

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)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАСОБІВ
ІНТЕГРАЦІЇ КОНТЕНТУ НАВЧАЛЬНИХ КУРСІВ ОНЛАЙН
ПЛАТФОРМ

RESEARCH AND SOFTWARE IMPLEMENTATION OF TOOLS FOR
INTEGRATING THE CONTENT OF EDUCATIONAL COURSES ON ONLINE
PLATFORMS

Виконав: студент(ка) 2 курсу, групи КНТ-129м
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Сиротенко І.О.

(прізвище та ініціали)

Керівник Табунщик Г. В.

(прізвище та ініціали)

Рецензент Шитікова О.В.

(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 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
“ ” 20 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА

Сиротенка Ігоря Олексійовича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація засобів інтеграції контенту навчальних курсів онлайн платформ. Research and Software Implementation of Tools for Integrating the Content of Educational Courses on Online Platforms

керівник проєкту (роботи) Табунщик Галина Володимирівна, професор, к.т.н.,
(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) 14 грудня 2020 року

3. Вихідні дані до проєкту (роботи) користувачі, навчальні курси, параметри конфігурації застосунку

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз проблеми та постановка завдань дослідження. 2. Розробка архітектури програми. 3. Опис програми. 4. Керівництво програміста. 5. Експлуатація, тестування та експериментальне дослідження програми.

6. Економіко-організаційна частина. 7. Охорона праці та безпека в надзвичайних ситуаціях.

8. Додатки

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	Табунщик Г. В., професор		
Організаціо-економічна частина	Пожуєва Т.О., професор		
Охорона праці та цивільна безпека	Коробко О.В., ст. викладач		
Нормоконтролер	Камінська Ж.К. , асистент		

7. Дата видачі завдання “4” вересня 2020 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2–3 тижні	Розділ 1
3	Розробка та удосконалення методів, моделей й алгоритмів вирішення задачі.	4–5 тижні	Розділ 2
4	Розробка архітектури програми.	6 тиждень	Розділ 3
5	Розробка програми.	7–8 тижні	Розділ 4
6	Тестування та експериментальне дослідження програмного забезпечення.	9 тиждень	Розділ 5
7	Розробка організаційно-економічної частини	10 тиждень	Розділ 6
8	Розробка заходів з охорони праці та цивільної безпеки.	11 тиждень	Розділ 7
9	Оформлення пояснювальної записки та документів до неї.	12-13 тиждень	Додатки
10	Нормоконтроль та рецензування.	14–15 тижні	
11	Захист роботи.	16 тиждень	

Студент

(підпис)

Сиротенко І.О.

(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис)

Табунщик Г.В.

(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломного проєкту магістра: 109 с., 37 рис., 12 табл., 2 додатки, 47 джерел.

СИСТЕМА ДИСТАНЦІЙНОГО НАВЧАННЯ, LMS, SCORM, LTI, DOCKER, MOODLE, SAML, NODEJS, TYPESCRIPT, EXPRESS, MOUSTACHE.

Об'єкт дослідження – засоби інтеграції навчального контенту до систем дистанційного навчання.

Предмет дослідження – системи дистанційного навчання, навчальні матеріали та розширення до них.

Мета роботи – дослідження способів інтеграції навчального контенту та розширень до систем дистанційного навчання, розробка власного розширення для додавання навчального контенту та інструментів до навчальних курсів.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування, платформа NodeJS, мови програмування Javascript, TypeScript, персональний комп'ютер з процесором Intel Core i7-10700k, операційна система Ubuntu та Windows 10, браузер Google Chrome, сучасні інструменти розробки.

Результати. Створено серверний застосунок, що працює за LTI протоколом для додавання власного навчального контенту та інструментів до навчальних курсів систем дистанційного навчання.

Висновки. В результаті проведених досліджень та практичної роботи було розроблено програмний застосунок для розширення функціоналу та контентної бази систем дистанційного навчання. Було проаналізовано способи інтеграції та розширення навчальних курсів систем дистанційного навчання.

Галузь використання – системи дистанційного навчання, онлайн освіта та навчання.

ABSTRACT

Master's thesis: 109 pages, 37 figures, 12 tables, 2 appendixes, 47 sources.

LEARNING MANAGEMENT SYSTEM, LMS, SCORM, LTI, DOCKER, MOODLE, SAML, NODEJS, TYPESCRIPT, EXPRESS, MOUSTACHE.

The object of research is solutions for educational content integration into learning management systems.

The subject of research – learning management systems, educational materials and extensions for them.

The purpose of the work is to study ways of educational content integration and extensions for learning management systems, to develop own extension for educational content and tools addition to learning courses.

Materials, methods and technical means: object-oriented programming, NodeJS platform, Javascript and TypeScript programming language, personal computer with Intel Core i7-10700k processor, Ubuntu and Windows 10 operating system, Google Chrome browser.

Results. A server application based on the LTI protocol has been created to provide ability of own educational content and tools addition to learning courses of learning management systems.

Conclusions. As a result of research and practical work, a software application was developed to expand functionality and content base of learning management systems. Ways of courses content integration and extension learning management systems were analyzed.

Field of use – learning management systems, online education and learning.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ.....	9
1 Аналіз проблеми та постановка завдань дослідження	12
1.1 Поняття системи дистанційного навчання.....	12
1.2 Концепція системи дистанційного навчання	12
1.3 Вимоги до системи дистанційного навчання.....	13
1.4 Огляд популярних систем дистанційного навчання	16
1.5 Огляд систем масових відкритих онлайн-курсів.....	25
1.6 Огляд протоколів або стандартів для повторного використання навчальних матеріалів	27
1.7 Висновки до розділу 1	33
2 Розробка архітектури програми.....	35
2.1 Вибір системи дистанційного навчання для тестування застосунку	35
2.2 Розгортання системи дистанційного навчання та її вимоги.....	36
2.3 Вибір мови програмування для реалізації розширення	38
2.4 Вибір технологій для реалізації розширення	39
2.5 Висновки за розділом 2	41
3 Опис програми	42
3.1 Функціональні можливості розробленого застосунку	42
3.2 Додавання застосунку до системи	42
3.3 Алгоритм роботи застосунку	45
3.4 Головні модулі LTI сервера	47
3.5 Висновки за розділом 3	49
4 Керівництво програміста	50
4.1 Призначення та умови застосування програми	50
4.2 Характеристики програми.....	50
4.3 Звернення до програми.....	52
4.4 Вхідні та вихідні дані	53
4.5 Повідомлення та обробка помилкових ситуацій	53

5	Експлуатація, тестування та експериментальне дослідження програми	54
5.1	Призначення програми	54
5.2	Умови виконання програми	54
5.3	Виконання програми	55
5.4	Повідомлення оператору	59
6	Економіко-організаційна частина	60
6.1	Планування дослідження та розробки програмного продукту	61
6.2	Визначення витрат на розробку	64
6.3	Розрахунок кошторисної вартості робіт	71
6.4	Оцінка економічної ефективності	72
7	Охорона праці та безпека в надзвичайних ситуаціях	73
7.1	Аналіз потенціальних небезпек	73
7.2	Заходи із забезпечення безпеки	75
7.3	Заходи з забезпечення виробничої санітарії та гігієни праці	78
7.4	Заходи безпеки у надзвичайних ситуаціях	82
7.5	Висновки з розділу	87
	Висновки	88
	Перелік джерел посилання	89
	Додаток А Текст програми	94
	Додаток Б Слайди презентації	103

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API - Application Programming Interface;

CAL - Computer Aided Learning;

FTP – File Transfer Protocol;

HTTP - Hypertext Transfer Protocol;

IP – Internet Protocol;

JSON - JavaScript Object Notation;

LMS – Learning Management System;

LTI - Learning Tools Interoperability;

SAML - Security Assertion Markup Language;

SCORM - Sharable Content Object Reference Model;

URL - Uniform Resource Locator;

XML - Extensible Markup Language;

СКБД – Система керування базами даних;

МВОК - Масовий відкритий онлайн-курс.

ВСТУП

Актуальність теми. На сьогоднішній день системи дистанційного навчання та онлайн освіта вже не є чимось новим та експериментальним. Багато освітніх закладів світу [1] включаючи й українські [2] вже давно інтегрували такі засоби до навчального процесу і використовують їх повсякденно. Таким чином, інформаційні технології приходять на допомогу викладачам та студентам: існує багато інструментів та продуктів, розроблених для підтримки онлайн-навчання, і їх можна підключити, не виходячи з дому, маючи лише доступ до інтернету.

Дистанційне навчання вирішує проблему не лише фізичної присутності студента та викладача в одній лекційній аудиторії, а й полегшує доступ до навчальних матеріалів, тестових завдань та надає можливість перевірки завдань онлайн [3]. Сам процес навчання у випадку такого підходу стає асинхронним і гнучким, що дозволяє студентам самим обирати час виконання поставлених перед ними задач виходячи лише з обмежень за термінами здачі.

Системи дистанційного навчання покладають на себе вирішення багатьох проблем навчального процесу і автоматизують деякі його аспекти, хоча багато де і потребують ручного управління та коректного налаштування. Серед багатьох функцій систем дистанційного навчання значної уваги заслуговують можливості інтеграції [4], додаткових навчальних матеріалів, сторонніх ресурсів та інструментів тому що ніяка система не може по своїй суті охопити всі можливі варіанти потреб користувачів. Саме в можливостях до розширення та перевикористання навчальних матеріалів полягає одна з ключових причин успіху систем дистанційного навчання в світі.

Саме тому можна вважати що проблематика інтеграції контенту систем дистанційного навчання є актуальною і в ході даної роботи значну увагу приділено дослідженню протоколів та стандартів інтеграції, можливостям які вони надають, проблемам інтеграції та порівнянню підходів до інтеграції навчального контенту.

Метою роботи є дослідження процесу інтеграції та програмна реалізація застосунку, що надає можливості інтеграції контенту та власних інструментів до навчальних курсів.

Для досягнення поставленої мети необхідно вирішити наступні задачі:

- дослідити сучасні системи дистанційного навчання;
- дослідження методи розширення систем дистанційного навчання та інтеграції контенту до них;
- запустити власний сервіс системи дистанційного навчання;
- розробити застосунок, що надає можливості розширення систем;
- виконати експериментальне дослідження готової системи.

Об'єкт дослідження – засоби інтеграції контенту до систем дистанційного навчання.

Предмет дослідження – системи дистанційного навчання, навчальні матеріали та розширення до них.

Методи дослідження. Для розробки програми інтеграції додаткового контенту та інструментів до систем дистанційного навчання було застосовано LTI, IP, HTTPS протоколи та сучасні технології для розробки веб застосунків, що працюють в інтернет мережі. В основі роботи систем дистанційного навчання та розширень до них лежать браузерні технології.

Наукова новизна роботи полягає у тому, що виконано дослідження засобів інтеграції зовнішніх модулів у системи дистанційного навчання та їх поєднання з сучасними технологіями веб розробки, що відкривають двері до комплексних систем з широкими можливостями.

Практична цінність результатів роботи полягає у тому, що розроблено програмний застосунок, який дозволяє за допомогою зовнішніх засобів розширювати систему дистанційного навчання додатковими інструментами або навчальним контентом.

Апробація результатів. За результатами роботи було опубліковано тези, що були подані на X Міжнародну науково-практичну конференцію «Сучасні

проблеми і досягнення в галузі радіотехніки, телекомунікацій та інформаційних технологій».

Дослідження виконано в рамках міжнародного проекту Virtual Master Cooperation Data Science (ViMaCs) за підтримки DAAD.

1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Поняття системи дистанційного навчання

Система дистанційного навчання (LMS) [5] – це система управління навчальною діяльністю, яка використовується для розробки, управління та поширення навчальних онлайн-матеріалів із забезпеченням спільного доступу. Дане поняття одночасно є достатньо простим і широким так як воно включає в себе цілу низку систем та платформ що надають можливості надання і отримання знань, підвищення кваліфікації, нормоконтролю, перевірки та закріплення навчальних досягнень.

В той час як саме поняття має декілька тлумачень і є широким, найбільш лаконічним і точним описом є застосунок що дозволяє автоматизувати управління, стеження та звітність навчального процесу. Такий застосунок має відповідати наступним функціональним критеріям:

- централізоване та автоматизоване управління;
- бути самостійним та використовувати інші самостійні сервіси;
- швидко збирати та доставляти навчальний контент;
- консолідація навчальних ресурсів на масштабованій вебплатформі;
- підтримка мобільності і стандартів;
- персоналізація контенту;
- можливість перевикористання знань.

Більш того, якісна система дистанційного навчання має надавати можливості інтеграції до інших корпоративних чи то університетських систем навчання або обліку, дозволяти збирати статистику та вимірювати ефективність, вплив та загальні витрати часу і ресурсів всіх навчальних ініціатив.

1.2 Концепція системи дистанційного навчання

Ідея створення таких систем в ІТ просторі зародилася у 1990і роки. Тоді вона реалізувалась просто як мережеві ресурси із спільним доступом та

розмежуванням прав користувачів, для реалізації чого вистачало засобів операційної системи (наприклад FTP ресурсів та методичних матеріалів). Але такий підхід є абсолютно не гнучким і функціонально роздробленим, бо розподіл контенту та перевірка знань абсолютно не залежать одне від одного і не являють одне ціле. Такий метод називають CAL (Computer Aided Learning), що передбачає підхід drill-and-practice (зубри і роби). LMS у своїй основі несе більш соціалізований підхід ILS (Integrated Learning Systems), що окрім навчального курсу надавала б засоби спілкування, менеджменту і трекінгу. Насамперед у концепцію LMS покладено простий принцип побудування інфраструктури, що забезпечує доступ та керує навчальними матеріалами, задає та оцінює навчальні та організаційні цілі, відстежує прогрес досягнення цих цілей, збирає та надає данні для спостереження за навчальним процесом цілком. LMS не тільки надає доступ до інформації, а й відповідає за реєстрацію та керування курсами. Окремо слід зазначити, що у деяких LMS використовується система аналізу розривів, системи оцінювання за методом 360 (метод оцінки компетентності персоналу) та система балів і винагород, що робить їх більш бізнес орієнтованими. Основоположною ж функцією надалі залишається функція планування, що вказує користувачу часові рамки на виконання тої, чи іншої задачі.

1.3 Вимоги до системи дистанційного навчання

Достатньо легко визначити як користувачі будуть користуватись конкретною системою дистанційного навчання адже це впливає з її технічних можливостей, функціоналу та обмежень. В той же час необхідно приділити значну увагу зазначеним особливостям систем при підборі конкретного продукту для інтеграції в навчання чи роботу. Треба чітко сформулювати та пріоретизувати вимоги і розглянути додаткові параметри на випадок майбутнього росту чи трансформації конкретної організації. Як тільки список необхідних вимог буде сформовано слід розпочати вибір конкретної системи згідно відповідності її функціоналу до конкретних параметрів.

В загальному випадку рекомендується розглянути наступні функціональні можливості як основні критерії при виборі системи [6] для майбутньої інтеграції перелічені в даному розділі.

Інтеграція з системами обліку кадрів – важлива технічна можливість систем дистанційного навчання що відповідає за синхронізацію членів організації та користувачів систем дистанційного навчання. Підтримка даної функції повністю автоматизує процес обліку і дозволяє зменшити затрати часу та необхідну роботу у випадку прибуття або відбуття нових членів організацій. Відповідно до ролі користувача в організації буде обрано роль в системі дистанційного навчання, у випадку деактивації чи вибуття користувача система автоматично призупинить роботу для конкретного члена організації.

Засоби адміністративного управління – можливість адміністраторів керування обліковими записами користувачів, їх реєстрацією, управління ролями та навчальними курсами, призначенням викладачів та менеджерів, назначення навчальних програм і курсів, розподіл внутрішніх фінансів та оплати. Адміністраторам важливо мати доступ до бази даних курсів щоб створювати стандартні звіти по групам користувачів чи ідивідуально. Звіти мають бути масштабованими і включати в себе як всіх користувачів, так і окремі групи з платформи. Система також має надавати можливість створювати розклади для викладачів, занять та окремих класних кімнат або курсів. Більш того, всі зазначені можливості мають бути автоматизованими, ергономічним та мати дружній до користувача інтерфейс.

Доступ до контенту – включає в себе як засоби доставки інформації (флешки, диски, онлайн джерела), так і методи (самостійний), мова та цільова аудиторія(робітники, учні) контентної бази.

Розробка контенту – основними характеристиками є створення, підтримка та збереження навчальної інформації. Це саме те місце що відповідає за перевикористання навчальних даних, контроль версій та інструменти створення навчального контенту.

Інтеграція контенту – важлива характеристика системи дистанційного навчання що дозволяє впроваджувати навчальні курси від сторонніх систем на нативному рівні. При виборі конкретного продукту слід звернути увагу на те що деякі системи сумісні лише з курсами прив'язаними до конкретної платформи, а деякі роблять більше не зазначено в стандарті. Постачальник повинен засвідчити, що сторонній вміст буде працювати в межах їх системи і доступ до сторонніх курсів повинні бути такими же простими, як використання інтерфейсного спадного меню.

Управління навичками – оцінка навичок та можливості управління обертаються навколо тих, хто навчається, оцінюючи прогалини у своїх компетенціях. Оцінки навичок можуть бути зкомпоновані з багатьох джерел, включаючи експертні відгуки та інструменти 360-зворотного зв'язку. Менеджери повинні мати можливість дізнатися чи зважуються, усереднюються або погіршуються результати у порівнянні за деякий проміжок часу для визначення розриву у кваліфікації. Компанії також можуть використовувати цю функцію для пошуку у своїх співробітників спеціальних навичок.

Можливості оцінки – важливою є функція оцінки, яка дає змогу створити якийсь відгук в межах продукту та включає оцінки як частину кожного навчального курсу. Механізми оцінки, тестування та відгуків допомагають розробникам створити програму, яка з часом стає ціннішою.

Дотримання стандартів – застосунок повинен намагатися підтримувати стандарти, такі як SCORM. Підтримка стандартів означає що система дистанційного навчання може імпортувати та використовувати навчальне програмне забезпечення, яке відповідає стандартам незалежно від авторської система, яка його створила. Треба приділити увагу тому факту, що якщо постачальник не засвідчив роботу свого контенту з вашою системою дистанційної освіти, то треба планувати додаткові витрати.

Налаштовуваність – якщо організації потрібно повністю переробити свою внутрішню систему і процеси для встановлення LMS або є необхідність використання дорогих ресурсів програмування для внесення зміни до системи

дистанційного навчання, то це, мабуть, не зовсім коректний вибір для конкретної організації. Крім того, корисно, якщо програмісти та дизайнери можуть отримати доступ до системи на низькому рівні та до її вихідного коду, особливо коли їм потрібно встановити процеси та стандарти що базуються на політиці компанії. Щоб зробити деякі системи інформаційними та зручними для користування, деякі провайдери систем дистанційного навчання мають громадські групи користувачів або консультації для клієнтів, які надають уявлення про встановлення або модернізацію систем.

Безпека – як правило, це одна з найпріоритетніших функцій у будь-якій системі даних, де міститься інформація про працівника та приватні дані. Заходи безпеки найчастіше включають в себе паролі, контроль доступу та шифрування.

1.4 Огляд популярних систем дистанційного навчання

Moodle (Modular Object-Oriented Dynamic Learning Environment, рис 1.1) [7] – це модульне об'єктно-орієнтоване динамічне навчальне середовище, яке називають також системою дистанційного навчанням, системою управління курсами, віртуальним навчальним середовищем або просто платформою для навчання, яка надає викладачам, учням та адміністраторам дуже розвинутий набір інструментів для комп'ютеризованого навчання, в тому числі дистанційного. Moodle можна використовувати в навчанні школярів, студентів, при підвищенні кваліфікації, бізнес-навчанні, як в комп'ютерних класах навчального закладу, так і для самостійної роботи вдома.

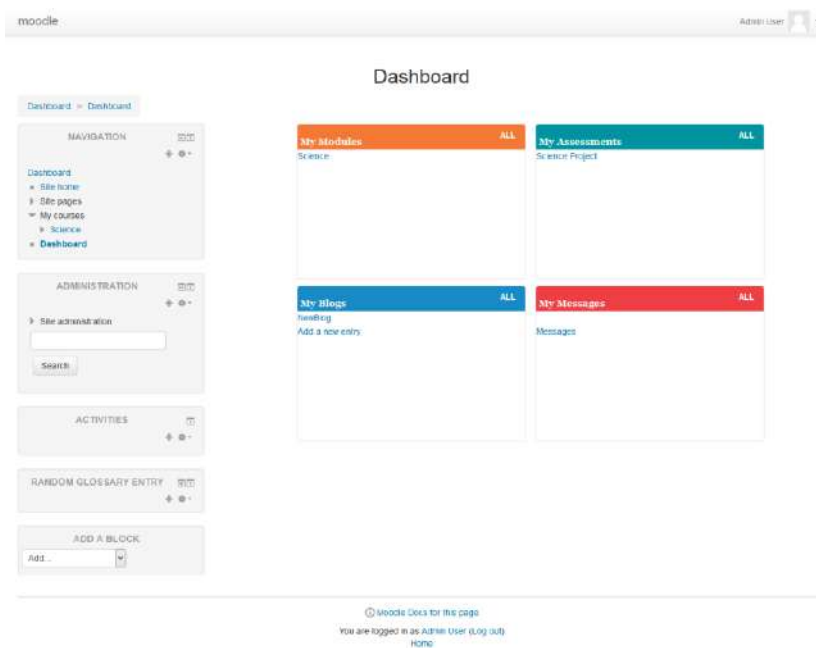


Рисунок 1.1 – Інтерфейс системи Moodle [8]

Moodle – це безкоштовна, відкрита система [9]. Вона не лише безкоштовна сама, а й не потребує для своєї роботи жодного платного програмного забезпечення. Тобто кожний навчальний заклад може впровадити у себе не просто безкоштовну і найбільш досконалу, а ще й абсолютно ліцензійну систему, не витративши жодної копійки на придбання програмного забезпечення. При цьому він може вносити зміни у код у відповідності до своїх потреб.

Автор концепції платформи e-learning Moodle австралієць Мартін Доугіамас (Martin Dougiamas) вважає, що головною її метою було створення системи, відмінної від доступних на ринку, а саме такої, в якій враховувалися б педагогічні аспекти, що базуються на основах пізнавальної психології, коли студент (учень) це активний суб'єкт, який самостійно створює свою власну систему знань, користуючись доступними йому джерелами. При цьому роль викладача (тьютора) полягає, в основному, в мотивуванні й підтримці своїх підопічних шляхом підготовки завдань для самостійного опрацювання, оцінювання результатів їх виконання, коригування знань студентів (учнів). Відповідно до основ суспільного конструктивізму, конструйоване знання

найбільш ефективно, коли студенти (учні) навчаються в співпраці. Це можливо тоді, коли студент (учень) працює в групі, ділиться своїм досвідом і думками, будучи відкритим для досвіду і думок інших. Важливою характеристикою проекту Moodle є його вебсайт, котрий є централізованим джерелом відомостей про систему, а також місцем для дискусій та співпраці користувачів Moodle: системних адміністраторів, викладачів, дослідників, проектувальників і розробників. Завдяки цьому Moodle підтримує інтерфейс більш ніж 80 мовами, зокрема є локалізація системи й українською мовою. Система використовується більше ніж у 60 тисячах організацій з більш ніж 200 країн світу.

Dosebo (рис 1.2) [10] – дана система має три користувацькі плани - Community, Reseller, Enterprise. Надає достатньо широкі можливості налаштування інтерфейсу для користувачів. Найчастіше використовується великими компаніями наступних професійних секторів: фінанси та страхування, охорона здоров'я, університети та школи, служби державного управління.



