УДК 004.75 DOI https://doi.org/10.32782/1995-0519.2023.2.9

# РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ ДИСТАНЦІЙНОГО НАВЧАННЯ: СЕРВЕРНА ЧАСТИНА

**Оксана Кошова**
кандидат педагогічних наук, доцент кафедри комп'ютерних наук та інформаційних технологій
Полтавський університет економіки і торгівлі, вул. Коваля, 3, Полтава, Україна, 36014,
koshova.o111@gmail.com
**ORCID: 0000-0003-0794-6774**

**Дмитро Ольховський**
кандидат фізико-математичних наук, доцент кафедри комп'ютерних наук та інформаційних технологій
Полтавський університет економіки і торгівлі, вул. Коваля, 3, Полтава, Україна, 36014,
dmitriy@olhovsky.name
**ORCID: 0000-0003-0313-6977**

**Оксана Черненко**
кандидат фізико-математичних наук, доцент кафедри комп'ютерних наук та інформаційних технологій
Полтавський університет економіки і торгівлі, вул. Коваля, 3, Полтава, Україна, 36014,
oksanachernenko7@gmail.com
**ORCID: 0000-0002-9084-0999**

**Іван Шаповалов**
бакалавр спеціальності комп'ютерні науки
Полтавський університет економіки і торгівлі, вул. Коваля, 3, Полтава, Україна, 36014, sluvkann@gmail.com
**ORCID: 0009-0007-4375-6704**

**Володимир Тур**
аспірант
Полтавський університет економіки і торгівлі, вул. Коваля, 3, Полтава, Україна, 36014,
Tur.vladimir1983@gmail.com
**ORCID: 0009-0003-2825-1434**

У роботі зроблено огляд систем дистанційного навчання, виділено їхні позитивні та негативні якості. Сформульовано вимоги до функціональності та безпеки для розробленої системи. Виконано опис проєктних рішень, інструментів і підходів до розроблення системи дистанційного навчання. Обрані засоби й інструменти для розроблення. Побудовано діаграму модулів системи. Розроблено серверну частину системи з використанням клієнт-серверної архітектури. Клієнтська частина виконана у вигляді вебдодатка з усіма описаними можливостями. Розроблена документація створеного серверного API. Серверна частина складається з таких модулів: бази даних; конфігурацій (взаємодія зі змінними середовища, глобальний); взаємодії з файловим сховищем (глобальний); файлів; авторизації; користувачів; кешу; курсів (курси, учасники курсу, матеріали курсу, виконані практичні завдання).

Одним із головних завдань сервісу є робота з даними в СУБД. Ця робота налаштована за допомогою ORM, яка бере на себе роль створення необхідних запитів до бази даних. Особливу увагу було приділено базі даних, у якій зберігаються всі дані про користувачів, курси, файли тощо. Невід'ємною складовою частиною якісного та стійкого API є валідація параметрів вхідних запитів. Опис валідації вхідних параметрів реалізований за допомогою декораторів сторонньої бібліотеки class-validator, яка також допомагає генерувати повідомлення про помилки, якщо деякі з параметрів запиту не пройшли валідацію.

Під час реалізації проєкту використано фреймворк для розроблення серверних додатків NestJS, систему управління базами даних PostgreSQL, Amazon S3-подібне файлове сховище, набір інструментів для документування серверного API Swagger, редактор коду Visual Studio Code, мову програмування TypeScript, систему контролю версій Git та інші допоміжні засоби та бібліотеки.

Розроблена система є результатом автоматизації навчального процесу в дистанційній формі. Саме тому вона покриває основні потреби студентів і викладачів у навчальному процесі. Серверна частина надає клієнтській частині все необхідне API для доступу до різних частин системи, але лише за умови, коли користувач має необхідні права доступу до даних. Результати розробки впроваджено в начальний процес Полтавського університету економіки та торгівлі.

**Ключові слова:** фреймворк, декоратор, модуль, контролер, сервіс.

**Актуальність роботи.** Популярність упровадження інтернет-технологій у більшість аспектів життя людини сьогодення досягла свого піку. Не залишилось майже жодної ніші, куди б не зазирнули прогрес та інновації, пов'язані з автоматизацією та поширенням доступності [1].

