Product Requirements Document (PRD)

Realtime Chat Application

A web-based one-to-one realtime chat application with OTP-based authentication, presence, typing indicators, message history retention for one week, and deployment on a domain with HTTPS. Target scale is 100 users for learning, with a production-ready design philosophy.

Objectives

- OTP login via phone number with delivery over email.
- Realtime chat with low latency.
- Presence, typing indicators, and read receipts.
- Seven-day chat history with automatic retention purging.
- Production-capable deployment over Kubernetes with HTTPS.

User Personas

- **Registered User**: authenticated, able to chat.
- **Guest**: cannot chat until authenticated.
- **Administrator (optional)**: inspects system health/status.

User Stories

Authentication

- Enter phone number to request login.
- Receive OTP via email.
- Verify OTP to access application.
- Remain logged in until logout or expiry.

Contacts and Presence

- View contact list.
- View realtime online/offline presence.

Messaging

- One-to-one chat.
- Realtime send and receive without refresh.
- Typing indicators and read receipts.
- Seven-day message history.

Detailed Requirements

Authentication and Sessions

- Phone number + OTP login.
- OTP delivery via SMTP provider.
- Server-side session stored in Redis.
- OTP throttling and rate limits.

Messaging Architecture

- Transport: WebSockets.
- Broker: Kafka for decoupled event flow.
- Message fields: sender, recipient, timestamp, read status, payload text.

Presence and Typing

- Presence tracked in Redis with expirations.
- Typing events via WebSocket.
- Read receipts via view events.

History and Retention

- Messages persisted in Postgres.
- Retention period: seven days.
- Cleanup via weekly scheduled cron.

Privacy and Security

- Mandatory HTTPS.
- Sensitive phone number handling.
- CSRF protection for REST endpoints.
- Input sanitization for messages.
- No public discovery of users.

Non Functional Requirements

Performance

- Presence and typing latency under 300 ms.
- Message delivery under 500 ms for 100 users.

Scalability

- Kafka and Kubernetes enable modular scaling.

Availability

- Graceful WebSocket reconnection.

Reliability

- Persist-before-ack for messages where feasible.
- Kafka prevents loss during gateway failures.

UI Requirements

- Vue + Vuetify chat UI.
- Contact sidebar and chat panel.
- Presence indicators and typing status.
- Timestamps and read receipts.
- OTP screen and session flows.

Constraints

Frontend: Vue 3 + Vuetify + Pinia + WebSocket.
Backend: Spring Boot + Kafka + Redis + Postgres + SMTP.
Infra: k3s + Traefik + Docker + HTTPS + domain.

Out of Scope

Group chats, attachments, audio/video, mobile app, SMS notifications, contact sync.

Acceptance Criteria

OTP login operational. Realtime chat operational. Presence, typing, receipts functional. Seven-day retention. Automated purge. Production deployment with HTTPS. Stable for 100 users.