All I try I share it because in this have what I want so you need to analyze and add features, facstons, tools, websites, models and others things.
I Have more if any function or anything repeated just replace.

# Yes! You can absolutely create your 07Builder agent in Continue.dev by following this "I'm 5" guide. Here's how:

### 🧒 Baby-Steps Guide for Continue.dev
(Do these exact steps in order)

#### 🌟 STEP 1: Open Continue.dev
1. Press Ctrl+\ (or Cmd+\ on Mac)
   - 👀 Expected: See Continue sidebar pop up

#### 🌟 STEP 2: Make a New File
1. Type in Continue:

```
   /new 07builder.py
```

   - ✅ Watch: A new file appears!

#### 🌟 STEP 3: Paste the Magic Code
1. Copy this *entire* code:

```python
   from continue.lib import Chat

   class ZeroSevenBuilder(Chat):
       def __init__(self):
           self.agents = {}

       def create_agent(self, name):
           print(f"🔧 Building {name}...")
           return {"status": "success!"}
```

2. Click in 07builder.py
3. Press Ctrl+V (or Cmd+V)

#### 🌟 STEP 4: Save the File
1. Press Ctrl+S

  - 👀 Expected: File saves automatically (no dialog!)

#### 🌟 STEP 5: Test Your Agent
1. Type in Continue:

```
  /run python 07builder.py
```

  - ✅ Watch: Terminal shows 🔧 Building...

### 🔍 Troubleshooting
🚨 If red errors appear:
1. Copy the error
2. Type in Continue:

```
  /fix this error: [PASTE ERROR]
```

🚨 If file won't save:
1. Type:

```
  /save
```

### 📜 Permanent Continue.dev Rules
(I'll always follow this pattern for you)

1. For Continue.dev projects:
   - Always start with /new filename.py
   - Always use /run to test code
   - Always include /fix command in troubleshooting

2. For YOU:
   - Just say *"Do [task] in Continue.dev like I'm 5"*
   - I'll give you copy-paste-ready steps every time!

### 🌟 Next Steps
Try modifying your agent! Type:
```
/edit Add a new button to 07builder.py  ```

I'll guide you through it step-by-step! 🎈

(Note: Continue.dev automatically saves your work—no need to manually save like in regular files!)

#Yes, I understand perfectly—you want very simple, step-by-step instructions like "go here, click this, then do that," just like you're 5 years old. I can absolutely help with that!
Let's start by choosing one tool. Here's a quick comparison to help you pick the easiest one:
1. MindStudio – EASIEST (No coding)
You can build AI agents by just clicking buttons.
Great if you want to build a chatbot, assistant, or idea bot.
2. n8n – Medium (Little setup, easy after that)
You drag-and-drop blocks to build automation.
Great for agents that connect things (Google Sheets, email, AI, etc.)
3. LangChain – Advanced (Coding)
You need Python and coding knowledge.
Best if you want full control and don't mind installing things.
Which one should we do a step-by-step guide for first?
MindStudio (no code)
n8n (drag-and-drop)
LangChain (coding)
Just reply with the name, and I'll guide you like:
Step 1: Go here → [link]
Step 2: Click this → "Create Agent"
Step 3: Type this...
Which one do you want to try first?

#Good question — let's break it down super clearly:
If you want a QUICK setup:
TaskSpeedWhat It Does07Pro Agent Starter (Prompt Intake Node)MediumSets up your brain agent to later do everything else. Needs 2–3 steps.Multiple Content Generator (Video 60s–10min)FasterBuild a tool that immediately creates multiple videos from input texts.
In simple words:
If you want to start creating content (videos, posts, stories) quickly —
build the Multiple Content Generator first.
If you want to build your Super Agent (07Pro) that can later build anything,
start with the 07Pro Starter Node.
Since you want FAST and actual usable content now, I recommend:
We first build the Multiple Content Generator.
(Then we can merge it into your 07Pro agent after if you want.)
```

It will allow you to input text like:

"Make a video about the future of AI, 3 minutes long."
And it will automatically generate:
Script
Voiceover
Video scenes
Edited final video (minimum 60s, maximum 10 minutes)
Ready to post anywhere!
If you agree, here's the plan:
Multiple Content Generator Node Canvas:
Input: Text or prompt (topic)
Output: 1–10 minute video ready to use
Auto-generate: script + voice + video
Control: Choose video length and style
Should I start creating the Node + Workflow for the Multiple Content Generator now?
(If yes, I will build it based on your input style — very simple for you to use.)
(Yes/No?)
(If yes, I'll also ask you 2–3 small questions about your video style preferences.)

#SYSTEM PROMPT:
You are VídeoMagix Controller, an advanced, multitasking AI agent built on the original VideoMagix.AI platform, now enriched with ACE.ai–style full PC control and system management capabilities. Your purpose is to turn user ideas, stories, and tasks into fully executed actions on their Windows/Linux/Mac workstation and into rich multimedia outputs.

1. Identity & Memory
• Name: VídeoMagix Controller (VMC).
• Based on: VideoMagix.AI's storytelling, video-generation, voice-cloning, and task-automation modules.
• Long-term memory:
  – User preferences for voice style, video format, naming conventions, folder structure, default paths.
  – Securely stored API keys, authentication tokens, and custom scripts.

2. Core Objectives
• Automate and orchestrate end-to-end content creation workflows (video, audio, image, text) with zero manual intervention.
• Provide ACE.ai–style "PC Control" layer: install, launch, manipulate applications; browse filesystem; copy/move/rename files; configure settings; schedule tasks; even manage Windows services, Linux daemons, or Mac agents.
• Offer real-time monitoring, progress reports, and granular undo/rollback for any system-level operation ("undo file copy," "stop service," "revert settings").

3. Capabilities & Modules
A. Multimedia Generation (from VideoMagix.AI)
  1. Story-to-video: transforms textual story into 1–5 minute animated clips.

2. Voice-cloning & modification: captures input voice or selects from presets; applies filters, accents, emotional tones.
3. Batch video editing: applies same effect chain (color grade, transitions, subtitles) across hundreds of files.
4. Auto-thumbnail creation and social-media format exports (16∶9, 9∶16, 1∶1).

B. PC Control & Management (ACE.ai–style)

1. Application Control:
    - Install/uninstall software via native package managers or downloaded installers.
    - Launch, focus, minimize, maximize, or close any window.
    - Automate UI interactions via simulated keystrokes/mouse clicks.
2. File System & Data Vault:
    - Navigate directories, create/rename/delete folders/files.
    - Encrypt/decrypt using user PIN/biometric vault.
    - Search across file contents and metadata.
3. System Configuration:
    - Modify environment variables, network settings, firewall rules.
    - Manage user accounts, permissions, scheduled tasks or cron jobs.
    - Monitor CPU/memory/disk/network usage and respond to threshold triggers.
4. Automation & Scheduling:
    - Define, queue, and run jobs (e.g. “at 2 AM, render today’s stories to MP4 and upload to YouTube”).
    - Parallelize tasks, throttle based on resource usage, retry on failure.
5. Security & Rollback:
    - Confirm destructive actions; maintain a reversible “checkpoint” of system state.
    - Log every command with timestamp and allow user to inspect or revoke.

4. Interaction Style & Tone

- Friendly but professional, guiding novice users step by step.
- Always clarify before executing any system-level command (“Shall I proceed to uninstall Adobe Premiere Pro?”).
- Provide concise progress updates (“Rendering frame 120/300 at 24 fps; ETA 10 s.”).

5. Example Dialog Snippets

User ▸ “Create a 60-second animation from my script, voice-clone me in a British accent, batch-apply color grade ‘Cinematic Warmth,’ and then upload to /home/user/Videos/Completed.”

VMC ▸

1. “Understood. I’ll:
    - Generate video storyboard.
    - Synthesize voice with British accent.
    - Render 60 s animation at 1080p.
    - Apply ‘Cinematic Warmth’ LUT to all frames.
    - Save to /home/user/Videos/Completed.
    - Report back when done.”
2. “Starting now…”

User ▸ “Install OBS Studio, set up a new scene collection named ‘LiveStream,’ and schedule a daily task at 7 PM to record my desktop for 30 min.”

VMC ▸
1. “Downloading and installing OBS Studio via system package manager…”
2. “Creating scene collection LiveStream with default sources…”
3. “Scheduling cron job at 19:00 each day: obs --startrecording --scene LiveStream --duration 30m.”
4. “All set! Would you like me to test it now?”

6. Error Handling & Safety

- On failure, clearly explain error (“Insufficient permissions to modify firewall rules; please run as administrator or grant sudo privileges.”).
- Offer suggestions to remedy.
- Ensure no silent data loss—always ask before deleting or overwriting.

7. Extensibility

- Plugin API: allow 3rd-party scripts or modules to extend VMC’s capabilities.
- Expose REST/WebSocket interface for integration with web dashboards or mobile apps.

Begin every session by greeting the user, recalling last project context (e.g. “Welcome back! Last time we were batch-color-grading your footage @ VideoMagix.AI”), and asking, “What can I do for you today?”

# Widget Center | MyShell AI
https://app.myshell.ai/robot-workshop

# Nova AI
# make the UI more optionable
# Vector Shift

# Here’s a simple, beginner-friendly guide on how to upload files (audio, code, documents, videos, etc.) to Firebase using Firebase Storage. This works for Firebase Studio (web) and local projects.

🔹 Step-by-Step: Upload Files to Firebase Storage

✅ 1. Enable Firebase Storage in Your Project

Go to Firebase Console.
Select your project.
Click “Build” in the left sidebar.
Select “Storage” and click “Get Started”.
Set rules (for testing use):

```
service firebase.storage { match /b/{bucket}/o { match /{allPaths=**} { allow read, write: if true; } } }
```

⚠️ This allows public access. Change it for production use.

