# FinBrain Expense Tracker - Technical Audit Summary

\*\*Audit Date:\*\* August 12, 2025 06:59:53 UTC

\*\*Git Commit:\*\* e0d5a18 (HEAD -> main) Saved your changes before starting work

\*\*Auditor:\*\* Read-only technical audit

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 1. Project Snapshot

\*\*Project Name:\*\* FinBrain - Multi-Platform Expense Tracker

\*\*Primary Language:\*\* Python 3.11

\*\*Framework:\*\* Flask 3.1.1 + SQLAlchemy 2.0.42 + APScheduler

\*\*Entrypoint:\*\* `main.py` → `app.py`

\*\*Start Command:\*\* `gunicorn --bind 0.0.0.0:5000 --reuse-port --reload main:app`

\*\*Runtime:\*\* Gunicorn 23.0.0 with sync workers, PostgreSQL 16

\*\*Replit Configuration:\*\* `.replit` configured for autoscale deployment

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 2. File & Folder Map

├── app.py # Flask app initialization & routes

├── main.py # Entry point (imports app)

├── models.py # SQLAlchemy database models

├── pyproject.toml # Python dependencies

├── uv.lock # Dependency lock file (917 lines)

├── .replit # Replit deployment config

├── .env.example # Environment variables template

├── replit.md # Project documentation

├── static/

│ └── finbrain\_logo.png # Application logo

├── templates/

│ └── dashboard.html # Web dashboard template

└── utils/ # Core business logic modules

├── \_\_init\_\_.py

├── categories.py # Expense categorization logic

├── db.py # Database operations

├── expense.py # Expense processing core

├── facebook\_handler.py # Facebook Messenger integration

├── rate\_limiter.py # Rate limiting functionality

├── report\_generator.py # Automated report generation

├── scheduler.py # APScheduler configuration

├── security.py # Security utilities (hashing)

└── whatsapp\_handler.py # WhatsApp/Twilio integration

**Critical Files:**

* \*\*Server Entry:\*\* `main.py` → `app.py`
* \*\*Webhook Handler:\*\* `app.py:webhook()` (unified endpoint)
* \*\*Workers:\*\* APScheduler background tasks
* \*\*Scheduler:\*\* `utils/scheduler.py`
* \*\*Config:\*\* `.replit`, `.env.example`, `pyproject.toml`

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 3. Configuration & Secrets

### Required Environment Variables:

* `DATABASE\_URL` - PostgreSQL connection string
* `SESSION\_SECRET` - Flask session security key
* `TWILIO\_ACCOUNT\_SID` - Twilio account identifier
* `TWILIO\_AUTH\_TOKEN` - Twilio authentication token
* `TWILIO\_WHATSAPP\_NUMBER` - WhatsApp business number
* `FACEBOOK\_PAGE\_ACCESS\_TOKEN` - Facebook Graph API token
* `FACEBOOK\_VERIFY\_TOKEN` - Webhook verification token

### Optional Variables:

* `DAILY\_MESSAGE\_LIMIT` (default: 50)
* `HOURLY\_MESSAGE\_LIMIT` (default: 10)
* `GOOGLE\_API\_KEY` - For future AI enhancements
* `GOOGLE\_SHEETS\_CREDS\_JSON` - For future integrations
* `CSV\_ENCRYPTION\_KEY` - For data backups
* `REPORT\_TIMEZONE` - Scheduler timezone
* `DAILY\_REPORT\_TIME`, `WEEKLY\_REPORT\_TIME` - Report scheduling

### Status:

✅ All required secrets are configured in environment

⚠️ Default fallback values used in code (security risk)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 4. Dependencies & Tooling

### Core Dependencies:

apscheduler>=3.11.0 # Background job scheduling

flask>=3.1.1 # Web framework

flask-sqlalchemy>=3.1.1 # ORM integration

gunicorn>=23.0.0 # WSGI server

psycopg2-binary>=2.9.10 # PostgreSQL adapter

sqlalchemy>=2.0.42 # Database ORM

twilio>=9.7.0 # WhatsApp integration

werkzeug>=3.1.3 # WSGI utilities

email-validator>=2.2.0 # Input validation

### Risk Assessment:

* ✅ All dependencies are current stable versions
* ✅ No deprecated packages detected
* ✅ Security-focused package selection
* ⚠️ No explicit version pinning (using >=)

