УДК 004.78:654.9

РОЗРОБКА СЕРВЕРНОЇ ЧАСТИНИ МЕСЕНДЖЕРУ TELEGRAM

Аранжий Р.В.
e-mail: ruslan.aranzhyi@gmail.com
Науковий керівник – ст. викл. Олійник О.В.
Харківський національний університет радіоелектроніки, каф. ПІ
м. Харків, Україна

This thesis explores the development of an open-source server-side implementation inspired by Telegram. The project aims to create an independent environment for testing unofficial client applications, experimenting with messaging protocols. The proposed architecture includes a gateway for request handling, worker services for business logic processing, MySQL for persistent data storage, Redis for caching, and Taskiq with RabbitMQ or Redis as a task broker for asynchronous processing. The solution provides flexibility, scalability, and full control over data management and privacy, making it suitable for educational, research, and private deployment scenarios.

Telegram є одним із найпопулярніших месенджерів у світі та став важливою частиною повсякденного життя мільйонів людей [1]. Він використовується для особистого спілкування, роботи, навчання, ведення бізнесу, адміністрування спільнот і поширення інформації. Завдяки швидкості, зручності та широкому функціоналу платформи Telegram сформувалася ціла екосистема клієнтських застосунків, ботів і сервісів, що взаємодіють із нею. Такий масштаб поширення та інтеграції робить тему розробки серверної частини месенджера особливо актуальною.

У зв’язку з цим виникає ідея створення відкритої (open-source) реалізації серверної частини як альтернативного середовища. Такий сервер дозволяє моделювати взаємодію клієнта й бекенду, перевіряти роботу неофіційних застосунків, тестувати власні модифікації та експериментальні функції без прив’язки до офіційної інфраструктури. Для розробників це можливість відлагоджувати клієнтські застосунки, створювати й тестувати ботів, досліджувати механізми авторизації, обробки повідомлень і синхронізації даних між пристроями. Також це зручна платформа для навчання – студенти та інженери можуть практично ознайомитися з принципами побудови мережевих сервісів.

Для звичайних користувачів подібне рішення може застосовуватися як приватний сервер для невеликих спільнот, внутрішніх команд або організацій, де важливо мати окреме середовище для комунікації. Це може бути локальна інфраструктура для навчального закладу, компанії або дослідницької групи.

Окрім цього, використання власної серверної частини забезпечує повний контроль над даними та рівнем приватності. Усі повідомлення, облікові записи та журнали подій зберігаються на інфраструктурі,

яка адмініструється самостійно. Можна визначати політику зберігання інформації, налаштовувати механізми шифрування, резервного копіювання, обмеження доступу та моніторингу. Це дозволяє підвищити рівень конфіденційності та адаптувати систему відповідно до власних вимог безпеки.

Архітектура серверної частини будується за модульним принципом. Вхідною точкою є gateway, який приймає запити від клієнтів через протокол MTProto [2], що використовується у Telegram для організації захищеної та ефективної передачі даних. MTProto відповідає за встановлення сесії, шифрування повідомлень і їх серіалізацію, а в межах реалізації може працювати поверх TCP, HTTP або WebSocket залежно від способу підключення клієнта [3]. Gateway виконує первинну автентифікацію та маршрутизує трафік до внутрішніх сервісів. Основна бізнес-логіка реалізується через workers – окремі процеси, що відповідають за створення й доставку повідомлень, обробку чатів та інші операції. Для зберігання структурованих даних використовується MariaDB, де розміщуються облікові записи користувачів, повідомлення та метадані. Для прискорення роботи застосовується Redis як система кешування – він зберігає активні сесії, кешовані дані про користувачів, чати та інші тимчасові дані, зменшуючи навантаження на базу даних. Асинхронна обробка задач реалізується за допомогою Taskiq із використанням RabbitMQ та Redis як брокера повідомлень, що дозволяє ефективно розподіляти навантаження та підтримувати стабільність системи навіть при зростанні кількості запитів.

У результаті формується гнучка та серверна система, придатна для експериментів, тестування та побудови власних ізольованих комунікаційних рішень.

Список використаних джерел:

1. 700 Million Users and Telegram Premium. URL: https://telegram.org/blog/700-million-and-premium (дата звернення: 23.02.2026).

2. MTProto Mobile Protocol. URL: https://core.telegram.org/mtproto (дата звернення: 23.02.2026).

3. Transports. URL: https://core.telegram.org/mtproto/transports (дата звернення: 23.02.2026).