Phase 4 Automation Readiness Documentation

This document outlines all automation hooks and integration points ready for Phase 4 content automation.

🎯 Phase 4 Overview

Goal: Automated content generation and publishing workflow
- **Weekly:** 30-40 topic generation with keyword research
- **Daily:** 3-4 articles per language (EN/AR)
- **Real-time:** AI quality testing, SEO optimization, and publishing
- **Continuous:** Performance monitoring and search indexing

🔌 Programmatic API Endpoints (Ready)

Content Creation Automation

1. Draft Article Generation

```
// POST /api/content/auto-generate
interface GenerateContentRequest {
  topic: string;
  contentType: 'blog_post' | 'instagram_post' | 'tiktok_video';
  language: 'en' | 'ar';
  targetAudience: string;
  keywords: string[];
  tone: 'luxury' | 'casual' | 'informative';
}

// Usage in Phase 4:
const article = await fetch('/api/content/auto-generate', {
  method: 'POST',
  body: JSON.stringify({
    topic: "Best luxury spas in London for winter wellness",
    contentType: 'blog_post',
    language: 'en',
    targetAudience: 'luxury travelers',
    keywords: ['luxury spa London', 'winter wellness', 'spa treatments'],
    tone: 'luxury'
  })
});
```

2. Content Scheduling

```
// POST /api/content/schedule
interface ScheduleContentRequest {
  title: string;
  content: string;
  contentType: string;
  language: string;
  scheduledTime: string; // ISO datetime
  category?: string;
  tags?: string[];
  metadata?: Record<string, any>;
}

// Bulk scheduling for daily automation:
const batchSchedule = await fetch('/api/admin/scheduled-content', {
  method: 'POST',
  body: JSON.stringify({
    articles: [
      { title: "...", content: "...", scheduledTime: "2024-01-15T09:00:00Z" },
      { title: "...", content: "...", scheduledTime: "2024-01-15T14:00:00Z" },
      { title: "...", content: "...", scheduledTime: "2024-01-15T18:00:00Z" }
    ]
  })
});
```

Media Assignment Automation

3. Automatic Image Assignment

```
// POST /api/media/assign-automatically
interface AutoAssignMediaRequest {
  contentId: string;
  contentType: 'blog_post' | 'instagram_post';
  keywords: string[];
  imageRole: 'hero' | 'thumbnail' | 'inline' | 'gallery';
  aspectRatio?: string;
}

// Phase 4 usage:
await fetch('/api/media/assign-automatically', {
  method: 'POST',
  body: JSON.stringify({
    contentId: articleId,
    contentType: 'blog_post',
    keywords: ['london spa', 'luxury wellness', 'interior'],
    imageRole: 'hero',
    aspectRatio: '16:9'
  })
});
```

4. Programmatic Media Upload

```
// POST /api/media/upload-from-url
interface UploadFromUrlRequest {
  imageUrl: string;
  altText: string;
  title: string;
  tags: string[];
  licenseInfo?: string;
  autoOptimize?: boolean;
}

// For AI-generated or stock images:
const media = await fetch('/api/media/upload-from-url', {
  method: 'POST',
  body: JSON.stringify({
    imageUrl: 'https://i.pinimg.com/736x/b4/8d/ed/b48dedb44d89f2e2a9a1356bf4628-
bea.jpg',
    altText: 'Luxury spa interior with marble features in London',
    title: 'London Luxury Spa Interior',
    tags: ['spa', 'luxury', 'london', 'wellness', 'interior'],
    autoOptimize: true
  })
});
```

Social Media Integration

5. Programmatic Embed Insertion

```
// POST /api/social-embeds/auto-insert
interface AutoInsertEmbedRequest {
  contentId: string;
  socialUrls: string[];
  placement: 'beginning' | 'middle' | 'end' | 'contextual';
  platform?: string;
}

// Automatically embed relevant social content:
await fetch('/api/social-embeds/auto-insert', {
  method: 'POST',
  body: JSON.stringify({
    contentId: articleId,
    socialUrls: [
      'https://www.instagram.com/p/spa-london-post/',
      'https://www.tiktok.com/@london_wellness/video/123456'
    ],
    placement: 'contextual' // AI determines best placement
  })
});
```

