УДК 004.42

ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ КЕРУВАННЯ ЕЛЕКТРОННИМИ ЖУРНАЛОМ УСПІШНОСТІ ТА РОЗКЛАДОМ ЗАНЯТЬ У ЗАКЛАДАХ ВИЩОЇ ОСВІТИ

М. П. Галенюк, Д. О. Бачманюк
здобувачі вищої освіти першого (бакалаврського) рівня вищої освіти, 2 курс,
спеціальність «Комп'ютерна інженерія»,
навчально-науковий інститут кібернетики, інформаційних технологій та інженерії
Науковий керівник – к.т.н., доцент М. В. Бойчура

Національний університет водного господарства та природокористування,
м. Рівне, Україна

Розроблено інформаційну систему для керування електронним журналом успішності та розкладом занять. В її основі лежать фреймворк ASP.NET Core, бібліотека React та Clean Architecture. Ключовими особливостями є використання сучасних дизайну, технологій та підходів, що забезпечує зручність, гнучкість та розширюваність додатку.
Ключові слова: ASP.NET, React, розклад, журнал успішності, вебдодаток.

An information system for managing an electronic gradebook and class timetable has been developed. It is based on the ASP.NET Core framework, React library and Clean Architecture. The key features are the use of modern design, technologies and approaches, which ensures the convenience, flexibility and extensibility of the application.
Keywords: ASP.NET, React, timetable, gradebook, web application.

Сучасні вебінструменти для формування та відображення електронного журналу з оцінками студентів і розкладу мають суттєві недоліки. Головними з них є недостатньо сучасний дизайн та незручність розміщення елементів інтерфейсу. Зазвичай причинами цього є використання конструкторів вебсайтів (що суттєво знижує гнучкість) та дещо застарілих інструментів і технологій веброзробки.

Зокрема система Moodle [1] є недостатньо зручною з точки зору UX/UI, має низький рівень адаптивності та є порівняно складною у навігації (через перевантаженість сторінок різноманітними посиланнями). Також в багатьох закладах вищої освіти часто використовується сервіс Google Classroom [2]. Проте він має стандартизований інтерфейс, який неможливо адаптувати під індивідуальні потреби викладачів або студентів, що обмежує можливість персоналізації навчального процесу. Окрім цього, Google Classroom не надає достатньої гнучкості в налаштуванні прав доступу. Також важливо зазначити, що можливості інтеграції з зовнішніми системами є обмеженими, зокрема при бажанні організації взаємодії з локальними базами даних, сторонніми сервісами або спеціалізованими освітніми платформами. Ще одним популярним ресурсом є ПОЛІТЕК-СОФТ [3], проте він є платним і потребує покращень з точки зору UX/UI.

Метою роботи є розробка комплексного інструменту для ведення та відображення електронного журналу і розкладу занять із використанням сучасних технологій веброзробки.

Розробка вебдодатка в умовах сьогодення початково потребує створення дизайну, після чого – реалізації Front-End [4] та Back-End [5] складових. Найбільш поширеним інструментом для формування дизайну є Figma [6]. Не зменшуючи загальності, візуальну складову вебсайту розроблятимемо на прикладі брендбука Національного університету водного господарства та природокористування (НУВГП).

Дизайн додатка витриманий у сучасному, візуально «чистому» стилі з використанням відтінку #256589, заданого як CSS-змінна --nuwmcolor. Це дозволяє легко підтримувати єдину колірну палітру на всіх вебсторінках. Шрифт Montserrat додає інтерфейсу елегантності, легкості та зручності для сприйняття тексту, незалежно від його контексту та розміщення. Фонове зображення містить логотип навчального закладу сірого кольору, не відволікаючи таким чином користувача від вмісту, але задаючи загальний настрій, створюючи глибину, атмосферу та естетичний акцент. Контент організовано через центральний контейнер з фіксованою максимальною шириною, що забезпечує гармонійне вирівнювання елементів і візуальний баланс, незалежно від кількості елементів на вебсторінці. Такий підхід до оформлення дозволяє створити візуально приємний і цілісний вигляд інтерфейсу, водночас зберігаючи структурованість, послідовність і професійність на кожному етапі взаємодії з користувачем. З метою підвищення рівня зручності перегляду на різних пристроях було створено дизайн під мобільну версію вебдодатка.

