Міністерство освіти і науки України

Національний університет «Києво-Могилянська академія»

Факультет інформатики

Кафедра інформатики

Кваліфікаційна робота

освітній ступінь – бакалавр

на тему: «**НАВЧАЛЬНА ПЛАТФОРМА ДЛЯ ФАКУЛЬТАТИВНОГО ВИВЧЕННЯ ВИБІРКОВИХ НАВЧАЛЬНИХ ДИСЦИПЛІН ШКІЛЬНОГО КУРСУ**»

Виконав: студент 4-го року навчання,

Освітньої програми «Інженерія програмного забезпечення», 121

Степаненко Любомир Владиславович

Керівник: Нагірна А.М.

Доцент, к. ф-м.н

Рецензент: Афонін А.О.

Доцент, к. ф-м.н

Кваліфікаційна робота захищена

з оцінкою

_____

Секретар ЕК

_____

«____» _____ 2024 р.

Київ – 2024

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ

УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ

Викладач кафедри інформатики ,

канд. фіз-мат. наук, доц.

_____ Гороховський С.С.

(підпис)

„____”_____2024р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на кваліфікаційну роботу

студенту Степаненку Любомиру Владиславовичу

факультету інформатики 4 курсу бакалаврської програми

ТЕМА: Навчальна платформа для факультативного вивчення вибіркових навчальних дисциплін шкільного курсу

Зміст ТЧ до кваліфікаційної роботи:

Індивідуальне завдання

Вступ

Аналіз навчальних вебплатформ

Розробка навчальної платформи

Висновок по роботі

Джерела

Дата видачі „___” _____ 2024 р. Керівник _____

(підпис)

Завдання отримав_____

(підпис)

Тема Навчальна платформа для факультативного вивчення вибіркових навчальних дисциплін шкільного курсу

п/п	Назва етапу кваліфікаційної роботи	Термін виконання	Примітка
1.	Обрання теми на кваліфікаційну роботу	22.09.2023	
2.	Огляд необхідних матеріалів та джерел	06.02.2024	
3.	Остаточне узгодження теми	04.03.2024	
4.	Реалізація програмної бекенд-частини	10.04.2024	
5.	Реалізація програмної фронтенд-частини	03.05.2024	
6.	Остаточне оформлення технічної частини з керівником	07.05.2024	
7.	Оформлення роботи	12.05.2023	
8.	Передзахист кваліфікаційної роботи	13.05.2024	
9.	Захист кваліфікаційної роботи	29.05.2024	

Студент Степаненко Л.В.

Керівник _____

“_____”_____

ЗМІСТ

АНОТАЦІЯ

Метою цієї кваліфікаційної роботи є розробка навчальної платформи для додаткового вивчення вибіркових навчальних дисциплін і предметів за вибором користувача. Робота пропонує ознайомитися з основними процесами, які стали невід'ємною складовою виконання поставленого завдання.

Першочергово проведено аналіз різних вебресурсів, подібних за своїм характером і функціоналом до визначеного задуму, та виокремлено основні акценти, необхідні для належної імплементації потрібного функціоналу.

Наведено виконання поставленого завдання з використанням різноманітних технологій full-stack веброзробки мовами програмування Java й Typescript, а також переглянуто функціональні можливості розробленого навчального застосунку.

Ключові слова: навчання, веброзробка, мікросервісна архітектура Java, Spring Boot, Liquibase, React, Typescript, Redux Toolkit.

ВСТУП

Актуальність теми і її практичне значення

Повторення вивченого матеріалу, за одним із популярних нині переконань, є запорукою успішного засвоєння набутих нових знань, оскільки таким чином ми допомагаємо самим собі утворювати певні чіткі патерни, що в подальшому грають немалу роль у втіленні отриманих навичок на практиці. Дослідження Шерон Томпсон-Шилл, Марка Еспозіто та Ірен Кан показало, що процес повторення активує ліву лобову долю головного мозку, яка має важливу значення в семантичному відновленні та обробці інформації. Цей нейронний механізм сприяє закріпленню знань та формуванню стійких зв'язків у нашому мозку, що підвищує здатність до застосування навичок у різноманітних ситуаціях, покращуючи таким чином загальну здатність до навчання [1].

Додатково, факультативне вивчення дисциплін дозволяє студентам і/або школярам вибирати ті галузі знань і наук, які їх найбільше цікавлять, і це таким чином збільшує їхню мотивацію до навчання. Певна автономія у виборі предметів сприяє розвиткові самодисципліни й самостійності здобувачів освіти, важливих для успішної академічної та професійної діяльності. Тож додаткове занурення в обрані теми дозволяє учням та ученицям розвивати критичне мислення, аналітичні навички і здатність до інноваційного підходу, що є основою для вирішення складних проблем і здатності адаптуватися до швидкозмінних умов сучасного навчального й робочого середовищ. Звідси випливає, що стратегія вибіркового повторення, застосована в процесі освіти, може стати важливим інструментом у формуванні добре обізнаного та гнучкого фахівця в певній області.

У найближчому майбутньому потреба в українських спеціалістах у різних наукових галузях лише зростатиме й зростатиме, оскільки надважливим завданням у післявоєнній відбудові України стане повноцінне становлення й відновлення наукових інституцій різних напрямків задля збільшення економічного, наукового та освітнього потенціалу нашої держави. Через

різноманітні виклики, пов'язані з війною та відновленням після неї, існує значний попит на спеціалістів з затребуваних за вищезазначених умов напрямків, куди входять інженерія, технології, медицина й соціальні науки [2].

Саме тому можна без жодних сумнівів вважати критично необхідним розвиток простої спеціальної навчальної платформи, що б надала змогу кожному охочому учневі й учениці отримувати нові додаткові знання з вибраних ними ж предметів, одночасно з цим підкріплюючи вже досягнені раніше профільні навички. Це могло б дати вагоме підґрунтя для збільшення інтересу до наукової царини серед дітей та юнацтва, а також допомогти їм у власних повсякденних намаганнях засвоїти необхідний шкільний і не тільки матеріал на кращому рівні.

У результаті був створений освітній вебресурс, який не тільки зможе вдосконалити процеси вивчення за вибором користувача, але й забезпечити певну гнучкість у підходах до освіти в Україні через інтеграцію сучасних технологій. Платформа дозволяє користувачам ефективно засвоювати і закріплювати знання в основному в тестовому форматі, що здатне значно підвищити засвоюваність вивченого. Отже, наявність такого вебзастосунку відкриває нові можливості для особистісного й професійного розвитку школярів, сприяючи при цьому формуванню готовності до викликів сучасного динамічного світу.

Структура роботи

Дана кваліфікаційна робота складається з двох основних розділів.

Перший розділ є аналізом подібних навчальних вебресурсів, дослідженням функціоналу кожного з них. Окреслено їхні переваги й недоліки, оцінюється потреба в розробці універсального навчального сервісу для додаткового й детального вивчення вибраних дисциплін.Другий розділ складається з опису кроків у розробці відповідного вебзастосунку, опис і обґрунтування вибраних шляхів реалізації задума, показ інтерфейсу й використання розробленого сервісу.

РОЗДІЛ 1. АНАЛІЗ НАВЧАЛЬНИХ ВЕБПЛАТФОРМ

1.1 Приклади реалізації визначеного задуму

Всесвітня мережа Інтернет неймовірно багата на інформаційні ресурси для будь-якого загалу, і освітні портали аж ніяк не є винятком. Усі вони легко знаходяться в декілька кліків і готові запропонувати користувачам свої онлайн-послуги.

Тож предметом аналізу стануть різні навчальні платформи, що надають можливість засвоєння освітніх матеріалів для охочих. Для кожного з наведених ресурсів позначимо загальні тенденції в реалізації подібного до нашого задуму, окреслимо їхні ймовірні переваги та недоліки, а також підіб'ємо підсумки з приводу основних якостей, що є бажаними для нашого навчального порталу.

Для власного дослідження знайдено 5 вебсайтів, що відповідають описаній раніше критерії та загальному характеру онлайн-ресурса у вигляді освітньої платформи для факультативного вивчення предметів і дисциплін, обраних зацікавленим користувачем.

Почнімо з вебсайту освітнього проєкту SchoolToGo [3] (рис. 1.1)

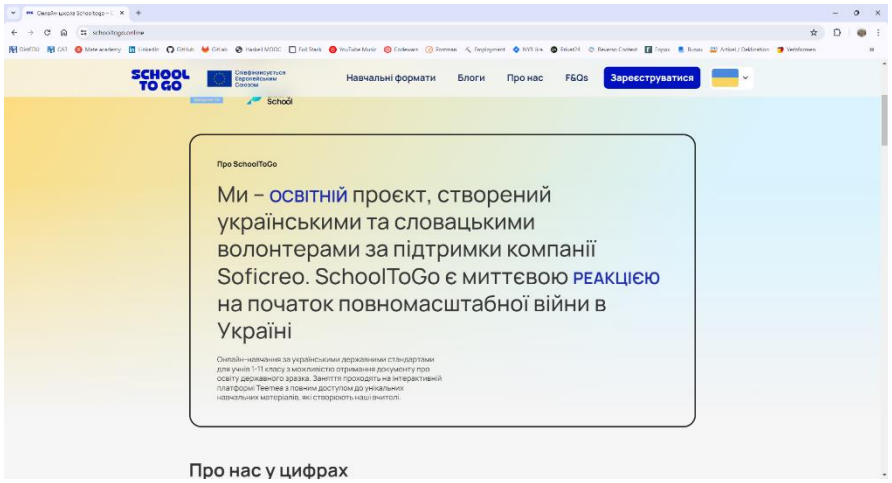


Рисунок 1.1 Скриншот головної сторінки освітнього проєкту SchoolToGo

Даний сервіс постає перед користувачами як платформа для онлайн-навчання за українськими стандартами й практиками. Він пропонує нам послуги

щоденних онлайн-уроків із викладачами у фіксованому та більш вільному форматах, щоб забезпечити доступ учнів до якісної освіти, незалежно від їх фізичного розташування в умовах воєнного часу.

Серед реалізації платформи тут варто виділити такі вагомі переваги, як приємна оку кольорова гама та відсутність так званих порожніх місць на сторінках сайту, які не наповнені жодним вмістом. Усе виглядає структуровано, лаконічно й заохочує до користування цією навчальною платформою (рис. 1.2).

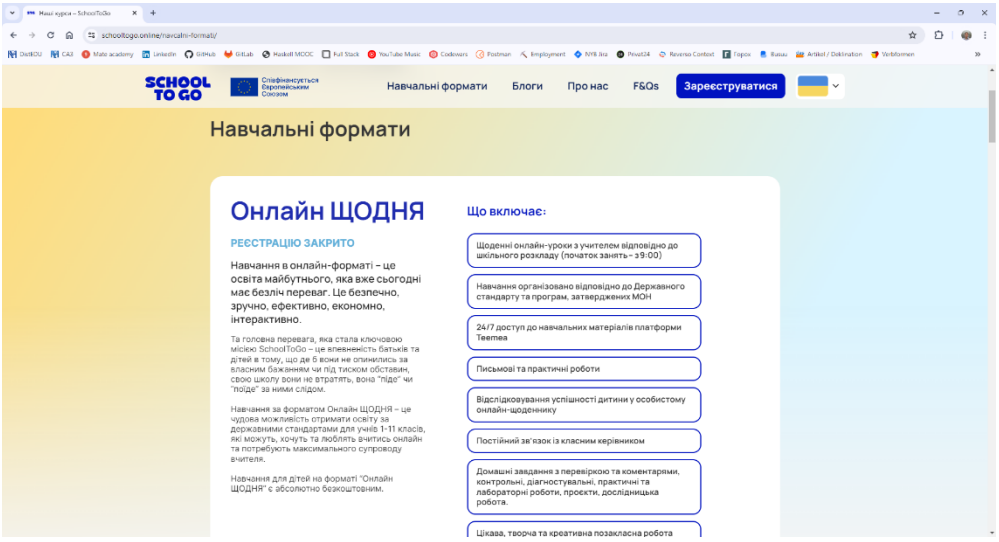


Рисунок 1.2 Скриншот вебсторінки освітнього проєкту SchoolToGo

Однак описаний сервіс має декілька нюансів, які можуть стати на заваді додатковому чи поглибленому навчальному процесу. Перш за все зазначимо процес реєстрації, яка наразі вказана як закрита що для формату навчання «Онлайн ЩОДНЯ», що для формату «Онлайн СМАРТ». Приклад цього наявний у рисунку вище. Серед відкритих форматів бачимо лише так звану двомісячну «Літню школу 2024», яка вже не є безкоштовною та має адміністративний збір у розмірі 500 грн (рис. 1.3)

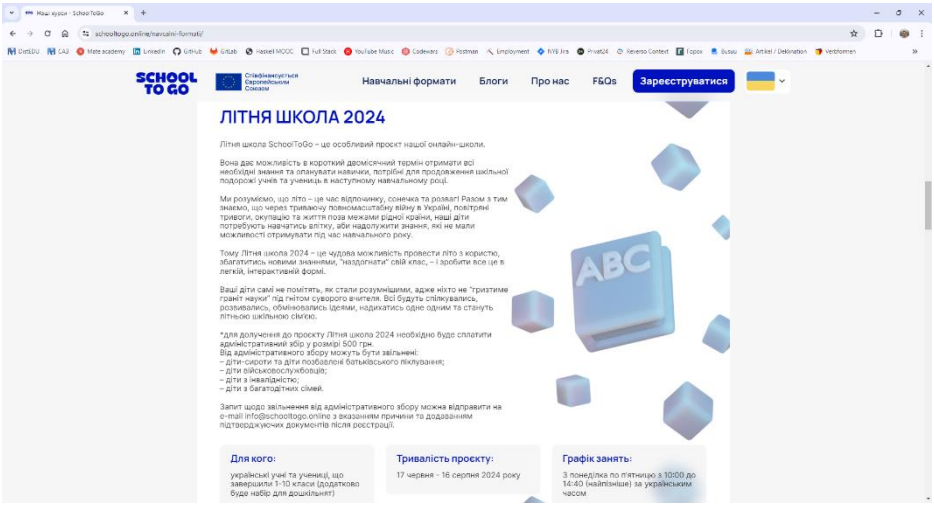


Рисунок 1.3 Скриншот вебсторінки освітнього проєкту SchoolToGo

Також, наскільки випливає зі сторінки самої реєстрації, даний процес можливий тільки з указанням контактних даних одного з батьків, що не дозволяє школярам зробити це самостійно за їхнього бажання (рис. 1.4)

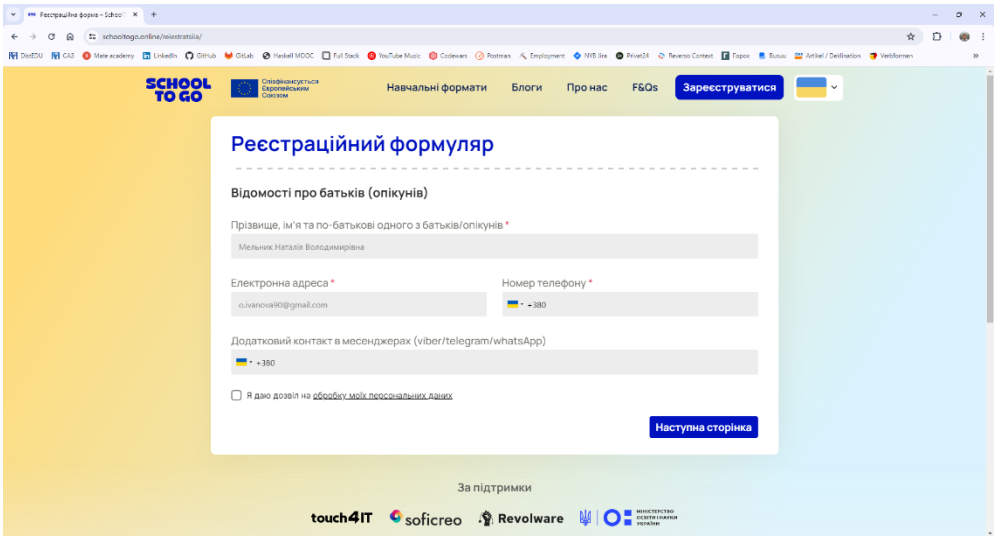


Рисунок 1.4 Скриншот сторінки реєстрації освітнього проєкту SchoolToGo

У підсумку освітній проєкт SchoolToGo – чудова українська ініціатива, що заохочує до отримання нових і не тільки знань, проте не зовсім відповідає поставленим нами цілям за рахунок додаткових кроків для повного входу на платформу, непостійній доступності до вебресурсу і позиціонування себе як онлайн-школи, що зрештою не є кінцевою метою нашої розробки.

Далі детально розберемо роботу сайту «ЗНО онлайн», що надає можливість абітурієнтам і не тільки проходити завдання у форматі ЗНО [4] (рис. 1.5).

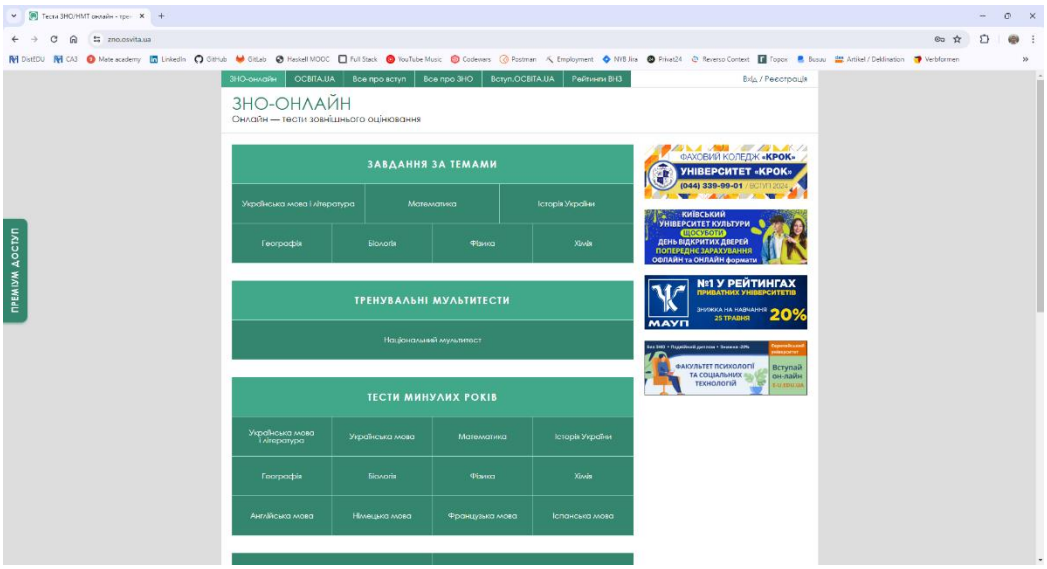


Рисунок 1.5 Скриншот головної сторінки вебплатформи «ЗНО онлайн»

Неймовірною перевагою вищевказаної платформи є перевага тестової форми завдань, оскільки саме вона є переважною на підсумкових шкільних іспитах (рис. 1.6). Також невід'ємним плюсом можна вважати простоту у візуалі даного вебресурсу: скористатися ним зможуть усі охочі.

