Tech Implemenation

1. HIGH-LEVEL ARCHITECTURE: THE "HYBRID CLOUD"

We are utilizing a **Hybrid approach** where **N8N** acts as the API Gateway/Orchestrator, but all heavy computation and storage are offloaded to **AWS** and specialized APIs.

The 4-Layer Stack

Layer	Component	Technology Stack	Function
1. Interaction	**Frontend / UI**	React / Next.js	User Dashboard, Approval Interface, Upload Portal.
2. Orchestration	**The Controller**	N8N (Self-Hosted)	Traffic control, Webhook listening (Reviews), API routing.
3. Intelligence	**The Brain (RAG)**	OpenAI (GPT-4o) + Pinecone	Scriptwriting, Semantic Search, Brand Voice Memory.
4. Factory	**The Engine**	AWS Lambda + S3 + Shotstack	File storage, Computer Vision, Beat-Sync, Rendering.

---

2. CORE COMPONENT DEEP DIVE

A. The "Data Vault" (Storage & Ingestion)

Design Pattern: Asynchronous S3 Trigger

We do not pass video files through the application server. We use a **Direct-to-Cloud** pattern.

1. **Architecture:** Amazon S3 (Simple Storage Service) configured with Transfer Acceleration.
2. **Security:** All uploads use **Presigned URLs**. The frontend requests a temporary "Key" to upload directly to AWS. This keeps our servers lightweight.
3. **Event Loop:**
   - Event: File lands in s3://raw-uploads.
   - Trigger: AWS Lambda function fires automatically.
   - Action: Calls **Banana.dev / Google Gemini Vision** to analyze the footage.
   - Output: Metadata tags [Sunset, Oysters, Luxury, 4K] are sent to the Vector Database.

B. The "Nexus Brain" (RAG - Retrieval Augmented Generation)

Design Pattern: Vector Semantic Search

Standard AI (ChatGPT) hallucinates because it lacks context. We implement **RAG** to ground the AI in reality.

1. **Vector Database (Pinecone):** We convert every asset tag and every client brand document into "Embeddings" (numerical representations of meaning).
2. **The Query Logic:**
   - Input: "Make a video for a luxury foodie couple."

- Vector Search: The system queries Pinecone for vectors matching "Luxury" + "Food" + "Couples" + "High Resolution."
- Retrieval: It pulls the exact File IDs for the best footage from the Data Vault.

3. **Script Generation:** GPT-4o receives the Context (the selected file descriptions) and generates the JSON script.

C. The "Video Factory" (Rendering Engine)

Design Pattern: Serverless Media Processing

This is the core engineering challenge. We use **Shotstack** driven by code.

1. **Audio Sync Engine (Python):**
   - We run a Python microservice using the librosa library.
   - It analyzes the selected music track to extract **Onsets** (Beats) and **Energy Levels**.
   - It mathematically calculates the cut points (e.g., Cut at 3.4s, 5.8s, 9.2s) to ensure the video is rhythmically satisfying.
2. **The Assembly (JSON):**
   - The Logic Layer compiles a JSON payload containing: S3 URLs for video, HTML/CSS for text overlays (Reviews), and the Cut Points.
   - This is sent to the Shotstack API for cloud rendering.

D. The Fallback Protocol (Nano Banana / Gemini)

Design Pattern: Conditional Logic

1. **Confidence Check:** If the Vector Search returns a match score < 0.3 (Missing Footage), the system flags a "Gap."
2. **Generative Pipeline:**

- **Step 1:** Call **Nano Banana Pro (Gemini 3 Image)** to generate a photorealistic static image.
- **Step 2:** Pipe the image to **Runway/Luma API** (Image-to-Video) to add 2 seconds of ambient motion (steam, light leaks, water movement).
- **Step 3:** Inject the resulting URL into the Shotstack timeline.

---

3. DATA FLOW DIAGRAMS

Pipeline 1: The "Smart Ingestion" (Building the Moat)

> Goal: Turn raw files into searchable data without human tagging.
> User Upload -> AWS S3 Bucket -> (Trigger) -> AWS Lambda (Computer Vision) -> Extract Tags -> Store in Pinecone (Vector DB)

Pipeline 2: The "Review-to-Revenue" Engine

> Goal: Autonomous Video Creation.

1. **Trigger:** Google Maps Webhook (New 5-Star Review).
2. **Analysis:** N8N parses text -> GPT-4o extracts Keywords & Sentiment.
3. **Retrieval:** Pinecone searches for matching Video Assets + Cross-Pollination Partner Assets.

4. **Fallback:** If asset missing -> Trigger Generative AI Pipeline.
5. **Logic:** Python calculates Music Beat Sync.
6. **Assembly:** N8N constructs Shotstack JSON.
7. **Render:** Shotstack processes 4K video -> Saves to s3://rendered-output.
8. **Notify:** User receives SMS/Email with Approval Link.

Pipeline 3: Cross-Pollination Logic (Relational Filtering)

> Goal: Promoting the right partner.
> We query the **PostgreSQL** relational database before the Vector search.
> Query: Partner Business
> WHERE distance < 2 miles
> AND category = 'Restaurant'
> AND tier != 'Budget' (If Client is Luxury)
> RETURN Top 3 Partner IDs.

---

4. DATABASE SCHEMA DESIGN

We utilize a **Dual-Database Strategy**:

A. Relational DB (PostgreSQL) - Structured Data

- **Table: Users** (Auth, Credits, Role)
- **Table: Businesses** (Name, Location, Brand_Tier, Google_Place_ID)
- **Table: Assets** (S3_URL, File_Type, Owner_ID, License_Status)
- **Table: Relations** (Which businesses are allowed to cross-pollinate)

B. Vector DB (Pinecone) - Unstructured Intelligence

- **Index: Brand_Voice** (Embeddings of previous successful captions/scripts)
- **Index: Asset_Library** (Embeddings of visual content descriptions)

---

5. SECURITY & INFRASTRUCTURE

Multi-Tenancy (Data Isolation)

To address your concern about confidentiality, we implement **Row Level Security (RLS)** in the database.

- **Rule:** A client can only query assets where Owner_ID == Self OR Asset_Type == Public_B_Roll.
- **Logic:** The API enforces this filter on every request. Even if the AI wants to use a competitor's clip, the database layer will block access before it happens.

Scalability

- **Statelessness:** Our N8N and Python workers are stateless.
- **Load Handling:** If 500 reviews come in at once, they are added to a **Message Queue (AWS SQS)**. The rendering engine processes them in order (FIFO), ensuring the server never crashes, no matter the volume.

Here is the architecture diagram: Tech architecure

Summary

This architecture moves Viral Vacation from a "Tool" to a **"Platform."** By leveraging **S3 for heavy storage**, **Vector DBs for intelligence**, and **Shotstack for rendering**, we create a system that is robust, secure, and infinitely scalable.

Chandan & Tushar

Revision #2
Created 2026-01-14 09:12:55 UTC by Chandan Kumar
Updated 2026-01-14 09:17:20 UTC by Chandan Kumar