Розробка Front-End складової передбачається на основі запропонованого дизайну з використанням бібліотеки React [7]. Ключовими її перевагами є висока продуктивність, швидкість виконання певних функцій та легка інтеграція з ASP.NET Core [5] (популярний фреймворк для побудови Back-End, про який згадуватиметься нижче). Це забезпечує ефективний обмін даними між сервером та клієнтом. Адаптивність реалізована через використання медіа-запитів, які дозволяють змінювати стилі елементів залежно від ширини екрану (смартфони, планшети і комп'ютери). При реалізації додатку важливо дотримуватись принципів компонентного підходу, що є необхідним для ефективної розробки в React: вся логіка інтерфейсу поділена на окремі незалежні частини. Компоненти побудовані за принципами модульності, тобто кожен виконує одну конкретну задачу, що спрощує розробку, тестування та масштабування. Структура додатку логічно розділена на папки сторінок (Pages) та окремих елементів інтерфейсу (Components), згідно з найкращими практиками організації React-проєктів. Для передачі даних між компонентами використовуються props, а стан (state) локалізовано в межах конкретних компонентів. Завдяки цьому проєкт є легко розширюваним та підтримує адаптивність при додаванні нових функцій.

Back-End складова вебсайту будується на основі фреймворку ASP.NET Core, архітектури Clean Architecture та патернів Dependency Injection і MVC [5]. Таке поєднання забезпечує наступні переваги: завдяки патерну MVC забезпечується висока гнучкість у розробці, адже наявний чіткий поділ на представлення, модель і контролер. Це дозволяє розробникам незалежно один від одного працювати над різними частинами додатка, що сприяє легкому додаванню нового функціонала і загальному прискоренню розробки вебдодатка. Clean Architecture дозволятиме у майбутньому легко масштабувати додаток, що спрощуватиме додавання нових складових без впливу на інші. Поєднання наведених підходів також полегшує тестування коду та його подальшу підтримку.

Для роботи із базою даних у цьому випадку доцільно використовувати патерн Repository [8], фреймворк Entity Framework, інструменти міграції та специфікації доступу до бази даних [9]. Це сприяє зручному керуванню інформацією завдяки, так званому, об'єктно-реляційному механізму відображення. Такий підхід дає змогу зручно будувати складні запити та фільтри у вигляді окремих модулів, що підвищує гнучкість і повторне використання коду, подальшу підтримку та масштабування додатку. Задля зручності користування базою даних було обрано систему Microsoft SQL Server 2022 [5; 9], яка забезпечує стабільну роботу з великими обсягами даних. У додатку застосовуються моделі Data Transfer Object (DTO) [9], які виступають посередниками між клієнтською частиною застосунку та внутрішніми доменними моделями. Такий підхід підвищує безпеку, дозволяючи уникати надлишкової передачі інформації. Перетворення між DTO та

доменними моделями здійснюється з використанням AutoMapper [9], що підвищує структурованість коду.

Задля організації зв'язку із Front-End складовою передбачена розробка API та налаштування CORS [10]. А для коректної Google-автентифікації необхідне підключення та налаштування протоколу OAuth 2.0, що надає змогу інтеграції Google Sing-In для ідентифікації користувача [11]. Це забезпечує легку побудову системи безпеки додатка, одночасно із розподілом ролей між певними користувачами та надання доступу до відповідних API.

Розроблюваний додаток, очевидно, повинен містити наступні ролі: студент, викладач та адміністратор. Задля забезпечення зручності та безпечності використання вебдодатка варто використовувати лише Google-автентифікацію на основі наперед внесених у систему електронних пошт.

У результаті розроблено вебдодаток із сучасним дизайном та використовуваними фреймворками для Front-End та Back-End складових. Типовий сценарій роботи адміністратора початково передбачає Google-автентифікацію. Після цього, з використанням спеціалізованої панелі, можливим є внесення переліку дисциплін, контингенту студентів і викладачів, налаштування зв'язку між ними, а також формування розкладу занять (рис. 1).

