Полтавський університет економіки і торгівлі

Кафедра комп'ютерних наук та інформаційних технологій

КОМП'ЮТЕРНІ НАУКИ ТА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ (КНІТ-2026)

МАТЕРІАЛИ НАУКОВО-ПРАКТИЧНОГО СЕМІНАРУ

Випуск 5

Вересень 2025 р. - червень 2026 р.

Присвячено 65-річчю ПУЕТ

Полтава 2026

КОМП'ЮТЕРНІ НАУКИ ТА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ (КНІТ-2026): матеріали науково-практичного семінару. Випуск 5 / за ред. О.В. Ольховської – Полтава: Кафедра КНІТ ПУЕТ, 2026. – 54 с.

Збірник матеріалів науково-практичного семінару містить добірку праць присвячених актуальній проблематиці, що висвітлює питання галузі сучасних інформаційних технологій, кібернетики, інформатики, математичного моделювання, системного аналізу, програмного забезпечення інформаційних систем та теорії прийняття оптимальних рішень.

У збірці представлено матеріали, що відображають проблематику підготовки фахівців з комп'ютерних наук та інформаційних технологій.

Всі публікації викладено в авторській редакції.

Збірник присвячено 65-річчю університету.

Ум. друк. арк. 3,4

ЗМІСТ

УДК 004.9:37.018.43

РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ЕЛЕКТРОННОГО ЖУРНАЛУ ВИКЛАДАЧА ІЗ РОЗШИРЕНИМ ФУНКЦІОНАЛОМ АНАЛІТИКИ ТА ЗВІТНОСТІ

***К. Г. Крамаренко**, студент спеціальності «Комп'ютерні науки», група КН б-41*
***Т. О. Парфьонова**, кандидат фізико-математичних наук, доцент кафедри комп'ютерних наук та інформаційних технологій*
Полтавський університет економіки і торгівлі

Ключові слова: ЕЛЕКТРОННИЙ ЖУРНАЛ, ІНФОРМАЦІЙНА СИСТЕМА, АНАЛІТИКА ДАНИХ, ОСВІТНІ ТЕХНОЛОГІЇ, ОБЛІК УСПІШНОСТІ

Keywords: ELECTRONIC GRADEBOOK, INFORMATION SYSTEM, DATA ANALYTICS, EDUCATIONAL TECHNOLOGIES, STUDENT PERFORMANCE

У сучасних умовах цифрової трансформації освіти важливим напрямом розвитку є впровадження інформаційних систем, що забезпечують автоматизацію процесів управління навчальною діяльністю. Особливу увагу приділяють системам обліку відвідуваності та успішності студентів, які традиційно ведуться у паперовому вигляді, що ускладнює доступ до інформації, її аналіз та збереження.

Метою роботи є розробка інформаційної системи електронного журналу викладача з розширеним функціоналом аналітики та звітності, яка дозволяє автоматизувати процес ведення обліку навчальних досягнень студентів і підвищити ефективність освітнього процесу.

У ході дослідження проаналізовано сучасні підходи до створення електронних освітніх систем, визначено основні функціональні та нефункціональні вимоги до програмного продукту. Запропонована система передбачає реалізацію таких можливостей:

• ведення електронного журналу відвідуваності студентів;

• внесення та редагування оцінок за різні види навчальної діяльності;

• автоматичне обчислення середнього балу та рейтингу студентів;

• формування звітів за заданими параметрами (група, дисципліна, період навчання);

• візуалізація статистичних даних у вигляді графіків та діаграм;

• збереження історії змін та можливість аудиту даних.

Архітектурно система реалізована як клієнт-серверний застосунок із використанням сучасних технологій програмування. Для реалізації логіки застосунку може бути використано мову програмування Python або JavaScript (з використанням відповідних фреймворків), а для зберігання даних — реляційну базу даних (наприклад, SQLite або PostgreSQL). Особливу увагу приділено забезпеченню цілісності даних, безпеці доступу та можливості масштабування системи.

Інтерфейс користувача розроблено з урахуванням принципів зручності та доступності. Викладач отримує можливість швидко вносити дані, переглядати статистику та формувати звітність без необхідності виконання складних дій. Наявність аналітичного модуля дозволяє оцінювати динаміку успішності студентів та приймати обґрунтовані педагогічні рішення.

Результати впровадження системи свідчать про підвищення ефективності роботи викладача за рахунок скорочення часу на обробку даних та покращення якості контролю навчального процесу. Використання електронного журналу також сприяє підвищенню прозорості оцінювання та доступності інформації для всіх учасників освітнього процесу.

У подальшому передбачається розширення функціональних можливостей системи, зокрема інтеграція з іншими освітніми платформами, впровадження мобільної версії застосунку, а також використання технологій машинного навчання для прогнозування успішності студентів.

Таким чином, розроблена інформаційна система електронного журналу викладача є ефективним інструментом цифровізації освітнього процесу та може бути використана як у закладах вищої освіти, так і в інших навчальних установах.

***Література*:**

1. Sommerville I. Software Engineering. – Boston: Pearson, 2016.
2. Pressman R. Software Engineering: A Practitioner's Approach. – New York: McGraw-Hill, 2014.
3. Python documentation. [Електронний ресурс]. – URL: https://docs.python.org/
4. PostgreSQL documentation. [Електронний ресурс]. – URL: https://www.postgresql.org/docs/
5. MDN Web Docs. [Електронний ресурс]. – URL: https://developer.mozilla.org/
6. ISO/IEC 25010:2011 Systems and software quality models.

УДК 004.42:004.65

АРХІТЕКТУРНІ ТА ТЕХНОЛОГІЧНІ АСПЕКТИ РОЗРОБКИ ВЕБЗАСТОСУНКУ ВЕТЕРИНАРНОЇ КЛІНІКИ ЗАСОБАМИ ASP.NET Core MVC

Д. М. Кришталь, *студент, КНб-41* «Полтавський університет економіки і торгівлі»
denkryshtal73737r74@gmail.com
О. П. Кошова*, к. пед. н., доцент кафедри комп'ютерних наук та інформаційних технологій.*
koshova.o111@gmail.com
Полтавський університет економіки і торгівлі

У статті розглянуто особливості проєктування та програмної реалізації вебзастосунку ветеринарної клініки на основі технологій ASP.NET Core MVC, Entity Framework Core та Microsoft SQL Server.

Kryshtal D., Koshova O. Architectural and technological aspects of developing a veterinary clinic web application using ASP.NET Core MVC. The article considers the peculiarities of designing and implementing a veterinary clinic web application based on ASP.NET Core MVC, Entity Framework Core and Microsoft SQL Server technologies.

Ключові слова: *ВЕБЗАСТОСУНОК, ASP.NET CORE MVC, ENTITY FRAMEWORK CORE, ВЕТЕРИНАРНА КЛІНІКА, ІНФОРМАЦІЙНА СИСТЕМА, ВЕБТЕХНОЛОГІЇ.*
Keywords: WEB APPLICATION, ASP.NET CORE MVC, ENTITY FRAMEWORK CORE, VETERINARY CLINIC, INFORMATION SYSTEM, WEB TECHNOLOGIES.

Сучасний розвиток інформаційних технологій супроводжується активним впровадженням цифрових сервісів у різні сфери діяльності. Особливого значення набуває автоматизація процесів у медичних та ветеринарних установах, де необхідно забезпечити

ефективне зберігання інформації, взаємодію з клієнтами та підтримку адміністративних процесів. Використання веборієнтованих інформаційних систем дозволяє централізувати дані, підвищити оперативність обслуговування та створити єдине інформаційне середовище для користувачів і персоналу установи.

У наукових дослідженнях відзначається важливість використання сучасних інформаційних технологій для автоматизації діяльності ветеринарних клінік та покращення якості надання послуг [4]. Водночас зростає потреба у створенні спеціалізованих систем, які поєднують інформаційні, сервісні та адміністративні функції. У роботах, присвячених розробці платформ онлайн-запису та систем управління ветеринарними закладами, наголошується на необхідності інтеграції механізмів взаємодії з клієнтами, засобів адміністрування та централізованого зберігання даних у межах єдиної програмної системи [5, 6].

Метою роботи є дослідження архітектурних і технологічних особливостей розробки вебзастосунку ветеринарної клініки на основі сучасних технологій платформи .NET та аналіз реалізованих програмних рішень.

Однією з ключових вимог до сучасних веборієнтованих систем є забезпечення масштабованості, підтримуваності та можливості подальшого розвитку функціоналу. Для досягнення цих вимог під час розробки було обрано архітектурний шаблон Model-View-Controller, який є одним із найбільш поширених підходів до створення вебзастосунків [3]. Архітектура MVC забезпечує розподіл відповідальності між окремими компонентами системи, що дозволяє ізолювати логіку роботи з даними від інтерфейсу користувача та бізнес-логіки.

Для реалізації серверної частини використано платформу ASP.NET Core MVC, яка забезпечує високу продуктивність, кросплатформність та підтримку сучасних принципів розробки програмного забезпечення [1]. Основною мовою програмування виступає C#, що дозволяє ефективно реалізовувати багаторівневу архітектуру застосунку та використовувати широкий набір інструментів екосистеми .NET. При цьому використання сучасних вебтехнологій і багаторівневих архітектурних рішень є одним із ключових напрямів розвитку вебзастосунків, що підтверджується

результатами досліджень у галузі розробки вебсервісів та програмних платформ [7].

Важливою складовою системи є рівень доступу до даних. Для його реалізації використано Entity Framework Core, який забезпечує механізм об'єктно-реляційного відображення та дозволяє працювати з базою даних через програмні моделі [2]. Застосування ORM-технології спрощує реалізацію операцій створення, отримання, оновлення та видалення даних, а також підвищує читабельність програмного коду. Додатково для формування вибірок та обробки інформації використовується LINQ, що забезпечує зручний механізм роботи з колекціями та сутностями.

Архітектура розробленого вебзастосунку передбачає взаємодію трьох основних рівнів: рівня представлення, рівня бізнес-логіки та рівня доступу до даних. Рівень представлення реалізовано засобами Razor, HTML5, CSS3, JavaScript і Bootstrap. Рівень бізнес-логіки забезпечує обробку запитів користувачів, реалізацію функціональних сценаріїв та взаємодію між контролерами і моделями. Рівень доступу до даних відповідає за роботу з базою даних Microsoft SQL Server та забезпечує централізоване зберігання інформації.

Для підвищення гнучкості програмного забезпечення використано Repository Pattern. Даний підхід дозволяє відокремити механізми взаємодії з базою даних від інших компонентів системи та спрощує подальшу модифікацію програмного забезпечення. Використання подібних архітектурних рішень відповідає сучасним рекомендаціям щодо розробки масштабованих вебзастосунків [3].

Структура бази даних охоплює основні сутності предметної області. У системі реалізовано зберігання інформації про працівників клініки, ветеринарні послуги, записи на прийом, категорії, публікації, коментарі, теги, навігаційні елементи та параметри вебресурсу. Такий підхід забезпечує централізоване керування інформацією та підтримує цілісність даних у межах усієї системи.

Важливе місце у вебзастосунку займає користувацький інтерфейс, який повинен забезпечувати швидкий доступ до необхідної інформації та зручну навігацію між функціональними

модулями. Загальний вигляд головної сторінки вебзастосунку наведено на рисунку 1.

