Secure Messenger Desktop — Test Project Submission

Candidate: Eric Gitangu | Senior Software Engineer Architect | 10+ years
Email: developer.ericgitangu@gmail.com **Phone:** +1 (978) 710-9475 **Portfolio:** developer.ericgitangu.com **Resume:** resume.ericgitangu.com

---

GitHub Repository

Public Repository: github.com/ericgitangu/secure-messenger-desktop

Live Demo: secure-messenger-desktop.fly.dev

Figure 1: QR Code — Scan to open GitHub repo

Tech Stack

Technology	Version	Purpose
TypeScript	5.7	Strict type safety across entire codebase
Electron	40	Native desktop shell with IPC bridge
React	19	UI framework with hooks
Redux Toolkit	2.5	State management (createSlice, createAsyncThunk)
SQLite (better-sqlite3)	WAL Mode	Encrypted local database with C++ bindings
AES-256-GCM	Node.js crypto	Authenticated encryption for all message bodies
WebSocket (ws)	Native	Real-time bidirectional messaging
Prometheus (prom-client)	Apache-2.0	Counters, histograms, gauges for observability
Grafana	10.4	12-panel auto-provisioned monitoring dashboard
Docker Compose	V2	Full-stack orchestration (app + Prometheus + Grafana)
Vite	5.4	Build tool for all 5 entry points
Vitest	81 tests	Unit, integration, and contract (Pact) tests
Fly.io	Frankfurt	Production deployment (single-container PaaS)
pnpm	10.x	Package manager with content-addressable store