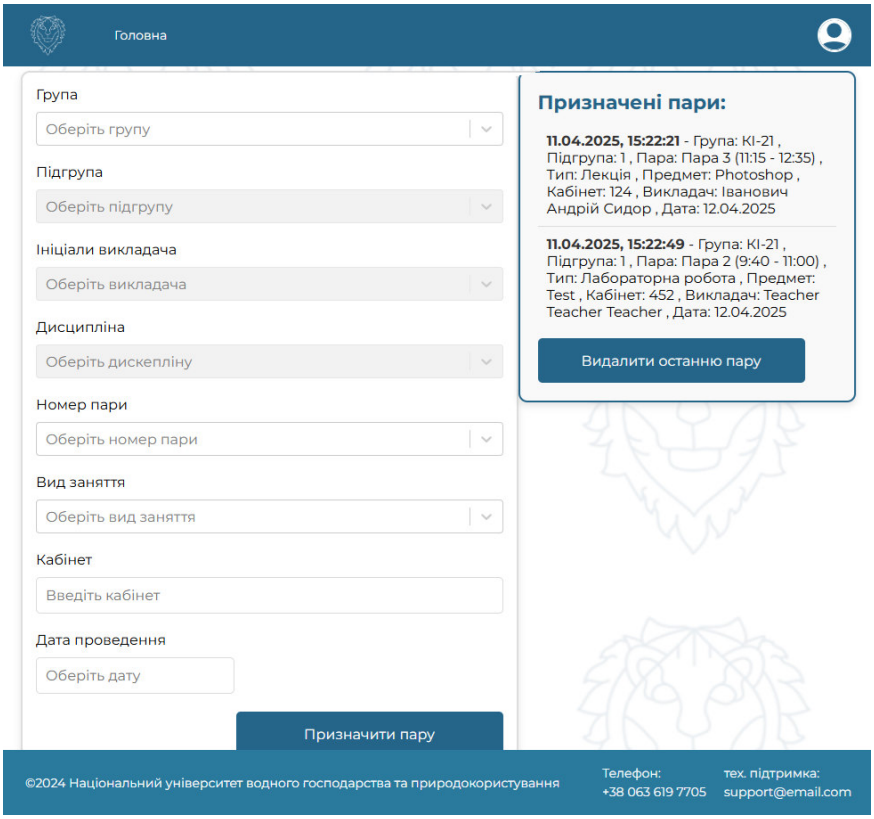

Рис. 1. Сторінка додавання розкладу адміністратором

Діяльність викладача в межах розробленого додатка також вимагає проходження кроку Google-автентифікації. Після чого він може переглядати розклад своїх пар, вносити оцінки за окремі дисципліни для студентів тієї чи іншої групи, а також зв'язуватись з технічною підтримкою.

Студент же після Google-автентифікації може переглядати розклад пар своєї групи, оцінки за увесь час навчання та бачити загальні бали кожного з студентів своєї групи (тобто рейтинг), а також зв'язуватись з технічною підтримкою (рис. 2 та рис. 3).

Розроблено інформаційну систему для ведення та відображення електронного журналу і розкладу занять у вигляді вебдодатка із використанням фреймворку ASP.NET Core, бібліотеки React та архітектури Clean Architecture. Таке поєднання забезпечує легку розробку візуально привабливих та функціональних вебсторінок, а також широкі можливості для подальшого удосконалення функціоналу вебзастосунку. Використання сучасних підходів (зокрема Clean Architecture, API, специфікацій доступу до баз даних) сприяє легкій підтримуваності та розширюваності коду додатка.