Рисунок 1.2 – Інтерфейс системи Dosebo [11]

eFront (рис 1.3) [10] – надає свої послуги в трьох користувацьких версіях: Community (базова), Educational, Enterprise. Виокремлюється між інших

приємним інтерфейсом та можливістю розширення за рахунок численних модулів і додатків. Більш продвинуті плани надають значно розширені і кращі засоби адміністрації та обліку.

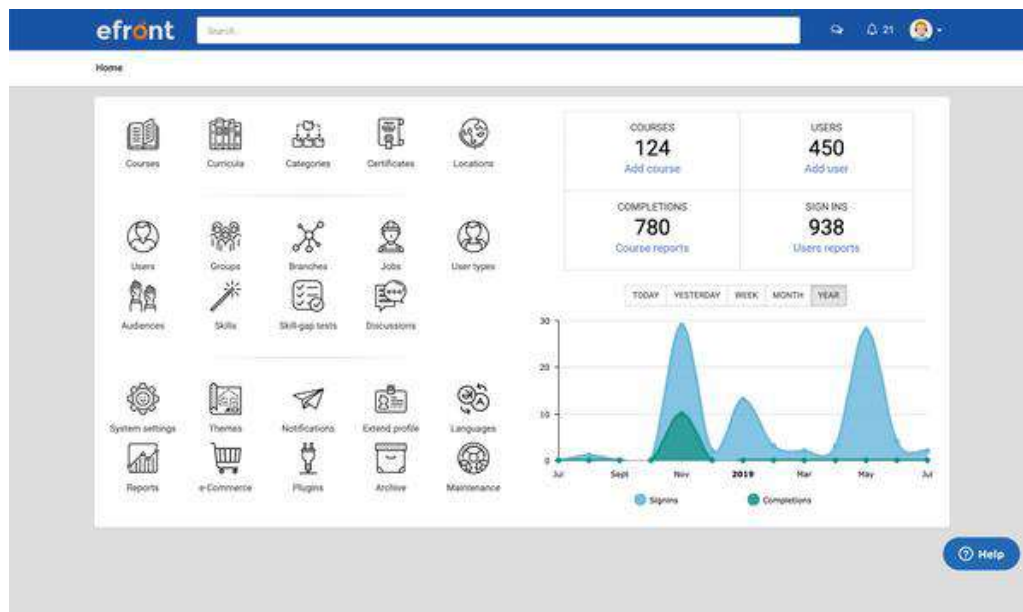


Рисунок 1.3 – Інтерфейс системи eFront [12]

Dokeos (рис 1.4) [10] – надає всі необхідні можливості для онлайн та змішаного навчання. Надає для користувачів наступні плани: Free (базовий), Education, Pro та Medical. Дана система дистанційного навчання надає безкоштовні навчальні матеріали та засоби з шаблонами для швидкого наповнення платформи власними курсами, засоби для збереження зображень, ведення конференцій та взаємодії в реальному часі, засоби для відстеження прогресу і часу, взаємодії між різними командами. Dokeos має вбудовані функції що полегшують роботу з різними мовами.

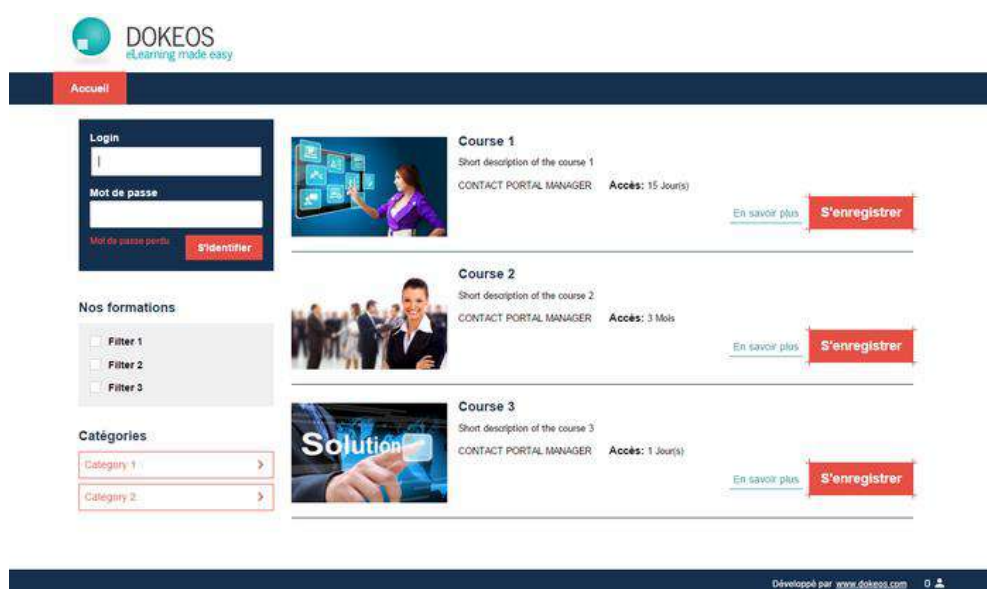


Рисунок 1.4 – Інтерфейс системи Dokeos [13]

Claroline (рис. 1.5) [10] – більше всього підходить для навчання, а не для ізольованих корпоративних систем з вузькою спеціальністю. Дана система надає можливості для вчителів одночасного створення онлайн курсів та управління навчальним процесом за допомогою вебсайту. Claroline було перекладено на 35 мов, дана система має велику спільноту користувачів та розробників по всьому світу.

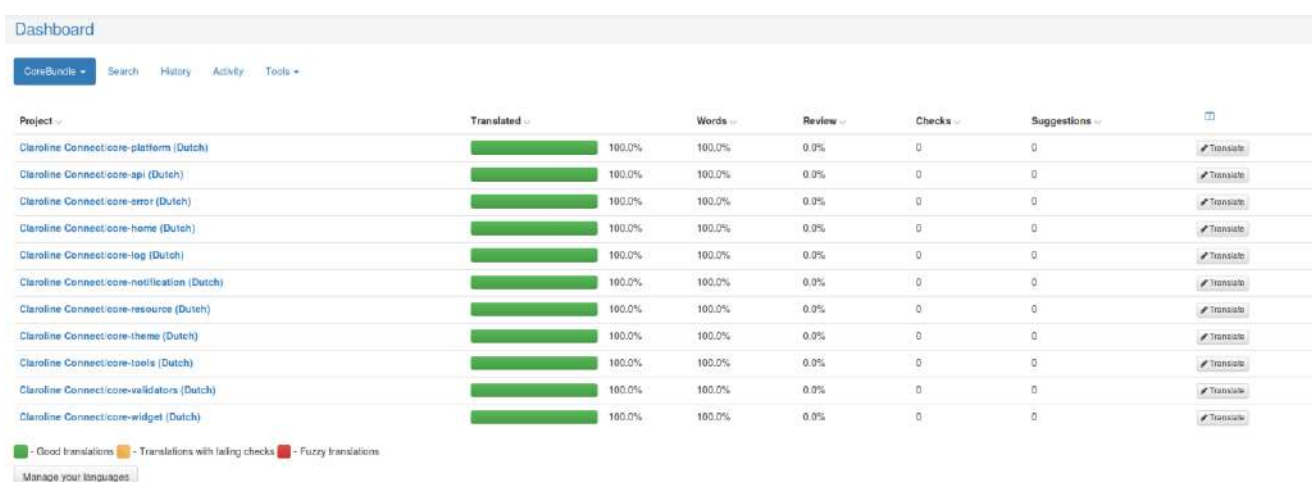


Рисунок 1.5 – Інтерфейс системи Claroline [14]

ATutor (рис. 1.6) [10] – пріоритетом даної системи є надзвичайно легка доступність. Характерною особливістю є розбиття на декілька модулів що є цілою системою (Atutor, Acontent, ATutor social). ATutor social є повноцінною соціальною мережею що дозволяє користувачам спілкуватись та взаємодіяти. Учні системи можуть зберігати контакти, створювати публічні профілі, спостерігати за активністю користувачів в мережі, вступати в групи та налаштовувати свій робочий простір за допомогою тисяч різних гаджетів та додатків доступних в мережі. ATutor Social можна використовувати незалежно як звичайне програмне забезпечення для персонального комп'ютера.

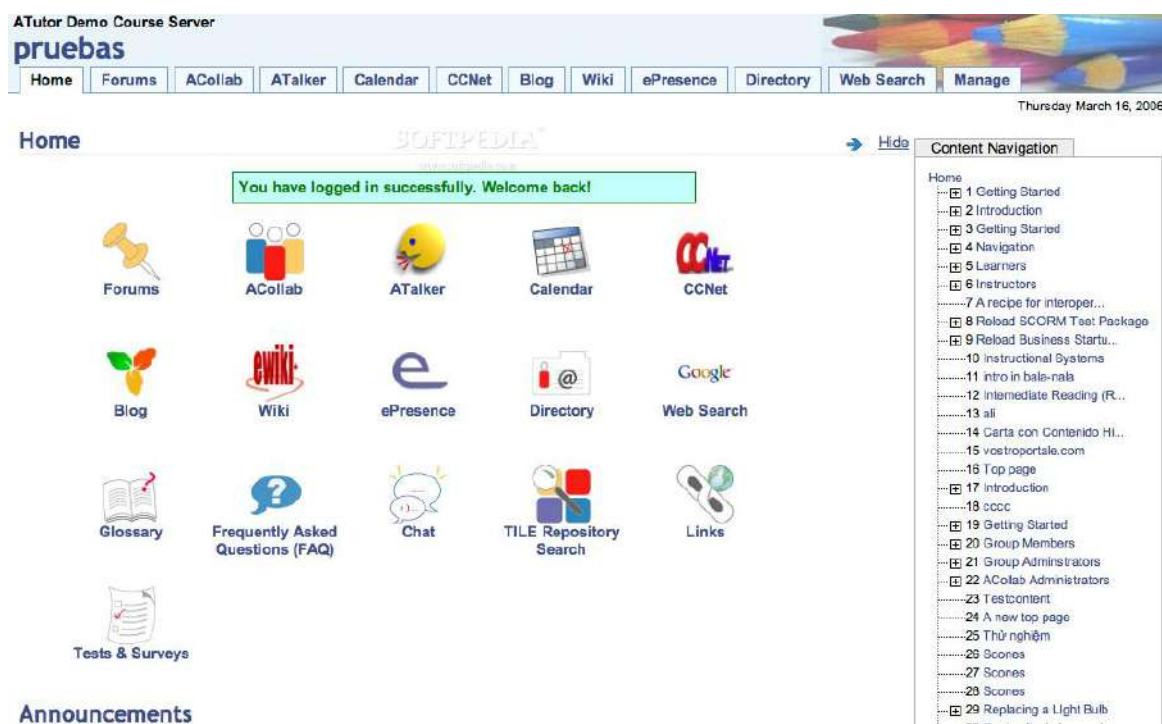


Рисунок 1.6 – Інтерфейс системи ATutor [15]

ILIAS (рис 1.7) [10] – характерною особливістю даної системи є те що вона надає передові засоби тестування, перевірки та надає засоби комунікацій та співпраці (чат, форум) і технології поширення (RSS та подкасти). Учні можуть налаштовувати свій робочий простір і заповнювати його всією необхідною для навчання чи то певних задач інформацією. Робочий простір включає в себе такі блоки як новини, приватні повідомлення, навчальні процеси, персональні помітки, закладки, зовнішні розсилки та іншу інформацію. Користувач

системи може переналаштувати конкретні елементи під свої потреби. Авторство контенту та менеджменту обмежене до xml модулів, глосаріїв та власних розділів вікіпедії.

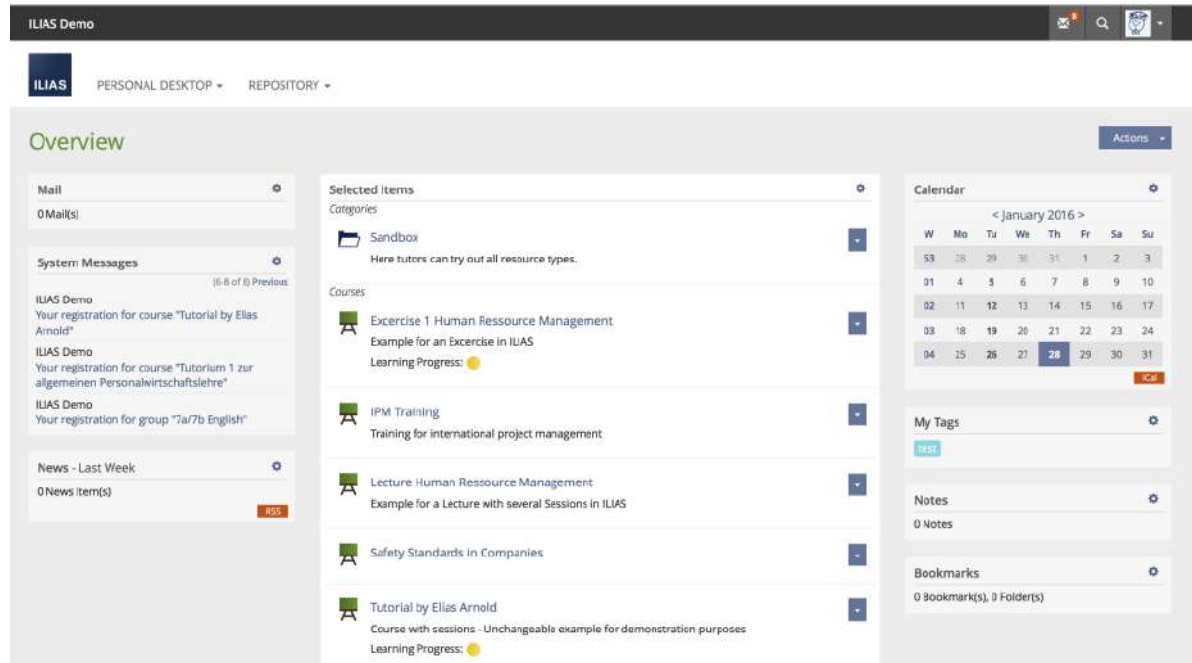


Рисунок 1.7 – Інтерфейс системи ILIAS [16]

.LRN (рис 1.8) [10] – одна з найбільш поширених систем з відкритим вихідним кодом, система має чималий список можливостей для спільнот що швидко розширюються. Дана система підтримує великий список навчальних стилів що починаються з традиційних підходів структурованого навчання до групової співпраці. Макет системи дозволяє налаштування під свої потреби. LRN було побудовано в першу чергу для навчальних спільнот, а не для більш стандартного підходу менеджменту навчальних курсів. Характерною особливістю даного застосунку є власна безкоштовна функція під назвою Jackdraw, що дозволяє створювати власні навчальні курси які потім можна експортувати в SCORM сумісному форматі та інтегрувати в інші системи.



Рисунок 1.8 – Інтерфейс системи .LRN [17]

OLAT [10] – в той час як дана система була розроблена університетом Цюриха, вона доступна і для інших навчальних закладів та бізнес організацій. Дана система написана на Java, Web 2.0 сумісна, гнучка та дружня до користувачів. В той же час дану систему досить важку запускати і встановлювати через її програмні особливості. Даний застосунок може витримувати навантаження до 700 одночасних користувачів. Якщо потребується більш широка підтримка, то дана система підтримує масштабування до 30000 користувачів. Користувачам даної системи надається як власний простір (профіль, сторінка), так і загальні навігація на навчальна панелі. Включає в себе засоби створення та редагування курсів.

Sakai CLE (Collaboration & Learning Environment) (рис. 1.9) [10] – система що достатньо стрімко розвивається, процес начання тут базується на взаємодії користувачів та обміні знаннями. Система включає в себе всі необхідні функції стандартної LMS та включає в себе цілу низку соціальних функцій таких як глосарій, вікі, новини та форуми. Користувачі можуть створювати різнопланові документи і ділитись ними завдяки інтеграції до сервісів Google. Система досить поширена як серед університетів, так і серед приватних компаній.

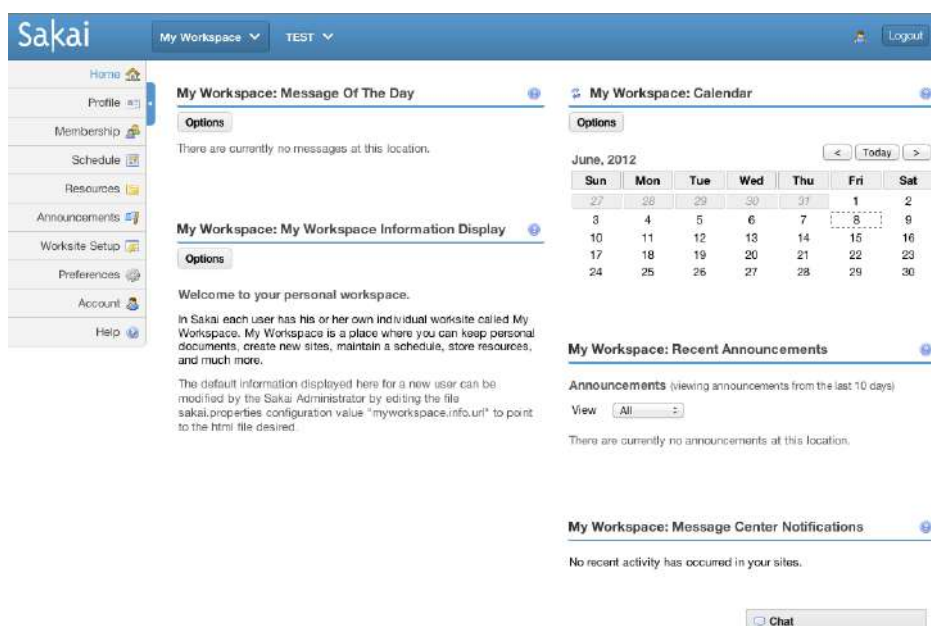


Рисунок 1.9 – Інтерфейс системи Sakai CLE [18]

1.4.1 Порівняння популярних систем дистанційного навчання

В результаті огляду сучасних систем дистанційного навчання і їх порівняння (табл 1.1) можна зробити висновок що сучасні системи хоча і мають різні підходи до візуального представлення курсів, монетизації і організації даних, але всі вони рухаються в напрямленні підтримки загальноприйнятих міжнародних стандартів і норм.

Таблиця 1.1 – Порівняння систем дистанційного навчання [10]

Функція	Docebo	eFront	Dokeos	Claroline	Moodle	Ilias	LRN
1	2	3	4	5	6	7	8
Підтримує багато мов	+	+	частково	+	+	+	+
Легке встановлення	+	+	+	+	+	+	+
Мова програмування	php	php	php	php	php	php	Open ACS

Продовження табл 1.1

1	2	3	4	5	6	7	8
Управління курсами	+	+	+	+	+	+	+
Засоби звітності	-	+	+	-	+	-	-
Управління контентом	+	+	+	+	+	+	+
Аутентифікація	+	+	+	+	+	+	+
SCORM сумісність	+	+	+	+	+	+	+
Налаштування груп користувачів	+	-	+	+	+	+	+
Створення власних курсів	-	-	+	-	-	-	-
Засоби комунікації	+	-	+	+	+	+	+
Модульність	+	+	+	+	+	+	+
Засоби онлайн оцінювання	-	+	-	-	-	+	+
Корпоративний план	+	+	+	-	-	-	-

1.5 Огляд систем масових відкритих онлайн-курсів

Масовий відкритий онлайн-курс (МВОК) являє собою інтернет-курс з необмеженою участю і відкритим доступом через Інтернет [19]. У доповненні до традиційних матеріалів курсу, такі як лекції, читання і задачі, більша частина МВОК надають інтерактивні форуми користувачів для підтримки взаємодії серед студентів, викладачів і асистентів. Сам термін з'явився в 2008 році, його авторство приписують Д. Кормье і Б. Александер. Вперше про відкриті онлайн-курсах заговорили більше десяти років тому, коли канадським педагогам з університету Манітоби вдалося запустити курс «Connectivism і Connective знань», який зібрав більше двох тисяч передплатників. З ростом кількості і популярності перших онлайн-курсів, стало зрозуміло, що це не новий модний інтернет-тренд, а сучасний відповідь на багато недоліків і упущення традиційної освітньої системи. Такі проблеми, як нестача компетентних викладачів, висока

вартість навчання, неактуальний матеріал і застарілі методи роботи з інформацією загальні для багатьох світових університетів. Онлайн-освіта була покликана їх вирішити.

Перша відкрита масова освітня платформа - Khan Academy (Академія Хана) - з'явилася в онлайн-просторі в 2006 році. Сьогодні Khan Academy надає більше п'яти тисяч абсолютно безкоштовних курсів, а її команда з однієї людини доросла до колективу з 80 - її співробітниками. Але справжній ривок онлайн-освіти відбувся в 2012 році, який газета The New York Times назвала «роком масових відкритих онлайн курсів». Причиною тому стала поява сайтів EDX, Coursera і Udacity, які з моменту заснування отримали істотну фінансову підтримку. І якщо ще десять років тому записатися на курси Массачусетського технологічного інституту або вивчати основи програмування під керівництвом викладачів Гарвардського університету могла тільки жменька обраних студентів, яким таке навчання обходилося в десятки тисяч доларів, з появою університетських курсів онлайн безкоштовний доступ до кращих кладезем знань на планеті став відкрито всім, у кого є комп'ютер і підключення до інтернет мережі. У той же час, стало зрозуміло, що потенціал онлайн-освіти набагато більше, ніж можливість слухати лекції в мережі. Інтернет не тільки зробив навчання доступним широким масам людей (як це сталося у випадку з поширенням університетських курсів), але і повністю змінив сам підхід до процесу навчання. Це, в свою чергу, дає значну перевагу при пошуку нової роботи. Саме тому слідом за платформи, що надає доступ до онлайн-курси університетів, все активніше став розвиватися сервіси інший формату. Вони вибрали інший шлях - структурування навчального контенту, зміна форматів подачі матеріалу і орієнтацію на практичні професійні теми. Результатом стала можливість вивчити невеликий обсяг інформації, яку можна негайно застосувати на практиці. Найбільш популярні серед таких сервісів - Udacity, Udemy, Lyndy безліч інших. Цінність освітніх платформ в тому, що вони видають строго необхідну інформацію. Питання відсікання непотрібного - найгостріший за методичної опрацювання програм.

Крім можливості швидко отримати нову кваліфікацію або оновити існуючу, короткі онлайн-формати зробили більш доступним отримання знань в області хобі. Згадувані вище платформи, як Coursera, Udacity, EDX, Udemy - лише одні з найбільш популярних. Існує десятки МВОК платформ, в тому числі відомий в Україні Prometheus.

1.6 Огляд протоколів або стандартів для повторного використання навчальних матеріалів

В процесі активного розвитку інформаційно-комунікаційних технологій завжди була актуальною проблема повторного використання програмного коду, щоб створене одного разу програмне забезпечення можна було легко використовувати і в майбутньому. В тому числі, в галузі освіти, важливе місце займає проблема забезпечення можливості повторного використання електронних навчальних ресурсів, наприклад, використання у новостворюваних курсах ресурсів, розроблених для інших систем, курсів, інших платформ тощо. Це дає великий позитивний наслідок, економлячи час і кошти, та дозволяє переносити на новітні платформи кращі розробки минулих років, відкриває можливості для створення набору навчальних ресурсів з можливістю повторного використання не тільки на рівні окремих розробників курсів, а й на корпоративному, національному та міжнародному рівнях.

