ВЕБДОДАТОК ДЛЯ АВТОМАТИЗАЦІЇ РЕСТОРАННОГО БІЗНЕСУ З ВИКОРИСТАННЯМ ASP.NET CORE ТА VUE.JS

Бабенко А.Р.
Київський столичний університет імені Бориса Грінченка, м. Київ

ВСТУП

Актуальність і постановка проблеми. Ресторанний бізнес активно впроваджує інформаційні технології для обслуговування клієнтів, управління меню та координації персоналу. Традиційні підходи - паперові меню, усне приймання замовлень, відсутність єдиної бази цін і наявності позицій - часто призводять до затримок, помилок і витрат на друк [6]. Поширення безконтактного обслуговування з використанням QR-кодів забезпечує швидкий доступ клієнтів до актуального меню на смартфоні. Водночас готові SaaS-рішення (Choice, SmartSpot, Expirenza) зазвичай мають обмежену гнучкість, залежність від тарифів і слабкі можливості адаптації під нетипову структуру мережі [6; 7; 8]. Тому актуально розробити власний вебдодаток, який поєднує зручність QR-меню з повним контролем бізнес-логіки та можливістю масштабування.

Мета дослідження. Розробити веб-додаток на ASP.NET Core та Vue.js, який забезпечує клієнтам доступ до меню через QR-код і можливість оформлення замовлень, а також надає персоналу інструменти для управління мережею закладів, меню, замовленнями, користувачами та базовою аналітикою продажів.

Аналіз останніх досліджень і публікацій. Теоретичну базу сформовано з праць з інженерії програмного забезпечення та архітектури корпоративних застосунків [1; 2; 3], а також з матеріалів об'єктно-орієнтованого проєктування [4; 5]. Серед комерційних аналогів розглянуто Choice - орієнтований на швидкий запуск електронного меню з обмеженим керуванням замовленнями [6]; SmartSpot - з ширшим функціоналом і базовою аналітикою, але складнішим інтерфейсом [7]; Expirenza - комплексну платформу з інтеграцією платежів, у якій значна частина функціоналу доступна лише в платних тарифах [8]. Для обґрунтування стеку використано офіційну документацію Vue.js, ASP.NET Core, Entity Framework Core, PostgreSQL, Vite, Axios, BCrypt і Docker [9; 10; 11; 12; 13; 14; 15; 16]. На основі проведеного аналізу встановлено, що типові рішення добре реалізують функцію онлайн QR-меню. Водночас такі можливості, як гнучке розмежування прав доступу, збереження історії замовлень і підтримка мережі закладів, часто потребують власної розробки. Для закладів із кількома точками важливо мати єдину систему обліку товарів і замовлень із можливістю відрізняти ціни та наявність без втрати цілісності даних. Саме це враховано у запропонованому підході до проєктування системи.

Короткий опис дослідження, його методів і засобів. Проведено аналіз предметної області та існуючих рішень, виконано проєктування, моделювання реляційної бази даних, розробку REST API, функціональне тестування та оцінку продуктивності. Сформульовано функціональні вимоги: перегляд меню через QR-код, оформлення замовлення клієнтом, управління категоріями й товарами, обробка замовлень, адміністрування користувачів і ролей; нефункціональні вимоги - продуктивність, адаптивність інтерфейсу, масштабованість, базова безпека даних і надійність.

Технологічний стек: серверна частина - ASP.NET Core [11]; доступ до даних - Entity Framework Core [12]; СУБД - PostgreSQL [13]; хешування паролів - BCrypt [14]. Клієнтська частина - Vue.js [10] з компонентною архітектурою; HTTP-запити - Axios [16]; збірка - Vite [15]. Розгортання уніфіковано через Docker [14]. Поєднання ASP.NET Core на бекенді та Vue на фронтенді забезпечує можливість розвивати клієнтські інтерфейси незалежно від серверних контролерів і сервісів, зберігаючи стабільний контракт API для обох застосунків.

Архітектура - клієнт-сервер із розділенням рівнів представлення, бізнес-логіки та доступу до даних [3; 5]. Реалізовано два фронтенд-застосунки (QR-меню для відвідувачів і адміністративна панель) і два відповідні серверні застосунки; спільне ядро містить сутності домену, конфігурацію ORM і спільну логіку [2; 4], що зменшує дублювання коду. Розділення серверних застосунків дозволяє спростити структуру публічного API для гостей (без зайвої адміністративної логіки) й водночас зосередити перевірки прав у панелі персоналу. Обмін даними - REST і JSON, що полегшує подальше підключення мобільних клієнтів або зовнішніх інтеграцій без переробки ядра.