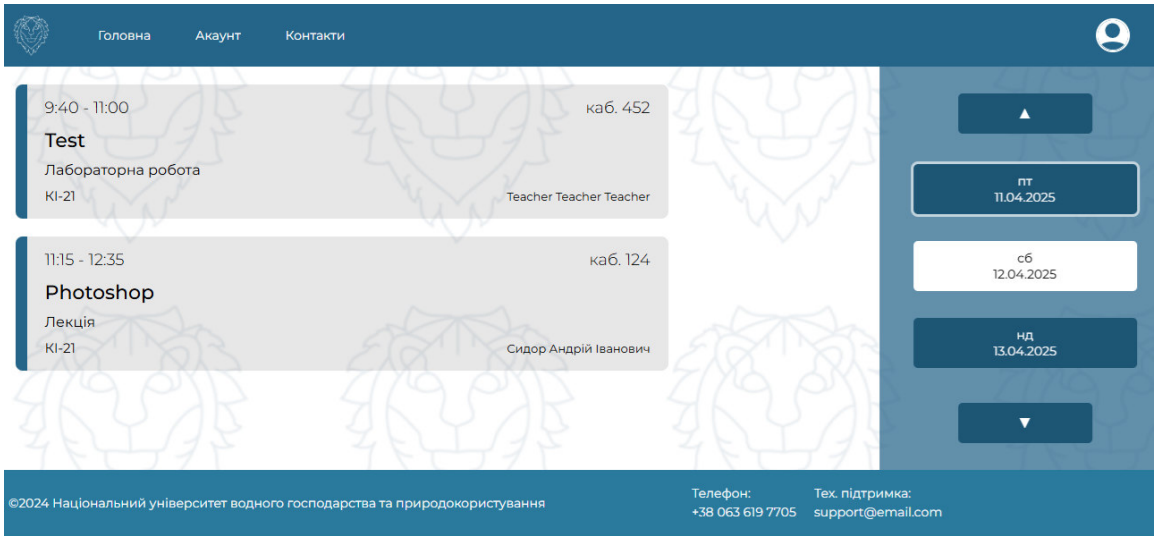

Рис. 2. Сторінка перегляду розкладу студентом

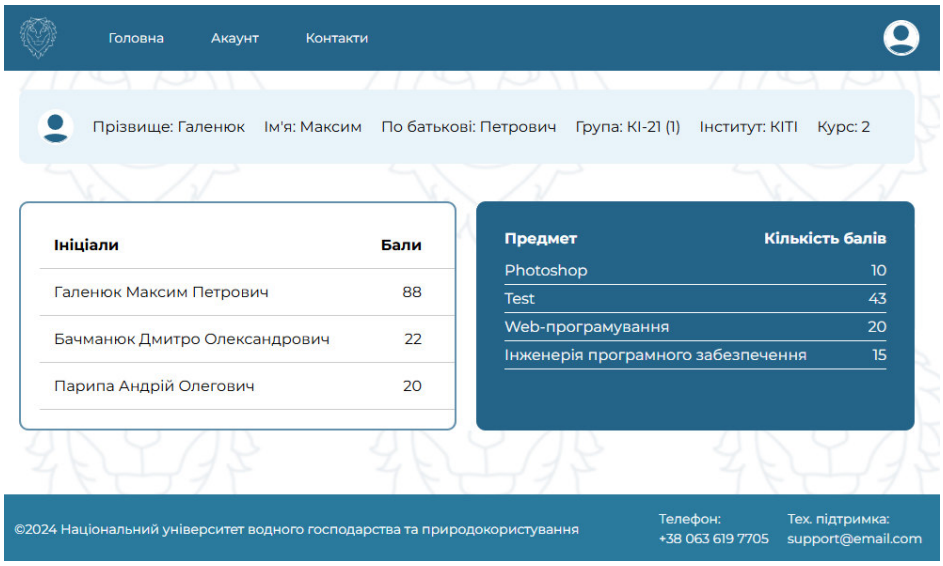

Рис. 3. Сторінка із поточними балами автентифікованого студента та рейтингом в групі

У перспективі передбачається інтеграція додаткових заходів безпеки у вебдодаток, додавання інших ролей користувачів (завідувач кафедри та контент-адміністратор) і низки фільтрів задля покращення користувацького досвіду.

1. Home | Moodle.org. URL: https://moodle.org/ (дата звернення: 25.03.2025). **2.** Google Classroom. URL: https://classroom.google.com/ (дата звернення: 25.03.2025). **3.** ПП «Політек-СОФТ». Головна. URL: http://www.politek-soft.kiev.ua/ (дата звернення: 25.03.2025). **4.** Front-End. URL: https://dan-it.com.ua/uk/blog/rozrobka-z-boku-front-end-shho-ce-take-i-chim-vidriznjaietsja-vid-back-end/ (дата звернення: 25.03.2025). **5.** ASP.NET Core | Open-source web framework for .NET. URL: https://dotnet.microsoft.com/en-us/apps/aspnet (accessed: 19.11.2024). **6.** Figma. URL: https://www.figma.com/about/ (дата звернення: 25.03.2025). **7.** Sakhniuk M., Boduch A. React and React Native - Fifth Edition: Build cross-platform JavaScript and TypeScript apps for the web, desktop, and mobile. 5th ed. Birmingham : Packt Publishing, 2024. 508 p. **8.** Repository. URL: https://dotnettutorials.net/lesson/repository-design-pattern-csharp/ (дата звернення: 25.03.2025). **9.** Smith J. P. Entity Framework Core in Action. 2nd ed. Shelter Island : Manning, 2021. 624 p. **10.** Gunasundaram R., Goya R. CORS Essentials: Access web resources on different domains. 1st ed. Birmingham: Packt Publishing, 2017. 248 p. **11.** Google Sing-In. URL: https://developers.google.com/identity (дата звернення: 25.03.2025).