Multi-Tenant Messaging Provider Integration

Complete Implementation Guide

MSG91 + Twilio Integration with Pay-As-You-Go Support

Version: 1.0

Comprehensive guide for building a Spotify-style BYOA (Bring Your Own Account) messaging platform

# Table of Contents

1. Executive Summary

2. Architecture Overview

3. Provider Comparison: MSG91 vs Twilio

4. Database Schema

5. Backend Implementation

6. Frontend Implementation

7. Security Best Practices

8. Deployment Checklist

9. Testing Guide

10. Troubleshooting

# 1. Executive Summary

## Project Overview

This implementation guide provides a complete solution for building a multi-tenant SaaS platform where users can connect their own messaging provider accounts (MSG91 or Twilio) and pay the provider directly with no markup from your platform.

## Key Features

* • Multi-provider support (MSG91 and Twilio)
* • Bring Your Own Account (BYOA) model
* • Secure credential encryption using AES-256-GCM
* • Real-time credential validation
* • Automatic webhook handling for delivery status
* • Usage tracking and analytics
* • Support for SMS, WhatsApp, Email, and Voice

## Why MSG91 + Twilio?

Strategic Coverage:

* • Twilio: Industry standard, global coverage, enterprise trust
* • MSG91: Best pricing for UAE/Middle East/Asia, startup-friendly

Market Segmentation:

* • Enterprise/Global clients will choose Twilio (brand trust)
* • Startups/Regional businesses will choose MSG91 (cost savings)

# 2. Architecture Overview

## System Flow

1. User signs up on your platform

2. User navigates to Settings → Integrations

3. User selects provider (MSG91 or Twilio)

4. User enters their API credentials

5. System validates credentials in real-time

6. Credentials are encrypted and stored securely

7. When user sends messages, your platform uses their credentials

8. Provider bills user directly

## Technology Stack

Backend:

* • Node.js + Express
* • PostgreSQL database
* • AES-256-GCM encryption for credentials

Frontend:

* • React.js
* • Modern UI components

External APIs:

* • Twilio API
* • MSG91 API

# 3. Provider Comparison: MSG91 vs Twilio

## Pricing Comparison

|  |  |  |
| --- | --- | --- |
| Feature | MSG91 | Twilio |
| SMS (UAE) | ~$0.003-0.004 | ~$0.0075-0.0079 |
| WhatsApp | Meta pricing only | $0.005 + Meta fees |
| Voice Calls | Custom pricing | From $0.0085/min |
| Global Coverage | Strong in Asia/ME | 200+ countries |
| Best For | UAE, Startups, Budget | Global, Enterprise |

## Key Features

MSG91 Strengths:

* • Most affordable SMS rates for Middle East/Asia
* • Excellent startup support program
* • Built-in business tools (Hello, Campaign, Segmento)
* • Excel and Google Sheets plugins
* • Regional language support

Twilio Strengths:

* • Industry-leading documentation
* • 99.95% uptime SLA
* • Global brand recognition and trust
* • Advanced voice features (IVR, conferencing)
* • Official WhatsApp Business Solution Provider

# 4. Database Schema

## PostgreSQL Schema

### Users Table

CREATE TABLE users (

id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

email VARCHAR(255) UNIQUE NOT NULL,

created\_at TIMESTAMP DEFAULT NOW()

);

### Messaging Providers Table

CREATE TABLE messaging\_providers (

id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_id UUID REFERENCES users(id) ON DELETE CASCADE,

provider\_name VARCHAR(50) NOT NULL,

is\_active BOOLEAN DEFAULT true,

credentials\_encrypted TEXT NOT NULL,

created\_at TIMESTAMP DEFAULT NOW(),

updated\_at TIMESTAMP DEFAULT NOW(),

last\_verified\_at TIMESTAMP,

UNIQUE(user\_id, provider\_name)

);

### Message Logs Table

CREATE TABLE message\_logs (

id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_id UUID REFERENCES users(id),

provider\_id UUID REFERENCES messaging\_providers(id),

message\_type VARCHAR(20),

destination VARCHAR(255),

status VARCHAR(50),

external\_id VARCHAR(255),

cost\_usd DECIMAL(10, 6),

created\_at TIMESTAMP DEFAULT NOW()

);

### Indexes

CREATE INDEX idx\_provider\_user ON messaging\_providers(user\_id);

CREATE INDEX idx\_message\_logs\_user ON message\_logs(user\_id);

CREATE INDEX idx\_message\_logs\_created ON message\_logs(created\_at);

# 5. Backend Implementation

## Encryption Module (config/encryption.js)

This module handles secure encryption and decryption of user credentials using AES-256-GCM:

const crypto = require('crypto');

const ENCRYPTION\_KEY = process.env.ENCRYPTION\_KEY;

const ALGORITHM = 'aes-256-gcm';

Key Features:

* • Uses AES-256-GCM for authenticated encryption
* • Generates random IV for each encryption
* • Includes authentication tag for integrity verification

