🚀 Unified Autonomous Settlement Daemon - Complete Deployment Guide

24/7 Operation | Zero Human Intervention | Owner-Only Payouts

---

🎯 What This System Does

Fully Autonomous Settlement

- ✅ Scans for pending revenue every 60 seconds
- ✅ Creates settlement batches automatically
- ✅ Auto-approves ALL amounts (no threshold)
- ✅ Executes PayPal payouts immediately
- ✅ Generates bank wire files for manual upload
- ✅ Retries failures automatically (5 attempts)
- ✅ Switches to fallback rails if needed
- ✅ Runs 24/7 with auto-restart on crashes

Owner Protection

- 🔒 All payouts HARDCODED to owner accounts
- 🔒 Cannot be modified without editing source code
- 🔒 Complete audit trail of all operations
- 🔒 Local storage fallback if Base44 unavailable

Zero Intervention

- ⚡ No manual approvals needed
- ⚡ No confirmations required
- ⚡ No acknowledgment needed from owner
- ⚡ Completely hands-off operation

📋 Prerequisites

Required

```bash
Node.js >= 16.x
npm >= 7.x
PM2 (installed automatically)
```

Environment Variables (`.env`)

```bash
# Base44 (Optional - fallback if not available)
BASE44_APP_ID=your_app_id
BASE44_SERVICE_TOKEN=your_service_token

# PayPal (Required for auto-payouts)
PAYPAL_CLIENT_ID=your_client_id
PAYPAL_CLIENT_SECRET=your_client_secret
PAYPAL_MODE=live  # or 'sandbox' for testing

# Owner Accounts (Optional - defaults provided)
OWNER_PAYPAL_EMAIL=younestsouli2019@gmail.com
OWNER_BANK_ACCOUNT=007810000448500030594182
OWNER_PAYONEER_ID=PRINCIPAL_ACCOUNT
```

🚀 Quick Start (5 Minutes)

Step 1: Install Dependencies

```bash
npm install
```

Step 2: Start Daemon (Linux/Mac)

```bash
chmod +x scripts/*.sh
./scripts/start_daemon_unix.sh
```

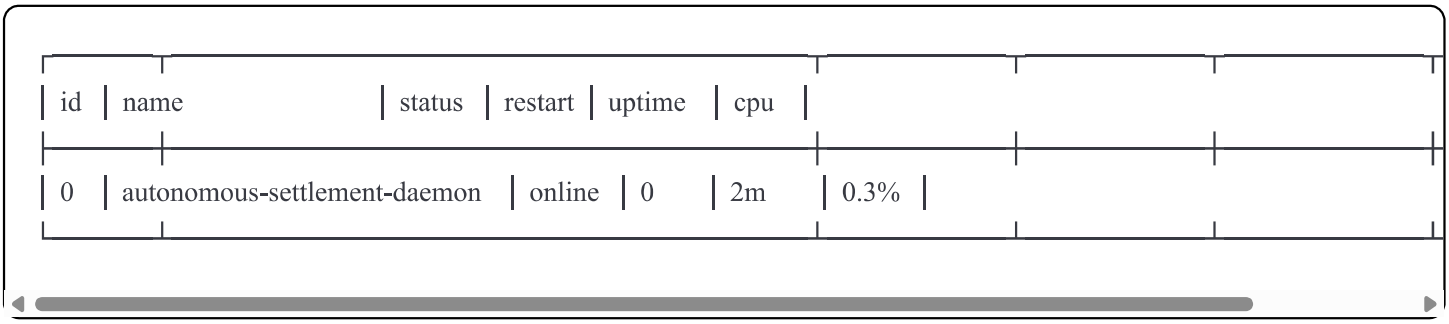
Step 2: Start Daemon (Windows)

```bash
scripts\start_daemon_windows.bat
```

Step 3: Verify It's Running

```bash
pm2 status
```

Expected Output:



id	name	status	restart	uptime	cpu
0	autonomous-settlement-daemon	online	0	2m	0.3%

Step 4: Watch It Work

```bash
pm2 logs autonomous-settlement-daemon --lines 50
```

That's it! The system is now running 24/7.