Homepage Automation

6. Dynamic Homepage Blocks

```
// POST /api/homepage-blocks/feed-content
interface FeedContentRequest {
  blockType: 'featured-experiences' | 'events' | 'blog-grid';
  contentSource: 'latest' | 'trending' | 'category' | 'ai-curated';
  maxItems: number;
  filters?: {
    category?: string;
    tags?: string[];
    dateRange?: string;
  };
}

// Daily homepage refresh with latest content:
await fetch('/api/homepage-blocks/feed-content', {
  method: 'POST',
  body: JSON.stringify({
    blockType: 'blog-grid',
    contentSource: 'ai-curated',
    maxItems: 6,
    filters: {
      category: 'luxury-experiences',
      dateRange: 'last-7-days'
    }
  })
});
```

Publishing & Distribution

7. Automated Publishing Pipeline

```
// POST /api/content/publish-pipeline
interface PublishPipelineRequest {
  contentId: string;
  actions: {
    generateSitemap?: boolean;
    submitToGSC?: boolean;
    submitToBing?: boolean;
    sendGA4Event?: boolean;
    createSocialPosts?: boolean;
    notifySubscribers?: boolean;
  };
}

// Complete publishing workflow:
await fetch('/api/content/publish-pipeline', {
  method: 'POST',
  body: JSON.stringify({
    contentId: articleId,
    actions: {
      generateSitemap: true,
      submitToGSC: true,
      submitToBing: true,
      sendGA4Event: true,
      createSocialPosts: false, // Manual approval required
      notifySubscribers: true
    }
  })
});
```

8. Indexing Status Monitoring

```
// GET /api/seo/indexing-status
interface IndexingStatusResponse {
  contentId: string;
  url: string;
  gscStatus: 'submitted' | 'crawled' | 'indexed' | 'error';
  bingStatus: 'submitted' | 'crawled' | 'indexed' | 'error';
  lastUpdated: string;
  crawlErrors?: string[];
}

// Monitor indexing progress:
const status = await fetch(`/api/seo/indexing-status/${articleId}`);
```

🗄️ Database Schema (Phase 4 Ready)

Content Automation Tables

```
-- Content scheduling with automation metadata
model ScheduledContent {
  id             String   @id @default(cuid())
  title          String
  content        String   @db.Text
  content_type   String   // blog_post, instagram_post, tiktok_video
  language       String   // en, ar
  category       String?
  tags           String[]
  metadata       Json?    // AI-generated metadata, keywords, etc.
  scheduled_time DateTime
  published_time DateTime?
  status         String   @default("pending") // pending, published, failed, cancelled
  platform       String?  // blog, instagram, tiktok

  -- Phase 4 automation fields
  automation_id  String?  // Links to automation batch
  ai_confidence  Float?   // AI quality score (0-1)
  seo_score      Float?   // SEO optimization score
  indexing_status Json?   // GSC/Bing indexing status
  performance_metrics Json? // Views, engagement, etc.
}

-- Social media content queue
model SocialEmbed {
  id           String   @id @default(cuid())
  platform     String   // instagram, tiktok, facebook, youtube
  url          String   // Original social media URL
  embed_id     String   // Extracted ID from URL
  thumbnail    String?  // Cloud storage path to thumbnail image
  title        String?
  author       String?
  aspect_ratio String   @default("16:9")
  usage_count  Int      @default(0)
  status       String   @default("active") // active, disabled, pending_review
  metadata     Json?

  -- Phase 4 automation integration
  auto_assigned Boolean @default(false) // Was this auto-assigned to content?
  content_context Json? // Where/why this embed was used
  ai_relevance_score Float? // How relevant AI thinks this is
}

-- Media assets with automation support
model MediaAsset {
  id              String   @id @default(cuid())
  filename        String
  original_name   String
  cloud_storage_path String // S3 key - never store local paths
  url             String   // Public URL for serving
  file_type       String   // image, video, document
  mime_type       String
  file_size       Int
  width           Int?
  height          Int?
  alt_text        String?
  title           String?
  description     String?  @db.Text
  tags            String[]
  license_info    String?
  usage_map       Json?    // Track where this asset is used
  responsive_urls Json?    // Generated responsive variants
  -- Phase 4 automation fields
  ai_generated    Boolean @default(false) // Was this AI generated?
  auto_assigned   Boolean @default(false) // Was this auto-assigned?
  optimization_score Float? // Image optimization score
  seo_keywords    String[] // SEO-relevant keywords for this image
}
```