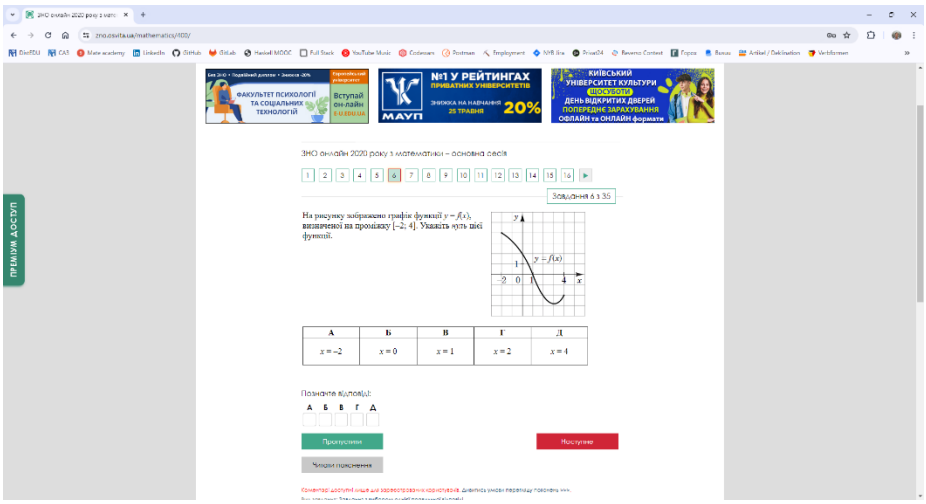


Рисунок 1.6 Скриншот сторінки завдання вебплатформи «ЗНО онлайн»

Тим не менш, сервіс дещо розчаровує своєю насиченістю на кожній можливій сторінці, що по-своєму насторожує, відволікає та відштовхує під час

роботи з ним (рис. 1.7). Хоч яка б не було забезпечення даного освітнього сайту, така величезна кількість онлайн-оголошень неприпустима й відлякуюча.

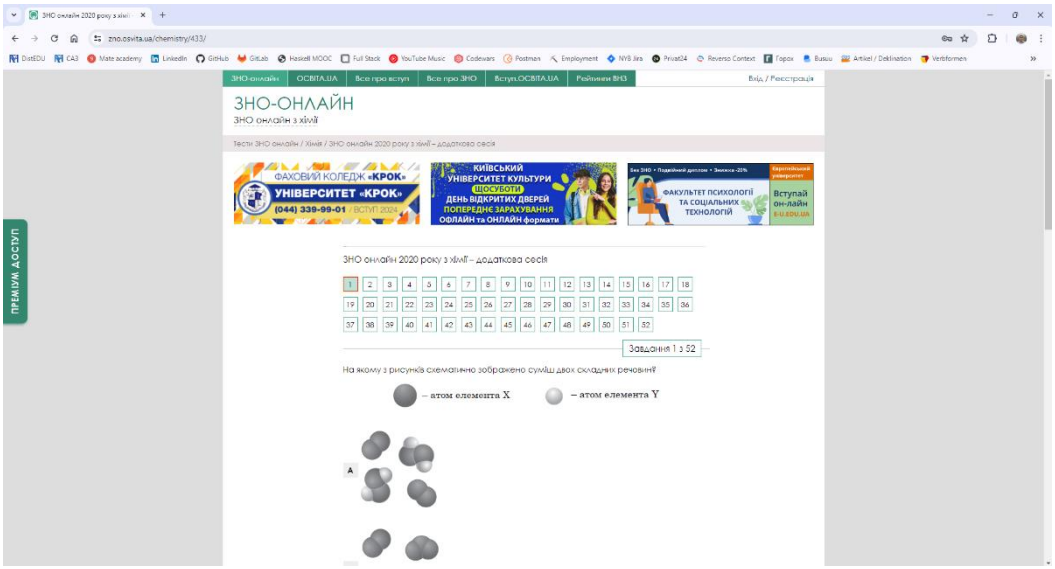


Рисунок 1.7 Скриншот прикладу насиченості рекламою вебплатформи «ЗНО онлайн»

Із точки зору функціоналу є прикрістю відсутність звичайних чи певних поглиблених навчальних тестів, окрім як формату ЗНО. Для нашого процесу розробки такий варіант не можна вважати до кінця підхожим.

На черзі маємо сервіс Learning.ua, який готовий запропонувати своїй аудиторії навчальні тести в ігровій формі від найменших користувачів до шкільних випускників [5]. Можна виділити якісний дизайн даного вебресурсу, який здатен неодмінно зацікавити охочих учнів та учениць (рис. 1.8).

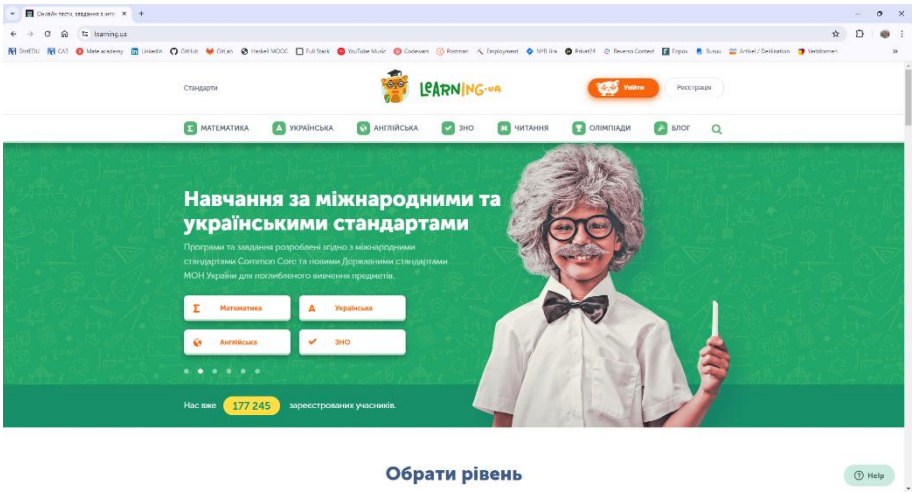


Рисунок 1.8 Скриншот головної сторінки сайту Learning.ua

На додачу варто підкреслити привабливу інтерактивність сервісу, забезпечену його здебільшого ігровою формою та цікавими візуальними рішеннями (рис. 1.9).

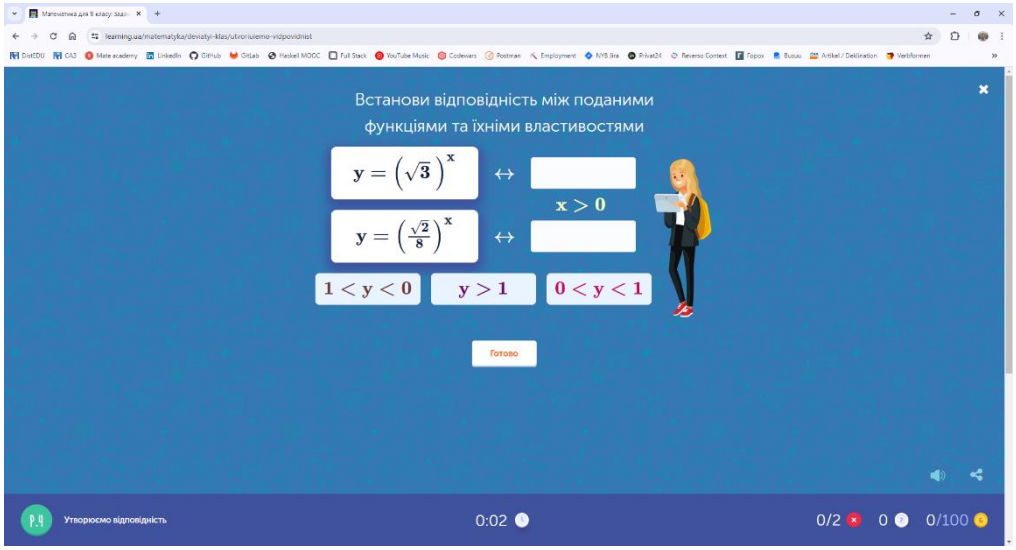


Рисунок 1.9 Приклад сторінки завдання на сайті Learning.ua

Вагомим мінусом цієї платформи хочеться назвати зовсім невелику кількість запропонованих дисциплін, серед яких математика, українська та англійська мови. Пропонуються також тести ЗНО в онлайн-вигляді, проте за це вже в основному відповідає сервіс «ЗНО онлайн», про який написано раніше.

Наступним освітнім порталом стане «На Урок», багатий на різноманітні навчальні матеріали [6] (рис. 1.10).

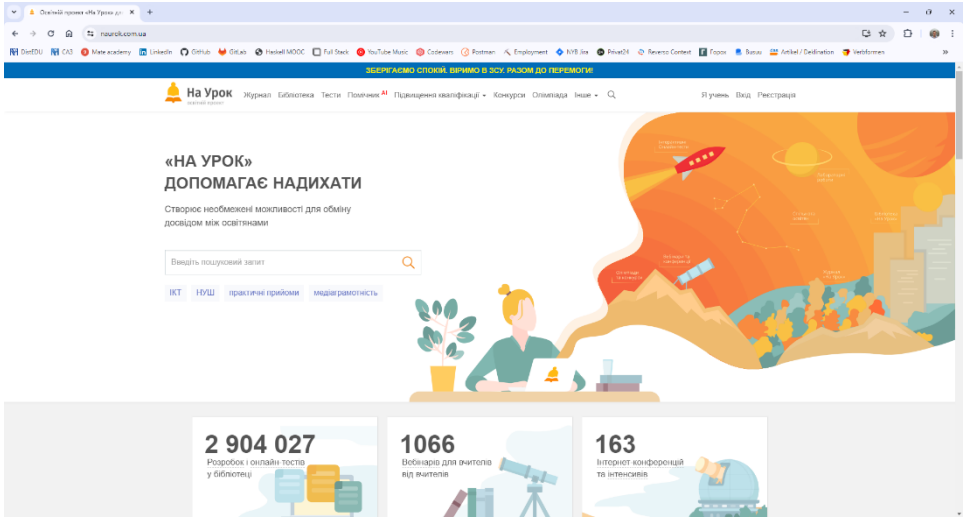


Рисунок 1.10 Скриншот головної сторінки порталу «На Урок»

Даний сайт має величезний об’єм різнопланового наповнення, чим він точно може похизуватися в порівнянні з іншими розглянутими платформами (рис. 1.11). Варто на додачу відмітити приємні для людського ока візуальні елементи.

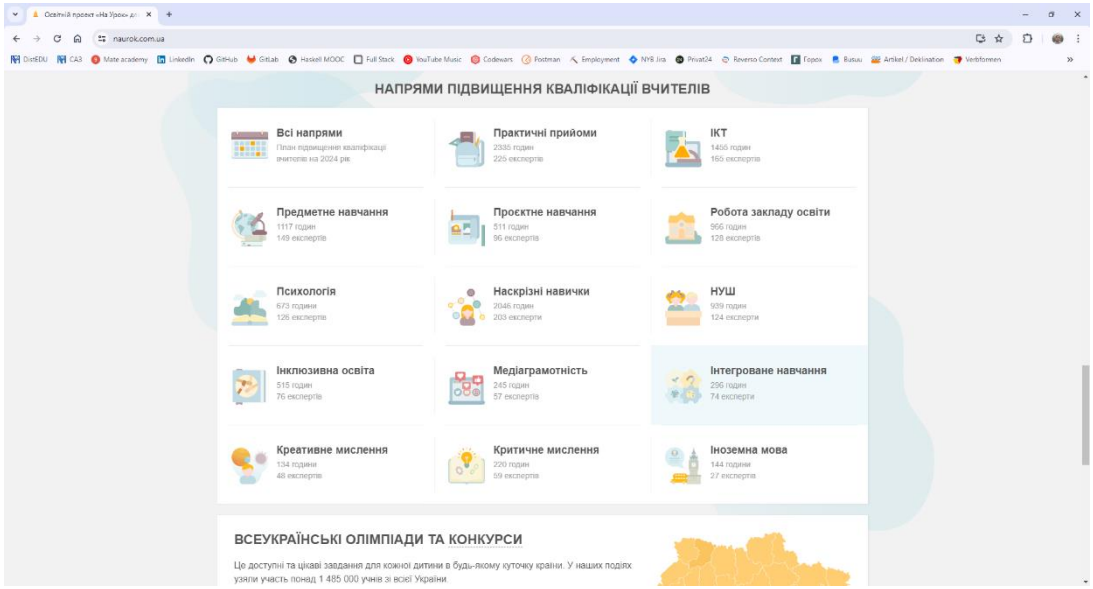


Рисунок 1.11 Скриншот головної сторінки порталу «На Урок»

Тим не менш, немає можливості розглядати цей вебресурс як вдалий приклад рішення описаного нами завдання, адже цільова аудиторія порталу «На Урок» – учителі українських шкіл, які прагнуть підвищити власну кваліфікацію, а не учні та учениці відповідно (рис. 1.12).

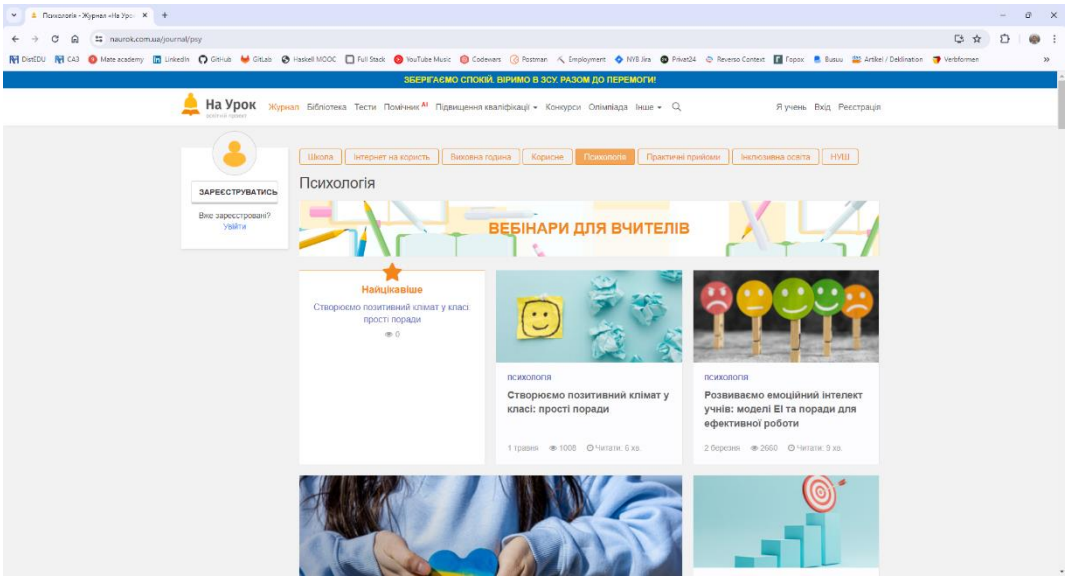


Рисунок 1.12 Скриншот з прикладом орієнтованості порталу «На Урок» на вчительську аудиторію

Під кінець розберімося з таким освітнім проєктом, як «МійКлас» [7] (рис. 1.13).

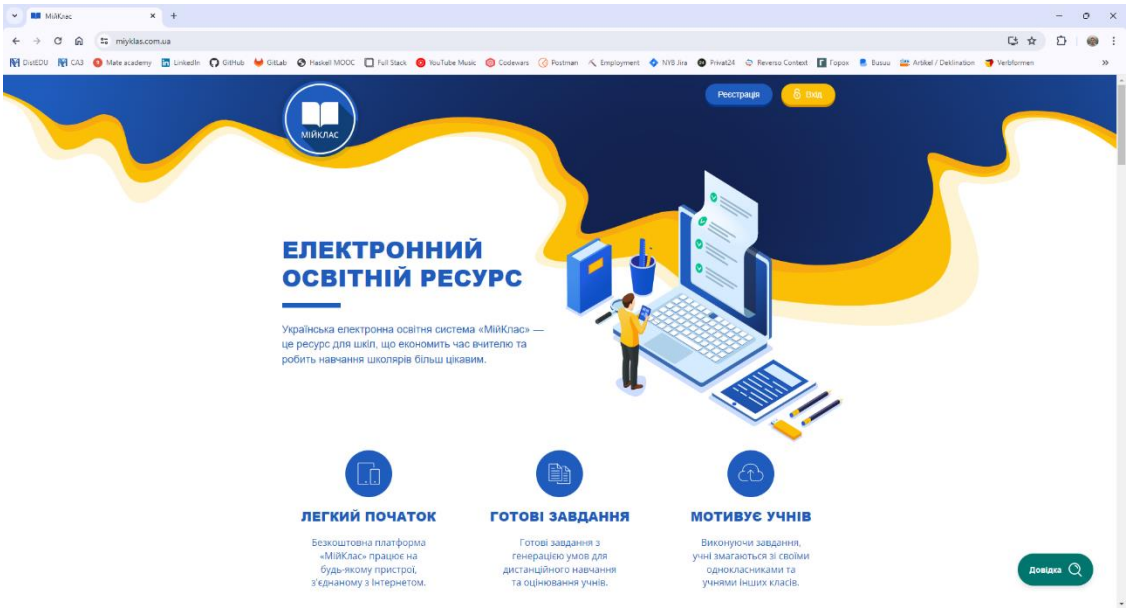


Рисунок 1.13 Скриншот головної сторінки проєкту «МійКлас»

Найосновнішою перевагою даної платформи є її доступність у поєднанні з офіційним схваленням віл МОН (рис. 1.14).

