ВЕБЗАСТОСУНОК ОБМІНУ ПОВІДОМЛЕННЯМИ З ПІДТРИМКОЮ ТЕХНОЛОГІЇ SOCKET.IO

Биков А. О.
Київський столичний університет імені Бориса Грінченка, м. Київ

ВСТУП

У роботі розглядаються підходи до вирішення задачі організації обміну повідомленнями в реальному часі між користувачами у браузерному середовищі. Для вирішення проблеми пропонується розробка спеціалізованого вебзастосунку **Chatico**, що реалізує приватні та групові чати, забезпечує миттєву доставку повідомлень на основі технології WebSocket, не вимагає встановлення стороннього ПЗ та повністю функціонує у браузері.

Актуальність і постановка проблеми. З розвитком цифрових технологій та масовим поширенням інтернету комунікація в режимі реального часу стала невід'ємною частиною повсякденного та професійного життя. Попри велику кількість існуючих рішень (Telegram, WhatsApp, Discord), вони або мають закритий вихідний код, або обмежений функціонал у вебверсії, або надмірно складну архітектуру для практичного навчання та демонстрації сучасних вебтехнологій. Відтак, актуальним є створення повнофункціонального відкритого вебзастосунку, який демонструє принципи MERN-розробки з використанням технології реального часу.

Метою роботи є розробка вебзастосунку для обміну повідомленнями в реальному часі з підтримкою приватних і групових чатів, що забезпечує зручну комунікацію між користувачами та демонструє можливості MERN-стека для створення сучасних вебсистем.

Аналіз існуючих рішень. Аналіз месенджерів Telegram, WhatsApp та Discord показав: жоден із них не є повністю відкритим браузерним рішенням, придатним для навчальної демонстрації стеку MERN. Telegram має обмежений вебклієнт і власний закритий протокол MTProto; WhatsApp орієнтований на мобільні пристрої та має закритий код; Discord має надмірно складну серверну архітектуру. Саме це підтверджує доцільність розробки власного вебзастосунку на відкритих технологіях.

РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

Теоретичні основи та архітектура рішення

Для створення Chatico обрано архітектуру клієнт-сервер із розділенням фронтенду та бекенду: React.js (клієнтська частина) + Node.js / Express.js (серверна логіка та REST API) + MongoDB (нереляційна база даних) + Socket.io (WebSocket-комунікація в реальному часі). Така архітектура забезпечує масштабованість, швидкість розробки та широку екосистему бібліотек.

Архітектура застосунку складається з трьох рівнів. Рівень подання (React.js + Chakra UI) відповідає за адаптивний інтерфейс користувача. Рівень логіки (Node.js + Express.js) реалізує REST API для автентифікації, керування чатами та повідомленнями. Рівень даних (MongoDB Atlas) зберігає інформацію про користувачів, чати та повідомлення. Двостороннє з'єднання між клієнтом і сервером забезпечується бібліотекою Socket.io через протокол WebSocket (рис. 1).

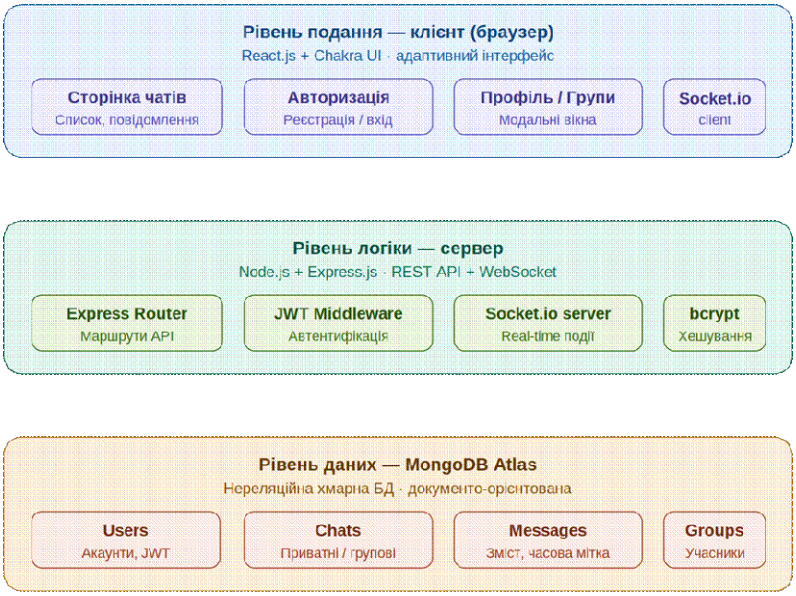

Рис. 1. Архітектура вебзастосунку Chatico (MERN + Socket.io)

Порівняльний аналіз рішень для обміну повідомленнями

На основі шести визначених критеріїв оцінювання проведено порівняльний аналіз існуючих месенджерів та розробленого вебзастосунку (табл. 1).