Не є винятком і сучасний підхід до навчання, до якого кожного дня впроваджуються все нові й нові підходи для поліпшення та спрощення навчального процесу як для студентів, так і для викладачів [2]. Для цього зазвичай використовуються системи управління навчанням (LMS), які адаптували в себе навчальні заклади всіх рівнів – від шкіл до коледжів і університетів. Дані системи надають учасникам навчального процесу великий інструментарій засобів, які можна легко використовувати для будь-якого типу викладання навчального матеріалу, як очно, так і дистанційно [3].

Огляд найбільш використовуваних натепер онлайн-платформ, зокрема Google Classroom [4], Moodle [5], Canvas [6], для навчання показав, що всі вони мають як переваги, так і недоліки.

**Метою роботи** є розроблення програмного забезпечення системи дистанційного навчання: серверна частина.

**Предметом розробки** є програмне забезпечення системи дистанційного навчання з використанням TypeScript фреймворку NestJS для створення серверної частини додатка.

**Матеріал і результати досліджень.** В основі роботи всієї системи лежить перевірка того, чи має конкретний користувач доступ до конкретного ресурсу. Для цього користувачі використовують облікові записи, які містять необхідну для перевірки доступу інформацію. Така перевірка має проходити під час кожного нового запиту до сервера, якщо доступ до запитуваного ресурсу якось обмежений. Наприклад, доступ до ендпоінту авторизації повинен мати кожний клієнт, який робить запит на сервер. Обліковий запис містить таку інформацію про користувача: ідентифікатор користувача; ідентифікатор поточної сесії; ідентифікатор прив'язаного телеграм-акаунта; ім'я; прізвище; ім'я по батькові; адресу електронної пошти; номер телефону (не обов'язково); пароль; роль; прив'язані групи; фото профілю; дату створення облікового запису; дату останнього внесення змін до облікового запису.

Необхідною умовою безпеки для користувача є наявність хешування пароля перед його зберіганням до бази даних. Після того, як користувач захоче повторно увійти до свого облікового запису, наданий ним пароль буде захешований, отриманий хеш буде порівняно з тим, що збережений у базі даних. Такий спосіб збереження пароля захистить його від можливого витоку даних користувачів.

Після успішного процесу авторизації користувач повинен отримати спеціальний підписаний сервером токен, який буде містити необхідну для сервера інформацію для подальшої валідації доступу до ресурсів, на які користувач робить запити.

Ідентифікатор необхідний для ідентифікації конкретного зареєстрованого в системі користувача.

Ідентифікатор поточної сесії використовується для того, щоб валідувати, чи робить користувач запит до сервера за допомогою токену, який був наданий йому під час останнього проходження процесу авторизації. Якщо вхід до облікового запису було виконано з будь-якого іншого пристрою, то сесія на всіх інших девайсах стає невалідною, він не матиме права доступу до серверного API, де умовою є бути авторизованим у системі.

Ідентифікатор прив'язаного телеграм-акаунта буде вказувати на телеграм-акаунт, з якого користувач використовує телеграм-бота.

Адреса електронної пошти є необхідною складовою частиною процесу авторизації та дає можливості відновлення доступу до облікового запису в разі втрати користувачем пароля.

Кожний користувач має одну з доступних у системі ролей. Такими можуть бути: студент, викладач, адміністратор. Даний параметр використовується для валідації доступу до конкретних ресурсів сервера, які мають обмежений доступ для деяких ролей.

Прив'язані групи необхідні користувачу для отримання поточного розкладу занять із системи, запит на який робиться до іншої системи, яка не прив'язана до поточної.

Дата створення та дата останнього редагування облікового запису необхідні для збереження останніх дій користувача з ним.

Ім'я, прізвище, ім'я по батькові, номер телефону та фото профілю необхідні лише для відображення всередині системи для інших користувачів, будь то студенти, викладачі чи адміністратори.

Важливою умовою є те, що облікові записи користувачів може створити лише адміністратор, причому пароль до ньих повинен бути згенерований системою автоматично та надісланий за вказаною користувачем адресою електронної пошти.

