Avatar Lab G331 Prasanna Kumar Kota

Key Learnings & Outcomes

01.

Built a multi-step AI video generation pipeline end-toend Integrated ML

02.

models (Lina Speech, DiffDub, RVM) with a web application

03.

Learned how to manage async video generation using background queues

04.

Handled media uploads and previews using GridFS



Front End



Next.js : Server Side

Rendering Tailwind CSS:

Styling ShadCN: UI

Components

Backend



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



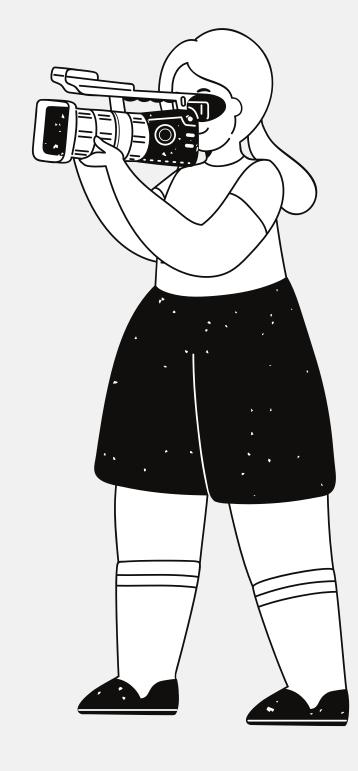
MongoDB: Client Data

(User)

Postgres SQL : Pre-

Defined Data Redis: Job

data







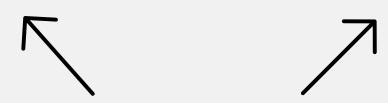
Challenge

Decoupled ML tasks to prevent UI lag

User lost progress if page refreshed

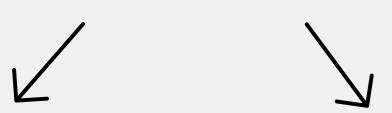
Large video files – slow I/O & retrieval

Incomplete video generation on errors



Challenges

& How we solved them



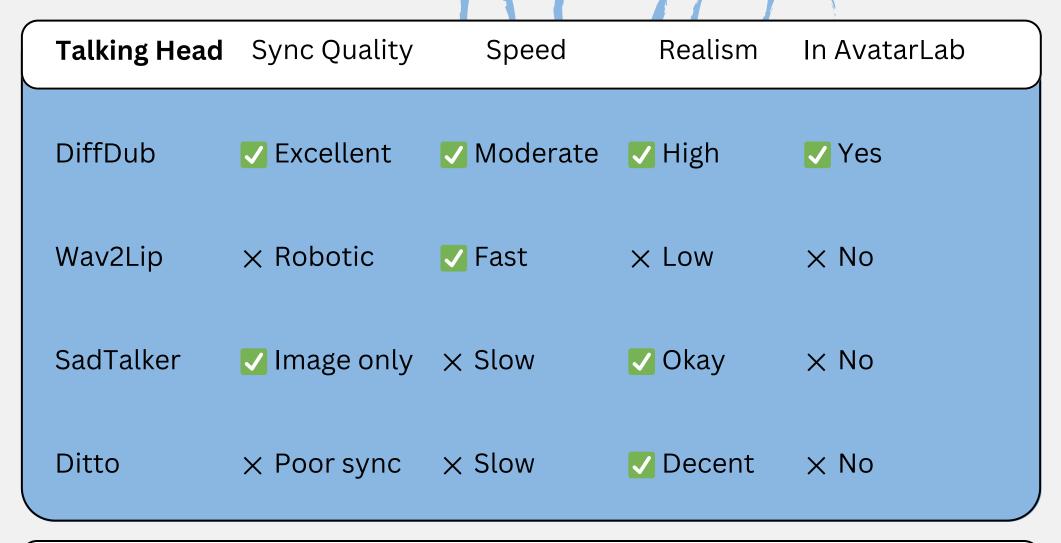
Solution

Using BullMQ + Redis background queue

LocalStorage to track job IDs and status

Implemented MongoDB GridFS

Failure checks + Error logging



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

Model Choices - Why Small-E & DiffDub



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	

Deployment Decisions

Frontend

On Vercel since its free and optimized for NextJS

Backend API

On modal.com
which provides
free limited GPU
for runnign
inference

Model Execution

A Docker image so that the requirements for models can be met

Thank you very much!

Avatar Lab G331