Функціонал повторного використання електронних навчального контенту потребує забезпечення сумісності з різними навчальними середовищами і системами дистанційного навчання. Наукова спільнота давно зрозуміла важливість даної проблеми. Тому у світі ведеться розробка стандартів у галузі електронних засобів навчання, способів їх інтеграції та перевикористання. Серед найавторитетніших міжнародних організацій, які цим опікуються, можна назвати Learning Technology Standards Committee – LTSC, IMS Global Learning Consortium, що об'єднує близько 50 учасників, Global Learning Objects Brokering Exchange, до якого входять Ariadne Foundation та інші

організації. Серед найвідоміших, розроблених ними стандартів можна назвати IEEE Learning Object Metadata, IMS Content Packaging, IMS Learning Design та IMS Simple Sequencing.

Нині найбільшого поширення набув набір специфікацій і стандартів під загальною назвою SCORM (Sharable Content Object Reference Model – зразкова модель об'єкта змісту для спільного використання) [20], перша версія якого була розроблена у 2001 р. ініціативною групою ADL (Advanced Distributed Learning). Даний стандарт включає вимоги до організації навчального контенту, таким чином він забезпечує сумісність та надає можливість їх повторного перевикористання. Даний стандарт представляє навчальний матеріал окремими невеликими блоками, описаними XML, які можуть бути включеними у різні навчальні курси і використовуватися в системах дистанційного навчання. Нині розроблено багато застосунків для редагування та створення власних навчальних об'єктів у стандарті SCORM [21] і програм, що дозволяють зчитувати ці об'єкти [22]. Недоліком використання пакетів SCORM та інших розглянутих вище стандартів є те, що навчальний об'єкт у них, зазвичай, представлений у вигляді архівованого набору файлів (SCORM-пакета), що експортується з одного середовища і потім імпортується до іншого. Однак, сучасні засоби дистанційного навчання надають перевагу навчальним об'єктам, представленим не як файли, а як послуги веб-сервера. Такий підхід до розв'язання проблеми повторного використання навчальних ресурсів втілений, зокрема, в ідеї розподіленого навчального середовища, що передбачає спільне використання багатьох різномірних віртуальних середовищ, які взаємодіють з користувачем і між собою. Для забезпечення такої взаємодії IMS Global Learning Consortium створив новий стандарт – Learning Tools Interoperability (LTI) [23]. Він регламентує обмін даними між двома віртуальними середовищами, у процесі якого користувачі середовища А мають змогу одержувати доступ до ресурсів середовища В, наприклад, виконувати там певну навчальну діяльність. Водночас від середовища А до середовища В передаються необхідні для виконання цієї діяльності дані користувачів середовища А, а в зворотному напрямку (від

середовища В до середовища А) можуть передаватися результати виконання завдання, зокрема одержані при цьому оцінки. Хоча перша версія цього стандарту вийшла ще в 2010 р., однак він досі не знайшов ні належного висвітлення у наукових публікаціях, ні втілення в реальних проектах.

Варто зазначити, що до появи стандарту LTI задача інтеграції платформ дистанційного навчання з іншими програмними засобами розв'язувалася шляхом створення спеціальних плагінів. Наприклад, для платформи Moodle був розроблений плагін, що дозволяє інтегрувати в неї систему для проведення відеоконференцій BigBlueButton. Однак недоліком такого підходу є те, що подібні плагіни доводилось розробляти для кожної платформи і кожного зовнішнього програмного засобу, з яким вона інтегрується, а потім їх доводилось оновлювати під час переходу на кожну наступну версію як самої платформи, так і зовнішнього засобу. Завдяки стандартизації відтепер вже немає потреби ні розробляти, ні оновлювати такі плагіни, якщо платформа і зовнішні програмні засоби підтримують стандарт LTI. У системі управління навчанням Moodle цей стандарт підтримується, починаючи з версії 2.2, що вийшла в 2011 р. Там серед базових елементів курсу з'явилась діяльність під назвою Зовнішній засіб (External Tool), форма редагування, де зазначаються основні параметри: URL запуску, ключ клієнта і пароль. Ці параметри забезпечують використання в курсі Moodle тестового сервера платформи BigBlueButton як зовнішнього засобу. Отже, створивши діяльність із вказаними вище значеннями параметрів, будь-хто має змогу випробувати на своєму сайті Moodle як роботу діяльності Зовнішній засіб, так і роботу платформи відеоконференцій BigBlueButton. Це може бути не тільки випробування, а й практичне використання, адже не кожний має змогу встановити BigBlueButton на власному сервері, тоді як можливостей цього тестового сервера може бути цілком достатньо для використання платформи відеоконференцій BigBlueButton у невеликих масштабах.

1.6.1 LTI протокол та його робота

LTI (Learning Tools Interoperability) — стандартний протокол, що описує взаємодію навчальних платформ між собою [24]. Даний протокол надає можливість автоматично аутентифікувати користувачів з однієї платформи до іншої і передавати дані про успішності з другої на першу. Даний протокол може бути підтримуваним як тільки в одну зі сторін, так і в обидві паралельно. Завдяки підтримці даного протоколу користувачі можуть налаштовувати інтеграцію таких популярних платформ як Moodle, Open edX, Coursera, Blackboard LMS, Instructure Canvas, Sakai та інших. Платформа з якою планується інтеграція має підтримувати протокол LTI як learning platform (навчальна платформа). Дана технологія націлена на те щоб існував лише один уніфікований фреймворк для інтеграції будь-якої системи дистанційного навчання в будь-який програмний продукт для навчання.

Ключовими цілями даного стандарту є [25]:

- забезпечити просту і стислу модель взаємодії що включає в себе URL-адресу, ключ та секрет, які адміністратор системи або викладач курсу можуть ввести всистему дистанційного навчання;
- визначити протокол для запуску зовнішньої програми з системи управління вмістом таким чином, що буде підтримувати єдиний вхід що зберігає контекст навчання та ролі користувачів у цьому контексті;
- зробити портативні посилання на зовнішні програми, визначивши елементи даних, які можуть бути вбудовані в елементи внутрішньої системи.

Базовий життєвий цикл LTI взаємодії починається коли викладач чи адміністратор системи отримує доступ до навчальних засобів що знаходяться за межами даної системи. Адміністратор навчальних засобів повідомляє адміністратору чи то викладачу системи дистанційного навчання посилання, секрет і ключ від конкретного навчального засобу.

У випадку викладача найчастіше використовується LTI засіб для того щоб додати якийсь сторонній ресурс до їх загальної структури курсу як посилання в

інтерфейсі контрольної панелі системи дистанційного навчання. Інструктор додає посилання, ключа та секрет до елемента курсу, а учні в свою чергу при виборі засобу вже автоматично надають необхідні метадані в `iframe` елементі або ж новому вікні браузера.

У випадку адміністратора найчастіше додається "віртуальний засіб" до всієї системи, який потім перевикористовується учнями і викладачами. В даному випадку користувачі навіть можуть не знати що вони користуються елементами зовнішньої системи. В даному випадку при створенні або налаштуванні зовнішній засіб зазначається як елемент системи дистанційного навчання.

В обох випадках зовнішній засіб отримує запит на запуск який включає в себе інформацію про курс, роль, особистість викладача, ключ та підпис. Інформація про запуск посилається засобами HTTP форми що генерується в браузері користувача з прихованими елементами даних LTI протоколу. Приватні дані користувачів захищаються і підписуються OAuth стандартом захисту так, щоб інформація в період між генерацією і посиланням до зовнішнього засобу точно не була модифікованою або перехопленою. Після виконання запиту напуск користувач може бути перенаправленим в інше місце чи одразу почати роботу з засобом у випадку успіху операції.

1.6.2 SAML стандарт

SAML (Security Assertion Markup Language) [26] - мова розмітки, заснована на мові XML. Технологія розроблена для обміну даними про автентифікацію та авторизацію між захищеними доменами, зокрема між постачальником облікових записів і постачальником сервісу. Однією з важливих проблем, яку намагається вирішити SAML, є забезпечення наскрізної аутентифікації (Технологія єдиного входу, Single Sign On) при роботі через браузер.

SAML та LTI розвивались паралельно та незалежно на початку своєї епохи. Проблеми які вирішують обидва стандарти пересікаються в сфері

налагодження роботи авторизації та створення деякої "федеральної особистості користувача" в системі. SAML це шаблонна техніка для авторизації, транспортування та розпізнавання захищених даних яка вбудовується в великі системи що не мають конкретної прив'язки до технологій.

На загальному рівні проблеми які вирішують SAML та LTI в цілому відрізняються. LTI дозволяє встановити довіру між одним застосунком та зовнішнім засобом в той час як SAML надає можливість встановити взаємозв'язок між організацією та зовнішнім засобом за допомогою технології наскрізної аутентифікації (SSO). LTI транспортує інформацію при кожному запуску про користувацьку особистість, курс та систему звідки приходить запит кожного разу, також в запит включаються ідентифікатор сесії та системи, конкретну роль користувача, налаштування що були встановлені для конкретного курсу, користувацька інформація така як ім'я чи то пошта. SAML же надає ж засіб що включає в себе інформацію про особистість користувача на рівні організації, роль в організації та дані про пошту або назву облікового засобу.

В обох протоколах значною мірою захищається чутлива інформація про користувачів і надається лише у випадку необхідності. Як SAML, так і LTI розглядають випадки коли інформація про конкретного користувача як коректні. Частіше за все достатньо ідентифікатора користувача у випадку LTI чи то контролера у випадку SAML. У будь-якому випадку, зовнішня система не має опиратись на приватну інформацію про користувача як на внутрішній ключ. Значно відрізняються обидва стандарти на технічному рівні і в рівні складності реалізації кожного з них. SAML надає широкі та гнучкі інструменти і в той же час потребує значних об'ємів роботи та залишає після себе чималий слід. Зазвичай SAML інтегрується в вже готову там велику інфраструктуру адже для невеликих застосунків де використовуються економічні хостинги та провайдери зазвичай не надається доступ або ресурси необхідного рівня. LTI же використовує простий OAuth 1.0a протокол розроблений технічними гігантами. Через те що OAuth був розроблений для простої та швидкої розробки, слід який

залишає LTI є порівняно малим і не потребує вкладу значного об'єму ресурсів. Також перевагою OAuth є те, що переважно для кожної технології чи то мови програмування вже існують безкоштовні та робочі реалізації які досить легко інтегруються в вже існуючу інфраструктуру.

В контексті поширення та масовості використання LTI більше підходить для систем які взаємодіють з тисячами незалежних сторонніх засобів. Через те що дану технологію достатньо легко використовувати після реалізації і те що не потребуються значні об'єми додаткових ресурсів досить легко створювати цілі низки різнопланових. Програми основані ж на SAML дозволяють інтегрувати сторонні засоби лише після повного переходу системи на аутентифікацію основану на цьому протоколі і найбільш підходять в ситуаціях коли деякі організації ділять якісь великі об'єми даних або навчальних матеріалів з іншими системами основаних на SAML. SAML стандарт ідеально підходить для повної інтеграції деяких застосунків чи то систем застосунків де потребується взаємодія за межами простого перевикористання навчальних матеріалів чи курсів. LTI ж найбільш підходить для одного або декількох застосунків що є системами дистанційного навчання або схожі на них і потребують взаємодії з значною кількістю сторонніх засобів що можуть активно змінюватись чи бути зміненими на інші.

Системи дистанційного навчання мають досить унікальну потребу в передачі навчального контексту для кожного курсу. Якщо SAML надає загальну інформацію про роль на рівні організації, яка є більш статичною і глобальною, то в найбільш частих випадках системи дистанційного навчання потребують інформацію на рівні певного навчального курсу або семестру.

1.7 Висновки до розділу 1

В ході роботи було проаналізовано ключові поняття що стосуються систем дистанційного навчання та сучасну концепцію управління навчальним процесом. Було розглянуто основні функціональні можливості, загальні

характеристики стандартної системи і необхідні критерії при виборі застосунків. Серед можливих функцій було окреслено як необхідні для будь-якої системи, так і додаткові можливості які можуть полегшити роботу викладача чи студентів.

В процесі розгляду альтернативних застосунків, присутніх на ринку програмного забезпечення, було проаналізовано та порівняно цілу низку систем дистанційного навчання, що надають схожі функції, на основі чого і базується подальша робота. Результатом даного етапу роботи є табличні дані, на які можна опиратись при загальному порівнянні даних систем.

Подальший аналіз функціональних можливостей інтеграції систем дистанційного навчання включає в себе характеристику стандартних протоколів LTI та SAML, їх порівняння та опис їх роботи на абстрактному рівні. Дані підходи є загальноприйнятими для сучасних застосунків дистанційного навчання і, в окремому випадку, реалізують описані стандарти, хоча можуть бути розбіжності у версіях чи то реалізації.

2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

Попередній аналіз навчальних платформ і набору технологій надав базову інформацію для планування і проєктування програмного застосунку. Від готового застосунку очікується можливість інтеграції до наявних на даний момент систем дистанційного навчання і надання базових можливостей тестування чі то надання якоїсь інформації для вивчення.

2.1 Вибір системи дистанційного навчання для тестування застосунку

Оскільки розроблюваний застосунок буде інтегруватися до систем дистанційного навчання за LTI протоколом, головним критерієм до системи тестування є саме підтримка даного протоколу і можливість інтеграції сторонніх засобів до системи. Підтримка даного стандарту передбачає те що зовнішні елементи будуть вбудовуватись до системи за допомогою вебфреймів і не очікують від системи якихось технічних особливостей адже головну роль в даному випадку відіграє браузер користувача.

Досить важливими критеріями при виборі системи є відкритість коду, можливість локального розгортання системи, невелике навантаження на апаратну частину та безоплатність сервісів.

В процесі аналізу та перебору можливих варіантів вибір припав саме на систему дистанційного навчання Moodle через те що вона знаходиться в множині варіантів що відповідають вище описаним критеріям, є достатньо популярною в Україні та активно використовується в навчальних закладах. Розгортання даної системи для проведення тестів не потребує особливих умов від комп'ютерної системи, а наявність обширної документації та спільноти розробників полегшує вирішення можливих проблем в процесі роботи.

2.2 Розгортання системи дистанційного навчання та її вимоги

Розгортання системи дистанційного навчання не є тривіальною задачею оскільки готовий застосунок потребує окрім запуску власної кодової бази роботу низки сервісів [27] таких як база даних, веб сервер, інтерпретатор PHP та системні засоби операційної системи Linux. В стандартному випадку розгортання виконується без засобів автоматизації і заключається в запуску окремих модулів, завантаженні та встановленні необхідних бібліотек та ресурсів.

Одним з ключових елементів системи є база даних [28], Moodle підтримує цілу низку можливих варіантів для інтеграції до системи (табл. 2.1). Серед можливих варіантів є СКБД PostgreSQL, MySQL, MariaDB, Oracle Database.

Таблиця 2.1 – Вимоги до СКБД системи Moodle

СКБД	Мінімальна версія	Рекомендована версія
PostgreSQL	9.6	Остання
MySQL	5.7	Остання
MariaDB	10.2.29	Остання
Microsoft SQL Server	2012	Остання
Oracle Database	11.2	Остання

Конкретно в даному випадку вибір бази даних не грає великої ролі оскільки задача не сконцентрована навколо нюансів збереження даних у системі, тому вибір був за першим варіантом запропонованим документацією – на PostgreSQL.

Для коректної роботи системи необхідно встановити інтерпретатор PHP версії 7.2 або пізнішої, також потребуються деякі бібліотеки і розширення для PHP окрім стандартних. Статичні файли та обробка динамічних файлі обслуговується сервером Apache.

Зазвичай розгортання системи Moodle потребує від декількох годин до цілого дня якщо виникає необхідність в тому щоб запустити сервіси на

нововстановленій операційній системі. Для спрощення даної задачі достатньо зручним варіантом є обертання необхідних сервісів у віртуальні контейнери і подальший запуск їх у віртуальному середовищі Docker.

Docker – це платформа сервісів [29] і програмне забезпечення що використовують процесорні можливості віртуалізації для запуску програм в ізольованих пакунках що називаються контейнерами. Контейнери в системі є ізольованими один від одного та включають в себе власне програмне забезпечення, бібліотеки та ресурси. Контейнери комунікують між собою за конкретно вказаними каналами і на відміну від віртуальних машин і запускаються лише одним ядром операційної системи тому потребують менше системних ресурсів.

Такий підхід дозволить уникнути проблем з сумісністю системних сервісів, описати необхідні ресурси в одному конфігураційному файлі (рис. 2.1) та дозволить встановити і запустити всю інфраструктуру [30] за допомогою декількох консольних команд.

```
version: "2"
services:
  webserv:
    image: "moodlehq/moodle-php-apache:${MOODLE_DOCKER_PHP_VERSION}"
    depends_on:
      - db
    volumes:
      - "${MOODLE_DOCKER_VMWROOT}:/var/www/html"
      - "${ASSETDIR}/web/apache2_faildumps.conf:/etc/apache2/conf-enabled/apache2_faildumps.conf"
    environment:
      MOODLE_DOCKER_DBTYPE: postgres
      MOODLE_DOCKER_DBNAME: moodle
      MOODLE_DOCKER_DBUSER: moodle
      MOODLE_DOCKER_DBPASS: "moodle3ing"
      MOODLE_DOCKER_BROWSER: firefox
      MOODLE_DOCKER_WEB_HOST: "${MOODLE_DOCKER_WEB_HOST}"
  db:
    image: postgres:11
    environment:
      POSTGRES_USER: moodle
      POSTGRES_PASSWORD: "moodle3ing"
      POSTGRES_DB: moodle
  exttests:
    image: moodlehq/moodle-exttests
  selenium:
    image: "selenium/standalone-firefox:${MOODLE_DOCKER_SELENIUM_SUFFIX1:2.53.1}"
    volumes:
      - "${MOODLE_DOCKER_VMWROOT}:/var/www/html:ro"
```

Рисунок 2.1 – Конфігураційний файл інфраструктури Moodle

В результаті запуску системної інфраструктури за допомогою Docker вдається ізолювати системний простір від роботи Moodle, а в разі необхідності

можливо проаналізувати кожен контейнер на наявність проблем чи замінити його іншим не впливаючі на інші елементи. В конкретному випадку значно полегшується вибір бази даних та її розгортання, замість повного встановлення СКБД та конфігурації достатньо змінити декілька конфігураційних параметрів і почати процес оновлення середовища контейнерів.

2.3 Вибір мови програмування для реалізації розширення

Оскільки готове розширення буде працювати за LTI протоколом, немає ніяких обмежень та специфічних критеріїв до обратної технології окрім підтримки стандартів і надання відкритого URL для обробки серверних запитів. Готовий застосунок буде працювати на сервері та оброблювати запити користувачів інтернет мережі, однак для відображення інформації та даних в браузері потрібно створити і представлення для цих даних.

В результаті порівняння можливих альтернатив (табл. 2.3) було обрано мову програмування Javascript для того щоб прискорити розробку застосунку тому що вона може бути використаною як на клієнтській частині застосунку, так і на сервері.

Таблиця 2.3 – Альтернативні варіанти вибору мов програмування

Мова	Рік	Парадигма	Серверна	Клієнтська
1	2	3	4	5
Java	1995	Об'єктно-орієнтована, імперативна	+	-
Scala	2004	Функційна, імперативна	+	-
Ruby	1993	Об'єктно-орієнтована	+	-
PHP	1995	Імперативна, об'єктно-орієнтована	+	-

Продовження табл. 2.3

1	2	3	4	5
Javascript	1995	Імперативна, функційна, прототипно-орієнтована	+	+
Go	2009	Функційна, імперативна, об'єктно-орієнтована	+	-
Python	1991	Імперативна, об'єктно- орієнтована, функціональна	+	-
Rust	2010	Імперативна, структурна, функційна	+	-

2.4 Вибір технологій для реалізації розширення

2.4.1 NodeJS

Так як мовою програмування було обрано Javascript, область вибору була значно звуженою до конкретного набору технологій. Сам по собі Javascript є інтерпретованою мовою програмування та здебільшого виконується браузерними інтерпретаторами, хоча в 2009 році інтерпретатор відділили від браузера і на даний момент можна його використовувати як окремий елемент системи. В результаті серверний Javascript отримав власну назву – NodeJS та цілу екосистему.

NodeJS [31] – платформа з відкритим кодом для виконання високопродуктивних мережеских застосунків, написаних мовою Javascript. Якщо раніше Javascript застосовувався для обробки даних в браузері користувача, то NodeJS надав можливість виконувати JavaScript-скрипти на сервері та відправляти користувачеві результат їхнього виконання. Платформа Node.js перетворила JavaScript на мову загального використання з великою спільнотою розробників.

NodeJS має наступні властивості:

- асинхронна одно-нитева модель виконання запитів;

- неблокуючий ввід/вивід;
- система модулів CommonJS;
- рушій JavaScript Google V8.

Для керування модулями використовується пакетний менеджер npm (node package manager).

2.4.2 Typescript

Typescript [32] – це мова програмування з відкритим вихідним кодом що базується на Javascript і є одним з найпопулярніших її розширень. Головною її відмінністю є додавання статичного аналізатору типів. Типи надають можливість описувати об'єкти для того щоб код був більш документованим і інтерпретатор міг валідувати програмний код ще до безпосереднього виконання. Написання типів є опціональним тому що вбудована система типового аналізу в більшості випадків коректно розпізнає код і може доповнювати його та робити висновки в ході виконання програмного коду.

Альтернативою Typescript є Flow [33], це статичний аналізатор коду що має частково ті ж функції, але з часом втрачає популярність і не витримує конкуренцію з Typescript. В цілому Flow також надає можливість записувати анотації типів та перевіряти кодову базу проекту, але розвивається з часом все повільніше конкурента та не має такої ж широкої підтримки від спільноти розробників. Саме тому перевагу було надано Typescript.

2.4.3 ExpressJS

Express.js, або просто Express [34] – програмний каркас розробки серверної частини веб-застосунків для NodeJS, реалізований як вільне і відкрите програмне забезпечення під ліцензією MIT. Він спроектований для створення веб-застосунків і API. Де-факто є стандартним каркасом для Node.js. Автор фреймворка, TJ Holowaychuk, описує його як створений на основі написаного на

мові Ruby каркаса Sinatra, маючи на увазі, що він мінімалістичний, але має велику кількість плагінів, що підключаються.

2.4.4 Handlebars

Handlebars [35] – це система для серверного рендерингу шаблонів вебсторінок, що має готову реалізацію для таких мов програмування як ActionScript, C++, Clojure, CoffeeScript, ColdFusion, Common Lisp, D, Dart, Delphi, Erlang, Fantom, Go, Haskell, Io, Java, JavaScript, Julia, Lua, .NET, Objective-C, OCaml, Perl, PHP, Pharo, Python, R, Racket, Raku, Ruby, Rust, Scala, Smalltalk, Swift, Tcl, CFEngine та XQuery. Основою Handlebars є шаблонізатор Mustache що відрізняється від згаданої вище системи лише обмеженим функціоналом.

Серед альтернативних варіантів було надано вибір даних системі обробки шаблонів вебсторінок через її простоту, лаконічність та відмінну підтримку платформою NodeJS і бібліотекою ExpressJS. Хоча на даному етапі можна обрати будь-яку сумісну систему, в цьому немає великої необхідності тому що функціонал Handlebars повністю задовольняє потреби поставленої задачі.

2.5 Висновки за розділом 2

В результаті планування розробки програмного застосунку було обрано систему дистанційного навчання Moodle для тестування готового продукту через її простоту та активну підтримку спільнотою розробників.

Для реалізації програми було обрано набір технологій що зробить розробку простою, ефективною та надійною. Мову Javascript було обрано для того щоб зробити кодову базу клієнтської та серверної частини уніфікованою і більш простою. Використання Typescript надало можливість писати більш надійний та коректний програмний код з додатковими можливостями валідації без значних показників регресії при зміні функціональних модулів програми.

