**Organise Your Local File Directories With AI.txt**

**Workflow: Obsidian Notes Read Aloud: Available as a Podcast Feed**

**Overview**

This workflow is designed to convert Obsidian notes into an audio podcast feed. It enables you to send notes from Obsidian via a webhook, where they are processed to generate audio files (using OpenAI's TTS capabilities) and then made available as part of a podcast feed. The audio files are uploaded to Cloudinary and the resulting RSS feed is generated for podcast distribution.

**Nodes and Their Functions**

1. **OpenAI1**

* **Type:** OpenAI Node (@n8n/n8n-nodes-langchain.openAi)
* **Purpose:** Converts the content of an Obsidian note into an audio file using OpenAI's text-to-speech (TTS) API.
* **Parameters:**
  + **Input:** The note's content (retrieved from the webhook payload).
  + **Options:** Specifies that the response should be an MP3 audio file.
  + **Resource:** Audio generation.
* **Credentials:** Uses the configured OpenAi account.

2. **Sticky Note**

* **Type:** Sticky Note Node (n8n-nodes-base.stickyNote)
* **Purpose:** Provides setup instructions for sending notes to the webhook.
* **Content:** Instructions for installing the Post Webhook Plugin in Obsidian and details on how to send a note or selection to the assistant.

3. **Sticky Note1**

* **Type:** Sticky Note Node
* **Purpose:** Provides information about the generic podcast feed module.
* **Content:** Describes how to generate a standard RSS feed from source data (such as Google Sheets) and includes details about podcast metadata. This module is reusable across workflows.

4. **Sticky Note2**

* **Type:** Sticky Note Node
* **Purpose:** Describes the process of creating audio from notes.
* **Content:** Explains that OpenAI TTS converts the note to audio while a messaging model generates concise descriptions for podcast apps.

5. **Sticky Note3**

* **Type:** Sticky Note Node
* **Purpose:** Provides guidance for appending essential podcast parameters to Google Sheets for feed generation.
* **Content:** Instructions on saving details like title, link, description, and duration in a Google Sheet.

6. **Webhook GET Note**

* **Type:** Webhook Node (n8n-nodes-base.webhook)
* **Purpose:** Receives notes from Obsidian through a POST request.
* **Configuration:**
  + **Path:** Unique webhook path.
  + **HTTP Method:** POST.
  + **Response:** Returns a response to confirm receipt.

7. **Webhook GET Podcast Feed**

* **Type:** Webhook Node
* **Purpose:** Serves the generated podcast RSS feed.
* **Configuration:**
  + **Path:** Unique webhook path for accessing the feed.
  + **Response Mode:** Returns the RSS XML as plain text.

8. **Upload Audio to Cloudinary**

* **Type:** HTTP Request Node
* **Purpose:** Uploads the generated audio file (MP3) to Cloudinary.
* **Configuration:**
  + **URL:** Cloudinary upload endpoint.
  + **Method:** POST (using multipart/form-data).
  + **Body Parameters:** Include the file, upload preset, and resource type.
  + **Authentication:** Configured using Cloudinary credentials.

9. **Give Audio Unique Name**

* **Type:** Set Node
* **Purpose:** Assigns a unique filename to the generated audio file, using a timestamp.
* **Output:** The unique filename is attached to the data.

10. **Send Audio to Obsidian**

* **Type:** Respond to Webhook Node
* **Purpose:** Sends the generated audio file back to the source (Obsidian) as a response to the webhook.
* **Configuration:**
  + **Response:** The audio file in binary format.
  + **Response Headers:** Sets the content type to audio/mpeg.

11. **OpenAI (for Transcription and Summary)**

* **Type:** OpenAI Node (n8n-nodes-base.openAi)
* **Purpose:** Generates a concise description for the audio or note content.
* **Configuration:**
  + **Model:** Uses GPT-4O-MINI.
  + **System Prompt:** Instructs the model to generate a description of 50–150 characters, focusing on key takeaways.
* **Credentials:** Uses the configured OpenAi account.

12. **Merge**

* **Type:** Merge Node
* **Purpose:** Combines outputs from various nodes to prepare the final data for further processing.
* **Configuration:** Default merge behavior.

13. **Append Item to Google Sheet**

* **Type:** Google Sheets Node
* **Purpose:** Appends podcast parameters (title, link, description, date, duration) to a specified Google Sheets document.
* **Configuration:**
  + **Sheet and Document Details:** Set to the appropriate Google Sheets document for the podcast feed.
* **Credentials:** Uses Google Sheets API credentials.

14. **Return Podcast Feed to Webhook**

* **Type:** Respond to Webhook Node
* **Purpose:** Returns the final RSS feed as an XML response.
* **Configuration:**
  + **Response Headers:** Sets the content type to application/xml.
  + **Response Body:** Contains the generated RSS feed XML.

15. **Manually Enter Other Data for Podcast Feed**

* **Type:** Set Node
* **Purpose:** Provides additional metadata for the podcast feed (e.g., base URL, podcast title, description, author info, etc.).
* **Configuration:**
  + **Fields:** Include baseUrl, podcastTitle, podcastDescription, authorName, ownerName, ownerEmail, coverImageUrl, language, explicitContent, and itunesCategory.

16. **Write RSS Feed**

* **Type:** Code Node
* **Purpose:** Generates an RSS feed XML from the data provided.
* **Functionality:**
  + Formats the podcast metadata and episode items into an XML structure following the RSS feed specification.
  + Each podcast episode includes title, description, link, publication date, duration, and other necessary tags.
* **Output:** The complete RSS XML feed.

**Workflow Execution Flow**

1. **Trigger:** The workflow starts when a note is sent via the **Webhook GET Note** node from Obsidian.
2. **Audio Generation:** The note's content is sent to **OpenAI1**, which converts the text into an MP3 audio file.
3. **File Upload:** The generated audio is then uploaded to Cloudinary using the **Upload Audio to Cloudinary** node.
4. **Filename Assignment:** The **Give Audio Unique Name** node assigns a unique name to the audio file based on a timestamp.
5. **Response Delivery:** The audio file is sent back to Obsidian via the **Send Audio to Obsidian** node.
6. **Additional Processes:** Optionally, the workflow also triggers the generation of a podcast RSS feed and logs podcast episode details to Google Sheets.
7. **Final Output:** The **Webhook GET Podcast Feed** node allows retrieval of the complete podcast RSS feed.

**Setup Instructions**

1. **Install Post Webhook Plugin in Obsidian:** Follow the provided link in the **Sticky Note** node for instructions.
2. **Configure API Credentials:** Ensure that the OpenAI, Cloudinary, and Google Sheets credentials are correctly set in n8n.
3. **Webhook URLs:** Use the webhook URLs provided by n8n for receiving notes and for retrieving the podcast feed.
4. **Testing:** Manually trigger the workflow to test the audio generation, upload, and RSS feed creation processes.
5. **Customization:** Adjust the system prompts and metadata in the **Manually Enter Other Data for Podcast Feed** node as needed.
6. **Activation:** Once tested, activate the workflow to start processing notes from Obsidian automatically.

**Troubleshooting**

* **API Errors:** Check the API credentials and endpoints for OpenAI, Cloudinary, and Google Sheets.
* **File Processing:** Ensure that the local file system and Docker volume (if used) have appropriate permissions.
* **Webhook Issues:** Verify that the webhook URLs are correctly configured and accessible from Obsidian.
* **RSS Feed Validation:** Use an RSS feed validator to ensure that the generated XML meets all specifications.