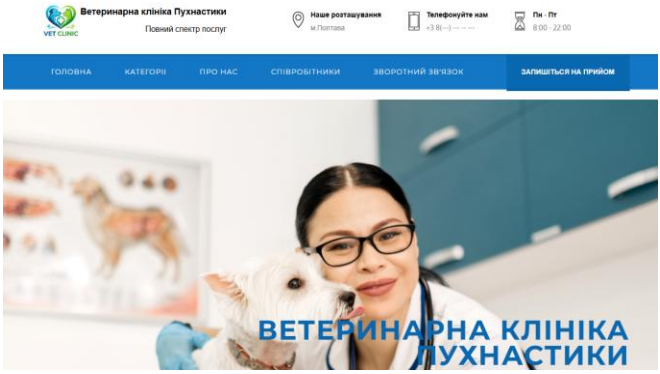

Рисунок 1. Головна сторінка вебзастосунку ветеринарної клініки

Інтерфейс побудований на основі спільного макета сторінок, що забезпечує єдину структуру оформлення для всіх розділів сайту. Використання компонентного підходу дозволяє повторно використовувати однакові елементи інтерфейсу, зокрема логотип, меню навігації, контактну інформацію та інформаційні блоки. Це позитивно впливає на підтримуваність програмного забезпечення та спрощує його подальший розвиток.

Однією з ключових функціональних можливостей системи є представлення ветеринарних послуг. Для цього реалізовано окремий модуль, який забезпечує динамічне відображення інформації про послуги, що зберігаються у базі даних. Приклад реалізації відповідного функціоналу наведено на рисунку 2.

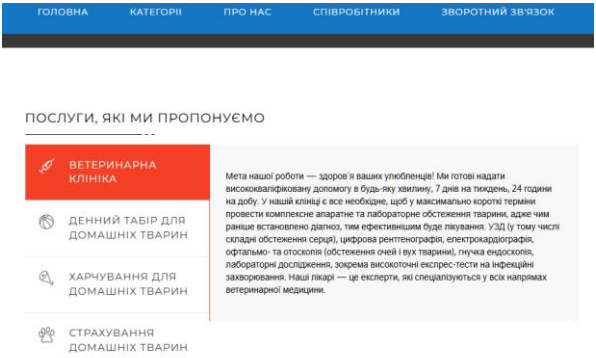

Рисунок 2. Секція послуг вебзастосунку

Особливістю реалізації є динамічне формування інтерфейсу на основі даних, отриманих із серверної частини застосунку. Завдяки цьому будь-які зміни, виконані адміністратором через панель керування, автоматично відображаються у публічній частині сайту без необхідності втручання в програмний код.

Важливим елементом вебзастосунку є модуль публікацій, який використовується для розміщення новин, інформаційних матеріалів та рекомендацій щодо догляду за тваринами. Даний функціонал сприяє підвищенню інформативності ресурсу та покращує взаємодію між клінікою і відвідувачами сайту. Приклад сторінки публікацій наведено на рисунку 3.

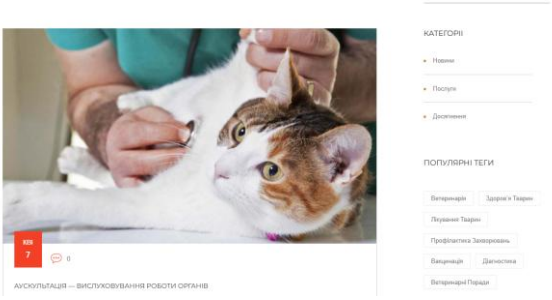

Рисунок 3. Сторінка публікацій вебзастосунку

Одним із найбільш важливих сервісних модулів системи є

механізм запису на прийом. Необхідність реалізації такого функціоналу підтверджується сучасними дослідженнями у сфері ветеринарних інформаційних систем, де онлайн-запис розглядається як одна з базових складових цифрового сервісу [5]. У розробленому вебзастосунку форма запису інтегрована безпосередньо до структури головної сторінки, що дозволяє зменшити кількість переходів між сторінками та спрощує взаємодію користувача із системою.

Для забезпечення коректності введених даних застосовано механізми валідації ASP.NET Core. Перевірка правильності введення інформації виконується на рівні моделей за допомогою DataAnnotations, що дозволяє попереджати помилки ще до моменту збереження інформації в базі даних. Такий підхід підвищує надійність роботи системи та забезпечує цілісність даних.

Окрему увагу приділено реалізації адміністративної частини вебзастосунку. На відміну від публічного інтерфейсу, її основним призначенням є керування даними та адміністрування функціональних модулів системи. Через адміністративну панель здійснюється управління послугами, працівниками, категоріями, публікаціями, коментарями та записами на прийом.

Результат реалізації адміністративної підсистеми наведено на рисунку 4.

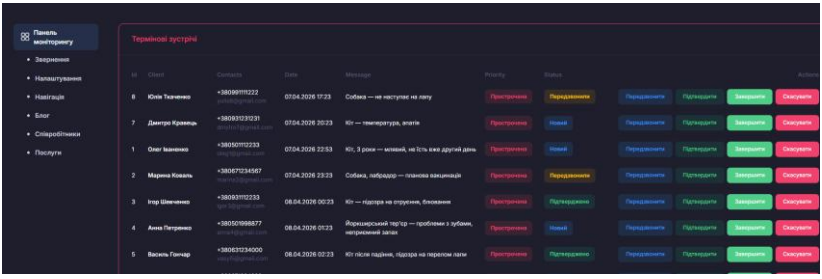

Рисунок 4. Адміністративна панель керування записами на прийом

Інтерфейс адміністративної панелі дозволяє групувати записи відповідно до їхнього статусу та часу виконання, що спрощує

планування роботи клініки та забезпечує оперативний контроль звернень клієнтів. Використання подібного підходу дозволяє автоматизувати значну частину адміністративних процесів і підвищити ефективність роботи персоналу.

Проведений аналіз показує, що реалізований вебзастосунок відповідає сучасним тенденціям розвитку ветеринарних інформаційних систем. На відміну від традиційних сайтів-візитівок, він поєднує інформаційні, сервісні та адміністративні функції в межах єдиного програмного середовища. Подібний підхід узгоджується з результатами сучасних досліджень, присвячених розробці інформаційних систем для ветеринарних установ [4–6].

Отже, використання ASP.NET Core MVC, Entity Framework Core та сучасних підходів до проєктування вебзастосунків дозволило створити масштабовану інформаційну систему для ветеринарної клініки. Реалізоване програмне забезпечення забезпечує централізоване керування даними, підтримку запису на прийом, представлення послуг і публікацій, а також повноцінне адміністрування основних об'єктів системи. Отримані результати підтверджують ефективність використання технологій платформи .NET для побудови сучасних веборієнтованих інформаційних систем.

Література:

1. ASP.NET Core MVC with EF Core. Режим доступу: https://learn.microsoft.com/en-us/aspnet/core/data/ef-mvc/?view=aspnetcore-10.0.
2. Overview of Entity Framework Core - EF Core. Режим доступу: https://learn.microsoft.com/en-us/ef/core/.
3. Common web application architectures - .NET. Режим доступу: https://learn.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures.
4. Application of information technologies in veterinary medicine. Режим доступу: https://nvlvet.com.ua/index.php/journal/article/view/5462.
5. An Online Scheduling Platform for Veterinary Appointments.

Режим доступу: https://journal.uitm.edu.my/ojs/index.php/JI/article/view/6631.
6. The Development of Veterinary Clinic Management Application System. Режим доступу: https://publisher.uthm.edu.my/periodicals/index.php/aitcs/article/download/2339/1370/28224.
7. Кошова О. П., Ольховська О. В., Тацій Д. С., Олексійчук Ю. Ф., Черненко О. О. Розробка веб-додатків та сервісів на платформі Node.js // Таврійський науковий вісник. Серія: Технічні науки. 2023. Вип. 2. С. 78–89. Режим доступу: https://journals.ksauniv.ks.ua/index.php/tech/issue/view/26 < https://journals.ksauniv.ks.ua/index.php/tech/article/view/358/331>

УДК 004.738.5

ВЕБ-ПОРТАЛ ДЛЯ ОНЛАЙН-БРОНЮВАННЯ СПОРТИВНИХ МАЙДАНЧИКІВ І ЗАЛІВ НА ОСНОВІ СТЕКУ REACT І NODE.JS

***С. В. Лянник**, спеціальність «Комп'ютерні науки», група КН б-41;*
***О. П. Кошова**, доцент кафедри комп'ютерних наук та інформаційних технологій, канд. пед. наук, доцент*
Полтавський університет економіки і торгівлі

Запропоновано клієнт-серверний веб-портал «Sport Booking Portal» для онлайн-бронювання спортивних майданчиків і залів, що забезпечує повний цикл взаємодії з кінцевим користувачем — від пошуку відповідного об'єкта у каталозі до перевірки доступності часових слотів, формування замовлення, мокової оплати та подальшого керування активними бронюваннями. Розробка спеціалізованого програмного продукту для предметної області спортивних послуг є актуальною з огляду на стрімку цифрову трансформацію галузі та обмеженість наявних рішень — комерційні агрегатори утримують комісії з кожного замовлення, а універсальні платформи запису не враховують специфіку слотової моделі обмежених ресурсів і не унеможливлюють подвійних бронювань на ту саму годину [1].

Архітектура побудована за класичним трирівневим принципом з чітким поділом на рівень подання, бізнес-логіки та даних і реалізована у форматі монорепозиторію на основі pnpm workspaces. Клієнтська частина являє собою односторінковий веб-застосунок (SPA) на бібліотеці React 19 з мовою програмування TypeScript та збирачем Vite [2]. Серверна частина побудована на платформі Node.js з фреймворком Express 5 та ORM Prisma 6, що працює з реляційною базою даних PostgreSQL 16 [3][4]. Реляційна модель складається із семи сутностей (User, RefreshToken, Venue, VenueImage, VenueSchedule, Booking, MockPayment) і покриває облік користувачів, спортивних об'єктів, тижневого розкладу, бронювань та мокових платіжних транзакцій.

Модуль автентифікації реалізовано на основі стандарту JSON Web Token (RFC 7519) із застосуванням схеми двох токенів —

короткоживучого access-токена з часом життя вісім годин та довгоживучого refresh-токена із семиденним терміном дії та автоматичною ротацією при кожному використанні [5]. Обидва токени зберігаються лише в HttpOnly cookie з атрибутом sameSite: «lax», що виключає доступ із клієнтського JavaScript і знижує ризик атак XSS та CSRF. У базі зберігаються не самі refresh-токени, а їхні SHA-256-хеші, що унеможливлює відновлення оригінального значення у разі компрометації сховища. Паролі хешуються за алгоритмом bcrypt із десятьма раундами; на маршрутах реєстрації та входу застосовано обмеження частоти запитів через бібліотеку express-rate-limit для запобігання атакам перебору.

Центральний модуль бронювання реалізує алгоритм перевірки доступності часових слотів у межах транзакції PostgreSQL для забезпечення атомарності. Алгоритм виконує послідовно завантаження майданчика з тижневим розкладом, перевірку відповідності інтервалу робочим годинам і пошук конфліктних бронювань за подвійною умовою перетину інтервалів startTime < newEndTime AND endTime > newStartTime з фільтром за дозволеними статусами. Транзакційна модель гарантує відсутність гонитви даних (race condition) при одночасних запитах від різних користувачів і повністю унеможливлює подвійне бронювання одного об'єкта на однаковий інтервал часу. Валідація вхідних даних на всіх маршрутах виконується бібліотекою Zod, що автоматично нормалізує значення та формує структуровані повідомлення про помилки.