3 ОПИС ПРОГРАМИ

3.1 Функціональні можливості розробленого застосунку

Розроблений застосунок слугує базовим каркасом для багатьох можливих варіантів LTI розширень, що можуть слугувати постачальниками контенту і навчальних матеріалів. Як варіант, можуть бути розробленими LTI модулі для перевірки знань, їх закріплення чи просто надання якоїсь інформації. Одним із головних плюсів даної розробки, як і будь-якого вебсайту, є масштабованість та можливість регулярних оновлень без необхідності підтримки старих версій чи зупинки роботи сервісу.

На даний момент розширення може являти собою або простий обліковий засіб для того щоб рахувати студентів, що підтвердили свою присутність в якомусь із курсів, або ж в другому варіанті надає можливість вводити програмний код учням і надсилати його на перевірку.

Для керівників курсів в системі даний застосунок виглядає як контролюючий засіб і відображає список учнів, що підтвердили свою присутність у першому випадку, або надає можливість перевірити програмний код кожного з учнів окремо у другому випадку.

Вибір активного модуля залежить повністю від того як керівник курсу налаштує його і які параметри вкаже при додаванні URL посилання LTI застосунку.

3.2 Додавання застосунку до системи

Розроблений застосунок надає можливості інтеграції до систем що підтримують LTI протокол. На прикладі Moodle (рис 3.4) ми можемо побачити додавання розширення як зовнішнього засобу.

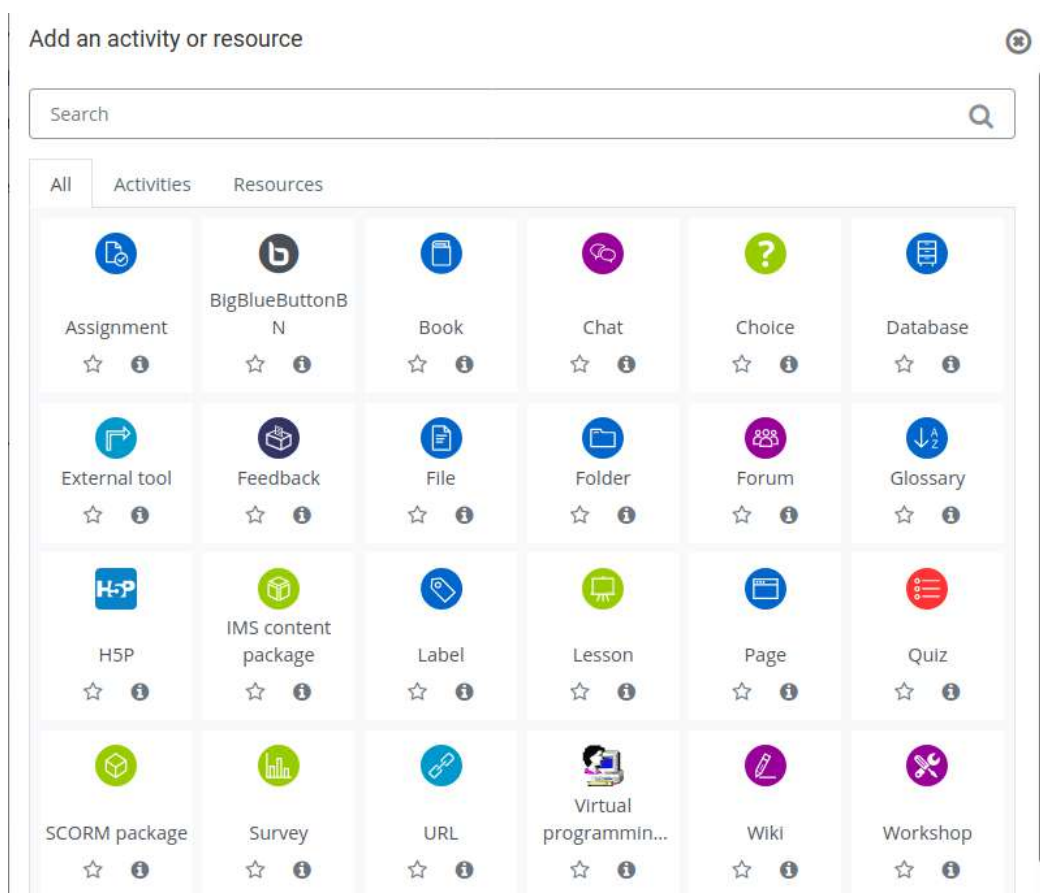


Рисунок 3.1 – Додавання зовнішнього засобу до системи Moodle

Подальший процес потребує від адміністратора курсу додаткових даних, що є стандартними в процесі інтеграції зовнішніх засобів за LTI протколом. В даному випадку в URL адресою слугує “localhost:3000/launch_lti”, оскільки локально застосунок запущено у вигляді веб сервера, що прослуховує порт 3000 і надає окремий шлях для запуску розширення і створення сесії користувача. Окремо додаються поля ключа та секрету системи, до якої додається розширення. Зазвичай ці дані генеруються в повністю реалізованих сервісах після процесу реєстрації, активації та підтвердження облікового засобу користувача, а в окремих випадках дані послуги можуть монетизуватися за допомогою продажів підписки чи то ключів.

Оскільки розроблювана програма є більше дослідницькою та демонстраційною, коректні дані було вбудовано в програму без ускладнень процесу і додано як зовнішній засіб до системи (рис. 3.5).

The screenshot shows the Moodle course configuration page for adding an external tool. The page is titled "Adding a new External tool to topic 2". The left sidebar shows the course structure with "Topic 2" selected. The main content area is divided into two sections: "General" and "Administration".

General section:

- Activity name:** Example tool for diploma
- Activity description:** A rich text editor with a toolbar containing icons for bold, italic, underline, list, link, unlink, and other formatting options.
- Display description on course page:** ☐
- Display activity name when launched:** ☒
- Display activity description when launched:** ☐
- Preconfigured tool:** Automatic, based on tool URL. A "Select content" button is visible below.
- Tool URL:** http://localhost:3000/launch_iti. A green checkmark indicates "Using custom tool configuration".
- Secure tool URL:** (Empty field)
- Launch container:** Default
- Consumer key:** demo
- Shared secret:** k2C342AScx

Administration section:

- Course administration:**
 - Edit settings
 - Turn editing off
 - Course completion
 - Users
- Unenrol me from:** ExampleCourse
- Filters:**
 - Reports
 - Gradebook setup
 - Badges
 - Backup
 - Restore
 - Import
 - Copy course
 - Reset

Рисунок 3.2 – Заповнення інформації про зовнішній засіб

Після активації застосунку і додавання коректних даних зовнішній засіб можна побачити у списку конкретної теми курсу до якої його було додано (рис 3.6).

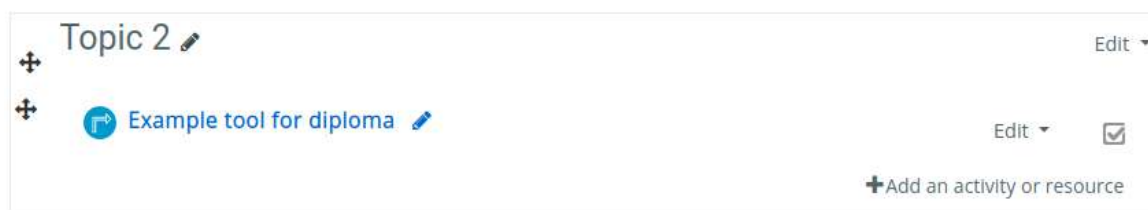


Рисунок 3.3 – Доданий зовнішній засіб до курсу

Подальша робота зовнішніх засобів, як правило, базується на технологіях вбудовування контенту зовнішніх сайтів. Даний підхід дозволяє засобами браузера захищено інтегрувати контент з інших ресурсів без ризиків безпеки користувачів чи нанесення шкоди системі.

3.3 Алгоритм роботи застосунку

Алгоритм роботи самої програми є набагато простішим ніж процес роботи інтеграції за LTI протоколом. Дане розширення розпочинає роботу одразу після завантаження сторінки вебсайту системи Moodle і функціонує в залежності від типу користувача та етапу його роботи з розширенням. Процес роботи програми (рис 3.7) є достатньо простим та ясным.

Алгоритм включає в себе етапи відправлення запиту до вебсервера середовища Moodle та його перенаправлення до LTI застосунку, що є стандартним процесом для такого роду розширень. Деякі кроки діяльності програми описані на високому рівні і включають в себе менші елементи, що є нормальними для серверного застосунку, такі як рендеринг представлень, завантаження ресурсів, повідомлення про помилки та обробка HTTP запитів що відповідають за клієнт-серверну взаємодію.

Після повної передачі контексту та управління даними користувача, вебсервер розширення генерує відповідь і повертає її безпосередньо користувачу системи.

У випадку адміністратора курсу буде підсумовано дані та надано звітність про заповнення форми присутності чи програмних коди студентів. Далі ці дані буде візуально представлено в залежності від типу активного застосунку.

Для учня буде надано інформацію про заповнення форми, або ж альтернативний варіант для того щоб підтвердити присутність чи надати якусь інформацію.

Робота завершується після закриття сторінки вебсайту.

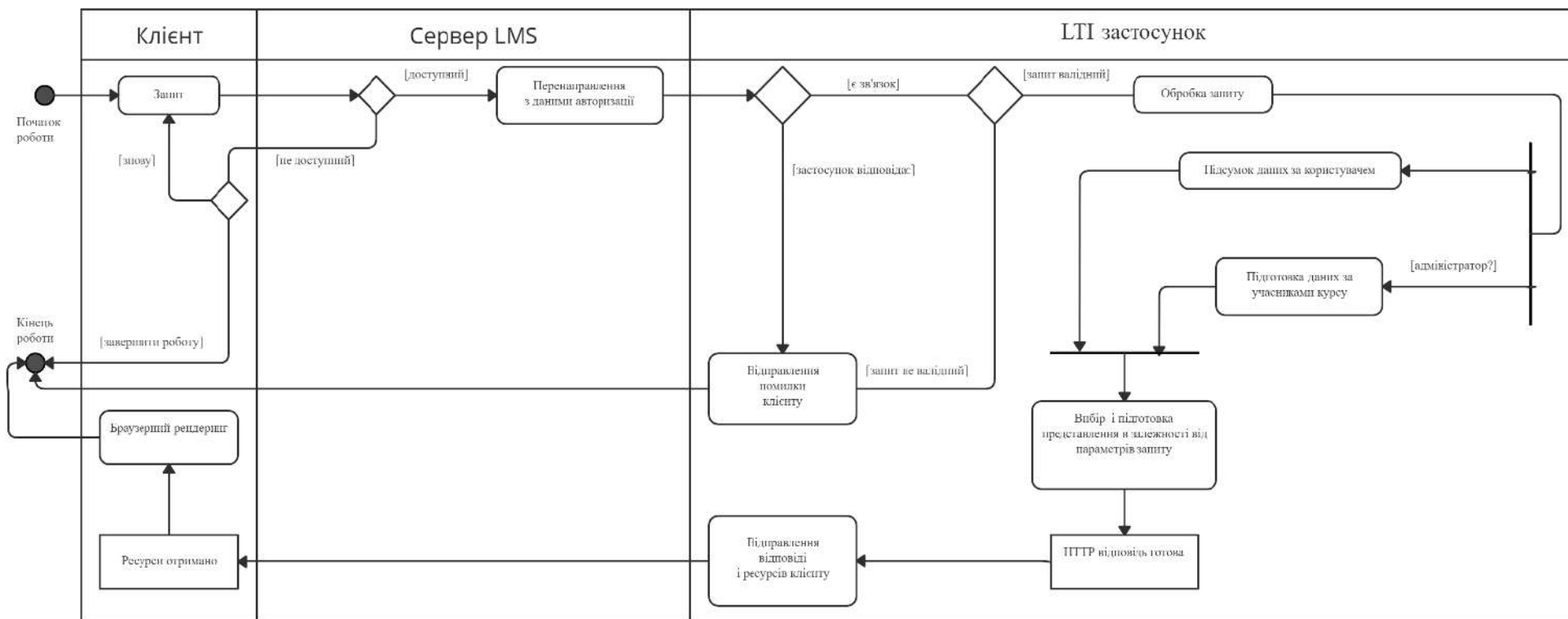


Рисунок 3.4 – Діаграма діяльності застосунку

3.4 Головні модулі LTI сервера

Модулі розробленого застосунку поєднуються як незалежні елементи системи що є ізольованими та інкапсульованими. В процесі розробки було виключено варіанти коли модулі порушують цей принцип, особливо це стосується циклічних залежностей і посилань.

Файлова організація проєкту розширення (рис 3.8) співпадає з модульним розподіленням для полегшення підтримки та внесення змін до програми.

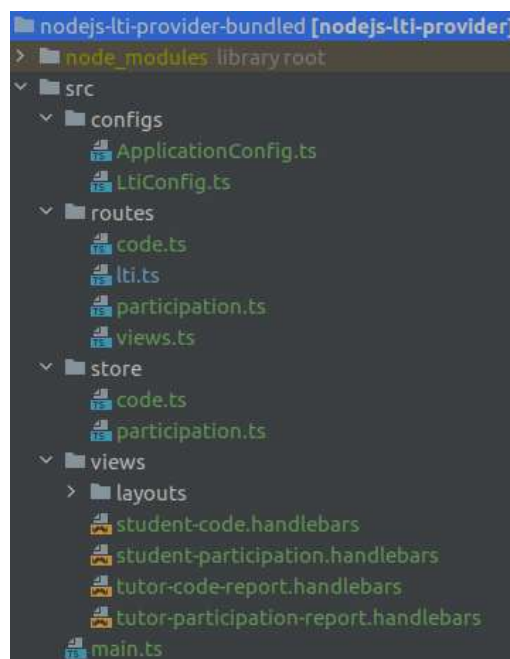


Рисунок 3.5 – Файлова структура проєкту

3.4.1 Головний модуль

Основний модуль програми відповідає за створення процесу сервера, налаштування його контексту і поєднання роботи останніх модулів програми. Такий підхід до організації дозволяє інженерам в ході роботи створювати нові елементи системи і реєструвати їх в головному модулі програми без необхідності модифікації вже готових модулів. Після налаштування і валідації роботи починається прослуховування системних портів.

3.4.2 Модуль конфігурації

Модуль конфігурації застосунку є достатньо простим і відповідає за налаштування середовища, валідацію та надає параметри виконання роботи сервера та LTI для інших елементів системи..

3.4.3 Модуль роутингу і управління

Модуль роутингу відповідає за розподілення зон відповідальності функціональних компонентів готової системи. В залежності від наданих параметрів та шляху користувацьких HTTP запитів буде виконано абсолютно різні дії та надано відповідь.

Також даний компонент відповідає за обробку контексту, парсинг та валідацію отриманих даних, обробку API запитів.

3.4.4 Модуль зберігання даних системи

Даний модуль відповідає за організацію та збереження серіалізованих моделей даних що необхідні в процесі роботи розширення, їх опис і організацію. В готовому до публікації проекті даний модуль, як правило, інкапсулює логіку взаємодії з базами даних та іншими модулями, проте, в даному випадку всі дані зберігаються в оперативній пам'яті для простоти, легкості і швидкості LTI розширення.

3.4.5 Модуль рендерингу представлень

Модуль рендерингу і шаблонізації відповідає за користувацький інтерфейс, представлення і візуалізацію даних. В результаті обробки запиту до розширення буде надано API відповідь в JSON форматі, або ж HTML розмітку для браузерного відображення.

3.5 Висновки за розділом 3

В даному розділі описано загальну логіку роботи програми, її алгоритм та зображено процес додавання її до системи дистанційного навчання на прикладі Moodle. Було наведено узагальнений опис основних модулів застосунку та зони їх відповідальності в контексті даної програми.

Архітектурно розроблений застосунок є вебсайтом, як і будь-який інший елемент що реалізує постачальника LTI сервісів для сторонніх систем.

Готовий програмний продукт підтверджує те що концепція інтеграції зовнішніх засобів до систем дистанційного навчання є достатньо простою і відкриває двері для широких можливостей розширення навчального процесу з перспективою подальшого перевикористання даних та сервісів.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення та умови застосування програми

Основною задачею позробленого застосунку є розширення функціоналу систем дистанційного навчання в результаті вбудовування його елементів. Такого роду функціонал не є вичерпним, але в конкретному випадку дана система надає можливості додавання до курсу інструменту обліку присутності студентів або ж редагування та надсилання програмного коду з користувацького браузера.

Необхідні характеристики для коректного запуску застосунку користувачем збігаються з системними вимогами для браузерів. Для запуску самого ж сервера що обслуговує LTI клієнтів достатньо мінімальної конфігурації найдешевшого варіанту віртуальної машини що надають провайдери хостингу, але найбільший вплив має навантаження на систему. Оскільки дана система практично не може бути навантаженою на 100% оскільки функціонал не є трудомістким, можна вважати наступні параметри за мінімально необхідні і достатні для її роботи:

- 64-бітна ОС Ubuntu 18;
- Intel Pentium 4 процесор або новіший;
- 1 ГБ оперативної пам'яті;
- доступ до інтернет мережі з шириною каналу від 10 мегабіт;
- 1 Гб вільного місця на жорсткому диску пристрою.

4.2 Характеристики програми

Розробку даного застосунку було виконано за допомогою Javascript та Typescript мов програмування, платформою для виконання програмного коду є NodeJS.

Програма є серверним застосунком і в процесі своєї роботи прослуховує один з системних мережливих портів, в даному випадку порт 3000. Всі запити

подані за локальною адресою що включає даний порт будуть обслуговуватись розширенням і отримувати відповідь в залежності від наданих параметрів та авторизаційних даних. На даних момент застосунок прослуховує маршрути що відповідають за модуль введення та збереження програмного коду, модуль обліку наявності студентів, модуль рендерингу користувацьких представлень та модуль підключення LTI (табл 5.1).

Таблиця 5.1 – Робочі маршрути серверного застосунку

Шлях	Метод	Призначення
/	GET	Отримання статусу застосунку
/application	GET	Рендеринг користувацьких представлень
/launch_lti	POST	Підключення зовнішньої сесії за LTI протоколом
/code	POST	Завантаження коду студентом за конкретним курсом
/participate	POST	Підтвердження участі в курсі студентом
*	*	Помилка зі статусом 404

Рендеринг застосунку відбувається в залежності від наданого параметру (табл. 5.2) типу до головного шляху вказаного вище.

Таблиця 5.2 – Рендеринг представлення в залежності від системою наданих параметрів

Параметр	Значення	Результат
type	participation	Представлення для моніторингу присутності
type	code	Представлення для редагування програмного коду
type	undefined	Текст про необхідність вказати тип представлення
type	*	Повідомлення про те що представлення не знайдено

Інфраструктура проекту на даний момент не потребує ніяких сторонніх сервісів і зберігає дані в оперативній пам'яті, але в майбутньому може бути доданою працездатна база даних для коректного зберігання даних.

4.3 Звернення до програми

Для запуску застосунку необхідно в першу чергу завантажити залежні модулі за допомогою менеджера пакунків Nodemon. Подальше робота потребує лише звернення до інтерпретатора Typescript з вказаними параметрами точки входу програми (рис 4.1).

```

neloreck@neloreck-development:~/Documents/Projects/diploma/nodejs-lti-provider-bundled$ npx ts-node --project ./tsconfig.json -r tsconfig-paths/register ./src/main.ts
App listening on port: 3000

```

Рисунок 4.1 – Запуск застосунку з командної стрічки Ubuntu

Для коректної роботи програми треба впевнитись що порт 3000 не використовується системою на даний момент. Даний застосунок може працювати лише в одному екземлярі оскільки він займає системний порт що може бути доступним тільки для одного процесу.

У випадку відсутності якихось залежностей чи відмові системи прослуховувати порт помилка буде виведено до терміналу (рис 4.2).

```

> nodejs-lti-provider@1.0.0 start /home/neloreck/Documents/Projects/diploma/nodejs-lti-provider-bundled
> ts-node --project ./tsconfig.json -r tsconfig-paths/register ./src/main.ts
Error: listen EADDRINUSE: address already in use :::3000
    at Server.setupListenHandle [as listen2] (net.js:1307:16)
    at listenInCluster (net.js:1355:12)
    at Server.listen (net.js:1441:7)
    at Function.listen (/home/neloreck/Documents/Projects/diploma/nodejs-lti-provider-bundled/node_modules/express/lib/application.js:641:8)
    at Object.<anonymous> (/home/neloreck/Documents/Projects/diploma/nodejs-lti-provider-bundled/src/main.ts:40:13)
    at Module.compile (internal/modules/cjs/loader.js:1138:30)
    at Module.m._compile (/home/neloreck/Documents/Projects/diploma/nodejs-lti-provider-bundled/node_modules/ts-node/src/index.js:858:23)
    at Module._extensions..js (internal/modules/cjs/loader.js:1158:10)
    at Object.require.extensions.<computed> [as .ts] (/home/neloreck/Documents/Projects/diploma/nodejs-lti-provider-bundled/node_modules/ts-node/src/index.js:858:23)
    at Module.load (internal/modules/cjs/loader.js:986:32)

```

Рисунок 4.2 – Повідомлення про зайнятий системний порт 3000

4.4 Вхідні та вихідні дані

Вхідними дані програми:

- облікові дані LTI запису;
- тип розширення для конкретного курсу;
- дані облікового запису користувача;
- підтвердження присутності користувача;
- програмний код користувача.

Вихідні дані програмного застосунку:

- візуальне представлення в залежності від стану системи;
- звіт присутності користувачів;
- звіт наданих програмних кодів користувачів;
- вже введений код активного користувача.

4.5 Повідомлення та обробка помилкових ситуацій

Програма в ході своєї роботи може повідомляти про помилки в процесі своєї роботи за допомогою консольних логів. Оскільки на даний момент програмний продукт є на завершеному етапі розробки, помилок не має бути. Можливі варіанти включають в себе також випадки коли залежності відсутні або ж системний порт 3000 зайнятий іншим процесом. За відображення можливих проблем відповідає інтерпритатор NodeJS, а в деяких випадках власні логи що його перевикористовують.

У випадку помилки при обробці запиту користувач отримає відповідь з HTTP статусом 500. У випадку некоректної авторизаційної інформації чи спроби обробки запиту за невідомим шляхом користувач отримає відповідь з HTTP статусом 400.

5 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

5.1 Призначення програми

Розроблене програмне забезпечення надає користувачам можливість розширювати стандартний список інструментів систем дистанційного навчання. Додатковий функціонал коректно розпізнає користувачів, курс до якого він доданий та тип інструменту який було обрано при додаванні. В залежності від того хто активує сторінку розширення контент буде різним і відіграти ролі активного користувача.

На даний момент розширення надає такі можливості:

- створення інструменту обліку присутності;
- створення інструменту редагування і відправки програмного коду в редакторі коду з підсвіченням синтаксису;
- облік учнів що підтвердили свою присутність;
- облік і перевірка відправлень коду учнів;
- підтвердження своєї присутності на курсі;
- відправлення та редагування програмного коду в веб редакторі.

5.2 Умови виконання програми

Системні вимоги для роботи застосунку повністю співпадають з вимогами систем дистанційного навчання, в даному випадку – Moodle.

Moodle сумісний з будь-яким стандартним браузером [36]. Дана система регулярно тестується і відповідно працює з наступними браузерами для персонального комп'ютера:

- Chrome;
- Firefox;
- Safari;

- Edge.

Починаючи з версії 3.10 Moodle не підтримує Internet Explorer 11. Safari та нижче має відомі проблеми з сумісністю починаючи з версії Moodle 3.10.

Наступні браузері підтримуються для мобільних телефонів:

- MobileSafari
- Google Chrome

Для найкращого користувацького досвіду і безпеки рекомендується використовувати останні версії актуальних браузерів.

Робота програми вимагає від користувача доступу до інтернет мережі або ж до локальної мережі де знаходиться система дистанційної навчання.

