Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

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

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

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

на тему РОЗРОБКА ВЕБЗАСТОСУНКУ ДЛЯ ОНЛАЙН-НАВЧАННЯ
(назва теми)
DEVELOPMENT OF A WEB APPLICATION FOR ONLINE STUDY

Виконав(ла): студент(ка) 4 курсу, групи КНТ-129
Спеціальності 121- Інженерія програмного забезпечення
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Інженерія програмного забезпечення

ОМЕЛЬЧЕНКО С. Р.
(ПРІЗВИЩЕ та ініціали)

Керівник ГОЛУБ Т. В.
(ПРІЗВИЩЕ та ініціали)
Рецензент ПОЛЯКОВ М. О.
(ПРІЗВИЩЕ та ініціали)

2023

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
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.
Сергій СУББОТІН
«_____» _____2023 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДНТА(КИ)

ОМЕЛЬЧЕНКО Станіслава Руслановича
(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка вебзастосунку для онлайн-навчання Development of a Web Application for Online Study
керівник проєкту (роботи) к.т.н., ГОЛУБ Тетяна Василівна,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)
затверджені наказом закладу вищої освіти від « 4 » квітня 2023року № 87
2. Строк подання студентом проєкту (роботи) 1 червня 2023 року
3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Аналіз існуючих інструментів та методів. 3. Опис, проектування та реалізація програмного забезпечення. 4. Тестування та експлуатація створеного вебзастосунку.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ГОЛУБ Т. В., доцент		
Нормоконтроль	КАМІНСЬКА Ж. К., асистент		

7. Дата видачі завдання « 13 » березня 2023року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1-2 тижні	Розділ 1
3	Вибір мови програмування та інших інструментів.	2-3 тижні	Розділ 2
4	Розробка архітектури програми.	3-4 тижні	Розділ 3
5	Розробка програми	5-6 тижні	Розділи 3,4
6	Тестування та експлуатація створеного вебзастосунку	7 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	7 тиждень	Додатки
8	Нормоконтроль та рецензування	8 тиждень	
9	Захист роботи	9 тиждень	

Студент(ка) _____ Станіслав ОМЕЛЬЧЕНКО
(підпис) (Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи) _____ Тетяна ГОЛУБ
(підпис) (Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 130 с., 63 рис., 14 табл., 3 дод., 19 джерел.

ВЕБЗАСТОСУНОК, ВЕБСАЙТ, ОНЛАЙН-НАВЧАННЯ, JAVA, MYSQL, SPRING BOOT.

Об’єкт дослідження – процес навчання за допомогою онлайн-застосунків. Предмет дослідження – програмне забезпечення для онлайн-навчання. Мета роботи – розробка вебзастосунку, що забезпечує зручне навчання в режимі онлайн.

Матеріали, методи та технічні засоби: мова програмування Java, фреймворк Spring Boot, персональний комп’ютер під управлінням операційної системи Microsoft Windows 11, база даних MySQL.

Результати. Створено вебзастосунок для онлайн-навчання, який надає можливість користувачам створювати власні курси або приєднуватися до існуючих курсів та працювати з ними.

Висновки. Розроблено вебзастосунок для онлайн-навчання, який надає можливість створення курсів та роботи з ними. Виконано тестування вебзастосунку на різних пристроях.

Галузь виконання – онлайн-навчання для слухачів різних освітніх рівнів.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 130 pages, 63 figures, 14 tables, 3 appendixes, 19 sources.

WEB APPLICATION, WEBSITE, ONLINE STUDY, JAVA, MYSQL, SPRING BOOT.

The object of the study is the process of learning with the help of online applications. The subject of research is software for online learning. The purpose of the work is to develop a web application that provides convenient online training.

Materials, methods and technical means: Java programming language, Spring Boot framework, personal computer running Microsoft Windows 11 operating system, MySQL database.

Results. Created a web application for online learning that allows users to create their own courses or join and work with existing courses.

Conclusions. A web application for online learning has been developed, which provides the ability to create and work with courses. Tested the web application on different devices.

The field of performance is online training for students of various educational levels.

ЗМІСТ

С.

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

ПЗ – програмне забезпечення

API – Application Programming Interface

HTML – Hyper Text Markup Language

JDK – Java Development Kit

JSON – JavaScript Object Notation

MVC – Model View Controller

REST – Representational State Transfer

STOMP – Simple (or Streaming) Text Oriented Message Protocol

UUID – Universally Unique Identifier

ВСТУП

У наш час інформаційних технологій не можливо уявити повсякденне життя без технологічних пристроїв та зокрема інтернету, який надав нам можливість користуватися різного роду вебзастосунками. Тому, саме інтернет надав нам таку можливість, як робота на відстані. Крім того, декілька років тому ми всі зрозуміли, що не тільки робота може бути на відстані, а також і навчання.

Існує велика кількість вебзастосунків, які реалізують таку можливість – навчатися дистанційно. Велика кількість з яких є платними, проте існує також багато дуже гарних та безкоштовних застосунків, які реалізують велику кількість функцій. Але в той же час з ними важко взаємодіяти, бо вони можуть здаватися на перший погляд складними або деякі з їхній функцій можуть бути не до кінця реалізовані.

Тому, було вирішено в рамках кваліфікаційної роботи бакалавра створити зручний для користувача вебзастосунок з достатньою кількістю функціоналу для ведення простого курсу без складних завдань або використання застосунку як доповнення до повноцінного очного навчального курсу в школі чи то у вищому навчальному закладі.

В роботі було установлено за мету розв’язати такі задачі:

- виконати аналіз предметної області та вже існуючих програмних реалізацій;

- спроектувати програмне забезпечення вебзастосунку для онлайн-навчання;

- розробити інтуїтивно зрозумілий вебзастосунок для онлайн-навчання;

- провести тестування створеного в рамках кваліфікаційної роботи вебзастосунку.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Поняття про інтернет систему

В наш час немає жодної людини, яка б не чула нічого про Інтернет. Зараз ми не можемо уявити наше життя без цього великого відкриття, яке зародилось в часи «холодної війни». Інтернет став революційним відкриттям для усього світу, як і відкриття електрики в свій час [1-2].

Інтернет став для людства системою для пошуку інформації, надав можливість спілкування між людьми з усіх куточків світу, також не треба забувати про можливість здійснення покупок і так далі. Тобто, можна сказати, що на сьогоднішній день всесвітня павутина (World Wide Web), як ще часто називають Інтернет, це створений людиною світ, в якому може опинися кожен бажаючий за допомогою комп'ютерах [1-2].

Перші прояви так званої всесвітньої павутини з'явилися у 1969 році, коли було створено першу комп'ютерну мережу під назвою ARPANET, яка стала прототипом для створення Інтернету, що ми знаємо сьогодні. Так на початку 90-х вже існувала система пошуку та передачі інформації Gopher. Проте, оскільки вона була створена в Університеті Міннесоти, право користування системою надавалося по ліцензійній підписці. Тому через деякий час організація ядерних досліджень (ЦЕРН), в якій працював англійський програміст Тім Бернерс-Лі, створила систему управління інформацією з можливістю легкого переходу на інші джерела інформації завдяки посиланням, яку Тім Бернерс-Лі назвав World Wide Web (Всесвітня павутина). Саме легкість доступу до великої кількості інформації привертала увагу людей. Так з початку створення Інтернету по 2011 в мережі було зареєстровано близько 1 млрд. користувачів [1-2].

Так, вебсторінка, яка є інформаційним ресурсом World Wide Web і яку можна переглянити у веббраузері, є ключовим аспектом доступу до безмежної

кількості різноманітної інформації. Для перегляду конкретної інформації користувач Інтернету переходять на вебсайти, які, в свою чергу, містять в собі вебсторінки, об'єднанні як за змістом, так і за навігацією. На даний момент існує дуже багато різноманітних вебсайтів, які зберігають величезну кількість інформації та допомагають нам завдяки цьому в повсякденному житті [3].

Іще одним важливим інструментом взаємодії з інформацією є вебзастосунок – це програмне забезпечення, яке суттєво відрізняється від вебсайту. Так як вебсайт несе в собі переважно інформаційний зміст, вебзастосунок служить для того, щоб користувач міг взаємодіяти с інформацією завдяки мовам програмування та різноманітним принципам розробки вебзастосунків [4].

1.2 Відомості щодо дистанційного навчання

У сучасному світі, коли користувачі проводять значну кількість часу в Інтернеті, сервіси для дистанційного навчання стають все більш затребувані для освітнього процесу та розвитку користувачів. Саме поняття дистанційного навчання декілька років тому здавалося чимось не зрозумілим та абсурдним, проте зараз все змінилося. Основними принципами системи дистанційної освіти є відкритість, гнучкість, динамічність, модульність, адаптивність, неперервність, креативність. Як ми розуміємо, дистанційна освіта базується переважно на самостійному отриманні студентом необхідного обсягу і якості знань та передбачає поєднання широкого спектра традиційних і новітніх інформаційних технологій [5].

Системи для дистанційного навчання можуть бути спрямованні на самостійне опрацювання матеріалу, де учень проходить курс самотужки в зручний йому час та не має ніяких обмежень по часу, або спрямованні на роботу з викладачем, де так званий учень курсу зможе отримувати відгуки на свої відповіді по ходу проходження курсу. Саме концепція, коли вчитель працює безпосередньо з учнями, набрала популярності, бо є дуже близькою до

шкільного навчання та може бути гарним допоміжним засобом до основного очного курсу в будь-якому навчальному закладі.

Виходячи з того, що тема дистанційного навчання є актуальною в наш час, було вирішено створити вебзастосунок для онлайн навчання, який би передбачав зв'язок учнів з викладачем, а також був інтуїтивно зрозумілим, та служив як допоміжний до основного курсу в навчальному закладі.

1.3 Дослідження існуючих систем, що вирішують поставлене завдання

До найрозповсюдженіших сервісів для онлайн навчання, на яких вже існує безліч різноманітних курсів, відносяться:

- Google Classroom;
- Moodle.

Проте існують інші менш відомі вебсервіси подібного типу:

- Edapp;
- Classdojo.

1.3.1 Google Classroom

Одним з най популярніших сервісів для онлайн-навчання є Classroom. Ця система створена дуже відомою компанією Google.

Цей застосунок має зрозумілий і дуже легкий в освоєнні інтерфейс, тому кожен з легкістю може створити свій курс або приєднатися до існуючого курсу. Також з легкістю можна створити завдання та додати його до курсу. Застосунок також включає в себе систему оцінювання учасників курсу. Проте один з недоліків Classroom – це те, що не має можливості створення чату. Хоч і сам чат присутній, проте він не створений як єдине ціле з курсом. Тим не менш, є можливість додавати коментарі під завданнями, що уможливлює зв'язок між учасниками курсу [6].

На рисунках 1.1 та 1.2 наведені головна сторінка вебзастосунку та список можливих варіантів завдань відповідно.

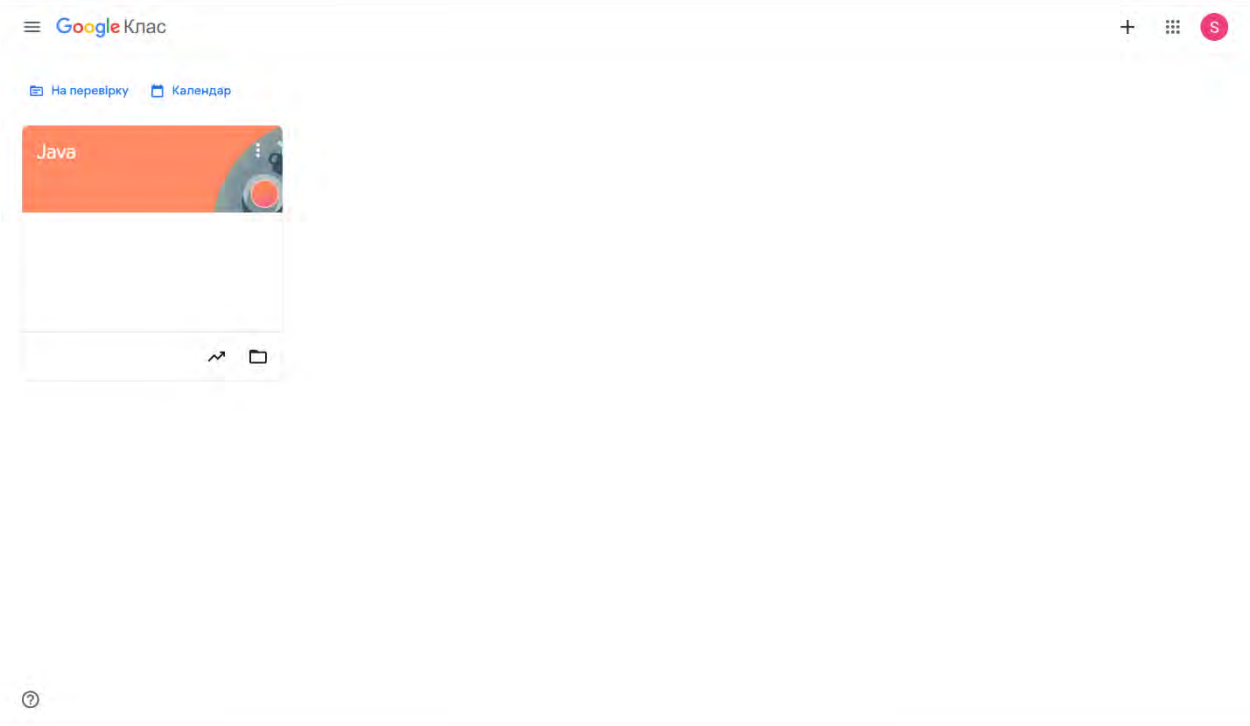


Рисунок 1.1 – Головна сторінка вебзастосунку авторизованого користувача Classroom [6]

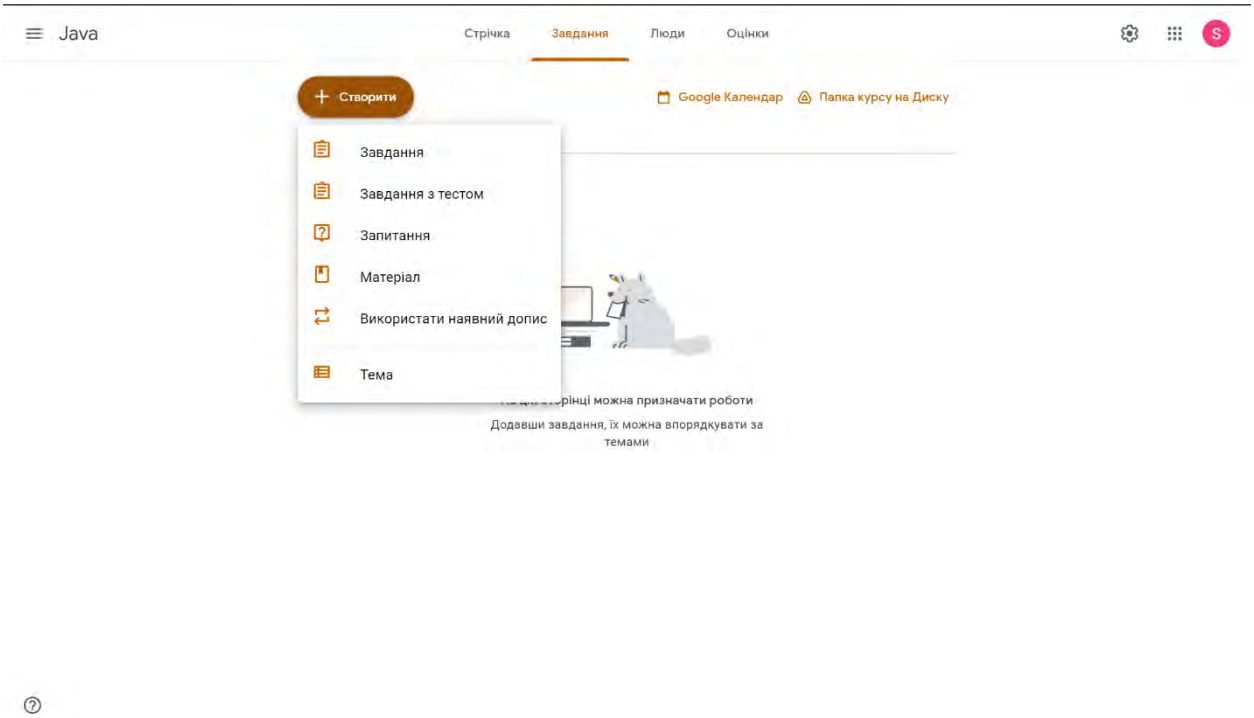


Рисунок 1.2 – Список можливих варіантів створення завдань [6]

1.3.2 MOODLE

Іще одним інструментом для навчання є вебзастосунок під назвою Moodle, він також є дуже відомим застосунком в багатьох країнах світу. MOODLE (Modular Object Oriented Dictance Learning Environment) – це система, яка дозволяє керувати навчальним контентом (LCMS – Learning Content Management Systems). За допомогою цієї системи можна створювати електронні курси та проводити навчання як у формі очної зустрічі, так і на відстані, зокрема, дистанційно [7].

Головним недоліком цієї системи є те, що її не так легко запустити, як то можна було зробити з Classroom. Ця система для запуску пропонує декілька способів, зокрема, серверний або хмарний. Тому це робить цю систему не такою легкою в освоєнні. Проте хоч Moodle може здатися складним, але надає дуже велику кількість можливостей та має багато переваг порівняно з іншими подібними системами [6].

Отже, якщо обрати створення цієї системи на базі сервера, то скоріше за все потрібно буде оплачувати хостинг. А якщо обирається хмарна версія MoodleCloud, то з нею можливо працювати тільки заплативши за підписку. Проте цей вебзастосунок надає можливість користування ним протягом перших 45 днів, як пробний період. Тому для дослідження ми обираєм саме цю систему [7-8].

На рисунках 1.3 та 1.4 наведені головна сторінка вебзастосунку та список можливих варіантів завдань відповідно.

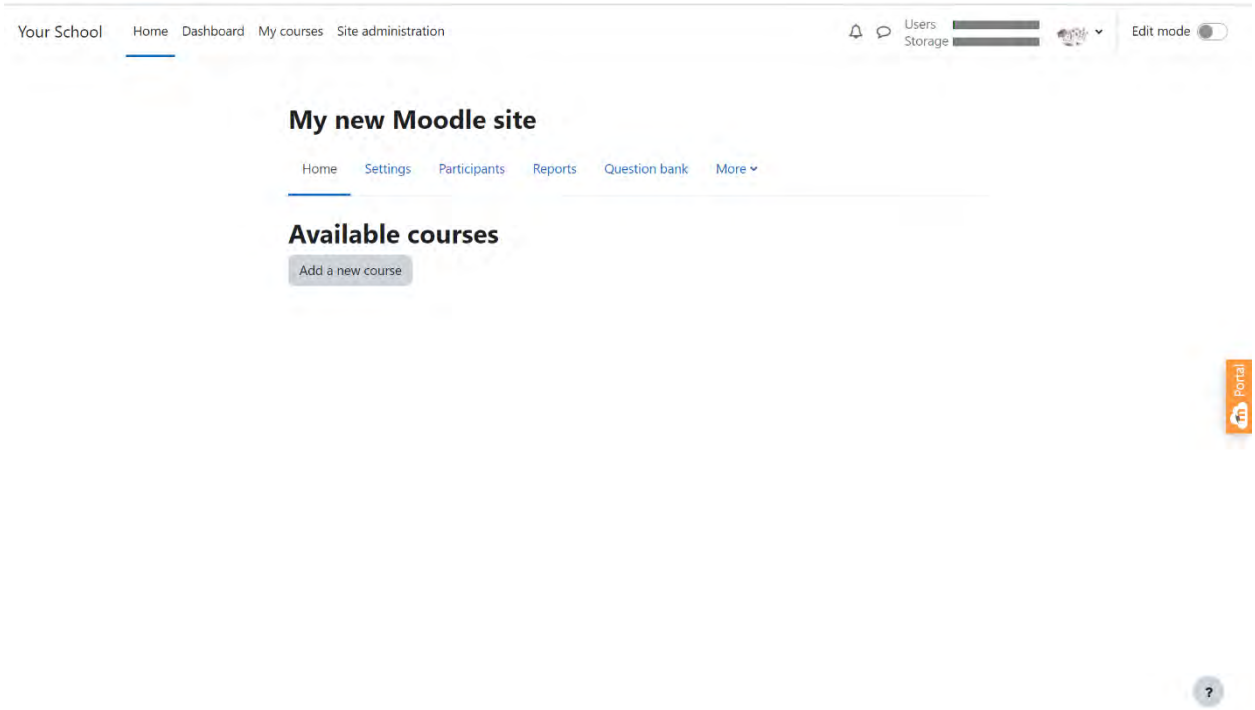


Рисунок 1.3 – Головна сторінка вебзастосунку авторизованого користувача MoodleCloud [8]

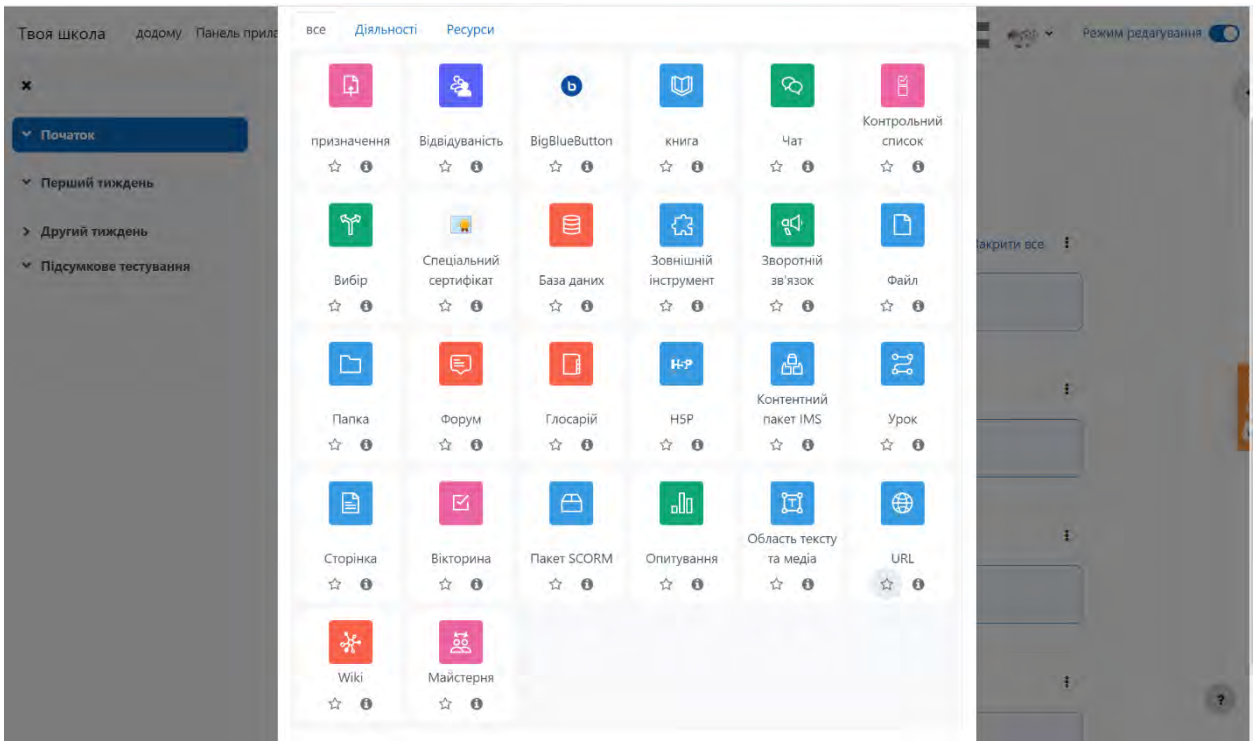


Рисунок 1.4 – Список можливих варіантів створення структури курсу [8]

На відміну від Classroom, ця система є дуже обширною. В ній є дуже велика кількість заготовлених в самій системі інструментів для створення гарної структури курсу. Система налічує функції створення курсів, систему

завдань, тестування, оцінювання і багато іншого. Також присутня краща система чату [7-8].

1.3.3 Edapp

Вебзастосунок Edapp створений теж для онлайн навчання. З його мінусів можна відзначити те, що є платний контент, який стісняє роботу з курсом. Без платного контенту немає можливості створити текстові питання, які зможе перевірити викладач, та неможливо створити свого роду чат, де користувачі зможуть обсудити теми або питання інших користувачів. Проте є в наявності пробний період на 30 днів. Також одним із мінусів є те, що тут немає системи оцінювання, як в попередніх прикладах, вона тут дуже своєрідна та не зрозуміла. Проте можливість створювати дуже гарні тестові запитання та інтерактивні ігри є перевагою цього вебзастосунку, а ще за проходження курсу можна сформувати сертифікат [9].

На рисунках 1.5 та 1.6 наведені головна сторінка вебзастосунку та список можливих варіантів завдань відповідно.

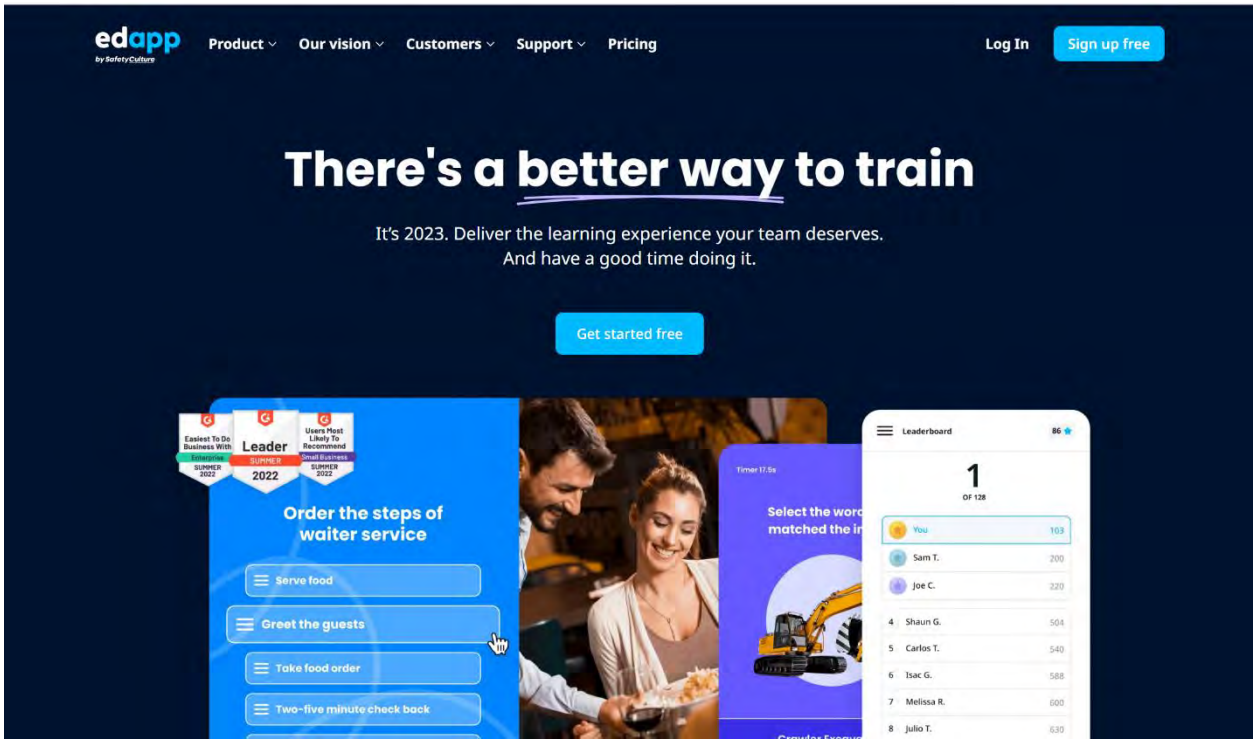


Рисунок 1.5 – Головна сторінка вебзастосунку Edapp [9]

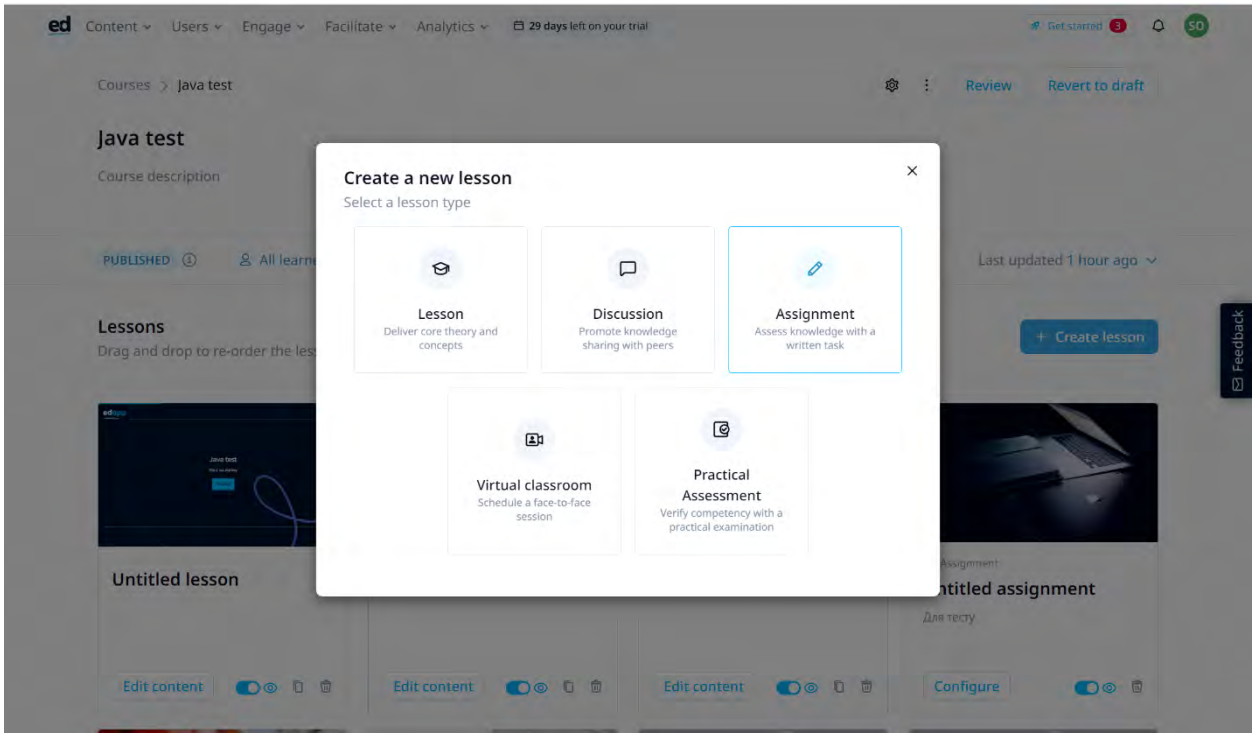


Рисунок 1.6 – Список можливих варіантів створення структури курсу [9]

Проаналізувавши цей вебзастосунок, можна сказати, що він більше підходить не для ведення повного курсу, а для використання в якості допоміжного та інформативного, який буде більше личити як доповнення до основного курсу, за проходження якого, як приклад, можна поставити додаткову оцінку.

1.3.4 Classdojo

Classdojo – це вебзастосунок, створений переважно для початкового шкільного навчання. Ключовою відмінністю цього застосунку є те, що він надає можливість 4 різних ролей, а саме:

- директор – створює кімнату (школу), в якій потім зможе створювати курси, завдання та додавати вчителів;
- вчитель – може створювати класи та завдання для учнів;
- учень – може виконувати завдання вчителів;
- батьки – можуть переглядати активність їхньої дитини.