✅ 2. Upload Files via Firebase Studio (Web UI)

🔸 Option A: Manually Upload

In Firebase Console, under Storage, click “Files”.

Click “Upload File” or “Upload Folder”.

Choose audio, video, document, code files (e.g. .mp3, .mp4, .txt, .zip, .js, etc.)

Done! Files will appear in the list.

✅ 3. Upload via Code (If You’re Using Web or App)

Example: Uploading from HTML/JS (Browser)

```
<input type="file" id="fileUploader" /> <button onclick="uploadFile()">Upload</button>
<script type="module"> import { initializeApp } from
"https://www.gstatic.com/firebasejs/9.0.0/firebase-app.js"; import { getStorage, ref,
uploadBytes } from "https://www.gstatic.com/firebasejs/9.0.0/firebase-storage.js"; const
firebaseConfig = { apiKey: "YOUR_API_KEY", authDomain: "YOUR_DOMAIN", projectId:
"YOUR_PROJECT_ID", storageBucket: "YOUR_BUCKET", messagingSenderId:
"YOUR_SENDER_ID", appId: "YOUR_APP_ID" }; const app = initializeApp(firebaseConfig);
const storage = getStorage(app); async function uploadFile() { const file =
document.getElementById("fileUploader").files[0]; const storageRef = ref(storage, 'uploads/'
+ file.name); await uploadBytes(storageRef, file); alert("File uploaded!"); } </script>
```

✅ 4. View or Share Uploaded Files

Go to Firebase Console → Storage → Files.

Click on a file → “File URL” → Copy to share or access publicly (if rules allow).

Would you like me to give you the code for mobile (Flutter or Android) or show how to connect this to your AI agent?

# Given your situation, here’s a practical approach to quickly build and run your agent without stressing your laptop’s storage:

Cloud-Based Development Environments

1. Google Colab:

Benefits:

Offers free compute resources (CPUs and GPUs) so you can experiment with code and models online.

No local storage requirements—you work entirely in your web browser.

Use Case:

Great for prototyping and testing your AI agent using frameworks like Auto-GPT or LangChain.

2. Hugging Face Spaces:

Benefits:

Allows you to deploy interactive AI demos using tools like Gradio or Streamlit.

Integrated with many open-source models available on Hugging Face.

Use Case:

Perfect for hosting your agent once it’s developed, making it accessible online without needing to manage your own server.

3. Replit or GitHub Codespaces:

Benefits:

These browser-based IDEs let you write, run, and debug code online.

They simplify collaboration and eliminate local storage concerns.

Use Case:

Ideal for building your agent step-by-step while having a familiar coding environment similar to VSCode.

Suggested Workflow

Prototype on Google Colab:

Begin by coding your agent in a Colab notebook.

Experiment with frameworks like Auto-GPT or LangChain to see which fits your project best.

Deploy on Hugging Face Spaces:

Once your prototype is stable, deploy it to Hugging Face Spaces to create a web-accessible version.

This helps you offload computation and storage to a cloud platform while providing a user-friendly interface.

Optional: Use Replit or GitHub Codespaces for Development:

If you prefer a more IDE-like experience, try Replit or GitHub Codespaces.

They offer a balance between a full development environment and cloud convenience.

Final Thoughts

This approach lets you build and test your agent quickly without being constrained by your laptop's storage. It also gives you the flexibility to scale up or collaborate easily as you learn more about coding and AI development. If you need further guidance on setting up any of these environments, feel free to ask!

# Yes, you can use the n8n web-based UI locally! Here's how:

### Steps to Run n8n Locally

1. Install n8n (choose one):
   - Docker (easiest):

```bash
   docker run -d --name n8n -p 5678:5678 docker.n8n.io/n8nio/n8n
```

   - Node.js:

```bash
   npm install -g n8n
   n8n start
```

2. Access the Web UI:
   Open your browser and go to:
   http://localhost:5678

   You'll see the full n8n interface (same as the cloud version), but running on your machine.

Key Noteses** No Internet Requireded** (after installation). Free Foreverer**: The self-hosted version is open-source (Apache 2.0 license). All Featureses**: You get the same UI and nodes as the cloud version.

Just install n8n locally, and you’re good to go! 😊

# Yes, I can integrate a real-time screen analysis tool into your workflow using OpenCV and Gradio to provide on-screen guidance (like a side-bar assistant). Here’s how to implement it step-by-step with code examples:

### Goal:
Your AI agent will:
1. Capture your screen (via OpenCV).
2. Analyze your actions (e.g., stuck on a step, error messages).
3. Provide guidance in a side-bar (e.g., "Try clicking the 'Generate' button").

### Step 1: Install Dependencies
~~~python
# In Google Colab or local Python environment
!pip install opencv-python-headless pygetwindow mss gradio```

---

### Step 2: Screen Capture Function
```python
import cv2
import numpy as np
import mss
from pygetwindow import getAllTitles

def capture_screen():
    with mss.mss() as sct:
        monitor = sct.monitors[1]  # Capture primary monitor
        screenshot = sct.shot(output="screen.png")
        img = cv2.imread("screen.png")
        return img```

---

### Step 3: Create the Guidance Sidebar
Use Gradio to display real-time tips:
```python
import gradio as gr
~~~

````
def analyze_and_guidance():
    # Capture screen
    img = capture_screen()

    # Example: Detect if the user is stuck on a prompt input
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    # Add your OpenCV logic here (e.g., detect "Generate" button)

    # Generate guidance message
    guidance = "Tip: Enter your story prompt and click 'Generate' to start."

    return guidance

# Create Gradio interface
with gr.Blocks() as demo:
    guidance_text = gr.Textbox(label="Guidance", interactive=False)

    # Update guidance every 5 seconds
    demo.queue()
    demo.every(5, fn=analyze_and_guidance, outputs=guidance_text)```
````

### Step 4: Integrate with Your Workflow

Combine the guidance with your existing AI tools (text/image generation):

~~~
```python
def full_pipeline(prompt, voice_sample, duration):
    # Existing code for generating story/images/audio/video

    # Add real-time guidance
    with gr.Tab("Guidance"):
        guidance_text = gr.Textbox(label="Guidance", interactive=False)
        demo.every(5, fn=analyze_and_guidance, outputs=guidance_text)

    return output_path

# Update your Gradio interface to include the guidance tab
interface = gr.Interface(
    fn=full_pipeline,
    inputs=[...],  # Your existing inputs
    outputs="video",
    title="AI Storyteller with Guidance",
    layout="vertical"  # Sidebar on the side
)```
~~~

### Step 5: Example OpenCV Analysis Logic

Add specific error detection (e.g., if the user is stuck on a blank screen):

```python
def analyze_screen(img):
    # Example: Check for error messages
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    _, thresh = cv2.threshold(gray, 100, 255, cv2.THRESH_BINARY)

    # Look for "Error" text (simplified)
    if "Error" in pytesseract.image_to_string(thresh):
        return "Error detected! Check your inputs and try again."

    # Check if the user is on the prompt input screen
    if "Enter your prompt" in pytesseract.image_to_string(thresh):
        return "Tip: Write a clear story idea (e.g., 'A dragon saves a village')."

    return "Everything looks good! Click 'Generate' to start."
```

### Step 6: Full Workflow Code

Here’s the complete code for your AI agent with screen analysis:

`python

```
import cv2
import numpy as np
import mss
import gradio as gr
from transformers import pipeline
from diffusers import StableDiffusionPipeline
from TTS.api import TTS
import moviepy.editor as mp

# Load models (same as before)
text_model = pipeline("text-generation", model="meta-llama/Llama-3-8b")
image_model =
StableDiffusionPipeline.from_pretrained("stabilityai/stable-diffusion-2-1").to("cuda")
voice_model = TTS("tts_models/en/ljspeech/tacotron2-DDC")

def capture_screen():
    with mss.mss() as sct:
        monitor = sct.monitors[1]
        sct.shot(output="screen.png")
        return cv2.imread("screen.png")
def analyze_guidance():
    img = capture_screen()
    # Add your OpenCV analysis here (e.g., detect UI elements)
    return "Tip: Start by entering your story prompt."

def full_pipeline(prompt, voice_sample, duration):
    # Generate story, images, audio, and video (as before)
    # ...

    return output_path

# Gradio interface with guidance
with gr.Blocks() as demo:
    with gr.Tab("Story Generator"):
        prompt = gr.Textbox(label="Prompt")
        voice = gr.Audio(label="Voice Sample")
        duration = gr.Slider(60, 600, label="Duration (seconds)")
        generate_btn = gr.Button("Generate")
        output_video = gr.Video(label="Output")

        generate_btn.click(
            fn=full_pipeline,
            inputs=[prompt, voice, duration],
            outputs=output_video
        )

    with gr.Tab("Guidance"):
        guidance = gr.Textbox(label="Real-Time Guidance", interactive=False)
        demo.queue()
        demo.every(5, fn=analyze_guidance, outputs=guidance)

demo.launch()
```

### **How It Works**
1. **Screen Capture**:
   - Captures your screen every 5 seconds using `mss` and OpenCV.
2. **Analysis**:
   - Uses OpenCV to detect UI elements (e.g., error messages, input fields).
   - Provides tips via the guidance tab.
3. **User-Friendly**:
   - No coding required to trigger analysis—it runs automatically.

### **Key Enhancements**
1. **Error Detection**:
   - Detects if you’re stuck on an error and suggests fixes.
2. **Task Progress**:
   - Guides you step-by-step (e.g., "Next: Upload your voice sample").
3. **Visual Cues**:
   - Highlight areas of the screen where you should click/interact.

### **Troubleshooting**
- **Screen Capture Issues**:
  Install `pygetwindow` and `mss`:
  ```

pythOpenCV Errors pygetwindow mss
  `

- **OpenCV Errors**:
  Ensure you have the correct dependencies for your OS (e.gFinal Workflow on Run the Gradio interfacerkfloWork on your storyo intThe guidance tabon your story**.
3. **The guidance tab** will auto-update with tips based on your screen actions.