Базу даних спроєктовано відповідно до принципів нормалізації. Основні сутності: мережі та заклади, категорії й товари, окремі таблиці цін і фото (різні ціни на один товар у різних точках мережі), замовлення та позиції зі збереженням snapshot (назва, ціна, категорія на момент замовлення), історія статусів, персонал замовлення, столи з QR, користувачі, ролі, права, зв'язки користувач-заклад, сесії. Підхід snapshot для позицій замовлення гарантує, що зміни в картці товару не спотворюють уже оформлені замовлення й звіти. Застосовано зовнішні ключі, унікальні обмеження (наприклад, назва товару в межах мережі, номер столу в межах закладу) та індекси для прискорення типових вибірок за замовленнями, товарами й користувачами.

РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

Спроєктовано й реалізовано вебдодаток із такими модулями: замовлення (створення з QR-меню, історія статусів, кілька виконавців за ролями); мережа закладів і столи з унікальними QR-кодами; користувачі, ролі та запрошення для підключення персоналу; товари й категорії з

цінами по закладах і вибірковою наявністю позицій; аналітика обсягів продажів і прибутку. Разом ці модулі покривають цикл від прийому замовлення гостем до обліку результатів на рівні мережі закладів.

Онлайн QR-меню (рис 3) - адаптивний SPA зі сторінками: головна, меню за категоріями , кошик, перегляд замовлень і статусів. Типовий сценарій "сканування QR - меню - кошик - замовлення" виконується за 2-3 кліки без реєстрації. Інтерфейс оптимізовано для смартфонів.

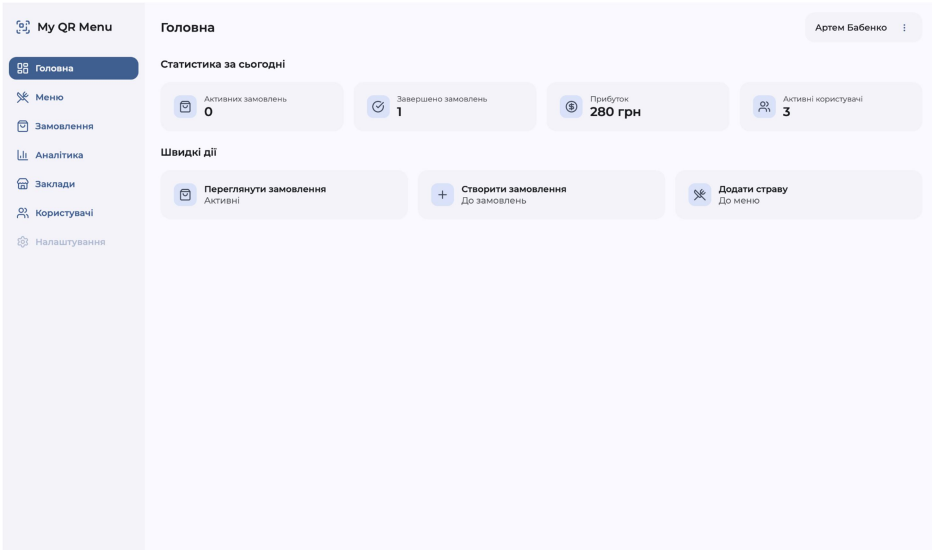

Рис. 1. Головна сторінка з швидкими діями.

Адміністративна панель (рис 1-2) забезпечує редагування меню й цін, обробку замовлень зі зміною статусів, керування закладами та генерацію QR для столів, адміністрування користувачів і ролей, перегляд аналітики; реалізовано світлу/темну тему та адаптивність. Контроль доступу реалізовано на рівні окремих дій [1; 2].

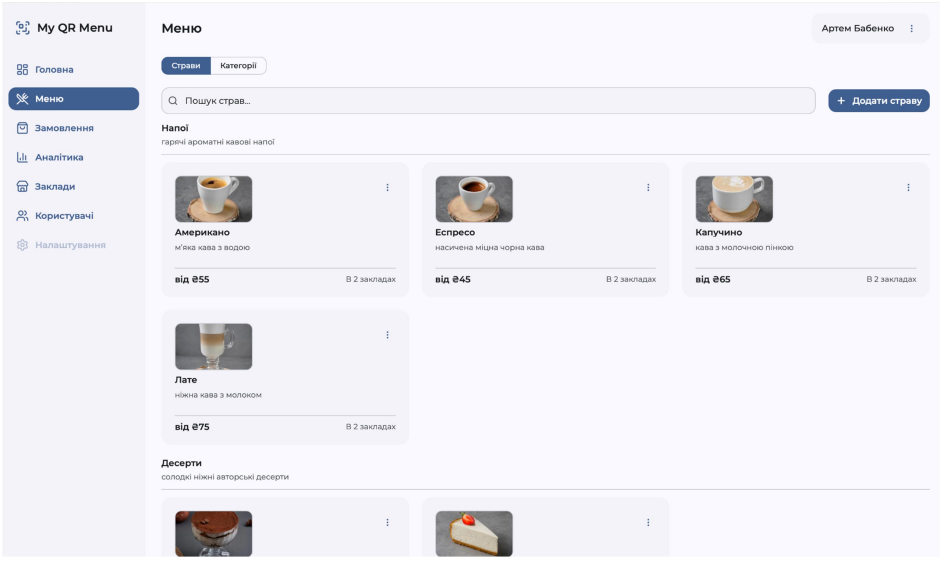

Рис.2. Сторінка налаштування меню та категорій.