5.3 Виконання програми

5.3.1 Виконання програми адміністратором курсу

Першою точкою взаємодії програмного застосунку і адміністратора є додавання зовнішнього інструменту (рис. 5.1) до системи дистанційного навчання, в даному випадку - Moodle.

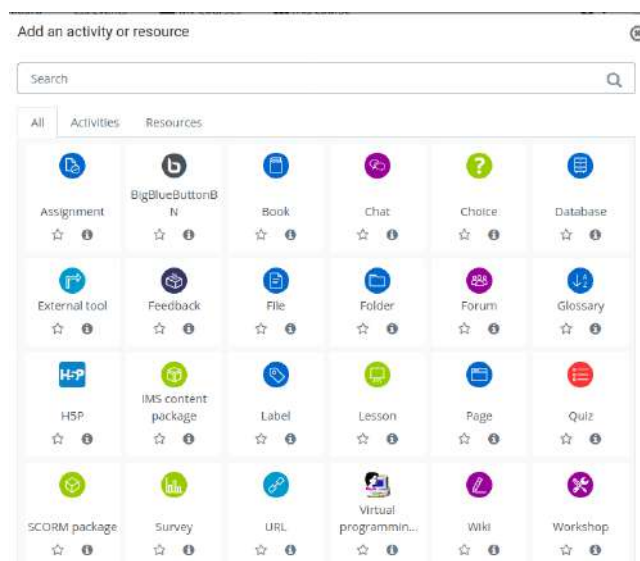


Рисунок 5.1 – Додавання зовнішнього засобу до Moodle

Налаштування зовнішнього засобу полягає в наданні посилання до працюючого сервера що обробляє запити і в залежності від вказаних параметрів надає необхідний функціонал. В даному випадку розглядається варіант з підтвердженням присутності (рис 5.2).

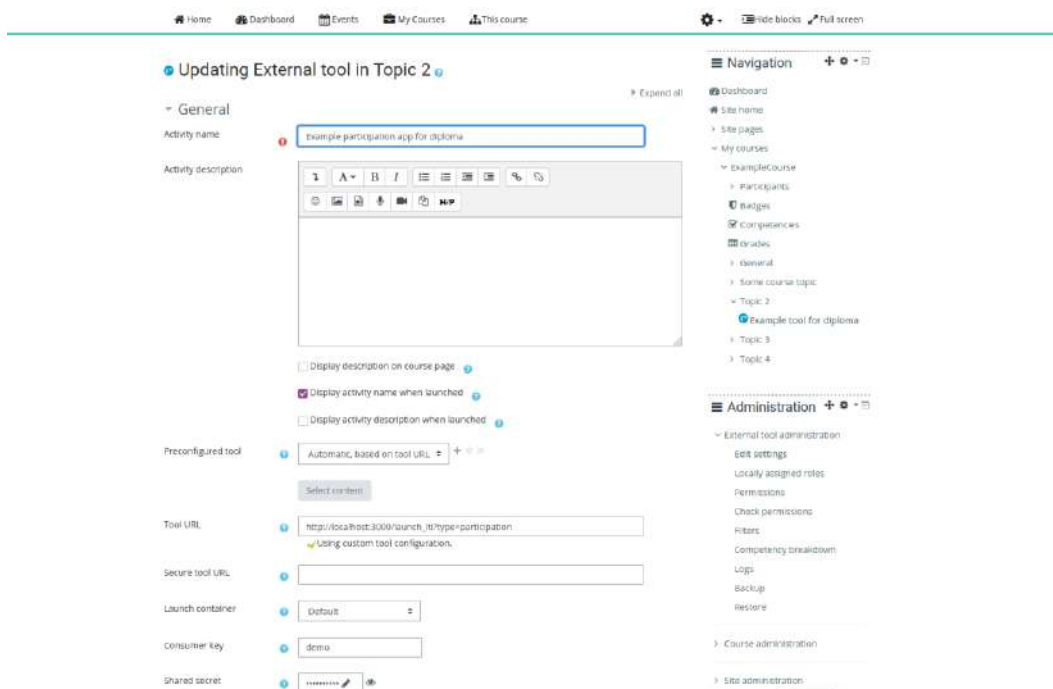


Рисунок 5.2 – Налаштування зовнішнього засобу

Після коректного додавання засобу до курсу можна побачити його в списку додаткових елементів при огляді всіх його етапів (рис 5.3).

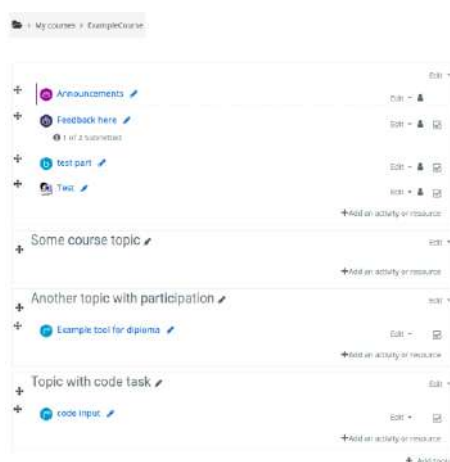


Рисунок 5.3 – Вікно огляду активного курсу

Подальша робота адміністратора переходить безпосередньо до самого застосунку і в залежності від участі користувачів в ньому повідомляє що ніхто не активував кнопку присутності (рис 5.4) або ж надає список учасників (рис 5.5). Також дане вікно надає контексну інформацію LTI розширення.

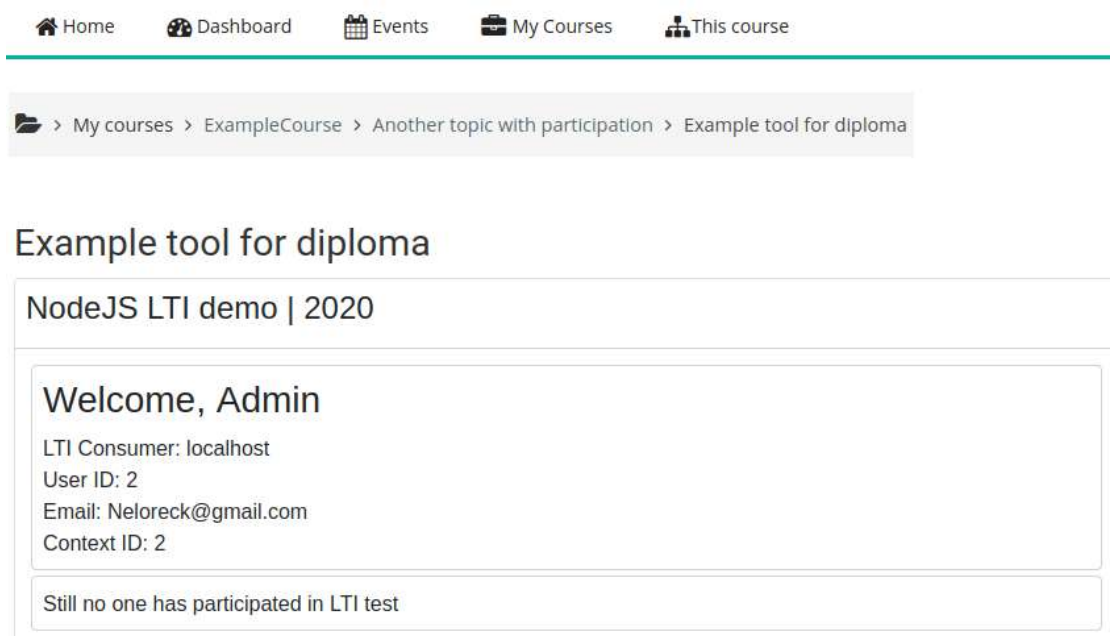


Рисунок 5.4 – Повідомлення про відсутність користувачів

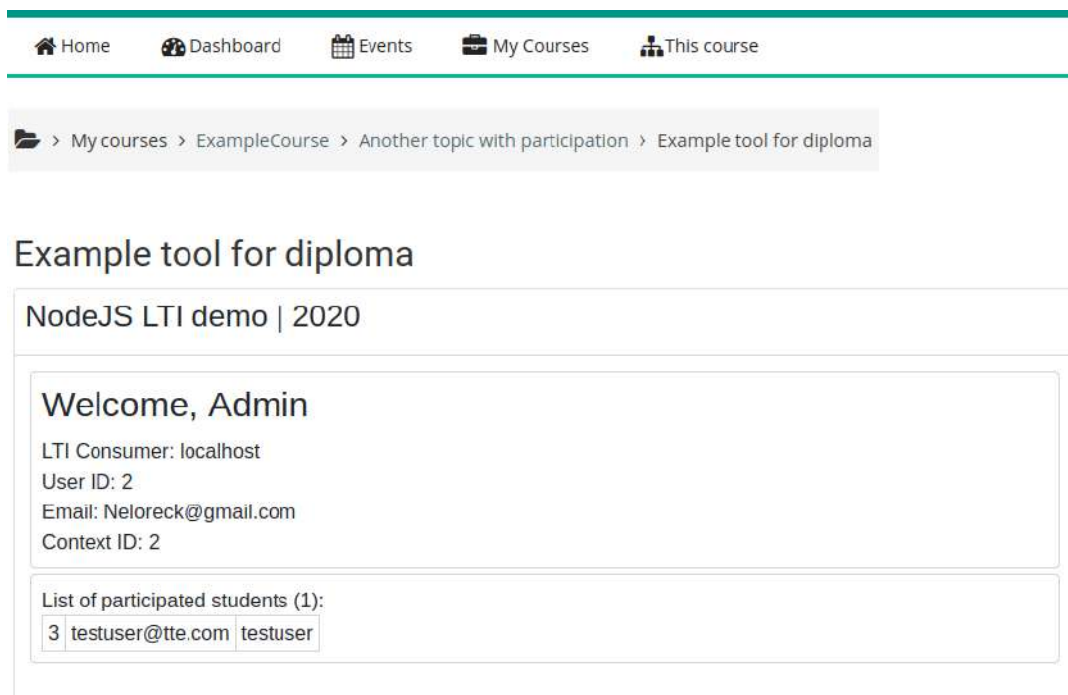


Рисунок 5.5 – Список користувачів що підтвердили участь

5.3.2 Виконання програми учнем курсу

Учні курсу системи дистанційного навчання розпочинають роботу з застосунком лише після додавання відповідного елементу системи адміністратором. Далі після навігації до розширення користувач може підтвердити свою участь насканням кнопки (рис 5.5) або ж побачить повідомлення про те що підтвердження вже надходило від даного облікового запису (рис. 5.6).

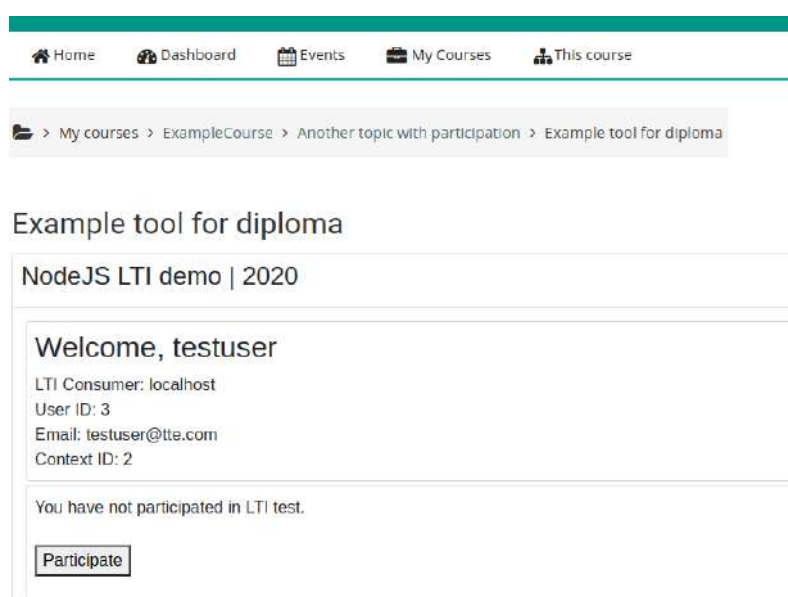


Рисунок 5.6 – Форма підтвердження присутності

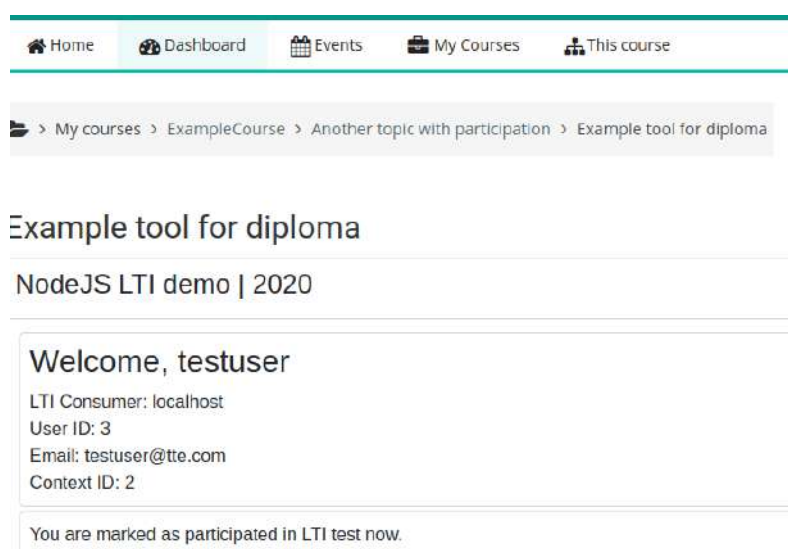
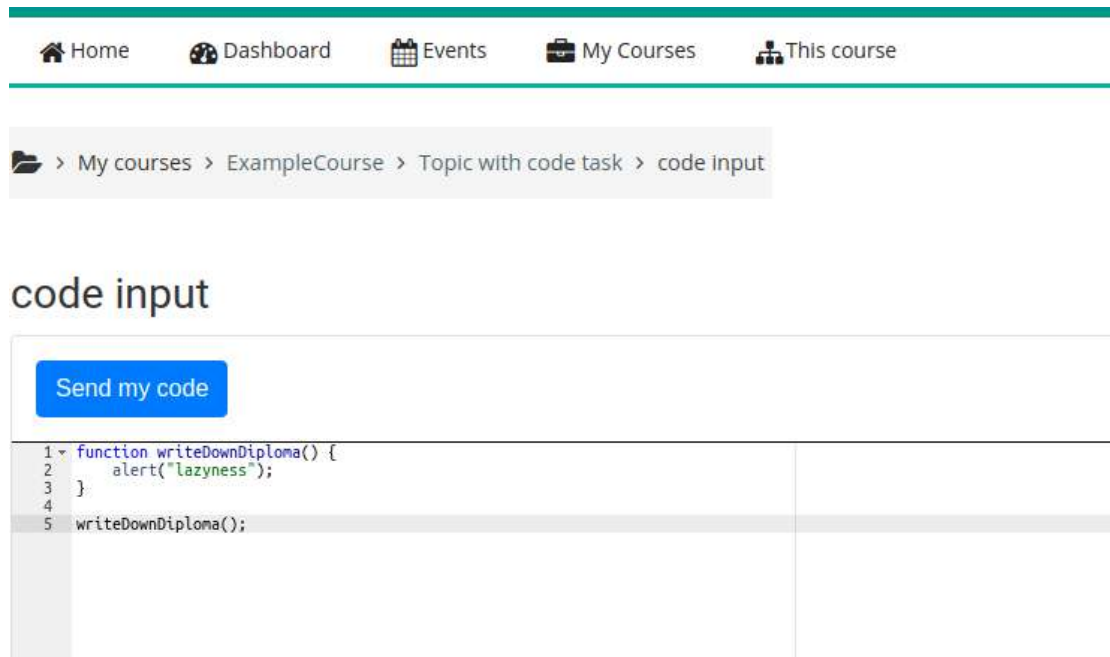


Рисунок 5.7 – Повідомлення про заповнення

У випадку якщо розширення було іншого типу і відповідало типу редактора коду, користувач побачить відповідну форму редагування коду і кнопку надсилання або оновлення даних (рис 5.7).



The screenshot shows a web application interface. At the top, there is a navigation bar with links: Home, Dashboard, Events, My Courses, and This course. Below the navigation bar, there is a breadcrumb trail: > My courses > ExampleCourse > Topic with code task > code input. The main content area is titled "code input". Below the title, there is a blue button labeled "Send my code". Below the button, there is a code editor with the following code:

```
1 function writeDownDiploma() {  
2   alert("lazyness");  
3 }  
4  
5 writeDownDiploma();
```

Рисунок 5.8 – Форма введення коду

5.4 Повідомлення оператору

Оператор програми інформується про стан виконання програми здебільшого за допомогою написів на елементах сторінки та інформаційних повідомлень. Оскільки застосунок є простим, то кожному етапу роботи відповідає якийсь конкретний шаблон в залежності від ролі користувача.

6 ЕКОНОМІКО-ОРГАНІЗАЦІЙНА ЧАСТИНА

В процесі виконання магістерської дипломної роботи було проаналізовано та досліджено способи інтеграції сторонніх засобів до систем дистанційного навчання та розроблено приклад простої інтеграції перевикористовуваного програмного засобу. Метою розроблюваного програмного продукту є можливість додавання до системи дистанційного навчання Moodle розширеного функціоналу власної розробки.

Для порівняння було розглянуто низку альтернативних засобів та систем дистанційного навчання що надають можливості інтеграції сторонніх засобів або ж виступають в ролі постачальників додаткового контенту, а саме – Docebo, eFront, Dokeos, Claroline, Moodle, ILIAS, LRN та деякі веб сервіси. Характерною особливістю більшості систем є те що вони надають свої послуги з моделлю підписки і помісячною оплатою або цілими ентерпрайз пакетами з персональними контрактами. Також майже всі системи надають свої послуги майже аналогічного плану і відрізняються здебільшого контентною базою та деякими модифікаціями.

Наявні системи дистанційного навчання мають відносно велику ціну на підписку в середньому до 2800 гривень в місяць, або ж монетизуються за рахунок продажів навчальних матеріалів з середньою ціною від 500-600 гривень одноразово. Більшість застосунків є закритими та не надають можливостей для модифікації, а розробка готових розширень потребує значних затрат часу для ознайомлення з документацією, проектування та реалізації готового програмного продукту.

В результаті аналізу ситуації на ринку систем дистанційного навчання та систем масових відкритих онлайн курсів було вирішено обрати трохи інакшу модель монетизації та підходу до продажу готового продукту адже майже неможливо конкурувати з титанами індустрії, особливо без значних інвестицій грошей та часу. Готовий проект планується продавати як бібліотеку функцій чи шаблон для майбутніх проектів, що надасть можливість значно пришвидшити

розробку покупцями власних застосунків. На противагу готовим альтернативам орієнтацію було встановлено на категорію покупців що є розробниками або планують розроблювати власні розширення до навчальних систем. Планується продавати створене забезпечення ціною 2846 гривень за одиницю в кількості 20 копій.

6.1 Планування дослідження та розробки програмного продукту

Даний проект планується розробити в декілька етапів, де для кожного з них планується закласти трудомісткість, тривалість та кількість людей необхідних для виконання поставленої задачі.

Для реалізації готового продукту планується залучити до роботи програміста та консультанта. Очікувана сумарна тривалість роботи – до місяця робочого часу, найпізніша можлива кінцева дата – 30 листопада, роботу планується почати 1 жовтня і закінчити до 16 жовтня.

Для коректних обрахунків необхідні наступні початкові дані [37]:

- робота програміста – до 2 неділь;
- робота аналітика – до 2 днів.

Формула для визначення тривалості роботи:

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

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

R – кількість виконавців, чол;

Q – трудомісткість, людино-днів.

Результати планування заносяться до таблиці 6.1.

Таблиця 6.1 – Характеристика запланованих робіт

Найменування робіт	Трудомісткість		Виконавці	Тривалість, днів
	люд.- днів	% до підсумку	спеціальність	
1 Аналіз технічної частини та протоколів для інтеграції	3	18.75	програміст	2
			аналітик	1
2 Визначення вимог до програмного продукту	2	12.50	програміст	1
			аналітик	1
3 Проектування структури програми	2	12.50	програміст	1
			аналітик	1
4 Розробка схеми функціонування програми	1	6.25	аналітик	1
5 Розробка програмного продукту	4	25.00	програміст	4
6 Тестування програмного продукту	2	12.50	програміст	1
			аналітик	1
7 Складання програмної документації	2	12.50	програміст	1
			аналітик	1
Разом	16	100.00		16

За даними таблиці 6.1 складаємо діаграму Ганта - графік планування розробки продукту (табл. 6.2). Дана схема надасть можливість наглядно побачити шлях розробки програмного продукту поетапно з відносною кількістю затраченого часу на кожен з етапів [37].

Таблиця 6.2 – Стрічковий графік планування розробки продукту

Найменування робіт	Календарний період, дні	
	02.10-06.10	09.10-13.10
1 Аналіз технічної частини та протоколів для інтеграції	 	
		
2 Визначення вимог до програмного продукту		
		
3 Проектування структури програми		
		
4 Розробка схеми функціонування програми		
5 Розробка програмного продукту		   
6 Тестування програмного продукту		 
7 Складання програмної документації		 

 - аналітик - програміст

6.2 Визначення витрат на розробку

Для визначення витрат на аналіз методологій та інструментів керування програмними проектами та розробку програмного продукту складається калькуляція кошторисної вартості робіт, яка включає наступні пункти [37]:

- основна заробітна плата;
- додаткова заробітна плата;
- відрахування на єдиний соціальний внесок (ЄСВ);
- витрати на спеціальне обладнання;
- матеріали і комплектуючі вироби;
- накладні витрати;
- податки.

6.2.1 Розрахунок основної та додаткової заробітної плати

В основу облічення необхідної заробітної праці буде закладено дані на основі раніше закладеного та обраховано робочого часу спеціалістів:

- час на розробку – до 16 днів;
- робота програміста – 10 робочих днів (0.5 місяця) з заробітною платою (ЗП) 8400 грн/міс;
- робота консультанта – 6 днів (0.3 місяця) з ЗП 5600 грн/міс.

Додаткову заробітну плату приймають рівною 10% від основної заробітної плати працівників [37] і розраховують за формулою 6.2:

$$ЗП_{\text{дод}} = ЗП_{\text{осн}} \cdot 0.1. \quad (6.2)$$

Результати розрахунків основної заробітної плати наведено в таблиці 6.3.

Таблиця 6.3 – Розрахунок основної та додаткової заробітної плати

Посада виконавця	Чисельність фахівців, осіб	Місячний оклад, грн	Кількість робочих міс	Сума основної ЗП, грн	Сума додаткової ЗП, грн
Програміст	1	8400	0.5	4200	420
Аналітик	1	5600	0.3	1680	168
Разом	2			5880	588

6.2.2 Відрахування на єдиний соціальний внесок

Єдиний соціальний внесок - обов'язковий платіж до системи загальнообов'язкового державного соціального страхування, що справляється в Україні з метою забезпечення страхових виплат за поточними видами загальнообов'язкового державного соціального страхування [38].

Для відрахування соціального внеску використовується наступна формула [37]:

$$\text{ВІД} = (\text{ЗП}_{\text{осн}} + \text{ЗП}_{\text{дод}}) \cdot 0.22, \quad (6.3)$$

де $K_{\text{с.з.}}$ – коефіцієнт відрахувань на соціальні заходи (єдиний соціальний внесок), згідно діючого законодавства становить 0.22;

$\text{ЗП}_{\text{осн}}$ – основна заробітна плата, грн;

$\text{ЗП}_{\text{дод}}$ – додаткова заробітна плата, грн.

$$B_{\text{с.з.}} = (5880 + 588) \cdot 0.22 = 1422.96 \text{ грн.}$$

6.2.3 Визначення витрат на матеріали

Матеріали, які знадобилися для розробки даної програми представлені в табл. 6.4.