Кожен курс складається з основної інформації про курс, тем, матеріалів курсу, учасників і зданих робіт учасників курсу. Матеріал курсу може бути як лекційним, так і практичним, учасники курсу повинні виконати та здати викладачу на оцінювання завдання. Матеріал є частиною конкретної теми.

Основна інформація про курс: ідентифікатор курсу; назва курсу; фото оформлення; викладач; учасники; теми; дата створення; дата останнього внесення змін; дата видалення.

Учасник курсу має таку інформацію: ідентифікатор учасника; ідентифікатор користувача; ідентифікатор курсу; здані практичні роботи; дата створення; дата останнього внесення змін.

Теми курсу містять таку інформацію: ідентифікатор теми; назву; ідентифікатор курсу; матеріали курсу; дату створення; дату останнього внесення змін.

Матеріал курсу містить таку інформацію: ідентифікатор матеріалу; назву; опис; тип; прикріплені файли; ідентифікатор теми; здані практичні роботи; дату створення; дату останнього внесення змін.

Здана практична робота включає таке: ідентифікатор зданої практичної роботи; оцінку; ідентифікатор матеріалу; ідентифікатор учасника; файл; дату створення; дату останнього внесення змін.

Файл матеріалу чи зданої практичної роботи включає: ідентифікатор файлу; посилання на файл у віддаленому сховищі; ідентифікатор користувача (не обов'язково); ідентифікатор матеріалу (не обов'язково); ідентифікатор зданої практичної роботи (не обов'язково); ім'я; кодування; ключ; тег сутності; дату створення; дату останнього внесення змін.

Особливу увагу було приділено налаштуванню прав доступу до всіх компонентів курсу.

Зважаючи на вимоги до програмного продукту, система була розроблена з використанням клієнт-серверної архітектури. Клієнтська частина виконана у вигляді вебдодатка з усіма описаними можливостями.

Серверна частина додатка була реалізована за допомогою мови програмування TypeScript [7]. Її вибір зумовлений простотою написання коду та наявністю великої кількості бібліотек і фреймворків для написання вебдодатків.

Середовищем запуску коду була обрана платформа Node.js [8], яка працює на основі двигуна V8, надає можливість запускати код JavaScript як серверний [9].

СУБД для серверної частини була обрана PostgreSQL – це потужна об'єктно-реляційна СУБД з відкритим вихідним кодом, активна розробка якої триває понад 35 років, що заслужило їй міцну репутацію надійності, надійності функцій і продуктивності.

Фреймворком для розробки було обрано NestJS – це платформа для розробки вебдодатків із використанням паттерна MVC (Model View Controller), мови TypeScript і всіх її нововведень.

ORM для роботи з базою було обрано TypeORM – ORM, яка використовує можливості TypeScript для покращення процесу роботи з базою даних. Вона підтримує всі сучасні СУБД, зокрема і PostgreSQL. Вибір цієї ORM зумовлений легкою інтеграцією з NestJS, що дозволяє з легкістю використовувати можливості TypeORM усередині сервісів.

Сховищем для користувацьких файлів системи було обрано Amazon Simple Storage Service (Amazon S3) – це служба зберігання об'єктів, яка пропонує найкращі в галузі масштабованість, доступність даних, безпеку та продуктивність.

Під час розробки сервера для платформи дистанційного навчання також використовувалися різноманітні технології (Git, Prettier, ESLint, Swagger, Docker, MinIO) для покращення процесу написання вихідного коду. Такі інструменти ніяк не впливають на роботу самої системи, але допомагають розробникам полегшити процес створення необхідної функціональності, покращити якість написаного коду, прискорити швидкість його написання, зберегти його на віддаленому сховищі для запобігання його втраті або навіть налаштування автоматизованого процесу тестування та розгортання [10].

