 2020 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз та дослідження предметної області	1-2 тиждень	
2	Розробка та затвердження технічного завдання	3 тиждень	Технічне завдання
3	Дослідження методів розв’язання проблеми	4 тиждень	
4	Проектування структури програмного забезпечення	5 тиждень	Структурна схема
5	Розроблення схеми функціонування програмного забезпечення та модулів	6-7 тиждень	Функціональна схема. Модулі
6	Розроблення програмного коду	8 тиждень	Текст програми
7	Тестування та відлагодження програмного забезпечення	9 тиждень	Програма
8	Розробка розділів пояснювальної записки	10-11 тиждень	Пояснювальна записка
9	Розробка графічного матеріалу	12 тиждень	Графічний матеріал
10	Захист магістерської роботи	13 тиждень	

Студент(ка)

(підпис) Костюченко І.В.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Льовкін В.М.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
114 с., 27 рис., 7 табл., 3 дод., 32 джерела.

ДИСТАНЦІЙНЕ НАВЧАННЯ, ТРИВАЛІСТЬ НАВЧАННЯ,
ПРОГНОЗУВАННЯ ЧАСОВОГО РЯДУ, РЕКУРЕНТНІ НЕЙРОННІ
МЕРЕЖІ, КЕРОВАНІЙ РЕКУРЕНТНИЙ БЛОК.

Об'єкт дослідження – процес дистанційного навчання.

Предмет дослідження – методи визначення тривалості навчання.

Мета роботи – розроблення програмного забезпечення підтримки дистанційного навчання з визначенням тривалості навчання для прийняття рішень про організацію роботи над вивченням дисциплін.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування, мова програмування Python, система MySQL, штучний інтелект, нейронні мережі.

Результати. Розглянуто проблему підтримки дистанційного навчання, виконано проєктування та реалізацію програмного забезпечення підтримки дистанційного навчання на основі розроблення власного методу.

Наукова новизна дипломної роботи полягає в розробленому методі визначення тривалості навчання, що виконує прогнозування фактичної тривалості вивчення дисципліни студентом на основі тривалості вивчення попередніх дисциплін ним та параметрів, що визначають нову дисципліну, на основі використання моделі нейронної мережі GRU, що дозволяє виконувати прийняття рішень про організацію роботи над вивченням дисципліни.

Висновки. Розроблено програмне забезпечення підтримки дистанційного навчання у вигляді вебзастосунку.

Галузь використання – організація дистанційного навчання студентів закладів вищої освіти та навчання онлайн.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 114 pages, 27 figures, 7 tables, 3 appendixes, 32 sources.

DISTANCE LEARNING, TRAINING DURATION, TIME
FORECASTING, RECURRENT NEURAL NETWORKS, GATED
RECURRENT UNIT.

The object of research is the process of distance learning.

The subject of research is methods for determining the duration of training.

The purpose of the work is to develop software to support distance learning with the definition of the duration of training to make decisions about the organization of work on the study of disciplines.

Materials, methods and technical means: object-oriented programming, Python programming language, MySQL system, artificial intelligence, neural networks.

Results. The problem of distance learning support is considered, design and implementation of distance learning support software based on the development of the own method is performed.

The scientific novelty of the thesis lies in the developed method of determining the duration of study, which predicts the actual duration of the discipline by the student based on the duration of the previous disciplines and the parameters that determine the new discipline, based on the GRU neural network model, which allows to make decisions on work on the study of the discipline.

Conclusions. Distance learning support software has been developed as a web application.

Field of use is the organization of distance learning for students of higher education and online learning.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз проблеми підтримки дистанційного навчання	10
1.1 Дистанційне навчання	10
1.2 Програмне забезпечення підтримки процесу дистанційного навчання	13
1.3 Висновки за розділом 1	16
2 Матеріали і методи.....	18
2.1 Задача визначення тривалості навчання.....	18
2.2 Рекурентні нейронні мережі	19
2.3 Розроблений метод визначення тривалості навчання	22
2.4 Висновки за розділом 2	24
3 Проєктування програмного забезпечення	26
3.1 Вибір засобів реалізації програмного забезпечення.....	26
3.2 Функціональні вимоги до програмного забезпечення	28
3.3 Проєктування бази даних	30
3.4 Структура програмного забезпечення	37
3.5 Висновки за розділом 3	38
4 Основні рішення щодо реалізації компонентів системи.....	40
4.1 Алгоритм функціонування системи.....	40
4.2 Опис модулів реалізованої системи	43
4.3 Висновки за розділом 4	45
5 Експлуатація та тестування програми	46
5.1 Призначення програми	46
5.2 Умови виконання програми	46
5.3 Виконання програми.....	46
6 Економіко-організаційна частина.....	54
6.1 Планування розробки програмного продукту.....	54
6.2 Визначення витрат на розробку програми	57

6.2.1 Розрахунок основної заробітної плати	57
6.2.2 Розрахунок додаткової заробітної плати	57
6.2.3 Відрахування на соціальне страхування та інші фонди.....	58
6.2.4 Визначення витрат на матеріали	58
6.2.5 Витрати на спеціальне обладнання	59
6.2.6 Інші прямі витрати	62
6.2.7 Розрахунок накладних витрат.....	63
6.3 Розрахунок економічної ефективності програмного продукту.....	64
7 Охорона праці та безпека в надзвичайних ситуаціях.....	65
7.1 Аналіз потенційних небезпек.....	65
7.2 Заходи забезпечення безпеки.....	66
7.3 Заходи з виробничої санітарії та гігієни праці.....	67
7.4 Заходи з пожежної безпеки	72
7.5 Заходи з забезпечення безпеки у надзвичайних ситуаціях.....	73
Висновки	75
Перелік джерел посилання	77
Додаток А Технічне завдання	81
Додаток Б Текст програми	86
Додаток В Слайди презентації.....	107

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

GRU – gated recurrent unit;

ЕОМ – електронно-обчислювальна машина;

ПК – персональний комп'ютер.

ВСТУП

З настанням пандемії коронавірусу в світі дистанційні процеси охопили різні галузі знань. Серед них однією з найбільш активно охоплених є сфера освіти, оскільки на цей період майже всі процеси були перенесені в дистанційну форму. Відповідно наявність якісного інструменту для організації даного процесу є важливим завданням.

На даний момент ринок систем керування або підтримки процесу навчання є достатньо широким. Існує достатньо багато різних механізмів, частина з яких є безкоштовними, а частина платними.

При цьому важливо і розуміти індивідуальні особливості студентів, і коригувати програму під них для досягнення успішного результату. Адже без ефективного індивідуального підходу досягнення результату ускладниться ще більше ніж в очній формі. Окрім того в залежності від індивідуальних особливостей можна приймати рішення щодо змінювання завдання, змінювання форм, які використовуються, матеріалів.

Тому проблему дистанційного навчання загалом і визначення тривалості навчання у її складі можна вважати актуальними.

Об'єкт дослідження – процес дистанційного навчання. Предмет дослідження – методи визначення тривалості навчання.

Мета роботи – розроблення програмного забезпечення підтримки дистанційного навчання з визначенням тривалості навчання для прийняття рішень про організацію роботи над вивченням дисциплін.

1 АНАЛІЗ ПРОБЛЕМИ ПІДТРИМКИ ДИСТАНЦІЙНОГО НАВЧАННЯ

1.1 Дистанційне навчання

На даний момент окремо розрізняють навчання онлайн та дистанційне навчання.

Дистанційне навчання дозволяє навчатись дистанційно без регулярних очних контактів з викладачем у класі або аудиторії.

Навчання онлайн – це форма навчання, за якої викладачі та студенти використовують освітні засоби, доступні в Інтернеті або мережі: це означає, що студенти також можуть користуватися Інтернет-інструментами, перебуваючи фізично в класі разом зі своїм викладачем та однолітками, таке навчання можна використовувати де завгодно і коли завгодно, тому викладачі можуть запропонувати студентам використовувати такі інструменти як на уроці, так і для підготовки та виконання завдань вдома, інтернет-засоби навчання часто використовуються для створення змішаних навчальних середовищ у класі, це допомагає тримати учнів залученими до занять та до матеріалу, дистанційне навчання також допомагає викладачам економити час на підготовку перед заняттям, за допомогою освітніх інструментів в Інтернеті викладачі можуть витрачати більше часу на оцінювання робіт, приділяючи студентам особисту увагу і, можливо, отримуючи трохи вільного часу для робочого графіка [1].

Різниця між навчанням онлайн та дистанційним навчанням полягає в наступному:

- місцезнаходження: навчання онлайн може включати використання Інтернет-інструментів та платформ, перебуваючи в звичайному класі, однак дистанційне навчання є дистанційним і не передбачає жодної взаємодії між учнем та викладачем;

- взаємодія: навчання онлайн може включати взаємодію з викладачами та однолітками, тоді як дистанційне навчання не має особистої взаємодії;

– намір: навчання онлайн може бути використано як доповнення для викладачів на їх курсах, тоді як дистанційне навчання замінює вчителів інструкціями, попередньо встановленими на навчальній платформі [1]-[2].

Розрізняють наступні види дистанційного навчання:

– відеоконференції – це звичний спосіб безпосередньої взаємодії викладачів зі студентами на уроках у прямому ефірі, це може бути індивідуальний сеанс або подібний до класу сценарій, коли кілька учнів підключаються до викладача в прямому ефірі;

– синхронне навчання – це варіант дистанційного навчання, коли всі студенти навчаються разом одночасно (і часто навіть у одному місці), але викладач знаходиться в іншому місці, у ньому часто є відео- або телеконференції, які цифровим способом з'єднують викладачів та учнів;

– асинхронне навчання – це менш зв'язаний, але також менш обмежений формат: замість живих уроків онлайн студентам дають навчальні завдання із зазначенням термінів, потім вони самостійно навчаються, щоб виконати завдання;

– курси онлайн з відкритим графіком додають ще один рівень гнучкості – це тип асинхронного налаштування курсу, за винятком того, що також немає ніяких термінів, це ідеально підходить для учнів, які мають інші вимоги до свого часу, таких як професіонали чи батьки, які не сидять вдома;

– інтернет-курси з фіксованим робочим часом – це тип синхронного курсу, який вимагає від користувачів Інтернету відвідувати певне віртуальне місце у встановлений час (наприклад, вебінар), на відміну від більш жорстких синхронних уроків це дозволяє студентам з будь-якої точки світу підключатися та взаємодіяти в Інтернеті;

– комп'ютерна дистанційна освіта – це синхронний урок на комп'ютерах, як правило, в комп'ютерній лабораторії, це найчастіше зустрічається в існуючих установах, які вже мають доступ до необхідних пристроїв;

– гібридне навчання – це специфічний тип змішаного навчання, коли студенти вивчають один і той же урок у режимі реального часу (тобто синхронне дистанційне навчання), але частина студентів присутня фізично, тоді як інші навчаються дистанційно [3].

До переваг дистанційного навчання належать наступні:

- студенти можуть зручно працювати з матеріалами за курсом, що по суті доставляються їм додому;
- студенти можуть надбати корисні навички, такі як планування та дослідження;
- студенти можуть легко залишити свої відгуки;
- немає витрат часу на переїзд;
- доступ до студентів без можливості особистого навчання;
- дистанційне навчання забезпечує своєчасне навчання;
- дистанційне навчання може охопити ширшу аудиторію;
- дистанційне навчання може призвести до більшої взаємодії;
- дистанційне навчання може зрівняти доступ до освіти;
- дистанційне навчання робить інформацію та конспекти лекцій відкритими для всіх;
- дистанційне навчання часто підвищує ефективність навчання за рахунок швидшого досягнення кращих результатів [4].

Окрім того важливо, що за деяких обставин в іншій формі процес навчання не може реалізовуватися. Тоді така форма стає єдиним прийнятним варіантом, а для її роботи обов'язково потрібне програмне забезпечення, адже інакше неможливо організувати дистанційне навчання.

Огляд існуючих програмних рішень для підтримки процесу дистанційного навчання виконано у наступному підрозділі.

1.2 Програмне забезпечення підтримки процесу дистанційного навчання

Одним з найвідоміших сервісів керування навчанням на даний момент є система Google Classroom.

Система Google Classroom (рис. 1.1) дозволяє виконувати наступні дії викладачу:

- завантажувати матеріали за курсом;
- визначати завдання в межах курсу;
- створювати теми та категорії в межах курсу;
- додавати тестове завдання;
- визначати модулі або блоки;
- визначати час, до якого має бути виконано завдання;
- переглядати студентів, які підписалися на курс;
- переглядати перелік робіт, які було отримано в розрізі кожного завдання;
- створювати оголошення;
- переглядати роботи, які надіслав студент;
- отримувати сповіщення про отримані роботи;
- оцінювати надіслані студентами роботи;
- виконувати рецензування роботи шляхом визначення коментарів до роботи;
- інтегрувати сервіс з іншими сервісами від компанії Google.

Система Google Classroom дозволяє виконувати наступні дії студенту:

- підписуватися на проходження курсу;
- переглядати матеріали і завдання за курсом;
- додати роботу за завданням для перевірки викладачем;
- переглядати зауваження викладача;
- переглядати власні оцінки за курс;
- створювати коментарі до курсу.

Все це робить загалом її однією з найпотужніших систем керування навчанням.

При цьому до недоліків системи Google Classroom можна віднести наступне:

- система не має засобів автоматизації тестування;
- система повністю належить компанії Google, відповідно пов'язана з залежністю від зовнішнього ресурсу, тобто якщо компанія Google вирішить позбавитися такого інструменту, то університет може загубити напрацьований засіб, відповідно встановити програмне забезпечення на власному сервері і підтримувати його неможливо.

Рисунок 1.1 – Створення завдання в системі Google Classroom

Альтернативою даному сервісу є широко відома навчальна платформа Moodle, яка до того ж є безкоштовною. Дана платформа має багато засобів для організації навчання, які можна розширювати за допомогою додаткових модулів.

На відміну від попередньої альтернативи дане програмне забезпечення встановлюється на власний сервер, відповідно університет або інша організація, що організовує навчання, може контролювати повністю роботу.

Платформа для навчання Moodle зокрема дозволяє реалізовувати наступні функції:

- переглядати дисципліни, на які підписаний студент;
- навчатися за матеріалами дисципліни;
- проходити тестування;
- завантажувати матеріали за дисципліною;
- комунікувати з іншими користувачами;
- отримувати результати навчання;
- створювати власні дисципліни з широкими можливостями, включаючи завантаження будь-яких матеріалів, проведення тестування; визначення тем за дисципліною.

При цьому переваги Moodle є і недоліками системи, адже надто велика кількість функцій призводить до складності використання програмного забезпечення, окрім того для деяких викладачів може бути ускладнене розуміння загалом виконання деяких дій. Окрім того Moodle не розділяє користувачів, відповідно один користувач може виступати десь викладачем, а десь – студентом.

Налаштування системи на використання вимагає довгої підготовки. У протилежному випадку функціональність використовуваної системи буде надто обмеженою.

Усі ці фактори говорять про те, що важливим практичним завданням є розроблення програмного забезпечення підтримки процесу дистанційного навчання, що відповідає наступним критеріям:

- може бути встановлене на серверах організації;
- є простим у використанні і налаштуванні;
- підтримує можливість визначення тривалості навчання.

Загалом будь-яка система підтримки навчання повинна виконувати наступні функції:

- управління користувачами, дисциплінами, ролями та створення звітів: це допомагає завантажувати курси, розподіляти ролі та створювати різні звіти;
- повідомлення;
- оцінювання на основі проведення тестування;
- відображення отриманих балів: це допомагає вести облік навчальних курсів, результатів діяльності людини та надавати сертифікати тим, хто успішно пройшов навчання;
- управління курсами під керівництвом викладача;
- адміністрування [5].

Саме на досягнення описаних вище можливостей та виконання зазначених вимог до програми і буде направлена розробка, що буде вестися в роботі.

1.3 Висновки за розділом 1

Виконано аналіз проблеми підтримки дистанційного навчання. Розглянуто форми виконання дистанційного навчання, переваги даного процесу.

Виконано огляд програмного забезпечення, що може використовуватися для підтримки процесу навчання. Розглянуто системи Google Classroom і Moodle. Виділено їх особливості і недоліки. Сформульовано вимоги до програмної системи, що розробляється.

Завдання дипломної кваліфікаційної роботи магістра полягають в наступному:

- дослідження методів визначення тривалості навчання;
- розробка технічного завдання;
- вибір засобів розробки;

- проєктування структури програмного забезпечення;
- розробка програмного забезпечення підтримки дистанційного навчання;
- тестування та відлагодження програми;
- економічне обґрунтування доцільності розробки.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Задача визначення тривалості навчання

Важливість задачі визначення тривалості навчання полягає в тому, що викладач, визначаючи тривалість вивчення дисципліни дуже часто покладається на своє представлення і власний досвід, не маючи при цьому адекватних механізмів зокрема моделювання цих показників. Зокрема можна порівняти, що в сфері керування проєктами існує багато різноманітних в тому числі практичних засобів, які дозволяють як відслідковувати виконання проєкту на різних етапах, так і моделювати тривалість вивчення дисципліни, зокрема на основі порівняння з аналогами. Тож дану логіку можна перенести на сферу освіти і надати таким чином інструмент для визначення тривалості навчання, зокрема і прогнозування такої тривалості.

Задача визначення тривалості навчання полягає в прогнозуванні тривалості вивчення конкретним студентом конкретної дисципліни. Для прогнозування має бути оцінено попередній досвід, тобто визначено тривалість навчання студента за дисципліною.

Для прогнозування тривалості навчання потрібно з одного боку оцінити минулий досвід вивчення студентом інших дисциплін, а із іншого боку потрібно оцінити саму дисципліну. Тобто саме комплексний вплив індивідуальних і загальних факторів визначає в результаті тривалість навчання.

Задачу визначення тривалості навчання можна визначити як задачу визначення функціональної залежності (2.1):

$$T_i^s = g(C_1^i, C_2^i, \dots, C_k^i, T_{i-1}^s, T_{i-2}^s, \dots, T_{i-e}^s), \quad (2.1)$$

де T_i^s – тривалість вивчення студентом s дисципліни з номером i , при чому номер дисципліни визначає порядковий номер вивчення даної дисципліни для студента;

$C_1^i, C_2^i, \dots, C_k^i$ – послідовність значень параметрів, що характеризують дисципліну i ;

$T_{i-1}^s, T_{i-2}^s, \dots, T_{i-e}^s$ – тривалість вивчення студентом s дисциплін з номерами $i-1, i-2, \dots, i-e$.

Тобто для обчислення тривалості навчання в майбутній момент потрібно використати значення тривалості навчання за попередені e кроків.

У процесі розроблення власного методу визначення тривалості навчання потрібно виконати наступні завдання:

- визначити інструмент, який буде використовуватися для моделювання функціональної залежності вигляду (2.1);
- зафіксувати спосіб обчислення тривалості навчання;
- визначити набір параметрів, які характеризують дисципліну.

2.2 Рекурентні нейронні мережі

Застосування інструментів прогнозування полягає в застосуванні одночасно методів, які визначають спосіб виконання прогнозного моделювання, та машинного навчання для аналізу минулих даних з метою прогнозування майбутніх тенденцій, при цьому нейронні мережі відрізняються від звичайних інструментів прогнозування, найбільш часто використовуваною серед яких є модель на основі лінійної регресії, найголовніше завдяки тому, що нейронні мережі працюють краще в прогнозуванні через наявність прихованих шарів, моделі лінійної регресії використовують лише вхідні та вихідні вузли для прогнозування, а нейромережа також використовує прихований шар, щоб зробити прогнози більш точними, серед недоліків нейронних мереж можна відзначити наступне:

- вони вимагають величезних обчислювальних потужностей, тому вони є не вигідними;

– нейронні мережі найкраще працюють, коли навчаються з великими наборами даних [6]-[7].

Враховуючи, що сфера, для якої виконується розробка, – освіта, є досить потужною, університети мають у своєму використанні достатньо потужні сервери, то створення такої системи в межах університету з використанням нейронних мереж є достатньо придатним рішенням.

