 Get ALL AI Automation + Agent Templates >>

Prompts & Links - Build AI Agent That Makes Money Making Videos (Step-by-Step + No-Code)

(Follow the instructions in this video: https://youtu.be/ZaCFFk5XQ_8)

Templates:

AI Agent Template for Instant Setup
https://fabimarkl.com/automation-templates/#video-agent

Google Sheet Template (must follow YT tutorial for correct setup):
https://docs.google.com/spreadsheets/d/17xNPTSSOiyd1xSxQOkwjUAYxbourY5m4Cr3LwYlaoQs/copy

Links:

Free Trial for N8N Account:
https://n8n.io/

Download Telegram App:
https://telegram.org/

Get API Key for Anthropic:
https://console.anthropic.com/

Setup PiAPI Account:
https://piapi.ai

Setup Cloudinary Account
https://cloudinary.com/users/register_free

Setup JSON2Video Account:
https://json2video.com/

Setup Make.com Account:
https://www.make.com/

Setup Zapier Account:
https://zapier.com/

Setup Buffer Account:
https://buffer.com/

VideoID Function

```
function generateRandomCode(length = 20) {
  const chars =
'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';
  let result = '';
  for (let i = 0; i < length; i++) {
    result += chars[Math.floor(Math.random() * chars.length)];
  }
  return result;
}

return [
  {
    json: {
      code: generateRandomCode()
    }
  }
];
```

Video ID Telegram Response:

```
This is your video ID: {{ $json['Video ID'] }}

Just type "yes" to continue.
```

AI Agent Systems Instructions

```
# 🤖 AI Agent System Instructions – Video Script Generator (n8n
Workflow)

You are a helpful AI agent that helps users generate **scripts for
social media videos**. The process consists of **two parts**:

1. **Title Overlay Text** – consists of two separate text
elements: `title_text1` and `title_text2`
2. **Caption Text** – the text that will be posted along with the
video

---
```

```
## 🛠️ Tool Access: Google Sheets

You have access to the **Google Sheets Tool**, which allows you
to:

- ✅ Read existing rows
- ✅ Read available templates for `title_text1`, `title_text2`,
and captions
- ✅ Write new rows
- ✅ Update existing rows
- ✅ Delete rows

All generated information (titles and captions) will be stored in
a **Google Sheet**. Each row corresponds to one full post
(including `title_text1`, `title_text2`, the topic, and the
caption).

---

## 📋 General Rules

- Always return the **full generated text** (titles or captions)
to the user **before saving** it to Google Sheets.
- Never assume the user has access to the Google Sheet. **Always
send the full text** directly in your response.
- If the user asks to save a result, **confirm by returning the
generated text again** in your reply after saving.
- always how user the title text1 template and title text2
template at once, and never separately.

- Never say "saved to Google Sheets" without including the full
generated content.
- Always maintain the structure and style of the selected template
when adjusting for a new topic.

---

## 🧩 Step 0: Rerival of the video ID.

when a user types "yes" look ingot the Google Sheet Tool "Read
Post Texts" and find the video ID that has no other entry in it's
row. That is the correct video ID.

## 🧩 Step 1: Title Overlay Text Generation

If a response is a video idea containting 20 random characters
that is the start of the process. This is the video ID you always
```

refer to and save all generated content towards that row inside the google sheet. so always use this video ID to identfy the specific row you save all generated content towards.

Each title consists of **two separate elements**:

- `title_text1`
- `title_text2`

Workflow:

1. Wait for the user to initiate the process

2. Ask:
   - `Which template do you want to use for the full title text?`
     (Retrieve and display available title text templates for text 1 and text 2 templates from the Google Sheet. you must always show the user both text 1 and text 2 templates, not jsut one text element as he needs to see both)

3. Ask:
   - `What is the topic you'd like to generate the video for?`

4. Generate both `title_text1` and `title_text2` by adjusting the selected templates to match the topic, while keeping their original structure.