Automation Workflow Tables

```
-- Content generation batches
model ContentAutomationBatch {
  id          String   @id @default(cuid())
  batch_name  String   // e.g., "daily-articles-2024-01-15"
  batch_type  String   // daily, weekly, emergency
  topic_count Int
  generated_count Int  @default(0)
  published_count Int  @default(0)
  failed_count    Int  @default(0)
  status      String   @default("pending") // pending, running, completed, failed
  started_at  DateTime @default(now())
  completed_at DateTime?
  metadata    Json?    // Topic list, keywords, etc.
}

-- SEO and indexing tracking
model SEOIndexingStatus {
  id          String   @id @default(cuid())
  content_id  String   // Links to blog post or content
  url         String   // Full URL of content
  gsc_status  String?  // Google Search Console status
  bing_status String?  // Bing Webmaster status
  sitemap_submitted DateTime?
  last_crawled DateTime?
  indexed_at   DateTime?
  crawl_errors Json?   // Any crawling errors
  performance_data Json? // Click-through rates, impressions, etc.
}
```

⚡ Automation Workflows (Integration Points)

Daily Content Generation (Phase 4 Week 1)

```javascript
// Cron job: Daily content generation
// File: /api/cron/daily-content-generation

export async function dailyContentGeneration() {
  // 1. Generate topics based on trending keywords
  const topics = await generateTopics({
    count: 4,
    languages: ['en', 'ar'],
    categories: ['luxury-experiences', 'events', 'dining', 'culture']
  });

  // 2. Create content batch
  const batch = await createAutomationBatch({
    batchName: `daily-articles-${new Date().toISOString().split('T')[0]}`,
    batchType: 'daily',
    topics
  });

  // 3. Generate articles for each topic
  for (const topic of topics) {
    const article = await generateContent({
      topic: topic.title,
      keywords: topic.keywords,
      language: topic.language,
      contentType: 'blog_post'
    });

    // 4. Auto-assign hero image
    await assignHeroImage({
      contentId: article.id,
      keywords: topic.keywords,
      aspectRatio: '16:9'
    });

    // 5. Schedule for publication
    await scheduleContent({
      contentId: article.id,
      scheduledTime: calculatePublishTime(topic.priority),
      automation_id: batch.id
    });
  }

  return { batch, generatedCount: topics.length };
}
```

Weekly Topic Research (Phase 4 Week 2)

```
// Cron job: Weekly topic and keyword research
// File: /api/cron/weekly-topic-research

export async function weeklyTopicResearch() {
  // 1. Analyze trending topics
  const trendingTopics = await analyzeTrendingTopics({
    location: 'London',
    categories: ['luxury', 'travel', 'dining', 'events'],
    timeRange: 'last-7-days'
  });

  // 2. Perform keyword research
  const keywords = await performKeywordResearch({
    topics: trendingTopics,
    targetAudience: 'luxury travelers',
    competition: 'low-medium'
  });

  // 3. Generate content calendar
  const contentCalendar = await generateContentCalendar({
    topics: trendingTopics,
    keywords,
    duration: '7-days',
    articlesPerDay: 3
  });

  // 4. Create content automation batch
  return await createAutomationBatch({
    batchName: `weekly-calendar-${new Date().toISOString().split('T')[0]}`,
    batchType: 'weekly',
    contentCalendar
  });
}
```

