PHP & MySQL Technical Implementation

PHP & MySQL Technical Implementation

Purpose

This document provides a detailed, implementation-level guide for the PMTNM affiliate content syndication system using **PHP** and **MySQL**. It documents the actual implementation as of December 2024.

System Overview

High-Level Flow

1. Affiliate publishes public HTML content using the `#pmtnm-news` container
2. PMTNM cron job (or manual trigger) fetches affiliate pages via `scripts/fetch_affiliate_news.php`
3. PHP parser extracts and sanitizes content using DOMDocument/XPath
4. Parsed content is stored in MySQL (replaces previous items for each affiliate)
5. Homepage (`index.php`) and Affiliate News page (`affiliate-news.php`) render cached content
6. Content changes are versioned in history table and logged in fetch log

---

Database Schema (MySQL)

affiliates

Stores affiliate metadata and feed configuration.

Location: `database/migrations/create_affiliates_table.sql`

```
CREATE TABLE affiliates (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    slug VARCHAR(100) UNIQUE NOT NULL,
    feed_url VARCHAR(255) DEFAULT NULL COMMENT 'URL to scrape for #pmtnm-news content',
```

```
    base_url VARCHAR(255) DEFAULT NULL COMMENT 'Affiliate website base URL',
    description TEXT,
    logo_url VARCHAR(255) DEFAULT NULL,
    contact_email VARCHAR(255) DEFAULT NULL,
    is_active TINYINT(1) DEFAULT 1,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    updated_at DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,

    INDEX idx_slug (slug),
    INDEX idx_is_active (is_active)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

Key Fields:

- `slug`: URL-friendly identifier (e.g., "albuquerque", "santa-fe")
- `feed_url`: URL of page containing `#pmtnm-news` container (can be NULL)
- `base_url`: Base website URL for resolving relative links/images

---

affiliate_news_items

Stores the *current* syndicated items (max 4 per affiliate).

Location: `database/migrations/create_affiliate_news_tables.sql`

```
CREATE TABLE affiliate_news_items (
    id INT AUTO_INCREMENT PRIMARY KEY,
    affiliate_id INT NOT NULL,
    title VARCHAR(255) NOT NULL,
    description TEXT NOT NULL,
    image_url VARCHAR(255) DEFAULT NULL,
    link_url VARCHAR(255) NOT NULL,
    content_hash CHAR(64) NOT NULL COMMENT 'SHA-256 hash for change detection',
    display_order TINYINT DEFAULT 0 COMMENT 'Order within affiliate items (0-3)',
    fetched_at DATETIME NOT NULL,

    INDEX idx_affiliate_id (affiliate_id),
    INDEX idx_fetched_at (fetched_at),
    INDEX idx_content_hash (content_hash),

    FOREIGN KEY (affiliate_id) REFERENCES affiliates(id) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

Behavior: On each fetch, all items for an affiliate are deleted and re-inserted.

---

affiliate_news_history

Stores historical versions for audit and reporting.

```
CREATE TABLE affiliate_news_history (
    id INT AUTO_INCREMENT PRIMARY KEY,
    affiliate_id INT NOT NULL,
    title VARCHAR(255),
    description TEXT,
    image_url VARCHAR(255) DEFAULT NULL,
    link_url VARCHAR(255),
    content_hash CHAR(64),
    recorded_at DATETIME DEFAULT CURRENT_TIMESTAMP,

    INDEX idx_affiliate_id (affiliate_id),
    INDEX idx_recorded_at (recorded_at),
    INDEX idx_content_hash (content_hash),

    FOREIGN KEY (affiliate_id) REFERENCES affiliates(id) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

Behavior: Append-only table. Never deleted, only appended on each fetch.

---

affiliate_fetch_log

Tracks fetch attempts and errors for debugging.