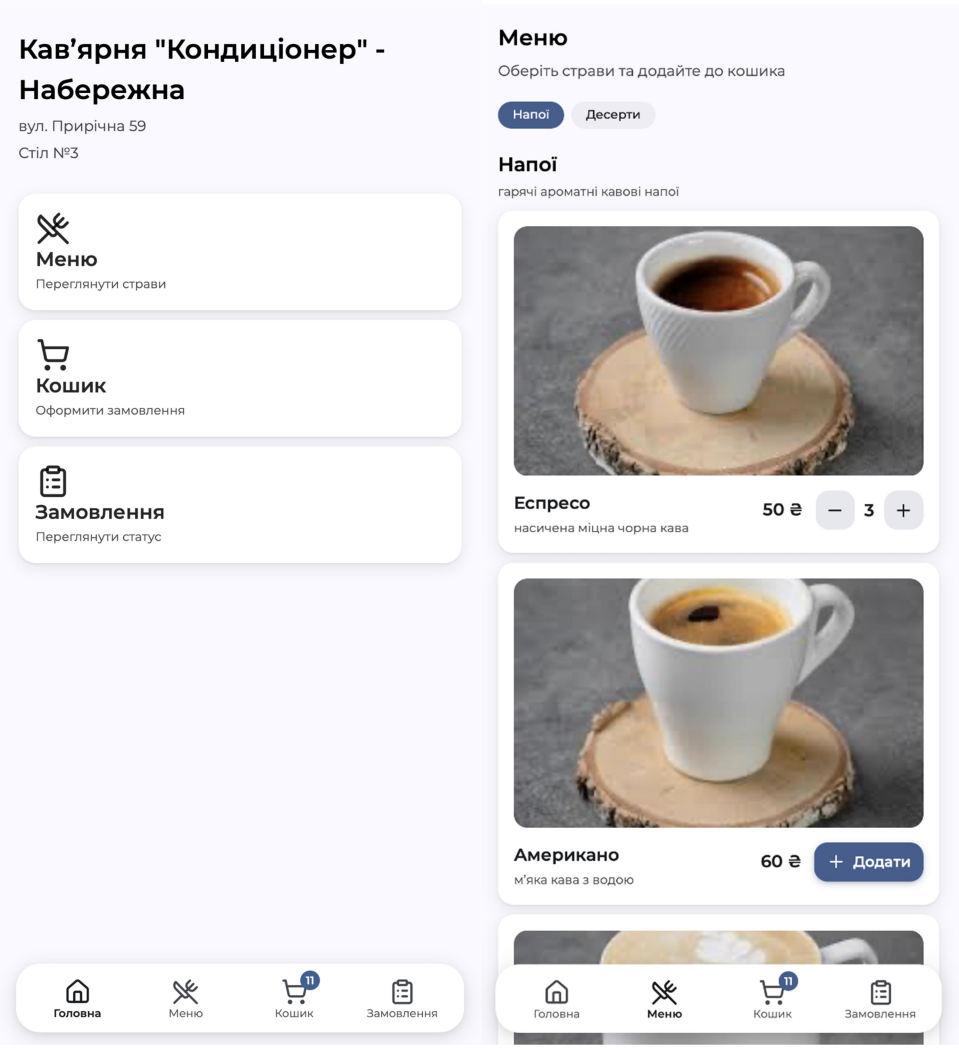

Рис.3. "Головна" сторінка та сторінка "Меню"

Зміни, внесені в панелі (ціни, наявність страв, статуси замовлень), відображаються в QR-меню завдяки спільній базі даних і узгодженій бізнес-логіці в ядрі. Модуль аналітики візуалізує обсяги продажів і прибуток, що спрощує прийняття оперативних рішень для керівництва точки або мережі.

Під час тестування проведено перевірку обох інтерфейсів, коректність API, синхронізацію змін між клієнтом і панеллю та обробку некоректних даних. Для QR-меню підтверджено перехід за QR з прив'язкою до закладу й столу, завантаження категорій і товарів, роботу кошика й створення замовлення; для адмін-панелі - CRUD товарів і категорій, зміну статусів замовлень, генерацію QR для столів, запрошення користувачів і дотримання обмежень ролей. При помилкових вводах система повертає зрозумілі повідомлення без збереження некоректного стану.

