**Master Prompt – AIVidCast: AI Podcast Generator**

**Goal:**  
Build a fully automated, AI-powered **video podcast generation and publishing system** with a web interface. The system allows authenticated users to:

1. Upload a sample speaker video (for voice, posture, and gesture cloning)
2. Enter **either** written podcast script **or just a topic**
3. Generate a full video podcast mimicking the speaker using **free/open-source AI tools**
4. Automatically publish the final podcast video to YouTube
5. Include full user authentication with **signup, login, and admin controls**

**🧩 Functional Requirements**

**🔐 User Authentication & Access**

* Users must **sign up and log in** to access podcast generation tools
* User data (username, email, hashed password, active status) stored in **YAML or SQLite**
* Product Owner (Admin) has:
  + **Admin interface** to view users
  + **Enable/disable users**
  + **Reset passwords**
  + **Add/remove users**

**🧠 AI Podcast Generation (2 Modes)**

1. **Manual Mode**
   * User enters their own written script
   * System converts script → voice → animated video podcast
2. **Topic-Only Mode**
   * User provides only a topic
   * System uses **LLM agent** to:
     1. Research the topic (DuckDuckGo/Wikipedia)
     2. Generate podcast script (using local LLM like LLaMA 3)
     3. Pass script through rest of the pipeline (TTS → video → publish)

**🧰 AI Tools & Pipeline (Only Free/Open Source)**

| **Step** | **Tool** |
| --- | --- |
| 🔎 Topic → Script | Local LLM (llama3, GPT4All, Mistral) via Ollama |
| 🗣 Script → Audio | TTS with Coqui, Bark, or Tortoise TTS |
| 🧑‍🎤 Audio + Avatar → Video | SadTalker (for talking head) |
| 👄 (Optional) Lip sync refinement | Wav2Lip |
| 🎞 Video editing | ffmpeg, moviepy, opencv-python |
| 📤 Auto YouTube Upload | YouTube Data API v3 using OAuth2 |

**🌍 Web Interface (Using Streamlit)**

* Upload speaker sample video (only once per user)
* Enter podcast **script or topic**
* Choose mode (Script / Topic)
* Click to generate → preview → publish to YouTube
* Track status/progress of job

**🔧 Admin Control Features**

* Admin logs in with special credentials
* Can view all registered users
* Can:
  + Enable/disable any user
  + Reset user passwords
  + Add/remove users manually
* Admin UI integrated into Streamlit

**🗃 Storage & Integration**

| **Component** | **Notes** |
| --- | --- |
| Speaker samples | Stored per user (video/image) |
| Audio files | Intermediate .wav from TTS |
| Videos | Final .mp4 output |
| Model checkpoints | Pre-loaded SadTalker + TTS models |
| DB/File storage | YAML (for simple user mgmt) or SQLite for scalable DB |
| Hosting | Dockerized deployment, local or on Render/Streamlit Cloud |

**🚀 Deployment/Execution Features**

* Fully containerized with **Docker**
* One-click install for dependencies + models
* Local or cloud deployment (Render, Railway, etc.)
* Modular components:
  + Agent-based orchestration
  + Pluggable TTS and video modules
  + Configurable model backend (Llama3, Coqui, SadTalker)

**📌 Optional Future Enhancements (Phase 2+)**

* Full-body video generation (via AnimateDiff)
* Custom backgrounds, branding overlays
* User analytics (video views, publish history)
* Email notifications on video readiness
* Scheduled YouTube posting

# 🎙️ PramaVidCast – AI-Driven Video Podcast Generator

\*\*PramaVidCast\*\* is an AI-powered, fully automated system that allows authenticated users to generate video podcasts using only a written script or topic. It uses free and open-source AI tools to mimic a speaker’s voice, posture, and gestures, and automatically publishes the final video to YouTube.

---

## 🚀 Project Objectives

- Clone a speaker’s \*\*voice, gestures, and posture\*\* using uploaded video

- Accept user input as either:

- ✍️ A \*\*written script\*\*

- 🧠 A \*\*topic\*\*, which is researched and turned into a podcast script using an LLM agent

- Generate full podcast videos using open-source AI tools

- Automatically \*\*upload the podcast to YouTube\*\*

- Include \*\*secure user authentication\*\* with admin-level user control

---

## 🔐 User Management

### ✅ Features

- \*\*Signup and Login\*\* for users

- Passwords securely hashed and stored

- \*\*Admin interface\*\* for product owner:

- View, add, or remove users

- Enable/disable user access

- Reset passwords

### 🧾 User Storage Options

- Simple: YAML file (for fast prototyping)

- Scalable: SQLite or PostgreSQL (for production)

---

## 🧠 AI Podcast Generation Workflow

### 👤 Step 1: Speaker Modeling

- User uploads sample video of the speaker

- Extracts facial image for pose/gesture reference

- Stores voice/visual identity for reuse

### ✍️ Step 2: Script Input

- Option A: User enters their own \*\*written script\*\*

- Option B: User enters a \*\*topic\*\*

- System uses an LLM agent (LLaMA3 via Ollama or GPT4All) to:

- Perform research (DuckDuckGo/Wikipedia)

- Summarize findings

- Generate a 1–2 minute podcast script

### 🗣 Step 3: Text-to-Speech (TTS)

- Uses \*\*open-source tools only\*\* like:

- `Coqui TTS`

- `Bark`

- `Tortoise TTS`

- Outputs high-quality `.wav` audio

### 🧑‍🎤 Step 4: Talking Head Video Generation

- Uses `SadTalker` with the extracted speaker image and TTS audio

- Optionally uses `Wav2Lip` for enhanced lip sync

- Produces a full animated talking-head `.mp4` video

### 🎞️ Step 5: Video Assembly

- Uses `ffmpeg` or `moviepy` to:

- Merge audio and video

- Add background or overlay (future)

- Finalize video podcast file

### 📤 Step 6: YouTube Publishing

- Uses \*\*YouTube Data API v3\*\*

- Automatically uploads the video

- Adds title, description, and tags

- Optional: Schedule for future release

---

## 🌍 Web Interface

Built using \*\*Streamlit\*\*, the user-facing UI includes:

- Signup/Login

- Upload speaker video

- Text or topic input

- Mode selection (Script / Topic)

- Generation status and progress

- Final video preview and YouTube publish button

---

## ⚙️ System Architecture

```text

User Interface (Streamlit)

┌────────────────────────────────────┐

│ Upload Speaker │ Enter Script/Topic│

└────────────────┬───────────────────┘

│

┌────────▼─────────┐

│ AI Podcast Agent│

└────────┬─────────┘

│

┌────────────────▼────────────────────┐

│ LLM + TTS + SadTalker + Video Tools│

└────────────────┬────────────────────┘

│

┌────────▼────────┐

│ YouTube Uploader│

| **Function** | **Tool** |
| --- | --- |
| LLM Agent | LLaMA3 via Ollama, GPT4All |
| Research | DuckDuckGo Search API |
| Script Generation | LangChain or Custom Agent |
| TTS | Coqui TTS, Bark, Tortoise TTS |
| Face & Gesture Clone | SadTalker |
| Lip Sync (Optional) | Wav2Lip |
| Audio/Video Merge | ffmpeg, moviepy |
| YouTube Upload | YouTube Data API v3 |
| Web UI | Streamlit |
| Auth System | streamlit-authenticator |
| User Store | YAML (initial), SQLite (optional) |

**🐳 Docker-Based Deployment**

A complete Dockerfile will allow local or cloud deployment with:

* Preinstalled dependencies
* Downloaded SadTalker models
* Automatic launching of the app via Streamlit

**✅ Key Features Recap**

* Upload speaker video
* Script-based or topic-based generation
* Voice + facial gesture cloning
* AI-generated podcast script from topic
* Free & open-source TTS and animation
* Automated video generation and stitching
* YouTube auto-publishing
* Secure user authentication system
* Admin dashboard to manage users

**📌 Optional Future Enhancements**

* Full-body animated avatars (AnimateDiff)
* Background replacement or slides overlay
* Video branding and watermarking
* Analytics dashboard for admins
* Email notifications for video readiness
* Scheduled or batched YouTube uploads

**🛠️ Next Steps**

This project will be broken down into modules:

1. Authentication system with admin UI
2. Streamlit UI for input (speaker + script/topic)
3. Agent for LLM-based script generation
4. TTS generation module
5. SadTalker video module
6. Video merging and editing
7. YouTube publishing
8. Docker packaging and deployment

**📬 Contact**

Built by **Pramod Singh**  
PhD in Machine Learning, Architect @ Prama AI

Want to contribute or deploy it? Let’s collaborate on **PramaVidCast**!

yaml

CopyEdit

---

Let me know if you'd like this saved as a `.md` file or want to start implementing the first component (e.g., `Streamlit` auth UI or LLM agent).

Ask ChatGPT