Avatar Lab

Building Smart & Realistic AI Avatars

G331

Nikhilesh Nilagiri

Key Learnings & Outcomes

- Built a multi-step AI video generation pipeline end-to-end
- Integrated ML models (Lina Speech, DiffDub, RVM) with a web application
- Learned how to manage async video generation using background queues
- Handled media uploads and previews using GridFS

Tech Stack & Reasoning

Front End

Next.js : Server Side Rendering

Tailwind CSS : Styling

ShadCN : UI Components

Back End

Flask : Model API

BullMQ : Video processing queue

REST APIs : Integration Layer

Models

Small-E : Text-To-Speech

DiffDub : Talking Head Generation

RVM : Back Ground Editing

DataBase

MongoDB : Client Data (User)

Postgres SQL : Pre-Defined Data

Redis : Job data

Challenges & How We Solved Them

Challenge

- ❌ Decoupled ML tasks to prevent UI lag
- ❌ User lost progress if page refreshed
- ❌ Large video files – slow I/O & retrieval
- ❌ Incomplete video generation on errors

Solution

- ✅ Using BullMQ + Redis background queue
- ✅ LocalStorage to track job IDs and status
- ✅ Implemented MongoDB GridFS
- ✅ Failure checks + Error logging

Model Choices – Why Small-E & DiffDub

TTS Model	Voice	Speed	Fine-tuning	In AvatarLab
Small-E	✅ Natural	✅ Fast	✅ Yes	✅ Yes
Vall-E	✅ HQ	❌ Slow	✅ Yes	❌ No
Bark TTS	✅ Expressive	❌ Slow	❌ No	❌ No
Google TTS	✅ Studio	✅ Cloud	❌ No	❌ No

Talking Head	Sync Quality	Speed	Realism	In AvatarLab
DiffDub	✅ Excellent	✅ Moderate	✅ High	✅ Yes
Wav2Lip	❌ Robotic	✅ Fast	❌ Low	❌ No
SadTalker	✅ Image only	❌ Slow	✅ Okay	❌ No
Ditto	❌ Poor sync	❌ Slow	✅ Decent	❌ No

Deployment Decisions

	Deployment Platform	Notes
Frontend	Vercel	Free, optimized for Next.js
Backend API	**Railway** or **Render** (Free tier)	Hosts Flask + Models (limited runtime)
Model Execution	Docker on Railway/Render	Simple container-based deployment

THANK YOU