УДК 004.738.5:371.315.7

Снитко А. Р., *здобувачка вищої освіти 4 курсу спеціальності 122 Комп'ютерні науки*,
Антонов Ю. С., *канд. фіз.-мат. наук, доцент, доцент кафедри інформаційних технологій*

ВЕБДОДАТОК ДЛЯ ТЕСТУВАННЯ ЗНАНЬ ТА ПРОВЕДЕННЯ ОПИТУВАНЬ

Донецький національний університет імені Василя Стуса, м. Вінниця

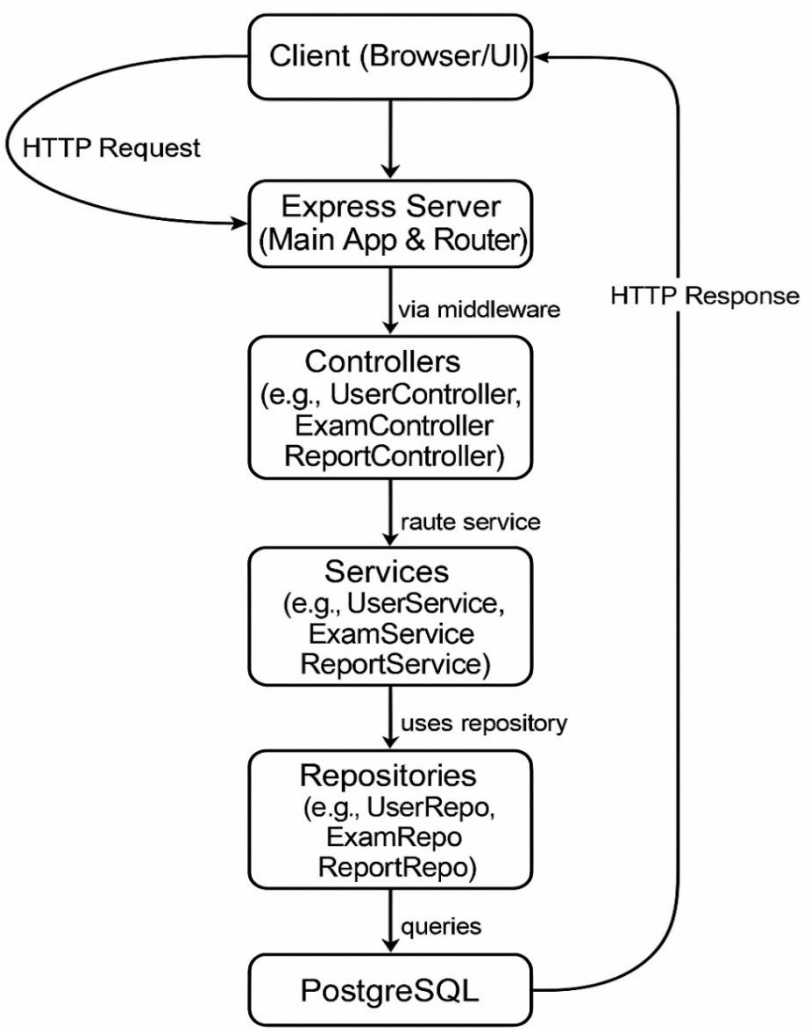
В умовах стрімкої цифровізації освіти та бізнесу вебсистеми для проведення опитувань і тестування знань набувають критичної важливості, особливо зважаючи на поширення дистанційних форм навчання та роботи [1–5]. Однак наявні платформи часто мають технічні та методичні обмеження, що знижують їх ефективність, зручність використання й адаптивність до потреб користувачів, зокрема у контексті забезпечення масштабованості, надійності та підтримки мотивації.

Для досягнення мети були реалізовані такі кроки:

- проаналізовано еволюцію систем тестування й опитувань, визначено їх сильні й слабкі сторони;
- сформовано функціональні та нефункціональні вимоги до нової системи;
- спроєктовано модель даних у PostgreSQL, що підтримує нормалізацію (1–3 НФ) та запитання з кількома правильними відповідями;
- розроблено back-end на Node.js з фреймворком Express та Drizzle ORM, реалізовано REST-API, JWT-захист та механізми логування;
- створено front-end SPA на React з використанням бібліотеки компонентів Ant Design, адаптивним інтерфейсом та клієнтським роутингом;
- інтегровано механізми логування й базову аналітику для відстеження взаємодії користувачів та формування звітів.

Система реалізована мовою JavaScript із застосуванням сучасного стеку технологій. Серверна частина (back-end) розроблена на платформі Node.js з використанням фреймворку Express.js та Drizzle ORM для взаємодії з реляційною СУБД PostgreSQL. Клієнтська частина (front-end) є односторінковим додатком (SPA) на базі бібліотеки React та UI-компонентів Ant Design.