### Process Management:

* \*\*WSGI Server:\*\* Gunicorn with reload enabled
* \*\*Scheduler:\*\* APScheduler BackgroundScheduler
* \*\*Task Queue:\*\* None (direct execution)
* \*\*HTTP Client:\*\* Python requests library

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 5. Runtime & Process Model

### Application Start:

1. `main.py` imports Flask app from `app.py`
2. `app.py` initializes database with `db.create\_all()`
3. APScheduler starts background tasks
4. Gunicorn binds to `0.0.0.0:5000`

### Concurrency Model:

* \*\*Sync Workers:\*\* Gunicorn default (single-threaded)
* \*\*Database:\*\* Connection pooling with pre-ping
* \*\*Scheduler:\*\* Single background thread
* \*\*Webhooks:\*\* Synchronous request handling

### Production Switches:

* `FLASK\_ENV` environment variable
* Debug mode enabled in development
* No explicit feature flags for dev/prod

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 6. HTTP Endpoints & Webhooks

### Routes Inventory:

| Method | Path | Purpose | Auth | Rate Limit |

|--------|------|---------|------|------------|

| GET | `/` | Dashboard view | None | None |

| GET | `/health` | Health check | None | None |

| GET | `/webhook` | Facebook verification | Token | None |

| POST | `/webhook` | Unified message handler | Platform-specific | Per-user |

### Webhook Details:

**`/webhook` Endpoint:**

* \*\*WhatsApp (Twilio):\*\* `application/x-www-form-urlencoded`
* \*\*Facebook Messenger:\*\* `application/json`
* \*\*Platform Detection:\*\* Content-Type header inspection
* \*\*Signature Verification:\*\* Facebook verify\_token check
* \*\*Response:\*\* "OK" (200) for WhatsApp, "EVENT\_RECEIVED" (200) for Facebook
* \*\*Idempotency:\*\* Message SID/timestamp-based unique IDs

**Facebook Graph API Integration:**

* \*\*API Version:\*\* v17.0
* \*\*Endpoint:\*\* `https://graph.facebook.com/v17.0/me/messages`
* \*\*Authentication:\*\* Page Access Token
* \*\*Rate Limits:\*\* Facebook platform limits (not enforced locally)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 7. Background Jobs, Queues, and Schedules

### APScheduler Configuration:

# Daily reports: 8 PM local time

CronTrigger(hour=20, minute=0)

# Weekly reports: Sunday 9 AM

CronTrigger(day\_of\_week=0, hour=9, minute=0)

### Jobs:

1. \*\*Daily Reports\*\* (`send\_daily\_reports`)

- Trigger: Daily at 20:00

- Function: Generate and send expense summaries

- Recipients: All active users

1. \*\*Weekly Reports\*\* (`send\_weekly\_reports`)

- Trigger: Sundays at 09:00

- Function: Generate and send weekly analytics

- Recipients: All active users

### Worker Management:

* \*\*Single Process:\*\* No leader election implemented
* \*\*Persistence:\*\* Jobs stored in memory (lost on restart)
* \*\*Error Handling:\*\* Exception logging, no retry mechanism

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 8. Database & Storage

### Database Type:

* \*\*Primary:\*\* PostgreSQL 16 (Replit managed)
* \*\*ORM:\*\* SQLAlchemy 2.0 with DeclarativeBase
* \*\*Connection:\*\* Pool recycling (300s), pre-ping enabled

### Schema Summary:

-- Core Tables

expenses # Detailed transaction logs

users # User profiles with hashed IDs

monthly\_summaries # Aggregated analytics

rate\_limits # Usage tracking per user/platform

### Table Details:

**`expenses`:**

* Primary key: `id` (auto-increment)
* User identification: `user\_id` (SHA-256 hash)
* Financial data: `amount` (Numeric 10,2), `currency`
* Categorization: `category`, `description`
* Temporal: `date`, `time`, `month` (YYYY-MM)
* Platform: `platform` (whatsapp/facebook)
* Deduplication: `unique\_id`

**`users`:**

* User identity: `user\_id\_hash` (unique, SHA-256)
* Analytics: `total\_expenses`, `expense\_count`
* Rate limiting: `daily\_message\_count`, `hourly\_message\_count`
* Timestamps: `created\_at`, `last\_interaction`