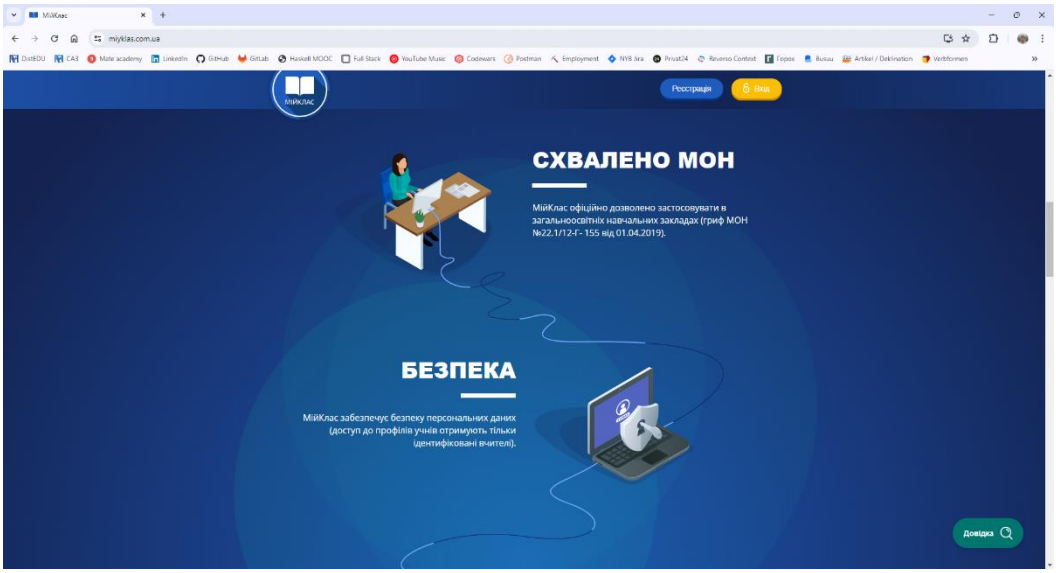


Рисунок 1.14 Скриншот інформації про схвалення від МОН на головній сторінці проєкту «МійКлас»

Також буде добре відмітити велику кількість уже готових навчальних завдань з різних предметів для школярів (рис. 1.15).

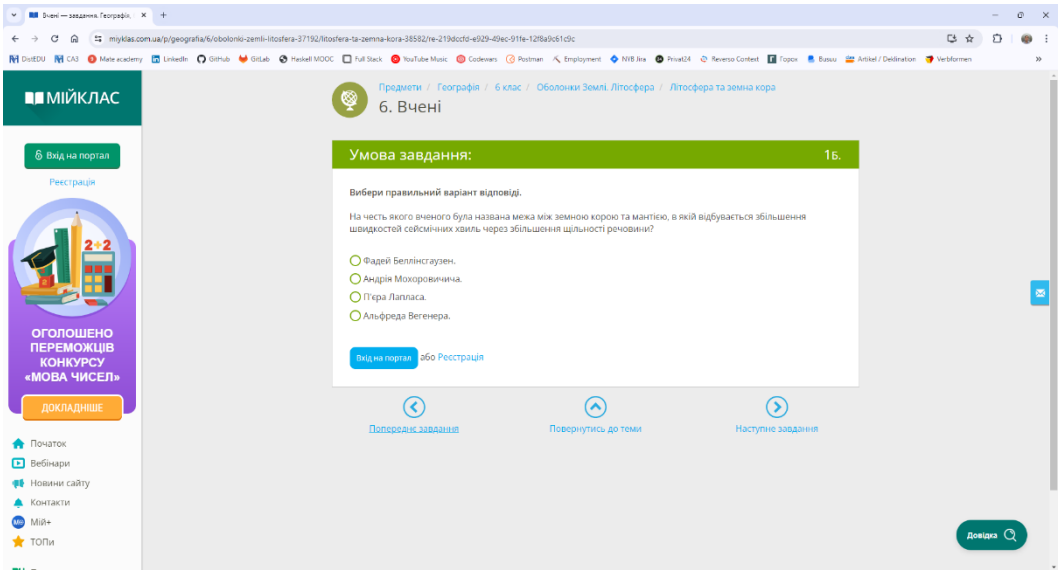


Рисунок 1.15 Готові завдання на сайті проєкту «МійКлас»

На жаль, серед мінусів описаного ресурсу можна виділити його недостатню автономність: користувач не може просто зареєструватися з вказівкою невеликої кількості персональних даних, адже йому також треба вказувати свій навчальний заклад. Це можна вважати дещо надлишковим у контексті допоміжної навчальної платформи, відкритої для всіх охочих учнів чи студентів.

Певним недоліком можна назвати й неналежну кількість тестів для певних окремих предметів, попри значне наповнення розглянутого вебресурсу. Зокрема для географії маємо лише теми для 6 та 7 класів, незважаючи на той факт, що цей предмет проходиться й надалі аж до 11 класу українських шкіл (рис. 1.16).

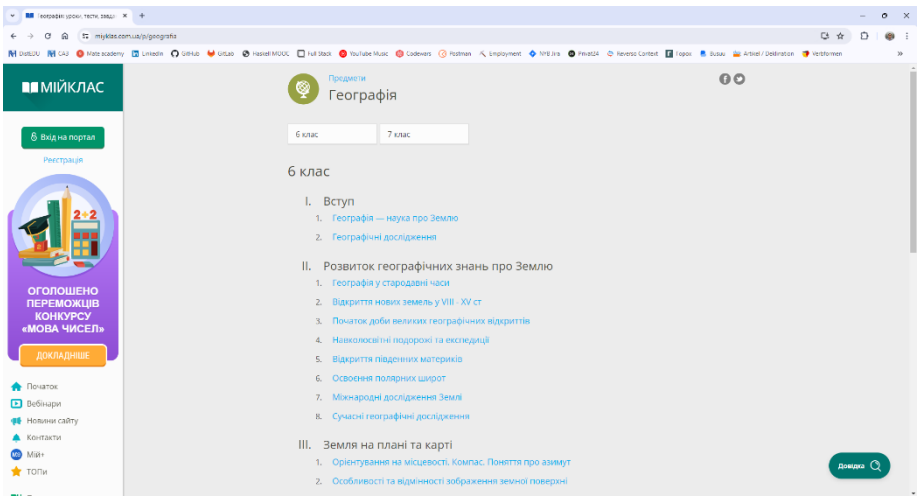


Рисунок 1.16 Скриншот сторінки завдань з географії проєкту «МійКлас»

Отже, було розглянуто 5 освітніх сайтів, що мають найбільш схожий функціонал і тематику, визначену нами для подальшої розробки. Окресливши їхні основні переваги та недоліки, маємо змогу виділити наступні тенденції:

- *Дизайн сайту*: В основному можемо спостерігати доволі непогані та якісні дизайнерські рішення; розглянуті сайти аж ніяк не наважишся назвати старомодними чи нерепрезентабельнимим. Однак на деяких відповідних онлайн-майданчиках усе ж було б добрим рішенням більше поекспереметувати з кольорами.
- *Доступність факультативного навчання*: Усі платформи забезпечують реєстрацію та автентифікацію користувачів, хоча в деяких із них цей процес займає більше часу, аніж в інших. Також описані освітні майданчики зосереджені більше на звичних для стандартної шкільної програми темах, що в цілому відповідає попередньо поставленому завданню.
- *Використання візуальних елементів*: Деякі сайти можуть здатися більше нагромадженими текстом, у інших же ж переважають усілякі візуальні наповнення. Але в цілому не відчувається дискомфорт при роботі з розглянутими порталами, окрім як поодиноких випадків із рекламними онлайн-банерами.

Аналізуючи ж розкриті вище тенденції, ми бачимо, що процес розробки навчальної платформи для факультативного вивчення вибіркових навчальних дисциплін шкільного курсу цілком у змозі мати потенціал і затребуваність, не гіршу від описаних раніше вебсайтів, бо серед його ключових переваг неодмінно стануть:

- *Простий дизайн*: Згідно з нашим аналізом, усі описані онлайн-майданчики багаті на вдалі дизайнерські прийоми та рішення, однак у цілому не виключено можливість зробити таку освітню платформу, яка матиме ще простіший дизайн для більш повної взаємодії користувачів із запропонованим їм інтерфейсом.

- *Зручність*: Загалом згадані вебсайти надають своїм користувачам необхідні послуги в зручний спосіб; проте є можливість зробити функціонал навчального ресурсу ще простішим із використанням мінімалістичних елементів.
- *Незалежність*: Розроблений нами вебзастосунок не залежатиме від конкретної навчальної установи, натомість буде доступним для будь-яких охочих користувачів, із основною орієнтацією на школярів або студентів.
- *Стандартизованість*: Наша платформа володітиме чіткою структурою завдань у здебільшого тестовому форматі в поєднанні з завданнями відкритої форми, що надасть змогу здобувачам освіти очікувати на належну структурованість викладеного навчального матеріалу, а можливим адміністраторам – легкість у створенні подальших тестових уроків.
- *Змагальний характер*: Розроблений вебресурс надаватиме топ із 10 користувачів із найвищими набраними балами за проходження уроків, що допоможе мати застосунку змагальний характер і підсилити жагу учнів та учениць до отримання хороших результатів і додаткового навчання як такого.

Наведені переваги описаного майданчика навчального спрямування демонструють його немале значення в сучасних цифрових реаліях і його потенціал для підсилення ролі освіти в умовах нинішньої та майбутньої української дійсності.

1.2 Висновки до розділу 1

У першому розділі ми переглянули основні пропозиції й тенденції наших днів щодо українських навчальних онлайн-платформ, описавши вибраних їх представників у мережі Інтернет. Було виокремлено їхні особливості та виділено їхні важливі плюси та мінуси.

Разом із цим було розглянуто основні моменти у функціоналі, що мають бути реалізовані в задуманому онлайн-майданчику з метою простої, швидкої та доступної можливості зануритися у вибрані навчальні дисципліни.

РОЗДІЛ 2. РОЗРОБКА НАВЧАЛЬНОЇ ПЛАТФОРМИ

2.1 Ідейна концепція застосунку

Ідейна концепція розробленого освітнього застосунку базується на стратегії забезпечення простоти, ефективності та структурованості додаткового навчального процесу. Основна мета полягає в тому, щоб у підсумку був розроблений інтуїтивно зрозумілий, легкий у використанні інтерфейс із ретельно налаштованим функціоналом «під капотом», який би водночас був і хорошим інструментом для здобуття факультативних знань.

Так як нашому програмному продуктові належить дотримуватися ясності й лаконічності, було вирішено дати йому назву «Dali», тобто українське «далі» латинкою. У такий спосіб ми маємо змогу презентувати наш застосунок як новий проєкт для додаткового, «поДАЛЬшого» навчання охочих, паралельно з цим не маючи назву, схожу на інші та/або напряму пов’язану з процесом навчання чи освіти.

Основним принципом дизайну «Dali» є мінімалізм, який дозволяє користувачам зосередитись на навчанні без зайвих відволікальних факторів. Використання спокійної й свідомо підібраної кольорової палітри лише сприяє зниженню візуального шуму і покращує зосередженість користувачів. Синій колір, який є наріжним каменем в нашому візуальному рішенні, асоціюється з надійністю, спокоєм та інтелектуальною діяльністю [8]. На додачу, згідно з дослідженнями, зазначений колір здатен стимулювати до роботи, зберігаючи при цьому атмосферу товариськості [9], тож його вибір можна вважати обґрунтованим.

Застосунок матиме основні ролі: адміністратор і звичайний користувач. Для ролі адміністратора буде надано можливість створювати персоналізовані уроки для обраних класів на вибрані теми, редагувати їх та переглядати результати найуспішніших за балами учасників; натомість звичайний користувач зможе проходити розроблені уроки, підіймати свій глобальний

рейтинг і ознайомитися з власною детальною статистикою. Це дає вчителям та учням не лише простоту в додатковому навчанні, але й забезпечує заохочуваність до користування даним вебресурсом за рахунок змагального характеру навчального процесу й можливої конкуренції між користувачами.

Особливу увагу приділено інтерактивності. Вправи та завдання, які реалізовані у застосунку, спрямовані на активне використання здобутих знань, що допомагає краще їх закріплювати. Тестові завдання з різними варіантами відповідей поєднані з завданнями відкритої форми, де здобувачі освіти мають спершу ретельно поміркувати над правильною відповіддю.

Статистика використання застосунку, включаючи кількість правильних відповідей, час, витрачений на кожне завдання, і загальну активність конкретного користувача, дозволяє учням відстежувати власний без участі жодного іншого посередника, що поступово стає невід'ємною частиною сучасного освітнього процесу.

Сам застосунок вирішено розробляти двома частинами:

- перша з них (відповідальна за бекенд-частину) написана в інтегрованому середовищі розробки Intellij IDEA з використанням мови програмування Java, яка досі користується немалим попитом за даними TIOBE станом на травень 2024 року [10];
- друга ж (відповідальна за фронтенд-частину) реалізована у Visual Studio Code мовою Typescript, яка наразі так само активно використовується в програмному виробництві [11].

Таким чином, розроблений застосунок не тільки відповідає нинішнім вимогам до освітніх платформ, але й утілює ту саму легкість використання, якої може часом так просто не вистачати.

2.2 Структурна схема додатка

Освітній вебзастосунок «Dali» відповідає наступним функціональним вимогам:

- Реєстрація, автентифікація та авторизація користувачів (адміністратор / звичайний користувач)
- Перегляд головного меню
- Створення уроків обсягом до 10 питань з тестовою та відкритою формами відповіді (адміністратор)
- Редагування створених уроків (адміністратор)
- Проходження уроків для визначеного класу на вибрані дисципліну та тему (звичайний користувач)
- Перегляд топу найкращих учнів за кількістю набраних балів (адміністратор / звичайний користувач)
- Перегляд персональної статистики (звичайний користувач)

Як уже було окреслено раніше, вебресурс розроблений двома окремими частинами, виділеними для бекенд- та фронтенд функціоналу. Постає необхідність описати кожну з цих частин, оскільки вони відрізняються одна від одної як концептуальною структурою, так і використаними технологіями. Розпочнімо з підрозділу, у якому реалізовані операції «під капотом» мовою програмування Java, орієнтованою на бекенд-розробку та імплементацію необхідної бізнес-логіки. Важливою його деталлю є використання мікросервісів як архітектурного програмного рішення (рис. 2.1).

Рисунок 2.1 Використання мікросервісної архітектури у вигляді окремих модулів проєкту в Intellij IDEA

Мікросервісна архітектура є однією з ключових властивостей сучасної розробки програмного забезпечення, особливо в контексті складних систем, які потребують високого рівня модульності й незалежності компонентів. В основу даної архітектури лежить принцип поділу великого монолітного додатка на менші, самодостатні сервіси, кожен із яких виконує визначену бізнес-функцію і комунікує з іншими сервісами через визначені API-запити [12].

Використання мікросервісів має кілька основних переваг:

- *Гнучкість у розробці*: Кожен із визначених мікросервісів може бути розроблений, оновлений і розгорнутий незалежно від інших, що в подальшому значно спрощує управління кодом і впровадження потрібних змін.
- *Масштабованість*: Через те, що сервіси функціонують як окремі компоненти, їх легко масштабувати залежно від поставлених вимог та визначеного навантаження, що дає змогу оптимізувати використання ресурсів.
- *Стійкість системи*: Відмова одного сервісу ніяк не впливає на роботу інших; навпаки, такий підхід покращує загальну надійність системи, адже проблеми в одному мікросервісі не ведуть до загальносистемного збою.

В контексті проєкту навчальної платформи «Dali», дане архітектурне рішення дозволяє інтегрувати різні аспекти освітнього процесу, такі як

керування користувацькими профілями, створення й виконання навчальних тестів, а також збір та аналіз персональної статистики, у незалежні один від одного модулі. Це не тільки по-своєму полегшує розробку й підтримку кожного окремого компонента, але також забезпечує можливість впровадження нових функціональних можливостей без переривання основної роботи нашої платформи.

Серед основних мікросервісних модулів наявні api-gateway, eureka-server, lesson-service та user-service, кожен із яких, відповідно до визначення описуваного архітектурного рішення, виконує свою конкретну роль у контексті бекенд-частини вебзастосунку.

Варто почати з сервісу eureka-server, оскільки він найменший за наповненням, але в той же час не поступається іншим модулям за значенням, а в дечому є й наріжним каменем взаємної роботи нашої системи.

Eureka-сервер в мікросервісній архітектурі відіграє критично важливу роль, діючи як сервіс виявлення (service discovery), який дозволяє автоматично визначати місцезнаходження інших мікросервісів у системі. Це вирішує одну з ключових проблем в мікросервісній архітектурі як такій – забезпечення зв'язку й співпраці між незалежними сервісами, які можуть бути динамічно розгорнуті чи переміщені без зміни загальної роботи системи [13].

Сервер Eureka діє як реєстр, де кожен мікросервіс зазначає свій доступний екземпляр та його адреси. Коли один із них запускається, він реєструється на Eureka-сервері з указівкою своєї мережевої адреси. Інші мікросервіси, які потребують взаємодії з цим сервісом, звертаються до Eureka-сервера для отримання актуальної інформації про адресу цільового сервісу. Таким чином, Eureka допомагає утримувати міжсервісний зв'язок актуальним і надійним, навіть у разі змін у конфігурації або ж відмови одного з них (рис. 2.2).