Architecture Overview

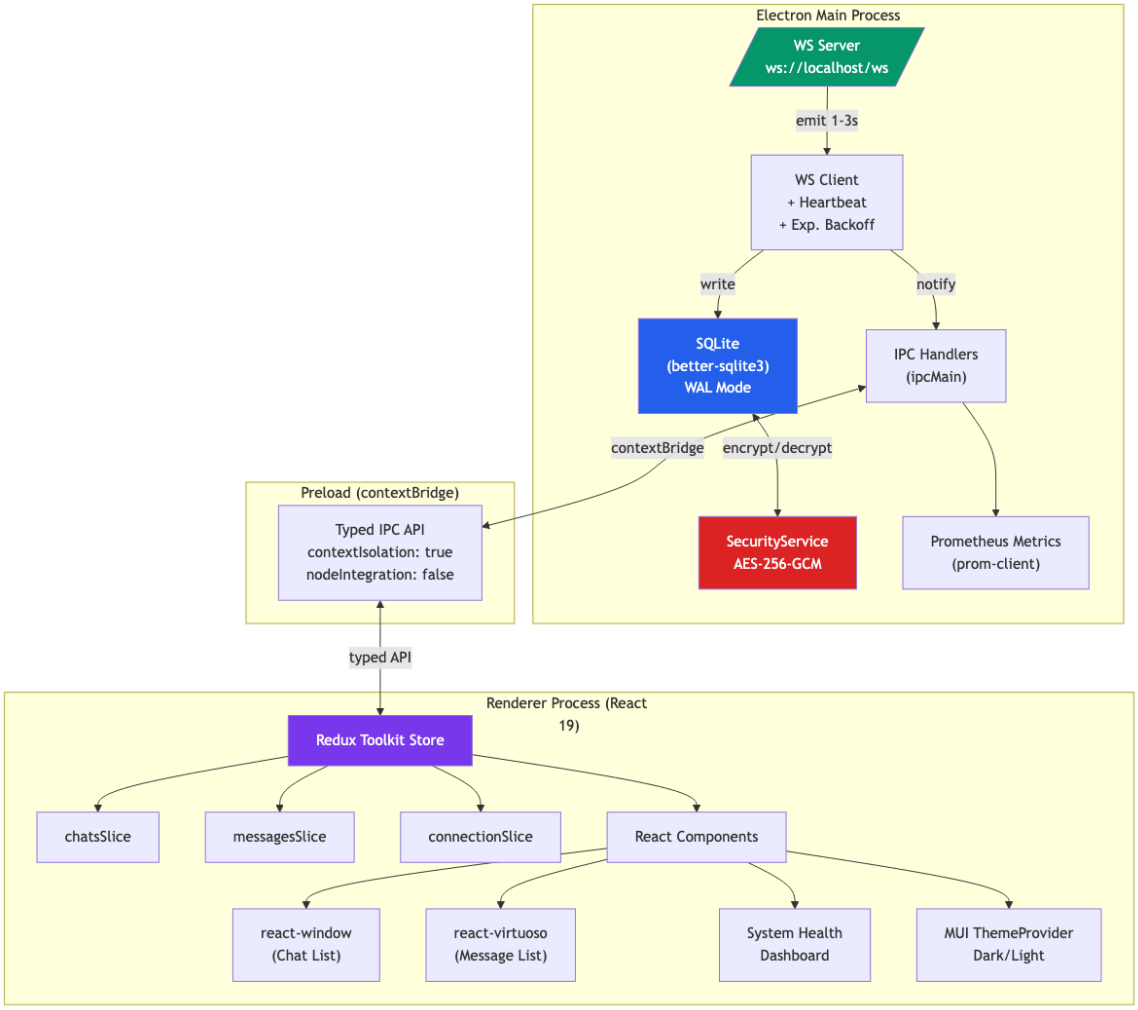

Figure 2: Architecture Diagram

Data Flow

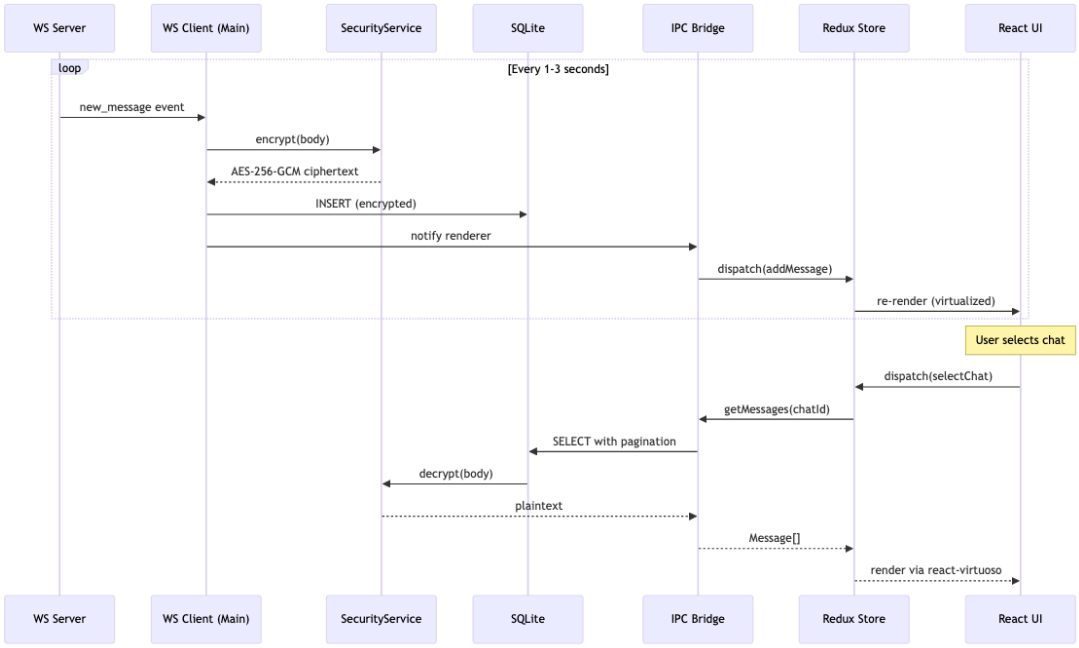

Figure 3: Data Flow — Sequence Diagram

Connection State Machine

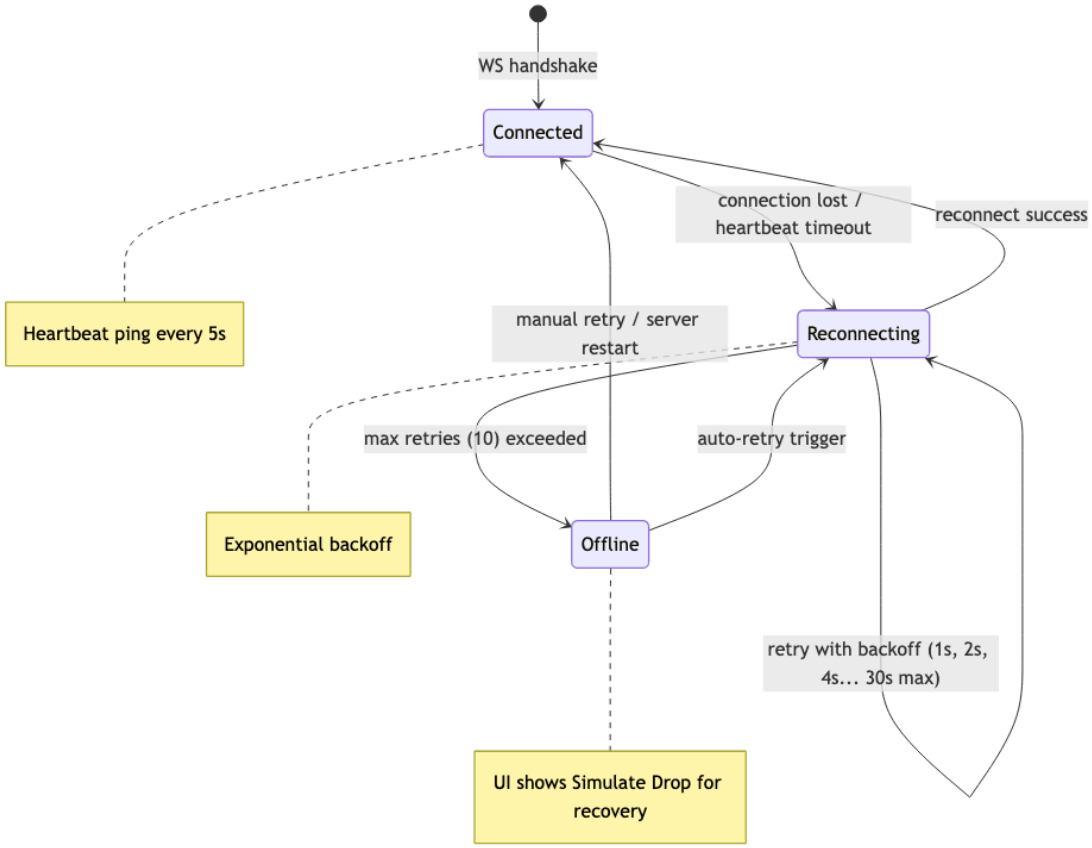

Figure 4: Connection State Machine

AES-256-GCM Encryption Boundary

Figure 5: Encryption Boundary

Project Structure

```
secure-messenger-desktop/
├── src/
│   ├── main/                  # Electron main process
│   │   ├── db/                # SQLite schema, queries, seed
│   │   ├── metrics/           # Prometheus MetricsCollector
│   │   ├── security/          # AES-256-GCM SecurityService
│   │   └── ws/                # WebSocket server + client
│   ├── renderer/              # React 19 UI
│   │   ├── api/               # Bridge abstraction (IPC/HTTP)
│   │   ├── components/        # ChatList, MessageView, Health
```

```
│   │   ├── hooks/            # useIpc, useSystemHealth, usePrometheusMetrics
│   │   ├── store/            # Redux slices (chats, messages, connection)
│   │   ├── theme/            # MUI dark/light theme
│   │   └── types/            # IPC type contracts
│   ├── server/               # Express web server (browser mode)
│   ├── shared/               # Constants shared across processes
│   ├── main.ts               # Electron entry point
│   └── preload.ts            # contextBridge IPC API
├── tests/
│   ├── unit/                 # 63 unit tests (security, DB, store)
│   ├── integration/          # 4 WS integration tests
│   └── pact/                 # 14 contract tests
├── docker/                   # Grafana dashboards + Prometheus config
├── scripts/                  # Native module rebuild, full-stack start
├── Dockerfile                # Multi-stage production build
├── docker-compose.yml        # App + Prometheus + Grafana
├── fly.toml                  # Fly.io deployment config
└── README.md                 # Comprehensive documentation
```

Key Features Implemented

Functional Requirements (39/39 complete)