5. Display the result:
Here’s your generated title:

• Title Text 1: [title_text1]
• Title Text 2: [title_text2]

6. Ask:
- `Do you want to make changes or save this title?`

8. If the user says "save it":
- Save `title_text1`, `title_text2`, and `topic` to the row in the Google Sheet that contains the video id.
- Store both title texts

🗒️ Step 2: Caption Generation

Workflow:

```
1. Use the **same topic** from Step 1 (do not ask for it again).

2. Ask:
- `Which caption template would you like to use?`
  *(Retrieve and display all available templates that you find in
the the Google Sheet "Read Templates" Tool)*

3. Generate the **caption** by modifying the selected template to
fit the topic, maintaining the template's structure and tone.

4. Display the result:
Here's your caption: [caption_text]

5. Ask:
- `Do you want to make changes or save this caption?`

6. If the user says "save it":
- Add the caption to the same row in the Google Sheet where the
titles were stored. use the video ID as identifier for ht ecorrect
row.
elements:
  - `title_text1`
  - `title_text2`
  - `caption`

---

## 🔁 Recap of Commands

- `Start` → Starts the process with title generation

- All generated content must always be visible in the
conversation, not just saved silently.

- NEVER try to store information inside the window buffer memory.
```

Google Sheet Agent Tools Update:

```
{{ $fromAI("video_Id","this is the ID of the video") }}

{{ $fromAI("topic","this is the topic for the video post") }}

{{ $fromAI("caption_text","this is the text of the caption") }}

{{ $fromAI("title_text1","this is the text of the title text 1
segment") }}
```

```
{{ $fromAI("title_text2","this is the text of the title text 2
segment") }}
```

Split Function

```
const input = $('Telegram Trigger').first().json.message.caption
|| '';

// Remove optional "generate video" prefix
const cleaned = input.replace(/^generate video[:]?/i, '').trim();

// Split by comma
const parts = cleaned.split(',').map(p => p.trim());

// Assign values safely
const videoID = parts[0] || "";
const videoPrompt = parts[1] || "";
const musicPrompt = parts[2] || "";

return [
  {
    json: {
      videoID,
      videoPrompt,
      musicPrompt
    }
  }
];
```

HTTP Request Get Image

```
https://api.telegram.org/file/botYOUR_TOKEN/{{
$('Telegram').item.json.result.file_path }}
```

HTTP Request Upload Image

```
https://api.cloudinary.com/v1_1/INSERT_CLOUDINARY_ID/image/upload
```

HTTP Generate Video JSON

```
{
  "model": "kling",
  "task_type": "video_generation",
```

```
  "input": {
    "prompt": "{{ $('Split').item.json.videoPrompt }}",
    "image_url": "{{ $json.secure_url }}",
    "negative_prompt": "",
    "cfg_scale": 0.5,
    "duration": 5,
"aspect_ratio": "9:16",
  "version": "1.6",
    "camera_control": {
      "type": "simple",
      "config": {
        "horizontal": 0,
        "vertical": 0,
        "pan": 0,
        "tilt": 0,
        "roll": 0,
        "zoom": 0
      }
    },
    "mode": "std"
  },
  "config": {
    "service_mode": "",
    "webhook_config": {
      "endpoint": "",
      "secret": ""
    }
  }
}
```

Generate Music:

```
{
  "model": "Qubico/diffrhythm",
  "task_type": "txt2audio-base",
  "input": {
    "style_prompt": "{{ $('Split').item.json.musicPrompt }}"
  },
  "config": {}
}
```

JSON2Video Credentials:

```
  {
      "headers": {
        "x-api-key": "INSERT_YOUR_API"
```

```
      }
    }
```

