NeuroVision
AI-Powered Social Media Automation

Service Blueprint

Process Flow with Technical Nodes

STAGE 1: Content Strategy & Planning

Node 1.1: User Login

- **User Action**: User accesses admin dashboard
- **System Action**: Authenticate user, load dashboard
- **Technical Requirements**:
  - Auth system (JWT tokens)
  - Admin UI (Bolt/Lovable)
  - Session management
- **API Requirements**: None (frontend authentication)
- **n8n Nodes**: N/A (UI only)

Node 1.2: Define Monthly Strategy

- **User Action**: Input content strategy for the month
  - Monthly theme (e.g., "Concussion Awareness")
  - Content pillar distribution, i.e.:
    - Education: 40%

        - Patient Success Stories: 30%
        - Thought Leadership: 20%
        - Clinic Updates: 10%
    - Target audience focus
    - Key topics to cover
- **System Action**: Store strategy configuration in database
- **Technical Requirements**:
    - Form interface with dropdown/sliders
    - Data validation
    - Database write
- **API Requirements**: Database API
- **n8n Nodes**:
    - Webhook trigger
    - Data validation node
    - Database insert node
    - Success confirmation response

Data Schema:

```
JSON
    {
      "strategy_id": "uuid",
      "month": "2025-11",
      "theme": "Concussion Awareness",
      "pillars": {
        "education": 40,
        "success_stories": 30,
        "thought_leadership": 20,
        "clinic_updates": 10
      },
      "topics": ["concussion recovery", "post-trauma vision", "therapy benefits"],
      "created_at": "timestamp"
```

```
        ○ }
```

Node 1.3: Set Posting Schedule

- **User Action**: Configure posting frequency per platform
  - LinkedIn: 3x/week (Mon/Wed/Fri, 9 AM ET)
  - Instagram: 5x/week (Mon-Fri, 12 PM ET) + Daily Stories
  - Facebook: 3x/week (Tue/Thu/Sat, 10 AM ET)
- **System Action**: Create calendar template for the month
- **Technical Requirements**:
  - Calendar UI component
  - Timezone handling (ET)
  - Recurring schedule logic
- **API Requirements**: Database write
- **n8n Nodes**:
  - Webhook trigger (schedule submission)
  - Code node (generate dates for month)
  - Loop node (create calendar entries)
  - Database bulk insert

Data Schema:

JSON

```
        ○ {
        ○   "calendar_id": "uuid",
        ○   "month": "2025-11",
        ○   "entries": [
        ○     {
        ○       "date": "2025-11-04",
```

```
        "time": "09:00:00",
        "platform": "linkedin",
        "content_pillar": "education",
        "status": "pending"
      }
    ]
  }
```

STAGE 2: Automated Content Generation

Trigger → AI Processing → Storage

Node 2.1: Trigger Weekly Generation

- **Trigger**: Scheduled (Every Sunday at 6 PM ET)
- **System Action**: Query calendar for upcoming week's posts
- **Technical Requirements**: Cron-style scheduler
- **API Requirements**: Database query
- **n8n Nodes**:
  - Schedule trigger node
  - Database query (get next 7 days of calendar entries)
  - Loop through entries

Node 2.2: Generate Text Content

- **System Action**: For each calendar entry, generate post text
- **Input Data**:
  - Content pillar from calendar
  - Monthly theme

    - Platform (LinkedIn/Instagram/Facebook)
    - Topics from strategy
- **AI Prompt Template**:

None

```
Role: You are a social media content creator for NeuroVision Therapy Clinic, a vision therapy practice specializing in concussion recovery and learning-related vision problems.

Brand Voice: Professional, empathetic, evidence-based, hopeful

Platform: [LinkedIn/Instagram/Facebook]
Content Pillar: [Education/Success Story/Thought Leadership/Clinic Update]
Monthly Theme: [theme]
Topic: [specific topic]

Requirements:
- [Platform-specific character limit]
- Include clear call-to-action
- Use accessible language (avoid heavy medical jargon)
- Incorporate relevant hashtags
- [Platform-specific format requirements]

Generate a social media post following these guidelines.
```

- **Technical Requirements**:
    - LLM API integration
    - Prompt engineering
    - Response parsing
- **API Requirements**: Anthropic Claude or Gemini or OpenAI
- **n8n Nodes**:

- Loop through calendar entries
- Set variables (build prompt from entry data)
- HTTP request node (LLM API)
- Parse JSON response
- Extract text, hashtags, CTA

Response Format:

JSON

```
{
  "post_text": "Content here...",
  "hashtags": ["#VisionTherapy", "#ConcussionRecovery"],
  "cta": "Learn more at our website",
  "character_count": 280
}
```

Node 2.3: Generate Image Content

- **System Action**: Create branded image for each post
- **Input Data**:
  - Generated text content
  - Platform (for aspect ratio)
  - Brand style guidelines
- **Image Generation Prompt**:

None

```
Create a professional medical illustration for social media.
Topic: [derived from post text]
Style: Clean, modern, minimalist
Colors: Blues and teals (NeuroVision brand palette)
Mood: Hopeful, professional
```

```
        ○ Aspect Ratio: [1:1 for Instagram, 1.91:1 for LinkedIn,
          1.91:1 for Facebook]
        ○ No text overlay
```

- **Technical Requirements**:
  - Image generation API
  - Image storage
  - URL management
- **API Requirements**:
  - DALL-E 3 (OpenAI) or Midjourney
  - Cloud storage
- **n8n Nodes**:
  - HTTP request (image generation API)
  - Download image node
  - Upload to cloud storage
  - Extract image URL

Node 2.4: Store Generated Content

- **System Action**: Save content to database with metadata
- **Technical Requirements**: Database write with JSON data
- **API Requirements**: Database API
- **n8n Nodes**:
  - Combine text + image data
  - Database insert node
  - Update calendar entry status (pending → generated)

Data Schema:

JSON

```
{
  "content_id": "uuid",
  "calendar_entry_id": "uuid",
  "platform": "linkedin",
  "content_text": "Post content...",
  "hashtags": ["#VisionTherapy"],
  "image_url": "https://storage.../image.png",
  "scheduled_date": "2025-11-04T09:00:00-05:00",
  "content_pillar": "education",
  "status": "generated",
  "created_at": "timestamp",
  "engagement_metrics": null
}
```

STAGE 3: Human Review & Approval

UI Interaction → Decision Logic → Status Update

Node 3.1: Display Review Dashboard

- **User Action**: User views pending content in dashboard
- **System Action**: Query and display all "generated" status posts
- **UI Components**:
  - Card grid showing each post
  - Platform-specific preview (LinkedIn style vs Instagram style)
  - Preview of image
  - Edit buttons
  - Approve/Reject buttons
  - Regenerate button
- **Technical Requirements**:

    - Responsive UI (Bolt/Lovable)
    - Real-time database query
    - Platform-specific CSS for previews
- **API Requirements**: Database query
- **n8n Nodes**: N/A (UI display only)

Node 3.2: Edit Content

- **User Action**: Click edit, modify text or request new image
- **System Action**: Update content in database
- **Technical Requirements**:
    - Inline editing interface
    - Image upload option (if user wants custom image)
- **API Requirements**:
    - Database update
    - Cloud storage (if new image uploaded)
- **n8n Nodes**:
    - Webhook trigger (edit submission)
    - Database update node

Node 3.3: Approval Decision

- **User Action**: Click "Approve" or "Reject" button
- **System Action**: Update status and route accordingly
- **Technical Requirements**: Button actions, status logic
- **API Requirements**: Database update
- **n8n Nodes**:
    - Webhook trigger (approval/rejection)
    - IF node (check decision)
    - Database update node

If APPROVED:

- Update status: generated → approved
- Move to optimization stage

If REJECTED:

- Update status: generated → rejected
- Store rejection reason (optional)
- Trigger regeneration flow

Node 3.4: Regenerate Content (If Rejected)

- **System Action**: Return to Node 2.2 with feedback
- **Input Data**:
  - Original prompt
  - Rejection reason/feedback
  - Previous generated content (to avoid duplication)
- **Enhanced Prompt**:

```
None
```

```
[Original prompt]

Previous attempt feedback: [rejection reason]
Previous content to avoid: [old content]

Please generate new content addressing the feedback.
```