Одним із головних завдань сервісу є робота з даними в СУБД. Ця робота налаштована за допомогою ORM, яка бере на себе роль створення необхідних запитів до бази даних. Цією ORM стала TypeORM, з якою дуже зручно пра-

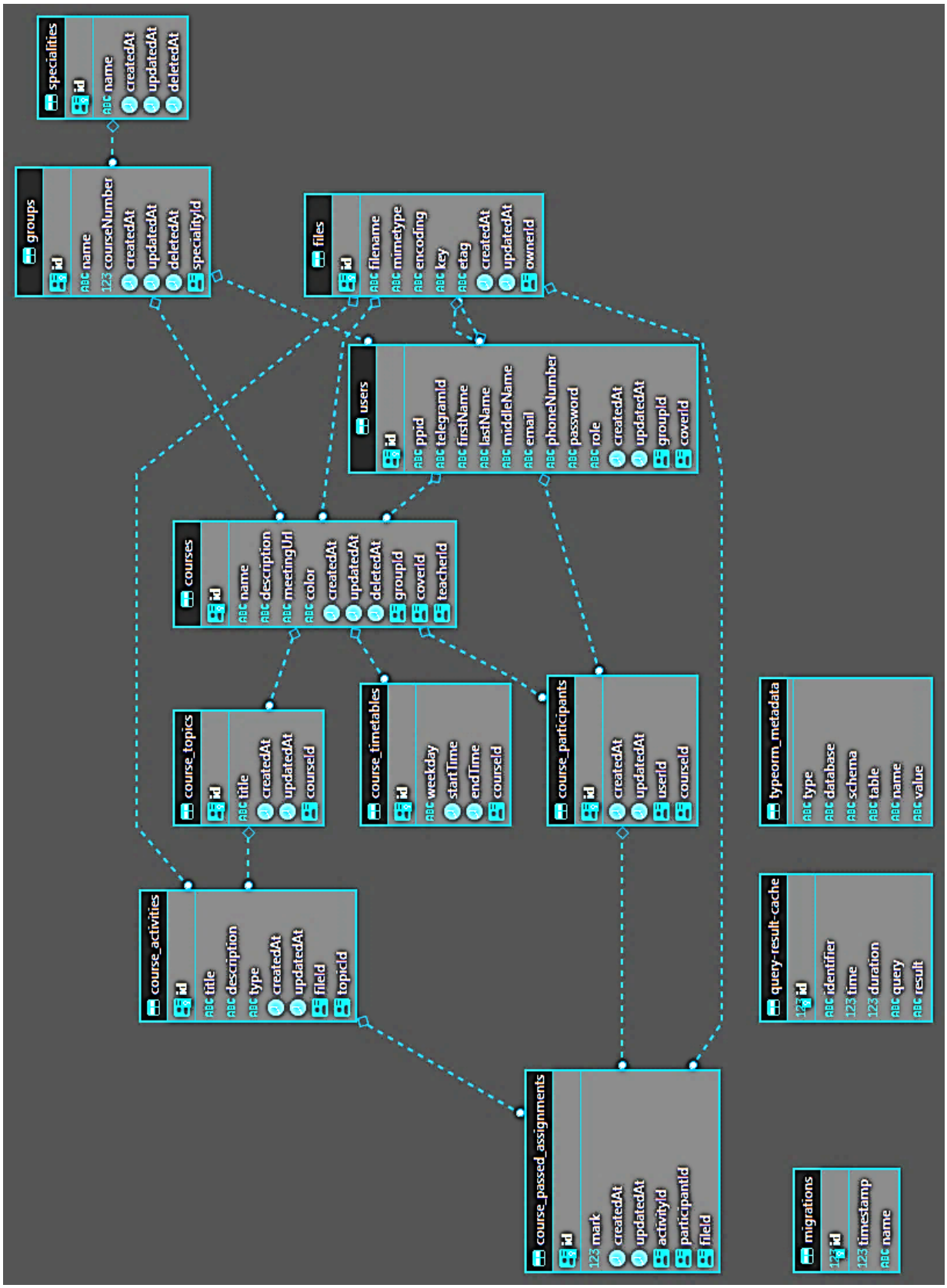


Рис. 1. Схема даних системи

цювати в комбінації з обраним фреймворком NestJS. Щоб ORM розуміла із чим їй доведеться працювати, були створені класи, які будуть описувати сутності. Кожній сутності була відведена окрема таблиця в базі даних. Сутність має вигляд класу з набором полів, які будуть зберігатися в таблицях, та набором декораторів для кожного з полів, які надає сама ORM. За допомогою декораторів конкретні поля були описані параметрами, якими так само можна було б описати поле в таблиці бази даних (первинний ключ, тип даних поля, чи є воно nullable або унікальним тощо). Після надання ORM класів-сутностей вона надає об'єкти класів, через які і можна працювати з даними. Класи цих об'єктів є реалізаціями паттерну «Репозиторій», який дозволяє працювати за допомогою простих абстракцій над СУБД. Також ORM має важливу особливість – вона бере на себе роль захисту від ін'єкцій SQL, які можуть негативно вплинути на цілісність даних у базі даних і роботу всієї системи загалом.

Особливу увагу було приділено базі даних, у якій зберігаються всі дані про користувачів, курси, файли тощо. Її структура перебуває у третій нормальній формі, що вказує на те, що схема даних створена максимально оптимізовано та не має надлишкових даних. У базі даних будуть наявні такі таблиці: користувачі; курси; учасники курсів; теми курсів; матеріали курсів; здані практичні роботи курсів; файли; міграції; кеш результатів запитів; метадані ORM [11].

Усі таблиці так чи так пов'язані одна з одною вторинними ключами. Таблиці міграцій, кешу результатів запитів і метадані ORM необхідні для коректної та швидкої роботи ORM, через яку проходять усі дії з базою даних. Далі наведена візуальна схема даних (рис. 1).