```
CREATE TABLE affiliate_fetch_log (
    id INT AUTO_INCREMENT PRIMARY KEY,
    affiliate_id INT NOT NULL,
    fetch_url VARCHAR(255) NOT NULL,
    status ENUM('success', 'error', 'no_content', 'timeout') NOT NULL,
    items_found INT DEFAULT 0,
    items_stored INT DEFAULT 0,
    error_message TEXT DEFAULT NULL,
    response_time_ms INT DEFAULT NULL,
    fetched_at DATETIME DEFAULT CURRENT_TIMESTAMP,

    INDEX idx_affiliate_id (affiliate_id),
```

```
    INDEX idx_status (status),
    INDEX idx_fetched_at (fetched_at),

    FOREIGN KEY (affiliate_id) REFERENCES affiliates(id) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

Status Values:

- `success` : Content fetched and items stored
- `error` : Connection/timeout error
- `no_content` : No `#pmtnm-news` container found or no items in container
- `timeout` : Request timed out

Cron Job Configuration

Example cron entry (runs every 30 minutes):

```
*/30 * * * * /usr/bin/php /var/www/pmtnm/scripts/fetch_affiliate_news.php
```

PHP Fetch & Parse Script

fetch_affiliate_news.php (core logic)

```
<?php
require_once __DIR__ . '/../config/db.php';
require_once __DIR__ . '/../lib/HtmlSanitizer.php';

$affiliates = $db->query("SELECT * FROM affiliates WHERE is_active = 1");

foreach ($affiliates as $affiliate) {
    $html = @file_get_contents($affiliate['feed_url']);
    if (!$html) continue;

    $dom = new DOMDocument();
    libxml_use_internal_errors(true);
    $dom->loadHTML($html);
    libxml_clear_errors();

    $xpath = new DOMXPath($dom);
```

```
    $container = $xpath->query("//*[@id='pmtnm-news']")->item(0);
    if (!$container) continue;

    $items = $xpath->query(".//div[contains(@class,'pmtnm-news-item')]", $container);

    $db->prepare("DELETE FROM affiliate_news_items WHERE affiliate_id = ?")
       ->execute([$affiliate['id']]);

    $count = 0;
    foreach ($items as $item) {
        if ($count >= 4) break;

        $title = trim($xpath->query(".//div[contains(@class,'pmtnm-news-title')]", $item)->item(0)->textContent ?? '');
        $desc  = trim($xpath->query(".//div[contains(@class,'pmtnm-news-description')]", $item)->item(0)->textContent ?? '');
        $img   = $xpath->query(".//img", $item)->item(0)?->getAttribute('src');
        $link  = $xpath->query(".//a", $item)->item(0)?->getAttribute('href');

        $hash = hash('sha256', $title . $desc . $img . $link);

        // Store history
        $db->prepare("INSERT INTO affiliate_news_history (affiliate_id,title,description,image_url,link_url,content_hash)
                      VALUES (?,?,?,?,?,?)")
           ->execute([$affiliate['id'],$title,$desc,$img,$link,$hash]);

        // Store current
        $db->prepare("INSERT INTO affiliate_news_items (affiliate_id,title,description,image_url,link_url,content_hash,fetched_at)
                      VALUES (?,?,?,?,?,?,NOW())")
           ->execute([$affiliate['id'],$title,$desc,$img,$link,$hash]);

        $count++;
    }
}
```

Sanitization & Security

Text Sanitization

- **Strip all HTML tags**: Uses `strip_tags()` to remove all HTML
- **Decode entities**: Converts HTML entities to plain text
- **Normalize whitespace**: Collapses multiple spaces/newlines to single space
- **Result**: Plain text only, no HTML/scripts

URL Sanitization

- **Block javascript: URLs**: Prevents XSS via javascript: protocol
- **Resolve relative URLs**: Converts relative URLs to absolute using `base_url`
- **Validate URLs**: Uses `filter_var()` with `FILTER_VALIDATE_URL`
- **Result**: Only valid HTTP/HTTPS URLs stored

Security Considerations

- **No script execution**: All HTML is stripped, only text content extracted
- **No inline event handlers**: Not possible since HTML is removed
- **HTTPS preferred**: URLs can be HTTP or HTTPS (not enforced, but recommended)
- **Read-only access**: Affiliate sites are only read, never written to

---

Display Component

affiliate_news_feed.php

Location: `includes/components/affiliate_news_feed.php`

Usage:

```
$affiliate_news_limit = 4;  // Max items to show (default: 4)
$affiliate_news_show_affiliate = true;  // Show affiliate name badge
include __DIR__ . '/includes/components/affiliate_news_feed.php';
```

Query:

```
$stmt = $pdo->prepare("
    SELECT n.*, a.name as affiliate_name, a.slug as affiliate_slug
    FROM affiliate_news_items n
    JOIN affiliates a ON a.id = n.affiliate_id
    WHERE a.is_active = 1
```

```
    ORDER BY n.fetched_at DESC, n.display_order ASC
    LIMIT ?
");
$stmt->execute([$affiliate_news_limit]);
```