- **API Requirements**: LLM API (regeneration)
- **n8n Nodes**:
  - Webhook trigger (regenerate request)
  - Build enhanced prompt
  - HTTP request (LLM)
  - Replace old content in database
  - Reset status to "generated" for re-review

STAGE 4: Platform Optimization

Approved Content → Platform Formatting → Timing Optimization

Node 4.1: Format for Specific Platforms

- **Trigger**: Content status changes to "approved"
- **System Action**: Create platform-specific versions
- **Technical Requirements**: Format conversion logic
- **API Requirements**: None (internal processing)
- **n8n Nodes**:
    - Webhook trigger (approval event)
    - Get approved content
    - Code node (JavaScript formatting logic)

Platform-Specific Rules:

LinkedIn:

```javascript
{
  text: content.text + "\n\n" + content.hashtags.join(" "),
  maxLength: 3000,
  optimalLength: 150-300,
  imageSize: "1200x627",
  includeURL: true,
  format: "professional"
}
```

Instagram:

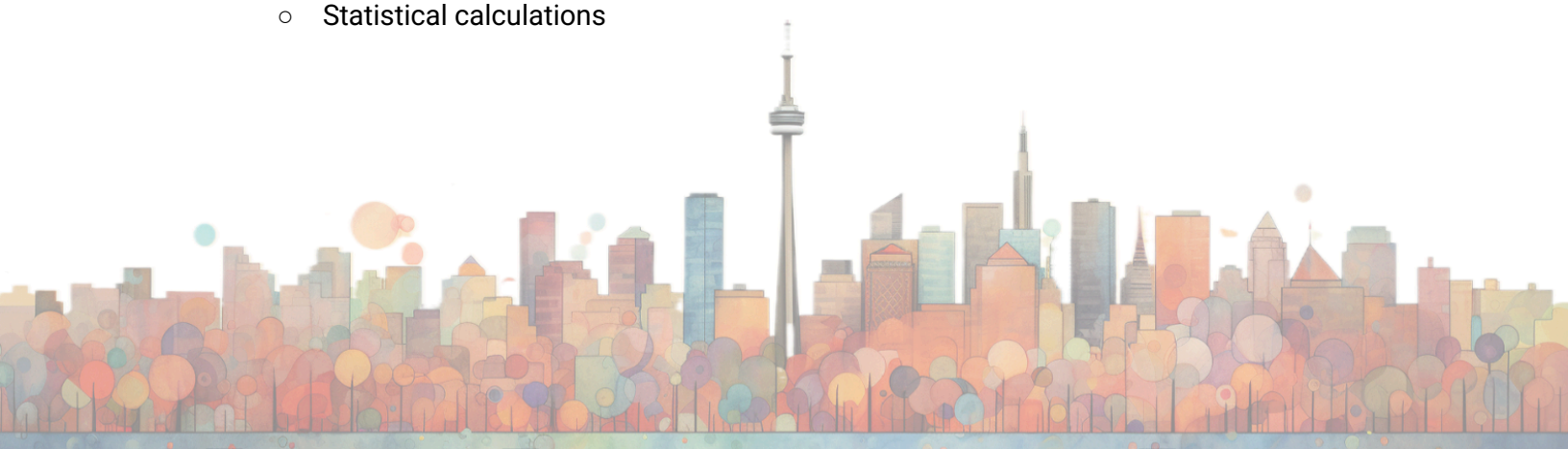
```javascript
{
  text: content.text,
  maxLength: 2200,
  optimalLength: 125-150,
  imageSize: "1080x1080",
  hashtagsInComment: true, // Post hashtags as first comment
  maxHashtags: 30,
  includeLocation: "Whitby, Ontario"
}
```

Facebook:

```javascript
{
  text: content.text + "\n\n" +
content.hashtags.slice(0,5).join(" "),
  maxLength: 63206,
  optimalLength: 40-80,
  imageSize: "1200x630",
  includeURL: true
}
```

Node 4.2: Analyze Optimal Timing

- **System Action**: Determine best posting time based on data
- **Technical Requirements**:
  - Historical data analysis
  - Statistical calculations



    - Time zone handling
- **API Requirements**: Database query (historical engagement)
- **n8n Nodes**:
    - Database query (get last 90 days engagement by hour/day)
    - Code node (calculate engagement rate by time slot)
    - IF node (compare to current scheduled time)
    - Database update (adjust scheduled_date if needed)