### Migrations:

* \*\*Strategy:\*\* `db.create\_all()` on startup
* \*\*Schema Evolution:\*\* No migration system implemented
* \*\*Data Safety:\*\* No backup/restore mechanism

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 9. Feature Flags & Safe Switches

### Rate Limiting Controls:

* `DAILY\_MESSAGE\_LIMIT`: Messages per user per day (default: 50)
* `HOURLY\_MESSAGE\_LIMIT`: Messages per user per hour (default: 10)

### Debug Controls:

* `FLASK\_ENV`: Development/production mode
* `logging.basicConfig(level=logging.DEBUG)`: Debug logging

### Platform Controls:

* No explicit enable/disable flags for WhatsApp or Facebook
* Platform detection via Content-Type headers

### Missing Flags:

* No outbound message toggle
* No maintenance mode
* No user whitelist/blacklist
* No category system disable

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 10. Observability & Ops

### Health Monitoring:

* \*\*Endpoint:\*\* `GET /health`
* \*\*Checks:\*\* Database connectivity test
* \*\*Response:\*\* JSON with service status and platform support

### Logging Strategy:

* \*\*Level:\*\* DEBUG in development
* \*\*Format:\*\* Python logging module defaults
* \*\*Destinations:\*\* Console output only
* \*\*Structure:\*\* Unstructured text logs

### Metrics:

* \*\*Built-in:\*\* None implemented
* \*\*Custom:\*\* User count, expense totals via dashboard
* \*\*APM:\*\* Not configured

### Admin Interface:

* \*\*Dashboard:\*\* Web UI at `/` showing:

- Total users count

- Monthly expense totals

- Transaction counts

- Recent expenses list

* \*\*Controls:\*\* Read-only statistics display

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 11. Security Posture

### Input Validation:

✅ \*\*Amount Parsing:\*\* Regex-based extraction with error handling

✅ \*\*User ID Hashing:\*\* SHA-256 for all identifiers

✅ \*\*Input Sanitization:\*\* Basic SQL injection prevention

⚠️ \*\*Message Size:\*\* No explicit length limits

⚠️ \*\*XSS Protection:\*\* Minimal template escaping

### Authentication:

✅ \*\*Facebook Verification:\*\* Webhook token validation

✅ \*\*Twilio Integration:\*\* Account SID/Auth Token

❌ \*\*User Authentication:\*\* No login system

❌ \*\*Admin Access:\*\* No authentication on dashboard

### Secret Management:

✅ \*\*Environment Variables:\*\* All sensitive data externalized

⚠️ \*\*Default Values:\*\* Fallback secrets in code (security risk)

✅ \*\*Session Security:\*\* Configurable SESSION\_SECRET

### Common Risks:

❌ \*\*Open Dashboard:\*\* No authentication on `/` endpoint

❌ \*\*Debug Mode:\*\* Enabled in production builds

⚠️ \*\*Rate Limiting:\*\* Basic implementation, no distributed tracking

✅ \*\*HTTPS:\*\* Enforced via ProxyFix middleware

### PII Handling:

✅ \*\*Data Minimization:\*\* User phone numbers/PSIDs hashed

✅ \*\*No Storage:\*\* Raw personal identifiers not persisted

⚠️ \*\*Message Content:\*\* Original messages stored in plaintext

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 12. Performance & Reliability Risks

### High-Risk Areas:

**Blocking I/O in Webhook Handler:**

* \*\*Issue:\*\* Synchronous external API calls in request thread
* \*\*Impact:\*\* WhatsApp/Facebook webhook timeouts
* \*\*Location:\*\* `utils/whatsapp\_handler.py:send\_whatsapp\_message()`
* \*\*Location:\*\* `utils/facebook\_handler.py:send\_facebook\_message()`

**Database Connection Bottlenecks:**

* \*\*Issue:\*\* Single connection pool, no horizontal scaling
* \*\*Impact:\*\* High load performance degradation
* \*\*Mitigation:\*\* Pool recycling configured

**Memory Scheduler Storage:**

* \*\*Issue:\*\* APScheduler jobs stored in memory
* \*\*Impact:\*\* Job loss on application restart
* \*\*Risk:\*\* Reports not sent during outages