Окрім традиційних багатошарових перцептронів, які можна використовувати для прогнозування, існують також рекурентні нейронні мережі, які направлені на прогнозування послідовностей.

Оскільки в задачі (2.1) використовується послідовність даних, то рекурентні нейронні мережі можуть бути вдалим механізмом для прогнозування в межах даної задачі.

Принцип роботи рекурентного нейрону загалом наступний:

- один часовий крок вводу подається в мережу;
- обчислюється поточний стан, використовуючи комбінацію поточного входу та попереднього стану;
- поточний стан стає попереднім станом для наступного часового кроку;
- можна використати стільки часових кроків, скільки вимагає проблема, та об'єднати інформацію з усіх попередніх станів;
- після завершення всіх часових кроків для розрахунку виходу використовується кінцевий поточний стан;
- результат порівнюється з фактичним результатом і генерується помилка;
- потім помилка повторно розповсюджується на мережу для оновлення вагових коефіцієнтів, і мережа проходить навчання [8]-[9].

Тобто рекурентні нейронні мережі мають наступні відмінності відносно багатошарових перцептронів: використовують в якості входу для кожного наступного нейрону не тільки поточне вхідне значення з

послідовності даних, але і значення, отримані під час попередніх розрахунків.

Серед рекурентних нейронних мереж можна виділити наступні архітектури:

- long short term memory;
- gated recurrent units (GRU).

Дані дві архітектури відрізняються тим, що у варіанті GRU представлено два шлюзи, а в першому їх одразу три. Тобто принципи організації роботи у них достатньо подібні, але GRU є по суті дещо спрощеною архітектурою порівняно з long short term memory, яка при цьому зарекомендувала себе на практиці. Це означає, що для багатьох практичних задач достатньо такої архітектури.

Першою була створена саме long short term memory, а потім як її спрощення – GRU [9]. Ці дві архітектури мають наступні переваги порівняно з простими рекурентними нейронними мережами:

- усувають проблеми з навчанням: для простих рекурентних нейронних мереж відома проблема загасаючого градієнта, на розв’язання якої як раз і були направлені long short term memory;

- прості рекурентні нейронні мережі не здатні вивчити довгострокові залежності: у них є короткочасна пам’ять, але вони не здатні пам’ятати те, що трапилося довгий час тому і що може мати відношення до того, що відбувається зараз.

Тому ґрунтуючись на цих принципах, а також враховуючи, що рекурентний шар буде в результуючій мережі не єдиним шаром, адже окрім послідовності значень є також в задачі (2.1) і набір параметрів, що всі разом представляють дисципліну, то для спрощення даної частини в межах єдиного розв’язання було віддано перевагу архітектурі GRU для реалізації розв’язання даної проблеми.

Принцип організації обчислень всередині однієї чарунки GRU представлено на рис. 2.1 [10].

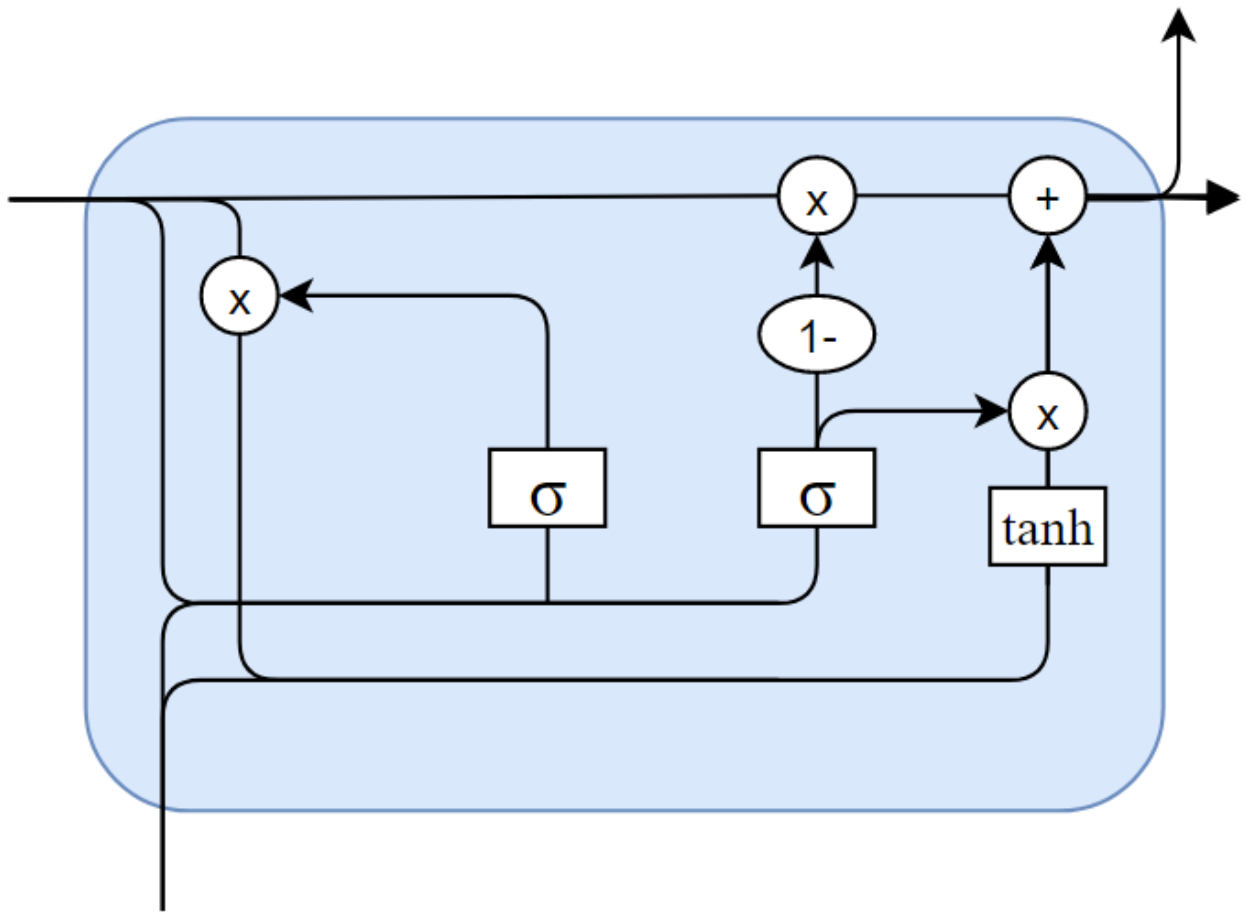


Рисунок 2.1 – Архітектура GRU

2.3 Розроблений метод визначення тривалості навчання

Основною складністю задачі, що розглядається, є те, що в ній окрім безпосередньо послідовності, є ще додаткові значення, а тому має бути побудована архітектура нейронної мережі, що враховує цей фактор.

Далі буде описано розроблений метод визначення тривалості навчання, що складається з набору етапів.

На першому етапі має бути сформовано дані про результати навчання студентів. Для цього для кожного студента має бути визначено тривалість його навчання за кожною дисципліною. При цьому має враховуватися саме фактичний часовий проміжок. Таким проміжком можна вважати проміжок часу, що починається з моменту першого перегляду будь-яких даних дисципліни в програмі студентом, а завершаючим моментом є момент

отримання оцінки за останню здану роботу, якщо обчислення оцінки відбувається автоматично, або момент виставлення підсумкової оцінки у випадку, коли для цього проводиться певна співбесіда зі студентом.

На другому етапі має бути сформовано масив даних про всі дисципліни, що характеризуються наступними параметри:

- кількість лабораторних та практичних робіт;
- кількість самостійних робіт;
- кількість тестів для перевірки знань студентів;
- кількість матеріалів за дисципліною;
- форма контролю знань;
- кількість тем, що розглядаються в дисципліні;
- середня тривалість навчання за дисципліною.

Останній показник розраховується на основі аналізу всіх вивчень дисципліни різними студентами та усереднення отриманого значення.

Це загальні параметри, які впливають на вивчення дисципліни та отримання підсумкової оцінки будь-яким студентом.

На третьому етапі відбувається формування вибірки даних. Для цього має бути впорядковано всі вивчені кожним студентом дисципліни за принципом, що спочатку розташовані всі дисципліни першого студента за порядком їх вивчення, далі другого і т.д.

Для кожного студента, починаючи з дев'ятої дисципліни відбувається формування окремого екземпляру вибірки даних за такою дисципліною. До екземпляру входять параметри цієї дисципліни та послідовність часу вивчення попередніх восьми дисциплін. Тобто для забезпечення прогнозування студент має навчатися хоча би семестр, тоді для нього буде сформовано набір даних, достатній для виконання прогнозування.

Кожен екземпляр вибірки даних представляє собою набір значень у вигляді множини (2.2):

$$< C_l^i, C_s^i, C_t^i, C_m^i, C_c^i, C_{tn}^i, C_{lt}^i, T_{i-1}^s, T_{i-2}^s, \dots, T_{i-8}^s >, \quad (2.2)$$

де C_l^i – кількість лабораторних та практичних робіт;

C_s^i – кількість самостійних робіт;

C_t^i – кількість тестів для перевірки знань студентів;

C_m^i – кількість матеріалів за дисципліною;

C_c^i – форма контролю знань;

C_{tn}^i – кількість тем, що розглядаються в дисципліні;

C_{lt}^i – середній час вивчення дисципліни студентом.

На четвертому етапі виконується створення моделі прогнозування тривалості навчання.

Дана модель повинна складатися з шарів чарунок GRU, які обробляють послідовність $T_{i-1}^s, T_{i-2}^s, \dots, T_{i-8}^s$, та окремого повнозв'язного шару, що в якості вхідних даних отримує характеристики дисципліни $C_l^i, C_s^i, C_t^i, C_m^i, C_c^i, C_{tn}^i$.

Запропоновано в процесі практичного вивчення використовувати 2 шари GRU і окремий повнозв'язний шар, а далі з'єднувати дані цих шарів в єдиний за допомогою додаткового повнозв'язного шару. На виході отримується значення тривалості для даного випадку вивчення дисципліни даним студентом.

На п'ятому етапі забезпечується навчання створеної описаним чином нейронної мережі.

На шостому етапі виконується прогнозування тривалості навчання студентом за дисципліною для заданого набору значень параметрів (2.2).

2.4 Висновки за розділом 2

Виконано постановку задачі визначення тривалості навчання. Обґрунтовано вибір рекурентних нейронних мереж для розв'язання задачі. Розглянуто архітектури рекурентних нейронних мереж, в результаті обрано

архітектуру GRU. Розроблено власний метод визначення тривалості навчання.

Наукова новизна дипломної роботи полягає в розробленому методі визначення тривалості навчання, що виконує прогнозування фактичної тривалості вивчення дисципліни студентом на основі тривалості вивчення попередніх дисциплін ним та параметрів, що визначають нову дисципліну, на основі використання моделі нейронної мережі GRU, що дозволяє виконувати прийняття рішень про організацію роботи над вивченням дисципліни.

3 ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вибір засобів реалізації програмного забезпечення

Важливим фактором для реалізації проєкту є вибір мови програмування, оскільки від цього залежить те, чи вдасться виконати розробку в межах дипломної кваліфікаційної роботи магістра, і рівень виконаної роботи.

У процесі порівняння мов Python і Java для реалізації програмного забезпечення було виділено наступне:

- Python використовує динамічну типізацію, відповідно під час написання коду не потрібно визначати типи змінних, оскільки інтерпретатор робить висновок про ці типи, а перевірки будуть виконуватися під час виконання, це призводить до спрощення синтаксису, який є цілком схожим на англійську мову, більше того, Python не використовує фігурні дужки та дотримується правил відступу, що робить код досить простим для читання та зручним, у той же час Java дотримується суворих правил синтаксису, це статично типізована мова, де потрібно чітко оголосити типи змінних;

- Python – це інтерпретована мова, тоді як Java – компільована: інтерпретовані мови перетворюють код у машиночитаний код на ходу, оскільки програма виконує команди, що полегшує перегляд або налагодження, а компільовані мови повинні перевести вихідний код у машинний код перед запуском, що ускладнює перегляд або налагодження коду, але при цьому пришвидшує час виконання програми в результаті;

- оскільки Python є синтаксично дуже легкою, але при цьому повноцінною мовою програмування загального призначення, він став популярним варіантом для людей з різних дисциплін, які хотіли експериментувати з машинним навчанням та застосувати штучний інтелект у відповідній галузі, тому багато розробок у галузі штучного інтелекту та машинного навчання здійснюється за допомогою Python з величезною екосистемою та бібліотеками (існують бібліотеки TensorFlow, Keras, Scikit-

Learn та PyTorch) на даний момент це найпопулярніша мова у цій галузі, Java також вважається хорошим варіантом, коли мова йде про машинне навчання, її можна налагоджувати та використовувати під застосунки різного рівня, хоча підтримка цих рішень відрізняється порівняно з Python в гірший бік, відповідно важкий процес входження в галузь, окрім того такі програми вимагають написання більшої кількості коду, що призводить до уповільнення процесу розробки [11]-[13].

Результати виконаного порівняння мов програмування представлено в табл. 3.1.

Таблиця 3.1 – Порівняння мов програмування

Критерій	Python	Java
Головна парадигма	Об'єктно-орієтована	Об'єктно-орієтована
Специфікація	Загального призначення	Загального призначення
Принцип виконання	Інтерпретований	Компільований
Вид типізації	Динамічний	Статичний
Засоби штучного інтелекту	Потужні, гнучкі, велика підтримка	Наявні, менша гнучкість
Процес написання коду	Простий	Складний

Остаточний вибір було виконано на користь мови Python, оскільки вона дозволяє виконувати розробку проєктів даного рівня швидше і надає більш підтримувані бібліотеки для організації штучного інтелекту, зокрема архітектура нейронних мереж GRU може бути застосована з готових рішень для організації шарів замість того, щоб створювати такі шари вручну.

3.2 Функціональні вимоги до програмного забезпечення

Програмне забезпечення підтримки дистанційного навчання має бути організовано у відповідності з результатами огляду програмних аналогів, виконаного в першому розділі, з концентрацією на концепції визначення тривалості навчання.

Для цього було визначено функціональні вимоги до програмного забезпечення підтримки дистанційного навчання, що має виконувати наступні функції:

- створення дисципліни для викладання;
- визначення тем за дисципліною;
- визначення набору робіт за дисципліною;
- внесення матеріалів для викладання дисципліни;
- підписування студентів на проходження дисциплін;
- створення тестів за дисципліною;
- проходження тестів за дисципліною;
- обчислення результату проходження тесту;
- перегляд матеріалів за дисципліною;
- перегляд студентів, підписаних на дисципліну;
- перегляд дисциплін, на які підписався студент;
- прогнозування тривалості навчання за дисципліною;
- завантаження виконаних робіт;
- створення коментарів щодо роботи;
- виставлення оцінки за роботу;
- створення облікових записів студентів та викладачів;
- створення груп студентів;
- додавання викладачів для роботи над дисципліною.

Для забезпечення реалізації даного набору функціональних вимог було визначено такі групи користувачів:

- викладачі, які забезпечують керування самим процесом навчання;

- студенти, які безпосередньо беруть участь в процесі навчання;
- адміністратори, які забезпечують внесення даних найвищого рівня.

Сценарії роботи з програмою виділено у вигляді діаграми прецедентів (рис. 3.1).

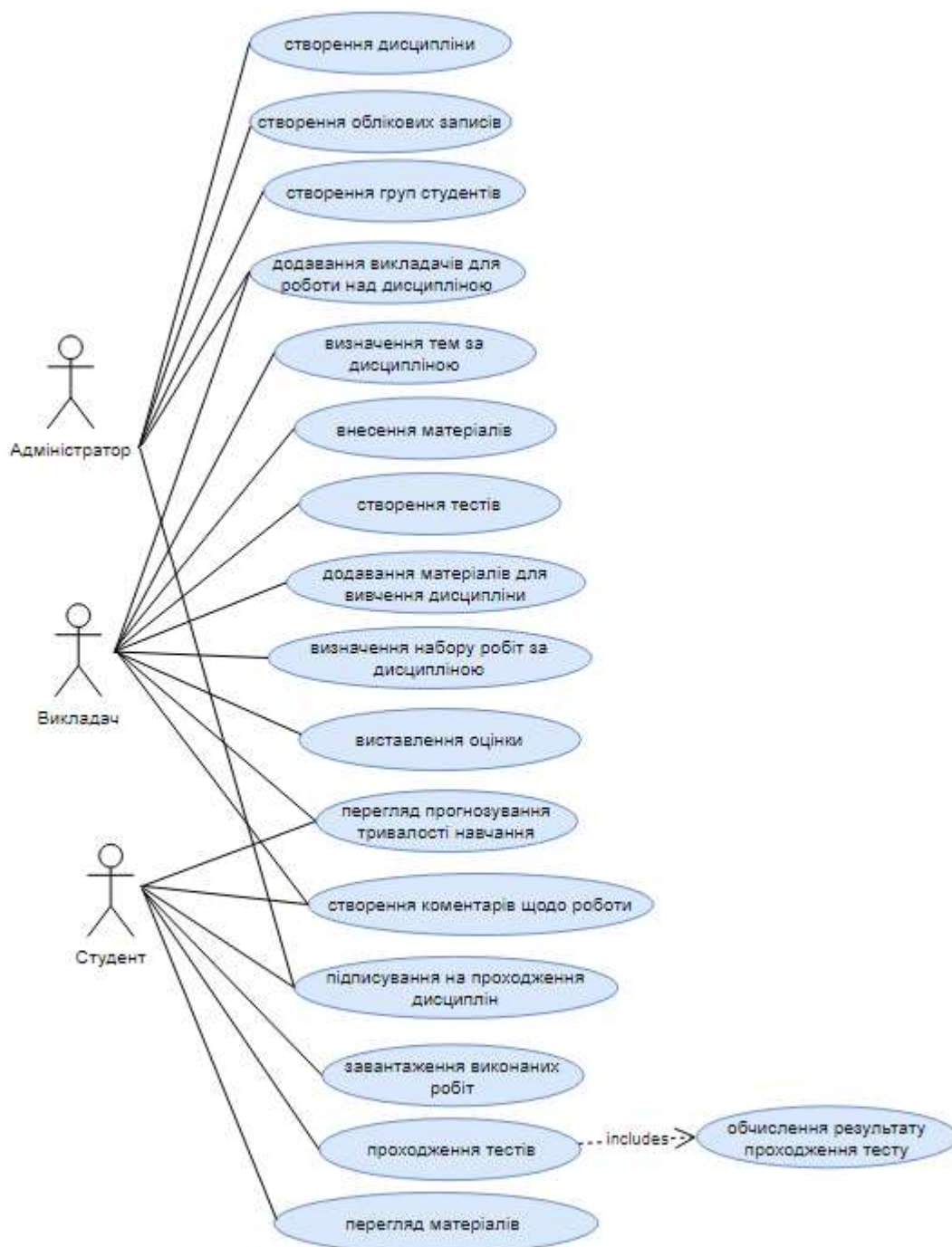


Рисунок 3.1 – Діаграма прецедентів

3.3 Проєктування бази даних

Для проєктування бази даних спочатку було обрано в якості системи керування базами даних MySQL.

Основні сутності і відповідні таблиці, які було виділено в результаті проєктування бази даних:

- Student – студент;
- Group – група;
- Course – дисципліна;
- Teacher – викладач;
- User – користувач;
- Control – контроль знань студента;
- Materials – матеріали за дисципліною;
- Question – запитання;
- Answer – відповіді;
- QuestionType – тип запитання;
- Topic – тема, яку вивчають за дисципліною;
- Messages – коментарі викладача щодо робіт студента (комунікація за роботою).

До зв'язуючих таблиць належать:

- StudentCourse – вивчення студентом дисципліни (зв'язує таблиці Student і Course);
- CourseTeacher – викладання дисциплін викладачами (зв'язує таблиці Teacher і Course);
- StudentAnswer – відповідь студента;
- CourseWorks – роботи, які потрібно виконати за дисципліною;
- StudentWorks – роботи, виконані студентом за дисципліною;
- StudentControl – результати тестування студента.