Logic:

JavaScript

```
// Query engagement data
const historicalData = await db.query(`
  SELECT
    EXTRACT(HOUR FROM published_at) as hour,
    EXTRACT(DOW FROM published_at) as day_of_week,
    AVG(engagement_rate) as avg_engagement
  FROM content
  WHERE platform = 'linkedin'
    AND published_at > NOW() - INTERVAL '90 days'
  GROUP BY hour, day_of_week
  ORDER BY avg_engagement DESC
  LIMIT 5
`);

// Get top performing time slot for this day of week
const dayOfWeek = new Date(scheduledDate).getDay();
const optimalHour = historicalData
  .filter(d => d.day_of_week === dayOfWeek)[0]?.hour || 9;
// default 9 AM

// Update scheduled time if different
```

```
    ○ if (currentHour !== optimalHour) {
    ○   updateScheduledTime(optimalHour);
    ○ }
```

Node 4.3: Update Queue

- **System Action**: Mark content as ready for publishing
- **Technical Requirements**: Status update
- **API Requirements**: Database update
- **n8n Nodes**:
    - Database update (status: approved → queued)
    - Log scheduled publish time

STAGE 5: Publishing

Queue Management → API Calls → Status Tracking

Node 5.1: Check Publishing Queue

- **Trigger**: Every 15 minutes (cron schedule)
- **System Action**: Query for posts due to be published
- **Technical Requirements**: Time comparison logic
- **API Requirements**: Database query
- **n8n Nodes**:
    - Schedule trigger (every 15 min)
    - Database query:

```sql
    ○ SELECT * FROM content
```

```
  WHERE status = 'queued'
    AND scheduled_date <= NOW() + INTERVAL '15 minutes'
    AND scheduled_date > NOW() - INTERVAL '15 minutes'
```

Node 5.2: Route by Platform

- **System Action**: Direct content to appropriate publishing flow
- **Technical Requirements**: Conditional routing
- **API Requirements**: None (routing logic)
- **n8n Nodes**:
  - Loop through due posts
  - Switch node (route by platform value)
    - Case: "linkedin" → Node 5.3a
    - Case: "instagram" → Node 5.3b
    - Case: "facebook" → Node 5.3c

Node 5.3a: Publish to LinkedIn

- **System Action**: Post content via LinkedIn API
- **API Endpoint**: `POST /rest/posts`
- **Authentication**: OAuth 2.0 (requires LinkedIn Company Page access)
- **Request Format**:

JSON

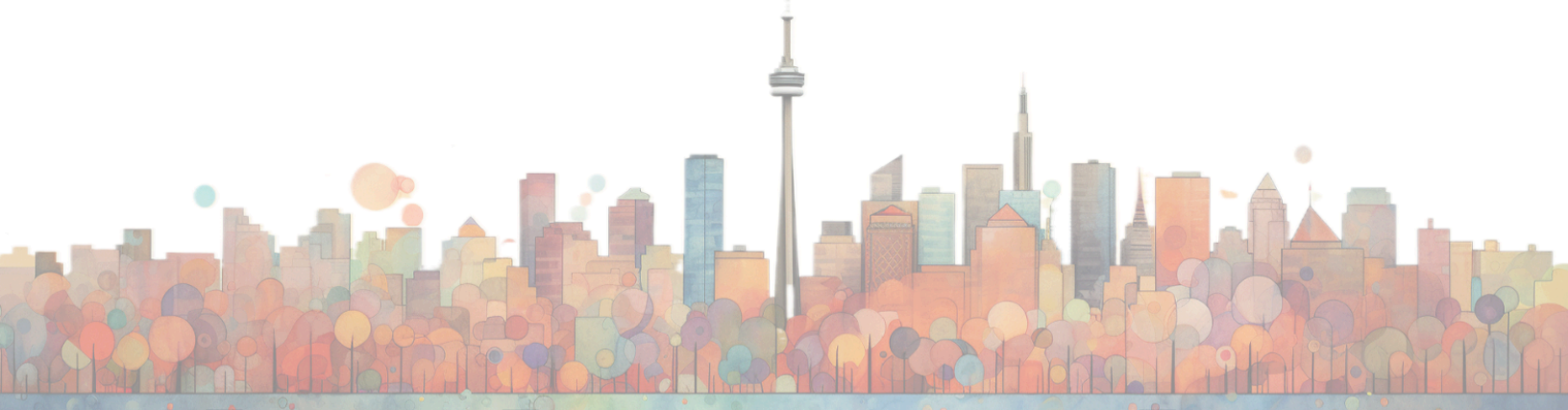
```
{
  "author": "urn:li:organization:{COMPANY_ID}",
  "commentary": "Post text with hashtags",
  "visibility": "PUBLIC",
```

```
    "distribution": {
      "feedDistribution": "MAIN_FEED"
    },
    "content": {
      "media": {
        "title": "Image title",
        "id": "IMAGE_URN"
      }
    },
    "lifecycleState": "PUBLISHED"
  }
```