Клієнтська частина організована як SPA з маршрутизацією на React Router 7 і централізованим управлінням сесією через контекст AuthContext, що автоматично відновлює авторизацію при перезавантаженні сторінки через захищений ендпоінт /auth/me. Доступ до приватних і адміністративних маршрутів захищено компонентом-обгорткою ProtectedRoute з прапорцем requireAdmin для рольового розмежування. Реалізована мокова платіжна система з трьома сценаріями завершення (успіх, скасування, відхилення) дозволяє повністю емулювати поведінку реального платіжного шлюзу без здійснення фактичних транзакцій і архітектурно готова до заміни на інтеграцію зі Stripe, LiqPay або

Fondy без зміни моделі даних. Адміністративна панель надає повноцінний інтерфейс керування каталогом об'єктів, тижневими розкладами, зображеннями та бронюваннями.

Коректність ключових сценаріїв бізнес-логіки верифіковано набором автоматизованих тестів на основі вбудованого тестового модуля Node.js (node:test) із застосуванням бібліотеки supertest та ізольованої тестової бази даних. Покрито сценарії реєстрації нового користувача, заборони дублювання адреси електронної пошти, перетину часових слотів при одночасних бронюваннях та повного циклу мокової оплати з трьома результатами. Розгортання усіх компонентів системи (клієнт, сервер, база даних) виконується через Docker Compose з автоматичним застосуванням міграцій схеми та заповненням бази тестовими даними при першому старті, що відповідає принципу 12-Factor App та забезпечує відтворюваність середовища.

Висновок. Спроєктований і реалізований веб-портал «Sport Booking Portal» поєднує сучасний типобезпечний стек технологій (React 19, TypeScript, Node.js з Express 5, Prisma 6, PostgreSQL 16), наскрізне дотримання практик безпеки (ротація refresh-токенів, HttpOnly cookie, bcrypt-хешування, обмеження частоти запитів, серверна валідація через Zod) та надійний алгоритм перевірки доступності слотів у транзакційному режимі. Розроблена відкрита архітектура, що повністю контролюється власником, придатна для впровадження у діяльність фітнес-центрів, спортивних клубів та муніципальних спортивних установ як легковагова безкомісійна альтернатива комерційним агрегаторам типу Playtomic або Booksy, а закладена в схему даних модель платежів забезпечує можливість безболісної міграції на реальні платіжні шлюзи без зміни бізнес-логіки серверних модулів.

Література:

1. Fielding R. T. Architectural Styles and the Design of Network-based Software Architectures. University of California, Irvine, 2000. URL: https://ics.uci.edu/~fielding/pubs/dissertation/top.htm
2. React Documentation. URL: https://react.dev/
3. Express. Fast, unopinionated, minimalist web framework for Node.js. URL: https://expressjs.com/

4. Prisma Documentation. URL: https://www.prisma.io/docs
5. Jones M., Bradley J., Sakimura N. RFC 7519: JSON Web Token (JWT). URL: https://datatracker.ietf.org/doc/html/rfc7519
6. PostgreSQL 16 Documentation. The PostgreSQL Global Development Group, 2023. URL: https://www.postgresql.org/docs/16/index.html
7. OWASP Foundation. OWASP Top Ten Web Application Security Risks. URL: https://owasp.org/www-project-top-ten/

УДК 004.738.5:519.854

ІНТЕРАКТИВНИЙ ВЕБЗАСТОСУНОК ДЛЯ ВИВЧЕННЯ МЕТОДУ ГІЛОК ТА МЕЖ У ЗАДАЧАХ ОПТИМІЗАЦІЇ НА ПЕРЕСТАВЛЕННЯХ

***Н. О. Малахов**, спеціальність «Комп'ютерні науки», група КН б-41;*
***Т. В. Чілікіна**, доцент кафедри математичного моделювання та соціальної інформатики, канд. фіз.-мат. наук, доцент*
Полтавський університет економіки і торгівлі

Запропоновано навчальний вебзастосунок «Permutex» для вивчення теми «Метод гілок та меж для задач оптимізації на переставленнях» курсу «Економіко-математичні методи». Метод гілок та меж є класичним інструментом точного розв'язання задач комбінаторної оптимізації, а задачі на переставленнях — характерним економіко-математичним об'єктом, що моделює розподіл обсягів виробництва, транспортні задачі з дискретними обсягами перевезень та маршрутизацію [1]. Опанування цієї теми ускладнюється рекурсивним характером алгоритму, великою кількістю формальних позначень і відсутністю україномовних інтерактивних інструментів для відпрацювання обчислень оцінок ν, ξ та правил відсікання.

Архітектура застосунку побудована як чотирирівнева система з відокремленням обчислювального ядра, логіки тренажерів, сховища стану та користувацького інтерфейсу. Реалізація виконана на TypeScript у фреймворку Next.js 16 з React 19, App Router та Turbopack [2]. Для візуалізації дерева галуження застосовано бібліотеку React Flow (@xyflow/react), для рендерингу математичних формул — KaTeX, для управління станом — Zustand [3]. Користувацький інтерфейс реалізовано на основі shadcn/ui поверх Tailwind CSS 4 з підтримкою світлої та темної тем оформлення.

Обчислювальне ядро покриває чотири типи задач на переставленнях: безумовну лінійну задачу, лінійну задачу з обмеженнями, комбінаторну транспортну задачу та задачу про найкоротший маршрут у зваженому графі [1]. Обчислення оцінок ν

і ξ виконано за теоремами 2 і 3 лекції 7 — з використанням нерівності про перестановки [4] для побудови мінімального скалярного добутку c^*. Для комбінаторної транспортної задачі реалізовано правила A і B з теорем 4 і 5, що дозволяє відсікати інфізибельні бруньки одразу після їх породження.

Ключовим нововведенням є інтерактивний тренажер — окремий компонент, у якому студент сам ухвалює всі рішення алгоритму, а програма перевіряє кожен крок. Машина станів проводить через п'ять фаз: вибір активної бруньки з фронту, фіксацію значення з мультимножини G, введення оцінки ν, введення оцінки ξ та ухвалення рішення про відсікання чи продовження галуження. На кожній фазі двигун обчислює канонічну відповідь і порівнює її з введенням студента, ведучи лічильник правильних, помилкових відповідей і використаних підказок. Реалізовано окремі тренажери для всіх чотирьох типів задач — з урахуванням матричної структури транспортної задачі та графової структури маршруту.

Інтерфейс підтримує покрокову навігацію по дереву галуження з кнопками вперед/назад, режимом автозапуску та клавіатурними скороченнями [5]. Кожен крок алгоритму супроводжується текстовим поясненням українською мовою з вставленими формулами через KaTeX. Передбачено готові приклади з лекцій 6 і 7 для відтворення еталонних оптимумів $F = 27$, $F = 4$, $F_0 = 203$ та $F_0 = 240$, а також генератор випадкових задач, режим самоперевірки Quiz і покроковий онбординг для нових користувачів.

Обчислювальне ядро та логіка тренажерів покриті 57 модульними тестами на Vitest, які виконуються за час менше 1 секунди [6]. Експериментальне тестування ефективності методу гілок та меж проведене для розмірностей $k = 3..9$: для $k = 9$ кількість перевірених листків зменшується із $9! = 362\,880$ до приблизно 124, що відповідає економії 99,97 %. Усі обчислення виконуються у браузері без сервера, що робить застосунок зручним для використання як на лекційних, так і на самостійних заняттях.

Висновок.Розроблений вебзастосунок «Permutex» забезпечує інтерактивне навчання методу гілок та меж для всіх чотирьох

типів задач на переставленнях курсу «Економіко-математичні методи» з україномовним інтерфейсом, покроковою візуалізацією та тренажером для самостійної побудови дерева пошуку. Поєднання демонстраційного та тренажерного режимів суттєво підвищує глибину засвоєння алгоритму порівняно з традиційними навчальними матеріалами. Застосунок може бути впроваджений у навчальний процес кафедри математичного моделювання та соціальної інформатики ПУЕТ як інструмент для лекційних демонстрацій, самостійної роботи студентів та підготовки до контрольних робіт.

Література:

1. Ємець О. О. Економіко-математичні методи: лекції 6-7. Метод гілок та меж для задач оптимізації на переставленнях. – Полтава : ПУЕТ, 2024. – 36 с.
2. Next.js Documentation. URL: https://nextjs.org/docs
3. Zustand — Bear necessities for state management. URL: https://zustand.docs.pmnd.rs/
4. Hardy G. H., Littlewood J. E., Pólya G. Inequalities: rearrangement inequality. Cambridge Mathematical Library, 2nd ed. URL: https://en.wikipedia.org/wiki/Rearrangement_inequality
5. React Flow (xyflow) Documentation. URL: https://reactflow.dev/learn
6. Vitest — Next Generation Testing Framework. URL: https://vitest.dev/guide/

УДК 004.738.5:339.178

ВЕБ-САЙТ ЕЛЕКТРОННОГО МАГАЗИНУ КНИГ З ІНТЕГРАЦІЄЮ ПЛАТІЖНИХ СЕРВІСІВ

***Д. Д. Матвєєнко**, спеціальність «Комп'ютерні науки», група КН б-41;*
***О. В. Ольховська**, завідувач кафедри комп'ютерних наук та інформаційних технологій, канд. фіз.-мат. наук, доцент*
Полтавський університет економіки і торгівлі

Запропоновано веб-сайт електронного магазину книг «BookNest» з інтеграцією платіжних сервісів, що забезпечує повний цикл онлайн-продажу — від перегляду каталогу до оплати замовлення та його адміністрування. Електронна комерція є однією з найдинамічніших галузей цифрової економіки, а книжкова ніша особливо органічна для онлайн-формату, оскільки книга є стандартизованим товаром із чіткими атрибутами, зручним для каталогізації та порівняння. Ключовим елементом інтернет-магазину є інтеграція платіжного сервісу, адже саме механізм онлайн-оплати перетворює каталог товарів на повноцінний інструмент продажу [1].

Систему побудовано як монорепозиторій з розподілом на клієнтський застосунок на фреймворку Next.js та серверне REST API на фреймворку NestJS з мовою TypeScript. Сховищем даних обрано реляційну СУБД PostgreSQL з ORM Prisma; схему нормалізовано до третьої нормальної форми та описано у вигляді типізованих моделей із версіонованими міграціями [2][3]. Уся інфраструктура магазину — клієнтський застосунок, серверне API, фоновий обробник черг, база даних, кеш, пошуковий рушій, об'єктне сховище та поштовий сервіс — розгортається у вигляді контейнерів Docker, описаних у єдиному декларативному файлі, що забезпечує відтворюваність середовища [4].

Модуль каталогу реалізує перегляд книг із фільтрацією за категорією, автором, ціною, мовою та наявністю, сортуванням і пагінацією; запити до бази виконуються параметризованими викликами ORM, що виключає SQL-ін'єкції. Повнотекстовий пошук делеговано окремому пошуковому рушію Meilisearch з

толерантністю до помилок у запитах, завдяки чому користувач отримує релевантні результати навіть за наявності одруку. Початкове наповнення каталогу виконує сценарій, що звертається до відкритого API Open Library для отримання реальних метаданих та обкладинок книг [5].