Таблиця Student включає до свого складу такі характеристики студента:

- stid – номер студента;
- stsurname – прізвище студента;
- stname – ім'я студента;
- stpatr – по батькові студента;
- stbirth – дата народження студента;
- stuser – обліковий запис студента (зовнішній ключ з таблиці User);
- stgroup – група студента (зовнішній ключ з таблиці Group);
- stvar – варіант роботи студента з системою (може особисто обирати дисципліну чи ні).

Таблиця Group включає до свого складу такі характеристики групи:

- grid – номер групи;
- grname – назва групи (формальний номер, який використовується для нумерування групи в університеті);
- gryear – рік початку навчання групи.

Таблиця Course включає до свого складу такі характеристики дисципліни:

- crid – номер дисципліни;
- crname – назва дисципліни;
- crterm – семестр проходження дисципліни;
- crform – форма контролю вивчення дисципліни;
- crvar – варіант підписки на дисципліну (адміністратором або з можливістю особистого вибору студентом).

Таблиця Teacher включає до свого складу такі характеристики викладача:

- tid – номер викладача;
- tsurname – прізвище викладача;
- tname – ім'я викладача;
- tpatr – по батькові викладача;
- tdepart – відділення, на якому працює викладач (кафедра);
- tscideg – науковий ступінь викладача;

- tacstat – вчене звання викладача;
- tposition – посада викладача;
- tuser – обліковий запис викладача (зовнішній ключ з таблиці User).

Таблиця User включає до свого складу такі характеристики облікового запису:

- usid – номер;
- uslogin – логін;
- uspass – пароль.

Таблиця StudentCourse включає до свого складу такі характеристики навчання студента за курсом:

- scid – номер;
- scstud – студент, який вивчає дисципліну (зовнішній ключ з таблиці Student);
- sccourse – дисципліна, яку вивчає студент (зовнішній ключ з таблиці Course);
- scscore – оцінка за вивчення дисципліни;
- scbdate – час початку роботи над дисципліною;
- scedate – час завершення роботи над дисципліною;
- scptime – прогнозований час вивчення дисципліни.

Таблиця CourseTeacher включає до свого складу такі характеристики викладання дисципліни викладачем:

- ctid – номер;
- ctcourse – дисципліна (зовнішній ключ з таблиці Course);
- ctteacher – викладач (зовнішній ключ з таблиці Teacher);
- ctposition – роль (лектор, асистент).

Таблиця Topic включає до свого складу такі характеристики теми дисципліни:

- toid – номер;
- toname – тема;
- toperiod – дата завершення вивчення теми;

– tocourse – дисципліна, за якою вивчається дана тема (зовнішній ключ з таблиці Course).

Таблиця Control включає до свого складу такі характеристики тестів з дисципліни:

– coid – номер;
– cotopic – тема, за якою проводиться тест (зовнішній ключ з таблиці Topic);

- coname – назва тесту;
- coform – форма тесту;
- cobdate – дата початку проведення тесту;
- coedate – дата завершення проведення тесту;
- coduration – тривалість виконання тесту.

Таблиця Materials включає до свого складу такі характеристики матеріалів з дисципліни:

– mid – номер;
– mtopic – тема, до вивчення якої належить матеріал (зовнішній ключ з таблиці Topic);

- mname – назва матеріалу;
- msite – зовнішнє посилання на матеріал;
- mfile – файл матеріалу;
- mdesc – опис матеріалу.

Таблиця Question включає до свого складу такі характеристики запитання тесту:

– qid – номер;
– qtext – текст запитання;
– qcontrol – тест, до якого належить запитання (зовнішній ключ з таблиці Control);
– qresult – оцінка за правильну відповідь;
– qtype – тип запитання.

Таблиця Answer включає до свого складу такі характеристики відповідей на запитання тесту:

- aid – номер;
- atext – текст відповіді;
- apos – порядковий номер відповіді;
- aresult – оцінка за правильну відповідь;
- aquestion – запитання, на яке зазначено відповідь (зовнішній ключ з таблиці Question).

Таблиця StudentAnswer включає до свого складу такі характеристики відповідей студента на запитання тесту:

- said – номер;
- saquestion – запитання, на яке дано відповідь (зовнішній ключ з таблиці Question);
- sastud – студент, який дав відповідь (зовнішній ключ з таблиці Student);
- saname – відповідь, яку обрав студент;
- sascore – оцінка за відповідь.

Таблиця CourseWorks включає до свого складу такі характеристики робіт за дисципліною:

- cwid – номер;
- cwsite – зовнішнє посилання на матеріали роботи;
- cwname – назва роботи;
- cwdesc – опис роботи;
- cwtopic – тема роботи (зовнішній ключ з таблиці Topic);
- cwmater – матеріал до роботи (зовнішній ключ з таблиці Materials);
- cwmaxscore – максимальна оцінка за виконання роботи;
- cwdeadline – останній строк здачі роботи.

Таблиця StudentWorks включає до свого складу такі характеристики робіт студента:

- swid – номер;

- swfile – файл студентської роботи;
- swdesc – опис до роботи від студента;
- swscore – оцінка за роботу;
- swcourw – робота з дисципліною (зовнішній ключ з таблиці CourseWorks);
- swstud – студент, який виконував роботу (зовнішній ключ з таблиці Student).

Таблиця StudentControl включає до свого складу такі характеристики результату проходження тесту студентом:

- scoid – номер;
- scocontrol – тест (зовнішній ключ з таблиці Control);
- scostudent – студент, який пройшов тест (зовнішній ключ з таблиці Student);
- scoscore – оцінка;
- scodate – дата проходження тесту.

Таблиця Messages включає до свого складу такі характеристики повідомлень:

- meid – номер;
- memessid – повідомлення, відповіддю на яке є дане повідомлення (номер meid, може бути порожнім);
- mestudwid – студентська робота, реакцією на яку є відповідь (зовнішній ключ з таблиці StudentWorks);
- mecoursteach – викладач за дисципліною, який дав відповідь (зовнішній ключ з таблиці CourseTeacher);
- metext – текст відповіді.

Дана таблиця направлена на підтримку з одного боку взаємодії за дисципліною між викладачем та студентом, а з іншого боку – на організацію звітування за результатами.

Описану логіку структурування даних було представлено у вигляді схеми бази даних (рис. 3.2).

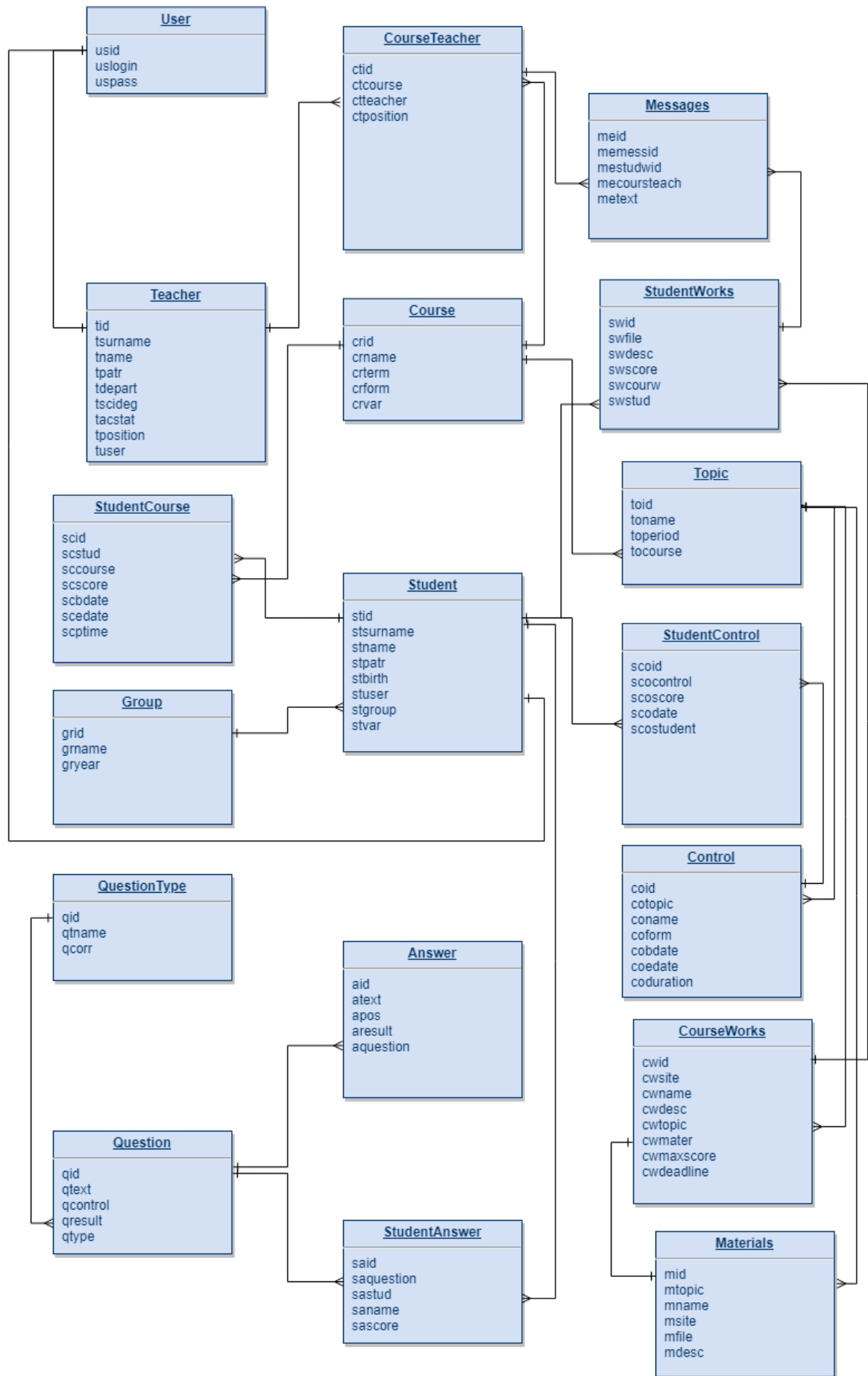


Рисунок 3.2 – Схема бази даних

Таблиця QuestionType включає до свого складу такі характеристики категорій запитань тесту:

- qid – номер;
- qtname – тип запитання;
- qscore – кількість правильних відповідей.

3.4 Структура програмного забезпечення

Структуру програмної системи (рис. 3.3) було обрано на основі визначених функціональних вимог та необхідності програмної реалізації розробленого методу.

У відповідності з функціональними вимогами до програми усі функції розбиваються на 3 групи, які між собою майже не перетинаються.

Відносно програмної реалізації єдине, що їх функціонально об'єднує, – це основа, що полягає у взаємодії з базою даних. Саме цю підсистему було винесено окремо.

Для кожної групи функціональних характеристики відповідно було створено окрему підсистему:

- підсистему навчання студента;
- підсистему роботи викладача для керування процесом навчання;
- підсистему роботи адміністратора для керування роботою всіх груп користувачів.

Окремо було винесено підсистему визначення тривалості навчання для реалізації засобів розробленого методу.

Три підсистеми, виділені для окремих ролей користувачів, повинні мати графічний інтерфейс користувача і програмно взаємодіяти з іншими двома підсистемами. Виклик підсистеми визначення тривалості навчання має відбуватися в той момент, коли студент підписується на вивчення дисципліни або адміністратор підписує його на вивчення дисципліни. В цей

момент повинно виконуватися звернення до моделі і відбуватися прогнозування для заданих вхідних даних.



Рисунок 3.3 – Структурна схема програмної системи

3.5 Висновки за розділом 3

Обрано для реалізації програми мову Python, оскільки вона дозволяє виконувати розробку проєктів даного рівня швидше і надає більш

підтримувані бібліотеки для організації штучного інтелекту.

Визначено функціональні вимоги до програми та побудовано відповідну діаграму прецедентів.

Створено структурну схему бази даних з визначенням основних сутностей предметної області в реляційній формі і зв'язків між ними та забезпечено відповідне створення такої бази даних на основі використання системи MySQL.

Побудовано структуру програмної системи на основі розподілу функціональних обов'язків різних груп користувачів, виділення підсистеми реалізації розробленого методу та підсистеми взаємодії з базою даних.

4 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

4.1 Алгоритм функціонування системи

На рис. 4.1 представлено схему функціонування програми для викладача.



Рисунок 4.1 – Схема функціонування програми для викладача

Взаємодія викладача з програмою ґрунтується на початковому виборі дисципліни. Викладач може працювати тільки з тими дисциплінами, які для нього задав адміністратор або інші викладачі. Тобто якщо адміністратор визначив автора дисципліни, то далі даний автор зможе працювати з дисципліною і в тому числі додавати інших викладачів для викладання дисципліни у визначених ним ролях.

Після виконання вибору дисципліни, далі відбувається підтримка роботи з обраною дисципліною. В межах такої підтримки є декілька режимів функціонування:

- керування темами, звідки можна не тільки створювати теми і переглядати створені раніше, але і визначати за темою матеріали, виконувати роботи та тести, які мають пройти студенти;
- керування викладачами, звідки можна переглянути склад викладачів, що працюють над дисципліною, а також додати іншого викладача, визначаючи для нього роль у викладанні;
- перевірка поданих студентами робіт;
- підведення результатів студентів за дисципліною, звідки доступні всі виставлені оцінки і можна підводити підсумки навчання.

На рис. 4.2 представлено схему функціонування програми для студента. Початковий вибір студента полягає у виборі режиму роботи зі своїми власними дисциплінами або підписці на іншу дисципліну. Для підписки студенту доступні тільки ті дисципліни, на які дозволене вільне підписування. Окрім того студенту також має бути встановлений доступ на вільне підписування. Не всім студентам такий доступ може бути оформлено.

Після вибору дисципліни студент працює з програмою в одному з наступних режимів:

- перегляд тем дисципліни, звідки отримує доступ до матеріалів;
- проходження тестів (не всі тести можуть бути доступними одночасно);
- виконання робіт за дисципліною;

– перегляду власних результатів навчання, звідки доступна можливість і переглянути прогнозовану тривалість навчання.

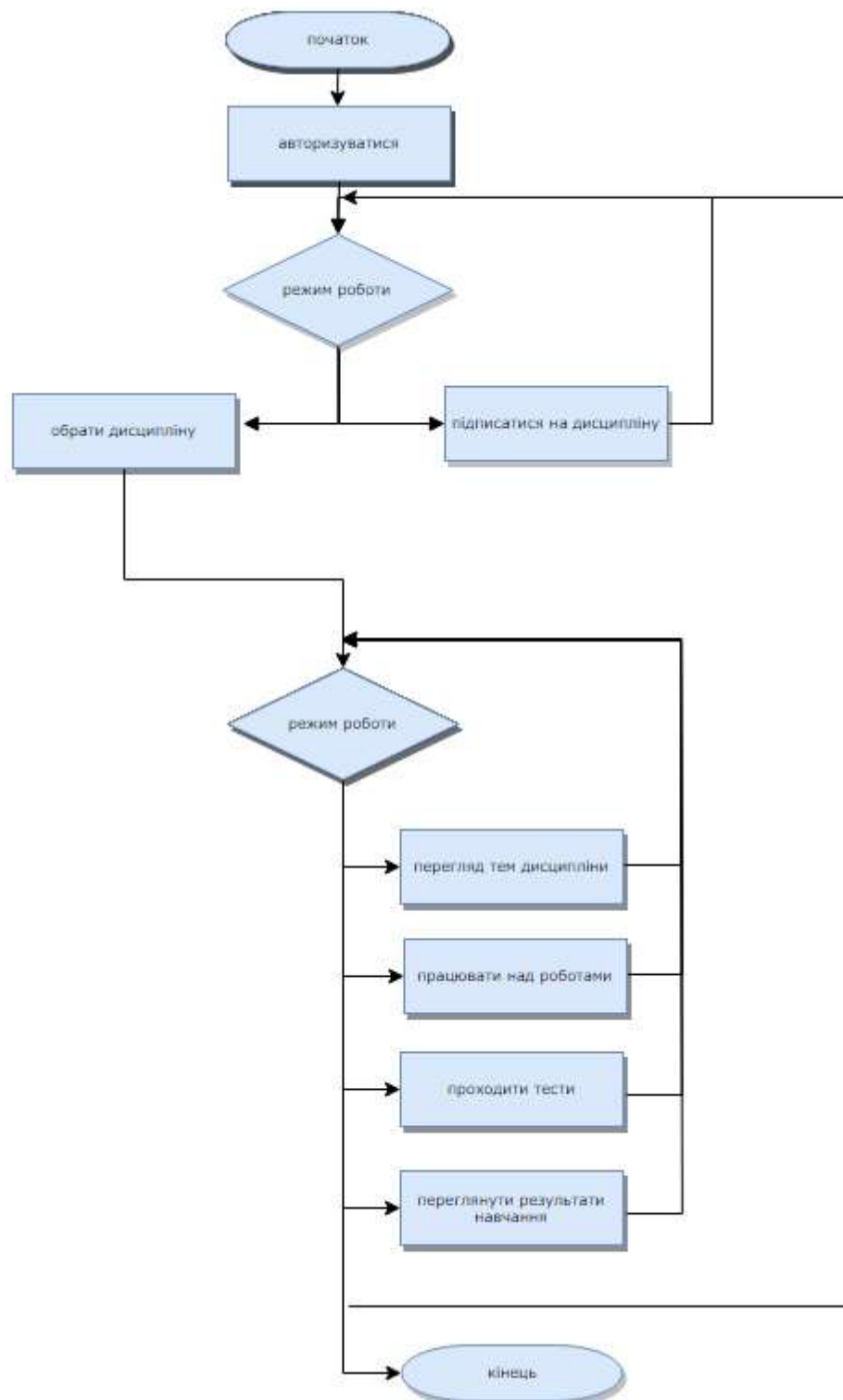


Рисунок 4.2 – Схема функціонування програми для студента

4.2 Опис модулів реалізованої системи

Програмне забезпечення реалізовано у вигляді вебзастосунку.

Процес роботи в межах застосунку розділено на декілька етапів. На першому, в межах підсистеми взаємодії з базою даних, реалізується витягання та заповнення даних. Безпосередньо представлення даних реалізується на рівні підсистем роботи адміністратора, викладача та студента.

Для формування вебпредставлення використано вебфреймворк Flask. Виділення окремого шару для роботи з базою даних відбувалося головним чином через те, що в перспективі заплановано створення настільного застосунку мовою програмування Python, відповідно потрібен універсальний спосіб для роботи з даними.

Модуль для роботи з базою даних представлено файлом `mysqldatawork.py`. Він визначає засоби роботи з базою даних, зокрема витягання даних, заповнення даних, а також початкового створення бази даних і таблиць.

Функція `create()` відповідає за створення бази даних та таблиць цієї бази даних у випадку їх відсутності в базі даних.

Внесення даних в таблицю бази даних реалізується шляхом створення курсору для виконання запиту до бази даних:

```
jeahcrs = JEAHdb.cursor()
SQLq = "INSERT INTO DEHdb.Course (cname, crterm, crform)
VALUES (%s, %s, %s)"
zm = (name, term, form)
jeahcrs.execute(SQLq, zm)
JEAHdb.commit()
```

Отримання даних про об'єкт (у даному випадку дисципліну) за його ідентифікаційним номером реалізується наступним чином:

```

jeahcrs = JEAHdb.cursor()
jeahcrs.execute("SELECT * FROM DEHdb.Course WHERE crid =
"+str(id))
data = jeahcrs.fetchone ()
JEAHdb.commit()
return data

```

Для створення вебсторінок застосунку було використано безкоштовний шаблон Landed [14]. На основі його використання було створено вебсторінки для організації функціональності, описаної в розділі 3. Вебсторінки зокрема включають:

- login.html – сторінка авторизації;
- studdisc.html – базова сторінка вибору дисципліни для студента;
- studadddisc.html – сторінка підписки користувачем на дисципліну;
- teachdisc.html – базова сторінка вибору дисципліни для викладача;
- teachtopics.html – сторінка роботи викладача з темами дисципліни;
- teachaddtopic.html – сторінка додавання теми викладачем;
- teachstuds.html – сторінка роботи з переліком студентів за дисципліною;
- teachdteach.html – сторінка керування викладачем викладачами за дисципліною;
- teachworks.html – сторінка перегляду переліку робіт за дисципліною;
- teachwork.html – сторінка оцінювання роботи студента викладачем;
- teachstud.html – сторінка перегляду результатів студентів, що навчаються за дисципліною;
- studdisctopics.html – сторінка перегляду тем дисципліни студентом.