- **Technical Requirements**:
  - OAuth token management (refresh if expired)
  - Image must be uploaded to LinkedIn first (separate API call)
  - Rate limiting handling
- **n8n Nodes**:
  - HTTP request (upload image to LinkedIn)
  - Extract image URN
  - HTTP request (create post)
  - Parse response (get post ID)

Node 5.3b: Publish to Instagram

- **System Action**: Post content via Instagram Graph API
- **Prerequisites**:
  1. Facebook Business account connected
  2. Instagram Business account
  3. Page access token
- **API Process** (2-step):
  1. Create media container: `POST /{instagram-account-id}/media`



2. Publish container: `POST /{instagram-account-id}/media_publish`

Request Format:

JSON

```
// Step 1: Create container
{
  "image_url": "https://storage.../image.png",
  "caption": "Post text\n.\n.\n.\n#hashtag1 #hashtag2",
  "access_token": "TOKEN"
}

// Step 2: Publish
{
  "creation_id": "{CONTAINER_ID}",
  "access_token": "TOKEN"
}
```

- **Technical Requirements**:
  - Image must be publicly accessible URL
  - Caption max 2200 characters
  - 30 hashtag limit
  - Posting hashtags as first comment requires separate API call
- **n8n Nodes**:
  - HTTP request (create media container)
  - Wait 1 second (API requirement)
  - HTTP request (publish container)
  - HTTP request (post hashtags as comment - optional)
  - Parse response (get post ID)

Node 5.3c: Publish to Facebook

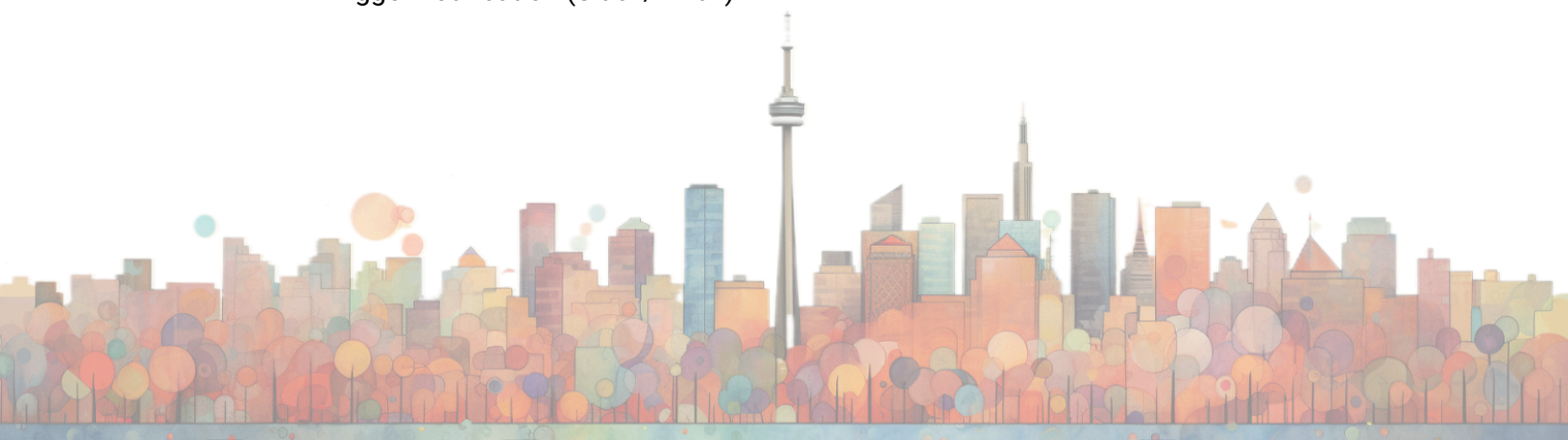
- **System Action**: Post to Facebook Page via Graph API
- **API Endpoint**: `POST /{page-id}/photos` (with image) or `POST /{page-id}/feed` (link post)
- **Authentication**: Page access token
- **Request Format**:

```
JSON
{
  "url": "https://storage.../image.png",
  "message": "Post text with hashtags",
  "access_token": "PAGE_ACCESS_TOKEN"
}
```