Таблиця 6.4 – Витратні матеріали

Матеріал	Кількість	Вартість, грн
Папір формату А4	500 листів	105
Флеш накопичувач	1 шт.	200
Ручка	2 шт.	10
Папка для документації	3 шт.	5

Витрати на матеріали розраховуються за формулою [37]:

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

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

$K_{\text{т.з.}}$ – коефіцієнт, що враховує транспортно-заготівельні витрати (0.03 – 0.1);

$\bar{C}_i$ – ціна i -го найменування матеріалу, напівфабрикату, комплектуючого, грн.;

N_i – ціна i -го матеріалу, напівфабрикату, комплектуючого;

C_{io} – ціна поворотних відходів i -го найменування матеріалу, грн.;

N_{io} – кількість зворотних відходів i -го найменування;

n – кількість найменувань матеріалів, напівфабрикатів, комплектуючих.

Необхідні вхідні дані: $\bar{C}_i = 0$; $N_{io} = 0$; $K_{\text{т.з.}} = 0.05$.

Розрахуємо витрати на матеріали:

$$M = (1 + 0,05) \cdot (105 + 200 + 10 \cdot 2 + 5 \cdot 3) = 357 \text{ грн.}$$

Разом витрати на матеріали складають 357 грн.

6.2.4 Витрати на спеціальне обладнання

В цю статтю входять витрати на придбання, транспортування, монтаж і налагодження обладнання, а також враховуються витрати на оплату машинного часу електронно-обчислювальної машини (ЕОМ).

Амортизаційні відрахування визначають за формулою (6.5) [37]:

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

де Φ_6 – балансова вартість обчислювальної техніки, грн;

N_a – норма амортизаційних відрахувань на повне відновлення обчислювальної техніки.

Балансова вартість обчислювальної техніки становить 16000 грн, а норма амортизаційних відрахувань – 25%. Таким чином амортизаційні відрахування становлять:

$$A = 16000 \cdot 0.25 = 4000 \text{ грн.}$$

В результаті отримуємо 4000 грн.

В загальному випадку балансова вартість машини визначається як вартість машини плюс 5% на транспортні витрати і монтаж.

Витрати на електроенергію (6.6):

$$C_e = N_n \cdot \Phi_{\text{еф}} \cdot K_{\text{зв}} \cdot K_{\text{зм}} \cdot C_{\text{ел.е}}, \quad (6.6)$$

де N_n – номінальна потужність ЕОМ, кВт;

$\Phi_{\text{еф}}$ – річний ефективний фонд часу роботи ЕОМ, ч.;

$K_{\text{зв}}$ – середній коефіцієнт завантаження за часом;

$K_{\text{зм}}$ – коефіцієнт завантаження по потужності;

$C_{\text{ел.е}}$ – ціна одного кВт·ч електроенергії, грн.

Номінальна потужність ЭВМ – 0,35 кВт. Річний ефективний фонд часу роботи ЕОМ становить 1800 годин. Середні коефіцієнти завантаження за часом і за потужністю рівні відповідно 0,9 і 0,6. Ціна однієї кіловат-години електроенергії становить 1,68 грн. Таким чином одержуємо:

$$C_e = 0,35 \cdot 1800 \cdot 0,9 \cdot 0,6 \cdot 1,68 = 571.53 \text{ грн.}$$

В результаті витрати на електроенергію становлять 571.53 грн.

Зарплата обслуговуючого персоналу розраховується по формулі (6.7) [37]:

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

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

$K_{\text{відрах}}$ – коефіцієнт, що враховує відрахування на соціальне страхування і 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.

Заробітна плата обслуговуючого персоналу становить 3500 грн/міс.

Річний ефективний фонд робочого часу обслуговуючого ПК працівника дорівнює 1750 г/рік.

На обслуговування одного ПК по 2 години на місяць становить на рік 24 годин.

Отримаємо:

$$\Phi 3\Pi_{\Gamma} = 3500 \cdot 12 = 42000 \text{ грн};$$

$$K_{\text{отч}} = 0,22; t_{\text{обсл}} = 2 \text{ год./міс.}; \Phi_{\text{еф.обсл}} = 1750 \text{ г/рік};$$

$$3\Pi_{\text{обсл}} = 42000 \cdot (1 + 0,22) \cdot \frac{24}{1750} = 702.72 \text{ грн.}$$

Таким чином заробітна плата обслуговуючого персоналу становить 702.72 грн.

Сума витрат за розділом «Поточний ремонт устаткування» приймається рівною 3% від балансової вартості обладнання [37], що дорівнює 16000 грн, становить 480.00 грн. Інші витрати складають 5% від суми всіх попередніх витрат на утримання та експлуатацію обладнання за розділом «Інші витрати». Сума всіх попередніх розділів дорівнює 5274.25 грн, 5% від суми становлять 263.71 грн. Зведемо отримані дані в таблиці 6.5.

Таблиця 6.5 – Кошторис витрат на утримання і експлуатацію обладнання

Найменування статей витрат	Сума, грн
Амортизація обладнання	4000.00
Експлуатація обладнання (крім витрат на поточний ремонт)	571.53
Основна і додаткова заробітна плата обслуговуючого персоналу з відрахуваннями на ЄСВ	702.72
Поточний ремонт обладнання	480.00
Інші витрати	263.71
Всього	6017.96

Витрати на оплату машинного часу ЕОМ для написання і налагодження програми визначають по формулі (6.8):

$$C_{\text{мо}} = P_{\text{екс}} \cdot t_{\text{мо}}, \quad (6.8)$$

де $C_{\text{мо}}$ – витрати на оплату машинного часу, грн;

$P_{\text{екс}}$ – експлуатаційні витрати на одну годину машинного часу, грн/машино-год;

$t_{\text{мо}}$ – машинний час ЕОМ для написання і відладки цього програмного продукту, машино-год.

Для розрахунку експлуатаційних витрат на одну годину машинного часу використовуваної ЕОМ розраховують діленням суми витрат за кошторисом витрат на утримання та експлуатацію обладнання [37] (таблиця 6.5) на річний ефективний фонд часу роботи ПК. Річний ефективний фонд часу роботи ПК дорівнює 1800 годин.

В результаті експлуатаційні витрати на одну годину машинного періоду складають 3.34 грн/машино-год.

Розробка та відладка програмного продукту займає 16 днів в одну зміну по 8 робочих годин, що в сумі становить 128 годин.

Таким чином:

$$C_{\text{мо}} = 3.34 \cdot 128 = 427.52 \text{ грн.}$$

Витрати на оплату машинного часу склали 427.52 грн.

6.2.5 Інші прямі витрати

В інші прямі витрати входять витрати на комерційне програмне забезпечення, яке використовується при розробці, а саме загальне ПЗ, що використовується постійно при роботі з ПК (Windows 10 Professional 32/64bit) – 3800 грн, без ПДВ.

Розрахунок інших прямих витрат виконується за формулою 6.9 [37]:

$$S = \frac{C_{\text{по}} \cdot T_{\text{ктс}}}{\Phi_{\text{еф.ктс}} \cdot T_{\text{с.по}}}, \quad (6.9)$$

де $C_{\text{по}}$ – ціна програмного забезпечення;

$T_{\text{ктс}}$ – машинний час КТС, необхідне користувачу для розробки системи, машино-год/рік;

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

$T_{\text{с.по}}$ – строк служби програмного забезпечення, років.

$$S = \frac{5699 \cdot 256}{2 \cdot 1800} = 405.26 \text{ грн.}$$

6.2.6 Накладні витрати

До накладних витрат відносяться витрати на загальне управління і загальногосподарські потреби (заробітна плата апарату управління, канцелярські витрати і так далі), зміст і експлуатацію будинків. Накладні витрати включаються у вартість розробки програми непрямым шляхом – у відсотках до основної заробітної плати розробників. В даному випадку накладні витрати складають 40% до основної заробітної плати розробників, що складає 5880 грн, становить 2352 грн.

6.3 Розрахунок кошторисної вартості робіт

Визначення повної собівартості програмного продукту здійснюється шляхом додавання операційних витрат та витрат по збуту. Визначення сукупності операційних витрат здійснюється шляхом додавання витрат на оплату праці, відрахувань на ЄСВ, матеріальних витрат, витрат по спожитим послугам, амортизації та іншим.

Визначення ціни реалізації програмного продукту здійснюється виходячи з повної собівартості інноваційного продукту та відсотка очікуваного прибутку, що становить 30%, і відсотка податку на додану вартість (ПДВ), що становить 20% [37].

Результати визначення витрат на розробку у вигляді калькуляції кошторисної вартості робіт наведені в таблиці 6.6.

Таблиця 6.6 – Калькуляція кошторисної вартості робіт з розробки програмного продукту та ціна продукту

Найменування статей витрат	Сума, грн	Питома вага до підсумку, %
Основна заробітна плата	5880.00	51.40
Додаткова заробітна плата	588.00	5.14
Відрахування на ЄСВ	1422.96	12.44
Витрати на матеріали	357.00	3.12
Витрати на оплату машинного часу	427.52	3.73
Інші витрати	405.26	3.54
Накладні витрати	2352.00	20.57
Разом витрати	11432.74	100.00
Прибуток (30%)	3429.82	—
Всього	14862.56	—
ПДВ (20%)	2972.51	—
Всього	17835.07	—

6.4 Оцінка економічної ефективності

Надходження від продажу повинні складати 56920 гривень, з урахування ризику 50% надходження складатиме 28460 гривень. З огляду на те, що очікуване надходження від продажів вище, ніж ціна розробки, в якій врахована рентабельність 30%, можна зробити висновок про економічну доцільність розробки.

7 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

7.1 Аналіз потенціальних небезпек

Характер роботи багато в чому впливає на стан здоров'я користувача комп'ютера. Небезпечним називається виробничий фактор, вплив якого на працівника в певних діях призводить до травми або погіршення стану здоров'я. Якщо виробничий фактор призводить лише до захворювання або зниженню працездатності, то його називають шкідливим. Залежно від рівня та тривалості впливу шкідливий фактор може стати небезпечним. Небезпечні та шкідливі фактори розподіляються на чотири групи: фізичні, хімічні, біологічні та психофізичні. Можна виділити основні потенційні небезпеки при роботі з комп'ютером:

- небезпека ураження електричним струмом, внаслідок недотримання правил електробезпеки або виходу з ладу електроприладів;
- негативний вплив електромагнітних полів, які створюються офісним обладнанням, на організм працюючого;
- негативний вплив на опорно-руховий апарат тривалим перебуванням працюючого в одній позі без активних рухів.
- незадовільній ергономічності робочого місця та неправильній організації робочого часу;
- негативний вплив недостатнього освітлення робочої зони на зір та продуктивність роботи працюючого, внаслідок несправності освітлювальних приладів або неправильного проектування освітлювальної системи;
- негативний вплив на зір через тривалу роботу з об'єктами малого розміру;
- негативний вплив незадовільних параметрів повітряного середовища робочої зони на здоров'я працюючого, внаслідок неправильного проектування системи вентиляції або несправності її несправності;

- негативний вплив підвищеного рівня шуму на психоемоційний стан працюючого, який пов'язаний з використанням застарілої периферійної техніки, кондиціонерів, копіювальної техніки, освітлювальних приладів;

- небезпека загоряння у зв'язку із несправністю електричного обладнання, недотримання, або порушення правил протипожежної безпеки обслуговуючим персоналом, що може призвести до пожежі.

Серед основних небезпек можна виділити наступні навантаження:

- інтелектуальні навантаження: необхідність швидкого прийняття рішень, творча діяльність, постійне сприйняття і оцінка нової інформації, високий ступінь складності завдання;

- емоційні навантаження: ступінь відповідальності за завдання для виконання, дефіцит часу, значимість помилки;

- монотонністю трудового процесу: багаторазове повторення одноманітних дій, тривалість зосередженого спостереження;

- небезпека виникнення пожеж, неправильні дії персоналу у надзвичайних ситуаціях.

Для користувачів можуть бути виявлені наступні небезпечні чинники при роботі за комп'ютером:

- мерехтіння екрану або періодична зміна його яскравості;
- нестабільність зображення на екрані;
- підвищена зовнішня освітленість екрану;
- аномальний рівень освітленості робочої зони;
- підвищена нерівномірність розподілу яскравості в полі зору користувача та підвищені пульсації світлового потоку джерел світла;

- підвищена яскравість фрагментів світлового зображення або світла, що потрапляє в поле зору користувача.

Статистичні дані показують, що основними і найбільш часто виникаючими захворюваннями при роботі з комп'ютером є:

- запалення очей, глаукома, катаракта, різь в очах;
- головний біль;

- хвороби серцево-судинної системи;
- біль у хребті, біль при русі рук, болі в області м'язів обличчя і шиї;
- свербіж шкіри, дерматити, лущення;
- нервово-психічні розлади.

Для створення сприятливих умов роботи з комп'ютером необхідна відповідність нормативно-правовим актам з охорони праці, які регламентують показники стану навколишнього середовища:

- достатню площу і об'єм приміщення, де встановлено комп'ютер;
- показники мікроклімату робочого приміщення (температура в межах 22-24°C, відносна вологість не вище 75%, та ін.);
- параметри світлового середовища робочого місця і робочої поверхні (наявність природного і штучного освітлення, освітленість екрана монітора, освітленість горизонтальної поверхні столу, відсутність прямого і відбитого сліпучого відблиску джерел світла, відсутність пульсації освітленості);
- рівень шуму і вібрації робочих місць в межах норм;
- відсутність в повітрі робочої зони шкідливих речовин.

Нижче розглядаються заходи з охорони праці, виробничої санітарії, гігієни праці та пожежної безпеки при експлуатації ЕОМ з візуальними дисплейними терміналами (ВДТ) в будівлях і приміщеннях.

7.2 Заходи із забезпечення безпеки

Забезпечення безпеки праці, організації оптимальних умов праці, а також захисту навколишнього середовища в умовах роботи з персональним комп'ютером (ПК) досягається шляхом дотримання норм і проведення заходів, передбачених законодавчими актами України. З метою зменшення негативного впливу ПК на організм людини враховані вимоги при організації роботи [39].

В першу чергу було створено необхідні умови для роботи з комп'ютером в спеціальному приміщенні:

- вологість повітря – не вище 75%;

- температура в межах 22-24°C;
- відсутність струмопровідного пилу і електробезпеки підлогового покриття.

Приміщення з візуальними дисплейними терміналами (ВДТ) обладнано аптечкою першої медичної допомоги. Комп'ютери розташовані в зоні недосяжності металевих предметів: батареї опалення, трубопроводів.

Приміщення комп'ютерної лабораторії відноситься до приміщень без підвищеної небезпеки ураження електричним струмом, тобто там відсутні умови, що створюють особливу або підвищену небезпеку

У приміщенні відсутні відкриті струмопровідні частини з напругою понад 12В. При виконанні правил ПУЕ «Правила улаштування електроустановок» ураження електричним струмом можливо тільки в разі несправності апаратури і живлячих кабелів.

Обладнання, яке живиться від змінного струму 220В – небезпечного для життя, вимагає забезпечення електробезпеки. Таким чином, щоб уникнути ураження електричним струмом все електрообладнання на робочому місці заземлено [40]. Захисне заземлення виконано навмисним електричним з'єднанням металевих частин електроустановок з «землею» або її еквівалентом. Електрична ізоляція забезпечується конструкціями пристроїв і способами їх підключення. В процесі експлуатації періодично проводиться перевірка якості ізоляції, шляхом вимірювання її опору. Відповідно до ПУЕ в електроустановках до 1000 В мінімальне значення опору ізоляції силових і освітлювальних електропроводок на ділянці між суміжними запобіжниками або за останніми запобіжниками між будь-яким проводом і землею становить не менше 0,5 МОм.

Відповідно до ПУЕ захисне заземлення застосовується в усіх електроустановках змінного струму під напругою 380В і вище і постійного струму 440В і вище. Захисному заземленню підлягають корпуси трансформаторів, оболонки кабелів, металеві огорожі електроустановок [40].

Заземлюючі пристрої, що використовуються для заземлення електроустановок відповідають всім вимогам, що пред'являються до заземлення цих електроустановок.

Матеріал, конструкція і розміри заземлювачів забезпечують стійкість до механічних, хімічних і термічних дій на весь період експлуатації.

Для вирівнювання потенціалів металеві будівельні та виробничі конструкції приєднані до мережі заземлення.

Зменшення напруги електроустановки істотно підвищує умови електробезпеки. Так при напрузі 10-12 В і електричному опорі тіла людини 5-6 кОм сила струму, що протікає через людину, не перевищить 2 мА [40].

Заходи щодо усунення небезпеки ураження електричним струмом зводяться до правильного розміщення устаткування та електричних кабелів.

За способом захисту людини від ураження електричним струмом приміщення відповідає згідно до ГОСТ 12.2.007.0-75* (2001) ССБТ «Вироби електротехнічні. Загальні вимоги безпеки» I (стаціонарні комп'ютери,) та II (освітлювальні прилади, кондиціонери, опалювальні пристрої, ноутбуки, сканери) класам захисту.

В якості профілактичних заходів використано скриту електромережу, надійні розетки з пожежобезпечних матеріалів, силові мережі живлення устаткування були виконані кабелями, що розраховані на підключення в 3-5 разів більшого навантаження, включення та виключення живлення обладнання виконано за допомогою штатних вимикачів. Над кожною розеткою є помітка «220В». Регулярно проводиться очистка внутрішніх частин комп'ютерів від пилу, комп'ютери розташовані на окремих неспалюваних столах. Для запобігання іскріння рідше встромляються і виймаються штепсельні вилки з розеток. Також працюючи з комп'ютером не допускається попадання будь-якої рідини на корпус та інші прилади ПК, не кладуться на комп'ютерну техніку сторонні предмети та не закриваються вентиляційні отвори.

Індикаторні лампи в блоках ЕОМ сигналізують про включення блоку в мережу.

Дотримання правил та вимог дозволяє максимально забезпечити безпеку користувача від ураження електричним струмом.

Згідно норм [41] неприпустимо розташування приміщень, призначених для роботи з ВДТ в підвалах і цокольних поверхах. Також забороняється розміщення вибухонебезпечних [42] приміщень категорії А і Б та виробництв з мокрими технологічними процесами поряд з приміщенням, де розташовуються ЕОМ, а також над такими приміщеннями, або під ними. Крім того, приміщення для роботи з ВДТ не повинні межувати з приміщеннями, в яких рівень шуму і вібрації перевищує допустимі значення [43].

Відповідно до вимог виконані основні вимоги до моніторів, які жорстко регламентують безпечні умови роботи і захист здоров'я осіб, що працюють з комп'ютерами:

- символи на екрані чіткі і добре розрізняються;
- зображення позбавлене блимання;
- яскравість та / або контрастність легко регулюються;
- екрани вільні від відблисків і відбиття;
- випромінювання знижені до надзвичайно малих рівнів.

Дотримання правил та вимог дозволить максимально забезпечити безпеку працівників, а також надати максимально комфортні умови праці.

7.3 Заходи з забезпечення виробничої санітарії та гігієни праці

Згідно до вимог [44] передбачено гігієнічні вимоги до об'ємно-планувальних рішень будівель та приміщень для роботи з ВДТ ЕОМ. Розміщення робочих місць з ВДТ ЕОМ в підвальних приміщеннях, на цокольних поверхах заборонено.

В обчислювальному центрі дотримуються норми – в приміщенні використано природне та штучне освітлення. Відносно природного освітлення: здійснено через світлові прорізи, які забезпечують коефіцієнт природної освітленості (КПО) не нижче 1,5%. Для захисту від прямих сонячних променів,

які створюють прямі та відбиті відблиски на поверхні екранів, на вікнах встановлені ролети. Віконні отвори приміщень обладнані регульованими пристроями. Для внутрішнього оздоблення приміщень використовують дифузно-відбивні матеріали з коефіцієнтами відбиття для стелі 0.7-0.8, для стін 0.5-0.6.

Штучне освітлення в приміщенні здійснено за допомогою системи загального рівномірного освітлення. Освітленість поверхонь робочих місць в горизонтальній площині на рівні розміщення документів, клавіатур становить 300-400лк, а коефіцієнт пульсації освітленості – не більше 10%. Джерелами штучного освітлення в приміщенні забезпечуються розташовані на стелі світильники з люмінесцентними лампами типу ЛБ.

Під час роботи з комп'ютером існують фактори психофізіологічної форми – перенапруження органів зору – користувач кидає близько 30000 поглядів на екран. Для зниження навантаження на органи зору користувача при роботі з комп'ютером дотримуються такі умови зорової роботи: при роботі на ПК користувач виконує роботу високої точності, при мінімальному розмірі об'єкта розрізнення 0.3-0.5мм – товщина символу на екрані, розряду роботи III, де екран – фон світлий, символ – об'єкт розрізнення – темний або навпаки.

До потенційних шкідливих факторів також відноситься шум. Тривале перебування під впливом шуму викликає зниження працездатності, прискорює розвиток зорового стомлення, змінює кольоровідчуття, підвищує витрату енергії, може поступово призвести до погіршення слуху і до глухоти. Тривале знаходження людини в зоні комбінованого впливу різних несприятливих факторів може призвести до професійного захворювання. Основними заходами боротьби з шумом є усунення або ослаблення причин шуму в самому його джерелі у процесі проектування, використання засобів звукопоглинання, раціональне планування виробничих приміщень. Звукоізоляція огорожувальних конструкцій приміщення з ВДТ забезпечує параметри шуму, що відповідають вимогам [43].

Згідно з ГОСТ 26329-84 зниження рівня шуму в приміщенні здійснено за допомогою:

- використання більш сучасного обладнання;
- розташування принтерів та різноманітного устаткування колективного користування на значній відстані від більшості робочих місць працівників;
- переведення жорсткого диска в режим сну (Standby), якщо комп'ютер не працює протягом визначеного часу;
- використання блоків живлення ПК з вентиляторами на гумових підвісках.

Нормовані параметри мікроклімату, іонного складу повітря, вмісту шкідливих речовин відповідають вимогам [44].

У приміщенні забезпечений приплив свіжого повітря, кількість якого визначається техніко-економічним розрахунком і вибором схеми системи вентиляції.

При невеликому забрудненні зовнішнього повітря кондиціонування приміщень здійснюється із перемінними витратами зовнішнього повітря й рециркуляційного.

Роботи в комп'ютерних офісах належать до категорії Іб – легка робота, тому встановлені наступні оптимальні значення параметрів мікроклімату:

- у холодний період року: температура 22-24°C; відносна вологість: 40-60%;
- у теплий період року: температура 23-25°C; відносна вологість: 40-60%.

В нормах передбачені гігієнічні вимоги до організації обладнання робочих місць ВДТ ЕОМ. Конструкція робочого столу повинна відповідати сучасним вимогам ергономіки і забезпечувати оптимальне розміщення на робочій поверхні використовуваного обладнання (дисплея, клавіатури, принтера) і документів. Висота робочої поверхні робочого столу з ВДТ повинні регулюватися в межах 680-800 мм, а ширина і глибина – забезпечувати

можливість виконання операцій в зоні досяжності моторного поля (рекомендовані розміри: 600-1400 мм, глибина – 800-1000 мм). Приймаємо робочу поверхню з розмірами 1400×800. Щоб уникнути відблисків на екрані вікон доцільно розмістити робоче місце торцем до вікна не ближче ніж на 1 м. Площа дозволяє розмістити додаткові меблі для зберігання документів.

Для підвищення продуктивності праці, а також зниження рівня стомлюваності при роботі з ЕОМ передбачені десятихвилинні перерви через кожну годину.