Кожна вебсторінка є по суті шаблоном, адже окрім стандартного HTML містить ще конструкції для роботи з даними, які на основі звернень до модуля роботи з базою даних витягають ці дані і надають для виведення на сторінці.

Передача даних для роботи вебсторінки реалізується наступним чином:

```
render_template('teachtopics.html', topics=topics)
```

Відповідно заповнення набору тем `topics` реалізується за рахунок виклику функції, що виконує звернення до бази даних і витягання тем з неї.

Для створення моделі нейронної мережі було застосовано бібліотеку Keras. На її основі реалізується створення моделі, навчання та прогнозування за її допомогою тривалості на основі вхідних параметрів.

Зокрема створення шару GRU виконується наступним чином:

```
EducationModel.add(GRU(32, return_sequences = False))
```

4.3 Висновки за розділом 4

Приведено схеми функціонування програми для викладача та студента, описано логіку взаємодії.

Реалізовано програмне забезпечення у вигляді вебзастосунку. Для реалізації використано вебфреймворк Flask. Описано основні принципи реалізації модулів програмної системи.

5 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

5.1 Призначення програми

Програмне забезпечення призначене для підтримки процесу дистанційного навчання. Програма дозволяє виконувати керувати матеріалами за дисциплінами та процесом навчання загалом, забезпечувати можливість навчання за допомогою програми, прогнозувати тривалість вивчення дисципліни, визначати як особисту підписку на вивчення дисципліни, так і підписування на навчання зі сторони адміністратора.

Програмне забезпечення розроблено у вигляді вебзастосунку.

5.2 Умови виконання програми

Мінімальні вимоги до серверу, на якому має забезпечуватися доступ до бази даних:

- двоядерний процесор з частотою 2 ГГц;
- обсяг оперативного запам'ятовуючого пристрою, що дорівнює 1 Гб;
- вільний простір на жорсткому диску, що дорівнює 1 Гб.

Найголовнішою вимогою до організації програмної сторони роботи сервера є встановлення та підтримка на ньому системи керування базами даних MySQL версії не нижче ніж 5.6. Надання постійного доступу до сервера бази даних є обов'язковою умовою забезпечення експлуатації програми.

5.3 Виконання програми

Для виконання програми потрібно запустити її на сервері через запуск інтерпретатором скрипта `manage.py` з ключем `runserver`.

Після звернення з клієнтів через веббраузер до сервера відкривається стандартна сторінка авторизації. Неавторизовані користувачі не можуть працювати з програмою.

Якщо авторизувався обліковий запис, який пов'язаний з викладачем, то для нього відкривається сторінка, на якій він має обрати одну з власних дисциплін для подальшої роботи з нею (рис. 5.1).

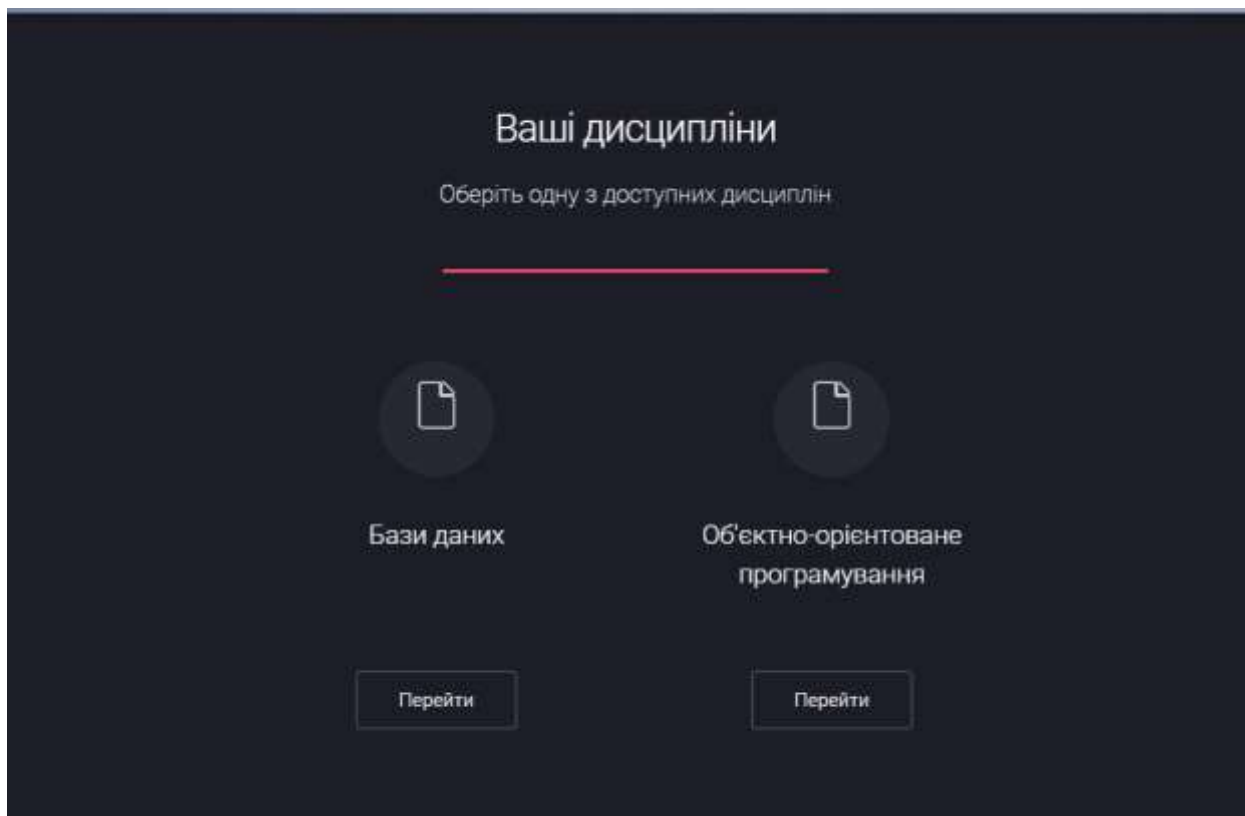


Рисунок 5.1 – Вибір дисципліни викладачем

Після того, як буде відкрито відповідну сторінку для роботи над дисципліною, буде надано доступ до наступних розділів, оформлених у вигляді відповідних пунктів меню:

- розділ «Дисципліни» дозволяє повернутися до вибору дисципліни викладачем;
- розділ «Теми» дозволяє структурувати роботи з дисципліною за темами;

- розділ «Студенти» дозволяє працювати зі всіма студентами, які навчаються дисципліні;
- розділ «Роботи» дозволяє окремо переглянути роботи, подані студентами;
- розділ «Викладачі» дозволяє керувати роботою викладачів над курсом (рис. 5.2).

Об'єктно-орієнтоване програмування

Дисципліни Теми Студенти Роботи Викладачі **Вийти**

Викладачі

Вже додані

Роль	Ім'я	Академічні дані	Дія	
Лектор	Гепко Ігор Петрович	к.т.н., доцент, доцент кафедри програмних засобів	Редагувати	Вилучити
Асистент	Зайченко Віктор Леонідович	ст. викладач кафедри програмних засобів	Редагувати	Вилучити

Додати викладача

Черненко

Шукати

Черненко Іван Олексійович

Асистент

Додати

Рисунок 5.2 – Вибір викладачів для роботи над дисципліною

Розділ «Викладачі» дозволяє переглянути, які викладачі працюють над дисципліною, а також змінити їх ролі і призначити нові. У даному розділі відображається таблиця з викладачами та їх роллю. У кожному рядку цієї таблиці для кожного викладача відображаються кнопки «Редагувати» і «Вилучити», що дозволяють змінити права щодо роботи викладача над

дисципліною. За замовчуванням асистентам заборонено виставляти загальну оцінку за дисципліною, але при цьому він може оцінювати окремі роботи.

Окрім того надається можливість додати нову роль. Для цього потрібно ввести прізвище викладача, потім натиснути на кнопку «Шукати», і випадаючий перелік нижче буде заповнено даними викладачів, які мають таке прізвище. Для обраного викладача можна обрати його роль, а для збереження цих даних потрібно натиснути на кнопку «Додати».

Через розділ «Теми» викладач може додавати матеріали за дисципліною, створювати тести, додавати необхідні роботи, а також керувати цими даними.

Для оцінювання робіт студента викладачу потрібно перейти в розділ «Роботи», де відкриється сторінка з таблицею доступних робіт для перевірки, на яку можна вплинути за допомогою випадаючого переліку, тим самим визначаючи, який саме студент цікавить викладача. Далі за ним можна відкрити потрібну для перевірки роботу. Відповідна сторінка відкриється на наступному кроці (рис. 5.3).

На даній сторінці відображається інформація про назву роботи, її порядковий номер, студента, який виконав роботу, опис, отриманий від студента. За допомогою кнопки «Відкрити» біля поля «Файл» можна відкрити файл звіту за роботою.

Можна виставити оцінку роботі, при цьому відображається одразу і максимальний результат, який можна отримати за виконання роботи.

Можна також написати зауваження за роботою, яке відобразиться студенту. Зауваження можна залишити без виставлення оцінки, якщо викладач рекомендує продовжити працювати над роботою.

На завершення потрібно натиснути на кнопку «Повернути», тоді внесені дані будуть збережені і відображені студенту, коли він відкриє сторінку роботи. Таким чином, викладачу надається гнучкий і простий засіб керування роботою зі студентами щодо усунення недоліків роботи.

Об'єктно-орієнтоване програмування

Дисципліни Теми Студенти Роботи Викладачі **Вийти**

Робота #3

Наслідування

Студент

Хлопотько Сергій Вікторович

Опис від студента

Для мене це була складна робота. Потребую додаткової консультації

Файл

Відкрити

Оцінка

0/10

☒ зауваження

Введіть повідомлення

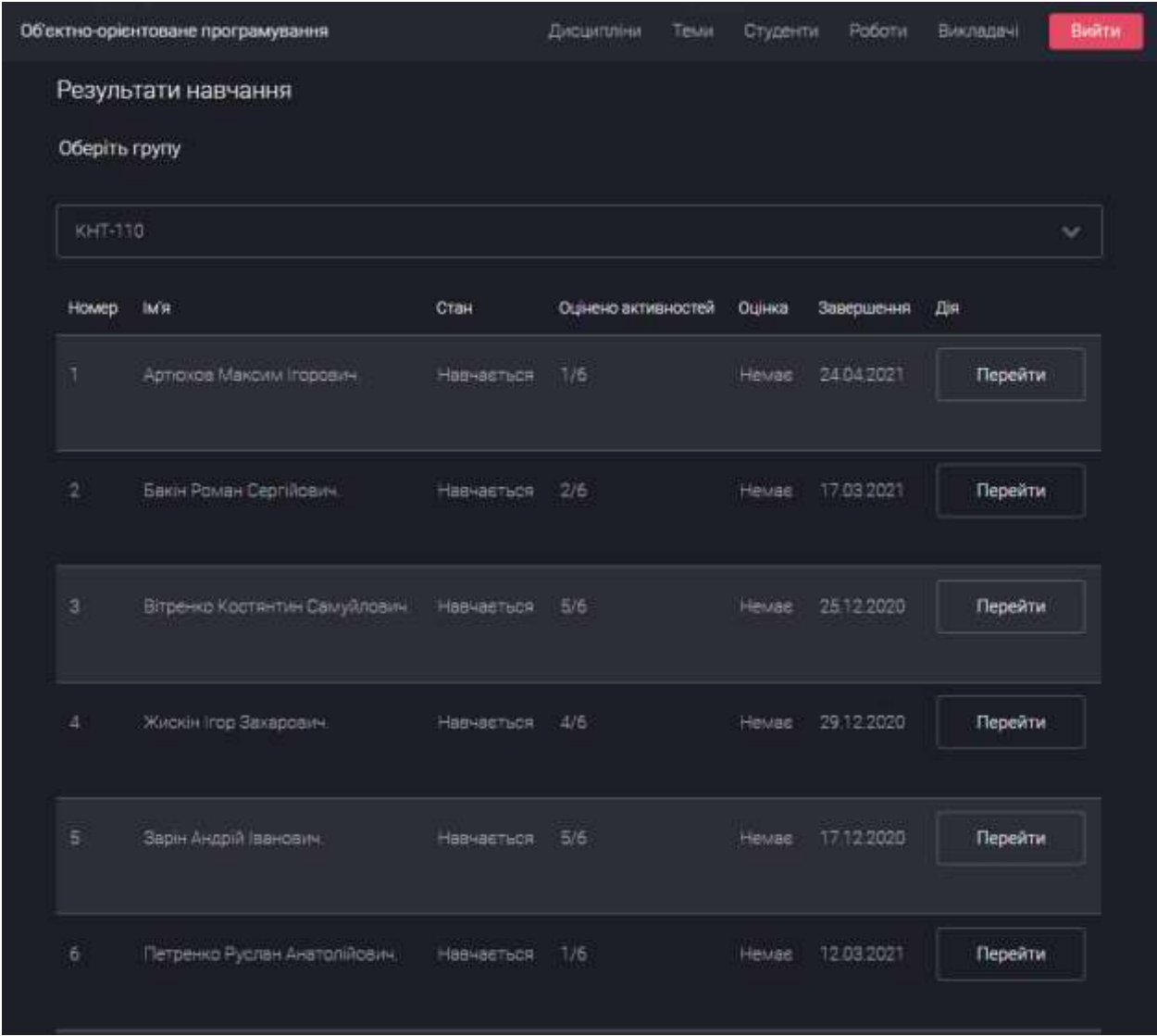
Повернути

Рисунок 5.3 – Оцінювання роботи студента

Пункт меню «Студенти» (рис. 5.4) дозволяє переглянути всіх студентів, що вивчають дисципліну, та переглянути їх результати. Для цього спочатку потрібно обрати номер групи з випадającego переліку. Далі буде відображено всіх студентів групи в таблиці. В таблиці надається інформація

про кількість виконаних активностей і активностей загалом. До активностей об'єднано роботи і тести.

Для кожного студента за дисципліною показується або стан «Навчається» і тоді в стовпці «Завершення» виводиться прогнозоване значення завершення навчання студента за даною дисципліною. Тобто це значення як раз і отримується на основі виконаного прогнозування тривалості навчання на основі розробленого методу.



Номер	Ім'я	Стан	Оцінено активностей	Оцінка	Завершення	Дія
1	Артюхов Максим Ігорович	Навчається	1/5	Немає	24.04.2021	Перейти
2	Бакін Роман Сергійович	Навчається	2/6	Немає	17.03.2021	Перейти
3	Вітренко Костянтин Самуїлович	Навчається	5/6	Немає	25.12.2020	Перейти
4	Жискін Ігор Захарович	Навчається	4/6	Немає	29.12.2020	Перейти
5	Зв'їн Андрій Іванович	Навчається	5/6	Немає	17.12.2020	Перейти
6	Петренко Руслан Анатолійович	Навчається	1/6	Немає	12.03.2021	Перейти

Рисунок 5.4 – Перегляд результатів студентів за дисципліною

Звідси можна перейти до сторінки докладнішого перегляду результатів обраного з таблиці студента.

У той же час, якщо в застосунку авторизується студент, то для нього буде відображено інші сторінки з іншим меню.

Спочатку йому також потрібно буде обрати дисципліну. Окрім того він може виконати пошук і підписатися на дисципліну, якщо для цієї сторінки цей процес відкритий.

Меню студента включає наступні пункти:

- розділ «Дисципліни», що дозволяє повернутися до вибору дисципліни;
- розділ «Теми», що дозволяє переглянути всі теми за дисципліною;
- розділ «Роботи», що дозволяє переглянути власні здані роботи та завдання на них;
- розділ «Тести» (рис. 5.5) дозволяє пройти доступні на даний момент тести;
- розділ «Результати» дозволяє переглянути результати вивчення дисципліни (бали загальні зокрема і за окремі роботи).

Об'єктно-орієнтоване програмування

Дисципліни Теми Роботи Тести Результати Вийти

Тест #1

Залишилось часу: 29:20

1. Як називається функція, яка має бути перевизначена в похідних класах?

☒ виртуальна ☐ збудована

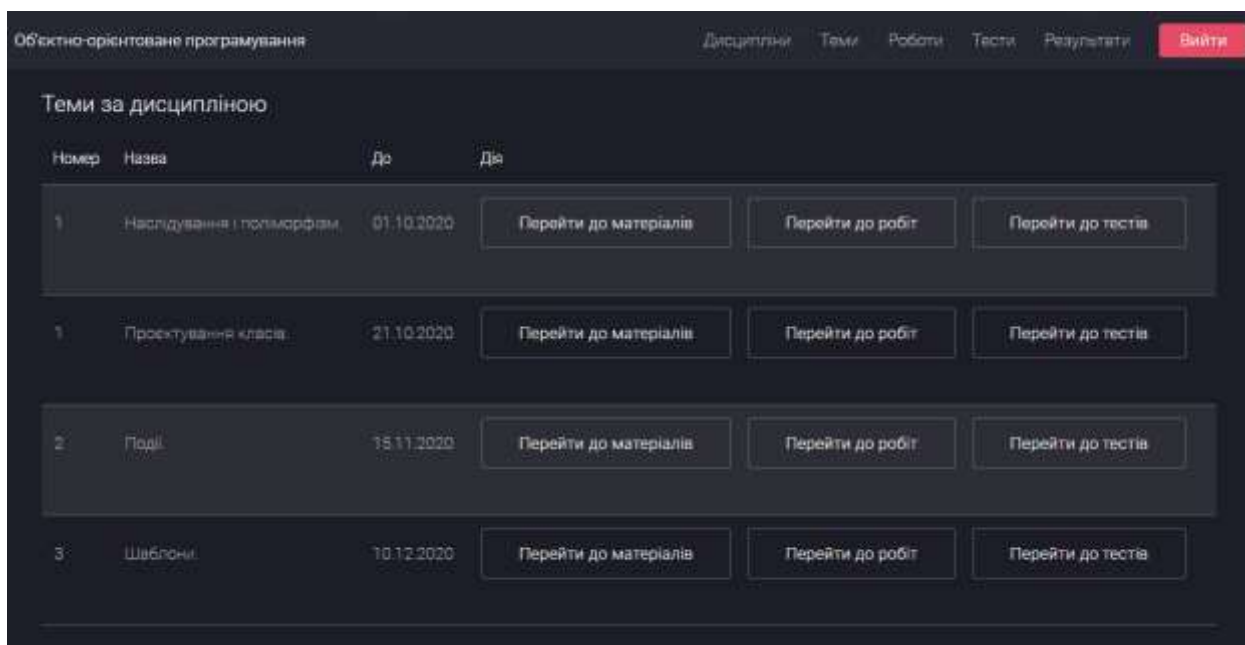
☐ перевантажена ☐ інша відповідь

Назад Далі

Рисунок 5.5 – Проходження тесту студентом

Під час проходження тесту студент може переміщуватися між запитаннями, обирати один з варіантів відповіді, як в даному випадку, а також обирати декілька правильних варіантів на запитання, де обрано такий варіант. Студенту відображається також час до завершення роботи над тестом.

Рис. 5.6 відображає перелік тем за дисципліною. Для кожної дисципліни можна окремо перейти до перегляду матеріалів, робіт та тестів.