Сам застосунок дуже легкий в освоєнні, має гарний інтерфейс, також неможливо не відзначити простоту реєстрації кожного користувача. Вебзастосунок має платний контент, без якого можливо в повній мірі користуватися застосунком, і це дуже добре. Проте із мінусів можна відзначити те, що варіативність завдань дуже мала, тому не підходить для старшої школи. Також присутній чат, проте він доступний тільки вчителям та батькам. Присутня система оцінювання, але вона своєрідна, більше схожа на систему відзнак. Великим плюсом можна відзначити наявність української локалізації [10].

На рисунках 1.7 та 1.8 наведені головна сторінка вебзастосунку та список можливих варіантів завдань відповідно.

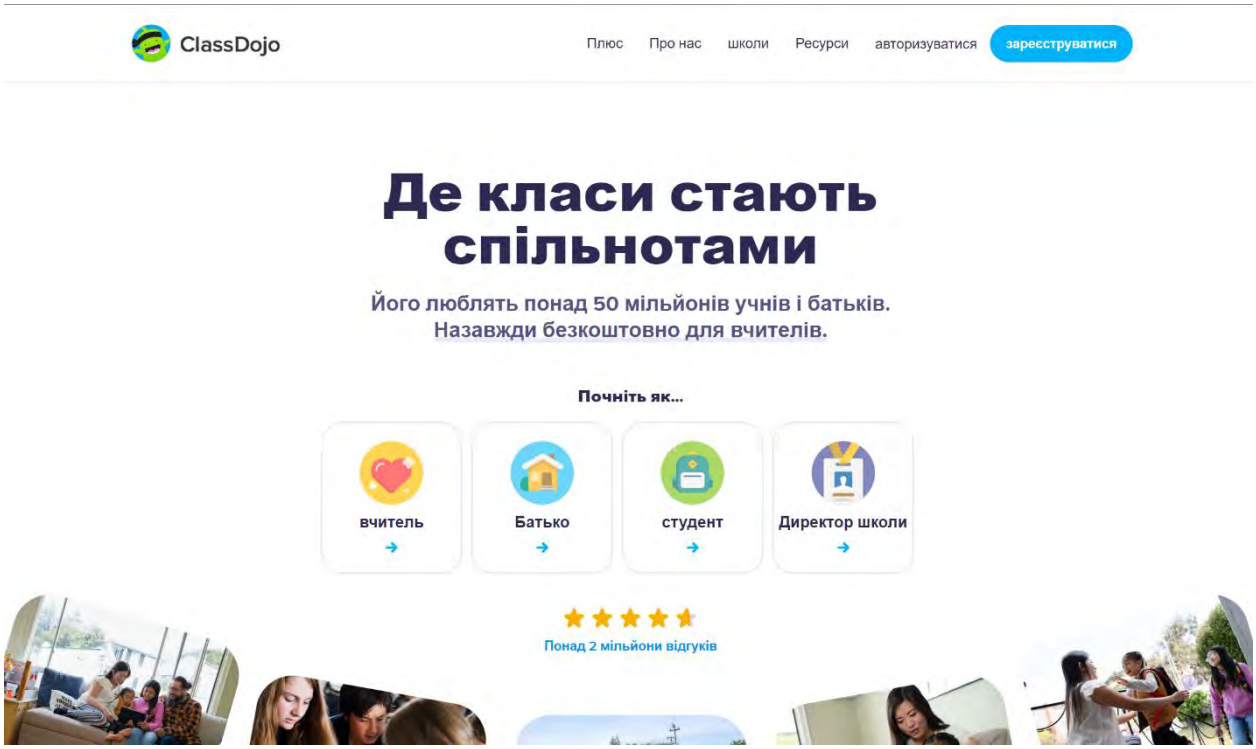


Рисунок 1.7 – Головна сторінка вебзастосунку Classdojo [10]

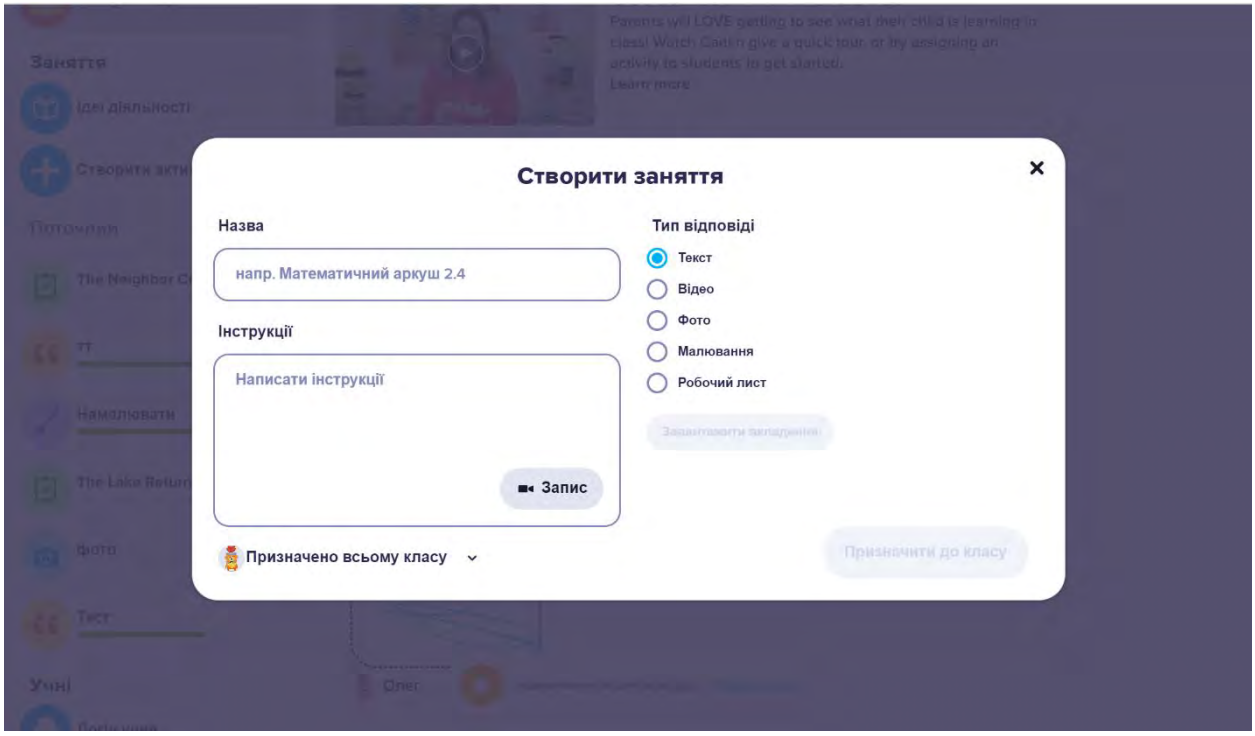


Рисунок 1.8 – Список можливих варіантів створення завдань [10]

1.3.5 Аналіз вебзастосунку для онлайн-навчання

На основі проведеного аналізу було виділено основні переваги та недоліки існуючого програмного забезпечення для онлайн-навчання. Результати аналізу наведені в порівняльній таблиці 1.1.

Таблиця 1.1 – Переваги та недоліки вебзастосунків

Критерій для порівняння	Classroom	MoodleCloud	Classdojo	Edapp
Інтуїтивно зрозумілий інтерфейс	+	-	+	+-
Обширність інструментів	+-	+	-	-
Наявність чату	+-	+	+-	+-
Система оцінювання	+	+	+-	+-
Наявність української локалізації	+	+	+	-
Необхідність купівлі ліцензії	+	-	+-	-

Таким чином, найбільш зручним застосунком для онлайн-навчання є MoodleCloud, проте він має досить складний інтерфейс і вимагає платної ліцензії. Інші інструменти характеризуються не зручністю у використанні чату, або не достатньою варіативністю завдань для ведення курсу, а в деяких випадках за додаткову варіативність доведеться заплатити.

1.4 Постановка завдання

Проаналізувавши вебзастосунки, представленні вище, та теоретичний матеріал по темі, було вирішено створити програмне забезпечення (ПЗ), зокрема, вебзастосунок для ведення простого курсу без складних завдань або використання застосунку як доповнення до повноцінного очного навчального курсу в школі чи в вищому навчальному закладі. Головними перевагами застосунку повинні бути інтуїтивно зрозумілий інтерфейс, щоб будь-який користувач з легкістю міг освоїтись та почати роботу з вебзастосунком, а також простий та зручний чат, який повинен забезпечити комунікацію між учасниками курсу. Основні типи завдань для оцінювання учасників курсу, а також, на додаток до цього, проста система перевірки та оцінювання цих завдань. Виходячи з цього, створений вебзастосунок повинен виконувати нижче зазначені функції:

- система реєстрації;
- створення курсів;
- можливість приєднуватись до курсу за посиланням;
- редагування профілю користувача;
- розподіл учасників курсу на ролі;
- додавання до курсу завдань різного типу (текстовий, тестовий, та візуальний);
- створення в курсі оголошень;
- наявність системи оцінювання учасників курсу;
- наявність можливості комунікації.

2 АНАЛІЗ ІСНУЮЧИХ ІНСТРУМЕНТІВ ТА МЕТОДІВ

2.1 Дослідження, вибір мови програмування та фреймворку

Для того, щоб створити вебзастосунок для онлайн навчання, можна використовувати різноманітні мови програмування. Існує багато мов програмування, на яких можна написати вебзастосунок. Основними мовами розробки вебзастосунків на сьогоднішній день є такі мови, як:

- JavaScript – мова, завдяки якій можна написати як клієнтську частину, так і серверну, тому є дуже популярною для написання такого роду ПЗ;

- Ruby – мова програмування для розробки вебзастосунку, основною перевагою якої є її простота та швидше написання вебзастосунків у порівнянні з іншими мовами;

- Python – універсальна мова для написання програмних продуктів будь-якого типу, в тому числі і вебзастосунків. Сама мова дуже легка в освоєнні та є дуже гарним варіантом для написання такого роду ПЗ;

- Java – одна з найстарших мов програмування, на якій можливо написати практично будь-що. Також, хоч і була створена відносно дуже давно, проте все ще можна побачити її в топ 3 мов програмування.

Для більш точної та об’єктивної оцінки вище наведених мов програмування проведемо їх порівняльний аналіз. Проте це не всі існуючі мови, на яких можливо написати вебзастосунок. Також можна згадати такі мови, як PHP, C#, Swift, Kotlin. Ці мови теж підходять для вирішення поставленого завдання.

Однією з найпопулярніших мов програмування для вебзастосунків є JavaScript, та фреймворк Node.js. З плюсів цього способу створення вебзастосунку можна виділити те, що Node.js чудово підходить для API. Саме JS створено для асинхронних запитів, що надає високу швидкість та зменшує використання оперативної пам’яті. Тому добре підходить для програм в реальному часі [11-12].

Головним недоліком можна виділити часті зміни, які не є зворотно сумісними. Означає це, що розробники будуть вимушені вносити зміни в код, щоб зробити його сумісним з останньою версією [11-12].

Також, популярна мова для створення вебзастосунку є Python, та його фреймворк Django. Сам Django – вебфреймворк високого рівня, завдяки легкості якого можна швидко розгорнути вебзастосунок. Основною перевагою, окрім легкості написання, можна виділити те, що сам Python можна широко використовувати, тому з легкістю можна створити вебзастосунки, в яких будуть вбудовані елементи аналітики та машинного навчання [12].

Із недоліків виділяють достатньо повільну роботу програми, також сам фреймворк змінює парадигму MVC на «Шаблон представлення моделі». MVC – це парадигма архітектури ПЗ, що надає можливість відокремити бізнес-логіку програми від його інтерфейсу та способу взаємодії з користувачем [12].

Серед не менш популярних мов програмування для створення вебзастосунків є Ruby, з його фреймворком Rails. Rails має головну перевагу у швидкому розгортанні завдяки акценту на простоті налаштування. Це зробило його популярним серед спільноти стартапів, тому що дозволяє розробникам швидко створювати мінімальний життєздатний продукт. Також завдяки великій спільноті, Rails має достатню кількість готових бібліотек та модулів [12].

З мінусів цього прикладу розробки можна виділити те, що Ruby on Rails більше підходить для маленьких проектів і буде погано себе показувати при створенні дійсно масштабного проекту. Завдяки тому, що кожен запит запускає свій власний екземпляр Rails, він може бути витратним з точки зору пам’яті [12].

Не можливо не згадати мову програмування Java, та її потужний фреймворк Spring Boot. Spring Boot – це дуже популярний фреймворк для розробки вебзастосунків на мові Java. Сама мова, як і фреймворк, є дуже надійними. Не дарма на Java створюють ПЗ для банківських систем. Головними перевагами, як вже було зазначено, є її надійність, мова добре

підходить для масштабних проектів, а також не можливо не зазначити безпечність роботи ПЗ, розробленої за використанням даної мови [11].

Основним недоліком можна виділити те, що об’єм коду буде значно більшим у порівнянні з представленими вище прикладами, також розробка цим способом може для початку здатися складною [11].

Отже проаналізувавши декілька варіантів, можна створити таблицю переваг того чи іншого варіанту див. табл. 2.1.

Таблиця 2.1 – Порівняння представлених способів розробки

Критерії для порівняння	Ruby on Rails	Python Django	Java Spring Boot	JavaScript Node.js
Об’єм написаного коду	+	+	+-	+
Легкість засвоєння	+	+	+-	+-
Стабільність та надійність	+-	+-	+	-
Масштабність проекту	-	-	+	+-
Зручність в створенні ПЗ для онлайн-навчання	+	+	+	+
Швидкість розробки	+	+	+-	+-
Підтримка спільнотою	+	+	+	+
Підтримка від провідних технологічних компаній	+	+	+	+
Зручність в розгортанні	+	+	+	+
Безпека ПЗ	+-	+-	+	+-

Отже, ознайомившись з вищевказаними способами розробки, було вирішено обрати мову програмування Java та її потужний фреймворк Spring Boot, завдяки багатьом перевагам перед іншими можливими способами. Також, велика кількість відкритої інформації в інтернеті по цьому фреймворку полегшує розробку цим способом, що надає значну перевагу.

2.2 Вибір бази даних та середовища розробки

На даний момент в світі існує безліч різноманітних варіантів баз даних (БД), які можна використовувати з технологією Spring Boot. Ось декілька прикладів можливих БД:

- MySQL – відкрита та безкоштовна реляційна база даних;
- PostgreSQL – також відкрита та дуже потужна база даних;
- MongoDB – безсхемна база даних, яка зберігає дані у вигляді JSON документів;
- Redis – база даних, яка реалізована за принципом key-value, що використовується для кешування та швидкого зберігання даних в базі даних.

Також існують і інші можливі варіанти, такі як Oracle, Microsoft SQL Server, Cassandra та інші.

Найбільш популярні варіанти БД з наведених вище – це MySQL та PostgreSQL. Проте обидві ці БД дуже потужні, тому важко їх порівнювати між собою, але можна виділити ключовий аспект, який полягає в підходах до транзакцій та відновлення даних після аварій. MySQL забезпечує більш швидкий підхід до транзакцій, у той час як PostgreSQL забезпечує більш надійний підхід. Також PostgreSQL підтримує роботу з форматом даних JSON. Тому можливо сказати, що MySQL підходить для більш простих проектів, де буде краще звернути увагу на продуктивність та швидкість [13]. Представимо порівняння баз даних в таблиці 2.2.

Таблиця 2.2 – Порівняння баз даних

Критерії для порівняння	MySQL	PostgreSQL
Складність	+	-
Масштабність	+	++
Швидкість	+	+-
Надійність	-	+
Сумісність	++	+

Тому для реалізації поставленої задачі було обрано БД MySQL, оскільки саме вона гарно підходить для реалізації поставлених завдань та має велику кількість відкритої інформації при роботі з нею.

При виборі середовища розробки було переглянуто декілька популярних варіантів, а саме:

- Eclipse є одним з найпопулярніших середовищ розробки на Java завдяки безлічі розширень та плагінів, які дозволяють розробникам отримати широку функціональність. Крім того, Eclipse має велику спільноту користувачів та надає допомогу в розробці різних типів додатків на Java;

- NetBeans – це популярна платформа розробки з простим у використанні та дружнім інтерфейсом для користувача. Вона надає засоби для створення різноманітних додатків на Java, а також має вбудовані засоби для розробки вебдодатків та мобільних додатків;

- IntelliJ IDEA – це інтегроване середовище розробки, яке надає широкі можливості для створення високоякісних додатків на Java. Воно має безліч корисних функцій, таких як автоматичне завершення коду, контроль якості коду та інспекцію коду, що полегшують процес розробки;

- Jdeveloper – це середовище розробки, розроблене компанією Oracle, яке дозволяє розробникам створювати додатки на Java для різних платформ. Воно має вбудовану підтримку баз даних, що дозволяє розробникам легко працювати з ними, а також має інструменти для розробки вебдодатків та мобільних додатків.

Переглянувши вище вказані варіанти, середовищем розробки було обрано програму IntelliJ IDEA Community Edition. Це середовище розробки є одним із най популярніших для роботи з проектами на мові програмування Java. IntelliJ IDEA має численні переваги для розробки програмного забезпечення, такі як підтримка різних фреймворків та бібліотек, автоматичне завершення коду, інструменти для рефакторинга, вбудований дебагер та багато інших корисних функцій, які допомагають зробити процес розробки швидким та ефективним.

Отже, переглянувши декілька можливих стеків для розробки вебзастосунку для онлайн-навчання за проведеним аналізом було обрано мову програмування Java та її потужний фреймворк Spring Boot, який безумовно підходить для виконання поставленого завдання. Також базою даних було обрано MySQL, яка забезпечить швидкість і надійність при роботі з даними. Протягом розробки буде використовуватися середовище розробки IntelliJ IDEA Community Edition для зручного та ефективного написання коду.

3 ОПИС, ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Опис використаних інструментів

Для створення вебзастосунку з використанням технології Java та Spring Boot, треба для початку створити проект на офіційному сайті Spring [14].

Щоб в застосунку реалізувати функції, які були визначені в попередніх розділах, до проекту було додано наступні інструменти:

- Spring Web – це бібліотека, яка допомагає розробникам створювати вебдодатки швидко та ефективно. Він надає велику кількість утиліт для розробки вебдодатків, таких як MVC, REST API, Websockets та інші;

- Lombok – це бібліотека, яка допомагає розробникам уникнути написання стандартного та повторюваного коду, такого як гетери та сетери, конструктори та інші методи. Вона дозволяє писати більш чистий та зрозумілий код, а також зменшує кількість помилок та прискорює розробку;

- Spring Security – це бібліотека, яка забезпечує безпеку вебзастосунків, таку як аутентифікація та авторизація користувачів, захист від атак та інші. Вона надає розробникам потужні інструменти для захисту вебзастосунків та забезпечує високий рівень безпеки;

- WebSocket – це протокол, який дозволяє забезпечити двосторонній зв’язок між клієнтом та сервером в режимі реального часу. Дозволяє розробникам створювати інтерактивні та динамічні вебзастосунки, які можуть обмінюватися даними між користувачем та сервером без перезавантаження сторінки;

- Spring Data JPA – це модуль Spring, який допомагає розробникам працювати з базами даних з використанням JPA. Надає простий та ефективний спосіб доступу до БД, забезпечуючи стандартні інтерфейси для взаємодії з ними;

- MySQL Driver – це драйвер бази даних, який дозволяє взаємодіяти з БД MySQL з використанням Java. Дозволяє розробникам створювати вебзастосунки, які можуть підключатися до MySQL – сховища даних,

отримувати доступ до даних, виконувати запити на читання та запис даних. Цей драйвер забезпечує ефективну роботу з БД MySQL і виконання SQL – запитів;

- Apache Freemarker – це бібліотека для генерування HTML сторінок на основі шаблонів. Дозволяє розділити логіку програми від відображення даних на сторінках. Таким чином, розробники можуть створювати динамічні вебсторінки, які легко змінювати та налаштовувати.

Для створення вебзастосунку було обрано інструмент Maven. Сам Maven – це інструмент для управління проектом. Він потрібен для автоматизації процесів збірки, залежностей та розгортання Java-проектів. Для проекту було обрано Spring Boot версії 2.7.11 та 11 версія Java. Хоча можливо було використати і більш новіші версії, які з'явилися зовсім нещодавно, але з версії Spring Boot 3 були введенні деякі зміни в процес створення вебзастосунку. А по причині того, що зараз не має достатньої кількості матеріалу по розробці на новій версії, це стало проблемою у використанні Spring Boot 3.

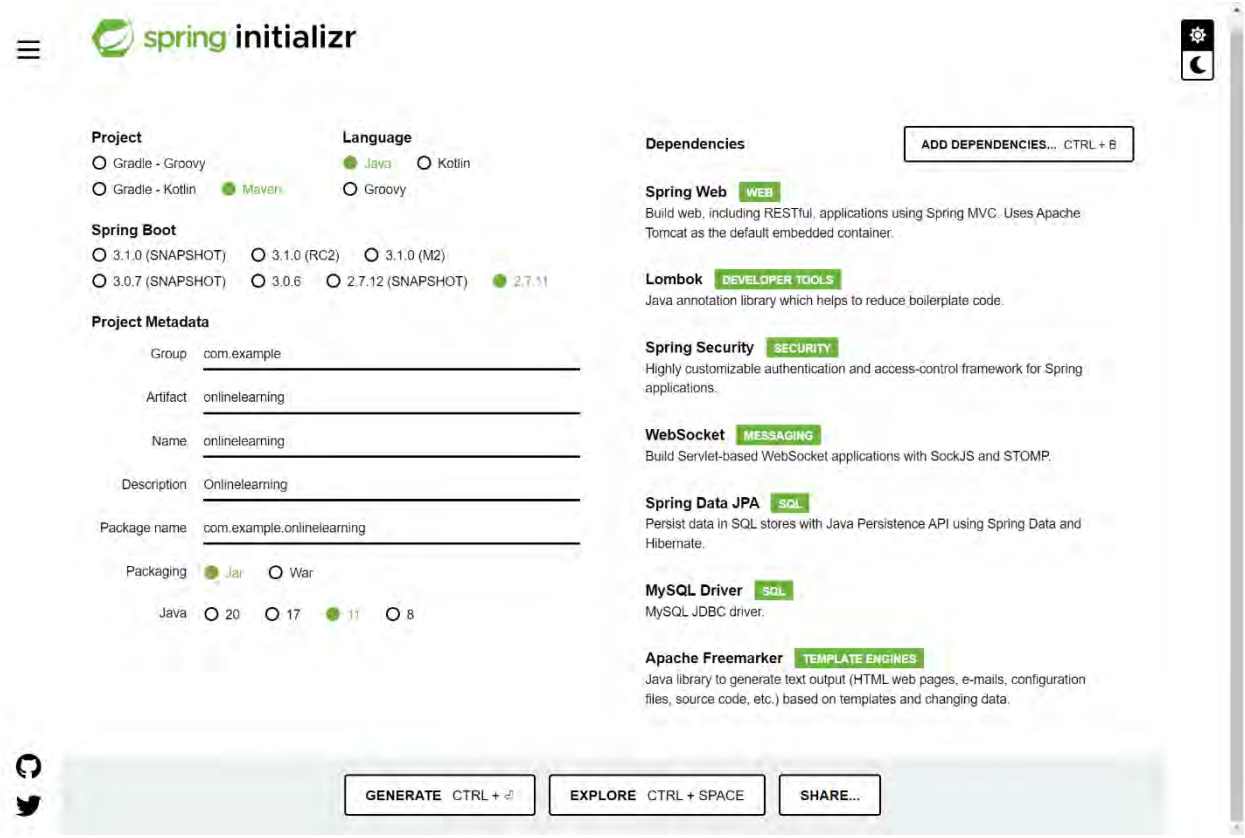


Рисунок 3.1 – Сторінка створення Spring проекту [14]

3.2 Реалізація функціональної частини програмного забезпечення

Для реалізації поставлених завдань проект можна розділити на такі функціональні підтеми:

- авторизація та реєстрація користувача вебзастосунку;
- функціональність авторизованих користувачів;
- розподіл на ролі учасників курсу;
- система комунікації (чат);
- система оцінювання.

3.2.1 Функція авторизації та реєстрації користувачів вебзастосунку

Для реалізації функції авторизації та реєстрації користувачів вебзастосунку було використано Spring Security. За допомогою цієї бібліотеки було налаштовано доступ до вебсторінок. Для того, щоб користувач міг в повній мірі користуватися вебзастосунком, він повинен обов'язково авторизуватися на сайті. Тому, поки користувач не авторизується, йому будуть доступні тільки вебсторінки з формою реєстрації та авторизації в системі.

Для створення такого підходу було створено клас під назвою SecurityConfing, який наслідує WebSecurityConfigurerAdapter. В цьому класі перезначено деякі методи, щоб заборонити доступ не авторизованим користувачам до вебсторінок, окрім вебсторінки з реєстрацією та авторизацією. [15-16]. Код даного підходу:

```
@Override
    protected void configure(HttpSecurity http) throws Exception
{
        http
                .authorizeRequests()
                .antMatchers("/css/**","/register")
                .permitAll()
                .anyRequest().authenticated()
                .and()
                .formLogin()
                .loginPage("/login")
```

```
                .permitAll()
                .and()
                .logout()
                .logoutUrl("/logout")
                .logoutSuccessUrl("/login")
                .invalidateHttpSession(true)
                .deleteCookies("JSESSIONID");
    }

    @Override
    protected void configure(AuthenticationManagerBuilder auth)
throws Exception   {
        auth.userDetailsService(userDetailsService)
                .passwordEncoder(passwordEncoder());
    }

    @Bean
    public PasswordEncoder passwordEncoder() {
        return new BcryptPasswordEncoder(8);
    }
}
```

А також для зберігання паролів користувачів в базі даних було використано BcryptPasswordEncoder – це клас шифрування, який забезпечує зручний і безпечний спосіб зберігання паролів користувачів у базі даних. Цей клас генерує хеш-значення пароля з використанням алгоритму bcrypt, який є одним з найбільш безпечних способів зберігання паролів. Параметр, який передається в нього, вказує на складність алгоритму bcrypt. Тому, чим більше цей параметр, тим більше часу потрібно для генерації хеш-значення [15-16].

3.2.2 Функціональність авторизованих користувачів

Вже після того, як користувач авторизувався на вебсайті, він потрапляє на головну вебсторінку сайту, де отримує можливість взаємодії в повній мірі з веб застосунком. Авторизованому користувачу надається можливість редагувати свій профіль, де він зможе змінювати свої персональні дані. Також користувач має можливість створення своїх курсів та підключення до вже існуючих курсів. Зазначена схема наведена на діаграмі прецедентів (рис. 3.2).

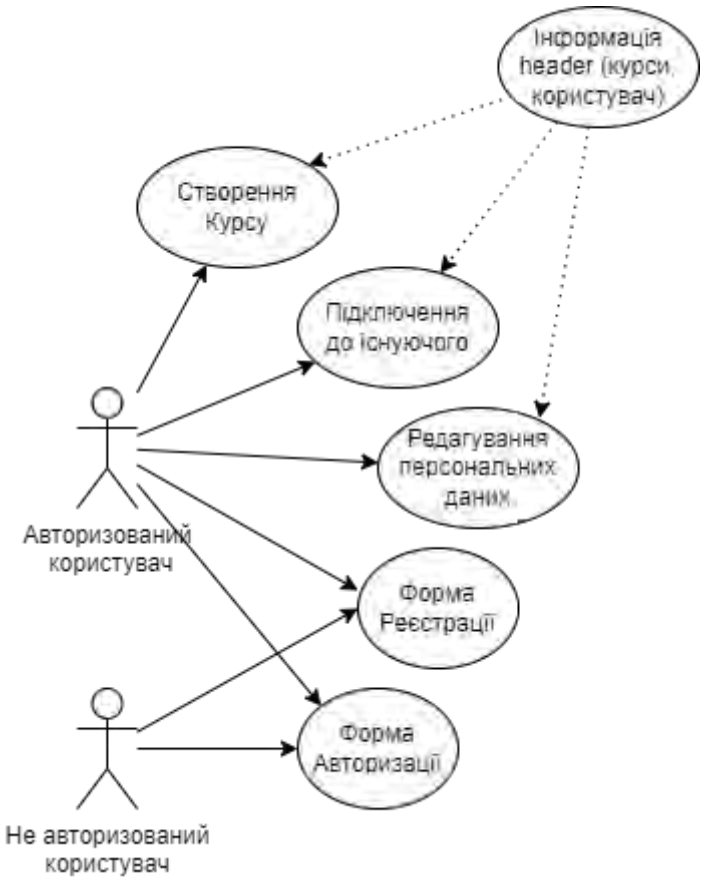


Рисунок 3.2 – Діаграма прецедентів

Таким чином, користувач зможе створити власний курс. При створені курсу користувачу необхідно буде вказати назву, як обов'язковий параметр, а також, за бажанням, заповнити поля назви групи та опис курсу.

Для того, щоб користувач міг приєднатися до вже створених курсів, йому необхідно ввести код-запрошення, який, в свою чергу, буде надаватися автоматично кожному створеному курсу, доступ до якого будуть мати вчителі курсу. Код автоматично генерується за допомогою java.util.UUID:

```
private String codeinvite = UUID.randomUUID().toString();
```

UUID генерує випадковий унікальний ідентифікатор, який складається з 32 символів, розділених дефісами. Це допоможе уникнути випадків однакових кодів-запрошень в ПЗ.

Для того, щоб змінити інформацію, користувач переходить на вебсторінку з інформацією користувача. Користувач у формі змінює

інформацію, та надсилає нову інформацію на сервер. В свою чергу, сервер робить пошук користувача в БД та змінює його дані. Також користувач може змінити псевдонім (нікнейм) – це поле знадобиться для чату. Саме цього поля немає при реєстрації, воно заповнюється автоматично ім’ям користувача.

3.2.3 Розподіл на ролі учасників курсу

Для реалізації поставленої задачі щодо взаємодії учнів та вчителів, в кожному курсі було створено розподіл на різні типи ролей, такі як:

- організатор;
- вчитель;
- учень.

Тому функціонал кожного типу з ролей буде відрізнятися деякими функціями. Так роль під назвою “ROLE_CREATER”, тобто організатор, буде мати такі функціональні можливості (рис. 3.3):

- редагування персональних даних;
- перегляд завдань;
- створення завдань;
- видалення завдань;
- редагування обраного завдання;
- створення питання;
- видалення питання;
- перегляд оголошень;
- створення оголошень;
- видалення оголошень;
- перегляд оцінок учнів;
- зміна поточних картинок курсу;
- використання чату;
- перевірка вирішених завдань;
- детальний перегляд вирішеного завдання;
- видалення даних вирішеного завдання;

- оцінювання вирішеного завдання;
- видалення курсу;
- перегляд учасників курсу;
- зміна ролей учнів та вчителів.