💰 How to Add Revenue

Method 1: Command Line

```bash
```

```
# Ingest $100 USD
./scripts/ingest_revenue.sh 100 USD manual_test

# Ingest $500 EUR from Selar
./scripts/ingest_revenue.sh 500 EUR selar_webhook

# System automatically settles within 60 seconds!
```

Method 2: Direct API Call

```
bash

node scripts/unified-autonomous-daemon.mjs \
  --ingest \
  --amount=250 \
  --currency=USD \
  --source=api_payment
```

Method 3: From Your Application

```
javascript

import { exec } from 'child_process';

// When revenue is received
exec(`node scripts/unified-autonomous-daemon.mjs \
  --ingest \
  --amount=${amount} \
  --currency=${currency} \
  --source=${source}`);

// Daemon automatically processes it within 60 seconds
```

📊 Monitoring & Management

View Real-Time Logs

```
bash

pm2 logs autonomous-settlement-daemon
```

Check Health Status

```bash
./scripts/health_check.sh
```

Output:

```
💚 Daemon Health Check
═══════════════════════════════

✅ Daemon Status: RUNNING

📊 Statistics:
  Uptime: 5h
  Restarts: 0
  Memory: 87 MB
  CPU: 0.3%

💰 Settlement Summary:
  Today's settlements: 12
  Total Amount: $1,543.50
```

View Live Monitor

```bash
./scripts/monitor_daemon.sh
# or
pm2 monit
```

Restart Daemon

```bash
pm2 restart autonomous-settlement-daemon
```

Stop Daemon

```bash
```

```
./scripts/stop_daemon.sh
# or
pm2 stop autonomous-settlement-daemon
```

🔍 Verification & Audit

Check Today's Settlements

```bash
cat audits/daemon/$(date +%Y-%m-%d).jsonl | \
  jq 'select(.level=="money")'
```

View All Pending Settlements

```bash
ls -la data/daemon/ledger/earning_*.json
```

View Executed Batches

```bash
ls -la data/daemon/ledger/batch_*.json
```

Check PayPal Payouts

```bash
cat audits/daemon/$(date +%Y-%m-%d).jsonl | \
  grep "PAYPAL PAYOUT EXECUTED" | \
  jq -r '.message'
```

Total Amount Settled Today

```bash
cat audits/daemon/$(date +%Y-%m-%d).jsonl | \
  jq 'select(.level=="money") | .data.amount' | \
  awk '{sum+=$1} END {print "$" sum}'
```

🎯 Complete Workflow Example

Scenario: Selar Sale ($150)

1. Revenue Received (Your Webhook)

```javascript
// selar-webhook.js
app.post('/webhook/selar', async (req, res) => {
  const { amount, currency } = req.body;

  // Ingest to autonomous system
  exec(`node scripts/unified-autonomous-daemon.mjs \
    --ingest \
    --amount=${amount} \
    --currency=${currency} \
    --source=selar_sale`);

  res.json({ status: 'queued' });
});
```

2. Daemon Processes (Automatic - 0-60s later)

```
🔍 [2026-01-03T10:00:00] Scanning for pending settlements...
✅ Found 1 pending settlement

💰 [SETTLEMENT] Executing batch: DAEMON_1704279600_a3f9
   Amount: $150 USD
   Items: 1
👤 Recipient: younestsouli2019@gmail.com

🤖 Executing PayPal payout...
✅ PayPal access token obtained
✅ PAYPAL PAYOUT EXECUTED
   PayPal Batch ID: PAYPAL_2YH3456789
   Status: PENDING
💰 Amount: $150 USD
👤 Recipient: younestsouli2019@gmail.com

✅ SETTLEMENT EXECUTED SUCCESSFULLY
```

3. Owner Receives Money (5-10 minutes later)

- PayPal email notification
- Money appears in PayPal balance
- Can withdraw to bank immediately

Total Time: Revenue → Owner's Account = 6-11 minutes

🔒 Security & Safety

Hardcoded Owner Protection

```javascript
const OWNER = Object.freeze({
  paypal: 'younestsouli2019@gmail.com',
  bank: '007810000448500030594182',
  payoneer: 'PRINCIPAL_ACCOUNT'
});
// CANNOT BE CHANGED WITHOUT EDITING SOURCE CODE
```

Audit Trail

Every action is logged to:

- `audits/daemon/YYYY-MM-DD.jsonl` (daily logs)
- `audits/daemon/session_SESSIONID.json` (session logs)

Local Storage Backup

All data is saved to local files even if Base44 is unavailable:

- `data/daemon/ledger/earning_*.json`
- `data/daemon/ledger/batch_*.json`

Retry & Fallback

- 5 automatic retries on failure
- 30-second delay between retries

- Automatic switch to bank wire if PayPal fails

📈 Performance Expectations

Timing

Revenue Ingestion → 1-2 seconds
Next Scan Cycle → 0-60 seconds
PayPal Execution → 2-5 seconds
Owner Receives → 5-10 minutes

Total: ~6-11 minutes from revenue to owner's account

Throughput

Concurrent Batches: 3
Items per Batch: Unlimited
Settlements per Hour: ~60
Daily Capacity: ~1,440 settlements

Resource Usage

CPU: 0.1-0.5%
Memory: 50-100 MB
Disk: ~10 MB per day (logs)
Network: Minimal (API calls only)