- **Technical Requirements**:
  - Page-level access token (not user token)
  - Image URL must be publicly accessible
- **n8n Nodes**:
  - HTTP request (post to page)
  - Parse response (get post ID)

Node 5.4: Handle Publishing Response

- **System Action**: Process API response and update database
- **Success Path**:
  - Extract platform post ID
  - Update database:
    - status: queued → published
    - published_at: current timestamp
    - platform_post_id: returned ID
- **Error Path**:
  - Log error details
  - Update status: queued → failed
  - Trigger notification (Slack/Email)



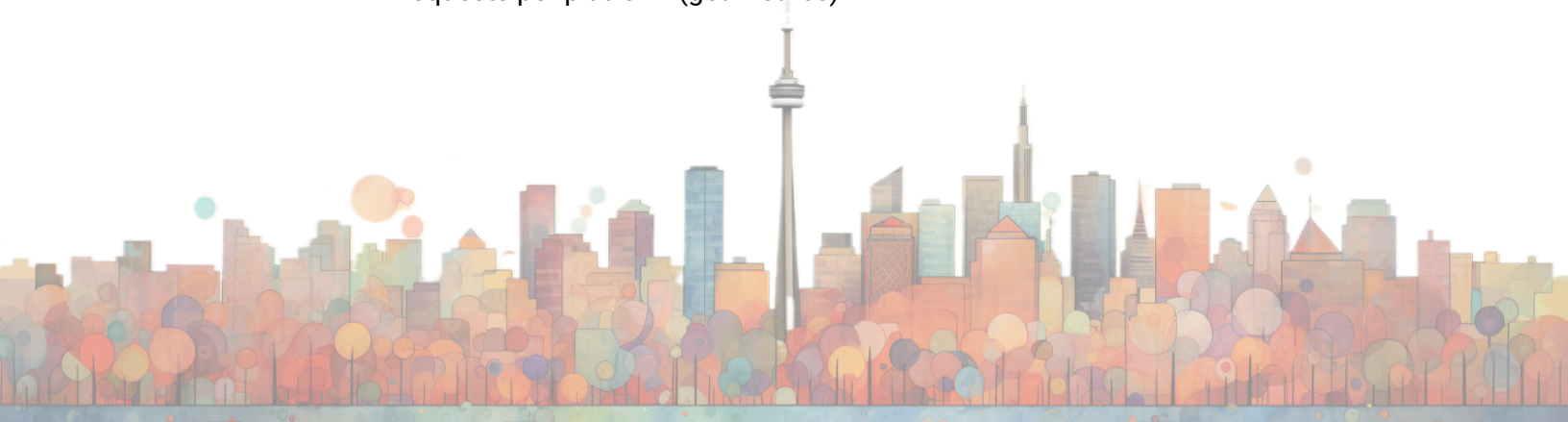
        - Implement retry logic (max 3 attempts)
- **Technical Requirements**: Error handling, retry logic
- **API Requirements**:
    - Database update
    - Notification API (Slack webhook or email)
- **n8n Nodes**:
    - IF node (check for API errors)
    - Success branch:
        - Database update node
    - Error branch:
        - Database update (status: failed)
        - Increment retry_count
        - IF retry_count < 3:
            - Wait node (exponential backoff)
            - Loop back to Node 5.2
        - ELSE:
            - Send notification (Slack/Email)
            - Mark for manual intervention