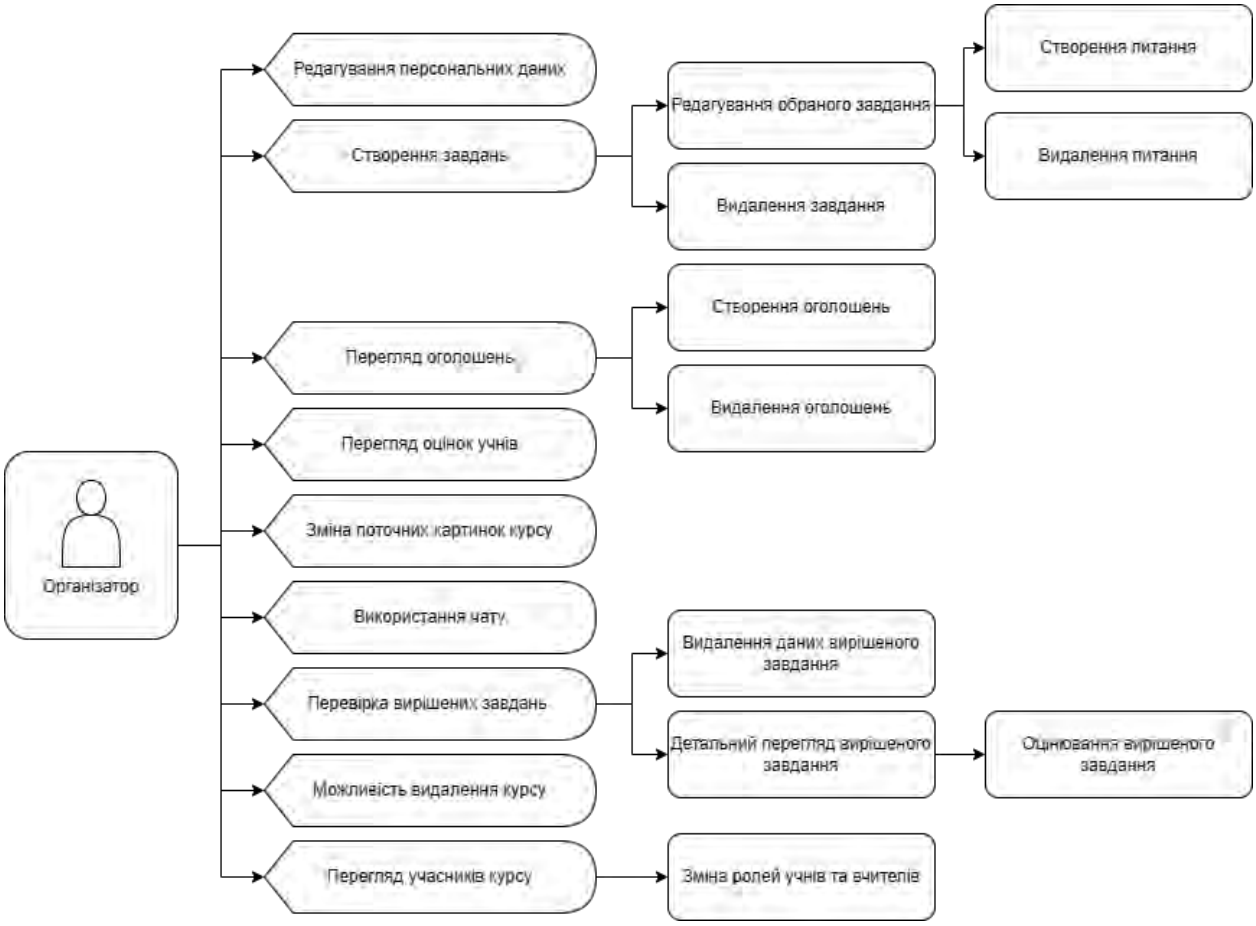


Рисунок 3.3 – Діаграма можливостей організатора

Схожий функціонал буде мати і вчитель “ROLE_ADMIN”. Цю роль може надавати учню організатор або другий вчитель. Проте, щоб був хоча б один вчитель в курсі, потрібно, щоб організатор надав роль вчителя комусь з учнів. Якщо дана роль нікому не надається, це може означати, що організатор буде самотужки справлятися з керуванням курсу. Вчитель має незначні відмінності в можливостях, якщо порівнювати з організатором (рис.3.4):

- редагування персональних даних;
- перегляд завдань;
- створення завдань;
- видалення завдань;

- редагування обраного завдання;
- створення питання;
- видалення питання;
- перегляд оголошень;
- створення оголошень;
- видалення оголошень;
- перегляд оцінок учнів;
- зміна поточних картинок курсу;
- використання чату;
- перевірка вирішених завдань;
- детальний перегляд вирішеного завдання;
- видалення даних вирішеного завдання;
- оцінювання вирішеного завдання;
- можливість покинути курс;
- перегляд учасників курсу;
- зміна ролі учня на роль вчителя.

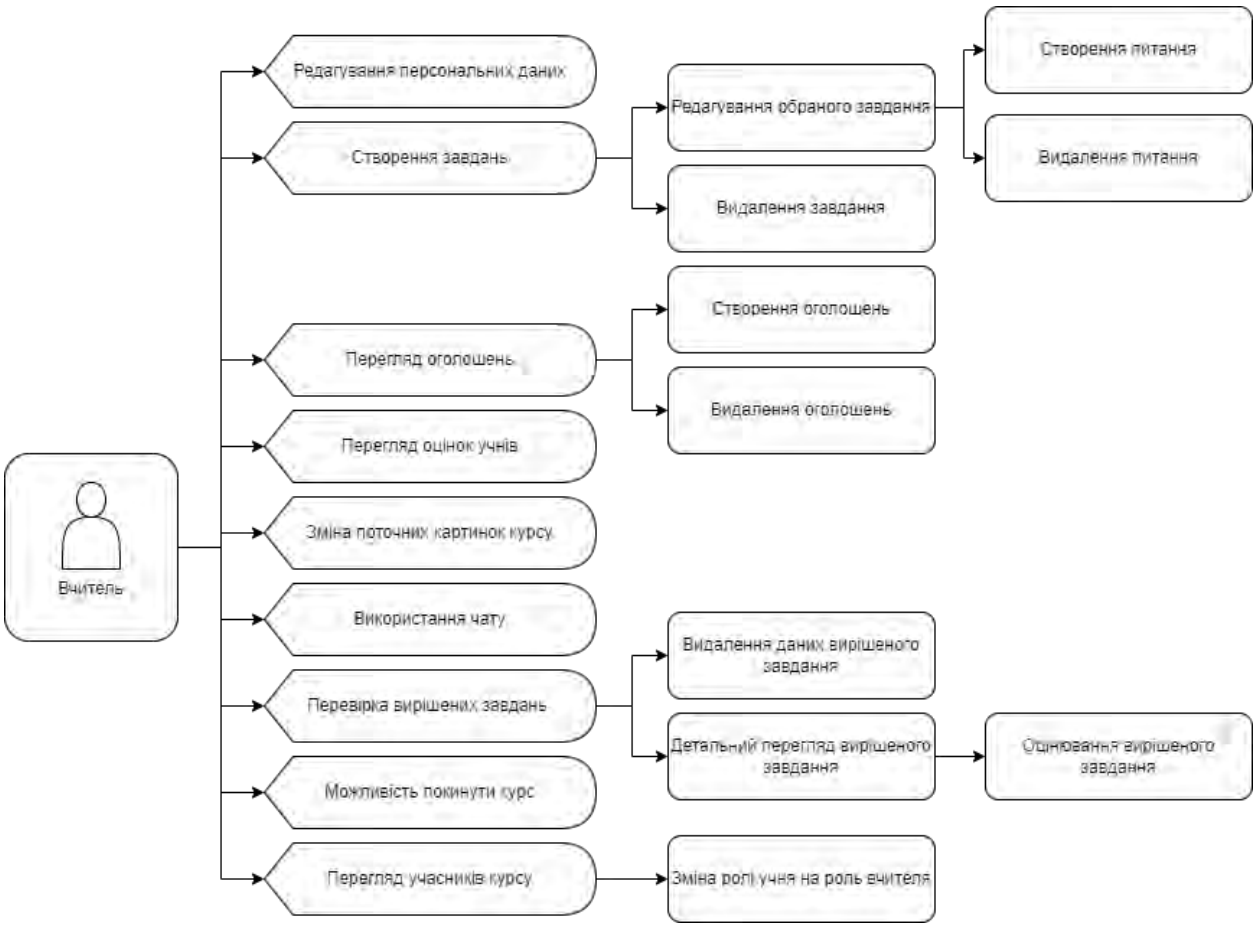


Рисунок 3.4 – Діаграма можливостей вчителя

Функціональні можливості учня “ROLE_USER”, вже не будуть мати таких можливостей, як створення завдань, об’яв і так далі. Учень буде мати інші функції щодо роботи з вебзастосунком, а саме (рис. 3.5):

- редагування персональних даних;
- перегляд завдань;
- перегляд статусів завдань;
- перегляд окремого завдання;
- вирішення завдання;
- перегляд оголошень;
- перегляд оцінок учнів;
- використання чату;
- можливість покинути курс;
- перегляд учасників курсу.

Рисунок 3.5 – Діаграма можливостей учня

Представимо наведений вище функціонал на діаграмі прецедентів (рис. 3.6).

Рисунок 3.6 – Діаграма прецедентів

3.2.4 Система комунікації (чат)

Для створення чату було використано бібліотеку WebSocket. За допомогою цієї бібліотеки було створено клас WebSocketConfig, який наслідує AbstractWebSocketMessageBrokerConfigurer. В цьому класі було перевизначено деякі методи, які налаштовують WebSocket з використанням простого брокера та реєструють ендпоінт для спілкування з клієнтом за допомогою SockJS [17]. Код створеного класу WebSocketConfig:

```
@Override
public void configureMessageBroker(MessageBrokerRegistry config)
{
    config.enableSimpleBroker("/topic");
```

```
    config.setApplicationDestinationPrefixes(“/app”);
}

@Override
public void registerStompEndpoints(StompEndpointRegistry
registry) {
    registry.addEndpoint(“/course/{id}/chat”).withSockJS();
}
```

Також в цьому випадку для відправки даних на сервер було використано мову JavaScript, а також бібліотеки SockJS та stompjs.

SockJS – це бібліотека JavaScript, яка дозволяє вебзастосункам використовувати протокол WebSockets, якщо він доступний, в іншому випадку переключатись на інші протоколи, які можуть забезпечити двосторонню комунікацію між клієнтом та сервером. SockJS надає надійні засоби забезпечення зв’язку, навіть якщо протокол WebSockets не підтримується браузером або сервером [18].

Сам STOMP – це протокол мережевої взаємодії, який використовується для обміну повідомленнями між програмами та пристроями з різними мовами програмування. STOMP розроблений з урахуванням простоти в розумінні та розробці, а також надійності і підтримки різноманітних транспортних протоколів [19].

Коли учасник курсу заходить на вебсторінку чату, він автоматично підключається до WebSocket сесії. Також, якщо учасник закриває вебсторінку, то він автоматично виходить з сесії. Щоб створити повідомлення, потрібно написати своє повідомлення та натиснути кнопку відправки. Після цього JavaScript відправить запит на сервер, не перезавантажуючи вебсторінку. На сервері цей запит приймається, заноситься до БД, та відправляє повідомлення всім підключеним учасникам сесії. Код для реалізації даного підходу:

```
Var stompClient = null;
window.onload = function() {connect();};
window.addEventListener(‘beforeunload’, function()
```

```
{stompClient.disconnect();});
 function disconnect() {
    stompClient.disconnect();
    console.log("Disconnected");
}
function connect() {
    var socket = new SockJS('/course/${course.id}/chat');
    stompClient = Stomp.over(socket);
    stompClient.connect({}, function(frame) {
        console.log('Connected: ' + frame);

stompClient.subscribe('/topic/course/${course.id}/chat/greetings
', function(greeting){
            showGreeting(JSON.parse(greeting.body));
        });
    });
}
function sendMess() {
    var message = document.getElementById('createMess').value;
    if(message != "") {
        stompClient.send("/app/course/${course.id}/chat", {},
JSON.stringify({ 'message': message, 'courseid':${course.id}
}));
        document.getElementById('createMess').value = "";
    }
}
function showGreeting(greet) {
    const newDiv = document.createElement('div');
    newDiv.classList.add('mess');
    newDiv.innerHTML =`
            <div style="display:flex; align-items:center;">
            <img style="width:55px;height:55px;border-
radius:50%;" src="`+greet.srcimg+`" alt="Фото користувача">
            <h1 style="margin-left:25px;">`+greet.nikname+`</h1>
            </div>
            <p>`+greet.content+`</p>
            <p style="font-size:15px;text-shadow: none;font-
```

```
width: bold;”>`+greet.time+`</p>`;
    var messages = document.getElementById(‘messages’);
    messages.appendChild(newDiv);
}
```

Таким чином забезпечується комунікація в реальному часі між учасниками сесії.

3.2.5 Система оцінювання

Як було зазначено в постановці завдання до проекту, вебзастосунок повинен мати систему оцінювання. Для вирішення даної задачі викладачам надана можливість перевіряти відповіді учнів та ставити за виконану роботу оцінки. Ці оцінки зберігатимуться в БД, а також будуть відображатися всім учасникам курсу на окремій вебсторінці.

3.3 Структура вебзастосунку

При створенні вебзастосунків з використанням Spring Boot. Було створено наведену нижче структуру проекту. Загальна структура програмного забезпечення зображена на рисунку 3.7.

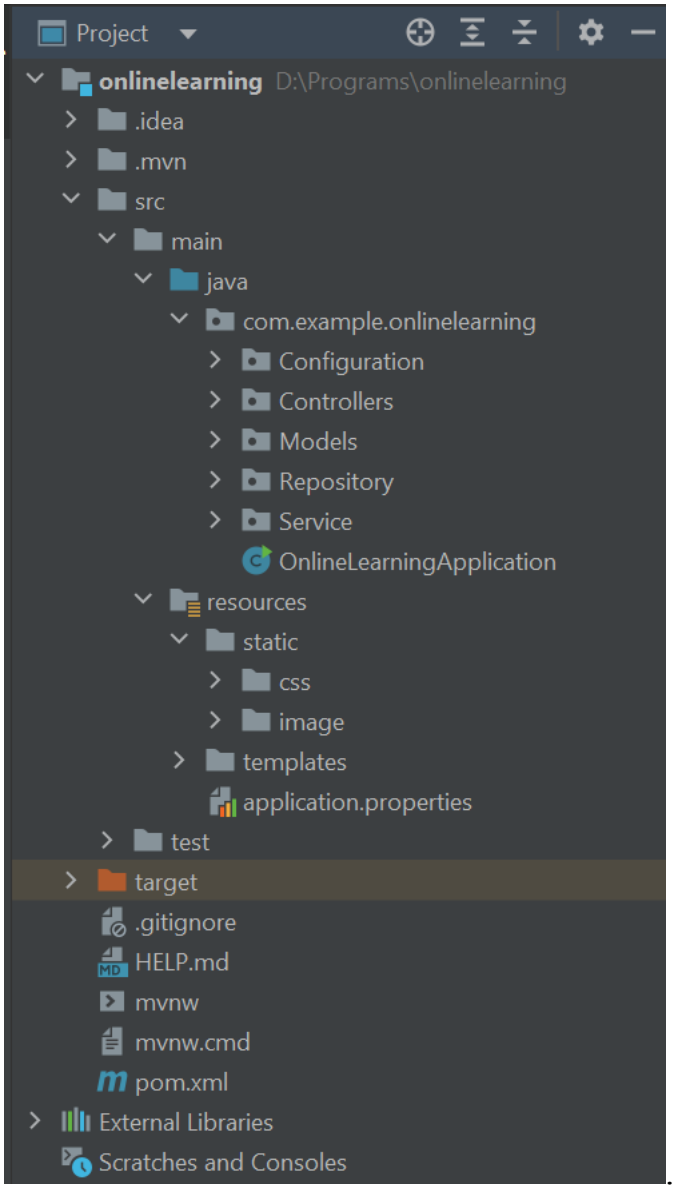


Рисунок 3.7 – Загальна структура програмного забезпечення

На рисунках 3.8 та 3.9 зображено вміст папок Configuration, Controllers, Models, Repository, Service.

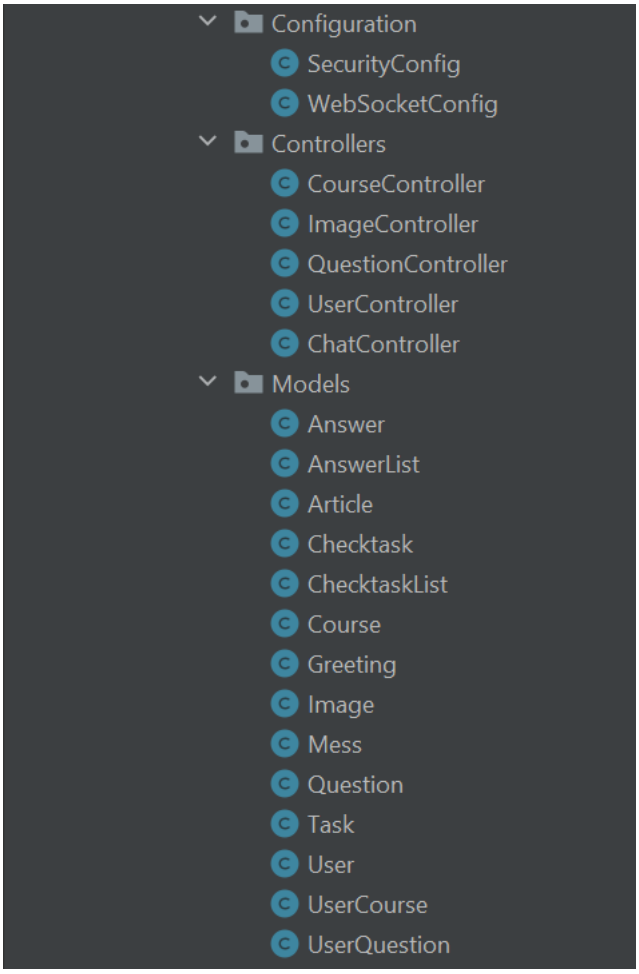


Рисунок 3.8 – Вміст папок Configuration, Controllers, Models

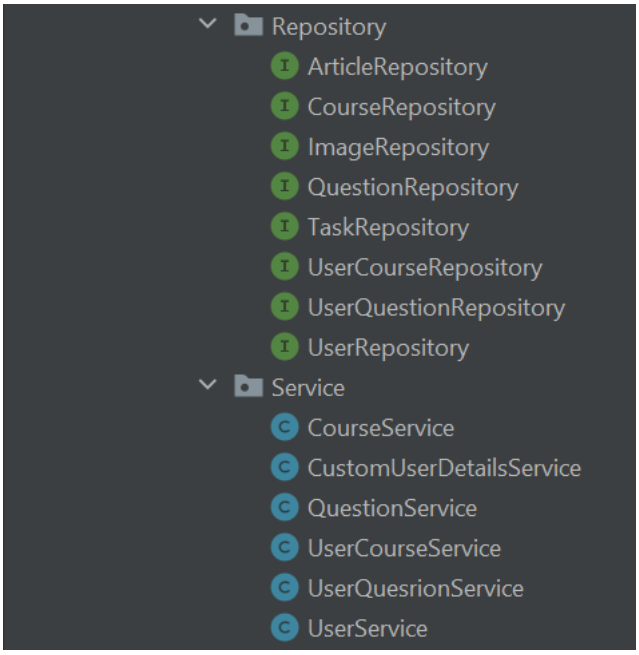


Рисунок 3.9 – Вміст папок Repository, Service

Вміст папок css, image, templates зображено на рисунку 3.10.

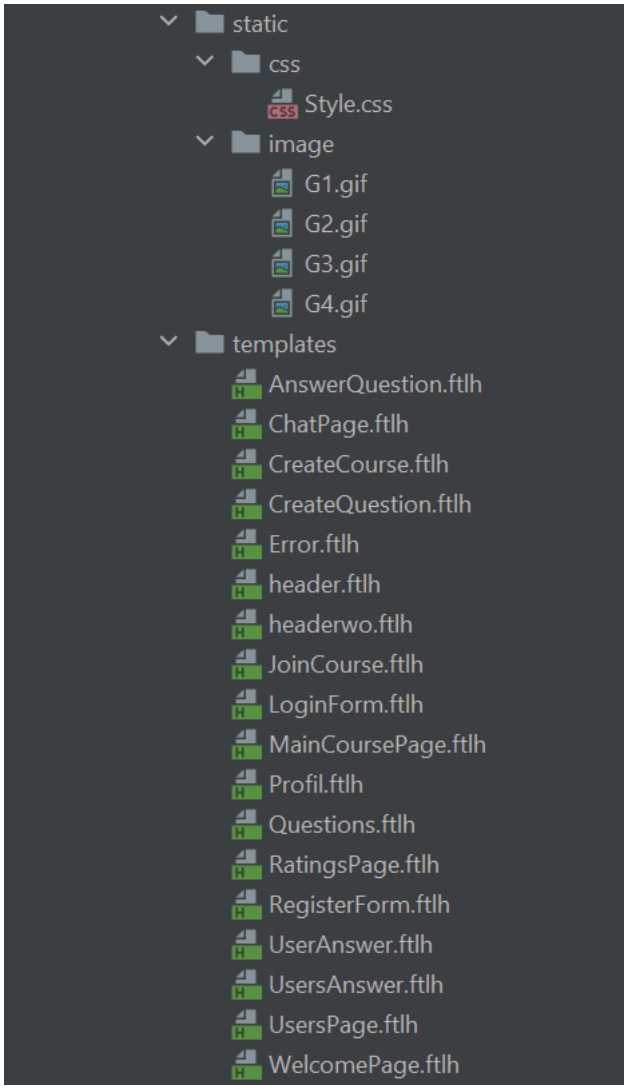


Рисунок 3.10 – Вміст папок css, image, templates

Вебзастосунок поділений на такі категорії:

- Configuration – в цій папці зберігаються конфігураційні файли;

- Controllers – в цій папці зберігаються файли контролерів, які відповідають за обробку запитів користувача та передачу необхідних даних між компонентами;

- Models – в цій папці зберігаються файли, які описують сутності, більша частина яких буде зберігатися в БД;

- Repository – в цій папці зберігаються файли, які являються абстрактним шаром між вебзастосунком і базою даних, і дозволяють легко і безпечно працювати з даними;

- Service – в цій папці зберігаються файли, які забезпечують бізнес-логіку вебзастосунку;

- static – ця папка розділена ще на 2 папки: css, де зберігається файл з стилями вебзастосунку, а також папка image, де зберігаються картинки, які будуть використовуватися в вебзастосунку;

- templates – в цій папці розміщуються файли, які використовуються для зберігання шаблонів HTML-сторінок, що генеруються на серверному боці. Проте, так як ми використовуємо в проекті бібліотеку Freemarker, то файли мають розширення ".ftlh". Ці шаблони містять статичний контент та динамічно згенерований контент, який передається з контролера в шаблон для відображення на вебсторінці.

3.3.1 Модуль Controllers

До модулю Controllers входять такі файли:

а) CourseController призначений для роботи з запитами, які відносяться до курсу:

1) відповідає на такі GET-запити як:
   - "/course/{id}";
   - "/course/{id}/users";
   - "/course/{id}/ratings";
   - "/course/{id}/chat".
2) приймає такі POST-запити, як:
   - "/course/{id}/addimage";
   - "/course/{id}/addlogo";
   - "/course/{id}/{usercourseid}/changerole";
   - "/course/{id}/delete";
   - "/course/{id}/conectdelete";
   - "/course/{id}/{idarticle}/deletearticle".

б) ImageController призначений для роботи з запитами, які відносяться до рисунків. Відповідає тільки на один GET-запит "/images/{id}", який

призначений для створення сторінки, яка приймає з БД дані про рисунок та створює на ній цей рисунок з отриманих даних. Така реалізація потрібна для відображення рисунків в тегах <img>;

в) QuestionController призначений для роботи із запитами, які відносяться до завдань курсу:

1) відповідає на такі GET-запити, як:

- “/course/{id}/question”;
- “/course/{id}/useranswer”;
- “/course/{id}/useranswer/{userid}/{45uestioned}”;
- “/course/{id}/question/{45uestioned}”.

2) приймає такі POST-запити, як:

- “/course/{id}/useranswer/{userid}/{45uestioned}”;
- “/course/{id}/question”;
- “/course/{id}/question/{45uestioned}/createanswer”;
- “/course/{id}/question/{45uestioned}/delete”;
- “/course/{id}/question/{45uestioned}/{45uesti}/delete”;
- “/course/{id}/question/{45uestioned}”;
- “/course/{id}/{userquestionid}/deleteuserinfo”.

г) UserController призначений для роботи із запитами, які відносяться до користувача:

1) відповідає на такі GET-запити як:

- “/login”;
- “/register”;
- “/”;
- “/profil”;
- “/createcourse”;
- “/joincourse”.

2) приймає такі POST-запити, як:

- “/register”;
- “/profil/addphoto”;
- “/profil/renameinfo”;

- “/createcourse”;
- “/joincourse”.

ґ) ChatController призначений для роботи із запитами, які відносяться до системи комунікації. В системі чату, як було описано вище, запит на сервер передається завдяки JavaScript та технології WebSocket. Сервер обробляє такі запити, як:

1) “/course/{id}/chat” – по цьому адресу сервер отримує повідомлення від учасників чату;

2) “/topic/course/{id}/chat/greetings” – по цьому адресу сервер розсилає повідомлення всім учасникам чату.

3.3.2 Модуль Service, бізнес-логіка

В модуль Service входять файли, що відповідають за бізнес-логіку вебзастосунку. До них відносяться:

а) CourseService – цей сервіс обробляє основні функції сутності курс, а саме:

1) функції для зберігання картинок курсу (saveCourseImageByID, saveCourseLogoByID);

2) функція для зберігання курсу в БД (saveCourse);

3) функція для отримання з БД курсу (getCourseByID);

4) функція для отримання курсу з БД по коду-запрошення (getCourseByCodeinvite);

5) функція для видалення курсу з БД (deleteCourseByCourse);

6) функція для створення оголошень в курсі та БД (createArticleByCourse);

7) функція для видалення оголошень з курсу та БД (deleteArticle).

в) CustomUserDetailsService – цей сервіс був створений для роботи з Spring Security. Він наслідує клас UserDetailsService та перевизначає тільки один метод loadUserByUsername, який служить для пошуку з БД користувача по його Email;

г) QuestionService – цей сервіс обробляє основні функції сутності завдань, а саме:

1) функція для отримання завдання з БД (getQuestionByID);

2) функція для зберігання завдання в БД (saveQuestion);

3) функція для видалення завдання з БД (deleteQuestion);

4) функція для видалення запитання із завдання та БД (deleteTaskById).

ґ) UserCourseService – цей сервіс служить для роботи учасника курсу та самого курсу. Реалізує такі функції:

1) функція для створення курсу та збереження зв’язку між користувачем та курсом в БД (createUserCourse);

2) функція для приєднання користувача до курсу та збереження зв’язку між користувачем та курсом в БД (JoinUserCourse);

3) функція для отримання з вказаного курсу всіх учасників з роллю учня (getOnlyUsersByCourse);

4) функція для отримання з вказаного курсу всіх учасників з роллю вчителя (getOnlyAdminsByCourse);

5) функція для отримання ролі учасника курсу (getUserRole);

6) функція для зміни ролей для вказаного учасника (changeUserRole);

7) функція для видалення курсу та всіх його зв’язків з БД (deleteCourseByCourse);

8) функція для видалення зв’язку з курсу та БД (deleteUserCourse);

9) функція для отримання зв’язку з БД (getUserCourseByID).

д) UserQuesrionService – цей сервіс служить для роботи учасника курсу та завдань. Реалізує такі функції:

1) функцію для створення відповіді на завдання та збереження її в БД (createUserQuestion);

2) функція для отримання відповіді з БД (getUserQuestionByUserAndQuestion);

3) функція для збереження відповіді в БД (save);

4) функція для видалення завдання з БД (deleteQuestionByQuestion);

5) функція, яка потрібна, коли вчитель змінює структуру запитань в завданні (reinfoQuestion);

6) функція для отримання всіх відповідей учасника курсу з БД (getAllUserQuestionByUser);

7) функція для видалення з БД відповіді (deleteUserQuestionById).

е) UserService – цей сервіс обробляє основні функції для роботи з сутністю користувача. Реалізує такі функції :

1) функція для зберігання користувача в БД (saveUser);

2) функція для створення користувача (userCreate);

3) функція для додавання користувачу фотографії (saveUserPhotoByPrincipal);

4) функція для отримання користувача з БД за допомогою id (getUserByID);

5) функція для отримання користувача з БД за допомогою Principal (getUserByPrincipal).

3.4 Опис бази даних

Для роботи вебзастосунку було створено такі сутності: users, user_question, user_course, tasks, questions, messages, images, courses, checktasks, articles, answers.

Сутність users зберігає в собі інформацію про користувачів. Представимо цю сутність в табл. 3.1.

Таблиця 3.1 – Сутність users

Назва поля	Тип змінної	Призначення
id	Long	Унікальне числове значення
email	String	Для аутентифікації
name	String	Ім’я

Продовження таблиці 3.1

Назва поля	Тип змінної	Призначення
familyname	String	Ім’я по батькові
surname	String	Прізвище
nikname	String	Псевдонім потрібен для чату
password	String	Пароль
image_id	Long	Id фотографії

Сутність courses зберігає в собі інформацію про курси. Представимо цю сутність в табл. 3.2.

Таблиця 3.2 – Сутність courses

Назва поля	Тип змінної	Призначення
id	Long	Унікальне числове значення
codeinvite	String	Код-запрошення
description	String	Опис курсу
groupname	String	Група курсу
title	String	Заголовок курсу
image_id	Long	Id головної картинки курсу
logoimage_id	Long	Id логотипу курсу

Сутність images зберігає в собі інформацію про картинки. Представимо цю сутність в табл. 3.3.

Таблиця 3.3 – Сутність images

Назва поля	Тип змінної	Призначення
id	Long	Унікальне числове значення
bytes	Byte []	Код-запрошення
contentType	String	Тип файлу
originalFileName	String	Назва файлу

Продовження таблиці 3.3

Назва поля	Тип змінної	Призначення
size	Long	Розмір файлу
answer_id	Long	Id до якої відповіді належить ця картинка

Сутність user_course зберігає в собі інформацію про зв’язок між курсом та користувачем. Представимо цю сутність в табл. 3.4.

Таблиця 3.4 – Сутність user_course

Назва поля	Тип змінної	Призначення
id	Long	Унікальне числове значення
course_id	Long	Id курсу
user_id	Long	Id користувача
role	String	Роль користувача в даному курсі

Сутність questions зберігає в собі інформацію про завдання курсу. Представимо цю сутність в табл. 3.5.

Таблиця 3.5 – Сутність questions

Назва поля	Тип змінної	Призначення
id	Long	Унікальне числове значення
title	String	Заголовок завдання
course_id	Long	Id курсу

Сутність user_question зберігає в собі інформацію про зв’язок між завданням та користувачем. Представимо цю сутність в табл. 3.6.

Таблиця 3.6 – Сутність user_question

Назва поля	Тип змінної	Призначення
id	Long	Унікальне числове значення
question_id	Long	Id завдання
user_id	Long	Id користувача
rating	String	Оцінка
status	String	Статус завдання (не виконано, перевіряється, виконано)

Сутність tasks зберігає в собі інформацію про запитання (підзавдання). Представимо цю сутність в табл. 3.7.

Таблиця 3.7 – Сутність tasks

Назва поля	Тип змінної	Призначення
id	Long	Унікальне числове значення
quest	String	Текст запитання
type	String	Тип запитання (текст, картинка, тест)
question_id	Long	Id завдання

Сутність answers зберігає в собі інформацію про відповідь користувача. Представимо цю сутність в табл. 3.8.

Таблиця 3.8 – Сутність answers

Назва поля	Тип змінної	Призначення
id	Long	Унікальне числове значення
quest	String	Текст запитання
textarea	String	Текст відповіді на запитання типу “текст”

Продовження таблиці 3.8

Назва поля	Тип змінної	Призначення
type	String	Тип запитання (текст, картинка, тест)
userquestion_id	Long	Id зв’язку користувача та запитання

Сутність checktasks зберігає в собі інформацію запитання типу “тест”. Представимо цю сутність в табл. 3.9.