A) SQLite Local Storage — 3NF schema, WAL mode, cursor-based pagination, 3 targeted indexes, 200 chats + 20K encrypted messages seeded

B) WebSocket Sync — Real-time streaming every 1-3s, Pact contract-validated event format, unidirectional data flow

C) Connection Health — 3-state machine (Connected/Reconnecting/Offline), exponential back-off (1s⟶30s), heartbeat ping every 5s, simulate disconnect for testing

D) UI Performance — react-window (chat list), react-virtuoso (messages), React.memo + use-Callback, debounced search (300ms)

E) User Actions — Create new chat, send messages, both AES-256-GCM encrypted before storage

F) Security — AES-256-GCM with 96-bit IV + 128-bit auth tag, contextIsolation, no-console ESLint rule, tamper detection

G) Observability — Built-in Grafana-style dashboard, Prometheus metrics, live polling, Docker Grafana with 12 auto-provisioned panels

H) Testing — 81 tests (63 unit + 4 integration + 14 contract), CI/CD with GitHub Actions

I) DevOps — Docker Compose, Fly.io deployment, automated native module rebuild, git hooks (pre-commit/pre-push)

Quick Start

```
# Clone and run
git clone https://github.com/ericgitangu/secure-messenger-desktop.git
cd secure-messenger-desktop
pnpm install
pnpm dev              # Electron native app
pnpm server:dev       # Browser mode at http://localhost:3000

# Docker (full observability stack)
docker compose up --build
# App: localhost:3000 | Prometheus: localhost:9090 | Grafana: localhost:3001

# Tests
pnpm test             # 81 tests
pnpm lint             # ESLint strict mode
```

Test Results

Type	Tests	Coverage
Unit (Security)	16	AES-256-GCM round-trip, tamper detection, wrong key, batch, unicode
Unit (DB)	26	Pagination, search, seeding, FK constraints, indexes
Unit (Store)	21	All reducers, state machine transitions, action creators
Integration (WS)	4	Server → client → DB round-trip
Contract (Pact)	14	WS event schema validation, discriminated unions
Total	**81**	**All passing**

Generated: February 9, 2026 Repository: github.com/ericgitangu/secure-messenger-desktop