Display Locations:

- **Homepage** (`index.php`): Shows 4 items
- **Affiliate News Page** (`affiliate-news.php`): Shows up to 20 items

Features:

- Responsive grid layout
- Lazy loading images
- Affiliate name badges
- Graceful degradation (silently fails if database unavailable)

Admin Interface

Affiliate Management

Location: `admin/affiliates/`

Pages:

- `index.php` : List all affiliates, create new, toggle active status
- `edit.php` : Edit affiliate details, manage affiliate admins
- `members.php` : View/manage members assigned to affiliate
- `fetch_news.php` : Manual trigger for fetching news, view fetch logs

Access Control:

- **Super Admin**: Full CRUD access, can assign affiliate admins
- **Admin**: Can view and edit affiliates
- **Affiliate Admin**: Can only edit `feed_url` for their assigned affiliate

AffiliateManager Class

Location: `includes/AffiliateManager.php`

Key Methods:

- `getAllAffiliates($activeOnly)` : Get all affiliates
- `getAffiliateById($id)` : Get affiliate by ID
- `getAffiliateBySlug($slug)` : Get affiliate by slug
- `createAffiliate($data)` : Create new affiliate
- `updateAffiliate($id, $data)` : Update affiliate
- `deleteAffiliate($id)` : Delete affiliate (super admin only)
- `assignMemberToAffiliate($email, $affiliateId)` : Assign member to affiliate
- `setAffiliateAdmin($email, $isAdmin)` : Grant/revoke affiliate admin status
- `canManageAffiliate($email, $affiliateId)` : Check if user can manage affiliate
- `getStatistics()` : Get system statistics

Error Handling & Monitoring

Fetch Logging

Every fetch attempt is logged to `affiliate_fetch_log` with:

- Status (success, error, no_content, timeout)
- Items found vs. items stored
- Response time in milliseconds
- Error messages (if any)

Error Handling

- **Connection failures**: Logged, affiliate skipped, fetch continues
- **Missing container**: Logged as `no_content`, affiliate skipped
- **Invalid HTML**: libxml errors suppressed, parsing continues
- **Database errors**: Logged to error_log, page render continues gracefully

Monitoring

- Fetch logs available in admin interface (`admin/affiliates/fetch_news.php`)
- Statistics dashboard shows active affiliates, feed URLs, member counts

- No public error exposure (errors logged internally only)

Governance Notes (Non-Technical but Critical)

- **Content ingestion is editorial, not approval-based**: Content is automatically ingested, no manual approval
- **No deletion of historical records**: `affiliate_news_history` is append-only
- **Logs are internal and role-restricted**: Only admins can view fetch logs
- **Affiliates retain ownership of content**: PMTNM only displays, doesn't claim ownership
- **Affiliate admins can update feed URLs**: Self-service capability for affiliates

---

File Structure

```
database/migrations/
  ├── create_affiliates_table.sql          # Affiliate metadata
  └── create_affiliate_news_tables.sql    # News items, history, fetch log

scripts/
  └── fetch_affiliate_news.php            # Fetch script (CLI/web)

includes/
  ├── AffiliateManager.php                # CRUD operations
  └── components/
      └── affiliate_news_feed.php         # Display component

admin/affiliates/
  ├── index.php                           # List/manage affiliates
  ├── edit.php                            # Edit affiliate
  ├── members.php                         # Manage members
  └── fetch_news.php                      # Manual fetch trigger

public/
  ├── affiliate-news.php                  # Dedicated news page
  └── affiliate-webmaster-guide.md        # Public webmaster guide

docs/
  ├── pmtnm_affiliate_content_syndication_technical_overview.md
  ├── pmtnm_affiliate_content_syndication_php_my_sql_technical_implementation.md
```

```
└── pmtnm_affiliate_content_syndication_webmasters_guide.md
```

Summary

This PHP/MySQL implementation:

- **Simple and auditable**: Clear code structure, comprehensive logging
- **Avoids API credential sharing**: Pull-based, no credentials needed
- **Preserves affiliate autonomy**: Affiliates control their own content
- **Scales to statewide networks**: Supports unlimited affiliates
- **Self-service capability**: Affiliate admins can update feed URLs
- **Comprehensive monitoring**: Fetch logs, statistics, error tracking
- **Production-ready**: Error handling, sanitization, security measures