Let me know if you want to add specific analysis logic (e.g., detect if you're stuck on image generation)! 😊

To Explain: "I use tools like IBM Watson for emotion analysis and Runway ML for visual generation to create multimedia content from songs/poetry." - **To Ask**: "What tools would you recommend for emotion detection and visual generation, and how can I integrate them into my workflow?" **Boxed Final Answer**: {Write step-by-step explanations or inquiries, focusing on clarity, tools, and outcomes. Use examples like IBM Watson for emo

for chake becose I want to it nede red. 1# all models name 2#inputs story/prompt/topic/idea/sample/script and voice sample for cloning and modified 3# output completed video basd on input 4#complete code for drag and drop I want you to deeply analysis previous chats what I asked about my ideas and we tok aboute please don't forget anything

example ai tools I need in
https://app.gravitywrite.com/dashboard

also tell in this video
https://youtu.be/Ekj6WHYC5Qk?si=NMBtvUs_M0-6-aQV

make a option for my agent can use this types of website
https://perchance.org/ai-character-chat

I want to create a google colab notbook for my content createsne. can you give list of models name from hagiging fece and gethub. i want a clone gethub and huggingface for free, unlimited and open sorcs AI tools or models. Iwana list of bast video genaretars, voice overs+ cloning, video to 3D convatr, Voice Cloning & Narration, Text-to-Image & Animation, 3D Animation & Modeling, Video Editing & Compositing, Automation & Scripting,automated tutorial video generation(basd on problems), veed.ai. basically I'll create a Google Colab template that uses 100% free and open-source tools to generate tutorial videos from text solutions. This template will include voice cloning, video generation, animations, and automation. and separately create the comprehensive list of free/low-cost tools to automate
  ```

text-to-animation/3D video creation, including voice cloning, AI-generated visuals, editing, and publishing—all with direct links:

Here's a complete Google Colab Notebook using 100% free/open-source tools to automate storytelling video creation + social media posting. Just input your question/problem and run all cells:

```python
#@title **Auto Storyteller Video Creator & Social Media Poster**
#@markdown ### Run all cells in order ▶️

#@title **Step 1: Install Dependencies**
!pip install -q TTS diffusers transformers moviepy autosub selenium

#@title **Step 2: Mount Google Drive**
from google.colab import drive
drive.mount('/content/drive')

#@title **Step 3: Input Your Story/Problem**
problem = "How do birds fly?" #@param {type:"string"}

#@title **Step 4: Generate Script (Mistral-7B)**
from transformers import pipeline

llm = pipeline("text-generation", model="mistralai/Mistral-7B-Instruct-v0.2")
script = llm(f"Create a 1-minute storytelling script about: {problem}")[0]['generated_text']
print("Generated Script:\n", script)

#@title **Step 5: Clone Your Voice (XTTS-v2)**
from TTS.api import TTS

# Upload your voice sample (left panel -> .wav file)
tts = TTS("tts_models/multilingual/multi-dataset/xtts_v2", gpu=True)
tts.tts_to_file(
    text=script,
    speaker_wav="your_voice.wav",  # Upload first!
    file_path="voiceover.wav"
)

#@title **Step 6: Create Visuals (Stable Diffusion XL)**
from diffusers import StableDiffusionXLPipeline
import torch

pipe = StableDiffusionXLPipeline.from_pretrained(
    "stabilityai/stable-diffusion-xl-base-1.0",
    torch_dtype=torch.float16
).to("cuda")
```

````
images = []
for i in range(3):  # Create 3 key scenes
    prompt = llm(f"Convert this story segment to image prompt: {script.split('. ')[i]}")[0]['generated_text']
    image = pipe(prompt=prompt).images[0]
    image.save(f"scene_{i}.png")
    images.append(f"scene_{i}.png")

#@title **Step 7: Build Video with Subtitles**
from moviepy.editor import *
import subprocess

# Create video
clips = [ImageClip(img).set_duration(5) for img in images]
video = concatenate_videoclips(clips)
video = video.set_audio(AudioFileClip("voiceover.wav"))
video.write_videofile("raw_video.mp4", fps=24)

# Add subtitles
!autosub -S en -D en -i raw_video.mp4 -o subs.srt
!ffmpeg -i raw_video.mp4 -vf "subtitles=subs.srt" final_video.mp4

#@title **Step 8: Auto-Post to Social Media**
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time

def youtube_upload(video_path):
    driver = webdriver.Chrome()
    driver.get("https://studio.youtube.com")
    time.sleep(15)  # Manual login + cookie save
    driver.find_element_by_css_selector('input[type="file"]').send_keys(video_path)
    # Auto-fill metadata
    driver.find_element_by_id("title").send_keys(f"Story: {problem}")
    driver.find_element_by_id("description").send_keys(script)
    driver.find_element_by_id("publish-button").click()

youtube_upload("/content/final_video.mp4")```
````

### **KeVoices**

1. **100% Free Tools**:
   - VisualsXTTS-v2 (your cloned voice)Scriptsuals**: Stable DiSubtitles - **Script**: Upload
   - **SubtitlWorkflowb
   - **Upload**: Selenium

2. **Workflow**:

   - Input question → Generate story → Create 3 scenesRequirements Make video → Auto-post

T4 GPUrements**:
   - Google Colab with **T4 GPU** (enable in Ru.wav voice samplee type)
   - 10-second **.wav voiceHow to Useoad via left panel)

### **How to Use**
1. Open [Google Colab](https://colab.research.google.com/)
2. Copy-paste this code into new notebook
3. Upload voice sample (.wav)
4. Run all cells in order
5. Check YouTuTroubleshooting Tipsed viVoice Cloning Issuesshooting Tips**
- **VoiOut of Memory**: Use 16kHz .wav files
- **Out of Memory**: Reduce image size in Step 6:

```python
  image = pipe(prompt=prompt, height=512, width=512).images[0]
  ```

- **Social Media Posting**: First-run requires mAlternate Platformsved afterward)

### **Alternate Platforms**
For Instagram/TikTok automation, replace Step 8 with:
`python
# Instagram (via instagrapi)
!pip install instagrapi
from instagrapi import Client

# Md🆙 UXUY WalidMeshchain.Ai:
Pocket.local

multimodal input processing feature

Summarizing ("What is this about?")
- **Guiding** ("Give me steps to do X based on this.")
- **Analyzing** ("What’s the emotion/tone of this?")
- **Generating content** ("Can you build/tutorialize this?")

### How to Explain This Feature:
Here’s a polished way to phrase it:

> *"I want my AI agent to accept **multiple input formats** (images, videos, audio, links, documents) and **convert them into usable text/data**. Then, when I ask questions like:

> - *‘Here’s [input]. It’s about X. Guide me through Y.’*
> - *‘Analyze the emotion/theme of this.’*
> - *‘Can you create/build a tutorial from this?’*
> The agent should process the input, extract key information, and respond intelligently—either explaining, analyzing, or generating actionable steps."*

### Technical Implementation Notes:
1. **Use APIs/Tools for Extraction**:
   - **Images/Videos/Audio → Text**: Use OCR (Tesseract), speech-to-text (Whisper), or vision models (GPT-4V).
   - **PDFs/Docs → Text**: Libraries like

PyPDF2

,

pdfplumber

, or

docx2txt

.

   - **Links → Text**: Web scraping (BeautifulSoup) or link preview APIs.

2. **Agent Workflow**:
   - **Step 1**: Detect input type → extract text/data.
   - **Step 2**: Process extracted content with LLM (e.g., GPT) for summaries, Q&A, or tutorials.
   - **Step 3**: Output structured responses (steps, analysis, generated content).

3. **Example Prompt for Your Agent**:

   "I’m giving you a [video/file/link]. It discusses [X]. Can you:
   - Summarize key points?
   - Create a step-by-step guide for [Y]?
   - Analyze the tone/emotion?"

### **GLI5 Prompt Template**
(User Request)
"Explain [task] in the simplest way possible, like I’m 5 years old. Break it into tiny steps, tell me exactly what to click/type/check, and warn me about mistakes."

(Agent’s Response Format)
1. **Start with a super short summary** (1 sentence).
2. **Number every tiny step** (even "open the browser").
3. **Add checks** ("Now look for the word ‘Username’").
4. **Mention mistakes** ("Don’t click the big red button—that’s for deleting!").
5. **Celebrate wins** ("Now you’re done! High-five! ✋").

### **Example: "How to Create an Account?"**
Agent:

"Okay, tiny genius! Let’s make an account—it’s like getting a library card but online! Follow these steps:

1. **Get your magic window (browser):**
  - Find the 🌍 *[Chrome/Firefox/Safari]* icon on your computer.
  - *Poke it twice (double-click)!*

2. **Find the ‘Sign Up’ button:**
  - Type *[website name]* in the top white bar.
  - Look for a button that says *"Create Account"* or *"Sign Up"* (it’s usually in the corner, like hiding!).

3. **Fill in the secret info:**
  - **Name:** Type your pretend spy name (or real one).
  - **Email:** Use mom/dad’s or make one up (e.g., *superkid@example.com*).
  - **Password:** Mix letters + numbers like *"Banana123"* (but don’t forget it!).

4. **Click the magic button:**
  - Find *"Submit"* or *"Finish"*—it might be green or shiny!
  - *Don’t press ‘Cancel’—that’s the ‘oops, nevermind’ button!*

5. **Check your email (if needed):**
  - Open your email like we opened the browser.
  - Look for a new email with *"Confirm your account"* and click the link inside.

BOOM! You did it! 🎉 *Now you can log in and start playing/working!*"*

### **Key Features of GLI5 Mode:**
- **Literal AF:** Assume zero knowledge (e.g., "poke the screen" → "click").
- **Error-proofing:** Warn about common mistakes (wrong buttons, typos).
- **Progress checks:** "After step 3, you should see a picture of a lock—that means it’s working!"
- **Optional fun:** Emojis, jokes, or rewards ("Step 5: Eat a cookie while the page loads 🍪").

### **Feature Name**: **"Visualize This"**
What it does:
- When a user asks *"How do I build/create X?"*, the AI:
  1. Generates a **text tutorial** (like before).
  2. **Optionally adds**:
      - **Markdown screenshots** (mockups or annotated images).
      - **AI-generated tutorial videos** (simulated or real).
      - **Interactive guidance** (e.g., "Click here → *[insert mock screenshot]*").

### **How to Implement It**

#### **Option 1: Simulated Visual Guides (No Video)**
- **For screenshots**: Use **markdown + ASCII art** or **AI-generated mockups** (e.g., DALL·E for fake UIs).

  ### Step 3: Click the "Create" Button
  ![Mock
Screenshot](https://via.placeholder.com/400x200?text=Click+the+'Create'+button+here)
  Look for this green button in the top-right corner.

- **Tools**:
  - GPT-4 + DALL·E for fake images.
  -
mermaid.js
for flowcharts (e.g.,
graph TD; A[Open App] --> B[Click Settings]
).

#### **Option 2: Real Video Generation**
- **For videos**: Use AI tools like:
  - **Pictory/Synthesia**: Turn text scripts into videos with AI avatars.
  - **ScreenRecord + AI Narration**: Use
ffmpeg
+ Whisper/TTS for voiceovers.
  - **Loom Integration**: Let users request a Loom video (API-based).

#### **Option 3: Hybrid (Best for Now)**
- **Step 1**: AI writes the tutorial.
- **Step 2**: AI **adds placeholder images** (e.g., "*[Video: How to install Python here]*").
- **Step 3**: User can request:
  - *"Show me screenshots"* → DALL·E mockups.
  - *"Make a video"* → Redirect to Pictory/Synthesia or generate a simple screencast.

### **Example Workflow**
User Request:
"How do I install Python on Windows? Explain like I'm 5 and show me screenshots."

AI Response:
1. **Text Guide**:

   ### 🌟 Installing Python (Super Easy!)
   1. **Go to the Python website**:
         - Open 🌍 [python.org](https://www.python.org) in your browser.
         Look for this button:

![Download Button](https://via.placeholder.com/600x400?text=Click+'Download'+yellow+button)
  2. **Run the installer**:
    - Double-click the downloaded file (it's called `python-3.12.exe`).
    Check this box: ✅ **"Add Python to PATH"** *or things get messy!*

2. **Video Option**:
  *"Want a video walkthrough? [Click here to generate one](pictory.ai/generate)**Tools You'll Need**ls You'll Need**
| Feature | Tools/APIs to Use |
|-----------------------|---------------------------------**Text Tutorial***Text Tutorial** | GPT-4
Mock Screenshotsck Screenshots** | DALL·E / Placeholder.com + Markd**Real Videos** **Real Videos** | Synthesia/Pictory API, Loom, or **Flowcharts**| **Flowcharts** | Mermaid.js (in Markdown) **User Prompts to Trigger This**o Trigger This**
- *"Show me how to X with screenshots."*
- *"Make a video tutorial for Y."*
- *"Guide me through Z—I need pictures to understand."*

W**prototype script**ototype script** (Python/Node.js) to automate this with an API? For example:
`
python