Модуль автентифікації побудовано за схемою двох токенів JSON Web Token (RFC 7519): короткоживучого access-токена та довгоживучого refresh-токена з ротацією при кожному оновленні, що дозволяє виявляти повторне використання викраденого токена. Паролі зберігаються у вигляді незворотного хешу, обчисленого алгоритмом Argon2id, а refresh-токени — у вигляді криптографічного хешу. Токени передаються у захищених cookie з атрибутами HttpOnly та SameSite; додатково застосовано захисні HTTP-заголовки, обмеження частоти запитів та валідацію вхідних даних відповідно до рекомендацій OWASP [6].

Інтеграцію платіжного сервісу реалізовано на основі моделі платіжного наміру Stripe Payment Intent. Серверна частина обчислює суму замовлення, створює замовлення у статусі «очікує оплати» та платіжний намір, після чого передає клієнтський секрет клієнтському застосунку, який відображає захищену форму введення картки — компонент Payment Element. Дані картки вводяться безпосередньо в інтерфейс Stripe і не передаються на сервер магазину, завдяки чому магазин звільняється від зберігання карткових даних та відповідної відповідальності за стандартом PCI DSS [7].

Остаточним підтвердженням оплати слугує асинхронний webhook від платіжного сервісу, обробка якого побудована з дотриманням двох вимог — перевірки криптографічного підпису запиту та ідемпотентності. Ідентифікатор кожної обробленої події зберігається в журналі з унікальним обмеженням, тож повторне надходження тієї самої події розпізнається як дублікат. При успішній оплаті в межах однієї транзакції бази даних виконуються списання складських залишків, переведення замовлення у статус «оплачено» та очищення кошика, після чого покупцеві надсилається лист-підтвердження. Для управління магазином реалізовано адміністративну панель, а для серверного API

автоматично згенеровано документацію за специфікацією OpenAPI 3.0.

Висновок.Спроєктований та реалізований веб-сайт електронного магазину книг «BookNest» з інтеграцією платіжних сервісів демонструє сучасний підхід до побудови застосунків електронної комерції: чіткий розподіл клієнтської та серверної частин, типізований код, контейнеризоване розгортання всієї інфраструктури однією командою та безпечну інтеграцію платіжного сервісу, за якої магазин не зберігає даних банківських карток, а остаточне рішення про оплату приймається на основі підписаного й ідемпотентно обробленого webhook. Розроблена система може бути впроваджена як основа реального інтернет-магазину книг або іншої товарної ніші, а також слугувати навчально-практичним прикладом побудови застосунку електронної комерції з інтеграцією платіжних сервісів.

Література:

1. Stripe API Reference. Stripe, Inc. URL: https://docs.stripe.com/api
2. Next.js Documentation. Vercel, Inc. URL: https://nextjs.org/docs
3. PostgreSQL 16 Documentation. The PostgreSQL Global Development Group, 2023. URL: https://www.postgresql.org/docs/16/index.html
4. Docker Documentation. Docker, Inc. URL: https://docs.docker.com/
5. Open Library Developers Center — APIs. Internet Archive. URL: https://openlibrary.org/developers/api
6. OWASP Top Ten 2021. The OWASP Foundation, 2021. URL: https://owasp.org/Top10/
7. Stripe Payments — Payment Intents API. Stripe, Inc. URL: https://docs.stripe.com/payments/payment-intents

УДК 004.415.2:004.455

ВІРТУАЛЬНІ ПОТОКИ В JAVA

***Б. В. Нелюба**, студент групи КН м-11*
Полтавський університет економіки і торгівлі
bogdanneliuba@ukr.net
***Ю. Ф. Олексійчук**, к.ф.-м.н., доцент*
Полтавський університет економіки і торгівлі
oleksiichuk.yurii@puet.edu.ua

В роботі розглядають особливості та переваги віртуальних потоків в Java. Віртуальні потоки є альтернативою реактивному програмуванню при I/O-навантаженні.

Neliuba B. V., Oleksiichuk Yu. F. This paper examines the features and advantages of virtual threads in Java. Virtual threads offer an alternative to reactive programming for I/O-bound workloads.

Ключові слова: ВІРТУАЛЬНІ ПОТОКИ, БАГАТОПОТОКОВЕ ПРОГРАМУВАННЯ, JAVA.
Keywords: VIRTUAL THREADS, MULTI-THREADED PROGRAMMING, JAVA.

Сучасні серверні застосунки стикаються з дедалі вищими вимогами до пропускної здатності та ефективності використання ресурсів. Традиційна модель конкурентності Java, що ґрунтується на відповідності "один запит — один платформний потік ОС", має фундаментальне обмеження масштабованості: кожен платформний потік споживає близько 1 МБ пам'яті стеку та пов'язаний з дорогою операцією перемикання контексту на рівні операційної системи. За умов інтенсивного I/O-навантаження, характерного для мікросервісних архітектур, переважна частина потоків перебуває у стані очікування, блокуючи ресурси і не виконуючи корисної роботи.

Для подолання цього обмеження впродовж останнього десятиліття активно розвивалась реактивна парадигма програмування. Spring WebFlux [1], побудований на основі Project Reactor та сервера Netty, реалізує неблокуючу асинхронну модель

на базі event loop, що дозволяє обробляти тисячі одночасних запитів на мінімальній кількості потоків ОС. Однак реактивний підхід має суттєву ціну: розробники змушені повністю переосмислити структуру коду, відмовившись від звичного імперативного стилю на користь функціонального ланцюжка операторів (Mono, Flux), що суттєво ускладнює розробку, тестування та діагностику.

Принципово інше вирішення проблеми запропоновано в рамках Project Loom. Віртуальні потоки, що вперше з'явилися в Java 21 [2] і отримали подальший розвиток у Java 25 LTS, є легковаговими потоками, керованими JVM, а не операційною системою. Ключова перевага полягає у збереженні звичного синхронного стилю програмування при досягненні масштабованості, порівнянної з реактивним підходом.

Віртуальні потоки працюють за моделлю M:N, тобто величезна кількість віртуальних потоків розподіляється між невеликою кількістю платформних потоків. Цим процесом керує планувальник на основі ForkJoinPool [3]. Під час виконання операцій введення-виведення, що зупиняють роботу, віртуальний потік відкріплюється від потоку-носія, і той одразу стає вільним для інших завдань. Це дозволяє підтримувати велику кількість одночасних запитів в очікуванні без пропорційного зростання кількості потоків операційної системи.

Важливо, що початкова кількість потоків-носіїв дорівнює кількості ядер процесора. Однак, якщо потоки-носії блокуються безпосередньо (наприклад, через специфічну синхронізацію чи системні виклики), планувальник автоматично розширює пул. Саме цей механізм забезпечує принципово іншу здатність до масштабування порівняно з класичними платформними потоками.

Spring Boot є популярним інструментом для backend-розробки [4]. Стандартним підходом для розробки API є використання Spring Web. Для переходу до асинхронного WebFlux потрібно дуже сильно змінювати проєкт. Натомість для переходу до віртуальних потоків достатньо додати лише 1 рядок в конфігурації.

Продуктивність віртуальних потоків розглядається в кількох роботах [5-7], але залишається актуальною задачею, особливо в щодо нової версії Spring Boot 4.

Навантажувальне тестування буде виконуватися з використанням k6 — сучасного інструменту написаного на Go. Можна застосовувати кілька типів тестів:

1) Ramping VUs — фіксована кількість віртуальних користувачів , вимірює поведінку при фіксованому рівні конкурентності;

2) Constant Arrival Rate — фіксована кількість запитів на секунду, вимірює затримку при заданій пропускній здатності незалежно від часу відповіді;

3) Ramping Arrival Rate — поступове збільшення навантаження для визначення точки насичення кожної конфігурації.

Отже, віртуальні потоки є важливим нововведенням останніх версіях Java і дослідження їх продуктивності є актуальною задачею.

***Література*:**

1. Marten D., Rubio D., Long J. Spring WebFlux. In Spring 6 Recipes: A Problem-Solution Approach to Spring Framework, Berkeley, CA: Apress, 2023, pp. 205-240, doi: 10.1007/978-1-4842-8649-4

2. JEP 444: Virtual Threads, https://openjdk.org/jeps/444.

3. Pinto G., Canino A., Castor F., Xu G., Liu Y. D. Understanding and overcoming parallelism bottlenecks in ForkJoin applications, 2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE), Urbana, IL, USA, 2017, pp. 765-775, doi: 10.1109/ASE.2017.8115687

4. Олексійчук Ю. Ф., Ольховська О. В., Ольховський Д. М., Орлова Д. І. Проєктування та розробка web-сервісу для генерування та розсилки pdf-документів. Системи та технології, 65(1), 2023, С. 39-45. doi: 10.32782/2521-6643-2023.1-65.5

5. Chirila C. B., Sora I. Java single vs. platform vs. virtual threads runtime performance assessment in the context of key class detection. In 2024 IEEE 18th International Symposium on Applied Computational Intelligence and Informatics (SACI), 2024, pp. 000273-000278.

6. Navarro A., Ponge J., Le Mouël F., Escoffier C. Considerations for integrating virtual threads in a Java framework: a Quarkus example in a resource-constrained environment. In Proceedings of the 17th ACM

International Conference on Distributed and Event-based Systems, 2023, pp. 103-114.
7. Sirotić Z., Sovilj S., Oršulić M., Pripužić K. Comparison of Java Virtual and Non-Virtual Threads in Parallel Programming. In 2025 MIPRO 48th ICT and Electronics Convention, IEEE, 2025, pp. 1956-1961.

УДК 004.738.5:004.43:378.147

ІНТЕРАКТИВНА НАВЧАЛЬНА ВЕБ-ПЛАТФОРМА ДЛЯ ОПАНУВАННЯ ОБРОБКИ ВИНЯТКОВИХ СИТУАЦІЙ У МОВІ JAVA

***А. Р. Олешко**, спеціальність «Комп'ютерні науки», група КН б-41;*
***Ю. Ф. Олексійчук**, доцент кафедри комп'ютерних наук та інформаційних технологій, канд. фіз.-мат. наук, доцент*
Полтавський університет економіки і торгівлі

Запропоновано інтерактивну веб-платформу «JavaExceptions» для навчання обробки виняткових ситуацій у мові Java у межах дисципліни «Об'єктно-орієнтоване програмування». Тема обробки винятків є фундаментальною у курсі ООП, проте традиційне навчання потребує попереднього встановлення JDK та інтегрованого середовища, не дає миттєвого зворотного зв'язку та не дозволяє викладачу відстежувати реальний прогрес студентів. Запропонована платформа усуває технічний поріг входу та поєднує теоретичний матеріал, виконання Java-коду у браузері й автоматизоване тестування знань в єдиному середовищі [1].

Архітектура побудована за принципом гібридного веб-додатку з серверним рендерингом на базі Next.js 16 (App Router) та React 19. Клієнтський шар реалізовано на TypeScript із застосуванням Tailwind CSS 4 для адаптивної стилізації та власних UI-примітивів на основі class-variance-authority. Серверний шар обмежено тонким API-маршрутом /api/run, що проксує запити до зовнішнього сервісу виконання коду. Управління станом — Zustand з мідлваром persist для серіалізації прогресу у localStorage, що знімає потребу у власній базі даних та системі автентифікації [2], [3].

Навчальний контент організовано як одинадцять послідовних тематичних модулів — від ієрархії класу Throwable, базових конструкцій try-catch-finally, throw/throws, multi-catch і try-with-resources до власних класів винятків, ланцюжків винятків, найкращих практик, продакшн-прийомів (структуроване логування, глобальний обробник, transaction boundaries, патерн wrap & translate) та підсумкового інтеграційного проєкту спрощеної банківської системи. Кожен модуль містить чотири-

п'ять практичних вправ чотирьох жанрів (доповнення коду, виправлення помилки, передбачення виводу, написання з нуля) та тест із восьми питань множинного вибору; загальний обсяг — 47 вправ і 88 тестових питань.