Таблиця 3.9 – Сутність checktasks

Назва поля	Тип змінної	Призначення
id	Long	Унікальне числове значення
answer	String	Поле відповіді на запитання
istrue	Boolean	Чи позначене як відповідь на запитання
task_id	Long	Id запитання
answer_id	Long	Id відповіді

Сутність articles зберігає в собі інформацію про оголошення курсу. Представимо цю сутність в табл. 3.10.

Таблиця 3.10 – Сутність articles

Назва поля	Тип змінної	Призначення
id	Long	Унікальне числове значення
description	String	Зміст оголошень
time	String	Час коли була створене оголошення
title	String	Заголовок оголошення
course_id	Long	Id курсу

Сутність messages зберігає в собі інформацію про повідомлення. Представимо цю сутність в табл. 3.11.

Таблиця 3.11 – Сутність messages

Назва поля	Тип змінної	Призначення
id	Long	Унікальне числове значення
message	String	Повідомлення
time	String	Час коли було відправлено повідомлення
user_id	Long	Id користувача
course_id	Long	Id курсу

Виходячи з поданих вище даних про сутності вебзастосунку, було створено діаграму сутностей рис. 3.11.

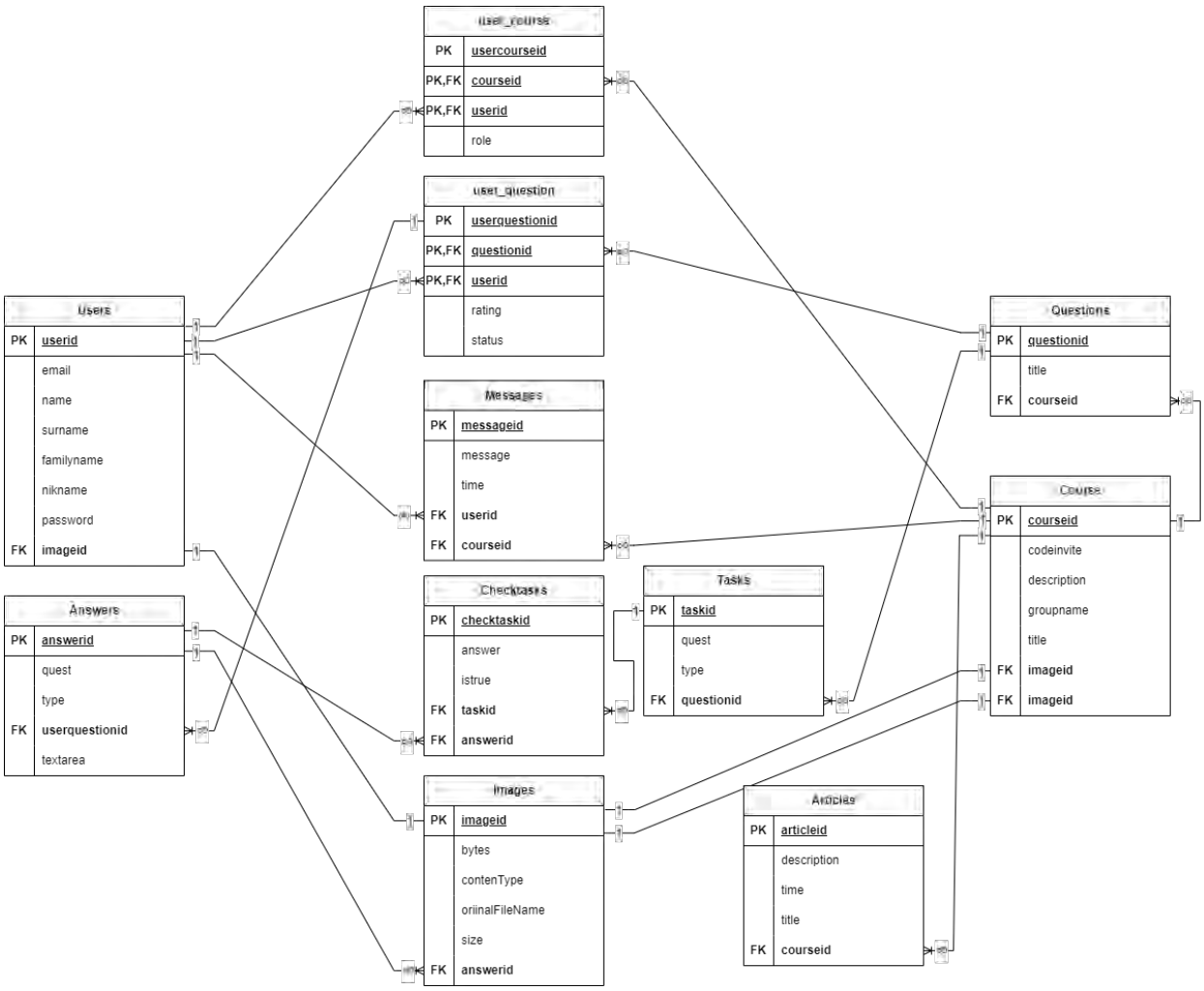


Рисунок 3.11 – Діаграма сутностей вебзастосунку

Отже, БД даних налічує в собі одинадцять сутностей, які забезпечують зручну роботу з вебзастосунком та виконують всі поставленні задачі до проекту.

3.5 Інтерфейс користувача вебзастосунку

Для зручного використання вебзастосунку, інтерфейс ПЗ повинен бути простим та зрозумілим, а також не перенавантаженим елементами. Інтерфейс розроблений з використанням 2 основних кольорів (#D9CAB6 та #F0E0D5). Як вже біло зазначено вище, користувач, який не авторизувався, не може користуватися функціоналом вебзастосунку, тому першими сторінками, які побачить користувач, будуть сторінка реєстрації та авторизації, зображені на рис. 3.12 , 3.13.

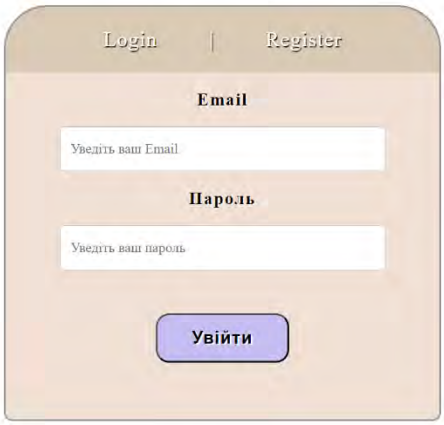


Рисунок 3.12 – Вебсторінка авторизації

Рисунок 3.13 – Вебсторінка реєстрації

Після авторизації користувач потрапляє на головну сторінку сайту (рис. 3.14).

Рисунок 3.14 – Головна вебсторінка

Користувачу стає доступна сторінка профілю (рис. 3.15).

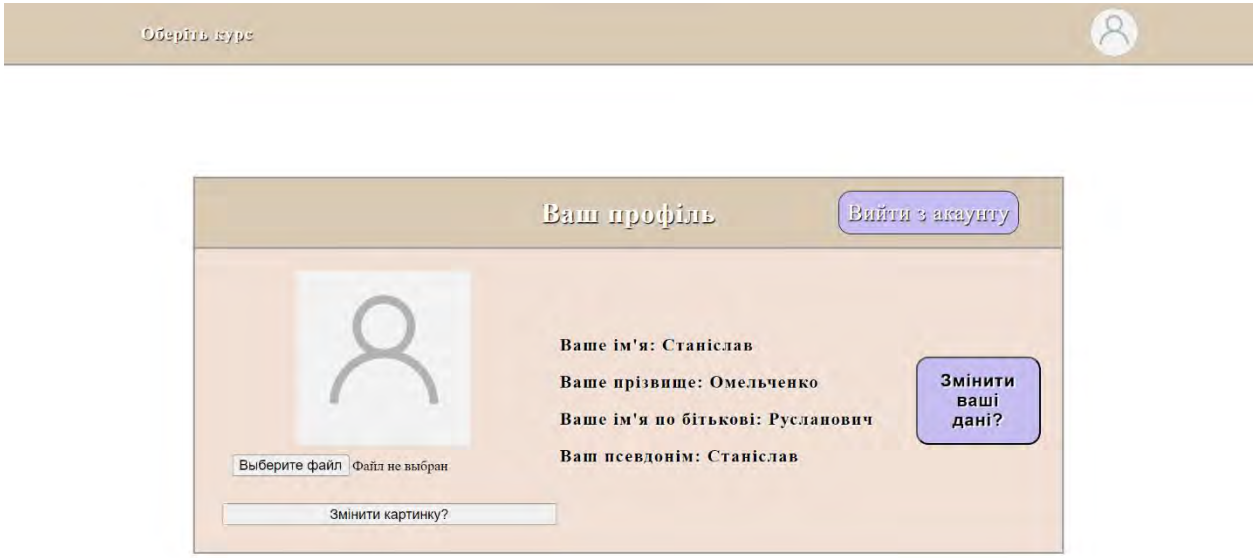


Рисунок 3.15 – Вебсторінка профілю користувача

А також, після авторизації вебзастосунок надає доступ до вебсторінки створення курсу (рис. 3.16) та вебсторінка приєднання до існуючого курсу (рис. 3.17).

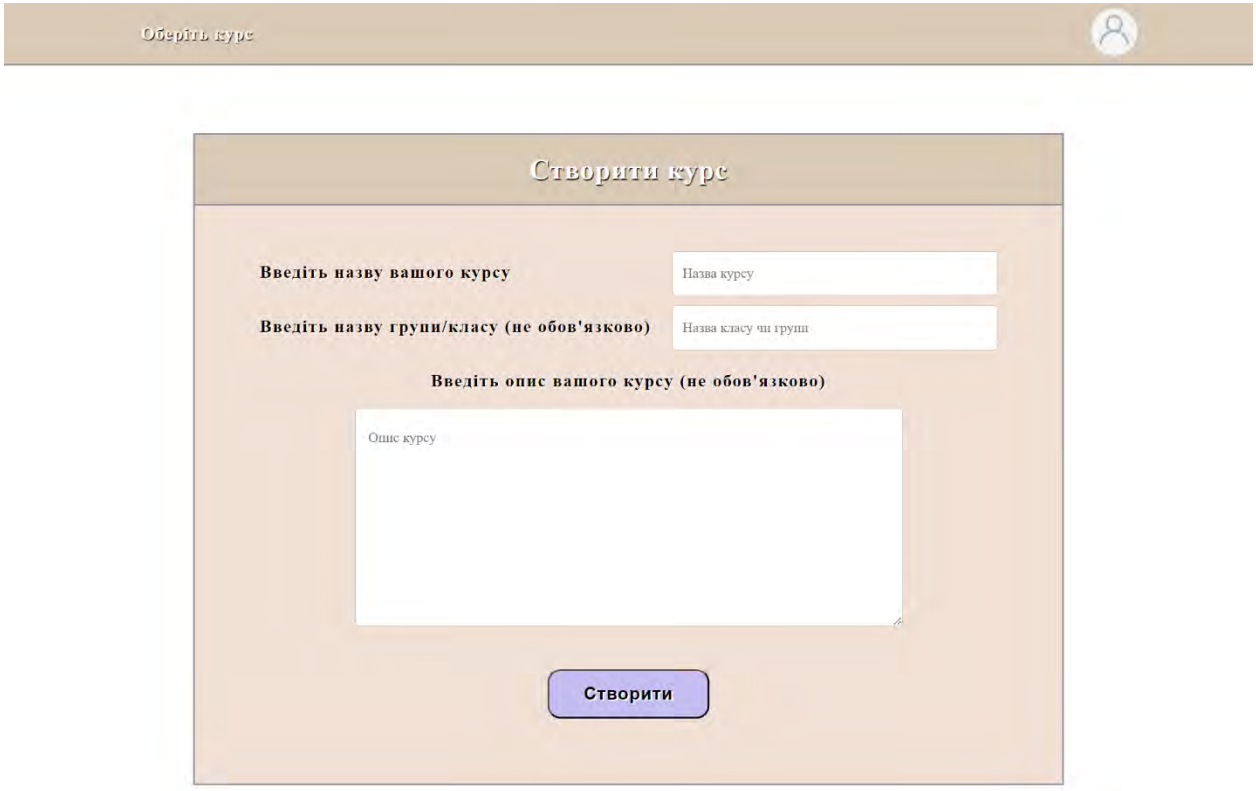


Рисунок 3.16 – Вебсторінка створення курсу

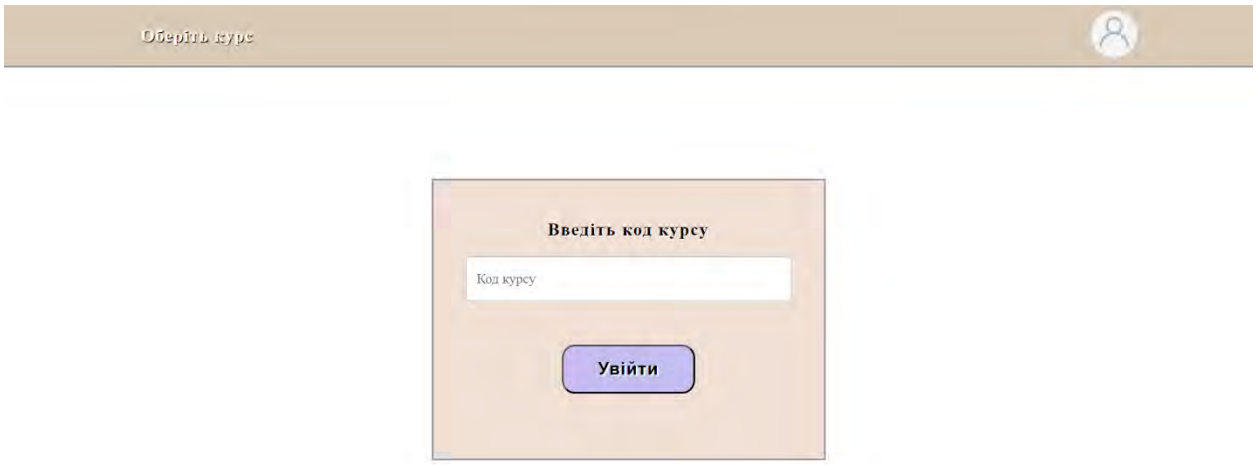


Рисунок 3.17 – Вебсторінка приєднання до курсу

Користувачі, які підключились до існуючого курсу, мають доступ до навігації по вебсторінкам курсу. Головна сторінка курсу зображена на рисунку 3.18.

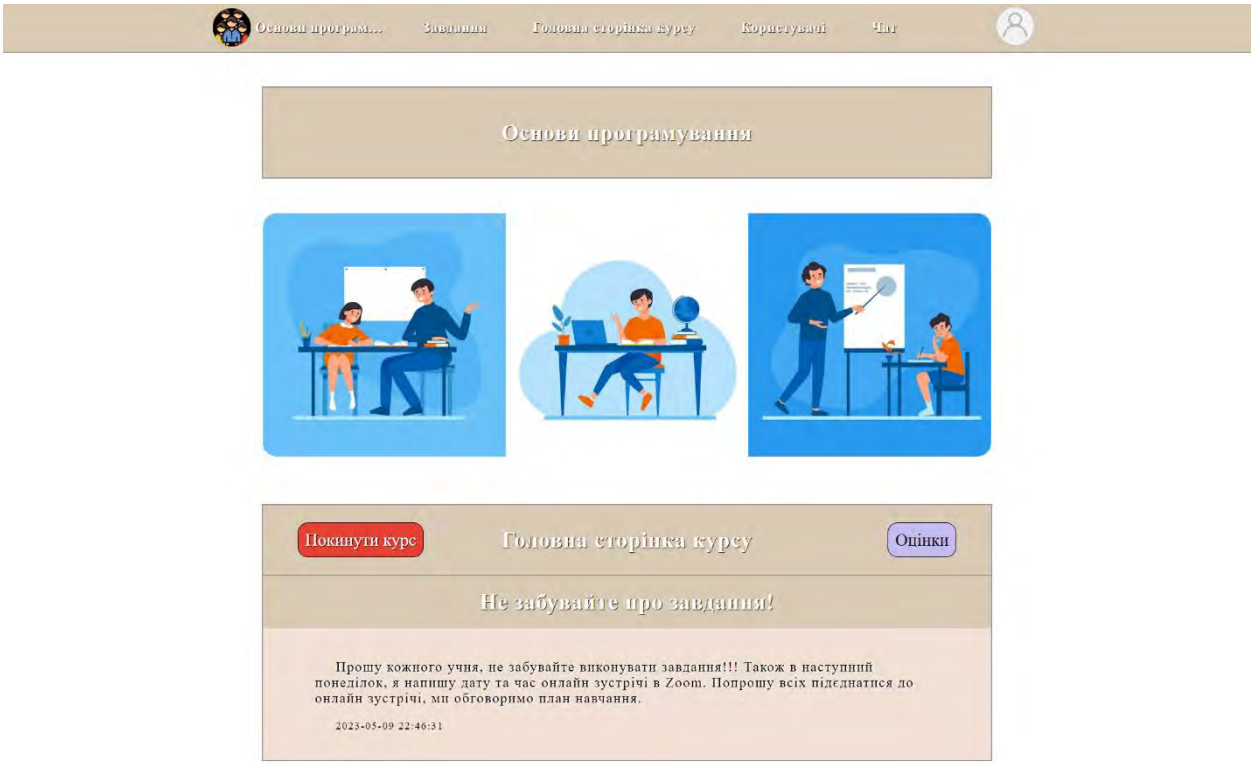


Рисунок 3.18 – Головна вебсторінка курсу

Сторінка, що забезпечує систему комунікації зображена на рисунку 3.19.

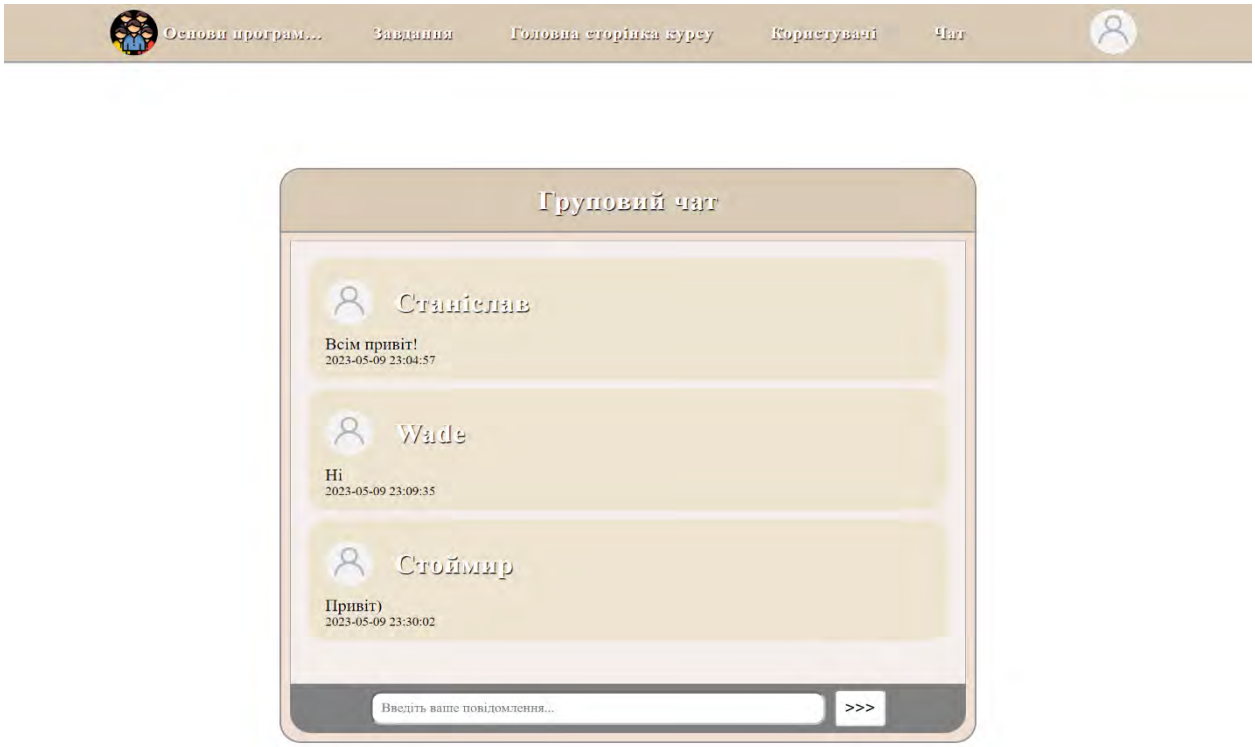


Рисунок 3.19 – Вебсторінка чату

Також, учаснику курсу доступна сторінка з завданнями див. рис. 3.20.

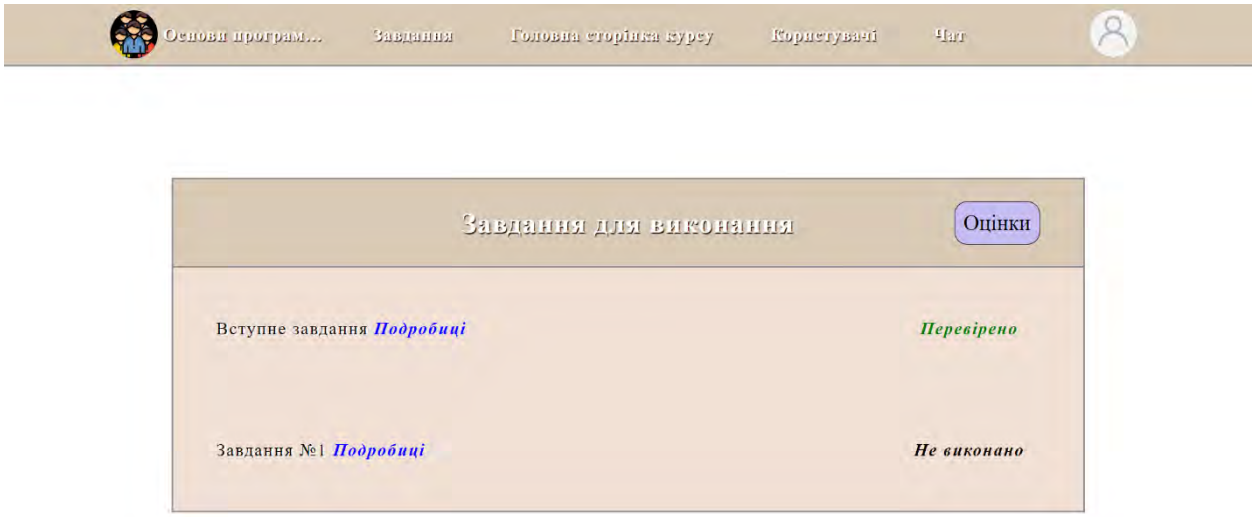


Рисунок 3.20 – Вебсторінка з завданнями

Учаснику курсу має доступ до сторінок з інформацією про учасників курсу (рис. 3.21), а також з поточними оцінками учасників (рис. 3.22). Оцінки виставляє вчитель при перевірці відповіді учня.

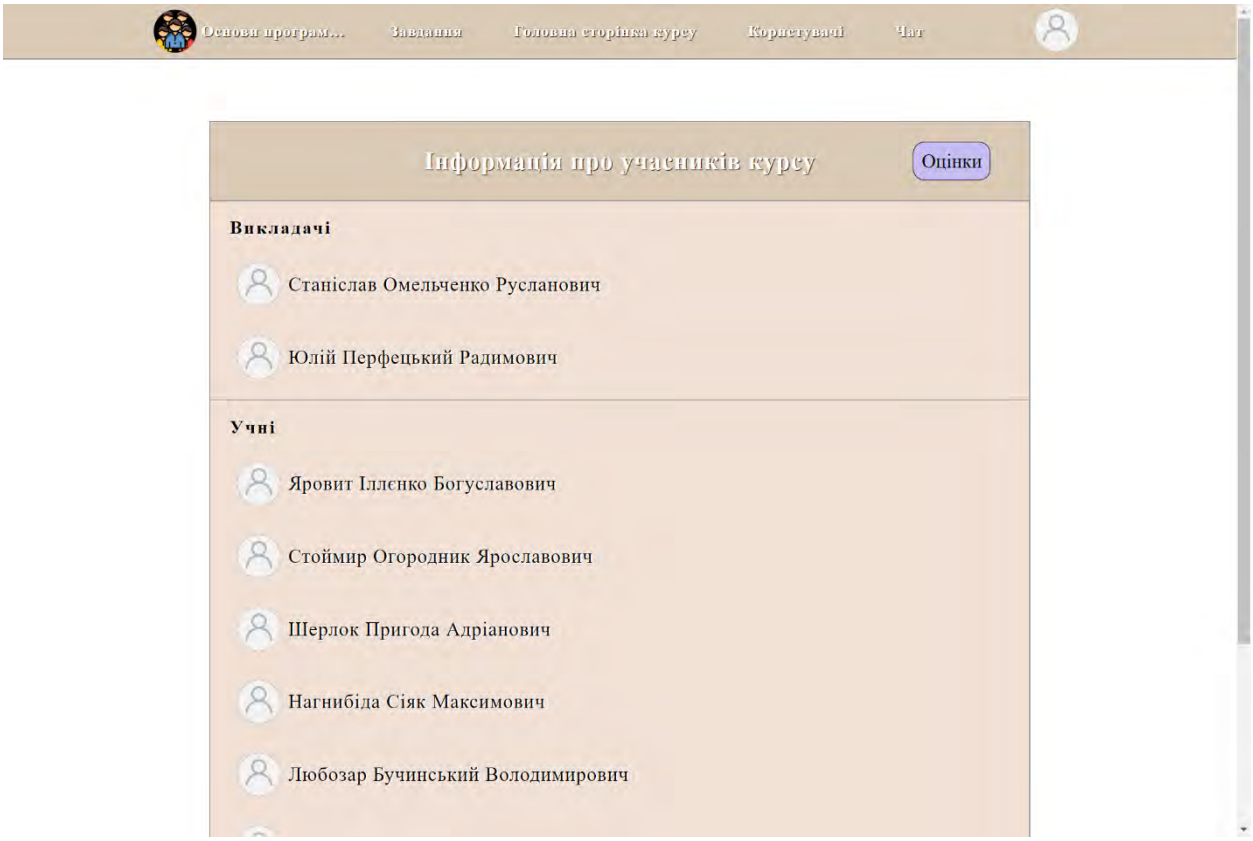
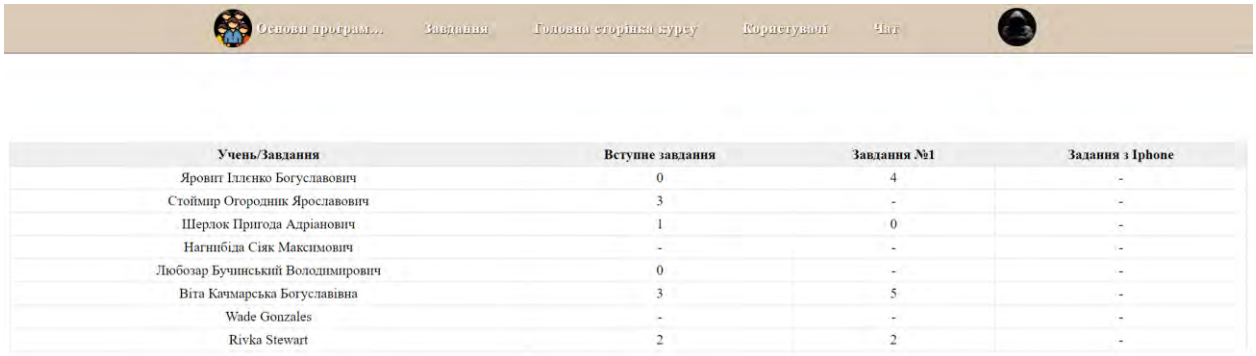


Рисунок 3.21 – Вебсторінка учасників курсу



Учень/Завдання	Вступне завдання	Завдання №1	Задання з Iphone
Яровит Ілліенко Богуславович	0	4	-
Стоймир Огородник Ярославович	3	-	-
Шерлок Пригода Адріанович	1	0	-
Нагнибіда Сіяк Максимович	-	-	-
Любозар Бучинський Володимирович	0	-	-
Віта Качмарська Богуславівна	3	5	-
Wade Gonzales	-	-	-
Rivka Stewart	2	2	-

Рисунок 3.22 – Вебсторінка з оцінками учасників курсу

Сторінки для роботи з завданнями наведені на рис. 3.23, 3.24.