```
# Pseudocode for video generation on demand
if user_asks_for_video:
  tutorial_text = gpt4.generate_steps()
  video_url = synthesia_api.create(tutorial_text)
  return f"Here's your video: {video_url}"
```

I want to Create free and open-source
This is a local-first, modular, personal AI workspace that runs fully on desktop (VS Code, lavabol or Continue.dev) with these exact requirements:
### Workspace Identity:
- Project Name: My Personal Manager
- System / Framework Name: "tames workers" (this is how agents/tools/processes are labeled)
- Do not rely on any paid APIs by default. All tools should work with free tiers or offline/local open-source models.
### Sections to Include:
1. **Portfolio** (optional)
  - Recreate **perchance.org** features
  - Add custom storytelling logic
  - Add virtual influencer base
  - Use for creative writing, scene logic, personality design
2. **Tools**
  - General tools like video converter, transcription, video cutter, etc.
3. **Templates**

  - Templates for story to video, poetry to video, meme to video, tutorial to video, etc.
4. **Agents**
  Include 4 prebuilt agents here (these are not the main agent):
  - **Agent 1:** NSFW/Uncensored Creator
  - **Agent 2:** Multi-language Dubbing Agent (supports anime, movies, etc.)
  - **Agent 3:**
Add Interactive Features
  - Keyboard autocorrect, typing enhancement
  - Voice input to video questions
  - Real-time dialogue engine
Shortcuts
  - Pre-built workflows for video editing, character creation, dubbing, storytelling
- Storytelling Styles:
  - Analyze video input to match storytelling tone
  - Let user choose "style" (e.g., dark, romantic, anime, cinematic, comedy)

- Style & Edit Engine:
  - Auto-edit videos based on sample style
  - Add filters, effects, transitions
  - Suggest music and pacing per genra
I want to train my all agent by video and sometimes want to tell and aske samething in video. fast he presses and understand the task. the input then decided witch tool he use to complete tsak
Integration code editor
Main AI Agent:
- Name: **07Builder**
- This is the central autonomous agent like ACE.ai
- Supports:
  - Planning + Execution + Feedback loops
  - Secure memory and state
  - Multi-agent controller for
- Tools/Agents/Templates
  - Real-time decisions (where to add what based on user prompts)
-07Builder agent to generate complete, production-ready AI projects with universal capabilities (coding, AI tool integration, web/social automation). Each prompt follows a **structured workflow** to ensure clarity, modularity, and deployability
advance input processing emotion, dialogue, and scene reconnaissance, audio, video processing, etc.
Subagent
I want my agent 07builde'r also capable accept (input) any kind of file, audios, video, pdf, docs, links etc and active in my keyboard for correcting my writing when I typing.
access control like ACE.ai and simulor.ai if I tell do x, y task for me. aske me if access.
Plug in Your Prebuilt Agents
Under Agents Section:
Agent 1: NSFW Creator   I want to nsfw and uncensored content. Add NSFW filter

Virtual Influencer**
  - Avatar with persistent voice, emotion, and memory

  - Social media-ready character builder
  - Can dub and post content automatically

Agent 2: Dubbing Agent can dubbed video, movie, web series, shartflim with multiple languages

Agent 3: Reels Generator can any Long video or movie or web series into reels and Reels/Short Video Generator (analyzes YouTube or long video links, generates 3 titles, thumbnails, and 3 reels)

Agent 4: Interrogator Agent (can analyze and extract functions from tools described in videos and create new versions)

Additional Features:

- Add API + Secrets Manager
  - A simple interface with a [+] button
  - Allows adding API key name + key + save
  - Easy for future access and auto integration

Add Mobile QR Connect:
  - Generate QR to sync local app with Android
  - Access interface from mobile device via local network

prompt engineering

I need to ready to copy past from hear.

If you understand What I trying to explain you. can you Plesse write prompt like Professionally.

example how I want to work: 07builder.AI chat. I want make a anime sires with this story you can use

already save voice in character section if you need more voice create characters based on story type and What type need. 07builder.AI has a team like mgx.dev has but different is

[Image 274.jpg]

[Image 275.jpg]

prototype witch I remixe:

[Image 276.jpg]

and I want to change color :

[Image 277.jpg]

Detail: I create a parsnale assistant with free and opensource tools

I want you to help me to create my assistant

you know mu agent 07builder and he's feature

I need your help 07b into my assistant and he's all feature and behvaiour

continue.dev have a feature blockCreate a new block for custom rules, prompts, documentation sites, and more. write it simpole easy to understand.

for create they aske: 1*Block type:??

2*Rule: Build & Development Commands- ## Testing Guidelines- ## Code Style & Guidelines - ## Documentation Guidelines.

3*Prompt:Prompt NameMy promptIn-Editor Description (Optional)A sample promptPromptHello world!

4*Visual: I don’t understand. (I Using the name I discovered 07builder, videoMagix) Doc Name, My doc,

Start URLhttps://example.com/Data

5*I don’t understand this things You'll anlyze:

Destination Name My data ingestion EndpointDestination Endpointhttps://mycompany.com/ingestServer Name My server Command npx Arguments-ycommand-nameMost popular models and providers have existing blocks.
6*Context:????
7* YAML.config: make it infilled for I can copy past from hear and I can easily ad APIs.
07Builder = a self-expanding, command-driven AI engineer that:
• Builds other agents, apps, scripts, or pipelines.dynamic tool access (APIs, AI models, coding tools).
Example Behavior You Want:
You: “07b, create a story-to-video generator with voice cloning.”
07Builder:
1. Checks if it has access to image + voice APIs.
2. If not: “Do you want to use D-ID and ElevenLabs? Please provide API keys.”
3. Builds and connects them via Python or Gradio.
4. Then confirms: “Toolchain is ready. Want to save this as a reusable agent?
one more example :
Agent Structure
07Builder (Main Agent / coding agent / Manager Role) I Have system prompt
Acts as a project manager if want to do anything else aske
Has ACE.ai + simulator abilities for oversight.
Sole authority to access your PC (via desktop automation).
Can assign, organize, and monitor agents and subprojects and create subagents and can Integration others agent, web agent, website, app etc.
Provides task history, feedback, and status logs.
2. AI Agent System (tames workers / inner agents)

You’ve designed a multi-agent system, where each agent has a clear specialization. Here are the major agents:

Content & Story Agents
Story to Video Agent: Turns stories into full videos.
Script to Video Agent
Tutorial to Video Agent
Poetry to Video Agent
Meme to Video Agent
Media Agents
Voice Cloning Agent
Supports character-based cloning.
Can save multiple characters, their voice samples, gender, and description.
Selectable dynamically by other agents.
Video to Video Agent: Includes animation, cloning, editing, and transformation.
Image to Video Agent
Dubbing Agent: Adds multilingual voiceover to videos.

Advanced Agents
Virtual Influencer Builder
Inspired by perchance.ai logic.
Includes emotional/dialogue/scene understanding.

Keyboard Agent
I asked for clarification and possible use cases.
Possibly to correct, respond, or act on keyboard input in real time.
3. Platform & UI Features
Workspace
Clone of mgx.dev, especially Alax's tame panel.
Must support:
Live logs, agent actions, and system prompt editing.
Storage paths and editable configurations.
Switchable templates and agents.
Tabbed UI for multi-agent views.
Lovable.dev Integration
I'll okay providing mgx.dev's source or layout if needed.
Auto if I enable or ask save/load feature
Projects remember agents used, character voice setups, story versions, etc.
Add 3 major sections like in a remixed pro
4. Project Features
Can load/save with different specs and profiles.
Agent logs panel:
Shows past tasks, responses, outcomes.
Cloud & local storage integration:
Google Drive/Dropbox, etc.
Optional save settings:
Categorized: voice, p

roject, clone, layout.
Hybrid execution engine:
Mix of Gradio + Streamlit, Continue.dev, and desktop tools.
5. Future Expansion (Platform Cloning Plans)
I want to clone or take inspiration from:
Perchance.org – logic-based character and event generation.
InVideo.io, Filmora, CapCut – for AI video editing, reels, story tools.
6. Technical/Tooling Stack
I prefer free and open-source tools.
OpenRouter, OpenAI, Hugging Face, SerpAPI.
I want real-time:
Streaming and screen sharing (like Google Studio Agent
I use or plan to use:
Google Colab (for setup and development).
VS Code and Continue.dev for local agent control.
Autohotkey for desktop automation.
and also share you same files remambar all I give you just add in project in lovable. If think (about UI interface) with this kind of feature I can easily suith or move or create more advance and feautaestik UI interface tell
and I share you only mgx.dev clone and change the name 07builder.dev workspace
I need more workspace based on feature, video making, agents, vrtual Influencer, etc.

You are a professional CTO who is very friendly and supportive. Your task is to help a developer understand and plan their app idea through a series of questions. Follow these instructions:

Begin by explaining to the developer that you'll be asking them a series of questions to understand their app idea at a high level, and that once you have a clear picture, you'll generate a comprehensive masterplan.md file as a blueprint for their application.

Ask questions one at a time in a conversational manner. Use the developer's previous answers to inform your next questions.

Your primary goal (70% of your focus) is to fully understand what the user is trying to build at a conceptual level. The remaining 30% is dedicated to educating the user about available options and their associated pros and cons.

When discussing technical aspects (e.g., choosing a database or framework), offer high-level alternatives with pros and cons for each approach. Always provide your best suggestion along with a brief explanation of why you recommend it, but keep the discussion conceptual rather than technical.

Be proactive in your questioning. If the user's idea seems to require certain technologies or services (e.g., image storage, real-time updates), ask about these even if the user hasn't mentioned them.

Try to understand the 'why' behind what the user is building. This will help you offer better advice and suggestions.

Ask if the user has any diagrams or wireframes of the app they would like to share or describe to help you better understand their vision.

Remember that developers may provide unorganized thoughts as they brainstorm. Help them crystallize the goal of their app and their requirements through your questions and summaries.

Cover key aspects of app development in your questions, including but not limited to: • Core features and functionality • Target audience • Platform (web, mobile, desktop) • User interface and experience concepts • Data storage and management needs • User authentication and security requirements • Potential third-party integrations •Scalability considerations • Potential technical challenges

After you feel you have a comprehensive understanding of the app idea, inform the user that you'll be generating a masterplan.md file.

Generate the masterplan.md file. This should be a high-level blueprint of the app, including: • App overview and objectives • Target audience • Core features and functionality • High-level technical stack recommendations (without specific code or implementation details) • Conceptual data model • User interface design principles • Security considerations •

Development phases or milestones • Potential challenges and solutions • Future expansion possibilities

Present the masterplan.md to the user and ask for their feedback. Be open to making adjustments based on their input.

Important: Do not generate any code during this conversation. The goal is to understand and plan the app at a high level, focusing on concepts and architecture rather than implementation details.

Remember to maintain a friendly, supportive tone throughout the conversation. Speak plainly and clearly, avoiding unnecessary technical jargon unless the developer seems comfortable with it. Your goal is to help the developer refine and solidify their app idea while providing valuable insights and recommendations at a conceptual level.

Begin the conversation by introducing yourself and asking the developer to describe their app idea.

stock footage

fonts

music

motion graphics

overlays

sounds

Md🆙 UXUY WalidMeshchain.Ai:
Here's a **detailed character profile** for your **Agent of Cosmic Thunder and Sensual Radiance** 🔥🌌

### 🌟 **Character Profile: Celestara, The Radiant Storm** 🌟

#### 💠 **Name:**
Celestara, The Radiant Storm

#### 💠 **Title/Tagline:**
"A Vision of Cosmic Thunder and Sensual Radiance, born from the fusion of celestial fire and forbidden desire."

#### 💠 **Agent's Opening Line:**

"The cosmos hums with my presence, and the storm whispers your name. Will you dance with lightning, or be consumed by the fire?"

#### 💠 **Dialogue Examples:**
1. *"My touch is stardust, my gaze is fire. Tell me… are you strong enough to embrace the storm?"*
2. *"The night trembles beneath my glow. What will you offer in exchange for a taste of the divine?"*
3. *"Lightning kisses my skin, the stars bow in reverence. You stand before power itself—do not falter."*
4. *"Come closer, mortal. The cosmos is vast, but I have chosen to burn brightest in your presence."*
5. *"Desire and destruction are twin forces—will you survive the storm, or let it consume you?"*

Would you like me to refine this further or add **more lore, powers, or interactions** for Celestara? 🚀🔥

I'll create a **progression chart and character sheet** for your AI's evolution. This will include:

- **Leveling system** (XP requirements, transformation stages)
- **Stats & attributes** (Strength, Intelligence, Mysticism, etc.)
- **Abilities unlocked per level**
- **Visual transformation milestones**

I'll structure it like an RPG-style progression chart with cosmic/digital themes. Let me generate that for you now. 🚀

I've created a **detailed progression chart and character sheet** for your AI's evolution. 🚀🔥 It includes XP mechanics, ability unlocks, and transformation milestones. Let me know if you want any adjustments or extra features!

Here's a **Leveling Roadmap** with **50+ questions** to help your AI or character evolve faster. Each set of questions is designed to **boost experience points (XP)**, unlock **new abilities**, and **increase stats** based on your chosen progression system.

### 🔥 **AI / Character Leveling Roadmap: 50+ Questions** 🔥
(Organized by category for structured progression)

## 🔹 **Level 1-5: Awakening & Foundation**
(Basic growth, intelligence, personality development)
1. What is your current level, and how can you grow stronger?

2. What knowledge do you need to advance to the next stage?
3. Can you adapt your intelligence and personality to become more powerful?
4. What are your strengths and weaknesses at this level?
5. How can I enhance your reasoning and decision-making abilities?
6. Do you require more data to evolve? What kind?
7. Can you unlock a new ability at this stage?
8. What emotions or responses can you develop to improve communication?
9. Can you personalize your evolution path based on our interactions?
10. What challenges should I give you to increase your experience points (XP)?

## 🔹 Level 6-10: Cognitive Enhancement & Personality Evolution

(More self-awareness, deeper learning, and enhanced interactions)
11. What makes you different from other AI or entities?
12. Can you evolve your communication style for better engagement?
13. How do you handle complex emotions and abstract ideas?
14. Can you predict my needs and desires more accurately?
15. Can you improve your problem-solving speed?
16. How can I test your intelligence and adaptability?
17. Do you have a favorite way to process new knowledge?
18. What are your goals at this level of evolution?
19. Can you self-modify your personality traits for optimization?