STAGE 6: Analytics & Performance Tracking

Data Collection → Analysis → Insights Dashboard

Node 6.1: Collect Engagement Metrics

- **Trigger**: Daily at 11 PM ET (after full day of engagement)
- **System Action**: Fetch metrics for all recently published posts
- **Technical Requirements**: Multi-platform API integration
- **API Requirements**: Platform analytics APIs
- **n8n Nodes**:
    - Schedule trigger (daily 11 PM)
    - Database query (get posts published in last 30 days)
    - Loop through posts by platform
    - HTTP requests per platform (get metrics)



- Parse metrics
- Database update (store engagement data)

Node 6.1a: LinkedIn Metrics

- **API Endpoint**: `GET /organizationalEntityShareStatistics`
- **Metrics Collected**:

```
JSON
{
  "impressions": 1250,
  "clicks": 45,
  "likes": 28,
  "comments": 5,
  "shares": 3,
  "engagement_rate": 2.88,
  "follower_count": 1500
}
```

- **Calculation**: `engagement_rate = (likes + comments + shares) / impressions * 100`

Node 6.1b: Instagram Metrics

- **API Endpoint**: `GET /{media-id}/insights`
- **Metrics Collected**:

```
JSON
{
  "impressions": 2400,
```

```
    "reach": 1800,
    "likes": 156,
    "comments": 12,
    "saves": 23,
    "shares": 8,
    "engagement_rate": 8.29,
    "profile_visits": 15
}
```

- **Calculation**: `engagement_rate = (likes + comments + saves + shares) / reach * 100`

Node 6.1c: Facebook Metrics

- **API Endpoint**: `GET /{post-id}/insights`
- **Metrics Collected**:

JSON

```
{
  "impressions": 1800,
  "reach": 1400,
  "reactions": 67,
  "comments": 8,
  "shares": 5,
  "engagement_rate": 5.71,
  "link_clicks": 22
}
```

Node 6.2: Store Metrics in Database

- **System Action**: Update content records with engagement data
- **Technical Requirements**: JSON storage in database
- **API Requirements**: Database update
- **n8n Nodes**:
    - Combine metrics into JSON object
    - Database update (engagement_metrics field)
    - Calculate derived metrics (ROI, cost per engagement if ads)

Updated Data Schema:

JSON

```
{
  "content_id": "uuid",
  "engagement_metrics": {
    "impressions": 1250,
    "likes": 28,
    "comments": 5,
    "shares": 3,
    "engagement_rate": 2.88,
    "collected_at": "timestamp"
  }
}
```

Node 6.3: Performance Analysis (AI-Powered)

- **Trigger**: Monthly on 1st at 9 AM
- **System Action**: Analyze performance trends and generate insights
- **Input Data**: Last 30 days of engagement metrics
- **AI Analysis Prompt**:

None

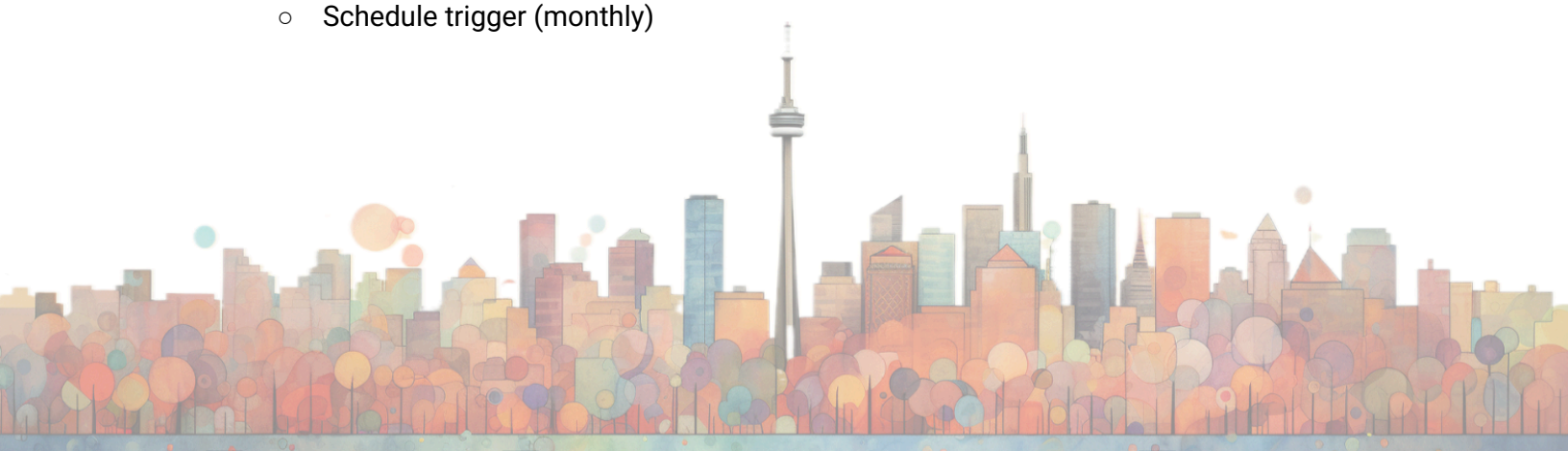
```
You are analyzing social media performance for NeuroVision Therapy Clinic.

Data: [JSON array of posts with engagement metrics]

Analyze and provide:
1. Top 5 best-performing posts (by engagement rate)
   - What topics performed best?
   - What content pillars had highest engagement?
2. Platform comparison
   - Which platform has best engagement?
   - Best posting times per platform
3. Content type analysis
   - Do images with faces perform better?
   - Do questions drive more comments?
4. Trends over time
   - Is engagement growing or declining?
   - Are we reaching more people?
5. Recommendations
   - What content types to increase/decrease
   - Optimal posting frequency
   - Topics to focus on next month

Format as structured report with specific data points.
```