Невід'ємною складовою частиною якісного та стійкого API є валідація параметрів вхідних запитів [12]. Саме для цього були створені так звані DTO (data transfer object), які мають вигляд звичайного класу з набором полів, які є вхідними параметрами конкретного запиту, та набором валідаторів у вигляді декораторів. Якщо валідація була пройдена успішно, об'єкт даного класу передається далі до контролера для подальшої роботи. Якщо дані не пройшли валідацію, сервер повертає помилку зі статусом 400 та текст з поясненням, чому дані не пройшли валідацію. Опис валідації вхідних параметрів реалізований за допомогою декораторів сторонньої бібліотеки class-validator, яка також допомагає генерувати повідомлення про помилки, якщо деякі з параметрів запиту не пройшли валідацію.

Важливою складовою частиною системи дистанційного навчання є зберігання файлів із можливістю надання контрольованого доступу до них. Це необхідний аспект системи, який не дає доступу до перегляду файлів тим користувачам, які не мають доступу до них. Хоча і немає нічого страшного в тому, щоб видавати на показ фото профілю користувача, але зовсім інша річ, коли йдеться про обмеження доступу до файлів практичних робіт, які студенти надсилають викладачам на оцінювання.

Середовищем для збереження файлів було обране Amazon S3 (Simple Storage Service), яке є натепер одним із найпопулярніших і простих способів зберігати файли для роботи різного роду сервісів. Його концепт є досить простим –


Рис. 2. Схема роботи системи з Amazon S3

файли зберігаються у вигляді так званого об'єкта, який складається із 3 частин, як-от: його контент (тобто файл), його унікальний ідентифікатор (unique object identifier) і метадані, що зберігаються як пари «ключ – значення» і містять таку інформацію, як ім'я, розмір, дата, атрибути безпеки, тип вмісту об'єкта. Самі ж об'єкти зберігаються в так званих бакетах, які є базовими логічними контейнерами, де зберігаються файли. Бакет не має обмежень за кількістю збережених об'єктів, але сам об'єкт не може бути більшим за 5 ТБ за розміром. Далі наведена схема роботи із сервісом Amazon S3 (рис. 2).

За допомогою даної технології була реалізована система для збереження та контрольованого доступу до файлів усієї системи. Доступ до файлів надається у вигляді підписаних посилань. За допомогою них можна надавати доступ до файлу на конкретний період часу у вигляді URL. Часом доступу до файлу був обраний проміжок в 1 годину.