Для виконання Java-коду у браузері інтегровано редактор Monaco Editor (з підсвічуванням синтаксису Java, темна тема vs-dark, нумерація рядків) та публічний API сервісу Piston (Java 15.0.2, ізольоване контейнерне виконання). Клієнтський компонент-контролер ExerciseRunner надсилає асинхронний запит до серверного маршруту /api/run, отримує структуру stdout, stderr та код виходу, нормалізує вивід і порівнює з очікуваним результатом. Тип помилки класифікується функцією detectErrorType через регулярний вираз: compile-error, runtime-error або wrong-output [4], [5].

Інтерфейс користувача спроєктовано адаптивним до десктопних, планшетних і мобільних розмірів екрана; підтримку темної та світлої тем реалізовано через CSS-медіазапит prefers-color-scheme та механізм класу dark Tailwind без додаткових бібліотек. Сторінкові маршрути модулів побудовано за схемою «тонкий серверний обгортка + клієнтське представлення» (ModuleTheoryView, ModuleExercisesView, QuizView). Підсумковий екзамен містить 20 випадково обраних питань з єдиного пулу 88 тестових питань курсу та обмежений у часі (20 хвилин) з машиною станів intro/running/finished і автозавершенням за таймером.

Прогрес користувача зберігається у локальному сховищі браузера під ключем java-exceptions-progress із підтримкою версіонування та міграції схеми; функція агрегованого прогресу реалізована динамічно і автоматично адаптується до зміни кількості модулів. Розгортання передбачено двома сценаріями: повна версія через Docker Compose з self-hosted інстансом Piston (включно з патчем UTF-8 для JVM) або статична демо-версія на хмарних хостингах з вимкненим виконанням коду через змінну середовища NEXT_PUBLIC_DISABLE_CODE_RUN [6], [7].

Висновок. Спроєктована та реалізована навчальна платформа поєднує структурований теоретичний матеріал, інтерактивні практичні вправи з автоматичною перевіркою та підсумкові тести

у єдиному веб-середовищі без потреби у локальному встановленні інструментарію Java. Архітектура без власної бази даних спрощує розгортання та супровід, а модульна структура контенту відкриває можливість подальшого розширення курсу. Розроблений програмний засіб може бути використаний у навчальному процесі ПУЕТ під час викладання дисципліни «Об'єктно-орієнтоване програмування», для самостійної підготовки студентів спеціальностей у галузі інформаційних технологій та як основа для аналогічних освітніх проєктів з інших тем мови Java.

Література:

1. Oracle. The Java Tutorials. Lesson: Exceptions. URL: https://docs.oracle.com/javase/tutorial/essential/exceptions/
2. Vercel. Next.js Documentation. App Router and Server Components. URL: https://nextjs.org/docs
3. Meta. React Documentation. URL: https://react.dev/
4. EngineerMan. Piston: a high performance general purpose code execution engine. URL: https://github.com/engineer-man/piston
5. Microsoft. Monaco Editor Documentation. URL: https://microsoft.github.io/monaco-editor/
6. Pmnd.rs. Zustand: a small, fast and scalable state-management solution. URL: https://github.com/pmndrs/zustand
7. Tailwind Labs. Tailwind CSS Documentation. URL: https://tailwindcss.com/docs

УДК 378.147:004.9

РЕАЛІЗАЦІЯ ГРАФІЧНОГО МЕТОДУ РОЗВ'ЯЗАННЯ ЗАДАЧ ЛІНІЙНОГО ПРОГРАМУВАННЯ У GEOGEBRA

***Т. О. Парфьонова**, к.ф.-м.н., доцент*
Полтавський університет економіки і торгівлі
tapa.poltava@gmail.com

У роботі розглянуто можливості використання програмного середовища GeoGebra для реалізації графічного методу розв'язання задач лінійного програмування.

Parfonova T.O. The paper considers the possibilities of using the GeoGebra software environment to implement a graphical method for solving linear programming problems.

Ключові слова: ОПТИМІЗАЦІЯ, ЛІНІЙНЕ ПРОГРАМУВАННЯ, ГРАФІЧНИЙ МЕТОД, GEOGEBRA, ЦІЛЬОВА ФУНКЦІЯ, ОБЛАСТЬ ДОПУСТИМИХ РОЗВ'ЯЗКІВ.

Keywords: OPTIMIZATION, LINEAR PROGRAMMING, GRAPHICAL METHOD, GEOGEBRA, OBJECTIVE FUNCTION, AREA OF FEASIBLE SOLUTIONS.

Лінійне програмування є одним із найбільш поширених розділів математичного програмування, в якому розглядаються різні методи розв'язування задач оптимізації в економіці, логістиці, управлінні виробництвом та інших галузях. Один із базових методів розв'язання задач лінійного програмування з двома змінними – графічний метод, який дозволяє наочно дослідити область допустимих розв'язків та знайти оптимальний план.

Сучасні програмні засоби динамічної математики відкривають нові можливості для візуалізації математичних моделей. Серед них особливе місце займає GeoGebra — безкоштовне програмне середовище, яке поєднує інструменти геометрії, алгебри, аналізу та статистики. Використання GeoGebra під час вивчення лінійного програмування дозволяє автоматизувати побудову графіків обмежень, визначення області допустимих розв'язків та аналіз

поведінки цільової функції.

Розглянемо задачу. Знайти графічним методом значення функції $F(x) = 6x_1 + x_2 \rightarrow \max (\min)$, за обмежень:

$5x_1 - 2x_2 \leq 10, 2x_1 + 3x_2 \geq 12, -x_1 + x_2 \leq 2, x_1 \geq 0, x_2 \geq 0,$

Згідно алгоритму графічного методу на першому етапі здійснюється побудова многокутника розв'язків. Для цього знаходяться півплощини, які відповідають обмеженням задачі. Перетин таких півплощин утворює область допустимих розв'язків. Для реалізації даного кроку в GeoGebra можна ввести у полі введення (ліворуч) систему нерівностей. Для виділення спільної області (многокутника розв'язків) зручно скористатись логічним "І" (з'єднуємо всі нерівності через &&) (рис.1).

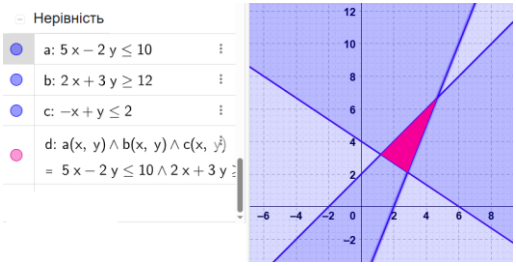

Рис.1 – Перший крок графічного методу в GeoGebra

Для подальшого відображення лише многокутника розв'язків можна приховати побудовані об'єкти, натиснувши ліворуч на кружечок у полі введення. Але разом з тим потрібно здійснити побудову прямих, що відповідають рівнянням обмежень задачі. Оптимальний розв'язок задачі лінійного програмування завжди лежить у вершинах області допустимих рішень. [1]

Далі можна скористатись інструментом «Точка», побудувати точки перетину прямих, з'єднати їх відрізками. Для виділення многокутника розв'язку можна використати інструмент «Многокутник» [2].

Наступний крок – побудова цільової функції у вигляді сімейства паралельних прямих, а її переміщення дозволяє визначити вершину області допустимих розв'язків, у якій досягається оптимальне значення. В роботі реалізовано побудову вектора цілі та рухомої прямої $c_1x_1 + c_2x_2 = l$, де параметри c_1, c_2

та l можна змінювати за допомогою повзунків (рис.2). Крім того, в GeoGebra є можливість обчислити значення цільової функції, а також реалізувати покрокову візуалізацію алгоритму методу.

Перевагами використання GeoGebra є інтерактивність, наочність та можливість оперативної зміни параметрів задачі. Студенти можуть спостерігати вплив коефіцієнтів цільової функції та обмежень на форму області допустимих розв'язків і положення оптимального розв'язку. Це сприяє формуванню глибшого розуміння математичної сутності задач оптимізації.

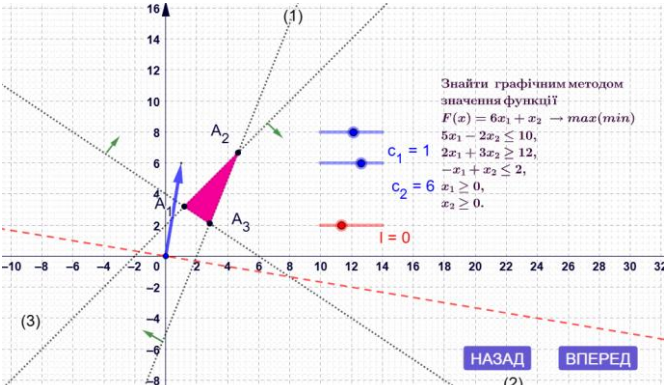

Рис.2 – Ілюстрація реалізації графічного методу в GeoGebra

Таким чином, застосування GeoGebra для реалізації графічного методу розв'язання задач лінійного програмування підвищує ефективність навчального процесу, забезпечує наочне представлення математичних моделей та сприяє розвитку практичних навичок використання інформаційних технологій у професійній діяльності.

Література

1. Мартинюк Г.В. Курс лекцій з дисципліни Методи оптимізації та дослідження операцій. / Г.В. Мартинюк, Н.Б. Марченко, Л.М. Щербак. – К.: МДУ, 2024. – 48 с.
2. Чемерис О. Майстерня GeoGebra: практичний підхід до візуалізації математики: методичні рекомендації / Ольга Чемерис, Алла Прус, Олена Фонарюк. Житомир: Вид-во ЖДУ ім. І. Франка, 2024. 46 с.

УДК 004.738.5:339.178

РОЗРОБКА ІНТЕРНЕТ-МАГАЗИНУ ВЕСІЛЬНОГО ОДЯГУ ТА АКСЕСУАРІВ

***О. Пригода**, спеціальність «Комп'ютерні науки», група КН б-41;*
Полтавський університет економіки і торгівлі

У сучасному цифровому середовищі важливим аспектом електронної комерції є створення зручних та ефективних веб-додатків, що забезпечують повноцінний досвід онлайн-покупок. Адаптивність, продуктивність та безпека є ключовими критеріями для успішного функціонування таких рішень, особливо у сфері весільної індустрії, де естетичність інтерфейсу та зручність підбору товарів відіграють визначальну роль. Метою цієї роботи є розробка веб-додатку інтернет-магазину весільної продукції «Sandrela» з акцентом на гнучку фільтрацію, захищену аутентифікацію та адаптивний дизайн.

У рамках проєкту було проведено аналіз існуючих рішень у сфері онлайн-магазинів весільної продукції, зокрема Wedboom.UA, Azazie та David's Bridal. Дослідження включали порівняння функціональних можливостей: систем фільтрації, відгуків, списків бажань, особистих кабінетів та адаптивності інтерфейсів. Використані джерела [1-5] висвітлюють ключові аспекти веб-розробки, проєктування клієнт-серверної архітектури та забезпечення безпеки веб-додатків.

