УДК 004.42

А. В. Смолінський
О. В. Войцеховська

Кросплатформна система управління навчальним процесом

Вінницький національний технічний університет

Анотація

Розроблено кросплатформну систему управління навчальним процесом, яка складається із серверної частини, адміністративної панелі та клієнтської частини для студента. Запропонована система дає змогу керувати студентами, уроками та курсами в межах єдиного програмного середовища. Особливу увагу приділено реалізації модуля уроків, у якому вміст зберігається у вигляді JSON-структури з набором блоків різного типу, а також модуля курсів, де передбачено впорядковане додавання уроків і зміну їх послідовності. Для реалізації системи використано NestJS, React, PostgreSQL та JWT-автентифікацію.

Ключові слова: *система управління навчальним процесом, LMS, NestJS, React, PostgreSQL, JWT, навчальний курс, валідація даних DTO.*

Abstract

A cross-platform learning management system has been developed, consisting of a backend, an administrative panel, and a client part for the student. The proposed system makes it possible to manage students, lessons, and courses within a single software environment. Particular attention is paid to the implementation of the lessons module, in which content is stored in the form of a JSON structure with a set of blocks of different types, as well as to the courses module, where ordered addition of lessons and changes in their sequence are provided. NestJS, React, PostgreSQL, and JWT authentication were used to implement the system.

Keywords: *learning management system, LMS, NestJS, React, PostgreSQL, JWT, educational course, DTO data validation.*

Вступ

Сьогодні цифрові інструменти дедалі активніше використовуються в освітньому процесі. Це стосується як дистанційного навчання, так і внутрішньої організації навчальних матеріалів, доступу до них, розподілу курсів між студентами та загального адміністрування навчального середовища. Якщо навчальний контент зберігається у розрізнених файлах або підтримується вручну, це швидко створює проблеми: важко оновлювати матеріали, контролювати доступ, підтримувати єдину структуру та розвивати систему далі.

Тому розробка кросплатформної системи управління навчальним процесом, у якій адміністративна панель, серверна частина та студентський застосунок працюють узгоджено, є актуальною. Метою роботи є створення такої системи для керування студентами, уроками та курсами з безпечним доступом до даних і зручною архітектурою для подальшого розвитку, яка дасть можливість централізовано керувати навчальним процесом.

Проектування архітектури системи

Розроблена система містить три основні складові: серверну частину на NestJS, адміністративний вебінтерфейс та клієнтську частину як окремий студентський фронтенд. Адміністративна частина вирішує задачі створення та редагування даних, а студентська – задачі перегляду доступного навчального контенту, що дозволяє не змішувати різні сценарії використання і підвищити контроль доступу користувачів. Схему архітектури розробленої кросплатформної системи управління навчальним процесом подано на рис. 1.

Для реалізації системи було обрано сучасний технологічний стек. Серверна частина побудована на NestJS [1], що забезпечує модульність, наявність dependency injection, guards, валідацію даних Data Transfer Object (DTO) та зручну організацію бізнес-логіки. База даних PostgreSQL використовується для зберігання структурованих даних і підходить для поєднання класичних таблиць із JSON-полями [2].

Клієнтська частина реалізована на React із використанням TypeScript [3]. Це дозволяє отримати типобезпечний інтерфейс та уникати значної кількості помилок ще на етапі розробки. Для роботи із серверним станом використано React-Query, що спрощує отримання, оновлення та кешування даних. Окремо використано сервісний підхід до побудови фронтенду, коли запити до API винесені в окремі сервіси, а компоненти залишаються більш “чистими”.

Реалізація серверної частини

Серверну частину побудовано за модульним принципом. Для кожного окремого напряму виділено власний модуль: автентифікація та авторизація, студенти, уроки, курси. Кожен із них містить контролери, сервіси, DTO та типізацію даних. Такий підхід спрощує підтримку коду, оскільки логіка розділена на зрозумілі частини, а внесення змін до одного модуля мінімально впливає на інші.

