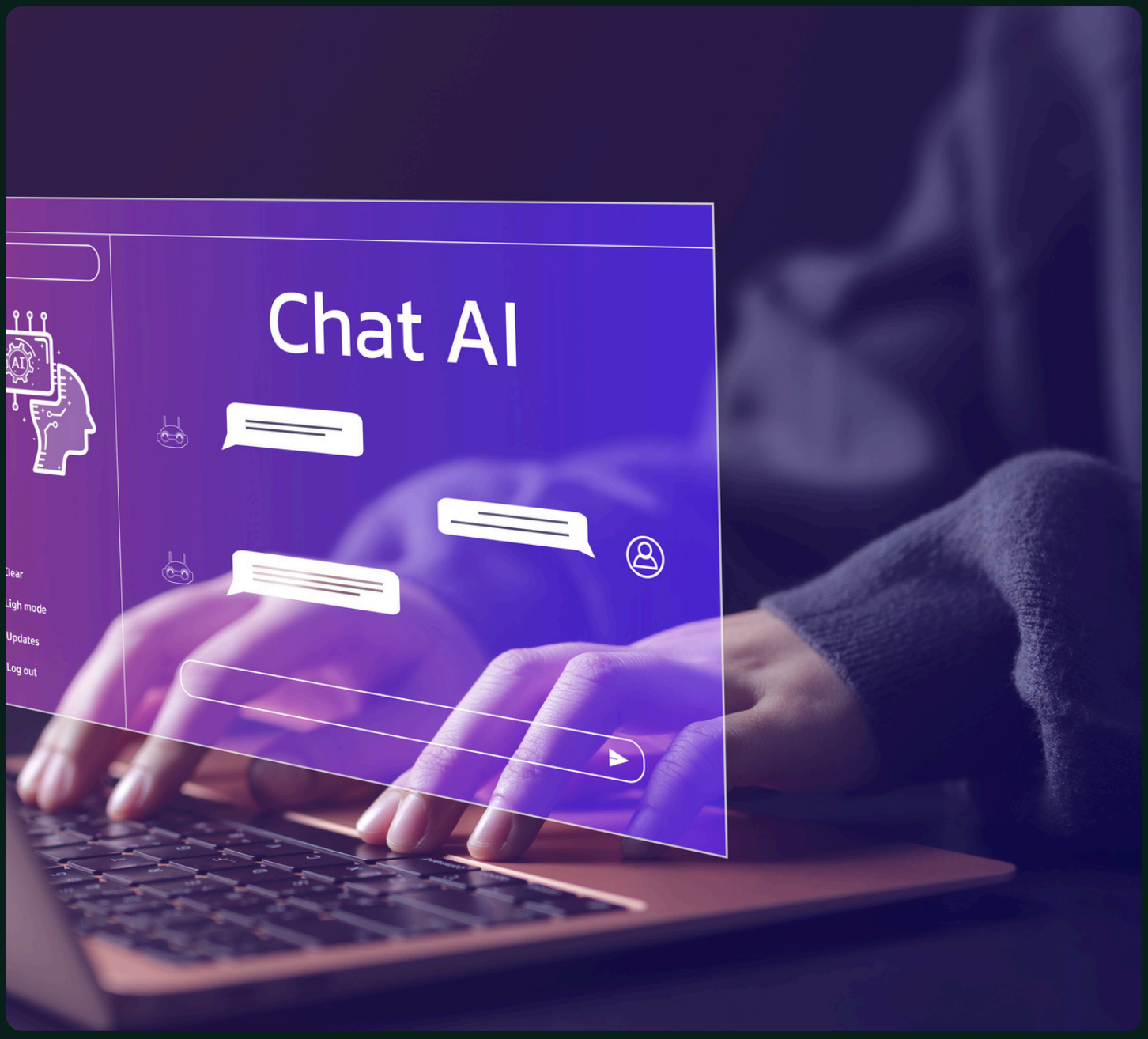
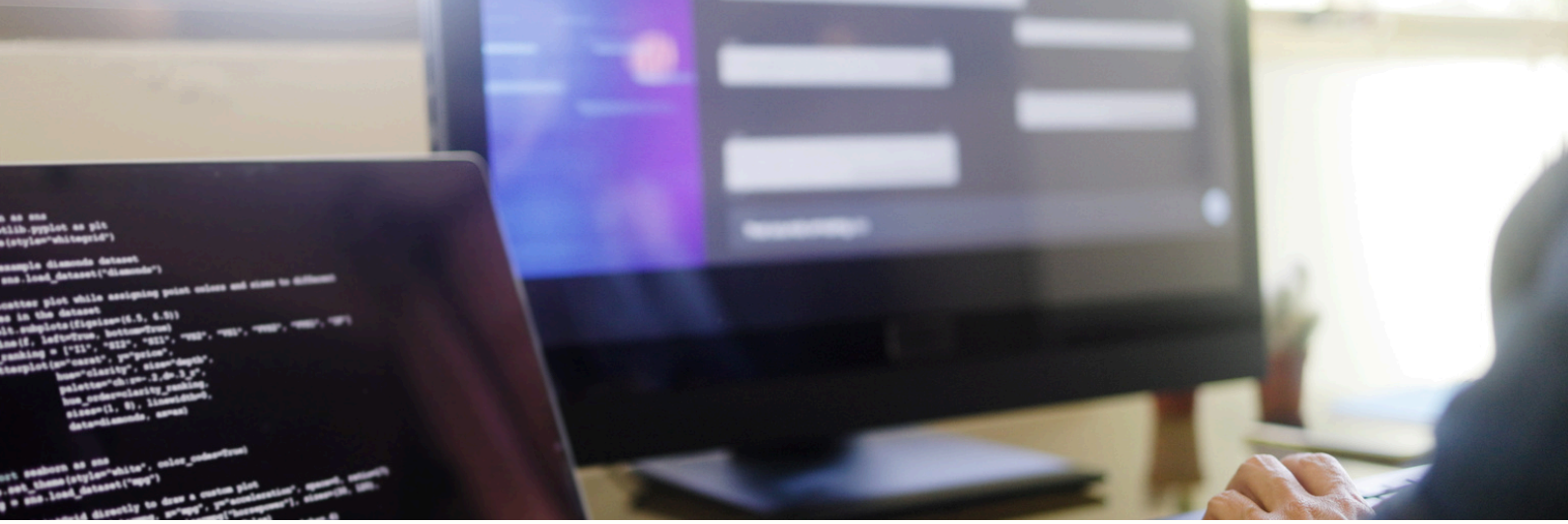
Digit88