Розроблений веб-додаток включає:

- каталог товарів: перегляд весільних суконь та аксесуарів із системою фільтрації за категоріями, розмірами та ціновим діапазоном, сортуванням за ціною, новизною та популярністю;
- кошик та оформлення замовлення: додавання товарів із вибором розміру та кількості, вибір адреси доставки, способу доставки та методу оплати. Система аутентифікації: реєстрація, авторизація на основі JWT-токенів (access + refresh), відновлення паролю через OTP-код на email;

- особистий кабінет: управління замовленнями з експортом у CSV, адресами доставки, способами оплати, списком бажань, відгуками та активними сесіями;
- адаптивність: коректне відображення інтерфейсу на мобільних пристроях, планшетах та настільних комп'ютерах завдяки системі breakpoints Material UI.

Процес розробки включав такі етапи:

- дослідження вимог — аналіз існуючих аналогів, визначення функціональних та нефункціональних вимог до системи, формування порівняльної таблиці можливостей;
- проєктування архітектури — побудова клієнт-серверної моделі (React ↔ NestJS ↔ PostgreSQL ↔ Cloudflare R2) із модульною структурою та REST API;
- реалізація серверної частини — створення 11 функціональних модулів на NestJS із валідацією через class-validator, JWT-аутентифікацією через Passport, обробкою зображень через Sharp та хмарним зберіганням у Cloudflare R2;
- реалізація клієнтської частини — побудова інтерфейсу на React із TypeScript, Material UI, Redux Toolkit для управління станом та React Query для кешування серверних даних;
- тестування — перевірка адаптивності на різних пристроях та аналіз швидкодії за допомогою Google Lighthouse.

Одним із ключових аспектів проєкту стало забезпечення безпеки та масштабованості. Для цього було застосовано такі підходи:

- використання JWT-токенів (access + refresh) у httpOnly cookies із захистом через Helmet та обмеженням запитів через ThrottlerModule;
- хешування паролів через bcrypt та відновлення доступу через OTP-код із відправкою email через Nodemailer;
- модульна архітектура NestJS із розділенням на контролери, сервіси та DTO, що дозволяє розширювати функціонал без перебудови існуючої структури;

У процесі розробки також було впроваджено заходи для підвищення продуктивності додатку:

- lazy loading сторінок через React.lazy() та code splitting для розділення бандлу на окремі чанки;

- відкладене завантаження зображень через react-lazy-load-image-component та конвертація у WebP через Sharp на сервері;
- кешування серверних даних через React Query із налаштуванням staleTime для зменшення кількості повторних запитів.

Цей підхід дозволив створити повнофункціональний, адаптивний та безпечний веб-додаток для інтернет-магазину весільної продукції, який надає користувачам зручний інструмент для підбору, порівняння та придбання товарів із повноцінним управлінням замовленнями та персональним кабінетом.

Література:

1. Banks, A., & Porcello, E. (2020). Learning React: Modern Patterns for Developing React Apps. O'Reilly Media. Retrieved from https://www.oreilly.com
2. Mysliwiec, K. (2026). NestJS: A Progressive Node.js Framework. Official Documentation. Retrieved from https://docs.nestjs.com
3. Prisma Team. (2026). Prisma ORM Documentation: Modern Database Access for TypeScript. Retrieved from https://www.prisma.io/docs
4. Material UI Team. (2026). MUI: The React Component Library. Official Documentation. Retrieved from https://mui.com
5. Jones, M., Bradley, J., & Sakimura, N. (2015). JSON Web Token (JWT). RFC 7519, IETF. Retrieved from https://tools.ietf.org/html/rfc7519

УДК 004.9

ОСОБЛИВОСТІ ОПТИМІЗАЦІЇ ПОБУТОВИХ ПРОЦЕСІВ У СТУДЕНТСЬКОМУ ГУРТОЖИТКУ: ПРОБЛЕМАТИКА ТА РІШЕННЯ ДЛЯ ПРАННЯ

Д.Ю.Сакун, *студент групи КН-б-41, спеціальності «Комп'ютерні науки»;*
О.В. Ольховська*, науковий керівник, завідувач кафедри, кандидат фізико-математичних наук*
Полтавський університет економіки і торгівлі

Сучасний ритм життя студента вимагає максимальної ефективності у розподілі часу. Побут у гуртожитку часто стає фактором, що непродуктивно поглинає ресурс, необхідний для навчання та саморозвитку. Хоча наявність автоматичних пральних машин вирішує проблему фізичного навантаження, головною проблемою залишається логістика та управління спільним ресурсом в умовах високого попиту [1,2].

Організація прання в умовах спільного проживання стикається з низкою проблем, які потребують не стільки розширення кількості побутової техніки, скільки сучасних організаційних рішень.

Нераціональне використання часу та застарілі методи адміністрування. Традиційний підхід до організації черги за допомогою паперового журналу запису є архаїчним і неефективним. Паперовий носій має низку критичних недоліків: записи часто дублюються або навмисне виправляються іншими мешканцями, а для перевірки вільного часу чи запису необхідно фізично відвідувати пральню, що призводить до втрати часу. Найбільш актуальне рішення цієї проблеми - впровадження хмарної електронної системи бронювання (наприклад, інтегрованої через Telegram-бот або мобільний застосунок), яка повністю замінить паперовий аналог.

Перехід на цифрову систему запису кардинально оптимізує організованість процесу та дає декілька ключових переваг:

1. Віддалений доступ та прозорість даних. Студент може бачити вільні слоти в реальному часі та бронювати їх зі свого смартфона, що жорстко алгоритмізує чергу та унеможливлює накладання графіків двох користувачів.

2. Автоматизація контролю. Система здатна надсилати автоматичні push-сповіщення про початок і наближення завершення циклу прання. Це мінімізує простій техніки через те, що користувач забув вчасно звільнити барабан машини.

3. Збір аналітичних даних. Оцифрування черги дозволяє адміністрації гуртожитку чи органам студентського самоврядування фіксувати реальне навантаження на техніку. Це дає змогу точно визначати пікові години та планувати графік технічного обслуговування апаратів на основі реальної статистики, а не спостережень.

Отже, ефективність побуту в гуртожитку сьогодні визначається не кількістю доступної побутової техніки, а якістю управління нею. Заміна застарілих паперових методів обліку на цифрові інструменти планування створює прозоре середовище, яке оптимізує рутину, знижує конфліктні ситуації та вивільняє час студентів для профільної діяльності.

***Література*:**

1. Пристемський О. С., Пашинний А. В. Вплив сучасних інформаційних технологій на ефективність управління підприємством. «Актуальні проблеми менеджменту: теоретичні і практичні аспекти»: Матеріали шостої міжнар. наук.–практ. конф., 28-29 вересня 2023 р.

2. Все про чат-боти: переваги, типи та схема роботи. [Електронний ресурс]. - 2020 -.https://www.interkassa.com/ua/blog/vse-ochat-botah-preimushchestva-tipy-i-shema-raboty/.

УДК 004.42:004.92

ТЕХНОЛОГІЇ ПОБУДОВИ АДАПТИВНИХ ВЕБІНТЕРФЕЙСІВ ДЛЯ ПЕРСОНАЛЬНИХ ВЕБРЕСУРСІВ

А. Б. Сірооченко, *студентка, КН-41*
alinasiroochenko@gmail.com
О. П. Кошова*, к. пед. н., доцент кафедри комп'ютерних наук та інформаційних технологій.*
koshova.o111@gmail.com
Полтавський університет економіки і торгівлі

У статті досліджуються технології побудови адаптивних вебінтерфейсів для персональних вебресурсів. Розглянуто сучасні підходи до проєктування користувацьких інтерфейсів та особливості використання HTML5, CSS3 і JavaScript під час створення вебсайту персонального портфоліо. Проаналізовано застосування технологій CSS Grid, Flexbox та медіа-запитів для забезпечення адаптивності інтерфейсу.

Siroochenko A., Koshova O. Technologies for building adaptive web interfaces for personal web resources. The article investigates technologies for building adaptive web interfaces for personal web resources. Modern approaches to user interface design and the use of HTML5, CSS3 and JavaScript in the development of a personal portfolio website are considered. The application of CSS Grid, Flexbox and media queries for ensuring website responsiveness is analyzed.

Ключові слова: *АДАПТИВНИЙ ВЕБДИЗАЙН, ВЕБІНТЕРФЕЙС, HTML5, CSS3, JAVASCRIPT, ПЕРСОНАЛЬНИЙ ВЕБРЕСУРС, ВЕБТЕХНОЛОГІЇ.*
Keywords: RESPONSIVE WEB DESIGN, WEB INTERFACE, HTML5, CSS3, JAVASCRIPT, PERSONAL WEB RESOURCE, WEB TECHNOLOGIES.

Цифрові технології стали невід'ємною складовою професійної діяльності сучасного фахівця. Представлення власних досягнень,

професійних компетентностей та результатів виконаних проєктів дедалі частіше здійснюється за допомогою персональних вебресурсів. Вебсайт-портфоліо дозволяє сформувати цілісне цифрове представлення особистості, забезпечує швидкий доступ до інформації про професійний досвід та створює додаткові можливості для взаємодії з роботодавцями, замовниками та професійною спільнотою.

Ефективність персонального вебресурсу значною мірою залежить від якості його інтерфейсу. Сучасні користувачі здійснюють доступ до вебресурсів із різноманітних пристроїв, що обумовлює необхідність використання адаптивного дизайну та технологій, здатних забезпечити коректне відображення контенту незалежно від роздільної здатності екрана. У зв'язку з цим особливого значення набувають питання побудови адаптивних вебінтерфейсів із використанням сучасних клієнтських вебтехнологій.

Дослідження у сфері веброзробки демонструють зростання ролі сучасних інструментів створення вебресурсів та орієнтацію на підвищення якості взаємодії користувача з цифровими системами. У роботі [1] розглядаються сучасні підходи до розробки вебдодатків і сервісів, які забезпечують ефективне представлення інформації та взаємодію користувачів із вебресурсами. Це підтверджує актуальність використання сучасних вебтехнологій під час створення персональних інформаційних систем.

Метою роботи є дослідження технологій побудови адаптивних вебінтерфейсів та аналіз їх практичного застосування під час розробки персонального вебресурсу.

Основою реалізації розробленого вебсайту виступають технології HTML5, CSS3 та JavaScript. Вибір зазначеного технологічного стеку обумовлений його широким використанням у сучасній веброзробці, підтримкою всіма популярними браузерами та можливістю створення високопродуктивних вебресурсів без використання додаткових програмних платформ.

Технологія HTML5 використовується для формування семантичної структури вебсторінок та логічної організації контенту [3]. Застосування семантичних тегів дозволяє підвищити доступність вебресурсу, спрощує його підтримку та позитивно

впливає на індексацію пошуковими системами. Структура розробленого вебсайту складається з окремих функціональних секцій, які забезпечують представлення професійної інформації про автора, навичок, освіти, реалізованих проєктів та засобів комунікації.

Для реалізації зовнішнього вигляду вебресурсу використано каскадні таблиці стилів CSS3. Значна увага приділялася забезпеченню адаптивності інтерфейсу та створенню сучасного візуального оформлення. Використання CSS дозволяє відокремити структуру сторінки від її оформлення та забезпечує гнучке налаштування зовнішнього вигляду вебресурсу [4].

Однією з основних вимог до розробленого вебсайту було забезпечення коректного відображення інтерфейсу на пристроях із різними характеристиками екрана. Для досягнення цієї мети використано адаптивний підхід до проєктування інтерфейсу, який базується на використанні медіа-запитів та сучасних механізмів верстки CSS.