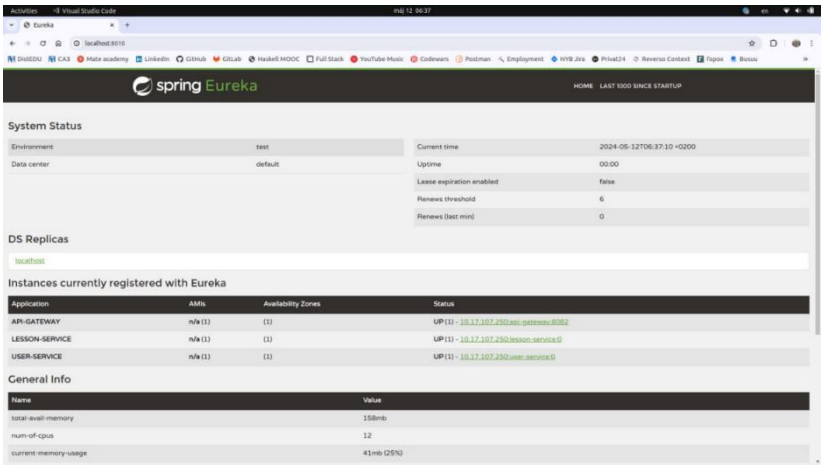


Рисунок 2.2 Приклад роботи Eureka-сервера в контексті роботи

У контексті освітнього проєкту "Dali" Eureka-сервер надзвичайно важливий з кількох причин:

- *Розподіленість і надійність*: Eureka дозволяє кожному сервісу знаходити інші собі подібні в нашій системі, забезпечуючи їх взаємодію та злагоджену роботу.
- *Розширюваність*: Проєкт може ефективно масштабуватися за рахунок додавання нових екземплярів мікросервісів без потреби вручну оновлювати конфігурації чи перезапускати чинні.
- *Самовідновлення*: У випадку відмови екземпляра сервісу, Eureka допомагає швидко визначити і виключити недоступні з них, дозволяючи системі продовжувати роботу, при цьому ж спираючись на інші доступні екземпляри сервісів.

Наступним вагомим мікросервісним модулем є api-gateway. Gateway API в мікросервісній архітектурі відіграє ключову роль як вхідна точка, що агрегує різні запити до служб і керує маршрутизацією, автентифікацією, авторизацією і моніторингом мікросервісів. Він служить як єдиний зовнішній інтерфейс, відкритим для користувачів та інших систем, тим самим абстрагуючись від внутрішньої складності розроблюваної системи [14].

У контексті нашого проєкту платформи для здобувачів освіти, на основі конфігурації Spring Cloud Gateway та її інтеграції з Eureka для виявлення сервісів,

Gateway забезпечує централізоване управління трафіком між користувачами та різними мікросервісами, які керують навчальними ресурсами, профілями користувачів і прогресом навчання. Він визначає різні маршрути до сервісів відповідно до URI-запитів і методів HTTP, описаних у визначеній нами конфігурації, що дозволяє нашій платформі масштабуватися й адаптуватися до змін у реалізації сервісів без зміни зовнішніх точок доступу (рис. 2.3).

Рисунок 2.3 Приклад визначених ендпойнтів для конфігурації Gateway API

Таким чином, відмінність у роботі Eureka-сервера та Gateway API полягає в тому, що перший фокусується на внутрішньому управлінні мікросервісами, забезпечуючи їх взаємне відкриття й довготривалість у рамках інфраструктури програми, у той час як Gateway API слугує як певний фасад, що керує взаємодією між клієнтами й внутрішніми сервісами, разом із цим обробляючи їхню комунікацію.

Далі розгляньмо модуль user-service, який цілком відповідає своїй назві: він займається стандартними запитами, пов’язаними з користувачами, такими як реєстрація, автентифікація, знаходження поточного користувача або ж користувача за вказаною електронною поштою, а також вихід з системи (рис. 2.4).

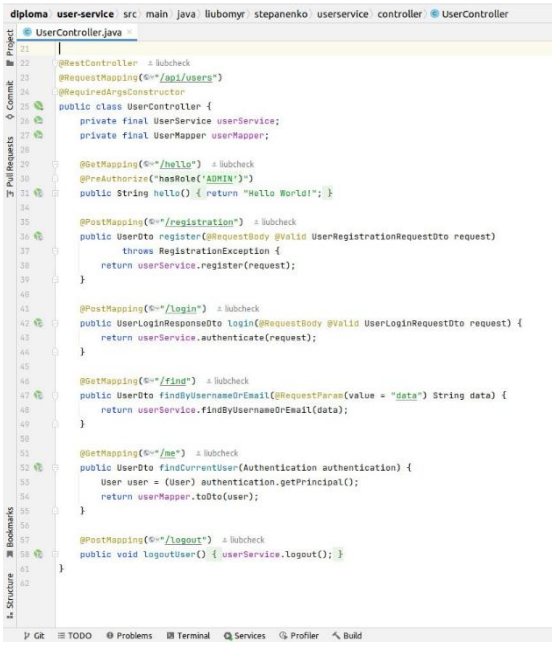


Рисунок 2.4 Скриншот класу UserController з мікросервісного модуля user-service

Зазначений мікросервіс уже на повну користується фреймворком Spring Boot, який хоч уже й використовувався до того в попередньо розглянутих модулях, але найосновніші його елементи застосовуються саме в мікросервісах user-service та lesson-service.

Spring Boot — це відкритий Java-фреймворк, призначений для програмування самостійних, орієнтованих на виробництво Spring-додатків із мінімальними зусиллями з боку розробника. Він широко використовується для побудови мікросервісів, веб-додатків та інших Java-орієнтованих проєктів завдяки його простоті використання та надійності [15].

Під час роботи зі Spring Boot не уникнути роботи з інверсією контролю (Inversion of Control, або ж IoC). Це один з основних принципів, на яких базується Spring Framework, і Spring Boot наслідує цей підхід. IoC забезпечує зниження залежності між програмними компонентами, дозволяючи керувати їхніми залежностями через зовнішнє джерело [16] (рис. 2.5).

Рисунок 2.5 Приклад роботи IoC у класі UserService мікросервісу user-service

Система роботи IoC в Spring Boot виглядає наступним чином:

- *IoC-контейнер*: У Spring Boot, як і в Spring, IoC реалізується за допомогою контейнера IoC, який керує створенням, налаштуванням і управлінням об'єктами (відомими як біни). Контейнер відповідає за життєвий цикл і конфігурацію бінів, а також за вирішення їхніх залежностей.
- *Dependency Injection (DI)*: IoC у Spring Boot часто реалізується через механізм Dependency Injection. DI дозволяє об'єктам отримувати їхні залежності ззовні через конструктори, сетери чи поля класів. Це зменшує залежність між компонентами програми та покращує можливості для тестування й підтримки коду.
- *Анотації*: Spring Boot спрощує конфігурацію IoC за допомогою анотацій. Наприклад, анотації *@Autowired* для автоматичного впровадження залежностей, *@Component* для оголошення бінів, а також *@Service* і *@Repository* для вказівки ролей компонентів у бізнес-логіці та доступі до даних відповідно.
- *Автоматична конфігурація*: Spring Boot автоматично конфігурує біни на основі вмісту класу та інших факторів. Це зменшує кількість необхідної конфігурації та дозволяє розробникам швидше почати виконання необхідних хм завдань.

Загалом же ж роботу мікросервісу user-service можна покроково описати його основними компонентами:

- *Контролери*: *UserController* обробляє HTTP-запити для реєстрації, входу в систему, пошуку користувачів і виходу з системи. Це демонструє, як

сервіс обробляє зовнішні запити та повертає відповіді. Даний контролер був висвітлений у рис. 2.4.

- *Сервіси*: *UserService* виконує бізнес-логіку, пов'язану з обробкою даних користувачів. Він включає методи для створення нових користувачів, аутентифікації, і видачі відповідей у разі включення в запити токенів доступу (рис. 2.6).

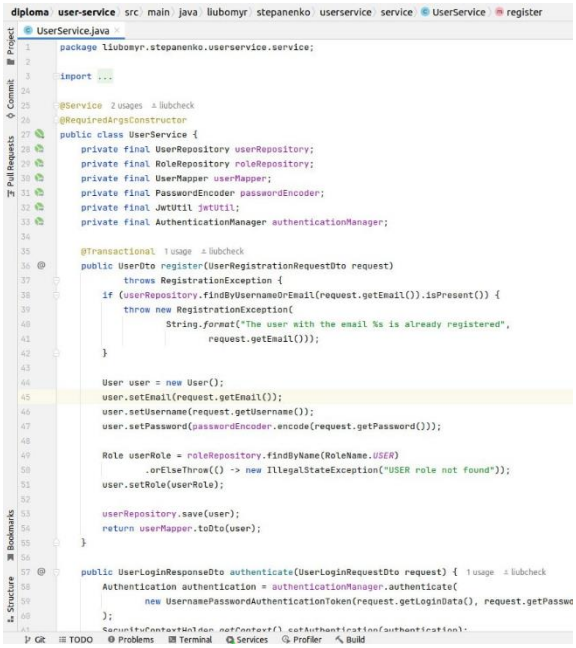


Рисунок 2.6 Скриншот класу UserService

- *Репозиторії*: Використання Spring Data JPA для взаємодії з базою даних здійснюється через *UserRepository* та *RoleRepository*, які дозволяють легко керувати даними користувачів і ролей у базі даних (рис. 2.7).

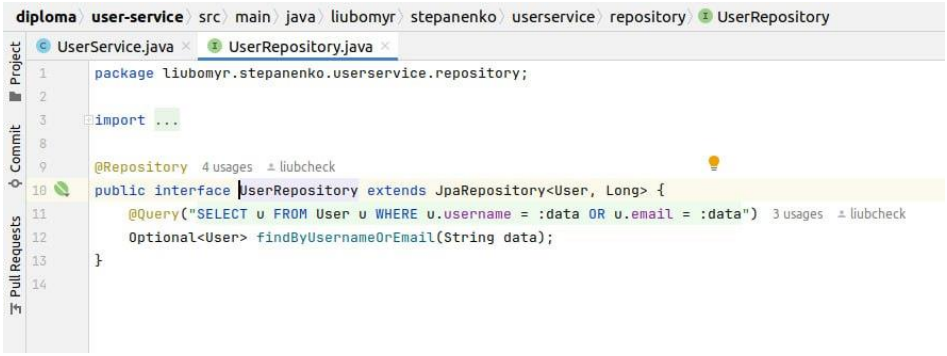


Рисунок 2.7 Скрншот інтерфейсу UserRepository

- *Мапери*: *UserMapper* та *RoleMapper* використовуються для перетворення між сутностями бази даних і передачі даних DTO (Data Transfer Objects), що є важливим для забезпечення розділення бізнес-логіки від клієнтської моделі даних (рис. 2.8).

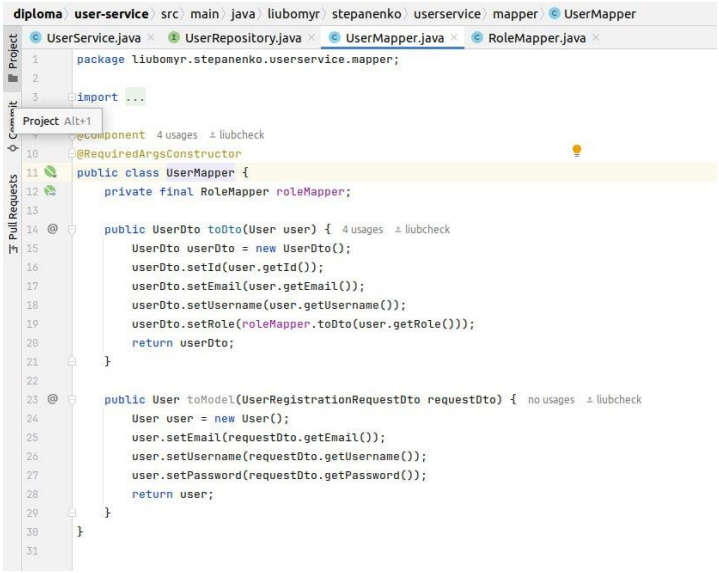


Рисунок 2.8 Скриншот класу UserMapper

- *Моделі*: класи *User* та *Role* є чи не найголовнішими в даному мікросервісі, оскільки вони напряму є відображенням тих таблиць бази даних, що використовуються нашою платформою. Моделі використовують анотації JPA для визначення відношень з таблицями користувачів і ролей та для управління

Рисунок 2.9 Скриншот класу сутності User

життєвим циклом сутностей через використання логічного видалення записів (рис. 2.9).

- *Безпека та автентифікація*: За допомогою Spring Security та JWT (JSON Web Tokens) сервіс надає можливість безпечної автентифікації й авторизації користувачів. Конфігурація ж самої безпеки за допомогою класу *SecurityConfig* визначає, як надати доступ до різних API на основі ролей користувачів (рис. 2.10).

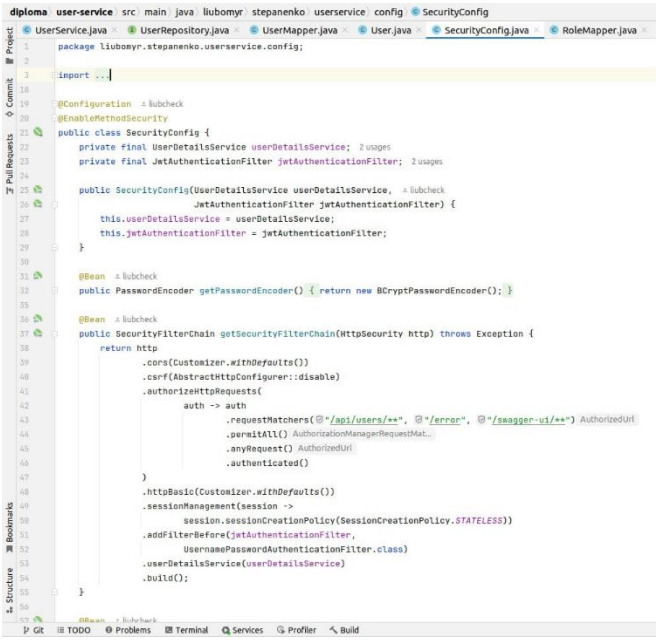


Рисунок 2.10 Скриншот класу SecurityConfig

Щодо крайнього модуля lesson-service вебресурсу «Dali», то там надзвичайно схожий принцип роботи, однак він уже зосереджений на взаємодії з сутностями уроку (*Lesson*), завдання (*Task*), варіанту відповіді (*Variant*) та одиниці користувацького прогресу за один урок (*Progress*).

Але яким чином відбувається взаємодія між мікросервісами? Яким чином ми, зрештою, можемо від модуля lesson-service достукатися до user-service або ж навпаки? На допомогу приходять так звані Feign-клієнти.

Feign – це декларативний HTTP-клієнт, що інтегрується зі Spring Boot і спрощує процес написання схожих клієнтів для взаємодії між мікросервісами. Він дозволяє розробникам створювати HTTP-клієнти легко та швидко за допомогою відповідних допоміжних анотацій [17] (рис. 2.11).

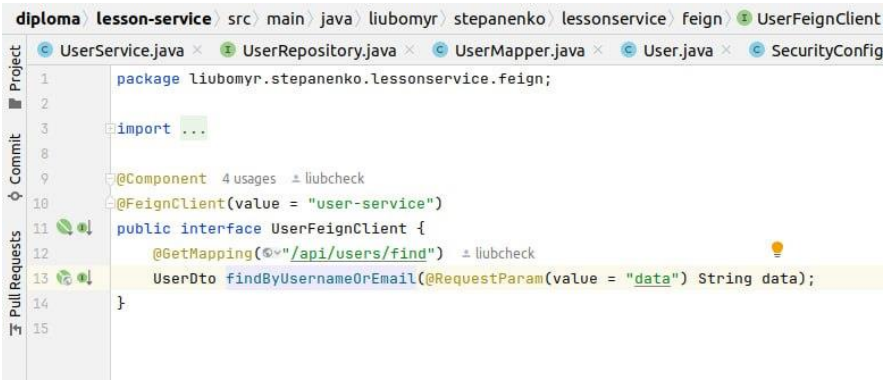


Рисунок 2.11 Скриншот інтерфейсу UserFeignClient для модуля lesson-service

Поряд із цим необхідно також обговорити систему контролю баз даних Liquibase, яка наявна що в модулі lesson-service, що в user-service для роботи з користувачами та ролями.

Liquibase – відкрите програмне забезпечення для контролю версій баз даних, яке дозволяє розробникам відстежувати, версіонувати й застосовувати зміни до схеми бази даних. Усе це спрощує процес управління базами даних у розподілених середовищах і сприяє покращенню співпраці між розробниками та іншими можливими сторонами [18]. Liquibase використовує формати опису змін, такі як XML, JSON, YAML або SQL для зберігання визначень змін, що дозволяє легко відстежити їхню історію й застосовувати ці зміни у різних середовищах (рис. 2.12, 2.13).

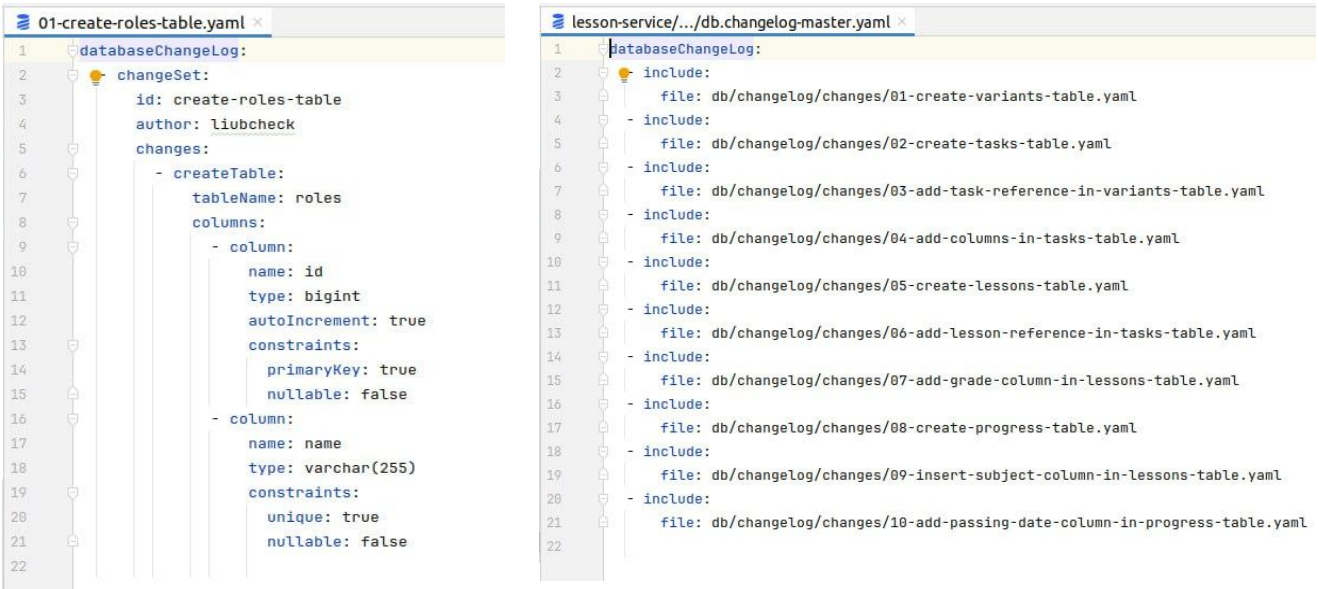


Рисунок 2.12, 2.13 Приклад роботи з Liquibase для платформи «Dali»

Зміни до схем можна застосовувати послідовно в будь-якому середовищі, що допомагає уникнути розбіжностей між різними стадіями розробки.