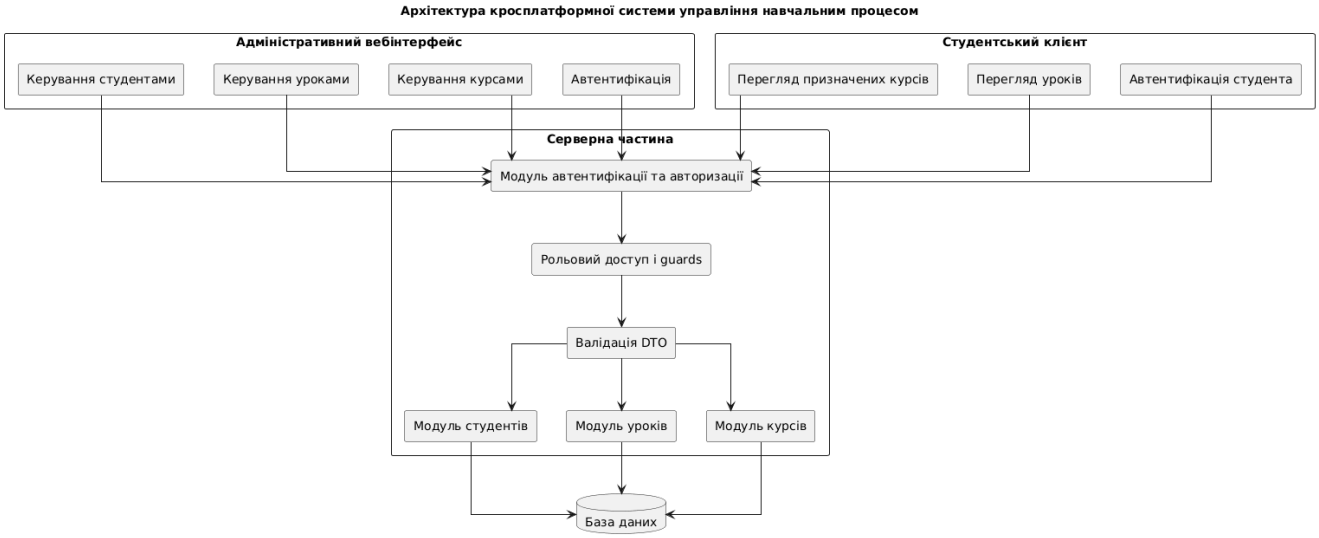

Рисунок 1 – Архітектура системи

Одним із основних елементів розробленої системи є підсистема автентифікації та авторизації. У проєкті використано JWT-токени та cookie-based підхід для зберігання сесії. Вибір такого рішення обумовлений тим, що зберігання токенів у cookies є більш безпечним підходом, ніж використання localStorage та таке рішення зручно поєднується з розмежуванням доступу за ролями.

На сервері перевірка доступу виконується за допомогою guards. Вони відповідають за перевірку токена, ролі користувача та права на виконання конкретної дії. У результаті адміністративний функціонал доступний лише адміністратору, а студентський — лише користувачу з відповідною роллю.

Модуль керування студентами призначений для керування користувачами, які навчаються в запропонованій системі. У ньому реалізовано створення, редагування, видалення, перегляд інформації про окремого студента та отримання списку студентів. Під час створення студента система формує обліковий запис користувача, створює профіль студента та генерує для нього тимчасовий пароль.

Виконується перевірка унікальності email-адреси та структури типів даних, що використовуються і на фронтенді, і на бекенді. Для клієнтської частини застосовано підхід із сервісами та окремими hooks для роботи з API, що дозволило структурувати код та відокремити логіку запитів від компонентів інтерфейсу.

Модуль уроків реалізовано з використанням бібліотек TinyMCE [4], react-chartjs-2 [5] та recharts [6], у вигляді текстового редактору, що відкривається у окремому вікні кросплатформної системи. При цьому урок не зберігається як звичайний текстовий запис, а для нього використовується складна структура з полями title, comment та payload, яка дозволяє повноцінне створення уроків. Поле payload містить JSON-об’єкт, у якому описується вміст уроку. Це дозволяє зробити урок гнучким, тобто він може складатися з текстових блоків, таблиць, кругових діаграм, лінійних графіків, вертикальних і горизонтальних діаграм, а також бульбашкових діагрім для аналізу числових залежностей. Використання різного типу діаграм дає можливість наочно показувати порівняння, динаміку, співвідношення, що полегшує сприйняття складної інформації студентами.

Отже урок, розроблений в даній системі, є набором блоків із певним порядком відображення, що набагато зручніше, ніж жорстко зводити все до одного текстового поля або створювати окрему таблицю в базі даних під кожен тип блоку. JSON-структура дозволила об’єднати різні типи контенту в межах одного уроку, забезпечити збереження порядку, споростити рендеринг на клієнтський частині та подальшу зміну без перебудови бази даних.

Реалізація адміністративної частини

Для створення і редагування уроків в адміністративній панелі реалізовано окремий конструктор контенту. У ньому користувач може по черзі додавати потрібні блоки, редагувати їх та змінювати порядок. Порядок блоків задається масивом і може змінюватися з допомогою методу drag-and-drop.