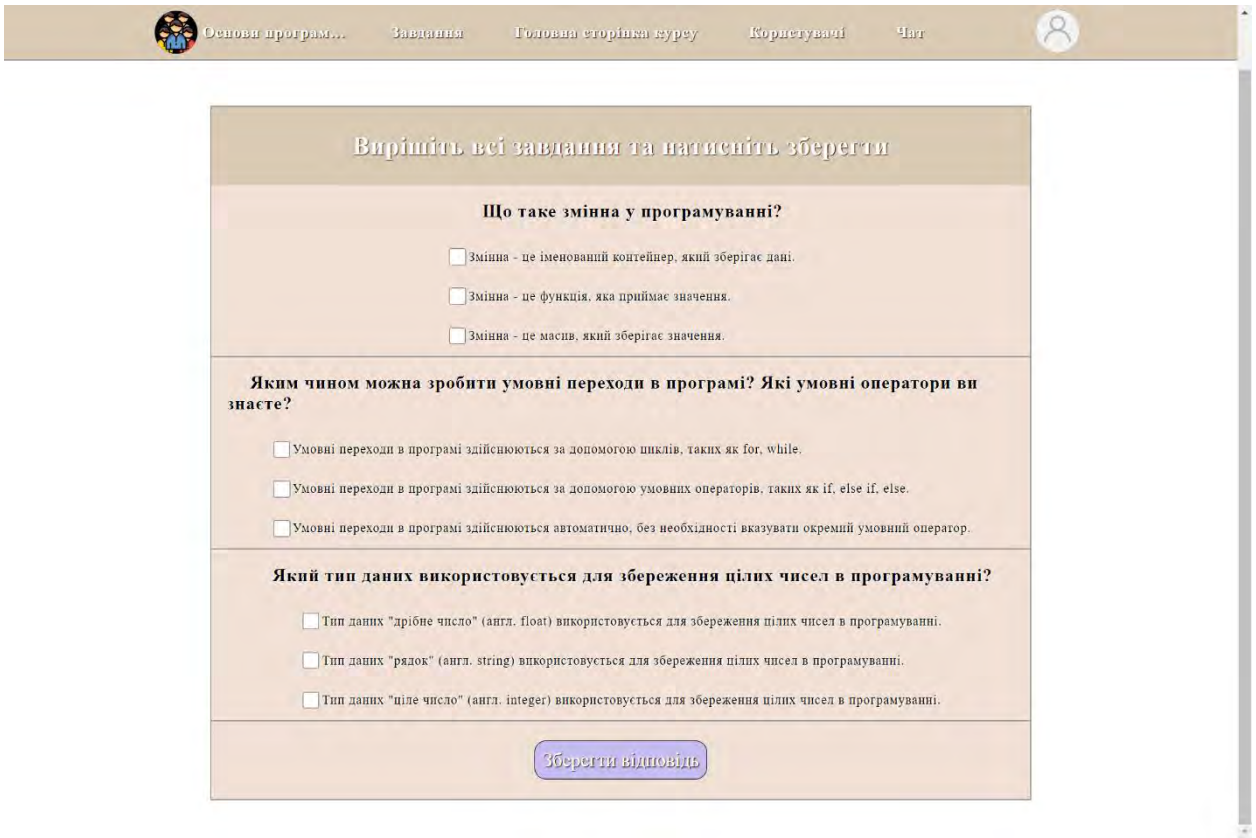


Рисунок 3.23 – Вебсторінка для створення відповіді на завдання

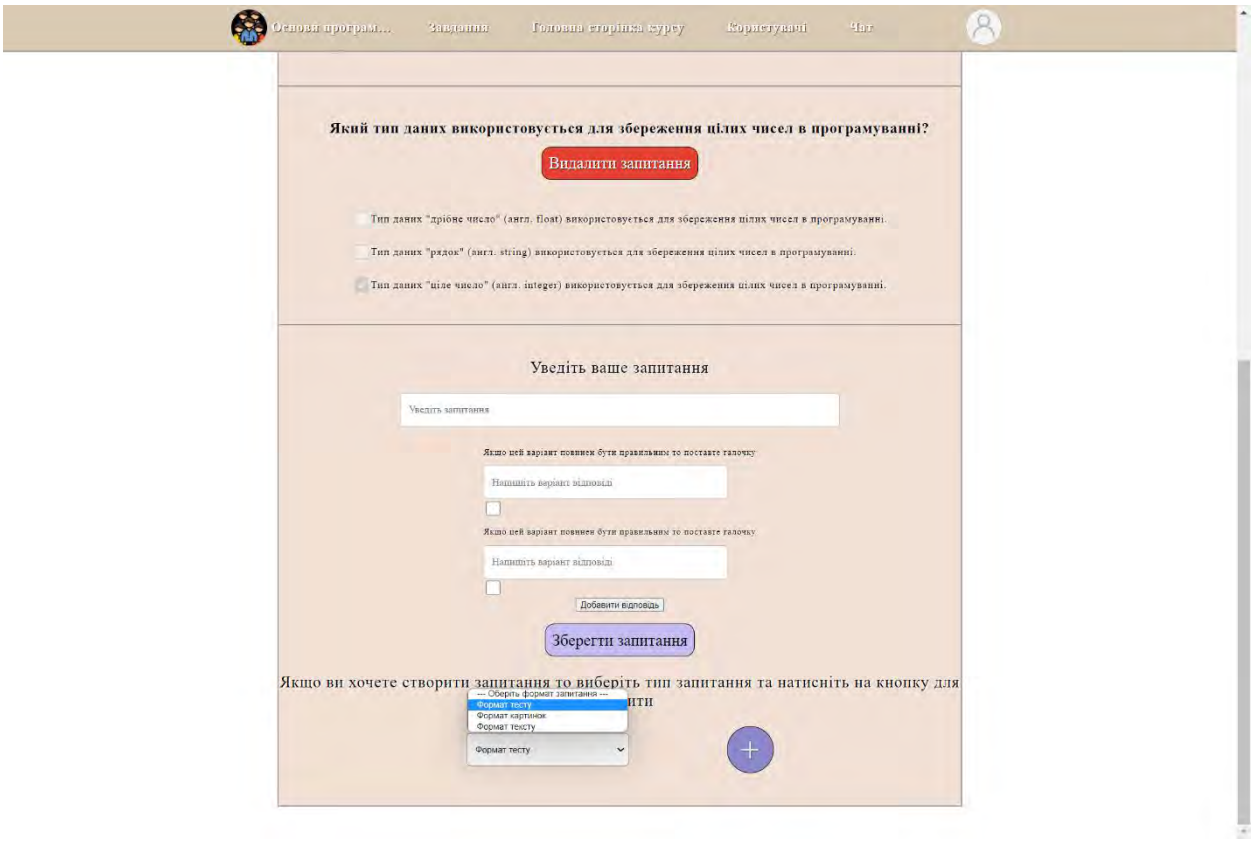


Рисунок 3.24 – Вебсторінка для створення завдання

Викладач також має можливість створювати оголошення. Форма для створення оголошень знаходиться на головній сторінці курсу (рис. 3.25).

Рисунок 3.25 – Форма для створення оголошень

Так, як у викладача є функція перевірки відповідей. Для цих функцій створено 2 вебсторінки: для перегляду всіх відповідей (рис. 3.26) та сторінка з обраною відповіддю (рис. 3.27). На сторінці з обраною відповіддю викладач перевіряє виконанні завдання та виставляє оцінку, яка зберігається в БД. А потім, як вже було зазначено вище, відображається на сторінці з оцінками учасників курсу.

Рисунок 3.26 – Вебсторінка для перевірки відповідей

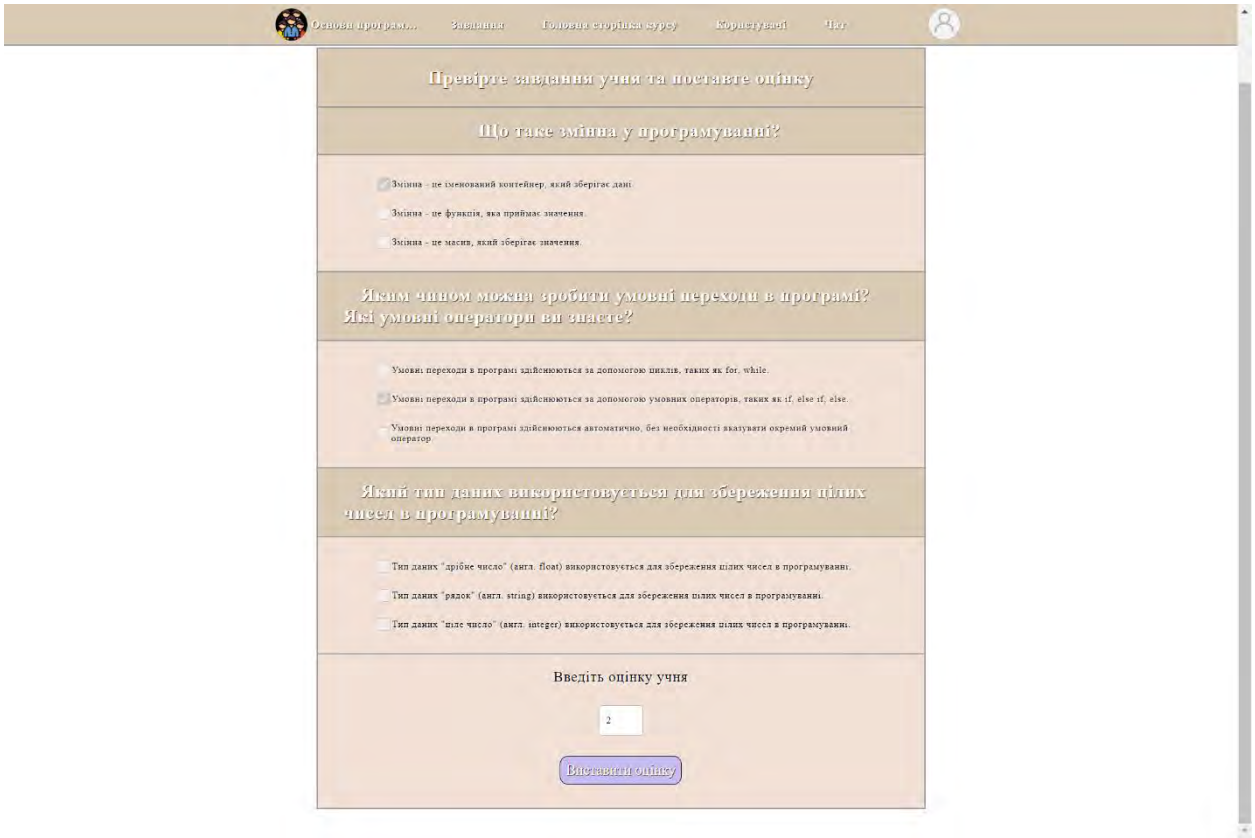


Рисунок 3.27 – Вебсторінка для перевірки конкретної відповіді

Представлення всіх функцій та можливостей вебзастосунку в графічному відображенні надає наглядний приклад роботи вебзастосунку. Завдяки використанню бібліотеки Freemarker було досягнуто відображення різних вебелементів для різних ролей користувачів курсу (організатор, вчитель, учень). Створений інтерфейс надає можливості всім користувачам з легкістю користуватися ПЗ завдяки своїй простоті та не завантаженості різноманітними елементами.

4 ТЕСТУВАННЯ ТА ЕКСПЛУАТАЦІЯ СТВОРЕНОГО ВЕБЗАСТОСУНКУ

4.1 Призначення та умови застосування програми

Вебзастосунок призначений для використання користувачем в роботі з курсами, а саме бути на боці вчителя або учня. Створене програмне забезпечення для онлайн-навчання надає можливості створювати курси, приєднуватися до існуючих курсів, створювати завдання, відповідати на завдання, надає можливість перевіряти виконані завдання вчителю та оцінювати їх, а також створювати об’яви. Також вебзастосунок має в наявності простий та зручний чат.

Для експлуатації вебзастосунку потрібно мати, як мінімум, таке апаратне забезпечення:

- не менше ніж 2 ГБ оперативної пам’яті;
- процесор Intel Dual-Core з тактовою частотою1.8 ГГц і вище ;
- монітор та маніпулятор типу «миша»;
- браузер для роботи з вебзастосунком.

4.2 Локальний сервер

Вебзастосунок запускається на локальному сервері. При створенні Spring Boot проекту, по стандарту, вже було отримано файл, в якому був створений головний файл, який запускає повністю вебзастосунок:

```
@SpringBootApplication
public class OnlineLearningApplication {
   public static void main(String[] args) {
      SpringApplication.run(OnlineLearningApplication.class,
args);
   }
}
```

Проте, для підключення БД даних та підключення Freemarker, було змінено наповнення файлу application.properties, в який було додано наступні поля:

```
spring.datasource.url=jdbc:mysql://localhost:3306/onlinelearning
spring.datasource.username=root
spring.datasource.password=admin
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.freemarker.expose-request-attributes=true
spring.servlet.multipart.max-file-size=100MB
spring.servlet.multipart.max-request-size=100MB
```

Тому для запуску програмного забезпечення потрібно відкрити в IntelliJ IDEA головний файл ПЗ OnlineLearningApplication та запустити його. Після запуску в консолі з’явиться повідомлення про успішний запуск програми на локальному сервері з портом 8080:

```
Tomcat started on port(s): 8080 (http) with context path ''
Started OnlineLearningApplication in 4.28 seconds (JVM running for
4.647).
```

4.3 Тестування вебзастосунку

Для проведення тестування ПЗ було створено 10 користувачів курсу, наведених на рис. 4.1. Всі користувачі були приєднані до одного курсу, в якому було створено декілька завдань. Деякі з учнів виконали всі завдання та отримали вже оцінки, а деякі ще ні. Також одному з учасників було змінено роль учня на роль вчителя.

id	email	familyname	name	nikname	password	surname	image_id
1	stas665544@gmail.com	Русланович	Станіслав	Станіслав	$2a$08$nrbwiCoxZdgkGxHVv/o7rO93piC1APtY...	Омельченко	251
2	user@gmail.com	Радимович	Юлій	Юлій Радимович	$2a$08$K8tOpGySqZ8sfqblk45Zwe1KFATsg76...	Перфецький	252
3	ukr@gmail.com	Богуславович	Яровит	Яровит	$2a$08$pJ/AP1esfWptXnaa4oFaFOFAzm84TC...	Ілленко	253
4	ukr1@gmail.com	Ярославович	Стоймир	Стоймир	$2a$08$E37WqgORHZn/KF05uZzyCeXW/Xu4rT...	Огородник	254
5	ukr2@gmail.com	Адріанович	Шерлок	Шерлок	$2a$08$b0xdETs/hOZfcSGZJk6E/ehzymIpla7hO...	Пригода	255
6	ukr3@gmail.com	Максимович	Нагнибіда	Нагнибіда	$2a$08$dnGn00v3Cw/gN.gDdbR.6.1tqPe/ja0D...	Сіяк	256
7	ukr4@gmail.com	Володимирович	Любозар	Любозар	$2a$08$fgauxO4Y3TgGFzywCNvuy.NUjHnnV.4...	Бучинський	NULL
8	eng@gmail.com		Wade	Wade	$2a$08$eci.Sw7Iq.l/Eimuf9uNauEk0zSEqy14oTt...	Gonzales	NULL
9	eng1@gmail.com		Rivka	Rivka	$2a$08$hyDVN7sOKL5FLFzo67s.xOCx.9P7iVE0...	Stewart	258
10	ukr5@gmail.com	Богуславівна	Віта	Vi_)	$2a$08$NK9s2JcTLRcHVkELsyURSezjtVZXFBM3...	Качмарська	257
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 4.1 – Таблиця створених користувачів

Також для перевірки всіх функцій, окрім комп'ютера, було використано смартфон на ОС IOS. Для того, щоб перевірити вебзастосунок на смартфоні, треба спочатку бути підключеним до однієї інтернет-мережі, що і сервер. Після чого потрібно запустити ПЗ на пристрої, який в даному випадку є сервером, та підключитися до локального серверу через веббраузер, вказавши IP-адрес пристрою, на якому було запущено вебзастосунок.

Щоб дізнатися IP-адресу, потрібно відкрити командний рядок та ввести команду ipconfig. Потім в полі Ipv4 адреси ми побачимо ту адресу, яка нам потрібна. Тому на смартфоні ми переходимо за цією адресою та додаємо порт 8080, який, як вже було зазначено вище, надається при запуску вебзастосунку.

Приклад пошуку IP-адресу, зображений на рисунку 4.2.

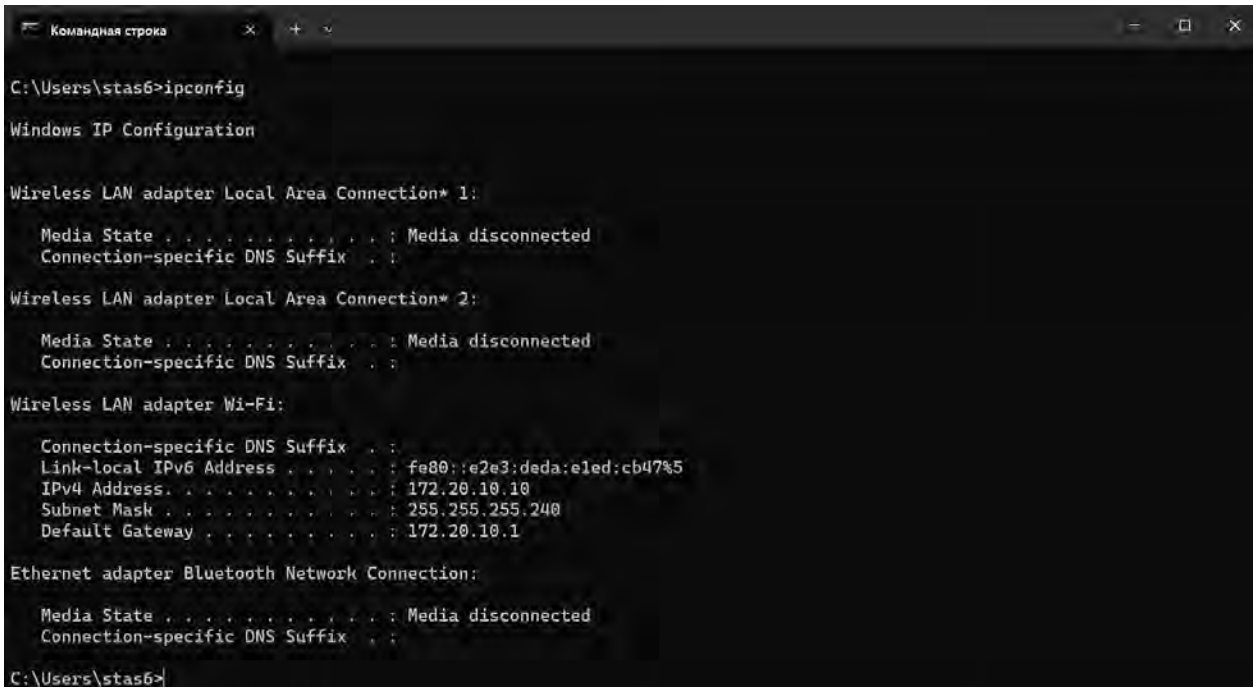


Рисунок 4.2 – Пошук IP-адресу

Дослідивши працездатність застосунку на мобільному пристрої, можна сказати, що всі функції ПЗ працюють коректно при роботі на локальному сервері. Головна вебсторінка курсу на мобільному пристрої зображена на рисунку 4.3.

Рисунок 4.3 – Головна вебсторінка курсу

Вебсторінка завдань курсу та вебсторінка учасників курсу, зображені на рис. 4.4, 4.5.

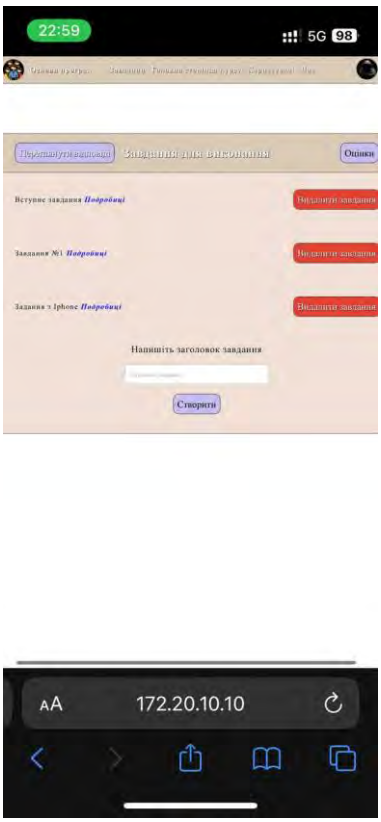
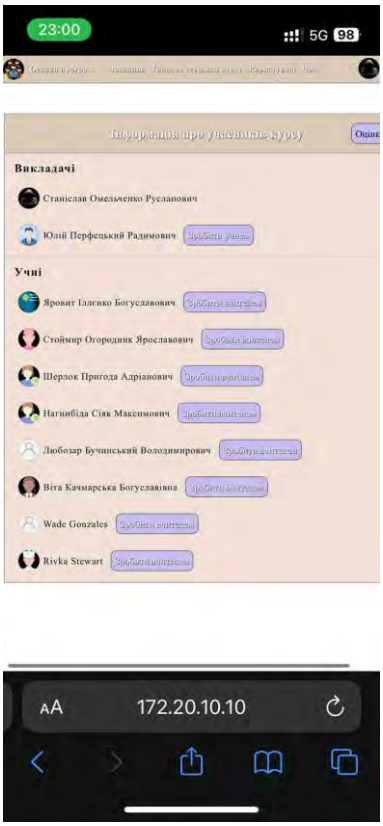


Рисунок 4.4 – Вебсторінка завдань курсу



Рисунок, 4.5 – Вебсторінка учасників курсу

Вебсторінка призначена для комунікації учасників курсу зображена на рисунку 4.6.

Рисунок 4.6 – Вебсторінка чату

Вебсторінка створення завдання наведена на рис. 4.7.

Рисунок 4.7– Вебсторінка створення завдання

Вебсторінка для перевірки відповідей наведена на рис. 4.8.

Рисунок 4.8 – Вебсторінка для перевірки відповідей

Вебсторінка для перевірки обраної відповіді зображена на рис. 4.9, а також вебсторінка для перегляду інформацію про оцінки зображена на рис. 4.10.

Рисунок 4.9 – Вебсторінка перевірки обраної відповіді

Рисунок 4.10 – Вебсторінка таблиці з оцінками учнів

Протестувавши всі функціональні можливості вебзастосунку на різних пристроях, можна сказати, що вебзастосунок виконує всі поставлені раніше задачі. Тому стане в нагоді багатьом викладачам для створення онлайн курсу.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра, було детально розглянуто теоретичні відомості предметної області, а також ряд застосунків, які реалізують схожі задачі. Виконавши аналіз мов програмування, для поставлених задач було обрано мову програмування Java та фреймворк Spring Boot з базою даних MySQL.

В рамках роботи розроблено вебзастосунок для онлайн-навчання.

Створений вебзастосунок має можливість реєстрації та авторизації користувачів. Була реалізована система розподілу користувачів вебзастосунку на окремі ролі в кожному курсі. Тому кожен користувач зможе мати роль викладача, учня , чи організатор.

Вебзастосунок надає можливість учаснику з роллю «Викладач» створювати оголошення, завдання, а також надає можливості перевірки та оцінювання відповідей учнів на ці завдання. В свою чергу учасники курсу з роллю «Учень» можуть виконувати завдання, присутні в курсі.

Також вебзастосунок включає в себе систему комунікації, тобто чат. Чат створений для легкої комунікації учасників курсу в реальному часі.

Було виконано проектування користувацького інтерфейсу для взаємодії користувача з вебзастосунком. Також було проведено тестування усіх функцій вебзастосунку, яке включало тестування на різних пристроях, включаючи смартфон на ОС IOS.

Розроблений вебзастосунок виконує всі поставленні задачі та функціональні вимоги. Він дозволяє реалізувати процес онлайн-навчання достатньо легко, надаючи можливість користувачам взаємодіяти між собою не тільки на рівні перевірки результатів завдань, але й з можливістю спілкування за допомогою чату, що полегшує процес комунікації.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Inventa [Електрон. Ресурс]. – Режим доступу: https://www.inventa.ua/ua/blog/910-z-istorii-stvorennia-vsesvitnoi-pavutyny-world-wide-web/

2. Internetsociety [Electronic resource]. – Access mode: https://www.internetsociety.org/internet/history-internet/brief-history-internet/

3. Поняття, структура та різновиди веб-сайтів. Автоматизоване розроблення веб-сайтів [Електрон. Ресурс]. – Режим доступу: http://www.ndu.edu.ua/liceum/web.pdf

4. Webcase [Електрон. Ресурс]. – Режим доступу: https://webcase.com.ua/uk/blog/cho-takoe-web-prilozhenie-vse-vidy/

5. Дистанційне навчання в системі вищої освіти: сучасні тенденції [Електрон. Ресурс] / Ярошенко Т. О. – Режим доступу: https://ekmair.ukma.edu.ua/server/api/core/bitstreams/442f2370-8e31-494c-8c3d-340dc169349b/content

6. Classroom [Electronic resource]. – Access mode: https://classroom.google.com/

7. Система електронного навчання ВНЗ на базі MOODLE: Методичний посібник / Ю. В. Триус, І. В. Герасименко, В. М. Франчук // Ю. В. Триус, І. В. Герасименко, В. М. Франчук // За ред. Ю. В. Триуса. – Черкаси. – 220 с. [Електрон. Ресурс]. – Режим доступу: https://moodle.org/pluginfile.php/1968620/mod_resource/content/1/%D0%A2%D1%80%D0%B8%D1%83%D1%81%20%D0%A1%D0%95%D0%9D%20%D0%92%D0%9D%D0%97%20Moodle%202013.pdf?__cf_chl_tk=eOKdHEIEcf0p1oRPwKMx0vW88llfdIodMWlHzrC3LJQ-1682884423-0-gaNycGzNDjs

8. MoodleCloude [Electronic resource]. – Access mode: https://moodlecloud.com/app/en/login

9. Edapp [Electronic resource]. – Access mode: https://www.edapp.com/

10. Classdojo [Electronic resource]. – Access mode: https://www.classdojo.com/

11. Wezom [Електрон. Ресурс]. – Режим доступу: https://wezom.com.ua/ua/blog/nodejs-protiv-java-chto-vybrat

12. Medium. Rails, Django, and Node.js [Electronic resource]. – Access mode: https://medium.com/@schulte.robert/rails-django-and-node-js-a-primer-on-web-frameworks-9c734dc1bd2c

13. Hi-news [Електрон. Ресурс]. – Режим доступу: https://hi-news.pp.ua/kompyuteri/14786-postgresql-vs-mysql-porvnyannya-osoblivost-ta-vdguki.html

14. Spring Initializr [Electronic resource]. – Access mode: https://start.spring.io/

15. Spring Security [Electronic resource]. – Access mode: https://spring.io/projects/spring-security

16. Spring Security Kerberos - Reference Documentation [Electronic resource]. – Access mode:https://docs.spring.io/spring-security-kerberos/docs/1.0.1.RELEASE/reference/htmlsingle/

17. Spring. Using WebSocket to build an interactive web application [Electronic resource]. – Access mode: https://spring.io/guides/gs/messaging-stomp-websocket/

18. Adly [Electronic resource]. – Access mode: https://ably.com/periodic-table-of-realtime/sockjs

19. Wikipedia [Electronic resource]. – Access mode: https://en.wikipedia.org/wiki/Streaming_Text_Oriented_Messaging_Protocol

ДОДАТОК А

Технічне завдання

Вступ

Вебзастосунок для онлайн-навчання надає незалежним слухачам і учасникам навчального процесу додатковий інструмент для організації навчання.

А.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Розробка вебзастосунку для онлайн-навчання», затверджене наказом Національного університету «Запорізька політехніка» №87 від 4 квітня 2023 р.

А.2 Призначення розробки

Вебзастосунок для онлайн-навчання призначений для ведення навчального процесу в дистанційному режимі.

А.3 Вимоги до програмного забезпечення

А.3.1 Вимоги до функціональних характеристик

Вебзастосунок для онлайн-навчання повинен виконувати такі функціональні задачі:

- система реєстрації;
- створення курсів;
- можливість приєднуватись до курсу за посиланням;
- редагування профілю користувача;
- розділ учасників курсу на ролі;
- додавання до курсу завдань різного типу (текстовий, тестовий, та візуальний);
- створення в курсі оголошень;
- наявність системи оцінювання учасників курсу;

- наявність можливості комунікації.

А.3.2 Вимоги до надійності

Розроблене програмне забезпечення повинно розпізнавати не коректні вхідні дані користувача, та обробку збійних ситуацій. Також включати механізм захисту від несанкціонованого виконання дій за допомогою маніпуляцій з формами, а також пропускна можливість програмного забезпечення повинна витримувати суттєві навантаження запитів користувачів.

А.3.3 Вимоги до інтерфейсу програми

Інтерфейс програмного забезпечення повинен бути простим та легким в освоєнні, а також не повинен бути перенавантажений різноманітними вебелементами.

А.3.4 Вимоги до технічних параметрів

Для експлуатації вебзастосунку потрібно мати, як мінімум, таке апаратне забезпечення:

- Не менше ніж 2 ГБ оперативної пам’яті;
- процесор Intel Dual-Core з тактовою частотою1.8 ГГц і вище ;
- монітор та маніпулятор типу «миша»;
- браузер для роботи з вебзастосунком.

А.3.5 Вимоги до маркування й пакування

Програмне забезпечення для онлайн-навчання може бути записане на носії. На пакуванні потрібно вказати назву програми – “Розробка вебзастосунку для онлайн-навчання”.

ДОДАТОК Б

Текст програми

Б.1 – Модуль Controllers

Б.1.1 – Модуль CourseController.java

```
package com.example.onlinelearning.Controllers;
import com.example.onlinelearning.Models.*;
import com.example.onlinelearning.Service.CourseService;
import com.example.onlinelearning.Service.UserCourseService;
import com.example.onlinelearning.Service.UserQuesrionService;
import com.example.onlinelearning.Service.UserService;
import lombok.RequiredArgsConstructor;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.multipart.MultipartFile;
import java.io.IOException;
import java.security.Principal;
@Controller
@RequiredArgsConstructor
public class CourseController {
    private final UserService userService;
    private final CourseService courseService;
    private final UserCourseService userCourseService;
    private final UserQuesrionService userQuesrionService;
    @GetMapping("/course/{id}")
    public String CourseMainPage(Principal principal, Model model,
@PathVariable Long id){
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);
        boolean ishave= false;
        for(UserCourse userCourse: user.getUserCourses()){
            if(userCourse.getCourse().getId()==id)ishave=true;
        }
        if(!ishave)return "redirect:/";
        model.addAttribute("course",course);
        model.addAttribute("user",user);
        model.addAttribute(
"role",userCourseService.getUserRole(course,user));
        return "MainCoursePage";
    }
    @PostMapping("/course/{id}/addimage")
    public String AddImageCourse(@RequestParam("file") MultipartFile file,
@PathVariable Long id) throws IOException {
        courseService.saveCourseImageByID(id,file);
        return "redirect:/course/"+id;
    }
    @PostMapping("/course/{id}/addlogo")
    public String AddLogoCourse(@RequestParam("file") MultipartFile file,
@PathVariable Long id) throws IOException {
        courseService.saveCourseLogoByID(id,file);
        return "redirect:/course/"+id;
    }
    @GetMapping("/course/{id}/users")
    public String UsersPage(@PathVariable Long id,Model model,Principal
principal){
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);
        boolean ishave= false;
        for(UserCourse userCourse: user.getUserCourses()){
            if(userCourse.getCourse().getId()==id)ishave=true;
        }
```

```
        if(!ishave)return "redirect:/";
        model.addAttribute("course",course);
        model.addAttribute("user",user);

model.addAttribute("admins",userCourseService.getOnlyAdminsByCourse(course));

model.addAttribute("users",userCourseService.getOnlyUsersByCourse(course));
        model.addAttribute(
"role",userCourseService.getUserRole(course,user));
        return "UsersPage";
    }
    @PostMapping("/course/{id}/{usercourseid}/changerole")
    public String ChangeRole(@PathVariable Long id,@PathVariable Long
usercourseid){

userCourseService.changeUserRole(courseService.getCourseByID(id),userCourseSe
rvice.getUserCourseByID(usercourseid));
        return "redirect:/course/"+id+"/users";
    }
    @PostMapping("/course/{id}/delete")
    public String DeleteCourse(@PathVariable Long id,Principal principal){
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);

if("ROLE_CREATER".equals(userCourseService.getUserRole(course,user))){
            for(Question question: course.getQuestions()){
                userQuesrionService.deleteQuestionByQuestion(question);
            }
            userCourseService.deleteCourseByCourse(course);
            return "redirect:/";
        }
        return "redirect:/course/"+id;
    }
    @PostMapping("/course/{id}/conectdelete")
    public String DeleteConetctCourse(@PathVariable Long id,Principal
principal){
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);

if("ROLE_USER".equals(userCourseService.getUserRole(course,user))||"ROLE_ADMI
N".equals(userCourseService.getUserRole(course,user))){
            userCourseService.deleteUserCourse(user,course);
            return "redirect:/";
        }
        return "redirect:/course/"+id;
    }
    @PostMapping("/course/{id}/createarticle")
    public String CreateArticle(@PathVariable Long id, Article article){
        Course course = courseService.getCourseByID(id);
        courseService.createArticleByCourse(course,article);
        return "redirect:/course/"+id;
    }
    @PostMapping("/course/{id}/{idarticle}/deletearticle")
    public String DeleteArticle(@PathVariable Long id,@PathVariable Long
idarticle,Principal principal){
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);
        String role =userCourseService.getUserRole(course,user);
        if("ROLE_CREATER".equals(role)||"ROLE_ADMIN".equals(role)){
            courseService.deleteArticle(course,idarticle);
            return "redirect:/course/"+id;
        }
        return "redirect:/course/"+id;
    }
    @GetMapping("/course/{id}/ratings")
    public String RankingPage(@PathVariable Long id, Model model, Principal
```

```
principal){
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);
        boolean ishave= false;
        for(UserCourse userCourse: user.getUserCourses()){
            if(userCourse.getCourse().getId()==id)ishave=true;
        }
        if(!ishave)return "redirect:/";
        model.addAttribute("course",course);
        model.addAttribute("user",user);
        model.addAttribute("questions",course.getQuestions());
        model.addAttribute("users",course.getUserCourses());
        return "RatingsPage";
    }
    @GetMapping("/course/{id}/chat")
    public String ChatPage(@PathVariable Long id, Model model, Principal
principal){
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);
        boolean ishave= false;
        for(UserCourse userCourse: user.getUserCourses()){
            if(userCourse.getCourse().getId()==id)ishave=true;
        }
        if(!ishave)return "redirect:/";
        model.addAttribute("course",course);
        model.addAttribute("user",user);
        model.addAttribute("messages",course.getChatmesseges());
        return "ChatPage";
    }
}
```