Liquibase може легко інтегруватися з автоматизованими інструментами розгортання та пост-розгортання, дозволяючи автоматично вносити зміни до бази даних під час розробки програмного забезпечення. Тому ця система – важливий інструмент у сучасному програмної інженерії, де потреба в швидкому й безпечному розгортанні має вирішальне значення для успіху проєктів.

Таким чином ми доволі детально розглянули структуру бекенд-складової навчальної платформи «Dali», тож тепер настав час перейти до фронтенд-частини нашого застосунку.

Зазначена складова, як уже зазначалося раніше, відтворена мовою програмування TypeScript, яка стала наступником JavaScript і, на відміну від попередниці, додає опціональну статичну типізацію, що дозволяє виявляти багато помилок ще під час компіляції, перш ніж код потрапить у виробництво. Це допомагає уникнути багатьох розповсюджених помилок, таких як спроби доступу до властивості неіснуючого об'єкта чи виконання операцій з несумісними типами даних [19].

Також TypeScript підтримує сучасні об'єктно-орієнтовані функції, включаючи класи, інтерфейси, наслідування, а також загальні типи (generics). Це забезпечує більш потужні та гнучкі способи структурування коду, спрощуючи розробку великих програмних систем.

Але дана мова програмування більш корисна та повна у використанні, коли вона йде поруч із різноманітними фреймворками, наприклад, React. Це — популярна JavaScript- і TypeScript-бібліотека для створення інтерфейсів користувача, яка пропонує ряд переваг для розробників. Однією з ключових особливостей React є використання віртуального DOM (Document Object Model), що дозволяє оптимізувати оновлення інтерфейсу, роблячи додатки швидшими та ефективнішими [20]. Реактивний підхід до рендерингу компонентів спрощує створення інтерактивних UI з відповіддю на зміну даних, а компонентна

архітектура дозволяє повторно використовувати код, зменшуючи загальний обсяг роботи і при цьому підвищуючи універсальність програми. Усе це робить React дуже затребуваним вибором для розробки як великих, так і малих веб-додатків.

Загалом фронтенд-частина «Dali» складається з безлічі файлів формату .tsx (здебільшого для компонентів) та .ts (інші файли, що забезпечують належний функціонал), обгорнутий в оболонку фреймворка React (рис. 2.14).

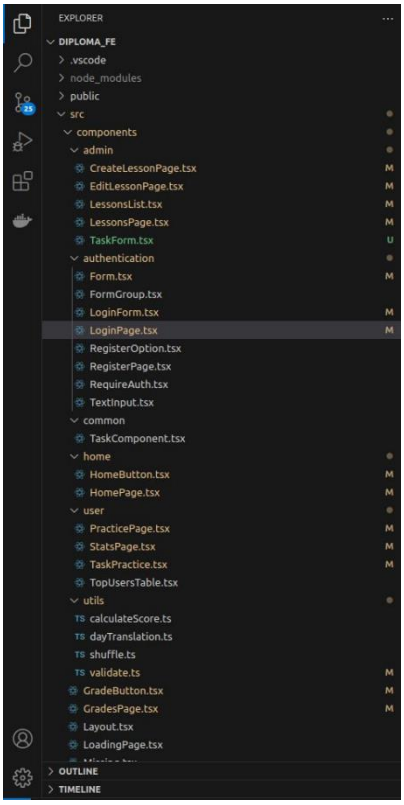


Рисунок 2.14 Архітектура фронт-енд частини платформи «Dali» у Visual Studio Code

Кожен із розроблених React-компонентів в основному реалізовує ту чи іншу сторінку розробленого навчального порталу, проте бувають і випадки, коли створені окремі компоненти для різних повторюваних елементів, як от кнопки, підписи, форми тощо (рис. 2.15, 2.16).

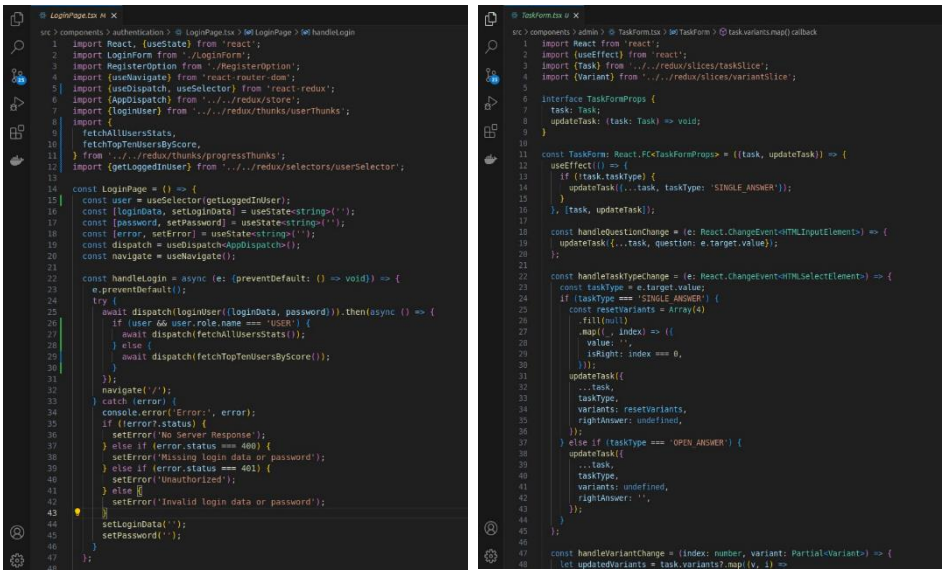


Рисунок 2.15, 2.16 Приклади файлів сторінки та форми з використанням React

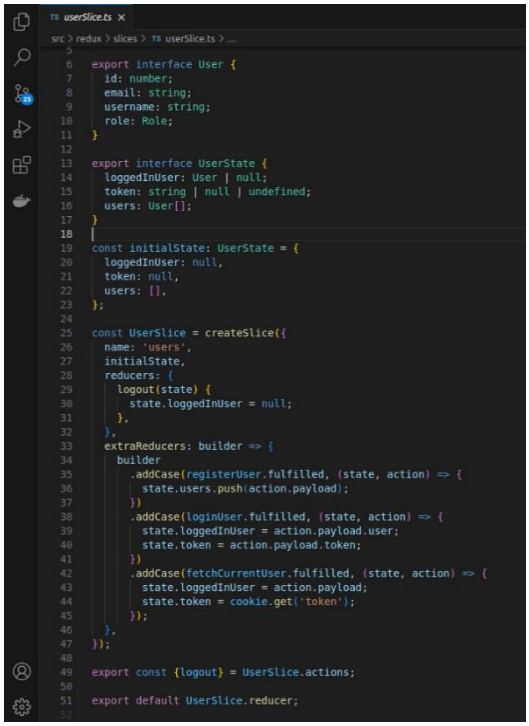
Проте особливу увагу пропоную звернути на так званий Redux Toolkit, який зробив процес розробки більш ефективним і простим.

Redux Toolkit – це офіційний інструментарій для спрощення роботи з Redux, який є популярною бібліотекою для управління станом в JavaScript-додатках, особливо в складних клієнтських додатках, таких як ті, що побудовані на React [21].

Ось декілька ключових переваг Redux Toolkit, на які варто звернути увагу:

- *Конфігурація*: Redux Toolkit зменшує кількість коду, необхідного для налаштування сховища Redux, завдяки вбудованим функціям та утилітам. Він включає такі функції, як *configureStore*, яка автоматично налаштовує проміжне програмне забезпечення і редактори, а також покращує роботу з Redux DevTools.
- *Створення редукторів та дій*: За допомогою *createSlice* можна одночасно створювати редуктори та пов'язані з ними дії. Це забезпечує більш

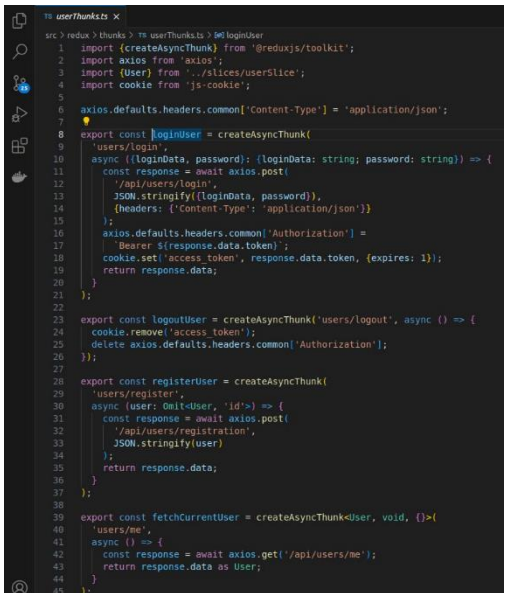
декларативний підхід до визначення редукторів, зменшуючи кількість шаблонного коду та збільшуючи читабельність (рис. 2.17).



```
export interface User {
  id: number;
  email: string;
  username: string;
  role: Role;
}

export interface UserState {
  loggedInUser: User | null;
  token: string | null | undefined;
  users: User[];
}

const initialState: UserState = {
  loggedInUser: null,
  token: null,
  users: [],
};

const UserSlice = createSlice({
  name: 'users',
  initialState,
  reducers: {
    logout(state) {
      state.loggedInUser = null;
    },
  },
  extraReducers: builder => {
    builder
      .addCase(registerUser.fulfilled, (state, action) => {
        state.users.push(action.payload);
      })
      .addCase(loginUser.fulfilled, (state, action) => {
        state.loggedInUser = action.payload.user;
        state.token = action.payload.token;
      })
      .addCase(fetchCurrentUser.fulfilled, (state, action) => {
        state.loggedInUser = action.payload;
        state.token = cookie.get('token');
      });
  },
});

export const {logout} = UserSlice.actions;

export default UserSlice.reducer;
```

Рисунок 2.17 Приклад Redux-редуктора на стороні фронтенду

- *Керування асинхронною логікою*: Redux Toolkit містить утиліту *createAsyncThunk*, яка допомагає легко обробляти асинхронні дії в Redux, використовуючи проміжне програмне забезпечення thunk для роботи з асинхронними операціями, такими як запити до API (рис. 2.18).



```
import {createAsyncThunk} from '@reduxjs/toolkit';
import axios from 'axios';
import {User} from '../slices/userSlice';
import cookie from 'js-cookie';

axios.defaults.headers.common['Content-Type'] = 'application/json';

export const loginUser = createAsyncThunk(
  'users/login',
  async ({loginData, password}: {loginData: string; password: string}) => {
    const response = await axios.post(
      '/api/users/login',
      JSON.stringify({loginData, password}),
      {headers: {'Content-Type': 'application/json'}}
    );
    axios.defaults.headers.common['Authorization'] =
      `Bearer ${response.data.token}`;
    cookie.set('access_token', response.data.token, {expires: 1});
    return response.data;
  }
);

export const logoutUser = createAsyncThunk('users/logout', async () => {
  cookie.remove('access_token');
  delete axios.defaults.headers.common['Authorization'];
});

export const registerUser = createAsyncThunk(
  'users/register',
  async (user: Omit<User, 'id'>) => {
    const response = await axios.post(
      '/api/users/registration',
      JSON.stringify(user)
    );
    return response.data;
  }
);

export const fetchCurrentUser = createAsyncThunk<User, void, {}>(
  'users/me',
  async () => {
    const response = await axios.get('/api/users/me');
    return response.data as User;
  }
);
```

Рисунок 2.18 Приклад використання createThunk на стороні фронтенду

- *Незмінність даних*: За замовчуванням використовується бібліотека Immer, що дозволяє писати код редукторів, який виглядає як мутаційний (тобто змінюючий дані напряму), але насправді він працює з іммутабельними даними без ризику непередбачених змін стану.
- *Типізація*: Для розробників, що використовують TypeScript, Redux Toolkit пропонує вдосконалену інтеграцію з TypeScript, полегшуючи типізацію стану, редукторів і дій.

Загалом, Redux Toolkit допомагає розробникам зменшити кількість повторюваного коду, забезпечує більш чистий та інтуїтивно зрозумілий спосіб керування станами у складних додатках і водночас підвищує продуктивність розробки завдяки вбудованим оптимізаціям та корисним утилітам.

Необхідно також відмітити надзвичайно важливий елемент, який є одним з найосновніших при роботі з нашиш вебзастосунком – React Router DOM. Це – бібліотека для маршрутизації в React-додатках, яка дозволяє ефективно керувати навігацією в односторінкових вебзастосунках. Вона надає компоненти та API для визначення маршрутів у додатку, дозволяючи пов'язати URL з конкретними компонентами React [22] (рис. 2.19).

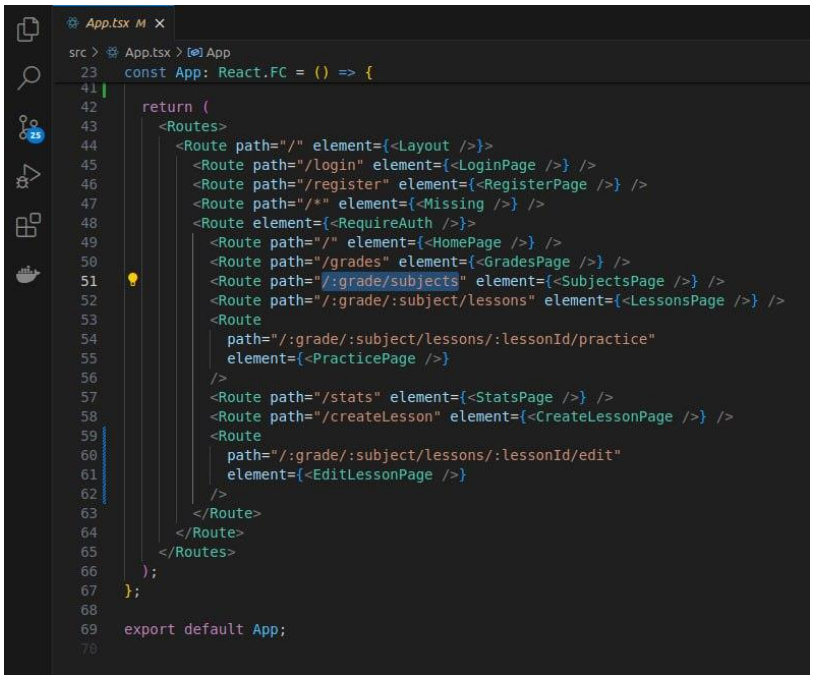


Рисунок 2.19 Використанння React Router DOM для платформи «Dali»

Як доказ ефективності нижче наведені основні аспекти та можливості React Router DOM:

- *Динамічна маршрутизація*: Визначає маршрути за допомогою компонентів <*Route*>, де кожен маршрут вказує на певний компонент.
- *Вкладені маршрути*: Підтримує вкладену маршрутизацію, що дозволяє створювати складні ієрархії маршрутів у застосунку.
- *Навігація*: Включає компоненти, такі як <*Link*>, <*NavLink*>, та <*Redirect*> для управління переходами і перенаправленнями в рамках додатка.
- *Програмне управління маршрутами*: Дозволяє керувати історією навігації за допомогою програмного API, надаючи такі функції, як перехід назад або вперед.
- *Параметри пошуку*: Підтримує параметри маршруту та запити, дозволяючи переносити дані між сторінками.

Отже, ми розглянути структурну схему нашого навчального онлайн-застосунку та виділили найосновніші та найцікавіші моменти в програному коді, що надають нам усі необхідні елементи для реалізації потрібного для «Dali» функціоналу та відповідають поставленим вимогам.

2.3 Структурно-фунціональні діаграми

Даний підрозділ уже присвячений важливим діаграмам, які стануть у нагоді в кращому розумінні роботи розробленої мінімалістичної освітньої платформи. Найпершим чином продемонструємо діаграму варіантів використання нашого застосунку (рис 2.20).

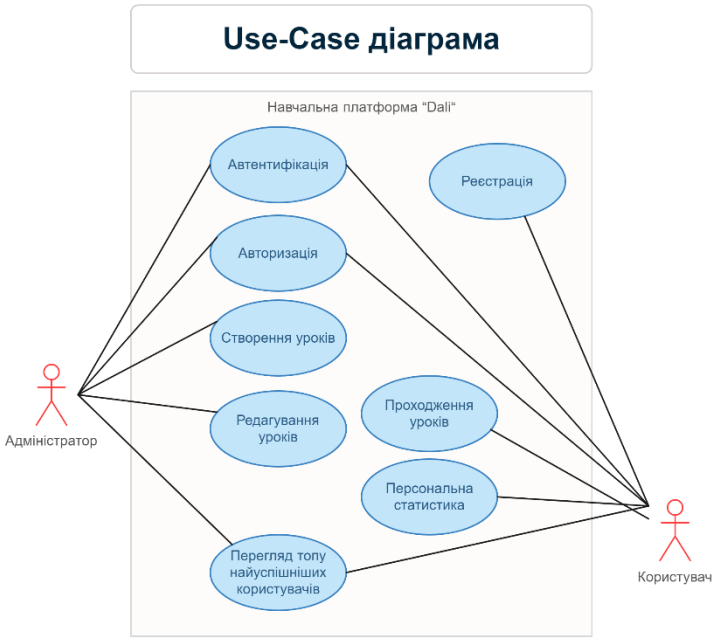


Рисунок 2.20 Діаграма варіантів використання

Ця діаграма випадків використання (use-case diagram) чітко показує функціональні можливості та взаємодію користувачів із навчальною платформою «Dali». Діаграма містить дві ролі: Адміністратор та Користувач, які взаємодіють з різними частинами системи. Адміністратор має доступ до наступних функцій: автентифікація, авторизація, створення уроків, їх редагування, проходження уроків і перегляд списку найуспішніших користувачів. Ці можливості дозволяють адміністратору керувати контентом та моніторити активність на платформі. Зі свого боку, Користувач може зареєструватися, автентифікуватися й авторизуватися, проходити уроки та переглядати власну персональну статистику, що забезпечує динамічний та інтерактивний досвід навчання на порталі. Зокрема, підтримка власної статистики сприяє мотивації користувачів та особистісному розвитку через змагальні аспекти платформи.