Орієнтовні показники продуктивності: перше завантаження 1-2 с, навігація між розділами 200-500 мс, відповідь API 100-300 мс (складні операції адмін-панелі до 400-600 мс), читання з БД 50-150 мс, запис до 200

мс. Навантаження з кількох клієнтів одночасно не призводило до помітного зниження продуктивності. Контейнеризація Docker [14] забезпечує відтворюване розгортання. З точки зручності (UX) клієнтський сценарій мінімізує кількість кроків і не вимагає реєстрації; адмін-панель має наскрізну навігацію й оновлення даних без повного перезавантаження сторінки. Окремо перевірялись узгодженість меню після змін у панелі, коректність підрахунку сум у кошику та збереження коментарів до замовлення; виявлені дрібні невідповідності інтерфейсу усунено без зміни архітектури. Архітектура з нормалізованою БД і чіткими ролями дозволяє нарощувати кількість закладів і користувачів без перепроєктування базової моделі даних. Функціональні та нефункціональні вимоги виконано; рішення придатне для закладів ресторанного бізнесу.

ВИСНОВКИ

У межах дослідження розроблено веб-додаток із QR-інтерфейсом для відвідувачів та адміністративною панеллю для персоналу. Наукова новизна полягає у запропонованій архітектурі з двома серверними застосунками та спільним ядром із мінімізацією дублювання логіки, а також у моделі мережі закладів із індивідуальними цінами та доступністю товарів для кожної точки, що відповідає реальним сценаріям функціонування франшиз і мереж, де асортимент і ціни відрізняються між закладами. Практичне значення полягає в можливості впровадження без залежності від стороннього SaaS: заклад отримує повний контроль над даними, ролями та розвитком функціоналу. Результати тестів підтверджують стабільність і достатній рівень продуктивності для щоденної експлуатації. Запропоноване рішення може слугувати основою для подальших наукових і практичних робіт у галузі автоматизації громадського харчування. Перспективи: інтеграція онлайн-оплати (зокрема Monobank Pay, Apple Pay, Google Pay), розширена кастомізація вигляду меню, публічні сторінки закладів, поглиблена аналітика та інтеграція з POS-системами.

ДЖЕРЕЛА

1. Martin R. C. Clean Code: A Handbook of Agile Software Craftsmanship. Boston : Prentice Hall, 2009. 464 p.

2. Sommerville I. Software Engineering. 10th ed. Boston : Pearson, 2016. 816 p.

3. Fowler M. Patterns of Enterprise Application Architecture. Boston : Addison-Wesley, 2002. 533 p.

4. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Boston : Addison-Wesley, 1994. 395 p.

5. Freeman E., Robson E. Head First Design Patterns. Sebastopol : O'Reilly, 2020. 694 p.

6. ChoiceQR. Онлайн QR меню для закладів [Електронний ресурс]. 2025. URL: https://choiceqr.com/uk/ (дата звернення: 02.05.2026).

7. SMART SPOT - інтерактивне QR-меню для ресторанів [Електронний ресурс]. 2025. URL: https://expertsolution.com.ua/statti/interaktyvne-qr-menyu-smart-spot/ (дата звернення: 02.05.2026).

8. Печериця, В. В., Бондарчук, А. П., & Замрій, І. В. (2021). Розробка методики прототипування об'єктів інформаційної системи на базі технології Java Script, Node. JS. Телекомунікаційні та інформаційні технології, (4), 12-19.

9. Expirenza by mono [Електронний ресурс]. 2026. URL: https://expz.monobank.ua (дата звернення: 02.05.2026).

10. Vue.js Documentation [Електронний ресурс]. 2025. URL: https://vuejs.org/ (дата звернення: 02.05.2026).

11. ASP.NET Core Documentation [Електронний ресурс]. 2025. URL: https://learn.microsoft.com/aspnet/core (дата звернення: 02.05.2026).

12. Entity Framework Core Docs [Електронний ресурс]. 2025. URL: https://learn.microsoft.com/ef/core (дата звернення: 02.05.2026).

13. PostgreSQL Documentation [Електронний ресурс]. 2025. URL: https://www.postgresql.org/docs/ (дата звернення: 02.05.2026).

14. Docker Documentation [Електронний ресурс]. 2025. URL: https://docs.docker.com/ (дата звернення: 02.05.2026).

15. Vite Documentation [Електронний ресурс]. 2025. URL: https://vitejs.dev/ (дата звернення: 02.05.2026).

16. Axios GitHub [Електронний ресурс]. 2025. URL: https://github.com/axios/axios (дата звернення: 02.05.2026).

17. BCrypt Documentation [Електронний ресурс]. 2025. URL: https://github.com/kelektiv/node.bcrypt.js (дата звернення: 02.05.2026).