Велике значення має задоволення обстановки ергономічним і естетичним вимогам. Цього домоглися за допомогою колірної рішення приміщення і обладнання обчислювального центру з урахуванням вимог [45] – в приміщеннях стіни пофарбовані фарбами пастельних теплих тонів.

Під час організації робочого простору враховано індивідуальні антропометричні параметри користувача з відповідними допусками на можливі зміни робочих поз та потребу у переміщеннях.

Проведено заходи мінімізації негативного впливу випромінювання від відео дисплейних терміналів на органи зору.

При організації роботи для збереження здоров'я, запобігання професійних захворювань і підтримки працездатності передбачені перерви для відпочинку. Тривалість обідньої перерви визначається чинним законодавством про працю і правилами внутрішнього трудового розпорядку підприємства.

Для зниження нервово-емоційного напруження, втомлення зорового аналізатору, поліпшення мозкового кровообігу, подолання несприятливих наслідків гіподинамії, запобігання втомі передбачено такі режими праці і відпочинку при роботі з комп'ютером:

- для розробників програм із застосуванням ЕОМ призначений регламентований перерву для відпочинку тривалістю 15 хвилин через кожну годину роботи на ВДТ;
- для операторів призначені регламентовані перерви тривалістю 15 хвилин через кожні дві години роботи;

– для операторів комп'ютерного набору призначені регламентовані перерви для відпочинку тривалістю 10 хвилин після кожної години роботи на ВДТ.

Під час роботи відстань до екрану монітора є не менш 70 см. Робоче місце є комфортним та достатньо освітленим.

У всіх випадках, коли виробничі обставини не дозволяють застосувати регламентовані перерви, тривалість безперервної роботи з ВДТ не перевищувала 4 години.

Всі перелічені вище заходи проведені з метою надання максимально комфортних умов праці і сприяє підвищенню ефективності праці.

7.4 Заходи безпеки у надзвичайних ситуаціях.

7.4.1 Заходи з пожежної безпеки

Закон України «Про пожежну безпеку» визначає загальні правові, економічні та соціальні основи забезпечення пожежної безпеки на території України, регулює відносини державних органів, юридичних і фізичних осіб у цій галузі незалежно від виду їх діяльності та форм власності. Пожежна безпека – стан об'єкта, при якому з регламентованою ймовірністю виключається можливість виникнення та розвиток пожежі і впливу на людей її небезпечних факторів, а також забезпечується захист матеріальних цінностей.

В приміщенні комп'ютерного центру джерелом загоряння можуть бути електронні схеми ПК, прилади для технічного обслуговування, електроустаткування, кондиціонери повітря, де за рахунок різних порушень відбувається перегрів тих чи інших елементів, електричні іскри, дуга, що можуть визвати загоряння горючих матеріалів, дерев'яні та пластикові стільці та столи, папір, перевантаження мережі живлення, людські чинники тощо.

Для забезпечення пожежної безпеки в установах проводиться пожежна профілактика, яка включає в себе комплекс організаційних і технічних заходів,

спрямованих на забезпечення безпеки людей, на запобігання пожежі, обмеження її поширення, а також на створення умов для успішного гасіння пожежі.

Заходи щодо протипожежного захисту розділяються на організаційні, експлуатаційні, технічні й режимні (спеціальні).

Організаційні заходи: навчання робітників та службовців правилам пожежної безпеки, організація пожежної охорони, проведення бесід, лекцій, видання необхідних інструкцій, плакатів і т.п.

Технічні заходи передбачають дотримання протипожежних правил і норм при устрої систем опалення, вентиляції, кондиціонуванні повітря, блискавокозахисті при спорудженні будинків, установці технологічного устаткування й ін.

Експлуатаційні заходи передбачають правильну експлуатацію систем опалення, вентиляції й кондиціонування повітря, блискавокозахисту технологічних машин і устаткування, правильне утримання будинків і територій і т.п.

Режимні заходи передбачають заборону або обмеження застосування відкритого вогню в вогнебезпечних місцях, паління в невстановлених місцях, обов'язкове дотримання норм і правил при роботі з вогнебезпечними й вибухонебезпечними речовинами.

До первинних засобів гасіння пожеж, призначених для локалізації невеликих загорянь, належать пожежні цівки, внутрішні пожежні водопроводи, вогнегасники, сухий пісок, азбестові ковдри й т.п. Згідно норм [47] при захисті від пожежі приміщення з ПК використовуються вуглекислотні вогнегасники ВВК-1,4, ВВК-2 або аерозольні водопінні вогнегасники ВВПА-400, можливо також використання порошкових вогнегасників. Згідно пожежних норм в громадських та адміністративних приміщеннях на кожному поверсі встановлено два та більше вогнегасники з масою заряду речовини для гасіння пожежі не менше 5 кг. Згідно зі стандартами приміщення з ПК оснащено переносними вуглекислотними вогнегасниками з розрахунку один вогнегасник ВВК-1,4 або ВВК-2 або ВВПА-400 на три ПК, але не менше ніж один вогнегасник, перелічених типів на приміщення. За наявності вогнегасників в приміщенні

пожежу можна погасити в перші чотири хвилини після займання, задовго до прибуття пожежних.

Згідно стандартів [46], Для подачі сигналу про пожежу встановлені пожежні сповіщувачі. Приміщення обчислювального центру обладнано автоматичними пожежними сповіщувачами, що реагують на підвищення температури, дим, полум'я, наприклад, сповіщувачі моделей ДТЛ, ІТМ.

На випадок пожежі передбачені шляхи евакуації працюючих – проходи, проїзди, евакуаційні виходи згідно до стандартів [46].

Евакуаційні виходи розташовуються розосереджено, а їх кількість – не менше двох на споруду. При кількості працюючих у зміні до 50 чоловік передбачений другий (запасний вихід) на зовнішні сходи.

7.4.2 Заходи з цивільного захисту. Порядок дій сил цивільного захисту при ліквідації наслідків стихійних лих

Природні стихійні лиха – це небезпечні природні явища, процеси атмосферного, гідрологічного, геологічного, біосферного або іншого походження таких масштабів, які призводять до катастрофічних ситуацій з раптовим порушенням систем життєдіяльності населення, руйнуванням і знищенням матеріальних цінностей, об'єктів народного господарства, що у свою чергу може спричинити аварії й катастрофи.

Кожне природне стихійне лихо має свої причини виникнення, особливості впливу на навколишнє середовище, фізичну суть і рушійні сили. Знаючи характер стихійних лих, причини їх виникнення, можна завчасно вжити заходів і тим самим запобігти деяким з них або значно зменшити їх руйнівний вплив, спланувати правильні дії населення для проведення рятувальних робіт.

Дії суб'єктів забезпечення цивільного захисту щодо реагування на НС повинні бути скоординовані та здійснюватися відповідно до планів реагування на НС, уточнених для умов конкретного виду та рівня НС.

Рятувальні та невідкладні роботи при ліквідації наслідків стихійного лиха і аварії проводять рятувальні формування, а також залучене працездатне населення. Роботи, які потребують спеціальної підготовки і залучення потужної техніки, виконують інженерні загони, створені на базі будівельних, ремонтно-будівельних, будівельно-монтажних, шляхово-будівельних організацій і відділів комунального господарства. При стихійних лихах, які не є катастрофічними, всі заходи організовують керівники районів, об'єктів, населених пунктів і начальники загонів служби порятунку. Начальник цивільного захисту району, населеного пункту чи об'єкта приймає рішення про проведення рятувальних і невідкладних робіт, в якому визначає: де зосередити основні зусилля, завдання рятувальних загонів і порядок введення їх на об'єкти робіт, початок і тривалість робіт, управління силами, які ведуть роботи, і порядок забезпечення їх дій. Рятування людей і матеріальних цінностей є основним завданням при стихійних лихах. Послідовність виконання цього завдання залежить від характеру лиха, його наслідків, наявності та підготовки сил цивільного захисту, періоду року, стану погоди та інших факторів. У завчасно розроблених планах ЦЗ на об'єктах господарювання визначають порядок приведення у готовність і дію рятувальних формувань при стихійних лихах. Плани складають на основі прогнозування можливих стихійних лих.

Ліквідація наслідків будь-яких стихійних лих полягає в:

- оповіщенні формувань і населення про небезпеку стихійного лиха;
- організації управління силами ЦЗ в районі лиха;
- веденні розвідки, встановленні ступеня і величини руйнувань, затоплення, зараження, пожеж та інших даних;
- виявленні об'єктів і населених пунктів, яким загрожують наслідки стихійного лиха;
- визначенні складу, чисельності сил і засобів, які залучаються для рятувальних робіт;
- організації медичної допомоги потерпілим і евакуації їх у лікувальні заклади, виведенні населення в безпечні місця та його розміщення;

- забезпеченні заходів безпеки при веденні рятувальних та інших невідкладних робіт;
- організації комендантської служби і підтримання громадського порядку в районі лиха;
- організації матеріального, технічного і транспортного забезпечення;
- проведенні інших заходів, спрямованих на підготовку і забезпечення рятувальних робіт, ліквідацію наслідків стихійного лиха.

Завдання для розвідки ставить начальник цивільного захисту об'єкта. Він визначає мету розвідки, які відомості та на який час необхідно одержати, де і на виконання яких завдань слід зосередити основні зусилля, які для цього використати сили і засоби.

У районі стихійних лих розвідка визначає:

- межі району лиха і напрямки його поширення;
- об'єкти і населені пункти, яким загрожує небезпека;
- місця знаходження людей і ступінь загрози їм; шляхи підходу формувань і техніки до місць роботи;
- стан пошкоджених будівель і споруд;
- наявність у них потерпілих, яким необхідна допомога у першу чергу;
- місця аварій на технологічних лініях, комунально-енергетичних мережах і розміри руйнувань на них;
- обсяги робіт, умови їх ведення;
- можливість використання техніки;
- найбільш зручні місця для розбирання завалів і звільнення потерпілих, а також для прокладання шляхів їх евакуації;
- стан вододжерел і можливість використання їх для господарських, питних потреб і гасіння пожеж;
- умови і доцільну черговість ведення робіт.

Зібрані дані про характер і обсяг майбутніх робіт розвідники доповідають начальнику в намічені строки.

Населення потрібно інформувати про режими поведінки. Інформація має бути чіткою, ясною, а в завданні на проведення робіт слід конкретно зазначити: хто, коли і що повинен зробити. Від цього буде залежати успіх проведення рятувальних і невідкладних робіт у районі небезпечного стихійного лиха. При спільних діях особового складу формувань цивільного захисту і населення, при високій дисципліні та організованості населення можна значно зменшити наслідки стихійного лиха.

7.5 Висновки з розділу

В ході написання розділу були наведені потенційні небезпеки на робочому місці, в робочій зоні і у виробничому середовищі відповідно до виду робіт.

Були обрані найбільш ефективні заходи, спрямовані на виключення або мінімізацію негативного впливу певних шкідливих факторів виробничого процесу виходячи з наведених даних про небезпеки на робочому місці. Виходячи з цього були розроблені заходи щодо забезпечення виробничої санітарії та гігієни праці, виробничих факторів санітарно-гігієнічного характеру, розроблені поліпшуючі заходи, приведені необхідні розрахунки.

Було описано заходи щодо пожежної безпеки, що спрямовані на збереження життя персоналу, обладнання, матеріальних ресурсів в умовах виникнення загорянь, організації ліквідації загорянь, а також евакуації персоналу.

Також було описано порядок дій сил цивільного захисту при ліквідації наслідків стихійних лих.

ВИСНОВКИ

В процесі виконання кваліфікаційної роботи магістра було досліджено сучасні системи дистанційного навчання та способи інтеграції навчального контенту і додаткових інструментів до них. Було розглянуто альтернативні варіанти для впровадження в навчальний процес, їх недоліки та переваги, стандарти організації перевикористовуваних матеріалів, функціональні можливості та налаштування процесу інтеграцію. Як альтернативу системам дистанційного навчання було розглянуто масові відкриті онлайн курси. Більш детально було проаналізовано засоби інтеграції, протоколи та стандарти для створення і дордавання контенту і власних інструментів до навчальних курсів.

На основі проведених досліджень було спроектовано програмний засіб що може інтегруватись до систем дистанційного навчання за LTI протоколом і, в результаті підтримки розглянутих стандартів, може бути перевикористаним в інших системах та розширюватись виходячи з потреб користувача. В результаті дослідницької та практичної роботи було обрано набір найбільш відповідних технологій та інструментів для реалізації даного застосунку виходячи з поставленого завдання та потреб середовища виконання даного роду програмних засобів. Готова програма має перевагу над специфічними для початкових платформ плагінами у вигляді можливості інтеграції до будь-яких LTI сумісних систем і значно менших вимогах до системних ресурсів ніж аналоги.

Практична значимість роботи полягає в можливості додавання власних веб-застосунків до вище зазначених систем дистанційного навчання та створенні програм з будь-яким функціоналом опираючись на обмеженнях LTI протоколу. Даний підхід дозволяє створювати програмні продукти до вже готових і налаштованих систем без змін до їх кодової бази. Також значною перевагою даного методу є можливість перевикористання застосунку та абсолютно вільний вибір технологічної бази, можливість розділення елементів системи на менші компоненти.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Top Learning Management System (LMS) Statistics For 2020 You Need To Know [Electronic resource]. Access mode : <https://elearningindustry.com/top-learning-management-system-lms-statistics-for-2020-infographic/>.
2. Навчальні заклади, які надають можливість навчатись віддалено [Електрон. ресурс]. – Режим доступу : <https://mon.gov.ua/ua/osvita/visha-osvita/distancijna-osvita/navchalni-zakladi-yaki-nadayut-mozhlivist-navchatis-viddaleno/>.
3. Дистанційна освіта [Електрон. ресурс]. – Режим доступу : <http://vnz.org.ua/dystantsijna-osvita/pro/>.
4. Increase the Power of Learning Management Systems with Integration [Electronic resource]. – Access mode : <https://www.mulesoft.com/resources/cloudhub/integration-lms-management-system/>.
5. Learning management system [Electronic resource]. – Access mode : https://en.wikipedia.org/wiki/Learning_management_system.
6. What Is A Learning Management System? LMS Basic Functions And Features You Must Know [Electronic resource]. – Access mode : <https://elearningindustry.com/what-is-an-lms-learning-management-system-basic-functions-features>.
7. Moodle [Електрон. ресурс]. – Режим доступу : <https://uk.wikipedia.org/wiki/Moodle>.
8. Moodle dashboard [Electronic resource]. – Access mode : https://moodle.org/pluginfile.php/50/local_plugins/plugin_screenshots/1444/KTree_Dashboard_Home.png.
9. Що таке Moodle [Електрон. ресурс]. – Режим доступу : <https://moodle.org/mod/page/view.php?id=8174/>.
10. E-learning methodologies. A guide for designing and developing e-learning courses [Electronic resource]. – Access mode : <http://www.fao.org/3/i2516e/i2516e.pdf/>.

11. Docebo dashboard [Electronic resource]. – Access mode : <https://i.pinimg.com/originals/92/44/e3/9244e3183e71238fc775d5cbb933a941.png/>.
12. Efront dashboard [Electronic resource]. – Access mode : https://software-advice.imgix.net/managed/products/screenshots/1__efront-admin-dashboard.png?auto=format&w=600&h=450/.
13. Dokeos dashboard [Electronic resource]. – Access mode : [https://lh3.googleusercontent.com/proxy/D2EaJWuTfJYJm4o_kQKUoePPbpUy7xqY2p0o8KrEKA6NtVDQJFrb7kLWJIDZGAtgVlsTzHk5SBReFyOEU8rEmdhPHfS3Eamofj1lVAb1eb-xH8SHN3qSj77GuD0DLihMUGTbj0 /](https://lh3.googleusercontent.com/proxy/D2EaJWuTfJYJm4o_kQKUoePPbpUy7xqY2p0o8KrEKA6NtVDQJFrb7kLWJIDZGAtgVlsTzHk5SBReFyOEU8rEmdhPHfS3Eamofj1lVAb1eb-xH8SHN3qSj77GuD0DLihMUGTbj0/).
14. Claroline dashboard [Electronic resource]. – Access mode : <https://claroline.gitbooks.io/translating-claroline-connect/content/images/dashboard.png/>.
15. ATutor dashboard [Electronic resource]. – Access mode : <https://www.opensourcecms.com/wp-content/uploads/ATutorEN.png/>.
16. ILIAS dashboard [Electronic resource]. – Access mode : <https://www.predictiveanalyticstoday.com/wp-content/uploads/2017/10/ILIAS-1000x497.jpg/>.
17. .LRN dashboard [Electronic resource]. – Access mode : <https://cdn.elearningindustry.com/wp-content/uploads/2018/08/aaff0a8259233c2d2856dc9ee7338e2d.png/>.
18. Sakai CLE dashboard [Electronic resource]. – Access mode : <https://confluence.sakaiproject.org/download/attachments/81758121/Screen%20Shot%202012-06-08%20at%209.28.03%20AM.png?version=1&modificationDate=1343913992000&api=v2/>.
19. Massive open online course [Electronic resource]. – Access mode : https://en.wikipedia.org/wiki/Massive_open_online_course/.
20. The standard SCORM and its application [Electronic resource]. – Access mode : <http://cccp.ifmo.ru/scorm/index.html/>.

21. How do I create SCORM content? [Electronic resource]. – Access mode: http://doc.claroline.net/en/index.php/How_do_I_create_SCORM_content%3F/.
22. Icodeon SCORM Player Version 2 (Java Edition) [Electronic resource]. – Access mode : http://www.icodeon.com/release/download/j2ee/release_2p0p4/docs/.
23. Learning Tools Interoperability – Новий стандарт інтеграції для платформ дистанційного навчання [Електрон. ресурс]. – Режим доступу : [https://www.researchgate.net/publication/331467546_LEARNING_TOOLS_INTEROPERABILITY_-
_NOVIJ_STANDART_INTEGRACII_DLA_PLATFORM_DISTANCIJNOGO_NAVCANNA](https://www.researchgate.net/publication/331467546_LEARNING_TOOLS_INTEROPERABILITY_-_NOVIJ_STANDART_INTEGRACII_DLA_PLATFORM_DISTANCIJNOGO_NAVCANNA).
24. Learning Tools Interoperability [Електрон. ресурс]. – Access mode : https://en.wikipedia.org/wiki/Learning_Tools_Interoperability.
25. Інтеграція за протоколом LTI [Електронний ресурс]. – Режим доступу : [https://support.stepik.org/hc/ru/articles/360010307320-
%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B0%D1%86%D0
%B8%D1%8F-%D0%BF%D0%BE-
%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB%
D1%83](https://support.stepik.org/hc/ru/articles/360010307320-%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B0%D1%86%D0%B8%D1%8F-%D0%BF%D0%BE-%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB%D1%83).
26. Connecting IMS Learning Tools Interoperability and SAML [Електронний ресурс]. – Access mode : [http://www.dr-chuck.com/csev-
blog/2012/03/connecting-ims-learning-tools-interoperability-and-saml/](http://www.dr-chuck.com/csev-blog/2012/03/connecting-ims-learning-tools-interoperability-and-saml/).
27. Installing Moodle [Electronic resource]. – Access mode : https://docs.moodle.org/310/en/Installing_Moodle/.
28. Moodle Architecture [Electronic resource]. – Access mode : https://docs.moodle.org/dev/Moodle_architecture.
29. Docker [Electronic resource]. – Access mode : [https://en.wikipedia.org/wiki/Docker_\(software\)/](https://en.wikipedia.org/wiki/Docker_(software)).
30. Overview of Docker Compose [Electronic resource]. – Access mode : <https://docs.docker.com/compose/>.

31. Node.js [Electronic resource]. – Access mode : <https://en.wikipedia.org/wiki/Node.js/>.

32. What is TypeScript? [Electronic resource]. – Access mode : <https://www.typescriptlang.org/>.

33. Flow is a static checker for javascript [Electronic resource]. – Access mode : <https://flow.org/>.

34. Express.js [Electronic resource]. – Access mode : <https://en.wikipedia.org/wiki/Express.js>.

35. Handlebars [Electronic resource]. – Access mode : <https://handlebarsjs.com/>.

36. Moodle release notes [Electronic resource]. – Access mode : https://docs.moodle.org/dev/Moodle_3.9_release_notes#Browser_support.

37. 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 /Укл.: Остапенко В.В., Сердюк Є.М.– Запоріжжя: ЗНТУ, 2015.– 22 с.

38. Єдиний соціальний внесок [Електрон. ресурс]. – Режим доступу : https://uk.wikipedia.org/wiki/%D0%84%D0%B4%D0%B8%D0%BD%D0%B8%D0%B9_%D1%81%D0%BE%D1%86%D1%96%D0%B0%D0%BB%D1%8C%D0%BD%D0%B8%D0%B9_%D0%B2%D0%BD%D0%B5%D1%81%D0%BE%D0%BA.

39. Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями. НПАОП 0.00-7.15-18. Державні санітарні правила і норми [Текст]. – [Чинний від 2018-02-14]. – Харків : Форт, 2018. – 48 с.

40. Електробезпека. Захисне заземлення, занулення: ГОСТ 12.1.030-81. – [Чинний від 1982-07-01]. – М.: Изд-во стандартов, 1982. – 10 с.

41. Правила охорони праці під час експлуатації електронно-обчислювальних машин : НПАОП 0.00-1.28-10. – [Чинний від 2010-03-26]. – К.: Державний комітет України з промислової безпеки, охорони праці та гірничого нагляду, 2010. – 8 с. – (Національний стандарт України).

42. Норми визначення категорій приміщень, будинків та зовнішніх установок по вибухопожежної та пожежної безпеки : НАПБ Б.03.002-2007. – [Чинний від 2007-12-03]. – К.: МНС України, 2007. – 9 с. – (Національний стандарт України).

43. ДСН 3.3.6.037-99 Санітарні норми виробничого шуму, ультразвуку та інфразвуку [Текст]. – [Чинний від 1999-12-01]. – К. : МОЗ України, 1999. – 24 с.

44. Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин: ДСанПіН 3.3.2.007-98. – К.: Держстандарт України, 1998. – 28 с. – (Державні санітарні правила і норми України).

45. ГОСТ 17.2.032-78 Рабочее место при выполнении работ сидя. Система стандартов безопасности труда [Текст]. – [Введен с 1979-01-01]. – М. : ИПК Издательство стандартов, 2001. – 7 с.

46. Протипожежна техніка. Терміни та визначення основних понять: ДСТУ 2273 – 2006. – К.: Держстандарт України, 2006. – 44 с. – (Національний стандарт України).

47. Типові норми належності вогнегасників: НАПБ Б.03.001-2004. – 2004. – К.: Держстандарт України, 2004. – 15 с. – (Національний стандарт України).