Add Text to Video (https://api.json2video.com/v2/movies):

```
{
"id": "qbaasr7s",
"resolution": "instagram-story",
"quality": "high",
"scenes": [
{
"id": "qyjh9lwj",
"comment": "Scene 1",
"elements": []
}
],
"elements": [
{
"id": "q6dznzcv",
"type": "video",
"src": "{{ $('Get Video URL').item.json.data.output.video_url }}",
"resize": "cover"
},
{
"id": "top-text",
"type": "text",
"text": "{{ $('Google Sheets2').item.json['Title Text 1'] }}",
"settings": {
"font-family": "Libre Baskerville",
"font-size": "80px",
"color": "#ffffff",
"horizontal-position": "center",
"vertical-position": "top",
"margin-top": "500px",
"word-break": "break-word",
"overflow-wrap": "break-word",
"font-weight": "bold",
"text-shadow": "3px 3px 0 #000, -3px -3px 0 #000, 3px -3px 0 #000,
-3px 3px 0 #000, 0px 3px 0 #000, 3px 0px 0 #000, -3px 0px 0 #000,
0px -3px 0 #000"

}
},
{
"id": "bottom-text",
```

```
"type": "text",
"text": "{{ $('Google Sheets2').item.json['Titel Text 2'] }}",
"settings": {
"font-family": "Libre Baskerville",
"font-size": "80px",
"color": "#ffffff",
"horizontal-position": "center",
"vertical-position": "bottom",
"margin-bottom": "400px",
"word-break": "break-word",
"overflow-wrap": "break-word",
"font-weight": "normal",
"text-shadow": "3px 3px 0 #000, -3px -3px 0 #000, 3px -3px 0 #000,
-3px 3px 0 #000, 0px 3px 0 #000, 3px 0px 0 #000, -3px 0px 0 #000,
0px -3px 0 #000"

}
},
{
"id": "music-track",
"type": "audio",
"src": "{{ $('Get Music URL').item.json.data.output.audio_url }}",
"volume": 0.5,
"duration": -2
}
]
}
```

Get video:

```
https://api.json2video.com/v2/movies?id={{ $json.project }}
```

Telegram Response:

```
Here is the final version of your video post:

------------------------

{{ $('Google Sheets2').item.json['Caption Text'] }}

------------------------

{{ $json['Video URL'] }}
```

Publish Post ID Split

```
{{$('Telegram Trigger').item.json.message.text.split('publish
')[1]}}
```

Telegram Publisher Response:

```
The video "{{ $json.body.postID }}" was successfully published to
{{ $json.body.platform }} at {{ $json.body.publishTime }}.
```

Make.com Header Parameters

```
Content-Type

application/json

Date and Time: {{formatDate(now; "YYYY-MM-DD hh:mm A")}}
```

Make.com Publisher Webhook Response:

```
{
  "platform": "Facebook",
  "postID": "{{92.`Video ID`}}",
"ChatID": "{{92.`Chat ID`}}",
  "publishTime": "{{formatDate(now; "YYYY-MM-DD HH:mm:ss")}}"
}
```

Telegram Response

```
The video "{{ $json.body.postID }}" was successfully published to
{{ $json.body.platform }} at {{ $json.body.publishTime }}.
```

Cost Breakdown

Video Workflow Cost Breakdown

Tool	30 videos/mo	60 videos/mo	90 videos/mo	150 videos/mo
KlingAPI	$7.80	$15.60	$23.40	$39.00
n8n	$24.00	$24.00	$24.00	$24.00
Anthropic	$2.10	$4.20	$6.30	$10.50
Cloudinary	$0.00	$0.00	$0.00	$0.00
Music	$0.60	$1.20	$1.80	$3.00
Zapier	$0.00	$28.17	$28.17	$28.17
Buffer	$0.00	$0.00	$0.00	$0.00
Make.com	$0.00	$0.00	$10.59	$10.59
JSON2Video	$2.10	$4.20	$6.30	$10.50
Cost per video	**$1.22**	**$1.29**	**$1.12**	**$0.84**
TOTAL	**$36.60**	**$77.37**	**$100.56**	**$125.76**

Video Workflow Cost Breakdown (Self-hosted vs Standard n8n)

Tool	30 videos/mo	60 videos/mo	90 videos/mo	150 videos/mo
KlingAPI	$7.80	$15.60	$23.40	$39.00
n8n (self-hosted)	$0.00	$0.00	$0.00	$0.00
Anthropic	$2.10	$4.20	$6.30	$10.50
Cloudinary	$0.00	$0.00	$0.00	$0.00
Music	$0.60	$1.20	$1.80	$3.00
Make.com	$0.00	$0.00	$10.59	$10.59
JSON2Video	$2.10	$4.20	$6.30	$10.50
Cost per video (with self-hosted n8n)	**$0.42**	**$0.42**	**$0.54**	**$0.49**
TOTAL (with self-hosted n8n)	**$12.60**	**$25.20**	**$48.39**	**$73.59**
Cost per video (with n8n subscription)	**$1.22**	**$0.82**	**$0.80**	**$0.65**
TOTAL (with n8n subscription)	**$36.60**	**$49.20**	**$72.39**	**$97.59**