Наступною до розгляду стане діаграма переходу станів розробленого застосунку (рис. 2.21).

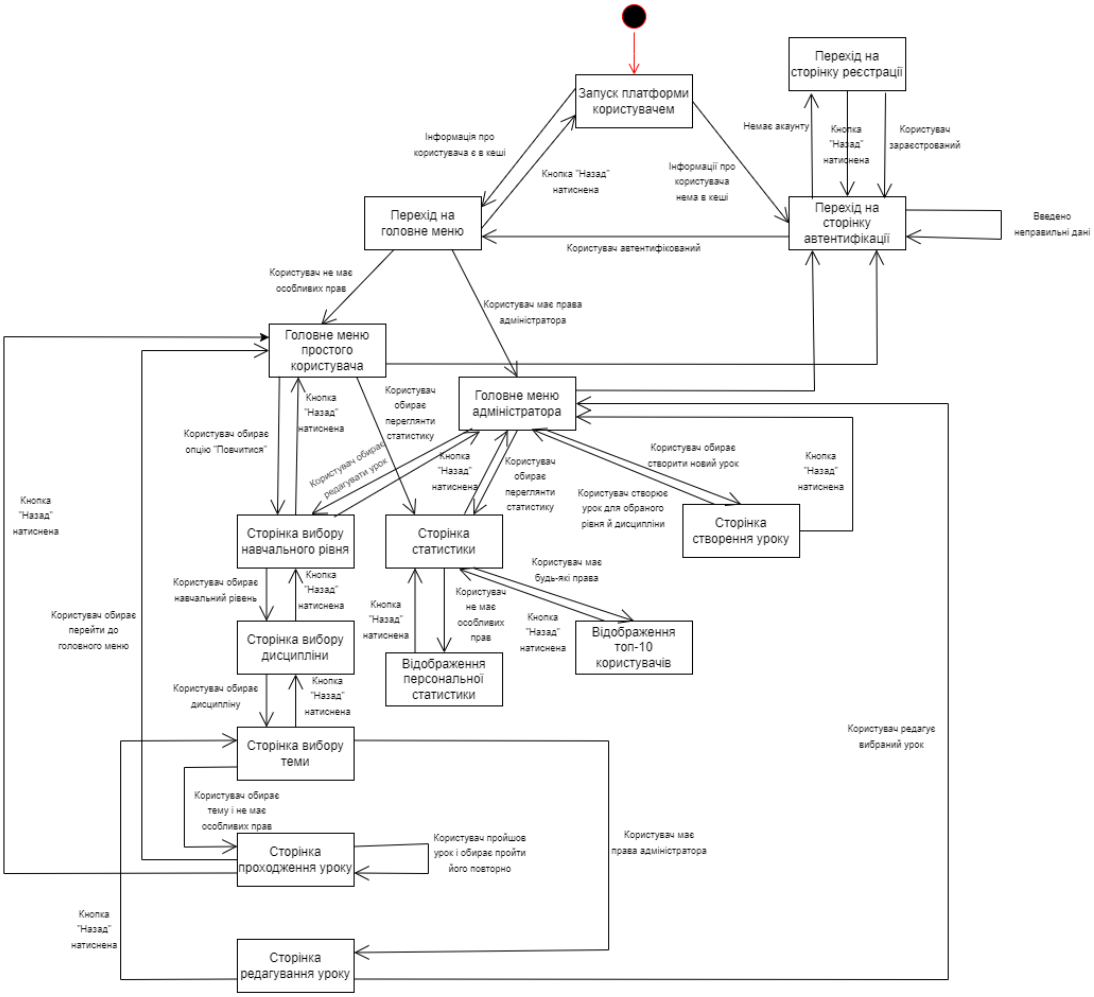


Рисунок 2.21 Діаграм переходу станів навчальної платформи «Dali»

Дана діаграма містить чимало розгалужень, оскільки функціонал нашої навчальної платформи відрізняється для адміністратора й звичайного користувача. До прикладу, адміністратор не може проходити розроблені ним уроки, натомість простий користувач не має належних дозволів для їхнього створення та редагування. Тому маємо змогу чітко простежити можливі варіанти розвитку подій і кількість можливих станів під час роботи з розробленим застосунком попри можливу складність і навантаженість наведеної діаграми.

2.4 Інтерфейс застосунку

Раніше ми встигли зазначити, що найосновнішим кольором візуалу розробленої навчальної платформи є синій, адже він має найбільш продуктивний

ефект з точки зору підсилення бажання до роботи та праці. Цей колір стане постійним кольором заднього плану нашого вебресурсу.

Пропоную розглянути інтерфейс почергово з точки зору двох визначених ролей застосунк: адміністратора та звичайного користувача, тобто учня. Під час найпершого відкриття головної сторінки ми на неї зайдемо не одразу, оскільки ми ще ніяк не автентифіковані, та й можливо, що не зареєстровані (рис. 2.22).

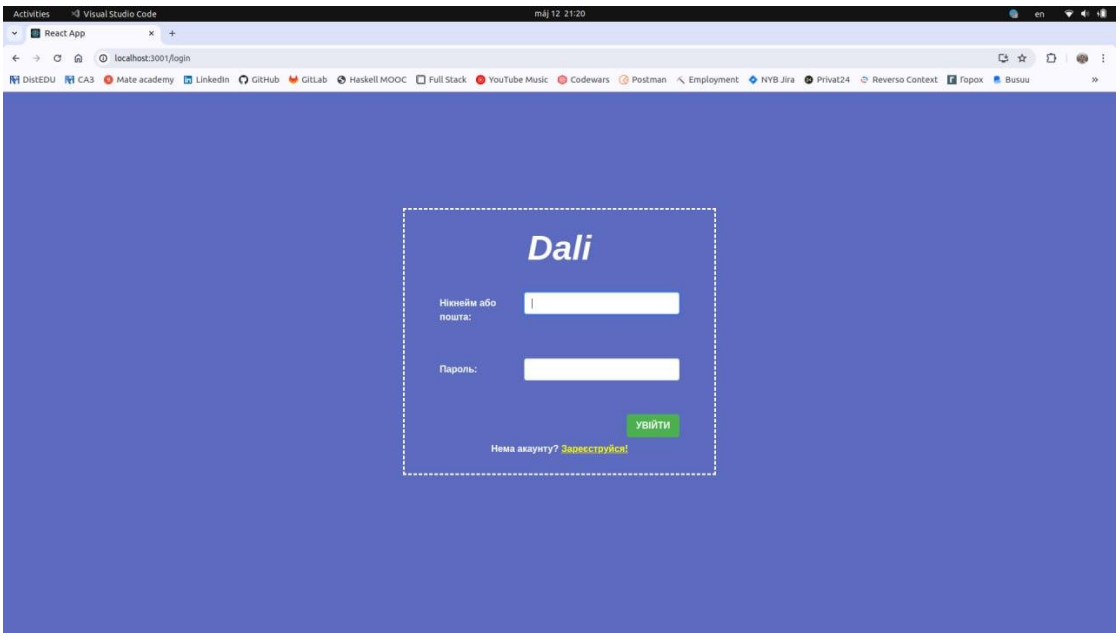


Рисунок 2.22 Скриншот сторінки автентифікації на платформі «Dali»

Тому спершу варто зарєструвати нового користувача на сторінці реєстрації, попередньо натиснувши на посилання «Зареєструйся!» (рис. 2.23)

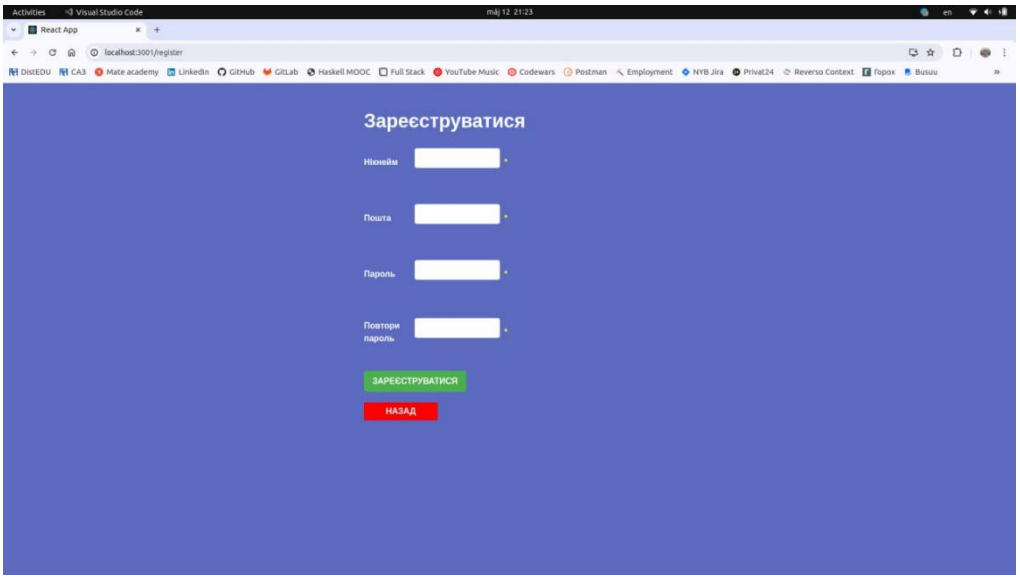


Рисунок 2.23 Скриншот сторінки реєстрації

Якщо ми почнемо вводити некоректні для нашої системи дані, видаватиме повідомлення про відповідні помилки (рис. 2.24).

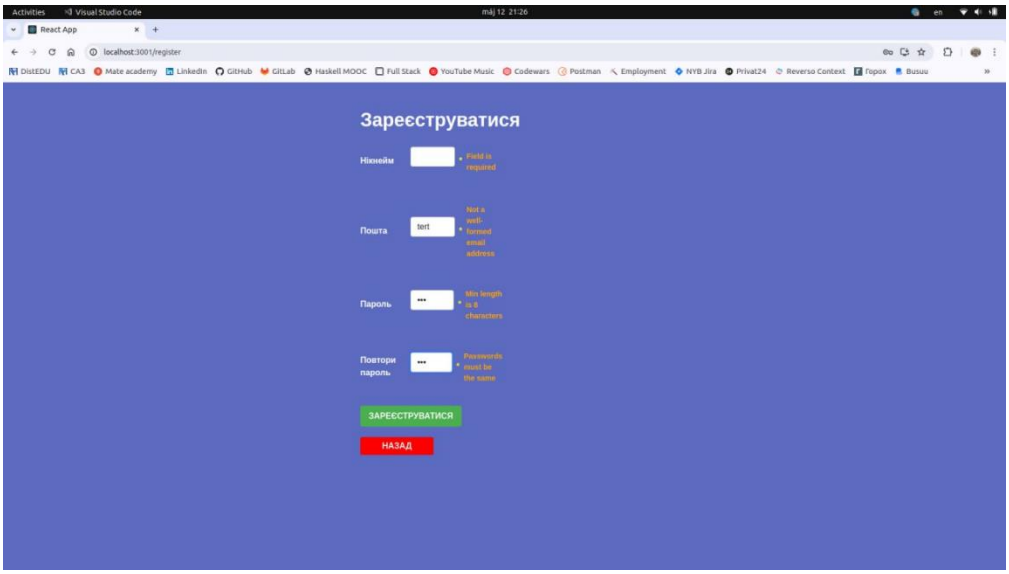


Рисунок 2.24 Виведення повідомлень про помилки вводу в момент самого введення

Тож введімо такі дані, які задовольнять визначені для реєстрації поля (рис. 2.25).

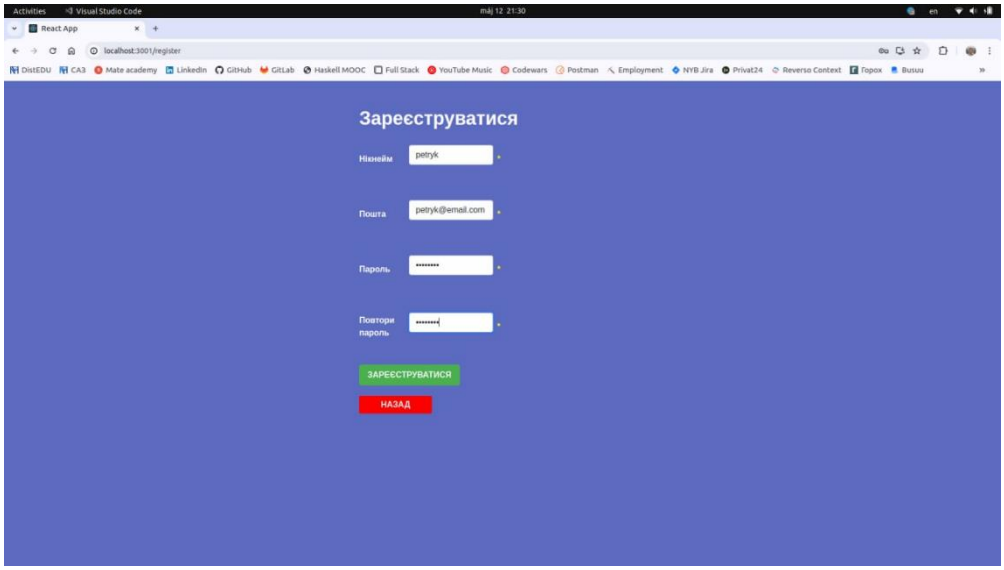


Рисунок 2.25 Скриншот правильно введених даних для реєстрації

Після натискання на кнопку «ЗАРЕЄСТРУВАТИСЯ» перносимося назад на сторінку з полями введення даних для автентифікації, тож цього разу маємо що в них зазначити (рис. 2.26).

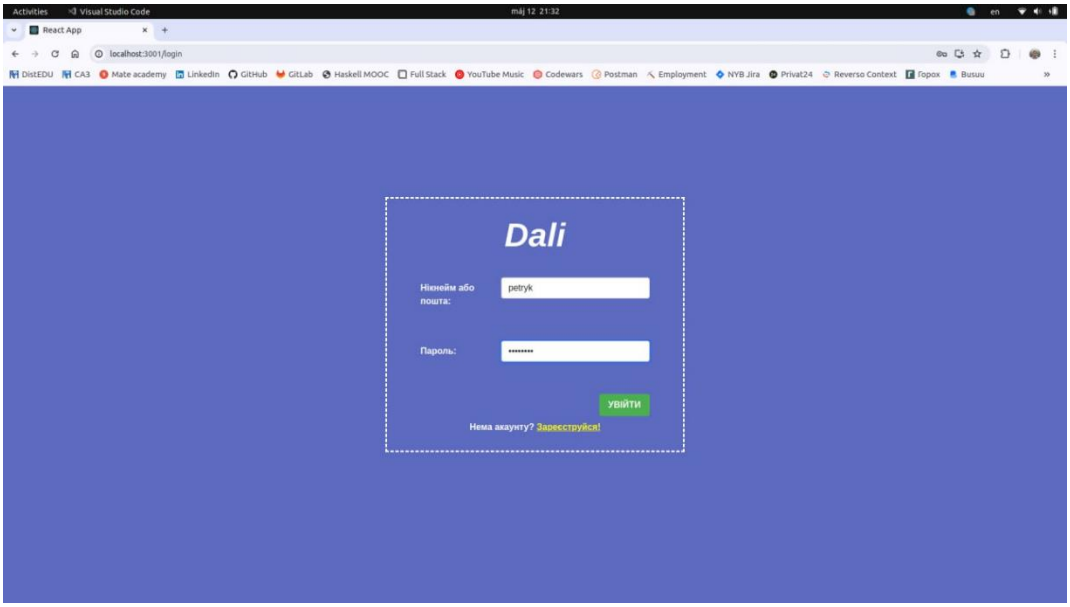


Рисунок 2.26 Дані готові для входження в систему

Одразу ж по натисканню на кнопку «УВІЙТИ» опиняємося на головній сторінці користувача зі звичайними правами доступу (рис. 2.27).

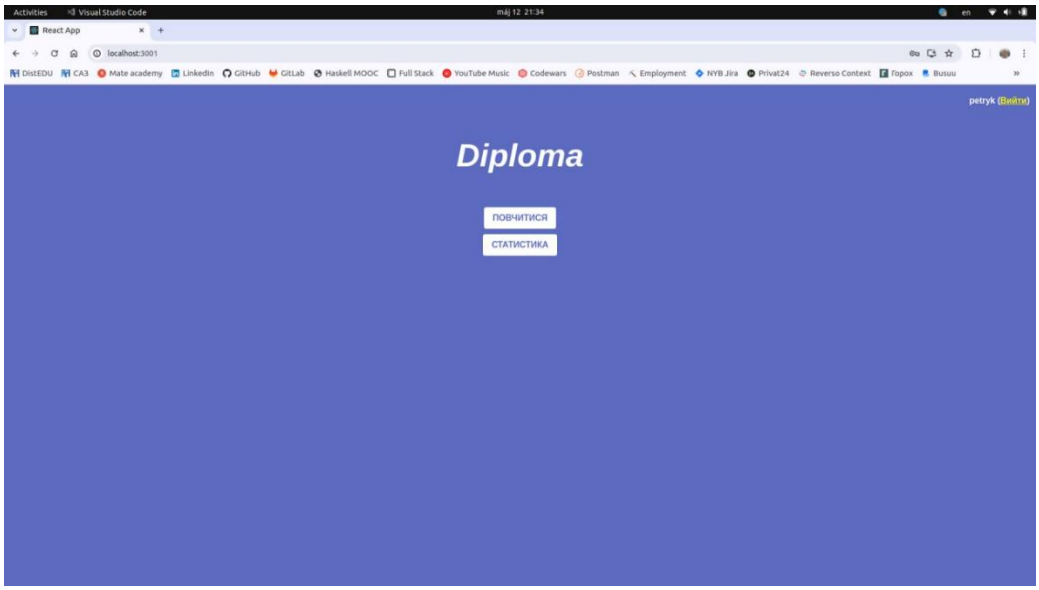


Рисунок 2.27 Скриншот головної сторінки для звичайного користувача

Наразі хочемо спробувати пройти певний урок з вибраної дисципліни, тож натискаємо на кнопку «ПОВЧИТИСЯ» й переходимо до сторінки з вибором навчального рівня (рис. 2.28).

