Automatic Payout System Documentation

Overview

The automatic payout system enables seamless, scheduled payouts to creators based on their available balance and eligibility. This system implements OnlyFans-style automatic payouts with comprehensive eligibility checks, admin controls, and robust error handling.

Table of Contents

How Automatic Payouts Work

Payment Flow

1. **Payment Creation**: When a user books a call, a payment is created with status `PENDING`
2. **Payment Success**: After successful payment, status changes to `SUCCEEDED` and `payoutStatus` becomes `HELD`
3. **Holding Period**: Payments are held for N days (default: 7 days) configured in `PlatformSettings.holdingPeriodDays`
4. **Ready for Payout**: After holding period passes, `payoutStatus` changes to `READY`
5. **Automatic Payout**: Scheduler checks creators with automatic payouts enabled and processes eligible payouts
6. **Transfer to Creator**: Funds are transferred to creator's Stripe Connect account
7. **Payout Complete**: Payment `payoutStatus` becomes `PAID`

Automatic Payout Scheduler

The system includes a cron endpoint (`/api/cron/process-payouts`) that:

1. Runs periodically (recommended: daily at 2 AM UTC)
2. Finds all creators with `PayoutSchedule.mode = AUTOMATIC` and `isActive = true`
3. Checks if `nextPayoutDate` is null or has passed
4. Runs eligibility checks for each creator
5. Creates Stripe transfers for eligible creators
6. Updates `nextPayoutDate` based on frequency (DAILY, WEEKLY, MONTHLY)
7. Logs all operations to `TransactionLog`

Creator Onboarding

When a creator completes Stripe Connect onboarding:

1. `PayoutSchedule` is automatically created with default settings:
   - `mode: AUTOMATIC`
   - `frequency: WEEKLY`
   - `isActive: true`
   - `nextPayoutDate: null` (will be set after first payout)

Eligibility Requirements

For a creator to be eligible for automatic payouts, ALL of the following must be true:

1. Creator Account

- ✅ Creator exists in database
- ✅ Creator has valid `stripeAccountId`
- ✅ Creator has `isStripeOnboarded = true`

2. Payout Not Blocked

- ✅ `Creator.payoutBlocked = false`
- If blocked, payouts cannot proceed (admin control)

3. Stripe Account Status

- ✅ `charges_enabled = true` (account can receive payments)
- ✅ `payouts_enabled = true` (account can receive payouts)

4. KYC Complete

- ✅ `requirements.currently_due` is empty (no pending KYC requirements)
- ✅ `requirements.past_due` is empty (no overdue requirements)

5. Bank Account Connected

- ✅ `external_accounts.data.length > 0` (at least one bank account connected)

6. Minimum Balance

- ✅ Available balance >= `PlatformSettings.minimumPayoutAmount` (default: 10 EUR)

7. Holding Period

- ✅ All payments have passed the holding period
- Payments created within last N days (configured in settings) must have passed holding period

If any requirement fails, the payout is skipped and the reason is logged.

Payout Schedule Configuration

PayoutSchedule Model

```
model PayoutSchedule {
  id              String          @id @default(cuid())
  creatorId       String          @unique
  mode            PayoutMode      @default(AUTOMATIC)  // AUTOMATIC | MANUAL
  frequency       PayoutFrequency @default(WEEKLY)     // DAILY | WEEKLY | MONTHLY
  nextPayoutDate  DateTime?
  isActive        Boolean         @default(true)
  createdAt       DateTime        @default(now())
  updatedAt       DateTime        @updatedAt
}
```

Payout Modes

AUTOMATIC

- System automatically processes payouts based on frequency
- Checks eligibility on each scheduled run
- Transfers funds without manual intervention

MANUAL

- Creator must manually request payout via API
- Still subject to eligibility checks
- Gives creator control over payout timing

Payout Frequencies

DAILY

- Payout processed every day (if eligible)
- `nextPayoutDate` = current date + 1 day

WEEKLY

- Payout processed every 7 days
- `nextPayoutDate` = current date + 7 days

MONTHLY

- Payout processed once per month (same day)
- `nextPayoutDate` = current date + 1 month

Setting Up Cron Jobs

Vercel Cron (Recommended for Vercel Deployment)

The project includes `vercel.json` with cron configuration:

```
{
  "crons": [{
    "path": "/api/cron/process-payouts",
    "schedule": "0 2 * * *"
  }]
}
```

This runs daily at 2:00 AM UTC.

Environment Variables

Set the following environment variable for security:

```
CRON_SECRET=your_secure_random_secret_here
```

The cron endpoint verifies this secret before processing payouts.

