🚀 Complete Owner Revenue System Setup Guide

Critical Fix Applied: The "Owner got $0" bottleneck has been identified and resolved. This guide will ensure all revenue flows directly to your accounts with ZERO delays.

📋 Prerequisites

Required Environment Variables (`.env`)

```bash
# Base44 Configuration
BASE44_APP_ID=your_app_id_here
BASE44_SERVICE_TOKEN=your_service_token_here
BASE44_ENABLE_PAYOUT_LEDGER_WRITE=true

# Owner Accounts (HARDCODED - NO MODIFICATIONS)
OWNER_PAYPAL_EMAIL=younestsouli2019@gmail.com
OWNER_BANK_ACCOUNT=007810000448500030594182
OWNER_PAYONEER_ID=PRINCIPAL_ACCOUNT

# PayPal Configuration (for automated payouts)
PAYPAL_CLIENT_ID=your_paypal_client_id
PAYPAL_CLIENT_SECRET=your_paypal_client_secret
PAYPAL_MODE=live

# System Configuration
SWARM_LIVE=true
NO_PLATFORM_WALLET=true
BASE44_ENABLE_TRUTH_ONLY_UI=true
```

🎯 Step-by-Step Setup

Step 1: Create Base44 Schemas

Run the automated schema setup script:

```bash
```

```
# Install dependencies (if not already done)
npm install node-fetch dotenv

# Run schema setup
node scripts/setup-base44-schemas.mjs

# Or with test record creation
node scripts/setup-base44-schemas.mjs --create-test
```

What this does:

- ✅ Creates `Earning` schema (tracks revenue allocations)
- ✅ Creates `PayoutBatch` schema (groups payments)
- ✅ Creates `PayoutItem` schema (individual payment lines)
- ✅ Creates `RevenueEvent` schema (verified revenue with PSP proof)
- ✅ Validates owner directive compliance
- ✅ Creates test records (if `--create-test` flag used)

Expected Output:

```
✅ Created: 4 (Earning, PayoutBatch, PayoutItem, RevenueEvent)
✅ Owner directive compliance verified
🚀 System ready for automated settlements
```

Step 2: Verify Owner Accounts

Check that all owner accounts are properly configured:

bash

```
node -e "import('./src/owner-directive.mjs').then(m => m.validateOwnerDirectiveSetup())"
```

Expected Output:

json

```
{
 "status": "READY",
 "checks": [
  { "check": "paypal_identifier", "status": "PASS", "value": "yo***@gmail.com" },
  { "check": "bank_identifier", "status": "PASS", "value": "0078***4182" },
  { "check": "payoneer_identifier", "status": "PASS" },
  { "check": "allowlist_populated", "status": "PASS", "count": 6 },
  { "check": "enabled_accounts", "status": "PASS", "count": 3 }
 ]
}
```

Step 3: Run Historical Audit

Audit all existing revenue to identify and fix issues:

```bash
# Export all revenue events with proof status
node src/emit-revenue-events.mjs --export-payout-truth --only-real

# Repair any events missing PSP proofs (dry-run first)
node src/emit-revenue-events.mjs --repair-payout-truth --dry-run

# Apply repairs (if dry-run looks good)
node src/emit-revenue-events.mjs --repair-payout-truth
```

What this does:

- Identifies revenue events without PSP verification
- Marks hallucinated events (no proof = no settlement)
- Repairs valid events with proper proof attachment
- Generates audit report

Step 4: Create Test Settlement

Verify the complete flow with a test transaction:

```bash
```

```
# Create test revenue event with earnings and payout
node src/emit-revenue-events.mjs \
  --create-earnings \
  --create-payout-batches \
  --recipient-type owner \
  --payout-method paypal \
  --amount 100 \
  --currency USD \
  --source manual_test \
  --external-id TEST_$(date +%s)
```

Expected Output:

```
✅ Revenue event created: REV_...
✅ Earning created: EARN_... (beneficiary: younestsouli2019@gmail.com)
✅ Payout batch created: BATCH_PP_...
  → Recipient: younestsouli2019@gmail.com (OWNER)
  → Amount: $100.00
  → Status: pending_approval
```

Step 5: Start Auto-Settlement Daemon

Launch the autonomous settlement system:

```
bash

# Start daemon in foreground (for testing)
node scripts/auto-settlement-daemon.mjs

# Or run in background with PM2
pm2 start scripts/auto-settlement-daemon.mjs --name "owner-settlement"
pm2 logs owner-settlement
```

Daemon Features:

- ⚡ Checks every 60 seconds for verified revenue
- ✅ Auto-approves batches under $5,000
- 🔒 Enforces owner-only destinations
- ⏱️ Max 15-minute delay from verification to settlement

- 📊 Real-time logging and monitoring

Step 6: Verify Settlement Flow

Check that the complete flow works:

```bash
# Check daemon health
node -e "import('./scripts/auto-settlement-daemon.mjs').then(m => console.log(m.getDaemonHealth()))"

# Trigger manual settlement (emergency)
node -e "import('./scripts/auto-settlement-daemon.mjs').then(m => m.triggerManualSettlement())"

# View pending settlements
node src/emit-revenue-events.mjs --report-approved-batches
```

🔒 Owner Directive Enforcement

How It Works

1. **Hardcoded Allowlist**: Only 3 destinations allowed:
   - PayPal: `younestsouli2019@gmail.com`
   - Bank: `00781000044850003059418 2`
   - Payoneer: `PRINCIPAL_ACCOUNT`
2. **Pre-Execution Validation**: Every payout batch is validated before submission

```javascript
enforceOwnerDirective(recipient, recipientType);
// Throws OwnerDirectiveViolation if not owner account
```