Номер	Назва	До	Дя
1	Наслідування і поліморфізм	01.10.2020	Перейти до матеріалів Перейти до робіт Перейти до тестів
1	Проектування класів	21.10.2020	Перейти до матеріалів Перейти до робіт Перейти до тестів
2	Події	15.11.2020	Перейти до матеріалів Перейти до робіт Перейти до тестів
3	Шаблони	10.12.2020	Перейти до матеріалів Перейти до робіт Перейти до тестів

Рисунок 5.6 – Перегляд тем за дисципліною

6 ЕКОНОМІКО-ОРГАНІЗАЦІЙНА ЧАСТИНА

Програмне забезпечення призначене для підтримки процесу навчання. Окрім безпосередньо забезпечення процесу віддаленого навчання, програмне забезпечення дозволяє ще прогнозувати тривалість навчання, таким чином оптимізуючи даний процес під конкретного студента.

Розроблене програмне забезпечення може використовуватися як самостійно у вигляді окремого програмного застосунку, так і у вигляді модуля, який підключається у програмне забезпечення для виконання прогнозування тривалості навчання. Відповідні програмні інтерфейси вимагають чималих витрат за їх використання.

Зокрема відповідний програмний інтерфейс iSpring Learn у версії, що відповідає розроблюваній функціональності, коштує 29858 грн./рік, за використання бібліотеки Mirapolis LMS для подібних цілей вартує 221120 грн./рік.

Дана розробка має поставлятися значно дешевше, що дозволить вигравати у перелічених аналогів за рахунок вартості. Розраховано поставити 100 копій програми з вартістю у 2500 грн. кожна.

6.1 Планування розробки програмного продукту

У даному підрозділі реалізується планування розробки програмного продукту.

В розробці приймають участь програміст протягом 3 місяців та консультант протягом 0,25 місяця. Розробка програмного забезпечення починається 7 вересня та повинна бути виконана до 30 листопада 2020 року. Тривалість робіт розраховується за формулою (6.1):

$$T_{\text{ц}} = \frac{Q}{R}, \quad (6.1)$$

де $T_{\text{ц}}$ – тривалість циклу, днів;

Q – трудомісткість, людино-днів;

R – кількість виконавців, люд. [15].

Отримана інформація зведена в табл. 6.1.

Таблиця 6.1 – Характеристика робіт по розробці програми

Назва роботи	Трудомісткість		Виконавці	Тривалість, днів
	люд.-дні	% від підсумку		
1. Аналіз предметної області	10	15%	Програміст Консультант	5
2. Визначення вимог до програмного продукту	5	8%	Програміст	5
3. Проектування структури програми	10	15%	Програміст	10
4. Розробка схеми функціонування програми	5	8%	Програміст	5
5. Написання програмного коду	15	24%	Програміст	15
6. Тестування та налагодження програмного продукту	10	15%	Програміст	10
7. Складання програмної документації	10	15%	Програміст	10
Всього	65	100		60

Найменування робіт	Календарний період, дні												
	07.09-11.09	14.09-18.09	21.09-25.09	28.09-02.10	05.10-09.10	12.10-16.10	19.10-23.10	26.10-30.10	02.11-06.11	09.11-13.11	16.11-20.11	23.11-27.11	30.11-30.11
1. Аналіз програмного забезпечення													
2. Визначення вимог до програмного продукту													
3. Проектування структури програми													
4. Розробка схеми функціонування програми													
5. Написання програмного коду													
6. Тестування та налагодження програмного продукту													
7. Складання програмної документації													

Чорним у табл. 6.2 позначено роботу програміста, а темно-сірим – консультанта.

6.2 Визначення витрат на розробку програми

6.2.1 Розрахунок основної заробітної плати

Витрати за цією статтею складаються з планового фонду зарплатні всіх категорій працівників, зайнятих у розробці програми, розрахунок ведеться на основі даних про трудомісткість, представлених в табл. 6.3 [15].

Таблиця 6.3 – Розрахунок основної заробітної плати

Посада виконавця	Численність, люд.	Місячний оклад, грн.	Кількість місяців роботи	Сума заробітної платні, грн
Програміст	1	12000	3	36000
Консультант	1	10000	0,25	2500
Всього	2			38500

6.2.2 Розрахунок додаткової заробітної плати

Додаткову заробітну плату $ЗП_{\text{дод}}$ приймають рівною 10% від основної заробітної плати працівників $ЗП_{\text{осн}}$ і розраховують по формулі (6.2):

$$ЗП_{\text{дод}} = ЗП_{\text{осн}} \cdot 0,1 \text{ [15]}. \quad (6.2)$$

Підставивши величину основної заробітної плати в дану формулу (6.2), отримуємо:

$$ЗП_{\text{дод}} = 38500 \cdot 0,1 = 3850 \text{ грн.}$$

6.2.3 Відрахування на соціальне страхування та інші фонди

Ці відрахування складаються з єдиного соціального внеску – 22%, і беруться від основної та додаткової зарплатні за допомогою формули (6.3):

$$\begin{aligned} OT &= (ЗП_{\text{осн}} + ЗП_{\text{дод}}) \cdot 0.22 \text{ [15]}, \\ OT &= (38500 + 3850) \cdot 0.22 = 9317 \text{ грн.} \end{aligned} \quad (6.3)$$

6.2.4 Визначення витрат на матеріали

Використовується 2 найменування матеріалів: картридж –115 грн. та папір 95 грн. (1 упаковка).

Витрати на матеріали розраховуються за формулою (6.4):

$$M = \sum_{i=1}^n (C_i \cdot N_i \cdot (1 + K_{\text{т.з.}}) - C_{\text{ів}} \cdot N_{\text{ів}}), \quad (6.4)$$

де M – витрати на матеріали, покупні напівфабрикати та комплектуючі вироби, грн.;

$K_{\text{т.з.}}$ – коефіцієнт, враховуючий транспортно-заготівельні витрати;

C_i – ціна i -го найменування матеріалу, напівфабрикату і комплектуючого, грн.;

N_i – потреба в i -ому матеріалі, напівфабрикаті і комплектуючому;

$C_{\text{ів}}$ – ціна зворотних відходів i -го найменування матеріалу, грн.;

$N_{\text{ів}}$ – кількість зворотних відходів i -го найменування;

n – кількість найменувань матеріалів, напівфабрикатів та комплектуючих [15].

Розрахуємо:

$$\begin{aligned} \Pi_{iB} &= 0, N_{iB} = 0, K_{T.3.} = 0,05, \\ M &= (1 + 0,05) \cdot (115 + 95) = 220,50 \text{ грн.} \end{aligned}$$

Всього, витрати на матеріали складають 220,50 грн.

6.2.5 Витрати на спеціальне обладнання

В цю статтю входять витрати на придбання, транспортування, монтаж та налагодження нестандартного обладнання, практично, у даному випадку, в цій статті враховуються витрати на оплату машинного часу електронно-обчислювальної машини (ЕОМ) для написання та налагодження даної програми [15].

Амортизаційні відчислення визначають за формулою (6.5):

$$A = \Phi_6 \cdot \frac{H_a}{100}, \quad (6.5)$$

де Φ_6 – балансова вартість обчислювальної техніки, грн.;

H_a – норма амортизаційних відрахувань на повне відновлення обчислювальної техніки, % [15].

Балансова вартість обчислювальної техніки складає 15000 грн.

Отримуємо:

$$A = 15000 \cdot 0,25 = 3750 \text{ грн.}$$

Статтю «Експлуатація обладнання» розраховують підсумовуванням витрат на електроенергію та допоміжні матеріали за допомогою формули (6.6):

$$C_E = N_H \cdot \Phi_{ef} \cdot K_{зч} \cdot K_{зп} \cdot C_e, \quad (6.6)$$

де N_n – номінальна потужність ЕОМ, кВт;

$\Phi_{\text{еф}}$ – річний ефективний фонд часу роботи ЕОМ, машино-год;

$K_{\text{зч}}$ – середній коефіцієнт завантаженості за часом;

$K_{\text{зп}}$ – коефіцієнт завантаженості по потужності;

$\text{Ц}_\text{е}$ – ціна однієї кВт·години електроенергії, грн./(кВт·годин) [15].

Номінальна потужність ЕОМ – 0,2 кВт. Річний ефективний фонд часу роботи ЕОМ складає 1800 годин. Середні коефіцієнти завантаження за часом и потужності рівні відповідно 0,9 і 0,6. Ціна однієї кіловат-години електроенергії складає 2,11 грн.

Отримуємо:

$$C_E = 0,2 \cdot 1800 \cdot 0,9 \cdot 0,6 \cdot 2,11 = 410,184 \text{ грн.}$$

Зарплатня обслуговуючого персоналу розраховується за формулою (6.7):

$$ЗП_{\text{обсл}} = \Phi ЗП_{\text{р}} \cdot (1 + K_{\text{відч}}) \cdot \frac{t_{\text{обсл}}}{\Phi_{\text{еф.обсл}}}, \quad (6.7)$$

де $\Phi ЗП_{\text{р}}$ – річний фонд заробітної плати (основної та додаткової) обслуговуючих працівників, грн.;

$K_{\text{відч}}$ – коефіцієнт, враховуючий відчислення на соціальне страхування та в інші фонди;

$t_{\text{обсл}}$ – час (протягом року), необхідний на технічне обслуговування ЕОМ, год/рік;

$\Phi_{\text{еф.обсл}}$ – річний ефективний фонд часу обслуговуючого персоналу, год./рік [15].

Місячна заробітна платня обслуговуючого персоналу складає 5000 грн., а річний фонд заробітної платні відповідно дорівнює 60000 грн. Річний ефективний фонд робочого часу працівника, що обслуговує персональний

комп'ютер (ПК), дорівнює 1750 год./рік. На обслуговування одного персонального комп'ютера витрачається 1 година на місяць.

Отримуємо:

$$ЗП_{\text{обсл}} = 60000 \cdot (1 + 0,22) \cdot \frac{12}{1750} = 501,94 \text{ грн.}$$

Стаття «Поточний ремонт обладнання» приймається рівною 3% від балансового кошту обладнання і складає 450 грн.

Стаття «Інші витрати» приймається рівною п'яти відсоткам від суми всіх попередніх витрат на утримання та експлуатацію обладнання [15]. Сума всіх попередніх статей дорівнює 5112,124 грн., 5% від суми складають 255,61 грн.

Розраховані статті витрат на утримання та експлуатацію обладнання внесені в табл. 6.4.

Таблиця 6.4 – Кошторис витрат на утримання та експлуатацію обладнання

Найменування статей витрат	Сума, грн.
Амортизація обладнання	3750,00
Експлуатація обладнання	410,18
Заробітна плата основна та додаткова обслуговуючих працівників з відрахуваннями на соціальні заходи	501,94
Поточний ремонт обладнання	450,00
Інші витрати	255,61
Всього	5367,73

Витрати на оплату машинного часу ЕОМ для написання та налагодження даної програми визначаються за формулою (6.8):

$$C_{mo} = B_{екс} \cdot t_{mo}, \quad (6.8)$$

де C_{mo} – витрати на оплату машинного часу, грн.;

$B_{екс}$ – експлуатаційні витрати на одну годину машинного часу цієї цифрової ЕОМ, грн./машино-год.;

t_{mo} – машинний час цифрової ЕОМ для написання та налагодження даного програмного продукту, машино-год. [15].

Річний ефективний фонд часу роботи електронно-обчислювальної машини (ЕОМ) дорівнює 1800 годин. У результаті експлуатаційні витрати на одну годину машинного часу дорівнюють:

$$B_{екс} = \frac{5367,73}{1800} = 2,982 \text{ грн/машино-годину.}$$

ЕОМ експлуатується 60 днів за один кошторис, що складає в сумі 480 годин. Таким чином, витрати на оплату машинного часу складають:

$$C_{mo} = 2,982 \cdot 480 = 1431,36 \text{ грн.}$$

6.2.6 Інші прямі витрати

Програмне забезпечення, що використовується постійно (Windows 10 Home), коштує, відповідно витрати обчислюються за формулою (6.9):

$$S_{\text{час.ПЗ}} = \frac{Ц_{\text{ПЗ Windows}} \cdot T_{\text{КТС}}}{\Phi_{\text{еф.КТС}} \cdot T_{\text{т.ПЗ}}}, \quad (6.9)$$

де $S_{\text{час.ПЗ}}$ – витрати на програмне забезпечення при написанні розроблюваного програмного забезпечення, грн;

$Ц_{\text{ПЗ Windows}}$ – ціна програмного забезпечення Windows, грн;

$T_{\text{КТС}}$ – машинний час, необхідний користувачу при розробці системи, машино-год./рік;

$\Phi_{\text{еф.кТС}}$ – річний ефективний фонд часу роботи, машино-год./рік;

$T_{\text{т.ПЗ}}$ – термін служби часткового програмного забезпечення, років [15].

Розрахуємо:

$$S_{\text{час.ПЗ}} = \frac{4159 \cdot 480}{1800 \cdot 5} = 221,8 \text{ грн.}$$

6.2.7 Розрахунок накладних витрат

У даному випадку накладні витрати складають 20% до основної заробітної плати розробників, що складає 7700 грн. Результати визначення витрат на розробку програми наведені в табл. 6.5.

Таблиця 6.5 – Калькуляція кошторисної вартості робіт з розробки програми та її ціна

№	Найменування статей витрат	Сума, грн.	Питома вага від підсумку, %
1	Основна заробітна плата	38500	62,87
2	Додаткова заробітна плата	3850	6,29
3	Єдиний соціальний внесок	9317	15,21
4	Матеріали та комплектуючі	220,50	0,36
5	Витрати на спец. обладнання	1431,36	2,34
6	Інші прямі витрати	221,8	0,36
7	Накладні витрати	7700	12,57
8	Всього ($S_{\text{бП}}$)	61240,66	100,00
9	Прибуток (30%)	18372,20	
10	Всього, з урахуванням податку на додану вартість (20%)($C_{\text{пн}}$)	95535,43	

6.3 Розрахунок економічної ефективності програмного продукту

Планується продати 100 копій програмного забезпечення по 2500 гривень кожна. З урахуванням ризику 50% надходження від продажів складуть 125 тис. грн., що більше ціни програми, в якій вже була врахована рентабельність 30%.

Таким чином впровадження програмного продукту є економічно доцільним.

7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

7.1 Аналіз потенційних небезпек

В дипломній кваліфікаційній роботі магістра проводиться дослідження та програмна реалізація методів визначення тривалості навчання. Основна робота при розробці буде проводитись за комп'ютером, тому слід визначити фактори, що так чи інакше впливають на людину:

- незадовільні параметри мікроклімату робочої зони (значення показників мікроклімату у виробничих приміщеннях повинні задовольняти нормам, зазначеними у Державних санітарних нормах 3.3.6.042-99 Санітарні норми мікроклімату виробничих приміщень [16]);

- відсутність або недостача природного освітлення;

- недостатня освітленість робочого місця, згідно Державних будівельних норм В.2.5-28:2018 [17];

- широкий спектр випромінювання від дисплея, який включає рентгенівську, ультрафіолетову і інфрачервону області, а так само широкий діапазон електромагнітних випромінювань інших частот (згідно Державних санітарних норм і правил 3.3.6.096-2002 «Державними санітарними нормами і правилами при роботі з джерелами електромагнітних полів» [18], Державних санітарних норм 239-96 «Державні санітарні норми і правила захисту населення від впливу електромагнітних випромінювань» [19] і ГОСТ 12.1.045-84 «Система стандартів безпеки праці. Електростатичні поля. Допустимі рівні на робочих місцях і вимоги до проведення контролю» [20]);

- небезпека виникнення пожежі (згідно ДСТУ 2272:2006 «Пожежна безпека» [21]);

- статична напруга, яка здатна вивести з ладу електронні компоненти;

- небезпечний рівень напруги який може видавати блок живлення ПК;

- погано організоване робоче місце, що може привести до уражень струмом, механічних травм та інше [21].

Розробка буде проводитись в приміщенні, що має розміри $25 \times 15 \text{ м}^2$, також у цьому приміщенні розташовано 24 робочих місця, обладнаних персональними комп'ютерами, слід відзначити, що приміщення має 3 досить великих вікна, але оскільки вони розташовані з одного боку та не по центру приміщення, то в приміщенні може виникнути проблема з освітленням, також приміщення обладнане великою кількістю розеток, що мають 3 контакти (фаза, нуль та заземлення), згідно з ДСТУ ІЕС 60884-1:2007, для забезпечення відповідного мікроклімату приміщення обладнано двома спліт-системами кондиціонування, вони ж і забезпечують іонізацію повітря та його зволоження [22].

7.2 Заходи забезпечення безпеки

Робота з обчислювальною технікою зумовлює виникнення фактору небезпеки, що пов'язаний з електрострумом.

Ураження електрострумом може бути наслідком невиконання техніки безпеки, при несправності електроприладів, при недотриманні вимог ДСТУ ІЕС 60884-1:2007 при обладнанні робочих місць тощо [22].

Дія електричного струму на людину має різносторонній характер. Електричний струм спричиняє термічну, електролітичну, біологічну та механічну дії [22].

Залежно від категорії приміщення здійснюються певні заходи, які забезпечують достатню безпеку при експлуатації і ремонті електроустаткування [23].

Для запобігання ураження людини, при дотику до частин ЕОМ, що знаходяться під напругою, виконується захисне заземлення. Суть захисного заземлення – зменшення напруги на корпусі електроустаткування при пробії фази на корпус до безпечної величини. Сучасні ЕОМ включаються в розетку за допомогою триполюсною вилки, один з полюсів якої використовується для заземлення. Згідно Правил улаштування електроустановок опір

захисного заземлення в приміщенні для комп'ютерної техніки не перевищує 4 Ом, що значно менше опору тіла людини, тому небезпека ураження при дотику до несправного електроустаткування значно зменшується, так як струм в цьому випадку піде шляхом найменшого опору [24].

При організації заземлення зв'язок корпусу ПК з робочим нулем мережі живлення неприпустимий, оскільки при обриві робочого нуля корпус ПК виявляється під «фазною» напругою, крім того, при великому навантаженні в електричній мережі на робочому нулі зосереджується значний потенціал [23].

Окрім враження електрострумом при розробці можна отримати механічні пошкодження від інструментів, що необхідні для розробки мережі, а саме, для роботи з кабелем (витою парою) необхідно використовувати спеціальний інструмент – кримпер, що має гострі леза для перерізання кабелів та має спеціальний механізм для стискання конекторів, тому це досить травмонебезпечний пристрій, який може спричинити механічні пошкодження частин тіла людини, тому при роботі з ним необхідно дотримуватись таких правил безпеки:

- ознайомитись з інструкцією приладу;
- одягнути спеціальні захисні перчатки для захисту рук;
- після роботи з інструментом, закрити гострі елементи спеціальним захисним ковпачком [25].

7.3 Заходи з виробничої санітарії та гігієни праці

У приміщеннях, де встановлені персональні комп'ютери, повинні дотримуватися визначені параметри мікроклімату, у Державних санітарних нормах 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень» установлені величини параметрів мікроклімату, що створюють комфортні умови, ці норми встановлюються в залежності від часу року, характеру трудового процесу і характеру виробничого приміщення [24].

На працівників впливає рівень шумів: допустимий рівень шуму в конструкторському приміщенні повинен становити 50 – 71 дБ згідно Державних санітарних норм 3.3.6-037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку» [26].

При проектуванні освітлювальної установки вирішуються такі питання:

- вибір типу джерела світла;
- визначення системи освітлення;
- вибір типу світильників з урахуванням умов середовища;
- розподіл світильників і визначення їх кількості;
- визначення норми освітленості на робочому місці [25].

Згідно Державних будівельних норм В.2.5-28:2018 “Природне і штучне освітлення” для найменшого розрізнення об'єктів (розряд зорової роботи III (б)) 0,3–0,5 мм значення коефіцієнта природного освітлення повинно дорівнювати 1,5 %, метою розрахунку штучного освітлення є перевірка його відповідності ДБН В.2.5-28:2018 [27].

Прийняті значення освітленості наведені в табл. 7.1.