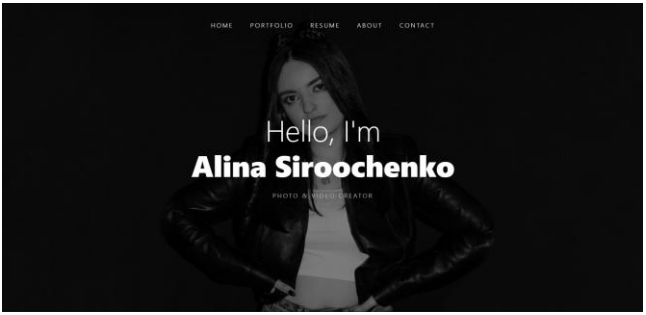

Рисунок 1. Головна сторінка персонального вебресурсу

Головна сторінка виконує роль центрального елемента навігації та забезпечує перше знайомство користувача з вебресурсом. На сторінці розміщено основну інформацію про автора, професійну спеціалізацію та засоби переходу до інших функціональних розділів сайту.

Отже, використання ASP.NET Core MVC, Entity Framework Core та сучасних підходів до проєктування вебзастосунків дозволило створити масштабовану інформаційну систему для

ветеринарної клініки. Реалізоване програмне забезпечення забезпечує централізоване керування даними, підтримку запису на прийом, представлення послуг і публікацій, а також повноцінне адміністрування основних об'єктів системи. Отримані результати підтверджують ефективність використання технологій платформи .NET для побудови сучасних веборієнтованих інформаційних систем.

Важливою складовою сучасних вебінтерфейсів є можливість ефективного використання доступного простору екрана. Для реалізації даної вимоги у проєкті застосовано технології CSS Grid та Flexbox. Ці механізми дозволяють створювати гнучкі макети сторінок, які автоматично адаптуються до параметрів пристрою користувача без необхідності створення окремих версій вебресурсу.

Технологія CSS Grid використовувалася для побудови складних структур сторінок, які містять декілька інформаційних блоків. Основною перевагою даного підходу є можливість створення двовимірних сіток, що дозволяє ефективно керувати розташуванням елементів інтерфейсу як по горизонталі, так і по вертикалі. Використання CSS Grid забезпечує зручне розміщення контенту та спрощує подальшу модифікацію структури вебсторінок.

Для побудови навігаційних елементів та вирівнювання окремих компонентів інтерфейсу застосовано технологію Flexbox. Її використання дозволило реалізувати адаптивне меню навігації, коректне вирівнювання текстових блоків та забезпечити узгоджене розташування елементів на сторінці незалежно від розміру робочої області браузера. Поєднання CSS Grid і Flexbox дозволяє реалізовувати сучасні інтерфейси без використання додаткових бібліотек або фреймворків.

Одним із функціональних блоків вебресурсу є секція представлення автора, яка містить фотографію, коротку інформацію про професійну діяльність та можливість завантаження резюме.

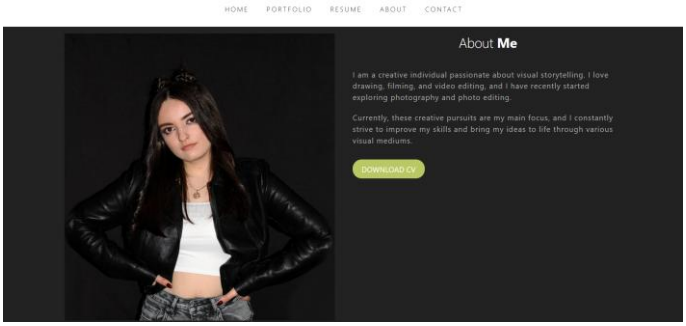

Рисунок 2. Секція представлення автора вебсайту

Під час проєктування інтерфейсу особлива увага приділялася забезпеченню зрозумілого представлення інформації та мінімізації кількості дій, необхідних для отримання доступу до основних відомостей про користувача. Для досягнення цієї мети використано лаконічний дизайн, чітку структуру контенту та єдину кольорову схему оформлення.

Важливим компонентом персонального вебресурсу є представлення освітньої підготовки та професійного розвитку. У межах розробленого сайту відповідна інформація відображається у вигляді окремих інформаційних карток, які містять відомості про період навчання, спеціальність та освітню установу. Такий підхід забезпечує структуроване подання інформації та покращує її сприйняття користувачами.

Рисунок 3. Секція освіти та професійної підготовки

Однією з головних функцій вебсайту-портфоліо є демонстрація результатів практичної діяльності автора. Для представлення реалізованих проєктів використано адаптивну сіткову структуру, побудовану засобами CSS Grid. Такий підхід дозволяє автоматично змінювати кількість елементів у рядку залежно від розміру екрана та забезпечує ефективне використання доступного простору інтерфейсу.

Секція портфоліо містить графічні матеріали та посилання на реалізовані роботи. Кожен проєкт представлено окремою карткою, що дозволяє забезпечити зручне сприйняття інформації та підтримує сучасні принципи організації контенту.

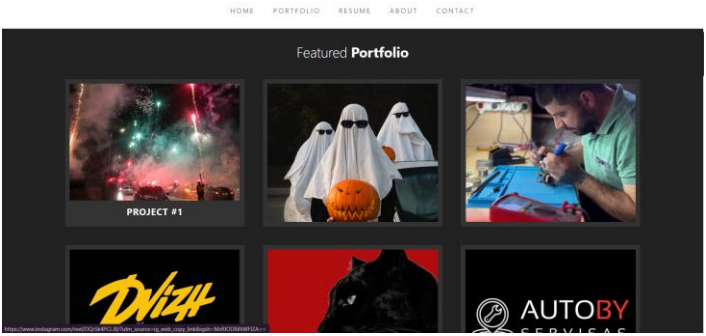

Рисунок 4. Секція реалізованих проєктів

Значна увага приділялася забезпеченню продуктивності вебресурсу. На відміну від складних вебсистем, що використовують значну кількість сторонніх бібліотек, розроблений сайт побудований із використанням базових клієнтських технологій. Такий підхід дозволив мінімізувати обсяг програмного коду, скоротити кількість зовнішніх залежностей та забезпечити швидке завантаження сторінок.

Для реалізації інтерактивних елементів інтерфейсу використано JavaScript [5]. Застосування клієнтських сценаріїв дозволило реалізувати механізми взаємодії користувача з вебресурсом без необхідності перезавантаження сторінок. Зокрема, програмними засобами реалізовано роботу мобільного меню навігації, динамічне відображення окремих елементів інтерфейсу та взаємодію з

формами введення даних.

Використання JavaScript сприяє покращенню користувацького досвіду та забезпечує більш комфортну взаємодію із вебресурсом. При цьому відсутність складної серверної логіки дозволяє зберігати високу швидкодію сайту та мінімізувати вимоги до середовища його розгортання.

Для забезпечення сучасного візуального оформлення інтерфейсу використано бібліотеки Google Material Icons та IonIcons. Використання векторних іконок дозволяє зберігати високу якість графічних елементів незалежно від роздільної здатності екрана та сприяє формуванню єдиного стилю оформлення вебресурсу. Застосування подібних інструментів відповідає сучасним тенденціям вебдизайну та дозволяє покращити сприйняття інформації користувачем.

Окрему увагу приділено забезпеченню зручності взаємодії користувача із вебресурсом. Одним із ключових елементів інтерфейсу виступає контактна секція, яка забезпечує можливість комунікації між відвідувачем та власником вебсайту. Для реалізації даного функціоналу створено форму зворотного зв'язку, яка містить поля введення імені, адреси електронної пошти, номера телефону та текстового повідомлення.

Інтерфейс контактної форми побудований із використанням гнучких механізмів CSS-верстки та забезпечує коректне відображення на пристроях різних типів. Крім форми введення даних, у відповідному розділі також відображаються контактні відомості користувача, що забезпечує додаткові канали комунікації.

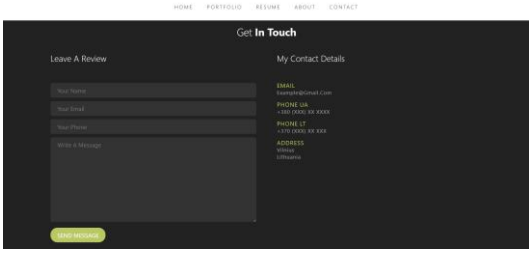

Рисунок 5. Контактна форма вебресурсу

Під час розробки вебресурсу також враховувалися вимоги щодо доступності та підтримуваності програмного коду. Застосування семантичної структури HTML-документів дозволяє спростити подальшу модифікацію вебсайту та покращує сумісність із сучасними програмними засобами обробки вебконтенту. Крім того, використання окремих файлів стилів та сценаріїв сприяє підвищенню рівня структурованості проєкту та полегшує його супровід.

Важливою перевагою розробленого вебресурсу є можливість його розгортання без використання складної серверної інфраструктури. Оскільки сайт побудовано на основі статичної архітектури, його можна розміщувати на спеціалізованих сервісах хостингу статичних вебресурсів. Для публікації проєкту використано платформу GitHub Pages, яка забезпечує безкоштовне розміщення вебсайтів та підтримує інтеграцію із системою контролю версій Git [2].

Використання GitHub Pages дозволяє спростити процес оновлення контенту, забезпечити постійну доступність вебресурсу та мінімізувати витрати на його підтримку. Такий підхід є особливо доцільним для персональних вебсайтів і портфоліо, які не потребують складної серверної логіки та обробки великих обсягів даних.

Проведений аналіз результатів розробки свідчить про ефективність використання сучасних клієнтських вебтехнологій для створення персональних вебресурсів. Застосування HTML5 забезпечило семантичну структуру сторінок та відповідність сучасним стандартам веброзробки. Використання CSS3, Flexbox і CSS Grid дозволило реалізувати адаптивний інтерфейс, який забезпечує комфортну взаємодію користувачів із вебресурсом на різних типах пристроїв. Використання JavaScript сприяло підвищенню інтерактивності інтерфейсу та покращенню користувацького досвіду.

Отримані результати підтверджують доцільність використання адаптивного підходу до проєктування вебінтерфейсів під час створення персональних вебресурсів. Реалізований вебсайт забезпечує ефективне представлення професійної інформації, підтримує сучасні принципи вебдизайну та відповідає актуальним

вимогам щодо продуктивності, доступності та зручності використання.

Таким чином, у результаті проведеного дослідження проаналізовано технології побудови адаптивних вебінтерфейсів та особливості їх практичного використання під час розробки персонального вебресурсу. Для реалізації програмного забезпечення використано HTML5, CSS3, JavaScript, CSS Grid та Flexbox. Розроблений вебсайт характеризується адаптивністю, високою швидкодією, сучасним дизайном та зручністю використання. Отримані результати можуть бути використані під час створення персональних вебсайтів, інформаційних ресурсів та інших вебпроєктів, орієнтованих на представлення професійної інформації у цифровому середовищі.

Література:

1. Кошова О. П., Ольховська О. В., Тацій Д. С., Олексійчук Ю. Ф., Черненко О. О. Розробка веб-додатків та сервісів на платформі Node.js // Таврійський науковий вісник. Серія: Технічні науки. 2023. Вип. 2. С. 78–89. Режим доступу: https://journals.ksauniv.ks.ua/index.php/tech/issue/view/26 < https://journals.ksauniv.ks.ua/index.php/tech/article/view/358/331>

2. Сірооченко А. Б. Проєктування веб-сайту власного портфоліо // Актуальні питання розвитку науки та забезпечення якості освіти у XXI столітті : тези доповідей XLIX Міжнародної наукової конференції студентів та аспірантів (м. Полтава, 23 квітня 2026 р.). Полтава : ПУЕТ, 2026. С. 455. Режим доступу: http://puet.edu.ua/wp-content/uploads/2026/06/zb_tez-2026-qual-osv-xxi.pdf

3. Документація MDN Web Docs: HTML [Електронний ресурс]. Режим доступу: https://developer.mozilla.org/uk/docs/Web/HTML.