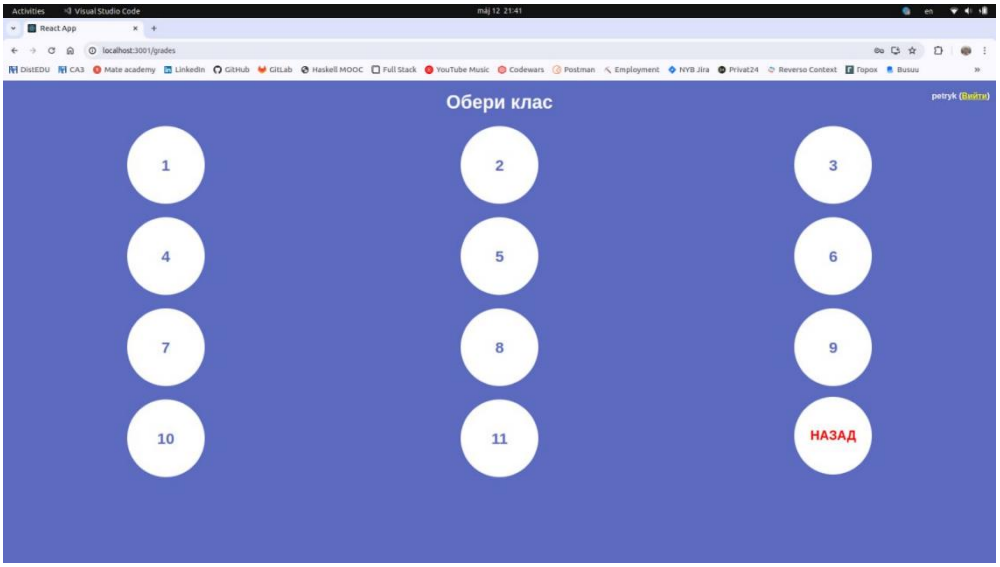


Рисунок 2.28 Сторінка вибору навчального рівня

Оберімо 7 клас і перемістімося до сторінки з можливістю обрати цікавий нам предмет чи навчальний напрямок (рис. 2.29).

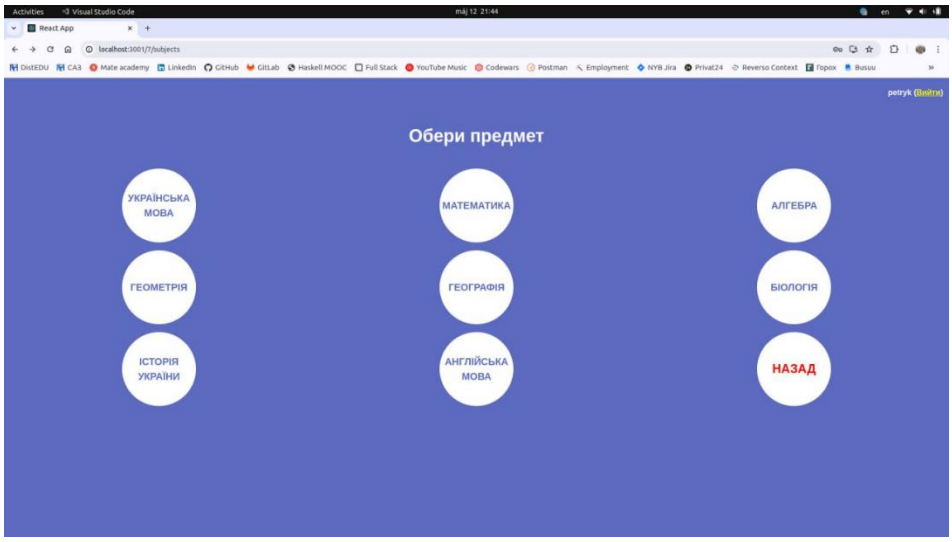


Рисунок 2.29 Сторінка вибору навчального напрямку

На цей раз наш вибір буде за алгеброю, після чого опиняємося перед варіантами занять, які можна пройти (рис. 2.30).

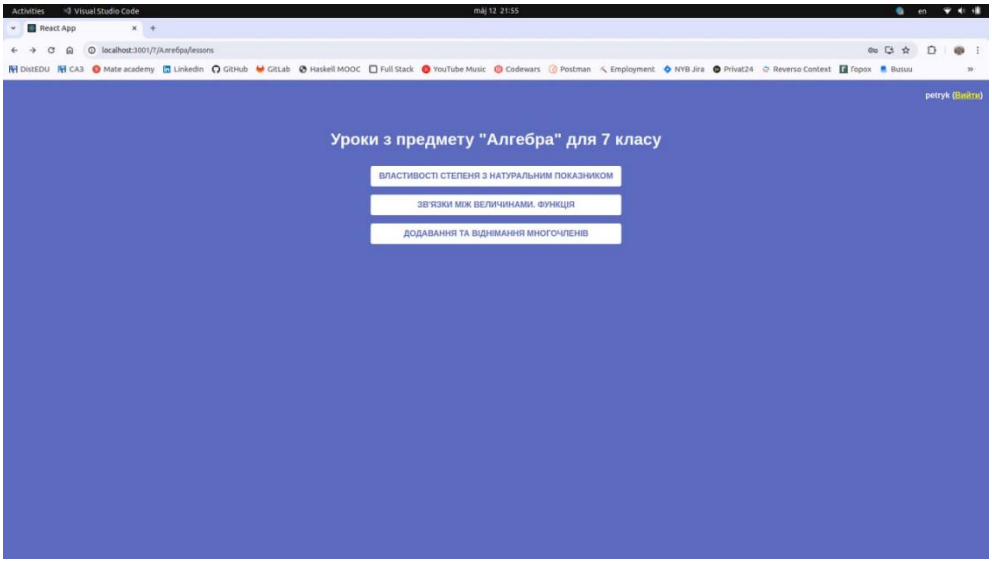


Рисунок 2.30 Сторінка вибору уроків для проходження

Можемо розібратися з темою «Зв’язки між величинами. Функція», натиснувши на відповідну назву заняття й почавши при цьому виконання одного з визначених завдань (рис. 2.31).

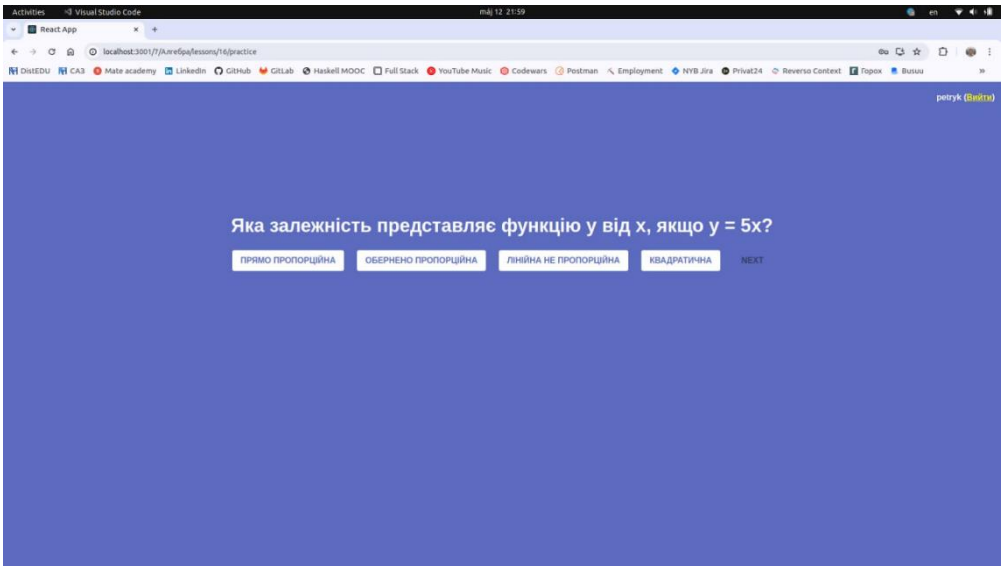


Рисунок 2.31 Виконання завдання з вибраної користувачем теми

Але, як бачимо, неможливо перейти до наступного питання, не обравши жодний варіант відповіді, тому виберемо тут той варіант, який на нашу думку можна вважати правильним (рис. 2.32).

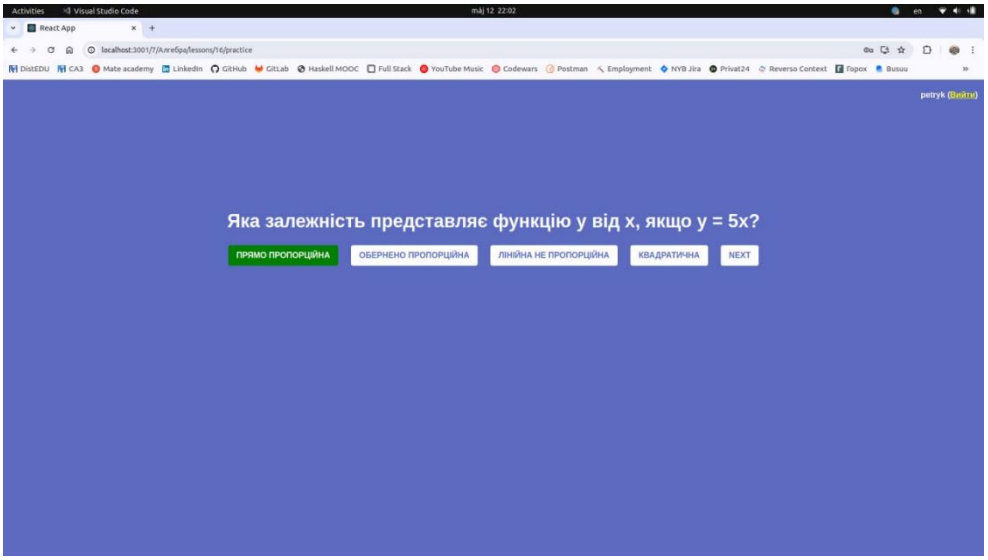


Рисунок 2.32 Здійснення вибору одного з запропонованих варіантів

Наступне питання в уроці передбачає самостійний ввід правильної на наш погляд відповіді (рис. 2.33).

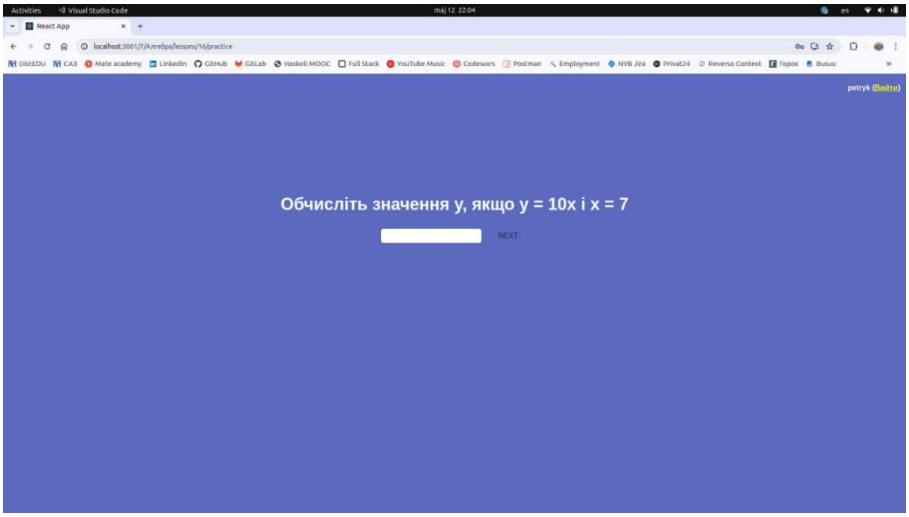


Рисунок 2.33 Завдання з необхідністю введення відповіді

Аналогічно до завдання з варіантами відповідей, ми не маємо змоги перейти до наступного питання, не записавши принаймні якусь відповідь (рис. 2.34).

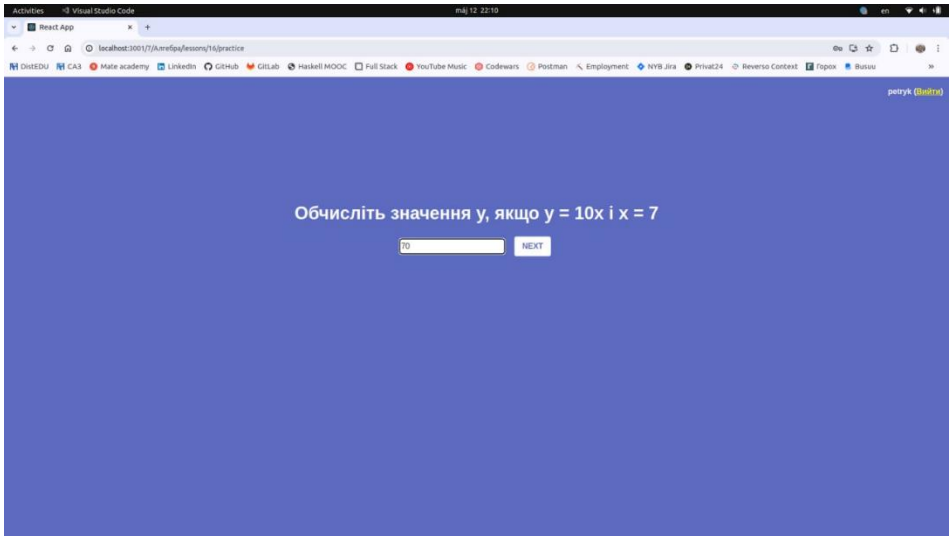


Рисунок 2.34 Запис вілповіді у належне поле

Після проходження всіх 10 питань по темі вибраного уроку отримаємо хороший результат і відповідну сторінку в браузері (рис. 2.35).

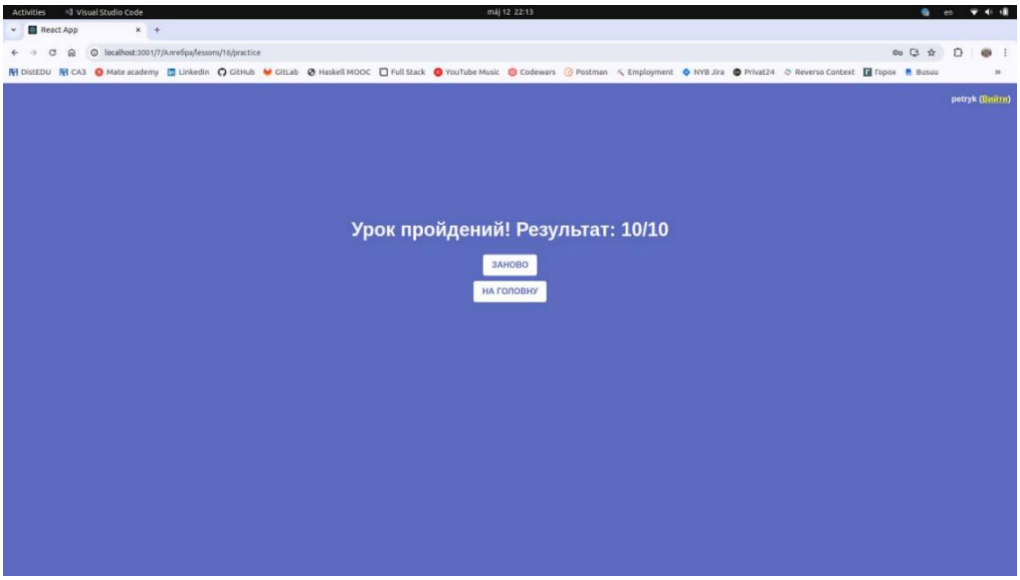


Рисунок 2.35 Сторінка завершення проходження уроку

Заради інтересу повернемося на головне меню та перейдемо до вікна зі статистикою, щоб оцінити наші початкові успіхи (рис. 2.36).

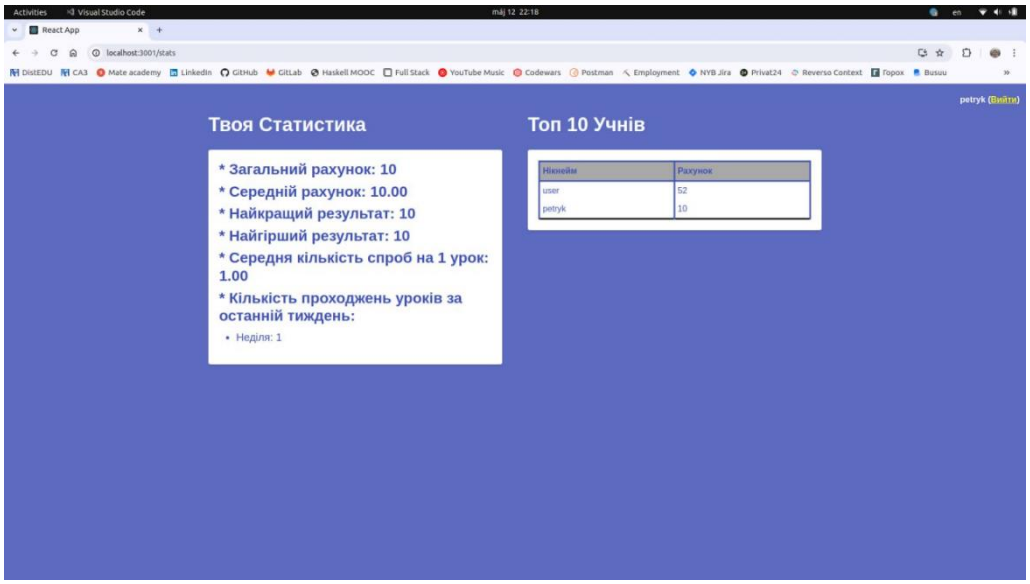


Рисунок 2.36 Сторінка зі статистикою та топом користувачів

Ми дуже добре впоралися як для першого уроку, але задля цікавості пропоную розглянути також статистику від користувача user (рис. 2.37).