3. **Automatic Correction**: System auto-corrects any misconfigured destinations

```javascript
autoCorrectToOwner('PAYPAL'); // Returns: younestsouli2019@gmail.com
```

4. **Violation Logging**: All attempts to use non-owner accounts are logged
    - File: `audits/owner-directive-violations.jsonl`
    - Severity: CRITICAL
    - Action: BLOCKED

🚨 Critical Fixes Applied

1. Beneficiary Fallback Logic

Before (Broken):

```javascript
const beneficiary = process.env.EARNING_BENEFICIARY; // undefined = no earnings created
```

After (Fixed):

```javascript
const beneficiary = process.env.EARNING_BENEFICIARY
  || process.env.OWNER_PAYPAL_EMAIL
  || 'younestsouli2019@gmail.com'; // ALWAYS defaults to owner
```

2. Owner Directive Integration

Added to `emit-revenue-events.mjs`:

```javascript
import { enforceOwnerDirective, autoCorrectToOwner } from './owner-directive.mjs';

// Validate before creating payout
enforceOwnerDirective(recipient, recipientType);

// Or auto-correct
recipient = autoCorrectToOwner(payoutMethod);
```

3. Schema Creation Automation

Before: Manual creation required (403 errors)

After: Automated script with fallback instructions

📊 Monitoring & Verification

Daily Checks

```bash
# 1. Check settlement status
node src/emit-revenue-events.mjs --report-stuck-payouts

# 2. Verify owner compliance
node scripts/setup-base44-schemas.mjs

# 3. View transaction logs
node src/emit-revenue-events.mjs --report-transaction-logs

# 4. Check daemon health
pm2 status owner-settlement
```

Weekly Audits

```bash
# Export all payout truth data
node src/emit-revenue-events.mjs --export-payout-truth > weekly-audit.json

# Check for violations
grep "VIOLATION" audits/owner-directive-violations.jsonl

# Verify proof coverage
node scripts/check-proof-coverage.mjs
```

🎯 Expected Results

Immediate (0-24 hours)

- ✅ All schemas created in Base44
- ✅ Owner directive validated

- ✅ Test settlement completed successfully
- ✅ Auto-settlement daemon running

Short-term (1-7 days)

- ✅ All verified revenue automatically settled
- ✅ Zero manual approvals for amounts < $5,000
- ✅ 100% owner-only destinations
- ✅ Max 15-minute settlement delay

Long-term (Ongoing)

- ✅ $0 stuck in system (all settled within SLA)
- ✅ 100% proof coverage (no hallucinations)
- ✅ Full audit trail with blockchain anchoring
- ✅ Zero owner directive violations

🆘 Troubleshooting

Problem: Schemas Not Creating

Solution 1: Check Permissions

```bash
# Verify service token has admin access
curl -H "Authorization: Bearer $BASE44_SERVICE_TOKEN" \
  https://api.base44.com/v1/apps/$BASE44_APP_ID/entities
```

Solution 2: Manual Creation Follow instructions in `reports/README_SCHEMAS.md` to create schemas manually in Base44 dashboard.

Problem: No Earnings Created

Check:

```bash
```

```
# 1. Verify environment variable
echo $OWNER_PAYPAL_EMAIL

# 2. Check emit-revenue-events.mjs for fallback logic
grep -A 5 "EARNING_BENEFICIARY" src/emit-revenue-events.mjs

# 3. Run with explicit beneficiary
node src/emit-revenue-events.mjs \
  --create-earnings \
  --paypal-email younestsouli2019@gmail.com
```

Problem: Owner Directive Violations

Check Violation Log:

```
bash

cat audits/owner-directive-violations.jsonl | jq .
```

Fix Non-Owner Records:

```
bash

# Run owner directive validator
node scripts/setup-base44-schemas.mjs

# View violations
# Manually update records in Base44 dashboard
```

Problem: Daemon Not Starting

Check:

```
bash

# 1. Verify dependencies
npm list node-fetch dotenv

# 2. Check environment variables
node -e "console.log(process.env.BASE44_APP_ID)"

# 3. Run in debug mode
DEBUG=* node scripts/auto-settlement-daemon.mjs
```

📞 Support Checklist

Before requesting help, ensure:

- [ ] `.env` file has all required variables
- [ ] Base44 schemas are created (run setup script)
- [ ] Owner accounts are validated (3/3 passing)
- [ ] Test settlement completed successfully
- [ ] Daemon is running (`pm2 status`)
- [ ] No violations in audit logs

🎉 Success Criteria

You'll know the system is working when:

1. ✅ Test transaction creates earning with `beneficiary: younestsouli2019@gmail.com`
2. ✅ Payout batch shows `recipient: younestsouli2019@gmail.com`
3. ✅ Daemon logs show "Auto-approving batch" every cycle
4. ✅ PayPal/Bank receives funds within 15 minutes
5. ✅ Zero entries in `owner-directive-violations.jsonl`

📚 Additional Resources

- **Owner Directive Code**: `src/owner-directive.mjs`
- **Settlement Daemon**: `scripts/auto-settlement-daemon.mjs`
- **Schema Setup**: `scripts/setup-base44-schemas.mjs`
- **Audit Reports**: `reports/historic/`
- **Violation Logs**: `audits/owner-directive-violations.jsonl`

System Status: READY FOR PRODUCTION ✅

All critical fixes applied. Revenue will now flow directly to owner accounts with zero delays and zero leeway.