Такий підхід є зручним не лише для адміністратора, а й для самої архітектури системи. Усі дані уроку збираються на клієнтській частині системи в єдину структуру, після чого зберігаються на сервері як JSON-вміст. При цьому для перегляду уроку використовуються ті самі компоненти відображення, що і в режимі редагування, але вже без можливості зміни. Це дало можливість повторно використовувати логіку рендерингу й не дублювати код.

В розробленому модулі курсів курс містить назву групи, назву курсу, коментар, дату початку та набір уроків, які входять до нього. Для курсів передбачено створення, редагування, видалення, перегляд деталей, а також список із пошуком і сортуванням за основними полями.

В даній системі реалізовано зв'язок між модулями курсів та уроків: курс не просто містить перелік уроків, а зберігає їх у певному порядку. На стороні адміністратора для цього передбачається додавання уроків через пошуковий компонент select та зміна їх послідовності за допомогою drag-and-drop. В такому випадку один і той самий урок можна повторно використовувати в різних курсах, не створюючи його заново. Якщо урок редагується, зміни стають актуальними в усіх курсах, де він використовується. Крім того, це створює можливості для подальшого розвитку системи, наприклад для відстеження прогресу, групування студентів, побудови навчальних програм тощо.

Реалізація клієнтської частини

На студентській стороні реалізується спрощений функціонал. Студенту не потрібен доступ до створення або редагування курсів і уроків, тому ця частина системи містить два сценарії: перегляд списку доступних курсів і перегляд деталей конкретного курсу разом із вкладеними уроками.

Доступ до курсів має бути безпечним, тому на сервері ідентифікатор студента витягується з JWT-токена, після чого повертаються лише ті дані, які дійсно належать цьому користувачу. Таким чином, студент не бачить сторонній контент і працює тільки з тими курсами, які йому призначені.

Висновки

1. Розроблено архітектуру та програмну реалізацію кросплатформної системи управління навчальним процесом. Система охоплює керування студентами, уроками та курсами, підтримує безпечний доступ через JWT-автентифікацію та використовує модульний підхід як на сервері, так і на фронтенді. Отримано цілісну систему, в якій дані зберігаються централізовано, обробляються через єдині серверні контракти та коректно відображаються у двох незалежних клієнтських частинах.

2. Реалізовано функціональні модулі системи та забезпечено їхню взаємодію. На серверній стороні створено модулі автенифікації й авторизації, керування студентами, уроками та курсами, також реалізовано валідацію даних за допомогою DTO, рольовий доступ і захист маршрутів за допомогою guards. На адміністративному вебінтерфейсі реалізовано створення, редагування, видалення та перегляд сутностей. На студентському фронтенді реалізовано доступ лише до призначених студенту матеріалів.

3. Розроблена кросплатформна система управління навчальним процесом показує, що поєднання NestJS, React, PostgreSQL, JWT-автентифікації та конструктора уроків на основі JSON для побудови LMS-платформи, забезпечує просте адміністрування, гнучке формування навчального контенту, спрощує його збереження та відображення, а також дає можливість розширювати функціональність системи без суттєвої зміни базової архітектури.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. NestJS Documentation [Електронний ресурс]. – Режим доступу: https://docs.nestjs.com/
2. PostgreSQL Documentation [Електронний ресурс]. – Режим доступу: https://www.postgresql.org/docs/.
3. React Documentation [Електронний ресурс]. – Режим доступу: https://react.dev/.
4. TinyMCE 8 Documentation[Електронний ресурс]. – Режим доступу: https://www.tiny.cloud/docs/tinymce/latest/.
5. React-chartjs-2 Documentation [Електронний ресурс]. – Режим доступу: https://react-chartjs-2.js.org/.
6. Recharts Documentation [Електронний ресурс]. – Режим доступу: https://recharts.github.io/.

Смолінський Андрій – студент групи 2СП-22б факультету інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, м. Вінниця, e-mail: awful.braid@gmail.com.

Войцеховська Олена Валеріївна **–** кандидат технічних наук, доцент кафедри обчислювальної техніки, Вінницький національний технічний університет, Вінниця.

Smolinskyi Andrii – student of group 2SP-22b of the Faculty of Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: awful.braid@gmail.com.

Voitsekhovska Olena **–** Candidate of Technical Sciences, Associate Professor of the Department of Computer Techniques, Vinnytsia National Technical University, Vinnytsia.