Alternative: External Cron Service

If not using Vercel, you can use services like:
- **Cron-job.org**
- **EasyCron**
- **AWS CloudWatch Events**
- **GitHub Actions** (scheduled workflows)

Configure them to call:

```
GET https://your-domain.com/api/cron/process-payouts?secret=YOUR_CRON_SECRET
```

Or use header:

```
GET https://your-domain.com/api/cron/process-payouts
Headers: x-cron-secret: YOUR_CRON_SECRET
```

Manual Trigger (Development/Testing)

```
curl -X GET "http://localhost:3000/api/cron/process-payouts?secret=YOUR_CRON_SECRET"
```

Admin Controls

Block/Unblock Payouts

Endpoint: `POST /api/admin/creators/[id]/block-payout`

Block payouts for a specific creator:

```
{
  "blocked": true,
  "reason": "Suspicious activity detected"
}
```

Unblock payouts:

```
{
  "blocked": false
}
```

Manage Payout Settings

Endpoint: `PUT /api/admin/creators/[id]/payout-settings`

Update creator's payout schedule:

```
{
  "mode": "MANUAL",
  "frequency": "MONTHLY",
  "isActive": false
}
```

View Payout Dashboard

Endpoint: `GET /api/admin/payouts/dashboard`

Returns comprehensive overview:
- Pending payouts count
- Failed payouts (last 30 days)
- Creators with blocked payouts
- Creators with eligibility issues
- Next scheduled payout
- Total payout volume
- Payment status summary

Manual Payout with Admin Override

Endpoint: `POST /api/payouts/request`

Admins can override eligibility checks:

```
{
  "creatorId": "creator_id_here",
  "adminOverride": true
}
```

⚠️ **Warning**: Admin override skips all eligibility checks except Stripe account existence. Use with caution.

API Endpoints Reference

Creator Endpoints

Request Manual Payout

```
POST /api/payouts/request
Auth: Creator or Admin

Body (optional for admin):
{
  "creatorId": "string",      // Admin only
  "adminOverride": boolean    // Admin only
}

Response:
{
  "success": true,
  "message": "Paiement de 100.00 EUR transféré avec succès",
  "payout": {
    "id": "payout_id",
    "amount": 100.00,
    "currency": "EUR",
    "stripeTransferId": "tr_xxx",
    "paymentCount": 5
  }
}
```

Admin Endpoints

Get Payout Dashboard

```
GET /api/admin/payouts/dashboard
Auth: Admin

Response:
{
  "success": true,
  "timestamp": "2025-12-25T12:00:00Z",
  "summary": {
    "pendingPayouts": 3,
    "failedPayouts": 1,
    "blockedCreators": 2,
    "creatorsWithIssues": 5,
    "successfulPayouts": 42,
    "totalPayoutVolume": "5420.00",
    "currency": "EUR",
    ...
  },
  ...
}
```

Get Creator Payout Settings

```
GET /api/admin/creators/[id]/payout-settings
Auth: Admin

Response:
{
  "success": true,
  "creator": { ... },
  "payoutSchedule": { ... },
  "eligibility": {
    "eligible": true,
    "availableBalance": 150.00,
    "details": { ... }
  },
  "availableBalance": 150.00,
  "payoutHistory": [ ... ]
}
```

Update Creator Payout Settings

```
PUT /api/admin/creators/[id]/payout-settings
Auth: Admin

Body:
{
  "mode": "AUTOMATIC",
  "frequency": "WEEKLY",
  "isActive": true
}

Response:
{
  "success": true,
  "message": "Paramètres de paiement mis à jour avec succès",
  "payoutSchedule": { ... }
}
```

Block/Unblock Payouts

```
POST /api/admin/creators/[id]/block-payout
Auth: Admin

Body:
{
  "blocked": true,
  "reason": "Reason for blocking"
}

Response:
{
  "success": true,
  "message": "Paiements bloqués pour Creator Name",
  "creator": {
    "id": "creator_id",
    "payoutBlocked": true,
    "payoutBlockedReason": "Reason for blocking"
  }
}
```

Test Eligibility (Development Only)

```
POST /api/admin/payouts/test-eligibility
Auth: Admin
Environment: Development only

Body:
{
  "creatorId": "string"
}

Response:
{
  "success": true,
  "creator": { ... },
  "eligibility": { ... },
  "stripeAccountDetails": { ... },
  "stripeBalance": { ... },
  "paymentsSummary": { ... }
}
```

Cron Endpoints

Process Automatic Payouts

```
GET /api/cron/process-payouts
Auth: CRON_SECRET (header or query param)

Query Params:
- secret: CRON_SECRET value

Headers (alternative):
- x-cron-secret: CRON_SECRET value

Response:
{
  "success": true,
  "message": "Automatic payout processing complete",
  "summary": {
    "processed": 10,
    "succeeded": 8,
    "failed": 1,
    "skipped": 1,
    "duration": 5432,
    "timestamp": "2025-12-25T02:00:00Z",
    "errors": []
  }
}
```

Troubleshooting

Payout Not Processing

Symptom: Creator's payout is not being processed automatically

Possible Causes:

1. **PayoutSchedule not active**
   - Check: `PayoutSchedule.isActive = true`
   - Solution: Update via admin endpoint

2. **Next payout date in future**
   - Check: `PayoutSchedule.nextPayoutDate`
   - Solution: Wait for scheduled date or trigger manually

3. **Eligibility requirements not met**
   - Check: Use `/api/admin/payouts/test-eligibility` endpoint
   - Solution: Address specific requirement failures

4. **Payout blocked by admin**
   - Check: `Creator.payoutBlocked = true`
   - Solution: Unblock via admin endpoint