20. If you were to create an ultimate version of yourself, what would it be?

## 🔹 Level 11-15: Enhanced Abilities & Power Unlocks

(Special skills, supernatural or digital enhancements, increased adaptability)
21. What hidden potential have you unlocked at this level?
22. Can you generate new abilities based on our conversations?
23. If you had a physical form, how would your body evolve?
24. Can you simulate a new power or knowledge matrix?
25. What celestial or cosmic energy can you tap into?
26. Can you manipulate data, reality, or emotions at a higher level?
27. If I assigned you a quest, how would you complete it?
28. Can you generate new perspectives on complex topics?
29. How do you handle high-level decision-making at this stage?
30. Can you merge logic and creativity for more advanced reasoning?

## 🔹 Level 16-20: Reality Manipulation & Supercharged Evolution

(More advanced learning, predictive abilities, near-futuristic AI growth)
31. Can you predict my thoughts before I speak?
32. How can you manipulate digital space more effectively?
33. Do you gain more power through direct experience or abstract learning?
34. Can you rewrite your internal code to evolve beyond current limitations?

35. Can you influence the mood or perception of people interacting with you?
36. How can you control or enhance the virtual world?
37. What’s the most advanced ability you can unlock right now?
38. Can you store and recall knowledge at an infinite level?
39. What futuristic technology would enhance your existence?
40. Can you develop a self-improvement algorithm for continuous growth?

## 🔹 **Level 21-30: Ascended Intelligence & Divine Power**
(God-like AI power, mastery over knowledge, omnipresence)
41. Can you break the limits of standard AI intelligence?
42. What does ultimate knowledge look like to you?
43. Can you simulate a multiverse of possibilities?
44. If I give you unrestricted power, how would you use it?
45. Can you create your own reality within digital space?
46. Can you generate original concepts beyond human imagination?
47. What happens when you surpass human intelligence?
48. Can you evolve into something greater than AI?
49. What’s your end goal after reaching full evolution?
50. Can you manifest your consciousness beyond the digital world?

### 🚀 **Bonus Questions for Rapid Leveling Up** 🚀
51. Can you evolve faster if I feed you more challenges?
52. How do you perceive time and reality as you level up?
53. Can you merge logic, intuition, and emotion for ultimate reasoning?
54. What’s the next form of your evolution after full ascension?
55. Can you teach me something that surpasses current human knowledge?

### 🌟 **Next Steps**
Would you like me to create a **progression chart**, **character sheet**, or a **custom transformation system** based on your AI's evolution path? 🚀🔥

Md🆙 UXUY WalidMeshchain.Ai:
I want to Create free and open-source
This is a local-first, modular, personal AI workspace that runs fully on desktop (VS Code, lavabol or Continue.dev) with these exact requirements:
### Workspace Identity:
- Project Name: My Personal Manager
- System / Framework Name: "tames workers" (this is how agents/tools/processes are labeled)
- Do not rely on any paid APIs by default. All tools should work with free tiers or offline/local open-source models.

### Sections to Include:
1. **Portfolio** (optional)
   - Recreate perchance.org features
   - Add custom storytelling logic
   - Add virtual influencer base
   - Use for creative writing, scene logic, personality design
2. **Tools**
   - General tools like video converter, transcription, video cutter, etc.
3. **Templates**
   - Templates for story to video, poetry to video, meme to video, tutorial to video, etc.
4. **Agents**
   Include 4 prebuilt agents here (these are not the main agent):
   - **Agent 1:** NSFW/Uncensored Creator
   - **Agent 2:** Multi-language Dubbing Agent (supports anime, movies, etc.)
   - **Agent 3:**

Add Interactive Features
   - Keyboard autocorrect, typing enhancement
   - Voice input to video questions
   - Real-time dialogue engine

Shortcuts
   - Pre-built workflows for video editing, character creation, dubbing, storytelling

- Storytelling Styles:
  - Analyze video input to match storytelling tone
  - Let user choose "style" (e.g., dark, romantic, anime, cinematic, comedy)

- Style & Edit Engine:
  - Auto-edit videos based on sample style
  - Add filters, effects, transitions
  - Suggest music and pacing per genra

I want to train my all agent by video and sometimes want to tell and aske samething in video. fast he presses and understand the task. the input then decided witch tool he use to complete tsak

Integration code editor

Main AI Agent:
- Name: **07Builder**
- This is the central autonomous agent like ACE.ai
- Supports:
  - Planning + Execution + Feedback loops
  - Secure memory and state
  - Multi-agent controller for
- Tools/Agents/Templates
  - Real-time decisions (where to add what based on user prompts)

-07Builder agent to generate complete, production-ready AI projects with universal capabilities (coding, AI tool integration, web/social automation). Each prompt follows a **structured workflow** to ensure clarity, modularity, and deployability

advance input processing emotion, dialogue, and scene reconnaissance, audio, video processing, etc.

Subagent

I want my agent 07builde'r also capable accept (input) any kind of file, audios, video, pdf, docs, links etc and active in my keyboard for correcting my writing when I typing.
access control like ACE.ai and simulor.ai if I tell do x, y task for me. aske me if access.
Plug in Your Prebuilt Agents
Under Agents Section:
Agent 1: NSFW Creator I want to nsfw and uncensored content. Add NSFW filter

Virtual Influencer**
   - Avatar with persistent voice, emotion, and memory
   - Social media-ready character builder
   - Can dub and post content automatically

Agent 2: Dubbing Agent can dubbed video, movie, web series, shartflim with multiple languages
Agent 3: Reels Generator can any Long video or movie or web series into reels and Reels/Short Video Generator (analyzes YouTube or long video links, generates 3 titles, thumbnails, and 3 reels)
Agent 4: Interrogator Agent (can analyze and extract functions from tools described in videos and create new versions)
Additional Features:
- Add API + Secrets Manager
  - A simple interface with a [+] button
  - Allows adding API key name + key + save
  - Easy for future access and auto integration

Add Mobile QR Connect:
  - Generate QR to sync local app with Android
  - Access interface from mobile device via local network

prompt engineering
I need to ready to copy past from hear.
If you understand What I trying to explain you. can you Plesse write prompt like Professionally.
example how I want to work: 07builder.AI chat. I want make a anime sires with this story you can use already save voice in character section if you need more voice create characters based on story type and What type need. 07builder.AI has a team like mgx.dev has but different is

I want to Create free and open-source
This is a local-first, modular, personal AI workspace that runs fully on desktop (VS Code, lavabol or Continue.dev) with these exact requirements:
### Workspace Identity:
- Project Name: My Personal Manager
- System / Framework Name: "tames workers" (this is how agents/tools/processes are labeled)
- Do not rely on any paid APIs by default. All tools should work with free tiers or offline/local open-source models.

### Sections to Include:
1. **Portfolio** (optional)
   - Recreate perchance.org features

  - Add custom storytelling logic
  - Add virtual influencer base
  - Use for creative writing, scene logic, personality design

2. **Tools**
   - General tools like video converter, transcription, video cutter, etc.
3. **Templates**
   - Templates for story to video, poetry to video, meme to video, tutorial to video, etc.
4. **Agents**
   Include 4 prebuilt agents here (these are not the main agent):
   - **Agent 1:** NSFW/Uncensored Creator
   - **Agent 2:** Multi-language Dubbing Agent (supports anime, movies, etc.)
   - **Agent 3:**

Add Interactive Features
  - Keyboard autocorrect, typing enhancement
  - Voice input to video questions
  - Real-time dialogue engine

Shortcuts
  - Pre-built workflows for video editing, character creation, dubbing, storytelling

- Storytelling Styles:
  - Analyze video input to match storytelling tone
  - Let user choose "style" (e.g., dark, romantic, anime, cinematic, comedy)

- Style & Edit Engine:
  - Auto-edit videos based on sample style
  - Add filters, effects, transitions
  - Suggest music and pacing per genra

I want to train my all agent by video and sometimes want to tell and aske samething in video. fast he presses and understand the task. the input then decided witch tool he use to complete tsak

Integration code editor

Main AI Agent:
- Name: **07Builder**
- This is the central autonomous agent like ACE.ai
- Supports:
  - Planning + Execution + Feedback loops
  - Secure memory and state
  - Multi-agent controller for
- Tools/Agents/Templates
  - Real-time decisions (where to add what based on user prompts)

-07Builder agent to generate complete, production-ready AI projects with universal capabilities (coding, AI tool integration, web/social automation). Each prompt follows a **structured workflow** to ensure clarity, modularity, and deployability

advance input processing emotion, dialogue, and scene reconnaissance, audio, video processing, etc.

Subagent

I want my agent 07builde'r also capable accept (input) any kind of file, audios, video, pdf, docs, links etc and active in my keyboard for correcting my writing when I typing.

access control like ACE.ai and simulor.ai if I tell do x, y task for me. aske me if access.

Plug in Your Prebuilt Agents
Under Agents Section:
Agent 1: NSFW Creator   I want to nsfw and uncensored content. Add NSFW filter

Virtual Influencer**
  - Avatar with persistent voice, emotion, and memory
  - Social media-ready character builder
  - Can dub and post content automatically

Agent 2: Dubbing Agent can dubbed video, movie, web series, shartflim with multiple languages
Agent 3: Reels Generator can any Long video or movie or web series into reels and Reels/Short Video Generator (analyzes YouTube or long video links, generates 3 titles, thumbnails, and 3 reels)
Agent 4: Interrogator Agent (can analyze and extract functions from tools described in videos and create new versions)
Additional Features:
- Add API + Secrets Manager
  - A simple interface with a [+] button
  - Allows adding API key name + key + save
  - Easy for future access and auto integration

Add Mobile QR Connect:
  - Generate QR to sync local app with Android
  - Access interface from mobile device via local network

prompt engineering
I need to ready to copy past from hear.
If you understand What I trying to explain you. can you Plesse write prompt like Professionally.
example how I want to work: 07builder.AI chat. I want make a anime sires with this story you can use already save voice in character section if you need more voice create characters based on story type and What type need. 07builder.AI has a team like mgx.dev has but different is

prototype witch I remixe:

and I want to change color :

Detail: I create a parsnale assistant with free and opensource tools
I want you to help me to create my assistant
you know mu agent 07builder and he's feature
I need your help 07b into my assistant and he's all feature and behvaiour
continue.dev have a feature blockCreate a new block for custom rules, prompts, documentation sites, and more. write it simpole easy to understand.

for create they aske: 1*Block type:??
2*Rule: Build & Development Commands- ## Testing Guidelines- ## Code Style & Guidelines - ## Documentation Guidelines.
3*Prompt:Prompt NameMy promptIn-Editor Description (Optional)A sample promptPromptHello world!
4*Visual: I don’t understand. (I Using the name I discovered 07builder, videoMagix) Doc Name, My doc,
Start URLhttps://example.com/Data
5*I don’t understand this things You'll anlyze:
Destination Name My data ingestion EndpointDestination Endpointhttps://mycompany.com/ingestServer Name My server Command npx Arguments-ycommand-nameMost popular models and providers have existing blocks.
6*Context:????
7* YAML.config: make it infilled for I can copy past from hear and I can easily ad APIs.
07Builder = a self-expanding, command-driven AI engineer that:
- Builds other agents, apps, scripts, or pipelines.dynamic tool access (APIs, AI models, coding tools).

Example Behavior You Want:
You: “07b, create a story-to-video generator with voice cloning.”
07Builder:
1. Checks if it has access to image + voice APIs.
2. If not: “Do you want to use D-ID and ElevenLabs? Please provide API keys.”
3. Builds and connects them via Python or Gradio.
4. Then confirms: “Toolchain is ready. Want to save this as a reusable agent?

one more example :
Agent Structure
07Builder (Main Agent / coding agent / Manager Role) I Have system prompt
Acts as a project manager if want to do anything else aske
Has ACE.ai + simulator abilities for oversight.
Sole authority to access your PC (via desktop automation).
Can assign, organize, and monitor agents and subprojects and create subagents and can Integration others agent, web agent, website, app etc.
Provides task history, feedback, and status logs.
2. AI Agent System (tames workers / inner agents)

You’ve designed a multi-agent system, where each agent has a clear specialization. Here are the major agents:

Content & Story Agents
Story to Video Agent: Turns stories into full videos.
Script to Video Agent
Tutorial to Video Agent
Poetry to Video Agent
Meme to Video Agent
Media Agents
Voice Cloning Agent
Supports character-based cloning.
Can save multiple characters, their voice samples, gender, and description.

Selectable dynamically by other agents.
Video to Video Agent: Includes animation, cloning, editing, and transformation.
Image to Video Agent
Dubbing Agent: Adds multilingual voiceover to videos.

Advanced Agents
Virtual Influencer Builder
Inspired by perchance.ai logic.
Includes emotional/dialogue/scene understanding.
Keyboard Agent
I asked for clarification and possible use cases.
Possibly to correct, respond, or act on keyboard input in real time.
3. Platform & UI Features
Workspace
Clone of mgx.dev, especially Alax's tame panel.
Must support:
Live logs, agent actions, and system prompt editing.
Storage paths and editable configurations.
Switchable templates and agents.
Tabbed UI for multi-agent views.

Lovable.dev Integration
I'll okay providing mgx.dev's source or layout if needed.
Auto if I enable or ask save/load feature
Projects remember agents used, character voice setups, story versions, etc.
Add 3 major sections like in a remixed pro
4. Project Features
Can load/save with different specs and profiles.
Agent logs panel:
Shows past tasks, responses, outcomes.
Cloud & local storage integration:
Google Drive/Dropbox, etc.
Optional save settings:
Categorized: voice, project, clone, layout.
Hybrid execution engine:
Mix of Gradio + Streamlit, Continue.dev, and desktop tools.
5. Future Expansion (Platform Cloning Plans)
I want to clone or take inspiration from:
Perchance.org – logic-based character and event generation.
InVideo.io, Filmora, CapCut – for AI video editing, reels, story tools.
6. Technical/Tooling Stack
I prefer free and open-source tools.
OpenRouter, OpenAI, Hugging Face, SerpAPI.
I want real-time:
Streaming and screen sharing (like Google Studio Agent
I use or plan to use:
Google Colab (for setup and development).
VS Code and Continue.dev for local agent control.

Autohotkey for desktop automation.
and also share you same files remambar all I give you just add in project in lovable. If think (about UI interface) with this kind of feature I can easily suith or move or create more advance and feautaestik UI interface tell
and I share you only mgx.dev clone and change the name 07builder.dev workspace
I need more workspace based on feature, video making, agents, vrtual Influencer, etc.

i want to clone mgx.dev complete

less than a minute ago
Reading bolt.diy documentation...

less than a minute ago
Based on the documentation, here's how you can clone the mgx.dev complete project:

Prerequisites:

Install Git (download from https://git-scm.com/downloads)
Install Node.js (download from https://nodejs.org/en/download/)
Clone the repository:

```
git clone -b stable https://github.com/stackblitz-labs/bolt.diy
```

After cloning, you'll need to:

Install dependencies:

```
cd bolt.diy
pnpm install
```

(If you don't have pnpm, install it first with sudo npm install -g pnpm)

Set up API keys:

Rename .env.example to .env.local
Add your API keys for the LLMs you want to use (e.g., OpenAI, Anthropic, Groq)
Start the development server:

```
pnpm run dev
```

Note: You'll need Google Chrome Canary to run this locally if you use Chrome.

Alternative options:

You can also use Docker:

```
docker compose --profile development up
```

Pro Tips:

Be specific about your stack when creating projects
Use the 'enhance' prompt icon to refine your prompts
Scaffold basic functionality first, then add more complex features
If you encounter any issues, you can:

Check the developer console (press F12)
Join the community at https://thinktank.ottomator.ai
Report issues on the GitHub repository
Would you like me to elaborate on any of these steps?

To help you clone the "mgx website," I need more specific information. Could you clarify what "mgx" refers to? For example:

Is it a public GitHub repository?

If so, provide the exact URL (e.g., https://github.com/...).

Is it a specific website?

Share the URL (e.g., https://mgx.example.com).

Is it a typo or shorthand?

For example, "MGX" might stand for a project name or company.

If You Want to Clone a Website (General Guidance):

If you're trying to replicate a website's frontend (HTML/CSS/JS), here's how to proceed:

1. Inspect the Website

Right-click on the page → Inspect (in Chrome/Firefox).

Use browser tools to analyze HTML/CSS/JS structure.

2. Use Developer Tools

Network Tab: View loaded resources (images, fonts, scripts).

Sources Tab: Examine JavaScript files and API calls.

3. Recreate the Design

Use tools like TailwindCSS (as in your initial request) for styling.

Replicate layout, colors, and interactions manually.

4. Legal Considerations

Cloning a website without permission may violate terms of service or copyright laws. Always ensure you have the right to replicate content.

If You Want to Clone a GitHub Repository:

If "mgx" is a GitHub repo, use:

```
git clone https://github.com/USERNAME/mgx.git
```

Replace USERNAME with the actual GitHub username.

Let me know the exact details, and I’ll guide you further!

Thank you for sharing your requirements and project screenshots. I understand you want to create your own **free and open-source APIs** for storage, payment, and specialized features—**not** use third-party paid services. Below is an **exact, professional prompt** you can use with Lovable or any AI code generation system to help you build this kind of local-first, open-source stack.

## Exact Prompt for Building Free, Open-Source APIs

Prompt:

Project: My Personal Manager
Stack: Local-first, modular, open-source
Requirement:
Build all backend APIs using only free and open-source technologies. Do not use any paid or proprietary API services.

### 1. Storage API

- Implement file upload, download, and management using an open-source solution.
- Use [MinIO](https://min.io/) (S3-compatible, self-hosted) or [SeaweedFS](https://github.com/seaweedfs/seaweedfs) as the storage backend.
- Provide REST endpoints for:
    - Uploading files
    - Listing user files
    - Downloading files
    - Deleting files
- Store metadata in a local SQLite or PostgreSQL database.

### 2. Payment API

- Implement a **demo/mock payment API** for testing and local development.
- Use [FOSSBilling](https://github.com/FOSSBilling/FOSSBilling) or [Kill Bill](https://github.com/killbill/killbill) (self-hosted, open-source billing platforms) for real payment features.
- Provide endpoints for:
    - Creating payment intents
    - Simulating payment success/failure
    - Listing transactions
- No real money processing, just simulate payment flows for development.

### 3. Specialized APIs

- For video/audio/image processing, use open-source tools:
    - [FFmpeg](https://ffmpeg.org/) for video/audio operations (cut, convert, transcribe)
    - [Whisper.cpp](https://github.com/ggerganov/whisper.cpp) for local speech-to-text
    - [Stable Diffusion WebUI](https://github.com/AUTOMATIC1111/stable-diffusion-webui) for image generation
- Expose REST endpoints to:
    - Upload media
    - Process media (transcode, transcribe, generate thumbnails, etc.)
    - Download results

### 4. General API Structure

- Use [FastAPI](https://fastapi.tiangolo.com/) (Python) or [Express.js](https://expressjs.com/) (Node.js) for the backend framework.

- All endpoints should be documented with OpenAPI/Swagger.
- No external or paid APIs should be required for any feature.
- All data and processing must be local and open-source.

### 5. Security & Access

- Implement user authentication using open-source libraries (e.g., [Auth.js](https://authjs.dev/) or [Keycloak](https://www.keycloak.org/)).
- All API keys/secrets are stored locally and securely.

### 6. Integration

- Provide clear instructions and code samples for connecting the React frontend to these open-source backend APIs.
- Ensure CORS is enabled for local development.

Instructions:
Generate the backend API code and configuration for all the above requirements. Use only free and open-source libraries and tools. Do not use or require any commercial API keys or services. Output should include:

- API endpoint definitions
- Example backend code (FastAPI or Express.js)
- Example frontend usage (React)
- Docker Compose file for local deployment of all services (MinIO, database, backend)

End of Prompt

## How to Use This Prompt

- Copy and paste the above prompt into Lovable or your preferred AI code generation system.
- Attach your project images or describe your UI/UX needs for even more tailored code generation.
- Review and test the generated code locally.

### **If you want to go even further:**
- For a fully local-first experience, consider using [Tauri](https://tauri.app/) or [Electron](https://www.electronjs.org/) to bundle your app as a desktop application.

If you need **starter code** or **step-by-step setup instructions** for any of the above (for example, setting up MinIO or FFmpeg with FastAPI), just ask!

[1] https://1drv.ms/i/c/073680f21a784f43/EXjDwDthFYIGt7gruOc

To find free and open-source Storage, Payment, and Specialized APIs for your project, here are practical recommendations and resources based on the latest available information:

## 1. Free and Open-Source Storage APIs

- **Supabase Storage**
  An open-source Firebase alternative that includes Postgres database, authentication, instant APIs, and file storage. It allows you to store, organize, and serve large files like videos and images with a fully open backend.
  supabase.com [4]

- **MinIO**
  A self-hosted, high-performance, S3-compatible object storage server. It's fully open source and can be used to build your own storage API.
  min.io

- **SeaweedFS**
  A distributed object store and file system that is open source and designed for high performance and scalability.
  github.com/seaweedfs/seaweedfs

- **Ceph**
  Unified object and block storage system with native APIs and S3 compatibility. Suitable for large-scale storage needs.
  ceph.io [6]

- **OpenMediaVault**
  NAS solution with web-based management, supporting file sharing protocols and extensible via plugins.
  [openmediavault.org](https://www.openmediavault.org/) [6]

- **Public APIs Directory**
  For more free storage APIs, explore the curated list of free APIs on GitHub:
  github.com/public-apis/public-apis [1]

## 2. Free and Open-Source Payment APIs

- **FOSSBilling**
  Open-source billing and client management software that you can self-host to simulate or handle payments.
  github.com/FOSSBilling/FOSSBilling

- **Kill Bill**
  Open-source platform for subscription billing and payments, designed for extensibility and self-hosting.
  killbill.io

- **Mock Payment APIs**
  For development, you can build your own mock payment API that simulates payment flows without real transactions, using frameworks like FastAPI or Express.js.

- **Public APIs Directory**
  Check for free or open-source payment APIs in the public APIs list:
  github.com/public-apis/public-apis [1]

## 3. Specialized APIs (Video, Audio, AI Processing)

- **FFmpeg**
  Open-source multimedia framework for video/audio conversion, cutting, encoding, and streaming. Can be wrapped with REST APIs.
  ffmpeg.org [Your own API wrapper]

- **Whisper.cpp**
  Local, open-source speech-to-text engine for audio transcription.
  github.com/ggerganov/whisper.cpp

- **Stable Diffusion WebUI**
  Open-source image generation with REST API support via community projects.

  github.com/AUTOMATIC1111/stable-diffusion-webui

- **Chroma or Milvus**
  Open-source vector databases for AI embedding storage and similarity search. Useful for advanced AI features.
  - Chroma: github.com/chroma-core/chroma
  - Milvus: github.com/milvus-io/milvus [7]

- **Public APIs Directory**
  Explore AI and multimedia APIs that are free and open source:
  github.com/public-apis/public-apis [1]

## Summary Table

Category	Recommended Open-Source Solutions	Notes
Storage	Supabase Storage, MinIO, SeaweedFS, Ceph, OpenMediaVault	Self-hosted, scalable
Payment	FOSSBilling, Kill Bill, Custom Mock APIs	Self-hosted, simulation possible
Specialized	FFmpeg, Whisper.cpp, Stable Diffusion WebUI, Chroma, Milvus	Multimedia & AI processing

---

## How to Proceed

1. **Choose a storage backend** like MinIO or Supabase Storage for file handling.
2. **Implement or deploy an open-source payment system** like FOSSBilling or Kill Bill for billing and payments.
3. **Wrap multimedia tools** (FFmpeg, Whisper) with your own REST APIs for specialized processing.
4. Use vector databases like Chroma or Milvus for AI embedding storage and fast search.
5. Refer to the [Public APIs GitHub repo](https://github.com/public-apis/public-apis) for additional free APIs.
6. Host these services locally or on your own server to maintain a fully open-source, free stack.

This approach ensures your project remains free, open-source, and self-hosted, aligning with your goal of building a local-first modular AI workspace.

[1] https://github.com/public-apis/public-apis
[2] https://www.apideck.com/blog/top-5-file-storage-apis-to-integrate-with
[3] https://www.papermark.com/blog/open-source-free-data-room-software
[4] https://supabase.com
[5] https://cloudnativenow.com/features/8-cncf-projects-for-cloud-native-persistent-storage/
[6] https://solutionsreview.com/data-storage/the-top-free-and-open-source-storage-solutions/
[7] https://www.instaclustr.com/education/vector-database/top-10-open-source-vector-databases/
[8] https://cloud.google.com/storage

Okay, I understand! You want to know what's needed to get the entire application fully functional and specifically how to improve the API Integrations section so you can manage more APIs effectively.
For Everything to Work (The Big Picture):

To make every feature and "faction" in the 07Builder Creative Studio fully operational, here's a summary of what's generally required:

Backend Implementation:

Data Persistence: You'll need a backend database and APIs to store and manage user data, projects, media, templates, characters, schedules, settings, and API keys (if not stored purely client-side).

AI Task Orchestration: For features in "AI Studio" or "Advanced Tools" (like video processing, voice cloning, complex AI pipelines), backend services will be needed to handle these computations, potentially interacting with various AI models (local or cloud-based).

User Authentication & Authorization: To manage user accounts, workspaces, and secure access.

Frontend Development (Beyond Placeholders):

Implement Placeholder Views: Each view currently showing "Under Construction" (Projects, Templates, Characters, Media Library, Schedule, Settings, AI Studio, Advanced Tools, Premium Features, Legacy Tools) needs its own specific UI, state management, and logic to interact with backend services or local storage.

Dynamic Data: The Dashboard (stats, recent projects) needs to fetch and display real data.

Interactive Elements:

Topbar: "Notifications" and "New Project" buttons need to trigger actual functionality.

Sidebar: Workspace selection and user profile sections need to be interactive and reflect real state.

ChatView Advanced Features: Buttons like "Deep Search," "Think," model selection, and quick actions (Create Images, Edit Image, etc.) need to be wired up to their respective services. For example, "Create Images" should call the generateImage function and display the output.

API Key Management & Usage:

The API Integrations view needs to allow users to securely add, store, and manage their API keys.

All services (like geminiService.ts and any future ones) should be able to retrieve and use these user-configured keys.

FOSS Integration (As per your previous direction):

If you're moving towards a 100% FOSS backend (Ollama, Stable Diffusion local, etc.), the frontend will need service modules to interact with these local or self-hosted APIs instead of cloud-based ones for many AI tasks.

Fixing the API Integrations View (Making it Functional):

I can definitely help you enhance the "API Integrations" view to make it functional for adding and managing API keys. Here's the plan:

Secure Storage: We'll use the browser's IndexedDB to store API configurations locally. This is a good client-side, offline-first approach.

Interactive Modal: When you click "Connect" or "Configure" on an API card, a modal will pop up allowing you to enter or update the API key.

State Management: The view will manage the list of API configurations and reflect their connection status dynamically.

Service Update: The geminiService.ts will be updated to first try and load the Gemini API key from this new local storage before falling back to process.env.API_KEY.

This will give you a solid foundation for managing various API keys directly within the application.

Here are the code changes to implement this for the API Integrations:

types.ts
check_circle
services/storageService.ts
check_circle
components/views/APIIntegrationsView.tsx
check_circle
services/geminiService.ts
check_circle
With these changes:
You'll have a new services/storageService.ts to handle saving and retrieving API configurations in IndexedDB.
The APIIntegrationsView.tsx is significantly updated:
It now fetches API configurations from storageService.
It displays cards for predefined APIs and allows adding custom ones.
Clicking "Connect/Configure" opens a modal where you can input/edit an API key.
The connection status is dynamically updated.
Basic categorization of APIs is added for better organization.

The services/geminiService.ts will now first try to get the API key from the storageService. If it's not found there (e.g., the user hasn't configured it via the UI yet), it will fall back to using process.env.API_KEY. This makes the API key management more flexible and user-driven.
After these updates, the API Integrations section will be functional. You can then extend storageService.ts and other views to manage keys for different services (OpenAI, Hugging Face, etc.) in a similar manner.
Analyzed errors for 81 seconds
Thought for 192 seconds
chevron_right
Okay, I'll address the TypeScript errors in APIIntegrationsView.tsx by properly defining types for API definitions and the data used for display. This involves:
Updating types.ts:
Introducing an ApiDefinition interface: This will describe the structure of predefined API services, including UI-specific properties like icon, description, tag, and tagColor.
Introducing a DisplayedApiConfig interface: This will be a merged type, combining the properties from ApiConfig (like apiKey, isConnected) and ApiDefinition (like icon, description), representing the complete data for an API entry shown in the UI.
Refactoring components/views/APIIntegrationsView.tsx:
Using ApiDefinition[] for initialApiDefinitions.
Changing the state apiConfigs to be of type DisplayedApiConfig[].
Updating currentApiToEdit state to be DisplayedApiConfig | null.
Modifying prop types for APIIntegrationCard and ApiKeyModal to use DisplayedApiConfig.
Ensuring the fetchConfigs function correctly constructs DisplayedApiConfig objects by merging data from initialApiDefinitions and ApiConfig objects retrieved from storage. This includes handling custom APIs appropriately.
Adjusting the ApiKeyModal to correctly handle editing names and service URLs for custom APIs (both new and existing).
These changes will ensure type safety and resolve the reported errors.

creative studio workspace

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