### Timeout & Retry Policies:

❌ \*\*HTTP Timeouts:\*\* Not configured for external APIs

❌ \*\*Retry Logic:\*\* No automatic retry for failed sends

⚠️ \*\*Rate Limiting:\*\* Basic daily/hourly counters only

❌ \*\*Circuit Breaker:\*\* No protection against API failures

### Idempotency Coverage:

✅ \*\*Message Processing:\*\* Unique ID deduplication

❌ \*\*External API Calls:\*\* No idempotency keys

❌ \*\*Database Operations:\*\* No transaction replay protection

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 13. Notable Findings (Issues & Gaps)

### Critical Issues:

1. \*\*Security: Default Secrets in Production\*\*

- \*\*File:\*\* `app.py:20`

- \*\*Issue:\*\* `app.secret\_key = os.environ.get("SESSION\_SECRET", "default\_secret\_key\_for\_dev")`

- \*\*Risk:\*\* Session hijacking if SESSION\_SECRET not set

1. \*\*Reliability: Blocking Webhook Processing\*\*

- \*\*Files:\*\* `utils/whatsapp\_handler.py`, `utils/facebook\_handler.py`

- \*\*Issue:\*\* Synchronous HTTP calls to external APIs in webhook handlers

- \*\*Risk:\*\* Webhook timeouts, message delivery failures

1. \*\*Observability: No Error Monitoring\*\*

- \*\*Issue:\*\* Exceptions logged but not monitored

- \*\*Risk:\*\* Silent failures in production

### Configuration Issues:

1. \*\*Database: No Migration System\*\*

- \*\*Issue:\*\* Schema changes require manual intervention

- \*\*Risk:\*\* Data loss during updates

1. \*\*Scheduler: Memory Persistence\*\*

- \*\*Issue:\*\* Jobs lost on application restart

- \*\*Risk:\*\* Missed automated reports

### Missing Features:

1. \*\*Rate Limiting: No Distributed Tracking\*\*

- \*\*Issue:\*\* Per-instance limits only

- \*\*Risk:\*\* Limit bypass in scaled deployments

1. \*\*Admin: No Authentication\*\*

- \*\*Issue:\*\* Dashboard accessible without authentication

- \*\*Risk:\*\* Data exposure

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 14. Quick Wins (Now), Next, Later

### Now (0–1 day):

✅ \*\*Remove Default Secrets\*\*

* Update `app.py:20` to fail fast if SESSION\_SECRET missing
* Add validation for all required environment variables

✅ \*\*Add Request Timeouts\*\*

* Configure timeouts for Twilio and Facebook API calls
* Implement basic retry logic with exponential backoff

✅ \*\*Dashboard Authentication\*\*

* Add simple HTTP basic auth to dashboard endpoint
* Configure admin credentials via environment variables

### Next (2–7 days):

🔄 \*\*Async Webhook Processing\*\*

* Implement background task queue (Redis + RQ)
* Move external API calls out of request handlers
* Add webhook response time monitoring

🔄 \*\*Error Monitoring\*\*

* Integrate with error tracking service (Sentry)
* Add structured logging with correlation IDs
* Implement health check dependencies

🔄 \*\*Database Migrations\*\*

* Add Alembic for schema version control
* Create migration scripts for existing schema
* Implement backup/restore procedures

### Later (>7 days):

🔮 \*\*Horizontal Scaling\*\*

* Move to Redis-backed rate limiting
* Implement distributed scheduler (Celery)
* Add load balancer health checks

🔮 \*\*Advanced Security\*\*

* Implement webhook signature verification
* Add message encryption for sensitive data
* Security audit and penetration testing

🔮 \*\*Analytics Enhancement\*\*

* Add user behavior analytics
* Implement expense prediction models
* Create business intelligence dashboard

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 15. UAT Checklist (for this repo)

### Critical Path Testing:

* [ ] Health endpoint returns 200 on deploy URL
* [ ] WhatsApp webhook processes test message successfully
* [ ] Facebook webhook processes test message successfully
* [ ] Duplicate message IDs are properly ignored
* [ ] Rate limits block excessive messages correctly
* [ ] Background scheduler starts without errors
* [ ] Daily/weekly reports generate correctly
* [ ] Dashboard loads and displays current data