## Messaging Service (services/messagingProviders.js)

Core service that handles all provider interactions:

Key Methods:

* • validateCredentials(provider, credentials) - Validates API credentials before saving
* • validateTwilio(credentials) - Tests Twilio credentials by fetching account info
* • validateMsg91(credentials) - Tests MSG91 credentials with balance check
* • sendSMS(userId, to, message) - Main method to send SMS through user's provider
* • sendViaTwilio(credentials, to, message) - Twilio-specific implementation
* • sendViaMsg91(credentials, to, message) - MSG91-specific implementation
* • logMessage(data) - Logs message delivery for analytics

## API Routes (routes/messaging.js)

RESTful API endpoints:

POST /api/messaging/providers

* • Save or update provider credentials
* • Validates credentials before storing
* • Encrypts credentials with AES-256-GCM

GET /api/messaging/providers

* • Retrieve user's configured providers
* • Returns provider name and status (does not expose credentials)

POST /api/messaging/send-sms

* • Send SMS through user's configured provider
* • Automatically selects active provider

POST /api/messaging/webhooks/twilio

* • Webhook endpoint for Twilio delivery status
* • Updates message status in database

POST /api/messaging/webhooks/msg91

* • Webhook endpoint for MSG91 delivery status
* • Updates message status in database

# 6. Frontend Implementation

## React Component Structure

### ProviderSelector Component

Main component that handles provider selection and configuration:

Key Features:

* • Visual provider comparison cards with pricing
* • Feature highlights for each provider
* • Dynamic credential form based on selected provider
* • Real-time validation feedback
* • Connection status indicators
* • Setup guide links for each provider

User Flow:

1. User views provider comparison cards

2. User clicks 'Connect' on preferred provider

3. Credentials form appears with provider-specific fields

4. User enters credentials from their provider account

5. System validates credentials in real-time

6. Success confirmation shown with balance (for MSG91)

7. Provider card updated with 'Connected' badge

### Provider Configuration Data

Each provider has detailed configuration including:

Twilio Configuration:

* • Required fields: Account SID, Auth Token, Phone Number
* • Features: Global reach, WhatsApp API, 99.95% uptime
* • Pricing: From $0.0079/SMS, $0.005/WhatsApp
* • Best for: Global businesses and enterprises

MSG91 Configuration:

* • Required fields: Auth Key, Sender ID
* • Features: Best Middle East pricing, Startup support, Business tools
* • Pricing: From $0.003/SMS, Meta pricing for WhatsApp
* • Best for: UAE businesses, startups, budget-conscious

# 7. Security Best Practices

## Credential Protection

Encryption:

* • Use AES-256-GCM for all credential storage
* • Generate random IV for each encryption operation
* • Store authentication tags for integrity verification
* • Never log or display decrypted credentials

Key Management:

* • Store ENCRYPTION\_KEY in environment variables only
* • Never commit encryption keys to version control
* • Use 32-byte (256-bit) keys generated with crypto.randomBytes
* • Implement key rotation strategy for production

Access Control:

* • Authenticate all API requests with JWT tokens
* • Validate user ownership before accessing credentials
* • Implement rate limiting on credential validation endpoints
* • Log all credential access attempts

## API Security

Input Validation:

* • Validate all user inputs before processing
* • Sanitize phone numbers and message content
* • Verify provider credentials format before validation
* • Use parameterized queries for database operations

Webhook Security:

* • Verify webhook signatures from Twilio
* • Use HTTPS for all webhook endpoints
* • Implement IP whitelisting for webhook sources
* • Log all webhook events for audit trail

## Database Security

* • Use SSL/TLS for database connections
* • Implement row-level security for multi-tenant isolation
* • Regular database backups with encryption
* • Separate read-only users for analytics queries

# 8. Deployment Checklist

## Pre-Deployment

Environment Setup:

☐ Generate and securely store ENCRYPTION\_KEY (32 bytes)

☐ Set up PostgreSQL database with SSL

☐ Configure environment variables (.env file)

☐ Set up JWT\_SECRET for authentication

☐ Configure CORS settings for frontend domain

Database:

☐ Run database migrations (create tables and indexes)

☐ Test database connectivity

☐ Set up database backups

☐ Configure connection pooling

Dependencies:

☐ Install Node.js dependencies (npm install)

☐ Verify twilio package installed

☐ Verify axios package installed

☐ Test all API endpoints locally

## Production Deployment

Infrastructure:

☐ Set up SSL/TLS certificates (HTTPS required)

☐ Configure load balancer if needed

☐ Set up logging service (e.g., Winston, Bunyan)

☐ Configure monitoring and alerts

☐ Set up error tracking (e.g., Sentry)

Provider Configuration:

☐ Register webhook URLs with Twilio

☐ Register webhook URLs with MSG91

☐ Test webhook endpoints are publicly accessible

☐ Verify webhook signature validation works

Security:

