Implementation Plan

Realtime Chat Application

This implementation plan converts technical design into actionable development phases, milestones, dependencies, and acceptance conditions for the realtime chat application.

Scope and Goals

Goal: deliver a production-capable realtime chat with OTP auth, one-to-one messaging, presence, typing, read receipts, seven-day retention, and HTTPS deployment on Kubernetes. Non-goals: group chat, attachments, mobile push, mobile clients.

Delivery Strategy

Incremental modular delivery with early infrastructure setup to allow continuous integration and end-to-end testing.

Milestones

- M1: Auth + session
- M2: Messaging end-to-end
- M3: Presence + typing + receipts
- M4: Retention + UI polish
- M5: Deployment + observability

Work Breakdown Structure

Phase A: Foundations

A1 Repo scaffolding (frontend + backend)
A2 Dependency setup (Spring Boot, Vue, Vuetify, Pinia)
A3 Dev services via Docker Compose (Postgres, Redis, Kafka, SMTP)
A4 k3s cluster bootstrap with Traefik + LetsEncrypt
A5 CI pipeline for Docker image builds

Phase B: Auth and Sessions

B1 OTP generation + throttle
B2 SMTP integration for OTP emails
B3 Redis session store
B4 CSRF for REST
B5 Frontend login + OTP forms
B6 Session persistence + logout

Definition of Done: OTP login with protected sessions.

Phase C: Messaging Infrastructure

C1 WebSocket gateway
C2 Kafka producer/consumer
C3 Message schema
C4 Delivery pipeline (send → Kafka → persist → fanout)
C5 WebSocket client

Definition of Done: end-to-end text messaging.

Phase D: Presence, Typing, Receipts

D1 Presence heartbeat via Redis TTL
D2 Typing indicators via WebSocket
D3 Read receipts persisted to DB
D4 Frontend state sync

Definition of Done: presence + typing + receipts functional.

Phase E: History and Retention

E1 History queries
E2 Pagination/time-based fetching
E3 Weekly retention purge (¿7 days)
E4 Scrollback behavior

Definition of Done: seven-day history + auto cleanup.

Phase F: Deployment and Domain

F1 Image publishing pipeline
F2 k3s manifests for services
F3 Traefik + HTTPS + WSS
F4 Domain setup

Definition of Done: public HTTPS app with WebSocket.

Phase G: Observability and QA

G1 Structured logs
G2 Optional metrics (Prometheus + Grafana)
G3 Functional testing (auth, messaging, retention)
G4 Load test for 100 concurrent users

Definition of Done: stable under expected load.

Dependencies and Sequencing

Hard deps: OTP → Session → WebSocket, Kafka → DB persist, Redis → presence, TLS → WSS.
Soft deps: UI polish after functional paths, observability after infra.

Risks and Mitigations

WebSocket TLS misconfig → test early
Kafka complexity → single broker first
SMTP latency → async delivery
State inconsistencies → idempotent consumer writes
Cluster overkill → single-node k3s

Definition of Done

Feature complete, reproducible deployment, HTTPS functional, messaging verified, retention verified, load tested, observable, no major defects.

Acceptance Tests

OTP failure and expiry handling
Realtime latency ¡500ms
Presence and typing with reconnection
Read receipts correctness
Retention policies enforced
Public domain with WSS
Load test at 100 concurrent users