**Висновки.** Використання сучасних технологій в організації навчального процесу в освітніх закладах будь-якого типу стало базовою складовою частиною цієї системи, що змінило її назавжди. Результатом проведеної роботи є розробка серверної частини для системи дистанційного навчання Полтавського університету економіки та торгівлі з урахуванням поставлених вимог. Проведено тестування всіх модулів і доступних можливостей програмного продукту. У перспективі планується працювати над її вдосконаленням і покращенням функціоналу.

**ЛІТЕРАТУРА**

1. Близнюк С., Онофрійчук О. Забезпечення інформаційної безпеки в університетах під час військового стану. *Вісник Кременчуцького національного університету імені Михайла Остроградського.* 2022. Вип. 1 (132). С.79–85.

2. Круглик В., Астаф'єв В. Особливості реалізації семантичної нейронної мережі створення генератора навчальних кросвордів. *Вісник Кременчуцького національного університету імені Михайла Остроградського*. 2021. Вип. 2 (127). С. 81–88.

3. University website quality characteristics and success: lecturers' perspective / S. Almahamid et al. *International Journal of Business Information Systems*. 2016. № 22 (1). P. 41–61.

4. About Classroom – Classroom Help. URL: https://support.google.com/edu/classroom/answer/6020279.

5. Moodle App Overview. URL: https://moodledev.io/general/app/overview.

6. What is Canvas? URL: https://community.canvaslms.com/t5/Canvas-Basics-Guide/What-is-Canvas/ta-p/45.

7. Vanderkam Dan. Effective TypeScript: 62 Specific Ways to Improve Your TypeScript. 1'st edition. O'Reilly Media, November 12, 2019. 264 p.

8. Casciaro Mario, Mammino Luciano. Node.js Design Patterns. 2'nd edition. Packt Publishing, July 18, 2016. 777 p.

9. Nest.js: A Progressive Node.js Framework / Jay Bell et al. 2018. 313 p.

10. Slabinoha Marian. Core php and php frameworks: comparative analysis of two approaches for backend development. *Вісник Кременчуцького національного університету імені Михайла Остроградського.* 2022. Вип. 1 (132). С. 122–127.

11. Auderer Michael. How To Get Started with Node.js and Express Node.js. 2021. 304 p.

12. Rennay Dorasamy API Marketplace Engineering. Apress, 2022. 229 p.

## DEVELOPMENT OF DISTANCE LEARNING SYSTEM SOFTWARE: SERVER PART

**Oksana Koshova**
Candidate of Pedagogical Sciences, Associate Professor,
Associate Professor of Department of Computer Science and Information Technology
Poltava University of Economics and Trade, 3 Kovalya str., Poltava, Ukraine, 36014, koshova.o111@gmail.com
**ORCID: 0000-0003-0794-6774**

**Dmytro Olkhovsky**
Candidate of Physical and Mathematical Sciences, Associate Professor,
Associate Professor of Department of Computer Science and Information Technology
Poltava University of Economics and Trade, 3 Kovalya str., Poltava, Ukraine, 36000, dmitriy@olhovsky.name
**ORCID: 0000-0003-0313-6977**

**Oksana Chernenko**
Candidate of Physical and Mathematical Sciences, Associate Professor,
Associate Professor of Department of Computer Science and Information Technology
Poltava University of Economics and Trade, 3 Kovalya str., Poltava, Ukraine, 36000, oksanachernenko7@gmail.com
**ORCID: 0000-0002-9084-0999**

**Ivan Shapovalov**
Bachelor's degree in Computer Science
Poltava University of Economics and Trade, 3 Kovalya str., Poltava, Ukraine, 36014, sluvkann@gmail.com
**ORCID: 0009-0007-4375-6704**

**Volodymyr Tour**
Postgraduate
Poltava University of Economics and Trade, 3 Kovalya str., Poltava, Ukraine, 36014, Tur.vladimir1983@gmail.com
**ORCID: 0009-0003-2825-1434**

The paper provides an overview of distance learning systems, highlights their positive and negative aspects. Functionality and security requirements for the developed system are formulated. A description of project solutions, tools and approaches to the development of a distance learning system has been made. Selected means and tools for development. A diagram of system modules is built. The server part of the system was developed using a client-server architecture. The client part is made in the form of a web application with all the described capabilities.