4. Документація MDN Web Docs: CSS [Електронний ресурс]. Режим доступу: https://developer.mozilla.org/uk/docs/Web/CSS.

5. Документація MDN Web Docs: JavaScript [Електронний ресурс]. Режим доступу: https://developer.mozilla.org/uk/docs/Web/JavaScript.

УДК 004.738.5:004.43

СЕРВЕРНА АРХІТЕКТУРА HTTP-СЕРВІСУ УПРАВЛІННЯ КОРИСТУВАЧАМИ НА ОСНОВІ МОВИ GO

***Р. С. Сизько**, спеціальність «Комп'ютерні науки», група КН б-41;*
***О. О. Черненко**, доцент кафедри комп'ютерних наук та інформаційних технологій, канд. фіз.-мат. наук, доцент*
Полтавський університет економіки і торгівлі

Запропоновано серверну архітектуру HTTP-сервісу управління ідентифікацією та доступом (Identity and Access Management, IAM) на основі мови програмування Go, що забезпечує повний цикл операцій з ідентичностями користувачів і може використовуватися як автономний бекенд для веб-додатків або як ядро мікросервісних систем. За даними галузевих звітів, понад 80 % витоків інформаційної безпеки пов'язані з компрометацією облікових записів, що підкреслює актуальність розробки спеціалізованого, компактного й добре керованого IAM-сервісу [1].

Архітектура реалізована за принципами Clean Architecture з чотиришаровою структурою (domain, service, repository, handler) і повністю написана мовою Go з використанням маршрутизатора chi версії 5 та нативного драйвера pgx для PostgreSQL. У ролі сховища даних обрано PostgreSQL 16 із розширенням pgcrypto для генерації UUID на стороні бази та системою версіонування міграцій golang-migrate з десятьма послідовними кроками [2][3]. Реляційна схема нормалізована до третьої нормальної форми і складається з тринадцяти таблиць, що покривають користувачів, ролі, групи, дозволи, токени, історію входів, журнал аудиту, секрети TOTP та API-ключі.

Модуль автентифікації реалізовано на основі стандарту JSON Web Token (RFC 7519) із застосуванням схеми двох токенів: короткоживучого access-токена з часом життя одна година та довгоживучого refresh-токена з ротацією при кожному використанні. Така схема забезпечує надійне відкликання сесій при компрометації та виявлення витоку токенів через провал легітимної спроби оновлення [4]. Паролі зберігаються у незворотному вигляді як bcrypt-хеші з адаптивним числом

ітерацій; додатково реалізовано тимчасове блокування облікового запису після п'яти невдалих спроб входу та глобальне обмеження частоти запитів за алгоритмом token bucket.

Модуль контролю доступу побудований за моделлю RBAC із підтримкою індивідуальних дозволів користувача та груп з успадкуванням прав. Загальний набір дозволів формується об'єднанням індивідуальних і групових через SQL UNION і завантажується у контекст запиту під час автентифікації, що забезпечує тонку перевірку прав через декларативний middleware RequirePermission. Двофакторна автентифікація реалізована за стандартом TOTP (RFC 6238) із повною сумісністю з мобільними застосунками Google Authenticator та Microsoft Authenticator [5].

Адміністративна панель реалізована як односторінковий веб-додаток на чистій мові JavaScript з використанням Bootstrap 5 без важких реактивних фреймворків, що знижує розмір bundle і спрощує підтримку. Інтерактивна документація API генерується автоматично з коментарів у Go-коді за допомогою інструмента swag і відповідає специфікації OpenAPI 3.0, що дозволяє розробникам клієнтських застосунків випробовувати взаємодію з сервісом безпосередньо з браузера через Swagger UI [6].

Безпека з'єднання з базою даних у промисловому розгортанні реалізована через TLS у режимі verify-full, а конфіденційні дані (паролі, секретні ключі, токени) зберігаються лише у незворотно перетвореному вигляді. Розгортання виконується через Docker і docker-compose з автоматичним застосуванням міграцій при старті, що відповідає принципу 12-Factor App. Загальний обсяг кодової бази становить близько 4500 рядків коду на мові Go без врахування адміністративної панелі, а ресурсні витрати в стані спокою — близько 30–60 МБ оперативної пам'яті.

Висновок.Спроєктована та реалізована серверна архітектура HTTP-сервісу управління користувачами на основі мови Go поєднує компактність кодової бази, низькі ресурсні вимоги та повноту функціонального покриття типових сценаріїв IAM (автентифікація, RBAC, двофакторна автентифікація, аудит дій, керування сесіями). Локальна архітектура з PostgreSQL і контейнеризованим розгортанням забезпечує контроль користувача над даними та незалежність від хмарних провайдерів,

а інженерні рішення (ротація refresh-токенів, обмеження частоти запитів, журнал аудиту, автоматичні міграції) утворюють основу масштабованого рішення, придатного як легковагова альтернатива промисловим IAM-платформам типу Keycloak і Auth0 для невеликих і середніх проєктів.

Література:

1. OWASP Top Ten 2021. The OWASP Foundation, 2021. URL: https://owasp.org/Top10/
2. The Go Programming Language Documentation. URL: https://go.dev/doc/
3. PostgreSQL 16 Documentation. The PostgreSQL Global Development Group, 2023. URL: https://www.postgresql.org/docs/16/index.html
4. Jones M., Bradley J., Sakimura N. RFC 7519: JSON Web Token (JWT). URL: https://datatracker.ietf.org/doc/html/rfc7519
5. M'Raihi D., Machani S., Pei M., Rydell J. RFC 6238: TOTP — Time-Based One-Time Password Algorithm. URL: https://datatracker.ietf.org/doc/html/rfc6238
6. swaggo/swag: Automatically generate RESTful API documentation with Swagger 2.0 for Go. URL: https://github.com/swaggo/swag

УДК 004.738.5:378.147

TELEGRAM-БОТ ДЛЯ ВДОСКОНАЛЕННЯ НАВИЧОК ПРОГРАМУВАННЯ У ФОРМАТІ МІКРОНАВЧАННЯ

***П. Р. Хряпченко**, спеціальність «Комп'ютерні науки», група КН б-41;*
***О. О. Черненко**, доцент кафедри комп'ютерних наук та інформаційних технологій, канд. фіз.-мат. наук, доцент*
Полтавський університет економіки і торгівлі

Запропоновано Telegram-бот «CodeQuest» для вдосконалення навичок програмування у форматі мікронавчання. Опанування програмування вимагає систематичної практики, проте саме регулярність є найслабшою ланкою самостійного навчання, а класичні платформи орієнтовані на тривалі сесії за комп'ютером і не дозволяють використовувати короткі проміжки вільного часу. Розроблений бот знижує поріг входу до практики: користувач навчається короткими уроками по дві-три хвилини безпосередньо у месенджері, а вся взаємодія здійснюється натисканням екранних кнопок без набору коду з клавіатури. Вибір месенджера як платформи усуває потребу у встановленні окремого застосунку та надає вбудований канал нагадувань для формування навчальної звички [1].

Систему реалізовано мовою програмування Python із застосуванням асинхронного фреймворку aiogram для роботи з Telegram Bot API. Архітектуру побудовано за принципом шарової організації коду: проміжне програмне забезпечення, обробники подій, сервіси з бізнес-логікою, репозиторії доступу до даних та моделі. Повністю асинхронна модель виконання забезпечує одночасне обслуговування багатьох користувачів, а чіткий розподіл відповідальності між шарами робить кодову базу придатною до супроводу й розширення [2],[3].

Центральним компонентом системи є рушій уроків, реалізований як скінченний автомат: стан проходження уроку зберігається у сховищі Redis, що забезпечує коректне відновлення сесії між окремими натисканнями кнопок та стійкість до перезапусків застосунку. Урок є послідовністю інтерактивних

вправ, для яких реалізовано вісім типів: картка теорії, тест із варіантами, передбачення виводу коду, упорядкування рядків, вставка пропущеного токена, пошук помилкового рядка, оцінка твердження та зіставлення пар. Правильність відповіді перевіряється миттєвим зіставленням з еталоном, без зовнішніх сервісів виконання коду.

Для підтримання мотивації та регулярності занять реалізовано систему гейміфікації. Вона включає бали досвіду, рівні з прогресивними порогами, звання, тринадцять досягнень та серію днів поспіль із виконаною щоденною ціллю. Передбачено щоденну ціль, челендж дня з бонусом досвіду та два рейтинги — глобальний і тижневий. Сукупність цих механік утворює цілісну систему позитивного підкріплення, у якій кожна навчальна дія користувача дає видиму винагороду.

Постійні дані — облікові записи користувачів, навчальний контент і прогрес — зберігаються у реляційній СУБД PostgreSQL, доступ до якої організовано через бібліотеку об'єктно-реляційного відображення SQLAlchemy у поєднанні з асинхронним драйвером. Реляційну схему з дванадцяти таблиць нормалізовано до третьої нормальної форми; завдяки стовпцям типу JSONB одна таблиця обслуговує всі вісім типів вправ попри відмінність їх внутрішньої структури. Окремо реалізовано модуль інтервального повторення, що повертає помилкові вправи на повторне опрацювання за зростаючими інтервалами та реалізує науково обґрунтований принцип розподіленого у часі закріплення матеріалу [4], [5].

Навчальний контент відокремлено від програмного коду й подано структурованими даними, що охоплюють п'ять мов програмування — Python, JavaScript, Java, C++ та C# — загалом тридцять дев'ять курсів, дев'яносто уроків і п'ятсот сорок вправ. Фоновий планувальник виконує надсилання нагадувань, генерацію челенджів дня та щотижневе скидання рейтингу. Розгортання системи виконується засобами Docker і охоплює три сервіси — застосунок бота, СУБД та сховище станів, — що робить запуск простим і відтворюваним на будь-якій платформі [6].

Висновок. Розроблений Telegram-бот «CodeQuest» поєднує мікронавчання як спосіб зниження порога входу, гейміфікацію як механізм підтримання мотивації та інтервальне повторення як

засіб надійного закріплення матеріалу. Повністю асинхронна шарова архітектура, взаємодія через екранні кнопки без набору коду, відокремлений від коду навчальний контент для п'яти мов програмування та контейнеризоване розгортання роблять систему зручним і практичним інструментом для самостійного вивчення програмування та підтримання набутих навичок.

Література:

1. Telegram Bot API. Telegram Messenger, 2024. URL: https://core.telegram.org/bots/api
2. Python 3.12 Documentation. Python Software Foundation, 2024. URL: https://docs.python.org/3/
3. aiogram 3 Documentation. URL: https://docs.aiogram.dev/en/latest/
4. SQLAlchemy 2.0 Documentation. URL: https://docs.sqlalchemy.org/en/20/
5. PostgreSQL 16 Documentation. The PostgreSQL Global Development Group, 2023. URL: https://www.postgresql.org/docs/16/index.html
6. Docker Compose specification. Docker Inc., 2024. URL: https://docs.docker.com/compose/