Для забезпечення нормальних санітарно-гігієнічних умов у відповідності до вимог Державних будівельних норм В.2.-67:2013 «Опалення, вентиляція та кондиціонування» в приміщеннях передбачається загально обмінна припливно-витяжна механічна вентиляція і кондиціонування повітря, основним завданням кондиціонування повітря є підтримка параметрів повітряного середовища в допустимих межах, що забезпечують надійну роботу ЕОМ, тривале зберігання носіїв інформації та комфортні умови для обслуговуючого персоналу, кондиціонування здійснюється побутовими кондиціонерами, встановленими в вікнах, окрім цього приміщення обладнане вентиляцією [27].

Таблиця 7.1 – Прийняті значення освітлення

Найменування ділянки	Розряд зорової роботи	Освітленість, (лк) і коефіцієнт запасу при використанні			Аварійне освітлення, лк	
		система комбінованого освітлення	система звичайного освітлення	коефіцієнт запасу	для продовження робіт	для евакуації
Виробнича	3 б	600	300	1,5	6	1 лк на виходах

Кількість вентиляційного повітря визначається за формулою (7.1):

$$V_{\text{вент}} = \frac{3600 \cdot Q_{\text{надл}}}{C \cdot \rho \cdot (t_{\text{вуд}} - t_{\text{пр}})}, \quad (7.1)$$

де $Q_{\text{надл}}$ – виділення в приміщенні явного тепла, $Вт$;

C – теплоємність повітря ($C = 1000 \text{ Дж} / \text{кг} \cdot \text{град}$);

ρ – питома щільність повітря ($\rho = 1,3 \text{ кг} / \text{м}^3$);

$t_{\text{вуд}}$ і $t_{\text{пр}}$ – температури повітря, що видаляється та приточного, $^{\circ}\text{C}$.

Температура повітря, що видаляється визначається з формули (7.2):

$$t_{\text{вуд}} = t_{\text{зн}} + d(h - 2), \quad (7.2)$$

де $t_{\text{зн}}$ – температура повітря в зоні праці ($t_{\text{зн}} = 21^{\circ}\text{C}$);

d – коефіцієнт наростання температури на кожен метр висоти ($d = 1,5^{\circ}\text{C} / \text{м}$);

h – висота приміщення ($h = 5 \text{ м}$).

Звідси:

$$t_{вид} = 21 + 1,5(5 - 2) = 25,5 \text{ } ^\circ\text{C}.$$

Кількість надлишкового тепла визначається з теплового балансу, як різниця між теплом, що надходить у приміщення і теплом, що видаляється з приміщення і що поглинається в ньому, за формулою (7.3):

$$Q_{надл} = Q_{надх} - Q_{випр}. \quad (7.3)$$

Тепло, що надходить у приміщення, визначається за формулою (7.4):

$$Q_{надх} = Q_{устатк} + Q_{л} + Q_{осв} + Q_{рад}, \quad (7.4)$$

де $Q_{устатк}$ – тепло від роботи устаткування;

$Q_{л}$ – тепло, що надходить від людей;

$Q_{осв}$ – тепло від джерел освітлення;

$Q_{рад}$ – тепло від сонячної радіації, що надходить через вікна.

Відповідно за формулою (7.5):

$$Q_{устатк} = E_{устатк} \cdot P_{уст} = 0,15 \cdot 750 = 112,5 \text{ Вт}, \quad (7.5)$$

де $E_{уст}$ – частка енергії, що переходить у тепло;

$P_{уст}$ – потужність установки.

Тепло, що надходить від людей, розраховується за формулою (7.6):

$$Q_{л} = n \cdot q = 10 \cdot 90 = 900 \text{ Вт}, \quad (7.6)$$

де n – кількість осіб у залі ($n = 10$);

q – кількість тепла, що виділяється людиною ($q = 90 \text{ Вт}$).

Відповідно за формулою (7.7):

$$Q_{осв} = E_{осв} \cdot P_{осв} = 0,4 \cdot 305 = 122 \text{ Вт}, \quad (7.7)$$

де $E_{л} = 0,4$ для люмінесцентних ламп;

$P_{осв}$ – потужність освітлювальної установки.

Тепло від сонячної радіації, що надходить через вікна, розраховується за формулою (7.8):

$$Q_{рад} = A \cdot k \cdot S \cdot m = 180 \cdot 0,8 \cdot 4,5 \cdot 3 = 1944 \text{ Вт}, \quad (7.8)$$

де A – теплонадходження в приміщення з 1 м^2 скла ($127 - 234 \text{ Вт/м}^2$);

S – площа вікна ($S = 4,5 \text{ м}^2$);

m – кількість вікон ($m = 3$);

k – коефіцієнт, що враховує характер скління ($k = 0,8$).

З формули (7.6) отримуємо:

$$Q_{надх} = 112,5 + 900 + 122 + 1944 \approx 3078,5 \text{ Вт}.$$

Останній показник потрібно розрахувати за формулою (7.9):

$$Q_{випр} = 0,1 \cdot Q_{надх} = 0,1 \cdot 3078,5 = 307,85 \text{ Вт}. \quad (7.9)$$

Звідси за формулою (7.5) знаходимо:

$$Q_{надл} = 3078,5 - 307,85 \approx 2770,65 \text{ Вт}.$$

Знаходимо необхідний повітрообмін:

$$V_{\text{вент}} = \frac{3600 \cdot 2770,65}{1000 \cdot 1,3 \cdot (25,5 - 18)} = \frac{9974340}{9750} \approx 1023 \text{ м}^3 / \text{г}.$$

Визначаємо необхідну кратність повітрообміну за формулами (7.10)-(7.11):

$$K = \frac{V_{\text{вент}}}{V_{\text{прим}}}, \quad (7.10)$$

$$V_{\text{прим}} = n \cdot S_{\text{ос}} \cdot h, \quad (7.11)$$

де n – число осіб у приміщенні;

$S_{\text{ос}}$ – площа виробничого приміщення, що приходить на 1 особу (за нормами для розумово-фізичної праці $S_{\text{ос}} = 6 \text{ м}^2$);

h – висота приміщення.

Кратність повітрообміну:

$$K = \frac{1023}{10 \cdot 6 \cdot 5} = 3.41.$$

7.4 Заходи з пожежної безпеки

У приміщенні з невеликою щільністю розміщені електронно-обчислювальні машини і периферійні пристрої. Робоче приміщення відповідно до Нормативно-правового акту з пожежної безпеки Б.03.002-2007 «Норми визначення категорії приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою» [28] можна віднести до категорії «В».

Можливими причинами виникнення пожежі в даному приміщенні можуть бути:

- коротке замикання проводки;
- несправність електрообладнання;
- використання побутових електроприладів;
- недотримання умов техніки безпеки;
- недотримання правил роботи з електрообладнанням;
- недотримання умов протипожежної безпеки [28].

У зв'язку з цим необхідно передбачити заходи щодо пожежної безпеки згідно з Державними будівельними нормами В.1.1.7-2002 «Захист від пожежі. Пожежна безпека об'єктів будівництва» [29].

Кількість вогнегасників має відповідати вимогам Нормативно-правового акту з пожежної безпеки Б.03.001-2004, якими передбачене обов'язкова наявність двох вогнегасників на 60 м² площі приміщень. Перезарядка і ремонт вогнегасників виробляються в спеціалізованих організаціях на зарядних станціях. Експлуатація вогнегасників без чеки і пломби заводу-виготовлювача або організації, що проводила перезарядку, не дозволяється [30].

7.5 Заходи з забезпечення безпеки у надзвичайних ситуаціях

Великі аварії і катастрофи на об'єктах можуть виникати в результаті стихійного лиха, а також порушення технології виробництва, правил експлуатації різних машин, обладнання та встановлених заходів безпеки, їх впливу подібні стихійних лих [31].

При загрозі для життя виконують евакуацію персоналу. Для евакуації людей використовують основні та запасні входи і виходи, віконні прорізи і балкони, люки в перекриттях, іноді спеціально проробляють отвори в перегородках і стінах [31].

При виникненні загрози нападу противника місцевими органами влади і штабами цивільного захисту за допомогою засобів масової інформації передаються населенню постанови чи розпорядження про порядок дій. З цього часу радіо, телевізори повинні бути постійно включені для прийому нових повідомлень. У найкоротші терміни населення має вжити необхідних заходів захисту і включитися у виконання заходів, що проводяться для цивільного захисту [32].

ВИСНОВКИ

У дипломній кваліфікаційній роботі магістра виконано аналіз проблеми підтримки дистанційного навчання. Розглянуто форми виконання дистанційного навчання, переваги даного процесу. Виконано огляд програмного забезпечення, що може використовуватися для підтримки процесу навчання. Розглянуто системи Google Classroom і Moodle. Виділено їх особливості і недоліки.

Виконано постановку задачі визначення тривалості навчання. Обґрунтовано вибір рекурентних нейронних мереж для розв'язання задачі. Розглянуто архітектури рекурентних нейронних мереж, в результаті обрано архітектуру GRU.

Обрано для реалізації програми мову Python. Визначено функціональні вимоги до програми та побудовано відповідну діаграму прецедентів. Створено структурну схему бази даних та забезпечено відповідне створення такої бази даних на основі використання системи MySQL. Побудовано структуру програмної системи на основі розподілу функціональних обов'язків різних груп користувачів, виділення підсистеми реалізації розробленого методу та підсистеми взаємодії з базою даних.

Розроблено програмне забезпечення у вигляді вебзастосунку, що надає можливість виконувати створення дисципліни для викладання, визначення тем, набору робіт, тестів і матеріалів для викладання дисципліни, підписування студентів на проходження дисциплін, проходження тестів за дисципліною, перегляду матеріалів за дисципліною, студентів, підписаних на дисципліну, дисциплін, на які підписався студент, прогнозування тривалості навчання за дисципліною, завантаження виконаних робіт, створення коментарів щодо роботи, виставлення оцінки за роботу.

Наукова новизна дипломної роботи полягає в розробленому методі визначення тривалості навчання, що виконує прогнозування фактичної тривалості вивчення дисципліни студентом на основі тривалості вивчення

попередніх дисциплін ним та параметрів, що визначають нову дисципліну, на основі використання моделі нейронної мережі GRU, що дозволяє виконувати прийняття рішень про організацію роботи над вивченням дисципліни.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. What is Distance Learning? The Benefits of Studying Remotely [Electronic resource]. – Access mode : <https://www.uopeople.edu/blog/what-is-distance-learning/>
2. Шинковська, І.Л. Особливості дистанційного навчання в системі вищої освіти [Електронний ресурс] / І.Л. Шинковська, І.П. Заєць // Збірник статей учасників XV Всеукраїнської практичнопізнавальної конференції «Наукова думка сучасності і майбутнього». – Режим доступу : <http://naukam.triada.in.ua/index.php/konferentsiji/45-p-yatnadtsyata-vseukrajinska-praktichno-piznavalna-internet-konferentsiya/269-osoblivosti-dstantsijnogo-navchannya-v-sistemi-vishchoji-osviti>
3. What Is Distance Learning? And Why Is It So Important? [Electronic resource]. – Access mode : <https://www.viewsonic.com/library/education/what-is-distance-learning-and-why-is-it-so-important/>
4. Keles, M.K. A Review of Distance Learning and Learning Management Systems [Electronic resource] / Mümine Kaya Keles, Selma Ayse Ozel. – Access mode : https://www.researchgate.net/publication/311753469_A_Review_of_Distance_Learning_and_Learning_Management_Systems
5. Sharma, A. Discovering Learning Management Systems: Basic Functions And Benefits [Electronic resource] / A. Sharma. – Access mode : <https://elearningindustry.com/discovering-learning-management-systems-basic-functions-benefits>
6. Хайкин, С. Нейронные сети [Текст] : полный курс / С. Хайкин. – М. : Вильямс, 2016. – 1104 с.
7. Prediction using neural networks [Electronic resource]. – Access mode : <https://expressanalytics.com/blog/prediction-using-neural-networks/>
8. Gupta, D. Fundamentals of Deep Learning – Introduction to Recurrent Neural Networks [Electronic resource] / D. Gupta. – Access mode :

<https://www.analyticsvidhya.com/blog/2017/12/introduction-to-recurrent-neural-networks/>

9. Николенко, С. Глубокое обучение [Текст] / С. Николенко, А. Кадури, Е. Архангельская. – СПб. : Питер, 2018. – 480 с. : ил.
10. Гудфеллоу, Я. Глубокое обучение [Текст] / Я. Гудфеллоу, И. Бенджио, А. Курвилль. – 2-е изд., испр. – М. : ДМК Пресс, 2018. – 652 с.
11. Nader, Y. Python vs Java in 2020 [Electronic resource] / Y. Nader. – Access mode : <https://hackr.io/blog/python-vs-java>
12. Python vs. Java [Electronic resource] : Comparing the Pros, Cons, and Use Cases. – Access mode : <https://www.stxnnext.com/blog/python-vs-java-comparison/>
13. Маккинли, У. Python и анализ данных [Текст] / У. Маккинли. – М. : ДМК Пресс, 2015. – 482 с.
14. Landed | HTML5 UP [Electronic resource]. – Access mode : <https://html5up.net/landed>
15. Методичні вказівки по економічному обґрунтуванню дипломних проектів для студентів спеціальностей 8.05010301 «Програмне забезпечення автоматизованих систем» та 8.05010201 «Інформаційні технології проектування» для усіх форм навчання [Текст] / В.В. Остапенко, Є.М. Сердюк. – Запоріжжя : ЗНТУ, 2015. – 22 с.
16. ДСН 3.3.6.042-99 Санітарні норми мікроклімату виробничих приміщень [Текст]. – Харків : Форт, 2000. – 24 с.
17. ДБН В.2.5-28:2018. Природне та штучне освітлення. Посібник [Текст] : проблеми природного і штучного освітлення. – К. : Етін, 2019. – 180 с.
18. ДСанНіП 3.3.6.096-2002 Державні санітарні норми і правила при роботі з джерелами електромагнітних полів [Текст]. – К. : МОЗ України, 2002. – 74 с.
19. ДСН 239-96 Державні санітарні норми і правила захисту населення від впливу електромагнітних випромінювань [Текст]. – К. : МОЗ

України, 1996. – 53 с.

20. ГОСТ 12.1.045-84 ССБТ. Электростатические поля. Допустимые уровни на рабочих местах и требования к проведению контроля [Текст]. – М. : Стандартинформ, 2006. – 11 с.

21. ДСТУ 2272:2006 Пожежна безпека [Текст]. – К. : Держспоживстандарт, 2007. – 74 с.

22. ДСТУ ІЕС 60884-1:2007 Вилки та розетки побутової та аналогічної призначеності [Текст]. – К. : Держспоживстандарт, 2009. – 91 с.

23. Жидецький, В. Ц. Основи охорони праці [Текст] : підручник / В. Ц. Жидецький. – 5-те вид., доп. – К. : Знання, 2014. – 373 с.

24. Правила улаштування електроустановок [Текст] : Нормативне виробничо-практичне видання : вид. 5-те, перероблене й доповнене : затв. М-вом енергетики та вугільної промисловості України 20.06.14 : введення в дію з 20.11.14. – Х. : Міненерговугілля України, 2014. – 793 с.

25. Основи охорони праці [Текст] : підручник / В. В. Березуцький, Т. С. Бондаренко, Г. Г. Валенко та ін. ; за ред. проф. В. В. Березуцького. – Х. : Факт, 2005. – 327 с.

26. ДСН 3.3.6.037-99 Санітарні норми виробничого шуму, ультразвуку та інфразвуку. – К. : МОЗ України, 1999. – 24 с.

27. ДБН В.2.-67:2013 Опалення, вентиляція та кондиціонування [Текст]. – Київ : Мінрегіон України, 2013. – 139 с.

28. Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою [Текст] : ДСТУ Б В.1.1-36:2016. – На заміну НАПБ Б.03.002-2007 ; чинний від 2017-01-01. – К. : Мінрегіонбуд України, 2016. – 66 с. – (Державний Стандарт України)

29. ДБН В.1.1.7:2016 Пожежна безпека об'єктів будівництва [Текст] : Загальні вимоги. – Київ : Міністерство регіонального розвитку, будівництва та житлово-комунального господарства України, 2017. – 47 с.

30. НАПБ Б.03.001-2004 Типові норми належності вогнегасників [Текст]. – Київ : МНС України, 2004. – 12 с.

31. Шоботов, В. М. Цивільна оборона [Текст] : Навчальний посібник / В. М. Шоботов. – Вид. 2-ге, перероб. – К. : Центр навчальної літератури, 2006. – 438 с.

32. Стеблюк, М. І. Цивільна оборона та цивільний захист [Текст] : навч. посіб. для вузів / М. І. Стеблюк. – К. : Знання, 2013. – 487 с.

ДОДАТОК А
Технічне завдання

Вступ

Створення рішення, яке дозволить якісно не тільки керувати процесом дистанційного навчання, але і забезпечувати для нього потужну базу прийняття рішень

A.1 Підстави для розробки

Розробка виконується на основі завдання на дипломну кваліфікаційну роботу магістра, яке було визначене за темою “Дослідження та програмна реалізація методів визначення тривалості навчання” і затверджене наказом № 301 від 3 листопада 2020 р. за Національним університетом «Запорізька політехніка».

A.2 Призначення розробки

Програмне забезпечення призначене для підтримки процесу дистанційного навчання. Розробка повинна дозволити виконувати керувати матеріалами за дисциплінами та процесом навчання загалом, забезпечувати можливість навчання за допомогою програми, прогнозувати тривалість вивчення дисципліни, визначати як особисту підписку на вивчення дисципліни, так і підписування на навчання зі сторони адміністратора.

A.3 Основні вимоги до програми

A.3.1 Вимоги до функціональних характеристик

У результаті програма повинна виконувати наступні функції:

- створення дисципліни для викладання;
- визначення тем за дисципліною;
- визначення набору робіт за дисципліною;

- внесення матеріалів для викладання дисципліни;
- підписування студентів на проходження дисциплін;
- створення тестів за дисципліною;
- проходження тестів за дисципліною;
- обчислення результату проходження тесту;
- перегляд матеріалів за дисципліною;
- перегляд студентів, підписаних на дисципліну;
- перегляд дисциплін, на які підписався студент;
- прогнозування тривалості навчання за дисципліною;
- завантаження виконаних робіт;
- створення коментарів щодо роботи;
- виставлення оцінки за роботу;
- створення облікових записів студентів та викладачів;
- створення груп студентів;
- додавання викладачів для роботи над дисципліною.

A.3.2 Вимоги до надійності

Надійність функціонування програмного забезпечення повинна забезпечуватися одночасно двома факторами:

- надійністю роботи сервера бази даних;
- надійністю роботи безпосередньо самої програми.

Для надійного функціонування безпосередньо самої програми має перевірятися результат виконання операцій і у випадку помилки виводитися повідомлення про це.

A.3.3 Умови експлуатації

Найголовнішою вимогою до організації програмної сторони роботи сервера є встановлення та підтримка на ньому системи керування базами

даних MySQL версії не нижче ніж 5.6. Надання постійного доступу до сервера бази даних є обов'язковою умовою забезпечення експлуатації програми.

A.3.4 Вимоги до складу та параметрів технічних засобів

Мінімальні вимоги до серверу, на якому має забезпечуватися доступ до бази даних:

- двоядерний процесор з частотою 2 ГГц;
- обсяг оперативного запам'ятовуючого пристрою, що дорівнює 1 Гб;
- вільний простір на жорсткому диску, що дорівнює 1 Гб.

A.3.5 Вимоги до транспортування та збереження

Не передбачається транспортування програми фізичним шляхом на фізичних носіях. Для доступу до програми має виконуватися збереження програми на сервері.

A.3.6 Вимоги до програмної документації

Текст програми повинен включати фрагмент програмного коду.

Технічне завдання визначає всі рамкові умови подальшої розробки.

A.3.7 Вимоги до маркування та упакування

Маркуватися програма повинна назвою виконуваної роботи, окрім того допускається за необхідності скорочення вводити назву JustEducationHelper.

А.4 Порядок контролю та приймання

Керівник роботи виконує контроль виконання календарного плану роботи протягом періоду її виконання. Для кожного етапу мають забезпечуватися строки і результати, позначення в календарному плані. Остаточні результати мають бути захищені перед екзаменаційною комісією.