Developed documentation of the created server API. The server part consists of the following modules: database; configurations (interaction with environment variables, global); interactions with file storage (global); files; authorization; users; cache; courses (courses, course participants, course materials, completed practical tasks).

One of the main tasks of the service is working with data in DBMS. This work is configured using an ORM, which takes over the role of creating the necessary database queries. Special attention was paid to the database, which stores all data about users, courses, files, and more. An integral component of a high-quality and stable API is the validation of the parameters of incoming requests. The description of the validation of input parameters is implemented using the decorators of the third-party class-validator library, which also helps to generate error messages if some of the query parameters have not passed validation.

The project uses the NestJS server application development framework, PostgreSQL database management system, Amazon S3-like file storage, Swagger server API documentation toolset, Visual Studio Code code editor, TypeScript programming language, Git version control system and other auxiliary tools. and libraries.

The developed system is the result of the automation of the educational process in a remote form. That is why it covers the basic needs of students and teachers in the educational process. The server part provides the client part with all the necessary API to access different parts of the system, but only if the user has the necessary data access rights. The results of the development were implemented in the initial process of the Poltava University of Economics and Trade.

**Key words:** framework, decorator, module, controller, service.

## REFERENCES

1. Blyzniuk, Serhii, Onofriichuk, Oleh (2022) Zabezpechennia informatsiinoi bezpeky v universytetakh pid chas viiskovoho stanu [Ensuring information security in universities during martial law]. *Visnyk Kremenchutskoho natsionalnoho universytetu imeni Mykhaila Ostrohradskoho* [*Visnyk Kremenchug National University named after Mykhailo Ostrogradskyi*]. Kremenchuk: KrNU. Vypusk 1 (132). P. 79–85 [in Ukrainian].

2. Kruhlyk, V.S., Astafiev, V.Yu. (2021). Osoblyvosti realizatsii semantychnoi neironnoi merezhi stvorennia heneratora navchalnykh krosvordiv [Peculiarities of implementing a semantic neural network for creating an educational crossword generator]. *Visnyk Kremenchutskoho natsionalnoho universytetu imeni Mykhaila Ostrohradskoho* [*Visnyk Kremenchug National University named after Mykhailo Ostrogradskyi*]. Kremenchuk: KrNU. Vypusk 2 (127). P. 81–88 [in Ukrainian].

3. Almahamid, S.M., Tweiqat, A.F., Almanaseer, M.S. (2016). University website quality characteristics and success: lecturers' perspective. *International Journal of Business Information Systems*, 22 (1), P. 41–61 [in English].

4. About Classroom – Classroom Help. URL: https://support.google.com/edu/classroom/answer/6020279 [in English].

5. Moodle App Overview. URL: https://moodledev.io/general/app/overview [in English].

6. What is Canvas? URL: https://community.canvaslms.com/t5/Canvas-Basics-Guide/What-is-Canvas/ta-p/45 [in English].

7. Vanderkam, Dan (2019). Effective TypeScript: 62 Specific Ways to Improve Your TypeScript – O'Reilly Media. 1'st edition. 264 p. [in English].

8. Casciaro, Mario, Mammino, Luciano (2016). Node.js Design Patterns – Packt Publishing. 2'nd edition. 777 p. [in English].

9. Bell, Jay, Magolan, Greg, Guijarro, David, Peretti, Adrien de, Housley, Patrick (2018). Nest.js: A Progressive Node.js Framework. 313 p. [in English].

10. Slabinoha, Marian (2022). Core php and php frameworks: comparative analysis of two approaches for backend development. *Visnyk Kremenchutskoho natsionalnoho universytetu imeni Mykhaila Ostrohradskoho* [*Visnyk Kremenchug National University named after Mykhailo Ostrogradskyi*]. Kremenchuk: KrNU. Vypusk 1 (132). P. 122–127 [in English].

11. Auderer, Michael (2021). How To Get Started with Node.js and Express Node.js. 304 p. [in English].

12. Dorasamy, Rennay (2022). API Marketplace Engineering. Apress, 229 p. [in English].

*Стаття надійшла 07.04.2023*