Digit88 Technologies
www.digit88.com
info@digit88.com
+91 99011 59096

CASE STUDY 2025

Modernizing a **Distributed Communication System** for Resilience and Scalability





Highlights

- Transition to **Protocol Buffers** (Protobuf) to serialize data efficiently, reducing message payload sizes and network latency compared to JSON.
- Integration of **gRPC** for internal service-to-service communication using Protobuf.
- Adoption of **Apache Kafka** as the primary distributed message broker to enable reliable, scalable, and decoupled message exchange.

Client

The Client is a global leader in **conversational AI and messaging solutions**, enabling brands to deliver human-like conversations across multiple channels. With a mission to make life easier for people and brands everywhere through trusted conversational AI, the Client serves major enterprises worldwide.

Challenge

The legacy architecture faced significant challenges in maintaining consistency and resilience across distributed services. Services within the ecosystem used heterogeneous communication protocols:

- **WebSocket** was used between Service A and B.
- A **queue-based mechanism** connected Service B and C.
- Communication from Service C back to Service A had to traverse through intermediate services, leading to inefficient message routing.

This complex and fragmented messaging flow introduced instability, especially during traffic surges or partial network/platform failures. The lack of fault tolerance and a consistent messaging protocol made the system difficult to scale and maintain.

Outcome

- **Reduced Latency & Payload Overhead** using Protobuf over JSON.
- **Improved System Resilience** with Kafka handling retries and backpressure.
- **Increased Scalability** through asynchronous, decoupled messaging.
- **Unified Protocol Stack** simplifying service development and deployment.

Strategy & Solution

To address these limitations**, a complete communication overhaul was implemented** with the following goals:

Unify the messaging strategy using a consistent protocol and modern message serialization formats.

Replace legacy queueing and WebSocket dependencies with scalable, distributed messaging systems.

Improve fault tolerance and traffic surge handling through asynchronous, resilient communication mechanisms.

Technologies Deployed

- **Node.js:** Backend service runtime
- **Protobuf & gRPC:** Efficient and typed service communication
- **Apache Kafka:** Scalable and distributed messaging backbone
- **Avro:** Schema management for Kafka message streams
- **Prometheus & Grafana:** Monitoring and observability
- **Nginx:** Reverse proxy and traffic routing
- **PM2:** Process manager for Node.js services