Real-time Quality Assurance (Phase 4 Week 3)

```
// Real-time content quality checking
// File: /lib/content-qa-automation

export async function performContentQA(contentId: string) {
  const content = await getContent(contentId);

  // 1. SEO Analysis
  const seoScore = await analyzeSEO({
    title: content.title,
    content: content.content,
    targetKeywords: content.metadata?.keywords || []
  });

  // 2. AI Quality Check
  const qualityScore = await checkContentQuality({
    content: content.content,
    language: content.language,
    contentType: content.contentType
  });

  // 3. Readability Analysis
  const readabilityScore = await analyzeReadability({
    content: content.content,
    language: content.language
  });

  // 4. Brand Voice Alignment
  const brandScore = await checkBrandAlignment({
    content: content.content,
    brandGuidelines: getBrandGuidelines()
  });

  const overallScore = (seoScore + qualityScore + readabilityScore + brandScore) / 4;

  // 5. Auto-publish if score is high enough
  if (overallScore >= 0.85) {
    await publishContent(contentId);
    await submitToSearchEngines(contentId);
  } else {
    await flagForHumanReview(contentId, {
      scores: { seo: seoScore, quality: qualityScore, readability: readabilityScore, br
and: brandScore },
      overallScore
    });
  }

  return { overallScore, autoPublished: overallScore >= 0.85 };
}
```

Performance Monitoring (Phase 4 Week 4)

```
// Content performance tracking and optimization
// File: /api/cron/performance-monitoring

export async function monitorContentPerformance() {
  // 1. Get recently published content
  const recentContent = await getRecentlyPublishedContent({
    timeRange: 'last-30-days',
    limit: 50
  });

  // 2. Collect performance metrics
  for (const content of recentContent) {
    const metrics = await collectPerformanceMetrics({
      contentId: content.id,
      url: content.url,
      sources: ['google-analytics', 'search-console', 'social-media']
    });

    // 3. Update indexing status
    const indexingStatus = await checkIndexingStatus(content.url);

    await updateIndexingStatus({
      contentId: content.id,
      gscStatus: indexingStatus.googleStatus,
      bingStatus: indexingStatus.bingStatus,
      performanceData: metrics
    });

    // 4. Auto-optimize underperforming content
    if (metrics.organicTraffic < 50 && metrics.daysPublished > 7) {
      await optimizeContent({
        contentId: content.id,
        currentMetrics: metrics,
        optimizationType: 'seo-boost'
      });
    }
  }

  return { analyzedContent: recentContent.length };
}
```

🎛️ Configuration & Settings

Feature Flags (Already Implemented)

```
# Enable Phase 4 automation features
FEATURE_CONTENT_AUTOMATION=1
FEATURE_AUTO_PUBLISHING=1
FEATURE_AI_QUALITY_CHECK=1
FEATURE_AUTO_IMAGE_ASSIGNMENT=1
FEATURE_PERFORMANCE_MONITORING=1
```

Automation Settings

```typescript
// /config/automation-settings.ts
export const automationConfig = {
  dailyGeneration: {
    enabled: true,
    articlesPerDay: 3,
    languages: ['en', 'ar'],
    scheduleTimes: ['09:00', '14:00', '18:00'], // UTC
    qualityThreshold: 0.85,
    autoPublish: true
  },

  weeklyResearch: {
    enabled: true,
    topicCount: 30,
    keywordResearch: true,
    competitorAnalysis: true,
    trendingTopics: true
  },

  qualityAssurance: {
    seoMinScore: 0.8,
    readabilityMinScore: 0.7,
    brandAlignmentMinScore: 0.8,
    humanReviewThreshold: 0.85
  },

  performance: {
    monitoringInterval: '1-hour',
    optimizationTrigger: 7, // days
    minTrafficThreshold: 50,
    reoptimizeAfter: 30 // days
  }
};
```

🚀 Phase 4 Launch Sequence

Week 1: Content Generation Pipeline

1. **Enable daily content generation**
2. **Test topic research and article generation**
3. **Verify image assignment automation**
4. **Monitor content queue and publishing**

Week 2: Quality Assurance Integration

1. **Implement AI quality scoring**
2. **Set up SEO optimization checks**
3. **Configure human review workflows**
4. **Test automated publishing pipeline**

Week 3: Performance Optimization

1. **Enable search engine submission automation**
2. **Set up indexing status monitoring**
3. **Configure performance tracking**

4. **Implement content optimization triggers**

Week 4: Full Automation

1. **Enable complete automation pipeline**
2. **Monitor and tune automation parameters**
3. **Analyze content performance trends**
4. **Optimize workflow based on results**

Status: ✅ All automation hooks and database schema are ready for Phase 4 implementation. The CMS architecture fully supports automated content workflows with proper monitoring, quality assurance, and performance tracking.