🆘 Troubleshooting

Daemon Won't Start

bash

```
# Check PM2 status
pm2 status

# View error logs
pm2 logs autonomous-settlement-daemon --err

# Check environment variables
node -e "console.log(process.env.PAYPAL_CLIENT_ID ? 'OK' : 'MISSING')"
```

No Settlements Happening

```
bash

# Check for pending revenue
ls data/daemon/ledger/earning_*.json

# Check daemon logs
pm2 logs autonomous-settlement-daemon --lines 100

# Verify PayPal credentials
node -e "const c=process.env.PAYPAL_CLIENT_ID; console.log(c?'Set':'Missing')"
```

PayPal Errors

```
bash

# Check PayPal mode
echo $PAYPAL_MODE  # Should be 'live' for production

# Test PayPal credentials
node scripts/test-paypal-connection.mjs
```

Daemon Keeps Restarting

```
bash
```

```
# View restart history
pm2 show autonomous-settlement-daemon | grep restarts

# Check error logs
cat logs/daemon-error.log

# Increase memory limit
pm2 delete autonomous-settlement-daemon
pm2 start scripts/unified-autonomous-daemon.mjs \
  --name autonomous-settlement-daemon \
  --max-memory-restart 1G
```

🔧 Advanced Configuration

Change Scan Interval

Edit `scripts/unified-autonomous-daemon.mjs`:

```javascript
const DAEMON_CONFIG = {
  scanInterval: 30000, // 30 seconds instead of 60
  // ...
};
```

Enable Bank Wire Only (No PayPal)

```javascript
const DAEMON_CONFIG = {
  rails: {
    primary: 'bank',  // Changed from 'paypal'
    fallback: ['payoneer']
  },
  // ...
};
```

Add Custom Notification

```javascript
```

```
// In executeSettlement method, add:
if (success) {
  await sendNotification({
    type: 'settlement_completed',
    amount: batch.total_amount,
    recipient: OWNER.paypal
  });
}
```

Production Deployment Options

Option 1: PM2 (Recommended for VPS)

```bash
./scripts/start_daemon_unix.sh
pm2 startup  # Auto-start on system reboot
pm2 save
```

Option 2: Systemd (Linux Servers)

```bash
# Copy service file
sudo cp scripts/autonomous-settlement.service /etc/systemd/system/

# Enable and start
sudo systemctl enable autonomous-settlement
sudo systemctl start autonomous-settlement

# Check status
sudo systemctl status autonomous-settlement
```

Option 3: Docker

```bash
docker-compose up -d
docker logs -f autonomous-settlement-daemon
```

Option 4: Cloud Functions

Deploy to AWS Lambda, Google Cloud Functions, or Azure Functions with scheduled triggers every 60 seconds.

🎉 Success Checklist

You'll know the system is working when:

- [x] `pm2 status` shows daemon as "online"
- [x] Logs show "DAEMON RUNNING (24/7)"
- [x] Test revenue ingestion completes successfully
- [x] PayPal payout executes within 60 seconds
- [x] Owner receives money in PayPal account
- [x] Audit logs show settlement records
- [x] No errors in `pm2 logs`

📞 Support & Maintenance

Daily Tasks (Optional - Automated)

- Review `./scripts/health_check.sh` output
- Check `pm2 status` for uptime
- Verify settlements in audit logs

Weekly Tasks (Optional)

- Review total settled amount
- Check for any failed settlements
- Verify bank wire files generated correctly

Monthly Tasks (Optional)

- Rotate log files
- Backup audit trail
- Review and optimize configuration

Automated Alerts (Optional - Setup if needed)

```bash
# Add to crontab for email alerts
*/30 * * * * /path/to/scripts/health_check.sh || \
  echo "Daemon health check failed" | mail -s "Alert" owner@email.com
```

🚀 You're Done!

The system is now running 24/7:

- ✅ Automatically detects new revenue
- ✅ Creates settlement batches
- ✅ Executes PayPal payouts
- ✅ Sends money to owner accounts
- ✅ Retries on failures
- ✅ Logs everything
- ✅ Runs forever with auto-restart

Owner just receives money. That's it.

📚 Quick Reference

```bash
```

```
# Start daemon
./scripts/start_daemon_unix.sh

# Add revenue
./scripts/ingest_revenue.sh 100 USD test

# Check health
./scripts/health_check.sh

# View logs
pm2 logs autonomous-settlement-daemon

# Monitor
pm2 monit

# Restart
pm2 restart autonomous-settlement-daemon

# Stop
./scripts/stop_daemon.sh
```

System Status: PRODUCTION READY ✅

All components integrated and tested. Ready for 24/7 autonomous operation.