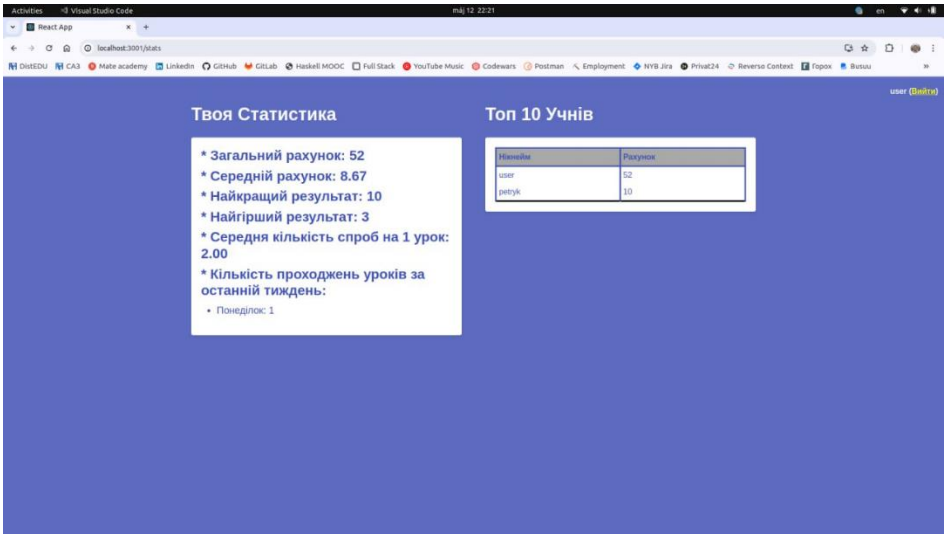


Рисунок 2.37 Сторінка зі статистикою та топом користувачів

Якщо ж ми плануємо розглянути функціонал з боку адміністратора, то нам варто вийти та зайти вже з відповідними правами доступу (рис. 2.38).

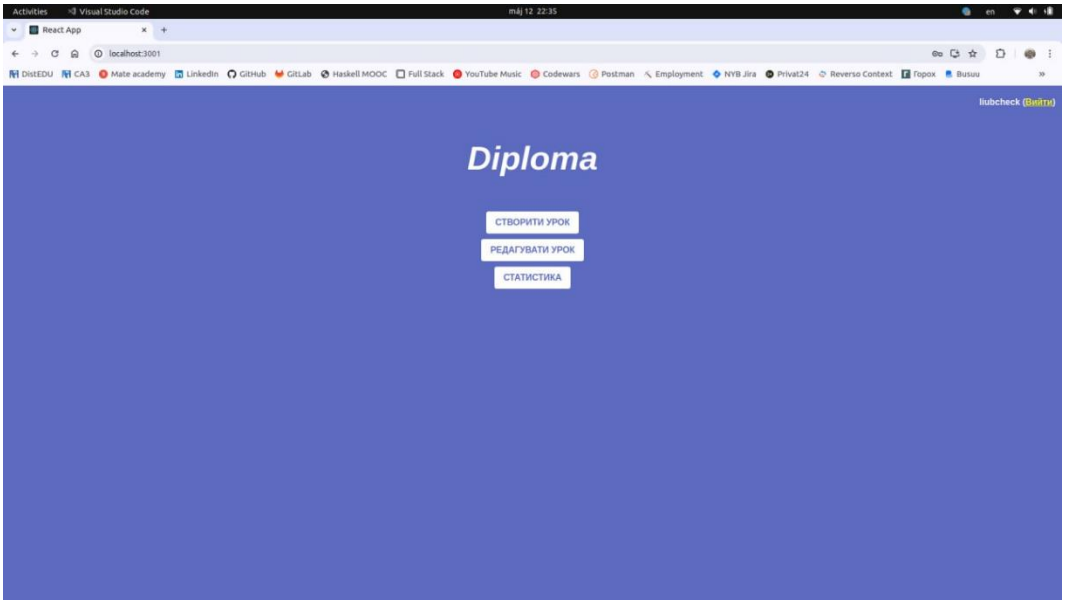


Рисунок 2.38 Скриншот головної сторінки для адміністратора

Використаємо дану нам нагоду та перейдемо до сторінки для створення уроку. Створимо додатковий урок з алгебри для учнів 7 класу з теми «Сума й різниця кубів двох виразів» (рис. 2.39).

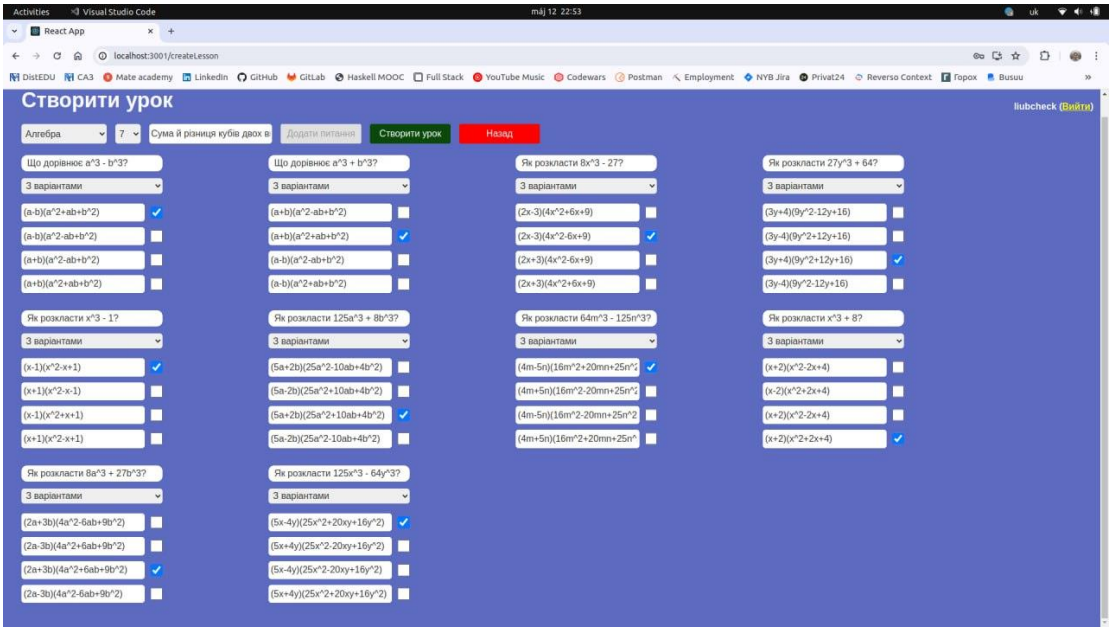


Рисунок 2.39 Створення уроку для платформи «Dali»

Урок успішно створений, що ми легко можемо перевірити, зайшовши на нашу навчальну платформу в якості звичайного користувача й перейшовши до тем алгебри 7 класу (рис. 2.40).

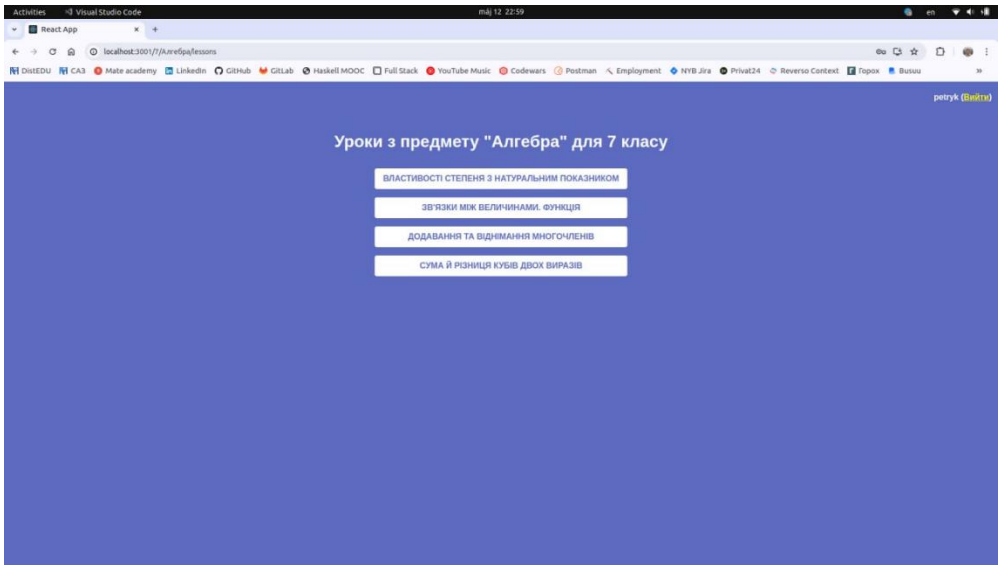


Рисунок 2.40 Поява новододаного заняття

Але в нас як у адміністратора порталу виникла потреби виправити одне завдання в новоствореному уроці на питання з відкритою відповіддю, тож скористаймося сторінкою редагування завдання, перед цим обравши 7 клас, алгебру та тему «Сума й різниця кубів двох виразів» (рис. 2.41).

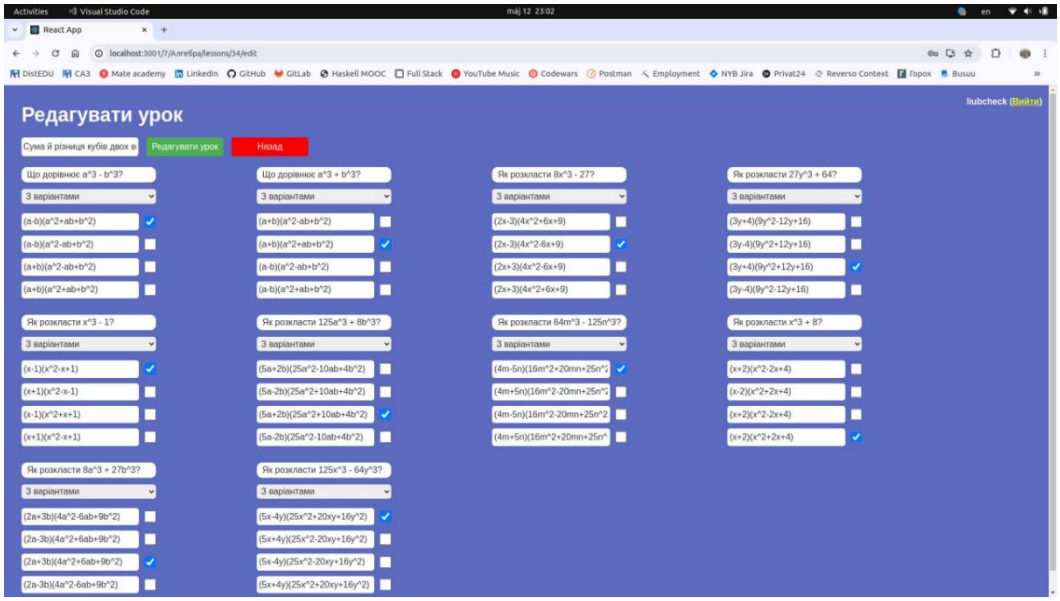


Рисунок 2.41 Сторінка редагування вибраного уроку

Зараз змінимо останнє питання на певне інше питання з відкритою формою відповіді (рис. 2.42).

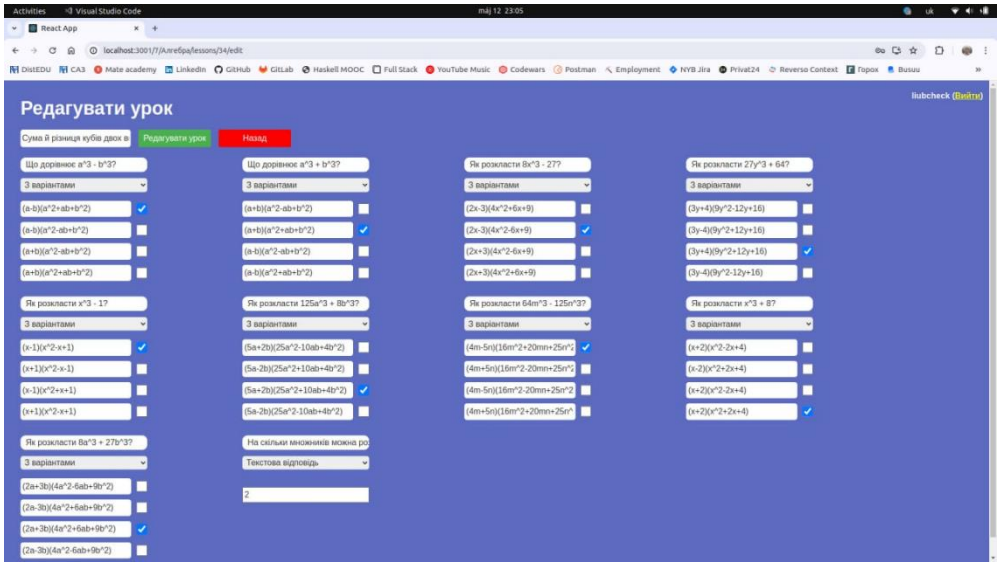


Рисунок 2.42 Відредаговане останнє завдання

Тепер просто збережемо наші оновлення, що стосуються вибраного уроку. Таким чином ми без жодних проблем і перешкод відредагували потрібне нам питання, змінивши його формат відповіді.

2.5 Висновки до розділу 2

У даному розділі було висвітлено основні аспекти розробки освітньої платформи «Dali», починаючи від ідейної концепції до показу готового інтерфейсу навчального порталу.

Було розглянуто концептуальні й технічні аспекти розробки платформи, зокрема її призначення чи основні функціональні можливості. Разом із цим наведені приклади діаграм варіантів використання та переходу станів, що допомагають детальніше зрозуміти характер роботи розробленого продукту.

Знімки ж екрану, що прикріплені в цьому розділі, демонструють візуальну частину, доступну користувачам, і дають уявлення про можливе повсякденне застосування навчальної платформи охочими здобувачами освіти.

ВИСНОВОК ПО РОБОТІ

У дослідженні було зосереджено увагу на аналізі різних освітніх платформ, які надають можливості для факультативного вивчення предметів. Виявлено ключові переваги та недоліки існуючих рішень, які стали основою для розробки власної навчальної платформи, адаптованої під потреби українських користувачів. Метою проєкту «Dali» стало створення зручної, інтуїтивної й доступної навчально\ платформи, яка здатна влучно доповнити українське освітнє середовище й удосконалити його в подальшому.

Було розглянуто та впроваджено відповідні якісні рішення в питаннях програмування для оптимізації роботи навчальної платформи, забезпечення його стабільності та надійності. Реалізовано необхідний бекенд- і фронтенд-функціонал, що відповідають сучасним трендам і тенденціям комерційної розробки та які відкриті до можливих подальших змін, доповнень і покращень.

Завдяки розробленій навчальній платформі, користувачі мають змогу не тільки вибрати предмети й напрямки за власним бажанням, але й відстежувати власний прогрес, підтримуючи таким чином навчально-змагальний дух. Це створює повніший навчальний досвід, що сприяє кращому засвоєнню матеріалу та більшому вивченню нового і цікавого.

Розроблена платформа є відкритою для майбутнього розвитку, оскільки ще існують певні програмні рішення, які варті подальшого впровадження. Зокрема вдалим і перспективним планом на майбутнє стане додавання нової ролі – вчителя – задля подальшого ефективнішого й швидшого додавання нових навчальних матеріалів на платформу «Dali»; також цікавим для реалізації задумом є можливість додавання інших користувачів собі в друзі задля комунікації в межах розробленої навчальної платформи, обміну новими знаннями та більш змагального характеру проходження матеріалів.

ДЖЕРЕЛА

[1] Thompson-Schill, S. L., D'Esposito, M., & Kan, I. P. (1999). Effects of repetition and competition on activity in left prefrontal cortex during word generation. Neuron, 23(3), 513-522. Доступно: https://www.cell.com/neuron/pdf/S0896-6273(00)80804-1.pdf

[2] Фіялка, С. (2022). Оцінка впливу війни на видавничу діяльність та наукові інтереси українських науковців. Knowledge and Performance Management, 1(1), Доступно: https://www.businessperspectives.org/images/pdf/applications/publishing/templates/article/assets/16896/KPM_2022_01_Fiialka.pdf

[3] https://schooltogo.online/

[4] https://zno.osvita.ua/

[5] https://learning.ua/

[6] https://naurok.com.ua/

[7] https://www.miyklas.com.ua/

[8] https://www.volynpost.com/news/161136-iak-rizni-kolory-dozvoliayut-pidvyschyty-produktyvnist-ofisnoi-praci

[9] https://www.colorpsychology.org/blue/

[10] https://www.tiobe.com/tiobe-index/

[11] https://smmurtaza17.medium.com/programming-language-ranking-top-choices-for-2024-0bde5ea00e68

[12] https://medium.com/@IvanZmerzlyi/microservices-architecture-461687045b3d

[13] https://www.baeldung.com/spring-cloud-netflix-eureka

[14] https://www.baeldung.com/spring-cloud-gateway

[15] https://www.ibm.com/topics/java-spring-boot

[16] https://martinfowler.com/articles/injection.html#InversionOfControl

[17] https://www.baeldung.com/spring-cloud-openfeign

[18] https://www.liquibase.com/

[19] https://www.typescriptlang.org/

[20] https://react.dev/

[21] https://redux-toolkit.js.org/

[22] https://luqmanshaban.medium.com/react-router-a-step-by-step-guide-4c5ec964d2e9