Таблиця 1

Порівняльна оцінка рішень для обміну повідомленнями

Критерій	Telegram	WhatsApp	Discord	Chatico
Веб-інтерфейс	Частково	Частково	Так	Так
Відкритий код	API	Ні	Ні	Так
Групові чати	Так	Так	Так	Так
Real-time повідомлення	Так	Так	Так	Так
Без реєстрації	Ні	Ні	Ні	Ні
Простота розширення	Середня	Низька	Низька	Висока

Практична реалізація вебзастосунку

Автентифікація та безпека. Система реєстрації та входу реалізована з хешуванням паролів через бібліотеку bcrypt та авторизацією на основі JWT (JSON Web Tokens). Токен зберігається на клієнті й додається до всіх захищених запитів; серверний middleware перевіряє права доступу до чатів і повідомлень.

Обмін повідомленнями в реальному часі. Socket.io встановлює двостороннє WebSocket-з'єднання між клієнтом і сервером. Повідомлення передаються у вигляді подій (send message, receive message), одночасно зберігаються у MongoDB Atlas та миттєво відображаються в інтерфейсі без перезавантаження сторінки. Затримка доставки у типових умовах — 0.1–0.3 секунди.

Чати та керування групами. Приватний чат ініціюється через пошук за ім'ям або email; повторний чат між тими самими користувачами не створюється. Груповий чат дозволяє будь-якому користувачу обрати декількох учасників, задати назву групи та отримати роль адміністратора. Усі операції виконуються через REST API з фіксацією в базі даних.

Інтерфейс користувача. Для побудови UI використано бібліотеку Chakra UI, що забезпечує адаптивну верстку, модальні вікна, індикатор набору тексту («typing...»), сповіщення про нові повідомлення та підтримку пристроїв із шириною екрана від 320 px.

Тестування та результати. Результати тестування представлені у табл. 2 нижче.

Таблиця 2

Результати тестування вебзастосунку Chatico

Тип тесту	Метод	Результат
Реєстрація / авторизація	Ручне	Успішно, з обробкою помилок
Відправлення повідомлень	Інтеграційне	В реальному часі, без затримок
WebSocket-з'єднання	Навантажувальне	Стабільне
Адаптивність UI	Ручне	Коректна на ПК, планшеті, смартфоні
Групові чати	Функціональне	Функціональні
Обробка виключень	Модульне (Postman)	Присутня

Функціональне тестування підтвердило виконання всіх функціональних вимог. Модульне тестування API проводилось за

допомогою Postman; навантажувальне — шляхом імітації одночасної роботи кількох користувачів із різних вкладок браузера. Усі критичні сценарії пройдено успішно; застосунок продемонстрував стабільну роботу в типових умовах.

ВИСНОВКИ

Розроблений вебзастосунок Chatico усуває ключові недоліки існуючих рішень: вперше реалізує повнофункціональний браузерний чат на відкритому стеку MERN з підтримкою реального часу (Socket.io), безпечною JWT-автентифікацією, приватними та груповими чатами й адаптивним дизайном (Chakra UI). Статична клієнтська частина (React.js) у зв'язці з динамічним серверним API (Node.js + Express) та нереляційною базою даних (MongoDB Atlas) забезпечує масштабованість і просте розширення функціоналу.

Результати тестування (стабільне WebSocket-з'єднання, затримка доставки 0.1–0.3 с, коректна адаптивність на всіх типах пристроїв) підтверджують практичну цінність та технічну якість розробленого рішення. Застосунок може бути використаний як основа для корпоративного чату, освітньої платформи або стартова точка для мобільного додатку (React Native / Flutter).

ДЖЕРЕЛА

1. Mozilla Developer Network (MDN). URL: https://developer.mozilla.org

2. React Documentation. URL: https://react.dev

3. Хлиста, І., & Бондарчук, А. (2023). Вирішення проблеми часткового оновлення вмісту веб-сторінки шляхом оптимізації параметрів SPA. Комп’ютерне моделювання та інформаційні технології, 286-291.

4. Бондарчук А.П. та Глушак О.М., Пронькін О. В., СтражніковА. А. (2025) Предиктивне управління оновленнями програмного забезпечення в інтернеті речей. Зв'язок (5). с. 13-17.

5. MongoDB Documentation. URL: https://www.mongodb.com/docs

6. Chakra UI Documentation. URL: https://chakra-ui.com

7. JSON Web Tokens (JWT). URL: https://jwt.io

8. WebSockets API (MDN). URL: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API

9. OWASP Foundation. URL: https://owasp.org

10. Express.js Documentation. URL: https://expressjs.com

11. Abramov, V., Astafieva, M., Boiko, M., Bodnenko, D., Bushma, A., Vember, V., Hlushak, O., Zhyltsov, O., Ilich, L., Kobets, N., Kovaliuk, T., Kuchakovska, H., Lytvyn, O., Lytvyn, P., Mashkina, I., Morze, N., Nosenko, T., Proshkin, V., Radchenko, S., & Yaskevych, V. (2021). Theoretical and practical aspects of the use of mathematical methods and information technology in education and science. https://doi.org/10.28925/9720213284km