☐ Enable rate limiting on all endpoints

☐ Configure firewall rules

☐ Set up DDoS protection

☐ Enable database connection encryption

☐ Review and rotate all API keys

## Post-Deployment

☐ Test Twilio credential validation end-to-end

☐ Test MSG91 credential validation end-to-end

☐ Send test SMS through both providers

☐ Verify webhook delivery status updates

☐ Check database logging is working

☐ Monitor error logs for 24 hours

☐ Set up automated health checks

# 9. Testing Guide

## Unit Testing

Encryption Module Tests:

* • Test encryption produces different outputs for same input
* • Test decryption recovers original data
* • Test authentication tag validation
* • Test error handling for corrupted data

Provider Validation Tests:

* • Test Twilio validation with valid credentials
* • Test Twilio validation with invalid credentials
* • Test MSG91 validation with valid credentials
* • Test MSG91 validation with invalid credentials
* • Test network error handling

## Integration Testing

Provider Configuration Flow:

1. Create test user account

2. Submit Twilio credentials via API

3. Verify credentials are encrypted in database

4. Retrieve provider list and verify Twilio is shown as active

5. Send test SMS and verify it's sent via Twilio

6. Verify message is logged in database

Message Sending Flow:

* • Test SMS sending with Twilio provider
* • Test SMS sending with MSG91 provider
* • Test error handling when no provider configured
* • Test message logging with both providers
* • Test delivery status webhook updates

## End-to-End Testing

Complete User Journey:

1. User registers on platform

2. User navigates to provider settings

3. User views provider comparison

4. User selects MSG91 and enters credentials

5. System validates credentials and shows balance

6. User sends SMS to test number

7. SMS is delivered successfully

8. Webhook updates delivery status

9. User views message log with delivery confirmation

# 10. Troubleshooting

## Common Issues and Solutions

Issue: Credential Validation Fails

Possible causes:

* • Invalid API credentials (check for typos)
* • Network connectivity issues
* • Provider API is down
* • Rate limiting from provider

Solutions:

* • Verify credentials directly on provider's console
* • Check server's internet connectivity
* • Test provider API with curl or Postman
* • Implement exponential backoff for retries

Issue: Messages Not Sending

Possible causes:

* • No active provider configured for user
* • Insufficient balance in provider account
* • Invalid phone number format
* • Provider credentials expired or revoked

Solutions:

* • Check user has configured provider in database
* • Verify provider account balance directly
* • Validate phone number using E.164 format
* • Revalidate provider credentials

Issue: Webhook Not Receiving Updates

Possible causes:

* • Webhook URL not registered with provider
* • Webhook endpoint not publicly accessible
* • SSL certificate issues
* • Firewall blocking provider's IP addresses

Solutions:

* • Verify webhook URL is configured in provider console
* • Test webhook endpoint with curl from external server
* • Ensure SSL certificate is valid and not self-signed
* • Whitelist provider IP ranges in firewall

Issue: Decryption Errors

Possible causes:

* • ENCRYPTION\_KEY changed or not set
* • Database corruption
* • Wrong encryption algorithm or mode

Solutions:

* • Verify ENCRYPTION\_KEY environment variable is set correctly
* • Check database data integrity
* • If key was rotated, implement key migration process
* • Have users reconfigure their provider credentials

## Logging and Monitoring

Key Metrics to Monitor:

* • Message delivery success rate per provider
* • API response times
* • Credential validation success rate
* • Webhook delivery rate
* • Database query performance
* • Error rates by error type

Log Files to Check:

* • Application error logs
* • Provider API request/response logs
* • Database query logs
* • Webhook event logs
* • Authentication logs

## Support Resources

Twilio Resources:

* • Documentation: https://www.twilio.com/docs
* • Status page: https://status.twilio.com
* • Support: https://support.twilio.com

MSG91 Resources:

* • Documentation: https://msg91.com/help
* • API Reference: https://docs.msg91.com
* • Support: Contact via MSG91 dashboard

# Appendix: Quick Reference

## Environment Variables

DATABASE\_URL=postgresql://user:pass@localhost:5432/yourdb

ENCRYPTION\_KEY=your-32-byte-hex-key-here

JWT\_SECRET=your-jwt-secret

NODE\_ENV=production

## Generate Encryption Key Command

node -e "console.log(require('crypto').randomBytes(32).toString('hex'))"

## Required NPM Packages

* • express - Web framework
* • pg - PostgreSQL client
* • twilio - Twilio API SDK
* • axios - HTTP client for MSG91
* • jsonwebtoken - JWT authentication
* • bcrypt - Password hashing
* • dotenv - Environment variables

## API Endpoints Summary

|  |  |  |
| --- | --- | --- |
| Method | Endpoint | Description |
| POST | /api/messaging/providers | Save provider credentials |
| GET | /api/messaging/providers | Get configured providers |
| POST | /api/messaging/send-sms | Send SMS message |
| POST | /api/messaging/webhooks/twilio | Twilio webhook |
| POST | /api/messaging/webhooks/msg91 | MSG91 webhook |

End of Implementation Guide

For technical support or questions, please refer to the provider documentation or contact their support teams.