Архітектура системи є трирівневою («клієнт – сервер застосунків – база даних»), що забезпечує модульність, масштабованість і чіткий розподіл відповідальності між компонентами. Ключові функціональні компоненти включають: модуль реєстрації та автентифікації користувачів (адміністратор, звичайний користувач); модуль створення та управління тестами й опитуваннями, що підтримує питання з кількома правильними відповідями; модуль проходження тестування з можливістю обмеження часу; модуль автоматичної обробки, збереження та перегляду результатів. На рис. 1 зображена архітектура back-end-частини вебдодатка з використанням багатошарової структури.



У підсумку реалізовано повнофункціональний вебдодаток для проведення опитувань і тестування знань. Система забезпечує реєстрацію та автентифікацію користувачів, створення й управління тестами та питаннями (включно з підтримкою питань із кількома правильними відповідями), проходження тестувань з обмеженням часу, автоматизовану обробку та перегляд результатів. Реалізовано розмежування ролей користувачів та механізми логування дій для аналізу та вдосконалення. Розробка має наукову новизну у комплексному підході до вирішення проблем наявних платформ шляхом інтеграції логування та базової аналітики, та практичну цінність для автоматизації процесів оцінювання в освітніх закладах і збору даних у бізнес-середовищі.

Список використаних джерел

1. Антонов Ю. С., Смоктій К. В. Використання експертних систем для аналізу відповідей у автоматизованих системах контролю знань. *Вісник Хмельницького національного університету. Технічні науки*. 2024. № 4(339). С. 323–331. DOI: 10.31891/2307-5732-2024-339-4-51.

2. Зінов'єва І. С., Зембіцька А. Г. Порівняльна характеристика сучасних онлайн-інструментів тестування знань при дистанційному навчанні. *Електронне моделювання.* 2021. Т. 43, № 3. С. 109–123. DOI: 10.15407/emodel.43.03.109 (дата звернення: 24.10.2022).

3. Антонов Ю. С. Комп'ютерні системи тестування на основі технології трирівневих баз даних. *Інформаційні технології і засоби навчання.* 2008. № 2. DOI: 10.33407/itlt.v6i2.133. URL: http://www.nbuv.gov.ua/e-journals/ITZN/em6/content/08aystdt.htm

4. Антонов Ю. С. Оцінка повноти відповідей в автоматизованих системах контролю знань. *Наукові праці Донецького національного технічного університету. Сер.: Інформатика, кібернетика та обчислювальна техніка.* 2012. № 15. С. 113–117.

5. Антонов Ю. С., Космінська О. М. Методика аналізу тестових завдань на основі отриманих результатів тестування. *Інформаційні технології і засоби навчання.* 2009. № 4. DOI: 10.33407/itlt.v12i4.81. URL: https://journal.iitta.gov.ua/index.php/itlt/article/view/81/

УДК 004.42:004.43

Чернишенко Я. А., *здобувач вищої освіти 4 курсу спеціальності 122 Комп'ютерні науки,*
Зелінська О. В., *канд. техн. наук, доцент,*
в. о. завідувача кафедри інформаційних технологій

ЗАСТОСУВАННЯ ШАБЛОНУ MVVM У РЕАЛІЗАЦІЇ UI-ЛОГІКИ ДЕСКТОПНИХ ДОДАТКІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасній розробці десктопних додатків шаблон Model-View-ViewModel (MVVM) став одним із ключових підходів до організації логіки користувацького інтерфейсу (UI). Він спрямований на чітке розділення графічного інтерфейсу (View) та прикладної логіки, що дає змогу уникнути надмірної залежності між ними. Завдяки MVVM інтерфейс користувача визначається окремо від коду, який обробляє дії користувача та оперує даними додатка. Такий підхід підвищує зрозумілість структури програми та закладає основу для більш керованої і масштабованої розробки [1].

Сьогодні MVVM фактично став стандартом проєктування інтерфейсної логіки у багатьох сучасних фреймворках для десктопних застосунків. Зокрема, у технологіях на кшталт Windows Presentation Foundation (WPF) та сучасних кросплатформних фреймворках – Avalonia, .NET MAUI [1]. Цей шаблон використовується для побудови чіткого розмежування між відображенням даних і бізнес-логікою. Популярність MVVM пояснюється тим, що він розв'язує типові проблеми розробки UI:

- ускладнення підтримки коду під час росту додатка;
- труднощі тестування;
- повторного використання компонентів.

Завдяки цьому MVVM залишається актуальним і вважається ефективним підходом для створення складних інтерфейсів користувача у сучасних десктопних застосунках.

Шаблон MVVM передбачає чітке розподілення відповідальностей між трьома складниками:

- Model;
- View;
- ViewModel.

Модель (Model) відповідає за управління даними і бізнес-логікою застосунку, тоді як вигляд (View) містить тільки опис графічного інтерфейсу – вікон, кно-