Б.1.2 – Модуль ImageController.java

```
package com.example.onlinelearning.Controllers;
import com.example.onlinelearning.Models.Image;
import com.example.onlinelearning.Repository.ImageRepository;
import lombok.RequiredArgsConstructor;
import org.springframework.core.io.InputStreamResource;
import org.springframework.http.MediaType;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RestController;
import java.io.ByteArrayInputStream;
@RestController
@RequiredArgsConstructor
public class ImageController {
    private final ImageRepository imagerepository;
    @GetMapping("/images/{id}")
    private ResponseEntity<?> getImageById(@PathVariable Long id) {
        Image image = imagerepository.findById(id).orElse(null);
        return ResponseEntity.ok()
                .header("fileName", image.getOriginalFileName())
                .contentType(MediaType.valueOf(image.getContentType()))
                .contentLength(image.getSize())
                .body(new InputStreamResource(new
ByteArrayInputStream(image.getBytes())));
    }
}
```

Б.1.3 – Модуль QuestionController.java

```
package com.example.onlinelearning.Controllers;
import com.example.onlinelearning.Models.*;
import com.example.onlinelearning.Service.*;
import lombok.RequiredArgsConstructor;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;
import java.io.IOException;
import java.security.Principal;
@Controller
@RequiredArgsConstructor
public class QuestionController {
    private final UserService userService;
    private final CourseService courseService;
    private final UserCourseService userCourseService;
    private final QuestionService questionService;
    private final UserQuesrionService userQuesrionService;
    @GetMapping("/course/{id}/question")
    public String QuestionPage(@PathVariable Long id, Model model, Principal
principal){
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);
        boolean ishave= false;
        for(UserCourse userCourse: user.getUserCourses()){
            if(userCourse.getCourse().getId()==id)ishave=true;
        }
        if(!ishave)return "redirect:/";
        model.addAttribute("course",course);
        model.addAttribute("user",user);
        model.addAttribute(
"role",userCourseService.getUserRole(course,user));

model.addAttribute("userQuestions",userQuesrionService.getAllUserQuestionByUs
er(user));
        return "Questions";
    }
    @GetMapping("/course/{id}/usersanswer")
    public String UsersAnswerPage(@PathVariable Long id, Model model,
Principal principal){
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);
        String role =userCourseService.getUserRole(course,user);
        boolean ishave= false;
        for(UserCourse userCourse: user.getUserCourses()){
            if(userCourse.getCourse().getId()==id)ishave=true;
        }
        if(!ishave)return "redirect:/course/"+id+"/question";
        if("ROLE_CREATER".equals(role)||"ROLE_ADMIN".equals(role)){
            model.addAttribute("course",course);
            model.addAttribute("user",user);
            return "UsersAnswer";
        }
        else {
            return "redirect:/course/"+id+"/question";
        }
    }
    @GetMapping("/course/{id}/usersanswer/{userid}/{questionid}")
    public String UserAnswerPage(@PathVariable Long id,@PathVariable Long
userid,@PathVariable Long questionid, Model model, Principal principal) {
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);
        Question question = questionService.getQuestionByID(questionid);
```

```
        String role =userCourseService.getUserRole(course,user);
        boolean ishave= false;
        for(UserCourse userCourse: user.getUserCourses()){
            if(userCourse.getCourse().getId()==id)ishave=true;
        }
        if(!ishave)return "redirect:/course/"+id+"/question";
        if("ROLE_CREATER".equals(role)||"ROLE_ADMIN".equals(role)){
            model.addAttribute("course",course);
            model.addAttribute("user",user);
            model.addAttribute("tasks",question.getTasks());

model.addAttribute("userquestion",userQuesrionService.getUserQuestionByUserAn
dQuestion(userService.getUserByID(userid),question));
            return "UserAnswer";
        }
        else {
            return "redirect:/course/"+id+"/question";
        }
    }
    @PostMapping("/course/{id}/usersanswer/{userid}/{questionid}")
    public String AddRaning(@PathVariable Long id,@PathVariable Long
userid,@PathVariable Long questionid,@RequestParam("rating") String rating){
        Question question = questionService.getQuestionByID(questionid);
        UserQuestion userQuestion =
userQuesrionService.getUserQuestionByUserAndQuestion(userService.getUserByID(
userid),question);
        userQuestion.setRating(rating);
        userQuestion.setStatus("done");
        userQuesrionService.save(userQuestion);
        return "redirect:/course/"+id+"/usersanswer";
    }
    @PostMapping("/course/{id}/question")
    public String CreateQuestion(@PathVariable Long id, Model model,
Principal principal,@RequestParam("title") String title){
        Question question = new Question();
        question.setTitle(title);
        Course course = courseService.getCourseByID(id);
        course.getQuestions().add(question);
        courseService.saveCourse(course);
        return "redirect:/course/"+id+"/question";
    }
    @GetMapping("/course/{id}/question/{questionid}")
    public String ReinfoQuestion(@PathVariable Long id,@PathVariable Long
questionid ,Model model, Principal principal){
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);
        String role =userCourseService.getUserRole(course,user);
        Question question = questionService.getQuestionByID(questionid);
        if("ROLE_CREATER".equals(role)||"ROLE_ADMIN".equals(role)){
            model.addAttribute("course",course);
            model.addAttribute("user",user);
            model.addAttribute("question",question);
            return "CreateQuestion";
        }
        model.addAttribute("course",course);
        model.addAttribute("user",user);
        model.addAttribute("question",question);

model.addAttribute("userquestion",userQuesrionService.getUserQuestionByUserAn
dQuestion(user,question));
        return "AnswerQuestion";
    }
    @PostMapping("/course/{id}/question/{questionid}/createanswer")
    public String CreateAnwer(@PathVariable Long id,@PathVariable Long
questionid,AnswerList answers, Principal principal) throws IOException {
        User user = userService.getUserByPrincipal(principal);
```

```
        Question question =questionService.getQuestionByID(questionid);

userQuesrionService.createUserQuestion(user,question,answers.getAnswers());
        return "redirect:/course/"+id+"/question";
    }
    @PostMapping("/course/{id}/question/{questionid}/delete")
    public String DeleteQuestion(@PathVariable Long id,@PathVariable Long
questionid){
        Question question = questionService.getQuestionByID(questionid);
        userQuesrionService.deleteQuestionByQuestion(question);
        return "redirect:/course/"+id+"/question";
    }
    @PostMapping("/course/{id}/question/{questionid}/{taskid}/delete")
    public String DeleteTask(@PathVariable Long id, @PathVariable Long
taskid, @PathVariable Long questionid){
        Question question = questionService.getQuestionByID(questionid);
        questionService.deleteTaskById(question,taskid);
        userQuesrionService.reinfoQuestion(question);
        return "redirect:/course/"+id+"/question/"+questionid;
    }
    @PostMapping("/course/{id}/question/{questionid}")
    public String CreateTasktoQuestion(@PathVariable Long id, @PathVariable
Long questionid, ChecktaskList checktasks,@RequestParam("quest") String
quest,@RequestParam("type") String type){
        Question question = questionService.getQuestionByID(questionid);
        Task t = new Task();
        t.setChecktasks(checktasks.getChecktasks());
        t.setQuest(quest);
        t.setType(type);
        question.getTasks().add(t);
        questionService.saveQuestion(question);
        userQuesrionService.reinfoQuestion(question);
        return "redirect:/course/"+id+"/question/"+questionid;
    }
    @PostMapping("/course/{id}/{userquestionid}/deleteuserinfo")
    public String DeleteUserInfo(@PathVariable Long id, @PathVariable Long
userquestionid){
        userQuesrionService.deleteUserQuestionById(userquestionid);
        return "redirect:/course/"+id+"/usersanswer";
    }
}
```

Б.1.4 – Модуль UserController.java

```
package com.example.onlinelearning.Controllers;
import com.example.onlinelearning.Models.Course;
import com.example.onlinelearning.Models.User;
import com.example.onlinelearning.Models.UserCourse;
import com.example.onlinelearning.Service.CourseService;
import com.example.onlinelearning.Service.UserCourseService;
import com.example.onlinelearning.Service.UserService;
import lombok.RequiredArgsConstructor;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.multipart.MultipartFile;
import java.io.IOException;
import java.security.Principal;
@Controller
@RequiredArgsConstructor
public class UserController {
```

```
    private final UserService userservice;
    private final CourseService courseService;
    private final UserCourseService userCourseService;
    @GetMapping("/login")
    public String LoginForm(){
        return "LoginForm";
    }
    @GetMapping("/register")
    public String RegisterForm(){
        return "RegisterForm";
    }
    @PostMapping("/register")
    public String CreateUser(User user, Model model){
        if(isValidPassword(user.getPassword())){
            if(!userservice.userCreate(user)){
                model.addAttribute("Error","Користувач з таким Email вже
існує.");
                return "RegisterForm";
            }
            return "redirect:/login";
        }
        model.addAttribute("Error","Пароль введено не вірно, пароль повинен
містити як мінімум 6 символів та 2 числа. Приклад: testin12");
        return "RegisterForm";
    }
    private boolean isValidPassword(String pass){
        int simCount=0;
        int digCount=0;
        for(int i=0;i<pass.length();i++){
            char buf = pass.charAt(i);
            if (Character.isDigit(buf)){digCount++;}
            else if(Character.isLetter(buf)) {simCount++;}
        }
        return pass.length() >= 8 && digCount >= 2 && simCount >= 6;
    }
    @GetMapping("/")
    public String HomePage(Principal principal,Model model){
        model.addAttribute("user",userservice.getUserByPrincipal(principal));
        return "WelcomePage";
    }
    @GetMapping("/profil")
    public String ProfilPage(Principal principal,Model model){
        model.addAttribute("user",userservice.getUserByPrincipal(principal));
        return "/profil";
    }
    @PostMapping("/profil/addphoto")
    public String AddPhotoToProfil(@RequestParam("file") MultipartFile file,
Principal principal) throws IOException {
        userservice.saveUserPhotoByPrincipal(principal,file);
        return "redirect:/profil";
    }
    @PostMapping("/profil/renameinfo")
    public String RenameUserInfo(Principal principal,@RequestParam("name")
String name,@RequestParam("surname") String surname,@RequestParam(value =
"familyname", defaultValue = "") String familyname,@RequestParam("nikname")
String nikname){
        User userDB = userservice.getUserByPrincipal(principal);
        userDB.setName(name);
        userDB.setSurname(surname);
        userDB.setFamilyname(familyname);
        userDB.setNikname(nikname);
        userservice.saveUser(userDB);
        return "redirect:/profil";
    }
    @GetMapping("/createcourse")
    public String CreateACoursePage(Principal principal,Model model){
```

```
        model.addAttribute("user",userservice.getUserByPrincipal(principal));
        return "CreateCourse";
    }
    @PostMapping("/createcourse")
    public String CreateACourse(Principal principal,Course course){
        if("".equals(course.getGroupname())){course.setGroupname("Не
вказано");}
        if("".equals(course.getDescription())){course.setDescription("На жаль
цей курс не має опису :(");}
        courseService.saveCourse(course);
        userCourseService.createUserCourse(principal,course.getId());
        Course course1 =
courseService.getCourseByCodeinvite(course.getCodeinvite());
        return "redirect:/course/"+course1.getId();
    }
    @GetMapping("/joincourse")
    public String JoinCoursePage(Principal principal,Model model) {
        model.addAttribute("user",userservice.getUserByPrincipal(principal));
        return "JoinCourse";
    }
    @PostMapping("/joincourse")
    public String JoinToCourse(Model model,@RequestParam("codeinvite") String
codeinvite,Principal principal){
        User user = userservice.getUserByPrincipal(principal);
        Course course = courseService.getCourseByCodeinvite(codeinvite);
        if (course==null){
            model.addAttribute("user",user);
            model.addAttribute("Error","Ви ввели не існуючий код
запрошення.");
            return "JoinCourse";
        }
        boolean ishave = true;
        for(UserCourse userCourse: user.getUserCourses()){
            if(userCourse.getCourse().getId()==course.getId())ishave=false;
        }
        if(!ishave){
            model.addAttribute("user",user);
            model.addAttribute("Error","Курс з таким кодом запрошення у вас
вже є.");
            return "JoinCourse";
        }
        if(!userCourseService.JoinUserCourse(user,course)){
            model.addAttribute("user",user);
            model.addAttribute("Error","Не вдалося вас добавити до курсу.");
            return "JoinCourse";
        }
        return "redirect:/course/"+course.getId();
    }
}
```

Б.1.5 – Модуль ChatController.java

```
package com.example.onlinelearning.Controllers;
import com.example.onlinelearning.Models.*;
import com.example.onlinelearning.Service.CourseService;
import com.example.onlinelearning.Service.UserService;
import lombok.RequiredArgsConstructor;
import org.springframework.messaging.handler.annotation.MessageMapping;
import org.springframework.messaging.handler.annotation.SendTo;
import org.springframework.stereotype.Controller;
import java.security.Principal;
```

```
@Controller
@RequiredArgsConstructor
public class ChatController {
   private final UserService userService;
   private final CourseService courseService;
   @MessageMapping("/course/{id}/chat")
   @SendTo("/topic/course/{id}/chat/greetings")
   public Greeting greeting(Mess message, Principal principal) throws Exception {
      User user = userService.getUserByPrincipal(principal);
      Course course = courseService.getCourseByID(Long.parseLong(message.getCourseid()));
      Mess mess = new Mess();
      mess.setUser(user);
      mess.setMessage(message.getMessage());
      course.getChatmesseges().add(mess);
      courseService.saveCourse(course);
      if(user.getPhoto()==null){return new
Greeting(user.getNikname(),message.getMessage(),message.getTime());}
      return new
Greeting("/images/"+user.getPhoto().getId(),user.getNikname(),message.getMessage(),message.
getTime());
   }
}
```

Б.2 – Модуль Repository

Б.2.1 – Модуль ArticleRepository.java

```
package com.example.onlinelearning.Repository;
import com.example.onlinelearning.Models.Article;
import org.springframework.data.jpa.repository.JpaRepository;
public interface ArticleRepository extends JpaRepository<Article,Long> {}
```

Б.2.2 – Модуль CourseRepository.java

```
package com.example.onlinelearning.Repository;
import com.example.onlinelearning.Models.Course;
import org.springframework.data.jpa.repository.JpaRepository;
public interface CourseRepository extends JpaRepository<Course,Long> {
    Course findByCodeinvite(String codeinvite);
}
```

Б.2.3 – Модуль ImageRepository.java

```
package com.example.onlinelearning.Repository;
import com.example.onlinelearning.Models.Image;
import org.springframework.data.jpa.repository.JpaRepository;
public interface ImageRepository extends JpaRepository<Image,Long> {}
```

Б.2.4 – Модуль QuestionRepository.java

```
package com.example.onlinelearning.Repository;
import com.example.onlinelearning.Models.Question;
import org.springframework.data.jpa.repository.JpaRepository;
public interface QuestionRepository extends JpaRepository<Question,Long> {}
```

Б.2.5 – Модуль TaskRepository.java

```
package com.example.onlinelearning.Repository;
import com.example.onlinelearning.Models.Task;
import org.springframework.data.jpa.repository.JpaRepository;
public interface TaskRepository extends JpaRepository<Task,Long> {}
```

Б.2.6 – Модуль UserCourseRepository.java

```
package com.example.onlinelearning.Repository;
import com.example.onlinelearning.Models.Course;
import com.example.onlinelearning.Models.User;
import com.example.onlinelearning.Models.UserCourse;
import org.springframework.data.jpa.repository.JpaRepository;
import java.util.List;
import java.util.Optional;
public interface UserCourseRepository extends JpaRepository<UserCourse,Long>
{
    Optional<UserCourse> findByUserAndCourse(User user, Course course);
    List<UserCourse> findByCourse(Course course);
}
```

Б.2.7 – Модуль UserQuestionRepository.java

```
package com.example.onlinelearning.Repository;
import com.example.onlinelearning.Models.Question;
import com.example.onlinelearning.Models.User;
import com.example.onlinelearning.Models.UserQuestion;
import org.springframework.data.jpa.repository.JpaRepository;
import java.util.List;
import java.util.Optional;
public interface UserQuestionRepository extends
JpaRepository<UserQuestion,Long> {
    Optional<UserQuestion> findByUserAndQuestion(User user, Question
question);
    List<UserQuestion> findAllByUser(User user);
}
```

Б.2.8 – Модуль UserRepository.java

```
package com.example.onlinelearning.Repository;
import com.example.onlinelearning.Models.User;
import org.springframework.data.jpa.repository.JpaRepository;
```

```
public interface UserRepository extends JpaRepository<User, Long> {
    User findByEmail(String email);
}
```

Б.3 – Модуль Service

Б.3.1 – Модуль CourseService.java

```
package com.example.onlinelearning.Service;
import com.example.onlinelearning.Models.Article;
import com.example.onlinelearning.Models.Course;
import com.example.onlinelearning.Models.Image;
import com.example.onlinelearning.Repository.ArticleRepository;
import com.example.onlinelearning.Repository.CourseRepository;
import com.example.onlinelearning.Repository.ImageRepository;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Service;
import org.springframework.web.multipart.MultipartFile;
import java.io.IOException;
@Service
@Slf4j
@RequiredArgsConstructor
public class CourseService {
    private final CourseRepository courseRepository;
    private final ArticleRepository articleRepository;
    private final ImageRepository imageRepository;
    public void saveCourse(Course course) {
        courseRepository.save(course);
    }
    public Course getCourseByID(Long id) {
        return courseRepository.findById(id).orElse(null);
    }
    public void saveCourseImageByID(Long id, MultipartFile file) throws
IOException {
        Image image;
        Course course = courseRepository.findById(id).orElse(null);
        if(file.getSize()!=0){
            if (course.getImage() != null) {
                Image im = course.getImage();
                course.setImage(null);
                courseRepository.save(course);
                imageRepository.delete(im);
            }
            image = toImageEntity(file);
            course.setImage(image);
            courseRepository.save(course);
        }
    }
    public void saveCourseLogoByID(Long id, MultipartFile file) throws
IOException {
        Image image;
        Course course = courseRepository.findById(id).orElse(null);
        if(file.getSize()!=0){
            if (course.getLogo() != null) {
                Image im = course.getLogo();
                course.setImage(null);
                courseRepository.save(course);
                imageRepository.delete(im);
            }
            image = toImageEntity(file);
            course.setLogo(image);
            courseRepository.save(course);
```

```
        }
    }
    private Image toImageEntity(MultipartFile file) throws IOException {
        Image image = new Image();
        image.setOriginalFileName(file.getOriginalFilename());
        image.setContentType(file.getContentType());
        image.setSize(file.getSize());
        image.setBytes(file.getBytes());
        return image;
    }
    public Course getCourseByCodeinvite(String codeinvite){
        return courseRepository.findByCodeinvite(codeinvite);
    }
    public void deleteCourseByCourse(Course course){
        courseRepository.delete(course);
    }
    public void createArticleByCourse(Course course, Article article){
        course.getArticles().add(article);
        courseRepository.save(course);
    }
    public void deleteArticle(Course course,Long id){
        Article ar = articleRepository.findById(id).orElse(null);
        if(ar!=null){
            course.getArticles().remove(ar);
            articleRepository.delete(ar);
            courseRepository.save(course);
        }
    }
}
```

Б.3.2 – Модуль CustomUserDetailsService.java

```
package com.example.onlinelearning.Service;
import com.example.onlinelearning.Repository.UserRepository;
import lombok.RequiredArgsConstructor;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import
org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Service;
@Service
@RequiredArgsConstructor
public class CustomUserDetailsService implements UserDetailsService {
    private final UserRepository userrepository;
    @Override
    public UserDetails loadUserByUsername(String email) throws
UsernameNotFoundException {
        return userrepository.findByEmail(email);
    }
}
```

Б.3.3 – Модуль QuestionService.java

```
package com.example.onlinelearning.Service;
import com.example.onlinelearning.Models.*;
import com.example.onlinelearning.Repository.QuestionRepository;
import com.example.onlinelearning.Repository.TaskRepository;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
```

```
import org.springframework.stereotype.Service;
@Service
@Slf4j
@RequiredArgsConstructor
public class QuestionService {
    private final QuestionRepository questionRepository;
    private final TaskRepository taskRepository;
    public Question getQuestionByID(Long id){
        return questionRepository.findById(id).orElse(null);
    }
    public void saveQuestion(Question question) {
        questionRepository.save(question);
    }
    public void deleteTaskById(Question question,Long id){
        Task task = taskRepository.findById(id).orElse(null);
        if(task!=null){
            question.getTasks().remove(task);
            taskRepository.delete(task);
            questionRepository.save(question);
        }
    }
    public void deleteQuestion(Question question){
        questionRepository.delete(question);
    }
}
```

Б.3.4 – Модуль UserCourseService.java

```
package com.example.onlinelearning.Service;
import com.example.onlinelearning.Models.Course;
import com.example.onlinelearning.Models.User;
import com.example.onlinelearning.Models.UserCourse;
import com.example.onlinelearning.Repository.UserCourseRepository;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Service;
import java.security.Principal;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
@Service
@Slf4j
@RequiredArgsConstructor
public class UserCourseService {
    private final UserCourseRepository userCourseRepository;
    private final UserService userService;
    private final CourseService courseService;
    public boolean createUserCourse(Principal principal,Long id){
        UserCourse userCourse = new UserCourse();
        User user = userService.getUserByPrincipal(principal);
        Course course = courseService.getCourseByID(id);
        userCourse.setRole("ROLE_CREATER");
        userCourse.setCourse(course);
        userCourse.setUser(user);
        userCourseRepository.save(userCourse);
        return true;
    }
    public boolean JoinUserCourse(User user,Course course){
        UserCourse userCourse = new UserCourse();
        userCourse.setRole("ROLE_USER");
        userCourse.setCourse(course);
        userCourse.setUser(user);
        userCourseRepository.save(userCourse);
```

```
        return true;
    }
    public List<UserCourse> getOnlyUsersByCourse(Course course){
        List<UserCourse> users = new ArrayList<>();
        for (UserCourse userCourse:course.getUserCourses()){

if("ROLE_USER".equals(userCourse.getRole()))users.add(userCourse);
        }
        return users;
    }
    public List<UserCourse> getOnlyAdminsByCourse(Course course){
        List<UserCourse> users = new ArrayList<>();
        for (UserCourse userCourse:course.getUserCourses()){

if("ROLE_ADMIN".equals(userCourse.getRole())||"ROLE_CREATER".equals(userCours
e.getRole()))users.add(userCourse);
        }
        return users;
    }
    public String getUserRole(Course course,User user){
        String role= "";
        for (UserCourse userCourse:course.getUserCourses()){
            if(userCourse.getUser().equals(user)) role =
userCourse.getRole();
        }
        return role;
    }
    public void changeUserRole(Course course,UserCourse userCourse){
        String role = getUserRole(course,userCourse.getUser());
        if("ROLE_USER".equals(role)){
            userCourse.setRole("ROLE_ADMIN");
            userCourseRepository.save(userCourse);
        } else if ("ROLE_ADMIN".equals(role)) {
            userCourse.setRole("ROLE_USER");
            userCourseRepository.save(userCourse);
        }
    }
    public void deleteCourseByCourse(Course course){
        for (UserCourse userCourse :
userCourseRepository.findByCourse(course)) {
            userCourseRepository.delete(userCourse);
        }
        courseService.deleteCourseByCourse(course);
    }
    public void deleteUserCourse(User user,Course course){
        UserCourse userCourse =
userCourseRepository.findByUserAndCourse(user,course).orElse(null);
        if(userCourse!=null) userCourseRepository.delete(userCourse);
    }
    public UserCourse getUserCourseByID(Long id){
        return userCourseRepository.findById(id).orElse(null);
    }
}
```

Б.3.5 – Модуль UserQuestionService.java

```
package com.example.onlinelearning.Service;
import com.example.onlinelearning.Models.*;
import com.example.onlinelearning.Repository.UserQuestionRepository;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Service;
import org.springframework.web.multipart.MultipartFile;
```

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
@Service
@Slf4j
@RequiredArgsConstructor
public class UserQuesrionService {
    private final UserQuestionRepository userQuestionRepository;
    private final QuestionService questionService;
    public void createUserQuestion(User user, Question question, List<Answer>
answers) throws IOException {
        UserQuestion userQuestion =
userQuestionRepository.findByUserAndQuestion(user,question).orElse(null);
        if(userQuestion==null){
            UserQuestion userQ = new UserQuestion();
            userQ.setQuestion(question);
            userQ.setUser(user);
            for(Answer ans : answers){
                if("images".equals(ans.getType())){
                    List<Image> images = new ArrayList<>();
                    for(MultipartFile file : ans.getFiles()){
                        if(file.getSize()!=0) {
                            images.add(toImageEntity(file));
                        }
                    }
                    answers.get(answers.indexOf(ans)).setImages(images);
                }
            }
            userQ.setAnswers(answers);
            userQ.setStatus("wait");
            userQuestionRepository.save(userQ);
        }else {
            userQuestionRepository.delete(userQuestion);
            UserQuestion userQ = new UserQuestion();
            userQ.setQuestion(question);
            userQ.setUser(user);
            for(Answer ans : answers){
                if("images".equals(ans.getType())){
                    List<Image> images = new ArrayList<>();
                    for(MultipartFile file : ans.getFiles()){
                        if(file.getSize()!=0) {
                            images.add(toImageEntity(file));
                        }
                    }
                    answers.get(answers.indexOf(ans)).setImages(images);
                }
            }
            userQ.setRating(userQuestion.getRating());
            userQ.setAnswers(answers);
            userQ.setStatus("wait");
            userQuestionRepository.save(userQ);
        }
    }
    private Image toImageEntity(MultipartFile file) throws IOException {
        Image image = new Image();
        image.setOriginalFileName(file.getOriginalFilename());
        image.setContentType(file.getContentType());
        image.setSize(file.getSize());
        image.setBytes(file.getBytes());
        return image;
    }
    public UserQuestion getUserQuestionByUserAndQuestion(User user,Question
question){
        return
userQuestionRepository.findByUserAndQuestion(user,question).orElse(null);
    }
```

```
    public void save(UserQuestion userQuestion){
        userQuestionRepository.save(userQuestion);
    }
    public void deleteQuestionByQuestion(Question question){
        for (UserQuestion userQuestion : question.getUserQuestions()) {
            userQuestionRepository.delete(userQuestion);
        }
        questionService.deleteQuestion(question);
    }
    public void reinfoQuestion(Question question){
        for (UserQuestion userQuestion : question.getUserQuestions()) {
            userQuestion.setRating("-");
            userQuestion.setStatus("none");
        }
        questionService.saveQuestion(question);
    }
    public List<UserQuestion> getAllUserQuestionByUser(User user){
        return userQuestionRepository.findAllByUser(user);
    }
    public void deleteUserQuestionById(Long id){
        UserQuestion userQuestion =
userQuestionRepository.findById(id).orElse(null);
        if(userQuestion!=null)userQuestionRepository.delete(userQuestion);
    }
}
```

Б.3.6 – Модуль UserService.java

```
package com.example.onlinelearning.Service;
import com.example.onlinelearning.Models.Image;
import com.example.onlinelearning.Models.User;
import com.example.onlinelearning.Repository.ImageRepository;
import com.example.onlinelearning.Repository.UserRepository;
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Service;
import org.springframework.web.multipart.MultipartFile;
import java.io.IOException;
import java.security.Principal;
@Service
@Slf4j
@RequiredArgsConstructor
public class UserService {
    private final ImageRepository imageRepository;
    private final UserRepository userrepository;
    private final PasswordEncoder passwordEncoder;
    public void saveUser(User user){
        userrepository.save(user);
    }
    public boolean userCreate(User user){
        String email = user.getEmail();
        if(userrepository.findByEmail(email)!=null) return false;
        user.setNikname(user.getName());
        user.setPassword(passwordEncoder.encode(user.getPassword()));
        userrepository.save(user);
        return true;
    }
    public void saveUserPhotoByPrincipal(Principal principal, MultipartFile
file) throws IOException {
            Image photo;
            User user = getUserByPrincipal(principal);
            if(file.getSize()!=0){
```

```
                if (user.getPhoto() != null) {
                    Image im = user.getPhoto();
                    user.setPhoto(null);
                    userrepository.save(user);
                    imageRepository.delete(im);
                }
                photo = toImageEntity(file);
                user.setPhoto(photo);
                userrepository.save(user);
            }
    }
    public User getUserByID(Long id ){return
userrepository.findById(id).orElse(null);}
    public User getUserByPrincipal(Principal principal){
        return userrepository.findByEmail(principal.getName());
    }
    private Image toImageEntity(MultipartFile file) throws IOException {
        Image image = new Image();
        image.setOriginalFileName(file.getOriginalFilename());
        image.setContentType(file.getContentType());
        image.setSize(file.getSize());
        image.setBytes(file.getBytes());
        return image;
    }
}
```

Б.4 – Модуль templates

Б.4.1 – Модуль AnswerQuestion.ftlh

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="/css/Style.css" rel="stylesheet" type="text/css">
    <title>Сторінка вирішення завдання</title>
</head>
<#include "header.ftlh">
<body>
    <div class="useranswer">
        <div class="header">
            <h1>Вирішіть всі завдання та натисніть зберегти</h1>
        </div>
            <form style="display:flex;align-items:center;justify-
content:center;flex-direction: column;" enctype="multipart/form-data"
action="/course/${course.id}/question/${question.id}/createanswer"
method="post">
                <#list question.getTasks() as task>
                <div class="contentform">
                <h1 style="margin:20px;">${task.quest}</h1>
                <input type="hidden" name="answers[${task_index}].quest"
value="${task.quest}">
                <input type="hidden" name="answers[${task_index}].type"
value="${task.type}">
                <#if task.type == "test">
                    <div style="display:flex;justify-content:center;flex-
direction: column; text-align: left;">
                    <#list task.getChecktasks() as checktask>
                        <div style="display:flex;align-items:center;"
class="taskcontent">
                        <input type="hidden"
name="answers[${task_index}].checktasks[${checktask_index}].answer"
value="${checktask.answer}">
                        <input type="checkbox"
name="answers[${task_index}].checktasks[${checktask_index}].istrue"
```

```
><p>${checktask.answer}</p>
                        </div>
                    </#list>
                    </div>
                <#elseif task.type == "images">
                    <div style="display:flex;justify-content:center;flex-
direction: column; text-align: left;margin-bottom:25px;">
                    <p>Виберіть декілька фото/картинок якщо потрібно</p>
                    <input type="file" name="answers[${task_index}].files"
multiple>
                    </div>
                <#elseif task.type == "textarea">
                    <#if userquestion??>
                    <#list userquestion.getAnswers() as useranswer>
                        <#if useranswer_index == task_index>
                        <textarea name="answers[${task_index}].textarea"
maxlength="2000">${useranswer.textarea}</textarea>
                        </#if>
                    </#list>
                    <#else>
                        <textarea name="answers[${task_index}].textarea"
maxlength="2000" placeholder="Ведіть вашу відповеть на запитання"></textarea>
                    </#if>
                </#if>
                </div>
                </#list>
                <input type="hidden" name="_csrf" value="${_csrf.token}">
                <button type="submit">Зберегти відповідь</button>
            </form>
    </div>
</body>
</html>
```