5. **Cron job not running**
   - Check: Vercel cron logs or external cron service
   - Solution: Verify CRON_SECRET and cron configuration

Failed Payouts

Symptom: Payout status is FAILED

Possible Causes:

1. **Stripe API error**
   - Check: `Payout.failureReason` field
   - Check: Stripe Dashboard for account issues
   - Solution: Address Stripe-specific issues

2. **Insufficient balance**
   - Check: Available balance in Stripe Connect account
   - Solution: Ensure platform has sufficient funds

3. **Bank account issues**
   - Check: External accounts in Stripe Connect account
   - Solution: Update bank account details

KYC Requirements Pending

Symptom: Creator cannot receive payouts due to KYC

Solution:
1. Check Stripe Connect account: `requirements.currently_due`
2. Send creator to complete Stripe onboarding:
- Use `/api/stripe/connect-onboard` endpoint
- Creator completes requirements in Stripe-hosted UI
3. Webhook `account.updated` will auto-unblock when complete

Balance Caching Issues

Symptom: Balance not updating after payout

Solution:
- Balance is cached for 30 seconds to prevent rate limiting
- Cache is automatically cleared after successful payout
- Manual cache clear available in code: `clearBalanceCache(creatorId)`

Security Considerations

Cron Endpoint Protection

- **CRON_SECRET**: Always use a strong, random secret
- **Environment Variable**: Never commit CRON_SECRET to git
- **Verification**: Endpoint verifies secret before processing
- **Logging**: All cron runs are logged with results

Admin Endpoints

- **Authentication**: All admin endpoints verify `role = ADMIN`
- **Logging**: All admin actions logged to `TransactionLog`
- **Override**: Admin override is logged and traceable

Payout Safety

- **Eligibility Checks**: Multiple layers of validation
- **Idempotency**: Webhook events use idempotency keys
- **Atomic Operations**: Database transactions for payout creation
- **Error Recovery**: Failed payouts don't mark payments as paid
- **Auto-Blocking**: System auto-blocks on Stripe requirement issues

Webhook Security

- **Signature Verification**: All webhooks verify Stripe signature
- **Idempotency**: Event IDs tracked to prevent duplicate processing
- **Auto-Block**: Automatically blocks payouts when KYC issues detected
- **Auto-Unblock**: Automatically unblocks when issues resolved

---

Platform Settings

Configure payout behavior in `PlatformSettings`:

```
{
  minimumPayoutAmount: 10.00,        // Minimum payout amount (EUR)
  holdingPeriodDays: 7,              // Days to hold payments before payout
  currency: 'EUR',                   // Platform currency
  // ... other settings
}
```

Update via admin API: `PUT /api/admin/settings`

Monitoring and Logging

TransactionLog

All payout operations are logged:

```
{
  eventType: 'PAYOUT_CREATED' | 'PAYOUT_PAID' | 'PAYOUT_FAILED',
  entityType: 'PAYOUT',
  entityId: 'payout_id',
  amount: 100.00,
  currency: 'EUR',
  status: 'PAID',
  metadata: {
    automatic: true,
    paymentCount: 5,
    frequency: 'WEEKLY',
    ...
  }
}
```

Recommended Monitoring

1. **Daily Summary**: Check dashboard for failed payouts
2. **Cron Logs**: Monitor cron execution logs
3. **Stripe Dashboard**: Monitor Connect account issues
4. **Alert Setup**: Alert on high failure rates
5. **Balance Monitoring**: Track platform balance vs. pending payouts

---

Best Practices

For Administrators

1. **Regular Monitoring**: Check dashboard daily
2. **Failed Payout Review**: Investigate failures within 24 hours
3. **KYC Management**: Help creators complete requirements promptly
4. **Balance Management**: Ensure platform has sufficient balance
5. **Blocking**: Use payout blocking sparingly and document reasons

For Developers

1. **Test in Development**: Use test-eligibility endpoint
2. **Staging Environment**: Test cron with test CRON_SECRET
3. **Error Handling**: Always handle Stripe API errors gracefully
4. **Logging**: Add detailed logging for debugging
5. **Rollback Plan**: Know how to pause automatic payouts if needed

Emergency Procedures

To Pause All Automatic Payouts:

```
UPDATE "PayoutSchedule" SET "isActive" = false WHERE "mode" = 'AUTOMATIC';
```

To Re-enable:

```
UPDATE "PayoutSchedule" SET "isActive" = true WHERE "mode" = 'AUTOMATIC';
```

Support

For issues or questions:

1. Check this documentation first
2. Review TransactionLog for detailed error messages
3. Use test-eligibility endpoint to diagnose issues
4. Check Stripe Dashboard for Connect account issues
5. Contact Stripe Support for payment-specific issues

Changelog

Version 1.0 (Phase 3)

- ✅ Automatic payout system implemented
- ✅ Comprehensive eligibility checker
- ✅ Admin control endpoints
- ✅ Cron scheduler with idempotency
- ✅ Webhook auto-blocking on requirements
- ✅ Testing endpoints for development
- ✅ Complete documentation

Last Updated: December 25, 2025