PHENIKAA UNIVERSITY

SCHOOL OF COMPUTING

Lab Report: Lab 1 - Requirements Elicitation & Modeling

Microservice Architecture for Real-Time Chat Application using Event-Driven and gRPC

Student Details: Hoàng Lê Đức Huy - 23010298 - Group 8

Nguyễn Hà Nguyên - 23010310 - Group 8

Nguyễn Đức Minh - 23010302 - Group 8

Course Info: Software Architecture

Ha Noi, December 5, 2025

Table of Contents

1. Abstract..2
2. Lab Specific Section: I. Requirements Elicitation & Modeling...........................2
2.1 Software Requirements Specifications (SRS)..2
2.1.1 Functional Requirements (FRs)...2
Functional behaviors the system must provide to its actors.............................. 2
2.1.2 Non-Functional Requirements (NFRs)...3
Quality attributes describing how the system must operate.............................. 3
2.1.3 Architecturally Significant Requirements (ASRs).................................... 4
Non-functional requirements that strongly shape architectural decisions...........4
2.2 Modeling Artifact: UML Use Case Diagram..5
3. Architectural Design...5
3.1 The Problem Statement.. 5
3.2 Impact of ASRs on Layered Architecture...5
4. Conclusion & Reflection...6

List of Tables

Table 1: Functional Requirements.. 2
Table 2: Non-Functional Requirements...3
Table 3: Architecturally Significant Requirements.. 4

List of Figures

Figure 1: Use Case Diagram...5

1. Abstract

The Real-Time Chat Application project aims to develop a robust communication platform. This initial lab focused on requirements elicitation and modeling. We documented essential Functional, Non-Functional, and Architecturally Significant Requirements (ASRs) in a structured format. The core system behavior was formally modeled using UML Use Case Diagram, clearly defining system boundaries, external actors (such as end users and system administrators), and key behavioral relationships (e.g., the inclusion of User Authentication within the Send Message use case).

2. Lab Specific Section: I. Requirements Elicitation & Modeling

2.1 Software Requirements Specifications (SRS)

The following tables document the elicited requirements for the Real-Time Chat Application. These requirements form the basis for subsequent system modeling and architectural design.

2.1.1 Functional Requirements (FRs)

Functional behaviors the system must provide to its actors.

Table 1: Functional Requirements

ID	Description	Priority
FR-01	The system must allow a User to sign up, sign in, and log out securely.	Critical
FR-02	The system must allow a User to send and receive real-time messages.	Critical
FR-03	The system must allow a User to send files and media attachments.	High
FR-04	The system must allow a User to join group chats and interact with other members.	High
FR-05	The system must allow a User to add friends and manage their contacts.	Medium
FR-06	A Group Admin must be able to create group chats and add members.	High
FR-07	A Group Admin must be able to forward, reply to, or moderate messages within their group.	Medium
FR-08	The System Administrator must be able to view all users and manage their accounts (ban/unban).	High

FR-09	The System Administrator must be able to manage chat rooms, including deleting groups.	High
FR-10	The System Administrator must be able to monitor basic system status.	Medium
FR-11	The system must validate message format and access permissions before sending.	High
FR-12	The system must authenticate users before granting access to messaging features.	Critical

2.1.2 Non-Functional Requirements (NFRs)

Quality attributes describing how the system must operate.

Table 2: Non-Functional Requirements

ID	Attribute	Description	Impact
NFR-01	Performance	The system must deliver messages with low latency suitable for real-time communication.	Critical
NFR-02	Security	All message content and user credentials must be encrypted in transit (TLS) and at rest.	Critical
NFR-03	Availability	The system must achieve at least 99% uptime to ensure continuous communication.	High
NFR-04	Reliability	Delivered messages must be stored durably to prevent loss after system failures.	High
NFR-05	Usability	The chat interface must provide a clear, responsive user experience across devices.	Medium
NFR-06	Scalability	The system must support thousands of concurrent WebSocket connections during peak usage.	High

2.1.3 Architecturally Significant Requirements (ASRs)

Non-functional requirements that strongly shape architectural decisions.

Table 3: Architecturally Significant Requirements

ASR ID	**Quality Attribute**	**Requirement Statement**	**Architectural Rationale**
ASR-01	Scalability	The system must support large numbers of concurrent real-time connections without degradation of performance.	Drives the use of event-driven, horizontally scalable real-time communication components and load balancing.
ASR-02	Security	Authentication and authorization must be enforced for all message-related operations.	Requires a dedicated security component and centralized token validation to protect user data and group access.
ASR-03	Modifiability	Adding new message types (e.g., emojis, reactions, voice notes) must not require changes to user management or group management modules.	Enforces separation of concerns and modular design for extensibility of messaging components.