Б.4.2 – Модуль ChatPage.ftlh

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="/css/Style.css" rel="stylesheet" type="text/css">
    <title>Чат</title>
</head>
<#include "header.ftlh">
<body>
    <div class="chat">
        <div class="header">
            <h1>Груповий чат</h1>
        </div>
        <div class="messagecontent">
            <div class="messages" id="messages">
                <#list messages as messeg>
                    <div class="mess">
                        <div style="display:flex; align-items:center; ">
                            <#if messeg.user.getPhoto()??>
                            <img style="width:55px;height:55px;border-
radius:50%;" src="/images/${messeg.user.getPhoto().id}" alt="Фото
користувача">
                            <#else>
                            <img style="width:55px;height:55px;border-
radius:50%;" src="https://raw.githubusercontent.com/itchief/modx-
solutions/main/Login/04-upload-photo/01-default.svg" alt="Фото користувача">
                            </#if>
                            <h1 style="margin-
```

```
left:25px;">${messeg.user.nikname}</h1>
                        </div>
                        <p>${messeg.message}</p>
                        <p style="font-size:15px;text-shadow: none;font-
width: bold;">${messeg.time}</p>
                    </div>
                </#list>
            </div>
            <div class="createmessage">
                <input type="text" maxlength="500" id="createMess"
placeholder="Введіть ваше повідомлення..."/>
                <button id="sendMess" onclick="sendMess();">>>>></button>
            </div>
        </div>
    </div>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/sockjs-
client/1.5.0/sockjs.min.js"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/stomp.js/2.3.3/stomp.min.js"></sc
ript>
    <script type="text/javascript">
        var stompClient = null;
        window.onload = function() {connect();};
        window.addEventListener('beforeunload', function()
{stompClient.disconnect();});
         function disconnect() {
            stompClient.disconnect();
            console.log("Disconnected");
        }
        function connect() {
            var socket = new SockJS('/course/${course.id}/chat');
            stompClient = Stomp.over(socket);
            stompClient.connect({}, function(frame) {
                console.log('Connected: ' + frame);

stompClient.subscribe('/topic/course/${course.id}/chat/greetings',
function(greeting){
                    showGreeting(JSON.parse(greeting.body));
                });
            });
        }
        function sendMess() {
            var message = document.getElementById('createMess').value;
            if(message != ""){
                stompClient.send("/app/course/${course.id}/chat", {},
JSON.stringify({ 'message': message, 'courseid':${course.id} }));
                document.getElementById('createMess').value = "";
            }
        }
        function showGreeting(greet) {
            const newDiv = document.createElement('div');
            newDiv.classList.add('mess');
            newDiv.innerHTML =`
                    <div style="display:flex; align-items:center;">
                    <img style="width:55px;height:55px;border-radius:50%;"
src="`+greet.srcimg+`" alt="Фото користувача">
                    <h1 style="margin-left:25px;">`+greet.nikname+`</h1>
                    </div>
                    <p>`+greet.content+`</p>
                    <p style="font-size:15px;text-shadow: none;font-width:
bold;">`+greet.time+`</p>`;
            var messages = document.getElementById('messages');
            messages.appendChild(newDiv);
        }
    </script>
```

```
</body>
</html>
```

Б.4.3 – Модуль CreateCourse.ftlh

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="/css/Style.css" rel="stylesheet" type="text/css">
    <title>Створення курсу</title>
</head>
<#include "header.ftlh">
<body>
    <div class="Create-main">
        <div class="header"><h1>Створити курс</h1></div>
        <form name="createcourse" action="/createcourse" method="post">
            <div class="main">
                <p>Введіть назву вашого курсу</p>
                <input type="text" maxlength="200" name="title"
placeholder="Назва курсу" required>
                <p>Введіть назву групи/класу (не обов'язково)</p>
                <input type="text" maxlength="50" name="groupname"
placeholder="Назва класу чи групи" >
            </div>
            <div class="description">
                <p>Введіть опис вашого курсу (не обов'язково)</p>
                <textarea name="description" maxlength="800"
placeholder="Опис курсу"></textarea>
            </div>
            <div class="btn">
                <input type="hidden" name="_csrf" value="${_csrf.token}">
                <button type="submit">Створити</button>
            </div>
        </form>
    </div>
</body>
</html>
```

Б.4.4 – Модуль CreateQuestion.ftlh

```
<!DOCTYPE html>
<html>
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
   <link href="/css/Style.css" rel="stylesheet" type="text/css">
   <title>Створення запитань</title>
</head>
<#include "header.ftlh">
<body>
   <div class="createquestion">
      <div class="header">
         <h1>Створення та редагування запитань</h1>
      </div>
      <div class="tasks">
         <#list question.getTasks() as task>
```

```
        <div>
          <h1 style="color:black;text-shadow: 1px 1px 1px white;word-wrap: break-wrap; text-
indent: 30px; font-size:23px;">${task.quest}</h1>
          <form class="btnfordeletearticle"
action="/course/${course.id}/question/${question.id}/${task.id}/delete" method="post">
            <input type="hidden" name="_csrf" value="${_csrf.token}">
            <button style="margin-left:0;" type="submit">Видалити запитання</button>
          </form>
          <#if task.type == "test">
            <div style="display:flex;justify-content:center;flex-direction: column; text-align:
left;">
            <#list task.getChecktasks() as ans>
              <div style="display:flex;align-items:center;" class="taskcontent">
              <#if ans.istrue??>
                <input type="checkbox" checked disabled><p style="font-
size:16px">${ans.answer}</p>
              <#else>
                <input type="checkbox" disabled><p style="font-
size:16px">${ans.answer}</p>
              </#if>
              </div>
            </#list>
            </div>
          </#if>
        </div>
        </#list>
      </div>
      <form class="formcontent" method="post"
action="/course/${course.id}/question/${question.id}">
      </form>
      <div class="cretetask">
        <p>Якщо ви хочете створити запитання то виберіть тип запитання та натисніть на
кнопку для створити</p>
        <div>
          <select id="my-select">
            <option value="">---  Оберіть формат запитання  ---</option>
            <option value="type1">Формат тесту</option>
            <option value="type2">Формат картинок</option>
            <option value="type3">Формат тексту</option>
          </select>
          <button class="btnforcreateform" id="btnforcreateform" type="button">+</button>
        </div>
      </div>
    </div>
    <script type="text/javascript">
    const btnform = document.getElementById('btnforcreateform');
    const select = document.getElementById('my-select');
    const formcontent = document.querySelector('.formcontent');
    let questid = 2;
    btnform.addEventListener("click", function() {
      let selectedValue = select.value;
      if (selectedValue == 'type1') {
        formcontent.innerHTML = `<p>Уведіть ваше запитання</p>
        <input type="text" style="width:650px" name="quest" placeholder="Уведіть
```

```
запитання"><br/>
        <input type="hidden" name="type" value="test">
        <div class="content">
          <p style="font-size: 13px">Якщо цей варіант повинен бути правильним то
поставте галочку </p>
          <input type="text" maxlength="200" name="checktasks[0].answer"
placeholder="Напишіть варіант відповіді">
          <input type="checkbox" name="checktasks[0].istrue">
          <p style="font-size: 13px">Якщо цей варіант повинен бути правильним то
поставте галочку </p>
          <input type="text" maxlength="200" name="checktasks[1].answer"
placeholder="Напишіть варіант відповіді">
          <input type="checkbox" name="checktasks[1].istrue">
        </div>
        <input type="hidden" name="_csrf" value="${_csrf.token}">
        <button id="addanswer" type="button">Добавити відповідь</button><br/>
        <button id="savequest" type="submit">Зберегти запитання</button>`;
        const addanswerebutton = document.getElementById("addanswer");
        const divcontentforaddanswer = document.querySelector('.content');
        addanswerebutton.addEventListener("click", function() {
          const newDiv = document.createElement('div');
          newDiv.innerHTML = `
          <p style="font-size: 13px">Якщо цей варіант повинен бути правильним то
поставте галочку </p>
          <input type="text" maxlength="200" name="checktasks[`+questid+`].answer"
placeholder="Напишіть варіант відповіді">
          <input type="checkbox" name="checktasks[`+questid+`].istrue">
          `;
          divcontentforaddanswer.appendChild(newDiv);
          questid ++;
        });
      } else if (selectedValue == 'type2') {
        formcontent.innerHTML = `<p>Уведіть ваше запитання</p>
        <input type="text" maxlength="200" style="width:650px" name="quest"
placeholder="Уведіть запитання"><br/>
        <input type="hidden" name="type" value="images">
        <input type="hidden" name="_csrf" value="${_csrf.token}">
        <button id="savequest" type="submit">Зберегти запитання</button>`;
      } else if (selectedValue == 'type3') {
        formcontent.innerHTML = `<p>Уведіть ваше запитання</p>
        <input type="text" maxlength="200" style="width:650px" name="quest"
placeholder="Уведіть запитання"><br/>
        <input type="hidden" name="type" value="textarea">
        <input type="hidden" name="_csrf" value="${_csrf.token}">
        <button id="savequest" type="submit">Зберегти запитання</button>`;
      }else{
        alert("Будьласка оберіть тип вашого запитання!");
      }

    });
  </script>
</body>
</html>
```

Б.4.5 – Модуль Error.ftlh

```
<div class="error">
    <div class="header">
        <h1>Помилка!!!</h1>
        <button id="closeerror" onclick="closeerror();">X</button>
    </div>
    <div class="content">
        <p>${Error}</p>
    </div>
</div>
<script type="text/javascript">
    function closeerror() {
    document.querySelector('.error').remove();
    }
</script>
```

Б.4.6 – Модуль header.ftlh

```
<header class="header">
    <div class="header-course">
        <#if course??>
        <#if course.getLogo()??>
        <img src="/images/${course.logo.id}">
        <#else>
        <img src="https://cdn-icons-
png.flaticon.com/512/477/477804.png?w=996&t=st=1681587991~exp=1681588591~hmac
=575ad9dbdc8dc3d3f1dd41d5528c51f198658832ca65a3df9c6bf836030673c5" >
        </#if>
        </#if>
        <button class="header-dropbtn">
            <h1>
                <#if course??>
                    ${course.title}
                <#else>
                    Оберіть курс
                </#if>
            </h1>
            <ul class="header-dropdawn">
                <li><a href="/createcourse">Створити свій курс</a></li>
                <li><a href="/joincourse">Увійти до існуючого курсу</a></li>
                <#list user.getUserCourses() as usercourse>
                <li><a
href="/course/${usercourse.course.id}">${usercourse.course.title}</a></li>
                </#list>
            </ul>
        </button>
    </div>
    <div class="header-list">
        <#if course??>
        <div>
            <a href="/course/${course.id}/question"><h1>Завдання</h1></a>
        </div>
        <div>
            <a href="/course/${course.id}"><h1>Головна сторінка
курсу</h1></a>
        </div>
        <div>
            <a href="/course/${course.id}/users"><h1>Користувачі</h1></a>
        </div>
```

```
        <div>
            <a href="/course/${course.id}/chat"><h1>Чат</h1></a>
        </div>
        </#if>
    </div>
    <div class="header-profil">
        <a href="/profil">
            <#if user.getPhoto()??>
            <img src="/images/${user.getPhoto().id}">
            <#else>
            <img src="https://raw.githubusercontent.com/itchief/modx-
solutions/main/Login/04-upload-photo/01-default.svg" alt="Фото користувача">
            </#if>
        </a>
    </div>
    <script type="text/javascript">
    const btn = document.querySelector('.header-dropbtn');
    const list = document.querySelector('.header-dropdawn');
    btn.addEventListener('mouseover', () => {
      list.style.display = 'block';
    });
    btn.addEventListener('mouseout', () => {
      list.style.display = 'none';
    });
    </script>
</header>
```

Б.4.7 – Модуль headerwo.ftlh

```
<header class="header">
  <div class="header-course">
    <button  class="header-dropbtn" />
      <h1>
        Оберіть курс
      </h1>
      <ul class="header-dropdawn">
        <li><a href="/createcourse">Створити свій курс</a></li>
        <li><a href="/joincourse">Увійти до існуючого курсу</a></li>
        <#list user.getUserCourses() as usercourse>
          <li><a href="/course/${usercourse.course.id}">${usercourse.course.title}</a></li>
        </#list>
      </ul>
    </button>
  </div>
  <div class="header-list" style="pointer-events: none; opacity: 0;">
    <div>
      <a href="Task.html"><h1>Завдання</h1></a>
    </div>
    <div>
      <a href="MainPage.html"><h1>Головна сторінка курсу</h1></a>
    </div>
    <div>
      <a href="UsersPage.html"><h1>Користувачі</h1></a>
    </div>
    <div>
      <a href="Chat.html"><h1>Чат</h1></a>
    </div>
```

```
  </div>
  <div class="header-profil">
    <a href="/profil">
      <#if user.getPhoto()??>
      <img src="/images/${user.getPhoto().id}">
      <#else>
      <img src="https://raw.githubusercontent.com/itchief/modx-solutions/main/Login/04-
upload-photo/01-default.svg" alt="Фото користувача">
      </#if>
    </a>
  </div>
  <script type="text/javascript">
  const btn = document.querySelector('.header-dropbtn');
  const list = document.querySelector('.header-dropdawn');
  btn.addEventListener('mouseover', () => {
   list.style.display = 'block';
  });
  btn.addEventListener('mouseout', () => {
   list.style.display = 'none';
  });
  </script>
</header>
```

Б.4.8 – Модуль JoinCourse.ftlh

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="/css/Style.css" rel="stylesheet" type="text/css">
    <title>Увіти до курсу</title>
</head>
<#include "headerwo.ftlh">
<body>
<#if Error??>
<#include "Error.ftlh">
</#if>
    <div class="JoinCourse">
        <form name="joincourseform" action="/joincourse" method="post">
            <p>Введіть код курсу</p>
            <input name="codeinvite" maxlength="50" placeholder="Код курсу">
            <input type="hidden" name="_csrf" value="${_csrf.token}">
            <button type="submit">Увійти</button>
        </form>
    </div>
</body>
</html>
```

Б.4.9 – Модуль LoginForm.ftlh

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

```
    <link href="/css/Style.css" rel="stylesheet" type="text/css">
    <title>Login</title>
</head>
<body>
<#if Error??>
<#include "Error.ftlh">
</#if>
<div class="Login">
    <div class="Login-header">
        <a href="/login">Login</a><p>|</p><a href="/register">Register</a>
    </div>
    <div class="Login-body">
        <form name="LoginForm" action="/login" method="post" >
            <p>Email</p>
            <input type="email" name="username" maxlength="50"
placeholder="Уведіть ваш Email" >
            <p>Пароль</p>
            <input type="password" name="password" maxlength="50"
placeholder="Уведіть ваш пароль">
            <input type="hidden" name="_csrf" value="${_csrf.token}">
            <button type="submit">Увійти</button>
        </form>
    </div>
</div>
</body>
</html>
```

Б.4.10 – Модуль MainCoursePage.ftlh

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="/css/Style.css" rel="stylesheet" type="text/css">
    <title>Головна сторінка курсу</title>
</head>
<#include "header.ftlh">
<body>
   <div class="maincourseinfo">
       <h1>${course.title}</h1>
       <#if role == "ROLE_ADMIN"||role=="ROLE_CREATER">
       <p>Код запрошення: ${course.codeinvite}</p>
       </#if>
   </div>
  <div class="image-course">
      <#if course.getImage()??>
      <img src="/images/${course.image.id}">
      <#else>
      <img src="https://img.freepik.com/free-vector/homeschooling-design-
concept-with-doodle-characters-of-children-and-adult-teachers-tutoring-with-
desks-and-placards-vector-illustration_1284-
76635.jpg?w=1380&t=st=1681590710~exp=1681591310~hmac=d7e7da9893e2b079899647ee
025ba6e3be3fd39fd9240b0edd072594adc3d7d2">
      </#if>
        <#if course.description?? && course.groupname??>
            <div class="description">Група курсу:
${course.groupname}<br/>${course.description}</div>
        <#elseif course.description??>
            <div class="description">${course.description}</div>
        <#elseif course.groupname??>
        <div class="description">Група курсу: ${course.groupname}<br/>А також
тут повинен бути опис курсу, але на жаль курс не має опису</div>
        <#else>
```

```
            <div class="description">Тут повинен бути опис курсу, але на жаль
курс не має опису</div>
        </#if>
  </div>
  <br/>
  <#if role == "ROLE_ADMIN"||role=="ROLE_CREATER">
      <div class="changeimage">
          <form class="addimage" name="changeimage"
action="/course/${course.id}/addimage" method="post"
enctype="multipart/form-data">
              <input type="file" name="file" accept="uk" aria-label="Оберіть
файл">
              <input type="hidden" name="_csrf" value="${_csrf.token}">
              <button type="submit">Змінити головну картинку?</button>
          </form>
          <form class="addimage" name="changeimage"
action="/course/${course.id}/addlogo" method="post"  enctype="multipart/form-
data">
              <input type="file" name="file" accept="uk" aria-label="Оберіть
файл">
              <input type="hidden" name="_csrf" value="${_csrf.token}">
              <button type="submit">Змінити картинку логотип?</button>
          </form>
      </div>
  </#if>

  <div class="main-course">
    <div class="main-header">
        <h1>Головна сторінка курсу</h1>
        <a class="evaluations" href="/course/${course.id}/ratings">Оцінки</a>
        <#if role=="ROLE_CREATER">
        <form class="btnfordeletecourse" action="/course/${course.id}/delete"
method="post">
            <input type="hidden" name="_csrf" value="${_csrf.token}">
            <button type="submit">Видалити курс</button>
        </form>
        <#elseif role=="ROLE_USER"||role == "ROLE_ADMIN">
        <form class="btnfordeletecourse"
action="/course/${course.id}/conectdelete" method="post">
            <input type="hidden" name="_csrf" value="${_csrf.token}">
            <button type="submit">Покинути курс</button>
        </form>
        </#if>
    </div>
    <#list course.getArticles() as article>
        <div class="article">
            <h1>${article.title}</h1>
            <div class="article-description">
                <p>${article.description}</p>
                <p style="font-size:15px;text-shadow: none;font-width:
bold;">${article.time}</p>
            </div>
            <#if role == "ROLE_ADMIN"||role=="ROLE_CREATER">
            <form class="btnfordeletearticle"
action="/course/${course.id}/${article.id}/deletearticle" method="post">
                <input type="hidden" name="_csrf" value="${_csrf.token}">
                <button type="submit">Видалити оголошення</button>
            </form>
            </#if>
        </div>
    </#list>
    <div class="contentform" id="contentform"></div>
    <#if role == "ROLE_ADMIN"||role=="ROLE_CREATER">
    <div class="createarticle">
        <p>Якщо ви хочите створити оголошення натисність на кнопку</p>
        <button class="btnforcreateform" id="btnforcreateform">+</button>
```

```
    </div>
    </#if>
  </div>
  <script type="text/javascript">
    const createarticlebutton = document.querySelector('#btnforcreateform');
    const divcontentforform = document.querySelector('#contentform');

    createarticlebutton.addEventListener('click', () => {
      const formHtml = `
        <form name="creatarticle" action="/course/${course.id}/createarticle"
method="post">
            <div class="description">
                <label for="title">Напишіть заголовок оголошення</label>
                <input maxlength="200" type="text" id="title" name="title"
placeholder="Заголовок оголошення" required>
                <label for="description">Напишіть зміст оголошення</label>
                <textarea maxlength="2000" id="description"
name="description" placeholder="Зміст оголошення" required></textarea>
            </div>
            <div class="btn">
                <input type="hidden" name="_csrf" value="${_csrf.token}">
                <button type="submit">Створити</button>
            </div>
        </form>
      `;
      divcontentforform.innerHTML = formHtml;
    });
  </script>
</body>
</html>
```

Б.4.11 – Модуль Profil.ftlh

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="/css/Style.css" rel="stylesheet" type="text/css">
    <title>Профіль користувача</title>
</head>
<#include "headerwo.ftlh">
<body>
    <div class="profil-main">
        <div class="header">
            <h1>Ваш профіль</h1>
            <form class="btnforlogout" action="/logout" method="post">
                <input type="hidden" name="_csrf" value="${_csrf.token}">
                <button type="submit">Вийти з акаунту</button>
            </form>
        </div>
        <div class="profil-info">
            <div class="user-photo">
                <#if user.getPhoto()??>
                <img src="/images/${user.getPhoto().id}">
                <#else>
                <img src="https://raw.githubusercontent.com/itchief/modx-
solutions/main/Login/04-upload-photo/01-default.svg" alt="Фото користувача">
                </#if>
                <form class="addimage" name="changephoto"
action="/profil/addphoto" method="post"  enctype="multipart/form-data">
                    <input type="file" name="file" >
                    <input type="hidden" name="_csrf" value="${_csrf.token}">
                    <button type="submit">Змінити картинку?</button>
```

```
                </form>
            </div>
            <div class="user-info">
                <p>Ваше ім'я: ${user.name}</p>
                <p>Ваше прізвище: ${user.surname}</p>
                <p>Ваше ім'я по бітькові: ${user.familyname}</p>
                <p>Ваш псевдонім: ${user.nikname}</p>
            </div>
            <button class="rename">Змінити ваші дані?</button>
        </div>
        <form class="renameinfo" name="renameinfo"
action="/profil/renameinfo" method="post">
            <input type="text" maxlength="50" name="name" maxlength="50"
value="${user.name}">
            <input type="text" maxlength="50" name="surname" maxlength="50"
value="${user.surname}" >
            <input type="text" maxlength="50" name="familyname"
maxlength="50" value="${user.familyname}">
            <input type="text" maxlength="25" name="nikname" maxlength="50"
value="${user.nikname}">
            <input type="hidden" name="_csrf" value="${_csrf.token}">
            <button type="submit">Зберегти</button>
        </form>
    </div>
    <script type="text/javascript">
    const btnrename = document.querySelector('.rename');
    const formrename = document.querySelector('.renameinfo');
    btnrename.addEventListener('click', function() {
          formrename.style.display = 'flex';
    });
    </script>
</body>
</html>
```

Б.4.12 – Модуль Questions.ftlh

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <link href="/css/Style.css" rel="stylesheet" type="text/css">
  <title>Завдання</title>
</head>
<#include "header.ftlh">
<body>
  <div class="task-course">
    <div class="task-header">
      <#if role == "ROLE_ADMIN"||role=="ROLE_CREATER">
      <a href="/course/${course.id}/usersanswer"
class="btnforcheckanswer">Переглянути відповіді</a>
      </#if>
      <h1>Завдання для виконання</h1>
      <a class="evaluations" href="/course/${course.id}/ratings">Оцінки</a>
    </div>
    <div class="tasks">
      <#list course.getQuestions() as question>
      <div class="task">
        <div class="title">
          <p>${question.title}<a
href="/course/${course.id}/question/${question.id}"> Подробиці</a></p>
        </div>
        <#if role=="ROLE_CREATER"||role=="ROLE_ADMIN">
        <form style="display:flex;align-items:center;justify-content:center;"
```

```
class="btnfordeletearticle"
action="/course/${course.id}/question/${question.id}/delete" method="post">
          <input type="hidden" name="_csrf" value="${_csrf.token}">
          <button style="margin:0;" type="submit">Видалити завдання</button>
        </form>
        <#else>
        <div class="checkblock">
          <#if userQuestions?has_content>
            <#assign found = false>
              <#list userQuestions as userquestion>
                <#if userquestion.question.id == question.id>
                <#assign found = true>
                  <#if userquestion.status =="wait">
                    <p style="color:grey">Перевіряється</p>
                    <#elseif userquestion.status =="done">
                    <p style="color:green">Перевірено</p>
                    <#else>
                    <p>Не виконано</p>
                  </#if>
                <#break>
                </#if>
              </#list>
            <#if !found>
            <p>Не виконано</p>
            </#if>
            <#else>
            <p>Не виконано</p>
          </#if>
        </div>
        </#if>
      </div>
      </#list>
    </div>
    <#if role=="ROLE_CREATER"||role=="ROLE_ADMIN">
    <div class="createquestion">
      <form name="creatarticle" action="/course/${course.id}/question"
method="post">
      <p style="text-indent: 0">Напишіть заголовок завдання</p>
      <input maxlength="200" type="text" name="title" placeholder="Заголовок
завдання" required>
      <input type="hidden" name="_csrf" value="${_csrf.token}">
      <button type="submit">Створити</button>
      </form>
    </div>
    </#if>
  </div>
</body>
</html>
```

Б.4.13 – Модуль RatingsPage.ftlh

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="/css/Style.css" rel="stylesheet" type="text/css">
    <title>Оцінки</title>
</head>
<#include "header.ftlh">
<body>
    <div style=" margin: 200px auto 50px auto;" class="ratings">
        <table class="table">
            <thead>
```

```
            <tr>
                <th>Учень/Завдання</th>
                <#list questions as question>
                <th title="${question.title}">${question.title}</th>
                </#list>
            </tr>
            </thead>
            <tbody>
            <#list users as us>
            <#if us.role == "ROLE_USER">
                <tr>
                    <td
title="${us.user.name}_${us.user.surname}_${us.user.familyname}">${us.user.na
me} ${us.user.surname} ${us.user.familyname}</td>
                    <#if us.user.getUserQuestions()?has_content>
                        <#list questions as question>
                            <#assign found = false>
                            <#list us.user.getUserQuestions() as usquest>
                                <#if question.id == usquest.question.id>
                                <#assign found = true>
                                <td>${usquest.rating}</td>
                                <#break>
                                </#if>
                            </#list>
                            <#if !found>
                            <td>-</td>
                            </#if>
                        </#list>
                    <#else>
                        <#list questions as question>
                                <td>-</td>
                        </#list>
                    </#if>
                </tr>
            </#if>
            </#list>
            </tbody>
    </table>
    </div>
</body>
</html>
```

Б.4.14 – Модуль RegisterForm.ftlh

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <link href="/css/Style.css" rel="stylesheet" type="text/css">
  <title>Реєстрація</title>
</head>
<body>
<#if Error??>
<#include "Error.ftlh">
</#if>
<div class="Login">
  <div class="Login-header">
    <a href="/login">Login</a><p>|</p><a href="/register">Register</a>
  </div>
  <div class="Login-body">
    <form name="RegisterForm" action="/register" method="post" >
      <p>Email</p>
      <input type="email" name="email" maxlength="50" placeholder="Уведіть
```

```
вашу електронну пошту" required>
      <p>Ім'я</p>
      <input type="text" name="name" maxlength="50" placeholder="Уведіть
ім'я" required>
      <p>Прізвище</p>
      <input type="text" name="surname" maxlength="50" placeholder="Уведіть
прізвище" required>
      <p>Ім'я по бітькові</p>
      <input type="text" name="familyname" maxlength="50"
placeholder="Уведіть ім'я по бітькові">
      <p>Пароль</p>
      <input type="password" name="password" maxlength="50"
placeholder="Придумайте та уведіть ваш пароль" required>
      <input type="hidden" name="_csrf" value="${_csrf.token}">
      <button type="submit" >Зареєструватися</button>
    </form>
  </div>
</div>
</body>
</html>
```

Б.4.15 – Модуль UserAnswer.ftlh

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="/css/Style.css" rel="stylesheet" type="text/css">
    <title>Відповідь учня</title>
</head>
<#include "header.ftlh">
<body>
    <div class="useranswer">
        <div style="border-bottom: 2px solid #959499;" class="header">
            <h1>Превірте завдання учня та поставте оцінку</h1>
        </div>
        <div class="question">
            <#list userquestion.getAnswers() as answer>
            <div style="padding:0px 45px;background-color:#D9CAB6;border-
bottom: 2px solid #959499;display:flex;aling-items:center;justify-
content:center;" >
                <h1 style="word-wrap: break-wrap;text-indent:
30px;">${answer.quest}</h1>
            </div>
            <#if answer.type == "test">
            <div style="padding:25px 100px;border-bottom: 2px solid
#959499;display:flex;aling-items:center;justify-content:center;flex-
direction: column; text-align: left;">
                <#list answer.getChecktasks() as ans>
                <div style="display:flex;align-items:center;">
                    <#if ans.istrue??>
                    <input style="width:22px;height:22px" type="checkbox"
checked disabled><p style="font-size:16px">${ans.answer}</p>
                    <#else>
                    <input style="width:22px;height:22px" type="checkbox"
disabled><p style="font-size:16px">${ans.answer}</p>
                    </#if>
                </div>
                </#list>
            </div>
            </#if>
            <#if answer.type == "images">
                <div style="flex-wrap: wrap;padding:5px 100px;border-bottom:
```

```
2px solid #959499;display:flex;aling-items:center;justify-content:center;">
                <#list answer.getImages() as img>
                    <a href="/images/${img.id}"><img
style="width:200px;height:200xp;margin:10px" src="/images/${img.id}"></a>
                </#list>
                </div>
            </#if>
            <#if answer.type == "textarea">
                <div style="padding:25px 200px;border-bottom: 2px solid
#959499;">
                    <p style="word-wrap: break-wrap;text-indent:
30px;">${answer.textarea}</p>
                </div>
            </#if>
            </#list>
            <form
action="/course/${course.id}/usersanswer/${userquestion.user.id}/${userquesti
on.question.id}" method="post">
                <p>Введіть оцінку учня</p>
                <input type="text" maxlength="10" name="rating">
                <input type="hidden" name="_csrf" value="${_csrf.token}">
                <button style="margin-bottom:40px;" type="submit">Виставити
оцінку</button>
            </form>
        </div>
    </div>
  </script>
</body>
</html>
```

Б.4.16 – Модуль UsersAnswer.ftlh

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="/css/Style.css" rel="stylesheet" type="text/css">
    <title>Сторінка перевірки завдань</title>
</head>
<#include "header.ftlh">
<body>
    <div class="useranswer">
        <div class="header">
            <h1>Оберіть завдання яке ви б хотіли перевірити</h1>
        </div>
        <div>
            <#list course.getQuestions() as question>
            <div style="padding:25px;display:flex;aling-items:center;justify-
content:center;background-color:#D9CAB6;border-top: 2px solid #959499;border-
bottom: 2px solid #959499;">
                <h1 style="margin:0;">${question.title}</h1>
            </div>
            <table class="table">
                <thead>
                    <tr>
                        <th>Ініціали учня</th>
                        <th>Оцінка</th>
                        <th>Перевірка відповіді</th>
                        <th>Видалити дані</th>
                    </tr>
                </thead>
                <#list question.getUserQuestions() as userquestion>
                <#if userquestion.status == "wait">
```

```
                    <tbody>
                        <tr>
                            <td>${userquestion.user.name}
${userquestion.user.surname} ${userquestion.user.familyname}</td>
                            <td>${userquestion.rating}</td>
                            <#if userquestion.rating == "-">
                            <td><a
href="/course/${course.id}/usersanswer/${userquestion.user.id}/${userquestion
.question.id}">Перевірити</a></td>
                            <td>
                                <form class="btnfordeletecourse"
action="/course/${course.id}/${userquestion.id}/deleteuserinfo"
method="post">
                                    <input type="hidden" name="_csrf"
value="${_csrf.token}">
                                    <input type="submit" value="X">
                                </form>
                            </td>
                            <#else>
                            <td><a style="color:red"
href="/course/${course.id}/usersanswer/${userquestion.user.id}/${userquestion
.question.id}">Повторно перевірити</a></td>
                            <td>
                                <form class="btnfordeletecourse"
action="/course/${course.id}/${userquestion.id}/deleteuserinfo"
method="post">
                                    <input type="hidden" name="_csrf"
value="${_csrf.token}">
                                    <input type="submit" value="X">
                                </form>
                            </td>
                            </#if>
                        </tr>
                    </tbody>
                <#elseif userquestion.status == "done">
                <tbody>
                    <tr>
                        <td>${userquestion.user.name}
${userquestion.user.surname} ${userquestion.user.familyname}</td>
                        <td>${userquestion.rating}</td>
                        <td><a style="color:green"
href="/course/${course.id}/usersanswer/${userquestion.user.id}/${userquestion
.question.id}">Перевіренно</a></td>
                        <td>
                            <form class="btnfordeletecourse"
action="/course/${course.id}/${userquestion.id}/deleteuserinfo"
method="post">
                            <input type="hidden" name="_csrf"
value="${_csrf.token}">
                            <input type="submit" value="X">
                            </form>
                        </td>
                    </tr>
                </tbody>
                </#if>
                </#list>
            </table>
            </#list>
        </div>
    </div>
</body>
</html>
```