ДОДАТОК А
Текст програми

```

// ApplicationConfig.ts
import * as path from "path";

export const applicationConfig = {
  PORT: process.env.PORT || 3000,
  VIEWS_DIR: path.resolve(__dirname, "../views/"),
  LAYOUTS_DIR: path.resolve(__dirname, "../views/layouts")
};

// LtiConfig.ts
export const ltiConfig = {
  SECRETS: {
    demo: "xzc342AScx",
    demo2: "dh43fvL-ew"
  } as { [index: string]: string }
};

// code.ts
import { default as express } from "express";

import { code } from "@store/code";

export const router = express.Router();

router.post("/code", (req, res, next) => {
  const session: { [index: string]: any } = req.session;
  const userId: string | undefined = session.userId;
  const uploadedCode: string = req?.body["code"];

  if (!req.body) {
    return next(new Error("Expected a body")); // todo: Status for proper 403.
  }

  if (userId) {
    code.set(userId, { userId, date: new Date(), text: uploadedCode }); // todo: Validation.

    res.json({ status: "OK" });
  } else {
    throw new Error("Cannot participate, user is not logged in");
  }
});

router.post("/remove_code", (req, res, next) => {
  const session: { [index: string]: any } = req.session;
  const userId: string | undefined = session.userId;

  if (userId) {
    code.delete(userId); // todo: Validation.

    // Redirect to actual handling of LTI after POST handling.
    res.redirect("/application?type=code");
  } else {
    throw new Error("Cannot participate, user is not logged in");
  }
});

// lti.ts
import { default as express } from "express";

import { ltiConfig } from "@configs/LtiConfig";

// eslint-disable-next-line @typescript-eslint/no-var-requires
const lti = require("ims-lti");

const nonceStore = new lti.Stores.MemoryStore();

export const router = express.Router();

function getConsumerSecret(consumerKey: string): string {

```

```

    if (ltiConfig.SECRETS[consumerKey]) {
      return ltiConfig.SECRETS[consumerKey];
    } else {
      throw new Error("Unknown consumer key provided");
    }
  }
}

/**
 * Handle LTI launch from external consumers.
 */
router.post("/launch_lti", (req, res, next) => {
  const consumerKey: string = req?.body["oauth_consumer_key"];
  const type: string = req.query.type as string;

  if (!req.body) {
    return next(new Error("Expected a body")); // todo: Status for proper 403.
  } else if (!consumerKey) {
    return next(new Error("Expected a consumer"));
  }

  if (!type) {
    return next(new Error("Expected application type from query"));
  }

  try {
    const consumerSecret: string = getConsumerSecret(consumerKey);
    const provider = new lti.Provider(consumerKey, consumerSecret, nonceStore,
lti.HMAC_SHA1);

    provider.valid_request(req, (error: Error, isValid: boolean) => {
      if (error) {
        return next(error);
      }

      if (isValid) {
        req.session.regenerate((err) => {
          if (err) {
            next(err);
          }

          const session: { [index: string]: any } = req.session;

          session.email = provider.body.lis_person_contact_email_primary;
          session.contextId = provider.context_id;
          session.userId = provider.userId;
          session.username = provider.username;
          session.ltiConsumer = provider.body.tool_consumer_instance_guid;
          session.isTutor = provider.instructor === true;

          return res.redirect(301, `/application?type=${type}`);
        });
      } else {
        // todo: Exception? When fires?
      }
    });
  } catch (error) {
    next(error);
  }
});

// participation.ts
import { default as express } from "express";

import { participants } from "@store/participation";

export const router = express.Router();

router.post("/participate", (req, res) => {
  const session: { [index: string]: any } = req.session;
  const userId: string | undefined = session.userId;

```

```

    if (userId) {
      participants.push({ userId, email: session.email, username: session.username });

      // Redirect to actual handling of LTI after POST handling.
      res.redirect("/application?type=participation");
    } else {
      throw new Error("Cannot participate, user is not logged in");
    }
  });

// views.ts
import { default as express, NextFunction, Request, Response } from "express";

import { code } from "@store/code";
import { feedbacks } from "@store/feedback";
import { participants } from "@store/participation";

export const router = express.Router();

function renderCodeApplication(req: Request, res: Response, next: NextFunction) {
  const session: { [index: string]: any } = req.session;

  if (session.userId) {
    const userCode = code.get(session.userId);

    const renderParams: { [index: string]: any } = {
      email: session.email,
      username: session.username,
      ltiConsumer: session.ltiConsumer,
      userId: session.userId,
      isTutor: session.isTutor,
      contextId: session.contextId,
      hasCode: Boolean(userCode),
      code: Array.from(feedbacks.values()),
      userCode: userCode
    };

    if (renderParams.isTutor) {
      return res.render("tutor-code-report", renderParams);
    } else {
      return res.render("student-code", renderParams);
    }
  } else {
    next(new Error("Session invalid. Please login via LTI to use this application."));
  }
}

function renderParticipationApplication(req: Request, res: Response, next: NextFunction) {
  const session: { [index: string]: any } = req.session;

  if (session.userId) {
    const renderParams: { [index: string]: any } = {
      email: session.email,
      username: session.username,
      ltiConsumer: session.ltiConsumer,
      userId: session.userId,
      isTutor: session.isTutor,
      contextId: session.contextId,
      hasParticipated: participants.some((it: { [index: string]: any }) => it.userId ===
session.userId),
      participants: participants
    };

    if (renderParams.isTutor) {
      return res.render("tutor-participation-report", renderParams);
    } else {
      return res.render("student-participation", renderParams);
    }
  } else {
    next(new Error("Session invalid. Please login via LTI to use this application."));
  }
}

```

```

    }
  }

  /**
   * Generic route for service status checking.
   */
  router.get("/", (req, res, next) => {
    return res.send({ status: "Up" });
  });

  /**
   * Generic route for service status checking.
   */
  router.get("/application", (req, res, next) => {
    const type: string = req.query.type as string;

    if (!type) {
      return next(new Error("Expected application type to be provided"));
    }

    if ((req?.session as any)?.userId) {
      switch (type) {
        case "participation":
          return renderParticipationApplication(req, res, next);

        case "code":
          return renderCodeApplication(req, res, next);

        default:
          return next(new Error("Expected type to be provided"));
      }
    } else {
      next(new Error("Session invalid. Please login via LTI to use this application."));
    }
  });

  // code.ts
  // In-memory storage instead of database.

  export interface ICode {
    text: string;
    userId: string;
    date: Date;
  }

  export const code: Map<string, ICode> = new Map();

  // participation.ts
  // In-memory storage instead of database.
  export const participants: Array<object> = [];

  // tutor-participation-report.handlebars
  <style>
    header {
      padding: 8px;
      width: 100%;
      border-bottom: 1px solid lightgray;
    }

    main {
      padding: 8px;
    }

    #context-info {
      padding: 8px;
      margin: 4px;
      border: 1px solid lightgray;
      border-radius: 4px;
    }
  </style>

```

```

#extension-info {
  padding: 8px;
  margin: 4px;
  border: 1px solid lightgray;
  border-radius: 4px;
}

table {
  padding: 8px;
  border: 1px solid lightgray;
}

td {
  border: 1px solid lightgray;
  padding: 4px;
}
</style>

<header>
  <nav>
    <h4> NodeJS LTI demo | 2020 </h4>
  </nav>
</header>

<main>
  <div id="context-info">
    <h3> Welcome, {{ username }} </h3>

    <div id="lti-info">
      <div> LTI Consumer: {{ ltiConsumer }} </div>
      <div> User ID: {{ userId }} </div>
      <div> Email: {{ email }} </div>
      <div> Context ID: {{ contextId }} </div>
    </div>
  </div>

  <div id="extension-info">
    {{#if participants.length }}
      <div>
        List of participated students ({{participants.length}}):
      </div>

      <table>
        {{#each participants}}
          <tr>
            <td> {{this.userId}} </td>
            <td> {{this.email}} </td>
            <td> {{this.username}} </td>
          </tr>
        {{/each}}
      </table>
    {{else}}
      <div>
        Still no one has participated in LTI test
      </div>
    {{/if}}
  </div>
</main>

// student-participation.handlebars
<style>
  header {
    padding: 8px;
    width: 100%;
    border-bottom: 1px solid lightgray;
  }

  main {
    padding: 8px;
  }

```

```

#context-info {
  padding: 8px;
  margin: 4px;
  border: 1px solid lightgray;
  border-radius: 4px;
}

#extension-info {
  padding: 8px;
  margin: 4px;
  border: 1px solid lightgray;
  border-radius: 4px;
}
</style>

<header>
  <nav>
    <h4> NodeJS LTI demo | 2020 </h4>
  </nav>
</header>

<main>
  <div id="context-info">
    <h3> Welcome, {{ username }} </h3>

    <div id="lti-info">
      <div> LTI Consumer: {{ ltiConsumer }} </div>
      <div> User ID: {{ userId }} </div>
      <div> Email: {{ email }} </div>
      <div> Context ID: {{ contextId }} </div>
    </div>
  </div>

  <div id="extension-info">
    {{#if hasParticipated }}
      <div>
        You are marked as participated in LTI test now.
      </div>
    {{else}}
      <div>
        You have not participated in LTI test.
        <br/>
        <br/>
        <form method="POST" action="/participate">
          <button> Participate </button>
        </form>
      </div>
    {{/if}}
  </div>
</main>

// student-code.handlebars
<style type="text/css" media="screen">
  main {
    height: 100%;
    display: flex;
    flex-direction: column;
  }

  #controls {
    padding: 16px;
    border-bottom: 1px black solid;
  }

  #display {
    position: relative;
    flex-grow: 1;
    padding: 4px;
  }

  #editor {

```

```

        position: absolute;
        top: 0;
        right: 0;
        bottom: 0;
        left: 0;
    }
</style>

<main>
    <div id="controls">
        <button id="send-button" class="btn btn-primary" disabled>
            {{#if hasCode }}
                Update my code
            {{else}}
                Send my code
            {{/if}}
        </button>
    </div>

    <div id="display">
        <div id="editor"></div>
    </div>
</main>

<script src="https://cdnjs.cloudflare.com/ajax/libs/ace/1.4.5/ace.js"
type="text/javascript" charset="utf-8"></script>

<script>
    const button = document.getElementById("send-button");

    let inputText = "";

    const editor = ace.edit("editor");

    editor.session.setMode("ace/mode/javascript");

    {{#if hasCode }}
        editor.insert(`{{ userCode.text }}`);
    {{/if}}

    editor.on("change", (e) => button.disabled = false);

    button.addEventListener("click", async () => {
        const body = { code: editor.getValue() };
        const response = await fetch("/code", { method: "POST", body:
JSON.stringify(body), headers: { "Content-Type": "application/json" } });

        location.href = "/application?type=code"
    });
</script>

// package.json
{
  "name": "nodejs-lti-provider",
  "version": "1.0.0",
  "description": "Example Node.js LTI provider",
  "main": "src/main.js",
  "scripts": {
    "start": "ts-node --project ./tsconfig.json -r tsconfig-paths/register ./src/main.ts",
    "lint": "npx eslint . --ext .ts,.js",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "engines": {
    "node": "10.14.1",
    "npm": "6.9.0"
  },
  "author": "Neloreck",
  "license": "MIT",
  "dependencies": {

```

```

    "body-parser": "^1.19.0",
    "express": "^4.17.1",
    "express-handlebars": "^5.2.0",
    "express-session": "^1.16.2",
    "handlebars": "^4.7.6",
    "ims-lti": "^3.0.2"
  },
  "devDependencies": {
    "@types/body-parser": "^1.19.0",
    "@types/express": "^4.17.9",
    "@types/express-handlebars": "^3.1.0",
    "@types/express-session": "^1.17.3",
    "@types/node": "^14.14.10",
    "@typescript-eslint/eslint-plugin": "^2.28.0",
    "@typescript-eslint/parser": "^2.28.0",
    "eslint": "^7.0.0",
    "eslint-plugin-import": "^2.20.2",
    "eslint-plugin-jest": "^23.11.0",
    "ts-node": "^9.0.0",
    "tsconfig-paths": "^3.9.0",
    "typescript": "^4.1.2"
  }
}

// tsconfig.json
{
  "compilerOptions": {
    "allowSyntheticDefaultImports": true,
    "esModuleInterop": true,
    "target": "es5",
    "module": "commonjs",
    "allowJs": true,
    "strict": true,
    "sourceMap": true,
    "declaration": true,
    "noImplicitAny": true,
    "moduleResolution": "node",
    "rootDir": "./",
    "baseUrl": "./",
    "resolveJsonModule": true,
    "emitDecoratorMetadata": false,
    "experimentalDecorators": true,
    "downlevelIteration": true,
    "types": [
      "node"
    ],
    "lib": [
      "es2017",
      "dom"
    ],
    "paths": {
      "@/*": [ "./src/*" ]
    }
  },
  "include": [
    "src/"
  ],
  "exclude": [
    "node_modules"
  ]
}

```

ДОДАТОК Б
Слайди презентації

Національний університет “Запорізька політехніка” Кафедра програмних засобів Дипломна робота магістра Дослідження та програмна реалізація засобів інтеграції контенту навчальних курсів онлайн платформ Виконав: студент групи КНТ-129М Керівник: к.т.н., професор		1 І. О. Сиротенко Г. В. Табунщик
--	--	---

Рисунок Б.1 – Слайд №1

Об’єкт, предмет та мета роботи Об’єкт – засоби інтеграції навчального контенту до систем дистанційного навчання Предмет – системи дистанційного навчання, навчальний контент та розширення до них Мета роботи - дослідження способів інтеграції навчального контенту та <u>розширень</u> систем дистанційного навчання, розробка власного розширення для додавання навчального контенту та інструментів до навчальних курсів	2
---	----------

Рисунок Б.1 – Слайд №2

Постановка завдання

1. Аналіз способів інтеграції розширень систем дистанційного навчання.
2. Аналіз сучасних систем дистанційного навчання і порівняння способів інтеграції розширень до них.
3. Вибір оптимального способу інтеграції контенту.
4. Проектування програмного застосунку.
5. Написання програмного коду.
6. Тестування та відлагодження розробленої програми.
7. Складання технічної документації.

Рисунок Б.1 – Слайд №3

Способи інтеграції розширень

API	Плагіни	SAML	LTI
- Залежність від реалізації платформи	- Залежність від реалізації платформи	- Стандартизований підхід	- Стандартизований підхід
- Тільки для однієї системи	- Тільки для однієї платформи	- Потребує окремої підтримки SAML	- Підтримується більшістю сучасних систем
- Не стандартизовано	- Стандартизовано в межах окремих систем	- Висока ступінь захищеності приватних даних	- Приватні дані не надаються стороннім розширенням
- Можливості обмежені	- В загальному випадку більш широкі можливості ніж API, більш низькорівневий доступ до ресурсів	- Значний відбиток необхідних бібліотек в кодовій базі	- Легковісний відбиток в кодовій базі
- Необхідність посилання до зовнішніх елементів	- Вбудовані до системи	- Потенційно може бути використаний в великих інфраструктурах	- Використовується тільки для розширень до систем дистанційного навчання

Рисунок Б.1 – Слайд №4

Загальна схема роботи LTI розширень

5

Обрані технології:

- LTI протокол
- NodeJS
- ExpressJS
- Typescript

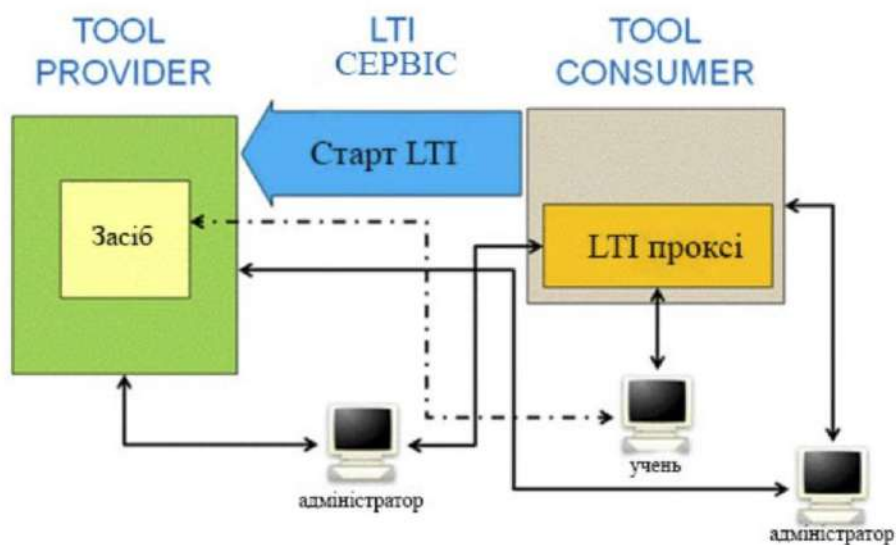


Рисунок Б.1 – Слайд №5

Діаграма діяльності застосунку

6

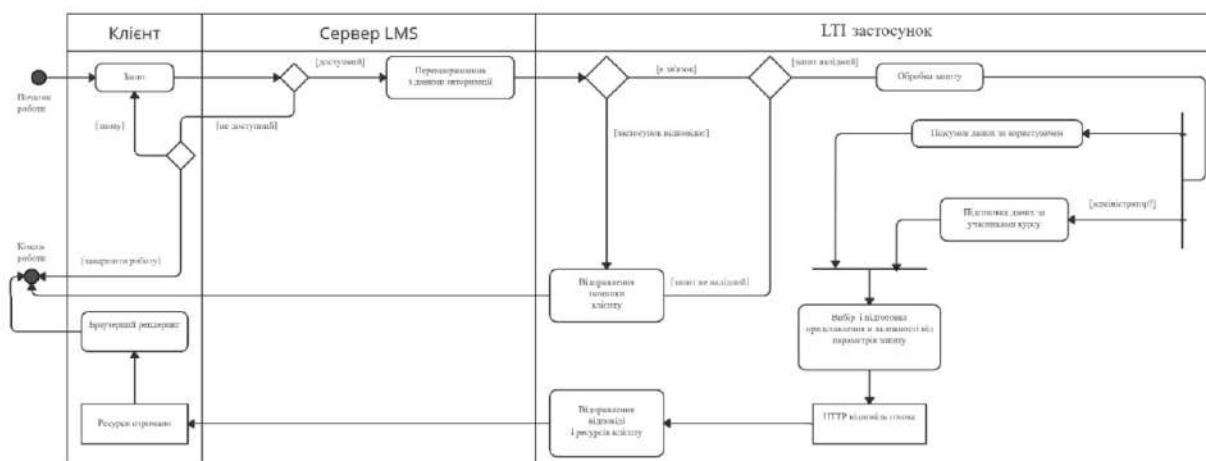


Рисунок Б.1 – Слайд №6

Огляд додавання розширення

7

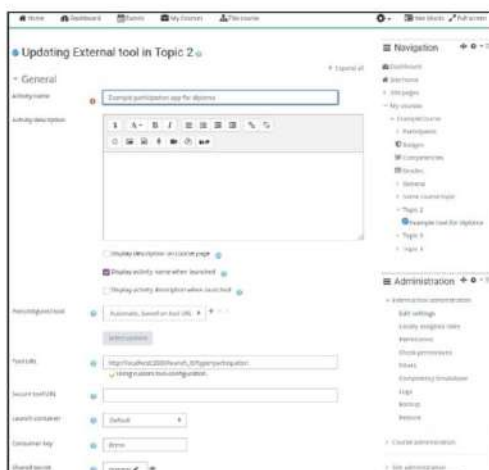


Рисунок 7.1 - Додавання LTI застосунку до системи

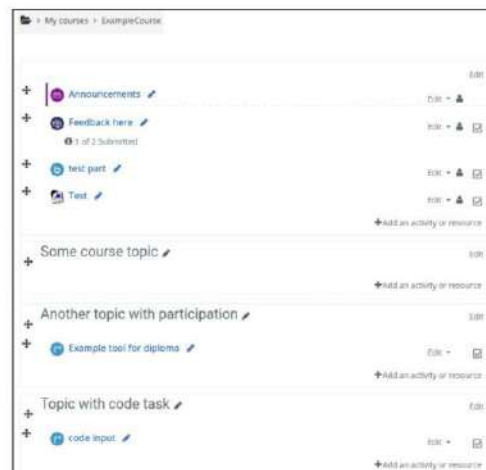


Рисунок 7.2 - Зовнішні засоби у списку курсу

Рисунок Б.1 – Слайд №7

Розподіл функціоналу розширення в залежності від параметрів

8

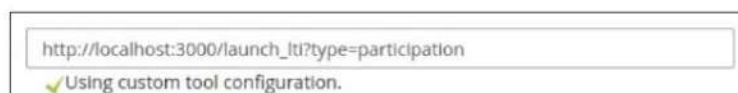


Рисунок 8.1 - Регулювання застосунку параметрами

Параметр	Значення	Результат
type	participation	Представлення для моніторингу присутності
type	code	Представлення для редагування програмного коду
type	undefined	Текстове повідомлення про необхідність вказати тип представлення
type	*	Повідомлення про те що представлення не знайдено

Рисунок Б.1 – Слайд №8

Результати роботи розширення

9

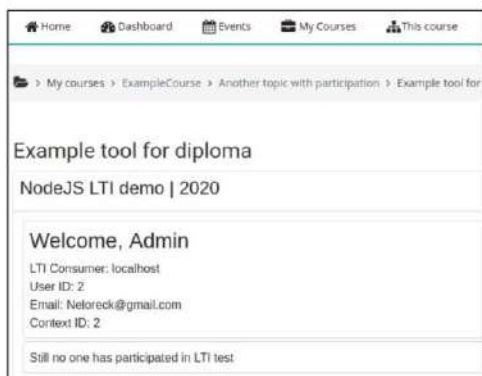


Рисунок 9.1 - Контекстна інформація та повідомлення про відсутність заповнень

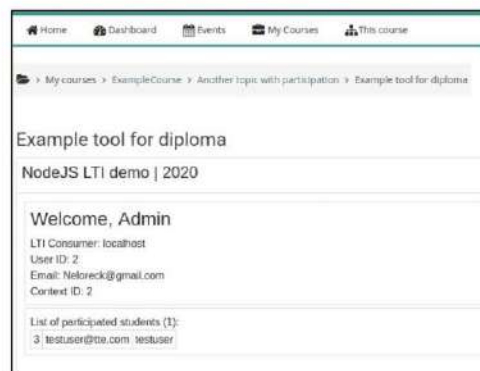


Рисунок 9.2 - Контекстна інформація та підсумки за заповненнями

Рисунок Б.1 – Слайд №9

10

Результати роботи розширення

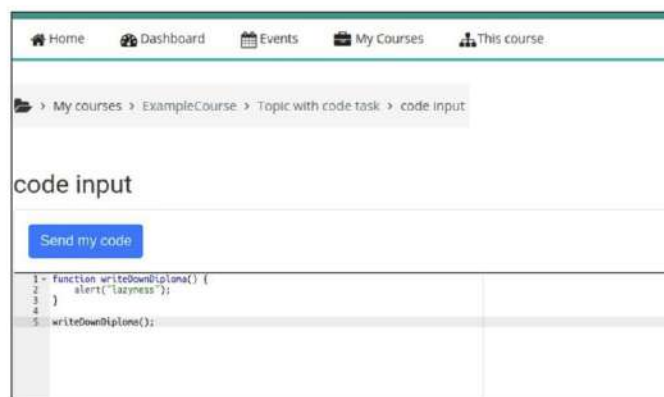


Рисунок Б.1 – Слайд №10

Висновки

В процесі виконання кваліфікаційної роботи магістра було досліджено сучасні системи дистанційного навчання та способи інтеграції навчального контенту і додаткових інструментів до них.

На основі проведених досліджень було спроектовано програмний засіб що може інтегруватися до систем дистанційного навчання за LTI протоколом і, в результаті підтримки розглянутих стандартів, може бути перевикористаним в інших системах та розширюватись виходячи з потреб користувача.

В результаті дослідницької та практичної роботи було обрано набір найбільш відповідних технологій та інструментів для реалізації даного застосунку виходячи з поставленого завдання та потреб середовища виконання даного роду програмних засобів.

Рисунок Б.1 – Слайд №11

Висновки

Готова програма має перевагу над специфічними для навчальних платформ плагінами у вигляді можливості інтеграції до будь-яких LTI сумісних систем і вимагає менш апаратних та програмних вимог.

За результатами роботи було опубліковано тези, що були подані на X Міжнародну науково-практичну конференцію «Сучасні проблеми і досягнення в галузі радіотехніки, телекомунікацій та інформаційних технологій».

Дослідження виконано в рамках міжнародного проекту Virtual Master Cooperation Data Science (ViMaCs) за підтримки DAAD.

Рисунок Б.1 – Слайд №12