ДОДАТОК Б
Текст програми

```

#текст файла mysqldatawork.py
import mysql.connector

JEAHdb = mysql.connector.connect(
    host="jeh.ser",
    user="root",
    passwd="...",
    database="DEHdb"
)

def coursetable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.Course(
        crid INT NOT NULL AUTO_INCREMENT,
        crname VARCHAR(100) NOT NULL,
        crterm INT,
        crform INT,
        PRIMARY KEY ( crid )
    ); """)
    JEAHdb.commit()

def getcoursebyid(id):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT * FROM DEHdb.Course WHERE crid =
"+str(id))
    data = jeahcrs.fetchone ()
    JEAHdb.commit()
    return data

def getcoursebyname(name):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT crid FROM DEHdb.Course WHERE crname =
"+str(name))
    data = jeahcrs.fetchall ()
    JEAHdb.commit()
    return data

def newcourse(name, term, form):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.Course (crname, crterm, crform)
VALUES (%s, %s, %s)"

```

```

        zm = (name, term, form)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

def courseedit(name, term, form, cid):
    jeahcrs = JEAHdb.cursor()
    SQLq = "UPDATE DEHdb.Course SET crname = %s, crform = %s,
crterm = %s WHERE crid = %s"
    zm = (name, form, term, cid)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def courseremove(curs):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("DELETE FROM DEHdb.Course WHERE crid =
"+str(curs))
    JEAHdb.commit()

def getcourses():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT * FROM DEHdb.Course")
    courses = jeahcrs.fetchall()
    JEAHdb.commit()
    return courses

def teachertable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.Teacher(
        tid INT NOT NULL AUTO_INCREMENT,
        tsurname VARCHAR(100) NOT NULL,
        tname VARCHAR(100) NOT NULL,
        tpatr VARCHAR(100) NOT NULL,
        tdepart VARCHAR(50) NOT NULL,
        tscideg VARCHAR(10) NOT NULL,
        tacstat VARCHAR(10) NOT NULL,
        tposition VARCHAR(50) NOT NULL,
        tuser INT,
        PRIMARY KEY ( tid ),
        FOREIGN KEY (tuser) REFERENCES User (id)
    ); """)
    JEAHdb.commit()

def authorize(login, password):

```

```

        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("SELECT usid FROM DEHdb.User WHERE uslogin =
"+str(login)+ " AND uspass= "+str(password))
        data = jeahcrs.fetchone ()
        JEAHdb.commit()
        return data

def getuserbylogin(login):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT usid FROM DEHdb.User WHERE uslogin =
"+str(login))
    data = jeahcrs.fetchall ()
    JEAHdb.commit()
    return data

def newteacher(surname, name, patr, depart, scideg, acstat,
position, user):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.Teacher (tsurname, tname, tpatr,
tdepart, tscideg, tacstat, tposition, tuser) VALUES (%s, %s, %s, %s, %s,
%s, %s, %s)"
    zm = (surname, name, patr, depart, scideg, acstat, position,
user)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def teacheredit(tnum, surname, name, patr, depart, scideg, acstat,
position, user):
    jeahcrs = JEAHdb.cursor()
    SQLq = "UPDATE DEHdb.Teacher SET tsurname = %s, tname = %s,
tpatr = %s, tdepart = %s, tscideg = %s, tacstat = %s, tposition = %s,
tuser = %s WHERE tid = %s"
    zm = (surname, name, patr, depart, scideg, acstat, position,
user, tnum)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def studentcoursetable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS
DEHdb.StudentCourse (
        scid INT NOT NULL AUTO_INCREMENT,
        scstud INT NOT NULL,

```

```

        sccourse INT NOT NULL,
        scscore INT,
        scbdate DATETIME,
        scedate DATETIME,
        scptime DATETIME,
        PRIMARY KEY ( scid ),
        FOREIGN KEY (scstud) REFERENCES Student ( stid ),
        FOREIGN KEY (sccourse) REFERENCES Course ( crid )
    ); """
    JEAHdb.commit()

def newcourseteacher (course, teacher, position):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.CourseTeacher (ctcourse, ctteacher,
ctposition) VALUES (%s, %s, %s)"
    zm = (stud, course, score, bdate, edate, ptime)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def courseteacheredit(course, teacher, position, ct):
    jeahcrs = JEAHdb.cursor()
    SQLq = "UPDATE DEHdb.CourseTeacher SET ctcourse = %s, ctteacher
= %s, ctposition = %s WHERE ctid = %s"
    zm = (course, teacher, position, ct)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def topictable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.Topic (
        toid INT NOT NULL AUTO_INCREMENT,
        toname VARCHAR (255),
        toperiod INT,
        tocourse INT NOT NULL,
        PRIMARY KEY ( toid ),
        FOREIGN KEY (tocourse) REFERENCES Course (crid)
    ); """)
    JEAHdb.commit()

def gettopicsbycourse(id):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT * FROM DEHdb.Topic WHERE tocourse =
"+str(id))

```

```

        data = jeahcrs.fetchone ()
        JEAHdb.commit()
        return data

def materialstable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.Materials (
        mid INT NOT NULL AUTO_INCREMENT,
        mtopic INT,
        mname VARCHAR (100) NOT NULL,
        msite VARCHAR (255),
        mfile VARCHAR (255),
        mdesc VARCHAR (255),
        PRIMARY KEY ( mid ),
        FOREIGN KEY (mtopic) REFERENCES Topic (toid)
    ); """)
    JEAHdb.commit()

def getmaterialsbytopic(id):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT * FROM DEHdb.Materials WHERE mtopic =
"+str(id))
    data = jeahcrs.fetchone ()
    JEAHdb.commit()
    return data

def materialsremove(m):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("DELETE FROM DEHdb.Materials WHERE mid =
"+str(m))
    JEAHdb.commit()

def questiontable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.Question (
        qid INT NOT NULL AUTO_INCREMENT,
        qtext TEXT,
        qcontrol INT,
        qresult FLOAT,
        qtype INT,
        PRIMARY KEY ( qid ),
        FOREIGN KEY (qcontrol) REFERENCES Control (coid)
    ); """)

```

```

JEAHdb.commit()

def studentworkscore(id, score):
    jeahcrs = JEAHdb.cursor()
    SQLq = "UPDATE DEHdb.StudentWorks SET swscore = %s WHERE (swid
= %s)"
    zm = (score, id)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def studentworksedit(file, desc, score, courw, stud, swi):
    jeahcrs = JEAHdb.cursor()
    SQLq = "UPDATE DEHdb.StudentWorks SET swfile = %s, swdesc = %s,
swscore = %s, swcourw = %s, swstud = %s WHERE swi = %s"
    zm = (file, desc, score, courw, stud, swi)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def studentcontroltable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS
DEHdb.StudentControl (
        scoid INT NOT NULL AUTO_INCREMENT,
        scocontrol INT,
        scostudent INT,
        scoscore FLOAT,
        scodate DATE,
        PRIMARY KEY ( scoid ),
        FOREIGN KEY (scostudent) REFERENCES Student ( stid )
        FOREIGN KEY (scocontrol) REFERENCES Control ( coid )
    ); """)
    JEAHdb.commit()

def newstudentcontrol(control, student, score, date):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.StudentControl (scocontrol,
scostudent, scoscore, scodate) VALUES (%s, %s, %s, %s)"
    zm = (control, student, score, date)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def create():

```

```

jeahcrs = JEAHdb.cursor()
SQLq = "CREATE DATABASEIF NOT EXISTS DEHdb"
jeahcrs.execute(SQLq)
JEAHdb.commit()

coursetable()
usertable()
grouptable()
teachertable()
studenttable()
studentcoursetable()
courseteachertable()
topictable()
controltable()
materialstable()
questiontable()
answertable()
studentanswertable()
courseworkstable()
studentworkstable()
studentcontroltable()
messagestable()

def newmessages (messid, studwid, coursteach, text):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.Messages (memessid, mestudwid,
mecoursteach, metext) VALUES (%s, %s, %s, %s)"
    zm = (control, student, score, date)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def messagesedit(messid, studwid, coursteach, text, me):
    jeahcrs = JEAHdb.cursor()
    SQLq = "UPDATE DEHdb.Messages SET memessid = %s, mestudwid =
%s, mecoursteach = %s, metext = %s WHERE meid = %s"
    zm = (messid, studwid, coursteach, text, me)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def messagesremove(me):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("DELETE FROM DEHdb.Messages WHERE meid =
"+str(me))

```

```

JEAHdb.commit()

def messagestable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.Messages (
        meid INT NOT NULL AUTO_INCREMENT,
        memessid INT,
        mestudwid INT,
        mecoursteach INT,
        metext TEXT,
        PRIMARY KEY (meid),
        FOREIGN KEY (mestudwid) REFERENCES StudentWorks ( swid )
        FOREIGN KEY (mecoursteach) REFERENCES CourseTeacher ( ctid
    )

    ); """)
    JEAHdb.commit()

def newuser(login, pass):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.User (uslogin, uspass) VALUES (%s,
%s)"

    zm = (login, pass)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def useredit(login, pass, us):
    jeahcrs = JEAHdb.cursor()
    SQLq = "UPDATE DEHdb.User SET uslogin = %s, uspass = %s WHERE
usid = %s"

    zm = (login, pass, us)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def newgroup (name, year):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.Group (grname, gryear) VALUES (%s,
%s)"

    zm = (name, year)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def groupedit(name, year, group):
    jeahcrs = JEAHdb.cursor()

```

```

        SQLq = "UPDATE DEHdb.Group SET grname = %s, gryear = %s WHERE
grid = %s"
        zm = (name, year, group)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

def getteacherbyid(id):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT * FROM DEHdb.Teacher WHERE tid =
"+str(id))
    data = jeahcrs.fetchone ()
    JEAHdb.commit()
    return data

def getteacherbyname(surname, name, patr):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT tid FROM DEHdb.Teacher WHERE tsurname =
"+str(surname) AND tname = "+str(name) AND tpatr = "+str(patr))
    data = jeahcrs.fetchall ()
    JEAHdb.commit()
    return data

def studenttable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.Student(
        stid INT NOT NULL AUTO_INCREMENT,
        stsurname VARCHAR(100) NOT NULL,
        stname VARCHAR(100) NOT NULL,
        stpatr VARCHAR(100) NOT NULL,
        stbirth DATETIME NOT NULL,
        stuser INT NOT NULL,
        stgroup VARCHAR(10) NOT NULL,
        stvar INT NOT NULL,
        stuser INT,
        PRIMARY KEY ( stid ),
        FOREIGN KEY (stuser) REFERENCES User ( usid ),
        FOREIGN KEY (stgroup) REFERENCES Group ( grid )
    ); """)
    JEAHdb.commit()

def newstudenttogroup(surname, name, patr, birth, user, group, var,
user):
    jeahcrs = JEAHdb.cursor()

```

```

        jeahcrs.execute("SELECT grid FROM DEHdb.Group WHERE grname =
\""+group+"\"")
        jgrid = jeahcrs.fetchall()
        SQLq = "INSERT INTO DEHdb.Student (stsurname, stname, stpatr,
stbirth, stuser, stgroup, stvar, stuser) VALUES (%s, %s, %s, %s, %s, %s,
%s, %s)"
        zm = (surname, name, patr, birth, user, jgrid[0][0], var, user)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

def studenttogroupedit (surname, name, patr, birth, user, group,
var, user,sid):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT grid FROM DEHdb.Group WHERE grname =
\""+group+"\"")
    jgrid = jeahcrs.fetchall()
    SQLq = "UPDATE DEHdb.Student SET stsurname = %s, stname = %s,
stpatr = %s, stbirth = %s, stuser = %s, stgroup = %s, stvar = %s, stuser
= %s WHERE (stid = %s)"
    zm = (surname, name, patr, birth, user, jgrid[0][0], var,
user,sid)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def courseteachertable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS
DEHdb.CourseTeacher (
        ctid INT NOT NULL AUTO_INCREMENT,
        ctcourse INT NOT NULL,
        ctteacher INT NOT NULL,
        ctposition INT,
        PRIMARY KEY ( ctid ),
        FOREIGN KEY (ctcourse) REFERENCES Course ( crid ),
        FOREIGN KEY (ctteacher) REFERENCES Teacher ( tid )
    ); """)
    JEAHdb.commit()

def getcourseteachers(cid, tid):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT * FROM DEHdb.CourseTeacher WHERE
ctteacher = "+str(tid)+ " AND ctcourse = "+str(cid))
    data = jeahcrs.fetchone ()

```

```

JEAHdb.commit()
return data

def controtable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.Control (
        coid INT NOT NULL AUTO_INCREMENT,
        cotopic INT,
        coname VARCHAR (100) NOT NULL,
        coform INT,
        cobdate DATETIME,
        coedate DATETIME,
        coduration DATETIME,
        PRIMARY KEY ( coid ),
        FOREIGN KEY (cotopic) REFERENCES Topic (cotopic)
    ); """)
    JEAHdb.commit()

def newcontrol (topic, name, form, bdate, edate, duration):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.Control (cotopic, coname, coform,
cobdate, coedate, coduration) VALUES (%s, %s, %s, %s, %s, %s)"
    zm = (topic, name, form, bdate, edate, duration)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def questionremove(q):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("DELETE FROM DEHdb.Question WHERE qid =
"+str(q))
    JEAHdb.commit()

def answertable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.Answer (
        aid INT NOT NULL AUTO_INCREMENT,
        atext TEXT,
        apos INT,
        aresult FLOAT,
        aquestion INT,
        PRIMARY KEY ( aid ),
        FOREIGN KEY (aquestion) REFERENCES Question (qid)
    ); """)

```

```

JEAHdb.commit()

def studentansweredit(question, stud, name, score, sa):
    jeahcrs = JEAHdb.cursor()
    SQLq = "UPDATE DEHdb.StudentAnswer SET saquestion = %s, sastud
= %s, saname = %s, sascore = %s WHERE said = %s"
    zm = (question, stud, name, score, sa)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def studentanswerremove(sa):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("DELETE FROM DEHdb.StudentAnswer WHERE said =
"+str(sa))
    JEAHdb.commit()

def studentworkstable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS
DEHdb.StudentWorks (
        swid INT NOT NULL AUTO_INCREMENT,
        swfile VARCHAR(255) NOT NULL,
        swdesc TEXT,
        swscore FLOAT,
        swcourw INT,
        swstud INT,
        PRIMARY KEY ( cwid ),
        FOREIGN KEY (swstud) REFERENCES Student ( stid )
        FOREIGN KEY (swcourw) REFERENCES CourseWorks ( cwid )
    ); """)
    JEAHdb.commit()

def getstudentworks(id):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT * FROM DEHdb.StudentWorks WHERE
StudentWorks. swstud = "+str(id))
    data = jeahcrs.fetchone ()
    JEAHdb.commit()
    return data

def newstudentworks (file, desc, score, courw, stud):
    jeahcrs = JEAHdb.cursor()

```

```

        SQLq = "INSERT INTO DEHdb.StudentWorks (swfile, swdesc,
swscore, swcourw, swstud) VALUES (%s, %s, %s, %s, %s)"
        zm = (file, desc, score, courw, stud)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

```

```

def usertable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.User(
        usid INT NOT NULL AUTO_INCREMENT,
        uslogin VARCHAR(100) NOT NULL,
        uspass VARCHAR(100) NOT NULL,
        PRIMARY KEY ( usid )
    ); """)
    JEAHdb.commit()

```

```

def userremove(us):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("DELETE FROM DEHdb.User WHERE usid = "+str(us))
    JEAHdb.commit()

```

```

def grouptable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.Group(
        grid INT NOT NULL AUTO_INCREMENT,
        grname VARCHAR(10) NOT NULL,
        gryear INT NOT NULL,
        PRIMARY KEY ( grid )
    ); """)
    JEAHdb.commit()

```

```

def grouppremove(group):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("DELETE FROM DEHdb.Group WHERE grid =
"+str(group))
    JEAHdb.commit()

```

```

def getteachers():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT * FROM DEHdb.Teacher)

```

```

        teachers = jeahcrs.fetchall()
        JEAHdb.commit()
        return teachers

def getstudents ():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT * FROM DEHdb.Student")
    students = jeahcrs.fetchall()
    JEAHdb.commit()
    return students

def getstudentbyid(id):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("SELECT * FROM DEHdb.Student WHERE stid =
"+str(id))
    data = jeahcrs.fetchone ()
    JEAHdb.commit()
    return data

def teacherremove(teach):
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("DELETE FROM DEHdb.Teacher WHERE tid =
"+str(teach))
    JEAHdb.commit()

def newstudent (surname, name, patr, birth, user, group, var,
user):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.Student (stsurname, stname, stpatr,
stbirth, stuser, stgroup, stvar, stuser) VALUES (%s, %s, %s, %s, %s, %s,
%s, %s)"
    zm = (surname, name, patr, birth, user, group, var, user)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def studentedit(surname, name, patr, birth, user, group, var, user,
stud):
    jeahcrs = JEAHdb.cursor()
    SQLq = "UPDATE DEHdb.Student SET stsurname = %s, stname = %s,
stpatr = %s, stbirth = %s, stuser = %s, stgroup = %s, stvar = %s, stuser
= %s WHERE stid = %s"
    zm = (surname, name, patr, birth, user, group, var, user, stud)
    jeahcrs.execute(SQLq, zm)

```

```

        JEAHdb.commit()

    def studentremove(stud):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("DELETE FROM DEHdb.Student WHERE stid =
"+str(stud))
        JEAHdb.commit()

    def getstudentbyname(surname, name, patr):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("SELECT stid FROM DEHdb.Student WHERE stsurname
= "+str(surname) AND stname = "+str(name) AND stpatr = "+str(patr))
        data = jeahcrs.fetchall ()
        JEAHdb.commit()
        return data

    def getstudentcourses(id):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("SELECT * FROM DEHdb.StudentCourse WHERE scstud
= "+str(id))
        data = jeahcrs.fetchone ()
        JEAHdb.commit()
        return data

    def newstudentcourse (stud, course, score, bdate, edate, ptime):
        jeahcrs = JEAHdb.cursor()
        SQLq = "INSERT INTO DEHdb.StudentCourse (scstud, sccourse,
scscore, scbdate, scedate, scptime) VALUES (%s, %s, %s, %s, %s, %s)"
        zm = (stud, course, score, bdate, edate, ptime)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

    def studentcourseedit(stud, course, score, bdate, edate, ptime,
stc):
        jeahcrs = JEAHdb.cursor()
        SQLq = "UPDATE DEHdb.StudentCourse SET scstud = %s, sccourse =
%s, sccscore = %s, scbdate = %s, scedate = %s, scptime = %s WHERE scid =
%s"
        zm = (stud, course, score, bdate, edate, ptime, stc)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

    def studencoursetremove(sc):

```

```

        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("DELETE FROM DEHdb.StudentCourse WHERE scid =
"+str(sc))
        JEAHdb.commit()

    def newtopic (name, period, course):
        jeahcrs = JEAHdb.cursor()
        SQLq = "INSERT INTO DEHdb.Topic (toname, toperiod, tocourse)
VALUES (%s, %s, %s)"
        zm = name, period, course)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

    def topicedit(name, period, course, topic):
        jeahcrs = JEAHdb.cursor()
        SQLq = "UPDATE DEHdb.Topic SET toname = %s, toperiod = %s,
tocourse = %s WHERE toid = %s"
        zm = (name, period, course, topic)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

    def topicremove(t):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("DELETE FROM DEHdb.Topic WHERE toid = "+str(t))
        JEAHdb.commit()

    def controledit(topic, name, form, bdate, edate, duration, contr):
        jeahcrs = JEAHdb.cursor()
        SQLq = "UPDATE DEHdb.Control SET cotopic = %s, coname = %s,
coform = %s, cobdate = %s, coedate = %s, coduration = %s WHERE coid = %s"
        zm = (topic, name, form, bdate, edate, duration, contr)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

    def controlremove(c):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("DELETE FROM DEHdb.Control WHERE coid =
"+str(c))
        JEAHdb.commit()

    def studentanswertable():
        jeahcrs = JEAHdb.cursor()