Б.4.17 – Модуль UsersPage.ftlh

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <link href="/css/Style.css" rel="stylesheet" type="text/css">
  <title>Користувачі</title>
</head>
<#include "header.ftlh">
<body>
    <div class="users-main">
      <div class="users-header">
        <h1>Інформація про учасників курсу</h1>
        <a class="evaluations" href="/course/${course.id}/ratings">Оцінки</a>
      </div>
      <div class="adminsinfo">
        <h2>Викладачі</h2>
        <#list admins as usercourse>
        <#if usercourse.user.getPhoto()??>
        <div class="admin">
          <img src="/images/${usercourse.user.getPhoto().id}" alt="Фото
користувача"><p>${usercourse.user.name} ${usercourse.user.surname}
${usercourse.user.familyname}</p>
          <#if role=="ROLE_CREATER"&& usercourse.role != "ROLE_CREATER">
          <form class="btnforchangerole"
action="/course/${course.id}/${usercourse.id}/changerole" method="post">
            <input type="hidden" name="_csrf" value="${_csrf.token}">
            <button type="submit">Зробити учнем</button>
          </form>
          </#if>
        </div>
        <#else>
        <div class="admin">
          <img src="https://raw.githubusercontent.com/itchief/modx-
solutions/main/Login/04-upload-photo/01-default.svg" alt="Фото
користувача"><p>${usercourse.user.name} ${usercourse.user.surname}
${usercourse.user.familyname}</p>
          <#if role=="ROLE_CREATER"&& usercourse.role != "ROLE_CREATER">
          <form class="btnforchangerole"
action="/course/${course.id}/${usercourse.id}/changerole" method="post">
            <input type="hidden" name="_csrf" value="${_csrf.token}">
            <button type="submit">Зробити учнем</button>
          </form>
        </#if>
        </div >
        </#if>
        </#list>
      </div>
      <hr/>
      <div class="usersinfo">
        <h2>Учні</h2>
        <#list users as usercourse>
        <#if usercourse.user.getPhoto()??>
        <div>
          <img src="/images/${usercourse.user.getPhoto().id}" alt="Фото
користувача"><p>${usercourse.user.name} ${usercourse.user.surname}
${usercourse.user.familyname}</p>
          <#if role == "ROLE_ADMIN"||role=="ROLE_CREATER">
          <form class="btnforchangerole"
action="/course/${course.id}/${usercourse.id}/changerole" method="post">
            <input type="hidden" name="_csrf" value="${_csrf.token}">
            <button type="submit">Зробити вчителем</button>
          </form>
          </#if>
        </div>
```

```
        <#else>
        <div>
          <img src="https://raw.githubusercontent.com/itchief/modx-
solutions/main/Login/04-upload-photo/01-default.svg" alt="Фото
користувача"><p>${usercourse.user.name} ${usercourse.user.surname}
${usercourse.user.familyname}</p>
          <#if role == "ROLE_ADMIN"||role=="ROLE_CREATER">
          <form class="btnforchangerole"
action="/course/${course.id}/${usercourse.id}/changerole" method="post">
            <input type="hidden" name="_csrf" value="${_csrf.token}">
            <button type="submit">Зробити вчителем</button>
          </form>
          </#if>
        </div>
        </#if>
        </#list>
      </div>
    </div>
</body>
</html>
```

Б.4.18 – Модуль WelcomePage.ftlh

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <link href="/css/Style.css" rel="stylesheet" type="text/css">
    <title>Ласкаво просимо</title>
</head>
<#include "headerwo.ftlh">
<body>
<#if Error??>
<#include "Error.ftlh">
</#if>
    <div class="welcome">
        <div class="header">
            <h1 style="color:white;">Вітаю вас на нашому сайті</h1>
        </div>
        <div class="content">
            <h1>Щоб створити або увійти до курсу, вам потрібно в лівому
випадаючому списку обрати «Створити свій курс» або «Увійти до існуючого
курсу». Також в ньому будуть відображатися всі курси, до яких ви вже
увійшли</h1>
            <img src="/image/G1.gif" alt="Гайд">
        </div>
        <div class="content">
            <img src="/image/G2.gif" alt="Гайд">
            <h1>С права натиснув на картинку користувача, можна змінити вашу
особисту інформацію та встановити фотографію, а також змінити псевдонім.
Псевдонім буде відображатися в чаті курсу, тому відвідайте цю сторінку
першою</h1>
        </div>
        <div class="content">
            <h1>В кожному курсі є сторінка завдання, сторінка учасників, чат,
таблиця оцінок, а також на головній сторінці курсу розміщені оголошення</h1>
            <img src="/image/G3.gif" alt="Гайд">
        </div>
        <div class="content">
            <img src="/image/G4.gif" alt="Гайд">
            <h1>Кожен вчитель може створювати завдання, перевіряти виконані
завдання та ставити оцінки. Також вчителі можуть змінювати роль учнів на роль
вчителя</h1>
```

```
        </div>
    </div>
</body>
</html>
```

ДОДАТОК В

Слайди презентації

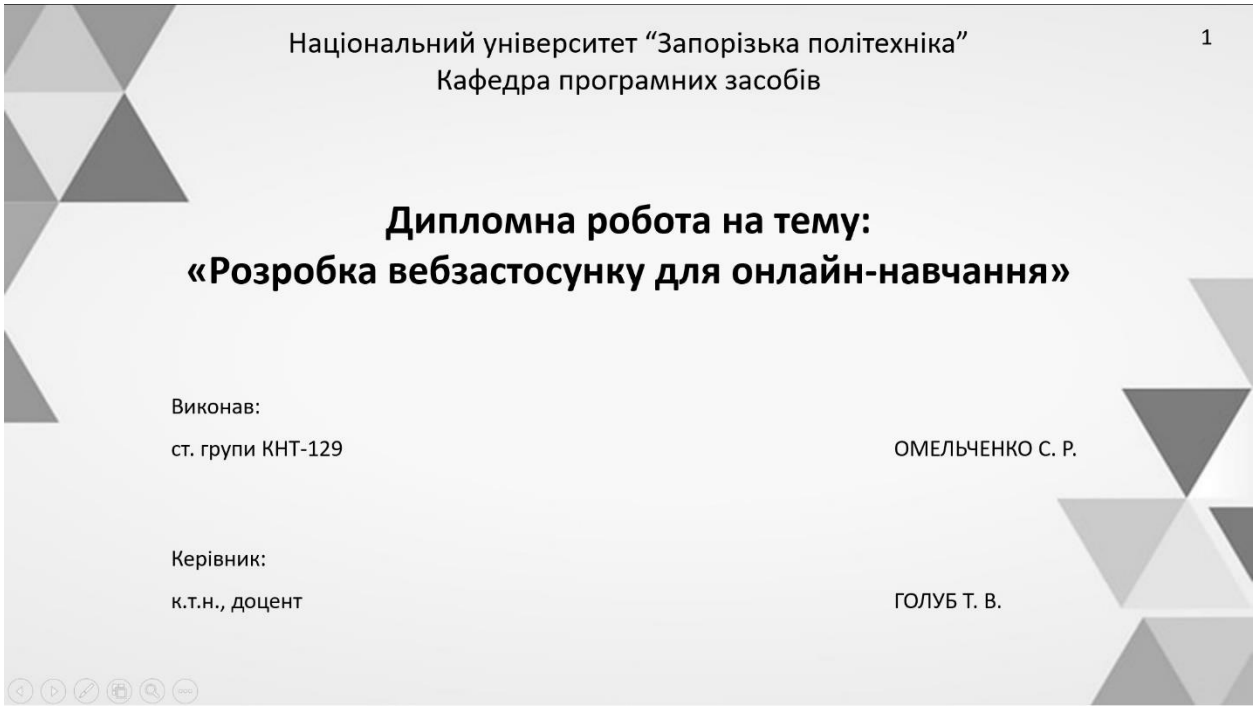


Рисунок В.1 – Слайд 1

Рисунок В.2 – Слайд 2

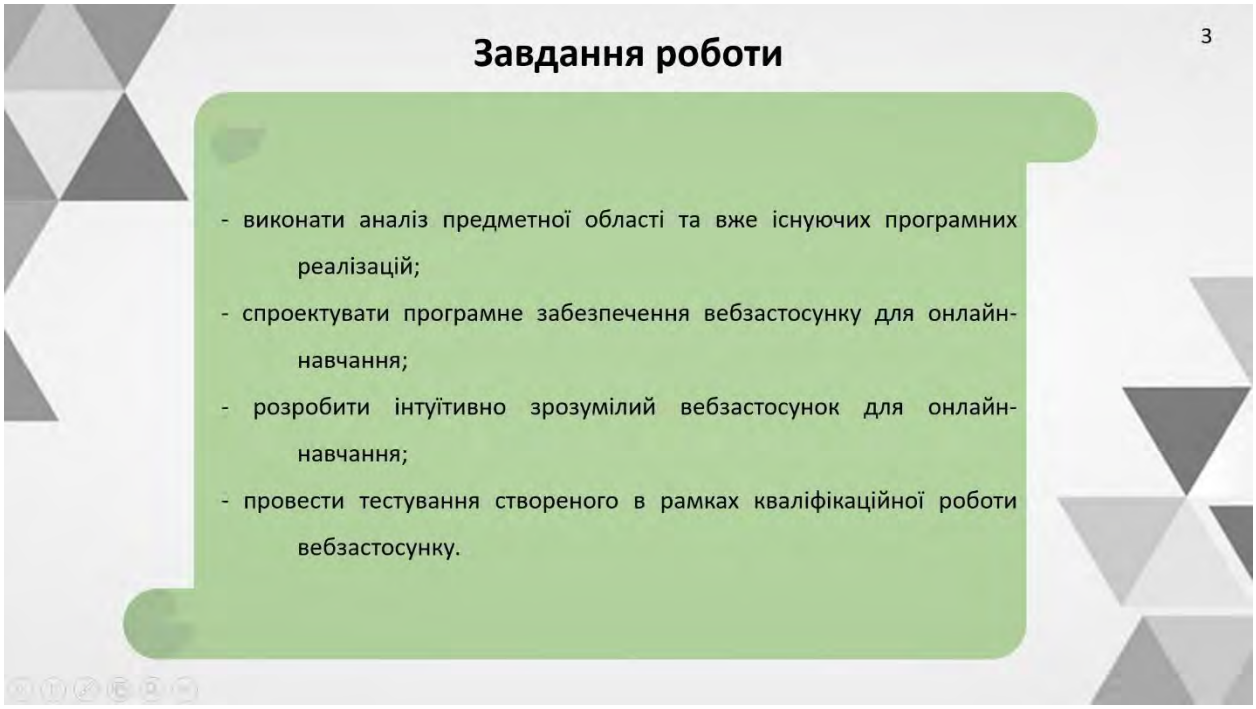


Рисунок В.3 – Слайд 3

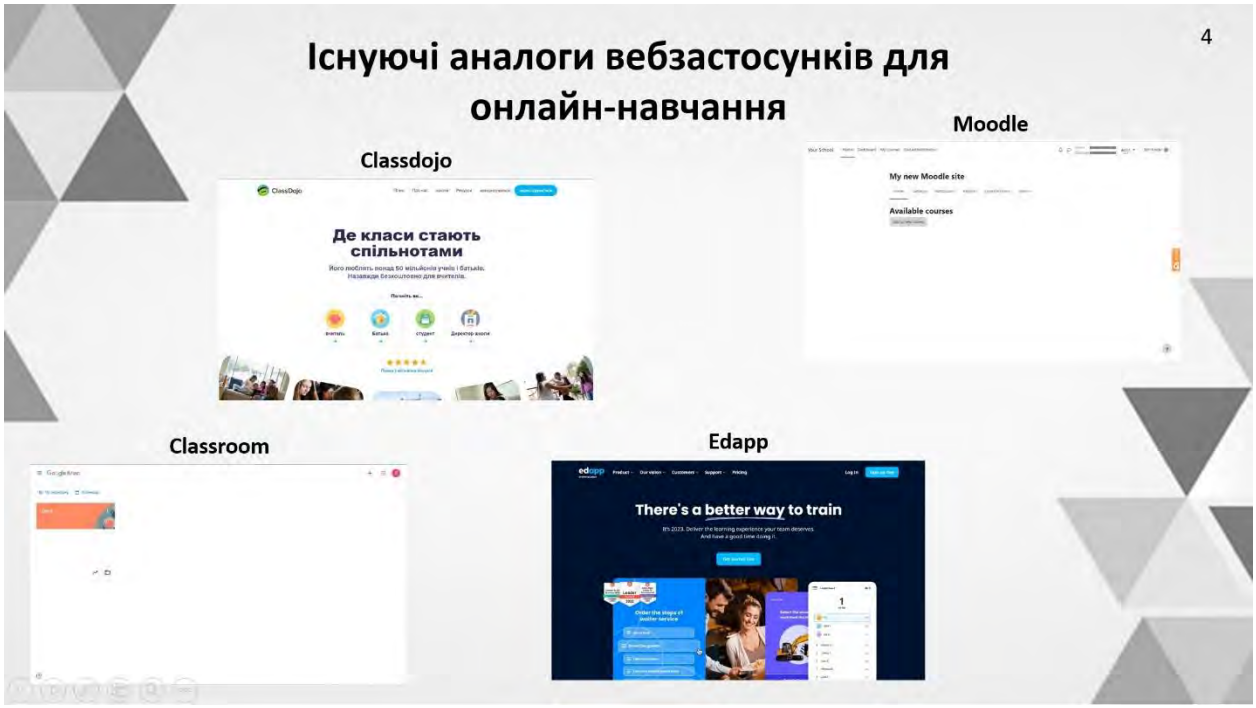


Рисунок В.4 – Слайд 4

Рисунок В.5 – Слайд 5

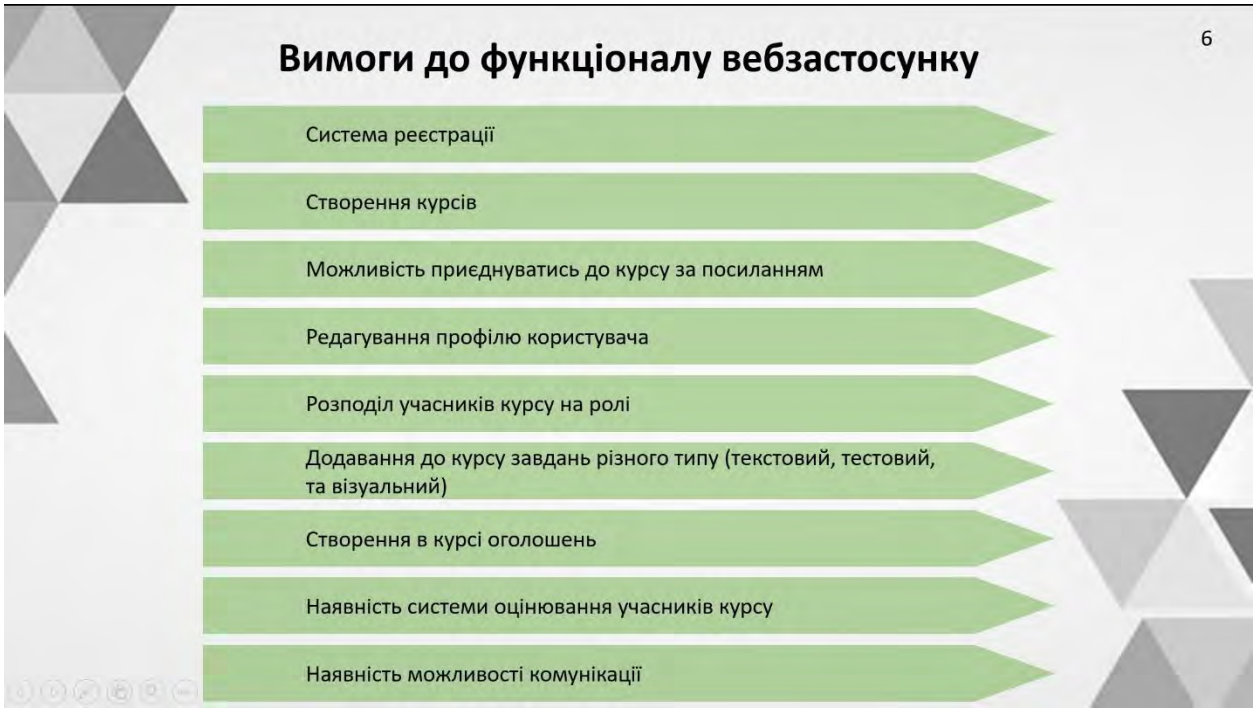


Рисунок В.6 – Слайд 6

Рисунок В.7 – Слайд 7

Рисунок В.8 – Слайд 8

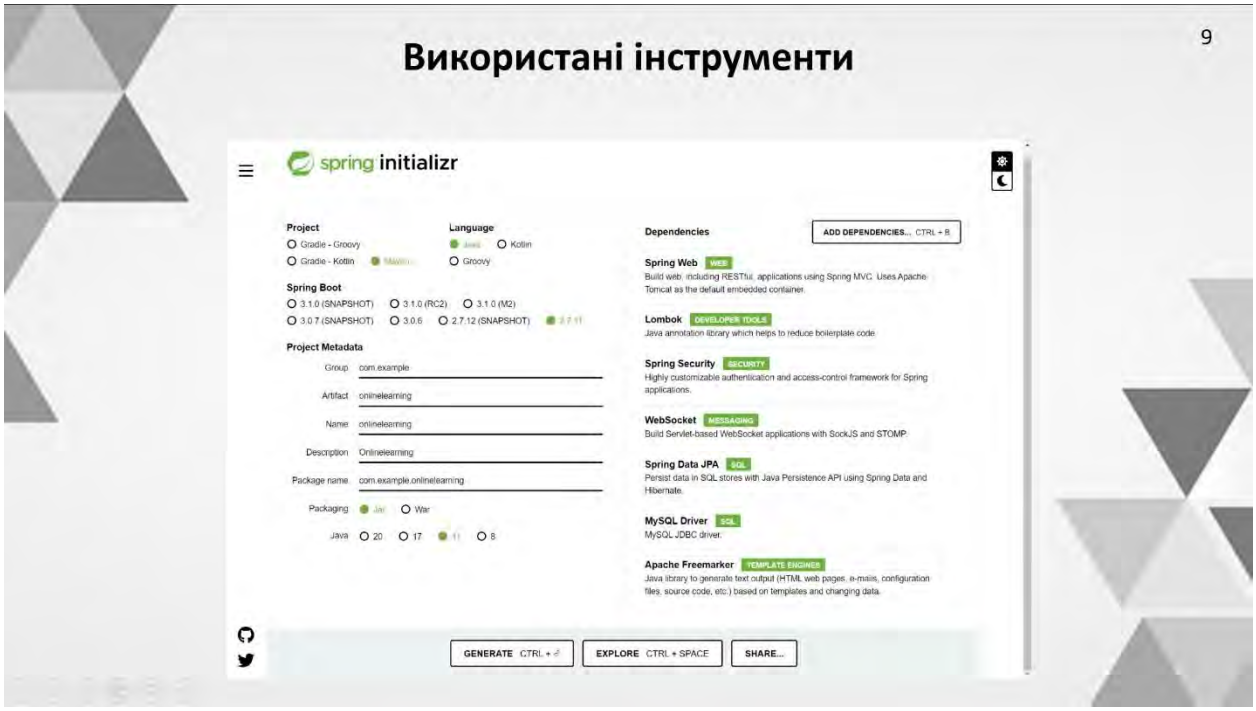


Рисунок В.9 – Слайд 9

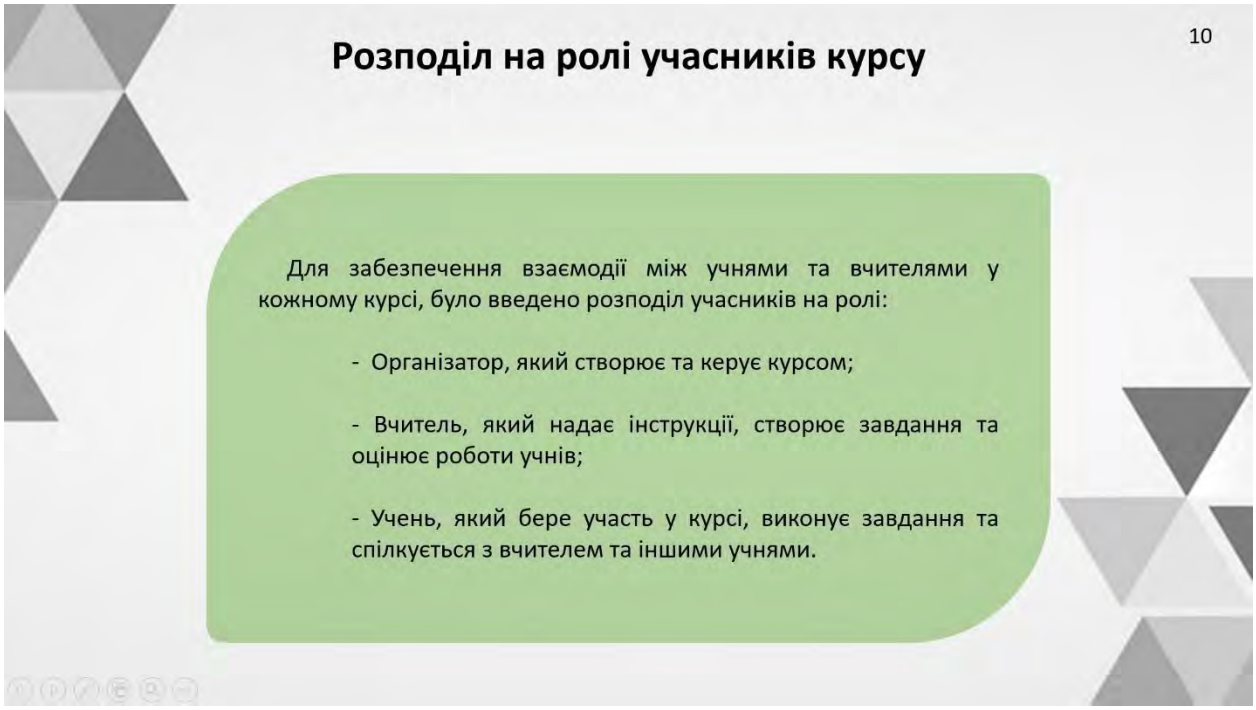


Рисунок В.10 – Слайд 10

Рисунок В.11 – Слайд 11

Рисунок В.12 – Слайд 12

Рисунок В.13 – Слайд 13

Рисунок В.14 – Слайд 14

Рисунок В.15 – Слайд 15

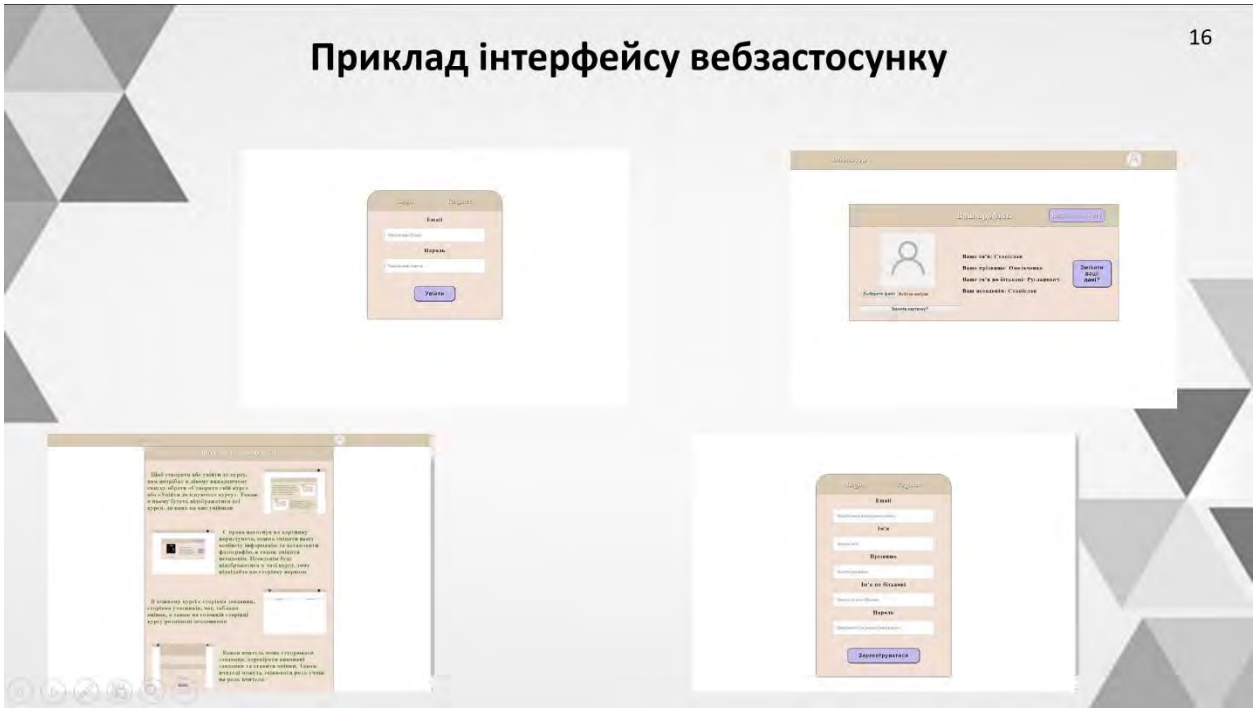


Рисунок В.16 – Слайд 16

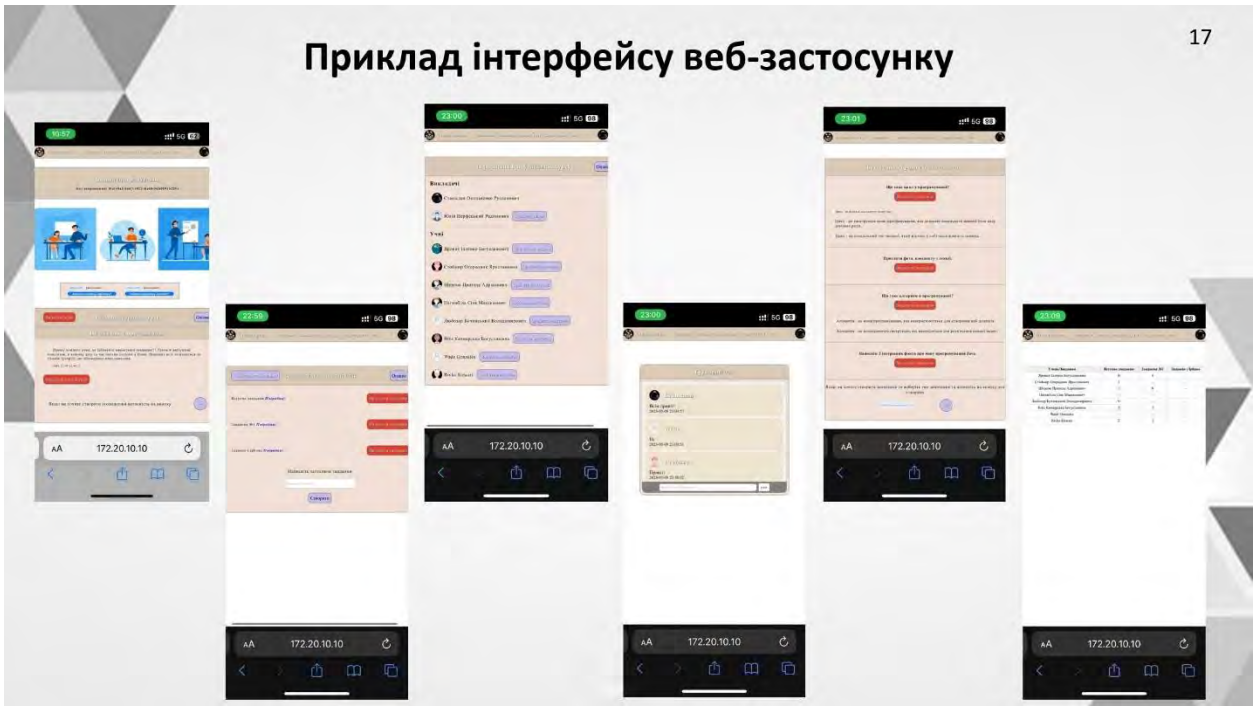


Рисунок В.17 – Слайд 17

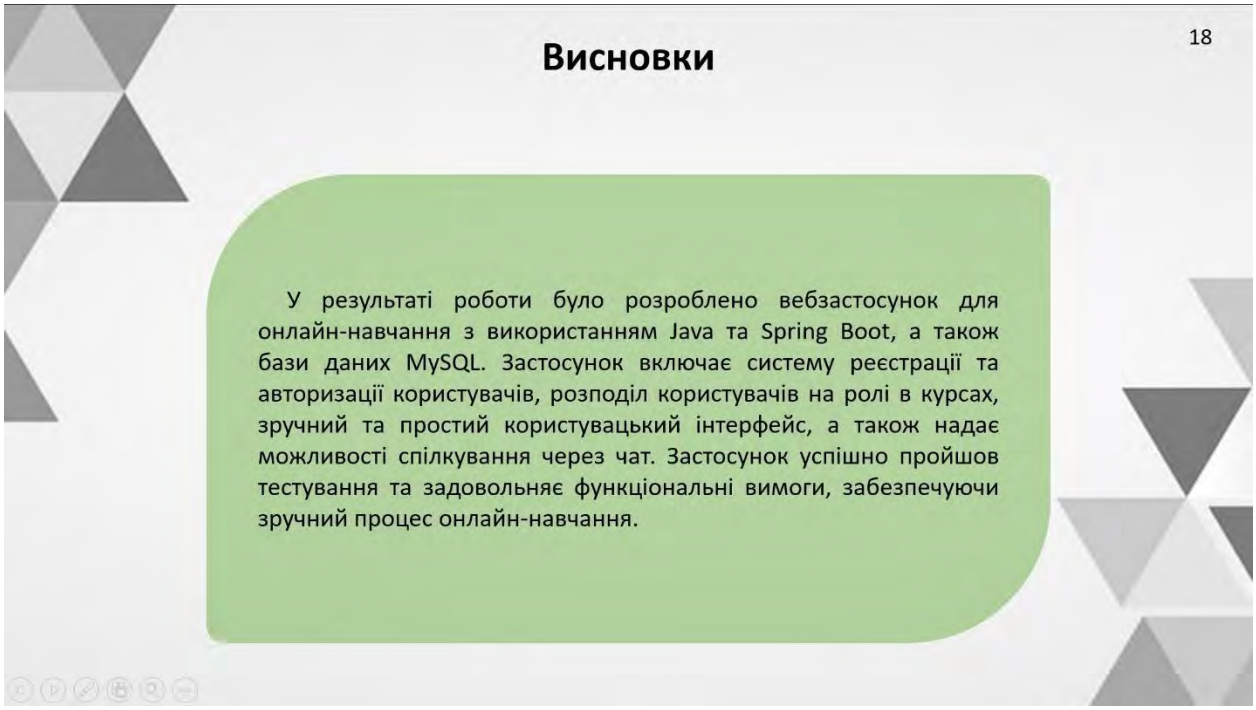


Рисунок В.18 – Слайд 18