2.2 Modeling Artifact: UML Use Case Diagram

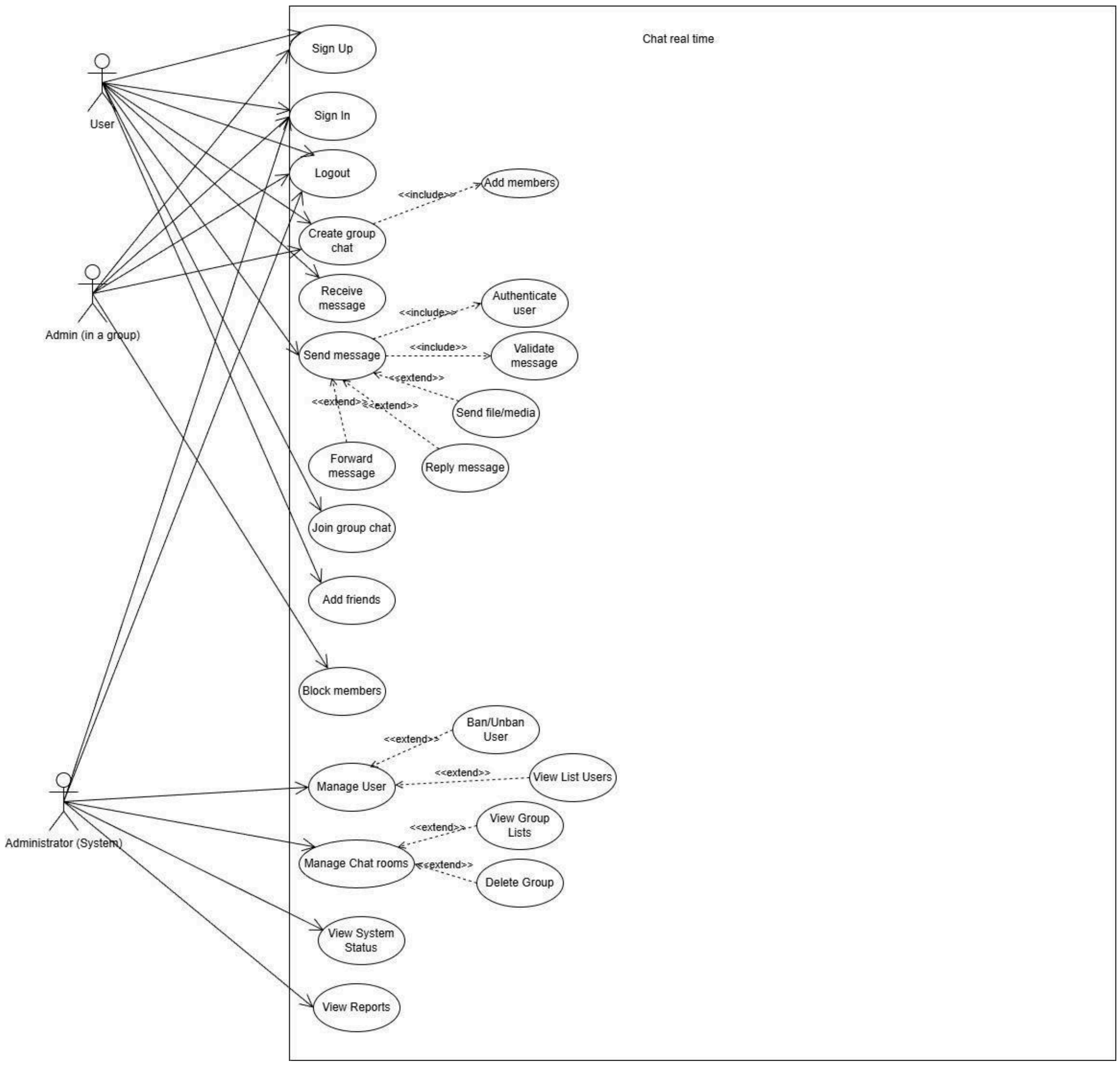

Figure 1: Use Case Diagram

3. Architectural Design

The requirements elicited in Lab 1 have a direct impact on the design challenge of Lab 2: Layered Architecture Design.

3.1 The Problem Statement

The problem is to design an architecture that satisfies the core requirements while enforcing separation of concerns and strict dependency rules. The Layered Architecture pattern is chosen as the first structural approach to address the Modifiability (ASR-03) and Security (ASR-02) ASRs identified in Lab 1.

3.2 Impact of ASRs on Layered Architecture

- ASR-03 (Modifiability) → Layered Structure: ASR-03 demands that component changes remain isolated (e.g., adding a new feature like Voice

Notes must not affect the user management code). The Layered Architecture inherently supports this by a layered structure separating presentation, business logic, and data management concerns. This means:

  - The Business Logic (where message processing/feature logic resides) can be modified (e.g., updating media handling) without impacting the Presentation Layer (the user interface).
  - The Persistence Layer (e.g., switching the database for storing messages) can be changed without affecting the Business Logic Layer, provided the component interfaces remain consistent.

- ASR-02 (Security) → Business Logic Layer: ASR-02 requires rigorous authorization for admin tasks (like managing accounts) and message-related operations. In the Layered Architecture, this logic is enforced in the Business Logic Layer. This ensures that security checks are applied to the core services (e.g., UserService, MessageService), protecting them from unauthorized access regardless of whether the request originates from a web UI (Presentation Layer) or an internal script.

The next step (Lab 2) will use these requirements to formally define the four architectural layers and model the logical components (Controller, Service, Repository) and their interfaces within those layers.

4. Conclusion & Reflection

The requirements elicitation phase for the Real-Time Chat Application successfully defined the project scope through detailed Functional, Non-Functional, and Architecturally Significant Requirements (ASRs). The UML Use Case Diagram provides a clear visual model of user interactions and the critical paths for messaging and administration.

The identified ASRs—especially those relating to Modifiability (ASR-03) and Security (ASR-02)—directly necessitate the adoption of a structured pattern like the Layered Architecture for the subsequent design phase (Lab 2). This ensures the resulting architecture can sustainably support the required system behaviors, maintain a clear separation of concerns, and allow for future feature extensibility without major system overhauls.