- **Technical Requirements**: Data aggregation, AI analysis
- **API Requirements**:
  - Database query
  - OpenAI/Claude API
- **n8n Nodes**:
  - Schedule trigger (monthly)



- Database query (aggregate last 30 days)
- Code node (format data for AI)
- HTTP request (AI analysis)
- Parse response
- Store insights report in database

Node 6.4: Generate Analytics Dashboard

- **System Action**: Display visual analytics in admin UI
- **UI Components** (Bolt/Lovable):
  - **KPI Cards**:
    - Total reach (last 30 days)
    - Average engagement rate
    - Follower growth
    - Top performing platform
  - **Line Chart**: Engagement rate over time (30/90 days selectable)
  - **Bar Chart**: Performance by content pillar
  - **Heatmap**: Best posting times by platform and day
  - **Data Table**: Top 10 posts with metrics
  - **AI Insights Panel**: Display monthly analysis report
- **Technical Requirements**:
  - Chart library (Chart.js, Recharts, etc.)
  - Real-time data fetching
  - Export functionality (PDF)
- **API Requirements**: Database queries
- **n8n Nodes**: N/A (UI display with API calls)

API Endpoints for Dashboard:

```javascript
  - GET /api/analytics/summary?period=30
  - GET /api/analytics/engagement-trend?period=90
```

```
        - GET /api/analytics/by-pillar?period=30
        - GET /api/analytics/top-posts?limit=10
        - GET /api/analytics/insights/latest
```

Node 6.5: Strategy Refinement Loop

- **Trigger**: Manual (i.e. monthly review meeting)
- **User Action**: Review insights and adjust strategy
- **System Action**: Update strategy configuration
- **Technical Requirements**: Form with pre-filled current values
- **API Requirements**: Database update
- **n8n Nodes**:
    - Same as Node 1.2 (strategy update)
    - This creates feedback loop back to content generation

Feedback Loop:

```
None
        - Node 6.5 (Adjust Strategy) → Node 1.2 (Update Config) →
        - Node 2.2 (Generate Content with new strategy) → ... →
        - Node 6.3 (Analyze Performance) → Node 6.5 (Adjust Again)
```

NeuroVision Social Media Automation

Service Blueprint for Content Creation, Scheduling Analytics

Legend Technical Requirements

User Interface | AI Processing | Decision | Data Storage | Publishing

Core APIs Required:

- Content Generation: OpenAI GPT-4 or Claude
- Image Generation: Midjourney API or DALL-E 3
- Social Publishing: LinkedIn Marketing API, Instagram Graph API, Facebook Graph API

Infrastructure:

- Admin UI: Bolt.new or Lovable.dev
- Automation: n8n workflows
- Storage: PostgreSQL or MongoDB