УДК 004.41

СИСТЕМА ОНЛАЙН-БРОНЮВАННЯ НЕРУХОМОСТІ З БАГАТОРІВНЕВОЮ МОДЕЛЛЮ ДОСТУПУ

Дулов О.А.
alex.dulov2006@gmail.com
Черкаський державний фаховий бізнес-коледж
Немченко В.Ю.
м. Черкаси, Україна

Процеси цифровізації охоплюють дедалі більше сфер економіки, і ринок нерухомості не є винятком. Сучасні системи онлайн-бронювання мають забезпечувати не лише зручний інтерфейс для кінцевого користувача, а й гарантувати високий рівень безпеки персональних даних та цілісність фінансових транзакцій. Проблематика розробки таких систем полягає у необхідності розмежування прав доступу між різними категоріями користувачів (орендарями, орендодавцями та адміністраторами), що вимагає впровадження гнучких багаторівневих моделей авторизації.

Питання побудови розподілених систем бронювання та захисту даних у веб-середовищі розглядали багато вітчизняних та закордонних вчених. Проте швидкий розвиток хмарних технологій та нових методів аутентифікації потребує постійного оновлення підходів до проектування архітектури таких систем.

Метою роботи є проектування та обґрунтування архітектури системи онлайн-бронювання нерухомості з використанням багаторівневої моделі доступу – Role-Based Access Control (RBAC), що дозволить мінімізувати ризики несанкціонованого доступу до конфіденційної інформації.

Для реалізації системи обрано клієнт-серверну архітектуру. Backend-частина базується на REST API, що дозволяє в майбутньому легко масштабувати систему та підключати мобільні додатки. Основним елементом безпеки є впровадження багаторівневої моделі доступу.

Розглянемо детальніше функціональні рівні доступу:

1. Рівень гостя (неавторизований користувач): доступ лише до перегляду загального каталогу нерухомості, фільтрації за ціною та локацією.

2. Рівень клієнта (орендар): можливість створення профілю, управління кошиком бронювань, здійснення оплати та написання відгуків. Доступ до власних замовлень захищено індивідуальними токенами.
3. Рівень орендодавця (власник об'єкта): повний контроль над своїми оголошеннями, перегляд статистики зайнятості об'єктів, управління ціновою політикою та підтвердження заявок.
4. Рівень адміністратора: повний доступ до бази даних для модерації контенту, розв'язання конфліктних ситуацій та управління обліковими записами.

База даних системи розроблена з урахуванням нормалізації для забезпечення цілісності. Основні таблиці включають «Users», «Properties», «Bookings» та «Roles». Зв’язок між таблицями «Users» та «Roles» реалізовано за принципом «багато до багатьох», що дозволяє одному користувачу мати декілька ролей (наприклад, бути одночасно орендарем та орендодавцем).

Особлива увага приділена механізму «Race Condition» при одночасному бронюванні одного об’єкта двома користувачами. Для вирішення цієї проблеми застосовано песимістичне блокування записів у базі даних на рівні транзакцій, що унеможливлює паралельну зміну стану одного й того ж ресурсу. Це гарантує, що об’єкт змінить статус на «заброньовано» лише для того користувача, чий запит першим пройшов валідацію сервером.

Додатково розглянуто альтернативний підхід із використанням оптимістичного блокування, який базується на перевірці версій записів, проте він є менш надійним у сценаріях із високою конкуренцією запитів. Обраний підхід забезпечує цілісність даних та узгодженість транзакцій у реальному часі, що є критично важливим для систем онлайн-бронювання.

Безпека передачі даних забезпечується протоколом HTTPS та використанням JWT (JSON Web Tokens) для підтримки сесій. Кожен запит до захищених ресурсів перевіряється сервером на наявність дійсного токена та відповідність ролі користувача запитуваній дії.

Додатково реалізовано механізм оновлення токенів (refresh tokens), що підвищує безпеку та зручність роботи користувача без необхідності повторної аутентифікації. Для захисту системи від поширених веб-загроз застосовано перевірку вхідних даних та механізми протидії SQL-ін’єкціям, XSS та CSRF-атакам. Паролі користувачів зберігаються у вигляді хешів із використанням сучасних криптографічних алгоритмів.

Для підвищення надійності та контролю доступу передбачено ведення журналів подій (логування) та аудит дій користувачів, що дозволяє виявляти підозрілу активність і своєчасно реагувати на потенційні загрози.

Розроблена концепція системи онлайн-бронювання нерухомості з багаторівневою моделлю доступу дозволяє створити безпечне середовище для взаємодії власників та клієнтів. Використання моделі RBAC спрощує адміністрування системи та забезпечує масштабованість. Впровадження таких систем на вітчизняному ринку дозволить значно знизити витрати часу на пошук та оформлення оренди нерухомості, підвищуючи загальну прозорість ринку

Список використаних джерел:

1. Про авторське право і суміжні права : Закон України від 01.12.2022 № 2811-IX. URL: https://zakon.rada.gov.ua/laws/show/2811-20 (дата звернення: 17.03.2026).
2. Мазурок І. Є. Проектування інформаційних систем : навчальний посібник. Одеса : ОНЕУ, 2020. 184 с.
3. Жежнич П. І. Технології створення та ведення баз даних : підручник. Львів : Видавництво Львівської політехніки, 2018. 216 с.
4. Role-Based Access Control (RBAC). *NIST Computer Security Resource Center*. URL: https://csrc.nist.gov/projects/role-based-access-control (дата звернення: 17.03.2026).
5. Richardson L., Ruby S. RESTful Web Services: Service-Oriented Infrastructures for Distributed Applications. O'Reilly Media, 2017. 448 p.

6. Introduction to JSON Web Tokens. *Auth0 Docs*. URL: https://jwt.io/introduction/.
7. Спиваковський О. В., Щедролосєв Д. Є. Побудова безпечних розподілених веб-систем. Херсон : ХДУ, 2019. 150 с.

УДК 004.41

ПРОГРАМОТЕХНІКА ТА ПРОЄКТУВАННЯ КОМП'ЮТЕРНИХ СИСТЕМ: АРХІТЕКТУРА КЛІЄНТСЬКОГО РІВНЯ ТА РЕАКТИВНІ ПАТЕРНИ

Кондратенко Є.С.
ekqwert12345@gmail.com
Черкаський державний фаховий бізнес-коледж
Подорошко Д.І.
м. Черкаси, Україна

У сучасній розробці програмного забезпечення клієнтська частина (front-end) вже не є лише оболонкою для відображення даних. Великі односторінкові застосунки (SPA) та корпоративні системи стикаються з подібними архітектурними ризиками, як і серверні рішення: неконтрольоване зростання зв'язаності модулів, каскадні збої та деградацію підтримуваності з часом. Браузерні застосунки функціонують автономно, реалізуючи складні обчислення, структурні патерни та бізнес-логіку на рівні, що відповідає серверній частині [1]. Проте специфіка середовища виконання, зокрема обмежені ресурси пристрою, відкритість коду та непередбачуваність мережі, унеможливлює пряме перенесення серверних рішень без адаптації. Архітектурні рішення, прийняті на ранніх етапах розробки клієнтського застосунку, визначають його довгострокову підтримуваність і стійкість до збоїв [2].

Компонентний підхід і сучасні практики у фреймворку Angular. Розробка інтерфейсів корпоративного рівня вимагає чіткої модульності та відокремлення бізнес-логіки від логіки користувацького інтерфейсу. Angular демонструє застосування інженерних методів у створенні клієнтських систем [3]. На відміну від легких бібліотек, Angular використовує статичну типізацію (TypeScript). Це