### Platform Integration:

* [ ] Twilio WhatsApp sandbox configuration verified
* [ ] Facebook Messenger webhook verification passes
* [ ] Database connections established successfully
* [ ] All required environment variables configured

### Error Handling:

* [ ] Invalid expense messages handled gracefully
* [ ] External API failures don't crash application
* [ ] Database connection errors recovered automatically
* [ ] Malformed webhook payloads rejected safely

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 16. Appendices

### A. Exact Start/Run Commands

# Development

python main.py

# Production (Replit)

gunicorn --bind 0.0.0.0:5000 --reuse-port --reload main:app

# Configured in .replit

gunicorn --bind 0.0.0.0:5000 main:app

### B. Environment Variable Matrix

| Variable | Required | Used In | Notes |

|----------|----------|---------|-------|

| DATABASE\_URL | ✅ | app.py | PostgreSQL connection |

| SESSION\_SECRET | ✅ | app.py | Flask session security |

| TWILIO\_ACCOUNT\_SID | ✅ | whatsapp\_handler.py | Twilio authentication |

| TWILIO\_AUTH\_TOKEN | ✅ | whatsapp\_handler.py | Twilio authentication |

| TWILIO\_WHATSAPP\_NUMBER | ✅ | whatsapp\_handler.py | WhatsApp business number |

| FACEBOOK\_PAGE\_ACCESS\_TOKEN | ✅ | facebook\_handler.py | Graph API access |

| FACEBOOK\_VERIFY\_TOKEN | ✅ | app.py | Webhook verification |

| DAILY\_MESSAGE\_LIMIT | ❌ | rate\_limiter.py | Default: 50 |

| HOURLY\_MESSAGE\_LIMIT | ❌ | rate\_limiter.py | Default: 10 |

### C. Dependency Risk Table

| Package | Version | Purpose | Risk Level | Notes |

|---------|---------|---------|------------|-------|

| flask | >=3.1.1 | Web framework | Low | Current stable |

| sqlalchemy | >=2.0.42 | Database ORM | Low | Current stable |

| gunicorn | >=23.0.0 | WSGI server | Low | Production-ready |

| psycopg2-binary | >=2.9.10 | PostgreSQL driver | Low | Stable binary |

| twilio | >=9.7.0 | WhatsApp integration | Medium | External dependency |

| apscheduler | >=3.11.0 | Background tasks | Medium | Memory persistence |

### D. Route Inventory

| Method | Path | Purpose | Auth | Rate Limit | Timeout |

|--------|------|---------|------|------------|---------|

| GET | `/` | Dashboard | None | None | Default |

| GET | `/health` | Health check | None | None | 5s |

| GET | `/webhook` | FB verification | Token | None | Default |

| POST | `/webhook` | Message processing | Platform | Per-user | 30s |

### E. File-specific Notes

**app.py:**

* Line 20: Default secret key (security risk)
* Line 98: Facebook token verification
* Lines 122-150: WhatsApp message handling
* Lines 151-180: Facebook message handling

**utils/expense.py:**

* Lines 10-28: Amount parsing with regex
* Lines 52-79: Core expense processing logic
* Error handling for malformed messages

**utils/categories.py:**

* Lines 7-84: Category definitions with keywords
* Lines 86-112: Smart categorization algorithm
* 10 predefined categories with emoji support

### F. Integration Touchpoints

**Twilio WhatsApp Business API:**

* \*\*Endpoint:\*\* `https://api.twilio.com/2010-04-01/Accounts/{sid}/Messages.json`
* \*\*Authentication:\*\* Basic Auth (SID:Token)
* \*\*Payload:\*\* Form data (From, To, Body)
* \*\*Headers:\*\* `application/x-www-form-urlencoded`
* \*\*Webhook:\*\* Form data with SmsMessageSid

**Facebook Graph API v17.0:**

* \*\*Endpoint:\*\* `https://graph.facebook.com/v17.0/me/messages`
* \*\*Authentication:\*\* Page Access Token
* \*\*Payload:\*\* JSON (recipient, message)
* \*\*Headers:\*\* `application/json`
* \*\*Webhook:\*\* JSON with entry/messaging structure

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**End of Audit - Document Complete**