```

```

        jeahcrs.execute("""CREATE TABLE IF NOT EXISTS
DEHdb.StudentAnswer (
            said INT NOT NULL AUTO_INCREMENT,
            saquestion INT,
            sastud INT,
            saname INT,
            sascore FLOAT,
            PRIMARY KEY ( aid ),
            FOREIGN KEY (saquestion) REFERENCES Question (qid)
            FOREIGN KEY (sastud) REFERENCES Student ( stid )
        ); """)
        JEAHdb.commit()

def newstudentanswer (question, stud, name, score):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.StudentAnswer (saquestion, sastud,
saname, sascore) VALUES (%s, %s, %s, %s)"
    zm = (question, stud, name, score)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def courseworkstable():
    jeahcrs = JEAHdb.cursor()
    jeahcrs.execute("""CREATE TABLE IF NOT EXISTS DEHdb.CourseWorks
(
        cwid INT NOT NULL AUTO_INCREMENT,
        cwsite VARCHAR(255),
        cwname VARCHAR(255) NOT NULL,
        cwdesc TEXT,
        cwtopic INT,
        cwmater INT,
        cwmaxscore FLOAT,
        cwdeadline DATETIME,
        PRIMARY KEY ( cwid ),
        FOREIGN KEY (cwmater) REFERENCES Materials ( toid )
        FOREIGN KEY (cwtopic) REFERENCES Topic ( stid )
    ); """)
    JEAHdb.commit()

def newcourseworks (site, name, desc, topic, mater, maxscore,
deadline):
    jeahcrs = JEAHdb.cursor()

```

```

        SQLq = "INSERT INTO DEHdb.CourseWorks (cwsite, cwname, cwdesc,
        cwtopic, cwmater, cwmaxscore, cwdeadline) VALUES (%s, %s, %s, %s, %s, %s,
        %s)"

        zm = (site, name, desc, topic, mater, maxscore, deadline)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

    def studentworksremove(swi):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("DELETE FROM DEHdb.StudentWorks WHERE swid =
        "+str(swi))
        JEAHdb.commit()

    def studentcontroledit(control, student, score, date, sco):
        jeahcrs = JEAHdb.cursor()
        SQLq = "UPDATE DEHdb.StudentControl SET scocontrol = %s,
        scostudent = %s, scoscore = %s, scodate = %s WHERE scoid = %s"
        zm = (control, student, score, date, sco)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

    def studentcontrolremove(sco):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("DELETE FROM DEHdb.StudentControl WHERE scoid =
        "+str(sco))
        JEAHdb.commit()

    def courseteacherremove(ct):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("DELETE FROM DEHdb.CourseTeacher WHERE ctid =
        "+str(ct))
        JEAHdb.commit()

    def getmaterialsbydiscipline(id):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("SELECT Materials.name, Materials.msite,
        Materials.mfile, Materials.mdsc FROM DEHdb.Materials INNER JOIN
        DEHdb.Topic ON Topic.tid = Materials.mtopic WHERE Topic.tocourse =
        "+str(id))
        data = jeahcrs.fetchone ()
        JEAHdb.commit()
        return data

```

```

def newmaterials (topic, name, site, file, desc):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.Materials (mtopic, mname, msite,
mfile, mdesc) VALUES (%s, %s, %s, %s, %s)"
    zm = (topic, name, site, file, desc)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def materialsedit(topic, name, site, file, desc, mat):
    jeahcrs = JEAHdb.cursor()
    SQLq = "UPDATE DEHdb.Materials SET mtopic = %s, mname = %s,
msite = %s, mfile = %s, mdesc = %s WHERE mid = %s"
    zm = (topic, name, site, file, desc, mat)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def newquestion (text, control, result, type):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.Question (qtext, qcontrol, qresult,
qtype) VALUES (%s, %s, %s, %s)"
    zm = (text, control, result, type)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def questionedit(text, control, result, type, quest):
    jeahcrs = JEAHdb.cursor()
    SQLq = "UPDATE DEHdb.Question SET qtext = %s, qcontrol = %s,
qresult = %s, qtype WHERE qid = %s"
    zm = (text, control, result, type, quest)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def newanswer (text, pos, result, question):
    jeahcrs = JEAHdb.cursor()
    SQLq = "INSERT INTO DEHdb.Answer (atext, apos, aresult,
aquestion) VALUES (%s, %s, %s, %s)"
    zm = (text, pos, result, question)
    jeahcrs.execute(SQLq, zm)
    JEAHdb.commit()

def answeredit(text, pos, result, question, ans):
    jeahcrs = JEAHdb.cursor()

```

```

        SQLq = "UPDATE DEHdb.Answer SET atext = %s, apos = %s, aresult
= %s, aquestion = %s, qtype = %s WHERE aid = %s"
        zm = (text, pos, result, question, ans)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

    def answerremove(a):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("DELETE FROM DEHdb.Question WHERE aid =
"+str(a))
        JEAHdb.commit()

    def courseworksedit(site, name, desc, topic, mater, maxscore,
deadline, cwi):
        jeahcrs = JEAHdb.cursor()
        SQLq = "UPDATE DEHdb.CourseWorks SET cwsite = %s, cwname = %s,
cwdesc = %s, cwtopic = %s, cwmater = %s, cwmaxscore = %s, cwdeadline = %s
WHERE cwid = %s"
        zm = (site, name, desc, topic, mater, maxscore, deadline, cwi)
        jeahcrs.execute(SQLq, zm)
        JEAHdb.commit()

    def courseworksremove(cwi):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("DELETE FROM DEHdb.CourseWorks WHERE cwid =
"+str(cwi))
        JEAHdb.commit()

    def getmessstudentworks(id):
        jeahcrs = JEAHdb.cursor()
        jeahcrs.execute("SELECT * FROM DEHdb.Messages WHERE Messages.
mestudwid = "+str(id))
        data = jeahcrs.fetchone ()
        JEAHdb.commit()
        return data

```

ДОДАТОК В
Слайди презентації



Рисунок В.1 – Слайд 1



Рисунок В.2 – Слайд 2

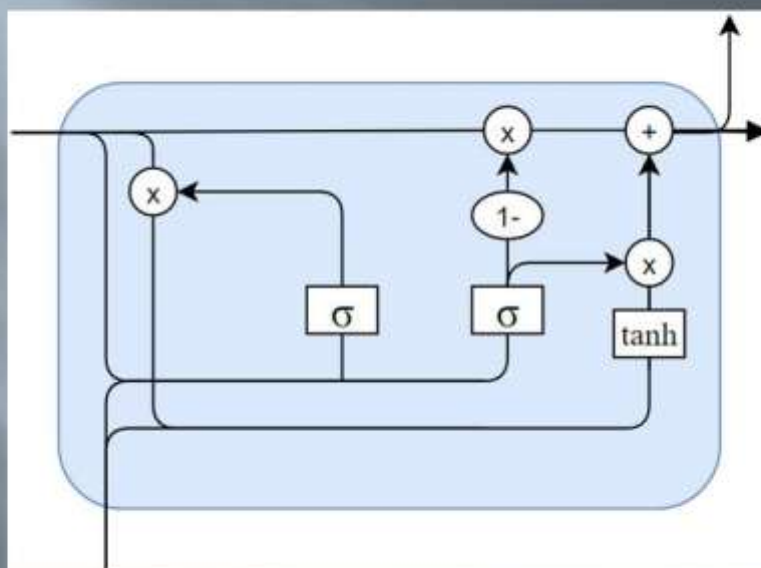
Завдання

- ✓ дослідження методів визначення тривалості навчання;
- ✓ розробка технічного завдання;
- ✓ вибір засобів розробки;
- ✓ проєктування структури програмного забезпечення;
- ✓ розробка програмного забезпечення підтримки дистанційного навчання;
- ✓ тестування та відлагодження програми;
- ✓ економічне обґрунтування доцільності розробки.

3

Рисунок В.3 – Слайд 3

Архітектура GRU



4

Рисунок В.4 – Слайд 4

Метод визначення тривалості навчання

На першому етапі має бути сформовано дані про результати навчання студентів. Для цього для кожного студента має бути визначено тривалість навчання за кожною дисципліною. На другому етапі має бути сформовано масив даних про характеристики кожної дисципліни.

На третьому етапі відбувається формування вибірки даних. Для цього має бути впорядковано всі вивчені кожним студентом дисципліни. Кожен екземпляр вибірки даних представляє собою набір значень у вигляді множини:

$$\langle C_l^i, C_s^i, C_t^i, C_m^i, C_c^i, C_{tn}^i, C_{lt}^i, T_{i-1}^s, T_{i-2}^s, \dots, T_{i-8}^s \rangle,$$

де C_l^i – кількість лабораторних та практичних робіт;

C_s^i – кількість самостійних робіт;

C_t^i – кількість тестів для перевірки знань студентів;

C_m^i – кількість матеріалів за дисципліною;

C_c^i – форма контролю знань;

C_{tn}^i – кількість тем, що розглядаються в дисципліні;

C_{lt}^i – середній час вивчення дисципліни студентом.

На четвертому етапі виконується створення моделі прогнозування тривалості навчання.

На п'ятому етапі забезпечується навчання створеної описаним чином нейронної мережі.

На шостому етапі виконується прогнозування тривалості навчання студентом за дисципліною для заданого набору значень параметрів

5

Рисунок В.5 – Слайд 5

Вибір мови програмування

Критерій	Python	Java
Головна парадигма	Об'єктно-орієтована	Об'єктно-орієтована
Специфікація	Загального призначення	Загального призначення
Принцип виконання	Інтерпретований	Компільований
Вид типізації	Динамічний	Статичний
Засоби штучного інтелекту	Потужні, гнучкі, велика підтримка	Наявні, менша гнучкість
Процес написання коду	Простий	Складний

6

Рисунок В.6 – Слайд 6



Схема бази даних



Рисунок В.8 – Слайд 8

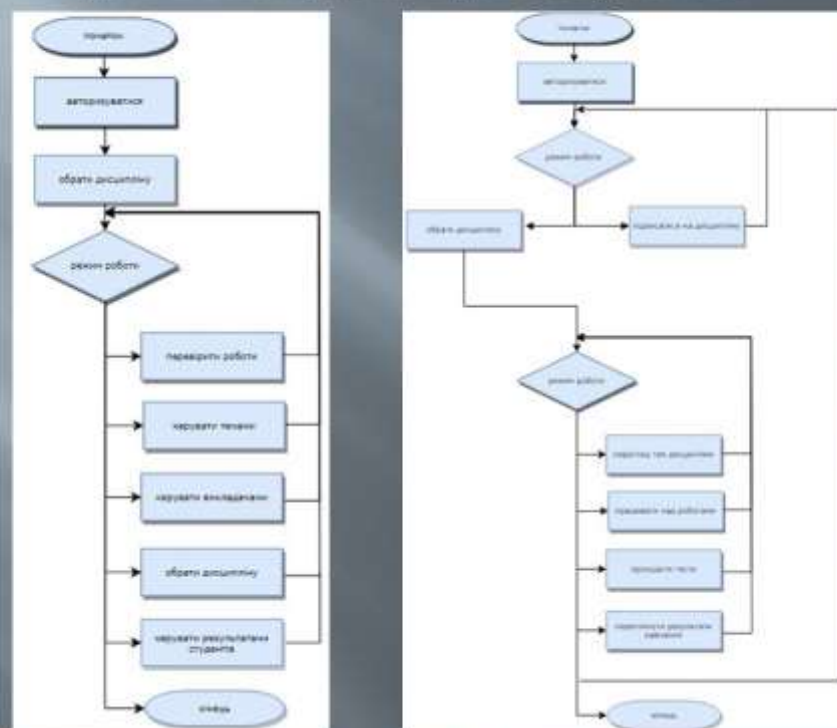
Структурна схема програмної системи



9

Рисунок В.9 – Слайд 9

Функціональна схема програми для викладача і для студента



10

Рисунок В.10 – Слайд 10

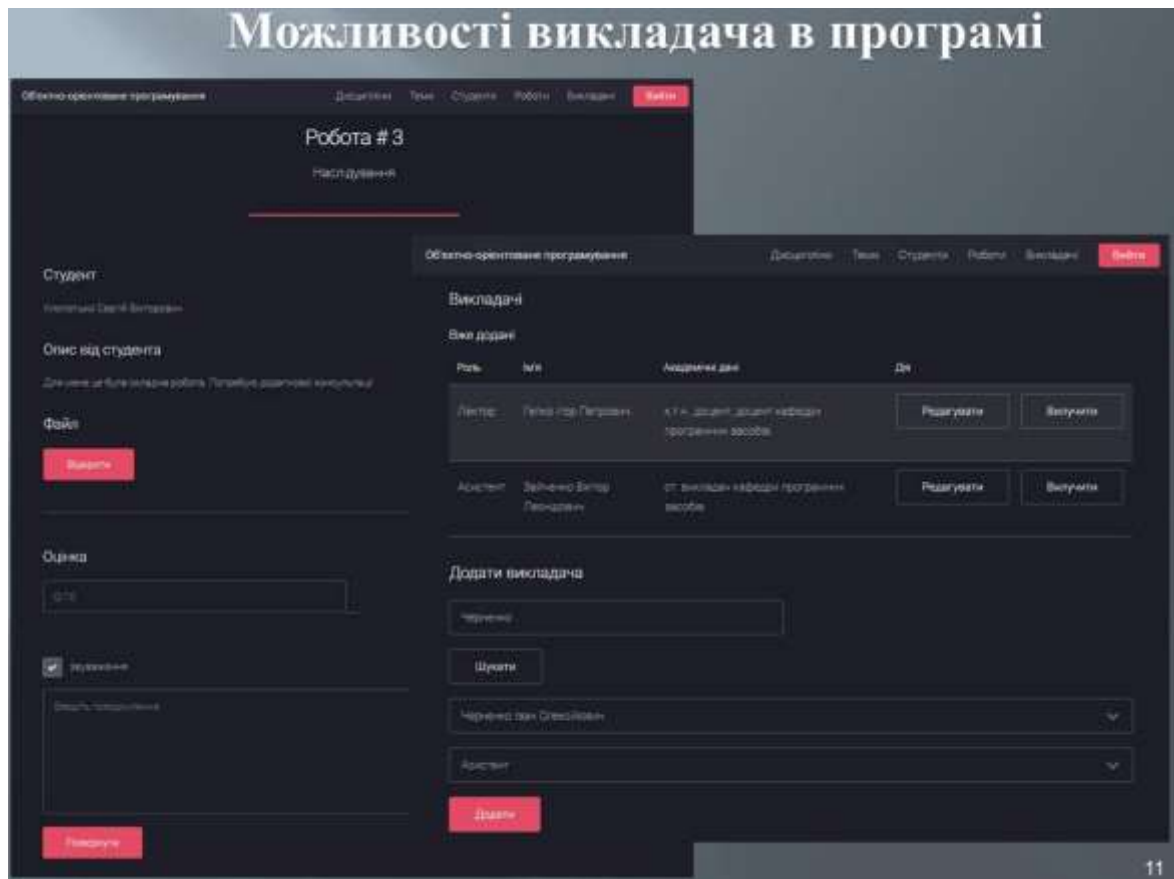


Рисунок В.11 – Слайд 11

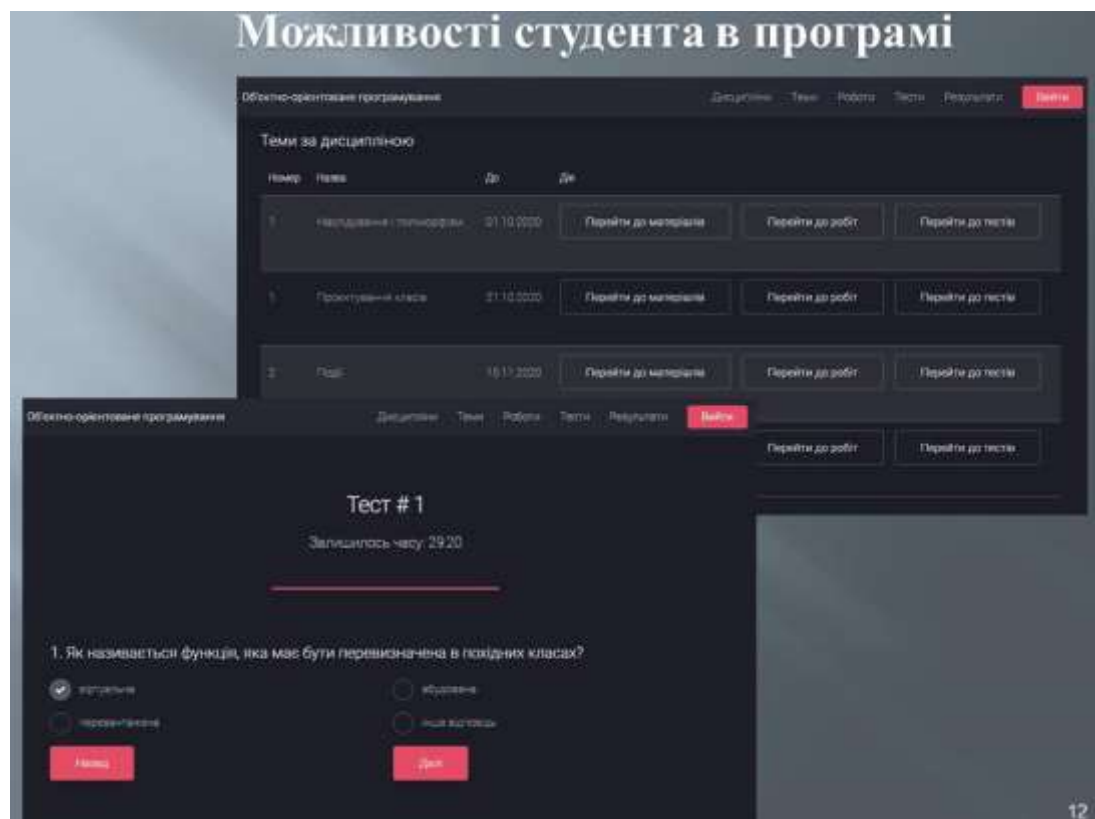


Рисунок В.12 – Слайд 12

Визначення тривалості навчання

Об'єктно-орієнтоване програмування
Дистанційна Тести Студенти Роботи Викладачі **Війти**

Результати навчання

Оберіть групу

КНТ-110

Номер	Ім'я	Стан	Оцінювальності	Оцінка	Закінчення	Дія
1	Артюхов Максим Ігоревич	Навчається	1/5	Нова	24.04.2021	Перейти
2	Бакін Роман Сергійович	Навчається	2/5	Нова	17.03.2021	Перейти
3	Вітернюк Костянтин Сергійович	Навчається	5/5	Нова	25.12.2020	Перейти
4	Жискін Ігор Захарович	Навчається	4/5	Нова	29.12.2020	Перейти
5	Зарин Андрій Іванович	Навчається	5/5	Нова	17.12.2020	Перейти
6	Петренко Руслан Анатолійович	Навчається	1/5	Нова	12.03.2021	Перейти

13

Рисунок В.13 – Слайд 13

Висновки

Розроблено програмне забезпечення, яке надає можливість виконувати створення дисципліни для викладання, визначення тем, набору робіт, тестів і матеріалів для викладання дисципліни, підписування студентів на проходження дисциплін, проходження тестів за дисципліною, перегляду матеріалів за дисципліною, студентів, підписаних на дисципліну, дисциплін, на які підписався студент, прогнозування тривалості навчання за дисципліною, завантаження виконаних робіт, створення коментарів щодо роботи, виставлення оцінки за роботу.

Наукова новизна дипломної роботи полягає в розробленому методі визначення тривалості навчання, що виконує прогнозування фактичної тривалості вивчення дисципліни студентом на основі тривалості вивчення попередніх дисциплін ним та параметрів, що визначають нову дисципліну, на основі використання моделі нейронної мережі GRU, що дозволяє виконувати прийняття рішень про організацію роботи над вивченням дисципліни.

14

Рисунок В.14 – Слайд 14